Silver Bullet for High Processor Usage Issues

zerosalarium.com/2025/01/100 CPU High Processor Usage Silver Bullet.html

Zero Salarium January 17, 2025

100 CPU: Silver Bullet for High Processor Usage Issues

I. Introduction

Is your computer as hot as a frying pan full of boiling oil due to high CPU consumption issues? Have you searched the entire Internet for solutions, but no one seems to have encountered the same situation as you? At this point, you should conduct CPU troubleshooting meticulously to identify the root cause of the problem.

I will introduce and guide you on how to use "**100 CPU**" for CPU troubleshooting. This is an open-source tool that is very simple to use.

II. Main section

1. Some common processes related to high CPU usage issues

In practice, any program that is corrupted or encounters a logic bug during operation has a high likelihood of causing excessive CPU usage. Below are some programs that frequently experience this issue:

- Antimalware Service Executable high cpu
- WMI Provider Host high cpu
- Desktop Window Manager high cpu
- Microsoft Compatibility Telemetry high cpu
- …

Although the processes may be the same, the causes of high processor usage can vary from machine to machine. Users often give up on this issue because they cannot understand what is happening on their computer that has led to the high CPU usage problem.

2. Introduction to "100 CPU"

"**100 CPU**" is an open-source CPU troubleshooting program. Its purpose is to simplify the monitoring of program activity, collect statistics, and identify bottlenecks occurring on the computer.

The real cause of high processor usage is when a program enters an infinite loop that it cannot escape from. The commands within this loop will be executed repeatedly without interruption, which will result in you observing 100% usage of CPU when monitoring with **Task Manager**.

Some practical examples that cause high CPU consumption include:

- A program attempts to write to a file, but this file may be placed in a sandbox, resulting in a failed write operation. However, the programmer, when creating the program, has placed the file writing operation in a loop that continues until the write is successful and complete. This condition will never be met, causing the program to get stuck in a state of continuously attempting to write to the file.
- An updater downloads a patch from the Internet. The download process fails (due to the patch being deleted, network disconnection, non-existent URL, etc.). Instead of exiting or waiting for a period, it continuously requests the download until it succeeds, pushing CPU usage to high levels.
- A program reads configuration keys from the registry. However, these keys may be corrupted or inadvertently changed (due to user edits, deletion during an antivirus scan, etc.). Without sufficient and correct configuration information to operate, this leads to subsequent logic bugs. Alternatively, it may enter an infinite loop while reading the registry. If this occurs for an extended period, you will experience laptop CPU overheating.

With "**100 CPU**" you can pinpoint where the program is stuck in an infinite loop without any breakpoints. From this, tailored solutions can be devised for each specific case.

