VOLKSWAGEN

AKTIENGESELLSCHAFT

Secure on-board communication

Protocol definition

Technical development, cross-sectional load booklet: LAH.000.900.AG

Author	Chache, Alexander
Dept./OU	EEKS/5
Phone	05361-9-14949
Mobile	-
Fax	-
E-mail	alexander.tschache@volkswagen.de

First edition	18.07.2017
Change status	09.09.2016
Specification version	3.0
Baseline	3.0 ( )

Distributor
Dept./OU, Cost center, Name
Dept./OU, Cost center, Name
Dept./OU, Cost center, Name
Dept./OU, Cost center, Name

Table of contents

1 General

1.1 Purpo se

[I: SOK_PROT_7]

This document describes the concept of secure onboard communication for the authen- tic transmission of data between control units within a vehicle.

1.2 Abbreviations

[I: SOK_PROT_21]

Table: Abbreviations

Abbreviation	Description
SOK	Secure on-board communication

1.3 Definitions

[I: SOK_PROT_23]

Table: Definitions

Designation	Definition
(unprotected) Time message	The message containing the current SOK time
Authentic Time message	A message which contains the current SOK time, and which is associated with a Signature is protected
Challenge message	A message which contains a random number as a challenge, whose Basis a signature is calculated
Data message	A message containing data to be protected
Protocol data unit (PDU)	The actual data, which is transmitted via a communication channel between is transferred to two SOK participants
Protection message	A message containing the signature over messages to be protected contains
SOK participants	A control unit which communicates via SOK
SOK Time	A 10Hz counter, which is authenticated over all SOK participants. is synchronized
Responsible function	The function that initiates the transmission of data in the vehicle and determines which of its data is protected with SOK must.
Virtual Protection message	A message that does not contain any applicative user data and is used only for the Generation of a signature serves

2 Functional description

2.1 Functional scope and delimitation

[I: SOK_PROT_11]

SOK offers the possibility to transfer data between ECUs in the vehicle in a tamper-proof way.

[I: SOK_PROT_47]

For this purpose, cryptographic protection information is transmitted in addition to the data to be protected, with which an SOK subscriber can verify the authenticity of the received data. This prevents an SOK subscriber from processing manipulated data.

[I: SOK_PROT_483]

However, SOK does not use encryption and therefore cannot guarantee the confidentiality of data.

[I: SOK_PROT_52]

SOK requires cryptographic keys to generate this protection information. It is not the task of SOK to introduce these keys into the ECUs and to store them in a protected manner. In the following, it is assumed that the necessary keys are already in the ECUs involved.

[I: SOK_PROT_367]

Two methods are used to protect against replay attacks:
1. Challenge response
2. Implicit timestamps through a previously authentically distributed time

[I: SOK_PROT_77]

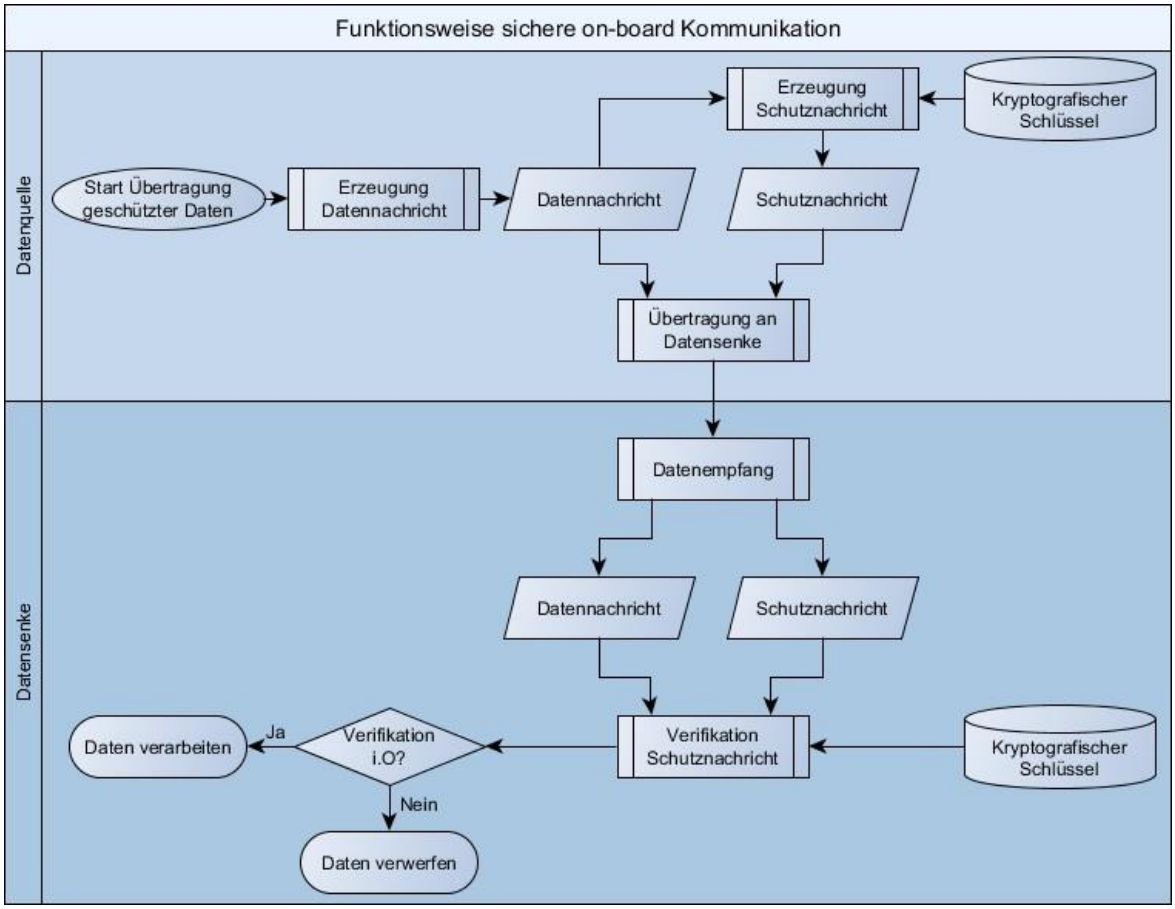
SOK is designed to be implemented independently of the underlying communication channel (CAN, CAN-FD, Ethernet/IP, FlexRay, etc.), unless otherwise noted.

