INTERNATIONAL STANDARD

ISO/IEC 9798-6

First edition
2005-08-01

Information technology — Security techniques — Entity authentication —

Part 6: Mechanisms using manual data transfer

Technologies de l'information — Techniques de sécurité — Authentification d'entité —

Partie 6: Mécanismes utilisant un transfert manuel de données

Reference number
ISO/IEC 9798-6:2005(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) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national 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 and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 9798-6 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

ISO/IEC 9798 consists of the following parts, under the general title *Information technology — Security techniques — Entity authentication*:

— *Part 1: General*

— *Part 2: Mechanisms using symmetric encipherment algorithms*

— *Part 3: Mechanisms using digital signature techniques*

— *Part 4: Mechanisms using a cryptographic check function*

— *Part 5: Mechanisms using zero-knowledge techniques*

— *Part 6: Mechanisms using manual data transfer*

Introduction

Within networks of communicating devices it is often necessary for two devices to perform an entity authentication procedure using a channel which may be subject to both passive and active attacks, where an active attack may include a malicious third party introducing data into the channel and/or modifying, deleting or repeating data legitimately sent on the channel. Other parts of this International Standard describe entity authentication mechanisms applicable when the two devices share a secret key, or where one device has an authenticated copy of a public key for the other device.

In this part of ISO/IEC 9798, entity authentication mechanisms, referred to as manual authentication mechanisms, are specified where there is no such assumption of pre-established keying relationships. Instead entity authentication is achieved by manually transferring short data strings from one device to the other, or by manually comparing short data strings output by the two devices.

For the purposes of this part of ISO/IEC 9798, the meaning of the term entity authentication is different to the meaning applied in other parts of ISO/IEC 9798. Instead of one device verifying that the other device has a claimed identity (and vice versa), both devices in possession of a user verify that they correctly share a data string with the other device at the time of execution of the mechanism. Of course, this data string could contain identifiers for one or both of the devices.

As described in informative annexes A and B, a manual authentication mechanism may be used as the basis for secret key establishment or reliable exchange of public keys. A manual authentication mechanism could also be used for reliable exchange of other secret or public security parameters, including security policy statements or timestamps.

Information technology — Security techniques — Entity authentication —

Part 6:
Mechanisms using manual data transfer

1 Scope

This part of ISO/IEC 9798 specifies four entity authentication mechanisms based on manual data transfer between authenticating devices. As described in Annexes A and B, these mechanisms may be used to support key management functions; guidance on secure choice of parameters for the mechanisms is provided in Annex C.

Such mechanisms may be appropriate in a variety of circumstances. One such application occurs in personal networks, where the owner of two personal devices capable of wireless communications wishes them to perform an entity authentication procedure as part of the process of preparing them for use in the network.

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/IEC 9798-1:1997, *Information technology — Security techniques — Entity authentication — Part 1: General*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 9798-1 and the following apply.

3.1
check-value
string of bits, computed as the output of a check-value function, sent from the data originator to data recipient that enables the recipient of data to check its correctness

3.2
check-value function
function f which maps strings of bits and a short secret key, i.e. a key that can readily be entered into or read from a user device, to fixed-length strings of bits, satisfying the following properties:

— for any key k and any input string d, the function $f(d, k)$ can be computed efficiently;

— it shall be computationally infeasible to find a pair of data strings (d, d') for which the number of keys which satisfy $f(d, k) = f(d', k)$ is more than a small fraction of the possible set of keys.

NOTE 1 In practice, a short key would typically contain 4-6 digits or alphanumeric characters.

NOTE 2 In practice, security is maximized if the set of possible outputs from the check-value function is the same size as the set of possible keys.

3.3
data origin authentication
corroboration that the source of data received is as claimed

[ISO 7498-2]

3.4
manual authentication certificate
combination of a secret key and a check value generated by one of the two devices engaging in manual authentication, with the property that, when entered into the other device, these values can be used to complete the manual authentication process at some later time

3.5
Message Authentication Code
MAC
string of bits which is the output of a MAC algorithm

[ISO/IEC 9797-1]

3.6
Message Authentication Code algorithm
MAC algorithm
algorithm for computing a function which maps strings of bits and a secret key to fixed-length strings of bits, satisfying the following properties:

— for any key and any input string the function can be computed efficiently;

— 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.

[ISO/IEC 9797-1]

3.7
manual entity authentication
process achieving entity authentication between two devices using a combination of message exchanges via a (potentially insecure) communications channel and the manual transfer of limited amounts of data between the devices

3.8
simple input interface
interface for a device that shall allow the user to indicate to the device the successful or unsuccessful completion of a procedure, e.g. as could be implemented as a pair of buttons or a single button which is either pressed or not within a certain time interval

3.9
simple output interface
interface for a device that shall allow the device to indicate to the user the successful or unsuccessful completion of a procedure, e.g. as could be implemented by red and green lights or as single light which is lit in different ways to indicate success or failure

4 Symbols and abbreviated terms

A, B	Labels used for the two devices engaging in a manual entity authentication mechanism
D	Data string whose value is established between devices *A* and *B* as the result of performing a manual entity authentication mechanism

I_A, I_B	Distinguishing identifiers of *A* and *B* respectively.
K	(Short) secret key used with a check-value function in mechanisms 1 and 2
K_A, K_{Ai}, K_B, K_{Bi}	Random MAC keys used in mechanisms 3 and 4
MAC	Message Authentication Code
R	(Short) random bit-string used in mechanisms 3 and 4

5 Requirements

The authentication mechanisms specified in this part of ISO/IEC 9798 have the following requirements.

a) The pair of devices performing the manual authentication procedure shall be connected via a communications link (e.g. a wireless link). No security assumptions are made regarding this link; that is, the mechanisms are designed to operate securely even in an environment where an attacker can monitor and change data transferred on this link.

b) The pair of devices performing the manual authentication procedure shall both have a user interface capable of data input and data output.

c) The user data input interface for a device shall, at minimum, be capable of indicating successful or unsuccessful completion of a procedure (e.g. as could be implemented by using either two buttons or a single button which is either pressed or not within a certain time interval); such a means of data input is referred to below as a *simple* input interface. By contrast, a *standard* input interface shall provide means for the input of a short string of symbols, e.g. a numeric, hexadecimal or alphanumeric keypad. Unless explicitly stated otherwise, it is necessary that every device has a standard means of data input.

d) The user data output interface for a device shall, at minimum, be capable of indicating either success or failure of an authentication procedure (e.g. as could be implemented by means of red and green lights); such a means of data output is referred to below as a *simple* output interface. By contrast, a *standard* output interface shall provide means for the output of a short string of symbols, e.g. a numeric, hexadecimal or alphanumeric display. Unless explicitly stated otherwise, it is necessary that every device has a standard means of data output.

e) For mechanisms 1 and 2, the two devices performing the entity authentication procedure shall have agreed on the use of a specific check-value function, and shall have the means to implement this function.

