Working Implementation Agreements for Open Systems Interconnection Protocols: Part 11 - Directory Services Protocols

Output from the June 1991 NIST Workshop for Implementors of OSI

SIG Chair: **Youbong Weon-Yoon, AT&T Bell Labs**
SIG Editor: **Michael Ransom, NIST** Workshop Editor: **Brenda Gray**

Foreword

This part of the Working Implementation Agreements was prepared by the Directory Services Special Interest Group (DSSIG)of the National Institute of Standards and Technology (NIST) Workshop for Implementors of Open Systems Interconnection (OSI). See Procedures Manual for Workshop charter. Text in this part has been approved by the Plenary of the above mentioned Workshop. This part replaces the previously existing chapter on Directory Services Protocol.

Future changes and additions to this version of these Implementor Agreements will be published as change pages. Deleted and replaced text will be shown as strikeout. New and replacement text will be shown as shaded.

Table of Contents

Part 11 - Directory Services Protocols

0 Introduction

Refer to clause 0 of Stable Agreements Version 4 as of June 14, 1991.

1 Scope

Refer to clause 1 of Stable Agreements Version 4 as of June 14, 1991.

2 Normative references

Refer to clause 2 of Stable Agreements Version 4 as of June 14, 1991. .

3 Status

Refer to clause 3 of Stable Agreements Version 4 as of June 14, 1991.

4 Use of the Directory

This clause will contain introductory text.

4.1 MHS

(TBD)

4.2 FTAM

(TBD)

5 Directory ASEs and Application Contexts

Refer to clause 5 of Stable Agreements Version 4 as of June 14, 1991.

6 Schema

Refer to clause 6 of Stable Agreements Version 4 as of June 14, 1991.

6.1 Support of Structures and Naming Rules

Refer to 6.1 of Stable Agreements Version 4 as of June 14, 1991.

6.2 Support of Object Classes and Subclasses

Refer to 6.2 of Stable Agreements Version 4 as of June 14, 1991.

6.3 Support of Attribute Types

Refer to 6.3 of Stable Agreements Version 4 as of June 14, 1991.

6.4 Support of Attribute Syntaxes

Refer to 6.4 of Stable Agreements Version 4 as of June 14, 1991.

6.5 Naming Contexts

Refer to 6.5 of Stable Agreements Version 4 as of June 14, 1991.

6.6 Common Profiles

Refer to 6.6 of Stable Agreements Version 4 as of June 14, 1991.

6.6.1 OIW Directory Common Application Directory Profile

Refer to 6.6.1 of Stable Agreements Version 4 as of June 14, 1991.

6.6.1.1 Standard Application Specific Attributes and Attribute Sets

Refer to 6.6.1.1 of Stable Agreements Version 4 as of June 14, 1991.

6.6.1.2 Standard Application Specific Object Classes

Refer to 6.6.1.2 of Stable Agreements Version 4 as of June 14, 1991.

6.6.2 OIW Directory Strong Authentication Directory Profile

Refer to 6.6.2 of Stable Agreements Version 4 as of June 14, 1991.

6.6.2.1 Other Profiles Supported

Refer to 6.6.2.1 of Stable Agreements Version 4 as of June 14, 1991.

6.6.2.2 Standard Application Specific Object Classes

Refer to 6.6.2.2 of Stable Agreements Version 4 as of June 14, 1991.

6.7 Restrictions on Object Class Definitions

Refer to 6.7 of Stable Agreements Version 4 as of June 14, 1991.

7 Pragmatic Constraints

Refer to clause 7 of Stable Agreements Version 4 as of June 14, 1991.

7.1 General Constraints

Refer to 7.1 of Stable Agreements Version 4 as of June 14, 1991.

7.1.1 Character Sets

Refer to 7.1.1 of Stable Agreements Version 4 as of June 14, 1991.

7.1.2 DSP Result APDI Size

Refer to 7.1.2 of Stable Agreements Version 4 as of June 14, 1991.

In the process of chaining requests, it is possible that a chaining DSA may receive, invoke or return APDUs that exceed its capacity. A DSA shall be capable of receiving result APDUs up to and including 256K. A DSA receiving a result APDU greater than 256K may discard it.

7.1.3 Service Control (SC) Considerations

Refer to 7.1.3 of Stable Agreements Version 4 as of June 14, 1991.

7.1.4 Priority Service Control

Refer to 7.1.4 of Stable Agreements Version 4 as of June 14, 1991.