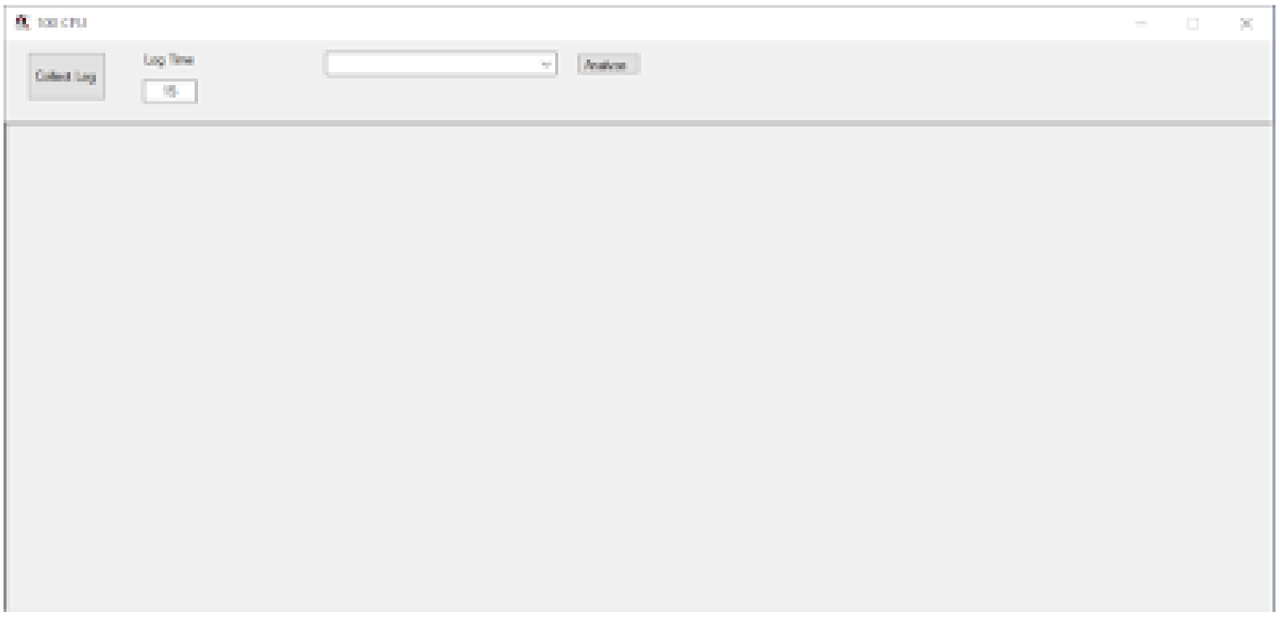
3. Utilizing "100 CPU" to address the issue of high CPU consumption

A. How to use "100 CPU"

First, you can download the "**100 CPU**" program, including its source code, from the GitHub link:

https://github.com/TwoSevenOneT/100-CPU

The program has a very simple interface, as shown in the image below.



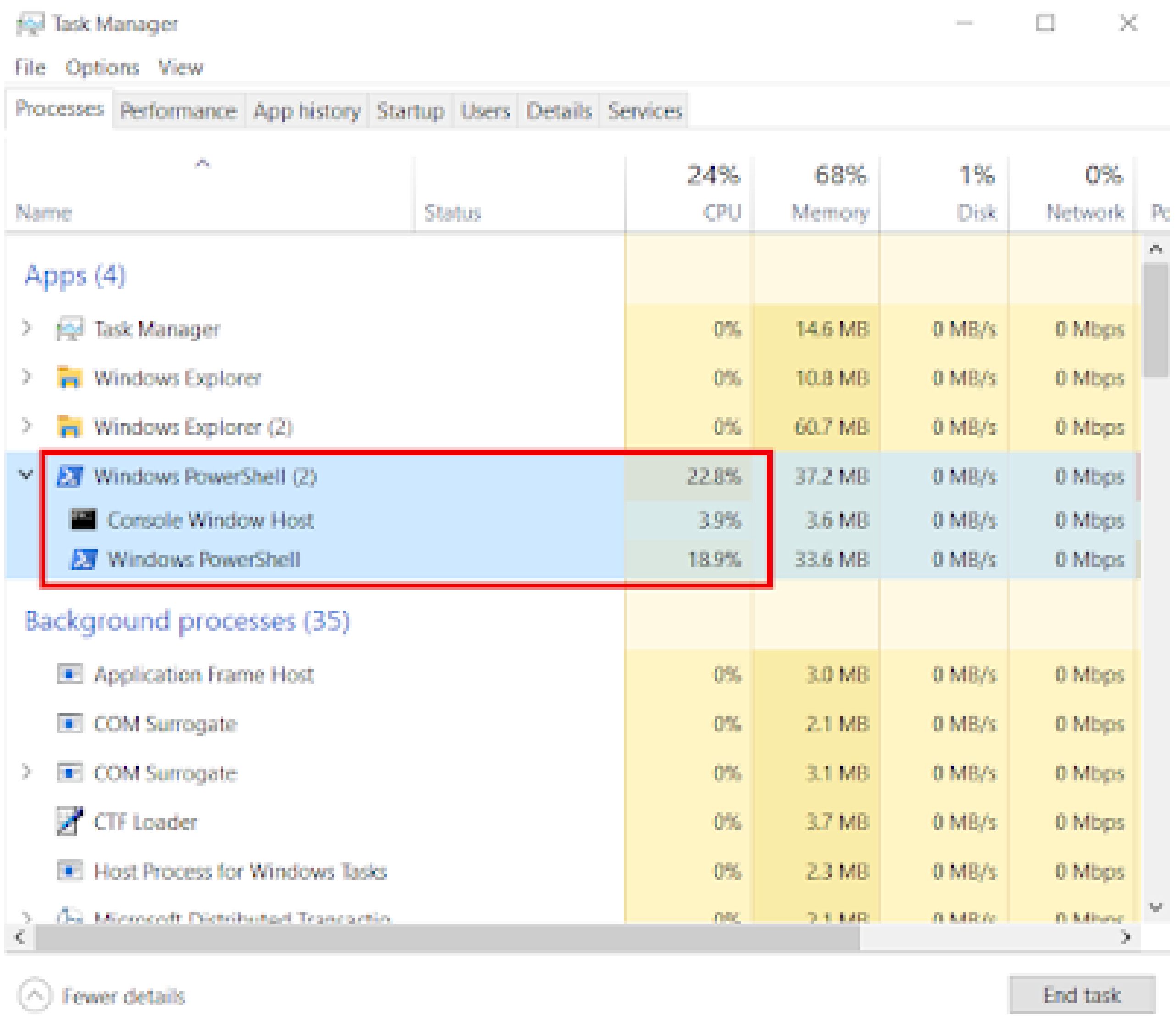
After downloading and extracting, you can proceed with the following steps:

