INTERNET-DRAFT   Editor: R. Harrison
draft-ietf-ldapbis-authmeth-08.txt   Novell, Inc.
Obsoletes: 2251, 2829, 2830   26 October 2003
Intended Category: Draft Standard

LDAP: Authentication Methods and Connection Level Security Mechanisms

Status of this Memo

Copyright Notice

Abstract

This document describes authentication methods and connection level security mechanisms of the Lightweight Directory Access Protocol (LDAP).

This document details the simple Bind authentication method including anonymous, unauthenticated, and plain-text password methods and the SASL (Simple Authentication and Security Layer) Bind authentication method including the use of DIGEST-MD5 and EXTERNAL mechanisms.

This document also details establishment of TLS (Transport Layer Security) using the Start TLS operation.

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

This document also prescribes DIGEST-MD5 as LDAP's mandatory-to-implement strong authentication mechanism.

1. Introduction

The Lightweight Directory Access Protocol (LDAP) [Protocol] is a powerful access protocol for 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:

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

(2) Unauthorized access to reusable client authentication information by monitoring others' access,

(3) Unauthorized access to directory data by monitoring others' access,

(4) Unauthorized modification of directory data,

(5) Unauthorized modification of configuration information,

(6) Unauthorized or excessive use of resources (denial of service), and

(7) Spoofing of directory: Tricking a 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 connection. Also, tricking a client into sending privileged information to a hostile entity that appears to be the directory but is not.

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

LDAP can be protected with the following security mechanisms:

(1) Client authentication by means of the Secure Authentication and Security Layer (SASL) [SASL] mechanism set, possibly backed by the Transport Layer Security (TLS) [TLS] credentials exchange mechanism,

(2) Client authorization by means of access control based on the requestor's authenticated identity,

(3) Data integrity protection by means of TLS or SASL mechanisms with security layers that provide data integrity services,

(4) Data confidentiality protection against snooping by means of the TLS protocol or SASL mechanisms that provide data confidentiality services,

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

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

At the moment, imposition of access controls is done by means outside the scope of LDAP.

It seems clear that allowing any implementation, faced with the above requirements, to simply pick and choose among the possible alternatives is not a strategy that is likely to lead to interoperability. In the absence of mandates, clients will be written that do not support any security function supported by the server, or worse, they will support only mechanisms like the LDAP simple bind using clear text passwords that provide inadequate security for most circumstances.

Given the presence of the Directory, there is a strong desire to see mechanisms where identities take the form of an LDAP distinguished name [LDAPDN] and authentication data can be stored in the directory. This means that this data must be updated outside the protocol or only updated in sessions well protected against snooping. It is also desirable to allow authentication methods to carry authorization identities based on existing--non-LDAP DN--forms of user identities for backwards compatibility with non-LDAP-based authentication 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. Appendix A contains example deployment scenarios that list the mechanisms that might be used to achieve a reasonable level of security in various circumstances.

This document is an integral part of the LDAP Technical Specification [Roadmap]. This document replaces RFC 2829 and portions of RFC 2830 and RFC 2251.

2. Conventions Used in this Document

2.1. Glossary of Terms

The following terms are used in this document. To aid the reader, these terms are defined here.

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

- "connection" and "LDAP connection" both refer to the underlying transport protocol connection between two protocol peers.

- "TLS connection" refers to a TLS-protected LDAP connection.

- "association" and "LDAP association" both refer to the association of the LDAP connection and its current authentication and authorization state.

2.2. Security Terms and Concepts

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 B 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 B before reading the remainder of this document.

2.3. Keywords

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

3. Bind Operation

The Bind operation defined in section 4.2 of [Protocol] allows authentication information to be exchanged between the client and server to establish a new LDAP association. The new LDAP association is established upon successful completion of the authentication exchange.

3.1. Implied Anonymous Bind on LDAP Association

Prior to the successful completion of a Bind operation and during
any subsequent authentication exchange, the session has an anonymous

LDAP association. Among other things this implies that the client need not send a Bind Request in the first PDU of the connection. The client may send any operation request prior to binding, and the server MUST treat it as if it had been performed after an anonymous bind operation. This authentication state on an LDAP association is sometimes referred to as an implied anonymous bind.

3.2. Simple Authentication

The simple authentication choice provides minimal facilities for establishing an anonymous association or for establishing an LDAP association based upon credentials consisting of a name (in the form of an [LDAPDN] and a password.

The simple authentication choice provides two different methods for establishing an anonymous association: anonymous bind and unauthenticated bind (see section 6.1).

The simple authentication choice provides one method for establishing a non-anonymous association: simple password bind.

3.3. SASL Authentication Profile

LDAP allows authentication via any SASL mechanism [SASL]. As LDAP includes native anonymous and plaintext authentication methods, the "ANONYMOUS" [ANONYMOUS] 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 ([SASL] section 5). This section explains how each of these profiling requirements are met by LDAP.

3.3.1. SASL Service Name for LDAP

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

3.3.2. SASL authentication initiation and protocol exchange

SASL authentication is initiated via an LDAP bind request ([Protocol] 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 [SASL] sections 5 and 6.1).

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 invoking the BindRequest multiple times. A challenge is indicated by the server sending a BindResponse with the resultCode set to saslBindInProgress. This indicates that the server requires the client to send a new bind request, with the same sasl mechanism to continue the authentication process.

To the encapsulating protocol, these challenges and responses are opaque binary tokens of arbitrary length. LDAP servers use the serverSaslCreds field, an OCTET STRING, in a bind response message to transmit each challenge. LDAP clients use the credentials field, an OCTET STRING, in the SaslCredentials sequence of a bind request message to transmit each response. Note that unlike some Internet application protocols where SASL is used, LDAP is not text-based, thus no Base64 transformations are performed on these challenge and response values.

Clients sending a bind request with the sasl choice selected SHOULD NOT send a value in the name field. Servers receiving a bind request 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 with a different value in the mechanism field of SaslCredentials, or 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 authMethodNotSupported as the resultCode. This will allow clients 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 bind response in which the resultCode is either success, or an error indication.

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

3.3.3. Octet where negotiated security mechanisms take effect

When negotiated, SASL security layers take effect following the transmission by the server and reception by the client of the final BindResponse in the exchange.

Once a SASL security layer providing 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).

3.3.4. Determination of supported SASL mechanisms

An LDAP client may determine the SASL mechanisms a server supports by performing a search request on the root DSE, requesting the supportedSASLMechanisms attribute. The values of this attribute, if any, list the mechanisms the server supports.

3.3.5. Rules for using SASL security layers

If a SASL security layer is negotiated, the client SHOULD discard information about the server fetched prior to the initiation of the SASL negotiation and not obtained through secure mechanisms.

If the client is configured to support multiple SASL mechanisms, it SHOULD fetch the supportedSASLmechanisms list both before and after the SASL security layer is negotiated. This allows the client to detect active attacks that remove supported SASL mechanisms from the supportedSASLMechanisms list and allows the client to ensure that it is using the best mechanism supported by both client and server. (In particular, this allows for environments where the supportedSASLMechanisms list is provided to the client through a different trusted source, e.g. as part of a digitally signed object.)

If a lower level security layer (such as TLS) is negotiated, any SASL security services SHALL be layered on top of such security layers regardless of the order of their negotiation.

3.3.6. Use of EXTERNAL SASL Mechanism

A client can use the "EXTERNAL" SASL mechanism to request the LDAP server to make use of security credentials exchanged by a lower layer. If authentication credentials have not been established at a lower level (such as by TLS authentication or IP-level security [RFC2401]), the SASL EXTERNAL bind MUST fail with a resultCode of inappropriateAuthentication. Any client authentication and authorization state of the LDAP association is lost, so the LDAP association is in an anonymous state after the failure (see [Protocol] section 4.2.1).

3.4. SASL Authorization Identity

When the "EXTERNAL" SASL mechanism is being negotiated, if the SaslCredentials credentials field is present, it contains an authorization identity. Other mechanisms define the location of the authorization identity in the credentials field. In either case, the authorization identity is represented in the authzId form described below.

3.4.1. Authorization Identity Syntax

The authorization identity is a string of [UTF-8] encoded [Unicode] characters corresponding to the following ABNF grammar [RFC2234]:

```
; Specific predefined authorization (authz) id schemes are
; defined below -- new schemes may be defined in the future.

authzId = dnAuthzId / uAuthzId

DNCOLON  = %x64 %x6e %x3a ; "dn:"
UCOLON = %x75 %x3a ; "u:"

; distinguished-name-based authz id.
dnAuthzId = DNCOLON dn
dn = utf8string    ; with syntax defined in [LDAPDN] section 3.


; unspecified authorization id, UTF-8 encoded.
uAuthzId = UCOLON userid
userid = utf8string    ; syntax unspecified
```

The dnAuthzId choice allows client applications to assert authorization identities in the form of a distinguished name to be matched in accordance with the distinguishedName matching rule [Syntaxes]. The decision to allow or disallow an authentication identity to have access to the requested authorization identity is a matter of local policy ([SASL] section 4.2). For this reason there is no requirement that the asserted dn be that of an entry in directory.

The uAuthzId choice allows for compatibility with client applications that wish to assert an authorization identity to a local directory but do not have that identity in distinguished name form. The value contained within a uAuthzId MUST be prepared using SASLprep before being compared octet-wise. The format of utf8string is defined as only a sequence of of [UTF-8] encoded [Unicode] characters, and further interpretation is subject to prior agreement between the client and server.

For example, the userid could identify a user of a specific directory service or be a login name or the local-part of an RFC 822 email address. A uAuthzId SHOULD NOT be assumed to be globally unique.

Additional authorization identity schemes may be defined in future versions of this document.

4. Start TLS Operation

The Start Transport Layer Security (StartTLS) operation defined in section 4.13 of [Protocol] provides the ability to establish [TLS] on an LDAP association.

4.1. Sequencing of the Start TLS Operation

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

Note that the precise effects, on a client's authorization identity, of establishing TLS on an LDAP association are described in detail in section 4.2.

