

بسمه تعالی



مرکز مدیریت امداد و هماهنگی  
عملیات رخدادهای رایانه ای

راهنمای امن سازی پاورشل در ویندوز

---



PowerShell

## مقدمه

امن‌سازی پاورشل در سیستم‌عامل ویندوز از اهمیت زیادی برخوردار است زیرا پاورشل علاوه بر آن که ابزاری برای نوشتن اسکریپت‌های مختلف است، یک شیله خط فرمان هم می‌باشد. این ابزار، دسترسی نامحدود به منابع سیستم‌عامل ویندوز را فراهم کرده و می‌تواند برنامه‌هایی نظیر Exchange را خودکار کند. از این رو مهاجمان از پاورشل در حملات خود به‌صورت گسترده بهره‌برداری می‌کنند. بنابراین ضروری است از روش‌هایی برای مقابله با بهره‌برداری از پاورشل در حملات مهاجمان استفاده شود.

یکی از روش‌های اولیه که برای مقابله با بهره‌برداری از پاورشل به ذهن می‌رسد، مسدود کردن کامل پاورشل در سیستم است؛ اما به دلیل این که پاورشل ابزار قدرتمند مورد استفاده بسیاری از ادمین‌ها است این امر ادمین سیستم را با محدودیت مواجه می‌کند. بنابراین به جای مسدودسازی کامل پاورشل، بهترین راهکار استفاده از ماژول‌های خود پاورشل و تنظیمات امن پاورشل است.

در این گزارش به معرفی روش‌هایی برای امن‌سازی پاورشل می‌پردازیم. در بخش اول اهمیت به‌روزرسانی پاورشل و نحوه‌ی غیرفعال کردن نسخه‌های قبلی آن بیان شده است. در بخش دوم، شیوه‌ی محدود کردن اسکریپت‌های اجرایی در پاورشل بیان شده است. با توجه به این که پاورشل پلتفرمی برای اجرای اسکریپت‌های مختلف است، کاربران برای اجرای اسکریپت‌ها و دسترسی به منابع سیستمی آزاد هستند. بنابراین محدود کردن اجرای اسکریپت‌ها در پاورشل توسط کاربران غیرقابل اعتماد، از اهمیت بسیار بالایی برای حفظ امنیت سیستم‌عامل ویندوز برخوردار است. در نهایت در بخش سوم نحوه‌ی جمع‌آوری لاگ‌های مختلف پاورشل و نظارت و بررسی گزارش‌ها و لاگ‌های آن بیان شده است. با استفاده از این روش می‌توان استفاده‌های مخرب از پاورشل را شناسایی کرد. همچنین با توجه به حساسیت اطلاعات ذخیره شده در لاگ‌ها، روش رمزنگاری و ذخیره امن لاگ‌ها در این بخش شرح داده شده است. در بخش چهارم استفاده امن از قابلیت Powershell Remoting بیان شده است.

مدیران و کارشناسان IT و کاربران سیستم‌عامل ویندوز می‌توانند با استفاده از این راهنما به امن‌سازی سیستم خود بپردازند. هنگامی که پاورشل به درستی تنظیم و پیکربندی شود می‌تواند برای رسیدگی به رخدادهای سیستم، خودکارسازی وظایف، بررسی و رصد فعالیت‌های سایبری و موارد مشابه مفید باشد.

## روش‌های امن‌سازی پاورشل

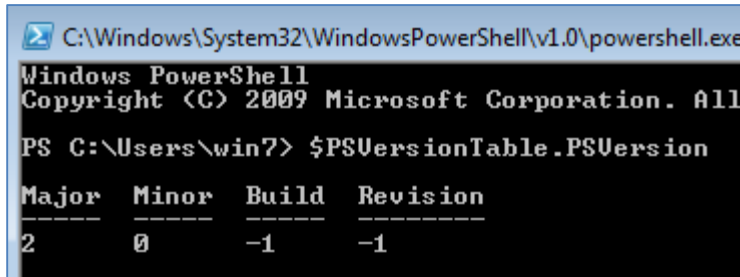
در این بخش نحوه‌ی امن‌سازی پاورشل در سیستم‌عامل ویندوز بررسی می‌شود.

## ۱-۱ بهروزرسانی پاورشل

بهتر است به جای مسدودسازی و حذف کامل پاورشل برای امن سازی از ویژگی های امنیتی آن استفاده شود. این ویژگی ها در نسخه ۵ به بالا ارتقا یافته اند. به عنوان نمونه AMSI که قابلیت اسکن محتوای حافظه برای تشخیص اسکریپت های بدخواه را دارد در نسخه ۲ فعال نیست. بنابراین ضروری است که پاورشل به آخرین نسخه موجود بهروزرسانی شود. لازم به ذکر است نسخه ۵ به طور پیش فرض در سیستم عامل های ویندوز ۱۰ و ۱۱ نصب شده است. به منظور بررسی نسخه پاورشل نصب شده بر روی سیستم از دستور زیر استفاده می شود:

```
$PSVersionTable.PSVersion
```

نمونه ای از خروجی اجرای این دستور در شکل ۱ نمایش داده شده است. در این سیستم، پاورشل نسخه ۲ نصب شده است.



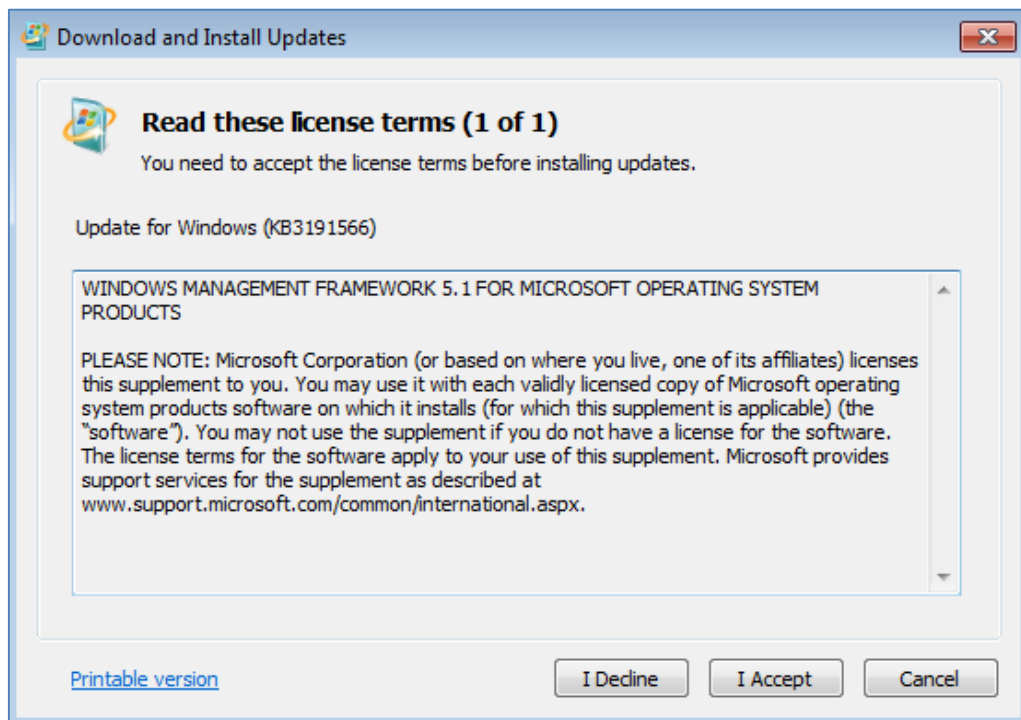
```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\win7> $PSVersionTable.PSVersion

Major Minor Build Revision
-----
2      0      -1      -1
```

شکل ۱. اطلاع از نسخه پاورشل نصب شده در سیستم

در صورت قدیمی بودن نسخه پاورشل، می بایست آن را بهروزرسانی کرد. برای ارتقای نسخه پاورشل در ابتدا لازم است که *NET Framework 4.5.2* یا بالاتر بر روی سیستم نصب شده باشد. سپس می بایست فریمورک *WMF 5.1* را از طریق لینک <https://www.microsoft.com/en-us/download/details.aspx?id=54616> دانلود و نصب کرد. قسمتی از مراحل نصب این فریمورک در شکل ۲ نمایش داده شده است.



شکل ۲. نصب WMF 5.1

پس از انجام عملیات نصب، پاورشل با موفقیت به نسخه ۵ ارتقا می‌یابد. نتیجه اجرای دستور تعیین نسخه پاورشل در شکل ۳ نمایش داده شده است.

```
Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\win7> $PSVersionTable.PSVersion

Major Minor Build Revision
-----
5      1      14409  1005
```

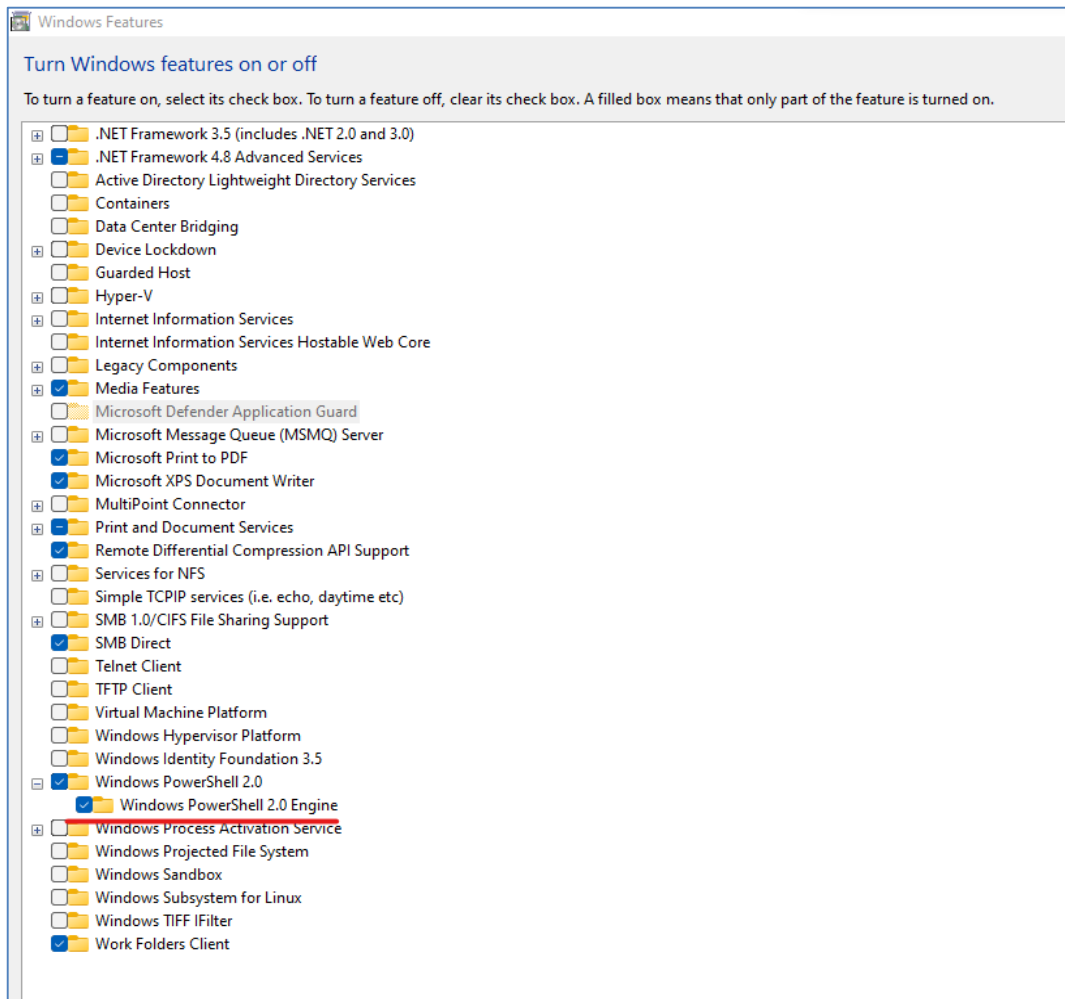
شکل ۳. پاورشل به روزرسانی شده

علاوه بر ارتقای پاورشل، لازم است که پاورشل نسخه‌ی ۲ به صورت کامل از سیستم حذف شود. بدین منظور می‌توان از دستور زیر در پاورشل استفاده کرد:

```
Disable-WindowsOptionalFeature -Online -FeatureName MicrosoftWindowsPowerShellV2Root
```

