Telecommunications and Information Exchange Between Systems

ISO/IEC JTC 1/SC 6

Document Number:	N14274
Date:	2010-04-28
Replaces:	
Document Type:	Text for PDAM Ballot
Document Title:	Text for PDAM ballot, Amendment 2 to ISO/IEC 9594-All parts, Information technology – Open Systems Interconnection – The Directory
Document Source:	SC 6/WG 8 Geneva meeting
Project Number:	
Document Status:	SC 6 P-members are requested to ballot on this PDAM text through the e-balloting system (www.iso.org/jtc1/sc6) no later than 2010-07-28.
Action ID:	LB
Due Date:	2010-07-28
No. of Pages:	42

ISO/IEC JTC1/SC6 Secretariat Ms. Jooran Lee, KSA (on behalf of KATS)

Korea Technology Center #701-7 Yeoksam-dong, Gangnam-gu, Seoul, 135-513, Republic of Korea ;

Telephone: +82 2 6009 4808 ; Facsimile: +82 2 6009 4819 ; Email : jooran@kisi.or.kr

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Open Systems Interconnection – The Directory

Amendment 2

Password policy

Summary

Password policy is a set of rules that controls how passwords are used and administered in the Directory. It improves the security of the Directory and makes it difficult for password cracking programs to break into the Directory. These rules ensure that users change their passwords periodically, that passwords meet quality requirements that re-use of old passwords is restricted, and that users are locked out after a certain number of failed attempts.

CONTENTS

CONTENTS 1

ISO/IEC 9594-1: 2008, Information Technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services 3

1) Subclause A.3.8 3

ISO/IEC 9594-2: 2008, Information Technology – Open Systems Interconnection – The Directory: Models 5

1) Subclause 11.5.2 5

2) Subclause 14.3 5

3) Subclause 14.4.3 5

4) Subclause 14.9 5

5) Annex B 6

ISO/IEC 9594-3: 2008, Information Technology – Open Systems Interconnection – The Directory: Abstract service definition 9

1) Clause 8 9

2) Subclause 8.1.1 and Annex A 9

3) Subclause 8.1.2 9

4) Subclause 8.1.3 10

5) Subclause 8.1.4 10

6) New subclause 11.5 10

7) Subclause 12.7 11

8) Subclause 12.9 12

9) Annex A 13

ISO/IEC 9594-4: 2008, Information Technology – Open Systems Interconnection – The Directory: Procedures for distributed operation 15

1) Annex A 15

ISO/IEC 9594-5: 2008, Information Technology – Open Systems Interconnection – The Directory: Protocol specifications 17

1) Subclause 6.4.1 and Annex A 17

2) Subclause 12.2.3 17

3) Subclause 13.1.1 17

ISO/IEC 9594-6: 2008, Information Technology – Open Systems Interconnection – The Directory: Selected attribute types 19

1) Subclause 2.2 19

2) Subclause 6.2.12 19

3) Subclause 8.9 19

4) Annex A 20

ISO/IEC 9594-7: 2008, Information Technology – Open Systems Interconnection – The Directory: Selected object classes 21

1) Subclause 6.5 and Annex A 21

2) Annex A 21

ISO/IEC 9594-8: 2008, Information Technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks 23

1) Subclause 3.3 23

2) Subclause 18.1.3 23

3) Subclause 18.1.4 25

4) Subclause 18.1.5 26

5) Subclause 18.1.6 27

6) Subclause 18.1.8 31

7) Subclause 18.1.10 32

8) SubClause 18.2.1 32

9) SubClause 18.2.2.1 32

10) Annex A.1 32

11) Annex F 36

12) Annex L 36

ISO/IEC 9594-9: 2008, Information Technology – Open Systems Interconnection – The Directory: Replication 39

1) Subclause 9.2.2 39

2) Subclause 9.2.4 39

3) Subclause 9.2.4.5 39

ISO/IEC 9594-10: 2008, Information Technology – Open Systems Interconnection – The Directory: Use of systems management for administration of the Directory 41

ISO/IEC 9594-1: 2008, Information Technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services

1) Subclause A.3.8

Add at the end of the second paragraph:

A password policy can be used for administration of passwords to improve the security of the Directory and make it more difficult for password cracking programs to break into the Directory. A password policy defines the rules to ensure that users change their passwords periodically, that passwords meet quality requirements, that re-use of old passwords is restricted, and that users are locked out after a certain number of failed bind attempts or password comparison. A password can only be changed by the owner of the entry or by an administrator of the Directory (for example when a user has lost his password).

ISO/IEC 9594-2: 2008, Information Technology – Open Systems Interconnection – The Directory: Models

1) Subclause 11.5.2

Replace the first paragraph and the enumeration with:

In the same way that an Administrative Authority may operate in a specific role, entries in an administrative area may be considered in terms of a specific administrative function. When viewed in this context, an administrative area is termed a *specific administrative area*. There are six kinds of specific administrative area:

– subschema administrative areas;
– access control administrative areas;
– collective-attribute administrative areas;
– context default administrative areas;
– service administrative areas;
– password administrative areas.

2) Subclause 14.3

Add the following new value to the **`administrativeRole`** *attribute:*

`id-ar-pwdAdminSpecificArea`

3) Subclause 14.4.3

Add at the end of 14.4.3

The **`pwdAdminSubentryList`** operational attribute identifies the password administration subentry, if any, that affect the entry. It is available in every entry affected by any such subentry.

```
pwdAdminSubentryList  ATTRIBUTE  ::=  {
       WITH SYNTAX                  DistinguishedName
       EQUALITY MATCHING RULE       distinguishedNameMatch
       SINGLE VALUE                 TRUE
       NO USER MODIFICATION         TRUE
       USAGE                        directoryOperation
       ID                           id-oa-pwdAdminSubentryList }
```

4) Subclause 14.9

Add a new subclause after 14.8 and renumber subsequent subclauses:

14.9 System schema supporting password administration

If a subentry holds password policy information, then its **`objectClass`** attribute shall contain the value **`pwdAdminSubentry`**:

```
pwdAdminSubentry  OBJECT-CLASS  ::=  {
       KIND              auxiliary
       MUST CONTAIN      { pwdAttribute }
       ID                id-sc-pwdAdminSubentry }
```

A subentry of the object class **`pwdAdminSubentry`** may contain the following attributes **`pwdModifyEntryAllowed`**, **`pwdChangeAllowed`**, **`pwdMaxAge`**, **`pwdExpiryAge`**, , **`pwdMinLength`**, **`pwdVocabulary`**, **`pwdAlphabet`**, **`pwdExpiryWarning`**, **`pwdGraces`**, **`pwdFailureDuration`**, **`pwdLockoutDuration`**, **`pwdMaxFailures`**, **`pwdMaxTimeInHistory`**, **`pwdMinTimeInHistory`**, **`pwdHistorySlots`**, **`pwdRecentlyExpiredDuration`**, **`pwdEncAlg`**.

`pwdAttribute` contains the password attribute that is being controlled by the password administration subentry. Every password attribute can only have at most one password policy that applies to it. If two or more subtree specifications overlap, then only one of them can apply to each entry in the overlapping space as controlled by the **`pwdAdminSubentryList`** attribute in each entry.

```
pwdAttribute      ATTRIBUTE ::= {
       WITH SYNTAX            ATTRIBUTE.&id
       EQUALITY MATCHING RULE objectIdentifierMatch
       SINGLE VALUE           TRUE
       ID                     id-at-pwdAttribute }
```

One password attribute is currently defined, **`userPwd`** which contains a password stored in clear text or encrypted. This attribute shall have a matching rule for comparison of a proposed password value with the password value stored in the Directory. For each defined password attribute, two attributes for password history and recently expired password respectively are needed as well as a matching rule for comparison of a presented password value with a password stored in the history. The attribute **`userPwdHistory`** and the matching rule **`userPwdHistoryMatch`** are defined for the **`userPwd`** password attribute. The attribute **`userPwdRecentlyExpired`** and the matching rule **`userPwdHistoryMatch`** are defined for the **`userPwd`** using the **`UserPwd`** type.

If new password attributes using other syntaxes are needed, new attributes and new matching rules will also be defined. The following parameterized objects can be used for that.

