INTERNATIONAL STANDARD

ISO 15764

First edition
2004-08-15

Road vehicles — Extended data link security

Véhicules routiers — Sécurité étendue de liaison de données

Reference number
ISO 15764:2004(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 15764 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

Introduction

This International Standard is intended initially to supplement ISO 15031-7 in extending the security provisions of, and facilitating access to, remote sources of sensitive data. PC-based external test equipment based on ISO 15031-4, modified to incorporate the facilities described herein, could then access the vehicle using the challenge-response provisions of ISO 15031-7, and the remote source using the extended security offered by the present document.

While this would fully protect the transmission of data from the remote source to the external test equipment, it would leave the data between the external test equipment and the vehicle unprotected, which might be acceptable in a controlled environment. Where the electronic control unit (ECU) is capable of supporting the encryption/decryption burden of full PKI infrastructure, this International Standard offers end-to-end security in an open system in which the participants are not previously known to each other. It also includes provisions for end-to-end security in a closed system where the symmetrical key is established with both participants prior to use and the computing burden is reduced.

It is anticipated that this International Standard will be used, for example, by a vehicle manufacturer to send data to a franchised dealer to enable the programming of an unprogrammed stock ECU or to release immobiliser re-setting codes to approved users. Ultimately, it would protect over-air messages sent directly to a vehicle for software corrections, service interrogation or other remote services.

In the vehicle manufacturer's case, the present document extends the provisions of ISO 15031-7 in respect of data link security to cover the access to data remote from the vehicle, such as that contained in a manufacturer's database — extensions which allow for control and monitoring of such access and thus enhance the security of the data itself. No matter whether the amount of data is small, as in gaining entry to the vehicle, or large, as in a complete code download for powertrain control, it establishes uniform practice for protecting vehicle modules from unauthorized intrusion through a vehicle data link. The security system described represents a recommendation for motor vehicle manufacturers while providing the flexibility for them to tailor their systems to their specific needs.

The vehicle modules addressed are those able to of have solid state memory contents accessed through a data communication link. Improper memory content alteration could potentially damage the electronics or other vehicle components; or risk the vehicle compliance to government legislated requirements or the vehicle manufacturer's security interests. Improper access to secure information could compromise security and privacy of the vehicle or operator.

Other applications are envisaged. In many cases there will be a need for secured data transmission on internal vehicle communication networks such as CAN (controller area network), and between after-market equipment on the one hand, and components of the initial vehicle electronics or other-after market equipment on the other. In particular, this document can be used to enable a tachograph reader to authenticate the data sent by the on-vehicle recorder of the tachograph, for example, in tolling applications. It defines the procedures to establish and use a secured data link and the specific security related data elements. It is communication protocol independent. Another possible implementation is given by the SecuredDataTransmission (84 hex) service defined in ISO 14229-1 on diagnostic services, with whose defined properties its specification of data elements is in line.

Road vehicles — Extended data link security

1 Scope

This International Standard describes an extension of data link protocols for enhancing the security of data transfers between electronic control units (ECUs) connected by a communication network used in road vehicles. It is based on cryptographic methods that include encryption, digital signatures and message authentication codes (MACs). It provides a description of services to establish ECUs as trusted parties in respect of one another and to protect against specific threats. It is applicable to all data links between pairs of ECUs capable of storing and processing secret data so that unauthorized third parties are denied access to it. Parameters are provided to enable the level of security in the data link to be selected.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 3779:1983, *Road Vehicles — Vehicle identification number (VIN) — Content and structure*

ISO 3780:1983, *Road vehicles — World manufacturer identifier (WMI) code*

ISO/IEC 8824-1, *Information technology — Abstract Syntax Notation One (ASN.1) — Specification of basic notation — Part 1*

ISO/IEC 8825-1, *Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) — Part 1*

ISO/IEC 9594-8, *Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks — Part 8*

ISO/IEC 9797-2:2002, *Information technology — Security techniques — Message Authentication Codes (MACs) — Part 2: Mechanisms using a dedicated hash-function*

ISO/IEC 10116, *Information technology — Security techniques — Modes of operation for an n-bit block cipher*

ISO/IEC 10118-3, *Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash-functions*

ISO 11898 (all parts), *Road vehicles — Controller area network (CAN)*

ISO 14229-1, *Road vehicles — Unified diagnostic services (UDS) — Part 1: Specification and requirements* [1)]

ISO 14230-4, *Road vehicles — Diagnostic systems — Keyword Protocol 2000 — Part 4: Requirements for emission-related systems*

1) Under preparation.

ISO 14816, *Road transport and traffic telematics — Automatic vehicle and equipment identification — Numbering and data structure* [2)]

ISO 15031-3, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 3: Diagnostic connector and related electrical circuits, specification and use*

ISO 15031-4, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 4: External test equipment* [3)]

ISO 15031-7, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 7: Data link security*

ISO 16844-1 (all parts), *Road vehicles — Tachograph systems* [3)]

ISO/IEC 18033-3, *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers* [1)]

IETF RFC 2437, *PKCS #1: RSA Cryptography Specifications*, Version 2.0, October, 1998

IETF RFC 2459, *X.509 Internet Public Key Infrastructure Certificate and CRL Profile*

SAE J1939 (all parts), *Recommended Practice for a Serial Control and Communications Vehicle Network*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
certification authority
CA
centre trusted to create and assign public key certificates or, optionally, which may create and assign keys to the entities

[ISO/IEC 11770-1:1996, definition 3.2]

3.2
client
entity initiating the message exchange by sending some request to the other entity

3.3
confidentiality
property that information is not made available or disclosed to unauthorized individuals, entities or processes

[ISO 7498-2:1989, definition 3.3.16]

3.4
data integrity
property that data has not been altered or destroyed in an unauthorized manner

[ISO 7498-2:1989, definition 3.3.21]

2) To be published. (Revision of ISO/TS 14816:2000)

3) To be published.

3.5
delay time
DT
time period inserted between access attempts

3.6
data origin authentication
corroboration that the source of data received is as claimed

NOTE Adapted from ISO 7498-2:1989.

3.7
digital signature
data appended to, or a cryptographic transformation of, a data unit that allows the recipient of the data unit to prove the origin and integrity of the data unit and protect against forgery, e.g. by the recipient

[ISO/IEC 9798-1:1997, definition 3.1.3]

3.8
eavesdropping
activity leading to loss of confidentiality, in which a third party obtains data sent between the trusted electronic units, knowledge of which it is not entitled to possess

3.9
entity authentication
corroboration that an entity is the one claimed

[ISO/IEC 11770-2:1996, definition 3.1.2]

3.10
false access attempt
FAA
error in the received signature, message authentication code or previously unused number

3.11
hash-code
string of bits which is the output of a hash-function

[ISO/IEC 14888-1:1998, definition 4.7]

3.12
hash-function
function which maps strings of bits to fixed-length strings of bits, such that it is computationally infeasible to find an input which maps to this output or a second input which maps to the same output

NOTE Adapted from ISO/IEC 9797-2:2002.

3.13
key
sequence of symbols that controls the operation of a cryptographic transformation

EXAMPLE Encipherment, decipherment, cryptographic check function computation, signature generation or signature verification

[ISO/IEC 11770-1:1996, definition 3.5]

3.14
manipulation
changing by a third party of data transferred between trusted electronic control units

3.15
masquerade
sending of data by a third party under the pretence that it originates from a trusted electronic control unit

3.16
message authentication code
MAC
string of bits which is the output of a MAC algorithm

[ISO/IEC 9797-1:1999, definition 3.2.5]

3.17
MAC algorithm
algorithm for computing a function which maps strings of bits and a private key to fixed-length strings of bits, such that for any key and any input string the function can be computed efficiently, and that for any fixed key, and given no prior knowledge of the key, it is computationally infeasible to compute the function value on any new input string, even given knowledge of the set of input strings and corresponding function values, where the value of the ith input string may have been chosen after observing the value of the first $i-1$ function values

NOTE Adapted from ISO/IEC 9797-1:1999, 3.2.6.

3.18
private key
key of an entity's asymmetric key pair intended to be used only by that entity

NOTE Adapted from ISO/IEC 11770-1:1996, 3.13.

3.19
public key
key of an entity's asymmetric key pair which can be made public

[ISO/IEC 11770-1:1996, definition 3.14]

3.20
public key certificate
public key information of an entity signed by the certification authority and thereby rendered unforgettable

[ISO/IEC 11770-1:1996, definition 3.15]

3.21
replay
resending by a third party of data that was transferred earlier between trusted units, under the pretence that it is “fresh”

NOTE This includes the illicit copying of valuable software destined for one ECU into another ECU.

3.22
repudiation
action by which one of the entities participating in a data transfer afterwards denies having originated the generated data

3.23
secret key
key, established for use by the client and the server, to a message sequence for symmetric encryption and decryption

3.24
server
entity towards which the request of the client is directed

NOTE The server could require information from the client in order to approve the exchange and to react to the request in the intended manner.

