bitsum technologies

Process Lasso

The ultimate Windows automated process optimizer!

PDF COMPILED BY: TINFOIL HAT / DRAGONS FILES

Process Lasso

Overview

Process Lasso is an automated Windows process (program) management and optimization utility. By managing the programs running on your computer, Process Lasso increases system responsiveness and helps to prevents system stalls. It accomplishes this through our ProBalance technology. This proprietary algorithm dynamically adjusts the priorities of running programs so that some are given higher priority access to the CPU(s) than others. With Process Lasso, no longer will single, or multiple, processes be able to bring your system to a virtual stall. Process Lasso will let you keep interacting with your computer, even when it is totally swamped.

In addition to the ProBalance technology, users can tweak how their programs are run. Process Lasso offers a number of unique and helpful capabilities. These include **default priority classes**, **default CPU affinities**, **disallowed processes**, and **process instance count limits**. By 'default' we mean each time a program runs, it will run at the priority class and/or CPU affinity of your choice. These features give you the ability to control how programs utilize your computer's resources. With Process Lasso, you can decide which programs are the most important, and configure them accordingly.

Process Lasso uses virtually no system resources itself, being written in highly optimized native x32 and x64 C++. The core engine can run stand-alone and consume as little as 1-3MB of RAM.

The Parts of Process Lasso

1. About ProBalance
    1. How is my CPU shared?
    2. How does ProBalance work?
    3. Proof of concept
2. About CPU Throttling
3. About Gaming Mode
4. Using the Graphical User Interface (GUI)
    1. The Main Window
    2. Graph
        1. Per-process CPU history
    3. All Processes Tab
        1. Select what process information to show
        2. Rules column meaning
        3. Single selection process context menu
        4. Multiple selection process context menu
    4. Active Processes Tab
        1. Select what process information to show
        2. Double clicking a process
        3. Single selection process context menu
        4. Multiple selection process context menu
    5. Configuration Dialogs
        1. ProBalance Options
        2. Default CPU Affinities
        3. Default Priorities
        4. Auto-Terminate List
        5. Instance count limits
        6. Keep running processes
        7. Gaming processes
        8. Anti-sleep processes
        9. High Performance Power Scheme processes
    6. Main Menu Options
5. Using The Core Engine
    1. Overview
    2. Running as a Service
6. Using the INI Configuration File
7. Command Line Arguments
8. Advanced Tools
    1. TweakScheduler
    2. Vista Multimedia Scheduler Configuration Tool
9. Enterprise Deployment
    1. Deployment methods
    2. Service mode
    3. Silent installation
10. Frequently Asked Questions
11. Credits
12. Licensing

The Parts of Process Lasso

Process Lasso consists of a core engine and a user interface. The two are separated so that users can completely close the user interface while keeping Process Lasso active, managing processes. The core engine is run as ProcessGovernor.exe, while the user interface is ProcessLasso.exe. Running only the core engine results in memory usage of a mere 1-3MB (typically) and almost no CPU use.

When the user interface, ProcessLasso.exe, is launched it automatically launches an instance of the ProcessGovernor.exe core engine if one isn't already running for the current user. When you close the user interface, you are asked whether you want to keep the core engine running or not.

Each user session is meant to have their own instance of ProcessLasso.exe and ProcessGovernor.exe. This is so that they can have individual configurations and process rule sets, and so that each instance is limited to its own processes.

Component	Filename	Description
Process Lasso Graphical User Interface	ProcessLasso.exe	This the graphical user interface that serves the purposes of configuring the product, manually managing processes, and displaying current activity by Process Lasso in real time. It also provides a graph to view current CPU usage and system responsiveness. This component is optional and does not need to be installed or running for proper operation of Process Lasso's core engine (Process Governor). For usage documentation, **see section Using the Graphical User Interface**.
Process Governor Core Engine	ProcessGovernor.exe	This is the core engine that applies process management rules. **This can run by itself for minimal resource consumption.**
Installer for Process Governor	InstallHelper.exe	This utility allows you to configure how the Process Governor starts at system boot or user login.
Configuration File	ProLasso.ini	This configuration file holds the process rules and options for Process Lasso. It can actually be named anything, and be placed anywhere, including remote shares. You can change its location using the Process Lasso Management Console, supply command line parameters, or manually editing the registry (HKCU).
Log File	ProLasso.log	This log file keeps a record of recent actions. You can disable it to save a little more system resources. Its size is automatically trimmed periodically in accordance with the logging configuration in the INI file. The log can be stored anywhere, including remote shares. You can change its location using the Process Lasso Management Console, supply command line parameters, or manually editing the registry (HKCU).

Process Lasso's Process Balance Technology (ProBalance)

How is a CPU shared amongst all running processes?

A process has a collection of threads (essentially running tasks). This is what the operating system sees, and executes. It basically keeps a long list of threads to which it dedicates the CPU(s) for brief spurts of execution (time slices).

A CPU can only do one thing at a time. Therefore, to multitask, the CPU is quickly switched between all active threads on the system. It switches between threads so quickly that it appears multiple processes are running at the exact same time, when in fact they are just taking turns sharing the CPU (running concurrently). For multi-CPU (multi core) systems, this is still true, except that there are more CPUs available for thread execution, thus allowing some true simultaneous execution instead of just concurrent execution.

When the threads of multiple processes need to use the CPU at the same time, the priority level of each thread is used to help determine which should get the CPU next (which is most important). By temporarily lowering the priority class of overly active background processes, your system is kept responsive even in the face of high CPU loads. Lowering the priority class of a process lowers the priority of all threads in that process. When acting on a process, you are essentially changing the base priority from which the individual priorities of all threads of that process are derived from. In this way, the relative priorities between threads in a process are kept the same.

What is the problem?

Windows has a particularly bad problem dealing with threads that decide they want to consume every bit of CPU time they can get their hands on (a CPU bound thread). A single 'CPU bound' thread running at Normal priority can bring an entire single-CPU system to a stall, as demonstrated by our graphical demo below. Yes, it is true - believe it or not! It is this worst case scenario that Process Lasso was originally written to address. By temporarily lowering the priority of the offending process, your PC can be saved from a virtual stall. It was later discovered that, with a few refinements, our algorithm could improveme system responsiveness during periods of high CPU loads - in addition to saving the PC from a worst case scenario.

Some schedulers, such as the default one of the Linux kernel, penalize CPU bound (CPU hungry) threads while rewarding I/O bound threads. This is similar to what ProBalance effectively does. You see, on Windows systems *most* threads are I/O bound. They give up their time slices pre-maturely as they enter a wait state for some type of I/O. Sadly, CPU bound (CPU hungry) threads also exist from time to time, and do a real number of the less greedy I/O bound threads.

This does NOT mean you should go and reprioitize all your running processes, ranking them in importance to you. Let our carefully designed ProBalance algorithm do that automatically. Do NOT make manual priority adjustments unless you know what you are doing. It can cause more harm than good. We have written our software to operate safely and effectively in its default configuration, but can't say the same for custom tweaks made by the user.

Why doesn't foreground boosting work?

First, for those who don't know, the foreground process is the process that owns the window that has keyboard and mouse focus. This means there is only ever one foreground process at a time. It is essentially the program you are currently working with. Some programs of lesser quality believe boosting the priority of the active foreground process is effective at increasing performance or responsiveness. The truth is that it does neither and more likely to do the opposite.

Windows already applies foreground boosting, and gives the foreground thread longer time slices. Further increasing the priority of the foreground process and/or the specific foreground thread is not effective at all. Remember, giving a process a higher priority does not mean it will run faster. It simply means if several processes are active at once, it will have a higher precedence. However, if any single process has too high a precedence over other processes, as foreground boosting would result in, complications can occur. In fact, our research has shown that additional foreground boosting can cause a number of complications with OS components and various applications.

Be very wary of any utility that claims to boost PC performance by increasing the priority of the foreground process. These utilities are extremely easy to create, and yet another gimmicky thing people are trying to push. In some cases, the author of the program simply doesn't understand the CPU scheduler and why this is a bad idea. In other cases, companies don't care and are just trying to make a profit. In short, it is a very wrong solution and should be avoided at all costs.

What is Process Lasso's 'ProBalance' technology?

ProBalance, which stands for Process Balance, is the name of our proprietary algorithm that temporarily adjusts the priority of running processes in an effort to keep your system responsive during high loads.

When there is a high load on the system CPU(s), the responsiveness of the PC can be severely impacted. On a single processor PC, it takes only one CPU hungry thread running at normal priority to bring the entire system to stall. Even high priority threads can get starved of CPU cycles by a CPU hungry thread running at normal priority. This problem is inherent to the design of the Windows CPU scheduler and general OS architecture. Sure, it seems like such a problem should NOT exist -- but it does, and its easily demonstrated (see 'skeptics' section below).

The casual user will often find their PC responsiveness diminished for brief periods of time (a micro lag). Sometimes this is due to an error occurring in a background process, and other times it is simply because there is such a high load on the CPU. You probably have experienced this before - mouse movement gets jerky and every action is terribly slow.

Process Lasso's ProBalance intelligently lowers the priority class of background processes when they may be interfering with system responsiveness. This doesn't hurt the performance of background processes since they still get a considerable amount of the available CPU time, but it helps the responsiveness of foreground processes tremendously. After all, usually it just takes a few CPU cycles to keep the foreground process responsive. Taking these from background processes, when necessary, is hardly detrimental to them. Also, the adjustment is temporary, so its undone as soon as system conditions change. Process Lasso is designed to be minimally obtrusive, lowering priorities only when appropriate, and making sure that the background processes still perform just fine.

Is ProBalance useful on multicore PCs?

Yes, it is. Multicore PCs are subject to the same sort of problem, except that it will take a greater number of badly behaved threads to affect system responsiveness. Our CPU Eater will demonstrate the problem even on multicore PCs. Of course, Process Lasso also offers an assortment of features other than ProBalance that are useful to multicore PC owners.

Should I reprioitize all my processes, ranking them in importance to me?

Absolutely **NOT**. Some programs out there encourage you to do just that, but it is dangerous. The authors of these programs are probably not aware of the dangers, or otherwise don't care. While Process Lasso supports setting default (sticky) priorities, it is only meant to be used under certain appropriate circumstances. If you aren't an expert, it is best to let Process Lasso's carefully designed

ProBalance algorithm dynamically adjust process priorities as necessary.

Proof of Concept

With all the various scams out there, such as RAM defragmenters, it is natural for some users to be skeptical. The most common argument comes from those who have overconfidence in the performance of the Windows CPU scheduler. These people believe that *if* the Windows scheduler is written as it should be, then surely normal priority threads can't substantially interfere with the performance of other normal (and above) priority threads. This argument is echoed particularly in non-Windows crowds who haven't experienced the real-world performance of the Windows CPU scheduler.

The truth is that, on a single CPU system, a single thread of normal priority can bring the entire PC to a near freeze! Even high priority threads can get starved of CPU time by a CPU hungry thread of normal priority. Yes, amazing but true. For multi-CPU systems, the number of threads required to bring the system to a complete stall is equal to the number of CPUs available. Don't believe us? Try our demonstration app, or write your own. Any thread that does nothing but eat up CPU cycles without restraint will demonstrate this phenomenon.

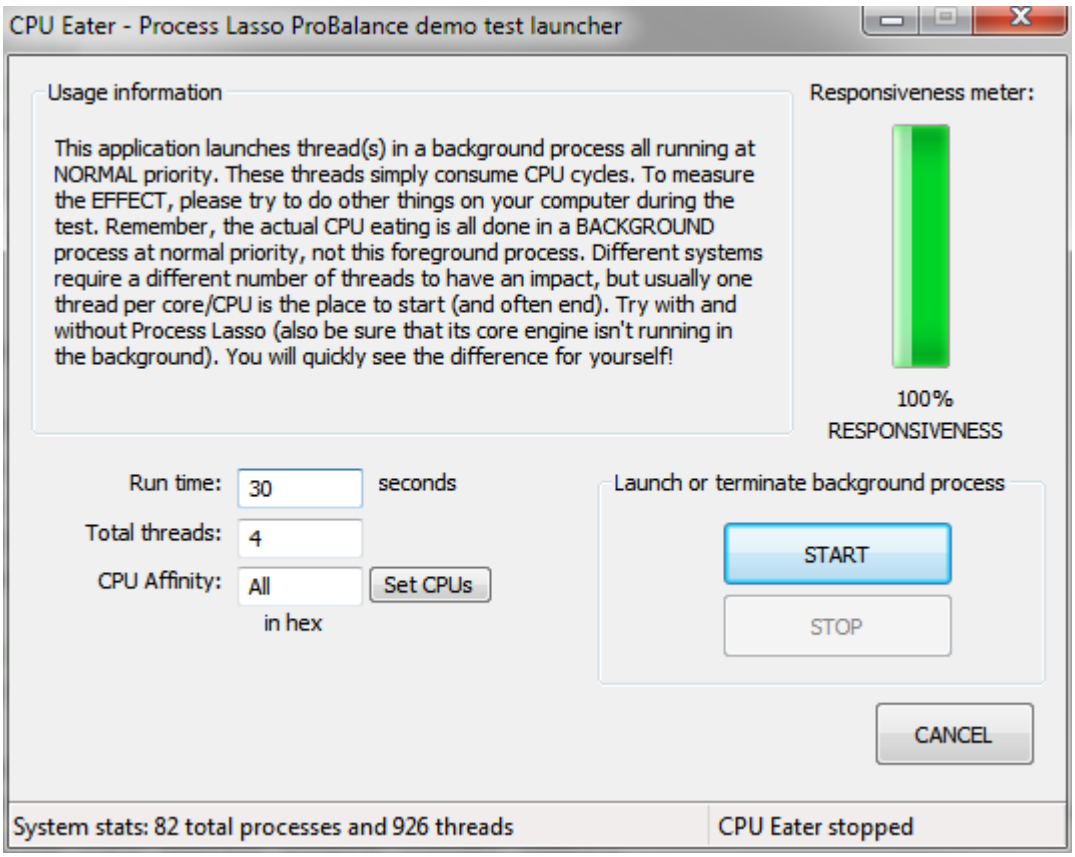
Demonstration

We have developed a 'CPU eater' that you can test Process Lasso with. This CPU Eater creates a handful of threads all running at NORMAL priority. These threads simply use up CPU cycles. Now, ***if*** the Windows CPU scheduler behaved like you might expect, other running processes would get enough CPU cycles to remain responsive -- even while this CPU Eater is running in the background. Unfortunately, this isn't the result. Your computer's responsiveness will substantially degrade while the CPU Eater is running! Now, see for yourself ...

If you wish, you can create your own 'CPU eater', or test similar utilities instead of ours. Some CPU Eaters out there raise their priority, because they don't realize that it only takes normal priority to 'do the job'. For those, be sure to turn OFF the option 'Options / ProBalance Settings / Exclude non-normal priority'. Normally, Process Lasso leaves high priority processes alone because it assumes they have raised their priority for a good reason.

This demo defaults to being 'heavy handed', but feel free to lower the thread count. It usually takes far,far fewer to induce an effect than is the default value! We are working on better (most conservative) default demo values for all hardware, and will include that in the next update.

Also REMEMBER, the threads created here are background threads. Out of control *foreground* threads cause ever more harm since they get considerably longer time slices (depending on scheduler config or OS edition).



- Download 32-bit CPU Eater
  (http://www.bitsum.com/files/CPUEaterDist32.exe)
- Download 64-bit CPU Eater (for Windows x64 OS only)

  (http://www.bitsum.com/files/CPUEaterDist64.exe)

Usage instructions:

1. Download and run the above link appropriate to your operating system (32-bit or 64-bit)
2. It will automatically start the CPU Eater control application for you
3. Press the START button to initiate the test in a background process running at NORMAL priority
4. If you PC seems to freeze because Process Lasso's ProBalance is not active, WAIT and everything will return to normal after the test terminates in a minute (or however long you specified). We also temporarily increase the priority of the control application so that you will hopefully be able to press the STOP button to terminate the background process that is hogging CPU cycles. Of course, if Process Lasso is running and ProBalance enabled then you won't have this problem.

Remember, if you don't install Process Lasso the test will simply show the deficiencies in the Windows CPU scheduler. You must install Process Lasso to see the benefits of ProBalance.

After installing Process Lasso, turn OFF ProBalance (or close ProcessLasso AND its core engine ProcessGovernor). Then start the CPU Eater to see how your computer behaves with ProBalance OFF. After completion of this first test, turn ProBalance back ON or restart Process Lasso. Now run the same test again. You will be amazed at the difference, no matter how many cores/CPUs your computer has!

Remember, the threads of the background process the CPU Eater launches are all running at NORMAL priority, and they aren't doing anything besides run in pointless infinite loops. No

adjustments are made, we just let Windows handle it. Any application running on your computer has the potential to cause the same effect as our CPU Eater did.

WARNING: As a precaution, we recommend saving all open documents before doing this test. Also, be sure not to reset your computer when your are testing the CPU eater without ProBalance. Your computer will start responding better again in 60 seconds. You agree to accept all risks in running the above test. There are no extraordinary risks, but your system is put to a heavy load, which can cause problems on systems that can not adequately handle heat dissipation. In general, perform this test at your own risk. We must say this, and you must agree if you plan to try this test.

The above simple test takes 2 minutes. Give it a try!

How our test works

A demonstration of the deficiencies in the Windows scheduler is easy. First, for the sake of simplicity, we'll start describing a system that has a single CPU. The demonstration is easily scaled to multiple cores, but it is easier to describe the concept for a single CPU system.

First, create a process that does nothing but run in an infinite loop, thereby consuming all CPU cycles available to it. We'll call this a 'CPU eater'. Now, launch this process as any other application, at a normal priority class. If the Windows CPU scheduler works as one would expect it to, the performance of other processes on the system, and the foreground process, shouldn't be severely impacted. Unfortunately, you'll find that this single process, running at normal priority, actually so starves other processes of CPU cycles that the system grinds to a near halt -- even termination of the offending process is very difficult. This is true even with the Vista CPU scheduler, despite its improvements.

To scale this argument up to multiple CPUs, simply allow the 'CPU eater' process to have multiple threads, or launch more instances of the CPU eater process. The same effect will be seen.

Manual tweaks can help

With Process Lasso's other great features, like default process priorities and affinities, you can really fine tune how your system performs. ProBalance is a great way to automatically improve your system's responsiveness, but if you desire even more tweaking, Process Lasso also gives you that power. No automated algorithm can replace intelligently chosen human tweaks.

