Xconf Server - User guide for configuration and feature validation

- Source code repository
- Software Requirements (Reference setup)
- Installation of dependencies
    - 1. Install Java JDK
    - 2. Install Maven (Version 3.6.0)
    - 3. Download xconf server code
    - 4. Install and setup Cassandra(Version 3.11.9)
        - Install Cassandra
        - Configure Cassandra
- Configuration and Service startup
    - 1. Start Cassandra Service
    - 2. Configure and Start Application Services
        - a. Build Project
        - b. Configure Angular Admin UI
            - Active Profile Option (for development purpose):
        - c. Configure DataService
            - Data Service Endpoints
    - 3. Expected build issues
- Configuration and Validation of services
    - Admin UI Common Configuration
        - Define Environments
        - Define Models
        - Define MAC List
        - Define IP list
    - Feature Validation(RFC)
        - Configuration
            - Define the feature:
            - Define the Feature Rule
        - Verification
        - Client end verification (RPI)
    - Feature Validation (LogUpload)
        - Configuration
        - Verification
        - Client end verification (RPI)
    - Feature Validation (Telemetry)
        - Configuration
        - Verification
        - Client end verification (RPI)
    - Feature Validation (Firmware update)
        - Configuration
        - Add TFTP location
        - Override the default download Location set at Download Location Filter
            - Override Firmware Location with http
        - Verification
        - Client end verification (RPI)
        - Percent Filter
- Import and Export Feature
- FAQs and Common Issues faced in setup

Source code repository

Xconf consists of 2 web applications - Xconf dataservice and Xconf admin. Xconf DataService is the app that the STBs talk to. Xconf Admin allows humans to enter all the information necessary for Xconf to provide the correct information to STBs.

This repo contains the source code for both the applications - https://github.com/rdkcentral/xconfserver

Software Requirements (Reference setup)

Component	Recommendation
System/OS	Ubuntu 18.04.1 LTS 64 bit
Disk space	> 5GB
GIT	Version 2.17.1

Python	2.7x
Maven	3.6.0
Java/JDK	Java 8 (JDK version 1.8.0_282)

Installation of dependencies

1. Install Java JDK

Java JDK version should be **8.** Get the supported version from Oracle or use the OpenJDK packages.

Steps to install Open JDK

```
    $ sudo apt-get update
    $ sudo apt-get install openjdk-8-jdk

Check your installation using the command :
    $ java  -version
```

2. Install Maven (Version 3.6.0)

Maven version should be 3 +.

To install maven follow these steps:

```
    $ sudo apt update
    $ sudo apt install maven

Check the installation using :
    $ mvn -version
```

3. Download xconf server code

We can download the latest xconfserver code from https://github.com/rdkcentral/xconfserver. The latest version of the code is available in main branch.

-
  ```
  Create a folder
  $ mkdir xconf
  ```

-
  ```
  Step into the folder & clone the repo
  $ cd xconf/
  $ git clone https://github.com/rdkcentral/xconfserver.git -b main
  ```

- To clone a particular tagged release :
  $ You will get the name and details of each tag here in this page -https://github.com/rdkcentral/xconfserver/tags
  $ Clone step git clone --depth 1 --branch <tag-name> https://github.com/rdkcentral/xconfserver.git
  eg : git clone --depth 1 --branch v1.3.11 https://github.com/rdkcentral/xconfserver.git

4. Install and setup Cassandra(Version 3.11.9)

Install Cassandra

To install Cassandra , follow the below steps

- Download the tarball file for the version 3.11.9 :
  ```
  $ wget -c https://archive.apache.org/dist/cassandra/3.11.9/apache-cassandra-3.11.9-bin.tar.gz
  ```

-
  ```
  Unpack the tarball :
  $ tar -xvf  apache-cassandra-3.11.9-bin.tar.gz
  ```
- Step into apache-cassandra-3.11.9 folder :
  ```
  $ cd apache-cassandra-3.11.9
  ```
- To start Cassandra, run the following command
  ```
  $ sudo bin/cassandra
  ```
- To verify that Cassandra is up and running, enter the following command :
  ```
  $ bin/nodetool status
  ```

Configure Cassandra

Note : For the next step, make sure that python is installed. Because cqlsh is python based command line tool. If python is not installed , use this command : sudo apt install python2.7

- schema.cql file is available in 'xconf-angular-admin/src/test/resources/schema.cql'. We can use this cql file to create a corresponding schema . Open another terminal , step into apache-cassandra-3.11.9 folder and run the following command

```
    $ bin/cqlsh -f  { path-to-the-schem.cql file}
```

```
eg : $ bin/cqlsh  -f ~/xconf/xconfserver/xconf-angular-admin/src/test/resources/schema.cql
```

- To check if tables are created successfully, we can use cqlsh
- To start cqlsh, step into cassandra folder and enter the command:

```
  $ bin/cqlsh
```

- It gives cassandra cqlsh prompt as output. To check if all the tables are present enter the following commands in cqlsh prompt :

```
  cqlsh>  USE "demo";

  cqlsh>  DESCRIBE KEYSPACE;
```

- To exit from cqlsh prompt,

```
  cqlsh> quit
```

Production Installation

The production installation should be similar to the local installation, except that Cassandra will be installed to multiple hosts. Please see the Apache Cassandra documentation for more information.

Configuration and Service startup

1. Start Cassandra Service

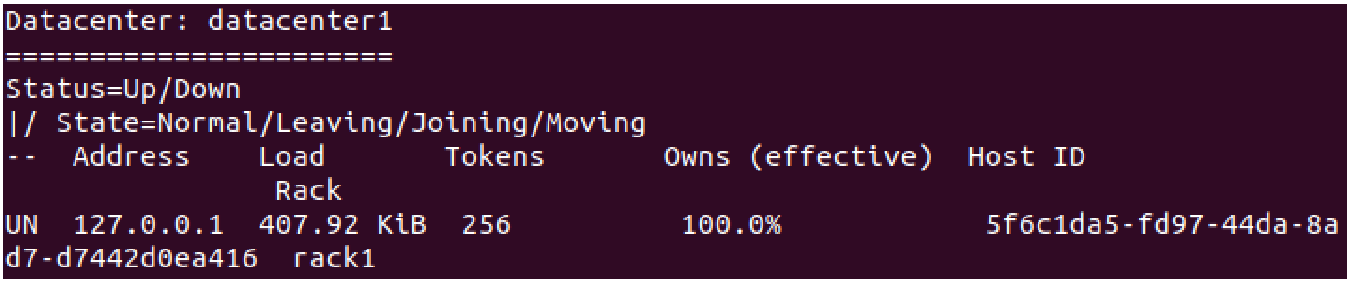
- To start an Xconf application, start the Cassandra server by executing the following commands:

```
$ cd apache-cassandra-3.11.9
$ sudo bin/cassandra
```

- Status of xconf server can be verified by using the command

```
$ bin/nodetool status
```

You will get an output like this



```
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address    Load       Tokens       Owns (effective)  Host ID
                Rack
UN  127.0.0.1  407.92 KiB  256          100.0%            5f6c1da5-fd97-44da-8a
d7-d7442d0ea416  rack1
```

2. Configure and Start Application Services

Build and run steps mentioned below is based on these steps - https://github.com/rdkcentral/xconfserver#run-application.

a. Build Project

- Go to the xconf-server folder and run the following command to download all dependencies.

```
$ cd ~/xconf/xconfserver
```

- Run the following command from the xconfserver folder

  $ mvn clean install

- Or you can run this command with unit tests skipped
  $ mvn clean install -DskipTests=true

b. Configure Angular Admin UI

For first time application deployment, create a "*service.properties"* file under the path xconfserver/xconf-angular-admin/src/main/resources/service.properties with the following contents.

The sample service.properties file will be available in xconf-angular-admin/src/test/resources/service.properties, the below content is copied from the sample with a modification in cassandra port you can use this.

```
cassandra.keyspaceName=demo
cassandra.contactPoints=127.0.0.1
cassandra.username=
cassandra.password=
cassandra.port=9042
cassandra.authKey=

dataaccess.cache.tickDuration=60000
dataaccess.cache.retryCountUntilFullRefresh=10
dataaccess.cache.changedKeysTimeWindowSize=900000
dataaccess.cache.reloadCacheEntries=false
dataaccess.cache.reloadCacheEntriesTimeout=1
dataaccess.cache.reloadCacheEntriesTimeUnit=DAYS
dataaccess.cache.numberOfEntriesToProcessSequentially=10000
dataaccess.cache.keysetChunkSizeForMassCacheLoad=500
dataaccess.cache.changedKeysCfName=XconfChangedKeys4
```

- Go to xconf-angular-admin folder

```
$ cd ~/xconf/xconfserver/xconf-angular-admin
```

- Run the following command from xconf-angular-admin folder

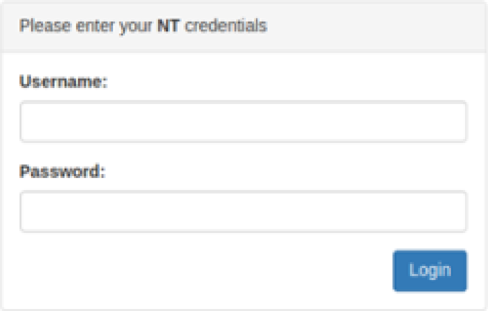
```
$ mvn jetty:run -DappConfig=${path-to-service-properties} -f pom.xml
```

- For first time run only, we need to specify the path to service.properties . For the subsequent runs execute the below command in the folder xconfserver/xconf-angular-admin:

```
$ mvn jetty:run
```

To run the admin UI launch it as **http://<XCONF-SERVER-IP>:19093/admin/** in any browser(Default port is set as 19093, it can be changed by using the option -Djetty.port=[port number]). This will redirect to the login page.

To launch in localhost : http://127.0.0.1:19093/admin



If the user wants both read and write permissions ,then enter username and password for the login as **admin** and **admin** respectively

If the user wants only read permissions ,then enter username and password for the login as **user** and **user** respectively.

Active Profile Option (for development purpose):

If xconf-angular-admin is run with -Dspring.profiles.active=dev UI will use not compiled .js and .css files but the source files. See xconf-angular-admin/src/main/webapp/WEB-INF/jsp/xconfindex.jsp for details. That can be useful for local development purpose, to update UI it is just needed to reload page with cache refresh option.

c. Configure DataService

For first time application deployment, create a "*service.properties"* file under the path xconfserver/xconf-dataservice/src/main/resources/service.properties with the following contents. The sample service.properties file will be available in xconf-dataservice/src/test/resources/sample-service.properties(There are some mistakes in that sample file - 1. cassandra.keyspaceName=demo 2. dataaccess.cache.changedKeysCfName=XconfChangedKeys4 that is rectified below. You can also edit that file with the changes 1 and 2. Then rename it to be used here), the below content is taken from there and modified with change in cassandra port .

```
cassandra.keyspaceName=demo
cassandra.contactPoints=127.0.0.1
cassandra.username=
cassandra.password=
cassandra.port=9042
cassandra.authKey=

dataaccess.cache.tickDuration=60000
dataaccess.cache.retryCountUntilFullRefresh=10
dataaccess.cache.changedKeysTimeWindowSize=900000
dataaccess.cache.reloadCacheEntries=false
dataaccess.cache.reloadCacheEntriesTimeout=1
dataaccess.cache.reloadCacheEntriesTimeUnit=DAYS
dataaccess.cache.numberOfEntriesToProcessSequentially=10000
dataaccess.cache.keysetChunkSizeForMassCacheLoad=500
dataaccess.cache.changedKeysCfName=XconfChangedKeys4
```

- Step into xconf-dataservice folder

  ```
  $ cd ~xconf/xconfserver/xconf-dataservice
  ```

- Run the following command from xconf-dataservice folder

  ```
  $ mvn jetty:run -DappConfig=${path-to-service-properties} -f pom.xml
  ```

- For first time run only, we need to specify the path to service.properties . For the subsequent runs execute the below command in the folder xconfserver/xconf-dataservice:

```
$ mvn jetty:run
```

To launch the application go to **http://<XCONF-SERVER-IP>:19092/queries/environments** (Default port is set as 19092, it can be changed by using the option -Djetty.port=[port number]) . To verify, add an entry in the environments tab of the Xconf admin application and check whether the same is updated here in data service.

To launch in localhost : http://127.0.0.1:19092/queries/environments

- **Data Service Endpoints**

  The endpoints available in data service is listed and described in below link:

  https://github.com/rdkcentral/xconfserver#endpoints

NOTE: To run the Admin UI and data service applications in background start jetty server as follows: **nohup mvn jetty:run &**

3. Expected build issues

Below exceptions may be observed during the mvn clean install . This is an exception from unit test , but the tests will run successfully.