14.9.1 Definition of an history attribute from the password attribute, the history matching rule and an object identifier

```
pwdHistory{ATTRIBUTE:passwordAttribute,MATCHING-RULE:historyMatch,
       OBJECT IDENTIFIER:id} ATTRIBUTE ::= {
       WITH SYNTAX            PwdHistory{passwordAttribute}
       EQUALITY MATCHING RULE historyMatch
       USAGE                  directoryOperation
       ID                     id}

PwdHistory{passwordAttribute} ::= SEQUENCE {
       time        GeneralizedTime,
       password    passwordAttribute.&Type}
```

14.9.2 Definition of a recently expired password attribute from the password attribute and an object identifier

```
pwdRecentlyExpired{ATTRIBUTE:passwordAttribute,OBJECT IDENTIFIER:id} ATTRIBUTE ::= {
       WITH SYNTAX            passwordAttribute.&Type
       EQUALITY MATCHING RULE passwordAttribute.&equality-match
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id}
```

14.9.3 Definition of a password history matching rule from the password attribute and an object identifier

```
pwdHistoryMatch{ATTRIBUTE:passwordAttribute,OBJECT IDENTIFIER:id} MATCHING-RULE ::= {
       SYNTAX      passwordAttribute.&Type
       ID          id}
```

5) Annex B

Add after the definition of ***`serviceAdminSubentryList`****:*

```
pwdAdminSubentryList  ATTRIBUTE  ::=  {
       WITH SYNTAX                 DistinguishedName
       EQUALITY MATCHING RULE      distinguishedNameMatch
       SINGLE VALUE                TRUE
       NO USER MODIFICATION        TRUE
       USAGE                       directoryOperation
       ID                          id-oa-pwdAdminSubentryList }
```

Add after the definition of ***`SearchRuleDescription`****:*

```
pwdAdminSubentry  OBJECT-CLASS  ::=  {
       KIND             auxiliary
       MUST CONTAIN     { pwdAttributeType }
       ID               id-sc-pwdAdminSubentry }

pwdAttribute      ATTRIBUTE ::= {
       WITH SYNTAX             ATTRIBUTE.&id
       EQUALITY MATCHING RULE  objectIdentifierMatch
       SINGLE VALUE            TRUE
       ID                      id-at-pwdAttribute }

pwdHistory{ATTRIBUTE:passwordAttribute,MATCHING-RULE:historyMatch,
                   OBJECT IDENTIFIER:id}
       ATTRIBUTE ::= {
       WITH SYNTAX             PwdHistory{passwordAttribute}
       EQUALITY MATCHING RULE  historyMatch
       USAGE                   directoryOperation
       ID                      id}

PwdHistory{ATTRIBUTE:passwordAttribute} ::= SEQUENCE {
       time        GeneralizedTime,
       password    passwordAttribute.&Type}

pwdRecentlyExpired{ATTRIBUTE:passwordAttribute,OBJECT IDENTIFIER:id} ATTRIBUTE ::= {
       WITH SYNTAX             passwordAttribute.&Type
       EQUALITY MATCHING RULE  passwordAttribute.&equality-match
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id}

pwdHistoryMatch{ATTRIBUTE;passwordAttribute,OBJECT IDENTIFIER:id} MATCHING-RULE ::= {
       SYNTAX      passwordAttribute.&Type
       ID          id}
```

Add at the end of the list of attributes

```
id-at-pwdAttribute            OBJECT IDENTIFIER ::=   {id-at 84}
```

Add at the end of the list of operational attributes:

```
id-oa-pwdAdminSubentryList    OBJECT IDENTIFIER ::=   {id-oa 21}
```

Add at the end of the list of subentry classes:

```
id-sc-pwdAdminSubentry        OBJECT IDENTIFIER ::=   {id-sc 5}
```

Add to the end of the list of administrative roles:

```
id-ar-pwdAdminSpecificArea    OBJECT IDENTIFIER::=    {id-ar 9}
```

ISO/IEC 9594-3: 2008, Information Technology – Open Systems Interconnection – The Directory: Abstract service definition

1) Clause 8

Replace the title of clause 8 with:

8 Bind, Unbind and Change Password and Administer Password operations

2) Subclause 8.1.1 and Annex A

Replace the ASN.1 definition of **`SimpleCredentials`** *with:*

```
SimpleCredentials ::= SEQUENCE {
       name [0] DistinguishedName,
       validity [1] SET {
              time1 [0] CHOICE {
                    utc UTCTime,
                    gt GeneralizedTime } OPTIONAL,
              time2 [1] CHOICE {
                    utc UTCTime,
                    gt GeneralizedTime } OPTIONAL,
              random1 [2] BIT STRING OPTIONAL,
              random2 [3] BIT STRING OPTIONAL } OPTIONAL,
       password [2] CHOICE {
              unprotected OCTET STRING,
              protected HASH {OCTET STRING},
              userPwd    [0] UserPwd } OPTIONAL}
```

Replace the ASN.1 definition of **`DirectoryBindResult`** *with:*

```
DirectoryBindResult ::= SET {
       credentials       [0]   Credentials OPTIONAL,
       versions          [1]   Versions DEFAULT {v1},
       pwdResponseValue  [2]   PwdResponseValue OPTIONAL }
```

Insert after definition of **`Versions`**, *the following definitions:*

```
PwdResponseValue ::= SEQUENCE {
       warning [0] CHOICE {
              timeLeft          [0]   INTEGER (0..MAX),
              graceRemaining    [1]   INTEGER (0..MAX)} OPTIONAL,
       error [1] ENUMERATED {
              passwordExpired (0),
              changeAfterReset(1) } OPTIONAL }
```

3) Subclause 8.1.2

Replace the second paragraph of 8.1.2 with:

If **`simple`** is used, it consists of a **`name`** (always the distinguished name of an object), an optional **`validity`**, and an optional **`password`**. This provides a limited degree of security. The **`password`** may be **`unprotected`**, or it may be **`protected`** (either Protected1 or Protected2) as described in 18.1 of ITU-T Rec. X.509 | ISO/IEC 9594-8 or it may be the **`userPwd`** attribute. The **`validity`** supplies **`time1`**, **`time2`**, **`random1`** and **`random2`** arguments, which derive their meaning by bilateral agreement, and which may be used to detect replay. In some instances a protected password may be checked by an object which knows the password only after locally regenerating the protection to its own copy of the password and comparing the result with the value in the bind argument (**`password`**). In other instances, a direct comparison may be possible. A possible approach for protected password may be found in an informative annex of ITU-T Rec. X.509 | ISO/IEC 9594-8. If the **`userPwd`** attribute is used, the password may be transmitted in the clear or encrypted and the matching rule is defined in 18.1.8 if ITU-T Rec. X.509 | ISO/IEC 9594-8.

4) Subclause 8.1.3

Insert at end the following text:

The following applies independently whether the DSA holds the responder's master entry or a replicated entry.

a) if the **`warning.timeLeft`** component is present and different from zero, the **`error`** component shall be absent;

b) if the **`warning.graceRemaining`** component is present, the **`error.passwordExpired`** may be set.

The following applies when the DSA holds the master entry for the requestor:

a) if **`warning`** is present with either the **`timeLeft`** set to zero or **`graceRemaining`** set to zero and **`error.passwordExpired`** set, only a change-password operation is accepted;

b) if **`error.changeAfterReset`** is set, **`warning`** shall not be present. Only a change-password operation is accepted.

5) Subclause 8.1.4

*Replace the text after the sentence "A **`securityError`** or **`serviceError`** shall be supplied as follows:"*

```
–     securityError    inappropriateAuthentication
                       invalidCredentials
                       blockedCredentials
                       passwordPolicyRequired
                       passwordExpired
                       inappropriateAlgorithms
–     serviceError     unavailable
                       saslBindInProgress
```

6) New subclause 11.5

Add new subclause 11.5

11.5 Change Password

This operation is intended to be used by Directory users to change their own passwords.

11.5.1 Change Password syntax

A Directory Change Password operation is used by a user to change a password to prevent password expiration or after password reset by an administrator. The password may be changed at any time during an application-association. The user is allowed as many attempts as specified in the **`pwdMaxCompareFailure`** attribute. When this limit is reached, the DSA shall unbind the application-association and, if the **`pwdCompareLockout`** attribute is **`TRUE`**, lock the account for **`pwdCompareLockoutDuration`**.

```
changePassword OPERATION ::= {
       ARGUMENT          ChangePasswordArgument
       RESULT            ChangePasswordResult
       ERRORS            { securityError | updateError }
       CODE              id-opcode-changePassword }

ChangePasswordArgument ::= OPTIONALLY-PROTECTED-SEQ {
       SEQUENCE {
             object      [0]   DistinguishedName,
             oldPwd      [1]   userPwd,
             newPwd      [2]   userPwd }}

ChangePasswordResult ::= CHOICE {
       null NULL,
       information OPTIONALLY-PROTECTED-SEQ {
             SEQUENCE {
                  COMPONENTS OF CommonResultsSeq}}}
```

11.5.2 Change Password arguments

The current password (**`oldPwd`** component) and the new password (**`newPwd`** component) have to be supplied in a Change Password operation. The **`oldPwd`** and **`newPwd`** components shall content a clear or encrypted password.

11.5.3 Change Password results

If the password is changed successfully, the operation returns no information and normal communication may continue.

11.5.4 Change Password errors

Should the request fail, a **`securityError`** or **`updateError`** shall be supplied as follows:

```
–     securityError    inappropriateAlgorithms

–     updateError      pwdInsufficientQuality
                       pwdInHistory
                       pwdHistoryFull
```

The circumstances under which other errors shall be reported are defined in clause 12.

11.6 Administer Password

This operation is intended to be used by Directory Administrators to change users' passwords. If two free slots are not available in the **`userPwdHistory`** attribute, this operation will free two slots before proceeding. At the end of the successful operation, there will be one free slot for the user to change the password which has been set by the Administrator.

