



EMC® Watch4net®

Version 6.3u2

VMware Collector Administration Guide

P/N: pending

Rev : 45657

EMC²

Copyright ©2010 - 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
Purpose	iii
Intended Audience	iii
Prerequisites	iii
Common Abbreviations and Acronyms	iv
Document Organization	iv
Where to get help	v
1 Collector Manager Configuration	1
2 vSphere Documentation	3
1 Important information:	3
3 VMware Collector Configuration	5
1 APG Values	5
2 Scope	5
3 VMware Collector context handlers	6
3.1 Array handler	6
3.2 Datastore handler	6
3.3 Datastore Info handler	6
3.4 Grouped By Columns Array handler	7
3.5 MOR handler	7
3.6 MOR Array handler	7
3.7 Sensor Array handler	7
3.8 Example	8
4 Configuration Tags Reference	18
4 VMware Collector Troubleshooting	21
1 User access	21
2 Counters	21

Preface

Purpose

The *VMware Collector* is a collecting component for *APG* that allows topology information and performance statistics to be collected directly from a VSphere/VCenter.

Intended Audience

This guide is intended for advanced *APG* Administrators that have a good knowledge of the *APG* collecting framework.

Prerequisites

The *VMware Collector* can only be used within an instance of *Collector Manager*. Thus, to be able to use this collecting component, you must have a working *Collector Manager* ready.

The *VMware Collector* is designed to integrate into the standard *APG* collecting/backend structure. Therefore, components such as filters, connectors and the backend can be used along with the *VMware Collector* to achieve the desired results.

Common Abbreviations and Acronyms

MOB	Managed Object Browser
MOR	Managed Object Reference (MoRef)

Document Organization

1. <i>Collector Manager Configuration</i>	How to configure the <i>Collector Manager</i> to include a <i>VMware Collector</i> .
2. <i>vSphere Documentation</i>	Important information about vSphere and its requirements.
3. <i>VMware Collector Configuration</i>	How to configure the <i>VMware Collector</i> itself.
4. <i>VMware Collector Troubleshooting</i>	Usefull information when troubleshooting the <i>VMware Collector</i> .

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.

1

***Collector Manager* Configuration**

To use a *VMware Collector* instance, please follow this *Collector Manager* configuration file example:

```
<collectors>
...
  <!-- This is the VMware collector -->
  <collector enabled="true" name="VMwareCollector" next="Backend" type="VMWare-Collector"
config="VMWare-Collector/Default/conf/vmware-collector.xml" />
...
</collectors>
```


2

vSphere Documentation

In order to correctly configure the *VMware Collector*, a good comprehension of the vSphere MOB is needed. If needed please consult the VMware documentation or [community blog](#).

1 Important information:

Your VCenter must be configured with a specific Statistic-Level. This setting is available in your vSphere client in the menu Administration → vCenter Server Settings → Statistics.

- Statistic-Level 3 get all Watch4net needed metric including virtual-interface IO and virtual disk latency.
- Statistic-Level 2 is also supported and will provide all possible reports, except virtual-interface IO and Virtual Disk Latency.

Additional Reports on vCenter Events and Task:

- In case you purchased the Watch4net Datastore license, you can install the event-processing components and then after importing your ReportPack into the Watch4net Portal, go in edit mode and under the report branch "All » ReportPack for VMware » Events & Tasks" validate that you have "Always show the node".

3

VMware Collector Configuration

This section is about the configuration file for the *VMware Collector*.

1 APG Values

The *VMware Collector* produces APG raw values with properties.

The `value` tags define decimal values that will be created.

The `property` tags define the properties that will be set for all values in the [Scope](#).

2 Scope

The configuration can contain multiple scopes defined by the `element` tags.

Properties defined by the `property` tags in one scope only apply to this scope and its children scopes.

3 VMware Collector context handlers

A context handler defines a specific way to access the content of a *Managed Object Browser*.

Each handlers expect a `path` where the specific object is.

If for any reason the object is the same as the current scope, use the `path "."`.

Here's the list of the available context handlers:

3.1 Array handler

A Generic Array handler defined by the `array` tag knows how to handle a simple dynamic data array. Each extracted properties will be joined in a single field of each [APG Values](#). The optional attribute `restrict-on` can be used to filter on one data type of element in the array. The content is expected to be the object Java class name.

This handler can contain the following sub handlers: [MOR handler](#), [Datastore handler](#), [Array handler](#), [MOR Array handler](#) and [Grouped By Columns Array handler](#).

