Telecommunications and Information Exchange Between Systems

ISO/IEC JTC 1/SC 6

Document Number:	N13989
Date:	2009-06-10
Replaces:	
Document Type:	Working Draft
Document Title:	Working Draft of Amendment 2 to ISO/IEC 9594-All
Document Source:	SC 6/WG 8 Tokyo meeting
Project Number:	
Document Status:	For your information.
Action ID:	FYI
Due Date:	
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

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 14.3

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

id-ar-pwdAdminSpecificArea

2) Subclause 14.4.3

Add at the end of 14.4.3

The **pwdAdminSubentryList** operational attribute identifies all password administration subentries, 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
      NO USER MODIFICATION        TRUE
      USAGE                       directoryOperation
      ID                          id-oa-pwdAdminSubentryList }
```

3) 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      { myPwdAttribute }
      ID                id-sc-pwdAdminSubentry }
```

myPwdAttribute 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, they can only do so if they apply to different password attributes, i.e. the values of the of **myPwdAttribute** are different in the different subtree specifications.

```
myPwdAttribute          ATTRIBUTE ::= {
      WITH SYNTAX                 OBJECT IDENTIFIER
      EQUALITY MATCHING RULE      objectIdentifierMatch
      SINGLE VALUE                TRUE
      ID                          id-at-myPwdAttribute }
```

A subentry of this object class may contain the following attributes **modifyEntryPwdAllowed**, **changePwdAllowed**, **pwdMaxAge**, **pwdExpiryAge**, **pwdQualityRule**, **pwdExpiryWarning**,, **pwdGraces**, **pwdLockoutDuration**, **pwdMaxFailures**, **pwdTimeInHistory**, **pwdHistorySlots**, **recentlyExpiredPwdDuration**, **pwdEncAlg**.

Two password attributes are currently defined: **userPwd** (password stored in clear text) and **encUserPwd** (password stored encrypted). These attributes 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 the history. The attribute **pwdHistory** and the matching rule **pwdHistoryMatch** are defined for password attributes using **UTF8String** syntax. The attribute **encUserPwdHistory** and the matching rule **encPwdHistoryMatch** are defined for password attributes using **EncUserPwd** syntax

If new password attributes using other syntax 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 history attribute from the password attribute, the history matching rule and an object identifier

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

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

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

```
recentlyExpiredPassword{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 password history matching rule from the password attribute and an object identifier

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

4) Annex B

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

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

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

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

myPwdAttribute   ATTRIBUTE ::= {
        WITH SYNTAX                 OBJECT IDENTIFIER
        EQUALITY MATCHING RULE      objectIdentifierMatch
        SINGLE VALUE                TRUE
        ID                          id-at-myPwdAttribute }

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

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

recentlyExpiredPassword{ATTRIBUTE:passwordAttribute,OBJECT IDENTIFIER:id} ATTRIBUTE ::= {
```

```
        WITH SYNTAX                 passwordAttribute.&Type
        EQUALITY MATCHING RULE      passwordAttribute.&equality-match
        SINGLE VALUE                TRUE
        USAGE                       directoryOperation
        ID                          id}

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

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 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 of password operations

2) Subclause 8.1.1 and Annex A

*Replace the ASN.1 definition of **Credentials** with:*

```
Credentials ::= CHOICE {
        simple              [0]   SimpleCredentials,
        strong              [1]   StrongCredentials,
        externalProcedure   [2]   EXTERNAL,
        spkm                [3]   SpkmCredentials,
        sasl                [4]   SaslCredentials,
        encrypted           [5]   EncryptedCredentials }
```

*Add, after the definition of **ub-saslMechanism**, the following definition:*

```
EncryptedCredentials ::= SEQUENCE {
        credentials         [0]   SEQUENCE OF EncUserPwd 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

Add after paragraph 8, the following paragraph:

The **encrypted** alternative is used when the password is stored encrypted by the DSA. If the DSA does not use the encrypted password mechanism, a **SecurityError** of **inappropriateAuthentication** shall be returned. If the DSA uses the encrypted password mechanism and if the algorithms present in **credentials** component of **EncryptedCredentials** do not permit password checking a **securityError** of **inappropriateAlgorithms** with the expected algorithms shall be returned.

