Java Email Server
(JES)

version 2.8.x

License Information

Java Runtime

JES 2.8.x compiles under Java SE 6, 7. The testing suite has been executed using the SE 6 JRE in windows 7.

Dependencies

Runtime

commons-codec 1.4
commons-dbcp 1.3 (or 1.4)
commons-pool 1.5.4
commons-logging 1.1.1
derby 10.10.1.1
derbyclient 10.10.1.1
derbynet 10.10.1.1
JES-DNS 1.0
JES-Crypto 1.0
unixUID 1.0

Test (in addition to the runtime libraries)

javax-mail 1.5.1
junit 4.11
log4j 1.2.17
pop3 1.5.1
smtp 1.5.1

All the aforementioned libraries are downloaded when the maven project compiles. There are no custom built versions of external source code. JES is however dependent on three subprojects, namely unixUID, JES-Crypto and JES-DNS. They are available in the sourceforge svn repository and the prepackaged release files. The corresponding jar executables have to be manually installed in the local maven repository.

Table of Contents

1. Important considerations

a. Do not attempt to run JES simply by invoking the jar as an argument for a java executable, unpleasant discomfort will be the only result. Certain arguments have to be passed to the JVM and JES itself. See sections "Java Service Wrapper", "Running JES using mail.bat in Windows / mail.sh in Unix-likes" for details regarding these arguments and/or the mail.bat / mail.sh code.

b. JES ships with the security manager off by default. You must change the relative entry in section Server Security Policy of mail.xml to enable it. You are strongly encouraged to read the present document in its entirety.

c. Starting with version 2.5 JES no longer supports Java versions prior to 6. If you are would rather use an earlier Java version please select one of the previous JES releases.

2. Installation

To install and run this mail server, carry out the following:

Install a Java JDK or JRE version 6 or greater by visiting:
http://www.oracle.com/technetwork/java/index.html or http://www.java.com)

Optionally install the JCE (Jave Cryptography Extension). Read the "Java Cryptographic Key Strength Policy" section for details.

2.1 Installing JES on a Windows PC

1. Expand the distribution archive to the directory of choice.

2. Modify the included script files for your local file system (if necessary).

3. Edit the mail.xml, user.conf (if using the File backend option), log4j.properties (optionally), wrapper.conf files as needed. See section entitled "Setting up JES for the first time".

4. Supply (if needed) the necessary passwords in the "password" file in the security directory. Delete the password file in the security directory and rename passwordWin to password, if you plan to use the supplied keystore without changing the supplied password (keeping this password is of course not recommended).

5. Select one of three (predefined) ways to run the server:
   a. as a standalone console application using mail.bat
   b. as a wrapped console application using jes.bat
   c. as a wrapped service by executing Installjes-NT.bat and starting it by:
      I. typing in the command console: net start Java_Email_Server,
      II. accessing the Services console, selecting Java Email Server and clicking on 'Start the Service',
      III. restarting the system.

Java Wrapper (if used) must be configured for proper JES operation. Read further on.
If you plan to run JES using mail.bat please read ahead.

2.2 Installing JES on a Linux[1] PC

Extract the contents of the bin release archive to a directory, preferably to

1 The example applies to a Ubuntu installation although it should work with any Debian release with trivial modifications. Furthermore, linux systems, as well as other UNIX/POSIX systems, assign (by default) the first 1024 ports exclusively to privileged users (the root in the case of linux). This has to be taken into account since the mail related protocols use (by default) these privileged (well-known) ports.

/usr/local/jes or /opt/jes.

If JES is to be executed at linux (ubuntu/Debian) system start (thus with root privileges, which are revoked by downgrading to a less privileged user) :

> Create a system user explicitly called jeserver with the following command using root privileges:
> *sudo adduser --system --ingroup <userGroup> jeserver*
> where <userGroup> should ideally be substituted by the group the administrator of JES belongs to. The wrapper start script has been altered to read the uid of the jeserver user from the system.
> Make sure everything in the install dir is owned by the jeserver user and admin group:
> *sudo chown -R jeserver:<userGroup> /usr/local/jes*
> Restrict the access to the installed files as desired. Make sure the jes.sh script in the bin dir is executable. Edit the jeserver script file in the bin install dir in order to set the JESPATH argument to the directory JES has been installed in.
> For example: *JESPATH=/usr/local/jes* or *JESPATH=/opt/jes*
> Move or copy the jerser script to /etc/init.d and make sure it is executable:
> *sudo chmod ug+x /etc/init.d/jeserver*
> Optionally prevent others from executing it:
> *sudo chmod o-x /etc/init.d/jeserver*
> In order to make JES start at boot time simply execute:
> *sudo update-rc.d /etc/init.d/jeserver defaults 40 60*
> The 40 60 figures are simply a recommendation.
> The jeserver script executes the jes.sh script. This two are to be used in conjuction under the assumptions laid out in this section. See section "Running JES using jes.sh in Unix likes" for more info.

