NetworkComputer™ Training

Version: 2016.09

Runtime Design Automation
www.rtda.com
info@rtda.com

Table of Contents

Table of Contents

1 User Tutorials

1.1 Introduction

In the following User tutorials, we will experiment with the NetworkComputer on most topics that a user would be most interested in, including submitting jobs, tracking job information, analyzing and solving common problems, etc.

1.2 Getting Help About NetworkComputer

The documentation is available in HTML, PDF and TXT format.

1.2.1 Accessing documentation when NetworkComputer is running

When NetworkComputer is running, it serves documentation through its browser interface. To access it from browser, you need to know which host and port the NetworkComputer is running on. Ask your administrator, or find the URL for NetworkComputer with the following command:

```
% nc cmd vovbrowser
http://comet:6271/project
```

Here we assume NetworkComputer is running on host comet, port 6271. Then the Url for NetworkComputer is:

`http://comet:6271`

So, url for the Flowtracer™, NetworkComputer™, LicenseMonitor™ and the VOV subsystem documentation is:

`http://comet:6271/cgi/bookshelf.cgi`

1.2.2 Accessing documentation when NetworkComputer is not running

All the documentation files are under NetworkComputer install directory, so you can access them even if NetworkComputer is not running. To do this, point your browser to
`file:///whereever/flowtracer/is/installed/common/doc/html/bookshelf.html`

(We recommend that your bookmark the above URL for future reference.)

1.2.3 Getting documentation in PDF files

Runtime also provides PDF files for each of the documentation book, if it is your preference, and also for easy printing. All the PDF files are in the directory:

`/whereever/flowtracer/is/installed/common/doc/html/pdf`

1.2.4 Getting help from command line

The main commands of NetworkComputer are `nc` and `ncmgr`, with some subcommands and options. For almost all commands, you can get help of its usage, description and examples by just running the command without any option or with `-h` option. For example,

```
% nc info -h
nc:
nc: NC INFO:
nc:     Get information about a specific job or list of jobs.
nc: USAGE:
nc:         % nc info <jobId> [options] ...
nc: OPTIONS:
nc:     -h               -- Show this message
nc:     -l               -- Show the log file (actually, it shows all outputs)
nc:
```

1.2.5 Getting help from us

The following address was current at the time of publication; please check our website http://www.runtimeinc.com for the latest.

```
Runtime Inc.
2560 Mission College Blvd. Ste 130
Santa Clara, CA 95054
U. S. A.

Tel: +1-408-475-7832 ( +1-408-475-RTDA )
Tel: +1-408-492-0942

Email:   support@runtimeinc.com

Website: http://www.runtimeinc.com
```

1.3 User Account Setup

To set up your user account with NetworkComputer you need to know where the Runtime software has been installed. Ask your system administrator.

1.3.1 Setup on Unix

Runtime supplies a script `vovsetupuser` to help get your shell configured to use Runtime commands. The setup script creates a `.vovrc` or `.vovrc.sh` file in your home directory, and modifies your shell's startup script to source the new file.

If you have more than one version of Runtime software, the one referenced by the `.vovrc` file is determined by the version from which you run the setup script.

Execute the script `vovsetupuser`, following the instructions:

```
% cd /whereever/flowtracer/is/installed
% cd common/scripts
% ./vovsetupuser
FlowTracer(tm)

Installing FlowTracer's shell setup file.

Install dir   : /home/dexin
Using VOVDIR  : /remote/release/VOV/2013.09/linux64

Would you like to continue? [y/N] y
Checking if .cshrc is sourcing .vovrc... YES.
vovsetupuser: Testing the PATH in a non-interactive shell, using rsh... OK
Installation is done.

To use FlowTracer, please source the .vovrc file now with
% source /home/dexin/.vovrc
or start a new shell.
```

You need to answer "yes - y" to continue and do REMEMBER to source the designated file or start a new shell as instructed.

1.3.2 Setup on Windows

If NetworkComputer is installed in directory `R:\rtda\2013.09`, you can set up your cmd shell by executing:

```
c:> R:\rtda\2013.09\win64\bat\vovinit.bat
```

1.3.3 Verify that your setup works

Try to run a Runtime command to verify that your setup works:

```
% vovarch
linux64
```

If you get an error (e.g. command not found), please refer to the documentation for troubleshooting tips at `/doc/FlowTracer_UserGuide/troublesetup.html`.

1.3.4 Remote shell (ssh) setup

It is helpful to have a working remote shell capability to start vovslaves on non-local hosts. Runtime supplies a script that can help, provided your home directory is auto-mounted.

```
% vovsshsetup
vovsshsetup: Generating 1024 bit keys...
  (other output omitted)
vovsshsetup: now try:   ssh <remote-host> date
```

Then, if you substitute 'vovversion' for 'date' in the above command, it tests both that your remote shell setup works, and that the expected Runtime version is in the path.

1.4 Run Basic Jobs

Submitting jobs to NetworkComputer is quite simple: just use `nc run` followed by the command you would use without NetworkComputer.

1.4.1 Run a sleep job

```
% cd                # Go to your home directory
% nc run sleep 10
Resources= linux
Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Command  = vw vwrap sleep 10
Logfile  = vnc_logs/20021230/103409.13784
JobId    = 04194422
nc: message: Scheduled jobs: 1       Total estimated time: 0s
```

1.4.2 Run sleep job with `-r` option to override default resources

By default, `nc run` takes the architecture of the machine from which the job is submitted as the job's resource. This can be overriden with `-r` option when needed.

This job requires no resource

```
% nc run -r "" -- sleep 10
Resources=
Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Command  = vw vwrap sleep 10
Logfile  = vnc_logs/20021230/103653.13794
JobId    = 04194459
nc: message: Scheduled jobs: 1       Total estimated time: 0s
```

Note: since the `-r` option accepts multiple arguments, you need to terminate the resource list explicitly with another option, or "--" if no other options are needed.

This job requires spice_license

```
% nc run -r spice_license -- sleep 10
Resources= spice_license
Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Command  = vw vwrap sleep 10
Logfile  = vnc_logs/20021230/103948.13803
JobId    = 04194497
nc: message: Scheduled jobs: 1       Total estimated time: 0s
```

1.4.3 Run job with `-e` option to override default environment

By default, `nc run` takes a snapshot of the environment as the job's environment and uses this to run your job. You can override this with `-e` option to use a named environment.

```
% nc run -e BASE sleep 10
Resources= linux
Env      = BASE
Command  = vw vwrap sleep 10
Logfile  = vnc_logs/20021230/122737.14946
JobId    = 04195924
nc: message: Scheduled jobs: 1       Total estimated time: 0s
```

Use the command `vel` to list all the available environments:

```
% vel
vel: message: Environment directories:
1 /remote/release/VOV/7.0u3/linux64/local/environments
1 . tcl BASE            UNIX utilities, X windows, and VOV.
1 . tcl D               Define variables: Usage: ves "+D(VAR1=value1,...)"
1 * tcl DEFAULT         Just a name for whatever you already have.
1 . tcl HSIM            Fake Nassda hsim env for Virage testing
1 . csh SYNOPSYS        Synopsys tools
```

Scripts that implement named environments may be written in csh, sh, or Tcl syntax. Cross-platform environments between Windows and Unix/Linux must be written in Tcl. You can use named environments from the command line using the command `ves`.

1.5 Get Summary Info About Jobs

`nc summary`, `nc list` and `nc info` *`jobId`* will be the three commands that you use frequently to get information about the jobs managed by NetworkComputer.

1.5.1 Get summary of all my jobs: `nc summary`

```
% nc summary
NC Summary For User dexin
TOTAL JOBS         11      Duration: 23s
  Done              1
  Idle              0
  Queued            5
  Running           2
  Failed            3

  JOBS   GROUP    TOOL        WAITING FOR...
     1   alpha    sleep       'aa'
     3   alpha    sleep       'linux'
```

In the above example, the command displays summary information for user dexin. There are a total of 11 jobs, 1 of which is Done, 2 Running, 3 Failed, etc. In the bottom part, it shows the summary for jobs that are queued, that is, there is 1 sleep job queued because it is waiting for resource 'aa', and there are 3 sleep jobs waiting for resource 'linux'.

To get an idea of what is going on in the whole NC system, the following shows all the jobs and what the jobqueue buckets are waiting for. Note that NC subcommands may be abbreviated.

```
% nc sum -a -b
```

1.5.2 Get a list of my jobs: `nc list`

This command, without any option, displays the last 20 of your jobs in the format of "jobId status command".

```
% nc list
04193437 Done    sleep 1
04193911 Done    sleep 5
04193913 Failed  sleep 10
04193917 Failed  sleep 20
04193919 Idle    sleep 60
04193937 Running sleep 10
04193943 Queued  sleep 56
```

1.5.3 Status Meaning

In NetworkComputer, each job is assigned a "Network Computing Status" which is defined as follows:

Status	Color *	Explanation
Queued	**Cyan**	The job is scheduled to be executed.
Running	**Orange**	The job is currently executing
Done	**Green**	The job ran successfully
Failed	**Red**	The job ran and failed.
Idle	**BlueViolet**	The job needs to be run, but it is not scheduled.

* (Colors may look different on some systems)

1.5.4 Some options about `nc list`

`nc list` command has several useful options. Here we introduce some of them.

- Get detailed usage of this command:

```
% nc list -h
```

- List all jobs, including others' jobs:

```
% nc list -a
04193437 Done    alpha    dexin    rhino      sleep 1
04193898 Done    users    integ    rhino      sleep 2
04193939 Done    alpha    dexin    rhino      sleep 15
04193941 Done    alpha    dexin    rhino      sleep 20
04193943 Done    alpha    dexin    rhino      sleep 60
```

- Control output format, show job Id the (first) tool of job only:

```
% nc list -O "@ID@ @TOOL@"
04193437 sleep
04193898 sleep
04193939 sleep
04193941 sleep
04193943 sleep
```

In the above example, we use "fields", i.e., ID and TOOL, surrounded by two "@" signs, to format strings. Please refer to the documentation for allfields (at `/doc/FlowTracer_UserGuide/fields.html`).

1.5.5 Get detailed information about a job: `nc info jobId`

Without options, this command displays the basic information about the job, like user, group, directory, command, environment, queue time, etc.

```
% nc info 04193937
Id,User,Group   04193937,dexin,alpha
Environment     D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Directory       ${HOMES}/dexin
Command         sleep 10
Status          Done
   Host         rhino
   QueueTime    1s
   Duration     10s
   Age          20m37s
   AutoForget   1
```

With option `-l`, this command shows the contents of the log file.

```
% nc info -l 04193937
Log file is: '${HOMES}/dexin/vnc_logs/20021230/095337.13512.3'
vwrap: message: Start date: Mon Dec 30 09:53:38 PST 2002
vwrap: message: On host: rhino
vwrap: message: Sourcing environment vnc_logs/envdexin36362.env
vwrap: message: Running: 'sleep 10'
vwrap: message: Exit status: 0
vwrap: message: End   date : Mon Dec 30 09:53:48 PST 2002
```

1.6 Run Jobs With Various Options

In this section, we will exercise some options in `nc run` command. To get the detailed usage, use the following command:

```
% nc run -h
```

1.6.1 Submit multiple jobs at once: -f <file>

1. Prepare a file with one command on each line. Empty lines are ignored and lines that begin with # are considered comments.

```
# Example of file used to submit multiple jobs at once.
sleep 10
sleep 11
sleep 12
sleep 13
```

2. Use the option `-f` to specify the command file, as in the following example:

```
% nc run -f commandFile
```

All jobs submitted with this method share the same environment, the same resources, and are scheduled at the same priority level. Each job has its own id.

