Sign in

PhrozenIO / PowerRunAsSystem Public

Notifications | Fork 29 | Star 246

Code | Issues | Pull requests | Actions | Projects | Security | Insights

main | Go to file | Code

- PowerRunAsSystem
- images
- LICENSE
- README.md

README | Apache-2.0 license

PowerRunAsSystem is a PowerShell script, also available as an installable module through the PowerShell Gallery, designed to impersonate the **NT AUTHORITY/SYSTEM** user and execute commands or launch interactive processes without relying on third-party tools. It achieves this using only native Windows build-in features.

Traditionally, elevating privileges to the SYSTEM user from an administrator account requires using tools like PsExec from Sysinternals or creating a custom service. With PowerRunAsSystem, you can accomplish the same goal using the built-in Windows Task Scheduler, eliminating the need for external utilities.

This tool allows you to:

About

PowerRunAsSystem is a PowerShell script, also available as an installable module through the PowerShell Gallery, designed to impersonate the NT AUTHORITY/SYSTEM user and execute commands or launch interactive processes without relying on third-party tools. It achieves this using only native Windows build-in features.

windows | system | powershell | interactive | process | nt

- Readme
- Apache-2.0 license
- Activity
- Custom properties
- 246 stars
- 4 watching
- 29 forks

Report repository

Releases 5

PowerRunAsSystem 5.0 Latest

- Impersonate the SYSTEM user in the current terminal session
- Run non-interactive commands as SYSTEM
- Launch a new interactive process as SYSTEM (tied to the active terminal session)

In cases where graphical access to the machine is unavailable, you can redirect the input/output of the spawned SYSTEM process to a listener (e.g., a Netcat listener) for interaction.

> It's important to note that administrative privileges are required to spawn a SYSTEM process in a standard configuration. Ensure that you either access a remote terminal (e.g., SSH or WinRM) with administrative rights or open a new terminal with elevated privileges (Run as Administrator).

Exported Functions

- `Invoke-SystemCommand`
- `Invoke-InteractiveSystemProcess`
- `Invoke-ImpersonateSystem`
- `Invoke-RevertToSelf`

Installation

PowerShell Gallery (Recommended)

The following commands for the installation process may require privileges (e.g., Administrative rights, appropriate Execution Policy settings). Ensure that you understand and meet these requirements before proceeding.

```
Install-Module -Name PowerRunAsSystem
Import-Module -Name PowerRunAsSystem
```

2 weeks ago

+ 4 releases

Packages

No packages published

Languages

● PowerShell 100.0%

Importing as a Script

```
IEX(Get-Content .\PowerRunAsSystem.ps1 -Raw -En
```

Usage

`Invoke-SystemCommand`

Spawn a new process as the SYSTEM user via Task Scheduler. Note that the SYSTEM process will not be tied to the active terminal session, meaning it won't be interactive. This is useful for quickly running commands as SYSTEM without needing direct interaction with the process.

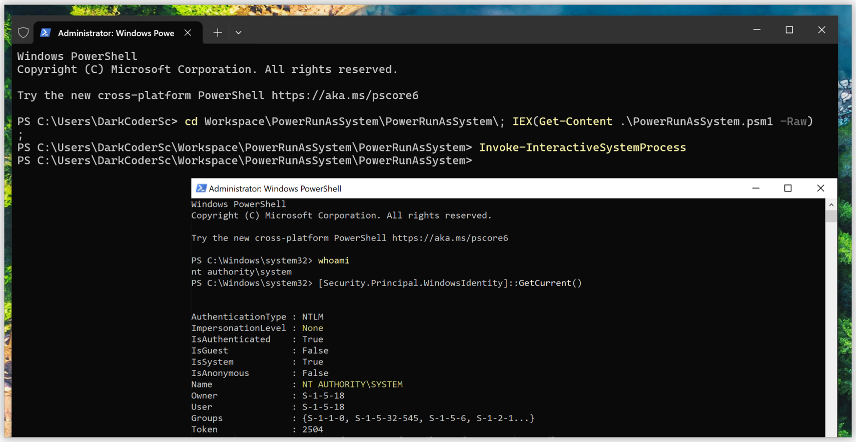
⚙️ Available Arguments

Parameter	Type	Default	Description
Application	String	powershell.exe	Program to execute
Argument	String	-Command "whoami | Out-File C:\result.txt"	Optional program arguments

`Invoke-InteractiveSystemProcess`

Spawn a new interactive process as the SYSTEM user, which will be tied to the active terminal session and, if selected, visible on the current desktop.

This can be particularly useful in scenarios where an interactive SYSTEM process is needed. For instance, when using Arcane Server, running it as an interactive SYSTEM process allows you to capture both the desktop and LogonUI/UAC prompts.


