Network Working Group
Request for Comments: 4531
Category: Experimental

K. Zeilenga
OpenLDAP Foundation

June 2006

Lightweight Directory Access Protocol (LDAP) Turn Operation

Status of This Memo

Copyright Notice

Abstract

This specification describes a Lightweight Directory Access Protocol (LDAP) extended operation to reverse (or "turn") the roles of client and server for subsequent protocol exchanges in the session, or to enable each peer to act as both client and server with respect to the other.

Table of Contents

1. Background and Intent of Use

The Lightweight Directory Access Protocol (LDAP) [RFC4510][RFC4511] is a client-server protocol that typically operates over reliable octet-stream transports, such as the Transport Control Protocol (TCP). Generally, the client initiates the stream by connecting to the server's listener at some well-known address.

There are cases where it is desirable for the server to initiate the stream. Although it certainly is possible to write a technical specification detailing how to implement server-initiated LDAP sessions, this would require the design of new authentication and other security mechanisms to support server-initiated LDAP sessions.

Instead, this document introduces an operation, the Turn operation, which may be used to reverse the client-server roles of the protocol peers. This allows the initiating protocol peer to become the server (after the reversal).

As an additional feature, the Turn operation may be used to allow both peers to act in both roles. This is useful where both peers are directory servers that desire to request, as LDAP clients, that operations be performed by the other. This may be useful in replicated and/or distributed environments.

This operation is intended to be used between protocol peers that have established a mutual agreement, by means outside of the protocol, that requires reversal of client-server roles, or allows both peers to act both as client and server.

1.1. Terminology

Protocol elements are described using ASN.1 [X.680] with implicit tags. The term "BER-encoded" means the element is to be encoded using the Basic Encoding Rules [X.690] under the restrictions detailed in Section 5.1 of [RFC4511].

2. Turn Operation

The Turn operation is defined as an LDAP-Extended Operation [Protocol, Section 4.12] identified by the 1.3.6.1.1.19 OID. The function of the Turn Operation is to request that the client-server roles be reversed, or, optionally, to request that both protocol peers be able to act both as client and server in respect to the other.

2.1. Turn Request

The Turn request is an ExtendedRequest where the requestName field contains the 1.3.6.1.1.19 OID and the requestValue field is a BER-encoded turnValue:

```
turnValue ::= SEQUENCE {
     mutual         BOOLEAN DEFAULT FALSE,
     identifier     LDAPString
}
```

A TRUE mutual field value indicates a request to allow both peers to act both as client and server. A FALSE mutual field value indicates a request to reserve the client and server roles.

The value of the identifier field is a locally defined policy identifier (typically associated with a mutual agreement for which this turn is be executed as part of).

2.2. Turn Response

A Turn response is an ExtendedResponse where the responseName and responseValue fields are absent. A resultCode of success is returned if and only if the responder is willing and able to turn the session as requested. Otherwise, a different resultCode is returned.

3. Authentication

This extension's authentication model assumes separate authentication of the peers in each of their roles. A separate Bind exchange is expected between the peers in their new roles to establish identities in these roles.

Upon completion of the Turn, the responding peer in its new client role has an anonymous association at the initiating peer in its new server role. If the turn was mutual, the authentication association of the initiating peer in its pre-existing client role is left intact at the responding peer in its pre-existing server role. If the turn was not mutual, this association is void.

The responding peer may establish its identity in its client role by requesting and successfully completing a Bind operation.

The remainder of this section discusses some authentication scenarios. In the protocol exchange illustrations, A refers to the initiating peer (the original client) and B refers to the responding peer (the original server).

3.1. Use with TLS and Simple Authentication

```
A->B: StartTLS Request
B->A: StartTLS(success) Response
A->B: Bind(Simple(cn=B,dc=example,dc=net,B's secret)) Request
B->A: Bind(success) Response
A->B: Turn(TRUE,"XXYYZ") Request
B->A: Turn(success) Response
B->A: Bind(Simple(cn=A,dc=example,dc=net,A's secret)) Request
A->B: Bind(success) Response
```

In this scenario, TLS (Transport Layer Security) [RFC4346] is started and the initiating peer (the original client) establishes its identity with the responding peer prior to the Turn using the DN/password mechanism of the Simple method of the Bind operation. After the turn, the responding peer, in its new client role, establishes its identity with the initiating peer in its new server role.

3.2. Use with TLS and SASL EXTERNAL

```
A->B: StartTLS Request
B->A: StartTLS(success) Response
A->B: Bind(SASL(EXTERNAL)) Request
B->A: Bind(success) Response
A->B: Turn(TRUE,"XXYYZ") Request
B->A: Turn(success) Response
B->A: Bind(SASL(EXTERNAL)) Request
A->B: Bind(success) Response
```

In this scenario, TLS is started (with each peer providing a valid certificate), and the initiating peer (the original client) establishes its identity through the use of the EXTERNAL mechanism of the SASL (Simple Authentication and Security Layer) [RFC4422] method of the Bind operation prior to the Turn. After the turn, the responding peer, in its new client role, establishes its identity with the initiating peer in its new server role.

3.3. Use of Mutual Authentication and SASL EXTERNAL

A number of SASL mechanisms, such as GSSAPI [SASL-K5], support mutual authentication. The initiating peer, in its new server role, may use the identity of the responding peer, established by a prior authentication exchange, as its source for "external" identity in subsequent EXTERNAL exchange.

