

 hlldz / **Phant0m** Public archive
Notifications
Fork 297
Star 1.8k

- [Code](#)
[Issues](#)
[Pull requests](#)
[Actions](#)
[Projects](#)
[Security](#)
[Insights](#)

master   [Code](#)

		🕒 19 Commits
📁 images	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
📁 phant0m	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
📄 README.md	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
📄 phant0m.cna	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>

 README





# Phant0m | Windows Event Log Killer

Svchost is essential in the implementation of so-called shared service processes, where a number of services can share a process in order to reduce resource consumption. Grouping multiple services into a single process conserves computing resources, and this consideration was of particular concern to NT designers because creating Windows processes takes more time and consumes more memory than in other operating systems, e.g. in the Unix family.<sup>1</sup>






This means briefly that; On Windows operating systems, `svchost.exe` manages the services and services are actually running under `svchost.exe`'s as threads. `Phant0m` targets the Event Log service and finding the process responsible for the Event Log service, it detects and kills the threads responsible for the Event Log service. Thus, while the Event Log service appears to be running in the system (because `Phant0m` didn't kill process), it does not actually run (because `Phant0m` killed threads) and the system does not collect logs.

## How It Works & How To Use

## About

# Windows Event Log Killer

- windows    cpp    powershell    cobalt-strike  
eventlog    reflective-dll    eventlog-service

-  [Readme](#)
-  [Activity](#)
-  [1.8k stars](#)
-  [61 watching](#)
-  [297 forks](#)

Report repository

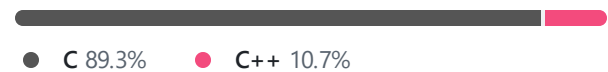
## Releases

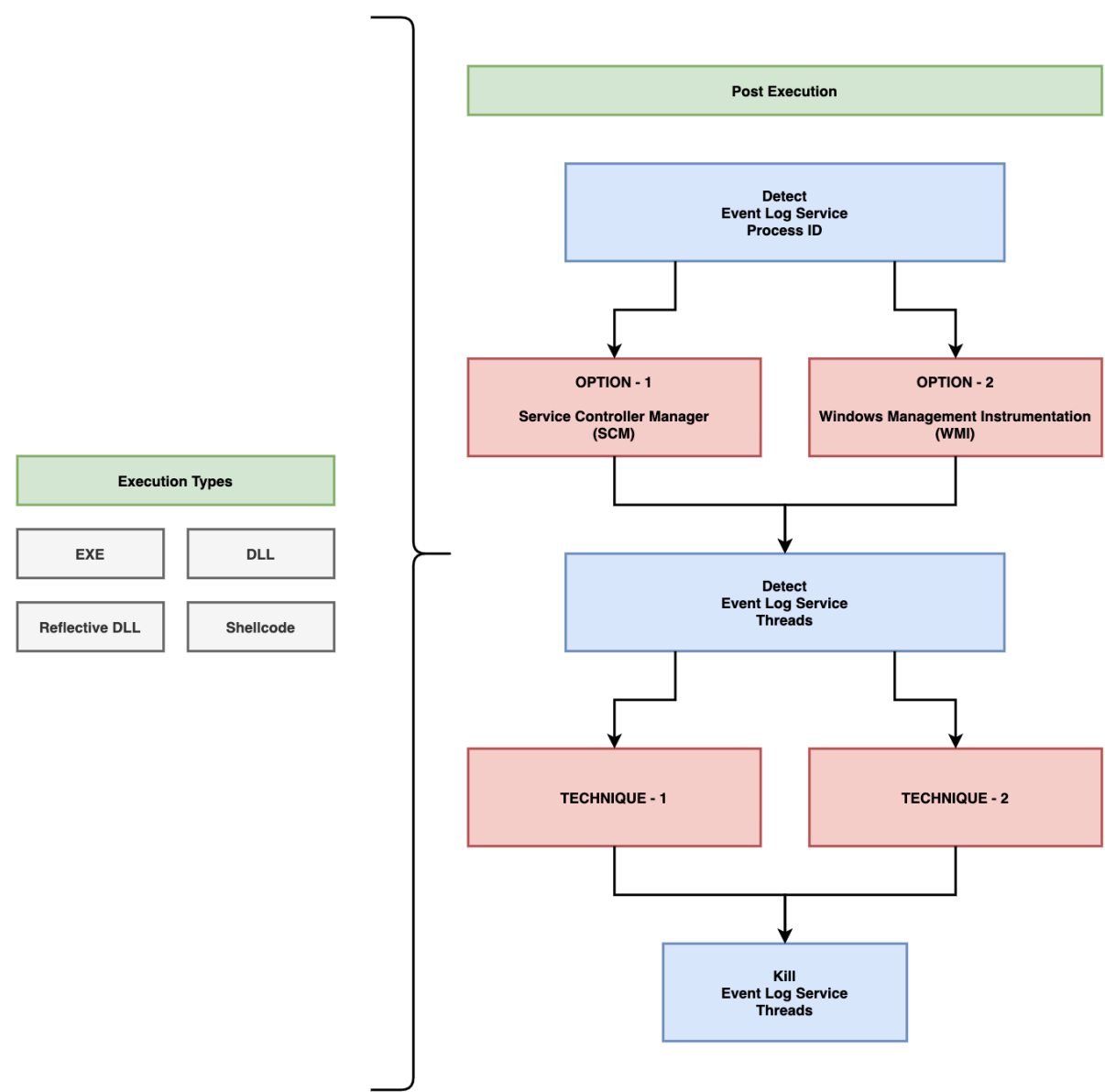
No releases published

## Packages

No packages published

## Languages





## Detecting Event Log Service

Phant0m uses two different options to detect the Process ID of the Event Log service. The first is to detect via the SCM (Service Control Manager) and the second is to detect via WMI (Windows Management Instrumentation). With which method you want Phant0m to detect the Process ID of the Event Log service, change the following lines in the main.cpp file.

For example, if you want the Process ID to be detected via SCM, you should edit it as follows. (Do not set all values at the same time, set only the one technique you want.)

```

// PID detection techniques configuration section.
#define PID_FROM_SCM 1 // If you set it to 1, the PID of the Event Log service will be detected via SCM.
#define PID_FROM_WMI 0 // If you set it to 1, the PID of the Event Log service will be detected via WMI.

```

For example, if you want threads to be killed using Technique-1, you should edit it as follows. (Do not set all values at the same time, set only the one technique you want.)

```

// TID detection and kill techniques configuration section.
#define KILL_WITH_T1 1 // If you set it to 1, Technique-1 will be used to kill the threads.
#define KILL_WITH_T2 0 // If you set it to 1, Technique-2 will be used to kill the threads.

```

## Detecting and Killing Threads

Phant0m uses two different options to detect and kill the threads of the Event Log service.

### Technique-1

When each service is registered on a machine running Windows Vista or later, the Service Control Manager (SCM) assigns a unique numeric tag to the service (in ascending order). Then, at service creation time, the tag is assigned to the TEB of the main service thread. This tag will then be propagated to every thread created by the main service thread. For example, if the Foo service thread creates an RPC worker



- Event Log Service – Between Offensive And Defensive - <https://blog.cybercastle.io/event-log-service-between-offensive-and-defensive/>
- Hunting Event Logging Coverup - <https://malwarenailed.blogspot.com/2017/10/update-to-hunting-mimikatz-using-sysmon.html>
- Defense Evasion: Windows Event Logging (T1562.002) - <https://hacker.observer/defense-evasion-windows-event-logging-t1562-002/>
- Pwning Windows Event Logging with YARA rules -