4) Subclause 8.1.3

Insert at end 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.graceRemaing** 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 **gracesRemaining** 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 8.3

Add new subclause 8.3

8.3 Directory Change of password

8.3.1 Directory Change of password syntax

A Directory Change of 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 change 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 {
    oldPwd    [0] CHOICE {
        simple      [0]   SimpleCredentials,
        encrypted   [1]   EncryptedCredentials },
    newPwd    [1] CHOICE {
        simple      [0]   SimpleCredentials,
        encrypted   [1]   EncUserPwd  } } }

ChangePasswordResult ::= NULL
```

8.3.2 Directory Change of password arguments

The current password (**oldPwd** component) and the new password (**newPwd** component) have to be supplied in a Change of operation. If the DSA uses clear password, the **oldPwd** and **newPwd** shall use the **simple** alternative. If the DSA uses encrypted password, the **oldPwd** and **newPwd** components shall use the **encrypted** alternative and the **oldPwd** component shall content the results of encryption of the current password using the algorithms contained in the history.

8.3.3 Directory Change of password results

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

8.3.4 Directory Change of password errors

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

– **securityError** **passwordModNotAllowed**
inappropriateAlgorithms

– **updateError** **insufficientPasswordQuality**
passwordInHistory
noPasswordSlot

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}),
        pwdQualityRule   OBJECT IDENTIFIER 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)}
```

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) **passwordModNotAllowed** – The requestor has not the right to change his own password.

l) **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 needed to check the password and optionally an object identifier value defining the quality rule.

Note 1 - 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.

Note 2 - 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.

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

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

o) **noPasswordSlot** – 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, InvokeId, OPERATION
        FROM CommonProtocolSpecification commonProtocolSpecification
```

Insert after -- Operations, arguments, and results –

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

ChangePasswordArgument ::= OPTIONALLY-PROTECTED-SEQ {
    SEQUENCE {
        oldPwd    [0] CHOICE {
            simple      [0]   SimpleCredentials,
            encrypted   [1]   EncryptedCredentials },
        newPwd    [1] CHOICE {
            simple      [0]   SimpleCredentials,
            encrypted   [1]   EncUserPwd }}}

ChangePasswordResult ::= NULL
```

Replace the definition of SecurityError with:

```
securityError  ERROR  ::=  {
    PARAMETER     OPTIONALLY-PROTECTED { SET {
                  problem         [0]   SecurityProblem,
                  spkmInfo        [1]   SPKM-ERROR,
```

```
                algorithms      [2]   SEQUENCE OF AlgorithmIdentifier{{SupportedAlgorithms}},
                                      COMPONENTS OF CommonResults } }
        CODE    id-errcode-securityError }
```

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, CommonResults, compare, directoryBind, list,
modifyDN, modifyEntry, read, referral, removeEntry, search, SecurityParameters
FROM DirectoryAbstractService directoryAbstractService

Add, a*t the end of the list of chained operations:*

chainedChangePassword OPERATION ::= chained { changePassword }

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

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

1) Annex A

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

```
-- id-at-myPwdAttributeType     OBJECT IDENTIFIER   ::=   {id-at 83}
-- id-at-userPwd                OBJECT IDENTIFIER   ::=   {id-at 84}
-- id-at-encUserPwd             OBJECT IDENTIFIER   ::=   {id-at 85}
```

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

```
-- id-mr-encUserPwdMatch        OBJECT IDENTIFIER   ::=   {id-mr 70}   X.509|Part8
-- id-mr-pwdEncAlgMatch         OBJECT IDENTIFIER   ::=   [id-mr 71]   X.509|Part8
-- id-mr-pwdHistoryMatch        OBJECT IDENTIFIER   ::=   {id-mr 72}   X.509|Part8
-- id-mr-encPwdHistoryMatch     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 5.4 and Annex A

*Replace the current definition of **OrganizationalAttributeSet** with:*