1. Reset Windows or close all running programs. Ensure that the program causing excessive CPU usage is active. The purpose of this is to minimize interference from other programs, allowing for more accurate monitoring.
2. Run "**100 CPU.exe**" with Administrator privileges. This is necessary to have sufficient permissions to monitor the entire system.
3. Adjust the "**Log Time**" as desired (measured in seconds). "**Log Time**" is the duration for which "**100 CPU**" will monitor and record events on your machine into its local database. Then, click the "**Collect Log**" button.
4. The program may appear to be frozen during the "**Log Time**". Since "**100 CPU**" uses Process Monitor to collect events, it will pause and wait for **Process Monitor** to run in the background until event collection is complete. This helps to minimize the amount of CPU it consumes. Your machine is experiencing high CPU consumption, right?
5. Once the logs have been collected, select the process that you know is causing the high processor usage. Alternatively, if you're unsure, allow the "**100 CPU**" program to analyze and identify the bottleneck by leaving the default setting as "**Auto Detect**".
6. Click "**Analyze**".

"**100 CPU**" will provide information on the activity of the program of interest that is causing high processor usage. Alternatively, with "**Auto Detect**" it will automatically identify which program is leading to high CPU usage and the reasons behind it. All information will be displayed in the log window for easy monitoring.

B. "100 CPU" real-world use case

Currently, there is a PowerShell program on my machine causing high processor usage, and I will use "**100 CPU**" to analyze this case.


Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Fewer details
End task

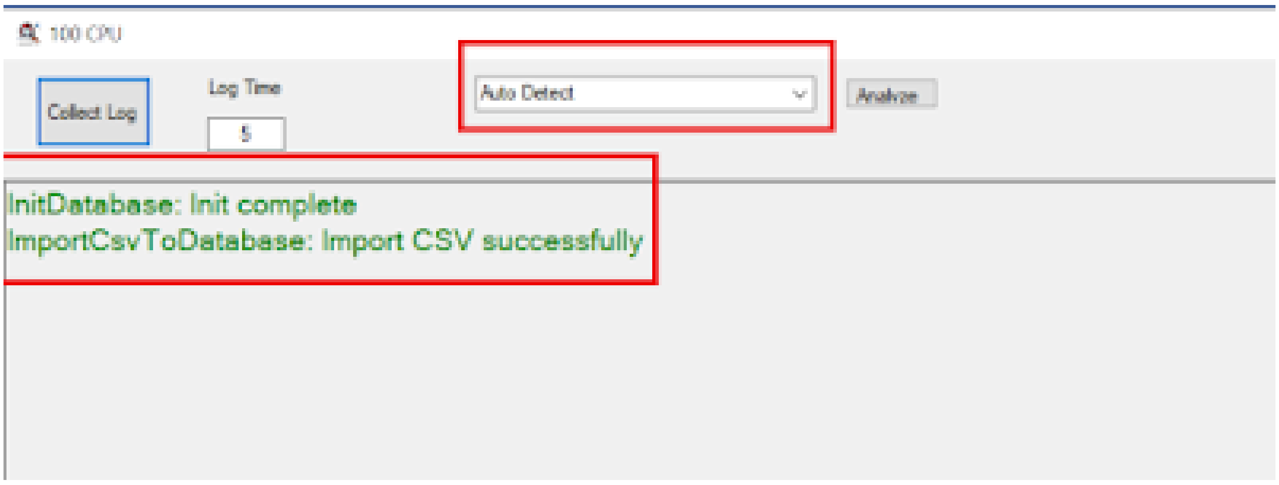
Name	Status	24% CPU	68% Memory	1% Disk	0% Network
Apps (4)					
Task Manager		0%	14.6 MB	0 MB/s	0 Mbps
Windows Explorer		0%	10.8 MB	0 MB/s	0 Mbps
Windows Explorer (2)		0%	60.7 MB	0 MB/s	0 Mbps
Windows PowerShell (2)		22.8%	37.2 MB	0 MB/s	0 Mbps
Console Window Host		3.9%	3.6 MB	0 MB/s	0 Mbps
Windows PowerShell		18.9%	33.6 MB	0 MB/s	0 Mbps
Background processes (35)					
Application Frame Host		0%	3.0 MB	0 MB/s	0 Mbps
COM Surrogate		0%	2.1 MB	0 MB/s	0 Mbps
COM Surrogate		0%	3.1 MB	0 MB/s	0 Mbps
CTF Loader		0%	3.7 MB	0 MB/s	0 Mbps
Host Process for Windows Tasks		0%	2.3 MB	0 MB/s	0 Mbps
Microsoft Distributed Transactio		0%	2.1 MB	0 MB/s	0 Mbps

Step 1: I reset Windows and noticed that the PowerShell program is still causing excessive CPU usage. To avoid interference and achieve the most accurate results, I will close all unrelated programs.

Step 2: I run the "**100 CPU**" program (with Administrator privileges) and adjust the "**Log Time**" to 5 seconds. Then, I click on the large "**Collect Log**" button.

Step 3: After clicking the "**Collect Log**" button, you will notice that "**100 CPU**" seems to freeze and become unresponsive for about 5 seconds, along with some windows briefly appearing. This is when it relinquishes full operational control to the **Process Monitor** running in the background.

