Windows Powershell v roce 2013

Powershell v současné praxi Windows I.

Patrik Malina

patrikmalina.eu

O čem bude řeč

- Powershell: kde jej najdeme
- Hranice ovládání Powershellem
- Po staru... a v Powershellu
- Data v Powershellu
- Powershell... a co za to?

Kde najdeme Powershell

- Verze Powershellu
  - Prehistorie – v1
  - Praxe – v2
  - Nástup – v3

Kde najdeme Powershell

- Přirozený výskyt Powershellu
  - V3
    - Windows 8/2012
  - V2
    - Windows 7/2008R2
  - V1
    - Windows 2008

Kde najdeme Powershell

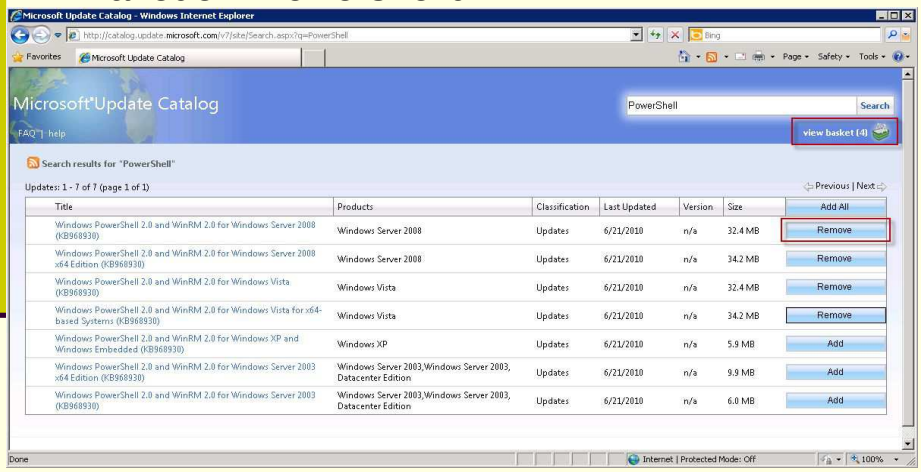
- Instalace WMF 3.0
  - Windows 7 Service Pack 1
  - Windows Server 2008 R2 SP1
  - Windows Server 2008 Service Pack 2

  http://www.microsoft.com/en-us/download/details.aspx?id=34595

  - Čtěte System Requirements
    - Kompatibilita

Kde najdeme Powershell

- Zavedení Powershellu



Kde najdeme Powershell

- Kompatibilita skriptů
  - Jazykové nuance
  - Novější cmdlety
  - Prevence
  - Aplikovaný Powershell

Kde najdeme Powershell

Kompatibilita skriptů – jazyk

```
Get-Process | where {$_.processname -like "*powers*"}

Get-Process | where processname -like "*powers*"

(Get-Process | foreach name).ToUpper()
```

Kde najdeme Powershell

Kompatibilita skriptů – nové cmdlety

```
Get-CimInstance

Connect-PSSession
Disconnect-PSSession

Update-Help

Invoke-WebRequest
```


Kde najdeme Powershell

Kompatibilita skriptů – vylepšené cmdlety

```
Add-Member -NotePropertyMembers $hashTable

Get-Credential -Message „“
```

Kde najdeme Powershell

Kompatibilita skriptů – ošetření

```
#requires -version 2

Help about_Requires

Powershell.exe -version 2.0
```

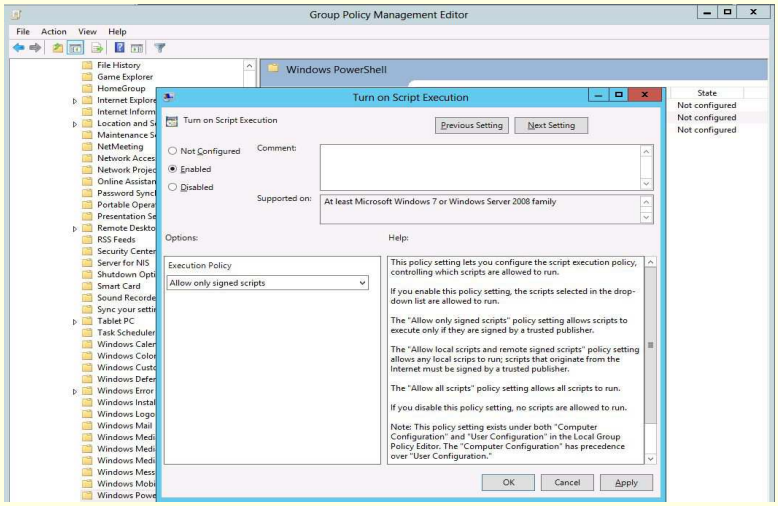

Kde najdeme Powershell

- Aplikovaný Powershell
  - Kompatibilita služby WSMan a klíčových knihoven

    http://blogs.msdn.com/b/powershell/archive/2012/12/20/windows-management-framework-3-0-compatibility-update.aspx
  - MS Exchange
  - MS SharePoint
  - SCOM...