3.25
RSA cipher
public key encryption algorithm named after its inventors, Rivest, Shamir and Adleman

4 Symbols and abbreviated terms

4.1 General

AES	advanced encryption standard in accordance with ISO/IEC 18033-3
ASN.1	abstract syntax notation one in accordance with ISO/IEC 8824-1
CA	certification authority
CAN	controller area network
DER	distinguished encoding rules in accordance with ISO/IEC 8825-1
ECU	electronic control unit
IV	initializing value
OID	object identifier (ASN.1 type)
SHA-1	secure hash algorithm, Revision 1 (equivalent to dedicated hash-function 3 according to ISO/IEC 10118-3)

4.2 Notation used in the message sequence specified in Clause 6

Cert$A$	certificate for the public key of entity A
Sig$_A$[X]	signature of entity A on data X (computed using A's private signature key), with this notation intended to include a copy of the data X
[X]$_A$	asymmetric encryption of data X using entity A's public encryption key
$e_K(X)$	symmetric encryption of data X using the secret key K
MAC$_n$	MAC included in Message n
n	message number
$f_{K'}(X)$	MAC computed on data X using the key K
K'	is a variant of K used for MAC calculation (the same key should not be used for encryption and MAC calculation)
Vx	version number field of the ISO 15764 edition used, where x is initially 1 and increases with each technical revision of the standard (see 7.6.8)
N_1	previously unused number generated by the client, which should not be confused with the version number (see 5.3.4)
N_2	previously unused number generated by the server
ID$_A$	unique identifier of entity A, also included in Cert$A$ (see 7.6.3)
APar	administration parameter indicating the selection of optional parameters and security related options
IV	initializing value, a random number used for cipher block chaining in the symmetric encryption procedure
||	concatenation of two data elements

$Data_n$ the data contained in message n

C client

S server

5 Secured data link configuration

5.1 General

The services provided here are independent of the communication protocol between client and server. It is up to individual applications and standards to define how these services are to be encoded at the lower protocol layers. As an example each transaction described between client and server could require multiple transactions at the lower layer protocol.

The standard offers a procedure for securing the data to be transmitted, based on a message sequence with some mandatory elements and some options. A choice can be made among these options according to the security requirements for a given application. The options relate to

— audit trail facilities to assist in the validation of both client and server and to provide the opportunity of a detailed record of the exchange,

— messages to be included to prevent repudiation,

— encryption to prevent loss of confidentiality,

— time-out between sending a request message and receiving the response to it, and

— use of digital signatures to establish a secure connection.

Any examples given are purely to assist understanding and do not suggest that the services must be implemented in these situations.

5.2 Architecture

In its simplest form the system to be covered is shown in Figure 1.

Figure 1 — Overview of system to be protected

It consists of at least one module, the client, communicating with the server through a communication chain. Since, as a minimum, the client must make a request to which the server will respond, two-way point-to-point communication is required. The client may be an entity which requires information held by the server. The server may require further information from the client before permitting the transfer. This simple system can be extended to include further modules at each end. Full end-to-end protection can be provided where encryption/decryption facilities are in place at each end and the intermediate modules are transparent.

In a vehicle module re-programming example, the module in question would be connected to a scan tool which would then be connected via a secure link by land line or mobile telephone to the remote database. Encryption/decryption could be in the scan tool or in the module as required. Details are given in 8.1.

In a tachograph example, the vehicle unit is simply connected to the recording unit by a CAN (controller area network) network. Details are given in 8.2.

In a closed system, where the communicating devices are all under the control of a single entity (e.g. a vehicle manufacturer), certain simplifications to the message exchange are possible. Details are given in 8.3.

For a secured data transmission according to this standard, the data to be transmitted has to be processed in a defined manner by the sender before sending it, and has to be processed again by the receiver before using it. These activities take place within the security sub-layer, according to the layers introduced in the open system interconnection (OSI) reference model. The security sub-layer lies within one of the upper layers specified in the OSI model. It is assumed that the applications on the client and server side decide on the protection type needed, and that the corresponding protection measures, including the management of cryptographic keys within the client and the server, are under the responsibility of the security sub-layer.

5.3 Protection

5.3.1 General

This standard addresses the means of securing the communication path between the client and the server. In doing so it specifies a set of security mechanisms which are designed to provide certain identified security services. These security services have, in turn, been designed to address certain identified threats.

5.3.2 Security threats and services

The security services provided by the scheme specified in this standard address the identified security threats as follows.

a) **Masquerade** is prevented at the set-up of a message sequence by use of an entity authentication service. Substitution of a malicious third party once the message sequence has been established is prevented by the chaining of the messages using a key established as part of the entity authentication service.

b) **Replay** is prevented by two different security services:

   1) the initial exchange of messages is protected against replay using the entity authentication service;

   2) replay of subsequent messages is prevented through the use of a data integrity service.

c) **Eavesdropping** is addressed by the provision of a confidentiality service.

d) **Manipulation** is not prevented but is detected by the use of a data integrity service.

e) **Repudiation** is prevented through the provision of a non-repudiation service.

5.3.3 Use of cryptography

5.3.3.1 General

Before considering the security mechanisms in detail, the following subclauses briefly consider the use of cryptography in the mechanisms specified in this standard.

5.3.3.2 Encryption

Encryption is used to provide confidentiality for transmitted data. Encryption techniques can be divided into symmetric encryption algorithms and asymmetric encryption algorithms. Symmetric algorithms use a shared private key, where the same secret value can be used by both the message sender to encrypt the data and the message recipient to decrypt the data. Asymmetric encryption algorithms require two different values to be used for encrypting and decrypting. These are referred to as public and private keys. The public key and the

algorithm for its use are open to all and can be used to encrypt data, which can only be decrypted using the private key (which is kept secret by its owner).

In this standard the initial exchange is based on the use of public and private keys. Subsequent messages will be encrypted using a symmetric encryption algorithm, whose key is transferred during the initial identification process. This key will be referred to as the secret key. In cases where the two parties already share a secret key, the two messages making up the initial exchange may be omitted.

The public and private keys used to protect the initial exchange will be longer than the secret key and their use will require more computing power. Their use is therefore restricted.

5.3.3.3 Digital signatures

Digital signatures are used to confirm that the signed data has been produced by the entity claimed, and has not been modified since the signature was generated. This is done in such a way that evidence is provided to the verifier which may prevent repudiation of the signed data string by the originator. Signature algorithms use pairs of keys in a similar way to asymmetric encryption algorithms, except that in the case of a signature the private key is used to sign a message, and the public key is used to verify it.

Signatures are used in the scheme specified in this standard to implement the entity and data authentication service, in which an entity proves its identity by providing a signed data string which only it can produce, and which can be verified as genuine and “fresh” by the recipient. Signatures are also used within this standard to create public key certificates, described in 5.3.5.

In summary, signatures are employed by three different types of party within this standard: clients, servers and certification authorities.

5.3.3.4 Message authentication codes

Message Authentication Codes (MACs) are used to guarantee the origin and integrity of messages. MACs are symmetric cryptographic techniques, i.e. they rely on the use of shared secret keys. The sender uses the secret key to compute the MAC, which is appended to the message to be protected. The receiver uses the same secret key with the received message to re-compute the MAC, which is compared with the transmitted value. If the two values agree then the receiver knows that the message has not been tampered with.

Unlike digital signatures, MACs do not provide any protection against the repudiation of messages. This is because both sender and receiver have the same secret key, and hence the receiver could forge a message and its MAC.

5.3.4 Previously unused numbers

The exchange of messages used to provide the entity authentication service makes use of values called *previously unused numbers* and labelled N_1 and N_2 (in fact these values are bit-strings or octet-strings). The client chooses N_1 and sends it to the server, and the server chooses N_2 and sends it to the client. The inclusion of these numbers in responses subsequently received by client and server guarantees that these responses have been newly generated.

It is important that the client chooses N_1 so that it is different from all previous values chosen for N_1 during the lifetime of the current public/private key pair (or, in cases where messages 1 and 2 of the message exchange are omitted and a previously established key K is used, during the lifetime of this pre-established key K). Similar restrictions apply to the server’s choice for N_2 (although it is immaterial whether the same value is used for N_1 and for N_2). This can be achieved in many ways. One possibility is for N_1 and N_2 to be chosen at random from all bit-strings or octet-strings of the appropriate length. Another is to use a counter to generate them, although the counter value must be stored securely so that a power failure or other event cannot cause the counter value to be re-initialized. Conveniently, these numbers might be used as job numbers for administrative purposes.

5.3.5 Certificates and certification authorities

Distribution of public keys (as required for asymmetric encryption and digital signatures) is a non-trivial exercise, since, although public keys do not need to be kept secret, the user of a public key needs to be sure that it is a valid key for the entity to whom it claims to belong. This is addressed here by the establishment of certification authorities (CAs), having their own public/private key pair for use with a digital signature algorithm. A CA will sign public key certificates, containing a public key for an entity, an identifier for that entity, an expiry date for the certificate, and other relevant information.

Any other entity which has a trusted copy of the CA's public key can verify the signature on such a certificate, and thereby obtain a trusted copy of the public key of the subject of the certificate. It is therefore necessary for all parties who wish to verify a public key certificate to possess a trusted copy of the public key of the CA that created the certificate. The necessary CA public keys can be put in place by manual means.