What's left after uninstall?

Say a person does try Process Lasso, but then uninstalls it. What's left behind? **Nothing**. No permanent system changes were made, and no files remain behind. Just like you, we don't like 'junk' left behind after an uninstall, and we especially don't like software that makes permanent system configuration changes that aren't undone on uninstall. As a side note, Process Lasso makes NO adjustments to your system configuration, it simply manages processes ONLY when its core engine is running. The advanced tools included with Process Lasso, TweaskScheduler and the Multimedia Scheduler Configuration Utility DO make permanent changes, but backups are created first to ensure you can revert to your previous state.

A STRONG WARNING AND EULA

Process Lasso was created with one primary rule 'Do no harm'. Therefore, its ProBalance algorithm only makes changes believed to be entirely safe. Further, when a change is needed, it makes the most minimal one possible. We look at this as a conservative approach. We've seen other software not be so kind, believe it or not. Beware of gimmicks out there that boost priorities carelessly. Many of these gimmicks boost the foreground process priority to High, which is not only unhelpful, but is

also unsafe. Try those utilities with our demo - or write your own CPU Eater if you don't trust ours, it couldn't be more simple to create. In C, a 'while(1);' (infinite loop) is all it takes...

When you go to make manual changes to certain system processes, or enable certain features of Process Lasso, warnings are shown that inform you of possible risks. However, we can't possibly account for every conceivable user induced error and/or software environment, so its possible custom modifications to Process Lasso's configuration could cause unforeseen negative consequences. We furthermore can't warn of all possible risks, so we simply recommend NOT changing the default settings unless you are an expert. Carelessly changing the priority class of processes is potentially determinantal to the stability of your PC. We recommend you just let Process Lasso's ProBalance algorithm do its job, unless you know what you are doing.

Since Process Lasso allows such advanced configuration tweaks, we must issue a legal warning at install (a EULA). As always, you assume all risk and liability for any damages, tangible or intangible, resulting from the use or misuse of our software. If you do not agree with that, then we can not allow use of our software. The full EULA is presented prior to install. If you can not read AND agree to it, then you must cancel the install.

Third-party research and links

The issues that Process Lasso's ProBalance addresses are well documented throughout the internet. Since we may be perceived to have a bias (though we are biased only towards the truth), we decided to publish some of this third party research. Here are some quick links, and we'll add more soon.

- Everything you never wanted to know about OS multi-tasking (and why out of control processes need priority adjustment)

  (http://www.tmurgent.com/WhitePapers%5CSchedulingWP.pdf)

  > "I should also note that Windows has other scheduling complications, such as "foreground" and "background" tasks [...] However, the bottom line, in my humble opinion, is that the operating system is more concerned in ensuring that a task that is not ready to run does not, than making sure all tasks get a fair shake. [...] We can, however, detect thread and process CPU utilization, and help the operating system to adjust the prioritizations."
  > -(c) Tim Murgent of TMergent Technologies

- Hardware Virtualization: the Nuts and Bolts

  (http://it.anandtech.com/IT/showdoc.aspx?i=3263)

  > "One process that takes up 100% of the CPU time may slow the other applications to snail speed for example, despite the fact that modern OSes use preemptive multitasking."
  > -(c) Johan De Gelas, AnandTech

- Quote from Inside the Linux scheduler (IBM) - an example of another OS scheduler's attempt to address this

  (http://www.ibm.com/developerworks/linux/library/l-scheduler/)

  > To prevent tasks from hogging the CPU and thus starving other tasks that need CPU access, the Linux 2.6 scheduler can dynamically alter a task's priority. It does so by penalizing tasks that are bound to a CPU and rewarding tasks that are I/O bound. I/O-bound tasks commonly use the CPU to set up an I/O and then sleep awaiting the completion of the I/O. This type of behavior gives other tasks access to the CPU.

CPU Throttling

CPU Throttling is VERY DIFFERENT from ProBalance dynamic priority adjustments. CPU Throttling applies a forcible limit on the number of CPU cycles a process can consume in a given period. This actually slows the process down. It is only useful in very specific situations, and should not be used carelessly. Most users should instead rely only on ProBalance to help improve system responsiveness. CPU Throtting is a feature intended for advanced users who know what they are doing. There was some hesitation in making this feature available due to fear of misuse. Please, use it with caution!

Generally speaking, simply temporarily lowering the priority of an out-of-control background process is sufficient to improve system responsiveness (as ProBalance does).

Gaming Mode

Process Lasso offers a proprietary 'Gaming mode' algorithm which SAFELY and EFFECTIVELY manages process priorities in a way optimal for full screen games. Furthermore, it temporarily places your PC in the 'High Performance' power scheme.

When in Gaming Mode, Process Lasso gives the foreground application an even higher precedence over background applications. Remember, further foreground boosting isn't very helpful, so don't try to run your game with a priority higher than what Process Lasso sets it to. Process Lasso has been extensively researched and tested, so its changes are much safer than your own custom tweaks. Also remember that Windows already does foreground boosting of its own. We say this to discourage readers from simply setting their game to a 'High' priority. Doing that would not be helpful, and could cause problems.

The end effect is that your game operates with all the resources your PC has at its disposal. In addition, with ProBalance also continuing to restrain unruly background processes, micro-lags are reduced or eliminated completely.

Using the Graphical User Interface

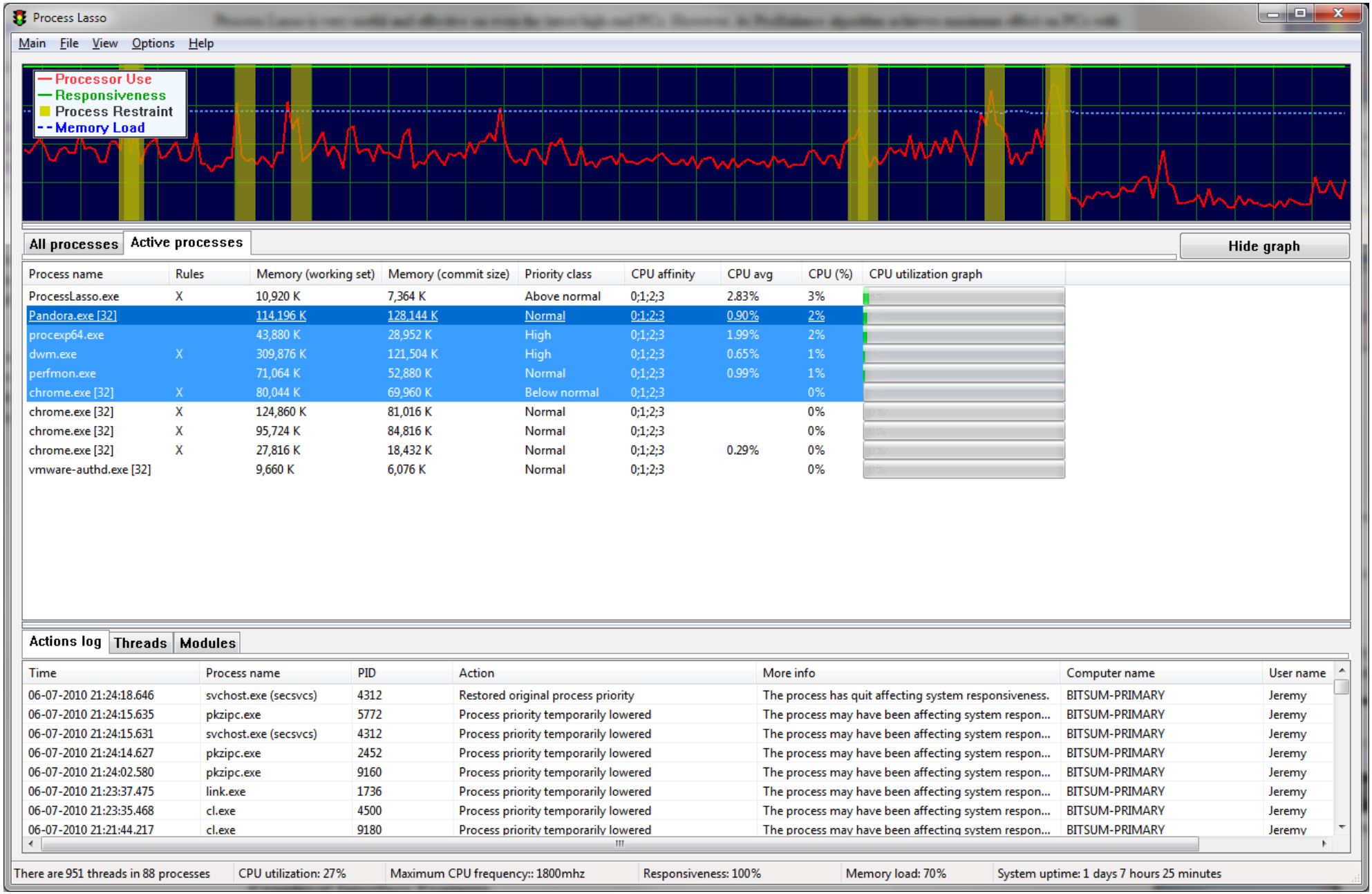
The Process Lasso Graphical User Interface is an easy to use application that allows the user to configure the rules governing running processes, view actions taken, list and manipulate running processes, and watch CPU usage, system responsiveness, and memory load on a graph. When run, its main window will be displayed and a notification icon that looks like a traffic light will be placed in the system tray area. Closing the main window will minimize it to the system tray. The main window can be reopened by clicking the system tray icon.

Remember, the Graphical User Interface (GUI) of Process Lasso does NOT need to be running for process rules to be enforced. That said, it consumes very little resources, especially when minimized to the system tray. However, you can completely close the GUI and simply keep the core engine (processgovernor.exe) running silently in the background to get all the benefits of Process Lasso with absolutely minimal overhead. When you exit the Process Lasso GUI, you will be asked if you would like to keep the core engine (processgovernor.exe) running or not. When configuration changes or status updates are wanted, you can then start the GUI from the system Start Menu.

During the install process, you can decide if you want to start the GUI and/or core engine for all users at login. Many times, people prefer to simply start the core engine for all users, getting the benefit of ProBalance without any additional overhead. Of course, you may then forget just how much benefit Process Lasso is doing you, but you'll be reminded if you ever uninstall the product ;). Again, the GUI really doesn't consume many resources at all, especially when minimized, so its recommended to let them both start up, except in extreme cases where RAM is really limited.

Main Window

The main window consists of a CPU utilization and system responsiveness graph, a list of running processes, and a list of recent actions (log). The configuration of Process Lasso is made easy to tweak through the menu system. In the main menu you'll find all the general configuration options. Right-clicking on a process, or multiple processes, shows available operations on those process(es).


Process Lasso
Main File View Options Help
Processor Use
Responsiveness
Process Restraint
Memory Load
All processes
Active processes
Hide graph
Process name | Rules | Memory (working set) | Memory (commit size) | Priority class | CPU affinity | CPU avg | CPU (%) | CPU utilization graph
ProcessLasso.exe | X | 10,920 K | 7,364 K | Above normal | 0;1;2;3 | 2.83% | 3%
Pandora.exe [32] | | 114,196 K | 128,144 K | Normal | 0;1;2;3 | 0.90% | 2%
procexp64.exe | | 43,880 K | 28,952 K | High | 0;1;2;3 | 1.99% | 2%
dwm.exe | X | 309,876 K | 121,504 K | High | 0;1;2;3 | 0.65% | 1%
perfmon.exe | | 71,064 K | 52,880 K | Normal | 0;1;2;3 | 0.99% | 1%
chrome.exe [32] | X | 80,044 K | 69,960 K | Below normal | 0;1;2;3 | | 0%
chrome.exe [32] | X | 124,860 K | 81,016 K | Normal | 0;1;2;3 | | 0%
chrome.exe [32] | X | 95,724 K | 84,816 K | Normal | 0;1;2;3 | | 0%
chrome.exe [32] | X | 27,816 K | 18,432 K | Normal | 0;1;2;3 | 0.29% | 0%
vmware-authd.exe [32] | | 9,660 K | 6,076 K | Normal | 0;1;2;3 | | 0%
Actions log
Threads
Modules
Time | Process name | PID | Action | More info | Computer name | User name
06-07-2010 21:24:18.646 | svchost.exe (secsvcs) | 4312 | Restored original process priority | The process has quit affecting system responsiveness. | BITSUM-PRIMARY | Jeremy
06-07-2010 21:24:15.635 | pkzipc.exe | 5772 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
06-07-2010 21:24:15.631 | svchost.exe (secsvcs) | 4312 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
06-07-2010 21:24:14.627 | pkzipc.exe | 2452 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
06-07-2010 21:24:02.580 | pkzipc.exe | 9160 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
06-07-2010 21:23:37.475 | link.exe | 1736 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
06-07-2010 21:23:35.468 | cl.exe | 4500 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
06-07-2010 21:21:44.217 | cl.exe | 9180 | Process priority temporarily lowered | The process may have been affecting system respon... | BITSUM-PRIMARY | Jeremy
There are 951 threads in 88 processes
CPU utilization: 27%
Maximum CPU frequency:: 1800mhz
Responsiveness: 100%
Memory load: 70%
System uptime: 1 days 7 hours 25 minutes

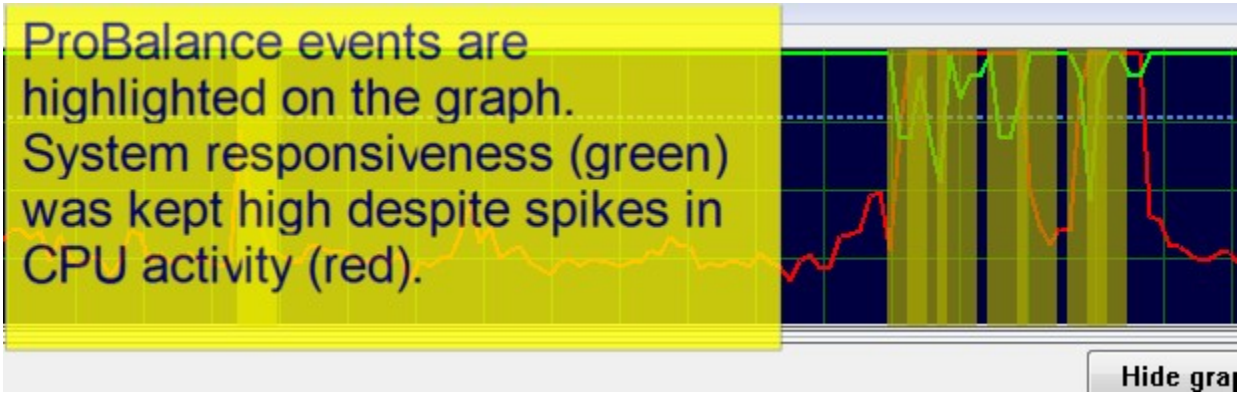
The Graph

The Process Lasso graph shows a calculation of system responsiveness and overall CPU utilization. The system responsiveness is calculated using a proprietary algorithm that measures the latency in the user interface (windowing) subsystem.

Highlighted portions of the graph indicate that an out-of-control process restraint occurred during that period. This can help you see the impact of Process Lasso on system responsiveness. Note that the highlighted areas of the graph may not be 100% accurate in their timing, but are somewhere close. As future versions come the accuracy of the highlighted area will improve.

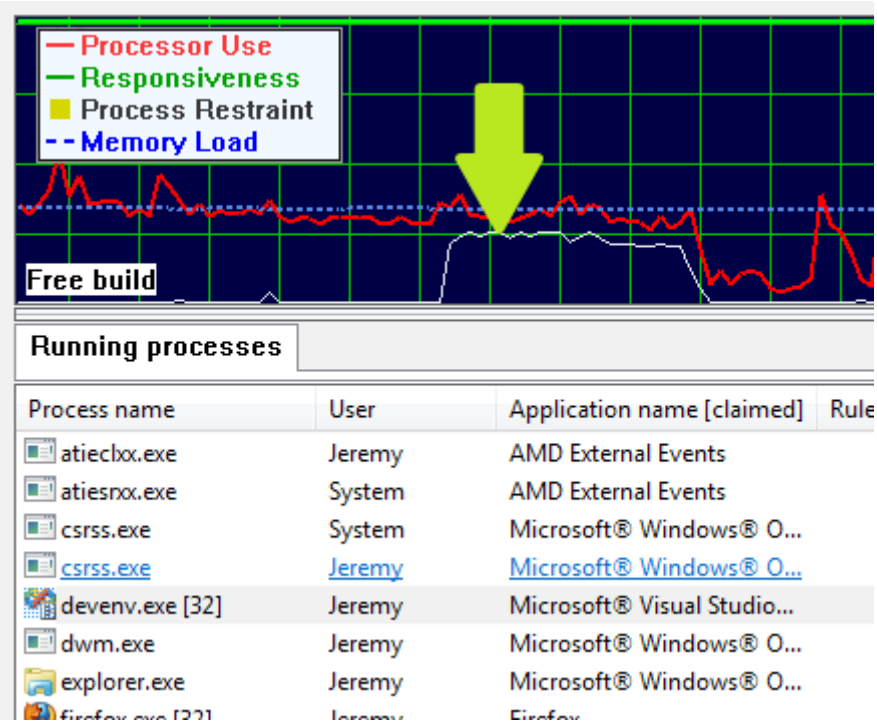
Highlighted ProBalance events

During periods when ProBalance takes action to ensure your system repsonsiveness remains high despite a spike in CPU use, the graph will be highlighted. The log can be referenced to see what actions were taken.



