NightStar RT Installation Guide

Version 5.1

(RedHawk™ Linux®)

concurrent REAL-TIME

0898008-5.1
June 2023

Contents

1.0. Introduction

NightStar RT Version 5.1 is a production release of NightStar RT running under Concurrent Real-Time's RedHawk Linux.

NightStar RT version 5.1 requires modern Linux distributions, as outlined in "Host Systems" on page 2 and "Target Systems" on page 2.

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.

Installation instructions have changed in NightStar 5.1. Pay particular attention to URLs used in network installation as described in "Changes in NightStar RT 5.1" on page 11 and "Network Installation for Ubuntu" on page 8, as they have changed.

For a general description of NightStar, refer to "Overview of NightStar RT" on page 24.

IMPORTANT!

There are limitations on NightStar functionality when used with RedHawk 9.2.

Please see "Known Problems" on page 44 for details.

2.0. Supported Systems

Prerequisites for NightStar RT Version 5.1 for both the host system and target system are as follows:

2.1. Host Systems

Any of the following Linux distributions:

- Concurrent Real-Time RedHawk Linux 7.5 - 9.2
- Red Hat Enterprise 7.5 - 9.2
- CentOS 7.5 - 8.4
- Rocky Linux 8.4 - 9.2 (NightProbe and Datamon have significant limitations on 9.2. See "Known Problems" on page 44 for details.)
- Oracle Linux 9.2 (NightProbe and Datamon have significant limitations on 9.2. See "Known Problems" on page 44 for details.)
- Ubuntu 16.04 - 22.04 (NightProbe and Datamon have significant restrictions on Ubuntu 20.04 and 22.04. See "Known Problems" on page 44 for details.)
- Debian 10-11

running on Intel and AMD x86_64 processors and selected ARM64 systems.

Native Intel 32-bit support is no longer available; however, you may build and analyze 32-bit programs on a 64-bit host machine.

NightStar supports 32-bit Intel programs on x86_64 hardware.

2.2. Target Systems

RedHawk Linux is required for all target systems.

NightStar supports any Intel or AMD x86_64 systems running the RedHawk Linux kernel versions 7.5 - 9.2.

The following ARM64 systems are supported:

- From NVIDIA™ Corporation, the Jetson TX1, TX2, PX2-Drive, AGX-Drive, NX-Xavier, AGX-Xavier, AGX Orin, and Nano development kits.

3.0. Installing NightStar RT

There are three methods of installing NightStar RT.

- DVD Installation
- Network Installation for RHEL, CentOS, Rocky Linux, and Ubuntu Linux
- Network Installation for Ubuntu

IMPORTANT

Before actual installation, it is highly recommended that you read the entirety of the installation method you have selected

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
```

The remainder of the commands in this section require that you run them as the root user.

3.1. DVD Installation

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

- Insert the *NightStar RT Installation DVD* in the DVD-ROM drive; on newer systems it will automatically mount at one of the following locations:
  - **/media/NightStar-RT-5.1**
  - **/run/media/{user-name}/NightStar-RT-5.1**
- If the DVD does not auto-mount, mount the DVD-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 DVD device may be something other than **/dev/sr0**.
- Change the current working directory to the directory where the DVD is mounted and invoke the following script as the root user:

  ```
  ./install-nstar
  ```

  During installation on an x86_64 system, you will be asked if you wish to have 32-bit APIs and debugging capabilities installed. This is completely optional. If you choose not to at this time, you can always install them by rerunning the install-nstar script in the future.

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 DVD and invoke the

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

script instead of the install-nstar script.

3.2. Network Installation for CentOS and Red Hat

3.2.1. Remove Old NightStar Versions

If you have NightStar 4.8 or older already installed on your system, you must remove it first. The **remove-nstar** script is provided for this purpose and can be found in the base directory of the DVD and also online at https://redhawk.concurrent-rt.com/NightStar/remove-nstar. Execute the script as the root user.

```
bash ./remove-nstar
```

The script may leave a few NightStar packages on the system. If this happens, it will tell you why and which packages will remain. This is fine and you can proceed from this step.

The md5sum check sum for the **remove-nstar** script is 8b39af5a576641a3d784d6ac0d3f3d8f.

3.2.2. Installing via NUU, yum, or dnf

Network installation is accomplished via https://redhawk.concurrent-rt.com using any of the following commands: **nuu**, **yum** or **dnf**. **nuu** is Concurrent Real-Time's Network Update and installation Utility.

Use of **nuu** is preferred because it manages your **redhawk.concurrent-rt.com** login and password required to access the repositories.

If you do not have NUU installed on your system but wish to use it, visit the following URL and follow the instructions on installing and configuring NUU and then return to the section below:

https://redhawk.concurrent-rt.com/network/yum.html#NUU

3.2.2.1. Yum Configuration File

Regardless of the command you use, the following configuration file must be present on your system:

```
/etc/yum.repos.d/ccur-nstar-rt.repo
```

Create the file if it does not yet exist. Edit the file so that it has the proper content as described below, replacing the colored italicized text with your specific information.

```
[ccur-nstar-rt]
name=NightStar RT
baseurl=https://redhawk.concurrent-rt.com/rhel/Login=login/Password=password/Night-
Star/RT/version/distro/$basearch
gpgcheck=0
enabled=1
```

IMPORTANT

The critical line in the file is the one that starts with “baseurl” and ends with “$basearch”. It must be a single line without any spaces, hyphenation or line breaks (regardless of how it might appear in this document).

The bold ***green*** portions above must be replaced with your redhawk.concurrent-rt.com ***login*** ID and ***password*** assigned to you by Concurrent Real-Time. That information is included in a cover letter which shipped with your system and/or software. The Login ID is also your Site ID, which typically starts with LA or LR followed by 5-7 decimal digits. If you cannot readily locate these values, please contact Concurrent Real-Time Software Support as shown in the support section entitled “Direct Software Support” on page 47. You can set the "enabled" variable on the last line of the file to 0 until you are able to fill in the information.

The ***version*** component in italicized bold red is the NightStar version, and must be one of the following:

5.1 or ***current***. Older versions (***4.1***–***4.6***, ***4.8***, or ***5.0***) are also available.

Use ***5.1*** for this release of NightStar. Alternatively, use the value ***current***. It is a placeholder for the most current production release. Using ***current*** for the value is recommended so that you will always have access to the latest production release of NightStar; e.g. when NightStar 5.2 is released.

NOTE:

In previous releases, the value ***RedHawk*** was used in place of ***current***; the former handle is being deprecated.

In NightStar version 5.1, the base URL must include the ***distro*** component -- a Linux distribution abbreviation. Use one of the following values from the table below that best describes your system’s distribution:

Linux Distribution Abbreviations

Distribution	Abbreviations
CentOS	***rhel7, rhel8***
RHEL	***rhel7, rhel8, rhel9***
Rocky Linux	***rhel8, rhel9***
Oracle Linux	***rhel9***
Ubuntu	***ubu16, ubu18, ubu19, ubu20, ubu22***
Debian	***deb10, deb11***

IMPORTANT:

If specifying a NightStar version prior to 5.0, omit the /***dis-tro*** component completely.

Generally, you should not replace the characters $basearch with any other value. The URL should end in $basearch, which is a YUM variable which is automatically set by yum utilities at runtime.

Your configuration file might look like the following:

[ccur-nstar-rt]
name=NightStar RT
baseurl=https://redhawk.concurrent-rt.com/rhel/Login=***LA12345***/Password=***secret***/Night-Star/RT/***5.1***/***rhel8***/$basearch
gpgcheck=0
enabled=1

IMPORTANT

The critical line in the file is the one that starts with "baseurl" and ends with "$basearch". It must be a single line without any spaces, hyphenation or line breaks (regardless of how it might appear in this document).

3.2.2.2. Installation via NUU

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

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

If not running as root, NUU will prompt you first for the root password. Once NUU starts processing, you may be prompted for your redhawk.concurrent-rt.com login and password information. This is especially true if you haven't run NUU before.

If NightStar RT is not already installed you will need to change NUU's mode to *Installable*, as it defaults to *Updateable*. This setting can be selected from a drop-down menu located in the upper right quadrant of the main window. Once set, you should be able to locate **`ccur-NightStar-RT-RedHawk`** in the list of packages. Select that package for installation and click the **`Apply`** button. That package's dependencies will cause the entire NightStar RT product to be installed, except for optional 32-bit support. Installation of 32-bit support is handled separately below.

If NightStar RT is already installed on your system, any out of date packages will be in the updateable list on the screen. Select them for update and click the **`Apply`** button.

IMPORTANT

When updating be careful. NUU installs software from the underlying distribution as well as NightStar. Make sure you are not accidentally updating your entire system unless that is your intention. After hitting **Apply,** NUU will pop up a dialog that will show you exactly which packages will be updated. Carefully examine that list before proceeding. All Concurrent Real-Time packages start with a ccur- prefix.

The QuickStart.pdf document included in the NUU download kit obtained from **https://redhawk.concurrent-rt.com/network/yum.html#NUU** explains how to use NUU in more detail.

If you wish to install 32-bit NightStar support, use NUU to explicitly install the

