

Microsoft Baseline Configuration Analyzer Model Authoring Guide

Microsoft Corporation

Published: November 2012

Abstract

This guide describes how to design and create models and submodels for Microsoft Baseline Configuration Analyzer (MBCA) 2.0.



Copyright information

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2012 Microsoft Corporation. All rights reserved.

Windows, Windows NT, Windows Server, Windows 7, and Windows Vista are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

[Microsoft Baseline Configuration Analyzer Model Authoring Guide 5](#_Toc254268754)

[Prerequisites for authoring MBCA models 5](#_Toc254268755)

[Create an MBCA model 5](#_Toc254268756)

[Model Id guidelines 5](#_Toc254268757)

[Files required for defining a model 6](#_Toc254268758)

[Run discovery file on local computer 6](#_Toc254268759)

[Deploy an MBCA model 6](#_Toc254268760)

[Example 6](#_Toc254268761)

[Create an MBCA submodel 7](#_Toc254268762)

[Submodel ID 7](#_Toc254268763)

[Files required for defining a submodel 7](#_Toc254268764)

[Remoteable submodel 8](#_Toc254268765)

[Deploy an MBCA submodel 8](#_Toc254268766)

[Example 8](#_Toc254268767)

[Required files for MBCA models and submodels 9](#_Toc254268768)

[Manifest file 9](#_Toc254268769)

[Name of manifest file 9](#_Toc254268770)

[Content of manifest file 9](#_Toc254268771)

[ModelFormatVersion 10](#_Toc254268772)

[CompanyName 10](#_Toc254268773)

[DisplayName 10](#_Toc254268774)

[ModelVersion 10](#_Toc254268775)

[Sample manifest file 10](#_Toc254268776)

[Schema file 11](#_Toc254268777)

[Name of schema file 11](#_Toc254268778)

[Content of schema file 11](#_Toc254268779)

[Namespace in XSD document 11](#_Toc254268780)

[Sample schema file 11](#_Toc254268781)

[Discovery file 12](#_Toc254268782)

[Name of discovery file 12](#_Toc254268783)

[Content of discovery file 12](#_Toc254268784)

[Namespace in PS1 file 12](#_Toc254268785)

[Sample of discovery file 12](#_Toc254268786)

[Rules file 14](#_Toc254268787)

[Name of rules file 14](#_Toc254268788)

[Content of rules file 14](#_Toc254268789)

[Namespace 14](#_Toc254268790)

[Context 14](#_Toc254268791)

[Report 14](#_Toc254268792)

[Assert 14](#_Toc254268793)

[TestClause 14](#_Toc254268794)

[Severity 15](#_Toc254268795)

[Category 15](#_Toc254268796)

[Help Id 15](#_Toc254268797)

[Title 15](#_Toc254268798)

[Compliant 15](#_Toc254268799)

[Help topic 16](#_Toc254268800)

[Problem 16](#_Toc254268801)

[Impact 16](#_Toc254268802)

[Resolution 16](#_Toc254268803)

[Source 16](#_Toc254268804)

[Dynamic substitution 16](#_Toc254268805)

[Sample of rules file 16](#_Toc254268806)

[Content file 19](#_Toc254268807)

[Name of content file 19](#_Toc254268808)

[Contents of content file 19](#_Toc254268809)

[Sample of content file 19](#_Toc254268810)

[Dynamic parameter support 20](#_Toc254268811)

[Limitation on defining dynamic parameters for an MBCA model 21](#_Toc254268812)

[Limitation on defining dynamic parameters for an MBCA submodel 21](#_Toc254268813)

[Example 1 21](#_Toc254268814)

[Example 2 21](#_Toc254268815)

[Example 3: 22](#_Toc254268816)

[MBCA types 22](#_Toc254268817)

[Example 25](#_Toc254268818)

[Samples 26](#_Toc254268819)

[Antivirus model 26](#_Toc254268820)

[DiskSpace model 26](#_Toc254268821)

[System model 26](#_Toc254268822)

# Microsoft Baseline Configuration Analyzer Model Authoring Guide

The Microsoft Baseline Configuration Analyzer (MBCA) Model Authoring Guide provides guidance for authoring MBCA models. MBCA models contain best practice rules that you can use to scan, report, and analyze the behavior of applications that run on your computers. Models can have submodels that define a set of rules for the parent model that are logically grouped together. The following topics are covered in this guide.

