Network Working Group
Request for Comments: 4513
Obsoletes: 2251, 2829, 2830
Category: Standards Track

R. Harrison, Ed.
Novell, Inc.

June 2006

Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms

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.

Copyright Notice

Abstract

This document describes authentication methods and security mechanisms of the Lightweight Directory Access Protocol (LDAP). This document details establishment of Transport Layer Security (TLS) using the StartTLS operation.

This document details the simple Bind authentication method including anonymous, unauthenticated, and name/password mechanisms and the Simple Authentication and Security Layer (SASL) Bind authentication method including the EXTERNAL mechanism.

This document discusses various authentication and authorization states through which a session to an LDAP server may pass and the actions that trigger these state changes.

This document, together with other documents in the LDAP Technical Specification (see Section 1 of the specification's road map), obsoletes RFC 2251, RFC 2829, and RFC 2830.

Table of Contents

1. Introduction

The Lightweight Directory Access Protocol (LDAP) [RFC4510] is a powerful protocol for accessing directories. It offers means of searching, retrieving, and manipulating directory content and ways to access a rich set of security functions.

It is vital that these security functions be interoperable among all LDAP clients and servers on the Internet; therefore there has to be a minimum subset of security functions that is common to all implementations that claim LDAP conformance.

Basic threats to an LDAP directory service include (but are not limited to):

(1) Unauthorized access to directory data via data-retrieval operations.

(2) Unauthorized access to directory data by monitoring access of others.

(3) Unauthorized access to reusable client authentication information by monitoring access of others.

(4) Unauthorized modification of directory data.

(5) Unauthorized modification of configuration information.

(6) Denial of Service: Use of resources (commonly in excess) in a manner intended to deny service to others.

(7) Spoofing: Tricking a user or client into believing that information came from the directory when in fact it did not, either by modifying data in transit or misdirecting the client's transport connection. Tricking a user or client into sending privileged information to a hostile entity that appears to be the directory server but is not. Tricking a directory server into believing that information came from a particular client when in fact it came from a hostile entity.

(8) Hijacking: An attacker seizes control of an established protocol session.

Threats (1), (4), (5), (6), (7), and (8) are active attacks. Threats (2) and (3) are passive attacks.

Threats (1), (4), (5), and (6) are due to hostile clients. Threats (2), (3), (7), and (8) are due to hostile agents on the path between client and server or hostile agents posing as a server, e.g., IP spoofing.

LDAP offers the following security mechanisms:

(1) Authentication by means of the Bind operation. The Bind operation provides a simple method that supports anonymous, unauthenticated, and name/password mechanisms, and the Simple Authentication and Security Layer (SASL) method, which supports a wide variety of authentication mechanisms.

(2) Mechanisms to support vendor-specific access control facilities (LDAP does not offer a standard access control facility).

(3) Data integrity service by means of security layers in Transport Layer Security (TLS) or SASL mechanisms.

(4) Data confidentiality service by means of security layers in TLS or SASL mechanisms.

(5) Server resource usage limitation by means of administrative limits configured on the server.

(6) Server authentication by means of the TLS protocol or SASL mechanisms.

LDAP may also be protected by means outside the LDAP protocol, e.g., with IP layer security [RFC4301].

Experience has shown that simply allowing implementations to pick and choose the security mechanisms that will be implemented is not a strategy that leads to interoperability. In the absence of mandates, clients will continue to be written that do not support any security function supported by the server, or worse, they will only support mechanisms that provide inadequate security for most circumstances.

It is desirable to allow clients to authenticate using a variety of mechanisms including mechanisms where identities are represented as distinguished names [X.501][RFC4512], in string form [RFC4514], or as used in different systems (e.g., simple user names [RFC4013]). Because some authentication mechanisms transmit credentials in plain text form, and/or do not provide data security services and/or are subject to passive attacks, it is necessary to ensure secure interoperability by identifying a mandatory-to-implement mechanism for establishing transport-layer security services.

The set of security mechanisms provided in LDAP and described in this document is intended to meet the security needs for a wide range of deployment scenarios and still provide a high degree of interoperability among various LDAP implementations and deployments.

1.1. Relationship to Other Documents

This document is an integral part of the LDAP Technical Specification [RFC4510].

This document, together with [RFC4510], [RFC4511], and [RFC4512], obsoletes RFC 2251 in its entirety. Sections 4.2.1 (portions) and 4.2.2 of RFC 2251 are obsoleted by this document. Appendix B.1 summarizes the substantive changes made to RFC 2251 by this document.

This document obsoletes RFC 2829 in its entirety. Appendix B.2 summarizes the substantive changes made to RFC 2829 by this document.

Sections 2 and 4 of RFC 2830 are obsoleted by [RFC4511]. The remainder of RFC 2830 is obsoleted by this document. Appendix B.3 summarizes the substantive changes made to RFC 2830 by this document.

1.2. Conventions

The key words "MUST", "MUST NOT", "SHALL", "SHOULD", "SHOULD NOT", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

The term "user" represents any human or application entity that is accessing the directory using a directory client. A directory client (or client) is also known as a directory user agent (DUA).

The term "transport connection" refers to the underlying transport services used to carry the protocol exchange, as well as associations established by these services.

The term "TLS layer" refers to TLS services used in providing security services, as well as associations established by these services.

The term "SASL layer" refers to SASL services used in providing security services, as well as associations established by these services.

The term "LDAP message layer" refers to the LDAP Message (PDU) services used in providing directory services, as well as associations established by these services.

The term "LDAP session" refers to combined services (transport connection, TLS layer, SASL layer, LDAP message layer) and their associations.

In general, security terms in this document are used consistently with the definitions provided in [RFC2828]. In addition, several terms and concepts relating to security, authentication, and authorization are presented in Appendix A of this document. While the formal definition of these terms and concepts is outside the scope of this document, an understanding of them is prerequisite to understanding much of the material in this document. Readers who are unfamiliar with security-related concepts are encouraged to review Appendix A before reading the remainder of this document.

2. Implementation Requirements

LDAP server implementations MUST support the anonymous authentication mechanism of the simple Bind method (Section 5.1.1).