11.6.1 Administer Password syntax

Administer password operation is used by an administrator to change a user's password.

```
administerPassword OPERATION ::= {
       ARGUMENT          AdministerPasswordArgument
       RESULT            AdministerPasswordResult
       ERRORS            { securityError | updateError }
       CODE              id-opcode-administerPassword }

AdministerPasswordArgument ::= OPTIONALLY-PROTECTED-SEQ {
       SEQUENCE {
             Object      [0]   DistinguishedName,
             newPwd      [1]   userPwd }}

AdministratorPasswordResult ::= CHOICE
       null NULL,
       information OPTIONALLY-PROTECTED-SEQ {
             SEQUENCE {
```

COMPONENTS OF CommonResultsSeq}}}11.6.2 Administer Password arguments

The new password (**`newPwd`** component) have to be supplied in Administer password operation. The newPwd component shall contain a clear or encrypted password.

11.6.3 Administer Password results

If the password is changed successfully, the operation returns no information and normal communication may continue.

11.6.4 Administer Password errors

Should the request fail, a **`securityError`** or **`updateError`** shall be supplied as follows:

```
–     securityError    inappropriateAlgorithms

–     updateError      pwdInsufficientQuality
                       pwdInHistory
```

The circumstances under which other errors shall be reported are defined in clause 12.

7) Subclause 12.7

Replace the definition of SecurityError with:

```
securityError  ERROR  ::=  {
       PARAMETER   OPTIONALLY-PROTECTED { SET {
                   problem          [0]   SecurityProblem,
                   spkmInfo         [1]   SPKM-ERROR OPTIONAL,
                   encPwdInfo       [2]   EncPwdInfo OPTIONAL,
                                          COMPONENTS OF CommonResults } }
```

```
        CODE   id-errcode-securityError }

EncPwdInfo ::= SEQUENCE {
        algorithms        SEQUENCE OF AlgorithmIdentifier{{SupportedAlgorithms}) OPTIONAL,
        pwdQualityRule    SEQUENCE OF AttributeTypeAndValue OPTIONAL }
```

Replace the definition of **`SecurityProblem`** *with:*

```
SecurityProblem  ::=  INTEGER {
        inappropriateAuthentication        (1),
        invalidCredentials                 (2),
        insufficientAccessRights           (3),
        invalidSignature                   (4),
        protectionRequired                 (5),
        noInformation                      (6),
        blockedCredentials                 (7),
        -- invalidQOPMatch                 (8), obsolete
        spkmError                          (9),
        unsupportedAuthenticationMethod    (10),
        passwordExpired                    (11),
        inappropriateAlgorithms            (12)}
```

Insert after item i) the new following items:

j) **`passwordExpired`** – The requestor cannot log onto the DSA because the password has expired. The password has to be reset by an administrator.

k) **`inappropriateAlgorithms`** – The algorithms used to encrypt the password are not compatible with the algorithms stored in the DSA for the entry. The **`algorithms`** parameter contains the list of algorithms supported by the DSA.

NOTE 1 - For bind operation or compare operation, one or two algorithms can be specified to check the proposed password with the encrypted password and the possible recently expired encrypted password. For change password operation the algorithm used by the current password and all the algorithms used by the password present in the history shall be returned.

l) **`pwdInsufficientQuality`** – The password quality was insufficient and the **`pwdQualityRule`** parameter specifies the quality attributes required by the DSA.

NOTE 2 - When the password is not transmitted in clear text to the DSA, the quality rule cannot be checked by the DSA but only by the DUA.

8) Subclause 12.9

Replace the definition of **`UpdateProblem`** *with:*

```
UpdateProblem  ::=  INTEGER {
        namingViolation                    (1),
        objectClassViolation               (2),
        notAllowedOnNonLeaf                (3),
        notAllowedOnRDN                    (4),
        entryAlreadyExists                 (5),
        affectsMultipleDSAs                (6),
        objectClassModificationProhibited  (7),
        noSuchSuperior                     (8),
        notAncestor                        (9),
        parentNotAncestor                  (10),
        hierarchyRuleViolation             (11),
        familyRuleViolation                (12),
        insufficientPasswordQuality        (13),
        passwordInHistory                  (14),
        noPasswordSlot                     (15) }
```

Insert after item l) the new following items:

m) **`pwdInsufficientQuality`** – The new password does not satisfy the quality rules (no trivial passwords, mixture of characters, too short, etc) imposed by the Directory.

n) **`pwdInHistory`** – The new password has been found in the history kept by the Directory.

o) **`pwdHistoryFull`** – There are no free slots left in the password history.

9) Annex A

In the IMPORTS clause replace:

```
-- from ITU-T Rec. X.519 | ISO/IEC 9594-5
       Code, ERROR, id-errcode-abandoned, id-errcode-abandonFailed,
       id-errcode-attributeError, id-errcode-nameError, id-errcode-referral,
       id-errcode-securityError, id-errcode-serviceError, id-errcode-updateError,
       id-opcode-abandon, id-opcode-addEntry, id-opcode-compare, id-opcode-list,
       id-opcode-modifyDN, id-opcode-modifyEntry, id-opcode-read,
       id-opcode-removeEntry, id-opcode-search, InvokeId, OPERATION
             FROM CommonProtocolSpecification commonProtocolSpecification
```

with:

```
-- from ITU-T Rec. X.519 | ISO/IEC 9594-5
       Code, ERROR, id-errcode-abandoned, id-errcode-abandonFailed,
       id-errcode-attributeError,id-errcode-nameError, id-errcode-referral,
       id-errcode-securityError, id-errcode-serviceError,id-errcode-updateError,
       id-opcode-abandon, id-opcode-addEntry, id-opcode-compare, id-opcode-list,
       id-opcode-modifyDN, id-opcode-modifyEntry, id-opcode-read,
       id-opcode-removeEntry, id-opcode-search, id-opcode-changePassword,
       id-opcode-administerPassword, InvokeId,
       OPERATION
             FROM CommonProtocolSpecification commonProtocolSpecification
```

and:

```
-- from ITU-T Rec. X.509 | ISO/IEC 9594-8
       AlgorithmIdentifier{}, CertificationPath, ENCRYPTED {}, HASH {}, SIGNED {},
       SupportedAlgorithms,
             FROM AuthenticationFramework authenticationFramework
```

with:

```
-- from ITU-T Rec. X.509 | ISO/IEC 9594-8
       AlgorithmIdentifier{}, CertificationPath, ENCRYPTED {}, HASH {}, SIGNED {},
       SupportedAlgorithms, UserPwd
             FROM AuthenticationFramework authenticationFramework
```

Insert after -- Operations, arguments, and results –

```
changePassword OPERATION ::= {
       ARGUMENT          ChangePasswordArgument
       RESULT            ChangePasswordResult
       ERRORS            { securityError | updateError }
       CODE              id-opcode-changePassword }

ChangePasswordArgument ::= OPTIONALLY-PROTECTED-SEQ {
       SEQUENCE {
             object      [0]   DistinguishedName,
             oldPwd      [1]   UserPwd,
             newPwd      [2]   UserPwd }}

ChangePasswordResult ::= NULL

administerPassword OPERATION ::= {
       ARGUMENT          AdministerPasswordArgument
       RESULT            AdministerPasswordResult
       ERRORS            { securityError | updateError }
       CODE              id-opcode-administerPassword }

AdministerPasswordArgument ::= OPTIONALLY-PROTECTED-SEQ {
       SEQUENCE {
             Object      [0]   DistinguishedName,
             newPwd      [1]   userPwd }}

AdministratorPasswordResult ::= NULL
```

Replace the definition of ***SecurityError*** *with:*

```
securityError  ERROR  ::=  {
       PARAMETER   OPTIONALLY-PROTECTED { SET {
                   problem          [0]   SecurityProblem,
                   spkmInfo         [1]   SPKM-ERROR,
                   encPwdInfo       [2]   EncPwdInfo OPTIONAL,
                                          COMPONENTS OF CommonResults } }
       CODE   id-errcode-securityError }

EncPwdInfo ::= SEQUENCE {
       algorithms        SEQUENCE OF AlgorithmIdentifier{{SupportedAlgorithms}) OPTIONAL,
       pwdQualityRule    SEQUENCE OF AttributeTypeAndValue OPTIONAL }
```

Replace the definition of ***SecurityProblem*** *with:*

```
SecurityProblem  ::=  INTEGER {
       inappropriateAuthentication             (1),
       invalidCredentials                      (2),
       insufficientAccessRights                (3),
       invalidSignature                        (4),
       protectionRequired                      (5),
       noInformation                           (6),
       blockedCredentials                      (7),
       -- invalidQOPMatch                      (8), obsolete
       spkmError                               (9),
       unsupportedAuthenticationMethod         (10),
       passwordExpired                         (11),
       passwordModNotAllowed                   (12),
       inappropriateAlgorithms                 (13) }
```

Replace the definition of ***UpdateProblem*** *with:*

```
UpdateProblem  ::=  INTEGER {
       namingViolation                   (1),
       objectClassViolation              (2),
       notAllowedOnNonLeaf               (3),
       notAllowedOnRDN                   (4),
       entryAlreadyExists                (5),
       affectsMultipleDSAs               (6),
       objectClassModificationProhibited  (7),
       noSuchSuperior                    (8),
       notAncestor                       (9),
       parentNotAncestor                 (10),
       hierarchyRuleViolation            (11),
       familyRuleViolation               (12),
       insufficientPasswordQuality       (13),
       passwordInHistory                 (14),
       noPasswordSlot                    (15) }
```

ISO/IEC 9594-4: 2008, Information Technology – Open Systems Interconnection – The Directory: Procedures for distributed operation

