INTERNATIONAL STANDARD

# ISO/IEC 9594-4

Ninth edition
2020-11

# Information technology — Open systems interconnection —

## Part 4: The Directory: Procedures for distributed operation

Reference number
ISO/IEC 9594-4:2020(E)

Documents
Document Sharing Hub

© ISO/IEC 2020

Full document details: ISO_IEC 9594-4_2020

**COPYRIGHT PROTECTED DOCUMENT**

© ISO/IEC 2020

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland



© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO_IEC 9594-4_2020

# Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

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

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

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

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

This document was prepared by ITU-T as ITU-T X.518 (10/2019) and drafted in accordance with its editorial rules, in collaboration with Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*.

This ninth edition cancels and replaces the eighth edition (ISO/IEC 9594-4:2017), which has been technically revised.

A list of all parts in the ISO/IEC 9594 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

© ISO/IEC 2020 – All rights reserved


Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

Documents
Document Sharing Hub

Full document details: ISO_IEC 9594-4_2020

**CONTENTS**


| | | Page |
|---|---|---|
| 1 | Scope | 1 |
| 2 | References | 1 |
| | 2.1 Normative references | 1 |
| | 2.2 Non-normative reference | 2 |
| 3 | Definitions | 2 |
| | 3.1 Basic Directory definitions | 2 |
| | 3.2 Directory model definitions | 2 |
| | 3.3 DSA information model definitions | 2 |
| | 3.4 Abstract service definitions | 3 |
| | 3.5 Protocol definitions | 3 |
| | 3.6 Directory replication definitions | 3 |
| | 3.7 Distributed operation definitions | 3 |
| 4 | Abbreviations | 5 |
| 5 | Conventions | 5 |
| 6 | Overview | 6 |
| 7 | Distributed Directory system model | 7 |
| 8 | DSA interactions model | 8 |
| | 8.1 Decomposition of a request | 8 |
| | 8.2 Uni-chaining | 8 |
| | 8.3 Multi-chaining | 9 |
| | 8.4 Referral | 10 |
| | 8.5 Mode determination | 11 |
| 9 | Overview of DSA abstract service | 12 |
| 10 | Information types | 12 |
| | 10.1 Introduction | 12 |
| | 10.2 Information types defined elsewhere | 12 |
| | 10.3 Chaining arguments | 13 |
| | 10.4 Chaining results | 15 |
| | 10.5 Operation progress | 16 |
| | 10.6 Trace information | 17 |
| | 10.7 Reference type | 17 |
| | 10.8 Access point information | 17 |
| | 10.9 DIT bridge knowledge. | 18 |
| | 10.10 Exclusions | 19 |
| | 10.11 Continuation reference | 19 |
| 11 | Bind and Unbind | 20 |
| | 11.1 DSA Bind | 20 |
| | 11.2 DSA Unbind | 21 |
| 12 | Chained operations | 21 |
| | 12.1 Chained operations | 22 |
| | 12.2 Chained Abandon operation | 22 |
| | 12.3 Chained operations and protocol version | 23 |
| 13 | Chained errors | 23 |
| | 13.1 Introduction | 23 |
| | 13.2 DSA referral | 23 |
| 14 | Introduction | 24 |
| | 14.1 Scope and limits | 24 |
| | 14.2 Conformance | 24 |
| | 14.3 Conceptual model | 24 |
| | 14.4 Individual and cooperative operation of DSAs | 24 |


© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

Page

14.5 Cooperative agreements between DSAs ... 25

15 Distributed Directory behaviour ... 25
15.1 Cooperative fulfilment of operations ... 25
15.2 Phases of operation processing ... 25
15.3 Managing Distributed Operations ... 26
15.4 Loop handling ... 27
15.5 Other considerations for distributed operation ... 28
15.6 Authentication of Distributed operations ... 29

16 The Operation Dispatcher ... 30
16.1 General concepts ... 30
16.2 Procedures of the Operation Dispatcher ... 35
16.3 Overview of procedures ... 36

17 Request Validation procedure ... 37
17.1 Introduction ... 37
17.2 Procedure parameters ... 38
17.3 Procedure definition ... 39

18 Name Resolution procedure ... 42
18.1 Introduction ... 42
18.2 Find DSE procedure parameters ... 42
18.3 Procedures ... 43

19 Operation evaluation ... 52
19.1 Modification procedures ... 53
19.2 Single entry interrogation procedure ... 60
19.3 Multiple entry interrogation procedure ... 60

20 Continuation Reference procedures ... 74
20.1 Chaining strategy in the presence of shadowing ... 74
20.2 Issuing chained subrequests to a remote DSA or LDAP server ... 76
20.3 Procedures' parameters ... 76
20.4 Definition of the procedures ... 77
20.5 Abandon procedures ... 86
20.6 DAP request to LDAP request procedure ... 88
20.7 LDAP result to DAP reply procedure ... 92

21 Results Merging procedure ... 94

22 Procedures for distributed authentication ... 96
22.1 Requester authentication ... 96
22.2 Results authentication ... 97

23 Knowledge administration overview ... 98
23.1 Maintenance of knowledge references ... 98
23.2 Requesting cross reference ... 99
23.3 Knowledge inconsistencies ... 100

24 Hierarchical operational bindings ... 101
24.1 Operational binding type characteristics ... 101
24.2 Operational binding information object Class definition ... 103
24.3 DSA procedures for hierarchical operational binding management ... 104
24.4 Procedures for operations ... 107
24.5 Use of application contexts ... 108

25 Non-specific hierarchical operational binding ... 108
25.1 Operational binding type characteristics ... 108
25.2 Operational binding information object class definition ... 109
25.3 DSA procedures for non-specific hierarchical operational binding management ... 109
25.4 Procedures for operations ... 111
25.5 Use of application contexts ... 111


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

*Page*

Annex A – ASN.1 for Distributed Operations ..... 112
Annex B – Specification of hierarchical and non-specific hierarchical operational binding types ..... 116
Annex C – Example of distributed name resolution ..... 118
Annex D – Distributed use of authentication ..... 120
D.1 Summary ..... 120
D.2 Distributed protection model ..... 120
D.3 Signed chained operations ..... 120
Annex E – Knowledge maintenance example ..... 122
Annex F – Amendments and corrigenda ..... 125

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

## Introduction

This Recommendation | International Standard, together with other Recommendations | International Standards, have been produced to facilitate the interconnection of information processing systems to provide directory services. A set of such systems, together with the directory information that they hold, can be viewed as an integrated whole, called the *Directory*. The information held by the Directory, collectively known as the Directory information base (DIB), is typically used to facilitate communication between, with or about objects such as application entities, people, terminals and distribution lists.

The Directory plays a significant role in Open Systems Interconnection, whose aim is to allow, with a minimum of technical agreement outside of the interconnection standards themselves, the interconnection of information processing systems:

- from different manufacturers;
- under different managements;
- of different levels of complexity; and
- of different ages.

This Recommendation | International Standard specifies the procedures by which the distributed components of the Directory interwork in order to provide a consistent service to its users.

This Recommendation | International Standard provides the foundation frameworks upon which industry profiles can be defined by other standards groups and industry forums. Many of the features defined as optional in these frameworks may be mandated for use in certain environments through profiles. This ninth edition technically revises and enhances the eighth edition of this Recommendation | International Standard.

This nineth edition specifies versions 1 and 2 of the Directory protocols.

Rec. ITU-T X.511 (1993) | ISO/IEC 9594-3 (1995), Rec. ITU-T X.518 (1993) | ISO/IEC 9594-4 (1995) and Rec. ITU-T X.519 (1993) | ISO/IEC 9594-5 (1995) and their previous edition specified only version 1. Most of the services and protocols specified in this edition are designed to function under version 1. However, some enhanced services and protocols, e.g., signed errors, will not function unless all Directory entities involved in the operation have negotiated version 2. Whichever version has been negotiated, differences between the services and between the protocols defined in the nine editions, except for those specifically assigned to version 2, are accommodated using the rules of extensibility defined in Rec. ITU-T X.519 | ISO/IEC 9594-5.

Annex A, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for directory distributed operations.

Annex B, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module providing definitions for hierarchical operational bindings.

Annex C, which is not an integral part of this Recommendation | International Standard, describes an example of distributed name resolution.

Annex D, which is not an integral part of this Recommendation | International Standard, describes authentication in the distributed operations environment.

Annex E, which is not an integral part of this Recommendation | International Standard, illustrates knowledge maintenance.

Annex F, which is not an integral part of this Recommendation | International Standard, lists the amendments and defect reports that have been incorporated to form this edition of this Recommendation | International Standard.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

**INTERNATIONAL STANDARD ISO/IEC 9594-4**
**RECOMMENDATION ITU-T X.518**

# Information technology – Open Systems Interconnection – The Directory: Procedures for distributed operation

SECTION 1 – GENERAL

## 1 Scope

This Recommendation | International Standard specifies the behaviour of DSAs taking part in a distributed directory consisting of multiple Directory systems agents (DSAs) and/or LDAP servers with at least one DSA. The allowed behaviour has been designed to ensure a consistent service given a wide distribution of the DIB across a distributed directory. Only the behaviour of DSAs taking part in a distributed directory is specified. The behaviour of LDAP servers are specified in relevant LDAP specifications. There are no special requirements on an LDAP server beyond those given by the LDAP specifications.

The Directory is not intended to be a general purpose database system, although it may be built on such systems. It is assumed that there is a considerably higher frequency of queries than of updates.

## 2 References

### 2.1 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

#### 2.1.1 Identical Recommendations | International Standards

- Recommendation ITU-T X.500 (2019 | ISO/IEC 9594-1:2020, *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services*.
- Recommendation ITU-T X.501 (2019) | ISO/IEC 9594-2:2020, *Information technology – Open Systems Interconnection – The Directory: Models*.
- Recommendation ITU-T X.509 (2019) | ISO/IEC 9594-8:2020, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks*.
- Recommendation ITU-T X.511 (2019) | ISO/IEC 9594-3:2020, *Information technology – Open Systems Interconnection – The Directory: Abstract service definition*.
- Recommendation ITU-T X.519 (2019) | ISO/IEC 9594-5:2020, *Information technology – Open Systems Interconnection – The Directory: Protocol specifications*.
- Recommendation ITU-T X.520 (2019) | ISO/IEC 9594-6:2020, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types*.
- Recommendation ITU-T X.521 (2019) | ISO/IEC 9594-7:2020, *Information technology – Open Systems Interconnection – The Directory: Selected object classes*.
- Recommendation ITU-T X.525 (2019) | ISO/IEC 9594-9:2020, *Information technology – Open Systems Interconnection – The Directory: Replication*.
- Recommendation ITU-T X.680 (2015) | ISO/IEC 8824-1:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

### 2.1.2 Other references

- Recommendation ITU-T X.681 (2015) | ISO/IEC 8824-2:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.
- Recommendation ITU-T X.682 (2015) | ISO/IEC 8824-3:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification*.
- Recommendation ITU-T X.683 (2015) | ISO/IEC 8824-4:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.
- IETF RFC 4511 (2006), *Lightweight Directory Access Protocol (LDAP): The Protocol*.
- IETF RFC 4514 (2006), *Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names*.

## 2.2 Non-normative reference

- IETF RFC 4510 (2006), *Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map*.

# 3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply:

## 3.1 Basic Directory definitions

The following terms are defined in Rec. ITU-T X.500 | ISO/IEC 9594-1:

a) *(the) Directory*;

b) *Directory Information Base*.

## 3.2 Directory model definitions

The following terms are defined in Rec. ITU-T X.501 | ISO/IEC 9594-2:

a) *access point*;

b) *alias*;

c) *Directory Information Tree* ;

d) *Directory System Agent (DSA)*;

e) *Directory User Agent (DUA)*;

f) *distinguished name*;

g) *relative distinguished name*.

## 3.3 DSA information model definitions

The following terms are defined in Rec. ITU-T X.501 | ISO/IEC 9594-2:

a) *category*;

b) *commonly usable*;

c) *context prefix*;

d) *cross reference*;

e) *DIB fragment*;

f) *DSA information tree*;

g) *DSA-Specific Entry (DSE)*;

h) *DSE type*;

i) *immediate superior reference*;

j) *knowledge information*;

k) *knowledge reference category*;

l) *knowledge reference type*;


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

m) *naming context*;

n) *non-specific knowledge*;

o) *non-specific subordinate reference*;

p) *operational attribute*;

q) *reference path*;

r) *specific knowledge*;

s) *subordinate reference*;

t) *superior reference*.

## 3.4 Abstract service definitions

The following terms are defined in Rec. ITU-T X.511 | ISO/IEC 9594-3:

a) *reply*;

b) *request*;

c) *requester*.

## 3.5 Protocol definitions

The following terms are defined in Rec. ITU-T X.519 | ISO/IEC 9594-5:

a) *application-association*;

b) *application-entity-title*.

## 3.6 Directory replication definitions

The following terms are defined in Rec. ITU-T X.525 | ISO/IEC 9594-9:

a) *attribute completeness*;

b) *shadowing operational binding*;

c) *subordinate completeness*;

d) *unit of replication*.

## 3.7 Distributed operation definitions

The following terms are defined in this Recommendation | International Standard:

**3.7.1 base object**: The object or alias entry that is the target for an operation as issued by the requester.

**3.7.2 bound DSA**: The DSA to which the requesting DUA or LDAP client has bound, by having performed a Bind operation with that DSA.

**3.7.3 bound-DSA paged results**: The paging is performed entirely by the DSA to which the DUA is bound.

NOTE – This is the only mode of paging supported by systems conforming to Rec. ITU-T X.518 (2001) | ISO/IEC 9594-4:2001 or prior editions.

**3.7.4 chaining**: The generic term for uni-chaining or multi-chaining.

**3.7.5 context prefix information**: Operational and user information supplied by the superior DSA to the subordinate DSA in an RHOB regarding DIT vertices superior to the subordinate context prefix.

