about_Logging_Windows - PowerShell

By sdwheeler

Archived: 2026-04-05 21:49:28 UTC

PowerShell logs internal operations from the engine, providers, and cmdlets to the Windows event log.

PowerShell logs details about PowerShell operations, such as starting and stopping the engine and providers, and executing PowerShell commands.

For information about logging in Windows PowerShell 5.1, see about_Logging.

PowerShell supports configuring two categories of logging:

- Module logging - Record the pipeline execution events for members of specified modules. Module logging must be enabled for both the session and specific modules. For more information about configuring this logging, see about_PowerShell_Config.

  If module logging is enabled through configuration, you can enable and disable logging for specific modules in a session by setting the value of the **LogPipelineExecutionDetails** property of the module.

  For example, to enable module logging for the **PSReadLine** module:

  ```
  $psrl = Get-Module PSReadLine
  $psrl.LogPipelineExecutionDetails = $true
  Get-Module PSReadLine | Select-Object Name, LogPipelineExecutionDetails
  ```

  ```
  Name       LogPipelineExecutionDetails
  ----       ---------------------------
  PSReadLine                        True
  ```

- Script block logging - Record the processing of commands, scriptblocks, functions, and scripts whether invoked interactively, or through automation.

  When you enable Script Block Logging, PowerShell records the content of all scriptblocks that it processes. Once enabled, any new PowerShell session logs this information. For more information, see Enabling scriptblock Logging.

Unlike Linux or macOS, Windows requires the event provider to be registered before events can be written to the event log. To enable the PowerShell event provider, run the following command from an elevated PowerShell prompt.

```
$PSHOME\RegisterManifest.ps1
```

PowerShell logs can be viewed using the Windows Event Viewer. The event log is located in the **Application and Services Logs** group and is named **PowerShellCore**. The associated ETW provider GUID is `{f90714a8-5509-434a-bf6d-b1624c8a19a2}`.

When Script Block Logging is enabled, PowerShell logs the following events to the **PowerShellCore/Operational** log:

Field	Value
EventId	`4104` / `0x1008`
Channel	`Operational`
Level	`Verbose`
Opcode	`Create`
Task	`CommandStart`
Keyword	`Runspace`

Registering the event provider places a lock in the binary library used to decode events. To update this library, the provider must be unregistered to release this lock.

To unregister the PowerShell provider, run the following command from an elevated PowerShell prompt.

```
$PSHOME\RegisterManifest.ps1 -Unregister
```

After updating PowerShell, run `$PSHOME\RegisterManifest.ps1` to register the updated event provider.

When you enable Script Block Logging, PowerShell records the content of all scriptblocks that it processes. Once enabled, any new PowerShell session logs this information.

Note

It's recommended to enable Protected Event Logging, as described below, when using Script Block Logging for anything other than diagnostics purposes.

Script Block Logging can be enabled via Group Policy or a registry setting.

To enable automatic transcription, enable the **Turn on PowerShell Script Block Logging** feature in Group Policy through **Administrative Templates** -> **PowerShell Core**.

Run the following function:

```
function Enable-PSScriptBlockLogging {
    $basePath = @(
        'HKLM:\Software\Policies\Microsoft'
        'PowerShellCore\ScriptBlockLogging'
    ) -join '\'

    if (-not (Test-Path $basePath)) {
        $null = New-Item $basePath -Force
    }

    Set-ItemProperty $basePath -Name EnableScriptBlockLogging -Value "1"
}
```

You can set the `ScriptBlockLogging` option in the `powershell.config.json` file that controls how PowerShell behaves. For more information, see about_PowerShell_Config.

Increasing the level of logging on a system increases the possibility that logged content may contain sensitive data. For example, with script logging enabled, credentials or other sensitive data used by a script can be written to the event log. When a machine that has logged sensitive data is compromised, the logs can provide an attacker with information needed to extend their reach.

To protect this information, Windows 10 introduces Protected Event Logging. Protected Event Logging lets participating applications encrypt sensitive data written to the event log. Later, you can decrypt and process these logs on a more secure and centralized log collector.

Event log content is protected using the IETF Cryptographic Message Syntax (CMS) standard. CMS uses public key cryptography. The keys used to encrypt content and decrypt content are kept separate.

The public key can be shared widely and isn't sensitive data. Any content encrypted with this public key can only be decrypted by the private key. For more information about Public Key Cryptography, see Wikipedia - Public Key Cryptography.

To enable a Protected Event Logging policy, deploy a public key to all machines that have event log data to protect. The corresponding private key is used to post-process the event logs at a more secure location such as a central event log collector, or SIEM aggregator. You can set up SIEM in Azure. For more information, see Generic SIEM integration.

To enable Protected Event Logging, enable the `Enable Protected Event Logging` feature in Group Policy through `Administrative Templates -> Windows Components -> Event Logging`. This setting requires an encryption certificate, which you can provide in one of several forms:

- The content of a base-64 encoded X.509 certificate (for example, as offered by the `Export` option in Certificate Manager).
- The thumbprint of a certificate that can be found in the Local Machine certificate store (can be deployed by PKI infrastructure).

- The full path to a certificate (can be local, or a remote share).
- The path to a directory containing a certificate or certificates (can be local, or a remote share).
- The subject name of a certificate that can be found in the Local Machine certificate store (can be deployed by PKI infrastructure).

The resulting certificate must have `Document Encryption` as an enhanced key usage (`1.3.6.1.4.1.311.80.1`), and either `Data Encipherment` or `Key Encipherment` key usages enabled.

Warning

The private key shouldn't be deployed to the machines logging events. It should be kept in a secure location where you decrypt the messages.

The following script retrieves and decrypts events, assuming that you have the private key:

```
Get-WinEvent Microsoft-Windows-PowerShell/Operational |
   Where-Object Id -EQ 4104 | Unprotect-CmsMessage
```

- about_Logging_Non-Windows
- PowerShell the Blue Team
- Generic SIEM integration
