2

2. Installation: Linux

Release Notes

Mascot Server 3.1 is compiled for Intel/AMD 64-bit Linux. Refer to the release notes for last-minute additions to documentation and the Matrix Science web site support page for patches and known issues:
https://www.matrixscience.com/mascot_support.html

Cluster Mode

If you have a licence to run Mascot on multiple processors, and plan to do so on a networked cluster of machines, please familiarise yourself with the material in Chapter 11, Cluster Mode, before proceeding with the installation.

System Requirements

Disk Space

The Mascot Server program files require 5.5 GB of Disk space, SwissProt requires 3.9 GB and PRIDE Contaminants 0.3 GB.

Memory

To get the best performance from Mascot, the database files need to be memory mapped. It is recommended that you have at least 16 GB of RAM.

Web Server

Mascot is compatible with most web servers. Appendix D, Web Server Configuration provides configuration information for Apache.

If a web server is being installed for the first time, in connection with the installation of Mascot, it is essential to verify that it is serving documents correctly before attempting to install Mascot.

Perl

Mascot includes a ‘private’ copy of Perl. If a different version of Perl is already installed or is installed later, this will not affect Mascot and the Mascot copy of Perl will not be visible to other applications.

Hyper-threading

Hyper-threading (Intel) and Symmetric multi-processing (AMD) is a hardware technique to improve the performance of multi-threaded programs. Hyper-threading does not double performance because pairs of cores share other resources, such as the on-chip cache. On some systems, a BIOS setting can be used to enable and disable hyper-threading.

Hyper-threading is detected automatically. Each CPU in the Mascot licence enables up to 4 cores to be used for searches. Hyper-threading is ignored when counting cores, so that you may see a 1 CPU licence using 8 threads on a system with a quad core processor with hyper-threading enabled.

Intel hybrid architectures

Intel desktop and workstation processors from the 12th generation onward may use a hybrid architecture. These processors have both ‘P’ (performance) and ‘E’ (efficiency) cores. Mascot can be run on both ‘P’ and ‘E’ cores, but the performance of the ‘E’ cores is poor. It is best to pin Mascot to only use ‘P’ cores. If your processor has ‘E’ cores, then after installation, search for ProcessorSet in Chapter 6, Configuration & Log Files.

File System

The file system needs to support file locking and memory mapping. We recommend installing Mascot on a local disk, as many network filesystems do not have sufficient support for file locking.

The following files will be locked/unlocked using the fcntl(,F_SETLKW,) system call: `mascot.job, getseq.job, mascot.control, mascotnode.control`. If Mascot Daemon, Mascot Distiller or any application using the task management functions in `client.pl` is used, then there will be a task_id file in each data/yyyymmdd directory that will be locked/unlocked. The following files will be memory mapped for r/w: `mascot.control, mascotnode.control`.

The location of these files can all be specified in the options section of `mascot.dat`. If you install Mascot on a network mount (e.g., NFS), the above files can be put in a directory on a local file system.

System limits

There are several types of memory limits and system limits that can stop Mascot from running. Please consult subsection Detailed Information on system limits at the end of this chapter for details.

Preparation

Clean Installation

Follow these preparation steps if you are making a clean installation.

1. If SELinux is enabled and in enforcing mode, before installing Mascot, set permissive mode in the configuration file (`SELINUX=permissive` in /etc/selinux/config) and reboot the computer.

2. Select a user account that should run ms-monitor.exe. It is simplest to run ms-monitor.exe as root or with root privileges, which is also the default if ms-monitor.exe is started from a system service. If this is not desirable, please refer to Appendix G.

3. Create a directory for the Mascot program files. In the following, this is assumed to be called *`/usr/local/mascot`*, but any path can be used. This directory should *not* be in a path mapped to a web server URL.

Then proceed with unpacking the Mascot file system.

Version upgrade

Follow these steps if you are updating an existing Mascot Server installation.

1. Locate the Mascot installation path. This is typically *`/usr/local/mascot`*, but any path could have been used.

2. Check how Mascot Monitor is started. It may have been configured as a system service (***`systemctl status mascot`***) or ms-monitor.exe may have been run from the command line. In the former case, ms-monitor.exe is usually running as root. In the latter case, it may be run as root or as some other user.