**3.7.6 directory server**: A DSA or an LDAP server.

**3.7.7 distributed directory**: An interconnected set of directory servers where at least one directory server shall be a DSA.

**3.7.8 distributed name resolution**: The process by which name resolution starts in a DSA and continues in one or more Directory servers.

**3.7.9 DSP paged results**: The DSP protocol provisions when a performing DSA is different from a bound DSA, whereby paged results by the initial performer are accomplished.

**3.7.10 error**: Information sent from the performer to the requester conveying a negative outcome of a previously received request.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

**3.7.11 hard error**: A definite error which indicates that the operation cannot currently be performed without external intervention.

**3.7.12 hierarchical operational binding (HOB)**: Relationship between two master DSAs holding naming contexts, one of which is immediately subordinate to the other, in which the superior DSA holds a subordinate reference to the subordinate DSA.

**3.7.13 initial performer**: The first DSA or LDAP server to start performing on an operation, i.e., the first DSA or LDAP server to enter the evaluation phase of the operation.

**3.7.14 LDAP requester**: A DSA that has the ability to access an LDAP server by using the LDAP protocol.

**3.7.15 modification operations**: These are the Directory Modify operations, i.e., Modify Entry, Add Entry, Remove Entry, Modify DN, Change Password and Administer Password operations.

**3.7.16 multi-chaining**: A mode of interaction in which a DSA processing a request itself sends multiple requests either in parallel or sequentially to a set of other DSAs.

**3.7.17 multiple entry interrogation operations**: These are the Directory Search operations, i.e., List and Search operations.

**3.7.18 name resolution**: The process of locating an entry by sequentially matching each RDN in a purported name to a vertex of the DIT.

**3.7.19 non-specific hierarchical operational binding (NHOB)**: Relationship between two master DSAs holding naming contexts, one of which is immediately subordinate to the other, in which the superior DSA holds a non-specific subordinate reference to the subordinate DSA.

**3.7.20 NSSR decomposition**: Decomposition of non-specific knowledge references into subrequests for other DSAs to pursue; these subrequests may be either chained to these DSAs by the DSA performing the decomposition, or a continuation reference identifying the DSAs may be returned to the requester for it to pursue, or the decomposing DSA may pursue some of the subrequests, leaving others unexplored for the requester to pursue.

**3.7.21 operation progress**: A set of values which denotes the extent to which name resolution has taken place.

**3.7.22 paging**: A `search` or `list` result is returned piecewise in the form of one or more pages that are comprised by a limited number of entries.

**3.7.23 performer**: DSA receiving a request (i.e., to perform an operation).

NOTE – The performer is also the initial performer except possibly for operations that involve more than one DSA for their evaluation.

**3.7.24 procedure**: An (informal) specification of how a DSA maps a given set of input arguments and its DSA information tree into a result.

NOTE – Input arguments and results may correspond to information received in a requested operation and information sent in a reply, or they may represent intermediate stages in the computation of a reply from a requested operation. In clause 14.2, the former variety of input arguments and results are termed external.

**3.7.25 relevant hierarchical operational binding (RHOB)**: Either an HOB or an NHOB, depending on the context.

**3.7.26 referral**: An outcome which can be returned by a DSA or LDAP server which cannot perform an operation itself, and which identifies one or more other DSAs or LDAP servers more able to perform the operation.

**3.7.27 request decomposition**: Decomposition by a DSA of a request into subrequests for other Directory servers to pursue; these subrequests may be either chained to these Directory servers by the DSA performing the decomposition, or continuation references identifying the Directory servers may be returned to the requester for it to pursue, or the decomposing DSA may pursue some of the subrequests, leaving others unexplored for the requester to pursue.

**3.7.28 single entry interrogation operations**: These are the Directory Read operations, i.e., Read and Compare operations.

**3.7.29 soft error**: An error which may be transient, or which may indicate a localized problem, in which case the use of a different knowledge reference or access point may enable a result or hard error to be obtained.

**3.7.30 subordinate DSA**: Of the two DSAs sharing an HOB or an NHOB, the DSA holding the subordinate naming context.

**3.7.31 subrequest**: A request generated by request decomposition.

**3.7.32 superior DSA**: Of the two DSAs sharing an HOB or an NHOB, the DSA holding the superior naming context.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

**3.7.33 superior, subordinate DSA**: Two master DSAs holding naming contexts, one of which is immediately subordinate to the other; the relationship between the two DSAs is managed explicitly via an HOB (or NHOB), or exists implicitly by virtue of the superior DSA holding a subordinate (or non-specific subordinate) reference to the subordinate DSA.

**3.7.34 target object name**: The name of an entry either to which the operation is to be directed at a particular stage of name resolution, or which is involved in the evaluation of the operation.

**3.7.35 uni-chaining**: A mode of interaction optionally used by a DSA which cannot perform an operation itself. The DSA *chains* by invoking an operation of another DSA or LDAP server and then relaying the outcome to the original requester.

## 4 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply:

| | |
|---|---|
| ASN.1 | Abstract Syntax Notation One |
| DAP | Directory Access Protocol |
| DIB | Directory Information Base |
| DISP | Directory Information Shadowing Protocol |
| DMD | Directory Management Domain |
| DOP | Directory Operational Binding Management Protocol |
| DSA | Directory System Agent |
| DSE | DSA-Specific Entry |
| DSP | Directory System Protocol |
| DUA | Directory User Agent |
| HOB | Hierarchical Operational Binding |
| LDAP | Lightweight Directory Access Protocol |
| NHOB | Non-specific Hierarchical Operational Binding |
| NSSR | Non-Specific Subordinate Reference |
| RHOB | Relevant Hierarchical Operational Binding |

## 5 Conventions

The term "Directory Specification" (as in "this Directory Specification") shall be taken to mean Rec. ITU-T X.518 | ISO/IEC 9594-4. The term "Directory Specifications" shall be taken to mean the Rec. ITU-T X.500-series | ISO/IEC 9594-1, Rec. ITU-T X.501 | IO/IEC 9594-2, Rec. ITU-T X.511 | ISO/IEC 9594-3, Rec. ITU-T X.518 | ISO/IEC 9594-4, Rec. ITU-T X.519 | ISO/IEC 9594-5, Rec. ITU-T X.520 | ISO/IEC 9594-6, Rec ITU-T X.521 | ISO/IEC 9594-7 and Rec. ITU-T X.525 | ISO/IEC 9594-9.

If an International Standard or ITU-T Recommendation is referenced within normal text without an indication of the edition, the edition shall be taken to be the latest one as specified in the normative references clause.

Prior to year 2020, the parts making up the Directory Specifications progressed together and can therefore collectively be identified as the Directory Specifications of a specific edition using the format: Rec. ITU-T X.5** (yyyy) | ISO/IEC 9594-*:yyyy (e.g.; Rec ITU-T X.5** (1993) | ISO/IEC 9594-*:1995).

This Directory Specification makes extensive use of Abstract Syntax Notation One (ASN.1) for the formal specification of data types and values, as it is specified in Rec. ITU-T X.680 | ISO/IEC 8824-1, ITU-T X.681 (2015) | ISO/IEC 8824-2, ITU-T X.682 (2015) | ISO/IEC 8824-3, ITU-T X.683 (2015) | ISO/IEC 8824-4 and Rec. ITU-T X.690 | ISO/IEC 8825-1.

This Directory Specification presents ASN.1 notation in the bold Courier New typeface. When ASN.1 types and values are referenced in normal text, they are differentiated from normal text by presenting them in the bold Courier New typeface. The names of procedures, typically referenced when specifying the semantics of processing, are differentiated from normal text by displaying them in bold Times New Roman. Access control permissions are presented in italicized Times New Roman.

If the items in a list are numbered (as opposed to using "–" or letters), then the items shall be considered steps in a procedure.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

SECTION 2 – OVERVIEW

## 6 Overview

The Directory abstract service allows the interrogation, retrieval and modification of Directory information in the DIB. This service is specified in in Rec. ITU-T X.511 | ISO/IEC 9594-3. Similarly, the lightweight directory access protocol (LDAP) allows the interrogation, retrieval and modification of Directory information in the DIB. This protocol and the services it enables are specified in IETF RFC 4511.

The abstract service as specified in Rec. ITU-T X.511 | ISO/IEC 9594-3 does not address the specification of Directory system agents (DSA) within which the DIB is stored and managed, and through which the service is provided. Furthermore, it does not consider whether the DIB is centralized, i.e., contained within a single DSA, or distributed over a DSA and a number of additional DSAs and/or LDAP servers. Directory server is the common name for a DSA or an LDAP server. Consequently, the requirements for DSAs to have knowledge of, navigate to and cooperate with other DSAs and or LDAP servers, in order to support the abstract service in a distributed environment is also not covered by the abstract service specification.

This Directory Specification specifies how a set of one or more DSAs and zero or more LDAP servers collectively constitute the distributed directory service.

In addition, this Directory Specification specifies the permissible ways in which the DIB may be distributed over one or more DSAs and zero or more LDAP servers. For the limiting case where the DIB is contained within a single DSA, the Directory is in fact centralized; for the case where the DIB is distributed over two or more DSAs, knowledge and navigation mechanisms are specified which ensure that the whole of the DIB is potentially accessible from all DSAs that hold constituent entries.

Portions of the DIB may also be replicated in multiple DSAs. The protocols described in this Directory Specification allow the use of replicated information to improve the availability, performance and efficiency of the distributed directory service. The use of replicated information is, to some extent, under the user's control, through the use of service control options. The procedures described in this Directory Specification also indicate some of the opportunities for design optimizations when using the replicated information.

Additionally, request handling interactions are specified that enable particular operational characteristics of the Directory to be controlled by its users. In particular, the user has control over whether a DSA, responding to a directory inquiry pertaining to information held in other directory server(s), has the option of interrogating the other DSA(s) directly (chaining) or, whether it should respond with information about other directory server(s) which could further progress the inquiry (referral).

Generally, the decision by a DSA to chain or refer is determined by the service controls set by the user, and by the DSA's own administrative, operational or technical circumstances.

Recognizing that, in general, the Directory will be distributed, and that directory inquiries will be satisfied by an arbitrary number of cooperating DSAs which may arbitrarily chain or refer according to the above criteria, this Directory Specification specifies the appropriate procedures to be effected by DSAs in responding to distributed directory inquiries. These procedures will ensure that users of the distributed Directory service perceive it to be both user-friendly and consistent.

NOTE – Although an LDAP server may participate in a distributed operation, it is not aware of this cooperation.


© ISO/IEC 2020 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

SECTION 3 – DISTRIBUTED DIRECTORY MODELS

## 7 Distributed Directory system model

The Directory abstract service, as defined in Rec. ITU-T X.511 | ISO/IEC 9594-3, models the Directory as an entity which provides a set of directory services to its users. Users of the Directory access its services through an access point.

Figure 1 illustrates the distributed directory model which will be used as the basis for specifying the distributed aspects of the directory. It illustrates the Directory as comprising a set of one or more DSAs and zero or more LDAP servers.

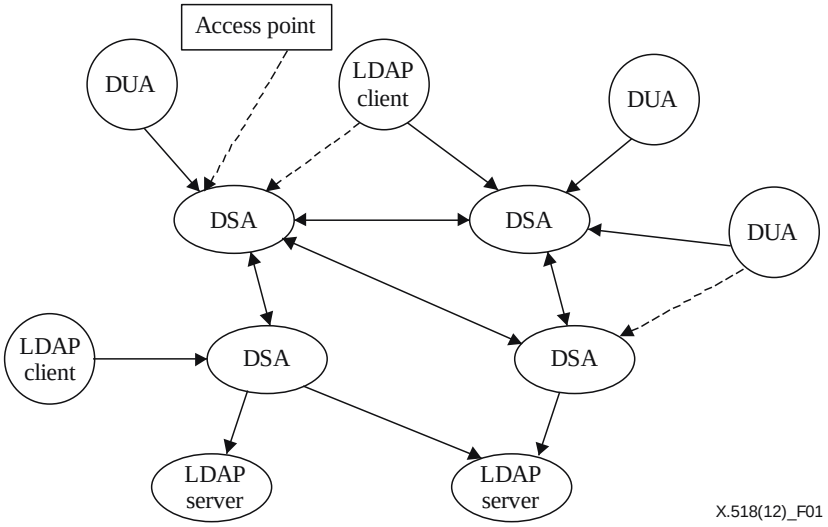


**Figure 1 – The distributed directory model**

DSAs are specified in detail in the subsequent clauses of this Directory Specification. This clause merely states a number of their characteristics, in order to serve as an introduction and to establish the relationship between this Directory Specification and the other Directory Specifications.

DSAs are defined in order that distribution of the DIB can be accommodated and that a number of physically distributed DSAs and LDAP servers can interact in a prescribed, cooperative manner to provide directory services to the users of the directory (DUAs and/or LDAP clients).

Figure 1 illustrates the relationship between the directory abstract service and the DSA abstract service. The directory abstract service defined in Rec. ITU-T X.511 | ISO/IEC 9594-3 is provided through a number of Directory operations. To realize this service, the DSAs and LDAP servers that comprise the Directory interact with one another. The nature of this interaction is defined in terms of the service that one DSA may provide to another DSA, the DSA abstract service. In addition, a DSA may interact with an LDAP server using the LDAP protocol as defined by IETF RFC 4511. When doing this, the DSA is called an *LDAP requester* and the DSA abstract service does not apply for this type of interaction. A DSA that is directly bound to a DUA or LDAP client is called the *bound DSA* (for that DUA or LDAP client).

As indicated in Figure 1, each of two interacting DSAs may provide a DSA abstract service to the other DSA. However, an LDAP server is not able to send requests to a DSA or to another LDAP server. LDAP servers are therefore always at the edge of the infrastructure.

The DSA abstract service is provided through a number of operations, termed chained operations, each having a counterpart in the Directory abstract service. Thus, a given operation in the directory abstract service, e.g., Read, may require that the DSA providing the service interact with one or more other DSAs using chained operations, e.g., Chained Read.