4.1.1. Requesting to Start TLS on an LDAP Connection

The client MAY send the Start TLS extended request at any time after establishing an LDAP connection, except:

- when TLS is currently established on the connection,
- when a multi-stage SASL negotiation is in progress on the connection, or
- when there are one or more outstanding LDAP operations on the connection.

The result of violating any of these requirements is a resultCode of operationsError, as described in [Protocol] section 4.13.2.2. Client implementers should note that it is possible to receive a resultCode of success for a Start TLS operation that is sent on a connection with outstanding LDAP operations and the server has sufficient time to process them prior to its receiving the Start TLS request. Implementors should ensure that they do not inadvertently depend upon this race condition for proper functioning of their applications.

In particular, there is no requirement that the client have or have not already performed a Bind operation before sending a Start TLS operation request. The client may have already performed a Bind operation when it sends a Start TLS request, or the client might have not yet bound.

If the client did not establish a TLS connection before sending any other requests, and the server requires the client to establish a TLS connection before performing a particular request, the server MUST reject that request by sending a resultCode of confidentialityRequired or strongAuthRequired.

4.1.2. Starting TLS

The server will return an extended response with the resultCode of success if it is willing and able to negotiate TLS. It will return other resultCodes (documented in [Protocol] section 4.13.2.2) if it is unable to do so.

In the successful case, the client (which has ceased to transfer LDAP requests on the connection) MUST either begin a TLS negotiation or close the connection. The client will send PDUs in the TLS Record Protocol directly over the underlying transport connection to the server to initiate [TLS] negotiation.

4.1.3. TLS Version Negotiation

Negotiating the version of TLS or SSL to be used is a part of the [TLS] Handshake Protocol. Please refer to that document for details.

4.1.4. Discovery of Resultant Security Level

After a TLS connection is established on an LDAP association, both parties MUST individually decide whether or not to continue based on the security level achieved. Ascertaining the TLS connection's security level is implementation dependent and accomplished by communicating with one's respective local TLS implementation.

If the client or server decides that the level of authentication or security is not high enough for it to continue, it SHOULD gracefully close the TLS connection immediately after the TLS negotiation has completed (see [Protocol] section 4.13.3.1 and section 4.2.3 below). If the client decides to continue, it may gracefully close the TLS connection and attempt to Start TLS again, it may send an unbind request, or it may send any other LDAP request.

4.1.5. Server Identity Check

The client MUST check its understanding of the server's hostname against the server's identity as presented in the server's Certificate message in order to prevent man-in-the-middle attacks.

Matching is performed according to these rules:

- The client MUST use the server provided by the user (or other trusted entity) as the value to compare against the server name as expressed in the server's certificate. A hostname derived from the user input is to be considered provided by the user only if derived in a secure fashion (e.g., DNSSEC).

- If a subjectAltName extension of type dNSName is present in the certificate, it SHOULD be used as the source of the server's identity.

- Matching is case-insensitive.

- The "*" wildcard character is allowed. If present, it applies only to the left-most name component.

  For example, *.bar.com would match a.bar.com and b.bar.com, but it would not match a.x.bar.com nor would it match bar.com. If more than one identity of a given type is present in the certificate (e.g. more than one dNSName name), a match in any one of the set is considered acceptable.

If the hostname does not match the dNSName-based identity in the
certificate per the above check, user-oriented clients SHOULD either
notify the user (clients may give the user the opportunity to
continue with the connection in any case) or terminate the

connection and indicate that the server's identity is suspect. Automated clients SHOULD close the connection, returning and/or logging an error indicating that the server's identity is suspect.

Beyond the server identity checks 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 observed to provide. The client may need to make use of local policy information in making this determination.

4.1.6. Refresh of Server Capabilities Information

Upon TLS session establishment, the client SHOULD discard or refresh all information about the server fetched prior to the initiation of the TLS negotiation and not obtained through secure mechanisms. This protects against active-intermediary attacks that may have altered any server capabilities information retrieved prior to TLS establishment.

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

4.2. Effects of TLS on a Client's Authorization Identity

This section describes the effects on a client's authorization identity brought about by establishing TLS on an LDAP association. The default effects are described first, and next the facilities for client assertion of authorization identity are discussed including error conditions. Finally, the effects of closing the TLS connection are described.

Authorization identities and related concepts are described in Appendix B.

4.2.1. Default Effects

Upon establishment of the TLS session onto the LDAP association, any previously established authentication and authorization identities MUST remain in force, including anonymous state. This holds even in the case where the server requests client authentication via TLS -- e.g. requests the client to supply its certificate during TLS negotiation.

4.2.2. Client Assertion of Authorization Identity

The client MAY, upon receipt of a Start TLS response indicating success, assert that a specific authorization identity be utilized

in determining the client's authorization status. The client
accomplishes this via an LDAP Bind request specifying a SASL
mechanism of "EXTERNAL" [SASL]. A client may either implicitly
request that its LDAP authorization identity be derived from its

authenticated TLS credentials or it may explicitly provide an authorization identity and assert that it be used in combination with its authenticated TLS credentials. The former is known as an implicit assertion, and the latter as an explicit assertion.

4.2.2.1. Implicit Assertion

An implicit authorization identity assertion is accomplished after TLS establishment by invoking a Bind request of the SASL form using the "EXTERNAL" mechanism name [SASL] [Protocol] that SHALL NOT include the optional credentials octet string (found within the SaslCredentials sequence in the Bind Request). The server will derive the client's authorization identity from the authentication identity supplied in the client's TLS credentials (typically a public key certificate) according to local policy. The underlying mechanics of how this is accomplished are implementation specific.

4.2.2.2. Explicit Assertion

An explicit authorization identity assertion is accomplished after TLS establishment by invoking a Bind request of the SASL form using the "EXTERNAL" mechanism name [SASL] [Protocol] that SHALL include the credentials octet string. This string MUST be constructed as documented in section 3.4.1.

The server MUST verify that the client's authentication identity as supplied in its TLS credentials is permitted to be mapped to 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.

4.2.2.3. Error Conditions

Additionally, with either form of assertion, if a TLS session has not been established between the client and server prior to making the SASL EXTERNAL Bind request and there is no other external source of authentication credentials (e.g. IP-level security [RFC2401]), or if during the process of establishing the TLS session, the server did not request the client's authentication credentials, the SASL EXTERNAL bind MUST fail with a resultCode of inappropriateAuthentication.

After the above Bind operation failures, any client authentication and authorization state of the LDAP association is lost (see [Protocol] section 4.2.1), so the LDAP association is in an anonymous state after the failure. The TLS session state is unaffected, though a server MAY end the TLS session, via a TLS close_notify message, based on the Bind failure (as it MAY at any time).

4.2.3. TLS Connection Closure Effects

Closure of the TLS session MUST cause the LDAP association to move
to an anonymous authentication and authorization state regardless of

the state established over TLS and regardless of the authentication and authorization state prior to TLS session establishment.

5. LDAP Association State Transition Tables

To comprehensively diagram the various authentication and TLS states through which an LDAP association may pass, this section provides a state transition table to represent a state diagram for the various states through which an LDAP association may pass during the course of its existence and the actions that cause these changes in state.

5.1. LDAP Association States

The following table lists the valid LDAP association states and provides a description of each state. The ID for each state is used in the state transition table in section 5.4.

```
ID State Description
-- --------------------------------------------------------------
S1 Anonymous
    no Authentication  ID is associated with the LDAP connection
    no Authorization ID is in force
    No security layer is in effect.
    No TLS credentials have been provided
    TLS: no Creds, OFF]
S2 no Auth ID
    no AuthZ ID
    [TLS: no Creds, ON]
S3 no Auth ID
    no AuthZ ID
    [TLS: Creds Auth ID "I", ON]
S4 Auth ID = Xn
    AuthZ ID= Y
    [TLS: no Creds, OFF]
S5 Auth ID = Xn
    AuthZ ID= Yn
    [TLS: no Creds, ON]
S6 Auth ID = Xn
    AuthZ ID= Yn
    [TLS: Creds Auth ID "I", ON]
S7 Auth ID = I
    AuthZ ID= J
    [TLS: Creds Auth ID "I", ON]
S8 Auth ID = I
    AuthZ ID= K
    [TLS: Creds Auth ID "I", ON]
```

5.2. Actions that Affect LDAP Association State

The following table lists the actions that can affect the state of an LDAP association. The ID for each action is used in the state transition table in section 5.4.

ID	Action
A1	Client binds anonymously
A2	Inappropriate authentication: client attempts an anonymous bind or a bind without supplying credentials to a server that requires the client to provide some form of credentials.
A3	Client Start TLS request Server: client auth NOT required
A4	Client: Start TLS request Server: client creds requested Client: [TLS creds: Auth ID "I"]
A5	Client or Server: send TLS closure alert ([Protocol] section X)
A6	Client: Bind w/simple password or SASL mechanism (e.g. DIGEST-MD5 password, Kerberos, etc., except EXTERNAL [Auth ID "X" maps to AuthZ ID "Y"]
A7	Client Binds SASL EXTERNAL with credentials: AuthZ ID "J" [Explicit Assertion (section 4.2.1.2.2)]
A8	Client Bind SASL EXTERNAL without credentials [Implicit Assertion (section 4.2.1.2.1)]
A9	Client abandons a bind operation or bind operation fails

5.3. Decisions Used in Making LDAP Association State Changes

Certain changes in the state of an LDAP association are only allowed if the server can affirmatively answer a question. These questions are applied as part of the criteria for allowing or disallowing a state change in the state transition table in section 5.4.

ID	Decision Question
D1	Can TLS Credentials Auth ID "I" be mapped to AuthZ ID "J"?
D2	Can a valid AuthZ ID "K" be derived from TLS Credentials Auth ID "I"?