7.2 Constraints on Operations

Refer to 7.2 of Stable Agreements Version 4 as of June 14, 1991.

7.2.1 Filters

Refer to 7.2.1 of Stable Agreements Version 4 as of June 14, 1991.

7.2.2 Errors

Refer to 7.2.2 of Stable Agreements Version 4 as of June 14, 1991.

7.2.3 Error Reporting - Detection of Search Loop

Refer to 7.2.3 of Stable Agreements Version 4 as of June 14, 1991.

7.3 Constraints Relevant to Specific Attribute Types

Refer to 7.3 of Stable Agreements Version 4 as of June 14, 1991.

> **Editor's Note -** The following proposed changes in Table 1 increase the size of ub-postal-string from 30 to 60. Implementors should take note that the limit in the current Stable Agreements is 30.

Table 1 - Pragmatic Constraints for Selected Attributes

Attribute Type	Content	Constraints	Primary Source	Notes
Aliased Object Name	Distinguished Name			Note 3
Business Category	T.61 or Printable String	ub-business-category 128	CCITT X.520	
Common Name	T.61 or Printable String	ub-common- name 64	CCITT X.520	
Country Name	Printable String	2	ISO 3166	
Description	T.61 or Printable String	ub-description 1024	CCITT X.520	About 1 screen full
Destination Indicator	Printable String	ub-destination-indicator 128	CCITT X.520	
Facsimile Telephone Number	Facsimile Telephone Number	ub-telephone-number 32	CCITT X.520	Optionally includes G3 non-basic parameters (Upper bounds ffs)
International ISDN Number	Numeric String	ub-isdn-address 16	CCITT X.520	E.164 Internat'l ISDN Number
Knowledge Information	T.61 or Printable String	1024	OIW	About 1 screen full
Locality Name	T.61 or Printable String	ub-locality-name 128	CCITT X.520	
Member	Distinguished Name			Note 3
Object Class	Object Identifier	256 octets	OIW	
Organization Name	T.61 or Printable String	ub-organization-name 64	CCITT X.520	
Organizational Unit Name	T.61 or Printable String	ub-organizational-unit- name 64	CCITT X.520	
Owner	Distinguished Name			Note 3
Physical Delivery OfficeName	T.61 or Printable String	ub-physical-office-name 128	CCITT X.520	

Table 1 - Pragmatic Constraints for Selected Attributes (continued)

Attribute Type	**Content**	**Constraints**	**Primary Source**	**Notes**
Post Office Box	T.61 or Printable String	ub-post-office-box 40	CCITT X.520	
Postal Address	Postal Address	ub-postal-line6 ub-postal-string60	OIW	UPU
Postal Code	T.61 or Printable String	ub-postal-code 40	CCITT X.520	
Presentation Address	Presentation Address	224 octets	NIST	Note 2(page ?), ISO 7498.3 & X.200
Registered Address	Postal Address	ub-postal-line6 ub-postal-string60	OIW	
Role Occupant	Distinguished Name			Note 3
Search_Guide	Guide	256	OIW	
See Also	Distinguished Name			Note 3 (page ?)
Serial Number	Printable String	ub-serial-number 64	CCITT X.520	
State or Province Name	T.61 or Printable String	ub-state-name 128	CCITT X.520	
Street Address	T.61 or Printable String	ub-street-address 128	CCITT X.520	
Supported Application Context	Object Identifier	256	OIW	
Surname	T.61 or Printable String	ub-surname 64	CCITT X.520	
Telephone Number	Printable String	ub-telephone-numb er 32	CCITT X.520	E.123

Table 1 - Pragmatic Constraints for Selected Attributes (concluded)

Attribute Type	Content	Constraints	Primary Source	Notes
Teletex Terminal Identifier	Teletex Terminal Identifier	ub-teletex-terminal-id 1024	CCITT X.520	Optionally includes Teletex non-basic parameters (upper bound ffs)
Telex Number	Telex Number	ub-telex-number14 ub-country-code4 ub-answerback 8	CCITT X.520	Contains sequence of telex number, country code, and answerback
Title	T.61 or Printable String	ub-title 64	CCITT X.520	
User Password	Octet String	ub-user-password 128	CCITT X.520	Allow long passwords generated by machine
X.121 Address	Numeric String	ub-x121-address 15	CCITT X.520	X.121

NOTES

1 The pragmatic constraints of these parameters are defined in other standards. We will accommodate these values in our pragmatic constraints.