A DUA or LDAP client can only access the Directory by interacting with a DSA.

NOTE – An LDAP client interaction with an LDAP server is specified by IETF RFC 4510 and is outside the scope of these Directory Specifications.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

# 8 DSA interactions model

A basic characteristic of the Directory is that, given a distributed DIB, a requester should potentially be able to have any service request satisfied (subject to security, access control, service restrictions and administrative policies) irrespective of the access point at which the request originates. In accommodating this requirement, it is necessary that any DSA involved in satisfying a particular service request has some knowledge (as specified in Rec. ITU-T X.501 | ISO/IEC 9594-2) of where the requested information is located and either return this knowledge to the requester or attempt to have the request satisfied on its behalf. (The requester may be a DUA, an LDAP client or another DSA (possibly on behalf of bound LDAP clients). In the latter case, both DSAs shall support the DSP.

Three modes of DSA interaction are defined to meet these requirements, namely "uni-chaining", "multi-chaining", and "referral". Throughout the remainder of this Directory Specification, the generic term chaining is used to refer to uni-chaining and/or multi-chaining as appropriate to the particular situation. "Chaining" refers to the attempt by a DSA to satisfy a request by sending one or more chained operations to other DSAs; "referral", refers to the return of knowledge information to the requester, which may then itself interact with the DSA(s) identified in the knowledge information.

LDAP servers may be components of a Directory. A DSA may chain requests to an LDAP server using the LDAP protocol as specified in IETF RFC 4511. A DSA able to chain requests in this way is called an *LDAP requester*.

Uni-chaining or a referral interaction may result from a single request. Alternatively, the request may be decomposed into several subrequests prior to the interaction. Multi-chaining or referral interactions, or a mixture of the two, may result from a decomposed request. Two types of decomposition are defined: NSSR decomposition and request decomposition.

## 8.1 Decomposition of a request

### 8.1.1 NSSR decomposition

NSSR decomposition is the process of preparing (semantically) identical requests ready for transfer (either sequentially or in parallel) to several subordinate directory servers as a result of encountering an NSSR during name resolution. Non-specific subordinate references do not hold the RDNs of the referenced subordinate naming contexts, so the referencing DSA is unable to tell which subordinate directory server holds which subordinate naming context(s). During name resolution, a DSA encountering NSSRs shall send a semantically identical request to each subordinate directory server (in the absence of shadowing). This may be done sequentially or in parallel. Typically, only one directory server will be able to continue with name resolution; the other DSAs will return a **`serviceError`** with problem **`unableToProceed`**, while other LDAP servers will return **`noSuchObject`**. In certain (rare) circumstances, it is possible that more than one DSA will continue with name resolution, giving rise to duplicate results.

### 8.1.2 Request decomposition

Request decomposition, the other form of decomposing a request, is a process performed internally by a DSA prior to communication with one or more other directory servers. A request is decomposed into several, possibly different, subrequests such that each of the subrequests accomplishes a part of the original task. Request decomposition can be used only during operation evaluation of a List, Search or LDAP Search operation. After request decomposition, each of the subrequests may then be chained to other directory servers to continue the task, or a partial result (an embedded referral) may be returned to the requester. An example of the same subrequest being generated to different directory servers is when an entry has subordinate references and/or NSSRs that together reference more than one directory server. An example of different subrequests being generated to the same or different directory servers is when two different entries are encountered during a Search (subtree), and each has a subordinate reference.

## 8.2 Uni-chaining

This mode of interaction (depicted in Figure 2) may be used by one DSA to pass on a request to another directory server when the former has knowledge about naming contexts held by the latter. Uni-chaining may be used to contact a single DSA or LDAP server pointed to in a cross reference or a subordinate reference. In addition, a superior reference, supplier reference or a master reference may be used to contact a single DSA.

NOTE – In Figure 2, the order of interactions is defined by the numbers associated with the interaction lines.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

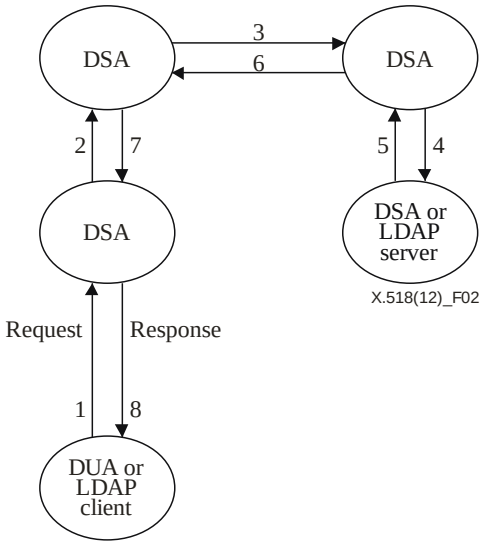


**Figure 2 – Uni-chaining mode**

## 8.3 Multi-chaining

This mode of interaction is used by a DSA for transferring several outgoing requests which have resulted from one incoming request, as a result of either request decomposition or NSSR decomposition.

### 8.3.1 Parallel multi-chaining

With parallel multi-chaining, the DSA transfers several outgoing requests simultaneously (see Figure 3a). Whilst parallel multi-chaining may give improved performance, it may under certain circumstances, e.g., in the presence of shadowing, cause duplicate results to be received.

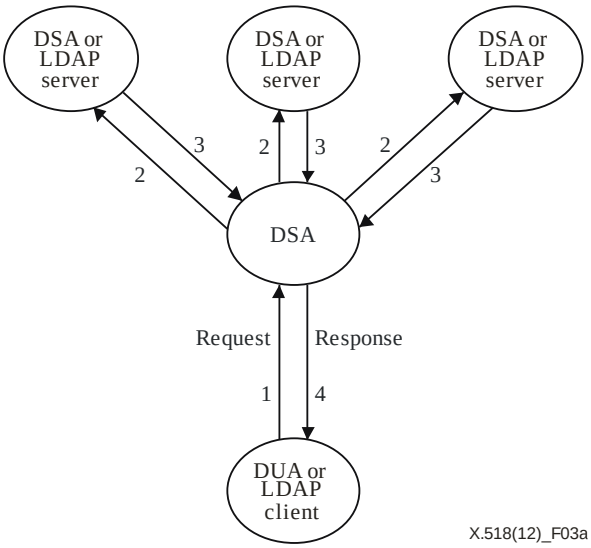


**Figure 3a – Parallel multi-chaining**

### 8.3.2 Sequential multi-chaining

With sequential multi-chaining, the DSA transfers one outgoing request at a time and waits for the result or error of one request before sending the next (see Figure 3b). Whilst sequential multi-chaining may not be the quickest mode of interaction, it is unlikely that duplicate results will be received.

NOTE – A DSA may use a combination of parallel multi-chaining and sequential multi-chaining.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

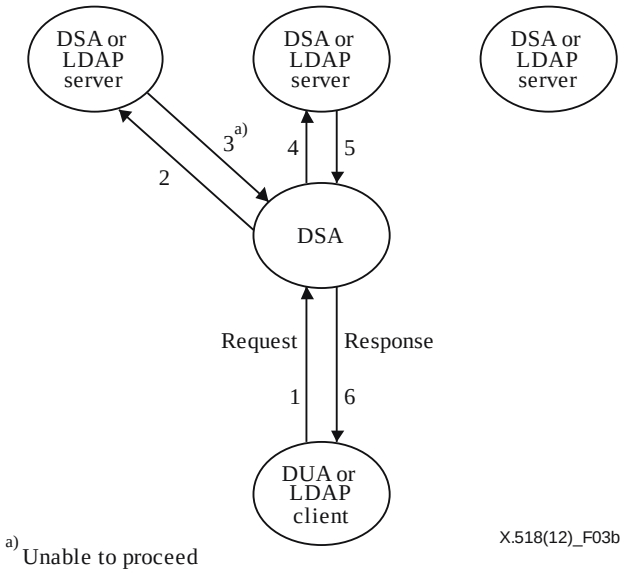


**Figure 3b – Sequential multi-chaining (as a result of NSSR decomposition)**

## 8.4 Referral

A referral (depicted in Figures 4a and 4b) is returned by a DSA in response to a request from a DUA, an LDAP client or another DSA. The referral may constitute the whole response (in which case it is categorized as an error) or just part of the response. An LDAP server may also return a referral as specified in IETF RFC 4511. A referral returned by a DSA contains a knowledge reference, which may be either a superior, subordinate, cross, non-specific subordinate, supplier, or master reference. A referral from an LDAP server is equivalent to a cross reference.

The DSA (Figure 4a) receiving the referral may use the knowledge reference contained therein, to subsequently chain or multi-cast (depending upon the type of reference) the original request to other directory servers. Alternatively, a DSA receiving a referral, may in turn pass the referral back in its response. A DUA or LDAP client (Figure 4b) receiving a referral may use it to contact one or more other DSAs to progress the request.

NOTE 1 – In Figures 4a and 4b, the order of interactions is defined by the numbers associated with the interaction lines.

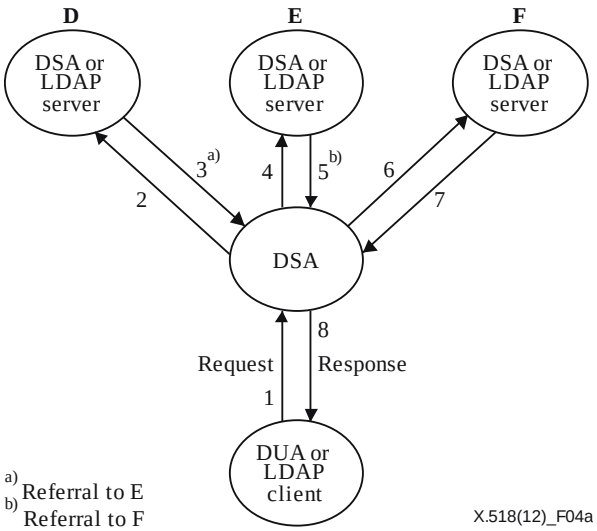


**Figure 4a – Referral mode (DSA acts on referrals)**


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

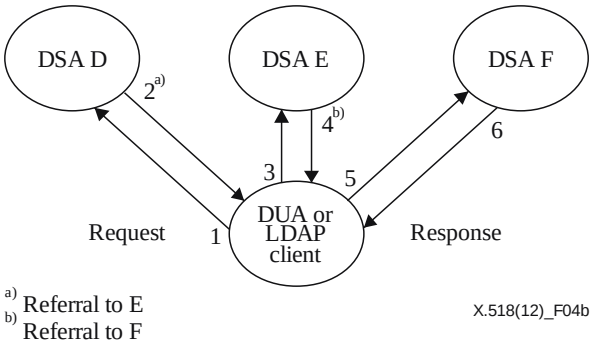


**Figure 4b – Referral mode (DUA or LDAP client acts on referrals)**

NOTE 2 – In Figure 4b, a referral to an LDAP client may be to an LDAP server. However, this situation is outside the scope of this Directory Specification.

## 8.5 Mode determination

If a DSA cannot itself fully resolve a request, it shall chain the request (or a request formed by decomposing the original one), to another DSA or LDAP server, unless:

a) chaining is prohibited by the requester or a bound DSA on behalf of an LDAP client via the **`chainingProhibited`** service control, in which case the DSA shall return a referral or a **`serviceError`** with problem **`chainingRequired`**; or

b) the DSA has administrative, operational, or technical reasons for preferring not to chain, in which case the DSA shall return a referral.

NOTE 1 – A "technical reason" for not chaining is that the DSA identified in the knowledge reference does not support the DSP.

NOTE 2 – If the **`localScope`** service control is set, then the DSA (or DMD) shall either resolve the request or return an error.

NOTE 3 – If the user prefers referrals, the user should set **`chainingProhibited`**.

NOTE 4 – An LDAP server will always return a referral.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

SECTION 4 – DSA ABSTRACT SERVICE

# 9 Overview of DSA abstract service

The service of the Directory is fully described in Rec. ITU-T X.511 | ISO/IEC 9594-3. When such a service is provided in a distributed environment, as modelled in clause 7, it can be regarded as being provided by means of a set of DSAs and zero or more LDAP servers. This is illustrated in Figure 1.

For each operation defined in the Directory service, a corresponding "chained" operation is defined in the DSA abstract service for use between DSAs cooperating in the accomplishment of that Directory service operation. Thus, a DSA receiving a Read operation from a DUA might require the assistance of another directory server (e.g., a DSA holding the target entry or a copy of it) to satisfy it, and so send that DSA a Chained Read operation.

The information types exchanged in the DSA abstract service are defined in clause 10. The operations and errors of the DSA abstract service are defined in clauses 11 to 13.

NOTE – Communication between a DSA (LDAP requester) and an LDAP server is outside the scope of DSA abstract service.

# 10 Information types

## 10.1 Introduction

This clause identifies, and in some cases defines, a number of information types which are subsequently used in the definition of the various operations of the DSA abstract service. The information types concerned are those which are common to more than one operation or which are sufficiently complex or self-contained to merit being defined separately from the operation which uses them.

Several of the information types used in the definition of the DSA abstract service are defined elsewhere. Clause 10.2 identifies these types and indicates the source of their definition. Clauses 10.3 to 10.10 each identifies and defines an information type.

## 10.2 Information types defined elsewhere

The following information types are defined in Rec. ITU-T X.501 | ISO/IEC 9594-2:

(Attributes)

– **aliasedEntryName**;

(Data types)

– **DistinguishedName**;

– **Name**;

– **RelativeDistinguishedName**.

The following information types are defined in Rec. ITU-T X.511 | ISO/IEC 9594-3:

(Operations)

– **abandon**;

(Errors)

– **abandoned**;

– **attributeError**;

– **nameError**;

– **securityError**;

– **serviceError**;

– **updateError**.

