NightStar RT Installation Guide

Version 4.5

(RedHawk™ Linux®)

concurrent

0898008-4.5
April 2016

Contents

1.0. Introduction

NightStar RT Version 4.5 is a production release of the NightStar Tools running under RedHawk Linux.

This release includes enhancements and corrections to NightStar. It is all inclusive, insomuch as it contains the original NightStar RT 4.4 content plus all changes to date.

NightStar RT 4.5 is required for use with RedHawk 7.2. Previous NightStar RT CDs may not even install on RedHawk 7.2 due to changes in RPM file sharing rules by the underlying Linux distribution.

1.1. What is NightStar RT?

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.

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 13 to get an overview of the new interface.

2.0. New in NightStar RT 4.5

Apart from bug fixes and minor enhancements, this release adds aarch64 (aka ARM64) to the family of supported NightStar architectures.

You can expect full NightStar RT functionality on aarch64 with the following caveats:

- CUDA support is not yet available (expected in early summer of 2016)
- Application Illumination (AI) is not yet available. This affects the following tools: nlight, illuminate, and illuminator. AI will be made available in a future update.

2.1. Changes in NightView

2.1.1. New Features

NightView 7.6, a component of NightStar RT 4.5, includes the following new features.

- Reduced disk, memory, and CPU footprint for target systems
- Reduced RPM dependencies for target systems
- Debugging of aarch64 systems, including cross debugging from x86_64
- Unrestricted debugging of 32-bit x86 applications on x86_64 systems

2.1.2. Remote Debugging ssh Connections

When remote debugging, NightView uses **ssh** to connect to the target system. Mulitple connection are required - up to three for each remote session. As such, NightView may prompt you multiple times for authentication information for the same target system. The best way to avoid multiple authentication requests is to use an **ssh** agent (see **ssh-agent(1)**) with pre-loaded authentication keys. If properly configured, NightView would then only prompt you for authentication information if the target system rejects your agent-supplied keys.

An alternative is to use **ssh** control path forwarding. This provides for port forwarding across an active **ssh** connection. Currently NightView uses this feature sparingly, because some recent Linux distributions have problems setting up port forwarding when using control path forwarding; e.g. an initial ssh connection such as **ssh -M -o file** *target*, and subsequent **ssh** port forwarding commands referencing the connection master.

You can fully enable this feature in NightView by setting a debug flag in your NightView session:

```
set-debug use-control-path-forwarding=1
```

When fully enabled, attempts to create the second or third **ssh** connection to the target system may hang. Our experience is that if they do not hang and actually do connect, they operate correctly.

To completely disable control path forwarding, use the following NightView command:

```
set-debug no_ssh_port_forwarding=0
```

We recommend that you use an ssh agent (**ssh-agent(1)**) with pre-loaded authentication keys. If set up properly, you can avoid having to interactively provide authentication information, regardless of the port forwarding scheme in use.

2.1.3. RPM Reorganization

NightView 7.6 changed the names of NightView RPMs from previous releases. The old RPMs:

- **ccur-NightView**
- **ccur-Nviewp**
- **ccur-Nview-ptrace**
- **ccur-NightView-docs-rt**

will be replaced during the installation or update of NightView 7.6 by the following new RPMs:

- **ccur-nview**
- **ccur-nview-target**
- **ccur-nview-${arch-tag}-support**
- **ccur-nview-docs-rt**

Each NightView installation will include its native architecture support package. This allows it to debug native architecture programs locally as well as being able to debug same-architecture programs running on target systems. We call the later "remote debugging". Remote debugging requires that an ssh connection can be made by NightView from the host system to the target system.

For situations where you have constraints on the target system's resources, the only NightStar packages required are **ccur-nview-target** and **ccur-nstar-fs**. You can install these target-side RPMs using the **install-nstar-server** command found in the base directory of the CD:

```
./install-nstar-server --nview
```

If using a repository, install the package called **ccur-NightView-Server**:

```
yum install ccur-NightView-Server
```

Such "server" or "target" RPMs are available for each of the five tools. You can install them individually using commands similar to the above, or you can install all target-side packages via the CD by omitting the **--nview** argument:

```
./install-nstar-server
```

or simply:

```
yum install ccur-NightStar-Server
```

2.1.4. Cross Debugging

We use the term cross-debugging for the case of a remote debugging session between a host and target system with differing architectures. Cross-debugging requires the installation of additional **ccur-nview-*-support** RPMs on the host system.

NightView on i386 supports cross-debugging to x86_64 and aarch64 target systems. To use the cross-debugging features, the host system requires the installation of the **ccur-nview-amd64-support** or **ccur-nview-aarch64-support** RPM, respectively.

Similarly, NightView 7.6 on x86_64 supports cross-debugging to i386 and aarch64 target systems. This requires that **ccur-nview-i386-support** or **ccur-nview-aarch64-support** be installed on the host, respectively. Further, since x86_64 program can execute i386 programs directly, NightView 7.6 supports debugging i386 programs directly on the host as well. The previous NightView restriction of requiring a separate 32-bit shell for i386 debugging has been lifted. You can now even debug i386 programs that are `exec`'d (see **exec(2)**) from x86_64 programs, and vice versa. This mixed-architecture debugging requires the installation of **ccur-nview-i386-support** on the host x86_64 system.

At the current time, NightView 7.6 on aarch64 systems does not support cross-debugging to i386 or x86_64 targets. However, you can still do remote debugging from an aarch64 host to an aarch64 target system.

2.2. Changes in NightTrace

NightTrace 7.5, a component of NightStar RT 4.5, includes the following new and improved features:

2.2.1. Support for aarch64

NightTrace can now operate on user data or kernel data logged on aarch64 systems.