2 Presentation address is composed of "X" NSAP addresses, and three selectors, (20X + 32 + 16 + 16), e.g., if X= 1, this would be 84. These numbers are based on the most recent implementors' agreements. With 8 NSAP addresses this value is 224.

3 Pragmatic constraints are only applied to the individual components of Distinguished Name as defined in the Directory Documents, Part 2. Not all components of a DN will necessarily be understood by an implementation.

4 UPU agreements use only first 30 characters of ub-postal-string. This limitation should be observed whenever possible.

8 Conformance

Refer to clause 8 of Stable Agreements Version 4 as of June 14, 1991.

8.1 DUA Conformance

Refer to 8.1 of Stable Agreements Version 4 as of June 14, 1991.

8.2 DSA Conformance

Refer to 8.2 of Stable Agreements Version 4 as of June 14, 1991.

8.3 DSA Conformance Classes

Refer to 8.3 of Stable Agreements Version 4 as of June 14, 1991.

8.4 Authentication Conformance

Refer to 8.4 of Stable Agreements Version 4 as of June 14, 1991.

8.5 Directory Service Conformance

Refer to 8.5 of Stable Agreements Version 4 as of June 14, 1991.

8.6 The Directory Access Profile

Refer to 8.6 of Stable Agreements Version 4 as of June 14, 1991.

8.7 The Directory System Profile

Refer to 8.7 of Stable Agreements Version 4 as of June 14, 1991.

8.8 Digital Signature Protocol Conformance Profile

Refer to 8.8 of Stable Agreements Version 4 as of June 14, 1991.

8.9 Strong Authentication Protocol Conformance Profile

Refer to 8.9 of Stable Agreements Version 4 as of June 14, 1991.

8.10 Replication Conformance

A DSA implementing DISP shall conform to the basic conformance requirements for a DSA as defined in the Directory Documents, part 5, clause 9.2. However, it is not required for such a DSA to be either centralized or distributed as defined by 8.3 of this implementation agreement.

8.10.1 Shadowing Roles

All DSAs implementing DISP shall be capable of acting both as a shadow supplier and as a shadow consumer as defined in the Directory Documents, part 9, clause 3, and as such shall meet conformance requirements stated in part 5, 9.3 and 9.4.

8.10.2 Minimum Shadowing Requirements

Additionally, conformance to this profile requires a minimum as listed below:

a) support for the directoryShadowConsumerAC application context;

b) support for an UpdateMode whose mode choice includes a specification of schedulingParameters;

c) support for schedulingParameters specifications which specify a periodic strategy.

8.10.3 Support for Unit of Replication

This profile defines three classes regarding the level of refinement to be supported by a DSA in the definition of a unit of replication. A conforming implementation shall state which of the following Unit of Replication Conformance Classes it supports:

a) Unit of Replication Conformance Class 0 - Basic UoR is as follows:

1) A DSA conforming to this class is capable of shadowing a Unit of Replication with the following characteristics:

a) the area includes a subtree with a specified base component;

b) the knowledge includes a specified knowledgeType;

b) Unit of Replication Conformance Class 1 - Intermediate UoR is a follows:

1) A DSA conforming to this class has all the capabilities of a Basic UoR DSA. In addition, it is capable of shadowing a Unit of Replication with the following characteristics:

a) the area includes a subtree with a specified chop component;

b) the knowledge includes the extendedKnowledge elementwith value TRUE;

c) Unit of Replication Conformance Class 2 - Super Duper is as follows:

1) A DSA conforming to this class has all capabilities of an Intermediate UoR DSA. In addition, it shall be capable of shadowing a Unit of Replication with the following characteristics:

a) the attributes include AttributeSelection;

1) Furthermore, a DSA conforming to this class shall be capable of supporting overlapping replicated areas as described in the Directory Documents, part 9, 9.2.5.

8.11 Recommended Practices

8.11.1 APDU Size

In shadowing, an entire Unit of Replication is carred in one APDU. Since the size of such an APDU is application-specific, no pragmatic constraint has been specified in the Directory Documents or Implementation Agreements.

Some examples of APDU size implementors can expect would be useful. For instance, an entry size of 2000 octets and a Unit of Replication consisting of 2000 entries would result in a APDU of 4 Megabytes. It is recommended that DSA implementations be capable of supporting an APDU of at least this size. This example does not reflect entries which include large attributes, such as photographic images.