LDAP implementations that support any authentication mechanism other than the anonymous authentication mechanism of the simple Bind method MUST support the name/password authentication mechanism of the simple Bind method (Section 5.1.3) and MUST be capable of protecting this name/password authentication using TLS as established by the StartTLS operation (Section 3).

Implementations SHOULD disallow the use of the name/password authentication mechanism by default when suitable data security services are not in place, and they MAY provide other suitable data security services for use with this authentication mechanism.

Implementations MAY support additional authentication mechanisms. Some of these mechanisms are discussed below.

LDAP server implementations SHOULD support client assertion of authorization identity via the SASL EXTERNAL mechanism (Section 5.2.3).

LDAP server implementations that support no authentication mechanism other than the anonymous mechanism of the simple bind method SHOULD support use of TLS as established by the StartTLS operation (Section 3). (Other servers MUST support TLS per the second paragraph of this section.)

Implementations supporting TLS MUST support the TLS_RSA_WITH_3DES_EDE_CBC_SHA ciphersuite and SHOULD support the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA ciphersuite. Support for the latter ciphersuite is recommended to encourage interoperability with implementations conforming to earlier LDAP StartTLS specifications.

3. StartTLS Operation

The Start Transport Layer Security (StartTLS) operation defined in Section 4.14 of [RFC4511] provides the ability to establish TLS [RFC4346] in an LDAP session.

The goals of using the TLS protocol with LDAP are to ensure data confidentiality and integrity, and to optionally provide for authentication. TLS expressly provides these capabilities, although the authentication services of TLS are available to LDAP only in combination with the SASL EXTERNAL authentication method (see Section 5.2.3), and then only if the SASL EXTERNAL implementation chooses to make use of the TLS credentials.

3.1. TLS Establishment Procedures

This section describes the overall procedures clients and servers must follow for TLS establishment. These procedures take into consideration various aspects of the TLS layer including discovery of resultant security level and assertion of the client's authorization identity.

3.1.1. StartTLS Request Sequencing

A client may send the StartTLS extended request at any time after establishing an LDAP session, except:

- when TLS is currently established on the session,
- when a multi-stage SASL negotiation is in progress on the session, or
- when there are outstanding responses for operation requests previously issued on the session.

As described in [RFC4511], Section 4.14.1, a (detected) violation of any of these requirements results in a return of the operationsError resultCode.

Client implementers should ensure that they strictly follow these operation sequencing requirements to prevent interoperability issues. Operational experience has shown that violating these requirements

causes interoperability issues because there are race conditions that prevent servers from detecting some violations of these requirements due to factors such as server hardware speed and network latencies.

There is no general requirement that the client have or have not already performed a Bind operation (Section 5) before sending a StartTLS operation request; however, where a client intends to perform both a Bind operation and a StartTLS operation, it SHOULD first perform the StartTLS operation so that the Bind request and response messages are protected by the data security services established by the StartTLS operation.

3.1.2. Client Certificate

If an LDAP server requests or demands that a client provide a user certificate during TLS negotiation and the client does not present a suitable user certificate (e.g., one that can be validated), the server may use a local security policy to determine whether to successfully complete TLS negotiation.

If a client that has provided a suitable certificate subsequently performs a Bind operation using the SASL EXTERNAL authentication mechanism (Section 5.2.3), information in the certificate may be used by the server to identify and authenticate the client.

3.1.3. Server Identity Check

In order to prevent man-in-the-middle attacks, the client MUST verify the server's identity (as presented in the server's Certificate message). In this section, the client's understanding of the server's identity (typically the identity used to establish the transport connection) is called the "reference identity".

The client determines the type (e.g., DNS name or IP address) of the reference identity and performs a comparison between the reference identity and each subjectAltName value of the corresponding type until a match is produced. Once a match is produced, the server's identity has been verified, and the server identity check is complete. Different subjectAltName types are matched in different ways. Sections 3.1.3.1 - 3.1.3.3 explain how to compare values of various subjectAltName types.

The client may map the reference identity to a different type prior to performing a comparison. Mappings may be performed for all available subjectAltName types to which the reference identity can be mapped; however, the reference identity should only be mapped to types for which the mapping is either inherently secure (e.g., extracting the DNS name from a URI to compare with a subjectAltName

of type dNSName) or for which the mapping is performed in a secure manner (e.g., using DNSSEC, or using user- or admin-configured host-to-address/address-to-host lookup tables).

The server's identity may also be verified by comparing the reference identity to the Common Name (CN) [RFC4519] value in the leaf Relative Distinguished Name (RDN) of the subjectName field of the server's certificate. This comparison is performed using the rules for comparison of DNS names in Section 3.1.3.1, below, with the exception that no wildcard matching is allowed. Although the use of the Common Name value is existing practice, it is deprecated, and Certification Authorities are encouraged to provide subjectAltName values instead. Note that the TLS implementation may represent DNs in certificates according to X.500 or other conventions. For example, some X.500 implementations order the RDNs in a DN using a left-to-right (most significant to least significant) convention instead of LDAP's right-to-left convention.

If the server identity check fails, user-oriented clients SHOULD either notify the user (clients may give the user the opportunity to continue with the LDAP session in this case) or close the transport connection and indicate that the server's identity is suspect. Automated clients SHOULD close the transport connection and then return or log an error indicating that the server's identity is suspect or both.

Beyond the server identity check described in this section, clients should be prepared to do further checking to ensure that the server is authorized to provide the service it is requested to provide. The client may need to make use of local policy information in making this determination.

3.1.3.1. Comparison of DNS Names

If the reference identity is an internationalized domain name, conforming implementations MUST convert it to the ASCII Compatible Encoding (ACE) format as specified in Section 4 of RFC 3490 [RFC3490] before comparison with subjectAltName values of type dNSName. Specifically, conforming implementations MUST perform the conversion operation specified in Section 4 of RFC 3490 as follows:

* in step 1, the domain name SHALL be considered a "stored string";
* in step 3, set the flag called "UseSTD3ASCIIRules";
* in step 4, process each label with the "ToASCII" operation; and
* in step 5, change all label separators to U+002E (full stop).

