



EMC® Watch4net®

Version 6.3u2

Outage Manager Administration Guide

P/N: pending

Rev : 40030

EMC²

Copyright ©2008 - 2014 EMC Corporation. All rights reserved. Published in the USA.

Published April, 2014

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

The information in this publication is provided as is. EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose. Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

EMC², EMC, and the EMC logo are registered trademarks or trademarks of EMC Corporation in the United States and other countries. All other trademarks used herein are the property of their respective owners.

For the most up-to-date regulatory document for your product line, go to EMC Online Support (<https://support.emc.com>).

Contents

Preface	iii
Document Organization	iii
Where to get help	v
1 Introduction	1
1 The <i>Outage Manager</i> server	1
2 The <i>Outage Detector</i>	2
3 The <i>Dynamic Time Filter</i> manager	2
2 Configuration	3
1 The <i>Outage Manager</i> server	3
1.1 <i>daemon.properties</i>	3
1.2 <i>store.properties</i>	4
2 The <i>Outage Detector</i>	5
3 The <i>Dynamic Time Filter</i> Manager	7
3 Usage	9
1 Outage management	9
2 Usage in reports	10
2.1 Exclude outages in reports	10
2.2 Availability from outages	10
A <i>EMC² Smarts InCharge</i> integration	11

Preface

Document Organization

1. Introduction	This section describes the <i>Outage Manager</i> system and each of its components.
2. Configuration	This section describes how to configure each of the <i>Outage Manager</i> components.
3. Usage	This section describes how to use the <i>Outage Manager</i> for the reporting.

Where to get help

EMC support, product, and licensing information can be obtained as follows:

PRODUCT INFORMATION - For documentation, release notes, software updates, or information about EMC products, go to EMC Online Support at:

<https://support.emc.com>

TECHNICAL SUPPORT - Go to EMC Online Support and click Service Center. You will see several options for contacting EMC Technical Support. Note that to open a service request, you must have a valid support agreement. Contact your EMC sales representative for details about obtaining a valid support agreement or with questions about your account.

Introduction

This section describes the main components of the *Outage Manager* system. Each component rely on the *Outage Manager* server which stores and manages outages.

1 The *Outage Manager* server

The *Outage Manager* is a standalone server which will manage its own outage database. An outage database is called a *category*. A *category* groups outages related to the same kind of objects (for example, device outages would be stored in a single category). A single *Outage Manager* server can manage several *categories*.

Within a *category*, an outage is uniquely identified by three attributes:

- An object name: it is the name of the object this outage belongs to.
- A start date: this is the beginning of an outage
- A end date: this is the end of an outage. Note that an outage can be *open* which means that it doesn't have an end date yet (it would be equals to the biggest timestamp then).

Additionally, an outage stores several other properties:

- A integer type which is not used internally and can be used for external tool integration.
- A bit mask which represents some flags. Three flags are already defined:
 - *planned*: this flag is informational, and indicates whether this outage was expected or not
 - *excluded*: this flag is used by application to exclude the time frame of this outage from the computation
 - *auto generated*: this flag is used internally to identify outages which have been automatically generated (by the *Outage Detector* for example)
- A comment: something which describes the outage

The *Outage Manager* is accessible via an *XML-RPC* interface for external tool integration and other *APG* components.

2 The *Outage Detector*

The *Outage Detector* is a collecting filter (*DCF*) which monitors the collected data on-line and creates outages automatically. This filter synchronizes new and closed outages with the remote *Outage Manager* servers using its *XML-RPC* interface.

3 The *Dynamic Time Filter* manager

The *Dynamic Time Filter* is the component, within the Web portal, which retrieves and uses outages for computation. The *Dynamic Time Filter* remotely connects to one or more *Outage Manager* server using its *XML-RPC* interface.

The concept of *outages* in the Web portal is usually called *Time filter* or *Maintenance Period*. The *Dynamic Time Filter* provides the ability to specify an object specific maintenance period based on outages declared in an *Outage Manager* server category. The *Dynamic Time Filter* also enables availability computation from declared outages, which has the advantage to be very fast and accurate.

