

Forefront TMG – Scripting mit VBScript und Powershell

Forefront TMG kommt wird mit einer Read Only Powershell Unterstuetzung geliefert. Das folgende Bilderbuch zeigt einige Powershell-Befehle, sowie die Moeglichkeiten mit Hilfe von VBScript und Jscript TMG zu administrieren.

Quellen:

www.isascripts.org – einige VBScript und Jscript Beispiele

<http://www.microsoft.com/downloads/en/details.aspx?FamilyID=8809cfda-2ee1-4e67-b993-6f9a20e08607&displaylang=en> – Forefront TMG SDK

Anzeige aller neuen Forefront TMG COM Elemente

<http://msdn.microsoft.com/en-us/library/dd447763.aspx>

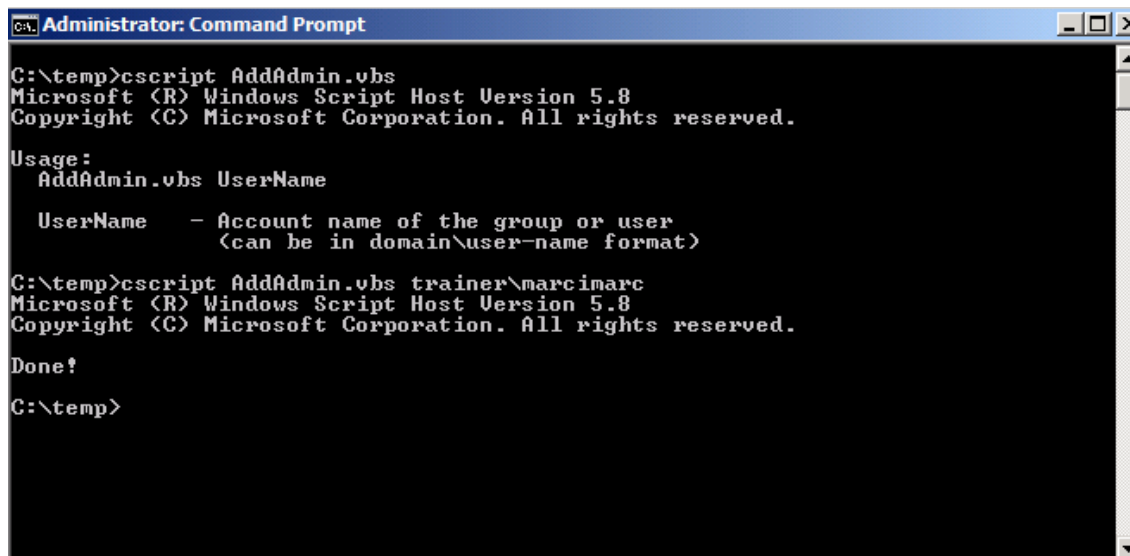
Forefront TMG SDK – Administration script examples

The screenshot shows the 'Forefront TMG Administration Script Samples' page. The left pane shows a tree view of the documentation structure. The main content area contains a table of script examples.

