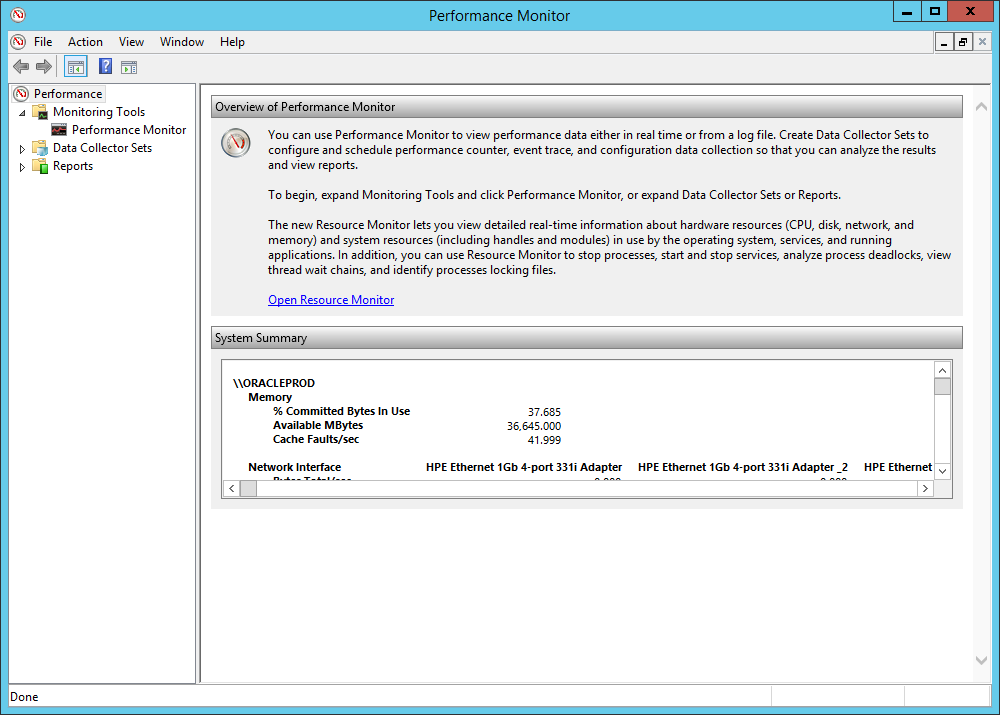
**How to create a perfmon-counter?**

# Understanding Processor (% Processor Time) and Process (%Processor Time)

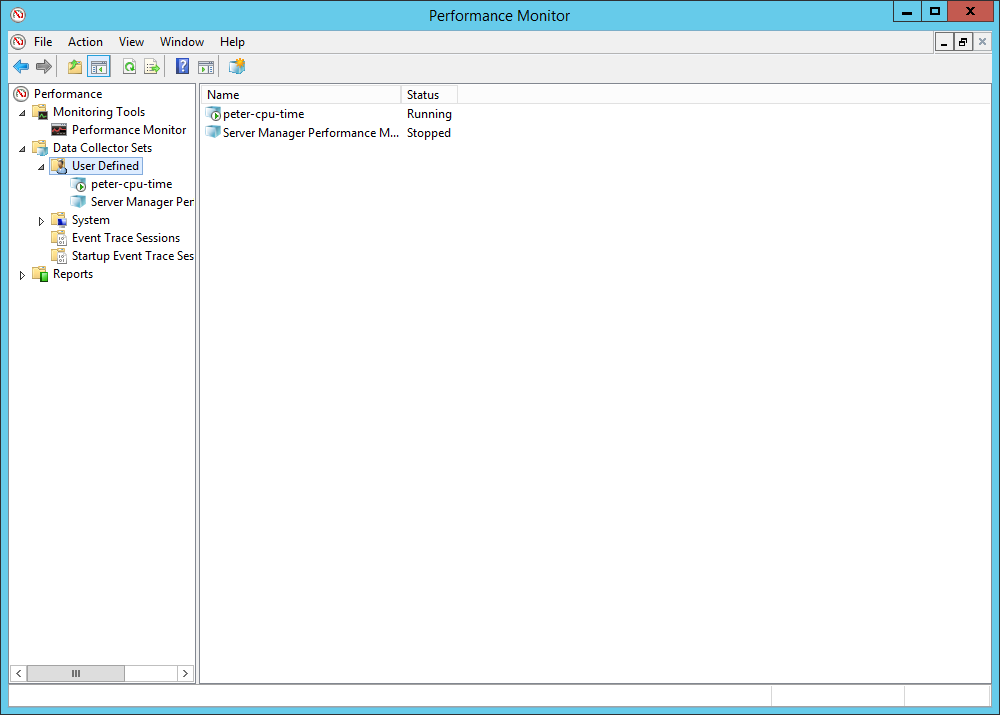
We come up so many times with the counters Processor (%Processor Time) and Process (%Processor Time) and when we see that there is high CPU in some server we want to analyze where this high CPU is coming from, which process is taking up our CPU. We may come up with a scenario where we observe a high CPU in the server at a particular point of time and want to find the cluprit.  
The Processor Object in the perfomance counter gives us a complete details on how our processor that is driving our computer is being used. If you are using a Multi Core processor you will see the instances of the each and every core of the server, and selecting the appropriate counter we can even calculate on how each individual CPU is being utilized.  
  
And now if we go to the description of the counter (%Processor Time) in the Processor Object you will get it as  
  
**% Processor Time is the percentage of elapsed time that the processor spends to execute a non-Idle thread.   
It is calculated by measuring the percentage of time that the processor spends executing the idle thread and then subtracting that value from 100%. (Each processor has an idle thread that consumes cycles when no other threads are ready to run).   
This counter is the primary indicator of processor activity, and displays the average percentage of busy time observed during the sample interval. It should be noted that the accounting calculation of whether the processor is idle is performed at an internal sampling interval of the system clock (10ms). On todays fast processors, % Processor Time can therefore underestimate the processor utilization as the processor may be spending a lot of time servicing threads between the system clock sampling interval. Workload based timer applications are one example  of applications  which are more likely to be measured inaccurately as timers are signaled just after the sample is taken.**So we can generally take this Processor Time Counter as a Bench Mark in understanding the utilization of our Processor.  
Suppose lets think over a standalone server we have hosted multiple SQL Instances and one of the Instance is consuming most of the CPU, or some antivirus or some Disk Backup is consuming the processor leading to the CPU Bottle necks for the other instance. If at that point of time if we are logged on to the server and if we see the task Manager and if we sort out the processes on the basis of the CPU we will be able to identify the process. What if it is happening daily at midnight when you have to go to sleep. How you are going to Monitor it?  
  
