ALTINBAŞ UNIVERSITY

Electrical and Computer Engineering

ENHANCED SECURE COMMUNICATION SCHEMES FOR MACHINE-TO-MACHINE AND VEHICULAR AD HOC NETWORKS

Uğur CORUH

Ph.D. Dissertation

Prof. Dr. Oğuz BAYAT

Istanbul, 2019

ENHANCED SECURE COMMUNICATION SCHEMES FOR MACHINE-TO-MACHINE AND VEHICULAR AD HOC NETWORKS

by

Uğur Coruh

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

ALTINBAŞ UNIVERSITY

2019

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

____

Prof. Dr. Oğuz BAYAT

Supervisor

Examining Committee Members (first name belongs to the chairperson of the jury and the second name belongs to supervisor)

Prof. Dr. Oğuz BAYAT	School of Engineering and Natural Science, Altınbaş University	________________
Asst. Prof. Dr. Çağatay AYDIN	School of Engineering and Natural Science, Altınbaş University	________________
Asst. Prof. Dr. Doğu Çağdaş ATİLLA	School of Engineering and Natural Science, Altınbaş University	________________
Prof. Dr. Hasan Hüseyin BALIK	Air Force Academy, National Defense University	________________
Asst. Prof. Dr. Adil Deniz DURU	Faculty of Sport Sciences, Marmara University	________________

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

____

Asst. Prof. Dr. Çağatay AYDIN

Head of Department

Approval Date of Graduate School of
Science and Engineering: 13 /06/ 2018

____

Prof. Dr. Oğuz BAYAT

Director

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Uğur CORUH

DEDICATION

To my beloved parents, and all my family for their love, endless support, encouragement, and sacrifices.

ACKNOWLEDGEMENTS

In the name of Allah, the Beneficent and Merciful. Praise and Gratitude be to Allah for giving me strength and guidance, so that this thesis can be finished accordingly.

I would like to take this possibility to express my heartfelt gratitude to all those who helped me to make my thesis work a success. First, I would like to thank my esteemed mentor and advisor Prof. Dr. Oğuz BAYAT who has provided me the strength to do justice to my work and contribute my best to it so that it has turned out to be a successful venue.

My deepest gratitude goes to my dearest parents, grandparents, and my sister, for their support and encouragement throughout my life. And I present my deepest respects and love to my wife for her immense patience and unconditional support on this journey.

I want to thank all the Altınbaş University, college of engineering, department of the computer.

As a final word, I would like to thank each who has been a source of support and encouragement and helped me to achieve my goal and complete my dissertation work successfully

ABSTRACT

ENHANCED SECURE COMMUNICATION SCHEMES FOR MACHINE-TO-MACHINE AND VEHICULAR AD HOC NETWORKS

Coruh, Uğur,

PhD, Electrical and Computer Engineering, Altınbaş University,

Supervisor: Prof. Dr. Oğuz BAYAT

Date: 06/2019

Pages: 171

In this thesis, we examined several Machine-to-Machine (M2M) and Vehicular Ad Hoc Networks (VANETs) authentication and revocation schemes. According to our analyses, we introduce new schemes without flaws of examined designs. A password-based authentication for mutual authentication is suggested in the first analyzed M2M communication system. Furthermore, their suggested secure channel setup protocol utilizes symmetrical encryption and single way hash algorithms and considers that portable consumer devices or smart home networks are using their secure channel architecture. We propose that the missing part of the current M2M secure communication system be completed. An improvement of the system can be made by protecting privacy and altering the messages. Elliptic Curve Diffie Hellman (ECDH) cryptography based secure key sharing scheme, introduced in both initial setup and key-injection stage to provide safe client enrollment, device key change, and home gateway network's connection stages. The second studied VANET secure authentication and revocation system partly replaces Certificate Revocation Lists (CRLs) monitoring technique with the Hash-based Message Authentication Code (H-MAC) cryptogram verification. Public Key Infrastructure (PKI) and CRLs used communication systems have critical delays during the monitoring of CRLs. This latency has an essential impact on VANETs taken by PKI. We evaluated the

VANETs system and resolved common open problems. We also conducted missing components of comparable systems with efficiency improvements. These can occur through revocation key sender verification and revocation version validation to protect sensitive assets from invalid updates, address privacy preservation with keyed trimmed H-MAC based pseudo ID creation, set message identity to perform sensitive information transmission, lastly extract, and integrate applications to execute high-speed revocation. Furthermore, our reforms prepared for system reliability, durability, and MITM, substitution, false message, message modification, user manipulation, and successive response attacks resistance with anomaly detection. For M2M scheme, we simulated both proposed and analyzed plans for performance, network congestion, and resource usage. Also for VANETs scheme, we simulated three cases the standard, the proposed and analyzed models. We analyzed schemes for network congestion and performance. According to our simulation results, we proposed efficient schemes for both M2M and VANETS.

Keywords: M2M Security, VANET Security, Revocation Scheme, Authentication Scheme, IoT Security

TABLE OF CONTENTS

Pages

LIST OF TABLES

<u>Pages</u>

LIST OF FIGURES

Pages

LIST OF ABBREVIATIONS

IoT	:	Internet-of-Things
IoV	:	Internet-of-Vehicular
M2M	:	Machine-to-Machine
VANET	:	Vehicular Ad Hoc Network
MITM	:	Man-In-The-Middle
DoS	:	Denial-of-Service
ECDSA	:	Elliptic Curve Digital Signature Algorithm
ECDH	:	Elliptic Curve Diffie Hellman
ECDLP	:	Elliptic curve discrete logarithm problem
ECIES	:	Elliptic Curve Integrated Encryption Scheme
IBC	:	Identity-Based Cryptography
PID	:	Pseudo Identification Data
PKI	:	Public Key Infrastructure
CRL	:	Certificate Revocation List
H-MAC	:	Hash-based Message Authentication Code
RSU	:	Road-Side-Unit
R-TA	:	Root-Trusted-Authority
TA	:	Trusted Authority
OBU	:	On-Board-Unit

V2V	:	Vehicle-to-Vehicle
V2I	:	Vehicle-to-Infrastructure
V2I2V	:	Vehicle-to-Infrastructure-to-Vehicle
WAVE	:	The Wireless Access in Vehicular Environments
TD-SCMA	:	Time Division-Synchronous Code Division Multiple Access
DSRC	:	Dedicated Short Range Communication
EMAP	:	Expedite Message Authentication Protocol
ESAR	:	Enhanced Secure Authentication and Revocation
OTP	:	One Time Password
AES	:	Advanced encryption standard
DES	:	Data Encryption Standard
SHA-1	:	Secure Hash Algorithm 1
ESSPR	:	Encryption with Vehicle Proxy Re-encryption
MA-PMIP	:	Multihop Authenticated Proxy Mobile IP
MAC	:	Message Authentication Code
HSM	:	Hardware Security Module

1. INTRODUCTION

The most visible challenges in the Internet of Things (IoT) systems are safety tradeoff with limited system assets. The most predominant part of an IoT network is Machine-to-Machine (M2M) interaction interoperability and interconnection.

The M2M argument includes a broad range of fields, including home systems and telecom equipment with the security of privacy and trusted identity, authentication and authorization requests with inadequate system assets.

The advancement of customer devices also increases intelligent home technologies, leading to the expansion of per consumer of wireless systems. In contrast, this situation increases the use of secret data in public systems and increase safety requirements for M2M links as a result of assaults in [1].

In another aspect, Vehicular Ad Hoc Networks (VANETs) have got respects and positive support in parallel with the changes in transportation technologies. The use of VANETs applications and demands has increased, mostly due to the improvements in the smart city and the smart car ideas and the requirements of interconnection and interoperability between the smart cars and the smart city scenarios for road safety and marketing plans.

VANETs consist of items including infrastructure Road-Side-Units (RSU), Root-Trusted-Authority (R-TA) – related to Trusted Authority (TA) – and On-Board-Units (OBUs). Optionally Lower-Level-Trusted-Authority (LL-TA) can be stated in that infrastructure which advised in [2]. OBUs placed on cars, and RSUs put on roadsides. R-TA and LL-TA are services that provide Public Key Infrastructure (PKI) certificates for authentication and revocation. In standard, RSUs direct interface to R-TA. In some cases of handling heavy traffic, there can be a sub-management item called LL-TA located among R-TA and RSUs. That infrastructure implements a hierarchical communication scheme. The primary forms of interaction that are available to OBUs separately to work with one another and with the RSUs are Vehicle-to-Vehicle (V2V), Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure-to-Vehicle (V2I2V) [3] contacts.

In fast-moving vehicle situations, each VANETs assistance defect and attack can disable VANETs support and cause fatal circumstances. Wireless vehicle communication carriers are sensitive to attacks declared in [1]. Official regulations force systems to perform fast and secure communication infrastructure for Internet-Of-Vehicle (IoV) ecosystems. In usually, adopted VANETs secure communication infrastructure is PKI based solution with Certificate Revocation Lists (CRLs) checking. In basic PKI systems, TA provides PKI certificates and PKI key pairs for each OBU. During V2V connections OBUs check the revocation situation of the sender OBU with CRLs search and verifies certificate signature to guarantee the source of the data is trustworthy. CRLs operated by TA and sent to OBUs.

This thesis focuses on the protection of privacy considered in [4] and message modification, state management, the discovery of anomalies in time for reliable communication and finally the revocation of home gateways and user devices by M2M Secure Communication Systems. Also, we have revisited present stages of the system and introduced password changes security, user registration, and house gateway access security. In this research, we aim to overcome security deficiencies in [5] so that the practical use of M2M networks is entirely secure.

In protocol [5], privacy conservation and message change protection are not provided for under the safety protocol. For addressing these issues, we offer methods. Design [5] fails to acknowledge messages and the device status for accurate information exchange. Also, communication is not well preserved. The house gateway and user equipment cancellations in their model do not take into account.

Design [5] shall not be subject to the threats referred to in [4], [6], [7] or to the alteration of a message referenced in [8]. Our involvement is to improve present security for several assaults, and we suggest that a secure authentication and communications system be developed End-to-End. [5] transfers the public channel a plain password and ID(identification data) and expects third-party companies to protect this network during the registering and change of password activities. The GLARM Scheme [4] compared with [5] and examined the protection of privacy. This dissertation also examines the phase retention costs under research guidelines [1].The scheme [5] associated future research [9] carried out in our suggested scheme on the home network safety framework.

Data protection, strong identity, authentication and permission requirements with limited system resources must be met in IoV procedures, and such operations must be carried out at critical times. CRL checks take a long time to match the magnitude of the CRLs implemented in [10], [11] during testing.

The Wireless Access in Vehicular Environments (WAVE) [12], [13] and Dedicated Short Range Communication (DSRC) [14] standards set message broadcast period for each OBU is 300 msec. This message combines telematic car information such as car and driver situation, position and speed. In heavy traffic, each OBU can take lots of information and should verify each certificate before processing the information. According to the time limitation above Expedite Message Authentication Protocol (EMAP) for VANETs [10] partially displace CRLs checking method with Hash-based Message Authentication Code (H-MAC) cryptogram validation defined in [15].

The additional contributions of this thesis on VANETs secure communication channel is to solve EMAP flaws and development of enhanced authentication and revocation algorithms in the EMAP scheme. Also, our scheme gives robust message modification protection, privacy-preserving discussed in [4]. Besides, we provide message identification for reliable VANETs connection.

In this article, we aim to address safety problems within EMAP in order to attain end-to-end safety for IoV networks. In EMAP, the safety agreement contains several faults, which were not resistant to the assaults mentioned in [4], [6], [7], or the message change set in [8], for example, Denial-Of-Service (DoS). EMAP also has time sync issue to process revocation message that creates defective EMAP key updates that will permanently deactivate assistance for VANETs for different OBUs.

Current scheme overall system performance not well studied, EMAP focus on CRLs checking acceleration but security trade-off and resource optimization comparison not examined for all system states. There are additional requests during revocation and authentication process that generates excessive network congestion on VANETs. Privacy-preserving utilization for external eavesdropper [16], [17] with anonymous certificates for attackers [18]–[20] can cause wrong identification for receiver entity. The reason for that problem is the receiver OBU cannot distinguish sender telematic data owner. Setting each PKI certificate with arbitrary pseudo-identity has a practical usage problem. Unique pseudo-identity is a better solution to this problem. Also, another issue with EMAP is related to key-pair-sharing. TA can share each key-pair with multiple OBUs. EMAP suggest revoking all EMAP and PKI key pairs so shared keys will create management difficulties and faulty revocations or performance difficulties. EMAP also does not differentiate messages and system unit status is not well managed for the secure transfer of information. We present techniques for these issues to be resolved.

In many cases, we provide an improved, fast, lasting and faultless end-to-end safe authentication and revocation system and our contributions improve safety. EMAP is a safe encryption scheme mentioned in [1] in comparison to other IoV systems. With the present scheme, EMAP scheduled to authenticate quickly. In this dissertation, we achieve their objective.

The dissertation is structured following. Section 2 outlines the secure communication protocols for M2M and IoV networks, Section 3 reviews compared M2M scheme [5] and Section 4 shows its [5] vulnerabilities. Section 5 reviews EMAP [10] and Section 6 shows its [10] weaknesses. Section 7 presents our proposed secure communication scheme for M2M networks and within Section 8 presents another proposal for our Enhanced Secure Authentication and Revocation (ESAR) scheme for VANETs.

The protocol and safety analyze for our advanced M2M system are presented in Section 9, and another protocol and security assessment is presented in Section 10 for our proposed VANET ESAR system. The safety and performance comparison of the proposed M2M system and [5] is presented in Section 11. The safety and efficiency comparisons of the proposed ESAR and EMAP layout are also presented in Section 12. Section 13 ends the dissertation with potential papers.

2. RELATED WORKS

Secure communication studies VANET and M2M indicate that before any scheme is deployed, it is necessary to fulfill every design unforgeability, anonymity, and traceability in real life experiences. The survey [1] examines our targeted safe M2M interaction system [5]. About a One-Time-Password (OTP) schemes in [21], Scheme [5] is effective concerning efficiency and network congestion; however, confidentiality is not evaluated and contrasted with GLARM Schemes [4]. Furthermore, the system [5] does not take into account the storage cost and has not a sufficient PBA evaluation report in [1]. In [21] and [22], it is suggested the current scheme [5], which is the scheme of a house network model the OTP-based encryption system is [21], while the encryption system oriented on biometric data is [22]. [5] is effective relative to [21] in terms of operating performance and in terms of network congestion.

Table 2.1 shows countermeasures and cryptosystems of the M2M Authentication Network according to [1]. Moreover, in Table 2.2, various attack schemes were evaluated and categorized [1]. In Table 2.2 we have introduced our scheme. The plans for assaults in Table 2.2 from [1] are "completely endorsed" and "partly endorsed." If the design is considered fully supported by the attack, this means that the authors of the system have established the reliability of their model against the selected assault and protection analyses in all situations with proven formal verification methods or simulations. If a scheme says an assault is defeated, but the official verification and simulation are insufficient, then the scheme is partly backed by an assault. If the scheme does not carry out attack safety, the design shall be categorized as not endorsed. Furthermore, in [23]–[26], this relevant notation is also used.

Table 2.1: Summary of Cryptosystems and Countermeasures in M2M Communication [1]

Cryptosystems and Countermeasures	**[27]**	**[4]**	**[28]**	**[6]**	**[8]**	**[5]**	**[7]**	**[29]**	**[30]**	**Ours**
Secure cryptographic hash function [31]		✓	✓	✓	✓	✓		✓	✓	✓
Original data acquisition								✓		
Spatial-Domain transformation								✓		
Time-domain transformation								✓		
Correlation Coefficient-based Matching Algorithm (C-MA)	✓									
Deviation ratio-based Matching Algorithm (D-MA)	✓									
Aggregate Message Authentication Codes (AMACs) [32]		✓							✓	
Certificateless aggregate signature [33]			✓							
Elliptic Curve Diffie-Hellman (ECDH) [34]				✓						✓
ID-based signature scheme [32]					✓					
Advanced encryption standard (AES) [35]						✓				✓
Hybrid Linear Combination Encryption [36]							✓			

Supported = ✓, Empty Not Supported

Table 2.2: Summary of Attacks in M2M Networks and Defense Protocols [1]

Adversary model	[27]	[4]	[28]	[6]	[8]	[5]	[7]	[29]	[30]	Ours
Audio Replay attack	☑	✓		✓	✓	✓			1	☑
Changing the distance attack	☑									☑
Same-type-device attack	☑									☑
Composition attack	☑									☑
Redirection attack	✓	☑	✓	☑			✓		☑	☑
Man-in-the-middle attack	✓	☑	✓	☑	✓	✓			☑	☑
Substitution attack	✓	✓	✓	✓	✓					☑
DoS attack		☑		☑			☑			☑
Replay attack	✓			☑	✓	☑			☑	☑
Forging attack				✓						☑
Colluding attack	✓			✓			✓			☑
Flooding attack	✓						✓		✓	☑
Side-channel attack	✓						✓		✓	☑
False messages attack	✓				✓	✓	✓		✓	☑
Sybil attack					✓	✓			✓	☑
Movement tracking					✓				✓	☑
Message modification					✓					☑
Impersonation attack					✓	☑	☑			☑
Guessing attack						☑				☑
Stolen-verifier attack						☑				☑
Wormhole attack	✓	✓		✓		✓			✓	☑
Blackhole attack	✓	✓		✓	✓	✓			✓	☑
Attribute-trace attack					✓					☑
Eavesdropping attack					✓	✓			✓	☑
Chosen-plaintext attack					✓				✓	☑
Spam attack	✓				✓	✓			✓	☑
Identity theft attack	✓					✓				☑
User manipulation attack	✓					✓	✓		✓	☑
Routing attack	✓					✓				☑
Linkability attack	✓									☑
Rejection attack										☑
Successive-response attack										☑
Packet analysis attack		✓				✓			✓	☑
Packet tracing attack		✓				✓			✓	☑
Brute-force attack	✓	✓		✓	✓		✓	✓		☑

Fully Supported = ☑, Partially Supported = ✓, Empty Not Supported

With the respect of VANET secure connection schemes. Hubaux [37] and Raya et. al’s [18] consider PKI systems with anonymous certificates to provide privacy-preserving on VANETs for each OBU, Ferrag and Ahmim [38] propose Encryption with Vehicle Proxy Re-encryption (ESSPR) to provide privacy-preserving in VANETs and efficient to against attacks such as eavesdropping attack, wormhole attack, packet analysis attack, packet tracing attack and replay attack. Studer et al.'s[39] propose certificates for a group in VANETs to implement fast authentication and revocation, but in real life implementation, it has performance flaws. Also, platoon member anonymity protection proposed in [40] for group setup and anonymous authentication. PKI demands of CRLs distributions studied in [41]–[44] and proposed efficient large CRLs distributions methods. Probabilistic key distribution approaches for IoV networks are considered in [45]–[47]. The MITM security technique as suggested in PKI scheme is uniformly certified [48] This technique works well for the computing difficulty compared to previous procedures such as [49], [50], authentication key exchange (AKE). The security association between asymmetric link for V2V communication thought in [51] and named Multihop Authenticated Proxy Mobile IP (MA-PMIP) scheme. MA-PMIP authenticates and prevents authentication assaults, like replay assaults MITM and DoS. Moreover, MA-PMIP is efficient than [52] about OBU telematics data authentication. However, privacy-preserving not analyzed. The non-reputation demand studied in PBA [53]. With small memory usage, PBA can withstand packet loss and retain high-level network traffic. The PBA ensures prompt authentication, DoS resistant attacks and packet loss resistant non-repudiation. PBA build on Merkle hash tree [54]. Substitution attack resistance studied in [55] and provided with two-factor authentication. Identity-Based Aggregate Signatures for VANETs introduced in [2], [56] and efficient in terms of the Elliptic Curve Digital Signature Algorithm (ECDSA) protocol described in [13]. Decentralized group model examined in [57] and proposed group signature satisfies, unforgeability, anonymity and traceability on VANETs. Finally, EMAP[10] propose message authentication acceleration by replacing CRLs search checking methods with keyed H-MAC. That utilization overrides the issue of the lengthy interval for each authentication check status. However, this is not appropriate for decentralized designs mentioned in [1]. In the study [1], EMAP is examined, saying that entry delay is constant irrespective of the number of certificates which have been revoked EMAP has unique rekeying techniques centered on bilinear pairing [58]and used with hash chains [59] produced with Secure Hash Algorithm-1 (SHA-1) [60]. EMAP offers unforgeable safety with a discrete logarithm problem with the Elliptic curve (ECDLP) [61]

EMAP also specify and compare CRLs search methods [62] such as linear search, binary search and lookup hash tables those are not in [12]. Table 2.3 shows the IoV authentication protocol countermeasures and cryptosystems. Besides, [1] analyzed and organized several systems to attacks in Table 2.4. We expanded their results with our proposed scheme in Table 2.4.

Table 2.3: Summary of Cryptosystems and Countermeasures in Internet-of-Vehicle (IoV) [1] (updated)

Cryptosystems and Countermeasures	[51]	**[57]**	[53]	**[2]**	[48]	[55]	[56]	[40]	**EMAP[10]**	**ESAR**
Secure cryptographic hash function [31]		✓	✓	✓	✓		✓	✓	✓	✓
Proxy Mobile IP (PMIP) [63]	✓									
Symmetric polynomials [64]	✓									
Search algorithms [62]									✓	✓
Group signature [65], [66]		✓								
Merkle hash tree (MHT) [67]			✓							
TESLA scheme [68]			✓							
ECDSA signature [13]			✓							
Multiplicative secret sharing technique [69]				✓						
Identity-based public key cryptosystem [70]							✓			
Identity-based aggregate signature [71]							✓			
Digital signatures [72]					✓				✓	✓
Anonymous attribute-based group setup scheme [73]								✓		
Keyed-hashing for message authentication (HMAC) [15]						✓			✓	✓
Elliptic Curve Diffie-Hellman (ECDH) [34]									✓	✓
Elliptic curve discrete logarithm problem (ECDLP) [61]									✓	✓
Advanced encryption standard (AES) [35]									✓	✓
Identity-Based Cryptography (IBC) [74]									✓	✓

Supported = ✓, Empty Not Supported

Table 2.4: Summary of Attacks in IoV Networks and Defense Protocols [1]

Adversary model	[51]	[57]	[53]	[2]	[48]	[55]	[56]	[40]	EMAP [10]		ESAR
									The analysis in [1]	Our Analysis	
Audio replay attack	✓	✓		✓	✓		✓		✓	☑	☑
Changing the distance attack										☑	☑
Same-type-device attack										☑	☑
Composition attack										☑	☑
Redirection attack	✓								✓	✓	✓
Man-in-the-middle attack	☑	✓			☑	✓			✓		☑
Substitution attack	✓	✓			✓	☑			✓		☑
DoS attack	☑		☑	☑	☑					✓	☑
Replay attack	☑	☑		✓	✓	✓	☑	✓	☑	☑	☑
Forging attack	✓				✓				☑	☑	☑
Colluding attack	✓		✓						☑	☑	☑
Flooding attack			✓							☑	☑
False messages attack				☑				✓			☑
Sybil attack	✓			☑	✓			✓		☑	☑
Message modification						✓	☑				☑
Wormhole attack	✓			✓		✓	✓	✓	✓	☑	☑
Blackhole attack	✓			✓		✓	✓	✓	✓	✓	☑
Attribute-trace attack					✓			✓		✓	☑
Eavesdropping attack		✓	✓	✓			✓	✓		☑	☑
Chosen-plaintext attack			✓			✓		✓		☑	☑
Spam attack			✓		✓	✓				✓	☑
Identity theft attack			✓			✓				☑	☑
User manipulation attack			✓			✓	✓				☑
Routing attack	✓	✓		✓		✓	✓	✓		☑	☑
Linkability attack					✓		✓			☑	☑
Rejection attack					✓		✓	✓		☑	☑
Successive-response attack					✓						☑
Packet analysis attack	✓			✓	✓		✓	✓	✓	☑	☑
Packet tracing attack	✓			✓	✓		✓	✓	✓	☑	☑
Brute-force attack					✓		✓	✓		✓	✓

Fully Supported = ☑, Partially Supported = ✓, Empty Not Supported

3. REVIEW OF SUN ET AL.’S [5] M2M SCHEME

In this section, we review the M2M domestic network safety identification system using Sun et al.'s current Time Division-Synchronous Code Division Multiple Access (TD SCMA) [5]. Its design consists of three main components: the user device, the M2M server, and the home station. The scheme [1] comprises of five stages: installation, user registration, client login, and authentication, change of user password and access to the TD-SCMA network from home gateway. As shown in Figure 3.1, the M2M server is the primary server in this system.

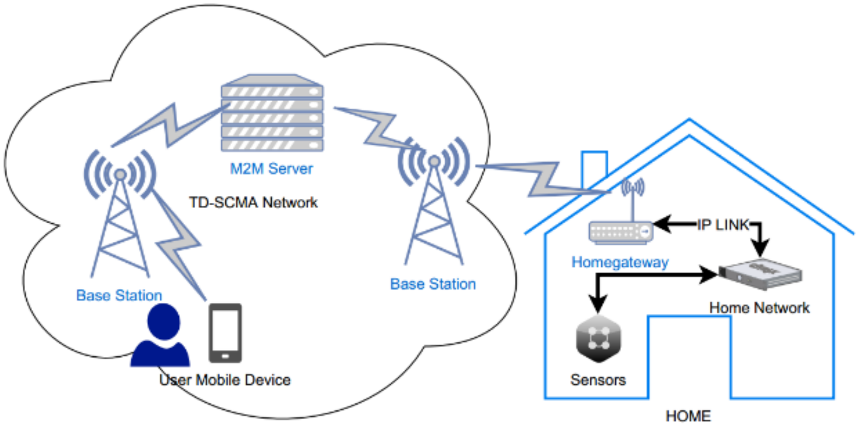


Figure 3.1: Structure of Sun et al. ’s [5] M2M Scheme

M2M servers depict themselves and include confidential data encryption necessary with a unique identifier. For registration via unsecured transmission, a user selects a password and a user ID that moves the equipment to the M2M server. The M2M server ciphers and saves the password and the M2M Server generates, transmits and reserves random data in M2M's server database to Mobiles for shared authentication. The portable client device stores this arbitrary information. The user equipment creates a message with a user ID and a ciphered random number during the login and authentication implementation. This arbitrary customer identification number is ciphered by the user equipment. This message is sent for registration by the user device to the M2M server. Each unit calculates and verifies one session key. The key for mutual authentication is used by each organization as the primary key. Authenticated users can use their mobile device to access their home gateway data over the M2M server. Each home gateway connects the system to its ID in this mode. During a home-gateway boot, it gives its ID to the M2M Server, where the server searches this user ID, and if it contains and matches, it then calculates the hash of the consolidated user ID, arbitrary user information and M2M Server ID data and sends it for event key calculation to the home gateway equipment. Furthermore, the M2M server calculates the session key for a house gateway for information exchange services. Table 3.1 contains the notations of the Protocol, and the parameters used in the present scheme in Table 3.2 are displayed.

Table 3.1: Notation of Sun et al. ’s [5] Scheme

Notation	Description
k	The secret key of the M2M server
uid	Mobile user’s ID
mid	M2M server’s ID
pw	User’s password
umk	The session key between a user and the M2M server
hid	Home gateway ID
sf	Status flag of the mobile user
$h(.)$	Cryptographic hash function
$h(m,n)$	Hashing of the concatenation of m and n
$[m]^n$	Most significant n bits of string m
$F_{e-AES}(a,b)$	Using AES algorithm to cipher a with b as the key
$F_{d-AES}(a,b)$	Using AES algorithm to decipher a with b as the key

Table 3.2: Protocol Parameter Settings at Sun et al. 's [5] Scheme

Parameters	**Settings and description**
s	Salt, a 16-bit random number
k	The secret asymetric key of M2M server, 64 bits
uid	The equipped sim card number, 80 bits
mid	The equipped sim card number, 80 bits
hid	The equipped sim card number, 80 bits
umk	A session key,128 bits
pw	User's password,64 bits
sf	User's statue flag, 1 bit
x_1	128 bits
x_2	128 bits
$h(.)$	Sha-1 hash, 160 bits output
m	128 bits
n	128 bits
t	128 bits
$F_{e-AES}(a,b)$	AES encryption function with 128 bits input, 128 bits key and 128 bits output
$F_{d-AES}(a,b)$	AES decryption function with 128 bits input, 128 bits key and 128 bits output

The MATLAB performance measures and comparisons for the application of AES in Figure 3.2 and Figure 3.3 are shown by the scheme [5] and the suggested system. Design [5] did not demonstrate their analysis in encoding and decryption activities carried out by the AES key schedule method. We realized that the outcomes from their analysis close to AES encoding and decryption without the outcomes of a cipher key schedule method. End-to-end scheme performance metrics are closely related to the results of the measurement in [5]. Current assessment results indicate that AES ciphering is faster than decryption, as indicated in the scheme [5]. The results are similar. These findings are shown in Figure 3.2 and Figure 3.3. Figure 3.4 demonstrates the results metrics for the [5] and the system suggested as well as references for the application of the SHA-1 in MATLAB.

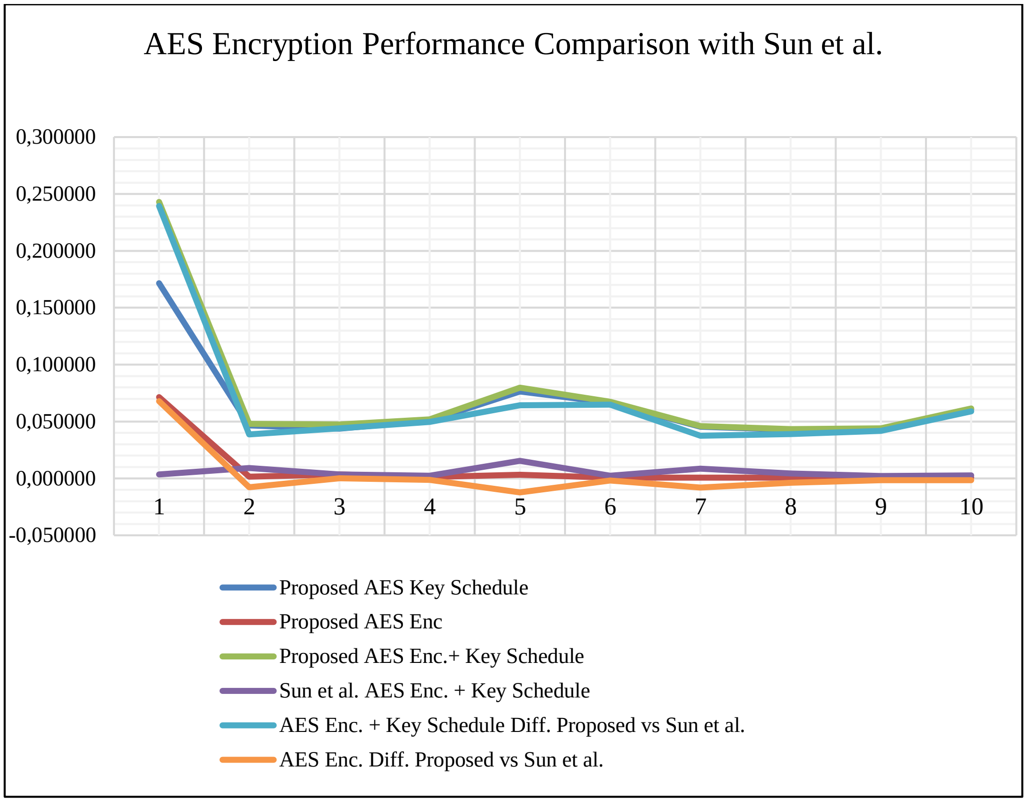


Figure 3.2: AES Encryption Performance Comparison with Sun et al. 's [5]

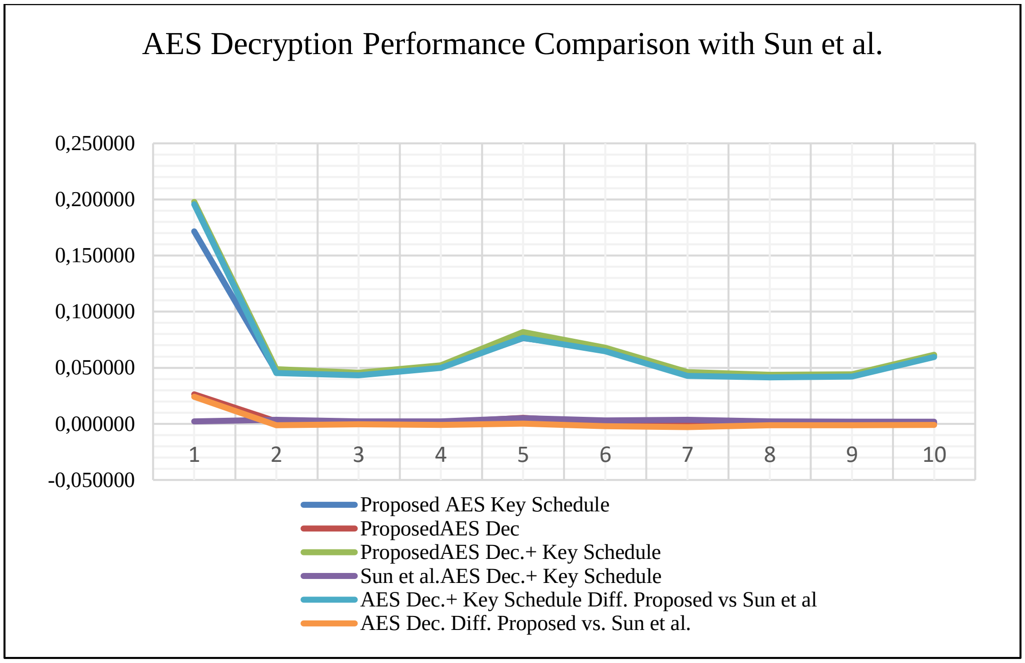


Figure 3.3: AES Decryption Performance Comparison with Sun et al. 's [5]

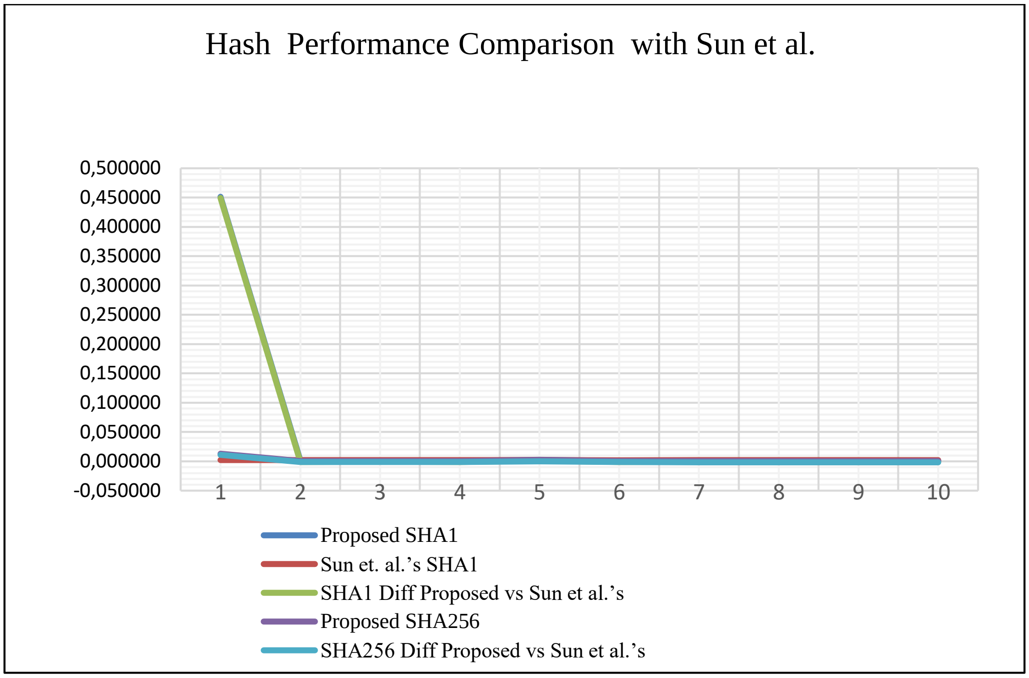


Figure 3.4: Hash Performance Comparison with Sun et al. 's [5]

The following paragraphs cover the whole stage of scheme analysis[5].

3.1 SUN ET AL.'S [5] M2M SCHEME SETUP PHASE

The M2M server chooses 64-bit confidential server key (k) during configuration procedure. Cryptographic methods do not use this key directly. M2M server uses this key as an input to calculate the service key. M2M server does not specify the ID personalization in the set-up phase for each item. M2M server specifies server key during the set-up phase in Figure 3.5, Figure 3.6 and Figure 3.7.

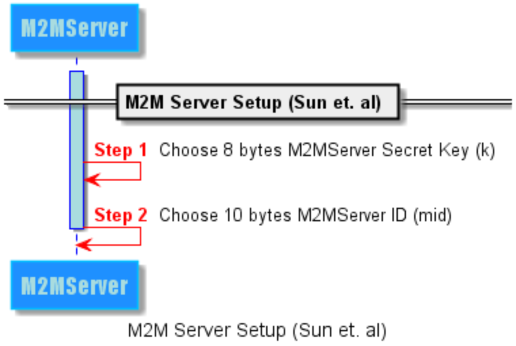

M2MServer
M2M Server Setup (Sun et. al)
Step 1 Choose 8 bytes M2MServer Secret Key (k)
Step 2 Choose 10 bytes M2MServer ID (mid)
M2MServer
M2M Server Setup (Sun et. al)

Figure 3.5: M2M Server Setup for Sun et al. 's Scheme

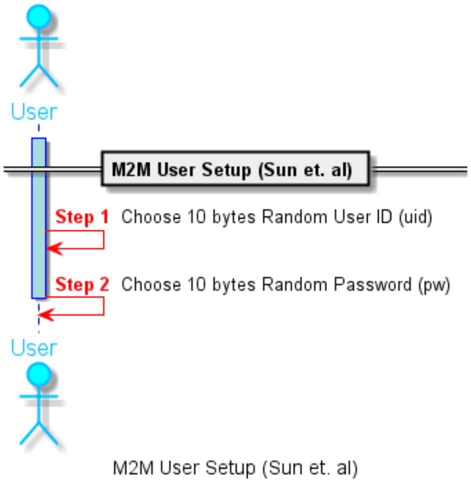

User
M2M User Setup (Sun et. al)
Step 1 Choose 10 bytes Random User ID (uid)
Step 2 Choose 10 bytes Random Password (pw)
User
M2M User Setup (Sun et. al)

Figure 3.6: User Setup for Sun et al. 's Scheme

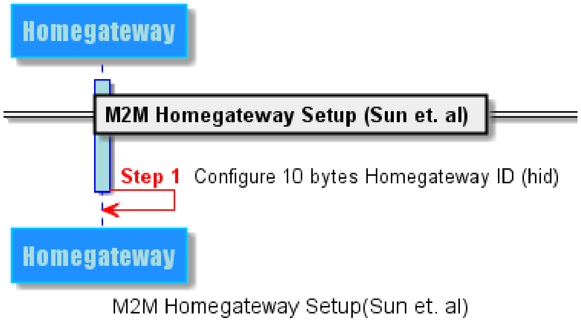

Homegateway
M2M Homegateway Setup (Sun et. al)
Step 1 Configure 10 bytes Homegateway ID (hid)
Homegateway
M2M Homegateway Setup(Sun et. al)

Figure 3.7: Home Gateway Setup for Sun et al. 's Scheme

To determine the M2M server, there must be one 80-bits M2M server ID given to the smart card. Smartcards are available for the server. The home gateway computer has a unique 80-bits ID and a smartcard given to which device supplier is using the home gateway device. Also, the 80-bits ID stored on the smart card recognizes the clients

3.2 SUN ET AL.’S [5] M2M SCHEME USER REGISTRATION PHASE

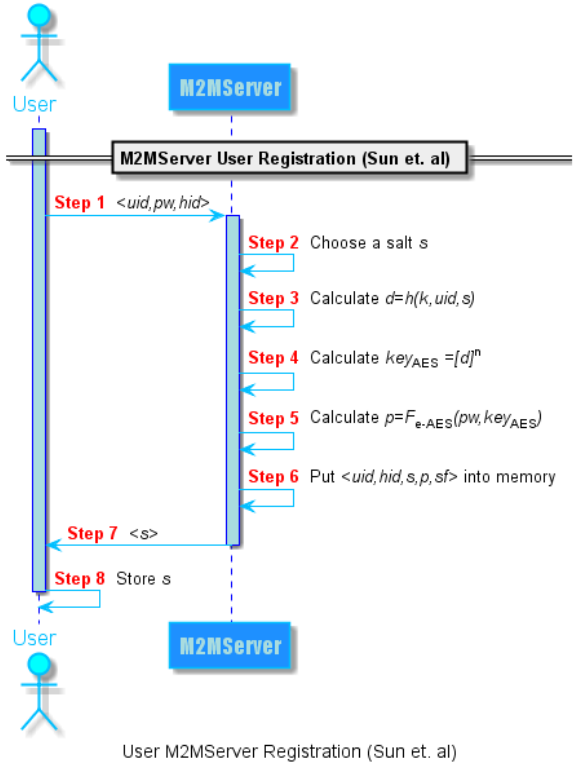

User
M2MServer
M2MServer User Registration (Sun et. al)
Step 1 <uid,pw,hid>
Step 2 Choose a salt s
Step 3 Calculate d=h(k,uid,s)
Step 4 Calculate key_AES =[d]^n
Step 5 Calculate p=F_e-AES(pw,key_AES)
Step 6 Put <uid,hid,s,p,sf> into memory
Step 7 <s>
Step 8 Store s
User
M2MServer
User M2MServer Registration (Sun et. al)

Figure 3.8: M2M Server User Registration Phase for Sun et al. ’s Scheme

A new system user is logged in with the 80-bit user ID, 64-bit transparent password and 80-bit portal identification following $< uid, pw, hid >$ for registering. This is a user register in Figure 3.8 above. [5] considers that the third-party system protects information channel and for this reason channel assumed as secure. For this perception, the entities in the system transmit the password in the open form. In addition, the present architecture claims that entities can reliably perform the interaction. No definition of the message is accessible for this purpose. A M2M server should select an arbitrary salt number (s) and condense with the user ID and the M2M secret server key The M2M server will then calculate the SHA-1 one-way buffer and the AES authentication key will be provided with the first 128-bits of the 160-bit SHA-1 output as follows:

$$d = h(k, uid, s) \tag{3.1}$$

$$key_{AES} = [d]^n \tag{3.2}$$

The M2M server encrypts the memory of client password as follows for database storage:

$$p = F_{e-AES}(pw, key_{AES}) \tag{3.3}$$

The user and home gateway ID, salt, encrypted password and status flag $< uid, hid, s, p, sf >$ are placed in the database of the M2M server. Finally, salt is stored by the client equipment for login and authentication.

3.3 SUN ET AL.'S [5] M2M SCHEME USER LOGIN AND AUTHENTICATION PHASE

This phase ensures secure client communication through the execution of mutual authentication across the TD-SCMA network between the M2M server and the user unit. This system does not recognize the messages and does not consider a verifiable source. Figure 3.9 shows this phase transition for an authentication application, the client device chooses a random number x_1 and calculates the following parameters:

$$g = F_{e-AES}(x_1, pw) \tag{3.4}$$

$$h_1 = h(x_1) \tag{3.5}$$

The device transfers the request parameters $< uid, g, h, s >$ to the M2M server, which is used in the M2M server database to search for uid and verify. The M2M server denies the application if uid does not occur.

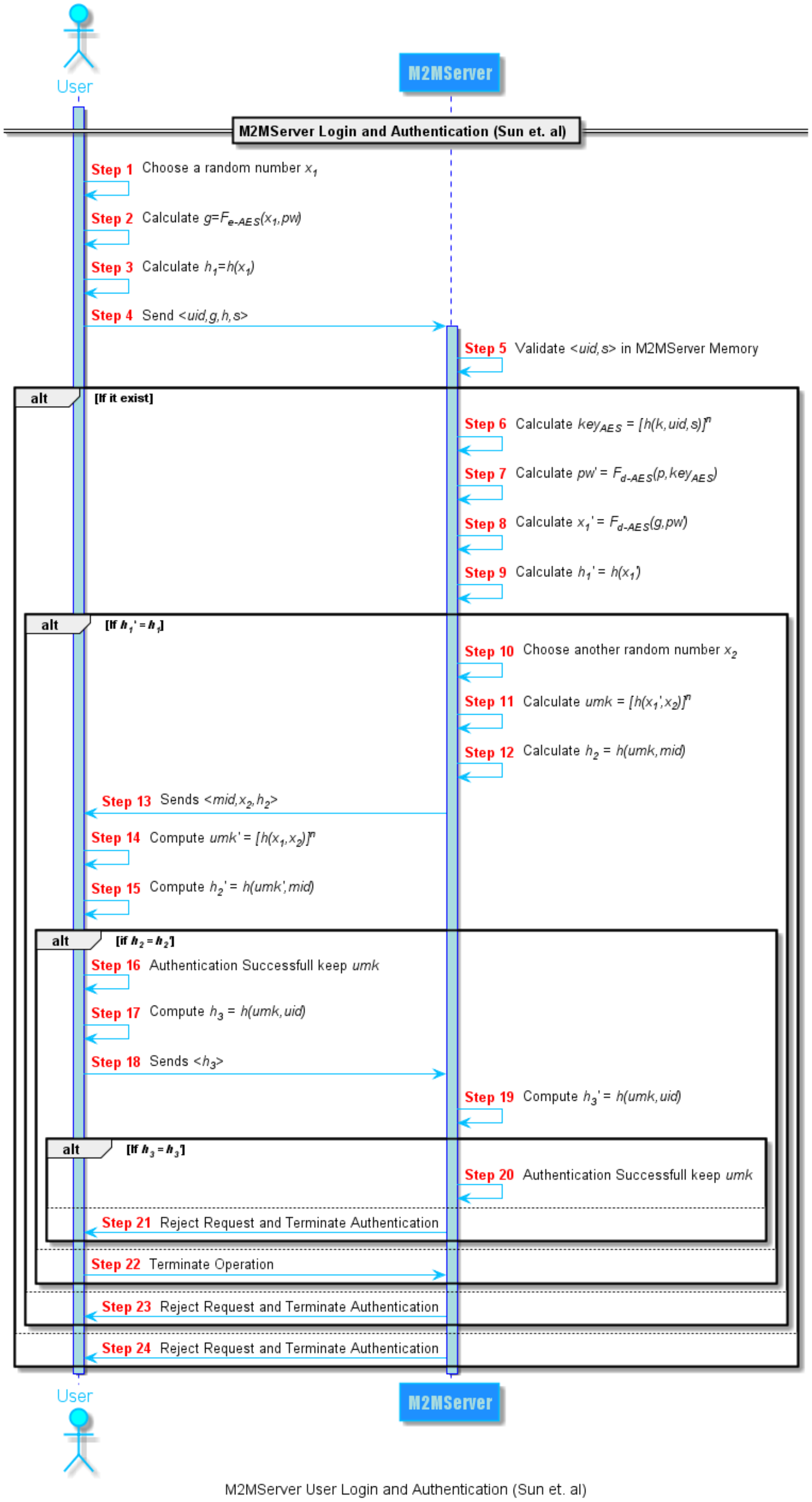

User
M2MServer
M2MServer Login and Authentication (Sun et. al)
Step 1 Choose a random number x_1
Step 2 Calculate $g=F_{e\text{-}AES}(x_1,pw)$
Step 3 Calculate $h_1=h(x_1)$
Step 4 Send $<uid,g,h,s>$
Step 5 Validate $<uid,s>$ in M2MServer Memory
alt [If it exist]
Step 6 Calculate $key_{AES} = [h(k,uid,s)]^n$
Step 7 Calculate $pw' = F_{d\text{-}AES}(p,key_{AES})$
Step 8 Calculate $x_1' = F_{d\text{-}AES}(g,pw)$
Step 9 Calculate $h_1' = h(x_1')$
alt [If $h_1' = h_1$]
Step 10 Choose another random number x_2
Step 11 Calculate $umk = [h(x_1',x_2)]^n$
Step 12 Calculate $h_2 = h(umk,mid)$
Step 13 Sends $<mid,x_2,h_2>$
Step 14 Compute $umk' = [h(x_1,x_2)]^n$
Step 15 Compute $h_2' = h(umk',mid)$
alt [if $h_2 = h_2'$]
Step 16 Authentication Successfull keep umk
Step 17 Compute $h_3 = h(umk,uid)$
Step 18 Sends $<h_3>$
Step 19 Compute $h_3' = h(umk,uid)$
alt [If $h_3 = h_3'$]
Step 20 Authentication Successfull keep umk
Step 21 Reject Request and Terminate Authentication
Step 22 Terminate Operation
Step 23 Reject Request and Terminate Authentication
Step 24 Reject Request and Terminate Authentication
User
M2MServer
M2MServer User Login and Authentication (Sun et. al)

Figure 3.9: M2M Server User Login and Authentication Phase for Sun et al. 's Scheme

The M2M server will retrieve the key from the message and decrypt the saved password and prepare the same calculations to check the mark to ensure the user's password is correct as below:

$$key_{AES} = [h(k, uid, s)]^n \tag{3.6}$$

$$pw' = F_{d-AES}(p, key_{AES}) \tag{3.7}$$

$$x_1' = F_{d-AES}(g, pw') \tag{3.8}$$

$$h_1' = h(x_1{}') \tag{3.9}$$

When the value of h_1' received equals the content of h_1, it ensures the correct user password and ID and establishes the user on the M2M Server. On the opposite, the transaction is refused by the M2M server. The M2M server selects a random x_2 to set the umk for the transmission session key and gives the necessary objects to the client devices for the appropriate session key generation, as follows:

$$umk = [h(x_1', x_2)]^n \tag{3.10}$$

$$h_2 = h(umk, mid) \tag{3.11}$$

The M2M server saves and sent to a user's equipment $< mid, x_2, h_2 >$. The user unit uses a generated $x_1$and the mid, x_2, and h_2 for production of session key umk and checks the h_2 content for M2M servers as follows:

$$umk' = [h(x_1, x_2)]^n \tag{3.12}$$

$$h'_2 = h(umk', mid) \tag{3.13}$$

The session key for safe transmission is saved in the mobile user equipment if the calculated value of respect h'_2 is equal to the h_2 data approved and, finally, the device uses a hash h_3 as:

$$h_3 = h(umk', uid) \tag{3.14}$$

The user device transmits the h_3 to the M2M server, and the h_3 hash is finished by the M2M server:

$$h'_3 = h(umk, uid) \tag{3.15}$$

If the h_3 is equal to the h'_3 value, the M2M server establishes a mutual authentication and user devices and umk for secure data transmission.

3.4 SUN ET AL.'S [5] M2M SCHEME USER PASSWORD CHANGE PHASE

A user password introduces time-based safety, which is why the customer needs to systematically update the password. The change of password is shown in Figure 3.10.