3.2 Datastore handler

A Datastore handler defined by the `datastore` tag knows how to handle correctly a Datastore managed entity.

This handler can contain the following sub handlers: [Datastore Info handler](#), [MOR handler](#), [Array handler](#) and [MOR Array handler](#).

3.3 Datastore Info handler

A Datastore Info handler defined by the `datastore-info` tag is only available inside a [Datastore handler](#). It know how to access the restricted `info` field of a Datastore managed entity.

This handler can contain the following sub handlers: [Array handler](#) and [MOR Array handler](#).

3.4 Grouped By Columns Array handler

A Grouped By Columns Array handler defined by the `group-by-columns-array` tag will handle a simple dynamic data array but will extract at least one value. The values extracted will contain all the properties extracted from the fields of the same index in the array. The optional attribute `restrict-on` can be used to filter on one data type of element in the array. The content is expected to be the object Java class name.

This handler can contain the following sub handlers: [MOR handler](#), [Datastore handler](#), [Array handler](#), [MOR Array handler](#) and [Grouped By Columns Array handler](#).

3.5 MOR handler

A MOR handler defined by the `mor` tag knows of the dereference a *Managed Object Reference (MoRef)* which allows its fields to be accessible.

This handler can contain the following sub handlers: [MOR handler](#), [Datastore handler](#), [Array handler](#), [MOR Array handler](#) and [Grouped By Columns Array handler](#).

3.6 MOR Array handler

A MOR Array handler defined by the `mor-array` tag knows how to handle an array of *Managed Object Reference (MoRef)*.

If there's no value defined in this handler, the extracted properties will be joined in a single field in the [Scope](#) while the handler with values will handle each *Managed Object Reference (MoRef)* like a [Grouped By Columns Array handler](#).

This handler can contain the following sub handlers: [MOR handler](#), [Datastore handler](#), [Array handler](#), [MOR Array handler](#) and [Grouped By Columns Array handler](#).

3.7 Sensor Array handler

A Sensory Array handler defined by the `sensors` tag knows how to handle an array of sensors of a managed entity.

This handler cannot contain other context handlers.

3.8 Example

The following example is the default *VMware Collector* configuration. It is split into several sections (all in the same file) to improve clarity.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration xmlns="http://www.watch4net.com/VMWare-Collector" xmlns:xsi="http://www.w3.org
/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.watch4net.com/VMWare-Collector_.../vmware-collectorV2.xsd">

  <!-- The number of thread that will parse each element (MOB entity) in parallel -->
  <collecting-threads>4</collecting-threads>
  <!-- The time in seconds between to polling -->
  <polling-interval>300</polling-interval>
  <!-- The time in seconds between a full synchronization (update the properties, entities and
inventory cache -->
  <resync-interval>3600</resync-interval>
  <!-- The APG collecting group -->
  <collecting-group>group</collecting-group>
  <!-- The APG collecting source -->
  <source>VMwareCollector</source>

  <!-- One <connection> tag per actual vCenter connection in order to poll multiple vCenters
with the same polling configuration -->
  <connection timeout="Optional_connection_timeout_(will_be_at_10_seconds_by_default)">
    <!-- The vSphere service URL -->
    <url>https://RP_SYS_HOST/sdk/vimService</url>
    <!-- The MOB connection username -->
    <username>RP_SYS_USER</username>
    <!-- The MOB connection password. Can be encrypted -->
    <password>RP_SYS_PASS</password>
  </connection>

  <!-- The time before a connection will be retried once invalid credentials has been detected
(If omitted, 30 minutes will be used) -->
  <connection-retry-time>30m</connection-retry-time>
  <!-- The property separator when a single field contains an array -->
  <property-separator>|</property-separator>
```

Note:

- One or more <connection> elements may be declared in a configuration in order to poll several vCenter instances with the same collector.
- The older syntax, which allows the polling of only one vCenter instance per collector instance, is still supported for compatibility reasons. (The <url>, <username> and <password> elements are not contained in a <connection> element). Please update those configurations as soon as possible to enjoy the benefits of the new configuration model.