The first thing that will come into your mind is configuring a Data Collector Set, Now what are you going to add in it?  
  
So the solution is we have this object called **Process.**This Process object has the instances of all the process that are running over the server. And this counter has the objects which give us the Properties and the Settings that these processes are consuming. You will find so many Counters in the Process object, but today our main discussion is over Processor Time.  
  
If you look into the description of the Process (%Processor Time) it looks like  
  
**% Processor Time is the percentage of elapsed time that all of process threads used the processor to execution instructions. An instruction is the basic unit of execution in acomputer, a thread is the object that executes instructions, and a process is the object created when a program is run. Code executed to handle some hardware interrupts and trap conditions are included in this count.**

Now this Processor time counter under the **Process**Object gives the amount of CPU this Process is taking individually. This value is calculated over the base line of  
(No of Logical CPUS \* 100),  So this is going to be a calculated over a baselin of more than 100.  
  
Lets think you are using a QUADCore System, so the baseline over which this Value is going to be calculated is 400. So now if we run the Datacollector Set and capture all the Process and sort them by the order of their Processor time, we will come up with the Process that is taking the most CPU in the server.  
  
There may come up a condition that this Value is more than 400 ie. more than (No of Logical CPUS \* 100) this indicates that the Process is extensively using Multithreading and using the Processor to more than its capacity.  
  
If there more than one process of the same type are running in the Server you have to tactifully identify the right instnace of the processes by just adding the thread count. for example: you may stumble upon the instances SQLSrvr, SQLsrvr\_1, SQLSrvr\_2. So these are individual processes by simply adding the thread count in TaskManager and the Thread Count in the Processor Object you can easily identify which Processor is which SQL.

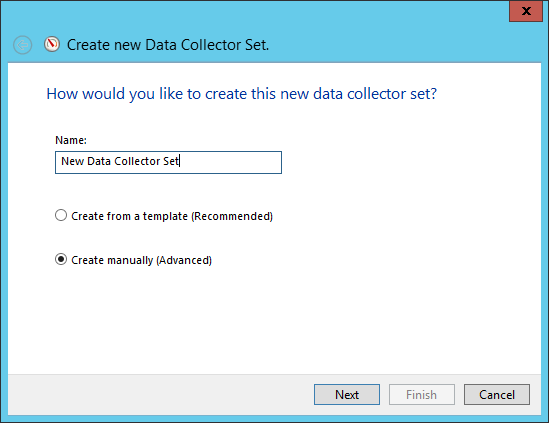
**Open perfmon**



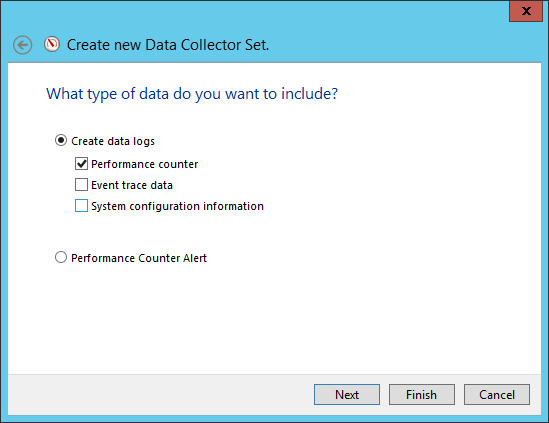
**Ga naar DATA-COLLECTOR-SETS**



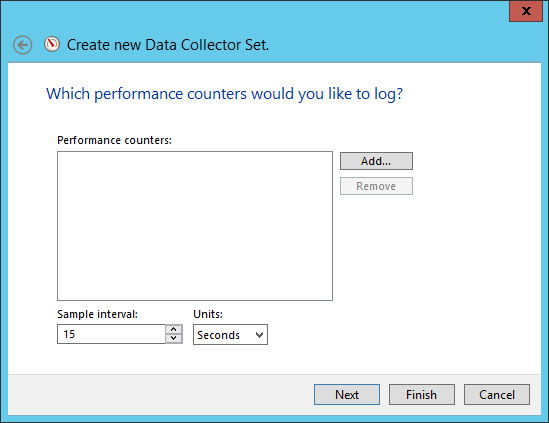
**Ga naar USER-DEFINED en kies vervolgens via RMK de optie [NEW] + [ DATA-COLLECTOR-SET]**



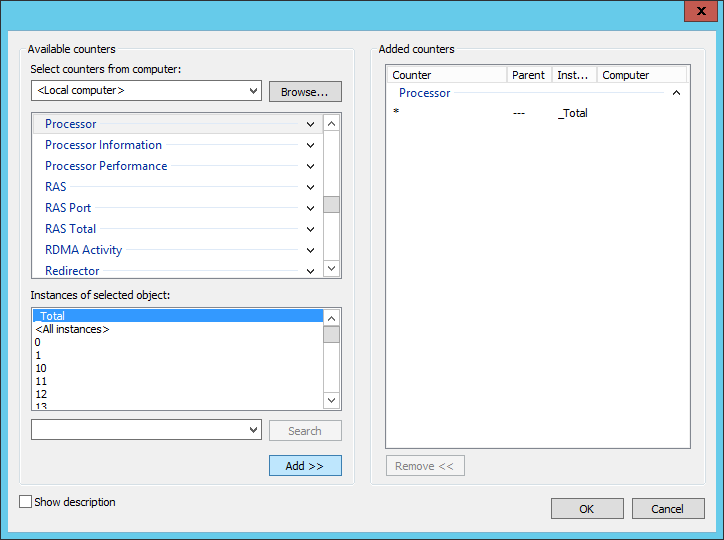
**Geef een naam op, en kies optie [create manually]**



