Authors:
Steven Newhouse, Microsoft
Murali Krishna K, Altair

24 June 2009

HPCBP Advanced Filter Extension

Status of this Memo

This memo provides information to the Grid community regarding the specification of the HPC Basic Profile (HPCBP) Advanced Filter Extension. Distribution is unlimited.

Copyright Notice

Abstract

This document defines the Advanced Filter extension – an alternative to the Basic Filter element described in the HPC Basic Profile 1.0 specification. The Basic Filter provides only an ‘on’ or ‘off’ approach to returning information about the activities or resources operating within the Basic Execution Service container. The Advanced Filter extension provides more flexibility to the client in returning information from an HPC Basic Profile complaint endpoint.

Contents

Abstract 1
1 Introduction 3
2 Notational Conventions 3
3 Advanced Filter Extension 3
3.1 Request 3
3.2 Response 6
3.3 Faults 6
4 Conformance Compliance 6
5 Security Considerations 6
6 Author Information 7
7 Contributors & Acknowledgements 7
8 Full Copyright Notice 7
9 Intellectual Property Statement 7
10 Normative References 8
11 Appendix A: AdvancedFilter Schema Definition 9

1 Introduction

The HPC Basic Profile Advanced Filter extension is a document that is used to describe an extension to the HPC Basic Profile (HPCBP) [HPCBP10] in order to provide greater control to the client as to the information returned from the 'GetFactoryAttributesDocument' operation. Currently, the Basic Filter element defined in the HPCBP specification does not meet the needs of users operating on HPCBP services that have many contained resources or many activities operating within the container. The greatest concern is that without improved filtering capability the size of the returned XML documents can become very large.

2 Notational Conventions

The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," "RECOMMENDED," "MAY," and "OPTIONAL" are to be interpreted as described in RFC-2119 [RFC 2119].

The document refers to an "HPCBP Advanced Filter extension" as a "Compliant system".

This specification uses namespace prefixes throughout; they are listed in Table 2-1. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

Table 2-1: Prefixes and namespaces used in this specification.

Prefix	Namespace
xsd	http://www.w3.org/2001/XMLSchema
jsdl	http://schemas.ggf.org/jsdl/2005/11/jsdl
bes	http://schemas.ggf.org/bes/2006/08/bes-factory
hpcp-bp	http://schemas.ogf.org/hpcp/2007/01/bp
hpcp-af	http://schemas.ogf.org/hpcp/2007/11/af

3 Advanced Filter Extension

3.1 Request

The filtering declarations in the Advanced Filter extension are applied cumulatively to the list of activities recorded by the underlying BES container before those activities that meet the filtering criteria are returned by the bes:GetFactoryAttributesDocument operation in its response document. Declarations of the same type (e.g. UserName or State) are 'OR'ed together while those of different type are 'AND'ed. For instance, the following example document fragment would return all activities in the Pending OR Running state, AND that have Bob as the designated executing user account. It would not return all activities from Bob and all Pending and Running activities within the container.

```
<hpcp-af:AdvancedFilter>
   <hpcp-af:UserName>Bob</hpcp-af:UserName>
   <hpcp-af:State>Pending</hpcp-af:State>
   <hpcp-af:State>Running</hpcp-af:State>
</hpcp-af:AdvancedFilter>
```

The AdvancedFilter element is placed as an extensibility element in the GetFactoryAttributesDocument request message.

If both the BasicFilter and AdvancedFilter elements are specified, then the semantics defined in the BasicFilter still apply, i.e. if the activities filter is specified as false then no activities are returned regardless of the contents of the AdvancedFilter element, but if the activities filter is true then the directives in the AdvancedFilter are applied before any activities are returned to the client.

