Tivoli Netcool Support's Guide to the Generic 3GPP Probe

by

Jim Hutchinson

Document release: 2.1



Table of Contents

1Introduction....2

1.1Overview....2
1.2Probe Connections....3
1.3The Generic 3GPP Probes life cycle....3
1.4Main probe properties....4
1.4.1Connecting to the CORBA EMS....4
1.4.2Setting the version of 3GPP CORBA....5
1.4.3Character encoding....7
1.4.4AlarmIRPName and NotificationIRPName....8
1.4.5ORBLocalHost and ORBLocalPort....9
1.4.6TmeTick and Inactivity....9

2The CORBA Framework....10

3Ericsson OSS-RC EMS....11

3.1The Ericsson OSS-RC Probe....11
3.2The Generic 3GPP probe....11
3.3Property file....12
3.4jacorb.properties....13
3.5Alarm and Notification IRPs....13
3.5.1Naming Service....13
3.6Migrating from the Generic 3GPP v320 probe....14
3.6.1Rules file token mapping....14

4Troubleshooting....15

4.1Debug logging....15
4.2Example Information MessageLevel logging....16
4.3Incorrect Release3GPP setting....17
4.4Incorrect tokenisation....17
4.5Incorrect AlarmIRP....18
4.6Incorrect NotificationIRP....19
4.7Checking the ORBLocalPort....20
4.8ORBLocalHost not accessible from EMS....21

IBM Copyright 2021

1 Introduction

1.1 Overview

The Generic 3GPP probe is the main probe integration for connecting to CORBA servers that support Generic 3GPP CORBA.

The two main Generic 3GPP CORBA probes are:

- Generic 3GPP probe
- Generic TMF814 probe

The two main vendor specific 3GPP CORBA probes are:

- Huawei U2000 3GPP CORBA Probe - nco_p_huawei_u2000_3gpp_corba
- Nokia NetAct 3GPP CORBA Probe – nco_p_nokia_netact_3gpp_v6

The Generic 3GPP CORBA probe uses the CORBA Framework module, which is based on JacORB which is the free Java implementation of the OMG's CORBA standard. The CORBA Framework includes support for TMF814 CORBA.

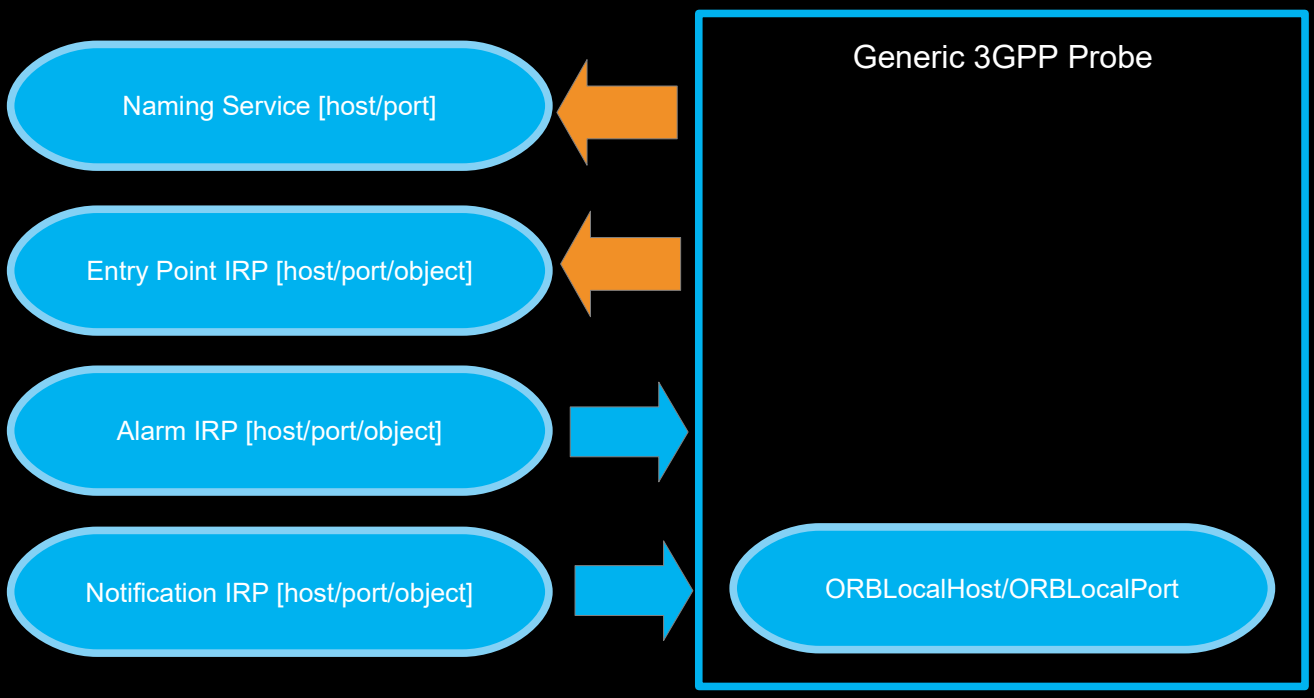
The Generic 3GPP CORBA probe uses the following supporting packages:

- probe-corba-framework
- probe-sdk-java
- probe-nonnative-base

IBM Copyright 2021

1.2 Probe Connections

The Key Generic 3GPP probe connections



1.3 The Generic 3GPP Probes life cycle

- Start-up - Initialise probe instance and property settings
- Connection - Connect to Entry Point IRP / Naming Service, retrieve Alarm IRP and Notification IRP reference and versions
- Subscription - Subscribe to alarm notification on Notification IRP using attach_push() call
  - The probe stays idle and waits for new alarms (notifications) as available on the subscribed in Notification IRP
- Resynchronization - Retrieve resynchronization alarms on Alarm IRP using get_alarm_list() and next_alarmInformations() for alarm in batches
- Disconnection - Disconnect from Notification IRP using detach() call.
- Shutdown – close all running threads

IBM Copyright 2021

1.4 Main probe properties

1.4.1 Connecting to the CORBA EMS

There are three methods that can be used to connect to the CORBA EMS

Method 1: Entry Point IOR string and object names

The CORBA EMS Administrator needs to provide the correct settings for the following properties:

- EntryPointIORFile – The Entry Point IOR String
- AlarmIRPName – The exact name of the Alarm IRP the probe needs to connect to
- NotificationIRPName – The exact name of the Notification IRP the probe needs to connect to

Method 2: Object IOR strings

The CORBA EMS Administrator needs to provide the correct settings for the following properties:

- AlarmIRPIORFile – The Alarm IRP IOR string
- NotificationIORFile – The Notification IRP IOR string

Method 3: Naming Service Host and port

- NamingServiceHost – The FQDN or IP Address of the Naming Service host
- NamingServicePort - Naming Service port

OR

- NamingServiceIorFile – The Naming Service IOR string

AND

- AlarmIRPName - The exact name of the Alarm IRP the probe needs to connect to
- NotificationIRPName - The exact name of the Notification IRP the probe needs to connect to

For method 1 and 2, the EMS Administrator has to provide all of the details. This is because the Entry Point service is a secure connection method, so the details of the objects available in the CORBA Server are hidden from external sources.

Method 3 allows the CORBA server to be interrogated, using the dumpns script, provided the Naming Service host and port are known. If only the Naming Service IOR is provided, it is possible to determine the Naming Service host and port using the dumpior script, then interrogate the CORBA Server to determine the Alarm IRP and Notification IRP names.

IBM Copyright 2021

1.4.2 Setting the version of 3GPP CORBA

It is important to set the correct version of 3GPP CORBA so that the probe uses the correct XML, JAR and IDL files for the version of the EMS.

The manual and README files will list the supported Release3GPP property values and the IDLAttrMapFile files exist in the probes/includes directory.

Example setting for the Ericsson OSS-RC EMS is:

```
Release3GPP     : 'V3.2'
IDLAttrMapFile  : '$NCHOME/omnibus/probes/includes/generic_3gpp_v3_2_RuleElementMap.xml'
```

Please refer to the documentation for current supported property settings.

```
Release3GPP : "V3.2"
IDLAttrMapFile : generic_3gpp_v3_2_RuleElementMap.xml
Version 3GPP 3.2:
32.111-3 V3.2.0 - Alarm IRP
32.303 V3.2.0 - Notification IRP

Release3GPP : "V5.5.1"
IDLAttrMapFile : generic_3gpp_v5_5_1.RuleElementMap.xml
Version 3GPP 5.5.1:
32.111-3 V5.5.1 - Alarm IRP
32.303 V5.2.0 - Notification IRP
32.323 V5.2.0 - Generic Network

Release3GPP : "V6.0"
IDLAttrMapFile : generic_3gpp_v6_3_RuleElementMap.xml
Version 3GPP 6.0:
32.111-3 V6.3.0 - Alarm IRP
32.303   V6.4.0 - Notification IRP
32.363   V6.3.0 - Entry Point IRP

Release3GPP : "V6.3"
IDLAttrMapFile : generic_3gpp_v6_3_RuleElementMap.xml
Version 3GPP 6.3:
32.111-3 V6.3.0 - Alarm IRP
32.303 V6.3.0 - Notification IRP
32.363 V6.3.0 - Entry Point IRP

Release3GPP : "V6.4"
IDLAttrMapFile : generic_3gpp_v6_4_RuleElementMap.xml
Version 3GPP 6.4:
32.111-3 V6.4.0 - Alarm IRP
32.303 V6.3.0 - Notification IRP
32.363 V6.3.0 - Entry Point IRP

Release3GPP : "V7.0"
IDLAttrMapFile : generic_3gpp_v7_0_RuleElementMap.xml
Version 3GPP 7.0:
32.111-3 V7.0.0 - Alarm IRP
32.303 V7.0.0 - Notification IRP
32.363 V7.0.0 - Entry Point IRP

Release3GPP : "V9.0"
IDLAttrMapFile : generic_3gpp_v9_0_RuleElementMap.xml
Version 3GPP 9.0:
32.111-3 V9.1.0 - Alarm IRP
32.303   V9.0.0 - Notification IRP
32.363   V9.0.0 - Entry Point IRP

Release3GPP : "V9.1"
IDLAttrMapFile : generic_3gpp_v9_0_RuleElementMap.xml
Version 3GPP 9.1:
32.111-3 V9.1.0 - Alarm IRP
32.303 V9.0.0 - Notification IRP
32.363 V9.0.0 - Entry Point IRP
```