1) Annex A

Replace the following text:

```
-- from ITU-T Rec. X.511 | ISO/IEC 9594-3
       abandon, addEntry, CommonResults, compare, directoryBind, list,
       modifyDN, modifyEntry, read, referral, removeEntry, search, SecurityParameters
             FROM DirectoryAbstractService directoryAbstractService
```

with:

```
-- from ITU-T Rec. X.511 | ISO/IEC 9594-3
       abandon, addEntry, changePassword, administerPassword, CommonResults, compare,
       directoryBind, list, modifyDN, modifyEntry, read, referral, removeEntry, search,
       SecurityParameters
             FROM DirectoryAbstractService directoryAbstractService
```

Add, at the end of the list of chained operations:

```
chainedChangePassword        OPERATION ::=  chained { changePassword }

chainedAdministerPassword    OPERATION ::=  chained { chainedAdministerPassword }
```

ISO/IEC 9594-5: 2008, Information Technology – Open Systems Interconnection – The Directory: Protocol specifications

1) Subclause 6.4.1 and Annex A

Add after ***`id-opcode-modifyDN`****, the following definition:*

```
id-opcode-changePassword     Code      ::= local:11
id-opcode-admnisterPassword  Code      ::= local:12
```

2) Subclause 12.2.3

Add the following note at the end of 12.2.3:

NOTE – This latter extension rule requires that any new operation that might be chained must have its argument defined as a sequence type where the first component shall be the name of the object to which the operation is addressed.

3) Subclause 13.1.1

Replace the paragraph b) with the following text:

b) The bind security level(s) for which conformance is claimed (none, simple, strong – and if simple, then whether without password, with password or with protected-password); and whether the DUA can generate signed arguments or validate signed results;

Replace the paragraph e) and f) with the following texts:

e) If conformance is claimed to the application-context specified by **directoryAccessAC** and/or associated with the **dap-ip** protocol, the bind security level(s) for which conformance is claimed (none, simple, strong, SPKM, SASL – and if simple, then whether without password, with password, with protected password or the **userPwd** is supported for password policy); whether the DSA can perform originator authentication as defined in 22.1 of ITU-T Rec. X.518 | ISO/IEC 9594-4 and if so, whether identity-based or signature-based; and whether the DSA can perform result authentication as defined in 22.2 of ITU-T Rec. X.518 | ISO/IEC 9594-4.

f) If conformance is claimed to the application-context specified by **directorySystemAC** and/or associated with the **dsp-ip** protocol, the bind security level(s) for which conformance is claimed (none, simple, strong, SPKM, SASL – and if simple, then whether without password, with password, or with protected password); whether the DSA can perform originator authentication as defined in 22.1 of ITU-T Rec. X.518 | ISO/IEC 9594-4 and if so, whether identity-based or signature-based; and whether the DSA can perform result authentication as defined in 22.2 of ITU-T Rec. X.518 | ISO/IEC 9594-4.

ISO/IEC 9594-6: 2008, Information Technology – Open Systems Interconnection – The Directory: Selected attribute types

1) Subclause 2.2

Add a new reference after IETF RFC 3454

– IETF RFC 3986 (2005), *Uniform Resource Identifie (URI): Generic Syntax*.

2) Subclause 6.2.12

Add a new clause 6.2.12

6.2.12 uRI

The *URI* attribute type is used for holding a Uniform Resource Identifier (URI) as defined in RFC 3986.

```
uRI ATTRIBUTE ::= {
       WITH SYNTAX             UTF8String
       EQUALITY MATCHING RULE  uRIMatch
       ID                      id-at-uri }
```

3) Subclause 8.9

Add a new clause 8.9

8.9 uRI Match

The *uRI Match* rule compares a presented value with an attribute value and is defined as:

```
uRIMatch  MATCHING-RULE  ::=  {
             SYNTAX     URI
             ID         id-mr-uRImatch }
```

This rule is conform to RFC 3986 6.2.2: the two UTF8String values are normalized as described in RFC 3986:

a) Case normalization: the hexadecimal digits within a percent-encoding triplet shall be normalized to use uppercase letters for digits A-F;

b) Percent-encoding normalization: any percent-encoded octet that corresponds to an unreserved character (uppercase letters, lowercase letters, digits, HYPHEN MINUS, PERIOD, LOW LINE and TILDE) shall be decoded;

c) Path segment normalization: path normalization permits simplification of a path containing "." or ".." complete path segments. This normalization uses two buffers (an input buffer containing the path and an empty output buffer which will contain the result) and loops as follows until the input buffer is empty:

- If the input buffer begins with a prefix of "../" or "./", then remove the prefix from the input buffer; otherwise,
- If the input buffer begins with a prefix of "/./" or "/.", where "." is complete path segment, then replace that prefix with "/" in the input buffer; otherwise,
- If the input buffer begins with a prefix of "/../" or "/..", where ".." is a complete path segment, then replace that prefix with "/" in the input buffer and remove the last segment and its preceding "/" (if any) from the output buffer; otherwise,
- If the input buffer consists only of "." or "..", then remove that from the input buffer; otherwise,
- Move the first path segment in the input buffer to the end of the output buffer, including the initial "/" character (if any) and any subsequent characters up to, but not including the next "/" character of the end of the input buffer

d) Scheme based normalization: components which are empty or equal to the default for the scheme shall be removed.

4) Annex A

Add after **`UUID`** *ASN.1 type definition:*

```
uRI ATTRIBUTE ::= {
       WITH SYNTAX             UTF8String
       EQUALITY MATCHING RULE  uRIMatch
       ID                      id-at-uri }
```

Add after the definition of zonalMatch:

```
uRIMatch  MATCHING-RULE  ::=  {
             SYNTAX      URI
             ID          id-mr-uRImatch }
```

Add the following definitions after the line beginning with: "-- id-at-permission":

```
id-at-uri                     OBJECT IDENTIFIER       ::=    {id-at 83}

-- id-at-pwdAttribute         OBJECT IDENTIFIER       ::=    {id-at 84}

-- id-at-userPwd              OBJECT IDENTIFIER       ::=    {id-at 85}    X.509|Part 8
```

Add the following definitions after the line beginning with: "-- id-mr-dualStringMatch":

```
id-mr-uRIMatch                OBJECT IDENTIFIER       ::=    {id-mr 70}

-- id-mr-usrPwdMatch          OBJECT IDENTIFIER       ::=    {id-mr 71}  X.509|Part8

-- id-mr-pwdEncAlgMatch       OBJECT IDENTIFIER       ::=    {id-mr 72}  X.509|Part8

-- id-mr-userPwdHistoryMatch  OBJECT IDENTIFIER       ::=    {id-mr 73}  X.509|Part8
```

ISO/IEC 9594-7: 2008, Information Technology – Open Systems Interconnection – The Directory: Selected object classes

1) Subclause 6.5 and Annex A

Add a new auxiliary class called password after userSecurityInformation

```
userPwdClass  OBJECT-CLASS  ::=  {
       KIND              auxiliary
       MAY CONTAIN       { userPwd }
       ID                id-oc-userPwdClass }
```

2) Annex A

Replace the third part of the ***IMPORTS*** *clause with:*

```
-- from ITU-T Rec. X.509 | ISO/IEC 9594-8
       authorityRevocationList, cACertificate, certificateRevocationList,
       crossCertificatePair, deltaRevocationList, supportedAlgorithms, userCertificate,
       userPassword, userPwd
             FROM AuthenticationFramework authenticationFramework
```

Add a new definition after ***id-oc-integrityInfo***

```
id-oc-userPwdClass        OBJECT IDENTIFIER ::= {id-oc 41}
```

ISO/IEC 9594-8: 2008, Information Technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks

1) Subclause 3.3

Add the following definitions after 3.3.37 and renumber the existing subclauses 3.3.38 to 3.3.60 as 3.3.41 to 3.3.63 :

3.3.38 Password expiration: the situation where a user password has reached the end of its validity period: the account is locked and the user has to change the password before doing any other directory operation.

3.3.39 Password quality attributes: attributes that specify how a password shall be constructed. Password quality attributes include things like minimum length, mixture of characters (uppercase, lowercase, figures, punctuations, etc), and avoidance of trivial passwords.

3.3.40 Password history: list of old passwords and the times they were inserted in the history..

2) Subclause 18.1.3

Renumber the existing subclause 18.1.3 as 18.1.4 and add a new subclause 18.1.3 as follows:

18.1.3 Password policy

Password policy is a set of rules that controls how passwords are used and administered in the Directory. It improves the security of the Directory and makes it difficult for password cracking programs to break into the Directory. These rules ensure that users change their passwords periodically, that passwords meet quality requirements, that re-use of old password is restricted, and that users are locked out after a certain number of failed attempts. This policy also forces the user to update its password after it has been set for the first time, or has been reset by a password administrator. However, in some cases, it is desirable to disallow users from adding and updating their own passwords.

A password is supposed not to be well known. If a password is frequently changed, the chance of misuse is minimized. Password policy administrators may deploy a password policy that causes passwords to expire after a given amount of time thus forcing users to change their passwords periodically. There must be a way to make users aware of the need to change their password before being locked out of their accounts. One or both of the following methods could be used:

– A warning may be returned to the user sometime before the password is due to expire. If the user ignores this warning before the expiration time, the account will be locked.