If the BasicFilter is not specified then the AdvancedFilter directives are applied to all activities and resources.

```
<hpcp-af:AdvancedFilter>
   <hpcp-af:UserName>xsd:string</hpcp-af:UserName>*
   <hpcp-af:Owner>xsd:string</hpcp-af:Owner>*
   <hpcp-af:State>bes:ActivityStateEnumeration</hpcp-af:State>*
   <hpcp-af:ActivityIdRange start=”xsd:long” end=”xsd:long” >*
   <hpcp-af:ActivityId>xsd:string</ActivityId>*
   <hpcp-af:DateTimeRange from=”xsd:datetime” until=”xsd:datetime” >*
   <hpcp-af:CompactResources>?
   <hpcp-af:UsageRecordFormat>?
   <hpcp-af:NodeStatus>xsd:string</hpcp-af:NodeStatus>?
</hpcp-af:AdvancedFilter>
```

/hpcp-af:UserName

Only activities that will be executed under the specified user account name are returned in the activities list. Generally, the name of the executing user account will be derived when the activity is created from the identity used to authenticate the client to the HPCBP service. An alternative identity may be provided within the HPCP Application element or through the ActivityCredential schema and could be used by the HPCBP service to instantiate the activity. An implementation may choose to apply an additional level of filtering to any specified UserName filtering request in the interest of privacy, i.e. only activities executing under the requested user account may be returned to that user by the Advanced Filter specification. If requested activities are not being returned to the client because of insufficient authorization then a bes:NotAuthorisedFault MUST be generated, i.e. if user Bob requests activities relating to Bob and Alice and it is the policy of the implementation not to show Alice’s activities to Bob then the bes:NotAuthorisedFault MUST be generated and no output MUST be returned. Multiple UserName elements MAY be specified in a single AdvancedFilter element.

/hpcp-af:Owner

Only activities that have been submitted by the specified identity, the owner of the activity, are returned in the activities list. The identity of the owner will generally be taken from the authenticated credentials provided by the client during activity creation. An implementation may choose to apply an additional level of filtering to any specified Owner filtering request in the interest of privacy, i.e. only activities submitted under by the requested owner may be returned to that individual by the Advanced Filter specification. If requested activities are not being returned to the client because of insufficient authorization then a bes:NotAuthorisedFault MUST be generated, i.e. if user Bob requests activities relating to Bob and Alice and it is the policy of the implementation not to show Alice’s activities to Bob then the bes:NotAuthorisedFault MUST be generated and no output MUST be returned. Multiple Owner elements MAY be specified in a single AdvancedFilter element.

/hpcp-af:State

Only activities in the specified bes:ActivityStateEnumeration will be returned to the client. Multiple State elements MAY be specified in a single within a single AdvancedFilter element.

/hpcp-af:ActivityIdRange

Most job scheduling systems that are accessed through the HPCBP use a numerical job identifier. These are frequently returned as wsa:ReferenceParameters in the Activity EPR returned by the bes:CreateActivity operation. These activity identifiers can be used to constrain the activities returned by the HPCBP by specifying the inclusive @start and @end attributes in the ActivityIdRange element. Multiple ActivityIdRange elements MAY be specified in a single within a single AdvancedFilter element.

/hpcp-af:ActivityId

Multiple ActivityId elements MAY be specified within the AdvancedFilter element. The enclosed xsd:string is used to explicitly search for a local job reference. This enables clients to access non-numeric activity references (as is not the case in the ActivityIdRange) or to request the details of activities through non-sequential numerical job identifiers.

/hpcp-af:DateTimeRange

This directive specifies that only activities that were submitted into the container between the date and time range defined by the @from and @until attributes in the DateTimeRange element. Multiple DateTimeRange elements MAY be specified in a single within a single AdvancedFilter element. The data type of the from and until attributes is a composite of a Gregorian calendar date and a time and time zone on that day.

/hpcp-af:CompactResources

If specified the contained resources are returned in a compact format defined below. Identical resources (other than the resource name) that would otherwise be returned in multiple ConstrainedResource elements are combined and returned within a single ConstrainedResource element with all of the resource names of the identical resources placed in a comma separated list in the ResourceName element.

