GCCF/AccessKey – Integrating a New Application

|  |  |  |  |
| --- | --- | --- | --- |
| **--**  **-- @X:\All Key Documents\AccessKey\Documents\AccessKey-Integrate\_new\_application\_v6.docx**  **--**  **-- AUTHOR: Brian Matchim**  **--**  **-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **-- Ver Date Description Initials**  **-- ===========================================================================================** | | | |
| **Ver** | **Date** | **Description** | **Initials** |
| -- 0.6 | 2012-06-03 | Draft document | BM |
| -- 0.7 | 2012-06-04 | Separation of Tomcat integration from the AccessKey integration | BM |
| -- 0.8 | 2012-06-06 | Added al section outlining the removal of a runtime configuration | BM |
| -- 1.0 | 2012-06-08 | Added comments from Ehsan re: tomcat (SSC) IP binding | BM |
| -- 1.01 | 2014-04-22 | Modified tomcat port to 8080 from 8082 | BM |
| -- 1.02 | 2017-03-22 | Added hyperlink highlighting; removed CGI provisioning; minor edits | JH |

# Create a New Apache Daemon with GetAccess Integration

1. Log on to the AccessKey/GCCF webserver for the environment to be edited:
   1. Core-dev
   2. Core-test
   3. Hcswpak-web
2. Go to the directory containing the http daemons and create a new application directory (as root)

# sudo su -

# cd /usr/local/apps **OR** # cd /opt/apache2.2.9/apps/

# mkdir <APP\_DIR>

1. Copy the template files to the new directory

# cp –pR TEMPLATES/\* <APP\_DIR>

1. Modify the following files as required
   1. vi <APP\_DIR>/conf/httpd.conf
      1. Update the ServerRoot to reflect the new directory

ServerRoot "/usr/local/apps/<APP\_DIR>"

**OR**

ServerRoot "/opt/apache2.2.9/apps/<APP\_DIR>"

* + 1. Update the Listen directive with the correct IP (and port if necessary)

Listen 172.30.x.x:443

* + 1. Update the ServerName line with the appropriate DNS entry

ServerName <appname><-env-><nomdapp>.hc-sc.gc.ca:443

Ex: ServerName myApp-*dev*-monApp.hc-sc.gc.ca:443

ServerName myApp-*test*-monApp.hc-sc.gc.ca:443

* + 1. Update the DocumentRoot line with the appropriate directory

DocumentRoot "/usr/local/apps/<APP\_DIR>/htdocs"

**OR**

DocumentRoot "/ opt/apache2.2.9/apps/<APP\_DIR>/htdocs"

* + 1. Update the Directory line with the appropriate directory

<Directory "/usr/local/apps/<APP\_DIR>/htdocs">

**OR**

<Directory "/opt/apache2.2.9/apps/<APP\_DIR>/htdocs">

* + 1. ~~Update the cgi directory root~~

~~<Directory "/usr/local/apps/<APP\_DIR>/cgi-bin">~~

**~~OR~~**

~~<Directory "/opt/apache2.2.9/apps/<APP\_DIR>/cgi-bin">~~

* + 1. This document assumes this to be a standard AccessKey application that uses both SSL (or TLS) and mod\_jk.

The Include statement for mod\_jk is commented and will be dealt with later in this document. However, if you aren’t using these technologies ensure that each line has a hashmark (#) at the beginning of either or both of these lines in httpd.conf. This will exclude these objects from loading into the webserver as the daemon starts.

If NOT using mod\_jk or SSL, the final result should be:

* + - 1. #Include conf/mod\_jk.conf
      2. #Include conf/extra/httpd-ssl.conf
    1. Complete any other custom Apache configuration as required by the application (see section 4.2 of the document at sftp://sadateftp@utildev01/srv/ftp/sadateftp/CANLINE/Installation%20Instructions\_CANLINE\_V4.0.0\_External.doc )

Save the file and exit vi.