Build process may stall for some time but the build will be successful and the application can be launched successfully. So it can be ignored as well.

```
[INFO] Running com.comcast.xconf.CompleteTestSuite

no libsigar-amd64-linux.so in java.library.path

org.hyperic.sigar.SigarException: no libsigar-amd64-linux.so in java.library.path
                at org.hyperic.sigar.Sigar.loadLibrary(Sigar.java:172)
                at org.hyperic.sigar.Sigar.<clinit>(Sigar.java:100)
                at org.apache.cassandra.utils.SigarLibrary.<init>(SigarLibrary.java:47)
                at org.apache.cassandra.utils.SigarLibrary.<clinit>(SigarLibrary.java:28)
                at org.apache.cassandra.service.StartupChecks$7.execute(StartupChecks.java:216)
                at org.apache.cassandra.service.StartupChecks.verify(StartupChecks.java:112)
                at org.apache.cassandra.service.CassandraDaemon.setup(CassandraDaemon.java:196)
                at org.apache.cassandra.service.CassandraDaemon.activate(CassandraDaemon.java:601)
                at org.cassandraunit.utils.EmbeddedCassandraServerHelper$1.run(EmbeddedCassandraServerHelper.
java:133)
                at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
                at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
                at java.lang.Thread.run(Thread.java:748)
```

Solution : This exception can be resolved by copying the .so file to the path /usr/lib.

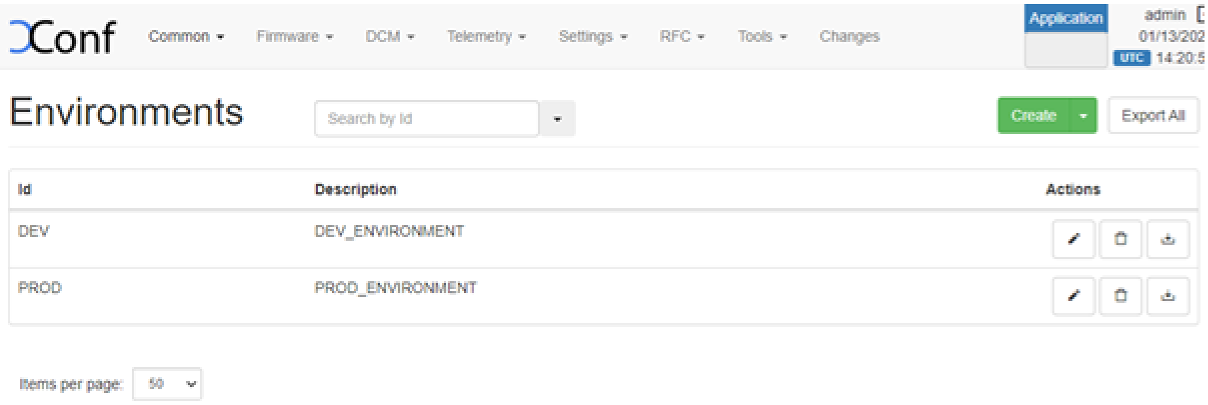
Configuration and Validation of services

Admin UI Common Configuration

Below steps will affect all the features in Xconf system and should be configured after initial setup. Go to the steps given in site navigation step and press on create button to create new entries.

Define Environments

Site Navigation: <xconf-server>:19093 >> Common >> Environments | Example URL:

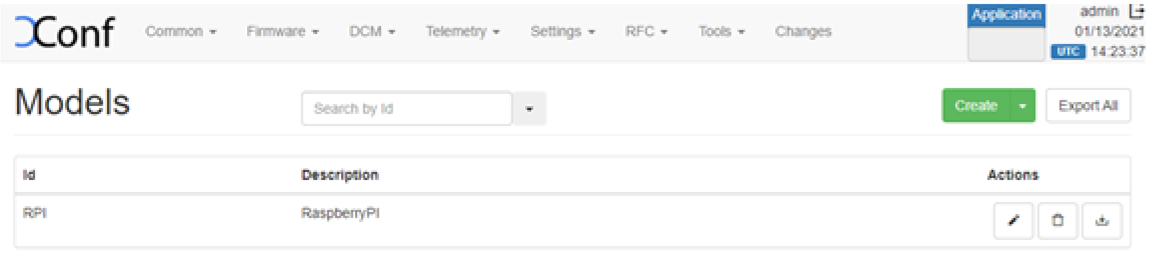


Environments

Id	Description	Actions
DEV	DEV_ENVIRONMENT	
PROD	PROD_ENVIRONMENT	

Define Models

Site Navigation: http://<XCONF-SERVER>:19093 >> Common >> Models | Example URL :

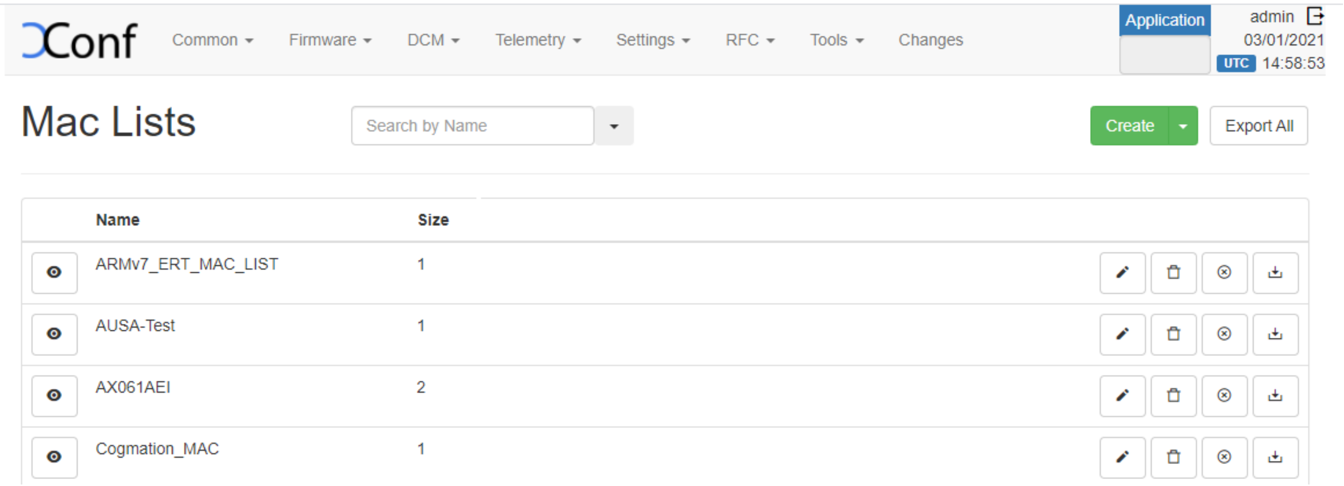


Id	Description	Actions
RPI	RaspberryPI	

Define MAC List

(This will be used to target certain list of MACs against a particular feature configuration)

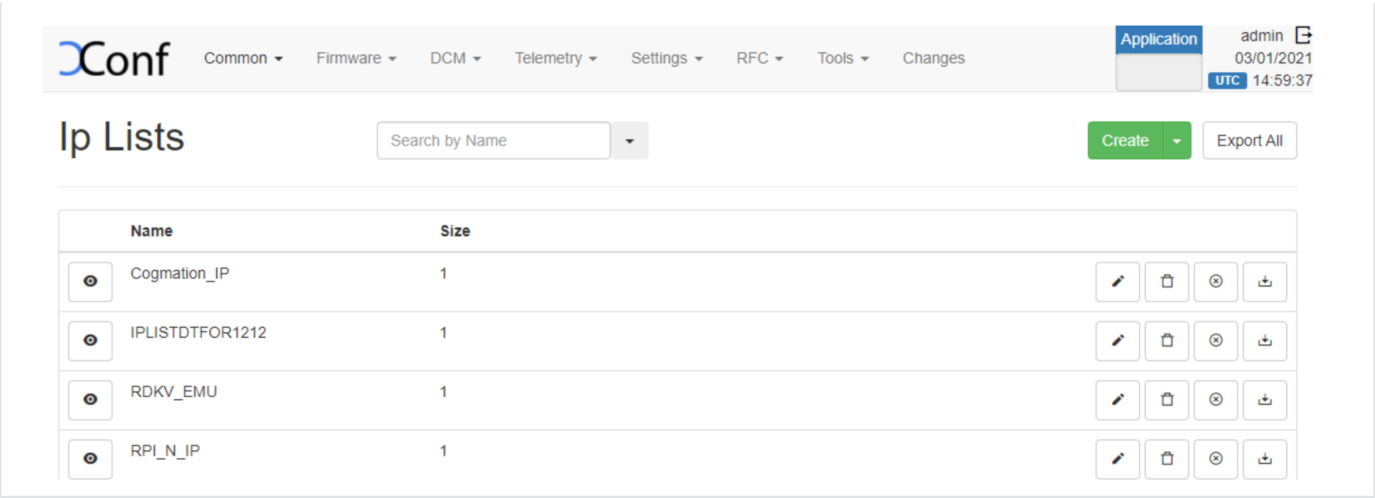
Site Navigation http://<XCONF-SERVER>:19093 >> Common >> MAC List | Example URL :



Name	Size
ARMv7_ERT_MAC_LIST	1
AUSA-Test	1
AX061AEI	2
Cogmation_MAC	1

Define IP list

Site Navigation http://<XCONF-SERVER>:19093 >> Common >> IP List | Example URL :



Ip Lists

Name	Size
Cogmation_IP	1
IPLISTDTFOR1212	1
RDKV_EMU	1
RPI_N_IP	1

Feature Validation(RFC)

Configuration

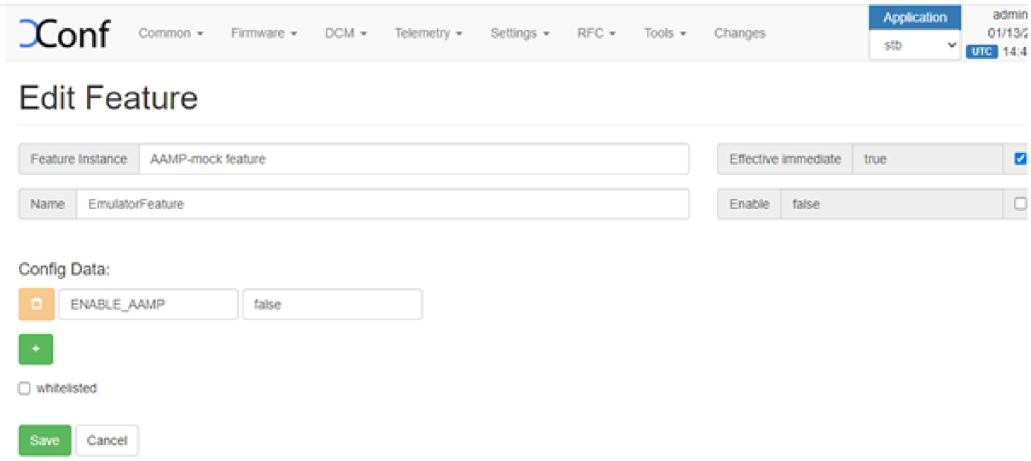
RDK Feature control configuration can be added by adding below 2 sections

1. Define the Feature
2. Define the Feature Rule

Define the feature:

A new feature can be defined via RFC-> *Feature -> Create.* 'Feature Name' should be unique and understandable, 'Config data' should be key value pairs.

Site Navigation | http://<XCONF_SERVER>:19093 >> RFC >> Feature



Edit Feature

Feature Instance: AAMP-mock feature

Name: EmulatorFeature

Effective immediate: true ☑

Enable: false ☐

Config Data:

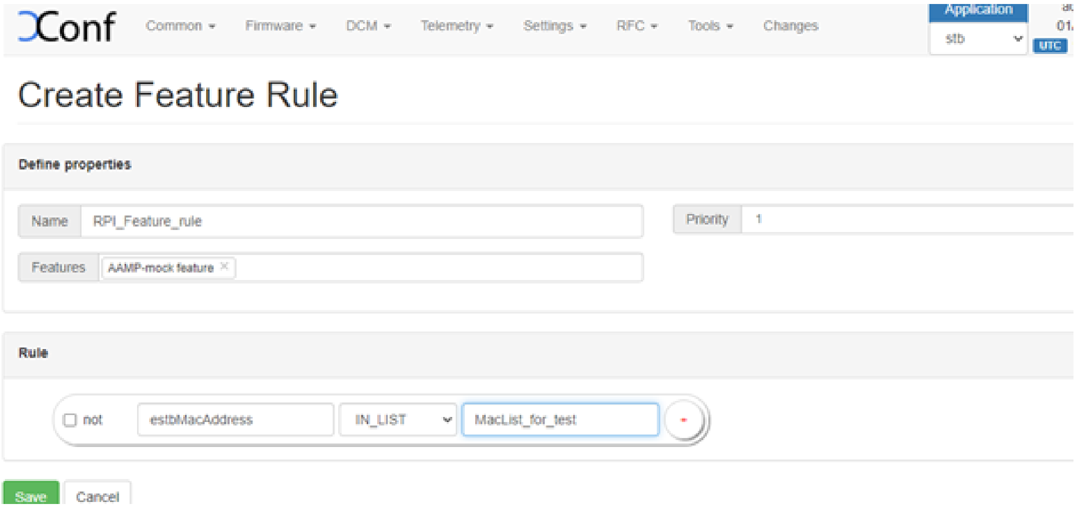
ENABLE_AAMP false

☐ whitelisted

Define the Feature Rule

Feature rule is to map devices to a particular feature. A new feature rule can be created via RFC->feature rule -> Create

Site Navigation | http://<XCONF_SERVER>:19093 >> RFC >> Feature Rule



Create Feature Rule

Define properties

Name: RPI_Feature_rule

Priority: 1

Features: AAMP-mock feature ×

Rule

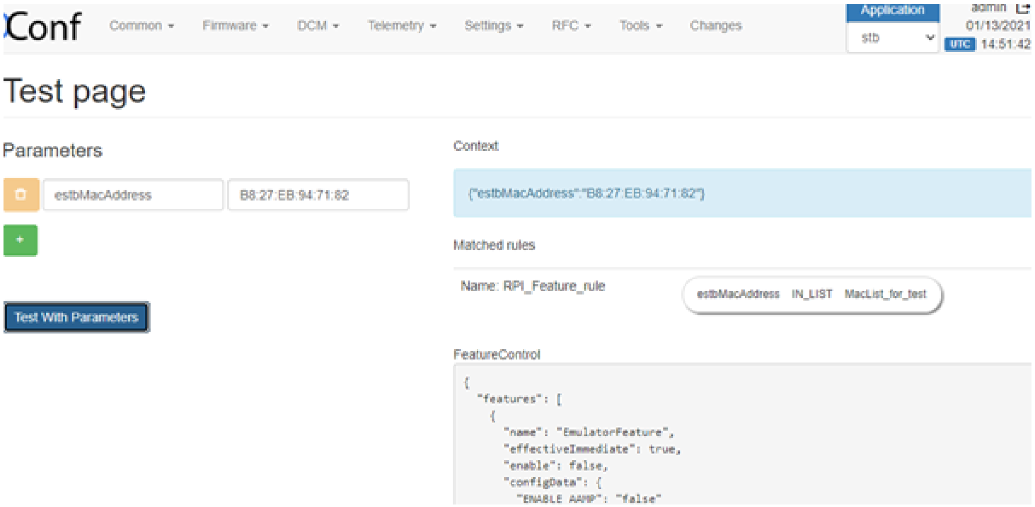
□ not | estbMacAddress | IN_LIST | MacList_for_test | -

Verification

a. Verification of feature and feature rule via test page.