8.11.2 Duplicate Shadow Agreements

Administrators should not allow duplicate shadow agreements between DSAs. Duplicate shadow agreements are those which include the same consumer, supplier, and Unit of Replication.

However, in order not to restransmit an entire replicated area when a parameter of an agreement, such as frequency of update, is changed, duplicate agreements may exist temporarily.

Refer to the Directory Documents, part 9, 8.2.2.

8.11.3 Consistency Between Supplier and Consumer Information

After an updateShadowOperation, the standard does not guarantee consistency between the resulting shadowed information in the consumer DSA and the information in the replicated area in the supplier DSA, since changes may be made during assembly of the APDU containing the shadowed information.

If consistency between the supplier and consumer information is required, the contents of the replicated area in the supplier DSA must not be modified while the APDU is being assembled.

However, the shadowed information must be internally consistent. For example, while the shadowed information is being assembled, changing a distinguished name within the replicated area could lead to internal inconsistency.

8.12 Static Requirements

8.12.1 Reference Types

This Functional Standard requires conforming implementations to be able to hold and use reference types as summarised below (and clarified in clause 7.2.2):

HOLDING
REFERENCE TYPES NOTES CAPABILITY
Superior See note Non-first-level DSAs shall hold precisely one single SUperior Reference. A First-Level DSA does not hold any Superior Reference
Subordinate Mandatory
Non-specific Optional
Subordinate
Cross-reference Mandatory

8.12.2 Superior References and Root Contexts

8.12.2.1 The Root Context

The root context as held by a First Level DSA consists of the Root and a number of Subordinate References to Naming Contexts held (as master copies) by the DSA and by other First Level DSAs. It is replicated to each First Level DSA and comprises full knowledge of the naming contexts immediately subordinate to the root of the DIT. The means of this replication is not standardised.

8.12.2.2 First-Level DSAs

A DSA conformant to this Functional Standard acting as a First Level DSA shall be able to hold and use the Root Context and in addition shall hold as master (i.e. have administrative authority for) at least one Naming Context immediately subordinate to the root of the DIT. A DSA conforming to this Functional Standard is not, however, required to have the capability of being a First Level DSA.

During name resolution, a First-Level DSA shall act on a name whose first RDN corresponds neither to a locally held Naming Context nor to a Root Context Subordinate Reference as if that entry does not exist. In particular, it shall contain no entry whose name ‘s first RDN is unknown in this way.

> **NOTE -** The root context never contains any non-specific subordinate references and First Level DSAs should not hold such references in respect of the root context.

8.12.2.3 Return-Cross-References

The support of the "return-cross-references" facility, either as requester or as supplier, as defined in (ISO 9594-4 | CCITT X.518| clause 10.4.) is optional. (Viz. "returnCrossRefs" in Table 8.1 of A/712.)

8.13 Support of Application Contexts

All DSAs compliant with this Functional Standard shall support the DirectoryAccessAC or DirectorySystemAC or both.

A DSA which is to permit the dissemination of its knowledge references to one or more DSAs within another DMD (Directory Management Domain) is obliged to support the DirectorySystemAC, at least as a responder to chained operations. (Viz (ISO 9594-5 | CCITT X.519] Clause 9.2. 1a.)

> **NOTE -** If a DSA does not support the DirectorySystemAC, it will normally not be able to carry out simple authentication of a user whose entry is not held by that DSA (viz. clauses 7.3.3 and 8.9 of this functional Standard).

A DSA that can only act as an acceptor is not obliged to be able to generate a DSA-BIND (or DSA-UNBIND). It must, however, be able to invoke an A-ABORT on an incoming DSP association.

8.13.1 Referral Mode

A DSA compliant with this Functional Standard shall be able to use the referral mode of interaction, even if it only supports the DirectorySystemAC.

8.13.2 Chained Mode

A DSA may (but need not) use the chained mode of interaction. If it does, it shall support the DirectorySystemAC, with the capability of both invoking and performing operations.

A DSA may support the DirectorySystemAC without being obliged to use the chained mode of interaction; it then acts as the performer of chained operations, and must continue distributed operations (if necessary) by means of referrals.

8.13.3 DSAs Known by DSAs in Other DMDs

If a DSA is to be able to carry out simple authentication of a user whose entry is potentially held by some other DSA, the DSA must be able to invoke DSP "compare" or "read" operations to complete authentication by reference to other DSAs. Thus, unless this requirement can be met by some external means, all such DSAs shall support the DirectorySystemAC.

