Network Working Group
Request for Comments: 2830
Category: Standards Track

J. Hodges
Oblix Inc.
R. Morgan
Univ of Washington
M. Wahl
Sun Microsystems, Inc.

May 2000

Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security

Status of this Memo

Copyright Notice

Abstract

This document defines the "Start Transport Layer Security (TLS) Operation" for LDAP [LDAPv3, TLS]. This operation provides for TLS establishment in an LDAP association and is defined in terms of an LDAP extended request.

1. Conventions Used in this Document

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 [ReqsKeywords].

2. The Start TLS Request

This section describes the Start TLS extended request and extended response themselves: how to form the request, the form of the response, and enumerates the various result codes the client MUST be prepared to handle.

The section following this one then describes how to sequence an overall Start TLS Operation.

2.1. Requesting TLS Establishment

A client may perform a Start TLS operation by transmitting an LDAP PDU containing an ExtendedRequest [LDAPv3] specifying the OID for the Start TLS operation:

1.3.6.1.4.1.1466.20037

An LDAP ExtendedRequest is defined as follows:

```
ExtendedRequest ::= [APPLICATION 23] SEQUENCE {
        requestName             [0] LDAPOID,
        requestValue            [1] OCTET STRING OPTIONAL }
```

A Start TLS extended request is formed by setting the requestName field to the OID string given above. The requestValue field is absent. The client MUST NOT send any PDUs on this connection following this request until it receives a Start TLS extended response.

When a Start TLS extended request is made, the server MUST return an LDAP PDU containing a Start TLS extended response. An LDAP ExtendedResponse is defined as follows:

```
ExtendedResponse ::= [APPLICATION 24] SEQUENCE {
        COMPONENTS OF LDAPResult,
        responseName     [10] LDAPOID OPTIONAL,
        response         [11] OCTET STRING OPTIONAL }
```

A Start TLS extended response MUST contain a responseName field which MUST be set to the same string as that in the responseName field present in the Start TLS extended request. The response field is absent. The server MUST set the resultCode field to either success or one of the other values outlined in section 2.3.

2.2. "Success" Response

If the ExtendedResponse contains a resultCode of success, this indicates that the server is willing and able to negotiate TLS. Refer to section 3, below, for details.

2.3. Response other than "success"

If the ExtendedResponse contains a resultCode other than success, this indicates that the server is unwilling or unable to negotiate TLS.

If the Start TLS extended request was not successful, the resultCode will be one of:

operationsError (operations sequencing incorrect; e.g. TLS already established)

protocolError (TLS not supported or incorrect PDU structure)

referral (this server doesn't do TLS, try this one)

unavailable (e.g. some major problem with TLS, or server is shutting down)

The server MUST return operationsError if the client violates any of the Start TLS extended operation sequencing requirements described in section 3, below.

If the server does not support TLS (whether by design or by current configuration), it MUST set the resultCode to protocolError (see section 4.1.1 of [LDAPv3]), or to referral. The server MUST include an actual referral value in the LDAP Result if it returns a resultCode of referral. The client's current session is unaffected if the server does not support TLS. The client MAY proceed with any LDAP operation, or it MAY close the connection.

The server MUST return unavailable if it supports TLS but cannot establish a TLS connection for some reason, e.g. the certificate server not responding, it cannot contact its TLS implementation, or if the server is in process of shutting down. The client MAY retry the StartTLS operation, or it MAY proceed with any other LDAP operation, or it MAY close the connection.

3. 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 5.

3.1. Requesting to Start TLS on an LDAP Association

The client MAY send the Start TLS extended request at any time after establishing an LDAP association, except that in the following cases the client MUST NOT send a Start TLS extended request:

- if TLS is currently established on the connection, or
- during a multi-stage SASL negotiation, or
- if there are any LDAP operations outstanding on the connection.

The result of violating any of these requirements is a resultCode of operationsError, as described above in section 2.3.

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 with a confidentialityRequired or strongAuthRequired result. The client MAY send a Start TLS extended request, or it MAY choose to close the connection.

3.2. 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 above, if it is unable.

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 [TLS].