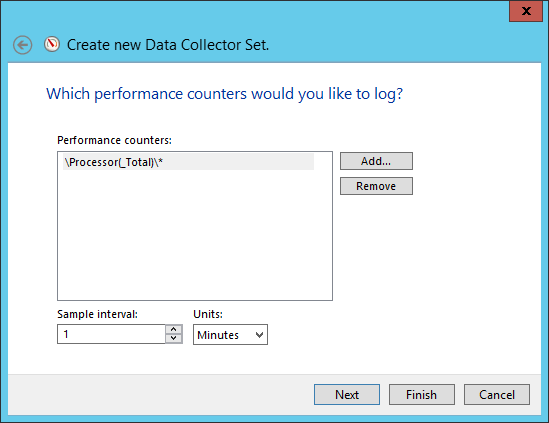
**Selecteer [performance counter]**



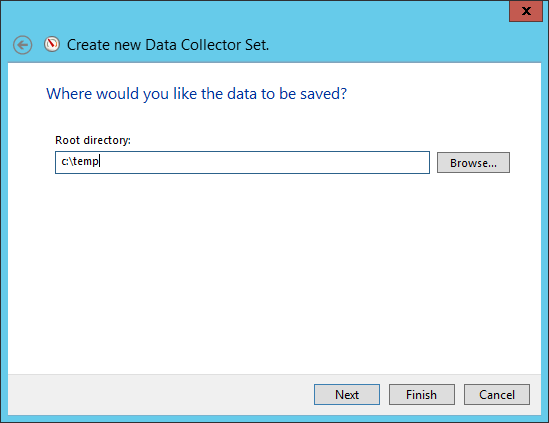
**Kies [ADD]**



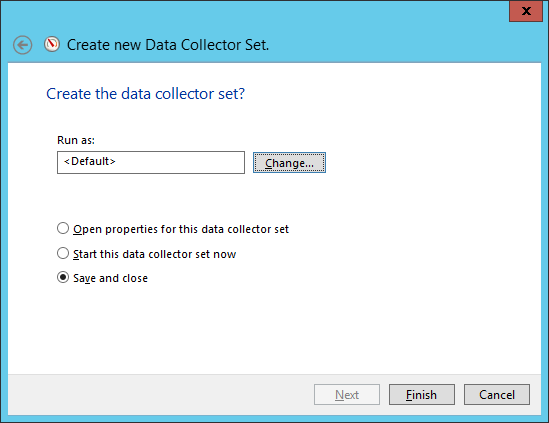
**Kies processor + \_total en ADD deze naar het rechter gedeelte van het scherm**



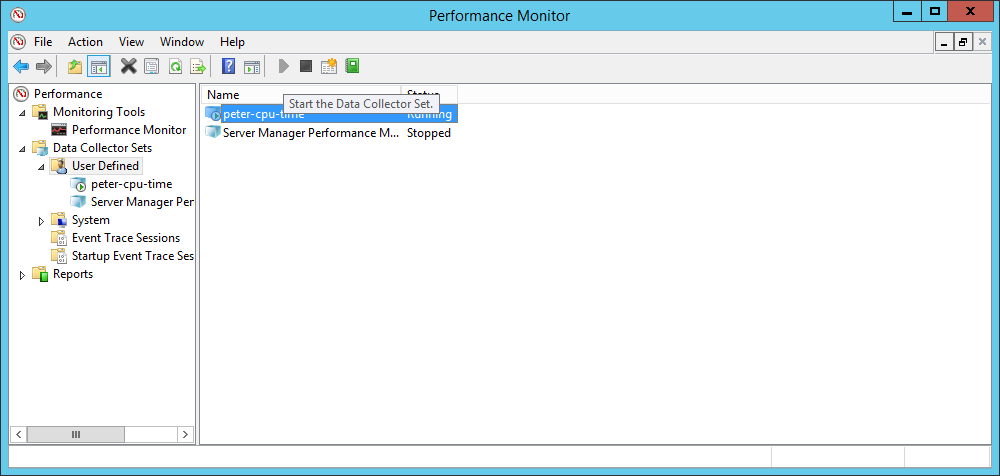
**Vul frequentie ook nog in, bijv 1x/min.**



**Geef directory op waar log-file bewaard moet worden**



**Kies default [save + close] en FINISH**



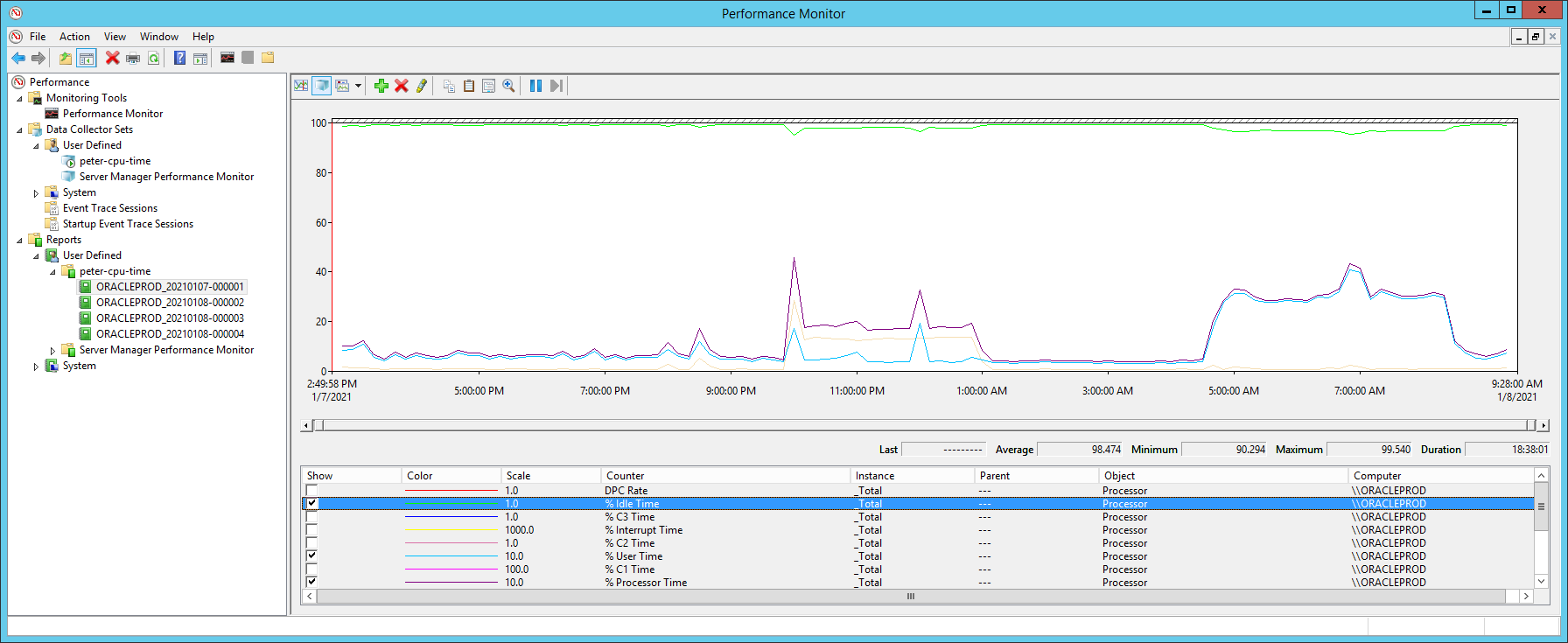
**vanuit perfmon selecteer de collector + kies voor de [START] van de collector.**

