**[PowerShell – Program List – 32/64 Bit – Local/Remote Machies](http://sqlpowershell.wordpress.com/2014/01/20/powershell-program-list-3264-bit-localremote-machies/)**

Posted on [January 20, 2014](http://sqlpowershell.wordpress.com/2014/01/20/powershell-program-list-3264-bit-localremote-machies/) by [Prashanth Jayaram](http://sqlpowershell.wordpress.com/author/prashanthjayaram/)

This Powershell script list all the installed application on both 32 and 64 bit applications, particularly useful for people managing both 32-bit and 64-bit applications.   This code also contains an exclusion array where you can exclude list of program that you don’t want to show.

* The function allows -ComputerName parameter so you can connect to any machines
* Application architecture detection (32-bit or 64-bit) using Win32\_processor
* Output is Powershell console and GridView

Function call:- The computer name is mandatory to retrive the software list.

Get-InstalledApplication -Computername <Computername>

Code:

Function Get-InstalledApplication

{

Param(

[Parameter(Mandatory=$true)]

[string[]]$Computername)

#Registry Hives

$Object =@()

$excludeArray = ("Security Update for Windows",

"Update for Windows",

"Update for Microsoft .NET",

"Security Update for Microsoft",

"Hotfix for Windows",

"Hotfix for Microsoft .NET Framework",

"Hotfix for Microsoft Visual Studio 2007 Tools",

"Hotfix")

[long]$HIVE\_HKROOT = 2147483648

[long]$HIVE\_HKCU = 2147483649

[long]$HIVE\_HKLM = 2147483650

[long]$HIVE\_HKU = 2147483651

[long]$HIVE\_HKCC = 2147483653

[long]$HIVE\_HKDD = 2147483654

Foreach($EachServer in $Computername){

$Query = Get-WmiObject -ComputerName $Computername -query "Select AddressWidth, DataWidth,Architecture from Win32\_Processor"

foreach ($i in $Query)

{

 If($i.AddressWidth -eq 64){

 $OSArch='64-bit'

 }

Else{

$OSArch='32-bit'

}

}

Switch ($OSArch)

{

 "64-bit"{

$RegProv = GWMI -Namespace "root\Default" -list -computername $EachServer| where{$\_.Name -eq "StdRegProv"}

$Hive = $HIVE\_HKLM

$RegKey\_64BitApps\_64BitOS = "Software\Microsoft\Windows\CurrentVersion\Uninstall"

$RegKey\_32BitApps\_64BitOS = "Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall"

$RegKey\_32BitApps\_32BitOS = "Software\Microsoft\Windows\CurrentVersion\Uninstall"

#############################################################################

# Get SubKey names

$SubKeys = $RegProv.EnumKey($HIVE, $RegKey\_64BitApps\_64BitOS)

# Make Sure No Error when Reading Registry

if ($SubKeys.ReturnValue -eq 0)

{  # Loop Through All Returned SubKEys

ForEach ($Name in $SubKeys.sNames)

 {

$SubKey = "$RegKey\_64BitApps\_64BitOS\$Name"

$ValueName = "DisplayName"

$ValuesReturned = $RegProv.GetStringValue($Hive, $SubKey, $ValueName)

$AppName = $ValuesReturned.sValue

$Version = ($RegProv.GetStringValue($Hive, $SubKey, "DisplayVersion")).sValue

$Publisher = ($RegProv.GetStringValue($Hive, $SubKey, "Publisher")).sValue

$donotwrite = $false

if($AppName.length -gt "0"){

 Foreach($exclude in $excludeArray)

                        {

                        if($AppName.StartsWith($exclude) -eq $TRUE)

                            {

                            $donotwrite = $true

                            break

                            }

                        }

            if ($donotwrite -eq $false)

                        {

            $Object += New-Object PSObject -Property @{

            Appication = $AppName;

            Architecture  = "64-BIT";

            ServerName = $EachServer;

            Version = $Version;

            Publisher= $Publisher;

           }

                        }

}

  }}

#############################################################################

$SubKeys = $RegProv.EnumKey($HIVE, $RegKey\_32BitApps\_64BitOS)

# Make Sure No Error when Reading Registry

if ($SubKeys.ReturnValue -eq 0)

{

  # Loop Through All Returned SubKEys

  ForEach ($Name in $SubKeys.sNames)

  {

    $SubKey = "$RegKey\_32BitApps\_64BitOS\$Name"

$ValueName = "DisplayName"

$ValuesReturned = $RegProv.GetStringValue($Hive, $SubKey, $ValueName)

$AppName = $ValuesReturned.sValue

$Version = ($RegProv.GetStringValue($Hive, $SubKey, "DisplayVersion")).sValue

$Publisher = ($RegProv.GetStringValue($Hive, $SubKey, "Publisher")).sValue

 $donotwrite = $false

if($AppName.length -gt "0"){

 Foreach($exclude in $excludeArray)

                        {

                        if($AppName.StartsWith($exclude) -eq $TRUE)

                            {

                            $donotwrite = $true

                            break

                            }

                        }

            if ($donotwrite -eq $false)

                        {

            $Object += New-Object PSObject -Property @{

            Appication = $AppName;

            Architecture  = "32-BIT";

            ServerName = $EachServer;

            Version = $Version;

            Publisher= $Publisher;

           }

                        }

           }

    }

}

} #End of 64 Bit

######################################################################################

###########################################################################################

"32-bit"{

$RegProv = GWMI -Namespace "root\Default" -list -computername $EachServer| where{$\_.Name -eq "StdRegProv"}

$Hive = $HIVE\_HKLM

$RegKey\_32BitApps\_32BitOS = "Software\Microsoft\Windows\CurrentVersion\Uninstall"

#############################################################################

# Get SubKey names

$SubKeys = $RegProv.EnumKey($HIVE, $RegKey\_32BitApps\_32BitOS)

# Make Sure No Error when Reading Registry

if ($SubKeys.ReturnValue -eq 0)

{  # Loop Through All Returned SubKEys

  ForEach ($Name in $SubKeys.sNames)

  {

$SubKey = "$RegKey\_32BitApps\_32BitOS\$Name"

$ValueName = "DisplayName"

$ValuesReturned = $RegProv.GetStringValue($Hive, $SubKey, $ValueName)

$AppName = $ValuesReturned.sValue

$Version = ($RegProv.GetStringValue($Hive, $SubKey, "DisplayVersion")).sValue

$Publisher = ($RegProv.GetStringValue($Hive, $SubKey, "Publisher")).sValue

if($AppName.length -gt "0"){

$Object += New-Object PSObject -Property @{

            Appication = $AppName;

            Architecture  = "32-BIT";

            ServerName = $EachServer;

            Version = $Version;

            Publisher= $Publisher;

           }

           }

  }}

}#End of 32 bit

} # End of Switch

}

#$AppsReport

$column1 = @{expression="ServerName"; width=15; label="Name"; alignment="left"}

$column2 = @{expression="Architecture"; width=10; label="32/64 Bit"; alignment="left"}

$column3 = @{expression="Appication"; width=80; label="Appication"; alignment="left"}

$column4 = @{expression="Version"; width=15; label="Version"; alignment="left"}

$column5 = @{expression="Publisher"; width=30; label="Publisher"; alignment="left"}

"#"\*80

"Installed Software Application Report"

"Numner of Installed Application count : $($object.count)"

"Generated $(get-date)"

"Generated from $(gc env:computername)"

"#"\*80

$object |Format-Table $column1, $column2, $column3 ,$column4, $column5

$object|Out-GridView

}