INTERNATIONAL STANDARD ISO/IEC 9594-8:2005
TECHNICAL CORRIGENDUM 2

Published 2009-11-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks

TECHNICAL CORRIGENDUM 2

Technologies de l'information — Interconnexion de systèmes ouverts (OSI) — L'annuaire: Cadre général des certificats de clé publique et d'attribut

RECTIFICATIF TECHNIQUE 2

Technical Corrigendum 2 to ISO/IEC 9594-8:2005 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*, in collaboration with ITU-T. The identical text is published as ITU-T Rec. X.509 (2005)/Cor.2 (11/2008).

ICS 35.100.70 **Ref. No. ISO/IEC 9594-8:2005/Cor.2:2009(E)**

Published in Switzerland

(Blank page)

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks

Technical Corrigendum 2

1) Correction of the defects reported in defect report 326

a) *Add a new paragraph to the end of 18.1.1*

Annex J provides a suggested algorithm to be used for protected passwords.

b) *Add Annex J and renumber subsequent annexes:*

Annex J

Use of Protected Passwords for Bind operations

(This annex does not form an integral part of this Recommendation | International Standard)

The protected component of `SimpleCredentials` specifies an `OCTET STRING` to be hashed. This annex provides information about how this octet-string may be constructed. It also proposes some suggested associated procedures.

In its simple form, the octet-string is constructed as the DER encoding of the following:

```
SEQUENCE {
      name          DistinguishedName,
      time1         GeneralizedTime,
      random1       BIT STRING,
      password      OCTET STRING }
```

The `name` component is the distinguished name of the sender and the `password` component is the password of the sender.

The sender generates the two other values as follows:

a) The `time1` value should specify the time after which the authentication should fail. This time should be "closely" after the current time.

b) The `random1` value is a new random number generated for each authentication attempt. The value should be sufficiently large to prevent the same number to be generated frequently.

The same pair of `time1` and `random1` should never be used more than once.

The same value of `name`, `time1` and `random1` shall be supplied in the `SimpleCredentials` data type of the Bind.

NOTE 1 – The hashing algorithm is also transferred.

The receiver of a Bind request/result will perform the authentication as follows:

a) If the value in `time1,` as supplied in the `SimpleCredentials,` is less than the current time seen by the recipient, the authentication already fails here. Also, the time value should be different from recently received time values.

b) If the value in the `random1,` as supplied in the `SimpleCredentials,` is equal to a value received in a recent Bind request/response, the authentication also fails.

c) If `time1` and `random1` appear to be valid, the `name`, `time1` and `random1` included in the Bind request/result, together with the local copy of the password, are used to generate a copy of the message digest using the algorithm indicated.

d) If the generated message digest is equal to the message digest received in the Bind request/result, the authentication is positive, otherwise it fails.

The above procedure allows the password to be protected during transfer and it prevents replay of the transmission sequence. If the attempted reply is done early, the random number will cause the authentication to fail. If the reply is attempted sometime later, the random number may be accepted, but the authentication will fail due to the time value.

The scheme above may be extended by using the following sequence.

```
SEQUENCE {
        f1          OCTET STRING,     -- hashed octet string from above
        time2       GeneralizedTime,
        random2     BIT STRING }
```

The DER encoding of this data type is then used as the octet-string in the `SimpleCredentials`.

In this case, also the `time2` and `random2` have to be included in `SimpleCredentials`.

The hashing algorithm used for producing the `f1` component shall be the same as used for the hashing, as indicated within the `HASH` data type within `SimpleCredentials`.

NOTE 2 – This Directory Specification does not give any recommendation as to how values for `time2` and `random2` are selected.

2) Correction of the defects reported in defect report 330

a) *In clause 7, in the paragraph starting with "The* **extensions** *field allows..", replace the existing text:*

"If the criticality flag is **TRUE**, unrecognized extensions shall cause the structure to be considered invalid, i.e., in a certificate, an unrecognized critical extension would cause validation of a signature using that certificate to fail. When a certificate-using implementation recognizes and is able to process an extension, then the certificate-using implementation shall process the extension regardless of the value of the criticality flag. Note that any extension that is flagged non-critical will cause inconsistent behaviour between certificate-using systems that will process the extension and certificate-using systems that do not recognize the extension and will ignore it.

If unknown elements appear within the extension, and the extension is not marked critical, those unknown elements shall be ignored according to the rules of extensibility documented in 12.2.2 in ITU-T Rec. X.519 | ISO/IEC 9594-5."

With the following text:

"If the criticality flag is **TRUE**, unrecognized extensions shall cause the structure to be considered invalid, i.e., in a certificate, an unrecognized critical extension would cause validation of a signature using that certificate to fail. When a certificate-using implementation recognizes and is able to fully process an extension, then the certificate-using implementation shall process the extension regardless of the value of the criticality flag. When a certificate-using implementation recognizes and is able to partially process an extension for which the criticality flag is **TRUE**, then its behaviour in the presence of unrecognized elements is extension specific and may be documented in each extension. However, the default behaviour, when not specified specifically for an extension, is to treat the entire extension as unrecognised. If unrecognized elements appear within the extension, and the extension is not marked critical, those unrecognized elements shall be ignored according to the rules of extensibility documented in 12.2.2 in ITU-T Rec. X.519 | ISO/IEC 9594-5.

Note that any extension that is flagged non-critical will cause inconsistent behaviour between certificate-using systems that will process the extension and certificate-using systems that do not recognize the extension and will ignore it. The same may be true for extensions that are flagged critical, between certificate-using systems that can fully process the extension and those that can partially process the extension, depending upon the extension."

b) *In clause 7 replace the following paragraph:*

"A validation engine has two possible actions to take with respect to an extension:

i) it can ignore the extension and accept the certificate (all other things being equal);

ii) it can process the extension and accept or reject the certificate depending on the content of the extension and the conditions under which processing is occurring (e.g., the current values of the path processing variables)."