NOTE Guidance on appropriate choices for check-value functions and lengths for check-values and random keys for use in mechanisms 1 and 2 is provided in Annex C. A construction for an unconditionally secure check-value function suitable for use with mechanisms 1 and 2 is given in Annex D.

f) For mechanisms 3 and 4, the two devices performing the entity authentication procedure shall have agreed on the use of a specific MAC algorithm, and shall have the means to implement this algorithm.

NOTE Guidance on appropriate choices for MAC algorithms and lengths for MACs and random keys for use in mechanisms 3 and 4 is provided in Annex C.

g) Prior to invocation of one of the manual authentication mechanisms, the two devices performing the mechanism shall have exchanged a data string D. This may be generated by one device and sent to the other device, or it may consist of the concatenation of data generated by both devices and sent in both directions across the communications link.

h) Either a single human user shall be in possession of both devices and shall operate them both, or the two devices shall be operated by two users who share a trusted means of communication.

6 Mechanisms using a short check-value

6.1 General

In this clause two manual authentication mechanisms are specified that are based on the use of a check-value. The two mechanisms are appropriate for different types of devices. Specifically,

— the first mechanism (mechanism 1) is appropriate for the case where one device has a simple input interface and the other has a simple output interface, and

— the second mechanism (mechanism 2) is appropriate for the case where both devices have a simple input interface.

A standard input or output interface can emulate a simple interface, and hence if both devices have standard input and output interfaces then either of the mechanisms may be used.

Both mechanisms operate in the following general way. A data string D is transferred from one device to the other (or is the concatenation of data transferred in both directions) via the communications link between them. The manual entity authentication mechanism is then executed. As a result of the mechanism both devices are provided with assurance that the data string D they possess is the same as the value held by the other device.

6.2 Mechanism 1 – One device with simple input, one device with simple output

6.2.1 Requirements

This mechanism has the following specific requirements.

a) The mechanism specified in this subclause is appropriate for the case where one device (device A) has a simple input interface and the other (device B) has a simple output interface.

b) Device A shall have the means to generate keys.

6.2.2 Specification of data exchanged

The following data exchanges and operations shall take place (see also Figure 1).

a) Both devices shall output a signal to acknowledge that they have received data D and that they are ready for the authentication mechanism to commence. On observing that both devices are ready, the user shall then enter a signal into device A to notify it that the mechanism can start.

b) Device A shall generate a random key K, where K is suitable for use with the check-value function shared by the two components. Using this key K, device A shall compute a check-value as a function of the data D. The check-value and the key K shall then be output via the output interface of device A. The user shall read the check-value and the key K from the output interface.

c) The user shall enter the check-value and the key K output by device A to device B using its input interface. Device B shall use the key K to re-compute the check-value as a function of its stored version of data D. If the two check-values agree, then device B shall output a success signal to the user via its simple output interface. Otherwise it shall give a failure signal.

d) The user shall enter the result output by device B, i.e. success or failure, into device A via its simple input interface.

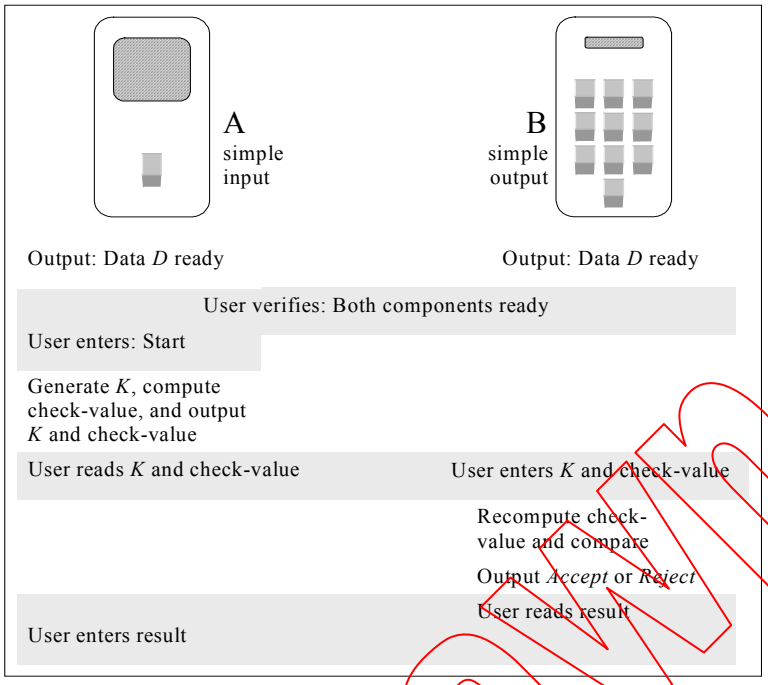

Figure 1 — Manual authentication mechanism 1

6.2.3 Manual authentication certificates

Manual authentication mechanism 1 has the property that no authentication information is transmitted over the insecure channel. Therefore, it does not affect the security of the mechanism if the manual authentication values K and check-value are transferred from device A to device B before the latter has received the actual data D. Naturally, such an approach is applicable only to situations where device A generates the data D. However, in such a case, mechanism 1 offers a means of authenticating data to be received at some later time. Such authentication means is called a manual authentication certificate. A protocol for data origin authentication using a manual authentication certificate is now specified (with the same requirements as specified in subclause 6.2.1). Note that this protocol does not provide entity authentication.