8.14 DSA-level Security

A DSA may (as a consequence of its security policy):

Refuse associations from any or particular DSAs

Refuse invokes on existing associations, for example based upon examination of the operation or its parameters (and responding with a Security- or Service-Error)

8.15 Aliases

DSAs conformant with this specification shall be able to carry out Name Resolution and search continuation with respect to Aliases held outside the DSA (as well as those held inside the DSA).

8.16 Authentication for DSA-Bind

8.16.1 Detection of Search Loop

A search operation may encounter a looping situation when the search encompasses "whole-subtree", and an alias is encountered which is superior to some other subtree that has been encountered during the search.

DSAs should be able to detect this situation. One possible method is by:

a) Maintaining a list of the base objects of searches initiated as a consequence of following aliases, including evidence of following aliases within the TraceInformation element;

Determining whether a new base object is superior, equal to or subordinate to any base object on this list.

A new base object which would cause a loop in this way should be discarded, but no protocol error arises. The circumstances should be logged, so that it may be reported to the appropriate Administrative Authority for rectification.

8.16.2 Generation of Trace Information

TraceInformation shall be ordered earliest information first.

A TraceInformation value carries forward a record of the DSAs which have been involved in the performance of an operation. It is used to detect the existence of, or avoid loops, which might arise from inconsistent knowledge or from the presence of alias loops in the DIT.

Each DSA which is propagating an operation to another adds a new item to the trace information. If the propagation of a Search operation involves the creation of a new Search, the trace information shall not be reset, but the full trace information for the overall search operation to the point where the new Search was generated shall be included in the new Search.

NOTE - See also Directory Implementor's Guide.

8.17 Integrity of Operation Arguments

For any operation argument in the abstract service (ReadArgument, etc.) that can (in principle) be signed, the content of any such argument shall always be passed on unchanged (subject only to variations in ASN.1 encoding which do not affect primitive values).

8.18 Referrals and Chaining

It is recommended that a DSA which has chained a request act upon any referrals which it receives, rather than returning them to the requestor if the "prefer-chaining" service control is present, unless prevented from doing so by administrative limitations or service policies.

However, if a DSA which is carrying out a List or a Search operation receives a set of unexplored Continuation References, it shall never pursue these if the result was signed (but was not collated by the DSA with other results), since this will result in duplication. If the result was unsigned, it may act on them (removing them from the consolidated result), or it may pass them back to the Invoker of the operation. The DSA can act on the references and remove them if collated.

8.18.1 Name-Error: "invalid-attribute-syntax"

This error shall only be generated when the DSA determines that there is an incompatibility in an AVA in that part of the name which it is expected to resolve.

If a multicasting DSA receives this error and the matched part of the name is equal to or longer than that indicated by the next RDN to be resolved, name resolution shall be taken as having progressed. The error shall be relayed.

If a chaining or multicasting DSA receives this error and the matched part of the name is not equal to or longer than that indicated by the next RDN to be resolved, the error indicates an incompatibility in schema between the DSA and the one to which chaining takes place. Multicasting may continue, and the error in that case may be ignored. A DSA, having received such an error during name resolution, nay but need not relay it.

8.18.2 Service-Error: "invalid-reference"

A DSA (having received a chained operation as a result of an NSSR) shall only generate a Service-Error: "invalid-reference" if it has determined that it does not hold an entry which is the immediate subordinate of the immediate superior of the next RDN to be resolved.

8.18.3 Illegal or Unsupported Attributes

A DSA may receive an AVA that is unsupported by the DSA. If the DSA is not required to act on it, or to store it within an entry, it shall handle it by passing it on by chaining, or providing a referral, and in particular shall not return an error response on its own initiative.

8.18.4 Matching Names in Trace Information

A DSA may be required to match names in TraceInformation; in the (unlikely) event of the attribute type of an AVA in such a name being unsupported by the DSA, the matching shall use an algorithm which reliably matches two names having the same primitive content.

8.19 Digital Signatures

DSAs supporting DSP shall accept signed chained-operations and their results intended for other DSAs; but they need not be capable of the evaluation of the signature. If they are capable of evaluating signed operations for local purposes, they shall be capable of evaluating both levels of signature (i.e. at both the operation and chained-operation levels).

DSAs are not obliged to be capable of evaluating digital signatures to be conformant to this Functional Standard.

