Lifespan Machine Software Installation Guide 1.02

Nicholas Stroustrup. 10/08/2013
nstroustrup@post.harvard.edu
Harvard Systems Biology Department.

Table of Contents

Lifespan Machine Software Installation Guide 1.01 1
Table of Contents 1
Overview of Software Components 2
Required Computing Infrastructure 3
Installation of the Worm Browser and Image Analysis Server (Windows) 4
Installation of the Image Acquisition Server (Linux) 4
Install the Scientific Linux Operating System 4
Install software packages using the package manager 5
Download the Lifespan Machine source code and manually install additional software packages 6
Compile and install the image acquisition server 7
Configure the mysql server and apache webserver httpd 7
Install and Configure the image server web interface 8
(Optional) Setting up the web interface to show captured image data 8
(Optional but recommended) Mount a Network Accessible Storage directory for long-term storage of images 9
(Optional but recommended) Disable Automatic Updates 9
Set the image server to run at startup [optional] 10
Configure the image server software 10
Run the Image Acquisition Server (Linux) 12
Naming Scanners and Generating Barcodes 13
Getting Scanners Detected 14
Security 14
A note about institutional IT departments 14
Useful Linux Commands 15

Overview of Software Components

The Lifespan Machine provides an automated means for performing lifespan experiments on *Caenorhabditis* nematodes. The lifespan machine operates continuously over multiple weeks, acquiring and interpreting images of nematode populations. Routine execution of such a task in a research setting requires an persistent imaging platform robust enough to withstand variety problems including network outages and scanner errors, and smart enough to handle such errors in a way that does not compromise data quality. The lifespan machine accomplishes this through the combined efforts of three compiled software components.

The first component is an ***image acquisition server*** runs under Linux on a dedicated PC. This PC is directly connected to scanners via USB cables. A single computer running an image acquisition server can manage up to 20 scanners. Multiple computers each running an acquisition server can be operated simultaneously, allowing very large numbers of scanners to be operated simultaneously.

The second component is an ***image processing server***, which runs under windows on any computer available—for example a personal laptop or lab desktop machine. Multiple image processing servers can be run on multiple machines in parallel, allowing rapid image processing. Image processing servers run persistently in the background, analyzing images automatically as they are collected by the image acquisition server. Image processing servers can be started and stopped at any time. When running, an image processing server works its way down a central queue of image processing tasks. Turning off an image processing server simply pauses this work; turning the server back on simply resumes analysis from where it left off. The operation of several image processing servers ensures that image processing time does not become the rate-limiting step in automated lifespan experiments. When shared computers are used to run image processing servers, the bulk of analysis tends to be accomplished on nights and weekends when more computational resources are available.

The third component is a GUI client program called *the* ***worm browser***, which runs under windows on any computer, for example your personal laptop. This program is used directly by the experimenter, allowing experiments to be scheduled, data files to be generated, and automated results to be validated.

All three software components communicate through a single, ***central MySQL database***, usually hosted on one of the image acquisition servers. This database contains metadata describing collected image data, and also coordinates the automated analysis of images. All three components require access to a ***central file server,*** where image data is written and retrieved. This is often a departmental or university file server, but lab-specific Network Accessible Storage (NAS) can be set up in cases where pre-existing IT infrastructure is lacking.

The image acquisition server has been most thoroughly tested running in Scientific Linux versions 5.9, 6.2 and 6.3. Scientific Linux is a free Redhat variant maintained by Fermilab and CERN. The image processing server and worm browser have been most thoroughly tested under Windows XP and Windows 7. A port of the image processing server and worm browser to x86 OSX should be straightforward to someone familiar with C++ and XCode—any volunteers? All three software components of the lifespan machine are written in C++. Windows binaries are provided for the image acquisition server and worm browser; the image acquisition server needs to be compiled *in situ* on its Linux server.

Users interact with the Lifespan Machine through their web browser. The latter is used to access each lab's locally hosted ***lifespan machine web interface***, which provides access to experiment information and allows image processing jobs to be scheduled. The web interface is served by a machine on the lab's local network, usually the same hardware that runs the image acquisition server. The web interface consists of a set of PHP scripts, usually served by the httpd apache web server.