Configuration

1 The *Outage Manager* server

The *Outage Manager* sever has two configuration files located in `conf`:

- `daemon.properties` setups the *XML-RPC* interfaces
- `store.properties` setups the outage storage backends for each category

1.1 `daemon.properties`

Here is an example of `daemon.properties` configuration file:

```
# XML-RPC configuration
outage.daemon.xmlrpc.port=8888
outage.daemon.xmlrpc.accept.0=127.0.0.1
#outage.daemon.xmlrpc.deny.0=10.*
outage.daemon.smarts.xmlrpc.port=8889
#outage.daemon.smarts.xmlrpc.accept.0=127.0.0.1
```

```
#outage.daemon.smarts.xmlrpc.deny.0=10.*

# SAM XML-RPC specific setup
outage.daemon.smarts.broker=127.0.0.1:426
#outage.daemon.smarts.broker.user=admin
#outage.daemon.smarts.broker.pass=changeme
outage.daemon.smarts.object=DisplayName
outage.daemon.smarts.category=device
outage.daemon.smarts.dm.0=INCHARGE-AM-PM
outage.daemon.smarts.dm.user.0=admin
outage.daemon.smarts.dm.pass.0=changeme
```

This mainly configures the *XML-RPC* interfaces port. The *Outage Manager* comes with two *XML-RPC* interfaces:

- the main one, on port 8888 allow direct access to the outage categories. It is used by every *APG* components and should be restricted to *APG* servers IP addresses.
- the second one, on port 8889 allow *EMC² Smarts InCharge* SAM integration. Since this one will be protected by *EMC² Smarts InCharge* authentication, it can be unrestricted. See [EMC² Smarts InCharge integration](#) (page 11) for more information.

The accept and deny properties can appear several time, using an index (e.g. outage.daemon.xmlrpc.accept.0, outage.daemon.xmlrpc.accept.1, ...). The accepted values are IP addresses or patterns.

The *EMC² Smarts InCharge* SAM configuration will be described in [EMC² Smarts InCharge integration](#) (page 11).

1.2 store.properties

Here is an example of store.properties configuration file. You should not have to modify this file.

```
outage.store.default=com.watch4net.apg.v2.outage.impl.JDBCOutageStore
#outage.store.cat1=com.watch4net.apg.v2.outage.impl.JDBCOutageStore

##
# Use an embedded database by default.
##
outage.store.default.driver=org.h2.Driver
outage.store.default.url=jdbc:h2:data/${category};TRACE_LEVEL_FILE=0
outage.store.default.username=sa
outage.store.default.password=sa
```

```

outage.store.default.statement.validate=SELECT * FROM ${category}_outages WHERE 1!=1
outage.store.default.statement.create=CREATE TABLE ${category}_outages (object VARCHAR, startts
INT, endts INT, flags INT NOT NULL DEFAULT 0, type INT NOT NULL DEFAULT 0, comment VARCHAR NOT
NULL DEFAULT '', PRIMARY KEY (object, startts, endts)); CREATE INDEX ON ${category}_outages(
startts); CREATE INDEX ON ${category}_outages(endts)
outage.store.default.statement.destroy=DROP TABLE ${category}_outages
outage.store.default.statement.insert=MERGE INTO ${category}_outages VALUES (?, ?, ?, ?, ?, ?)
outage.store.default.statement.remove=DELETE FROM ${category}_outages WHERE object=? AND
startts=? AND endts=?
outage.store.default.statement.select.objects=SELECT DISTINCT object FROM ${category}_outages
WHERE object LIKE ? ORDER BY object ASC
outage.store.default.statement.select.all.outage=SELECT * FROM ${category}_outages
outage.store.default.statement.select.object.outage=SELECT * FROM ${category}_outages WHERE
object=?
outage.store.default.statement.select.all.time.outage=SELECT * FROM ${category}_outages WHERE
startts BETWEEN ? AND ? OR endts BETWEEN ? AND ? OR ? BETWEEN startts AND endts
outage.store.default.statement.select.object.time.outage=SELECT * FROM ${category}_outages
WHERE object=? AND (startts BETWEEN ? AND ? OR endts BETWEEN ? AND ? OR ? BETWEEN startts AND
endts)

```

In this configuration file, you can choose which storage backend will be used for each category. To define a backend for all the categories, simply use `default` instead of the category name. You can use `${category}` after to reference the actual category name.

The default setting use an embedded database (located in `data`) which requires no maintenance at all. Since storing outages is not as big as storing raw availability, this database should not get very big.

2 The Outage Detector

The *Outage Detector* is a collecting filter, and then it should be added in your running `collecting.xml` according to the collector configuration reference in the *Backend Administration guide*. The filter declaration should look like this:

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE config SYSTEM "collecting.dtd">
<config>
  <!-- ... -->
  <filters>

```

```
<filter name="OutageDetector" next="...">
  <classname>com.watch4net.apg.v2.outage.filter.OutageDetector</classname>
  <config-file>conf/outage-detector.xml</config-file>
</filter>
</filters>
<!-- ... -->
</config>
```

You can have as many *Outage Detector* as you need, each of them will manage values for a specific *category*.

The *Outage Detector* is configured with its own configuration file, usually called `outagedetector.xml`. Here is the default configuration file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE config SYSTEM "outage-detector.dtd">
<config>
  <remote-manager type="xmlrpc" category="device">
    http://localhost:8888/xmlrpc
  </remote-manager>
  <object-property>device</object-property>
  <object-availability threshold="100">
    name='Availability' & !parttype
  </object-availability>
</config>
```

The `remote-manager` element specifies the *Outage Manager* server *XML-RPC* address and the *category* to store detected outages in.