[A: SOK_PROT_169]

The byte order of all data in this document is Big Endian.
If data types are smaller than a field to be written, the datum is extended to the required length by adding logical zero bits starting from the most significant bit.
Bit position 0 is the least significant bit. All data specifications of this document are left-flush.

[I: SOK_PROT_56]

The following graphic roughly illustrates how SOK works.



[I: SOK_PROT_49]

Which data is to be protected is determined by the respective function that uses the data.

[I: SOK_PROT_370]

Protected data received that could not be verified as authentic by SOK is not forwarded within the SOK subscriber and thus does not reach the associated function. The substitute response for missing messages is to reply to the corresponding function.

2.2 Function distribution

[I: SOK_PROT_16]

This chapter describes the roles of individual SOK control units.

2.2.1 Data source

[A: SOK_PROT_18]

The data source is the originating control device of the data to be protected and its associated protection messages.

[A: SOK_PROT_61]

The data source is responsible for signing the data worth protecting.

[A: SOK_PROT_82]

For signing, the data source first generates the data message. The data message is signed together with additional information.

[A: SOK_PROT_364]

The data source transmits the data message and the associated signature in a protection message to the data sink or to a group of data sinks.

[A: SOK_PROT_62]

The data source must use the same cryptographic key for signing that is used by the data sink to verify the signature.

2.2.2 Data sink

[A: SOK_PROT_183]

The data sink receives the data messages transmitted by a data source and the corresponding protection message.

[A: SOK_PROT_184]

The data sink verifies the signature received in the protection message via the data message and can thus determine whether the data message is authentic. It uses the same cryptographic key that the data source used to create the signature.

2.2.3 Time server

[A: SOK_PROT_63]

The SOK time server is responsible for the distribution of a vehicle-wide authentic SOK time.

[A: SOK_PROT_186]

Exactly one SOK time server exists in the vehicle.

[A: SOK_PROT_64]

The SOK time server must send the current SOK time every second as an unprotected message to all SOK participants.

[I: SOK_PROT_95]

This unprotected time serves the other SOK participants as a reference value for the detection of clock jitter.

[A: SOK_PROT_94]

The SOK time server must send an SOK participant a current SOK time that can be authenticated by the SOK participant.

[A: SOK_PROT_96]

The authentic transmission of the SOK time is done via the SOK Challenge-Response Protokoll.

[A: SOK_PROT_496]

The control device acting as SOK time server can also be an SOK subscriber itself, which sends or receives messages protected by SOK independently of its function as SOK time server.

2.3 Protocol definition

2.3.1 Message types

[I: SOK_PROT_67]

This section describes the different message types used by SOK.

[A: SOK_PROT_497]

The term "message" is to be understood here as a container into which data re levant for the SOK protocols are aggregated, and not as a message carried directly over a communication channel.

2.3.1.1 Data message

[A: SOK_PROT_76]

A data message the following format:

Bit position	0 ~ Z-1	Z ~ Z+L-1	Z+L ~ ...
Content	User data Part 1Part 2	Counter	User data

Z is the position of the counter and configurable for a data message. L is the length of the counter in bits.

[I: SOK_PROT_85]

The counter can thus be placed according to the requirements of the responsible function. Depending on the transmission frequency of the message to be protected, the length of the counter can be varied.

[A: SOK_PROT_436]

The minimum length of the counter depends on the transmission frequency of the data message. For effective replay protection, the following inequality must be observed:
<Cycletime of the data message in milliseconds> * (2<counter length$^{\text{in bit}}$>) > 100.

[I: SOK_PROT_437]

This ensures that a counter value is not repeated within 100 milliseconds. These 100 milliseconds correspond to the incrementing frequency of the SOK time.

[A: SOK_PROT_88]

The counter is incremented by 1 with each data message sent and reset to 0 on overflow.

[A: SOK_PROT_128]

If the data message to be protected already has E2E security, it contains an E2E counter with Z = 7 and L = 4. This is reused by SOK.

2.3.1.2 Protection message

[A: SOK_PROT_372]

The protection message contains the signature of an associated data message.

[A: SOK_PROT_473]

A protection message can be transmitted either within the same PDU as the data message or in a separate PDU.

[A: SOK_PROT_79]

A protection message has the following format:

Bit position	0 ~ N-1	N ~ N+L'-1
Content	Signature counter	

N is the length of the signature in bits and L' is the length of the counter in the protection message.

[A: SOK_PROT_80]

If the associated data message contains a counter and the protection message is transmitted in a separate PDU, the value and length of the counter of the signature message is adopted for the counter of the protection message.

2.3.1.3 Challenge message

[A: SOK_PROT_87]

The challenge message is used exclusively within the challenge response protocol.

[A: SOK_PROT_89]

A challenge message has the following format:

Bit position	**0 ~ 63**
Content	Challenge

[A: SOK_PROT_90]

The challenge is a 64-bit random value that is re- chosen for each challenge message sent.

2.3.1.4 Time message

[A: SOK_PROT_93]

Time messages are sent exclusively by the time server.

[A: SOK_PROT_98]

A distinction is made between unprotected time messages and authentic time messages.