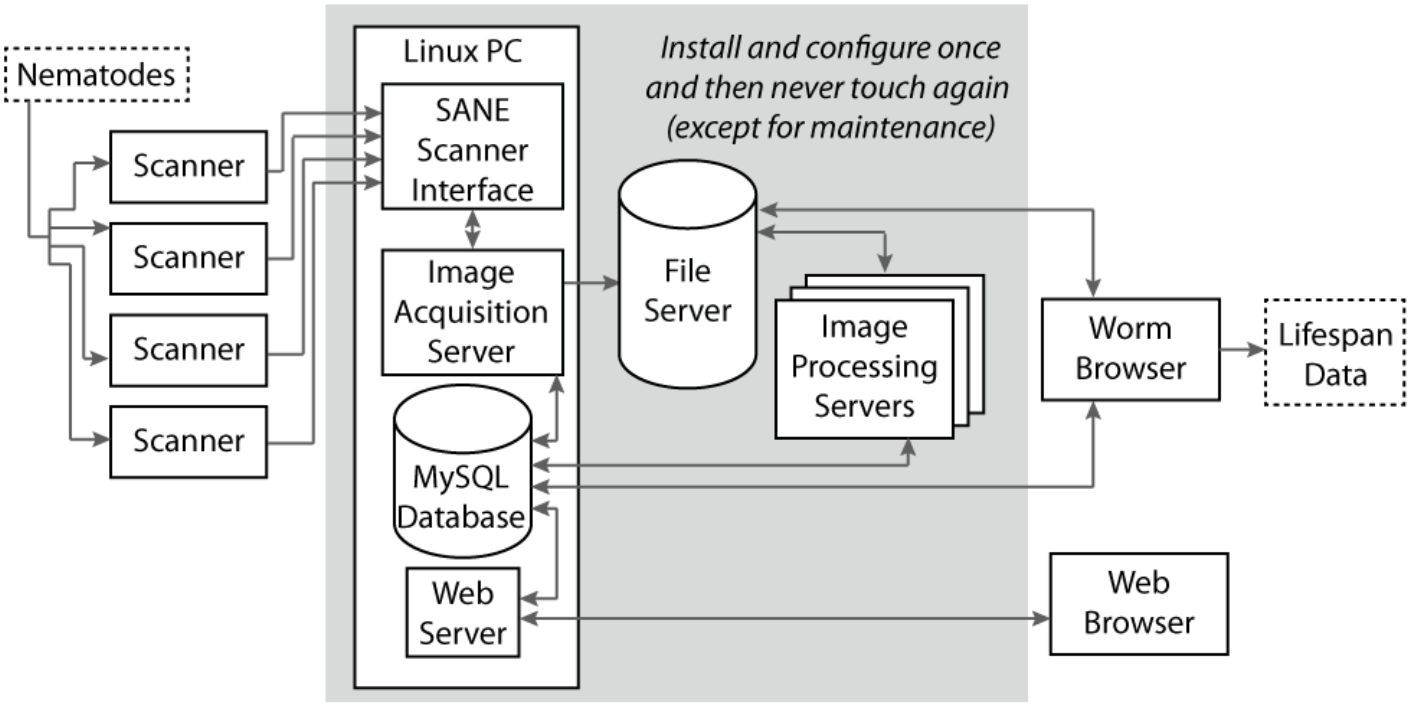


Figure 1—Lifespan Machine Schematic. The lifespan machine collects images of nematodes and interprets them to estimate the lifespan of each individual. This is accomplished through interaction of several hardware and software components, as shown. It is important to note that routine use requires an experimenter to interact only with scanners (to load animals), the worm browser (to schedule experiments and validate automated results), and a web browser (to supervise the process and schedule image processing tasks). Most users need not understand all the details of these components, but someone needs to be available to solve problems when they arise. Problems are most frequently encountered at the very beginning of an experiment, when animals are loaded into scanners and the experiments are set up. This document details the installation and configuration of all components shown above, excepting scanners whose configuration is detailed elsewhere.

Required Computing Infrastructure

- A PC running the Image Acquisition Server
  - At least two Gb of RAM
  - Any x86 processor made after 2010
  - A USB 2.0 port for direct connections to scanners
  - Scientific Linux. The 64bit version of Release 6.3 was used as the basis for these instructions. Later versions will likely work best but will likely require small alterations in the installation procedure.
  - A hard drive large enough to hold buffered images (e.g. >250Gb)
  - The server will host a network-accessible MySQL database and apache web server.
- A PC running the Worm Browser and image processing server
  - Any Windows XP, Vista, or Windows 7. Windows 8 should work OK but has not been tested. 64bit versions will allow more memory to be used, but 32bit will work.
  - Any x86 processors made after 2010. Four core Intel i7 processors can run eight image analysis server nodes simultaneously, enough to handle the output of a cluster of several scanners.

  - At least two Gb of RAM; four is better.
- A long-term image storage location accessible over the network, for example an institutional fileserver, a stand-alone NAS attached to the local ethernet, or a directory on the machine hosting the image acquisition server shared via Samba or CIFS. All captured images will be stored here, so you'll want at least one terabyte per four week lifespan performed across 10 scanners. Note that at 2013 prices, one terabyte of storage costs less than the plates required to run the experiment itself. The storage location must be accessible by the image acquisition server, the worm browser, and the image processing server.

