SSM Messaging Protocol

Introduction

The protocol used by the SSM was designed about 18 months ago, according to the limitations of the EGI broker network. In the light of feedback and changes to the network, we plan to reconsider the protocol used. This document describes the way that the SSM currently communicates.

The SSM uses STOMP as the underlying protocol. All messages sent by clients are sent to one topic (denoted <topic>), but those messages must have a temporary, unique 'reply-to' queue in the headers (denoted <tmp q>). Each time the server SSM receives a message from a client, it responds to that queue.

The protocol

Sender	**Headers**		**Message content**	**Notes**
	Key	**Value**		
Client	**'ssm-msg-type'**	**'certreq'**	None	These messages are used for the client to get the server certificate, which is required for encrypting messages. The messages are not compulsory.
	'destination'	<topic>		
	'reply-to'	<tmp q>		
Server	**'ssm-msg-type'**	**'certresp'**	Server X509 certificate in PEM format (cleartext)	
	'destination'	<tmp q>		
Client	**'ssm-msg-type'**	**'msg'**	`encrypt( sign( base64encode( compress( msg1 ) ) ) )`	The actual message, compressed, encrypted and signed. Optionally may include a reack to indicate receipt of an ack for a previous message
	'destination'	<topic>		
	'ssm-msg-id'	<unique id>		
	'ssm-reack'	md5(msg0)		
	'reply-to'	<tmp q>		
Server	**'ssm-msg-type'**	**'ack'**	None	The ack message. 'REJECTED' signifies some error with the message, which has not been saved by the server.
	'destination'	<tmp q>		
	'ssm-msg-id'	md5(msg1) OR 'REJECTED'		
Any	**'ssm-msg-type'**	**'ping'**	None	Optional. May include a reack header if sent from the client
	'ssm-reack'	md5(previous message)		

Encryption and signatures

- The `compress()` function is python's `zlib.compress()`. This is compatible with other zlib implementations.
- The `base64encode()` function is standard base64 encoding.
- The `sign()` function is equivalent to:
    - `openssl smime -sign –inkey <path-to-private-key> -signer <path-to-cert> -text`
- The `encrypt()` function is equivalent to:
    - `openssl smime -encrypt –des3 <path-to-server-cert>`

The server must reverse these four functions on receipt of a message.

- The `md5()` function is the standard MD5 hash.

Notes

- You may add a 'receipt' header to any message. This will cause the broker to reply with a RECEIPT frame to indicate that the broker has received the message
- If a client already has the server certificate, it does not need to send a 'certreq' message and wait for a 'certresp' message in response
- Once a message is sent to a topic, the client should wait for an ack message, indicating that the server has received the message, before sending a subsequent message.
- The purpose of the SSM_REACK header on a message sent by the client is to inform the server that an ack for an earlier message has been received. It is not strictly necessary for the client, but allows the server to clean up.