A Method for Achieving Confidentiality and Integrity in IEC 61850 GOOSE Messages

S. M. Suhail Hussain, *Member, IEEE*, Shaik Mullapathi Farooq, *Member, IEEE*, and Taha Selim Ustun, *Member, IEEE*

***Abstract*—IEC 62351-6 standard stipulates the use of digital signatures for ensuring integrity and authenticity in IEC 61850 Generic Object-Oriented Substation Events (GOOSE) message exchanges. Yet, it does not specify any method for ensuring confidentiality in GOOSE messages. With rapid growth of IEC 61850 from substation automation to power management domain, sensitive data that requires privacy is carried over GOOSE messages. In this letter, a novel method employing Authenticated Encryption with Associated Data (AEAD) algorithms is proposed to achieve both confidentiality and integrity of GOOSE messages. Further, lab tests are run to observe the timing performance and applicability of these algorithms to GOOSE messages with strict timing requirements.**

***Index Terms*—IEC 61850, GOOSE, IEC 62351, AEAD.**

I. Introduction

IEC 62351-6 [1] standard stipulates security requirements for IEC 61850 based communication messages. For multicast messages such as Generic Object-Oriented Substation Events (GOOSE) and Sampled Value (SV), it recommends only the message integrity and authenticity checks that use digital signatures with Rivest–Shamir–Adleman (RSA) algorithms. From previous studies, it has been observed that the generation of digital signatures with RSA algorithms requires high computational times in the order of few milli seconds; and use of Elliptic Curve Digital Signature Algorithm (ECDSA) based signatures is proposed as a solution [2], [3]. Despite having a comparatively better performance, ECDSA algorithms still require long processing times when applied to GOOSE messages [3]. Hence, in [4] authors proposed different Message Authentication Code (MAC) algorithms for message integrity and authenticity checks in GOOSE messages, which resulted in much lower processing times.

Considering that GOOSE messages have a strict real time delivery requirement of 3 ms, IEC 62351-1 stipulates that encryption algorithms should not be applied [5]. The reasoning is that processing times for encryption of GOOSE messages would be high with the limited computation capacity of the Intelligent Electronic Device (IED). With the extension of IEC 61850's use beyond SAS to power utility automation such as Distributed Energy Resources (DERs) [6], virtual power plants [7], electric vehicles [8], smart grid automation [9]; sensitive data is carried over IEC 61850 messages. Generally, in Substation Automation Systems (SAS), GOOSE messages are used to carry the breaker trip or close commands. The confidentiality of these messages inside a substation environment is not a strict requirement. However, when GOOSE messages are used to send commands to different DERs for energy management or market purposes, their confidentiality becomes very important. Hence, GOOSE messages in these new scenarios must be encrypted to achieve the confidentiality requirement. Also, the encryption algorithms must adhere to stringent GOOSE timing requirements of 3 ms.

However, till now the security mechanisms reported in the literature for GOOSE messages do not consider confidentiality requirement [2]–[4]. In this letter, to address the above-mentioned gap, a novel method is proposed to ensure confidentiality and message authentication of GOOSE messages by employing Authenticated Encryption with Associated Data (AEAD) algorithms. Further, C-library based implementations are developed in the lab to test the timing performance and feasibility of the proposed security method to GOOSE messages.

TABLE I
DIFFERENT HMAC VARIANTS PROPOSED BY THE STANDARD

S.No	MAC Algorithm	Hash Function	*MAC* value (Size in bytes)
1	HMAC-SHA256-80	SHA-256	10
2	HMAC-SHA256-128	SHA-256	16
3	HMAC-SHA256-256	SHA-256	32
4	AES-GMAC-64	-	8
5	AES-GMAC-128	-	16

II. Proposed Method for Achieving Confidentiality in GOOSE Messages

In the proposed method, three variants of the AEAD algorithms are employed to achieve confidentiality along with message integrity and authenticity in GOOSE messages. These are Encrypt-then-MAC (EtM), Encrypt-and-MAC (E&M) and MAC-then-Encrypt (MtE).

In EtM, the GOOSE Application Protocol Data Unit (APDU) is first encrypted using algorithms such as AES 128 or AES 256. Then, *MAC value* of the encrypted GOOSE APDU is generated by one of the MAC algorithms listed in Table I. This *MAC value* is appended to the GOOSE Protocol Data Unit (PDU) in the "Extension" field as depicted in Fig. 1. As this increases the length of the GOOSE frame, the difference is reflected in the 2nd byte of "Reserved1"

Manuscript received June 26, 2019; revised December 11, 2019 and April 16, 2020; accepted April 24, 2020. Date of publication April 27, 2020; date of current version September 23, 2020. Paper no. PESL-00146-2019. *(Corresponding author: S. M. Suhail Hussain.)*

S. M. Suhail Hussain and Taha Selim Ustun are with the Fukushima Renewable Energy Institute, AIST (FREA), National Institute of Advanced Industrial Science and Technology (AIST), Koriyama 963-0298, Japan (e-mail: suhail.hussain@aist.go.jp; selim.ustun@aist.go.jp).