Kde najdeme Powershell

- Konfigurace Powershellu



Kam dosáhne Powershell

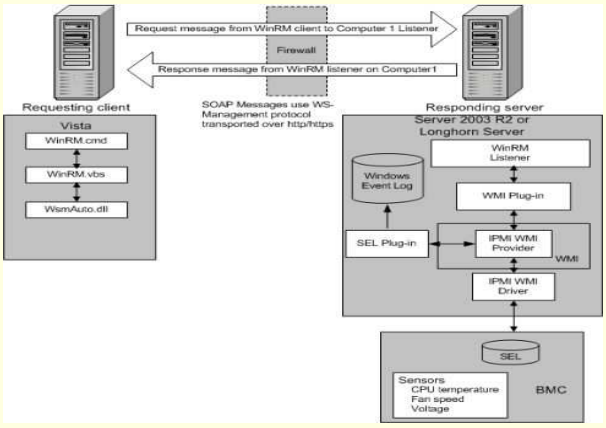
- Vzdálený přístup I.
  - WMI tradičně
    - RPC
  - WMI/CIM nově
    - WinRM: HTTP/S

Kam dosáhne Powershell

- Vzdálený přístup II.
  - Co nahradí PSExec?
  - winrm get wmi/root/cimv2/Win32_Service?Name=WinRM
  - Remoting v Powershellu

Kam dosáhne Powershell

Vzdálený přístup II.



Kam dosáhne Powershell

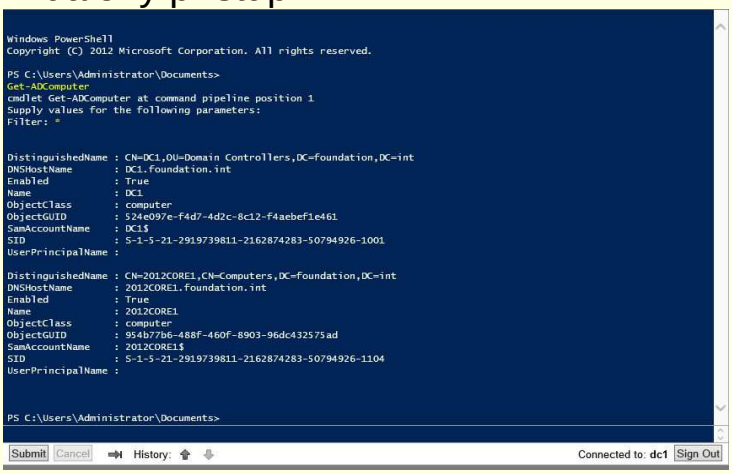
- Vzdálený přístup III.
  - Přímá relace
    - Enter-PSSession
  - Nepřímé spouštění
    - Invoke-Command

Kam dosáhne Powershell

- Vzdálený přístup IV.
  - Osvobození klientu
    - Powershell Web Access

Kam dosáhne Powershell

- Vzdálený přístup IV.



Po staru a v Powershellu

- Powershell není
  - Náhrada všech myslitelných programů pro konzoli
- Powershell je
  - Interpretovaný .NET Framework
  - Spouštěč tradičních konzolových programů

Po staru a v Powershellu

- Proč postaru
  - Rychlost kompilovaných nástrojů
    - Robocopy
    - Ldifde
    - Logparser
- Proč v Powershellu
  - Objektový výstup
  - Nezávislost na dodatečných nástrojích

Po staru a v Powershellu

- Co používají správci ve Windows

```
Systeminfo
Ipconfig
Tasklist/Taskkill
Netstat
Type
Net start/stop  Net Use Net User   Net Group
NsLookup
Ping/Tracert
GPResult
NetSh
```

Po staru a v Powershellu

- Co používají správci jinde (a ve Windows) I.

```
Tr => -replace
Sort => Sort-Object
Uniq => Sort-Object -Unique (Compare-Object)
```

```
find . -iname "*.png" -ls | awk '{s += $7} END {print s}'
                                  =>
Dir * -include *.png | measure-object length

hexdump  new.zip | head -5
                                  =>
Get-content new-zip -encoding byte | select -first 5
```