CPU history of selected process(es)

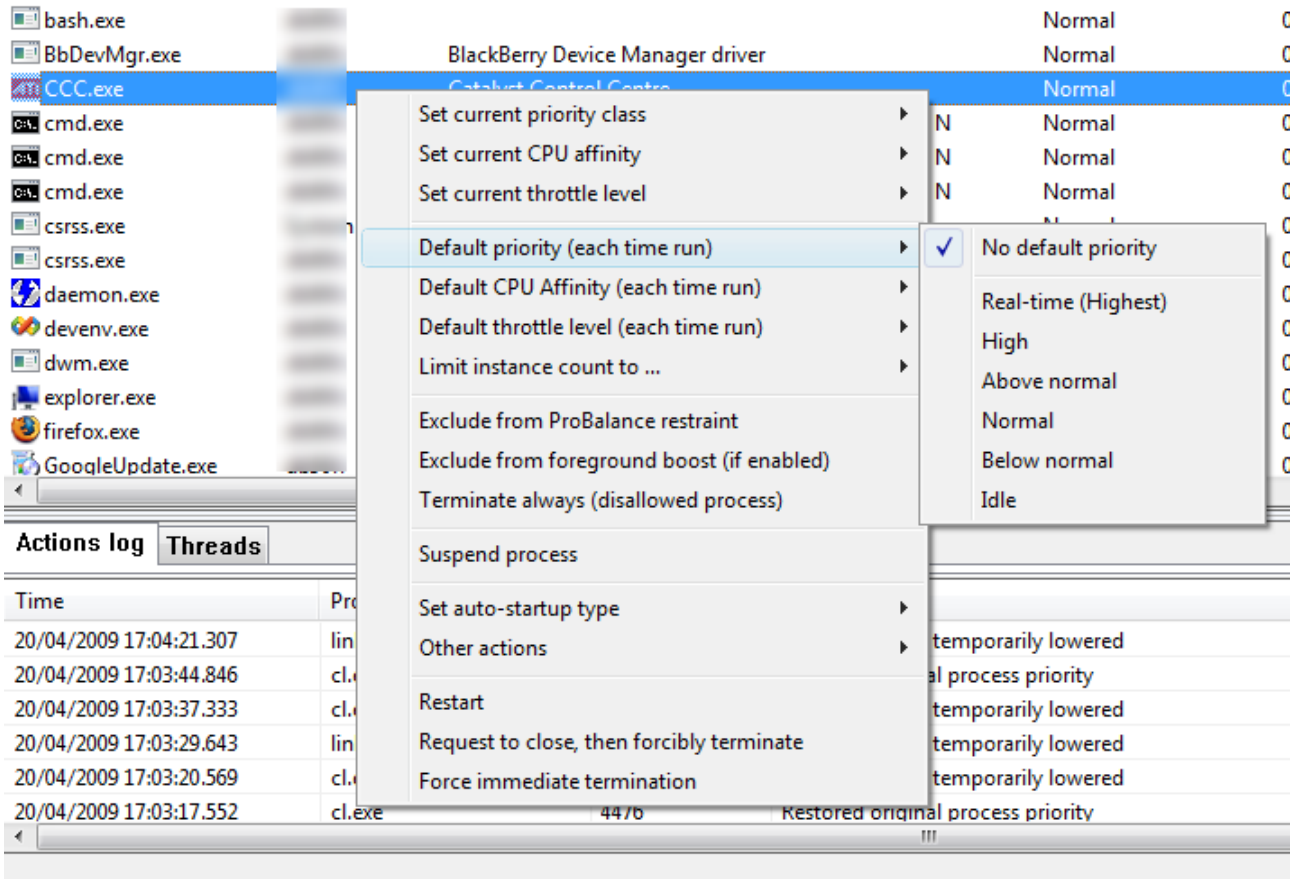
With Process Lasso you can see the individual CPU history of process(es) you select in the process list. The CPU utilization history of all processes currently selected (highlighted) in the process list is drawn as a smaller white line on the graph. For example, the user below selected 'devenv.exe' and its CPU history was drawn onto the graph as a white line.



The All Processes Tab

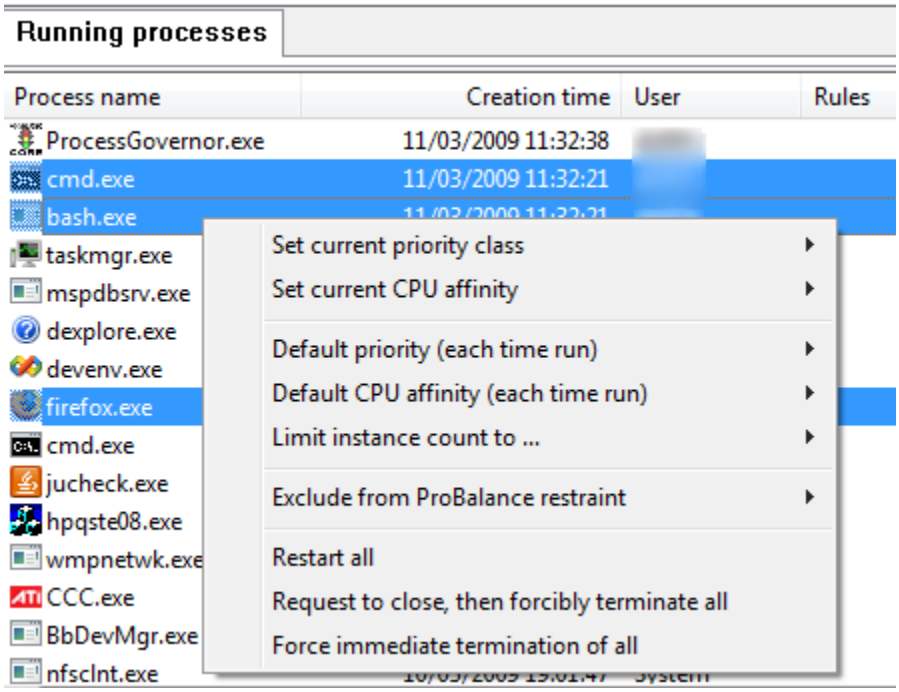
The Process List shows running processes and allows for easy rule creation. You can right click on any process, or on multiple processes, to get a context menu of available options. Amongst many other things, the options include setting current priorities and affinities, as well as default priorities and affinities.

Single process context menu



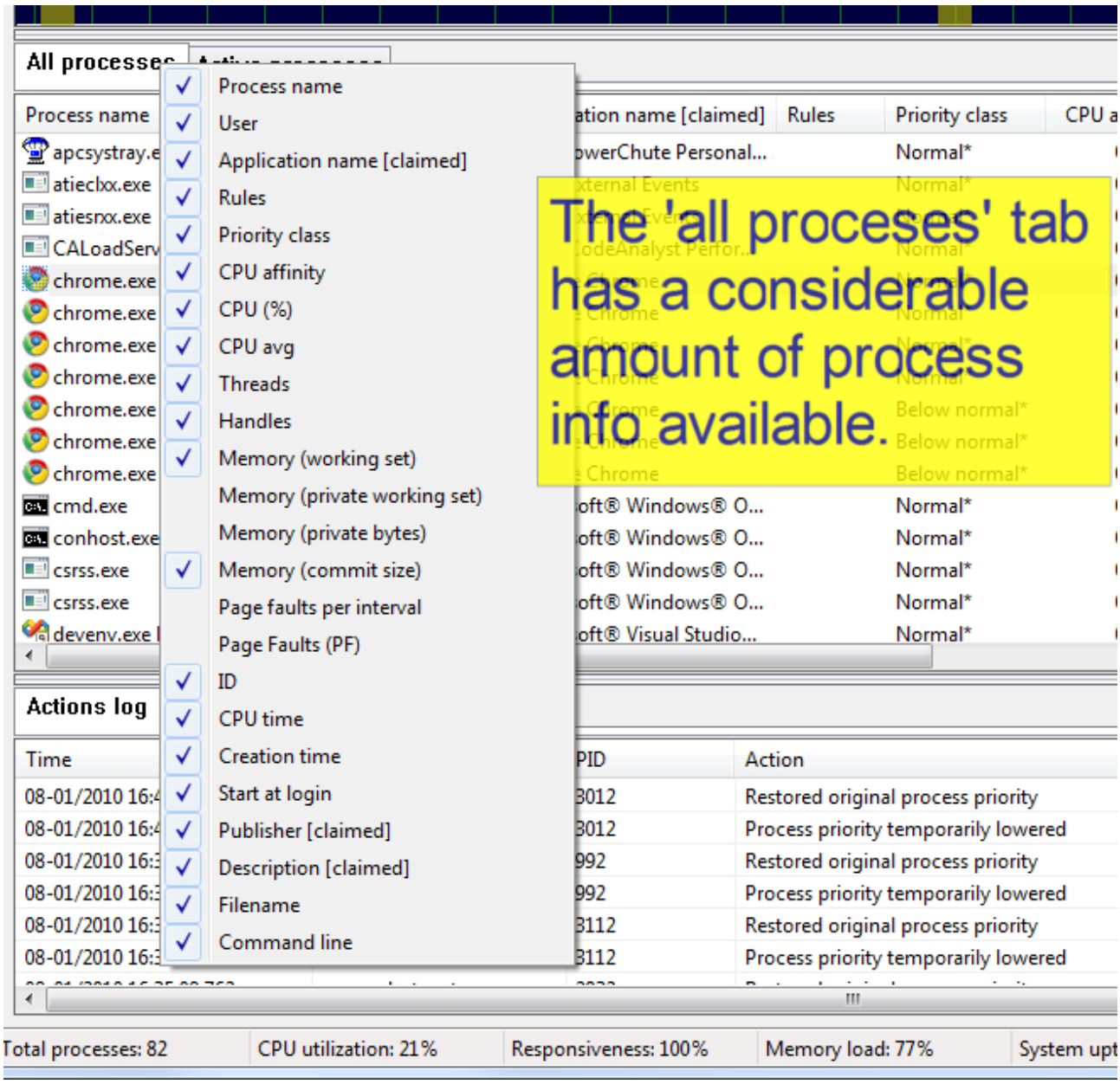
The context menu when multiple processes are selected is different than when a single process is selected, since not all operations can be performed on multiple processes.

Multiple process context menu



Selecting which columns to show

You can select which columns you want visible by right-clicking on the 'All processes' tab, right-clicking on the list header, or using the 'View / Select Process Columns' menu. In addition, you can resize the columns (at their headers) and change the column ordering by dragging the column headers. These view changes will be remembered by Process Lasso. The same applies for the 'Active processes' tab. Note that right-clicking the 'Active processes' tab allows for setting process information shown in that column. Information in the 'Active processes' tab is more limited than information in the 'All processes' tab. However, double clicking on any process in the 'Active processes' tab will take you to its entry in hte 'All processes' tab.



Rules Column

The 'rules' column in the running process list gives a quick summation of rules that match the process. Its format is terse, but users get quickly acclimated to it once they understand how to read it. The format of the Rules column follows.

Rules column format: `[X][x][K][g][<-<<<<][#n][RHANBI][1-31]`

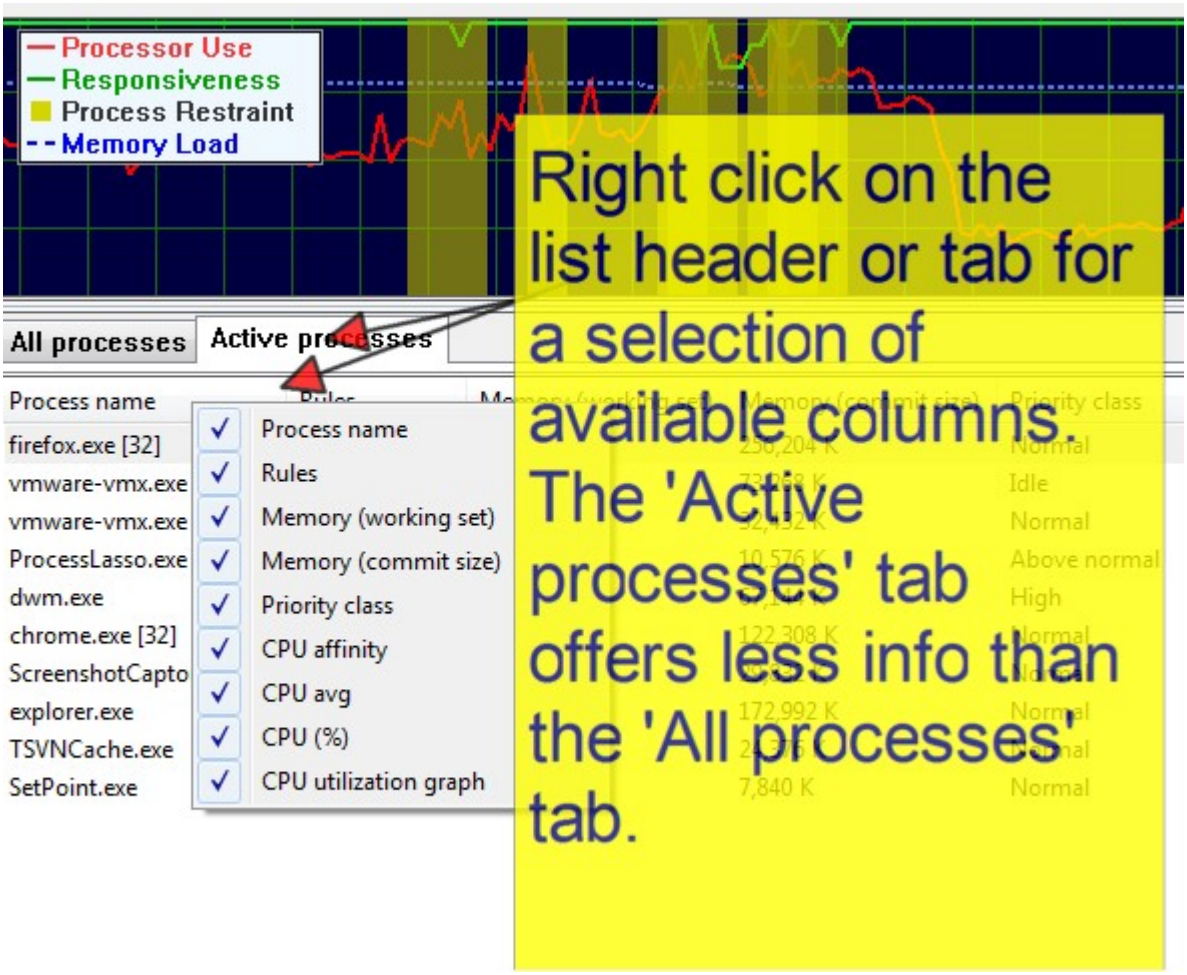
Character(s)	Meaning
X	Excluded from ProBalance restraint
x	Excluded from foreground boosting
K	Keep process running (auto-restart if terminated)
g	Process will induce gaming mode
s	Process will prevent the computer and display from sleeping
p	Process will cause the computer to enter the high performance power mode

Character(s)	Meaning
#n	Instance count limit of n, i.e. #2 for an instance limit of 2
R	Default priority class: Real time
H	Default priority class: Highest
A	Default priority class: Above normal
N	Default priority class: Normal
B	Default priority class: Below normal
I	Default priority class: Idle
1-31	Default CPU affinity, i.e. '02' for CPUs 0 and 2
<	Throttle level lowest
<<	Throttle level low
<<	Throttle level moderate
<<<	Throttle level high

The Active Processes Tab

Selecting which columns to show

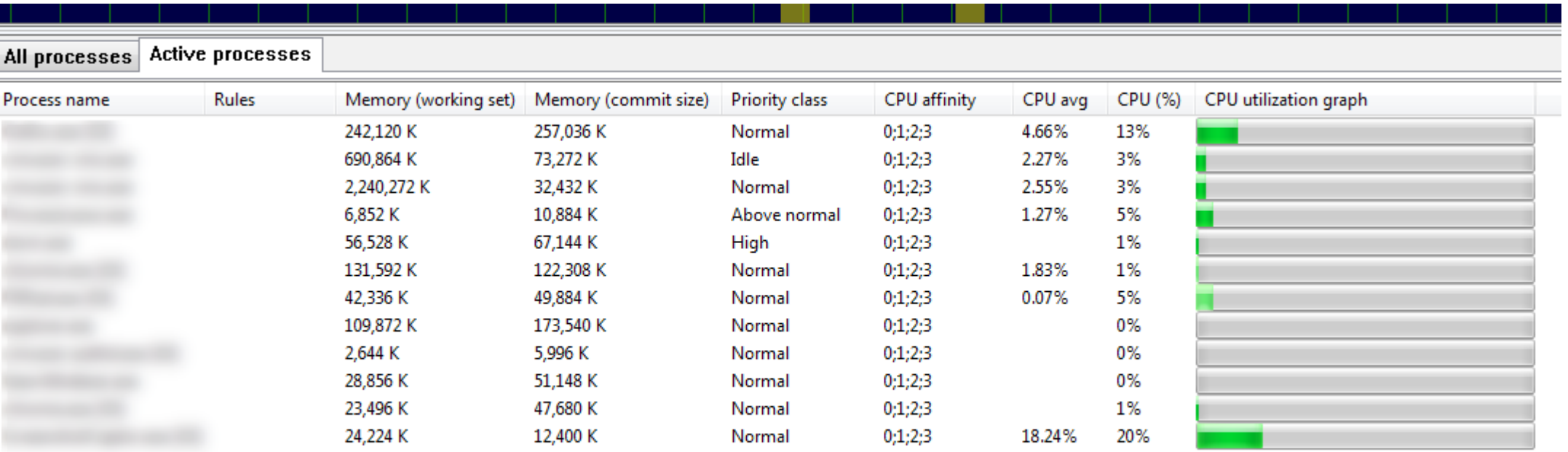
You can select which columns you want visible in the 'Active Processes' tab the same way you can in the 'All Processes' tab. by right-clicking on the 'Active processes' tab or by right-clicking on the list header. In addition, you can resize the columns (at their headers) and change the column ordering by dragging the column headers. These view changes will be remembered by Process Lasso. Double clicking on any process in the 'Active processes' tab will take you to its entry in the 'All processes' tab, where additional information is available.



The Active Processes tab shows only processes that are actively utilizing the system CPU(s). It displays basic information about them, and a horizontal bar graph to visually depict their active CPU utilization.

> **Right clicking** on one or more of the processes in the 'Active Processes' list show the same context menu as found in the 'All Processes' tab.
> **Double clicking** on a process in the 'Active Processes' list will find the corresponding process in the 'All Processes' tab and make it visible.

