Document Title	Specification of Secure Onboard Communication Protocol
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	969

Document Status	published
Part of AUTOSAR Standard	Foundation
Part of Standard Release	R23-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2023-11-23	R23-11	AUTOSAR Release Management	• Removal of implementation specific contents • Editorial changes
2022-11-24	R22-11	AUTOSAR Release Management	• Removal of implementation specific contents • Update of configuration parameters • Editorial changes
2021-11-25	R21-11	AUTOSAR Release Management	• no content changes
2020-11-30	R20-11	AUTOSAR Release Management	• Initial release

Contents

1 Introduction and overview

Authentication and integrity protection of sensitive data is necessary to protect correct and safe functionality of the vehicle systems - this ensures that received data comes from the right ECU and has the correct value.

The SecOC protocol as described in this document provides a mechanism to verify the authenticity and freshness of PDU based communication between ECUs within the vehicle architecture. The approach requires both the sending ECU and the receiving ECU to implement a SecOC module.

To provide message freshness, the SecOC module on the sending and receiving side get freshness from an external Freshness Manager for each uniquely identifiable Secured I-PDU, i.e. for each secured communication link.

On the sender side, the SecOC module creates a Secured I-PDU by adding authentication information to the outgoing Authentic I-PDU. The authentication information comprises of an Authenticator (e.g. Message Authentication Code) and optionally a Freshness Value. Regardless if the Freshness Value is or is not included in the Secure I-PDU payload, the Freshness Value is considered during generation of the Authenticator. When using a Freshness Counter instead of a Timestamp, the Freshness Counter should be incremented by the Freshness Manager prior to providing the authentication information to the receiver side. On the receiver side, the SecOC module checks the freshness and authenticity of the Authentic I-PDU by verifying the authentication information that has been appended by the sending side SecOC module. To verify the authenticity and freshness of an Authentic I-PDU, the Secured I-PDU provided to the receiving side SecOC should be the same Secured I-PDU provided by the sending side SecOC and the receiving side SecOC should have knowledge of the Freshness Value used by the sending side SecOC during creation of the Authenticator.

1.1 Protocol purpose and objectives

The SecOC protocol aims for resource-efficient and appropriate authentication mechanisms for critical data on the level of PDUs. The authentication mechanisms shall be seamlessly integrated with the current AUTOSAR communication systems. The impact with respect to resource consumption should be as small as possible in order to allow protection as add-on for legacy systems. The specification is based on the assumption that mainly symmetric authentication approaches with message authentication codes (MACs) are used. They achieve the same level of security with much smaller keys than asymmetric approaches and can be implemented compactly and efficiently in software and in hardware. However, the specification provides the necessary level of abstraction so that both, symmetric approaches as well as asymmetric authentication approaches can be used.

1.2 Applicability of the protocol

The SecOC protocol is used in all ECUs where secure communication is necessary.

1.2.1 Constraints and assumptions

1.2.1.1 Adaptation in case of asymmetric approach

Although this document consequently uses the terms and concepts from symmetric cryptography, the SecOC protocol supports both symmetric and asymmetric cryptographic algorithms. In case of an asymmetric approach using digital signatures instead of the MAC-approach described throughout the whole document, some adaptations must be made:

1. Instead of a shared secret between sender and (all) receivers, a key pair consisting of public key and secret key is used. The secret (or private) key is used by the sender to generate the signature, the corresponding public keys is used by (all) receiver(s) to verify the signature. The private key must not be feasibly computable from the public key and it shall not be assessable by the receivers.
2. In order to verify a message, the receiver needs access to the complete signature / output of the signature generation algorithm. Therefore, a truncation of the signature as proposed in the MAC case is NOT possible. The parameter SecOCAuthInfoTruncLength has to be set to the complete length of the signature.
3. The signature verification uses a different algorithm then the signature generation. So instead of "rebuilding" the MAC on receiver side and comparing it with the received (truncated) MAC as given above, the receiver / verifier performs the verification algorithm using the DataToAuthenticator (including full counter) and the signature as inputs and getting a Boolean value as output, determining whether the verification passed or failed.

1.2.2 Limitations

The protocol specification aims to ensure compatibility between AP and CP, and it assumes the communication is realized over ethernet.

Depending of the communication paradigm between AP and CP, the functionality of the protocol is limited. In the case of SOME/IP, the protocol will not support separate transmission of Authentic PDU and Cryptographic PDU and will not support usage of part of the payload as freshness information. (the details are described in the chapter Configuration Parameters.)

1.3 Dependencies

1.3.1 Dependencies to other protocol layers

The interaction of SecOC with the lower layer of the communication stack will depend on the on the platform architecture (AP or CP), and in the case of a CP implementation, it will also depend on the type of transmission: direct transmission, triggered transmission or transport protocol. These design specific dependencies are not part of the protocol specification.

1.3.2 Dependencies to other standards and norms

[1] *IEC 7498-1 The Basic Model, IEC Norm, 1994*

[2] *National Institute of Standards and Technology (NIST): FIPS-180-4, Secure Hash Standard (SHS), March 2012*, available electronically at http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

[3] *FIPS Pub 197: Advanced Encryption Standard (AES), U.S. Department of Commerce, Information Technology Laboratory (ITL), National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, Federal Information Processing Standards Publication, 2001*, electronically available at http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

1.3.3 Dependencies to the Application Layer

The SecOC protocol does not have dependencies to typical Automotive application. However, it relies on the existence of a software component that provides a freshness information. In addition, there could also be specialized applications that trigger a modification in SecOC behavior (e.g. for development purpose) or applications that monitor the verification results.