Installation of the Worm Browser and Image Analysis Server (Windows)

The Worm Browser and Image Analysis Servers are distributed as Windows Binaries, available on the github website in the binaries directory. Currently, these files can be accessed at

https://github.com/nstroustrup/lifespan/tree/master/binaries/windows

Also included is the xvidcore.dll file (used to generate movies of worms), which should be placed in the same directory as the executable files. These tools require minimal configuration beyond editing of the configuration files ns_image_server.ini and ns_worm_browser.ini. When each program is downloaded and run for the first time, the two configuration files are created automatically, each including a description of the configuration parameters. However, the two components will not function until the can contact the mysql database running on the image acquisition server. This dictates that the acquisition server be installed first. Note: the image analysis server is designed to run many instances in parallel, to allow fast processing of images. This is set in the ns_image_server.ini option nodes_per_machine. On Intel Core Duo, i3, i5, and i7 processors, the best results are obtained when nodes_per_machine is set either to twice the number of processor cores on the machine, or twice the number of GB of memory, whichever is smaller.

The Worm Browser and Image Analysis Server can, optionally, be compiled from source using the provided Microsoft Visual Studio 2010 solution and project files. As of 2013, Microsoft Visual Studio Express is available free of charge.

Installation of the Image Acquisition Server (Linux)

This document assumes a basic proficiency in working with Linux to install packages and use the autotools compilation system.

Install the Scientific Linux Operating System

The details of how to install and maintain a Linux server are up to you. There are countless ways to configure everything, depending on personal preference and institutional policy. Refer to the section "Required Computing Infrastructure" above for details on what is used here. These installation guidelines provide a reference setup corresponding to that used by the methods paper "The *C. elegans* Lifespan Machine" (2013).

1. Download the latest version of Scientific Linux and burn it onto a DVD. Boot the computer on which you're installing linux using the DVD. The installation procedure should run automatically. The installation might offer to pre-install various packages. Nothing important needs to be installed at this time—any components not added during installation can be added later. The standard "Deskop" configuration tends to be the easiest to interact with in a laboratory context.

- Note—certain versions of Scientific Linux install with configuration problems in the GNOME windows environment. If GNOME doesn't start up correctly, tyep
  yum reinstall gnome*
  yum reinstall libgnome*

2. Make sure that the network connection is enabled (this is usually done via an icon on the Linux desktop in the top right menu bar)
3. (optional) Allow remote access to the machine via VNC
   - Select System/Preferences/Remote Desktop
   - Select "Allow others users to control your desktop"
   - Deselect "you must confirm each access to this machine"
   - Require a password
   - Select "Configure network automatically"
4. (optional) Configure your network connection to connect automatically (allowing you to log in remotely after a restart).
   - Click on the menu item System/Preferences/ Network Connections in the scientific linux menu bar
   - Select your Ethernet device and click "Edit"
   - Select the field "Connect Automatically"
5. Allow the ports through the machine's firewall,
6. following standard ports through the machine's firewall, using the System/Administration/Firewall dialog
   - Allow the following "Trusted services" ports: SSH, Samba, Secure WWW (HTTPS), and WWW ports through your firewall.
   - Allow the following "Other ports": mysql port 3306 (both tcp and udp), and (optionally) the VNC ports 5900 and 5901.
7. Add a new use to the system: ns_image_server. Create a home directory ~ns_image_server

Install software packages using the package manager

These packages must be installed in order to compile and run the image acquisition server. They can be installed simply by finding the "System/Administration/Add Remove Software" interface in the Scientific Linux Desktop Menu, and selecting them for download and installation.

- gcc-g++
- httpd (the apache web server)
- libtiff-devel
- libjpeg-devel
- freetype 2 devel
- mysql-devel
- mysql-server
- php-5
- php-mysql-5
- libusb-devel
- libusb-static
- libpng
- zlib
- git
- openjpeg-devel
- libtool
- autoconf
- automake
- libusb1-devel (NOTE: different than libusb-devel)
- (if you're using samba to mount NAS storage)
  - samba
  - samba-client
  - samba-common

IMPORTANT: Make sure the default SANE scanner package (sane-backends and sane-backends-devel) are uninstalled. Most distributions ship with SANE version 1.0.21 or less, which do not handle the Epson v700 scanner correctly. These

problems are fixed in version 1.0.23 of the SANE scanner drivers. These can be compiled from the SANE source code included in the lifespan machine git repository. Instructions for this are described in the next step.

Previous versions of the lifespan machine source code included patched versions of sane-backends-1.0.20. These can now be uninstalled and replaced by the newer 1.0.23 drivers.

