Introduction

The CPE 5.5.12 release includes a preview of a new feature called **Content Validation**. This feature allows a custom plugin executing in the server to inspect content as it is uploaded and reject that which is deemed inappropriate, for example if it contains executable code or a virus.

As part of the preview a sample content validator is being provided which submits the content to a ClamAV virus detection daemon service.

Configuring the ClamAV Content Validator

In order to experiment with the sample handler you must deploy and manage a ClamAV instance which is reachable by a non-SSL TCP socket connection by the CPE servers. Make note of the host name or IP address of the ClamAV instance and the port on which it is listening. You may then configure the validator for one or more storage areas.

Two methods are provided for configuring the validator, manually using ACCE or automatically by executing code within the handler jar itself. The latter method will enable the validator for all storage areas defined within the targeted object store, so should be used cautiously. The manual methods allows selectivity over which storage areas are enabled.

It is helpful to understand the manual procedure even if you use the automatic method.

Do not mix manual and automatic methods in a given object store.

Manual configuration through ACCE

Open the object store in which you want to enable content validation.

The first step is to create a custom subclass of CmContentValidationAction an instance of which will define and configure the ClamAV validation handler.

- Create two property templates with symbolic names ClamAVHost, of type singleton string, and ClamAVPort, of type singleton integer.
- Under Data Design\Classes\Other Classes right click on Content Validation Action and select New Class. You can call the new class whatever you like, for example “ClamAV Validation Action”. Click Next, Finish then Open.
- Switch to the Property Definitions tab, select Add, then select both the ClamAV properties from the list that appears, followed by OK, then Save for the Class Definition. Leave the tab open.

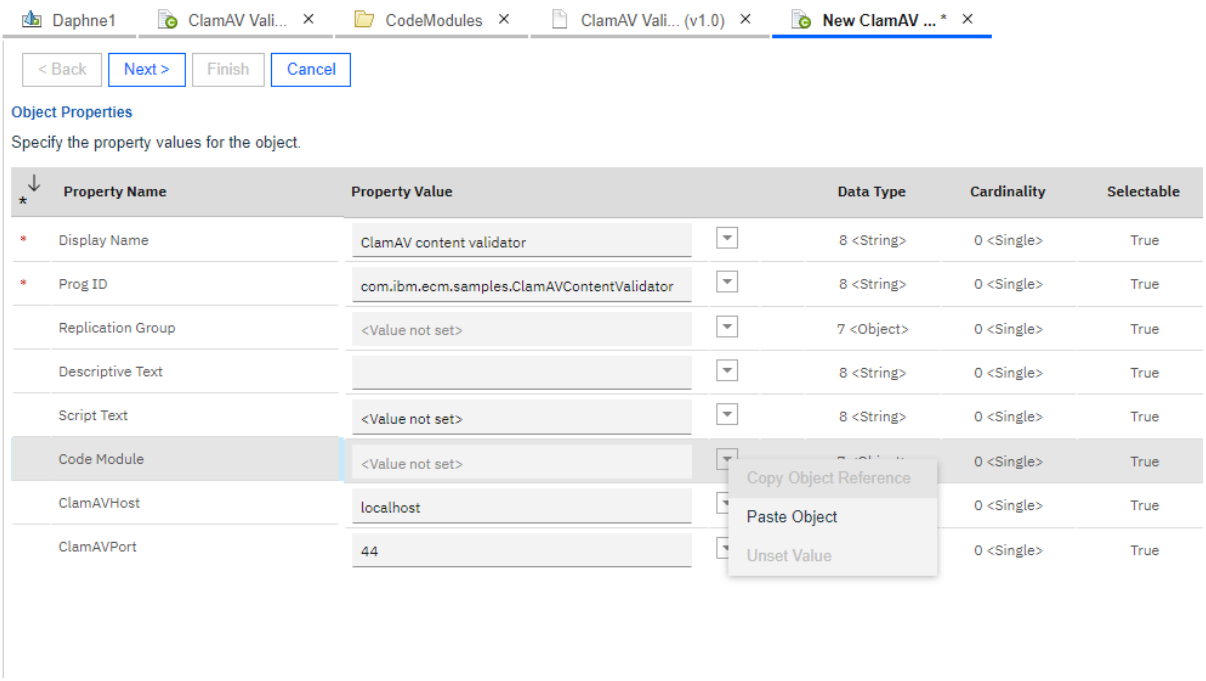
Next you will create a code module containing the code of the ClamAV handler.