همچنین می‌توان برای غیرفعال کردن پاورشل نسخه‌ی ۲ از مسیر زیر در سیستم‌عامل‌های ویندوز ۸ و ویندوز ۱۰ استفاده کرد (شکل ۴):

```
control panel > Programs and Features > Turn Windows-Features on or off
```



شکل ۴. غیرفعال کردن PowerShell نسخه ۲

## ۱-۲ محدودسازی اجرای اسکریپت‌ها

### ۱-۲-۱ تنظیم سیاست اجرایی اسکریپت‌ها

سیاست اجرایی<sup>۱</sup> یکی از ویژگی‌های پاورشل است که شرایط اجرای اسکریپت‌ها در پاورشل را مشخص می‌کند. در سیستم عامل ویندوز، می‌توان برای کاربر جاری، رایانه محلی یا یک نشست خاص، سیاست اجرایی تنظیم کرد. سیاست‌های اجرایی پاورشل عبارتند از:

<sup>۱</sup> execution policy

- **AllSigned**: در این سیاست فقط اسکریپت‌های امضا شده اجازه اجرا دارند. این اسکریپت‌ها شامل اسکریپت‌های محلی ویندوز هستند. در این سیاست تمامی اسکریپت‌های داخلی و دانلود شده از اینترنت برای اجرا نیازمند امضا شدن هستند و پیش از اجرای اسکریپت از کاربر خواسته می‌شود تأیید کند که آیا به منتشر کننده اسکریپت اعتماد دارد یا خیر.
- **Bypass**: در این سیاست، اجرای همه انواع اسکریپت‌ها بدون هیچ هشدار و تأییدی از سمت کاربران امکان‌پذیر است. این سیاست عمدتاً برای پیکربندی‌هایی تنظیم شده که یک اسکریپت پاورشل در یک برنامه بزرگتر جاسازی شده است.
- **RemoteSigned**: این سیاست به طور پیش فرض در سیستم‌عامل‌های ویندوز سرور تنظیم شده است. در این سیاست، اسکریپت‌های نوشته شده در رایانه محلی برای اجرا شدن نیازی به امضاء ندارند اما برای اجرای اسکریپت‌های دانلود شده از اینترنت می‌بایست اسکریپت از منتشر کننده‌های قابل اعتماد امضای دیجیتال شده باشد.
- **Restricted**: این سیاست به طور پیش فرض در سیستم‌عامل‌های ویندوزی کلاینتی تنظیم شده است و اجازه اجرای اسکریپت‌ها را نمی‌دهد اما فرمان‌ها به صورت تکی قابلیت اجرا شدن دارند.
- **Undefined**: در محدوده فعلی هیچ سیاست اجرایی تنظیم نشده است.
- **Unrestricted**: در این سیاست همه اسکریپت‌ها بدون هیچ محدودیتی اجازه‌ی اجرا شدن دارند.

سیاست اجرایی مورد ترجیح برای مراقبت از اسکریپت‌ها، استفاده از سیاست اجرایی **AllSigned** است.

محدوده‌های زیر برای تنظیم سیاست‌های اجرایی در پاورشل وجود دارند:

- **MachinePolicy**: در این محدوده، سیاست اجرایی توسط group policy، برای همه کاربران رایانه تنظیم می‌شود.
- **UserPolicy**: در این محدوده، سیاست اجرایی توسط group policy، فقط برای کاربر جاری رایانه تنظیم می‌شود.

- **Process**: در این محدوده، سیاست اجرایی فقط برای نشست فعلی پاورشل تنظیم می‌شود.
- **CurrentUser**: در این محدوده، سیاست اجرایی فقط برای کاربر جاری رایانه تنظیم می‌شود.
- **LocalMachine**: در این محدوده، سیاست اجرایی برای همه کاربران رایانه تنظیم می‌شود.

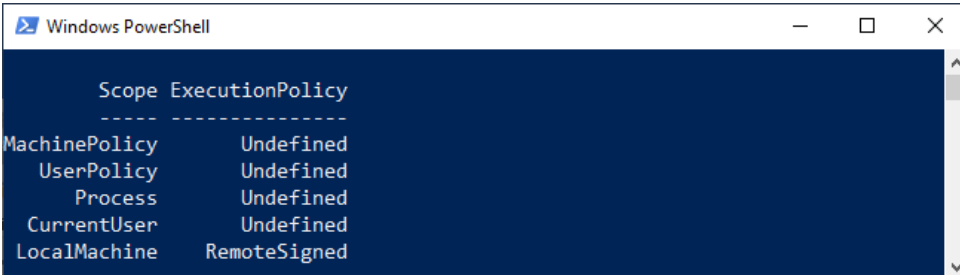
برای مشاهده سیاست اجرایی نشست فعلی پاورشل می‌توان از دستور زیر استفاده کرد:

```
Get-ExecutionPolicy
```

برای مشاهده تمامی سیاست‌های اجرایی تنظیم شده نیز می‌توان از دستور زیر استفاده کرد:

```
Get-ExecutionPolicy -list
```

خروجی اجرای این دستور در شکل ۵ نمایش داده شده است.



Scope	ExecutionPolicy
MachinePolicy	Undefined
UserPolicy	Undefined
Process	Undefined
CurrentUser	Undefined
LocalMachine	RemoteSigned

شکل ۵. مشاهده تمامی سیاست‌های اجرایی

برای تغییر در سیاست اجرایی از دستور زیر استفاده می‌شود:

```
Set-ExecutionPolicy "policy-value"
```

برای نمونه:

```
Set-ExecutionPolicy AllSigned
```

برای تعیین محدوده سیاست اجرایی می‌توان در این دستور از پارامتر **Scope** استفاده کرد. برای نمونه:

```
Set-ExecutionPolicy AllSigned -Scope CurrentUser
```



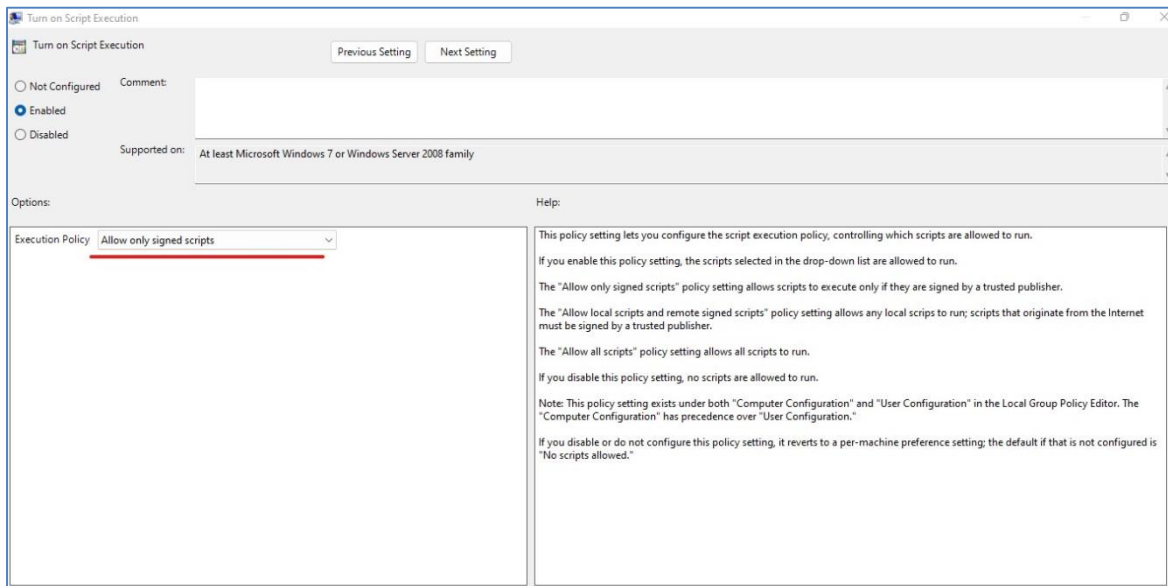
تفاوت اجرای یک اسکریپت که با استفاده از دو سیاست اجرایی *Restricted* و *AllSigned* در رایانه به صورت محلی نوشته شده در شکل ۶ نمایش داده شده است. در هر دو حالت اسکریپت اجرا نشده است؛ در حالت *Restricted* دلیل اجرا نشدن، غیرفعال بودن کلی اجرای اسکریپت در سیستم و در حالت *AllSigned* امضا نشدن اسکریپت دلیل عدم اجرا بوده است.

```
PS C:\Users\win7\Desktop> .\myscript.ps1
.\myscript.ps1 : File C:\Users\win7\Desktop\myscript.ps1 cannot be loaded
because running scripts is disabled on this system. For more information, see
about_Execution_Policies at http://go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\myscript.ps1
+ ~~~~~
+ CategoryInfo          : SecurityError: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\win7\Desktop> Set-ExecutionPolicy -ExecutionPolicy AllSigned -Force
PS C:\Users\win7\Desktop> .\myscript.ps1
.\myscript.ps1 : File C:\Users\win7\Desktop\myscript.ps1 cannot be loaded. The
file C:\Users\win7\Desktop\myscript.ps1 is not digitally signed. You cannot
run this script on the current system. For more information about running
scripts and setting execution policy, see about_Execution_Policies at
http://go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\myscript.ps1
+ ~~~~~
+ CategoryInfo          : SecurityError: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

شکل ۶. تفاوت دو سیاست اجرایی *restricted* و *allsigned*

علاوه بر قابلیت تنظیم سیاست‌های اجرایی از طریق خط فرمان، قابلیت تنظیم سیاست اجرایی به شیوه گرافیکی نیز فراهم است. بدین منظور بایستی مراحل زیر را طی نمود:

۱. در خط فرمان یا پاورشل، دستور «*gpedit.msc*» وارد شود تا **Group Policy Editor** باز شود.
۲. مراجعه به مسیر **Computer Configuration > Administrative Templates > Windows Components > Windows PowerShell**
۳. در پنجره سمت راست روی گزینه **Turn on Scrip Execution** دوبار کلیک کرده و سپس در پنجره‌ی نمایش داده شده گزینه **Enabled** انتخاب شود.
۴. در پارامتر **Execution Policy** سیاست **Allow only Signed Scripts** را انتخاب و تایید کرد (شکل ۷).



شکل ۷. فعال سازی Turn on Script Execution

## ۱-۲-۲ امضای اسکریپت‌های پاورشل

در صورتی که به دلایل امنیتی نیازی به اجرای کدهای اسکریپتی در پاورشل از هر منبعی نباشد، می‌توان اجرای اسکریپت‌ها را به منابع شناخته‌شده و قابل اعتماد محدود کرد. بدین منظور با تنظیم سیاست مناسب می‌توان تعیین کرد که تنها اسکریپت‌های امضا شده با یک گواهی<sup>۳</sup> خاص توانایی اجرا در پاورشل را داشته باشند. برای امضای اسکریپت در ابتدا نیاز به یک گواهی است. یکی از روش‌های تولید گواهی در ادامه معرفی شده است.

### دستور العمل تولید گواهی برای امضای اسکریپت:

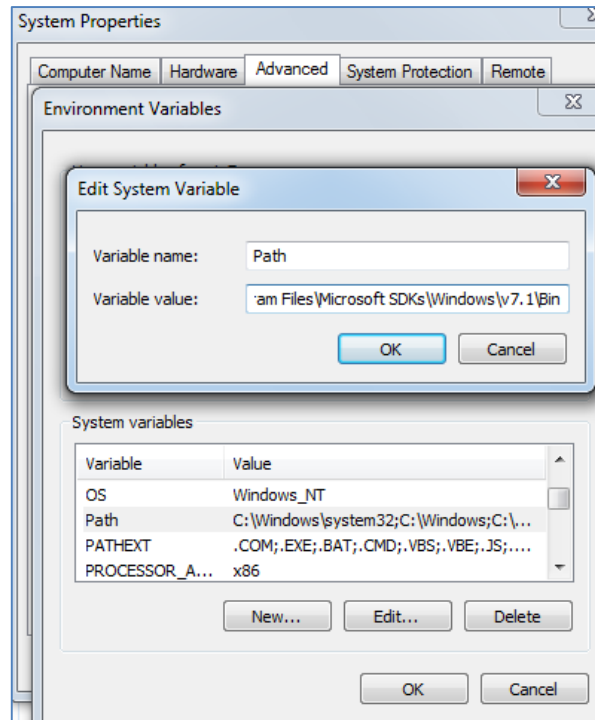
۱. در ابتدا می‌بایست، پلتفرم `windows SDK` از طریق یکی از دو لینک زیر دانلود و در سیستم نصب شوند:

- <https://developer.microsoft.com/en-us/windows/downloads/sdk-archive/>
- <https://developer.microsoft.com/en-us/windows/downloads/windows-sdk/>

۲. سپس مسیر `SDK` به متغیرهای محیطی سیستم اضافه شود (شکل ۸). `SDK` در ویندوز ۷ و ویندوز ۱۰ به ترتیب در مسیرهای زیر قرار گرفته است:

- `C:\Program Files\Microsoft SDKs\Windows\v۷.۱\`

- C:\Program Files(x86)\Microsoft SDKs\Windows\v10.0A\bin\



شکل ۸. افزودن SDK به متغیرهای محیطی سیستم

۳. سپس می‌بایست پاورشل در به عنوان ادمین اجرا شود.

۴. مرحله بعدی اجرای دستور زیر برای ایجاد یک CA محلی است:

```
makecert -n "CN=PowerShell Local Certificate Root" -a sha1 -eku 1.3.6.1.5.5.7.3.3 -r -sv root.pvk root.cer -ss Root -sr localMachine
```

۵. پس از اجرای دستور فوق، می‌بایست برای کلید خصوصی رمز عبور مناسبی تنظیم شود.

۶. حال بایستی دستور زیر را برای ایجاد گواهی با استفاده از CA ایجاد شده در مرحله قبلی اجرا نمود:

```
makecert -pe -n "CN=PowerShell User" -ss MY -a sha1 -eku 1.3.6.1.5.5.7.3.3 -iv root.pvk -ic root.cer
```

۷. در این مرحله مجدداً رمز عبور کلید خصوصی درخواست می‌شود.

به این ترتیب گواهی مورد نظر برای امضای اسکریپت تولید می‌شود. در نهایت می‌بایست اسکریپت مورد نظر خود را با استفاده از گواهی تولید شده امضاء کرد. به این منظور لازم است دستورهای زیر اجرا شوند:

```
$cert = Get-ChildItem -Path Cert:\CurrentUser\My -CodeSigningCert
Set-AuthenticodeSignature -FilePath '.\myscript.ps1' -Certificate $cert
```

پس از اجرای این دستور مشاهده می‌شود که اسکریپت، امضاء شده است. در شکل ۹ نمونه اسکریپت امضاء شده نمایش داده شده است.

```
myscript - Notepad
File Edit Format View Help
write-Host 'Script is running!' -BackgroundColor Black -ForegroundColor Yellow
# SIG # Begin signature block
# MIIEMwYJKoZIhvcNAQcCoIIEDCCBCACAQEXCZAJBgURDgMCGGUAAMGkGCisGAQQB
# gjCCAQsgwZBMDQGCisGAQQBgjCCAR4wJgIDAQAABBAfzDtgwUsITrck0sYpfvNR
# AgEAAGAAgEAAGAAgEAAGAAgEAAGAAgEAAGAAgEAAGAAgEAAGAAgEAAGAAgEAAG
# 5gKgggI9MIICOTCCAaagAwIBAgIQ7q3xcrRN6KJIMCP5CQJr3DAJBgURDgMCHQUA
# MCwxKjAoBgNVBAMTIVBvd2VyU2h1bGwGTG9jYywwGQ2Vydg1mawNhdGUGum9vdDae
# Fw0yMjEwMjEwOTI0NTJaFw0zOTEyMzEzMzU5NTlAMBoxGDAwBgNVBAMTD1Bvd2Vy
# U2h1bGwGTG9jYywwGQ2Vydg1mawNhdGUGum9vdDaeFw0zOTEyMzEzMzU5NTlAMBox
# SnrFnEK45XQ/0pe/emidPAOH79+F2x1Li+tfwv/HE3618me3xB5bv31N/xnhEFeg
# cpScQyLFpUKiotyAlvdwziOY2r8NrhuBcbCYP8ATRbz1xD1v5w1VknWpSfAURRk
# Efd/dpsk+hyw4EgIHSrblNk4chrocsAwEAAa2MHQwEwYDVR01BAwwCgYIKwYB
# BQUHAWMwXQYDVR0BBFYwVIAQSuTj2F6Tgtu0URXDowI456EuMCwxKjAoBgNVBAMT
# IVBvd2VyU2h1bGwGTG9jYywwGQ2Vydg1mawNhdGUGum9vdIIQ7YzZtqi4ZKpDRdRz
# qG2Q7DAJBgURDgMCHQUAA4GBACK995+rXz3eMB86BfGd+0Fjd14NzJnc+wUxuyx0
# g2n+AmBYJP/ae+74uHL7rAm1G2bMpxM5kxB4UUNDUNZOekvg9kqIX87I6fLsF+9b
# 5wQdtmNwdFuBdct60q+Jacjqxe3Wgm5v5NAGoPRxZEWP6urB1Zg7Poqz/m9s2r7y
# vtXwMYIBYDCCAvaWCAQEWQDAsMSowKAYDVQQDEYFqb3d1c1NoZwxsIEExvY2FsIEN1
# cnRpZmljYXRlIFJvb3QCE06t8XK0TeiSDHD0gkCa9wwCQYFKw4DAhoFAKB4MBGg
# CisGAQQBgjCCAQwxcjAIOAKAAKECGAAwGQYJKoZIhvcNAQkDMQwGCisGAQQBgjCC
# AQQwHAYKkwyBBAGCNwIBCzEOMAwGCisGAQQBgjCCARUwIwYJKoZIhvcNAQkEMRYE
# FFxtD9ZLW8bJ0xb+Xqjn4LpVSGN8MA0GCSqGSIb3DQEBAQUABIGASAU/na8P1kmo
# Dn9ubq6EMLpLp0++uWNIj4TI575U27HdepTtw2atV+63MbcNd9DrHrOUCQ4knkTm
# p8S190w9esyYUokLDSpp53bz51dCMHokZoAk2MwGs1XwFu2zPNKGLQdwZqDvyxvU
# oErsyqCucdiXIjzKAKdfrpiE1/1DaM4=
# SIG # End signature block
```

شکل ۹. اسکریپت امضاء شده

به این ترتیب می‌توان اسکریپت امضاء شده را در سیاست اجرایی *AllSigned* با موفقیت اجرا کرد (شکل ۱۰).

```

PS C:\Windows\system32> Set-ExecutionPolicy -ExecutionPolicy AllSigned -Force
PS C:\Windows\system32> cd C:\Users\win7\Desktop
PS C:\Users\win7\Desktop> .\myscript.ps1

Do you want to run software from this untrusted publisher?
File C:\Users\win7\Desktop\myscript.ps1 is published by CN=PowerShell User and
is not trusted on your system. Only run scripts from trusted publishers.
[U] Never run [D] Do not run [R] Run once [A] Always run [?] Help
(default is "D"):R
Script is running!

```

شکل ۱۰. اجرای اسکریپت امضاء شده در حالت AllSigned

دستورالعمل زیر روشی برای تولید گواهی در ویندوز ۱۰ است. پس از اجرای دستورالعمل، یک گواهی با نام TESTCERT در سه مکان Personal certificate store، Trusted Publishers store و Trusted Root Certification Authorities ایجاد و ذخیره می‌شود.

#### دستورالعمل:

۱. ابتدا پنجره‌ی PowerShell با سطح دسترسی کاربر ادمین باز شود.
۲. دستورهای زیر در پاورشل اجرا شوند:

```

$authenticode = New-SelfSignedCertificate -Subject "TESTCERT" -CertStoreLocation Cert:\LocalMachine\My -
Type CodeSigningCert
$rootStore = [System.Security.Cryptography.X509Certificates.X509Store]::new("Root", "LocalMachine")
$rootStore.Open("ReadWrite")
$rootStore.Add($authenticode)
$rootStore.Close()
$publisherStore =
[System.Security.Cryptography.X509Certificates.X509Store]::new("TrustedPublisher", "LocalMachine")
$publisherStore.Open("ReadWrite")
$publisherStore.Add($authenticode)
$publisherStore.Close()

```

پس از اجرای دستورالعمل فوق، گواهی ایجاد شده است. برای بررسی صحت ایجاد گواهی می‌توان از دستورهای زیر استفاده کرد:

```
Get-ChildItem Cert:\LocalMachine\My | Where-Object {$_.Subject -eq "CN=TESTCERT"}
Get-ChildItem Cert:\LocalMachine\Root | Where-Object {$_.Subject -eq "CN=TESTCERT"}
Get-ChildItem Cert:\LocalMachine\TrustedPublisher | Where-Object {$_.Subject -eq "CN=TESTCERT"}
```

خروجی اجرای دستورها در شکل ۱۱ نمایش داده شده است. در این شکل، اطلاعات گواهی شامل Subject و Thumbprint قابل مشاهده است:

```
PS Cert:\CurrentUser\root> Get-ChildItem Cert:\LocalMachine\My | Where-Object {$_.Subject -eq "CN=TESTCERT"}

PSParentPath: Microsoft.PowerShell.Security\Certificate::LocalMachine\My
Thumbprint           Subject
-----
4B1041B1DA0770B41CC68C32C01372EDD0BEE3BC CN=TESTCERT

PS Cert:\CurrentUser\root> Get-ChildItem Cert:\LocalMachine\Root | Where-Object {$_.Subject -eq "CN=TESTCERT"}

PSParentPath: Microsoft.PowerShell.Security\Certificate::LocalMachine\Root
Thumbprint           Subject
-----
4B1041B1DA0770B41CC68C32C01372EDD0BEE3BC CN=TESTCERT

PS Cert:\CurrentUser\root> Get-ChildItem Cert:\LocalMachine\TrustedPublisher | Where-Object {$_.Subject -eq "CN=TESTCERT"}

PSParentPath: Microsoft.PowerShell.Security\Certificate::LocalMachine\TrustedPublisher
Thumbprint           Subject
-----
4B1041B1DA0770B41CC68C32C01372EDD0BEE3BC CN=TESTCERT
```

شکل ۱۱. تولید موفق گواهی

حال می توان برای امضا کردن اسکریپت ها از گواهی ایجاد شده استفاده کرد. دستورهایی لازم عبارتند از:

```
$codeCertificate = Get-ChildItem Cert:\LocalMachine\My | Where-Object {$_.Subject -eq "CN=TESTCERT"}
Set-AuthenticodeSignature -FilePath E:\myscript.ps1 -Certificate $codeCertificate
```

پس از اجرای این دستور، اسکریپت با موفقیت امضا می شود. خروجی اجرای دستور در شکل ۱۲ نمایش داده شده است.