In any other case:

> Make sure the mail.sh script in the bin dir is executable.
> Read section "Running JES using mail.bat in Windows / mail.sh in Unix likes" for changes needed before calling the script.
> Edit the mail.xml, user.conf (if using the File backend option), log4j.properties / jdk14.properties (optionally), wrapper.conf files as needed. See section entitled "Setting up JES for the first time".
> Have the needed passwords available in the "password" file in the security directory. Call a script as root with a start argument or restart the system.
> Java Wrapper (if used) must be configured for proper JES operation. See section 3.2.

Recommendation

Remember to read the rest of this document thoroughly when you have the time or should you encounter any issues before posting to the project's mailing list. Most issues are covered by the documentation.

3. <u>Additional Notes</u>

3.1 Setting up JES for the first time

No special changes are needed for the optional log4j.properties / jdk14.properties except enabling the SocketAppender / SocketHandler and setting the RemoteHost and Port to be used by CHAINSAW (or any application setup to listen for events on the specified network address/port). You can also define the default logging level (info, debug etc.) for any of the loggers. If you plan to use a different logging facility set it up accordingly.

3.1.1 Using a filesystem as a backend

The mail.xml configuration file needs very few changes for initial usage. One such is adding at least one local domain to "backend/File/domains" and optionally a default user to "backend/File/defaultMailbox" that MUST be a member of the domain defined in domains. If you 're satisfied with the default options there are no more changes needed to mail.xml. See entries in the configuration file for information concerning their usage.

Having added a local domain to mail.xml to be administered by JES, users (along with their password) belonging to the aforementioned domain are entered into file user.conf. At least one user should be registered with JES. See user.conf for details concerning the format of entries in that file.

3.1.2 Using a database as a backend

The users.conf and realms.conf as well as the entries “domains” and “defaultMailbox” in mail.xml are ignored. All entries pertaining to domains/users/realms are handled via the database. No domains/users/realms are needed to boot JES in this mode. All such entries are passed to the database through the ConnectionBasedConfigurator. Therefore you **must** enable it by setting the desired values for the “cbc/listenAddress” and “cbc/port” entries in mail.xml. Instructions on how to use the CBC can be found in section 4.

3.2 Java Service Wrapper

The Java Service Wrapper library is offered as a means of starting JES. In order for JES to execute properly a modification has to be applied to wrapper.conf. A value has to be defined for the following entry:

wrapper.app.parameter.2=

This option corresponds to the first argument of the JES main class which the mail server considers to be the JES install dir. It is strongly recommended that an absolute path is provided (e.g. c:\jes on windows or /usr/local/jes on linux). Additionally, for each jar that implements a specific external module, an entry MUST be present in wrapper.conf that matches the complete jar path. Such an entry can, for instance, be:

wrapper.java.classpath.3=../external/JES-Module1.jar

3.3 Running JES using jes.sh in Unix likes

This script is only to be used (without modifications) under the assumption that a system user is created as described in the unix installation process in subsection 2.2. Linux ports under 1024 are privileged therefore only the root may use them. After setting up the ports, JES switches to the non privileged jeserver user. Do not try to run this script without following the process dictated in subsection 2.2. Use mail.sh if you are not worried about system security. (Thanks Loulou)

3.4 Running JES using mail.bat in Windows / mail.sh in Unix likes

In order to properly execute JES from the command prompt using mail.bat/mail.sh the said batch file has to be modified. JAVA_EXEC need not be altered if the default installed JAVA runtime is acceptable to be used with JES. JES_HOME MUST be set to the directory JES was installed into. If you plan to use external modules with JES, the full file path must be appended to the classpath (-cp) entry.

To execute the script, no arguments are required at the command line. Nonetheless, a single script argument can be defined, to be interpreted either as the “testing” property or the uid to downgrade to. In the latter case, if JES is running on Windows, the property will be ignored. If however JES is running on Linux and can not downgrade (because the script was not executed as root) the execution will end abruptly.

3.5 JES-DNS

JES (starting with version 2.8) is using an in-house software library to resolve IP and dns names, in a transparent way to the application. No special handling is needed, nor any arguments are required to be entered in the start scripts.

Two fixed arguments are currently present in the execution scripts (mail.bat, mail.sh, wrapper.conf), namely:

dns.simple=true and
dns.mode=recursive.

These are required currently, to assure the most reliable use of JES-DNS. The library is at this time under development and in future releases, these arguments may no longer be necessary.

In recursive mode (which is not the default JES-DNS mode of operation, iterative is) JES-DNS auto-detects locally available DNS servers. If specific dns servers are desirable, or JES-DNS is unable to find DNS servers, a command separated list of IP addresses of name servers (or resolvers) can be supplied via the dns.servers argument.
For more information concerning JES-DNS and further configuration options, please consult the JES-DNS documentation.

3.6 Java Cryptographic Key Strength Policy

Java doesn't support the entire range of cryptographic key strengths out of the box due to US policy export restrictions imposed on the distributed java bundles. A separate download which can be retrieved from:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

called Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files exists, that allows Java to unleash full key strength support. Follow the instructions contained in the downloaded archive to install it. For the RTFM challenged, one simply needs to copy

local_policy.jar

US_export_policy.jar

from the downloaded archive to the <used-jre-install-path>\lib\security directory replacing the files of the same name.