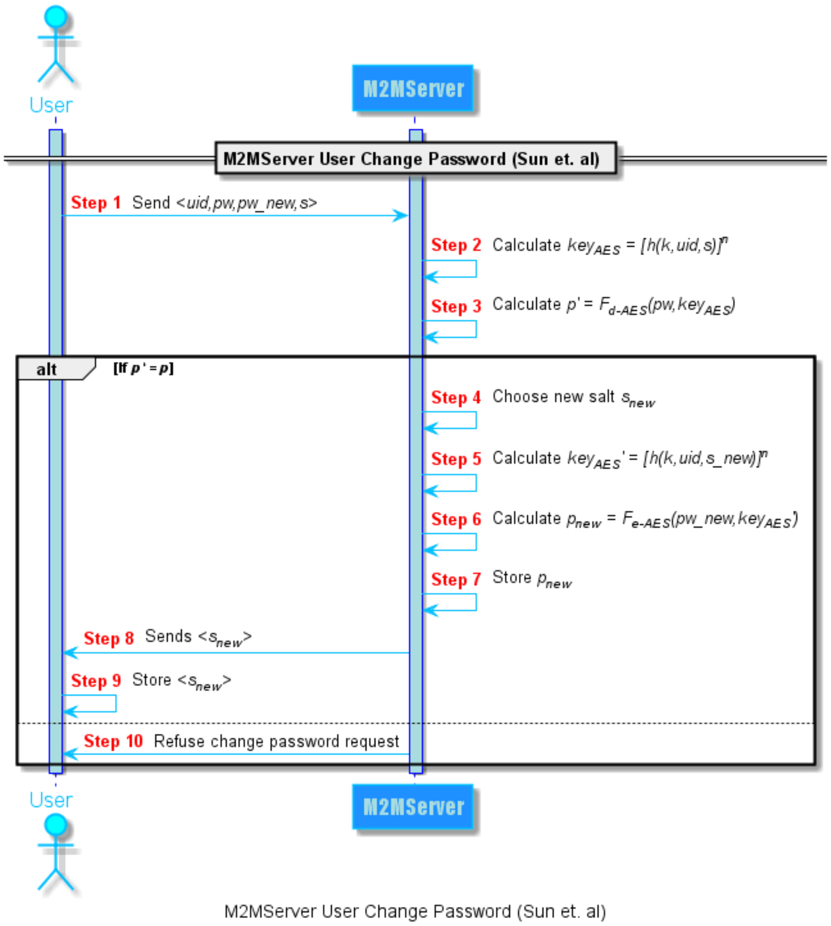

User
M2MServer
M2MServer User Change Password (Sun et. al)
Step 1 Send <uid,pw,pw_new,s>
Step 2 Calculate key_{AES} = $[h(k,uid,s)]^n$
Step 3 Calculate p' = $F_{d\text{-}AES}(pw,key_{AES})$
alt
[If p' = p]
Step 4 Choose new salt s_{new}
Step 5 Calculate key_{AES}' = $[h(k,uid,s\_new)]^n$
Step 6 Calculate p_{new} = $F_{e\text{-}AES}(pw\_new,key_{AES}')$
Step 7 Store p_{new}
Step 8 Sends <s_{new}>
Step 9 Store <s_{new}>
Step 10 Refuse change password request
User
M2MServer
M2MServer User Change Password (Sun et. al)

Figure 3.10: User Password Change Phase for Sun et al. 's Scheme

At this stage the user will refresh the revoked password, updating the next login and authentication functions with his own arbitrary salt. Scheme [5] claims that for the method of password change, the information transportation channel is reliable and safe. The process of password begins with the user equipment transmitting the user ID, plain old password, removing the new password, and formatting updated salt content $< uid, pw, pw_{new}, s >$ to the M2M server. The M2M server searches for the user ID and retrieves a ciphered password p in its database, after which the authentication key is retrieved and the password pw is ciphered for checking by certain for later calculations:

$$key_{AES} \ = [h(k, uid, s)]^n \tag{3.16}$$

$$p' = F_{e-AES}(pw, key_{AES}) \tag{3.17}$$

The M2M server will select a new, arbitrary salt (s_{new}), and cipher the new key in the M2M server database if the computed p' password matches that recorded p as follows:

$$key'_{AES} \ = [h(k, uid, s_{new})]^n \tag{3.18}$$

$$p_{new} = F_{e-AES}(pw_{new}, key'_{AES}) \tag{3.19}$$

The M2M computer publishes the encrypted password p_{new} and gives new salt s_{new} to the client host. The device unit will store s_{new} for a further login and validation method.

3.5 SUN ET AL.'S [5] M2M SCHEME HOME GATEWAY JOIN TD-SCMA NETWORK

When powered, home gateway equipment sends requests to join the network to M2M servers, as shown in Figure 3.11.

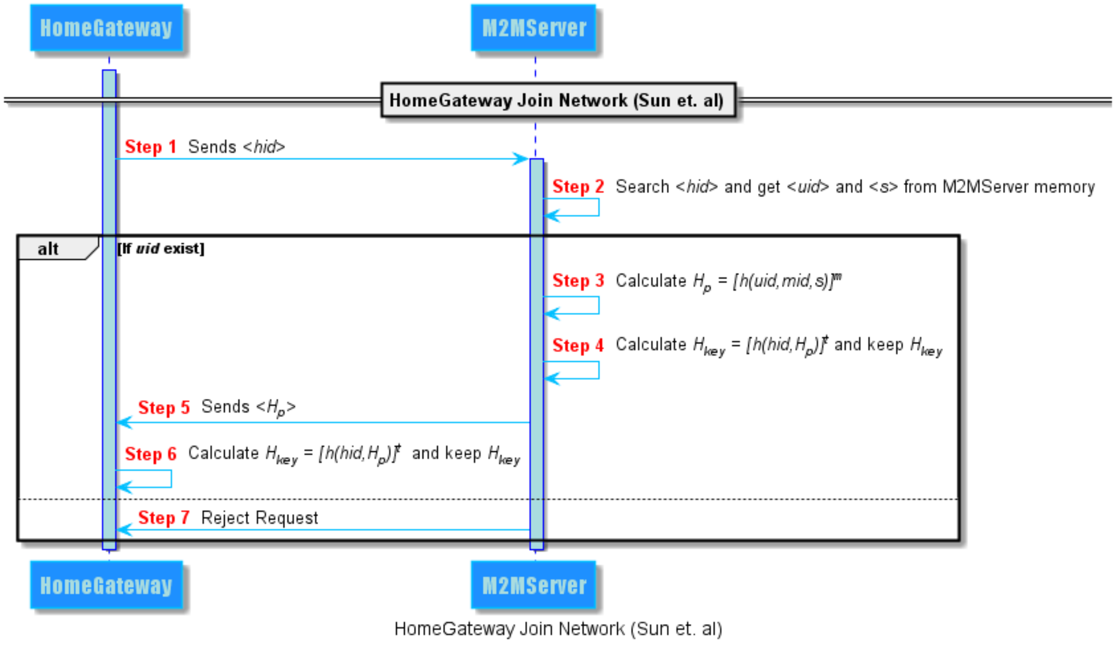

HomeGateway
M2MServer
HomeGateway Join Network (Sun et. al)
Step 1 Sends <hid>
Step 2 Search <hid> and get <uid> and <s> from M2MServer memory
alt
[If uid exist]
Step 3 Calculate $H_p = [h(uid,mid,s)]^m$
Step 4 Calculate $H_{key} = [h(hid,H_p)]^t$ and keep H_{key}
Step 5 Sends <H_p>
Step 6 Calculate $H_{key} = [h(hid,H_p)]^t$ and keep H_{key}
Step 7 Reject Request
HomeGateway
M2MServer
HomeGateway Join Network (Sun et. al)

Figure 3.11: Home gateway Join for Sun et al. 's Scheme

This application involves a distinctive pre-personalized identifier $< hid >$ saved on a smart card by card issuers. Each home gateway must be connected to the M2M server's client equipment The M2M server tests user devices related to this home gateway equipment ID for this purpose. If user equipment exists, the M2M server calculates the information processing session key H_{key} followed by the information recovery key H_p for the home gateway system, thus:

$$H_p \ = [h(uid, mid, s)]^m \tag{3.20}$$

$$H_{key} \ = [h(hid, H_p)]^t \tag{3.21}$$

H_{key} will be saved on the M2M server and H_p is placed on the home gateway device, calculating H_{key} for data transfer. It is not necessary to recognize the user identity in the session key generation of the Home gateway equipment,thus:

$$H_{key} \ = [h(hid, H_p)]^t \tag{3.22}$$

4. WEAKNESSES OF SUN ET AL.'S [5] M2M SCHEME

In comparison to the system in [21], the system of Sun et al. [5] is lower in calculation costs. Scheme [5] may withstand the assault from stolen-verifier, replay attack, guessing assault and impersonation attack. In general safety, there are still several safety vulnerabilities.

Scheme [5] is not anonymous. During the authentication phase, the current system sends customer identity information plain. Also, while the encryption stage does not examine confidentiality, this issue is stated in [1], which allows us to compare the confidentiality of the system [5] to GLARM [4]. GLARM uses temporary user identification data, which is derived by the central entity from original user identification data, and the network entities transmit temporary user identification for privacy conservation in data messages. Never pass real client identifier in messages to network agents. The Schema [5] does not analysis storage costs, but in our opinion, we did simulate the present scheme and analysis for each stage of each resource and memory change. For other models, a quality assessment is insufficient. In addition, the efficiency assessment of original settings does not show their job. Our simulation includes and analysis of all scheme phases, our used performance and memory usages analysis.

Scheme [5] has no safe registration. Their system sends the password with an original user ID in an accessible format. They suppose that this carrier is protected and therefore we do not need further authentication, if we have a secure channel that is already safe.

Scheme [5] does not also offer a secure transaction of the change of password, such as registering. During this operation, user devices will send clear user ID and password.

A significant privacy issue in Scheme [5] persists in the clear password sending process during registration and a password change process with the original user ID. Section 3 describes the system [5] change of password and messages transmitted during delivery. For each stage, there is no MAC (Message Authentication Code) verification. During operations, there is no reliability. Scheme entities do not separate messages with a sequence or message tag, and messages between communicating entities are not confirmed. Scheme [5] has no timeout alert and detection system for anomalies. Also, in the system periods for lost or damaged messages and sessions there is no communication recovery procedure. The home device leads unprotected against DoS assaults into the network join stage. A fraudulent home gate equipment can try to enter the network as this procedure is not restricted. If an attacker is using a home gateway ID from which exposed from other home gateway device connection, then the M2M server for the fraudulent device or device simulators can conduct session important calculation operation. Scheme [5] does not include revocation of users and devices [5]. Salt data will be used for session key agreement in the home gateway network stage, and the salt parameter will be updated to the M2M server and user equipment during a password modification phase. The new salt value will be saved after the request for a password change by the user device and the M2M server, but the home gateway device uses the session key created with the old salt, and the salt and session key on the home gateway side can not be upgraded again. When client updates periodically password owned for safety purposes, this change does not affect the home device and increases the danger.

5. REVIEW OF EMAP [10] VANET SCHEME

This section evaluates EMAP's suggested safe encryption system in VANETs[10] The three main components of their scheme are OBUs, TA and RSUs. The [10] scheme is comprised of three stages: initialization of the scheme, OBU to OBU authentication and revocation of OBU. As illustrated in Figure 5.1 TA is the main server in these systems.

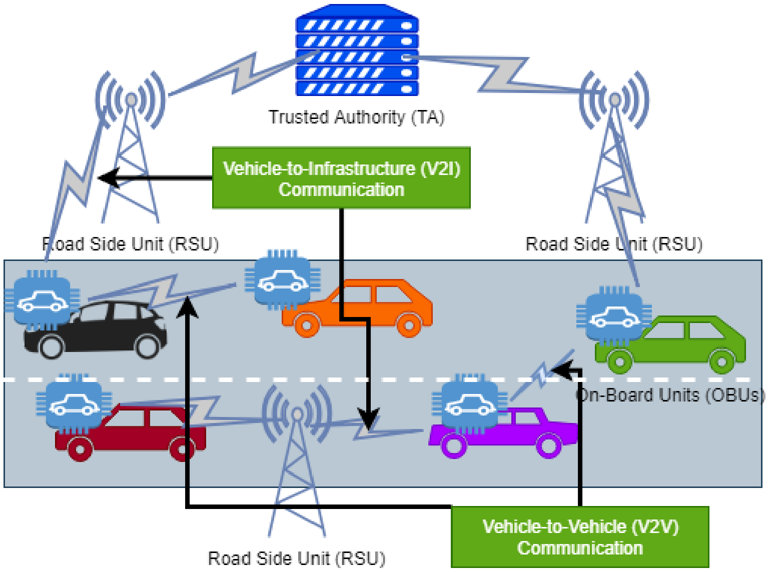


Figure 5.1: Structure of EMAP[10] VANETs Model

OBUs equipped with Hardware Security Module (HSM) [75] which is attached to the car. Sensitive assets such as secret keys stored in HSM, which is tamper proof resistance. Additionally, all cryptographic operations processed in HSM. TA connected to OBUs over RSUs those fixed units and located on the roadsides. The communication interface between RSUs and OBUs called V2I, and another interface between OBUs called V2V communication. The notation of protocols mentioned in Table 5.1 and Table 5.2 displays the present scheme parameter configurations. EMAP uses H-MAC cryptogram validation for revocation check method, instead of CRLs list search. EMAP propose bilinear pairing [58] based rekeying mechanism for H-MAC key replacement.

The TA provides following sensitive and cryptographic assets for each OBU for authentication and revocation:

First assets are PKI public and private key pair sets for message signing and verification. Moreover, PKI key pairs encrypt sensitive and cryptographic asset for secure transmission between OBUs. The ESAR scheme not considered the PKI algorithm; for this reason, we choose secp256k [76], [77] elliptic curve for ECDSA and Elliptic Curve Integrated Encryption Scheme (ECIES) operations.

Additional assets are PKI anonymous certificates for privacy preserving. In EMAP design, each OBU can have multiple pseudo identification data (PID) for certificate identification. PID usage provides privacy-preserving but traceability not considered. Certificates validated after EMAP revocation check control each OBU should verify TA certificate signature and message signature before processing the message. TA master private key stored on the server and corresponding public key deployed to OBUs. In the EMAP scheme, TA PKI certificate generation methods not considered.

Next assets are EMAP revocation check secret key for expediting message authentication that we have used for asymmetric cryptographic message authentication code (MAC) generation. EMAP calculate revocation check value with keyed H-MAC or AES-CBC-MAC. in the current scheme [10], revocation checking secret key's asymmetric cryptographic usage not described, for this reason, in the ESAR scheme, we use secret key point x-axis value as a secret key component and trim for related MAC algorithm's default key block size. Revocation check operation, secret key initial OBU injection security not considered in current design [10]. During EMAP rekeying and revocation process, revocation check key encrypts EMAP key pair rekeying hash chain value.

Additionally, EMAP provides revocation key pairs for secure rekeying the revocation check key. Shared revocation check secret key renewed with asymmetric operation by using these key pairs and hash chain values. Rest of paper we call this cryptographic-key-pair structure as EMAP Key Infrastructure (EKI)

Final provided assets by TA are Signed CRLs for revocation check during EMAP revocation process. TA maintains PKI CRLs and broadcast to the network for other OBUs. CRLs use during secret-key revocation process and not use in the authentication process.

Table 5.1: Notation of EMAP[10] Scheme

Notation	Description
$\mathbb{G}_1$	Additive Group Of Prime Order q
$\mathbb{G}_2$	Multiplicative Group Of Same Order
P, Q	EKI KeyPair Public Key (P) and Private Key (Q) Generator $\in \mathbb{G}_1$
e	Bilinear Mapping
$a \parallel b$	Concatenation of a and b
PK_u^i	i_{th} PKI public key of OBU_u
SK_u^i	i_{th} PKI private key of OBU_u
PID_u^i	Pseudo ID (PID) of OBU_u
s	Master PKI private key of TA
P_o	Master PKI public key of TA
$sig_b(a) = sH(a)$	Signature of a with b secret key s
$CERT_u$	Anonymous certificate of TA for OBU
C	OBU anonymous certificates size
j	Hash chain size
m	EKI key pair size of OBU
l	EKI key pair pool size
$h(.)\ and\ H(.)$	Hash functions for a hash chain
$v \in \mathbb{Z}_q^*$	Hash Chain Initial Secret Value
V	Hash Chain Set
K_i^-	EKI, the private key of OBU
K_i^+	EKI, the public key of OBU
K_g	Revocation check key

U_s	Private key pool
U_p	Public key pool
RS_u	Private keys deployed to OBU_u
RP_u	Public keys deployed to OBU_u
$\mathcal{M}$	Message between OBUs
T_{stamp}	Current timestamp value
$HMAC(a, b \parallel c)$	HMAC of the concatenation of b and c with key a.
REV_{check}	Revocation check value
M	Valid EMAP private key ID for OBU
K_{im}	Intermediate key to update K_g
$\widetilde{K}_g$	EMAP revocation check new key
ver	Revocation version
ver_{last}	Last received revocation version
$ver_{\vert missed}$	Missed revocation version list
v_{j-ver}	Hash chain value of revocation version ver.
$ID_{rev_{key}}$	Identities of the revoked key pairs
$ID_{rev_{key\vert missed}}$	Identities of the missed revoked key pairs
$enc_a(b)$	AES encryption of b with a as the key
K_{msg}	TA Key Update Message
CRL	Revoked certificates PID list for PKI.
REV_{msg}	TA Signed Revocation Message
$\widetilde{K}_i^-$	EMAP New Private Key
$\widetilde{K}_i^+$	EMAP New Public Key

Table 5.2: Protocol Parameter Settings at EMAP[10] Scheme for Secp256k Curve

Parameters	**Settings and Description**
P, Q	ECC Base points, 64 bytes
PK_u^i	ECC Public Key, 64 bytes
SK_u^i	ECC Private Key, 32 bytes
PID_u^i, M	Pseudo ID and Valid EMAP key ID, 8 bytes
s	ECC Private key, 32 bytes
P_o	ECC public key, 64 bytes
$sig_b(a) = sH(a)$	ECC Signature, 64 bytes
$CERT_u$	ECC Anonymous Certificate, 104 bytes
C	OBU anonymous certificates size variable
j	Hash chain size variable
m	EMAP key pair size of OBU variable
l	EMAP key-pair pool size variable
$h(.)\ and\ H(.)$	HMAC or SHA-1 output 20 bytes and AES-CBC-MAC 16 bytes.
$v\ \in\ \mathbb{Z}_q^*$	20 bytes SHA-1 digest
V	Hash Chain Set size variable
$K_i^-, \widetilde{K}_i^-$	ECC Private keys, 32 bytes
$K_i^+, \widetilde{K}_i^+$	ECC Public keys, 64 bytes
$K_g, \widetilde{K}_g$	64 bytes, ($[K_g.x]^{128}$ is AES key)
K_{im}	ECC Intermediate key, 64 bytes

U_s, U_p	Private and Public key pool size l
RS_u, RP_u	Deployed EMAP key pairs, size m
$\mathcal{M}$	The message between OBUs 32 bytes
T_{stamp}	Timestamp value 8 bytes
$HMAC(a, b \parallel c)$	HMAC output 20 bytes and AES-CBC-MAC 16 bytes.
REV_{check}	Revocation check value depends on algorithm HMAC output 20 bytes and AES-CBC-MAC 16 bytes.
ver, ver_{last}, $ver_{\|missed}$	Revocation versions, 4 bytes
v_{j-ver}	SHA-1 digest 20 bytes.
$ID_{rev_{key}}$, $ID_{rev_{key\|missed}}$	Depends on missed revocations
$enc_a(b)$	128-bit input, 128-bit key ,and 128-bit output are AES encryption functions
K_{msg}	Depends on revoked EMAP key size
CRLs	n*8 bytes | n = revoked certificate ID
REV_{msg}	Depends on revoked EMAP key size

EMAP uses asymmetric and symmetric cryptographic operations. Also, use linear and binary search methods for CRLs checking. For the Crypto++ [78] adopted C++ simulation of EMAP and ESAR with the hardware and software environments listed in Table 5.3.

Table 5.3: Simulation and Test Environment

Hardware Environment	
CPU	Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz (8 CPUs)
RAM	16211 MB
SSD	Samsung PM961 Polaris 512GB M.2 NGFF PCIe Gen3 x 4, NVME Solid state drive SSD
Software Environment	
OS	Windows 10 Home 64 bits (10.0 Build 17134)
Crypto++	5.05.2002
Dev. IDE	Visual Studio 2017 version 15.9.1
VC++ Version	14.16

In our simulation, we simulated all process units in EMAP and the same codebase used for ESAR scheme to compare results. Hash chain generated with SHA-1. Revocation check value calculated with H-MAC or AES-CBC-MAC with 128 bits secret key value. EMAP process, CRLs search operation as file searching operation but test environment memory access or read speed not considered that has a significant effect on search algorithms. EMAP search for concatenated PIDs in the text file with linear and binary search methods. Binary search requires a sorted list and EMAP does not define a sorting algorithm for PIDs and the unsorted case for linear search is not measured. We use in-memory data linear and binary searches [79] to ignore the read speed of file locations. 20000 PIDs storage cost is only 156.25 kilobytes, and this can be loaded too fast memory during boot, for fast search operations. We use quicksort [79] to sort PIDs for binary search. Additionally, Our research simulates the linear search unsorted case. Moreover, our simulation compares the revocation check and hash chain operation performances, shown in

Figure 5.2. This comparison results show that EMAP proposed H-MAC verification method for revocation check is nearly 12 times and AES-CBC-MAC 26 times faster than the fastest search method. These results are close to the EMAP reported results.

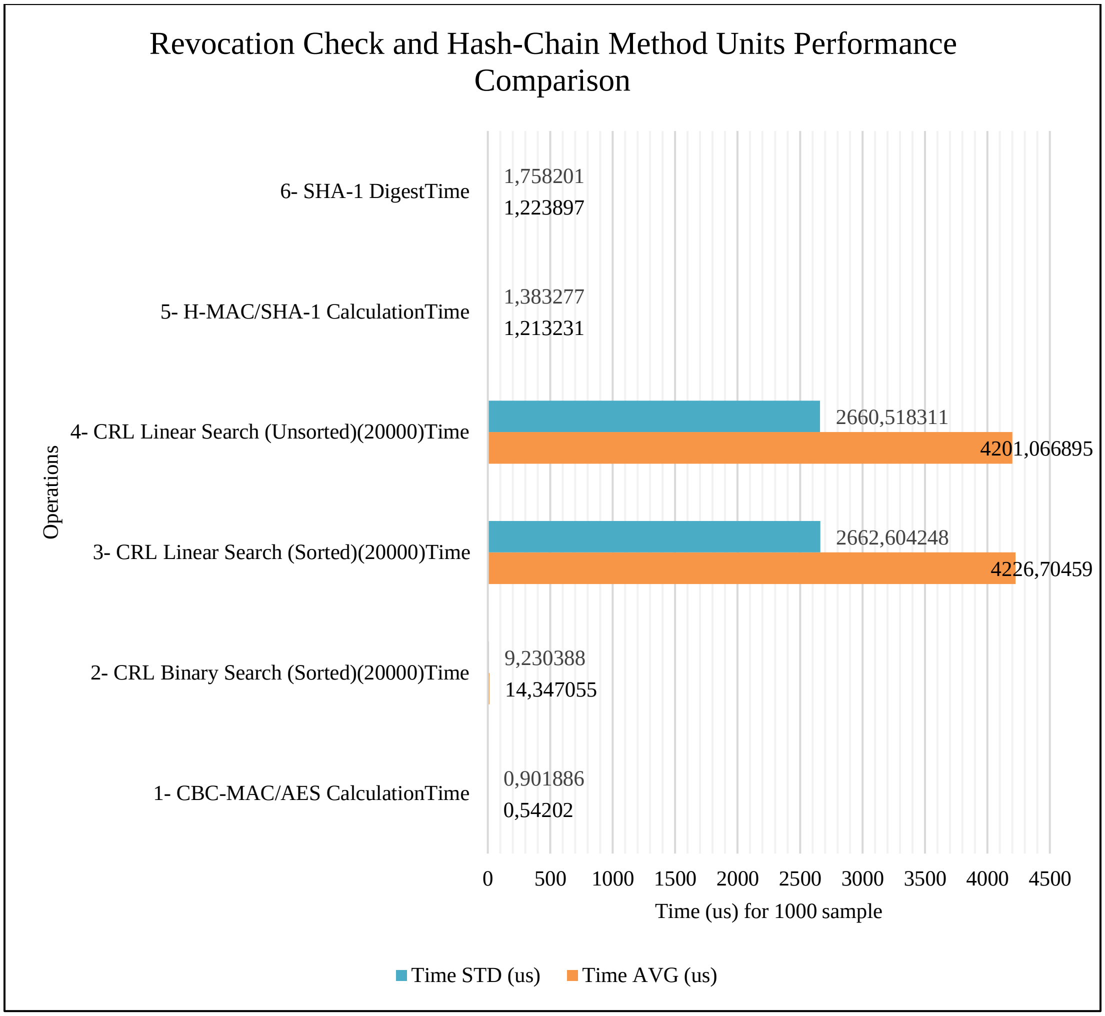


Figure 5.2: Revocation Check and Hash-Chain Method Units Performance Comparison

EMAP does not specify ECC curve for the sake of algorithm generality, for this reason, we implemented configurable simulation but measurements configured for the Secp256k1 curve with SHA-1 usage. ECC key generation, certificate sign, and verification, message signing and verification, ECC point multiplication performances measured besides revocation check key ECIES encryption and decryption. Rekeying hash chain value AES-CBC decryption and EMAP key pair update operation performances measured and compared each other in Figure 5.3.

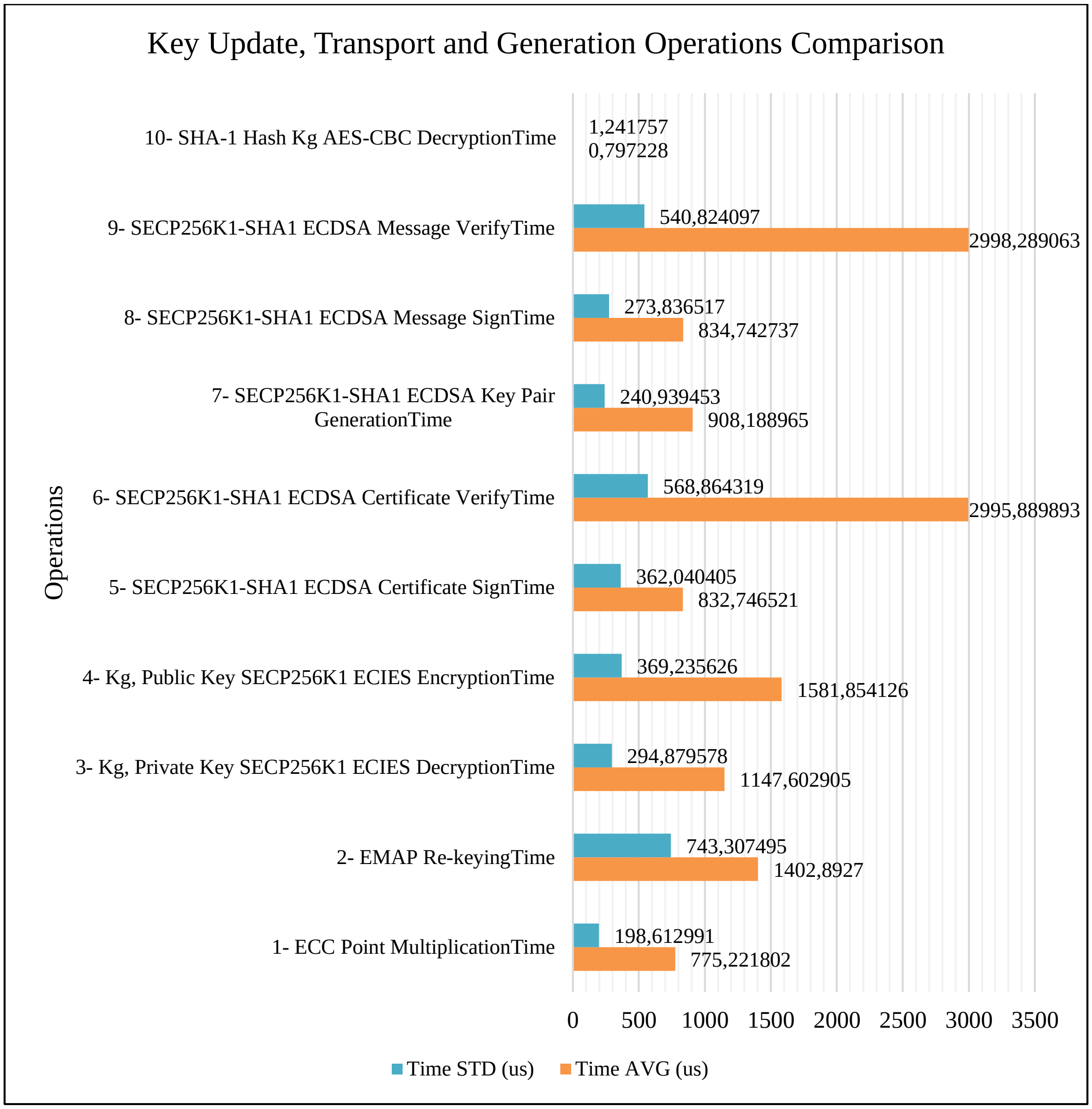


Figure 5.3: Key Update, Transport, and Generation Operations Comparison

Figure 5.3 shows that signature verification is that processed during authentication is the most costly operation and in [10] symmetric operations are faster than asymmetric operations. The following subsections include the entire phase analysis of the EMAP scheme.

5.1 EMAP [10] SYSTEM INITIALIZATION PHASE

System initialization is in Figure 5.4 and Figure 5.5. During the system initialization operation, the TA server is select two generator $P, Q \in \mathbb{G}_1$of subgroup order $q. P, Q$ are base points for elliptic curves. Elliptic curves formally defined as $6 - tuple(p, a, b, g, n, h)$ those described in Table 5.4. In our simulation, we used Secp256k1-SHA1 parameters for EMAP and ESAR schemes.

Table 5.4: Elliptic Curve Tuple Parameters

Elliptic Curve Parameter	**Description**
p	Field Characteristic
$a\ and\ b$	Curve Coefficients
n	Base Point
q	Sub Group Order
h	Sub Group Cofactor

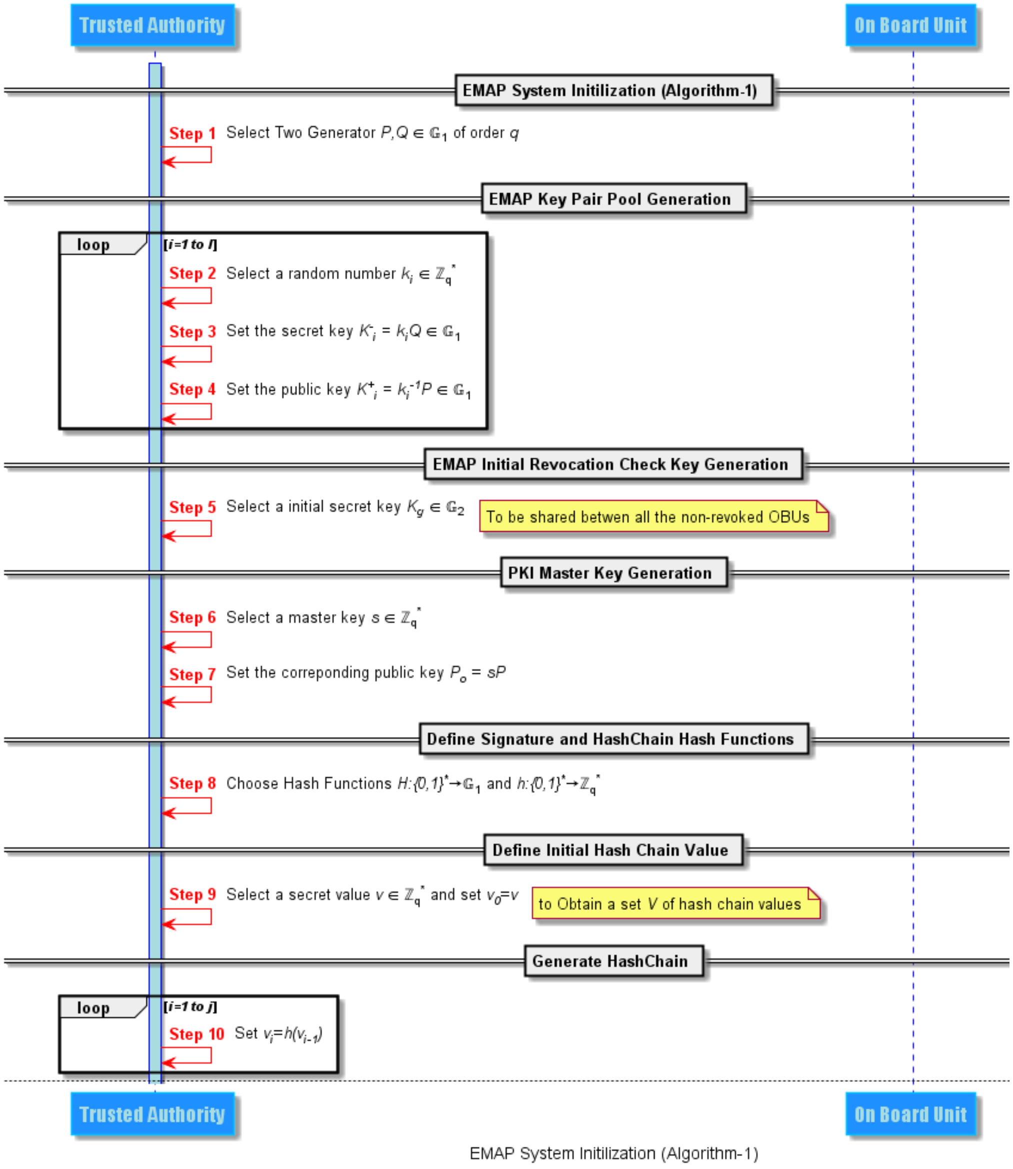

Trusted Authority
On Board Unit
EMAP System Initilization (Algorithm-1)
Step 1 Select Two Generator $P,Q \in \mathbb{G}_1$ of order q
EMAP Key Pair Pool Generation
loop
[i=1 to l]
Step 2 Select a random number $k_i \in \mathbb{Z}_q^*$
Step 3 Set the secret key $K^-_i = k_iQ \in \mathbb{G}_1$
Step 4 Set the public key $K^+_i = k_i^{-1}P \in \mathbb{G}_1$
EMAP Initial Revocation Check Key Generation
Step 5 Select a initial secret key $K_g \in \mathbb{G}_2$
To be shared betwen all the non-revoked OBUs
PKI Master Key Generation
Step 6 Select a master key $s \in \mathbb{Z}_q^*$
Step 7 Set the correponding public key $P_o = sP$
Define Signature and HashChain Hash Functions
Step 8 Choose Hash Functions $H:\{0,1\}^* \rightarrow \mathbb{G}_1$ and $h:\{0,1\}^* \rightarrow \mathbb{Z}_q^*$
Define Initial Hash Chain Value
Step 9 Select a secret value $v \in \mathbb{Z}_q^*$ and set $v_0=v$
to Obtain a set V of hash chain values
Generate HashChain
loop
[i=1 to j]
Step 10 Set $v_i=h(v_{i-1})$
Trusted Authority
On Board Unit
EMAP System Initilization (Algorithm-1)

Figure 5.4: EMAP System Initialization Part-1

TA serves a key pair pool. For each key pair select an arbitrary number $k \in \mathbb{Z}_q^*$ and calculate the secret key with group multiplication as follows:

$$K^- = kQ \tag{5.1}$$

Moreover, the same public key as follows:

$$K^{+} = \left(\frac{1}{k}\right)P \tag{5.2}$$

With group multiplication of the inverse module and store this key pair as follows:

$$\{K^{+},\ K^{-}\} \tag{5.3}$$

After this method TA will have a private key pool, thus:

$$U_s = \{K_i^{-} =\ k_i Q\ |1 \le i \le l\} \tag{5.4}$$

Also, public key pool as follow:

$$U_p = \{K_i^{+} = (\frac{1}{k_i})P|1 \le i \le l\} \tag{5.5}$$

First revocation check $K_g \in\ \mathbb{G}_2$ randomly set and upload to OBU's HSM and this initial upload protection is not thought. PKI scheme master secret key $s\ \in\ \mathbb{Z}_q^*$ randomly set and corresponding public key computed as follows:

$$P_o = sP \tag{5.6}$$

Moreover, TA anonymous certificates created with $\{P_o, s\}$ keypair. TA describes two has function one, thus:

$$h\colon \{0,1\}^* \to\ \mathbb{Z}_q^* \tag{5.7}$$

For hash chain creation and another one and following is for signature production and verification, thus:

$$H\colon \{0,1\}^* \to\ \mathbb{G}_1 \tag{5.8}$$

TA select a hidden $v\ \in\ \mathbb{Z}_q^*$ value for initial hash chain value and compute hash chain as follows:

$$v_0 = v, v_i\ = h(v_{i-1})\forall\ 1 \le i \le j \tag{5.9}$$

Also, store hash chain set as follow for rekeying and revocation, thus:

$$V = \{v_i \,| 0 \leq i \leq j\} \tag{5.10}$$

Secret v value is a critical sensitive asset that attacker can expose all hash chain. This attack caused to recover keys for authentication by revoked OBUs, which have initial hash chain value. After the hash chain generation operation, TA randomly chooses $\{K^{+},\ K^{-}\}$ key pair set which size is m and upload to OBU's HSM device. After this procedure, each OBU will have private keys as follows:

$$RS_u \ \subset \ U_s, |RS_u| = m \tag{5.11}$$

Also, public keys as follows:

$$RP_u \ \subset \ U_p \,, |RP_u| = m \tag{5.12}$$

In this situation, the same key pair can be shared with many OBUs and secure key upload defense not recognized by EMAP. TA create anonymous certificate set for each OBU as with PKI scheme master secret key $s \ \in \ \mathbb{Z}_q^*$ as follows:

$$CERT_u = \{cert_u^i(PID_u^i,\ PK_u^i, sig_{TA}(PID_u^i \ \| \ PK_u^i))\} \tag{5.13}$$

Moreover, upload to HSM for privacy preserving and each OBU can have various PID that associated with real IDs for TA. Finally, H, h, P, Q, and P_o distributed to the OBUs by TA.

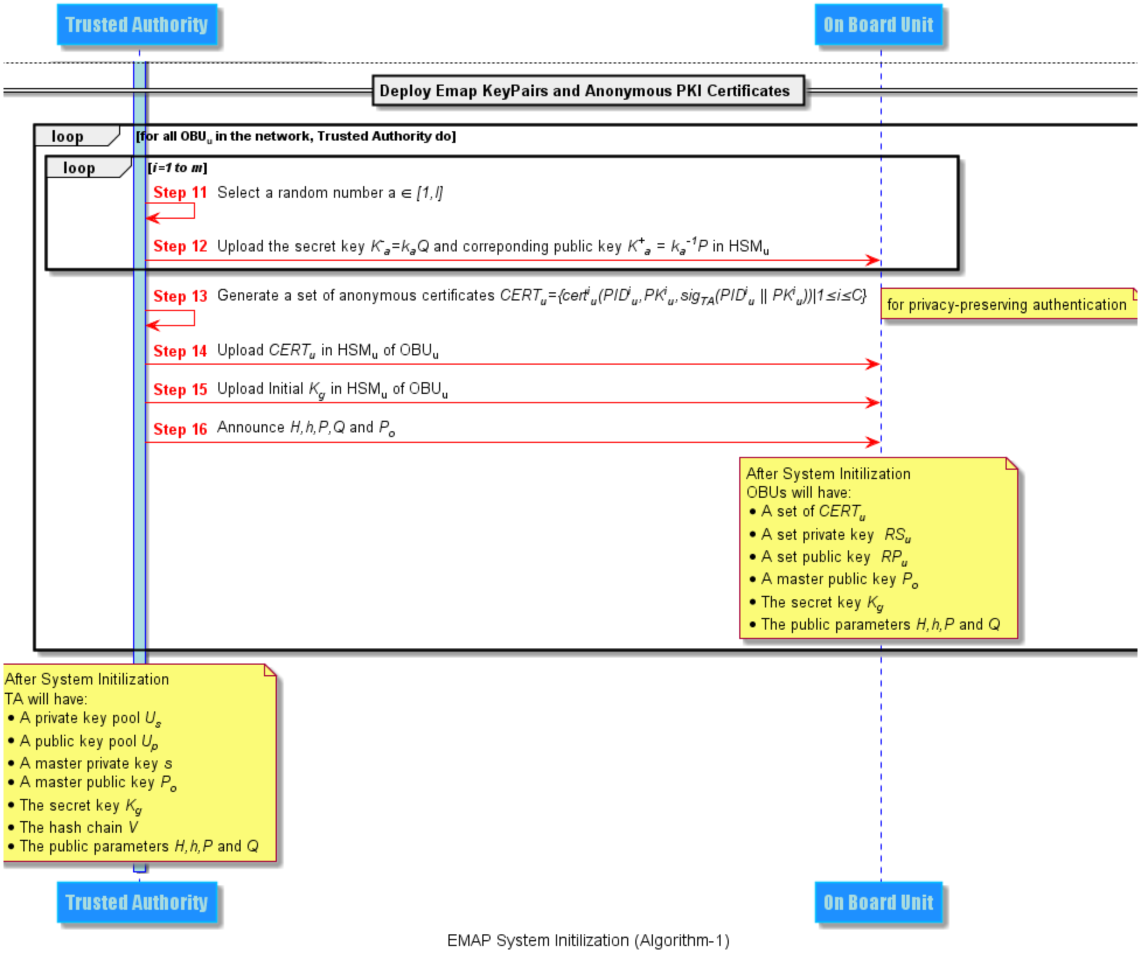

Trusted Authority
On Board Unit
Deploy Emap KeyPairs and Anonymous PKI Certificates
loop
[for all OBU_u in the network, Trusted Authority do]
loop
[i=1 to m]
Step 11 Select a random number a ∈ [1,l]
Step 12 Upload the secret key K^-_a=k_aQ and correpondi ng public key K^+_a = $k_a^{-1}P$ in HSM_u
Step 13 Generate a set of anonymous certificates $CERT_u$={$cert^i_u$(PID^i_u,PK^i_u,sig_{TA}(PID^i_u ‖ PK^i_u))|1≤i≤C}
for privacy-preserving authentication
Step 14 Upload $CERT_u$ in HSM_u of OBU_u
Step 15 Upload Initial K_g in HSM_u of OBU_u
Step 16 Announce H,h,P,Q and P_o
After System Initilization
OBUs will have:
• A set of $CERT_u$
• A set private key RS_u
• A set public key RP_u
• A master public key P_o
• The secret key K_g
• The public parameters H,h,P and Q
After System Initilization
TA will have:
• A private key pool U_s
• A public key pool U_p
• A master private key s
• A master public key P_o
• The secret key K_g
• The hash chain V
• The public parameters H,h,P and Q
Trusted Authority
On Board Unit
EMAP System Initilization (Algorithm-1)

Figure 5.5: EMAP System Initialization Part-2

5.2 EMAP [10] OBU TO OBU AUTHENTICATION PHASE

Authentication fired with message signing by sender OBU and end with message verification by target OBU in Figure 5.6. Sender OBU first determines revocation check value that proposed by EMAP as follows:

$$REV_{check} = HMAC(K_g, PID_u \parallel T_{stamp}) \tag{5.14}$$

moreover, provide a signed message sequence of telematics message, timestamp, TA certificate, message signature, and revocation check value as follows:

$$(\mathcal{M} \parallel T_{stamp} \parallel cert_u(PID_u, PK_u, sig_{TA}(PID_u^i \parallel PK_u^i)) \parallel sig_u(\mathcal{M} \parallel T_{stamp}) \parallel REV_{check}) \tag{5.15}$$

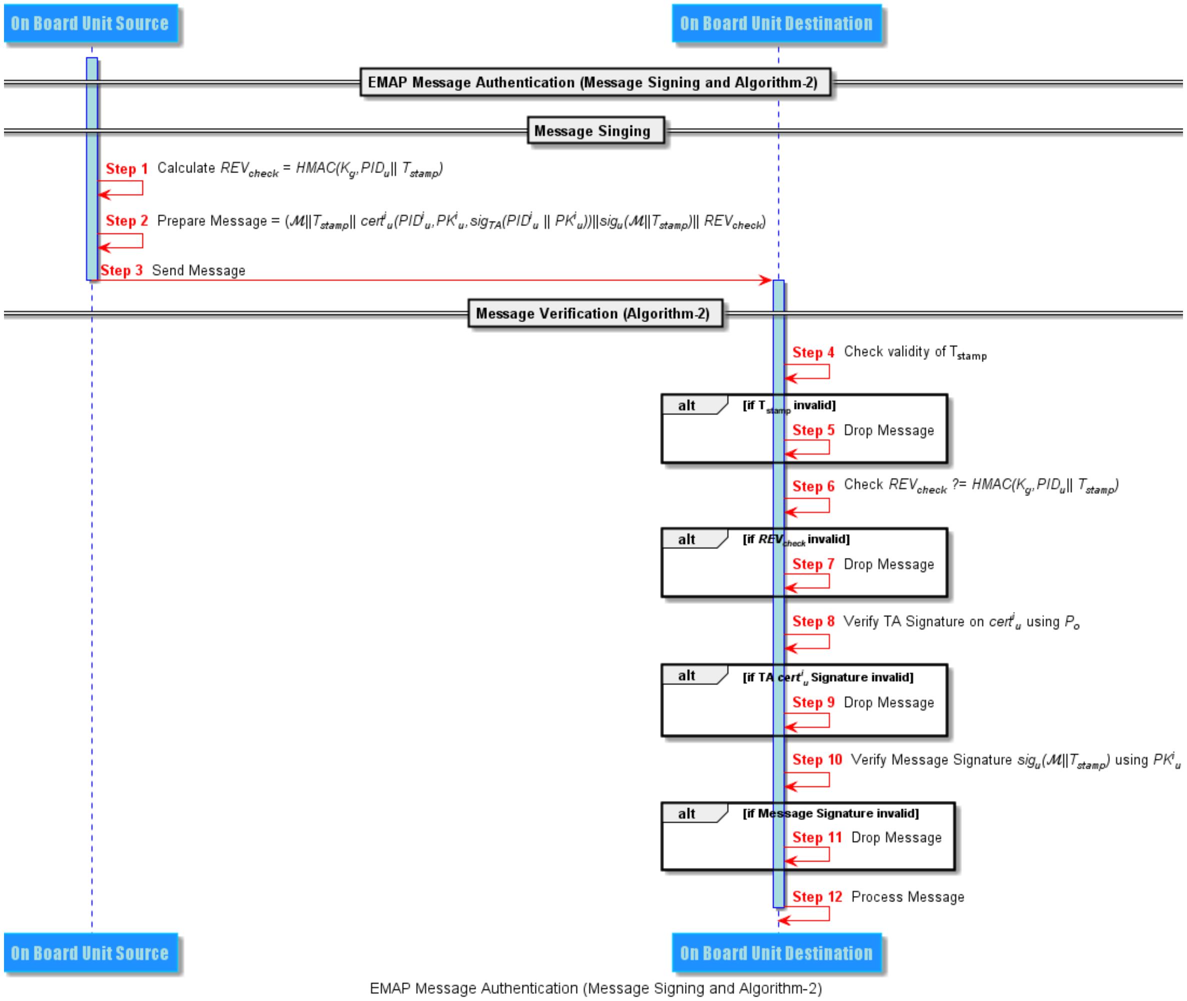

On Board Unit Source
On Board Unit Destination
EMAP Message Authentication (Message Signing and Algorithm-2)
Message Singing
Step 1 Calculate $REV_{check} = HMAC(K_g, PID_u \parallel T_{stamp})$
Step 2 Prepare Message = $(\mathcal{M} \parallel T_{stamp} \parallel cert^i_u(PID^i_u, PK^i_u, sig_{TA}(PID^i_u \parallel PK^i_u)) \parallel sig_u(\mathcal{M} \parallel T_{stamp}) \parallel REV_{check})$
Step 3 Send Message
Message Verification (Algorithm-2)
Step 4 Check validity of T_{stamp}
alt
[if T_{stamp} invalid]
Step 5 Drop Message
Step 6 Check REV_{check} ?= $HMAC(K_g, PID_u \parallel T_{stamp})$
alt
[if REV_{check} invalid]
Step 7 Drop Message
Step 8 Verify TA Signature on $cert^i_u$ using P_o
alt
[if TA $cert^i_u$ Signature invalid]
Step 9 Drop Message
Step 10 Verify Message Signature $sig_u(\mathcal{M} \parallel T_{stamp})$ using PK^i_u
alt
[if Message Signature invalid]
Step 11 Drop Message
Step 12 Process Message
On Board Unit Source
On Board Unit Destination
EMAP Message Authentication (Message Signing and Algorithm-2)

Figure 5.6: EMAP OBU to OBU Authentication

Note that OBU_u message signature $sig_u(\mathcal{M} \parallel T_{stamp})$ equal to $SK_u^i H(\mathcal{M} \parallel T_{stamp})$ and TA certificate signature $sig_{TA}(PID_u^i \parallel PK_u^i)$ equal to $sH(PID_u^i \parallel PK_u^i)$. Receiving OBU manner following directions and in any wrong case drop message. First, check T_{stamp} is valid else drop the message. In WAVE [12], [13] recommended timeout is 300 msec. Then check REV_{check} is correct by calculating, thus:

$$HMAC\left(K_g, PID_u \parallel T_{stamp}\right) \tag{5.16}$$

Else dismiss the message. Then verify TA signature by processing as follows:

$$\begin{aligned} e\left(sig_{TA}\left(PID_u^i \parallel PK_u^i\right), P\right) &= e\left(sH\left(PID_u^i \parallel PK_u^i\right), P\right) \\ &= e\left(H\left(PID_u^i \parallel PK_u^i\right), sP\right) = e\left(H\left(PID_u^i \parallel PK_u^i\right), P_o\right) \end{aligned} \tag{5.17}$$

If not valid then drop the message. If the TA certificate is correct, then use certificate OBU public key and validate the message signature as follows:

$$\begin{aligned} e\left(sig_u(\mathcal{M} \parallel T_{stamp}), P\right) &= e\left(SK_u^i H\left(\mathcal{M} \parallel T_{stamp}\right), P\right) \\ &= e\left(H\left(\mathcal{M} \parallel T_{stamp}\right), SK_u^i P\right) = e\left(H\left(\mathcal{M} \parallel T_{stamp}\right), PK_u^i\right) \end{aligned} \tag{5.18}$$

Finally, if the signature is valid then process the information.

5.3 EMAP [10] REVOCATION PHASE

Revocation process originated by TA when any OBU_u revoked in Figure 5.7, Figure 5.8 and Figure 5.9. TA choose a valid EMAP key pair M from its database and select a random $t \in \mathbb{Z}_q^*$ and determine the intermediate key as follows:

$$K_{im} = tK_M^+ = \frac{t}{k_M} P \in \mathbb{G}_1 \tag{5.19}$$

Moreover, new revocation key $\widetilde{K}_g$ as follows:

$$\widetilde{K}_g = e(K_M^-, K_{im}) = e\left(k_M Q, \frac{t}{k_M} P\right) = e(Q, P)^{k_M \cdot \frac{t}{k_M}} = e(Q, P)^t \in \mathbb{G}_2, t \in \mathbb{Z}_q^* \tag{5.20}$$

If receiver OBU has key pair M, then it can recover new revocation key $\widetilde{K}_g$ with the same method above. Also, TA gets the current version (ver) hash chain value v_{j-ver} from $V = \{v_i \mid 0 \le i \le j\}$ and encrypt with $\widetilde{K}_g$ as follows:

$$enc_{\widetilde{K}_g}(v_{j-ver}) \tag{5.21}$$

All update required EMAP key ID list store on $ID_{rev_{key}}$. TA concatenate these parameters and create a key update message as follows:

$$K_{msg} = \left(ver \ \| \ M \ \| \ ID_{rev_{key}} \ \| \ K_{im} \ \| \ enc_{\widetilde{K}_g}(v_{j-ver})\right) \tag{5.22}$$

Additionally, TA provides a PKI scheme CRLs and develops signed revocation message as follows:

$$REV_{msg} = \left(CRL \ \| \ K_{msg} \ \| \ sig_{TA}(CRL \ \| \ K_{msg})\right) \tag{5.23}$$

Moreover, broadcast message to the network.