After creating the feature and feature rule, go to RFC->Test page and give a parameter that will match the one of the rules that you have created. The matched rule and JSON response will be displayed similar to below example.

Site Navigation: http://<XCONF_SERVER>:19093 >> RFC >> TestPage



01/13/2021 14:51:42

Test page

Parameters

estbMacAddress | B8:27:EB:94:71:82

Context

{"estbMacAddress":"B8:27:EB:94:71:82"}

Matched rules

Name: RPI_Feature_rule

estbMacAddress IN_LIST MacList_for_test

FeatureControl

```
{
  "features": [
    {
      "name": "EmulatorFeature",
      "effectiveImmediate": true,
      "enable": false,
      "configData": {
        "ENABLE_AAMP": "false"
```

b. Verification via curl command

The curl command mocks the request being sent from an STB like below and sample response is also given. It can be given as a curl command or as a get request via postman or browser

eg :

$ curl 'http://<XCONF_IP>:19092/featureControl/getSettings?estbMacAddress= B8:27:EB:94:71:82'

*(*Here the feature rule mapped to this particular mac address will be obtained)

Sample Response:

```
{
    "featureControl": {

        "features": [
            {
                "name": "EmulatorFeature",
                "effectiveImmediate": true,
                "enable": false,

                "configData": {
                    "ENABLE_AAMP": "false"
                },

                "featureInstance": "AAMP-mock feature"
            }
        ]
    }
}
```

Client end verification (RPI)

Verification and setup from RPI

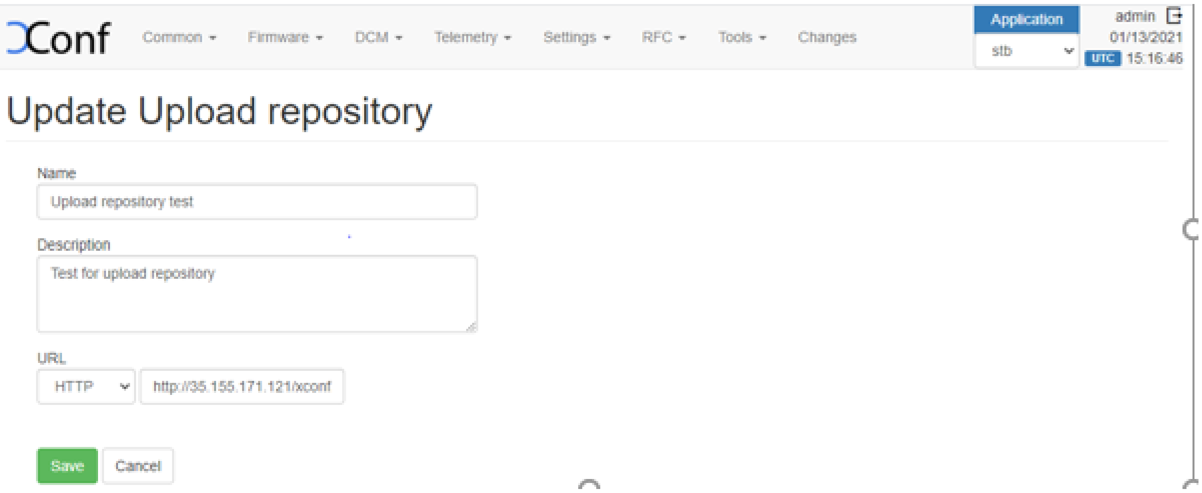
CURL Command	curl 'http://<XCONF_IP>:19092/featureControl/getSettings?estbMacAddress=B8:27:EB:FF:54:95&firmwareVersion=rdk-generic-hybrid-wpe-image_default_20190702100618&env=pi&model=RPI&ecmMacAddress=B8:27:EB:FF:54:95&controllerId=2504&channelMapId=2345&vodId=15660&partnerId=&accountId=Unknown&version=2'
CPE Script (RDK-V)	/lib/rdk/RFCbase.sh
CPE Service (RDK-V)	/lib/systemd/system/rfc-config.service

Feature Validation (LogUpload)

Configuration

1. Create upload repository via DCM->Upload repository -> Create.   Here we can add where to configure the log upload, i.e. the upload URL and protocol (This will be the URL of logopload server that is setup to upload the log files, it can be http, https or tftp servers).

Site Navigation | http://<XCONF_SERVER>:19093 >> DCM >> UploadRepository



Update Upload repository

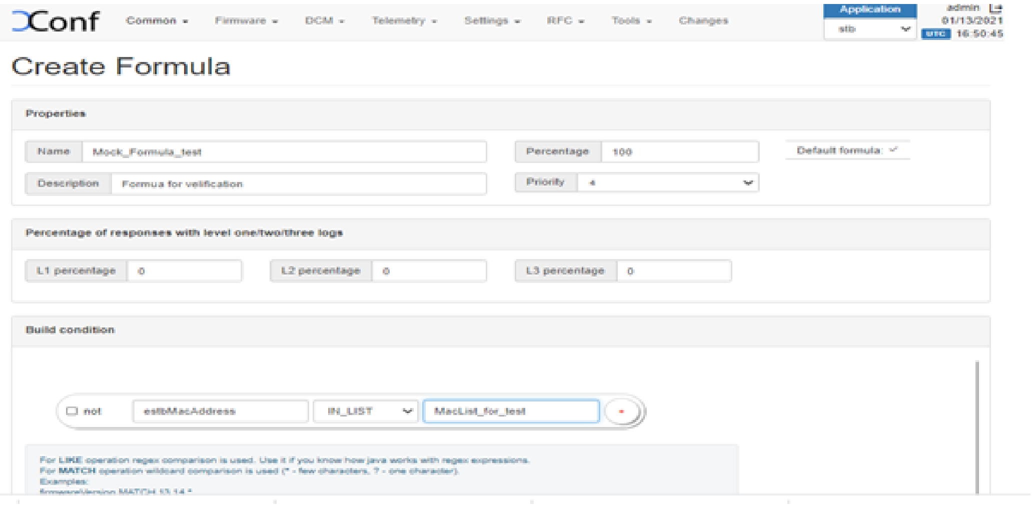
Name
Upload repository test

Description
Test for upload repository

URL
HTTP http://35.155.171.121/xconf

2. Create rule via DCM->Formulas->Create.

Site Navigation | http://<XCONF_SERVER>:19093 >>DCM >> Formulas



Create Formula

Properties

Name: Mock_Formula_test

Description: Formua for velification

Percentage: 100

Priority: 4

Default formula: ✓

Percentage of responses with level one/two/three logs

L1 percentage: 0

L2 percentage: 0

L3 percentage: 0

Build condition

□ not | estbMacAddress | IN_LIST | MacList_for_test

For **LIKE** operation regex comparison is used. Use it if you know how java works with regex expressions.
For **MATCH** operation wildcard comparison is used (* - few characters, ? - one character).
Examples:
firmwareVersion MATCH 13.14.*

3. Once you save the created formula , then a ‘define Settings’ tab with ‘Create Device Settings’, ‘Create log settings’, Create VOD settings’ will be available

4. Click on 'Create Device Settings' tab and edit the Device Settings.

5. Edit the Log upload Setting (Create schedule & Add the upload repository created beforehand).

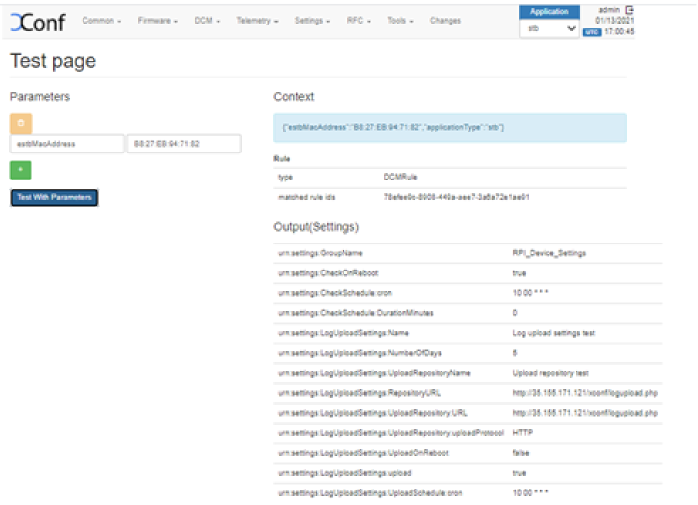
6. **Note** : The formula will be effective only if we select 'Are Settings Active' option to 'true' in 'Create Device Settings' and 'Log Upload settings'

Verification

a.Verification of log upload settings test page.

After creating the feature and feature rule, go to DCM->Test page and give a parameter that will match the one of the formulas that you have created. Then matched rule and the settings will be displayed like below

Site Navigation: <xconf-server>:19093 >> DCM >> Test Page



Test page

Parameters

estbMacAddress: B8:27:EB:94:71:82

Context

{"estbMacAddress":"B8:27:EB:94:71:82","applicationType":"stb"}

Rule

type	DCMRule
matched rule ids	78afee0c-8908-440a-aee7-3a8a72e1ae91

Output(Settings)

urn:settings:GroupName	RPI_Device_Settings
urn:settings:CheckOnReboot	true
urn:settings:CheckSchedule:cron	10 00 * * *
urn:settings:CheckSchedule:DurationMinutes	0
urn:settings:LogUploadSettings:Name	Log upload settings test
urn:settings:LogUploadSettings:NumberOfDays	5
urn:settings:LogUploadSettings:UploadRepositoryName	Upload repository test
urn:settings:LogUploadSettings:RepositoryURL	http://35.155.171.121/xconf/logupload.php
urn:settings:LogUploadSettings:UploadRepository:URL	http://35.155.171.121/xconf/logupload.php
urn:settings:LogUploadSettings:UploadRepository:uploadProtocol	HTTP
urn:settings:LogUploadSettings:UploadOnReboot	false
urn:settings:LogUploadSettings:upload	true
urn:settings:LogUploadSettings:UploadSchedule:cron	10 00 * * *

b. Verification via curl command

The curl command mocks the request being sent from an STB like below and sample response is also given. It can be given as a curl command or as a get request via postman or browser *eg :*

$ curl 'http://<XCONF_IP>: 19092/loguploader/getSettings?estbMacAddress=B8:27:EB:94:71:82’.

Sample response :

```
{
    "urn:settings:GroupName": "RPI_Device_Settings",

    "urn:settings:CheckOnReboot": true,
    "urn:settings:CheckSchedule:cron": "10 00 * * *",
    "urn:settings:CheckSchedule:DurationMinutes": 0,
    "urn:settings:LogUploadSettings:Message": null,

    "urn:settings:LogUploadSettings:Name": "Log upload settings test",
    "urn:settings:LogUploadSettings:NumberOfDays": 5,

    "urn:settings:LogUploadSettings:UploadRepositoryName": "Upload repository test",
    "urn:settings:LogUploadSettings:RepositoryURL": "http://35.155.171.121/xconf/logupload.php",
    "urn:settings:LogUploadSettings:UploadOnReboot": false,
    "urn:settings:LogUploadSettings:UploadImmediately": false,
    "urn:settings:LogUploadSettings:upload": true,

    "urn:settings:LogUploadSettings:UploadSchedule:cron": "10 00 * * *",
    "urn:settings:LogUploadSettings:UploadSchedule:levelone:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:leveltwo:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:levelthree:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:DurationMinutes": 0,
    "urn:settings:VODSettings:Name": null,
    "urn:settings:VODSettings:LocationsURL": null,
    "urn:settings:VODSettings:SRMIPList": null
}
```

Client end verification (RPI)

CURL Command	curl 'http://<XCONF_IP>:19092/loguploader/getSettings?estbMacAddress=B8:27:EB:FF:54:95&firmwareVersion=rdk-generic-hybrid-wpe-image_default_20190702100618&env=dev&model=RPI&ecmMacAddress=B8:27:EB:FF:54:95&controllerId=2504&channelMapId=2345&vodId=15660&timezone=&partnerId=&accountId=Unknown&version=2'
CPE Script (RDK-V)	/lib/rdk/StartDCM.sh /lib/rdk/DCMscript.sh
CPE Service (RDK-V)	/lib/systemd/system/dcm-log.service

Feature Validation (Telemetry)

1. Telemetry configuration can be done by adding a permanent profile which contains below objects
   a. Upload repository
   b. Profile options (Header, content, frequency etc.)
2. Creating a targeting rule which is basically mapping the profile to a set of MAC/IP/Device etc.