* 1. vi <APP\_DIR>/conf/extra/httpd-ssl.conf and update <APP\_DIR> as necessary
     1. SSLSessionCache "shmcb:/usr/local/apps/<APP\_DIR>/logs/ssl\_scache(512000)"
     2. SSLMutex "file:/usr/local/apps/<APP\_DIR>/logs/ssl\_mutex"
     3. Update this line with the application IP

<VirtualHost 172.30.x.x:443>

* + 1. Ignore the DocumentRoot statement at this time
    2. Update the ServerName entry to match the DNS

ServerName <appname><-env-><nomdapp>.hc-sc.gc.ca:443

* + 1. ErrorLog "/usr/local/apps/<APP\_DIR>/logs/error\_log"
    2. TransferLog "/usr/local/apps/<APP\_DIR>/logs/access\_log"
    3. Generally rewrite rules aren’t required for html pages.

However, if you require a rewrite rule but aren’t using java and therefore not following the java integration steps below, you can jump ahead to section 12)e.iii. to use the RewriteRule and then return to ix below.

* + 1. SSLCertificateFile "/usr/local/apps/<APP\_DIR>/conf/ssl.crt"
    2. SSLCertificateKeyFile "/usr/local/apps/<APP\_DIR>/conf/ssl.key"
    3. In some cases an application may require a certificate chain file. If so, uncomment this line and update the location.

#SSLCertificateChainFile "/usr/local/apps/<APP\_DIR>/conf/server-ca.crt"

**NOTE:** Steps viii, ix and xi should be replaced by the installation of actual certificates in the Test and Prod environments

* + 1. <Directory "/usr/local/apps/<APP\_DIR>/cgi-bin">
    2. CustomLog "/usr/local/apps/<APP\_DIR>/logs/ssl\_request\_log" \

Save the file and exit vi.

* 1. vi /usr/local/httpd-2.2.9/conf/extra/httpd-ssl.conf as necessary
     1. Locate the template text at the bottom of the file. (Included here for reference)

##############

#

# TEMPLATE entry for adding new applications as daemons. Do NOT modify.

# ---------------------------------------------------------------------

#

# To use the template, copy and paste the eight lines below as a whole.

#

# Remove three hashmarks outlined in the integration guide, or

# remove one hashmark from each line. Lines with two hashmarks are meant

# to remain commented after this editing.

#

# Modify the Virtual host IP to be the IP assigned to your application.

#

# **The virtual host definition MUST be added before running**

# **the ConfigureVS.sh command.**

#

##############

#<VirtualHost 172.30.x.x:443>

## General setup for the virtual host

## DocumentRoot "/usr/local/apps/<APP\_DIR>/htdocs"

# ServerName <appname>-dev-<nomdapp>.hc-sc.gc.ca:443

## ServerAdmin you@example.com

## GAVirtualServerID 172.30.x.x\_443\_esd-dev-sed.hc-sc.gc.ca\_7160

#</VirtualHost>

* + 1. Select and copy the last eight lines of the template text (which is the <VirtualHost/> tag) and paste it above the template text.
    2. Follow the directions from the template and edit each of the newly pasted line as necessary. The result will be the 3 blue hash marks shown above will be removed and the appropriate IP entered for the daemon as a property in the VirtualHost opening tag.

1. Test the webserver configured as basic https webserver
   1. Modify the file /usr/local/apps/<APP\_DIR>/htdocs/index.html

‘vi’ the file and place any custom text in the template and save the file.