```
<!-- Polling every VMs on the MOB -->
<element id="VM" type="VirtualMachine" filter=".*" variable-id="deviceid_name">
```

An <element> defines a scope for a single MOB entity. It can contain sub element tags that define only a scope change, the polling will be done on the same element as the parent. It contains the following *attributes*:

- id: (Mandatory) Only for logging purpose.
- type: A valid managed type (VirtualMachine, HostSystem, Datastore, ...). **Mandatory and available only for top level elements.**
- filter: A valid Java regex used to reduce the number of entities polled for the given type. **Optional and available only for top level elements.**
- variable-id: (Optional) List of space separated properties used to create the variable ID for each values extracted in the scope. Default value is: source device part name.
- devtype: Can force the devtype property to other thing than the type attribute. **Optional and available only for top level elements.**

```
<value name="Heartbeat" value="sys.heartbeat" type="counter" rollup-type="summation"/>
<value name="Uptime" value="sys.uptime" type="counter" rollup-type="latest" />
<value name="cpuCount" value="config.hardware.numCPU" type="property-value" />
<value name="diskCount" value="summary.config.numVirtualDisks" type="property-value" />
```

An `<element>` usually contains `<value>` tags. `<value>` tags declare the list of extracted APG values from either MOB counters or MOB properties. A `<value>` contains the following *attributes*:

- name: (Mandatory) Resulting value of the name property.
- value: (Mandatory) The MOB path
- unit: (Optional) Defines the value of the unit property in the APG values
- type: (Optional) Used only when defining a value under an element, MOR or MOR array tags. Defines if the value is from a counter (type="counter") or a property (type="property-value"). Default is counter.
- instance: (Optional) Used only when the type is counter. It will define the behavior of the resulting counters when they are attached to multiple instance. Possible values for this attribute are:
 - both: Defines that the summary and children values will be generated.
 - children: Filters only on the children..
 - summary: Filters only on the summary which is the default behavior.
- instance-property: (Optional) Used only when the type is counter and when the counter is attached to multiple instance of the entity. This field defines which property in the APG values will have the MOB instance value as suffix.
- rollup-type: (Optional) Defines which counter to take when multiple sampling is available. The rollup-type is counter specific and the list of available sampling per counter should be available in the vCenter documentation. If not, see section 4 for more details. Available sampling are:
 - none: Counter is only keeping a count.
 - average: (Default) Counter is computing the average of the value.
 - minimum: Counter is keeping the minimum value.
 - maximum: Counter is keeping the maximum value.
 - summation: Counter is computing the sum of the value.
 - latest: Counter is keeping the last value.

```
<value name="Availability" value="runtime.powerState" type="property-value">
  <replace value="poweredOn" by="100" />
  <replace value="poweredOff" by="0" />
  <replace value="suspended" by="50" />
</value>
<value name="CurrentSnapshot" value="snapshot.currentSnapshot" type="property-value">
  <replace value="" by="0" />
  <replace value="com.vmware.*" by="1" type="regex"/>
</value>
```

The <value> tags can also contain <replace> tags, which are used to transform text to a numerical value. Matching on the polled value is done via a regular expression.

```
<property name="device" type="data" value="name" />
<property name="deviceid" type="data" value="config.instanceUuid" />

<property name="path" type="data" value="summary.config.vmPathName" />
<property name="status" type="data" value="summary.overallStatus">
  <replace value="green" by="Normal" />
  <replace value="yellow" by="Warning" />
  <replace value="red" by="Alert" />
</property>
```

The <property> is a Watch4Net property coming from either a MOB field or an hard coded value. A <property> contains the following *attributes*:

- name: (Mandatory) Resulting name of the new property.
- value: (Mandatory) Defines either the MOB path or the hard coded value if available. If the path is %MOT% for a data property, the result will be the actual Managed Object Type of the entity (represented by the Java class name).
- type: (Optional) Defines if the property comes from a MOB field (data) or an hard coded value. Default value is data.

A <property> can also contain a <replace> tag to substitute a polled value with another one (textual or numerical).

```

<property name="ip" type="data" value="guest.ipAddress" />
<property name="hostname" type="data" value="guest.hostName" />
<property name="devdesc" type="data" value="guest.guestFullName" />
<property name="toolvrs" type="data" value="guest.toolsVersionStatus" />
<property name="toolstat" type="data" value="guest.toolsRunningStatus">
  <replace value="guestToolsNotRunning" by="Not_Running" />
  <replace value="guestToolsNotInstalled" by="Not_Installed" />
  <replace value="guestToolsCurrent" by="Current" />
  <replace value="guestToolsRunning" by="Running" />
  <replace value="guestToolsNeedUpgrade" by="Need_Upgrade" />
  <replace value="guestToolsUnmanaged" by="Unmanaged" />
</property>

<mor path="parent">
  <property name="folder" type="data" value="name" />
</mor>
<mor path="runtime.host">
  <property name="hypervsr" type="data" value="name" />
</mor>

```

The <mor> defines that a needed Managed Object Reference is present in the scope and must be retrieved and cached.