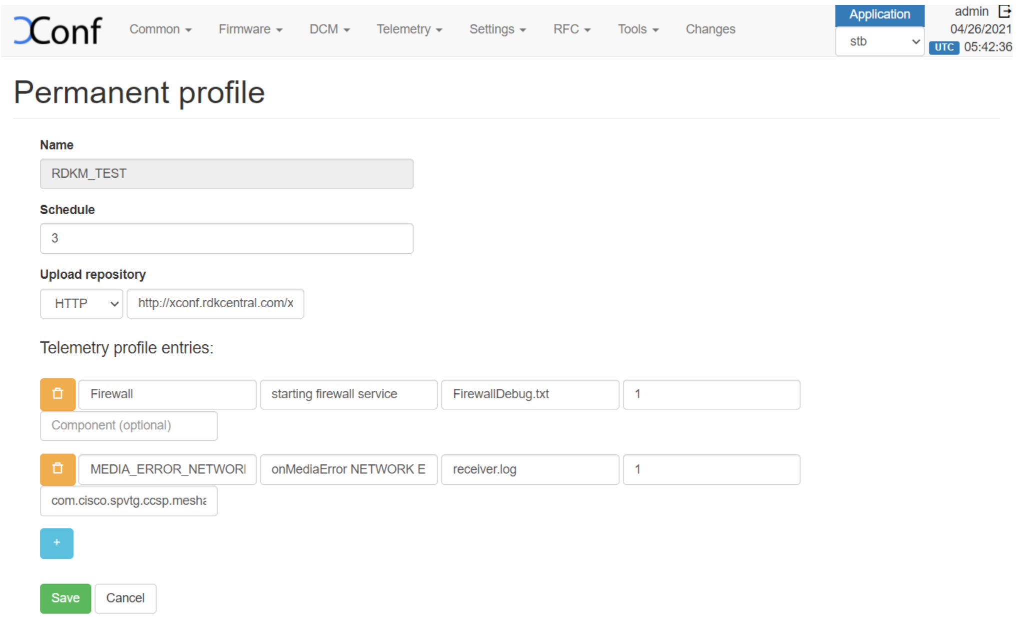
Note : DCM settings should be already done for the devices that you are going to set telemetry configuration

Configuration

1. Create a permanent profile Telemetry - > Permanent Profiles -> Create

In the Telemetry Permanent Profile page, there will be a 5th column that allows a component name to be entered. The component name is optional and may be present for only some of the entries in the Telemetry profile.

Site Navigation: http://<XCONF_SERVER>:19093 >> Telemetry >> Permanent Profiles



Permanent profile

Name

RDKM_TEST

Schedule

3

Upload repository

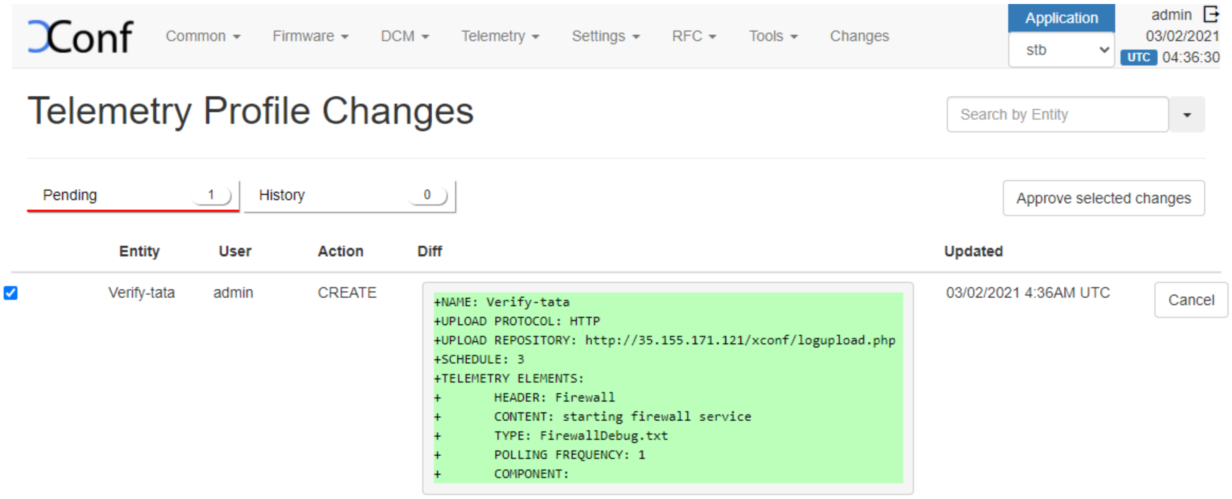
HTTP | http://xconf.rdkcentral.com/x

Telemetry profile entries:

Firewall	starting firewall service	FirewallDebug.txt	1
Component (optional)			
MEDIA_ERROR_NETWORI	onMediaError NETWORK E	receiver.log	1
com.cisco.spvtg.ccsp.mesha			

2. Once you save the permanent profile, you will get a message overlay ‘Profile added to the pending changes’. Then go to Changes -> select the profile you create -> Click on “Approve selected changes”.
Then the permanent profile will be listed under Telemetry - > Permanent profiles

Site Navigation: http://<XCONF_SERVER>:19093 >> Changes



Telemetry Profile Changes

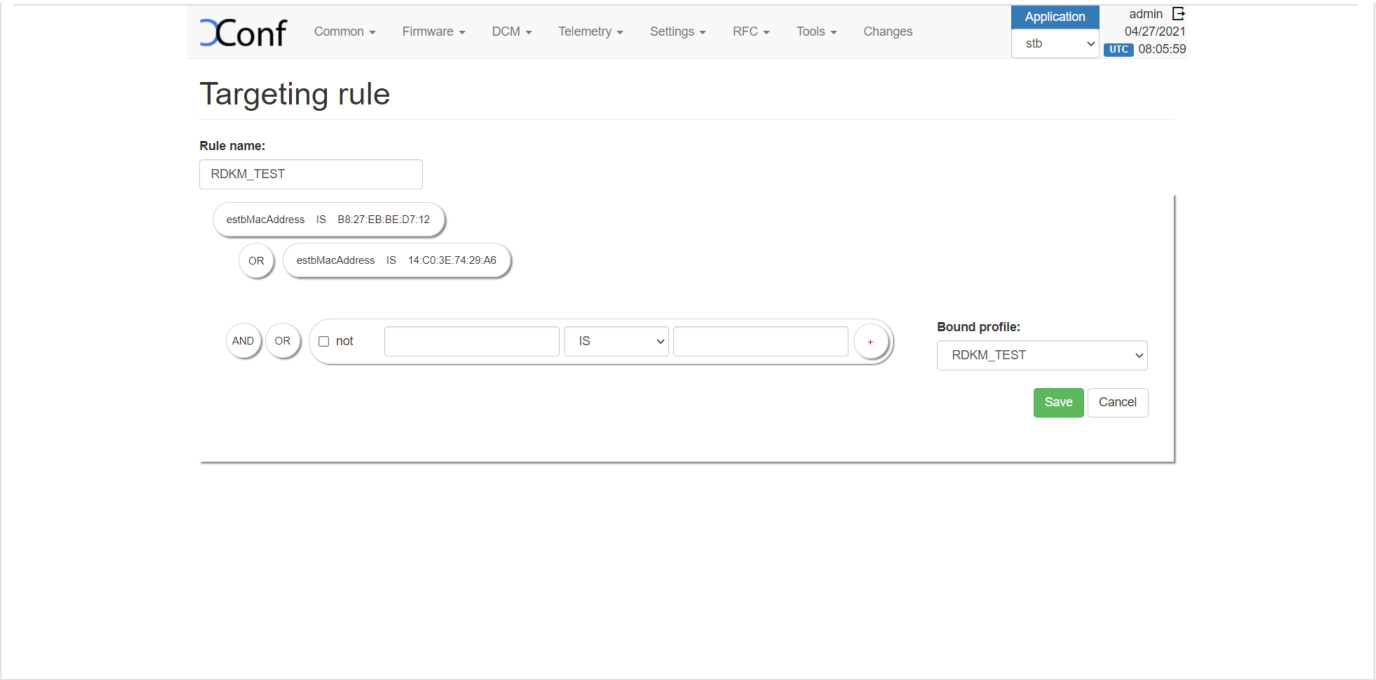
Pending 1 | History 0

	Entity	User	Action	Diff	Updated
☑	Verify-tata	admin	CREATE		03/02/2021 4:36AM UTC

```
+NAME: Verify-tata
+UPLOAD PROTOCOL: HTTP
+UPLOAD REPOSITORY: http://35.155.171.121/xconf/logupload.php
+SCHEDULE: 3
+TELEMETRY ELEMENTS:
+       HEADER: Firewall
+       CONTENT: starting firewall service
+       TYPE: FirewallDebug.txt
+       POLLING FREQUENCY: 1
+       COMPONENT:
```

3. Create targeting rule via Telemetry -> Targeting rules. Targeting rules is to map the profiles with rules.

Site Navigation: http://<XCONF_SERVER>:19093 >> Telemetry >> Targeting Rule



Targeting rule

Rule name: RDKM_TEST

estbMacAddress IS B8:27:EB:BE:D7:12

OR estbMacAddress IS 14:C0:3E:74:29:A6

□ not IS

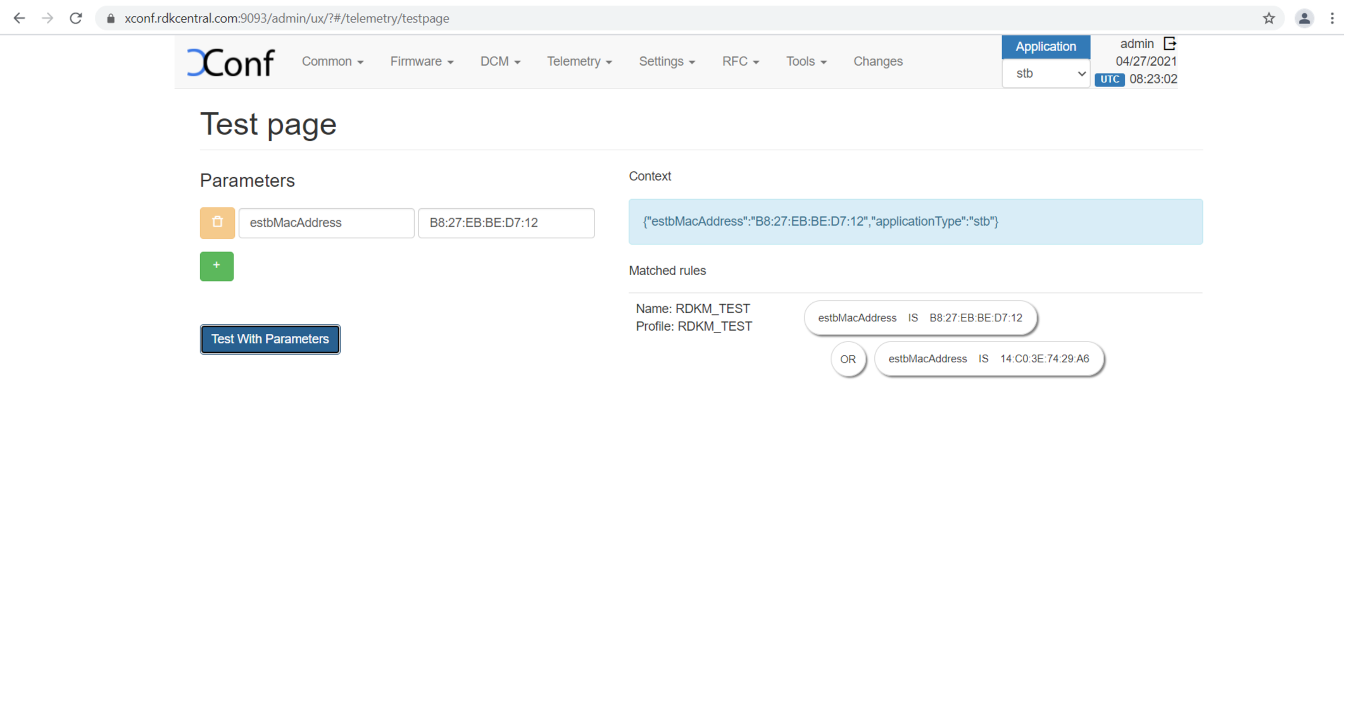
Bound profile: RDKM_TEST

Verification

a. Verification of telemetry test page.