Download the Lifespan Machine source code and manually install additional software packages

The lifespan machine source distribution can be downloaded from the online source repository. This can be done via accessing the github repository website and downloading a zip file containing the source code. More conveniently, the same files can be accessed using the git commandline client. This second method will make it easier to obtain the latest bug fixes as they are released.

1. Download the source code by entering the directory on your linux server /home/ns_image_server and typing the command
    - git clone https://github.com/nstroustrup/lifespan.git
2. Enter the base directory of the source code (if you used git, it will be /home/ns_image_server/lifespan)
3. Unzip external libraries and external_compile_libraries archives:
    - tar -xvjf external_compile_libraries.tar.bz2
    - tar -xvjf external _libraries.tar.bz2
4. A set of additional software packages needed for compilation are included in the directory external_compile_libraries. Three of these, listed below, need to be compiled and installed. These packages use autotools, so you should enter each directory and then type
   ./configure
   and then
   make
   and then
   make install
    - dmtx (barcode reading library)
    - sane-backends-1.0.23
    - xvid
        - This package contains compilation scripts for multiple architectures. You should use the scripts for linux, located in the directory xvidcore\build\generic
        - IMPORTANT: In the default install xvidcore, does not properly register all of its libraries. In the directory /usr/local/lib/ (or wherever you set xvidcore to install) you need to add the necessary symbolic link by typing “ln –s libxvidcore.so.4.3 libxvidcore.so” and “ln –s libxvidcore.so.4.3 libxvidcore.so.4”)
    - The packages, including NASM, FLTK, are required only for compilation under windows and need not be installed under linux. freetype should already have been installed as a package and so the code in the external_libraries can be ignored.

   Some users report file type errors, if so try using the command
   dos2unix ./*

5. The following packages are required for compilation of the lifespan machine source code under linux, but do not need to be actively installed by the user. If the contents of package external_libraries.tar.bz2 is unzipped into the default location, external_libraries/ , then this should work without further attention by the user.
    - ctmf
    - libhungarian
    - libsvm
    - tiff-3.8.2 (patched source)
    - tinyxml
    - triangle
    - wm4_bspline (a subset of the Wild Magic graphics library)

Compile and install the image acquisition server

The image acquisition server is designed to be installed using the autotools suite.

1. Enter the source code subdirectory ns_image_server
2. We have encountered several conflicts between different versions of automake and autoconf. The included configure script might run without problem, however what seems to work generally is to re-generate the configure file using your version of autotools, which can be done by executing the following commands in order:
    - aclocal
    - autoheader
    - automake --force-missing --add-missing
    - autoconf
    - ./configure
    - make install
        - If the following steps are run out of order, certain files can be scrambled. The following command, run in the source code subdirectory ns_image_server, can usually help restore the default parameters and allow the user to try again:
        - rm -r -f compile config.guess config.sub depcomp missing install-sh INSTALL autom4te.cache aclocal.m4
3. If the command make install completes without error, the executable ns_image_server should be installed in /usr/local/bin

Configure the mysql server and apache webserver httpd

1. Make sure that mysql and httpd are installed. Set mysqld and httpd to start at init level 4 and 5 (you can do this under “System/Administration/Services” in the linux desktop menu)
    - If httpd doesn’t appear to work correctly, one common problem is that the server name is set incorrectly. There is a line in the http.conf file (usually located in the directory /etc/httpd/conf/httpd.conf) that specifies the ServerName variable. It should be changed to ServerName localhost
2. Make sure you’ve set the root mysql password to something. Refer to the MysSQL documentation for details, but the command usually is mysqladmin -u root password ‘your_root_password’
3. Log into the MySQL database with the command
   mysql –u root –p
   The password required is the mysql root password you specified in step #2.
4. Create the user accounts for the image server with the command
   CREATE USER ‘image_server’@’*’ identified by ‘yourpassword’;
5. Create the central database for the image server with the command
   CREATE DATABASE image_server;

and for the local buffer
CREATE DATABASE image_server_buffer;

6. Grant access permissions
    - GRANT ALL ON image_server.* TO 'image_server'@'localhost';
    - GRANT ALL ON image_server_buffer.* TO 'image_server'@'localhost';
    - GRANT ALL ON image_server.* TO 'image_server'@'%';
    - GRANT ALL ON image_server_buffer.* TO 'image_server'@' %';
      The first two commands will allow software running on the server to access the mysql databases. The second two commands will allow other machines to connect via Ethernet to the mysql database on the server.
7. Note that sometimes mysql servers default installation sets the memory allocations ridiculously low; like 128k in some cases. In mysql.conf, the mysql configuration file, bump this up to several hundred megabytes. Check out the mysql documentation for more information
8. Install the mysql database schema as specified in the file /files/image_server_db_schema.sql This can be done by typing the command

   mysql -u root -p image_server < files/image_server_db_schema.sql
   the command should be typed from the linux shell, not from inside the mysql client.