2 Use Cases

ID	Name	Description
UC_001	SecOC SOME/IP	Secure communication between AP and CP using SOME/IP
UC_002	SecOC SignalBased	Secure communication between AP and CP using signal-based communication and SignalToService translation

3 Protocol Requirements

3.1 Requirements Traceability

Requirement	Description	Satisfied by
[RS_Main_00510]	Secure Onboard Communication	[PRS_SecOc_00101] [PRS_SecOc_00102] [PRS_SecOc_00103] [PRS_SecOc_00104] [PRS_SecOc_00105] [PRS_SecOc_00200] [PRS_SecOc_00206] [PRS_SecOc_00208] [PRS_SecOc_00210] [PRS_SecOc_00211] [PRS_SecOc_00213] [PRS_SecOc_00215] [PRS_SecOc_00216] [PRS_SecOc_00220] [PRS_SecOc_00221] [PRS_SecOc_00223] [PRS_SecOc_00300] [PRS_SecOc_00306] [PRS_SecOc_00309] [PRS_SecOc_00313] [PRS_SecOc_00314] [PRS_SecOc_00315] [PRS_SecOc_00316] [PRS_SecOc_00317] [PRS_SecOc_00320] [PRS_SecOc_00341] [PRS_SecOc_00342] [PRS_SecOc_00600] [PRS_SecOc_00610] [PRS_SecOc_00620] [PRS_SecOc_00630]
[SRS_BSW_00337]	Classification of development errors	[PRS_SecOc_00500]
[SRS_BSW_00350]	All AUTOSAR Basic Software Modules shall allow the enabling/ disabling of detection and reporting of development errors.	[PRS_SecOc_00500]
[SRS_BSW_00385]	List possible error notifications	[PRS_SecOc_00330] [PRS_SecOc_00340] [PRS_SecOc_00341] [PRS_SecOc_00500]
[SRS_BSW_00450]	A Main function of a un-initialized module shall return immediately	[PRS_SecOc_00500]

Table 3.1: RequirementsTracing

4 Definition of terms and acronyms

4.1 Acronyms and abbreviations

Abbreviation / Acronym:	Description:
SecOC	Secure Onboard Communication
MAC	Message Authentication Code
FV	Freshness Value
FM	Freshness Manager

4.2 Definition of terms

Terms:	Description:
Authentic I-PDU	An Authentic I-PDU is an arbitrary AUTOSAR I-PDU the content of which is secured during network transmission by means of the Secured I-PDU. The secured content comprises the complete I-PDU or a part of the I-PDU.
Authentication	Authentication is a service related to identification. This function applies to both entities and information itself. Two parties entering into a communication should identify each other. Information delivered over a channel should be authenticated as to origin, date of origin, data content, time sent, etc. For these reasons, this aspect of cryptography is usually subdivided into two major classes: entity authentication and data origin authentication. Data origin authentication implicitly provides data integrity (for if a message is modified, the source has changed).
Authentication Information	The Authentication Information consists of a Freshness Value (or a part thereof) and an Authenticator (or a part thereof). Authentication Information are the additional pieces of information that are added by SecOC to realize the Secured I-PDU.
Authenticator	Authenticator is data that is used to provide message authentication. In general, the term Message Authentication Code (MAC) is used for symmetric approaches while the term Signature or Digital Signature refers to asymmetric approaches having different properties and constraints.
Data integrity	Data integrity is the property whereby data has not been altered in an unauthorized manner since the time it was created, transmitted, or stored by an authorized source. To assure data integrity, one should have the ability to detect data manipulation by unauthorized parties. Data manipulation includes such things as insertion, deletion, and substitution.
Data origin authentication	Data origin authentication is a type of authentication whereby a party is corroborated as the (original) source of specified data created at some (typically unspecified) time in the past. By definition, data origin authentication includes data integrity.

Terms:	Description:
Distinction unilateral / bilateral authentication	In unilateral authentication, one side proves identity. The requesting side is not even authenticated to the extent of proving that it is allowed to request authentication. In bilateral authentication, the requester is also authenticated at least (see below) to prove the privilege of requesting. There is an efficient and more secure way to authenticate both endpoints, based on the bilateral authentication described above. Along with the authentication (in the second message) requested initially by the receiver (in the first message), the sender also requests an authentication. The receiver sends a third message providing the authentication requested by the sender. This is only three messages (in contrast to four with two unilateral messages).
Entity authentication	Entity authentication is the process whereby one party is assured (through acquisition of corroborative evidence) of the identity of a second party involved in a protocol, and that the second has actually participated (i.e., is active at, or immediately prior to, the time the evidence is acquired). **Note:** Entity authentication means to prove presence and operational readiness of a communication endpoint. This is for example often done by proving access to a cryptographic key and knowledge of a secret. It is necessary to do this without disclosing either key or secret. Entity authentication can be used to prevent record-and-replay attacks. Freshness of messages only complicates them by the need to record a lifetime and corrupt either senders or receivers (real-time) clock. Entity authentication is triggered by the receiver, i.e. the one to be convinced, while the sender has to react by convincing. Record and replay attacks on entity authentication are usually prevented by allowing the receiver some control over the authentication process. In order to prevent the receiver from using this control for steering the sender to malicious purposes or from determining a key or a secret ("oracle attack"), the sender can add more randomness. If not only access to a key (implying membership to a privileged group) but also individuality is to be proven, the sender additionally adds and authenticates its unique identification.
Message authentication	Message authentication is a term used analogously with data origin authentication. It provides data origin authentication with respect to the original message source (and data integrity, but no uniqueness and timeliness guarantees).
Secured I-PDU	A Secured I-PDU is an AUTOSAR I-PDU that contains Payload of an Authentic I-PDU supplemented by additional Authentication Information.
Transaction authentication	Transaction authentication denotes message authentication augmented to additionally provide uniqueness and timeliness guarantees on data (thus preventing undetectable message replay).