1.6.2 Use environments: -e <env>

1. Get a list of all available environments:

```
% vel
vel: message: Environment directories:
1 /release/VOV/latest/linux64/local/environments
1 . tcl BASE            UNIX utilities, X windows, and Flowtracer.
1 . tcl D               Define vars: Usage: ves "+D(VAR1=value1,...)"
1 . tcl DEFAULT         Just a name for whatever you already have.
1 . csh SPICE           The Analog simulator SPICE3.
1 . csh JAVA            JAVA Development Environment (1.2.2)
```

2. Use proper environment(s) to submit jobs, for example:

```
% nc run -e BASE sleep 10
% nc run -e BASE+SPICE spice chip.spi
```

1.6.3 Use of Resources: -r <res1 res2 ...>

1. Submit a job that only runs on Linux machine:

```
% nc run -r linux -- sleep 10
```

2. Submit a job that requires resources "hpux" and "hspice":

```
% nc run -r hpux hspice -- sleep 10
```

Note: since this option accepts multiple arguments, you need to terminate the resource list explicitly with another option, or "--" if no other option is needed.

1.6.4 Wait for jobs: -w

Waiting for jobs is especially useful for scripts. By default, `nc run` returns immediately after you submit a job. This allows you to submit multiple jobs at once. There could be times when you want to run jobs in sequence, in which case option `-w` is very useful. With this option, `nc run` waits for the job(s) to finish (Done or Failed).

```
% nc run -w sleep 10
Resources= linux
Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Command  = vw vwrap sleep 10
Logfile  = vnc_logs/20021231/111314.21781
JobId    = 04211346
vnc: message: Scheduled jobs: 1       Total estimated time: 0s
```

1.6.5 Wait for jobs and show log file of the last job: -wl

With this option, `nc run` waits for the job(s) and shows the log file of the last job

```
% nc run -wl date
Resources= linux
Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Command  = vw vwrap date
Logfile  = vnc_logs/20021231/111530.21819
JobId    = 04211392
nc: message: Scheduled jobs: 1       Total estimated time: 0s
<<<STARTING ON rhino>>>
Tue Dec 31 11:15:31 PST 2002
<<<END OF LOG>>>
<<<EXIT STATUS 0>>>
```

Please only use `-wl` for jobs where you are actively monitoring the output. Such jobs listen to the vovservers event stream and are called 'notify clients'. They are more resource-intensive than plain batch jobs.

1.6.6 Wait for jobs by command nc wait

You can also wait for jobs using the command `nc wait`. We will cover that in next tutorial "Job Control".

1.6.7 (Advanced) Specify name of logfile: -l <logfile>

The default logfile name has the form of `./vnc_logs/date/time`. You can explicitly specify the logfile of the jobs you submitted by this option, for example:

```
% nc run -l /home/john/logs/log1.log sleep 10
```

Warning: conflicts may occur if same log file is used for multiple jobs. New users are not recommended to use this option.

1.7 Job Control

NetworkComputer provides several commands of job controls, including wait, stop and forget.

1.7.1 Waiting for jobs

Besides option `-w` and `-wl` with `nc run` command, you can also wait for job(s) to finish(Done or Failed) after they are submitted using the command `nc wait`. Here are some examples:

1. Get usage help
   ```
   % nc wait -h
   ```
2. Wait for a job
   ```
   % nc run sleep 10
   Resources= linux
   Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
   Command  = vw vwrap sleep 10
   Logfile  = vnc_logs/20021231/140024.23307
   JobId    = 04213283
   nc: message: Scheduled jobs: 1       Total estimated time: 0s

   % nc wait 04213283
   ```
3. Wait for all jobs in current directory
   ```
   % nc wait -dir .
   nc: message: Job 04193913 is already FAILED
   nc: message: Job 04193915 is already FAILED
   nc: message: Job 04193917 is already FAILED
   nc: message: Job 04193919 is not scheduled
   nc: message: Job 04194308 is not scheduled
   nc: message: Job 04211259 is already FAILED
   nc: message: Job 04211268 is not scheduled
   nc: message: Job 04213283 is already VALID
   nc: message: Job 04213297 is already VALID
   nc: message: Exiting with status 2 (Failed jobs)
   ```
4. Wait for all jobs using tool spice
   ```
   % nc wait -select "tool==spice"
   ```

   In the above example we use "Selection Rules" to perform a "wait" on those jobs that satisfy that rule. This could be useful for other commands as well, including `nc list`, `nc forget`, etc. Refer to the NetworkComputer User Guide for more information.

1.7.2 Forgetting jobs

The NetworkComputer server remembers the jobs you submitted for some configurable time. You can explicitly forget them by `nc forget` command, which will delete all job information from the server database.

1. Get a list of jobs:

```
% nc list
04146420 Done    sleep 1    #in the form of "jobId status command"
04146425 Done    sleep 5
04146427 Running sleep 10
04146429 Running sleep 15
04146431 Queued  sleep 20
04146433 Queued  sleep 60
```

2. Forget some of them:

```
% nc forget 04146420 04146425
nc: message: Forgetting 2 jobs
```

3. List jobs again (notice those two are gone):

```
% nc list
04146427 Done    sleep 10
04146429 Done    sleep 15
04146431 Done    sleep 20
04146433 Done    sleep 60
```

4. Forget all my jobs:

```
% nc forget -mine
nc: message: Forgetting 4 jobs
```

5. Now list jobs again, you should have nothing left:

```
% nc list
```

1.7.3 Stopping jobs

A job can be stopped when it is either Running or Queued. Stopping a job does not forget it from the server database. "Running" jobs will exit, and "Queued" jobs will be dequeued when you stop them.

1. Stop some jobs:

```
% nc run sleep 60
Resources= linux
Env      = D(VOV_ENV_SOURCE=vnc_logs/envdexin36362.env)
Command  = vw vwrap sleep 60
Logfile  = vnc_logs/20021231/140803.23386
JobId    = 04213393
nc: message: Scheduled jobs: 1       Total estimated time: 0s

nc stop 04213393
nc: message: Stopping RETRACING job 04213393
```

2. Stop all my jobs:

```
% nc stop -mine
nc: message: Stopping RETRACING job 04146627
nc: message: Stopping RETRACING job 04146629
nc: message: De-queuing  job 04146631
nc: message: De-queuing  job 04146633
```

1.7.4 (Advanced) Rerunning Jobs

The `nc rerun` command initiates the scheduling and execution of jobs that are already in the server database. By default, only the jobs that are Idle or Queued are affected by this command. If you want to force the rerunning of jobs that are either Done or Failed, use the option -F.

1. Rerun a "Done" job won't do anything:

```
% nc rerun 04146622
nc: message: Not rerun: 04146622
```

2. Force rerunning a "Done" job:

```
% nc rerun -F 04146622
nc: message: Job 04146622 is already VALID.
nc: message: Scheduled jobs: 1       Total estimated time: 1s
```

3. Rerun "Idle" jobs:

```
% nc rerun 04146631 04146633
nc: message: Scheduled jobs: 1       Total estimated time: 0s
nc: message: Scheduled jobs: 1       Total estimated time: 0s
```

1.8 Getting information about a job

1.8.1 Getting information about a job

The `nc info` command gives information about a job.

Usage: `'nc info -h'`

```
vnc: Usage Message
  NC INFO:
      Get information about a specific job or list of jobs.
  USAGE:
      % nc info [options] <jobId> ...
  OPTIONS:
      -h               -- Show this message.
      -v               -- Increase verbosity.
      -l               -- Show the log file (actually, it shows all outputs).
      -e               -- Show the environment name, or contents if a snapshot.
      -c               -- Show slave compatibility table (which slave can run a job).
      -sc              -- Show slave compatibility in normal output.
      -dep             -- Show job dependencies.
      -J   <jobname>   -- Show the jobs with given name.

  EXAMPLES:
      % nc info 00123456         -- Show info about specific job.
      % nc info !                -- Show info about most recent job in current dir.
      % nc info -l 12345         -- Show log file(s) of job.
      % nc info -J MyJob         -- Show info about all jobs called "MyJob".
      % nc info -sc 0012345      -- If job is Scheduled, also show the summary of slave compatibility.
```

Without options, `info` displays the start time, completion time, duration, resources, priority, and exit status.

Examples:

```
% nc in 02930498
Id,User,Group   02930498,john,alpha
Environment     BASE
Directory       ${HOMES}/john
Command         sleep 10
Status          Done
Host         alpaca
QueueTime    0s
Duration     10s
Age          55m49s
AutoForget   1

% nc info -l 3452
vwrap: Start date: Sun Jun  2 10:57:32 PDT 2002
vwrap: On host: alpaca
vwrap: Running: sleep 10
vwrap: Exit status: 0
vwrap: End   date : Sun Jun  2 10:57:42 PDT 2002
```

1.8.2 Getting detailed information about a job

The `nc getfield` command gives information about a job in a form that is useful in scripts.

Usage: `'nc getfield'`

```
vnc: Usage Message

  NC GETFIELD:
      Get one or all fields of one or more NC jobs.  Specify the jobID
      or use '!' for the most recent job in the current working directory.

      If the -J jobName option is given, only the first match
      is reported.  If there is no match, an error is reported.

  OPTIONS:
      -h           Show this help. You can also get the
                   usage message by specifying no option at all.
      -v           Increase verbosity.
      -J JOBNAME   Find first job with given JOBNAME
                   The search is restricted to the jobs that
                   belong to the current user.
                   This is significantly more expensive than
                   using jobIds.  Use sparingly.
      -f field     Specify field when giving multiple jobIDs
      -showid      Show jobId
      -s           Same as -showid
      -sep STRING  Use STRING as separator (default is a single space)
      -tab         Use a TAB character as separator.

  EXAMPLES:
      % nc getfield -h
      % nc getfield 01234455
      % nc getfield 00123445 jobclass
      % nc getfield ! status
      % nc getfield -J JOBNAME
      % nc getfield 01234455 0123458 -f jobclass
      % nc getfield -s 01234455 0123458 -f jobclass
```

Examples:

```
% nc getfield 00012345 jobclass
normal
% nc getfield 00012345 cputime
7.125
% nc getfield 00012345
... get list of all known fields (more than 100 of them)...
```

1.9 The Graphical User Interface

1.9.1 Monitoring with NetworkComputer Monitor

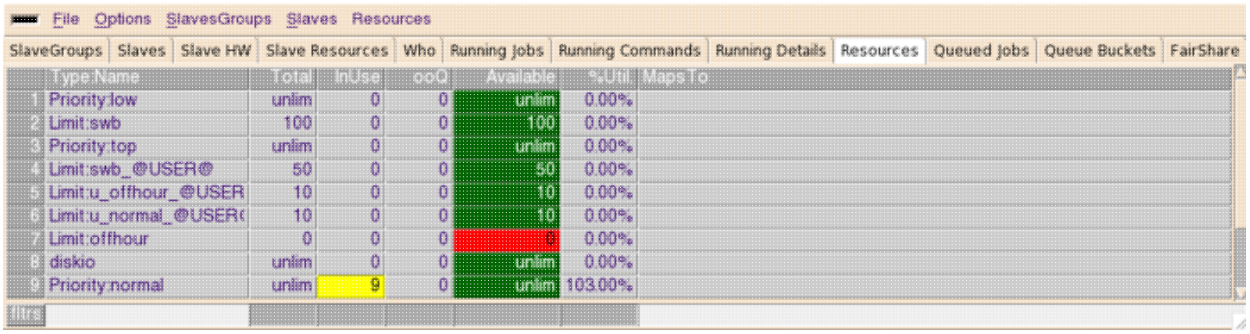
The activity of NetworkComputer can be monitored using the dialog shown below. This dialog is invoked with:

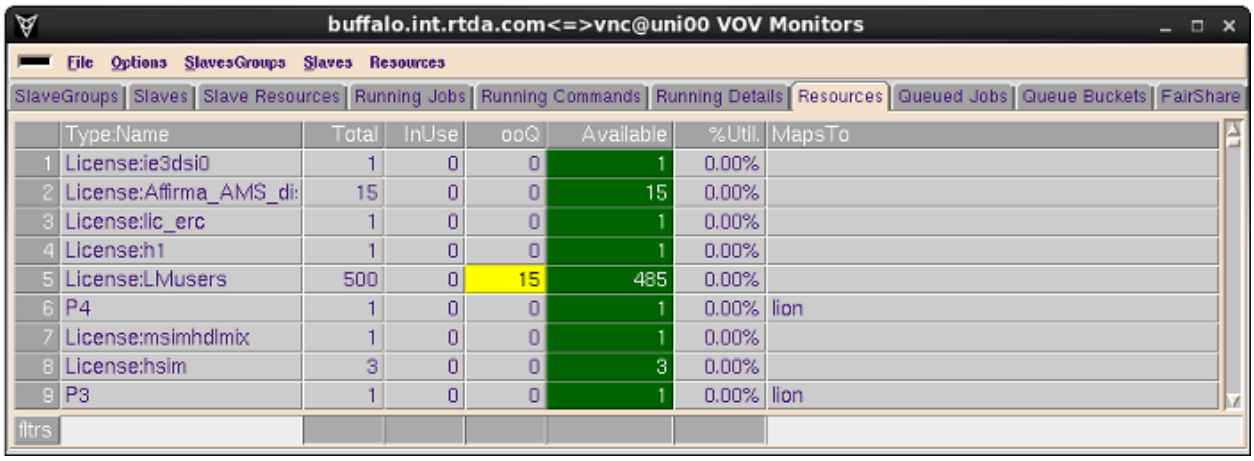
```
% nc monitor
% nc mon        # Abbreviations also work
```

The dialog gives you access to 12 panels to monitor:

1. **SlaveGroups:** The activity of slave groups
2. **Slaves:** The activity of slaves
3. **Slave HW:** The hardware offered by slaves
4. **Slave Resources:** The resources offered by slaves
5. **Who:** Who is running jobs
6. **Running Jobs:** The progress of running jobs
7. **Running Commands:**The details of running commands

8. **Running Details:**The details of running jobs
9. **Resources**The usage and availability of resources
10. **Queued Jobs**The jobs in the job queue
11. **Queue Buckets**The jobs in the job queue organized by groups of similar jobs (called 'buckets');
12. **FairShare:** The FairShare statistics.