(Information Object Class)

– **OPTIONALLY-PROTECTED**;

(Data Type)

– **SecurityParameters**.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

The following information type is defined in Rec. ITU-T X.520 | ISO/IEC 9594-6:

(Data Type)

– **PresentationAddress**.

## 10.3 Chaining arguments

The **ChainingArguments** shall be present in each chained operation, to convey to a DSA the information needed to successfully perform its part of the overall task:

```
ChainingArguments ::= SET {
  originator                  [0]  DistinguishedName OPTIONAL,
  targetObject                [1]  DistinguishedName OPTIONAL,
  operationProgress           [2]  OperationProgress
                                 DEFAULT {nameResolutionPhase notStarted},
  traceInformation            [3]  TraceInformation,
  aliasDereferenced           [4]  BOOLEAN DEFAULT FALSE,
  aliasedRDNs                 [5]  INTEGER OPTIONAL,
  returnCrossRefs             [6]  BOOLEAN DEFAULT FALSE,
  referenceType               [7]  ReferenceType DEFAULT superior,
  info                        [8]  DomainInfo OPTIONAL,
  timeLimit                   [9]  Time OPTIONAL,
  securityParameters          [10] SecurityParameters DEFAULT {},
  entryOnly                   [11] BOOLEAN DEFAULT FALSE,
  uniqueIdentifier            [12] UniqueIdentifier OPTIONAL,
  authenticationLevel         [13] AuthenticationLevel OPTIONAL,
  exclusions                  [14] Exclusions OPTIONAL,
  excludeShadows              [15] BOOLEAN DEFAULT FALSE,
  nameResolveOnMaster         [16] BOOLEAN DEFAULT FALSE,
  operationIdentifier         [17] INTEGER OPTIONAL,
  searchRuleId                [18] SearchRuleId OPTIONAL,
  chainedRelaxation           [19] MRMapping OPTIONAL,
  relatedEntry                [20] INTEGER OPTIONAL,
  dspPaging                   [21] BOOLEAN DEFAULT FALSE,
  --                          [22] Not to be used
  --                          [23] Not to be used
  excludeWriteableCopies      [24] BOOLEAN DEFAULT FALSE,
  ... }

Time ::= CHOICE {
  utcTime          UTCTime,
  generalizedTime  GeneralizedTime,
  ... }

DomainInfo ::= ABSTRACT-SYNTAX.&Type
```

The various components have the following meaning:

a) The **originator** component need not be present if the **requester** component is present in **CommonArguments** value, if the **certification-path** component is present in the **SecurityParameters** value, or if requester information is only made available in the request, but not during the Bind operation. It shall not be present if requester information is not available. It shall be present, if requester information is only available as the result of the Bind operation.

b) The **targetObject** component shall convey the name of the object whose directory entry is being routed to. The role of this object depends on the particular operation concerned: it may be the object whose entry is to be operated on, or which is to be the base object for a request or subrequest involving multiple objects (e.g., **chainedList** or **chainedSearch**). This component need not be present if it has the same value as the object or base object parameter in the chained operation, in which case its implied value is that value. Otherwise, it shall be present.

c) The **operationProgress** component is used to inform the DSA of the progress of the operation, and hence of the role which it is expected to play in its overall performance. The information conveyed in this component is specified in clause 10.5. The default value indicates that the name resolution has not started.

d) The **traceInformation** component shall be used to prevent looping among DSAs when chaining is in operation. A DSA shall add a new element to trace information prior to chaining an operation to another DSA. On being requested to perform an operation, a DSA shall check, by examination of the trace information, that the operation has not formed a loop. The information conveyed in this component is specified in clause 10.6.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

e) The **aliasDereferenced** component is a **BOOLEAN** value which shall be used to indicate whether or not one or more alias entries have so far been encountered and dereferenced during the course of distributed name resolution. The default value of **FALSE** indicates that no alias entry has been dereferenced.

f) The **aliasedRDNs** component indicates how many of the RDNs of the **targetObject** component have been generated from the **aliasedEntryName** attributes of one (or more) alias entries. The integer value is set whenever an alias entry is encountered and dereferenced. This component shall be present if the **aliasDereferenced** component is **TRUE**. Otherwise, it shall be absent.

NOTE – This component is provided for compatibility with implementations based on Rec. CCITT X.518 (1988) | ISO/IEC 9594-4:1990. DUAs (and DSAs) implemented according to later editions of the Directory Specifications shall always omit this parameter from the **CommonArguments** of a subsequent request. In this way, the Directory will not signal an error if aliases dereference to further aliases.

g) The **returnCrossRefs** component is a Boolean value which indicates whether or not knowledge references, used during the course of performing a distributed operation, are requested to be passed back to the initial DSA as cross references, along with a result or referral. The default value of **FALSE** indicates that such knowledge references shall not be returned.

h) The **referenceType** component, when present, shall indicate, to the DSA being asked to perform the operation, what type of knowledge was used to route the request to it. The DSA may therefore be able to detect errors in the knowledge held by the invoker. If such an error is detected, it shall be indicated by a **serviceError** with problem **invalidReference**. **ReferenceType** is specified fully in clause 10.7. If the **referenceType** is absent, then the value **superior** shall be assumed.

i) The **info** component is used to convey DMD-specific information among DSAs which are involved in the processing of a common request. This component is of type **DomainInfo**, which is of unrestricted type:

j) The **timeLimit** component, if present, shall indicate the time by which the operation is to be completed (see clause 16.1.4.1). Before a value of **Time** is used in any comparison operation and if the syntax of **Time** has been chosen as the **UTCTime** type, the value of the two-digit year field shall be rationalized into a four-digit year value as follows:

– If the 2-digit value is 00 to 49 inclusive, the value shall have 2000 added to it.

– If the 2-digit value is 50 to 99 inclusive, the value shall have 1900 added to it.

NOTE 3 – The use of **GeneralizedTime** may prevent interworking with implementations unaware of the possibility of choosing either **UTCTime** or **GeneralizedTime**. It is the responsibility of those specifying the domains in which this Directory Specification will be used, e.g., profiling groups, as to when the **GeneralizedTime** may be used. In no case shall **UTCTime** be used for representing dates beyond 2049.

k) The **SecurityParameters** data type is specified in Rec. ITU-T X.511 | ISO/IEC 9594-3. Its absence is deemed equivalent to there being an empty set of security parameters.

l) The **entryOnly** component shall be set to **TRUE** if the original operation was a Search with the **subset** argument set to **oneLevel**, and an alias entry was encountered as an immediate subordinate of the **baseObject**. The DSA which successfully performs name resolution on the **targetObject** name shall perform object evaluation on only the named entry.

m) **uniqueIdentifier** component is optionally supplied when it is required to confirm the requester name. The **UniqueIdentifier** data type is described in Rec. ITU-T X.501 | ISO/IEC 9594-2.

n) **authenticationLevel** component, when present, shall indicate the authentication level as established during the Bind operation. If this component is absent, a performing DSA shall assume that there has been no authentication (anonymous Bind). This component should be present whenever the requester has been authenticated. The **AuthenticationLevel** data type is described in Rec. ITU-T X.501 | ISO/IEC 9594-2.

o) The **exclusions** component has significance only for Search operations; it shall indicate, if present, which subtrees of entries subordinate to the **targetObject** shall be excluded from the result of the Search operation (see clause 10.10).

p) The **excludeShadows** component has significance only for Search, LDAP Search and List operations; it indicates that the search shall be applied to entries and not to entry copies. If a request is received by an LDAP requester for forwarding it to an LDAP server, the request shall not be forwarded if the referenced naming context within the LDAP server is of **category shadow** (see clause 10.8). This optional component may be used by a DSA as one way to avoid the receipt of duplicate results (see clause 20.1).


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

q) The **nameResolveOnMaster** component only has significance during name resolution, and is only set if NSSRs have been encountered. If set to **TRUE**, it signals that subsequent name resolution, i.e., matching the remaining RDNs from **nextRDNToBeResolved**, shall not employ entry copy information, including writeable copies in a multi-master LDAP server implementation; subsequent resolution of each remaining RDN shall be done in the master DSA for the entry identified by that RDN (see clause 20.1).

r) The **operationIdentifier** component facilitates the correlation of DAP operations with subsequent related DSP operations as well as with results. It is assigned by the DSA that first receives a DAP request or is copied from the chaining arguments of DSP requests that require further chaining. The DSA assigning the **operationIdentifier** shall not reuse the assigned integer for a sufficiently long time period. The correlation of related DAP and DSP requests and results are facilitated by DSA logging, for each operation and result, the **operationIdentifier** together with the name of the DSA that assigned it (the first DSA in **traceInformation** on a chained request). Such correlation may be useful for the purposes of logging, auditing, charging and settlements, etc.

s) The **searchRuleId** component, when present, shall convey the unique identity of a search-rule. It is included by the DSA performing the initial **Search procedure (I)** in case this procedure starts within a service-specific administrative area and the search operation is progressed to other DSAs either when progressing down the DIT, when following aliases or when following hierarchical group pointers.

t) The **chainedRelaxation** component enables relaxation to be carried out in a distributed manner for chained Search operations. If a DSA received a chained search operation, and supports relaxation policies, it can use the supplied **chainedRelaxation** component in place of any other relaxation policy that it might implement, thereby enabling relaxation to be coordinated among the DSAs that potentially return search results.

u) The **relatedEntry** element shall be present whenever the receiving DSA is required to resolve related entries. When present, the receiving DSA shall respond *only* to the specific related entry element specified by the **relatedEntry** value in **joinAttributes** of the **SearchArgument**. Thus, a **relatedEntry** value of zero shall select the first element in the **joinAttributes** sequence the **SearchArgument**. The value shall never exceed one less than the number of elements in the **joinAttributes** component of **SearchArgument**. The absence of the **relatedEntry** element in the **ChainingArguments** of a DSP operation specifying related entries shall indicate that the distributed operation being chained on is the **base** search, and not the related entry part of the search.

If a DSA to which chaining is being carried out is required to handle both normal search results and related-entry results, this shall be done by sending the DSA two distinct DSP operations.

When the **relatedEntry** element is present, the following special rules shall apply:

– in evaluating the **infoTypes** subcomponent of **selection** component of **SearchArgument**, **infoTypes** shall be taken as having the value **attributeTypesAndValues**, whatever the originally specified value;

– all attributes specified in any **joinAtt** component of **JoinAttPair** shall be included in the selection, whether or not previously included there;

– the DSA coordinating related entry results shall omit values and unspecified arguments, so as to make the result conform with the original user request.

The **relatedEntry** argument shall be passed on in consequent outgoing **ChainingArguments** by a DSA that supports related entries.

v) The **dspPaging** component may be used to request DSP paging. If the bound DSA is different from the initial performer (see clause 15.5.4) and the bound DSA supports DSP paged results, it may set this component to **TRUE** to instruct the initial performer to provide DSP paged results. If this component is **FALSE** (default), the initial performer shall not perform DSP paged results. An initial performer that supports DSP paged results shall not forward this component to DSA(s) to which it is sending subrequests.

w) The **excludeWriteableCopies** component has significance only for an LDAP requester receiving a Search, LDAP search and List operation to be forwarded to an LDAP server. The request shall not be forwarded if the referenced naming context within the LDAP server is of **category writeableCopy** (see clause 10.8). This optional component may be used by a DSA as one way to avoid the receipt of duplicate results (see clause 20.1).

## 10.4 Chaining results

The **ChainingResults** are present in the result of each operation and provide feedback to the DSA which invoked the operation.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

```
ChainingResults ::= SET {
  info                [0]  DomainInfo OPTIONAL,
  crossReferences     [1]  SEQUENCE SIZE (1..MAX) OF CrossReference OPTIONAL,
  securityParameters  [2]  SecurityParameters DEFAULT {},
  alreadySearched     [3]  Exclusions OPTIONAL,
  ... }

CrossReference ::= SET {
  contextPrefix  [0]  DistinguishedName,
  accessPoint    [1]  AccessPointInformation,
  ... }
```

The various components have the following meaning:

a) The **info** component, when present, shall be used to convey DMD-specific information among DSAs which are involved in the processing of a common request. This component is of type **DomainInfo**, which is of unrestricted type.

b) The **crossReferences** component shall not be present in the **ChainingResults** unless the **returnCrossRefs** component of the corresponding request had the value **TRUE**. This component consists of a sequence of **CrossReference** items, each of which contains a **contextPrefix** and an **accessPoint** descriptor (see clause 10.8).

   A **CrossReference** may be added by a DSA when it matches part of the **targetObject** argument of an operation with one of its context prefixes. The administrative authority of a DSA may have a policy not to return such knowledge, and will, in this case, not add an item to the sequence.

c) The **SecurityParameters** data type is specified in Rec. ITU-T X.511 | ISO/IEC 9594-3. The absence of the **securityParameters** component is deemed equivalent to there being an empty set of security parameters.

d) The **alreadySearched** component, if present, shall indicate which subordinate RDNs subordinate to the **targetObject** have been processed as part of a chained Search operation and therefore shall be excluded in a subsequent subrequest.