5 Protocol specification

5.1 Specification of the security solution

The SecOC protocol as described in this document provides a mechanism to verify the authenticity and freshness of PDU based communication between ECUs within the vehicle architecture. The approach requires both the sending ECU and the receiving ECU to implement a SecOC module.

To provide message freshness, the SecOC module on the sending and receiving side get freshness from an external Freshness Manager for each uniquely identifiable Secured I-PDU, i.e. for each secured communication link.

On the sender side, the SecOC module creates a Secured I-PDU by adding Authentication Information to the outgoing Authentic I-PDU. The Authentication Information comprises of an Authenticator (e.g. Message Authentication Code) and optionally a Freshness Value. Regardless if the Freshness Value is or is not included in the Secure I-PDU payload, the Freshness Value is considered during generation of the Authenticator. When using a Freshness Counter instead of a Timestamp, the Freshness Counter should be incremented by the Freshness Manager prior to providing the Authentication Information to the receiver side.

On the receiver side, the SecOC module checks the freshness and authenticity of the Authentic I-PDU by verifying the Authentication Information that has been appended by the sending side SecOC module. To verify the authenticity and freshness of an Authentic I-PDU, the Secured I-PDU provided to the receiving side SecOC should be the same Secured I-PDU provided by the sending side SecOC and the receiving side SecOC should have knowledge of the Freshness Value used by the sending side SecOC during creation of the Authenticator.

5.1.1 Basic entities of the security solution

The term Authentic I-PDU refers to an AUTOSAR I-PDU that requires protection against unauthorized manipulation and replay attacks.

The payload of a Secured I-PDU consists of the Authentic I-PDU and an Authenticator (e.g. Message Authentication Code). The payload of a Secured I-PDU may optionally include the Freshness Value used to create the Authenticator (e.g. MAC). The order in which the contents are structured in the Secured I-PDU is compliant with Figure 5.1.

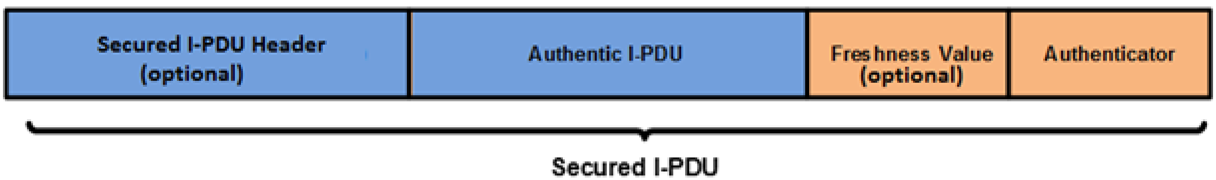

Figure 5.1: Secured I-PDU contents

The length of the Authentic I-PDU, the Freshness Value and the Authenticator within a Secured I-PDU may vary from one uniquely indefinable Secured I-PDU to another.

The Authenticator (e.g. MAC) refers to a unique authentication data string generated using a Key, Data Identifier of the Secured I-PDU, Authentic Payload, and Freshness Value. The Authenticator provides a high level of confidence that the data in an Authentic I-PDU is generated by a legitimate source and is provided to the receiving ECU at the time in which it is intended for.

Depending on the authentication algorithm used to generate the Authenticator, it may be possible to truncate the resulting Authenticator (e.g. in case of a MAC) generated by the authentication algorithm. Truncation may be desired when the message payload is limited in length and does not have sufficient space to include the full Authenticator.

The Authenticator length contained in a Secured I-PDU (parameter SecOCAuthInfoTruncLength) is specific to a uniquely identifiable Secured I-PDU. This allows provision of flexibility across the system (i.e. two independent unique Secured I-PDUs may have different Authenticator lengths included in the payload of the Secured I-PDU) by providing fine grain configuration of the MAC truncation length for each Secured I-PDU.

If truncation is possible, the Authenticator should only be truncated down to the most significant bits of the resulting Authenticator generated by the authentication algorithm. Figure 5.2 shows an example of the truncation of the Authenticator and the Freshness Values respecting the parameter SecOCFreshnessValueTruncLength and SecOCAuthInfoTruncLength.