3. Check the current ownership of the Mascot directories. For example, this could be www-data:www-data. This ownership should be restored after unpacking the Mascot program files.

4. You might wish to make a backup of configuration files in the *`mascot/config`* directory. Database Manager configuration files, *`mascot.dat`* and security settings will be retained. If you are upgrading from 2.5 or later, your locally defined modifications will be retained. Other configuration files in the *`config`* directory will be overwritten.

   All results files and sequence databases will be retained (apart from SwissProt and PRIDE Contaminants, if you choose to unpack them).

5. If the Mascot data or sequence directories use symbolic links to shared storage, make a note of the target paths using:

   ```
   ls -l /usr/local/mascot
   ```

   The symbolic links must be recreated after updating Mascot.

6. Ensure that no-one will try to use Mascot during the upgrade procedure.

7. If SELinux is enabled and in enforcing mode, before updating Mascot, set permissive mode in the configuration file (`SELINUX=permissive` in /etc/selinux/config) and reboot the computer.
8. Kill the ms-monitor.exe process or stop the Mascot service.
9. Delete the *mascot/perl64* directory, if it exists.
10. Delete the *mascot/bin/ML_adapters/matrix_science* directory, if it exists.

Then proceed with unpacking the Mascot file system.

Installation

Unpack the Mascot file system

1. If you have a physical DVD containing the Mascot program files, mount this. If you downloaded an ISO image file, this can usually be mounted directly, e.g.

   ```
   sudo mkdir /mnt/mount_point
   sudo mount -o loop mascot_3_1_0_linux.iso /mnt/mount_point
   ```

   If you have downloaded the installation package as a tar.bz2 archive, extract it in a temporary directory:

   ```
   mkdir ~/mascot_tmp
   cd ~/mascot_tmp
   tar xvf /path/to/mascot_3_1_0_linux.tar.bz2
   ```

   The ISO image and the tar.bz2 archive contain the same files.

2. Decompress and unpack the file *mascot.tar.bz2.* For example (your paths may be different):

   ```
   mkdir -p /usr/local/mascot
   cd /usr/local/mascot
   tar xvpf /mnt/mount_point/mascot.tar.bz2
   ```

   Unpacking *mascot.tar.bz2* will create the directory structure illustrated in Figure 1.1. The tar **p** flag is essential to preserve permissions unless tar is executed by root.

3. If this is a **clean installation**: Also unpack the files *PRIDE_Contaminants.tar.bz2* and *swissprot.tar.bz2*.

   ```
   tar xvpf /mnt/mount_point/PRIDE_Contaminants.tar.bz2
   tar xvpf /mnt/mount_point/swissprot.tar.bz2
   ```

   If this is a **version upgrade**: Unpacking SwissProt and PRIDE_Contaminants is unnecessary if you already have up-to-date copies.

4. Change the ownership of the files to match the user and group that your web server is configured to use. The archives have been created using root:root. The required ID when Apache is installed from a RedHat RPM will be apache:apache. On Ubuntu or Debian, it will be www-data:www-data. On OpenSUSE it will be wwwrun:www.

   ```
   sudo chown apache:apache -R /usr/local/mascot/*
   ```

   If this is not acceptable, then the following directories, and the files they contain, must be made writeable by the web server process:

   logs config sessions data taxonomy unigene

5. These steps will be sufficient if Mascot Monitor is to be run as root. If you are required to run Mascot Monitor as an unprivileged user, please refer to Appendix G.

Create a symbolic link for Perl

6. If you have installed Mascot in */usr/local/mascot*, no symbolic link is required for Perl.

   Otherwise, create a symbolic link as follows, where the first path in the link is the path where Mascot has been installed (in this example, */opt/mascot*):

   ```
   sudo mkdir -p /usr/local/mascot

   sudo chmod 775 /usr/local/mascot

   sudo ln -s /opt/mascot/perl64 /usr/local/mascot/
   ```

Create URL mappings

7. If this is a **clean installation**: add the following mappings to your web server configuration, substituting your actual disk path to the new `mascot` directory.

Disk path	URL	Executable
/usr/local/mascot/cgi	/mascot/cgi	Yes
/usr/local/mascot/x-cgi	/mascot/x-cgi	Yes
/usr/local/mascot/html	/mascot	No

   Example configuration entries for Apache can be found in the file *config/apache.conf*. Notes on web server configuration can be found in Appendix D, Web Server Configuration.