**Na een bepaalde periode als je genoeg statistieken verzameld hebt dan stop je de COLLECTOR.**

**Hierna kun je mbv een REPORT de resultaten ervan bekijken.**

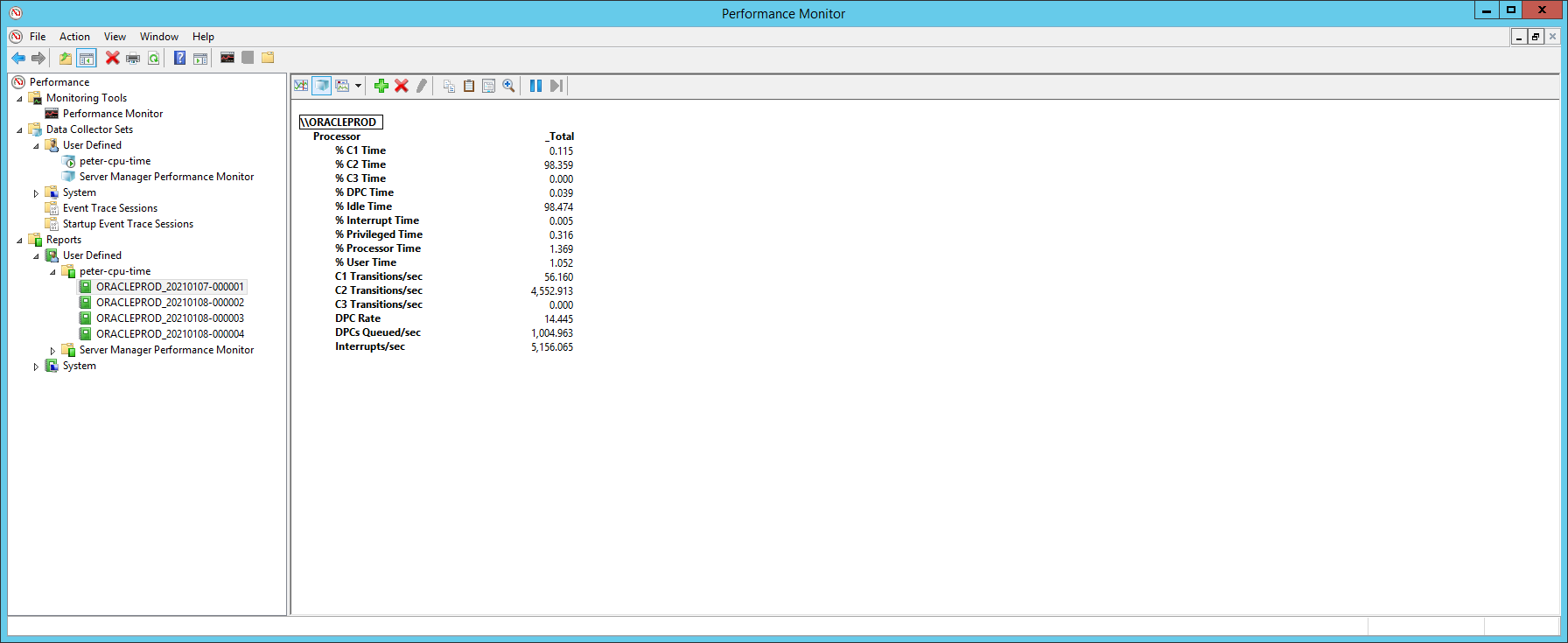
**Ga vanuit performance-monitor naar [REPORTS] en kies daar het bijbehorende REPORT**

**% Privileged Time + % User Time + % Idle Time = 100%**

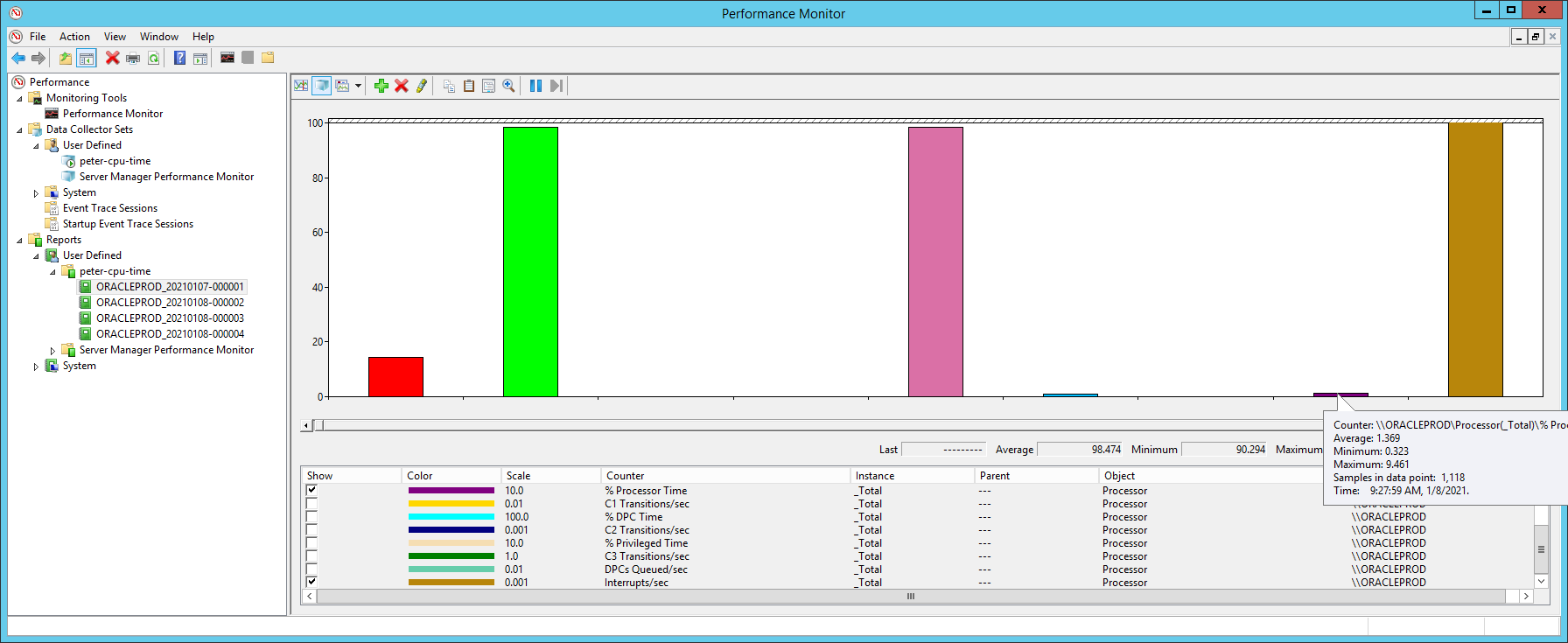


**Let op: IDLE-TIME is normale SCALE, maar OVERIGE zijn met factor-10 vermenigvuldig om ze zichtbaar te maken in de grafiek !!!.**