[A: SOK_PROT_99]

An unprotected time message has the following format:

Bit position	**0 ~ 55**
Content	current SOK time

[A: SOK_PROT_100]

An authentic time message is a data message with counter length 0, which contains the current SOK time as useful data, and has the following format:

Bit position	**0 ~ 55**
Content	current SOK time

[A: SOK_PROT_450]

Unlike the unprotected time message, the authentic time message is protected with a protection message.

2.3.1.5 Virtual protection message

[A: SOK_PROT_102]

A virtual protection message contains only a counter and a signature. It does not contain any additional user data.

[I: SOK_PROT_439]

The virtual protection message is used for the "Authenticity message" function.

[A: SOK_PROT_103]

The virtual protection message has the following format:

Bit position	**0 ~ L**	**L+1 ~ ...**
Content	Counter signature	

L is the length of the counter in bits.

2.3.2 Protocol data unit (PDU)

[I: SOK_PROT_471]

A PDU refers to the data transmitted over a communication channel.

[I: SOK_PROT_472]

A PDU contains the messages specified for SOK.

[A: SOK_PROT_480]

A PDU can contain individual messages or combinations of messages. As a rule, a PDU contains either a single data or protection message or a data message with associated protection message.

[A: SOK_PROT_391]

The following combinations are
possible:
1. The PDU contains both the data message and the associated protection message
2. The PDU contains only the data message
3. The PDU contains only the protection message
4. The PDU contains only one challenge message
5. The PDU complies with only one virtual guard message (special case of 1)

2.3.3 Protocol procedures

[A: SOK_PROT_106]

This section describes the various protocol procedures for SOK.

[A: SOK_PROT_452]

For each data message to be protected, an SOK PDU ID is defined by the SOK function owner. This is used to uniquely identify the data message and the associated configuration of SOK across all SOK subscribers and is independent of the transmission path.

2.3.3.1 Signature transfer and verification

[A: SOK_PROT_375]

The transmission of the protection message of a data message can be done in two ways:
1. Within the same PDU as the data message
2. In a separate PDU

[A: SOK_PROT_446]

If the protection message is transmitted in the same PDU as the data message, the protection message must directly follow the associated data message within the PDU.

[A: SOK_PROT_376]

Transmission of the protection message in a separate PDU is selected only if it is not possible to place the protection message in the same PDU as the data message.

[I: SOK_PROT_377]

There may be several reasons for this (selection):
1. The existing data definition must be changed, but not all controllers receiving the original data can be changed.
2. The original PDU is already filled to the point where there is no more room for the protection message.

[I: SOK_PROT_378]

If the protection message is transmitted as part of the same PDU, it is possible for the data sink to verify the authenticity of the contained data message immediately after receiving the PDU.

[A: SOK_PROT_379]

If the protection message is transmitted in a separate PDU, the data sink must wait until it has received both a data message and an associated protection message for an SOK PDU ID, and only then performs verification.

[A: SOK_PROT_443]

If the protection message is transmitted in a separate PDU, the data source must transmit this PDU immediately after the data message associated with the PDU. It must ensure that both PDUs are sent with the shortest possible time interval.

[I: SOK_PROT_380]

If available, the counter of the data message is repeated in the protection message. This results in a simple assignment possibility between an instance of a data message and the associated protection message.

[A: SOK_PROT_381]

The data sink buffers both the last data message received and the last protection message received. Any messages that have already been buffered are overwritten when the next message is received.

[A: SOK_PROT_382]

The data sink only performs verification if the counter values of the currently buffered data message and protection message match. Buffered data messages may only be processed further after their verification has been successful.

2.3.3.2 Challenge response

[I: SOK_PROT_108]

The challenge-response protocol is used when a data sink requests authenticated data from a data source.

[I: SOK_PROT_185]

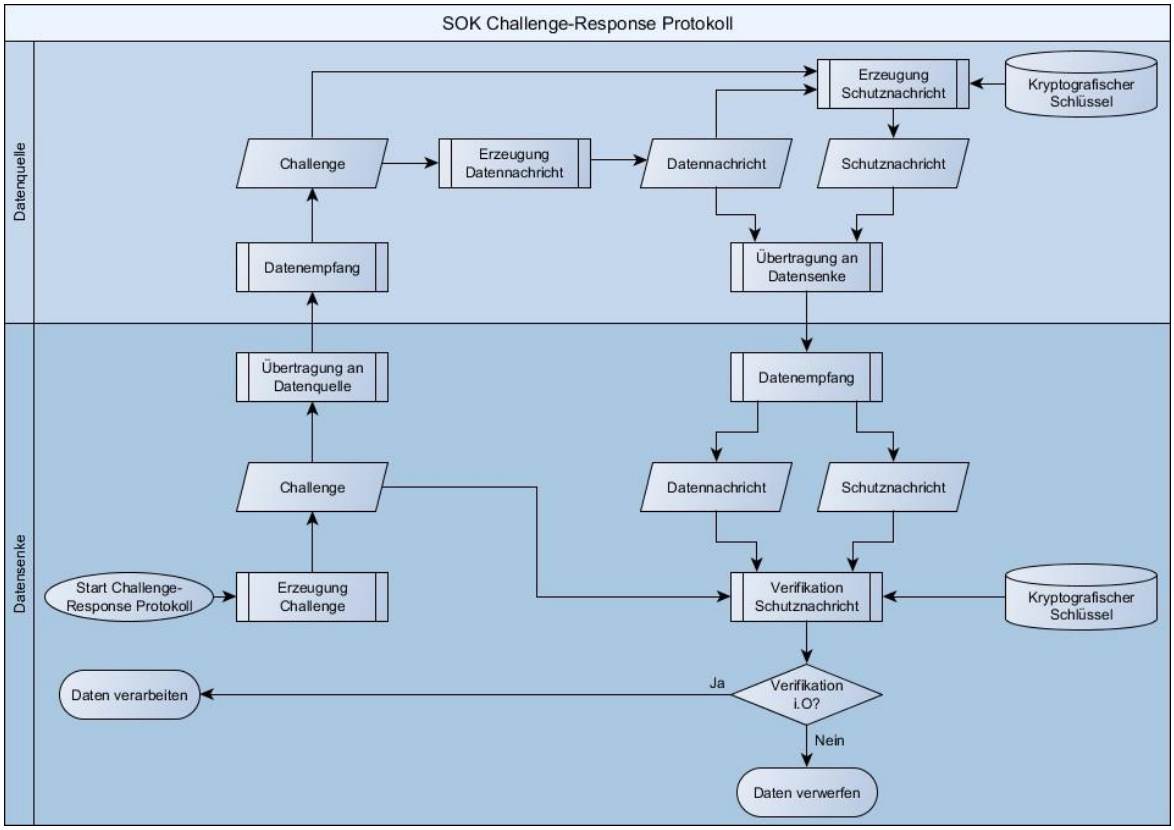
The challenge-response protocol requires a one-to-one relationship between data source and data sink.

