INTERNATIONAL STANDARD

ISO/IEC 20009-4

First edition
2017-08

Information technology — Security techniques — Anonymous entity authentication —

Part 4: Mechanisms based on weak secrets

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

Partie 4: Mécanismes basés sur des secrets faibles

Reference number
ISO/IEC 20009-4:2017(E)

Contents

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

A list of all parts in the ISO/IEC 20009 series can be found on the ISO website.

Introduction

Inputting a user's "identity (ID)" together with a "password" has almost certainly been the most common method of user authentication since the advent of computers and remains very widely used. Every day, there are probably billions of instances of password-based user authentications in cyberspace. One reason for the wide acceptance of password-based authentication is portability; no dedicated device is required, and a user needs only memorize a password and can then be authenticated anywhere and anytime. ISO/IEC 11770-4 specifies key management mechanisms that are based on passwords (usually passwords are weak secrets). These mechanisms can be used to achieve password-based entity authentication.

Individual privacy in cyberspace is an area of increasing concern. Protection of user privacy during entity authentication is a critical step towards individual privacy protection in cyberspace. ISO/IEC 20009 specifies privacy preserving entity authentication techniques, supporting anonymous entity authentication. This document focuses on anonymous entity authentication mechanisms based on weak secrets. In particular, it specifies password-based anonymous entity authentication (PAEA) mechanisms that enable password authentication with simultaneous protection of user privacy.

PAEA mechanisms need to address the fact that use of a weak secret such as a password with an anonymous authentication mechanism intended to be used with a strong secret cannot protect user privacy because a weak secret reveals information. This document specifies two types of PAEA mechanisms: password-only PAEA mechanisms and storage-extra PAEA mechanisms. In a password-only PAEA mechanism, users register their password verification data at the authentication server and remember their passwords in the same way as when using non-anonymous password authentication mechanisms. In a storage-extra PAEA mechanism, users not only remember their passwords, but also hold password-wrapped credentials that can be revealed to adversaries without compromising user privacy. In mechanisms of the latter type, user password verification data are not saved at the server. Mechanisms of both types have advantages in certain scenarios.

NOTE Annex A gives object identifiers for the PAEA mechanisms specified in this document.

International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Information may be obtained from:

National Institute of Advanced Industrial Science and Technology
1-1-1 Umezono
Tsukuba
Ibaraki 305-8560
Japan

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and/or IEC shall not be held responsible for identifying any or all such patent rights.

ISO (www.iso.org/patents) and IEC (http://patents.iec.ch) maintain on-line databases of patents relevant to their standards. Users are encouraged to consult the databases for the most up to date information concerning patents.

Information technology — Security techniques — Anonymous entity authentication —

Part 4: Mechanisms based on weak secrets

1 Scope

This document specifies anonymous entity authentication mechanisms based on weak secrets. The precise operation of each mechanism is specified, together with details of all inputs and outputs. This document is applicable to situations in which the server only verifies that the user belongs to a certain user group without obtaining any information that can be used to identify the user later on.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements 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 9797-2, *Information technology — Security techniques — Message authentication codes (MACs) — Part 2: Mechanisms using a dedicated hash-function*

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

ISO/IEC 11770-4:2006, *Information technology — Security techniques — Key management — Part 4: Mechanisms based on weak secrets*

ISO/IEC 18033-4, *Information technology — Security techniques — Encryption algorithms — Part 4: Stream ciphers*

ISO/IEC 19772:2009, *Information technology — Security techniques — Authenticated encryption*

ISO/IEC 20009-1, *Information technology — Security techniques — Anonymous entity authentication — Part 1: General*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 20009-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at http://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
abelian group
group $(S, *)$ such that $a*b = b*a$ for every a and b in S

[SOURCE: ISO/IEC 15946-1:2016, 3.1]

3.2
authenticated encryption
(reversible) transformation of data by a cryptographic algorithm to produce ciphertext that cannot be altered by an unauthorized entity without detection, i.e. provides data confidentiality, data integrity, and data origin authentication

[SOURCE: ISO/IEC 19772:2009, 3.1]

3.3
authentication credential
credential (3.7) containing information that can be used to help authenticate the entity

3.4
authenticator
data string that is sent to and verified by the other entity as part of the mechanism

3.5
claimant
entity which is or represents a principal for the purposes of authentication

Note 1 to entry: A claimant includes the functions and the private data necessary for engaging in authentication exchanges on behalf of a principal.

[SOURCE: ISO/IEC 9798-1:2010, 3.6]

3.6
collision-resistant hash-function
hash-function satisfying the following property: it is computationally infeasible to find any two distinct inputs which map to the same output

Note 1 to entry: Computational feasibility depends on the specific security requirements and environment.

[SOURCE: ISO/IEC 10118-1:2016, 3.1]

3.7
credential
representation of an identity

Note 1 to entry: A credential is typically made to facilitate data authentication of the identity information in the identity it represents.

Note 2 to entry: The identity information represented by a credential can be printed on paper or stored within a physical token that typically has been prepared in a manner to assert the information as valid.

EXAMPLE A credential can be a username, a username with a password, a PIN, a smartcard, a token, a fingerprint, a passport, etc.

