NightStar RT Installation Guide

Version 4.3

(RedHawk™ Linux®)

concurrent

0898008-4.3
February 2013

Contents

1.0. Introduction

NightStar RT Version 4.3 is a production release of the NightStar Tools running under RedHawk Linux.

This release includes major releases of NightTrace and NightView.

NightStar RT Version 4.3 also includes all product updates that have been released since the 4.2 CD was created.

NightStar RT consists of the NightProbe data monitor, NightSim application scheduler, NightTrace event analyzer, NightTune system and application tuner, NightView source-level debugger, Datamon data monitoring API, and Shmdefine shared memory utility.

This release also includes enhanced support for debugging and analyzing 32-bit applications on a 64-bit platform. See “Architecture Interoperability” on page 29 for more details.

If you haven’t used NightStar RT since version 3.2, you’ll notice a new look and feel to the graphical interface components. Read “NightStar RT GUI Features” on page 7 to get an overview of the new interface.

1.1. New Features In NightView

The following new features were added to NightView 7.4, which is part of this release.

- Added support for debugging 32-bit programs on 64-bit systems.
- Added Smart Printing support which allows users to customize how variables of complex types are shown in the debugger.
- Added Branch Tracking which retains a list of PC where branches occurred on a per-thread basis. This feature requires RedHawk 6.0 or higher and reasonably current Intel-based hardware.
- Added support for debugging CUDA code executed by an NVIDIA GPU.

1.2. New Features In NightTrace

The following new features were added to NightTrace 7.3, which is a part of this release.

- Extended the range of trace ID numbers to a much larger number space; this is especially useful when using Application Illumination on large amounts of code.
- Added support for analyzing and capturing 32-bit trace data on 64 bit systems.
- Allow analysis of data captured from multiple systems w/o RCIM synchronization. The systems must be synchronized by some means; typically PTP or NTP.
- Search and Summary panels are now dialogs, providing less distraction to timeline and event analysis.
- Added support for Application Illumination function calls summaries in the graphical interface as well as in batch mode. You can obtain a summary of all function calls including their count, their minimum, maximum, and average duration. You can also obtain a detailed table of all the calls for a single function.
- Formalized the Auxiliary API which allows you to translate any data into NightTrace format for subsequent analysis with **`ntrace`**; see **`/usr/include/ntrace_aux.h`**.
- Added the ability to graph condition gaps.

2.0. Software Installation

Follow the instructions under “Installing NightStar RT” on page 2 to install NightStar RT on your system.

Then take a look at “NightStar RT GUI Features” on page 7 to learn about the new graphical interface of the NightStar RT tools.

2.1. Prerequisites

Prerequisites for NightStar RT Version 4.3 for both the host system and target system are as follows:

2.1.1. Host System

Any of the following distributions:

- Red Hat Enterprise 3.0
- Red Hat Enterprise 4.0
- Red Hat Enterprise 5.0
- Red Hat Enterprise 6.0
- RedHawk 2.1 or higher

Any Intel Pentium Xeon or Intel EMT64 or AMD Opteron supported by the operating system distribution.

2.1.2. Target System

Any of the following distributions:

- RedHawk 2.1 or higher

Any Intel Pentium Xeon or Intel EMT64 or AMD Opteron system supported by RedHawk.

2.2. Installing NightStar RT

There are two methods of installing NightStar RT.

- Network Installation
- CD Installation

Network installation is highly recommended as an alternative to using the CD image, because it includes configuration of the YUM repository for NightStar RT Version 4.3 and allows you to keep your system up to date with updates.

If you have NightStar icons on your desktop, remove them before proceeding with the installation. After installation is complete, reinstall the icons using the following command:

```
/usr/lib/NightStar/bin/install_icons
```

2.2.1. Network Installation

Network installation is accomplished using NUU, Concurrent's Network Update and installation Utility.

If you do not have NUU installed on your system, visit the following web site and follow the instructions on installing and configuring NUU and installing this release:

http://redhawk.ccur.com/updates

Once NUU is installed, you can install or upgrade to this release by invoking NUU as follows:

```
/usr/bin/nuu --enablerepo=ccur-nstar-rt
```

The QuickStart.pdf document included in the NUU download kit obtained from the URL above explains how to use NUU to install and update products. The document is also available on-line on the web page referenced by the URL above.

After installation, you can keep your system up to date interactively using NUU or even install a **cron** job to update your system nightly; e.g.:

```
1 0 * * * /usr/bin/yum -y --disablerepo=* --enablerepo=ccur-nstar-rt
update
```

2.2.1.1. Advanced Information

NUU is a graphical interface which uses the YUM and RPM subsystems to install and update software which is provided via a software repository.

Concurrent provides software repositories for many of its commercial products, including NightStar RT.

Any YUM-compatible client may be used to access the repositories.

If you installed and configured NUU via the instructions mentioned above, the NightStar RT repository is already configured and activated on your system.

However, if you prefer to use an alternative YUM client, you will need the following information:

The URL for the NightStar RT repository is:

https://redhawk.ccur.com/buffet?Login=*login***&Password=***password***&path=NightStar/RT/RedHawk/$basearch**

where *login* and *password* should be replaced with your login and password assigned to you for the redhawk.ccur.com site. (Note: there are no spaces in the URL above, but your document reader may make it appear as if there were).

2.2.2. CD Image Installation

To install NightStar RT using the *NightStar RT Installation CD*:

- Insert the *NightStar RT Installation CD* in the CD-ROM drive.
- If the CD does not auto-mount, mount the CD-ROM drive (assuming the standard mount entry for the CD-ROM device is **/media/cdrom** in **/etc/fstab**)

```
mount /media/cdrom
```

- Double-click on the CD icon including the notation NightStar RT on your desktop.
- Double-click on the icon labeled Launch Install Script.

NOTE

If you are not using a file browser to access the CD, change the current working directory to **/media/cdrom** and invoke the **./install-nstar** script.

IMPORTANT

If attempts to invoke **./install-nstar** are unsuccessful and generate a message similar to

```
unable to exec...
```

or if the Launch Install Script immediately exits without useful information, it may be that your system has been configured to prevent execution of scripts from mounted CDs. If this is the case, the mount options would have included the **noexec** option. You can correct this problem by executing the following command as the `root` user (substituting the actual mount point for **/media/cdrom**):

```
mount -o exec,remount /media/cdrom
```

NOTE

The installation step above installs the entire NightStar product on your system. For embedded systems, you may only want to install the server-side portion of NightStar and do all GUI operations from another system targeting your embedded system. To install the server-side portion only, change the current working directory to **/media/cdrom** and invoke the **./install-nstar-server** script instead.

- Double-click on the icon labeled Install Desktop Icons.

NOTE

Since you are running as root, these icons will only be installed on root's desktop. To install these on your normal user's desktop, run the following script when logged on as your normal user:

```
/usr/lib/NightStar/bin/install_icons
```

2.3. Obtaining License Keys

The NightStar RT Version 4.3 software uses the same license manager and license features as the production version of NightStar RT. If you already are using NightStar RT, you do not need to obtain new licenses.

Permanent License Keys

- If you have purchased NightStar RT, you can obtain your permanent license keys at the following URL:

  http://real-time.ccur.com/NightStarRTKeys

  You will need your site ID, your email address, and your system identification number which was displayed during product installation. You can obtain that number again by running the following command on the system where the license keys will be installed:

  ```
  /usr/bin/nslm_admin --code
  ```

See “NightStar RT Licensing” on page 23 for more detailed information about the NightStar License Manager (NSLM).

2.4. Removing NightStar RT

To remove NightStar RT, mount the CD or ISO image as described in “CD Image Installation” on page 3 and execute the following command as root:

```
./nstar-uninstall
```

3.0. Documentation

The following table lists the NightStar RT 4.3 documentation available from Concurrent.

NightStar RT Version 4.3 Documentation

Manual Name	Pub. Number
NightStar RT Installation Guide (Version 4.3)	0898008-4.3
NightStar RT Tutorial (Version 4.3)	0898009-080
NightProbe User's Guide (Version 4.2)	0898465-108
NightSim User's Guide (Version 4.3)	0898480-075
NightTrace User's Guide (Version 7.3)	0898398-180
NightTune User's Guide (Version 3.5)	0898515-043
NightView User's Guide (Version 7.4)	0898395-350
Data Monitoring Reference Manual	0898493-022
Quick Reference for shmdefine	0898010-050

Additionally, the manuals are available:

- in PDF format in the **`documentation`** directory of the *NightStar RT Installation CD*
- on the Concurrent Computer Corporation web site at http://redhawk.ccur.com/docs

and, after installation:

- online using the NightStar integrated HTML viewer **`/usr/bin/nhelp`**
- in PDF format in the directory **`/usr/share/doc/NightStar/pdf`**
- in HTML format in the directory **`/usr/share/doc/NightStar/html`**