It may be necessary to mark certain public keys as being invalid before the expiry date of the certificate in which they have been distributed. This may be necessary, for example, if the corresponding private key has been compromised (or even if compromise is simply suspected). In such an event all certificates containing this public key shall be revoked, and all users of certificates shall be informed of which certificates have been revoked. A revoked certificate shall always be rejected as invalid. The means by which the distribution of revocation information is achieved is outside the scope of this standard. Whenever verification of certificates is specified in this standard, then this automatically includes the checking of certificate revocation information, where this is available to the verifier.

5.3.6 Security mechanisms

5.3.6.1 General

The mechanisms specified in this standard are designed to provide the specified security services. They have been designed on the basis of the following assumptions.

— The generation, distribution and storage of security keys is secure.

— The encryption/decryption process is protected against manipulation by unauthorized parties.

— The hardware of the client and the server is resistant to reading out or rewriting stored information, or at least all stored information with security relevance (memory containing secret and private keys, the secure part of the operating system, the loader, cryptographic routines, etc.).

— The CA(s) used to sign the public key certificates are known to the parties needing to verify the certificates, where being "known" implies possession of a trusted copy of the CA's public key needed to verify the certificate.

— The equipment using the mechanisms (e.g. the tachograph) has been through an appropriate certification process.

— The cryptographic algorithms in use are known to, and trusted by, the communicating parties.

— The identities of the client and server tools are recognized by each other.

5.3.6.2 Entity authentication service

The entity authentication service is provided by use of an entity authentication mechanism. In the full message sequence case, this consists of an exchange of messages in which each party signs some data which can then be checked by the other using the first party's public key. The signed data includes previously unused numbers, which are used to guarantee that the messages are "fresh". The public keys necessary for verification may either be in place by an arrangement between the parties before a request for data, or may be transmitted along with the request in the form of a public key certificate, provided the certificate is signed by a CA whose public key is known to the recipient. The entity authentication mechanism also serves as an authenticated key establishment mechanism, used to establish a secret key which is then employed to protect the integrity and confidentiality of subsequently exchanged messages.

In the case where the full message sequence is not used, and a previously established secret key is employed, the entity authentication mechanism consists of an exchange of messages containing MACs, which can be verified using the shared secret key. The MACs are computed on sequences of data including previously unused numbers, which are used to guarantee that the messages are “fresh”.

5.3.6.3 Confidentiality service

The confidentiality service is provided by the use of data encryption, as applied to the sensitive parts of the contents of exchanged messages. Intercepted data is rendered meaningless by encryption using a secret key. This key is either generated by the server and sent to the client at the start of the message exchange, or is an existing shared secret between client and server. Data encryption is also used to provide the confidentiality service for the secret key (if it is transferred from server to client).

5.3.6.4 Data integrity and data origin authentication services

The data integrity and data origin authentication services are provided as follows. The origin and integrity of the first two messages are protected using a digital signature. The origin and integrity of all subsequent messages are protected using a MAC generated using a secret key established as part of the entity authentication service. The integrity of the sequence of messages is guaranteed by the fact that the MAC of each message includes among its inputs the MAC of the previous message in the sequence. An alternative approach to the provision of these services omits the first two messages, and establishes these services using MACs only (where the MACs are computed using a pre-established shared secret key).

5.3.6.5 Non-repudiation service

The optional non-repudiation service is provided by a combination of different mechanisms. Repudiation of either of the first two messages may be prevented by the retention of these digitally signed messages by either party. Messages subsequently exchanged during a session are protected against repudiation by retaining the entire message sequence. Protection for this message sequence may be provided by requesting the exchange of signatures.

a) The two session termination messages closing a message sequence are both signed, and these signatures, in combination with the sequence of MACs computed during the session, protect all the messages in the session against repudiation.

b) At any time one party may request the other party to send a signed message. This signature, in combination with the sequence of MACs computed during the message sequence, protects all the messages previously sent during the sequence against repudiation by the party that generates the signature.

5.4 Audit trail

The audit trail is provided as a deterrent in that a malicious user might be deterred if it were known that the transaction was recorded and traceability to the user and the time of day was in place.

Furthermore, if transmitted sufficiently early in the transaction, the audit trail information can be used to further qualify an application for information. Thus data can be withheld at certain times of the day or withheld completely from a client tool reported as stolen.

6 Message content

6.1 General

The prerequisite for securing communications is that all entities can be confident that the other entities involved have the right to access the functions and data concerned. This may be achieved by the sequence described below for each connection. In the case of a vehicle, an interface and a remote database, the sequence would be required for the vehicle-interface connection and the database-interface connection. It is

important to note that not only does the vehicle need to be convinced that it is talking to an authorized tool, but also that the tool needs to be confident that it is talking to an authorized vehicle.

Since it is the client who requires the server to perform some task (e.g. to send some data), it is the client who must initiate the sequence and it is for the server to respond. However, since it is the server who is responsible for the confidentiality of the information, it is the server that will select the secret key and check the security parameters for the interchange (if a pre-established key is not used).

6.2 Message sequence

6.2.1 General

In order to establish the secured data link and exchange data on it the following sequences are required. The sequences are described as a series of messages between two entities: the client and the server. The first two messages within this sequence (Message 1 and Message 2 specified in 6.2.2 and 6.2.3) are used to set up the secured data link and are optional; they may be excluded only if the client and server already share a secret key K. In such an event, the message sequence will commence with Message 3, and the pre-established shared key K shall be used in the computation of these messages. In certain applications, it is also possible to send only Messages 1 and 2. In such a case (and only in such a case) it is permitted to simplify the structure of Messages 1 and 2, in accordance with 6.2.2 and 6.2.3.

If the same pre-established secret key K is used to protect a number of message sequences, then care should be taken to ensure that risks do not arise from repeated use of the same key. Such risks are in particular associated with use of the 3DES encryption algorithm (see 7.6.4); hence, in cases where the same secret key K is used to protect multiple message exchanges, use of the AES encryption algorithm is recommended.

If a secured link was set up using Messages 1 and 2, and if it is no longer trusted by either the client or the server, then the message sequence shall be terminated and there is the option to set up a new message sequence, starting with Message 1 again. The sequence termination procedure is specified in 6.2.7. For the set-up of the new secured link there can be a need to change the public and the private Key of the client or the server, together with the corresponding certificate. For instance, this will be the case if the certificate expired or was revoked. In any case, the new message sequence will be based on a new secret key that is different from the old one.

For some data elements contained in the messages, there are several options. The coding of the data elements must be such that the receiver can identify which options were taken.

In any one message sequence, messages must alternate between client and server. The messages with odd numbers are from the client to the server and are entitled request messages. The messages with even numbers are from the server to the client and are entitled response messages. Parallel message sequences are permitted, where supported by lower layer protocols, in which the roles of the client and server may be reversed.

NOTE If the implementation of the secured data link is according to ISO 14229-1 then each message of the sequence forms the content of a securityDataRequestRecord or securityDataResponseRecord parameter of the SecuredDataTransmission (84 hex) service, respectively.

The following messages use the notation described in Clause 4.

6.2.2 Step 1: Secured Link Set-up Request

The client shall send the following message to the server

Message 1: Sig_C [APar || Vx || ID_S || N_1 || [Data_1]$_S$] || Cert$C$

where

N_1 is a “previously unused number”, which is retained by the client at least until Step 3 is complete;

Cert$_C$ is the public key certificate of the client, which may optionally be replaced by ID$_C$ (e.g. if the server already has the certificate for this client);

|| denotes concatenation of two data items.

Data$_1$ might comprise

— Audit Trail Information;

— Application Specific Request function.

The presence of an application specific request function in Message 1 is optional and should not be considered if this request function needs protection against replay attacks, as such a protection is not provided in this message.

The indicated encryption of Data$_1$ is optional, i.e. in the above message [Data$_1$]$_S$ may be replaced by Data$_1$. For encryption to be possible, the public key of the server must be available to the client at the time this message is constructed.

Furthermore, if Message 2 is to be the final message in the exchange and the full set of security services are not required, then the signature computation, the identifier of the server and the value N_1 may also be omitted. In such a case, Message 1 will simply consist of the string

APar || Vx || Data$_1$.

where Data$_1$ must comprise an application specific-request function.

NOTE The use of this special abbreviated message exchange is discussed further in 8.3.

Each of the data parameters in the list is described in more detail in 7.6.

6.2.3 Step 2: Secured Link Set-up Response

The server will first verify the client's certificate (Cert$_C$), if present. This is achieved by, first, verifying the signature on the certificate using the trusted public key of the CA which signed this certificate and, second, checking the contents. The server will then use the client's public key (either obtained from the certificate or previously held) to verify the signature on Vx || ID$_S$ || N_1 || [Data$_1$]$_S$. If this verification process fails, then an *exception handling* applies. If the verification process succeeds, then the server will verify that the indication of its identity is correct; if this check fails, then the exception handling applies. See 6.4 for further details of exception handling.

At this point the server cannot be certain that the received message is not a replay of a previously transmitted valid message. Hence the server shall not react to the receipt of this message with any action that is critical with respect to replay (e.g. change its current configuration).

If the verification process succeeds, then the server may, at its discretion, check the Audit Trail Information and Request in order to approve the client, recording such information as it wishes.