## 10.5 Operation progress

An **OperationProgress** value describes the state of progress in the performance of an operation which several DSAs shall participate in.

```
OperationProgress ::= SET {
  nameResolutionPhase  [0]  ENUMERATED {
    notStarted  (1),
    proceeding  (2),
    completed   (3),
    ... },
  nextRDNToBeResolved  [1]  INTEGER OPTIONAL,
  ... }
```

The various components have the following meaning:

a) The **nameResolutionPhase** component shall indicate which phase has been reached in handling the **targetObject** name of an operation. Where this indicates that name resolution has **notStarted**, then a DSA has not hitherto been reached with a naming context containing the initial RDN(s) of the name. If name resolution is **proceeding**, then the initial part of the name has been recognized, although the DSA holding the target object has not yet been reached. The **nextRDNToBeResolved** indicates how much of the name has already been recognized (see item b). If name resolution is **completed**, then the DSA holding the target object has been reached, and performance of the operation proper is proceeding.

b) The **nextRDNToBeResolved**, when present, shall indicate to the DSA which of the RDNs in the **targetObject** name is the next to be resolved. It takes the form of an integer in the range one to the number of RDNs in the name. This component shall be absent if the **nameResolutionPhase** component does not have the value **proceeding**.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

## 10.6 Trace information

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

```
TraceInformation ::= SEQUENCE OF TraceItem

TraceItem ::= SET {
  dsa                 [0]  Name,
  targetObject        [1]  Name OPTIONAL,
  operationProgress   [2]  OperationProgress,
  ... }
```

Each DSA, which is propagating an operation to another DSA, shall add a new **TraceItem** to the end of the **TraceInformation**. Each such **TraceItem** value has the following components:

a) the **dsa** component shall hold the name of the DSA which is adding the item;

b) the **targetObject** component, when present, shall be the value received on the **targetObject** component of the **ChainingArguments** value of the incoming request. This parameter shall be omitted if:

- the request being chained came from a DUA, in which case its implied value is the **object** or **baseObject** in the DAP operation;
- the request is received from an LDAP client, in which case its implied value is the **object** or **baseObject** of the LDAP request; or
- if its value is the same as the (actual or implied) **targetObject** in the **ChainingArgument** of the outgoing request;

c) the **operationProgress** component shall have a value determined as follows:

- If the incoming request is received from a DUA, the value shall be taken from the **operationProgress** component of the **CommonArguments** of the DAP request. If this component is absent on the DAP request, the default value **notStarted** shall be used.
- If the incoming request is received from an LDAP client, the value **notStarted** shall be used.
- If the incoming request is received from a DSA, the value shall be taken from the **operationProgress** component of the **ChainedArguments** value. If this component is absent on the request, the default value **notStarted** shall be used.

## 10.7 Reference type

A **ReferenceType** value indicates one of the various kinds of references defined in Rec. ITU-T X.501 | ISO/IEC 9594-2.

```
ReferenceType ::= ENUMERATED {
  superior               (1),
  subordinate            (2),
  cross                  (3),
  nonSpecificSubordinate (4),
  supplier               (5),
  master                 (6),
  immediateSuperior      (7),
  self                   (8),
  ditBridge              (9),
  ... }
```

## 10.8 Access point information

There are three types of access points:

```
AccessPoint ::= SET {
  ae-title            [0]  Name,
  address             [1]  PresentationAddress,
  protocolInformation [2]  SET SIZE (1..MAX) OF ProtocolInformation OPTIONAL,
  --                  [6]  Not to be used
  ... }
```

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

```
MasterOrShadowAccessPoint ::= SET {
  COMPONENTS OF AccessPoint,
  category          [3]  ENUMERATED {
    master            (0),
    shadow            (1),
    writeableCopy     (2),
    ... } DEFAULT master,
  chainingRequired  [5]  BOOLEAN DEFAULT FALSE,
  ... }

MasterAndShadowAccessPoints ::= SET SIZE (1..MAX) OF MasterOrShadowAccessPoint
```

a) An **AccessPoint** value identifies a particular point at which access to the Directory, specifically to a DSA or LDAP server, can occur. When referring to a DSA, the access point shall have a **Name**, that of the DSA concerned. It shall have a **PresentationAddress** to be used in communications to that DSA or LDAP server (see clause 11 of Rec. ITU-T X.519 | ISO/IEC9594-5 for additional information about NSAP formats). When referring to a DSA using IDM protocol, **pSelector**, **sSelector** and **tSelector** components shall be ignored. When referring to an LDAP server, the **ae-title**, **protocolInformation** **pSelector**, **sSelector** and **tSelector** components shall be ignored (see clause 11.3 of Rec. ITU-T X.519 | ISO/IEC 9594-5).

b) A **MasterOrShadowAccessPoint** value identifies an access point to the Directory. The **category**, either **master** or **shadow**, of the access point is dependent upon whether it points to a naming context or to a commonly usable replicated area. The **category writeableCopy** is only applicable if the access point is for an LDAP server with writeable copy entries. The **chainingRequired** component indicates whether chaining is required for that DSA, i.e., a referral shall not be returned for that DSA.

c) A **MasterAndShadowAccessPoints** value identifies a set of access points to the Directory, i.e., a set of related Directory servers. These access points share the property that each refers to a DSA or LDAP server holding entry information from a common naming context (or a common set of naming contexts mastered in one DSA when the value is a value of the **nonSpecificKnowledge** attribute). A **MasterAndShadowAccessPoints** value indicates the **category** of each **AccessPoint** value it contains. The access point of the master DSA or LDAP server of the naming context need not be included in the set.

NOTE – Implementors should recognize that it is possible for an LDAP server, even if identified as **shadow**, to update entries in response to an LDAP update operation that it receives.

An **AccessPointInformation** value identifies one or more access points to the Directory.

```
AccessPointInformation ::= SET {
  COMPONENTS OF          MasterOrShadowAccessPoint,
  additionalPoints  [4]  MasterAndShadowAccessPoints OPTIONAL,
  ... }
```

In the case of DSAs based on Rec. CCITT X.518 (1988) | ISO/IEC 9594-4:1990 producing an **AccessPointInformation** value, the optional component of the set is absent. In the case of DSAs based on Rec. CCITT X.518 (1988) | ISO/IEC 9594-4:1990 interpreting an **AccessPointInformation** value, any **MasterAndShadowAccessPoints** value present is ignored.

In the case of DSAs based on Rec. ITU-T X.518 (1993) | ISO/IEC 9594-4:1995 or later editions, the **MasterOrShadowAccessPoint** value component produced for an **AccessPointInformation** value may be of category master or shadow, as determined by the knowledge selection procedure of the DSA producing the value. It may be viewed as a suggested access point provided by the DSA generating the value to the DSA receiving it. A **MasterAndShadowAccessPoints** value may optionally also be produced for an **AccessPointInformation** value. This constitutes additional information which may be employed by the receiving DSA's knowledge selection procedure to determine an alternative access point.

## 10.9 DIT bridge knowledge.

A **DitBridgeKnowledge** value identifies a particular point at which access to another DIT, specifically to a DSA or an LDAP server, can occur. **DitBridgeKnowledge** specifies an access point at which that DSA or LDAP server may be accessed.

```
DitBridgeKnowledge ::= SEQUENCE {
  domainLocalID  UnboundedDirectoryString OPTIONAL,
  accessPoints   MasterAndShadowAccessPoints,
  ... }
```


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

**domainLocalID** contains a readable description identifying the DIT included in the reference.

## 10.10 Exclusions

As defined in clause 10.3, the **exclusions** component of **ChainingArguments** is used to limit the scope of a Search operation by identifying a number of entries subordinate to the target object which, together with all of their subordinates, shall not be included in the processing of a Search operation. The **exclusion** component is defined as a value of the ASN.1 type **Exclusions**.

```
Exclusions ::= SET SIZE (1..MAX) OF RDNSequence
```

Each **RDNSequence** value in the **Exclusions** set should identify the context prefix of a naming context subordinate to the target object. If a DSA receives a **search** request with an **RDNSequence** value that does not conform to this constraint, the DSA may ignore that value. The **RDNSequence** is relative to the target object, and is not the distinguished name of the context prefix.

**Exclusions** can, besides being part of a user request, be used by DSAs to minimize duplicate information returned from Search subrequests performed in the presence of shadowed information.

Figure 5 illustrates an example of the use of **Exclusions**. In this example, a DSA holds two replicated areas, one beneath the other. One starts with context prefix X, the other with context prefix C. An entry copy at Y has three subordinate references to naming contexts, A, B and C.

If, as an example, a subtree Search is performed in this DSA, starting with a base object within naming context X, the DSA can provide information from replicated areas X and C. The information from naming contexts A and B has to be provided via the subordinate references. When performing request decomposition, continuation references, to be used in either **partialResults** or chaining, will specify Y as the target object and C as a single element of an **Exclusions** set.

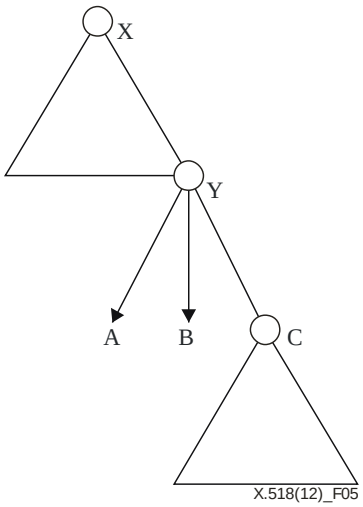


**Figure 5 – Exclusions**

## 10.11 Continuation reference

A **ContinuationReference** describes how the performance of all or part of an operation can be continued at one or more different directory servers. It is typically returned as a referral when the DSA involved is unable or unwilling to propagate the request itself.

```
ContinuationReference ::= SET {
  targetObject          [0]  Name,
  aliasedRDNs           [1]  INTEGER OPTIONAL,
  operationProgress     [2]  OperationProgress,
  rdnsResolved          [3]  INTEGER OPTIONAL,
  referenceType         [4]  ReferenceType,
  accessPoints          [5]  SET OF AccessPointInformation,
  entryOnly             [6]  BOOLEAN DEFAULT FALSE,
  exclusions            [7]  Exclusions OPTIONAL,
  returnToDUA           [8]  BOOLEAN DEFAULT FALSE,
  nameResolveOnMaster   [9]  BOOLEAN DEFAULT FALSE,
  ... }
```

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

The various components have the following meaning:

a) The **targetObject** component shall indicate the name which is proposed to be used in continuing the operation. This might be different from the name received in **targetObject** of the incoming request if, for example, an alias has been dereferenced, or the base object in a search has been located.

b) The **aliasedRDNs** component indicates how many (if any) of the RDNs in the target object name have been produced by dereferencing an alias. The argument is only present if an alias has been dereferenced.

NOTE – This component is provided for backward compatibility with implementations based on Rec. CCITT X.5** (1988) | ISO/IEC 9594-*:1990.

This component shall not be included in **CommonArguments** when implementing according to Rec. ITU-T X.5** (1993) | ISO/IEC 9594-*:1995 or later editions of these Directory Specifications.

c) The **operationProgress** shall indicate the amount of name resolution which has been achieved, and which will govern the further performance of the operation by the DSAs named, should the DSA or DUA receiving the **ContinuationReference** wish to follow it up.

d) The **rdnsResolved** component value (which need only be present if some of the RDNs in the name have not been the subject of full name resolution, but have been assumed to be correct from a cross reference) shall indicate how many RDNs have actually been resolved, using internal references only.

e) The **referenceType** component shall indicate what type of knowledge was used in generating this continuation.

f) The **accessPoints** component shall indicate the access points which are to be contacted to achieve this continuation. Only where non-specific subordinate references are involved can there be more than one **AccessPointInformation** item.

g) The **entryOnly** component shall be set to **TRUE** if the original operation was a search, with the **subset** argument set to **oneLevel**, and an alias entry was encountered as an immediate subordinate of the **baseObject**. The DSA which successfully performs name resolution on the **targetObject** name shall perform object evaluation on only the named entry. If the original operation was not a search, with the **subset** argument set to **oneLevel**, this component shall be absent or take the value **FALSE**.

h) The **exclusions** component, when present, shall identify a set of subordinate naming contexts that should not be explored by the receiving DSA.

i) The **returnToDUA** element is optionally supplied when the DSA creating the continuation reference wishes to indicate that it is unwilling to return information via an intermediate DSA (e.g., for security reasons), and wishes to indicate that information may be directly available via an operation over a DAP or LDAP between the originating DUA or LDAP client and the DSA. When **returnToDUA** is set to **TRUE**, **referenceType** may be set to **self**.

j) The **nameResolveOnMaster** element is optionally supplied when the DSA creating the continuation reference has encountered NSSRs. If set to **TRUE**, it signals that subsequent name resolution, i.e., matching the remaining RDNs from **nextRDNToBeResolved**, shall not employ entry copy information including writeable copies in a multi-master LDAP server implementation; subsequent resolution of each remaining RDN shall be done in the master DSA for the entry identified by that RDN (see clause 20.1).

# 11 Bind and Unbind

DSA Bind and DSA Unbind operations, respectively, are used by a DSA at the beginning and at the end of a period of accessing another DSA. The binding or unbinding of a DSP application-association shall not, of itself, cause the loss of any distributed paged results which were requested in the course of the application-association.

## 11.1 DSA Bind

### 11.1.1 DSA Bind syntax

A **dSABind** operation is used to begin a period of cooperation between two DSAs providing the directory service.

```
dSABind OPERATION ::= {
  ARGUMENT     DSABindArgument
  RESULT       DSABindResult
  ERRORS       { directoryBindError } }

DSABindArgument  ::=  SET  {
  credentials  [0]  DSACredentials OPTIONAL,
```


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

```
  versions      [1]  Versions DEFAULT {v1} }

DSACredentials  ::=  CHOICE  {
  simple             [0]  SimpleCredentials,
  strong             [1]  StrongCredentials,
  externalProcedure  [2]  EXTERNAL,
  spkm               [3]  SpkmCredentials }

DSABindResult  ::=  DSABindArgument
```

### 11.1.2 DSA Bind arguments

The components of the **DSABindArgument** are identical to their counterparts in the DirectoryBind (see Rec. ITU-T X.511 | ISO/IEC 9594-3) with the following differences:

- The **Credentials** of the **DirectoryBindArgument** allows information identifying the AE-Title of the initiating DSA to be sent to the responding DSA. The AE-Title shall be in the form of a Directory Distinguished Name.
- The **SaslCredentials** shall not be included in the **Credentials**.
- The **Credentials** of the **DirectoryBindResult** allows information identifying the AE-Title of the responding DSA to be sent to the initiating DSA. The AE-Title shall be in the form of a Distinguished Name.

