MyProxy Protocol

Status of This Memo

This memo provides information to the Grid community. Distribution is unlimited.

Abstract

This document describes a protocol for storing and retrieving X.509 proxy credentials [RFC3820] to and from a server. The protocol uses proxy delegation [ProxyDelegation] to transfer credentials without transferring private keys.

Contents

1. Introduction

The MyProxy protocol [MyProxy] supports storing and retrieving X.509 proxy credentials [RFC3820] to and from a server. The protocol allows long-lived keys to be secured on the server while allowing convenient access to short-lived proxy credentials as needed. The protocol was first implemented in September 2000 and has been widely used in grids since that time.

This document specifies the core MyProxy protocol as implemented by existing MyProxy software, representing the minimum requirements for an interoperable MyProxy implementation. The protocol is extensible, and protocol extensions may be specified in other documents in the future.

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 [RFC2119].

2. Initializing the Connection

Upon TCP connection establishment, the client and server perform the TLS protocol [RFC2246] to establish a private, authenticated channel. Server authentication via TLS is REQUIRED. The server's authenticated identity MUST have a Common Name of "fqhn", "host/fqhn", or "myproxy/fqhn", where "fqhn" is the fully qualified hostname of the server. Otherwise, the client MUST terminate the connection immediately. Clients and servers MUST support SSL 3.0 as described in [RFC2246] Section E. Clients and servers MUST NOT support SSL 2.0.

Client authentication via TLS is REQUIRED for all commands except the 'Get' command (COMMAND=0), for which TLS client authentication is OPTIONAL. The server SHOULD support client authentication with proxy credentials, i.e., the server's TLS implementation SHOULD understand the proxy certificate extensions [RFC3820]. The negotiated TLS channel MUST be private, i.e., the client and server MUST NOT negotiate the NULL TLS cipher suite.

Once the TLS handshake is complete and the secure channel has been established, the client sends one byte with zero value over the TLS channel. The server MUST ignore this first byte. The MyProxy application protocol then proceeds over the TLS channel as follows.

3. Messaging Overview

MyProxy protocol messages are NULL-terminated UTF-8 text strings containing one or more lines of ATTRIBUTE=VALUE statements, separated by the newline character 0x0a ('\n').

Client requests contain:

```
VERSION=MYPROXYv2
COMMAND=<decimal string>
USERNAME=<string>
PASSPHRASE=<string>
LIFETIME=<decimal string>
```

followed optionally by additional ATTRIBUTE=VALUE lines. The COMMAND line specifies the requested command. The USERNAME line specifies a string identifying the "account" for storing the proxy credential. This "account" need not correspond to any system account outside MyProxy. The PASSPHRASE line specifies a passphrase used to protect the stored proxy credential. It MUST be a minimum of 6 characters in length. The LIFETIME line specifies the

lifetime of the retrieved proxy credential in seconds, not to exceed one billion seconds. These parameters are specified in detail for each command in the following sections.

The server replies to the requests with the following to indicate success:

```
VERSION=MYPROXYv2
RESPONSE=0
```

or replies with the following to indicate error:

```
VERSION=MYPROXYv2
RESPONSE=1
ERROR=<error text>
ERROR=<error text>
...
```

There may be one or more lines of error text, with the intent that the client may concatenate them together (separated with carriage returns) for display. After sending an error response, the server MUST immediately close the connection.

For protocol extensibility, clients and servers MUST ignore lines in messages that they don't understand. Clients and servers MUST send lines in the order specified.

In cases where multiple rounds of request and reply messages are possible, they are presented sequentially (non-pipelined) below. Clients MAY send multiple request messages before receiving replies for improved performance. On error, the server MUST ignore any remaining incoming data and terminate the connection.

4. The 'Get' Command (COMMAND=0)

This command retrieves a proxy credential from the server. The client sends:

```
VERSION=MYPROXYv2
COMMAND=0
USERNAME=<username>
PASSPHRASE=<passphrase>
LIFETIME=<requested lifetime>
```

The USERNAME line identifies the account in which the credential to be retrieved has been stored. The PASSPHRASE line specifies the passphrase protecting the credential. The server MUST verify the passphrase and reject the request if the passphrase does not match. The LIFETIME is the requested proxy credential lifetime in seconds. The server MAY return a credential with a shorter lifetime according to its local policies.

The MyProxy server will then send a RESPONSE message. If the RESPONSE indicates failure (RESPONSE=1), the server will terminate the connection. If the RESPONSE indicates success (RESPONSE=0), the protocol proceeds as follows.

The client will generate a public/private key pair and send a NULL-terminated DER-encoded PKCS#10 [RFC2986] certificate request to the server. The server will then send a proxy certificate [RFC3820], containing the public key from the certificate request, signed by the private key of the stored credential, followed by the corresponding certificate chain, back to the client. The server will set the subject of the proxy certificate as specified in [RFC3820] Section 3.4, ignoring the subject in the client's certificate request. The format of the server's message is: one byte (unsigned binary) encoding the number of certificates to be sent (255 maximum), followed by

the newly signed proxy certificate, followed by the certificates in the chain. Each certificate is DER-encoded. The certificate chain MUST include the end-entity certificate signed by a CA, and MAY also include CA certificates.

The server will then send a standard RESPONSE message, and both client and server will close the connection.