9 Distributed Operations

Refer to clause 9 of Stable Agreements Version 4 as of June 14, 1991.

9.1 Referrals and Chaining

Refer to 9.1 of Stable Agreements Version 4 as of June 14, 1991.

9.2 Trace Information

Refer to 9.2 of Stable Agreements Version 4 as of June 14, 1991.

10 Underlying Services

Refer to clause 10 of Stable Agreements Version 4 as of June 14, 1991.

10.1 ROSE

Refer to 10.1 of Stable Agreements Version 4 as of June 14, 1991.

10.2 Session

Refer to 10.2 of Stable Agreements Version 4 as of June 14, 1991.

10.3 ACSE

Refer to 10.3 of Stable Agreements Version 4 as of June 14, 1991.

11 Access Control

11.1 Use of localQualifier in AuthenticationLevel

Editor's Note - for future study

11.2 Distributed Administrative Areas

11.3 ProtectedItem Granularity

11.4 UserClass Granularity

12 Test Considerations

Refer to clause 12 of Stable Agreements Version 4 as of June 14, 1991.

12.1 Major Elements of Architecture

Refer to 12.1 of Stable Agreements Version 4 as of June 14, 1991.

12.2 Search Operation

Refer to 12.2 of Stable Agreements Version 4 as of June 14, 1991.

13 Errors

Refer to clause 13 of Stable Agreements Version 4 as of June 14, 1991.

13.1 Permanent vs. Temporary Service Errors

Refer to 13.1 of Stable Agreements Version 4 as of June 14, 1991.

13.2 Guidelines for Error Handling

Refer to 13.2 of Stable Agreements Version 4 as of June 14, 1991.

13.2.1 Introduction

Refer to 13.2.1 of Stable Agreements Version 4 as of June 14, 1991.

13.2.2 Symptoms

Refer to 13.2.2 of Stable Agreements Version 4 as of June 14, 1991.

Editor's Note: In support of Basic Access Control, the following new error symptoms are proposed:

a) E_ENTRY_VISIBILITY: denotes the case where, for a given abstract operation, the Access Control Decision Function has denied visibility to a particular object during the visibility check specified in the figures of Annex C of Amendment 1 of ISO/IEC 9594-3;

b) E_ACCESS_AON: The Access Control Decision Function has denied permission to use the aliasedObjectName during an attempt to dereference;

c) E_ENTRY_LEVEL_ACCESS: Annex C of Amendment 1 to ISO/IEC 9594-3 (currently appears in 21N5953) specifies, for each abstract operation, that the Access Control Decision Function is used to check if the requestor is granted permission for the requested operation applied to a particular entry; this is a check of permissions on the ProtectedItem Entry. E_ENTRY_LEVEL_ACCESS denotes the situation where the ACDF has denied access to the ProtectedItem Entry.

13.2.3 Situations

Refer to 13.2.3 of Stable Agreements Version 4 as of June 14, 1991.

Editor's Note: In support of Basic Access Control, the following new error situation is proposed:

a) CHECK_ENTRY_VISIBILITY: The Access Control Decision Function has been envoked to determine if the requestor is granted permission to know that a particular entry exists.

13.2.4 Error Actions

Refer to 13.2.4 of Stable Agreements Version 4 as of June 14, 1991.

13.2.5 Reporting

Refer to 13.2.5 of Stable Agreements Version 4 as of June 14, 1991.

14 Specific Authentication Schemes

Editor's Note - The following text is proposed to replace the existing introductory paragraph for clause 14 of Stable Agreements Version 4 as of June 14, 1991.

This clause identifies authentication algorithms for use in Directory authentication. Informative text and ASN.1 definitions describing these algorithms appears in Part 12 (Security). Use of algorithms other than those cited in this clause or described in the Directory Documents is by bilateral agreement.

14.1 Specific Strong Authentication Schemes

Editor's Note - The following text is proposed to replace all text currently in 14.1 of Stable Agreements Version 4 as of June 14, 1991.

This subclause cites one alternative to the RSA digital signature scheme, the "ElGamal" digital signature scheme. Future contributions may result in other alternatives being added to this subclause.

Implementors may choose to provide digital signature capability based on RSA, ElGamal, or some other scheme appropriate for use in the OSI Directory environment.

It should be noted that both the use of RSA and ElGamal are governed by U.S. patent law.

14.1.1 ElGamal

Editor's Note - The following text is proposed to replace all text currently in 14.1.1 of Stable Agreements Version 4 as of June 14, 1991.

