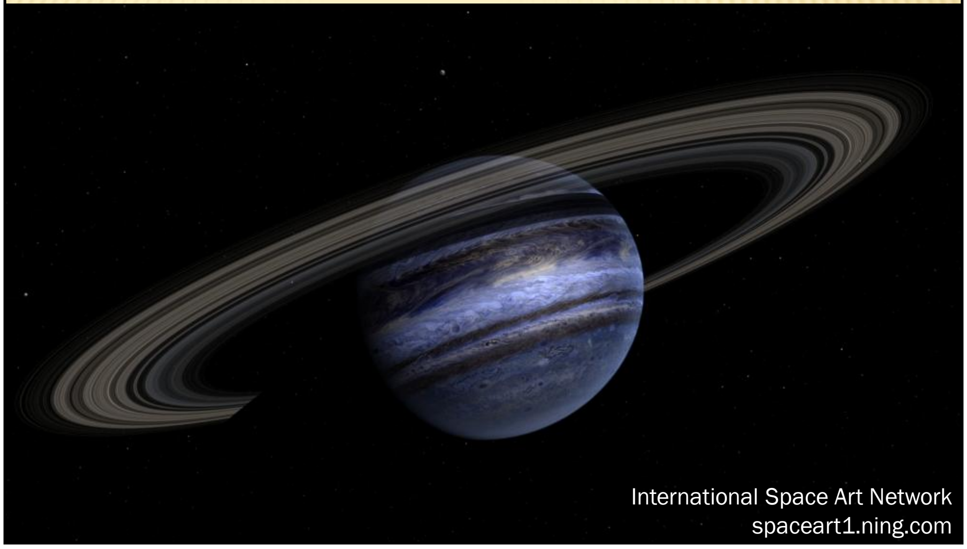
Bob McCoy
Radio Engineering Industries, Inc.
bmccoy@radioeng.com

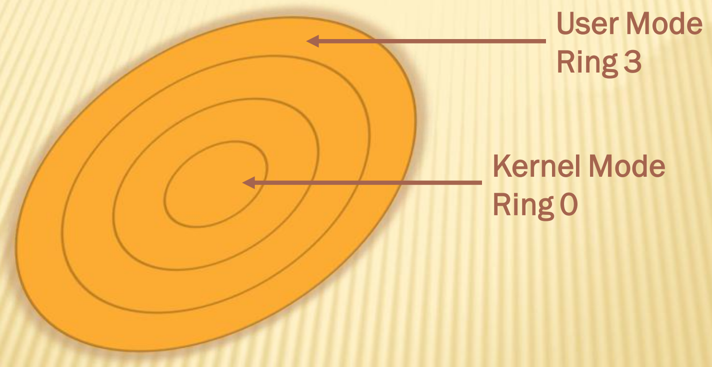
“NO SUDO FOR YOU!”