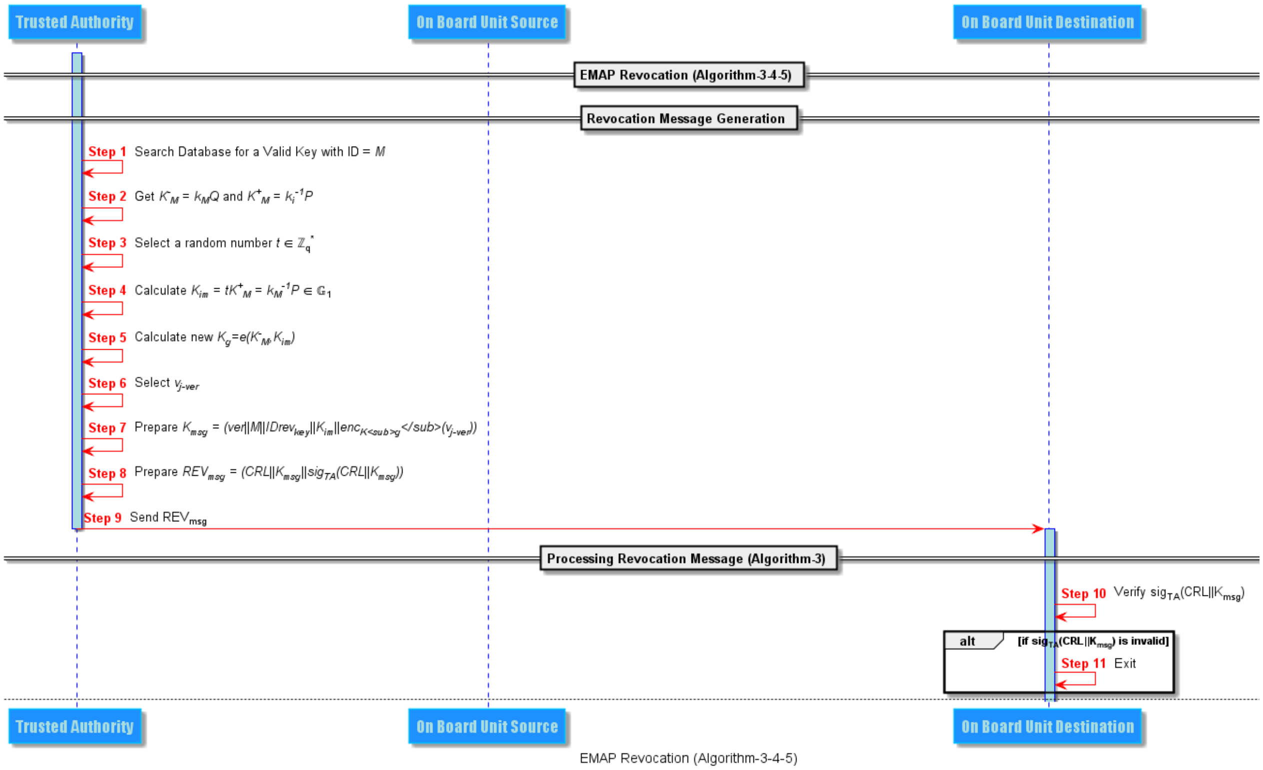

Trusted Authority
On Board Unit Source
On Board Unit Destination
EMAP Revocation (Algorithm-3-4-5)
Revocation Message Generation
Step 1 Search Database for a Valid Key with ID = M
Step 2 Get $K^-_M = k_M Q$ and $K^+_M = k_i^{-1} P$
Step 3 Select a random number $t \in \mathbb{Z}_q^*$
Step 4 Calculate $K_{im} = tK^+_M = k_M^{-1} P \in \mathbb{G}_1$
Step 5 Calculate new $K_g = e(K^-_M, K_{im})$
Step 6 Select $v_{j\text{-}ver}$
Step 7 Prepare K_{msg} = $(ver||M||IDrev_{key}||K_{im}||enc_{K_{g}$}$(v_{j\text{-}ver}))$
Step 8 Prepare $REV_{msg} = (CRL||K_{msg}||sig_{TA}(CRL||K_{msg}))$
Step 9 Send REV_{msg}
Processing Revocation Message (Algorithm-3)
Step 10 Verify $sig_{TA}(CRL||K_{msg})$
alt
[if $sig_{TA}(CRL||K_{msg})$ is invalid]
Step 11 Exit
Trusted Authority
On Board Unit Source
On Board Unit Destination
EMAP Revocation (Algorithm-3-4-5)

Figure 5.7: EMAP Revocation Part-1

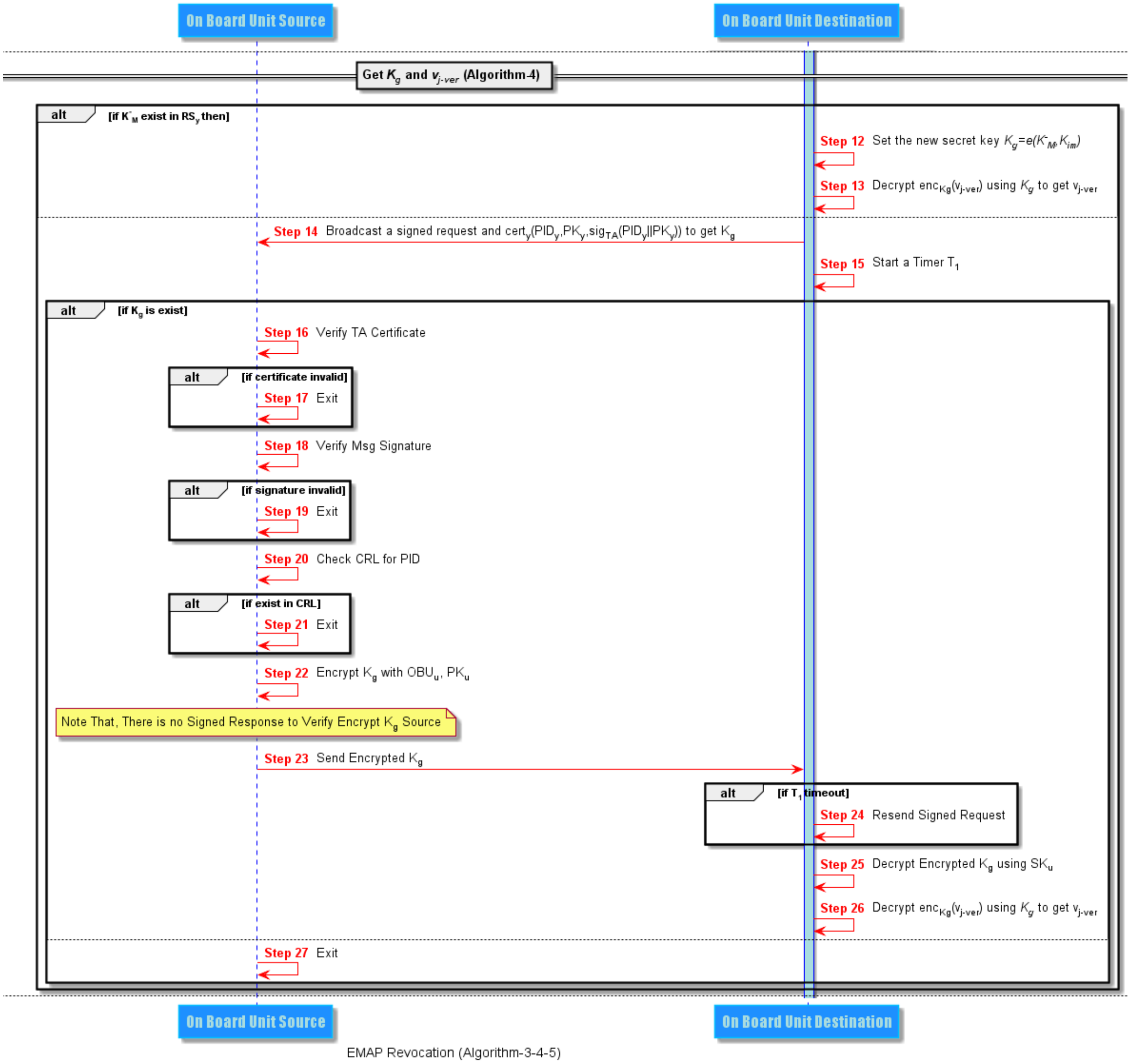
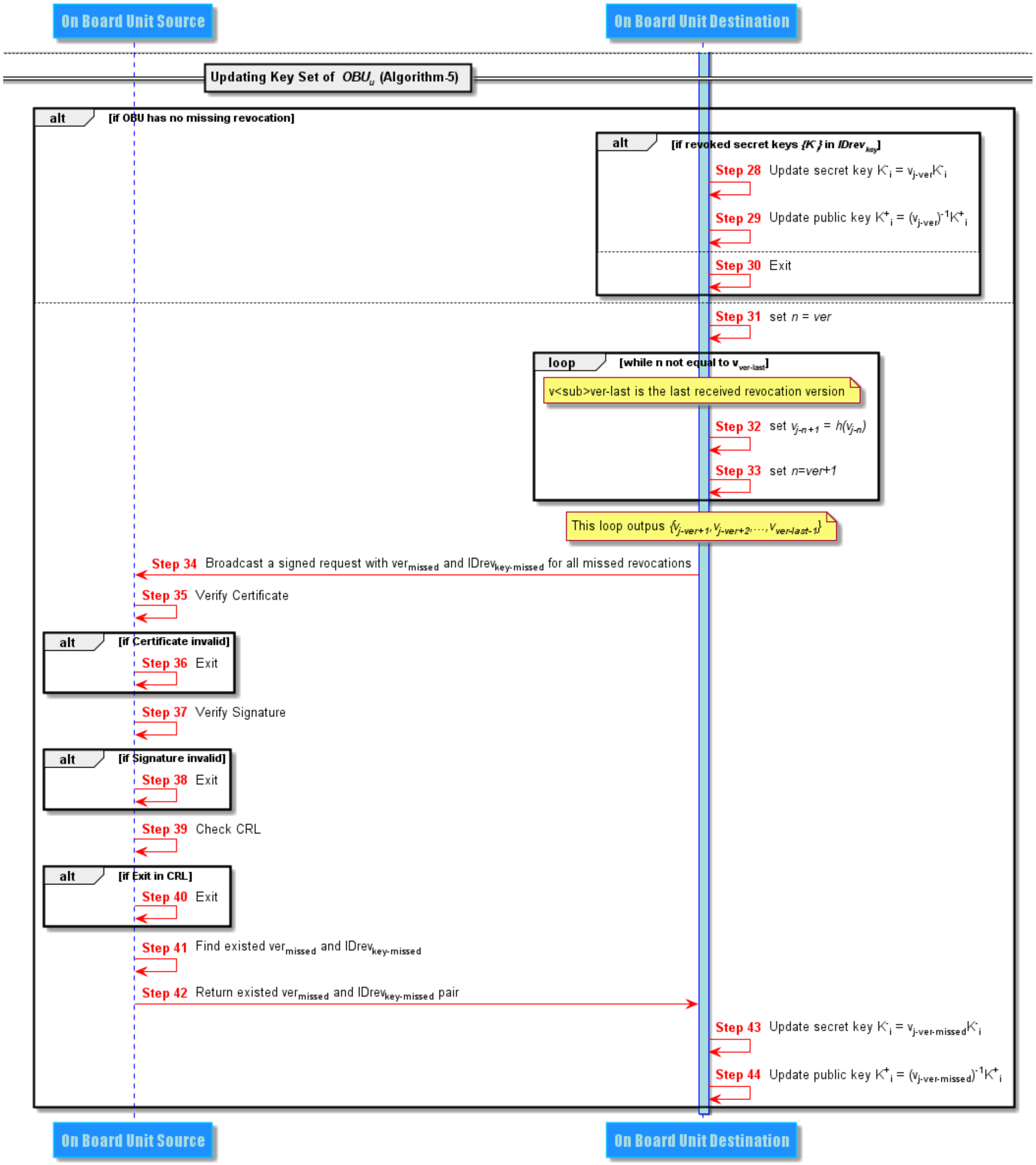

On Board Unit Source
On Board Unit Destination
Get K_g and $v_{j\text{-}ver}$ (Algorithm-4)
alt [if $K^{`}_M$ exist in RS_y then]
Step 12 Set the new secret key $K_g=e(K^{`}_M, K_{im})$
Step 13 Decrypt $enc_{Kg}(v_{j\text{-}ver})$ using K_g to get $v_{j\text{-}ver}$
Step 14 Broadcast a signed request and $cert_y(PID_y,PK_y,sig_{TA}(PID_y||PK_y))$ to get K_g
Step 15 Start a Timer T_1
alt [if K_g is exist]
Step 16 Verify TA Certificate
alt [if certificate invalid]
Step 17 Exit
Step 18 Verify Msg Signature
alt [if signature invalid]
Step 19 Exit
Step 20 Check CRL for PID
alt [if exist in CRL]
Step 21 Exit
Step 22 Encrypt K_g with OBU_u, PK_u
Note That, There is no Signed Response to Verify Encrypt K_g Source
Step 23 Send Encrypted K_g
alt [if T_1 timeout]
Step 24 Resend Signed Request
Step 25 Decrypt Encrypted K_g using SK_u
Step 26 Decrypt $enc_{Kg}(v_{j\text{-}ver})$ using K_g to get $v_{j\text{-}ver}$
Step 27 Exit
On Board Unit Source
On Board Unit Destination
EMAP Revocation (Algorithm-3-4-5)

Figure 5.8: EMAP Revocation Part-2


On Board Unit Source
On Board Unit Destination
Updating Key Set of OBU_u (Algorithm-5)
alt [if OBU has no missing revocation]
alt [if revoked secret keys $\{K^-\}$ in $IDrev_{key}$]
Step 28 Update secret key $K^-_i = v_{j-ver}K^-_i$
Step 29 Update public key $K^+_i = (v_{j-ver})^{-1}K^+_i$
Step 30 Exit
Step 31 set $n = ver$
loop [while n not equal to $v_{ver-last}$]
v<sub>ver-last is the last received revocation version
Step 32 set $v_{j-n+1} = h(v_{j-n})$
Step 33 set $n=ver+1$
This loop outpus $\{v_{j-ver+1}, v_{j-ver+2}, \dots, v_{ver-last-1}\}$
Step 34 Broadcast a signed request with ver_{missed} and $IDrev_{key-missed}$ for all missed revocations
Step 35 Verify Certificate
alt [if Certificate invalid]
Step 36 Exit
Step 37 Verify Signature
alt [if Signature invalid]
Step 38 Exit
Step 39 Check CRL
alt [if Exit in CRL]
Step 40 Exit
Step 41 Find existed ver_{missed} and $IDrev_{key-missed}$
Step 42 Return existed ver_{missed} and $IDrev_{key-missed}$ pair
Step 43 Update secret key $K^-_i = v_{j-ver-missed}K^-_i$
Step 44 Update public key $K^+_i = (v_{j-ver-missed})^{-1}K^+_i$
On Board Unit Source
On Board Unit Destination

EMAP Revocation (Algorithm-3-4-5)

Figure 5.9: EMAP Revocation Part-3

When OBU gets revocation message, first verifies TA signature $sig_{TA}(CRL \parallel K_{msg})$ and as a next move is processed to get $\widetilde{K}_g$ and v_{j-ver}. If OBU has key pair M then easily calculates $\widetilde{K}_g$ and decrypt $enc_{\widetilde{K}_g}(v_{j-ver})$ to get plain v_{j-ver}. If key pair M is not existed then OBU distribute a signed request that not described in EMAP but includes the anonymous certificate of OBU, thus:

$$cert_u^i\left(PID_u^i, PK_u^i, sig_{TA}\left(PID_u^i \parallel PK_u^i\right)\right) \tag{5.24}$$

In a loop. If receiver OBU has $\widetilde{K}_g$, then verify the certificate and check PID is in CRLs. If the certificate is not valid or PID exists in CRL then reject the request. Else, use PK_u^i to encrypt $\widetilde{K}_g$ and send target OBU. Encryption method not described. However, we use ECIES for simulation. Also, there is no method provided to check receiver OBU has $\widetilde{K}_g$ revocation check. Key requested OBU decrypt encrypted $\widetilde{K}_g$ with own secret key SK_u^i and use recovered $\widetilde{K}_g$ to decrypt $enc_{\widetilde{K}_g}(v_{j-ver})$ to get plain v_{j-ver}. Please note that encrypted $\widetilde{K}_g$ is not signed and wrong key recovery can corrupt plain $enc_{\widetilde{K}_g}(v_{j-ver})$ decryption. If everything accomplished until this step non-revoked OBUs would have $\widetilde{K}_g$ and v_{j-ver}, and the next step will cover key set update for OBU.

If OBU is not missed any revocation message and has one of EMAP key pair ID in $ID_{rev_{key}}$, then update related keys as follows:

$$\widetilde{K}_i^- = v_{j-ver}K_i^-, \widetilde{K}_i^+ = (\frac{1}{v_{j-ver}})K_i^+ \tag{5.25}$$

Else if EMAP key pair ID does not exist in $ID_{rev_{key}}$ then end operation. Otherwise, if OBU missed any revocation then calculate missing hash chain values from v_{j-ver}. Also, prepare a signed request to get revoked versions missing key IDs as $ID_{rev_{key|\,missed}}$. Receiver OBU verifies sender certificate, and message signature also controls CRL for $ID_{rev_{key|missed}}$ request. If everything flourishing then returns own already updated key ID list as $ID_{rev_{key|missed}}$ When OBU receives $ID_{rev_{key|missed}}$ then find $v_{j-ver_{missed}}$ from the calculated list and update keys as follows:

$$\widetilde{K}_i^- = v_{j-ver_{missed}}K_i^- \ \widetilde{K}_i^+ = (\frac{1}{v_{j-ver_{missed}}})K_i^+ \tag{5.26}$$

After this procedure revocation achieved for each OBU and each OBU should store received revocation version (ver) and detailed updated keypair ID list ($ID_{rev_{key}}$).

6. WEAKNESSES OF EMAP [10] VANET SCHEME

During V2V authentication, EMAP substitutes for lengthy CRL testing with fast MAC verification. Scheme [10] claims to be able to withstand the forgery, replay, collude assaults and provide forward privacy. There are however different safety deficiencies in general.

During the revocation process, not verified encrypted $\widetilde{K}_g$ source can disable the receiver OBU forever. This problem occurs on the case when OBU does not have missing revocation but do not have key pair M to calculate $\widetilde{K}_g$. In this case, OBU prepares a signed request to get encrypted $\widetilde{K}_g$ from other OBUs and receiver OBUs check CRLs and if it is valid then encrypt $\widetilde{K}_g$ and send to source OBU without TA and message signature. Receiver OBU decrypts $\widetilde{K}_g$ without signature check and use for key pair update and store this new key value. If the sender is an attacker then can send wrong $\widetilde{K}_g$ and this cause to corrupt EMAP key pairs and also stored $\widetilde{K}_g$, and This OBU cannot authenticate with other OBUs forever, because there is no recovery mechanism defined for this kind of attacks. Besides, this can occur during system failure a broken OBU can send random buffer as encrypted $\widetilde{K}_g$ to other OBUs. Also in EMAP, encrypted $\widetilde{K}_g$ requested OBU can process this request only if it has fresh $\widetilde{K}_g$ But own fresh $\widetilde{K}_g$ detection for OBU is not defined in EMAP. An OBU can assume it has already updated $\widetilde{K}_g$ and send its $\widetilde{K}_g$ to requested OBU and this can corrupt key pairs and revocations secret-key as a timing issue. Missing revocation version check, cause network congestion. If any OBU has missing revocation, it should stop sending and processing telematics messages. Also in ESAR, new $\widetilde{K}_g$ detection mechanism implements with revocation version check. If OBU has, missing revocation then should stop processing encrypted $\widetilde{K}_g$ request. Missing revocation version checking, cause the network congestion, extra power and resource usage on OBUs. System initialization and secure OBU registration not considered. Default K_g, PKI and EMAP keypair injection security not defined. OBUs, which revoked and shared the same keys can authenticate each other. This situation can cause sending wrong telematics each other and create network congestion. OBU traceability is an issue because each OBU has multiple PID for anonymous certificates to provide privacy preserving. During V2V communication, repeated messages include an anonymous certificate, which has different PIDs. Receiver OBU cannot distinguish messages and assume each telematics (location update, warning, ...) come from different OBU. This cause ghost traffic and wrong information for users. TA side EMAP key pair synchronization after revocation not considered. Messages are not distinguishable; there is no message identification for EMAP messages. Also, there is no state management for OBU and TA. In EMAP scheme, a key pair can be shared with multiple OBUs randomly. Single OBU revocation is an issue. In this case, revocation monitoring and maintenance is hard. There is an additional request; encrypted $\widetilde{K}_g$ and $ID_{rev_{key|missed}}$ requests can merge. Therefore, there will be a single sign operation for two requests. Analyses are cover frequently in the authentication phase. Initialization phase analysis not examined. All EMAP scheme builds on SHA-1 hash chain values. After a few revocations, modified OBU can store clear hash values, and a few clear hash value can expose the initial hash chain value. After this issue, This OBU can pass all revocations. Besides, SHA-1 is a weak hash function; NIST stops using SHA-1 on digital signatures [80].

7. PROPOSED M2M SCHEME

This passage includes our suggested plan ,and we have developed an improved scheme to complete the missing pieces of the design of Sun et al [5]. Table 7.1 involves the design notes suggested, and our contributions have been organized as follows.

Table 7.1: Proposed System Notations

Notation	**Description**
k	The M2M server's secret key asset
uid	Mobile device user idendification data
t_uid	Mobile device user temporal identification data
mid	Identification data of M2M servers
pw	User password to authenticate to M2M servers
umk	The secret session key among M2M server and User Mobile Device
hid	Home Gateway ID
t_hid	Temporal Home Gateway ID
u_st	User State
h_st	Home Gateway State
$a \mid\mid b$	concatenation of a and b
$[m]^n$	Most significant n bits of string m
$F_{SHA256}(.)$	Cryptographic SHA256 hash function
$F_{SHA1}(.)$	Cryptographic SHA1 hash function
$F_{e-AES}(a,b)$	Using AES function to cipher a with b as the key
$F_{d-AES}(a,b)$	Using AES function to decipher a with b as the key
$F_{H-MAC}(a,key)$	Keyed Cryptographic Mac and Hash Function
$ecdh\_id$	User ECDH Key Pairing Identifier
key_{ECDH}	ECDH Secret Key
pub_{user}	ECDH User Public Key
$priv_{user}$	ECDH User Private Key
pub_{m2m}	ECDH M2M Server Public Key
$priv_{m2m}$	ECDH M2M Server Private Key
$[pub_x, priv_x] = F_{i-ECDH}()$	ECDH Key Pair Generation for Entity x

$key_{ECDH} = F_{s-ECDH}(pub_{user}, priv_{m2m})$	M2M Server ECDH Secret Key Derivation
$F_{e-ECDH}(a, key_{ECDH})$	ECDH Encryption with input a
$F_{d-ECDH}(a, key_{ECDH})$	ECDH Decryption with input a
key_{enc}	Encryption Key
key_{mac}	MAC Key

7.1 MESSAGE MODIFICATION PROTECTION WITH SHORTENED MAC

Our first addition to the design [5] is to protect the message during transfers; the scheme [5] messages can be changed. No MAC safety check is available. Also, a MAC calculator source must be verified by the recipient item. To a certain case we preferred to use keyed MAC such as DES (Data Encryption Standard) or AES, as C-MAC [81] or H-MAC [82]. This privacy technique is implemented by [8].

We chose to use a reduced MAC as described in the PBA design [53] to check messages according to the performance and resource restrictions. The MAC algorithm stated in [83]. is also used by the GLARM scheme [4]. As described in [84], we used a H-MAC truncated MAC-algorithm, which has the same offset range for HOTP (H-MAC-based One-Time-Password). Each application and response in the public transport provider is secured by network organizations with the MAC and the receiver will drop the request if MAC verification fails. The H-MAC setup performances for SHA-256 (Secure Hash Algorithm 256) and SHA-1 with 16-byte and 8-byte keys were evaluated with MATLAB simulation as shown in Table 7.2.

Table 7.2: Truncated MAC Calculation Performances

Algorithm	Measurement Mean (sn)	Measurement Std. (sn)
H-MAC-SHA256 16 bytes key	0.000273	0.000094
H-MAC-SHA1 16 bytes key	0.000268	0.000189
H-MAC-SHA256 8 bytes key	0.000267	0.000109
H-MAC-SHA1 8 bytes key	0.000283	0.000134
ISO9797-ALGO-3 16 bytes key	0.000392	0.000226

7.1 TEMPORAL ID USAGE FOR PRIVACY-PRESERVING

Secondly, the original user ID is replaced by a temporary ID to protect the privacy as stated in the GLARM scheme [4]. In the GLARM system, the temporary ID and timestamp used in all procedures is obtained using a pre-shared key. [5] concentrates on privacy safety, but they transmit simple user ID and password during password change and enrollment. Also, scheme [5] transmits the original user ID for all stages in the open format. Therefore, data security has a critical problem in the Scheme [5] and is not anonymous. We have a solid way of solving this issue, i.e. ECDH-key pairing during registering. Entities identify key message pairings with the ECDH key identification parameter $ecdh\_id$. That $ecdh\_id$ value is generated with a trimmed SHA 1 function from the customer ID (uid), during the user configuration stage:

$$[ecdh\_id] = F_{SHA-1}(uid)^{n} \tag{7.1}$$

The user transmits the public key (pub_{user}) and $ecdh\_id$ the M2M server according to the registration stage requirements to prevents to reveal uid, as follows:

$$< ECD\_SYN, ecdh\_id, pub_{user} > \tag{7.2}$$

The aim of ECDH secure key sharing, the M2M server uses its private key ($priv_{m2m}$) to receive the public key (pub_{user}) in order to derive encrypt key (key_{ECDH}), thus:

$$key_{ECDH} = F_{s-ECDH}(pub_{user}, priv_{m2m}) \tag{7.3}$$

With key_{ECDH}, the M2M server enables confidentiality and integrity checks. The M2M server sends a single message to the user device with device encrypted key and the M2M server public key (pub_{m2m}), in order to reduce the demand volume of the suggested system, thus:

$$Enc(key_{enc} \parallel key_{mac}) = F_{e-ECDH}(key_{enc} \parallel key_{mac}, key_{ECDH}) \tag{7.4}$$

$$< ECDH\_ACK, ecdh\_id, \mathrm{pub}_{\mathrm{m2m}}, Enc(key_{enc} \parallel key_{mac}) > \tag{7.5}$$

The client receives the public key and calculates the ECDH key for decrypting the key items that have been encoded. Once the keys have been calculated, the user device will prepare the registration request ($REGISTER\_SYN$) by using the registration request MAC and the confidentiality key id ($ecdh\_id$) the user ID (uid), password (pw) and house portal ID (hid), thus:

$$mac_{msg} = \mathrm{F}_{H-MAC}(REGISTER\_SYN \parallel ecdh_{id} \parallel uid \parallel pw \parallel hid, key_{mac}) \tag{7.6}$$

$$enc_{msg} = \mathrm{F}_{e-AES}\left(uid \parallel pw \parallel hid \parallel mac_{msg}, key_{enc}\right) \tag{7.7}$$

$$< REGISTER\_SYN, ecdh\_id, enc_{msg} > \tag{7.8}$$

The issue in [5] is that the request message $REGISTER\_SYN$ contains a transparent format for valuable property. However, we safeguard confidentiality and integrity of these valuable items. When a $REGISTER\_SYN$ message is sent to the M2M server, it is decoded by a key_{ECDH} the function of the ECDH server and the clear user ID is obtained, after which a temporary ID is calculated using a H-MAC based simplified MAC algorithm by the use of the integrity key. The algorithm's output length is the same as that of the original ID. The transient ID calculation process is also presented in the client device and the following applications are implemented via the temporary ID to safeguard network user privacy. Typical resistance to brute force assaults is a shortened MAC algorithm centered on H-MAC. Furthermore, ECDH key pairing protects against MITM assaults and MAC prevents alteration of messages Scheme [5] does not provide these mentioned privacy techniques.

7.1 FORWARD AND BACKWARD CONFIDENTIALITY ENHANCEMENT

The third input relates to privacy improvement. Our approach improved operations with H-MAC algorithm for MAC and Temporal ID. We use ECDH keys to encapsulate security keys to improve forward and backward privacy.

Scheme [5] stages of registering and change of password are not safe. We safe every step we have suggested, however our scheme follows standard ECDH, Trimmed H-MAC, and AES algorithms. During the interaction, these algorithms safeguard critical properties and each technique is resistant to assaults by brute force.

7.2 STATE MANAGEMENT WITH RELIABLE OPERATON PROCESSING

The fourth input is about the reliability of the provider. Scheme [5] does not contain the timeout, message verification section, and message identification section of the procedure list. The state management of users and gateways is big issue, so. One bit is available to save online or offline user status. With an internal unit clock without a timestamp, we introduced a timeout determination. An internal RTC (real time clock) module for accessories does not need this sort of exercise. If there is a timeout while interaction occurs, organizations will withdraw or resend applications. In order to identify eavesdropping assaults by tracking the transaction handling times used during a timeout test. Each message is distinguished with a one-byte tag that allows products to assess calls. Table 7.3 contains the suggested safe message identification systems.

Table 7.3: Proposed M2M Scheme Message Identifiers

Message Name	Message Phase	Message Sender	Message Receiver
ECDH_SYN	User Registration and Key Pair	User	M2M Server
ECDH_ACK	User Registration and Key Pair	M2M Server	User
REGISTER_SYN	User Registration and Key Pair	User	M2M Server
REGISTER_ACK	User Registration and Key Pair	M2M Server	User
KEY_SYN	User Home Gateway Key Injection	User	Home Gateway
KEY_ACK	User Home Gateway Key Injection	Home Gateway	User
JOIN_SYN	Home gateway Join Network	Home Gateway	M2M Server
JOIN_ACK	Home gateway Join Network	M2M Server	Home Gateway
JOIN_OK	Home gateway Join Network	Home Gateway	M2M Server
LOGIN_SYN	User Login and Authentication	User	M2M Server
LOGIN_ACK	User Login and Authentication	M2M Server	User
LOGIN_OK	User Login and Authentication	User	M2M Server
PWD_SYN	User Change Password	User	M2M Server
RE_JOIN	User Change Password	M2M Server	Home Gateway
PWD_ACK	User Change Password	M2M Server	User
PWD_NACK	User Change Password	User	M2M Server
PWD_OK	User Change Password	User	M2M Server
LOGOUT_SYN	M2MServer User Logout	User	M2M Server
LOGOUT_ACK	M2MServer User Logout	M2M Server	User

On top of that, we updated the state machines for the current scheme. Figure 7.1 and Figure 7.2 contains enhanced states in the proposed design.

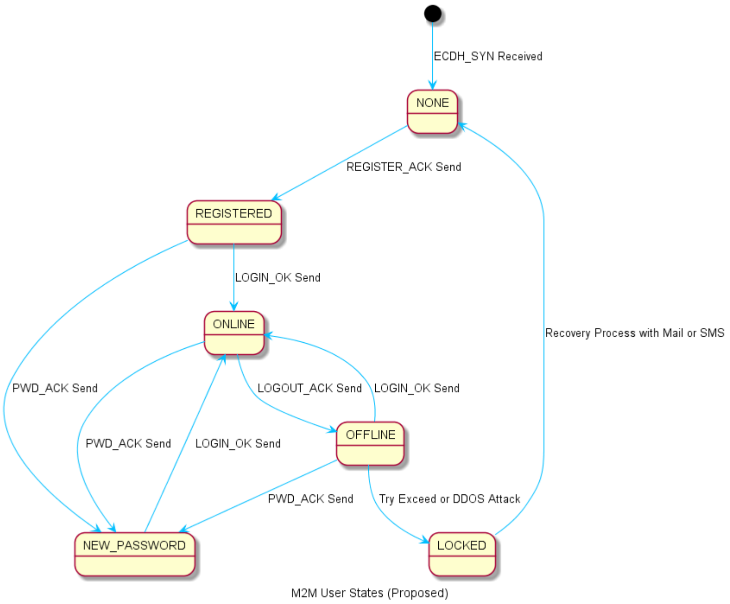


Figure 7.1: Proposed M2M Scheme User State Processing

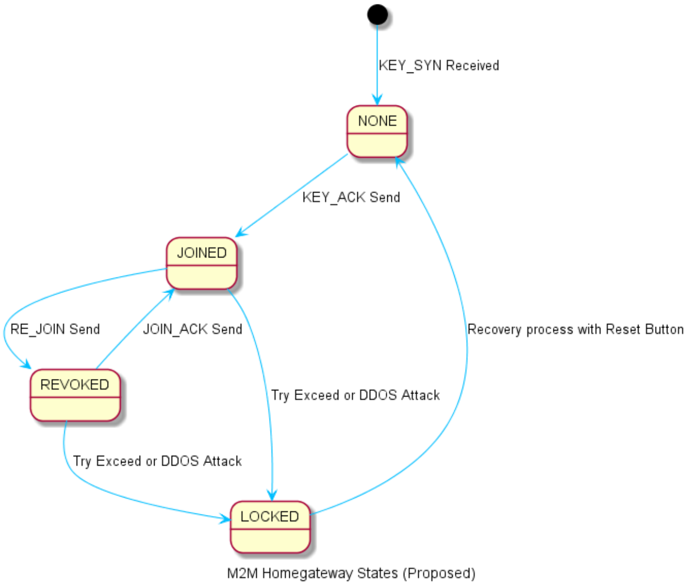

KEY_SYN Received
NONE
KEY_ACK Send
JOINED
RE_JOIN Send
JOIN_ACK Send
Recovery process with Reset Button
REVOKED
Try Exceed or DDOS Attack
Try Exceed or DDOS Attack
LOCKED
M2M Homegateway States (Proposed)

Figure 7.2: Proposed M2M Scheme Home gateway State Processing

7.3 PROPOSED M2M SCHEME PHASES

The ten stages of our suggested scheme are included. In the current network, we accomplished an effectively appropriate scheme ,and we assumed that no secure transmitting operator was available over the public network. Also, WIFI connectivity via the personal home network is possible in our scheme of home gateway and user devices. ECDH has been added to the user and the M2M server for key sharing with the registration stage, we strengthen this service. We have also added a new stage to the home gateway equipment for key-injection according to present scheme and established the client logout technique in the protocol suggested.

7.3.1 Proposed M2M Scheme Server Setup

The server configuration procedure is displayed in Figure 7.3. The ECDH pub_{m2m} and $priv_{m2m}$ key pairs of the M2M server will be generated as follows during configuration:

$$[pub_{m2m}, priv_{m2m}] = F_{i-ECDH}() \qquad (7.9)$$

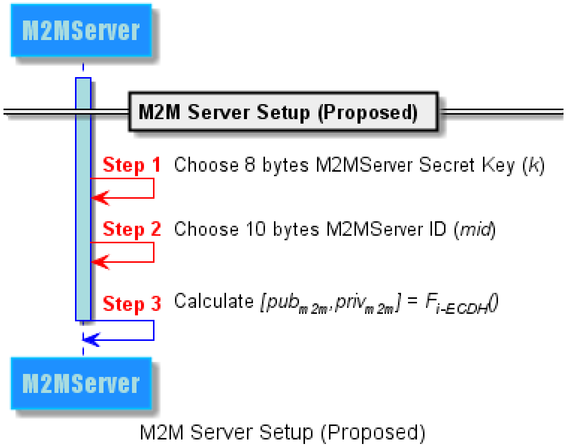

M2MServer
M2M Server Setup (Proposed)
Step 1 Choose 8 bytes M2MServer Secret Key (k)
Step 2 Choose 10 bytes M2MServer ID (mid)
Step 3 Calculate [pub_m2m, priv_m2m] = F_i-ECDH()
M2MServer
M2M Server Setup (Proposed)

Figure 7.3: Proposed M2M Scheme Server Setup Flow

The private key is 832-bits and 576-bits is the public key length. Also, a 64-bit secret key (k), introduced in the scheme, is selected from the M2M server [5]. A unique 80 bits M2M server ID (mid) must be customised to the card during the setup phase, to set the M2M server in the network. The smart cards are available for the server.

7.3.2 Proposed M2M Scheme Home Gateway Setup

Setup of the home gate in Figure 7.4. The home gateway device is relevant in its set-up operation with a 80-bit ID (hid) and is sent to a smartcard that is used on the home gateway equipment.

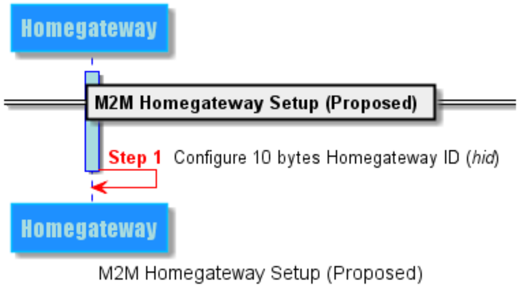

Homegateway
M2M Homegateway Setup (Proposed)
Step 1
Configure 10 bytes Homegateway ID (hid)
Homegateway
M2M Homegateway Setup (Proposed)

Figure 7.4: Proposed M2M Scheme Home gateway Setup Flow

7.3.3 Proposed M2M Scheme User Setup

Setup of the user unit in Figure 7.5. Users uniquely identified with an 80-bits ID (uid) that is stored on the smart card during the set-up operation. In addition, the user device of the key pair produces ECDH pub_{m2m} and $priv_{m2m}$ key pairs for registering, thus:

$$[pub_{user}, priv_{user}] = F_{i-ECDH}() \tag{7.10}$$

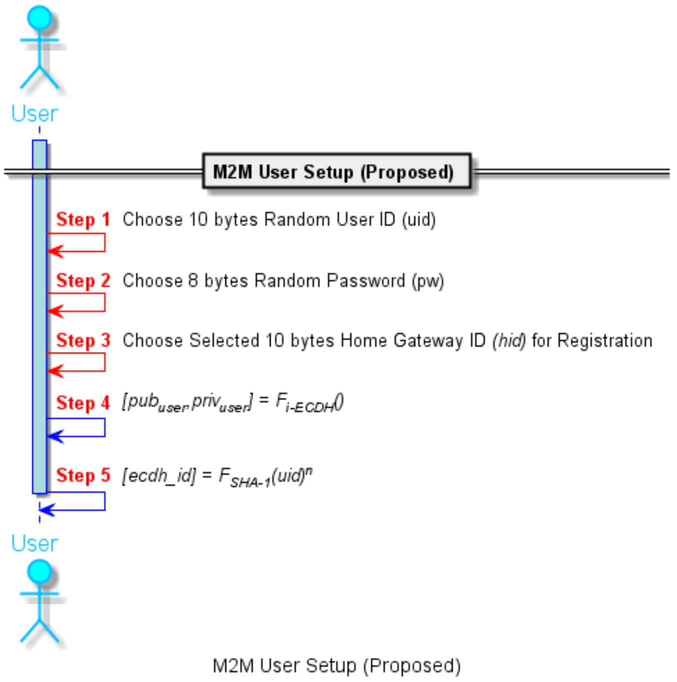

User
M2M User Setup (Proposed)
Step 1 Choose 10 bytes Random User ID (uid)
Step 2 Choose 8 bytes Random Password (pw)
Step 3 Choose Selected 10 bytes Home Gateway ID (hid) for Registration
Step 4 $[pub_{user}, priv_{user}] = F_{i\text{-}ECDH}()$
Step 5 $[ecdh\_id] = F_{SHA\text{-}1}(uid)^n$
User
M2M User Setup (Proposed)

Figure 7.5: Proposed M2M Scheme User Setup Flow

For a single key pairing produced in this stage, a temporal 80-bits $ecdh\_id$ is used with the following procedure as follows:

$$[ecdh\_id] = F_{SHA-1}(uid)^n \tag{7.11}$$

The generated length of the public key and the configuration of the private key length are equal to the parameters of the M2M server. The user must also define for registration the 64-bits password (pw) and 80-bits home gateway id (hid).

7.3.4 Proposed M2M Scheme User Registration and Key Pairing

This phase begins with the user operation processing's timer reset. The processing timer reset is a requirement for the $ECDH\_SYN$ request transmission process. The user device will send the $ECDH\_SYN$ request for registration to the M2M server in the next steps as follows:

$$< ECD\_SYN, ecdh\_id, pub_{user} > \tag{7.12}$$

The M2M server derives key_{ECDH} for use as a transport key with pub_{user} and its $priv_{m2m}$, hence:

$$key_{ECDH} = F_{s-ECDH}(pub_{user}, priv_{m2m}) \tag{7.13}$$

The M2M server produces a protected message with 16-byte key_{enc} and key_{mac}. These keys are transferred between entities in a secure way under key_{ECDH}, thus:

$$Enc(key_{enc} \parallel key_{mac}) = F_{e-ECDH}(key_{enc} \parallel key_{mac}, key_{ECDH}) \tag{7.14}$$

The M2M server resets the timer of the procedure of the receiving of the $ECDH\_ACK$ application and sends the $ECDH\_ACK$ signal to the client for the key pair as follows:

$$< ECDH\_ACK, ecdh\_id, \mathrm{pub}_{\mathrm{m2m}}, Enc(key_{enc} \parallel key_{mac}) > \tag{7.15}$$

The client instrument checks whether the application period for the $ECDH\_SYN$ is within a reasonable period of moment; if not, it clears and cancels valuable properties; othermore, key_{ECDH} is derived as follows:

$$key_{ECDH} = F_{s-ECDH}(pub_{m2m}, priv_{user}) \tag{7.16}$$

$Enc(key_{enc} \parallel key_{mac})$ is decrypted by a user device using key_{ECDH} and $key_{enc} \parallel key_{mac}$ is saved in the temporary memory. The M2M server completes user-specific important matching at the start, and the MAC numbers for sensitive activities are safely communicated as follows:

$$[key_{enc} \parallel key_{mac}] = F_{d-ECDH}(Enc(key_{enc} \parallel key_{mac}), key_{ECDH}) \tag{7.17}$$

Operations until this step is in Figure 7.6.

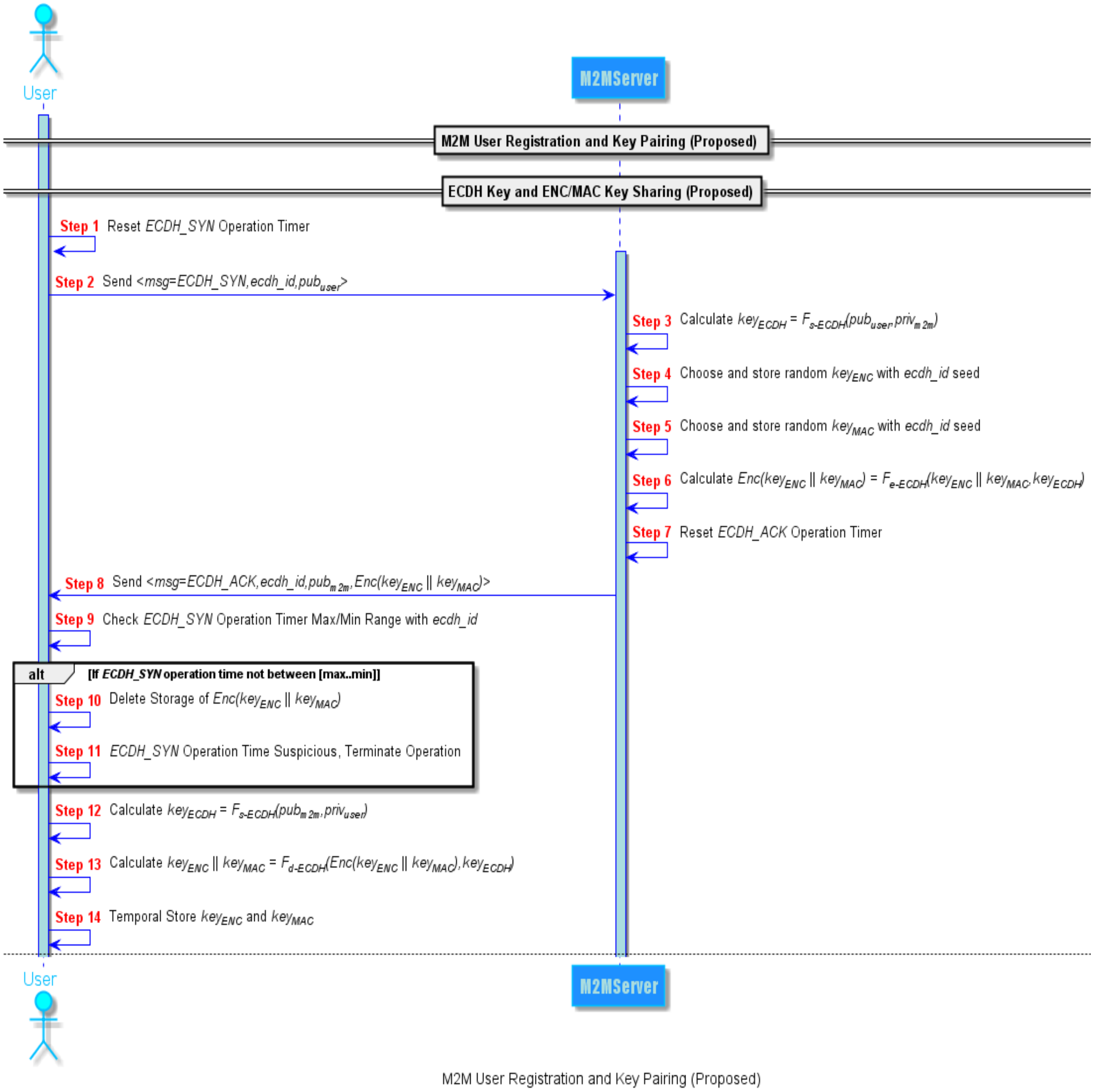

User
M2MServer
M2M User Registration and Key Pairing (Proposed)
ECDH Key and ENC/MAC Key Sharing (Proposed)
Step 1 Reset $ECDH\_SYN$ Operation Timer
Step 2 Send $<msg=ECDH\_SYN, ecdh\_id, pub_{user}>$
Step 3 Calculate $key_{ECDH} = F_{s\text{-}ECDH}(pub_{user}, priv_{m2m})$
Step 4 Choose and store random key_{ENC} with $ecdh\_id$ seed
Step 5 Choose and store random key_{MAC} with $ecdh\_id$ seed
Step 6 Calculate $Enc(key_{ENC} \| key_{MAC}) = F_{e\text{-}ECDH}(key_{ENC} \| key_{MAC}, key_{ECDH})$
Step 7 Reset $ECDH\_ACK$ Operation Timer
Step 8 Send $<msg=ECDH\_ACK, ecdh\_id, pub_{m2m}, Enc(key_{ENC} \| key_{MAC})>$
Step 9 Check $ECDH\_SYN$ Operation Timer Max/Min Range with $ecdh\_id$
alt
[If $ECDH\_SYN$ operation time not between [max..min]]
Step 10 Delete Storage of $Enc(key_{ENC} \| key_{MAC})$
Step 11 $ECDH\_SYN$ Operation Time Suspicious, Terminate Operation
Step 12 Calculate $key_{ECDH} = F_{s\text{-}ECDH}(pub_{m2m}, priv_{user})$
Step 13 Calculate $key_{ENC} \| key_{MAC} = F_{d\text{-}ECDH}(Enc(key_{ENC} \| key_{MAC}), key_{ECDH})$
Step 14 Temporal Store key_{ENC} and key_{MAC}
User
M2MServer
M2M User Registration and Key Pairing (Proposed)

Figure 7.6: Proposed M2M Scheme User Registration and Key Pairing Flow-1

The registration process starts with the client unit after the key establishment of registry information transfer. With the following procedures the client unit sets the MAC for the $REGISTER\_SYN$ submission and the submission information entered:

$$mac_{msg} = \mathrm{F}_{H-MAC}(REGISTER\_SYN \parallel ecdh_{id} \parallel \ uid \parallel pw \ \parallel hid, key_{mac}) \qquad (7.18)$$

$$enc_{msg} = \mathrm{F}_{e-AES}\left(uid \parallel pw \parallel hid \parallel mac_{msg}, key_{enc}\right) \tag{7.19}$$

The REGISTER SYN process time is reset by the user and the message is transferred to the M2M server, thus:

$$< REGISTER\_SYN, ecdh\_id, enc_{msg} > \tag{7.20}$$

The M2M server scans the queries and searches for a related decryption key using $ecdh\_id$, and verifies whether a cycle time of $ECDH\_ACK$ is in effect over time; otherwise it will destruct sensitive items and cancel the process if $ecdh\_id$ is not discovered. If not, the ciphered payload is decrypted and the MAC is validated as follows:

$$\left[uid \parallel pw \parallel hid \parallel mac_{msg}\right] = \mathrm{F}_{d-AES}\left(enc_{msg}, key_{enc}\right) \tag{7.21}$$

$$mac'_{msg} = \mathrm{F}_{H-MAC}(REGISTER\_SYN \parallel ecdh_{id} \parallel uid \parallel pw \parallel hid, key_{mac}) \tag{7.22}$$

The M2M server clears sensitive properties and cancel the procedure, if mac_{msg} and mac'_{msg} do not equal. It selects a salt s otherwise. It also calculates the following for operations 10-bytes of temporary ID (t_uid) as follows:

$$t\_uid = \mathrm{F}_{H-MAC}(uid, key_{enc}) \tag{7.23}$$

The SHA-1 value of password used as AES cipher key (key_{AES}) is calculated by the M2M server as follows:

$$d = \mathrm{F}_{SHA-1}(k \parallel t\_uid \parallel s) \tag{7.24}$$

$$key_{AES} = [d]^n \tag{7.25}$$

The M2M server encrypts the password (pw) of the user with the key AES as follows:

$$p = \mathrm{F}_{e-AES}(pw, key_{AES}) \tag{7.26}$$

The M2M server computes the 10-bytes transient home gateway ID (t_hid) for storing as described:

$$t\_hid = \mathrm{F}_{e-AES}(\ hid, key_{enc}) \tag{7.27}$$

The M2M server configures $REGISTERED$ for the client status (u_st) and $NONE$ for the internet gateway status (h_st), thus:

$$u\_st = \text{REGISTERED} \tag{7.28}$$

$$h\_st = \text{NONE} \tag{7.29}$$

The last parameter on the M2M server are:

$$< t\_uid, t\_hid, s, p, sf, key_{enc}, key_{mac} > \tag{7.30}$$

The REGISTER ACK request MAC and encoded payload are generating from the M2M Server after that phase:

$$mac_{msg} = \mathrm{F}_{H-MAC}(REGISTER\_ACK \parallel ecdh\_id \parallel\ s, key_{mac}) \tag{7.31}$$

$$enc_{msg} = \mathrm{F}_{e-AES}\big(\ s\ \parallel\ mac_{msg}, key_{enc}\big) \tag{7.32}$$

Operations until this point is in Figure 7.7.