 [Prerequisites for authoring MBCA models](#z1)

 [Create an MBCA model](#z2)

 [Deploy an MBCA model](#z3)

 [Create an MBCA submodel](#z4)

 [Deploy an MBCA submodel](#z5)

 [Required files for MBCA models and submodels](#z6)

 [Dynamic parameter support](#z7)

 [MBCA types](#z8)

 [Samples of models](#z9)

## Prerequisites for authoring MBCA models

We recommend a working understanding of the following technologies to help you write MBCA models.

 Windows PowerShell scripting

 XML

 XML schema definitions (XSD)

 XML schema validation (SCH)

Note

In this MBCA release UI depends on .NET 2.0 and PowerShell 2.0 but when scanning a remote machine it uses .NET 4 and PowerShell 3.0. The impact of this is if the model itself has a script that contains a breaking change from PowerShell 2.0 to PowerShell 3.0 then scanning the model locally may behave differently than scanning the model remotely.

## Create an MBCA model

An MBCA model defines a set of rules that are logically grouped together. The MBCA model is stored in the model’s root folder.

### Model Id guidelines

When defining a model Id, use the following guidelines:

 A model Id should not contain more than 50 characters.

 A model Id should contain no spaces.

 A model Id should contain only characters that are allowed for a Windows folder name.

### Files required for defining a model

The following files are required to define a model:

 Manifest File

 Schema File

 Discovery File

 Rules File

 Content File

For information about how to write these files, see [Required files for MBCA models and submodels](#z6) in this document.

### Run discovery file on local computer

Create your model such that the discovery file is always run on a local computer. That means it is possible for the model to call a submodel of another model.

Use all public APIs or cmdlets to collect information in the discovery file.

## Deploy an MBCA model

To deploy an MBCA model, you must have the following information:

 Model Id

 Manifest file

 Schema file

 Discovery file

 Rules file

 Content file

After you have this information, perform the following steps:



|  |
| --- |
| 1. After MBCA is installed on the computer, open %programdata%\Microsoft\Microsoft Baseline Configuration Analyzer 2\Models2. Create a folder named the model Id that you want to use. See [Model Id guidelines](#z10) in this document for more information.3. Store all model files (manifest, schema, discovery, rules, and content files) in the folder.4. If your model will be localized or translated, create a folder named en-US in the model Id folder, and then store a copy of the content file in the en-US folder. |

### Example

The following information is available:

|  |  |
| --- | --- |
| Model root: | %programdata%\Microsoft\Microsoft Baseline Configuration Analyzer 2\Models |
| Model Id: | Antivirus |
| Manifest file: | Manifest.psd1 |
| Schema file: | Antivirus.xsd |
| Discovery file: | Antivirus.ps1 |
| Rules file: | Antivirus.sch |
| Content file: | Antivirus.psd1 |

Creation of model folder and placement of model files:

(Model Root)\Antivirus

(Model Root)\Antivirus\Manifest.psd1

(Model Root)\Antivirus\Antivirus.xsd

(Model Root)\Antivirus\Antivirus.ps1

(Model Root)\Antivirus\Antivirus.sch

(Model Root)\Antivirus\Antivirus.psd1

If localization is supported, copy Antivirus.psd1 file to the en-US folder:

(Model Root)\Antivirus\en-US

(Model Root)\Antivirus\en-US\Antivirus.psd1

## Create an MBCA submodel

An MBCA model can also contain multiple submodels. Each submodel defines a set of rules that are logically grouped together. The submodel is stored in a subfolder of the model folder.

### Submodel ID

As with model IDs, consider the following as you define a submodel Id:

 A submodel Id should not contain more than 50 characters.

 No spaces are allowed in submodel Ids.

 A submodel Id should contain only characters that are allowed for a Windows folder name.