Screenshot of the 'Active Processes' view:

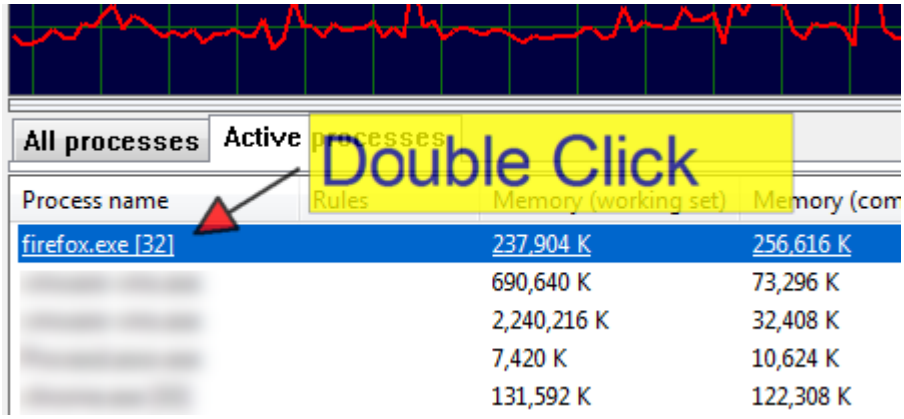


All processes | Active processes

Process name	Rules	Memory (working set)	Memory (commit size)	Priority class	CPU affinity	CPU avg	CPU (%)	CPU utilization graph
[redacted]		242,120 K	257,036 K	Normal	0;1;2;3	4.66%	13%	
[redacted]		690,864 K	73,272 K	Idle	0;1;2;3	2.27%	3%	
[redacted]		2,240,272 K	32,432 K	Normal	0;1;2;3	2.55%	3%	
[redacted]		6,852 K	10,884 K	Above normal	0;1;2;3	1.27%	5%	
[redacted]		56,528 K	67,144 K	High	0;1;2;3		1%	
[redacted]		131,592 K	122,308 K	Normal	0;1;2;3	1.83%	1%	
[redacted]		42,336 K	49,884 K	Normal	0;1;2;3	0.07%	5%	
[redacted]		109,872 K	173,540 K	Normal	0;1;2;3		0%	
[redacted]		2,644 K	5,996 K	Normal	0;1;2;3		0%	
[redacted]		28,856 K	51,148 K	Normal	0;1;2;3		0%	
[redacted]		23,496 K	47,680 K	Normal	0;1;2;3		1%	
[redacted]		24,224 K	12,400 K	Normal	0;1;2;3	18.24%	20%	

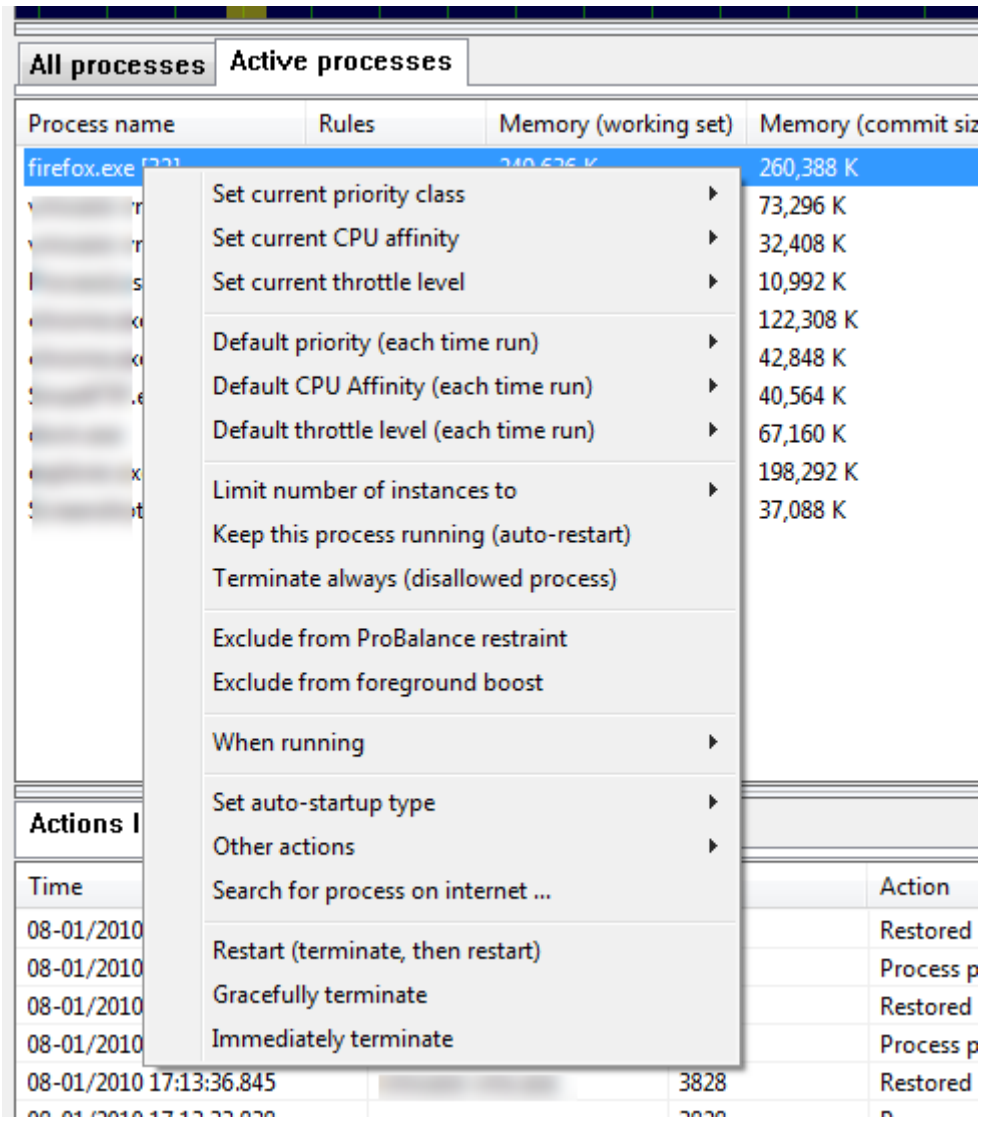
Double clicking a process in the active processes view

You can double click a process in the 'Active processes' tab to go to that process in the 'All processes' tab, where extended information is available.



Single process context menu

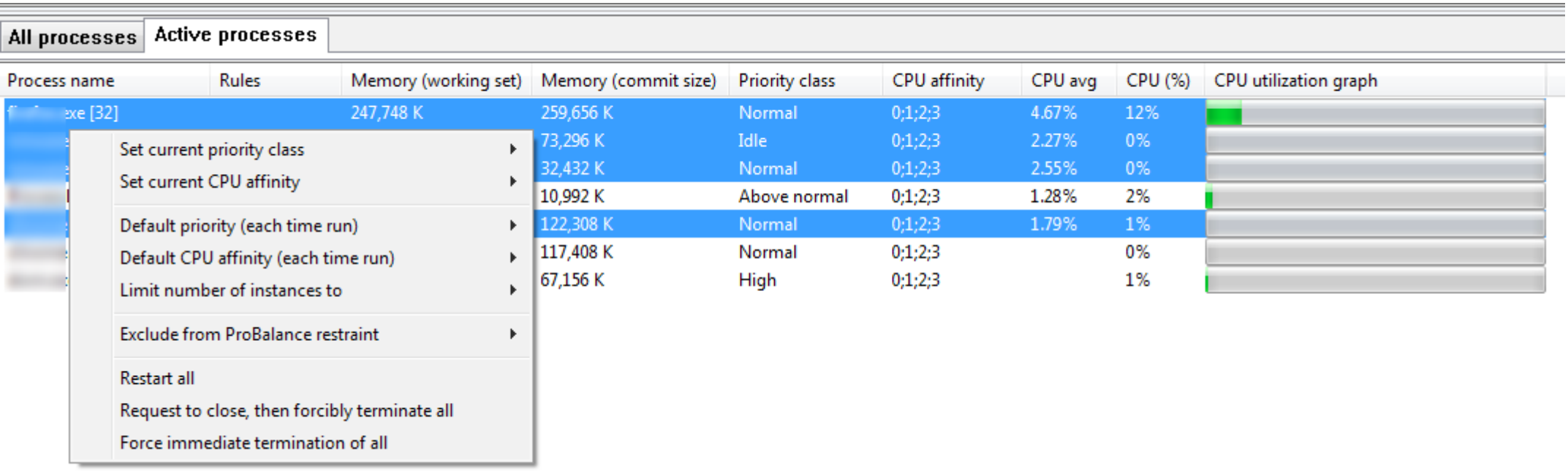
Right-clicking on a process in the 'Active Processes' tab shows the same context menu as in the 'All processes' tab. You can perform any available operation on the process.



The context menu when multiple processes are selected is different than when a single process is selected, since not all operations can be performed on multiple processes.

Multiple process context menu

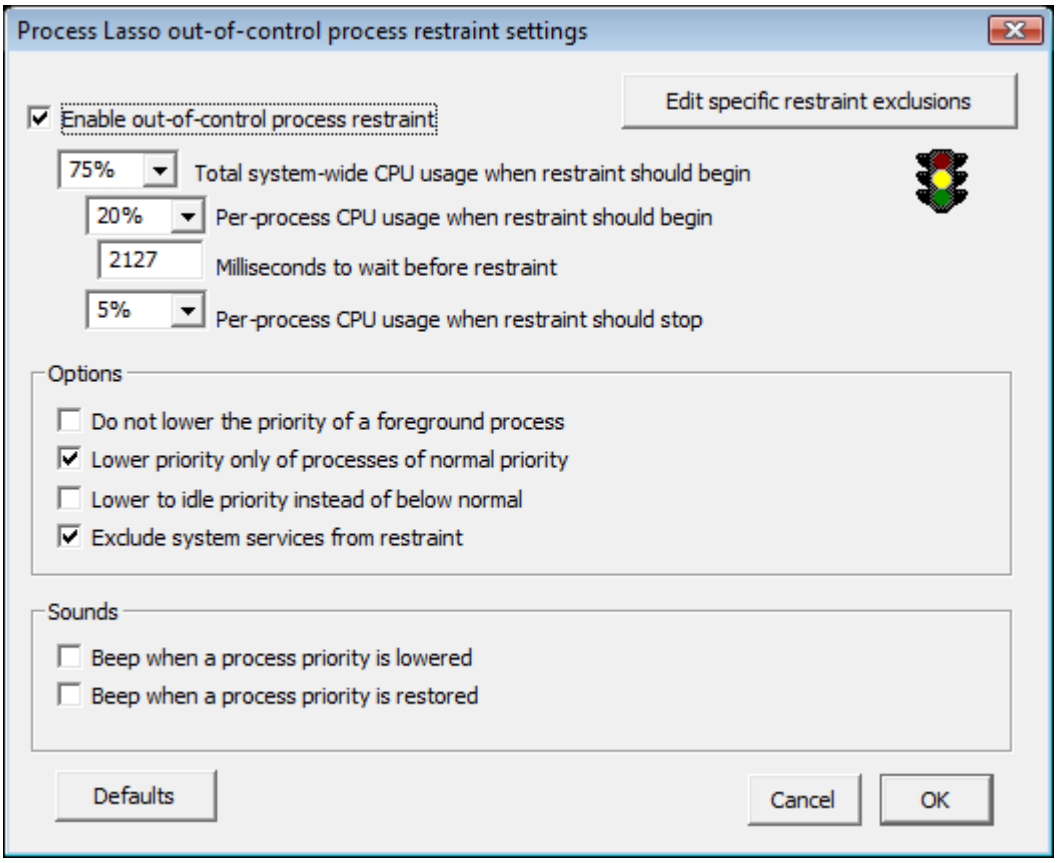
Just as in the 'All processes' tab, you can select multiple processes and right-click on them to perform an operation on all of them.


All processes
Active processes
Process name
Rules
Memory (working set)
Memory (commit size)
Priority class
CPU affinity
CPU avg
CPU (%)
CPU utilization graph
exe [32]
247,748 K
259,656 K
Normal
0;1;2;3
4.67%
12%
73,296 K
Idle
0;1;2;3
2.27%
0%
32,432 K
Normal
0;1;2;3
2.55%
0%
10,992 K
Above normal
0;1;2;3
1.28%
2%
122,308 K
Normal
0;1;2;3
1.79%
1%
117,408 K
Normal
0;1;2;3
0%
67,156 K
High
0;1;2;3
1%
Set current priority class
Set current CPU affinity
Default priority (each time run)
Default CPU affinity (each time run)
Limit number of instances to
Exclude from ProBalance restraint
Restart all
Request to close, then forcibly terminate all
Force immediate termination of all

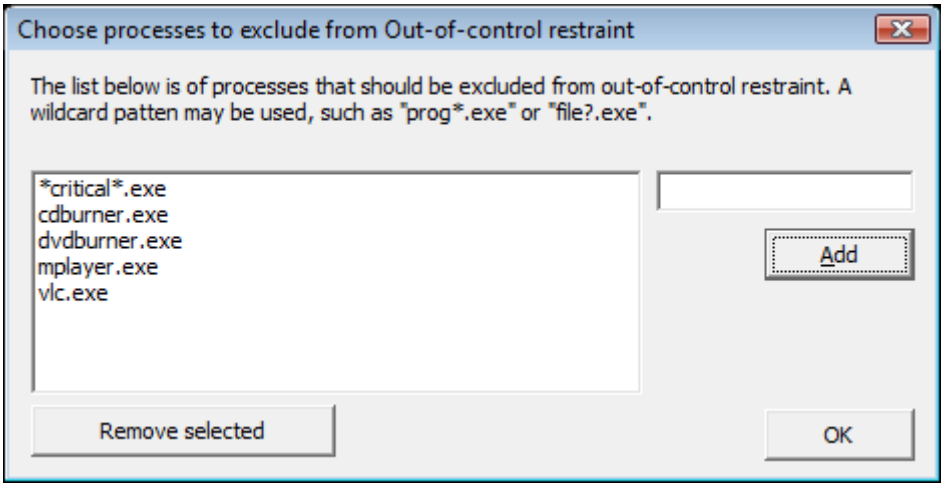
Out-of-Control Options

Tooltips!

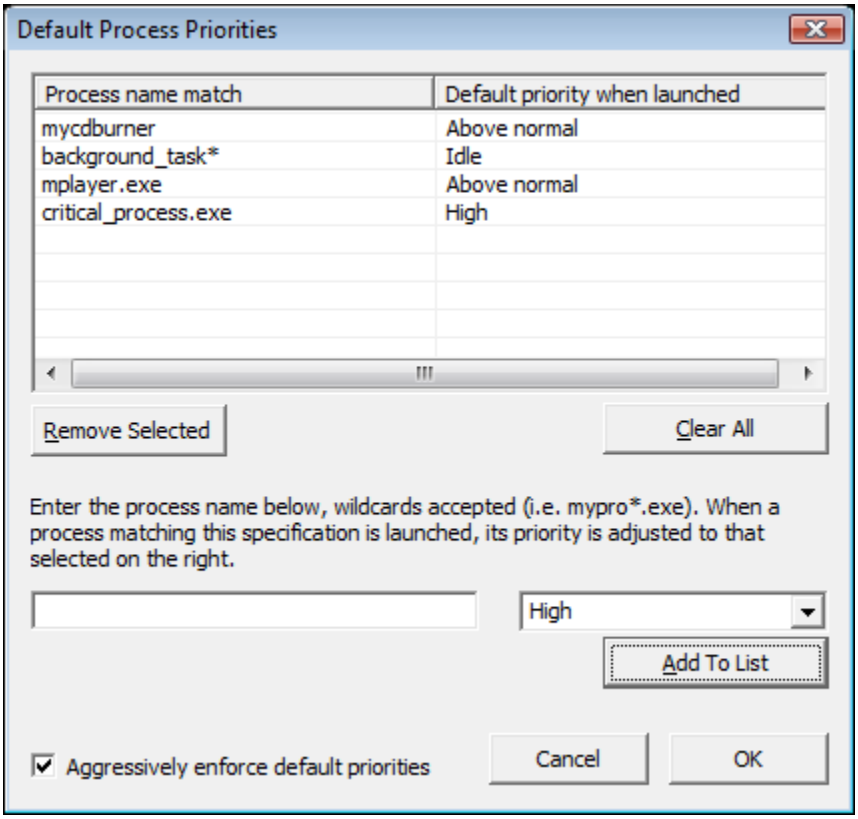
Help on the options in this dialog can be obtained by hovering the mouse cursor over the item you want help on. After a few seconds, a popup tooltip will appear that describes the field you are hovering over.