1.4.3 Character encoding

The default character encoding is usually sufficient:

ORBCharEncoding
Use this property to specify the native character encoding set used by the Object Request Broker (ORB) for character data.
The default is **ISO-8859-1.**

ORBWCharDefault
Use this property to specify what wide character (wchar) set the IBM ORB uses when communicating with other ORBs that do not publish a wchar set.
The default is **UTF16**.

However, devices such as the Ericsson OSS-RC EMS require specific settings:

```
ORBCharEncoding : "ISO8859_1"
ORBWCharDefault : "UTF16"
```

If you have the IOR files from the EMS you can check the supported character sets using the dumpior or dior script. e.g.

```
        #1: TAG_ORB_TYPE
                Type: 1245790976 (JacORB)
        #2: TAG_CODE_SETS
                ForChar native code set Id: UTF8
                Char Conversion Code Sets: ISO8859_1, ISO8859_15
                ForWChar native code set Id: UTF16
                WChar Conversion Code Sets: UTF8, UCS2
```

If an unsupported character set is defined, the probe may not be able to read the events, therefore it is important to set these properties to values supported by the EMS.

1.4.4 AlarmIRPName and NotificationIRPName

The AlarmIRPName and NotificationIRPName properties need to be set correctly. Typically the EMS administrator provides the details. If the Naming Service is running, the dumpns script can be used to check or get the required values.

The AlarmIRPName and NotificationIRPName properties need to be set if the Alarm and Notification IRP properties are not being used, as these IOR strings include the target host/port/object, which allow direct connections:

- AlarmIRPIOR
- AlarmIRPIORFile

- NotificationIRPIOR
- NotificationIRPIORFile

When the AlarmIRPName and NotificationIRPName need to be set, then either the Entry Point or Naming Service needs to be connected to, and this service needs knowledge of the given AlarmIRPName and NotificationIRPName names.

- EntryPointIOR
- EntryPointIORFile

- NamingServiceHost and NamingServicePort
- NamingServiceIORfile

An example of the problems experienced with defining the IRP Names, is that the server may require different annotations from those provided by the EMS administrator or dumpns output.

For example, the Ericsson OSS-RC EMS may accept the dot or slash annotation:

```
# Dot notation:
AlarmIRPName                    : 'com.ericsson.nms.fm_cirpagent.AlarmIRP'
NotificationIRPName             : 'com.ericsson.nms.cif.service.NMSNAConsumer'

# Slash notation:
AlarmIRPName                    : 'com/ericsson/nms/fm_cirpagent/AlarmIRP'
NotificationIRPName             : 'com/ericsson/nms/cif/service/NMSNAConsumer'
```

If required the forward slash '\' can be used to escape special characters.

1.4.5 ORBLocalHost and ORBLocalPort

The ORB Local properties are used to define the listening port the Generic 3GPP probe uses to receive notification events. These settings are used when there is a firewall between the probe and the EMS. The EMS communicates to the probe, using these settings. A hostname or IP Address can be used for the ORBLocalHost, with the EMS being aware of, and having access to the same naming service results as the probe server. If necessary the two servers can have each others hostname and IP Addresses added to the local /etc/hosts file. The ORBLocalPort must be a port that is free on the probe server, and accessible, from the EMS.

Example checks available on Red Hat linux systems:
For the settings

```
ORBLocalHost : 'probeserver'
ORBLocalPort : 9191
```

On the probe server:

```
netstat -na | grep 9191
<nothing returned>
```

Confirms the port is free.

On the probe server, start a netcat listener:

```
nc -l -p 9191
```

On the EMS use telnet or a telnet like client to connect to the probe server:

```
telnet probeserver 9191
Trying …
Escape character is '^]'.
hello
```

Make sure that the firewall administrator is aware that the EMS sends data to the probe, and if necessary ask for the ORBLocalPort to be opened in both directions.