### Files required for defining a submodel

The following files are required to define a submodel:

 Schema file

 Discovery file

 Rules file

 Content file

For information about how to write these files, see [Required files for MBCA models and submodels](#z6) in this document.

### Remoteable submodel

Create a submodel so that the discovery file can be run on a remote computer. To do this, do not call local cmdlets or APIs in the submodel discovery file.

Use all public APIs or cmdlets to collect information in the discovery file.

## Deploy an MBCA submodel

You must have the following information to deploy an MBCA submodel:

 Model Id

 Submodel Id

 Schema file

 Discovery file

 Rules file

 Content file

After you have this information, perform the following steps:



|  |
| --- |
| 1. After MBCA has been installed, go to %programdata%\Microsoft\Microsoft Baseline Configuration Analyzer 2\Models.2. Open the model Id folder.3. Create a folder named with the submodel Id.4. Store all files (schema, discovery, rules, and content files) in that folder.5. If the submodel will be localized or translated, create a folder named en-US in the model Id folder, and then store a copy of the content file in the en-US folder. |

### Example

The following information is available:

|  |  |
| --- | --- |
| Model root: | %programdata%\Microsoft\Microsoft Baseline Configuration Analyzer 2\Models |
| Model Id: | System |
| Submodel Id: | Antivirus |
| Schema file: | Antivirus.xsd |
| Discovery file: | Antivirus.ps1 |
| Rules file: | Antivirus.sch |
| Content file: | Antivirus.psd1 |

Creation of model folder and placement of model files:

(Model Root)\System\Antivirus

(Model Root)\System\Antivirus\Antivirus.xsd

(Model Root)\System\Antivirus\Antivirus.ps1

(Model Root)\System\Antivirus\Antivirus.sch

(Model Root)\System\Antivirus\Antivirus.psd1

If localization is supported, copy Antivirus.psd1 file to the en-US folder:

(Model Root)\System\Antivirus\en-US

(Model Root)\System\Antivirus\en-US\Antivirus.psd1

## Required files for MBCA models and submodels

This section provides descriptions and samples of the required files for defining MBCA models and submodels.

### Manifest file

Note

The manifest file is required for an MBCA model. The manifest file contains the branding information of a model. This section contains details for defining a manifest file.

The manifest file is used only in MBCA models, not in submodels.

#### Name of manifest file

The name of the manifest file for any model should always be Manifest.psd1.

#### Content of manifest file

The following information is required in a manifest file:

##### ModelFormatVersion

(Mandatory. Type = string. Format = ’n1.n2’)

This field represents the version number of the model format. MBCA Model Format (version number should be 2.0)

##### CompanyName

(Mandatory, Type = string)

Company or vendor name of the model. This information is displayed in the MBCA GUI.

##### DisplayName

(Mandatory, Type = string)

Display name of the model. This information is displayed in the MBCA GUI.

##### ModelVersion

(Mandatory, Type = string. Format = ‘n1.n2.n3.n4’)

The version number of the model. This information is displayed in the MBCA GUI.

#### Sample manifest file

The following is an example of the manifest file in a model called Antivirus.

Manifest.psd1

@{

# This field represents the version number of format file. The format of this field will be n1.n2.

error message i = '2.0'

# Company or vendor of this model, this information will be shown in the Cmdlet output.

CompanyName = 'Microsoft'

# Display name of this model, this information will be shown in the Cmdlet output.

DisplayName = 'Disk Space'

# Version number of this model, this information will be shown in the Cmdlet output.

ModelVersion = '1.0'

}

### Schema file

The schema file contains the type definition of the XML document that is returned by the discovery script. The type definition of the XML document is defined in XML schema definition language (XSD). This section contains information about creating an XSD file.

#### Name of schema file

The name of a schema file should always be <ModelId>.xsd. If the model Id is Antivirus, the file name should be Antivirus.xsd.

#### Content of schema file

The file contains the XML schema definition. Details about how to define an XML schema are out of scope for this guide.

The following reference materials, and samples provided with this guide, can help you write a schema file.