After creating the permanent profile and targeting rules, go to Telemetry->Test page and give a parameter that will match the one of the rule that you have created. Then matched rule will be displayed like below.

Site Navigation: <xconf-server>:19093 >> Telemetry >> Test Page



xconf.rdkcentral.com:9093/admin/ux/?#/telemetry/testpage

Test page

Parameters

estbMacAddress B8:27:EB:BE:D7:12

Context

{"estbMacAddress":"B8:27:EB:BE:D7:12","applicationType":"stb"}

Matched rules

Name: RDKM_TEST
Profile: RDKM_TEST

estbMacAddress IS B8:27:EB:BE:D7:12

OR estbMacAddress IS 14:C0:3E:74:29:A6

b. Verification via curl command

The curl command mocks the request being sent from an STB like below and sample response is also given. It can be given as a curl command or as a get request via postman or browser. The same url used for logupload verification can be used here too, the response will have telemetry settings data like below (urn:settings:TelemetryProfile)

The new API for Telemetry is getT2Settings. It will take the same parameters as the current API, /loguploader/getSettings.

If the component name has been defined for an entry, the response will be in the new format. The second and third columns for that entry will not be used in the response. The content field comes from the fifth column (component name). The type field will be a constant string "<event>".

Example for getT2Settings:

{"header":"MEDIA_ERROR_NETWORK_ERROR","content":"com.cisco.spvtg.ccsp.meshagent","type":"<event>","pollingFrequency":"0"}

If the component name has not been defined for an entry, the response will be in the current format.

Example for getSettings:

{"header":"MEDIA_ERROR_NETWORK_ERROR","content":"onMediaError NETWORK ERROR(10)","type":"receiver.log","pollingFrequency":"0"}

eg :

$ curl 'http://<XCONF_IP>: 19092/loguploader/getSettings?estbMacAddress=B8:27:EB:BE:D7:12’

Sample Response :

```
{
    "urn:settings:GroupName": "RDKM_TEST",
    "urn:settings:CheckOnReboot": true,
    "urn:settings:CheckSchedule:cron": "2 1 2 1 1",
    "urn:settings:CheckSchedule:DurationMinutes": 0,
    "urn:settings:LogUploadSettings:Message": null,
    "urn:settings:LogUploadSettings:Name": "RDKM_TEST",
    "urn:settings:LogUploadSettings:NumberOfDays": 1,
    "urn:settings:LogUploadSettings:UploadRepositoryName": "RDKM_TEST",
    "urn:settings:LogUploadSettings:RepositoryURL": "http://{loguoloadserver}/xconf/logupload.php",
    "urn:settings:LogUploadSettings:UploadOnReboot": true,
    "urn:settings:LogUploadSettings:UploadImmediately": false,
    "urn:settings:LogUploadSettings:upload": true,
    "urn:settings:LogUploadSettings:UploadSchedule:cron": "2 1 1 1 1",
    "urn:settings:LogUploadSettings:UploadSchedule:levelone:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:leveltwo:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:levelthree:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:DurationMinutes": 0,
    "urn:settings:VODSettings:Name": null,
    "urn:settings:VODSettings:LocationsURL": null,
    "urn:settings:VODSettings:SRMIPList": null,
    "urn:settings:TelemetryProfile": {
        "id": "69e37757-b463-47aa-94a8-2ce438e26a50",
        "telemetryProfile": [
            {
                "header": "Firewall",
                "content": "starting firewall service",
                "type": "FirewallDebug.txt",
                "pollingFrequency": "1"
            },
            {
                "header": "MEDIA_ERROR_NETWORK_ERROR",
                "content": "onMediaError NETWORK ERROR(10)",
                "type": "receiver.log",
                "pollingFrequency": "1"
            }
        ],
        "schedule": "3",
        "expires": 0,
        "telemetryProfile:name": "RDKM_TEST",
        "uploadRepository:URL": "http://{logupload-server}/xconf/logupload.php",
        "uploadRepository:uploadProtocol": "HTTP"
    }
}
```

eg :

$ curl 'http://<XCONF_IP>: 19092/loguploader/getT2Settings?estbMacAddress=B8:27:EB:BE:D7:12'

Sample Response :

```
{
    "urn:settings:GroupName": "RDKM_TEST",
    "urn:settings:CheckOnReboot": true,
    "urn:settings:CheckSchedule:cron": "2 1 2 1 1",
    "urn:settings:CheckSchedule:DurationMinutes": 0,
    "urn:settings:LogUploadSettings:Message": null,
    "urn:settings:LogUploadSettings:Name": "RDKM_TEST",
    "urn:settings:LogUploadSettings:NumberOfDays": 1,
    "urn:settings:LogUploadSettings:UploadRepositoryName": "RDKM_TEST",
    "urn:settings:LogUploadSettings:RepositoryURL": "http://{log-upload-server}/xconf/logupload.php",
    "urn:settings:LogUploadSettings:UploadOnReboot": true,
    "urn:settings:LogUploadSettings:UploadImmediately": false,
    "urn:settings:LogUploadSettings:upload": true,
    "urn:settings:LogUploadSettings:UploadSchedule:cron": "2 1 1 1 1",
    "urn:settings:LogUploadSettings:UploadSchedule:levelone:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:leveltwo:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:levelthree:cron": null,
    "urn:settings:LogUploadSettings:UploadSchedule:DurationMinutes": 0,
    "urn:settings:VODSettings:Name": null,
    "urn:settings:VODSettings:LocationsURL": null,
    "urn:settings:VODSettings:SRMIPList": null,
    "urn:settings:TelemetryProfile": {
        "id": "69e37757-b463-47aa-94a8-2ce438e26a50",
        "telemetryProfile": [
            {
                "header": "Firewall",
                "content": "starting firewall service",
                "type": "FirewallDebug.txt",
                "pollingFrequency": "1"
            },
            {
                "header": "MEDIA_ERROR_NETWORK_ERROR",
                "content": "com.cisco.spvtg.ccsp.meshagent",
                "type": "<event>",
                "pollingFrequency": "1"
            }
        ],
        "schedule": "3",
        "expires": 0,
        "telemetryProfile:name": "RDKM_TEST",
        "uploadRepository:URL": "http://log-upload-server}/xconf/logupload.php",
        "uploadRepository:uploadProtocol": "HTTP"
    }
}
```

Client end verification (RPI)

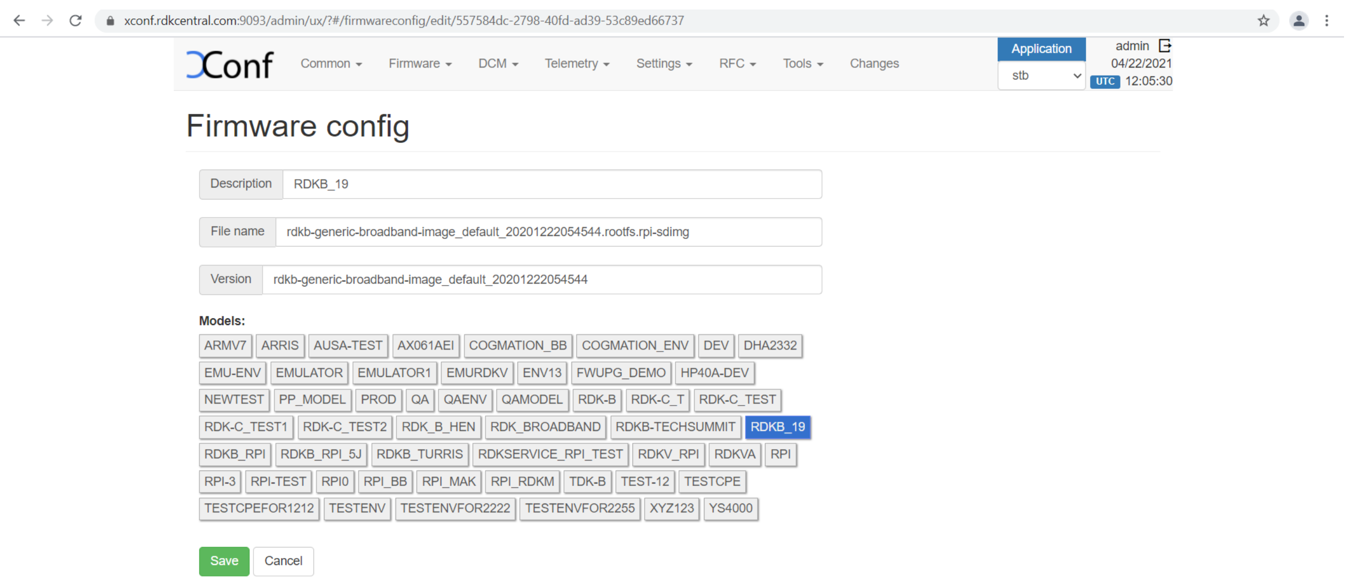
CURL Command	curl 'http://<XCONF_IP>:19092/loguploader/getSettings?estbMacAddress=B8:27:EB:FF:54:95&firmwareVersion=rdk-generic-hybrid-wpe-image_default_20190702100618&env=dev&model=RPI&ecmMacAddress=B8:27:EB:FF:54:95&controllerId=2504&channelMapId=2345&vodId=15660&timezone=&partnerId=&accountId=Unknown&version=2'
CPE Script (RDK-V)	/lib/rdk/DCMscript.sh /lib/rdk/dca_utility.sh
CPE Service (RDK-V)	/lib/systemd/system/dcm-log.service

Feature Validation (Firmware update)

Configuration

1. Firmware config can be created via Firmware - > Firmware Configs -> Create. Enter a description for this config. Also we can define the file name and version of the image/firmware that need to be downloaded to the CPE device. The models that we defined in Common Models section will be available here, We can select the required models by clicking on it.

Site Navigation: http://<XCONF_SERVER>:19093 >> Firmware >> Firmware config



Firmware config

Description: RDKB_19

File name: rdkb-generic-broadband-image_default_20201222054544.rootfs.rpi-sdimg

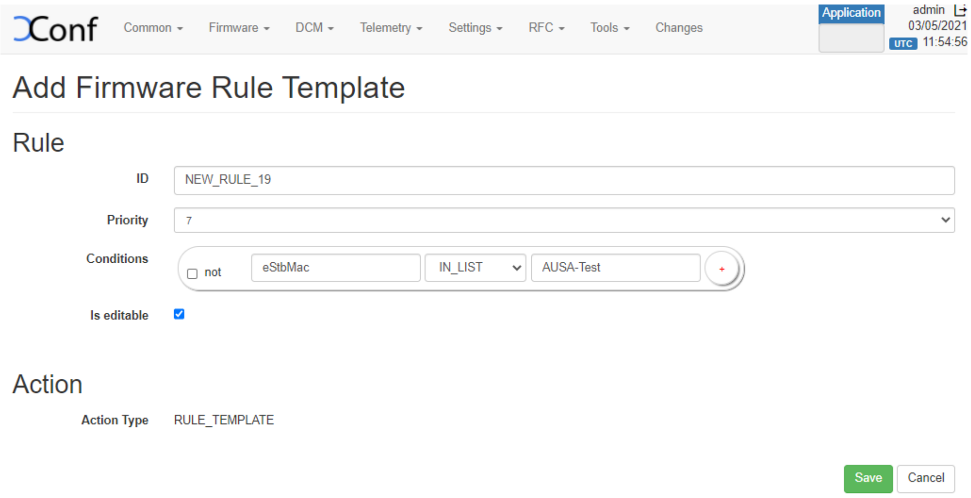
Version: rdkb-generic-broadband-image_default_20201222054544

Models:

ARMV7 | ARRIS | AUSA-TEST | AX061AEI | COGMATION_BB | COGMATION_ENV | DEV | DHA2332 | EMU-ENV | EMULATOR | EMULATOR1 | EMURDKV | ENV13 | FWUPG_DEMO | HP40A-DEV | NEWTEST | PP_MODEL | PROD | QA | QAENV | QAMODEL | RDK-B | RDK-C_T | RDK-C_TEST | RDK-C_TEST1 | RDK-C_TEST2 | RDK_B_HEN | RDK_BROADBAND | RDKB-TECHSUMMIT | RDKB_19 | RDKB_RPI | RDKB_RPI_5J | RDKB_TURRIS | RDKSERVICE_RPI_TEST | RDKV_RPI | RDKVA | RPI | RPI-3 | RPI-TEST | RPI0 | RPI_BB | RPI_MAK | RPI_RDKM | TDK-B | TEST-12 | TESTCPE | TESTCPEFOR1212 | TESTENV | TESTENVFOR2222 | TESTENVFOR2255 | XYZ123 | YS4000

2. To create a firmware template, go to Firmware Firmware templates. Enter the ID name. Select priority from the 'Priority' drop down menu. Add conditions. There are some already existing templates, if you are using the existing Firmware Templates for configuration ,you can skip this step.

