Tivoli Netcool Support's Guide to the Tivoli EIF Gateway

by

Jim Hutchinson

Document release: 2.1

Table of Contents

1 Introduction

1.1 Overview

The Tivoli EIF Gateway is used in the integration with the TEC system, and forwards events from the given Object Server to a Tivoli EIF server. It can be used with other Tivoli EIF servers.

1.2 Version details

The Tivoli EIF gateways version details provided details as to the software being used:

```
nco_g_tivoli_eif -version
Netcool/OMNIbus EIF Gateway - Version 8.1.0 64-bit
(C) Copyright IBM Corp. 1994, 2012
Code Revision: 5.0.3

Library Revisions:
        libnetcool: 5.50.86
        libnipc_client: 5.50.78
        libniduc_client: 5.50.20
        libnipc: 5.50.78
        libnstk: 5.50.31
        libngtk: 5.50.69
        libngobjserv: 5.50.85
        network::ipv6: 5.50.20

Compilation Date:       Wed May 20 09:33:42 UTC 2015
Compilation Machine:    rhat5es-build1.hursley.ibm.com
Compilation System:     Linux 2.6.18-274.17.1.el5 #1 SMP Wed Jan 4 22:45:44 EST 2012

Code Generation: PRODUCTION
```

2 Example Configuration

Environment:
Netcool/OMNIbus 8.1
Red Hat 7.4

Directory: $NCHOME/omnibus/gates/G_EIF
Gateways configuration files:

- G_EIF.props
- tivoli_eif_config
- tivoli_eif.map
- tivoli_eif.rdrwtr.tblrep.def
- tivoli_eif.startup.cmd

It is possible to set any specific environment variables in the gateways environment file:
e.g.
$NCHOME/omnibus/platform/solaris2/bin/nco_g_tivoli_eif.env

2.1 G_EIF.props

```
Gate.RDRWTR.Server                    : 'AGG_V'
Gate.RDRWTR.Username                  : 'root'
Gate.RDRWTR.Password                  : ''
Name                                  : 'G_EIF'
Gate.TIVOLI_EIF.GWInstanceName        : 'TIVOLI_EIFGW'
Gate.RDRWTR.Description               : 'TIVOLI_EIF_Gateway'
# Settings accept $OMNIHOME paths
MessageLog                            : '$OMNIHOME/log/G_EIF.log'
Gate.MapFile                          : '$OMNIHOME/gates/G_EIF/tivoli_eif.map'
Gate.StartupCmdFile                   : '$OMNIHOME/gates/G_EIF/tivoli_eif.startup.cmd'
Gate.RDRWTR.TblReplicateDefFile       : '$OMNIHOME/gates/G_EIF/tivoli_eif.rdrwtr.tblrep.def'
# Provide the full path to the file
Gate.TIVOLI_EIF.ConfigFile            : '/opt/IBM/tivoli/netcool/omnibus/gates/G_EIF/tivoli_eif_config'
Gate.TIVOLI_EIF.MinimumSeverity       : 0
Gate.RDRWTR.IducFlushRate             : 59
Gate.TIVOLI_EIF.FullResync            : FALSE
Gate.TIVOLI_EIF.Resync                : TRUE
MaxLogFileSize                        : 10240
MessageLevel                          : 'debug'
Gate.TIVOLI_EIF.DeleteClass           : 'OMNIbus_Delete'
Gate.RDRWTR.SAF                       : TRUE
# Provide the full path to the file
Gate.RDRWTR.SAFFile                   : '/opt/IBM/tivoli/netcool/omnibus/var/tivoli_eif/G_EIF_RDRWTR.store'
Gate.RDRWTR.SAFReplayOnResync         : TRUE
```

2.2 tivoli_eif_config

The tivoli_eif_config file sets the target EIF server and other Tivoli EIF settings:

```
TransportList=t1
t1Type=SOCKET
t1Channels=c1
# Target EIF host and port
c1ServerLocation=eif-server.ibm.com
c1Port=12345
ConnectionMode=connection_oriented
# Cache file full path location
BufEvtPath=/opt/IBM/tivoli/netcool/omnibus/var/tec/G_EIF.cache
# It is recommended that events are not buffered
BufferEvents=NO
#BufferEvents=YES
BufEvtMaxSize=640000
#EOF
```