[SOURCE: ISO/IEC 24760-1:2011, 3.3.5]

3.8
cyclic group
abelian group (3.1) (G, *) such that there exists an element g in G, called the generator of the group, such that for any element a in G, there exists an integer i with $a = g^i$

3.9
distinguishing identifier
information which unambiguously distinguishes an entity

[SOURCE: ISO/IEC 11770-1:2010, 2.9]

3.10
exhaustive search
attack technique, also known as brute-force, involving searching through all possibilities for a secret value

3.11
field
set of elements S and a pair of operations (+, *) defined on S such that: (i) $a*(b + c) = a*b + a*c$ for every a, b and c in S, (ii) S together with + forms an *abelian group* (3.1) (with identity element 0), and (iii) S excluding 0 together with * forms an abelian group

[SOURCE: ISO/IEC 15946-1:2016, 3.4]

3.12
finite field
field containing a finite number of elements

Note 1 to entry: For any positive integer, m, and a prime, p, there exists a finite field containing exactly p^m elements. This field is unique up to isomorphism and is denoted by $F(p^m)$, where p is called the characteristic of $F(p^m)$.

[SOURCE: ISO/IEC 15946-1:2016, 3.5]

3.13
group
set of elements S and an operation * defined on the set of elements such that (i) $a*(b*c) = (a*b)*c$ for every a, b and c in S, (ii) there exists an identity element e in S such that $a*e = e*a = a$ for every a in S, and (iii) for every a in S, there exists an inverse element a^{-1} in S such that $a*a^{-1} = a^{-1}*a = e$

[SOURCE: ISO/IEC 15946-1:2016, 3.6]

3.14
group generator
generator of a cyclic group

3.15
homomorphic encryption
symmetric or asymmetric encryption that allows third parties to perform operations on plaintext data while keeping them in encrypted form

Note 1 to entry: Third parties refer to parties that are neither the encryptor nor the decryptor.

3.16
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 two 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 a set of input strings and corresponding function values, where the value of the ith input string might have been chosen after observing the value of the first i-1 function values (for integers $i > 1$).

Note 1 to entry: A MAC algorithm is sometimes called a cryptographic check function (see for example ISO 7498-2).

Note 2 to entry: Computational feasibility depends on the user's specific security requirements and environment.

[SOURCE: ISO/IEC 9797-1:2011, 3.10]

3.17
offline exhaustive search
exhaustive search or brute-force attack that is performed without interacting with any of the authorized parties

3.18
password
secret word, phrase, number or character sequence used for entity authentication, which is a memorized weak secret

[SOURCE: ISO/IEC 11770-4:2006, 3.25]

3.19
password verification data
data that is used to verify an entity's knowledge of a specific *password* (3.18)

[SOURCE: ISO/IEC 11770-4:2006, 3.29]

3.20
password wrapped credential
authentication credential (3.3) generated as a function of a *password* (3.18) in a protected format that resists *offline exhaustive search* (3.17) of passwords

3.21
prime field
finite field (3.12) containing a prime number of elements

3.22
pseudo distinguishing identifier
information which unambiguously distinguishes an entity, but which is valid for only a limited period of time

3.23
secure prime
prime integer p such that $(p - 1) / 2$ has only large prime factors

3.24
system parameters
choice of parameters that selects a particular cryptographic scheme or function from a family of cryptographic schemes or functions

[SOURCE: ISO/IEC 18033-2:2006, 3.41]

4 Symbols, abbreviated terms and conversion functions

4.1 Symbols and abbreviated terms

For the purposes of this document, the symbols and abbreviated terms given in ISO/IEC 20009-1 and the following apply.

AD_K	decryption function for an authenticated encryption scheme using K as the key; that shall be selected from among those standardized in ISO/IEC 19772
AE_K	encryption function for an authenticated encryption scheme using K as the key; that shall be selected from among those standardized in ISO/IEC 19772
$a \bmod n$	for an integer a and a positive integer n, the unique integer $r \in [0, n - 1]$ such that $r \equiv a \pmod n$
$a \equiv b \pmod n$	for a non-zero integer n, a relation between integers a and b that holds if and only if a and b are congruent modulo n, i.e. $n \mid (a - b)$
$a^{-1} \bmod n$	for an integer a and a positive integer n such that $\gcd(a, n) = 1$, the unique integer $b \in [1, n - 1]$ such that $ab \equiv 1 \pmod n$

CRL	credential revocation list
DEC_{pw}	decryption function for a stream cipher algorithm using a key derived from a password, pw, (and using a random initial value if the algorithm requires it); shall be selected from among those standardized in ISO/IEC 18033-4
ENC_{pw}	encryption function for a stream cipher algorithm using a key derived from a password, pw, (and using a random initial value if the algorithm requires it); shall be selected from among those standardized in ISO/IEC 18033-4
$E(F(p))$	a set of $F(p)$-valued points of an elliptic curve and an extra point O_E referred to as the point at infinity (see ISO/IEC 15946-1)
$F(p)$	a finite field containing p elements, where p is a prime or a power of a prime
$\mathrm{gcd}(a, b)$	for integers a and b, the greatest common divisor of a and b, i.e. the largest positive integer that divides both a and b (or 0 if $a = 0$ or $b = 0$)
g^x	if g is an element of the finite field $F(p)$, then this denotes g^x computed in $F(p)$ where g is a generator of a cyclic group of prime order, q. If g is an elliptic curve point, then this denotes an elliptic curve scalar multiplication $[x]g$, where x is a scalar and g is a base point of prime order q on the elliptic curve
$g_1{*}g_2$	if g_1 and g_2 are elements of the finite field $F(p)$, then this denotes $g_1{*}g_2$ computed in $F(p)$. If g_1 and g_2 are elliptic curve points, then this denotes $g_1 + g_2$ for the cyclic group operation + defined over the elliptic curve
g_1/g_2	if g_1 and g_2 are elements of the finite field $F(p)$, then this denotes g_1/g_2 computed in $F(p)$. If g_1 and g_2 are elliptic curve points, then this denotes $g_1 - g_2$ for the cyclic group operation − defined over the elliptic curve
H	a collision-resistant hash-function taking a bit/octet string as input and giving a bit/octet string as output; shall be selected from among those standardized in ISO/IEC 10118
HD_{sk}	decryption function under a private key sk of a homomorphic encryption scheme
HE_{pk}	encryption function under a public key pk of a homomorphic encryption scheme
H_g	a collision-resistant hash-function taking a bit/octet string as input and giving a generator of a cyclic group of prime order, q, as output, which shall be made up of the composition of a hash-function selected from among those standardized in ISO/IEC 10118-3 and a function $R_{1\mathrm{DL}}$ or $R_{1\mathrm{EC}}$ in ISO/IEC 11770-4 that converts a bit string to a cyclic group element
H_λ	a collision-resistant hash-function taking a bit/octet string as input and giving a bit/octet string of bit length, λ, as output; shall be selected from among those standardized in ISO/IEC 10118-3 and whose output is truncated to λ bits
I_S	octet string representing the distinguishing identifier of the authentication server S
I_U	octet string representing the distinguishing identifier of a user U, a claimant or a group of users
I_{U_i}	octet string representing the distinguishing identifier of a user U_i from a group of users $\mathrm{U}_1, \mathrm{U}_2, \ldots, \mathrm{U}_n$
$\underline{I}_{\mathrm{U}_i}$	octet string representing the pseudo distinguishing identifier for a user U_i