Site Navigation: http://<XCONF_SERVER>:19093 >> Firmware >>Firmware Templates



Add Firmware Rule Template

Rule

ID: NEW_RULE_19

Priority: 7

Conditions: ☐ not | eStbMac | IN_LIST | AUSA-Test

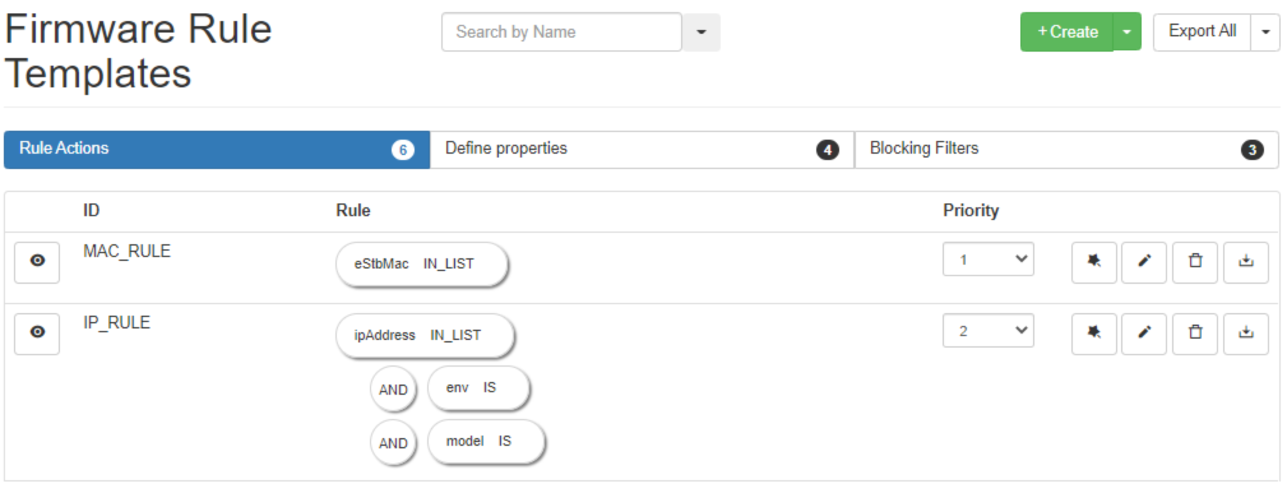
Is editable: ☑

Action

Action Type: RULE_TEMPLATE

3. Firmware rule can be create via Firmware -> Firmware rules -> Rule Action -> Create. On clicking on Create button, a list of templates will be presented. We can select the required template (There will be default templates like ENV_MODEL_RULE, IP_RULE, MAC_RULE etc. and also the custom templates created from Firmware->Firmware template -> Create).

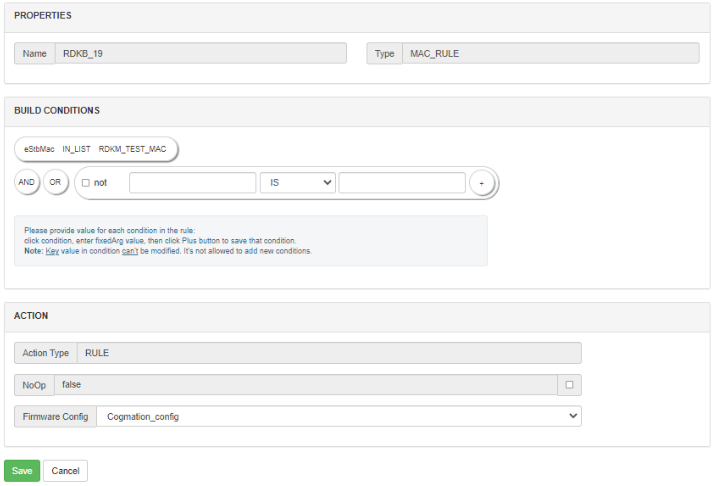
Site Navigation: http://<XCONF_SERVER>:19093 >> Firmware >> Firmware Rules



Firmware Rule Templates

ID	Rule	Priority
MAC_RULE	eStbMac IN_LIST	1
IP_RULE	ipAddress IN_LIST AND env IS AND model IS	2

After we select the required template, 'Add firmware Rule' page will be displayed. Here the build conditions will be present from the 'template' that we added and in addition to that we can add additional Build Conditions also. To add firmware config , go to the 'Actions' tab and select the firmware config from 'Firmware config' drop down list (Select the firmware config that you have created).



PROPERTIES

Name: RDKB_19 Type: MAC_RULE

BUILD CONDITIONS

eStbMac IN_LIST RDKM_TEST_MAC

Please provide value for each condition in the rule:
click condition, enter fixedArg value, then click Plus button to save that condition.
Note: Key value in condition can't be modified. It's not allowed to add new conditions.

ACTION

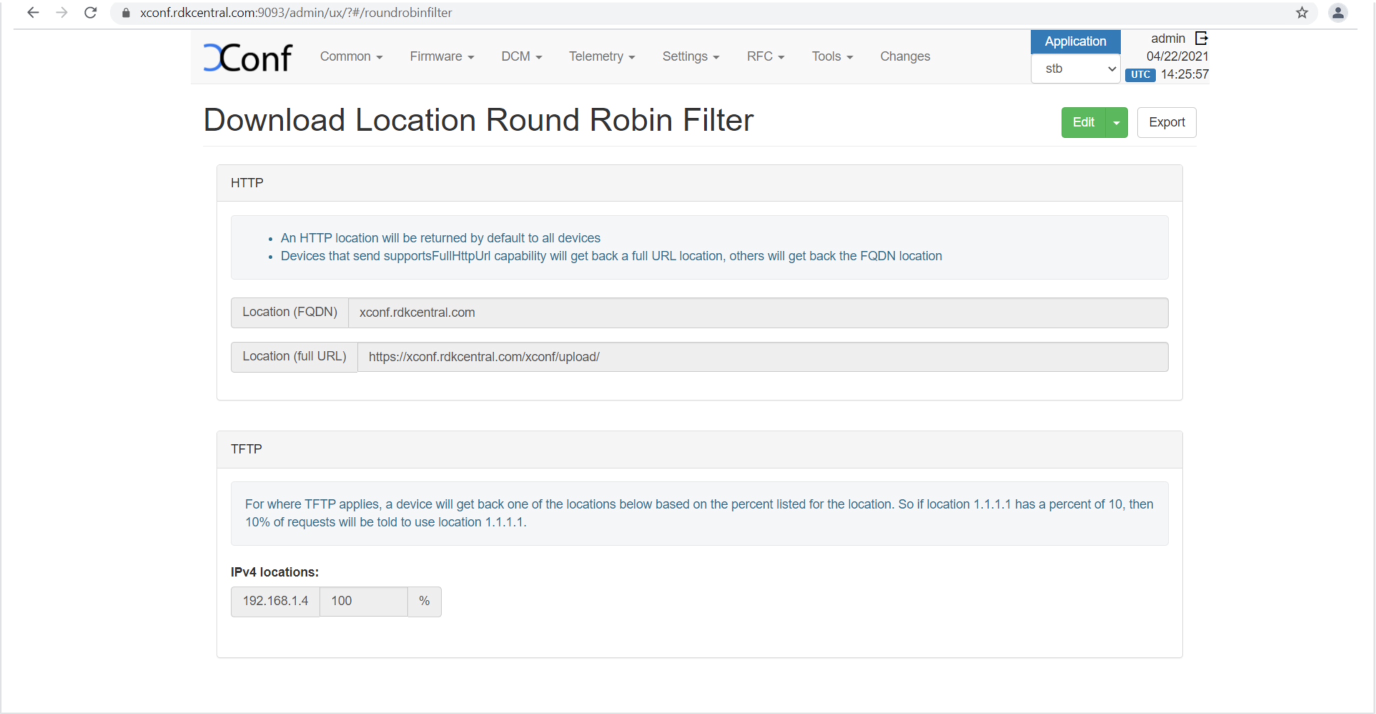
Action Type: RULE

NoOp: false □

Firmware Config: Cogmation_config

4. The download location needs to be specified so that it can be returned in the response. Choose Firmware -> Download location filter-> Edit, where we can specify the location from where we can download the firmware. Enter the FQDN and Full http location for the firmware download server . Http location will be returned by default to all devices.

Site Navigation: http://<XCONF_SERVER>:19093 >> Firmware >> Download Location Round Robin Filter



Download Location Round Robin Filter

HTTP

- An HTTP location will be returned by default to all devices
- Devices that send supportsFullHttpUrl capability will get back a full URL location, others will get back the FQDN location

Location (FQDN) xconf.rdkcentral.com

Location (full URL) https://xconf.rdkcentral.com/xconf/upload/

TFTP

For where TFTP applies, a device will get back one of the locations below based on the percent listed for the location. So if location 1.1.1.1 has a percent of 10, then 10% of requests will be told to use location 1.1.1.1.

IPv4 locations:

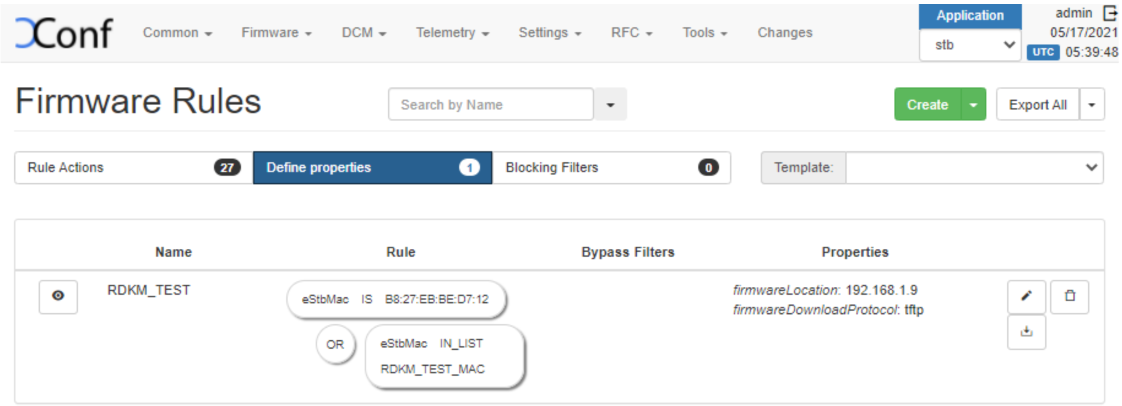
192.168.1.4 100 %

Add TFTP location

An HTTP location will be returned by default to all devices. To enable tftp(If you have the download location of the firmware as tftp, then only you need to setup this)as download location for a particular set of devices, we need to override it from firmware rules.

1. Go to Firmware -> Firmware Rules -> Define Properties -> Create

Site Navigation: <xconf-server>:19093 >>Firmware >> Firmware Rules >> Define Properties

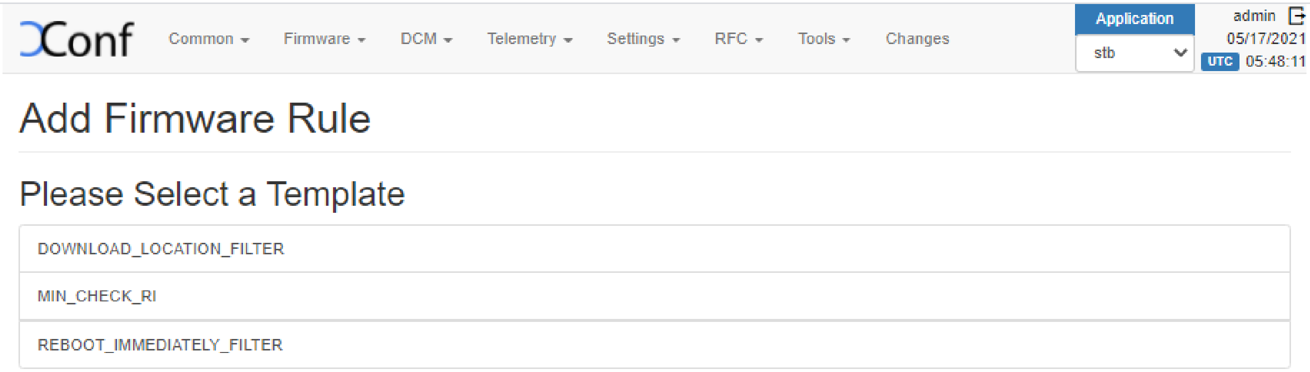


Firmware Rules

Name	Rule	Bypass Filters	Properties
RDKM_TEST	eStbMac IS B8:27:EB:BE:D7:12 OR eStbMac IN_LIST RDKM_TEST_MAC		*firmwareLocation*: 192.168.1.9 *firmwareDownloadProtocol*: tftp

2. A page will be displayed with options to select the template. Select ‘DOWNLOAD_LOCATION_FILTER’ from the list

Site Navigation: <xconf-server>:19093 >>Firmware >> Firmware Rules >> DefineProperties



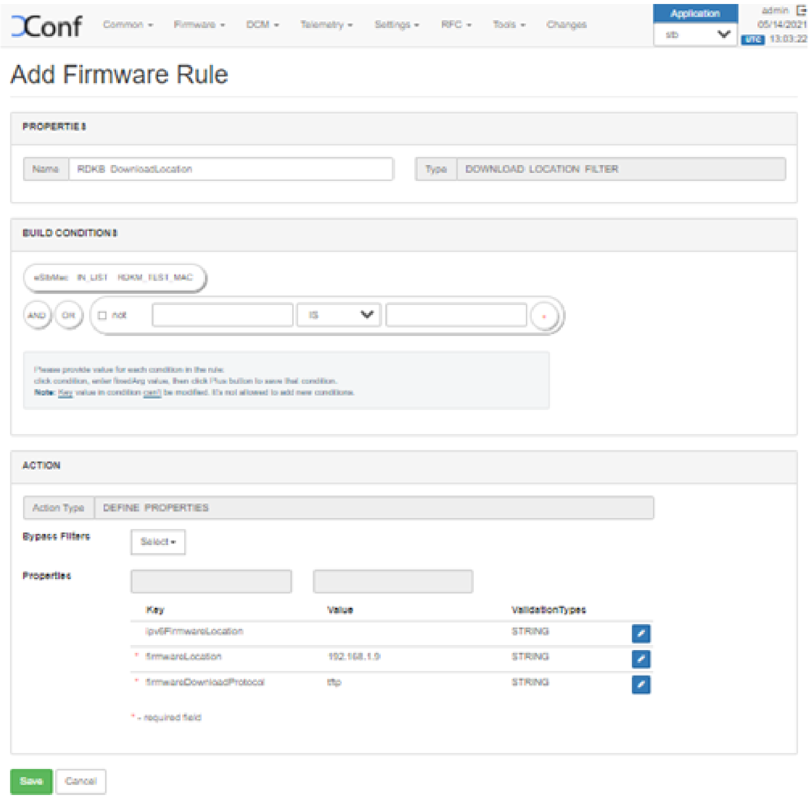
Add Firmware Rule

Please Select a Template

- DOWNLOAD_LOCATION_FILTER
- MIN_CHECK_RI
- REBOOT_IMMEDIATELY_FILTER

3. In this 'firmware rule' page with type 'DOWNLOAD_LOCATION_FILTER', we need to specify the ‘Build Conditions’ and ‘Action’ , The build condition should be same as that we used to set rule actions(which will set rules for our intended devices). In ‘Properties’ option under ‘Action’, add ‘firmwareDownloadProtocol’ as ‘tftp’, ‘firmwareLocation’ as ‘your tftp location IPV4 address’. This property will override the default value set from ‘Download Round Robin location filter’.