Name	Description
ActiveSessions.vbs	A script that creates and executes a query on the <code>FPCSessionsMonitor</code> collection for the active sessions of the <code>Microsoft Firewall Service</code> and displays the active sessions that existed when the query started.
AddAdmin.vbs	A script that adds a specific user as an administrator with permissions to monitor the Forefront TMG computer and network activity, for example, to view logs and reports, but not to configure any specific monitoring functionality.
AddCacheRule.vbs	A script that creates a new URL set in the URL sets collection of the proxy server, adds URLs to the URL set, and creates two new cache rules for caching content with a fixed <code>Time to Live (TTL)</code> range from all sites on the External network except the sites in the new URL set.
AddConnectivityVerifier.vbs	A script that creates a new connectivity verifier.
AddRuleAndURLSet.vbs	A script that creates a new URL set in the URL sets collection, adds sites to the URL set, creates a new access rule, and adds the new URL set to the objects referenced in the <code>URLSets</code> property of the access rule.
ConfigureAlerts.vbs	A script that retrieves the collection of alerts defined in the containing <code>array</code> , iterates through the collection, and sets the e-mail address associated with each alert definition.
ConnectToCSS.vbs	A script that connects to the Configuration Storage server specified by the user using the credentials of the logged-on user, the credentials of a specified enterprise administrator with read-write permissions for accessing the stored enterprise configuration, or the credentials of a specified enterprise auditor with read-only permissions for accessing the stored enterprise configuration. The script then displays the name of the enterprise and disconnects from the Configuration Storage server (not applicable to Forefront TMG Medium Business Edition).
ControlAccessByScheduleAndUserSet.vbs	A script that creates the access rules, user set, and URL set needed to allow a specific group of workers in an organization restricted access to the Internet. The group is allowed to access only the sites listed in the URL set and only during the hours specified in the Work hours schedule supplied with Forefront TMG. All other workers using computers that belong to the Internal network are granted unlimited access to the Internet.
CreateEnterprisePolicy.vbs	A script that connects to the Configuration Storage server specified by the user using the credentials of the logged-on user or the credentials of a specified user that has the permissions needed to modify the stored enterprise configuration, and creates a new enterprise policy with the name specified by the user (not applicable to Forefront TMG Medium Business Edition).
HttpFilterConfig.vbs	A script that exports the configuration for the HTTP Filter Web filter from the corresponding vendor parameters set of the specified policy rule to the specified file, or imports the configuration for the HTTP Filter Web filter from the specified file to a new vendor parameters set of the specified policy rule.
ImportExport.vbs	A script that exports the configuration of the array for a Forefront TMG computer to a specified XML file or imports the configuration in a specified XML file to the array object of the Forefront TMG computer.
SetNetworkRelation.vbs	A script that creates a new network rule that defines a NAT relationship between any network belonging to the predefined All Protected Networks network set and the External network. This rule will apply to any new perimeter network created in the future because such a network will be included automatically in the All Protected Networks network set.
ShowICMPSystemPolicy.vbs	A script that retrieves the collection of system policy rules defined for an Forefront TMG computer and then implicitly uses the <code>_NewEnum</code> property to iterate through the collection and display the names of the system policy rules for ICMP along with the user sets to which each rule applies.

This section contains the following topics with code examples taken from the samples to help you write Forefront TMG administration scripts in Microsoft® Visual Basic® Scripting Edition (VBScript):

Einen User zu einer ISA/TMG Server Rolle hinzufuegen



```
Administrator: Command Prompt
C:\temp>cscript AddAdmin.vbs
Microsoft (R) Windows Script Host Version 5.8
Copyright (C) Microsoft Corporation. All rights reserved.

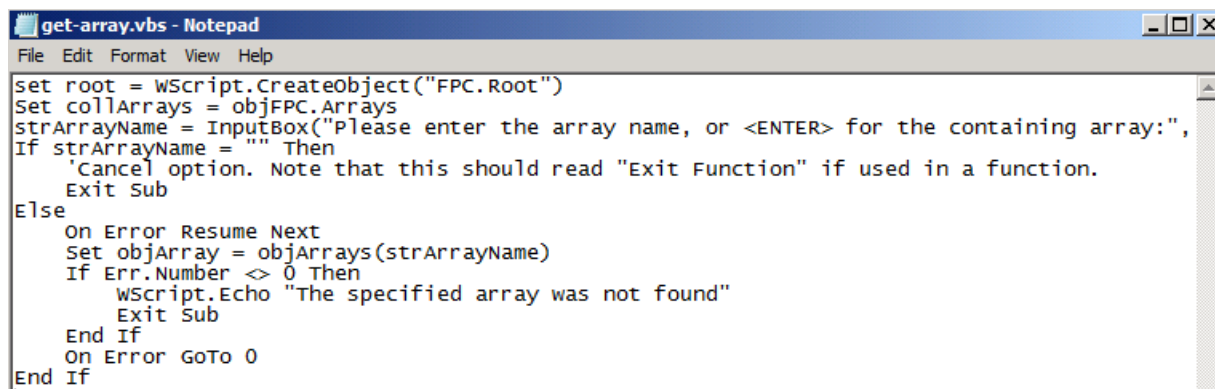
Usage:
  AddAdmin.vbs UserName

  UserName - Account name of the group or user
             (can be in domain\user-name format)

C:\temp>cscript AddAdmin.vbs trainer\marcinarc
Microsoft (R) Windows Script Host Version 5.8
Copyright (C) Microsoft Corporation. All rights reserved.

Done!
C:\temp>
```