- Open up Browse/Root Folder/CodeModules and select Action v New Document. Set the class to Code Module, give a title and enable With Content then click Next. On the content elements page click add and browse to and select ClamAV.jar
- Complete the wizard, and after Finish select Open. Leave the tab open.

Next you will create an instance of ClamAV Validation Action, link it to the code module just created and set other required properties.

- Return to the open tab of the ClamAV Validation Action class created earlier and click Actions v Create Instance. A Properties pane for the new instance will appear.

- Set a display name, enter com.ibm.ecm.samples.ClamAVContentValidator as the Prog ID and then fill in the ClamAVHost and ClamAVPort properties as appropriate for your ClamAV service instance.
- Switch to the open tab for the code module created earlier and from Actions select Copy Object Reference.
- Return to the tab for the new ClamAV Validation Action instance, drop down beside Code Module and select Paste Object



- Click Next, Finish and then Open. Leave the tab open.

The final step is to apply the newly create ClamAV Validation Action instance to selected storage areas.

- Locate the intended storage area under Administrative\Storage\StorageAreas or Advanced Storage\Advanced Storage Areas and click on it to open it in the right pane.
- Switch to the Properties tab and scroll down to make the Content Validator property visible.
- Switch to the open tab for the ClamAV Validation Action instance just created and select Actions v Copy Object Reference.
- Switch back to the storage area Properties tab, drop down beside Content Validator and select Paste Object. Then click the Save button.

ClamAV content validation is now enabled for that storage area. Repeat for additional storage areas if desired.

You are then advised to test that the configuration is correct by creating a document in the storage area in question. You can specify the storage area via the drop down on the Advanced Features page of the document creation wizard.

If creation fails with a message indicating that the content was rejected by the configured content validation handler, the most likely reason is that the server is unable to connect to the ClamAV instance. If that is because you gave the host or port incorrectly, you can resolve that problem simply

by editing the properties on the ClamAV Validation Action instance and saving the changes. Otherwise you may need to verify that the host of the ClamAV instance is reachable from the CPE servers.

If you need to enable additional storage areas later, in a new ACCE session you will first have to re-open the ClamAV Validation Action instance. Do this via Search in a new Object Store Search, selecting the ClamAV Validation Action class in the Class drop-down. After Running the search, click the instance in the result to open up the Properties pane from which again you can select the Copy Object Reference action.

Automatic configuration

To use automatic configuration you will need place ClamAV.jar in a directory which also contains the Content Engine Java API jar, Jace.jar. That can be obtained by running the CPE Client Installer, for example.

You will need the following information to proceed:

- The username and password for an administrative user of the target domain.
- The WSI URL for connecting to a CPE instance through which the configuration can be done.
- The symbolic name of the object store in which you wish to configure validation.
- The host and port number for the ClamAV instance you wish to use.

Then open a command prompt, change to the directory in which you placed the two jars and issue the following command:

```
java -jar ClamAV.jar user <admin user> password <admin password> url <CPE WSI URL>
    objstore <object store name> host <ClamAV host name> port <ClamAV port>
```

This will do much as the manual ACCE steps but with one important distinction:

- Create a subclass ClamAVValidationAction of CmContentValidationAction with custom properties ClamAVHost and ClamAVPort
- Create a code module containing ClamAV.jar (the instance you just ran)
- Create an instance of ClamAVValicationAction using said code module and with ClamAV host and port as specified.
- *Apply that action to all storage areas*.

The first step is idempotent, so you can run more than once and it will not duplicate anything.

The second step always creates a new code module so you can update the code.

The third step will only ever create one instance but it will always apply the potentially new host and port and the definitely new code module.