5. The 'Put' Command (COMMAND=1)

This command stores a proxy credential on the server. The client sends:

```
VERSION=MYPROXYv2
COMMAND=1
USERNAME=<username>
PASSPHRASE=<passphrase>
LIFETIME=<lifetime>
```

The USERNAME line identifies the account in which to store the credential. The PASSPHRASE line specifies the passphrase for protecting the proxy credential. The passphrase MUST be at least 6 characters in length. The server MUST enforce this passphrase length restriction, and MAY optionally enforce additional passphrase quality policies. LIFETIME sets the maximum lifetime allowed for retrieved proxy credentials in seconds. The server MUST enforce this maximum lifetime on subsequent ‘Get’ requests.

Note that the passphrase does not authenticate the ‘Put’ request but will authenticate later ‘Get’ requests. The ‘Put’ request is authenticated via TLS.

The MyProxy server will then send a RESPONSE message. If the RESPONSE indicates failure (RESPONSE=1), the server will terminate the connection. If the RESPONSE indicates success (RESPONSE=0), the protocol proceeds as follows.

The server will generate a public/private key pair and send a NULL-terminated PKCS#10 [RFC2986] certificate request to the client. The client will then send a proxy certificate [RFC3820], containing the public key from the certificate request, signed by its private key, followed by the corresponding certificate chain, back to the server. The client will set the subject of the proxy certificate as specified in [RFC3820] Section 3.4, ignoring the subject in the server's certificate request. The format of the client's message is: one byte (unsigned binary) encoding the number of certificates to be sent (255 maximum), followed by the newly signed proxy certificate, followed by the certificates in the chain. Each certificate is DER-encoded. The certificate chain MUST include the end-entity certificate signed by a CA, and MAY also include CA certificates.

The server then stores the credentials and sends a standard RESPONSE message indicating whether the credentials were successfully stored. Then both client and server close the connection.

6. The 'Info' Command (COMMAND=2)

This command provides information about the existence of stored credentials. The client sends:

```
VERSION=MYPROXYv2
COMMAND=2
USERNAME=<username>
PASSPHRASE=PASSPHRASE
LIFETIME=0
```

USERNAME identifies the account for which information is requested. PASSPHRASE and LIFETIME are unused. The client SHOULD send the above values for PASSPHRASE and LIFETIME for backward compatibility. The server SHOULD ignore the PASSPHRASE and LIFETIME lines.

The MyProxy server will then send a RESPONSE message. If the credentials exist and match the client's authenticated identity, the server will send a successful response (RESPONSE=0). Otherwise, the server's response will indicate failure (RESPONSE=1). The client and server will then terminate the connection.

7. The 'Destroy' Command (COMMAND=3)

This command removes credentials stored on the server. The client sends:

```
VERSION=MYPROXYv2
COMMAND=3
USERNAME=<username>
PASSPHRASE=PASSPHRASE
LIFETIME=0
```

USERNAME identifies the account for the credential to be removed from the server. PASSPHRASE and LIFETIME are unused. The client SHOULD send the above values for PASSPHRASE and LIFETIME for backward compatibility. The server SHOULD ignore the PASSPHRASE and LIFETIME lines.

The MyProxy server will then send a standard RESPONSE message indicating whether the credentials were successfully removed. The client and server will then terminate the connection.

8. Security Considerations

The required use of server-authenticated TLS protects the MyProxy protocol against standard network-based attacks.

The MyProxy server must verify that the (TLS authenticated) client is authorized to perform requested operations and deny unauthorized requests, to (for example) protect against one user overwriting (via ‘Put’) or removing (via ‘Destroy’) another user’s credentials from the repository. Servers may also want to restrict the ‘Info’ command to authorized clients. In the case of the ‘Get’ command, where TLS client authentication is optional, the client must authenticate with the credential passphrase.

Care should be taken to maintain the integrity of client systems to protect against passphrase stealing attacks from keyboard sniffers, trojans, and similar techniques. Additionally, client-side trusted TLS certificates should be carefully configured to ensure the client is connecting to a MyProxy server with a certificate issued by a trusted authority.

The MyProxy credential repository must be well secured, as a large number of stored credentials could be vulnerable in the case of a server compromise. The server should be secured to the level of a Kerberos KDC.

MyProxy servers must encrypt stored credentials with the salted, user-chosen passphrase from the Put command, and servers must not store the passphrase. This forces an attacker who compromises the server to perform a brute-force, dictionary attack to decrypt the credentials or an online attack to obtain passphrases from Get commands on the server.

9. Acknowledgements

Jason Novotny, Steve Tuecke, and Von Welch designed the MyProxy protocol. Comments on drafts of this document from the GGF Steering Group and the GGF public comment period have significantly improved its quality.

Author Information

Jim Basney
National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign
1205 West Clark Street
Urbana, Illinois 61801
Email: jbasney@ncsa.uiuc.edu
Phone: (217) 244-1954
Fax: (217) 244-1987

Intellectual Property Statement

Full Copyright Notice

BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."

References

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

[RFC2246] T. Dierks and C. Allen, "The TLS Protocol Version 1.0," IETF RFC 2246 (Standards Track), January 1999.

[RFC2986] M. Nystrom and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7," IETF RFC 2314 (Informational), November 2000.

[RFC3820] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson, "Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile," IETF RFC 3280, June 2004.

[ProxyDelegation] V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder, and F. Siebenlist, "X.509 Proxy Certificates for Dynamic Delegation," 3rd Annual PKI R&D Workshop, 2004.

[MyProxy] Novotny, S. Tuecke, and V. Welch, “An Online Credential Repository for the Grid: MyProxy,” Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001, pages 104-111.