ISSPI Support Note 0006

Subject: What you must know about source level filters in VISPI policies for VMware monitoring through the OAVA

Severity: Important
Classification: Customer viewable
Audience: Customers, Software Support, Consultants

Release history
2015/04/13 First release (thierry.ledent@hp.com)

Get the latest version of this document at:
- Software Support Online (KM01512443)
- Operations Manager for Unix Support Customer Forum
- Operations Manager for Windows Support Customer Forum
- HP Connections (HP internal)

ABSTRACT

The VISPI policies for VMware monitoring through the OAVA use a feature known as "**source level filters**" that is specific to OAVA. This feature can dramatically improve the performance of VISPI policies, but it also introduces new challenges that, if not mastered, can lead to missing performance alerts from the VISPI policies. This support note explains what you need to know to avoid the pitfalls. It is divided in three parts:

1. Source level filters explained
2. Challenges introduced by source level filters
3. Recommendations for a good trade-off between performance and reliability

1. Source level filters explained

VISPI provides measurement threshold policies for OAVA to monitor VMware performance. These policies implement an alerting logic that uses performance metrics collected by OAVA to determine if the server is experiencing a performance bottleneck that justifies an alert message. Let us review the concepts of instances, classes and sources before explaining source level filters.

1.1 Instances and classes

OAVA collects metrics for ESXi hosts, virtual machines, clusters, resource pools, etc ... Each host, virtual machine, cluster, resource pool, etc... is stored in the OAVA database as a separate **instance** along with its own metrics. These instances are organized in the OAVA database by classes (a.k.a. object):

- Class Node contains instances of ESXi hosts, virtual machine and templates
- Class Cluster contains instances of clusters
- Class Datastore contains instances of datastores
- Class VirtualApp contains instances of virtual appliances

- Class Datacenter contains instances of data centers
- Class ResourcePool contains instances of resource pools

When OAVA runs a VISPI measurement threshold policy, it loops through all the instances of a specific class to execute the alerting logic.