Po staru a v Powershellu

- Co používají správci jinde (a ve Windows) II.

```
history|awk '{print $2}'|awk 'BEGIN {FS="|"} {print $1}'|sort|
uniq -c|sort -r
                                  =>
History | select -expand commandline | group | sort count
-Desc
```

Po staru a v Powershellu

- Prohledáváme texty

```
ls | grep ".aspx"
=>
ls | where {$_ -match ".aspx"}
ls * -Fi *.ps1
```


Po staru a v Powershellu

- Prohledáváme texty

```
cat filename.txt | grep term > outputfile.txt
FINDSTR /A:2F /C:Open *.ps1 > outputfile.txt
=>
Select-String -Pattern "Open" -Path *.ps1
```

Po staru a v Powershellu

- Sítě: konfigurace

IPConfig =>

```
[System.Net.NetworkInformation.NetworkInterface]::
GetAllNetworkInterfaces()

[System.Net.NetworkInformation.NetworkInterface]::
GetAllNetworkInterfaces() |
foreach {$_.GetIPProperties()} |
select -exp DNSAddresses
```


Po staru a v Powershellu

- Sítě: konfigurace

IPConfig =>

```
Get-WmiObject win32_networkadapterconfiguration
```

Po staru a v Powershellu

- Sítě: konfigurace

```
NsLookup =>

[system.net.dns] | Get-Member -Static

[system.net.dns]::Resolve("localhost")
```


Po staru a v Powershellu

- Sítě: konfigurace

```
NsLookup =>

Add-Type -Path .\JHSoftware.DnsClient.dll
[JHSoftware.DnsClient] | Get-Member -Static
[JHSoftware.DnsClient]::Lookup("gopas.cz","A")
```

Po staru a v Powershellu

- Sítě: sledování

```
NetStat =>

[System.Net.NetworkInformation.IPGlobalProperties]::
GetIPGlobalProperties() | Get-Member

[System.Net.NetworkInformation.IPGlobalProperties]::
GetIPGlobalProperties().GetActiveTcpConnections()

[System.Net.NetworkInformation.IPGlobalProperties]::
GetIPGlobalProperties().GetActiveTcpListeners()
```


Po staru a v Powershellu

- Uživatelé a relace

```
WhoAmI =>
[System.Security.Principal.WindowsIdentity]::
GetCurrent()
```

Po staru a v Powershellu

- Uživatelé a relace

PsLoggedOn =>

```
$errorActionPreference = "silentlycontinue"
New-PSDrive HKU Registry HKEY_USERS
dir hku: | foreach {($_.name -split "\\")[1]} | foreach
{$objSID = New-Object
System.Security.Principal.SecurityIdentifier($_)
$objUser =
$objSID.Translate( [System.Security.Principal.NTAccount
])
$objUser.Value } | sort -unique
```


Po staru a v Powershellu

- Uživatelé a relace

PsLoggedOn =>

```
Get-WmiObject Win32_Process -filter "Name =
'explorer.exe'" | foreach -process {$_.GetOwner().User}

Get-WmiObject Win32_ComputerSystem | Select -property
UserName
```

Po staru a v Powershellu

- Uživatelé a relace

```
PsLoggedOn =>
$server = "localhost"
$adsi = [adsi]"WinNT://$server/LanmanServer"
$users = $adsi.psbase.invoke("resources") | foreach
{$_.gettype().invokeMember("user","GetProperty",$null,
$_,$null)}
$paths = $adsi.psbase.invoke("resources") | foreach
{$_.gettype().invokeMember("path","GetProperty",$null,
$_,$null)}
for($i = 0;$i -lt $users.length; $i++) { "{0}: {1}" -f
$users[$i],$paths[$i] }
```


Po staru a v Powershellu

- Uživatelé a relace

```
PsLoggedOn =>

Get-WmiObject win32_serverconnection
-computerName "localhost" |
Format-Table
connectionid,username,computername,sharename,activetime
,number* -auto
```

Po staru a v Powershellu

- Chybí vám sudo?

```
function elevate-process
{
$file, [string]$arguments = $args;
$psi = new-object System.Diagnostics.ProcessStartInfo
$file;
$psi.Arguments = $arguments;
$psi.Verb = "runas";
$psi.WorkingDirectory = get-location;
[System.Diagnostics.Process]::Start($psi);
}

set-alias sudo elevate-process
sudo notepad.exe
```