5.4. LDAP Association State Transition Table

The LDAP Association table below lists the valid states for an LDAP association and the actions that could affect them. For any given row in the table, the Current State column gives the state of an LDAP association, the Action column gives an action that could affect the state of an LDAP assocation, and the Next State column gives the resulting state of an LDAP association after the action occurs.

The initial state for the state machine described in this table is S1.

Current State	Action	Next State	Comment
S1	A1	S1	

S1	A2	S1	Error: Inappropriate authentication
S1	A3	S2	
S1	A4	S3	
S1	A6	S4	
S1	A7	?	identity could be provided by another underlying mechanism such as IPSec.
S1	A8	?	identity could be provided by another underlying mechanism such as IPSec.
S2	A1	S2	
S2	A2	S2	Error: Inappropriate authentication
S2	A5	S1	
S2	A6	S5	
S2	A7	?	identity could be provided by another underlying mechanism such as IPSec.
S2	A8	?	identity could be provided by another underlying mechanism such as IPSec.
S3	A1	S3	
S3	A2	S3	Error: Inappropriate authentication
S3	A5	S1	
S3	A6	S6	
S3	A7 and D1=NO	S3	Error: InvalidCredentials
S3	A7 and D1=YES	S7	
S3	A8 and D2=NO	S3	Error: InvalidCredentials
S3	A8 and D2=YES	S8	
S4	A1	S1	
S4	A2	S1	Error: Inappropriate Authentication
S4	A3	S5	
S4	A4	S6	
S4	A5	S1	
S4	A6	S4	
S4	A7	?	identity could be provided by another underlying mechanism such as IPSec.
S4	A8	?	identity could be provided by another underlying mechanism such as IPSec.
S5	A1	S2	
S5	A2	S2	Error: Inappropriate Authentication
S5	A5	S1	
S5	A6	S5	
S5	A7	?	identity could be provided by another underlying mechanism such as IPSec.
S5	A8	?	identity could be provided by another underlying mechanism such

```
                                          as IPSec.
      S6    A1                S3
      S6    A2                S2   Error: Inappropriate Authentication
```

S6	A5	S1	
S6	A6	S6	
S6	A7 and D1=NO	S6	Error: InvalidCredentials
S6	A7 and D1=YES	S7	
S6	A8 and D2=NO	S3	Error: InvalidCredentials
S6	A8 and D2=YES	S8	
S7	A1	S3	
S7	A2	S2	Error: Inappropriate Authentication
S7	A5	S1	
S7	A6	S6	
S7	A7	S7	
S7	A8 and D2=NO	S3	Error: InvalidCredentials
S7	A8 and D2=YES	S8	
S8	A1	S3	
S8	A2	S2	Error: Inappropriate Authentication
S8	A5	S1	
S8	A6	S6	
S8	A7 and D1=NO	S6	Error: InvalidCredentials
S8	A7 and D1=YES	S7	
S8	A8	S8	
Any	A9	S1	See [Protocol] section 4.2.1.

6. Anonymous Authentication

Directory operations that modify entries or access protected attributes or entries generally require client authentication. Clients that do not intend to perform any of these operations typically use anonymous authentication. Servers SHOULD NOT allow clients with anonymous authentication to modify directory entries or access sensitive information in directory entries.

LDAP implementations MUST support anonymous authentication, as defined in section 6.1.

LDAP implementations MAY support anonymous authentication with TLS, as defined in section 6.2.

While there MAY be access control restrictions to prevent access to directory entries, an LDAP server SHOULD allow an anonymously-bound client to retrieve the supportedSASLMechanisms attribute of the root DSE.

An LDAP server MAY use other information about the client provided by the lower layers or external means to grant or deny access even to anonymously authenticated clients.

6.1. Anonymous Authentication Procedure

Prior to successfully completing a Bind operation, the LDAP association is anonymous. See section 3.1.

An LDAP client may also explicitly establish an anonymous association. A client that wishes to do so MUST choose the simple authentication option in the Bind Request and set the password to be of zero length. (This is often done by LDAPv2 clients.) Typically the name is also of zero length. A bind request where both the name and password are of zero length is said to be an anonymous bind. A bind request where the name, a DN, is of non-zero length, and the password is of zero length is said to be an unauthenticated bind. Both variations produce an anonymous association.

6.2. Anonymous Authentication and TLS

An LDAP client MAY use the Start TLS operation (section 5) to negotiate the use of [TLS] security. If the client has not bound beforehand, then until the client uses the EXTERNAL SASL mechanism to negotiate the recognition of the client's certificate, the client is anonymously authenticated.

Recommendations on TLS ciphersuites are given in section 9.

An LDAP server which requests that clients provide their certificate during TLS negotiation MAY use a local security policy to determine whether to successfully complete TLS negotiation if the client did not present a certificate which could be validated.

7. Password-based Authentication

This section discusses various options for performing password-based authentication to LDAP compliant servers and the environments suitable for their use.

7.1. Simple Authentication

The LDAP "simple" authentication choice is not suitable for authentication in environments where there is no network or transport layer confidentiality. LDAP implementations SHOULD support authentication with the "simple" authentication choice when the connection is protected against eavesdropping using TLS, as defined in section 4. LDAP implementations SHOULD NOT support authentication with the "simple" authentication choice unless the data on the connection is protected using TLS or other data confidentiality and data integrity protection.

7.2. Digest Authentication

LDAP servers that implement any authentication method or mechanism (other than simple anonymous bind) MUST implement the SASL DIGEST-MD5 mechanism [DigestAuth].

Support for subsequent authentication is OPTIONAL in clients and servers.

Implementors must take care to ensure that they maintain the semantics of the DIGEST-MD5 specification even when handling data that has different semantics in the LDAP protocol.
For example, the SASL DIGEST-MD5 authentication mechanism utilizes realm and username values ([DigestAuth section 2.1) which are syntactically simple strings and semsantically simple realm and username values. These values are not LDAP DNs, and there is no requirement that they be represented or treated as such. Username and realm values that look like LDAP DNs in form, e.g. "cn=bob, o=Ace Industry ", are syntactically allowed, however DIGEST-MD5 treats them as simple strings for comparison purposes. To illustrate further, the two DNs "cn=bob, o=Ace Industry" (space between RDNs) and "cn=bob,o=Ace Industry" (no space between RDNs) would be equivalent when being compared semantically as LDAP DNs, however they are not equivalent if they were used to represent username values in DIGEST-MD5 because simple octet-wise comparision semantics are used by DIGEST-MD5.

7.3. "simple" authentication choice under TLS encryption

Following the negotiation of an appropriate TLS ciphersuite providing connection confidentiality, a client MAY authenticate to a directory that supports the simple authentication choice by performing a simple bind operation

Simple authentication with TLS encryption protection is performed as follows:

1. The client will use the Start TLS operation [Protocol] to negotiate the use of TLS security [TLS] on the connection to the LDAP server. The client need not have bound to the directory beforehand.

   For the subsequent authentication procedure to be performed securely, the client and server MUST negotiate a ciphersuite which contains a bulk encryption algorithm of appropriate strength. Recommendations on cipher suites are given in section 9.

2. Following the successful completion of TLS negotiation, the client MUST send an LDAP bind request with the version number of 3, the name field containing a DN, and the "simple" authentication choice, containing a password.

7.3.1. "simple" Authentication Choice

DSAs that map the DN sent in the bind request to a directory entry with an associated set of one or more passwords will compare the presented password to the set of passwords associated with that entry. If the presented password matches any member of that set, then the server will respond with a success resultCode, otherwise the server will respond with an invalidCredentials resultCode.

7.4. Other authentication choices with TLS

It is also possible, following the negotiation of TLS, to perform a SASL authentication that does not involve the exchange of plaintext reusable passwords. In this case the client and server need not negotiate a ciphersuite that provides confidentiality if the only service required is data integrity.

8. Certificate-based authentication

LDAP server implementations SHOULD support authentication via a client certificate in TLS, as defined in section 8.1.

8.1. Certificate-based authentication with TLS

A user who has a public/private key pair in which the public key has been signed by a Certification Authority may use this key pair to authenticate to the directory server if the user's certificate is requested by the server. The user's certificate subject field SHOULD be the name of the user's directory entry, and the Certification Authority that issued the user's certificate must be sufficiently trusted by the directory server in order for the server to process the certificate. The means by which servers validate certificate paths is outside the scope of this document.

A server MAY support mappings for certificates in which the subject field name is different from the name of the user's directory entry. A server which supports mappings of names MUST be capable of being configured to support certificates for which no mapping is required.

The client will use the Start TLS operation [Protocol] to negotiate the use of TLS security [TLS] on the connection to the LDAP server. The client need not have bound to the directory beforehand.

In the TLS negotiation, the server MUST request a certificate. The client will provide its certificate to the server, and the server MUST perform a private key-based encryption, proving it has the private key associated with the certificate.

In deployments that require protection of sensitive data in transit, the client and server MUST negotiate a ciphersuite that contains a bulk encryption algorithm of appropriate strength. Recommendations of cipher suites are given in section 9.

The server MUST verify that the client's certificate is valid. The server will normally check that the certificate is issued by a known certification authority (CA), and that none of the certificates on the client's certificate chain are invalid or revoked. There are several procedures by which the server can perform these checks.

Following the successful completion of TLS negotiation, the client will send an LDAP bind request with the SASL "EXTERNAL" mechanism.

9. TLS Ciphersuites

A client or server that supports TLS MUST support TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA and MAY support other ciphersuites offering equivalent or better protection.

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 LDAP 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 confidentially 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 tolerable.

9.1. TLS Ciphersuites Recommendations

As of the writing of this document, the following recommendations regarding TLS ciphersuites are applicable. Because circumstances are constantly changing, this list must not be considered exhaustive, but is hoped that it will serve as a useful starting point for implementers.