You can analyze the data with the exact same techniques and session files as the same user interface is provided; the addition of aarch64 was done seamlessly.

NOTE

Application Illumination (nlight) was not available for aarch64 at the time of this release. It will be available in a future update of NightTrace 7.5

2.2.2. Improved PID Tracking for Kernel and User Trace Data

The ability to provide a program name along with a process ID greatly enhances the readability of trace data.

In the past, process names were available for a majority of trace events, but there were many cases where only the PID was available. Such cases included processes seen at the start of kernel tracing before any context switching had occurred as well as processes that were started after kernel tracing started but exited before kernel tracing stopped.

With this release, NightTrace has enhanced process name determination. Most all events now have process names as well as PID values. This includes user events when accompanied by kernel events.

However, there are some limitations. Much of process name determination happens retroactively, as more trace data is consumed.

While using ntrace to view trace data from files, you should encounter few events without process names.

However, when streaming trace events, there will be times initially where the process name may not be available, but those will subsequently be identified as more trace data is consumed.

The NightTrace Analysis API has a similar limitation. Full process name determination cannot be completed until data from the end of a trace is consumed. Thus if you want to ensure that all process names are determined, you should use code similar to the following:

```
#include <ntrace_analysis.h>
...
tr t = tr_init()
...
-- Ensure the end of trace data has been seen
while (tr_next_event(t)!=TR_EOF){}
tr_seek(t,0);
...
-- Now install your conditions and states and callback
-- functions and start processing for real.
...
```

While this is wasteful in terms of processing, it is effective. A more elegant solution is being considered for the future.

2.3. Changes in NightTune

2.3.1. New Features

NightTune 3.7, a component of NightStar RT 4.5, has the following new features.

- Support of CUDA 7.5 on x86_64 systems. Support for CUDA on aarch64 will be made available in a future update.
- Addition of a PCI Panel which graphically shows the layout of PCI devices, ports, bridges, and buses. You can easily see devices which share IRQs by simply selecting a device in the panel -- those that share an IRQ will be highlighted in red. The panel also provides users insight into how a specific PCI device might be affected by other PCI traffic.

2.3.2. RPM Renaming

NightTune 3.7 renamed a few of its RPMs, making the previous names obsolete. RPMs **`ccur-ntuneserv`** and **`ccur-ntunecommon`** have been replaced with **`ccur-ntune-server`** and **`ccur-ntune-common`**, respectively.

During installation the old RPMs will be replaced with the new RPMs.

2.4. Changes in NightProbe and NightSim

2.4.1. Maintenance Release

NightProbe 4.3 and NightSim 4.4, both components of NightStar RT 4.5, did not change significantly from their previous versions. The new releases contain bug fixes and adjusted RPM dependencies.

3.0. Software Installation

Follow the instructions under “Installing NightStar RT” on page 6 to install NightStar RT on your system.

Then take a look at “NightStar RT GUI Features” on page 13 to learn about the new graphical interface of the NightStar RT tools.

3.1. Prerequisites

Prerequisites for NightStar RT Version 4.5 for both the host system and target system are as follows:

3.1.1. Host System

Any of the following distributions:

- RedHawk 2.3 or higher
- Red Hat Enterprise 3-7
- CentOS 5-7

3.1.2. Target System

Any of the following distributions:

- RedHawk 2.3 or higher

3.2. 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 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 when logged in as your normal user:

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

3.2.1. 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.

IMPORTANT

Do not attempt to update or install the following obsolete RPMs:

- ccur-NightView, ccur-NightView-docs-rt, ccur-Nviewp, ccur-Nview-ptrace
- ccur-ntunecommon, ccur-ntuneserv
- ccur-ntracelog, ccur-ntraceapi
- ccur-nsimserver
- ccur-nprobeserv

These have been replaced by RPMs with new RPMs with different names. Normally, all these obsoleted RPMs will be removed during a CD installation or yum/nuu update in favor of the newly required RPMs.

If the installation fails because **`libstdc++.so.5`**, or other RPMs, are not available, install the **`compat-libstdc++-33`** RPM which is readily available with CentOS and exists in an optional repository for Red Hat Enterprise users. You may also be able to locate the RPM from an older Linux system or distribution media.

If you experience any problems during the update or installation process, please contact Concurrent support (see “Direct Software Support” on page 39).

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
```

3.2.1.1. 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 all of its commercial real-time 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). The $basearch notation above literally means exactly those characters, including the $ character. $basearch is a YUM variable which is provided by YUM (or NUU) at runtime.

3.2.2. CD Image Installation

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

- Insert the *NightStar RT Installation CD* in the CD-ROM drive; on newer systems it will automatically mount at one of the following locations:
  - **/media/NightStar-RT-4.5**
  - **/run/media/{user-name}/NightStar-RT-4.5**
- If the CD does not auto-mount, mount the CD-ROM drive in a manner similar to the following:

```
[ -d /mnt/cdrom ] || mkdir /mnt/cdrom;
mount -t iso9660 -o ro /dev/sr0 /mnt/cdrom
```

  Your CD device may be something other than **/dev/sr0**.

- Double-click on the CD icon including the notation NightStar RT 4.5 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 or the icon fails to appear on your desktop, change the current working directory to the directory where the CD is mounted and invoke the following script: **./install-nstar**.

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 a command similar to the following:

mount -o exec,remount *mountpoint*

NOTE

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

NOTE

If the installation fails due to an unmet dependency, retry the installation with an additional option by running the following command while positioned in the directory where the CD is mounted:

./install-nstar --no-disable

By default, the installation instructs yum to ignore all RPM repositories except the one included natively on the CD.

The **--no-disable** option, as shown above, allows (enabled) network repositories to participate in meeting RPM requirements.

If the install still fails due to an unmet dependency on **libstdc++.so.5**, install the **compat-libstdc++-33** RPM which is readily available from CentOS installation media or network repositories and exists in an optional repository with Red Hat Enterprise. You may also be able to locate the RPM from an older Linux distribution to which you have access.

If you experience any problems during the update or installation process, please contact Concurrent support (see "Direct Software Support" on page 39).

- 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
```

3.3. Obtaining License Keys

The NightStar RT Version 4.5 software uses the same license manager and license features as the previous version of NightStar RT 4.*. 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 30 for more detailed information about the NightStar License Manager (NSLM).