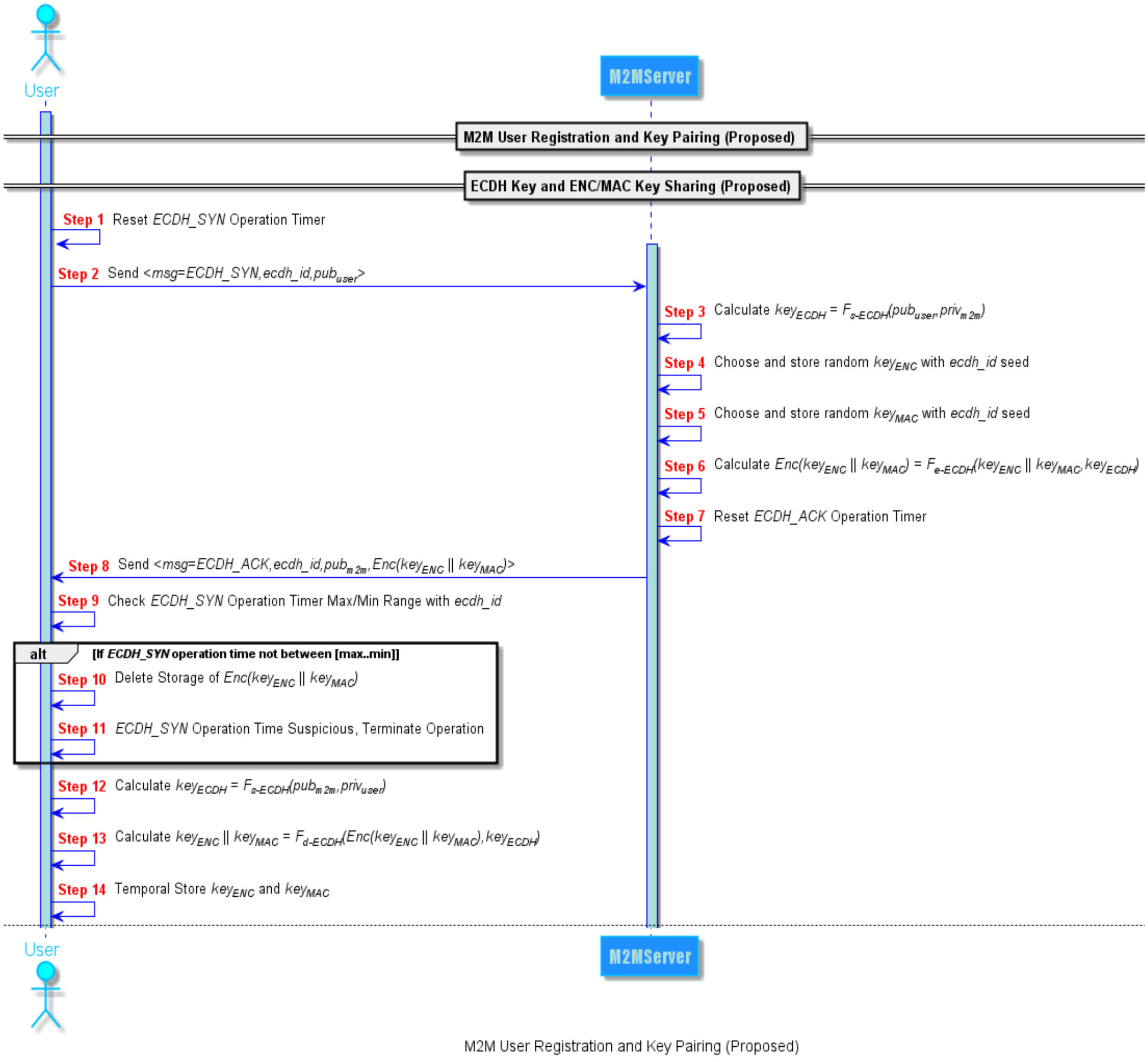

User
M2MServer
M2M User Registration and Key Pairing (Proposed)
ECDH Key and ENC/MAC Key Sharing (Proposed)
Step 1 Reset ECDH_SYN Operation Timer
Step 2 Send <msg=ECDH_SYN,ecdh_id,pub_{user}>
Step 3 Calculate $key_{ECDH} = F_{s\text{-}ECDH}(pub_{user}, priv_{m2m})$
Step 4 Choose and store random key_{ENC} with ecdh_id seed
Step 5 Choose and store random key_{MAC} with ecdh_id seed
Step 6 Calculate $Enc(key_{ENC} \| key_{MAC}) = F_{e\text{-}ECDH}(key_{ENC} \| key_{MAC}, key_{ECDH})$
Step 7 Reset ECDH_ACK Operation Timer
Step 8 Send <msg=ECDH_ACK,ecdh_id,pub_{m2m},$Enc(key_{ENC} \| key_{MAC})$>
Step 9 Check ECDH_SYN Operation Timer Max/Min Range with ecdh_id
alt
[If ECDH_SYN operation time not between [max..min]]
Step 10 Delete Storage of $Enc(key_{ENC} \| key_{MAC})$
Step 11 ECDH_SYN Operation Time Suspicious, Terminate Operation
Step 12 Calculate $key_{ECDH} = F_{s\text{-}ECDH}(pub_{m2m}, priv_{user})$
Step 13 Calculate $key_{ENC} \| key_{MAC} = F_{d\text{-}ECDH}(Enc(key_{ENC} \| key_{MAC}), key_{ECDH})$
Step 14 Temporal Store key_{ENC} and key_{MAC}
User
M2MServer
M2M User Registration and Key Pairing (Proposed)

Figure 7.7: Proposed M2M Scheme User Registration and Key Pairing Flow-2

$REGISTER\_ACK$ timer is reset and the send to the user unit as follows:

$$< REGISTER\_ACK, ecdh\_id, enc_{msg} > \tag{7.33}$$

Users control whether the $REGISTER\_SYN$ handling period is within a reasonable scope, otherwise the critical properties will be removed and the procedure will be cancelled. Otherwise the ciphered payload is decrypted and the MAC is validated as below:

$$[s \parallel mac_{msg}] = \mathrm{F}_{d-AES}(enc_{msg}, key_{enc}) \tag{7.34}$$

$$mac'_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = REGISTER\_ACK \parallel ecdh\_id \parallel s, key_{mac}) \tag{7.35}$$

When mac'_{msg} and mac_{msg} are not equal, the client unit removes and cancels the method of critical assets. If not, key_{mac} and key_{enc} memory will be saved and committed in other activities. Figure 7.8 will operate until this phase.

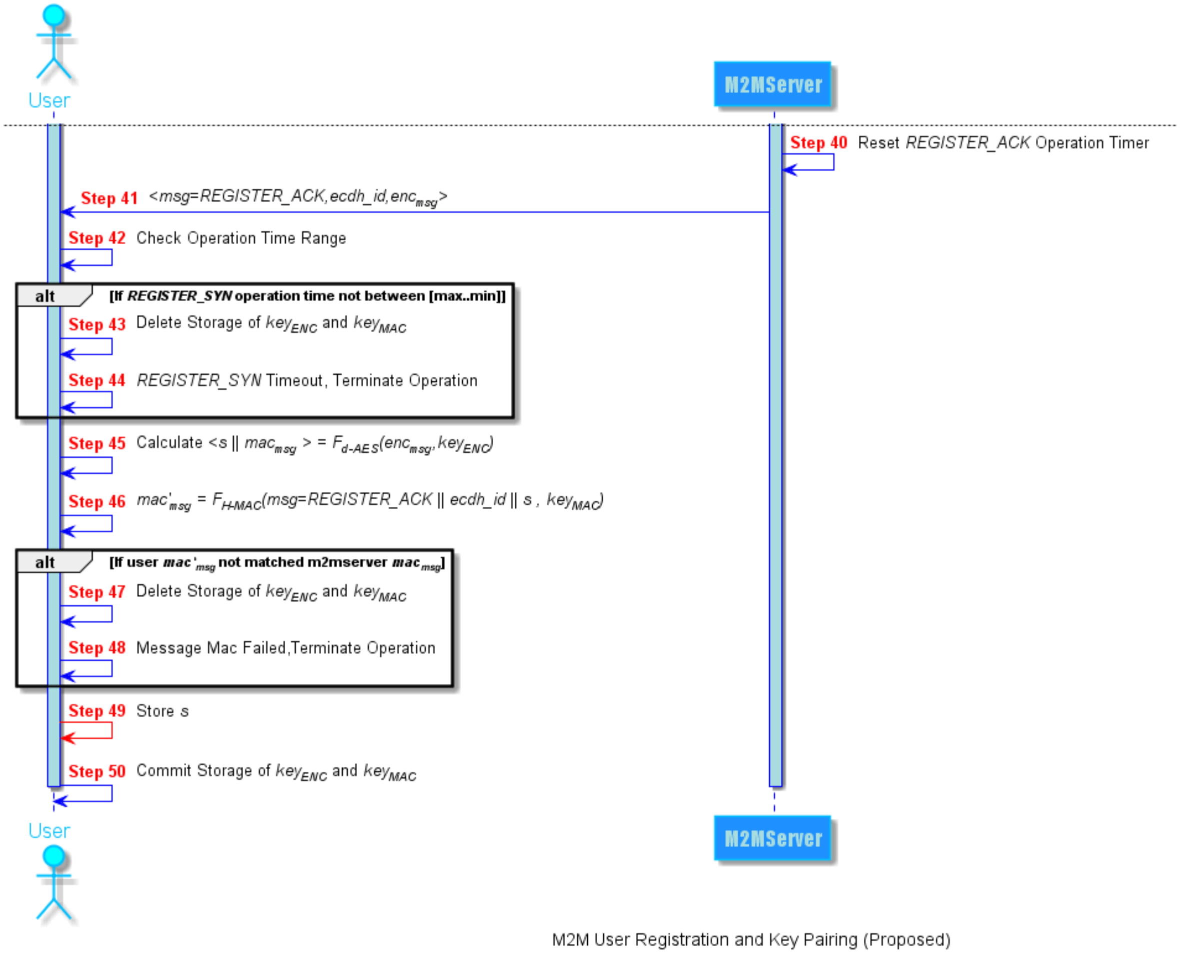

User
M2MServer
Step 40 Reset REGISTER_ACK Operation Timer
Step 41 <msg=REGISTER_ACK,ecdh_id,enc$_{msg}$>
Step 42 Check Operation Time Range
alt
[If REGISTER_SYN operation time not between [max..min]]
Step 43 Delete Storage of key$_{ENC}$ and key$_{MAC}$
Step 44 REGISTER_SYN Timeout, Terminate Operation
Step 45 Calculate <s || mac$_{msg}$ > = F$_{d-AES}$(enc$_{msg}$, key$_{ENC}$)
Step 46 mac'$_{msg}$ = F$_{H-MAC}$(msg=REGISTER_ACK || ecdh_id || s , key$_{MAC}$)
alt
[If user mac'$_{msg}$ not matched m2mserver mac$_{msg}$]
Step 47 Delete Storage of key$_{ENC}$ and key$_{MAC}$
Step 48 Message Mac Failed,Terminate Operation
Step 49 Store s
Step 50 Commit Storage of key$_{ENC}$ and key$_{MAC}$
User
M2MServer
M2M User Registration and Key Pairing (Proposed)

Figure 7.8: Proposed M2M Scheme User Registration and Key Pairing Flow-3

7.3.5 Proposed M2M Scheme for the User Login and Authentication Phase

The customer device chooses an arbitrary x_1 numbers and parameters for encryption g and h_1:

$$g = \mathrm{F}_{e-AES}(x_1\ , pw) \tag{7.36}$$

$$h_1 = \mathrm{F}_{SHA-1}(\mathrm{x}_1) \tag{7.37}$$

The user derives t_uid as the following in accordance with the privacy of the user ID:

$$t\_uid = F_{H-MAC}(uid, key_{enc}) \tag{7.38}$$

A $LOGIN\_SYN$ application MAC is prepared by the user unit as follows:

$$mac_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = LOGIN\_SYN \parallel t\_uid \parallel\ g \parallel h_1 \parallel\ s, key_{mac}) \tag{7.39}$$

The user device resets $LOGIN\_SYN$'s processing timer and sends the message to the M2M server:

$$< LOGIN\_SYN, t\_uid, g, h_1, s, mac_{msg} > \tag{7.40}$$

t_uid and s are validated in the database by the M2M Server. The M2M server cancels the method if the record is not discovered. If the client is discovered and the state is $LOCKED$ or not equal to $REGISTERED$, $OFFLINE$ and $NEW\_PASSWORD$, the procedure will end. When all the terms are true, the $LOGIN\_SYN$ message MAC will be checked as follows:

$$mac'_{msg} = \mathrm{F}_{H-MAC}(LOGIN\_SYN \parallel t\_uid \parallel\ g \parallel h_1 \parallel\ s, key_{mac}) \tag{7.41}$$

Operations until this step is in Figure 7.9.

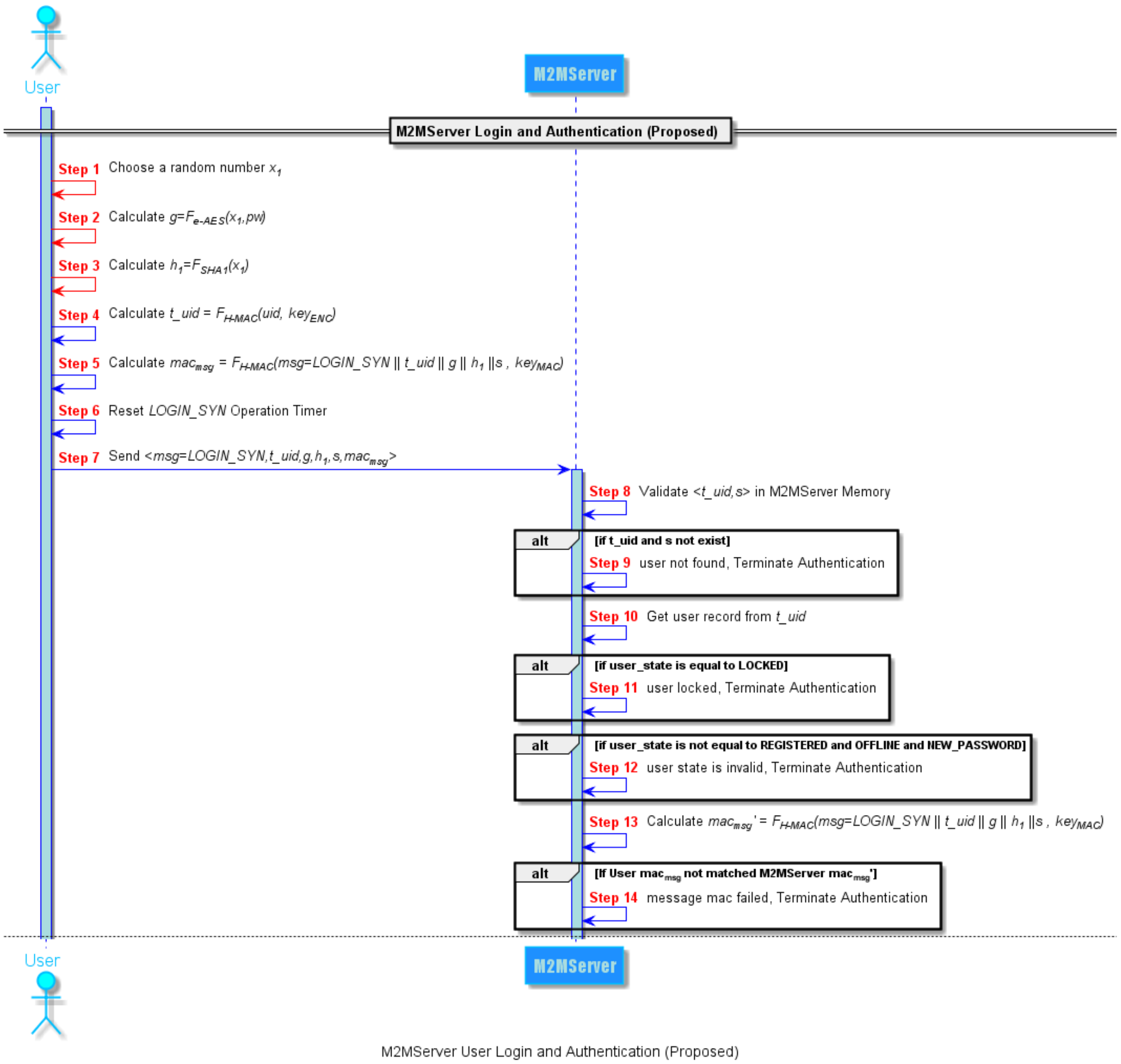

User
M2MServer
M2MServer Login and Authentication (Proposed)
Step 1 Choose a random number x_1
Step 2 Calculate $g=F_{e\text{-}AES}(x_1,pw)$
Step 3 Calculate $h_1=F_{SHA1}(x_1)$
Step 4 Calculate $t\_uid = F_{H\text{-}MAC}(uid, key_{ENC})$
Step 5 Calculate $mac_{msg} = F_{H\text{-}MAC}(msg=LOGIN\_SYN \| t\_uid \| g \| h_1 \| s , key_{MAC})$
Step 6 Reset LOGIN_SYN Operation Timer
Step 7 Send $<msg=LOGIN\_SYN,t\_uid,g,h_1,s,mac_{msg}>$
Step 8 Validate <t_uid,s> in M2MServer Memory
alt
[if t_uid and s not exist]
Step 9 user not found, Terminate Authentication
Step 10 Get user record from t_uid
alt
[if user_state is equal to LOCKED]
Step 11 user locked, Terminate Authentication
alt
[if user_state is not equal to REGISTERED and OFFLINE and NEW_PASSWORD]
Step 12 user state is invalid, Terminate Authentication
Step 13 Calculate $mac_{msg}' = F_{H\text{-}MAC}(msg=LOGIN\_SYN \| t\_uid \| g \| h_1 \| s , key_{MAC})$
alt
[If User mac_{msg} not matched M2MServer mac_{msg}']
Step 14 message mac failed, Terminate Authentication
User
M2MServer
M2MServer User Login and Authentication (Proposed)

Figure 7.9: Proposed M2M Scheme User Login and Authentication Flow-1

The AES key calculation remains to check h_1 if mac_{msg} and the calculated mac'_{msg} is equivalent. The M2M server calculates the SHA-1 value of the password code as key_{AES}, thus:

$$d = \mathrm{F}_{SHA-1}(k \;\| \; t\_uid \;\| \; s) \qquad (7.42)$$

$$key_{AES} = [d]^n \qquad (7.43)$$

$$pw' = \mathrm{F}_{d-AES}(\,p, key_{AES}) \tag{7.44}$$

$$x'_1 = \mathrm{F}_{d-AES}(\,g, pw') \tag{7.45}$$

$$h'_1 = \mathrm{F}_{SHA-1}(\mathrm{x}'_1) \tag{7.46}$$

If the value of h_1 received and the value of h'_1 calculated do not match, the password stored is wrong and the M2M server cancels the operation; otherwise, a new random number x_2 is chosen by the M2M server and umk and h_2 are calculated as follows:

$$umk = [\,\mathrm{F}_{SHA-1}(\mathrm{x}'_1 \ \| \ x_2)]^n \tag{7.47}$$

$$h_2 = F_{SHA1}(umk \ \| \ mid) \tag{7.48}$$

The $LOGIN\_ACK$ message MAC calculates the following M2M server:

$$mac_{msg} = \mathrm{F}_{H-MAC}(LOGIN\_ACK \ \| \ mid \ \| \ \ x_2 \ \| \ h_2 \ \| \ \ s, key_{mac}) \tag{7.49}$$

Operations until this step is in Figure 7.10.

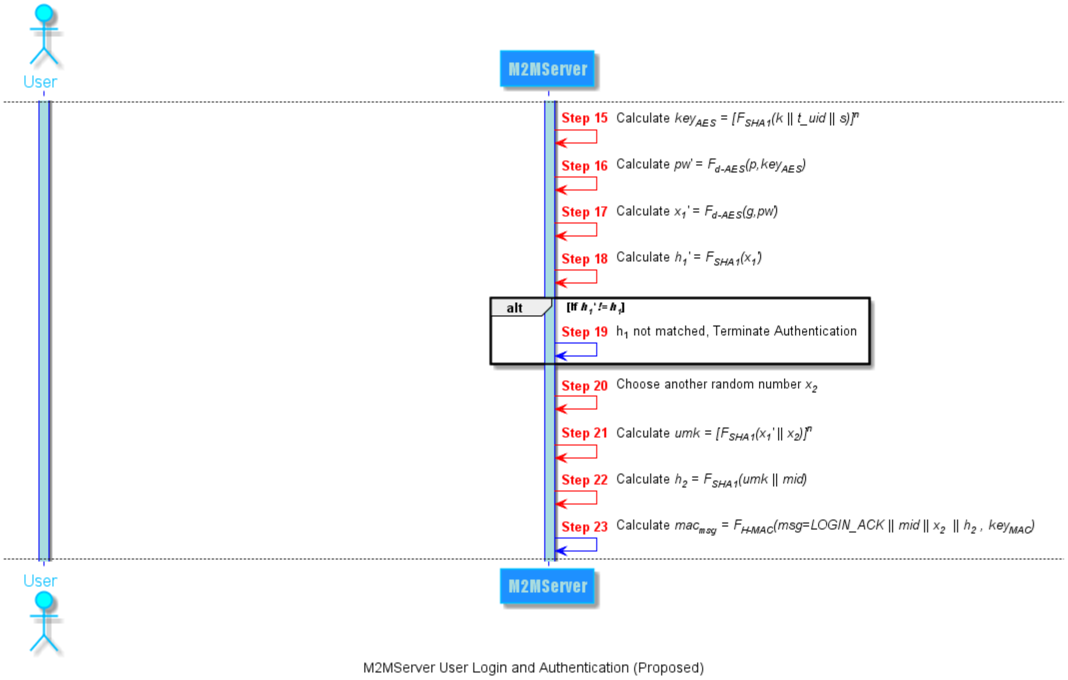

User
M2MServer
Step 15 Calculate $key_{AES} = [F_{SHA1}(k \| t\_uid \| s)]^n$
Step 16 Calculate $pw' = F_{d\text{-}AES}(p, key_{AES})$
Step 17 Calculate $x_1' = F_{d\text{-}AES}(g, pw)$
Step 18 Calculate $h_1' = F_{SHA1}(x_1)$
alt
[If h_1' != h_1]
Step 19 h_1 not matched, Terminate Authentication
Step 20 Choose another random number x_2
Step 21 Calculate $umk = [F_{SHA1}(x_1' \| x_2)]^n$
Step 22 Calculate $h_2 = F_{SHA1}(umk \| mid)$
Step 23 Calculate $mac_{msg} = F_{H\text{-}MAC}(msg=LOGIN\_ACK \| mid \| x_2 \| h_2 , key_{MAC})$
User
M2MServer
M2MServer User Login and Authentication (Proposed)

Figure 7.10: Proposed M2M Scheme User Login and Authentication Flow-2

After that, the M2M server resets the $LOGIN\_ACK$ operating timer and transmits the request to the user devices:

$$< LOGIN\_ACK, mid, x_2, h_2, mac_{msg} > \tag{7.50}$$

The client equipment verifies the validity of the processing time of $LOGIN\_SYN$ procedure. If time is not valid then the procedure is terminated els the $LOGIN\_ACK$ application MAC will be validated as follow:

$$mac'_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = LOGIN\_ACK \parallel mid \parallel x_2 \parallel h_2 \parallel s, key_{mac}) \tag{7.51}$$

If computed mac'_{msg} is not validated with received mac_{msg}, the action will be canceled; otherwise it will verify that h_2 is received on the M2M server, as follows:

$$umk' = \mathrm{F}_{SHA1}(x_1 \parallel x_2)^n \tag{7.52}$$

$$h'_2 = \mathrm{F}_{SHA1}(umk' \parallel mid) \tag{7.53}$$

If the h_2 value obtained is combined with that produced by the h'_2, then verification is done. Following this phase, the client devices will store umk' for safe transfer of information. Otherwise the method will be terminated by the user unit. The user checks the M2M server until this step, calculating the h_3 authentication for M2M server as follows:

$$h_3 = \mathrm{F}_{SHA1}(umk' \parallel t\_uid) \tag{7.54}$$

The user equipment the creates $LOGIN\_OK$ request MAC, and sends the $LOGIN\_OK$ requests to the M2M server:

$$mac_{msg} = \mathrm{F}_{H-MAC}(LOGIN\_OK \parallel h_3, key_{mac}) \tag{7.55}$$

$$< LOGIN\_OK, h_3, mac_{msg} > \tag{7.56}$$

Operations until this point is in Figure 7.11.

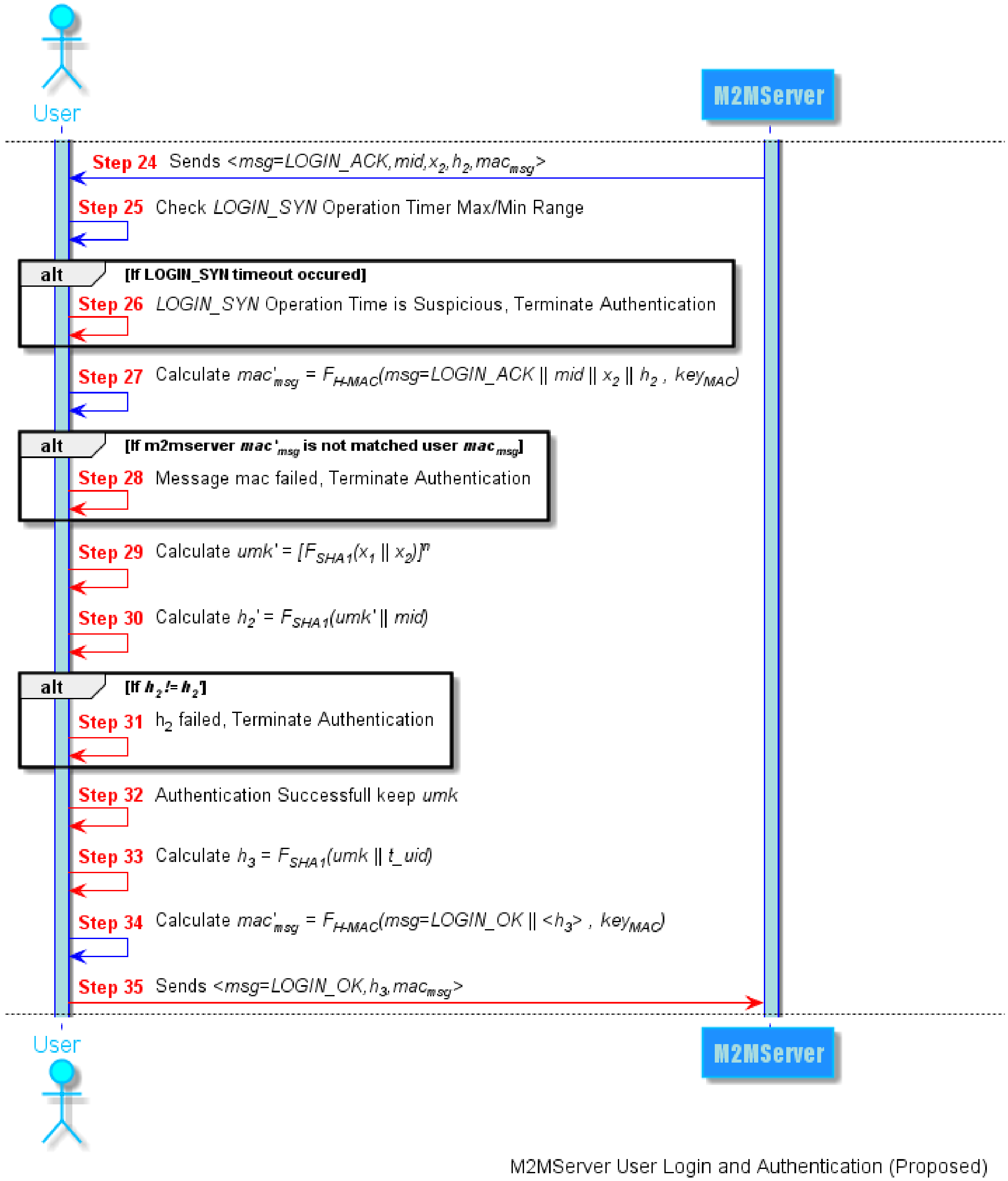

User
M2MServer
Step 24 Sends <msg=LOGIN_ACK, mid, x_2, h_2, mac_{msg}>
Step 25 Check LOGIN_SYN Operation Timer Max/Min Range
alt
[If LOGIN_SYN timeout occured]
Step 26 LOGIN_SYN Operation Time is Suspicious, Terminate Authentication
Step 27 Calculate mac'_{msg} = $F_{H\text{-}MAC}$(msg=LOGIN_ACK || mid || x_2 || h_2 , key_{MAC})
alt
[If m2mserver mac'_{msg} is not matched user mac_{msg}]
Step 28 Message mac failed, Terminate Authentication
Step 29 Calculate umk' = $[F_{SHA1}(x_1 \| x_2)]^n$
Step 30 Calculate h_2' = F_{SHA1}(umk' || mid)
alt
[If h_2 != h_2']
Step 31 h_2 failed, Terminate Authentication
Step 32 Authentication Successfull keep umk
Step 33 Calculate h_3 = F_{SHA1}(umk || t_uid)
Step 34 Calculate mac'_{msg} = $F_{H\text{-}MAC}$(msg=LOGIN_OK || <h_3> , key_{MAC})
Step 35 Sends <msg=LOGIN_OK, h_3, mac_{msg}>
User
M2MServer
M2MServer User Login and Authentication (Proposed)

Figure 7.11: Proposed M2M Scheme User Login and Authentication Flow-3

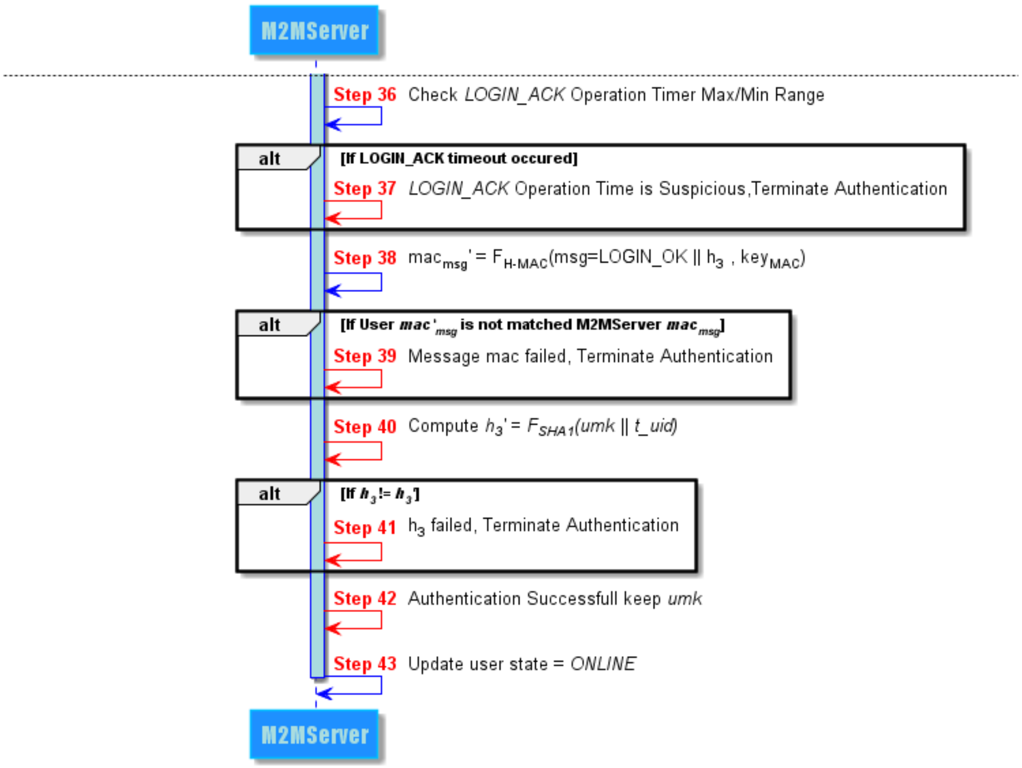
The M2M server checks whether the operational processing time for *LOGIN_ACK* is within a valid time frame; otherwise, the operation will be completed. The *LOGIN_OK* MAC message otherwise checks:

$$mac'_{msg} = \mathrm{F}_{H-MAC}(\mathrm{msg} = LOGIN\_OK \parallel h_3, key_{mac}) \qquad (7.57)$$

The M2M server verifies h_3 as follows if the calculated mac'_{msg} is equal to that of the obtained mac_{msg}:

$$h'_3 = \mathrm{F}_{SHA1}(umk \parallel t\_uid) \qquad (7.58)$$

The validation is finished if the obtained h_3 and h'_3 matches are calculated. The M2M server sets $ONLINE$ to the client device and saves for safe transfer of information. Figure 7.12 shows activities up to this phase.


M2MServer
alt
[If LOGIN_ACK timeout occured]
alt
[If User mac'_{msg} is not matched M2MServer mac_{msg}]
alt
[If h_3 != h_3]
M2MServer

Step 36 Check *LOGIN_ACK* Operation Timer Max/Min Range

Step 37 *LOGIN_ACK* Operation Time is Suspicious,Terminate Authentication

Step 38 $mac_{msg}' = F_{H\text{-}MAC}(msg{=}LOGIN\_OK \parallel h_3 , key_{MAC})$

Step 39 Message mac failed, Terminate Authentication

Step 40 Compute $h_3' = F_{SHA1}(umk \parallel t\_uid)$

Step 41 h_3 failed, Terminate Authentication

Step 42 Authentication Successfull keep *umk*

Step 43 Update user state = *ONLINE*

M2MServer User Login and Authentication (Proposed)

Figure 7.12: Proposed M2M Scheme User Login and Authentication Flow-4

7.3.6 Proposed M2M Scheme User Home Gateway Key Injection

The key_{enc} and the key_{mac} parameters of a valid registered client device. Sensitive assets are protected during the transfer between the home gateway devices, the M2M server and the client devices. The home gateway unit can communicate to the house network in the present design [5] which implies that they can be connected through WIFI or Ethernet with IP (Internet Protocol). We suppose that this unit is linked with the home IP network and is linked to the same network via a client device looking through the home network.

The home gateway equipment without a home network, can set up a personal-private network through the access point method and allow the user unit to link to the home gateway device via prefabricated, specified data. Also, Bluetooth-capable house gateways can authenticate this link. A safe information transfer channel is provided through this personal-private-network. Figure 7.13 is the whole flow.

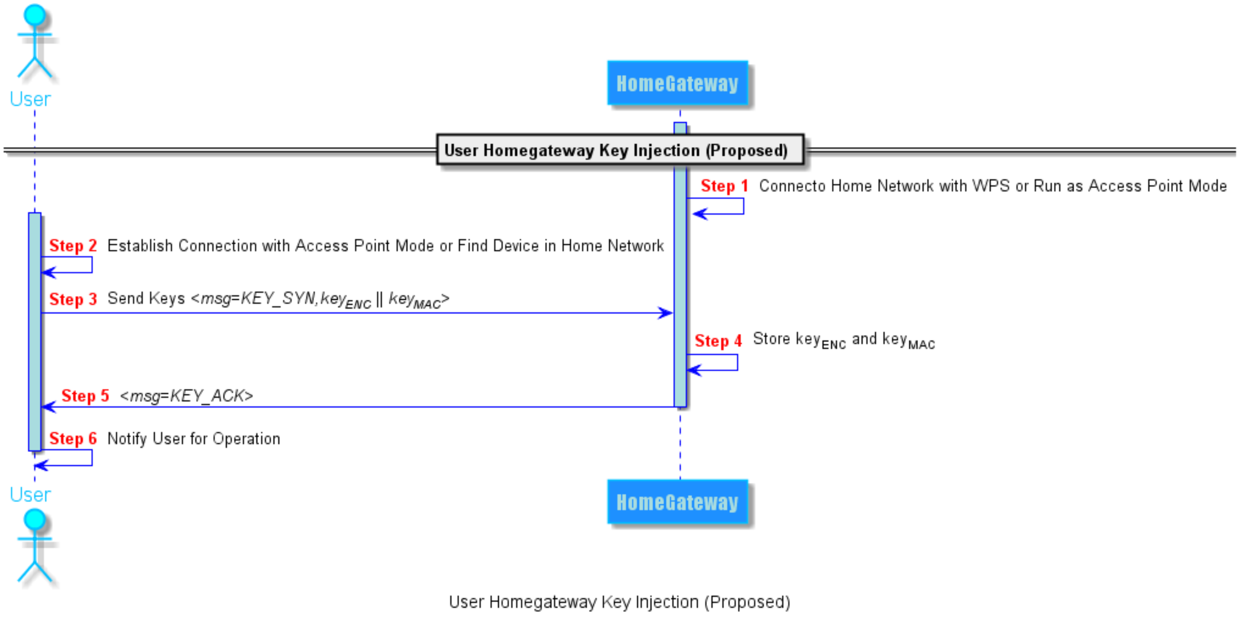

User
HomeGateway
User Homegateway Key Injection (Proposed)
Step 1 Connecto Home Network with WPS or Run as Access Point Mode
Step 2 Establish Connection with Access Point Mode or Find Device in Home Network
Step 3 Send Keys <msg=KEY_SYN,key_ENC || key_MAC>
Step 4 Store key_ENC and key_MAC
Step 5 <msg=KEY_ACK>
Step 6 Notify User for Operation
User
HomeGateway
User Homegateway Key Injection (Proposed)

Figure 7.13: Proposed M2M Scheme User Home Gateway Key Injection Flow

With a KEY_SYN application, the customer transfers data to the home gateway device via this safeguarded channel:

$$< KEY\_SYN, key_{enc} \parallel key_{mac} > \tag{7.59}$$

The system for home gateways maintains the key_{enc} and key_{mac} parameters for the phase of connection and returns the KEY_ACK user unit reply as follows:

$$< KEY\_ACK > \tag{7.60}$$

The key injection is done after this application.

7.3.7 Proposed M2M Scheme Home Gateway Join Network

The home gateway equipment can safely integrate the network after the key injection procedure. The transient home gateway ID (t_hid) is calculated by actual home gateway ID (hid) in the first step.

$$t\_hid = F_{H-MAC}(hid, key_{enc}) \tag{7.61}$$

The home gateway prepares MAC requests for $JOIN\_SYN$ and resets the timer of the operation for this message and sends it to the M2M server as follows:

$$mac_{msg} = F_{H-MAC}(JOIN\_SYN \parallel t\_hid, key_{mac}) \tag{7.62}$$

$$< JOIN\_SYN, t\_hid, mac_{msg} > \tag{7.63}$$

M2M Server searches for t_hid and tries to obtain the accurate record in the server database. Works are performed in Figure 7.14 until this phase.

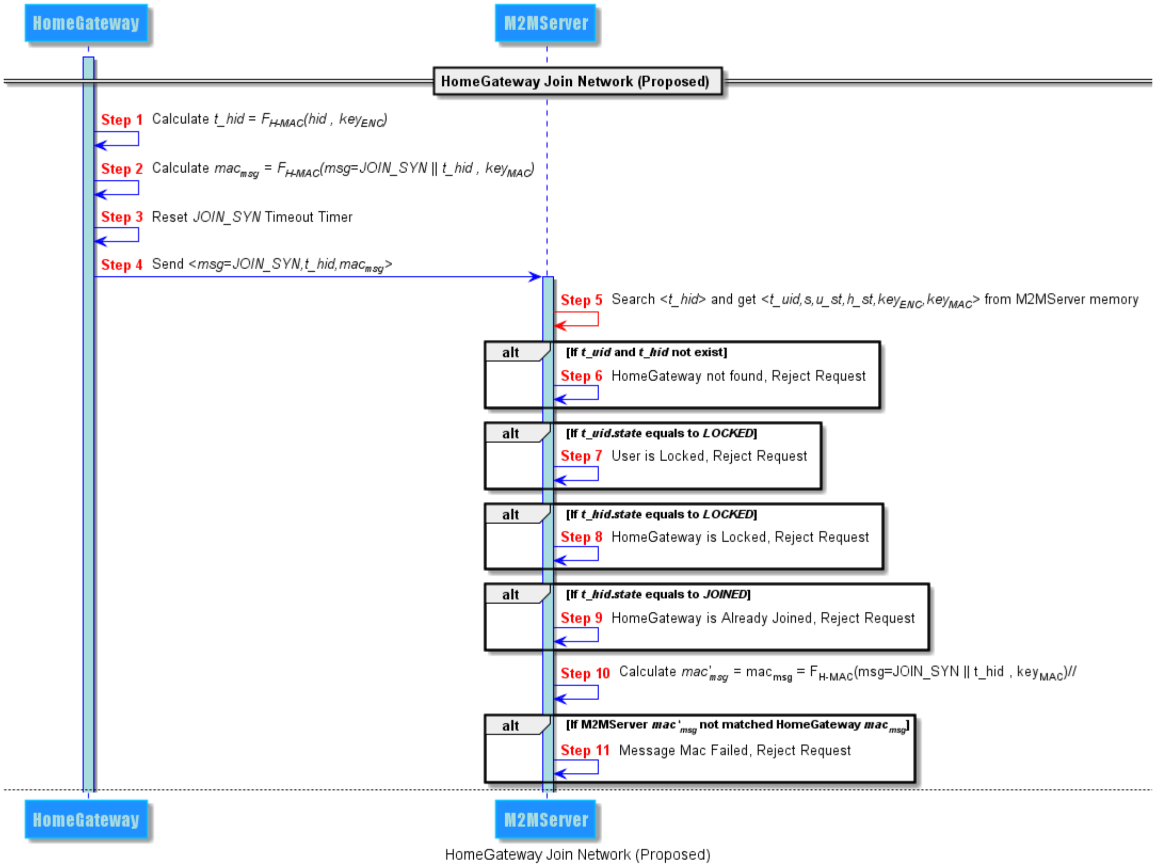

HomeGateway
M2MServer
HomeGateway Join Network (Proposed)
Step 1 Calculate t_hid = F_H-MAC(hid , key_ENC)
Step 2 Calculate mac_msg = F_H-MAC(msg=JOIN_SYN || t_hid , key_MAC)
Step 3 Reset JOIN_SYN Timeout Timer
Step 4 Send <msg=JOIN_SYN,t_hid,mac_msg>
Step 5 Search <t_hid> and get <t_uid,s,u_st,h_st,key_ENC,key_MAC> from M2MServer memory
alt
[If t_uid and t_hid not exist]
Step 6 HomeGateway not found, Reject Request
alt
[If t_uid.state equals to LOCKED]
Step 7 User is Locked, Reject Request
alt
[If t_hid.state equals to LOCKED]
Step 8 HomeGateway is Locked, Reject Request
alt
[If t_hid.state equals to JOINED]
Step 9 HomeGateway is Already Joined, Reject Request
Step 10 Calculate mac'_msg = mac_msg = F_H-MAC(msg=JOIN_SYN || t_hid , key_MAC)//
alt
[If M2MServer mac'_msg not matched HomeGateway mac_msg]
Step 11 Message Mac Failed, Reject Request
HomeGateway
M2MServer
HomeGateway Join Network (Proposed)

Figure 7.14: Proposed M2M Scheme Home Gateway Join Network Flow-1

If the user related device is not found by the M2M server, then abort the procedure. Otherwise, for the related operation, the M2M server gets t_uid u_st, h_st, s, key_{enc}, and key_{mac}. If the condition of the home portal (h_st) or client state (u_st) is $LOCKED$, or the condition of the home portal is already $JOINED$, the server will abort the procedure. Otherwise, $JOIN\_SYN$ request MAC is validated by the M2M server as follows:

$$mac'_{msg} = F_{H-MAC}(JOIN\_SYN \parallel t\_hid, key_{mac}) \tag{7.64}$$

If the mac_{msg} obtained and the mac'_{msg} calculated do not match, the procedure will be aborted by the home gateway. Otherwise, the server will cipher the session key. For the home gateway device, the M2M server calculates the session key, H_{key} and the session key parameter H_p as follows:

$$H_p = [F_{SHA1}(t\_uid \parallel mid \parallel s)]^m \tag{7.65}$$

$$H_{key} = [F_{SHA1}\left(t\_hid \parallel H_p\right)]^t \tag{7.66}$$

The M2M server stores H_{key} and combines the $JOIN\_ACK$ request with MAC and the ciphered data, as follows:

$$mac_{msg} = F_{H-MAC}(JOIN\_ACK \parallel H_p, key_{mac}) \tag{7.67}$$

$$enc_{msg} = F_{e-AES}(H_p \ \parallel mac_{msg}, key_{enc}) \tag{7.68}$$

$$< JOIN\_ACK, enc_{msg} > \tag{7.69}$$

The unit of the home gateway checks whether the processing period of the $JOIN\_SYN$ application is true; if not, the device will end the operation. Otherwise, the unit will decrypt and verify the $JOIN\_SYN$ signal as follows:

$$[H_p \ \parallel mac_{msg}] = F_{d-AES}(enc_{msg}, key_{enc}) \tag{7.70}$$

$$mac'_{msg} = F_{H-MAC}(JOIN\_ACK \parallel H_p, key_{mac}) \tag{7.71}$$

If the mac_{msg} obtained and calculated does not match mac'_{msg}, the unit of the home portal will end the procedure. Otherwise, for safe communication, the unit calculates H'_{key} and holds it in memory as follows:

$$H'_{key} = [F_{SHA1}\left(t\_hid \parallel H_p\right)]^t \tag{7.72}$$

The home gateway device combines the M2M server's $JOIN\_OK$ request with MAC, as follows:

$$mac_{msg} = F_{H-MAC}(JOIN\_OK \parallel t\_hid, key_{mac}) \tag{7.73}$$

$$< JOIN\_OK, t\_hid, mac_{msg} > \tag{7.74}$$

The M2M server checks whether the handling period of the $JOIN\_OK$ application is appropriate; if not, the procedure will be terminated. Otherwise, $JOIN\_OK$ MAC will be verified and the home gateway (h_st) will be set to $JOINED$ as follows:

$$mac'_{msg} = F_{H-MAC}(JOIN\_OK \parallel t\_hid, key_{mac}) \tag{7.75}$$

If MAC does not match or a timeout occurs and $JOIN\_OK$ is not received, the operation will be aborted by the M2M server. Operations are in Figure 7.15 until this phase.

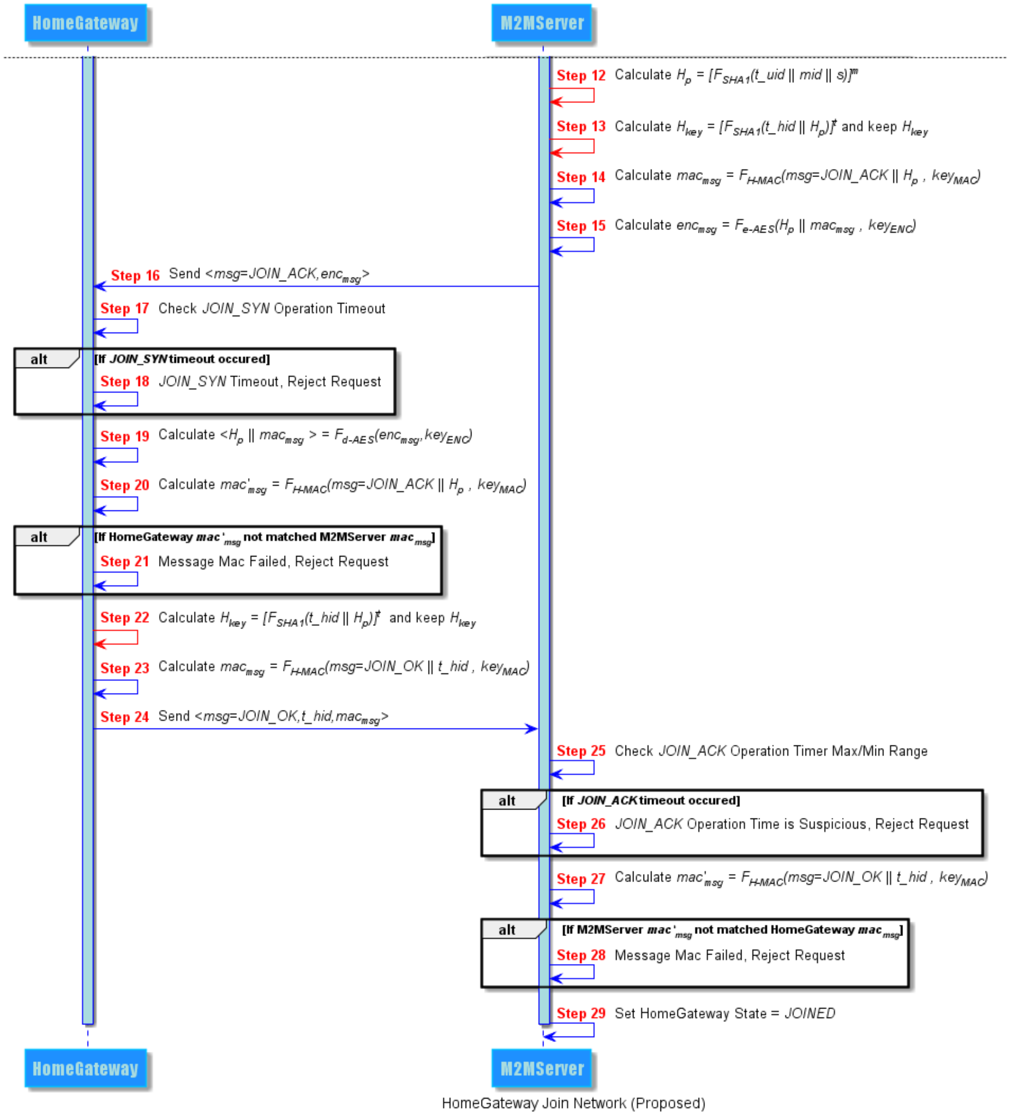

HomeGateway
M2MServer
Step 12 Calculate $H_p = [F_{SHA1}(t\_uid \parallel mid \parallel s)]^m$
Step 13 Calculate $H_{key} = [F_{SHA1}(t\_hid \parallel H_p)]^t$ and keep H_{key}
Step 14 Calculate $mac_{msg} = F_{H-MAC}(msg{=}JOIN\_ACK \parallel H_p , key_{MAC})$
Step 15 Calculate $enc_{msg} = F_{e-AES}(H_p \parallel mac_{msg} , key_{ENC})$
Step 16 Send $<msg{=}JOIN\_ACK, enc_{msg}>$
Step 17 Check $JOIN\_SYN$ Operation Timeout
alt
[If $JOIN\_SYN$ timeout occured]
Step 18 $JOIN\_SYN$ Timeout, Reject Request
Step 19 Calculate $<H_p \parallel mac_{msg}> = F_{d-AES}(enc_{msg}, key_{ENC})$
Step 20 Calculate $mac'_{msg} = F_{H-MAC}(msg{=}JOIN\_ACK \parallel H_p , key_{MAC})$
alt
[If HomeGateway mac'_{msg} not matched M2MServer mac_{msg}]
Step 21 Message Mac Failed, Reject Request
Step 22 Calculate $H_{key} = [F_{SHA1}(t\_hid \parallel H_p)]^t$ and keep H_{key}
Step 23 Calculate $mac_{msg} = F_{H-MAC}(msg{=}JOIN\_OK \parallel t\_hid , key_{MAC})$
Step 24 Send $<msg{=}JOIN\_OK, t\_hid, mac_{msg}>$
Step 25 Check $JOIN\_ACK$ Operation Timer Max/Min Range
alt
[If $JOIN\_ACK$ timeout occured]
Step 26 $JOIN\_ACK$ Operation Time is Suspicious, Reject Request
Step 27 Calculate $mac'_{msg} = F_{H-MAC}(msg{=}JOIN\_OK \parallel t\_hid , key_{MAC})$
alt
[If M2MServer mac'_{msg} not matched HomeGateway mac_{msg}]
Step 28 Message Mac Failed, Reject Request
Step 29 Set HomeGateway State = $JOINED$
HomeGateway
M2MServer
HomeGateway Join Network (Proposed)

Figure 7.15: Proposed M2M Scheme Home Gateway Join Network Flow-2

7.3.8 Proposed M2M Scheme for the User Password Change

The client equipment for the method determines t_uid. It also creates a application for PWD_SYN with MAC and the following ciphered information:

$$t\_uid = F_{H-MAC}(uid, key_{enc}) \tag{7.76}$$

$$mac_{msg} = F_{H-MAC}(PWD\_SYN \parallel t\_uid \parallel pw \ \parallel pw\_new \parallel s, key_{mac}) \tag{7.77}$$

$$enc_{msg} = F_{e-AES}(pw \ \parallel pw\_new \parallel s \parallel mac_{msg}, key_{enc}) \tag{7.78}$$

The user resets the PWD_SYN operation timer and sends the message as follows to the M2M server:

$$< PWD\_SYN, t\_uid, enc_{msg} > \tag{7.79}$$

The M2M server collects t_uid data and controls the user's status. If the user is $LOCKED$ or not equal to $ONLINE$, the operation will be terminated by the M2M server. Otherwise, the M2M server will acquire keys and decrypt PWD_SYN and verify the MAC signal as follows:

$$[pw \ \parallel pw\_new \parallel s \parallel mac_{msg}] = F_{d-AES}(enc_{msg}, key_{enc}) \tag{7.80}$$

$$mac'_{msg} = F_{H-MAC}(PWD\_SYN \parallel t\_uid \parallel pw \ \parallel pw\_new \parallel s, key_{mac}) \tag{7.81}$$

If the received mac_{msg} does not match the calculated mac'_{msg}, then the operation is terminated by the M2M server. Otherwise, AES key will be calculated by the M2M server to confirm that the password is right:

$$key_{AES} = [\, F_{SHA1}(k \parallel \ t\_uid \parallel \ s)]^n \tag{7.82}$$

$$p' = F_{e-AES}(pw, key_{AES}) \tag{7.83}$$

Operations until this step is in Figure 7.16

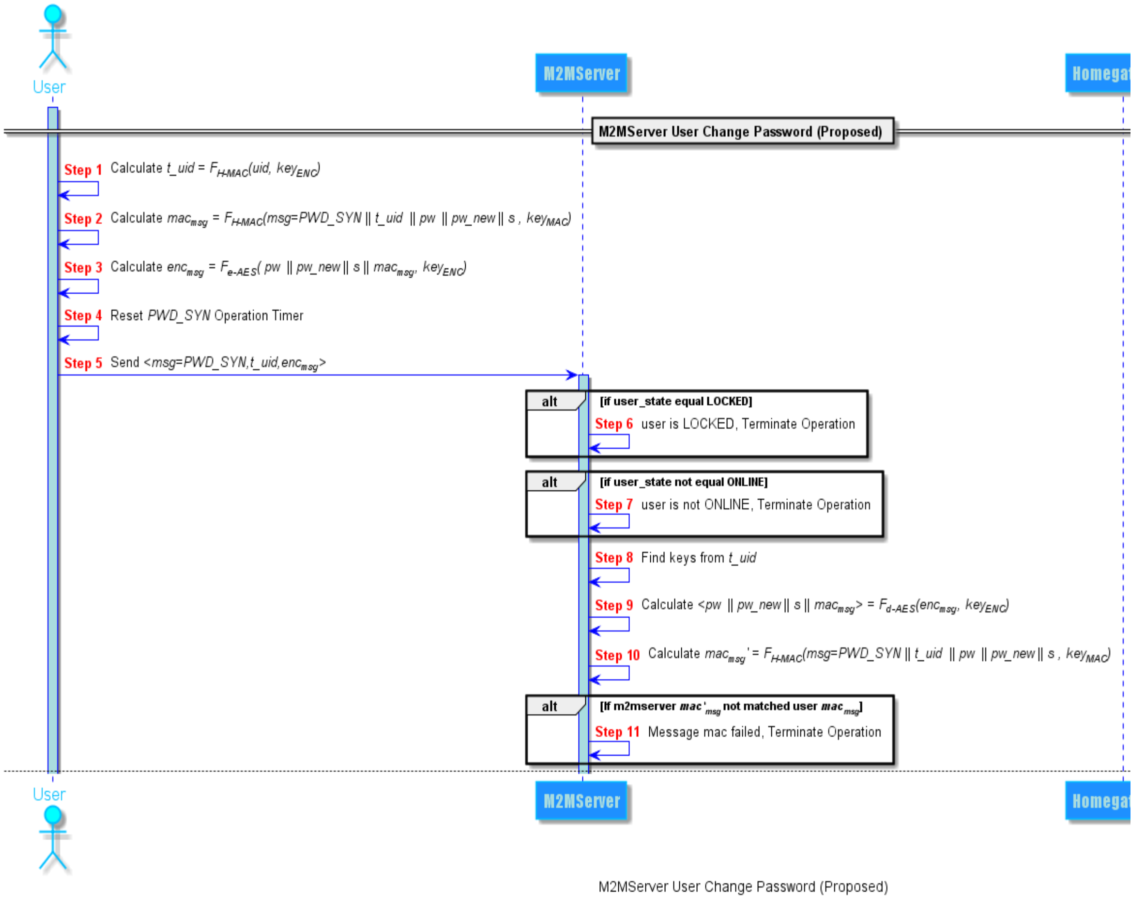

User
M2MServer
Homega
M2MServer User Change Password (Proposed)
Step 1 Calculate t_uid = F_H-MAC(uid, key_ENC)
Step 2 Calculate mac_msg = F_H-MAC(msg=PWD_SYN || t_uid || pw || pw_new || s , key_MAC)
Step 3 Calculate enc_msg = F_e-AES(pw || pw_new || s || mac_msg, key_ENC)
Step 4 Reset PWD_SYN Operation Timer
Step 5 Send <msg=PWD_SYN,t_uid,enc_msg>
alt
[if user_state equal LOCKED]
Step 6 user is LOCKED, Terminate Operation
alt
[if user_state not equal ONLINE]
Step 7 user is not ONLINE, Terminate Operation
Step 8 Find keys from t_uid
Step 9 Calculate <pw || pw_new || s || mac_msg> = F_d-AES(enc_msg, key_ENC)
Step 10 Calculate mac_msg' = F_H-MAC(msg=PWD_SYN || t_uid || pw || pw_new || s , key_MAC)
alt
[If m2mserver mac'_msg not matched user mac_msg]
Step 11 Message mac failed, Terminate Operation
User
M2MServer
Homega
M2MServer User Change Password (Proposed)

Figure 7.16: Proposed M2M Scheme User Change Password Flow-1

If stored p is not equal to p', the verification of the password fails and the M2M server ends the processing. Otherwise, a new salt s_{new} is chosen and a new key'_{AES} is calculated by the M2M server. The M2M server finally encrypts the password for record; as follows:

$$key'_{AES} = [\, F_{SHA1}(k \parallel \ t\_uid \parallel \ s_{new})]^{n} \tag{7.84}$$

$$p'_{new} = F_{e-AES}(pw\_new\ , key'_{AES}) \tag{7.85}$$

The M2M server records p'_{new} and prepares MAC and encrypted data for the PWD_ACK message. The ready signal is forwarded as follows to the customer unit:

$$mac_{msg} = F_{H-MAC}(PWD\_ACK \parallel s_{new}, key_{mac}) \tag{7.86}$$

$$enc_{msg} = F_{e-AES}(s_{new} \parallel mac_{msg}, key_{enc}) \tag{7.87}$$

$$< PWD\_ACK, enc_{msg} > \tag{7.88}$$

Operations until this step is in Figure 7.17.