IMPORTANT NOTICE: Should you decide not to install the JCE, JES2 will run just fine with two (2) security exceptions: The enabled TLS/SSL ciphers will be restricted to AES_128 and DES and JES vault will be encrypted with a 128bit AES key.

3.7 Developing / checking out JES under NetBeans / Eclipse

3.7.1 Using NetBeans IDE

Regardless of simply unpacking a source distribution of JES or checking out the sourceforge SVN repository, NetBeans IDE has everything pre-installed (even the basic Java SE download bundle) so no additional actions are required.

3.7.2 Using Eclipse IDE

A few plugins need to be installed in Eclipse in order to fully support JES. At first the subclipse plugin (http://subclipse.tigris.org/) must be installed in order to add SVN support to Eclipse. Maven support must then be added. m2e (http://eclipse.org/m2e/) is the recommended plugin. There is a known benign bug in m2e that manifests itself whenever executing a maven command through the IDE. To overcome bug 387064, follow the instructions in Comment #12. This appears to be the cleanest solution.

3.8 Build Instructions for version 2.8.x and higher

Building JES from source requires Maven (version 3.0.x is used and recommended).

JES can be built by invoking the *mvn package* or *mvn clean package* commands. Only unit testing takes place in this case. If the more elaborate simulation runs are desired, the command to invoke is *mvn integration-test.* The unit tests can also be run directly (by invoking *mvn test* at a command prompt/terminal).

The jars JES-DNS-1.0.jar, JES-DNS-1.0.jar and unix-UID-1.0.jar must be manually installed in the local repo. They are included in the release jars and the sourceforge svn repository.

There is the option of specifying the directory to where a testing installation of JES resides, so that a copy of the newly built jes.jar file can be automatically transfered to. All that is required is the presence of a properties file entitled copyTo.properties in the root directory of the JES project. A single entry pointing to the desired directory, with the key “test.dir” and the corresponding value, must be included.

3.9 Security policy entries that need manual editing

The cumbersome and error prone procedure of manually specifying policy.file entries has been dropped, in favor of an automated policy.file generation scheme.

3.10 Issues with (Free) E-mail Services

For some time now, hotmail, gmail, yahoo and most likely other mail servers (free or not) have been using spamhaus' PBL to reject unauthenticated mail from requests originating from servers running on dynamic IPs. Therefore, the reverseDNS strategy solution previously employed has been rendered ineffective. The most viable solution at this time is to force the use of a smarthost. The way to do this is by adding one or more hosts in the “mail/outgoingSMTPServer/server” element of mail.xml to point to your ISP's mail server or a commercially available smarthost.

3.11 Logging Facility

JES allows commons-logging to determine the logging system to be used. While log4j is included in the supporting libraries, it is only specified externally via wrapper.conf or mail.bat. No hard coded entry in the manifest file exists.
Two settings are used to achieve this in case log4j is used:

```
org.apache.commons.logging.Log=org.apache.commons.logging.impl.Log4JLogger
log4j.configuration=file:../conf/log4j.properties
```

The second should point to a valid URL (as specified in java).

Since the logging system's libraries have to be located in the classpath an entry has to be present in mail.bat:

```
-cp <other entries>;%JES_HOME%\lib\log4j-1.2.17.jar
```

and/or wrappers.conf:

```
wrapper.java.classpath.4=../lib/log4j-1.2.17.jar
```

By default, JES uses log4j out of the box with no user intervention, by means of the aforementioned settings. If a diferrent version and/or file location of log4j is to be used, the log4j.configuration setting and the classpath entry have to be altered accordingly.

Starting with JES 2.5.x the native Java logging facility, commonly known as the jdk14 logger, can be used almost out of the box. A few modifications to the starting scripts is all that is needed. Note, that in mail.bat and mail.sh they are already in place. All that is required is to comment out the log4j execution entry and uncomment the jdk14 entry.
Regardless, the script entries are listed next.

org.apache.commons.logging.Log=org.apache.commons.logging.impl.Jdk14Logger
java.util.logging.config.class=com.ericdaugherty.mail.server.logging.jdk14.LoggingConfigLoader
java.util.logging.config.file=$JES_HOME/conf/jdk14.properties or *%JES_HOME%/conf/jdk14.properties*
jes.install.directory=$JES_HOME or *%JES_HOME%*

Should a different logging suit version or different settings for any logging medium be desired, commons-logging and the logging facility of choice must be explicitly configured.

Of special note is the fact that for its own testing purposes, JES explicitly uses log4j, so a dependency is present in the pom but is restricted to the test scope.

3.12 kerberos 5 Principals

In order to use one or more kerberos 5 principals one needs not only define the relative settings in mail.xml but also edit a few entries in the security policy file. Default entries are already included in the policy file that have to be altered.
Specifically:
Have the entry permission <<java.net.SocketPermission "*.example.com:88">> point to a specific host that the kdc is running on, setting a port other than 88 if necessary.
For each principal two entries are required:
<<permission javax.security.auth.AuthPermission
"createLoginContext.com.ericdaughery.mail.server.auth.GSSServerMode<protocol>">>
<<permission javax.security.auth.kerberos.ServicePermission
"<protocol>/host.example.com@EXAMPLE.COM", "accept">>
Substitute <protocol> with the name of the protocol for the principal in question. Replace host.example.com with the host name of the server the service is running on. Rename EXAMPLE.COM to the realm name of choice using upper case.