[I: SOK_PROT_109]

The data sink first transmits a 64-bit challenge to the data source. The data source then transmits the requested data within a data message together with a signature. The transmitted challenge is included in the calculation of the signature. The data sink checks the signature of the received data, whereby the previously transmitted challenge is included in the check.

[I: SOK_PROT_405]

The following graphic illustrates the flow of the SOK Challenge-Response protocol:



[I: SOK_PROT_110]

The following describes the SOK challenge-response protocol flow.

[A: SOK_PROT_111]

The data sink generates a 64 bit challenge (i.e. a random number) and generates a challenge message with it.

[A: SOK_PROT_112]

The data sink transmits the challenge message to the data source.

[A: SOK_PROT_190]

The data source infers the requested data message from the received challenge message.

[A: SOK_PROT_113]

The data source generates the requested data message.

[A: SOK_PROT_114]

The data source calculates the signature for the data message. The challenge is included in the calculation of the signature.

[A: SOK_PROT_115]

The data source transmits the data message and the associated signature in a protection message to the data sink.

[A: SOK_PROT_117]

The data sink calculates the expected signature based on the received data message and its previously sent challenge and compares it with the received signature.

[A: SOK_PROT_119]

If the values of the signatures are the same, the received data of the data message is authentic and the challenge-response protocol has been performed successfully. From the-

At this point, the used challenge is discarded, so that a replay attack is not possible.

[A: SOK_PROT_118]

If the values of the signatures differ, the received data is not authentic. However, the challenge is not discarded and the data sink continues to check received data messages and signatures.

[A: SOK_PROT_120]

If the data source does not receive authentic data within a time period defined by the responsible function, the challenge-response protocol terminates. The enforcement of this timeout as well as the replacement response is the responsibility of the function that requested the data.

2.3.3.3 Time distribution

[A: SOK_PROT_122]

SOK distinguishes between two methods for time distribution: cyclic (unprotected) time distribution and authentic time distribution via the challenge-response method.

[A: SOK_PROT_123]

Both types of time distribution are the responsibility of the SOK time server.

[A: SOK_PROT_124]

For cyclic time distribution, the SOK time server sends an unprotected time message to active SOK participants every second, which contains its current SOK time.

[A: SOK_PROT_125]

An SOK subscriber uses the received unprotected SOK time as a benchmark to calculate the deviation of its internal authenticated SOK time from the vehicle-wide time.

[A: SOK_PROT_126]

If an SOK participant does not yet have an internal authenticated SOK time or if it detects from an unprotected time message that its internal authenticated SOK time deviates by more than 50ms from the vehicle-wide SOK time, it starts the protocol for authentic time distribution. Detailed requirements can be found in the chapter on time management.

[A: SOK_PROT_131]

The authentic time distribution is implemented via the SOK challenge-response protocol between the SOK time server and SOK participant. Here, the user data consists of the current SOK time of the SOK time master.

[A: SOK_PROT_129]

To initiate the time distribution protocol, the SOK participant sends a challenge message to the SOK time server.

[A: SOK_PROT_130]

The SOK time server receives the challenge message. It waits until its internal SOK time is incremented the next time and then generates an authentic time message with associated signature for the requesting SOK participant and transmits it.

[I: SOK_PROT_492]

The SOK time server's waiting for the next increment of its internal SOK time ensures that at the time the SOK subscriber receives the authentic time, there is as little difference as possible between the received SOK time and the actual SOK time of the SOK time server.

[A: SOK_PROT_132]

The SOK subscriber receives the authentic time message with the associated signature. If the signature check is successful, the SOK station adopts the received SOK time as the internal authenticated SOK time.

[A: SOK_PROT_133]