4.0. NightStar RT GUI Features

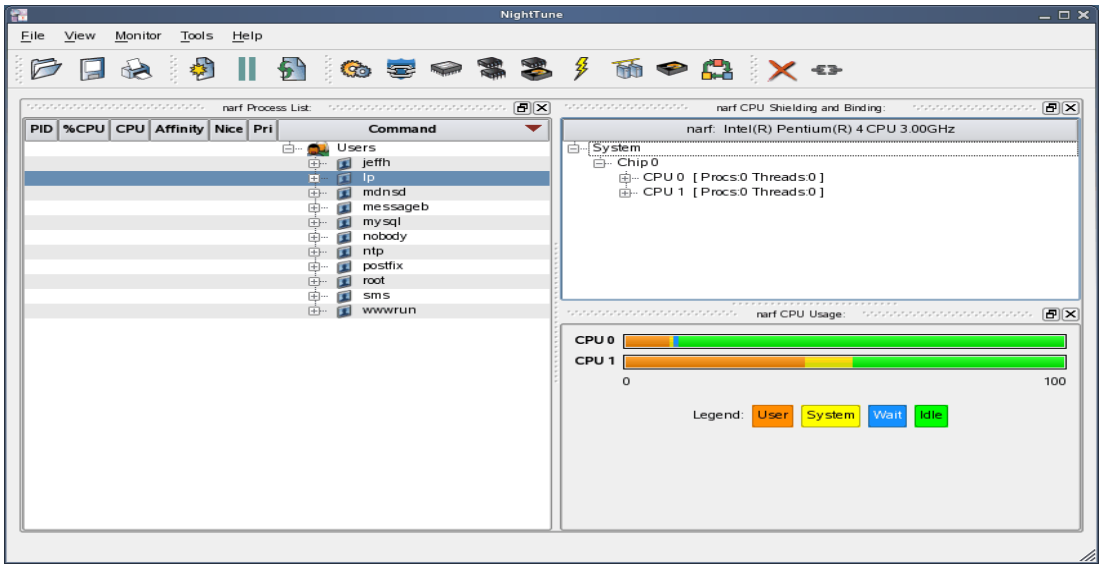
Some of the common features of the NightStar RT Tools graphical user interface include:

- movable and resizable panels
- tabbed pages
- context menus

4.1. Movable and Resizable Panels

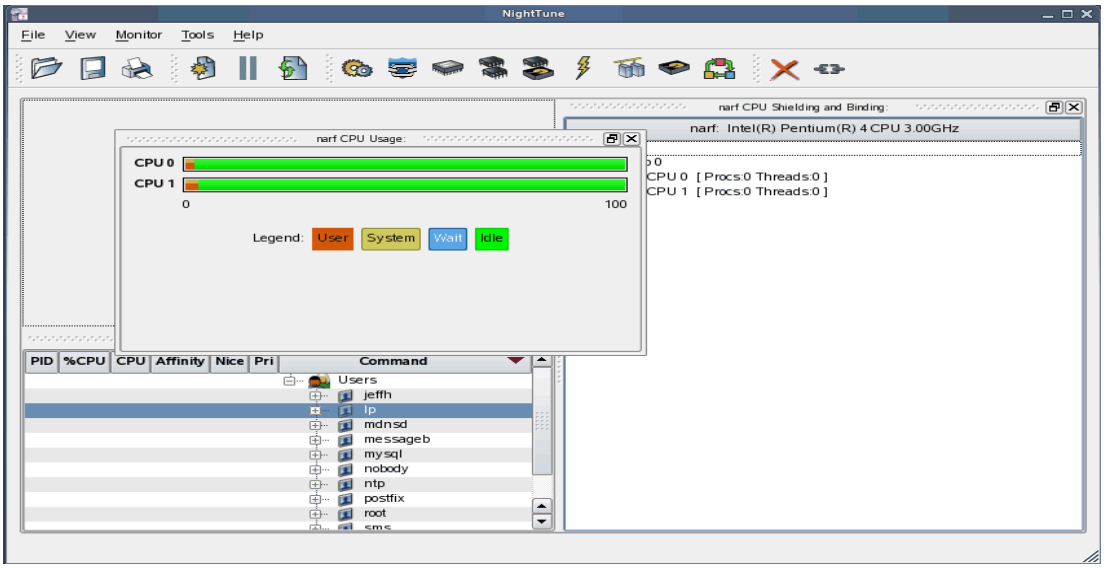
The NightStar RT Tools allow the user flexibility in configuring the graphical user interface to suit their needs through the use of resizable and movable panels.

For instance, consider the default configuration for NightTune. When NightTune is invoked, the graphical user interface looks similar to the following figure:

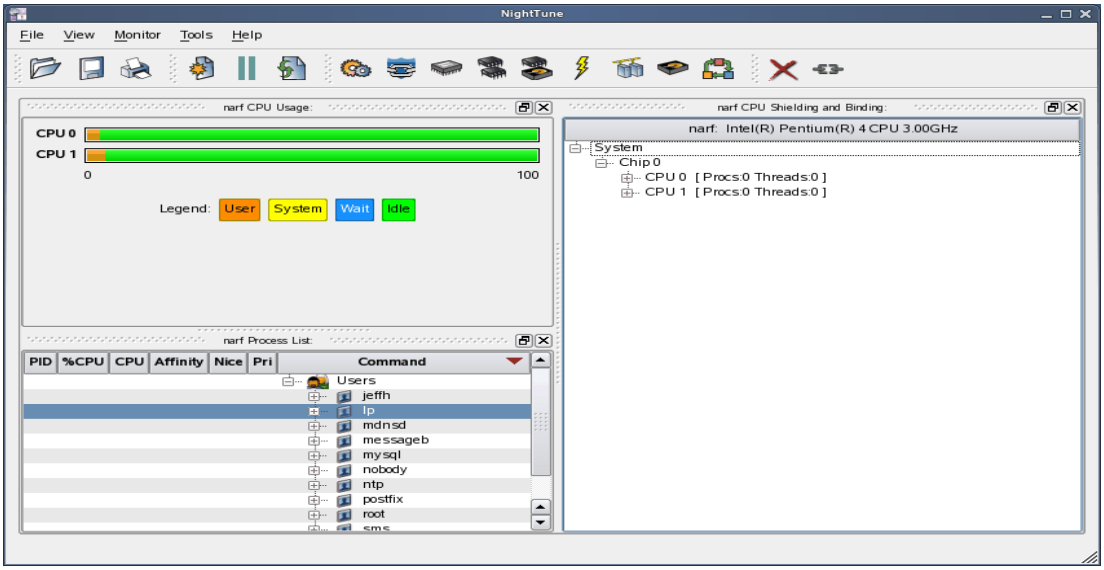


To move one of the panels in the current page, left-click on the title bar for the panel you wish to move and begin to drag the panel to the desired location. The application will respond by creating space on the page based on where you move the panel while resizing and moving the other panels accordingly.

For instance, to move the CPU Usage panel above the Process List panel, left-click on the title bar of the CPU Usage panel and begin to drag it up and to the left. NightTune will respond by creating space above the Process List panel as shown in the figure below:

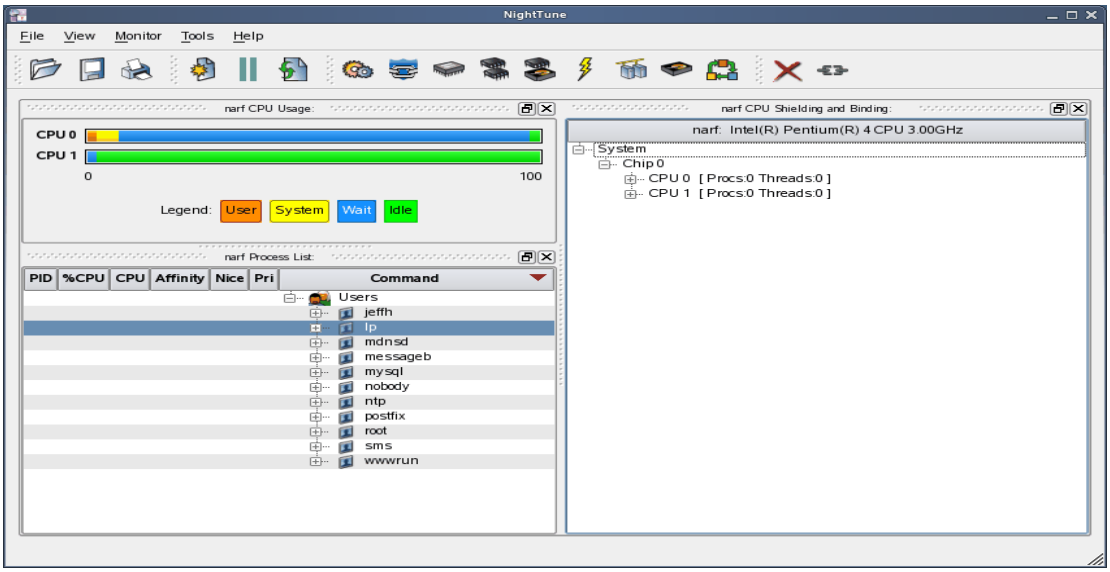


Release the mouse button when NightTune has opened a space where you desire and NightTune will place the panel in that location. The CPU Usage panel now resides in the upper left corner of the NightTune display.



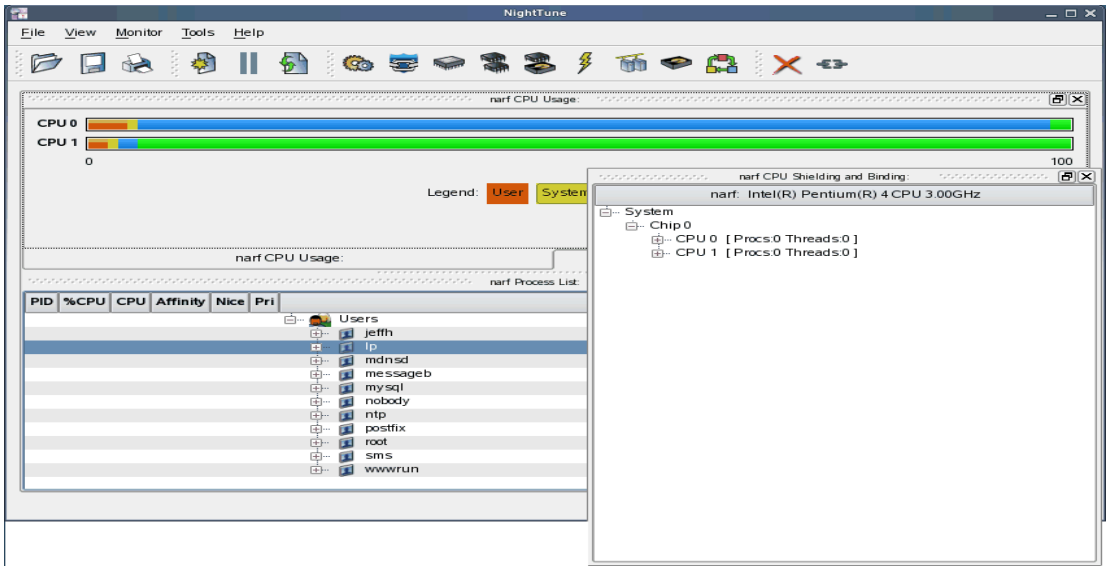
If an empty space does not appear where you desire it, try increasing the size of the main window, decreasing the size of the undocked panel, and moving an alternative edge of the undocked panel near where you want to place it.