Suppose device A has data D that needs to be sent to device B at some later time.

a) Device A generates a random key K, where K is suitable for use with the check-value function shared by the two devices. Using this key K, device A computes a check-value as a function of the data D. The check-value and the key K are then output to the user by the output interface of device A. The user reads the output check-value and key K.

b) The user enters the check-value and key K output from device A to the input interface of device B. The key K and the check-value are stored in device B.

c) When device B at some later time receives data D, it can verify the authenticity of the data using the stored values of K and check-value. Device B uses the key K to recompute the check-value as a function of the received data D. If the two check-values agree then device B accepts the data and outputs a success signal to the user. Otherwise it gives a failure signal.

The manual authentication certificate consists of K and the check-value computed as a function of K and D.

NOTE An example of data that could be included in D are a public key of a device, its identity, the domain of service, etc. In Annex A an example is provided of how manual authentication certificates can be used to establish a shared secret key between two devices.

6.3 Mechanism 2 – Devices with simple input capabilities

6.3.1 Requirements

This mechanism has the following specific requirements.

a) The mechanism specified in this subclause is appropriate for the case where both devices (A and B) have a simple input interface.

b) One of the devices (the device labelled A below) shall have the means to generate keys.

6.3.2 Specification of data exchanged

The following data exchanges and operations shall take place (see also Figure 2).

a) Both devices shall output a signal to acknowledge that they have received data D and that they are ready for the authentication mechanism to commence. On observing that both devices are ready, the user shall then enter a signal into device A to notify it that the mechanism can start.

b) Device A shall generate a random key K, where K is suitable for use with the check-value function shared by the two components. Using this key K, device A shall compute a check-value as a function of the data D. The check-value and the key K shall then be output via the output interface of device A. Device A shall also transmit the key K to device B via the communications link.

c) Device B shall use the key K to compute the check-value as a function of its stored version of data D, and shall output the key K and the computed check-value.

d) The user shall compare the two output check-values and the two output keys. If the values agree, then the user enters a signal of acceptance into both devices. If the check-values **or** the key values are different then the mechanism has failed and the user shall enter a rejection signal into the devices. The devices shall interpret the absence of an acceptance signal as a failure signal (this will require the implementation of a time-out mechanism).

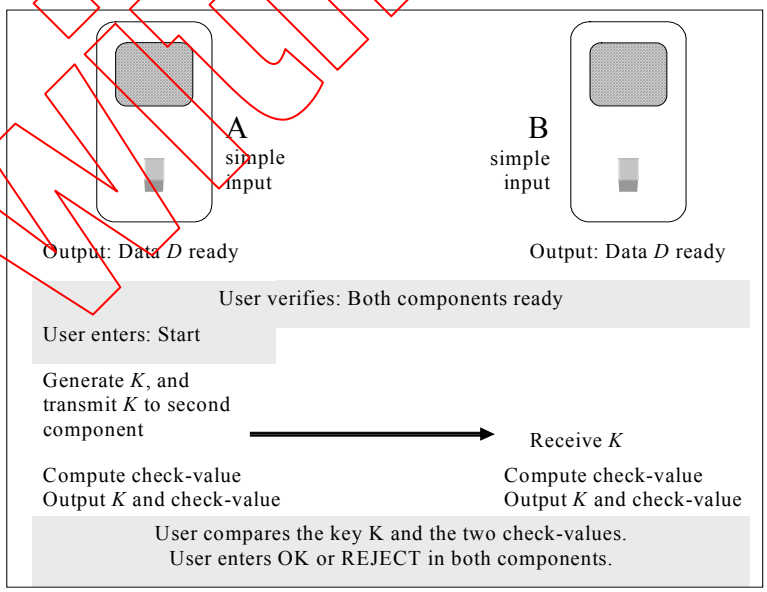

Figure 2 — Manual authentication mechanism 2

7 Mechanisms using a MAC

7.1 General

In this clause two manual authentication mechanisms are specified that are based on the use of a Message Authentication Code (MAC). The two mechanisms are appropriate for different types of devices. Specifically,

— the first mechanism (mechanism 3) is appropriate for the case where both devices have a simple output interface, and

— the second mechanism (mechanism 4) is appropriate for the case where one device has a simple input interface and the other has a simple output interface,

A standard input or output interface can emulate a simple interface, and hence if both devices have standard input and output interfaces then either of the mechanisms may be used.

Both mechanisms operate in the following general way. A data string D is transferred from one device to the other (or is the concatenation of data transferred in both directions) via the communications link between them. The manual entity authentication mechanism is then executed. As a result of the mechanism both devices are provided with assurance that the data string D they possess is the same as the value held by the other device.

7.2 Mechanism 3 – Devices with simple output capabilities

7.2.1 General

This mechanism has two variants (3a and 3b). Mechanism 3a, specified in clause 7.2.3, requires fewer interactions between the two devices, whereas mechanism 3b, specified in clause 7.2.4, requires less manual user interactions.

7.2.2 Requirements

This mechanism has the following specific requirements.

a) The two variants of the mechanism specified in this subclause are appropriate for the case where both devices (A and B) have a simple output interface.

b) Both devices shall have the means to generate random MAC keys, and the user shall have the means to generate short random bit-strings.

NOTE If the user makes poor choices for the random bit-string, e.g. the user always chooses the same value, then there is a greatly increased risk of successful attack on the mechanism.