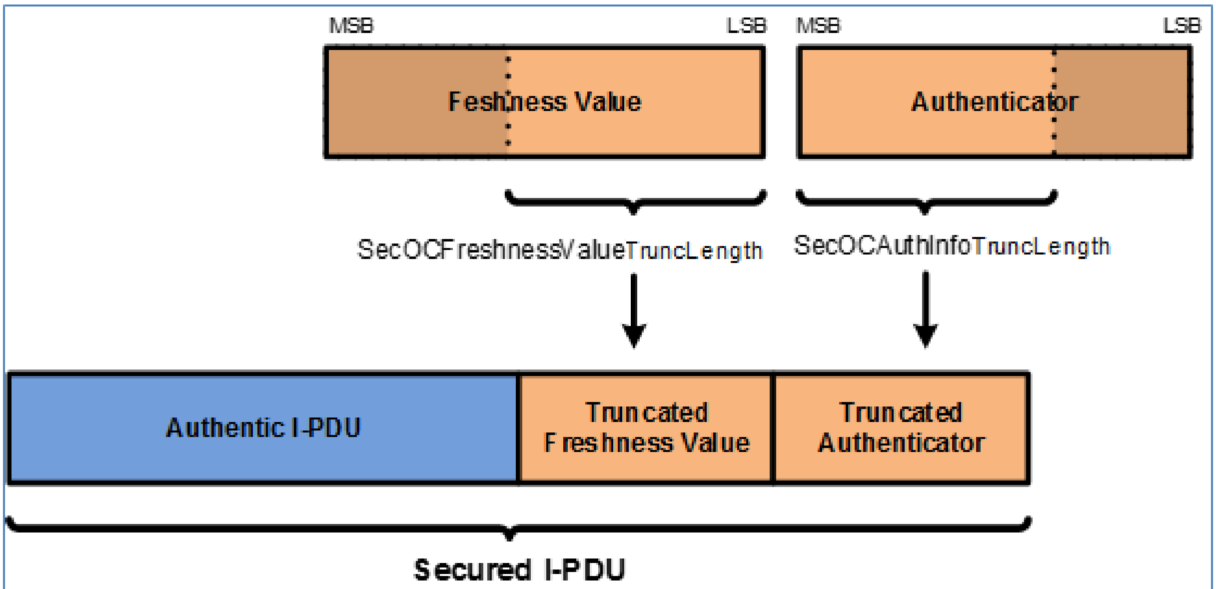

Figure 5.2: An example of Secured I-PDU contents with truncated Freshness Counter and truncated Authenticator (without Secured I-PDU Header)

Note: For the resource constraint embedded use case with static participants, we propose using Message Authentication Codes (MACs) as a basis for authentication (e.g. a CMAC [4] based on AES [3] with an adequate key length).

Note: In case a MAC is used, it is possible to transmit and compare only parts of the MAC. This is known as MAC truncation. However, this results in a lower security level at least for forgery of single MACs. While we propose to always use a key length of at least 128 bits, a MAC truncation can be beneficial. Of course, the actual length of the MAC for each use case has to be chosen carefully. For some guidance, we refer to appendix A of [4]. In general, MAC sizes of 64 bit and above are considered to provide sufficient protection against guessing attacks by NIST. Depending on the use case, different MAC sizes can be appropriate, but this requires careful judgment by a security expert.

[PRS_SecOc_00101] ⌈All SecOC data (e.g. Freshness Value, Authenticator, Data Identifier, SecOC message link data,...) that is directly or indirectly transmitted to the other side of a communication link shall be encoded in Big Endian byte order so that each SecOC module interprets the data in the same way.⌋*(RS_Main_00510)*

[PRS_SecOc_00102] ⌈The Secured I-PDU Header shall indicate the length of the Authentic I-PDU in bytes. The length of the Header shall be configurable by the parameter SecOCAuthPduHeaderLength.⌋*(RS_Main_00510)*

Each Secured I-PDU is configured with at least one Freshness Value. The Freshness Value refers to a monotonic counter that is used to ensure freshness of the Secured I-PDU. Such a monotonic counter could be realized by means of individual message counters, called Freshness Counter, or by a time stamp value called Freshness Timestamp. Freshness Values are to be derived from a Freshness Manager.

[PRS_SecOc_00103] ⌈If the parameter SecOCFreshnessValueTruncLength is configured to a smaller length than the actual freshness value, SecOC shall include only the least significant bits of the freshness value up to SecOCFreshnessValueTruncLength within the Secured I-PDU.
If the parameter SecOCFreshnessValueTruncLength is configured to 0, the freshness value shall not be included in the Secured I-PDU.⌋*(RS_Main_00510)*

[PRS_SecOc_00104] ⌈If SecOCUseAuthDataFreshness is set to TRUE, SecOC shall use a part of the Authentic I-PDU as freshness. In this case, SecOCAuthDataFreshnessStartPosition determines the start position in bits of the freshness inside the Authentic I-PDU and SecOCAuthDataFreshnessLen determines its length in bits.⌋*(RS_Main_00510)*

Note: This allows reusing existing freshness values from the payload which are guaranteed to be unique within the validity period of a Freshness Timestamp, e.g. a 4-bit E2E counter. In this case SecOC does not need to generate any additional counter values.

[PRS_SecOc_00105] ⌈The Freshness Manager provides or receives freshness information in interface functions as byte arrays. The freshness is always aligned to the MSB of the first byte in the array. The 15th bit of the freshness is the MSB of the 2nd byte and so on. Unused bits of the freshness array must be set to 0. The associated length information must be given in bits.⌋*(RS_Main_00510)*

5.1.2 Authentication of I-PDUs

[PRS_SecOc_00200] ⌈The creation of a `Secured I-PDU` consists of the following steps in this order:

1. Generate `Authenticator`.
2. Construct `Secured I-PDU`.
3. Increment `Freshness Counter`.

It shall be ensured that the Authenticator is generated before the freshness counter is incremented.⌋*(RS_Main_00510)*

[PRS_SecOc_00206] ⌈If the Freshness value calculation or the authenticator calculation fails and a default Pattern is configured by parameter `SecOCDefaultAuthenticationInformationPattern`, then Secured I-PDU shall use the default pattern for all the bytes of Freshness Value and Authenticator.⌋*(RS_Main_00510)*

[PRS_SecOc_00221] ⌈If the Freshness value calculation or the authenticator calculation fails and a default Pattern is not configured by parameter `SecOCDefaultAuthenticationInformationPattern`, then Secured I-PDU shall be dropped.⌋*(RS_Main_00510)*

Note:
Example:
`SecOCFreshnessValueTxLength` = 4bits
`SecOCAuthInfoTxLength` = 20 bits
`SecOCDefaultAuthenticationInformationPattern` = 0xA5
The resulting default `Authentication Information` within the secured PDU would be 0x05 (Truncated Freshness Value) | 0xA5 0xA5 0xA0 (Truncated `Authenticator`). "|" denotes concatenation.