```
PS C:\Windows\system32> Set-AuthenticodeSignature -FilePath E:\myscript.ps1 -Certificate $codeCertificate

Directory: E:\

SignerCertificate           Status           Path
-----
4B1041B1DA0770B41CC68C32C01372EDD0BEE3BC Valid           myscript.ps1
```

شکل ۱۲. امضای موفق اسکریپت

## ۱-۲-۳ محدودسازی نوع دستوره‌های قابل اجرا در پاورشل

یکی از مکانیزم‌های امنیتی پاورشل، تنظیم `Language Mode` به حالت «`ConstrainedLanguage`» است. این حالت برای محدود کردن نوع دستورهایی استفاده می‌شود که توسط پاورشل اجرا می‌شوند. در حالت پیش‌فرض `Language Mode` مقدار «`Full language mode`» دارد که در این حالت تمامی توابع و ماژول‌ها در پاورشل در دسترس هستند.

## اجرای دستورالعمل از طریق خط فرمان:

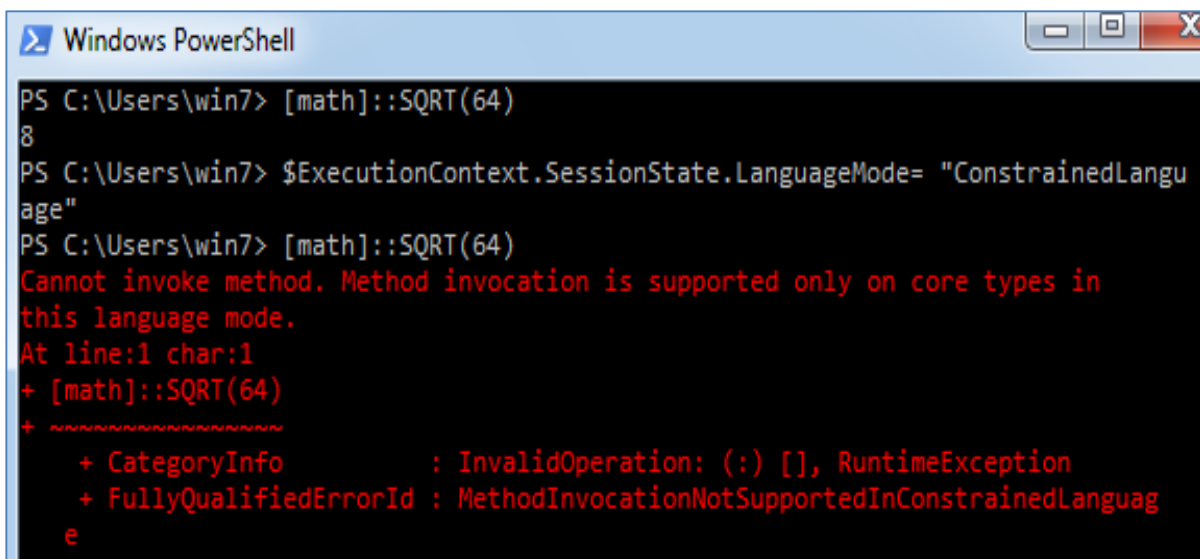
۱. ابتدا می‌توان با استفاده از دستور زیر تنظیم `Language Mode` سیستم را بررسی نمود:

```
$ExecutionContext.SessionState.LanguageMode
```

۲. سپس بایستی `Language Mode` به `ConstrainedLanguage` تغییر داده شود:

```
$ExecutionContext.SessionState.LanguageMode = " ConstrainedLanguage "
```

در شکل ۱۳ تفاوت اجرای یک دستور پیش و پس از فعال کردن حالت `constrainedLanguage` نمایش داده شده است.



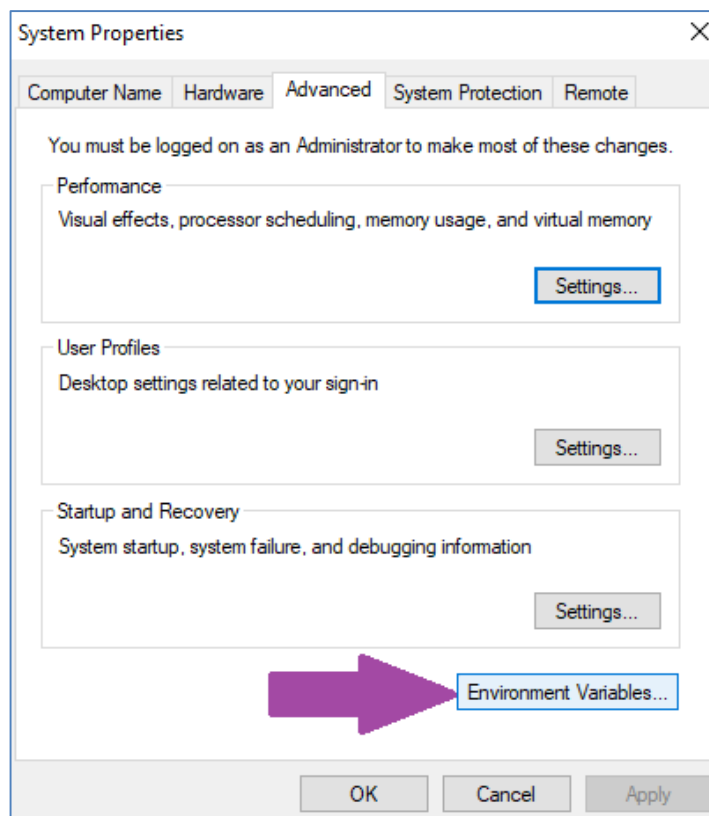
```
Windows PowerShell
PS C:\Users\win7> [math]::SQRT(64)
8
PS C:\Users\win7> $ExecutionContext.SessionState.LanguageMode= "ConstrainedLanguage"
PS C:\Users\win7> [math]::SQRT(64)
Cannot invoke method. Method invocation is supported only on core types in this language mode.
At line:1 char:1
+ [math]::SQRT(64)
+ ~~~~~
+ CategoryInfo          : InvalidOperation: (:) [], RuntimeException
+ FullyQualifiedErrorId : MethodInvocationNotSupportedInConstrainedLanguage
```

شکل ۱۳. محدود کردن نوع دستور قابل اجرا در پاورشل

محدودسازی از طریق `Environment Variable`:

۱. ابتدا بایستی از طریق جستجوی «environment» در جستجوی ویندوز، به پنجره `Edit System`

`Environment Variables` مراجعه نمود (شکل ۱۴).



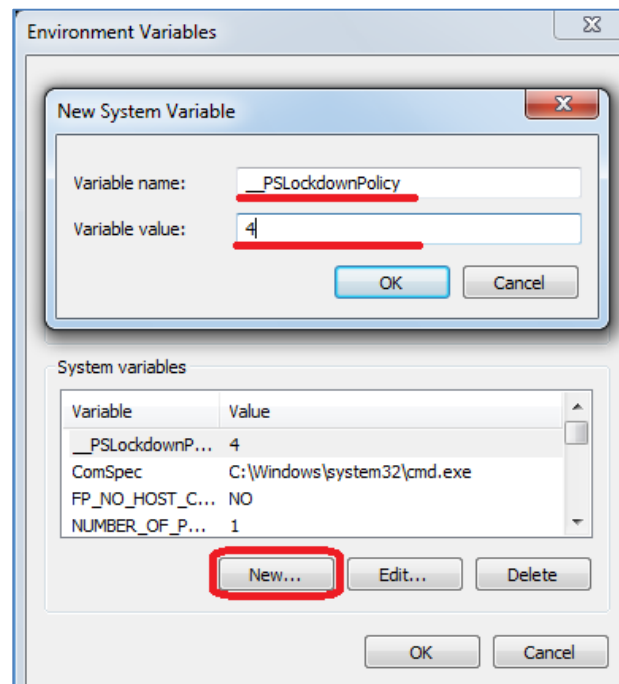
شکل ۱۴. پنجره متغیرهای محیطی سیستم

۲. سپس یک متغیر محیطی سیستمی جدید با نام `PSLockdownPolicy` و مقدار ۴ اضافه کرد (شکل ۱۵). با

این کار پاورشل به حالت `constrainedLanguage` تنظیم می‌شود. برای بازگشت به حالت `full language`

می‌بایست مقدار `PSLockdownPolicy` برابر ۸ تنظیم شود.





شکل ۱۵. محدودسازی پاورشل با استفاده از متغیرهای محیطی

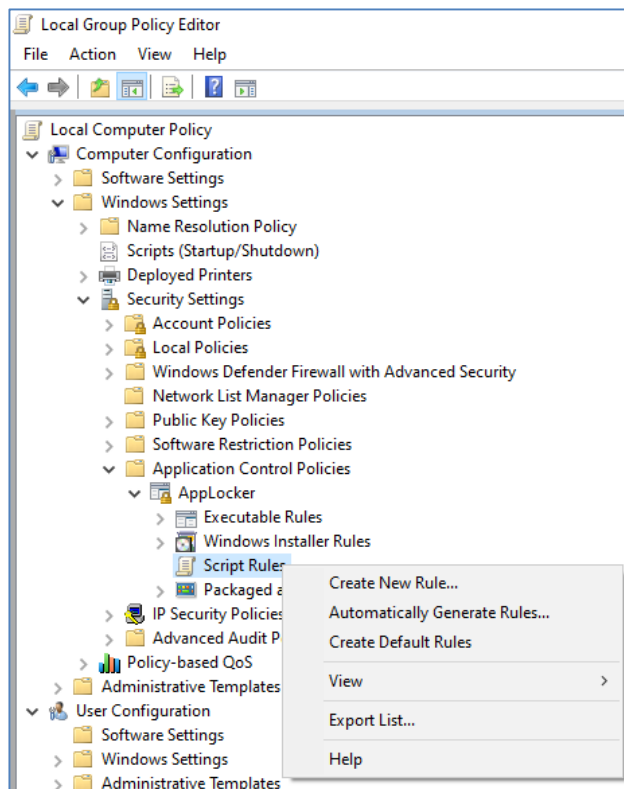
## ۱-۲-۴ محدودسازی اسکریپت‌ها با استفاده از AppLocker

AppLocker برنامه‌ای برای کنترل اجرای برنامه‌ها و فایل‌ها توسط کاربران است که در ویندوز ۱۰ تعبیه شده است. AppLocker قابلیت تنظیم اسکریپت‌های قابل اجرا در سیستم را دارد. برای مسدود کردن اسکریپت‌ها، می‌بایست قوانینی تعریف شوند تا اسکریپت‌ها تنها از مسیر ویندوز و یا دایرکتوری‌های برنامه‌ها، اجازه اجرا داشته باشند. در این مسیرها کاربران توانایی ذخیره و تغییر فایل‌ها را ندارند.

### دستورالعمل:

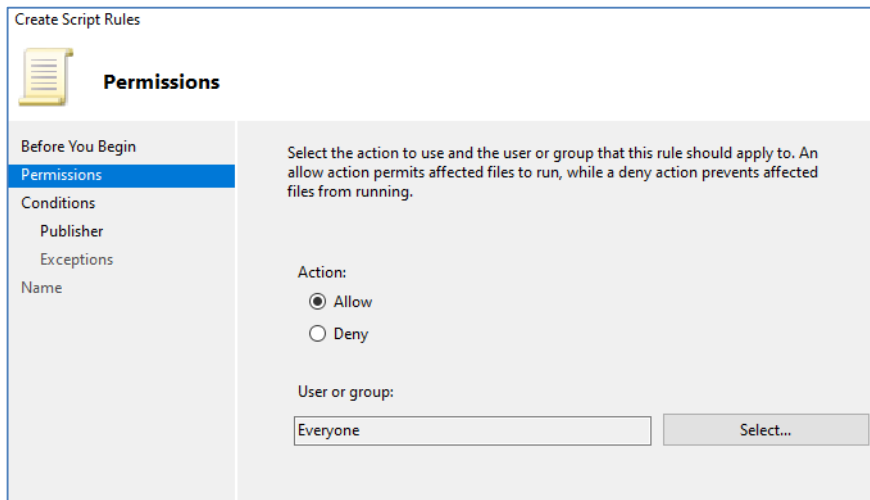
۱. در ابتدا می‌بایست خط فرمان پاورشل در سطح ادمین اجرا شود.
۲. سپس دستور `sc.exe config appidsvc start= auto` اجرا شود تا پس از هر بار بوت سیستم قوانین تعریف شده اعمال شوند.
۳. در خط فرمان یا پاورشل، دستور «gpedit.msc» وارد شود تا Group Policy Editor باز شود.
۴. به مسیر `Computer Configuration > Windows Setting > Security Setting > Application` به مسیر `Control Policy > AppLocker` مراجعه شود.