– The user may Bind to the directory a certain number of times after the password has expired. If the user fails to change the password following one of the 'grace' authentications, the account will be locked.

Password quality rules are rules for how a password shall be constructed. It is not the intention to provide specification for password qualities, as requirements on quality may change over time. password quality includes things like:

– minimum length;

– mixture of characters (uppercase, lowercase, figures, punctuations, etc.); and

– avoidance of trivial passwords

A particular quality rule requires specialised code within the implementation. It may therefore be advantage to standardise password quality rules and assign object identifiers to such rules. An implementation may then claim support to one or more of such standardised quality rules.

An intruder may try to guess a password to get access to protected information. Currently, two different safeguards have been identified:

– Specification of the maximum number of failed attempts before a successful attempt within a given time span (which could be indefinitely). This approach allows for "denial of service attacks". One or more genuine users could have their access to directory barred by action of an attacker.

– The other mechanism is to insert a delay before returning information on authentication failure, and increasing this delay for repeated failed authentications on the same connection. This approach slows authentication, and makes brute force attacks impractical.

Password history is a mechanism to prevent password re-use. Previously used passwords should be stored to allow the Directory to ensure that a new password has not been previously used. Old passwords are stored for a time specified by the password policy, and after this time a password may be re-used. The history is maintained in a **`userPwdHistory`** multi-valued operational attribute. A value is purged after a specific time, and the purged password may in principle be reused. The maximum time a password is kept in the in **`userPwdHistory`** attribute is specified in the **`pwdMaxTimeInHistory`** operational attribute, and the minimum time is specified in the **`pwdMinTimeInHistory`** operational attribute. The number of passwords stored is limited by the **`pwdHistorySlots`** operational attribute and the password cannot be changed if there is no free slot in the history and no passwords in the history have been for less than the **`pwdMinTimeInHistory`**, so a user cannot revert to a "preferred password" simply by making lots of password changes.

The password policy can be used with clear passwords (using the **`clear`** alternative of the **`userPwd`** attribute), or with encrypted passwords (using the **`encrypted`** alternative of the **`userPwd`** attribute) or with another password attribute. All entries in the same specific password administrative area shall use the same password attribute.

The password policy uses specific operational attributes to register policy parameters, times and dates related to password management.
When a password value is first stored in the directory, in the **`userPwd`** attribute, the **`pwdStartTime`** operational attribute is set (figure 10). The **`pwdExpiryTime`** operational attribute which contains the expiration of the password may either be automatically computed from the **`pwdExpiryAge`** operational attribute or set by explicit administrator action. It is an implementation option whether the value is dynamically computed by addition of the **`pwdExpiryAge`** to the **`pwdStartTime`** of the entry, in which case it does not need to be stored in the directory entry, or is set by an administrator, in which case it shall be stored in the directory entry. The **`pwdEndTime`** operational attribute which contains the expiration of the account may either be automatically computed from the **`pwdMaxAge`** operational attribute or set by explicit administrator action. It is an implementation option whether the value is dynamically computed by addition of the **`pwdMaxAge`** to the **`pwdStartTime`** of the entry, in which case it does not need to be stored in the directory entry, or is set by an administrator, in which case it shall be stored in the directory entry.

The **`pwdStartTime`** operational attribute may also be set by an Administrator to specify that the account cannot be used before a given time. If the **`pwdStartTime`** contains a time greater than **`pwdEndTime`**

When the user (or an administrator acting on behalf of the user) changes the **`userPwd`** attribute within the **`pwdMaxAge`** period, the **`pwdStartTime`** operational attribute should be updated. The **`pwdExpiryTime`** and the **`pwdEndTime`** operational attributes should be recomputed and updated to reflect the new password creation time.

NOTE – If a user does not log into the Directory for a long time, the values of **`pwdExpiryTime`** and **`pwdEndTime`** operational attributes may be exceeded and the account automatically locked.

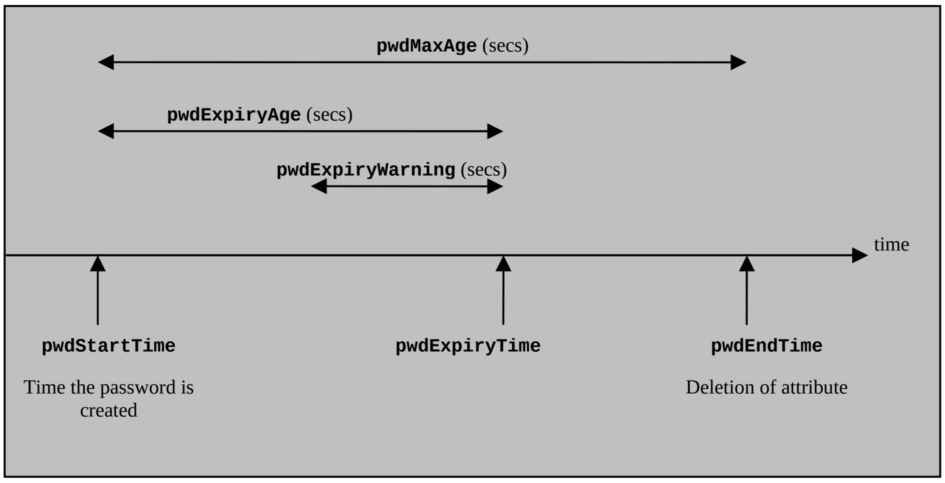

Figure 10 – Time chart for password attributes

When the user (or an administrator acting on behalf of the user) changes the value of the password, the new value is generally not known by all the Directory servers immediately because of replication delays. To prevent authentication problems, the previous password remains available for the **`pwdRecentlyExpiredDuration`** duration time (which shall be greater than the replication periods used in the Directory system).

When the user (or an administrator acting on behalf of the user) changes the value of the password, the old value should be copied into the recently expired password attribute. (The **`userPwd`** attribute is copied into the **`userPwdRecentlyExpired)`**. When the recently expired password duration time is over, the recently expired password attribute (**`userPwdRecentlyExpired`**) should be deleted. If the user (or an administrator acting on behalf of the user) changes their password again during the recently expired password duration time, then their recently expired password should be overwritten and the duration should be set to start again (see figure 11). Thus a recently expired password will only be kept in the recently expired password attribute for the shorter of the recently expired password duration time or until the user changes their password again. However, it will be kept in the password history table.

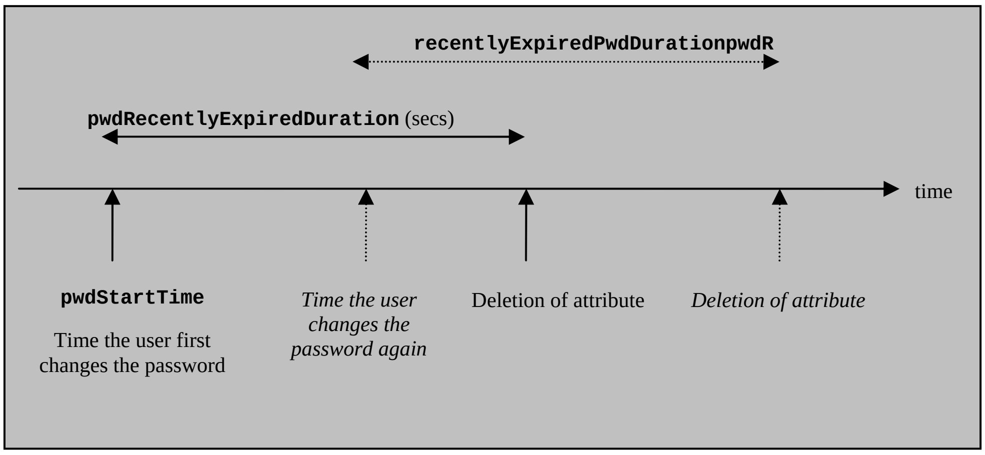
recentlyExpiredPwdDurationpwdR
pwdRecentlyExpiredDuration (secs)
time
pwdStartTime
Time the user first changes the password
Time the user changes the password again
Deletion of attribute
Deletion of attribute

Figure 11 – Time line for recently expired passwords

The password history attribute is used to prevent password re-use, by storing old values of the user's password so that the user cannot re-use the same password again whilst it is stored in the password history (see figure 12). When the user (or an administrator acting on behalf of the user) changes their password, it may be copied into the password history (**`userPwdHistory`**) operational attribute along with the time that the password was changed. The password maximum time in history attribute (**`pwdMaxTimeInHistory`**) specifies, the maximum duration (in seconds) that a password should remain in the password history. Once this time has expired for a particular password, then it is removed from the password history, and the user may use this password again.

The number of slots in the password history table (or password history attribute values) is defined in the **`pwdHistorySlots`** operational attribute. When all the slots are filled, the oldest password may be removed subject to it having been in the history for a minimum duration time (as specified in the **`pwdMinTimeInHistory`** attribute). If the user forgets his password when all the history slots are full and no password are older than **`pwdMinTimeInHistory`**, then the administrator must free two slots in the history table (i.e. delete two attribute values), reset the user's password to a temporary value (which is copied into the history), leaving one spare slot for the user to choose their own new password.