[PRS_SecOc_00222] ⌈The Data Identifier of the `Secured I-PDU` (`SecOCDataId`) has a size of 16-bits.⌋*()*

[PRS_SecOc_00208] ⌈The data, on which the `Authenticator` is calculated, consists of `SecOCDataId`, `Authentic I-PDU` data and Complete Freshness Value in the given order. These are concatenated together respectively to make up the bit array that is passed into the authentication algorithm for Authenticator generation/verification.

`DataToAuthenticator` = Data Identifier | secured part of the `Authentic I-PDU` | Complete Freshness Value.⌋*(RS_Main_00510)*

Note: "|" denotes concatenation

[PRS_SecOc_00210] ⌈The Authenticator shall be truncated down to the number of bits specified by the parameter SecOCAuthInfoTruncLength.⌋*(RS_Main_00510)*

[PRS_SecOc_00211] ⌈The Secured I-PDU shall be constructed by adding the Secured I-PDU Header (optional), the Freshness Value (optional) and the Authenticator to the Authentic I-PDU.

The scheme for the Secured I-PDU (includes the order in which the contents are structured in the Secured I-PDU) shall be compliant with below:

SecuredPDU = SecuredIPDUHeader (optional) | AuthenticIPDU | FreshnessValue [SecOCFreshnessValueTruncLength] (optional) | Authenticator [SecOCAuthInfoTruncLength]⌋*(RS_Main_00510)*

Note: The Freshness Counter and the Authenticator included as part of the Secured I-PDU may be truncated per configuration specific to the identifier of the Secured I-PDU. Also, Freshness Value may be a part of Authentic I-PDU.

Note: This means there is no dependency between the IF/TP configuration of Up versus Lower PDU interfaces.

[PRS_SecOc_00213] ⌈If SecOCTxSecuredPduCollection is used, then SecOC shall transmit the Secured I-PDU as two messages: The original Authentic I--PDU and a separate Cryptographic I-PDU. The Cryptographic I-PDU shall contain all Authentication Information of the Secured I-PDU, so that the Authentic I-PDU and the Cryptographic I-PDU contain all information necessary to reconstruct the Secured I-PDU.⌋*(RS_Main_00510)*

[PRS_SecOc_00215] ⌈If SecOCTxSecuredPduCollection is used then SecOC shall repeat a part of the Authentic I-PDU inside the Cryptographic I-PDU as Message Linker and the Cryptographic I-PDU shall be constructed as Cryptographic I-PDU = Authentication Data | Message Linker⌋*(RS_Main_00510)*

Note: "|" denotes concatenation

[PRS_SecOc_00216] ⌈If SecOCUseMessageLink is used then SecOC shall use the value at bit position SecOCMessageLinkPos of length SecOCMessageLinkLen bits inside the Authentic I-PDU as the Message Linker.⌋*(RS_Main_00510)*

[PRS_SecOc_00220] ⌈For a Tx Secured I-PDU with SecOCAuthPduHeaderLength > 0, the SecOC module shall add the Secured I-PDU Header to the Secured I-PDU with the length of the Authentic I-PDU within the Secured I-PDU, to handle dynamic Authentic I-PDU.⌋*(RS_Main_00510)*

5.1.3 Verification of I-PDUs

[PRS_SecOc_00300] ⌈The verification of a Secured I-PDU consists of the following 3 steps:

1. Calculate the expected Freshness Value.

2. Construct Data for Authentication.

3. Verify Authentication Information.

⌋(RS_Main_00510)

[PRS_SecOc_00306] ⌈If the verification of the Authenticator could be successfully executed but the verification failed (e.g. the MAC verification has failed or the key was invalid), the authentication verify attempt counter shall be incremented, and the freshness value shall be updated to the next valid value.⌋(RS_Main_00510)

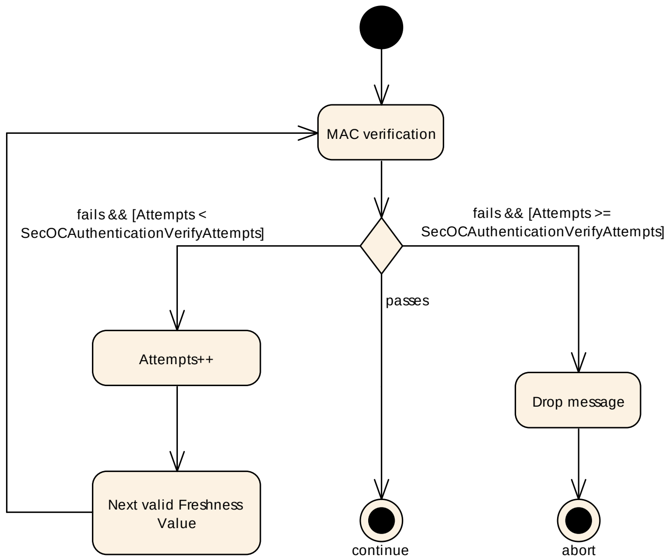

Figure 5.3: Verification of MAC

[PRS_SecOc_00309] ⌈If the authentication verify attempt counter has reached the threshold of the SecOCAuthenticationVerifyAttempts on a failed verify attempt, then the received PDU shall be discarded.⌋(RS_Main_00510)

[PRS_SecOc_00223] ⌈If the parameter SecOCEnableForcedPassOverride is true, then the SecOC shall provide a mechanism to allow reception of the PDUs that failed in authentication.⌋(RS_Main_00510)

[PRS_SecOc_00313] ⌈If SecOCRxSecuredPduCollection is used, then SecOC shall not perform any verification until it has received both the Authentic I-PDU and Cryptographic I-PDU which make up the Secured I-PDU. Only after both have been received SecOC shall attempt to verify the resulting Secured I-PDU. If SecOC_VerifyStatusOverride is used, the verification result and I-PDU are handled according to overrideStatus value.⌋(RS_Main_00510)

Note: This applies to all instances when a Secured I-PDU is received by SecOC from the lower layer, which happens in parts as described above when SecOCRxSecuredPduCollection is used. There is no further distinction made throughout this document to avoid duplication and clutter.