۵. در پنجره سمت راست روی گزینه *Script rules* کلیک راست شود. در این مرحله می توان بر حسب نیاز از قوانین پیش فرض استفاده کرد یا قوانین جدیدی ایجاد کرد. در قوانین پیش فرض، اجازه اجرای اسکریپت های موجود در مسیر فولدرهای *windows* و *Program Files* برای همه کاربران وجود دارد. همچنین کاربر ادمین می تواند تمامی اسکریپت ها را اجرا کند.



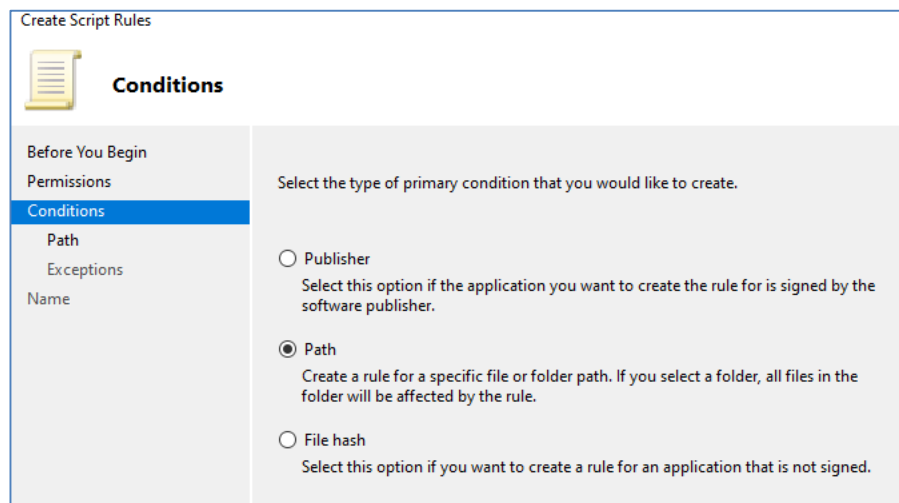
شکل ۱۶. تعریف قانون در AppLocker

علاوه بر استفاده از قوانین پیش فرض، می توان قوانین جدیدی ایجاد کرد. در مرحله اول تعریف قانون، می بایست مشخص شود که این قانون برای کدام کاربران اعمال شود. همچنین می بایست تعیین کرد که سیاست بر اجازه دادن است (*Allow*) یا ممنوع کردن (*Deny*). قوی ترین شکل امنیت، زمانی است که یک سیستم از *AppLocker* در حالت *Allow Mode* استفاده می کند، جایی که فقط برنامه های شناخته شده خاص مجاز به اجرا هستند.

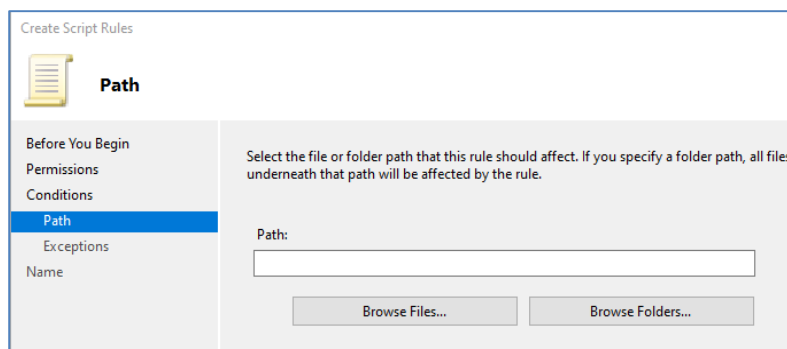


شکل ۱۷. مرحله اول در تعریف قانون جدید

در مرحله بعدی می‌بایست مسیر اسکریپت‌های قابل اجرا در سیستم را مشخص کرد.



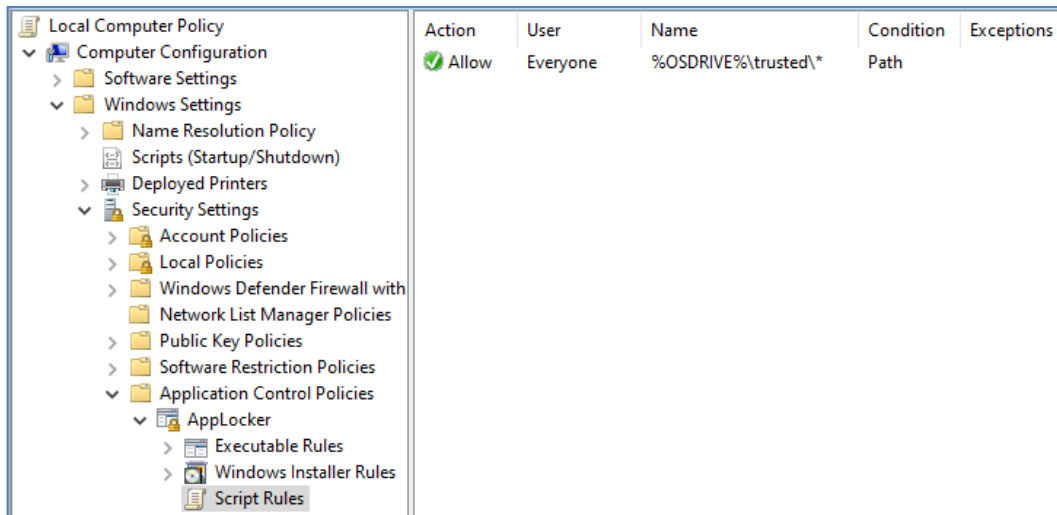
شکل ۱۸. نوع قانون ایجاد شده



شکل ۱۹. تعیین مسیر اجرای اسکریپت‌ها

به عنوان نمونه در شکل ۲۰ یک قانون برای مجوزدهی به اجرای اسکریپت‌ها فقط از مسیر "c:\trusted" تعریف شده

است.



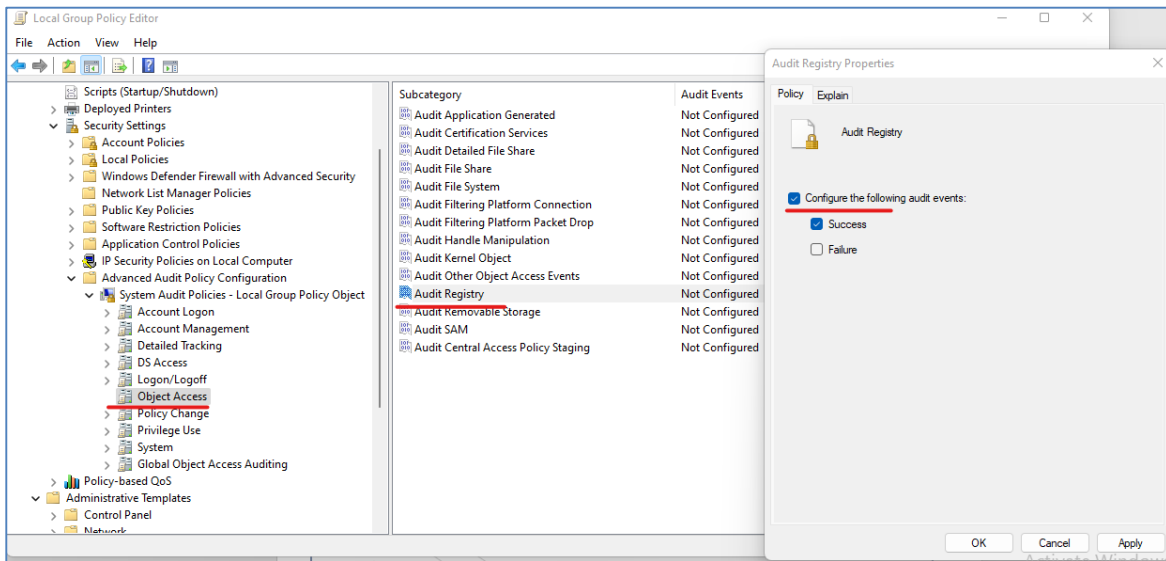
شکل ۲۰. نمونه یک قانون تعریف شده

## ۱-۲-۵ مراقبت از سیاست‌های اجرایی

رجیستری ویندوز حاوی تنظیمات حیاتی امنیتی متعددی است که مهاجم می‌تواند آن‌ها را دستکاری کند تا مکانیزم‌های حفاظتی مهم نادیده گرفته شوند. به عنوان مثال، یک مهاجم می‌تواند از آن برای دور زدن *group policy* سوء استفاده کند تا علی‌رغم تنظیم کردن ذخیره لاگ‌های پاورشل، پس از فعالیت وی اثری در لاگ‌ها دیده نشود. به این منظور ابتدا لازم است *Registry Auditing* فعال‌سازی شود و سپس برای کلید رجیستری مرتبط با سیاست‌های تنظیم شده برای پاورشل دسترسی‌های لازم تنظیم شود. به منظور فعال‌سازی *Registry Auditing* می‌توان طبق دستورالعمل زیر اقدام نمود:

### دستورالعمل:

- در خط فرمان یا پاورشل، دستور «gpedit.msc» وارد شود تا Group Policy Editor باز شود.
- به مسیر *Computer Configuration > Windows Setting > Security Setting > Advanced Audit Policy Configuration > System Audit Policies > Object Access* مراجعه نمود.
- در پنجره سمت راست روی گزینه *Audit Registry* دوبار کلیک کرده و سپس در پنجره نمایش داده شده گزینه *Configure the following audit events* و سپس *Success* انتخاب شود.
- در نهایت می‌بایست دستور **gpupdate** را اجرا کرد.