**Graph-type = REPORT**



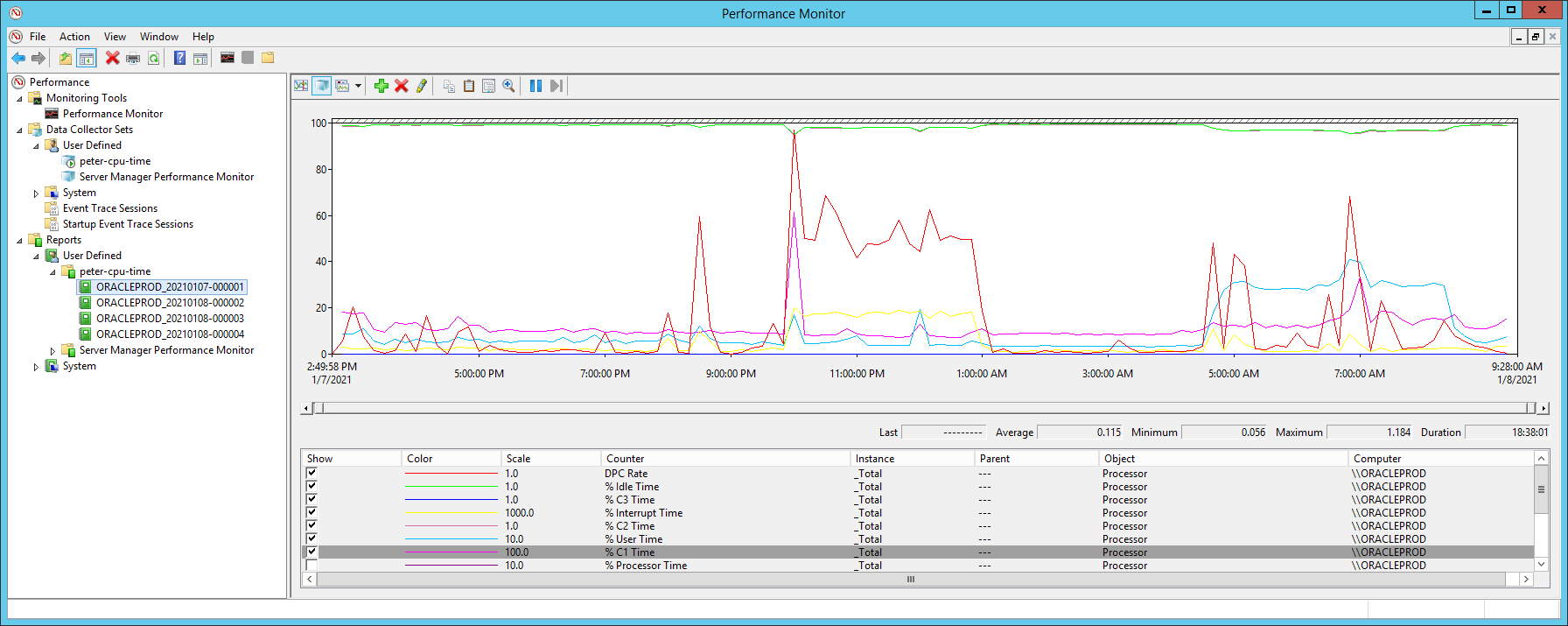
**Graph-type = HISTOGRAM-BAR**



**Graph-type = LINE**

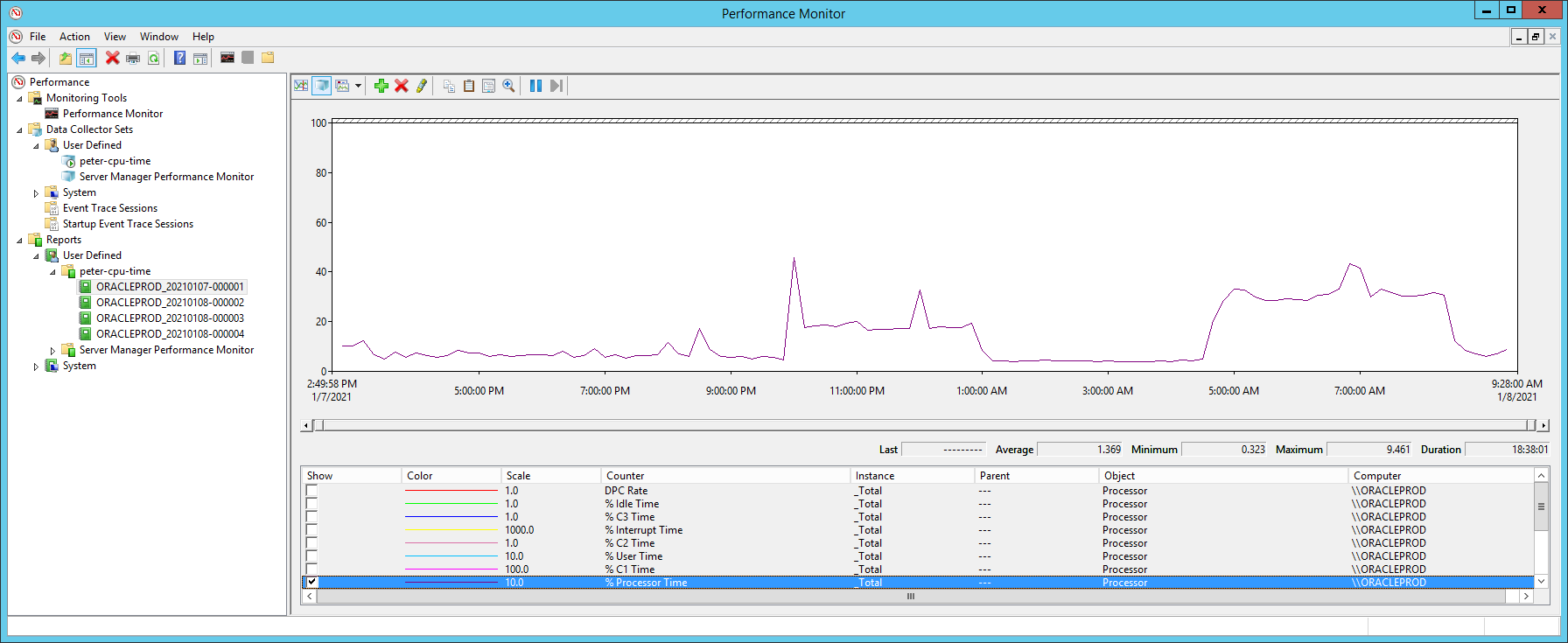
**Let op: Deze grafiek bevat per onderdeel een aparte SCALE, waardoor het eea. Vertekend beeld oplevert !!!**