### 11.1.3 DSA Bind results

The components of the DSABindResult are identical to their counterparts in the DirectoryBindResult (see Rec. ITU-T X.511 | ISO/IEC 9594-3) with the following differences:

- The **Credentials** of the **DirectoryBindResult** allows information identifying the AE-Title of the responding DSA to be sent to the initiating DSA. The AE-Title shall be in the form of a Distinguished Name.
- The **SaslCredentials** shall not be included in the **Credentials**.

### 11.1.4 DSA Bind errors

Should the Bind request fail, a Bind error shall be returned. If the Bind request was either using strong authentication or SPKM credentials are supplied, then the Bind responder may sign the error parameters.

The **versions** parameter of the **dSABindError** indicates which versions are supported by the responding DSA.

The SecurityParameters components (see clause 7.10 of Rec. ITU-T X.511 | ISO/IEC 9594-3) shall be included if the error is to be signed.

A **securityError** or **serviceError** shall be supplied as follows:

```
  securityError  inappropriateAuthentication
                 invalidCredentials
                 blockedCredentials

  serviceError   unavailable
```

## 11.2 DSA Unbind

The unbinding at the end of a period of cooperation between two DSAs providing the Directory service is for the OSI environment specified in clauses 7.6.4 and 7.6.5 of Rec. ITU-T X.519 | ISO/IEC 9594-5 and for the TCP/IP environment in clause 9.2.2 of Rec. ITU-T X.519 | ISO/IEC 9594-5.

# 12 Chained operations

For each of the operations used to access the directory abstract service, there is an operation used between cooperating DSAs in a one-to-one correspondence. The names of the operations have been chosen to reflect that correspondence by prefixing the names of operations used between cooperating DSAs with the term "Chained".

The arguments, results and errors of the chained operations are, with one exception, formed systematically from the arguments, results and errors of the corresponding operations in the Directory abstract service (as specified in clause 12.1). The one exception is the Chained Abandon operation, which is syntactically equivalent to its Directory service counterpart (specified in clause 12.2).

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

## 12.1 Chained operations

A DSA, having received an operation from a DUA, having created an LDAP Transport operation resulting from a received request from an LDAP client, or having created a Linked LDAP operation resulting from a partial result from an LDAP server, shall construct a chained form of that operation if it is to be propagated to another DSA. A DSA, having received a chained form of an operation, may also elect to chain it to another DSA. The DSA invoking a chained form of an operation may sign the argument of the operation; the DSA performing the operation, if so requested, may sign the result or error returned by the responder of the operation.

The chained form of an operation is specified using the parameterized type **chained { }**.

```
chained{OPERATION:operation} OPERATION ::= {
  ARGUMENT              OPTIONALLY-PROTECTED {SET {
    chainedArgument       ChainingArguments,
    argument        [0]   operation.&ArgumentType } }
  RESULT OPTIONALLY-PROTECTED {SET {
    chainedResult         ChainingResults,
    result          [0]   operation.&ResultType}}
  ERRORS
    {operation.&Errors EXCEPT referral | dsaReferral}
  CODE                  operation.&operationCode }
```

NOTE 1 – The operations of the Directory abstract service which may be used as the actual parameter of **chained { }** include the **abandoned** error. The presence of this error among the set of possible errors of a chained operation reflects the possibility discussed in clause 12.2, that a **chainedAbandon** can be generated for a **chainedModify** operation when a linked association fails.

NOTE 2 – The definitive specification of the DSA abstract service in Annex A applies this parameterized type to construct all the chained operations of the abstract service.

The argument of the derived operation has the components:

a) **chainedArgument** – This is a value of **ChainingArguments** that contains the information, supplementing the information provided in the DAP argument. This additional information is needed in order for the receiving DSA to handle the operation properly. This data type is defined in clause 10.3.

b) **argument** – This is a value **operation.&Argument** and consists of the original DAP argument, as specified in the appropriate clause of Rec. ITU-T X.511 | ISO/IEC 9594-3.

Should the request succeed, the result of the derived operation has the components:

a) **chainedResult** – This is a value of **ChainingResults** which contains that information, over and above that to be supplied in the DAP result, which may be needed by previous DSAs in a chain. This information type is defined in clause 10.4.

b) **result** – This is a value **operation.&Result** and consists of the result which is being returned by the performer of this operation, and which is intended to be passed back in the result to the requester. This information is as specified in the appropriate clause of Rec. ITU-T X.511 | ISO/IEC 9594-3.

Should the request fail, one of the errors of the set **operation.&Errors** will be returned, except that **dsaReferral** is returned instead of **referral**. The set of errors, which may be reported, is as described for the corresponding operation in Rec. ITU-T X.511 | ISO/IEC 9594-3. The error **dsaReferral** is described in clause 13.2.

## 12.2 Chained Abandon operation

A **chainedAbandon** request is used by one DSA to indicate to another that it is no longer interested in having a previously invoked distributed operation performed. This may be for any number of reasons, of which the following are examples:

– the operation which led to the DSA originally chaining has itself been abandoned, or has implicitly been aborted by the breakdown of an application-association;

– the DSA has obtained the necessary information in another way, e.g., from a faster responding DSA involved in the parallel multi-chaining.

A DSA is never obliged to issue a **chainedAbandon** request, or indeed to actually abandon an operation if requested to do so.

If the Chained Abandon operation actually succeeds in stopping the performance of an operation, then a result will be returned, and the subject operation will return an **abandoned** error. If the **chainedAbandon** does not succeed in stopping the operation, then it will return an **abandonFailed** error itself.

A DSA may also issue an **AbandonRequest** to an LDAP server under the same conditions as above.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

### 12.3 Chained operations and protocol version

Operations which require a protocol version greater than v1 (such as the **modifyEntry** operation with certain arguments) or which return different results when used with a protocol version greater than v1 (such as **modifyEntry** with a signed argument) shall only be chained on application-associations with the same or a greater version number than that used to convey the request.

## 13 Chained errors

### 13.1 Introduction

For the most part, the same errors can be returned in the DSA abstract service which can be returned in the directory abstract service. The exceptions are that the **dsaReferral** "error" is returned (see clause 13.2), instead of **referral**, and the following service problems have the same abstract syntax but different semantics:

a) **serviceError** with problem **invalidReference** – The DSA returning this error detected an error in the calling DSA's knowledge as specified in the **referenceType** chaining argument.

b) **serviceError** with problem **loopDetected** – The DSA returning this error detected a loop in the knowledge information in the Directory.

The precedence of the errors which may occur is as for their precedence in the directory abstract service, as specified in Rec. ITU-T X.511 | ISO/IEC 9594-3.

If an error occurs during a chained operation, the responding DSA may sign the error returned.

### 13.2 DSA referral

The **dsaReferral** error shall be generated by a DSA when, for whatever reason, it does not wish to continue performing an operation by chaining the operation to one or more other directory servers. The circumstances where it may return a referral are described in clause 8.3.

```
dsaReferral ERROR ::= {
  PARAMETER           OPTIONALLY-PROTECTED { DsaReferralData }
  CODE                id-errcode-dsaReferral }

DsaReferralData ::= SET {
  reference      [0]  ContinuationReference,
  contextPrefix  [1]  DistinguishedName OPTIONAL,
  ...,
  ...,
  COMPONENTS OF       CommonResults }
```

The various components of **DsaReferralData** data type have the following meaning:

a) The **reference** component contains the information needed by the invoker to propagate an appropriate further request, perhaps to another DSA or to an LDAP server. The ContinuationReference data type is specified in clause 10.11.

b) The **contextPrefix** component may optionally be included if the **returnCrossRefs** component of the **ChainingArguments** for this operation had the value **TRUE**, and the referral is being based upon a subordinate or cross-reference. Otherwise, it shall be absent. The administrative authority of any DSA will decide which knowledge references, if any, can be returned in this manner (the others, for example, may be confidential to that DSA).

The information provided can optionally be qualified by the use of the **notification** component of **CommonResults**.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

SECTION 5 – DISTRIBUTED PROCEDURES

# 14 Introduction

## 14.1 Scope and limits

This clause specifies the procedures for distributed operation that are performed by DSAs. Each DSA individually performs the procedures described below; the collective action of all DSAs produces the full set of services provided to users of the Directory.

## 14.2 Conformance

The description of DSA procedures in this section is based on the models in clauses 8 and 9 of Rec. ITU-T X.501 | ISO/IEC 9594-2 and clauses 7 and 8 of this Directory Specification. The flow charts and their corresponding textual descriptions are one of the means of mapping a given set of external (DAP, LDAP and/or DSP) inputs to a DSA into one or more external outputs (i.e., a result, error, referral, chained requests or LDAP request) produced by that DSA, depending on the particular DSA information tree held by that DSA.

It is probable that the Directory will be distributed across DSAs implemented according to different editions of these Directory Specifications. The DUA or LDAP client initiating the request will be unaware as to which edition the DSA or DSAs satisfying the DUA's or LDAP client's request will have been implemented. Therefore to allow operation in such a heterogeneous environment, a DSA shall be implemented according to the rules of extensibility defined in clause 12 of Rec. ITU-T X.519 | ISO/IEC 9594-5.

A DSA implementation shall be functionally equivalent to the external behaviour specified by the procedures described here. The algorithms used by a particular DSA implementation to derive the correct output(s) from the given inputs and DSA information tree held are not standardized.

NOTE – The flowcharts which accompany the procedures are intended to be used as aids towards understanding the procedures. They are not to be considered as being a precise alternative to the textual descriptions. Where there is a disparity between the textual description and the flowchart for a particular procedure, it is intended that the textual description take precedence.

### 14.2.1 Interaction involving a DSA based on Rec. CCITT X.5** (1988) | ISO/IEC 9594-*:1990

If the modify operations evaluate across DSA boundaries (i.e., **`addEntry`** with **`TargetSystem`**, Remove or Rename a context prefix), then this Directory Specification only specifies how two DSAs when implemented accordingly to Rec. ITU-T X.5** (1993) | ISO/IEC 9594-*:1995 or later edition shall behave. The interaction between two DSAs based on Rec. CCITT X.5** (1988) | ISO/IEC 9594-*:1990, or between a DSA based on Rec. CCITT X.5** (1988) | ISO/IEC 9594-*:1990 and a DSA based on Rec. ITU-T X.5** (1993) | ISO/IEC 9594-*:1995 or a later edition, is outside the scope of these Directory Specifications. When mixed edition DSAs have a hierarchical operational binding, knowledge of each other's edition may allow a consistent error to be given to the user.

## 14.3 Conceptual model

The complexity of the Directory's distributed operation gives rise to a need for conceptual modelling using both narrative and pictorial descriptive techniques. However, neither the narrative nor the diagrams should be construed as a formal description of distributed directory operation.

## 14.4 Individual and cooperative operation of DSAs

The model views DSA operation from two separate perspectives, which, taken together, provide a complete, operational picture of the Directory.

a) **DSA-centred perspective** – In this perspective, the set of procedures that support the Directory is described from the viewpoint of a single DSA. This makes it possible to provide a definitive specification of each procedure and to fully account for their interrelationships and overall control structure. Clauses 16 to 22 describe the DSA procedures from a DSA-centred perspective.

b) **operation-centred perspective** – The DSA-centred view provides complete detail but makes it difficult to understand the structure of individual operations, which may undergo processing by multiple DSAs. Consequently, clause 15 adopts a primarily operation-centred view to introduce the processing phases applicable to each.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

To support the distributed operation of the Directory, each DSA shall perform actions needed to realize the intent of each operation and additional actions needed to distribute that realization across multiple DSAs. Clause 15 explores the distinction between these two kinds of actions. In clauses 16 to 22, both kinds of actions are specified in detail.

### 14.5 Cooperative agreements between DSAs

All DSAs which are in a subordinate/superior relationship due to the naming contexts that they hold have hierarchical and/or non-specific hierarchical operational bindings between them, depending upon the types of knowledge reference held by those DSAs.

Hierarchical and non-specific hierarchical operational bindings between DSAs may be administered using the procedures of clauses 24 and 25, or by off-line means.

A DSA holding entries which are within the administrative area of its superior DSA shall administer the subschema, shall follow the governing-search-rule (if any) and shall control access to the entries, as required by the administrative authority. The regulation of entries within an administrative area may be performed as defined in Rec. ITU-T X.501 | ISO/IEC 9594-2 or may be performed by local mechanisms.

## 15 Distributed Directory behaviour

### 15.1 Cooperative fulfilment of operations

Each DSA is equipped with procedures capable of completely fulfilling all directory operations. In the case that a DSA contains the entire DIB, all operations are completely carried out within that DSA. In the case that the DIB is distributed across a distributed directory, the completion of a typical operation is fragmented, with just a portion of that operation carried out in each of potentially many cooperating directory servers.

In the distributed environment, the typical DSA sees each operation as a transitory event: the operation is invoked by a DUA, an LDAP client or some other DSA; the DSA carries out processing on the object and then directs it toward another directory server for further processing.

An alternative view considers the total processing experienced by an operation during its fulfilment by possible multiple, cooperating directory servers. This perspective reveals the common processing phases that apply to all operations.

### 15.2 Phases of operation processing

Every directory operation may be thought of as comprising three distinct phases:

a) the *Name Resolution* phase in which the name of the object on whose entry a particular operation is to be performed is used to locate the DSA or LDAP server that holds the entry;

b) the *Evaluation phase* in which the operation specified by a particular directory request (e.g., a Read operation) is actually performed;

c) the *Results Merging phase* in which the results of a specified operation are returned to the requesting DUA or LDAP client. If a chaining mode of interaction was chosen, the Results Merging phase may involve several DSAs, each of which chained the original request or subrequest (as defined in clause 15.3.1 – Request decomposition) to other DSAs and LDAP servers during either or both of the preceding phases.

In the case of a Read, Compare, List, Search, Modify Entry, Modify DN, Remove Entry, LDAP Transport or Linked LDAP operation, name resolution takes place on the object name provided in the argument of the operation. In the case of an Add Entry operation, name resolution's target entry is the immediately superior entry of that provided in the argument of the operation – it can be easily derived by removing the final RDN from the name provided in the operation argument. (This is done via local argument **m** in the **FindDSE** procedure of clause 18.3.1.)