MAC_λ(MK, MESSAGE)	a Message Authentication Code (MAC) algorithm taking a bit/octet string MK as the MAC generation key and a bit/octet string MESSAGE to be authenticated as input and giving a bit/octet string of bit length λ as output; shall be selected from among those standardized in ISO/IEC 9797-2
Ocheck	a function to check the order of a given group element is a prime integer, q, or larger, which depends on the type of the group element to check, 1) for a group element in a prime field $F(p)$ and p is a secure prime, it takes an integer b and outputs 'valid' if b is not equal to −1, 0 or 1 (mod p) otherwise outputs 'invalid'; 2) for a group element in a prime field $F(p)$ and p is not a secure prime, it takes an integer b and outputs 'valid' if b is not equal to −1, 0 or 1 (mod p) and $b^q \equiv 1$ (mod p) otherwise outputs 'invalid' and 3) for an elliptic curve point on an elliptic curve $E(F(p))$, it takes an element b in $F(p) \times F(p)$ and outputs 'valid' if b is on the elliptic curve and $b^{\#E/q}$ is not equal to the point at infinity O_E otherwise outputs 'invalid'
O_E	the elliptic curve point at infinity
PWF	a password file
params	public system parameters
pvd_U	password verification data associated with the user U
pw_U	a password associated with the user U
ServK	server key kept secret by the authentication server
SK	session key
secparam	security parameter, i.e. a number representing the number of bits of security to be offered by an instantiation of a scheme NOTE A typical choice should be 128 or more.
$\tilde{x} \| \tilde{y}$	if $\tilde{x}$ and $\tilde{y}$ are bit/octet strings, the concatenation of the two strings $\tilde{x}$ and $\tilde{y}$, resulting in the string consisting of $\tilde{x}$ followed by $\tilde{y}$
$\lvert x \rvert$	bit length of a number x
λ_k	bit length of a symmetric key k
λ_U	bit length of a pseudo distinguishing identifier I_U
λ_v	bit length of an authenticator denoted by V_U, V'_U, V_S and V'_S
$\#E$	the number of rational points on the elliptic curve $E(F(p))$, including the point at infinity O_E (see ISO/IEC 15946-1)
$[i, j]$	the set of integers m satisfying $i \le m \le j$
$<x_1, \ldots, x_l>$	an l-tuple of elements $x_1, \ldots, x_l$
$\perp$	empty

4.2 Conversion functions

EC2OSP	elliptic curve to octet string conversion primitive that is equivalent to the recommended conversion function EC2OS(, *compressed*) in ISO/IEC JTC 1/SC 27/WG 2 Standing Document 7
FE2OSP	field element to octet string conversion primitive as specified in ISO/IEC JTC 1/SC 27/WG 2 Standing Document 7
GE2OSP	cyclic group element to octet string conversion primitive. If the mechanism is defined over a finite field, it is the same as FE2OS. If it is defined over an elliptic curve, it is the same as EC2OSP

5 General model for password-based anonymous entity authentication

5.1 Participants

The participants in a password-based anonymous entity authentication (PAEA) mechanism are an authentication server and users who belong to certain groups.

NOTE As an example, a user group can comprise members who qualify to obtain a certain service or who have a certain attribute such as being over 18 years old, male or female, etc.

5.2 Types of PAEA mechanisms

Two types of PAEA mechanisms are specified in this document: password-only and storage-extra mechanisms. In a password-only PAEA mechanism, each user registers a password with the authentication server and, as a result, the server holds the password verification data; for authentication purposes, the password is the only long-term secret that a user needs and the corresponding password verification data is the only long-term secret that the server needs. In a storage-extra PAEA mechanism, each user registers a password with the authentication server and, as a result, gets back a password wrapped credential, but the server does not hold the password verification data. The user needs a storage device to manage the password wrapped credential, but the storage device itself needs no further protection (that is to say, a password-wrapped credential can be put in a public directory). At the point of authentication, a user first needs to recover the authentication credential from the user’s password wrapped credential using the password, and then proves that the user is a member of a certain user group using the recovered authentication credential without revealing the user's identity to the server.

5.3 Components of a password-only PAEA

A password-only PAEA mechanism consists of the following operations.