The following ciphersuites defined in [TLS] MUST NOT be used for confidentiality protection of passwords or data:

```
      TLS_NULL_WITH_NULL_NULL
      TLS_RSA_WITH_NULL_MD5
      TLS_RSA_WITH_NULL_SHA
```

The following ciphersuites defined in [TLS] can be cracked easily (less than a day of CPU time on a standard CPU in 2000) and are NOT RECOMMENDED for use in confidentiality protection of passwords or data.

```
         TLS_RSA_EXPORT_WITH_RC4_40_MD5
         TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
         TLS_RSA_EXPORT_WITH_DES40_CBC_SHA
         TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA
```

```
      TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA
      TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA
      TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA
      TLS_DH_anon_EXPORT_WITH_RC4_40_MD5
      TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA
```

The following ciphersuites are vulnerable to man-in-the-middle attacks:

```
      TLS_DH_anon_EXPORT_WITH_RC4_40_MD5
      TLS_DH_anon_WITH_RC4_128_MD5
      TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA
      TLS_DH_anon_WITH_DES_CBC_SHA
      TLS_DH_anon_WITH_3DES_EDE_CBC_SHA
```

10. Security Considerations

Security issues are discussed throughout this memo; the (unsurprising) conclusion is that mandatory security is important and that session confidentiality protection is required when snooping is a problem.

Servers are encouraged to prevent modifications by anonymous users.

Servers may also wish to minimize denial of service attacks by timing out idle connections, and returning the unwillingToPerform resultCode rather than performing computationally expensive operations requested by unauthorized clients.

The use of cleartext passwords is strongly discouraged over open networks when the underlying transport service cannot guarantee confidentiality.

Operational experience shows that clients can misuse unauthenticated access (simple bind with name but no password). For example, a client program might authenticate a user via LDAP and then grant access to information not stored in the directory on the basis of completing a successful bind. Some implementations will return a success response to a simple bind that consists of a user name and an empty password thus leaving the impression that the client has successfully authenticated the identity represented by the user name, when in reality, the directory server has simply performed an anonymous bind. For this reason, servers SHOULD by default reject authentication requests that have a DN with an empty password with an error of invalidCredentials.

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

A connection on which the client has not performed the Start TLS
operation or negotiated a suitable SASL mechanism for connection

integrity and encryption services is subject to man-in-the-middle attacks to view and modify information in transit.

10.1. Start TLS Security Considerations

The goals of using the TLS protocol with LDAP are to ensure connection confidentiality and integrity, and to optionally provide for authentication. [TLS] expressly provides these capabilities.

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

Once established, TLS only provides for and ensures confidentiality and integrity of the operations and data in transit over the LDAP association--and only if the implementations on the client and server support and negotiate it. The use of TLS does not provide or ensure for confidentiality and/or non-repudiation of the data housed by an LDAP-based directory server. Nor does it secure the data from inspection by the server administrators.

The level of security provided though the use of TLS depends directly on both the quality of the TLS implementation used and the style of usage of that implementation. Additionally, an active-intermediary attacker can remove the Start TLS extended operation from the supportedExtension attribute of the root DSE. Therefore, both parties SHOULD independently ascertain and consent to the security level achieved once TLS is established and before beginning use of the TLS connection. For example, the security level of the TLS connection might have been negotiated down to plaintext.

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

Client and server implementors SHOULD take measures to ensure proper protection of credentials and other confidential data where such measures are not otherwise provided by the TLS implementation.

Server implementors SHOULD allow for server administrators to elect whether and when connection confidentiality and/or integrity is required, as well as elect whether and when client authentication via TLS is required.

Additional security considerations relating to the EXTERNAL mechanism to negotiate TLS can be found in [SASL] and [TLS].

11. IANA Considerations

The following IANA considerations apply to this document:

Please update the GSSAPI service name registry to point to [Roadmap] and this document.

[To be completed]

Contributors

This document combines information originally contained in RFC 2829 and RFC 2830. The editor acknowledges the work of Harald Tveit Alvestrand, Jeff Hodges, Tim Howes, Steve Kille, RL "Bob" Morgan , and Mark Wahl, each of whom authored one or more of these documents.

Acknowledgements

This document is based upon input of the IETF LDAP Revision working group. The contributions and suggestions made by its members in shaping the contents and technical accuracy of this document is greatly appreciated.

Normative References

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

[RFC2234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[DigestAuth] Leach, P. C. Newman, and A. Melnikov, "Using Digest Authentication as a SASL Mechanism", draft-ietf-sasl-rfc2831bis-xx.txt, a work in progress.

[LDAPDN] Zeilenga, Kurt D. (editor), "LDAP: String Representation of Distinguished Names", draft-ietf-ldapbis-dn-xx.txt, a work in progress.

[Model] Zeilenga, Kurt D. (editor), "LDAP: Directory Information Models", draft-ietf-ldapbis-models-xx.txt, a work in progress.

[Protocol] Sermersheim, J., "LDAP: The Protocol", draft-ietf-ldapbis-protocol-xx.txt, a work in progress.

[Roadmap] K. Zeilenga, "LDAP: Technical Specification Road Map", draft-ietf-ldapbis-roadmap-xx.txt, a work in progress.

[SASL] Melnikov, A. (editor), "Simple Authentication and Security Layer (SASL)", draft-ietf-sasl-rfc2222bis-xx.txt, a work in progress.

[Syntaxes] Legg, S. (editor), "LDAP: Syntaxes and Matching Rules", draft-ietf-ldapbis-syntaxes-xx.txt, a work in progress.

[TLS] Dierks, T. and C. Allen. "The TLS Protocol Version 1.1", draft-ietf-tls-rfc2246-bis-xx.txt, a work in progress.

   [UTF-8] Yergeau, F., "UTF-8, a transformation format of ISO 10646",
        RFC 2279, January 1998.

[Unicode] International Organization for Standardization, "Universal Multiple-Octet Coded Character Set (UCS) - Architecture and Basic Multilingual Plane", ISO/IEC 10646-1 : 1993.

Informative References

[ANONYMOUS] Zeilenga, K.,"Anonymous SASL Mechanism", draft-zeilenga-sasl-anon-xx.txt, a work in progress.

[PLAIN] Zeilenga, K.,"Plain SASL Mechanism", draft-zeilenga-sasl-plain-xx.txt, a work in progress.

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

[RFC2401] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

Author's Address

Roger Harrison
Novell, Inc.
1800 S. Novell Place
Provo, UT 84606
+1 801 861 2642
roger_harrison@novell.com

Full Copyright Statement

HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Appendix A. Example Deployment Scenarios

The following scenarios are typical for LDAP directories on the Internet, and have different security requirements. (In the following discussion, "sensitive data" refers to information whose disclosure, alteration, destruction, or loss would adversely affect the interests or business of its owner or user. Also note that there may be data that is protected but not sensitive.) This is not intended to be a comprehensive list; other scenarios are possible, especially on physically protected networks.

(1) A read-only directory, containing no sensitive data, accessible to "anyone", and TCP connection hijacking or IP spoofing is not a problem. Anonymous authentication, described in section 7, is suitable for this type of deployment, and requires no additional security functions except administrative service limits.

(2) A read-only directory containing no sensitive data; read access is granted based on identity. TCP connection hijacking is not currently a problem. This scenario requires data confidentiality for sensitive authentication information AND data integrity for all authentication information.

(3) A read-only directory containing no sensitive data; and the client needs to ensure the identity of the directory server and that the directory data is not modified while being returned from the server. A data origin authentication service AND data integrity service are required.

(4) A read-write directory, containing no sensitive data; read access is available to "anyone", update access to properly authorized persons. TCP connection hijacking is not currently a problem. This scenario requires data confidentiality for sensitive authentication information AND data integrity for all authentication information.

(5) A directory containing sensitive data. This scenario requires data confidentiality protection AND secure authentication.

Appendix B. Authentication and Authorization: Definitions and Concepts

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.

B.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. A common expression of an access control policy is an access control list. Security objects and mechanisms, such as those described here, enable the expression of access control policies and their enforcement. Access control policies are typically expressed in terms of access control factors as described below.

B.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 (section 4.2 of [Protocol]). 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 connection via which the request is transmitted, others (e.g. time of day) may be "environmental".

Access control policies are expressed in terms of access control factors. E.g., 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.

B.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 an association 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. For example: X.509 certificates, Kerberos tickets, simple identity and password pairs. Note that an authentication mechanism may constrain the form of authentication identities used with it.

B.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; e.g., entity X can perform operation Y on resource Z.

The authorization identity bound to an association is often exactly 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 [SASL]. 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, 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 C. RFC 2829 Change History

This appendix lists the changes made to the text of RFC 2829 in preparing this document.

C.0. General Editorial Changes

Version -00

- Changed other instances of the term LDAP to LDAP where v3 of the protocol is implied. Also made all references to LDAP use the same wording.

- Miscellaneous grammatical changes to improve readability.

- Made capitalization in section headings consistent.

Version -01

- Changed title to reflect inclusion of material from RFC 2830 and 2251.

C.1. Changes to Section 1

Version -01

- Moved conventions used in document to a separate section.

C.2. Changes to Section 2

Version -01

- Moved section to an appendix.

C.3. Changes to Section 3

Version -01

- Moved section to an appendix.

C.4 Changes to Section 4

Version -00

- Changed "Distinguished Name" to "LDAP distinguished name".

C.5. Changes to Section 5

Version -00

- Added the following sentence: "Servers SHOULD NOT allow clients with anonymous authentication to modify directory entries or access sensitive information in directory entries."

C.5.1. Changes to Section 5.1

Version -00

- Replaced the text describing the procedure for performing an anonymous bind (protocol) with a reference to section 4.2 of RFC 2251 (the protocol spec).

Version -01

- Brought text describing procedure for performing an anonymous bind from section 4.2 of RFC 2251 bis. This text will be removed from the draft standard version of that document.

C.6. Changes to Section 6.

Version -00

Reorganized text in section 6.1 as follows:

1. Added a new section (6.1) titled "Simple Authentication" and moved one of two introductory paragraphs for section 6 into section 6.1. Added sentences to the paragraph indicating:

   a. simple authentication is not suitable for environments where confidentiality is not available.

   b. LDAP implementations SHOULD NOT support simple authentication unless confidentiality and data integrity mechanisms are in force.