3.13 Migration Tool

To facilitate an expedited transition to version 2 a SMTP message and password conversion utility is included in package com.ericdaugherty.mail.server.utils. To use it simply type at a command prompt:
java -cp <path to>/jes.jar com.ericdaugherty.mail.server.utils.Migrate arg1 arg2
where:

arg1 is the directory where JES 1.6.1 resides.
arg2 is a temporary directory where the converted smtp messages and the user.conf file are to be persisted. The migration utility will create this directory if it does not yet exist. If you plan to use a db as the backend (version 2.1.x and over) the user passwords need to be respecified since the password generation process is different to that of version 2.0.x

and earlier.

3.14 Testing the project using Maven

The testing facility has both a unit and a component integration testing aspect. The unit tests are executed during the test lifecycle maven phase. The integration tests (simulation runs) take place after JES has been packaged for distribution and only executed by invoking *mvn integration-test*[2]. The simulation runs can be configured using the tests.xml file located in the src/test/resource directory.

The new simulation testing facility expands on its predecessor by allowing a dynamic and easily configurable testing session. Specifically, an XML formatted file entitled tests.xml is now located in the src/test/resources, that allows setting among other things, whether to check the concurrency capabilities or not. Tests are defined with a name attribute. One or more tests can be skipped by declaring their name attribute in the skip element (using either whitespace, commas or dots as delimiters), instead of removing the relative node(s) from the xml tree. The number of messages to be sent by each user, as well as a per user repetition count can be set. The DIGEST-MD5 related REALMS entry in the javaMail element allows for two constant values or a specific realm name. The two constants are “none” which translates to using the *server* element value as the realm and “default” which defaults to the *realms* sub-element value of the *defaults* element to which the server value is appended.

The sample tests.xml file included in the appendix corresponds to a large degree to the 10 tests that were executed by previous versions of JES. The test cases to be executed by invoking “mvn integration-test” are setting up temporary instances of the server by creating installation folders in the system defined temporary folder whose location is retrieved through the system property “java.io.tmpdir”. When all the cases complete successfully the folders should be deleted. However, if at least one test is defined with a db as the backend, the derby instance never really shuts down when the server.shutdown() method is invoked. Only when the enveloping JVM shuts down does derby shutdown and release all the allocated resources. The implication is two fold. Any tests executed after the initial one (the first to use a db as the backend) actually use the derby instance created by this test. Moreover, file handles are not released, so when the simulation run finishes a single folder containing the derby instance remains in the system temp folder. This folder should be manually deleted.

Note: a number of folders may also remain in case of a test failure.

New with version 2.8: a new Db test run (11) to verify CBC commands reception and execution. Also, two (skipped by default) tests (namely *db_memory_profiler* and *file_memory_profiler*) meant to test for memory leaks have been added.

2 Or any phase later in the maven build lifecycle, e.g. *install.*

3.15 Using a recipient policy to handle message recipients

Under certain server usage scenarios, it may be desirable to restrict the domains allowed as message recipients either globally, or for a number of locally administered domains. Towards this end a recipient policy file has been placed in the configuration directory. Consult this file's header for instructions on how to setup the policy.

3.16 JSSE keystore password and/or kerberos 5 principal passwords

In order to use a SSL/TLS certificate either to support the STARTTLS SMTP/POP3 extension or to secure the CBC connection the certificate has to be retrieved from a keystore. The keystore password has to be supplied to JES. The means to do so is by using a file named “password”[3] created in the security folder. The file must be populated with a line that has the key/value pair “keystore=password”. If the password proves incorrect then a popup will be displayed requesting it; or a runtime exception will be thrown if the jvm is gui-less.

Kerberos 5 passwords are obtained in a similar fashion. The password file can again be used. An entry “serviceName=servicePassword” is required for the (each) principal. If use of a keytab is selected and the keytab location is set then the principal key is retrieved from it. If key retrieval fails then the password file is looked up. Should that fail the password is requested using a popup.

3.17 JES Vault and the master password

Starting with release 2.5 JES switches to an scrypt based system to persist all security sensitive information (keystore password, kerberos 5 principal(s) password(s), the db backend credentials etc).

The system generated master password has been replaced by a user supplied one via either the password file or a popup. It is no longer persisted by default and has to be supplied to the system at every startup. The password file entry uses the following format:
password=

The password is therefore the key and there is no value.

There is an important note concerning the ability of JES to restart. After being requested to alter at least one of its non dynamically updated settings, JES shutdowns and restarts. During the restart process, and in order not to nullify the option of remotely administering JES, access to the master password is offered without user intervention.

3 The password file must be encoded in UTF-8, should any character be outside the US-ASCII range. If a Unicode BOM (LE, BE, UTF-8) is present, it is parsed and the encoding is set accordingly.

