Network Working Group
Request for Comments: 1777
Obsoletes: 1487
Category: Standards Track

W. Yeong
Performance Systems International
T. Howes
University of Michigan
S. Kille
ISODE Consortium

March 1995

Lightweight Directory Access Protocol

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

The protocol described in this document is designed to provide access to the X.500 Directory while not incurring the resource requirements of the Directory Access Protocol (DAP). This protocol is specifically targeted at simple management applications and browser applications that provide simple read/write interactive access to the X.500 Directory, and is intended to be a complement to the DAP itself.

Key aspects of LDAP are:

- Protocol elements are carried directly over TCP or other transport, bypassing much of the session/presentation overhead.

- Many protocol data elements are encoding as ordinary strings (e.g., Distinguished Names).

- A lightweight BER encoding is used to encode all protocol elements.

1. History

The tremendous interest in X.500 [1,2] technology in the Internet has lead to efforts to reduce the high "cost of entry" associated with use of the technology, such as the Directory Assistance Service [3] and DIXIE [4]. While efforts such as these have met with success, they have been solutions based on particular implementations and as such have limited applicability. This document continues the efforts to define Directory protocol alternatives but departs from previous efforts in that it consciously avoids dependence on particular implementations.

2. Protocol Model

The general model adopted by this protocol is one of clients performing protocol operations against servers. In this model, this is accomplished by a client transmitting a protocol request describing the operation to be performed to a server, which is then responsible for performing the necessary operations on the Directory. Upon completion of the necessary operations, the server returns a response containing any results or errors to the requesting client. In keeping with the goal of easing the costs associated with use of the Directory, it is an objective of this protocol to minimize the complexity of clients so as to facilitate widespread deployment of applications capable of utilizing the Directory.

Note that, although servers are required to return responses whenever such responses are defined in the protocol, there is no requirement for synchronous behavior on the part of either client or server implementations: requests and responses for multiple operations may be exchanged by client and servers in any order, as long as clients eventually receive a response for every request that requires one.

Consistent with the model of servers performing protocol operations on behalf of clients, it is also to be noted that protocol servers are expected to handle referrals without resorting to the return of such referrals to the client. This protocol makes no provisions for the return of referrals to clients, as the model is one of servers ensuring the performance of all necessary operations in the Directory, with only final results or errors being returned by servers to clients.

Note that this protocol can be mapped to a strict subset of the directory abstract service, so it can be cleanly provided by the DAP.

3. Mapping Onto Transport Services

This protocol is designed to run over connection-oriented, reliable transports, with all 8 bits in an octet being significant in the data stream. Specifications for two underlying services are defined here, though others are also possible.

3.1. Transmission Control Protocol (TCP)

The LDAPMessage PDUs are mapped directly onto the TCP bytestream. Server implementations running over the TCP should provide a protocol listener on port 389.

3.2. Connection Oriented Transport Service (COTS)

The connection is established. No special use of T-Connect is made. Each LDAPMessage PDU is mapped directly onto T-Data.

4. Elements of Protocol

For the purposes of protocol exchanges, all protocol operations are encapsulated in a common envelope, the LDAPMessage, which is defined as follows:

```
  LDAPMessage ::=
      SEQUENCE {
           messageID      MessageID,
           protocolOp     CHOICE {
                               bindRequest         BindRequest,
                               bindResponse        BindResponse,
                               unbindRequest       UnbindRequest,
                               searchRequest       SearchRequest,
                               searchResponse      SearchResponse,
                               modifyRequest       ModifyRequest,
                               modifyResponse      ModifyResponse,
                               addRequest          AddRequest,
                               addResponse         AddResponse,
                               delRequest          DelRequest,
                               delResponse         DelResponse,
                               modifyRDNRequest    ModifyRDNRequest,
                               modifyRDNResponse   ModifyRDNResponse,
                               compareDNRequest    CompareRequest,
                               compareDNResponse   CompareResponse,
                               abandonRequest      AbandonRequest
                          }
      }

  MessageID ::= INTEGER (0 .. maxInt)
```

The function of the LDAPMessage is to provide an envelope containing common fields required in all protocol exchanges. At this time the only common field is a message ID, which is required to have a value different from the values of any other requests outstanding in the LDAP session of which this message is a part.

The message ID value must be echoed in all LDAPMessage envelopes encapsulting responses corresponding to the request contained in the LDAPMessage in which the message ID value was originally used.

In addition to the LDAPMessage defined above, the following definitions are also used in defining protocol operations:

```
  LDAPString ::= OCTET STRING
```

The LDAPString is a notational convenience to indicate that, although strings of LDAPString type encode as OCTET STRING types, the legal character set in such strings is limited to the IA5 character set.

```
  LDAPDN ::= LDAPString

  RelativeLDAPDN ::= LDAPString
```

An LDAPDN and a RelativeLDAPDN are respectively defined to be the representation of a Distinguished Name and a Relative Distinguished Name after encoding according to the specification in [5], such that

```
  <distinguished-name> ::= <name>

  <relative-distinguished-name> ::= <name-component>
```

where <name> and <name-component> are as defined in [5].

```
  AttributeValueAssertion ::=
      SEQUENCE {
           attributeType       AttributeType,
           attributeValue      AttributeValue
      }
```

The AttributeValueAssertion type definition is similar to the one in the X.500 Directory standards.

```
  AttributeType ::= LDAPString

  AttributeValue ::= OCTET STRING
```

An AttributeType value takes on as its value the textual string associated with that AttributeType in the X.500 Directory standards. For example, the AttributeType ’organizationName’ with object identifier 2.5.4.10 is represented as an AttributeType in this protocol by the string "organizationName". In the event that a protocol implementation encounters an Attribute Type with which it cannot associate a textual string, an ASCII string encoding of the object identifier associated with the Attribute Type may be subsitituted. For example, the organizationName AttributeType may be represented by the ASCII string "2.5.4.10" if a protocol implementation is unable to associate the string "organizationName" with it.

A field of type AttributeValue takes on as its value an octet string encoding of a Directory AttributeValue type. The definition of these string encodings for different Directory AttributeValue types may be found in companions to this document that define the encodings of various attribute syntaxes such as [6].

```
     LDAPResult ::=
         SEQUENCE {
             resultCode    ENUMERATED {
                             success                      (0),
                             operationsError              (1),
                             protocolError                (2),
                             timeLimitExceeded            (3),
                             sizeLimitExceeded            (4),
                             compareFalse                 (5),
                             compareTrue                  (6),
                             authMethodNotSupported       (7),
                             strongAuthRequired           (8),
                             noSuchAttribute              (16),
                             undefinedAttributeType       (17),
                             inappropriateMatching        (18),
                             constraintViolation          (19),
                             attributeOrValueExists       (20),
                             invalidAttributeSyntax       (21),
                             noSuchObject                 (32),
                             aliasProblem                 (33),
                             invalidDNSyntax              (34),
                             isLeaf                       (35),
                             aliasDereferencingProblem    (36),
                             inappropriateAuthentication  (48),
                             invalidCredentials           (49),
                             insufficientAccessRights     (50),
                             busy                         (51),
                             unavailable                  (52),
                             unwillingToPerform           (53),
                             loopDetect                   (54),
                             namingViolation              (64),
                             objectClassViolation         (65),
                             notAllowedOnNonLeaf          (66),
                             notAllowedOnRDN              (67),
                             entryAlreadyExists           (68),
                             objectClassModsProhibited    (69),
                             other                        (80)
                           },
             matchedDN     LDAPDN,
             errorMessage  LDAPString
         }
```

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.