```
ccur-nstar-32bit-support
```

RPM. This RPM will cause five additional RPMs to be installed so that you have full support of 32-bit APIs and debugging on an x86_64 system.

3.2.3. Installation via YUM and DNF

On modern RPM-based systems, the **yum** command has been replaced by the **dnf** command. In terms of the installation commands below, use whichever tool that is installed on your system.

IMPORTANT

If you have a NightStar version prior to 5.0 already installed, you must remove it before proceeding. See “Remove Old NightStar Versions” on page 4.

3.2.3.1. Installation

To install NightStar RT on a system that does not have NightStar, use the following command:

```
yum --disablerepo=ccur-* --enablerepo=ccur-nstar-rt install \
ccur-NightStar-RT-RedHawk
```

3.2.3.2. Updating NightStar in the Future

Concurrent provides updates to NightStar as required. Use the following command to update NightStar on a system that already has NightStar installed.

```
yum --disablerepo=ccur-* --enablerepo=ccur-nstar-rt update \
   ccur-NightStar-RT-RedHawk 'ccur-n*' ccur-qt5 \
   ccur-datamon ccur-shmdefine
```

3.2.3.3. Additional Network Installation Issues

NOTE

Do not attempt to update or install the following obsolete packages: ccur-NightView, ccur-NightView-docs-rt, ccur-Nviewp, ccur-Nview-ptrace, ccur-ntunecommon, ccur-ntuneserv, ccur-ntracelog, ccur-ntraceapi, ccur-nsimserver, ccur-nprobeserv, ccur-ntrace-api-libs, ccur-ntrace-java-api-libs or ccur-nprobe-api-libs.

After installation, you can keep your system up to date interactively using the update command shown above. Perhaps you might even install a **cron** job to update your system nightly; e.g.:

```
1 0 * * * /usr/bin/yum -y --disablerepo=* \
--enablerepo=ccur-nstar-rt update ccur-NightStar-RT-RedHawk \
     'ccur-n*' ccur-qt5 ccur-datamon ccur-shmdefine
```

If you wish to install 32-bit NightStar support, explicitly install the **ccur-nstar-32bit-support** RPM, using the following command:

```
yum install --disablerepo=ccur-* --enablerepo=ccur-ntar-rt \
   ccur-nstar-32bit-suppport
```

That RPM will cause five additional RPMs to be installed so that you have full support of 32-bit APIs and debugging on an x86_64 system.

3.3. Network Installation for Ubuntu

IMPORTANT

If you have a NightStar version prior to 5.0 already installed, you must remove it before proceeding. See “Remove Old NightStar Versions” on page 4.

Installation of NightStar RT on Ubuntu and Debian systems is easily accomplished using **apt** -- the standard installation method for those distributions.

Concurrent Real-Time’s public key should be added to the apt key ring using the **apt-key** command as shown below. Note that the public key file is also present on the Installation DVD.

```
wget https://redhawk.concurrent-rt.com/network/ccur-public-keys
apt-key add ccur-public-keys
```

On the more recent Ubuntu releases, **apt-key** will issue warnings about it being obsolete. It is safe to ignore those warnings. This will be addressed in a future NightStar release.

3.3.1. APT Repository Configuration File

The URL for the NightStar RT repository definition must be customized with your **redhawk.concurrent-rt.com** login information as well as your specific Linux distribution.

Create a file called **ccur-nstar.list** in the directory **/etc/apt/sources.list.d** with the following content, substituting the italicized colored text with information specific to your system:

```
deb [arch=myarch]\
   https://redhawk.concurrent-rt.com/ubuntu/login/password/nightstar \
   5.1 rt distro
```

IMPORTANT

The repository definition line starting with "deb" and ending in "*distro*" described above must be on a single line, regardless of how it is displayed above. Remove the backslash characters and keep single spaces between the 6 components.

The *myarch* item above must be replaced with either amd64 or arm64.

The *login* and *password* fields are your login and password for redhawk.concurrent-rt.com. These values are documented on the cover letter you received with your system or software purchase. The *login* portion is also called your site ID. Site IDs typically start with LA or LR, followed by 5-7 decimal digits.

The line should include the *distro* component -- a Linux distribution abbreviation. Use one of the following values that best describes your base operating system:

Linux Distribution Abbreviations

Distribution	Abbreviations
Ubuntu	*ubu16, ubu18, ubu19, ubu20, ubu22*
Debian	*deb10, deb11*

IMPORTANT

If specifying a NightStar version prior to 5.0, omit the *distro* component completely.

A common deb definition line would look like as follows:

```
deb [arch=amd64] https://redhawk.concurrent-rt.com/ubuntu/LA12345/sec
ret/nightstar \ 5.1 rt ubu20
```

all on one line, without the backslash (\) character and single spaces separating the six components.

3.3.2. APT Installation Commands

IMPORTANT

If you have a NightStar version prior to 5.0 already installed, you must remove it before proceeding. See "Remove Old NightStar Versions" on page 4.

Force the **apt** system to reread the repository source list. Issue the following command (do not be concerned about the word update, the command will not change any packages as part of this step):

```
apt-get update
```

If NightStar has not yet been installed, issue the following command:

```
apt-get install ccur-nightstar-rt
```

However, if NightStar is already installed on your system, you need to issue a more specific command, as shown here:

```
apt-get install --only-upgrade "ccur-n*" ccur-qt5 ccur-datamon
ccur-shmdefine
```

3.3.3. NightStar 32-bit Support on Ubuntu

NightStar still supports 32-bit programs being built, run, and debugged on 64-bit systems.

In order to install 32-bit packages on an x86_64 system you may need to add i386 to the list of supported package architectures. You can check to see if your system already allows i386 packages to be installed by issuing the following command:

```
dpkg --print-foreign-architectures
```

To add i386 to the list of architectures, execute the following command:

```
dpkg --add-architecture i386
```

Once you've ensured the system has i386 support, run the following two commands

```
apt-get update
apt-get install ccur-nstar-32bit-support
```

The first command, apt-get update, does not install any packages on your system. It only updates database information about all the repositories that are defined.

The second command will install the ccur-nstar-32-bit-support package as well as five dependent packages.

4.0. Changes in NightStar RT 5.1

NightStar 5.1 is primarily a maintenance release with additional support for the latest RedHawk version and its underlying supported Linux distributions.

4.1. New Features

The main changes in NightStar RT 5.1 are:

- Support of RedHawk 9.2.
- Support of NVIDIA's CUDA 12.1 Release
- Support of new Linux Distributions:
  - RHEL 9.2
  - Rocky Linux 9.2
  - Oracle Linux 9.2
  - Ubuntu 22.04
- Certification on additional ARM64 NVIDIA Development Kits

4.2. Recent Features

NightStar 5.1 includes all of the bug fixes and new features added in NightStar 5.0 as well.

4.3. Changes in NightView

Some of the recent changes in NightView include the following:

- Supports DWARF 5, which is the default debug information generated by **`-g`** on RHEL/Rocky Linux/Oracle Linux 9.2, and on Ubuntu 22.04.
- Supports CUDA 12.1.
- Supports debuginfo packages based on **`.note.gnu.build-id`** sections.
- Speed improvements for type resolution (coalescing of duplicate DWARF type descriptions, some of which are usually incomplete).
- Speed improvements for DWARF reading.
- Supports heapdebug in glibc versions without malloc hooks support.

4.4. Changes in NightTrace

Some of the recent changes in NightTrace include the following:

- NightTrace now provides a Logging API in Python3 (See"Python NightTrace Bindings" on page 12)

4.5. Changes in NightTune

Some of the recent changes in NightTune include the following:

- Supports systems with more than 256 CPUs.
- Supports systems with hybrid CPUs (e.g. Xeon Scalable 4th Generation, Alder Lake, Raptor Lake, etc.): identification of P-cores vs. E-cores; and handling of systems where P-cores are hyper-threaded, while E-cores are not.
- Supports CUDA 12.1.
- Supports CUDA Ada Lovelace (8.9) and Hopper (9.0) architectures.
- Experimental support for the RedHawk 9.2 **`downing=offline`** boot parameter.

4.5.1. Python NightTrace Bindings

In addition to Ada, C, Fortran, and Java bindings, NightTrace now offers Python3 as well as the previous Python2 support interface for logging trace events.

A simple use-case follows:

```
import sys
sys.path.append("/usr/lib/NightTrace/python/python3")
import ntrace
tr=ntrace.log("/tmp/user-data")
tr.Event(47)
tr.Event("pi", 3.14159)
tr.Event("init", "some interesting string")
tr.Event("my_list", [1,2,3,4])
tr.Event("my_tuple",(4.5,3.2))
```

Refer to **`/usr/lib/NightTrace/python/python3/ntrace.py`** for more information on the API and **`/usr/lib/NightTrace/python/python3/examples`** for further use cases.

The Python2 API is still available for systems that have Python2 installed. The paths are the same as those noted above, except change **`python3`** to **`python2`**.

The ccur-ntrace-python package contains the python2 API whereas the ccur-ntrace-python3 package contains the python3 API. Both packages can be installed at the same time, if both **`python2`** and **`python3`** are installed.

5.0. NightStar Remote Debugging

Generally, NightStar requires that **openssh-server** is installed on your target system if it differs from the host system where you run the tools.

When remote debugging, NightView uses **ssh** to connect to the target system. Multiple 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 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.

Similarly, NightProbe, NightSim, NightTrace and NightTune use ssh as the mechanism for communication between the host and target systems.

5.1. 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** packages on the host system and **ccur-nview-i386-target** on x86_64 systems that wish to debug 32-bit applications.

NightView 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, and **ccur-nview-i386-target.i386** on the target system.

When cross-debugging you need a compiler that can generate the target architecture's code. For i386, the situation is easy, since the x86_64 compilers support just that using the -m32 option.

When cross-debugging from x86_64 to arm64, you need a complete compilation tool chain. While not difficult to locate, additional software packages will need to be installed on your host system.

6.0. Obtaining License Keys

The NightStar RT Version 5.1 software uses the same license manager and license features as the previous versions of NightStar RT. If you are already 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:

  https://concurrent-rt.com/customer-support

  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 34 for more detailed information about the NightStar License Manager (NSLM).

6.1. Removing NightStar RT

To remove NightStar RT, you need to run the **`remove-nstar`** script as the root user.

The script is available on the Installation DVD and via Concurrent Real-Time’s network at https://redhawk.concurrent-rt.com/NightStar/remove-nstar.

The md5sum check sum of the **`remove-nstar`** script is 8b39af5a576641a3d784d6ac0d3f3d8f.

7.0. Documentation