Panels can be resized by left-clicking on the separator between the panels and dragging it to the desired size. For instance, to increase the height of the Process List panel (and thereby decrease the height of the CPU Usage panel), left-click on the separator between the two panels (the cursor will become a double-headed arrow) and drag the separator until the panels are the desired size.

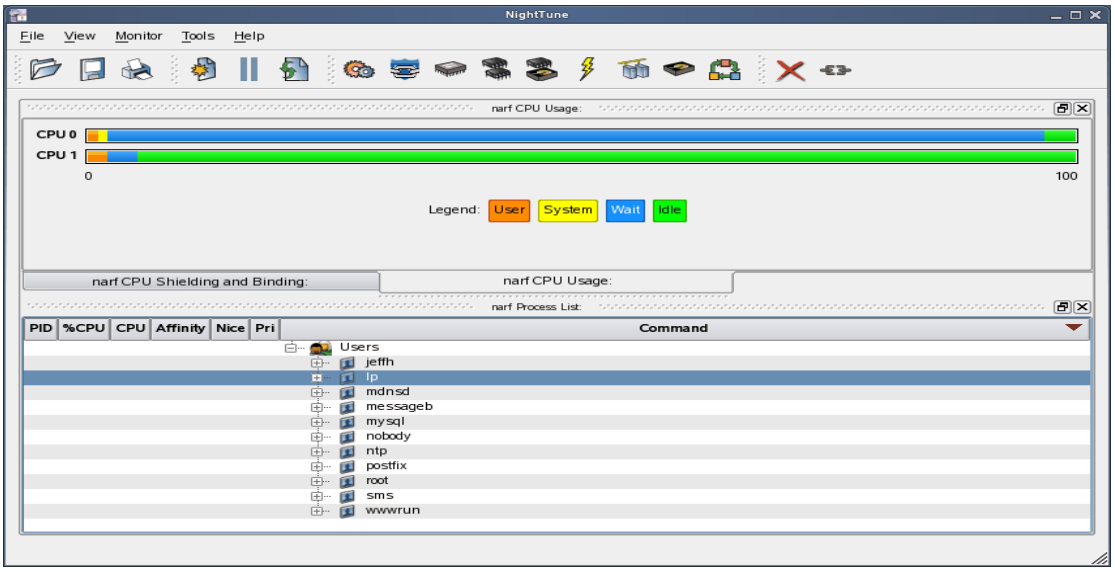


Another feature of the NightStar RT Tools graphical user interface is the use of tabbed panels. Tabbed panels allow you to maximize your GUI real estate by placing two or more panels in the same location. You can then switch between the panels using the tabs created.

In our example, we can configure NightTune so that the CPU Shielding and Binding panel and the CPU Usage panel share the same space. Left-click on the title bar of the CPU Shielding and Binding panel and drag it beneath the CPU Usage panel until you see a tab labeled “CPU Usage” created at the bottom of the CPU Usage panel as shown in the figure below.



Release the mouse button and NightTune places the CPU Shielding and Binding panel in the same location as the CPU Usage panel and creates two tabs underneath enabling you to switch back and forth between the two.

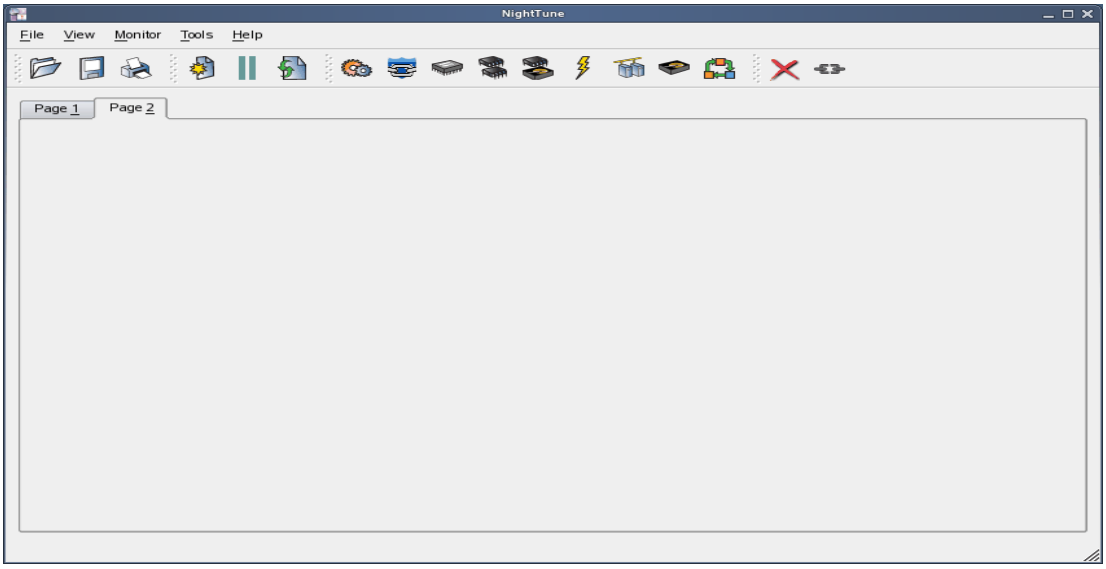


4.2. Tabbed Pages

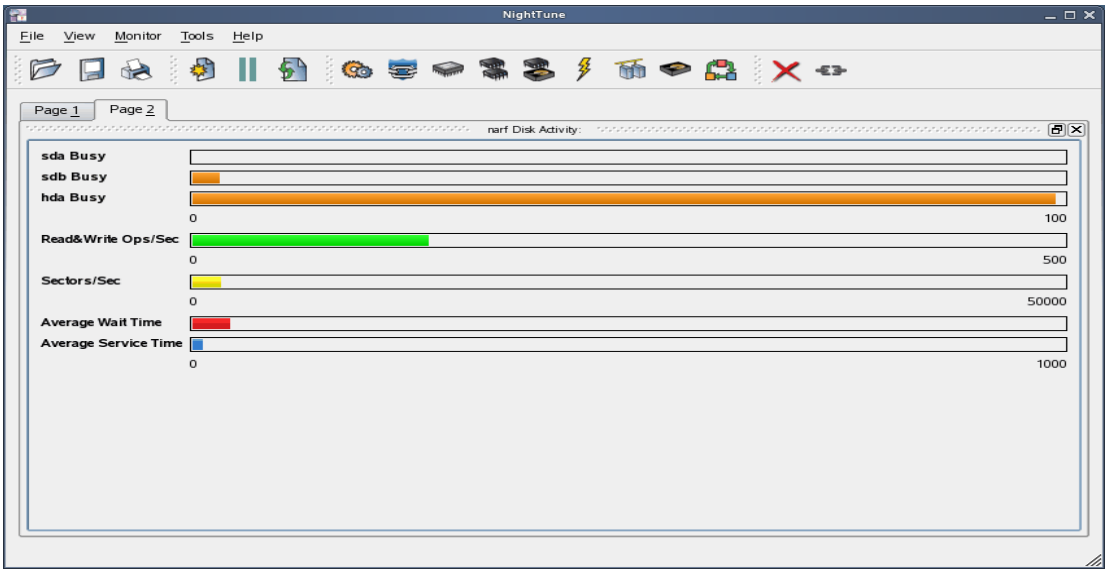
The NightStar RT Tools allow the user to maintain multiple views of data and the mechanisms that manipulate that data within each application through the use of tabbed pages. By default, only one page is displayed when the tool is invoked.

In our NightTune example from the previous section, we can create another page in which to display a different set of data. For instance, perhaps we would like to monitor disk activity, interrupt activity, and memory activity but do not want to clutter up our original page.