2. Moved first paragraph of section 6 (beginning with "LDAP implementations MUST support authentication with a password...") to section on Digest Authentication (Now section 6.2).

C.6.1. Changes to Section 6.1.

Version -00 Renamed section to 6.2

- Added sentence from original section 6 indicating that the DIGEST-MD5 SASL mechanism is required for all conforming LDAP implementations

C.6.2. Changes to Section 6.2

Version -00

- Renamed section to 6.3

- Reworded first paragraph to remove reference to user and the userPassword password attribute Made the first paragraph more general by simply saying that if a directory supports simple authentication that the simple bind operation MAY performed following negotiation of a TLS ciphersuite that supports confidentiality.

- Replaced "the name of the user's entry" with "a DN" since not all bind operations are performed on behalf of a "user."

- Added Section 6.3.1 heading just prior to paragraph 5.

- Paragraph 5: replaced "The server" with "DSAs that map the DN sent in the bind request to a directory entry with a userPassword attribute."

C.6.3. Changes to section 6.3.

Version -00

- Renamed to section 6.4.

C.7. Changes to section 7.

none

C.7.1. Changes to section 7.1.

Version -00

- Clarified the entity issuing a certificate by moving the phrase "to have issued the certificate" immediately after "Certification Authority."

C.8. Changes to section 8.

Version -00

- Removed the first paragraph because simple authentication is covered explicitly in section 6.

- Added section 8.1. heading just prior to second paragraph.

- Added section 8.2. heading just prior to third paragraph.

- Added section 8.3. heading just prior to fourth paragraph.

Version -01

- Moved entire section 8 of RFC 2829 into section 3.4 (Using SASL for Other Security Services) to bring material on SASL mechanisms together into one location.

C.9. Changes to section 9.

Version -00

- Paragraph 2: changed "EXTERNAL mechanism" to "EXTERNAL SASL mechanism."

- Added section 9.1. heading.

- Modified a comment in the ABNF from "unspecified userid" to "unspecified authz id".

- Deleted sentence, "A utf8string is defined to be the UTF-8 encoding of one or more ISO 10646 characters," because it is redundant.

- Added section 9.1.1. heading.

- Added section 9.1.2. heading.

Version -01

- Moved entire section 9 to become section 3.5 so that it would be with other SASL material.

C.10. Changes to Section 10.

Version -00

- Updated reference to cracking from a week of CPU time in 1997 to be a day of CPU time in 2000.

- Added text: "These ciphersuites are NOT RECOMMENDED for use... and server implementers SHOULD" to sentence just prior the second list of ciphersuites.

- Added text: "and MAY support other ciphersuites offering equivalent or better protection," to the last paragraph of the section.

C.11. Changes to Section 11.

Version -01

- Moved to section 3.6 to be with other SASL material.

C.12. Changes to Section 12.

Version -00

- Inserted new section 12 that specifies when SASL protections begin following SASL negotiation, etc. The original section 12 is renumbered to become section 13.

Version -01

- Moved to section 3.7 to be with other SASL material.

C.13. Changes to Section 13 (original section 12).

None

Appendix D. RFC 2830 Change History

This appendix lists the changes made to the text of RFC 2830 in preparing this document.

D.0. General Editorial Changes

- Material showing the PDUs for the Start TLS response was broken out into a new section.

- The wording of the definition of the Start TLS request and Start TLS response was changed to make them parallel. NO changes were made to the ASN.1 definition or the associated values of the parameters.

- A separate section heading for graceful TLS closure was added for parallelism with section on abrupt TLS closure.

Appendix E. RFC 2251 Change History

This appendix lists the changes made to the text of RFC 2251 in preparing this document.

E.0. General Editorial Changes

- All material from section 4.2 of RFC 2251 was moved into this document.

- A new section was created for the Bind Request

- Section 4.2.1 of RFC 2251 (Sequencing Bind Request) was moved after the section on the Bind Response for parallelism with the presentation of the Start TLS operations. The section was also subdivided to explicitly call out the various effects being described within it.

- All SASL profile information from RFC 2829 was brought within the discussion of the Bind operation (primarily sections 4.4 - 4.7).

Appendix F. Change History to Combined Document

F.1. Changes for draft-ldap-bis-authmeth-02

General

- Added references to other LDAP standard documents, to sections within the document, and fixed broken references.

- General editorial changes--punctuation, spelling, formatting, etc.

Section 1.

- Added glossary of terms and added sub-section headings

Section 2.

- Clarified security mechanisms 3, 4, & 5 and brought language in line with IETF security glossary.

Section 3.

- Brought language in requirement (3) in line with security glossary.

- Clarified that information fetched prior to initiation of TLS negotiation must be discarded

-Clarified that information fetched prior to initiation of SASL negotiation must be discarded

- Rewrote paragraph on SASL negotiation requirements to clarify intent

Section 4.4.

- Added stipulation that sasl choice allows for any SASL mechanism not prohibited by this document. (Resolved conflict between this statement and one that prohibited use of ANONYMOUS and PLAIN SASL mechanisms.)

Section 5.3.6

- Added a.x.bar.com to wildcard matching example on hostname check.

Section 6

- Added LDAP Association State Transition Tables to show the various states through which an LDAP association may pass along with the actions and decisions required to traverse from state to state.

Appendix A

- Brought security terminology in line with IETF security glossary
  throughout the appendix.

F.2. Changes for draft-ldap-bis-authmeth-03

General

- Added introductory notes and changed title of document and references to conform to WG chair suggestions for the overall technical specification.

- Several issues--G.13, G.14, G.16, G.17--were resolved without requiring changes to the document.

Section 3

- Removed reference to /etc/passwd file and associated text.

Section 4

- Removed sections 4.1, 4.2 and parts of section 4.3. This information was being duplicated in the protocol specification and will now reside there permanently.

Section 4.2

- changed words, "not recommended" to "strongly discouraged"

Section 4.3

- Based on ldapbis WG discussion at IETF52 two sentences were added indicating that clients SHOULD NOT send a DN value when binding with the sasl choice and servers SHALL ignore any value received in this circumstance.
-

Section 8.3.1

- Generalized the language of this section to not refer to any specific password attribute or to refer to the directory entry as a "user" entry.

Section 11

- Added security consideration regarding misuse of unauthenticated access.

- Added security consideration requiring access control to be applied only to authenticated users and recommending it be applied when reading sensitive information or updating directory information.

F.3. Changes for draft-ldap-bis-authmeth-04

   General

- Changed references to use [RFCnnnn] format wherever possible. (References to works in progress still use [name] format.)
- Various edits to correct typos and bring field names, etc. in line with specification in [Protocol] draft.

- Several issues--G.13, G.14, G.16, G.17--were resolved without requiring changes to the document.

Section 4.4.1.

- Changed ABNF grammar to use productions that are like those in the model draft.

Section 5

- Removed sections 5.1, 5.2, and 5.4 that will be added to [Protocol]. Renumbered sections to accommodate this change.
-

Section 6

- Reviewed LDAP Association State table for completeness and accuracy. Renumbered actions A3, A4, and A5 to be A5, A3, and A4 respectively. Re-ordered several lines in the table to ensure that actions are in ascending order (makes analyzing the table much more logical). Added action A2 to several states where it was missing and valid. Added actions A7 and A8 placeholders to states S1, S2, S4 and S5 pending resolution of issue G.28.

Section 11

- Modified security consideration (originally added in -03) requiring access control to be applied only to authenticated users. This seems nonsensical because anonymous users may have access control applied to limit permissible actions.
-

Section 13

- Verified all normative references and moved informative references to a new section 14.

F.4. Changes for draft-ldap-bis-authmeth-05

General

- General editory changes to fix punctuation, spelling, line length issues, etc.
- Verified and updated intra- and inter-document references throughout.
- Document-wide review for proper usage of RFC 2119 keywords with

several changes to correct improper usage.

Abstract

- Updated to match current contents of documents. This was needed due to movement of material on Bind and Start TLS operations to [Protocol] in this revision.

Section 3.

- Renamed section to "Rationale for LDAP Security Mechanisms" and removed text that did not support this theme. Part of the motivation for this change was to remove the implication of the previous section title, "Required Security Mechanisms", and other text found in the section that everything in the section was a requirement

- Information from several removed paragraphs that describe deployment scenarios will be added Appendix A in the next revision of the draft.

- Paragraph beginning, " If TLS is negotiated, the client MUST discard all information..." was moved to section 5.1.7 and integrated with related material there.

- Paragraph beginning, "If a SASL security layer is negotiated..." was moved to section 4.2

Section 4.l.

- Changed wording of first paragraph to clarify meaning.

Section 4.2.
- Added paragraph from section 3 of -04 beginning, "If a SASL security layer is negotiated..."

Section 4.3.3.
- Renamed to "Other SASL Mechanisms" and completely rewrote the section (one sentence) to generalize the treatment of SASL mechanisms not explicitly mentioned in this document.

Section 4.4.1.

- Added paragraph beginning, "The dnAuthzID choice allows client applications..." to clarify whether DN form authorization identities have to also have a corresponding directory entry. This change was based on editor's perception of WG consensus.

- Made minor clarifying edits in the paragraph beginning, "The uAuthzID choice allows for compatibility..."

Section 5.1.1.

- Made minor clarifying edits in the last paragraph of the
  section.

Section 5.1.7.

- Wording from section 3 paragraph beginning " If TLS is negotiated, the client MUST discard all information..." was moved to this section and integrated with existing text.

Section 5.2.

- Changed usage of "TLS connection" to "TLS session" throughout.

- Removed empty section 5.2.1 and renumbered sections it had previously contained.

Section 8.

- Added introductory paragraph at beginning of section.

Section 8.1.

- Changed term "data privacy" to "data confidentiality" to be consistent with usage in rest of document.

Section 8.2.

- Changed first paragraph to require implementations that implement *password-based* authentication to implement and support DIGEST-MD5 SASL authentication.