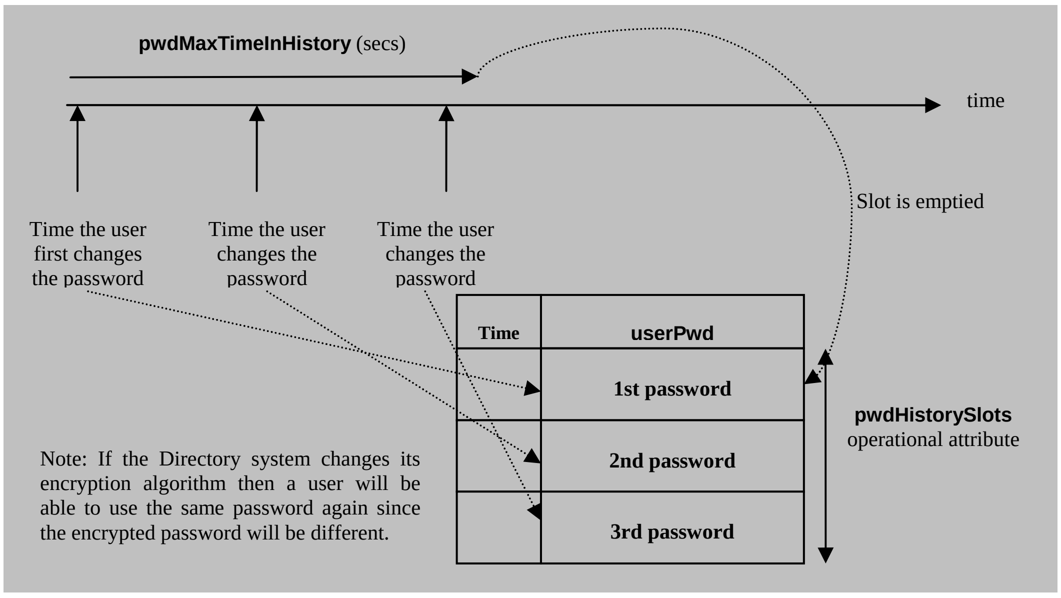
pwdMaxTimeInHistory (secs)
time
Slot is emptied
Time the user first changes the password
Time the user changes the password
Time the user changes the password
Time
userPwd
1st password
2nd password
3rd password
pwdHistorySlots operational attribute
Note: If the Directory system changes its encryption algorithm then a user will be able to use the same password again since the encrypted password will be different.

Figure 12 – userPwdHistory attribute

3) Subclause 18.1.4

Replace the current text preceding the **`userPassword`** *definition with:*

18.1.4 User Passwords attribute type

The multi-valued User Passwords attribute type contains the current and possibly previous passwords of an object. An attribute value for a user password is a string specified by the object.

4) Subclause 18.1.5

Add a new subclause 18.1.5

18.1.5 Simple Authentication attributes held by object entries

18.1.5.1 User Password attribute

The **userPwd** attribute type contains either the clear text password or the encrypted password of an object. The Directory can store either variants but, implementations be aware that storing encrypted passwords is not always compatible with passing encrypted passwords in the protocol. The encrypted alternative may be used for passing the password in the bind or compare operations but this can only be safely used when the passwords are stored in the clear (see section 18.1.8 **userPwdMatch** for more details). The attribute value of the encrypted alternative is an octet string containing the encrypted value, with the encryption algorithm identifier as well as parameters such as seeds. During password rollover, the old password value may be copied into the **userPwdRecentlyExpired** attribute value.

```
userPwd ATTRIBUTE  ::= {
       WITH SYNTAX            UserPwd
       EQUALITY MATCHING RULE userPwdMatch
       SINGLE VALUE           TRUE
       ID                     id-at-userPwd }

UserPwd ::= CHOICE {
       clear  UTF8String,
       encrypted   SEQUENCE {
             algorithmIdentifier    AlgorithmIdentifier{{SupportedAlgorithms}},
             encryptedString        OCTET STRING}}
```

Annex L contains examples of some encryption methods.

18.1.5.2 Password Creation Time

The **pwdStartTime** operational attribute indicates when the password has been created for the object represented by the entry in which the attribute is present.

```
pwdStartTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE generalizedTimeOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdStartTime }
```

18.1.5.3 Password Expiry Time

The **pwdExpiryTime** operational attribute indicates when the password will expire for the object represented by the entry in which the attribute is present. This is an optional attribute that can be set by an administrator. If the attribute is missing, its default value is computed by the addition of the **pwdExpiryAge** to the **pwdStartTime** of the entry.

```
pwdExpiryTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE generalizedTimeOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdExpiryTime }
```

18.1.5.4 Password End Time

The **pwdEndTime** operational attribute indicates when the password will be no longer valid for the object represented by the entry in which the attribute is present. This is an optional attribute that can be set by an administrator. If the attribute is missing, its default value is computed by the addition of the **pwdMaxAge** to the **pwdStartTime** of the entry.

```
pwdEndTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE  generalizedTimeOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdEndTime }
```

18.1.5.5 Password Fails attribute

The **`pwdFails`** operational attribute specifies the current number of consecutive failed bind or compare attempts on the password attribute. The value of this attribute is incremented by one after a failed bind or compare attempt and is reset to zero after a successful bind or compare operation.

```
pwdFails   ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (0..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   dSAOperation
       ID                      id-oa-pwdFails }
```

18.1.5.6 Password Failure Time attribute

The **`pwdFailureTime`** operational attribute specifies the time of the last failed bind or compare attempts on the password attribute. This attribute is only significant when the pwdFails operational attribute contains a non zero value.

```
pwdFailureTime ATTRIBUTE ::= {
       WITH SYNTAX             GeneralizedTime
       EQUALITY MATCHING RULE  generalizedTimeMatch
       ORDERING MATCHING RULE  generalizedTimeOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   dSAOperation
       ID                      id-oa-pwdFailureTime }
```

18.1.5.7 Graces Used attribute

The **`pwdGracesUsed`** operational attribute specifies the number of grace authentication attempts that have already been used with an expired password. The value of this attribute is set to 0 when the password is changed and incremented by one after successful authentication using an expired password. When the value is greater or equal to the **`pwdGraces`** attribute , the password is not usable again.

```
pwdGracesUsed ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (0..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   dSAOperation
       ID                      id-oa-pwdGracesUsed }
```

18.1.5.8 Password History

The **`userPwdHistory`** operational attribute is used to hold previous passwords for the user represented by the entry in which the attribute is present.

```
userPwdHistory ATTRIBUTE ::= pwdHistory{userPwd,userPwdHistoryMatch,id-oa-userPwdHistory}
```

This attribute is multi-valued. Each value consists of a sequence of the time the password was put in the history and the password.

18.1.5.9 Recently Expired Password

The **`userPwdRecentlyExpired`** attribute type contains the old user password after it has been replaced during the **`pwdRecentlyExpiredDuration`**. During this period, this password and the **`userPwd`** attribute are both considered to be valid. This attribute is removed when the **`pwdRecentlyExpiredDuration`** expires.

```
userPwdRecentlyExpired ATTRIBUTE ::= pwdRecentlyExpired{userPwd,id-oa-
userPwdRecentlyExpired}
```

5) Subclause 18.1.6

Add a new subclause 18.1.6

18.1.6 Password policy attributes

Password policy attributes may be placed in an object entry and/or in a subentry. If an object entry holds such an attribute and is also within the scope of a password administration subentry, the value of the attribute in the object entry itself takes precedence.

18.1.6.1 ModifyEntry Password Allowed attribute

The **pwdModifyEntryAllowed** operational attribute specifies if the password or the encrypted password of an entry can be modified by an Administrator with a Modify Entry operation. If this attribute is missing, or the value is FALSE, the password or the encrypted password cannot be modified with a Modify Entry operation.

```
pwdModifyEntryAllowed ATTRIBUTE ::= {
       WITH SYNTAX            BOOLEAN
       EQUALITY MATCHING RULE booleanMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdModifyEntryAllowed }
```

18.1.6.2 Change Password Allowed attribute

The **pwdChangeAllowed** operational attribute specifies if the password or the encrypted password of an entry can be modified by the owner of that entry with a Change Password operation. If this attribute is missing or the value is FALSE, the password or the encrypted password cannot be modified with a Change Password operation.

```
pwdChangeAllowed ATTRIBUTE ::= {
       WITH SYNTAX            BOOLEAN
       EQUALITY MATCHING RULE booleanMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdChangeAllowed }
```

18.1.6.3 Password Maximum Age attribute

The **pwdMaxAge** operational attribute holds the number of seconds after which a password will be no longer available. It shall have a value greater than zero.

If this attribute is missing, then the default value is infinity

```
pwdMaxAge ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1 .. MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMaxAge }
```

18.1.6.4 Password Expiry Age attribute

The **pwdExpiryAge** operational attribute holds the number of seconds after which a modified password will expire. It shall have a value greater than zero.