If the option of omitting the signature and the value N_1 is followed, then the server cannot perform any checking of Message 1. This means that, if the message exchange proceeds, the server will be responding to an unverifiable request. Such a process shall only be permitted where the security policy in force explicitly allows this.

The server will then send the following message to the client:

Message 2: Sig$_S$ [APar || ID$_C$ || N_1 || N_2 || [K || IV]$_C$ || e_K(Data$_2$)] || Cert$_S$

where

K is a secret key chosen by the server (which shall be retained by the server for the duration of the message sequence)

N_1 is the “previously unused number” recovered by the server from Message 1. N_1 shall be retained by the server at least until Step 4 is complete

N_2 is a “previously unused number” (which shall be retained by the server at least until Step 4 is complete)

ID_C is the unique identifier of the client as sent directly or as part of the client's certificate in Message 1

Cert$S$ is the public key certificate of the server. Note that this may optionally be omitted (e.g. if the client already has the certificate for this Server).

$Data_2$ might include

— Audit Trail Information

— Application Specific Function Response (e.g. request for a vehicle challenge in the example according to Clause 0)

The encryption of $Data_2$ is optional, i.e. in the above message $e_K(Data_2)$ may be replaced by $Data_2$.

If Message 1 omits use of the signature function, then the signature in Message 2 shall be computed solely as a function of an unencrypted version of $Data_2$ (i.e. ID_C, APar, N_1, N_2 and $[K]_C$ shall be omitted). In such a case Message 2 shall always be the final message of the exchange.

6.2.4 Step 3: First Secured Data Transmission Request

The client will first verify the server’s Certificate (CertS), if present. This is achieved by, first, verifying the signature on the certificate using the trusted public key of the CA that signed this certificate and, second, checking the contents. The client will then use the server’s public key (either obtained from the certificate or previously held) to verify the signature on ID_C || APar || N_1 || N_2 || $[K]_C$ || $e_K(Data_2)$. If this verification process fails then an exception handling applies. If the verification process succeeds then the server will verify that the indication of its identity is correct; if this check fails then the exception handling applies. See 6.4 for further details of exception handling.

An exception to the above process will occur if Message 1 omitted any cryptographic protection. In such a case the signature on $Data_2$ will be verified, and this will conclude the message exchange.

If the verification succeeds, then the client will next compare the value N_1 in the message with the value sent to the server; if the two do not agree, then the exception handling applies. If the check succeeds, then the client will decrypt $[K]_C$ using its private key to obtain the secret key K. The recovered value of K can then be used to decrypt $Data_2$ (if it is encrypted).

The client might wish to time the delay between sending the message in Step 1 and verifying the response in Step 3. If this delay exceeds a certain time-out value then the client might wish to reject the message sequence. Such a procedure could be used to deal with accidental or malicious insertion of delays into the communications path in environments where such delays could cause undesirable effects.

The client shall send the following message to the server:

Message 3: APar || N_3 || $e_K(Data_3)$ || MAC_3

where

$$MAC_3 = f_{K'}[N_2 \,||\, N_1 \,||\, N_3 \,||\, e_K(Data_3)]$$

where

N_1 is the “previously unused number” sent by the client in Message 1;

N_2 is the "previously unused number" sent by the server in Message 2;

N_3 is a "previously unused number" selected by the client (which is retained by the client at least until Step 5 has been completed).

NOTE N_3 is only included in the message and in MAC_3 if Messages 1 and 2 are omitted and K is a pre-established secret key. In this case, N_1 and N_2 are not used in the MAC_3 calculation.

Since the MAC mechanism is only employed from Message 3 onwards, there can be no MAC_1 or MAC_2. However, the MAC in Message 3 is labelled MAC_3 for consistency with the message number.

N_2 is not explicitly returned by the client; it is sufficient for the server to verify that N_2 was used to compute MAC_3.

If Messages 1 and 2 are omitted, then the version number field (Vx) will not be sent. In such a case it is expected that the version number will be implicitly associated with the pre-established shared secret *K*, and hence will not need to be transmitted.

6.2.5 Step 4: First Secured Data Transmission Response

The server will first decrypt e_K (N_3 || $Data_3$) using the secret key K to recover $Data_3$ and — if Messages 1 and 2 are omitted — N_3. The server will next verify MAC_3 using the secret key K and the stored values of N_1 and N_2 (if Message 1 and Message 2 are omitted then the recovered value of N_3 shall be used). If MAC_3 is not correct then an Exception Handling as specified in Clause 6.4 applies.

If Messages 1 and 2 were sent and if the MAC is found to be correct, then the entity authentication process is complete and the client and server have verified each other's presence. The server can now be certain that the received messages are not replays of previously transmitted valid messages. Hence, unlike at Step 2, the server can react to the receipt of this message with actions that are critical with respect to replay (e.g. change its current configuration).

If Messages 1 and 2 are omitted, then at this point the server and client have not authenticated one another. That is, the server cannot be certain that the received message is not a replay of a previously transmitted valid message. In this case, the server shall not react to the receipt of this message with any action that is critical with respect to replay (e.g. change its current configuration).

The server shall send the following message to the client:

Message 4: APar || N_4 || e_K ($Data_4$) || MAC_4

where

$Data_4$ is the response to the request of Message 3 and can safely contain sensitive information;

$MAC_4 = f_K$ [MAC_3 || N_4 || e_K($Data_4$)]

N_4 is a "previously unused number" selected by the server.

NOTE N_4 is only present if Messages 1 and 2 are omitted and if Message 4 is not to be the final message of the exchange.

6.2.6 Steps 5 to N-2 — Further Secured Data Transmission Requests and Responses

N is the number of the final message in the exchange and is even. The general request message form now becomes

APar || e_K($Data_n$) || MAC_n

where

$$MAC_n = f_{K'}[MAC_{n-1} \,||\, e_K(Data_n)]$$

n is the message number and is odd.

The general response message form now becomes

$$APar \,||\, e_K(Data_n) \,||\, MAC_n$$

where

$$MAC_n = f_{K'}[MAC_{n-1} \,||\, e_K(Data_n)]$$

n is the message number and is even.

Provided that the MACs continue to verify correctly and the media-related time-outs do not expire, the client and the server can continue to exchange messages as above. If the MACs fail to verify or the time-outs do expire, then both entities should record the certificates and the audit information and apply an exception handling according to 6.4.

If Messages 1 and 2 are omitted, then the client will have authenticated the server after successful receipt and verification of Message 4, and the server will have successfully authenticated the client after successful receipt and verification of Message 5. At these points in the message exchange, the client and server can be certain that the received messages are not replays of previously transmitted valid messages. Hence, after these points the client and server can react to the receipt of messages with actions that are critical with respect to replay (e.g. change current configuration information).

6.2.7 Message Sequence Termination

A message sequence may be terminated at any time in the course of an exception handling as specified in 6.4. Furthermore, the client may at any time decline to continue the message sequence in not sending new requests. The client may, if needed, start a new message sequence instead, by sending a new secured link set-up request (see 6.2.2).

The client may also terminate the message sequence by sending a Message Sequence Termination Indication in the administration parameter (see 7.6.2) and by signing the request message:

$$Sig_C[\, APar \,||\, e_K(Data_{N-1}) \,||\, MAC_{N-1}\,]$$

In this case, the server shall respond by sending a signed message, as well:

$$Sig_S[\, APar \,||\, e_K(Data_N) \,||\, MAC_N\,]$$

This shall terminate the message exchange. This procedure protects the whole sequence of messages against repudiation by either party (see 5.3.6.5).

If the server wants to terminate the message sequence, then it will not accept the secured data transmission request from the client, sending a negative response as specified in 6.4, with a negative response code indicating an insufficient protection. The client will then send the request again, now with a Message Sequence Termination Indication as described, to terminate the sequence with the response from the server.

6.2.8 Request Signature — Optional

The non-repudiation service specified in 5.3.6.5 provides for either party to request that the other signs the next message. The implementation of this service in the message sequence is as follows:

a) If the client intends to request a signature from the server, it indicates this in the APar of the Secured Data Transmission Request message. The server should respond with the message:

$$Sig_S[\, APar \,||\, e_K(Data_n) \,||\, MAC_n\,]$$

b) If the server intends to request a signature from the client, then it will reject the secured data transmission request from the client, sending a negative response as specified in 6.4, with an error code indicating an insufficient protection. The client will then send the request again, now including the signature:

$$Sig_C[\ e_K(APar \mid\mid Data_n\ ) \mid\mid MAC_n\ ]$$

The signatures ensure that the messages exchanged during a transaction cannot be repudiated by the party sending the signature. They may be added to any message subsequent to Message 3.

6.2.9 Illustration of the message sequence

The message sequence is illustrated in. Note that the exchange illustrated includes Messages 1 and 2 and incorporates a Message Sequence Termination exchange of signed messages, as specified in 6.2.7.