```

<datastore path="datastore">
  <property name="datastor" value="name"/>
</datastore>

```

The <datastore> defines that a needed datastore is present in the scope and allow the retrieval of the content under info.vmps.

```

<element id="CPU" variable-id="deviceid_parttype_part_name">
  <value name="Ready" value="cpu.ready" type="counter" rollup-type="summation" />
  <value name="UsageMHz" value="cpu.usagemhz" type="counter" rollup-type="average" />
  <value name="CurrentUtilization" value="cpu.usage" type="counter"
    rollup-type="average" />
  <property name="parttype" type="hard-coded" value="Processor" />
  <property name="part" type="hard-coded" value="System" />
</element>

```

```
<element id="DISK" variable-id="deviceid_parttype_part_name">
  <grouped-by-columns-array path="guest.disk">
    <property name="part" value="diskPath" type="data" />
    <property name="parttype" value="FileSystem" type="hard-coded" />

    <value name="AvailableSpace" value="freeSpace" unit="Bytes" />
    <value name="StorageSize" value="capacity" unit="Bytes" />
  </grouped-by-columns-array>
</element>
```

The `<grouped-by-column-array>` defines that a needed dynamic data array is present in the scope and must extract the values with its properties defined by the index of each columns.

```
<element id="MAC" variable-id="deviceid_parttype_part_name">
  <grouped-by-columns-array path="guest.net">
    <property name="network" value="network" type="data" />
    <property name="part" value="macAddress" type="data" />
    <property name="parttype" value="Interface" type="hard-coded" />

    <value name="Availability" value="connected">
      <replace value="true" by="100" />
      <replace value="false" by="0" />
    </value>
  </grouped-by-columns-array>
</element>

<element id="FILE" variable-id="deviceid_parttype_part_name">
  <grouped-by-columns-array path="layoutEx.file">
    <property name="part" value="name" type="data" />
    <property name="parttype" value="File" type="hard-coded" />

    <value name="StorageSize" value="size" unit="Bytes" />
  </grouped-by-columns-array>
</element>
```

```
<element id="DATASTORE" variable-id="deviceid_parttype_part_name">
  <grouped-by-columns-array path="storage.perDatastoreUsage">
    <property name="parttype" value="Datastore" type="hard-coded" />

    <mor path="datastore">
      <property name="part" value="name" type="data" />
    </mor>

    <value name="UsedSpace" value="committed" unit="Bytes" />
  </grouped-by-columns-array>
</element>

<element id="MEMORY" variable-id="deviceid_parttype_part_name">
  <value name="CurrentUtilization" value="mem.usage" type="counter" />
  <value name="Overhead" value="mem.overhead" type="counter" />
  <value name="Consumed" value="mem.consumed" type="counter" />
  <value name="Balloon" value="mem.vmmemctl" type="counter" />
  <value name="Active" value="mem.active" type="counter" />
  <value name="TotalMemory" type="property-value" value="config.hardware.memoryMB"
    unit="MB" />

  <property name="parttype" type="hard-coded" value="Memory" />
  <property name="part" type="hard-coded" value="System" />
</element>

<element id="SWAP" variable-id="deviceid_parttype_part_name">
  <value name="SwapIn" value="mem.swapinRate" type="counter" />
  <value name="SwapOut" value="mem.swapoutRate" />

  <property name="parttype" type="hard-coded" value="Memory" />
  <property name="part" type="hard-coded" value="Swap" />
</element>

<element id="INTERFACE" variable-id="deviceid_parttype_part_name">
  <value name="Usage" value="net.usage" type="counter" />

  <property name="parttype" type="hard-coded" value="Interface" />
  <property name="part" type="hard-coded" value="System" />
</element>
```

```

<element id="FILESYSTEM" variable-id="deviceid_parttype_part_name">
  <value name="Usage" value="disk.usage" type="counter" />

  <property name="parttype" type="hard-coded" value="FileSystem" />
  <property name="part" type="hard-coded" value="System" />
</element>
</element>