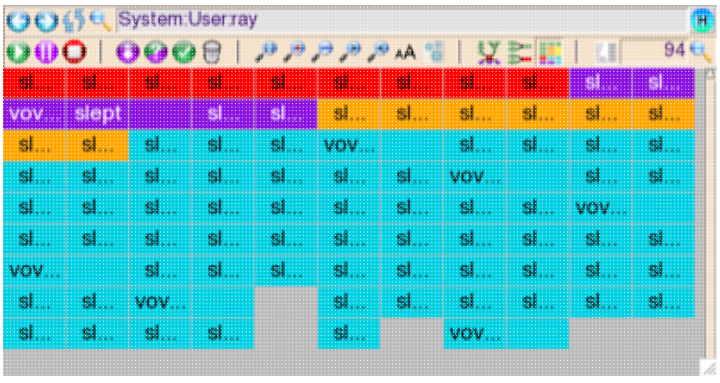




1.9.2 Monitoring Job Execution with NetworkComputer GUI

A convenient way to monitor the execution of the jobs is to use `nc gui`

```
% nc gui -all &
```



The GUI provided by this command is minimal by design. Each job is shown by a color-coded rectangle, and you can view very many at a time and zero in on the red FAILED ones that need your attention. Double-clicking a job brings up its details in the node editor.

1.10 Using The Web Browser

1.10.1 Find the URL for NetworkComputer:

```
% nc cmd vovbrowser
http://comet:6271/project
```

1.10.2 Using the browser UI

Enter the URL found above into your browser's (Firefox or Internet Explorer) location box. You will need to login unless your administrator has turned off authentication.

`http://`*`your_host`*`:`*`your_port_number`*`/project` is the home page of NetworkComputer. From this page, you can easily navigate to many other useful pages. Here we list some monitor pages that are similar to the ones from the GUI monitors.

- Slaves page:
  From home, click on "Slaves", in the top center.
- Workload pages (job queue and running jobs):
  From home, click on "Workload".
  For running jobs - click on "Running jobs" under "Workload" section.
- Resources page:
  From home, click on "Resources".
- Fairshare page:
  From home, click on "Fair share" under "Workload" section.

Among the many useful pages, two report pages may be especially helpful.

- Job queue report:
  From home, click on "Workload", then click "Job Queue Reports" button on the bottom of the page.
- Resources reports:
  From home, click on "Resources", then click "Resource Reports" button on the bottom of the page.

1.11 Troubleshooting

This section covers typical problems that come up in NetworkComputer. The command `nc summary` will be useful here, as it tells you how many jobs are failed, queued, or idle, and what the queued jobs are waiting for. For example:

```
% nc summary
NC Summary For User bkring
TOTAL JOBS               0      Duration: 0s
Done                     0
Idle                     0
Queued                   0
Running                  0
Failed                   0
```

1.11.1 Common Problems

1.11.1.1 My job won't start!

Often, your job won't start because it is waiting for a resource, usually a license, CPU, or memory.

To diagnose this, use the command:

```
nc info jobid
```

A job will only start if all resources requested by the job are available. If any resource is missing, the job will not start. You can look at the resources of all available vovslaves to see if there is any that can run the job with the command:

```
nc host
```

Additionally, a vovslave must either be READY or WRKNG (have a free job slot) to accept jobs. Any other condition will prevent the vovslave from taking the job.

1.11.1.2 My job failed!

You can find the reason for job failure with the command:

```
nc info jobid
```

Some common failure conditions include:

- The job failure has nothing to do with NetworkComputer. Run the job without NetworkComputer to verify this.
- The job command doesn't exist, possibly because of a typo.
- You are using a wrong, nonexistent, or incomplete environment with the `-e`. In this case, `nc info jobid` will tell you that it cannot switch to the environment.
- You have failed to specify (or specified the wrong) architecture or memory usage. This can be done with the `-r` option. For example, `-r linux64` for linux 64 bit or `-r RAM/2000` for 2GB of ram.

2 Administrator Tutorials

2.1 Administrator tutorials

In the following administrator tutorials, we will experiment with most issues that a NetworkComputer administrator will need to address, including start/stop the server, configure fairshare, resources, slaves, and environments,

We will start a test queue to make our experiment non-disruptive to the default NetworkComputer queue.

2.2 Runtime Licensing

Runtime (RTDA) supports two kinds of licensing:

- Runtime keyfile - read directly by vovserver, with hostid of the NC host
- Reprise License Manager (RLM) - provides floating licenses, with hostid of the license server host

An Runtime keyfile license contains a comment section and encrypted data representing the features. It should be placed in the .swd of the corresponding vovserver, or in one of the following locations.

```
$VOVDIR/local/license.{nc,lm}.key
```

2.2.1 Check the license

You can verify your keyfile license information this way.

- Use command `vovreadlicutil`

```
% vovreadlicutil -p nc -P 6271 -f $VOVDIR/local/license.nc.key
License file: $VOVDIR/local/license.key
Primary host: alpaca
    ----------------------------------------
    -- VOV License  Key  Information --

    License Number:  201104281356
    VovVersion:      2011.03

    Product:     nc
    Licensed to: Runtime Design Automation - Santa Clara
    For host id: MAC:00:d0:b7:75:b9:19
    For host:    panther
    For port:    6339
    Feature:     ALL
    License for: 6 slots
    StartDate:   Wed Apr 27 22:00:00 2011
    Expiration:  Sat Dec 31 21:00:00 2011
    ----------------------------------------
    (license information)
```

`vovreadlicutil` displays the license information of the license file specified by the -f option, with respect to the product and port number.

- View the license file itself

```
% more $VOVDIR/../../vnc/vnc.swd/license.key
#    ------------------------------------
#    -- VOV License  Key  Information --
#    License S/N: 201104281356
#    VovVersion:  2011.03
#    Product:     nc
#    Licensed to: Runtime - Design - Automation - Santa - Clara
#    For host id: MAC:00:14:2a:38:80:5b
#    For host:    croc
#    For port:    0
#    License for: 6 slots
#    StartDate:   Wed Apr 27 22:00:00 2011
#    Expiration:  Sat Dec 31 21:00:00 2011
#    ------------------------------------
#
EFIFFFegjdEECEmdedceCGodDDLEGGlefeEEBCfeIDDE@EOGad
ieieneC@feO@meo`LDHGFEgcKEaaadjaBEgaJEidngB@fdEDKB
maceNE``adma`cldfaG@baIFB@`acaEFNAfcCDiehfbdEDCEdc
ld`ahgm`ldj`@GB@neD@KFaajdL@IEJA`dg`NEndCCkdAEIDLE
FBhdd`fb@@EBG@ic``ifkdCDie```ebeBFneLDIGEABE``ddad
meng@EAEJE`addneBDefifbdDBHDGA`edangI@ABJCf`EDLBBD
g`ED@@D@ECAABBkaGCh`eemcicfdAEDDFEeecdK@aea`neJAfd
EAB@j`JAialeoaHAEEG@eekdECAEBEndM@FDdec`M@AFGDddf`
JGgb@AI@IBLCJ@cdA@IEFAbfCGaaceNBefm`dengFEfaGEfa@D
FBjeEFGEjfBEo`IEFFedddee`edfgeBDedbeKEkeBAgdKEid`a
gaL@FAbdLDAEBDfd@AIGNG`e
```

2.2.2 Start/Stop the RLM license server

If you are using an RLM license file instead of an Runtime keyfile, you will need to start the license server on the host for which the license is made. The other Runtime programs locate the RLM license server by referring to the RLM_LICENSE environment variable.

Usually the RLM server directory will contain the platform-specific binaries for the license server host, including `rlm`, `rlmutil`, `rtdad`, and symbolic links from rlmutil to the usual command names like `rlmstat`, etc.

We recommend disabling the built-in web server and broadcast (starting in RLM 10)

```
% cd /opt/rlm
% ./rlm -nows -noudp -c rtda.lic -dlog reprise.log &
```

```
% printenv RLM_LICENSE
7070@buffalo
% rlmstat -avail
```

```
% cd /opt/rlm
% ./rlmutil rlmdown RLM
rlmdown v11.2
Copyright (C) 2006-2014, Reprise Software, Inc. All rights reserved.

shutting down RLM - are you sure? [y/n]: n
```

2.3 Start a Test Queue

At a given site, we recommend a single queue, i.e. a single NetworkComputer setup, by default called "vnc". Such a setup is called a 'cluster' by other systems. The scheduler in NC does not have queues in the sense used by other batch systems.
In this tutorial, we will start a temporary NetworkComputer queue for our testing. This allows us to experiment with most of the administration tasks without disturbing your production NetworkComputer queue.

2.3.1 Server working directory

The NetworkComputer configuration directory for the default queue named "vnc" is in the Runtime hierarchy. The server working directory for this queue is:

```
$VOVDIR/../../vnc/vnc.swd
```

This directory contains the server configuration files and server/slave logs, etc.

2.3.2 Start a queue

The command to start a queue is `ncmgr start`. By default, this command will start the default queue in the directory of `$VOVDIR/../../vnc`.

1. Get usage of this command:

```
% ncmgr start -h

 USAGE:
    % ncmgr start [options]
 OPTIONS:
    -h                      This help.
    -force                  Do not ask confirmation.
    -queue <name>           Name for queue (default is vnc).
    -dir   <dir>            Directory of the server (default %VOVDIR/../../vnc).
```

2. Start our test queue in our home directory:

   Again, since our queue is just for testing purpose and temporary, we will start it in our home directory. We name our queue "vncdexin" (make sure you pick a name that does not conflict with any existing queue).

   For these training tutorials, use the name formed by prefixing your login with `vnc`. For example, if your login is 'danny', use the name 'vncdanny' for your queue.

```
+ start a shell on the machine where your NC vovserver should run
% cd                # Go to your home directory
% mkdir ncadmin     # Create a directory for our testing queue
% ncmgr start -dir ncadmin -queue vncdexin
 message: Checking the license...
 message: ... the license is good.
 message: Starting NC
 message:   with name    vncdexin
 message:   on host      alpaca
 message:   in directory /home/dexin/ncadmin
 message:   as user      dexin
Do you want to proceed? (yes/[no]) > yes
 message: Updating config file '/remote/release/VOV/2013.09/linux64/local/vncConfig/vncdexin.tcl'
 message: Waiting for server to be ready ...
 message: Sanity check...
 message: NC vncdexin@alpaca is ready.
```

Note: make sure the directory `$VOVDIR/local/vncConfig` is writable to you, because a config file needs to be written in that directory to start a new queue. The Runtime installer should have set this.

