SCRAM in LDAP

Better Password-based Authentication

Kurt Zeilenga
Isode Limited
What is SCRAM?

- Salted Challenge Response Authentication Mechanism

- SCRAM-SHA-1-PLUS

- An improved SASL mechanism for password authentication of the user.

- Designed to mitigate threats of the modern Internet and Enterprise networks.
Why is it “better”?

- Let’s review the history of password-based authentication in LDAP
Password authentication at the dawn of LDAP

- LDAP Simple Bind with DN & Password (in clear) without any confidentiality protection and without authentication of the server.
- Password stored in clear user entry
- Octet wise comparison (ASCII assumed)
LDAPv2 over SSL

- Data confidentiality protection needed to protect passwords during LDAP Simple Bind, as well as protect subsequent data exchange.

- While SSL provided for complete client verification of server X.509 certificate, server verification not commonly performed in early years.
Enter SASL

- RFC 2251 introduced SASL authentication to LDAP in 1997.
- Password-based SASL mechanisms at this time included PLAIN and CRAM-MD5.
• authcid + password
• plus an authzid for identity assumption

• Not much reason to use it (then or now) in LDAP, and generally not used in LDAP.
CRAM-MD5

- Primarily introduced to protect password during exchange.
- Used heavily in email protocols.
- Not generally used in LDAP as LDAP uses SSL to protect passwords.
- LDAP deployers generally wanted to not only to protect passwords during exchange, but protect all LDAP data. Hence, CRAM-MD5 did not offer much value.
DIGEST-MD5

- Introduced to address weaknesses of CRAM-MD5 and to provide compatibility with HTTP Digest.
- Provides for mutual authentication, but rarely properly implemented, rarely used.
- Provides for data integrity/confidential protection, but wasn’t well implemented and is not widely deployed.
- Provided for hashed password storage, but is a password equivalent.
Hashed Password Storage

• Introduced by RFC 2307 (Experimental)
• Compatible with use of Simple and PLAIN mechanisms and operating system login mechanism (e.g., crypt(3))
• Incompatible with DIGEST-MD5.
Enter RFC 2829/2930

• Detailed use of SASL and TLS in LDAP, including introduction of the Start TLS operation and the DIGEST-MD5 mandatory-to-implement requirement.

• Strongly encouraged use of TLS to protect Simple Bind (server authentication, confidentiality protection).
LDAP Password Modify

• RFC 3062 (Standard Track)
• Introduced to eliminate need for client to know where the user entry is, or where the password is, or how it’s to be prepared for storage, etc.
• Originally implemented in OpenLDAP, now broadly implemented (but not ubiquitous).
AuthPassword

Introduced

- RFC 3112 (Informational)
- Intended to address abuse of userPassword to hold hashed user passwords
- No known implementations
Enter RFC 4422/4510

• SASL and LDAPv3 Revised
• Detailed handling of international user names and passwords
• LDAP Mandatory-to-Implement authentication mechanism changed to StartTLS+Simple w/ DN & Password
Current Situation

- Simple Bind with DN & Password over TLS (startTLS or ldaps://) is ubiquitous.
- TLS certification checks generally implemented well but often not used or used improperly.
- Use of hashed password storage is common (often required), commonly newer RFC 2307 schemes such as Salted SHA1 or even Salted SHA2.
So what’s the problem?

- Users often send actual passwords without authenticating the server.

- User has to trust server with its actual password. Server can impersonate user elsewhere.

- Server has to assume client properly implements server authentication AND it’s properly used. Bad assumptions (especially the latter).
How does SCRAM address these problems?

- Actual password not sent. Derived value sent is not reusable.
- Mutual authentication: Server must produce a response that demonstrates it knowledge of the user credentials.
SCRAM
Password Storage

• Authentication information held in the server is not sufficient to impersonate the user, and is salted.

• <insert example>

• Simple Bind compatible (including same internationalization)
But we still have a problem...

- Server has to assume client properly implements server authentication AND it’s properly used. Bad assumptions (especially the latter).
And this is serious problem...

- Server not assured there is no man-in-middle.
Solution: SCRAM+ “Channel Bindings” to TLS

• Proves to the client that the SCRAM server entity is also in control of the server TLS end-point.

• Proves to the server that the SCRAM server entity is also in control of the client TLS end-point.
TCP → TLS → SASL → LDAP → Client

TCP → TLS → Relay

TCP → TLS → Server
Details

- **SCRAM**: RFC 5802
- **TLS Channel Bindings**: RFC xxxx
- **Further reading**: http://www.isode.com/whitepapers/scram.html
Questions