<!-- Polling all the datastores on the MOB -->
<element id="DATASTORE" type="Datastore" filter=".*">
  <datastore path=".">
    <property name="parttype" value="Datastore" type="hard-coded"/>
    <property name="devtype" value="Datacenter" type="hard-coded"/>
    <property name="part" value="name" type="data"/>

    <property name="status" value="overallStatus" type="data"/>
    <property name="shared" value="summary.multipleHostAccess" type="data"/>

    <mor path="parent">
      <mor path="parent">
        <property name="device" value="name" type="data"/>
      </mor>
    </mor>

    <array path="host">
      <mor path="key">
        <property name="esx" value="name"/>
      </mor>
    </array>

    <mor-array path="vm">
      <property name="virtual-machines" value="name"/>
    </mor-array>
  </datastore>
</element>

```

The <array> defines that a needed dynamic data array is present in the scope. The <mor-array> defines that a needed mor array is present in the scope and must be retrieved and cached.

```

<value name="capacity" value="summary.capacity"/>
<value name="freeSpace" value="summary.freeSpace"/>
<value name="uncommitted" value="summary.uncommitted"/>

<datastore-info path="info">
  <array path="vmfs.extent">
    <property name="lun-serial-number" value="diskName" />
  </array>
</datastore-info>
</datastore>
</element>

<!-- Polling all the ESXs on the MOB -->
<element id="ESX" type="HostSystem" filter=".*" variable-id="devtype_device_parttype_part_
name">
  <property name="product" type="data" value="config.product.fullName" />
  <property name="maintmod" type="data" value="runtime.inMaintenanceMode" />

  <property name="model" type="data" value="hardware.systemInfo.model" />
  <property name="vendor" type="data" value="hardware.systemInfo.vendor" />
  <property name="deviceid" type="data" value="hardware.systemInfo.uuid" />
  <property name="PwPolicy" type="data" value="hardware.cpuPowerManagementInfo.currentPolicy"
/>

  <property name="reboot" type="data" value="summary.rebootRequired" />
  <property name="devdesc" type="data" value="summary.config.product.fullName" />
  <property name="hypervsr" type="data" value="summary.config.name" />
  <property name="status" type="data" value="summary.overallStatus">
    <replace value="green" by="Normal" />
    <replace value="yellow" by="Warning" />
    <replace value="red" by="Alert" />
  </property>
  <property name="vmotion" type="data" value="summary.config.vmotionEnabled">
    <replace value="true" by="Yes" />
    <replace value="false" by="No" />
  </property>

  <mor path="parent">
    <property name="type" value="%MOT%" type="data"/>
    <property name="domain" type="data" value="name" />
  </mor>

```

```

<element id="CPU" variable-id="devtype_device_parttype_part_name">
  <value name="CurrentUtilization" value="cpu.usage" type="counter" />
  <value name="ProcessorSize" type="property-value" value="hardware.cpuInfo.hz" />

  <property name="parttype" type="hard-coded" value="Processor" />
  <property name="part" type="hard-coded" value="System" />
  <property name="partdesc" type="data" value="summary.hardware.cpuModel" />
</element>

<element id="MEMORY" variable-id="devtype_device_parttype_part_name">
  <value name="CurrentUtilization" value="mem.usage" type="counter" />
  <value name="TotalMemory" type="property-value" value="summary.hardware.memorySize" />

  <property name="parttype" type="hard-coded" value="Memory" />
  <property name="part" type="hard-coded" value="System" />
</element>

<element id="COUNT" variable-id="deviceid_name">
  <value name="nicCount" type="property-value" value="summary.hardware.numNics" />
  <value name="cpuCount" type="property-value" value="hardware.cpuInfo.numCpuPackages" />
  <value name="coreCount" type="property-value" value="hardware.cpuInfo.numCpuCores" />
</element>

<element id="SENSORS" variable-id="devtype_device_parttype_part_name">
  <property name="parttype" type="hard-coded" value="Sensor" />
  <property name="part" type="hard-coded" value="System" />