RINGS



International Space Art Network
spaceart1.ning.com

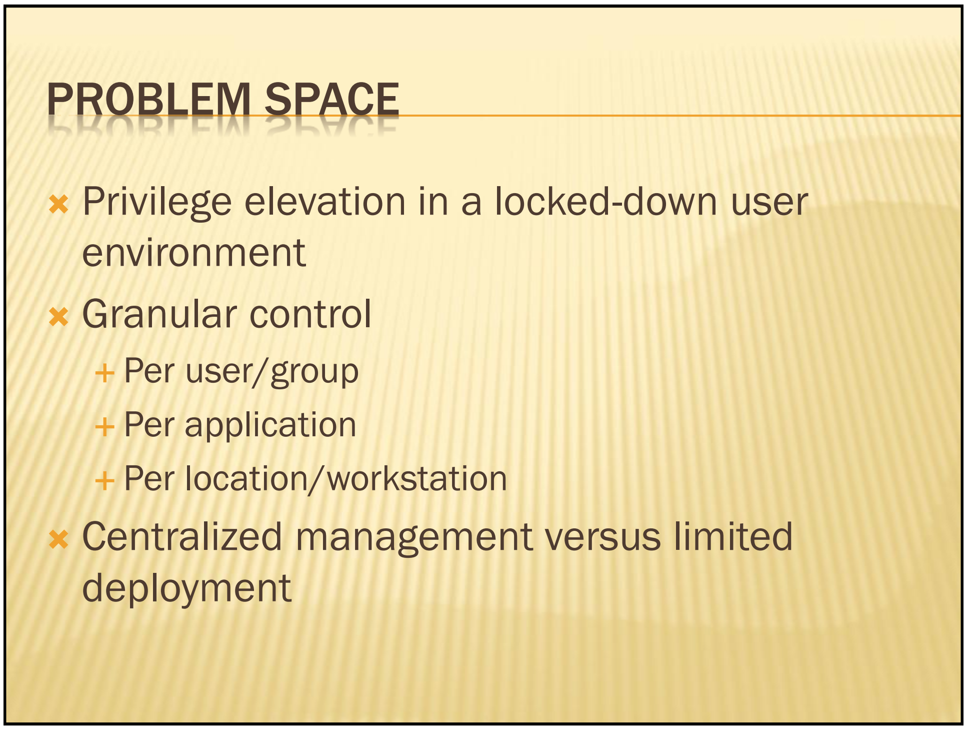
RINGS



Hierarchical Protection Domains
a.k.a. Protection Rings

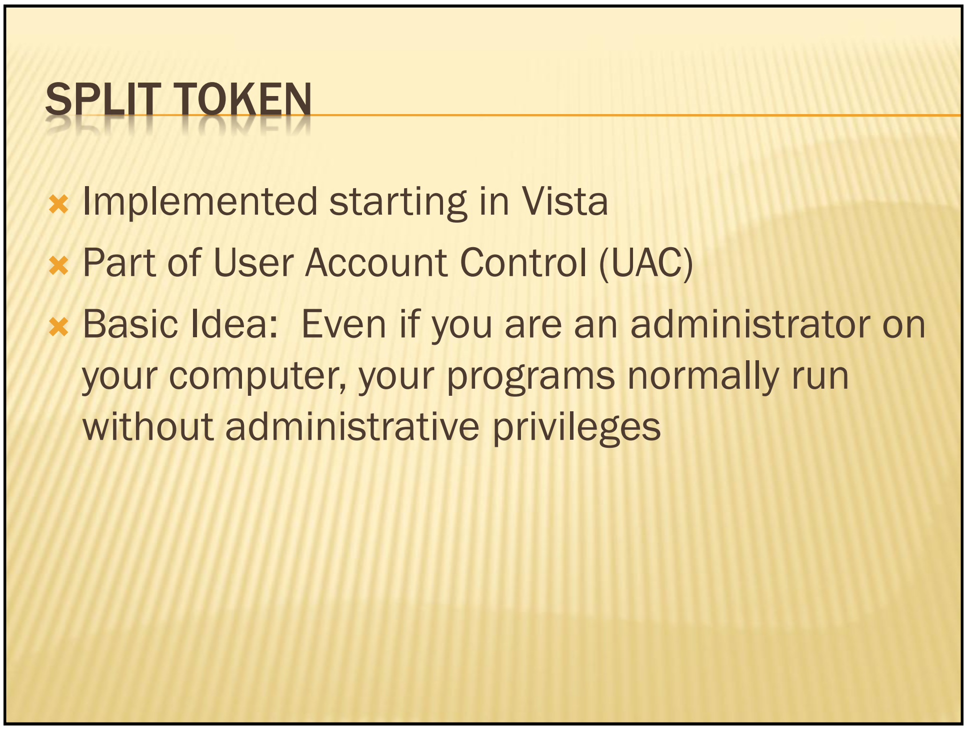
PROBLEM DEFINITION

- Video playback software
- USB-connected hard drive
- Special format optimized for video recording
- Driver loaded at run-time to provide block-level access to the hard drive
- Monolithic driver – runs in the user context
  - Admin passes
  - Normal user fails



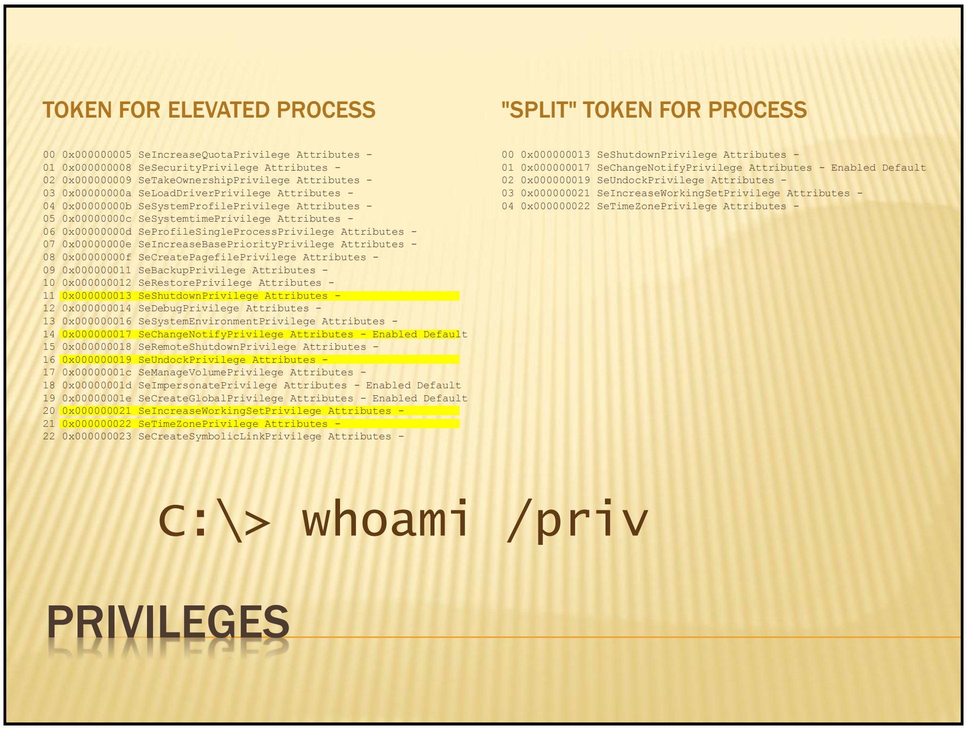
PROBLEM SPACE

- Privilege elevation in a locked-down user environment
- Granular control
  - Per user/group
  - Per application
  - Per location/workstation
- Centralized management versus limited deployment