 [World Wide Web Consortium XML Schema](http://www.w3.org/XML/Schema) (http://www.w3.org/XML/Schema)

 [W3Schools XML Schema Tutorial](http://www.w3schools.com/Schema/default.asp) (http://www.w3schools.com/Schema/default.asp)

 Hunter, David, et al. Beginning XML, 4th Edition (Programmer to Programmer). Wrox Publishing Co., 2007.

#### Namespace in XSD document

The namespace, which is defined in your XSD document, is very important. Make sure to define the same namespace in all three files (schema, discovery and rules); the namespace links all three documents.

Example of a namespace:

http://schemas.microsoft.com/mbca/models/Antivirus/2009/11

#### Sample schema file

The following is a sample schema file for a model called Antivirus:

Antivirus.xsd

<xs:schema

targetNamespace=" http://schemas.microsoft.com/mbca/models/Antivirus/2009/11"

 xmlns:tns=" http://schemas.microsoft.com/mbca/models/Antivirus/2009/11"

 elementFormDefault="qualified"

 xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <xs:element name="AntiVirusComposite" type="tns:AntiVirusCompositeType" />

 <xs:complexType name="AntiVirusCompositeType">

 <xs:sequence>

 <xs:element name="ScanBootSector" type="xs:string" minOccurs="0" maxOccurs="2"/>

 <xs:element name="AutoUpdateStatus" type="xs:string" minOccurs="0" maxOccurs="2"/>

 </xs:sequence>

 </xs:complexType>

</xs:schema

The top-level node defined in the above sample schema document is AntivirusComposite. AntivirusComposite contains the ScanBootSector and the AutoUpdateStatus child nodes.

### Discovery file

The discovery file contains a Windows PowerShell script that performs a discovery of the environment, and generates results as an XML document. This file has a PS1 file extension. Following is the information about how to write a Discovery file.

#### Name of discovery file

The name of a discovery file should always be <ModelId>.ps1. If the model Id is Antivirus, then the file name should be Antivirus.ps1.

#### Content of discovery file

The file contains a Windows PowerShell script that discovers an environment. Details about how to write a Windows PowerShell script are out of scope for this guide.

The following reference materials, and samples provided in this guide, can help you write a discovery file.

 [Windows PowerShell](http://technet.microsoft.com/en-us/library/bb978526.aspx) (http://technet.microsoft.com/en-us/library/bb978526.aspx)

 Payette, Bruce G. Windows PowerShell in Action. Manning Publications, 2007.

#### Namespace in PS1 file

The namespace defined in the discovery file should be same as that defined in the XSD file.

Example of a namespace:

http://schemas.microsoft.com/mbca/models/Antivirus/2009/11

#### Sample of discovery file

The following is an example of the discovery file of a model called Antivirus:

Antivirus.ps1

#

# Synclet PowerShell Script for Scenario: Antivirus

#

$p1 = 1

if ((test-path 'HKLM:\SOFTWARE\ComputerAssociates\eTrustAntivirus\CurrentVersion\Local Scanner') -eq $true)

{

$item1 = get-itemproperty -path 'HKLM:\SOFTWARE\ComputerAssociates\eTrustAntivirus\CurrentVersion\Local Scanner' -name 'Scan BootSector'

# get property

$p1=$item1.'Scan BootSector'

}

$item2 = get-itemproperty -path 'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Auto Update' -name 'AUOptions'

# get property

$p2=$item2.AUOptions

# targetNamespace

$tns=" http://schemas.microsoft.com/mbca/models/Antivirus/2009/11"

# construct XmlDocument and send to output stream

[xml] "<AntiVirusComposite xmlns='$tns'>

 <ScanBootSector>$p1</ScanBootSector>

 <ScanBootSector>999</ScanBootSector>

 <AutoUpdateStatus>$p2</AutoUpdateStatus>

 <AutoUpdateStatus>999</AutoUpdateStatus>

</AntiVirusComposite>"

The preceding sample script obtains the value of ScanBootSector and AutoUpdateStatus from the Windows registry, then sends the results to an XML document.

### Rules file

The rules file contains the Schematron rule, which is run to fire compliant and non-compliant messages. The rules file is also called an SCH file. This section contains information about writing a rules file.