* 1. Start the webserver

# cd /usr/local/httpd-2.2.9

# ./apachectl -f /usr/local/apps/<APP\_DIR>/conf/httpd.conf --k start

* 1. Open the application url in a browser to see the results
  2. Stop the webserver

# ./apachectl -f /usr/local/apps/<APP\_DIR>/conf/httpd.conf --k stop

1. Update the GetAccess Runtime to ‘view’ the new daemon
   1. Stop the main apache webserver and the daemon to be integrated

# cd /usr/local/http-2.2.9/bin

# ./apachectl --k stop

# ./apachectl --f /usr/local/apps/<APP\_DIR>/conf/httpd.conf --k stop

* 1. Remove the commented lines from the daemon http config file

# cd /usr/local/apps/<APP\_DIR>/conf

# vi httpd.conf

Remove the hashmarks from the following three lines:

#LoadModule ga\_module /usr/local/Entrust/GetAccess/Runtime/Apache22/lib/libApache22Connector.so

#gaRuntimeRoot "/usr/local/Entrust/GetAccess/Runtime/Apache22"

#gaWebAgent main-CORE-DEV.hc-sc.gc.ca

And change "CORE-DEV" in the last line to match the environment server being provisioned.

* 1. Restart the main apache webserver and the daemon instance

# cd /usr/local/http-2.2.9/bin

# ./apachectl --k start

# ./apachectl --f /usr/local/apps/<APP\_DIR>/conf/httpd.conf --k start

* 1. Run ConfigureVS
     1. Log onto the core-dev server as user ‘akgart’
     2. # cd /usr/local/Entrust/GetAccess/Runtime/Apache22/bin
     3. # ./ConfigureVS.sh

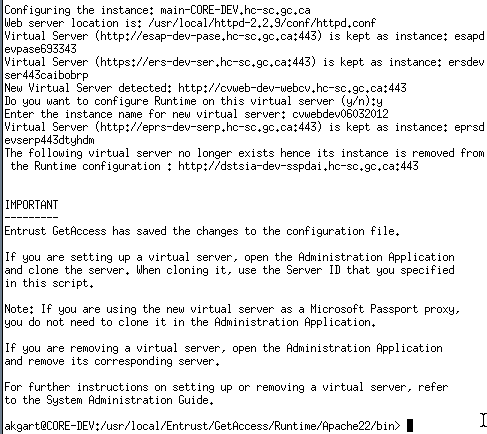
ConfigureVS should detect that your configuration has been changed and that a new runtime instance needs to be added.

* + 1. Answer ‘**y**’ to the prompt asking if you want to configure a runtime
    2. Create and enter a unique alpha-numeric id for your new webserver instance.

This unique ID should be partially related to the daemon being installed for human readability of the runtime configuration file, and also be partially random alphanumeric for security reasons.

This Id will be used as part of the key to bind the runtime to the GetAccess server

* + 1. Copy the ID to a piece of paper or Notepad. It will be required shortly when updating GetAccess (Server). See Pic below for the output of a successful integration.



NOTE: In the example above a daemon server was removed from the configuration at the same time. See the section ‘Removing a daemon from the GetAccess Runtime’ below for more details.

1. Create a New ‘Server’ on Entrust GetAccess
   1. Log onto the GetAccess Server (not the webserver, but the ‘SAML’ server)
   2. Select the ‘Servers’ tab
   3. Select a webserver that is on the same server as the one just installed and click clone
   4. Complete the information as required
      1. Enter the instance name that was enter in 6)d.v above as the webserver name
      2. Protocol is HTTP
      3. Servername.domain (without the unique ID, ex. cvweb-dev-webcv.hc-sc.gc.ca)
      4. Description (if desired)
   5. Keep the url entries the same as they are cloned
   6. Select ‘Save’
   7. Select the server just created in the list that appears and click ‘Update Webserver’ to complete an entry on the Runtime
2. Create a Resource Category and Resource with unique IDs for the server just created

NOTE: The Entrust GetAccess Business Administration Guide, or the HC GetAccess installation guide can be referenced for details if required

**NOTE:** You MUST select the server and click ‘Update Webserver’ again to push these changes to the runtime and in order to allow access to the directories of the app that you are adding.

1. Update the ‘extra/httpd-ssl.conf’ file for the daemon to specify the GAVirtualServerID:

If you try to access the server now, you will receive an access denied error. This is due to the configuration being used by HC. The daemon entry overwrites the parent entries and the webserver instance will not store its unique Id for GetAccess to recognize the server.