SPLIT TOKEN

- Implemented starting in Vista
- Part of User Account Control (UAC)
- Basic Idea: Even if you are an administrator on your computer, your programs normally run without administrative privileges



PRIVILEGES

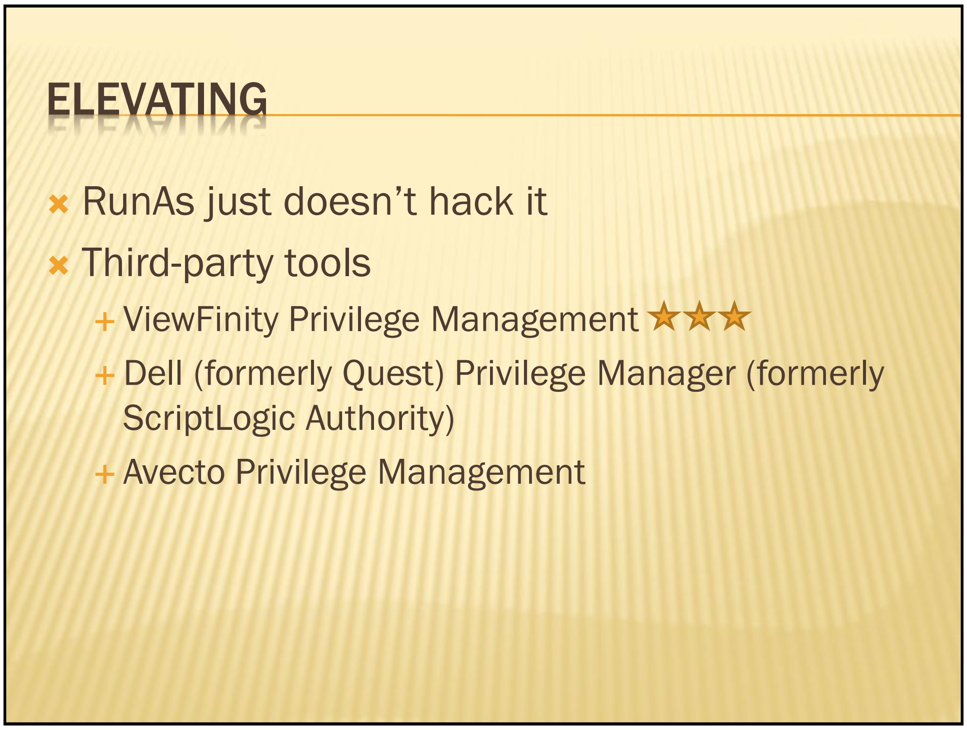
TOKEN FOR ELEVATED PROCESS

```
00 0x000000005 SeIncreaseQuotaPrivilege Attributes -
01 0x000000008 SeSecurityPrivilege Attributes -
02 0x000000009 SeTakeOwnershipPrivilege Attributes -
03 0x00000000a SeLoadDriverPrivilege Attributes -
04 0x00000000b SeSystemProfilePrivilege Attributes -
05 0x00000000c SeSystemtimePrivilege Attributes -
06 0x00000000d SeProfileSingleProcessPrivilege Attributes -
07 0x00000000e SeIncreaseBasePriorityPrivilege Attributes -
08 0x00000000f SeCreatePagefilePrivilege Attributes -
09 0x000000011 SeBackupPrivilege Attributes -
10 0x000000012 SeRestorePrivilege Attributes -
11 0x000000013 SeShutdownPrivilege Attributes -
12 0x000000014 SeDebugPrivilege Attributes -
13 0x000000016 SeSystemEnvironmentPrivilege Attributes -
14 0x000000017 SeChangeNotifyPrivilege Attributes - Enabled Default
15 0x000000018 SeRemoteShutdownPrivilege Attributes -
16 0x000000019 SeUndockPrivilege Attributes -
17 0x00000001c SeManageVolumePrivilege Attributes -
18 0x00000001d SeImpersonatePrivilege Attributes - Enabled Default
19 0x00000001e SeCreateGlobalPrivilege Attributes - Enabled Default
20 0x000000021 SeIncreaseWorkingSetPrivilege Attributes -
21 0x000000022 SeTimeZonePrivilege Attributes -
22 0x000000023 SeCreateSymbolicLinkPrivilege Attributes -
```

"SPLIT" TOKEN FOR PROCESS

```
00 0x000000013 SeShutdownPrivilege Attributes -
01 0x000000017 SeChangeNotifyPrivilege Attributes - Enabled Default
02 0x000000019 SeUndockPrivilege Attributes -
03 0x000000021 SeIncreaseWorkingSetPrivilege Attributes -
04 0x000000022 SeTimeZonePrivilege Attributes -
```

```
C:\> whoami /priv
```



