Accounting Messaging Protocol Improvements

This document references the previous document SSM Messaging Protocol.doc.

Introduction

The protocol used by the SSM for messaging is somewhat complex. It was designed in late 2010 with the limitations of the EGI broker network in mind, and in response to feedback and changes to the broker network we plan to simplify the protocol.

This document will discuss possible deficiencies and improvements one by one.

1. Message encryption; certificate request and response messages

Messages are encrypted before sending. For this reason, there are two separate message types for requesting and sending the server's certificate.

Reason

The accounting data contains personal information and must be encrypted in transit. The SSM built encryption into the transport mechanism, requiring the client to have the server's certificate so that it can encrypt the data. These messages are a way for the clients to get the server's certificate.

Proposed solutions

1. The broker network now supports SSL connections. If we can ensure that both client and server connect to the broker using SSL, additional encryption would not be required and these messages would not be necessary. We are already using SSL connections in our production system.
2. We could retain the encryption step. The certificate could be hosted on a web server and fetched separately by the clients that require it.

2. Signing messages, and the specification for this

The python implementation of the SSM uses openssl directly to sign and encrypt messages. The encryption scheme has not been well-defined.

Reason

We may be able to omit the encryption, but it is important for the APEL server to be able to trace the source of the messages it receives, and so we would still like the messages to be signed. We use openssl because the python encryption libraries are incomplete, and openssl is ubiquitous.

Proposed solution

Specify exactly the signature scheme, whether or not it is the existing openssl command.

3. The SSM implements synchronous messaging

When a client SSM sends a message, it waits for a reply from the server before sending the next message. This loses some of the benefits of the asynchronous nature of messaging.

As part of this, the server SSM listens to a topic, rather than a queue. If the server is not listening when a message is sent, the message is lost. Because the client does not receive an 'ack', it will re-send the message.

Reasons

1. APEL sends and receives a large quantity of data in a day – perhaps 2GB. If APEL were configured to use a persistent queue, we would rely on the brokers to withstand heavy load in the event of the APEL server going offline
2. If the broker network does not allow authentication per queue, if a message is sent to a queue and it is read by something other than the APEL server, the message's information would be lost.

Proposed solutions

1. Retain synchronous messaging. This has been tested and works reliably for a high load of messages, and places no extra load on the broker network.
2. Alternatively, investigate with the broker network whether they are happy to support a persistent queue with our message load.