**Deze waarde is vermenigvuldigingsfactor waarmee waarde is vergroot om deze in de grafiek te kunnen zien !!**

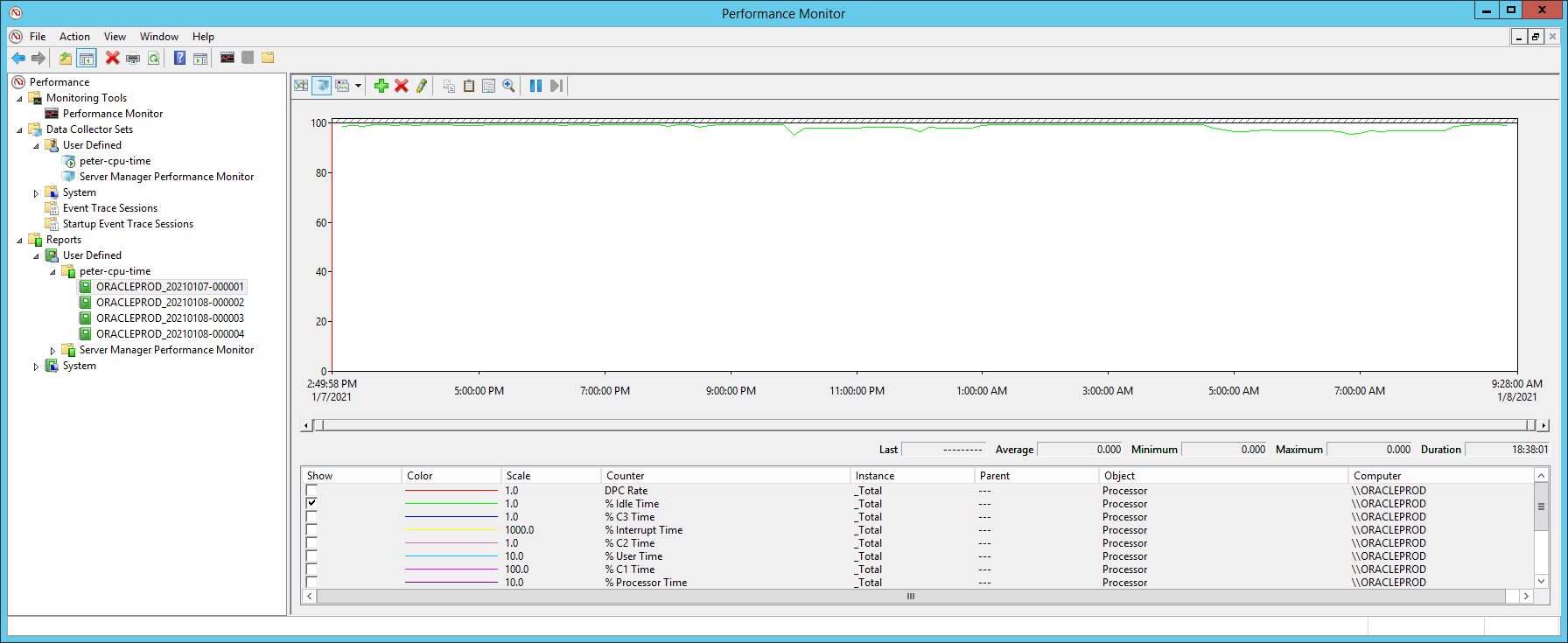


**Wat betekenen de verschillende componenten:**

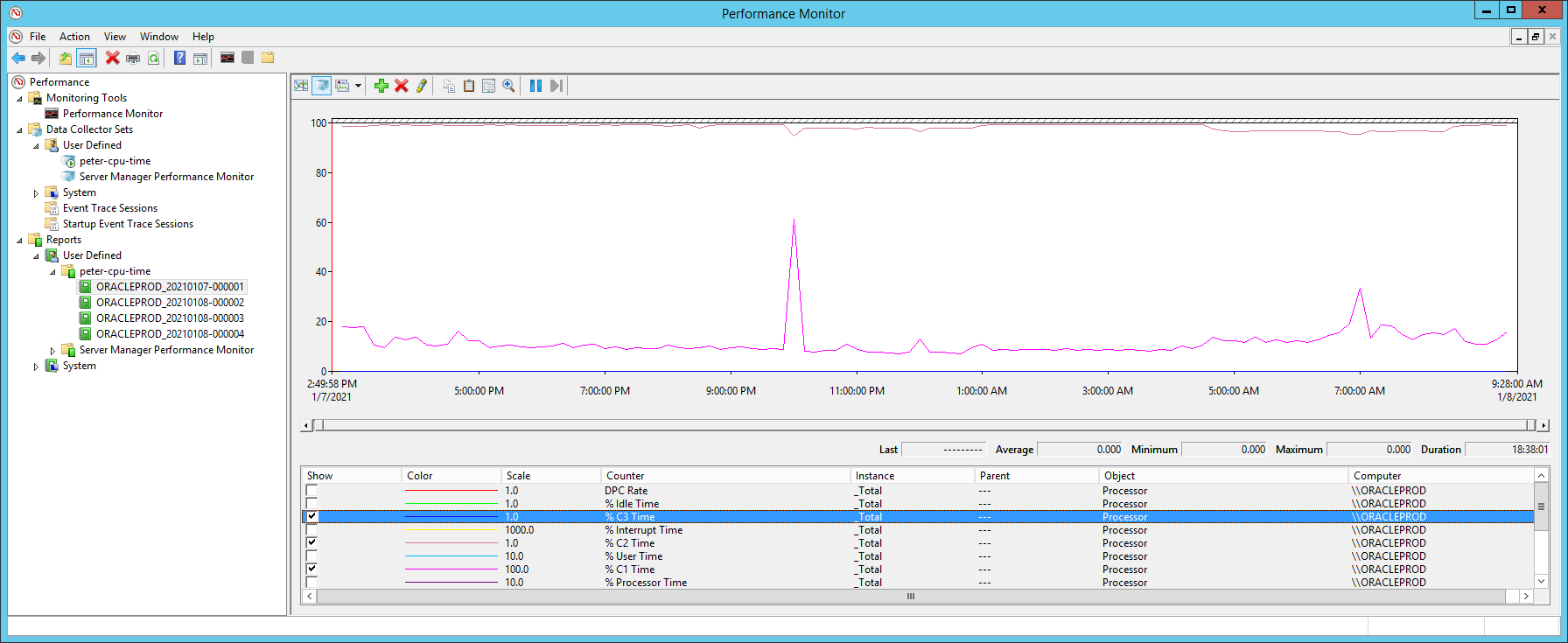
**% Processor Time**

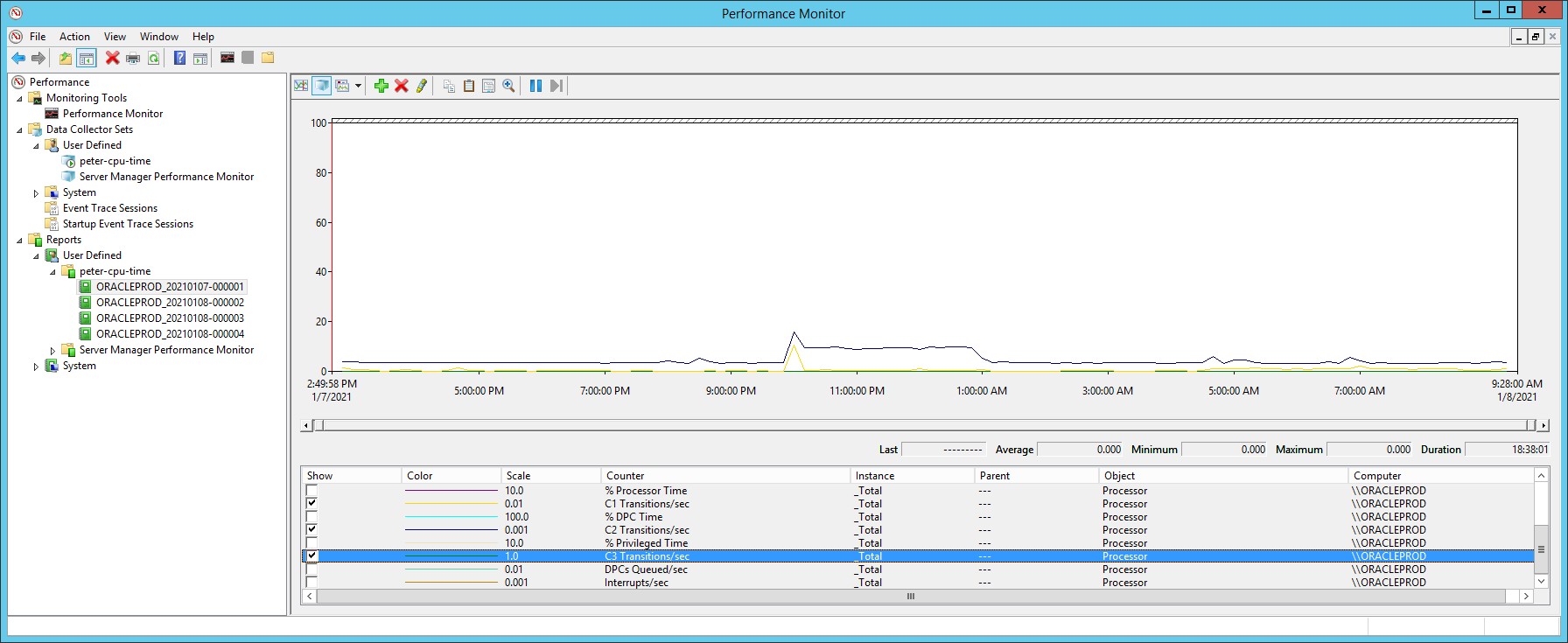


**%IDLE time:**



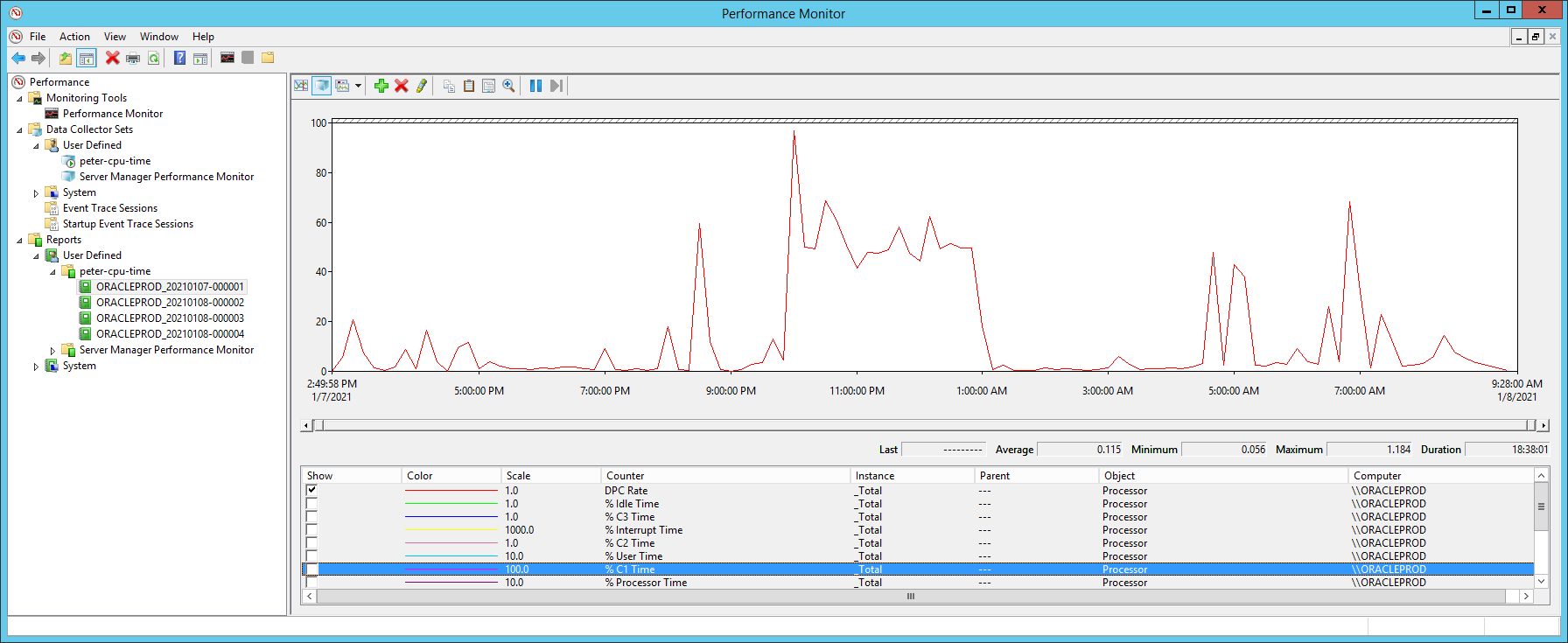
**%C1 time, %C2 time, %C3 time:**

  
The first three are %C1, %C2, and %C3 Time. The CPU can operate in any one of these power performance states (C1, C2, or C3). **C0 is normal power state**; and C1, C2, C3 are the low power states of operation. For example, when a system goes into sleep mode, the operating system will tell the CPU to go into the C3 low power operation state mode (lowest power state). The CPU will then reduce the voltage it's using in terms of power consumption. Although C3 is the lowest state, the other two states are often used by the CPU as well, based on what the operating system thinks it should be running at, relative to how much latency there is. Not all systems support these states, but most new servers are likely to. These counters tell the percentage of time the processor has spent in each state.