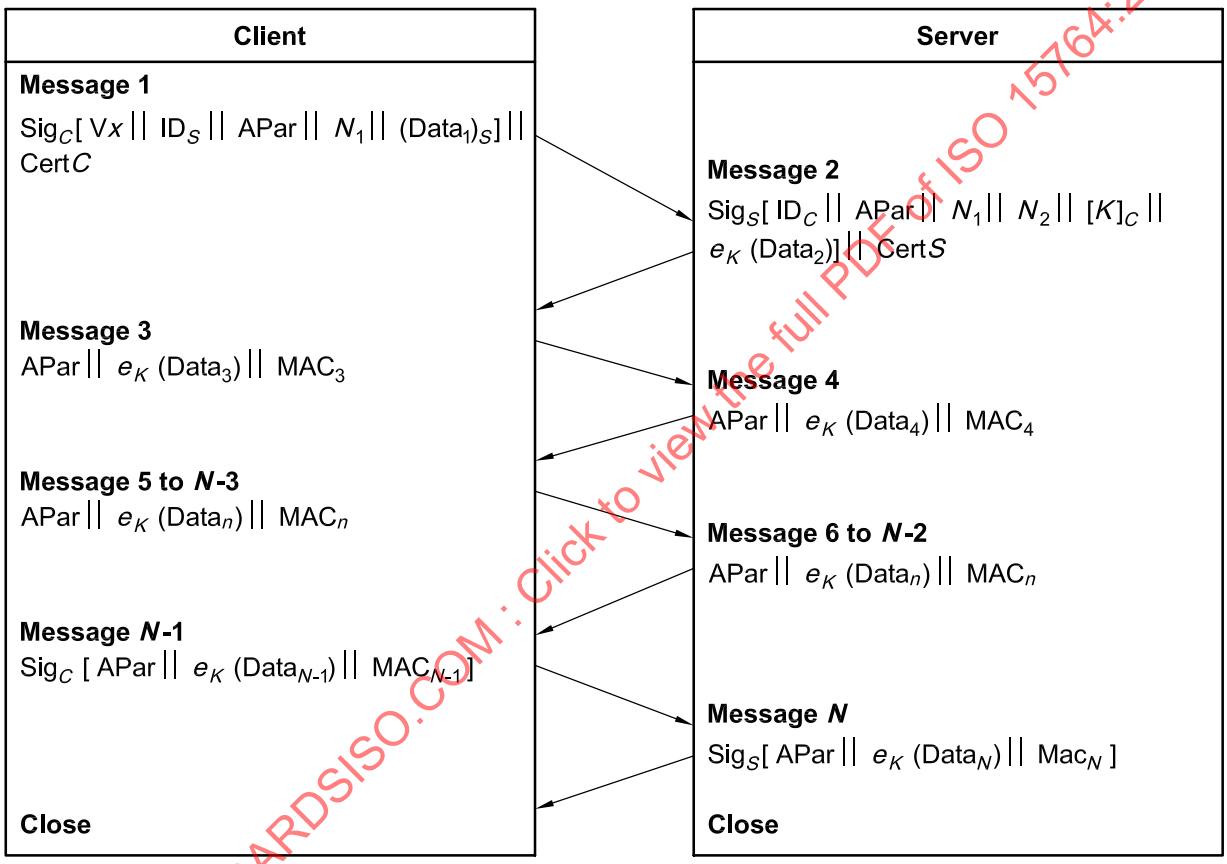


Figure 2 — General message sequence

6.3 Security parameters

CAUTION — Care should be taken when selecting the values of all the parameters since their combination determines the robustness of the security for an application or a system.

Different applications of the message exchanges specified in this standard may require different levels of security. If necessary, the parameters specified below may be changed to comply with the desired security level and computing power available. Selection of parameters different to those specified here should only be made as part of a system security assessment, the details of which are outside the scope of this standard. For a particular application the parameter values will normally be fixed for each client and each server. If the client sends request messages that do not conform to the security parameters of the server, then the server will send a negative response (see 6.4). Depending on the indicated error type, the client may change the request. Default values of the security parameters for specific applications are given in Clause 8.

See Table 1.

Table 1 — Security parameters

Parameter	Recommended value	Notes
Public key length	1 024 bit modulus 1 024 bit exponent	—
Private key length	1 024 bit modulus 1 024 bit exponent	—
Secret key length	128 bits	—
MAC key length	128 bits	Same as secret key.
MAC length	20 Bytes	If non-repudiation protection for Messages 3, 4, ... , N was not required, then a substantially shorter MAC could be used.
Number of times a secret key established at the beginning of a message sequence may be used	100 messages	
Delay time after security violation	1 second	Preferably increasing with number of false access attempts or limitation of number of re-access attempts. See 6.4.2 for handling.
Delay time on power-up	Depending on power-up properties of device	To inhibit power-up attacks. No message will be processed before this time has elapsed.

6.4 Exception detection and exception response

6.4.1 General

This standard only considers exceptions detected by the security sub-layer. It is the task of the security sub-layer to guarantee a secure transmission of the data and to detect errors, which indicate that security may be compromised.

It is possible that secure data may be incorrect from an application view point — for example, the wrong calibration data may have been sent because the vehicle details were sent incorrectly. The security sub-layer does not monitor the content of the data to be transmitted in the secured mode. If, for instance, the application finds an error in the service being transmitted in the secured mode and sends a negative response, this response is sent with the same security measures as would have applied to a positive response. Negative responses in the service to be sent in the secured mode do not influence the behaviour of the security sub-layer.

It is up to the lower level communication layers to ensure that the data they exchange and forward to the security sub-layer are free of non-malicious errors. It is assumed that the transfer of data from the lower communication layers, which is deemed to be correct but is in fact incorrect, is so unlikely that it can be dealt with as a non-recurring exception.

Two types of errors can be distinguished:

a) Security violations are exception-detected when checking the content of the security relevant parameters as indicated in 6.2 for each message. In this case, appropriate measures must be taken to avoid that the security is compromised by a third party systematically sending false messages in order to observe the security mechanisms of the receiver.

b) Administrative errors are related to the fact that the client and the server need to agree on the security level of a given secured data transmission and need additional data on the other entity for their security checks. These errors occur when the receiver of a message expected some other message or other form of the message, for instance with some additional parameters being included.

The security sub-layer should always check for administrative errors first and only react on security violations if there is no administrative error in the message.

The security sub-layer at the client or the server, detecting an exception, shall send a negative response message according to the following scheme.

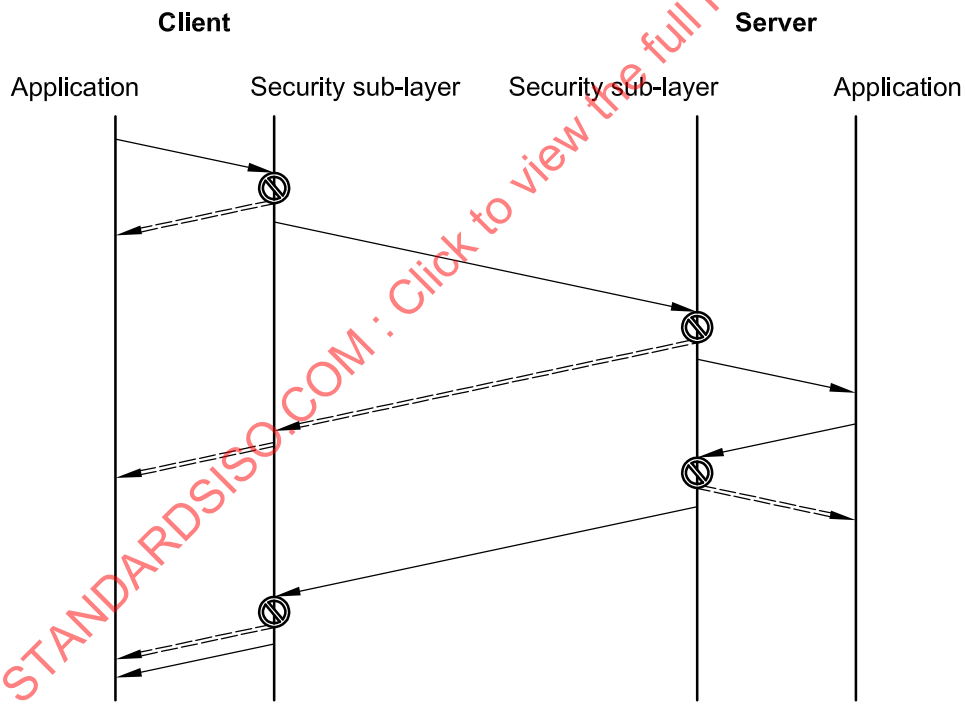
— If the security sub-layer on the client side detects an exception on the input for a request to be forwarded to the server, it will send an error indication to the application on the client side. The request will not be processed.

— If the security sub-layer on the server side, receiving a request from the client, detects an exception, then it will send a negative response to the security sub-layer on the client side, to be forwarded to the application on the client side. The request will not be forwarded and the application on the server side will not be notified.

— If the security sub-layer on the server side detects an exception on the input for a response to be forwarded to the client, it will send an error indication to the application on the server side. The response will not be forwarded and the client will not be notified. The application on the server side may change the input and ask once more to forward it.

— If the security sub-layer on the client side, receiving a response from the server, detects an exception, then it will send an error indication to the application on the client side. The response will not be forwarded. The server will not be notified.

The situation is shown in Figure 3.