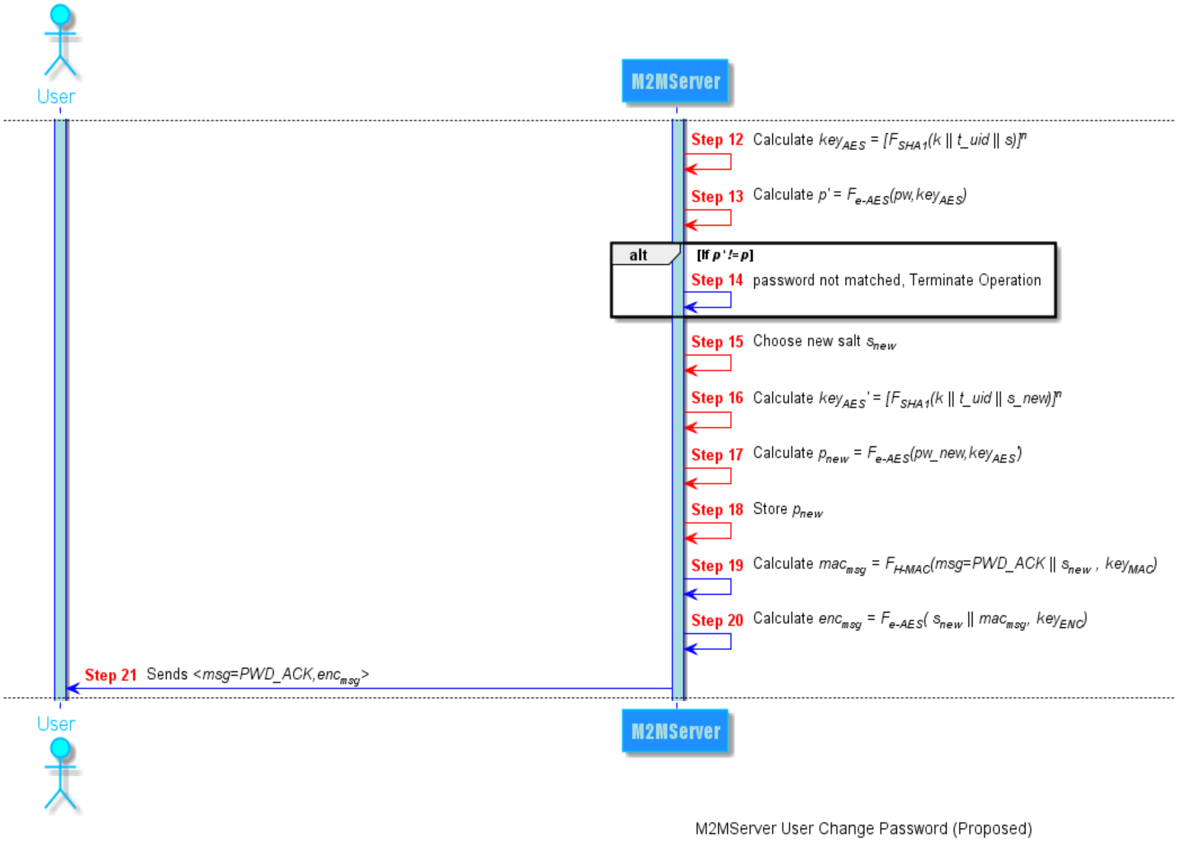

User
M2MServer
Step 12 Calculate $key_{AES} = [F_{SHA1}(k \parallel t\_uid \parallel s)]^n$
Step 13 Calculate $p' = F_{e\text{-}AES}(pw, key_{AES})$
alt
[If $p' != p$]
Step 14 password not matched, Terminate Operation
Step 15 Choose new salt s_{new}
Step 16 Calculate $key_{AES}' = [F_{SHA1}(k \parallel t\_uid \parallel s\_new)]^n$
Step 17 Calculate $p_{new} = F_{e\text{-}AES}(pw\_new, key_{AES})$
Step 18 Store p_{new}
Step 19 Calculate $mac_{msg} = F_{H\text{-}MAC}(msg{=}PWD\_ACK \parallel s_{new}, key_{MAC})$
Step 20 Calculate $enc_{msg} = F_{e\text{-}AES}(s_{new} \parallel mac_{msg}, key_{ENC})$
Step 21 Sends $<msg{=}PWD\_ACK, enc_{msg}>$
User
M2MServer
M2MServer User Change Password (Proposed)

Figure 7.17: Proposed M2M Scheme User Change Password Flow-2

The client equipment verifies that the processing period in PWD_SYN is within the appropriate range. If not, the fail flag will be set. If the test fails then, the system skips MAC validation and replies PWD_NACK. If not, the PWD_ACK MAC is validate by the client unit as follows:

$$[s_{new} \parallel mac_{msg}] = F_{d-AES}(enc_{msg}, key_{enc}) \tag{7.89}$$

$$mac'_{msg} = F_{H-MAC}(PWD\_ACK \parallel s_{new}, key_{mac}) \tag{7.90}$$

The user device sets err flag to true and sends PWD_NACK to the M2M server if the obtained mac_{msg} and calculated mac'_{msg} do not match; as follows:

$$mac_{msg} = F_{H-MAC}(PWD\_NACK \parallel t\_uid, key_{mac}) \tag{7.91}$$

$$< PWD\_NACK, t\_uid, mac_{msg} > \tag{7.92}$$

If a MAC fails, the server rejects client request. The M2M server checks the PWD_NACK MAC structure. If not, p_{new} is restored to p and the device's status is set to the prior condition by the server. If $fail\_flag$ is not true, s_new will be saved by the utiliser. Figure 7.18 operations are performed until this phase.

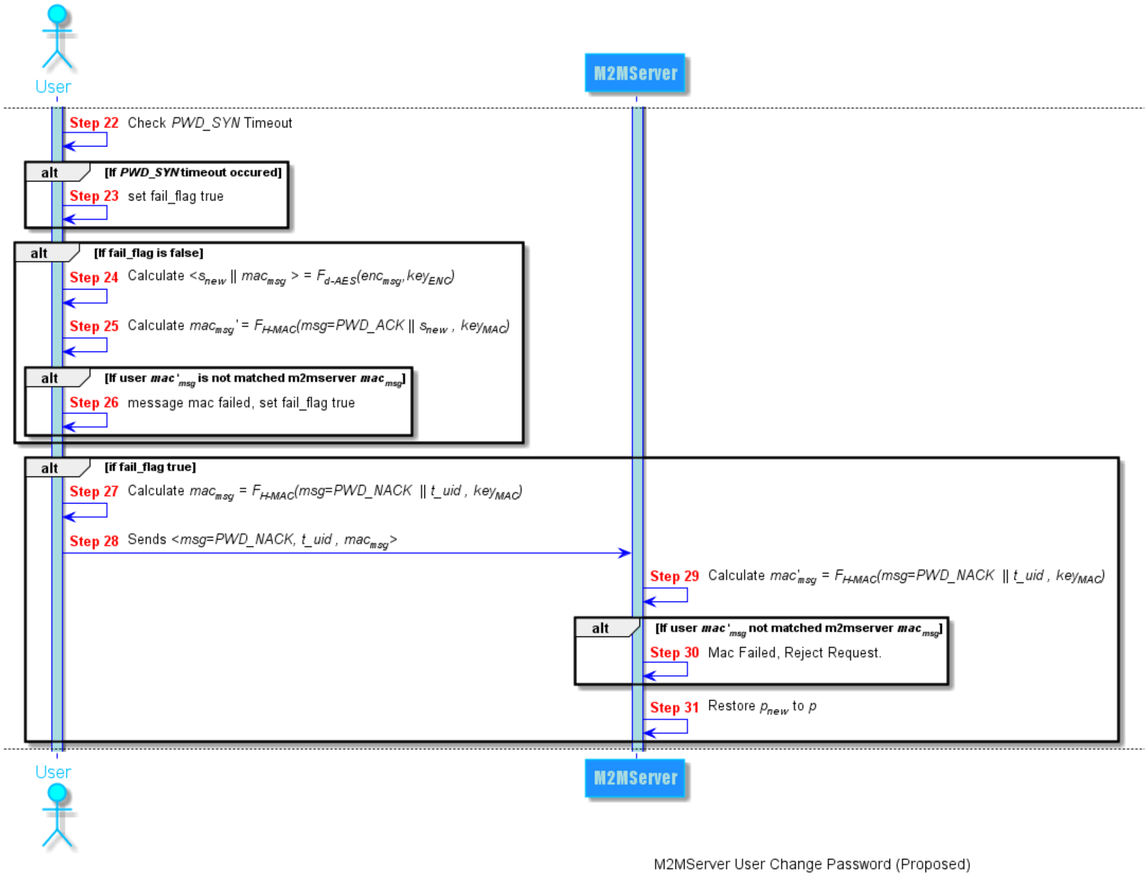

User
M2MServer
Step 22 Check PWD_SYN Timeout
alt [If PWD_SYN timeout occured]
Step 23 set fail_flag true
alt [If fail_flag is false]
Step 24 Calculate <s_{new} || mac_{msg} > = $F_{d-AES}(enc_{msg}, key_{ENC})$
Step 25 Calculate mac_{msg}' = F_{H-MAC}(msg=PWD_ACK || s_{new} , key_{MAC})
alt [If user mac'_{msg} is not matched m2mserver mac_{msg}]
Step 26 message mac failed, set fail_flag true
alt [if fail_flag true]
Step 27 Calculate mac_{msg} = F_{H-MAC}(msg=PWD_NACK || t_uid , key_{MAC})
Step 28 Sends <msg=PWD_NACK, t_uid , mac_{msg}>
Step 29 Calculate mac'_{msg} = F_{H-MAC}(msg=PWD_NACK || t_uid , key_{MAC})
alt [If user mac'_{msg} not matched m2mserver mac_{msg}]
Step 30 Mac Failed, Reject Request.
Step 31 Restore p_{new} to p
User
M2MServer
M2MServer User Change Password (Proposed)

Figure 7.18: Proposed M2M Scheme User Change Password Flow-3

In contrast, the M2M server with MAC is prepared with the following PWD_OK:

$$mac_{msg} = F_{H-MAC}(PWD\_OK \parallel t\_uid, key_{mac}) \tag{7.93}$$

$$< PWD\_OK, t\_uid, mac_{msg} > \tag{7.94}$$

The PWD_OK Message MAC is checked by the M2M server as:

$$mac'_{msg} = F_{H-MAC}(PWD\_OK \parallel t\_uid, key_{mac}) \tag{7.95}$$

The M2M server establishes $NEW\_PASSWORD$ to the client unit state and $REVOKED$ to the home gate way condition if the mac_{msg} obtained is equivalent to the calculated mac'_{msg}. The server denies the applications otherwise. With the RE_JOIN request from the M2M server, the home gateway key is revoked and the RE_JOIN message is compiled by MAC and then forwarded to the home gateway. The home portal system will start reassembling session key once the RE_JOIN notification MAC has been checked by the home portal device:

$$mac_{msg} = F_{H-MAC}(RE\_JOIN \parallel t\_hid, key_{mac}) \tag{7.96}$$

$$< RE\_JOIN, mac_{msg} > \tag{7.97}$$

Operations until this step is in Figure 7.19.

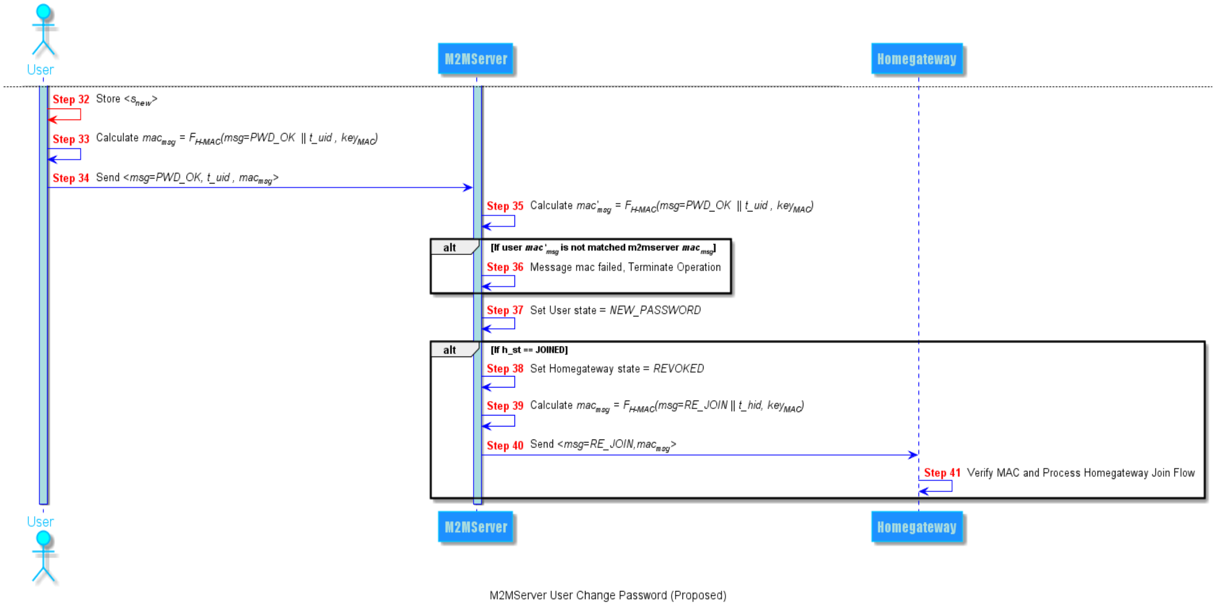

User
M2MServer
Homegateway
Step 32 Store <s_new>
Step 33 Calculate mac_msg = F_H-MAC(msg=PWD_OK || t_uid , key_MAC)
Step 34 Send <msg=PWD_OK, t_uid , mac_msg>
Step 35 Calculate mac'_msg = F_H-MAC(msg=PWD_OK || t_uid , key_MAC)
alt
[If user mac'_msg is not matched m2mserver mac_msg]
Step 36 Message mac failed, Terminate Operation
Step 37 Set User state = NEW_PASSWORD
alt
[If h_st == JOINED]
Step 38 Set Homegateway state = REVOKED
Step 39 Calculate mac_msg = F_H-MAC(msg=RE_JOIN || t_hid, key_MAC)
Step 40 Send <msg=RE_JOIN,mac_msg>
Step 41 Verify MAC and Process Homegateway Join Flow
User
M2MServer
Homegateway
M2MServer User Change Password (Proposed)

Figure 7.19: Proposed M2M Scheme User Change Password Flow-4

7.3.9 Proposed M2M Scheme User Logout

The one of the most critical feature is the user unit secure logout for unwanted activities. The user device calculates MAC requests for the M2M server for the temporary user ID (t_uid) and for the $LOGOUT\_SYN$ server:

$$mac_{msg} = F_{H-MAC}(LOGOUT\_SYN \parallel t\_uid, key_{mac}) \tag{7.98}$$

The device sends the M2M server with $LOGOUT\_SYN$ application:

$$< LOGOUT\_SYN, t\_uid, mac_{msg} > \tag{7.99}$$

The M2M server searches for the t_uid keys prepares the $LOGOUT\_SYN$ $mac'_{msg}$for the continuing operation and compares it to the recipients mac_{msg}:

$$mac'_{msg} = F_{H-MAC}(LOGOUT\_SYN \parallel t\_uid, key_{mac}) \tag{7.100}$$

The M2M server denies the procedure when the MAC is inconsistent. If not, the client unit is set to OFFLINE and the client system equipment an answer to LOGOUT ACK as follows:

$$mac_{msg} = F_{H-MAC}(LOGOUT\_ACK \parallel t\_uid, key_{mac}) \tag{7.101}$$

$$< LOGOUT\_ACK, t\_uid, mac_{msg} > \tag{7.102}$$

When the user device gets $LOGOUT\_ACK$, it validates MAC, as follows:

$$mac'_{msg} = F_{H-MAC}(LOGOUT\_ACK \parallel t\_uid, key_{mac}) \tag{7.103}$$

If the mac_{msg} obtained and the mac'_{msg} calculated match, the client device will update its condition to the situation $OFFLINE$. If not, the application would be dropped.

Operations until this step is in Figure 7.20.

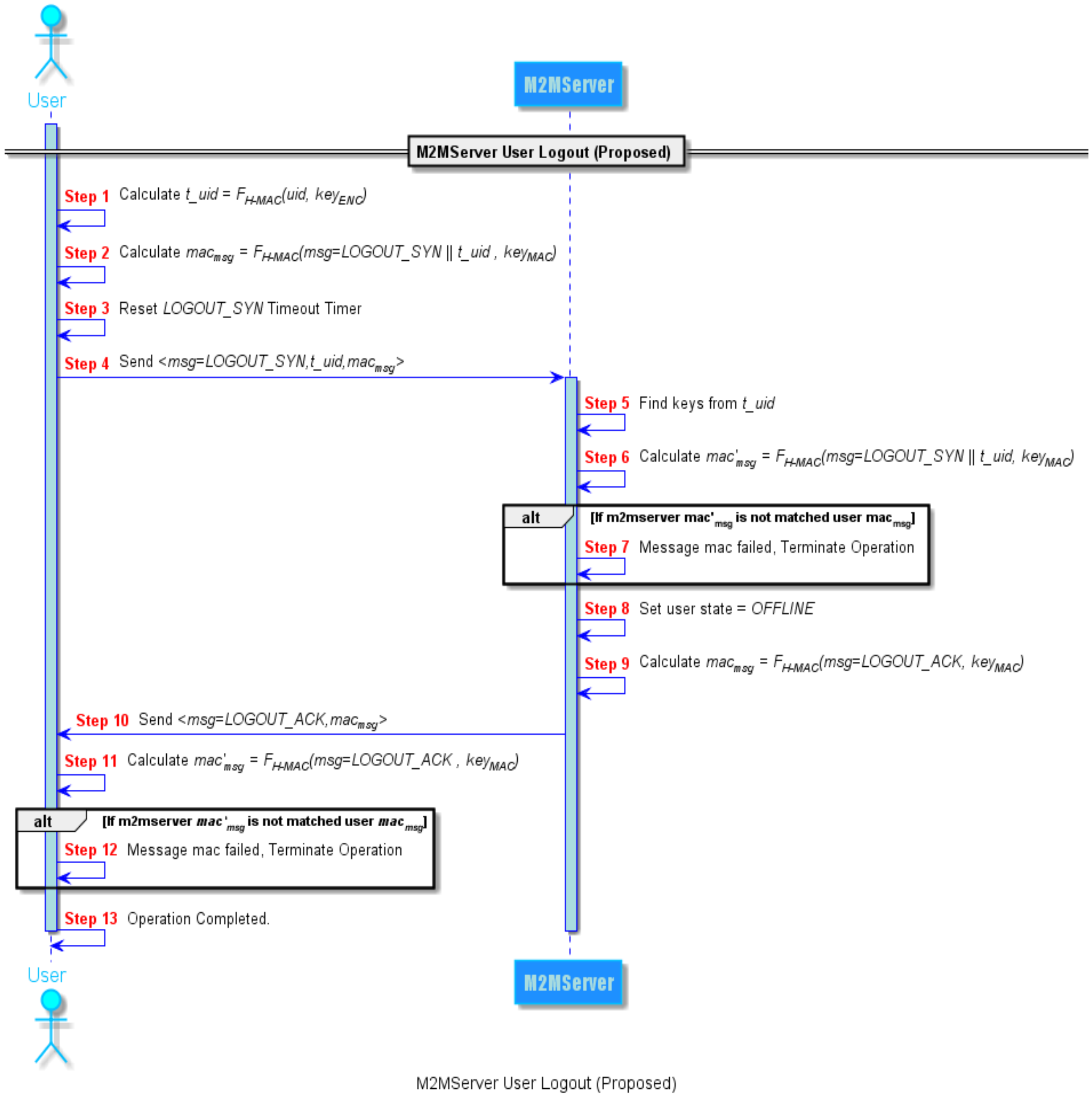

User
M2MServer
M2MServer User Logout (Proposed)
Step 1 Calculate $t\_uid = F_{H\text{-}MAC}(uid, key_{ENC})$
Step 2 Calculate $mac_{msg} = F_{H\text{-}MAC}(msg{=}LOGOUT\_SYN \| t\_uid, key_{MAC})$
Step 3 Reset *LOGOUT_SYN* Timeout Timer
Step 4 Send <*msg=LOGOUT_SYN,t_uid,*mac_{msg}>
Step 5 Find keys from *t_uid*
Step 6 Calculate $mac'_{msg} = F_{H\text{-}MAC}(msg{=}LOGOUT\_SYN \| t\_uid, key_{MAC})$
alt
[If m2mserver mac'_{msg} is not matched user mac_{msg}]
Step 7 Message mac failed, Terminate Operation
Step 8 Set user state = *OFFLINE*
Step 9 Calculate $mac_{msg} = F_{H\text{-}MAC}(msg{=}LOGOUT\_ACK, key_{MAC})$
Step 10 Send <*msg=LOGOUT_ACK,*mac_{msg}>
Step 11 Calculate $mac'_{msg} = F_{H\text{-}MAC}(msg{=}LOGOUT\_ACK, key_{MAC})$
alt
[If m2mserver mac'_{msg} is not matched user mac_{msg}]
Step 12 Message mac failed, Terminate Operation
Step 13 Operation Completed.
User
M2MServer
M2MServer User Logout (Proposed)

Figure 7.20: Proposed M2M Scheme User Logout Flow

7.3.10 Proposed M2M Scheme Home Gateway Re-Join Network

This procedure is processed by home gateway hardware after changing passwords. The M2M server establishes the home gateway machine to $REVOKED$ and prepares for the home gateway application the following RE_JOIN signal:

$$mac_{msg} = F_{H-MAC}(RE\_JOIN \parallel t\_hid, key_{mac}) \tag{7.104}$$

$$< RE\_JOIN, mac_{msg} > \tag{7.105}$$

The unit of the home gateway calculates t_hid and verifies RE_JOIN MAC. If MAC valid, the unit of the home portal runs the flow of the home gateway join network. operations until this step is in Figure 7.21.

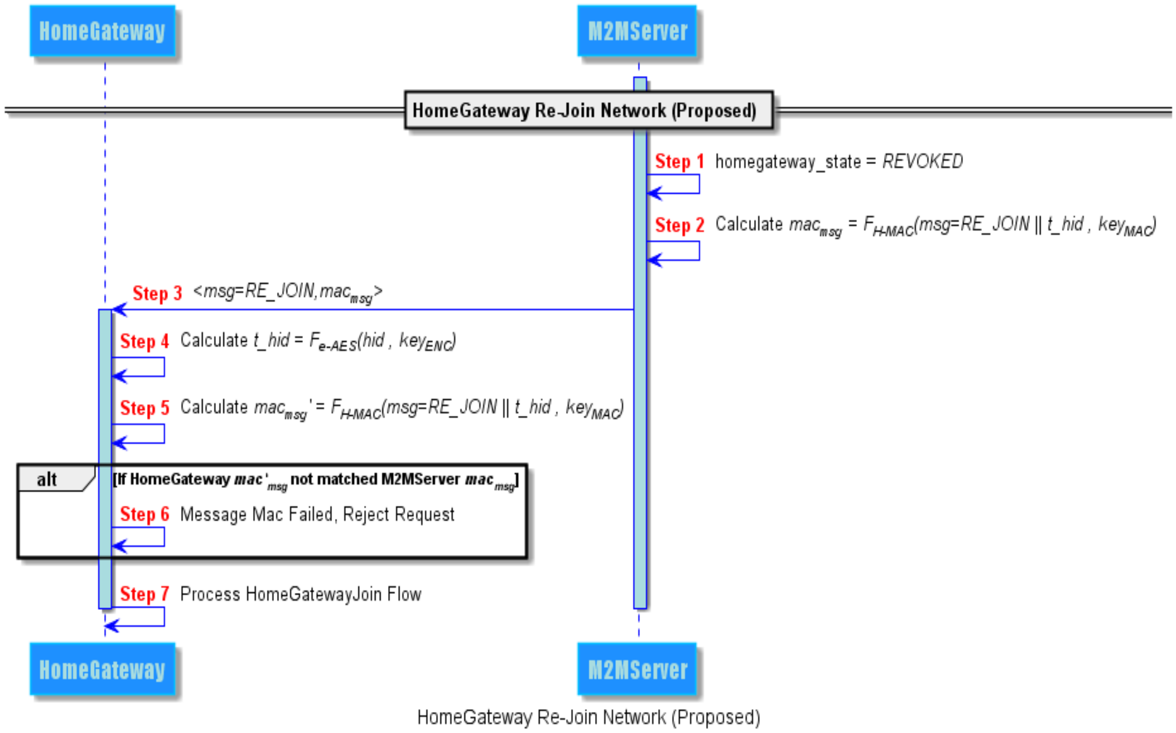

HomeGateway
M2MServer
HomeGateway Re-Join Network (Proposed)
Step 1 homegateway_state = REVOKED
Step 2 Calculate mac_{msg} = F_{H-MAC}(msg=RE_JOIN || t_hid , key_{MAC})
Step 3 <msg=RE_JOIN,mac_{msg}>
Step 4 Calculate t_hid = F_{e-AES}(hid , key_{ENC})
Step 5 Calculate mac_{msg}' = F_{H-MAC}(msg=RE_JOIN || t_hid , key_{MAC})
alt
[If HomeGateway mac'_{msg} not matched M2MServer mac_{msg}]
Step 6 Message Mac Failed, Reject Request
Step 7 Process HomeGatewayJoin Flow
HomeGateway
M2MServer
HomeGateway Re-Join Network (Proposed)

Figure 7.21: Proposed M2M Scheme Home Gateway Re-Join Network Flow

8. PROPOSED VANET SCHEME

We propose an ESAR scheme in this section, to supply the missing elements of the EMAP scheme. The ESAR notation in Table 8.1 and the following arrangements for our contributions:

Table 8.1: Notation of Proposed ESAR scheme

Notation	**Description**
$x \mid\mid y$	Concatenation of x and y
$len(x)$	Length of x in bytes
$cnt(x)$	Count of x
$[x]^y$	Most significant y bits of string x
x and $\tilde{x}$	$\tilde{x}$ shows a new value of x property
$sig_y(x)$	Signature of x with y's secret key
$CERT_u^i$	i_{th} Anonymous certificate of OBU_u by TA
P, Q	EKI KeyPair Public Key (P) and Private Key (Q) Generator $\in \mathbb{G}_1$ Shows defined parameters for the $secp256k$ curve
REV_{check}	Revocation check value that calculates by using $F_{mac.HMAC}(.)$ or $F_{mac.AES}(.)$ functions
M	The unique private key ID for update required keypair. Transport in the K_{msg}. The content randomly generated 8 bytes long and unique for each key pair
RID_u	Real Identity Data (8 bytes) of OBU_u. Randomly assigned
PID_u	Pseudo Identification Data of OBU_u. Generated with $[F_{mac.HMAC}(RID_u, K_{pid})]^{32}$ and used in anonymous certificates mentioned in [85]
K_{pid}	PID generation secret key (16 bytes). Located on TA.
$s,\ P_o$	TA master public (P_o) and private (s) key pair. TA sign certificates and messages with this and publish the public key to OBUs for verification.
PK_u^i, SK_u^i	i_{th} PKI public (PK_u^i) and private (SK_u^i) keypair of OBU_u. Generated by $F_{gen.PKI}(.)$ function.
K_i^-, K_i^+	i_{th} EKI public (K_i^+) and private (K_i^-) keypair of OBU. Generated by $F_{gen.EKI}(.)$ function.

Notation	Description
K_g	Revocation check key $\in \mathbb{G}_2$
K_{im}	Intermediate key to update $K_g \in \mathbb{G}_2$
K_{msg}	TA Key Update Message
CRL	Revoked certificates PID list for PKI.
REV_{msg}	TA Signed Revocation Message
$F_{INVMOD}(x, y)$	Inverse Module of x base to y
$F_{RAND}(.)$	Secure Random Function
$F_{SHA1}(x)$	The cryptographic SHA1 hash of x
$F_{enc.AES}(x, y)$	AES CBC function to encrypt x with y as the key
$F_{dec.AES}(x, y)$	AES CBC function to decrypt x with y as the key
$F_{mac.HMAC}(x, y)$	Using Keyed cryptographic MAC operation with SHA-1 for x with y as the key
$F_{mac.AES}(x, y)$	Using AES CBC MAC operation for x with y as the key
$F_{enc.ECIES}(x, y)$	Using ECIES function to encrypt x with y as the key
$F_{dec.ECIES}(x, y)$	Using ECIES function to decrypt x with y as the key
$F_{sig.ECDSA}(x, y)$	Using ECDSA scheme create a signature of x with y as the private key
$F_{val.ECDSA}(\mathrm{x, y, z})$	Using ECDSA scheme to verify signature x of y with z as the public key
$[\widetilde{K^+}, \widetilde{K^-}] = F_{upd.EKI}(K^+, K^-, v)$	Update EKI Key Pairs $K^+, K^-$with hash value v to get new key pair as $\widetilde{K^+}, \widetilde{K^-}$
$[PK, SK] = F_{gen.PKI}(rnd)$	ECDH PKI Public (PK) and Private (SK) Key Pair Generation with Random value as rnd
$[K^+, K^-] = F_{gen.EKI}(rnd)$	ECDH EKI Public (K^+) and Private (K^-) Key Pair Generation with Random value as rnd
C	OBU PKI anonymous certificates size
j	Hash chain size
m	EKI key pair size of OBU
l	EKI key pair pool size
$v \in \mathbb{Z}_q^*$	Hash Chain Initial Secret Value
V	Hash Chain Set

Notation	Description
$U_p,\ U_s$	Public (U_p) and Private (U_s) key pool for EKI
RP_u, RS_u	Public (RP_u) and Private (RS_u) keys deployed to OBU_u
$\mathcal{M}$	Message between OBUs
T_{stamp}	Current timestamp value
ver,ver_{last}, $ver_{\|missed}$	Current, last received and missed revocation versions.
v_{j-ver}	Hash chain value of revocation version ver.
$ID_{rev_{key}}, ID_{rev_{key\|missed}}$	Identities of the current and missed revoked key pairs

8.1 MERGE ENCRYPTED REVOCATION TRANSPORT KEY AND MISSED REVOKED KEY ID LIST REQUESTS

Our first contribution to the design [10] is merging two requests to improve performance. During the revocation process, if an OBU_s do not own key, which defined with ID in the message to recover $\widetilde{K}_g$ then, send a signed request to get this secret key from neighboring OBU_r's. After receiving encrypted $\widetilde{K}_g$, it decrypts and prepares for the key update. In the next stage, during the key update process, OBU_s check if it missed a revocation then make a new signed request to get missing revocation key IDs from neighboring OBU_rs. After receiving these IDs, OBU_s update own compromised keys. We send the following request to receive both encrypted $\widetilde{K}_g$ and $ID_{rev_{key|missed}}$ in a single signed message with time, signature, up-to-date and other security validation, thus:

$$\begin{aligned} < msg = KG\_KEY\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp} \parallel PID_s^i \parallel PK_s^i \\ \parallel sig_{TA}(PID_s^i \parallel PK_s^i) \parallel sig_s(msg = KG\_KEY\_REQ \\ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp}) > \end{aligned} \tag{8.1}$$

OBU_r's detect sender was missing revocation status After message signature, OBU_s certificate and message time validation by checking the following equation:

$$ver_{received} \neq ver_{stored} + 1 \tag{8.2}$$

If OBU_s is valid but do not have missing revocation then only send ECIES encrypted $\widetilde{K}_g$ by using OBU_s's PK_s^i, in the following response:

$$\begin{aligned} < msg = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \parallel\ & F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \\ & \parallel 0 \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \parallel sig_{TA}(PID_r^i \parallel\ PK_r^i) \parallel sig_r(msg \\ & = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \\ & \parallel\ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \parallel 0 \parallel T_{stamp}) > \end{aligned} \tag{8.3}$$

In above, zero denotes the represent empty $ID_{rev_{key|missed}}$ length. If missing revocations detected for OBU_s then following buffer prepared for $ID_{rev_{key|missed}}$ as follows:

$$\begin{aligned} & ID_{rev_{key|missed}} =< ver_i\ \parallel len_{\ i} \parallel M_{\ i_1} \parallel M_{\ i_2} \parallel \cdots \parallel M_{\ i_n} \parallel ver_j\ \parallel len_{\ j} \parallel \\ & M_{\ j_1} \parallel M_{\ j_2} \parallel \cdots \parallel M_{\ j_n} \parallel \cdots \parallel ver_m\ \parallel len_{\ m} \parallel M_{\ m_1} \parallel M_{\ m_2} \parallel \cdots \parallel \\ & M_{\ m_n}\ >| ver_{stored} \leq i < j < m \leq ver_{received} - 1 \end{aligned} \tag{8.4}$$

This buffer placed in response buffer as follows:

$$\begin{aligned} < msg = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \parallel\ & F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \\ & \parallel len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}}\ \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \\ & \parallel sig_{TA}(PID_r^i \parallel\ PK_r^i) \parallel sig_r(len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \\ & \parallel\ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \parallel len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}} \\ & \parallel T_{stamp}) > \end{aligned} \tag{8.5}$$

This amendment eliminates the unnecessary sign action and all expected variables sign with a single acknowledgment. This process is faster than the EMAP offered. Also, our improvement decrease network congestion because each request contains certificates. However, if OBU_s own key, which defined with ID in the message to recover $\widetilde{K_g}$ but has missing revocations then sends the following message to only get $ID_{rev_{key|missed}}$:

$$< msg = REV\_ID\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp} \parallel PID_s^i \parallel PK_s^i \parallel sig_{TA}(PID_s^i \parallel PK_s^i) \parallel sig_s(msg = REV\_ID\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp}) > \quad (8.6)$$

Receiver OBU will process CRL, certificate and signature controls and then if everything correct and any key ID exist then will return following response:

$$< msg = REV\_ID\_RSP \parallel len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}} \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \parallel sig_{TA}(PID_r^i \parallel PK_r^i) \parallel sig_r(len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}} \parallel T_{stamp}) > \quad (8.7)$$

8.2 ENCRYPTED REVOCATION TRANSPORT KEY SOURCE VERIFICATION

The second contribution to the EMAP is that we discussed before. If an OBU_s do not own key, which defined with ID in the message during revocation process to recover $\widetilde{K_g}$ then, send a signed request to get $\widetilde{K_g}$ from neighboring OBU_r's. After receiving encrypted $\widetilde{K_g}$, it decrypts and with own private key and uses plain $\widetilde{K_g}$ to decrypt $F_{enc.AES}(v_{j-ver}, [\widetilde{K_g}]^{128})$ which served as $enc_{\widetilde{K_g}}(v_{j-ver})$ in EMAP, to get v_{j-ver}. The integrity of $\widetilde{K_g}$ is essential to get correct v_{j-ver}. During the revocation process, an attacker can send corrupted buffer as encrypted $\widetilde{K_g}$ without integrity check OBU_r can corrupt own keys, thus:

$$[\widetilde{K^+}, \widetilde{K^-}] = F_{upd.EKI}(K^+, K^-, {v_{j-ver}}^{corrupted}) \quad (8.8)$$

There is no recovery flow defined in EMAP. Besides, this can occur by a system malfunction also due to environmental factors. As we described in before section, we propose the following response messages for OBU_r to verify sender OBU_s validity and $\widetilde{K_g}$ integrity with expired time validation:

$$\begin{aligned} < msg = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \parallel\ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \\ \parallel len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}} \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \\ \parallel sig_{TA}(PID_r^i \parallel\ PK_r^i) \parallel sig_r(msg = KG\_KEY\_RSP \\ \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \parallel\ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \\ \parallel len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}} \parallel T_{stamp}) > \end{aligned} \tag{8.9}$$

$$\begin{aligned} < msg = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \parallel\ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \\ \parallel 0 \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \parallel sig_{TA}(PID_r^i \parallel\ PK_r^i) \parallel sig_r(msg \\ = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \\ \parallel\ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \parallel 0 \parallel T_{stamp}) \end{aligned} \tag{8.10}$$

In our simulation, an attacker, send damaged buffer as encrypted $\widetilde{K}_g$ and without $\widetilde{K}_g$ source, and integrity validation receiver OBU updates own keys, thus:

$$\left[\widetilde{K^+}, \widetilde{K^-}\right] = F_{upd.EKI}(K^+, K^-, {v_{j-ver}}^{corrupted}) \tag{8.11}$$

Moreover, never authenticate to other OBUs. Also, these damaged OBUs create network congestion on VANETs by sending invalid message signatures to others. Each revocation iteration with corrupted encrypted $\widetilde{K}_g$ increase corrupted OBU size in the network, and after a while, all VANET services will be disabled for each OBU. We disable this kind of issues with this contribution.

8.3 RELIABLE OPERATION PROCESSING AND STATE MANAGEMENT

Service reliability is the third input. EMAP does not define the event plus timeout errors, message acceptance ,and message identifier pieces. Also, management by TA and OBU is low. No estate is available to maintain the OBU condition. We introduced lacking timestamp timeout values The timeout measure can also be used to identify attacks by tracking operation processing events used by time out control. This timeout can also be used. Every one-byte-marked message. With this identifier, items can assess applications. Identifiers of the ESAR system in Table 8.2

Table 8.2: Proposed ESAR Scheme Message Identifiers

Message Name	Message Phase	Message Sender	Message Receiver
REVOC_REQ	Revocation	TA	OBU
TEL_MSG	Authentication	OBU	OBU
KG_KEY_REQ	Revocation	OBU	OBU
KG_KEY_RSP	Revocation	OBU	OBU
REV_ID_REQ	Revocation	OBU	OBU
REV_ID_RSP	Revocation	OBU	OBU

We also modified the present layout status of the devices. Figure 8.1 and Figure 8.2 contain enhanced conditions within the suggested protocol.

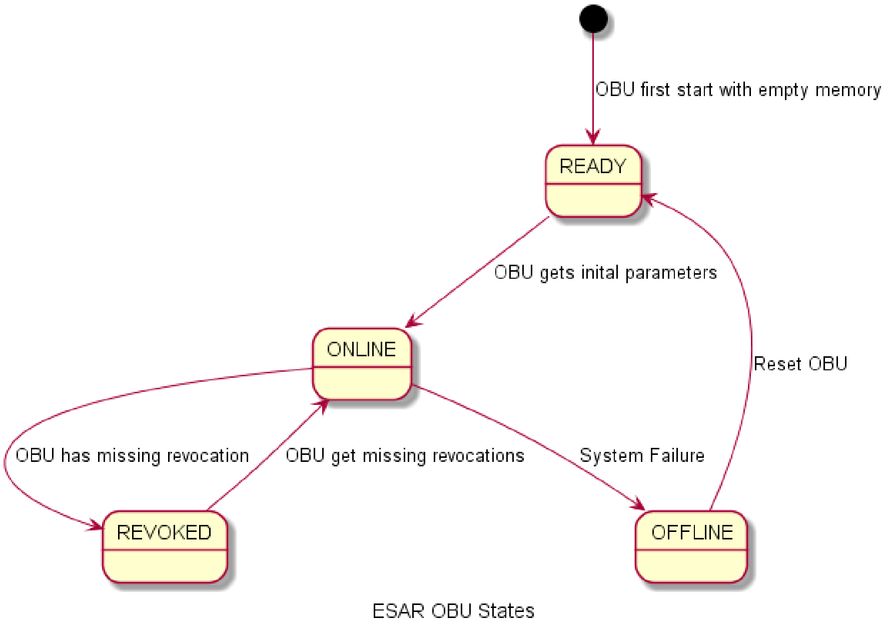

OBU first start with empty memory
READY
OBU gets inital parameters
ONLINE
Reset OBU
OBU has missing revocation
OBU get missing revocations
System Failure
REVOKED
OFFLINE
ESAR OBU States

Figure 8.1: On-Board-Unit State Processing

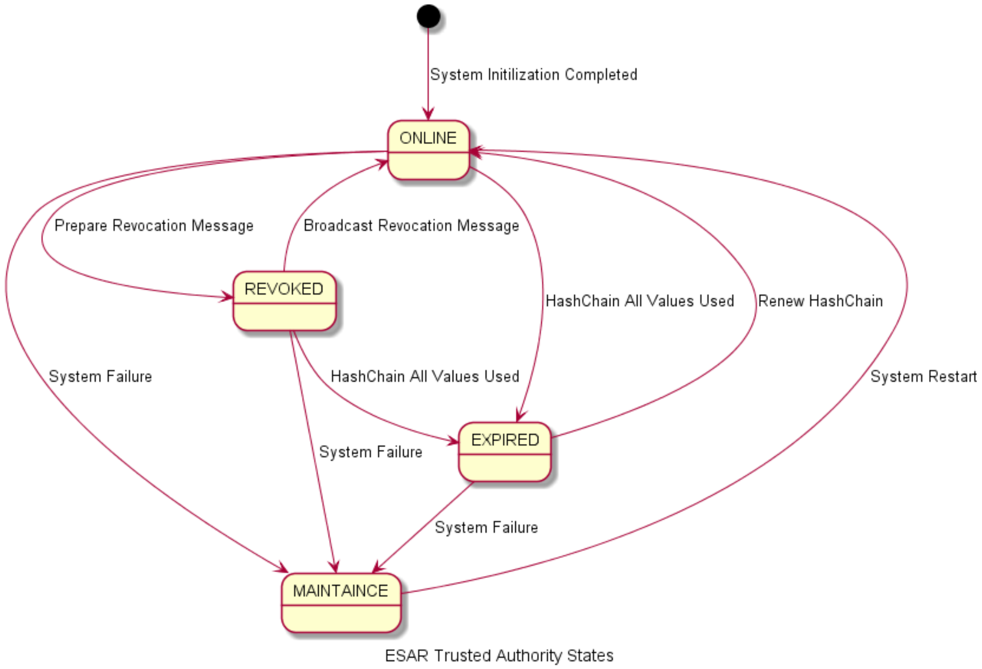

System Initilization Completed
ONLINE
Prepare Revocation Message
Broadcast Revocation Message
REVOKED
HashChain All Values Used
Renew HashChain
System Failure
HashChain All Values Used
System Restart
EXPIRED
System Failure
System Failure
MAINTAINCE
ESAR Trusted Authority States

Figure 8.2: Trusted Authority State Processing

8.4 REVOCATION VERSION CHECK

During the authentication process, we noticed that revoked OBUs authenticate each other in EMAP scheme. Because the authentication process does not check CRLs. This problem can cause send wrong telematics each other and TA would have the unmanaged network in VANETS. Also, this revoked OBUs create network congestion. Besides, there is no determined self up-to-date control that used for checking OBU has fresh $\widetilde{K}_g$ to send other OBUS. For this kind of reasons, the third enrichment is about adding up-to-date control by using $ver_{received}$ and ver_{stored}. During the authentication process, if an OBU has missing revocation, then we stop its message sending until it completes all revocations. This development avoids sending incorrect information and implement a managed network. Also, minimize network congestion. While message processing. If any telematics comes from other OBUs before extensive signature controls, we first check missing the revocation status, and if there is missing revocation, then we stop processing. This control protects this revoked OBU to get wrong information from an attacker and provide power saving on the overall network. During the revocation process, if an OBU_s request encrypted $\widetilde{K}_g$ from other OBU_r then before all validations, OBU_r checks own missing revocation status, and if there is a missing revocation, then it detects that own $\widetilde{K}_g$ is not fresh and stop processing message. If it sends own legacy $\widetilde{K}_g$ receiver OBU corrupt own EKI keys.

8.5 TRUSTED AUTHORITY REVOKED KEY SYNCHRONIZATION

During revocation, message construction revoked key IDs handled from TA data sources and received OBUs update their keys, but TA does not renew and synchronize own EKI key pairs after revocation, and in the next revocation this will cause a wrong key update. This operation not specified in EMAP revocation phase, but we implemented in our ESAR scheme and simulation. Following operation manners on both TA and OBU:

$$\left[\widetilde{K^+}, \widetilde{K^-}\right] = F_{upd.EKI}(K^+, K^-, v_{j-ver}) \tag{8.12}$$

moreover, TA replaces old keypairs with new ones.

8.6 DISTINGUISH REVOKED KEYPAIRS FOR EACH OBU

The OBU equipment can share EMAP EKI key pairs with other OBUs and this sharing randomly setup upon system initialization. During a revocation process, EMAP scheme revocation will affect many OBUs. These can cause maintainability problem. Because if we want to revoke two OBUs and each one is in a separate group we should revoke all these groups shared EKI pairs. For these reasons, we use unique EKI key pair for each OBU to manage revocation operations efficiently.

8.7 ENABLE TRACEABILITY WITH PRIVACY-PRESERVING

EMAP scheme utilizes anonymous certificates, and each certificate has arbitrary PID that related to RID in TA. During the authentication phase, distinct PIDs avoid traceability of data source. Telematics such as location updates comes with different PIDs, and each received message assume to come from different OBU. According to this reason receiver, OBU can calculate ghost traffic jam and autonomous systems can effect this kind of overlapped messages. For this purpose, we use single PID for each OBU that associated with RID. We opted to use keyed MAC algorithms for PID generation such as Cipher-based-MAC (C-MAC) [81] or the Keyed-Hash Message Authentication Code (H-MAC) [82] based on Data Encryption Standard (DES) or AES. This protection method implemented in [8]. We use a pre-shared key to device PID from RID that mentioned in The GLARM [4]. We have a modified H-MAC algorithm with the H-MAC-based one-time password (HOTP) of the same offset choice as described in [84]. For Secure Hash Algorithm 256 (SHA-256) and SHA-1 we evaluated the efficiency of the H-MAC setup, with 16-bytes, 8-bytes key with simulation introduced, which was shown in Table 7.2.

8.8 REORDER MESSAGE VERIFICATIONS TO INCREASE PERFORMANCE

In EMAP revocation phase, $ID_{rev_{key|missed}}$ request processing start with expensive comparisons such as certificate and message signature validation. After successful signature validations receiving entity checks own revoked key ID list. ID list search method is faster than the signature check. If we first check the ID list and then validate the signature, overall network performance will improve according to our simulation results. Also, in ESAR, we merged requests that we discussed before and then reordered validations to improve overall network performance.