After performing the "to-ASCII" conversion, the DNS labels and names MUST be compared for equality according to the rules specified in Section 3 of RFC3490.

The '*' (ASCII 42) wildcard character is allowed in subjectAltName values of type dNSName, and then only as the left-most (least significant) DNS label in that value. This wildcard matches any left-most DNS label in the server name. That is, the subject *.example.com matches the server names a.example.com and b.example.com, but does not match example.com or a.b.example.com.

3.1.3.2. Comparison of IP Addresses

When the reference identity is an IP address, the identity MUST be converted to the "network byte order" octet string representation [RFC791][RFC2460]. For IP Version 4, as specified in RFC 791, the octet string will contain exactly four octets. For IP Version 6, as specified in RFC 2460, the octet string will contain exactly sixteen octets. This octet string is then compared against subjectAltName values of type iPAddress. A match occurs if the reference identity octet string and value octet strings are identical.

3.1.3.3. Comparison of Other subjectName Types

Client implementations MAY support matching against subjectAltName values of other types as described in other documents.

3.1.4. Discovery of Resultant Security Level

After a TLS layer is established in an LDAP session, both parties are to each independently decide whether or not to continue based on local policy and the security level achieved. If either party decides that the security level is inadequate for it to continue, it SHOULD remove the TLS layer immediately after the TLS (re)negotiation has completed (see [RFC4511], Section 4.14.3, and Section 3.2 below). Implementations may reevaluate the security level at any time and, upon finding it inadequate, should remove the TLS layer.

3.1.5. Refresh of Server Capabilities Information

After a TLS layer is established in an LDAP session, the client SHOULD discard or refresh all information about the server that it obtained prior to the initiation of the TLS negotiation and that it did not obtain through secure mechanisms. This protects against man-in-the-middle attacks that may have altered any server capabilities information retrieved prior to TLS layer installation.

The server may advertise different capabilities after installing a TLS layer. In particular, the value of 'supportedSASLMechanisms' may be different after a TLS layer has been installed (specifically, the EXTERNAL and PLAIN [PLAIN] mechanisms are likely to be listed only after a TLS layer has been installed).

3.2. Effect of TLS on Authorization State

The establishment, change, and/or closure of TLS may cause the authorization state to move to a new state. This is discussed further in Section 4.

3.3. TLS Ciphersuites

Several issues should be considered when selecting TLS ciphersuites that are appropriate for use in a given circumstance. These issues include the following:

- The ciphersuite's ability to provide adequate confidentiality protection for passwords and other data sent over the transport connection. Client and server implementers should recognize that some TLS ciphersuites provide no confidentiality protection, while other ciphersuites that do provide confidentiality protection may be vulnerable to being cracked using brute force methods, especially in light of ever-increasing CPU speeds that reduce the time needed to successfully mount such attacks.

- Client and server implementers should carefully consider the value of the password or data being protected versus the level of confidentiality protection provided by the ciphersuite to ensure that the level of protection afforded by the ciphersuite is appropriate.

- The ciphersuite's vulnerability (or lack thereof) to man-in-the-middle attacks. Ciphersuites vulnerable to man-in-the-middle attacks SHOULD NOT be used to protect passwords or sensitive data, unless the network configuration is such that the danger of a man-in-the-middle attack is negligible.

- After a TLS negotiation (either initial or subsequent) is completed, both protocol peers should independently verify that the security services provided by the negotiated ciphersuite are adequate for the intended use of the LDAP session. If they are not, the TLS layer should be closed.

4. Authorization State

Every LDAP session has an associated authorization state. This state is comprised of numerous factors such as what (if any) authentication state has been established, how it was established, and what security services are in place. Some factors may be determined and/or affected by protocol events (e.g., Bind, StartTLS, or TLS closure), and some factors may be determined by external events (e.g., time of day or server load).

While it is often convenient to view authorization state in simplistic terms (as we often do in this technical specification) such as "an anonymous state", it is noted that authorization systems in LDAP implementations commonly involve many factors that interrelate in complex manners.

Authorization in LDAP is a local matter. One of the key factors in making authorization decisions is authorization identity. The Bind operation (defined in Section 4.2 of [RFC4511] and discussed further in Section 5 below) allows information to be exchanged between the client and server to establish an authorization identity for the LDAP session. The Bind operation may also be used to move the LDAP session to an anonymous authorization state (see Section 5.1.1).

Upon initial establishment of the LDAP session, the session has an anonymous authorization identity. Among other things this implies that the client need not send a BindRequest in the first PDU of the LDAP message layer. The client may send any operation request prior to performing a Bind operation, and the server MUST treat it as if it had been performed after an anonymous Bind operation (Section 5.1.1).

Upon receipt of a Bind request, the server immediately moves the session to an anonymous authorization state. If the Bind request is successful, the session is moved to the requested authentication state with its associated authorization state. Otherwise, the session remains in an anonymous state.

It is noted that other events both internal and external to LDAP may result in the authentication and authorization states being moved to an anonymous one. For instance, the establishment, change, or closure of data security services may result in a move to an anonymous state, or the user's credential information (e.g., certificate) may have expired. The former is an example of an event internal to LDAP, whereas the latter is an example of an event external to LDAP.

5. Bind Operation

The Bind operation ([RFC4511], Section 4.2) allows authentication information to be exchanged between the client and server to establish a new authorization state.

The Bind request typically specifies the desired authentication identity. Some Bind mechanisms also allow the client to specify the authorization identity. If the authorization identity is not specified, the server derives it from the authentication identity in an implementation-specific manner.

If the authorization identity is specified, the server MUST verify that the client's authentication identity is permitted to assume (e.g., proxy for) the asserted authorization identity. The server MUST reject the Bind operation with an invalidCredentials resultCode in the Bind response if the client is not so authorized.

5.1. Simple Authentication Method

The simple authentication method of the Bind Operation provides three authentication mechanisms:

- An anonymous authentication mechanism (Section 5.1.1).
- An unauthenticated authentication mechanism (Section 5.1.2).
- A name/password authentication mechanism using credentials consisting of a name (in the form of an LDAP distinguished name [RFC4514]) and a password (Section 5.1.3).