```
OrganizationalAttributeSet  ATTRIBUTE  ::=  {
        description |
        LocaleAttributeSet |
        PostalAttributeSet |
        TelecommunicationAttributeSet |
        businessCategory |
        seeAlso |
        searchGuide |
        userPassword |
        userPwd |
        userEncPwd}
```

2) Subclause 6.5 and Annex A

*Replace the current definition of **person** with:*

```
person  OBJECT-CLASS  ::=  {
        SUBCLASS OF     { top }
        MUST CONTAIN    { commonName | surname }
        MAY CONTAIN     { description |
                          telephoneNumber |
                          userPassword |
                          userPwd |
                          userEncPwd |
                          seeAlso }
        ID              id-oc-person }
```

3) 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, userEncPwd
                FROM AuthenticationFramework authenticationFramework
```

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 rules: rules that specify how a password shall be constructed. Password quality rules 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 **pwdHistory** multi-

valued operational attribute. A value is purged after a specific time, and the purged password may in principle be reused. This time period a password is kept in the in **pwdHistory** attribute is specified in the **timeInHistory** 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, 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 **userPwd** attribute), with encrypted passwords (using the **encUserPwd** attribute) or with another password attribute. All entries in the same administrative domain shall use the same password attribute (for example **userPwd** or **encUserPwd)**.

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 either the **userPwd** (figure 10) or **encUserPwd** (figure 11) attribute, the **pwdCreationTime** operational attribute respectively is set. The **pwdExpiryDate** operational attribute may either be automatically computed from the **pwdExpiryAge** operational attribute or set by explicit administrator action. The **pwdEndDate** operational attribute may either be automatically computed from the **pwdMaxAge** operational attribute or by explicit administrator action. If the **pwdExpiryWarning** operational attribute is set, then the user should be informed during this period that his or her password is about to expire.