ELEVATING

- RunAs just doesn't hack it
- Third-party tools
  - ViewFinity Privilege Management ★★★
  - Dell (formerly Quest) Privilege Manager (formerly ScriptLogic Authority)
  - Avecto Privilege Management

ELEVATE.EXE

- Johannes Passing's Blog
- “Launch elevated processes from the command line” http://jpassing.com/2007/12/08/launch-elevated-processes-from-the-command-line/
- Still prompts for credentials if not an admin

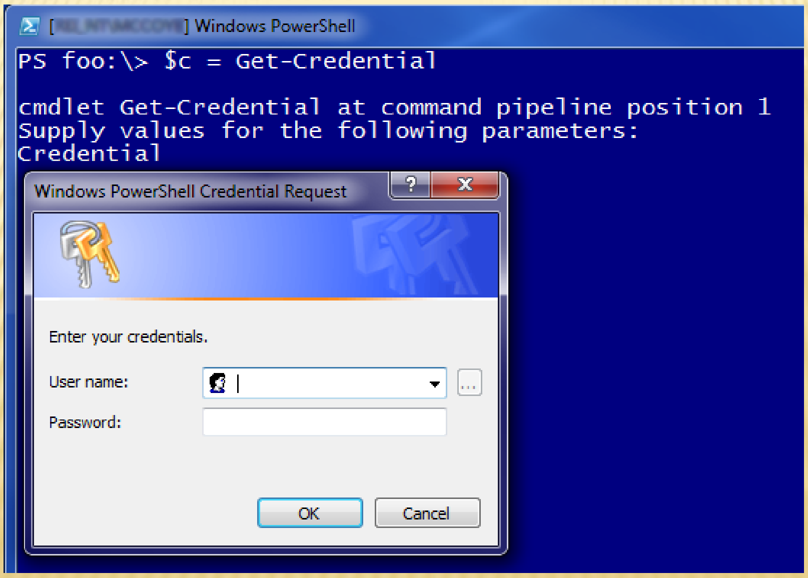
ELEVATION GOALS

- End user must be a standard user – local admin rights was not acceptable
- Inexpensive to free. ViewFinity was about $40 per license, but required 100 licenses.
- Simple to no supporting infrastructure. Rules out AD in many cases.
- Installed/configured by trusted administrator
- Should not provide a way of determining the alternative credentials

ALTERNATIVE CREDENTIALS IN POWERSHELL

- –Credential parameter
- Limited to Get-WMIObject in V1
- Greatly expanded in later versions
- Run a cmdlet with alternate credentials