7.2.3 Specification of data exchanged in mechanism 3a

The following data exchanges and operations shall take place (see also Figure 3). Note that steps b) and c) may occur in parallel, as may steps d)-e) and f)-g).

a) Both devices shall output, via their respective simple output interfaces, a signal to acknowledge that they have received data D and that they are ready for the authentication mechanism to commence. On observing that both devices are ready, the user shall generate a short random bit-string R. The user shall enter the random bit-string R into both devices, and shall then enter a signal into device A to notify it that the mechanism can start.

b) Device A shall generate a random key K_A, where K_A is suitable for use as a key with the MAC function shared by the two devices. Using K_A as the key, device A computes a MAC (labelled MAC_A) on the data string made up of the concatenation of I_A (an identifier for A), the data D, and the random bit-string R. Device A shall transmit MAC_A to device B via the communications link.

c) Device B shall generate a random key K_B, where K_B is suitable for use as a key with the MAC function shared by the two devices. Using K_B as the key, device B computes a MAC (labelled MAC_B) on the data string made up of the concatenation of I_B (an identifier for B), the data D, and the bit string R. Device B shall transmit MAC_B to device A via the communications link.

d) Once device A has received MAC_B (and not before), device A shall send K_A to device B.

e) On receipt of K_A, device B verifies that MAC_A equals a MAC value computed using the stored values of R, D, I_A, and the received value K_A as the key. If verification is successful, device B outputs an indication of success.

f) Once device B has received MAC_A (and not before), device B shall send K_B to device A.

g) On receipt of K_B, device A verifies that MAC_B equals a MAC value computed using the stored values of R, D, I_B and the received value K_B as the key. If verification is successful, device A outputs an indication of success.

h) The user verifies that both devices have given an indication of success, and, if so, enters a confirmation of success into both devices. If one or both of the devices give a failure indication, then the user shall enter a failure indication into both devices. If the user fails to enter a success notification into a device within a specified time interval, then this shall be interpreted as a failure of the mechanism.

NOTE Step g) in this mechanism prevents a substitution attack where the attacker tries to masquerade as device A to device B.

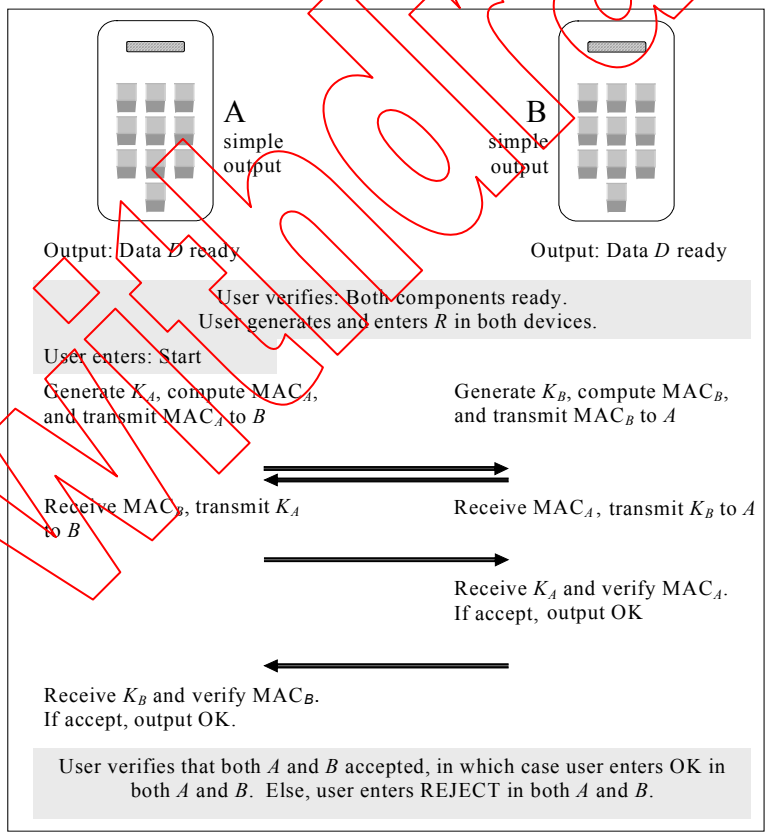

Figure 3 — Manual authentication mechanism 3a

7.2.4 Specification of data exchanged in mechanism 3b

The following data exchanges and operations shall take place (see also Figure 4).

a) Both devices shall output a signal to acknowledge that they have received data D and that they are ready for the authentication mechanism to commence. On observing that both devices are ready, the user shall generate a short random bit-string $R = (r_1,r_2,\ldots,r_n)$, where r_i is a bit and n is the number of bits in R. The user shall enter the value R into both devices, and shall then enter a signal into device A to notify it that the mechanism can start.

b) For i set consecutively to 1, 2, …, n, steps 1)-5) shall be executed. (Note that steps 1) and 2) may be executed in parallel).

1) Device A shall generate a random key K_{Ai}, where K_{Ai} is suitable for use as a key with the MAC function shared by the two devices. Using K_{Ai} as the key, device A computes a MAC (labelled MAC_{Ai}) on the data string made up of the concatenation of I_A (an identifier for A), the data D, and the random bit r_i. Device A shall transmit MAC_{Ai} to device B via the communications link.

2) Device B shall generate a random key K_{Bi}, where K_{Bi} is suitable for use as a key with the MAC function shared by the two devices. Using K_{Bi} as the key, device B computes a MAC (labelled MAC_{Bi}) on the data string made up of the concatenation of I_B (an identifier for B), the data D, and the random bit r_i. Device B shall transmit MAC_{Bi} to device A via the communications link.

3) On receipt of MAC_{Bi}, device A sends K_{Ai} to device B.

4) On receipt of MAC_{Ai} and K_{Ai}, device B verifies that MAC_{Ai}, as received in step 1, equals a MAC value computed using the stored values of r_i, D, I_A, and the received key K_{Ai}. If verification issuccessful, device B sends K_{Bi} to device A; otherwise it aborts the protocol.

5) On receipt of K_{Bi}, device A verifies that MAC_{Bi}, as received in step 3, equals a MAC value computed using the stored values of r_i, D, I_B and the received key K_{Bi}. If verification is unsuccessful, device A aborts the protocol.

NOTE In the case $i = n$, if verification is successful in steps 4) and 5), devices *B* and *A*, respectively, can output an indication of success. Whilst this is not an integral part of the protocol, it may be useful to give the device user an indication that the process has completed successfully.