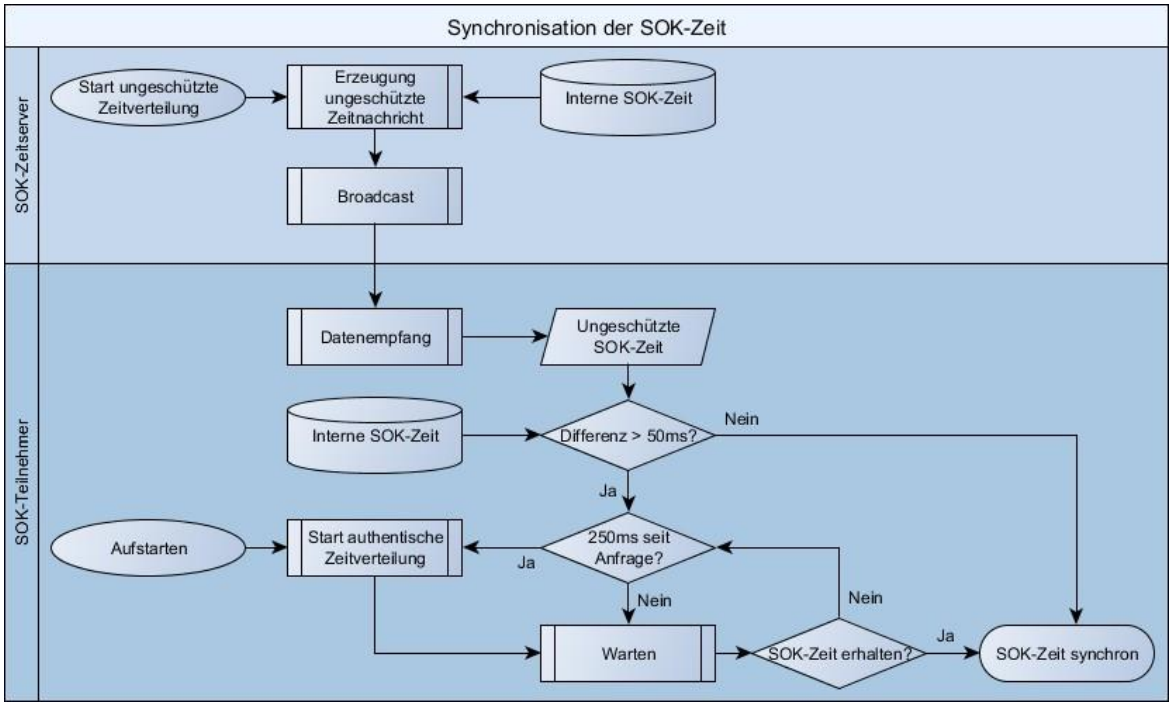
If the signature check fails, the SOK subscriber continues to use its current internal authenticated SOK time, if available.

[A: SOK_PROT_493]

If the SOK participant could not get an authentic SOK time from the SOK time server after 250 milliseconds after the start of the time distribution protocol, it starts the protocol again.

[A: SOK_PROT_407]

The following graphic illustrates the synchronization of the SOK time:



2.3.3.4 Authentic broadcast

[I: SOK_PROT_135]

In the authentic broadcast protocol, a data source sends data messages with an associated signature without a request from a data sink.

[I: SOK_PROT_136]

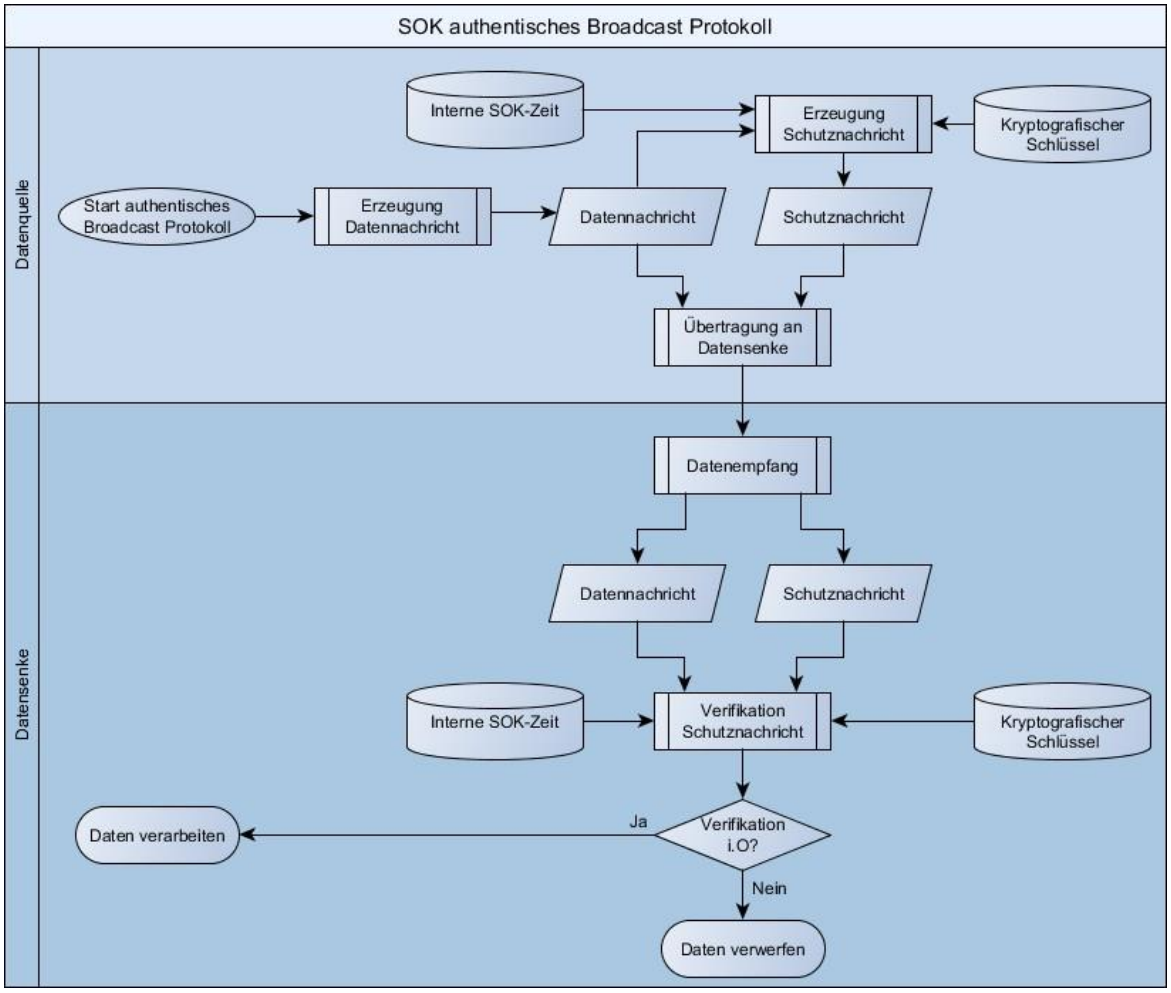
This protocol is used, for example, to protect data sent cyclically.