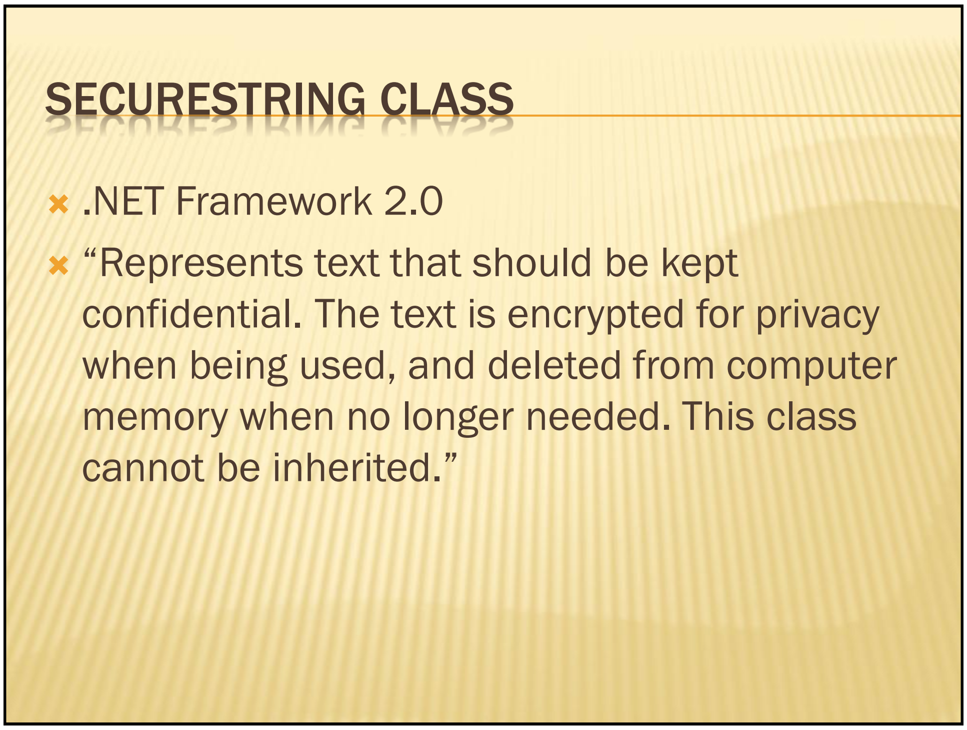
GET-CREDENTIAL



[[redacted]] Windows PowerShell

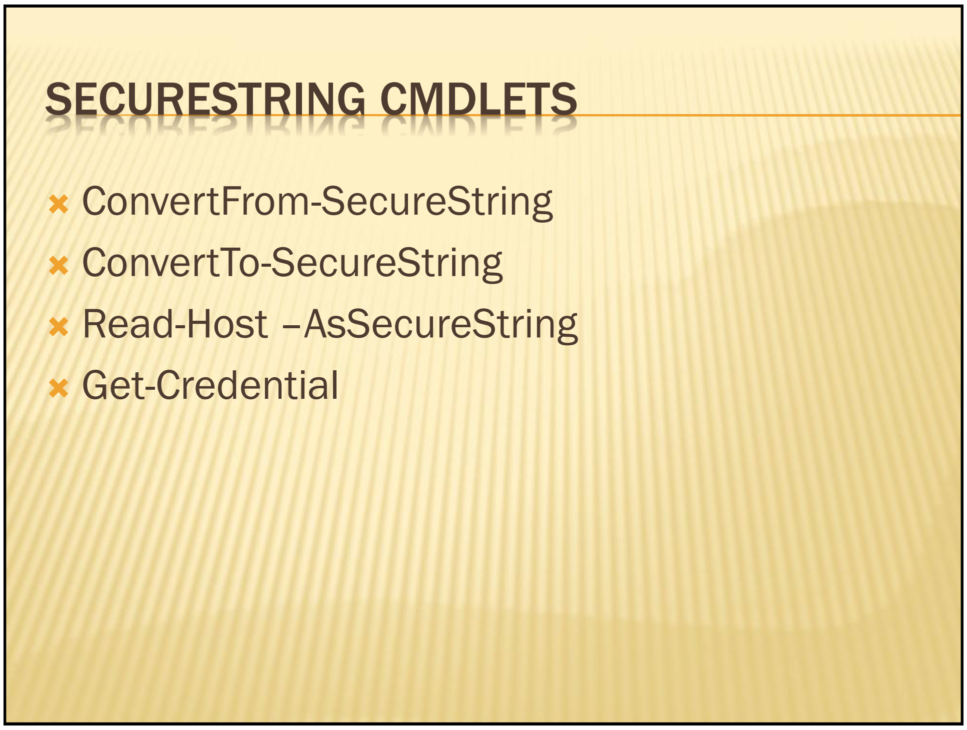
```
PS foo:\> $c = Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
```



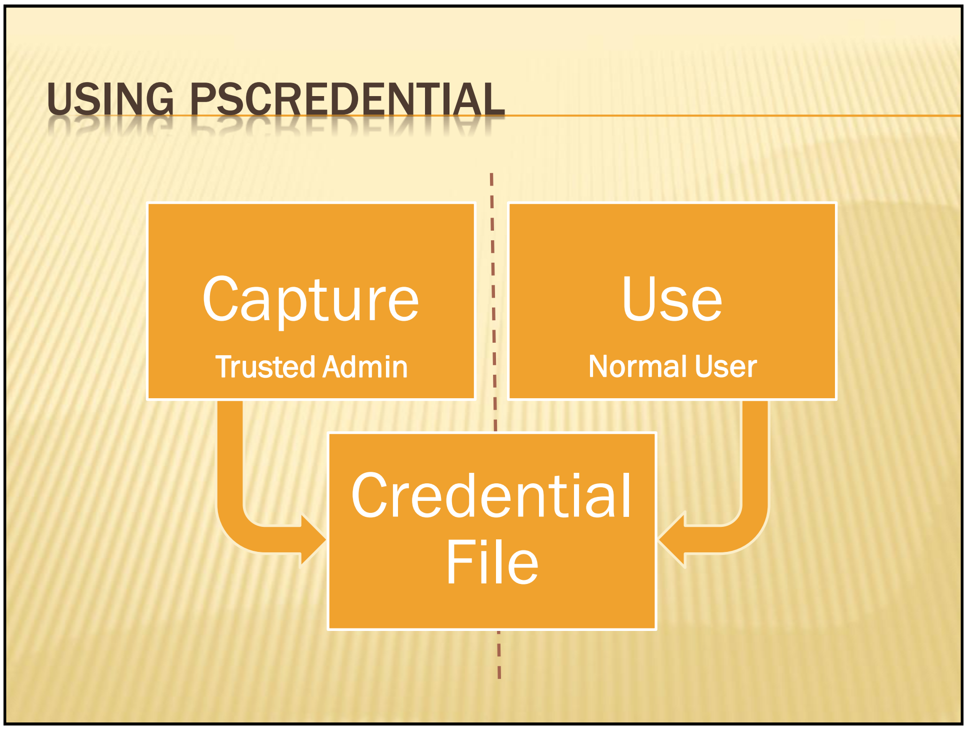
SECURESTRING CLASS

- .NET Framework 2.0
- “Represents text that should be kept confidential. The text is encrypted for privacy when being used, and deleted from computer memory when no longer needed. This class cannot be inherited.”