9. Start the httpd and mysql services. The easiest way to do this in Scientific Linux is via the menu option /Adminstration/Services . The two services can be started, and also set to start automatically when the machine is restarted using the "customize" option selecting runlevel 4 and 5.

Install and Configure the image server web interface

1. Copy the contents of the webcripts directory to /var/www/html . This can be done automatically by running the script located in the source code directory (the directory created when you downloaded the code from github) ./ns_install_website.sh
2. Allow PHP to access the mysql server using the command
    - setsebool –P httpd_can_network_connect=1
3. Allow PHP to access a cifs drive using the command
    - setsebool -P httpd_use_cifs=1
4. An example web interface configuration file is included,
   /var/www/html/image_server_web/ns_image_server_website_template.ini
   This file should be edited as necessary and renamed /var/www/html/
   image_server_web/ns_image_server_website.ini
5. (Optional) Php by default hides error messages, instead displaying a blank page. To have the server output errors to the final user (allowing problems to be more easily debugged), change the following options in the php.ini configuration file, usually located in /etc/php.ini
    - display_errors = On
    - error_reporting = E_ALL & ~E_Notice

(Optional) Setting up the web interface to show captured image data

The image server web interface can be set up to display images captured by scanners, making it easy to browse through collected imagery. The challenge here is that image data is often stored on a networked drive, mounted outside the standard directory for web data: /var/www/html . The web server needs to be configured to access the images at their location.

1) Make a symbolic link in the website directory tree, linking to the path at which your images are stored (the "long_term_image_storage" option in the ns_image_server.ini).
   ln –s /my_long_term_storage_mount_point /var/www/html/long_term_storage
2) By default, the apache web server follows symbolic links. However, on some installations this may need to be set explicitly, by editing the web server configuration file, httpd.conf . This is often located in the directory /etc/httpd/conf
3) Edit the ns_image_server_website.ini file located in /var/www/html/image_server_web
   The variable $ns_image_server_storage_directory should be set to the symbolic link you created in step one, *as a path relative to /var/www/html* . That means, if you created the symbolic link as /var/www/html/long_term_storage , $ns_image_server_storage_directory should be set to "/long_term_storage".
4) The variable $ns_image_server_storage_directory_absolute should be set to the symbolic link you created in step one, as an absolute path on your system. That means, if you created the symbolic link as /var/www/html/long_term_storage , $ns_image_server_storage_directory should be set to "/var/www/html/long_term_storage".
5) It is possible that selinux might disable access by the web server to the long term storage directory . One possible solution would be to enter the command
   setsebool -P httpd_use_cifs on

(Optional but recommended) Mount a Network Accessible Storage directory for long-term storage of images

NAS directories can be mounted by adding lines to the /etc/ fstab file. On some systems, an alternative is to use the autofs package, but the author does not have direct experience with this. To use the fstab file, first, create the desired mount point in the /mnt directory. For example /mnt/fontanalab

1. The following line is an example that instructs the machine to automatically mount the Harvard Medical School file server.
   - //files.med.harvard.edu/SysBio/FontanaCluster /mnt/fontanalab cifs credentials=/root/.cifspass 0 0
2. The .cifspass file will be used to automatically provide credentials to the file server. The .cifspass is a simple text file with two lines:
   username=myusername
   password=mypassword
   - Note: as of 2/10/2010 there's a bug in redhat's cifs implementation, where the credentials file is read incorrectly. If you have problems, ensure there are no trailing newlines in the password file; for example, using the command awk 'BEGIN { printf("%s\n%s","username=u","password=p")}
3. Mount the network folders via the commands "mount /mnt/fontanalab"

(Optional but recommended) Disable Automatic Updates

Automatic updates have been observed to crash running image acquisition servers. It is important to keep the server software up to date, but this is best done manually at regular intervals, rather than automatically at random times in the middle of the night.

1. Open the interface /System/Administration /Services
2. Uncheck "yum"

Set the image server to run at startup [optional]

The image server can be configured to run automatically at startup. This is useful if, for example, there is a brief power outage in the middle of the night.

1. The rc script named /files/ns_image_server_rc_script needs to be transferred to the directory and renamed /etc/rc.d/init.d/ns_image_server . This can be done automatically by typing the command ./ns_install_startup_script.sh
2. Go to /System/Administration/ Services and set ns_image_server for runlevel 4 and 5.

Configure the image server software

The lifespan machine must be configured to interact correctly with your SQL database and your file server. A variety of important configuration options are specified in a file located on each machine running image server software. This file does not have to be written from scratch; when the image server or worm browser is launched for the first time, it will create a template ns_image_server configuration file. On Linux systems, the default location for this will be /usr/local/etc /ns_image_server.ini , though on some systems the location may be different. On windows systems, the default location is the same directory as the program executable. Image server software components, when launched for the first time, will output a message telling you where this file is located.