[I: SOK_PROT_408]

In contrast to the SOK challenge-response protocol, the signature is not calculated on the basis of a challenge from the data sink, but on the basis of the common, authenticated SOK time. This makes it possible for multiple data sinks to receive and authenticate the same message, since the authenticated SOK time is the same in all SOK participants.

[I: SOK_PROT_409]

The following graphic illustrates the SOK protocol for authentic broadcast:



[A: SOK_PROT_194]

To execute the protocol, an authenticated SOK time is required in both the data source and the data sink so that it can be used for the creation and verification of sig- natures. If this is not yet available, the protocol must be performed for au- thentic time distribution.

[A: SOK_PROT_474]

Since a data source does not yet have an internal SOK time after it is started up, but still has to send protected data messages, it uses the constant value 0xFFFFFFFFFFFFFF as a substitute value for the internal authenticated SOK time for a period of 500 milliseconds. After this period, it uses the 64-bit representation of its internal SOK time.

[A: SOK_PROT_498]

This period applies separately for each SOK PDU ID and starts at the time of the first sending of the associated data message after the start of the data source.

[I: SOK_PROT_500]

For an initial SOK PDU ID for which the associated data message was sent for the first time at 100ms after startup of the data source, the time period thus extends from 100ms to 600ms after startup.
For another SOK PDU ID, for which the associated data message was sent for the first time 150ms after the start of the same data source, the time period thus extends to

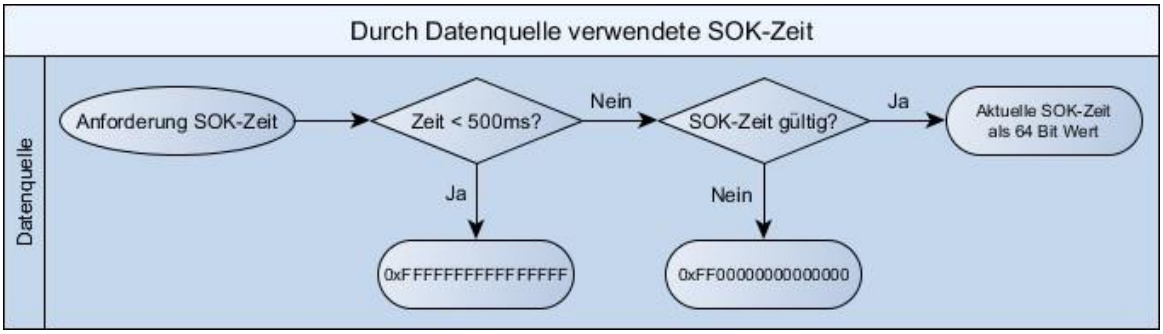
from 150ms to 650ms after startup. The period for the first SOK PDU ID remains unaffected.

[A: SOK_PROT_476]

If the data source still does not have an internal authenticated SOK time after the period from SOK_PROT_474 has expired, it uses the constant value 0xF00000000000 as a replacement value for the internal authenticated SOK time.

[I: SOK_PROT_502]

The following graphic illustrates the selection procedure for the current value of the SOK time of a data source. The value of "Time" corresponds to the time that has passed since the first verse of the corresponding data message.



[I: SOK_PROT_481]

As a consequence, SOK guarantees authenticity during the first 500 milliseconds after startup of an ECU, but cannot provide protection against replay attacks.

[I: SOK_PROT_484]

Furthermore, an SOK participant can always send messages protected by SOK, even if it does not have an authenticated SOK time. However, the value 0xF00000000000 cannot occur as an authentic SOK time. Thus, verification by a data sink will always fail after the 500 milliseconds have elapsed.

[A: SOK_PROT_137]

The data source generates a data message with an associated signature. The authenticated internal SOK time of the data source is included in the calculation of the signature.

[A: SOK_PROT_360]

The data source transmits the data message and the associated signature to the data sink(s).

[A: SOK_PROT_139]

The data sink verifies the signature for a maximum of four different possible time values: first for its current authenticated internal SOK time, then for the current internal au- thenticated SOK time minus 1, and finally for the current internal authenticated SOK time plus 1.

[A: SOK_PROT_140]

However, only times that correspond at least to the value of the SOK time with which the last successful verification for this SOK PDU ID took place are checked. If no successful verification has yet taken place, a maximum of all three times will be checked.

[A: SOK_PROT_141]

Furthermore, the times are not checked for which a successful verification has already been performed for the present counter value of the data message.

[A: SOK_PROT_475]

Furthermore, the data sink additionally checks the constant value 0xFFFFFFFFFFFFFF as SOK time for each data message for a maximum of 500 milliseconds after it has been started. Only after the verification of the data message on the basis of this constant value has failed, the data sink attempts a verification on the basis of the internal authenticated SOK time according to SOK_PROT_139 and SOK_PROT_477. After a verification based on the internal authenticated SOK time corresponding to SOK_PROT_139 has been successful, no more verification with the constant value is performed.