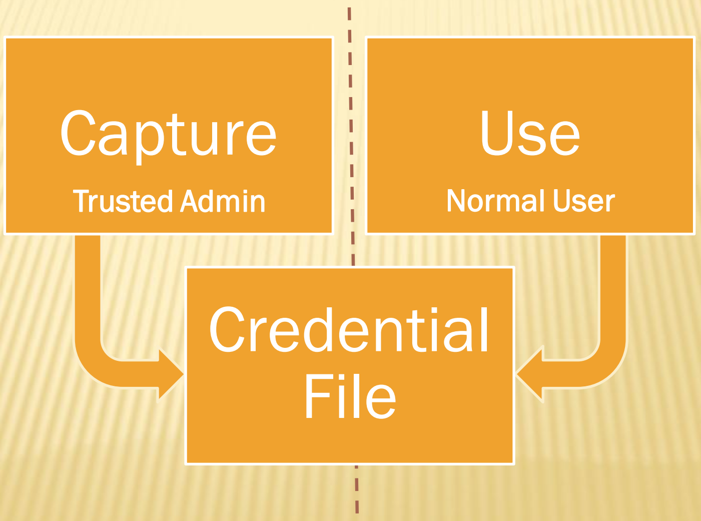


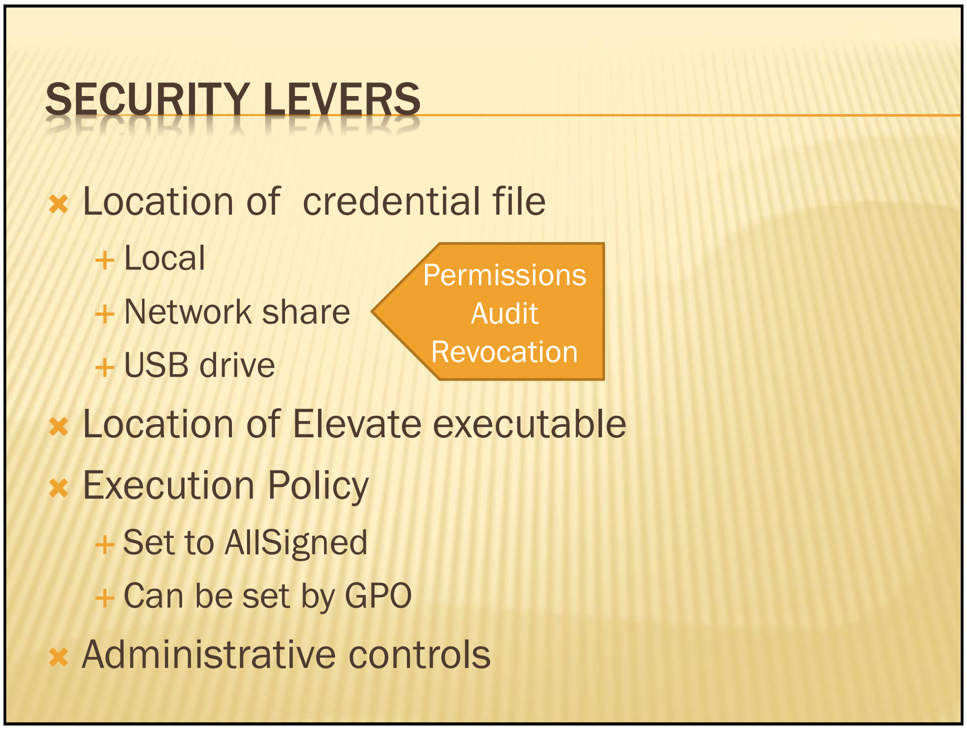
SECURESTRING CMDLETS

- ConvertFrom-SecureString
- ConvertTo-SecureString
- Read-Host –AsSecureString
- Get-Credential