Po staru a v Powershellu

- Chybí vám sudo?

```
Invoke-ElevatedCommand
http://poshcode.org/2179

Get-Process | Invoke-ElevatedCommand.ps1 {
    $input | Where-Object { $_.Handles -gt 500 } } |
Sort Handles

Invoke-ElevatedCommand
http://poshcode.org/3158
```

Po staru a v Powershellu

- Využíváme WMI

```
$query = "SELECT * FROM Win32_Service WHERE
Name='AudioSrv'"
Get-WMIObject -Query $query

$query = "SELECT * FROM Win32_Service
WHERE (State='Running' AND StartMode='Manual')
AND (Name LIKE '[af]%')"
Get-WMIObject -Query $query
```


Po staru a v Powershellu

- Využíváme WMI

```
$query = "ASSOCIATORS OF
{Win32_Service.Name='LanmanWorkstation'} WHERE
ResultRole=Dependent"
Get-WMIObject -Query $query
```

Po staru a v Powershellu

- Využíváme WMI

```
Get-WmiObject win32_logicaldisk | foreach {

   $devid = $_.deviceid
   Get-WmiObject -Query "associators of
{win32_logicaldisk.deviceid='$($_.deviceid)'} where
AssocClass=Win32_LogicalDiskToPartition" |
foreach {add-member -inputobject $_ -name origDeviceId
-membertype noteproperty -value $devid -passthru}
} | select
Name,BootPartition,PrimaryPartition,Index,origDeviceID
```


Po staru a v Powershellu

- Využíváme WMI

```
foreach ($instance in (get-ciminstance
win32_diskdrive)) {
$devid = $instance.deviceid
$devid
Get-CimAssociatedInstance -CimInstance $instance
-resultclassname win32_diskpartition |
foreach {add-member -inputobject $_ -name origDeviceId
-membertype noteproperty -value $devid -passthru} |
select
Name,BootPartition,PrimaryPartition,Index,origDeviceID}
```

Po staru a v Powershellu

- Active Directory

```
CSVDE
LDIFDE
Dsadd
Dsmod
Dsrm
Dsmove
Dsget
Dsquery
Adfind
Admod
```


Po staru a v Powershellu

- Active Directory

```
Get- Set- New- ADUser (QADUser)
Rename- Move- Remove- QADObject
```

Po staru a v Powershellu

- Lesk a bída kopírování

```
Xcopy
Robocopy
Copy-Item

Xcacls, SetACL
Get-Acl | Set-Acl
```


Data v Powershellu

- Život v rouře
- Objektové proměnné
- Objektové konzervy

Data v Powershellu

- Živor v rouře
  - Vše v paměti
  - Využijte výhodu filtrování

```
dir * -Recurse -Include *.txt
dir * -Recurse | where {$_.extension -eq ".txt"}

cat .\spanish.dic
cat .\spanish.dic -ReadCount 1000000
```


Data v Powershellu

- Objektové proměnné
  - Vše v paměti
  - Pozor na velikost objektů

```
$textfile = cat .\spanish.dic
$textfile = cat .\spanish.dic -ReadCount 1000000
```

Data v Powershellu

- Objektové proměnné
  - Vše v paměti
  - Pozor na velikost objektů

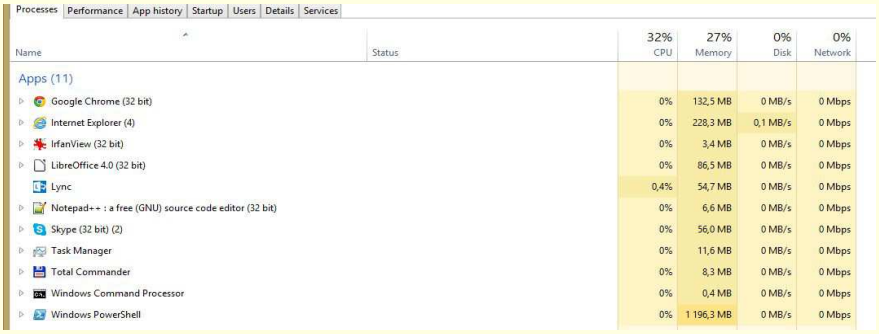
```
(dir .\JHSoftware.DnsClient.zip).length
275271

$binfile = cat JHSoftware.DnsClient.zip -Encoding byte
$binfile1 = cat JHSoftware.DnsClient.zip -Encoding byte

Get-Process powersh*
Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
    479      96  1269556    1287224  1783   140,96   5240 powershell
```