2.3.3 Using a particular queue

Now that you start a new queue, you have at least two NetworkComputer queues in your system. There are two ways to use a particular queue.

- Use `-q` or `-queue` option:

```
# Submit a job to queue "vncdexin"
% nc -q vncdexin run sleep 60

# List my jobs in queue "vncdexin"
% nc -q vncdexin list

# Get info about queue "vncdexin"
% ncmgr info -queue vncdexin

# Reset all slaves for queue "vncdexin"
% ncmgr reset -queue vncdexin -slaves
```

- Set environment variable NC_QUEUE

  You can also set an environment variable NC_QUEUE to the name of queue you want. Then you can execute all your `nc` and `ncmgr` commands in the context of that queue without having to use `-q` or `-queue` options.

```
% setenv NC_QUEUE vncdexin

# Now submit a job to queue "vncdexin"
% nc run sleep 61

# etc.
```

This way, you can easily switch between all the queues you have. We will set NC_QUEUE to our test queue "vncdexin" for our later tutorials.

2.4 Start/Stop NetworkComputer

The commands involved in this tutorial are `ncmgr info`, `ncmgr start` and `ncmgr stop`.

You should use `ncmgr`, especially when starting NC, because it checks for and applies any daemon and DB schema changes when moving to a new version.

NOTE: Please remember to use the shell where you set the NC_QUEUE environment variable to the name of your training tutorial queue.

2.4.1 View NetworkComputer status

Use command `ncmgr info` get a summary information of all slaves and current running jobs.

```
% ncmgr info
ncmgr: message: NC vncdexin@alpaca
1 001  alpaca   1/0  146198 Unlim.        IDLE
2 002 cheetah   1/0  235294 Unlim.        IDLE
3 004   bison   1/1   99206 Unlim.        FULL
     2s:       vw vwrap sleep 60 > vnc_log
4 000   pluto   1/0   75312 Unlim.        IDLE
5 003   rhino   1/0  181818 Unlim.        IDLE
```

2.4.2 Start NetworkComputer

If it is already running, `ncmgr` will tell you so.

```
% ncmgr start
ncmgr: USER ERROR: NC vncdexin@alpaca already running.
```

Now start it assuming it is not running. In production use, it is important to check the number of file descriptors available to vovserver.

```
% ncmgr start
ncmgr: message: Checking the license...
ncmgr: message: ... the license is good.
ncmgr: message: Starting NC
ncmgr: message:   with name    vncdexin
ncmgr: message:   on host      alpaca
ncmgr: message:   in directory /home/dexin/ftadmin
ncmgr: message:   as user      dexin
ncmgr: message:   with         1024 file descriptors
Do you want to proceed? (yes/[no]) > yes
ncmgr: message: Waiting for server to be ready ...
ncmgr: message: Sanity check...
ncmgr: message: NC vncdexin@alpaca is ready.
```

2.4.3 Stop NetworkComputer

```
% ncmgr stop
ncmgr: message: Checking if NC vncdexin@alpaca is running...
ncmgr: message:
You are about to stop NC vncdexin@alpaca
in directory /home/dexin/ftadmin
    Would you like to proceed (yes/[no]) ? > yes
ncmgr: message: Stopping slaves and server ...
ncmgr: message: NC vncdexin@alpaca has been stopped.
```

2.4.4 Re-start NetworkComputer training tutorial queue

You should re-start your training tutorial queue so that it will be running for later exercises.

```
% ncmgr start
ncmgr: message: Checking the license...
ncmgr: message: Checking the license...
ncmgr: message: ... the license is good.
ncmgr: message: Starting NC
ncmgr: message:   with name    vncdexin
ncmgr: message:   on host      alpaca
ncmgr: message:   in directory /home/dexin/ftadmin
ncmgr: message:   as user      dexin
    Do you want to proceed? (yes/[no]) > yes
ncmgr: message: Waiting for server to be ready ...
ncmgr: message: Sanity check...
ncmgr: message: NC vncdexin@alpaca is ready.
```

2.5 Browser-based Setup

NetworkComputer provides a simplified user-friendly browser-based setup, which is especially useful for new users.
To use this browser-base setup page, please first make sure NetworkComputer is running.

2.5.1 Find the URL

First find the Network Computer URL, using vovbrowser or vsi:

```
% nc cmd vovbrowser
http://yourhost:6295/project
% nc cmd vsi
  (more detailed output, including the NC vovserver URL)
```

The setup script is available at the URL `/cgi/setup.cgi` (for example, in this case, it is `http://alpaca:6295/cgi/setup.cgi`).

2.5.2 Setup using the page

Follow the instructions on the page to finish the basic setup. On the "slaves" setup page, try to add at least one slave for each type (Server, Workstation, Offhours). After adding the slaves, you can view the slave statuses at URL `/slaves`. Then you can go to try out with some test jobs at the last step of basic setup. You need a working remote-shell setup to start non-local vovslaves.

2.6 Configure Policy - Fair Share and Other Parameters

2.6.1 Configure fairshare

NetworkComputer has a time-windowing multi-level fairshare mechanism. This allocates CPU cycles to the fsgroups at each level according to the assigned weights of each group until the leaf nodes of the fairshare tree are reached. Each node of the fairshare tree may have a separate time window. We use the word 'fsgroup' as an abbreviation here.

To configure fairshare, you use the `vovfsgroup` command.

Access to fairshare groups is controlled by ACLs (Acces Control Lists), so you can configure them so only designated users can submit jobs in an fsgroup.

2.6.1.1 Locate server configuration directory to find policy.tcl

If you remember, we start our test queue vncdexin in directory `~/ncadmin`. So the server configuration directory for this NetworkComputer queue is `~/ncadmin/vncdexin.swd`. By default, the server configuration directory is `$VOVDIR/../../vnc/vnc.swd`.
If you forget, you can find it out by this command:

```
% nc cmd vovserverdir
/home/dexin/ncadmin
```

In this case, you will find `policy.tcl` file at `/home/dexin/ncadmin/vncdexin.swd/policy.tcl`.
You will find other configuration files in the same directory, for example, `slaves.tcl`, `resources.tcl`. We will cover those later or in documentation.

2.6.1.2 Configure policy.tcl for fairshare

Here in our example, we configure two groups `production` and `regression`.

```
# in policy.tcl
% vovfsgroup create /production -weight 400 -window 8h
% vovfsgroup create /regression -weight 100 -window 8h
```

In the above example, we define two groups `/production` and `/regression` When there are jobs from both `production` group and `regression` group, the target share ratio of CPUs will be 400:100.

2.6.1.3 Save the new configuration

Changes made by the `vovfsgroup` command are only in the NC vovserver's memory until the the next save, and may be lost if vovserver is restarted before then. We will save the configuration to a file so it can be reloaded.

```
% nc cmd vovfsgroup genconfig myfsconfig.tcl
```

2.6.1.4 Test the new fairshare configuration

You use `-g` option in command `nc run` to submit a job from a particular group.

```
% nc run -g regression -f listOfJobs
% nc run -g production -f listOfJobs
```

You can use the monitors GUI or browser page to monitor the dynamic changes of fairshare.

- Use the monitors: use command `nc monitor` to bring up the monitors and click on "Fair Share" tab.
- Use the browser: use command `nc cmd vovbrowser` to find the NetworkComputer URL, then find fairshare page at URL `/cgi/fairshare.cgi`.

2.6.2 Server configuration parameters

Certain parameters of the NetworkComputer vovserver are also configurable by means of entries in the `policy.tcl` file. The Server Configuration page (at `/doc/VOV_AdminGuide/srvconfig.html` describes these in detail.)
Here we give some examples:

```
# This is part of the policy.tcl file.
set config(maxQueueLength)       8000
set config(httpSecure)           1
set config(saveToDiskPeriod)     2h;
set config(autoLogout)           1h;      # Logout from browser interface.

# Used by NC for autoforget.
set config(autoForgetValid)      1h
set config(autoForgetFailed)     2d
set config(autoForgetOthers)     2d
set config(autoRescheduleThreshold) 2s
```

2.7 Advanced Policy Configuration

Besides the vovserver configuration file `policy.tcl`, the behavior of job submission to NetworkComputer can also be controlled by the file `$VOVDIR/local/vnc_policy.tcl`, which is used to define the following procedures:

VncPolicyDefaultResources	The default resources required by a job.
VncPolicyValidateResources	Make sure that the resource list for a job obeys any number of rules.
VncPolicyDefaultPriority { user }	Assign the default priority to a job based on the user.
VncPolicyMaxPriority { user priority }	Limit the priority based on the maximum allowed to the user.

In this tutorial, we will configure `VncPolicyDefaultResources` and `VncPolicyValidateResources`.

2.7.1 Configure default resources

By default, NetworkComputer takes machine architecture as the resource of the job that you submit from a particular machine. This is controlled by the default setting of `VncPolicyDefaultResources`:

```
proc VncPolicyDefaultResources {} {
    global env
    return "$env(VOVARCH)"
}
```

If you want to change this behavior, you can create or edit the file `$VOVDIR/local/vnc_policy.tcl`, and add or edit the procedure that follows. This procedure will set the default resources to be the architecture and 50mb of memory. If you have more than one NC setup, you can place the vnc_policy.tcl file in the .swd and it will apply only to that one.

```
proc VncPolicyDefaultResources {} {
    global env
    return "$env(VOVARCH) RAM/50"
}
```

2.7.2 Enforce some rules of job resources

You can also enforce some rules of job resources by overriding the procedure `VncPolicyValidateResources`. Here is the default behavior:

```
proc VncPolicyValidateResources { resList } {
    return $resList
}
```

which does nothing. Let's say you want to enforce a rule so that all jobs require a minimum RAM of 512 MB.Again, you can create or edit the file `$VOVDIR/local/vnc_policy.tcl` and add or edit `VncPolicyValidateResources` procedure as follows:

```
proc VncPolicyValidateResources { resList } {
#
# This policy adds a minimum RAM requirement
# for all submitted jobs.
#

if [regexp "RAM/" $resList] {
    # Already a RAM constaints.
} else {
    lappend resList "RAM/512"
}
return $resList
}
```

2.8 Configure Resources

NetworkComputer includes a sophisticated subsystem for the management of computing resources, which allows the design team to take into account all sorts of constraints regarding hardware and software resources, as well as site policy constraints. This mechanism is based on:

- The resources required by jobs
- The resources offered by slaves
- The ResourceMap, as described in the file `resources.tcl`

In this tutorial, we will configure the resource map in `resources.tcl`.

2.8.1 Find and view resources.tcl file

You can find this file in the server configuration directory. For default `vnc` queue, it is `$VOVDIR/../../vnc/vnc.swd`. And for our test queue, it is in directory `~/ncadmin/vncdexin.swd/`.

```
% cd ~/ncadmin/vncdexin.swd
% vi resources.tcl ; # Use the editor of your choice
```

```
# ... here we only show part of this file ...
vtk_resourcemap_set PRIORITY_LOW    1
vtk_resourcemap_set PRIORITY_NORMAL 10
vtk_resourcemap_set PRIORITY_HIGH   20
vtk_resourcemap_set PRIORITY_TOP    UNLIMITED
```

With above default configuration, there will be at most 1 low priority job running at any time, 10 for normal, 20 for high, and any number of top priority jobs could be running.

Let us do a simple test to verify that:

```
% nc run -f $VOVDIR/training/vnc/cmdlist.unix
```

You can use the monitors window to monitor the "Running Jobs" and "Resources" to see the running jobs and resources usage. You can also use browser to get similar information from "Running Jobs" page and "Resources" page.