1. Stop the daemon  
   # ./apachectl --f /usr/local/apps/<APP\_DIR>/conf/httpd.conf --k stop
2. Locate the string that starts with ‘GAVirtualServerID’ from the main webserver (/usr/local/httpd-2.2.9/conf/extra/httpd-vhosts.conf) )that identifies this daemon to Getaccess.  
   # grep GAVirtualServerID /usr/local/httpd-2.2.9/conf/extra/httpd-vhosts.conf
3. Select and copy the string, it should look something like:  
    EX: GAVirtualServerID 179.3x.x.x\_443\_cvweb-dev-webcv.hc-sc.gc.ca\_84444
4. Go to the ‘<VirtualHost>’ section of /usr/local/apps/<APP\_DIR>/conf/extra/httpd-ssl.conf that was created in step 4)c.iii

You can paste it below the following instruction in the template.

## After you have ran the ConfigureVS script as described in the

## integration guide, copy the GA Virtual Server Id from the "MAIN" webserver

## location (not a daemon instance). You will find this ID in the

## 'extra/httpd-vhosts.conf' file. Copy and paste into the line

## beneath the example below.

## Ex: GAVirtualServerID 172.30.x.x\_443\_eds-dev-sed.hcsc.gc.ca\_76931

1. Save the file
2. Restart the daemon
3. Access your web page and login with required credentials

# Integrating Java & Tomcat

If your application uses Java and Tomcat then do the following to complete the generic installation steps for integrating the Tomcat component based on the Templates. There will always be custom configuration for each java application within Tomcat.

This should be completed by the developers, in conjunction with SADATE where required.

1. Install the required version of Tomcat into :

/usr/local/apache-tomcat-<version>-<app\_abbr>

EX:

/usr/local/apache-tomcat-6.0.29-esap

get the war file from

into /usr/local/apache-tomcat-<version>-<app\_abbr >/webapps and rename it to clin-rcil.war and dry-deploy (expanding to the app dir webapps/clin-rcil)

1. Update /usr/local/apps/<APP\_DIR>/conf/httpd.conf
   1. Stop the daemon
   2. To integrate java/tomcat with apache remove the hashmarks below as required.
      1. Remove the hashmark from ‘#Include conf/mod\_jk.conf’
      2. Remove the seven hashmarks starting with #<IfModule worker.c> to the end of the ‘if’ section (hashmarks to be removed are shown below in blue).

The text in the template file is included below.

#############################

#

### Uncomment the Include statement below if your GCCF

### application uses tomcat/jsp/java to

### Include the mod\_jk plugin for the java application

#

#Include conf/mod\_jk.conf

#

# For mod\_jk.so to connect with Tomcat worker MPM

#

# StartServers: initial number of server processes to start

# MaxClients: maximum number of simultaneous client connections

# MinSpareThreads: minimum number of worker threads which are kept spare

# MaxSpareThreads: maximum number of worker threads which are kept spare

# ThreadsPerChild: constant number of worker threads in each server process

# MaxRequestsPerChild: maximum number of requests a server process serves

### Remove hashmarks and set the values as required by your application

#<IfModule worker.c>

# ServerLimit 16

# StartServers 2

# MaxClients 150

# MinSpareThreads 25

# MaxSpareThreads 75

#</IfModule>

#

#############################

save and exit vi

* 1. vi <APP\_DIR>/conf/mod\_jk.conf

Update these three lines with the application directory:

* + - 1. JkWorkersFile /usr/local/apps/<APP\_DIR>/conf/workers.properties
      2. JkLogFile /usr/local/apps/<APP\_DIR>/logs/mod\_jk.log
      3. JkShmFile /usr/local/apps/<APP\_DIR>/logs/jk-runtime-status

Update the JkLogLevel if required.

Save the file and exit vi.