8.9 PROPOSED ESAR SCHEME PHASES

There are three stages to our suggested system. We have conceived a virtually worthy application on the actual network and have taken care that the public channel does not have a safe interaction layer. In addition, EMAP stages are updated to quality and safety requirements as set out in section 6 above. In the ESAR system. The following explanations are provided for ESAR system stages:

8.9.1 Proposed ESAR Scheme TA and OBU Setup

During the setup procedure, TA chooses elliptic curve parameters (P, Q) and a hash function for operations. For our results and simulations, we used $secp256k$ elliptic curve parameters and SHA-1 function. These parameters are configurable in our implementation. First TA makes EKI key pair pool U_p, U_s. Each key generated with the following methods and each key pair distinguished by key ID (M):

$$[K^{+}, K^{-}] = F_{gen.EKI}(rnd) \tag{8.13}$$

$$F_{gen.EKI}(rnd) = \{K^{-} = rndQ \mid K^{+} = rnd^{-1}P\} \tag{8.14}$$

$$rnd = F_{RAND}(.) \tag{8.15}$$

$$rnd^{-1} = F_{INVMOD}(rnd, P.subGroupGen) \tag{8.16}$$

After EKI key pair generation TA generates a default revocation key K_g as follows:

$$t_1 = F_{RAND}(.) \tag{8.17}$$

$$t_2 = F_{RAND}(.) \tag{8.18}$$

$$K^{+}{}_{temp} = t_1{}^{-1}P \tag{8.19}$$

$$K^{-}{}_{temp} = t_1 Q \tag{8.20}$$

$$\mathrm{K_{im}} = \mathrm{t_2}\, K^{+}{}_{temp} \tag{8.21}$$

$$K_g = \mathrm{K_{im}}\, K^{-}{}_{temp} \tag{8.22}$$

Then TA generates PKI certification master key pair s, P_o as follows:

$$rnd = F_{RAND}(.) \tag{8.23}$$

$$[P_o, s] = F_{gen.PKI}(rnd) \tag{8.24}$$

$$F_{gen.PKI}(rnd) = \{\mathrm{s} = rnd \mid P_o = rndP\} \tag{8.25}$$

After key production procedures, TA makes hash chain $V = \{v_i \mid 0 \leq i \leq j\}$ and then start to generate anonymous certificates and deploy prepared keys and system parameters to OBUs. All generation flow until this step is in Figure 8.3

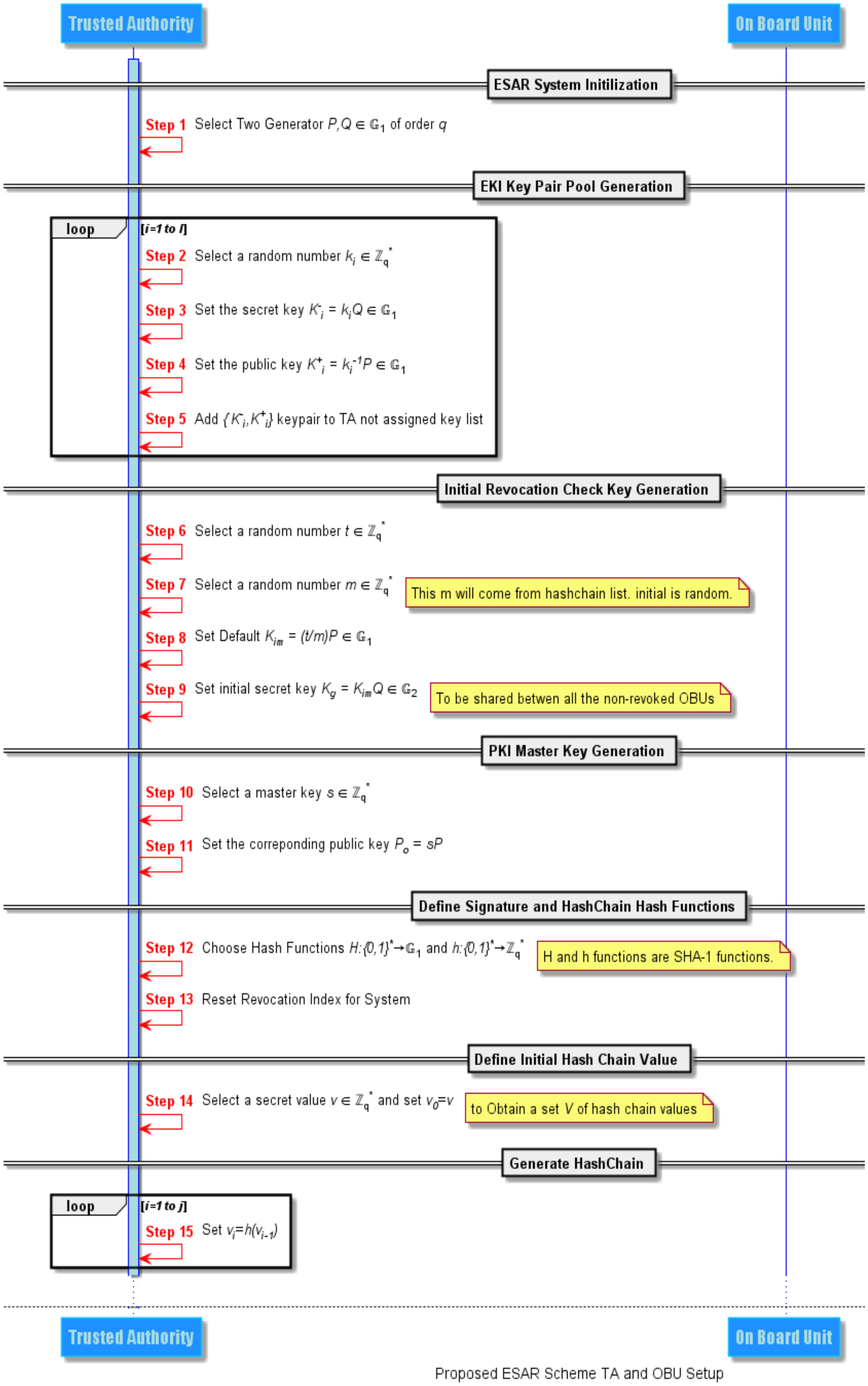

Trusted Authority
On Board Unit
ESAR System Initilization
Step 1 Select Two Generator $P,Q \in \mathbb{G}_1$ of order q
EKI Key Pair Pool Generation
loop
[i=1 to I]
Step 2 Select a random number $k_i \in \mathbb{Z}_q^*$
Step 3 Set the secret key $K^-_i = k_iQ \in \mathbb{G}_1$
Step 4 Set the public key $K^+_i = k_i^{-1}P \in \mathbb{G}_1$
Step 5 Add $\{K^-_i, K^+_i\}$ keypair to TA not assigned key list
Initial Revocation Check Key Generation
Step 6 Select a random number $t \in \mathbb{Z}_q^*$
Step 7 Select a random number $m \in \mathbb{Z}_q^*$
This m will come from hashchain list. initial is random.
Step 8 Set Default $K_{im} = (t/m)P \in \mathbb{G}_1$
Step 9 Set initial secret key $K_g = K_{im}Q \in \mathbb{G}_2$
To be shared betwen all the non-revoked OBUs
PKI Master Key Generation
Step 10 Select a master key $s \in \mathbb{Z}_q^*$
Step 11 Set the correponding public key $P_o = sP$
Define Signature and HashChain Hash Functions
Step 12 Choose Hash Functions $H:\{0,1\}^* \rightarrow \mathbb{G}_1$ and $h:\{0,1\}^* \rightarrow \mathbb{Z}_q^*$
H and h functions are SHA-1 functions.
Step 13 Reset Revocation Index for System
Define Initial Hash Chain Value
Step 14 Select a secret value $v \in \mathbb{Z}_q^*$ and set $v_0 = v$
to Obtain a set V of hash chain values
Generate HashChain
loop
[i=1 to j]
Step 15 Set $v_i = h(v_{i-1})$
Trusted Authority
On Board Unit
Proposed ESAR Scheme TA and OBU Setup

Figure 8.3: Proposed ESAR Scheme TA and OBU Setup Flow-1

This deployment security not recognized in both EMAP and ESAR schemes. TA randomly select an EKI key pair set from the pool that not assigned to other OBUs and deploys to each OBU and the following operation generate certificates for deploying to OBU:

$$PID = \left[F_{mac.HMAC}\left(RID, K_{pid}\right)\right]^{64} \tag{8.26}$$

$$rnd_u^i = \ F_{RAND}(.) \tag{8.27}$$

$$[PK_u^i, SK_u^i] = F_{gen.PKI}(rnd_u^i) \tag{8.28}$$

$$F_{gen.PKI}(rnd) = \left\{SK_u^i = rnd_u^i \ \middle| PK_u^i = rnd_u^i P\right\} \tag{8.29}$$

$$sig_{TA}\left(PID_u \parallel \ PK_u^i\right) = sH(PID_u \parallel \ PK_u^i) \tag{8.30}$$

$$cert_u^i(PID_u, PK_u^i, sig_{TA}(PID_u \parallel \ PK_u^i)) \tag{8.31}$$

PID is single for each OBU that derived from RID and 8 bytes. Each $cert_u^i$ and SK_u^i pairs deployed to OBU HSM with default K_g. Also, each OBU has configured to work with the same system parameters such as elliptic curve parameters P, Q and hash functions $H(.)$ and $h(.)$ that defined in setup. All deployment flow until this step is in Figure 8.4.

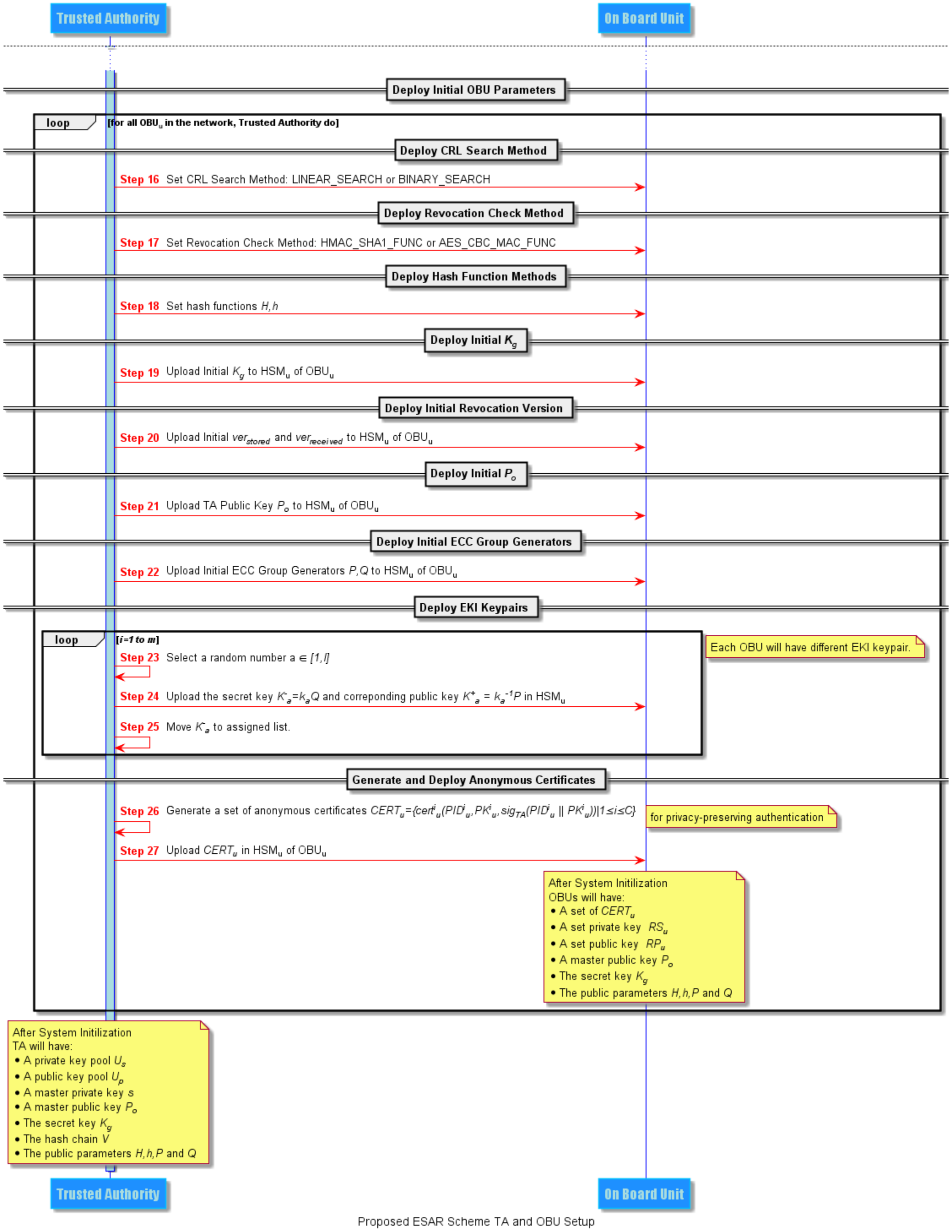

Trusted Authority
On Board Unit
Deploy Initial OBU Parameters
loop
[for all OBU$_u$ in the network, Trusted Authority do]
Deploy CRL Search Method
Step 16 Set CRL Search Method: LINEAR_SEARCH or BINARY_SEARCH
Deploy Revocation Check Method
Step 17 Set Revocation Check Method: HMAC_SHA1_FUNC or AES_CBC_MAC_FUNC
Deploy Hash Function Methods
Step 18 Set hash functions H,h
Deploy Initial K_g
Step 19 Upload Initial K_g to HSM$_u$ of OBU$_u$
Deploy Initial Revocation Version
Step 20 Upload Initial ver_{stored} and $ver_{received}$ to HSM$_u$ of OBU$_u$
Deploy Initial P_o
Step 21 Upload TA Public Key P_o to HSM$_u$ of OBU$_u$
Deploy Initial ECC Group Generators
Step 22 Upload Initial ECC Group Generators P,Q to HSM$_u$ of OBU$_u$
Deploy EKI Keypairs
loop
[$i=1$ to m]
Each OBU will have different EKI keypair.
Step 23 Select a random number a $\in [1,l]$
Step 24 Upload the secret key $K^-_a=k_aQ$ and correponding public key $K^+_a = k_a^{-1}P$ in HSM$_u$
Step 25 Move K^-_a to assigned list.
Generate and Deploy Anonymous Certificates
Step 26 Generate a set of anonymous certificates $CERT_u=\{cert^i_u(PID^i_u,PK^i_u,sig_{TA}(PID^i_u \| PK^i_u))|1\leq i\leq C\}$
for privacy-preserving authentication
Step 27 Upload $CERT_u$ in HSM$_u$ of OBU$_u$
After System Initilization
OBUs will have:
• A set of $CERT_u$
• A set private key RS_u
• A set public key RP_u
• A master public key P_o
• The secret key K_g
• The public parameters H,h,P and Q
After System Initilization
TA will have:
• A private key pool U_s
• A public key pool U_p
• A master private key s
• A master public key P_o
• The secret key K_g
• The hash chain V
• The public parameters H,h,P and Q
Trusted Authority
On Board Unit
Proposed ESAR Scheme TA and OBU Setup

Figure 8.4: Proposed ESAR Scheme TA and OBU Setup Flow-2

8.9.2 Proposed ESAR Scheme V2V Authentication

The authentication phase consists of signing and verification of telemetry messages between vehicles. According to WAVE [12], [13] standards each OBU broadcast own telemetry information to other OBUs at intervals of 300 ms. Sender OBU_S. Authentication process starts with self-check. If an OBU_shas missing revocation then stops, sending telemetry messages to other OBUs. This control process by $ver_{received}$ and ver_{stored} comparison. If $ver_{received}$ is initial value and not set or $ver_{received}$ equal to ver_{stored}, then there is no missing revocation else OBU detect missing revocation and stop sending. This state is continued until it complete missing revocations. If there is no missing revocation, OBU_s first, calculate revocation check value mentioned in EMAP REV_{check}. According to system setup REV_{check} can calculated by using $F_{mac.HMAC}(x,y)$ or $F_{mac.AES}(x,y)$ functions. In simulation but we choose $F_{mac.HMAC}(x,y)$ as follows:

$$REV_{check} = F_{mac.HMAC}(\left[K_g.x\right]^{128}, PID_s \parallel T_{stamp}) \tag{8.32}$$

Calculated revocation check value (REV_{check}), 32-bytes telemetry message ($\mathcal{M}$), certificate ($cert_s$), message signature (sig_s) and 1-byte message identification tag (TEL_MSG) concatenated as follow with the current timestamp (T_{stamp}):

$$cert_s\left(PID_s, PK_s^i, sig_{TA}\left(PID_s \parallel PK_s^i\right)\right) = PID_s \parallel PK_s \parallel sig_{TA}\left(PID_s \parallel PK_s^i\right) \tag{8.33}$$

$$\begin{gathered} < TEL\_MSG \parallel \mathcal{M} \parallel T_{stamp} \parallel cert_s\left(PID_s, PK_s^i, sig_{TA}\left(PID_s \parallel PK_s^i\right)\right) \\ \parallel sig_s\left(TEL\_MSG \parallel \mathcal{M} \parallel T_{stamp}\right) \parallel REV_{check} > \end{gathered} \tag{8.34}$$

Message generation is comparable to EMAP scheme, but in our scheme, we defined the usage of K_g key and added message identifier (TEL_MSG). Please note that secp256k signature and public key lengths ($len(PK_s)$) are 64-bytes.

Receiver OBU_r first checks message type and then take the authentication process and then check missing revocations. If OBU_r has missing revocation, then neglect all received messages until complete all missing revocations. If OBU_r has no missing revocation then first validate REV_{check} with own K_g as follows:

$$REV_{check} = F_{mac.HMAC}\left(\left[K_g.x\right]^{128}, PID_s \parallel T_{stamp}\right) \tag{8.35}$$

If not valid then reject message else verifies TA certificate signature by processing, thus:

$$\begin{aligned} e\left(sig_{TA}\left(PID_s \ \parallel PK_s^i\right), P\right) &= e\left(sH\left(PID_s \ \parallel PK_s^i\right), P\right) \\ &= e\left(H\left(PID_s \ \parallel PK_s^i\right), sP\right) = e\left(H\left(PID_s^i \ \parallel PK_s^i\right), P_o\right) \end{aligned} \tag{8.36}$$

If not valid then drop the message. If the TA certificate is valid, then use certificate OBU public key and validate the message signature as follows:

$$\begin{aligned} & e\left(sig_s(msg = TEL\_MSG \parallel \mathcal{M} \parallel \ T_{stamp}), P\right) \\ &\qquad = e\left(SK_s^i H\left(msg = TEL\_MSG \parallel \mathcal{M} \parallel \ T_{stamp}\right), P\right) \\ &\qquad = e\left(H\left(msg = TEL\_MSG \parallel \mathcal{M} \parallel \ T_{stamp}\right), SK_s^i P\right) \\ &\qquad = e\left(H\left(msg = TEL\_MSG \parallel \mathcal{M} \parallel \ T_{stamp}\right), PK_s^i\right) \end{aligned} \tag{8.37}$$

Finally, if the signature is valid then process the message ($\mathcal{M}$).

All authentication flow in Figure 8.5.

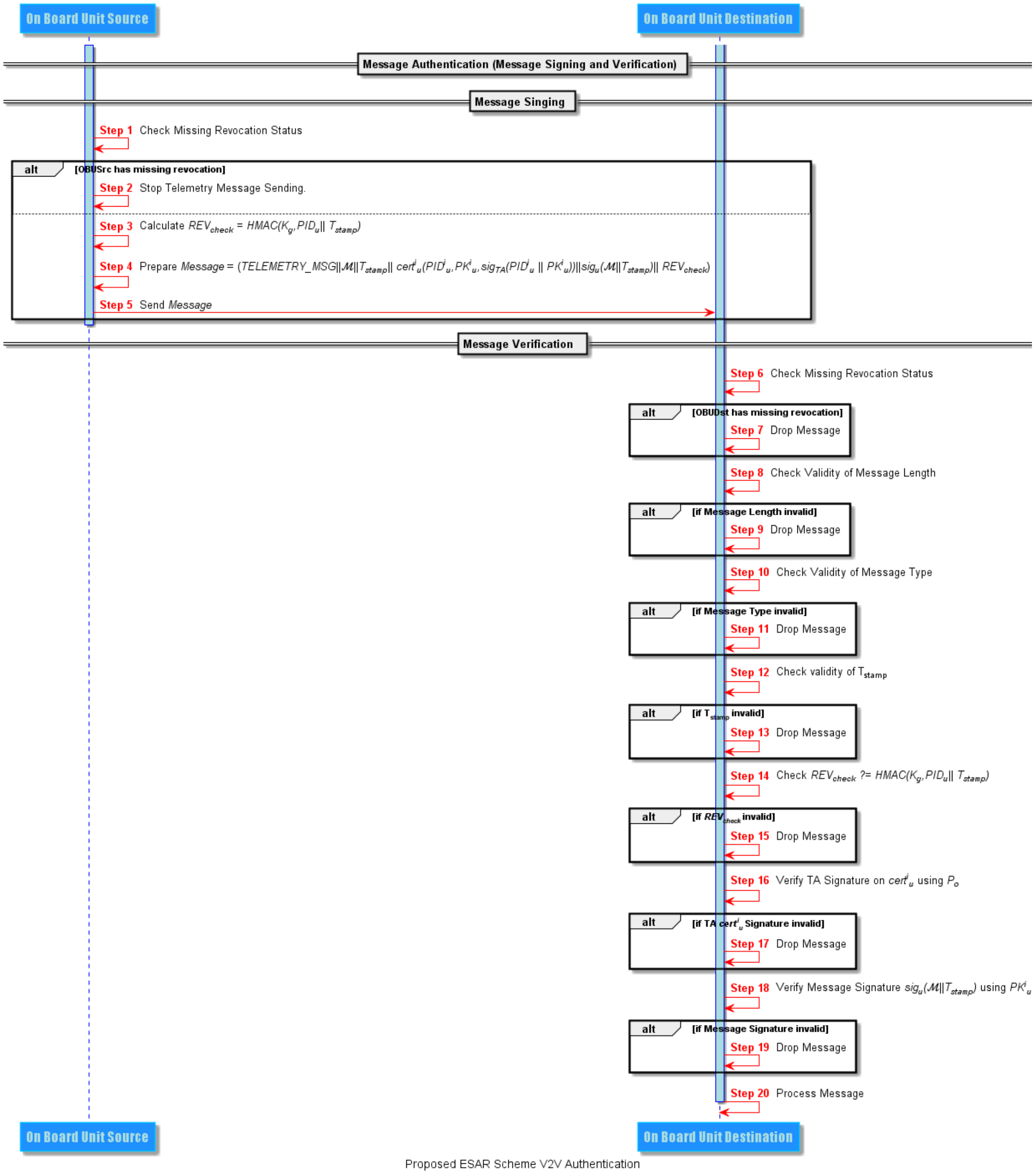

On Board Unit Source
On Board Unit Destination
Message Authentication (Message Signing and Verification)
Message Singing
Step 1 Check Missing Revocation Status
alt [OBUSrc has missing revocation]
Step 2 Stop Telemetry Message Sending.
Step 3 Calculate $REV_{check} = HMAC(K_g, PID_u \| T_{stamp})$
Step 4 Prepare $Message = (TELEMETRY\_MSG \| \mathcal{M} \| T_{stamp} \| cert^i_u(PID^i_u, PK^i_u, sig_{TA}(PID^i_u \| PK^i_u)) \| sig_u(\mathcal{M} \| T_{stamp}) \| REV_{check})$
Step 5 Send $Message$
Message Verification
Step 6 Check Missing Revocation Status
alt [OBUDst has missing revocation]
Step 7 Drop Message
Step 8 Check Validity of Message Length
alt [if Message Length invalid]
Step 9 Drop Message
Step 10 Check Validity of Message Type
alt [if Message Type invalid]
Step 11 Drop Message
Step 12 Check validity of T_{stamp}
alt [if T_{stamp} invalid]
Step 13 Drop Message
Step 14 Check REV_{check} ?= $HMAC(K_g, PID_u \| T_{stamp})$
alt [if REV_{check} invalid]
Step 15 Drop Message
Step 16 Verify TA Signature on $cert^i_u$ using P_o
alt [if TA $cert^i_u$ Signature invalid]
Step 17 Drop Message
Step 18 Verify Message Signature $sig_u(\mathcal{M} \| T_{stamp})$ using PK^i_u
alt [if Message Signature invalid]
Step 19 Drop Message
Step 20 Process Message
On Board Unit Source
On Board Unit Destination
Proposed ESAR Scheme V2V Authentication

Figure 8.5: Proposed ESAR Scheme OBU to OBU Authentication Flow

8.9.3 Proposed ESAR Scheme OBU Revocation

Revocation process similarly with EMAP, started by TA when any OBU_u revoked. TA select a valid EKI key pair M from its database and select a random $t \in \mathbb{Z}_q^*$ and calculate the intermediate key with the similar operation as follows:

$$rnd = F_{RAND}(.) \tag{8.38}$$

$$K_{im} = rndK_M^+ = \frac{rnd}{k_M}P \in \mathbb{G}_1 \tag{8.39}$$

Moreover, new revocation key $\widetilde{K}_g$as follows:

$$\widetilde{K}_g = e(K_M^-, K_{im}) = e\left(k_M Q, \frac{t}{k_M}P\right) = e(Q,P)^{k_M \cdot \frac{t}{k_M}} = e(Q,P)^t \in \mathbb{G}_2, t \in \mathbb{Z}_q^* \tag{8.40}$$

$$\widetilde{K}_g = \mathrm{K_{im}} K_M^- \tag{8.41}$$

TA get current version (ver) hash chain value $v_{j-ver}$from $V = \{v_i \,|0 \leq i \leq j\}$, and AES-CBC encrypt with $\widetilde{K}_g$ as follows:

$$enc_{\widetilde{K}_g}(v_{j-ver}) = F_{enc.AES}([\widetilde{K}_g.x]^{128}, v_{j-ver}) \tag{8.42}$$

v_{j-ver} is 20-bytes long for SHA-1, and during encryption, its padded and encrypted buffer is 32-bytes long. All update required EMAP key ID list is stored on $ID_{rev_{key}}$ as follows:

$$ID_{rev_{key}} = < M_1 \parallel M_2 \parallel \cdots \parallel M_n > \tag{8.43}$$

TA concatenate these parameters and build a key update message as follows:

$$K_{msg} = \left(ver \parallel M \parallel \mathrm{len}(ID_{rev_{key}}) \parallel ID_{rev_{key}} \parallel K_{im} \parallel enc_{\widetilde{K}_g}(v_{j-ver})\right) \tag{8.44}$$

Additionally, TA provides a PKI scheme CRLs as follows:

$$CRL = < PID_1 \parallel PID_2 \parallel \cdots \parallel PID_n > \tag{8.45}$$

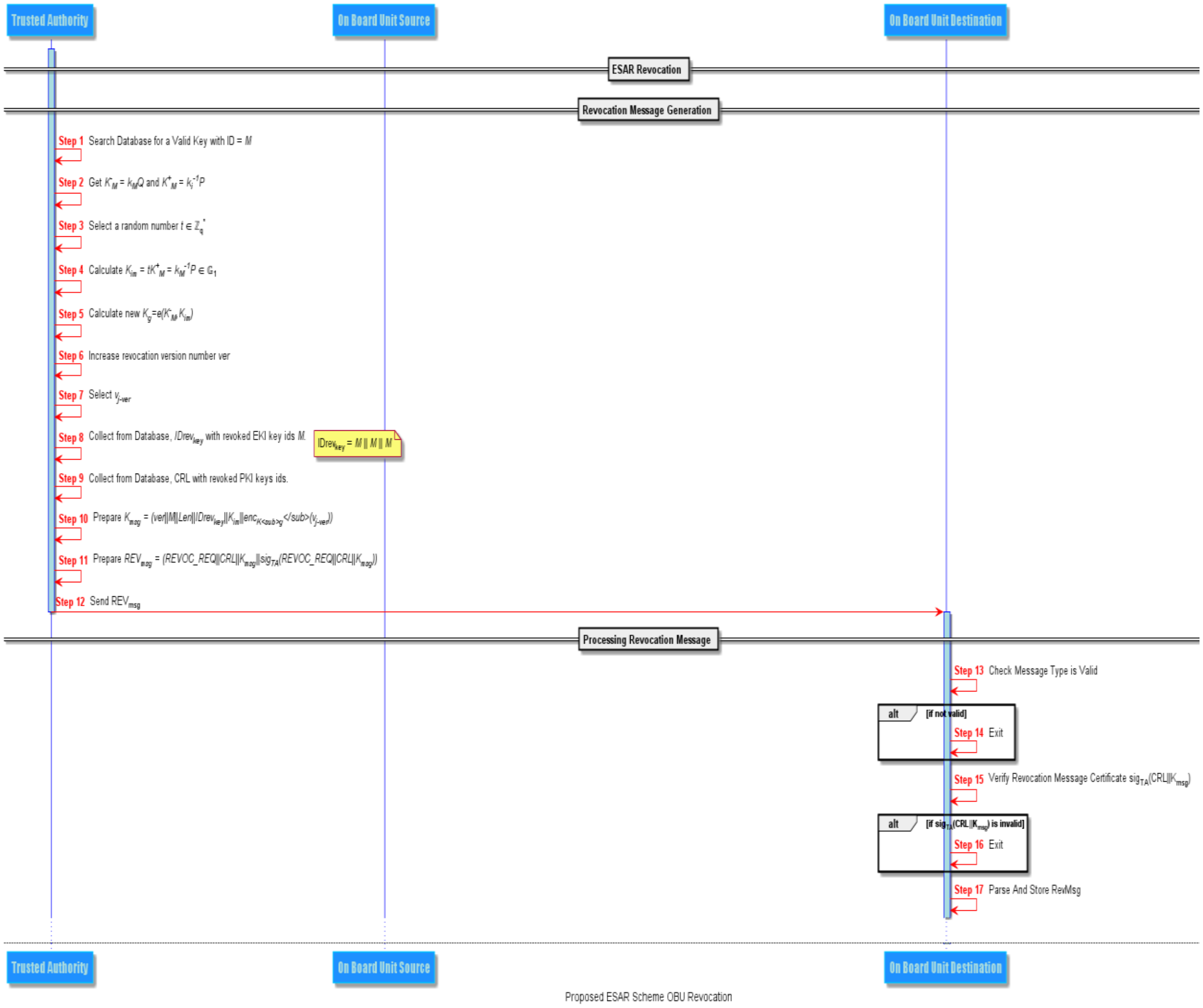
Also, prepares signed revocation message as follows:

$$REV_{msg} = \big(msg = REVOC\_REQ \ \| \ len(CRL) \ \| \ CRL \ \| \ K_{msg} \ \| \ sig_{TA}(msg = REVOC\_REQ \ \| \ len(CRL) \ \| \ CRL \ \| \ K_{msg})\big) \tag{8.46}$$

So broadcast to the network after this operation the TA synchronize own EKI keys with the following operations:

$$\widetilde{K}_i^- = v_{j-ver} K_i^- \ and \ \widetilde{K}_i^+ = (\frac{1}{v_{j-ver}}) K_i^+ . \tag{8.47}$$

Revocation flow until this step in Figure 8.6.



Trusted Authority
On Board Unit Source
On Board Unit Destination
ESAR Revocation
Revocation Message Generation
Step 1 Search Database for a Valid Key with ID = M
Step 2 Get $K^-_M = k_M Q$ and $K^+_M = k_i^{-1} P$
Step 3 Select a random number $t \in \mathbb{Z}_q^*$
Step 4 Calculate $K_{im} = tK^+_M = k_M^{-1} P \in \mathbb{G}_1$
Step 5 Calculate new $K_g = e(K^-_M, K_{im})$
Step 6 Increase revocation version number ver
Step 7 Select v_{j-ver}
Step 8 Collect from Database, $IDrev_{key}$ with revoked EKI key ids M.
$IDrev_{key} = M \| M \| M$
Step 9 Collect from Database, CRL with revoked PKI keys ids.
Step 10 Prepare $K_{msg} = (ver||M||Len||IDrev_{key}||K_{im}||enc_{K_{g}$}($v_{j-ver}$))
Step 11 Prepare $REV_{msg} = (REVOC\_REQ||CRL||K_{msg}||sig_{TA}(REVOC\_REQ||CRL||K_{msg}))$
Step 12 Send REV_{msg}
Processing Revocation Message
Step 13 Check Message Type is Valid
alt [if not valid]
Step 14 Exit
Step 15 Verify Revocation Message Certificate $sig_{TA}(CRL||K_{msg})$
alt [if $sig_{TA}(CRL||K_{msg})$ is invalid]
Step 16 Exit
Step 17 Parse And Store RevMsg
Trusted Authority
On Board Unit Source
On Board Unit Destination
Proposed ESAR Scheme OBU Revocation

Figure 8.6: Proposed ESAR Scheme OBU Revocation Flow-1

When OBU receives any message first checks message tags and redirect message to related flow. Additionally, when revocation message detected then verifies TA signature as follows:

$$sig_{TA}(msg = REVOC\_REQ \parallel len(CRL) \parallel CRL \parallel K_{msg}) \tag{8.48}$$

by processing, thus:

$$e\big(sig_{TA}\big(msg = REVOC\_REQ \parallel len(CRL) \parallel CRL \parallel K_{msg}\big), P\big) = e\big(H\big(msg = REVOC\_REQ \parallel len(CRL) \parallel CRL \parallel K_{msg}\big), sP\big) = e\big(H\big(msg = REVOC\_REQ \parallel len(CRL) \parallel CRL \parallel K_{msg}\big), P_o\big). \tag{8.49}$$

Next step is decoding the message and update CRL and parse the following message, thus:

$$K_{msg} = \left(ver \parallel M \parallel \mathrm{len}(ID_{rev_{key}}) \parallel ID_{rev_{key}} \parallel K_{im} \parallel enc_{\widetilde{K}_g}(v_{j-ver})\right) \tag{8.50}$$

To start the $\widetilde{K}_g$ and v_{j-ver} retrieve process. If OBU has key pair M then easily calculates $\widetilde{K}_g$ and decrypt $enc_{\widetilde{K}_g}(v_{j-ver})$ to get plain v_{j-ver} as follows:

$$\widetilde{K}_g = \mathrm{K_{im}} K_M^- \tag{8.51}$$

$$v_{j-ver} = F_{dec.AES}(\big[\widetilde{K}_g.x\big]^{128}, enc_{\widetilde{K}_g}(v_{j-ver})\,) \tag{8.52}$$

If an OBU_s do not own key which defined with M in the message to recover $\widetilde{K}_g$, and also have missing revocation, then, send a signed request to get this key from neighboring OBU_r's. We send the following request to receive both encrypted $\widetilde{K}_g$ and $ID_{rev_{key|missed}}$ in a single signed message with time, signature, up-to-date and other security validation,thus:

$$< msg = KG\_KEY\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp} \parallel PID_s^i \parallel PK_s^i \parallel sig_{TA}(PID_s^i \parallel PK_s^i) \parallel sig_s(msg = KG\_KEY\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp}) > \tag{8.53}$$

If an OBU_s request encrypted $\widetilde{K}_g$ from other OBU_r then, before all validations, OBU_r check own missing revocation status and if there is a missing revocation then it detect that own $\widetilde{K}_g$ is not fresh and stops processing message else OBU_r's search PID_s^i in CRLs with the linear or binary search, if exist then reject request else continue processing. OBU_r's detect sender missing revocation status after message signature, OBU_s certificate and message time validation by checking, thus:

$$ver_{received} \neq ver_{stored} + 1 \tag{8.54}$$

If OBU_s is valid but do not have missing revocation then only send ECIES encrypted $\widetilde{K}_g$ by using OBU_s's PK_s^i, in the following response:

$$\begin{aligned} < msg = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \parallel \ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \\ \parallel 0 \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \parallel sig_{TA}(PID_r^i \parallel \ PK_r^i) \parallel sig_r(msg \\ = KG\_KEY\_RSP \parallel len\left(F_{enc.ECIES}\left(\widetilde{K}_g, PK_s^i\right)\right) \\ \parallel \ F_{enc.ECIES}(\widetilde{K}_g, PK_s^i) \parallel 0 \parallel T_{stamp}) > \end{aligned} \tag{8.55}$$

In above, zero denotes the represent empty $ID_{rev_{key|missed}}$ length. If missing revocations detected for OBU_s then following buffer prepared for $ID_{rev_{key|missed}}$, thus:

$$\begin{gathered} ID_{rev_{key|missed}} = < ver_i \ \parallel len_{\ i} \parallel M_{\ i_1} \parallel M_{\ i_2} \parallel \cdots \parallel M_{\ i_n} \parallel ver_j \ \parallel \\ len_{\ j} \parallel M_{\ j_1} \parallel M_{\ j_2} \parallel \cdots \parallel M_{\ j_n} \parallel \cdots \parallel ver_m \ \parallel len_{\ m} \parallel M_{\ m_1} \parallel M_{\ m_2} \parallel \cdots \parallel \\ M_{\ m_n} \ > | ver_{stored} \leq i < j < m \leq ver_{received} - 1 \end{gathered} \tag{8.56}$$

This buffer placed in response buffer as follows:

$$
\begin{aligned}
< msg = KG\_KEY\_RSP \parallel len\Big(F_{enc.ECIES}\big(\widetilde{K}_g, PK_s^i\big)\Big) \\
\parallel\ F_{enc.ECIES}\big(\widetilde{K}_g, PK_s^i\big) \parallel len\big(ID_{rev_{key|missed}}\big) \\
\parallel ID_{rev_{key|missed}} \parallel T_{stamp} \parallel PID_r^i \parallel PK_r^i \parallel sig_{TA}\big(PID_r^i \\
\parallel\ PK_r^i\big) \parallel sig_r\Big(len\Big(F_{enc.ECIES}\big(\widetilde{K}_g, PK_s^i\big)\Big) \\
\parallel\ F_{enc.ECIES}\big(\widetilde{K}_g, PK_s^i\big) \parallel len\big(ID_{rev_{key|missed}}\big) \\
\parallel ID_{rev_{key|missed}} \parallel T_{stamp}\Big) >
\end{aligned}
\qquad (8.57)
$$

Revocation flow until this step in Figure 8.7.

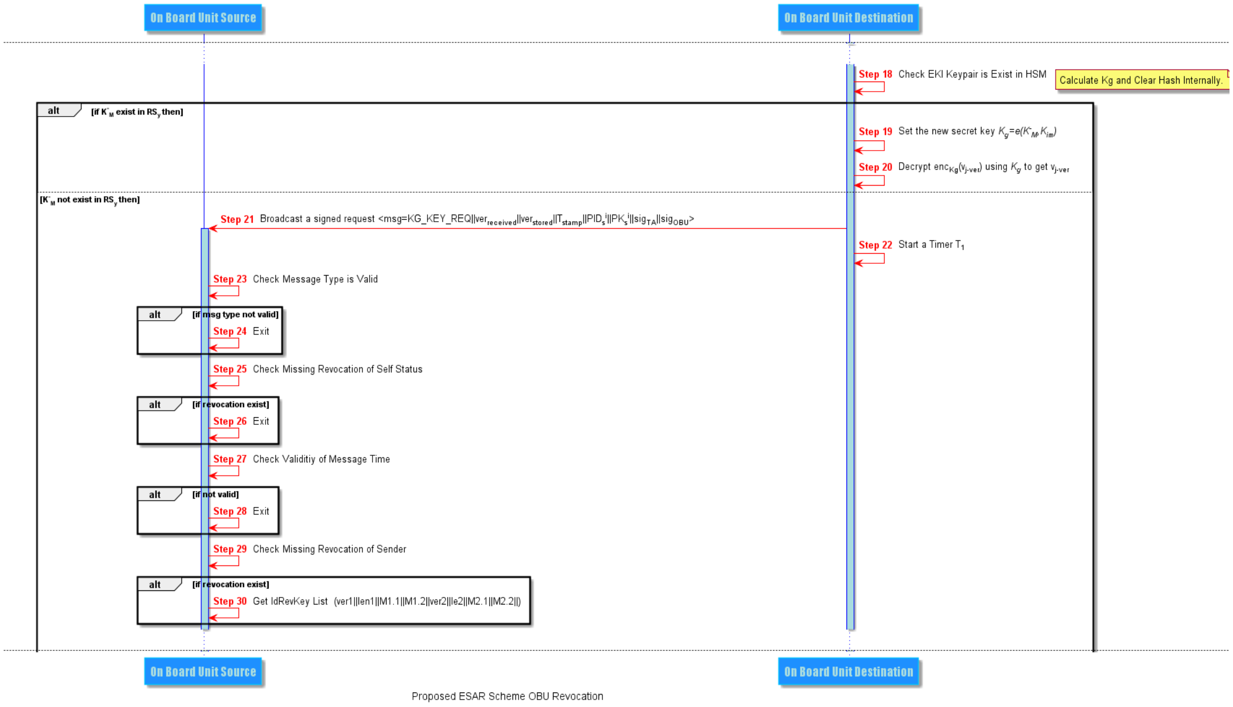

On Board Unit Source
On Board Unit Destination
Step 18 Check EKI Keypair is Exist in HSM
Calculate Kg and Clear Hash Internally.
alt
[if K*M exist in RSy then]
Step 19 Set the new secret key Kg=e(K*M,Kim)
Step 20 Decrypt encKg(vj-ver) using Kg to get vj-ver
[K*M not exist in RSy then]
Step 21 Broadcast a signed request <msg=KG_KEY_REQ||verreceived||verstored||Tstamp||PIDsi||PKsi||sigTA||sigOBU>
Step 22 Start a Timer T1
Step 23 Check Message Type is Valid
alt
[if msg type not valid]
Step 24 Exit
Step 25 Check Missing Revocation of Self Status
alt
[if revocation exist]
Step 26 Exit
Step 27 Check Validitiy of Message Time
alt
[if not valid]
Step 28 Exit
Step 29 Check Missing Revocation of Sender
alt
[if revocation exist]
Step 30 Get IdRevKey List (ver1||len1||M1.1||M1.2||ver2||le2||M2.1||M2.2||)
On Board Unit Source
On Board Unit Destination
Proposed ESAR Scheme OBU Revocation

Figure 8.7: Proposed ESAR Scheme OBU Revocation Flow-2

Receiver OBU_r verifies certificate and signature then decrypt, encrypted $\widetilde{K}_g$ as follows:

$$\widetilde{K}_g = F_{dec.ECIES}\big(\widetilde{K}_g, SK_s^i\big) \qquad (8.58)$$

Moreover, decrypt encrypted hash chain value, thus:

$$v_{j-ver} = F_{dec.AES}\left(\left[\widetilde{K}_g.x\right]^{128}, enc_{\widetilde{K}_g}(v_{j-ver})\right) \tag{8.59}$$

However, if OBU_s own key, which defined with M in the message to recover $\widetilde{K}_g$ but has missing revocations then sends the following message to only get $ID_{rev_{key|missed}}$:

$$\begin{aligned} < msg = REV\_ID\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp} \parallel PID_s^i \\ \parallel PK_s^i \parallel sig_{TA}(PID_s^i \parallel \; PK_s^i) \parallel sig_s(msg \\ = REV\_ID\_REQ \parallel ver_{received} \parallel ver_{stored} \parallel T_{stamp}) > \end{aligned} \tag{8.60}$$

Revocation flow until this step in Figure 8.8.

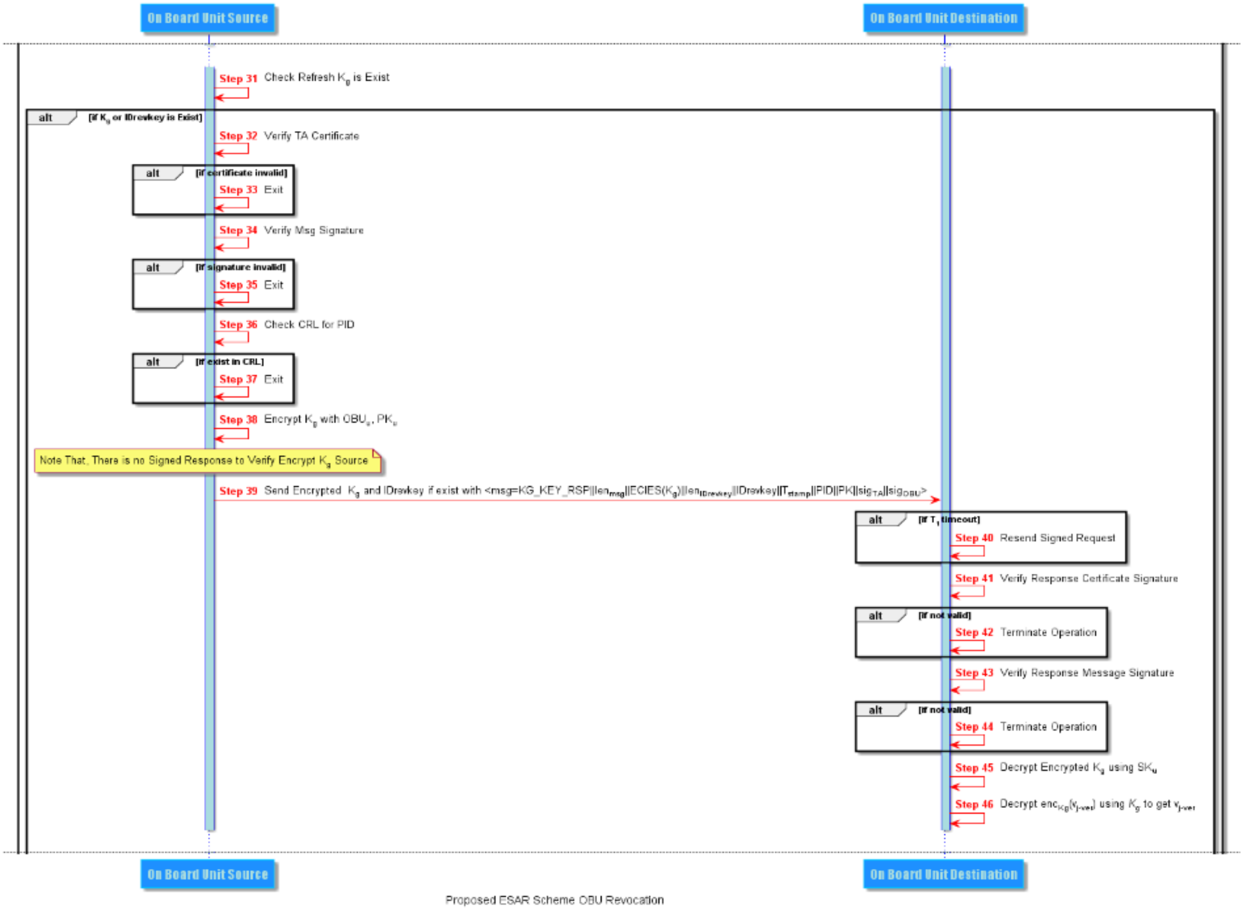

On Board Unit Source
On Board Unit Destination
Step 31 Check Refresh K$_g$ is Exist
alt [if K$_g$ or IDrevkey is Exist]
Step 32 Verify TA Certificate
alt [if certificate invalid]
Step 33 Exit
Step 34 Verify Msg Signature
alt [if signature invalid]
Step 35 Exit
Step 36 Check CRL for PID
alt [if exist in CRL]
Step 37 Exit
Step 38 Encrypt K$_g$ with OBU$_s$, PK$_s$
Note That, There is no Signed Response to Verify Encrypt K$_g$ Source
Step 39 Send Encrypted K$_g$ and IDrevkey if exist with <msg=KG_KEY_RSP||len$_{msg}$||ECIES(K$_g$)||len$_{IDrevkey}$||IDrevkey||T$_{stamp}$||PID||PK||sig$_{TA}$||sig$_{OBU}$>
alt [if T$_1$ timeout]
Step 40 Resend Signed Request
Step 41 Verify Response Certificate Signature
alt [if not valid]
Step 42 Terminate Operation
Step 43 Verify Response Message Signature
alt [if not valid]
Step 44 Terminate Operation
Step 45 Decrypt Encrypted K$_g$ using SK$_u$
Step 46 Decrypt enc$_{Kg}$(v$_{j-ver}$) using K$_g$ to get v$_{j-ver}$
On Board Unit Source
On Board Unit Destination
Proposed ESAR Scheme OBU Revocation

Figure 8.8: Proposed ESAR Scheme OBU Revocation Flow-3

Receiver OBU will process CRL, certificate and signature controls and then if everything valid and any key ID exist then will return following response:

$$\begin{aligned} < msg = REV\_ID\_RSP \parallel\ len(ID_{rev_{key|missed}}) \parallel ID_{rev_{key|missed}} \parallel T_{stamp} \\ \parallel PID_r^i \parallel PK_r^i \parallel sig_{TA}(PID_r^i \parallel\ PK_r^i) \parallel sig_r(len(ID_{rev_{key|missed}}) \\ \parallel ID_{rev_{key|missed}} \parallel T_{stamp}) > \end{aligned} \tag{8.61}$$

If KG_KEY_RSP or REV_ID_RSP message contains $ID_{rev_{key|missed}}$ then OBU_s will find $v_{j-ver_{missed}}$ from the calculated list and update keys as follows:

$$\widetilde{K}_i^- = v_{j-ver_{missed}} K_i^- \text{ and } \widetilde{K}_i^+ = (\frac{1}{v_{j-ver_{missed}}}) K_i^+. \tag{8.62}$$

Finally, update current revocation keys, thus:

$$\widetilde{K}_i^- = v_{j-ver} K_i^- \text{ and } \widetilde{K}_i^+ = (\frac{1}{v_{j-ver}}) K_i^+ \tag{8.63}$$

After this operation, revocation executed for each OBU and each OBU should store received revocation versions ($ver_{received}, ver_{stored}$) and associated updated key pair ID list ($ID_{rev_{key}}$) and CRLs. Finally, revocation flow until this step in Figure 8.9.