The "ElGamal" digital signature scheme was originally described by ElGamal in [ELGA85]. Part 12 (Security) of these agreements contains details on the use of ElGamal, including an informative description of the scheme using the notation defined in Part 8 of the Directory Documents and known constraints on algorithm parameters.

14.1.1.1 Background

Editor's Note - It is proposed that entire 14.1.1 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The text is proposed to move to Part 12 (Security). Deletion here is contingent on successful relocation to Part 12.

14.1.1.2 Digital Signature

Editor's Note - It is proposed that entire 14.1.1.2 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The text is proposed to move to Part 12 (Security). Deletion here is contingent on successful relocation to Part 12.

14.1.1.3 Verification

Editor's Note - It is proposed that entire 14.1.1.3 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The text is proposed to move to Part 12 (Security). Deletion here is contingent on successful relocation to Part 12.

14.1.1.4 Known Constraints on Parameters

Editor's Note - It is proposed that entire 14.1.1.4 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The text is proposed to move to Part 12 (Security). Deletion here is contingent on successful relocation to Part 12.

14.1.1.5 Note on subjectPublicKey

Editor's Note - It is proposed that entire 14.1.1.5 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The text is proposed to move to Part 12 (Security). Deletion here is contingent on successful relocation to Part 12.

14.1.2 One-Way Hash Functions

Editor's Note - It is proposed that the following text be inserted in 14.1.2 of Stable Agreements Version 4 as of June 14, 1991.

This subclause cites alternative one-way hash functions for use in Strong and Protected Simple Authentication. The Security SIG continues to investigate the security of additional one-way hash functions, and the Directory Services SIG will consider the applicability of these hash functions to Directory authentication.

A recent development in this area is the citation by the Security SIG of RSA MD4. In another recent development, the two-pass application of the SNEFRU algorithm was announced by Ralph Merkle to have been broken. Future study of MD4 and other contributions may result in other additions to this subclause.

At the present time, implementors may choose to provide one-way hash functionality based on MD2 or some other scheme aplpropriate for use in the OSI Directory environment.

14.1.2.1 SQUARE-MOD-N Algorithm

Refer to 14.1.2.1 of Stable Agreements Version 4 as of June 14, 1991.

14.1.2.2 MD2 Algorithm

Editor's Note - It is proposed that the second sentence of the existing text in 14.1.2.2 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The proposed deletion would reslult in a single sentence being left in 14.1.2.2 as follows: MD2 is a one-way hash function and is described in [RFC1115].

14.1.2.3 Study of Other One-Way Hash Functions

Editor's Note - It is proposed that 14.1.2.3 in Stable Agreements Version 4 as of June 14, 1991. be deleted. Note that the existing text has been incorporated in the proposed new text for 14.1.6.

14.1.2.4 Use of One-Way Hash Functions in Forming Signatures

Refer to 14.1.2.4 of Stable Agreements Version 4 as of June 14, 1991.

14.1.3 ASN.1 for Strong Authentication Algorithms

Editor's Note - It is proposed that the entire text currently in 14.1.3 of Stable Agreements Version 4 as of June 14, 1991. be replaced with the following text.

The Directory Services SIG has registered the use of MD2 as a hash algorithm, and the use of MD2 with RSA and MD2 with ElGamal as signature algorithms. The ASN.1 for the resulting object identifiers now appears alongside other security algotithm registrations in Part 12 (Security).

14.1.4 Note on the ENCRYPTED MACRO

Editor's Note - It is proposed that entire 14.1.4 of Stable Agreements Version 4 as of June 14, 1991. be deleted. The text is proposed to move to Part 12 (Security). Deletion here is contingent on successful relocation to Part 12.

14.2 Protected Simple Authentication

Refer to 14.2 of Stable Agreements Version 4 as of June 14, 1991.

14.3 Simple Authentication

Refer to 14.3 of Stable Agreements Version 4 as of June 14, 1991.

Annex A (normative)

Maintenance of Attribute Syntaxes

Refer to Annex A of Stable Agreements Version 4 as of June 14, 1991.

A.1 Introduction

Refer to A.1 of Stable Agreements Version 4 as of June 14, 1991.

A.2 General Rules

Refer to A.2 of Stable Agreements Version 4 as of June 14, 1991.

A.3 Checking Algorithms

Refer to A.3 of Stable Agreements Version 4 as of June 14, 1991.