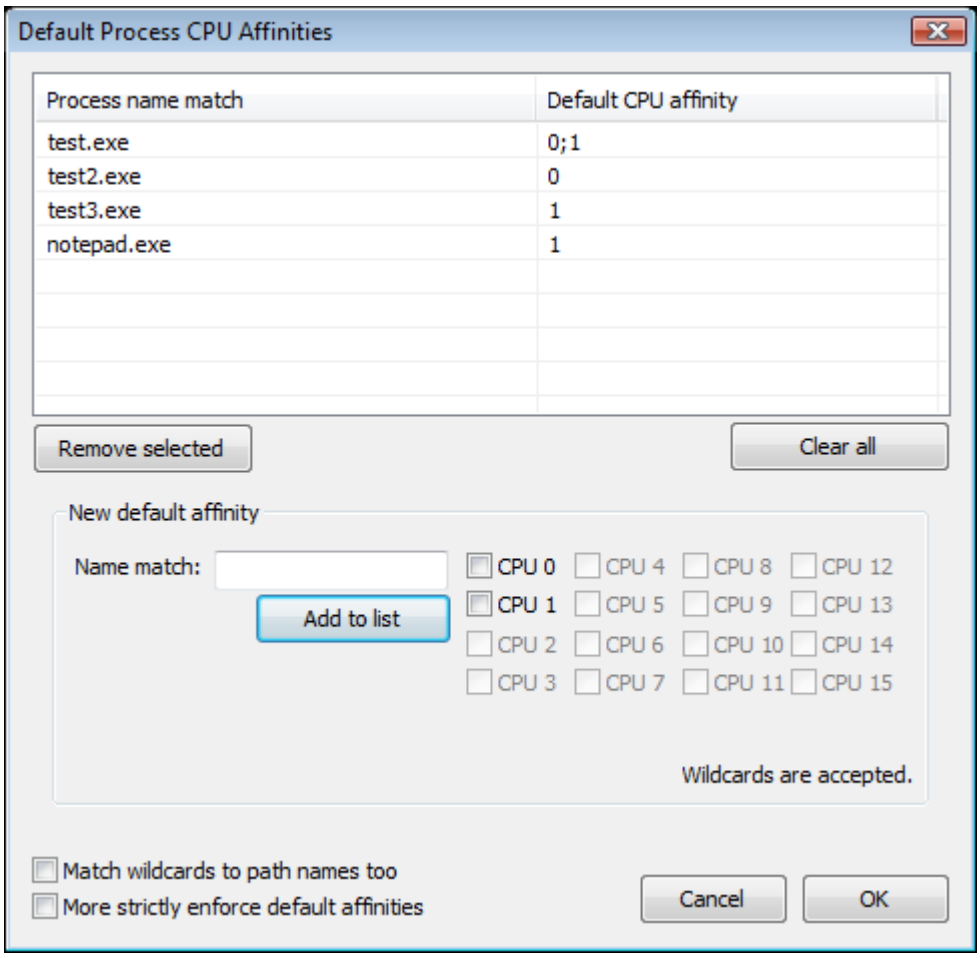
Out-of-Control Exclusions



Default Priorities

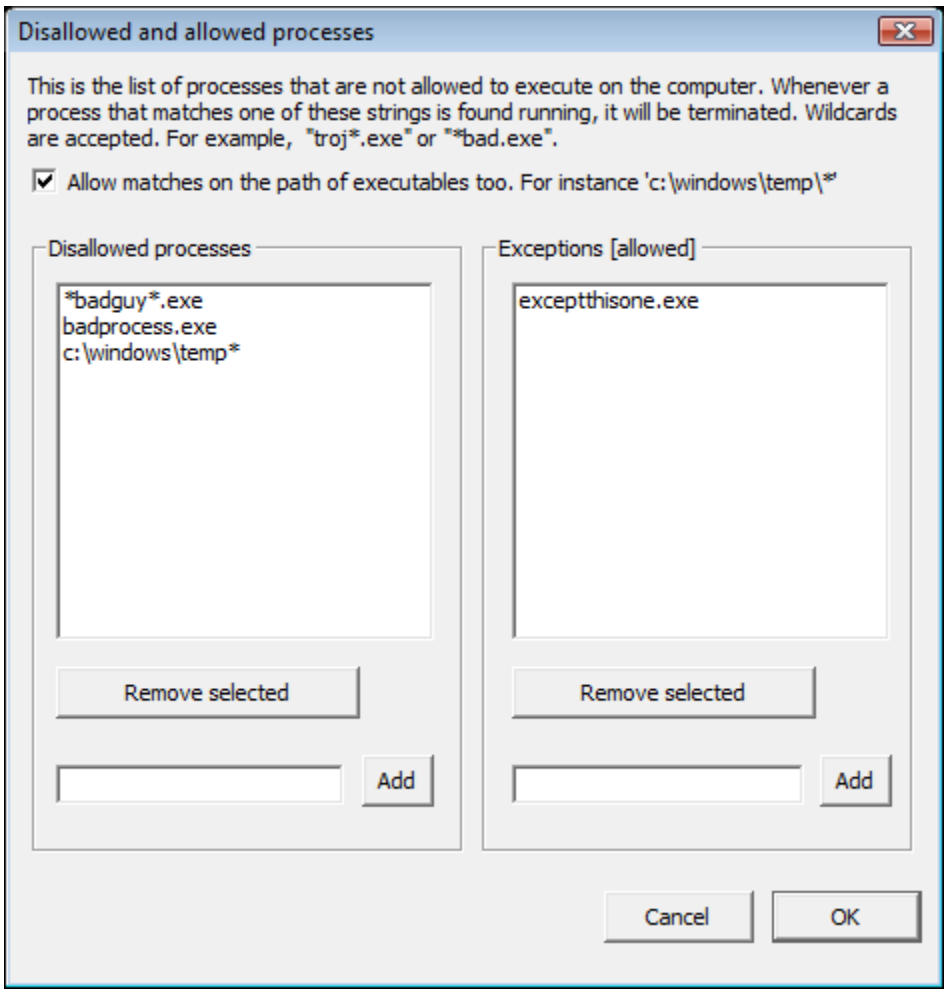


Default CPU Affinities



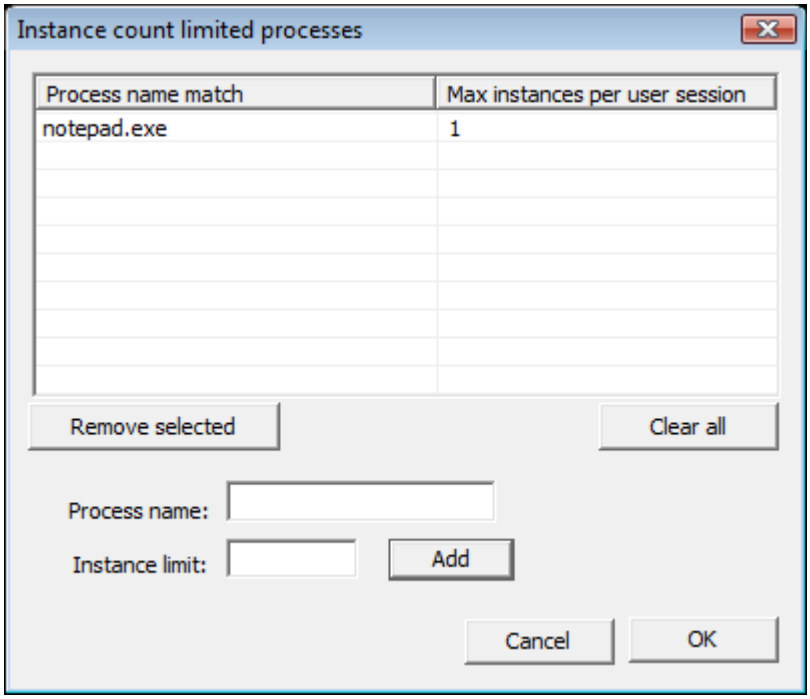
Auto-Terminate List

The processes listed here will get terminated when they are found to be running. Process Lasso can not (at present) actually prevent them from trying to start up, it just immediately terminates them when it finds them running.



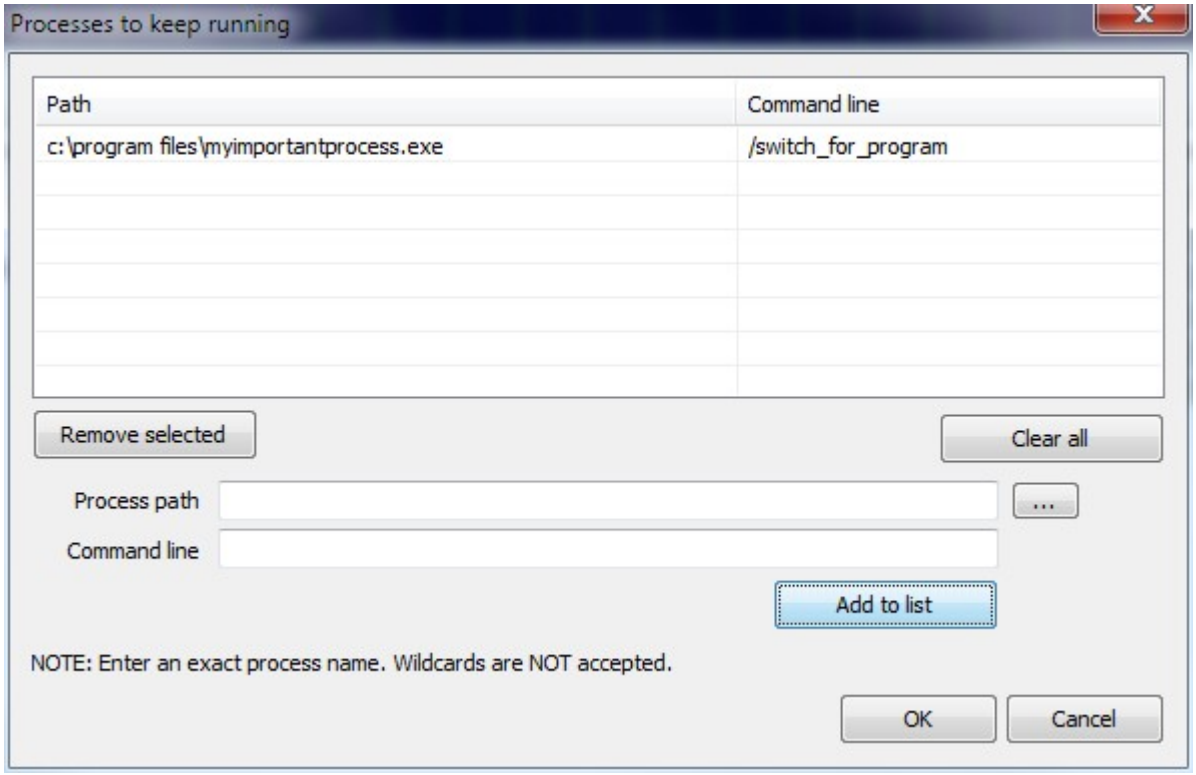
Instance Count Limits

You can limit the number of instances of a process allowed to be running at the same time (per user session) with this dialog. New instances of processes will be terminated if they match a pattern here and the number of instances is already equal to, or greater than, the instance count limit.



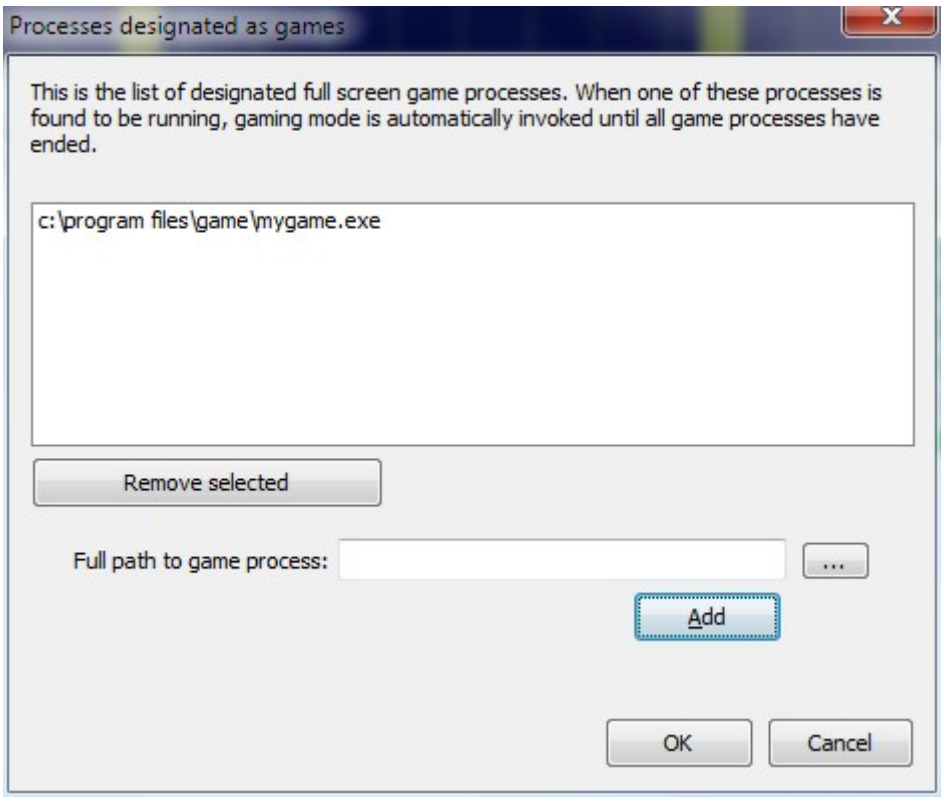
Keep Running Processes

You can ensure certain processes are kept running by entering in this dialog. You can do this by right-clicking on a process, using the 'Keep process running', or by using application menu opion at 'Options / Configure Keep running processes ...'.



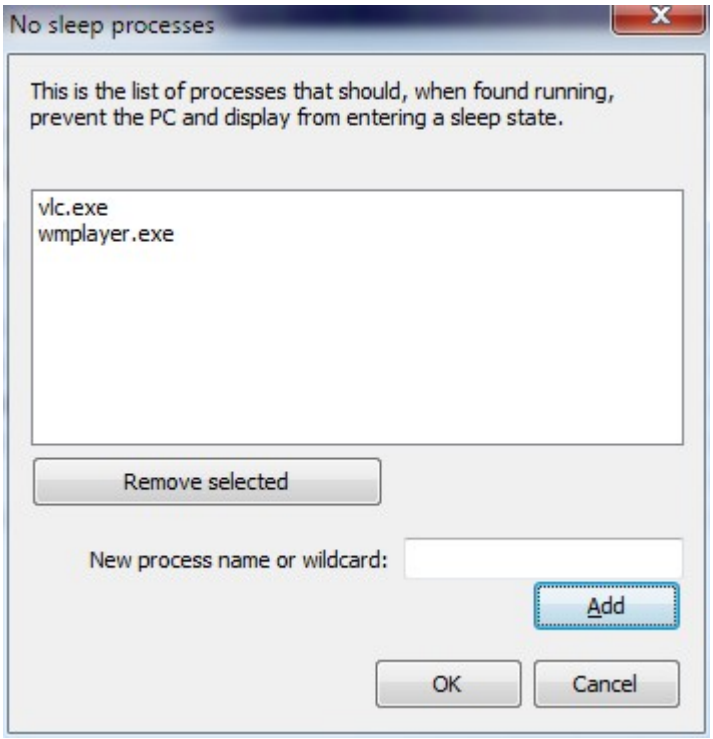
Gaming Processes

You can designate certain proceses as games. This causes Process Lasso to induce the High Performance Power Scheme and make certain adjustments to ProBalance in an effort to ensure all available CPU cycles go to the game. This mode should ONLY be used for games or other very CPU intensive applications that need every bit of processing power. You designate a gaming process by right-clicking on a process, using the 'When running / Turn on gaming mode' menu, or by using application menu opion at 'Options / Configure Game processes ...'.



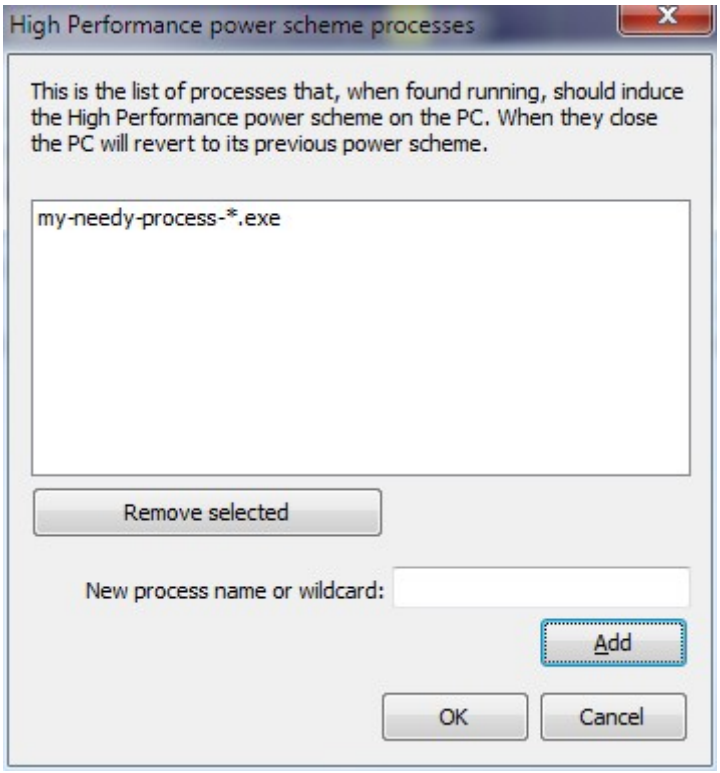
Anti-sleep Processes

You can prevent the PC and display from entering a sleep or hibernate state by adding them to the 'anti-sleep' list. You can do this by right-clicking on a process, using the 'When running' menu, or by using application menu opion at 'Options / Configure Anti-Sleep processes ...'.



High Performance Power Scheme Processes

You can set certain processes to cause the system to enter the 'High Performance' power scheme when they are running. You can do this by right-clicking on a process, using the 'When running' menu, or by using application menu opion at 'Options / Configure High Performance Power processes ...'. Entering this power scheume will disable CPU frequency scaling, giving you maximum performance. It will, however, drain the battery life of laptops, netbooks, and other portable computers faster than typical. When all 'High Performance' power mode processes end, the power scheme is reset back to whatever it originally was.



Main Menu Options

Process Lasso's extensive menu system allows for configuration tweaks and other operations. The menus available are:

- **The application main menu**
  Provides ability to to toggle numerous settings and more. These menu options are described below.
- **The process context menu (right-click)**
  This menu lets you set specific options for a process.
- **The multi-process context menu (right-click)**
  This menu lets you set specific options for several processes.

Using the Process Lasso application menu (the menu at the top) you can set a variety of options. They are explained below.

Item	Description
Main / Show/Hide Process Lasso main window	This will toggle the visibility of the main process management window. When the main window is minimized to the system tray, or otherwise not visible, it consumes less resources through a sleep like mode.

Main / Show balloon notifications	This toggle indicates to show, or not to show, popup balloon notifications in the system tray each time an action is taken.
Main / ProBalance enabled	This toggle allows for quick and easy enabling or disabling of ProBalance dynamic priority adjustments.
Main / All sounds off	This toggle allows you to quickly disable any sound notifications you have configured. Note that by default, no sounds events are enabled. See the ProBalance configuration dialog to enable them.
Main / Shut down Process Lasso	This terminates the Process Lasso GUI (processsupervisor.exe). You will no longer see the system tray icon, but process rules will still be enforced by the process governor (processgovernor.exe) unless you choose to terminate it to (you will be prompted).
File / Choose alternate configuration file	This allows you to select an alternate configuration file for use by Process Lasso. The INI configuration file includes all process rules and process lists.
File / Choose alternate log folder	This allows you to select an alternate location for the log of actions taken by Process Lasso.
File / Export configuration	This exports the INI configuration file to a folder of your choosing.
File / Import configuration	This imports an INI configuration file.
View / Show graph legend	Toggles the visibility of the legend/map that identifies what the colors of the graph indicate.
View / Hide process icons	This turns on or off the icons beside each process in the list. Turning OFF these icons conserves a considerable amount of memory. A restart of Process Lasso is required after changing this setting.
View / Technical columns first	This causes the process info columns to be reordered so that more technical columns are first in order. The user may also simply drag the column headers to whichever order they prefer. Process Lasso will remember the column order and sizes.
View / Reset column sizes	This resets the process info column sizes to their default values.
View / Reset column order	This resets the process info column ordering to the default state.
Options / General settings / Process Lasso startup options	This allows you to switch the startup type of the Process Lasso GUI. The core engine's startup type can be modified by running the installer.