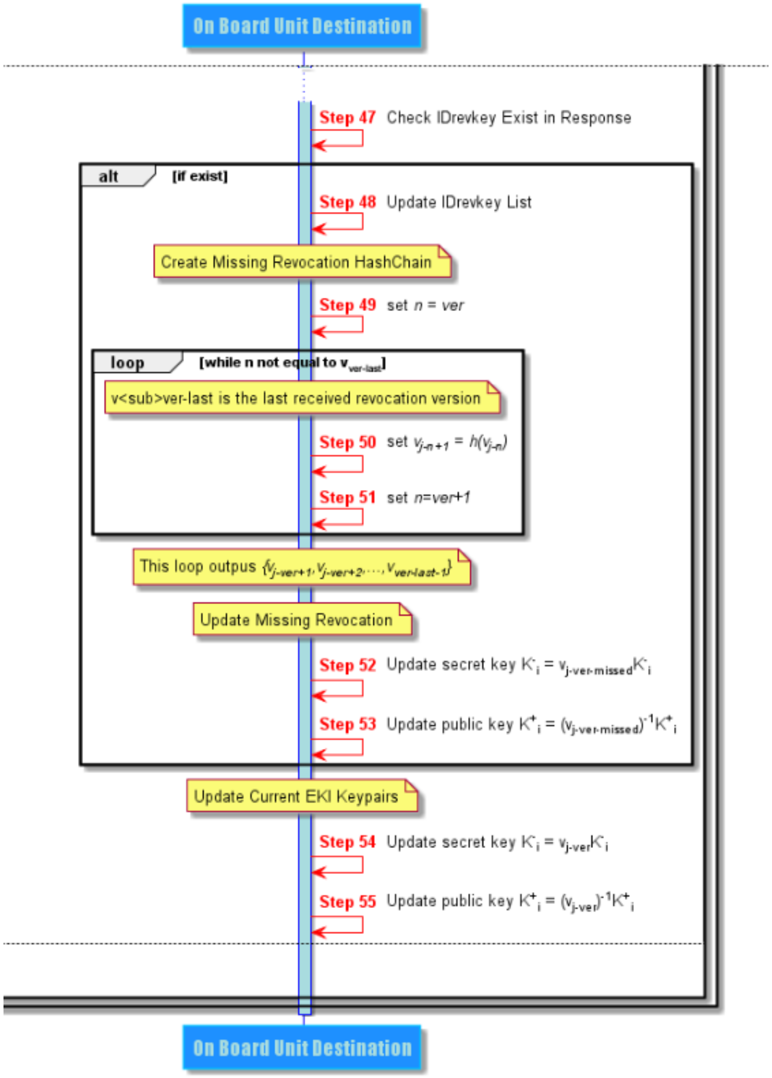

On Board Unit Destination
Step 47 Check IDrevkey Exist in Response
alt [if exist]
Step 48 Update IDrevkey List
Create Missing Revocation HashChain
Step 49 set $n = ver$
loop [while n not equal to $v_{ver-last}$]
v<sub>ver-last is the last received revocation version
Step 50 set $v_{j-n+1} = h(v_{j-n})$
Step 51 set $n=ver+1$
This loop outpus $\{v_{j-ver+1}, v_{j-ver+2}, \ldots, v_{ver-last-1}\}$
Update Missing Revocation
Step 52 Update secret key $K^-_i = v_{j-ver-missed}K^-_i$
Step 53 Update public key $K^+_i = (v_{j-ver-missed})^{-1}K^+_i$
Update Current EKI Keypairs
Step 54 Update secret key $K^-_i = v_{j-ver}K^-_i$
Step 55 Update public key $K^+_i = (v_{j-ver})^{-1}K^+_i$
On Board Unit Destination

Figure 8.9: Proposed ESAR Scheme OBU Revocation Flow-4

9. PROTOCOL AND SECURITY ANALYSIS OF M2M SCHEME

For our development and [5], we conducted a MATLAB simulation and examined valid and invalid case activities. The suggested system ensures safety against replay assaults, altering range assaults, similar device assaults, compositional assaults, redirection assaults, MITM assaults, replacement assaults, DoS assaults, forging assaults, colluding assaults, flooding assaults, fake resposne assaults, sybil assaults, message changes, wormhole assaults, black hole assaults, attribute trace assaults, eavesdropping an attack In order to safely communicate secrecy and confidentiality keys for necessary parts we have used a hybrid cryptographic system (ECDH, AES, SHA 1, and H MAC). The home gateway and client device identity information has been converted into transient anonymity identities using the safely shared MAC key. The suggested system also carries out important revocations for customer password and house gateway.

9.1 REPLAY ATTACK

Scheme [5] is categorized in [1] and [5] asserts security for the assault as partly supporting. They are supposed to have the login operation terminated according to the random number input x_1 verification if an intruder intercepts a valid login message and replays that message containing $< uid, g, h_1, s >$ after logout detection. However, no replay assault security is available for the home gateway unit. The attacker can intercept a call to a home station and send his own secret data (hid) to the M2M server. An attacker that is registered can send fake data across this channel to the M2M server. To safeguard device registration against this kind of assault, we introduced the house gateway key injection stage.

In addition, status flags identify false applications for replays. The operation is aborted on the M2M server hand by invalid status discovery. Another problem concerns [5] take the responsibility to protect providers that record and alter their password and to perform these activities on the user's devices. If a safe channel is available, however, then authentication is no longer needed. Our system implies that no safe channel is available to safeguard every stage for replay attacks. For these purposes, we have completely endorsed our protocol.

9.2 CHANGING DISTANCE ATTACK

Study [1] lists the scheme [5] that this assault has not been endorsed. For each stage of the operation, there is no time measurement [5]. An intruder may interrupt contact and send a query to the receiving entity after the examination. This assault has not been discovered, the evolving remote approach affects the processing time of the application, and the scheme suggested has a minimum and maximum time for extraordinary operational identification. Additional timeout detection for each implementation is provided for this security. We have identified our system as completely endorsed for these reasons.

9.3 SAME-TYPE-DEVICE ATTACK

Survey [1] categorizes the scheme [5] as not endorsed. Scheme [5] home portal cannot detect an intruder ,and the initial hid can accumulate from the public channel. To acquire the encryption key, the attacker uses hid to send multiple register messages to the M2M server. If there is more than one request sending the registration, sniffing, decoding of the home gateway or M2M server traffic will reveal the authentication elements. Furthermore, an attacker can modify the transmitted messages and transmit fake information to the M2M server to prevent facilities. Our suggested scheme recognizes only messages backed by MAC. After insertion of the client key a home gateway device receives a join signal. In addition, our design identifies each device with an ID in a unique way, and this ID protects the device against scratching on the public system with H-MAC diversification. These methods of safety protect against assaults of the same type of device. If an invader virtually registers the equipment on the M2M server, it should have a client integrity key that is safely inserted with the private network through the home portal as indicated in the enter stage segment of the home gateway. We designated our model as fully verified for these reasons.

9.4 COMPOSITION ATTACK

[1] mark schema [5] as a unsupported design for this assault. Attackers create structure assaults that collect multiple request attributes. This attribute's structure demonstrates valuable properties. The enrollment system [5] and changes to the password are not as safe as earlier stated. The

customer ID can also be scratched and used to gather data about bogus structures that is sensitive to the user. With the encryption key and MAC our system safeguards all critical items. These encryption keys are carried by our system under ECDH keys. This system prevents from the disclosure of valuable properties. We have marked our design as completely endorsed for these purposes.

9.5 REDIRECTION ATTACK

[1] labels the [5] system that this assault has not been endorsed. An attacker launches a redirection attack by simulating the M2M server to collect data from the network and valuable resources of the client and home gateway. Scheme [5] targets unlisted injection and protection of M2M server data and redirection applications to the attacker's database from a fraudulent base station.

We also remember that redirected messages are encrypted in our model to protect redirection assaults with time measures. In order to get out of messages, MAC prevents helpful data. The intruder is unable to alter messages. In addition, as stated above, we have a safe key injection for the safe registration of home gateway equipment. Our model never transmits valuable properties through a public network and secures MAC messages. That is why the hacker is unable to understand how to grab or alter messages with the authentication secure keys. We therefore requested full support for our system.

9.6 MITM ATTACK

[5] in this assault is marked as partially endorsed in [1]. However, no MAC security exists in Scheme [5] for transferred messages. Messages can be changed, and transactions stopped. In addition, terminated operations oblige the user to retry and so gather more information on the assaulted client in order to retrieve information. Another problem is the hacker can cause false information to be transmitted when modified. Our MAC control system offers security for message change attacks. With a confidentiality encryption key and MAC key for privacy, we protect valuable property. The sensitive property will never be clearly communicated over a

public network, which will resist a MITM assault. We considered our plan to be completely endorsed for this purpose.

9.7 SUBSTITUTION ATTACK

[1] defines [5] that this assault was not endorsed. The replacement attack is a specific form of MITM assault which substitutes initial algorithms with derivative implementation to distribute transmitting data. The system [5] does not protect password modifications for users, client registrations and access devices. An attacker can quickly retrieve a sensitive message user ID and password.

The design proposed never transfers actual user ID to an open systems. Furthermore, the ECDH important combination between the M2M server and user equipment and safe key insertion between the home gateway and user device enable our model to implement privacy safety. In this process, the key for defending the identification on the public system in the event of an attack is secure sharing between the home gateway device, the user device, and theM2M server. Consequently, a replacement assault is not appropriate for this concept and we instructed our scheme to be endorsed in its entirety.

9.8 DOS ATTACK

[5] is stated as not suggested for this assault in [1]. A DoS intruder sends large amounts of data to use more system assets to block servers. Scheme[5] has no safety or detection system for this type of assault. In our scheme, on the other side, DoS assaults are evaded with the resolve of operating time and state machine management. We also protect MAC messages and encryption keys. Attackers can not send legitimate messages on the M2M server to change state machine parameters. If an intruder receives messages which operating time is less than predicted within the minimum transaction handling era, then this is regarded as a suspicious process and the requesting entity cancels associated applications. In addition, if an intruder receives successive messages with incorrect MACs, there is a trial count handling; then the client or device is reported as BLOCKED and calls are denied. This blocking method prevents the use of system

assets, such as server cryptographic procedures. We have designated our system as completely endorsed for this assault.

9.9 FORGING ATTACK

[1] Labeling [5] as not endorsed. This assault was not endorsed. Schema [5], changes to client passwords, register of users and home gateway facilities are not safe. A vulnerable password and the user ID can be recovered from the data immediately. However, modifications to forging attacks are disappointed with MAC protection state machine manager, and transaction time controls in the suggested system. The attacker can not reveal secret keys to link the scheme and our standard, brute-force-resistant algorithms. For this assault we have identified our protocol as completely authorized.

9.10 COLLUDING ATTACK

Survey [1] shows scheme [5] that, this assault has not been maintained. The colluding assault is a guessing assault. For current demands, the intruder can use heritage details Scheme [5] utilize salt (s) to obtain a login key for a home gateway device. The method of changing user password refreshes the server's salt; but no revocation system exists and the home portal unit refreshes its own key after a change of password. Our request for scheme password change also has a key revocation process to refresh existing keys. Moreover, our application protects every message with MAC safety and never passes IDs over a public network. An intruder should acknowledge the MAC key for the colluding assault, and the scheme suggested is not suitable for this. For this purpose, we thought that our strategy for this assault was fully true.

9.11 FLOODING ATTACK

[5] is marked in [1] as being not endorsed. Attacks such as SYN FLOODING occur when the channel is configured A first request signal can be sniffed and data can be replayed to avoid services Scheme [5] has no count security attempt to avoid such an assault from being applied. With system time calculation, and state machine inspect techniques, our regime prevents assaults. Furthermore, our proposed scheme protects messages, with MAC confirmation. In order to

transmit the first application to the M2M server, the receiver should acknowledge the MAC key, so it is not feasible for the attacker. Our system fully resists this assault for this intent.

9.12 FALSE MESSAGE ATTACK

Scheme [5]is identified in[1]as partly endorsed. A malicious message assailant can use the scheme to enter fake messages. [5]has no verification of MAC or tries request count control. An attacker can not convey messages outside a MAC key in the suggested scheme. The timing control system to detect the attacker initiate bulk applications and MAC safety. In addition, denials never disclose precious attacker information. We have mentioned our scheme as suggested for this assault.

9.13 SYBIL ATTACK

[5] in [1], this assault is regarded as partly advised. Sybil assaults simulates the identification of the present system's customer phone or home gateway device, in an understandable format, the scheme [5] receives customer IDs and can be sniffed from an intruder. Sybil attacks in this suggested scheme are avoided by a transient ID, safety MAC, and confidential data encryption. For this assault, we categorized our scheme as completely preserved.

9.14 MESSAGE MODIFICATION ATTACK

[5] in [1] is categorized as not being endorsed. Scheme [5] does not protect messages with MAC and during transit, attackers may modify messages. The system suggested ensures the integrity of information with a MAC key. In order to change messages attackers should acknowledge the MAC key. We asserted that our layout was completely endorsed.

9.15 WORMHOLE ATTACK

[1] lists Scheme [5] for that assault as partly verified. For tailored inquiry or obstructing, Wormhole attackers tunnels messages from one location to another in the network. This assault shifts the time of transport and handling. Scheme [5] has no time assessment for call processes. However, this sort of assault can be recognized in our research, processing time determination.

Also, assailants should have MAC and cipher keys in the protected M2M network. The assault is completely endorsed by our scheme.

9.16 BLACKHOLE ATTACK

Scheme [5] has been reported in [1], this assault as having been partly confirmed. A blackhole assailant receives a fake response to spread messages to block the links between entities. Scheme [5] has no verification of the message, which could affect it. MAC verification is advocated for our suggested system calls and responses Without an integrity key, the attacker can not give a right answer to the request object The attacker can't change the link provider without these kinds of keys In addition, such an assault affects the timing of the application and the suggested scheme time assessment determine this assault. We find our approach to this assault to be entirely advisable.

9.17 ATTRIBUTE-TRACE ATTACK

[1] not suggested for such an assault marks the scheme [5]. An intruder will uncover information from all the gathered information and characteristics of research. We advocate the password modification and enrollment stage for this purpose. As discussed in Section III, sensitive entities like password and user ID may be scratched during transport. The suggested scheme encrypts important parameters, and utilizes an asymmetric key sharing for transiting the cipher key. Property tracing is therefore strengthened to disclose appropriate information. We have stated that our plan for this assault was completely advised.

9.18 EAVESDROPPING ATTACK

[1] assigns scheme [5] for this assault to be partly verified. The transmitting company monitors eavesdropping assaults and obtains critical data. This attack can easily influence the registration scheme [5] and change the password situations and reveal critical assets such as passwords and user ID rapidly. The suggested scheme blocks the ECDH-key-pair eavesdropping assault supplied for the MAC safety signal and transportation. In addition, it ciphers sensitive assets with

the encryption key and derives IDs for anonymity to a temporal ID on the government system. We have recognized our system as completely endorsed.

9.19 CHOSEN-PLAINTEXT ATTACK

[1] lists [5] that, this assault is not endorsed. The [5] key revocation of the home unit is not included. Sections I and III dealt with this scenario. The plaintext assault selected is not appropriate in our scheme. User revocation and key renewals processes are available. In addition, 12-bits AES encoding and ECDH keys are used in the databases that are not suitable for brute force assaults. Our technique for this assault is strongly recommended.

9.20 SPAM ATTACK

Survey [1] shows scheme [5] that, this assault was partially endorsed. In order to crack the communication provider and status flags during operations, the intruder can record the network and send inexistent information to the scheme or change the actual data. Messages may change, and false messages may be passed to the provider [5] to dismiss the communication. In order to prevent spam assaults the suggested system introduced message MAC assistance, time measurement handling, and entity state machine monitors. For these reasons, we have identified our layout as completely endorsed.

9.21 IDENTITY THEFT ATTACK

Examination [1] concept marks [5] as part of this attack's support. An attacker sniffs the ID of the item and replicates altered messages in [5]. Smartcard keeps user identifications, but the user equipment transmits the user ID in an understandable manner while the information is being transported. The intruder may sniff the user ID and replicate messages. The suggested design uses a transient ID and masks the identification with H-MAC. Actual IDs are never transmitted over public channels. We categorized our scheme as being highly suggested for this assault.

9.22 USER MANIPULATION ATTACK

[1] considered that the system [5] for this assault was in partly suggested. An enemy tries to look like a M2M server in this assault. The home gateway unit can then be transferred to the default M2M server and the message authentication key for the home gateway unit can then be retrieved from the messages. The intruder can sniff or change sensitive items, sent in the home gateway and M2M server messages. From a non-protected registration phase, the attacker may expose the registered device ID (hid) and user ID (uid). The enemy also sniffs transactions between the M2M server and the house portal to retrieve H_p for an important operation. Once this move has been completed the attacker can create an incorrect network for the attacked home portal to call the fraudulent M2M server and answer the session key from H_p to the home gateway. Legacy Scheme calculate H_{key} by using H_p with the home gateway devices. This approach cannot be distinguished from the opponent, and valuable data can be treated from the home gateway device, or requests can be sent to the home gateway to monitor and manage equipment as follows:

$$H_p = [h(uid, mid, s)]^m \tag{9.1}$$

$$H_{key} = [h(hid, H_p)]^t \tag{9.2}$$

The assault by users is not suitable in our scheme. Because all critical items are ciphered by H_p key and by keeping MAC as follows during transport as follow:

$$mac_{msg} = F_{H-MAC}(JOIN\_ACK \parallel H_p, key_{mac}) \tag{9.3}$$

$$enc_{msg} = F_{e-AES}(H_p \parallel mac_{msg}, key_{enc}) \tag{9.4}$$

$$< JOIN\_ACK, enc_{msg} > \tag{9.5}$$

Additional critical items are ciphered. Our plan endorsed the assault in its entirety.

9.23 ROUTING ATTACK

Survey [1] believes that the [5] plan for this assault was partly affirmed. First, the intruder utilizes the device modification approach to obtain the link key for the home gateway equipment and to avoid connecting client and home gateway. The intruder can obtain valuable data from the home portal during this phase. The method was identified in Scheme [5] was not correct. The suggested system, however, avoids routing assaults through MAC and can differentiate routing from encoded messages and activity time measures. To that end, our plan for this assault is fully approved.

9.24 LINKABILITY ATTACK

Survey [1] for the labeling of this attack, [5] is not suggested. The adversary can receive the inputs needed for determining or decoding critical objects from various messages. Schema [5] does not safeguard client login and alter activities by providing transparent format sending user identification and password. Furthermore, the house gateway connection stage is not adequately shielded and the intruder can disclose a safe channel key. In the user manipulation segment, the present assaults are also described.

Our scheme never passes usable and valuable assets through a public network. Therefore, in the suggested plan, the assault on linkability is not achieved. Finally, for this assault, our system verified.

9.25 REJECTION ATTACK

[5] shall be marked as not endorsed in [1]. This is a sort of assault by eavesdropping. An assailant simulates an M2M server and denies client equipment or call to avoid services from being provided by users. In the case of routing attack the [5], home gateway and user units can be redirected to the fraudulent M2M server. The suggested scheme avoids attacks from being eavesdropped and avoids the refusal of our system. After key injection and safety with a MAC confirmation, a home gateway device can be installed online. In order to transmit a valid rejection

signal for the desired object, the intruder should know the integrity key. This assault is not carried out in the suggested system. We completely endorse our plan for this assault for these reasons.

9.26 SUCCESSIVE RESPONSE ATTACK

Scheme [5] in the case of that assault is marked not permitted in [1]. An adversary server simulates the M2M server and enables illegal call permission in this assault. The intruder can pass fraudulent information on the home portal to the user's devices following this method. The scheme [5] change in password and customer registration is projected to operate on a safe transporter, so there is no safety. The home portal device also links the not strongly protected stage according to plan [5]. The home gateway protects the service key that the customer handling assault describes. With MAC verification our scheme defends messages Our proposition, therefore, sends out false messages and avoids consecutive assaults on response. If an intruder wishes to pass a consecutive response then the encryption key should be known and the suggested scheme is not appropriate. We fully approve of our technique of this attack for these reasons.

9.27 PACKET ANALYSIS ATTACK

For this assault method [5] is categorized under [1]. The attacker examines exchange during a transmission to obtain the password, the user ID and vulnerable resources. The login scheme [5] does not show the password; however the design [5] sends user ID and password in a comprehensible format during the registering and the password change phase, which allows to snip it, but ciphers other usages. The suggested packet examination of the system is strengthened and not suitable for the requirements of the attacker. It utilizes the ECDH-key-pair for key transport and message MAC safety. Our design cipher also transforms sensitive assets into a transient public system ID with a cipher key and diverse IDs for anonymity. For these purposes, our plan for this attack is fully supported by us.

9.28 PACKET TRACING ATTACK

Scheme [5] is marked partly backed in [1] related with this assault. This attack tracks items with attributes requests and responses that are supported by critical data processing or decoding identifiers. User ID can be used to define the design [5] data packages. However, all traffic with a session key is ciphered after login, so packet tracking is no problem. There is a considerable challenge with package tracking assaults, however, registry and password replacement requests. Scheme [5] transmits the password and user ID in a straightforward format and packets can be modified by the recipient. The suggested scheme permits an attacker to monitor packets without altering the content of the message. Critical resources for transport and resistance packets are not exposed to the recommended scheme. The structure suggested utilizes the ECDH-key-pair for key transport and message MAC safety. In addition, with the encryption key, our design cipher critical assets also convert IDs into a temporary ID for anonymity on the public network. Our scheme for this assault is completely maintainable.

9.29 BRUTE-FORCE ATTACK

Scheme [5] is not suggested in [1] for that assault. This is not suggested. Two types of brutal attacks are taking place. The first concerns sniffed traffic decoding. Unless the customer register and password update stages in [5] are considered unsafe, then the AES standard algorithm is susceptible to brute force assault. In addition to this, Scheme [5] sends open formatted information (the Home gateway ID, user ID, and password), which is critical, to user registration and password modification. We have safe interaction in our desing for all stages and have used conventional AES and chopped H-MAC algorithms that are resistant to assaults by brute forces. [5] also has no system for revocation. The scheme suggested prevents brute force with customer password, key revocation and the cancelation process for the home portal. The second is about login in on a regular basis. Scheme [5] for this attack set, there is no trial counter. The design suggested enables the attempt to circumvent invalid efforts. We have identified our scheme as being completely protected.

10. PROTOCOL AND SECURITY ANALYSIS OF VANET SCHEME

Cases used to verify formal protocol processed by a Simulation Program C++ supported by Crypto++ and checked for ESAR, EMAP [10] and Normal CRL[13] valid, invalid, and attacker case conduct. The simulation reports each entity performance and resource metrics. Codebase shared by all configurations.

The ESAR scheme guarantees protection against MITM attacks, substitution attacks, message modifications, black hole attacks, false message assaults, attribute-trace and, spam attacks, user manipulation and, successive response attacks which are not protected by EMAP. Additionally both ESAR and EMAP schemes resistance to replay attacks, redirection attacks, forging attacks, colluding attacks, wormhole attacks, changing distance attacks, composition attacks, same-type-device attacks, flooding attacks, eavesdropping attacks, chosen-plaintext attacks, Sybil attacks, identity theft attacks, routing attacks, linkability attacks, rejection attacks, packet analysis attacks, packet tracing attacks, brute-force attacks, DoS attacks. We use ECDSA, ECIES, AES, SHA-1, and H-MAC cryptographic schemes to provide a hybrid and secure authentication and revocation scheme to provide privacy-preserving, anonymity, traceability, data confidentiality and integrity on VANETs. Following subsections includes attack analysis.

10.1 REPLAY ATTACK

For this assault EMAP was appointed carefully recommended [1] and EMAP argues that it was protected. Authentication in the same security system included in ESAR for recurrent assaults is, however, stage shielded. REV_{check} includes a time stamp to provide message replay protection as follows:

$$REV_{check} = F_{mac.HMAC}\left(\left[K_g.x\right]^{128}, PID_s \parallel T_{stamp}\right) \tag{10.1}$$

For these purposes, our protocol has been fully supported by us for the corresponding attack.

10.2 REDIRECTION ATTACK

This assault was partly backed by EMAP in [1]. An intruder initiates a redirect action by simulating the critical resources TA server or RSUs. However, there are RSU certificates produced by TA. Also, TA certificate initial upload provided over a secure channel that not considered on both EMAP and ESAR schemes. For this, purpose both we list EMAP and ESAR scheme as being partially supported. If we create initial certificate upload security than this attack, security will be fully approved.

10.3 MITM ATTACK

In [1], EMAP has been recognized in its support. However, there is no encrypted K_g verification during the revocation process; for this reason, EMAP does not protect from MITM attacks. However, in ESAR we provide, encrypted K_g source verification with certificate control and integrity control with signature, also we provide a timestamp to detect value is new. We identified our system as completely endorsed for these artifacts.

10.4 SUBSTITUTION ATTACK

In [1], EMAP confessed that this assault was partially endorsed. The replacement attack is a sort of MITM assault to substitute initial implementation with derivative applications to leak transmitting data. The MITM attack scenario is similarly corrupt encrypted K_g. A device can work as OBU and send the wrong K_g in EMAP scheme. We provide a solution mentioned in the MITM attack. For these purposes, we have completely endorsed our system.

10.5 FORGING ATTACK

EMAP believed that this assault was completely supported [1]. EMAP build on ECDLP problem, so both EMAP and ESAR schemes similar provide security to forging attacks for sensitive assets such as keys. We structured our program as completely endorsed for these purposes.

10.6 COLLUDING ATTACK

[1] List the EMAP suggested for this assault carefully. For current demands, the intruder may use ancient information. Both EMAP and ESAR never provide true information capable of disclosing valuable property. Our scheme is considered to be completely endorsed by us in these analyzes.

10.7 WORMHOLE ATTACK

In[1], EMAP considered that this assault was partially accepted. Wormhole intruder can prevent or customize packets from one place to another on the network. This assault affects time and interaction. There are no delicate properties disclosed by both EMAP and ESAR and timeout security. That is why. It is considered that we are strongly recommending our ESAR system for this assault.

10.8 MESSAGE MODIFICATION ATTACK

In[1], EMAP mentioned that this assault is not being endorsed. The K_g encrypted origin and integrity verification are absent from the EMAP system revocation message This violates the security of the message change

All requests are protected by anonymous certificates and message signatures in the ESAR scheme. We accepted that our view for this assault was completely advised.

10.9 BLACKHOLE ATTACK

EMAP is allocated to this assault under [1] as partly endorsed. An intruder with a black hole gives a fake reply to messages broadcasted to block relationships. Requests and reactions are shielded both by EMAP and ESAR. Except for encrypted K_g request but it does not effect on this attack. Without an integrity key the attacker can't give a valid answer to the requester. The attacker cannot influence the communication channel without such keys. This type of assault also impacts the timing of the application, and this attack is detected by the suggested time assessment function. We say that our plan suggested this assault carefully.

10.10 CHANGING DISTANCE ATTACK

In [1], EMAP has been categorized as not endorsed. However, it is not valid authentication messages have REV_{check}, thus:

$$REV_{check} = F_{mac.HMAC}\left(\left[K_g.x\right]^{128}, PID_s \parallel T_{stamp}\right) \tag{10.2}$$

Revocation check value contains a timestamp, and this provides us a message expire time detection to protect from changing distance attacks. An assailant may block a application and then transmit one to the recipient organization after evaluation. The evolving range attacks have a significant impact on the timing and operational moment tracking for suspected activities of both EMAP and ESAR systems. This view also discovers timeouts for each application. We have instructed our system to be completely endorsed for these reasons.

10.11 COMPOSITION ATTACK

The labeling of EMAP as not endorsed in [1]. Composition assaults are based on gathering several communication channel message characteristics. The structure of these characteristics shows specific assets. There are no valuable assets revealed by both EMAP and ESAR systems. That is why we have completely endorsed both the EMAP and the ESAR schemes.

10.12 SAME-TYPE-DEVICE ATTACK

EMAP was categorized in [1] for this assault as not endorsed. However, due to HSM hardware, we endorsed EMAP completely for this assault. The same approach applied in ESAR is that of equipment HSM cannot reproduce intervention on HSM that damages all valuable property. We described our system as completely endorsed, according to these outcomes.

10.13 FLOODING ATTACK

In [1], EMAP recognized that this assault was not endorsed. During the configuration of the communication channel, flooding assaults such as SYN FLOODING happen. The first application may be sniffed and replayed by the intruder to disable facilities. Usually, our protocol does no longer have any SYN messages, which is why we do not consider this attack to be completely endorsed in both EMAP and ESAR systems.

10.14 FALSE MESSAGE ATTACK

EMAP was appointed in [1] for which it was not suggested. An attacker may send malicious messages in order to stop the procedure. EMAP revocation phase encrypted K_g source and integrity validation are missing. An attacker can send invalid encrypted buffer to corrupt target EKI keypairs. This situation disables OBU forever. In the ESAR scheme, We filled this missing source and integrity validations to avoid false message injections. We have identified ESAR as supporting the assault in its entirety.

10.15 ATTRIBUTE-TRACE ATTACK

In [1], EMAP has been marked as not endorsed. An intruder traces information from all gathered information and traces characteristics. EMAP characteristics, however which have not been disclosed, are also strengthened by ESAR for such assaults. Hence, tracing attributes strengthened for detection of critical information. For this assault we considered the ESAR system as completely endorsed.

10.16 EAVESDROPPING ATTACK

For this assault, EMAP has chosen by [1] as partly endorsed. Eavesdropping assaults relying on channel surveillance and helpful data on individuals For this purpose, EMAP and ESAR systems send out the encrypted confidential properties. We have identified our ESAR system as completely endorsed.

10.17 CHOSEN-PLAINTEXT ATTACK

For this assault EMAP was categorized in [1] as not endorsed. Selected plaintext is a brutal force assault which allows you to randomly select simple text and identify coded values in order to uncover secrets.

The current scheme is not suitable for this kind of attacks. The attacker can select no plaintext. Each property is stored and send in binary form. Both EMAP and ESAR schemes build on ECDLP problem, and this problem is hard for brute force processes. This attack was completely endorsed by our plan.

10.18 SYBIL ATTACK

For this assault in [1], EMAP marked partly endorsed. Sybil assaults by adding OBU identification to the existing network. Anonymous certificates protect EMAP and ESAR schemes from sybil attacks. We both recognized our project as completely endorsed, and EMAP was completely endorsed.

10.19 SPAM ATTACK

EMAP has been certified for this attack by [1] to be partly endorsed. The intruder can access the network and deliver the incorrect signal to the scheme, or alter the initial signal information, to break down the communication provider and states. The intruder can modify messages and submit the channel to stop interaction by sending false messages. EMAP and ESAR schemes provide anonymous certificates and message signatures and after signature validation received message is processed except encrypted K_g request during revocation. Spam attack can be efficient on the revocation process, but authentication guarded in EMAP. The ESAR scheme provides end-to-end security for both authentication and revocation. For these purposes, our plan has been recognized as completely endorsed.

10.20 IDENTITY THEFT ATTACK

In [1], EMAP was designated as partly endorsed. An assailant watches and simulates messages. EMAP and ESAR both current pseudo-identification information on anonymous certificates. Never on public broadcasters did the original identity information. That is why we considered this assault to be completely endorsed.

10.21 USER MANIPULATION ATTACK

In [1], EMAP has mentioned that this assault is not backed. During the revocation phase, encrypted K_g request can manipulate by the attacker, and for this reason, the user assumes that services blocked. This problem can be an effect on user behaviors during driving, and this issue can increase traffic jam. ESAR scheme solves this issue by protecting encrypted K_g request with signature validation. That is why we have regarded our system as completely endorsed.

10.22 ROUTING ATTACKS

EMAP has been rated not endorsed for this assault in [1], but timeouts can prevent routing assaults by both EMAP and ESAR. That is why our system is completely endorsed for this assault.

10.23 LINKABILITY ATTACK

In [1], EMAP is considered unavailable for the assault. The attacker can obtain the necessary inputs from several texts to determine or decode critical resources in this assault. secure and valuable resources are never transmitted to a public network. The connectivity assault therefore does not effect to the ESAR and EMAP systems that were suggested. Our system completely backed the assault.

10.24 REJECTION ATTACK

EMAP is marked as not endorsed in [1] for the assault. This is a sort of assault by eavesdropping. An assailant is simulating an OBU or TA server and denies blocking service applications. The assault of dismissal does not cover the ESAR and EMAP systems suggested. Our system endorsed this assault completely.

10.25 SUCCESSIVE RESPONSE ATTACK

For this assault, in [1], EMAP was categorized as unsupported. In this assault, an opponent OBU recognizes the application and transmits incorrect data to the recipient's set property and setup.

ESAR fully maintained for this attack, but EMAP revocation phase encrypted K_g request own open issue to the corrupt sensitive asset of the receiver. If an intruder wishes to provide a consecutive reply, a valid certificate for each message should be available. Our technique completely endorsed this assault for these purposes.

10.26 PACKET ANALYSIS ATTACK

For this assault in [1], EMAP is categorized as partly assured. The adversary examines operations during this assault to gather important property. The attacker can analyze OBU payload messages in the EMAP scheme that are not partially encrypted. Never released risky properties. Similarly, packet assessment of the ESAR system does not expose valuable properties. Our system carefully advocated this assault for these purposes.

10.27 PACKET TRACING ATTACK

For this assault EMAP is mentioned in [1] in part as supported. In this assault, packets monitored with characteristics and demands are combined with delicate data descriptors to retrieve or decode. EMAP and ESAR both traceable packets but do not expose risky property. Our system endorsed this assault completely for these purposes.

10.28 BRUTE-FORCE ATTACK

The EMAP is categorized in [1] as not supporting this assault, but for a brute-force assault, the ECDLP issue is difficult. Both ESAR and EMAP systems backed such an assault.

10.29 DOS ATTACK

In [1], EMAP has stated that this attack is not endorsed. An attacker DoS moves a majority of data to the scheme in order to use additional operating system resources to avoid servers. For identification of this type of assault, EMAP has timeout security. Furthermore, ESAR does not offer a complete DoS alternative. For this purpose, the ESAR layout was recognized as partly endorsed.

11. M2M SCHEMES PERFORMANCE AND SECURITY COMPARISONS

We compare our design protection with prior concepts in this chapter [4]–[8], [27]–[30], and compare our design output to past models [5], respectively. Our study requires the outcomes of the past survey [1] into account and adds to the outcomes the suggested system evaluation. Security evaluation attacks are displayed in Table 11.1 with system comparisons. Furthermore, how our suggested scheme offers more safety and countermeasures for attacks than former models described in Table 11.1.

The suggested technique was also contrasted with [5] for efficiency, network congestion ,and resource use. The operating notations are provided in Table 11.2 and Table 11.3 indicates operating system costs [5] and the design suggested. The calculated times are focused on single runtime and the recorded times are based on simulated MATLAB sequences. The efficiency similarities are shown in Figure 11.1. Seven additional H-MAC procedures in our implementation are consistent with our scheme [5] in the most frequently used login and authentication mechanism. The H-MAC provides protection for anonymity and change of messages. H-MAC procedure improves present login and authentication times by just 0.088101 seconds, so user behavior is acceptable according to [86]–[88] advices for system latency thus; 0.1 second latency is the limit for consumers to feel that they manipulate items in the user interface straight. 1 second latency is threshold for users who feel free to work through the safety zone without excessively waiting for the device and the user retains attention for 10 seconds for latencies. According to this constants our proposed M2M scheme all phases completes operations less than 1 second and users can accept this latency.

The system [5] does not describe re-joining procedure of a home gateway and initial key-injecting procedure of the customer logo and home gateway. We added these phases to the proposed M2M scheme. Home gateway re-join operation only occurs when the home gateway restarts. This operation only takes 0.212503 seconds, and the operation speed is acceptable. User logout is a frequently used operation, and it very fast takes only 0.005413 seconds. Moreover, the home gateway key's injection operation only used when the user and home gateway paired. This operation only takes 0.001622 seconds. All operation processing time increments in the proposed scheme are related to security improvements such as using H-MAC cryptogram. Figure 11.3 and Figure 11.2 includes network congestion and resource usages comparisons of schemes. Using new MACs, message identifiers and message acknowledgments increase network congestion. Adding additional security protections comes with new security assets and this cause a bit of resource usage during operations.

Table 11.1: Security Comparison [1]

	Replay Attack	Changing DistanceAttack	Same-Type-Device Attack	Composition Attack	Redirection Attack	Man-In-The-Middle Attack	Substitution Attack	DoS Attack	Forging Attack	Colluding Attack	Flooding Attack	False Message Attack	Sybil Attack	Message Modification	Wormhole Attack	Blackhole Attack	Attribute-Trace Attack	Eavesdropping Attack	Chosen-Plaintext Attack	Spam Attack	Identity Theft Attack	User Manipulation Attack	Routing Attack	Linkability Attack	Rejection Attack	Attack	Packet Analysis Attack	Packet Tracing Attack	Brute-Force Attack
Our	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
[5]	✓					✓						✓	✓		✓	✓		✓		✓	✓	✓	✓				✓	✓	
[4]	✓				☑	☑	✓	✓							✓	✓											✓	✓	✓
[6]	✓				☑	☑	✓	✓	✓	✓					✓	✓													✓
[7]					✓			✓		✓	✓	✓				✓						✓							✓
[8]	✓					✓	✓					✓	✓	✓		✓	✓	✓	✓	✓									✓
[27]	☑	☑	☑	☑	✓	✓	✓			✓	✓	✓			✓	✓				✓	✓	✓	✓	✓					✓
[28]					✓	✓	✓																						
[29]																													✓
[30]	✓				☑	☑					✓	✓			✓	✓		✓	✓	✓	✓	✓					✓	✓	

Fully Supported = ☑, Partial Supported = ✓, Empty Not Supported.

Table 11.2: Operation Notations and Approximately Single Operations

Operation Name	Operation Notation	Approximately Time (sn)
ECDH Generate Key Pair	EI	0.009486
ECDH Key Derive	EG	0.004638
ECDH Encryption	EE	0.001605
ECDH Decryption	ED	0.000293
AES Encryption	SE	0.073256
AES Decryption	SD	0.068992
H-MAC – SHA256	HM	0.000267
SHA-1	HH	0.046198

Table 11.3: Operation Comparison Between Sun et al.’s [5] and the Proposed Scheme

Scheme Operations	User		M2M Server		Home Gateway		Total	
	Proposed	[5]	**Proposed**	[5]	**Proposed**	[5]	**Proposed**	[5]
M2M Server Setup	0	0	$1EI$	0	0	0	$1EI$	0
Home Gateway Setup	0	0	0	0	0	0	0	0
User Setup	$1EG + 1HH$	0	0	0	0	0	$1EG + 1HH$	0
User Registration and Key Pairing	$1SE + 1SD + 2HM + 1EG + 1ED$	0	$2SE + 1SD + 4HM + 1EG + 1EE + 1HH$	$1SE + 1HH$	0	0	$3SE + 2SD + 6HM + 2EG + 1EE + 1ED + 1HH$	$1SE + 1HH$
User Login And Authentication	$1SE + 3HM + 4HH$	$1SE + 5HH$	$2SD + 4HM + 5HH$	$2SD + 4HH$	0	0	$2SD + 1SE + 7HM + 9HH$	$2SD + 1SE + 9HH$
User Home Gateway Key Injection	0	N/A	0	N/A	0	N/A	0	N/A
Home Gateway Join Network	0	0	$1SD + 2SE + 10HM + 5HH$	$2HH$	$1SD + 4HM + 1HH$	$1HH$	$2SD + 2SE + 14HM + 6HH$	$3HH$
User Password Change	$1SE + 1DE + 3HM$	0	$3SE + 1SD + 4HM + 2HH$	$2SE + 2HH$	0	0	$4SE + 2SD + 7HM + 2HH$	$2SE + 2HH$
User Logout	$2HM$	N/A	$3HM$	N/A	0	N/A	$5HM$	N/A
Home Gateway Re-Join Network	0	N/A	$1SD + 2SE + 11HM + 5HH$	N/A	$1SD + 5HM + 1HH$	N/A	$2SD + 2SE + 16HM + 6HH$	N/A

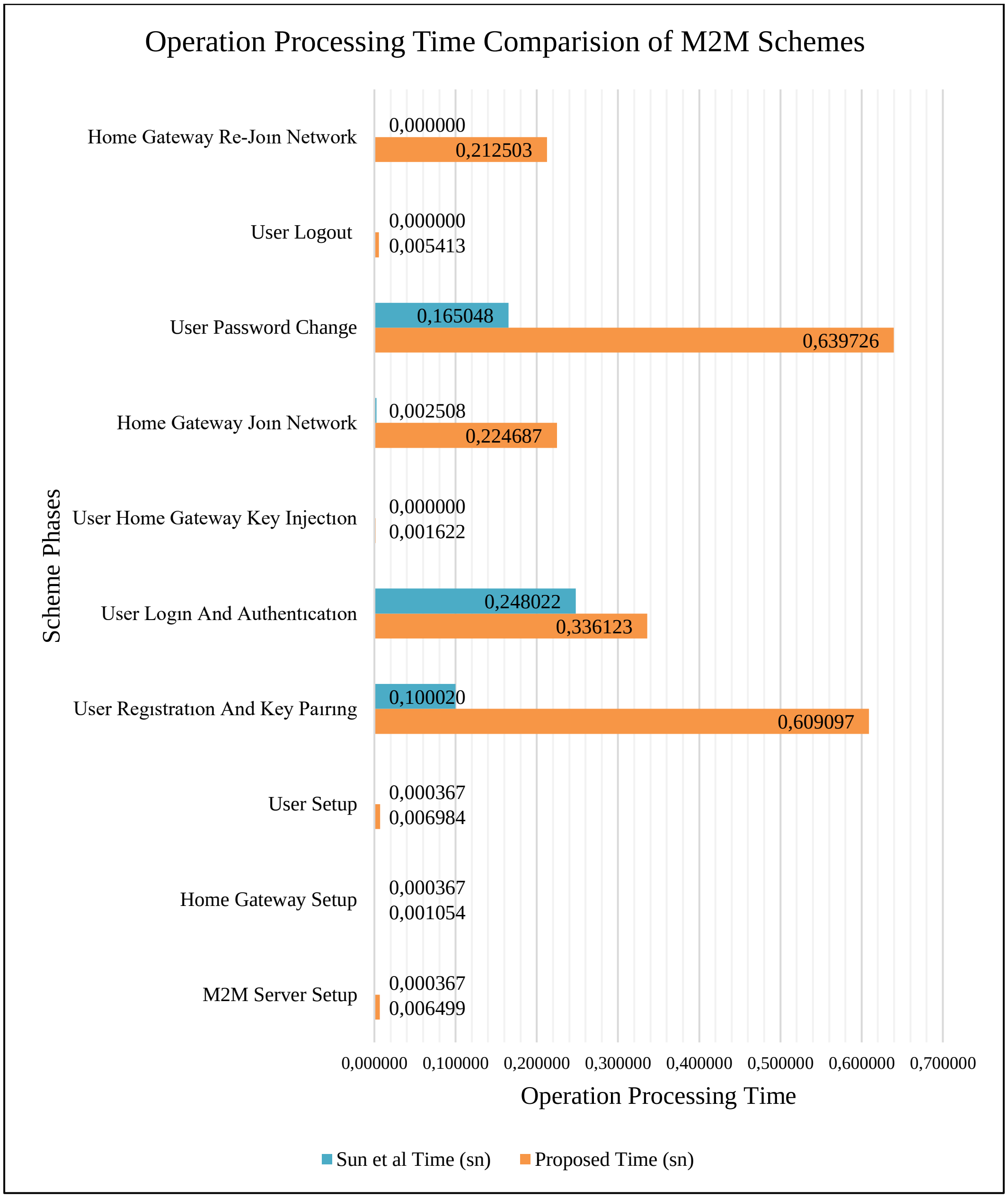


Figure 11.1: Comparison of Operational Performance Between Sun et al. ‘s[5] and the System Suggested

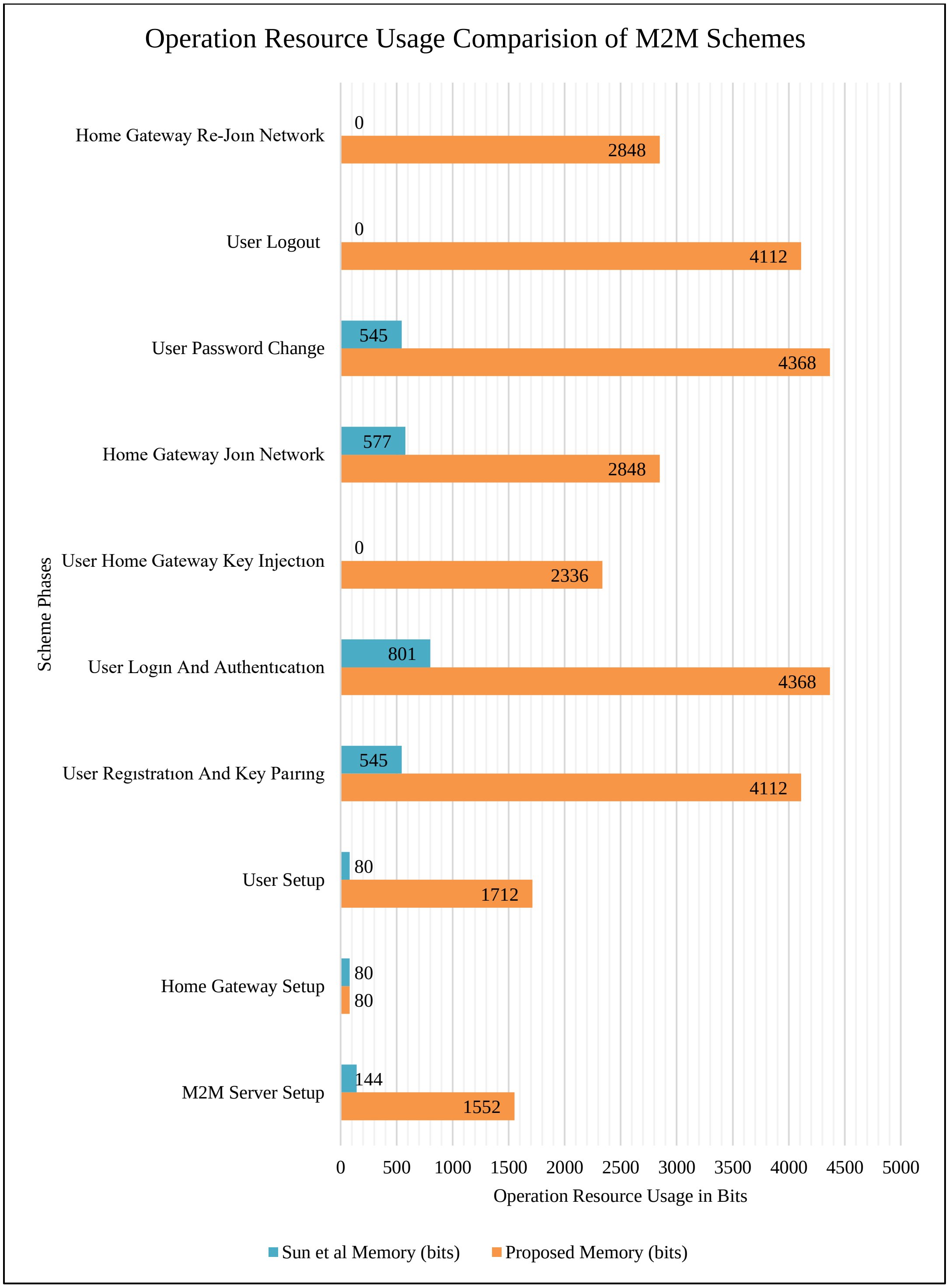


Figure 11.2: Comparison of Operating Resource Usage Between Sun et al. [5] and System Suggested

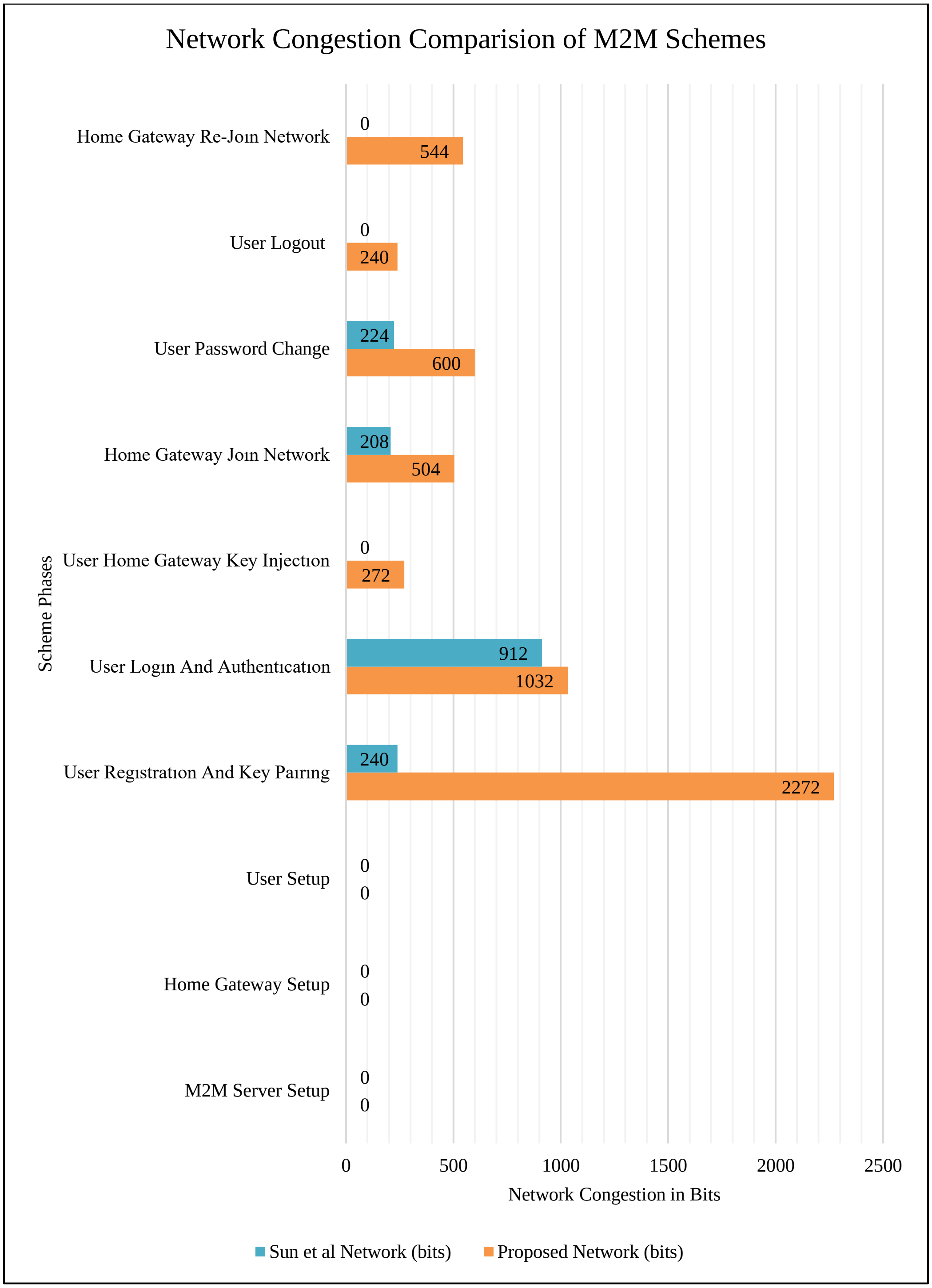


Figure 11.3: Comparison of Network Congestion Between the Suggested System and Sun et al. [5]

12. VANET SCHEMES PERFORMANCE AND SECURITY COMPARISONS

We compare the safety of our scheme in this chapter with earlier layouts [2], [40], [48], [51], [53], [55]–[57]. In addition, we compare the efficiency of our ESAR proposition to the earlier EMAP design [10]. The findings of the prior system are derived from the investigation [1], and linked to the outcomes is the suggested system evaluation. The threats in Table 12.2 showed the safety assessment. The ESAR regime offers for more safety and countermeasurements than prior designs in terms of safety comparative outcomes in Table 12.2. Table 12.2 shows. The suggested ESAR system has been contrasted with the efficiency and network congestion EMAP [10] system. Our development does not have an impact on storage resource usage; for this reason, we do not consider this comparison. Table 12.1 shows the operations notation description used on graphics.

Table 12.1: Graphic Operation Notations and Descriptions

Operation Name	**Operation Description**
Normal+LinearSearch	Only CRL used for revocation check
Normal+BinarySearch	Only CRL used for revocation check
Esar+LinearSearch	ESAR scheme with linear search
Esar+LinearSearch NoKgSenderVerify	ESAR scheme used for revocation check without encrypted K_g key sender verification disabled.
Esar+LinearSearch NoIDRevKeyMerge	ESAR scheme used for revocation check without $ID_{rev_{key}}$ and encrypted K_g request merge disabled.
Esar+BinarySearch	ESAR scheme with binary search
Esar+BinarySearch NoKgSenderVerify	ESAR scheme used for revocation check without encrypted K_g key sender verification disabled.
Esar+BinarySearch NoIDRevKeyMerge	ESAR scheme used for revocation check without $ID_{rev_{key}}$ and encrypted K_g request merge disabled.
Emap+LinearSearch	EMAP scheme with linear search
Emap+BinarySearch	EMAP scheme with binary search

LinearSearch [62] and BinarySearch [62] notation shows CRL search method used in the algorithm.

The binary search required sorting operation processed with quicksort [62]. The linear search list not sorted.