If this attribute is missing, then the default value is infinity

```
pwdExpiryAge ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1 .. MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdExpiryAge }
```

18.1.6.5 Passwords Quality Rule Attributes

18.1.6.5.1 Minimum Password Length Attribute

This specifies the minimum length, in characters, which is acceptable for a password.

```
pwdMinLength ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER
       EQUALITY MATCHING RULE integerMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMinLength }
```

18.1.6.5.2 Password Vocabulary Attribute

This specifies the type of word that are forbidden to be used for passwords. It a bit is set, the corresponding type of word is not allowed to be used on its own as a password.

```
pwdVocabulary ATTRIBUTE ::= {
       WITH SYNTAX            BIT STRING
                                    noDictionaryWords(0),
                                    noPersonNames(1),
                                    noGeographicalNames(2)
                                    }
       EQUALITY MATCHING RULE bitStringMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdVocabulary }
```

18.1.6.5.3 Password Alphabet Attribute

This specifies the sets of characters that shall be used in creating a password. The password shall contain at list one character of earch UTF8String of the value.

```
pwdAlphabet ATTRIBUTE ::= {
       WITH SYNTAX            SEQUENCE OF UTF8String
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdAlphabet }
```

18.1.6.5.4 Password Dictionaries Attribute

This attributes points to one or more dictionaries containing words that are forbidden from being passwords on their own.

```
pwdDictionaries ATTRIBUTE ::= {
             SUBTYPE OF             uRI
             USAGE                  directoryOperation
             ID                     id-oa-pwdDictionaries }
```

Editor Note: the attribute uRI will be defined in Amendment 1 on ISO/IEC 9594-All parts (Communication support enhancements) as follows:

```
uRI ATTRIBUTE ::= {
       WITH SYNTAX            UTF8String
       EQUALITY MATCHING RULE caseExactMatch
       ID                     id-at-uri }

id-at-uri                     OBJECT IDENTIFIER       ::=    {id-at 83}
```

18.1.6.6 Password Expiry Warning attribute

The **pwdExpiryWarning** operational attribute specifies a period in seconds before password expiration. During this period a warning indication shall be returned whenever an authenticating requestor binds. If this attribute is missing, then a warning indication shall not be returned.

```
pwdExpiryWarning ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdExpiryWarning }
```

If the user does not attempt to bind during this period, the account should be locked, but the user should have a chance to change the password.

18.1.6.7 Password Grace Limit attribute

The **pwdGraces** operational attribute specifies the number of times an expired password can be used to authenticate. If this attribute is missing, authentication shall fail.

```
pwdGraces ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdGraces }
```

18.1.6.8 Password Failure Duration attribute

The **pwdFailureDuration** operational attribute holds the number of seconds that the password cannot be used to authenticate after the first failed bind or compare attempt. How this attribute and the **pwdFails** attribute are combined to compute subsequent delays is application specific (e.g. could be linear or exponential). If this attribute is missing, the default time is zero.

```
pwdFailureDuration ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdFailureDuration }
```

18.1.6.9 Password Lockout Duration attribute

The **pwdLockoutDuration** operational attribute holds the number of seconds that the password cannot be used to authenticate due to too many successive failed bind or compare attempts (more than the limit specified by **pwdMaxFailures** operational attribute or its default). If this attribute is missing, the default time is infinity.

```
pwdLockoutDuration ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdLockoutDuration }
```

18.1.6.10 Password Maximum Failures attribute

The **pwdMaxFailures** operational attribute specifies the number of consecutive failed bind or compare attempts after which the password may not be used to authenticate. If this attribute is missing, there is no limit on failed attempts.

```
pwdMaxFailures ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMaxFailures }
```

18.1.6.11 Password Maximum Time in History attribute

The **pwdMaxTimeInHistory** operational attribute specifies the maximum time, in number of seconds, during which a replaced password is kept within the **userPwdHistory** operational attribute. If this attribute is missing, the default is infinity.

```
pwdMaxTimeInHistory ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMaxHimeInHistory }
```

18.1.6.12 Password Minimum Time in History attribute

The **pwdMinTimeInHistory** operational attribute specifies the minimum time, in number of seconds, during which a replaced password shall be kept within the **userPwdHistory** operational attribute. If this attribute is missing, the default time is zero seconds.

```
pwdMinTimeInHistory ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMinHimeInHistory }
```

18.1.6.13 Password History slots attribute

The **`pwdHistorySlots`** operational attribute specifies the number of slots in the history which can be used to store replaced passwords. The minimum number of slots is 2 because two slots are needed when an administrator has to reset a password.

```
pwdHistorySlots ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (2..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdHistorySlots }
```

18.1.6.14 Recently Expired Password Duration

The **`pwdRecentlyExpiredDuration`** attribute type defines the period during which an expired password is kept in the **`userPwdRecentlyExpired`** attribute.

```
pwdRecentlyExpiredDuration ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdRecentlyExpiredDuration }
```

18.1.6.15 Password Encryption Algorithm attribute

The **`pwdEncryptionAlg`** operational attribute indicates the algorithm to be used during the creation of an encrypted password.

```
pwdEncAlg ATTRIBUTE ::= {
       WITH SYNTAX            PwdEncAlg
       EQUALITY MATCHING RULE pwdEncAlgMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdEncAlg }

PwdEncAlg ::= AlgorithmIdentifier{{SupportedAlgorithms}}
```

6) Subclause 18.1.8

Add a new subclause 18.1.8

18.1.8 User Password matching rule

The **`userPwdMatch`** rule determines whether a presented clear text or encrypted password matches a clear text password stored in the Directory.

```
userPwdMatch MATCHING-RULE ::= {
       SYNTAX      UserPwd
       ID          id-mr-userPwdMatch }
```

It the presented password is clear text and the stored password is clear text, then comparison is performed using caseExactMatch

If the presented password is clear text and the stores password is encrypted, then the clear text assertion is encrypted using the algorithm identified in the stored password and the encrypted value is compared with the stored value using octetStringMatch.

If the presented password is encrypted and the stored password is clear text, then comparison is performed by encrypting the stored password using the encryption algorithm passed in the assertion and then the encrypted password is compared to the asserted encrypted password using octetStringMatch.

If the presented password is encrypted and the stored password is encrypted, then the algorithm ids and algorithm parameters are compared for equality. If they are different, matching fails. If they are the same, the encrypted passwords are compared using octetStringMatch.

7) Subclause 18.1.10

Add a new subclause 18.1.11

18.1.11 Password history matching rule

The **`userPwdHistoryMatch`** rule compares for equality a presented clear or encrypted password with a clear text or encrypted password stored as an attribute value of type **`pwdHistory`**. The timestamp component present in the **`userPwdHistory`** is ignored. The remaining passwords are compared using the userPwdMatch. Matching rule.

```
userPwdHistoryMatch MATCHING-RULE ::= pwdHistoryMatch{userPwd,id-mr-userPwdHistoryMatch}
```

8) SubClause 18.2.1

Rename the figures 10 and 11 to 13 and 14

9) SubClause 18.2.2.1

Rename the figure 12, 13 and 14 to 15, 16 and 17

10) Annex A.1

Replace the first three parts of IMPORTS clause with:

```
       id-at, id-nf, id-oa, id-mr, id-oc, id-sc, informationFramework,
       selectedAttributeTypes, basicAccessControl,certificateExtensions
             FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1)
                   usefulDefinitions(0) 6}

       Name, ATTRIBUTE, OBJECT-CLASS, NAME-FORM, top, MATCHING-RULE, DistinguishedName,
       pwdHistory{}, pwdRecentlyExpired{}, pwdHistoryMatch{}
             FROM InformationFramework informationFramework

       UniqueIdentifier, octetStringMatch, generalizedTimeMatch,caseExactMatch,
       generalizedTimeOrderingMatch,integerMatch, distinguishedNameMatch, booleanMatch,
       bitStringMatch, objectIdentifierMatch, commonName, UnboundedDirectoryString, uRI
             FROM SelectedAttributeTypes selectedAttributeTypes
```

*Add the following definitions after **`userPassword`** definition:*

```
userPwd ATTRIBUTE  ::= {
       WITH SYNTAX            UserPwd
       EQUALITY MATCHING RULE userPwdMatch
       SINGLE VALUE           TRUE
       ID                     id-at-userPwd }

UserPwd ::= CHOICE {
       clear UTF8String,
       encrypted   SEQUENCE {
       algorithmIdentifier    AlgorithmIdentifier{{SupportedAlgorithms}},
       encryptedString        OCTET STRING}}
```