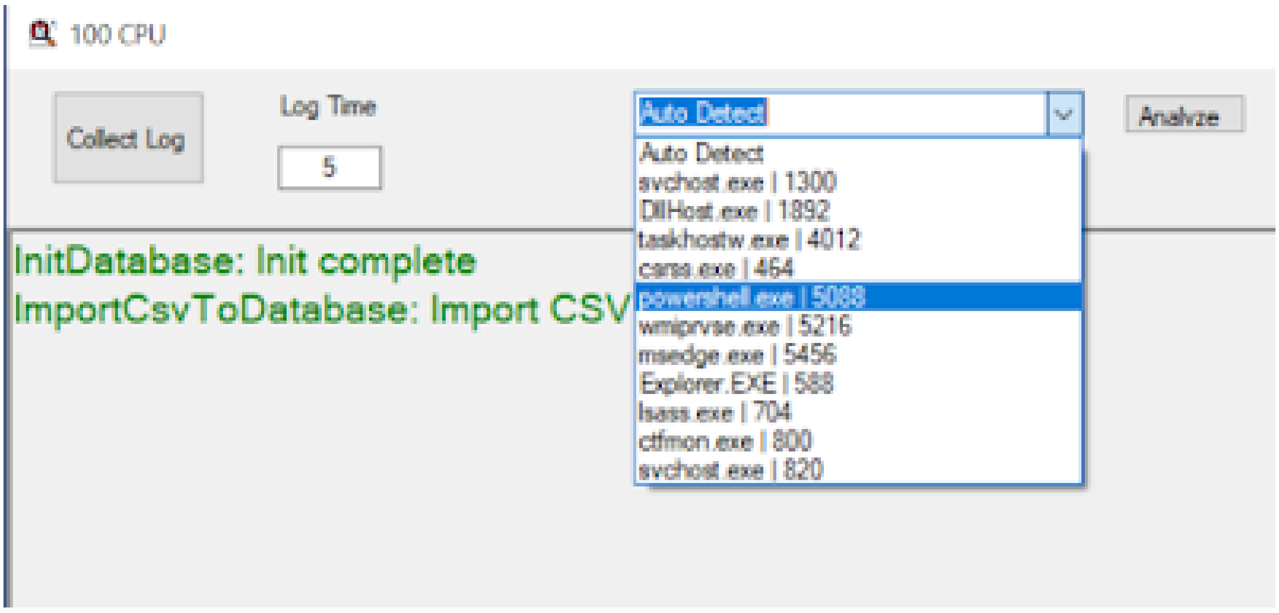
If "**100 CPU**" successfully collects the logs, there will be a green notification in the log window.



```
InitDatabase: Init complete
ImportCsvToDatabase: Import CSV successfully
```

If you pay attention, the combo box next to the "**Analyze**" button now also contains data for selection.

Step 4: Since I already know that the PowerShell program is experiencing high CPU consumption, I will select the process **Powershell.exe** from the combo box.

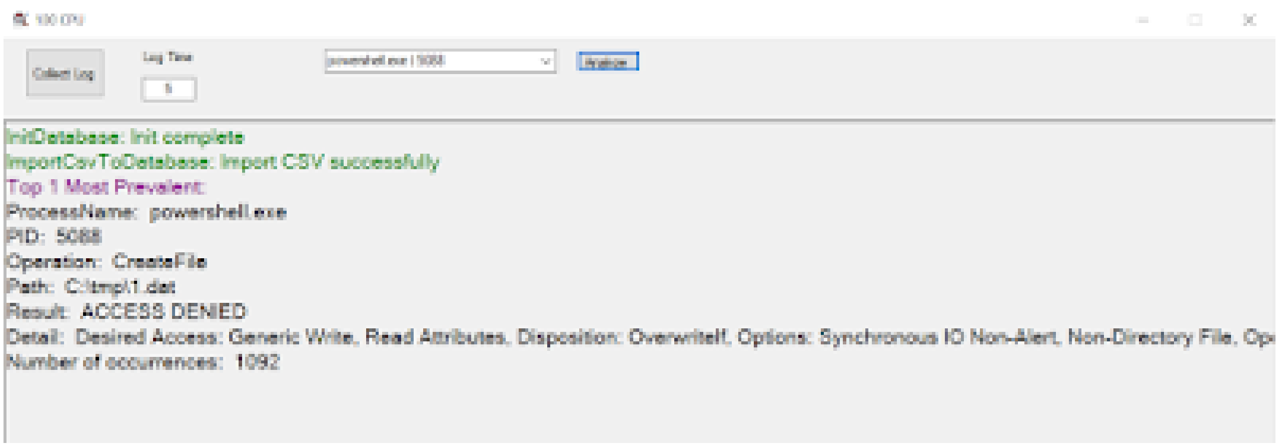


```
Auto Detect
svchost.exe | 1300
DllHost.exe | 1892
taskhostw.exe | 4012
csrss.exe | 464
powershell.exe | 5088
wmiprvse.exe | 5216
msedge.exe | 5456
Explorer.EXE | 588
lsass.exe | 704
ctfmon.exe | 800
svchost.exe | 820
```

```
InitDatabase: Init complete
ImportCsvToDatabase: Import CSV
```