3.17.1 Updating the master password

A new master password can only be supplied by using the password file. Both the old and new password need to be entered and have this format:
password=
new.password=

The new password entry has a leading “new.” part. The old password is needed for verification.

3.17.2 Master password strength

In order to maximize the entropy of the supplied master password a few restrictions are enforced. These are (in no particular order):

– A minimum of 10 letters/numbers/symbols
– No ASCII illegal (non printable) characters
– At most 2 numbers back to back
– At most 3 (same case) letters back to back
– At most 2 (left to right/right to left) QWERTY (US layout) keyboard letters/numbers/symbols back to back
– At most 2 repeating digits or neighboring ones (the difference of their unicode codepoint must have an absolute value larger than 1).
– At least 2 upper case ASCII letters on (case-indifferent) non-ASCII letters / numbers/symbols

No particular order is required, but avoiding symbols/numbers at the beginning or/and ending is encouraged. When a popup prompts for the master password a real time strength checker is provided.

3.18 Updating the bound SMTP/POP3 ports on a running JES instance

The SMTP/POP3 ports can be changed while JES is running, without restarting it. This operation is however OS dependent. On Win there is no concept of privileged ports, therefore any port in the standard range (1-65535) can be selected. On Linux however, the lower 1024 ports are deemed privileged and are selectable only when JES is executed by the root user that was not subsequently downgraded. When this condition is not held and a privileged port is indeed selected, a random port number substitutes the port in question. Further checks are also carried out with the intent of avoiding port conflict (both with other JES listening processes and non-JES ones).

4. The Configuration Manager BackEnd

4.1 Introduction

Starting with version 2.1.x JES provides the option of persisting the domains, users and digest-MD5 realms either to the filesystem (as was up to version 2.0.x) or an internally instantiated database. From version 2.8.x onwards, certain arguments must be passed in as JSON objects/arrays.

Once the server has been run for the first time using either of the two options there is but one way to switch to the other; manually insert the domain(s)/user(s)/realm(s) to the other backend.

The choice has to be weighted based on two variables: scalability and memory management. The filesystem backend loads all the domains, users and realms at startup in the memory. For a small number of entries it is the ideal choice. But should the user base grow substantially, scalability and memory issues will most definitely arise. A database backend would be the proper choice in such an occasion.

If memory is restricted on the system one should consider the fact that using the database backend increases the memory footprint by at least 20 MiB.

4.2 Apache derby

At the time of this writing apache derby has been selected as the database of choice. Whilst the embedded flavour would seem the preferable choice, it presented it self with a dilemma to consider. There is no obvious manner by which the database can be accessed outside the instance JVM. Therefore the network version of derby has been selected.

Only a single user has full access to the database. This user is also the database creator. There can be only one more user defined and this user's access is restricted to read-only.

4.2.1 Database hosting

There are only three attributes required to be defined in subsection *Db* of section *backend* in mail.xml so that derby is setup and running (barring the *secure* attribute specified in section backend that forces channel encryption -see the next subsection- and the user credentials supplied via the password file-section 4.5). See mail.xml (or mail.xsd) for a description of these attributes.

4.2.2 TLS secured connection

By default the database is allowed to be accessed from a loopback, sitelocal or the zero-based address without any security measures. Unrestricted remote access is granted **only** by rendering the database access secure. In this case the comm channel gets TLS encrypted.

Should the CBC be selected to be secure, the database access is forced to be secure as well. The security scheme requires that the client also authenticates itself.

4.3 ConnectionBasedConfigurator

The augmented functionality of the CBC allows anyone accessing it to fully administer the domains, users and digest-MD5 realms. See the appendix for a full list of the commands.

4.3.1 TLS secured connection

By default the CBC is allowed to be accessed from either the localhost or the site local address of the host without any security measures. Unrestricted remote access is granted **only** by rendering the CBC secure. In this case the comm channel gets TLS encrypted.

Should database access be selected to be secure, access to the CBC is forcibly secure as well. The security scheme requires that the client also authenticates itself. There is no setting to circumvent this process. No credentials are necessary to connect to the CBC nor is there a check on a subjectAltName. Rather, the JES administrator must have a CA certificate in the JES truststore that will be used to validate the client's certificate. Since any valid certificate (that is a client certificate issued by a CA whose trusted certificate is located in the JES truststore) is granted access to the CBC and by extension to the database itself, the JES administrator is tasked with preventing unauthenticated access. The best way to go about this is to setup a private CA then have it issue and sign the JES server and the connecting client's certificates. The openssl BSD licensed SSL suite is ideal for this task. A number of scripts to automate this process (issue a self-signed CA certificate, followed by the issuing and signing of a server and client certificates) are located in the openssl folder under the JES install dir. These scripts have been tested with ubuntu's openssl 0.98+ version. These can be easily adapted to run under any linux distro. They can also serve as a guide to using openssl's windows port.