3.4. Removing NightStar RT

To remove NightStar RT, mount the CD or ISO image as described in "CD Image Installation" on page 8 and execute the following command as root:

```
./remove-nstar
```

4.0. Documentation

The following table lists the NightStar RT 4.5 documentation available from Concurrent.

NightStar RT Version 4.5 Documentation

Manual Name	Pub. Number
NightStar RT Installation Guide (Version 4.5)	0898008-4.5
NightStar RT Tutorial (Version 4.5)	0898009-100
NightProbe User's Guide (Version 4.2)	0898465-108
NightSim User's Guide (Version 4.3)	0898480-075
NightTrace User's Guide (Version 7.4)	0898398-190
NightTune User's Guide (Version 3.6)	0898515-044
NightView User's Guide (Version 7.5)	0898395-360
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*
- from the Concurrent Software Library online at http://redhawk.ccur.com/docs
- 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`**

5.0. NightStar RT GUI Features

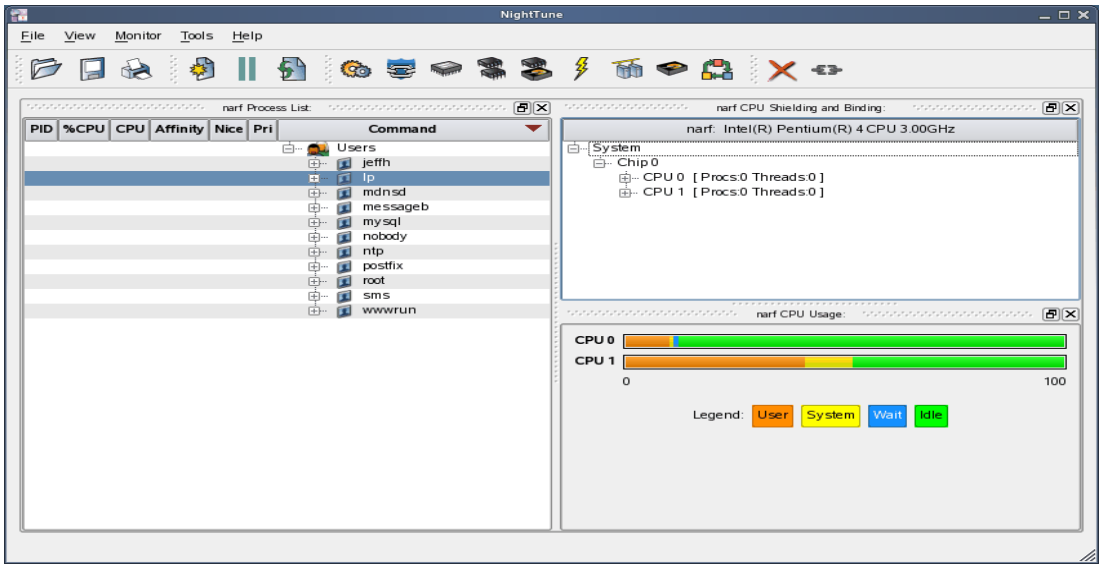
Some of the common features of the NightStar RT Tools graphical user interface include:

- movable and resizable panels
- tabbed pages
- context menus

5.1. Movable and Resizable Panels

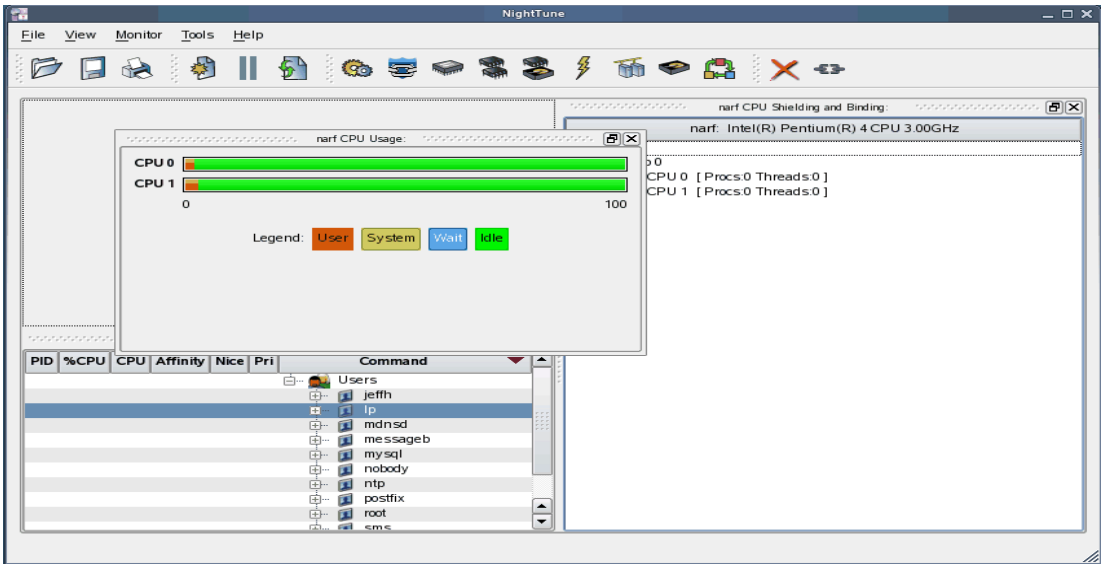
The NightStar RT Tools allow the user flexibility in configuring the graphical user interface to suit their needs through the use of resizable and movable panels.