Section 11.

- First paragraph: changed "session encryption" to "session confidentiality protection" to be consistent with usage in rest of document.

Appendix A.

- Began changes to incorporate information on deployment scenarios removed from section 3.

F.5. Changes for draft-ldap-bis-authmeth-06

General

- Combined Section 2 (Introduction) and Section 3 (Motivation) and moved Introduction to section 1. All following sections numbers were decremented by one as result.

- Edits to fix typos, I-D nits, etc.

- Opened several new issues in Appendix G based on feedback from WG. Some of these have been resolved. Others require further discussion.

Section 1

- Added additional example of spoofing under threat (7).

Section 2.1

- Changed definition of "LDAP association" and added terms, "connection" and "TLS connection" to bring usage in line with [Protocol].

Section 4.1.6

- Clarified sentence stating that the client MUST NOT use derived forms of DNS names.

Section 5.1

- Began edits to LDAP Association state table to clarify meaning of various states and actions.

- Added action A9 to cover abandoned bind operation and added appropriate transitions to the state transition table to accommodate it.

Section 7.2

- Replaced first paragraph to clarify that the "DIGEST-MD5" SASL mechanism is required to implement.

Section 9

- Rewrote the section to make the advice more applicable over the long term, i.e. more "timeless." The intent of content in the original section was preserved.

Section 10

- Added a clarifying example to the consideration regarding misuse of unauthenticated access.

F.6. Changes for draft-ldap-bis-authmeth-07

General

- Updated external and internal references to accommodate changes in recent drafts.

- Opened several new issues in Appendix G based on feedback from WG. Some of these have been resolved. Others require further discussion.

Section 3

- Rewrote much of section 3.3 to mee the SASL profile requirements of draft-ietf-sasl-rfc2222bis-xx.txt section 5.

- Changed treatement of SASL ANONYMOUS and PLAIN mechanisms to bring in line with WG consensus.

Section 4

- Note to implementers in section 4.1.1 based on operational experience.

- Clarification on client continuing by performing a Start TLS with TLS already established in section 4.1.4.

- Moved verification of mapping of client's authentication ID to asserted authorization ID to apply only to explicit assertion. The local policy in place for implicit assertion is adequate.

Section 7

- Removed most of section 7.2 as the information is now covered adequately via the new SASL profile in section 3.3. Added note to implementors regarding the treatment of username and realm values in DIGEST-MD5.

- Section 7.3. Minor clarifications in wording.

- Section 7.3.1. Clarification that a match of the presented value to any member of the set of stored passwords constitutes a successful authentication.

F.6. Changes for draft-ldap-bis-authmeth-08

General

- Changed usage from LDAPv3 to LDAP for usage consistency across LDAP technical specification.
- Fixed a number of usage nits for consistency and to bring doc in conformance with publication guidelines.

Abstract

- Significant cleanup and rewording of abstract based on WG feedback.

Section 2.1

- New definition of user.

Section 3

- Added 1.5 sentences at end of introductory paragraph indicating the effect of the Bind op on the LDAP association.

Section 3.1

- Retitled section and clarified wording

Section 3.2

- Clarified that simple authentication choice provides three types of authentication: anonymous, unauthenticated, and simple password.

Section 3.3.3

- New wording clarifying when negotiated security mechanisms take effect.

Section 3.3.5

- Changed requirement to discard information about server fetched prior to SASL negotion from MUST to SHOULD to allow for information obtained through secure mechanisms.

Section 3.3.6

- Simplified wording of first paragraph based on suggestion from WG.

Section 3.4

- Minor clarifications in wording.

Section 3.4.1

- Minor larifications in wording in first sentence.
- Explicitly called out that the DN value in the dnAuthzID form is to be matched using DN matching rules.
- Called out that the uAuthzID MUST be prepared using SASLprep rules before being compared.
- Clarified requirement on assuming global uniqueness by changing a "generally... MUST" wording to "SHOULD".

Section 4.1.1

- Simplified wording describing conditions when Start TLS cannot be sent.
- Simplified wording in note to implementers regarding race condition with outstanding LDAP operations on connection.

Section 4.1.5

- Removed section and moved relevant text to section 4.2.2.

Section 4.1.6

- Renumbered to 4.1.5.
- Updated server identity check rules for server's name based on WG list discussion.

Section 4.1.7

- Renumbered to 4.1.6
- Changed requirement to discard information about server fetched prior to TLS negotion from MUST to SHOULD to allow for information obtained through secure mechanisms.

Section 6.1

- Clarified wording.
- Added definition of anonymous and unauthenticated binds.

Section 10

- Added security consideration (moved from elsewhere) discouraging use of cleartext passwords on unprotected communication channels.

Section 11

- Added an IANA consideration to update GSSAPI service name registry to point to [Roadmap] and [Authmeth]

Appendix G. Issues to be Resolved

This appendix lists open questions and issues that need to be resolved before work on this document is deemed complete.

G.1.

Section 1 lists 6 security mechanisms that can be used by LDAP servers. I'm not sure what mechanism 5, "Resource limitation by means of administrative limits on service controls" means.

Status: resolved. Changed wording to "administrative service limits" to clarify meaning.

G.2.

Section 2 paragraph 1 defines the term, "sensitive." Do we want to bring this term and other security-related terms in alignment with usage with the IETF security glossary (RFC 2828)?

Status: resolved. WG input at IETF 51 was that we should do this, so the appropriate changes have been made.

G.3.

Section 2, deployment scenario 2: What is meant by the term "secure authentication function?"

Status: resolved. Based on the idea that a "secure authentication function" could be provided by TLS, I changed the wording to require data confidentiality for sensitive authentication information and data integrity for all authentication information.

G.4.

Section 3, deployment scenario 3: What is meant by the phrase, "directory data is authenticated by the server?"

Status: resolved. I interpreted this to mean the ability to ensure the identity of the directory server and the integrity of the data sent from that server to the client, and explictly stated such.

G.5.

Section 4 paragraph 3: What is meant by the phrase, "this means that either this data is useless for faking authentication (like the Unix "/etc/passwd" file format used to be)?"

Status: resolved. Discussion at IETF 52 along with discussions with the original authors of this material have convinced us that this reference is simply too arcane to be left in place. In -03 the text has been modified to focus on the need to either update password information in a protected fashion outside of the protocol or to update it in session well protected against snooping, and the reference to /etc/passwd has been removed.

G.6.

Section 4 paragraph 7 begins: "For a directory needing session protection..." Is this referring to data confidentiality or data integrity or both?

Status: resolved. Changed wording to say, "For a directory needing data security (both data integrity and data confidentiality)..."

G.7.

Section 4 paragraph 8 indicates that "information about the server fetched fetched prior to the TLS negotiation" must be discarded. Do we want to explicitly state that this applies to information fetched prior to the *completion* of the TLS negotiation or is this going too far?

Status: resolved. Based on comments in the IETF 51 LDAPBIS WG meeting, this has been changed to explicitly state, "fetched prior to the initiation of the TLS negotiation..."

G.8.

Section 4 paragraph 9 indicates that clients SHOULD check the supportedSASLMechanisms list both before and after a SASL security layer is negotiated to ensure that they are using the best available security mechanism supported mutually by the client and server. A note at the end of the paragraph indicates that this is a SHOULD since there are environments where the client might get a list of supported SASL mechanisms from a different trusted source.

I wonder if the intent of this could be restated more plainly using one of these two approaches (I've paraphrased for the sake of brevity):

Approach 1: Clients SHOULD check the supportedSASLMechanisms list both before and after SASL negotiation or clients SHOULD use a different trusted source to determine available supported SASL mechanisms.

Approach 2: Clients MUST check the supportedSASLMechanisms list both before and after SASL negotiation UNLESS they use a different trusted source to determine available supported SASL mechanisms.

Status: resolved. WG input at IETF 51 was that Approach 1 was probably best. I ended up keeping the basic structure similar to the original to meet this intent.

G.9.

Section 6.3.1 states: "DSAs that map the DN sent in the bind request to a directory entry with a userPassword attribute will... compare [each value in the named user's entry]... with the presented password." This implies that this applies only to user entries with userPassword attributes. What about other types of entries that might allow passwords and might store in the password information in other attributes? Do we want to make this text more general?

Status: resolved in -03 draft by generalizing section 8.3.1 to not refer to any specific password attribute and by removing the term "user" in referring to the directory entry specified by the DN in the bind request.

G.10 userPassword and simple bind

We need to be sure that we don't require userPassword to be the only attribute used for authenticating via simple bind. (See 2251 sec 4.2 and authmeth 6.3.1. Work with Jim Sermersheim on resolution to this. On publication state something like: "This is the specific implementation of what we discussed in our general reorg conversation on the list." (Source: Kurt Zeilenga)

Status: resolved in -03 draft by generalizing section 8.3.1 to not refer to any specific password attribute and by removing the term "user" in referring to the directory entry specified by the DN in the bind request.

G.11. Meaning of LDAP Association

The original RFC 2830 uses the term "LDAP association" in describing a connection between an LDAP client and server regardless of the state of TLS on that connection. This term needs to be defined or possibly changed.

Status: resolved. at IETF 51 Bob Morgan indicated that the term "LDAP association" was intended to distinguish the LDAP-level connection from the TLS-level connection. This still needs to be clarified somewhere in the draft. Added "LDAP association" to a glossary in section 1.

G.12. Is DIGEST-MD5 mandatory for all implementations?

Reading 2829bis I think DIGEST-MD5 is mandatory ONLY IF your server supports password based authentication...but the following makes it sound mandatory to provide BOTH password authentication AND DIGEST-MD5:

"6.2. Digest authentication

LDAP implementations MUST support authentication with a password using the DIGEST-MD5 SASL mechanism for password protection, as defined in section 6.1."

The thing is for acl it would be nice (though not critical) to be able to default the required authentication level for a subject to a single "fairly secure" mechanism--if there is no such mandatory authentication scheme then you cannot do that. (Source: Rob Byrne)

Status: resolved. -00 version of the draft added a sentence at the beginning of section 8.2 stating that LDAP server implementations must support this method.