* 1. vi <APP\_DIR>/conf/workers.properties

EX: Make updates as required

* + 1. Update the tomcat installation directory
    2. Update the worker name in the file if you prefer. If you change this name from the default wkr01, you must remember the value later in step 12)e.ii
    3. Update the port to avoid collisions
    4. Update pool and connections sizes

Save the file and exit vi.

* 1. vi <APP\_DIR>/conf/extra/httpd-ssl.conf and update <APP\_DIR> as necessary
     1. Assuming the tomcat installation is complete and you know the path for your app, uncomment and update the following line to point to the folder where your java application is stored. For CANLINE, both <app> and <app-deployment-dir> = clin-rcil

#DocumentRoot "/usr/local/apache-tomcat-<version>-<app>/webapps/<app-deployment-dir>"

* + 1. Update the Directory Index setting and worker rules (with name and location) if required

Update the following line to the page used by your application. This is the line for CANLINE:

DirectoryIndex index-ext-eng.jsp index-ext-fra.jsp

Ensure there is worker object entry that points (are entries that point) the worker to the application objects in the tomcat deployment directory.

If tomcat is installed to ‘/usr/local/apache-tomcat-6.0.35-clin-rcil/’, then the mount point below (jkMount) is referencing the deployment directory at ‘/usr/local/apache-tomcat-6.0.35-clin-rcil/webapps/clin-rcil’. If you have changed the worker name above in step 12)d.ii it must be changed below as well (wkr01 is the default).

JkMount /clin-rcil/\* wkr01

* + 1. If required, this step is optional, uncomment and modify the Rewrite rule. Hashmarks in blue should be removed and the template address updated.

# <IfModule mod\_rewrite.c>

# RewriteEngine On

### Examples:

### RewriteRule ^/jira - [L]

### RewriteRule ^/(.\*) /jira/secure/$1 [P]

# RewriteRule =https://<appname> <-env-><nomdapp>.hc-sc.gc.ca https://<appname> <-env-><nomdapp>.hc-sc.gc.ca/<ALIAS\_DIR>/<PAGE\_NAME>.jsp [R]

# </IfModule>

return to 4)b.ix if required.

1. Bind the tomcat server to the IP used by your webserver
   1. Stop the tomcat server
   2. Edit the file server.xml file to bind the IP
      1. # cd /usr/local/apache-tomcat-<version>-<app\_abbr>/conf
      2. vi server.xml
      3. Add the IP for the App (same as used by webserver) to the connector port description string

<Connector port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

*becomes*

<Connector port="8080" protocol="HTTP/1.1"

**address="172.30.x.x"**

connectionTimeout="20000"

redirectPort="8443" />

* + 1. Add the IP for the App (same as used by webserver) to the AJP connection string

<Connector port="8011" protocol="AJP/1.3" redirectPort="8443" />

*becomes*

<Connector port="8011" **address="172.30.x.x"** protocol="AJP/1.3" redirectPort="8443" />

* 1. Save the file and exit
  2. Restart tomcat and verify the app

NOTE: Your application may or may not be using the same ports as above. It doesn’t matter as long as the ports selected are unique within the IP of your application space.

1. Complete your application configuration requirements with any custom configuration you require for tomcat and java: see section 4 of the document at sftp://sadateftp@utildev01/srv/ftp/sadateftp/CANLINE/Installation%20Instructions\_CANLINE\_V4.0.0\_External.doc). Be sure to test at each step of the way by restarting your server to make sure the integration is still working.
2. It may be necessary to now modify the ‘protected’ resources on the GetAccess server that were completed in Step # 8 above.

# Removing a daemon from the GetAccess Runtime

If ever required, you can remove daemon from the runtime instance by completing the following:

1. Stop the daemon server that is to be removed
2. Stop the main webserver
3. Remove or comment out the virtual host definition section for the daemon from the main web server’s ‘extra/httpd-ssl.conf’ file
4. Restart the main webserver
5. Run ConfigureVS as user ‘akgart’ to perform the update

**NOTE:** Do not forget to remove the entries from the GetAccess server to complete the removal properly.