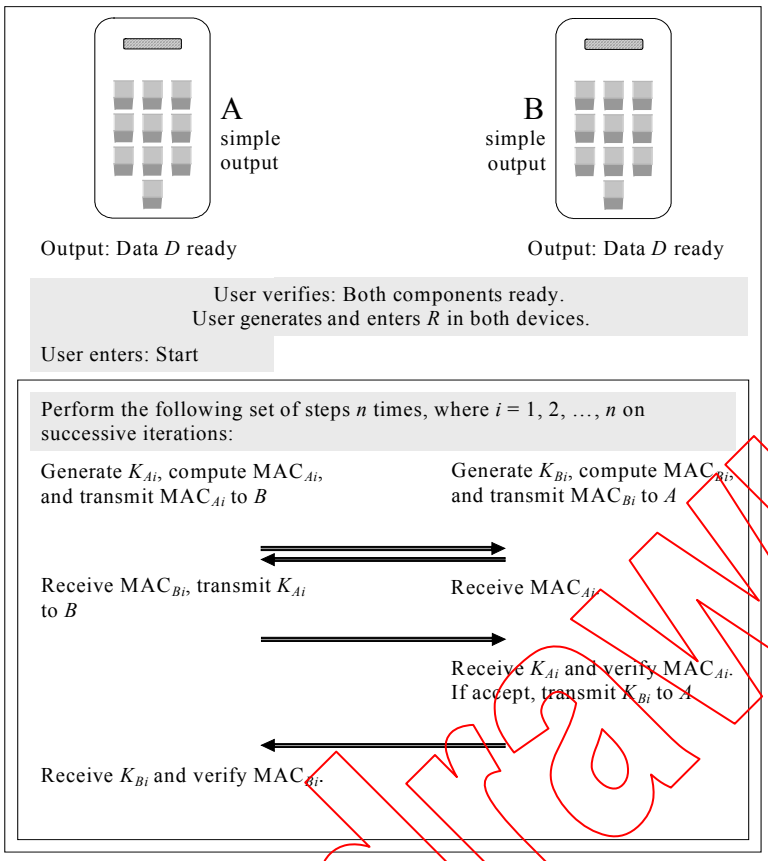

Figure 4 — Manual authentication mechanism 3b

7.3 Mechanism 4 – One device with simple input, one device with simple output

7.3.1 General

This mechanism has two variants (4a and 4b). Mechanism 4a requires fewer interactions between the two devices, whereas mechanism 4b requires less manual user interaction.

7.3.2 Requirements

This mechanism has the following specific requirements.

a) The two variants of the mechanism specified in this subclause are appropriate for the case where one device (A) has a simple input interface and the other device (B) has a simple output interface.

b) Both devices shall have the means to generate random MAC keys.

7.3.3 Specification of data exchanged in mechanism 4a

The data exchanges and operations are precisely the same as those for Mechanism 3a (as specified in clause 7.2.3) with the following exception:

— In step a) device A generates the random bit-string and displays it to the user, who copies it into device B. Thus in this mechanism the user is not required to generate a random bit-string.

This mechanism is shown in Figure 5.

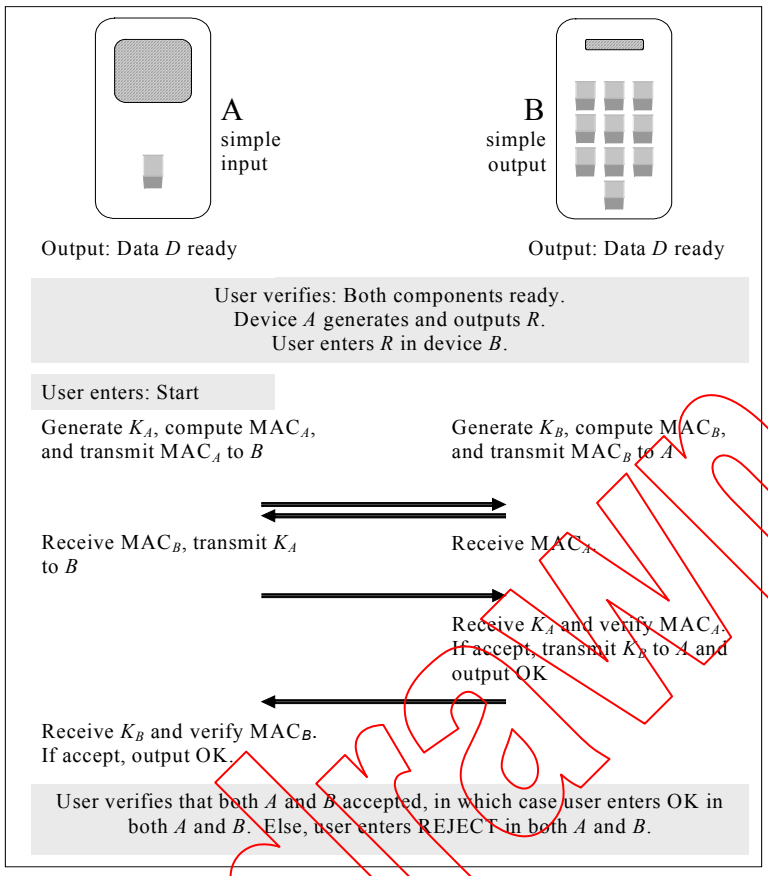

Figure 5 — Manual authentication mechanism 4a

7.3.4 Specification of data exchanged in mechanism 4b

The data exchanges and operations are precisely the same as those for mechanism 3b (as specified in clause 7.2.4) with the following exception:

— In step a) device A generates the random bit-string and displays it to the user, who copies it into device B. Thus in this mechanism the user is not required to generate a random bit-string.

Annex A
(informative)

Using manual authentication protocols for the exchange of secret keys

A.1 General

In this annex we describe methods for enabling devices to share a secret key using one of the manual authentication mechanisms specified in the body of this part of ISO/IEC 9798.

A.2 Authenticated Diffie-Hellman key agreement

The procedure described here is the Diffie-Hellman key agreement mechanism, which is authenticated using a manual authentication mechanism. The key agreement mechanism conforms to key agreement mechanism 4 of ISO/IEC 11770-3 (see also Annex B.5 of ISO/IEC 11770-3). The description given below is simplified, and for a full description the reader is referred to ISO/IEC 11770-3.