The format of the fields in this combo box is: **[Process name] | [PID]**.

Step 5: After selecting powershell.exe, I click "**Analyze**". "**100 CPU**" will provide the results of the activity that powershell.exe executed most frequently during the 5 seconds ("**Log Time**").

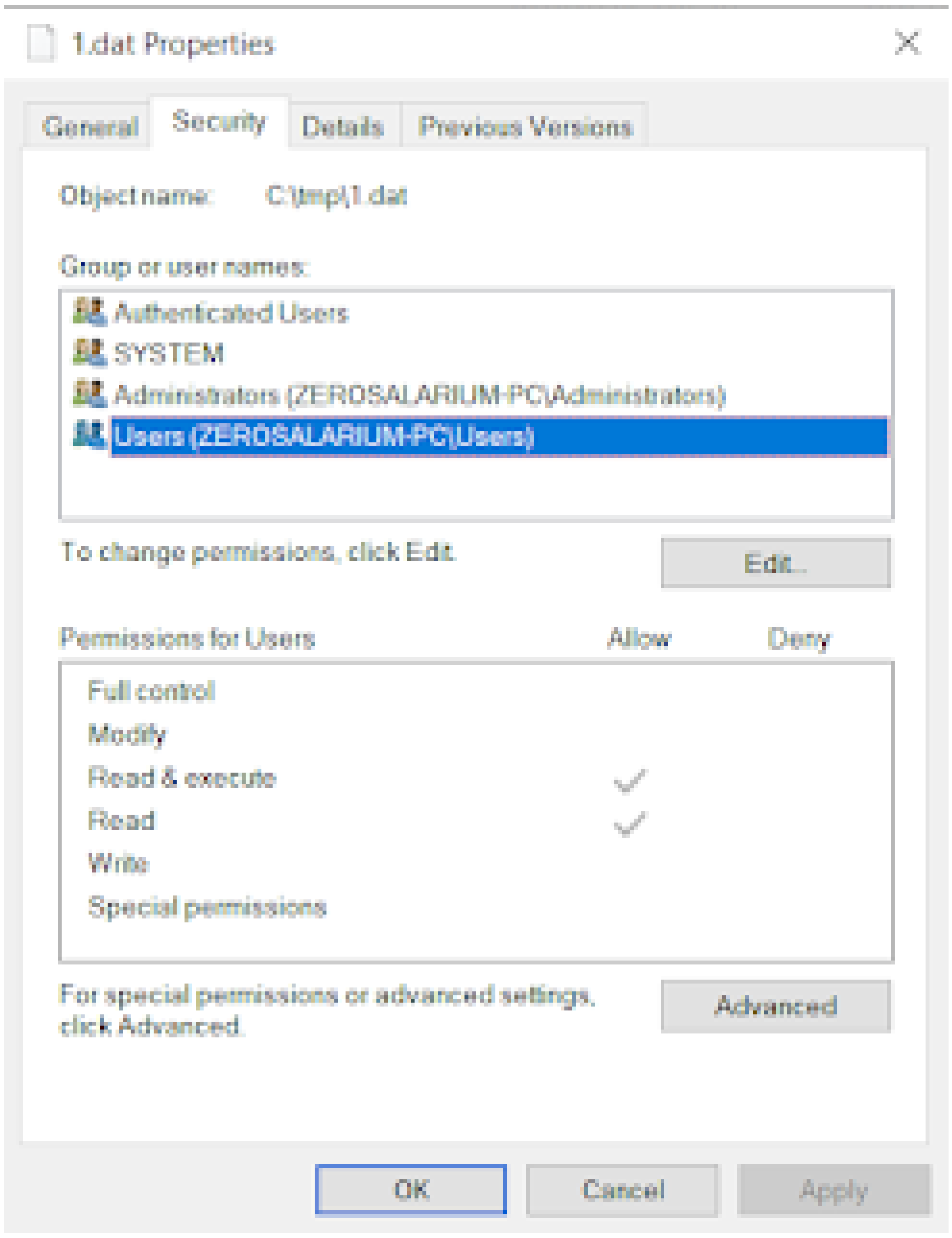


```
InitDatabase: Init complete
ImportCsvToDatabase: Import CSV successfully
Top 1 Most Prevalent:
ProcessName:  powershell.exe
PID:  5088
Operation:  CreateFile
Path:  C:\tmp\1.dat
Result:  ACCESS DENIED
Detail:  Desired Access: Generic Write, Read Attributes, Disposition: OverwriteIf, Options: Synchronous IO Non-Alert, Non-Directory File, Op
Number of occurrences:  1092
```