5.1.1. Anonymous Authentication Mechanism of Simple Bind

An LDAP client may use the anonymous authentication mechanism of the simple Bind method to explicitly establish an anonymous authorization state by sending a Bind request with a name value of zero length and specifying the simple authentication choice containing a password value of zero length.

5.1.2. Unauthenticated Authentication Mechanism of Simple Bind

An LDAP client may use the unauthenticated authentication mechanism of the simple Bind method to establish an anonymous authorization state by sending a Bind request with a name value (a distinguished name in LDAP string form [RFC4514] of non-zero length) and specifying the simple authentication choice containing a password value of zero length.

The distinguished name value provided by the client is intended to be used for trace (e.g., logging) purposes only. The value is not to be authenticated or otherwise validated (including verification that the DN refers to an existing directory object). The value is not to be used (directly or indirectly) for authorization purposes.

Unauthenticated Bind operations can have significant security issues (see Section 6.3.1). In particular, users intending to perform Name/Password Authentication may inadvertently provide an empty password and thus cause poorly implemented clients to request Unauthenticated access. Clients SHOULD be implemented to require user selection of the Unauthenticated Authentication Mechanism by means other than user input of an empty password. Clients SHOULD disallow an empty password input to a Name/Password Authentication user interface. Additionally, Servers SHOULD by default fail Unauthenticated Bind requests with a resultCode of unwillingToPerform.

5.1.3. Name/Password Authentication Mechanism of Simple Bind

An LDAP client may use the name/password authentication mechanism of the simple Bind method to establish an authenticated authorization state by sending a Bind request with a name value (a distinguished name in LDAP string form [RFC4514] of non-zero length) and specifying the simple authentication choice containing an OCTET STRING password value of non-zero length.

Servers that map the DN sent in the Bind request to a directory entry with an associated set of one or more passwords used with this mechanism will compare the presented password to that set of passwords. The presented password is considered valid if it matches any member of this set.

A resultCode of invalidDNSyntax indicates that the DN sent in the name value is syntactically invalid. A resultCode of invalidCredentials indicates that the DN is syntactically correct but not valid for purposes of authentication, that the password is not valid for the DN, or that the server otherwise considers the credentials invalid. A resultCode of success indicates that the credentials are valid and that the server is willing to provide service to the entity these credentials identify.

Server behavior is undefined for Bind requests specifying the name/password authentication mechanism with a zero-length name value and a password value of non-zero length.

The name/password authentication mechanism of the simple Bind method is not suitable for authentication in environments without confidentiality protection.

5.2. SASL Authentication Method

The sasl authentication method of the Bind Operation provides facilities for using any SASL mechanism including authentication mechanisms and other services (e.g., data security services).

5.2.1. SASL Protocol Profile

LDAP allows authentication via any SASL mechanism [RFC4422]. As LDAP includes native anonymous and name/password (plain text) authentication methods, the ANONYMOUS [RFC4505] and PLAIN [PLAIN] SASL mechanisms are typically not used with LDAP.

Each protocol that utilizes SASL services is required to supply certain information profiling the way they are exposed through the protocol ([RFC4422], Section 4). This section explains how each of these profiling requirements is met by LDAP.

5.2.1.1. SASL Service Name for LDAP

The SASL service name for LDAP is "ldap", which has been registered with the IANA as a SASL service name.

5.2.1.2. SASL Authentication Initiation and Protocol Exchange

SASL authentication is initiated via a BindRequest message ([RFC4511], Section 4.2) with the following parameters:

- The version is 3.
- The AuthenticationChoice is sasl.
- The mechanism element of the SaslCredentials sequence contains the value of the desired SASL mechanism.
- The optional credentials field of the SaslCredentials sequence MAY be used to provide an initial client response for mechanisms that are defined to have the client send data first (see [RFC4422], Sections 3 and 5).

In general, a SASL authentication protocol exchange consists of a series of server challenges and client responses, the contents of which are specific to and defined by the SASL mechanism. Thus, for some SASL authentication mechanisms, it may be necessary for the client to respond to one or more server challenges by sending BindRequest messages multiple times. A challenge is indicated by the server sending a BindResponse message with the resultCode set to

saslBindInProgress. This indicates that the server requires the client to send a new BindRequest message with the same SASL mechanism to continue the authentication process.

To the LDAP message layer, these challenges and responses are opaque binary tokens of arbitrary length. LDAP servers use the serverSaslCreds field (an OCTET STRING) in a BindResponse message to transmit each challenge. LDAP clients use the credentials field (an OCTET STRING) in the SaslCredentials sequence of a BindRequest message to transmit each response. Note that unlike some Internet protocols where SASL is used, LDAP is not text based and does not Base64-transform these challenge and response values.

Clients sending a BindRequest message with the sasl choice selected SHOULD send a zero-length value in the name field. Servers receiving a BindRequest message with the sasl choice selected SHALL ignore any value in the name field.

A client may abort a SASL Bind negotiation by sending a BindRequest message with a different value in the mechanism field of SaslCredentials or with an AuthenticationChoice other than sasl.

If the client sends a BindRequest with the sasl mechanism field as an empty string, the server MUST return a BindResponse with a resultCode of authMethodNotSupported. This will allow the client to abort a negotiation if it wishes to try again with the same SASL mechanism.

The server indicates completion of the SASL challenge-response exchange by responding with a BindResponse in which the resultCode value is not saslBindInProgress.

The serverSaslCreds field in the BindResponse can be used to include an optional challenge with a success notification for mechanisms that are defined to have the server send additional data along with the indication of successful completion.

5.2.1.3. Optional Fields

As discussed above, LDAP provides an optional field for carrying an initial response in the message initiating the SASL exchange and provides an optional field for carrying additional data in the message indicating the outcome of the authentication exchange. As the mechanism-specific content in these fields may be zero length, SASL requires protocol specifications to detail how an empty field is distinguished from an absent field.