Setup: Given a security parameter *secparam*, generate a tuple <*params*, $PWF = \perp$>, where *params* denotes public system parameters and *PWF* is a password file, initially set empty, that is kept private. The password file *PWF* consists of a set of ordered pairs, <I_U, pvd_U>, one for each member of the user group, where I_U and pvd_U are the distinguishing identifier and password verification data for the user, U, respectively. *params* will be the common input to the other operations of the mechanism, but for simplicity, it is deliberately omitted from the specifications of inputs of the operations.

User registration: Given a user's distinguishing identifier, I_U and a password, pw_U, from the user and a password file, *PWF*, from the server, generate password verification data, pvd_U, and then add <I_U, pvd_U> to the password file, *PWF*.

Anonymous authentication: This is an interactive process that takes place between a user and the authentication server. Given a password, pw_U, associated with user, U, from the user, and a password file, *PWF*, from the server, the two entities authenticate each other (and optionally negotiate a common session key). The user authentication is performed anonymously, i.e. the server only verifies that the user belongs to a certain user group without obtaining any information that can be used to identify the user later. The anonymous authentication operation outputs ACCEPT (and optionally a session key) only if both entities output ACCEPT. Otherwise, it outputs REJECT.

User revocation: Given a user identity, I_U, and a password file, *PWF*, remove $<I_U, pvd_U>$ from the password file, *PWF*, and then update the pre-computed values, if any.

NOTE 1 A basic requirement of any password-only PAEA mechanism is correctness. For any pair of $<I_U, pvd_U>$, the anonymous authentication operation should accept U if and only if $<I_U, pvd_U>$ is in *PWF*.

NOTE 2 The anonymous authentication operation is specified to include mutual authentication and key exchange. Depending on the application, it can be minimally specified to include only unilateral authentication from a user to a server or a server authenticates only that the user belongs to a certain user group, without the opposite direction of authentication and key exchange.

5.4 Components of a storage-extra PAEA

A storage-extra PAEA mechanism consists of the following operations.

Setup: Given a security parameter, *secparam*, generate a tuple <*params*, *ServK*>, where *params* denotes public system parameters and *ServK* denotes a server key kept secret by the server. *params* is an input to all other operations of the mechanism, but for simplicity, it is omitted from the specifications below.

User registration: Given a user's distinguishing identifier, I_U, and a password, pw_U, from the user and the server key, *ServK*, from the server, output a password wrapped credential, $cred_{pw_U}$. $cred_{pw_U}$ may be stored anywhere accessible to the user, e.g. in a portable storage device or a public directory.

Anonymous authentication: This is an interactive process that takes place between a user and the authentication server. Given a password wrapped credential, $cred_{pw_U}$, and a password, pw_U, from the user and the server key, *ServK*, from the server, the two entities authenticate each other (and optionally negotiate a common session key). The user authentication is performed anonymously, i.e. the server only verifies that the user belongs to a certain user group without obtaining any information that can be used to identify the user later. The anonymous authentication operation outputs ACCEPT (and optionally a session key) only if both entities output ACCEPT. Otherwise, it outputs REJECT.

User revocation: Given a user's distinguishing identifier, I_U, and password wrapped credential, $cred_{pw_U}$, revoke the authentication credential for I_U and update *CRL* (Credential Revocation List) such that $CRL_{new} = CRL_{old} \cup \{cred\}$, where CRL_{new} is the updated *CRL* and CRL_{old} is the old *CRL* and *cred* is the original credential contained in $cred_{pw_U}$. *CRL* is part of *params* and is initially set empty.

NOTE 1 A basic requirement of any storage-extra PAEA mechanism is correctness. For any pair of $< pw_U, cred_{pw_U} >$ associated with the same user, the anonymous authentication operation should accept if and only if $cred \notin CRL$.

NOTE 2 The anonymous authentication operation is specified to include mutual authentication and key exchange. Depending on the application, it can be minimally specified to include only unilateral authentication from a user to a server or a server authenticates only that the user belongs to a certain user group, without authentication from a server to a user and key exchange.

5.5 Operation of a PAEA

The authentication server performs the setup process by invoking the setup operation, in which it generates system parameters and, additionally, a server key in the case of a storage-extra PAEA mechanism. The system parameters are made public, while the server key is kept private by the server.

When a user enrols in the system, the user registers the user's password at the server by invoking the user registration operation. The server generates and keeps password verification data based on the user password in the case of a password-only PAEA mechanism. In contrast, in the case of a storage-extra PAEA mechanism, the server generates nothing for itself, but issues to the user a password-wrapped credential which is generated from an authentication credential protected using the user's password.

A user and the server perform mutual authentication and key exchange using the anonymous authentication operation.

The server can revoke a user's authentication privilege by invoking the user revocation operation, where, in the case of a storage-extra PAEA mechanism, it in effect revokes the user's authentication credential.

6 Password-only PAEA mechanisms

6.1 General

NOTE 1 The SKI mechanism is based on Reference [4] and the associated security analysis is also given in Reference [4].

NOTE 2 The YZ mechanism is based on Reference [6] and the associated security analysis is also given in Reference [6].