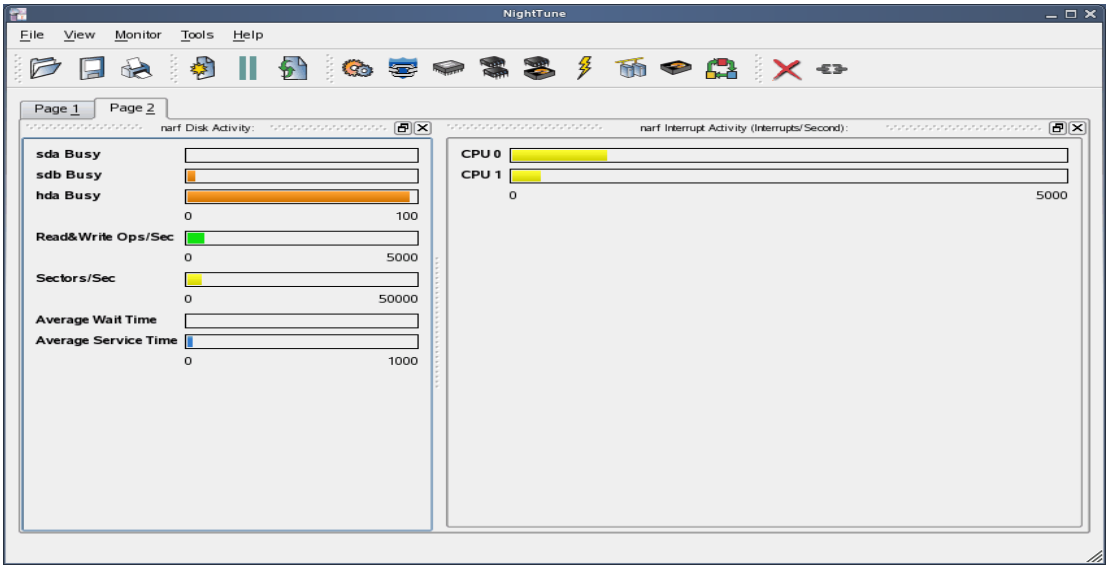
Select Add Page from the View menu. NightTune will create two tabbed pages; our original page is placed under the first tab and a new empty page will be presented under the second.



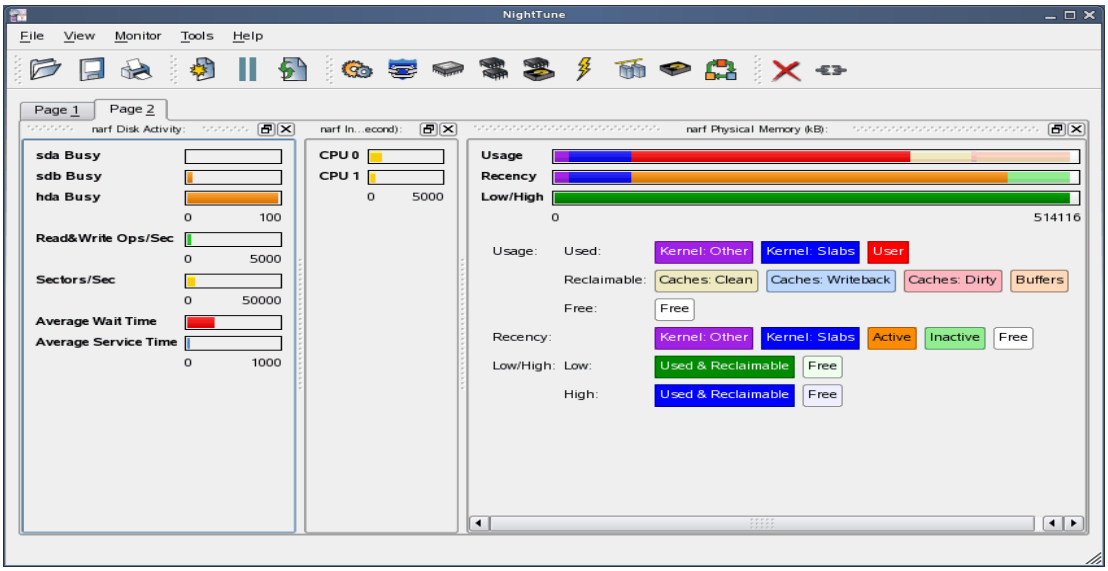
To add the desired NightTune panels, click on the Monitor menu item. You will be presented with a menu of panels to choose from. Select the Disk Activity menu item and then select Bar graph pane from the sub-menu. The Disk Activity panel displaying the information in bar graph format is added to our new page.



Select Bar graph pane from the Interrupt Activity sub-menu. The Interrupt Activity panel is added to the page.



Select Bar graph pane from the Memory: Physical sub-menu. The Memory Physical panel is added to the page.



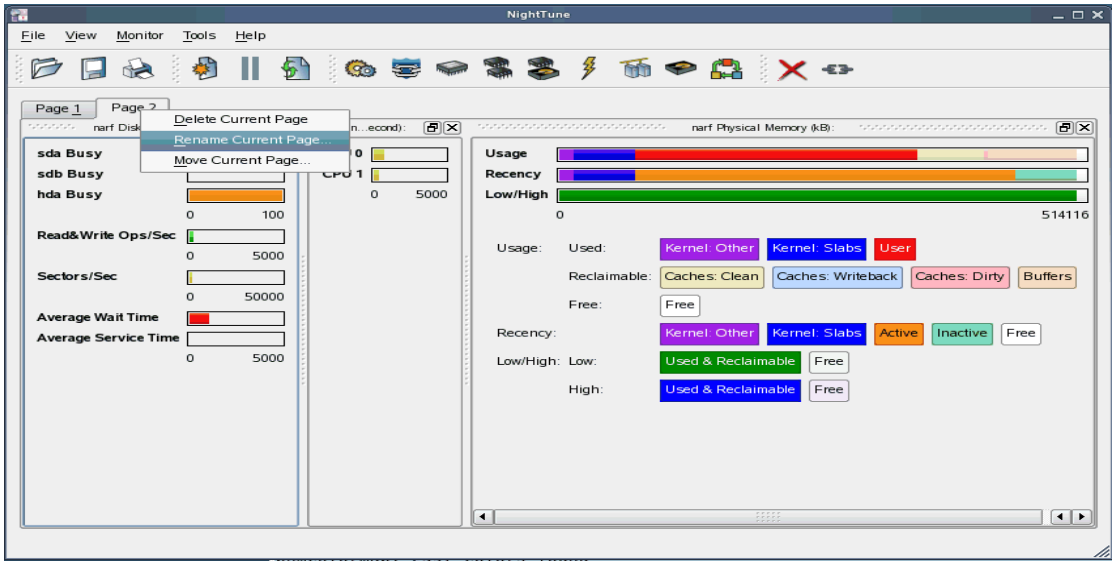
Our new page now contains the Disk Activity, Interrupt Activity, and Memory Physical panels all displaying their information in bar graph format. We can switch back to our first page by clicking on the tab labeled “Page 1” and return to our new page by clicking on the tab labeled “Page 2”.

4.3. Context Menus

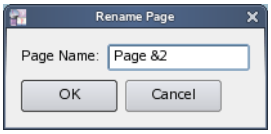
The NightStar RT Tools provide extensive use of context menus. Right-clicking in any of the NightStar RT Tools will provide the user with a menu containing items related to the location of the mouse in the tool.

We can demonstrate this feature using our NightTune example. For instance, perhaps we would like to give our new page that we created in “Tabbed Pages” on page 10 a more meaningful name.

Right-click on the tab labeled “Page 2”. We are presented with a context menu with the menu items Delete Current Page, Rename Current Page..., and Move Current Page....



Select Rename Current Page... from the context menu. The Rename Page dialog is presented.

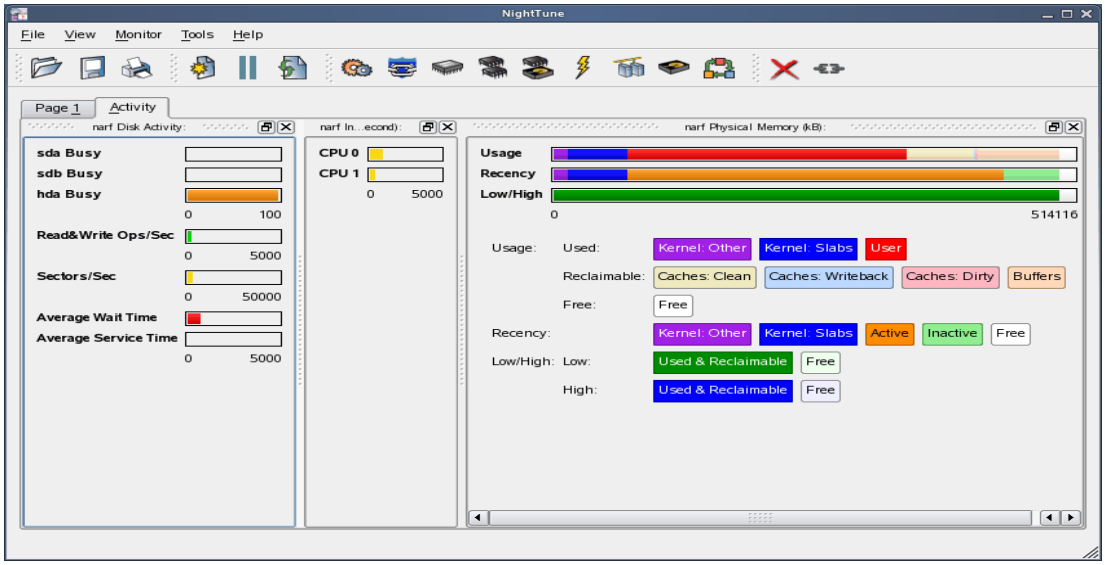


Change the Page Name to “&Activity”.

NOTE

An ampersand (&) before a particular character creates an accelerator for that page. The user can then switch to a particular page by holding down the Alt key and pressing the accelerator for that page. The accelerator is indicated on the tab by an underline.

Press Alt-1 to switch to our original page; press Alt-A to return to our Activity page.



5.0. Overview of NightStar RT

The following sections describe the basic features of each of the NightStar RT tools.

- NightProbe
- NightSim
- NightTrace
- NightTune
- NightView
- Datamon
- Shmdefine

5.1. NightProbe

The features of the NightProbe data monitor include:

- Non-intrusive sampling and recording of program data
- Synchronous and asynchronous data capture
- Flexible data display features
- Sampling, recording and replay APIs
- Time stamping of acquired data

NightProbe is a tool for independently monitoring, modifying and recording data values from multiple application resources, including programs, shared memory segments, and memory mapped files. NightProbe can be used in a development environment for debugging, analysis, proto-typing and fault injection, or in a production environment to create a GUI control panel for program input and output.