#### Name of rules file

The name of a rules file should be <ModelId>.sch. If the model Id is Antivirus, then the file name should be Antivirus.sch.

#### Content of rules file

A rules file contains a list of rules against which your model is evaluated for compliance with best practices. Each rule contains the following information.

##### Namespace

The namespace in the Schematron document should be defined the same as in the schema and discovery files.

Example of namespace:

http://schemas.microsoft.com/mbca/models/Antivirus/2009/11

##### Context

Context defines the applicability of a rule to a specific node in an XML document, and is mandatory. The following is an example of context.

context="/tns:DiskspaceComposite/tns:DiskInfo"

In this example, the rule applies to the DiskInfo node, in the DiskspaceComposite node. TNS represents the namespace. See [Sample of rules file](#z11) for more details.

##### Report

Report will fire if the test clause is true.

##### Assert

Assert will fire if the test clause is false.

##### TestClause

Mandatory in both the report and assert, the test clause is the Boolean XPath expression that returns values of either true or false. Details about how to write a test clause are out of scope for this guide.

The following reference materials, and samples provided in this guide, can help you write a Rules file.

[W3Schools XPath Tutorial](http://www.w3schools.com/xpath/default.asp) (http://www.w3schools.com/xpath/default.asp)

[World Wide Web Consortium XML Path Language (XPath) Version 1.0](http://www.w3.org/TR/xpath) (http://www.w3.org/TR/xpath)

Note

We recommend using a simple test clause in your XPath expression, and using a more complex analysis in your discovery file.

##### Severity

Mandatory in both report and assert.

The severity of a rule defines the level of compliance or noncompliance of the model with the rule. In report, the value of severity should always be info, for informational. In assert, valid values for severity should are either warning or error, depending on the severity of the rule.

##### Category

Mandatory in both report and assert.

Each rule must be in one of following categories. For more information about these categories, see the MBCA Help, included in the MBCA download package.

 Security

 Performance

 Operation

 Policy

 Configuration

 Predeployment

 Prerequisite

##### Help Id

Mandatory in both report and assert.

The help Id is a unique string in a model that identifies a rule.

##### Title

Localizable and mandatory in both report and assert.

The title node defines the title of the message that is shown in the MBCA GUI.

Note

If a single rule generates more than one message, then uniqueness should be ensured by dynamic substitution.

##### Compliant

Localizable and mandatory in report.

A compliant message describes a rule, and states that the model is in compliance with the rule.

##### Help topic

Mandatory in both report and assert.

The help topic node defines a link to Web-based documentation where details about the rule are available.

##### Problem

Localizable and mandatory in assert.

The problem node contains a brief explanation of how the model is noncompliant with a rule.

##### Impact

Localizable and mandatory in assert.

The impact node is a brief description of the problem’s impact.

##### Resolution

Localizable and mandatory in assert.

The resolution node contains brief detail about how to resolve the problem.

##### Source

The source node is not mandatory, but if it is defined, it should be defined in both report and assert.

The source node defines the source of the result message. For example, if the result message is generated by a specific server, the server name can be stored in this field. If the result message is generated by a Web site, the site name can be stored in this field. If the source is not provided in the rule, the value localhost is defined by the MBCA engine.

##### Dynamic substitution

If it is required to substitute dynamic information from an XML document into the rule, you can also do this by providing a value-of tag in the rule. Dynamic values can be provided and referenced through {0}, {1} … substitution strings.

#### Sample of rules file

The following is an example of the rules file of the Antivirus model:

Antivirus.sch

<schema xmlns="http://purl.oclc.org/dsdl/schematron"

 xmlns:mssml="http://schemas.microsoft.com/sml/extensions/2007/03"

 xmlns:mssmlbpa="http://schemas.microsoft.com/sml/bpa/2008/02">

 <ns prefix="tns" uri="http://schemas.microsoft.com/mbca/models/Antivirus/2009/11" />