   Note that some Linux distributions require you to enable CGI support in Apache:

   ```
   a2enmod cgi
   ```

   After modifying the Apache configuration in any way, Apache must be reloaded. On RHEL, CentOS, AlmaLinux and Rocky Linux:

   ```
   sudo systemctl reload httpd
   ```

   On Debian, Ubuntu and other distributions:

```
sudo systemctl reload apache2
```

If this is a **version update**: the above URL mappings should already exist in your web server configuration.

8. You may wish to restrict access to the administrative programs by setting a password or IP address restriction on `/mascot/x-cgi`. Appendix D, Web Server Configuration shows an example.

Run the installation script

9. Launch a JavaScript aware web browser, and navigate to the URL corresponding to `install.html`, e.g. http://*your.domain*/mascot/install.html

10. **Web server operation:** Follow the instructions on this web page and those that follow to perform some simple system checks and create or update the Mascot configuration file (`mascot.dat`).

    It is essential that the first page displays the message “Web server functioning correctly for documents” before trying to proceed.

11. **Perl:** Click the ‘Test Perl’ button. If you get an error message or a "File Save As..." dialog box, or if the text of a Perl script is displayed, there is a problem which must be corrected before proceeding.

    Possible reasons for this problem include:

    - Perl was not found at `/usr/local/mascot/perl64/bin/perl`, possibly because a symbolic link was not created correctly
    - The `mascot/cgi` directory is not configured for CGI execution.
    - CGI is not enabled in Apache
    - JavaScript is disabled
    - SELinux is enabled (see Appendix E, SELinux)
    - Missing system dependency

12. On some versions of CentOS and AlmaLinux, the Perl executable shipped with Mascot may fail to run when launching the Mascot install script (install1.pl). The symptom is an error while loading shared libraries, visible in the Apache error log:

    ```
    /usr/local/mascot/perl64/bin/perl: error while loading shared
    libraries: libcrypt.so.1: cannot open shared object file: No such
    file or directory: /usr/local/mascot/cgi/install1.pl, referer:
    http://localhost/mascot/install.html
    ```

    The solution is: ***`yum install libxcrypt-compat`***

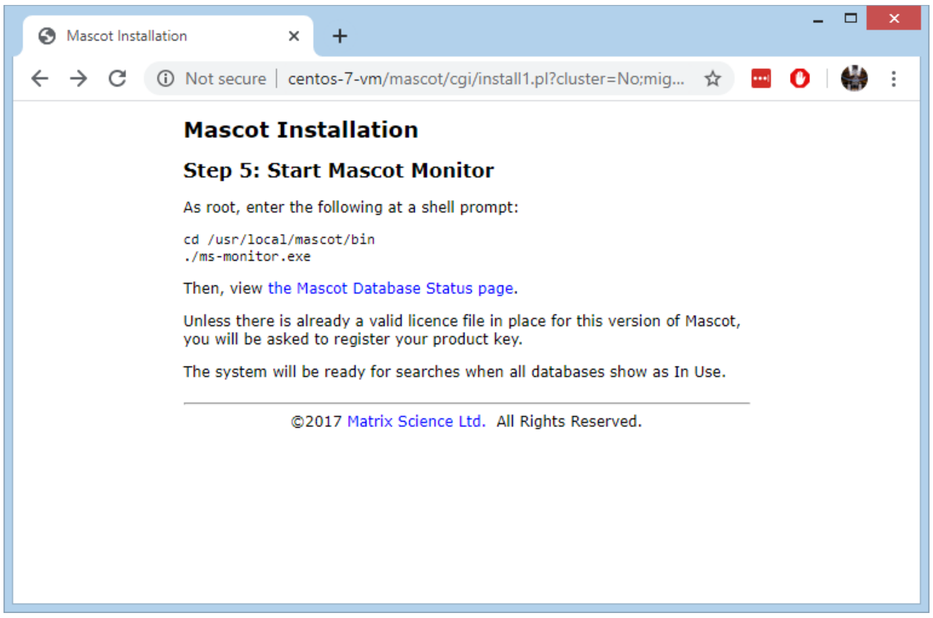
13. **Perl works correctly:** Assuming there are no problems, choose ‘Configure now’.