According to the information obtained from the log window, I discovered that in the past 5 seconds, the process "**powershell.exe**" performed the “**CreateFile**” ("**Operation**") to write ("**Detail**") to the file "**C:\tmp\1.dat**" ("**Path**") a total of **1092** times ("**Number of occurrences**").

And the reason for the repeated **CreateFile** operation is due to the interaction with the file for writing being "**ACCESS DENIED**". This may be because, during programming, the developer requested excessive permissions (only write access is needed), or the file's security attributes have been altered.

To resolve the issue of repetition in the "**CreateFile**" operation, the simplest and most feasible approach is to check the file security attributes of the file "**C:\tmp\1.dat**"


1.dat Properties

General | Security | Details | Previous Versions

Object name: C:\tmp\1.dat

Group or user names:

- Authenticated Users
- SYSTEM
- Administrators (ZEROSALARIUM-PC\Administrators)
- Users (ZEROSALARIUM-PC\Users)

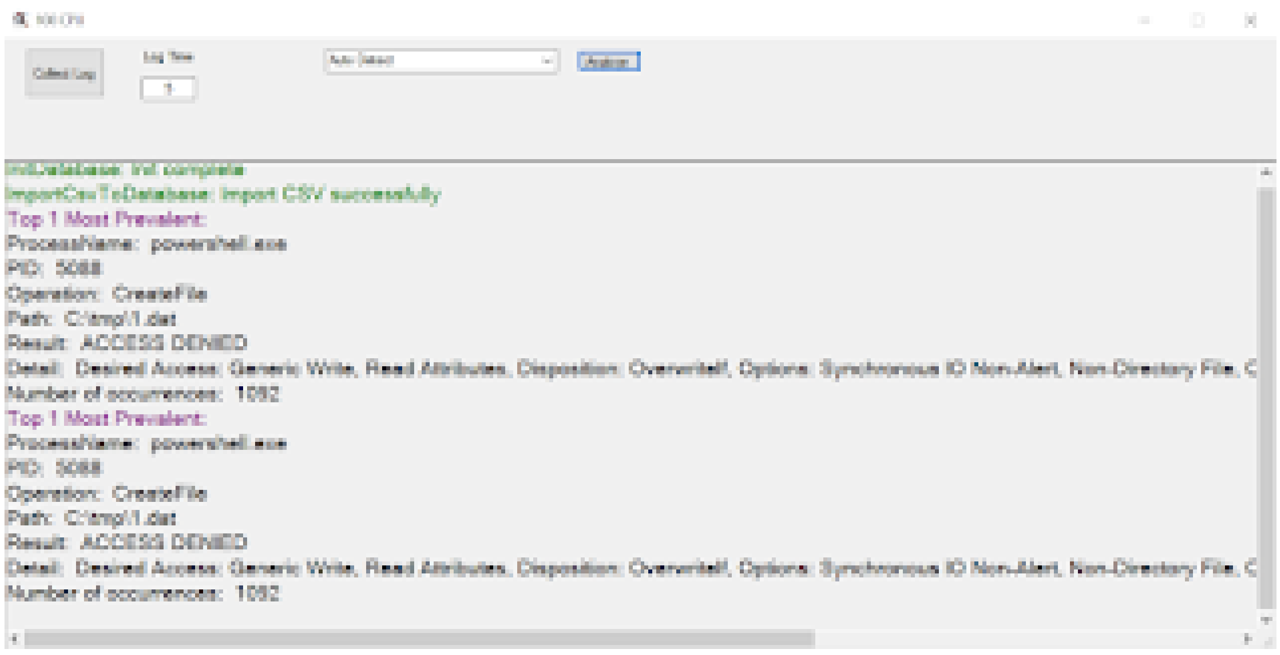
To change permissions, click Edit.