4.4 Security updates in the source

In order to prevent unauthorized access to the CBC, a security access check is performed to an incoming connection to the CBC listen address that verifies that the connecting address matches the address on which the CBC is listening to. Should the address differ a security exception is thrown. In addition, the restriction that the configuration listening address can only be the localhost or a site-local address the jes host can bind to (no domain name is allowed) means that only the host computer can access the CBC. This stringent approach obviously blocks (or at least tries to block) any remote access to the CBC for the obvious reasons. This approach is also justified by the fact that the CBC at this time doesn't require user credentials in order to enable access to the database.

4.5 User access to derby

Only two users are specified with access to the db. The one, which is also the db creator, is granted full access to the db. The other, is only allowed read access to the db and as the entry in mail.xml implies is to be used by a gui. Credentials are validated using a custom derby authenticator. These are supplied only via the *<jes-install path>/security/password* file. The entries are:
backend.db.username=*password* and gui.db.username=*password*.

4.5.1 Updating the db user credentials

The db creator plays a unique role in derby thus the specified username cannot be altered. The password however can be changed. Both the old and new credentials need to be supplied and the way to do it is to supply the old credentials in the usual manner (backend.db.username=password) and the new ones like so:
new.backend.db.username=*newpassword*

Note the leading "new." part in the key. The username has to be the same.

The gui user can be altered entirely. The two entries are:
gui.db.username=*password*
new. gui.db.(new)username=*newpassword*

Again the key part for the (possibly) new user has a leading "new." part. The username need not be the same (unlike the db creator case).

4.6 Settings not available via the CBC

A number of settings can not be retrieved or modified using the CBC facility, mostly on security grounds. These are the settings for the CBC itself, the javaWrapper section, as well as the allowRemoteRestart and legacyFileIOMode attributes in subsection security under the general section. Finally, no means of specifying the backend type exists, other than directly modifying mail.xml.

5. Final notes

Extensive documentation is provided within the configuration files. Try to gain as much insight into the inner workings of JES by carefully reading them. Don't be afraid to ask questions by posting to the user mailing list (a subscription is required). Albeit it should be a last resort.

Appendix

I. CBC Commands

Ia. CBC Commands in Db mode

The CBC does not follow the command-reply approach as used for instance by the smtp protocol. Rather, the connecting client issues a command followed optionally by the items that are to be inserted / updated / deleted / retrieved and finally a required period. Each line of command is terminated with CRLF. There are no restrictions on line length. The are two general syntaxes.

The first one concerns entries in the database (domains, users, realms):

```
command<CRLF>
item1:entry1,entry2,entry3<CRLF>
item2:entry4,entry5<CRLF>
itemn:entrym<CRLF>
.<CRLF>
```

The second one deals with the JES configuration:

```
apply-or-retrieve configuration command<CRLF>
configuration-section subcommand<CRLF>
?[new-configuration-settings<CRLF>]
.<CRLF>
```

The CBC submits the request for process. If the request requires a reply that contains data to be processed by the submitter of the command that data is appended first as a JSON object in a single line. Otherwise a control message is appended, indicating either success or failure. Finally, a single period is transmitted and CBC drops the connection.

The special notational items *, ? and | (colon) are translated to: zero or more items, one or no items and exactly one of the items respectively.

A few commands allow multiple executions, either as single entries or pairs. The special +**[...]** notation is used in the "Required entry is" entry to indicate that such an entry is allowed to appear more than once in a command. *Adding a domain* (*a*) is an example of multiple executions for a given command and *Removing Forward Addresses* (*h*) is a paired example.

The full command list is as follows:

a. Adding a domain

The command is ***insertDomain***

Required entry is +**[** ***domain:*** JSONArray of [*domain_name*] **]**

Example:

```
insertDomain
domain:["mydomain1", "mydomain2", "mydomain3"]
domain:["mydomain4", "mydomain5", "mydomain6"]
.
```

b. Deleting a domain

The command is ***deleteDomain***

Required entry is +**[** ***domainId:*** JSONArray of [*domain_id*] **]**

Example:

```
deleteDomain
domainId:[1,2,3]
domainId:[4,5,6,7]
.
```

c. Setting the default domain

The command is ***setDefaultDomain***

Required entry is ***domainId:****domain_id*

Example:

```
setDefaultDomain
domainId:1
.
```

d. Adding a user

The command is ***insertUser***

Required entries are +**[** JSONObject of {
username:*user_name*
password:*password*
domainId:*domain_id*

Optional entry is ***realm:***JSONArray of [*realm*]} **]**

Example 1 (no realms specified):

```
insertUser
{“username”:”user1”, “password”:”pass1”, “domainId”:1}
{“username”:”user2”, “password”:”pass2”, “domainId”:2}
.
```

Example 2:

```
insertUser
{“username”:”user1”, “password”:”pass1”, “domainId”:1, “realm”:[“realm1”,
            “realm2”]}
.
```

e. Deleting a user

The command is ***deleteUser***

Required entry is +**[** ***userId:***JSONArray of [*user_id*] **]**

Example:

```
deleteUser
userId:[1,123,45]
userId:[25,38]
.
```

f. Setting the user password

The command is ***setUserPassword***

Required entries are +[***userId:***JSONArray of [*user_id*]
password:JSONArray of [*password*] **]**
.