Data v Powershellu

- Objektové proměnné
  - Vše v paměti
  - Pozor na velikost objektů



Data v Powershellu

- Objektové konzervy

```
Export-Csv
Import-Csv
-Encoding -Append -Delimiter

Import-
Export-Clixml
-Depth -Encoding
```


Data v Powershellu

- Objektové konzervy

```
Select-String
-Pattern -CaseSensitive -List -Encoding -Context

Select-Xml
Select-Xml -Xml -XPath
```

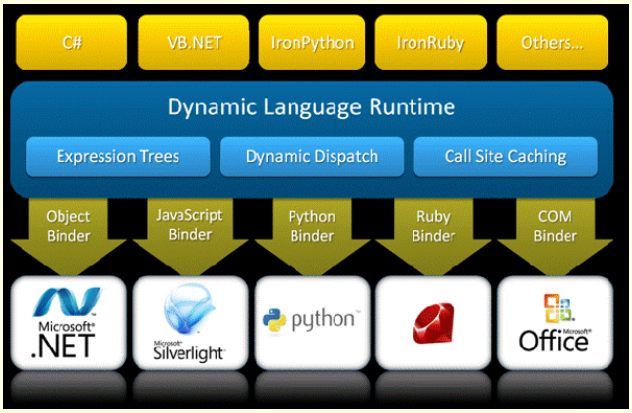
Powershellu... a co za to

- Powershell je:
  - Všestranný
  - Přenositelný
  - Ohebný
- Powershell nemůže být:
  - Nejlepší ve všem

Desetibojař nikdy nezaběhne stovku jako Bolt!

Powershellu... a co za to

- Interpretované prostředí



Powershellu... a co za to

- Co je jednoduché, nemusí být nejlepší
  - Pole
  - Řetězec

Powershellu... a co za to

- Co je jednoduché, nemusí být nejlepší

```
=>(cat .\spanish.txt | measure).count
10000

=>$pole = @()
=>measure-command {cat .\spanish.txt | foreach {
$pole += $_}}

Seconds           : 32
```

Powershellu... a co za to

- Co je jednoduché, nemusí být nejlepší

```
$pole1 = New-Object system.collections.arraylist

measure-command {cat .\spanish.txt | foreach
{ $pole1.Add($_)} }

Seconds           : 3
Milliseconds      : 957
```


Powershellu... a co za to

- Co je jednoduché, nemusí být nejlepší

```
=>$CelyText = ""
=>measure-command {cat .\spanish.txt | foreach {
$CelyText += $_}}
Seconds           : 29

=>$text1 = New-Object System.Text.StringBuilder
=>measure-command {cat .\spanish.txt | foreach {[void]
$text1.Append($_) }}
Seconds           : 3
Milliseconds      : 945
```

Powershellu... a co za to

- Objekty vždy a všude... a za co?
  - Tvorba objektů něco stojí
  - Mějte na paměti jejich počet

Powershellu... a co za to

- Objekty vždy a všude... a za co?

```
=>(dir .\spanish.dic).length / 1MB
4,07346725463867
=>cat .\spanish.dic | Measure-Object
Count    : 370470

=>measure-command {cat .\spanish.dic}
Seconds           : 34

=>measure-command {cat .\spanish.dic -ReadCount
1000000}
Seconds           : 0
Milliseconds      : 845
```

Powershellu... a co za to

- Objekty vždy a všude... a za co?

```
=>measure-command {
cat .\spanish.dic | foreach {$_ -replace "a","aa"}}
Minutes           : 2
Seconds           : 52

=>measure-command {cat .\spanish.dic -ReadCount 1000000
| foreach {$_ -replace "a","aa"}}
Minutes           : 0
Seconds           : 4
```


Powershellu... a co za to

- Je objektová roura zadarmo?

```
=>$cisla = 1..20000

=>Measure-Command {$cisla | ForEach-Object {$_ *
2264} }
Seconds           : 6
Milliseconds      : 344

=>Measure-Command {ForEach ($cislo in $cisla)
{$cislo * 2264} }
Seconds           : 0
Milliseconds      : 116
```

Zdroje informací

Internet

- MS PowerShell blog
  - http://blogs.msdn.com/b/powershell/
- Script Center
  - http://technet.microsoft.com/en-us/scriptcenter/bb410849.aspx
- Windows PowerShell Support for Windows Server 2012
  - http://technet.microsoft.com/en-us/library/hh801904.aspx

Dotazy

- … a diskuse

Další informace

- Autor
  www.patrikmalina.eu
- Kontat
  it@patrikmalina.eu