Zero-length initial response data is distinguished from no initial response data in the initiating message, a BindRequest PDU, by the presence of the SaslCredentials.credentials OCTET STRING (of length zero) in that PDU. If the client does not intend to send an initial response with the BindRequest initiating the SASL exchange, it MUST omit the SaslCredentials.credentials OCTET STRING (rather than include an zero-length OCTET STRING).

Zero-length additional data is distinguished from no additional response data in the outcome message, a BindResponse PDU, by the presence of the serverSaslCreds OCTET STRING (of length zero) in that PDU. If a server does not intend to send additional data in the BindResponse message indicating outcome of the exchange, the server SHALL omit the serverSaslCreds OCTET STRING (rather than including a zero-length OCTET STRING).

5.2.1.4. Octet Where Negotiated Security Layers Take Effect

SASL layers take effect following the transmission by the server and reception by the client of the final BindResponse in the SASL exchange with a resultCode of success.

Once a SASL layer providing data integrity or confidentiality services takes effect, the layer remains in effect until a new layer is installed (i.e., at the first octet following the final BindResponse of the Bind operation that caused the new layer to take effect). Thus, an established SASL layer is not affected by a failed or non-SASL Bind.

5.2.1.5. Determination of Supported SASL Mechanisms

Clients may determine the SASL mechanisms a server supports by reading the 'supportedSASLMechanisms' attribute from the root DSE (DSA-Specific Entry) ([RFC4512], Section 5.1). The values of this attribute, if any, list the mechanisms the server supports in the current LDAP session state. LDAP servers SHOULD allow all clients -- even those with an anonymous authorization -- to retrieve the 'supportedSASLMechanisms' attribute of the root DSE both before and after the SASL authentication exchange. The purpose of the latter is to allow the client to detect possible downgrade attacks (see Section 6.4 and [RFC4422], Section 6.1.2).

Because SASL mechanisms provide critical security functions, clients and servers should be configurable to specify what mechanisms are acceptable and allow only those mechanisms to be used. Both clients and servers must confirm that the negotiated security level meets their requirements before proceeding to use the session.

5.2.1.6. Rules for Using SASL Layers

Upon installing a SASL layer, the client SHOULD discard or refresh all information about the server that it obtained prior to the initiation of the SASL negotiation and that it did not obtain through secure mechanisms.

If a lower-level security layer (such as TLS) is installed, any SASL layer SHALL be layered on top of such security layers regardless of the order of their negotiation. In all other respects, the SASL layer and other security layers act independently, e.g., if both a TLS layer and a SASL layer are in effect, then removing the TLS layer does not affect the continuing service of the SASL layer.

5.2.1.7. Support for Multiple Authentications

LDAP supports multiple SASL authentications as defined in [RFC4422], Section 4.

5.2.1.8. SASL Authorization Identities

Some SASL mechanisms allow clients to request a desired authorization identity for the LDAP session ([RFC4422], Section 3.4). The decision to allow or disallow the current authentication identity to have access to the requested authorization identity is a matter of local policy. The authorization identity is a string of UTF-8 [RFC3629] encoded [Unicode] characters corresponding to the following Augmented Backus-Naur Form (ABNF) [RFC4234] grammar:

```
authzId = dnAuthzId / uAuthzId

; distinguished-name-based authz id
dnAuthzId =  "dn:" distinguishedName

; unspecified authorization id, UTF-8 encoded
uAuthzId = "u:" userid
userid = *UTF8 ; syntax unspecified
```

where the distinguishedName rule is defined in Section 3 of [RFC4514] and the UTF8 rule is defined in Section 1.4 of [RFC4512].

The dnAuthzId choice is used to assert authorization identities in the form of a distinguished name to be matched in accordance with the distinguishedNameMatch matching rule ([RFC4517], Section 4.2.15). There is no requirement that the asserted distinguishedName value be that of an entry in the directory.

The uAuthzId choice allows clients to assert an authorization identity that is not in distinguished name form. The format of userid is defined only as a sequence of UTF-8 [RFC3629] encoded [Unicode] characters, and any further interpretation is a local matter. For example, the userid could identify a user of a specific directory service, be a login name, or be an email address. A uAuthzId SHOULD NOT be assumed to be globally unique. To compare uAuthzId values, each uAuthzId value MUST be prepared as a "query" string ([RFC3454], Section 7) using the SASLprep [RFC4013] algorithm, and then the two values are compared octet-wise.

The above grammar is extensible. The authzId production may be extended to support additional forms of identities. Each form is distinguished by its unique prefix (see Section 3.12 of [RFC4520] for registration requirements).

5.2.2. SASL Semantics within LDAP

Implementers must take care to maintain the semantics of SASL specifications when handling data that has different semantics in the LDAP protocol.

For example, the SASL DIGEST-MD5 authentication mechanism [DIGEST-MD5] utilizes an authentication identity and a realm that are syntactically simple strings and semantically simple username [RFC4013] and realm values. These values are not LDAP DNs, and there is no requirement that they be represented or treated as such.

5.2.3. SASL EXTERNAL Authentication Mechanism

A client can use the SASL EXTERNAL ([RFC4422], Appendix A) mechanism to request the LDAP server to authenticate and establish a resulting authorization identity using security credentials exchanged by a lower security layer (such as by TLS authentication). If the client's authentication credentials have not been established at a lower security layer, the SASL EXTERNAL Bind MUST fail with a resultCode of inappropriateAuthentication. Although this situation has the effect of leaving the LDAP session in an anonymous state (Section 4), the state of any installed security layer is unaffected.

A client may either request that its authorization identity be automatically derived from its authentication credentials exchanged at a lower security layer, or it may explicitly provide a desired authorization identity. The former is known as an implicit assertion, and the latter as an explicit assertion.

5.2.3.1. Implicit Assertion

An implicit authorization identity assertion is performed by invoking a Bind request of the SASL form using the EXTERNAL mechanism name that does not include the optional credentials field (found within the SaslCredentials sequence in the BindRequest). The server will derive the client's authorization identity from the authentication identity supplied by a security layer (e.g., a public key certificate used during TLS layer installation) according to local policy. The underlying mechanics of how this is accomplished are implementation specific.