NightProbe utilizes a non-intrusive technique of mapping the target resource's address space into its own. Subsequent direct memory reads and writes by NightProbe allow it to sample and modify data without interrupting or otherwise affecting resources.

Synchronized and Asynchronous Logging

NightProbe can perform synchronous logging of data via a simple API. Asynchronous logging can be performed via on-demand sampling or a cyclic clock rate.

NightProbe provides for logging data items using tracepoints for simultaneous analysis by the NightTrace event analyzer. Sampled data can be combined with kernel trace and additional user trace data to obtain a synchronized picture of application and operating system behavior. NightProbe can record data to disk files or provide data directly to the NightTrace tool.

Interactive Sampling and Modification

NightProbe provides a flexible spreadsheet display for on-demand or cyclic sampling of data at user-specified refresh rates. Direct modification of user data is accomplished by typing in new values for data items into the spreadsheet. NightProbe provides colorized notification of violations of user-defined

data thresholds for individual data items. NightProbe allows sampled data to be timestamped and passed off to user applications written with the NightProbe API for subsequent analysis, recording or customized display.

NightProbe supports scalar and structured data types in C/C++ and Fortran that have statically-determined addresses and shapes. NightProbe scans the symbol table and debug information of user programs allowing the user to browse for data items or specifically enter the names of data items to be monitored. Any application that contains symbol table and debug information may be used with NightProbe. No application source code changes are required.

5.2. NightSim

The features of the NightSim application scheduler include:

- Periodic execution of multiple processes
- Major and minor cycles with frame overrun notification and control
- Single point of scheduling control for distributed systems
- Ideal for simulation applications

NightSim is a tool for scheduling and monitoring time-critical applications that require predictable, cyclic process execution. Ideal for simulation applications, NightSim allows developers to dynamically adjust the execution of multiple, coordinated processes, their priorities, scheduling policies, and CPU assignments. With NightSim, users can monitor the performance of applications by displaying period execution times, minimums and maximums, and can take action when frame overruns occur.

NightSim provides a graphical interface to the operating system's Frequency-Based Scheduler (FBS), a high-resolution task scheduler that enables processes to run in cyclical patterns. NightSim allows users to easily configure groups of processes to run on local or distributed systems, and save the resulting configurations for reuse. A performance monitor gathers CPU utilization data for processes running under the FBS.

NightSim may be used during the development, debug and production phases of a simulation application. Simulation configurations can be saved as a script, which can then be executed to repeat a simulation. NightSim scripts are useful in target environments where GUI processing is prohibited or undesired. In addition, configuration files and scripts may be placed under any version control system.

Synchronized Distributed Scheduling

In addition to symmetric multiprocessors, NightSim supports multiple systems connected via Concurrent's Real-Time Clock and Interrupt Module. NightSim simplifies the creation of distributed scheduling and provides a single-point-of-control for managing the synchronized timing (start/stop/resume) of individual schedulers distributed across multiple target systems.

NightSim handles the interface to hardware such as real-time clocks and distributed interrupt sources. Users don't need to interface with the underlying operating system for scheduling operations.

Extensive Performance Statistics

NightSim monitors up to 11 different performance-related statistics as well as up to 15 additional parameters for each scheduled process. Using statistics such as minimum and maximum cycle times, users can optimize CPU utilization by balancing their load across multiple processors. NightSim displays are customizable, allowing users to select specific statistics and processes to monitor and the sorting criteria for weighted display.

5.3. NightTrace

The features of the NightTrace event analyzer include:

- Synchronized graphical or text display of system application activity
- User-defined event logging in single or multi-threaded applications
- Kernel event logging including system calls, interrupts and exceptions
- Data analysis API

NightTrace is a tool for displaying and analyzing the dynamic behavior of applications, the Linux operating system and the interaction between them. NightTrace can log events from multiple processes executing simultaneously on multiple CPUs or systems. NightTrace can also combine user-defined application events with kernel events to present a synchronized view of the entire system. NightTrace then creates a graphical time-based view of all logged events. NightTrace allows users to zoom, search, filter, summarize and analyze events. Tracing analysis can be performed live or post execution.

NightTrace was specifically designed to meet the most stringent requirements of time-critical applications. Using synchronized, fast-access hardware clocks and kernel-free primitives, NightTrace tracepoints are logged with minimal overhead. Tracepoints can be inserted into device drivers, interrupt level code and any user application. Tracepoints can be left in production-quality applications even when not collecting trace data.

Graphical and Interactive

NightTrace graphically displays requested events and states along a timeline graph or event log to clearly show the relative timing of events and provide an overall picture of application and operating system activity. NightTrace can locate specific events and zoom in on them with a fine degree of granularity for precise timing observation. The NightTrace graphical display is completely user-configurable for customized viewing. Configurations can be saved and later recalled, and multiple configurations can be viewed simultaneously.

Kernel Trace Support

By combining system event information such as interrupts, exceptions, context switches, Linux system calls and device accesses together with event information from user applications, NightTrace provides a clear picture of the interaction between the kernel and user applications at any point during the application's run.

NightTrace provides statistical performance data about events and states, including frequency, time of occurrence, duration, gap and minimum and maximum times. Users can create state definitions and qualify events by specifying the applicable process, thread, CPU, system and event content. Conditional tracing can be expressed using C expression syntax. Displays can be customized to yield insight into operating system and application performance and behavior patterns.

NightTrace generates source code using an Analysis API that allows users to easily create custom applications that monitor or analyze application or system activity.

5.4. NightTune

The features of the NightTune system and application tuner include:

- Dynamic display of system and application performance
- Monitoring of CPU use, memory paging and network operation
- Interactive control of processes, priorities, policies and interrupts
- Dynamic CPU affinity control for processes, threads and interrupts

NightTune provides a graphical interface to system facilities for monitoring and tuning application and system performance. Users can monitor the priority, scheduling policy, CPU assignment and CPU usage of user applications. NightTune also monitors system CPU usage, context switches, interrupts, memory paging and network activity.

NightTune can monitor processes individually or in groups determined by user or by CPU. NightTune also displays information about individual threads or tasks within a process. Multiple frames and windows are used to display information allowing users to customize their display.

Application Tuning

NightTune allows users to change the process attributes of an individual thread, task, process or group of processes as a whole using pop-up dialogs and drag-and-drop actions. For example, dragging a process icon to a CPU icon binds the process to that processor. The user then instantly sees the results of the tuning effort both graphically and as text.

System Tuning

NightTune allows users to change the CPU assignment of interrupts using pop-ups or drag-and-drop actions. NightTune optionally provides a textual log of all application and system tuning actions taking during a NightTune session.

5.5. NightView

The features of the NightView source-level debugger include:

- Multi-system, multi-processor, multi-process, multi-thread debugging via single interface
- Hot patches including breakpoints, monitorpoints and watchpoints
- Application speed conditions
- Dynamic memory “heap” debugging
- Modification and display of variables during execution

NightView allows users to simultaneously debug multiple, time-critical processes. With NightView, a programmer can change program execution and modify or display data without stopping or interrupting the program. Eventpoint conditions, such as hit and ignore counts, are patched directly into an application and can execute at full application speed. NightView provides fine-grained control without adversely affecting application timing.

NightView monitorpoints can display expressions at user-selected locations without stopping a process, thus providing data displays that are synchronized with the application's algorithms. Watchpoints utilize hardware address trap features that cause an application to stop when user-specified variables or memory locations are selectively read or modified.

Language-sensitive Debugging

NightView supports the debugging of multiple applications written in any combination of C/C++ and Fortran. All variables and expressions in each program are referenced in the appropriate language. NightView is also integrated with the NightTrace event analyzer. NightView can insert tracepoints at user-specified locations for concurrent or post execution analysis by NightTrace.

More Powerful Than The Gnu Debugger

NightView offers many features not available in the gnu debugger (**`gdb`**). Advantages of NightView include the ability for users to debug multiple processes from a single session and processes started from scripts. With NightView, patched-in code runs at full speed. While a process is executing, hot patching can modify variables or add eventpoints. Monitorpoints can display expressions and stack variables, and signals can be sent directly to the process, bypassing the debugger.

Dynamic Memory Debugging

NightView includes an interactive memory debugger that helps find and eliminate memory problems during the debug process without code recompilation. NightView watches for heap memory leaks, monitors the amount of memory an application uses, and tracks how it allocates and frees memory. With its memory debugger enabled, NightView lets users track heap allocations and deallocations in real-time, thus allowing for more efficient debugging than post-run analysis. Programmers can stop execution, check for problems, test patches and then continue debugging. NightView can detect double-frees, dangling pointers, heap area overruns, and other common user application bugs.

5.6. Datamon

Datamon is a user application interface that allows user programs to monitor, record, and modify variables in independently executing processes in real-time. It includes the ability to scan a program file for eligible variables and obtain detailed information about their attributes, including type name, atomic type, bit size, bit offset, shape, component members, and address. Datamon utilizes a non-intrusive technique for accessing and modifying variables.

5.7. Shmdefine

Shmdefine aids in the sharing of data between independent programs. While most useful for sharing common blocks between Fortran programs, it helps Fortran, C, and Ada programs to effectively utilize the IPC shared memory services.

6.0. Getting Started

The *NightStar RT Tutorial* is ***highly recommended*** as an introduction to the NightStar RT product. This tutorial integrates all of the NightStar RT tools into one cohesive example incorporating various scenarios which demonstrate their extensive functionality.

The tutorial is available in PDF format in the **`documentation`** directory of the *NightStar RT Installation CD* as well as in **`/usr/lib/NightStar-RT/docs`** after installation.