 <pattern>

 <rule context="/tns:AntiVirusComposite/tns:ScanBootSector">

 <assert

 mssmlbpa:helpID="ScanBootSector"

 mssml:severity="warning"

 mssml:category="mssmlbpa:security mssmlbpa:markupv2"

 test=". = '1'">

 <value-of select="."/>

 <mssmlbpa:title mssml:locid="ScanBootSector\_title"/>

 <mssmlbpa:helpTopic>http://Antiviru.com/ScanBootSector.htm</mssmlbpa:helpTopic>

 <mssmlbpa:problem mssml:locid="ScanBootSector\_problem"/>

 <mssmlbpa:impact mssml:locid="ScanBootSector\_impact"/>

 <mssmlbpa:resolution mssml:locid="ScanBootSector\_resolution"/>

 </assert>

 <report

 mssmlbpa:helpID="ScanBootSector"

 mssml:severity="info"

 mssml:category="mssmlbpa:security mssmlbpa:markupv2"

 test=". = '1'">

 <value-of select="."/>

 <mssmlbpa:title mssml:locid="ScanBootSector\_title"/>

 <mssmlbpa:helpTopic> http://Antiviru.com/ScanBootSector.htm </mssmlbpa:helpTopic>

 <mssmlbpa:compliant mssml:locid="ScanBootSector\_compliant"/>

 </report>

 </rule>

 </pattern>

 <pattern>

 <rule context="/tns:AntiVirusComposite/tns:AutoUpdateStatus">

 <assert

 mssmlbpa:helpID="AutoUpdateStatus"

 mssml:severity="error"

 mssml:category="mssmlbpa:policy mssmlbpa:markupv2"

 test=". = '4'">

 <value-of select="."/>

 <mssmlbpa:title mssml:locid="AutoUpdateStatus\_title"/>

 <mssmlbpa:helpTopic>http://Antiviru.com/AutoUpdateStatus.htm</mssmlbpa:helpTopic>

 <mssmlbpa:problem mssml:locid="AutoUpdateStatus\_problem"/>

 <mssmlbpa:impact mssml:locid="AutoUpdateStatus\_impact"/>

 <mssmlbpa:resolution mssml:locid="AutoUpdateStatus\_resolution"/>

 </assert>

 <report

 mssmlbpa:helpID="AutoUpdateStatus"

 mssml:severity="info"

 mssml:category="mssmlbpa:policy mssmlbpa:markupv2"

 test=". = '4'">

 <value-of select="."/>

 <mssmlbpa:title mssml:locid="AutoUpdateStatus\_title"/>

 <mssmlbpa:helpTopic> http://Antiviru.com/AutoUpdateStatus.htm </mssmlbpa:helpTopic>

 <mssmlbpa:compliant mssml:locid="AutoUpdateStatus\_compliant"/>

 </report>

 </rule>

 </pattern>

</schema>

The content for all localizable strings is defined in the [Content file](#z12) section of this guide.

### Content file

The content file contains all localizable strings that are accessed by the rules file. This file is also called a PSD1 file. The following information can help you write a content file.

#### Name of content file

The name of file should always be <ModelId>.psd1 for a given model. If the model is Antivirus, then the file name should be Antivirus.PSD1.

#### Contents of content file

A content file contains a list of name and value pairs in a Windows PowerShell hash table form. The name is the Loc-Id,referenced throughout the rules file; the values are the localized strings that are displayed to MBCA users in the GUI.

#### Sample of content file

The following is an example of the content file of the Antivirus model:

Antivirus.psd1

# Only add new (name,value) pairs to the end of this table

# Do not remove, insert or re-arrange entries

ConvertFrom-StringData @'

 ###PSLOC start localizing

 #

 # helpID="ScanBootSector"

 #

 ScanBootSector\_title = Boot Sector Scan (Indicator Setting {0})

 ScanBootSector\_problem = tbd: Problem for Constant name issue-1

 ScanBootSector\_impact = tbd: Impact for Constant name issue-1

 ScanBootSector\_resolution = tbd: Resolution for Constant name issue-1

 ScanBootSector\_compliant = tbd: Compliant for Constant name issue-1

 #

 # helpID="AutoUpdateStatus"

 #

 AutoUpdateStatus\_title = Auto Update Status (Indicator Setting {0})

 AutoUpdateStatus\_problem = tbd: Problem for Constant name issue-2

 AutoUpdateStatus\_impact = tbd: Impact for Constant name issue-2

 AutoUpdateStatus\_resolution = tbd: Resolution for Constant name issue-2

 AutoUpdateStatus\_compliant = tbd: Compliant for Constant name issue-2

'@

## Dynamic parameter support

The discovery scripts of both model and submodel support dynamic parameters. Dynamic parameter support is achieved by defining a Param block in the script file.