G.13. Ordering of authentication levels requested

Again on the subject of authentication level, is it possible to define an ordering on authentication levels which defines their relative "strengths" ? This would be useful in acl as you could say things like"a given aci grants access to a given subject at this authentication level AND ABOVE". David Chadwick raised this before in the context of denying access to a subject at a given authentication level, in which case he wanted to express "deny access to this subject at this authentication level AND TO ALL IDENTITIES AUTHENTICATED BELOW THAT LEVEL". (Source: Rob Byrne)

Status: out of scope. This is outside the scope of this document and will not be addressed.

G.14. Document vulnerabilities of various mechanisms

While I'm here...in 2829, I think it would be good to have some
comments or explicit reference to a place where the security
properties of the particular mandatory authentication schemes are

outlined. When I say "security properties" I mean stuff like "This scheme is vulnerable to such and such attacks, is only safe if the key size is > 50, this hash is widely considered the best, etc...". I think an LDAP implementor is likely to be interested in that information, without having to wade through the security RFCs. (Source: Rob Byrne)

Status: out of scope. This is outside the scope of this document and will not be addressed.

G.15. Include a StartTLS state transition table

The pictoral representation it is nominally based on is here (URL possibly folded):

http://www.stanford.edu/~hodges/doc/LDAPAssociationStateDiagram-1999-12-14.html

(Source: Jeff Hodges)

Status: In Process. Table provided in -03. Review of content for accuracy in -04. Additional review is needed, plus comments from WG members indicate that additional description of each state's meaning would be helpful.

G.16. Empty sasl credentials question

I spent some more time looking microscopically at ldap-auth-methods and ldap-ext-tls drafts. The drafts say that the credential must have the form dn:xxx or u:xxx or be absent, and although they don't say what to do in the case of an empty octet string I would say that we could send protocolError (claim it is a bad PDU).

There is still the question of what to do if the credential is 'dn:' (or 'u:') followed by the empty string. (Source: ariel@columbia.edu via Jeff Hodges)

Status: resolved. Kurt Zeilenga indicated during ldapbis WG discussion at IETF 52 that SASL AuthzID credentials empty and absent are equivalent in the latest SASL ID. This resolves the issue.

G.17. Hostname check from MUST to SHOULD?

I am uneasy about the hostname check. My experience from PKI with HTTP probably is a contributing factor; we have people using the short hostname to get to a server which naturally has the FQDN in the certificate, no end of problems. I have a certificate on my laptop which has the FQDN for the casse when the system is on our Columbia network with a fixed IP; when I dial in however, I have some horrible dialup name, and using the local https server becomes annoying. Issuing a certificate in the name 'localhost' is not a solution! Wildcard match does not solve this problem. For these reasons I am inclined to argue for 'SHOULD' instead of 'MUST' in paragraph...

Also, The hostname check against the name in the certificate is a very weak means of preventing man-in-the-middle attacks; the proper solution is not here yet (SecureDNS or some equivalent). Faking out DNS is not so hard, and we see this sort of thing in the press on a pretty regular basis, where site A hijacks the DNS server for site B and gets all their requests. Some mention of this should be made in the draft. (Source: ariel@columbia.edu via Jeff Hodges)

Status: resolved. Based on discussion at IETF 52 ldapbis WG meeting, this text will stand as it is. The check is a MUST, but the behavior afterward is a SHOULD. This gives server implementations the room to maneuver as needed.

G.18. Must SASL DN exist in the directory?

If the 'dn:' form of sasl creds is used, is it the intention of the draft(ers) that this DN must exist in the directory and the client will have the privileges associated with that entry, or can the server map the sasl DN to perhaps some other DN in the directory, in an implementation-dependent fashion?

We already know that if *no* sasl credentials are presented, the DN or altname in the client certificate may be mapped to a DN in an implementation-dependent fashion, or indeed to something not in the directory at all. (Right?) (Source: ariel@columbia.edu via Jeff Hodges)

Status: resolved. (11/12/02)Based on my research I propose that the DN MUST exist in the directory when the DN form of sasl creds is used. I have made this proposal to the ldapbis mailing list.

(11/21/02) Feedback from mailing list has proposed removing this paragraph entirely because (1) explicit assertion of authorization identity should only be done when proxying (2) mapping of the asserted authorization identity is implementation specific and policy driven [SASL] section 4.2, and (3) keeping this paragraph is not required for interoperability.

G.19. DN used in conjunction with SASL mechanism

We need to specify whether the DN field in Bind operation can/cannot be used when SASL mechanism is specified. (source: RL Bob)

Status: resolved. (-03) Based on ldapbis WG discussion at IETF52 two sentences were added to section 4.3 indicating that clients SHOULD NOT send a DN value when binding with the sasl choice and servers SHALL ignore any value received in this circumstance. During edits for -04 version of draft it was noted that [Protocol] section 4.2 conflicts with this draft. The editor of [Protocol] has been notified of the discrepancy, and they have been handled.

G.20. Bind states

Differences between unauthenticated and anonymous. There are four states you can get into. One is completely undefined (this is now explicitly called out in [Protocol]). This text needs to be moved from [Protocol] to this draft. (source: Jim Sermersheim)

Status: Resolved. There are four states: (1) no name, no password (anon); (2) name, no password (anon); (3) no name, password (invalid); (4) name, password (simple bind). States 1, 2, and 4 are called out in [AuthMeth]. State 3 is called out in [Protocol]; this seems appropriate based on review of alternatives.

G.21. Misuse of unauthenticated access

Add a security consideration that operational experience shows that clients can misuse unauthenticated access (simple bind with name but no password). Servers SHOULD by default reject authentication requests that have a DN with an empty password with an error of invalidCredentials. (Source: Kurt Zeilenga and Chris Newman (Sun))

Status: Resolved. Added to security considerations in -03.

G.22. Need to move StartTLS protocol information to [Protocol]

Status: Resolved. Removed Sections 5.1, 5.2, and 5.4 for -04 and they are [Protocol] -11.

G.23. Split Normative and Non-normative references into separate
sections.

Status: Resolved. Changes made in -04

G.24. What is the authentication state if a Bind operation is
abandoned?

Status: Resolved.

(3/24/03) This following text appears in section 4.2.1 of [Protocol] revision -13 to cover what happens if a bind operation is abandoned:

A failed or abandoned Bind Operation has the effect of leaving the connection in an anonymous state. To arrive at a known authentication state after abandoning a bind operation, clients may unbind, rebind, or make use of the BindResponse.

(6/28/03): The state table in section 6 of [AuthMeth] has been updated to reflect this wording.

G.25. Difference between checking server hostname and server's
canonical DNS name in Server Identity Check?

Section 4.1.6: I now understand the intent of the check (prevent man-in-the-middle attacks). But what is the subtle difference between the "server hostname" and the "server's canonical DNS name"? (Source: Tim Hahn)

Status: Resolved.

(11/12/02) Sent suggested wording change to this paragraph to the ldapbis mail list and also asked for opinion as to whether we should discuss the distinction between server DNS hostname and server canonical DNS hostname in [AuthMeth].

(11/21/02): RL Bob Morgan will provide wording that allows derivations of the name that are provided securely.

(6/28/03): posted to the WG list asking Bob or any other WG member who is knowledgeable about the issues involved to help me with wording or other information I can use to make this change and close the work item.

(10/08/03): Based on WG list feedback, I've updated this text to read what I judge to be the WG consensus, "The client MUST use the server provided by the user (or other trusted entity) as the value to compare against the server name as expressed in the server's certificate. A hostname derived from the user input is to be considered provided by the user only if derived in a secure fashion (e.g., DNSSEC)."

G.26. Server Identity Check using servers located via SRV records

Section 4.1.6: What should be done if the server was found using SRV records based on the "locate" draft/RFC? (Source: Tim Hahn).

Status: Resolved. Section 5 of draft-ietf-ldapext-locate-08 specifically calls out how the server identity should be performed if the server is located using the method defined in that draft. This is the right location for this information, and the coverage appears to be adequate.

G.27 Inconsistency in effect of TLS closure on LDAP association.

Section 4.4.1 of authmeth -03 (section 4.1 of RFC2830) states that TLS closure alert will leave the LDAP association intact. Contrast this with Section 4.5.2 (section 5.2 of RFC2830) that says that the closure of the TLS connection MUST cause the LDAP association to move to an anonymous authentication.

Status: Resolved. (11/12/02) This is actually a [Protocol] issue because these sections have now been moved to [Protocol] -11. I have proposed the following text for Section 4.4.1 of [AuthMeth] -03 (section 4.13.3.1 of [Protocol]) to resolve this apparent discrepancy:

"Either the client or server MAY terminate the TLS connection on an
LDAP association by sending a TLS closure alert. The LDAP
connection remains open for further communication after TLS closure

occurs although the authentication state of the LDAP connection is affected (see [AuthMeth] section 4.2.2).

(11/21/02): resolution to this is expected in [Protocol] -12

(06/28/03): [Protocol]-15 clarifies that a TLS closure alert terminates the TLS connection while leaving the LDAP connection intact. The authentication state table in [AuthMeth] specifies the effect on the LDAP association.

G.28 Ordering of external sources of authorization identities

Section 4.3.2 implies that external sources of authorization identities other than TLS are permitted. What is the behavior when two external sources of authentication credentials are available (e.g. TLS and IPsec are both present (is this possible?)) and a SASL EXTERNAL Bind operation is performed?

Status: resolved. 11/20/02: Resolved by Section 4.2 of [SASL] which states that the decision to allow or disallow the asserted identity is based on an implementation defined policy.

G.29 Rewrite of Section 9, TLS Ciphersuites

This section contains anachronistic references and needs to be updated/rewritten in a way that provides useful guidance for future readers in a way that will transcend the passage of time.

Status: Resolved. (6/28/03): Rewrote the section to cover the general issues and considerations involved in selecting TLS ciphersuites.