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

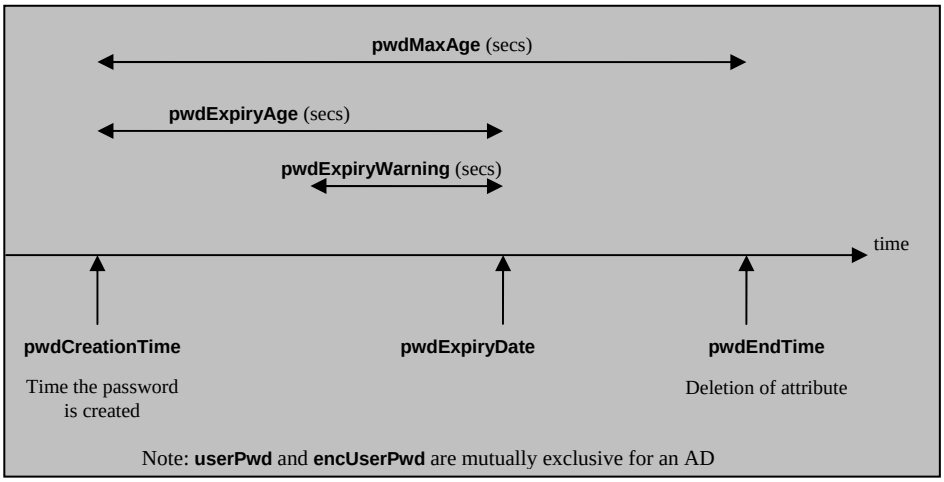

Figure 10 – UserPwd attribute

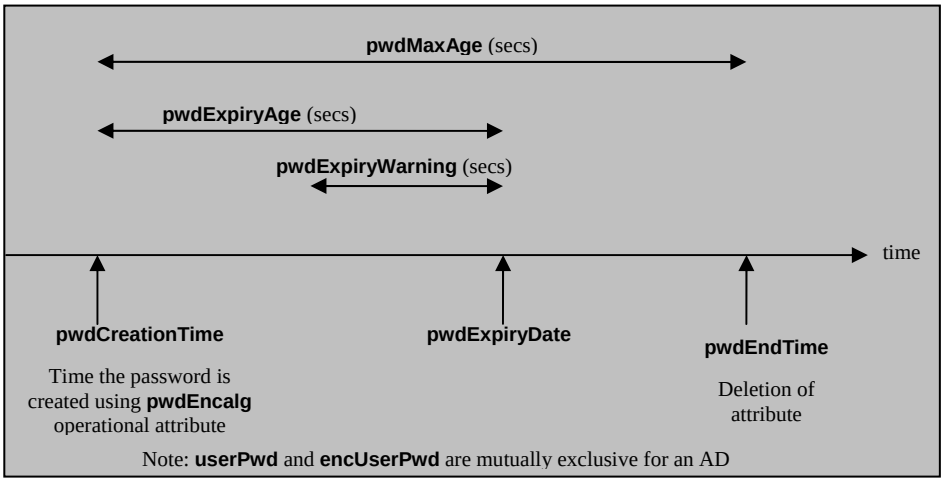

Figure 11 – encUserPwd attribute

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 remain available for the **recentlyExpiredPwdDuration** 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 **recentlyExpiredPwd** as shown in figure 12 and the **userEncPwd** is copied into the **recentlyExpiredEncPwd**. as shown in figure 13) The recently expired password creation time **expPwdCreationTime** should be set. When the recently expired password duration time is over, the recently expired password attribute (**recentlyExpiredPwd** or **recentlyExpiredEncPwd** resp.) 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. 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.