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

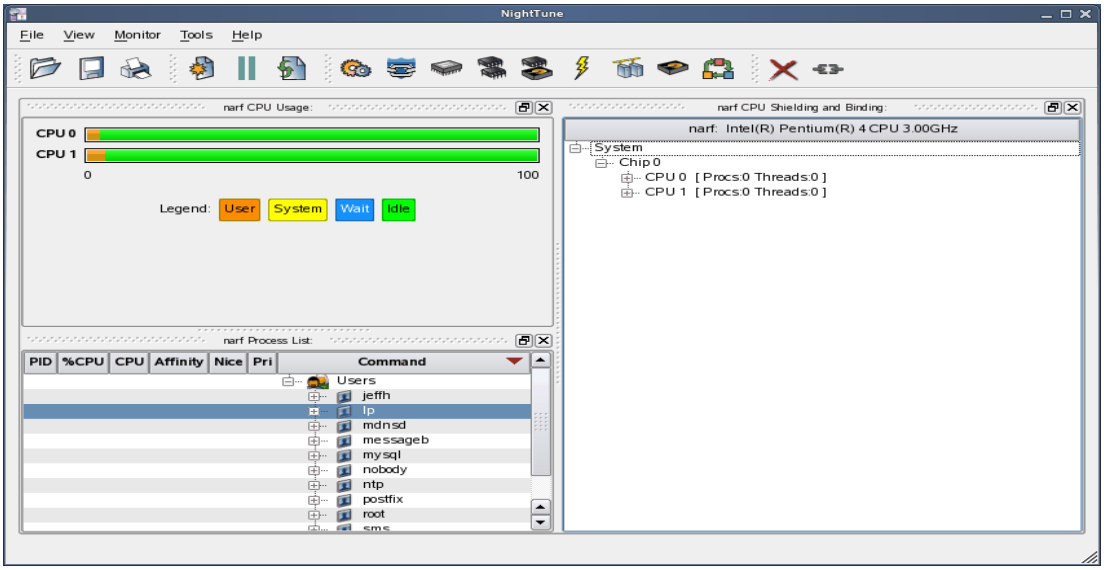


To move one of the panels in the current page, left-click on the title bar for the panel you wish to move and begin to drag the panel to the desired location. The application will respond by creating space on the page based on where you move the panel while resizing and moving the other panels accordingly.

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:

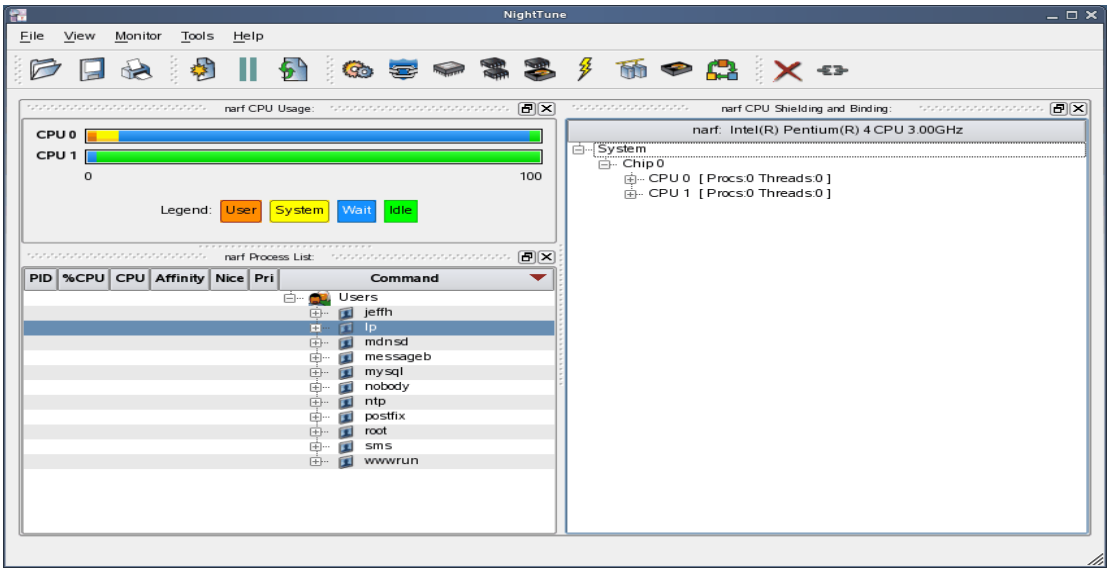


Release the mouse button when NightTune has opened a space where you desire and NightTune will place the panel in that location. The CPU Usage panel now resides in the upper left corner of the NightTune display.