2.8.2 A simple resource configuration example

For example, if you decide that no more than 4 normal priority jobs should be running at any time, you can edit `resources.tcl` and modify the value for normal priority to 4.

```
# ... here we only show part of this file ...
vtk_resourcemap_set PRIORITY_LOW    1

# Now we change this value to  4
vtk_resourcemap_set PRIORITY_NORMAL 4

vtk_resourcemap_set PRIORITY_HIGH   20
vtk_resourcemap_set PRIORITY_TOP    UNLIMITED
```

Save your change and do a reread and test the new configuration:

```
% nc cmd vovproject reread
% nc run -f $VOVDIR/training/vnc/cmdlist.unix
```

2.8.3 More examples

You can configure your resources similarly. For example, you have 10 calibre license, you can configure this by adding the following line in `resources.tcl`:

```
# In resources.tcl
vtk_resourcemap_set calibre_license 10
```

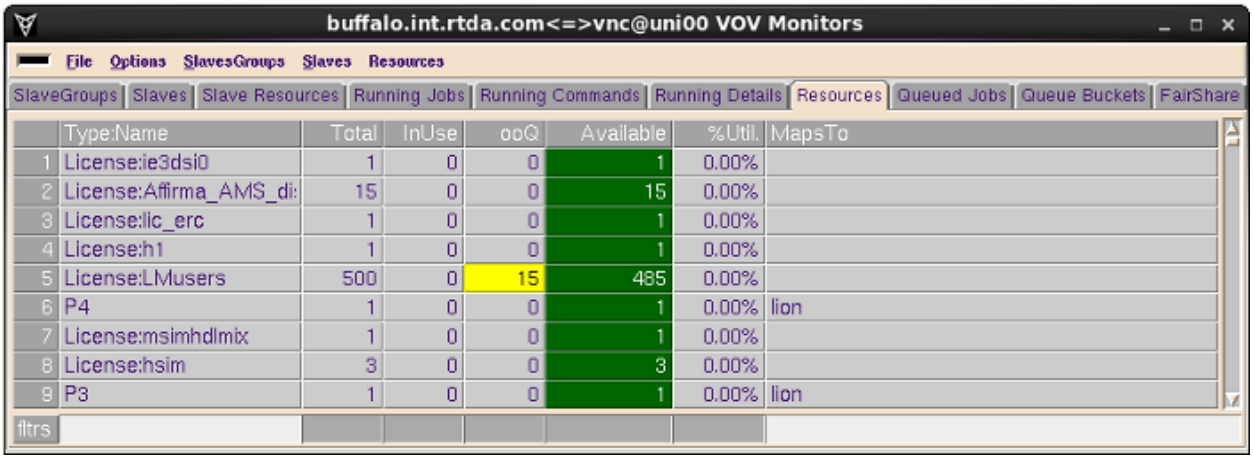
You can also associate a resource to other resource(s). For example, you have 4 licenses of spice that are only available on linux. You can configure this by add the following line in `resources.tcl`:

```
# In resources.tcl
vtk_resourcemap_set hspice_license 4 linux
```

2.8.4 The Resource Monitor

To monitor resource activity, you can call a GUI monitor with the command:

```
nc mon
```



2.9 Configure Security

Runtime software has 4 privilege levels: READONLY, USER, LEADER, ADMIN. For detailed information about security, please refer to documentation at `../VOV_AdminGuide/securityinfo.html`.

2.9.1 Locate the security configuration file: security.tcl

This file is in the server configuration directory, default `$VOVDIR/../../vnc/vnc.swd/security.tcl` and in our test setup, that is `~/ncadmin/vncdexin.swd/security.tcl`.

2.9.2 Example: Least restrictive security

The least restrictive security grants everybody full access from any host. This should not be used in production.

```
# All users (+) are administrators from all hosts (+).
vtk_security + ADMIN +
```

Or alternatively, a VovUserGroup may be utilized, to assign individuals in a group the ADMIN privilege. See the VovUserGroup page for more information on how to set up and administer these groups.

```
# Members of mygroup are administrators from all hosts (+).
vtk_security -group mygroup ADMIN +
```

2.9.3 Example: Most restrictive

```
# No rule defined gives only the owner of the project ADMIN privileges
# on the server host.
```

2.9.4 Example: Typical case

The following example shows a typical security file, in which different privileges are granted to different users. Also notice the use of variables and VovUserGroups in this example.
In the example, `mary` is an administrator for any host, and `dan` is an administrator only for `reno` and `milano`. The user `pat` is a LEADER for her machine `elko`, and `fred` has USER privileges for 4 machines listed in the variable `$allhosts`. Members of the VovUserGroup "operators" have ADMIN rights on `$allHosts`.

```
set servers      { reno milano }
set allhosts     { reno milano elko tahoe }

vtk_security mary      ADMIN   +
vtk_security john      ADMIN   tahoe
vtk_security dan       ADMIN   $servers
vtk_security pat       LEADER  elko
vtk_security fred      USER    $allhosts
vtk_security -group operators ADMIN   $allHosts
```

2.10 Configure Slaves

The file `slaves.tcl` describes the vovslaves for NetworkComputer. Refer to documentation for more information about vovslaves configuration.

2.10.1 Simple vovslave configuration

You can find the file `slaves.tcl` in server configuration directory, default at `$VOVDIR/../../vnc/vnc.swd/slaves.tcl`, and in our test case at `~/ncadmin/vncdexin.swd/slaves.tcl`.
This file is based on two commands (Tcl procedures),`vtk_slave_define` and `vtk_slave_set_default`:

```
# Set default behavior of vovslaves
vtk_slave_set_default [options]

# Define a vovslave
vtk_slave_define hostname [options]
```

If you set some options with `vtk_slave_set_default` command, all the following `vtk_slave_define` commands will use those options implicitly, and options set explicitly in `vtk_slave_define` overwrite those set in `vtk_slave_set_default`.

For example, to declare 3 vovslaves on the hosts apple, orange, and pear, use:

```
# In slaves.tcl
vtk_slave_define apple
vtk_slave_define orange
vtk_slave_define pear
```

or some equivalent code:

```
# In slaves.tcl
foreach host {apple orange pear} {
    vtk_slave_define $host
}
```

To understand the use of `vtk_slave_set_default`, let's say we define 3 vovslaves as follows:

```
vtk_slave_define apple  -resources "@STD@ big_memory" -CPUS 2
vtk_slave_define orange -resources "@STD@ big_memory" -CPUS 2
vtk_slave_define pear   -resources "@STD@ big_memory" -CPUS 4
```

`vtk_slave_set_default` does the following:

```
vtk_slave_set_defaults -resources "@STD@ big_memory" -CPUS 2
vtk_slave_define apple
vtk_slave_define orange
vtk_slave_define pear    -CPUS 4
```

2.10.2 More examples

The default `slaves.tcl` provides pretty good examples of configuring vovslaves. Most of the time, you just need to plug in some host names in brackets to define vovslaves. For example, here is one piece of code in `slaves.tcl`

```
# ADD THE NAMES OF THE COMPUTE SERVERS TO THE FOLLOWING LIST
set mainComputeServers {}

foreach host $mainComputeServers {
    vtk_slave_define $host -resources "VovResources::Standard @RAMTOTAL@ @SWAPFREE@"
}
```

(**Note:**`-resources "VovResources::Standard"` is equivalent to `-resources "@STD@"`)

To add some "Server" class vovslaves, you just need to add the names of those hosts into the list `mainComputerServers`, like the following:

```
set mainComputeServers { apple orange pear }
```

You can certainly add or modify the vovslave definition as you want(subject to license restriction). For example, you have many dual CPU machines, and you would like to make the `maxload` of these vovslave machines bigger, say, equal to 1.5 times of the CPUs, i.e. 3.0. Then you can do this:

```
set myDualCpuServers {}

foreach host $myDualCpuServers {
    vtk_slave_define $host  -maxload   3.0      -resources "VovResources::Standard @RAMTOTAL@ @SWAPFREE@"
}
```

2.10.3 Add Workstation/Offhours vovslaves

```
# Add a Workstation vovslave that will only start to accept
# job after 10 minutes of ilde and will only accept jobs
# with expected duration no longer than 5 minutes
vtk_slave_define ftcsun44  -resources "VovResources::Workstation -minIdle 10m -maxtime 5m @RAMTOTAL@"

# Add a Offhours vovslave that's only available from 7pm to 6am
# every weekday and on weekends
vtk_slave_define ftcsun66 -resources "VovResources::Offhours"
```

2.10.4 Make your new configuration work

Like other configurations, NetworkComputer will pick up the changes in `slaves.tcl` when the server is stopped and restarted. This is often too disruptive and not favorable, especially when there are jobs running in NetworkComputer. Here are some other ways to apply your changes:

- To start the vovslaves that you newly defined, use command:

  ```
  % ncmgr reset -slaves
  ```

- To start vovslaves that you just modified, you have to stop and start them.

2.10.5 Start/Stop vovslaves (advanced)

You can start/stop vovslaves from GUI and browser. Here we introduce an advanced command of FlowTracer `vovslavemgr` and show how to run any Runtime command in the context of NetworkComputer using `nc cmd`.

```
# List all vovslaves defined in slaves.tcl
# Also checks the slaves.tcl file for syntax errors
% nc cmd vovslavemgr list

# Start all vovslaves defined in slaves.tcl
% nc cmd vovslavemgr start

# Start some vovslave(s)
% nc cmd vovslavemgr start slave1 slave2

# Stop all vovslaves
% nc cmd vovslavemgr stop

# Stop some vovslave(s)
% nc cmd vovslavemgr stop slave1 slave2

# Show detail information of all vovslaves
% nc cmd vovslavemgr show

# Get usage of all vovslavemgr commands
% nc cmd vovslavemgr
```

2.10.6 Start a vovslave from command line (advanced)

You can also start a vovslave on the fly from the command line using the `vovslave` binary. This can sometimes be handy, for example, for debugging. This is the command that NetworkComputer uses to start the vovslaves after it reads the `slaves.tcl` configuration file.

To start a vovslave on a particular host, you need to go to that host and use command `nc cmd vovslave` with appropriate options, which are similar to the options of `vtk_slave_define`. Try the following examples and monitor the slaves using the Monitor GUI or browser slaves page.

- Get usage of this command:

```
% nc cmd vovslave
usage: vovslave [-a name] [-c coefficient] [-C cpus] [-d] [-D integer]
                [-f tclfile] [-h host] [-I tclfile] [-l logfile]
                [-M max_load] [-n <integer>] [-N] [-p project]
                [-r resources] [-s] [-t timeout] [-T max_tools]
                [-U updateInterval] [-v] [-V ncName@ncHost]
                [-z <timeSpec>] [-Z <timeSpec>] [-g name] [-u name]
                -a:         Name this vovslave
                -c:         Slave coefficient (positive, default 1.0)
                -C:         Number of CPU's in this machine (automatic on win64).
                -d:         Set debugging
                -D:         Min disk space in MB in /tmp and /usr/tmp (default 5)
                -f:         Source the given Tcl file
...OMITTED...
```

- Start a normal vovslave (with all default settings):

```
% nc cmd vovslave -N
vovslave Jan 10 13:44:42
Copyright © 1995-2016,Runtime Design Automation
Linux/7.1 Jan 10 2003 10:00:59
vnc@alpaca
vovslave Jan 10 13:44:42 Test 1: INTEGER OPS  W=  1.00 Reps= 500 T=   5.00ms
vovslave Jan 10 13:44:42 Test 1: DOUBLE  OPS  W=  1.00 Reps=  25 T=  10.00ms
vovslave Jan 10 13:44:42 Test 1: CHAR    OPS  W=  0.10 Reps=  10 T=  10.00ms
vovslave Jan 10 13:44:42 ---- Weighted time: 16.00ms
vovslave Jan 10 13:44:42 Test 2: INTEGER OPS  W=  1.00 Reps= 500 T=   5.00ms
vovslave Jan 10 13:44:42 Test 2: DOUBLE  OPS  W=  1.00 Reps=  25 T=  10.00ms
vovslave Jan 10 13:44:42 Test 2: CHAR    OPS  W=  0.10 Reps=  10 T=  20.00ms
vovslave Jan 10 13:44:42 ---- Weighted time: 17.00ms
vovslave Jan 10 13:44:42 Test 3: INTEGER OPS  W=  1.00 Reps= 500 T=   5.00ms
vovslave Jan 10 13:44:42 Test 3: DOUBLE  OPS  W=  1.00 Reps=  25 T=  10.00ms
vovslave Jan 10 13:44:42 Test 3: CHAR    OPS  W=  0.10 Reps=  10 T=  20.00ms
vovslave Jan 10 13:44:42 ---- Weighted time: 17.00ms
vovslave Jan 10 13:44:42 Best weighted time: 16.00ms
```