The Diffie-Hellman key agreement mechanism is described in terms of a general group G (expressed in multiplicative terminology), and an element g in G, which has a sufficiently large order. The steps of the procedure are as follows.

a) Device A generates randomly and privately an integer x, computes g^x and sends it to device *B*.

b) Device B generates randomly and privately an integer y, computes g^y and sends it to device A.

c) Devices A and B execute one of the manual authentication protocols for data D = (g^x || g^y || text) where 'text' is any additional data, e.g., each others' identifiers, that the devices may want to agree upon.

d) If the result of the manual authentication protocol is successful, then the components can compute the shared Diffie-Hellman key as $S = g^{xy}$.

The components can then derive secret cryptographic keys of the required length and format from the shared secret Diffie-Hellman key S.

A.3 Authenticated Diffie-Hellman key agreement using a manual authentication certificate

A.3.1 General

The procedure described here is the Diffie-Hellman key agreement mechanism, where one of the Diffie-Hellman public keys is authenticated using a manual authentication certificate (hence this procedure is specific to mechanism 1 and the requirements specified in subclause 6.2.1 must be met). Device B is authenticated to device A using an encrypted version of the check-value key K used in the mechanism. Note that this mechanism requires the two devices to agree and implement a symmetric encryption mechanism *e*, where $e_L(M)$ denotes the encryption of data M using secret key L. Symmetric encryption techniques are standardized in ISO/IEC 18033-3 and ISO/IEC 18033-4.

The Diffie-Hellman key agreement mechanism is described in terms of a general group G (expressed in multiplicative terminology), and an element g in G, which has a sufficiently large order. The procedure has two stages, Stage 1 and Stage 2.

In Stage 1, device A generates its Diffie-Hellman private key, computes the corresponding public key, and produces a manual authentication certificate for a data string including this public key. The certificate is transferred to device B. In Stage 2, device B receives the public key of device A, verifies it, and generates its own private and public Diffie-Hellman keys. Further, both devices compute the shared Diffie-Hellman secret, from which a secret encryption key is derived. Finally, device A verifies the encrypted version of the key K it receives from device B as part of the manual authentication process. As the result, the Diffie-Hellman secret shared by the two devices has been authenticated.

A.3.2 Stage 1

a) Device A generates randomly and privately an integer x and computes g^x. Device A then creates a manual authentication certificate on the data string D made up of g^x and any other data that needs to be reliably transferred to device B. The manual authentication certificate (K, check-value) is manually transferred to device B, and device B stores it. Device A stores x and g^x, and any other data items included within D.

A.3.3 Stage 2 (initiated by either device at some later time)

b) Device A sends g^x and possibly some other data to device B via the communications link. Device B verifies the authenticity of g^x based on the stored manual authentication certificate.

c) Device B generates randomly and privately an integer y, and computes g^y. Device B computes the Diffie-Hellman shared secret as $S = (g^x)^y$ and uses S to encrypt the key K, i.e. the key from the manual authentication certificate. Device B sends the encrypted key $e_S(K)$ and its Diffie-Hellman public key g^y to device A.

d) Device A computes its copy of the shared secret as $S = (g^y)^x$. Then it decrypts $e_S(K)$ and verifies that K is correct. If so, then device A can accept S as authenticated.

The two devices can then derive cryptographic keys of the required length and format from the shared secret Diffie-Hellman key S.

NOTE In the above description the key K from the manual authentication certificate was used in steps c) and d). Any other appropriate value, e.g., the check-value or some special purpose value, agreed between the parties at Stage 1, could be used instead.

A.4 More than two components

Means by which more than two devices can agree on a secret key using manual authentication techniques are now described.

a) One device acts as a 'master' device.

b) The master device executes the mechanism described in A.2 with every other device to establish a shared secret key with each of them.

c) Then any of the devices can generate the common secret key, which is then distributed encrypted and possibly also integrity-protected to the remainder of the devices via the master device.

If the number of devices is n, the master device needs to compute n exponentiations in the group G. Each of the other devices needs to compute two exponentiations. Therefore it is advisable to choose the master device to have sufficient computing power to perform the task.

Annex B
(informative)

Using manual authentication protocols for the exchange of public keys

B.1 General

In this annex methods are described that enable devices to reliably exchange a public key using one of the manual authentication mechanisms specified in the body of this part of ISO/IEC 9798. The context of the description is between a Certification Authority (CA) and a client of the CA. The CA needs to reliably transfer its public key to the client, and the client needs to reliably transfer its public key to the CA.

Two different cases are described, depending on whether the CA client generates its own private keys, or whether they are generated by a key management facility and then imported to the device.

B.2 Requirements

The CA must be equipped with a standard output interface, e.g. a display, and a simple input interface for giving it commands. The CA client must possess a standard input interface and a simple output interface, e.g. an audio output, to indicate success or failure of the process.

NOTE It is straightforward to adapt the procedure to the case where the CA and CA client have different types of user interfaces. Only the type of the manual authentication protocol needs to be changed.

The CA client and the CA share a communications channel.

B.3 Private key generated in device

The procedure operates as follows.

a) The CA must be reliably informed of the identifier for the CA client. This could, for example, be achieved by the user entering the identifier for the client into the input interface of the CA. However, it could also be achieved as part of the protocol itself (see below).

b) The CA sends its public key P_{CA} to the CA client, and the CA client sends its public key P_M to the CA. This transfer is assumed to take place via the (untrusted) communications link. Along with P_M, the CA client can send any other information it wishes to have included in the public key certificate, which will be generated by the CA. This could, for example, include the identifier for the client.

c) The CA and the client now perform the manual authentication mechanism specified in subclause 6.2 to verify that the exchanged public keys are correct. The CA takes the role of device A, and the client the role of device B. The data D used within the manual authentication mechanism consists of P_{CA}, P_M, and any other data supplied by the client and CA. This additional data may include the unique identifiers of the CA and the client.

d) If (and only if) the client (device B) emits a success indication, the user instructs the CA (device A) to generate an appropriate public key certificate. This certificate can then be sent (possibly unprotected) to the client via the communications link.

e) The client (device B) now performs two checks before accepting the certificate. Firstly, the client verifies the signature using the CA's public key (P_{CA}). Secondly the client verifies that the data fields within the certificate (including the public key P_M and the client identifier) are all as expected. The procedure is now complete.