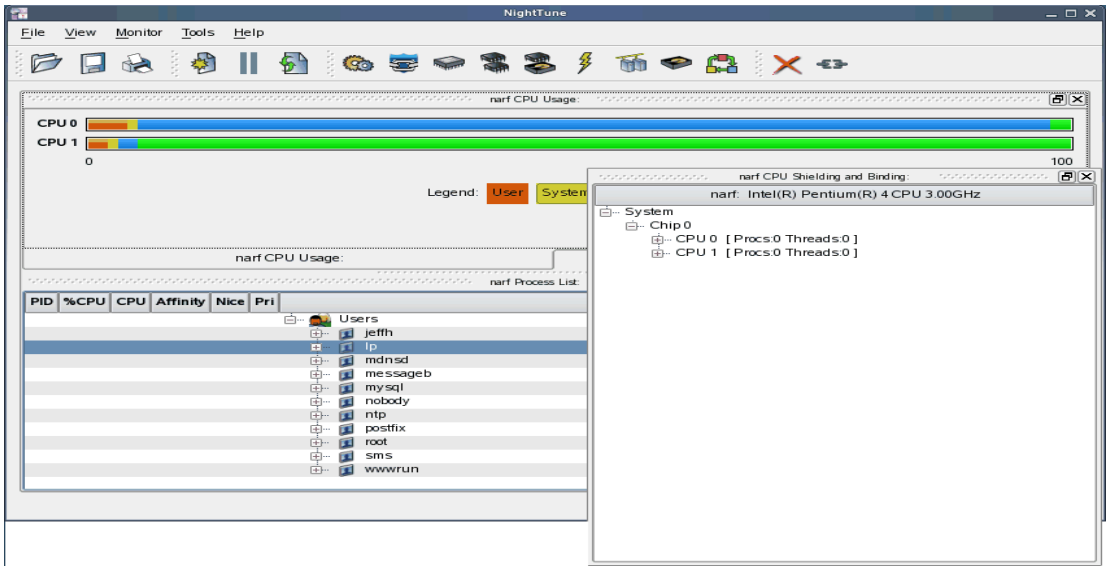
If an empty space does not appear where you desire it, try increasing the size of the main window, decreasing the size of the undocked panel, and moving an alternative edge of the undocked panel near where you want to place it.

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.

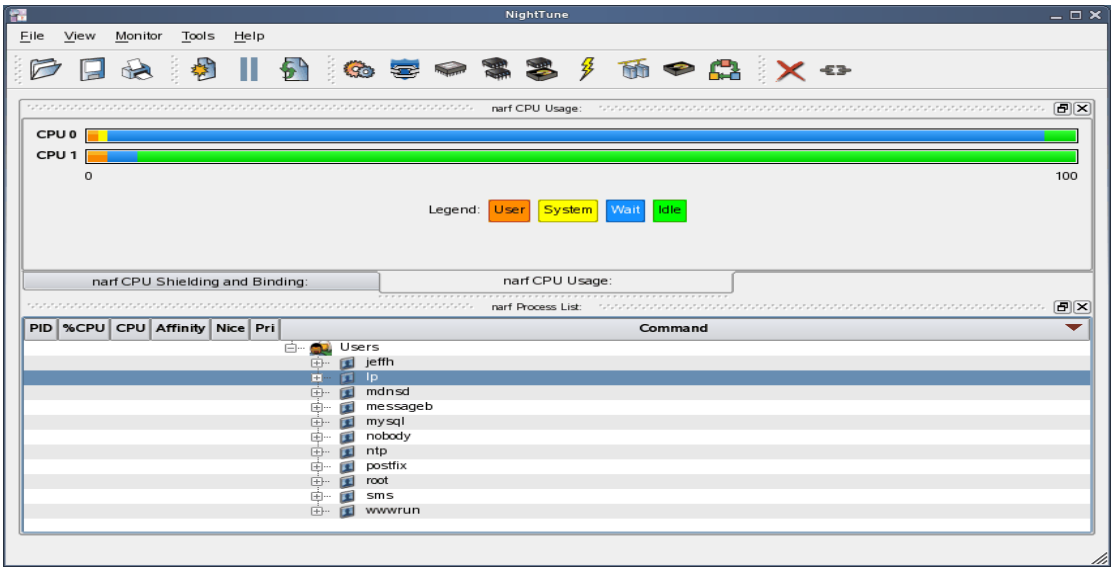


Another feature of the NightStar RT Tools graphical user interface is the use of tabbed panels. Tabbed panels allow you to maximize your GUI real estate by placing two or more panels in the same location. You can then switch between the panels using the tabs created.

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.

