Problem

How to establish secure communications?

Solution

This description addresses administrators.

What is it?

An Acs installation can be run under secure communication (via SSL). Secure Communication means

1. content sent from one party to the other party over the network is encrypted
2. a party has a guarantee that it is really talking to the party it wants to talk to

What features do we support?

We have investigated the procedure for Java Containers and the Manager. Both use JacORB, and the procedure boils down to configuring JacORB appropriately. You find below a work description for the following scenario:

- The Acs Manager and Containers shall allow secure and unsecure requests from client applications (in other words, manager and containers don't care who they would talk with)
- The client application shall insist on secure communication to Manager and Containers (in other words, the client wants to be sure who it talks to, and that nobody can eavesdrop the communication)

Note: from here on, we will talk of the **Server-side** when we mean the Acs Manager and Containers, and we will talk of the **Client-side** when we mean the client application (like ZLegacy/ACS.AcsCommandCenter, Alma ObservingTool, etc.).

Required preparations

The principle setup needed for SSL:

Every party needs

1. a private key
2. a public key
3. a key store

In the key store, every party stores

1. its private key
2. its public key
3. the public key of a party it wants to talk to securely

Server-side (Managers and Containers)

First, create a keystore and keys for the Manager and Containers. (You can figure out two passwords, or use the dummy passwords PASSWORD1 and PASSWORD2)

```
keytool -genkey -keyalg RSA -alias acs -validity 25000 \
        -keystore acsKeystore -storepass PASSWORD1 \
        -keypass PASSWORD2 \
        -dname "CN=Acs Manager/Container, O=Alma Acs"
```

Second, export the public key to an extra file

```
keytool -export -keystore acsKeystore -alias acs \
        -storepass PASSWORD1 -file acsCertificate
```

Client-side (Client Application)

First, create a keystore and keys for the Client Application. (You can figure out two passwords, or use the dummy passwords PASSWORD3 and PASSWORD4)

```
keytool -genkey -keyalg RSA -alias acsClient -validity 25000 \
        -keystore clientKeystore -storepass PASSWORD3 \
        -keypass PASSWORD4 \
        -dname "CN=Acs Client, O=Alma Acs"
```

Second, import the server-side's public key from the extra file

```
keytool -import -keystore clientKeystore -alias acs \
        -storepass PASSWORD3 -file acsCertificate
```

Starting Acs

Server-Side

As you know, when you run any one of

- acsStart
- acsStartManager
- acsStartContainer -java

it will create a new JacORB instance on the local machine to communicate with the world over CORBA.

The JacORB instances being created on the server-side (one for the Manager, one for each Container) need to be configured for SSL:

```
###### Jacorb SSL Configuration ######
### See the default .jacorb_properties and the
### jacorb programming guide for more information
###
jacorb.ssl.socket_factory=org.jacorb.security.ssl.sun_jsse.SSLSocketFactory
jacorb.ssl.server_socket_factory=org.jacorb.security.ssl.sun_jsse.SSLServerSocketFactory

jacorb.security.support_ssl=on
jacorb.security.ssl.client.supported_options=60
jacorb.security.ssl.client.required_options=0
jacorb.security.ssl.server.supported_options=60
jacorb.security.ssl.server.required_options=0

jacorb.security.keystore=/home/mschilli/seccom2006/acsKeystore  (...adjust to your system....)
jacorb.security.keystore_password=PASSWORD1
jacorb.security.default_user=acs
jacorb.security.default_password=PASSWORD2
jacorb.security.jsse.trustees_from_ks=off
```

To pass these config entries to JacORB, you have two options:

- either, you put them into the default *.jacorb_properties* (or *jacorb.properties*) file that is used anyhow
- or,
    - you put them into an extra file "/tmp/.jacorb_ssl_properties" (choose whatever path and name you like)
    - and do an export JAVA_OPTIONS="$JAVA_OPTIONS -Dcustom.props=/tmp/.jacorb_ssl_properties".
        - Note that e.g. *=$ACSROOT/.jacorb_ssl_properties* or *=~/.jacorb_ssl_properties* does not work: you cannot use variables or shell shortcuts here.

Client-Side

Where is the keystore?

The client needs to have its keystore available.

Configure JacORB

Also, you need to configure the JacORB again using one of the mechanisms described in the foregoing section. The proper configuration is this:

```
###### Jacorb SSL Configuration ######
### See the default .jacorb_properties and the
### jacorb programming guide for more information
###
jacorb.ssl.socket_factory=org.jacorb.security.ssl.sun_jsse.SSLSocketFactory
jacorb.ssl.server_socket_factory=org.jacorb.security.ssl.sun_jsse.SSLServerSocketFactory

jacorb.security.support_ssl=on
jacorb.security.ssl.client.supported_options=60
jacorb.security.ssl.client.required_options=20
jacorb.security.ssl.server.supported_options=60
jacorb.security.ssl.server.required_options=0

jacorb.security.keystore=/home/mschilli/seccom2006/clientKeystore  (...adjust to your system...)
jacorb.security.keystore_password=PASSWORD3
jacorb.security.default_user=acsClient
jacorb.security.default_password=PASSWORD4
jacorb.security.jsse.trustees_from_ks=on
```

Connect to Manager

When the client wants to connect to the Manager, there is one complication: the client needs the full IOR of the Manager.

This means, a simple *export MANAGER_REFERENCE=corbaloc::almahost:3000/Manager* will **not** work.

Instead you have to find out the IOR of the manager (it is printed on stdout when you run acsStart or acsStartManager) and then do *export MANAGER_REFERENCE=IOR:000000000000001E49444C3A636F73796C61622E<really long hex number>*.

Note to *Acs Command Center* users: You can paste the IOR into the *Host* field, leave the *Port* field empty, and then press "Add Manager".

Note to Java developers: When you develop a client in Java, you will probably use the ComponentClient class. To its constructor you can both pass a corbaloc or an IOR.

to be continued

In principle, jacorb can be configured to use a predefined port. Would allow to open exactly one port in a firewall. However, so far I always found jacorb opened two ports: the specified one and another somewhere in 30000 - 40000. May be control + data port, unclear what happens.

```
OASSLPort=3499
```

The following jacorb config properties somehow allow to use e.g. *corbaloc:ssliop:te22:3500/Manager*. But unclear how, didn't work for me, yet.

```
jacorb.security.ssl.corbaloc_ssliop.supported_options=0
jacorb.security.ssl.corbaloc_ssliop.required_options=0
```

-- MarcusSchilling - 08 Nov 2006

Related articles

- How can more people do development with ACS on the same machine without disturbing each other?
- Which ports are used by ACS?
- Problems connecting to ACS servers on a remote machine: bad /etc/hosts
- Why does the getComponent method of ZLegacy/ACS.ContainerServices return an object of type None?
- Why are some of my print statements not showing up in the container output section of acscommandcenter?