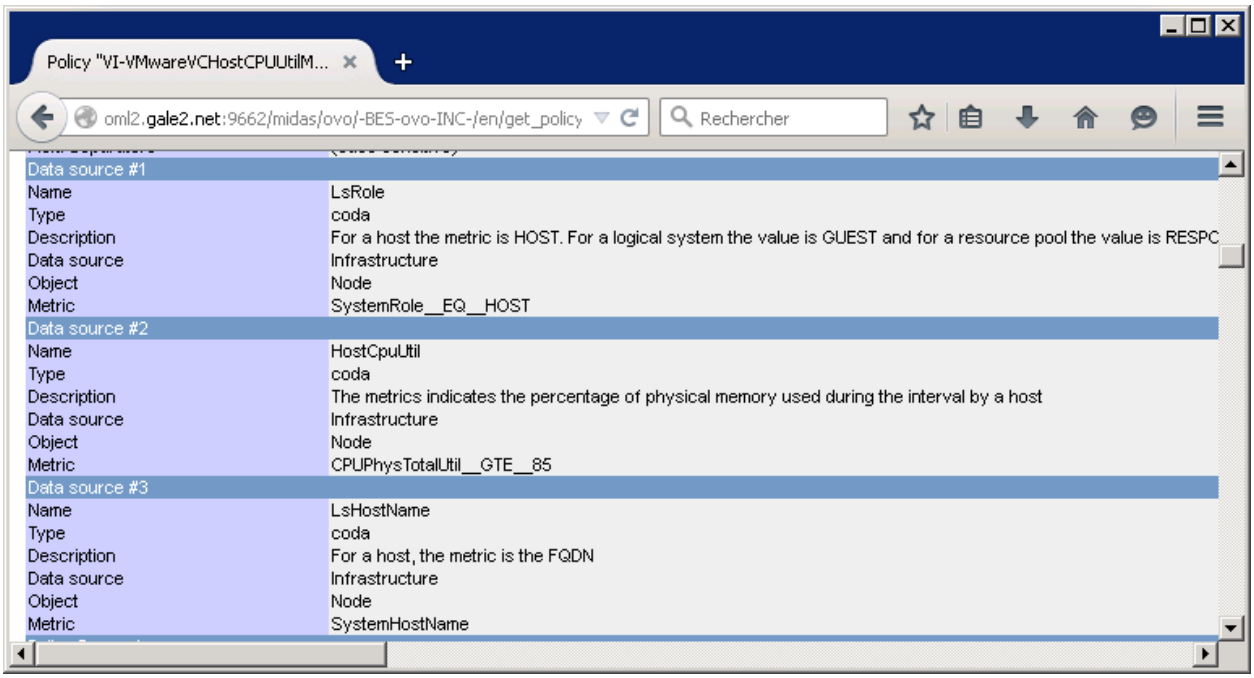
1.2 Sources

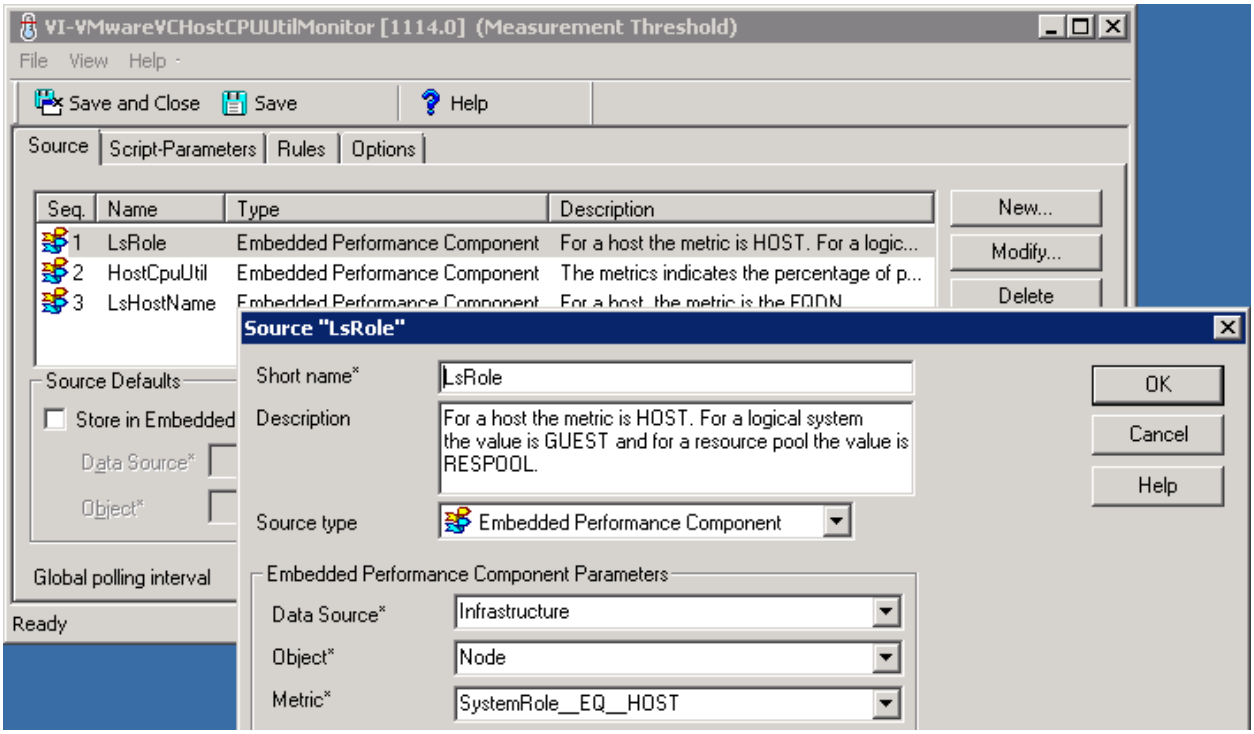
The "sources" are the parts of the policy that describe where the policy obtains the metric values that it will use for its alerting logic. Let us take the VISPI policy VI-VMwareVCHostCPUUtilMonitor as example as it is particularly simple.