Options / General settings / System tray icon	This allows modification of the system tray icon type. You can choose to either have a static icon, a representation of CPU utilization, or a representation of system responsiveness.
Options / General settings / System tray balloon popup notifications	This toggles whether you want balloon notification tips to be shown in the system tray area when certain actions are taken by Process Lasso. These are nice and informative, but may get annoying after a while, hence by default they are disabled.
Options / General settings / Refresh speed	Selects the desired refresh speed of the graphical user interface and the response time of the Process Governor. A lower value will cause a faster refresh speed and response time, but will utilize more CPU cycles. There is no recommended poll value here, and you can manually change it to whatever you like in the INI file.
Options / General settings / Reset language selection	This resets the language selection of Process Lasso. When Process Lasso is next launched, it will ask you which language you wish to use.
Options / General settings / Reset MessageBox choices	This resets the 'Do not ask again' type saved settings of all Message Box prompts within Process Lasso.
Options / General settings / Periodically check for product updates	Toggles whether Process Lasso should periodically check to see if updates are available. You will be notified before this check occurs, unless you configure it to proceed automatically (an option in the update checker dialog shown to you upon update check).
Options / ProBalance settings / Enable ProBalance priority adjustments	This toggle enables or disabled ProBalance. When enabled, background processes using a lot of CPU cycles will be temporarily lowered in priority, when appropriate, to prevent interruption of foreground processes.
Options / ProBalance settings / Configure ProBalance parameters	This invokes a dialog with configuration parameters for ProBalance, including CPU usage thresholds.
Options / ProBalance settings / Configure excluded processes	This allows for specification of which processes should be excluded from ProBalance priority adjustments. Any process specified here will never have its priority lowered by ProBalance. Wildcards are supported. Processes can also be added or removed from this list by right-clicking on them in the process list-view.
Options / ProBalance settings / Exclude foreground processes	This toggle indicates whether to exclude all foreground processes from ProBalance adjustments. The default value is ON, and it is recommended users leave this alone unless they have a need to switch it off.

Options / ProBalance settings / Exclude processes of non-normal priority	This toggle indicates whether to exclude processes of non-normal priority from ProBalance adjustments. The default value is ON.
Options / ProBalance settings / Exclude processes of non-normal priority	This toggle indicates whether to exclude service processes from ProBalance adjustments. The default value is OFF.
Options / ProBalance settings / Play a sound when action is taken	This toggle indicates whether to play a sound when a ProBalance action is taken. The default value is OFF.
Options / Foreground Boosting / Enable Foreground Process Priority Class Boost	This temporarily boosts the priority class of a process when one of its windows is in the foreground (has keyboard / mouse focus). This setting will boosts the priorities of all threads in that process. Note that this does not adjust Windows builtin foreground application boosting mechanism, which gives foreground applications longer time slices (see TweakScheduler for that). It is recommended you leave this option OFF unless you need it. The default value is OFF.
Options / Foreground Boosting / Enable Foreground Thread Priority Boost	This temporarily boosts the priority of an individual thread owning the foreground window. This can help increase responsiveness in some cases. Note that this does not adjust Windows builtin foreground application boosting mechanism, which gives foreground applications longer time slices (see TweakScheduler for that). It is recommended that you leave this option OFF unless you need it. The default value is OFF.
Options / Configure default priorities ...	This invokes a dialog allowing you to configure default priority classes of processes. Wildcards are supported. You can also set the default priority class of processes by right clicking on them in the process list-view.
Options / Configure default CPU affinities	This invokes a dialog allowing you to configure default CPU affinity of processes. The CPU affinity is the selection of CPUs the process is allowed to run on. Wildcards are supported. You can also set the default CPU affinities of processes by right clicking on them in the process list-view.
Options / Configure disallowed processes ...	This invokes a dialog box allowing you to choose which processes should be automatically terminated if they are found to be running. Wildcards are supported. You can also add processes to this list by right clicking on them in the process list-view.
Options / Configure process instance limits ...	This invokes a dialog box allowing you to place limits on the number of instances a program can have running in a user session. Wildcards are supported.

Options / More strictly enforce default priorities and affinities	This toggle is called 'Forced mode', and indicates to forcibly keep process default priorities and affinities applied, even if those processes try to adjust them back.
Options / Scheduled virtual memory trim	These menu options allow you to do a one time, or periodic, trim of the RAM used by all processes. This forces them to page their memory out to the swap file. As the memory is reaccessed by the processes, it is reloaded into RAM, incurring a performance penalty. Normally Windows manages virtual memory fine and manually forcing processes to page their memory out interferes with its optimality. However, for some users, this type of periodic operation may be useful.
Options / Log Settings /	These items allow for toggling whether or not certain events are logged. You can also set the maximum number of log lines here.
Help / Check for updates	This invokes the update check dialog.
Help / About	This invokes the Process Lasso About box, which contains information about Process Lasso.

Using the INI Configuration File

The configuration options and process rules are stored in an INI configuration file. You can manually edit this INI file, import/export it, choose an alternate INI file to use, or pretend it doesn't exist. Usually system and network administrators or those with a lot of process rules need to manually tweak the INI file. Otherwise, the Process Lasso graphical user interface (GUI) will help you edit it.

Where is the INI configuration file located?

By default, Process Lasso automatically generates a configuration file for each user context in which it runs. These configuration files are located in the "ProcessLasso" subfolder of the respective user's application data folder. The log is also stored in this same location. The 'About' window of Process Lasso will show the full paths.

Administrators can change the path of the configuration file so that all instances of Process Lasso/Process Governor use the same configuration file. This can be done running the installer again and choosing a global configuration path in the second configuration dialog.

Sample Configuration File

```
#
# prolasso.ini configuration file [sample]
# any line not recognized is ignored
#
[OutOfControlProcessRestraint]
OocOn=True
TotalProcessorUsageBeforeRestraint=85
PerProcessUsageBeforeRestraint=20
TimeOverQuotaBeforeRestraint=2800
PerProcessUsageForRestore=10
PlayOnRestraint=C:\Windows\media\Windows Pop-up Blocked.wav
PlayOnRestore=C:\Windows\media\Windows Feed Discovered.wav
MinimumTimeOfRestraint=6000
TameOnlyNormal=True
LowerToIdleInsteadOfBelowNormal=False
ExcludeServices=False
PlaySoundOnRestraint=False
PlaySoundOnRestore=False
RestrainByAffinity=False
RestraintAffinity=
ExcludeForegroundProcesses=True
DoNotLowerPriorityClass=False
OocExclusions=
[GUI]
HideGraph=False
HideProcessIcons=False
ShowGraphLegend=True
ClearLogAtExit=False
[Performance]
ManageOnlyCurrentUser=True
ExitOnCloseWindow=False
BoostForegroundThread=False
BoostForegroundProcess=False
SoundsOff=True
AggressivelyTrimProcessLassoWorkingSet=True
CloseApplicationTimeoutSeconds=10
ForcedMode=False
[Updates]
CheckForUpdates=False
[SystemTrayIcon]
UseStaticIcon=False
ShowResponsivnessInTrayInsteadOfProcessorUsage=False
[Logging]
LogAllProcessesExecuted=False
LogProcessesDisallowed=True
LogDefaultPriorityAdjustments=True
LogDefaultAffinityAdjustments=True
LogOutOfControlProcessesRestrained=True
LogOutOfControlProcessesRestored=True
LogInstanceLimitTerminations=True
```

```
MaximumLogEntries=50
[MemoryManagement]
TrimAllProcessesAtThisIntervalInMs=0
[SysTrayBalloons]
ShowBalloons=False
BalloonTipDuration=10000
ShowBalloonsForOocPriorityRestoration=False
[Performance]
UpdateSpeed=1500
[ProcessAllowances]
AllowedProcesses=badprog2.exe
DisallowedProcesses=badprog*.exe
InstanceLimitedProcesses=word.exe,excel.exe
[ProcessDefaults]
DefaultPriorities=notepad.exe,above normal;calc.exe,above normal
DefaultAffinities=notepad.exe,1;calc.exe,0
MatchWildcardsToPathnames=False
```

Names encapsulated in '[' and ']' are groups of options. Each group has one or more options (called keys) that are set to values. Each key is optional, and if it doesn't exist the default value is used. Lines that do not contain a recognized key or group name are ignored. This file is not case sensitive.

Groups Overview

Group	Description	Keys
Performance	Configures performance related parameters.	UpdateSpeed
OutOfControlProcessRestraint	Configures parameters related to restraint of out of control processes (those consuming too many system resources).	OocOn TotalProcessorUsageBeforeRestaint PerProcessUsageBeforeRestraint TimeOverQuotaBeforeRestraint ProcessUsageQuotaCausingRestore OocExclusions
ProcessAllowances	Configures processes allowed and disallowed from execution.	AllowedProcesses DisallowedProcesses InstanceLimitedProcesses MatchDisallowedWildcardsToPathnames
ProcessDefaults	Configures default process priorities.	DefaultPriorities
Logging	Configures logging options.	LogAllProcessesExecuted LogProcessesDisallowed LogDefaultPriorityAdjustments LogOutOfControlProcessesRestrained LogInstanceLimitTerminations MaximumLogEntries

Keys

Defined keys (options) and their accepted values are as follows:

Group	Key	Description	Accepted Values and Format	Default Value
Performance	Update Speed	The interval between enumerations of running processes by ProcessGovernor and ProcessLasso, in milliseconds. The lower the interval more responsive the applications are to changes in running processes, but more CPU cycles are used.	Any non-negative number. It is not recommended to exceed 4000ms or go below 500ms. It is recommended to keep this value the default.	1500
Performance	Forced Mode	When forced mode is enabled, Process Lasso more aggressively enforces default priorities and affinities. Some applications change their own priorities, and this setting may be necessary in such cases.	'True' if this feature is enabled, 'False' if not.	False
OutOfControlProcessRestraint	OocOn	Toggles out-of-control process restraint on or off.	'True' if this feature is enabled, 'False' if not.	True
OutOfControlProcessRestraint	TotalProcessorUsageBeforeRestraint	The percentage of processor(s) in use that initiates a check of processes that may be out-of-control. While the total processor usage is below this threshold, processes are not restrained. This value includes	Range: 1-100.	75

		usage of all processors on a system.		
OutOfControlProcessRestraint	TimeOverQuotaBeforeRestraint	Number of milliseconds above ProcessorUsageQuota before the process priority is lowered.	Any non-negative number.	1750
OutOfControlProcessRestraint	PerProcessUsageQuotaBeforeRestraint	The percentage of processor(s) in use by a process before its priority is reduced. Note that TotalProcessorUsageBeforeRestraint must have already been met.	1-100. This value must be equal to or lesser than ProcessorUsageQuota.	20
OutOfControlProcessRestraint	PerProcessUsageQuotaForRestore	The percentage of processor(s) in use by a process that is considered a safe enough value to restore a preciously restrained process to its original priority.	1-100. This value must be equal to or lesser than ProcessorUsageQuota.	5
OutOfControlProcessRestraint	OocExclusions	Processes excluded from out-of-control restraint.	This is a comma delimitered list of process names. Wildcards are allowed. Format: process1,process2,process3 Example: taskmgr.exe, game*.exe	None.
ProcessAllowances	DisallowedProcesses	Processes disallowed to execute. If a process is launched matching a specification in this list, it will be terminated.	This is a comma delimitered list of process names. Wildcards are allowed. Format: process1,process2,process3 Example: taskmgr.exe, game*.exe	
ProcessAllowances	AllowedProcesses	Processes allowed to execute. These are only applicable if they match a process specification in the DisallowedProcesses key. For example, if notepad*.exe is	This is a comma delimitered list of process names. Wildcards are allowed. Format: process1,process2,process3 Example: taskmgr.exe, game*.exe	None.

		lised in DisallowedProcesses and notepad1.exe is listed in AllowedProcesses, then notepad1.exe will be allowed to execute, but notrpad2.exe will not.		
ProcessAllowances	InstanceLimitedProcesses	This variables provides a list of processes who should be limited to no more than a certain number of instances in a given user session.	This is a comma delimitered list of process names, with semicolons used to indicate the maximum instance count. Example: myprogram.exe;2	
ProcessDefaults	DefaultPriorities	A list of default priorities for processes.	This is a comma delimitered list of process names and priorities. Wildcards are allowed. Valid priorities are: • Real time • High • Above normal • Normal • Below normal • Idle Format: process1,priority1,process2,priority2 Example: taskmgr.exe, high,*.scr,idle	None.
Logging	LogAllProcessesExecuted	Toggle logging of all processes executed.	'True' if enabled, 'False' if not.	False
Logging	LogProcessesDisallowed	Toggle logging of processes terminated because they were disallowed.	'True' if enabled, 'False' if not.	True
Logging	LogDefaultPriorityAdjustme	Toggle logging of processes whose priority was adjusted to a configured	'True' if enabled, 'False' if not.	True

	nts	default value.		
Logging	LogOutOfControlProcesses	Toggle logging of processes restrained because they were found to be out-of-control.	'True' if enabled, 'False' if not.	True
Logging	MaximumLogEntries	The maximum number of log entries. When the log file reaches this limit, it is resized to 80% of this value.	Minimum reasonable value is 10.	500
Logging	LogInstanceLimitTerminations	Set to log terminations of processes based on their instance limit, as configured.	'True' if enabled, 'False' if not	True

Peformance note: Process Lasso uses its own high performance INI parsing algorithm.

Command Line Arguments

Command Line Arguments for Installer (also valid for InstallHelper)

Switch (defaults in bold)	**Description**	**Examples**
/S (case sensitive)	Performs a silent installation. Since no configuration dialogs will be shown during install, you should compliment this switch with others (listed below) to specify how you would like Process Lasso configured.	ProcessLassoSetup.exe /S
/language=[**English**] • English • Russian • Serbian • Japanese • PortugueseBr • Dutch • TradChinese • SimpChinese	This indicates the language to use. If this value is not supplied, it defaults to English. Do not try to use unsupported languages, and be sure your language name is spelled as listed here (i.e. PortugueseBr).	ProcessLassoSetup.exe /S /language=English ProcessLassoSetup.exe /S /language=Serbian ProcessLassoSetup.exe /S /language=Japanese ProcessLassoSetup.exe /S /language=PortugueseBr
/gui_start_type=[all|current|manual]	This indicates whether to start the GUI at login for ALL users (all), for only the current user (current), or neither (manual). The current user is the user context in which the installer is running.	ProcessLassoSetup.exe /S /gui_start_type=all ProcessLassoSetup.exe /S /gui_start_type=current ProcessLassoSetup.exe /S /gui_start_type=manual
/governor_start_type=[all|current|service|manual]	This indicates whether to start the core engine (processgovernor) at login for ALL users (all), for only the current user (current), as a service, or neither (manual). The current user is the user context in which the installer is running. When /governor_start_type=service, the /username and /password switches can be used as modifiers to indicate in what user context the service should run. If a username and password isn't supplied, it will run in the system context.	ProcessLassoSetup.exe /S /governor_start_type=all ProcessLassoSetup.exe /S /governor_start_type=current ProcessLassoSetup.exe /S /governor_start_type=service ProcessLassoSetup.exe /S /governor_start_type=service /username=myuser /password=mypass

/rights=[highest|**normal**]	This indicates whether to run Process Lasso and its core engine with HIGHEST rights, or with NORMAL rights. When run with HIGHEST rights, each instance can manage the processes of ALL users on the system. When running with NORMAL rights, each instance only manages the processes in its current user context. Normally, each user has his or her own instance of Process Lasso to manage their processes, therefore NORMAL is the recommend and default value. When running the core engine as a service, this setting is forced to HIGHEST. Please note that if HIGHEST rights is selected, Windows Defender will BLOCK startup of the GUI in Vista /w UAC enabled. It will notify you and allow you to run the application at your choice. It will not block startup of the core engine when the core engine is running as a service. The default value of this parameter is NORMAL.	ProcessLassoSetup.exe /S /rights=normal ProcessLassoSetup.exe /S /rights=highest
/logfolder=[folder]	This indicates to use a global log folder for ALL users on the system. By default, each user has his or her own log folder in their respective application data directory. However, it is sometimes desirable to consolidate all log events into a single log folder. Be sure that this log folder is writable by all users on the system.	ProcessLassoSetup.exe /S /logfolder=c:\systemlogs ProcessLassoSetup.exe /S /logfolder=c:\systemlogs
/configfolder=[folder]	This indicates to use a global configuration folder for ALL users on the system. By default, each user has his or her own configuration in their respective application data directory. However, it is sometimes desirable to use the same configuration for all	ProcessLassoSetup.exe /S /configfolder=c:\pl_config ProcessLassoSetup.exe /S /configfolder=c:\pl_config

	users. Be sure that this configuration folder is at least readable by all users, and writable by those who you wish to allow configuration changes.	
/launch_gui=[**true**|false]	This indicates whether or not to launch the GUI after installation. Even when the GUI is launched, it remains minimized to the system tray.	ProcessLassoSetup.exe /S /launch_gui=true
/importconfigfrom=[path]	This indicates to import the configuration from the file you've specified. This should be used in conjunction with /configfolder. The file you specify here will be the initial configuration for Process Lasso. You can specify only a folder OR a specific filename here. If you specify only a folder, "prolasso.ini" will simply be appended to the path name.	ProcessLassoSetup.exe /S /configfolder=c:\pl_config /importconfigfrom=c:\temp ProcessLassoSetup.exe /S /configfolder=c:\pl_config /importconfigfrom=c:\temp\my_default_config.ini
/username=[user] /password=[pass]	When running the core engine as a service (/governor_start_type=service), these switches can be used to indicate under which user you should start the service. The user MUST have permission to 'logon as a service' (see our FAQ for info on allowing that). If a specific user is not supplied, the core engine will run in the system context. Please remember that the Windows user password is case sensitive.	ProcessLassoSetup.exe /S /governor_start_type=service /username=myuser /password=mypass
/configpw=[password]	This sets the configuration password. This password is required for by the GUI when a user tries to make a change to the configuration file. Use this parameter without a value to remove any existing password.	ProcessLassoSetup.exe /S /configpw=somepassword

Command Line Arguments for ProcessLasso.exe

Command line arguments for the GUI and core engine are available. You can use them to over-ride the configuration and log paths, but this is not normally necessary or recommended. Changes to the configuration and log paths should instead be performed by the installer (preferred) or the InstallHelper application.