- Start a vovslave with name "myvovslave", max load 4.0, and offers standard resources "@STD@" and resource "special_license":

```
% nc cmd vovslave -a myvovslave -M 4.0 -r "@STD@ special_license"
vovslave Jan 10 13:51:00
Copyright © 1995-2016,Runtime Design Automation
Linux/7.1 Jan 10 2003 10:00:59
vnc@alpaca
vovslave Jan 10 13:51:00 Test 1: INTEGER OPS  W=  1.00 Reps= 500 T=   5.00ms
vovslave Jan 10 13:51:00 Test 1: DOUBLE  OPS  W=  1.00 Reps=  25 T=   5.00ms
vovslave Jan 10 13:51:00 Test 1: CHAR    OPS  W=  0.10 Reps=  10 T=  20.00ms
vovslave Jan 10 13:51:00 ---- Weighted time: 12.00ms
vovslave Jan 10 13:51:00 Test 2: INTEGER OPS  W=  1.00 Reps= 500 T=   5.00ms
vovslave Jan 10 13:51:00 Test 2: DOUBLE  OPS  W=  1.00 Reps=  25 T=   5.00ms
vovslave Jan 10 13:51:00 Test 2: CHAR    OPS  W=  0.10 Reps=  10 T=  20.00ms
vovslave Jan 10 13:51:00 ---- Weighted time: 12.00ms
vovslave Jan 10 13:51:00 Test 3: INTEGER OPS  W=  1.00 Reps= 500 T=   5.00ms
vovslave Jan 10 13:51:00 Test 3: DOUBLE  OPS  W=  1.00 Reps=  25 T=   5.00ms
vovslave Jan 10 13:51:00 Test 3: CHAR    OPS  W=  0.10 Reps=  10 T=  20.00ms
vovslave Jan 10 13:51:00 ---- Weighted time: 12.00ms
```

2.11 Configure an Environment

VOV includes a sophisticated subsystem to manage environments. In NetworkComputer, an environment is a named collection of environment variables and their values. Many tools expect certain environment variables to be set for correct operation, and the variable PATH is used by most Unix shells to locate executable programs.

A VOV environment definition consists of three scripts defined in either Tcl, C-shell, or Bourne-shell syntax. Environments that will be used on both Unix and Windows must be written in Tcl. The names of these scripts must follow a naming convention. For example, an environment named ENV would use these three scripts:

- ENV.start.tcl
- ENV.end.tcl
- ENV.doc

The 'start' script is used when entering the environment, the 'end' script is used when leaving the environment, and the 'doc' script contains a one-line summary of the environment's purpose.

VOV provides commands for using environments from the command line:

`vel`	list available environments
`ves ENV`	switch to environment ENV
`vep`	change prompt to show environment name
`veprestore`	return to original prompt

In this lab, we will:

- look at some of the standard VOV environments

- configure an environment definition in the system-wide directory `$VOVDIR/local/environments`
- learn how to specify other environment directories
- learn how to compose 'good' environment scripts
- learn how to troubleshoot environment definitions

2.11.1 Find and view example environment scripts

It is interesting and instructive to examine some of the environment definitions which are shipped with VOV. One of the most important is the BASE environment.
The most common place for environment scripts is `$VOVDIR/local/environments`, but you can also specify additional directories where VOV will search. We will discuss this in more detail later in the lab.
The BASE definition is in the 'local' directory of the VOV software installation, at `$VOVDIR/local/environments/BASE.start.tcl`.

```
% pushd $VOVDIR/local/environments
% view BASE.start.tcl
```

```
# --
# -- Completely reset the environment to bare bones.
# -- We assume that VOVDIR and VOVARCH are set.
# --

set VOVDIR $env(VOVDIR)

if { $::tcl_platform(platform) eq "windows" } {
  #
  # --
  #
  nt_preprocess_env
  #    set vovenv(debug) 1
  set WINDIR $env(WINDIR)
  setenv PATH "$WINDIR\\system32;$WINDIR"

  set MKSDIR "c:/mksdemo"
  if [file exists $MKSDIR] {
        setenv ROOTDIR $MKSDIR
        vovenv PATH ";" APPEND $MKSDIR/mksnt
  }
  #
  # Depending on the installation, the binaries can be in
  # different directories.
  #
  foreach b { bin bin-O bin-g bat local/bat } {
        set p "$VOVDIR/$b"
        #
        # Eliminate double // as in w://bat
        #
        regsub -nocase {^([a-z])://} $p {\1:/} p

        if [file exists $p] {
            vovenv PATH ";" APPEND $p
        }
  }

  set version [exec vovversion]
  vovenv PATH ";" PREPEND c:/temp/vov/execache$version

  # vovenv PATH ";" PREPEND c:/temp/vov/execache

} else {
  #
  # -- Unix BASE environment.
  #

  setenv PATH    ""
  foreach b { bin bin-O bin-g } {
        if [file exists $VOVDIR/$b] {
            vovenv PATH : APPEND  $VOVDIR/$b
            break
        }
  }
  vovenv PATH : APPEND  $VOVDIR/scripts
  vovenv PATH : APPEND  $VOVDIR/local/scripts

  setenv MANPATH ""
  foreach mp [list /usr/man /usr/share/man /usr/local/man /usr/openwin/man $VOVDIR/man] {
        if [file isdirectory $mp] {
            vovenv MANPATH :  APPEND  $mp
        }
  }

  setenv LD_LIBRARY_PATH ""
  foreach lp [list /lib /usr/lib /usr/local/lib /usr/openwin/lib $VOVDIR/lib] {
        if [file isdirectory $lp] {
            vovenv  LD_LIBRARY_PATH : APPEND $lp
        }
  }

  foreach xapp { /usr/openwin/lib/app-defaults /usr/lib/X11/app-defaults } {
        if  [file isdirectory $xapp ]  {
            vovenv XFILESEARCHPATH : APPEND "$xapp/%N"
        }
  }

  # -- /opt/aCC/bin is for HPUX
  foreach p {
        /bin /usr/ucb /usr/bin /usr/local/bin
        /usr/X11/bin
        /usr/dt/bin
        /usr/openwin/bin /usr/bin/X11 /opt/aCC/bin
  } {
        if [file isdirectory $p] {
            vovenv PATH : APPEND $p
```

```
        }
  }
}
```

The BASE environment includes just the regular system commands and VOV. This is an example of a complex environment setup script; most of the ones you will need to define will be much simpler.

Let us do a simple test to verify that:

```
% vep
```

Notice that the prompt has changed to show that you are in the `DEFAULT` environment. This is the environment which is defined by your regular system setup files, e.g. `.cshrc`.

```
% ves BASE
% printenv PATH
```

Notice that the PATH environment variable is now (probably) much shorter, and contains the VOV software directories.

```
% veprestore
```

Notice that the prompt has been changed back to the original. NOTE: this only changes the prompt. Your shell still has whatever environment was most-recently set using `vep`.

2.11.2 Adding additional Environment directories

When you are testing and developing environments, or when you want to have project-specific ones which are not put in the system wide directory `$VOVDIR/local/environments`, you can use the environment variable `VOV_ENV_DIR` to specify additional directories.

We will use this capability in the lab, so that we don't need to put our environment scripts in the system's shared environment area. You should have a `~/ncadmin` directory from the previous labs, which contains the server working directory of your private NetworkComputer queue.

```
% setenv VOV_ENV_DIR ~/ncadmin/vnc-user-name.swd/environments
% pushd $VOV_ENV_DIR
```

BEWARE:The `enviroments` directory is generated by setup, but any user-made env-scripts placed in the directory will not be used unless VOV_ENV_DIR includes it.

2.11.3 Environment configuration example

In this example, we will create an enviroment which adds your personal `bin` directory to the path. We will put the definitions in our queue's environment directory.

We use Tcl for the environment scripts because it is more powerful than C-shell, and platform-independent.

2.11.3.1 Example 'start' script for JOHN environment

```
# add john bin directory to path
vovenv PATH : PREPEND /home/john/bin
```

This script adds the element `/home/john/bin` to the beginning of the PATH. There is also an `APPEND` operator which adds to the end of the PATH. If the exact element is already present in the path, it is left undisturbed. This avoids PATH overflow in C-shell.

2.11.3.2 Example 'end' script for JOHN environment

```
# remove john bin directory from path
vovenv PATH : DELETE /home/john/bin
```

This script removes the element `/home/john/bin` from the PATH.

2.11.3.3 Example 'doc' script for JOHN environment

```
prepend /home/john/bin to PATH
```

This file contains a one-line summary which is presented by the **`vel`** command.

Use the above examples to create similar files in your queue.

2.11.4 How to compose 'good' environment scripts

There are several characteristics of a 'good' environment setup.

- **minimal**
  A minimal environment only sets the variables needed to accomplish a specific purpose.
- **composable**
  A composable environment respects pre-existing values of variables, and adds to them, rather than overwriting them. For example, when modifying `LM_LICENSE_FILE`, use `PREPEND` and `APPPEND` rather than simply setting it. This allows you to use commands like `ves BASE+CADENCE+NASSDA` and have both Cadence and Nassda tools work.
- **reversible**
  A good environment definition unsets in its 'end' script what it sets in the 'start' script, so that the environment does not gradually become polluted with obsolete values.

2.11.5 Environments are Cached on the vovslave

To provide very low job dispatch latency, NetworkComputer caches the eight most-recently used environments in the vovslave process. This means that jobs which run in any of those environments may start immediately with out sourcing the environment definition script. It also means that if you change an env-script, you must tell the vovslaves by using the `vovslavemgr refresh` subcommand.

2.11.6 Troubleshooting Environments

-->

2.11.7 Summary

- Most environments are defined by scripts stored in `$VOVDIR/local/environments`
- Environment scripts are in Tcl or C-shell syntax

- You may use the `VOV_ENV_DIR` environment variable to specify additional directories to search for environment scripts.

- Environments are cached on the vovslave, and refreshed using `ncmgr reset -slaves`, or `nc cmd vovslavemgr refresh`

2.12 Logical Names (equivalences)

2.12.1 Introduction

Most sites set up their machines so that they all have uniform mountpoints and view of the filesystems. A common scheme is to place all user home directories under a single parent directory, often called `/home`. The actual filesystem is mounted using automount, and the physical filesystem is located by an NIS map.

Unlike some other batch/network computing systems, NetworkComputer does not require that all the compute machines have a uniform set of mountpoints. NetworkComputer can use what are called logical names or equivalences, to locate the data.

The default `equiv.tcl` file which is shipped with the software defines a logical name for the path to VOVDIR.
The `equiv.tcl` in some versions also defines a logical name HOMES for whatever the parent of your home directory as set in the environment variable `HOME`. The default one shipped with an early version did not, so we will show the steps to define a logical name using the browser-based setup.

2.12.2 Exercises

NOTE: Please remember to give these commands from a shell where you have set the environment variable `NC_QUEUE` to the name of your training tutorial queue. Otherwise, you will need to use the `-queue` option.

1. Use the browser-based setup to view the Logical Names which are defined. First, we find the URL of our queue's server, and connect a web browser to it.

```
% nc cmd vovbrowser --> http://host:port/project
```

```
% firefox http://host:port/cgi/setup.cgi
```

   The initial page will have a link on the left called **FileSystems**. Left-click on this link to display the logical names defined for the queue. You will see VOVDIR and VNCSWD.
2. Run a job in your home directory.

```
% cd
% nc run sleep 10
```

   This will often run, because the default is to require the architecture from which the job was submitted as a resource, so the job may run on your machine.