The alerting logic of this policy, at a high level, is:

```
For each instance of class Node:
{
  If this is an ESXi host (checks the metric of source LsRole)
  {
    If CPU utilization is above threshold (checks the metric of source HostCpuUtil)
    {
      Send an alert
    }
  }
}
```

Below screenshots show the definition of sources for this policy in OML and in OMW:





As we can see above, this policy defines three sources:

- Source LsRole is based on the metric Infrastructure/Node/SystemRole
- Source HostCpuUtil is based on the metric Infrastructure/Node/CPUPhysTotalUtil
- Source LsHostName is based on the metric Infrastructure/Node/SystemHostName

The source LsHostName does not participate to the alerting logic, but it is configured to provide access to the node hostname for generating meaningful alerts.

1.3 Source level filters

The policy VI-VMwareVCHostCPUUtilMonitor monitors the CPU utilization of ESXi hosts. For this, it loops through all instances of the class Node. As explained earlier, this class contains hosts, virtual machines and templates. So, for each instance, the policy needs to check if the instance is a host before checking the instance's CPU utilization (as described in the alerting logic above). In a typical VMware environment, there will be many more virtual machines than hosts, so that the policy will be looping through many nodes that are actually not of interest. This adds significantly to the processing time of the policy.

Source level filters enable to reduce this processing time by presenting only a filtered set of instances to the policy. Looking closer at the syntax of the LsRole source in the above screenshots, we can see that it is defined as:

```
SystemRole__EQ__HOST
```

This is a special syntax that the OAVA interprets as: pass only instances to the policy that have a metric value of SystemRole equal to "HOST". With this the policy alerting logic will loop through a significantly

reduced list of instances. Additionally, the initial test for for ESXi hosts can be removed since the policy is guaranteed to be presented only instances that are ESXi hosts. This brings a significant performance gain for the policy.

Similarly, the source HostCpuUtil is defined as:

```
CPUPhysTotalUtil__GTE__85
```

The OAVA interprets this as: pass only instances to the policy that have a metric value of CPUPhysTotalUtil greater or equal to 85%. This will again reduce the list of instances that are presented to the policy.

Practically, OAVA will run a SQL statement similar to:

```
select SystemRole,CPUPhysTotalUtil,SystemHostName from Nodes
where SystemRole='HOST' and CPUPhysTotalUtil>=85;
```

The result of this statement will be processed by the policy alerting logic.

2. Challenges introduced by source level filters

The policy VI-VMwareVCHostCPUUtilMonitor defines following default thresholds:

- Major: 95%
- Minor: 90%
- Warning: 85%

Let's assume now that we want to lower the warning threshold to 80%. On top of modifying the threshold value, we must remember to also adapt the source level filter of source HostCpuUtil to:

```
CPUPhysTotalUtil__GTE__80
```

If we forget to adapt the source level filter, OAVA will continue to pass only instances to the policy that have a metric value of CPUPhysTotalUtil greater or equal to 85%. In other words, no alert message will be sent for hosts until they reach 85% CPU utilization, despite setting the warning threshold to 80%.

Continuing with this example, another particularity happens when the host CPU utilization drops below the warning threshold after a warning alert was sent. Since the CPU utilization is now below the level defined in the source level filter, the instance is no longer passed to the policy, and no reset message will be sent. We could lower the source level filter value to, say 50%, e.g.:

```
CPUPhysTotalUtil__GTE__50
```

But if the CPU utilization drops in a short time from above 80% to below 50%, we may still miss the reset message. Practically, we cannot guarantee that a reset message will be sent if a filter is defined on this source.