The online version of the tutorial can be accessed by double-clicking on the NightStar RT Documentation icon installed on the desktop and selecting the NightStar RT Tutorial from the Bookshelf.

In addition, the tutorial can be launched from the Help menu of any NightStar RT tool or can be started by issuing the following command:

```
nhelp nstar-rt-tutorial
```

from the command line.

6.1. Capabilities

Most operations with NightStar RT do not require any special privileges. However, if you wish to take full advantage of NightStar RT capabilities without running as the root user, additional configuration steps are required.

Linux provides a means to grant otherwise unprivileged users the authority to perform certain privileged operations. The Pluggable Authentication Module (see **`pam_capability(8)`**) is used to manage sets of capabilities, called *roles*, required for various activities.

The following table lists the advantages granted to non-root users with the capabilities suggested for use with NightStar RT:

Capabilities and their Effects

Capability	Advantage
`CAP_IPC_LOCK`	Allows NightTrace to lock critical pages into memory related to User Trace event buffers.
`CAP_SYS_RAWIO`	Allows NightProbe to gain access to PCI devices and memory mapped system files, such as **`/dev/mem`**.
`CAP_SYS_NICE`	Allows the NightStar RT tools to set the scheduling policy, scheduling priority, and CPU affinity of processes. Allows NightTune to set the CPU affinity of interrupt and to shield CPUs from process, interrupts. and hyper-threading interference.

Linux systems should be configured with an *`nstaruser`* role which provides the `CAP_SYS_NICE`, `CAP_SYS_RAW_IO` and `CAP_IPC_LOCK` capabilities.

Edit **/etc/security/capability.conf** and define the `nstaruser` role (if it is not already defined) in the "ROLES" section:

```
role nstaruser CAP_SYS_NICE CAP_IPC_LOCK CAP_SYS_RAWIO
```

Additionally, for each NightStar RT user on the target system, add the following line at the end of the file:

```
user   username      nstaruser
```

where *username* is the login name of the user.

If the user requires capabilities not defined in the `nstaruser` role, add a new role which contains `nstaruser` and the additional capabilities needed, and substitute the new role name for `nstaruser` in the text above.

In addition to registering your login name in **/etc/security/capability.conf**, certain files under the **/etc/pam.d** directory must also be configured to allow capabilities to be activated.

To activate capabilities, add the following line to the end of selected files in **/etc/pam.d** if it is not already present:

```
session required pam_capability.so
```

The list of files to modify is dependent on the list of methods that will be used to access the system. The following table presents a recommended configuration that will grant capabilities to users of the services most commonly employed in accessing a system.

Recommended /etc/pam.d Configuration

/etc/pam.d File	Affected Services	Comment
remote	telnet rlogin rsh (when used w/o a command)	Depending on your system, the **remote** file may not exist. Do not create the **remote** file, but edit it only if it is present.
password-auth	Most all login mechanisms	This file is present in recent OS distributions. If it is present, add the clause mentioned above to this file.
login	local login (e.g. console) telnet* rlogin* rsh* (when used w/o a command)	*On some versions of Linux, the presence of the **remote** file limits the scope of the **login** file to local logins. In such cases, the other services listed here with **login** are then affected solely by the **remote** configuration file.
rsh	rsh (when used with a command)	e.g. **rsh system_name a.out**
sshd	ssh	You must also edit **/etc/ssh/sshd_config** and ensure that the following line is present: **UsePrivilegeSeparation no**
gdm	gnome sessions	
kde	kde sessions	

If you modify **/etc/pam.d/sshd** or **/etc/ssh/sshd_config**, you must restart the **sshd** service for the changes to take effect:

```
/sbin/service sshd restart
```

In order for the above changes to take effect, the user must log off and log back onto the target system.

NOTE

To verify that you have been granted capabilities, issue the following command:

```
/usr/sbin/getpcaps $$
```

The output from that command will list the roles currently assigned to you.

7.0. NightStar RT Licensing

NightStar RT uses the NightStar License Manager (NSLM) to control access to the NightStar RT tools.

License installation requires a licence key provided by Concurrent. The NightStar RT tools request a licence (see "License Requests" on page 24) from a license server (see "License Server" on page 24).

Two license modes are available, fixed and floating, depending on which product option you purchased. Fixed licenses can only be served to NightStar RT users from the local system. Floating licenses may be served to any NightStar RT user on any system on a network.

Tools are licensed per system, per concurrent user. Concurrent usage of any or all NightStar RT tools by the same user from the same system automatically share a single license. The intent is to allow *n* developers to fully utilize all the tools at the same time while only requiring *n* licenses. When operating the tools in remote mode, where a tool is launched on a local system but is interacting with a remote system, licenses are required only from the host system.

You can obtain a license report which lists all licenses installed on the local system, current usage, and expiration date for demo licenses (see "License Reports" on page 24).

The default operating system configuration may include a strict firewall which may interfere with floating licenses. See "Firewall Configuration for Floating Licenses" on page 25 for information on handling such configurations.

7.1. License Keys

Licenses are granted to specific systems to be served to either local or remote clients, depending on the license model, fixed or floating.

License installation requires a license key provided by Concurrent. To obtain a license key, you must provide your system identification code. The system identification code is generated by the **`nslm_admin`** utility:

```
nslm_admin --code
```

System identification codes are dependent on system configurations. Reinstalling Linux or NightStar RT on a system or replacing network devices may require you to obtain new license keys.

To obtain a license key, use the following URL and click on the ***Permanent*** link:

http://real-time.ccur.com/NightStarRTKeys

Provide the requested information, including the system identification code. Your license key will be immediately emailed to you.

Install the license key using the following command:

```
nslm_admin --install=xxxx-xxxx-xxxx-xxxx-xxxx
```

where *xxxx-xxxx-xxxx-xxxx-xxxx* is the key included in the license acknowledgment email.

If the required information is not readily available, or you have special circumstances, contact Concurrent support (see "Direct Software Support" on page 37 for more information).

7.2. License Requests

By default, the NightStar RT tools request a license from the local system. If no licenses are available, they broadcast a license request on the local sub-net associated with the system's hostname.

You can control the license requests for an entire system using the **/etc/nslm.config** configuration file.

By default, the **/etc/nslm.config** file contains a line similar to the following:

```
:server @default
```

The argument **@default** may be changed to a colon-separated list of system names, system IP addresses, or broadcast IP addresses. Licenses will be requested from each of the entities found in the list until a license is granted or all entries in the list are exhausted.

For example, the following setting prevents broadcast requests for licenses by only specifying the local system:

```
:server localhost
```

The following setting requests a license from **server1**, then **server2**, and then a broadcast request if those fail to serve a license:

```
:server server1:server2:129.134.30.0
```

Similarly, you can control the license requests for individual invocations of the tools using the **NSLM_SERVER** environment variable. If set, it must contain a colon-separated list of system names, system IP addresses, or broadcast IP addresses as described above. Use of the **NSLM_SERVER** environment variable takes precedence over settings defined in **/etc/nslm.config**.

7.3. License Server

The NSLM license server is automatically installed and configured to run when you install NightStar RT.

The **nslm** service is automatically activated for run levels 2, 3, 4, and 5. You can check on these settings by issuing the following command:

```
/sbin/chkconfig --list nslm
```

In rare instances, you may need to restart the license server via the following command:

```
/sbin/service nslm restart
```

See **nslm(1)** for more information.

7.4. License Reports

A license report can be obtained using the **nslm_admin** utility.

```
nslm_admin --list
```

lists all licenses installed on the local system, current usage, and expiration date (for demo licenses). Use of the **--verbose** option also lists individual clients to which licenses are currently granted.

Adding the **--broadcast** option will list this information for all servers that respond to a broadcast request on the local sub-net associated with the system's hostname.

See **nslm_admin(1)** for more options and information.

7.5. Firewall Configuration for Floating Licenses

The default Red Hat configuration includes a strict firewall which interferes with floating licenses.

If such a system is used to serve licenses, then at least one port must be opened in its firewall to allow server requests to pass. See "Serving Licenses with a Firewall" on page 25 for more information.

Similarly, if such a system is host to the NightStar RT tools, then at least one port must be opened in its firewall so that it can receive licenses from the license server. If this is not done, a tool requesting a floating license will not receive it and will not function properly. See "Running NightStar RT Tools with a Firewall" on page 26 for more information.

7.5.1. Serving Licenses with a Firewall

Following are a few approaches for allowing the NSLM license server to serve floating licences when the system on which it is running is configured with a firewall:

- disable the firewall on the system entirely
- allow NSLM license requests from a specific system (or one of several)
- allow NSLM license requests from any system on a particular subnet (or one of several)
- allow NSLM license requests from any system

NOTE

You must be root in order to modify the firewall configuration.

To disable the firewall entirely, execute:

```
service iptables stop
```

and then remove the **/etc/sysconfig/iptables** file:

```
rm -f /etc/sysconfig/iptables
```

This option may not be as dangerous as it seems. Often, whole networks are protected with a firewall so it is not necessary for individual systems on the network to be protected further. If unsure, check with your network administrator.

For the remaining cases, a simple modification should be made to the **/etc/sysconfig/iptables** file. By default, that file should contain a line like the following:

```
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
```

To allow NSLM license requests from a specific system, insert the following lines before the `REJECT` line:

```
-A RH-Firewall-1-INPUT -p udp -m udp -s system --dport 25517 -j ACCEPT
```

```
-A RH-Firewall-1-INPUT -p tcp -m tcp -s system --dport 25517 -j ACCEPT
```

Those lines can be repeated for multiple systems.