The following table lists the NightStar RT 5.1 documentation available from Concurrent Real-Time.

NightStar RT Version 5.1 Documentation

Manual Name	Pub. Number
Installation Guide (Version 5.1)	0898008-5.1
Tutorial (Version 5.1)	0898009-190
NightProbe User's Guide (Version 4.4)	0898465-120
NightSim User's Guide (Version 4.5)	0898480-080
NightTrace User's Guide (Version 7.6)	0898398-200
NightTune User's Guide (Version 3.8)	0898515-050
NightView User's Guide (Version 7.7)	0898395-400
Data Monitoring Reference Manual	0898493-030
Quick Reference for shmdefine	0898010-060

PDF versions of this *Installation Guide* and all other documents listed above can be found at the following locations:

- the **`documentation`** directory of the *NightStar RT Installation DVD*
- *online* at https://redhawk.concurrent-rt.com/docs
- the directory **`/usr/share/doc/NightStar/pdf`** after installation

After installation, HTML versions of all these documents can be accessed via:

- the Help menu from any the NightStar tool
- the command **`/usr/bin/nhelp`**
- in the directory **`/usr/share/doc/NightStar/html`**

8.0. NightStar RT GUI Features

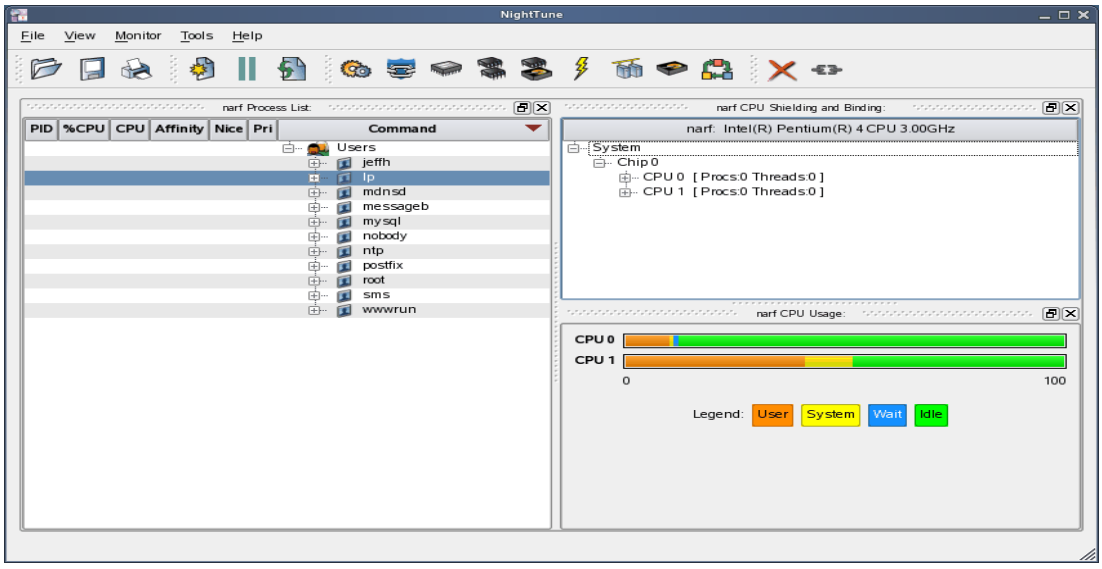
Some of the common features of the NightStar RT Tools graphical user interface include:

- movable and resizable panels
- tabbed pages
- context menus

8.1. Movable and Resizable Panels

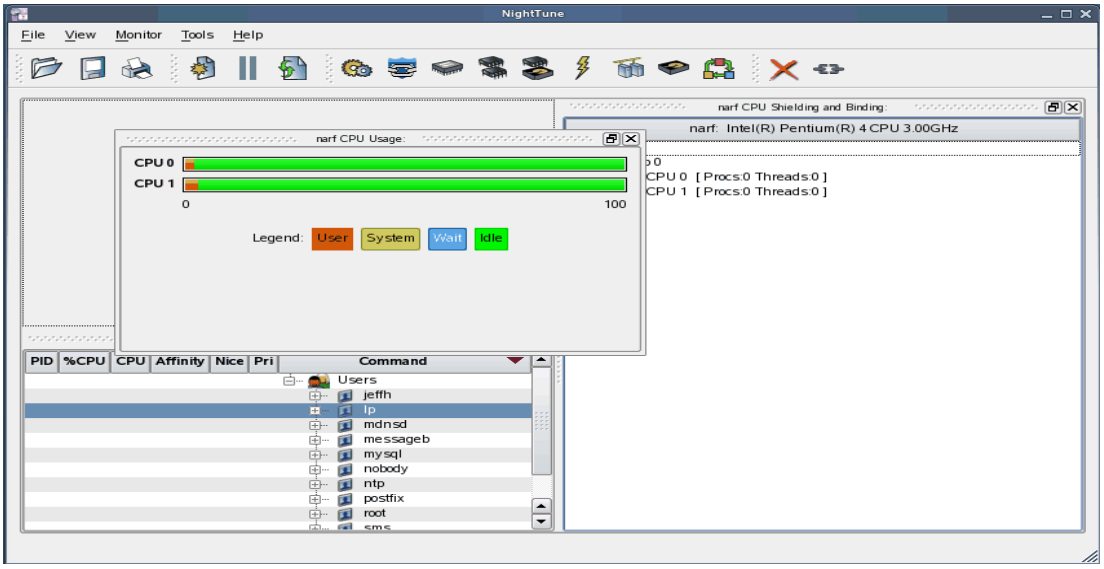
The NightStar RT Tools allow the user flexibility in configuring the graphical user interface to suit their needs through the use of resizable and movable panels.

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

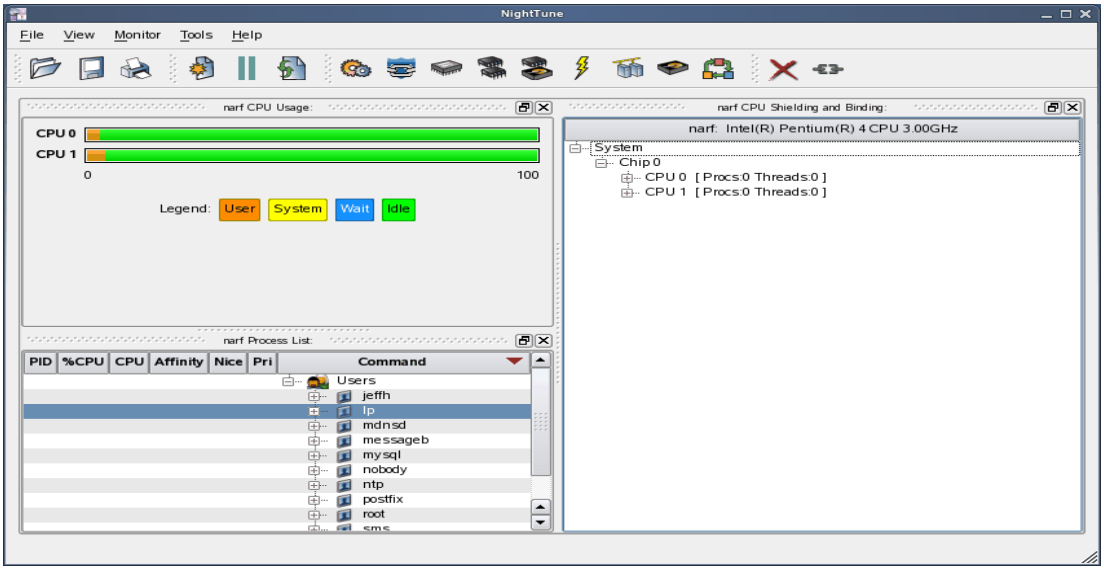


To move one of the panels in the current page, left-click on the title bar for the panel you wish to move and begin to drag the panel to the desired location. The application will respond by creating space on the page based on where you move the panel while resizing and moving the other panels accordingly.

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:

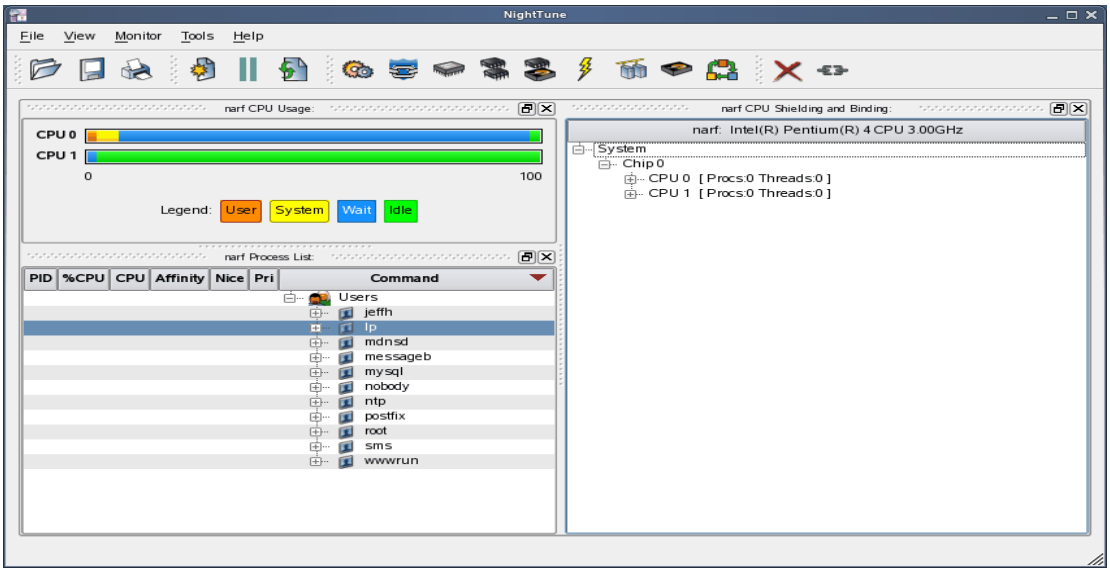


Release the mouse button when NightTune has opened a space where you desire and NightTune will place the panel in that location. The CPU Usage panel now resides in the upper left corner of the NightTune display.



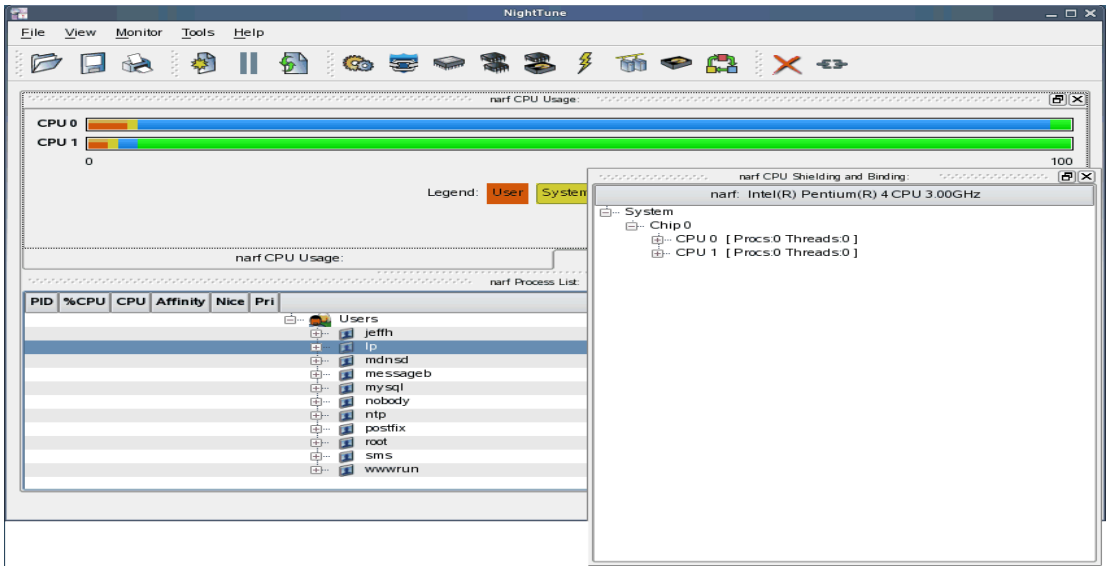
If an empty space does not appear where you desire it, try increasing the size of the main window, decreasing the size of the undocked panel, and moving an alternative edge of the undocked panel near where you want to place it.

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.

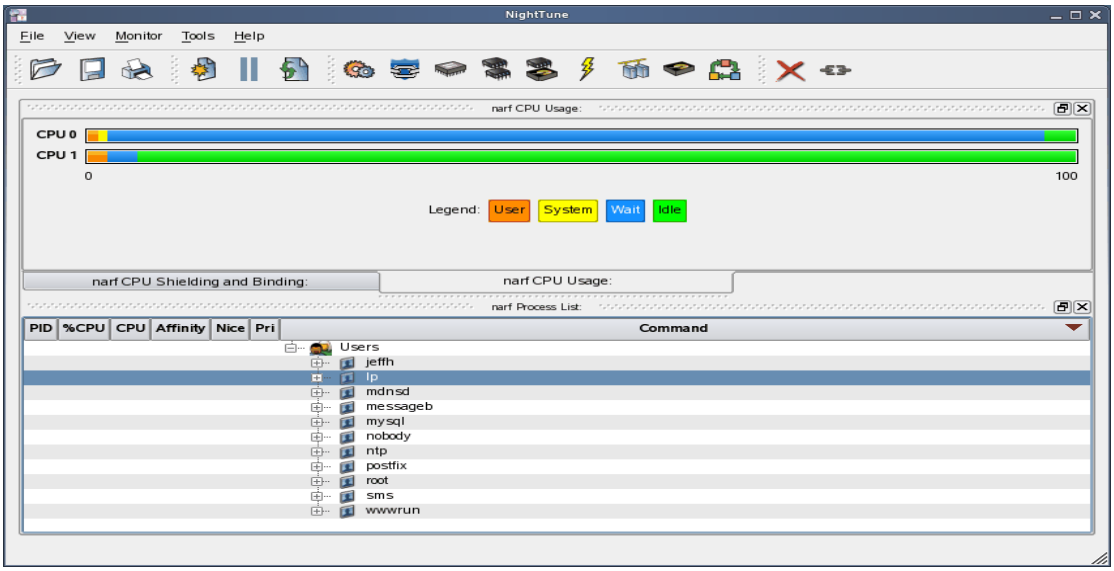


Another feature of the NightStar RT Tools graphical user interface is the use of tabbed panels. Tabbed panels allow you to maximize your GUI real estate by placing two or more panels in the same location. You can then switch between the panels using the tabs created.