An operation on a particular entry may initially be directed at any DSA in a distributed directory. That DSA uses its knowledge, possibly in conjunction with other Directory servers, to process the operation through the three phases.

#### 15.2.1 Name Resolution phase

Name Resolution is the process of sequentially matching each RDN in a purported Name to an arc (or vertex) of the DIT, beginning logically at the Root and progressing downwards in the DIT. However, because the DIT is distributed among arbitrarily many Directory servers, each DSA may only be able to perform a fraction of the Name Resolution process. A given DSA performs its part of the Name Resolution process by traversing its local DSA information tree. This process is described in clause 18 and the accompanying diagrams (see Figures 9 to 12). Based on its local DSA information tree,

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

and the knowledge information contained therein, a DSA is able to infer whether the resolution can be continued by one or more other DSAs/LDAP servers, or whether the name is erroneous.

NOTE – Name Resolution within an LDAP server is outside the scope of this Directory Specification.

The Name Resolution phase is constrained to work within a DSA information tree if the **manageDSAIT** service control option is set.

### 15.2.2 Evaluation phase

When the Name Resolution phase has completed, the actual operation required (e.g., Read or Search) is performed.

Operations that involve a single entry interrogation – Read, Search with **subset** set to **baseObject** and Compare – may be carried out entirely within the DSA or LDAP server in which the entry is located.

Operations that involve multiple entries interrogation – List, Search and an LDAP search carried by an LDAP Transport operation – need to locate subordinates of the target, which may or may not reside in the same DSA. If they do not all reside in the same DSA, operations need to be directed to the DSAs specified in the subordinate, non-specific subordinate, supplier, or master references (as appropriate) and/or LDAP servers specified in subordinate, non-specific subordinate or cross references to complete the evaluation process.

The Evaluation phase is constrained to work within a DSA information tree if the **manageDSAIT** service control option is set. Likewise, if the evaluation phase starts within a service-specific administrative area, the evaluation is constrained to that administrative area.

### 15.2.3 Results Merging phase

The Results Merging phase is entered once some of the results of the Evaluation phase are available.

In those cases where the operation affected only a single entry, the result of the operation can simply be returned to the requesting DUA or LDAP client. In those cases where the operation has affected multiple entries on multiple Directory servers, results can be combined. If signing is performed on the results of an operation initiated by a DUA, the results shall not be combined. The results should be returned to the DUA without performing merging.

The permissible responses returned to a requester after results merging include:

a) a complete result of the operation;

b) a result which is not complete because some parts of the DIT remain unexplored (applies to List, Search and LDAP search only). Such a *partial result* may include continuation references for those parts of the DIT not explored;

c) an error (a referral being a special case); and

d) if the requester was a DSA, a **ChainingResults** value.

## 15.3 Managing Distributed Operations

Information is included in the argument of each operation which a DSA may be asked to perform indicating the progress of each operation as it traverses various DSAs of the Directory. This makes it possible for each DSA to perform the appropriate aspect of the processing required, and to record the completion of that aspect before directing the operation outward toward further directory servers.

Additional procedures are included in the DSA to physically distribute the operations and support other needs arising from their distribution.

### 15.3.1 Request decomposition

Request decomposition is a process performed internally by a DSA prior to communication with one or more other directory servers. A request is decomposed into several subrequests such that each of the latter accomplishes a part of the original task. Request decomposition can be used, for example, in the search operation, after the base object has been found. After decomposition, each of the subrequests may then be uni-chained or multi-chained to other directory servers, to continue the task.

The **argument** of a chained request (see clause 12.1) or subrequest shall be the unmodified operation argument of the original DAP operation. A DSA receiving a chained request shall not change **argument** of the DAP request when doing request decomposition.

NOTE – The following subclauses specify this requirement for individual components of **argument**. This should not be interpreted to mean that the component not explicitly mentioned can be changed.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

### 15.3.2 DSA as Request Responder

A DSA that receives a request can check the progress of that request as follows:

- if the request is received from a DUA then using the **operationProgress** component of the **CommonArguments** value, or its default value;
- if the request is received from an LDAP client then assume that Name Resolution has not started; or
- if the request is received from a DSA then using the **operationProgress** component of the **ChainingArguments** value, or its default value.

This will determine whether the operation is still in the Name Resolution phase or has reached the evaluation phase, and what portion of the operation the DSA should attempt to satisfy. If the DSA cannot fully satisfy the request, it shall either pass (by uni-chaining or multi-chaining) the operation on to one or more Directory servers which can help to fulfil the request, or return a referral to another DSA or LDAP server, or terminate the request with an error.

### 15.3.3 Completion of operations

Each DSA that has initiated an operation or propagated an operation to one or more other Directory servers shall keep track of that operation's existence until each of the other Directory servers has returned a result or error, or the operation's maximum time limit has expired. This requirement applies to all operations, propagation modes and processing phases, except when an LDAP abandon request has been sent to an LDAP server (see clause 20.6.5). It ensures the orderly closing down of distributed operations that have propagated out into the Directory.

## 15.4 Loop handling

The DIT may be in a state that can cause looping. As an example, looping can occur during name resolution where dereferencing one or more aliases brings the resolution back to the same branch of the DIT. Another potential cause of looping is through misconfigured knowledge references.

Within the context of a particular directory operation, a loop occurs if at any time the operation returns to a previous *state*, where state is defined by the following components:

- the name of the DSA currently processing the operation;
- the name of the **targetObject** as contained within the **ChainingArguments** value of the operation;
- the **operationProgress** as contained within the **ChainingArguments** value of the operation and as defined in clause 10.5.

This does not mean that an operation cannot be processed multiple times by a particular DSA. However, it does mean that the DSA will not process the same operation in the same state multiple times.

Looping is controlled using the **TraceInformation** value of the **ChainingArguments** value as defined in clause 10.6, which records the sequence of states a particular operation has gone through. Two strategies are defined to determine whether looping has occurred, or is about to occur. These are loop detection and loop avoidance, and they are described in clauses 15.4.1 and 15.4.2, respectively.

Loop detection is mandatory and loop avoidance is optional.

NOTE – An LDAP server cannot perform loop detection. Loop detection has to be performed by the DSA acting as LDAP requester. Detection of a loop may be delayed if more than one LDAP requester is forwarding requests to the same LDAP server for the same operation.

### 15.4.1 Loop detection

On receipt of a directory operation, a DSA shall initially validate the operation to ensure that it can be progressed. An important task of validation is to check for loops, by determining whether the current state of the operation appears in the sequence of previous states recorded in the **TraceInformation** value for that operation. This step of loop checking is loop detection.

### 15.4.2 Loop avoidance

Loop avoidance requires that a DSA, immediately prior to forwarding an operation to another DSA as part of a chaining procedure, determines whether the consequential state of the operation (which is the **TraceItem** value that the receiving DSA will add to the **TraceInformation** value when it receives it) appears on the sequence of previous states recorded in the **traceInformation** argument for the original incoming operation.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

In the case where referrals are received or acted upon, loop avoidance and loop detection cannot be achieved purely by examining **traceInformation**. In this case, each time a DSA acts on a referral, it needs to store the consequential state of the operation (i.e., the **TraceItem** value that the receiving DSA is going to add when it receives the request) along with a record of the incoming request. Before acting on or returning a referral, a DSA needs to check through this list, in order to check that an identical request has not been previously sent whilst trying to service the incoming operation.

## 15.5 Other considerations for distributed operation

### 15.5.1 Service controls

Some service controls need special consideration in the distributed environment in order that the operation is processed the way that was requested.

a) **chainingProhibited** – A DSA consults this service control when determining the mode of propagation of an operation. If it is set, then the DSA shall always use referral mode. If, however, it is not set, the DSA can choose whether to use chaining or referral depending on its capabilities.

b) **timeLimit** – A DSA shall take account of this service control to ensure that the time limit is not exceeded in that DSA. A DSA requested to perform an operation by a DUA, initially heeds the **timeLimit** expressed by the DUA as the available elapsed time in seconds for completion of the operation. If chaining is required, the **timeLimit** is included in the chaining arguments to be passed to the next DSA(s). In this case, the same value of the limit is used for each chained request, and is the (UTC) time by which the operation shall complete to meet the originally specified constraint. On receiving **ChainingArguments** with a **timeLimit** specified, the receiving DSA respects this limit.

c) **sizeLimit** – A DSA shall take account of this service control for List, Search and LDAP Search operations to ensure that the list of results does not exceed the size specified. The limit, as included in the common argument of the original request, is conveyed unchanged as the request is chained. If request decomposition is required, the same value is included in the argument to be passed to the next DSA, the full limit is used for each subrequest. When the results are returned, the requester DSA resolves the multiple results and applies the limit to the total to ensure that only the requested number is returned. If the limit had been exceeded, this is indicated in the reply.

d) **priority** – In all modes of propagation, each DSA is responsible for ensuring that the processing of operations is ordered so as to support this service control, if present.

e) **localScope** – The operation is limited to a locally defined scope and each DSA shall not propagate the request outside of this.

f) **scopeOfReferral** – If the DSA returns a referral or partial result to a List or Search operation, then the embedded continuation references shall be within the requested scope.

All other service controls need to be respected, but their use does not require any special consideration in the distributed environment.

### 15.5.2 Extensions

If a DSA encounters an extended operation in the Name Resolution phase of processing and determines that the operation should be chained to one or more DSAs, it shall include unchanged in the chained operation any extensions present.

NOTE – An administrative authority may determine that it is appropriate to return a **serviceError** with problem **unwillingToPerform** if it does not wish to propagate an extension.

If a DSA encounters an extension it does not support in the evaluation phase of processing, two possibilities may arise. If the extension is not critical, the DSA shall ignore the extension. If the extension is critical, the DSA shall return a **serviceError** with problem **unavailableCriticalExtension**. A critical extension to a multiple object operation may result in both results and service errors of this variety. A DSA merging such results and errors shall discard these service errors and employ the **unavailableCriticalExtension** component of **PartialOutcomeQualifier** as described in Rec. ITU-T X.511 | ISO/IEC 9594-3.

If an LDAP requester is about to chain a request initiated by a DUA to an LDAP server and it does not have knowledge of an LDAP control corresponding to each critical extension, it shall return an **unavailableCriticalExtensions** as specified above. However, if this is not the case and there are no other reasons for not chaining the request, it shall in the chained request add all known LDAP controls corresponding to the extensions signalled in the received request.


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

### 15.5.3 Alias dereferencing

Alias dereferencing is the process of creating a new target object name, by replacing the alias entry distinguished name part of the original target object name with the **AliasedEntryName** attribute value from the alias entry. The **object** name in the operation shall not be affected by alias dereferencing.

### 15.5.4 Paged results

When a DUA includes the **PagedResultsRequest** in the **search** or **list** request (see clause 7.9 of Rec. ITU-T X.511 | ISO/IEC 9594-3), the paging may be performed by the bound DSA, or it can be performed by the DSA that holds the **baseObject**/**object** entry of the **search** or **list** request (possible after one or more alias dereferencings), also called the *initial performer*. If the paging is performed by the bound DSA, which could also be the initial performer, the paging is called *bound-DSA paged results*. If the paging is performed by the initial performer, and the initial performer is different from the bound DSA, then the paging is called *DSP paged results*.

A DSA that supports DSP paged results shall:

- support DSA-bound paged results;
- support DSP paged results as bound DSA;
- support DSP paged results as an initial performer; and
- support the **entryCount** subcomponent of the **PartialOutcomeQualifier**.

When a bound DSA receives a **search** or **list** request with the **PagedResultsRequest** included, and the bound DSA is not the initial performer for that request, then the bound DSA may elect to include the **dspPaging** component in the **ChainingArguments** value. The initial performer may elect to do DSP paged results. This is signalled to the bound DSA by including a **queryReference** in the **PartialOutcomeQualifier**. This is the **queryReference** returned to the DUA to be used for retrieval of the next page.

If the initial performer either does not support DSP paged results or chooses not to perform it, the bound DSA may perform normal bound-DSA paging.

A DSA that is a performer, but is not the initial performer, shall ignore a possible **dspPaging** component in the **ChainingArguments** value, and it shall honour the **sizeLimit** service control if present.

### 15.5.5 Handling requests from LDAP client

When an LDAP client sends a request into a boundary DSA, the request is formatted according to IETF RFC 4511. How the request is handled by the boundary DSA depends on the initial destination of the request:

a) If the bound DSA and/or the LDAP servers for which the bound DSA is an LDAP responder complete the operation, then the procedure is outside the scope of this Directory specification.

b) If the request is initially handled as in a) but it becomes necessary to chain one or more subrequests to other DSAs, then such subrequests shall be chained wrapped as LDAP Transport operations.

c) If the original request is chained to another DSA, it shall be chained wrapped as an LDAP Transport operation.

## 15.6 Authentication of Distributed operations

Users of the Directory together with the administrative authorities that provide directory services may, at their discretion, require that directory operations be authenticated. For any particular directory operation, the nature of the authentication process will depend upon the security policy in force.

Two sets of authentication procedures are available which collectively enable a range of authentication requirements to be met. One set of procedures are those provided by the Bind operation: these facilitate authentication between two directory application-entities for the purposes of establishing an application-association. The Bind procedures accommodate a range of authentication exchanges from a simple exchange of identities to strong authentication.

In addition to the peer entity authentication of an association as provided by the Bind operation, additional procedures are defined within the directory to enable individual operations to be authenticated. Two distinct sets of directory authentication procedures are defined. One facilitates requester authentication services, which address the authentication, by a DSA, of the requester of the original service request. The second set facilitates results authentication services which address the authentication, by a requester, of any results that are returned.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

For requester authentication, two procedures are defined, one based upon a simple exchange of identities, termed **identity based authentication**, and one based upon digital signature techniques, termed **signature based authentication**. The former of these procedures is rudimentary in nature since the identity exchange is based upon the exchange of distinguished names which are transmitted in the clear.