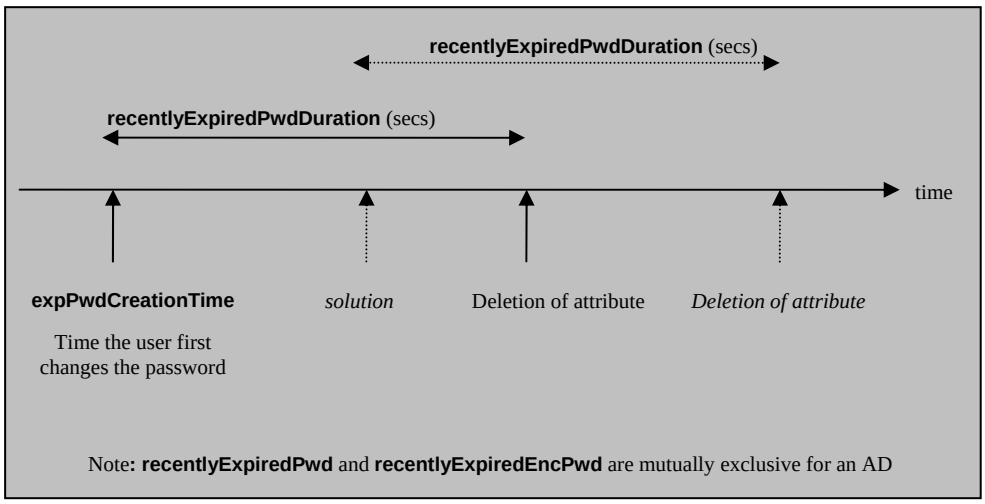

Figure 12 – recentlyExpiredPwd attribute

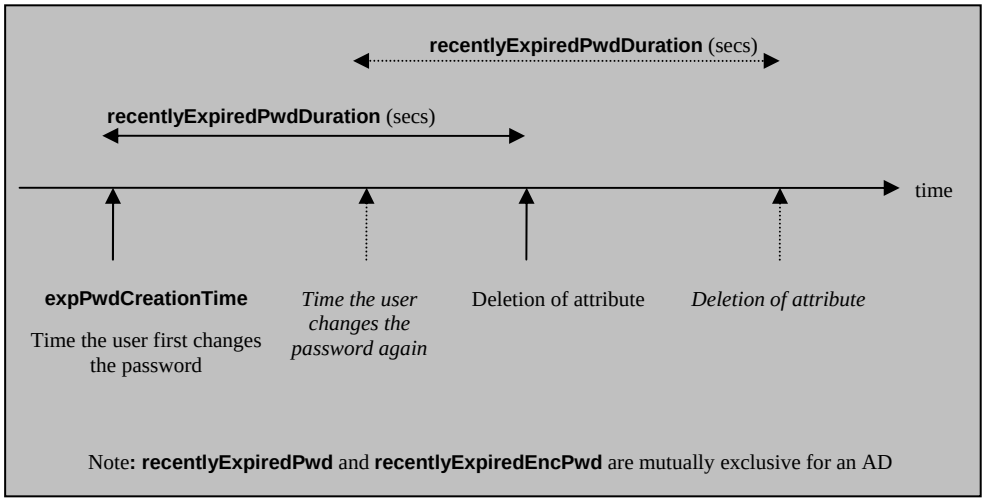

Figure 13 – recentlyExpiredEncPwd attribute

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 14). When the user (or an administrator acting on behalf of the user) changes their password, it may be copied into the password history (**pwdHistory** or **encPwdHistory**) operational attribute along with the time that the password was changed. The password time in history attribute specifies the 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 user is not allowed to change his password again until one of the old passwords has expired. If the user forgets his password when all the history slots are full, 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.

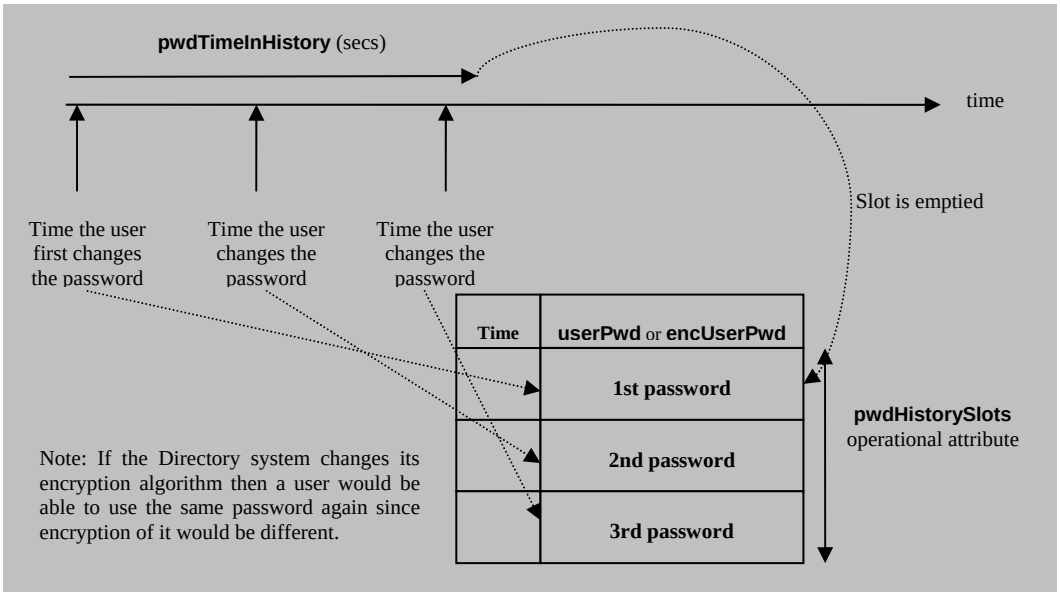

Figure 14 – pwdHistory or encPwdHistory attribute

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 the current password of an object. The attribute value for this single-valued user password is an octet string. During password rollover, the old password value may be copied into the **recentlyExpiredPwd** attribute value.

```
userPwd    ATTRIBUTE::=    {
      WITH SYNTAX                    UTF8String
      EQUALITY MATCHING RULE         caseExactMatch
      SINGLE VALUE                   TRUE
      ID                             id-at-userPwd}
```

18.1.5.2 Encrypted User Password attribute

The **EncUserPwd** attribute type contains the encrypted password of an object. The attribute value for this single-valued attribute is an octet string containing the encrypted value, with the encryption algorithm identifier. During password rollover, the old encrypted password value may be copied into the **recentlyExpiredEncPwd** attribute value.

```
encUserPwd      ATTRIBUTE::=    {
      WITH SYNTAX                    EncUserPwd
      EQUALITY MATCHING RULE         encUserPwdMatch
      SINGLE VALUE                   TRUE
      ID                             id-at-encUserPassword }

EncUserPwd ::= SEQUENCE {
      encryptedString   OCTET STRING,
      algorithmIdentifier AlgorithmIdentifier{{SupportedAlgorithms}}}
```

Annex L contains examples of two encryption methods.