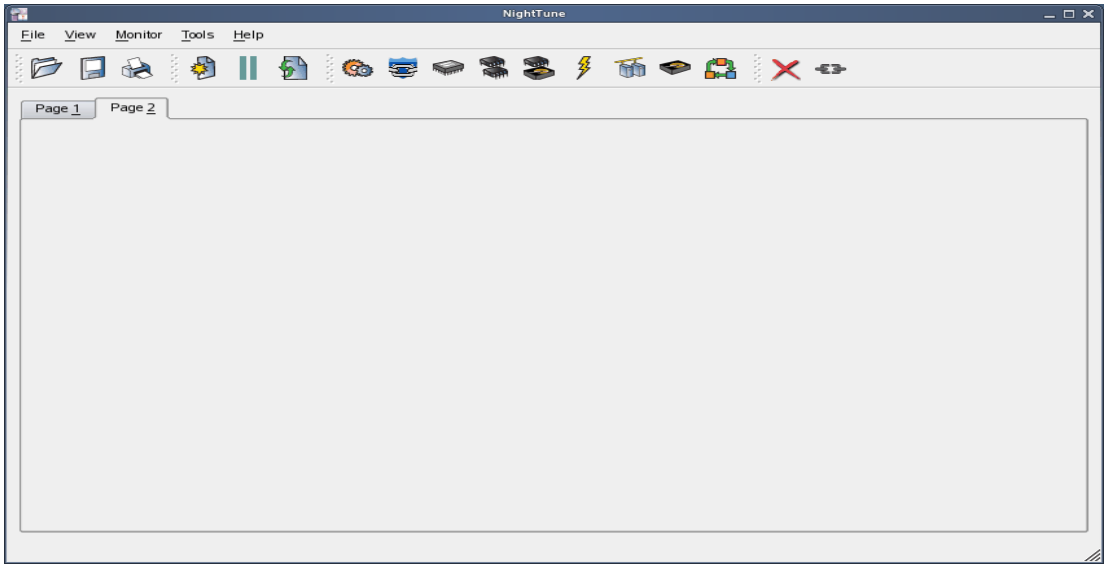


5.2. Tabbed Pages

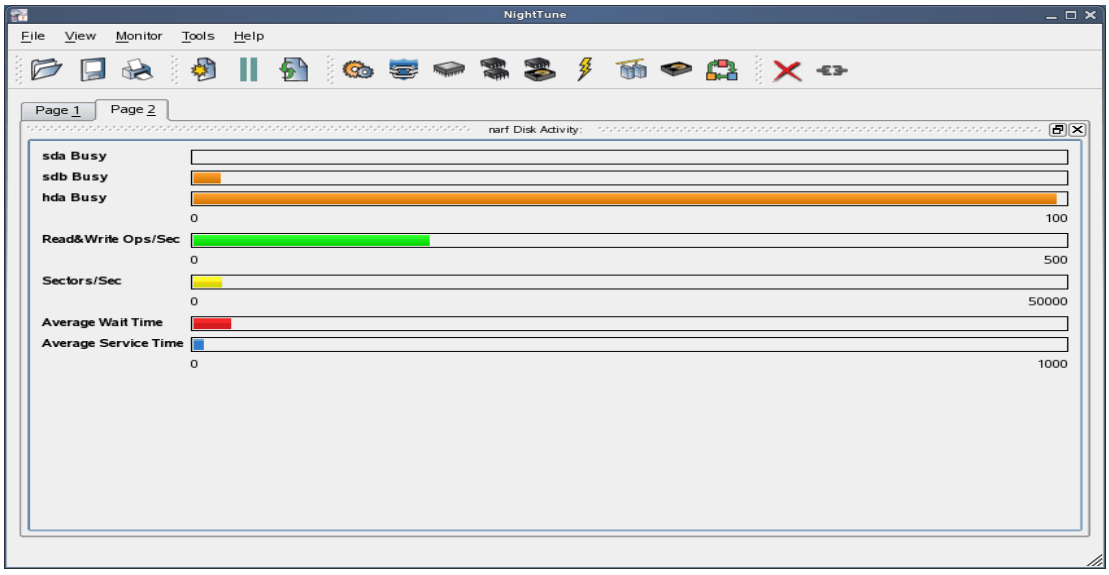
The NightStar RT Tools allow the user to maintain multiple views of data and the mechanisms that manipulate that data within each application through the use of tabbed pages. By default, only one page is displayed when the tool is invoked.

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.

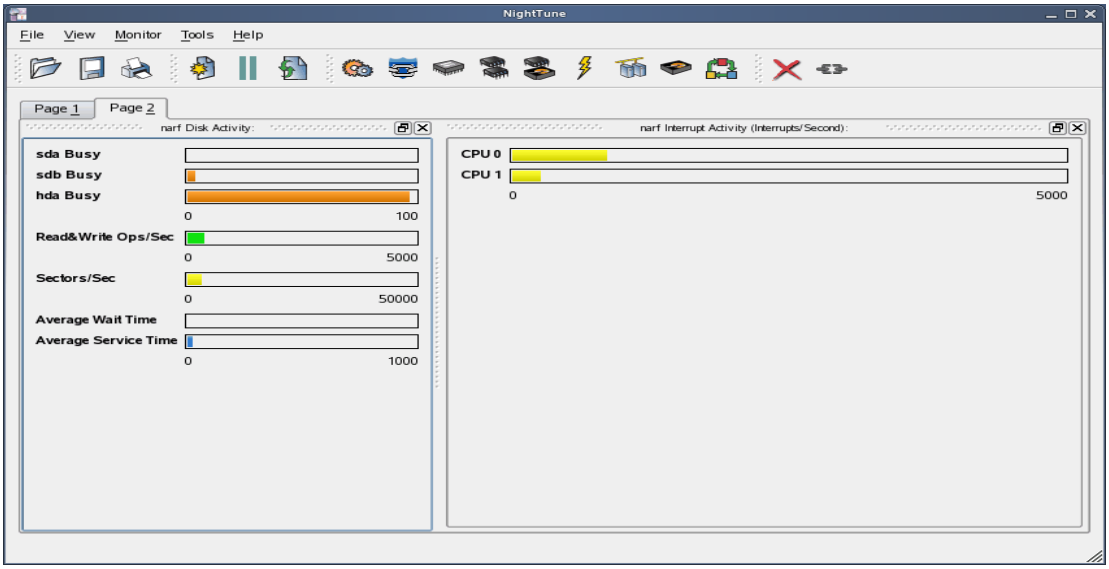
Select Add Page from the View menu. NightTune will create two tabbed pages; our original page is placed under the first tab and a new empty page will be presented under the second.