1.4.6 TmeTick and Inactivity

The TimeTick property can be set to cause the probes subscription to timeout after a period of inactivity.
The Inactivity property can be set to cause the probe to exit or send a ProbeWatch message to object server after a period of inactivity. The InactivityAction property allows the probe to 'SHUTDOWN' or 'CONTINUE' depending upon the way in which the Netcool/OMNIbus administrator wants to manage the situation when there is a period of inactivity.

The TimeTick property must be less than the Inactivity property value.

For example:

```
TimeTick : 11
Inactivity : 720
InactivityAction : ' SHUTDOWN'
```

Where TimeTick is 11 minutes or 660 seconds, and Inactivity is 720 seconds or 12 minutes.
In this manner the probes subscription will time out first, and after a minute, which is more than enough time for the EMS to clear-up and free the subscription, the probe will exit and be restarted by process control.

IBM Copyright 2021

2 The CORBA Framework

The CORBA Framework is a supporting package which provides the Generic 3GPP Probe with the JacORB API and related software. It is best to ensure that the latest CORBA Framework package is installed, so as to ensure that the probe with work correctly.

The CORBA Framework is added using the nco_p_generic_3gpp.env file which allows customisation of other settings as well as enabling the required classes to run the probe successfully.

The $NCHOME/omnibus/probes/java/corba directory holds the CORBA Framework's JAR files and JacORB API directory.

The dumpns script is provided in the $NCHOME/omnibus/probes/java/corba directory and can be used to check the contents of the Naming Service.

```
Usage: dumpns nshost nsport
Returns the naming context of the session
  nshost : Name Service Host
  nsport : Name Service Port
```

The dior script is provided in the $NCHOME/omnibus/probes/java/corba/jacorb-3.3/bin directory to assist in trroubleshooting.

The dior script allows an IOR string to be listed:

```
Usage: java PrintIOR [ -i ior_str ] [ -f filename] [-u | -c]
        ior_str  IOR as String
        -f       reads one or more IOR's from the file <filename>
        -u       extract object key in URL-Form instead of HEX
        -c       output corbaloc form of IOR string
```

This is useful when there is a firewall and the host/port need to be checked to confirm the firewall has opened the correct ports.

For example:

```
dior -f /tmp/ventIRP.ior
------IOR components-----
TypeId  :       IDL:EventIRPSystem/EventIRPOperations:1.0
TAG_INTERNET_IOP Profiles:
        Profile Id:             0
        IIOP Version:           1.2
        Host:                   <IPADDRESS>
        Port:                   <PORT>
        Object key (URL):       5039643940/%02%01L54%17!%18LL%0B%10%060F8%14%14%1BHL%1B
        Object key (hex):       0x35 30 33 39 36 34 33 39 34 30 2F 02 01 4C 35 34 17 21 18 4C 4C 0B
10 06 30 46 38 14 14 1B 48 4C 1B
        -- Found 3 Tagged Components--
        #0: TAG_SSL_SEC_TRANS
                target_supports :       Integrity, DetectReplay, DetectMisordering
                target_requires :
                SSL Port        :       58689
        #1: TAG_ORB_TYPE
                Type: 1245790976 (JacORB)
        #2: TAG_CODE_SETS
                ForChar native code set Id: UTF8
                Char Conversion Code Sets: ISO8859_1, ISO8859_15
                ForWChar native code set Id: UTF16
                WChar Conversion Code Sets: UTF8, UCS2
```

3 Ericsson OSS-RC EMS

3.1 The Ericsson OSS-RC Probe

There is a vendor specific probe available for the Ericsson OSS-RC EMS. This can be requested from the IBM Account team, and is installed and compiled locally, rather than downloaded from the IBM Support site;
The process is to request the Ericsson OSS-RC SoW probe, where SoW stands for Statement of Work, from the your IBM Account team.

The Ericsson OSS-RC EMS probe support's the vendor specific event streams, EventIRP and the EMS getsubscriptionstatus feature.

```
AlarmIRPName:         "com/ericsson/nms/fm_cirpagent/AlarmIRP"
EventIRPName:         "com/ericsson/nms/fm_cirpagent/EventIRP"
NotificationIRPName: "com/ericsson/nms/cif/service/NMSNAConsumer"
```

3.2 The Generic 3GPP probe

The Generic 3GPP probe can connect to the Ericsson OSS-RC EMS's 3GPP interface, but it cannot read events other than those presented via the Generic 3GPP interface. Events may need to be presented to the Generic 3GPP IRP's using the Generic 3GPP standards, if all events need to read.

The Ericsson OSS-RC EMS does not support the Generic 3GPP getsubscription status function, which means that the probes HeartbeatInterval needs to be unset (HeartbeatInterval:0).