G.30 Update to Appendix A, Example Deployment Scenarios

This section needs to be updated to indicate which security mechanisms and/or combinations of security mechanisms described elsewhere in the document can provide the types of protections suggested in this appendix.

G.31 Use of PLAIN SASL Mechanism

At least one LDAP server implementer has found the SASL "PLAIN" mechanism useful in authenticating to legacy systems that do not represent authentication identities as DNs. Section 3.3.1 appears to implicitly disallow the use of the SASL "PLAIN" mechanism with LDAP. Should we allow the use of this mechanism? I.e. is this "SASL" "PLAIN" MUST NOT be used with LDAP, or is it simply that LDAP doesn't define bindings for these mechanism. If SASL "PLAIN" is allowed, the following adjustments will be needed to section 3.3.1: (a) change section heading, (b) remove reference to "PLAIN" in the

section, (c) ensure wording of last sentence regarding non-DN
AuthZIDs is consistent with rest of the section.

Status: Resolved.

(6/28/03): email to WG list stating issue and asking if we should remove the reference to SASL "PLAIN".

For -07 draft I've generalized the SASL profile in section 3.3 to allow any SASL mechanism.

G.32 Clarification on use of SASL mechanisms

Section 3.3.1: BTW, what _are_ the "ANONYMOUS" and "PLAIN" SASL mechanisms? They are not defined in RFC2222. If you refer to other SASL mechanisms than those in rfc2222, Maybe you should only list which mechanisms _are_used, instead of which ones are _not. (Source: Hallvard Furuseth)

I (Kurt Zeilenga) note[s] as well that the ANONYMOUS/PLAIN section (4.2) should
be deleted. ANONYMOUS and PLAIN, like in other mechanism,
can be used in LDAP if a) supported and b) enabled. I note
that they each offer capabilities not found in their simple
bind equivalents (and hence are used in some deployments).
For example, PLAIN (over TLS) is quite useful when interacting
with legacy authentication subsystems. (Source: Kurt Zeilenga)

Status: Resolved.

For -07 draft I've generalized the SASL profile in section 3.3 to allow any SASL mechanism.

G.33 Clarification on use of password protection based on AuthZID form

Section 3.3.1: "If an authorization identity of a form different from a DN is requested by the client, a mechanism that protects the password in transit SHOULD be used." What has that to do with DNs? A mechanism that protects the password in transit should be used in any case, shouldn't it?

G.34 Clarification on use of matching rules in Server Identity Check

The text in section 4.1.6 isn't explicit on whether all rules apply to both CN and dNSName values. The text should be clear as to which rules apply to which values.... in particular, the wildcard rules. (Source: Kurt Zeilenga)

G.35 Requested Additions to Security Considerations

Requested to mention hostile servers which the user might have been fooled to into contacting. Which mechanisms that are standardized by the LDAP standard do/do not disclose the user's password to the server? (Or to servers doing man-in-the-middle attack? Or is that a stupid question?)

Requested to mention denial of service attacks.

Requested list of methods that need/don't need the server to know the user's plaintext password. (I say 'know' instead of 'store' because it could still store the password encrypted, but in a way which it knows how to decrypt.)

(Source: Hallvard Furuseth)

G.36 Add reference to definition of DIGEST-MD5

Need a reference to the definition of DIGEST-MD5 SASL mechanism in section 7.2 (Source: Hallvard Furuseth)

Status: Resolved. A reference to to the DIGEST-MD5 SASL mechanism, [DigestAuth], is included in the -07 revision.

G.37 Clarification on procedure for certificate-based authentication

8.1. Certificate-based authentication with TLS states: "Following the successful completion of TLS negotiation, the client will send an LDAP bind request with the SASL "EXTERNAL" mechanism." Is this immediately following, or just some time later? Should the wording, "the client will send..." actually read, "the client MUST send..."?

G.38 Effect of StartTLS on authentication state

Should the server drop all knowledge of connection, i.e. return to anonymous state, if it gets a StartTLS request on a connection that has successfully bound using the simple method?

G.39 Be sure that there is a consideration in [SCHEMA] that discusses multiple password values in userPassword

Allowing multiple values obviously does raise a number of security considerations and these need to be discussed in the document.

Certainly applications which intend to replace the userPassword with new value(s) should use modify/replaceValues (or modify/deleteAttribute+addAttribute). Additionally, server implementations should be encouraged to provide administrative controls which, if enabled, restrict userPassword to one value.

G.40. Clarify need to verify mapping between authentication identity and resulting authorization identity on implicit assertion of AuthZID.

4.2.2.3. Error Conditions

"For either form of assertion, the server MUST verify that the client's authentication identity as supplied in its TLS credentials is permitted to be mapped to the asserted authorization identity."

This makes sense for the explicit assertion case, but seems to be ambiguous for the implicit case.
IMHO, the mapping can be done as two steps:
a). deriving LDAP authentication identity from TLS credentials; If t this steps fails, EXTERNAL mechanism returns failure.
b). verify that the authorization identity is allowed for the derived authentication identity. This is always "noop" for the implicit case.
I am not sure that the text is saying this.
(Source: Alexey Melnikov email 8/1/2003 5:30:43 PM)

Status: Resolved in -07. After reading the comments and the text of the draft, I believe that this should be clarified. The local policy used to map the AuthNID to the AuthZID in the implicit case is sufficient and that no additional verification is useful or needed. This text has been moved to apply only to the explicit assertion case.

G.41. Section 7.2 contains unnecessary and misleading detail.

" I am not sure why this section is required in the document. DIGEST-MD5 is defined in a separate document and there should be nothing magical about its usage in LDAP. If DIGEST-MD5 description creates confusion for LDAP implementors, let's fix the DIGEST-MD5 document! Also, this section tries to redefine DIGEST-MD5 behavior, which is explicitly prohibited by the SASL specification."
(Source: Alexey Melnikov: email 8/1/2003 5:30:43 PM)

Status: Resolved.

After reading the comments and the text of the draft plus the related text in draft-ietf-sasl-rfc2831bis-02.txt plus http://www.ietf.org/internet-drafts/draft-ietf-sasl-rfc2222bis-02.txt, I am inclined to agree with Alexey. In -07 I rewrote section 3.3 (SASL mechanisms) to match the profiling requirements rfc2831bis. I then dramatically reduced the material in section 7.2 to a bare minimum and let the SASL profile stand on its own.

G.42. Does change for G.41 cause interoperability issue?

There is one issue with the way the authmeth draft is currently written that changes the SASL DIGEST-MD5 behavior on the way the server responds with the subsequent authentication information . This has been documented in this fashion since RFC 2829 (section 6.1) was originally published and may cause an interoperability issue at this point if it changed to follow the DIGEST-MD5 spec (as it was in -07 of AuthMeth). Take this issue to the list.

Status: Resolved

(10/08/03) This item was discussed on the WG list between 5/2/03 and 5/9/03. Consensus apppears to support the notion that RFC 2829 was in error and that the semantics of RFC 2831 are correct and should be reflected in authmeth. This is already the case as of the -07 draft.

G.43. DIGEST-MD5 Realms recommendations for LDAP

From http://www.ietf.org/internet-drafts/draft-ietf-sasl-rfc2222bis-02.txt: A protocol profile SHOULD provide a guidance how realms are to be constructed and used in the protocol and MAY further restrict its syntax and protocol-specific semantics."

I don't believe that any such guidance exists within the LDAP TS. The most likely place for this to reside is in the authmeth draft.

Related email from Alexey Melnikov (8/4/2003 1:08:40 PM):

"The problem I have with the document is that it references realm without explaining what it is (or at least some examples of valid values). For LDAP, some recommendations should be given. For example:
1). Use a hardcoded string as the realm (one of the implementations I worked on was doing that)
2). Use hostname (realm==host) or domain/cluster name (realm includes multiple hosts).
3). Use a node in DIT above user entry, for example for "cn=Barbara Jensen, ou=Accounting, o=Ace Industry, c=US"
and "cn=John Doe, ou=Accounting, o=Ace Industry, c=US" realm can be "ou=Accounting, o=Ace Industry, c=US"
(or "o=Ace Industry, c=US"); for "cn=Gern Jensen, ou=Product Testing,o=Ace Industry, c=US" realm can be "ou=Product Testing, o=Ace Industry, c=US".

Of course other choices are possible.

Alexey

To summarize: I'd like authmeth to define a realm name for use with Digest-MD5 that corresponds to LDAP DNs known to this server. Authzid is okay, but perhaps could be better put into context.

John McMeeking (5/12/2003)

G.44. Use of DNs in usernames and realms in DIGEST-MD5

In reading the discussion on the mailing list, I reach the following conclusions:

DIGEST-MD5 username and realm are simple strings. The syntax of
these strings allows strings that look like DNs in form, however,
DIGEST-MD5 treats them a simple strings for comparision purposes.
For example, the DNs cn=roger, o=US and cn=roger,o=us are equivalent

when being compared semantically as DNs, however, these would be considered two different username values in DIGEST-MD5 because simple octet-wise semantics (rather than DN semantics) are used to compare username values in DIGEST-MD5. Ditto for realm values.

Status: Resolved.

In -07 revision I added notes to implementors expressing this issue in section 7.2.

G.45: Open Issue: Is Simple+TLS mandatory to implement?

Going forward, it would be much better to clarify that simple +TLS is to be used for DN/password credentials and DIGEST-MD5 (or PLAIN+TLS) be used for username/password credentials. (Kurt Zeilenga, 5/12/2003)

I don't believe you can mandate simple/TLS! At the time RFC 2829 was debated, a large number on the WG wanted this. They did not get their way because of the complexity of the solution. It was argued that a password-based method would be better. I think they believed it would still be DN/password, though. (Ron Ramsay, 5/12/2003)

This was officially opened as an issue by WG co-chair Kurt Zeilenga on 5/12/03. Little direct discussion has occurred since, however there has been significant discussion on the use of DN values as the username for DIGEST-MD5.