3.3. TLS Version Negotiation

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

3.4. 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 privacy level achieved. Ascertaining the TLS connection’s privacy 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 privacy is not high enough for it to continue, it SHOULD gracefully close the TLS connection immediately after the TLS negotiation has completed (see sections 4.1 and 5.2, below).

The client MAY attempt to Start TLS again, or MAY send an unbind request, or send any other LDAP request.

3.5. Assertion of Client's Authorization Identity

The client MAY, upon receipt of a Start TLS extended 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]. See section 5.1.2, below.

3.6. 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 hostname it used to open the LDAP connection as the value to compare against the server name as expressed in the server's certificate. The client MUST NOT use the server's canonical DNS name or any other derived form of name.

- 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.

E.g. *.bar.com would match a.bar.com, b.bar.com, etc. but not 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.

3.7. Refresh of Server Capabilities Information

The client MUST refresh any cached server capabilities information (e.g. from the server's root DSE; see section 3.4 of [LDAPv3]) upon TLS session establishment. This is necessary to protect against active-intermediary attacks which may have altered any server capabilities information retrieved prior to TLS establishment. The server MAY advertise different capabilities after TLS establishment.

4. Closing a TLS Connection

4.1. Graceful Closure

Either the client or server MAY terminate the TLS connection on an LDAP association by sending a TLS closure alert. This will leave the LDAP association intact.

Before closing a TLS connection, the client MUST either wait for any outstanding LDAP operations to complete, or explicitly abandon them [LDAPv3].

After the initiator of a close has sent a closure alert, it MUST discard any TLS messages until it has received an alert from the other party. It will cease to send TLS Record Protocol PDUs, and following the receipt of the alert, MAY send and receive LDAP PDUs.

The other party, if it receives a closure alert, MUST immediately transmit a TLS closure alert. It will subsequently cease to send TLS Record Protocol PDUs, and MAY send and receive LDAP PDUs.

4.2. Abrupt Closure

Either the client or server MAY abruptly close the entire LDAP association and any TLS connection established on it by dropping the underlying TCP connection. A server MAY beforehand send the client a Notice of Disconnection [LDAPv3] in this case.

5. 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. Lastly, the effects of closing the TLS connection are described.

Authorization identities and related concepts are defined in [AuthMeth].

5.1. TLS Connection Establishment Effects

5.1.1. Default Effects

Upon establishment of the TLS connection 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 (see [TLS]).

5.1.2. Client Assertion of Authorization Identity

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.

5.1.2.1. 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, LDAPv3] 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.

5.1.2.2. 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, LDAPv3] that SHALL include the credentials octet string. This string MUST be constructed as documented in section 9 of [AuthMeth].

5.1.2.3. 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. The server MUST reject the Bind operation with an invalidCredentials resultCode in the Bind response if the client is not so authorized.

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 [IPSEC]), 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 result code of inappropriateAuthentication.

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

5.2. TLS Connection Closure Effects

Closure of the TLS connection 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 connection establishment.

6. 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, as described in [TLS].

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.

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. 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 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.

7. Acknowledgements

The authors thank Tim Howes, Paul Hoffman, John Kristian, Shirish Rai, Jonathan Trostle, Harald Alvestrand, and Marcus Leech for their contributions to this document.

8. References

[AuthMeth] Wahl, M., Alvestrand, H., Hodges, J. and R. Morgan, "Authentication Methods for LDAP", RFC 2829, May 2000.

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

[LDAPv3] Wahl, M., Kille S. and T. Howes, "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

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

[SASL] Myers, J., "Simple Authentication and Security Layer (SASL)", RFC 2222, October 1997.

[TLS] Dierks, T. and C. Allen. "The TLS Protocol Version 1.0", RFC 2246, January 1999.

9. Authors' Addresses

Jeff Hodges
Oblix, Inc.
18922 Forge Drive
Cupertino, CA 95014
USA

Phone: +1-408-861-6656
EMail: JHodges@oblix.com

RL "Bob" Morgan
Computing and Communications
University of Washington
Seattle, WA
USA

Phone: +1-206-221-3307
EMail: rlmorgan@washington.edu

Mark Wahl
Sun Microsystems, Inc.
8911 Capital of Texas Hwy #4140
Austin TX 78759
USA

EMail: M.Wahl@innosoft.com

10. Intellectual Property Rights Notices

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

11. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.