A recent APAR resolves the issue with HeartbeatInterval by continuing to perform the get subscription status rather than exiting:

IT29085 - GENERIC 3GPP PROBE : FIX FOR HEARTBEATINTERVAL FOR THE ERICSSON OSS0-RC EMS

The full set of test fix patches are:

- omnibus-probe-nco-p-generic-3gpp-1.3.5.3.zip
- omnibus-probe-corba-framework-1.10.1.0.zip
- omnibus-probe-sdk-java-1.12.1.0.zip

3.3 Property file

Example probe property settings

```
# Best practice
Server          : "COL_P"
ServerBackup    : "COL_B"
NetworkTimeout  : 15
PollServer      : 30
# Buffering
Buffering       : 1
BufferSize      : 200
FlushBufferInterval : 9
# Performance tuning
DisableDetails  : 1
# General settings
Manager         : 'ericsson_ossrc_generic_3gpp'
MessageLog      : '$OMNIHOME/log/ericsson_ossrc_generic_3gpp.log'
#
# Connection to EMS
#
NamingServiceHost : '<naming service host>'
NamingServicePort : <naming service port>
# 3GPP Version specific settings
Release3GPP       : "V3.2"
IDLAttrMapFile    :
'/opt/IBM/tivoli/netcool/omnibus/probes/includes/generic_3gpp_v3_2_RuleElementMap.xml'
  ORBCharEncoding : "ISO8859_1"
  ORBWCharDefault : "UTF16"
# Determine if slash or dot notation is required using dumpns
# Slash notation:
#AlarmIRPName : 'com/ericsson/nms/fm_cirpagent/AlarmIRP'
#NotificationIRPName    : 'com/ericsson/nms/cif/service/NMSNAConsumer'
# Dot notation:
AlarmIRPName : 'com.ericsson.nms.fm_cirpagent.AlarmIRP'
NotificationIRPName    : 'com.ericsson.nms.cif.service.NMSNAConsumer'
# Ericsson OSS-RC does not support heartbeating
# Without the latest test fix patch
HeartbeatInterval               : 0
# Timeout when needed
# Inactivity > TimeTick > HeartbeatInterval
# 15 minutes is the default EMS timeout
#
# Inactivity                    : 960
# TimeTick                      : 15
# HeartbeatInterval             : 60
#
# Debugging ORB if required - Disable afterwards
#
#ORBDebug: 'true'
#ORBDebugFile :

'$OMNIHOME/log/ericsson_ossrc_generic_3gpp_orb_debug.log'
#
# Local ORB details
ORBLocalHost : '<probe server>'
ORBLocalPort : <free port on probe server>
# EOF
```

3.4 jacorb.properties

There are a few settings that will improve and fix issues with JacORB.

File : $OMNIHOME/probes/java/corba/jacorb-3.3/etc/jacorb.properties

```
# Use DNS names in IORs
jacorb.dns.enable=off
# Timeout connection after 60 seconds
jacorb.connection.client.idle_timeout=60000
```

3.5 Alarm and Notification IRPs

The Ericsson OSS-RC Supports the following Alarm and Notification IRPs:

For Ericsson OSS-RC version 2.x:
Alarm IRP : com.ericsson.nms.fm.ALARM_IRP_CIRPAgent1
Notification IRP : com.ericsson.nms.fm.cif.service.NAConsumer

For Ericsson OSS-RC version 3.x and later:
Alarm IRP : com.ericsson.nms.fm_cirpagent.AlarmIRP
Notification IRP : com.ericsson.nms.cif.service.NMSNAConsumer

For Ericsson RANOS:
Alarm IRP : com.ericsson.nms.umts.ranos.AlarmService_R1 to
Notification IRP : com.ericsson.umts.ranos.NotificationService_R1

For Ericsson CN-OSS:
Alarm IRP : com.ericsson.nms.fm.ALARM_IRP_CIRPAgent1
Notification IRP : com.ericsson.nms.fm.cif.service.NMSNAConsumer

3.5.1 Naming Service

The naming service is a host/port that holds a list of all the objects held in the CORBA Server. The CORBA Framework package includes the dumpns script, which can be used to list the contents of the naming service.
e.g.

```
cd $NCHOME/omnibus/probes/java/corba
./dumpns <naming service host> <naming service port>

   com/
       ericsson/
                nms/
                     fm_cirpagent/
                                 EventIRP
                                 AlarmIRP
                     cif/
                         service/
                                 NMSNAConsumer
```

From this dump, and knowing the version of Ericsson OSS-RC, you can set the probes properties as:

```
Alarmirp       :  "com/ericsson/nms/fm_cirpagent/AlarmIRP"
Notificationirp:  "com/ericsson/nms/cif/service/NMSNAConsumer"
```

or

```
Alarmirp       :  "com.ericsson.nms.fm_cirpagent.AlarmIRP"
Notificationirp:  "com.ericsson.nms.cif.service.NMSNAConsumer"
```

3.6 Migrating from the Generic 3GPP v320 probe

The Generic 3GPP v320 probe was created after support for the Ericsson OSS-RC probe was removed. The Generic 3GPP v320 probe was subsequently replaced by the Generic 3GPP probe, which supports the Generic 3GPP standards for multiple versions of 3GPP. If the probe is being used to connect to the Ericsson OSS-RC EMS it is recommended that the deployment of the Ericsson OSS-RC probe is considered, as this probe will allow all of the Ericsson OSS-RC events to be read, and provide the better solution. If only the Generic 3GPP events are required, then the Generic 3GPP probe can be used.