[PRS_SecOc_00314] ⌈If SecOCRxSecuredPduCollection is used then SecOC shall not attempt to verify the Secured I-PDU until it has received and buffered an

Authentic I-PDU and Cryptographic I-PDU with matching Message Linker values. If SecOC_VerifyStatusOverride is used, the verification result and I-PDU are handled according to overrideStatus value.⌋(*RS_Main_00510*)

Note: If SecOCUseMessageLink has 0 multiplicity, it means If SecOCMessageLinkLen is 0 and that Message Linker Values are always matching.

[PRS_SecOc_00315] ⌈Upon reception of a Secured I-PDU, SecOC shall parse the Authentic I-PDU, the Freshness Value and the Authenticator from it.⌋(*RS_Main_00510*)

[PRS_SecOc_00316] ⌈The data that is used to calculate the Authenticator (DataToAuthenticator) on the receiver side shall be constucted. This data is comprised of SecOCDataId | AuthenticIPDU | FreshnessVerifyValue.⌋(*RS_Main_00510*)

[PRS_SecOc_00317] ⌈The SecOC module shall verify the Authenticator by passing DataToAuthenticator, length of DataToAuthenticator, the Authenticator parsed from Secured I-PDU, and SecOCAuthInfoTruncLength into the authentication algorithm corresponding to configured cryptographic service. If SecOC_VerifyStatusOverride is used, the verification result and I-PDU are handled according to overrideStatus value.⌋(*RS_Main_00510*)

Note: If the Freshness Manager requires the status of a Secured I-PDU if it was verified successfully or not, e.g. to synchronize time or counter, then this status shall be taken from the VerificationStatus service provided by SecOC.

5.1.3.1 Successful verification of I-PDUs

[PRS_SecOc_00320] ⌈If the verification of a Secured I-PDU was successful or the status override was set accordingly, the SecOC module shall pass the Authentic I-PDU to the upper layer communication modules using the lower layer interfaces of the communication stack.⌋(*RS_Main_00510*)

5.1.3.2 Skipping Authentication for Secured I-PDUs at SecOC

[PRS_SecOc_00330] ⌈For a Rx Secured I-PDU, there should be a configuration option to skip the verification. In this case the SecOC module shall extract the Authentic I-PDU without Authentication.⌋(*SRS_BSW_00385*)

5.1.3.3 Error handling and discarding of reception

[PRS_SecOc_00340] ⌈If the lower layer transport protocol module reports an error during reception of a Secured I-PDU, the SecOC module shall drop the Secured I-PDU and free all corresponding buffers.⌋(*SRS_BSW_00385*)

[PRS_SecOc_00341] ⌈If the Crypto module reports an error during verification (verification cannot be performed) of a Secured I-PDU, the SecOC module shall not provide the Authentic I-PDU. It shall keep the Secured I-PDU (if not overwritten by an incoming Secured I-PDU of the same type) and start the verification with the next call of the scheduled main function.⌋*(RS_Main_00510, SRS_BSW_00385)*

[PRS_SecOc_00342] ⌈If SecOC has received both an Authentic I-PDU and a Cryptographic PDU and the verification of the resulting Secured I-PDU fails, both the Authentic and Cryptographic I-PDU shall remain buffered and verification shall be reattempted each time new data for any of them is received.⌋*(RS_Main_00510)*

Note: This and the above requirement ensure that even if either an Authentic I-PDU or a Cryptographic I-PDU is lost in transit, SecOC will still function as expected as soon as an Authentic I-PDU and its corresponding Cryptographic I-PDU are received in direct succession.

5.2 Error detection

[PRS_SecOc_00500] ⌈The SecOC module shall be able to report errors in case the module is called before initialization, in case a freshness value cannot be provided or in case a there is no cryptographic operation configured for the verification check.⌋ *(SRS_BSW_00337, SRS_BSW_00350, SRS_BSW_00385, SRS_BSW_00450)*

5.3 Security Profiles

5.3.1 Overview of security profiles

Secure Onboard Communication protocol allows multiple cryptographic algorithms and modes for the MAC calculation and how the truncation of the MAC and freshness value(if applicable) shall be done.The security profiles provide a consistent set of values for a subset of configuration parameters that are relevant for the configuration of Secure Onboard Communication.