2.3 tivoli_eif.rdrwtr.tblrep.def

The general syntax is:

```
REPLICATE ALL | (INSERTS, UPDATES, DELETES)
FROM TABLE sourcetable
USING MAP mapname
[FILTER WITH filter]
[INTO targettable]
[ORDER BY order, ... ]
[AFTER IDUC DO afteriduc] ;
```

Example file:

```
REPLICATE ALL FROM TABLE 'alerts.status'
USING MAP 'StatusMap'
FILTER WITH 'TO_EIF>1'
order by 'ServerName,ServerSerial,LastOccurrence ASC'
AFTER IDUC DO 'TO_EIF=2'
;
#EOF
```

2.4 tivoli_eif.map

The mapping file converts the Object Server fields into fields sent to the EIF server.

```
CREATE MAPPING StatusMap
(
        'identifier'        =    '@Identifier',
        'server_identifier'=    '@ServerName' + " " + TO_STRING('@ServerSerial'),
        'sub_source'        =    '@AlertKey'             ON INSERT ONLY,
        'sub_origin'        =    '@AlertGroup'           ON INSERT ONLY,
        'msg'               =    '@Summary'              ON INSERT ONLY,
        'origin'            =    '@Node'                 ON INSERT ONLY,
        'node_alias'        =    '@NodeAlias'            ON INSERT ONLY
                                                         NOTNULL '@Node',
        'manager'           =    '@Manager'              ON INSERT ONLY,
        'source'            =    '@Agent'                ON INSERT ONLY,
        'severity'          =    '@Severity',
        'date'              =    '@LastOccurrence'       ON INSERT ONLY,
        'omnibus_last_modified_time' = '@InternalLast'  ON INSERT ONLY
);

#EOF
```

2.5 tivoli_eif.startup.cmd

This file is usually empty.

2.6 Object Server Updates

Add the filter and after IDUC flag to the object servers that the gateway connects to:
e.g.

```
alter table alerts.status add TO_EIF int;
go
```

In the new_row and deduplication triggers for the same object servers add entries to set TO_EIF to 1, so that any new rows or updates to events can be processed by the gateway.

If all events do not need to be processed then ensure the source systems set TO_EIF to 1, where required or else create custom triggers to manage event forwarding.

3 Special Configurations

3.1 Gateway Failover Failback

The Gate.TIVOLI_EIF.ConfigFile can be configured to send to one or more transport/channels. Events are forwarded to just one of the transport/channels in the transport list. This allows for failover, and failback in the event of the primary [first] transport/channel becoming unavailable.

e.g.

```
Gate.TIVOLI_EIF.ConfigFile  : '/opt/IBM/tivoli/netcool/omnibus/gates/G_EIF/tivoli_eif_config.failover'
```