The Ericsson OSS-RC probe is requested via the IBM Account team as a Statement of Work.

3.6.1 Rules file token mapping

```
# Mapping Generic 3GPP v320 probe tokens to the Geneirc 3GPP probe
$NV_ADDITIONAL_TEXT = $AdditionalText
$NV_PROPOSED_REPAIR_ACTIONS = $ProposedRepairActions
$EVENT_NAME = $EventName
$NV_PERCEIVED_SEVERITY = $PerceivedSeverity
$NV_ACK_USER_ID = $AckUserId
$NV_NOTIFICATION_ID = $NotificationID
$NV_SYSTEM_DN = $SystemDN
$NV_MANAGED_OBJECT_CLASS = $ManagedObjectClass
$DOMAIN_NAME = $DomainName
$NV_EVENT_TIME = $EventTime
$NV_ACK_TIME = $AckTime
$NV_PROBABLE_CAUSE = $ProbableCause
$NV_MANAGED_OBJECT_INSTANCE = $ManagedObjectInstance
$EVENT_TYPE = $EventType
$NV_ACK_STATE = $AckState
$NV_ALARM_ID = $AlarmId
$NV_SPECIFIC_PROBLEM = $SpecificProblem
```

4 Troubleshooting

4.1 Debug logging

The probes debug message level can be used to check the probes behaviour and determine whereabouts the probe is failing. This is most useful when troubleshooting start-up problems or for issues where the error messages are inconclusive or non-existent.

```
# Probe logging
MessageLevel            : 'debug'
MessageLog              : '$OMNIHOME/log/ericsson_ossrc.log'
```

The log file size may need to be increased:
e.g.

```
# 10 Mb log file
MaxLogFileSize                    : 10485760
```

The ORBDebug logging logs ORB messages to a given log file. This logging is useful for issues related to start-up and shutdown, or when there are problems with event reception.

```
# ORBDebug logging
ORBDebug                : 'true'
ORBDebugFile            : '/opt/IBM/tivoli/netcool/omnibus/log/ericsson_ossrc_debug.log'
```

Sometimes it is necessary to use tcpdump/snoop to capture the tcp/ip traffic between the probe and EMS. It is best to capture the traffic on both servers, and the traffic between the given interfaces.
e.g. for Linux

Check the interfaces:

```
netstat -ni
Kernel Interface table
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
ens192    1500 ########      0 ###### 0      #########      0      0      0 BMRU
lo       65536 ########      0      0 0      #########      0      0      0 LRU
```

Capture packets on the correct interface and for the given EMS:

```
tcpdump -i ens192 -s 0 -w /tmp/probeserver.pcap host emsserver
```

Where emsserver is the host or ip address of the EMS server.

4.2 Example Information MessageLevel logging

```
Information: I-JPR-000-000: ORBLocalHost : OAIAddr = <IPADDRESS>
Information: I-JPR-000-000: ORBLocalPort : OAPort = 10001
Information: I-JPR-000-000: ORBWCharDefault : jacorb.native_wchar_codeset = UTF16
Information: I-JPR-000-000: ORBCharEncoding : jacorb.native_char_codeset = ISO8859_1
Information: I-JPR-000-000: ORBDebug : jacorb.log.default.verbosity = 4
Information: I-JPR-000-000: ORBDebugFile : jacorb.logfile =
/opt/IBM/tivoli/netcool/omnibus/log/ericsson_ossrc_debug.log
Information: I-JPR-000-000: Jacorb custom properties file path : custom.props =
/opt/IBM/tivoli/netcool/omnibus/probes/java/corba/jacorb-3.3/etc/jacorb.properties
Information: I-JPR-000-000: ORB has been setup
Information: I-JPR-000-000: Completed ORBChannel setup routine
Information: I-JPR-000-000: Setup event processor....
Information: I-JPR-000-000: Completed setup routine
Information: I-UNK-000-000: Probewatch: Running ...
Information: I-JPR-000-000: Probe started
Information: I-UNK-000-000: Registering probe URI with HTTP context.
Information: I-JPR-000-000: Connecting to NotificationIRP
Information: I-JPR-000-000: Connecting to AlarmIRP
Information: I-JPR-000-000: Endpoint connected
Information: I-JPR-000-000: Successfully subscribed to notification
Information: I-JPR-000-000: Endpoint subscribed
Information: I-JPR-000-000: Waiting on resync to finish
Information: I-JPR-000-000: Resynchronizing Probe
Information: I-UNK-000-000: Probewatch: START SYNCHRONIZATION
Information: I-UNK-000-000: Probewatch: END SYNCHRONIZATION
Information: I-JPR-000-000: Resync finished
Information: I-JPR-000-000: Probe connected
```

In this logging we can see that the ORBLocalHost and ORBLocalPort are set.
These are used to setup a listener on the probe server to receive events from the EMS.
e.g.

```
netstat -na | grep 10001
<IP ADDRESS>.10001         *.*                0      0 32768      0 LISTEN
```

The EMS needs to be able to access this port.