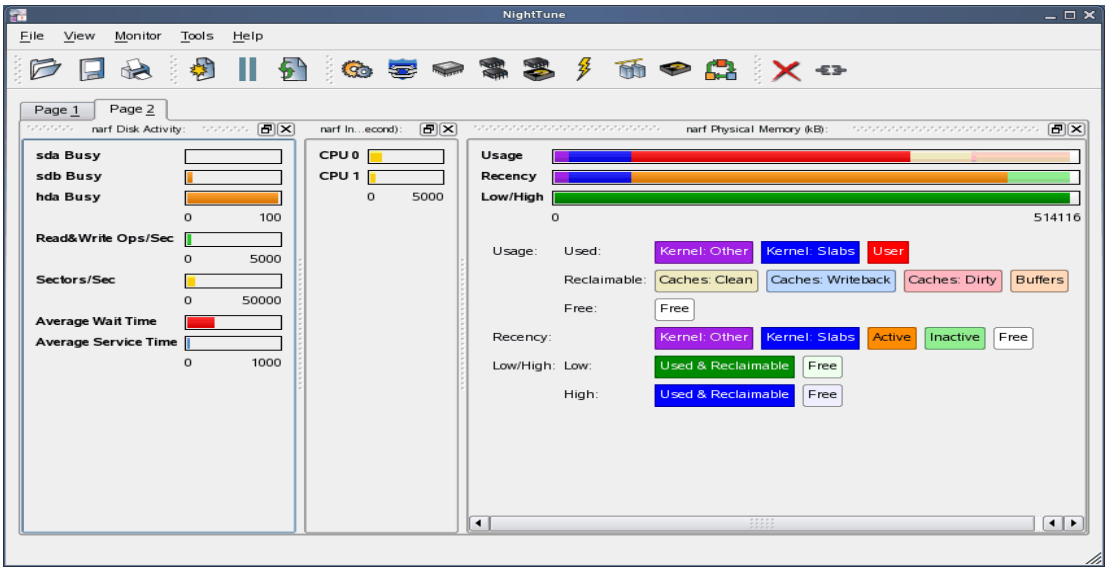
To add the desired NightTune panels, click on the Monitor menu item. You will be presented with a menu of panels to choose from. Select the Disk Activity menu item and then select Bar graph pane from the sub-menu. The Disk Activity panel displaying the information in bar graph format is added to our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



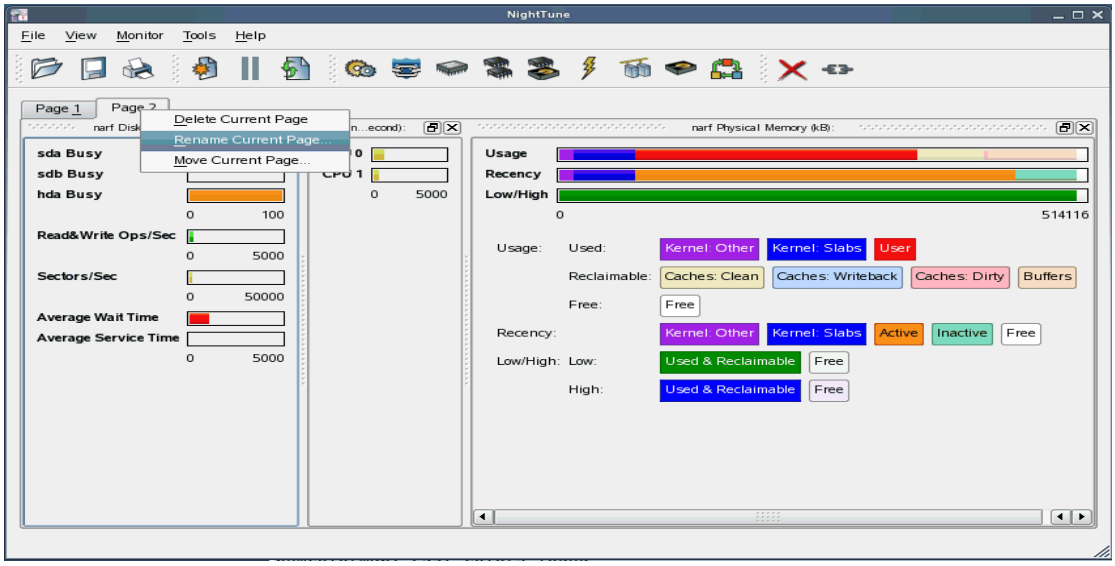
Our new page now contains the Disk Activity, Interrupt Activity, and Memory Physical panels all displaying their information in bar graph format. We can switch back to our first page by clicking on the tab labeled “Page 1” and return to our new page by clicking on the tab labeled “Page 2”.

5.3. Context Menus

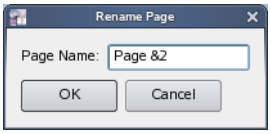
The NightStar RT Tools provide extensive use of context menus. Right-clicking in any of the NightStar RT Tools will provide the user with a menu containing items related to the location of the mouse in the tool.

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 16 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.