Shaik Mullapathi Farooq is with the Department of Computer Science and Engineering, YSR College of Engineering,Yogi Vemana University, Kadapa 516005, India (e-mail: smfarooq@ieee.org).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TPWRD.2020.2990760

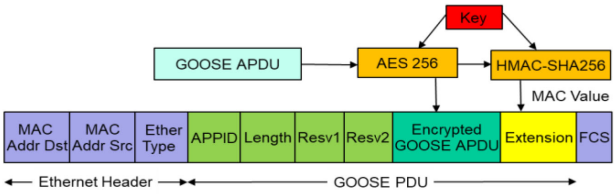

Fig. 1. Encrypt-then-MAC (EtM) algorithm applied to GOOSE PDU.

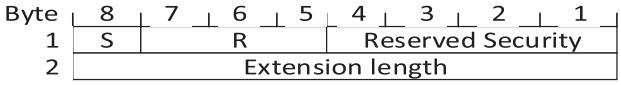

Fig. 2. Structure of Reserved1 field.

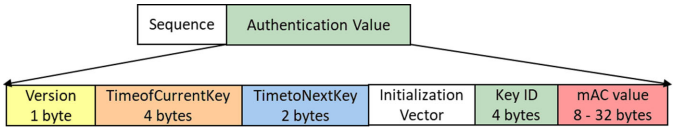

Fig. 3. Structure of Extension field.

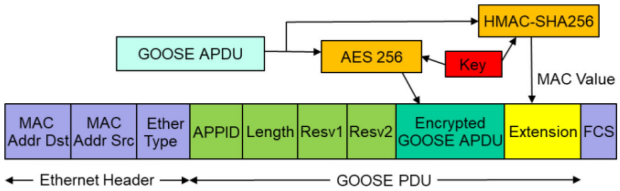

Fig. 4. Encrypt-and-MAC (E&M) algorithm applied to GOOSE PDU.

field, as shown in Fig. 2. The value of 2nd byte "Reserved 1" can be between 0 and 255 which represents the length of Extension field. When its value is 0 there is no extension field, indicating no security is applied to GOOSE messages. The structure of the extension field appended to the GOOSE PDU is shown in the Fig. 3.

At the receiver side; first, MAC value of received encrypted GOOSE APDU is calculated. This value is compared with the appended MAC value. If they match, GOOSE packet is authentic and decrypted; otherwise, it is discarded.

In E&M, encryption and *MAC* generation is performed on the original GOOSE APDU. The encrypted APDU and *MAC value* are added to "APDU" and "Extension" fields of GOOSE PDU, respectively. At the receiver; first, the encrypted APDU is decrypted and, then, MAC value for it is calculated. The calculated MAC value is compared with the received MAC value. If they match, GOOSE packet is authentic; otherwise, it is discarded. Fig. 4 shows the operation of E&M AEAD algorithm.

As the name implies, in MtE variant, first, MAC value for the GOOSE APDU is calculated. Plain APDU and calculated MAC value are added to the APDU and Extension fields of the GOOSE PDU format. Then, GOOSE PDU is encrypted and sent to the receiver. Receiver decrypts GOOSE PDU to obtain sent MAC value. It also calculates MAC value of the encrypted GOOSE APDU. Both the received MAC value and calculated MAC values are compared to detect any tampering of the GOOSE message. Fig. 5 depicts the operation of MtE AEAD.

In this method, the "R" field of 3 bits in "Reserved1" field of GOOSE message, as shown in Fig. 2, is utilized to identify the variant of AEAD encryption employed. If value of "R" field is

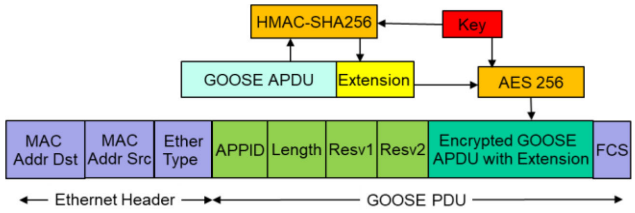

Fig. 5. MAC-then-Encrypt (MtE) algorithm applied to GOOSE PDU.