The probes character encoding are confirmed in the logging.

```
# ORBWCharDefault       : 'UTF16'
ORBCharEncoding         : 'ISO8859_1'
```

The path to the jacorb.properties file is confirmed. The jacorb.properties sets the way in which the ORB handles timeouts and naming services.

The probe successfully connects to and subscribes to the given objects: NotificationIRP and AlarmIRP

```
AlarmIRPName            : 'com/ericsson/nms/fm_cirpagent/AlarmIRP'
NotificationIRPName     : 'com/ericsson/nms/cif/service/NMSNAConsumer'
```

Once the probe completes the synchronization it confirms it is connected successfully to the EMS.

After this, the probe should receive notification events, which will be visible in the Debug: messages.

4.3 Incorrect Release3GPP setting

```
Failed to connect; ProbeException: Error connecting to EMS IRP :
Debug: D-JPR-000-000: com.ibm.tivoli.netcool.omnibus.probe.ProbeException: Error connecting to EMS IRP :
        at com.ibm.tivoli.netcool.integrations.corbaframework._3gpp.Endpoint3GPP.connect(Endpoint3GPP.java:164)
        at com.ibm.tivoli.netcool.integrations.corbaframework.base.CorbaProbeImpl.connect(Unknown Source)
        at com.ibm.tivoli.netcool.omnibus.probe.framework.ProbeRunner.connect(Unknown Source)
        at com.ibm.tivoli.netcool.omnibus.probe.framework.ProbeRunner.connectAndRun(Unknown Source)
        at com.ibm.tivoli.netcool.omnibus.probe.framework.ProbeRunner.run(Unknown Source)
        at com.ibm.tivoli.netcool.omnibus.oidk.Probe.start(Unknown Source)
        at com.ibm.tivoli.netcool.omnibus.oidk.Probe.main(Unknown Source)
Debug: D-JPR-000-000: com.ibm.tivoli.netcool.omnibus.probe.framework.ProbeRunner.connect EXITING
Information: I-JPR-000-000: DISCONNECT 'Unable to connect'
```

Check the probe property Release3GPP is set correctly for the target EMS.

4.4 Incorrect tokenisation

The expected event tokens are not available.
e.g.

```
[Event Processor] a: <value>
[Event Processor] b: <value>
[Event Processor] c: <value>
[Event Processor] d: <value>
```

Check that the IDLAttrMapFile is set correctly for the Release3GPP and that the file is up to date.
Try using the full path.
e.g.
ls -l /opt/IBM/tivoli/netcool/omnibus/probes/includes/generic_3gpp_v3_2_RuleElementMap.xml
/opt/IBM/tivoli/netcool/omnibus/probes/includes/generic_3gpp_v3_2_RuleElementMap.xml

4.5 Incorrect AlarmIRP

```
Debug: D-JPR-000-000: Resolving initial references to NamingService using hostPort for IRP:
com.ericsson.nms.fm_cirpagent.AlarmIRP
Debug: D-UNK-000-000: (Non-encrypted property) NamingServiceIORFile->
Information: I-UNK-000-000: Probewatch: Unable to get events. Failed to connect; ProbeException: Error
connecting to EMS IRP : java.lang.NullPointerException
Error: E-JPR-000-000: NamingContextPackage NotFound exception while resolve naming
context:IDL:omg.org/CosNaming/NamingContext/NotFound:1.0
```

Check that the name exists in the EMS using a dump of the objects.
For the Ericsson OSS-RC EMS, two notations are used, dot and slash.
e.g.

```
AlarmIRPName      : 'com/ericsson/nms/fm_cirpagent/AlarmIRP'
AlarmIRPName      : 'com.ericsson.nms.fm_cirpagent.AlarmIRP'
```

When a naming service is available the dumpns tool can be used to check the details.
e.g.

```
cd $NCHOME/omnibus/java/corba
./dumpns host port
com.ericsson.nms.fm_cirpagent.AlarmIRP
```

If the administrator is not sure what the AlarmIRP name is, they can provide the IOR string for the AlarmIRP.
The dior tool can then be used to dump the details of the IOR string.
e.g.

```
cd $NCHOME/omnibus/java/corba/jacorb-3.3/bin
./dior -f /tmp/AlarmIRP.IOR

TypeId   :       IDL:3gppsa5.org/AlarmIRPSystem/AlarmIRPOperations:1.0
TAG_INTERNET_IOP Profiles:
Profile Id:             0
IIOP Version:           1.2
Host:                   <IP>
Port:                   <PORT>
```