18.1.5.3 Password Creation Time

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

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

18.1.5.4 Expiry Password Creation Time

The **expPwdCreationTime** operational attribute indicates when the password was latest changed for the object represented by the entry in which the attribute is present.

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

18.1.5.5 Password Expiry Date

The **pwdExpiryDate** operational attribute indicates when the password will expires for the object represented by the entry in which the attribute is present. Its value may be obtained by addition of the **pwdExpiryAge** to the **pwdCreationTime** of the entry or set by an administrator.

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

18.1.5.6 Password End Date

The **pwdEndDate** operational attribute indicates when the password will be no longer valid for the object represented by the entry in which the attribute is present. Its value may be obtained by addition of the **pwdMaxAge** to the **pwdCreationTime** of the entry or set by an administrator.

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

18.1.5.7 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                       directoryOperation
    ID                          id-oa-pwdFails }
```

18.1.5.8 Remaining Authentication Attempts attribute

The **remAuthAttempts** operational attribute specifies the number of remaining authentication attempts with an expired password before this password will be unusable. The value of this attribute is set to the value of **pwdGraces** attribute when the password is changed and decremented by one after successful authentication using an expired password. When the value reaches 0, the password is unusable.

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

18.1.5.9 Password History

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

```
pwdHistory ATTRIBUTE ::= passwordHistory{userPwd,pwdHistoryMatch,id-oa-pwdHistory}
```

18.1.5.10 Encrypted Password History

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

```
encPwdHistory ATTRIBUTE ::= passwordHistory{encUserPwd,encPwdHistoryMatch,id-oa-encPwdHistory}
```

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

18.1.5.11 Recently Expired Password

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

```
recentlyExpiredPwd ATTRIBUTE ::= recentlyExpiredPassword{userPwd,id-oa-recentlyExpiredPwd}
```

18.1.5.12 Recently Expired Encrypted Password

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

```
recentlyExpiredEncPwd ATTRIBUTE ::= recentlyExpiredPassword{encUserPwd,id-oa-recentlyExpiredEncPwd}
```

5) Subclause 18.1.6

Add a new subclause 18.1.6

18.1.6 Simple Authentication attributes held by object entries or subentries

Attributes of this type 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 **modifyEnryPwdAllowed** 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, the password or the encrypted password cannot be modified with a Modify Entry operation.

```
modifyEntryPwdAllowed ATTRIBUTE ::= {
    WITH SYNTAX                 BOOLEAN
    EQUALITY MATCHING RULE      booleanMatch
    SINGLE VALUE                TRUE
```

```
    USAGE                   directoryOperation
    ID                      id-oa-modifyEntryPwdAllowed }
```

18.1.6.2 Change Password Allowed attribute

The **changePwdAllowed** 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, the password or the encrypted password cannot be modified with a Change Password operation.

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

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 attribute

The **pwdQualityRule** operational attribute holds an identification of a password quality rule.

```
pwdQualityRule ATTRIBUTE ::= {
    WITH SYNTAX             OBJECT IDENTIFIER
    EQUALITY MATCHING RULE  objectIdentifierMatch
    SINGLE VALUE            TRUE
    USAGE                   directoryOperation
    ID                      id-oa-pwdQualityRule }
```

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 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 PasswordTime in History attribute

The **pwdTimeInHistory** operational attribute specifies the delay, in number of seconds, during which a replaced password is kept within the **pwdHistory** operational attribute. If this attribute is missing, an implementation defined default should be used.

```
pwdTimeInHistory ATTRIBUTE ::= {
        WITH SYNTAX                 INTEGER (1..MAX)
        EQUALITY MATCHING RULE      integerMatch
```

```
    ORDERING MATCHING RULE integerOrderingMatch
    SINGLE VALUE           TRUE
    USAGE                  directoryOperation
    ID                     id-oa-pwdHimeInHistory }
```

18.1.6.12 Password History slots attribute

The **pwdHistorySlots** operational attribute specifies the number of slots in the history which can be used to store replace 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 }
```

18.1.6.13 Recently Expired Password Duration

The **recentlyExpiredPwdDuration** attribute type defines the period during which an expired password is kept in the **recentlyExpiredPwd** attribute.