3. Recommendations for a good trade-off between performance and reliability

Given the above challenges of source level filters, it is recommended to remove numerical source level filters. For the policy VI-VMwareVCHostCPUUtilMonitor that we use as example in this support note, the definition of source HostCpuUtil should become:

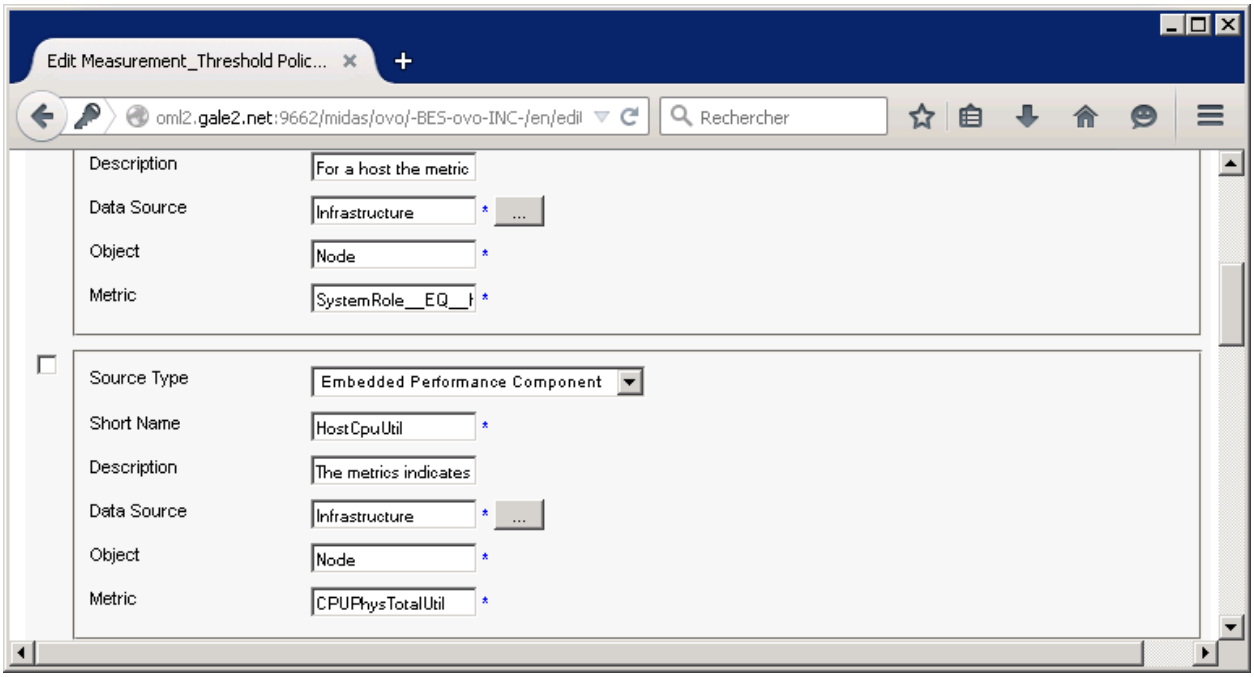
- Data Source: `Infrastructure`
- Object: `Node`
- Metric: `CPUPhysTotalUtil` (instead of `CPUPhysTotalUtil__GTE__85`)