6.2 SKI mechanism

6.2.1 Setup

Given a security parameter *secparam*, this operation generates or otherwise determines the following public system parameters.

a) A cyclic group, G, of prime order, q, and its generator, g, where

   1) the bit length of q shall be at least twice the value of *secparam*,

   2) G shall be chosen so that the difficulty of the discrete logarithm problem (measured in bits) on it shall be greater than or equal to *secparam*,

   3) when g is an element of the finite field $F(p)$, p should be a secure prime such that $(p - 1)/q = 2q_1q_2...q_t$ for primes $q_i > q$ for $1 \leq i \leq t$ (see ISO/IEC 11770-4:2006, Clause 5), to reduce the computational burden of checking the order of the group elements received from the counterpart party, or checking the order of them in the operation of the anonymous authentication mechanism if p is not such a secure prime, and

   4) when g is an elliptic curve point on an elliptic curve $E(F(p))$, the cofactor of $E(F(p))$, i.e. $\#E/q$, shall be 1 (or 2 if 1 is not possible) to reduce the computational burden of checking the order of the group elements received from the counterpart party.

b) An order check function *Ocheck*(), a MAC (Message Authentication Code) algorithm MAC_λ(MK, MESSAGE), collision-resistant hash-functions, $H_g()$ and $H_\lambda()$ and authenticated encryption operations $AE_K()$ and $AD_K()$, as shown in 4.1, respectively.

NOTE ISO/IEC 15946-1 contains recommendations for the generation of the cyclic groups and their generators on elliptic curves, and in most cases $\#E/q$ is 1 for elliptic curves over a prime field and 2 for those over a binary extension field.

6.2.2 User registration

The user registration operation works as follows.

a) Either a user U_i or a server S generates the user's password verification data, pvd_{U_i} where $pvd_{U_i} = H_g(I_{U_i} || pw_i)$ and I_{U_i} and pw_i are the user's distinguishing identifier and password, respectively.

b) The pair $< I_{U_i}, pvd_{U_i} >$ is added to the password file *PWF* by S.

c) U_i is added to a legitimate user group denoted by U.

d) U_i obtains a trusted copy of the public system parameters chosen in 6.2.1.

NOTE U_i can be added to multiple user groups as long as it is legitimate to be included in them.

6.2.3 Anonymous authentication

6.2.3.1 General

The anonymous entity authentication operation allows every member of a group of users $\{U_1, U_2, \ldots, U_n\}$ with identifier I_U to be anonymously authenticated by a server S, and, simultaneously, the server to be authenticated by the user. This operation consists of three phases: two preparation phases (which may be conducted in advance of the authentication protocol execution) and the protocol execution phase.

6.2.3.2 Preparation phase for *X*

In this phase, the server S pre-computes values X and $< \underline{I}_{U_j}, K_j >$ for $1 \leq j \leq n$ as follows, where n is the number of users in the user group specified by U.

a) Chooses a random integer x from $[2, q - 1]$.

b) Computes corresponding public value $X = g^x$.

c) Computes the user U_j's pseudo distinguishing identifier $\underline{I}_{U_j}$ of $\lambda_{\underline{U}}$ bits and a symmetric-key K_j of λ_k bits with $\left(\underline{I}_{U_j} \| K_j\right) = H_{\lambda_{\underline{U}} + \lambda_k}\left(I_{U_j} \| \mathrm{GE2OSP}(X) \| \mathrm{GE2OSP}\left(pvd_{U_j}\right) \| \mathrm{GE2OSP}\left(\left(pvd_{U_j}\right)^x\right)\right)$ for $1 \leq j \leq n$.

6.2.3.3 Preparation phase for C_j

In this phase, the server S pre-computes values C_j for $1 \leq j \leq n$ as follows, where n is the number of users in the user group specified by U.

a) Chooses a random master secret $\mathrm{MS} \in \{0, 1\}^{\lambda_k}$.

b) Generates a ciphertext $C_j = AE_{K_j}(\mathrm{MS})$ for each of the n users U_j for $1 \leq j \leq n$. The server determines a validity period t for the generated C_j and then sets the current time to t_0 as the generated time of a set of pairs $< \underline{I}_{U_j}, C_j >$ for $1 \leq j \leq n$, X and t.

6.2.3.4 Protocol execution phase

In this phase, a user U_i who is a member of the user group specified by U is anonymously authenticated by the server and U_i authenticates the server.

— **Step 1.** U_i initiates the protocol as follows.

a) Sends the user group's distinguishing identifier, I_U, to the server, S, and then obtains the generated time, t_0, of the server's current value of X and pairs of $< \underline{I}_{U_j}, C_j>$ for $1 \leq j \leq n$.

b) If the obtained t_0 is the same as the value the user obtained in the previous protocol execution or obtained from a public bulletin board, the user may skip obtaining $< \underline{I}_{U_j}, C_j>$ for $1 \leq j \leq n$, X and t from the server. Otherwise, the user obtains them from the server.

c) After the reception of the new value of X from S, U_i checks it. If $Ocheck(X)$ is invalid, the user outputs REJECT and terminates this anonymous authentication.

d) Chooses two integers a and b uniformly at random from $[2, q - 1]$, and then computes $B = g^b$ and $A' = g^a$.

e) Computes $pvd_{U_i} = H_g(I_{U_i} || pw_i)$ using the user's password and distinguishing identifier.

f) Computes $A = pvd_{U_i} * A'$.

g) Sends A and B to the server S.

— **Step 2.** While waiting for A and B, the server S performs the following steps.

a) Chooses an integer y uniformly at random from $[2, q - 1]$, computes $Y = g^y$ and looks for the value of x corresponding to U.

b) After the receipt of A and B, checks them. If either of $Ocheck(A)$ and $Ocheck(B)$ is invalid, the server outputs REJECT and terminates the anonymous authentication procedure.

c) Computes A^x, B^x and B^y using the exponents x and y.

d) Computes $MK = H(MS||GE2OSP(B^x)||GE2OSP(B^y))$, $V_S = MAC_{\lambda_v}$ (MK,1||U||S||GD||PD), $V_U = MAC_{\lambda_v}$ (MK,2||U||S||GD||PD), (and optionally a session key $SK = MAC_{\lambda_k}$ (MK,0||U||S||GD||PD)), where

1) PD = (GE2OSP(A)||GE2OSP(B)||GE2OSP(A^x)||GE2OSP(X)||GE2OSP(Y)), and

2) GD = $H(\{< \underline{I}_{U_j}, C_j>\}_{1 \leq j \leq n} || t || t_0)$ and "$\{< \underline{I}_{U_j}, C_j>\}_{1 \leq j \leq n}$" means the concatenation of the 2-tuple of elements $\underline{I}_{U_j}$ and C_j for $1 \leq j \leq n$.

e) Sends A^x, Y and V_S with its distinguishing identifier I_S to U_i.