[A: SOK_PROT_477]

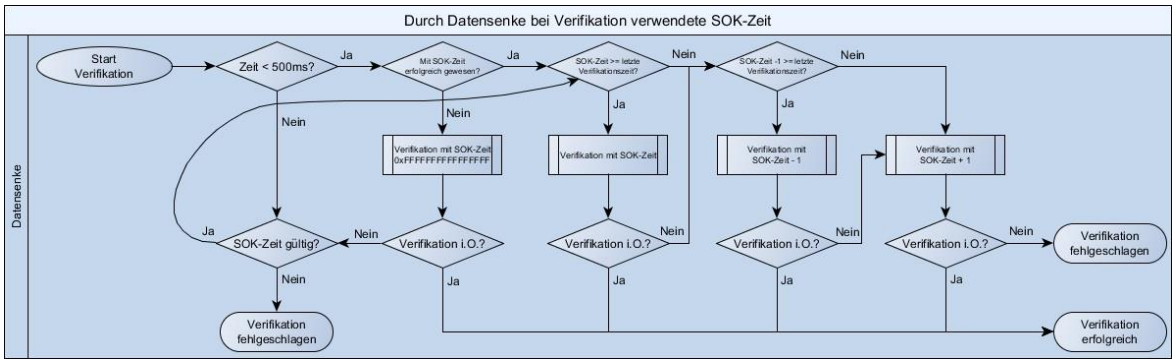
If the data source does not have an internal authenticated time after 500 milliseconds after the first reception of the data message, the signature verification automatically fails.

[A: SOK_PROT_478]

The time windows in SOK_PROT_475 apply per SOK PDU ID, i.e. they each start separately from the receipt of different data messages.

[I: SOK_PROT_504]

The following graphic illustrates the verification procedure of a data sink from the point of view of the selection of the possible values of the SOK time that can be used as an input value for the signature verification. The value of "Time" corresponds to the time that has passed since the first reception of the corresponding data message.



[A: SOK_PROT_142]

If the signature verification is successful for one of the checked times, the received data of the data message is authentic.

[I: SOK_PROT_479]

The requirements SOK_PROT_474, SOK_PROT_475, SOK_PROT_476 and SOK_PROT_477 ensure that two SOK participants can communicate for a maximum of 500 milliseconds from the start of communication even without an internal authenticated SOK time. After this time window has expired, all verifications on the data sink side fail.

2.3.3.5 Dataless authentication

[A: SOK_PROT_155]

With data-less authentication, only virtual protection messages are transmitted.

[I: SOK_PROT_359]

This protocol allows SOK participants to be authenticated without transmitting additional user data.

[A: SOK_PROT_156]

Data-less authentication can be used with the challenge-response protocol as well as with the authentic broadcast.

2.3.4 Cryptographic calculations

[A: SOK_PROT_161]

This section describes the cryptographic procedures necessary to calculate the signature.

2.3.4.1 Cryptographic signature

[A: SOK_PROT_163]

The following functions can be used to calculate signatures:
- SipHash24
- AES-CMAC

These functions are described in more detail in /1/.

[A: SOK_PROT_164]

For each SOK PDU ID, it is defined which function is used for signature generation . The function used in each case is referred to below as the signature function.

2.3.4.2 Signature calculation

[A: SOK_PROT_168]

The signature calculation of a data message is basically the same for all protocols. The difference is that for challenge-response protocols the challenge and for broadcast protocols the time and the counter value are used as input values.

[A: SOK_PROT_384]

The following steps are performed to execute the procedure ("||" describes the concatenation of byte values):
1. Construction of the input data as a union of the SOK PDU ID, the associated data message and the challenge or SOK time.
2. Processing of the entire data by the signature function

[A: SOK_PROT_385]

During signature generation for the challenge-response protocol, the following byte sequence is created:
DATA = <SOK-PDU-ID> || <Data message> || <Fresh value>
The SOK-PDU-ID is interpreted as a 16 bit long value.
The freshness value corresponds to the challenge in the case of the challenge-response protocol and to the 64-bit representation of the SOK time in the case of the authentic broadcast.

[A: SOK_PROT_390]

For signature calculation DATA is used as input vector for the signature function.

[A: SOK_PROT_171]

The signature of length L bytes are the first L bytes of the output vector of the signature function. The length L can be different for each SOK PDU ID.

2.3.4.3 Example vector

[I: SOK_PROT_489]

Configuration:
- Protocol: Challenge response
- SOK PDU ID: 274 (decimal) / 01 12 (hexadecimal)
- Signature function: SipHash24
- Signature length: 32 bit

[I: SOK_PROT_491]

Input parameters:
- Data message: 7C C7 8B 7A 57 C6 1F
- Challenge: 1A EF 95 9D AD 06 BD 05
- Cryptographic key: DF 2A 8B A6 5F B1 BC 72 E2 0C C0 F4 68 88 BA 90

[I: SOK_PROT_490]

Resulting processing:
- Input vector: 01 12 7C C7 8B 7A 57 C6 1F 1A EF 95 9D AD 06 BD 05
- Output vector: 67 DB 80 84 D8 00 16 ED
- Call number: 67 DB 80 84

2.4 SOK time management

[A: SOK_PROT_199]

The SOK time is a 56 bit counter that is incremented every 100ms.

[A: SOK_PROT_201]

The initial value of the SOK time is a random number and is generated by the time master. For this purpose, it generates a 56-bit random number whose highest bit is set to 0.

[I: SOK_PROT_202]

The initial value is thus in the range from 0 to 2^55 - 1. If the SOK time is stored in a 64-bit unsigned data type, this ensures that the SOK time does not overflow during the life cycle of a vehicle. In the implementation, it is therefore not necessary to check for corresponding overflows.

[A: SOK_PROT_217]

The time master generates a new SOK time at the beginning of each watch cycle.

[A: SOK_PROT_203]

The SOK time of the time master is distributed to all SOK participants via the time distribution protocol.

[A: SOK_PROT_205]

After a SOK participant has received an authentic time, it calculates the future time values itself.

[I: SOK_PROT_208]

In the following, "clock_current" denotes a counter that is incremented using an internal timer.

[A: SOK_PROT_204]

After successful verification of an authentic time message, an SOK station takes over the contained SOK time and stores it volatile as "time_current". Furthermore, it sets the value of clock_current to 0.

[A: SOK_PROT_209]

The SOK participant increments clock_current by 5 every 5 milliseconds.

[A: SOK_PROT_466]

As soon as clock_current reaches the value 100, the SOK participant increments time_current by 1 and resets the value of clock_current to 0.