The *Outage Detector* will look for outages in values which match the filter specified in the `object-availability` element. Values under the specified `threshold` will open an outage and value above or equal to the `threshold` will close an opened outage. The created outages will use the property specified in the `object-property` element as the outage object name.

WARNING: You **MUST** configure collectors to include properties **in each polled values** to use the *Outage Detector*. Values without properties will not be analysed by the *Outage Detector*.

3 The *Dynamic Time Filter* Manager

The *Dynamic Time Filter* Manager is configured through the following file:

<APG>/Backends/Outage-Manager/<instance name>/conf/**common.properties**

The *Dynamic Time Filter* Manager requires 4 properties to be set.

- `remote.outage.alias.0` is a convenient alias for the next three properties. Each time we want to reference a specific category from a specific *Outage Manager* in the Web portal, we will use this name.
- `remote.outage.property.0` is the name of the property from which object name is taken from.
- `remote.outage.url.0` is the *Outage Manager* server *XML-RPC* URL.
- `remote.outage.category.0` is the *Outage Manager* category where outages are stored.

This group of properties can be added several times changing the index at the end of their name (`alias.0`, `alias.1`, ...). This way, the *Dynamic Time Filter* Manager can be configured to use several category on several *Outage Manager* servers.

Usage

1 Outage management

The outages can be viewed, created, modified and deleted from the Web portal. This is an administrative privilege. You can go to the manager from the Administration interface, clicking on the *Outage Editor* link.

From this interface, after having selected the alias and an object name, you can review, modify or delete outages for this object. The changes will be reflected in real time for all users.

For an outage, you cannot change its start date, end date and object name. All the other properties can be changed.

2 Usage in reports

This section will describe how to use outages from the *Outage Manager* servers in the reports generated by the *APG* Web portal.

2.1 Exclude outages in reports

To exclude time ranges from report, based on outages declared from the *Outage Manager* server, you will have to create a dynamic maintenance period for the report node. To do that, go to *Edit mode* and select the *Report Configuration* tab. In the *Advanced Settings*, use the *Dynamic Maintenance Periods* setting.

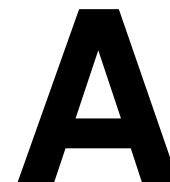
You will have to choose the alias (referring to the *Outage Manager* instance) and a node property to use as the object name. By default, *APG* will use node's name as the object name but you can change it if needed. For example, you can enter `parent.name` to use the name of the parent node to determine the outage period for this node. The maintenance period matching the name of the parent node is thus applied to the (child) node you are configuring.

When you go back to Browse mode, select the node you've just set up and you should see the excluded outages on the graph (yellow area). The yellow areas will be excluded from any computation.

2.2 Availability from outages

When you have setup a *dynamic maintenance period* on a node, you can use the *AvailabilityFromOutage* formula on the same node. This formula will recompute the availability metric from the outages, assuming that the device was up at some time unless an outage exists. The formula will honor excluded outages and discard them from computation.

The main advantage this formula is that you can get availability for an object in a very efficient way, accurate to the minute even several months in the past.



EMC² **Smarts InCharge** integration

The *Outage Manager* server can be used from *EMC² Smarts InCharge* SAM domain to insert outages from an event. This functionality is provided by the specialized *XML-RPC* listening on port 8889 by default.

This interface provides the ability to add an outage from an *EMC² Smarts InCharge* object. The *Outage Manager* server will then query the SAM domain to retrieve the object and find out which domains it comes from to retrieve the real object name from the actual domain.

A wrapper class has been written (the source is also provided in `misc`) to add an outage from command line (`store-sam-outage.sh`) or from a client tool.

To enable this specialized *XML-RPC*, make sure that `daemon.properties` contains the following lines:

```
...  
# SAM XML-RPC specific setup  
outage.daemon.smarts.broker=127.0.0.1:426
```

```
#outage.daemon.smarts.broker.user=admin  
#outage.daemon.smarts.broker.pass=changeme  
outage.daemon.smarts.object=DisplayName  
outage.daemon.smarts.category=device  
outage.daemon.smarts.dm.0=INCHARGE-AM-PM  
outage.daemon.smarts.dm.user.0=admin  
outage.daemon.smarts.dm.pass.0=changeme
```

You will have to set the broker address (with username and password if broker authentication is required) and the credential from each of the SAM domains an object may belongs to. Additionally, you will have to specify the object attribute used to create outage object name and the category where to store outages.