B.4 Private key generated externally

If the private key of the CA client is generated by the CA or any other trusted key generation service, then the private key must be securely transported to the client.

A procedure for transporting a private key from the CA to the client is now described. The steps of the procedure are as follows:

a) The CA and the client establish a shared secret key as described in Annex A.

b) Using the secret key established in step a, the CA sends the client private key encrypted and integrity protected to the client, where it is stored securely. The CA also sends its public key (P_{CA}) integrity protected to the client, again using the secret key established in step a. Symmetric encryption techniques are standardized in ISO/IEC 18033-3 and ISO/IEC 18033-4.

c) The client now sends any information it wants to have included in its certificate to the CA, integrity protected using the secret key established in step a.

d) The CA generates the certificate for the client's public key P_M. This certificate can then be sent (possibly unprotected) to the client via the communications link.

e) The client now performs two checks before accepting the certificate. Firstly the client checks the signature using the CA's public key (P_{CA}). Secondly the client verifies that the data fields within the certificate (including the public key P_M and the identifier for the client) are all as expected. The procedure is now complete.

Annex C
(informative)

On mechanism security and choices for parameter lengths

C.1 General

In this annex the security of the four manual authentication mechanisms specified in this standard is discussed. Guidance is also provided on choices for the lengths of check-values, MACs, random bit-strings and keys.

C.2 Use of mechanisms 1 and 2

All data to be transferred via the communications link between the two devices is assumed to be public, even if in some cases part of data D may be secret. The security goal of the manual authentication mechanisms is to protect the integrity of the data, not its confidentiality. The necessary integrity protection is performed using a checking procedure based on the check-value.

A check-value function is a mapping, f, from a data space, D, and a key space, K, to a check-value space, C:

$$f : D \times K \rightarrow C, \qquad c = f(d,k).$$

In mechanisms 1 and 2 check-values are used to protect the integrity of data. Therefore in manual authentication the security is based on the unconditional security of the check-value function rather than computational security. The unconditional security of check-value functions is based on results developed by *message authentication theory,* see, for example, Section 4.5 of [11]. Two main types of attack are normally considered:

- Impersonation attacks, and
- Substitution attacks.

In an *impersonation attack*, the attacker tries to convince a receiver that data was sent by a legitimate sender without observing any prior data exchange between the sender and the receiver. In a *substitution attack*, the attacker first observes some data d and then *replaces* it with some other data $\hat{d} \neq d$. The probabilities for the attacker to succeed in an impersonation attack and a substitution attack are denoted by P_I and P_S, respectively, and they can be expressed as

$$P_I \hat{=} \max_{c \in C, d \in D} P_k(c = f(d,k)),$$

$$P_S \hat{=} \max_{\substack{c,\hat{c} \in C \\ d,\hat{d} \in D, d \neq \hat{d}}} P_k(c' = f(d',k) \mid c = f(d,k)).$$

The security of both Mechanisms depends on the probability of an attacker successfully replacing the observed data d with some other data $\hat{d} \neq d$. The attacker succeeds if $\hat{d}$ is accepted by the component as valid data. Since we assume that the two devices are physically close to each other and we do not accept any data unless both devices signal that they are ready, the impersonation attack does not apply to the manual authentication scenario. Furthermore, the normal situation for integrity protection using a MAC is that *both* the data *and* the MAC are transmitted and can be observed by the attacker. This is not the case for Mechanisms

1 and 2, which is why a check-value is used instead of a MAC in these mechanisms. Here only the data is sent over a public channel, and the attacker does not know the output of the check-value function until after the data D has been transferred (in fact, in mechanism 1 the attacker will never have access to the output of this function). This simplifies the security analysis and the expression for a successful substitution attack. Hence, the probability of successful substitution attack for Mechanisms 1 and 2 can be expressed as

$$P_S = \max_{\substack{d,\hat{d}\in D \\ d\neq\hat{d}}} P(f(d,k) = f(\hat{d},k) \mid d \text{ is observed})$$

Thus, given that the key k is chosen uniformly at random from the key space, K, the probability above can be expressed as

$$P_S = \max_{d,\hat{d}\in D, d\neq\hat{d}} \frac{|\{k \in K : f(d,k) = f(\hat{d},k)\}|}{|K|},$$

where $|K|$ denotes the cardinality of the set K. It follows from this equation that in order to provide high security, the collision probability of the check-value function must be low. This can be guaranteed by using check-value functions obtained from error correcting codes, such as the scheme specified in Annex D.

Based on the above analysis, a key length of 16-20 bits and a check-value length of 16-20 bits are recommended. Tables giving the probabilities of successful attacks for 16 and 20 bit lengths are provided in Annex D.

C.3 Use of mechanisms 3 and 4

The security of mechanisms 3 and 4 is based on different principles to those on which mechanisms 1 and 2 are based. Instead of a check-value function, a MAC function must be employed with these mechanisms. MAC functions are standardized in ISO/IEC 9797, and a MAC function from this standard is recommended for use with Mechanisms 3 and 4.

A random bit-string R of 16-20 bits is recommended for this case, but the MAC should be longer. The size of the output of the MAC function to be used for mechanisms 3 and 4 should be in the range 128–160 bits. Similarly, the random keys K_A and K_B (and K_{Ai} and K_{Bi}), used as keys for the MAC function, should be of approximately the same size, i.e. 128-160 bits. Time-out procedures should also be implemented to detect possible interruptions to the mechanism.

Annex D
(informative)

A method for generating short check-values

D.1 General

In this annex a check-value function suitable for use with mechanisms 1 and 2 is specified. The probabilities of successful attacks on these mechanisms when the proposed check-value scheme is employed are also considered. Using the expression for successful attack in Annex C, a straightforward approach is to use check-value functions derived from coding theory. The relationship between error correcting codes and such check-values is discussed in [8].