Example of Param block:

Param

(

 [ValidateSet("Steve", "Mary", "Carl")]

 [String] $UserName,

 [String] $Address

)

Details about how to write a Param block are out of scope for this guide.

The following Web site, and samples provided in this guide can help you write a Param block in the Discovery file.

[about\_Functions\_Advanced\_Parameters on MSDN](http://technet.microsoft.com/en-us/library/dd347600.aspx) (http://technet.microsoft.com/en-us/library/dd347600.aspx)

There is a general restriction on defining parameters for both a model and submodel.

Note

Do not use the same parameter name as a cmdlet parameter name.

The following is a list of cmdlet parameter names.

 ModelId

 RepositoryPath

 Mode

 SubModelId

 Context

 ComputerName

 CertificateThumbprint

 ConfigurationName

 Credential

 Authentication

 Port

 ThrottleLimit

 UseSsl

 Verbose

 Debug

 ErrorAction

 WarningAction

 ErrorVariable

 WarningVariable

 OutVariable

 OutBuffer

### Limitation on defining dynamic parameters for an MBCA model

Because dynamic parameters that are defined in the MBCA model file are displayed in the GUI, it is recommended that you define Windows PowerShell native-type parameters. Default values for parameters are ignored by the GUI. Additionally, Bool arrays are not recognized by the GUI.

### Limitation on defining dynamic parameters for an MBCA submodel

MBCA submodel dynamic parameters that cannot take null values should be defined as mandatory in the Param block. The following examples show how to resolve this limitation.

#### Example 1

Incorrect:

 [ValidateSet("Steve", "Mary", "Carl")]

 [String] $UserNames

Correct:

 [Parameter (Mandatory=$true)]

 [ValidateSet("Steve", "Mary", "Carl")]

 [String] $UserNames

#### Example 2

Incorrect:

 [Bool] $Success

Correct:

 [Parameter (Mandatory=$true)]

 [Bool] $Success

#### Example 3:

Incorrect:

 [DateTime] $DateOfJoin

Correct:

 [Parameter (Mandatory=$true)]

 [DateTime] $ DateOfJoin

## MBCA types

MBCA supports a set of built-in complex types that are not directly available through XML types. The purpose of these types is to encourage model writers to use them so that these types are able to cross reference among multiple models.

The following is an example of the ‘MsMbcaTypes.xsd’:

MsMbcaTypes.xsd

<?xml version="1.0" encoding="utf-8"?>

<xs:schema targetNamespace="http://schemas.microsoft.com/mbca/2010/01"

 xmlns:tns="http://schemas.microsoft.com/mbca/2010/01"

 xmlns:xs="http://www.w3.org/2001/XMLSchema"