Administrator: Windows Powe
Administrator: Windows PowerShell

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

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\DarkCoderSc> cd Workspace\PowerRunAsSystem\PowerRunAsSystem\; IEX(Get-Content .\PowerRunAsSystem.psm1 -Raw)
;
PS C:\Users\DarkCoderSc\Workspace\PowerRunAsSystem\PowerRunAsSystem> Invoke-InteractiveSystemProcess
PS C:\Users\DarkCoderSc\Workspace\PowerRunAsSystem\PowerRunAsSystem>
```

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

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> whoami
nt authority\system
PS C:\Windows\system32> [Security.Principal.WindowsIdentity]::GetCurrent()


AuthenticationType : NTLM
ImpersonationLevel : None
IsAuthenticated    : True
IsGuest            : False
IsSystem           : True
IsAnonymous        : False
Name               : NT AUTHORITY\SYSTEM
Owner              : S-1-5-18
User               : S-1-5-18
Groups             : {S-1-1-0, S-1-5-32-545, S-1-5-6, S-1-2-1...}
Token              : 2504
```

⚙️ Available Arguments

Parameter	Type	Default	Description
CommandLine	String	powershell.exe	The complete command line to execute.
Hide	Switch	None	If present, the process is no visible.
RedirectKind	Choice	None	If the process input/output needs to be redirected to an external source (as discussed below)...
Address	String	None	Used if the **RedirectKin** is set (as described below).

Port	Int (R: 0-65535)	None	Used if the **RedirectKinc** is set (as described below).

Advanced Usage : RedirectKind Flag

`None` **(Default)**

No specific redirection is used; the process is spawned normally. To interact with the process, you must do so through the desktop.

If RedirectKind Flag is specified, the `stdout`, `stderr`, and `stdin` of the process are redirected to a network socket. This setup enables interaction with the spawned process without requiring access to the desktop, which is particularly useful when the process is initiated from an SSH or WinRM session.

`Bind`

Spawn your interactive SYSTEM process:

```
Invoke-InteractiveSystemProcess -RedirectKind "I
```

In the context of a bind shell, the address specifies the network interface to bind to. Using `0.0.0.0` means the shell will listen on all available network interfaces, while `127.0.0.1` restricts it to the loopback interface, making it accessible only from the local machine.

Then, with netcat, connect to listener:

```
nc 127.0.0.1 4444
```

In the context of a bind shell, it is important to note that the temporary SYSTEM process acting as the **launcher** will remain in a hanging state until a client connects to the listener. Only one client can connect to the listener, only once. Once connected, an interactive SYSTEM process will be established. When the session/process, both the client and listener will be released, marking the termination of the **launcher**.

`Reverse`

Create a new Netcat listener (adapt the command according to your operating system and version of Netcat):

```
nc -l 4444
```

Then, spawn your interactive SYSTEM process:

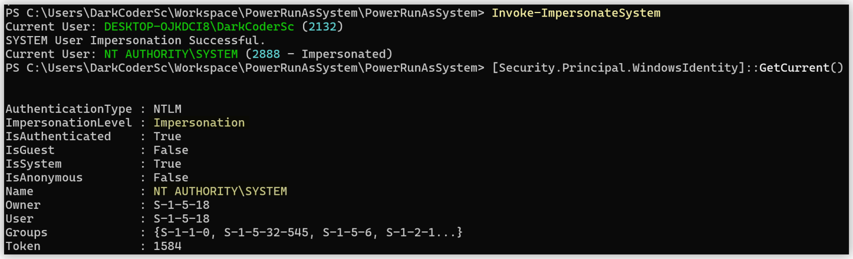
```
Invoke-InteractiveSystemProcess -RedirectKind "|
```

In the context of a reverse shell, it is important to note that a listener must be started before executing the reverse shell command. If the listener is not active, the attempt to spawn an interactive SYSTEM process will fail.

Enjoy your SYSTEM shell 🐚

`Invoke-ImpersonateSystem`

Impersonate the SYSTEM user within the current terminal session.

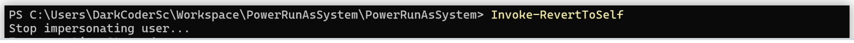


```
PS C:\Users\DarkCoderSc\Workspace\PowerRunAsSystem\PowerRunAsSystem> Invoke-ImpersonateSystem
Current User: DESKTOP-OJKDCI8\DarkCoderSc (2132)
SYSTEM User Impersonation Successful.
Current User: NT AUTHORITY\SYSTEM (2888 - Impersonated)
PS C:\Users\DarkCoderSc\Workspace\PowerRunAsSystem\PowerRunAsSystem> [Security.Principal.WindowsIdentity]::GetCurrent()


AuthenticationType : NTLM
ImpersonationLevel : Impersonation
IsAuthenticated    : True
IsGuest            : False
IsSystem           : True
IsAnonymous        : False
Name               : NT AUTHORITY\SYSTEM
Owner              : S-1-5-18
User               : S-1-5-18
Groups             : {S-1-1-0, S-1-5-32-545, S-1-5-6, S-1-2-1...}
Token              : 1584
```

`Invoke-RevertToSelf`

Stop user impersonation



```
PS C:\Users\DarkCoderSc\Workspace\PowerRunAsSystem\PowerRunAsSystem> Invoke-RevertToSelf
Stop impersonating user...
```