Forefront TMG Array anzeigen



```
get-array.vbs - Notepad
File Edit Format View Help
set root = WScript.CreateObject("FPC.Root")
set collArrays = objFPC.Arrays
strArrayName = InputBox("Please enter the array name, or <ENTER> for the containing array:",
If strArrayName = "" Then
  'Cancel option. Note that this should read "Exit Function" if used in a function.
  Exit Sub
Else
  On Error Resume Next
  Set objArray = objArrays(strArrayName)
  If Err.Number <> 0 Then
    WScript.Echo "The specified array was not found"
    Exit Sub
  End If
  On Error GoTo 0
End If
```

Der Klassiker - TMG Konfigurationsbackup

```
Dim fileName
Dim WSHNetwork
Dim shareName: shareName = WScript.Arguments(0)
Dim xmldom : set xmldom = CreateObject("Msxml2.DOMDocument")
Dim fpc : set fpc = WScript.CreateObject("Fpc.Root")
Dim array : set array = fpc.GetContainingArray
set WSHNetwork = CreateObject("WScript.Network")
fileName=shareName & "\" & WSHNetwork.ComputerName & "-" &
Month(Now) & "-" & Day(Now) & "-" & Year(Now) & ".xml"
array.Export xmldom, 0
xmldom.save(fileName)
```

Die Codezeilen in eine Textdatei mit der Extension .VBS speichern und dann per Cscript ausfuehren

Bsp.: Cscript TMGBACKUP.VBS \\ENTFERNTERSERVER\TMG-BACKUP

Beispiel: Auslesen der VPN Konfiguration und Import in Forefront TMG

Quelle: <http://www.msisafaq.de/Anleitungen/TMG/VPN/VPNImport.htm> (Author: Christian Groebner)

Mittels folgendem Skript können Sie die Einstellungen aus der Konfigurationssicherung von ISA Server 2006 auslesen und diese in die Konfiguration von Microsoft TMG übernehmen. Kopieren Sie hierzu das Skript in einen Texteditor, z.B. Notepad und speichern Sie es unter dem Dateinamen **vpnfix.vbs** ab.

```
#####  
#####  
  
Dieses Skript übernimmt die IPSec-Einstellungen für Phase I und II der IPSec-VPN-Tunnel  
aus der ISA Server 2006-Konfiguration nach dem Import und wendet diese auf die  
Konfiguration  
von Microsoft TMG an.  
  
Die Verwendung dieses Skripts erfolgt auf eigene Verantwortung.  
Es wird keine Haftung für eventuelle Schäden übernommen!  
  
Geschrieben von Christian Gröbner [MVP Forefront]  
#####  
#####  
  
'----- Sub restore_ipsec_settings -----  
  
Sub restore_ipsec_settings(fpcRoot, VPN_Name, Int_PhaseI, Enc_PhaseI, Int_PhaseII,  
Enc_PhaseII)  
  
Dim Intproviders  
Dim Encproviders  
  
Intproviders = Array("SHA1","MD5")  
Encproviders = Array("DES","3DES")  
  
set objIPSec =  
fpcRoot.GetContainingArray.NetworkConfiguration.Networks.Item(VPN_Name).VPNConf  
figuration.IPSecSettings  
  
wscript.echo "Restoring IPSec-settings for network" & VPN_Name & vbCrLf  
wscript.echo "Phase I integrity : " & Intproviders(Int_PhaseI)  
objIPSec.Phase1Integrity = Int_PhaseI  
wscript.echo "Phase I encryption : " & Encproviders(Enc_PhaseI)  
objIPSec.Phase1Encryption = Enc_PhaseI  
wscript.echo "Phase II integrity : " & Intproviders(Int_PhaseII)  
objIPSec.Phase2Integrity = Int_PhaseII  
wscript.echo "Phase II encryption : " & Encproviders(Enc_PhaseII) & vbCrLf  
objIPSec.Phase2Encryption = Enc_PhaseII  
wscript.echo "Successfully applied the settings"  
wscript.echo "-----" & vbCrLf
```

```
End Sub
```

```
' ----- Sub Main -----
```

```
Sub Main()
```

```
Dim PhaseI_Int  
Dim PhaseI_Enc  
Dim PhaseII_Int  
Dim PhaseII_Enc  
Dim config
```

```
config = Inputbox("Please enter the complete path and filename with extension to the  
existing configuration file of ISA 2006 : (Example: C:\Temp\config.xml)")
```

```
Set xmlFile = CreateObject("Microsoft.XMLDOM")
```

```
If xmlFile.load(config) then
```

```
set objFPC = CreateObject("FPC.Root")
```

```
Set networkNodes = xmlFile.getElementsByTagName("fpc4:Network")
```

```
For each networkNode in networkNodes
```

```
If (Not(networkNode.selectSingleNode("fpc4:NetworkConnectionType") is Nothing)) Then
```

```
If (networkNode.selectSingleNode("fpc4:NetworkConnectionType").Text = 4) Then
```

```
PhaseI_Int = 0
```

```
PhaseI_Enc = 1
```

```
PhaseII_Int = 0
```

```
PhaseII_Enc = 1
```

```
Name = networkNode.selectSingleNode("fpc4:Name").Text
```

```
Set ipsecSettingsNode =  
networkNode.selectSingleNode("fpc4:VpnNetworkConfiguration/fpc4:VpnNetworkIPSecSet  
tings")
```

```
If (Not(ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase1Encryption") is  
Nothing)) Then PhaseI_Enc =
```

```
ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase1Encryption").Text
```

```
If (Not(ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase1Integrity") is  
Nothing)) Then PhaseI_Int =
```

```
ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase1Integrity").Text
```

```
If (Not(ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase2Encryption") is  
Nothing)) Then PhaseII_Enc =
```

```
ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase2Encryption").Text
```

```
If (Not(ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase2Integrity") is
```

```

Nothing)) Then PhaseII_Int =
ipsecSettingsNode.selectSingleNode("fpc4:VpnNetworkPhase2Integrity").Text

    restore_ipsec_settings objFPC, Name, PhaseI_Int, PhaseI_Enc, PhaseII_Int, PhaseII_Enc

End If

End If

Next

objFPC.GetContainingArray.Save

Else

wscript.echo("The file does not exist!")

End If

End Sub

'----- Start the script -----

Main