Switch	Description	Examples
/Tray	Causes the main window to not be immediately displayed.	ProcessLasso.exe /tray
/ConfigFile=	Indicates the configuration file that should be used.	ProcessLasso.exe /ConfigFile="c:\program files\processsupervisor\myconfig.ini" ProcessLasso.exe /ConfigFile=c:\config\psconfig.ini ProcessLasso.exe /ConfigFile=\\server1\configfiles\prolasso.ini
/LogFolder=	Indicates the folder containing the log files. This folder requires read and write access.	ProcessLasso.exe /LogFolder=\\server1\logfolder ProcessLasso.exe /LogFolder=c:\pslogs
/Config	Invokes the Out-of-control Process Restraint configuration dialog. This is primarily only used by the installer. Note that this creates a separate instance of Process Lasso that displays the configuration dialog, then exits when the dialog is closed.	ProcessLasso.exe /config

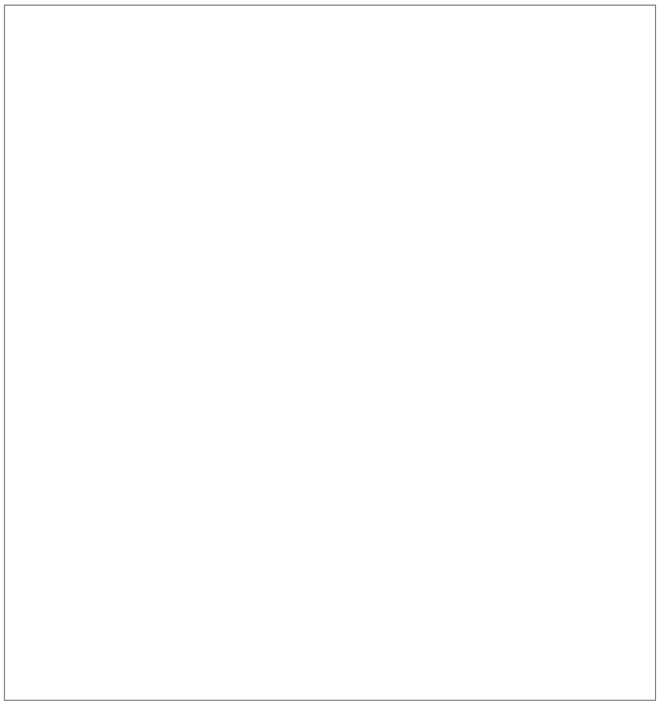
Command Line Arguments for ProcessGovernor.exe

Switch	Description	Examples
/ConfigFile=	Indicates the configuration file that should be used.	ProcessGovernor.exe /ConfigFile="c:\program files\processsupervisor\myconfig.ini" ProcessGovernor.exe /ConfigFile=c:\config\psconfig.ini ProcessGovernor.exe /ConfigFile=\\server1\configfiles\prolasso.ini
/LogFolder=	Indicates the folder containing the log files. This folder requires read and write access.	ProcessGovernor.exe /LogFolder=\\server1\logfolder ProcessGovernor.exe /LogFolder=c:\pslogs

Advanced Tools

TweakScheduler

This utility allows for adjustment of the NT CPU scheduler. Tweaking this changes how CPU time slices are allocated to running threads. It is not generally recommended that users modify these settings, but in some special cases users may have a need.



Vista Multimedia Scheduler Configuration Tool

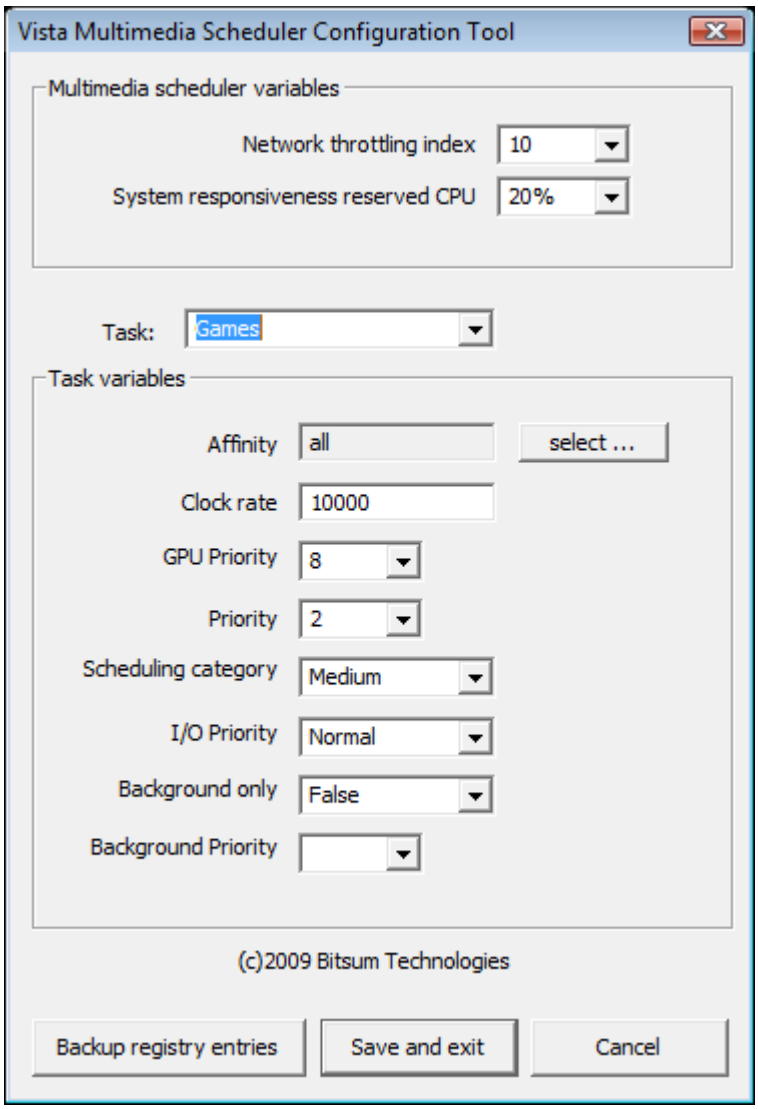
This utility allows for adjustment of the Windows Vista/7 Multimedia Class Scheduler. The Multimedia Class Scheduler (MMCSS) is a new component added to Windows Vista to allow multimedia applications to better guarantee the availability of CPU cycles to their threads. Multimedia applications, which includes games, are grouped into different categories. Each category has its own scheduling settings. Note that applications must register themselves with the Multimedia Scheduler in order to take advantage of it. Most multimedia applications written for Windows Vista should do this, but some legacy applications may not.

Global Options

Option	Description
Network Throttling Index	This is the number of network packets per millisecond allowed during multimedia playback. The higher you set this value, the more network traffic will be allowed during playback and the more likely glitches in playback will occur. It is recommended to leave it at its default value unless you know what you are doing.
System Responsiveness Reserved CPU	This is the percentage of available CPU cycles that should be reserved for non-multimedia threads during multimedia playback. This prevents multimedia application(s) from bringing your system to a halt. Higher settings increase responsiveness, while potentially lowering the quality of realtime multimedia playback.

Task Options

Option	Description
Name	This is the name given to the task. Multimedia applications notify the scheduler that they are about to their thread is about to perform a certain task by specifying its name. The thread then adopts the attributes set for that task type.
Affinity	This is the selection of CPUs that multimedia applications in this class are able to use.
Clock Rate	This is the maximum guaranteed clock rate given to tasks of this classification (in 100 nanosecond units).
GPU Priority	This is the default GPU priority level for tasks in this classification. It is currently unused by Windows.
Priority	This is the default CPU priority level that should be given to tasks in this classification. If the scheduling category is set to 'High', this value will be overridden to '2'.
Scheduling Category	This is the scheduling category for tasks of this classification.
SF I/O Priority	This is the default SuperFetch I/O priority for tasks of this classification.
Background Only	This setting indicates whether or not to run tasks in this classification run in the background ONLY, with NO user interface.
Background Priority	When tasks of this classification are running in the background, this is the priority they should have.



References: Vista Multimedia Class Scheduler (MSDN)

(http://msdn.microsoft.com/en-us/library/ms684247(VS.85).aspx)

In the Enterprise

Process Lasso can be used in network client/server environments in several ways. The fact that the core engine is seperate from the GUI makes it easy to install without it being visible to the end user. You can even run the core engine as a system service and perform silent installs, feature many network admins will appreciate.

Methods of Deployment

The big question is whether you want to use a centralized network configuration and log store, a per-machine store, or a per-user store.

Option 1: Centralized configuration and log

In this mode, all workstation are configured to use the same configuration file and log folder. These can be on network shares, or can be pushed out to workstations through alternate methods.

Option 2: Per-workstation configuration file and log

In this mode, you'd install Process Lasso on each workstation and use a common configuration and log path that every user of the workstation would access.

Option 3: Per-user configuration file and log

In this mode, you'd install Process Lasso on each workstation and let each user have their own configuration and log. This is the default behavior of Process Lasso. If you don't need or want to make changes to the configuration or modify the log files, this is a good way to run Process Lasso.

Option 4: Some combination of the above, or an alternate method

In the end, you can utilize Process Lasso just about however you want. You could have workstations launch it from a network share, for instance, without ever having to install Process Lasso on those workstations. The prerequisites to running Process Lasso are very few. A single COM control (ProcessControl.dll) should be registered, and older systems (XP before SP2) will require Microsoft's GDIPLUS.DLL to be installed before the Process Lasso GUI can run. However, the core engine does not require the GDIPLUS library to be present.

Service Mode

You can install Process Lasso's core engine, processgovernor, as a service. This option is presented to you during the install process of Process Lasso Pro. It is also available during silent installs using the command line switches.

Running the core engine as a service has its advantages as disadvantages. There are a few minor caveats to running the core engine as a service, so be aware of them.

Silent Installation

Process Lasso (as of v3.53) now supports completely silent installation. For more information, see the command line switches documentation.

More help?

If you have any questions about getting Process Lasso up and running in your enterprise, please email us at support@bitsum.com. Customized solutions are also available.

Clearing up Misconceptions

There has been some understandable user confusion about Process Lasso, so first let's clear up a few common misconceptions. If your question isn't answered here, then see the FAQ below.

Misconception #1: Process Lasso is yet another super-charged task manager

> NO! Process Lasso is NOT a task manager at all. It has some task management capabilities, but its not meant to replace a task manager. Its meant to act in the background, automatically enforcing rules upon processes in order to improve your system responsiveness (or for whatever other purpose you desire).

Misconception #2: A user needs to set default priorities on their processes to rank their importance.

> NO, PLEASE DON'T DO THIS. The idea isn't to reprioritize all your processes, rating them in importance to you. Doing that is risky and harmful to your PC performance. Just let Process Lasso do its job, UNLESS you know what you are doing.

Misconception #3: Process Lasso is only for older computers

> It may be true that older computers get the most benefit from Process Lasso, but Process Lasso is helpful to even brand new 'top of the line' PCs. Additionally, its extra features are quite handy for anyone (such as anti-sleep processes and High Performance Power processes). Those with a lot of CPU cores may also enjoy the default CPU affinty feature, where they can limit programs to specific CPUs - keeping the others free for use by other procesess.

Frequently Asked Questions (FAQ)

Is the free build completely free?

Yes, home and academic users can freely use it for as long as they want. However, after the product has been installed a while, a couple features are disabled and some nags are shown to encourage the user to upgrade to Process Lasso Pro. These nags are tolerable, as they are meant to be reminders, not to annoy. Even after this point, the software is still quite usable and not by any means crippled. These actions help us to continue to publish a free edition. We encourage you to give Process Lasso a try. It may be just what you need, and the Pro build may be well worth the price of a couple cheese burgers.

What is the difference between the free and Pro versions of Process Lasso?

The differences are described here.

The Pro build has many benefits over the free build. The table below lists the differences between the two builds.

Feature	Process Lasso Free	Process Lasso Pro
ProBalance	YES	YES
Gaming mode	YES	YES
Persistent priorities	YES	YES
Persistent CPU affinities	YES	YES
Emergency stall recovery *new*	TRIAL PERIOD ONLY*	YES
Instance count limits	TRIAL PERIOD ONLY*	YES
Automatic gaming mode	TRIAL PERIOD ONLY*	YES
CPU throttling	TRIAL PERIOD ONLY*	YES
Anti-sleep processes	TRIAL PERIOD ONLY*	YES
Keep processes running	TRIAL PERIOD ONLY*	YES
High performance power scheme processes	TRIAL PERIOD ONLY*	YES
Core engine as service	--	YES
TweakScheduler tool	--	YES
MM Scheduler tool	--	YES
Access to beta versions	YES	YES
Access to older versions	--	YES

*The trial period may vary, but is typically 7 to 14 days.
**Other unlisted features may also be disabled in the free build.

What exactly is a 'process'?

When you run a program on your computer, it creates a process. Every program you have running is represented by a process. When you have multiple copies of a program running, it has multiple processes. For instance, if you were to run Microsoft Word, it would create a process named 'msword.exe'. Each process has its own memory and threads (you can think of threads as tasks for your CPU to execute). Furthermore, each process is isolated from other processes, so that a crash in one program won't cause a crash in another.

What do the various Virtual Memory metrics indicate?

Ah, the ever-so-confusing virtual memory defintions. Even programmers are sometimes confused by them. Here I'll try to clear them up. I haven't much time, so its just a quick explanation. You will need to know the basics of virtual memory to understand it.

Virtual memory is an abstraction layer, allowing what you (and processes) think of as 'memory' to be stored in any combination of phsyical media and shared between processes. It has numerous benefits. For one, it lets you have a very large amount of memory, since virtual memory is limited only by the space available on the page files' storage medium (hard drives or SSDs). Secondly, it is easily shared since duplicated memory can simply 'point' to the first copy of it. There are many sother benefits as well.

When virtual memory is accessed (a hard page fault), it is moved into RAM (page in or swapped in). When it is unused for a longe period, it is moved to the SWAP file (paged out). All this is transparent to the process. The process simply accesses memory at address X, and the OS makes sure it is in RAM (if not already). Ideally, you WANT all frequently accessed virtual memory to be in RAM, so it doesn't have to be read from the much slower page file.

Commit Charge can be best defined as the amount of virtual memory committed to that application (whether it resides in RAM on the page file). In contrast, the 'Working set' is the amount of memory actively in RAM, excluding the portion swapped out to the page file.

Working Set is the amount of virtual memory actively in RAM, excluding the portion swapped out to the page file.

Private Bytes is the amount of virtual memory in the process space that is not shared from some other process. When virtual memory pages are duplicated, they can be shared between different processes in memory.

Private Working Set is the amount of RAM used exclusively by this process. This value is normally equal to the Commit Charge, but could vary if the process is sharing a lot of its memory with another process (i.e. another instance of itself). Google Chrome is an example where this will vary a lot. This metric requires more CPU cycles to compute, and is therefore not recommended to use unless you absolutely must know it. You will see a warning when enabling it.

Total Virtual Memory Usage is the sum of all virtual memory mapped into a process's address space. This value will seem huge, but it includes stack space and other reserved virtual memory that isn't used. Most users should ignore this field.

I do not see some of my processes, why not?

Process Lasso ignores processes it has no access to. In Windows Vista+ (including Windows 7), this means you must set Process Lasso to run with 'Highest' rights. You can do this through the menu option 'Options / General Process Lasso Settings / Reconfigure the way Process Lasso starts ...'. There are some caveats to running Process Lasso in this mode, so you may see some warnings. However, you will see ALL processes, and it will manage ALL processes when in this mode.

How do get Process Lasso to manage the processes of all users currently logged in, not just my own?

During the install, you are asked if you want to run Process Lasso with 'highest' rights. This is in the

last configuration dialog. Simply check this box during install and any instance of Process Lasso will manage the processes of all users. However, this is not recommended unless you are running the core engine as a service. Process Lasso is designed to have an instance of the core engine running in each active user context, managing the processes for that user. This means Bob will have an instance of the core engine (processgovernor.exe) managing his processes, and Jane will have an instance of the core engine managing her processes. Also note that by default each user also has his or her own configuration and log, but that can be changed during install.

How do I get Process Lasso to start with HIGHEST rights in Vista+, but with NO UAC prompt?

Start the Windows 'Task Scheduler'. Create a new task to launch ProcessLasso.exe (e.g. "c:\program files\Process Lasso\ProcessLasso.exe"). Set the trigger for user login, then check the 'Run as administrator' box in the properties (there is a check box to show the properties dialog at the end of the task creation wizard). That is all there is to it! A separate task for the core engine is NOT necessary.

UPDATE: Process Lasso v3.99.3 alpha now properly schedules itself as a task to be launched at user login with highest rights (when the user configures it to do so). This new capability will go final in version 4.

If I stop Process Lasso Core Engine, what does it mean?

The 'core engine' (processgovernor.exe) is what actually 'does the work'. It applies ALL automated process rules, including ProBalance, default priorities, default affinities, disallowed processes.. everything. So, you can use the GUI (processlasso.exe) to create/edit process and settings, then close it completely when not needed. This minimizes resource usage. The core engine is silent though (not even a system tray icon), but it does write all its actions to a log you can view later with the GUI.

What is the difference between stopping the Core Engine and shutting down Process Lasso?

When shutting down Process Lasso, you are asked if you also want to stop the core engine. So, shutting down allows you to either close only the GUI, or both the GUI and core engine.

Is Gaming Mode recommended?

For Gaming and similar tasks it can be useful, but YMMV. This puts your PC in a high performance power scheme and tweaks ProBalance settings to help give the foreground process as much attention as it can. Now, it also does a temporary foreground boost, but only to Above Normal priority class. For this reason, I recommend simply trying it out. It may work great to avoid micro-lags, but that isn't guaranteed as interoperability issues with the priority class change could theoretically exist in rare cases. If it doesn't work, it won't cause any big problems.. For the vast majority for which it does work, it is just another little optimization to improve their PC gaming experience ;).

Does Process Lasso tame the Memory Usage of the various processes or does it tame the CPU Usage of the various processes?

Process Lasso's ProBalance doesn't tame anything, it just reprioritizes the importance of the processes (the threads of them) in the view of the Windows CPU scheduler. This allows it to give CPU cycles to preferred threads in periods of high CPU contention.

Virtual memory usage is NEVER changed/tweaked/adjusted. Now, for Vista and Windows 7, the SuperFectch priority of memory pages can be partially derived from the process priority class. Therefore, in an indirect way I suppose it might also aid that.. lol. That'd be a huge market-sleeze type claim there, and I won't make that one ;).