Once created, the configuration file can be modified with a text editor to specify a variety of important options. A list of such options and their description are included bellow.

host_name: Each instance of the image acquisition and image analysis servers needs to have a unique name to identify it. Thus, host_name should be set to a different value on every LINUX or Windows machine running the software. Use a name that you'll recognize, such as linux_server_on_my_desk, bob, or lab_desktop_1

long_term_storage_directory: All image server software must be able to access a central directory used to store images. This is often located on a NAS or institutional file server. This directory should be mounted as a path on the machine running the server. Set this parameter to the location of that directory

results_storage_directory: All image server software must be able to access a central directory used to store processed statistical data, including survival curves, descriptions of worm movement, etc. This is often located on a NAS or institutional file server. This directory should be mounted as a path of the machine running the server. Set this parameter to the location of that directory.

volatile_storage_directory: The image acquisition server and image analysis servers need to store temporary files on the local machine. Set this parameter to the location of that directory; it can be anywhere you like. For image acquisition servers, this is the local buffer for captured images pending transfer to the long term storage directory, so you should locate the directory on a drive with a couple hundred 100 GB of free space.

Access to the central SQL database

These parameters need to be set to match the account set up on your central sql database, to allow the server to log in.

central_sql_hostname: The IP address or DNS name of the computer running the central SQL server. On the linux server, this will be localhost. On other machines, this should be the ip address of the linux server.

central_sql_username: The username with which the software should log into the central SQL server

central_sql_password: The password with which the software should log into the central SQL server

central_sql_databases: The name of the database set up on the SQL server for the image server. It's possible to specify multiple independent databases, each separated by a semi-colon, but this is not needed in simple installations.

Access to the local SQL database

Image acquisition servers use a local SQL database to store metadata pending its transfer to the central SQL database. This lets acquisition servers continue to operate correctly through network disruptions, sql database crashes, etc. These parameters need to be set to match the account set up on the machine's local sql database, to allow the server to log in

local_buffer_sql_hostname: The IP address or DNS name of the computer running the local SQL buffer. This is only needed for image capture servers, and in all but exceptional cases should be set to localhost

local_buffer_sql_username: The username with which the software should log into the local SQL buffer

local_buffer_sql_database: The name of the local SQL buffer database

local_buffer_sql_password: The password with which the software should log into the local SQL buffer

Image Acquisition Server Settings

These settings control the behavior of image acquisition servers

act_as_image_capture_server: Should the server try to control attached scanners? (yes / no)

device_capture_command: the path to the SANE component scanimage, with which scans can be started

device_list_command: the path to the SANE component sane-find-scanners, with which scanners can be identified

device_barcode_coordinates: The coordinates of the barcode adhered to the surface of each scanner

simulated_device_name: For software debugging, an image acquisition server can simulate an attached device

device_names: This can be used to explicitly specify scanner names on an image acquisition server. These should be detected just fine automatically, and so in most cases this field can be left blank

Image Analysis Server Settings

These settings control the behavior of image processing servers

act_as_processing_node: Should the server run image processing jobs requested by the user via the website? (yes / no)

nodes_per_machine: A single computer can run multiple copies of the image processing server simultaneously, which allows many jobs to be processed in parallel. Set this value to the number of parallel servers you want to run on this machine. This can usually be set to the number of physical cores on the machine's processor, or the number of GB of RAM on the machine; whichever is smaller.

hide_window: On windows, specifies whether the server should start minimized. (yes / no)

compile_videos: Should the server process videos? (yes / no)

video_compiler_filename: Path to the x264 transcoder program required to generate videos. Only needed on image processing servers. If you don't have this, set compile_videos to no

video_ppt_compiler_filename: Path to the ffmpeg transcoder required to generate videos. Only needed on image processing servers. If you don't have this, set compile_videos to no

halt_on_new_software_release: Should the server shut down if a new version of the software is detected running on the cluster? (yes / no)

latest_release_path: Image acquisition servers can be set to automatically update if new versions of the software is identified as running on the cluster. This is the path name where the new software can be found.

run_autonomously: should the server automatically poll the MySQL database for new scans/jobs (yes) or should it only do this when a command is received from an external source (no). Most configurations set this to yes.