```

Anzeige der URL Kategorien von Forefront TMG

```

set root=CreateObject("FPC.Root")
For Each cat in root.GetContainingArray().RuleElements.UrlCategories
    wscript.echo "" & cat.Name & "" --> " & cat.CategoryID
Next

```

```

Administrator: Command Prompt
C:\temp>cscript url-category.vbs
Microsoft (R) Windows Script Host Version 5.8
Copyright (C) Microsoft Corporation. All rights reserved.

'Alcohol' --> 1
'Anonymizers' --> 2
'Art/Culture/Heritage' --> 3
'Blogs/Wiki' --> 4
'Botnet' --> 5
'Chat' --> 6
'Child Friendly Materials' --> 7
'Criminal Activities' --> 8
'Dating/Personals' --> 9
'Digital Postcards' --> 10
'Dubious' --> 11
'Edge Content Servers/Infrastructure' --> 12
'Education/Reference' --> 13
'Employment' --> 14
'Fashion/Beauty' --> 15
'Financial' --> 16
'Forum/Bulletin Boards' --> 17
'Free Hosting' --> 18
'Gambling' --> 19
'Games' --> 20
'General Business' --> 21
'General Entertainment' --> 22
'Government/Military' --> 23
'Hacking/Computer Crime' --> 24
'Hate/Discrimination' --> 25
'Health' --> 26
'Humor/Comics' --> 27
'Illegal Drugs' --> 28
'Internet Services' --> 29
'Legal Services & Reference' --> 30
'Malicious' --> 32
'Mature Content' --> 33
'Media Sharing' --> 34
'Motor Vehicles' --> 35
'News' --> 36

```

TMG und Powershell

TMG Root Object definieren (Root Object ist immer FPC.Root)

```

Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\administrator.TRAINER> $TMGRoot = New-Object -comObject FPC.root
PS C:\Users\administrator.TRAINER> _

```

Anzeige des TMG Roots

```

Administrator: Windows PowerShell
PS C:\Users\administrator.TRAINER> $TMGRoot

Arrays                : System.__ComObject
Enterprise             : System.__ComObject
ConfigurationStorageServer : TMG-EN.trainer.intern
RequireApplyChanges   : False
ChangesMade           : False
StorageChangeNumber   : 57360
IsaEdition             : 32
VendorMode            : False
ConfigurationMode     : 0

```

Abfrage einzelner Elemente (Hinter \$TMGRoot. mit TAB Taste alle Elemente anzeigen)

```
Administrator: Windows PowerShell
PS C:\Users\administrator.TRAINER> $TMGRoot.ConfigurationStorageServer
TMG-EN.trainer.intern
PS C:\Users\administrator.TRAINER> _
```