Site Navigation: <xconf-server>:19093 >>Firmware >> Firmware Rules >> DefineProperties



Add Firmware Rule

PROPERTIES

Name: RDKB_DownloadLocation
Type: DOWNLOAD_LOCATION_FILTER

BUILD CONDITIONS

eStbMac IN_LIST RDKM_TEST_MAC

Please provide value for each condition in the rule.
click condition, enter fixedArg value, then click Plus button to save that condition.
Note: Key value in condition can't be modified. It's not allowed to add new conditions.

ACTION

Action Type: DEFINE_PROPERTIES

Bypass Filters: Select

Properties

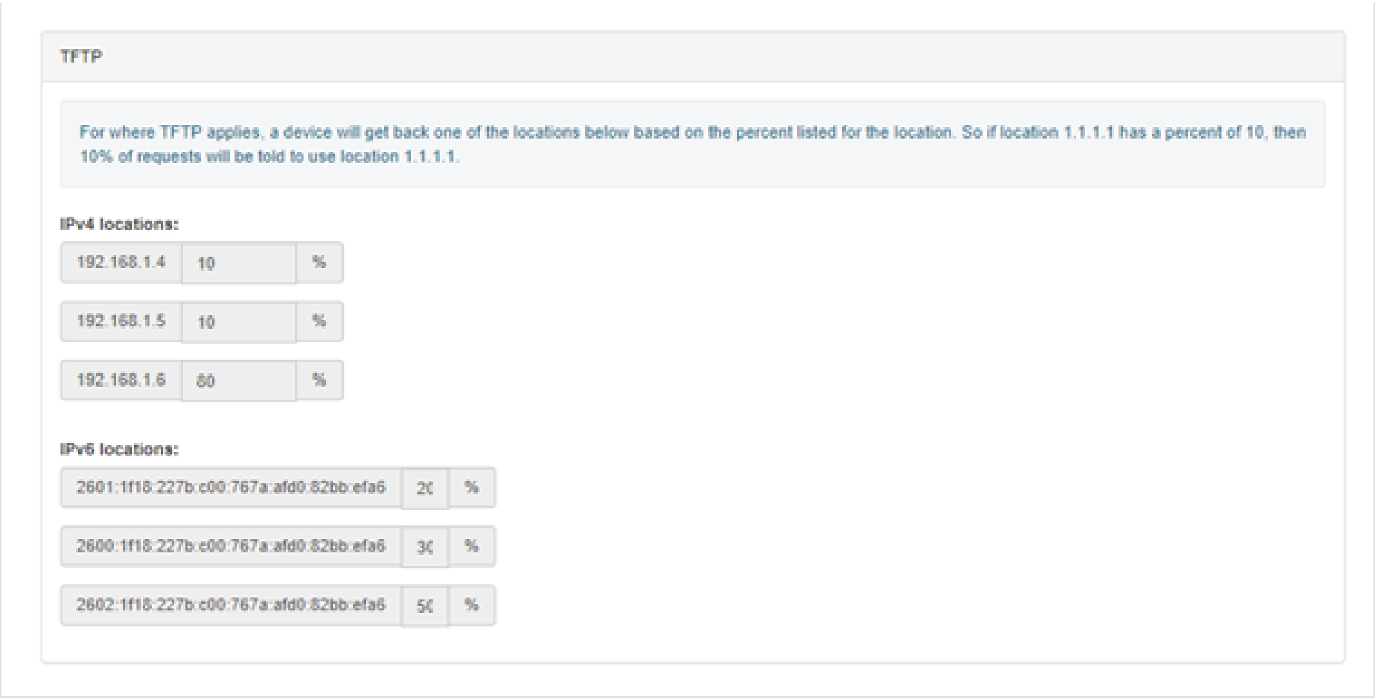
	Key	Value	ValidationTypes
	ipv6FirmwareLocation		STRING
*	firmwareLocation	192.168.1.9	STRING
*	firmwareDownloadProtocol	tftp	STRING

* - required field

4. To add IPV6 address of tftp servers, you can either specify it here in the define properties rule or else from ‘Download Round Robin Filter’ page. To add IPV6, add it as ‘ipv6FirmwareLocation’ Property in ‘define properties rule’ page (ipv6FirmwareLocation key will be there by default, you need to add the value as tftp ipv6 address).

5. To add IPV6 address of tftp servers in ‘Download Round Robin Filter’, Go to Firmware - > Download Location Filter. Enter the tftp IPV6 locations and also the percentages. The devices will get back one of the locations based on the percentage listed for the location.

Site Navigation: <xconf-server>:19093 >>Firmware >> Download Location Filter



TFTP

For where TFTP applies, a device will get back one of the locations below based on the percent listed for the location. So if location 1.1.1.1 has a percent of 10, then 10% of requests will be told to use location 1.1.1.1.

IPv4 locations:

192.168.1.4	10	%
192.168.1.5	10	%
192.168.1.6	80	%

IPv6 locations:

2601:1f18:227b:c00:767a:afd0:82bb:efa6	2(	%
2600:1f18:227b:c00:767a:afd0:82bb:efa6	3(	%
2602:1f18:227b:c00:767a:afd0:82bb:efa6	5(	%

6. Response example for https://{xconf-ip}:{port}/xconf/swu/stb?eStbMac={mac}. Here the 'firmwareLocation' and 'firmwareDownloadProtocol' are overrided at 'Define Properties' firmware rule.
The 'ipv6' addresses will be one of the addresses mentioned in the 'Download Filter' page . If you don't want "ipv6FirmwareLocation", then don't setup it in 'Define Properties' or in the 'Download location filter' pages and you will get only "firmwareLocation" in the response

```
{

  "firmwareDownloadProtocol": "tftp",

  "firmwareFilename": "rdkb-generic-broadband-image_default_20200406103506.rootfs.rpi-sdimg",

  "firmwareLocation": "192.168.1.9",

  "firmwareVersion": "rdkb-generic-broadband-image_default_20200406103506.txt",

  "ipv6FirmwareLocation": "2601:1f18:227b:c00:767a:afd0:82bb:efa6",

  "rebootImmediately": false

}
```

7. Setting up IPV4 locations via ‘Download Location Round Robin Filter’ is not supported. This can be set only by the property 'firmwareLocation' from the 'Define Properties' firmware rule page.

8. **Note** : Just like we added tftp location and protocol here, we can also override the default value with http as well. For firmwareDownloadProtocol, add 'http' and for the 'firmwareLocation', add http location

Override the default download Location set at Download Location Filter

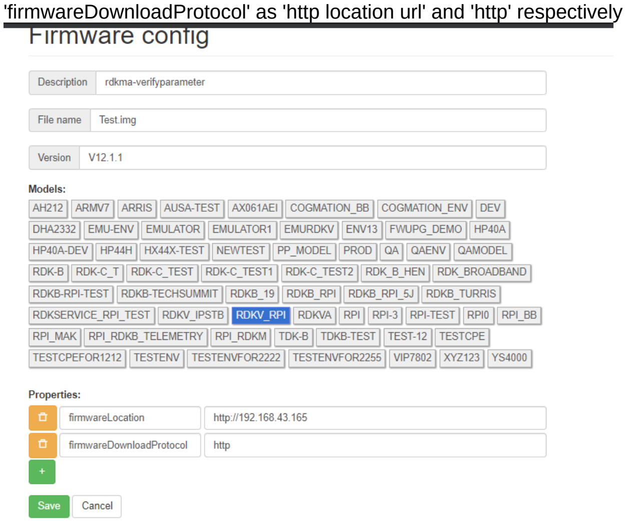
There is a new option added in the Firmware Config, where we can add parameters. For example if we add parameters 'firmwareLocation' and 'firmwareDownloadProtocol'. then we will be able to override the default download location set from the 'DownLoad Location Filter' page.

Override Firmware Location with http

To create a new firmware configuration for a particular set of devices with http download location :

1. Go to Firmware Firmware Configs Create. Enter a description for this config. Also we can define the file name and version of the image /firmware that need to be downloaded to the CPE device. The models that we defined in Common Models section will be available here, We can

select the required models by clicking on it. There is also an option 'Parameters'. Add the key values 'firmwareLocation' and 'firmwareDownloadProtocol' as 'http location url' and 'http' respectively



Firmware config

Description: rdkma-verifyparameter

File name: Test.img

Version: V12.1.1

Models:

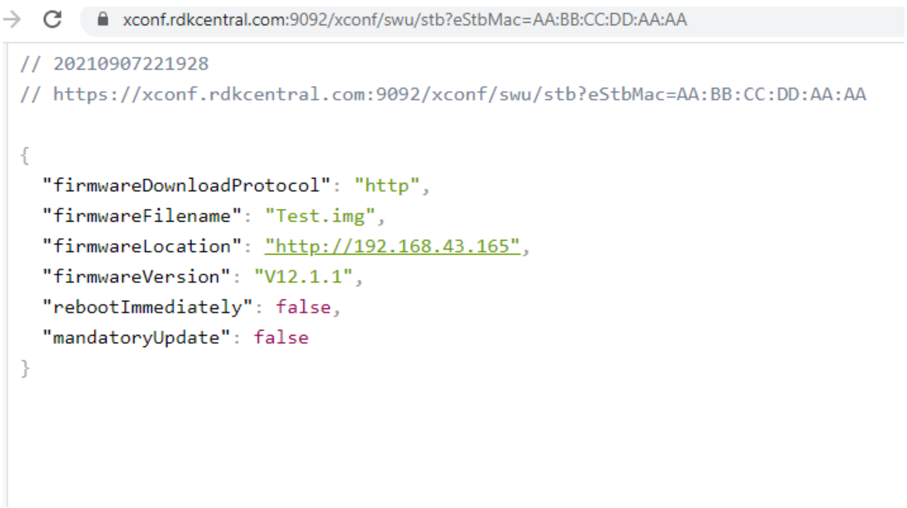
AH212 ARMV7 ARRIS AUSA-TEST AX061AEI COGMATION_BB COGMATION_ENV DEV DHA2332 EMU-ENV EMULATOR EMULATOR1 EMURDKV ENV13 FWUPG_DEMO HP40A HP40A-DEV HP44H HX44X-TEST NEWTEST PP_MODEL PROD QA QAENV QAMODEL RDK-B RDK-C_T RDK-C_TEST RDK-C_TEST1 RDK-C_TEST2 RDK_B_HEN RDK_BROADBAND RDKB-RPI-TEST RDKB-TECHSUMMIT RDKB_19 RDKB_RPI RDKB_RPI_5J RDKB_TURRIS RDKSERVICE_RPI_TEST RDKV_IPSTB RDKV_RPI RDKVA RPI RPI-3 RPI-TEST RPI0 RPI_BB RPI_MAK RPI_RDKB_TELEMETRY RPI_RDKM TDK-B TDKB-TEST TEST-12 TESTCPE TESTCPEFOR1212 TESTENV TESTENVFOR2222 TESTENVFOR2255 VIP7802 XYZ123 YS4000

Properties:

firmwareLocation	http://192.168.43.165
firmwareDownloadProtocol	http

2. Create a firmware rule like the steps given in 'Configuration' and map this Firmware Config to it. Check using the steps in below 'Verification' sections and verify if the firmwareLocation and firmwareDownloadProtocol are the same as we configured in FirmwareConfig page.