— **Step 3.** After the receipt of the message from the server S, the user U_i performs the following.

a) Checks them. If either of $Ocheck(A^x)$ and $Ocheck(Y)$ is invalid, the user outputs REJECT and terminates the anonymous authentication procedure.

b) Computes $(\underline{I}_{U_i} || K_i) = H_{\lambda_{\underline{U}} + \lambda_k}(I_{U_i} ||$GE2OSP$(X)||$GE2OSP$(pvd_{U_i})||$GE2OSP$(A^x/X^a))$, finds the pair $< \underline{I}_{U_j}, C_j >$ such that $\underline{I}_{U_i} = \underline{I}_{U_j}$ from the list of $< \underline{I}_{U_j}, C_j >$ for $1 \leq j \leq n$ sent by S or posted on the public bulletin board, and then lets $C_i = C_j$.

c) Decrypts the ciphertext, C_i, as follows: MS′ = $AD_{K_i}(C_i)$.

d) Computes a MAC key MK′ = H(MS′||GE2OSP(X^b)||GE2OSP(Y^b)), $V'_S = MAC_{\lambda_v}$(MK′,1||U||S||GD||PD), $V'_U = MAC_{\lambda_v}$(MK′,2||U||S||GD||PD) (and optionally a session key SK = MAC_{λ_k}(MK′,0||U||S||GD||PD)), where

1) PD = (GE2OSP(A)||GE2OSP(B)||GE2OSP(A^x)||GE2OSP(X)||GE2OSP(Y)), and

2) GD = $H(\{< \underline{I}_{U_j}, C_j >\}_{1 \leq j \leq n} || t || t_0)$ and "$\{< \underline{I}_{U_j}, C_j >\}_{1 \leq j \leq n}$" means that concatenation of 2-tuple of elements $\underline{I}_{U_j}$ and C_j for $1 \leq j \leq n$.

e) If $V_S = V'_S$, sends V'_U to S and outputs ACCEPT (and SK optionally). Otherwise, U_i outputs REJECT and terminates the protocol.

— **Step 4.** After the receipt of the message from user U_i, the server, S, performs the following.

a) Outputs ACCEPT (and SK optionally) if it receives V'_U and $V_U = V'_U$. Otherwise, outputs REJECT and terminates the protocol.

NOTE 1 If the preparation phases for both X and C_j are executed every time before the protocol execution phase, b, $B = g^b$, y, $Y = g^y$, Y^b, B^y, X^b, B^x, t and t_0 can be omitted.

NOTE 2 For the same value of X, the above protocol execution phases can be executed at most $(\log_2 q)^2$ times regardless of the type of q. However, if $q + 1$ and $q - 1$ consist of small primes whose product is shorter than $(\log_2 q)^2$ bits and the remaining factors are large primes (e.g. see Reference [5]), there is no limit on the number of executions of the protocol execution phase for the same value of X.

6.2.4 User revocation

The SKI mechanism supports two types of user revocation: 1) exclude the user only from some user groups while leaving the user in the other user groups and in the password file, *PWF*, 2) exclude the user from the password file, *PWF*. In the former case, execute only step b) below and in the latter case execute both a) and b).

a) Remove the user's entry from the password file *PWF*.

b) Execute "the preparation phase for C_j" for the updated user groups that exclude the revoked user.

6.3 YZ mechanism

6.3.1 Setup

This operation is the same as that given in 6.2.1.

6.3.2 User registration

This operation is the same as that given in 6.2.2.

6.3.3 Anonymous authentication

In this operation, the user authenticates the server and the server authenticates a user of a pre-specified user group U, while preserving the anonymity of the user.

The steps of the anonymous authentication operation are as follows.

— **Step 1.** Upon request by a user U_i, the server S initiates the protocol by performing the following steps.

a) Selects a random number r_S in Z_q*.

b) Computes the values A_j for the n users $U_j(1 \le j \le n)$ in U as $A_j = pvd_{U_j}^{\ r_S}$.

c) Sends $<I_S, \{A_j\}_{1 \le j \le n}>$ to U_i.

— **Step 2.** After receiving a message of the form $< I_S, \{A_j\}_{1 \le j \le n} >$ from the server S, user U_i performs the following.

a) Checks that all the values $\{A_j\}_{1 \le j \le n}$ are different from each other and all the $\{Ocheck(A_j)\}_{1 \le j \le n}$ are valid. If not, aborts and outputs REJECT.

b) Picks out the i-th entry A_i from $\{A_j\}_{1 \le j \le n}$, which corresponds to the user U_i.

c) Selects two random values r_C, x from Z_q*, computes $X = g^x$, $T = A_i^{\ r_c}$ and $X'' = T*X$ and $B = pvd_{U_j}^{\ r_c}$.

d) Sends $<X'', B>$ to the server.