A.3.1 distinguishedNameSyntax

Refer to A.3.1 of Stable Agreements Version 4 as of June 14, 1991.

A.3.2 integerSyntax

Refer to A.3.2 of Stable Agreements Version 4 as of June 14, 1991.

A.3.3 telephoneNumberSyntax

Refer to A.3.3 of Stable Agreements Version 4 as of June 14, 1991.

A.3.4 countryName

Refer to A.3.4 of Stable Agreements Version 4 as of June 14, 1991.

A.3.5 preferredDeliveryMethod

Refer to A.3.5 of Stable Agreements Version 4 as of June 14, 1991.

A.3.6 presentationAddress

Refer to A.3.6 of Stable Agreements Version 4 as of June 14, 1991.

A.4 Matching Algorithms

Refer to A.4 of Stable Agreements Version 4 as of June 14, 1991.

A.4.1 UTCTimeSyntax

Refer to A.4.1 of Stable Agreements Version 4 as of June 14, 1991.

A.4.2 distinguishedNameSyntax

Refer to A.4.2 of Stable Agreements Version 4 as of June 14, 1991.

A.4.3 caseIgnoreListSyntax

Refer to A.4.3 of Stable Agreements Version 4 as of June 14, 1991.

Annex B (informative)

Glossary

Refer to Annex B of Stable Agreements Version 4 as of June 14, 1991.

Annex C (informative)

Requirements for Distributed Operations

Refer to Annex C of Stable Agreements Version 4 as of June 14, 1991.

C.1 General Requirements

Refer to C.1 of Stable Agreements Version 4 as of June 14, 1991.

C.2 Protocol Support

Refer to C.2 of Stable Agreements Version 4 as of June 14, 1991.

C.2.1 Usage of ChainingArguments

Refer to C.2.1 of Stable Agreements Version 4 as of June 14, 1991.

C.2.2 Usage of ChainingResults

Refer to C.2.2 of Stable Agreements Version 4 as of June 14, 1991.

Annex D (informative)

Guidelines for Applications Using the Directory

Refer to Annex D of Stable Agreements Version 4 as of June 14, 1991.

D.1 Tutorial

Refer to D.1 of Stable Agreements Version 4 as of June 14, 1991.

D.1.1 Overview

Refer to D.1.1 of Stable Agreements Version 4 as of June 14, 1991.

D.1.2 Use of the Directory Schema

Refer to D.1.2 of Stable Agreements Version 4 as of June 14, 1991.

D.1.2.1 Use of Existing Object Classes

Refer to D.1.2.1 of Stable Agreements Version 4 as of June 14, 1991.

D.1.2.2 Kinds of Object Classes

Refer to D.1.2.2 of Stable Agreements Version 4 as of June 14, 1991.

D.1.2.3 Use of Unregistered Object Classes

Refer to D.1.2.3 of Stable Agreements Version 4 as of June 14, 1991.

D.1.2.4 Side Effects of Creating Unregistered Object Classes

Refer to D.1.2.4 of Stable Agreements Version 4 as of June 14, 1991.

D.2 Creation of New Object Classes

Refer to D.2 of Stable Agreements Version 4 as of June 14, 1991.

D.2.1 Creation of New Subclasses

Refer to D.2.1 of Stable Agreements Version 4 as of June 14, 1991.

D.2.2 Creation of New Attributes

Refer to D.2.2 of Stable Agreements Version 4 as of June 14, 1991.

D.3 DIT Structure Rules

Refer to D.3 of Stable Agreements Version 4 as of June 14, 1991.

D.4 Use of AETITLE

Applications wishing to make use of the AETitle field to access applicationEntity objects in the Directory are referred to Amendment 1 to ISO8650 for guidance on the purpose and appropriate useage of the AETitle field. In particular, implementors should be aware that:

a) AETitle should be used to uniquely distinguish individual application entities. It is inappropriate for applications to define a fixed AETitle to apply to all its instantiations;

b) The Directory does not perform name resolution on an object identifier (e.g., AETitle name form 2). The Directory does <u>not</u> support lookup based on OID, and AETitle name form 2 does <u>not</u> constitute a Directory Distinguished Name.

Annex E (informative)

Template for an Application Specific Profile for Use of the Directory

Refer to Annex E of Stable Agreements Version 4 as of June 14, 1991.

Annex F (informative)

Bibliography

Refer to Annex F of Stable Agreements Version 4 as of June 14, 1991.