# Citrix Hypervisor (XenServer) Backup Skript

Guten morgen allerseits
Da man wegen der aktuellen ausserordentlichen Lage (Corona-Virus) kann man kaum was machen, da habe ich mir mal ein wenig Gedanken gemacht, wie ich meine kleine Testumgebung einfach sichern kann.
Im geschäftlichen Umfeld gibt es div. Backup-Anbieter. Für meine Testumgebung wollte ich dafür jedoch kein Geld ausgeben und wollte auf Basis eines älteren Artikels ein komplettes Skript bauen und bin dabei auf was gestossen, was ich bisher immer ignoriert hatte…

Seit längerem hat Citrix ein SDK Kit in welchem auch ein PowerShell Modul für den Citrix Hypervisor mit dabei ist. Auf Basis des Blogartikels habe ich mich dann mal ans Werk gemacht.

Die Idee war ein Skript zu basteln welches folgendes macht (Ähnlichkeiten mit bekannten Backup Lösungen sind gewollt):
– Snapshot von definierten VMs
– erstellte Snapshots exportieren
– Snapshots wieder löschen

Als erstes muss das SDK auf einer definiertem definierten Windows Rechner installiert werden. Die Anleitung dazu ist im Download enthalten, daher gehe ich darauf nicht ein.

Wie in anderen Skripts wollte ich als erstes einen kompletten Block erstellen, in welchem erst einmal die verschiedenen Variablen definiert werden:

```
# Define Variables
$Xenhost = "192.168.199.210"
$XenUser = "root"
$XenPW = "PITPassword"
$BkpDate = Get-Date -Format yyyyMMdd-HHmm
$BkpStart = (Get-Date).AddHours(-1) # Subtract one hour for the system time / check with summer time
$BkpTag = "Backup"
$ExportPath = "S:\VM-Backups"
```

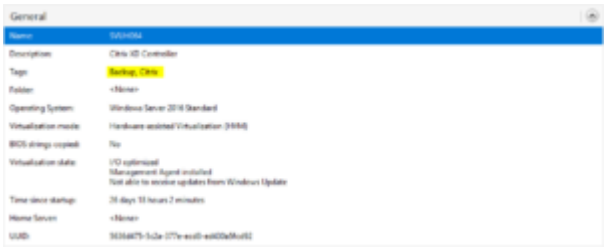
Die ersten drei definieren den Zugriff auf die Citrix Hypervisor Umgebung.
Mit dem *$BkpDate* lesen wir Datum/Zeit aus und generieren daraus einen Teil des späteren Snapshot Namens.
Das *$BkpStart* definiert den Start des Skriptes. Dieses nutzen wir später um nur Snapshots zu löschen, welche nach dem Skriptsstart erstellt wurden. Im Test musste ich noch eine Stunde abziehen, da die Systemzeit anscheinend UTC hat.
Die Variable *$BkpTag* definiert, auf welchen VM Tag geachtet werden muss.
Der *$ExportPath* definiert, wohin die Snapshots exportiert werden sollen.

Im zweiten Block wird die Verbindung zum Pool hergestellt. Hier habe ich die Vorlage aus dem Block fast 1:1 übernommen. Lediglich die Meldungen wurden ein wenig ergänzt:

```
# Connect to Citrix Hypervisor
# Try the defined host otherwise change to new PoolMaster
# --- Base scriptlet from pastebin.com ---
```

```
Try {
     $Session = Connect-XenServer -Url https://$Xenhost -
UserName $XenUser -Password $XenPW -NoWarnCertificates -
SetDefaultSession
      Write-Host -ForegroundColor DarkGreen "Successful
connected to $Xenhost"
    }
    Catch [XenAPI.Failure]
        {
        [string]$PoolMaster = $_.Exception.ErrorDescription[1]
         Write-Host -ForegroundColor Red "$($Pools.$Pool) is
slave, Master was identified as $PoolMaster, trying to
connect"
        $Pools.Pool = $PoolMaster
        $Session = Connect-XenServer -url "http://$PoolMaster"
-UserName $XenUser -Password $XenPW -NoWarnCertificates -
SetDefaultSession
         Write-Host -ForegroundColor DarkGreen "Successful
connected to $PoolMaster"
       }
```

Im dritten Block erstellen wir eine Liste aller VMs, die den vorher definierten Tag aufweisen. Damit dies funktioniert, muss den VMs der Tag zugewiesen worden sein:



```
# Enumerate all VMs
# --- Base scriptlet from pastebin.com ---
# Added selection of "Backup" Tag

$BkpVMs = Get-XenVM | Where {$_.is_a_template -eq $False -and
```

```
$_.is_a_snapshot -eq $False -and $_.domid -ne 0 -and $_.tags -contains $BkpTag}
```

Im vierten Block wird für jede gefundene VM ein Snapshot erstellt. Damit die Snapshots auseinander gehalten werden können, wird ein Name aus dem VM Namen und dem *$BkpDate* gebildet.

```
# Create Snapshot of each found VM
# --- Base scriptlet from pastebin.com ---
# Loop added and define the snapshot name based on VM name and date/time of backup start

$BkpVMs | ForEach-Object {
    $VMName = $_.name_label
    $VMUuid = $_.uuid
    $Snapshotname = "$VMName - $BkpDate"
     Write-Host -ForegroundColor Gray "Snapshot with name $Snapshotname for VM $VMName (UUID $VMUuid ) will be created"
     Invoke-XenVM -Uuid $VMUuid -XenAction Snapshot -NewName $Snapshotname
    }
```

Im fünften Block wird eine Liste aller Snapshots erstellt, welche den Tag aufweisen und die nach dem Skriptstart (*$BkpStart*) erstellt wurden.

```
# Enumerate all Snapshots
# --- Base scriptlet from pastebin.com ---
# Only snapshots of VMs with the "Backup" tag and snapshots newer the start time

$BkpVMs2 = Get-XenVM |  Where {$_.is_a_snapshot -eq $True -and $_.tags -contains $BkpTag -and $_.snapshot_time -gt $BkpStart}
```

Im sechsten Block werden alle gefundenen Snapshots ans definierte Ziel (*$ExportPath*) exportiert.

```
# Export the enumerated Snapshots
# --- Base scriptlet from pastebin.com ---

$BkpVMs2 | ForEach {
    $Snapshotname2 = $_.name_label
    $SnapshotUuid2 = $_.uuid
    $ExportFile = "$Exportpath\$Snapshotname2.xva"
    Write-Host -ForegroundColor DarkCyan "Snapshot with name $Snapshotname2 will be exported to $ExportFile"
    Export-XenVm -Uuid $SnapshotUuid2 -XenHost $Xenhost -path "$ExportFile"
    }
```

Im siebten und letzten Block werden alle gefundenen Snapshots wieder gelöscht.

```
# Delete snapshot of each VM which was created before
# --- Base scriptlet from pastebin.com ---
# Loop added to remove all snapshots created before

$BkpVMs2 | ForEach-Object {
    $Snapshotname3 = $_.name_label
    Write-Host -ForegroundColor DarkYellow "Snapshot with name $Snapshotname3 will be deleted"
    Remove-XenVM -Name $Snapshotname3
    }
```

Dieses simple Skript ist eine einfache Variante um schnell eine Sicherung einer Testumgebung zu erhalten. Dieses soll keinesfalls ein Ersatz von professionellen Lösungen sein!

Nun viel Spass beim Nachbauen :-)

XS-BackupCTXHYV.ps1_Herunterladen