/hpcp-af:UsageRecordFormat

If specified, the return message will include a compliant JobUsageRecord [URF10] element for any activities found using the other activity related search filters.

/hpcp-af:NodeStatus

If specified with no body text then all nodes are to be reported upon. If there is text in the body of the element it provides a comma separated list of resource node name for which the dynamic node status is being requested. The response is placed in a <hpcp-af:DynamicNodeStatus> element with child elements of the <hpcbp-af:NodeStatus> type containing each node's dynamic status. If a node is not on-line and available for running jobs it is not reported. The attributes reported on a node are:

- Memory – Used & Free Memory (in Mb)
- CPU – Used & Free compute resources and overall CPU Load (%). Generally, one activity on a node (i.e. task) will occupy one processing core or job slot. This attribute could be interpreted as how many jobs are running on a node and how many more would the scheduler allow to run on the node.
- Activity – The string identifier of the activity executing on the node. Multiple activities may be running on the node (signified by multiple elements) and an activity may appear as running on multiple nodes if it is a multi-task activity or represents a parallel job.

The following example provides information on two nodes in a cluster, each with 2GB of total memory and 8 processing cores. There are three activities running on 'ComputeNode01' and 1 activities on 'ComputeNode07'. The identifier for each activity is made up of their local numerical job scheduler id and that schedulers domain name.

```
<hpcp-af:DynamicNodeStatus>
  <hpcp-af:NodeStatus name="ComputeNode01">
    <hpcp-af:Memory used="512" free="1536" />
    <hpcp-af:CPU used="3" free="5" load="18" />
    <hpcp-af:Activity id="2304@enterprise.com" />
    <hpcp-af:Activity id="2316@enterprise.com" />
    <hpcp-af:Activity id="220@academic.org" />
  </hpcp-af:NodeStatus>
  <hpcp-af:NodeStatus name="ComputeNode07">
    <hpcp-af:Memory used="512" free="1536" />
    <hpcp-af:CPU used="1" free="7" load="5" />
    <hpcp-af:Activity id="2306@academic.org" />
  </hpcp-af:NodeStatus>
</hpcp-af:DynamicNodeStatus>
```

3.2 Response

The response from this extension is incorporated into the existing schemas defined within the HPC Basic Profile 1.0. No further changes are required.

3.3 Faults

The following service specific faults may be generated by the Advanced Filter extension:

- bes:NotAuthorisedFault – As described earlier in Section 3.1. An implementation MUST generate this fault if it is unable, due to privacy or other concerns, to not provide the information relating to a particular Owner or UserName filtering request as the entity requesting the information is not the specified individual. For some implementations this may not be a concern.
- bes:UnsupportedFeatureFault – An implementation MUST generate this fault if the filtering element specified in the AdvancedFilter element is not supported by this implementation. For instance an implementation that uses GUIDs to identify its activities as opposed to numerical sequence may not support filtering requests around the ActivityIdRange element.

4 Conformance Compliance

Support of this Advanced Filter extension is declared by returning the following XML segment within the BES:GetFactoryAttributesDocument:

```
<bes:BESExtension>
http://schemas.ogf.org/hpcp/2007/11/bp/AdvancedFilter
</bes:BESExtension>
```

5 Security Considerations

This extension is used as part of the HPCBP specification and would be initiated through an endpoint using security mechanisms defined in the HPCBP document. The privacy concerns that arise in providing information to one user about activities undertaken by other users have already been covered.

6 Author Information

Steven Newhouse
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
steven.newhouse@microsoft.com

Murali Krishna K
Altair Engineering, Inc.
1820 E. Big Beaver
Troy, MI 48083
USA
murali.konnipati@altair.com

7 Contributors & Acknowledgements

We gratefully acknowledge the contributions and discussions made to this specification by Chris Smith, Glenn Wasson, Donal Fellows and other members of the HPC Profile Working Group.

We would like to thank the people who took the time to read and comment on earlier drafts. Their comments were valuable in helping us improve the readability and accuracy of this document.