Key

transmitted data possible exceptions

negative response message

Figure 3 — Negative response messages in case of exception

When the security sub-layer on the server side detects an exception and sends a negative response message, this message contains only an error code. It is neither encrypted nor does it contain security specific parameters like a signature, a MAC, previously unused numbers etc. Error indications of the security sub-layer to the application will also contain an error code. The supported error codes are specified in Table 2.

Table 2 — Supported error codes

No.	Error code name and application conditions
0	**GeneralSecurityViolation** The server shall use this response code, if the decryption of encrypted data in the request failed. The server shall use this response code if the verification of the certificate in the request failed. This includes a failure in verifying the signature contained in the certificate or a certificate being sent that expired or is reported as being revoked. The server shall use this response code if the verification of the authentication parameter (signature or MAC) in the request failed or a previously unused number is not correctly returned. The security sub-layer on the client side shall generate this code if the verification of the certificate or of the authentication parameter (signature or MAC) in the response failed or if a previously unused number is not correctly returned. The security sub-layer on the client side shall use this response code if the client identifier in the response is not correct or if the decryption of encrypted data in the response failed.
1	**securedModeRequested** The server shall use this response code if the request is sent in normal mode, but should be sent using the secured mode.
2	**InsufficientProtection** The server shall use this response code if at least one of the options chosen by the client in the APar does not correspond to the server's security policy in general or in respect of the secured mode service to be processed. The server shall use this response code if the request contains a secured mode service to be processed, but the protection against replay attacks needed according to the security policy of the server for this service is not present.
3	**TerminationWithSignatureRequested** The server shall use this response code if, according to its security policy, the message sequence needs to be terminated and if the termination should include signatures as specified in 6.2.7. In the case of an immediate sequence termination requested without signed termination messages, the server sends the GeneralSecurityViolation error code.
4	**AccessDenied** The server shall use this response code if a message is received before the delay time after a false access attempt or the delay time on power up has elapsed, or if the number of security violations is above the given limit (see 6.4.2).
5	**VersionNotSupported** The security sub-layer on the client side shall use this response code if, in the request service primitive, a version of this standard is requested that it does not support. The server shall use this response code, if Vx, the version of the standard indicated by the client, is not supported by the server, or in the given context is not considered as sufficiently secure.
6	**SecuredLinkNotSupported** The security sub-layer on the client side shall use this response code if the use of a pre-established key is requested but no such key is available, or if there would be a need to set up a new secured link, but the corresponding functionality is not available or not working. The server shall use this response code if the server identifier in the request is not correct, if the set-up of a new secured link is requested but the corresponding functionality is not available or not working, or if a secured data transmission request message is sent without a secured link being established.
7	**CertificateNotAvailable** The server shall use this response code if there is no certificate included in the request and the server has not established the public key of the client as trusted. The server shall use this response code if there is a certificate included in the request and the server has not established the public key of the certification authority as trusted.
8	**auditTrailInformationNotAvailable** The server shall use this response code if there is some audit trail information requested that is not available.

Table 2 (*suite*)

No.	Error code name and application conditions
9	**requestMustContainAuditTrailInformation** The server shall use this response code if an action to be taken on the request or the information to be sent in the response is restricted to a situation where some audit trail information of the client is available at the server. The client shall then send the request again, this time including all audit trail information available.

NOTE 1 Table 2 includes only security-specific codes. More error codes could be available from the specific protocol used for the data transmission (e.g. ISO 14229-1). This specific protocol also indicates how to mark a response as being negative.

NOTE 2 When using the negative response codes in ISO 14229-1 services, 38 (hex) has to be added to the number in Table 2 to get the appropriate code number.

6.4.2 Security violations

The error code according to Table 2 for all security violations is 0, indicating a General Security Violation. No further details on the cause of the violation are given in the error indication or negative response message.

A security violation can occur either when a message is sent from the security sub-layer on the client side to the security sub-layer on the server side, or vice-versa. In both cases, the security sub-layer on the client side will be informed on the violation, as can be seen from Figure 3.

In case the violation occurs in a secured data transmission request or response message, the security sub-layer on the client side will immediately terminate the message sequence without additional sequence termination messages. It may then try to set up a new secured link with the same server. The security sub-layer on the server side, having sent a negative response indicating a security violation, will not accept any further messages in the same message sequence, but will accept a new secured link set-up request from the same client (or, in the case of a pre-established secret key, a new message sequence starting with Message 3).

If the security violation occurs in a secured link set-up message, then the client may try to start a secured link set-up request message once more (this being different from the previous one in the previously unused number). The security sub-layer on the server side, having sent a negative response indicating a security violation, will only accept a new secured link set-up request message after the time defined by the delay time has elapsed. On any such message received from any client before this time has elapsed, a negative response will be sent indicating AccessDenied. The client then will try to send the request message again, later.

The security sub-layer on the client side may, as a measure against false messages sent systematically by a third party, limit the number of attempts to set up a secured link with a specific server. After the limit is reached, an error code indicating AccessDenied will be sent to the application on the client side on each request for a secured mode service involving that server. If possible, the situation should be reported to human operators.

As a measure against false messages sent systematically by a third party, the security sub-layer on the server side may limit the total number of secured link set-up requests or the number of consecutive secured link set-up requests with security violation. After the limit is reached, a negative response indicating AccessDenied will be sent to any secured link set-up request from any client. Again, if possible, the situation should be reported to human operators. Message sequences already established should not be affected, and a client terminating such a sequence should be able to start the next one (having a limited additional number of access attempts).

6.4.3 Administrative errors

Administrative errors are all errors indicated in a negative response message distinct from the GeneralSecurityViolation. Where such an error is received, the corresponding request message will be deleted from the message sequence and the client may continue the sequence from the previous response message sent by the server.

7 Element description

7.1 General

This clause describes the form and content of elements used with the individual messages. Where applicable, it specifies the procedures needed by the security sub-layer to generate the elements. It attempts to be as prescriptive as possible, to allow non-security experts to implement the standard.

7.2 Security sub-layer service request parameters

7.2.1 General

The following defines the parameters to be forwarded to the security sub-layer on the client side to initiate a request message transmission. The same parameters are used for all request messages and it is up to the security sub-layer to decide which request message to forward to the lower layers, depending on the content of the service request parameters.

7.2.2 Request Parameter Overview

Table 3 gives an overview on the parameters needed for a security sub-layer service request.

Table 3 — Service-specific parameters of the request service primitive — Overview

Parameter length bytes	**Parameter Name**	**Cvt.**
Defined by protocol	**SecuredDataTransmission Request Service ID**	**M**
1	versionNumber	M
8	ServerIdentifier	M
2	securityProfile	M
1	securedModeServiceType	M
Depending on securedModeServiceType	securedModeServiceIdentifier	M
Defined by corresponding service	securedModeServiceRequestParameters	C_1
Depending on securityProfile	AuditTrailInformation	C_2

M Mandatory: the parameter shall be present in the service primitive.

C Conditional: the parameter can be present in the service primitive, based on the following criteria:

C_1: present if the service given by the securedModeServiceType and the securedModeServiceIdentifier requires additional request parameters;

C_2: present if indicated in the securityProfile parameter.

7.2.3 Version number

Definition: The versionNumber parameter identifies the version of this standard to be used for the protection of the service. Later versions of the standard than that indicated may be used as well, as it is assumed that they provide a higher level of protection.

Form: Octet with assigned values according to Table 4.

Table 4 — versionNumber value assignment

versionNumber value	ISO 15764 Version
0	Not specified
1	ISO 15764 — "Publication Date" with 3DES for symmetric encryption
2	ISO 15764 — "Publication Date" with AES for symmetric encryption
3-255	Reserved by ISO for future use

7.2.4 Server Identifier

Definition: The ServerIdentifier parameter contains a unique identifier of the server addressed in the service. It allows the security sub-layer of the client to check if a secured link with that Server is already established, to establish such a link if needed and to use it for the requested service. The method for the application on the client side to get the right Server identifier is outside the scope of this standard.

Form: Octet string of 8 bytes length. The ServerIdentifier must be unique within the whole security system. The procedures of assigning identifiers to Servers is outside the scope of this standard.

7.2.5 Security profile

Definition: securityProfile gives all relevant information on the protection of the data transmission requested by the application on the client side for the service. It includes indication of presence or absence of audit trail information parameters.

Form: Bit string of 16 bits length with bit assignment according to Table 5. If a bit is set to 1, then the corresponding feature is requested. If it is set to 0, then the corresponding feature is not requested.

7.2.6 Secured mode service type

Definition: The securedModeServiceType parameter identifies the type of service intended to be executed in a secured mode.

Form: The parameter is an octet string of one byte length. The values assignment is as given in Table 6.

7.2.7 Secured mode service identifier

Definition: The securedModeServiceIdentifier parameter identifies the requested service within the framework of the service type given in the securedModeServiceType parameter.

Form: The form is defined in the document specifying the corresponding service type.

7.2.8 Secured mode service request parameters

Definition: The securedModeServiceRequestParameters parameter contains all parameters needed for the service request of the service to be executed in the secured mode.

Form: The parameter is empty or contains one or more concatenated parameters. These parameters are defined in the specification document of the service given by the securedModeServiceType and securedModeServiceIdentifier parameters.