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.

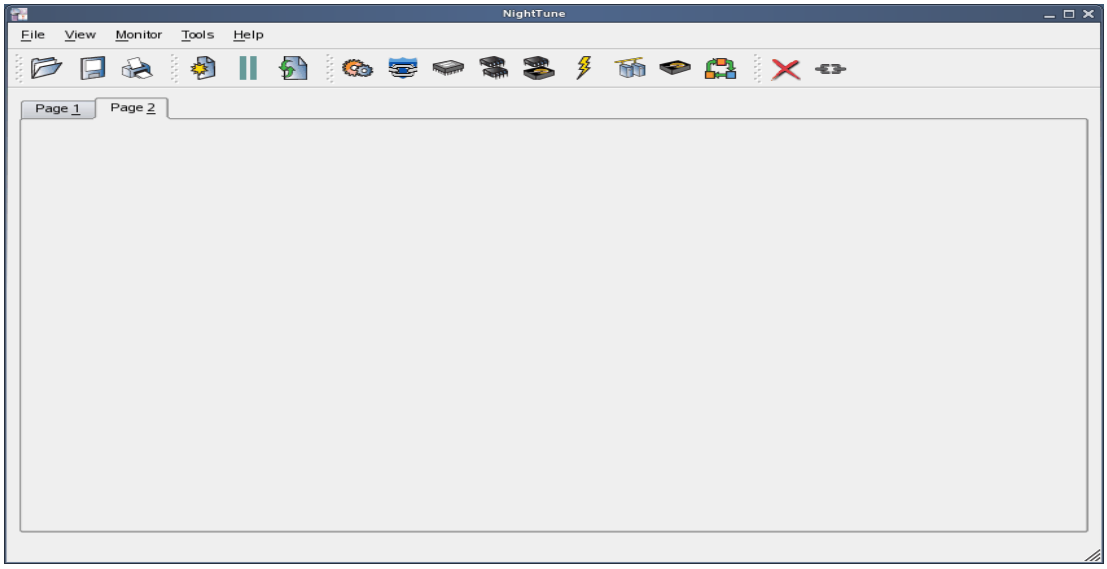


8.2. Tabbed Pages

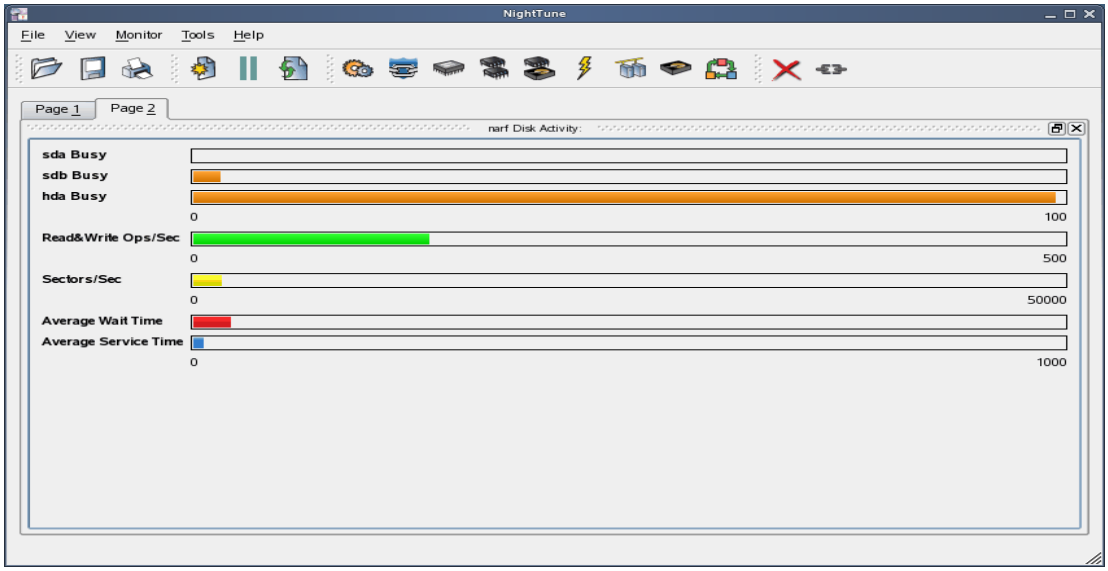
The NightStar RT Tools allow the user to maintain multiple views of data and the mechanisms that manipulate that data within each application through the use of tabbed pages. By default, only one page is displayed when the tool is invoked.

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.

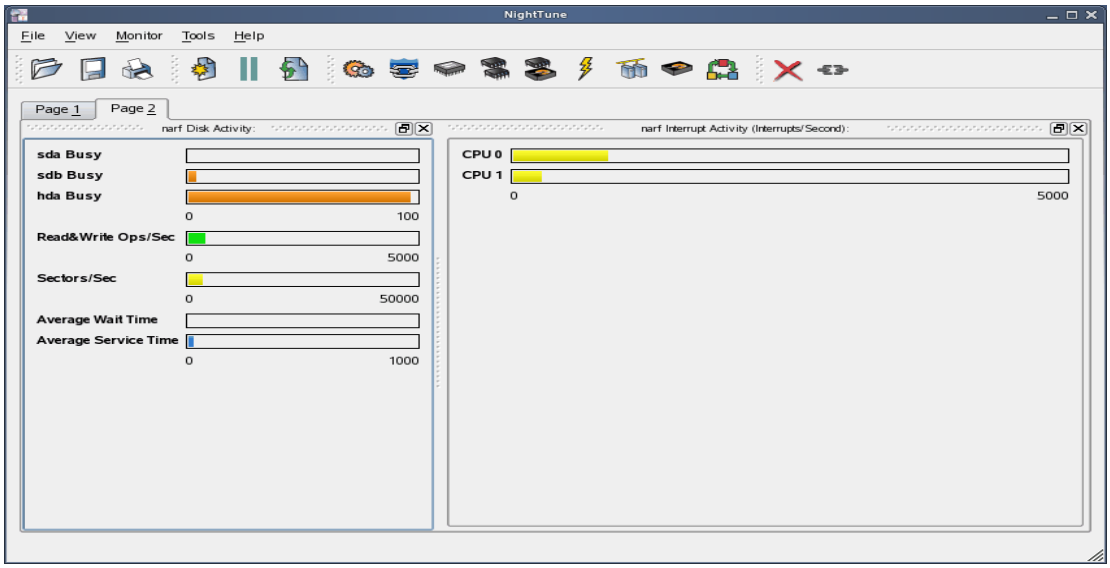
Select Add Page from the View menu. NightTune will create two tabbed pages; our original page is placed under the first tab and a new empty page will be presented under the second.



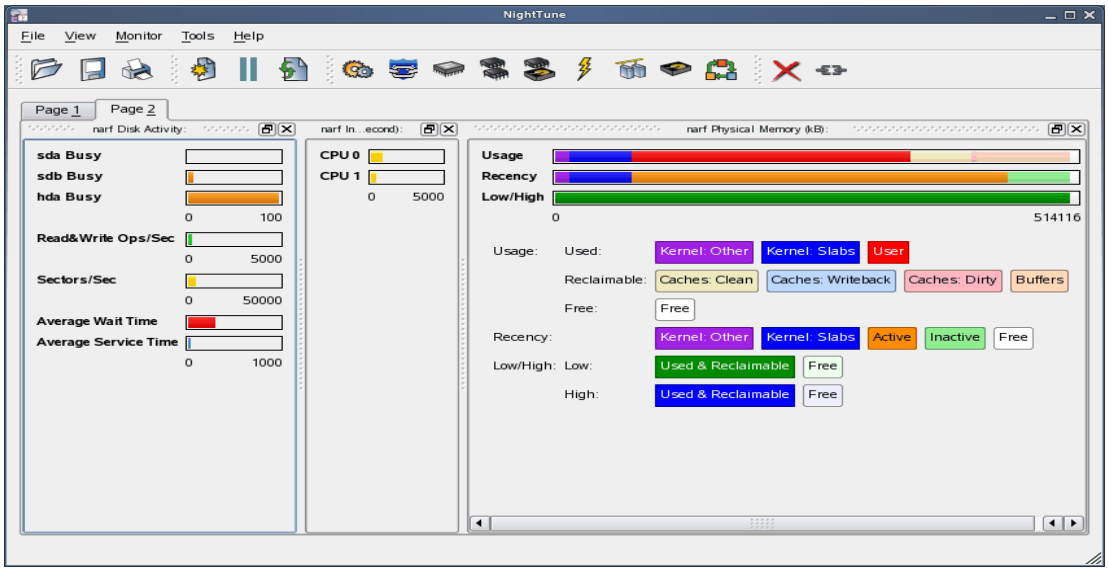
To add the desired NightTune panels, click on the Monitor menu item. You will be presented with a menu of panels to choose from. Select the Disk Activity menu item and then select Bar graph pane from the sub-menu. The Disk Activity panel displaying the information in bar graph format is added to our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



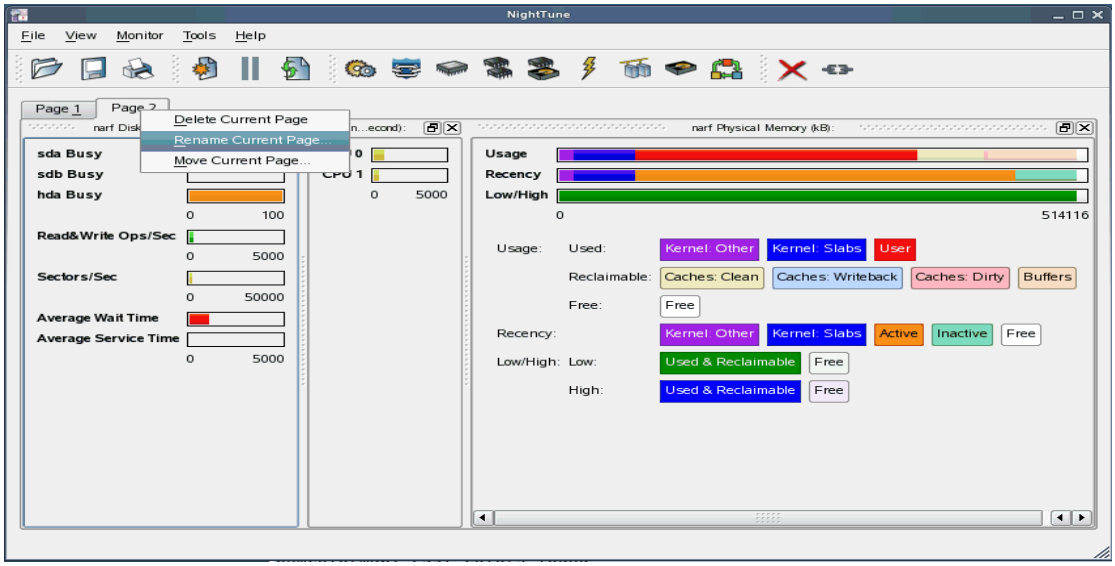
Our new page now contains the Disk Activity, Interrupt Activity, and Memory Physical panels all displaying their information in bar graph format. We can switch back to our first page by clicking on the tab labeled “Page 1” and return to our new page by clicking on the tab labeled “Page 2”.

8.3. Context Menus

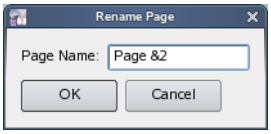
The NightStar RT Tools provide extensive use of context menus. Right-clicking in any of the NightStar RT Tools will provide the user with a menu containing items related to the location of the mouse in the tool.

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 19 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.

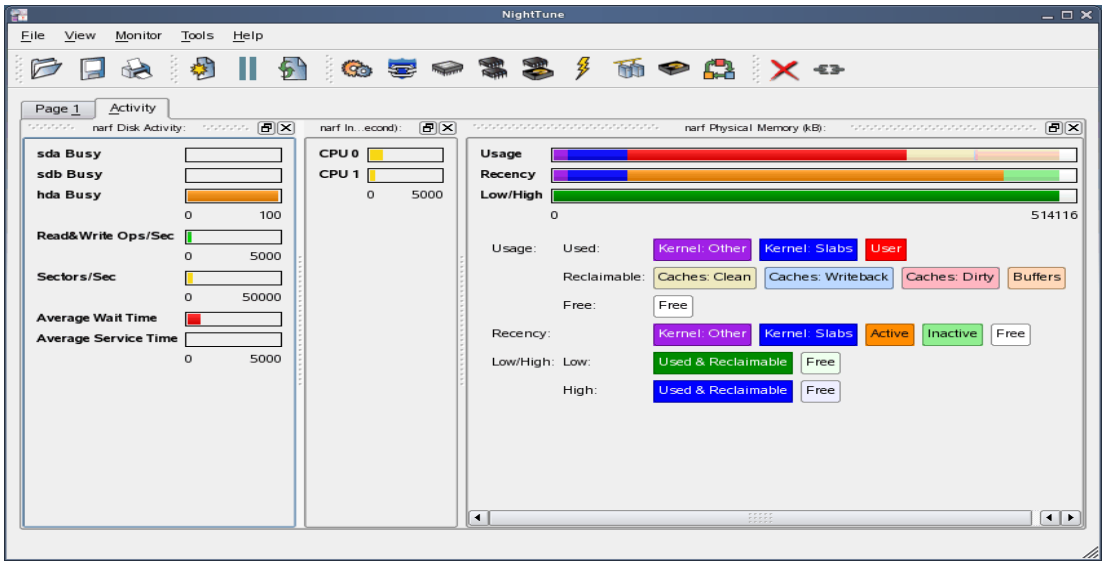


Change the Page Name to "&Activity".

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.



9.0. 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

9.1. 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.

9.2. 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.

9.3. 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.

9.4. 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.

9.5. 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.

9.6. 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.

9.7. 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.

10.0. 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 DVD* 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.

10.1. 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.

Systems with RedHawk installed should be already configured with a *`nightstar`* user role which provides the `cap_sys_nice, cap_sys_rawio` and `cap_ipc_lock` capabilities.

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