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

18.1.6.14 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 Encrypted User Password matching rule

The **encUserPwdMatch** rule determines whether a clear text password matches an encrypted password stored in the Directory.

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

The presented clear text password (**UTF8String**) is encrypted using the encryption algorithm stored in the **encUserPwd** attribute and then compared with the octet string stored in the **encUserPwd** attribute.

7) Subclause 18.1.9

Add a new subclause 18.1.9

18.1.9 Password Encryption Algorithm matching rule

The **pwdEncAlgMatch** rule compares for equality a presented value with an attribute value of type **pwdEncAlg**. This rule returns TRUE if the algorithms, defined as object identifier values, are equal as if the parameters, defined as bit string values, are also equal.

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

8) Subclause 18.1.10

Add a new subclause 18.1.10

18.1.10 Password history matching rule

The **pwdHistoryMatch** rule compares for equality a presented value of userPassword with an attribute value of type **pwdHistory**. This rule returns TRUE if the presented value is equal to the **password** component of the attribute value.

```
pwdHistoryMatch MATCHING-RULE ::= passwordHistoryMatch {userPwd,id-mr-pwdHistoryMatch}
        SYNTAX      PwdHistory,
        ID          id-mr-pwdHistoryMatch }
```

9) Subclause 18.1.11

Add a new subclause 18.1.11

18.1.11 Encrypted password history matching rule

The **encPwdHistoryMatch** rule compares for equality a presented value of userPassword with an attribute value of type **encPwdHistory**. This rule returns TRUE if the result of the encryption of the presented value using the algorithm specified in the parameter is equal to the password component of the attribute value.

```
encPwdHistoryMatch MATCHING-RULE ::= passwordHistoryMatch{encUserPwd,id-mr-ecnPwdHistoryMatch}
```

10) SubClause 18.2.1

Rename the figures 10 and 11 to 15 and 16

11) SubClause 18.2.2.1

Rename the figure 12, 13 and 14 to 17, 18 and 19

12) 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,
            passwordHistory{}, recentlyExpiredPassword{}, passwordHistoryMatch{}
            FROM InformationFramework informationFramework

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

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

```
userPwd    ATTRIBUTE::=    {
        WITH SYNTAX                     UTF8String
        EQUALITY MATCHING RULE          caseExactMatch
        SINGLE VALUE                    TRUE
        ID                              id-at-userPwd }

encUserPwd      ATTRIBUTE::=    {
        WITH SYNTAX                     EncUserPwd
        EQUALITY MATCHING RULE          encUserPwdMatch
        SINGLE VALUE                    TRUE
        ID                              id-at-encUserPwd }

EncUserPwd ::= SEQUENCE {
        encryptedString         OCTET STRING,
        algorithmIdentifier     AlgorithmIdentifier{{SupportedAlgorithms}}}

-- Operational attributes --

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


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

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

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

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

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

pwdHistory ATTRIBUTE ::= passwordHistory{userpwd,pwdHistoryMatch,id-oa-pwdHistory}
```

```
encPwdHistory ATTRIBUTE ::= passwordHistory{encUserPwd,encPwdHistoryMatch,id-oa-encPwdHistory}

recentlyExpiredPwd ATTRIBUTE ::= recentlyExpiredPassword{userPwd,id-oa-recentlyExpiredPwd}

recentlyExpiredEncPwd ATTRIBUTE ::= recentlyExpiredPassword{encUserPwd,id-oa-recentlyExpiredEncPwd}

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

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

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 }

pwdQualityRule ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        USAGE                           directoryOperation
        ID                              id-oa-pwdQualityRule }

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 }

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

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

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


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

PwdEncAlg ::= AlgorithmIdentifier{{SupportedAlgorithms}}

-- Encrypted User Password matching Rule --

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

-- Password Encryption Algorithm matching Rule --

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

-- Password History matching Rule --

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

-- Encrypted User Password History matching Rule --

encPwdHistoryMatch MATCHING-RULE ::= passwordHistoryMatch(encUserPwd,id-mr-encPwdHistoryMatch}
```