14. **Configuration:** Decide whether you want to configure Mascot as a single (SMP) server or as the master node of a cluster and choose ‘Configure Mascot’.

    If it is a **clean install:** a new `mascot.dat` will be created.

If this is a **version upgrade**: the main configuration file, *mascot.dat*, will be updated.

15. **Start Mascot Monitor**:



Mascot Installation

Step 5: Start Mascot Monitor

As root, enter the following at a shell prompt:

```
cd /usr/local/mascot/bin
./ms-monitor.exe
```

Then, view the Mascot Database Status page.

Unless there is already a valid licence file in place for this version of Mascot, you will be asked to register your product key.

The system will be ready for searches when all databases show as In Use.

©2017 Matrix Science Ltd. All Rights Reserved.

If you chose to configure Mascot as a single (SMP) server, you will see a screen similar to the one above, and can proceed to start Mascot Monitor. If you chose cluster mode, refer to Chapter 11, Cluster Mode for additional configuration information.

Starting Mascot Monitor

16. If this is a **clean installation**: Start Monitor at a shell prompt as root:

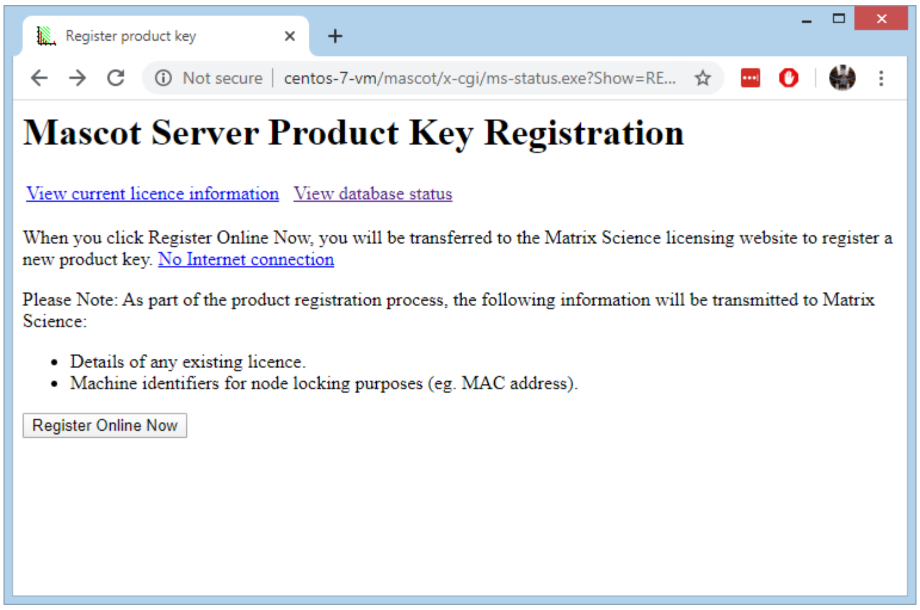
```
cd /usr/local/mascot/bin
sudo ./ms-monitor.exe
```

If this is a **version update**: Start Monitor. If it was previously configured as a system service, start the service.

These steps will be sufficient if Mascot Monitor is to be run as root. If this is not the case, please refer to Appendix G.

17. Follow the hyperlink to the Database Status page to register your product key.

Licence Registration


Register product key

Not secure | centos-7-vm/mascot/x-cgi/ms-status.exe?Show=RE...

Mascot Server Product Key Registration

View current licence information View database status

When you click Register Online Now, you will be transferred to the Matrix Science licensing website to register a new product key. No Internet connection

Please Note: As part of the product registration process, the following information will be transmitted to Matrix Science:

- Details of any existing licence.
- Machine identifiers for node locking purposes (eg. MAC address).

Register Online Now

18. A product key is required and must be registered online.

    If the Mascot server is connected to the Internet: Click Register Online Now to start the registration process.

    If the Mascot server is isolated from the Internet: Follow the link for 'No Internet connection'. A file containing registration information (registration.xml) can then be saved and copied to a system with Internet access for submission to the Matrix Science registration web site.

    After registration, the licence file can be saved directly to the Mascot Server. A copy of the licence file will also be sent by email.

    The registration form allows a second email address to be specified, in case the person installing Mascot is not the end-user. Ensure that the end-user email address is entered into the upper part of the form and the email address to which the licence file should be sent is entered into the CC email field in the lower part of the form.