5.2.3.2. Explicit Assertion

An explicit authorization identity assertion is performed by invoking a Bind request of the SASL form using the EXTERNAL mechanism name that includes the credentials field (found within the SaslCredentials sequence in the BindRequest). The value of the credentials field (an OCTET STRING) is the asserted authorization identity and MUST be constructed as documented in Section 5.2.1.8.

6. Security Considerations

Security issues are discussed throughout this document. The unsurprising conclusion is that security is an integral and necessary part of LDAP. This section discusses a number of LDAP-related security considerations.

6.1. General LDAP Security Considerations

LDAP itself provides no security or protection from accessing or updating the directory by means other than through the LDAP protocol, e.g., from inspection of server database files by database administrators.

Sensitive data may be carried in almost any LDAP message, and its disclosure may be subject to privacy laws or other legal regulation in many countries. Implementers should take appropriate measures to protect sensitive data from disclosure to unauthorized entities.

A session on which the client has not established data integrity and privacy services (e.g., via StartTLS, IPsec, or a suitable SASL mechanism) is subject to man-in-the-middle attacks to view and modify information in transit. Client and server implementers SHOULD take measures to protect sensitive data in the LDAP session from these attacks by using data protection services as discussed in this document. Clients and servers should provide the ability to be configured to require these protections. A resultCode of

confidentialityRequired indicates that the server requires establishment of (stronger) data confidentiality protection in order to perform the requested operation.

Access control should always be applied when reading sensitive information or updating directory information.

Various security factors, including authentication and authorization information and data security services may change during the course of the LDAP session, or even during the performance of a particular operation. Implementations should be robust in the handling of changing security factors.

6.2. StartTLS Security Considerations

All security gained via use of the StartTLS operation is gained by the use of TLS itself. The StartTLS operation, on its own, does not provide any additional security.

The level of security provided through the use of TLS depends directly on both the quality of the TLS implementation used and the style of usage of that implementation. Additionally, a man-in-the-middle attacker can remove the StartTLS extended operation from the 'supportedExtension' attribute of the root DSE. Both parties SHOULD independently ascertain and consent to the security level achieved once TLS is established and before beginning use of the TLS-protected session. For example, the security level of the TLS layer might have been negotiated down to plaintext.

Clients MUST either warn the user when the security level achieved does not provide an acceptable level of data confidentiality and/or data integrity protection, or be configurable to refuse to proceed without an acceptable level of security.

As stated in Section 3.1.2, a server may use a local security policy to determine whether to successfully complete TLS negotiation. Information in the user's certificate that is originated or verified by the certification authority should be used by the policy administrator when configuring the identification and authorization policy.

Server implementers SHOULD allow server administrators to elect whether and when data confidentiality and integrity are required, as well as elect whether authentication of the client during the TLS handshake is required.

Implementers should be aware of and understand TLS security considerations as discussed in the TLS specification [RFC4346].

6.3. Bind Operation Security Considerations

This section discusses several security considerations relevant to LDAP authentication via the Bind operation.

6.3.1. Unauthenticated Mechanism Security Considerations

Operational experience shows that clients can (and frequently do) misuse the unauthenticated authentication mechanism of the simple Bind method (see Section 5.1.2). For example, a client program might make a decision to grant access to non-directory information on the basis of successfully completing a Bind operation. LDAP server implementations may return a success response to an unauthenticated Bind request. This may erroneously leave the client with the impression that the server has successfully authenticated the identity represented by the distinguished name when in reality, an anonymous authorization state has been established. Clients that use the results from a simple Bind operation to make authorization decisions should actively detect unauthenticated Bind requests (by verifying that the supplied password is not empty) and react appropriately.

6.3.2. Name/Password Mechanism Security Considerations

The name/password authentication mechanism of the simple Bind method discloses the password to the server, which is an inherent security risk. There are other mechanisms, such as SASL DIGEST-MD5 [DIGEST-MD5], that do not disclose the password to the server.

6.3.3. Password-Related Security Considerations

LDAP allows multi-valued password attributes. In systems where entries are expected to have one and only one password, administrative controls should be provided to enforce this behavior.

The use of clear text passwords and other unprotected authentication credentials is strongly discouraged over open networks when the underlying transport service cannot guarantee confidentiality. LDAP implementations SHOULD NOT by default support authentication methods using clear text passwords and other unprotected authentication credentials unless the data on the session is protected using TLS or other data confidentiality and data integrity protection.

The transmission of passwords in the clear -- typically for authentication or modification -- poses a significant security risk. This risk can be avoided by using SASL authentication [RFC4422]

mechanisms that do not transmit passwords in the clear or by negotiating transport or session layer data confidentiality services before transmitting password values.

To mitigate the security risks associated with the transfer of passwords, a server implementation that supports any password-based authentication mechanism that transmits passwords in the clear MUST support a policy mechanism that at the time of authentication or password modification, requires that:

A TLS layer has been successfully installed.

OR

Some other data confidentiality mechanism that protects the password value from eavesdropping has been provided.

OR

The server returns a resultCode of confidentialityRequired for the operation (i.e., name/password Bind with password value, SASL Bind transmitting a password value in the clear, add or modify including a userPassword value, etc.), even if the password value is correct.

Server implementations may also want to provide policy mechanisms to invalidate or otherwise protect accounts in situations where a server detects that a password for an account has been transmitted in the clear.

6.3.4. Hashed Password Security Considerations

Some authentication mechanisms (e.g., DIGEST-MD5) transmit a hash of the password value that may be vulnerable to offline dictionary attacks. Implementers should take care to protect such hashed password values during transmission using TLS or other confidentiality mechanisms.

6.4. SASL Security Considerations

Until data integrity service is installed on an LDAP session, an attacker can modify the transmitted values of the 'supportedSASLMechanisms' attribute response and thus downgrade the list of available SASL mechanisms to include only the least secure mechanism. To detect this type of attack, the client may retrieve the SASL mechanisms the server makes available both before and after data integrity service is installed on an LDAP session. If the client finds that the integrity-protected list (the list obtained

after data integrity service was installed) contains a stronger mechanism than those in the previously obtained list, the client should assume the previously obtained list was modified by an attacker. In this circumstance it is recommended that the client close the underlying transport connection and then reconnect to reestablish the session.