```
role nightstar 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     nightstar
```

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

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

WARNING

You will be asked to edit files in **/etc/pam.d** in the remainder of this session. Do not make these changes to **/etc/pam.d** files unless your are certain that the corresponding shared libraries actually reside on the system, otherwise you may not be able to log in again.

It is a good idea to keep a native terminal session open running as `root` when you do these operations, so that you can recover easily if you make mistakes in editing these files. Be careful, for example if you **ssh** into a system as `root`, edit the files, and restart the **sshd** daemon you could lose your current root session and may not be able to log in again.

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
common-session	almost all	In newer systems, e.g. Ubuntu 16.04, this file is included by most services. If that is the case, just add the entry to this file (ignoring the rest of the table).
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	
lightdm	Mate sessions	
kde	kde sessions	
systemd-user	Some GUI sesssions	

If you modify **/etc/pam.d/sshd** or **/etc/ssh/sshd_config**, you must restart the **sshd** service for the changes to take effect, using one of the following commands, depending on your underlying OS version:

```
/bin/systemctl restart sshd
```

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

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

```
getpcaps $$
```

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

If that command is unavailable, execute the following command and you should see non-zero numbers in the lines that begin with CapPrm and CapEff. Those hexadecimal numbers shown are bit masks for individual capabilities.

```
cat /proc/self/status | egrep -e "^Cap"
```

```
CapInh: 0000000000000000
CapPrm: 0000000000824000
CapEff: 0000000000824000
```

10.2. Enabling Full NightStar Support

Some versions of Linux (e.g. Ubuntu 20.04), have a default setting which disables some features of NightStar.

If the system control variable **kernel.yama.ptrace_scope** has a value of **1**, then the following features will be disabled:

- NightView will not be able to attach to processes, even those owned by the same user running NightView
- NightProbe cannot probe programs - variable values will be zero or shown as "cannot access object"
- Datamon (see **dm_get_value(3)**) cannot read or write variables
- NightTune will not be able to trace system calls (see **strace(1)**)

You can remove these restrictions by using the **sysctl** command to change the value of the variable which controls the restriction. To remove the restriction, enter the following command from a shell, as the root user:

```
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.