Before considering concrete examples, two basic definitions from coding theory are given. For simplicity, only codes defined over a finite field $\boldsymbol{F}_q$ are considered. Denote a q-ary code over $\boldsymbol{F}_q$ by V. Suppose the codewords have length n. The code is a mapping from messages to codewords. Each message has its corresponding unique codeword. Then the code V consists of all vectors $\boldsymbol{v} \in V = \{\boldsymbol{v}^{(d)} : d \in D\}$, $\boldsymbol{v}^{(d)} = (v_1(d), v_2(d), \ldots, v_n(d))$, where $v_i(d) \in \boldsymbol{F}_q$.

Two further definitions are required.

Definition: If x and y are two q-ary tuples of length n, then we say that their *Hamming-distance* is

$$d_H(x,y) \hat{=} |\{i \in \{1,\ldots,n\} : x_i \neq y_i\}|.$$

Definition: The *minimum distance* of a code V is

$$d_H(V) \hat{=} \min_{x,y \in V, x \neq y} d_H(x,y).$$

We now show how to create a check-value function suitable for use with Mechanisms 1 and 2 based on a code. The construction is very simple, and the mapping from the message and key space is simply obtained as

$$f(d,k) = v_k(d),$$

where $k \in K = \{1,\ldots,n\}$. Hence, a check-value function is obtained with a key size equal to n and with message space size equal to the coding space size.

The probability of a successful substitution attack for this construction is determined as follows. From the expression for P_S in Annex C, it immediately follows that:

$$P_S = \max_{d,\hat{d} \in D, d \neq \hat{d}} \frac{|\{k \in K : f(d,k) = f(\hat{d},k)\}|}{|K|} = \max_{d,\hat{d} \in D, d \neq \hat{d}} \frac{|\{k \in K : v_k(d) = v_k(\hat{d})\}|}{|K|}$$

$$= \max_{d,\hat{d} \in D, d \neq \hat{d}} \frac{n - d_H(v^{(d)}, v^{(\hat{d})})}{n} = 1 - \frac{d_H(V)}{n}.$$

Given this exact expression for the probability of successful substitution attack, it is now appropriate to consider some concrete constructions. Rather long codes with very high minimum distance are required.

This property holds for the well-known Reed-Solomon (RS) codes [10]. An RS code can be constructed over an arbitrary finite field, $\boldsymbol{F_q}$. The calculation of a codeword is very simple and involves polynomial evaluation over the finite field. Express the data (message) to be encoded as a q-tuple of length t over $\boldsymbol{F_q}$, $d = d_0, d_1, .., d_{t-1}, d_i \in \boldsymbol{F_q}$. Then, the generalized RS encoding polynomial is given by

$$p^{(d)}(x) = d_0 + d_1 x + d_2 x^2 + \ldots + d_{t-1} x^{t-1}.$$

The check-value function is directly given by evaluating the polynomial at an arbitrary point $k \in \boldsymbol{F_q}$,

$$f(d,k) = v_k(d) = p^{(d)}(k) = d_0 + d_1 k + d_2 k^2 + \ldots + d_{t-1} k^{t-1}.$$

The generalized RS code has the following properties ([8]):

$n = q = |K|$,

$|D| = q^t = n^t$,

$d_H(V) = n - t + 1$.

This implies that $P_S = (t\text{-}1)/n$ for a check-value obtained from the generalized RS code. The probability increases with the size of the message space, D. Hence, a good approach is to **first** apply a good one-way hash-function, such as one of the dedicated hash-functions specified in ISO/IEC 10118-3, to the data and then use the output from the one-way hash-function as input to the Reed-Solomon code. This implies that we keep a low probability without significantly increasing the key length or the length of the output of the check-value function. By using this approach, a message space of around 128 bits (truncated SHA-1) gives sufficient security. In Table 1 two construction examples and the corresponding probabilities of successful attacks are given.

Table D.1 — RS code check-values: probability of successful substitution attack, P_S

$\log_2\|D\|$	$\log_2(n)$	P_S
128	16	2^{-13}–2^{-16}
256	16	2^{-12}–2^{-16}
128	20	2^{-17}–2^{-20}
256	20	2^{-16}–2^{-20}

As can be seen from the table, a code with a 4 hexadecimal digit key and check-value gives a forgery probability of around 2^{-12} or less. If this is increased to 5 hexadecimal digits, the probability decreases to around 2^{-17} or less.

Bibliography

[1] C. Gehrmann and K. Nyberg, 'Enhancements to Bluetooth baseband security', in *Proceedings of Nordsec 2001, Copenhagen, Denmark*, November 2001.

[2] ISO 7498-2: 1994, *Information processing systems – Open systems interconnection – Basic reference model – Part 2: Security architecture*.

[3] ISO/IEC 9797 (all parts), *Information technology – Security techniques – Message Authentication Codes (MACs)*.

[4] ISO/IEC 10118-3: 2004, *Information technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions*, 3rd edition.

[5] ISO/IEC 11770-3: 1999, *Information technology – Security techniques – Key management – Part 3: Mechanisms using asymmetric techniques*.

[6] ISO/IEC 18033-3 (to be published), *Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers*.

[7] ISO/IEC 18033-4 (to be published), *Information technology – Security techniques – Encryption algorithms – Part 4: Stream ciphers*.

[8] G. Kabatianskii, B. Smeets and T. Johansson, 'On the cardinality of systematic authentication codes via error correcting codes', *IEEE Transactions on Information Theory*, **IT-42** (1996) 566-578.

[9] J.-O. Larsson, 'Higher layer key exchange techniques for Bluetooth security', in *Opengroup Conference, Amsterdam*, October 2001.

[10] I. S. Reed and G. Solomon, 'Polynomial codes over certain finite fields', *SIAM Journal* **8** (1960) 300-304.

[11] D. Stinson, *Cryptography – Theory and Practice*, CRC Press, 2002, 2nd edition.

[12] SHAMAN Project Deliverable D13 (Annex 2), *Final technical report – Workpackage 2 – Security for distributed terminals*, 2003. Available at www.ist-shaman.org.

ICS 35.040

Price based on 20 pages