— **Step 3.** Upon receipt of the message $< X'', B>$, the server S proceeds with the following steps.

a) If any of $Ocheck(X'')$ and $Ocheck(B)$ is invalid, aborts and outputs REJECT.

b) Computes $T' = B^{r_S}$ with random value r_S and recovers $X' = X''/ T'$.

c) Selects a random value y in Z_q* and computes $Y = g^y$ and $K' = (X')^y$

d) If $Ocheck(K')$ is invalid, aborts and outputs REJECT.

e) Sets $Trans = I_S||\{\text{GE2OSP}(A_j)\}_{1 \le j \le n}||\text{GE2OSP}(X'')||\text{GE2OSP}(B)||\text{GE2OSP}(Y)$ and a MAC key MK = $H(GE2OSP(K'))$.

f) Generates an authenticator $V_S = MAC_{\lambda_v}$ (MK,1||$Trans$||GE2OSP(T')).

g) Sends $<Y, V_S>$ to the user.

— **Step 4.** When the message $<Y, V_S>$ is received by the user, it performs the following.

a) If either $Ocheck(Y)$ or V_S is invalid, aborts and outputs REJECT.

b) Computes $K = Y^x$.

c) Sets $Trans = I_S||\{\text{GE2OSP}(A_j)\}_{1 \le j \le n}||\text{GE2OSP}(X'')||\text{GE2OSP}(B)||\text{GE2OSP}(Y)$ and a MAC key MK' = H(GE2OSP(K)).

d) Checks whether $V_S = MAC_{\lambda_v}$ (MK′,1||$Trans$||GE2OSP(T)) holds. If it does not hold, aborts and outputs REJECT.

e) Otherwise, outputs ACCEPT (and a session key SK = MAC_{λ_v} (MK′,0||$Trans$||GE2OSP(T) optionally)).

f) Generates an authenticator $V_U = MAC_{\lambda_v}$ (MK′,2||$Trans$|| GE2OSP(T')) and sends it to the server.

— **Step 5.** When the message $<V_U>$ is received by the server, it performs the following.

a) Checks whether $V_U = MAC_{\lambda_v}$ (MK,2||*Trans*||GE2OSP(T)) holds. If it does not hold, aborts and outputs REJECT.

b) Otherwise, outputs ACCEPT (and a session key SK = MAC_{λ_k} (MK,0||*Trans*||GE2OSP(T)) optionally).

6.3.4 User revocation

A user U_i can be revoked by the server, S, as follows.

a) Inputs a distinguishing identifier, I_{U_i}, of the user who belongs to a group of users, U_j ($1 \leq j \leq n$).

b) Inputs the password file, *PWF*, that is kept secret by S.

c) Finds the entry $< I_{U_i}, pvd_{U_i} >$ in *PWF*, delete it, i.e. $PWF_{new} = PWF_{old} - \{< I_{U_i}, pvd_{U_i} >\}$.

7 Storage-extra PAEA mechanism

7.1 General

In Clause 7, one storage-extra PAEA mechanism, the YZW mechanism, is specified. This YZW mechanism is based on Reference [7].

7.2 YZW mechanism

7.2.1 General

This mechanism is specified to use the following primitives.

Taking a security parameter, *secparam*, the mechanism requires the following algorithms:

a) Homomorphic encryption

The mechanism specified in Clause 7 needs multiplicative homomorphic encryption where $HE_{pk}(\cdot)$ and $HD_{sk}(\cdot)$ denote the encryption function under a public key, *pk*, and the decryption function under a private key, *sk*, respectively. A multiplicative homomorphic encryption satisfies $HE_{pk}(m_1) \cdot HE_{pk}(m_2) = HE_{pk}(m_1 \cdot m_2)$, where $\cdot$ represents multiplication operation. The ElGamal encryption suffices to instantiate the needed multiplicative homomorphic encryption. A specification of the ElGamal encryption is given below.

Let the public/private key pair ($pk = g'^z$, $sk = z$), where $g' \in Z_q$, and $z \in Z_{q'}{}^*$ with $q = 2q' + 1$ and q, q' being prime numbers.

The instantiation of $HE_{pk}(m)$, encryption of $m \in Z_q$ proceeds as follows.

1) Select a random number $r \in Z_{q'}{}^*$, and compute $c_1 = g'^r \pmod q$, $c_2 = m \cdot pk^r = m \cdot g'^{zr} \pmod q$.

2) Set the ciphertext $C = (c_1, c_2)$.

Then, the instantiation of $HD_{sk}(C)$, decryption of $C = (c_1, c_2)$ proceeds as follows:

3) $m = c_2 / c_1{}^z \pmod q$

Furthermore, suppose two ciphertexts $C_1 = (g'^{r_1}, m_1 \cdot pk'^{r_1})$ and $C_2 = (g'^{r_2}, m_2 \cdot pk'^{r_2})$, then a multiplication of C_1 and C_2 is $C' = C_1 \cdot C_2 = (g'^{r_1 + r_2}, m_1 \cdot m_2 \cdot pk'^{r_1 + r_2})$. Obviously, C' is a valid ciphertext of $m_1 \cdot m_2$.

b) MAC algorithm defined in ISO/IEC 9797-2

— $MAC_{\lambda_v}: \{0,1\}^{\ell} \times \{0,1\}^* \rightarrow \{0,1\}^{\lambda_v}$.

c) Stream cipher algorithm defined in ISO/IEC 18033-4

— $ENC_{pw}(\cdot)$ and $DEC_{pw}(\cdot)$ denote the encryption function and the decryption function, respectively, under a key derived from a password, *pw*, (e.g. the key is derived by applying *pw* to a collision-resistant hash-function with output of an appropriate length).

d) Collision-resistant hash-function defined in ISO/IEC 10118-3

— $H_\lambda: \{0,1\}^* \rightarrow \{0,1\}^\lambda$.