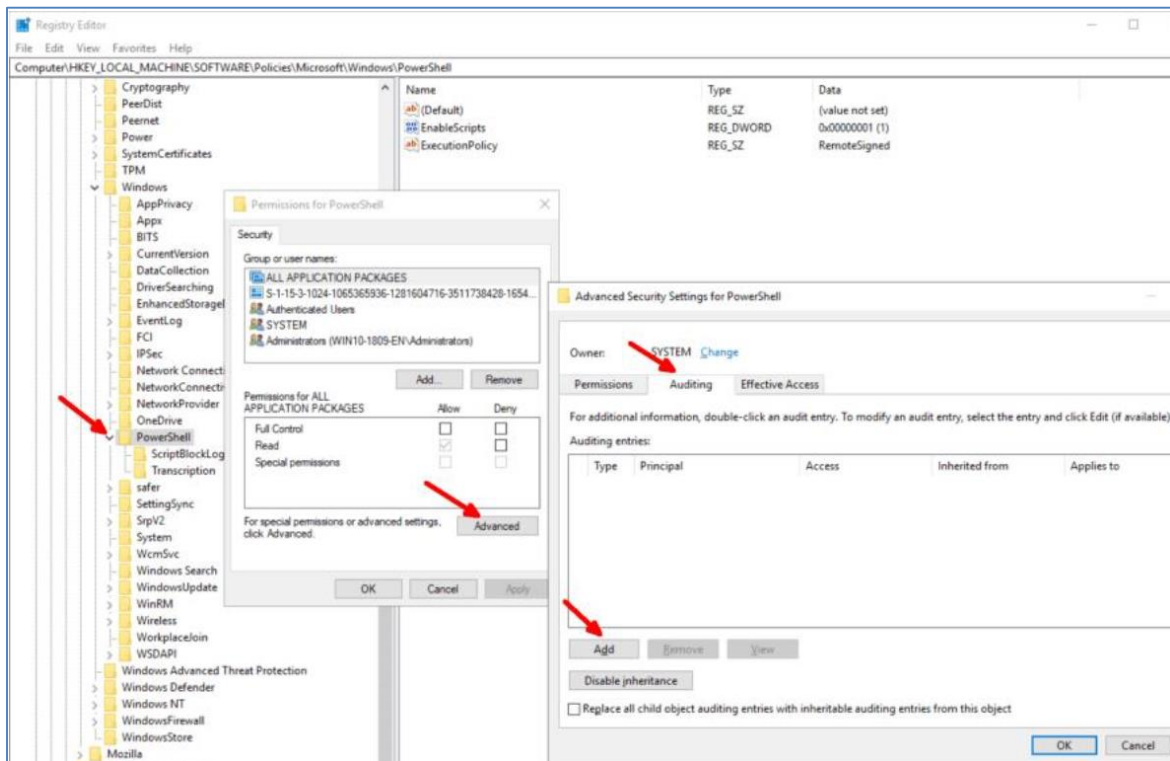


شکل ۲۱. فعال‌سازی Audit Registry

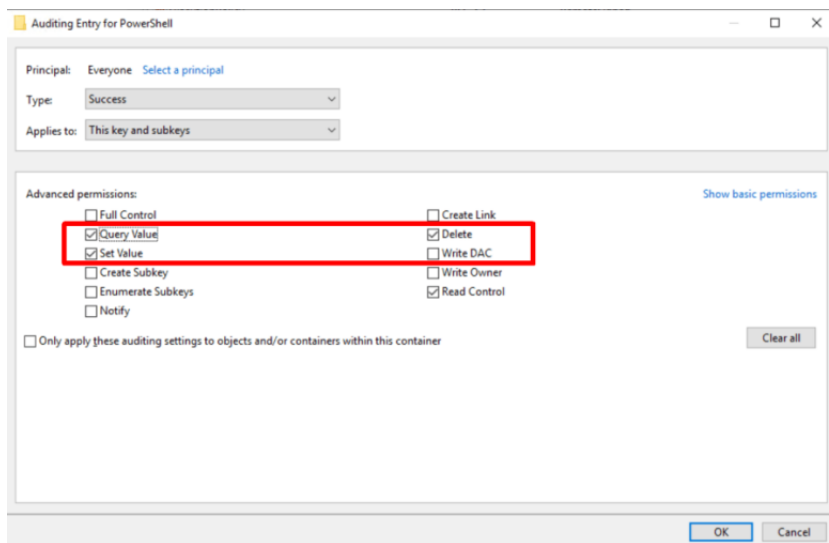
در ادامه لازم است که دسترسی‌ها به کلید رجیستری مرتبط با سیاست‌های پاورشل تنظیم شود. به این منظور می‌بایست طبق دستورالعمل زیر اقدام کرد:

**دستورالعمل:**

۱. در خط فرمان یا پاورشل عبارت "regedit" وارد شود.
۲. به مسیر `HKEY_LOCAL_MACHINE > SOFTWARE > Policies > Microsoft > Windows` به مسیری `PowerShell` مراجعه شود.
۳. روی `PowerShell` کلیک راست کرده و از طریق گزینه‌ی `Permissions` در پنجره‌ی باز شده، گزینه‌ی `Advanced` را انتخاب کرد.
۴. در پنجره‌ی باز شده، زیر قسمت `Auditing` گزینه‌ی `Add` را انتخاب کرد.
۵. سپس با انتخاب `Principle`، مانند `Everyone` گزینه‌های `Query Value`، `Set Value` و `Delete` را انتخاب کرد.



شکل ۲۲. تعیین دسترسی به Registry Key



شکل ۲۳. تعیین دسترسی به Registry Key

پس از اعمال تنظیمات فوق، می‌توان فعالیتهای مخرب صورت گرفته در راستای دور زدن سیاستهای اجرایی از طریق تغییر در کلیدهای رجیستری را در لاگها مشاهده کرد. این لاگها با شناسههای ۴۶۵۶، ۴۶۵۷، ۴۶۶۰ و ۴۶۶۳ قابل مشاهده هستند. تغییرات در موارد حیاتی دارای شناسههای ۴۶۵۷ و ۴۶۶۰ هستند. با استفاده از دستور زیر در پاورشل می‌توان این لاگها را مشاهده کرد. نمونه‌ای از لاگهای جمع‌آوری شده با این روش در شکل ۲۴ نمایش داده شده است.

```
Get-EventLog -LogName Security -Source "*auditing*" -InstanceId 4657,4660
```

```
PS C:\Windows\system32> Get-EventLog -LogName Security -Source "*auditing*" -InstanceId 4657,4660 | fl

Index           : 52024
EntryType       : SuccessAudit
InstanceId      : 4660
Message        : An object was deleted.

                Subject:
                Security ID:          S-1-5-18
                Account Name:        WIN10-1809-EN$
                Account Domain:     WINDOWSPRO
                Logon ID:            0x3e7

                Object:
                Object Server:      Security
                Handle ID:         0x171c

                Process Information:
                Process ID:         0x2d0
                Process Name:       C:\Windows\System32\svchost.exe
                Transaction ID:     {00000000-0000-0000-0000-000000000000}

Category       : (12801)
CategoryNumber : 12801
ReplacementStrings : {S-1-5-18, WIN10-1809-EN$, WINDOWSPRO, 0x3e7...}
Source         : Microsoft-Windows-Security-Auditing
TimeGenerated  : 12/15/2019 1:54:07 PM
TimeWritten    : 12/15/2019 1:54:07 PM
UserName       :

Index           : 52021
EntryType       : SuccessAudit
InstanceId      : 4657
Message        : A registry value was modified.
```

شکل ۲۴. مشاهده‌ی لاگ‌های تغییر در رجیستری

### ۳-۱ جمع‌آوری لاگ‌ها

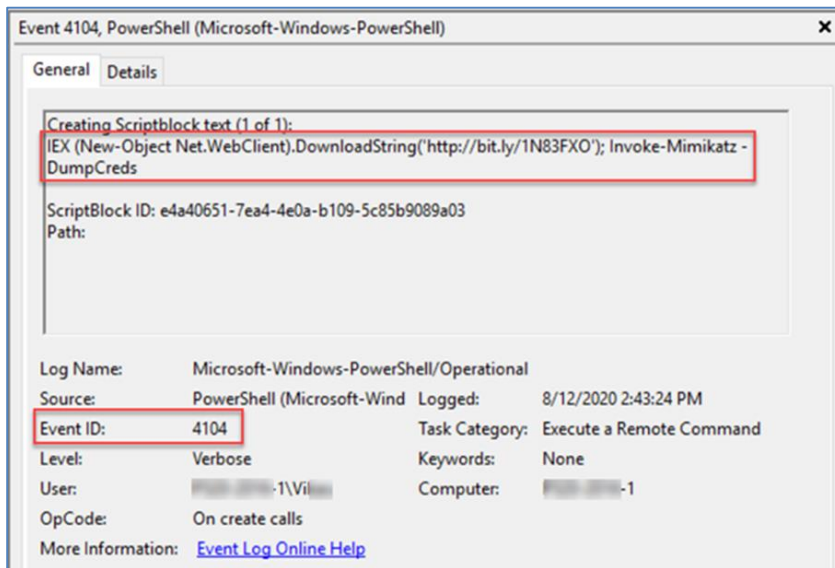
پاورشل ابزاری است که در بسیاری از حملات مهاجمان مورد سوء استفاده قرار می‌گیرد. به منظور تشخیص سوء استفاده‌ها از پاورشل در حملات مختلف، میکروسافت قابلیت ذخیره لاگ‌ها را برای نسخه‌های ۴ و ۵ از WMF فراهم کرده است که با استفاده از آن می‌توان تمام دستورات و اسکریپت‌های اجرا شده را جمع‌آوری کرد. این قابلیت به صورت پیش‌فرض در ویندوز ۱۰ قابل استفاده است اما کاربران ویندوز ۷ می‌بایست طبق دستورالعمل ذکر شده در بخش به‌روزرسانی پاورشل، نسخه پاورشل خود را ارتقاء دهند.

به سه شیوه مختلف می‌توان لاگ‌های پاورشل را ذخیره کرد:

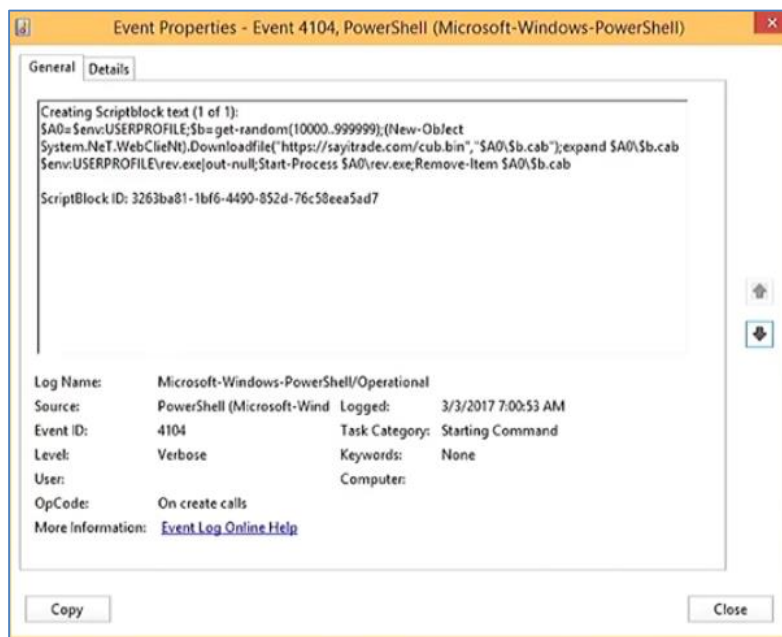
- **Module Logging:** در این روش پاورشل جزئیاتی راجع به مقداردهی اولیه متغیرها و فراخوانی فرمان‌ها در طول اجرا را لاگ‌برداری می‌کند. همچنین در این روش اطلاعاتی از اسکریپت‌های از حالت مبهم خارج شده و همچنین برخی از داده‌های خروجی ضبط می‌شوند. لاگ‌هایی که با این روش ذخیره می‌شوند با شناسه ۴۱۰۳ در `event viewer` ویندوز ذخیره و قابل مشاهده هستند.

- **Script Block Logging:** در این روش تمامی بلوک‌های کد پاورشلی در حال اجرا ذخیره می‌شوند. به عبارت دیگر محتوای کامل کد شامل کل اسکریپت و همه کامندها ضبط می‌شوند. همچنین در این روش همه کدهای `de-obfuscate` شده ضبط می‌شوند. برای نمونه در صورتی که کدی با استفاده از الگوریتم `Base64` کدگذاری شده باشد، اسکریپت واقعی کدگشایی شده در زمان اجرا ضبط می‌شود. برخلاف روش `Module logging` در این روش خروجی اسکریپت اجرا شده ضبط نمی‌شود. در صورتی که اندازه رخداد از بیشینه اندازه پیام در `event log` فراتر شود، در این روش رخداد به چندین رخداد جداسازی می‌شود. به علاوه در این روش رخدادهایی که با لیستی از کامندهای مشکوک مطابقت داشته باشند، در سطح `warning` لاگ‌برداری و لاگ‌های نرمال در سطح `information` یا `verbose` لاگ‌برداری می‌شوند. لاگ‌هایی که با این روش ذخیره می‌شوند با شناسه ۴۱۰۴ در `event viewer` ویندوز ذخیره و قابل مشاهده هستند. در صورت انتخاب گزینه `Start and Stop Events` برای جمع‌آوری لاگ‌ها، این لاگ‌ها با شناسه‌های ۴۱۰۵ و ۴۱۰۶ ذخیره می‌شوند. نمونه‌ای از لاگ‌های ذخیره شده به این روش در شکل‌های ۲۵ و ۲۶ نمایش داده شده است.