In order to have the kernel daemon command line tool **ntracekd** write to **/tmp**, set the following **sysctl** variable to zero:

```
sysctl -w fs.protected_regular=0
```

You may also want to add that variable into **/etc/rc.local** so it is set to zero on subsequent boots.

NOTE

If changing **sysctl** is a security concern for your system, then don't use files in **/tmp** with **ntracekd**.

11.0. NightStar RT Licensing

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

License installation requires a license key provided by Concurrent Real-Time. The NightStar RT tools request a license (see “License Requests” on page 36) from a license server (see “License Server” on page 36).

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. 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 36).

The default operating system configuration may include a strict firewall which may interfere with floating licenses. See “Firewall Configuration for Floating Licenses” on page 37 for information on handling such configurations.

11.1. 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 Real-Time. 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
```

IMPORTANT

System identification codes are dependent on system configurations. (See “Selecting a Network Device for Licensing” on page 35 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:

http://concurrent-rt.com/customer-support

Figure 0-1. License Activation Link

Click on "All Revisions" under **`NightStar License Key Request`**, as shown in the figure above.

Provide the requested information, including the system identification code (but first read "Selecting a Network Device for Licensing" on page 35). 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 Real-Time support (see "Direct Software Support" on page 47 for more information).

11.1.1. 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
```

11.2. 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:server2:10.134.30.255
```

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**.

11.3. 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:

- **/usr/bin/systemctl status nslm**

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

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

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

11.4. 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.

11.5. 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 37 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 38 for more information.

11.5.1. Serving Licenses with a Firewall

The following are a few approaches for allowing the NSLM license server to serve floating licenses 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.

These instructions are for a version of **iptables** for RHEL5. A newer version may have a different configuration method, or may have been replaced with a different firewall schema. Regardless, the information displayed below may help you understand the concepts involve.

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
```

11.5.2. 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 licenses 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
```

11.6. License Support

For additional aid with licensing issues, contact the Concurrent Real-Time Software Support Center. See "Direct Software Support" on page 47 for details.

12.0. 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.

12.1. i386 and x86_64 Architecture Issues

NightProbe

No limitations

NightSim

No limitations

NightTune

No limitations

NightTrace

Inter-architecture Capabilities in NightStar RT 5.1

- 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. However, the 32-bit applications that generated the user trace data must be linked with the version of the NightTrace Logging API from the base NightStar RT Version 5.1 (or newer).
- 64-bit applications using the NightTrace Analysis API can analyze data, either in stream or file mode, generated from 32-bit applications.

User Responsibility

- When analyzing 32-bit data on a 64-bit system, be aware that NightTrace will evaluate expression types 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 generated on a 32-bit system 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

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 programs that exec (see **`exec(2)`**) such programs; this includes 32-bit programs launched from a 64-bit shell.

12.2. Intel and ARM64 Interoperability

NightTrace

Binary NightTrace data files may be analyzed from either x86_64 or aarch64 systems. The limitations described in the *Inter-architecture Capabilities in NightStar RT 5.1* NightTrace section apply as well.

NightSim

No limitations.

NightProbe

No limitations.

NightTune

No limitations.

NightView

Since X86 and ARM64 are incompatible architectures you cannot run an x86 executable program on an aarch64 system, nor vice versa.

However, NightView can "cross-debug" from x86_64 to an aarch64 target system. This requires that you have the **`ccur-nview-aarch64-support`** package installed on the x86_64 host and the **`ccur-nview-target`** package installed on the aarch64 target.

For full cross debugging support (e.g. patching, conditional eventpoints, etc.), the GNU x86/aarch64 cross development packages should be installed on the host system as well. The minimal set of packages include:

- gcc-aarch64-linux-gnu
- binutils-aarch64-linux-gnu

These packages are generally available for CentOS-like and Ubuntu-like systems.

Currently, NightView only looks for the cross compiler at the following location:

`/usr/bin/aarch64-linux-gnu-gcc`

You may need to draw a symbolic link to the actual compiler if your distribution has it located elsewhere. NightView will be subsequently be modified to allow you to set the cross-compiler path from inside NightView.

If you build your full user application using the GNU cross development environment you will likely require additional packages. General cross development is outside the scope of this document.

13.0. Known Problems

The following sub-sections list known issues with NightStar.

13.1. Problem: Position Independent Executables

NightStar does not yet fully support Position Independent Executables (PIE). In modern Linux systems PIE is now the default. The following capabilities are currently unavailable for PIE executables:

- NightProbe
- Datamon
- Application Illumination (nlight)

These limitations may be lifted in the next major release. In the interim, you can still build non-PIE executables by using a combination of the following gcc and ld options:

```
gcc -fno-PIC -fno-PIE -no-pie ...
```

13.2. Problem: NightView Load Command

The **load** command is known to reject object files built on some versions of RedHawk 8.x and 9.x.

The gas assembler began using the **R_X86_64_PLT32** relocation code, previously reserved for PIC and/or PIE code.

This problem is known to affect object files built on distributions with **binutils** 2.31 or later.

13.3. 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 1:

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.

Solution 2:

Change the enforcement mode of SELinux to `permissive`. Edit the **/etc/selinux/config** file to make this change and reboot. The NightStar development team is working on another solution to this problem.

13.4. Problem: NightView Cannot Map Memory Segments

On some systems, SELinux is installed and set to `enforcing` mode. This can prevent NightView from mapping to application's memory segments, depending on your distribution type and version.

Solution:

Change the enforcement mode of SELinux to `permissive`. Edit the **`/etc/selinux/config`** file to make this change and reboot. The NightStar development team is considering alternative solutions to this problem.

13.5. Problem: NightView Cannot Attach To Processes

See "Enabling Full NightStar Support" on page 33.

13.6. Problem: Datamon & NightProbe on RedHawk

Datamon and NightProbe only work on RedHawk 9.2 when using the **`-gdwarf-3`** compiler options, as opposed to just **`-g`**.

13.7. Problem: NightProbe Cannot See Variable Values

See "Enabling Full NightStar Support" on page 33.

13.8. Problem: Datamon Cannot Fetch Variable Values

See "Enabling Full NightStar Support" on page 33.

13.9. Problem: NightTune Cannot Trace System Calls

See "Enabling Full NightStar Support" on page 33.

13.10. Problem: NightLight Does Not Work

The **`nlight`** tool only operates on programs built with **`-no-pie`** on the latest Linux distributions. This limitation will be lifted in a future release.

NightLight will only work on RedHawk 9.2 when using the **`-gdwarf-3`** compiler options, as opposed to just **`-g`**.

13.11. Problem: ntracekd Not Writing to /tmp

In order to have the kernel daemon command line tool **`ntracekd`** write to **`/tmp`**, set the following **`sysctl`** variable to zero:

```
sysctl -w fs.protected_regular=0
```

You may also want to add that variable into **/etc/rc.local** so it is set to zero on subsequent boots.

NOTE

If changing **sysctl** is a security concern for your system, then don't use files in **/tmp** with **ntracekd** .

14.0. Direct Software Support

Software support is available from a central source. If you need assistance or information about your system, please contact the Concurrent Real-Time 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 Real-Time Computer Corporation web site at **http://concurrent-rt.com/customer-support** or by sending an email to **support@concurrent-rt.com**.