7.2.2 Setup

The setup operation creates public system parameters and a secret server key, consisting of the following steps.

a) Establishes the set of base groups G_1, G_2, G_3 and a bilinear pairing $\mathbf{e}: G_1 \times G_2 \rightarrow G_3$, where all G_1, G_2, G_3 are cyclic groups of a prime order, q. Note that $|q|$ shall be at least twice the value of *secparam*. MNT elliptic curves with pairings suffice to instantiate e.

b) Selects random generators $a, b, d, g, g_0, g_1 \in G_1$ and a random generator $h \in G_2$. Selects a random number $x \in Z_q^*$, and compute $W = h^x$.

c) Generates a public/private key pair (pk, sk) for a multiplicative homomorphic encryption scheme, with $HE_{pk}(\cdot)$ and $HD_{sk}(\cdot)$ denoting encryption/decryption, respectively. Note that the multiplicative homomorphic encryption scheme shall be generated such that its plaintext space is $\{0,1\}^\ell$, where $\ell \geq |q|$ is a sufficiently large number with respect to *secparam* and $Z_q \subset \{0,1\}^\ell$.

d) Selects a collision-resistant hash-function $H_\lambda: \{0,1\}^* \rightarrow \{0,1\}^\lambda$ and a MAC algorithm $MAC_{\lambda_v}: \{0,1\}^\ell \times \{0,1\}^* \rightarrow \{0,1\}^{\lambda_v}$. Note that λ and λ_v shall be at least twice the value of *secparam*.

e) Sets the public system parameters as $params = \langle q, G_1, G_2, G_3, \mathbf{e}, a, b, d, g, g_0, g_1, h, W, pk, H_\lambda, MAC_{\lambda_v} \rangle$, and the secret server key as $servK = \langle x, sk \rangle$.

NOTE ISO/IEC 15946-1 contains recommendations for the generation of the cyclic groups, group generators and pairing.

It is recommended that applications should establish a methodology for changing the system parameters on a regular basis. In such cases, all users need to register again to the server to renew their credentials. However, this is out of scope of this document.

7.2.3 User registration

In the user registration operation, a user registers to the server such that the latter issues a password wrapped credential to the former. To this end, the user submits a password, *pw*, and the server issues the credential using the server key. The algorithm proceeds as follows.

Input:

— the system parameters *params*;

— the server key $servK = \langle x, sk \rangle$;

— user's distinguishing identifier, I_U, and a user password, *pw*.

Output:

— a password-wrapped credential, $cred_{pw}$.

Operation: Use the following steps to compute $cred_{pw}$.

a) Select random numbers k, s and m' where $k, s \in Z_q{}^*$, and $m' \in [0, 2^l - 1]$ where l is the bit length of the key stream of $ENC_{pw}()$ and $2^l - 1$ is by far larger than q, and then computes $m = m'$ (mod q) and $M = (a^m \cdot b^s \cdot d)^{1/(k+x)}$.

b) Encrypts m' with stream cipher under a key derived from pw, denoted as $ENC_{pw}(m')$.

c) Encrypts s with homomorphic encryption under pk, denoted as $HE_{pk}(s)$.

d) Sets $cred_{pw} = \langle M, ENC_{pw}(m'), k, HE_{pk}(s)\rangle$.

e) The user obtains a trusted copy of the public system parameters chosen in 7.2.1.

NOTE 1 In this algorithm, it is implicit that mechanisms should be in place to ensure that the server can verify the authenticity of the registering user, e.g. the user presents the user's IC to the server. Details of the verification is essential, but out of the scope of 7.2.

NOTE 2 It is obvious that, given $\langle M, m, k, s\rangle$, the user himself can generate the password-wrapped credential. Therefore, an alternative to the above procedure is that the user registers to the server by inputting the user's distinguishing identifier, I_U, and upon obtaining $\langle M, m, k, s\rangle$ from the server in a secure way, he generates $cred_{pw}$ by himself.

7.2.4 Anonymous authentication

The anonymous authentication operation allows both the server and a user to authenticate each other and (optionally) establish a common session key between them.

Input:

— the system parameters $params$;

— the server key $servK = \langle x, sk\rangle$;

— a password-wrapped credential $cred_{pw} = \langle elem_1, elem_2, elem_3, elem_4\rangle = \langle M, ENC_{pw}(m'), k, HE_{pk}(s)\rangle$ and a user password, pw.

Output:

— ACCEPT (and SK optionally) or REJECT.

The anonymous authentication operation consists of the following steps.

— **Step 1.** The user initiates the protocol by performing the following.

a) Computes $m' = DEC_{pw}(elem_2)$ and $m = m'$ (mod q), where $DEC_{pw}(\cdot)$ is the reverse function of $ENC_{pw}(\cdot)$.

b) Picks a random number $r \in Z_q{}^*$ and computes $s^* = HE_{pk}(r) \cdot elem_4 = HE_{pk}(r \cdot s)$.

c) Picks a random number $x_1 \in Z_q{}^*$ and computes $X = g^{x_1}$.

d) Picks a random number $N_U \in \{0, 1\}^\ell$ and computes $N_U{}^* = HE_{pk}(N_U)$.

e) Computes $\langle T_1, T_2, R_1, R_2\rangle$ as follows:

1) Computes $\gamma = r^{-1}$ (mod q)

2) Computes $B_e = (\mathbf{e}(M, W \cdot h^k)/(\mathbf{e}(a, h)^m \cdot \mathbf{e}(d, h)))$.

3) Selects a random number $\alpha, \mu \in Z_q{}^*$.

4) Computes $T_1 = M \cdot g_0{}^\alpha$ and $T_2 = g_1{}^\alpha . g_0{}^\mu$