شکل ۲۵. مشاهده اطلاعات حمله در لاگ‌های Script Block Logging



شکل ۲۶. Script Block Logging

- **Full Transcription Logging:** در این روش نسخه کاملی از هر نشست مجزای پاورشل همراه با داده‌های ورودی و خروجی آن لاگ‌برداری شده و هر نسخه در فایل‌های مجزایی ذخیره می‌شود. لازم به ذکر است که در این روش فقط آنچه که در ترمینال ویندوزی پاورشل ظاهر می‌شود در فایل‌ها ذخیره می‌شوند. نمونه‌ای از اطلاعات ذخیره شده به این روش در شکل ۲۷ نمایش داده شده است.

```

PowerShell_transcript.DESKTOP-FK34BR6.L8Z+fzsj.20221023122919.txt
244 >> ParameterBinding(Out-String): name="InputObject"; v ^
245 Cannot invoke method. Method invocation is supported o
246 At C:\Program Files\WindowsPowerShell\Modules\PSReadli
247 + [Microsoft.PowerShell.PSConsoleReadLine]::ReadLi
248 + ~~~~~
249 + CategoryInfo          : InvalidOperation: (:) []
250 + FullyQualifiedErrorId : MethodInvocationNotSuppo
251 Cannot invoke method. Method invocation is supported o
252 At C:\Program Files\WindowsPowerShell\Modules\PSReadli
253 + [Microsoft.PowerShell.PSConsoleReadLine]::ReadLi
254 + ~~~~~
255 + CategoryInfo          : InvalidOperation: (:) []
256 + FullyQualifiedErrorId : MethodInvocationNotSuppo
257 *****
258 Command start time: 20221023123150
259 *****
260 PS C:\> get-date
261
262 2022/10/23 12:31:50 PM
263
264
265 *****
266 Windows PowerShell transcript start
267 Start time: 20221023123150

Windows PowerShell
PS C:\> get-date
2022/10/23 12:31:50 PM

PS C:\>

```

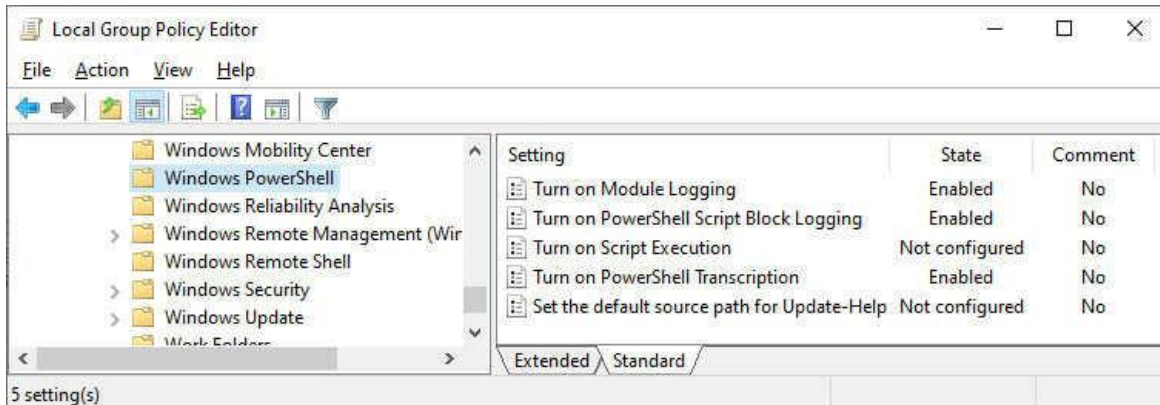
شکل ۲۷. Full Transcription Logging

لاگ‌های ذخیره شده به این روش‌ها از طریق **Event Viewer** ویندوز در مسیر **Application and Services Logs** ذخیره می‌شوند. برای فعال‌سازی مکانیزم جمع‌آوری لاگ‌ها نیاز است که طبق دستورالعمل زیر اقدام شود:

#### دستورالعمل:

۱. در خط فرمان یا پاورشل دستور «**gpedit.msc**» وارد شود تا **Group Policy Editor** باز شود.
۲. به مسیر **Computer Configuration > Administrative Templates > Windows Components > Windows PowerShell** مراجعه شود.
۳. بر روی روش‌های مورد نظر خود برای لاگ‌برداری در پنجره سمت راست دوبرار کلیک و گزینه‌ی **Enabled** انتخاب شود. در روش **PowerShell Transcription** می‌بایست از طریق پارامتر **Transcript Output Directory**، مسیر ذخیره‌ی لاگ‌ها را مشخص کرد. قابل توجه است که این فایل اطلاعات حساس اجرا شده در **PowerShell** را شامل می‌شود؛ بنابراین باید از دسترسی کاربران به آن جلوگیری کرد. بدین منظور نیاز است

که روی فایل مورد نظر کلیک راست کرده و گزینه‌ی **Properties** انتخاب شود. سپس در زیر مجموعه‌ی **Security > Advanced Security > Permissions** دسترسی به فایل را فقط به ادمین محدود کرد.



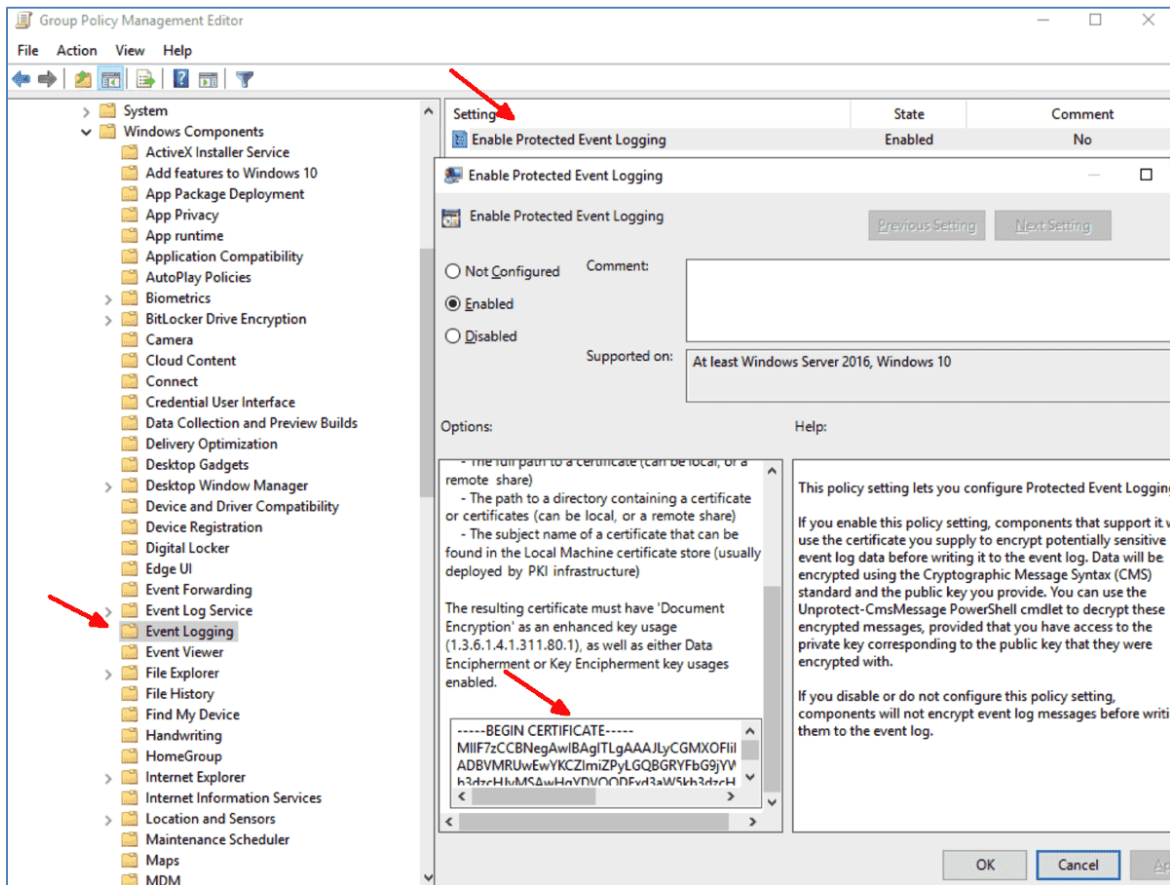
شکل ۲۸. فعال‌سازی جمع آوری لاگ در پاورشل

### ۱-۳-۱ رمزنگاری لاگ‌ها و فایل‌ها

لاگ‌های ذخیره شده توسط پاورشل ممکن است حاوی اطلاعات حساسی باشند. به این ترتیب برای محافظت از لاگ‌ها ویژگی **Protected Event Logging** در ویندوز ۱۰ و ویندوز سرور ۲۰۱۶ برای رمز کردن داده‌های حساس موجود در **Event log**ها معرفی شده است. برای فعال‌سازی رمزنگاری لاگ‌ها طبق دستورالعمل زیر عمل می‌شود:

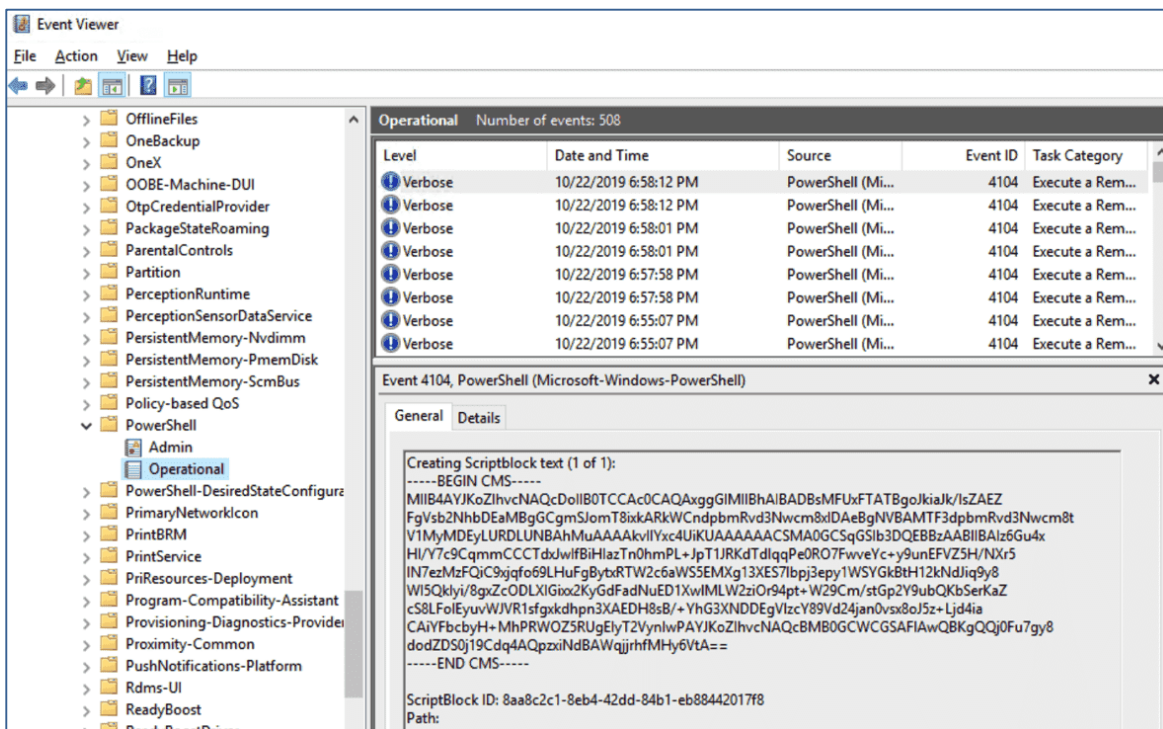
#### دستورالعمل:

۱. در خط فرمان یا پاورشل، دستور «**gpedit.msc**» وارد شود تا **Group Policy Editor** باز شود.
۲. به مسیر **Computer Configuration > Administrative Templates > Windows Components > Event Logging** مراجعه شود.
۳. در پنجره‌ی سمت راست روی گزینه‌ی **Enable Protect Event logging** دوبار کلیک کرده و سپس در پنجره‌ی نمایش داده شده گزینه‌ی **Enabled** انتخاب شود.
۴. در قسمت پایین پنجره در مربع نمایش داده شده، محتوای کدگذاری شده **Certificate** معتبر یا اطلاعاتی مانند **Subject** یا **Thumbprint** گواهی وارد شود. نمونه‌ای از این اطلاعات در شکل ۲۹ نمایش داده شده است.