8 Full Copyright Notice

9 Intellectual Property Statement

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director (see contact information at OGF website).

10 Normative References

[RFC 2119] Bradner, S. *Key words for use in RFCs to Indicate Requirement Levels*. Internet Engineering Task Force, RFC 2119, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt

[HPCBP10] Available at http://www.ggf.org/documents/GFD.114.pdf.

[URF10] Available at http://www.ogf.org/documents/GFD.98.pdf

11 Appendix A: AdvancedFilter Schema Definition

```
<xsd:schema id="hpcp-advancedfilter"
               targetNamespace="http://schemas.ogf.org/hpcp/2007/11/bp/AdvancedFilter"
               elementFormDefault="qualified"
               xmlns="http://schemas.ogf.org/hpcp/2007/11/bp/AdvancedFilter"
               xmlns:bes="http://schemas.ggf.org/bes/2006/08/bes-factory"
               xmlns:wsa="http://www.w3.org/2005/08/addressing"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema">


  <xsd:import namespace="http://schemas.ggf.org/bes/2006/08/bes-factory"
schemaLocation="./bes-factory.xsd" />

  <xsd:complexType name="AdvancedFilterType">
    <xsd:sequence>
      <xsd:element name="UserName" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
      <xsd:element name="Owner" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="State" type="bes:ActivityStateEnumeration" minOccurs="0"
maxOccurs="unbounded" />
      <xsd:element name="ActivityIdRange" type="ActivityIdRangeType" minOccurs="0"
maxOccurs="unbounded" />
     <xsd:element name="ActivityId" type="xsd:string" minOccurs="0"
maxOccurs="unbounded" />
      <xsd:element name="DateTimeRange" type="DateTimeRangeType" minOccurs="0"
maxOccurs="unbounded" />
      <xsd:element name="CompactResources" type="xsd:string" minOccurs="0"
maxOccurs="1" />
    <xsd:element name="UsageRecordFormat" type="xsd:string" minOccurs="0" maxOccurs="1"
/>
      <xsd:element name="NodeStatus" type="xsd:string" minOccurs="0" maxOccurs="1" />
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>

  <xsd:element name="AdvancedFilter" type="AdvancedFilterType" />

  <xsd:complexType name="DateTimeRangeType">
    <xsd:attribute name="from" type="xsd:dateTime" use="required" />
    <xsd:attribute name="until" type="xsd:dateTime" use="required" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
  </xsd:complexType>

  <xsd:complexType name="ActivityIdRangeType">
    <xsd:attribute name="start" type="xsd:long" use="required" />
    <xsd:attribute name="end" type="xsd:long" use="required" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
  </xsd:complexType>


  <xsd:complexType name="NodeStatusType">
    <xsd:sequence>
      <xsd:element name="Memory" type="MemoryType" minOccurs="1" maxOccurs="1" />
      <xsd:element name="CPU" type="CPUType" minOccurs="1" maxOccurs="1" />
      <xsd:element name="Job" type="ActivityIdType" minOccurs="0" maxOccurs="unbounded"
/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required" />
  </xsd:complexType>

  <xsd:complexType name="MemoryType">
    <xsd:attribute name="used" type="xsd:long" use="required" />
    <xsd:attribute name="free" type="xsd:long" use="required" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
```

`

```
  </xsd:complexType>

  <xsd:complexType name="CPUType">
    <xsd:attribute name="used" type="xsd:long" use="required" />
    <xsd:attribute name="free" type="xsd:long" use="required" />
    <xsd:attribute name="load" type="xsd:long" use="required" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
  </xsd:complexType>

  <xsd:complexType name="ActivityIdType">
    <xsd:attribute name="id" type="xsd:long" use="required" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
  </xsd:complexType>

  <xsd:complexType name="DynamicNodeStatusType">
    <xsd:sequence>
      <xsd:element name="NodeStatus" type="NodeStatusType" minOccurs="0"
maxOccurs="unbounded" />
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```