[A: SOK_PROT_215]

The time master uses the same incrementing procedure, but uses its own generated SOK time as time_current.

[A: SOK_PROT_216]

The time master sends an unprotected time message after every 10 increments of its SOK time. This time message contains its current SOK time.
So the time interval between two unprotected time messages is one second and the difference of sent values of SOK time is 10.

[A: SOK_PROT_210]

An SOK participant must calculate the deviation of its internal SOK time from the received SOK time each time it receives an unprotected time message.

[I: SOK_PROT_211]

In the following, "time_received" denotes the received unprotected SOK time and "jit- ter" denotes the deviation between the SOK time of the SOK subscriber and the received unprotected time in milliseconds.

[A: SOK_PROT_212]

A SOK participant calculates the deviation between his SOK time and the received unprotected time according to the following calculation rule:
jitter = 100 * (time_current - time_received) - clock_current

[A: SOK_PROT_213]

If the absolute value of jitter exceeds 50ms, the SOK participant must execute the time distribution protocol and obtain a new authentic SOK time from the time master.

[A: SOK_PROT_214]

As long as the SOK participant has not received a new authentic SOK time from the time master , it continues to use its internal SOK time.

2.5 Implementation on different transmission channels

[A: SOK_PROT_485]

SOK supports the CAN, CAN-FD, FlexRay, Ethernet and ISO-TP transmission channels.

[A: SOK_PROT_486]

SOK does *not* support the UDS, LIN and BAP transmission channels.

[A: SOK_PROT_495]

Transmission over different transmission channels is possible as long as the PDU containing the data message and the protection message is not changed on the entire transmission path.

[I: SOK_PROT_454]

The following minimum signature lengths are recommended depending on the transport channels used:

- CAN: 32 bit
- CAN-FD: 40 bit
- Ethernet, FlexRay: 64 bit
- ISO-TP: According to the underlying bus technology

If a protected message is transmitted over several communication channels, the shortest minimum length applies to this message.

[I: SOK_PROT_468]

If, for example, a message is transmitted on both CAN and CAN-FD, the minimum length defined for CAN applies.

[A: SOK_PROT_401]

If, in the case of CAN or CAN-FD, the protection message is transmitted in a separate PDU to the data message, its CAN ID should be selected so that its priority during bus arbitration is similar to that of the PDU of the associated data message.

[A: SOK_PROT_425]

When protecting data sent via ISO-TP, the data to be protected is protected as a whole before it is segmented by ISO-TP. The data message and its associated protection message are transmitted in the same PDU. This PDU is transmitted via the corresponding TP channel.

2.6 Implementation in control units

[I: SOK_PROT_219]

The functionality of SOK is implemented by the AUTOSAR SecOC module in version 4.3 together with the standard software "SOK Freshness Manager".

[I: SOK_PROT_253]

The implementation of SOK by SecOC and the SOK Freshness Manager is described in a separate document.

[A: SOK_PROT_447]

ECUs that do not implement AUTOSAR or for which the SecOC is not compatible with the AUTOSAR version must either reproduce the protocol and the remaining functions of the SOK Freshness Manager (diagnostics, etc.) themselves or integrate the SecOC module and the SOK Freshness Manager via a compatibility layer.

[A: SOK_PROT_276]

The configuration of the different SOK PDU ID of the SOK subscribers is date of the protection class "Authentic" according to /2/.

[A: SOK_PROT_277]

The cryptographic keys used by SOK are data of the protection classes "Confidential" and "Authentic" according to /2/.

[A: SOK_PROT_278]

All intermediate values in cryptographic calculations (signature, encryption, random numbers) are data of the protection classes "Confidential" and "Authentic" according to /2/.

3 Appendix

3.1 Change documentation

[A: SOK_PROT_45]

The change documentation listed here only gives a rough overview of the changes made between versions.

[I: SOK_PROT_3]

Table 1: Change documentation

Date	Version	Chapter	Description	Author/OE
22.09.2015	1.0	-	Initial creation	Chache, Alexander (EEKS)
25.11.2015	1.1	-	Spin-off software module; Revision input vectors	Chache, Alexander (EEKS)
15.04.2016	2.0	-	Revision due to implementation by AUTOSAR SecOC; Change of MAC from proprietary to HAIFA-based design	Chache, Alexander (EEKS)
19.04.2016	2.1	-	Graphical illustration for the various protocols; description for conversions to different transmission channels; definition of ChaCha20/12	Chache, Alexander (EEKS)
19.07.2016	2.2	-	Incorporation Reviews Audi; Switch to SipHash24	Chache, Alexander (EEKS)
02.08.2016	2.3	-	Restructuring and error corrections; inclusion of FlexRay; transfer of formulation "bus message" in "PDU" and "not authentic" in "unprotected	Chache, Alexander (EEKS)
09.09.2016	2.4	2.3.3.4	Definition behavior between Startup and time reception	Chache, Alexander (EEKS)
26.03.2017	2.5	-	Various corrections; Modification startup behavior	Chache, Alexander (EEKS/5)
19.04.2017	2.6	2.4	Correction of sign error for Jitter calculation	Chache, Alexander (EEKS/5)
06.06.2017	2.7	2.3.3.4	Correction of substitute values for SOK time for authentic broadcast	Chache, Alexander (EEKS/5)
17.07.02017	3.0	-	Revision for QLAH process	Chache, Alexander (EEKS/5)

3.2 Referenced documents

[I: SOK_PROT_5]

Table 2: Referenced documents

Ref.	Document/ Source	Version
/1/	Group standard VW 80180-1: Cryptographic algorithms and methods for use in control units, systems, functions and supplier-specific scopes	-
/2/	Group standard VW 80180-2: Implementation of protection target classes for data worthy of protection and cryptographic protection measures	-

4 Confidentiality notice

[A: SOK_PROT_25]