Anzeige der Eigenschaften von FPC.Root

```
Administrator: Windows PowerShell
PS C:\Users\administrator.TRAINER> $tmgroot | Get-Member -membertype *property

TypeName: System.__ComObject#<8bf0aefa-b4fd-4d81-8046-a069d948d673>

Name                MemberType Definition
----                -
Arrays              Property    IFPCArrays Arrays () {get}
ChangesMade         Property    bool ChangesMade () {get}
ConfigurationMode   Property    FpcConfigurationMode ConfigurationMode () {get}
ConfigurationStorageServer Property    string ConfigurationStorageServer () {get}
Enterprise          Property    IFPCEEEnterprise Enterprise () {get}
IsaEdition          Property    FpcIsaEditionType IsaEdition () {get}
RequireApplyChanges Property    bool RequireApplyChanges () {get} {set}
StorageChangeNumber Property    int64 StorageChangeNumber () {get}
VendorMode          Property    bool VendorMode () {get} {set}

PS C:\Users\administrator.TRAINER>
```

Ermitteln des TMG Enterprise und dessen Arrays. Anschliessender Export der Konfiguration

Quelle: <http://www.microsoft.com/learning/en/us/Book.aspx?ID=13148&locale=en-us>

```
#+++++
#
# This code is Copyright (c) 2009 Microsoft Corporation.
#
# All rights reserved.
#
# THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY
# OF
# ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED
# TO
# THE IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A
# PARTICULAR PURPOSE.
#
# IN NO EVENT SHALL MICROSOFT AND/OR ITS RESPECTIVE SUPPLIERS BE
# LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR
# ANY
# DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR
# PROFITS,
# WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER
# TORTIOUS
# ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR
# PERFORMANCE
# OF THIS CODE OR INFORMATION.
#
```

```

#-----
# declare and define the TMG root object
$oFPC = New-Object -comObject FPC.root

# declare and define the value that expresses if array work was successful
$bFailed = $False

# declare and define the value that expresses whether any changes occurred
$bChanges = $False

# declare and define the TMG arrays collection
$cArrays = $oFPC.Arrays

# declare the Array object variable
$oArray

# enumerate (walk through) the array list looking for the ones of interest
$result = "Outbound Proxy"

Foreach ($oArray in $cArrays)
{
write-host "Policy name:" $oArray.PolicyAssignment.EnterprisePolicyUsed.Name
If ( $oArray.PolicyAssignment.EnterprisePolicyUsed.Name -eq $result){
write-host "Array Name:" $oArray.name
# if we try to work an array and fail, we want to quit now
If ( UpdateArray($oArray) -eq $false )
{$bFailed = $True}
# otherwise, we declare that we made changes to at least one array
else {$bChanges = $True}
}
}
# if we made any changes without failures, now is the time to save them
If ($bChanges -ne ( $bFailed ))
{SaveChanges( $oArray )}
{SaveChanges( $oArray )}

Function UpdateArray( $oArray )
{
#define the default return value for this function
$updateArray = $False

#declare and define the TMG export file path
$szOutFilePath = "C:\TmgExportFile.xml"

#declare and define the optional data for the export method
$iOptionalData = 0

#declare and define the TMG export data password
$szPassword = ""

```



```

#declare and define the TMG export file comment section
$szComment = "Exported by ExportArrays.ps1 at "

#try to export the current configuration to a file
write-host "Name:" $oArray.Name
$oArray.ExportToFile($szOutFilePath, $iOptionalData, $szPassword, $szComment)

#disable script error handling
$bOverwrite= $False

#declare and define the TMG import services reset flag
$bReset = $False

#declare and define the TMG import policy reload flag
$bReload = $True

#declare and define the TMG import file path
$szInFilePath = "C:\TmgImportFile.xml"

#try to import the configuration update from a file
$oArray.ImportFromFile($szInFilePath, $iOptionalData, $szPassword, $bOverwrite,
$bReset, $bReload)
$updateArray = $True
}

Function SaveChanges( $oArray )
{
    Trap [Exception]
    {
        write-host "Failed to save the array configuration changes; "
        $_.Exception.GetType().FullName "; " $_.Exception.Message

        # if it fails, tell the user and bail out
        write-host "Failed to save the array configuration changes; "
        $_.Exception.GetType().FullName "; " $_.Exception.Message
        Exit
    }

    # define the default value of this function
    $SaveChanges = $False
    $oArray.Save

    # no failures, return "true"
    $SaveChanges = $True
    write-host "Changes Saved"
}

```