Other Settings

These settings control the behavior of image acquisition and image processing servers

verbose_debug_output: Set to true to generate verbose debug output

dispatcher_refresh_interval: How often should image acquisition servers check for pending scans? (in seconds). Also specifies how often analysis servers will check for new jobs.

mail_path: Each copy of the image server running on the cluster occasionally checks for errors occurring in other nodes, for example missed scans or low disk space. If problems are discovered, the image server can send users an email notifying them of the problem. To activate this feature, set mail_path to the POSIX mail program on the local system.

ethernet_interface: This field should be left blank if you want the server to access the network through the default network interface. If you have multiple network interfaces and want to use a specific one, specify it here.

dispatcher_port: Image acquisition and image processing servers open a TCP/IP port on the local machine through which control commands can be sent. dispatcher_port determines the specific port on which the dispatcher should listen for remote requests

server_crash_daemon_port: To provide some protection against server crashes, an image acquisition server running under linux launches a persistent second thread that checks whether the image acquisition server has crashed. In the event of a crash, the crash_daemon launches a new instance of the image acquisition server. Often, the crashed copy retains a lock its TCP/IP port, requiring that the crash daemon use a second port instead, specified here.

server_timeout_interval: How long should a server wait before giving up on a dead network connection (in seconds)

log_filename: Image acquisition and image processing servers keep a log file in the central SQL database. However, to help diagnose crashes, a text file containing the same log information is stored on the local machine. The log file is stored in the directory specified by the volatile_storage_directory option (described above), and its filename is specified by here.

Run the Image Acquisition Server (Linux)

1. Start the image server by typing ns_image_server. If the server detects a problem in its configuration, it will halt and display an error message. Correct the problem and try again, until the server starts correctly.
2. In some installations, the image server might report an error about not being able to access a log file. This might happen for two reasons:
    - The image server does not have access to the volatile storage directory set in ns_image_server.ini. This can be solved either by using chmod to set the permissions for that directory correctly, or by running the image server under a root user account. We usually choose the latter option.
    - The image server is already running. There are a variety of ways to fix this, including killing the process or typing ns_image_server stop.

3. If the command ns_image_server is entered without any arguments, the image server will try to launch. However, a variety of options can be provided that will alter the image server's behavior. These options can be seen by typing the command ns_image_server help
4. If you want the image server to detect scanners and run experiments, make sure to set the ns_image_server.ini option act_as_image_capture_server = yes
5. This image server should never be stopped during image acquisition. To shut down or restart the server, open a separate console window and enter the command ns_image_server stop or ns_image_server restart . If no scans are running, you can also press CTRL-C to send the image server a SIGINT signal. The previous command line options are preferred to CTRL-C, as the latter may terminate ongoing scans, leaving scanners in an error state, requiring they be power-cycled.

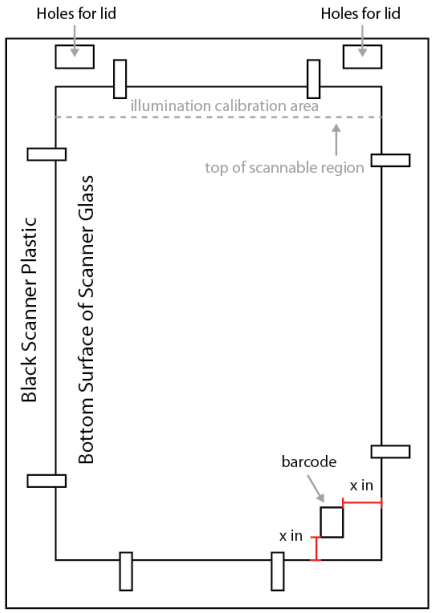
Naming Scanners and Generating Barcodes

Each scanner needs to be assigned a unique name. These names allow the server software to distinguish between scanners, and also allows the user an easy means to keep track of the locations of each plate under observation. We tend to use short, memorable names, for example cori, axel, or john . Scanners names should be three or more characters long. We affix these barcode on the inside surface of the bottom scanner glass. An example is shown to the right.


sydney

These barcodes are standard dot matrix codes , and a tool has been made to easily generate them, ns_image_server_barcodes.exe. This windows commandline binary is available from the github repository, currently accessible at https://github.com/nstroustrup/lifespan/tree/master/binaries/windows

The command ns_image_server_barcodes –c mybarcodes.tif robert linda gary
will generate a single file, mybarcodes.tif, containing the three barcodes requested. These should be printed out, cut to size, and affixed to the bottom surface of scanner glass.



The barcodes can be affixed by printing them on standard paper and using rubber cement as glue. Other types of glue may work also, but rubber cement does not dissolve printer ink, and furthermore is very easy to remove from scanner glass if spilled or smudged. Alternately, barcodes can be printed onto thin adhesive CD label paper. It is important that the paper lie flat without bubbles to the bottom of the scanner, so that it does not get caught by the scanner bar as it moves past. Many types of adhesive labels are too thick to fit in the space available, which is why "light" CD labels tend to work best. In all cases, all four corners of the barcode are securely affixed, to prevent them from catching the scanner bar and destroying the label.