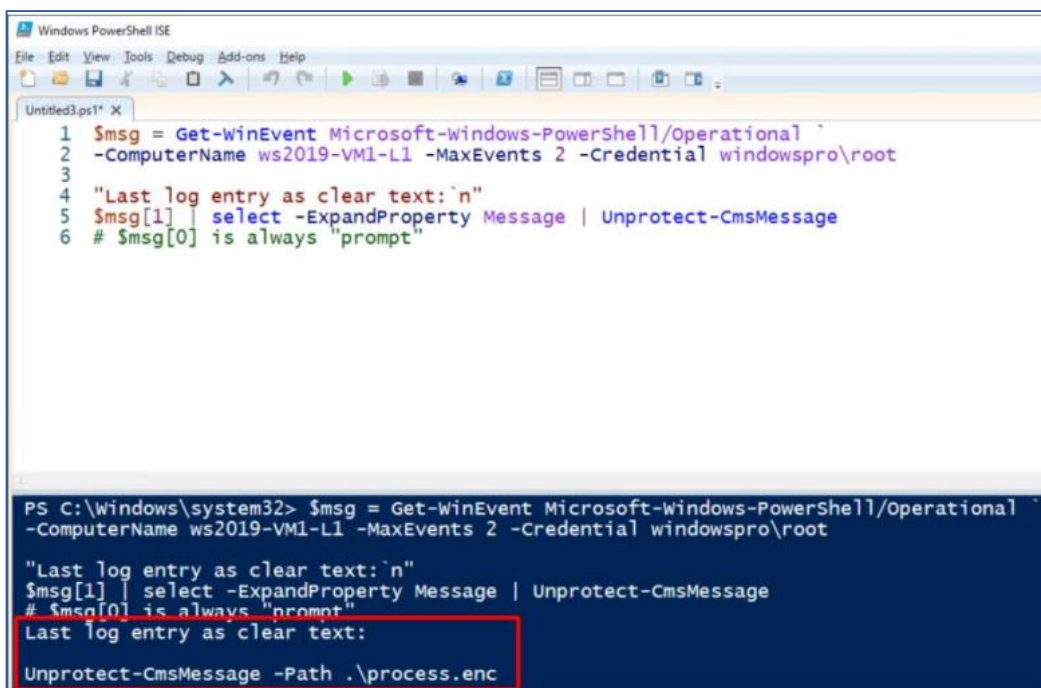
شکل ۲۹. رمزنگاری لاگ‌ها

با پیاده‌سازی تنظیمات فوق، تمامی لاگ‌های ایجاد شده به صورت رمز شده ذخیره می‌شوند. نمونه‌ای از لاگ‌های ذخیره شده به صورت رمز در شکل ۳۰ نمایش داده شده است.



شکل ۳۰. ذخیره لاگ‌ها به صورت رمزگذاری شده

برای رمزگشایی لاگ‌ها از دستور "Unprotect-CmsMessage" در پاورشل استفاده می‌شود. برای مثال در صورت نیاز به رمزگشایی آخرین لاگ PowerShell، می‌توان با استفاده از دستور زیر ابتدا با `Get-WinEvent` آن را فراخوانی کرده و سپس با دستور "Unprotect-CmsMessage" رمزگشایی کرد. شکل ۳۱ نحوه‌ی انجام این کار را نمایش می‌دهد.



## شکل ۳۱. نحوه‌ی رمزگشایی لاگ‌ها

## ۱-۳-۲ رمزنگاری فایل‌های حاوی رویداد پاورشل

برای رمزنگاری فایل‌ها، به‌خصوص فایل لاگ‌هایی که به روش **Transcription** ذخیره می‌شوند از دستور **"Protect-CmsMessage"** استفاده می‌شود. برای رمزنگاری فایل‌ها به یک **Certificate** معتبر نیاز است. به این منظور لازم است با استفاده از دستورهای زیر در پاورشل یک گواهی معتبر برای رمزنگاری تولید شود. پس از اجرای این دستورها یک گواهی با نام **TestCert.cer** ساخته می‌شود.

```
{[Version]
Signature = "$Windows NT$"

[Strings]
szOID_ENHANCED_KEY_USAGE = "2.5.29.37"
szOID_DOCUMENT_ENCRYPTION = "1.3.6.1.4.1.311.80.1"

[NewRequest]
Subject = "cn=CRYPT_CERT"
MachineKeySet = false
KeyLength = 2048
KeySpec = AT_KEYEXCHANGE
HashAlgorithm = Sha1
Exportable = true
RequestType = Cert
KeyUsage = "CERT_KEY_ENCRYPTERMENT_KEY_USAGE | CERT_DATA_ENCRYPTERMENT_KEY_USAGE"
ValidityPeriod = "Years"
ValidityPeriodUnits = "1000"

[Extensions]
%szOID_ENHANCED_KEY_USAGE% = "{text}%szOID_DOCUMENT_ENCRYPTION%"
} | Out-File -FilePath TestCert.inf

certreq.exe -new crypt.inf crypt.cer
```

حال با استفاده از گواهی تولید شده و اجرای دستور زیر می‌توان فایل‌های مورد نظر خود در سیستم را رمز کرد.

```
Protect-CmsMessage -To "CRYPT_CERT" -Path 'path_of_file' -OutFile encrypt2.cms
```

خروجی دستورهای فوق در شکل ۳۲ نمایش داده شده است.

```

PS E:\> <[Version]
>> Signature = "$Windows NT$"
>>
>> [Strings]
>> szOID_ENHANCED_KEY_USAGE = "2.5.29.37"
>> szOID_DOCUMENT_ENCRYPTION = "1.3.6.1.4.1.311.80.1"
>>
>> [NewRequest]
>> Subject = "cn=CRYPT_CERT"
>> MachineKeySet = false
>> KeyLength = 2048
>> KeySpec = AT_KEYEXCHANGE
>> HashAlgorithm = Sha1
>> Exportable = true
>> RequestType = Cert
>> KeyUsage = "CERT_KEY_ENCIPHERMENT_KEY_USAGE ; CERT_DATA_ENCIPHERMENT_KEY_USAGE"
>> ValidityPeriod = "Years"
>> ValidityPeriodUnits = "1000"
>>
>> [Extensions]
>> %szOID_ENHANCED_KEY_USAGE% = "{text}%szOID_DOCUMENT_ENCRYPTION%"
>> } ! Out-File -FilePath crypt.inf
PS E:\> certreq.exe -new crypt.inf crypt.cer
Installed Certificate:
Serial Number: 1317d7d5b648668e4de490a3fa9d156f
Subject: CN=CRYPT_CERT
NotBefore: 2022/10/26 11:54 AM
NotAfter: 3022/10/26 12:04 PM
Thumbprint: 69a2440982e6c937a9a3766f0013e898d368f2e5
Microsoft Strong Cryptographic Provider
00c4a0ea18b7373529a01c6e9c62b982_508a8900-d9a5-43ce-ab27-4bbd70630b32

CertReq: Certificate Created and Installed
PS E:\> Protect-CmsMessage -To "CN=CRYPT_CERT" -Path 'myscript.ps1' -OutFile encrypted.cms

```

شکل ۳۲. رمز کردن فایل لاگ

برای رمزگشایی فایل رمز شده از دستور زیر استفاده می‌شود:

```
Unprotect-CmsMessage -Path .\encrypted.cms
```

## ۱-۴ استفاده امن از قابلیت Powershell Remoting

Powershell Remoting قابلیت از پاورشل است که به کاربران امکان اتصال از طریق شبکه به صورت راه دور به پاورشل را فراهم می‌کند. اگرچه این قابلیت برای اهداف ادمین سیستم فراهم شده است اما ممکن است توسط مهاجمان در حملات مورد سوء استفاده قرار بگیرد.

در برخی از نسخه‌های ویندوز مانند ویندوز سرور ۲۰۱۲ این قابلیت به صورت پیش فرض فعال است. برای فعال کردن این قابلیت از دستور Enable-PSRemoting در پاورشل استفاده می‌شود. توصیه امنیتی بر این است که powershell Remoting فقط در صورت لزوم فعال شود و در غیر این صورت غیرفعال باشد.

### ۱-۴-۱ غیرفعال کردن Powershell Remoting

دستورالعمل:

۱. دستور `Disable-PSRemoting` در پاورشل اجرا شود.

۲. با اجرای دستورهای زیر در پاورشل، سرویس ویندوزی `winRM` که مسئول فراهم کردن دسترسی به پاورشل از راه دور از طریق شبکه است متوقف و غیرفعال شود.

```
Stop-Service WinRM -PassThru
Set-Service WinRM -StartupType Disabled -PassThru
```

در صورتی که سرویس `winRM` غیرفعال نشد، می‌بایست دستورهای زیر در پاورشل اجرا شوند:

```
dir wsman:\localhost\listener
Remove-Item -Path WSMan:\localhost\listener\<Listener>
```

۳. اجرای دستورهای زیر در پاورشل:

```
Set-NetFirewallRule -DisplayName 'Windows
Remote Management (HTTP-In)' -Enabled False -PassThru | Select -Property
DisplayName, Profile, Enabled
```

۴. اطمینان از صفر بودن مقدار کلید رجیستری زیر:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\
System\LocalAccountTokenFilterPolicy
```

## ۱-۴-۲ محدود کردن Powershell Remoting

در صورتی که `powershell Remoting` به دلایل ضروری در سیستم فعال است، به دلایل امنیتی می‌بایست محدود شود. `Just Enough Administration (JEA)` یک مکانیزم امنیتی پاورشل است که اصل حداقل امتیاز را پیاده‌سازی کرده است. این مکانیزم نشست‌های پاورشلی را با مجموعه محدودی از کارایی‌های پاورشل برای ادمین‌های سیستم ایجاد می‌کند. `JEA` از `Powershell Remoting` به نحوی استفاده می‌کند که کاربران می‌توانند به یک نشست پاورشل با قابلیت‌های محدود ارتباط برقرار کنند. این نشست `JAE endpoint` نامیده می‌شود. کاربران برای اتصال به این نشست از دستور پاورشلی `Enter-PSSession` استفاده می‌کنند.

پاورشل یک ابزار قوی در سیستم‌عامل ویندوز است که قابلیت‌های فراوانی در اختیار کاربران ویندوز قرار می‌دهد؛ از سوی دیگر در حملات بسیاری مورد سوءاستفاده مهاجمان قرار می‌گیرد. از این رو امن‌سازی پاورشل از درجه اهمیت بسیار بالایی برای جلوگیری از نفوذ مهاجمان برخوردار است. در این گزارش به بررسی مهم‌ترین روش‌های امن‌سازی پاورشل در سیستم عامل ویندوز پرداخته شده است.



## پایان بندی

در نهایت به کاربران توصیه می‌شود پس از اجرای تمامی موارد فوق، چک لیست زیر تکمیل شود.

### چک لیست امن‌سازی پاورشل

- پاورشل نسخه ۲ بر روی سیستم غیرفعال است.
- سیاست اجرایی مناسبی در پاورشل انتخاب و تنظیم شده است.
- با تنظیم *Language Mode*، نوع دستورهای قابل اجرا در پاورشل محدود شده است.
- از *AppLocker* برای تنظیم اسکریپت‌های قابل اجرا در سیستم استفاده شده است.
- روش مناسبی برای ذخیره لاگ‌های پاورشل انتخاب شده است.
- از روش‌های امنی برای رمز کردن لاگ‌های پاورشل استفاده شده است.
- استفاده از *powershell Remoting* غیرفعال یا محدود شده است.

## منابع خبر:

- <https://www.reliaquest.com/blog/powershell-security-best-practices/>
- <https://www.calcomsoftware.com/defend-against-powershell-attacks/>
- <https://www.calcomsoftware.com/basic-steps-for-powershell-attacks-prevention/>
- <https://www.scriptrunner.com/en/blog/powershell-security-best-practices>
-