To allow NSLM license requests from any system on a particular subnet, insert the following lines before the `REJECT` line:

```
-A RH-Firewall-1-INPUT -p udp -m udp -s subnet/mask --dport 25517 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m tcp -s subnet/mask --dport 25517 -j ACCEPT
```

The subnet might be of a form like `192.168.1.0` and the mask could be a traditional network mask like `255.255.255.0` or a single number like `24`, which indicates the number of bits from the left that are part of the mask. For example, `192.168.1.0/255.255.255.0` and `192.168.1.0/24` are equivalent.

Those lines can be repeated for multiple subnets.

To allow NSLM license requests from any system, insert the following lines before the `REJECT` line:

```
-A RH-Firewall-1-INPUT -p udp -m udp --dport 25517 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 25517 -j ACCEPT
```

After modifying **`/etc/sysconfig/iptables`**, execute:

```
service iptables restart
```

7.5.2. Running NightStar RT Tools with a Firewall

Following are a few approaches for allowing a NightStar RT tool to receive floating licenses from a license server, when the system running the NightStar RT tool is configured with a firewall:

- disable the firewall on the requesting system entirely
- allow NSLM licenses from a specific license server (or one of several)
- allow NSLM licenses from any system on a particular subnet (or one of several)
- allow NSLM licenses from any system

NOTE

You must be root in order to modify the firewall configuration.

To disable the firewall entirely, execute:

```
service iptables stop
```

and then remove the **`/etc/sysconfig/iptables`** file:

```
rm -f /etc/sysconfig/iptables
```

This option may not be as dangerous as it seems. Often, whole networks are protected with a firewall so it is not necessary for individual systems on the network to be protected further. If unsure, check with your network administrator.

For the remaining cases, a simple modification should be made to the **/etc/sysconfig/iptables** file. By default, that file should contain a line like the following:

```
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
```

To allow NSLM licenses from a specific system running a license server, insert the following line before the REJECT line:

```
-A RH-Firewall-1-INPUT -p udp -m udp -s server --sport 25517 -j ACCEPT
```

That line can be repeated for multiple servers.

To allow NSLM licenses from any system running a license server on a particular subnet, insert the following before the REJECT line:

```
-A RH-Firewall-1-INPUT -p udp -m udp -s subnet/mask --sport 25517 -j ACCEPT
```

The subnet might be of a form like 192.168.1.0 and the mask could be a traditional network mask like 255.255.255.0 or a single number like 24, which indicates the number of bits from the left that are part of the mask. For example, 192.168.1.0/255.255.255.0 and 192.168.1.0/24 are equivalent.

That line can be repeated for multiple subnets.

To allow NSLM licenses from any system running a license server, insert the following line before the REJECT line:

```
-A RH-Firewall-1-INPUT -p udp -m udp --sport 25517 -j ACCEPT
```

After modifying **/etc/sysconfig/iptables**, execute:

```
service iptables restart
```

Following are a few approaches for allowing the NSLM license server to serve floating licences when the system on which it is running is configured with a firewall:

- disable the firewall on the system entirely
- allow NSLM license requests from a specific system (or one of several)
- allow NSLM license requests from any system on a particular subnet (or one of several)
- allow NSLM license requests from any system

NOTE

You must be root in order to modify the firewall configuration.

To disable the firewall entirely, execute:

```
service iptables stop
```

7.6. License Support

For additional aid with licensing issues, contact the Concurrent Software Support Center. See “Direct Software Support” on page 37 for details.

8.0. Architecture Interoperability

The NightStar RT tools were designed to be used in a self-hosted environment as well as remotely, separating the host processing from the time-critical target system.

The following sections describes the interoperability of each tool between systems of different word-size architectures in the Intel x86 and AMD64 families.

NightProbe

No limitations.

As of this release, the NightProbe Trigger API may be used by a 32-bit program on a 64-bit system. Additionally, playback of either data previously recorded by NightProbe from either architecture is supported on either architecture.

NightSim

No limitations within NightSim itself, however, RedHawk does not support 32-bit programs that use the Frequency Based Scheduler (FBS) services on a 64-bit system.

NightTune

No limitations.

NightTrace

Limitations

- NightTrace cannot interact with a NightTrace server on a different architecture (thus you cannot remotely capture trace data on a 64-bit system from a 32-bit system, and vice versa).
- NightTrace and programs using the NightTrace Analysis API on a 32-bit system cannot analyze data from a 64-bit system.

New Capabilities in this Release

- 32-bit applications can use the NightTrace Logging API and execute on a 64-bit system. The 64-bit NightTrace can capture and analyze data from such programs via **`ntraceud`** and **`ntrace`**. The 32-bit applications must be linked with the version of NightTrace Logging API from this release (or newer).
- NightTrace running on a 64-bit system can analyze data files generated on a 32-bit system; both user and kernel data. The 32-bit applications must be linked with the version of the NightTrace Logging API from this release (or newer).
- 64-bit applications using the NightTrace Analysis API can analyze data, either in stream or file mode, generated from 32-bit applications (or 32-bit kernel data).

User Responsibility

- When analyzing 32-bit data on a 64-bit system, be aware that NightTrace will evaluate argument expressions as per the type of the function as they would be evaluated on the 64-bit system. Thus, explicitly specifying `arg_long`() in a NightTrace expression will result in 8 bytes of data being extracted from the trace event, even though only 4 bytes were logged. The data types of concern are `long` and all `pointer` types.

NOTE

NightTrace automatically prints the arguments in Timelines and Event panels with the correct data type.

- When unpacking block arguments within NightTrace, you must unpack them as a 32-bit compiler would have laid out the structure. In addition to the differing sizes of `long`, `long double`, and all `pointer` types, 64-bit compilers pad structures differently. Use care. This includes using the information generated from a 32-bit Application Illumination session; references to `long` will extract 8 bytes even though it expects only 4.

NOTE

In reality, there is no problem using `arg_long_dbl`() in a 64-bit NightTrace session when the actual `long double` item was generated from a 32-bit program. Even though there are 4 extra bytes of data at the end of a `long double` on 64-bit systems, those extra bytes are completely ignored (currently) by the instructions that operation on such values.

NightView

Limitations

Obviously, NightView running on a 32-bit system cannot debug 64-bit programs on that system (since they can't execute!).

New Capabilities in this Release

- NightView can debug 32-bit applications on a 64-bit system with the following caveats:

  1. Start NightView with the **`--arch=i386`** command line option, or, create a Remote Shell and check the Debug 32-bit applications on the x86_64 target check-box in the Remote Shell dialog.

  2. When debugging 32-bit applications on a 64-bit system, a new 32-bit shell is launched within NightView in order to debug the application. If the 32-bit program (or the implicit 32-bit shell) execs a 64-bit program, that program cannot be debugged.

3. You can debug both 32-bit and 64-bit programs from the same NightView session (using a 32-bit Remote Shell as described in the previous caveat), but all 32-bit programs to be debugged must be launched from that shell, and all 64-bit programs to be debugged must be launched from a normal shell. It is recommended that you add a Shells item to a data panel (via the Data menu) in NightView to facilitate such debugging.

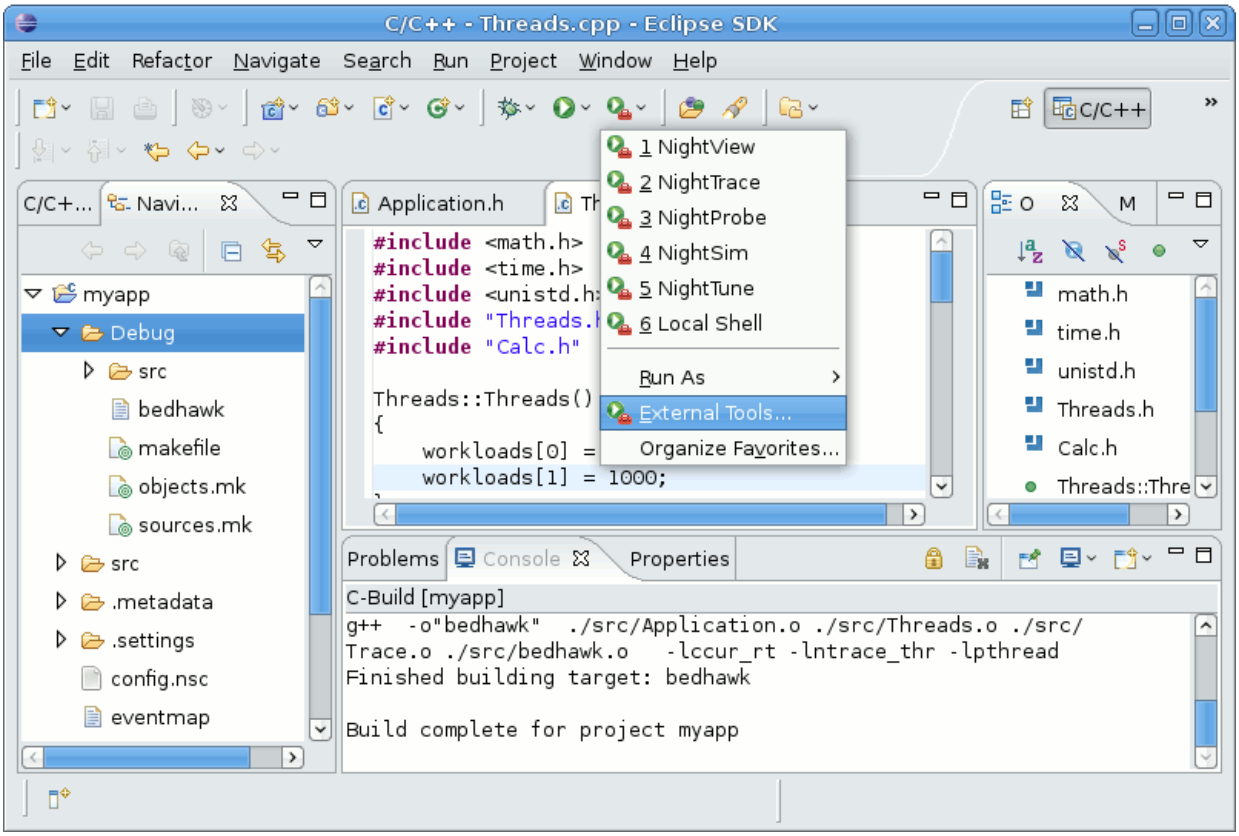
9.0. Operating NightStar RT From Within Eclipse

Eclipse is an Integrated Development Environment that is included on the Red Hat discs in the latest RedHawk media pack; specifically, Eclipse version 3.2 - Red Hat Edition.