C1 Transitions/sec, C2 transitions/sec, C3 transitions/sec  


The second three counters are C1, C2, and C3 Transitions/sec. These counters indicate the rate at which the CPU enters any one of these states. Both sets of counters become important when monitoring the strain on your server's processor(s).

**DPC-rate:** % DPC Time is the percentage of time that the processor spends receiving and servicing deferred procedure calls (DPCs). DPCs are interrupts that are run at a lower priority than standard interrupts. If a high % DPC Time is sustained there may be a processor bottleneck or an application or hardware related issue that can significantly diminish overall system performance.



A high % DPC Time value can be caused by one or more of the following:

* Processor bottleneck
* Software-related problem
* Hardware or device driver related problem

To determine the root cause of a high DPC, rate follow the process outlined below.  
Observe the proportion of the processor time that is spent servicing interrupts and DPCs.  
From Azure Monitor, use a monitoring chart to view the performance counters (metrics) over time:

* Processor Information\% Processor Time
* Processor Information\% Interrupt Time
* Processor Information\% DPC Time

Compare the values of the % Interrupt Time and % DPC Time counters to Processor Information\% Processor Time for each processor instance.

If a processor instance is running a sustained % Processor Time that is **greater than 85%** and it is also spending greater than 15% of that time servicing Interrupts and/or DPCs, the processor is probably the source of a performance bottleneck. This bottleneck can be addressed by upgrading or adding additional processors to the virtual machine.

If the processor is running a sustained % Processor Time **of less than 85%** and it is also spending less than 15% of that time servicing interrupts and/or DPCs, the performance issue may be the result of either an application or hardware related issue.

Where a hardware device is the root cause, an administrator will find that the % DPC Time has probably increased substantially over a short period time. This will often occur when new hardware is installed or drivers have been upgraded. If the administrator can isolate the issue to a hardware/device driver issue, it can be addressed by working with the vendor.

In cases where you are administering a multiprocessor system that does not distribute interrupts symmetrically, you can often improve the distribution of the processor workload by adding network adapters so that there is one adapter for every processor. Generally, you only add adapters when you need to improve the throughput of your system. Network adapters, like any additional hardware, have some intrinsic overhead. However, if one of the processors is nearly always active (that is, if Processor Information: % Processor Time = 100) and more than half of its time is spent servicing DPCs (if Processor Information: % DPC Time > 50), then adding an adapter is likely to improve system performance, as long as the available network bandwidth is not already saturated.

**Where an application is the root cause**, you will find that DPCs are probably being blocked by an application that has issued a call that is taking a substantial amount of time to complete. During this time DPCs are blocked and will be queued. To determine which application is the root cause, you must run advanced performance, tracing, and diagnostics to pin point the exact application that is responsible for the performance issue.

**How to: View Performance Counter Data for a Data Collector Set**

When you run a Data Collector Set, the data that is collected for performance counters is stored to a log file (.blg) in the location that was defined when the Data Collector Set was created. In Windows Performance Monitor, you can view log files to see a visual representation of performance counter data.

**Before viewing/analysing the LOG-file [STOP] the data-collector vanuit PERFMON !!!**

**To view the Data Collector Set log file**

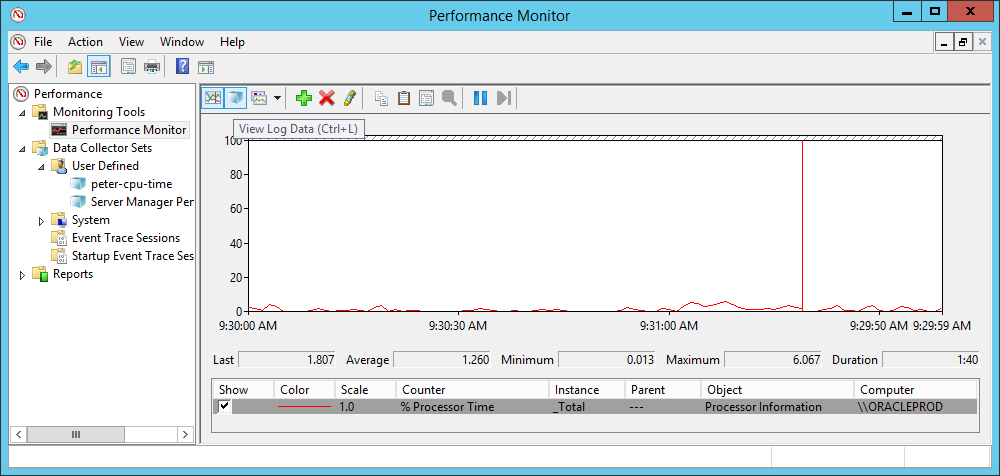
1. Start Windows Performance Monitor.
   * Choose **Start**, in the **Search** box, type **perfmon**, and then choose the related link.
2. In the navigation pane, expand **Monitoring Tools**, and then choose **Performance Monitor**.
3. In the console pane toolbar, choose the **View Log Data** button.

The **Performance Monitor Properties** page opens at the **Source** tab.

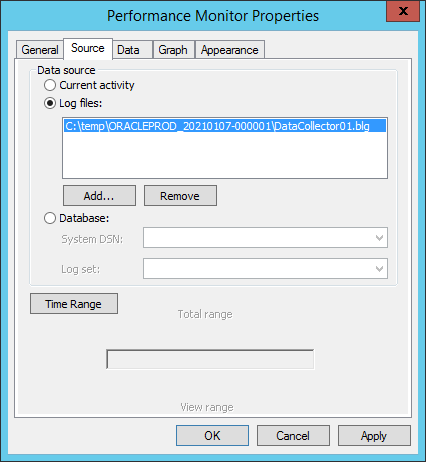
1. In the **Data Source** section, select **Log files**, and then choose the **Add** button.
2. Browse to the log file that you want to view, and then choose the **Open** button.
3. Choose the **OK** button.

**BEKIJK RESULTATEN**

Ik heb de CPU-counter nu bijna 16 uur laten lopen, er is file gegenereerd van ruim 8MB, valt dus alles mee.  
Open nu de PERFORMANCE-MONITOR, en klik op ICON op toolbar [VIEW LOG-FILE DATA] en kies voor de log-file



En open log-file



Hmm, daar ik krijg NIET de inhoud van log-file te zien, zoals bij het REPORT wel te zien krijg.