19. To be recognised, the licence file must be saved to the `config/licdb` directory as a file with the extension `.lic`.

Configuring Mascot Monitor as system service

20. If this is a **clean installation**: Usually, you'll want to add *ms-monitor.exe* to the system boot process, so that it is started automatically. A suitable SysV-style init script called `mascot` can be found in the Mascot `bin` directory.

    On most Linux distributions using systemd, it is sufficient to copy the file to `/etc/init.d`. Installation instructions can be found in the script header.

    If this is a **version update**: The update procedure does not change how Mascot Monitor is started on your system.

Recreating symbolic links to data and sequence

21. This step is only required if this is a **version update** and you had previously used symbolic links to the data or sequence directories.

    a. Inspect the directory listing you made in Preparation, version upgrade, step 5. If neither data nor sequence directories were symbolic links, you can skip step 20.

    b. Otherwise, for the sake of example, suppose the links were:

    ```
    /usr/local/mascot/data -> /mnt/nfs/data
    /usr/local/mascot/sequence -> /opt/mascot/sequence
    ```

    c. If data directory was a symbolic link, recreate it now:

    ```
    cd /usr/local/mascot

    mv data data.3.1.0

    ln -s /mnt/nfs/data data

    cp -p data.3.1.0/F*.* data
    ```

    The copy step is important for updating the example searches shipped with Mascot.

    d. If sequence directory was a symbolic link, recreate it now:

    ```
    cd /usr/local/mascot

    mv sequence sequence.3.1.0

    ln -s /opt/mascot/sequence sequence
    ```

Mascot Security

22. If this is a **clean installation**: Mascot security is disabled on installation. To enable Mascot security, refer to Chapter 12, Mascot Security.

    If this is a **version update**: the update procedure does not affect the status of Mascot Security.

Keyword Indexing

23. Users of Mascot may wish to be able to search the help text by keywords or phrases. The web pages are designed to work with an indexing tool called ht://Dig. This is standard in several Linux distributions. If not installed, we recommend a stable release (3.2.0).

    Red Hat/CentOS Linux:

    ```
    yum install htdig
    ```

    Debian/Ubuntu Linux:

    ```
    aptitude install htdig
    ```

    SuSE Linux:

    ```
    yast –i htdig
    ```

openSUSE:

```
zypper install htdig
```

A few binary packages are also available at http://htdig.sourceforge.net/

Alternatively, if you have a working development system with a C++ compiler, you can download the source code from http://htdig.sourceforge.net/

Note that htdig has been removed from the default repositories in recent distributions of RedHat, CentOS and AlmaLinux. If you want keyword searching of the Mascot Server help pages, you will need to build htdig from source or install it from the Trinity Repository, as described here - https://wiki.trinitydesktop.org/RedHat_Trinity_Repository_Installation_Instructions

24. Once installed, you'll need to edit the following values in the ht://Dig configuration file, *htdig.conf*

    ```
    start_url:  http://your_host/mascot/
    ```

    Ensure common_dir and image_url_prefix have the correct values for your installation. If either setting is not defined in the configuration file, add it.

    ```
      common_dir:         /usr/local/mascot/htdig/common
      image_url_prefix:   /mascot/images
    ```

    Ensure the following extensions all appear in the bad_extensions list:

    ```
      .pl .exe .gif .jpg .pdf .msi .png
    ```

25. Map htsearch to the mascot/cgi directory.

    a. Either copy the htsearch executable to mascot/cgi, and change the ownership to match the other CGI programs, for example:

       ```
       chown apache:apache mascot/cgi/htsearch
       ```

    b. Or add a suitable ScriptAlias to the Apache configuration, for example:

    ```
    ScriptAlias /mascot/cgi/htsearch /usr/lib/cgi-bin/htsearch
    ```

    In case b, you may also need to add the following if you get 403 errors, especially if you have Mascot defined in a separate virtual host:

    ```
    <Directory /usr/lib/cgi-bin>
      Order allow,deny
      Allow from all
      Require all granted
    </Directory>
    ```

26. Finally, build an index of the Mascot web site documents:

    ```
    rundig -vv
    ```

    This may need to be run by the web server user or root, depending on how htdig has been installed and configured. Indexing will only take a minute or two. Use of the -vv flag causes verbose progress reports to be generated.