Example:

setUserPassword
userId:[123,45,64]
password:[userpass5,userpass7, userpass4]
userId:[25,38]
password:[userpass3,userpass9]
.

g. Adding forward addresses

The command is ***addForwardAddress***

Required entries are +[***userId:***JSONArray of [*user_id*]
forwardAddress:JSONArray of [JSONArray of [*address*]] **]**

Example:

addForwardAddress
userId:[1,123,45]
forwardAddress:[[“address1”,“address6”],[”address9”],[”address22”]]
.

h. Removing forward addresses

The command is ***removeForwardAddress***

Required entries are +[***userId:***JSONArray of [*user_id*]
forwardAddressId:JSONArray of [*address_id*] **]**

Example:

removeForwardAddress
userId:[1,123,45]
forwardAddressId:[[address1_id,address6_id],[address9_id],[address22_id]]
userId:[64]
forwardAddressId:[[address34_id]]
.

i. Setting the default mailbox

The command is ***setDefaultMailBox***

Required entries are +[***domainId:***JSONArray of [domain_id]
userId:JSONArrayof [*user_id*] **]**

Example:

setDefaultMailBox
domainId:[15]
userId:[95]
.

j. Inserting realms

The command is **insertRealm**

Required entries are +**[** ***domainId:*** JSONArray of [*domain_id*]
realm:JSONArray of [JSONArray of [realm]] **]**

Example:

```
insertRealm
domainId:[15,24]
realm:[[“realm2”,”realm16”],”realm11”]
.
```

k. Deleting realms

The command is ***deleteRealm***

Required entry is +**[** ***realmId:***JSONArray of [realmId] **]**

Example:

```
deleteRealm
realmId:[15,24]
.
```

l. Adding users to realms

The command is ***addUserToRealm***

Required entries are +**[** JSONObject of {
username:*user_name*
userId:*user_id*
domainId:*domain_id*
password:*password*
realm:JSONArray of [*realm*]} **]**

Example:

```
addUserToRealm
{“username”:”user11”, “userId”:78, “domainId”:33, “password”:”pass5”,
            “realm”:[“realm2”,“realm16”,”realm45”]}
{“username”:”user12”, “userId”:79, “domainId”:34, “password”:”pass6”,
            “realm”:[“realm3”,“realm17”,”realm46”]}
.
```

m. Removing users from realms

The command is ***removeUserFromRealm***

Required entries are +**[*****userId:***JSONArray of [*user_id*]
realmId:JSONArray of [JSONArray of [*realm_Id*]] **]**

Example:

```
removeUserFromRealm
userId:[78,56]
realmId:[[23,45],[16,45]]
.
```

n. Retrieve configuration section settings

The command is ***retrieveConfig***

Required entry is ***(ConfigGeneral | ConfigBackend | ConfigMail | ConfigDirectories | ConfigAmavis-dnew | ConfigOther)***

Example:

```
retrieveConfig
ConfigBackend
.
```

o. Apply configuration section settings

The command is ***applyConfig***

Required entries are ***(ConfigGeneral | ConfigBackend | ConfigMail | ConfigDirectories | ConfigAmavis-dnew | ConfigOther)***
<JSON Object>

Example:

```
applyConfig
ConfigBackend
<JSON Object>
.
```

Ib. CBC Commands in File mode

Unlike in CBC mode, where a full set of control commands exists, there is only the option of adding a single user with an optional realm. That command is:

a**. Adding a user**

Note:*The username is required to contain the domain*

The command is ***add user***

Required entries are +**[** JSONObject of
{***username:****user_name@domain_name*
password:*password*

Optional entry is ***realm:****realm}* **]**

Example:

```
insertUser
{“username”:”user1@domain1”,”password”:”pass1”,”realm”:”realm1”}
{“username”:”user2@domain2”,”password”:”pass2”}
.
```

II. Password file entries and their respective (key/value) format

1. Master password: password=
(the user's password is actually used as a key and the value is empty)

2. Kerberos 5 principal: serviceName=servicePassword

3. JSSE key keystore: keystore=password
(the “keystore” word is used as a key and the password is the value)

4. backend.db.entries: backend.db.username=password
gui.db.username=password

III. Sample tests.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<tests>
  <skip>1</skip>
  <defaults>
    <settings>
      mail/SMTP;port=17025