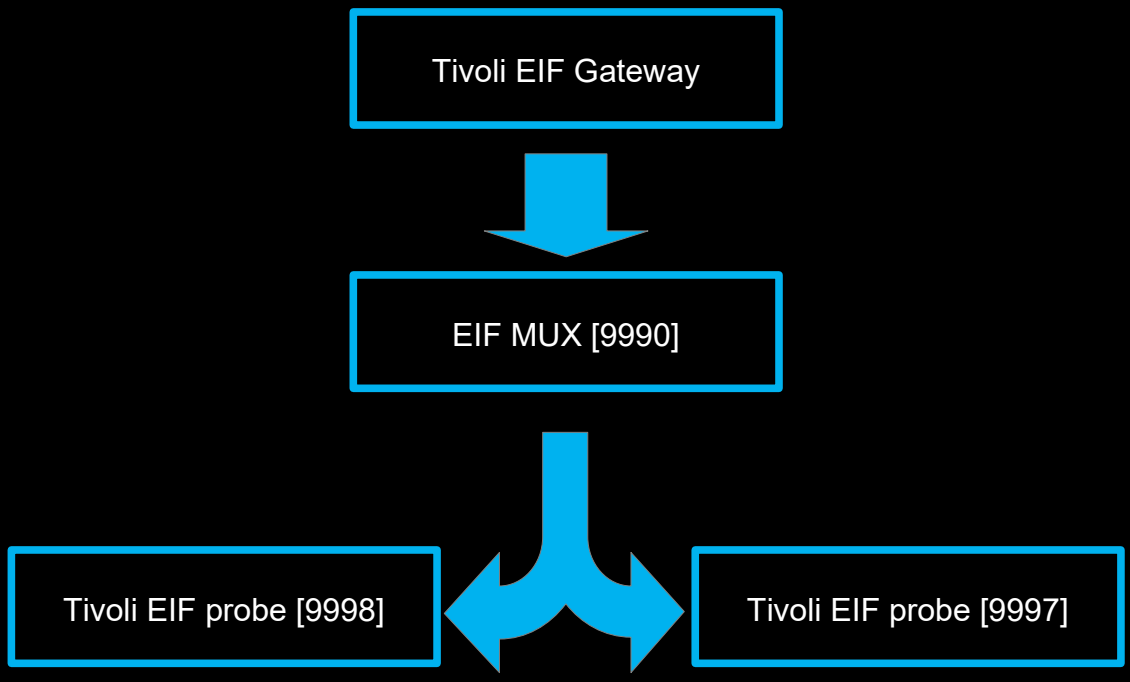
File : tivoli_eif_config.failover

```
# Target host/port
TransportList=t1,t2
# Define transport/channel #1
t1Type=SOCKET
t1Channels=c1
c1ServerLocation=localhost
c1Port=9998
# Define transport/channel #2
t2Type=SOCKET
t2Channels=c2
c2ServerLocation=localhost
c2Port=8887
# Buffering
BufferEvents=NO
# Buffering configuration
# BufferEvents=YES
# BufEvtPath=/opt/IBM/tivoli/netcool/omnibus/var/G_EIF/eif.cache
# EventMaxSize=262144
# EOF
```

3.2 Sending to two event readers

The Tivoli EIF API for the Tivoli EIF gateway does not allow events to be sent to two or more event readers. However, the EIF MUX allows for a clone reader to be configured using the clone.conf file. With buffering enabled the events will be kept in synchronisation between the two EIF readers, for a short time, should they require restarting. For test purposes the Tivoli EIF probe can be used to test the EIF MUX behaviour, alongside the posteifmsg command line tool, or Tivoli EIF Gateway.

The Support's guide to the EIF MUX provides a recommended integratio, complete with a directory structure that allows for configuration, buffer cache and logging. Within this environment the EIF MUX can be configured to send duplicate events to two EIF readers.



3.2.1 EIF MUX configuration settings

File : in.conf

```
BufEvtPath=../var/mux_in.dat
BufferEvents=NO
RetryInterval=1
#ed_diag_config_file=../conf/mux_diag_config.txt
TransportList=t2_
t2_Type=SOCKET
t2_Channels=c2_
c2_ServerLocation=myserver.mycomany.com
c2_Port=9990
# EOF
```

File : out.conf

```
# Buffering
BufferEvents=YES
BufEvtMaxSize=640000
BufEvtPath=../var/mux_primary_out.dat
#BufferEvents=NO
# Logging
ed_diag_config_file=../conf/eif_diag_config.txt
LogFileName=../logs/eifout.log
FQDomain=NO
LogLevel=NO
TestMode=NO
RetryInterval=1
# Transport/channel to probe
TransportList=t1_
t1_Type=SOCKET
t1_Channels=c1_
c1_ServerLocation=localhost
c1_Port=9998
ConnectionMode=connection_oriented
# EOF
```

File : clone.conf

```
# Buffering
BufferEvents=YES
BufEvtMaxSize=640000
BufEvtPath=../var/mux_clone_out.dat
#BufferEvents=NO
# Logging
FQDomain=NO
LogLevel=NO
LogFileName=../logs/eifclone.log
TestMode=NO
RetryInterval=1
# Define T2->C2 to Tivoli EIF ProbeServer
Location=localhost
ServerPort=9997
ConnectionMode=connection_oriented
# EOF
```