For authentication of results a single **results authentication** procedure is defined, based upon digital signature techniques; due to the generally complex nature of results collation, a simpler, identity-based procedure is not defined.

Authentication of error responses may be supported by these procedures.

The services described below are to be considered as augmenting those provided by the Bind service; Bind procedures are assumed to have been effected successfully prior to authentication of directory operations.

The procedures to be effected by a DSA in providing requester and results authentication are specified in clause 22.

# 16 The Operation Dispatcher

The **Operation Dispatcher** is the main controlling procedure in a DSA. It guides each operation through the three phases of processing a request. The **Operation Dispatcher** therefore makes use of a set of procedures to fully process the request as shown in Figure 6.

## 16.1 General concepts

### 16.1.1 Procedures

Each of the procedures employed by the **Operation Dispatcher** consists of a definition of its conceptual interface in terms of its parameters, i.e., arguments, results and errors, and a description of the procedure steps themselves. The behaviour of the procedures is described by flowcharts and text. Within a flow chart, the used symbols have the semantics shown in Figure 7.

### 16.1.2 Use of common data structures

All procedures make use of some data structures that are available during the processing of an operation within the **Operation Dispatcher**. These data structures serve to coordinate the data flow within the **Operation Dispatcher**. Most of these structures are directly associated with the argument of the operation and the result to be created for the operation. Components of the argument and result are referred to using their names within the associated ASN.1 definition (e.g., the **`operationProgress`** component of the chaining arguments). If any of these structures is a compound structure, a component of this structure may be referred to as **`compound.component`** (e.g., **`operationProgress.nameResolutionPhase`**).


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

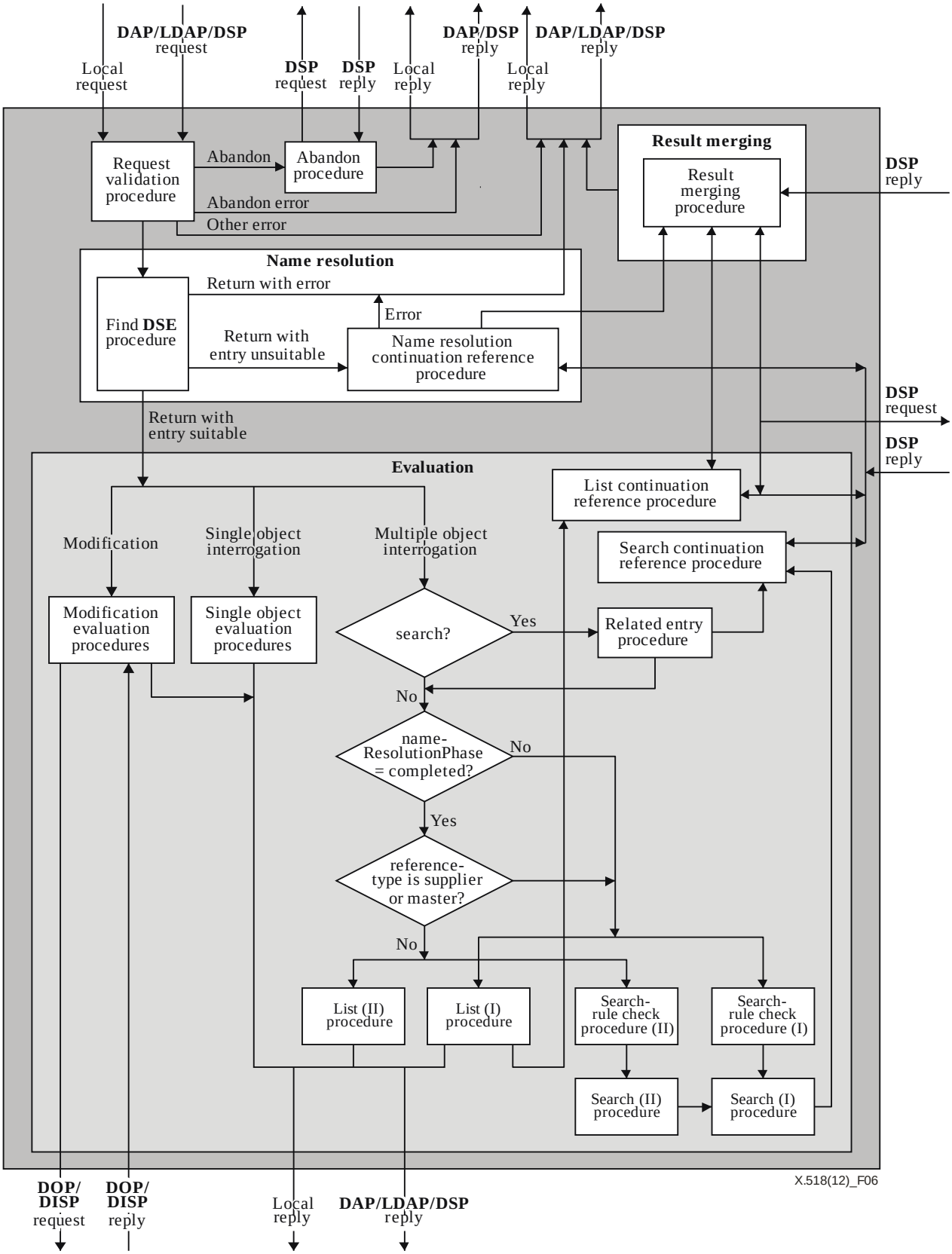


**Figure 6 – Operation Dispatcher**

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

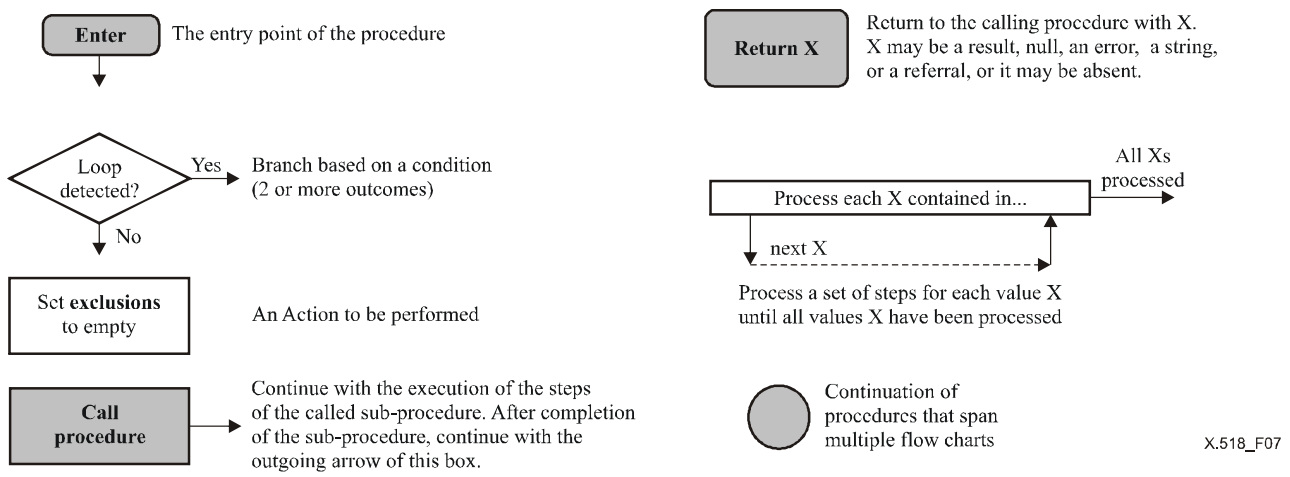


**Figure 7 – Symbols used in flow charts**

The following data structures are defined within the **Operation Dispatcher**:

- **NRcontinuationList** – A list of continuation references created for use in the **Name Resolution Continuation Reference** procedure.
- **SRcontinuationList** – A list of continuation references created for use in the **List** or **Search Continuation Reference** procedure.
- **admPoints** – A list of references to DSEs of type administrative point that is collected during Name Resolution.
- **referralRequests** – A list of the requests or subrequests which have been chained as a result of executing referrals. Each such request/subrequest is summarised in the form of a **`TraceItem`**. This list is used by the Loop Avoidance procedure of clause 15.4.2.
- **emptyHierarchySelect** – A Boolean type variable that can be set in the **Hierarchy Selection** procedure. The variable is assumed to be reset when entering **Hierarchy Selection** procedure the first time during a Search operation.

Furthermore, a procedure may use a set of locally defined variables.

### 16.1.3 Errors

At each stage of the processing, an error may be detected during the execution of any sub-procedure. The error identified within this sub-procedure is normally returned to the requester as a corresponding protocol error. In this case, the **Operation Dispatcher** is terminated immediately. In the case that multiple errors are received, one shall be selected to be returned (see clause 13.1 of Rec. ITU-T X.511 | ISO/IEC 9594-3).

Alternatively, a procedure may choose to process errors (e.g., if a **`serviceError`** with problem **`busy`** is returned to a chained search subrequest) at certain points of operation processing. In this case, the procedure continues with its execution and no error is returned to the requester.

The conditions under which a DSA may sign the errors returned are specified in clause 13 of Rec. ITU-T X.511 | ISO/IEC 9594-3.

### 16.1.4 Asynchronous events

During the processing of an operation request within the **Operation Dispatcher**, several asynchronous events may occur. The following subclauses specify how to handle an exceeded time limit or size limit or administrative limit, a loss of association and an **`abandon`** request for an operation that is being processed. The handling of all other asynchronous events, e.g., local policy decisions, etc., is outside the scope of this Directory Specification.

#### 16.1.4.1 Time limit

A **`timeLimit`**, as specified in the **`CommonArguments`** value, can expire at any point in time during the operation. In this case, normally a **`serviceError`** with problem **`timeLimitExceeded`** is returned to the requesting DUA or DSA and the **Operation Dispatcher** is terminated. If the requester is an LDAP client, the bound DSA shall convert the error to a **`timeLimitExceeded`** LDAP result code. Alternatively, a procedure may choose to handle this event in a different way (e.g., during processing of a **`search`** request).


© ISO/IEC 2020 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020

If a DSA receives a request from another DSA with the time limit exceeded, it shall send a **serviceError** with problem **timeLimitExceeded** without any further processing of the request.

If a DSA has outstanding (sub)requests, when the **timeLimit** expires, and there are no results available, it shall return a **serviceError** with problem **timeLimitExceeded** to the requester.

If the request was not an **ldapTransport** request, and if a DSA has outstanding subrequests, when the **timeLimit** expires, and there are results available, it shall return a result to the requester with the following contents:

a) all the collected results, up to the **timeLimit** expiring;

b) the **limitProblem** subcomponent of the **partialOutcomeQualifier** component shall be set to **timeLimitExceeded**; and

c) the **unexplored** subcomponent of the **partialOutcomeQualifier** component shall contain a continuation reference value for each set of directory servers to which subrequests were sent but the result of which is not included in the result to the requester, in addition to continuation references to DSAs to which this DSA did not attempt to send subrequests.

If the request was an **ldapTransport** request, and if a DSA has outstanding subrequests, when the **timeLimit** expires:

a) if there are retained results, these results shall be returned in a **linkedLDAP** request;

b) if the embedded LDAP request in the **ldapTransport** request was a **SearchRequest**, then return an **SearchResultDone** with **resultCode** equal to **timeLimitExceeded** embedded in an **ldapTransport** result;

c) if the embedded LDAP request in the **ldapTransport** request was not a **SearchRequest**, then return a result for the type operation with **resultCode** set to **timeLimitExceeded** embedded in an **ldapTransport** result.

#### 16.1.4.2 Loss of an application-association

If the application-association to the requester is lost, the possibility of returning results is lost. The DSA may optionally for each outstanding interrogation (sub)request to DSAs send a **chainedAbandon** request, unless the application-association to the DSA in question has also been lost. All replies to such **chainedAbandon** requests and all replies to outstanding (sub)requests shall be discarded. In the case of DSP paged results, the bound DSA should cancel outstanding paged results by generating a new paged result request by taking the **abandonQuery** alternative of the **PagedResultsRequest** value. The DSA may also for each outstanding interrogation (sub)request to LDAP servers send an LDAP AbandonRequest and all results to outstanding (sub)requests shall be discarded.

If the application-association to one of the outstanding chained subrequests is lost and the application-association with the requester is not lost, the DSA may, for interrogation operations only, optionally try any alternative reference to another directory server of the same type that is able to process the chained request (e.g., a reference to a shadow DSA, after loss of the association to the master DSA). If this does not succeed, the DSA shall act as follows:

1) If **operationProgress.nameResolution** is set to **notStarted** or **proceeding**, return either:

   – if the reply is to be returned to a DUA (or DSA), a **serviceError** with problem **unavailable** to the requester or a referral error whose continuation reference contains the set of directory servers that are able to continue the operation.

   – if the reply is to be returned to an LDAP client, an LDAP result with **resultCode** equal to **unavailable** or with **resultCode** set to **referral** and with referrals (URIs) to the set of directory servers that are able to continue the operation.

   If non-specific subordinate references are used during the Name Resolution phase and not all the application-associations in question are lost, optionally attempt to do the name resolution without the directory servers to which the associations are lost. If this fails:

   – if the reply is to be returned to a DUA (or DSA), return either a **serviceError** with problem **unavailable**, or a referral error containing the complete set of NSSRs.

   – if the reply is to be returned to an LDAP client, return either an LDAP result with **resultCode** set to **unavailable** or with **resultCode** equal to **referral** and with referrals (URIs) corresponding to the complete set of NSSRs.

   If the DSA using local knowledge knows, possibly reflected in the appropriate **MasterOrShadowAccessPoint** value, that chaining is required to a DSA to which an application-association is lost, it shall elect to send a **serviceError** with problem **unavailable**. The **notification** component of the **CommonResults** data type may be included and shall then contain:

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 9594-4_2020