mail/SMTP/authentication/IPOverride:127.0.0.1

mail/SMTP/authentication/POPBeforeSMTP;
enable=true
      mail/SMTP/extensions;HELO=false
      mail/POP3;port=17110
    </settings>
    <realms>
      users
    </realms>
  </defaults>
  <test name="1">
    <messagesPerUser>2</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <javaMail>
      SASL=DIGEST-MD5
      REALM=none
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
      mail/authMechs/DIGEST-MD5;enable=true
    </settings>
  </test>
  <test name="2">
    <messagesPerUser>1</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <javaMail>
      SASL=DIGEST-MD5
      REALM=default
      STARTTLS=true
      PROTOCOL=TLSv1
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
      mail/SMTP/delivery;interval=1
      mail/SMTP/secureChannel;enable=true
      mail/SMTP/authentication;
allowClearText=encryptedOnly
      mail/POP3/secureChannel;enable=true
      mail/POP3/authentication;
allowClearText=encryptedOnly
      mail/authMechs/DIGEST-MD5;enable=true
    </settings>
  </test>
  <test name="3">
    <messagesPerUser>1</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <javaMail>
      SASL=PLAIN
      STARTTLS=true
      PROTOCOL=TLSv1
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
      mail/SMTP/secureChannel;enable=true
      mail/SMTP/extensions;pipelining=true
      mail/POP3/secureChannel;enable=true
    </settings>
  </test>
  <test name="4">
    <messagesPerUser>6</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <multithreaded/>
    <javaMail>
      SASL=PLAIN
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
      mail/SMTP/delivery;interval=1
    </settings>
  </test>
  <test name="5">
    <messagesPerUser>6</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <multithreaded/>
    <javaMail>
      SASL=PLAIN
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
```

```
      mail/SMTP/delivery;interval=1
    </settings>
  </test>
  <test name="6">
    <messagesPerUser>6</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <multithreaded/>
    <javaMail>
      SASL=DIGEST-MD5
      REALM=default
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
      mail/SMTP/delivery;interval=1
      mail/authMechs/DIGEST-MD5;enable=true
    </settings>
  </test>
  <test name="7">
    <messagesPerUser>6</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <multithreaded/>
    <javaMail>
      SASL=PLAIN
      STARTTLS=true
      PROTOCOL=TLSv1
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/File/domains:localhost

backend/File/defaultMailbox:andreas@localhost
      mail/SMTP/delivery;interval=1
      mail/SMTP/secureChannel;enable=true
      mail/POP3/secureChannel;enable=true
    </settings>
  </test>
  <test name="8">
    <messagesPerUser>8</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <javaMail>
      SASL=PLAIN
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/Db;host=localhost;port=17527
      mail/SMTP/delivery;interval=1
      mail/SMTP/secureChannel;enable=true
      mail/POP3/secureChannel;enable=true
      cbc;enable=true;port=41002
    </settings>
  </test>
  <test name="9">
    <messagesPerUser>6</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <multithreaded/>
    <javaMail>
      SASL=PLAIN
      STARTTLS=true
      PROTOCOL=TLSv1
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/Db;host=localhost;port=17527
      mail/SMTP/delivery;interval=1
      mail/SMTP/secureChannel;enable=true
      mail/POP3/secureChannel;enable=true
      cbc;enable=true;port=41002
    </settings>
  </test>
  <test name="10">
    <messagesPerUser>6</messagesPerUser>
    <runsPerUser>1</runsPerUser>
    <multithreaded/>
    <javaMail>
      SASL=PLAIN
      STARTTLS=true
      PROTOCOL=TLSv1
    </javaMail>
    <server>localhost</server>
    <settings>
      backend/Db;host=localhost;port=17527
      mail/SMTP/delivery;interval=1
      mail/SMTP/secureChannel;enable=true
      mail/SMTP/authentication;
allowClearText=encryptedOnly
      mail/POP3/secureChannel;enable=true
      mail/POP3/authentication;
allowClearText=encryptedOnly
      mail/threads;number=8
      cbc;enable=true;port=41002
    </settings>
  </test>
</tests>
```

IV. Supported Authentication Mechanisms (per operation mode)

JES supports a variety of authentication mechanisms. The complete list follows:

Authentication Mechanism	SMTP Server	SMTP Client**	POP3 Server	TLS/SSL***
SASL PLAIN	√	√	√	Required by RFC, user defined
LOGIN	√	√	√	user defined
SASL CRAM-*	√	√ (MD5 only)	√	optional
SASL DIGEST-MD5	√	√	√	optional
SASL SCRAM-SHA-*	√	×	√	optional
SASL GSSAPI	√	×	√	optional

* While there is no RFC that specifies a CRAM mechanism other than CRAM-MD5, it is really straightforward to use other MACs. These are: SHA-1, SHA-256, SHA-384, SHA-512. In the SCRAM-SHA-* case, RFC 5802 allows for other MACs besides SHA-1. These are: SHA-256, SHA-384, SHA-512.
** SMTP client mode is implemented as SMTPRemoteSender. The authentication mechanisms are meant to be used when the servicing ISP requires all outgoing mail to be routed through its own mail server and user authentication is mandatory.
*** Plaintext passwords are expressly forbidden by current RFC nomenclature unless a TLS/SSL security layer is first established. Nonetheless, there is an option (labeled allowClearText) in mail.xml that allows their use without a security layer.

IV. JES execution arguments

JES is launched by invoking the static method instantiate(String[] args) from a running jvm instance or from the command line in typical fashion (e.g. java … com.ericdaugherty.mail.server.Mail <space separated argument list>). There are a number of arguments that should/can be passed to the application:

Argument 1: JES installation directory (required, absolute path)
Argument 2-3: The word “testing” or a number (where 0 <= x) that corresponds to the desired uid. The position of these two arguments is interchangeable.