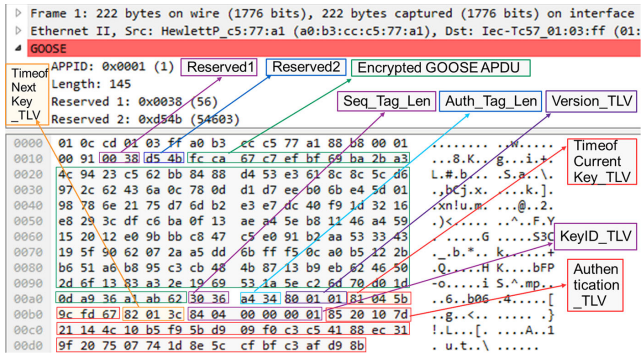

Fig. 6. Packet capture of encrypted and authenticated GOOSE message.

1, 2, 3 it corresponds to EtM, E&M, MtE respectively, when no encryption is employed its value is 0.

III. Implementation and Performance Evaluations

Proposed method employing AEAD algorithms for GOOSE security are implemented in the lab. Experiments are run to observe their performances and to investigate their feasibility for securing GOOSE messages. Authors have developed GoSV library that has both publishers and subscribers for GOOSE and SV messages. GoSV enables conducting experiments with custom messages. Secure GoSV (S-GoSV) library is developed by integrating AEAD algorithms discussed in previous section [10]. It secures GOOSE message communication by employing AES-128 encryption and HMAC SHA-256 message integrity. S-GoSV publisher and subscriber libraries are run in two separate terminals which act as two IEDs, e.g., protection device and circuit breaker. Secure GOOSE message generation algorithm, Gen_EtM(),is initiated at the publisher terminal, which generates a secure GOOSE message containing encrypted GOOSE APDU (E_d) and *MAC value "h"* using the symmetric *PreSharedKey k*. Wireshark capture of encrypted and authenticated GOOSE message published by EtM AEAD algorithm of S-GoSV library is as shown in Fig. 6.

The subscriber receives the secure GOOSE message and reads the encrypted APDU (E_d) values into ReceivedData buffer. Algorithm *verify_EtM* is utilized to check the integrity of GOOSE message. Firstly, a new *MAC value "h1"* is computed for the received encrypted GOOSE APDU (E_d) using the symmetric *PreSharedKey k*. *"h1"* is compared with the received *MAC value "h"*. If they match, then the received GOOSE PDU is accepted as a legitimate packet; otherwise, it is discarded. For the accepted packet the encrypted GOOSE APDU (E_d) is decrypted using the key *k* to get the APDU data. Likewise, secure GOOSE publisher and subscriber algorithms for E&M and MtE AEAD algorithms are developed as part of the S-GoSV library.

To observe timing performance and investigate their feasibility in time-restricted systems, the computational times for running all

TABLE II
COMPUTATIONAL TIMES FOR DIFFERENT AEAD ALGORITHMS (IN MS)

AEAD Algorithm	Publisher						Subscriber						Total time for security	
	AES 128 (Encrypt.)			HMAC SHA256			AES 128 (Decrypt.)			HMAC SHA256				
	Average	Stdev	Max.	Average	Stdev	Max.	Average	Stdev	Max.	Average	Stdev	Max.	Average	Max.
EtM	0.0797	0.0052	0.1	0.0112	0.0013	0.013	0.0932	0.0141	0.114	0.0115	0.0028	0.015	0.1956	0.242
E&M	0.0802	0.0028	0.1	0.0101	0.0004	0.011	0.0932	0.0155	0.113	0.0104	0.0005	0.011	0.1939	0.235
MtE	0.1131	0.0095	0.133	0.0103	0.0007	0.012	0.1156	0.0171	0.128	0.0104	0.0003	0.011	0.2494	0.284

Algorithm Gen_EtM( )	**Algorithm verify_EtM()**
1: goosePDU ← *GoSV()*	1: *ReceivedData* ← *goosePDU.APDU* = E_d
2: *InputData* ← *goosePDU.APDU*	2: h ← *goosePDU.extension*
3: k ← *PreSharedKey()*	3: k ← *PreSharedKey()*
4: E_d ← $Encrypt_K$ *(InputData)*	4: $h1$ ← MAC_k *(ReceivedData)*
5: h ← MAC_k (E_d)	5: if h =$h1$ *then*
6: *goosePDU.Extension* ← h	6: *APDU* ← $Decrypt_K$ (E_d)
7: *goosePDU.APDU* ← E_d	7: *else*
	8: *return "Reject GOOSE packet"*

TABLE III
END TO END DELAY FOR SECURE GOOSE MESSAGES (IN MS)

AEAD Algorithm	Message size (bytes)	Security processing delay	Communication delay		End to End delay	
			Normal	Worst-case	Normal	Worst-case
No Security	159	0	0.0664	1.330	0.0664	1.330
EtM	222	0.1956	0.0779	1.440	0.2735	1.6356
E&M	222	0.1939	0.0779	1.440	0.2718	1.6339
MtE	230	0.2494	0.0791	1.650	0.3285	1.8994

these three security variants are calculated. The computational times are calculated by sampling the CPU times at the start and end of codes in C program using the clock() functions of sys/time.h header files. The test system is an Intel(R) Celeron (R) with 4GB RAM runnig Ubuntu 18.04 LTS operating system and S-GoSV library with GCC compiler. This slow system is intentionally selected. If this system can meet timing requirements, then current IEDs should not face any difficulties as they have much higher computing power (Intel Core i7-3555LE with 8 GB RAM) [11].