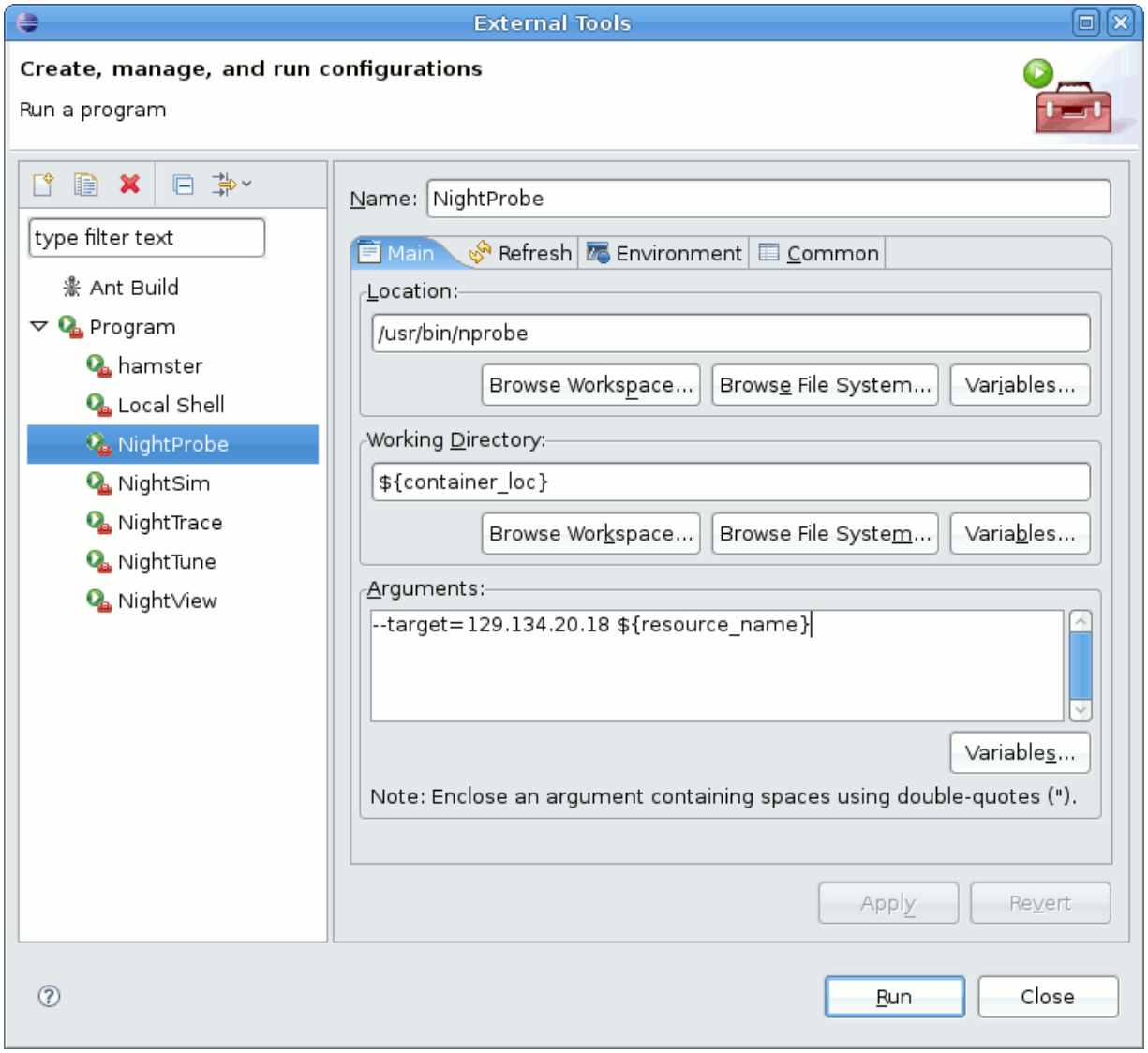
The NightStar RT tools can be launched directly from the Eclipse workspace using customized Eclipse menus.

In addition to the convenience afforded by launching the tools from within an Eclipse coding and building environment, you can take advantage of Eclipse's custom menu configuration to pass useful options and arguments to the NightStar tools.

To add an invocation of a NightStar tool, select the External Tools… menu option from the Favorites icon on the Eclipse toolbar.



The External Tools dialog allows you to customize the launch of a utility. In the figure below, the dialog shows the customization of a NightProbe invocation.



Press the New icon (the image of a page with a plus sign in the upper-right corner) and fill in the four text areas, Name, Location, Working Directory, and Arguments and then click Apply.

Use of Eclipse variables is recommended. The most useful variable is the **`${resource_name}`** variable (and its companions). This variable is replaced with the name of the program that is currently selected in Eclipse. Thus if you have multiple programs in your Eclipse workspace, you don't need individual NightStar menus to operate on each program of interest. Rather, the tool invocations can refer to the selected program via this variable.

The working directory can be set using the **`${container_loc}`** variable expression, which sets the current working directory during launch to the directory containing the currently-selected program in Eclipse.

Using **`${container_loc}`** is recommended for all NightStar invocations for the Working Directory field.

Use of the Arguments area differs for each tool. The sections below describe useful things to put into the area for each NightStar tool.

NightProbe

--target=*remote_system*

Use this option if your program is run on a remote system.

${resource_name}

This Eclipse expression specifies the name of the program you want to probe. The pathname is relative to the containing directory path, which is recommended for the Working Directory field above.

NightSim

--target=*remote_system*

Use this option if your program is run on a remote system, unless you also specify the **`--file`** option below.

--file=${project_loc}/*config_file*

Typically, NightSim users define their scheduler configuration and save the information to a file for reuse. The **`${project_loc}`** Eclipse expression provides the absolute path to the project directory where you would typically keep such files.

NightTrace

${project_loc}/*event_map*

If you maintain an event map file for NightTrace, you'll want to add that to the invocation. The **`${project_loc}`** Eclipse expression provides the absolute path to the project directory where you would typically keep such files.

${resource_name}

It's a good idea to add an argument to NightTrace that specifies the path of a user application program that participates in tracing. If the program has been illuminated with the NightTrace **`nlight`** utility, then it contains information about the event maps and daemon definitions set up by **`nlight`** (NightTrace will automatically extract that information from the executable file). Even if you're not using **`nlight`**, but your application logs trace points, it's still useful to specify its pathname as an argument, so that you can use the NightTrace `lookup_pc()` function to map PC values in trace data to file/line/routine information.

NightTune

--target=*remote_system*

Use this option if your program is run on a remote system. NightTune doesn't need any information specifically from your program executable files, but it will be convenient to be viewing the system where your program runs from within NightTune.

NightView

--target=*remote_system*

Use this option if your program is run on a remote system.

${resource_loc} *arg1 arg2 ...*

The **`${resource_loc}`** Eclipse expression provides the full path to your program on the local system. Note that if you are debugging the program on a remote system, it might be better to use the following:

*/path/on/remote/system/***`${resource_name}`** *arg1 arg2 ...*

This may be more convenient since you may be transferring your program to the remote system and placing it in a different location.

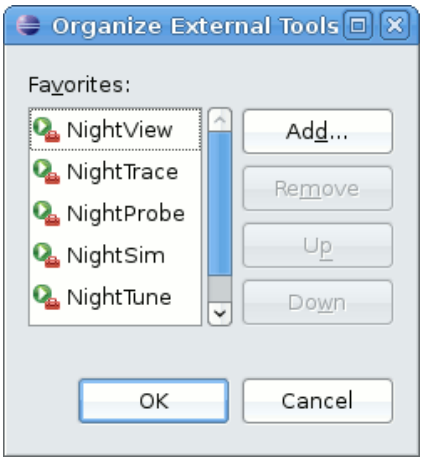
NOTE

When using remote debugging, it's convenient to add an additional External Tool definition that copies your program to the target system. For example, when defining an invocation for the secure shell copy program **`scp`**, you might use the following in the Arguments area:

`${resource_loc}` *remote_system:/path/on/remote/system*.

The inclusion of *arg1 arg2 ...* above represents arguments that you might want to pass to the program you are debugging. All arguments after the program name on a NightView invocation are passed to the program being debugged.

Once you have defined invocations for all the NightStar tools of interest, select the Organize Favorites… menu option from the from the Favorites icon on the Eclipse toolbar.



Use the Add… button to select the NightStar invocations from a list and place them into the Favorites category so they will be directly visible in the Favorites menu.

10.0. Direct Software Support

Software support is available from a central source. If you need assistance or information about your system, please contact the Concurrent Software Support Center at our toll free number 1-800-245-6453. For calls outside the continental United States, the number is 1-954-283-1822. The Software Support Center operates Monday through Friday from 8 a.m. to 5 p.m., Eastern Standard Time.

You may also submit a request for assistance at any time by using the Concurrent Computer Corporation web site at **http://real-time.ccur.com/support** or by sending an email to **support@ccur.com**.