The AlarmIRP.IOR and NotificationIRP.IOR files can then be used instead of the object names, after making sure the probe server has access to the IOR files host/ports.

```
AlarmIRPIORFile          : '/opt/IBM/tivoli/netcool/omnibus/var/AlarmIRP.IOR'
NotificationIRPIORFile   : '/opt/IBM/tivoli/netcool/omnibus/var/NotificationIRP.IOR'
```

4.6 Incorrect NotificationIRP

```
Debug: D-JPR-000-000: Resolving initial references to NamingService using hostPort for IRP:
com.ericsson.nms.cif.service.NMSNAConsumer
Debug: D-UNK-000-000: (Non-encrypted property) NamingServiceIORFile->
Information: I-UNK-000-000: Probewatch: Unable to get events. Failed to connect; ProbeException:
Error connecting to EMS IRP : null
Error: E-JPR-000-000: NamingContextPackage NotFound exception while resolve naming
context:IDL:omg.org/CosNaming/NamingContext/NotFound:1.0
Information: I-JPR-000-000: Connecting to NotificationIRP
Error: E-JPR-000-000: Failed to connect; ProbeException: Error connecting to EMS IRP : null
```

Check that the name exists in the EMS using a dump of the objects.
For the Ericsson OSS-RC EMS, two notations are used, dot and slash.
e.g.

```
NotificationIRPName  : 'com/ericsson/nms/cif/service/NMSNAConsumer'
NotificationIRPName  : 'com.ericsson.nms.cif.service.NMSNAConsumer'
```

When a naming service is available the dumpns tool can be used to check the details.
e.g.

```
cd $NCHOME/omnibus/java/corba
./dumpns host port
com/ericsson/nms/cif/service/NMSNAConsumer
```

If the administrator is not sure what the NotificationIRP name is, they can provide the IOR string for the NotificationIRP.
The dior tool can then be used to dump the details of the IOR string.
e.g.

```
cd $NCHOME/omnibus/java/corba/jacorb-3.3/bin
./dior -f /tmp/NotificationIRP.IOR

TypeId  : IDL:3gppsa5.org/NotificationIRPSystem/NotificationIRPOperations:1.0
TAG_INTERNET_IOP Profiles:
Profile Id:             0
IIOP Version:           1.2
Host:                   <IP>
Port:                   <PORT>
```

The AlarmIRP.IOR and NotificationIRP.IOR files can then be used instead of the object names, after making sure the probe server has access to the IOR files host/ports.

```
AlarmIRPIORFile          : '/opt/IBM/tivoli/netcool/omnibus/var/AlarmIRP.IOR'
NotificationIRPIORFile   : '/opt/IBM/tivoli/netcool/omnibus/var/NotificationIRP.IOR'
```

4.7 Checking the ORBLocalPort

The ORBLocalPort is used to receive events from the EMS. Therefore it needs to be available to the EMS from the EMS. The JacORB properties file can be used to define how the ORBLocalPort is addressed.

If the probe is using:

```
ORBLocalHost :  192.168.20.20
ORBLocalPort :  10001
```

Check the probe Is listening whilst the probe is running:

```
netstat -na | grep 10001
tcp6       0      0 192.168.20.20:10001     :::*                    LISTEN
```

From the EMS check that the ORBLocalPort is accessible whilst the probe is running:

```
telnet <ORBLocalHost > <ORBLocalPort>
telnet 192.168.20.20 10001
telnet probe_server.mycomany.com 10001
telnet probe_server  10001
```

Make sure that the probe and EMS are able to address each other consistently.

Ensure that any firewall rules allow traffic from the EMS to the probe server.

The ORBLocalHost is used to define the interface the probe port runs on, by defining the name or IP Address on which to listen.

4.8 ORBLocalHost not accessible from EMS

In some networks the Probe server is not accessible using the probes servers IP Address or hostname. In these cases there is a need to use the JacORB proxy host setting.

The best way to ensure the name given as the proxy host setting is used is to not define a correct name for ORBLocalHost setting in the probes property file:

Set the probe property ORBLocalHost to some unknown hostname:

e.g.

File : $NCHOME/omnibus/probes/linux2x86/generic_3gpp.props

```
ORBLocalHost : 'NOTINTHENAMINGSERVICE'
```

File : $NCHOME/omnibus/probes/java/corba/jacorb-3.3/etc/jacorb.properties

Define the proxy hostname for the probe server as required:

```
jacorb.ior_proxy_host=remotehostname
```

The JacORB API used by the CORBA Framework either uses the IP Address or the first name in the naming service lookup, when the ORBLocalHost property is set.

This behaviour is a feature of the JacORB API.