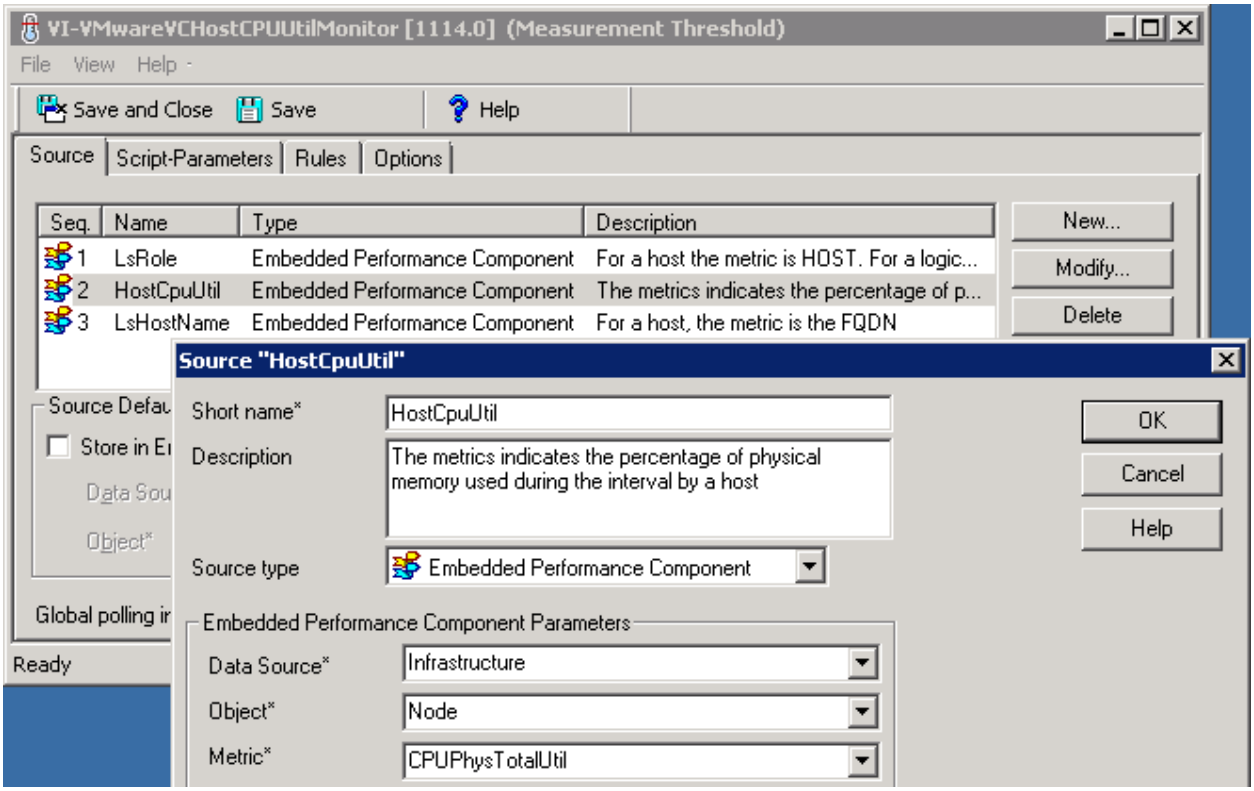
IMPORTANT: non-numerical source level filters, such as below, should NOT be removed:

`SystemRole__EQ__HOST`

This type of filter does not introduce any particular challenge and enables to significantly improve the performance of the policy. Additionally, the policy alerting logic relies on OAVA to only pass instances that pass this filter.

Below screenshots illustrate the recommended change for the numerical source level filter in souce HostCpuUtil for the policy VI-VMwareVCHostCPUUtilMonitor in OML and OMW:





The policy VI-VMwareVCHostCPUUtilMonitor was used as simple example in this support note. The same recommendations apply to the other VISPI policies for VMware monitoring through OAVA. The policies listed below contain numerical source level filters in VISPI version 11.13 and 11.14:

- VI-VMwareVCClusterCPUPerformanceMonitor
- VI-VMwareVCClusterMemoryPerformanceMonitor
- VI-VMwareVCGuestCPUPerformanceMonitor
- VI-VMwareVCGuestLatencyMonitor
- VI-VMwareVCGuestMemoryPerformanceMonitor
- VI-VMwareVCHostCPUSaturationMonitor
- VI-VMwareVCHostCPUUtilMonitor
- VI-VMwareVCHostMemUtilMonitor
- VI-VMwareVCRespoolCPUUtilMonitor

Note that there may be multiple numerical source level filters in the same policy. All the sources of the above policies should be reviewed.