Table 5 — Assignment of securityProfile bits

Bit No.	Meaning
1	Use pre-established key. If both a pre-established key and the procedure to set up a secured link with Messages 1 and 2 are available, this bit decides on which option to use. If only one option is available and the bit value doesn't correspond to this option, then the service request will be rejected.
2	Protection against eavesdropping needed.
3	Protection of the request against replay attacks needed. This bit is set to 1 if the server is expected to take some security critical action in response to the request.
4	Non-repudiation protection needed on current service.
5	Non-repudiation protection needed at the end of the message sequence. This bit is set to 1 if it is expected that, on termination of the message sequence, the whole sequence is protected against repudiation. This protection cannot be guaranteed under all circumstances. Therefore, bit 4 should be set to 1 wherever non-repudiation is vital.
6	Audit trail information to include date and time.
7	Audit trail information to include VIN.
8	Audit trail information to include user ID.
9	Audit trail information to include software number.
10	Audit trail information to include software version number.
11	Audit trail information to include exhaust regulation or type approval number.
12	Audit trail information in the response must include VIN.
13	Audit trail information in the response must include user ID.
14	Audit trail information in the response must include software number.
15	Audit trail information in the response must include software version number.
16	Audit trail information in the response must include exhaust regulation or type approval number.
When the audit trail information includes date and time (securityProfile bit number 6 set to 1), then date and time shall also be included in the audit trail information of the service response.	

Table 6 — securedModeServiceType value assignment

securedModeServiceType value	Service type
0	Not specified
1	Diagnostic service according to ISO 14229-1
2	Diagnostic service according to SAE J1939
3	Tachograph service according to ISO 16844-4 and ISO 16844-7
4-127	Reserved by ISO for future use
128-255	Vehicle manufacturer specific service

7.2.9 Audit trail information

7.2.9.1 General

Definition: The auditTrailInformation enables either party to record details of the other in association with a transaction. This may be used as part of the acceptance criteria before sensitive data is transmitted and would

allow, for example, for the rejection of a request from external test equipment that had been reported as stolen. It is also provided as a deterrent to miss-use as such use would be traceable.

Form: The auditTrailInformation parameter is a concatenation of parameters out of the list given in Table 7. The presence or absence of a parameter is indicated in the securityProfile parameter. The definition and form of the parameters is specified in the subsequent sub-clauses.

Table 7 — Audit trail information

Audit trail parameter name
dateTime
vehicleIdentificationNumber
userID
softwareNumber
softwareVersionNumber
exhaustRegulationTypeApprovalNumber

7.2.9.2 Date and time

Definition: The dateTime parameter indicates the date and time when, according to the clock of the sender, the service parameters are forwarded to the security sub-layer to be transmitted.

Form: Octet string of 8 bytes length. The value assignment is according to ISO 16844-4 and ISO 16844-7.

7.2.9.3 Vehicle identification number

Definition: The vehicleIdentificationNumber parameter contains the vehicle identification number (VIN) uniquely identifying the vehicle involved in the requested service either on the client side or on the server side.

Form: The parameter is the CS5 coding structure as defined in ISO 14816. It is a visible string with values assigned according to ISO 3779 and ISO 3780, including the world manufacturer identifier (WMI), the vehicle description section (VDS) and the vehicle indicator section (VIS).

7.2.9.4 User ID

Definition: In case the sender of the audit trail information has a user interface and is able to identify the user initiating the message exchange, then a unique identifier of this user may be included as the userID parameter in the audit trail information.

This parameter should only be used if a procedure to authenticate the user towards the equipment is implemented on the user interface.

Form: The form is the same as for the serverIdentifier parameter (see 7.2.4).

7.2.9.5 Software number

Definition: The softwareNumber parameter allows the manufacturer (or the entity responsible for software updates) to identify the software currently implemented in the client or the server.

Form: Equipment manufacturer-specific.

7.2.9.6 Software version number

Definition: The softwareVersionNumber parameter allows the manufacturer (or the entity responsible for software updates) to identify the software version currently implemented in the client or the server.

Form: Equipment manufacturer-specific.

7.2.9.7 Exhaust regulation or type approval number

Definition: The exhaustRegulationTypeApprovalNumber shall be used for clients or servers in the vehicle that require type approval, to reference the exhaust regulation or type approval number.

Form: The parameter is a visible string with equipment-specific content, as defined by the authority responsible for the exhaust regulation or the type approval.

7.3 Security sub-layer service indication parameters

The security sub-layer service indication parameters, being the parameters forwarded by the security sub-layer on the server side to the application addressed, are the same as the security sub-layer service request parameters, with the exception that the server identifier is replaced by the client identifier. The client identifier is defined in correspondence to the server identifier (cf. 7.2.4) and each client identifier must be distinct from any Server identifier.

7.4 Security sub-layer service response parameters

7.4.1 General

The following defines the parameters to be forwarded to the security sub-layer on the server side in response to a service indication received.

It is assumed that the security sub-layer is able to assign the response to the appropriate request. If this is not the case, then the client identifier shall be repeated in the response.

7.4.2 Response parameter overview

Table 8 gives an overview of the parameters needed for a security sub-layer service response.

Table 8 — Service-specific parameters of the response service primitive — Overview

Parameter length (bytes)	Parameter Name	Cvt.
Defined by protocol	**SecuredDataTransmission Response Service Id**	**M**
Depending on securedModeSeviceType	securedModeServiceIdentifier	M
Defined by corresponding service	securedModeServiceResponseParameters	C_1
Depending on securityProfile	auditTrailInformation	C_2

M Mandatory: the parameter shall be present in the service primitive.

C Conditional: the parameter can be present in the service primitive, based on the following criteria:

C_1: present if the service given by the securedModeServiceType and the securedModeServiceIdentifier requires additional request parameters;

C_2: present if indicated in the securityProfile parameter of the corresponding indication.

7.4.3 Secured mode service identifier

The securedModeServiceIdentifier parameter is defined in the same way as in the service request (see 7.2.7).

7.4.4 Secured mode service response parameters

The securedModeServiceResponseParameters parameter is defined analogous to the securedModeServiceRequestParameters parameter (see 7.2.8), now including the parameters needed for the service response of the service to be executed in the secured mode.

7.4.5 Audit trail information

The auditTrailInformation parameter is defined in the same way as in the service request (see 7.2.9). The selection of parameters out of the parameter list is as requested in the securityProfile parameter of the corresponding service indication.

7.5 Security sub-layer service confirmation parameters

The security sub-layer service confirmation parameters, being the parameters forwarded by the security sub-layer on the client side to the application addressed, are the same as the security sub-layer service response parameters.

7.6 Secured Data Transmission Parameters

7.6.1 General

The following specifies the parameters used in the messages listed in 6.2.

7.6.2 Administration parameter

Definition: The administration parameter APar is contained in the beginning of each message and allows identification of the message type and the message content.

Form: The parameter is a bit string of 16 bits length. The bit assignment shall be according to Table 9. If a bit is set to 1 then the corresponding feature is requested. If it is set to 0 then the corresponding feature is not requested.

Table 9 — Administration parameter (APar) bit assignment

Bit number	Meaning
1	Message is request message. (If not, it is a response message.)
2	Message makes part of secured link set-up. (If not, it makes part of a secured data transmission.)
3	Message makes part of a message sequence termination with signature.
4	A pre-established key is used. (If not, a key established in a secured link set-up is used.)
5	Message is encrypted.
6	Message is signed.
7	Signature on the response is requested.
8	Message has reduced form (as indicated in 6.2.2 and 6.2.3)
9	Message contains request/response function.
10	Message contains audit trail information.
11	Message contains certificate
12-16	Reserved by ISO for future use.

7.6.3 Identifiers

The ID_C and ID_S parameters, being the identifiers of the client and the server, have the same form and values as the clientIdentifier and the serverIdentifier of the corresponding request and response service primitives. It is assumed that the security sub-layer knows its own identifier and includes it in the message if needed.

7.6.4 Encryption

If the securityProfile parameter of the service request indicates that protection against eavesdropping is needed, then the sensitive parts of the corresponding request and response message are encrypted by the security sub-layer. This is indicated in the APar (bit number 5). For Message 1, RSA encryption is used. For the other messages symmetric encryption is used, based on the secret key.

RSA encryption shall be implemented using the padding and encoding conventions specified in PKCS #1.

The algorithm for symmetric encryption depends on the version number of this standard as given in the service request (see 7.2.2). For Version 1 it is triple DES (3DES) as defined in ISO/IEC 18033-3. For Version 2 it is AES (Rijndael) as defined in ISO/IEC 18033-3.

The value of the secret key should change after a given number of uses in a message sequence (except in cases where Messages 1 and 2 are omitted — see 6.2). For changing the value, a new message sequence shall be set up.

If the encryption option is used, then the parameters to be encrypted, as indicated in 6.2, are replaced by a sequence of two parameters:

a) The encryptedDataLength parameter is an octet string of one byte length and gives the length of the subsequent parameter in multiples of 8 bytes.

b) The encryptedData parameter contains the data string being the result of the encryption process.