   Now try the job, but requesting an architecture different from the machine on which you are working. This job will fail, because the directory on the remote machine does not have a logical name.

```
% nc run -r linux64 -e BASE pwd
```

```
Resources= linux64
Env      = BASE
Command  = vw vwrap pwd
Logfile  = vnc_logs/20030108/143125.27906
JobId    = 00000021
vnc: message: Scheduled jobs: 1       Total estimated time: 0s
```

```
% nc list
00000011 Done    sleep 10
00000021 Failed  pwd
```

```
% nc info -l 21

Log file is: '${VNCSWD}/vncjohn.swd/vnc_logs/20030108/143125.27906'
vnc: message: File ${VNCSWD}/vncjohn.swd/vnc_logs/20030108/143125.27906  does not exist (yet)
        (mapped to /usr2/home/john/ncadmin/vncjohn.swd/vnc_logs/20030108/143125.27906)
```

3. run job in /tmp

Now we will try to run a job in `/tmp`, which exists on all Unix machines, but is not usually exported to the network.

```
% cd /tmp
% nc run sleep 10
```

fails, because `/tmp` is not a network path

```
% nc run sleep 10

WARNING:
  The path to the current directory is not 'LOGICAL',
  i.e. it does not begin with a '$' sign.
EXPLANATION:
  The current directory
   /tmp
  may not be a valid directory path for all hosts in the network.
OPTIONS:
    (1)   Continue.
          By choosing this option, you assert that the path
          is valid everywhere.  If it is not, the job is likely
          to fail, because the remote host cannot reach
          the current directory.
          This option causes the creation of a flag file
          called   .vnc    which has the purpose of
          avoiding the repetition of this question for this
          directory and its subdirectories
    (2)   Abort.
          Please ask your NetworkComputer administrator to
          change the equiv.tcl file to define the rules that
          give a logical name to the current directory.
```

You receive the above warning, because the directory `/tmp` is not a network path on most computers. If you respond **2**, the job will not be submitted.
If you respond **1**, a file named `.vnc` will be created in the directory, and the job will be submitted. In the future, you will not receive the warning so long as the `.vnc` file exists.
The RTDA software internally uses logical names to refer to files.

2.12.3 Advanced

1. resource cache file

```
ls ~/ncadmin/vncjohn.swd/equiv.caches
cayman  pluto
cayman abbcanova@cayman DEFAULT+SUPPORT html/ftnctraining > cat !$/pluto
```

```
cat ~/ncadmin/vncjohn.swd/equiv.caches/pluto
VOVDIR /remote/release/VOV/7.0u2/linux64
VOVDIR /remote/release/VOV/7.0u2/linux64/../common
```

2. **`vovequiv`** command

```
% vovequiv -p .  # show the logical name for this directory
```

2.13 Section 2-13

2.13.1 Introduction

This exercise will walk you through some of the more advanced and complex issues of resource management in NetworkComputer. You should already be familiar with NetworkComputer and basic resource management, covered in the Configure Resources tutorial.

With NetworkComputer, Runtime includes a simplified version of our LicenseMonitor product, called LicenseMonitor-basic, which interfaces between NC and licensing systems like FLEXlm. Unlike the full LM product, LM-basic does not do history or denials.

2.13.2 Review

The following topics are review from the Basic Resource Configuration Tutorial:

- Find and view `resources.tcl` file

- The **`vtk_resourcemap_set`** procedure
- Static Resource (PRIORITY_LOW) configuration example

- Configure resources map on the fly using `vtk_resourcemap_set`

Topics of this tutorial:

- LicenseMonitor-Basic setup

- **`vtk_flexlm_monitor`** procedure
- browser-based setup
- resource throttling

In this tutorial, we will start LM-Basic, configure it, and use it to monitor licenses. There are two things necessary to monitor FLEXlm features:

1. Configure and start the `LM-Basic` vovserver.

2. Accept the default `vtk_flexlm_monitor_all`, or add specific `vtk_flexlm_monitor` statements to NC's `resources.tcl`

2.13.3 LM-Basic setup

Runtime uses a two-layer interface to FLEXlm. There is a daemon `vovresourced` which is started by the NC server whenever there are `vtk_flexlm_monitor` statements in the `resources.tcl` file. `vovresourced` gets license information from `licmon` via HTTP, and uses it to maintain NC's License: resources.

In the case where many NC queues or Flowtracer projects are running, it greatly reduces the load on FLEXlm and improves the consistency of the license information to run LM or LM-Basic. Additionally, `LicenseMonitor` offers a browser interface which shows licenses in use and license utilization information. Runtime recommends that you run this whenever you need to monitor FLEXlm licenses, even in cases where you only have one server running NetworkComputer.

2.13.3.1 Exercise: configure and manage `LM-Basic`

Both LM-Basic and full LicenseMonitor are managed by the `lmmgr` command. It is essential that you use this instead of other ways to start the LM vovserver, because it also takes care of any DB schema updates, auxiliary daemon and startup changes needed by newer versions.

You use the configuration file of `vovflexlmd` in LM-basic to specify which FLEXlm licenses you want the daemon to monitor. You do not need to monitor all licenses. In this exercise, we will monitor one. In most cases it is easier to use the browser UI to configure which license servers are monitored.

1. Change to the working directory of the daemon and see whether it is already running. If you see an `info.tcl` file, then check whether the process named there exists on that system.

   ```
   % cd $VOVDIR/local/vovflexlmd; ls
   ```

   Example `info.tcl` file:

   ```
   # This file is created automatically by vovlmd: DO NOT EDIT
   # If you touch this file, the process 17400 will exit.
   set host    "jaguar"
   set pid     17400
   set port    5555
   set cwd     "/remote/proj9/cadmgr/licmon/licmon.swd/vovlmd"
   set version "2.0"
   set timestamp 1441146545; # Tue Sep 01 15:29:05 PDT 2015
   ```

   BEWARE: If this directory already contains a file named `config.tcl`, your site may already be running . In this case, check for the file `info.tcl`. This file will contain the host name, process ID, and port of the daemon. You can verify that it

is running by pointing your web browser to the URL `http://host:port` using the port and host in the info file.

Even if the daemon is not running, we will use our own subdirectory to run our instance LM-basic, rather than use the default directory.

2. Create a directory to run our own LM-Basic instance.

```
% mkdir ~/ncadmin/licmon
% cd  ~/ncadmin/licmon
```

3. Start the LM-basic, picking a port which is different from the regular one, if you are running on the NetworkComputer server host. The default TCP/IP port number is 5555. You can find the ports in use by `vovproject list -a -l`. For this example, we use port 5575.

```
% cd ~/ncadmin/licmon
% lmmgr -name licmon<user> -dir . -port 5575
```

4. Prepare a configuration file with a text editor, which names one of the FLEXlm license files at your site.

```
% vovproject enable licmon<user>
% cd `vovserverdir -p vovlmd`
% vi config.tcl
```

Edit the default configuration file `config.tcl`, and enter at least one license file to monitor, using the `add_LM_LICENSE_FILE` procedure. The license may be in any format acceptable as a `-c` parameter to `lmstat`, i.e. a full pathname or in `port@host` notation. It is better to use port@host than file paths, so that NFS is not involved.

Example `config.tcl` file:

```
# config.tcl -- vovflexlmd FLEXlm configuration
add_LM_LICENSE_FILE -tag CDNS /remote/vendors/cadence/license/license.dat
add_LM_LICENSE_FILE -tag SNPS /remote/vendors/synopsys/license/pluto.dat
add_LM_LICENSE_FILE -tag SUNW 1726@saturn
```

5. Verify that LM-Basic is running and supplying license information. Examine the `info.tcl` file created by the vovlmd daemon, and point your web browser to the URL as derived above.

6. Stopping the `LM-Basic` vovserver.

Most of the NetworkComputer daemons may be stopped by touching the info file that they create. When it starts, the daemon records the timestamp of the file. Each cycle, the daemon examines the timestamp, and exits if it has changed.

For LM-basic, the auxilary daemons like `vovlmd` will stop when the LM-basic vovserver stops.

```
% vovproject enable licmon<user>
% vsi  # verify that we have the right LM instance
% lmmgr stop -name licmon<user>
```

2.13.4 vtk_flexlm_monitor procedures

Your NC setup will get license in-use info from your LM or LM-basic setup via HTTP on LM's /raw interface. The default `resources.tcl` file for NC includes the `vtk_flexlm_monitor_all` procedure. This converts every feature monitored by LM into an NC resource, the name of which is the feature name prefixed by 'License:'.

If you do not wish to have a large number of resources, you can comment out the above and put calls to the **`vtk_flexlm_monitor`** procedure in your `resources.tcl` file to specify which FLEXlm features NC should monitor.

2.13.5 Resource throttling

In some cases, you may wish to restrict the number of a feature which are consumed by NetworkComputer jobs, to leave some for 'fast' or interactive use.
To accomplish this, you can use an auxiliary resource called 'throttle_<resource>' to restrict the number of NetworkComputer jobs. Suppose you have 15 licenses of hsim, and want to leave 1 free for jobs that bypass the NetworkComputer system. You would create a resource 'throttle_hsim' with a count of 14, by adding statements like this. We map hsim_license to throttle_hsim, so that when the throttle resource is exhausted, jobs that need hsim will queue. Often such resources are named Limit:<something>.

```
vtk_flexlm_monitor hsim hsim_license throttle_hsim
vtk_resourcemap_set throttle_hsim 14
```

2.13.6 Conclusion

In this exercise, we have configured, started and stopped the `vovflexlmd`. We have also monitored a FLEXlm feature in FlowTracer by adding the `vtk_flexlm_monitor` statement to the `resources.tcl` file. We have learned how to restrict NetworkComputer jobs from consuming all of a particular feature.

-->

2.14 Upgrading NetworkComputer

There are two types of updates for NetworkComputer, patches and new versions. Please refer to documentation for how to install patches and new versions. Patches replace specific files in an installation and save an archive of the changed files so you can revert. A new release is a complete set of files, usually installed in a directory that is a sibling of your current version.

Here we copy materials from documentation "Changing to a Different NetworkComputer Server Version" to discuss how to switch from one NetworkComputer software version to another with minimum disruption.

The simplest method to change to a different server is to just use the following steps (shutdown and restart on new version):

- Install new software
- Notify users that NetworkComputer will be shut down
- At an idle time, shut down he NetworkComputer server with `ncmgr stop`

- Change startup files (`.vovrc`) to refer to the new software installation
- Start the server, running the new version, with `ncmgr start`
- Notify users that NetworkComputer is available again

In some cases, for example, it may be impractical to shut down the NetworkComputer server for even a short time:

- you have a very busy installation, and NetworkComputer is never idle
- you have jobs running and you do not want to lose them
- you want to keep the order of jobs in the queue

In such cases you can use one of the following methods.

1. Switch vovserver and vovslaves separately, using `ncmgr stop -freeze`.
2. Start a new NC queue on new version, sending new jobs to it, shutting down the current NC queue after all jobs have been retired.

Here are the steps for the `ncmgr -freeze` method.

1. From a shell with your current RTDA version, stop vovserver with `ncmgr stop -freeze`. This instructs vovslaves with jobs to wait for a new vovserver and reconnect to it.

2. vovslaves with no jobs should exit right away, and ones with jobs will enter the SUSP state. Their names will change to indicate they are stopping (and to permit new vovslaves to start with the regular names).

3. From a shell with the new RTDA version, start the vovserver using `ncmgr start`.

4. Check on the browser UI Admin page that vovserver is running the expected version.

Here are the steps for the overlapping queues method.

1. Create a new queue, running the new server software
2. Direct `nc run` to send new jobs to the new queue
3. After some time, all jobs in the old queue have completed
4. The old queue may be now be shut down

2.14.1 Detailed Steps for Overlapping Queues

Please read these steps to the end and understand them before doing the procedure. This procedure is preliminary, and may need to be adapted to your NetworkComputer configuration.