  <sensors path="runtime.healthSystemRuntime.systemHealthInfo.numericSensorInfo">
    <name-regex used-group="1">([\w\s]+\w)*\s*[-](.*)</name-regex>
    <sensor-name-property>name</sensor-name-property>
    <value value="power" unit="." />
    <value value="voltage" unit="." />
    <value value="fan" unit="." />
  </sensors>
</element>
</element>
</configuration>

```

The <sensors> defines the list of desired sensors on the current entity.

4 Configuration Tags Reference

- **element**: An element defines a scope for a single MOB entity. It can contains sub element tags that define only a scope change, the polling will be done on the same element as the parent

An element contains the following *attributes*:

- id: (Mandatory) Only for logging purpose.
- variable-id: (Optional) List of space separated properties used to create the variable ID for each values extracted in the scope. Default val
- allow-missing-property: (Optional) If it's possible that an enrichment property (not part of the variable ID or used as value) can be missing from some entities in the list, it's possible allow those properties to be missing in the resulting raw values instead of skipping completely the element. This feature should only be used for this reason and not as a possible safeguard..
- type: A valid managed type (VirtualMachine, HostSystem, Datastore, ...). **Mandatory and available only for top level elements.**
- filter: A valid Java regex used to reduce the number of entities polled for the given type. **Optional and available only for top level elements.**
- devtype: Can force the devtype property to other thing that the type attribute. **Optional and available only for top level elements.**

An element can contains the following optional *tags*:

- value: Defines a list of extracted APG values from either MOB counters or MOB properties.
- property: Defines the list of APG properties from either an extracted MOB field or an hard coded value.
- mor: Defines that a needed Managed Object Reference is present in the scope and must be retrieved and cached.
- datastore: Defines that a needed datastore is present in the scope and allow the retrieval of the content under info.vmfs.
- array: Defines that a needed dynamic data array is present in the scope.
- mor-array: Defines that a needed mor array is present in the scope and must be retrieved and cached.
- grouped-by-column-array: Defines that a needed dynamic data array is present in the scope and must extract the values with its properties defined by the index of each columns.

- sensors: Defines the list of desired sensors on the current entity.
- **value:** A value can either comes from an extracted MOB property or a MOB performance counter. Counters are only available under an element, MOR or MOR array tags.
A value contains the following *attributes*:
 - name: (Mandatory) Resulting value of the name property.
 - value: (Mandatory) The MOB path
 - unit: (Optional) Defines the value of the unit property in the APG values
 - type: (Optional) Used only when defining a value under an element, MOR or MOR array tags. Defines if the value is from a counter (type="counter") or a property (type="property-value"). Default is counter.
 - instance: (Optional) Used only when the type is counter. It will define the behavior of the resulting counters when they are attached to multiple instance. Possible values for this attribute are:
 - * both: Defines that the summary and children values will be generated.
 - * children: Filters only on the children..
 - * summary: Filters only on the summary which is the default behavior.
 - instance-property: (Optional) Used only when the type is counter and when the counter is attached to multiple instance of the entity. This field defines which property in the APG values will have the MOB instance value as suffix.
 - rollup-type: (Optional) Defines which counter to take when multiple sampling is available. The rollup-type is counter specific and the list of available sampling per counter should be available in the vCenter documentation. If not, see section 4 for more details. Available sampling are:
 - * none: Counter is only keeping a count.
 - * average: (Default) Counter is computing the average of the value.
 - * minimum: Counter is keeping the minimum value.
 - * maximum: Counter is keeping the maximum value.
 - * summation: Counter is computing the sum of the value.
 - * latest: Counter is keeping the last value.
- **property:** An APG property coming from either a MOB field or an hard coded value.
A property contains the following *attributes*:

- name: (Mandatory) Resulting name of the new property.
- value: (Mandatory) Defines either the MOB path or the hard coded value if available. If the path is %MOT% for a data property, the result will be the actual Managed Object Type of the entity (represented by the Java class name).
- type: (Optional) Defines if the property comes from a MOB field (data) or an hard coded value. Default value is data.
- **replace:** Both the value and property tags can nest replacements.
A replace tag works with the following *attributes*:
 - value: The perfect match for a value or property that must be replaced.
 - by: The actual replacement. Must be a number underneath a value tag.
 - type: (Optional) The type of replacer. Possible values are: equals (default) for stricly equals comparison or regex for pattern matching.

4

***VMware Collector* Troubleshooting**

1 User access

Polling user rights should be set to FULL Read-Only access in order to simplify troubleshooting in the case of missing information.

2 Counters

If for any reasons a configured counters doesn't appear in the resulting values, it's generally comes from one of the following causes:

- The counter statistics level may be misconfigured (i.e. set to a low value to reduce the vCenter workload). Check and increase this level as required.
- The polled counter values are the 5 minutes interval. Enable them if they are not available.
- A configured rollup-type may not be available for specific counter. In order to find out which counters are available for each entity, filter the element for a specific entity and run the collector with the logs set to FINEST to get a complete list of the available counters.