The barcode should be affixed such that it is scanned correctly within the region specified by the option device_barcode_coordinates specified in the ns_image_server.ini file. The default for this is "-l 0in -t 10.3in -x 8in -y 2in", meaning that the bar code should be within a two inch region 10.3 inches from the top of the reflective scanning region, and anywhere within 8 inches from the left side of the scanner surface. A diagram of this, taken from the perspective of inside the scanner, is shown to the left.

Getting Scanners Detected

The image server interacts with scanners using the linux open source scanner driver framework SANE. There is a bug in this code concerning Epson v700 scanners, so a patched version is distributed with the lifespan machine software. SANE is extensively documented online, and handles all of the scanner-related hardware issues so that the lifespan machine doesn't have to. The basic protocol for getting the lifespan machine running is as follows

1. Plug a scanner into the computer via USB
2. Confirm that the SANE system has detected your scanner, by inspecting the results of the command sane-find-scanner . If your scanner is on this list, you're in good shape! This usually just works right out of the box, but if not, you'll need to trouble shoot. Is your scanner turned on? Plugged in? Have you set up SANE to use USB (this is the default behavior)? Are you using a supported scanner model? Check out http://www.sane-project.org. Confirm that your scanner hardware is being detected by the command sane-find-scanner . Note that gt-x900 and v700 photo refer to the same model.
3. Confirm that you've correctly attached a barcode to the inside of the scanner surface, allowing the image server software to uniquely identify it. If you don't have a barcode yet, please refer to "Naming Scanners and Generating Barcodes".
4. Restart the image server, or request a "hotplug" discovery of scanners via the web interface. The software should scan the barcode of each scanner and report back whether the scanner could be identified.

Security

It is a poor idea to have your Linux server exposed to the public internet. Criminals looking to exploit systems have significantly more time and experience than the average researcher—the best way to keep your server secure is to keep it isolated within an institutional network, behind some type of hardware firewall. If this is not possible, a second option is to configure the machine to reject access from IP addresses outside the institutional network. There are many tutorials online describing how to do this, for example www.ghacks.net/2009/03/27/configure-a-linux-firewall-with-webmin/

It is possible to password protect the lifespan machine web interface. This can be done using the facilities provided by the apache web server. Specifically, password protection can be set up by placing an .htaccess file in the base directory of the lifespan machine web interface. There are many tutorials online on how to do this, for example http://www.linux.org/article/view/-htaccess-password-protection-securing-a-folder-in-a-website

A note about institutional IT departments

IT departments in research universities work hard to maintain complex services in a constantly evolving environment. This leads to a natural conservatism in access policies for network resources. The author of this document has had very good experiences working with IT departments to establish policies conducive to high-throughput automated microscopy. However, this experience may not be universally shared.

IT departments, for example, might be unhappy with having a web server running on their local network. Networked computers certainly should be forbidden from serving to the open internet, but skeptical IT staff should be encouraged to articulate the exact risks created by a properly firewalled local Linux-based web server. Some IT departments might be hesitant to grant automated software access to their file servers. However, the lifespan machine software is much better behaved and much more predictable than many organic operators. In one instance, an IT department shut down

a smoothly functioning image server because it had “opened too many files”. In 2013, this type of arbitrarily restrictive policies have little technical basis.

A frequent suggestion raised by IT departments is that labs should set up their own network or network file server, disjoint from the campus network resources. For the IT staff, this is certainly the simplest solution, as it requires no reconfiguration of institutional resources. For lab members this also might sound like the simplest solution, as the task of building a new network might be faster and more enjoyable than advocating for change in an institutional IT policy. In practice, however, the Do It Yourself approach to network infrastructure is rarely the best solution for a research lab. Many unforeseen complications will emerge—for example desktop computers will have to be connected to both networks. The lifespan machine will always be hard to access, isolated in its own network. Per byte, custom-built networks will be significantly more costly than large institutional network. Furthermore, the creation of a private network will force researches (usually Ph.D students or post docs in the life sciences) to shoulder the exact network administrative duties that IT departments exist to provide.

Users are encouraged to engage in a productive dialog with their institution’s IT department and administration, to develop a network environment that is both safe and useful.

Useful Linux Commands

(as suggested by Deborah McEwan and Annie Conery)

cd, changes directory (cd .. goes up one level)

pwd, lists current directory

ls, lists files in folder

ls –all, see all files including hidden ones

mkdir, makes a directory

gedit or vi, document editing

su, log into root (need to type this when reboot the computer to get access to anything) will need to enter root password next

ifconfig, get the IP address etc

<control>c, force quit

Dot in a file names means it’s hidden from the standard ls command, for example .cifspass