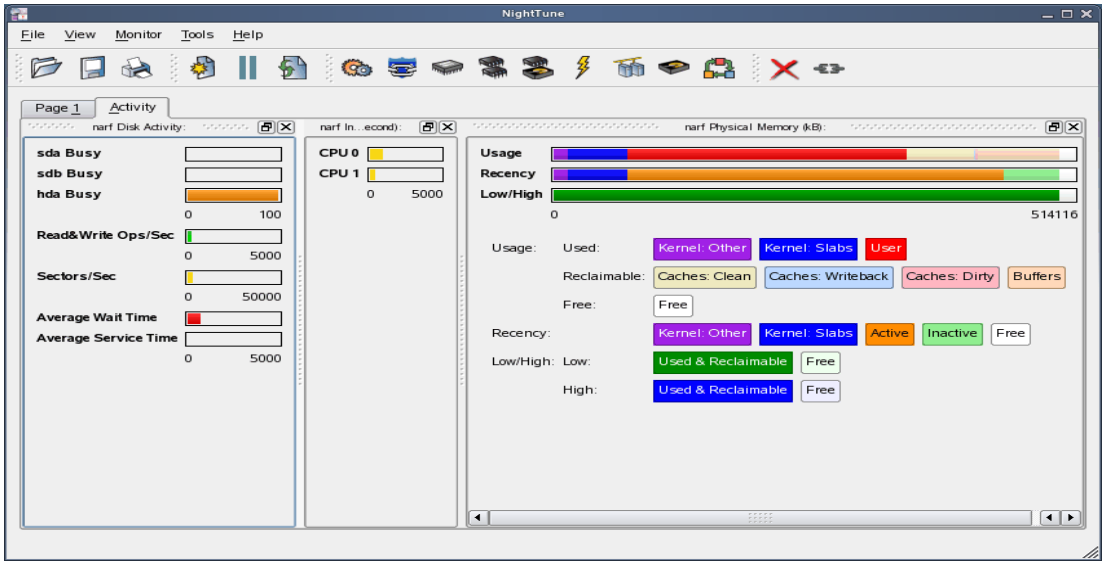


Change the Page Name to "&Activity".

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.



6.0. 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

6.1. 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, prototyping 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.

6.2. 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.

6.3. 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
- Automated instrumentation of user code

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.

NightTrace's Illumination tool (nlight) automatically instruments user code (executable images or .o files) with trace points for the entry and return of every function (the user has control over which functions to illuminate). It does this without modifying the executable images or .o files outright. NightTrace provides a description of the function call, including the values of all arguments, and the return value.

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.

6.4. 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.

6.5. 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.

6.6. 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.

6.7. 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.

7.0. 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/share/doc/NightStar/pdf`** 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
```

from the command line.

7.1. 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 interrupts 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.
pass-word-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.

7.1.1. Allowing NightView to Attach to Your Processes

By default, newer distribution of Linux restrict prevent you from attaching a debugger to your running process, even if you invoke the debugger with the same uid and gid as the process which you want to debug.

You can remove this restriction by using the **sysctl** command to change the value of the variable which represents the restriction. To remove the restriction, enter the following command from a shell:

```
sysctl -w kernel.yama.ptrace_scope=0
```

Once you have issued that command, log out and log in again and the restriction will be lifted. However, the setting is only effective until the next reboot. You may want to put the command above in **/etc/rc.local** (sometimes **/etc/rc.d/rc.local**) so that is applied every time the system boots.

8.0. 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 31) from a license server (see "License Server" on page 31).

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 32).

The default operating system configuration may include a strict firewall which may interfere with floating licenses. See "Firewall Configuration for Floating Licenses" on page 32 for information on handling such configurations.

8.1. 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. (See "Selecting a Network Device for Licensing" on page 31 for more information).

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 39 for more information).

8.1.1. Selecting a Network Device for Licensing

By default, **nslm** queries the system and examines all available network devices. **nslm** prefers real network devices over virtual ones, because the latter sometimes are not consistently available.

You can see the list of network devices that nslm can use, using the following command:

```
nslm_admin --devices
```

By default, **nslm_admin** picks the first device in the list.

If you wish to specify a specific device to be associated with your license key, you can add the **--device** option when obtaining your code:

```
nslm_admin --device=eth2 --code
```

8.2. 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 IP address of 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:server2:10.134.30.0
```

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**.

8.3. 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 one of following command, depending on you OS version:

- **/usr/bin/systemctl status nslm**
- **/sbin/chkconfig --list nslm**

In rare instances, you may need to restart the license server via one of the following commands:

- **/usr/bin/systemctl restart nslm**
- **/sbin/service nslm restart**

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

8.4. 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.

8.5. 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 32 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 33 for more information.

8.5.1. 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
```

8.5.2. 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
```

8.6. License Support

For additional aid with licensing issues, contact the Concurrent Software Support Center. See “Direct Software Support” on page 39 for details.

9.0. 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 describe the interoperability of each tool between 32-bit x86 and 64-bit x86.

NightProbe

No limitations.

NightSim

No limitations.

NightTune

No limitations.

NightTrace

Limitations

- NightTrace cannot control remote system tracing unless the host and target system have use the sambe bit-size for addresses. Thus you cannot connect NightTrace to a 32-bit system from a 64-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.

Inter-architecture Capabilities in NightStar RT 4.5

- 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!). Similarly, since x86 (32-bit or 64-bit) systems cannot execute aarch64 programs, NightView cannot debug them directly on the x86 system.

In previous versions, NightView required the --arch=i386 option in order debug 32-bit applications on a 64-bit machine. That restriction has been lifted. The --arch=i386 option has no effect in this release -- it is silently ignored. You can debug 32-bit x86 programs on 64-bit x86 systems freely, even intermixing such programs in same shell or `exec`'d (see **`exec(2)`**) from dissimilar programs.

In this release, NightView supports cross-debugging from x86 systems to aarch64 systems. See "Changes in NightView" on page 2 for more information.

10.0. Known Problems

The following sub-sections list known issues with NightStar.

10.1. Problem: Unable to attach to the target system

Several NightStar tools need to talk to their server processes in order to operate.

By default, they will attempt to locate their server processes on the local system using the value returned by the following command:

```
hostname
```

If there is no mapping to the hostname, these tools will fail.

Solution:

Ensure that **`/etc/hosts`** contains a mapping of the hostname to a valid IP address or that the mapping is made available by DNS or other means.

10.2. Problem: Unable to debug on VirtualBox Systems

VirtualBox incorrectly manages the debug registers of the underlying chip. This causes NightView to fail when starting to debug a process. This bug in Virtual Box has been reported for several years, and appears to continue to exist.

Solution:

Concurrent recommends KVM as an alternative solution to VirtualBox.

11.0. 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**.