6.5. Related Security Considerations

Additional security considerations relating to the various authentication methods and mechanisms discussed in this document apply and can be found in [RFC4422], [RFC4013], [RFC3454], and [RFC3629].

7. IANA Considerations

The IANA has updated the LDAP Protocol Mechanism registry to indicate that this document and [RFC4511] provide the definitive technical specification for the StartTLS (1.3.6.1.4.1.1466.20037) extended operation.

The IANA has updated the LDAP LDAPMessage types registry to indicate that this document and [RFC4511] provide the definitive technical specification for the bindRequest (0) and bindResponse (1) message types.

The IANA has updated the LDAP Bind Authentication Method registry to indicate that this document and [RFC4511] provide the definitive technical specification for the simple (0) and sasl (3) bind authentication methods.

The IANA has updated the LDAP authzid prefixes registry to indicate that this document provides the definitive technical specification for the dnAuthzId (dn:) and uAuthzId (u:) authzid prefixes.

8. Acknowledgements

This document combines information originally contained in RFC 2251, RFC 2829, and RFC 2830. RFC 2251 was a product of the Access, Searching, and Indexing of Directories (ASID) Working Group. RFC 2829 and RFC 2830 were products of the LDAP Extensions (LDAPEXT) Working Group.

This document is a product of the IETF LDAP Revision (LDAPBIS) working group.

9. Normative References

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC3454] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("stringprep")", RFC 3454, December 2002.

[RFC3490] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC4013] Zeilenga, K., "SASLprep: Stringprep Profile for User Names and Passwords", RFC 4013, February 2005.

[RFC4234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[RFC4346] Dierks, T. and E. Rescorla, "The TLS Protocol Version 1.1", RFC 4346, March 2006.

[RFC4422] Melnikov, A., Ed. and K. Zeilenga, Ed., "Simple Authentication and Security Layer (SASL)", RFC 4422, June 2006.

[RFC4510] Zeilenga, K., Ed., "Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map", RFC 4510, June 2006.

[RFC4511] Sermersheim, J., Ed., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006.

[RFC4512] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): Directory Information Models", RFC 4512, June 2006.

[RFC4514] Zeilenga, K., Ed., "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names", RFC 4514, June 2006.

[RFC4517] Legg, S., Ed., "Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules", RFC 4517, June 2006.

[RFC4519] Sciberras, A., Ed., "Lightweight Directory Access Protocol (LDAP): Schema for User Applications", RFC 4519, June 2006.

[RFC4520] Zeilenga, K., "Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP)", BCP 64, RFC 4520, June 2006.

[Unicode] The Unicode Consortium, "The Unicode Standard, Version 3.2.0" is defined by "The Unicode Standard, Version 3.0" (Reading, MA, Addison-Wesley, 2000. ISBN 0-201-61633-5), as amended by the "Unicode Standard Annex #27: Unicode 3.1" (http://www.unicode.org/reports/tr27/) and by the "Unicode Standard Annex #28: Unicode 3.2" (http://www.unicode.org/reports/tr28/).

[X.501] ITU-T Rec. X.501, "The Directory: Models", 1993.

10. Informative References

[DIGEST-MD5] Leach, P., Newman, C., and A. Melnikov, "Using Digest Authentication as a SASL Mechanism", Work in Progress, March 2006.

[PLAIN] Zeilenga, K., "The Plain SASL Mechanism", Work in Progress, March 2005.

[RFC2828] Shirey, R., "Internet Security Glossary", FYI 36, RFC 2828, May 2000.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4505] Zeilenga, K., "The Anonymous SASL Mechanism", RFC 4505, June 2006.

Appendix A. Authentication and Authorization Concepts

This appendix is non-normative.

This appendix defines basic terms, concepts, and interrelationships regarding authentication, authorization, credentials, and identity. These concepts are used in describing how various security approaches are utilized in client authentication and authorization.

A.1. Access Control Policy

An access control policy is a set of rules defining the protection of resources, generally in terms of the capabilities of persons or other entities accessing those resources. Security objects and mechanisms, such as those described here, enable the expression of access control policies and their enforcement.

A.2. Access Control Factors

A request, when it is being processed by a server, may be associated with a wide variety of security-related factors. The server uses these factors to determine whether and how to process the request. These are called access control factors (ACFs). They might include source IP address, encryption strength, the type of operation being requested, time of day, etc.. Some factors may be specific to the request itself; others may be associated with the transport connection via which the request is transmitted; and others (e.g., time of day) may be "environmental".

Access control policies are expressed in terms of access control factors; for example, "a request having ACFs i,j,k can perform operation Y on resource Z". The set of ACFs that a server makes available for such expressions is implementation specific.

A.3. Authentication, Credentials, Identity

Authentication credentials are the evidence supplied by one party to another, asserting the identity of the supplying party (e.g., a user) who is attempting to establish a new authorization state with the other party (typically a server). Authentication is the process of generating, transmitting, and verifying these credentials and thus the identity they assert. An authentication identity is the name presented in a credential.

There are many forms of authentication credentials. The form used depends upon the particular authentication mechanism negotiated by the parties. X.509 certificates, Kerberos tickets, and simple identity and password pairs are all examples of authentication

credential forms. Note that an authentication mechanism may constrain the form of authentication identities used with it.

A.4. Authorization Identity

An authorization identity is one kind of access control factor. It is the name of the user or other entity that requests that operations be performed. Access control policies are often expressed in terms of authorization identities; for example, "entity X can perform operation Y on resource Z".

The authorization identity of an LDAP session is often semantically the same as the authentication identity presented by the client, but it may be different. SASL allows clients to specify an authorization identity distinct from the authentication identity asserted by the client's credentials. This permits agents such as proxy servers to authenticate using their own credentials, yet request the access privileges of the identity for which they are proxying [RFC4422]. Also, the form of authentication identity supplied by a service like TLS may not correspond to the authorization identities used to express a server's access control policy, thus requiring a server-specific mapping to be done. The method by which a server composes and validates an authorization identity from the authentication credentials supplied by a client is implementation specific.