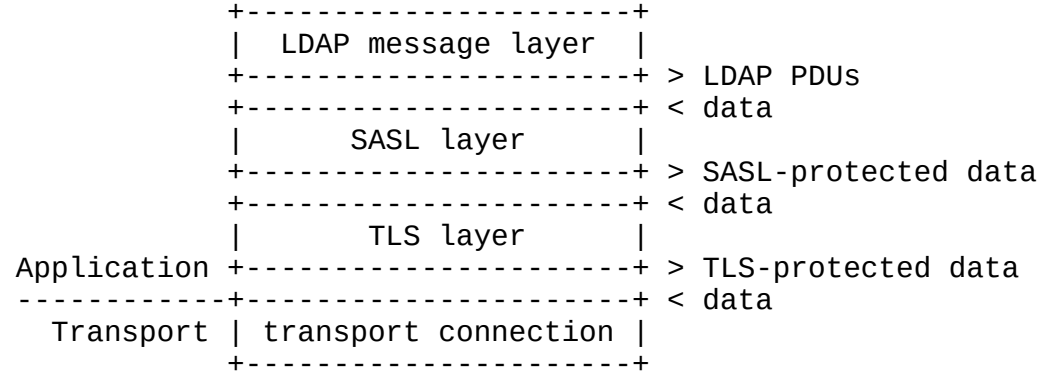
```
A->B: Bind(SASL(GSSAPI)) Request
<intermediate messages>
```

```
B->A: Bind(success) Response
A->B: Turn(TRUE,"XXYYZ") Request
B->A: Turn(success) Response
B->A: Bind(SASL(EXTERNAL)) Request
A->B: Bind(success) Response
```

In this scenario, a GSSAPI mutual-authentication exchange is completed between the initiating peer (the original client) and the responding server (the original server) prior to the turn. After the turn, the responding peer, in its new client role, requests that the initiating peer utilize an "external" identity to establish its LDAP authorization identity.

4. TLS and SASL Security Layers

As described in [RFC4511], LDAP supports both Transport Layer Security (TLS) [RFC4346] and Simple Authentication and Security Layer (SASL) [RFC4422] security frameworks. The following table illustrates the relationship between the LDAP message layer, SASL layer, TLS layer, and transport connection within an LDAP session.



This extension does not alter this relationship, nor does it remove the general restriction against multiple TLS layers, nor does it remove the general restriction against multiple SASL layers.

As specified in [RFC4511], the StartTLS operation is used to initiate negotiation of a TLS layer. If a TLS is already installed, the StartTLS operation must fail. Upon establishment of the TLS layer, regardless of which peer issued the request to start TLS, the peer that initiated the LDAP session (the original client) performs the "server identity check", as described in Section 3.1.5 of [RFC4513], treating itself as the "client" and its peer as the "server".

As specified in [RFC4422], a newly negotiated SASL security layer replaces the installed SASL security layer. Though the client/server

roles in LDAP, and hence SASL, may be reversed in subsequent exchanges, only one SASL security layer may be installed at any instance.

5. Security Considerations

Implementors should be aware that the reversing of client/server roles and/or allowing both peers to act as client and server likely introduces security considerations not foreseen by the authors of this document. In particular, the security implications of the design choices made in the authentication and data security models for this extension (discussed in Sections 3 and 4, respectively) are not fully studied. It is hoped that experimentation with this extension will lead to better understanding of the security implications of these models and other aspects of this extension, and that appropriate considerations will be documented in a future document. The following security considerations are apparent at this time.

Implementors should take special care to process LDAP, SASL, TLS, and other events in the appropriate roles for the peers. Note that while the Turn reverses the client/server roles with LDAP, and in SASL authentication exchanges, it does not reverse the roles within the TLS layer or the transport connection.

The responding server (the original server) should restrict use of this operation to authorized clients. Client knowledge of a valid identifier should not be the sole factor in determining authorization to turn.

Where the peers except to establish TLS, TLS should be started prior to the Turn and any request to authenticate via the Bind operation.

LDAP security considerations [RFC4511][RFC4513] generally apply to this extension.

6. IANA Considerations

The following values [RFC4520] have been registered by the IANA.

6.1. Object Identifier

The IANA has assigned an LDAP Object Identifier to identify the LDAP Turn Operation, as defined in this document.

```
Subject: Request for LDAP Object Identifier Registration
Person & email address to contact for further information:
     Kurt Zeilenga <kurt@OpenLDAP.org>
Specification: RFC 4531
Author/Change Controller: Author
Comments:
     Identifies the LDAP Turn Operation
```

6.2. LDAP Protocol Mechanism

The IANA has registered the LDAP Protocol Mechanism described in this document.

```
Subject: Request for LDAP Protocol Mechanism Registration
Object Identifier: 1.3.6.1.1.19
Description: LDAP Turn Operation
Person & email address to contact for further information:
     Kurt Zeilenga <kurt@openldap.org>
Usage: Extended Operation
Specification: RFC 4531
Author/Change Controller: Author
Comments: none
```

7. References

7.1. Normative References

[RFC4346] Dierks, T. and, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

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

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

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

[RFC4513] Harrison, R., Ed., "Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms", RFC 4513, June 2006.

[X.680] International Telecommunication Union - Telecommunication Standardization Sector, "Abstract Syntax Notation One (ASN.1) - Specification of Basic Notation", X.680(2002) (also ISO/IEC 8824-1:2002).

[X.690] International Telecommunication Union - Telecommunication Standardization Sector, "Specification of ASN.1 encoding rules: Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER)", X.690(2002) (also ISO/IEC 8825-1:2002).

7.2. Informative References

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

[SASL-K5] Melnikov, A., Ed., "The Kerberos V5 ("GSSAPI") SASL Mechanism", Work in Progress, May 2006.

Author’s Address

Kurt D. Zeilenga
OpenLDAP Foundation

EMail: Kurt@OpenLDAP.org

Full Copyright Statement

Intellectual Property

Acknowledgement

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