Permissions for Users	Allow	Deny
Full control		
Modify		
Read & execute	✓	
Read	✓	
Write		
Special permissions		

For special permissions or advanced settings, click Advanced.

According to the image above, the file "**1.dat**" currently only allows users to read and execute the file, and does NOT grant write permissions. The solution is to add write permissions for users in the security attributes of the file, or to change the settings so that "powershell.exe" runs with elevated privileges.

In cases where I am not entirely sure about identifying which process is causing excessive CPU usage, instead of selecting the exact process name, I will leave it set to "**Auto Detect**" by default.



```
InitDatabase: Init complete
ImportCsvToDatabase: Import CSV successfully
Top 1 Most Prevalent:
ProcessName:  powershell.exe
PID:  5088
Operation:  CreateFile
Path:  C:\tmp\1.dat
Result:  ACCESS DENIED
Detail:  Desired Access: Generic Write, Read Attributes, Disposition: OverwriteIf, Options: Synchronous IO Non-Alert, Non-Directory File, C
Number of occurrences:  1092
Top 1 Most Prevalent:
ProcessName:  powershell.exe
PID:  5088
Operation:  CreateFile
Path:  C:\tmp\1.dat
Result:  ACCESS DENIED
Detail:  Desired Access: Generic Write, Read Attributes, Disposition: OverwriteIf, Options: Synchronous IO Non-Alert, Non-Directory File, C
Number of occurrences:  1092
```

At this point, "**100 CPU**" also accurately identifies the process "**powershell.exe**" as the one causing high CPU consumption, yielding results similar to when I explicitly select the process name for analysis.

III. Summary

Excessive CPU usage occurs when a program encounters a logic bug during operation, and this bug causes the program's execution thread to get stuck in an infinite loop.

The causes of high CPU usage due to logic bugs often vary across different systems. This makes it very challenging for users to find effective solutions for remediation.

If the high processor usage persists for an extended period, you will notice that the issue of laptop CPU overheating occurs more frequently, negatively impacting the device's performance and longevity.

Some common real-world cases that users often encounter include: WMI Provider Host high CPU, Desktop Window Manager high CPU, Microsoft Compatibility Telemetry high CPU, and so on.

"**100 CPU**" is a free open-source tool. It simplifies the most challenging step of identifying the logical bug causing high CPU consumption, thereby helping users find accurate and effective solutions for remediation.

To keep your computers safe, clean, and operating more efficiently, you should explore the series of articles on basic Operations Security that I have written specifically for users of the Windows operating system.

You can contact me directly via X (Twitter): **Two Seven One Three** (@TwoSevenOneT)