Appendix B. Summary of Changes

This appendix is non-normative.

This appendix summarizes substantive changes made to RFC 2251, RFC 2829 and RFC 2830. In addition to the specific changes detailed below, the reader of this document should be aware that numerous general editorial changes have been made to the original content from the source documents. These changes include the following:

- The material originally found in RFC 2251 Sections 4.2.1 and 4.2.2, RFC 2829 (all sections except Sections 2 and 4), and RFC 2830 was combined into a single document.

- The combined material was substantially reorganized and edited to group related subjects, improve the document flow, and clarify intent.

- Changes were made throughout the text to align with definitions of LDAP protocol layers and IETF security terminology.

- Substantial updates and additions were made to security considerations from both documents based on current operational experience.

B.1. Changes Made to RFC 2251

This section summarizes the substantive changes made to Sections 4.2.1 and 4.2.2 of RFC 2251 by this document. Additional substantive changes to Section 4.2.1 of RFC 2251 are also documented in [RFC4511].

B.1.1. Section 4.2.1 ("Sequencing of the Bind Request")

- Paragraph 1: Removed the sentence, "If at any stage the client wishes to abort the bind process it MAY unbind and then drop the underlying connection". The Unbind operation still permits this behavior, but it is not documented explicitly.

- Clarified that the session is moved to an anonymous state upon receipt of the BindRequest PDU and that it is only moved to a non-anonymous state if and when the Bind request is successful.

B.1.2. Section 4.2.2 ("Authentication and Other Security Services")

- RFC 2251 states that anonymous authentication MUST be performed using the simple bind method. This specification defines the anonymous authentication mechanism of the simple bind method and requires all conforming implementations to support it. Other authentication mechanisms producing anonymous authentication and authorization state may also be implemented and used by conforming implementations.

B.2. Changes Made to RFC 2829

This section summarizes the substantive changes made to RFC 2829.

B.2.1. Section 4 ("Required security mechanisms")

- The name/password authentication mechanism (see Section B.2.5 below) protected by TLS replaces the SASL DIGEST-MD5 mechanism as LDAP's mandatory-to-implement password-based authentication mechanism. Implementations are encouraged to continue supporting SASL DIGEST-MD5 [DIGEST-MD5].

B.2.2. Section 5.1 ("Anonymous authentication procedure")

- Clarified that anonymous authentication involves a name value of zero length and a password value of zero length. The unauthenticated authentication mechanism was added to handle simple Bind requests involving a name value with a non-zero length and a password value of zero length.

B.2.3. Section 6 ("Password-based authentication")

- See Section B.2.1.

B.2.4. Section 6.1 ("Digest authentication")

- As the SASL-DIGEST-MD5 mechanism is no longer mandatory to implement, this section is now historical and was not included in this document. RFC 2829, Section 6.1, continues to document the SASL DIGEST-MD5 authentication mechanism.

B.2.5. Section 6.2 ("'simple' authentication choice under TLS encryption")

- Renamed the "simple" authentication mechanism to the name/password authentication mechanism to better describe it.

- The use of TLS was generalized to align with definitions of LDAP protocol layers. TLS establishment is now discussed as an independent subject and is generalized for use with all authentication mechanisms and other security layers.

- Removed the implication that the userPassword attribute is the sole location for storage of password values to be used in authentication. There is no longer any implied requirement for how or where passwords are stored at the server for use in authentication.

B.2.6. Section 6.3 ("Other authentication choices with TLS")

- See Section B.2.5.

B.2.7. Section 7.1 ("Certificate-based authentication with TLS")

- See Section B.2.5.

B.2.8. Section 8 ("Other mechanisms")

- All SASL authentication mechanisms are explicitly allowed within LDAP. Specifically, this means the SASL ANONYMOUS and SASL PLAIN mechanisms are no longer precluded from use within LDAP.

B.2.9. Section 9 ("Authorization Identity")

- Specified matching rules for dnAuthzId and uAuthzId values. In particular, the DN value in the dnAuthzId form must be matched using DN matching rules, and the uAuthzId value MUST be prepared using SASLprep rules before being compared octet-wise.

- Clarified that uAuthzId values should not be assumed to be globally unique.

B.2.10. Section 10 ("TLS Ciphersuites")

- TLS ciphersuite recommendations are no longer included in this specification. Implementations must now support the TLS_RSA_WITH_3DES_EDE_CBC_SHA ciphersuite and should continue to support the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA ciphersuite.

- Clarified that anonymous authentication involves a name value of zero length and a password value of zero length. The unauthenticated authentication mechanism was added to handle simple Bind requests involving a name value with a non-zero length and a password value of zero length.

B.3. Changes Made to RFC 2830

This section summarizes the substantive changes made to Sections 3 and 5 of RFC 2830. Readers should consult [RFC4511] for summaries of changes to other sections.

B.3.1. Section 3.6 ("Server Identity Check")

- Substantially updated the server identity check algorithm to ensure that it is complete and robust. In particular, the use of all relevant values in the subjectAltName and the subjectName fields are covered by the algorithm and matching rules are specified for each type of value. Mapped (derived) forms of the server identity may now be used when the mapping is performed in a secure fashion.

B.3.2. Section 3.7 ("Refresh of Server Capabilities Information")

- Clients are no longer required to always refresh information about server capabilities following TLS establishment. This is to allow for situations where this information was obtained through a secure mechanism.

B.3.3. Section 5 ("Effects of TLS on a Client's Authorization Identity")

- Establishing a TLS layer on an LDAP session may now cause the authorization state of the LDAP session to change.

B.3.4. Section 5.2 ("TLS Connection Closure Effects")

- Closing a TLS layer on an LDAP session changes the authentication and authorization state of the LDAP session based on local policy. Specifically, this means that implementations are not required to change the authentication and authorization states to anonymous upon TLS closure.

- Replaced references to RFC 2401 with RFC 4301.

Author's Address

Roger Harrison
Novell, Inc.
1800 S. Novell Place
Provo, UT 84606
USA

Phone: +1 801 861 2642
EMail: roger_harrison@novell.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).