[PRS_SecOc_00600] ⌈Each Security Profile shall provide the configuration values for the authentication algorithm (parameter `algorithmFamily`, `algorithmMode` and `algorithmSecondaryFamily` in `CryptoServicePrimitive`), length of freshness Value, if applicable (parameter `SecOCFreshnessValueLength`), length of truncated Freshness Value (parameter `SecOCFreshnessValueTruncLength`), length of truncated MAC (parameter `SecOCAuthInfoTruncLength`), and a description of the profile.⌋*(RS_Main_00510)*

5.3.2 SecOC Profile 1 (or 24Bit-CMAC-8Bit-FV)

[PRS_SecOc_00610] ⌈Using the CMAC algorithm based on AES-128 according to NIST SP 800-38B to calculate the MAC, use the eight least significant bit of the freshness value as truncated freshness value and use the 24 most significant bits of the MAC as truncated MAC.⌋*(RS_Main_00510)*

Parameter	Configuration value
The algorithm for the MAC (parameter algorithmFamily)	CRYPTO_ALGOFAM_AES
The algorithm mode for the MAC (parameter algorithmMode)	CRYPTO_ALGOMODE_CMAC
Additional algorithm family configuration (parameter algorithmSecondaryFamily, not used in this profile)	CRYPTO_ALGOFAM_NOT_SET
Length of Freshness Value (parameter SecOCFreshnessValueLength)	Not Specified
Length of truncated Freshness Value (parameter SecOCFreshnessValueTruncLength	8 bits
Length of truncated MAC (parameter SecOCAuthInfoTruncLength)	24 bits

5.3.3 SecOC Profile 2 (or 24Bit-CMAC-No-FV)

[PRS_SecOc_00620] ⌈Using the CMAC algorithm based on AES-128 according to NIST SP 800-38B to calculate the MAC, don't use any freshness value at all and use the 24 most significant bits of the MAC as truncated MAC. The profile shall only be used if no synchronized freshness value is established. There is no restriction to a special bus.⌋*(RS_Main_00510)*

Parameter	Configuration value
The algorithm for the MAC (parameter algorithmFamily)	CRYPTO_ALGOFAM_AES
The algorithm mode for the MAC (parameter algorithmMode)	CRYPTO_ALGOMODE_CMAC
Additional algorithm family configuration (parameter algorithmSecondaryFamily, not used in this profile)	CRYPTO_ALGOFAM_NOT_SET
Length of Freshness Value (parameter SecOCFreshnessValueLength)	0
Length of truncated Freshness Value (parameter SecOCFreshnessValueTruncLength	0 bits
Length of truncated MAC (parameter SecOCAuthInfoTruncLength)	24 bits

5.3.4 SecOC Profile 3 (or JASPAR)

[PRS_SecOc_00630] ⌈This profile depicts one configuration and usage of the JasPar counter base FV with Master-Slave Synchronization method. It uses the CMAC algorithm based on AES-128 according to NIST SP 800-38B Appendix-A to calculate the MAC. Use the 4 least significant bits of the freshness value as truncated freshness value and use the 28 most significant bits of the MAC as truncated MAC. Freshness Value provided to SecOC shall be constructed as described in the [UC_SecOC_00202]. The profile shall be used for CAN.⌋*(RS_Main_00510)*

Parameter	**Configuration value**
The algorithm for the MAC (parameter algorithmFamily)	CRYPTO_ALGOFAM_AES
The algorithm mode for the MAC (parameter algorithmMode)	CRYPTO_ALGOMODE_CMAC
Additional algorithm family configuration (parameter algorithmSecondaryFamily, not used in this profile)	CRYPTO_ALGOFAM_NOT_SET
Length of Freshness Value (parameter SecOCFreshnessValueLength)	64 bits
Length of truncated Freshness Value (parameter SecOCFreshnessValueTruncLength	4 bits
Length of truncated MAC (parameter SecOCAuthInfoTruncLength)	24 bits

6 Configuration parameters

The table below describes the configuration parameters for the protocol. For the communication between AP and CP using SOME/IP network binding or raw data streaming, the protocol has a reduced functionality, therefore some parameters are not available in AP or they are implementation specific. These are described in column "Applicability to AP SOME/IP network binding". As a consequence, the requirements referring these parameters are not applicable either. The parameters that are not available are because following features are not available:

- It is not possible to use part of the Authentic PDU to construct as freshness information
- It is not possible to separate the Secure PDU in two different PDUs: Authentic and Cryptographic PDUs
- Provision of Freshness value already truncated by FM (the truncation is always done by SecOC)

Parameter	***Description***	***Applicability to AP SOME/IP network binding***
SecOCAuthPduHeaderLength	This parameter indicates the length (in bytes) of the Secured I-PDU Header in the Secured I-PDU. The length of zero means there's no header in the PDU.	no
SecOCFreshnessValueTruncLength	This parameter defines the length in bits of the Freshness Value to be included in the payload of the Secured I-PDU. This length is specific to the least significant bits of the complete Freshness Counter. If the parameter is 0 no Freshness Value is included in the Secured I-PDU.	yes

▽

△

SecOCUseAuthDataFreshness	A Boolean value that indicates if a part of the Authentic I-PDU shall be passed on to the SWC that verifies and generates the Freshness. If it is set to TRUE, the values SecOCAuthDataFreshnessStartPosition and SecOCAuthDataFreshnessLen must be set to specify the bit position and length within the Authentic I-PDU.	no
SecOCAuthDataFreshnessStart-Position	This value determines the start position in bits (uint16) of the Authentic PDU that shall be passed on to the Freshness SWC.	no
SecOCAuthDataFreshnessLen	This attribute defines the length in bits of the authentic PDU data that is passed to the SWC that verifies and generates the Freshness.	no
SecOCProvideTxTruncatedFresh-nessValue	This parameter specifies if the Tx query freshness function provides the truncated freshness info instead of generating this by SecOC in this case, SecOC shall add this data to the Authentic PDU instead of truncating the freshness value.	no
SecOCAuthenticationBuildAt-tempts	Parameter specifies the number of authentication build attempts.	Implementation specific
SecOCDefaultAuthenticationInfor-mationPattern	The parameter describes the behavior of SecOC when authentication build counter has reached the configuration value SecOCAuthenticationBuildAttempts, or the query of the freshness function returns non-recoverable error (example: a systematic failure due to freshness value configuration) or the calculation of the Authenticator has returned a non-recoverable error (example: a systematic failure due wrong crypto configuration or missing ▽	Implementation specific