 elementFormDefault="qualified">

 <xs:element name="ADObject" type="tns:ADObjectType" />

 <xs:element name="ComputerName" type="tns:ComputerNameType" />

 <xs:element name="DomainName" type="tns:DomainNameType" />

 <xs:element name="IPv4Address" type="tns:IPv4AddressType" />

 <xs:element name="IPv6Address" type="tns:IPv6AddressType" />

 <xs:element name="MACAddress" type="tns:MACAddressType" />

 <xs:element name="Port" type="tns:PortType" />

 <xs:element name="RegistryValue" type="tns:RegistryValueType" />

 <xs:element name="URL" type="tns:URLType" />

 <xs:element name="WMIObject" type="tns:WMIObjectType" />

 <xs:complexType name="ADObjectType">

 <xs:sequence minOccurs="0" maxOccurs="unbounded">

 <xs:element name="Attribute" type="tns:ADAttributeType" />

 </xs:sequence>

 <xs:attribute name="DN" type="xs:string" />

 </xs:complexType>

 <xs:complexType name="ADAttributeType">

 <xs:simpleContent>

 <xs:extension base="xs:string">

 <xs:attribute name="Name" type="xs:string" use="required" />

 </xs:extension>

 </xs:simpleContent>

 </xs:complexType>

 <xs:simpleType name="ComputerNameType">

 <xs:restriction base="xs:string" />

 </xs:simpleType>

 <xs:simpleType name="DomainNameType">

 <xs:restriction base="xs:string" />

 </xs:simpleType>

 <xs:simpleType name="IPv4AddressType">

 <xs:restriction base="xs:string" />

 </xs:simpleType>

 <xs:simpleType name="IPv6AddressType">

 <xs:restriction base="xs:string" />

 </xs:simpleType>

 <xs:simpleType name="MACAddressType">

 <xs:restriction base="xs:string" />

 </xs:simpleType>

 <xs:simpleType name="PortType">

 <xs:restriction base="xs:int" />

 </xs:simpleType>

 <xs:complexType name="RegistryValueType">

 <xs:simpleContent>

 <xs:extension base="xs:string">

 <xs:attribute name="Path" type="xs:string" />

 <xs:attribute name="Name" type="xs:string" use="required" />

 <xs:attribute name="Type" type="tns:RegistryValueTypeType" use="required" />

 </xs:extension>

 </xs:simpleContent>

 </xs:complexType>

 <xs:simpleType name="RegistryValueTypeType">

 <xs:restriction base="xs:string">

 <xs:enumeration value="REG\_BINARY" />

 <xs:enumeration value="REG\_DWORD" />

 <xs:enumeration value="REG\_EXPAND\_SZ" />

 <xs:enumeration value="REG\_MULTI\_SZ" />

 <xs:enumeration value="REG\_QWORD" />

 <xs:enumeration value="REG\_SZ" />

 <xs:enumeration value="REG\_NONE" />

 </xs:restriction>

 </xs:simpleType>

 <xs:simpleType name="URLType">

 <xs:restriction base="xs:string" />

 </xs:simpleType>

 <xs:complexType name="WMIObjectType">

 <xs:sequence minOccurs="0" maxOccurs="unbounded">

 <xs:element name="Property" type="tns:WMIPropertyType" />

 </xs:sequence>

 <xs:attribute name="Path" type="xs:string" />

 </xs:complexType>

 <xs:complexType name="WMIPropertyType">

 <xs:simpleContent>

 <xs:extension base="xs:string">

 <xs:attribute name="Name" type="xs:string" use="required" />

 </xs:extension>

 </xs:simpleContent>

 </xs:complexType>

</xs:schema>

### Example

Following is an example in which we call some of the MBCA types.

<xs:schema

 targetNamespace=" http://schemas.microsoft.com/mbca/models/Example/2009/11"

 xmlns:tns=" http://schemas.microsoft.com/mbca/models/Example/2009/11"

 elementFormDefault="qualified"

    xmlns:mbcatype="http://schemas.microsoft.com/mbca/types/2010/01/v1.0"

 xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:import namespace="http://schemas.microsoft.com/mbca/types/2010/01/v1.0" />

 <xs:element name="ExampleComposite" type="tns:ExampleCompositeType" />

 <xs:complexType name="ExampleCompositeType">

 <xs:sequence>

 <xs:element ref="mbcatype:ComputerName"/>

 <xs:element ref="mbcatype:DomainName"/>

 </xs:sequence>

 </xs:complexType>

</xs:schema>

## Samples

The following are sample models that can help you understand how to write a model.

### Antivirus model

This model discovers the scan-boot-sector and auto-update-status information from the registry and performs analysis of whether those values are compliant.

###

### DiskSpace model

This model collects information related to all partitions present in a system, and provides an analysis of whether sufficient disk space is available in that partition.

###

### System model

The system model is an aggregate model that has two submodels, Antivirus and Diskspace. The system model allows users to provide the names of computers on which submodels are run as parameters to the submodel.

###