Process Lasso does have optional features to do things similar to those 'memory cleaners', but will see strong advice agaisnt utilizing them.

How do I assign a shortcut key combination to show the Process Lasso main window?

While there isn't a built in shortcut key to show the Process Lasso main window, you can easily add one yourself. To do this, right click on the Process Lasso start menu shortcut and select 'Properties'. There is a field 'shortcut key'. Using that field, you can assign Process Lasso a shortcut key (e.g. CTRL+ALT+K). Now when you press that shortcut combination, you'll see Process Lasso's main window pop up.

Why isn't Process Lasso's ProBalance acting on the processes associated with my anti-virus software?

Since anti-virus software processes are sensitive to priority adjustments, many common anti-virus softwares are excluded from restraint. Consider the real-time scanner of your anti-virus software. Whenever a file is opened, the process opening the file must WAIT for the scanning to complete. So, you don't want to lower the priority of the real-time scanner, as that would not make things happen quicker. In fact, it would have just the opposite effect. Other software isn't as careful as Process Lasso, but we wanted to **DO THINGS RIGHT!** .. even if it means sometimes people think that the product isn't not working because it skips a process that is using a lot of CPU.

If your anti-virus software isn't on our hard-coded exclusion list, that's no big deal because it probably would never get acted on anyway. However, if you see Process Lasso's ProBalance acting upon it, then you may want to exclude it from restraint. Most of the time, this isn't an issue since ProBalance would only act on a non-excluded critical process of anti-virus software under rare conditions. In the majority of cases, the default settings themselves implicitly exclude critical anti-virus processes.

Will Process Lasso's ProBalance adversely interefere with any of my progams?

No. All dynamic adjustments Process Lasso makes are completely safe and have virtually no possibility of severe complications. Additionally, after years of real-world testing, we've identified and fixed many specific interoperability issues that could have affected performance. Process Lasso is a very mature and well tested product.

What is the white line I sometimes see on the graph?

This line represents the CPU history of the process(es) you have currently selected. Yes, this is not indicated on the graph legend.

Process Lasso does not seem to take any action to 'restrain' overly active processes. What is going on?

By default, Process Lasso excludes the foreground process (the one that has keyboard and mouse focus). You should click to another window to allow Process Lasso to restrain the process. Alternatively, you can turn off exclusion of foreground processes. The occasional restraint of foreground processes won't hurt anything since any active background processes will also be restrained at the same time. Of course, by restraint we simply mean temporarily lower the priority class, not any sort of actual limit on CPU usage (see ProBalance docs).

Is Process Lasso recommended for gaming?

Yes, it will help improve the responsiveness of your games and reduce the occurances of lags and freezes. Older versions of Process Lasso required you to manually exclude your game processes from ProBalance restraint, but newer versions will not ever restrain foreground processes (by default).

Does Process Lasso's ProBalance out-of-control restraint slow down processes?

No. Process Lasso 'restrains' processes by temporarily lowering their priority. This simply allows other processes more of a chance to use the CPU, **IF** there are any processes needing the CPU. If there aren't, and until there aren't, the restrained process is still able to consume as many CPU cycles as are available to it. Therefore, a restrained process doesn't really slow down, though it can now yield to another process like a nice citizen of your computer. That little yield will make a big difference in responsiveness, but not a big difference in the speed of the background process ;).

For more information, read page Process Lasso's Process Balance Technology (ProBalance)

How will ProBalance affect CD/DVD Optical burners?

CD/DVD Burners typically don't use enough of the processor to be restrained, as they are usually waiting on disk or optical drive I/O. It is actually more likely that a buffer underrun would be prevented by Process Lasso because if you were in such a high-load situation, the other offending processes would have their priorities lowered, there-by giving the burning application more access to CPU cycles. Even if the burning process itself got lowered in priority, it'd be no worse off than without Process Lasso since the other active processes would surely have been lowered as well, there-by giving it equal footing, as it had originally.

Also, some burners raise their own priority and Process Lasso will NOT lower the priority of processes that have raised their own priorities, depending on the configuration of Process Lasso.

The bottom line is: Process Lasso is most likely to HELP, and even in a worst case scenario it shouldn't make things worse. That said, adding Optical Burning apps to the Process Restraint Exclusions list is probably a good idea, but far from necessary.

Process Lasso was designed to, above all, 'do no harm'.

Should I turn on foreground boosting?

Windows already does foreground boosting by giving foreground threads 3x longer time slices than background threads [depending on Windows version and configuration] and slightly increasing the thread priority. Process Lasso is capable of doing additional foreground boosting is a 'smart' way that compliments this built in mechanism. However, unless you require it, we generally don't recommend this option be on because the foreground process/thread is probably already boosted as much as will be effective by Windows. Additional boosting probably won't make a difference for most people, but it can in some situations. You can play with foreground thread and/or process boosting and see if you like it, as it should cause no harm -- but just may not benefit as much as you'd hope. The background process 'restraint' is more important function of Process Lasso and the proper way to handle situations where background processes are interfering with the foreground thread, which is already boosted in priority by Windows.

What is the difference between *foreground priority class* and *foreground thread boosting?*

Foreground priority class boosts the priority class of the entire process (aka application, or program) that is in the foreground (meaning it has the focus of the keyboard and/or mouse). In contrast, foreground thread boosting only boosts the single thread in that in the foreground. A layman could consider a thread as a piece of a program, instead of the entire program. Its brother and sister threads in the same process are not boosted.

This other program that does something similar seemed to act when Process Lasso didn't, or vice-versa. What's the deal?

Automatically managing process/thread priorities **without** interferring with system and application operation, and **effectively** increasing system responsiveness is a delicate business. It could be done in a dumb sort of way that 'just does it'. That's not Process Lasso. Process Lasso was designed to interfere as little as possible with the Windows scheduler. In fact, it is designed to compliment and work along-side the Windows scheduler. Unfortunately, many other products that have a less sophisticated approach may not be so elegant. However, in some cases alternate products may be desireable. Our product is free. We invite you to find what's right for you.

Why don't you recommend using the trim virtual memory function?

RAM is your fastest storage medium. You generally want to keep as much data in it as possible, not force pages out to the page file so you have more 'free' RAM for the system cache or new application launches. When you force pages out, they simply have to be reloaded again later as page faults occur on them (see other answer for explanation of page faults). For more information:

The Truth about Windows Memory Optimizers

Bitsum Technologies
Home of Process Lasso
http://www.bitsum.com

Memory boosters, optimizers, and washers --- whatever the name, they all do the same thing: free up physical RAM. They accomplish this by forcing RAM to be 'paged out'. That means, the memory is taken from RAM and put into the page file, which exists on the hard drive. Does this make your computer run faster? The short answer is **No**. In fact, just the opposite is usually the case.

What is virtual memory?

You must first understand what virtual memory is. All modern operating systems utilize it. Virtual memory is essentially memory that can exist either in RAM, or on the hard drive or other storage medium. That is why it is *virtual* memory, because it can be backed by any storage medium, not just RAM. This allows for your system to use an almost infinite amount of memory, by swapping pages between the page file(s) and RAM. When memory isn't being used, and RAM space is needed by other processes, that RAM is paged out to the page file (or swap partition in the case of many non-Windows operating systems - but we'll refer only to the page file since we're talking most about Windows). Windows applications only see virtual memory. They have no concept of how much free RAM is available, nor should they care. They don't know whether memory they've allocated is currently stored on disk or in RAM. The Windows operating system transparently handles all this.

When virtual memory is accessed, the system does a quick check to see if that page of memory is already in RAM. If it isn't, then a hard page fault occurs and that memory is loaded from the page file (or swap partition, etc..) back into RAM. This operation has a great amount of overhead since your hard drive must seek to the right location in the page file and read the necessary page(s). In general you want to reduce the number of hard page faults as much as possible. There are also soft page faults, which incur less overhead than hard page faults, but I won't get into detail to avoid reader confusion. A soft page fault is simply where the needed page of memory is already located somewhere in RAM, and can therefore be mapped into the accessing process without having to read from the page file.

Virtual memory is organized in pages. A page is a block of a few kilobytes of memory (typically, 4KB). The process of moving a page of memory from RAM to the page file is calling 'paging out', or 'swapping out'. Conversely, the process of moving a page of memory from the page file to RAM is called 'paging in' or 'swapping in'.

Windows manages virtual memory so that commonly used pages are attempted to be kept in RAM and less commonly used pages are stored in the page file. Since RAM is your computer's fastest storage medium, you want to make maximum use of it. You WANT to keep it full. In fact, the RAM that isn't used by your running programs is usually filled with cache data. Windows attempts to always make as much use of the RAM as possible.

Available virtual memory is limited only by the size of the page file(s). In this way, the system can use gigabytes of memory even if the RAM is only a few hundred megabytes. Often, the page file is configured to dynamically grow when needed. In such cases, available virtual memory is only limited by the free space on the partition(s) the page file(s) exists.

How memory optimizers work

Memory optimizers temporarily force all possible pages in RAM to the page file. Thus, the amount of free RAM is increased, but the amount of virtual memory in use is not affected. Later (or immediately), when the applications whose memory was paged out access that memory, it must be paged back into RAM -- incurring substantial overhead.

The ***only*** benefit from these applications is that if you were to load a program immediately after you've paged out all available memory, it ***might*** load faster in some cases because pages of RAM are already available and don't need to be paged out in order to make room for the program. One

way to look at this scenario is that the cost of freeing RAM is done before the program loads instead of as the program loads. However, this benefit is negligible and is more than paid for later when the RAM that was paged out must be paged back in.

The memory that is paged out must be paged back in for you to use the other applications running on your computer. Thus, when you click on a minimized application, for example, it takes longer to restore its window because more of its memory must be paged in. Furthermore, pages that are used by the operating system components or background processes are often paged back in immediately after the memory 'cleanup'.

These memory optimization applications interfere with the ability of Windows to efficiently manage virtual memory. Furthermore, many of them don't even free RAM in a proper way. Often, they do so by simply allocating as much RAM as possible, forcing Windows to page out the memory of all other applications. The correct technique is to use the SetProcessWorkingSetSize API on each running process to force as much virtual memory to be paged out as possible.

These programs are ridiculously easy to write and sell. They usually don't live up to the claims they make. Savvy users should beware these applications, and for that matter, any other software that makes outlandish claims.

Common myths

There are many misconceptions propagated by the authors of these applications and adopted by users who don't know any better. Here are a few:

Myth	Truth
Memory optimizers increase overall performance	False, as noted above in this article. They often decrease overall performance. Paging out all possible memory just forces it to be reloaded once it is needed again.
Programs crash because of lack of physical memory (RAM).	Programs only have access to virtual memory (exists in RAM as well as page file). They don't actually ever directly allocate physical memory. A crash due to the inability to allocate virtual memory is extremely rare. Regardless, these applications do not increase the amount of free virtual memory and therefore have no impact on programs that crash as a result of the inability to allocate memory.
RAM must be defragmented	This is completely false and quite absurd. Hard drives need defragmenting because the read head must seek the proper location on the disk to read from a file. If the file is located in pieces all over the disk, that means more seek time. In the case of RAM, there is no seek time, so defragmentation is absolutely pointless.
Unused DLLs are unloaded	This is simply false. When a program terminates, all DLLs it used are unmapped (as is its entire process space). Upon program crash, these DLLs may not be notified the program is closing, but they will still be deallocated from the crashed program's address space. A shared DLL may still be in use by other programs and be mapped into their process spaces, but that has nothing to do with the issue. When a DLL is no longer mapped into any process space, it ceases to exist. DLLs can not exist by themselves in virtual memory. They must be mapped into some process space.

Other unused resources and memory are unloaded.	This is simply false too. Nothing is left in memory when a program crashes and these memory washers have no way of knowing what a currently running program needs and doesn't need.

Summary

It is best to leave the management of virtual memory to Windows and not take it into your own hands. Any benefit from these programs is an illusion and temporary. You incur the penalties of paging memory back in sooner or later, even if it appears that the program you run after the cleanup loads faster.

If you decide on using one of these programs because your needs do actually warrant it, at least find one that pages out memory in a proper way (SetProcessWorkingSetSize API) instead of allocating gobs of memory to itself. I do not know which of these applications do it properly and which don't, but the less gimmicky they look, the better they probably are. An application like this written by a professional at least stands a better chance of performing its tasks in an optimal way and causing less performance degradation.

Jeremy Collake
Bitsum Technologies
http://www.bitsum.com

What does the system responsiveness metric indicate?

It represents the ability of the thread messages subsystem to keep up with thread/window message demand. The exact way we calculate this we don't publish, but its a pretty simple and accurate measurement.

What does it mean when the system tray icon changes to the traffic light?

If your settings are the default, that means a process priority was automatically lowered via the process restraint mechanism.

What are the highlighted portions of the graph?

These are times when a process's priority was temporarily lowered to improve system responsiveness. Seeing the highlight helps visualize the effects on system responsiveness, as shown by the green line on the graph.

What is 'More strictly enforce default priorities and affinities', aka 'forced mode'?

Sometimes applications like to control their own priorities. Process Lasso doesn't try to fight with them, instead only making a single attempt to set the default priority and/or affinity for the process. If you want Process Lasso to aggressively and forcibly enforce your default priorities and affinities, forced mode will do that. If you have a process that isn't staying at the default priority or affinity,

you can use this option. It should be noted though that application that set their own priorities or affinities may be doing so for a good reason.

What is this 'thread boost' or 'Windows dynamic thread priority boosting' in the process priority setting?

When this is enabled for a process, Windows temporarily boosts the priority of threads for that process under certain conditions (when it leaves a wait state). For specific information, see this MSDN page about Windows dynamic thread priority boosting (http://msdn.microsoft.com/en-us/library/ms684828%28VS.85%29.aspx). Turning this off can severely impact the performance of applications that have a graphical user interface (a window). For background applications, this setting is not usually important.

What is the '(*)' shown after priorities in the process list?

That indicates whether or not Windows dynamic thread priority boosting is enabled for that process (see other question about thread priority boosting for explanation).

What about I/O prioritization?

Vista's I/O prioritization is based on the priority of the thread doing the I/O, which is dependent on the priority class of the process doing the I/O. Therefore, adjustment of the thread priorities will propagate to the I/O of that thread. Specific I/O prioritization is not supported.

Process Lasso isn't minimizing when it is start at login, what is wrong?

This is probably due to the '/tray' parameter getting lost from the startup entry in the registry. The easiest way to fix it is to reinstall Process Lasso (no need to uninstall first, so your settings should be kept).

What about virtual memory prioritization?

Vista and above tag virtual memory pages with a priority, assigned by ReadyBoost. These priorities can be read, but are less easily adjusted. We are implementing a way to work with them, but we generally recommend users stay away from messing with default memory priorities. Remember, Windows and ReadyBoost are designed to work as optimal as possible. Too much tweaking can make matters worse. Process Lasso does a great job as is, so maybe you don't really need this feature.. though we will add it eventually.

I purchased the product, but am still seeing nags. What is wrong?

The Pro build is a completely separate download from the free build. Therefore, you likely mistakenly reinstalled the free version. You must be sure to install the copy you download from our User Services area.

I added critical system processes to the 'disallowed proceses' list and now my I've got problems. What do I do?

You need to delete your Process Lasso settings (located in the INI file). If you can't get to it, and can't use the GUI, try booting into safe mode. You can see where it is located at in the about box of Process Lasso. Uninstalling Process Lasso will also allow you to recover from this situation.

Process Lasso never stays set to start at boot. What is wrong?

Every report I've heard of this is caused by third-party software that monitors new additions to your start-at-login registry keys. The most common culprit has been WinPatrol, which some users reported didn't give them the indication they expected when it blocked Process Lasso. If you install Process Lasso, but then it doesn't start at boot, and you go check the settings only to see they say 'Do not start at login', then SOME third-party software deleted the registry value. Please double check any software that watches your startup entries.

Are there known processes Process Lasso works well with?

Yes, please see our case studies (under construction).

Does Process Lasso work with Windows 7?

Yes, Process Lasso is designed to work with Windows Vista and Windows 7. It will have the same benefits as it does under XP. Furthermore, Process Lasso has some specific features that compliment the Vista and Windows 7 CPU and I/O scheduler enhancements.

How can I run the core engine as a service under a specific user account in Windows Vista Home Edition? It doesn't include secpol.msc.

Since Vista Home and Starter editions don't include the local security policy management console snap-in, you must manually assign the SeServiceLogonRight token to user(s). You can do this by using Microsoft's NTRIGHTS (http://support.microsoft.com/kb/279664) console mode utility. Download (http://www.microsoft.com/downloads/details.aspx?FamilyID=9D467A69-57FF-4AE7-96EE-B18C4790CFFD&displaylang=en). More info (http://support.microsoft.com/kb/279664). Open an administrative console and run 'NTRIGHTS -u myusername +r SeServiceLogonRight'. From this point forward, the user you specified will have the right to logon as a service.

Alternatively, you could chose to run the Process Lasso core engine as a *system* service, instead of a service under a specific user account.

For most other editions of Windows, run 'secpol.msc' and edit "*Local Security Policies \ User Rights Assignment \ Logon as a service*".

Can I do some further reading on how the Windows CPU scheduler works?

Sure, here are some links, and I'll add more as time allows. These are all third party links and articles.

- Wikipedia: Thread scheduling (general, technical)
  (http://en.wikipedia.org/wiki/Scheduling_(computing))
- MSDN: Thread scheduling priorities
  (http://msdn.microsoft.com/en-us/library/ms685100.aspx)
- Everything you never wanted to know about OS multi-tasking (and why out of control processes need priority adjustment)

  (http://www.tmurgent.com/WhitePapers%5CSchedulingWP.pdf)

Credits

Process Lasso was written by Jeremy Collake. Additional credits are to:

- Some artwork by Nicolas Nguyen
- XMessageBox code by Hans Dietrich
- Testers and especially supportive users: Scott Danesi
- Translators of localized builds

About the author

Jeremy Collake has over 17 years programming experience. He specializes in low-level systems programming and compression algorithms. He has dedicated his career to contributing to the world in the best way he can.

Licensing

Process Lasso is free for HOME and ACADEMIC use, but users can purchase the Pro version for additional functionality. Commercial users must purchase Process Lasso Pro within 7 days of use.

- Home users:
  (http://www.prolasso.com/prolasso-donate.php)
- Commercial users:

  (http://www.prolasso.com/prolasso_commercial.php)