Table 12.2: Security Comparison [1]

	Replay Attack	Changing DistanceAttack	Same-Type-Device Attack	Composition Attack	Redirection Attack	Man-In-The-Middle Attack	Substitution Attack	DoS Attack	Forging Attack	Colluding Attack	Flooding Attack	False Message Attack	Sybil Attack	Message Modification	Wormhole Attack	Blackhole Attack	Attribute-Trace Attack	Eavesdropping Attack	Chosen-Plaintext Attack	Spam Attack	Identity Theft Attack	User Manipulation Attack	Routing Attack	Linkability Attack	Rejection Attack	Successive Response Attack	Packet Analysis Attack	Packet Tracing Attack	Brute-Force Attack
ESAR	☑	☑	☑	☑	✓	☑	☑	✓	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	✓
EMAP	☑	☑	☑	☑	✓			✓	☑	☑	☑		☑		☑	✓	✓	☑	☑	✓	☑		☑	☑	☑		☑	☑	✓
[40]	✓											✓	✓		✓	✓	✓	✓	✓				✓		✓		✓	✓	✓
[56]	☑													☑	✓	✓		✓				✓	✓	✓	✓		✓	✓	✓
[55]	✓					✓	☑							✓	✓	✓			✓	✓	✓	✓	✓						
[48]	✓					☑	✓	☑	✓				✓				✓			✓				✓	✓	✓	✓	✓	✓
[2]	✓							☑				☑	☑		✓	✓		✓					✓				✓	✓	
[53]								☑		✓	✓							✓	✓	✓	✓	✓							
[57]	☑					✓	✓											✓					✓						
[51]	☑				✓	☑	✓	☑	✓	✓			✓		✓	✓							✓				✓	✓	

Fully Supported = ☑, Partial Supported = ✓, Empty Not Supported.

We have implemented C++ simulation with parameters in Table 12.3

Table 12.3: VANET Simulation Parameters

Parameter	**Value**
OBU Size	40
Revoked OBU Size	10
OBU EKI KeyPair Size	50
EKI Key Pair Pool Size	2000
OBU PKI KeyPair Size	50
Hash Chain Size	200
Revocation Count	4
Missed Revocation Interval	2-3
Simulation Repeat Count	10

Simulation results reported for each configuration and each phase. We take into account the EMAP scheme with a binary search and the ESAR scheme with binary search configurations for comparisons. According to the results of our simulations, version controls, reordering operations, merging requests and provide signed encrypted K_g improvement provide 13.8262% better end-to-end operation performance than EMAP scheme that we show in Figure 12.1. Besides, setup phase performance is 0.08452% worse than EMAP, and the reason for that is using unique anonymous certificates and EKI key pairs for each OBU. performance results are in Figure 12.2.

Authentication phase performance is 12.14591% better than EMAP because of avoiding revoked OBUs signal processing and sending that shown in Figure 12.3. Also, we separately analyzed authentication performance for the sender and receiver OBUs. Authentication phase for sender OBU is 21.2976% better than EMAP that shown in Figure 12.5, and for the receiver, OBU is 15.4416% worse than EMAP that show in Figure 12.6.

Revocation phase performance has dramatic increment because of avoiding unnecessary operations by merging requests, checking revocation version and reordering operations. Revocation phase performance is 50.83966% better than EMAP that shown in Figure 12.4. Revocation phase detailed analyzes reported for TA and OBUs. Revocation performance for TA 55.1389% better than EMAP that shown in Figure 12.7. Besides, revocation phase OBU performance is 51.08154% better than EMAP that shown in Figure 12.8. Network congestion analysis shows that additional operation removing with merge request and avoid revoked OBUs request decrease network congestion and ESAR congestion is 13.29079% better than EMAP in Figure 12.9. Authentication phase network congestion is 20% better than EMAP in Figure 12.10. Also, adding sign request for encrypted Kg increase network congestion and this trade-off result of revocation phase network congestion is 24.5203% worse than EMAP in Figure 12.11. According to our results, ESAR is better than EMAP.

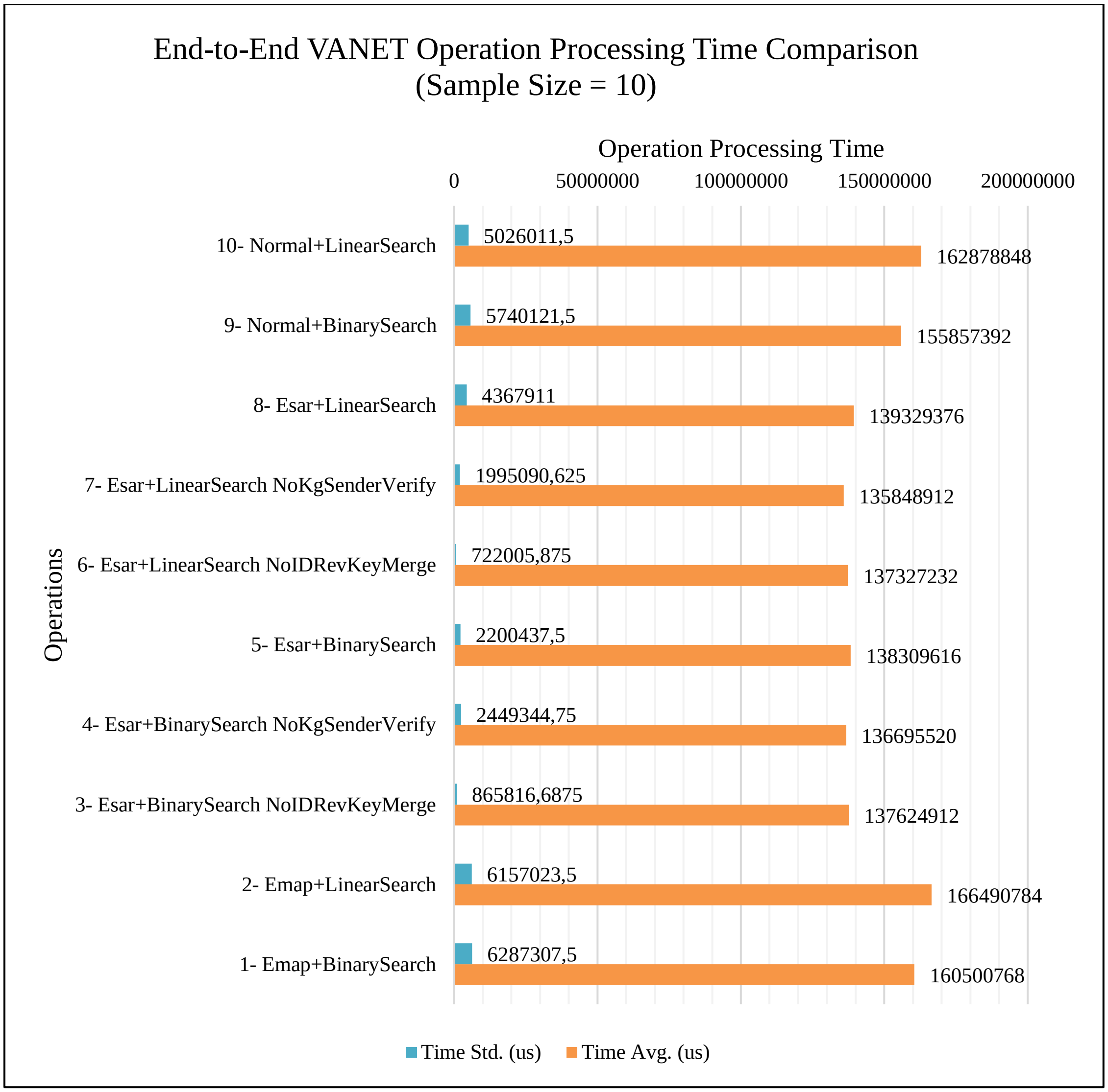


Figure 12.1: End-to-End Operation Time Comparison Between ESAR and EMAP Configurations.

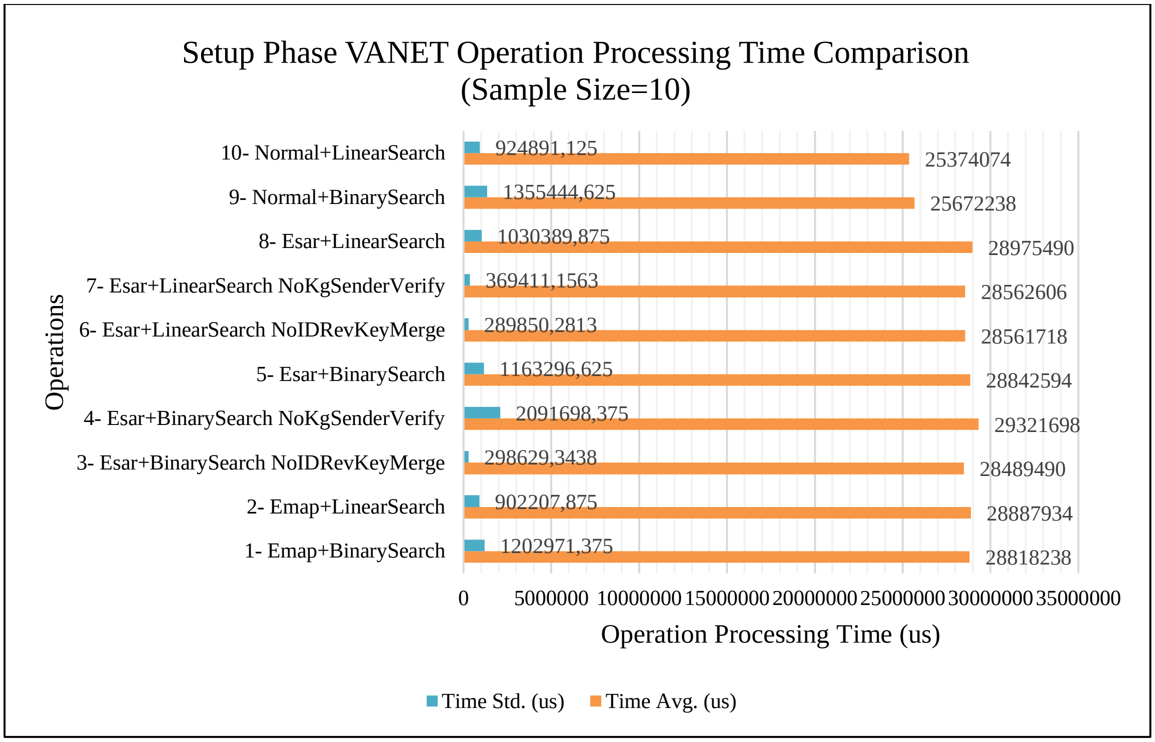


Figure 12.2: Setup Phase Operation Time Comparison Between ESAR and EMAP Configurations.

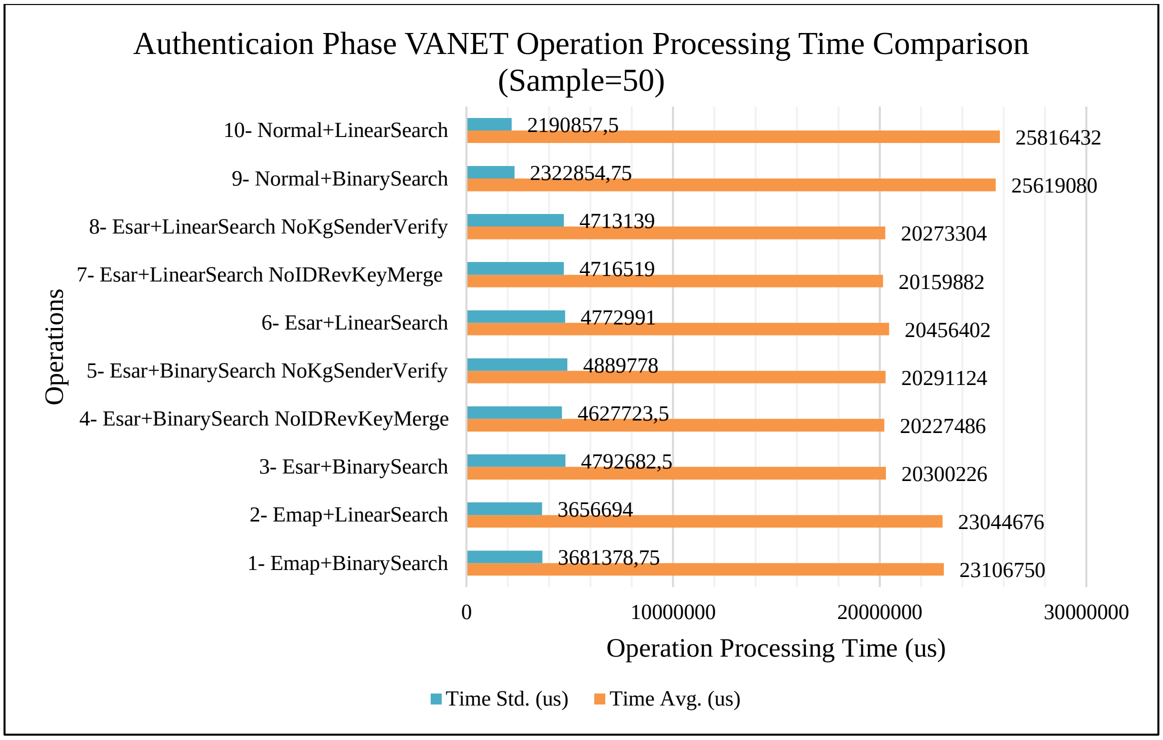


Figure 12.3: Authentication Phase Operation Time Comparison Between ESAR and EMAP Configurations.

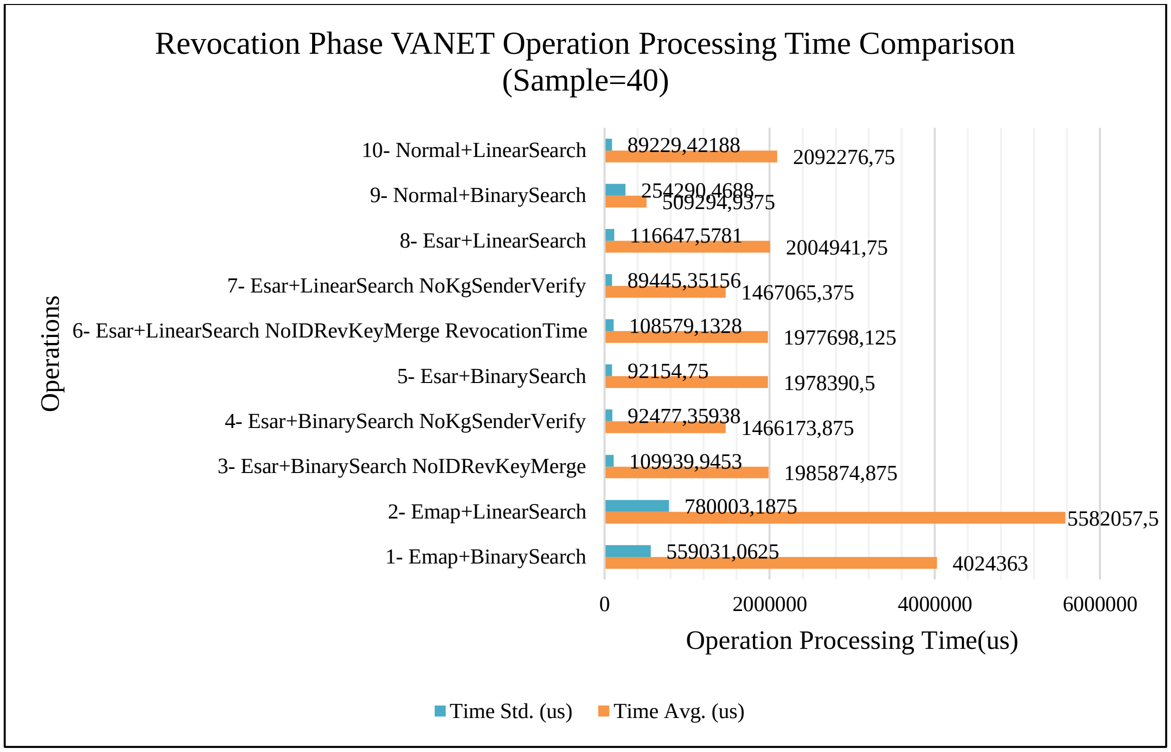


Figure 12.4: Revocation Phase Operation Time Comparison Between ESAR and EMAP Configurations.

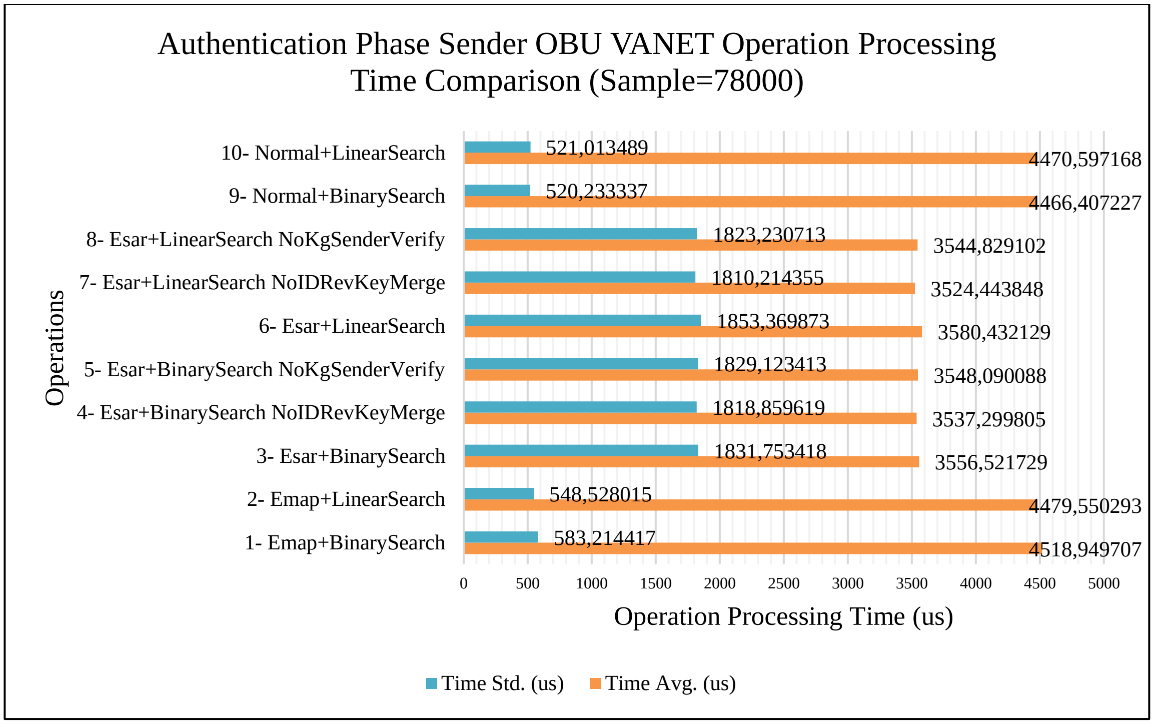


Figure 12.5: Authentication Phase Sender OBU Operation Time Comparison Between ESAR and EMAP Configurations.

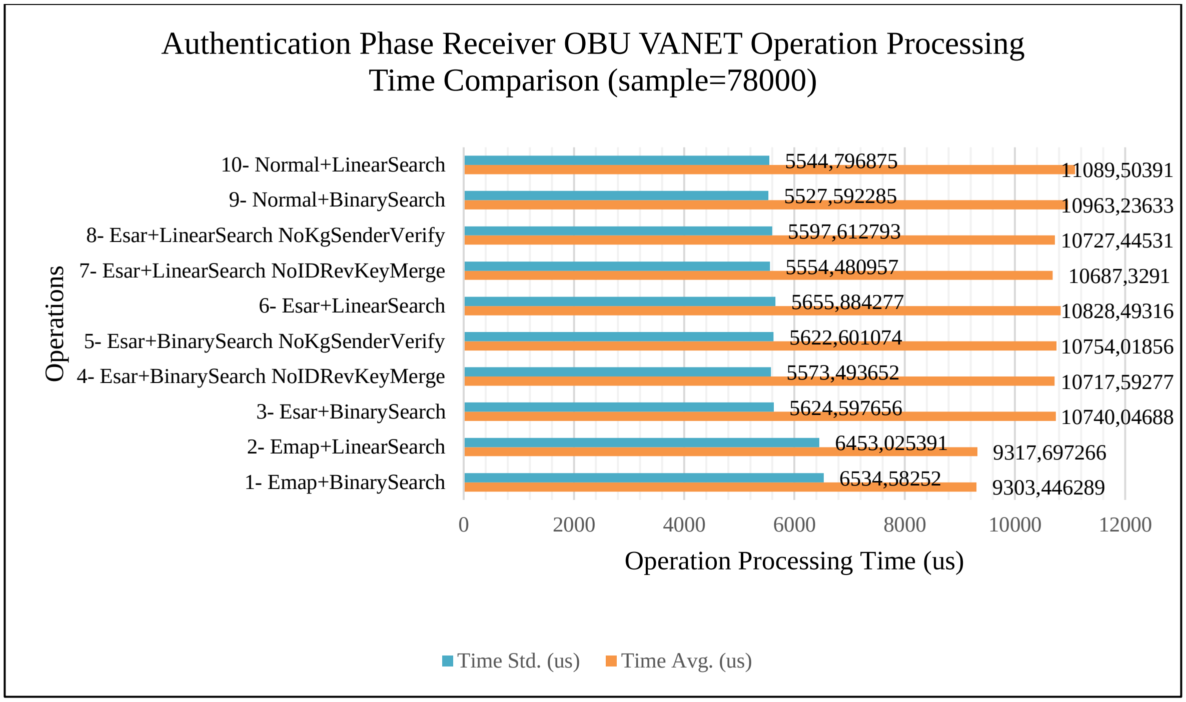


Figure 12.6: Authentication Phase Receiver OBU Operation Time Comparison Between ESAR and EMAP Configurations.

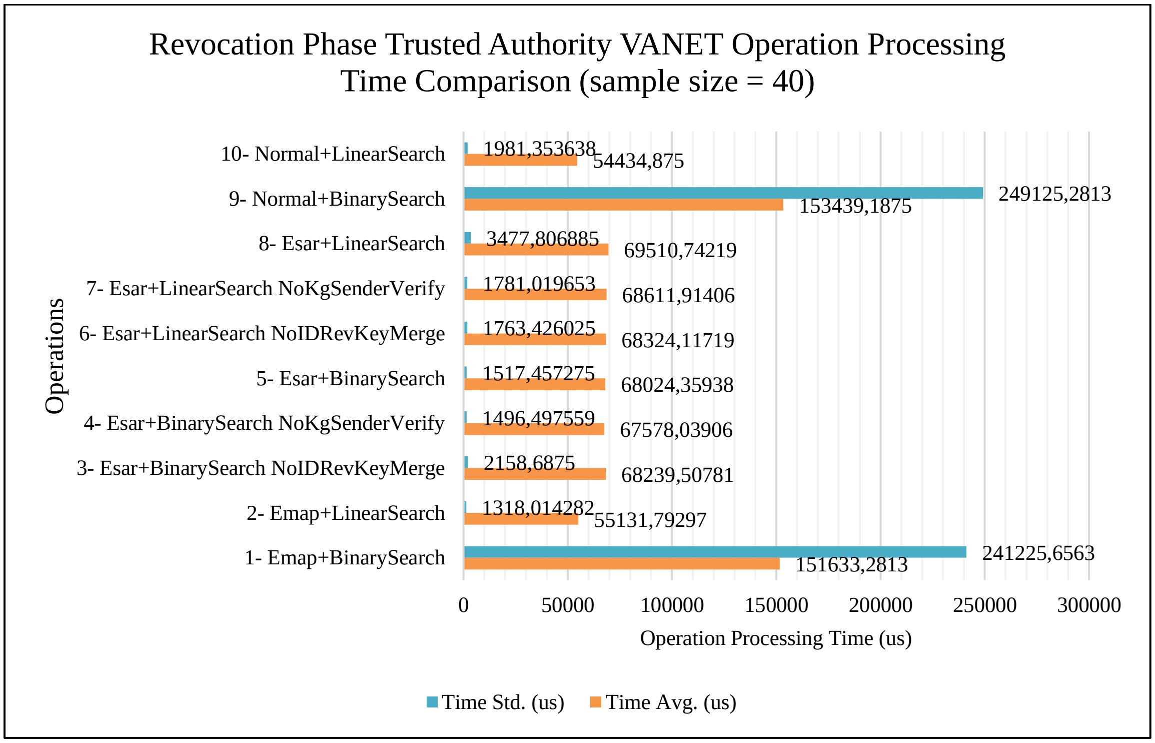


Figure 12.7: Revocation Phase Trusted Authority Operation Time Comparison Between ESAR and EMAP Configurations

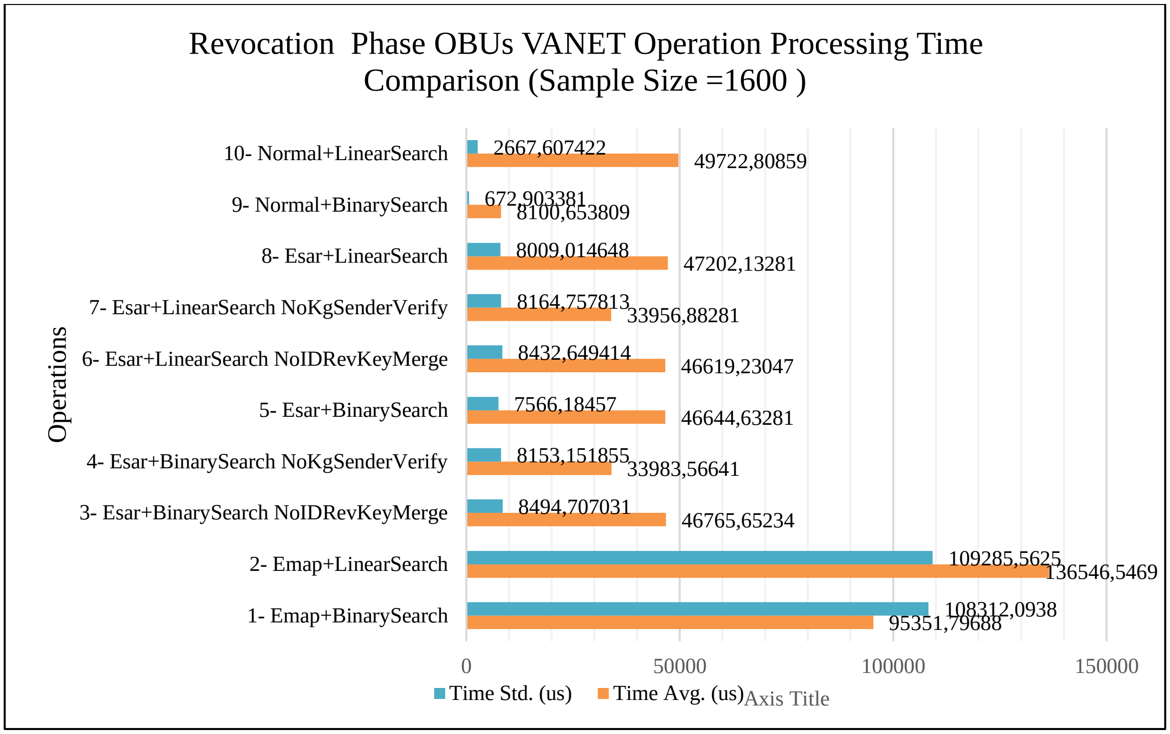


Figure 12.8: Revocation Phase OBUs Operation Time Comparison Between ESAR and EMAP Configurations.

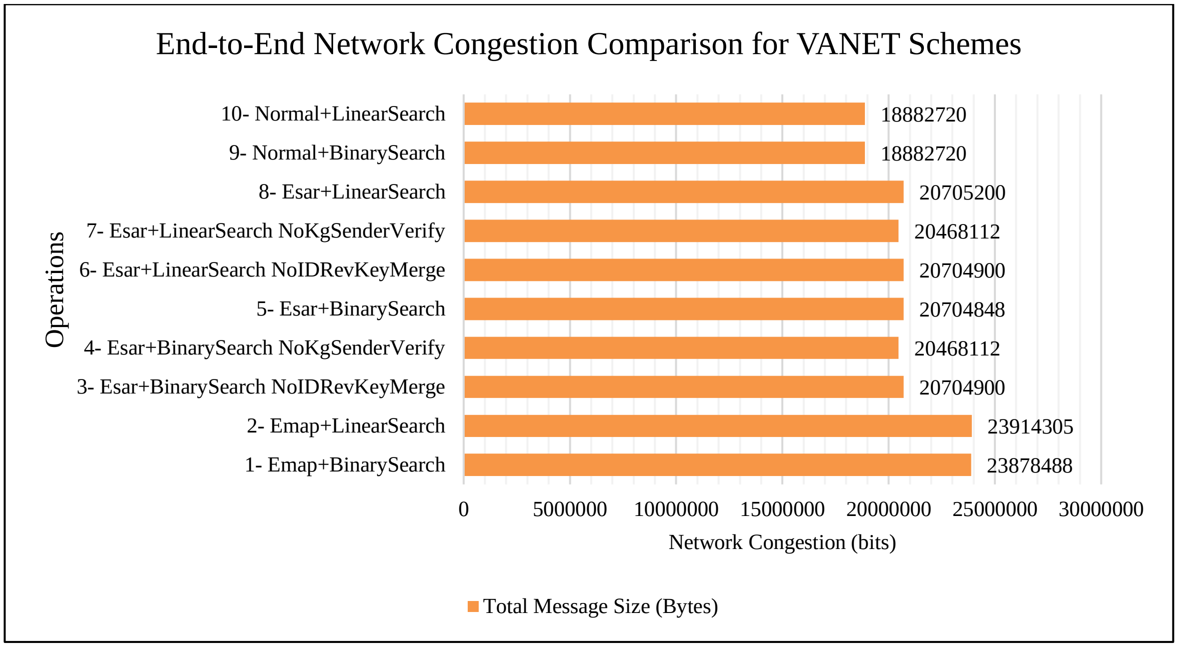


Figure 12.9: End-to-End Network Congestion Comparison Between ESAR and EMAP Configurations.

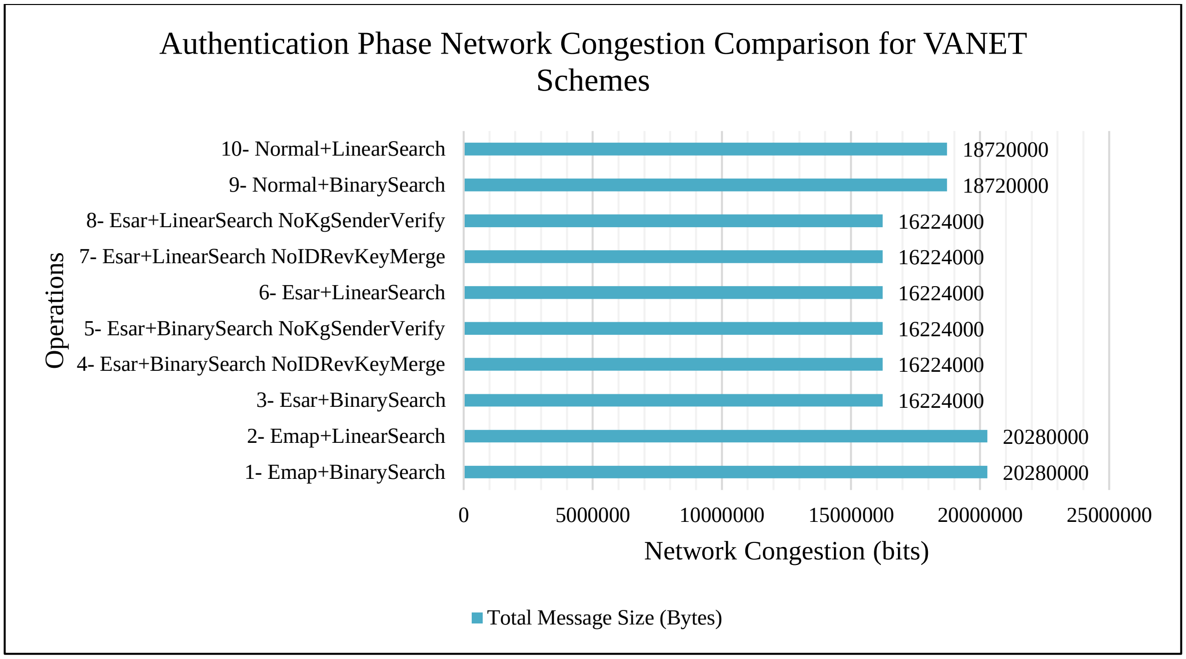


Figure 12.10: Authentication Phase Network Congestion Comparison Between ESAR and EMAP Configurations.

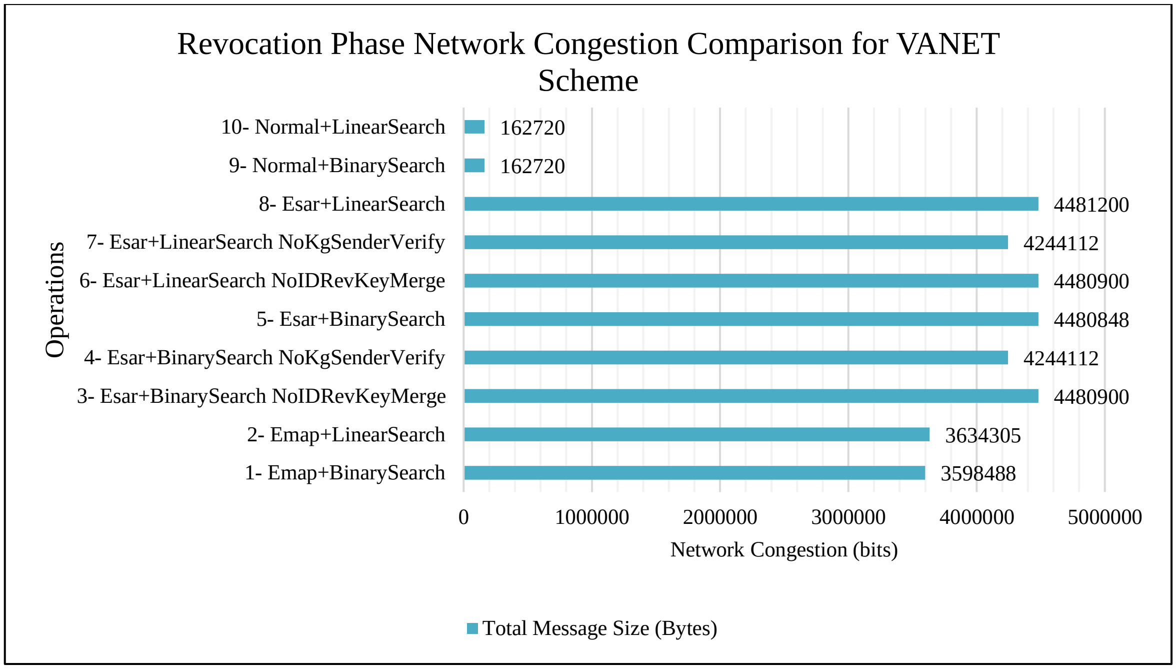


Figure 12.11: Revocation Phase Network Congestion Comparison Between ESAR and EMAP Configurations.

13. CONCLUSION AND FUTURE WORK

This thesis includes two end-to-end secure M2M and VANET communication systems. Both systems suggested are effective than systems comparable to work domain requirements. This thesis provides all necessary analyzes of the M2M system, and the interaction between sensors and home gateways will be considered for the planned study. We will provide secure revocation and authentication schemes for IoT devices. In another side, proposed ESAR scheme is efficient than EMAP scheme according to our simulation results. For plan, the registration phase will take into account and social network as a user will be linked to OBU to provide a social VANETs for drivers. Besides, we will provide crash evidence with the blockchain. We will keep a message sequence with blockchain to provide non-reputation and traceability in car crash events.

REFERENCES

[1] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, “Authentication Protocols for Internet of Things: A Comprehensive Survey,” *Secur. Commun. Networks*, vol. 2017, pp. 1–41, Dec. 2017.

[2] L. Zhang, Q. Wu, J. Domingo-Ferrer, B. Qin, and C. Hu, “Distributed Aggregate Privacy-Preserving Authentication in VANETs,” *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 3, pp. 516–526, Mar. 2017.

[3] T.-Y. Byun, “ICSW2AN : An Inter-vehicle Communication System Using Mobile Access Point over Wireless Wide Area Networks,” vol. 78, 2010, pp. 355–366.

[4] C. Lai, R. Lu, D. Zheng, H. Li, and X. (Sherman) Shen, “GLARM: Group-based lightweight authentication scheme for resource-constrained machine to machine communications,” *Comput. Networks*, vol. 99, pp. 66–81, Apr. 2016.

[5] X. Sun, S. Men, C. Zhao, and Z. Zhou, “A security authentication scheme in machine-to-machine home network service,” *Secur. Commun. Networks*, vol. 8, no. 16, pp. 2678–2686, Nov. 2015.

[6] C. Lai, H. Li, R. Lu, and X. (Sherman) Shen, “SE-AKA: A secure and efficient group authentication and key agreement protocol for LTE networks,” *Comput. Networks*, vol. 57, no. 17, pp. 3492–3510, Dec. 2013.

[7] C. Lai, H. Li, X. Liang, R. Lu, K. Zhang, and X. Shen, “CPAL: A Conditional Privacy-Preserving Authentication With Access Linkability for Roaming Service,” *IEEE Internet Things J.*, vol. 1, no. 1, pp. 46–57, Feb. 2014.

[8] A. Fu, S. Lan, B. Huang, Z. Zhu, and Y. Zhang, “A Novel Group-Based Handover Authentication Scheme with Privacy Preservation for Mobile WiMAX Networks,” *IEEE Commun. Lett.*, vol. 16, no. 11, pp. 1744–1747, Nov. 2012.

[9] X. Li, Y. Xiong, J. Ma, and W. Wang, “An efficient and security dynamic identity based

authentication protocol for multi-server architecture using smart cards," *J. Netw. Comput. Appl.*, vol. 35, no. 2, pp. 763–769, Mar. 2012.

[10] A. Wasef and X. Shen, "EMAP: Expedite Message Authentication Protocol for Vehicular Ad Hoc Networks," *IEEE Trans. Mob. Comput.*, vol. 12, no. 1, pp. 78–89, Jan. 2013.

[11] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, "Design and Analysis of a Lightweight Certificate Revocation Mechanism for VANET," in *Proceedings of the Sixth ACM International Workshop on VehiculAr InterNETworking*, 2009, pp. 89–98.

[12] "IEEE Trial-Use Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages," *IEEE Std 1609.2-2006*, pp. 0_1-105, 2006.

[13] IEEE, "IEEE Standard for Wireless Access in Vehicular Environments Security Services for Applications and Management Messages," 2013.

[14] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[15] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," Feb. 1997.

[16] P. Papadimitratos, A. Kung, J.-P. Hubaux, and F. Kargl, "Privacy and Identity Management for Vehicular Communication Systems: A Position Paper," in *Workshop Standards for Privacy in User- Centric Identity Management*, 2006.

[17] K. Sampigethaya, L. Huang, M. Li, R. Poovendran, K. Matsuura, and K. Sezaki, "CARAVAN : Providing Location Privacy for VANET," in *Embedded Security in Cars (ESCAR) Conf*, 2005.

[18] M. Raya and J.-P. Hubaux, "Securing Vehicular Ad Hoc Networks," *J. Comput. Secur.*, vol. 15, no. 1, pp. 39–68, 2007.

[19] Y. Sun *et al.*, "An Efficient Pseudonymous Authentication Scheme With Strong Privacy

Preservation for Vehicular Communications," *Veh. Technol. IEEE Trans.*, vol. 59, no. 7, pp. 3589–3603, 2010.

[20] R. Lu, X. Lin, T. H. Luan, X. Liang, and X. Shen, "Pseudonym Changing at Social Spots: An Effective Strategy for Location Privacy in VANETs," *Veh. Technol. IEEE Trans.*, vol. 61, pp. 86–96, 2012.

[21] J. Jeong, M. Y. Chung, and H. Choo, "Integrated OTP-Based User Authentication Scheme Using Smart Cards in Home Networks," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, 2008, pp. 294–294.

[22] Yun-kyung Lee, Hong-il Ju, Do-woo Kim, and Jong-wook Han, "Home Network Modelling and Home Network User Authentication Mechanism using Biometric Information," in *2006 IEEE International Symposium on Consumer Electronics*, 2006, pp. 1–5.

[23] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "A systematic review of data protection and privacy preservation schemes for smart grid communications," *Sustain. Cities Soc.*, vol. 38, pp. 806–835, Apr. 2018.

[24] M. A. Ferrag, L. Maglaras, A. Argyriou, D. Kosmanos, and H. Janicke, "Security for 4G and 5G cellular networks: A survey of existing authentication and privacy-preserving schemes," *J. Netw. Comput. Appl.*, vol. 101, pp. 55–82, Jan. 2018.

[25] M. A. Ferrag, L. Maglaras, A. Derhab, A. V. Vasilakos, S. Rallis, and H. Janicke, "Authentication schemes for Smart Mobile Devices: Threat Models, Countermeasures, and Open Research Issues," pp. 1–22, 2018.

[26] W. Han and Y. Xiao, "Privacy preservation for V2G networks in smart grid: A survey," *Comput. Commun.*, vol. 91–92, pp. 17–28, Oct. 2016.

[27] D. Chen *et al.*, "S2M: A Lightweight Acoustic Fingerprints based Wireless Device Authentication Protocol," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 1–1, 2016.

[28] C. Lai, H. Li, R. Lu, R. Jiang, and X. Shen, "SEGR: A secure and efficient group roaming

scheme for machine to machine communications between 3GPP and WiMAX networks," in *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 1011–1016.

[29] H. Zhu, X. Lin, Y. Zhang, and R. Lu, "Duth: a user-friendly dual-factor authentication for Android smartphone devices," *Secur. Commun. Networks*, vol. 8, no. 7, pp. 1213–1222, May 2015.

[30] Chengzhe Lai, Hui Li, Rongxing Lu, Rong Jiang, and Xuemin Shen, "LGTH: A lightweight group authentication protocol for machine-type communication in LTE networks," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 832–837.

[31] Y. Katz, Jonathan ; Lindell, *Introduction to Modern Cryptography*. 2007.

[32] M. Bellare and P. Rogaway, "The Exact Security of Digital Signatures-How to Sign with RSA and Rabin," in *Lecture Notes in Computer Science*, 1996, pp. 399–416.

[33] D. He, M. Tian, and J. Chen, "Insecurity of an efficient certificateless aggregate signature with constant pairing computations," *Inf. Sci. (Ny).*, vol. 268, pp. 458–462, Jun. 2014.

[34] E. Barker, L. Chen, A. Roginsky, and M. Smid, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography," Gaithersburg, MD, May 2013.

[35] P. Chown, "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)," Jun. 2002.

[36] J. Y. Hwang, S. Lee, B.-H. Chung, H. S. Cho, and D. Nyang, "Group signatures with controllable linkability for dynamic membership," *Inf. Sci. (Ny).*, vol. 222, pp. 761–778, Feb. 2013.

[37] J. P. Hubaux, S. Capkun, and J. Luo, "The Security and Privacy of Smart Vehicles," *Secur. Privacy, IEEE*, vol. 2, pp. 49–55, 2004.

[38] M. A. Ferrag and A. Ahmim, "ESSPR: an efficient secure routing scheme based on

searchable encryption with vehicle proxy re-encryption for vehicular peer-to-peer social network," *Telecommun. Syst.*, vol. 66, no. 3, pp. 481–503, 2017.

[39] A. Studer, E. Shi, F. Bai, and A. Perrig, "{TACKing} Together Efficient Authentication, Revocation, and Privacy in VANETs," in *Proceedings of IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON)*, 2009.

[40] C. Lai, R. Lu, and D. Zheng, "SGSA: Secure Group Setup and Anonymous Authentication in Platoon-Based Vehicular Cyber-Physical Systems," in *Wireless Algorithms, Systems, and Applications*, 2015, pp. 274–283.

[41] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux, "Eviction of Misbehaving and Faulty Nodes in Vehicular Networks," *IEEE J.Sel. A. Commun.*, vol. 25, no. 8, pp. 1557–1568, 2007.

[42] P. P. Papadimitratos, G. Mezzour, and J. Hubaux, "Certificate Revocation List Distribution in Vehicular Communication Systems," no. March 2014, pp. 12–14, 2008.

[43] K. Laberteaux, J. J. Haas, and Y.-C. Hu, "Security certificate revocation list distribution for VANET," in *Fifth ACM int'l Workshop VehiculAr Inter-NETworking*, 2008, pp. 88–89.

[44] J. J. Haas and K. P. Laberteaux, "Design and Analysis of a Lightweight Certificate Revocation Mechanism for VANET," pp. 89–98, 2009.

[45] A. Perrig and D. Song, "Random key predistribution schemes for sensor networks," in *2003 Symposium on Security and Privacy, 2003.*, 2003, pp. 197–213.

[46] L. Eschenauer and V. D. Gligor, "A Key-management Scheme for Distributed Sensor Networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, 2002, pp. 41–47.

[47] S. Zhu, S. Setia, S. Xu, and S. Jajodia, "GKMPAN: An efficient group rekeying scheme for secure multicast in ad-hoc networks," in *Journal of Computer Security - JCS*, 2004, vol. 14, pp. 42–51.

[48] S. Dolev, Ł. Krzywiecki, N. Panwar, and M. Segal, “Vehicle authentication via monolithically certified public key and attributes,” *Wirel. Networks*, vol. 22, no. 3, pp. 879–896, Apr. 2016.

[49] H. Krawczyk, “SIGMA: The `SIGn-and-MAc’ Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols,” in *Advances in Cryptology - CRYPTO 2003*, 2003, pp. 400–425.

[50] ISO/IEC, “ISO/IEC IS 9798-3, ‘Entity authentication mechanisms -- Part 3: Entity authentication using asymmetric techniques,’” 1993.

[51] S. Cespedes, S. Taha, and X. Shen, “A Multihop-Authenticated Proxy Mobile IP Scheme for Asymmetric VANETs,” *IEEE Trans. Veh. Technol.*, vol. 62, no. 7, pp. 3271–3286, Sep. 2013.

[52] R. Baldessari, A. Festag, W. Zhang, and L. Le, “A MANET-centric Solution for the Application of NEMO in VANET Using Geographic Routing,” in *Proceedings of the 4th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*, 2008, no. ACM 978-1-60558-009-8, p. 7.

[53] C. Lyu, D. Gu, Y. Zeng, and P. Mohapatra, “PBA: Prediction-Based Authentication for Vehicle-to-Vehicle Communications,” *IEEE Trans. Dependable Secur. Comput.*, vol. 13, no. 1, pp. 71–83, Jan. 2016.

[54] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen, “An Efficient Merkle-Tree-Based Authentication Scheme for Smart Grid,” *IEEE Syst. J.*, vol. 8, no. 2, pp. 655–663, Jun. 2014.

[55] A. C.-F. Chan and J. Zhou, “Cyber–Physical Device Authentication for the Smart Grid Electric Vehicle Ecosystem,” *IEEE J. Sel. Areas Commun.*, vol. 32, no. 7, pp. 1509–1517, Jul. 2014.

[56] L. Zhang, C. Hu, Q. Wu, J. Domingo-Ferrer, and B. Qin, “Privacy-Preserving Vehicular Communication Authentication with Hierarchical Aggregation and Fast Response,” *IEEE*

Trans. Comput., vol. 65, no. 8, pp. 2562–2574, Aug. 2016.

[57] J. Shao, X. Lin, R. Lu, and C. Zuo, “A Threshold Anonymous Authentication Protocol for VANETs,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 3, pp. 1711–1720, Mar. 2016.

[58] D. Boneh and M. K. Franklin, “Identity-Based Encryption from the Weil Pairing,” in *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, 2001, pp. 213–229.

[59] L. Lamport, “Password Authentication with Insecure Communication,” *Commun. ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[60] D. Eastlake and P. Jones, “US Secure Hash Algorithm 1 (SHA1),” Sep. 2001.

[61] N. Koblitz, A. Menezes, and S. Vanstone, “The State of Elliptic Curve Cryptography,” *Des. Codes Cryptogr.*, vol. 19, no. 2, pp. 173–193, Mar. 2000.

[62] Thomas H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, Massachusetts London, England: MIT, 2001.

[63] T. Schmidt, M. Waehlisch, and S. Krishnan, “Base Deployment for Multicast Listener Support in Proxy Mobile IPv6 (PMIPv6) Domains,” Apr. 2011.

[64] R. Blom, “An Optimal Class of Symmetric Key Generation Systems,” in *Advances in Cryptology*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 335–338.

[65] D. Chaum and E. van Heyst, “Group Signatures,” no. 1, 1991, pp. 257–265.

[66] D. Boneh, X. Boyen, and H. Shacham, “Short Group Signatures,” in *Lecture Notes in Computer Science*, vol. 3152, 2004, pp. 41–55.

[67] R. C. Merkle, “Secrecy, Authentication, and Public Key Systems,” 1979.

[68] A. Perrig, R. Canetti, D. Xiaodong Song, and J. D. Tygar, “Efficient and Secure Source Authentication for Multicast,” in *NDSS*, 2001.

[69] E. Kiltz and K. Pietrzak, “Leakage Resilient ElGamal Encryption,” in *Proceedings of the*

4th ACM international symposium on Mobile ad hoc networking & computing - MobiHoc '03, New York, New York, USA: ACM Press, 2010, pp. 595–612.

[70] D. Boneh and H. Shacham, "Group signatures with verifier-local revocation," in *Proceedings of the 11th ACM conference on Computer and communications security - CCS '04*, 2004, p. 168.

[71] D. Bleichenbacher and A. May, "New Attacks on RSA with Small Secret CRT-Exponents," 2006, pp. 1–13.

[72] D. Pointcheval and J. Stern, "Security Arguments for Digital Signatures and Blind Signatures," *J. Cryptol.*, vol. 13, no. 3, pp. 361–396, Jun. 2000.

[73] Bing Li, Zhijie Wang, and Dijiang Huang, "An Efficient and Anonymous Attribute-Based group setup scheme," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 861–866.

[74] H. Nicanfar and V. C. M. Leung, "EIBC: Enhanced Identity-Based Cryptography, a conceptual design," in *2012 IEEE International Systems Conference SysCon 2012*, 2012, pp. 1–7.

[75] L. Sustek, "Hardware Security Module," in *Encyclopedia of Cryptography and Security*, H. C. A. van Tilborg and S. Jajodia, Eds. Boston, MA: Springer US, 2011, pp. 535–538.

[76] Certicom Research, "Standards for efficient cryptography, SEC 1: Elliptic Curve Cryptography," *Stand. Effic. Cryptogr.*, vol. 1, no. Sec 1, pp. 1–22, 2009.

[77] C. Research, "Standards for efficient cryprography - SEC 2 : Recommended Elliptic Curve Domain Parameters," 2000.

[78] M. G. Mediawiki and A. Name, "Crypto++ ® Library 8.1," pp. 1–7, 2019.

[79] M. K. Saparbaev, a V Mazin, L. P. Ovchinnikova, G. L. Dianov, and R. I. Salganik, *Introduction to Algorithms Third Edition*, no. 2. 1988.

[80] M. Dworkin, "NIST Policy on Hash Functions," 2015. [Online]. Available:

https://csrc.nist.gov/projects/hash-functions/nist-policy-on-hash-functions.

[81] M. J. Dworkin, “Recommendation for block cipher modes of operation :,” Gaithersburg, MD, 2007.

[82] J. M. Turner, “The keyed-hash message authentication code (hmac),” *Fed. Inf. Process. Stand. Publ.*, no. July, 2008.

[83] J. Katz and A. Y. Lindell, “Aggregate Message Authentication Codes,” in *Topics in Cryptology – CT-RSA 2008*, vol. 4964 LNCS, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 155–169.

[84] D. M’Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen, “HOTP: An HMAC-Based One-Time Password Algorithm,” Dec. 2005.

[85] U. Coruh and O. Bayat, “Hybrid Secure Authentication and Key Exchange Scheme for M2M Home Networks,” *Secur. Commun. Networks*, vol. 2018, pp. 1–25, Nov. 2018.

[86] J. Nielsen, *Usability Engineering*. 1993.

[87] R. B. Miller, “Response time in man-computer conversational transactions,” in *Proceedings of the December 9-11, 1968, fall joint computer conference, part I on - AFIPS ’68 (Fall, part I)*, 1968, p. 267.

[88] S. K. Card, G. G. Robertson, and J. D. Mackinlay, “The Information Visualizer, an Information Workspace,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1991, pp. 181–186.