-- Operational attributes --

```
pwdStartTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE generalizedTimeOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdStartTime }

pwdExpiryTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE generalizedTimeOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdExpiryTime }
```

```
pwdEndTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE generalizedTimeOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdEndTime }

pwdFails   ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  dSAOperation
       ID                     id-oa-pwdFails }

pwdFailureTime ATTRIBUTE ::= {
       WITH SYNTAX            GeneralizedTime
       EQUALITY MATCHING RULE generalizedTimeMatch
       ORDERING MATCHING RULE generalizedTimeOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  dSAOperation
       ID                     id-oa-pwdFailureTime }


pwdGracesUsed ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  dSAOperation
       ID                     id-oa-pwdGracesUsed }

userPwdHistory ATTRIBUTE ::= pwdHistory{userPwd,userPwdHistoryMatch,id-oa-userPwdHistory}

userPwdRecentlyExpired ATTRIBUTE ::= pwdRecentlyExpired{userPwd,id-oa-
userPwdRecentlyExpired}

pwdModifyEntryAllowed ATTRIBUTE ::= {
       WITH SYNTAX            BOOLEAN
       EQUALITY MATCHING RULE booleanMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdModifyEntryAllowed }

pwdChangeAllowed ATTRIBUTE ::= {
       WITH SYNTAX            BOOLEAN
       EQUALITY MATCHING RULE booleanMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdChangeAllowed }

pwdMaxAge ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1 .. MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMaxAge }

pwdExpiryAge ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (1 .. MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdExpiryAge }

pwdMinLength ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER
       EQUALITY MATCHING RULE integerMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdMinLength }

pwdVocabulary ATTRIBUTE ::= {
       WITH SYNTAX       BIT STRING {
                               noDictionaryWords(0),
                               noPersonNames(1),
                               noGeographicalNames(2)
                               }
       EQUALITY MATCHING RULE  bitStringMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdVocabulary }

pwdAlphabet ATTRIBUTE ::= {
       WITH SYNTAX             SEQUENCE OF UTF8String
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdAlphabet }

pwdDictionaries ATTRIBUTE ::= {
       WITH SYNTAX             uRI
       USAGE                   directoryOperation
       ID                      id-oa-pwdDictionaries }

pwdExpiryWarning ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (1..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdExpiryWarning }

pwdGraces ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (0..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdGraces }

pwdFailureDuration ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER(0..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdFailureDuration }

pwdLockoutDuration ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (0..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdLockoutDuration }

pwdMaxFailures ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (1..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdMaxFailures }

pwdMaxTimeInHistory ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (1..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                      id-oa-pwdMaxTimeInHistory }

pwdMinTimeInHistory ATTRIBUTE ::= {
       WITH SYNTAX             INTEGER (0..MAX)
       EQUALITY MATCHING RULE  integerMatch
       ORDERING MATCHING RULE  integerOrderingMatch
       SINGLE VALUE            TRUE
       USAGE                   directoryOperation
       ID                     id-oa-pwdMinHimeInHistory }

pwdHistorySlots ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (2..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdHistorySlots }

pwdRecentlyExpiredDuration ATTRIBUTE ::= {
       WITH SYNTAX            INTEGER (0..MAX)
       EQUALITY MATCHING RULE integerMatch
       ORDERING MATCHING RULE integerOrderingMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdRecentlyExpiredDuration }

pwdEncAlg ATTRIBUTE ::= {
       WITH SYNTAX            PwdEncAlg
       EQUALITY MATCHING RULE pwdEncAlgMatch
       SINGLE VALUE           TRUE
       USAGE                  directoryOperation
       ID                     id-oa-pwdEncAlg }

PwdEncAlg ::= AlgorithmIdentifier{{SupportedAlgorithms}}
```

-- User Password matching Rule --

```
userPwdMatch MATCHING-RULE ::= {
       SYNTAX      OCTET STRING,
       ID          id-mr-userPwdMatch }
```

-- Password Encryption Algorithm matching Rule --

```
pwdEncAlgMatch MATCHING-RULE ::= {
       SYNTAX      pwdEncAlg,
       ID          id-mr-pwdEncAlgMatch }
```

-- User Password History matching Rule --

```
userPwdHistoryMatch MATCHING-RULE ::= pwdHistoryMatch(userPwd,
                                          id-mr-userPwdHistoryMatch}
```

Add the following definitions after id-at-pkiPath:

```
id-at-userPwd                OBJECT IDENTIFIER      ::=    {id-at 85}
```

Add the following definitions before the line "END":

-- operational attributes --

```
id-oa-pwdStartTime              OBJECT IDENTIFIER   ::=    {id-oa 22}

id-oa-pwdExpiryTime             OBJECT IDENTIFIER   ::=    {id-oa 23}

id-oa-pwdEndTime                OBJECT IDENTIFIER   ::=    {id-oa 24}

id-oa-pwdFails                  OBJECT IDENTIFIER   ::=    {id-oa 25}

id-oa-pwdFailureTime            OBJECT IDENTIFIER   ::=    {id-oa 26}

id-oa-pwdGracesUsed             OBJECT IDENTIFIER   ::=    {id-oa 27}

id-oa-userPwdHistory            OBJECT IDENTIFIER   ::=    {id-oa 28}

id-oa-userPwdRecentlyExpired    OBJECT IDENTIFIER   ::=    {id-oa 29}

id-oa-pwdModifyEntryAllowed     OBJECT IDENTIFIER   ::=    {id-oa 30}

id-oa-pwdChangeAllowed          OBJECT IDENTIFIER   ::=    {id-oa 31}

id-oa-pwdMaxAge                 OBJECT IDENTIFIER   ::=    {id-oa 32}

id-oa-pwdExpiryAge              OBJECT IDENTIFIER   ::=    {id-oa 33}

id-oa-pwdMinLength              OBJECT IDENTIFIER   ::=    {id-oa 34}

id-oa-pwdVocabulary             OBJECT IDENTIFIER   ::=    {id-oa 35}

id-oa-pwdAlphabet               OBJECT IDENTIFIER   ::=    {id-oa 36}
```

```
id-oa-pwdDictionaries            OBJECT IDENTIFIER   ::=   {id-oa 37}
id-oa-pwdExpiryWarning           OBJECT IDENTIFIER   ::=   {id-oa 38}
id-oa-pwdGraces                  OBJECT IDENTIFIER   ::=   {id-oa 39}
id-oa-pwdFailureDuration         OBJECT IDENTIFIER   ::=   {id-oa 40}
id-at-pwdLockoutDuration         OBJECT IDENTIFIER   ::=   {id-oa 41}
id-oa-pwdMaxFailures             OBJECT IDENTIFIER   ::=   {id-oa 42}
id-oa-pwdMaxTimeInHistory        OBJECT IDENTIFIER   ::=   {id-oa 43}
id-oa-pwdMinTimeInHistory        OBJECT IDENTIFIER   ::=   {id-oa 44}
id-oa-pwdHistorySlots            OBJECT IDENTIFIER   ::=   {id-oa 45}
id-oa-pwdRecentlyExpiredDuration OBJECT IDENTIFIER   ::=   {id-oa 46}
id-oa-pwdEncryptionAlg           OBJECT IDENTIFIER   ::=   {id-oa 47}
-- matching rules
Id-mr-userPwdMatch               OBJECT IDENTIFIER   ::=   {id-mr 71}
id-mr-userPwdHistoryMatch        OBJECT IDENTIFIER   ::=   {id-mr 72}
id-mr-pwdEncAlgMatch             OBJECT IDENTIFIER   ::=   {id-mr 73}
```

11) Annex F

Add a new category
-- categories of object identifier --

```
nullAlgorithm OBJECT IDENTIFIER ::= {algorithm 0}
```

12) Annex L

Add a new Annex L containing the following text and rename current annexes L and M to M and N

Annex L

(This annex forms an integral part of this Recommendation | International Standard)

Examples of password encryption algorithms

L.1 Null Hashing method

The null algorithm defined in Annex F is used when no hashing is to take place.

L.2 MD5 method

The hashed password is an octet string of 16 octets which is the MD5 digest of the concatenation of the clear password and the salt which is a bit string, parameter of the algorithm. This hashing method is defined by the following object:

```
mD5Algorithm ALGORITHM ::= { BIT STRING IDENTIFIED BY
       {iso(1) member-body(2) us(840) rsadsi(113549) digestAlgorithm(2) md5(5)}}
```

L.3 SHA-1 method

The hashed password is an octet string of 20 octets which is the SHA-1 digest of the concatenation of the clear password and the salt which is a bit string, parameter of the algorithm. This hashing method is defined by the following object:

```
sha1Algorithm ALGORITHM ::= { BIT STRING IDENTIFIED BY
       {iso(1) identified-organization(3) oiw(14) secsig(3) algorithm(2) 26}}
```

ISO/IEC 9594-9: 2008, Information Technology – Open Systems Interconnection – The Directory: Replication

1) Subclause 9.2.2

Add after the third paragraph the following text:

The following attributes shall be provided by the shadow supplier in the shadowed information (entries and subentries):

- pwdModifyEntryAllowed
- pwdChangeAllowed
- pwdMaxAge
- pwdExpiryAge
- pwdMinLength
- pwdVocabulary,
- pwdAlphabet,
- pwdDictionaries,
- pwdExpiryWarning
- pwdGraces
- pwdFailureDuration
- pwdLockoutDuration
- pwdMaxFailures
- pwdTimeInHistory
- pwdHistorySlots
- pwdRecentlyExpiredDuration
- pwdEncAlg

The following attributes shall be provided by the shadow supplier in the shadowed information (entries):

- pwdStartTime
- pwdExpiryTime
- pwdEndTime
- pwdBindFails
- pwdCompareFails
- pwdGracesUsed
- userPwdHistory
- userPwdRecentlyExpired
- pwdAdminSubentryList

2) Subclause 9.2.4

Replace the text of the existing subclause 9.2.4 with:

9.2.4 Subentries

Subentries are included in the unit of replication for access control, schema, collective attributes, contexts defaults, search-rules and password policy as described below.

3) Subclause 9.2.4.5

Add the new subclause 9.2.4.5

9.2.4.5 Password policy information

To have the password policy enforced by the shadow consumer, the **`pwdPolicySubentry`** subentries shall be included in the unit of replication.

ISO/IEC 9594-10: 2008, Information Technology – Open Systems Interconnection – The Directory: Use of systems management for administration of the Directory

No change