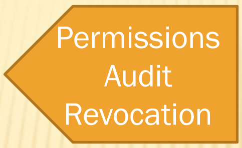
USING PSCREDENTIAL



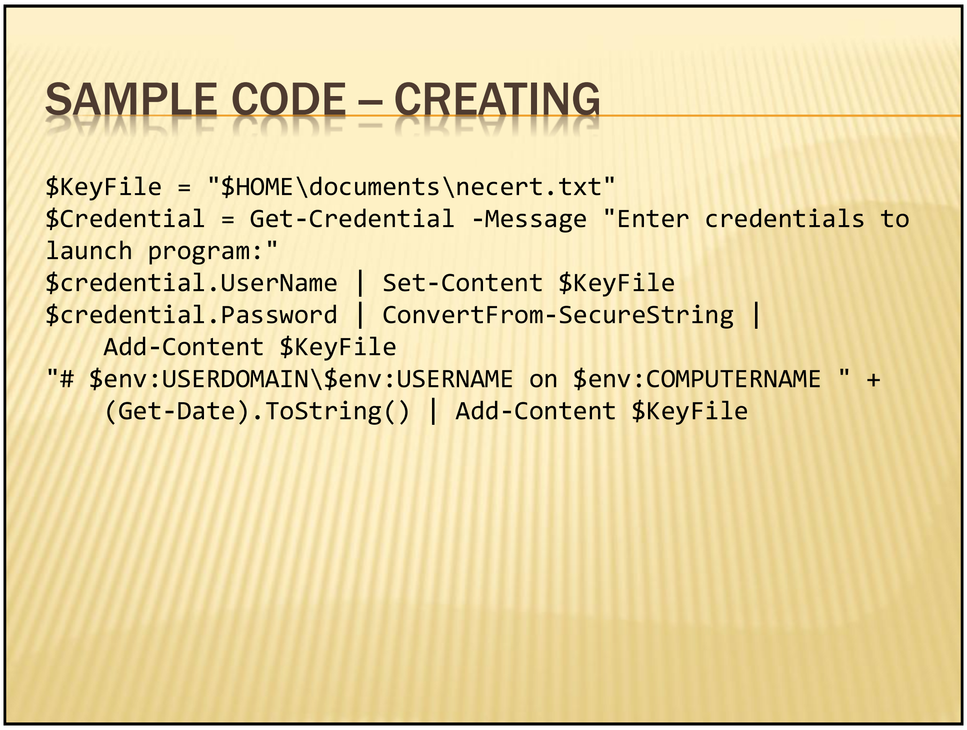


SECURITY LEVERS

- Location of credential file
  - Local
  - Network share
  - USB drive

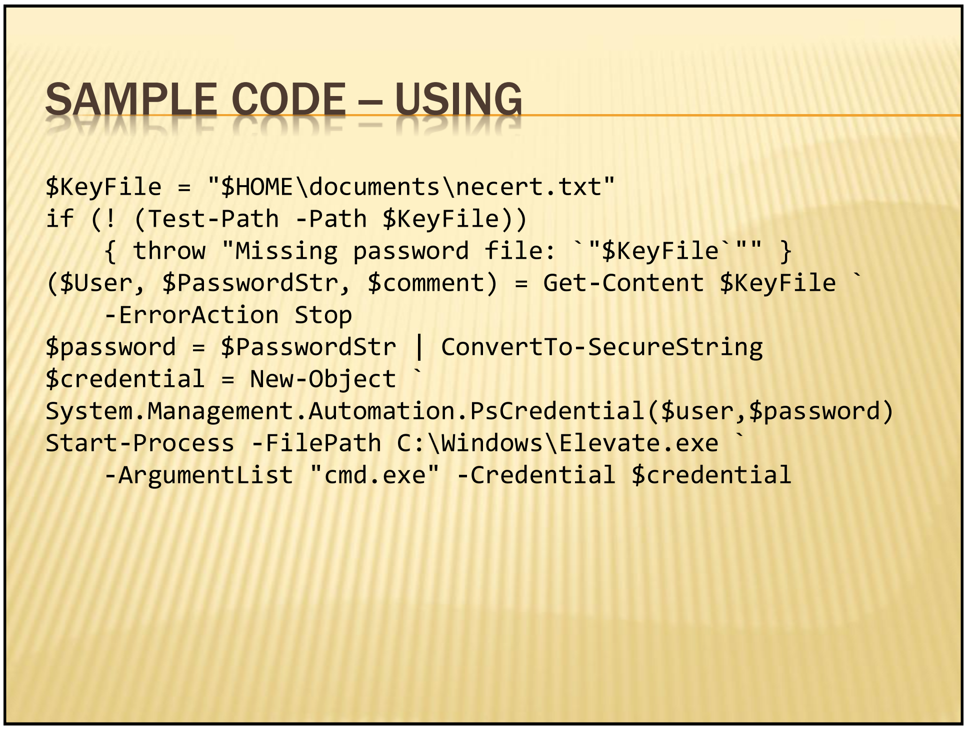


- Location of Elevate executable
- Execution Policy
  - Set to AllSigned
  - Can be set by GPO
- Administrative controls