▽

△

	△ key). If the configuration parameter is not present, SecOC module shall remove the Authentic I-PDU from its internal buffer and cancel the transmission request If the configuration parameter is present, SecOC will use this value for each byte of Freshness Value and Authenticator when building the Authentication Information, and will not cancel the transmission request.	
SecOCDataId	This parameter defines a unique numerical identifier for the Secured I-PDU.	yes
SecOCFreshnessValueID	This parameter defines the Id of the Freshness Value	yes
SecOCTxAuthServiceConfigRef	This reference is used to define which crypto service function is called for authentication	Implementation specific
SecOCAuthInfoTruncLength	This parameter defines the length in bits of the authentication code to be included in the payload of the Secured I-PDU.	yes
SecOC_VerifyStatusOverride	When this configuration option is set to TRUE then the functionality inside the function SecOC_-VerifyStatusOverride to send I-PDUs to upper layer independent of the verification result is enabled.	Implementation specific
SecOCTxSecuredPduCollection	Two separate Pdus are transmitted to the lower layer: Authentic I-PDU and Cryptographic I-PDU.	no

▽

△

SecOCFreshnessValueLength	This parameter defines the complete length in bits of the Freshness Value. As long as the key doesn't change the counter shall not overflow. The length of the counter shall be determined based on the expected lifetime of the corresponding key and frequency of usage of the counter.	yes
SecOCTxPduUnusedAreasDefault	The AUTOSAR SecOC module fills not used areas of a transmitted Secured Pdu with this byte pattern. This attribute is mandatory to avoid undefined behavior.	Implementation specific
SecOCUseMessageLink	SecOC links an Authentic I-PDU and Cryptographic I-PDU together by repeating a specific part (Message Linker) of the Authentic I-PDU in the Cryptographic I-PDU.	no
SecOCMessageLinkPos	The position of the Message Linker inside the Authentic I-PDU in bits. The bit counting is done according to 01068 and the bit ordering is done according to TPS_SYST_01069.	no
SecOCMessageLinkLen	Length of the Message Linker inside the Authentic I-PDU in bits.	no
SecOCSecuredRxPduVerification	This parameter defines whether the signature authentication or MAC verification shall be performed on this Secured I-PDU. If set to false, the SecOC module extracts the Authentic I-PDU from the Secured I-PDU without verification.	yes
SecOCUseTxConfirmation	A Boolean value that indicates if the function `SecOC_SPduTxConfirmation` shall be called for this PDU.	no

▽

△

SecOCVerificationStatusPropagationMode	This parameter is used to describe the propagation of the status of each verification attempt from the SecOC module to SWCs.	no
SecOCIgnoreVerificationResult	The result of the authentication process (e.g. MAC Verify) is ignored after the first try and the SecOC proceeds like the result was a success. The calculation of the authenticator is still done, only its result will be ignored. true: enabled (verification result is ignored). false: disabled (verification result is NOT ignored).	no
SecOCEnableForcedPassOverride	When this configuration option is set to TRUE then the functionality inside the function SecOC_VerifyStatusOverride to send I-PDUs to upper layer independent of the verification result is enabled.	no
SecOCPropagateOnlyFinalVerificationStatus	This parameter Is used to specify if the verification status shall be reported only after the final determination of the verification status (TRUE) or on every verification attempt (FALSE).	no
SecOCAuthenticationVerifyAttempts	This parameter specifies the number of authentication verify attempts that are to be carried out when the verification of the authentication information failed for a given Secured I-PDU. If zero is set, then only one authentication verification attempt is done.	yes

7 Protocol usage and guidelines

This chapter has no content.

8 References

[1] IEC:The Basic Model, IEC Norm

[2] NIST:Secure Hash Standard (SHS)
http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

[3] NIST:Announcing the Advanced Encryption Standard (AES)
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[4] NIST Special Publication 800-38B:Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication
http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf

A Change history of AUTOSAR traceable items

A.1 Change History of this document according to AUTOSAR Release R23-11

A.1.1 Added Specification Items in R23-11

[PRS_SecOc_00221] [PRS_SecOc_00222] [PRS_SecOc_00223]

A.1.2 Changed Specification Items in R23-11

[PRS_SecOc_00200] [PRS_SecOc_00206] [PRS_SecOc_00208] [PRS_SecOc_00210] [PRS_SecOc_00211] [PRS_SecOc_00300] [PRS_SecOc_00306] [PRS_SecOc_00309] [PRS_SecOc_00316]

A.1.3 Deleted Specification Items in R23-11

[PRS_SecOc_00100] [PRS_SecOc_00201] [PRS_SecOc_00202] [PRS_SecOc_00203] [PRS_SecOc_00204] [PRS_SecOc_00205] [PRS_SecOc_00207] [PRS_SecOc_00209] [PRS_SecOc_00212] [PRS_SecOc_00214] [PRS_SecOc_00217] [PRS_SecOc_00218] [PRS_SecOc_00219] [PRS_SecOc_00301] [PRS_SecOc_00302] [PRS_SecOc_00303] [PRS_SecOc_00304] [PRS_SecOc_00305] [PRS_SecOc_00307] [PRS_SecOc_00308] [PRS_SecOc_00310] [PRS_SecOc_00311] [PRS_SecOc_00312] [PRS_SecOc_00318]