   Sample curl response(refer below steps to check)



xconf.rdkcentral.com:9092/xconf/swu/stb?eStbMac=AA:BB:CC:DD:AA:AA

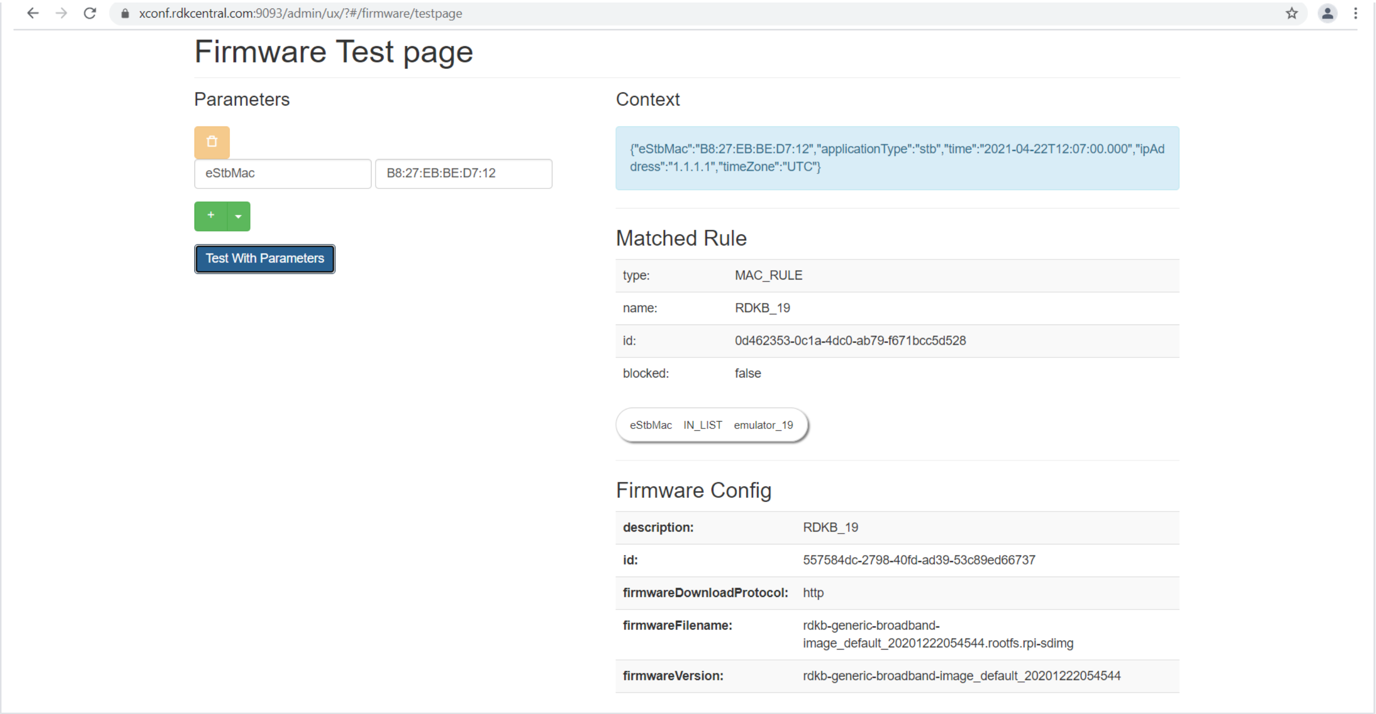
```
// 20210907221928
// https://xconf.rdkcentral.com:9092/xconf/swu/stb?eStbMac=AA:BB:CC:DD:AA:AA

{
  "firmwareDownloadProtocol": "http",
  "firmwareFilename": "Test.img",
  "firmwareLocation": "http://192.168.43.165",
  "firmwareVersion": "V12.1.1",
  "rebootImmediately": false,
  "mandatoryUpdate": false
}
```

Verification

a. Verification of Firmware test page.

After creating the Firmware configs and Firmware rules , go to Firmware->Test page and give a parameter that will match the one of the rule that you have created. Then matched rule will be displayed like below.

Site Navigation: <xconf-server>:19093 >>Firmware >> Test Page



Firmware Test page

Parameters

eStbMac: B8:27:EB:BE:D7:12

Context

{"eStbMac":"B8:27:EB:BE:D7:12","applicationType":"stb","time":"2021-04-22T12:07:00.000","ipAddress":"1.1.1.1","timeZone":"UTC"}

Matched Rule

type:	MAC_RULE
name:	RDKB_19
id:	0d462353-0c1a-4dc0-ab79-f671bcc5d528
blocked:	false

eStbMac IN_LIST emulator_19

Firmware Config

description:	RDKB_19
id:	557584dc-2798-40fd-ad39-53c89ed66737
firmwareDownloadProtocol:	http
firmwareFilename:	rdkb-generic-broadband-image_default_20201222054544.rootfs.rpi-sdimg
firmwareVersion:	rdkb-generic-broadband-image_default_20201222054544

b. Verification via curl command

The curl command mocks the request being sent from an STB like below and sample response is also given. It can be given as a curl command or as a get request via postman or browser.

eg :

$ curl 'https://<XCONF_IP>:19092/xconf/swu/stb?eStbMac=B8:27:EB:BE:D7:12

'

Sample Response :

```
 {
  "firmwareDownloadProtocol": "http",
  "firmwareFilename": "vip7802_FBT_rdk-next_20210610095056.pkg.tar.gz",
  "firmwareLocation": "xconf.rdkcentral.com",
  "firmwareVersion": "vip7802_FBT_rdk-next_20210610095056",
  "rebootImmediately": false,
  "mandatoryUpdate": false
}
```

Client end verification (RPI)

CURL Command	curl 'http://<XCONF_IP>:19092/xconf/swu/stb?eStbMac=B8:27:EB:BE:D7:12&model=ARMv7&capabilities=RCDL&capabilities=supportsFullHttpUrl'
CPE Script (RDK-V)	/lib/rdk/swupdate_utility.sh
CPE Service (RDK-V)	/lib/systemd/system/swupdate.service

Percent Filter

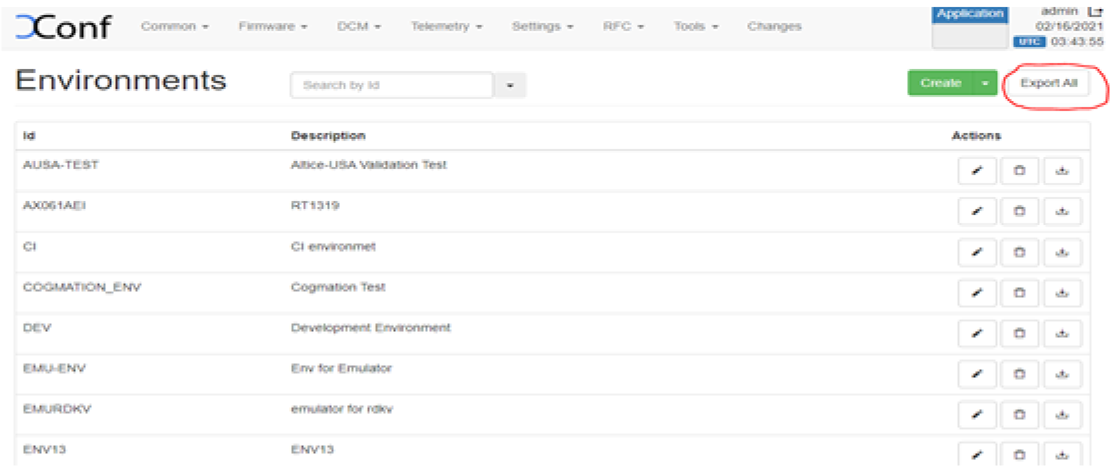
Percentage based filters allow us to block a certain percentage of Xconf responses that would otherwise have resulted in a change in firmware. The use case for this is when we have tons of STBs out there and we don't yet have scheduled downloads. We would like to be able to only service a certain percentage as a throttling mechanism so download servers aren't overwhelmed.

Import and Export Feature

We can import and export all the configuration data from the UI itself. This feature can be primarily used for transferring the configuration data from one xconfserver setup to the other one.

The export and import data need to be done separately for the Application - stb, xhome and rdkcloud for all the pages except those in Common tab.

To export data from a page, Click on ‘Export All’ button in that page. The configuration data will be downloaded as a JSON file.

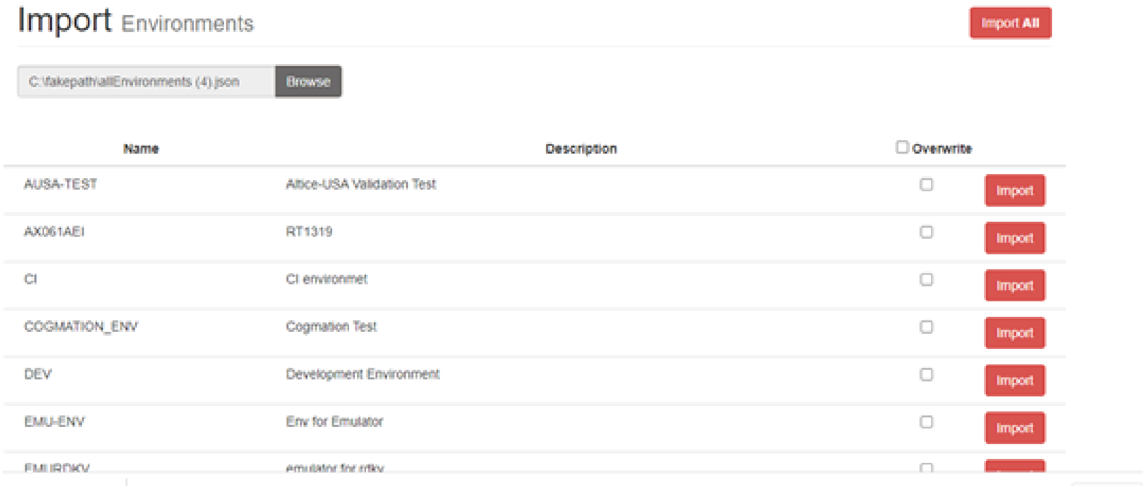


Environments

Id	Description	Actions
AUSA-TEST	Altice-USA Validation Test	
AX061AEI	RT1319	
CI	CI environmet	
COGMATION_ENV	Cogmation Test	
DEV	Development Environment	
EMU-ENV	Env for Emulator	
EMURDKV	emulator for rdkv	
ENV13	ENV13	

To import data :

1. Open the dropdown menu next to ‘Create’ button.
2. From the drop down menu, click on ‘Import’.
3. A new page will be displayed with option to browse the location of the JSONfile to be imported. Select the file that need to be imported.
4. All the data from the file will be listed in the page.
5. Click on ‘Import All’ to import all data



Import Environments

Name	Description	□ Overwrite	
AUSA-TEST	Altice-USA Validation Test	□	Import
AX061AEI	RT1319	□	Import
CI	CI environmet	□	Import
COGMATION_ENV	Cogmation Test	□	Import
DEV	Development Environment	□	Import
EMU-ENV	Env for Emulator	□	Import
EMURDKV	emulator for rdkv	□	

FAQs and Common Issues faced in setup

1. The 'mvn clean install' step is stuck at 'org.hyperic.sigar.SigarException: no libsigar-amd64-linux.so in java.library.path' . Is this an issue?

   This is an exception from unit test , but the tests will run successfully. Build process may stall for some time but the build will be successful and the application can be launched successfully. So it can be ignored as well. Or you can run the mvn build step - 'mvn clean install -DskipTests=true', instead of the 'mvn clean install'.

2. In Xconf server, what is the 'Environment' tab for? which module will reference it?

This can be added in the build conditions just like you add maclists or IPlists.

3. I am getting a 503 error on accessing the dataservice. What may be the reason?

Check whether Cassandra DB is up or not. If Cassandra DB is not up, then it may affect the admin UI as well. The admin UI may be up, but you may not be able to add data to the Application.

4. We are getting 'Failed to execute goal com.github.eirslett:frontend-maven-plugin:1.10.0:npm (Compile via NPM install) on project xconf-angular-admin: Failed to run task: 'npm install' failed.'. What will be the reason?

For running xconf-angular admin, frontend-maven-plugin is used and it will internally install node and npm. Some dependencies may not be getting installed due to network restrictions. Run the build command 'mvn clean install' using -e switch and you will get the full error trace. Rectify the network issue and continue

5. Is there a requirement for a GUI for deployment?

No, there is no requirement for GUI based deployment environment

6. Python 2.7 is mentioned here, Can we use any other Python versions?

At the time of reference setup, this python version worked with the Cassandra version we used. For the reference setup we used cassandra 3.11.9 and python 2.7 . For Cassandra 3X, python 2.7 is required, https://community.datastax.com/questions/11213/py3-support-for-cassandra-3116.html.

7. Can openjdk v11 be used?

No, Like mentioned in the userguide app requires java 8 and you can refer the prerequisites here in readme https://github.com/rdkcentral/xconfserver#readme

8. Local Firewall is required. Are there any known issues with this?

Firewall issue is not mentioned in the user guide. However there were issues in the past where partner would deploy the application in their VMs and due to the firewall , they may face issues to access it. This need to be resolved internally.

9. We are getting this issue on running Cassandra - "Java HotSpot (TM) 64-Bit Server VM warning: Cannot open file. /..logs/gc.log due to No such file or directory"

This may be an issue related to the memory. This issue has happened in the past when community members try to setup the applications in Virtual env like Oracle VM virtual box. Increase the allocated memory for the virtual machine and it will be resolved