7.6.5 Initializing value

The symmetric encryption of data shall be performed using the cipher block chaining mode, as specified in ISO/IEC 10116. The key used shall be the secret key. The initializing value (IV) used shall be transmitted in Message 2, together with the secret key and shall be fixed for the message sequence. It is a 64 bit random number for 3DES encryption and a 128 bit random number for AES encryption. If a pre-established key is used and Messages 1 and 2 are omitted, then the IV parameter shall be pre-established as well. The IV parameter is assumed to be available at the security sub-layer rather than at the application.

7.6.6 Secret key

The secret key K is used for symmetric encryption. It is recommended to be 128 bits in size for both encryption algorithms. It is either pre-established on both the client and the server side, or generated by the server and sent as a copy to the client in Message 2, together with the initializing value. In Message 2, the secret key and initializing value are RSA-encrypted using the public key of the client (see 6.2.3 and 7.6.4).

7.6.7 Previously unused numbers

A number of size 64 bits is to be generated by the security sub-layer at the commencement of a message sequence. The number may be generated randomly or sequentially. The main criterion is that the value for a given message sequence has not been used previously for the same purpose by the party generating it within the lifetime of the current public/private key pair. Conveniently, this number may be chosen to be identical to, or fulfil the role of, a job number, thereby providing a log for the client or the server.

7.6.8 ISO 15764 version

The parameter Vx, indicating the ISO 15764 version, has the same form as the versionNumber parameter defined in 7.2.3. It is fixed for the whole message sequence. If an application requests for a specific service a later version of ISO 15764 than established for the message sequence, then a new message sequence shall be set up.

7.6.9 Signature

The signature $Sig_A[X]$ by the entity A on the signed data X is an extra parameter to be put in by the security sub-layer together with the signed data. The signed data shall be put in first and the signature second. The signature length is the same as the modulus length of the public key of the signing entity, given in the corresponding certificate (see 7.6.10).

As part of the signature calculation, the data string to be signed is input to a hash-function. The hash-function to be used is that specified as dedicated hash-function 3 in ISO/IEC 10118-3, and otherwise known as SHA-1. This function reduces an arbitrary length data field to a 160-bit hash-code using an iterative process. The data to be hashed in each case is identified in the relevant clause of this standard. The signature is computed by applying the RSA algorithm to the hash-code.

Signatures based on combining SHA-1 with the RSA algorithm shall be implemented using the padding and encoding conventions specified in PKCS #1.

7.6.10 Certificate

The certificate of the entity A, $Cert_A$, is a parameter consisting of a number of other parameters, including the identifier of A, ID_A. It is assumed to be available at the security sub-layer of the entity.

Certificates used in conjunction with the message exchanges specified in this standard shall conform to ISO/IEC 9594-8, i.e. the certificates shall be of the type known as X.509 Version 3. The certificates shall be encoded using the Distinguished Encoding Rules (DER) specified in ISO/IEC 8825-1. The certificates shall conform to the certification profile specified in IETF RFC 2459.

Table 10, Table 11 and Table 12 define the uses that shall be made of the fields within the certificate, as constrained by RFC 2459.

Table 10 — Fields in SEQUENCE certificate

Field name	Value
tbsCertificate	This field contains an ASN.1 sequence TBSCertificate (defined below). This sequence includes the name of the subject and issuer (CA), a public key associated with the subject, a validity period, and other associated information.
signatureAlgorithm	This field contains an ASN.1 sequence AlgorithmIdentifier that identifies the algorithm used by the CA to sign this certificate. An AlgorithmIdentifier (see ISO/IEC 9594-8) contains an Object Identifier (OID) used to identify an algorithm, and optional parameters the contents of which vary according to the algorithm identified. For certificates used by applications conforming to this standard, the ASN.1 OID shall always be: sha-1WithRSAEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5 } The parameters component of the AlgorithmIdentifier shall be the ASN.1 type NULL.
SignatureValue	This field contains a digital signature computed upon the ASN.1 DER encoded tbsCertificate. The ASN.1 DER encoded tbsCertificate is used as the input to the signature function. The resulting signature value is then ASN.1 encoded as a BIT STRING.

Table 11 — Fields in SEQUENCE TBS certificate

Field name	Value
version	The field indicates the version of the encoded certificate. This field shall take the value 2, indicating X.509 version 3.
SerialNumber	The serial number is an integer assigned by the CA to each certificate. It shall be unique for each certificate issued by a given CA.
Signature	This field contains an ASN.1 sequence AlgorithmIdentifier that identifies the algorithm used by the CA to sign this certificate. It shall always be the same as the contents of the signatureAlgorithm field in the ASN.1 sequence certificate.
Issuer	This shall be an empty sequence.
Validity	This indicates the time interval during which the CA warrants that it will maintain information about the status of the certificate. The field is represented as an ASN.1 sequence of two dates: the date on which the certificate validity period begins (notBefore) and the date on which the certificate validity period ends (notAfter). Certificates shall always encode certificate validity dates up to the year 2049 as UTCTime; certificate validity dates from 2050 onwards shall always be encoded as GeneralizedTime. UTCTime and GeneralizedTime are standard ASN.1 types for expressing time/date values.
Subject	This shall be an empty sequence.
SubjectPublicKeyInfo	This field is used to carry the RSA public key and identify the algorithm with which the key is used (i.e. RSA). This is carried within an ASN.1 sequence of type SubjectPublicKeyInfo. This sequence contains two fields: algorithm (an AlgorithmIdentifier) and subjectPublicKey (a BIT STRING). The ASN.1 sequence AlgorithmIdentifier identifies the algorithm for the public key, i.e. RSA. An AlgorithmIdentifier (see ISO/IEC 9594-8) contains an Object Identifier (OID) used to identify an algorithm, and optional parameters the contents of which vary according to the algorithm identified. In this case the ASN.1 OID shall always be rsaEncryption OBJECT IDENTIFIER ::= { pkcs-1(1) 1 } pkcs-1 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) 1 } The parameters component of the AlgorithmIdentifier shall be the ASN.1 type NULL. The subjectPublicKey shall contain an encoding of the RSA public key of the subject of the certificate. The RSA public key shall be encoded using the ASN.1 type RSAPublicKey. This is an ASN.1 sequence defined as RSAPublicKey ::= SEQUENCE { modulus INTEGER, -- n ..publicExponent INTEGER -- e -- } The modulus is the RSA modulus (n) and the exponent is the public exponent (e). The DER encoded RSAPublicKey is the value of the BIT STRING subjectPublicKey.
IssuerUniqueID	This field shall contain a unique identifier of the certificate issuer and shall have the same form as the unique identifier of the server defined in 7.2.4.
SubjectUniqueID	This field shall contain a unique identifier of the client or the server, being the subject of the certificate, and shall have the same form as the unique identifier of the server defined in 7.2.4.
Extensions	This field contains an ASN.1 sequence Extensions (which shall contain three fields, as defined in Table 12).

Table 12 — Fields in SEQUENCE extensions

Extension name	Value
Authority Key Identifier	This extension provides a means of identifying the public key to be used to verify the certificate. It is particularly useful where a CA has multiple signing keys. The Authority Key Identifier is an ASN.1 sequence containing three items: keyIdentifier, authorityCertIssuer, and AuthorityCertSerialNumber. All three are optional and the last two shall be omitted from conforming implementations. The value of the keyIdentifier field shall be derived from the CA public key used to verify the certificate. This shall be achieved by inputting a BIT STRING encoding of the CA public key (encoded in precisely the same way as the BIT STRING subjectPublicKey but excluding the tag, length, and number of unused bits) to the SHA-1 hash algorithm, and using the 160-bit hash-code as the keyIdentifier.
Subject Key Identifier	This extension provides a means of identifying the public key within the certificate. It is particularly useful where a subject has multiple RSA keys pairs. The Subject Key Identifier is an object of type keyIdentifier. The value of the keyIdentifier field shall be derived from the public key in the certificate. This shall be achieved by inputting the BIT STRING subjectPublicKey (excluding the tag, length, and number of unused bits) to the SHA-1 hash algorithm, and using the 160-bit hash-code as the keyIdentifier.
Key Usage	This field defines the purpose of the key contained in the certificate. This extension shall be marked critical. KeyUsage is a BIT STRING, in which bits are allocated to indicate whether or not the key is available for certain purposes. In this application the following bits shall be set: — (0) digitalSignature — (1) nonRepudiation — (2) keyEncipherment — (3) dataEncipherment

The extension identifiers for all the extensions listed in Table 12 are specified in ISO/IEC 9594-8 (3rd edition).

The CA as the issuer of the certificate may, for example, be

— a dealer network approving each of its outlets,

— a vehicle manufacturer approving each of its franchisees,

— a recognized roadside repair organization approving each of its patrols, or

— an authority responsible for the verification of vehicle tachographs programming its certificate within the tachograph.

Self-certification would be acceptable, for example, in the case of a vehicle manufacturer releasing information relating to his own vehicles.

7.6.11 MAC

MAC_n is a parameter contained in all messages n from Message 3 onward. It is calculated by the security sub-layer on other parameters as indicated in Clause 6.2.4. The calculation method is according to ISO/IEC 9797-2. The function to be used shall be that specified as MAC algorithm 2 with dedicated hash function 3 (this combination is commonly known as HMAC-SHA-1). The MAC_n value shall be 20 bytes in length.