

```
function Enable-Privilege {  
    param(  
        ## The privilege to adjust. This set is  
        taken from  
        ##  
        http://msdn.microsoft.com/en-us/library/bb530716\(VS.85\).aspx  
        [ValidateSet(  
            "SeAssignPrimaryTokenPrivilege",  
            "SeAuditPrivilege", "SeBackupPrivilege",  
            "SeChangeNotifyPrivilege",  
            "SeCreateGlobalPrivilege",  
            "SeCreatePagefilePrivilege",  
            "SeCreatePermanentPrivilege",  
            "SeCreateSymbolicLinkPrivilege",  
            "SeCreateTokenPrivilege",  
            "SeDebugPrivilege",  
            "SeEnableDelegationPrivilege",  
            "SeImpersonatePrivilege",  
            "SeIncreaseBasePriorityPrivilege",  
            "SeIncreaseQuotaPrivilege",  
            "SeIncreaseWorkingSetPrivilege",  
            "SeLoadDriverPrivilege",  
            "SeLockMemoryPrivilege",  
            "SeMachineAccountPrivilege",  
            "SeManageVolumePrivilege",  
            "SeProfileSingleProcessPrivilege",
```

```
"SeRelabelPrivilege",
"SeRemoteShutdownPrivilege",
    "SeRestorePrivilege",
"SeSecurityPrivilege", "SeShutdownPrivilege",
"SeSyncAgentPrivilege",
    "SeSystemEnvironmentPrivilege",
"SeSystemProfilePrivilege",
"SeSystemtimePrivilege",
    "SeTakeOwnershipPrivilege",
"SeTcbPrivilege", "SeTimeZonePrivilege",
"SeTrustedCredManAccessPrivilege",
    "SeUndockPrivilege",
"SeUnsolicitedInputPrivilege"))]
$Privilege,
## The process on which to adjust the
privilege. Defaults to the current process.
$ProcessId = $pid,
## Switch to disable the privilege, rather
than enable it.
[Switch] $Disable
)
```

```
## Taken from P/Invoke.NET with minor
adjustments.
$definition = @'
using System;
using System.Runtime.InteropServices;
```

```

public class AdjPriv
{
    [DllImport("advapi32.dll", ExactSpelling =
true, SetLastError = true)]
    internal static extern bool
AdjustTokenPrivileges(IntPtr htok, bool
disall,
    ref TokPriv1Luid newst, int len, IntPtr
prev, IntPtr relen);

    [DllImport("advapi32.dll", ExactSpelling =
true, SetLastError = true)]
    internal static extern bool
OpenProcessToken(IntPtr h, int acc, ref IntPtr
phtok);
    [DllImport("advapi32.dll", SetLastError =
true)]
    internal static extern bool
LookupPrivilegeValue(string host, string name,
ref long pluid);
    [StructLayout(LayoutKind.Sequential, Pack =
1)]
    internal struct TokPriv1Luid
    {
        public int Count;
        public long Luid;
    }
}

```

```

    public int Attr;
}

    internal const int SE_PRIVILEGE_ENABLED =
0x00000002;
    internal const int SE_PRIVILEGE_DISABLED =
0x00000000;
    internal const int TOKEN_QUERY = 0x00000008;
    internal const int TOKEN_ADJUST_PRIVILEGES =
0x00000020;
    public static bool EnablePrivilege(long
processHandle, string privilege, bool disable)
    {
        bool retVal;
        TokPriv1Luid tp;
        IntPtr hproc = new IntPtr(processHandle);
        IntPtr htok = IntPtr.Zero;
        retVal = OpenProcessToken(hproc,
TOKEN_ADJUST_PRIVILEGES | TOKEN_QUERY, ref
htok);
        tp.Count = 1;
        tp.Luid = 0;
        if(disable)
        {
            tp.Attr = SE_PRIVILEGE_DISABLED;
        }
        else

```

```

    {
        tp.Attr = SE_PRIVILEGE_ENABLED;
    }
    retVal = LookupPrivilegeValue(null,
privilege, ref tp.Luid);
    retVal = AdjustTokenPrivileges(htok, false,
ref tp, 0, IntPtr.Zero, IntPtr.Zero);
    return retVal;
}
}
'@

```

```

$processHandle = (Get-Process -id
$ProcessId).Handle
$type = Add-Type $definition -PassThru
$type[0]::EnablePrivilege($processHandle,
$Privilege, $Disable)
}

```

Enable-Privilege SeTakeOwnershipPrivilege

```

# Change Owner to the local Administrators
group
$regKey =
[Microsoft.Win32.Registry]::LocalMachine.OpenS
ubKey("SOFTWARE\Microsoft\Windows
NT\CurrentVersion\NetworkList\DefaultMediaCost

```

```
",[Microsoft.Win32.RegistryKeyPermissionCheck]
::ReadWriteSubTree,[System.Security.AccessCont
rol.RegistryRights]::TakeOwnership)
$regACL = $regKey.GetAccessControl()
$regACL.SetOwner([System.Security.Principal.NT
Account]"Administrators")
$regKey.SetAccessControl($regACL)
# Change Permissions for the local
Administrators group
$regKey =
[Microsoft.Win32.Registry]::LocalMachine.OpenS
ubKey("SOFTWARE\Microsoft\Windows
NT\CurrentVersion\NetworkList\DefaultMediaCost
",[Microsoft.Win32.RegistryKeyPermissionCheck]
::ReadWriteSubTree,[System.Security.AccessCont
rol.RegistryRights]::ChangePermissions)
$regACL = $regKey.GetAccessControl()
$regRule = New-Object
System.Security.AccessControl.RegistryAccessRu
le
("Administrators","FullControl","ContainerInhe
rit","None","Allow")
$regACL.SetAccessRule($regRule)
$regKey.SetAccessControl($regACL)
```

```
$key='HKLM:SOFTWARE\Microsoft\Windows
```

```
NT\CurrentVersion\NetworkList\DefaultMediaCost
,
$exists=Test-Path -Path $key
if (!$exists) {$null = New-Item -Path $key
-Force}
'Name,Value,Type
3G,1,DWORD
4G,1,DWORD'| ConvertFrom-Csv |
Set-ItemProperty -Path $key -Name {$_.Name}
```