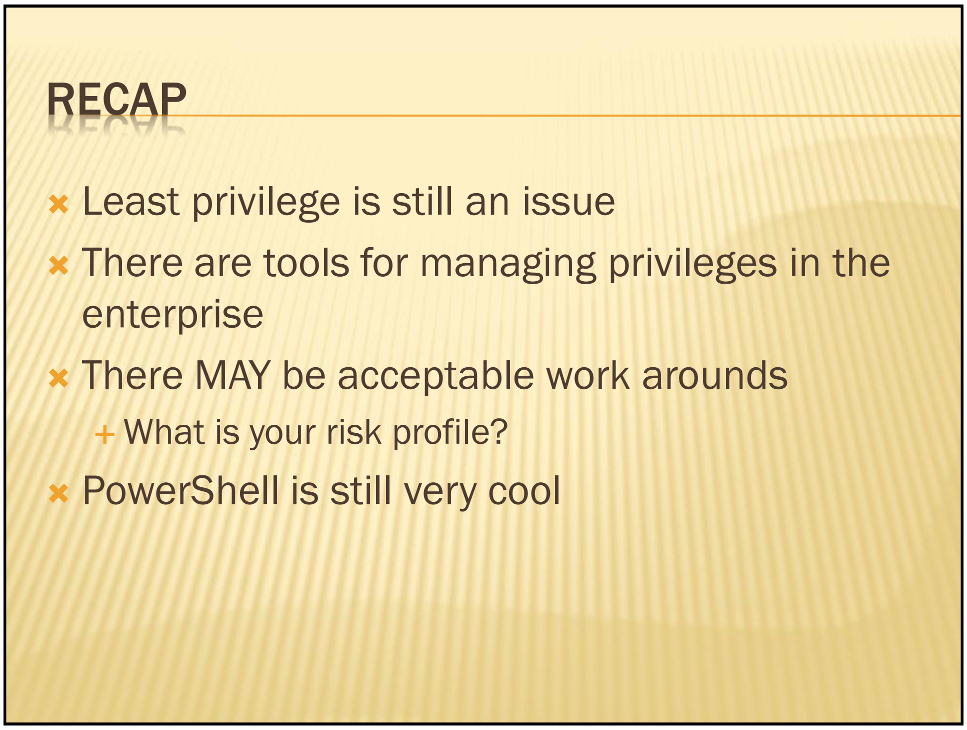
SAMPLE CODE – CREATING

```
$KeyFile = "$HOME\documents\necert.txt"
$Credential = Get-Credential -Message "Enter credentials to
launch program:"
$credential.UserName | Set-Content $KeyFile
$credential.Password | ConvertFrom-SecureString |
    Add-Content $KeyFile
"# $env:USERDOMAIN\$env:USERNAME on $env:COMPUTERNAME " +
    (Get-Date).ToString() | Add-Content $KeyFile
```



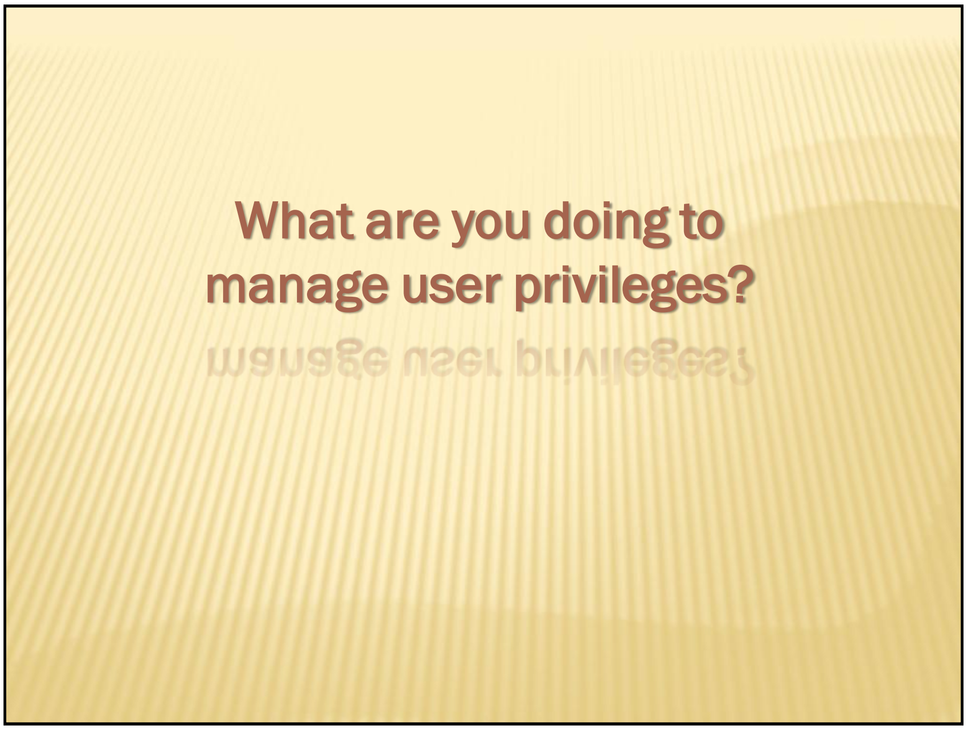
SAMPLE CODE – USING

```
$KeyFile = "$HOME\documents\necert.txt"
if (! (Test-Path -Path $KeyFile))
    { throw "Missing password file: `"$KeyFile`"" }
($User, $PasswordStr, $comment) = Get-Content $KeyFile `
    -ErrorAction Stop
$password = $PasswordStr | ConvertTo-SecureString
$credential = New-Object `
System.Management.Automation.PsCredential($user,$password)
Start-Process -FilePath C:\Windows\Elevate.exe `
    -ArgumentList "cmd.exe" -Credential $credential
```



RECAP

- Least privilege is still an issue
- There are tools for managing privileges in the enterprise
- There MAY be acceptable work arounds
  - What is your risk profile?
- PowerShell is still very cool



What are you doing to manage user privileges?