Add the following definitions after id-at-pkiPath:

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

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

Add the following definitions before the line "END":

```
-- operational attributes --

id-oa-pwdCreationTime               OBJECT IDENTIFIER    ::=    {id-oa 22}
id-oa-expPwdCreationTime            OBJECT IDENTIFIER    ::=    {id-oa 23}
id-oa-pwdExpiryDate                 OBJECT IDENTIFIER    ::=    {id-oa 24}
id-oa-pwdEndDate                    OBJECT IDENTIFIER    ::=    {id-oa 25}
id-oa-pwdFails                      OBJECT IDENTIFIER    ::=    {id-oa 26}
id-oa-remAuthAttempts               OBJECT IDENTIFIER    ::=    {id-oa 27}
id-oa-pwdHistory                    OBJECT IDENTIFIER    ::=    {id-oa 28}
id-oa-encPwdHistory                 OBJECT IDENTIFIER    ::=    {id-oa 29}
id-oa-recentlyExpiredPwd            OBJECT IDENTIFIER    ::=    {id-oa 30}
id-oa-recentlyExpiredEncPwd         OBJECT IDENTIFIER    ::=    {id-oa 31}
id-oa-modifyEntryPwdAllowed         OBJECT IDENTIFIER    ::=    {id-oa 32}
id-oa-changePwdAllowed              OBJECT IDENTIFIER    ::=    {id-oa 33}
id-oa-pwdMaxAge                     OBJECT IDENTIFIER    ::=    {id-oa 34}
id-oa-pwdExpiryAge                  OBJECT IDENTIFIER    ::=    {id-oa 35}
id-oa-pwdQualityRule                OBJECT IDENTIFIER    ::=    {id-oa 36}
id-oa-pwdExpiryWarning              OBJECT IDENTIFIER    ::=    {id-oa 37}
id-oa-pwdGraces                     OBJECT IDENTIFIER    ::=    {id-oa 38}
id-oa-pwdFailureDuration            OBJECT IDENTIFIER    ::=    {id-oa 39}
id-at-pwdLockoutDuration            OBJECT IDENTIFIER    ::=    {id-oa 40}
id-oa-pwdMaxFailures                OBJECT IDENTIFIER    ::=    {id-oa 41}
id-oa-pwdTimeInHistory              OBJECT IDENTIFIER    ::=    {id-oa 42}
id-oa-pwdHistorySlots               OBJECT IDENTIFIER    ::=    {id-oa 43}
id-oa-recentlyExpiredPwdDuration    OBJECT IDENTIFIER    ::=    {id-oa 44}
id-oa-pwdEncryptionAlg              OBJECT IDENTIFIER    ::=    {id-oa 45}

-- matching rules
Id-mr-encUserPwdMatch               OBJECT IDENTIFIER    ::=    {id-mr 70}
id-mr-pwdHistoryMatch               OBJECT IDENTIFIER    ::=    {id-mr 71}
id-mr-encPwdHistoryMatch            OBJECT IDENTIFIER    ::=    {id-mr 72}
id-mr-pwdEncAlgMatch                OBJECT IDENTIFIER    ::=    {id-mr 73}
```

13) Annex L

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

Annex L

Examples of password encryption algorithms

L.1 MD5 method

The encrypted 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 encryption 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.2 SHA-1 method

The encrypted 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 encryption 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):

- modifyEntryPwdAllowed
- changePwdAllowed
- pwdMaxAge
- pwdExpiryAge
- pwdQualityRule
- pwdExpiryWarning
- pwdGraces
- pwdFailureDuration
- pwdLockoutDuration
- pwdMaxFailures
- pwdTimeInHistory
- pwdHistorySlots
- recentlyExpiredPwdDuration
- pwdEncAlg

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

- pwdCreationTime
- expPwdCreationTime
- pwdExpiryDate
- pwdEndDate
- pwdBindFails
- pwdCompareFails
- remAuthAttempts
- pwdHistory
- recentlyExpiredPwd
- recentlyExpiredEncPwd
- 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