Table II shows the computational times for different AEAD variants. It can be observed that the total computational time required by GOOSE publisher and subscriber for EtM AEAD algorithm is 0.196 ms. Similarly, E&M and MtE AEAD algorithms have average delays of 0.1939 and 0.2494 ms respectively. Further, the maximum processing delay for security algorithms found to be 0.242, 0.235 and 0.284 ms for EtM, E&M and MtE algorithms respectively. Table III shows the size of secure GOOSE messages for different AEAD algorithms obtained from S-GoSV library implementations. Based on the size of secure GOOSE messages, communication delays are calculated by simulating a typical type D2-1 substation communication network (SCN) with 100 mbps links using NetSim network simulator tool [12]. A worst-case scenario where all the IEDs in SCN are generating messages at same time is also considered. Table III gives the end to end delays, includes both processing and communication delays, for communicating secure GOOSE messages both under normal and worst-case scenarios are well below the 3 ms limit. This shows that the proposed AEAD schemes can be efficiently applied to the GOOSE messages in real time applications.

Among the three AEAD variants that are tested, EtM has more advantages compared to other two. Since the MAC value is calculated for the encrypted data, if any tampering is detected the packet is dropped before applying decrypting algorithms. This saves valuable time in EtM, since in E&M and MtE, decryption must be performed to check message integrity.

IV. CONCLUSION

In this letter, AEAD algorithms for ensuing confidentiality and message integrity simultaneously for GOOSE is proposed. Three different AEAD variants are implemented with AES 128 encryption for confidentiality. For message authentication, MAC algorithms are used. The lab experiments conclude that the AEAD algorithms can be successfully applied to the GOOSE messages while meeting strict 3 ms delay requirements. Based on these findings, cybersecurity frameworks of future IEC 62351 standard can safely recommend encryption for GOOSE messages. This is very much required as novel implementations of IEC 61850 are related to sensitive ownership and finance data.

REFERENCES

[1] *Power Systems Management and Associated Information Exchange-Data and Communications Security, Part 6: Security for IEC 61850*, IEC/TS 62351-6:2007(E), 2007.

[2] S. M. Farooq, S. M. S. Hussain, and T. S. Ustun, "Performance evaluation and analysis of IEC 62351-6 probabilistic signature scheme for securing GOOSE messages," *IEEE Access*, vol. 7, pp. 32343–32351, 2019.

[3] T. T. Tesfay and J. Le Boudec, "Experimental comparison of multicast authentication for wide area monitoring systems," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4394–4404, Sep. 2018.

[4] S. M. S. Hussain, S. M. Farooq, and T. S. Ustun, "Analysis and implementation of message authentication code (MAC) algorithms for GOOSE message security," *IEEE Access*, vol. 7, pp. 80980–80984, 2019.

[5] *Power Systems Management and Associated Information Exchange-Data and Communications Security, Part 1: Introduction to Security Issues*, IEC/TS 62351-1, 2007.

[6] I. Ali and S. M. S. Hussain, "Communication design for energy management automation in microgrid," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 2055–2064, May 2018.

[7] N. Etherden, V. Vyatkin, and M. H. J. Bollen, "Virtual power plant for grid services using IEC 61850," *IEEE Trans. Ind. Informat.*, vol. 12, no. 1, pp. 437–447, Feb. 2016.

[8] S. M. S. Hussain, T. S. Ustun, P. Nsonga, and I. Ali, "IEEE 1609 WAVE and IEC 61850 standard communication based integrated EV charging management in smart grids," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7690–7697, Aug. 2018.

[9] G. Zhabelova, V. Vyatkin, and V. N. Dubinin, "Toward industrially usable agent technology for smart grid automation," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2629–2641, Apr. 2015.

[10] 'S-GoSV', 2019. [Online]. Available: https://github.com/61850security/S-GoSV-part-2, Accessed on: May 11, 2020.

[11] Data Sheet -SEL 3555 Real Time Automation Controller (RTAC), 2020. [Online]. Available: https://goo.gl/jjnfnV, Accessed on: May 11, 2020.

[12] NetSim – Tetcos, 2019. [Online]. Available: https://www.tetcos.com/