1. Install and verify the new software version. The `vovcheck` command may be helpful.
2. Start a new queue, for example with name `vnc2`

```
% ncmgr start -queue vnc2
```

3. Copy configuration files (`policy.tcl, resources.tcl, slaves.tcl` etc.) from the configuration directory of the old queue to that of the new queue in such a way that the two queues are functionally the same.
4. Get the new queue ready.

```
% ncmgr reset -queue vnc2 -full
```

5. In the `setup.tcl` file of the new queue, set the environment variables `NetworkComputer_OLDQUEUE` and `NetworkComputer_OLDVERSION` to point to the old queue.

```
# This is a fragment of the setup.tcl file for the new queue
# USE THESE FOR MIGRATION FROM AN OLD QUEUE.
setenv NetworkComputer_OLDQUEUE   vnc
setenv NetworkComputer_OLDVERSION 5.4.7
```

   NOTE: This goes in the `setup.tcl` file so that these environment variables will always be set in the context of `vnc2`.
6. Test the new queue. By setting the environment variable NetworkComputer_QUEUE to `vnc2`, the NetworkComputer commands will use that as the default queue.

```
% setenv NetworkComputer_QUEUE "vnc2"
% nc run sleep 10
% nc list
% nc mon
```

   NOTE: you may use the `-queue` option as an alternative to setting `NetworkComputer_QUEUE`. For example, you might use

```
% nc -queue vnc list -a            # Show all jobs in old queue.
% nc -queue vnc2 run sleep 10      # Submit a sleep job to new queue.
% nc -queue vnc2 list              # Show your jobs (only) in new queue.
```

7. Once you are satisfied that the second queue is ready, make it the default queue. The default queue is always named `vnc`, and is where the newly-submitted jobs will be placed.

```
% cd $VOVDIR/local/vncConfig
% mv vnc.tcl vnc1.tcl; ln -s vnc2.tcl vnc.tcl
```

   NOTE: in the above, we move `vnc.tcl` to `vnc1.tcl` to preserve its contents, in case you want to reactivate the original queue. The two commands are on the same line, separated by a semicolon, so that the file `vnc.tcl` will be unavailable for the shortest possible time. Check carefully!
8. Now all the newly-submitted jobs will go to `vnc2`, while commands such as `nc info`, `nc stop`, `nc forget`, etc. will be run on both queues.

   Error messages are filtered out, because a job ID which is valid in the first queue will not be valid in the second, and conversely. To see the results with messages, query the queues separately by using the `-queue` option of NetworkComputer commands.
9. When all jobs in the old queue have been retired, you can shut down the old queue.

3 Legal

3.1 Runtime Inc Copyright

3.1.1 Boost

Portions of the Boost collection of C++ libraries are used in certain Runtime software to aid in software portability across platforms.

3.1.2 Fossil SCM

The Fossil software configuration management tool is used in certain Runtime software to provide versioning capabilities for various configuration files. Fossil is released under a 2-clause BSD license:

OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The views and conclusions contained in the software and documentation are those of the authors and contributors and should not be interpreted as representing official policies, either expressed or implied, of anybody else.

3.1.3 GD Graphics Library

The built-in graphics capabilities of the software take advantage of the GD Graphics Library, which is available from http://www.boutell.com/gd. More detail on the GD Graphics Library is available at http://www.boutell.com/gd/index.html on the http://www.boutell.com web site.

The GD Graphics Library distribution used is *gd-2.0.33*.

3.1.4 GNU Utilities for Windows

The LicenseMonitor™ LicenseManager functionality employs CVS (Concurrent Versions System) to maintain a history of changes to licensing files. It also makes use of the file program in order to determine file types. This utility requires the magic, regex, and zlib DLLs on Windows. On Unix-based systems, these utilities and their required libraries are normally already present in the operating system loadset, but they are not normally present on Windows systems. The cvs, file, and gzip programs are included in the Windows distribution of Runtime software, as are the aforementioned DLLs that are required for the file program.

The distribution of CVS included in this version of Runtime software is 1.11.22. The distribution of file (and magic) included in this version of Runtime software is 5.03.3414.

The distribution of regex2 included in this version of Runtime software is 2.7.2853. The distribution of zlib1 included in this version of Runtime software is 1.2.8.

The distribution of gzip included in this version of Runtime software is 1.2.4.

All of these software components are released under the GNU Public License (GPL).

3.1.5 Graphviz -- Graphical Visualization Software

Runtime software makes use of Graphviz libraries as part of its console Graphical User Interface (GUI). The version of Graphviz distributed with this version of Runtime software is 2.38.0. The Graphviz license can be viewed at http://www.graphviz.org/License.php.

Runtime has modified the Graphviz libraries for its use with Runtime products. To obtain a copy of the modified Graphviz libraries, please contact support@rtda.com.

3.1.6 [incr Tcl]

The [incr Tcl] software is used in constructing single-file distributables of Runtime software. The distribution of [incr Tcl] included in this version of Runtime software is that which is included in the TclKit 1.8.5 distribution. [incr Tcl] is licensed under a BSD-style license:

This software is copyrighted by Lucent Technologies, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

3.1.7 Metakit

The Metakit software is used in constructing single-file distributables of Runtime software. The distribution of Metakit included in this version of Runtime software is that which is included in the TclKit 1.8.5 distribution. Metakit is licensed under an MIT-style license:

Copyright (c) 1996-2007 Jean-Claude Wippler

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

3.1.8 mysqltcl

Runtime Inc makes use of the mysqltcl MySQL Tcl Interface library. Distribution version: 3.05.

Copyright (c) 1994, 1995 Hakan Soderstrom, Enskede, Sweden and Tom Poindexter, Denver, Colorado

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice and this permission notice appear in all copies of the software and related documentation.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL HAKAN SODERSTROM OR SODERSTROM PROGRAMVARUVERKSTAD AB BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

3.1.9 nginx

The nginx software is used by Runtime software to provide the entry point for HTTP/HTTPS communications between the Runtime web server and remote web clients. The distribution of nginx included in this version of Runtime software is 1.9.2. nginx is licensed under a BSD-style license.

3.1.10 Octtools

This software also uses packages from the Octtools-5.1 distribution from UC Berkeley, namely: *errtrap options st timer utility*

Octtools are covered by the following copyright notice:

> Oct Tools Distribution 5.1
>
> Copyright © 1988, 1989, 1990, 1991 Regents of the University of California.
> All rights reserved.
>
> Use and copying of this software and preparation of derivative works based upon this software are permitted. However, any distribution of this software or derivative works must include the above copyright notice.
>
> This software is made available AS IS, and neither the Electronics Research Laboratory or the University of California make any warranty about the software, its performance or its conformity to any specification.
>
> Suggestions, comments, or improvements are welcome and should be addressed to:
>
> octtools@eros.berkeley.edu

These packages have been developed at UC Berkeley from 1985 to 1991 by the Berkeley CAD group. Special thanks to David Harrison, Tom Laidig, Peter Moore, Richard Rudell, Rick Spickelmeir.

3.1.11 OpenLDAP

The OpenLDAP client library is utilized by the vovserver binary to perform LDAP-based authentication. OpenLDAP software is released under the OpenLDAP Public License:

> The OpenLDAP Public License Version 2.8, 17 August 2003
>
> Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:
>
> 1. Redistributions in source form must retain copyright statements and notices,
>
> 2. Redistributions in binary form must reproduce applicable copyright statements and notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution, and
>
> 3. Redistributions must contain a verbatim copy of this document.
>
> The OpenLDAP Foundation may revise this license from time to time. Each revision is distinguished by a version number. You may use this Software under terms of this license revision or under the terms of any subsequent revision of the license.
>
> THIS SOFTWARE IS PROVIDED BY THE OPENLDAP FOUNDATION AND ITS CONTRIBUTORS ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OPENLDAP FOUNDATION, ITS CONTRIBUTORS, OR THE AUTHOR(S) OR OWNER(S) OF THE SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
>
> The names of the authors and copyright holders must not be used in advertising or otherwise to promote the sale, use or other dealing in this Software without specific, written prior permission. Title to copyright in this Software shall at all times remain with copyright holders.
>
> OpenLDAP is a registered trademark of the OpenLDAP Foundation.
>
> Copyright 1999-2003 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted.

3.1.12 OpenSSL

Runtime software makes use of OpenSSL libraries to provide secure communications to various services. The version of OpenSSL distributed with this version of Runtime software on Unix-based platforms is 1.0.2a, and on Windows, is 1.0.2e. The OpenSSL license can be viewed at https://www.openssl.org/source/license.html.

3.1.13 pgtcl-ng

Runtime software makes use of the pgtcl-ng PostgreSQL Tcl Interface library. Distribution version: 2.0.0.

3.1.14 PhantomJS

Runtime software makes use of the PhantomJS utility for parsing HTML files and working with JavaScript objects from the command line. Distribution version: 2.1.1.

3.1.15 PostgreSQL

Runtime software makes use of PostgreSQL database for its back-end data storage reporting needs. PostgreSQL database is released under the PostgreSQL License. Distribution version: 9.6.1.

> PostgreSQL is released under the PostgreSQL License, a liberal Open Source license, similar to the BSD or MIT licenses.
>
> PostgreSQL Database Management System (formerly known as Postgres, then as Postgres95)
>
> Portions Copyright (c) 1996-2010, The PostgreSQL Global Development Group
>
> Portions Copyright (c) 1994, The Regents of the University of California
>
> Permission to use, copy, modify, and distribute this software and its documentation for any purpose, without fee, and without a written agreement is hereby granted, provided that the above copyright notice and this paragraph and the following two paragraphs appear in all copies.
>
> IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
>
> THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

3.1.16 Reprise

This software includes and ships with Reprise Software Licensing Manager (RLM) binaries (v9.3rel) which are used to validate Runtime Licenses.

> Copyright © 2006-2011, Reprise Software, Inc. All Rights Reserved.
>
> Reprise License Manager, OpenUsage, and Transparent License Policy are all trademarks of Reprise Software, Inc. RLM contains software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org) Copyright © 1998-2003 The OpenSSL Project. All rights reserved.

http://www.reprisesoftware.com

3.1.17 SQLite

The SQLite database is public domain as described in http://www.sqlite.org/copyright.html. Distribution version: 3.8.3.

3.1.18 Tcl/Tk

The graphical user interface is implemented with Tcl 8.6.5/Tk 8.6.5.

Tcl/Tk includes the following copyright notice:

> This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.
>
> The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 © (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 © (1) of DFARs. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

3.1.19 tclrmq

The tclrmq library is used to interface with an externally-hosted RabbitMQ message broker server. The distribution of tclrmq included in this version of Runtime software is 1.3.8.

Copyright (c) 2017, FlightAware LLC All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of the FlightAware LLC nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3.1.20 TclVfs

The TclVfs software is used in constructing single-file distributables of Runtime software. The distribution of TclVfs included in this version of Runtime software is that which is included in the TclKit 1.8.5 distribution. TclVfs is licensed under a BSD-style license:

This software is copyrighted by the Vince Darley, and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE

BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

3.1.21 tcpkill

The utility 'tcpkill' has been included in the release, under the name 'vovtcpkill'. The utility comes with this copyright notice.

Copyright (c) 1999-2010 Dug Song <dugsong@monkey.org>, et al. All rights reserved, all wrongs reversed.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names of authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3.1.22 TkConsole

Copyright 1995-2002 Jeffrey Hobbs, jeff(a)hobbs(.)org

Release Info: v2.4, CVS v1.82 2004/11/11 17:22:13

Documentation available at: http://tkcon.sourceforge.net

3.1.23 TWAPI

On Windows, the TWAPI Tcl library is used to interface with the Windows API for various functions, such as service management. Distribution version: 3.1.17.

TWAPI includes the following copyright notice:

Copyright (c) 2003-2008, Ashok P. Nadkarni
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- The name of the copyright holder and any other contributors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES

OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3.1.24 XYNTService

The XYNTService utility is a general-purpose Windows service wrapper that allows for the creation of custom services for programs that do not have native service support. This utility is only included in the Windows distribution of Runtime software. The distribution of XYNTService included in this version of Runtime software is dated 02.22.2008 and is released under the Code Project Open License (CPOL).

3.1.25 Zlib

The 'zlib' compression library provides in-memory compression and decompression functions, including integrity checks of the uncompressed data. Distribution version: 1.2.8.

Zlib includes the following copyright notice:

> Copyright © 1995-1998 Jean-loup Gailly and Mark Adler
>
> This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.
>
> Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions: 1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required. 2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software. 3. This notice may not be removed or altered from any source distribution.
>
> Jean-loup Gailly jloup@gzip.org
> Mark Adler madler@alumni.caltech.edu
>
> The data format used by the zlib library is described by RFCs (Request for Comments) 1950 to 1952 in the files http://www.ietf.org/rfc/rfc1950.txt (zlib format), http://www.ietf.org/rfc/rfc1951.txt (deflate format) and http://www.ietf.org/rfc/rfc1952.txt (gzip format).