SELinux

27. Usually, Mascot Server is behind a secure firewall in a private network, and the additional security features of SELinux are not required. The simplest option is either to disable it or run in permissive mode. If SELinux must be enabled:

    Clean installation: please follow the steps in Appendix E, SELinux.

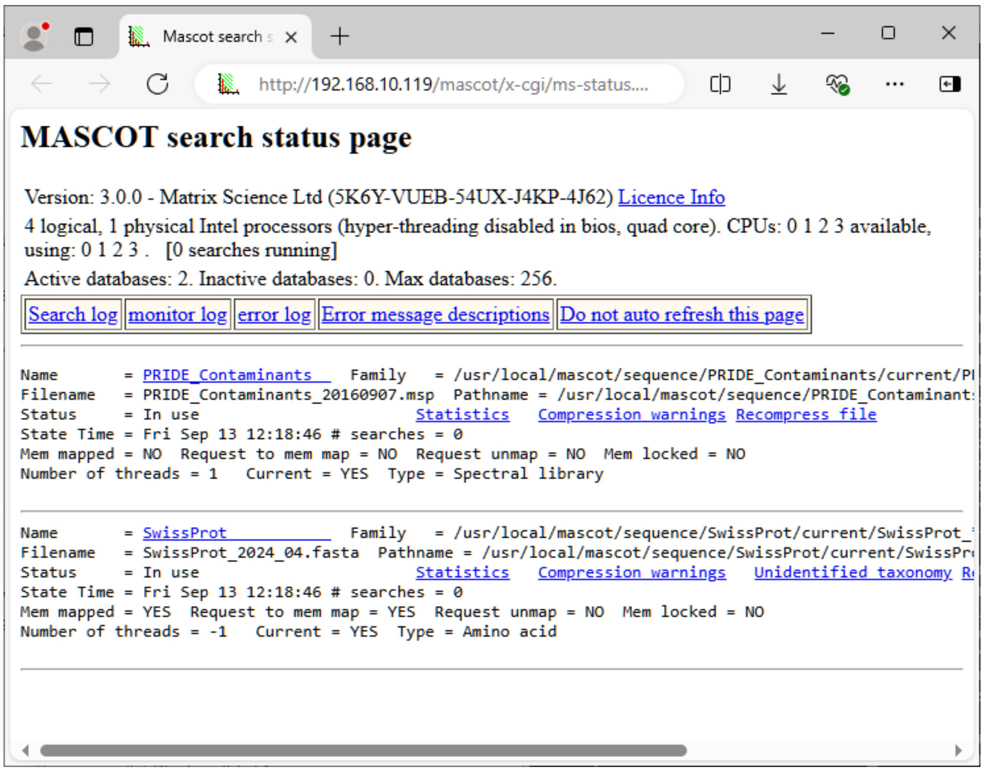
    Version update: please follow the steps in Appendix E, SELinux. The steps to set the security context and compile the security module must be repeated on version update.

Verify System Operation

A copy of the SwissProt database is included with the installation files. It is recommended that the operation of Mascot is verified and tested using this database before adding further databases or making configuration changes.

Mascot Monitor (`ms-monitor.exe`) is used to manage the swapping and memory mapping of the sequence databases used by Mascot. For Mascot to operate, `ms-monitor.exe` must be running at all times.

Once the new licence file is in place, follow the hyperlink to Database Status. You should see a display similar to the following:



MASCOT search status page

Version: 3.0.0 - Matrix Science Ltd (5K6Y-VUEB-54UX-J4KP-4J62) Licence Info

4 logical, 1 physical Intel processors (hyper-threading disabled in bios, quad core). CPUs: 0 1 2 3 available, using: 0 1 2 3 . [0 searches running]

Active databases: 2. Inactive databases: 0. Max databases: 256.

Search log | monitor log | error log | Error message descriptions | Do not auto refresh this page

```
Name       = PRIDE_Contaminants    Family   = /usr/local/mascot/sequence/PRIDE_Contaminants/current/PI
Filename   = PRIDE_Contaminants_20160907.msp  Pathname = /usr/local/mascot/sequence/PRIDE_Contaminant
Status     = In use                          Statistics   Compression warnings Recompress file
State Time = Fri Sep 13 12:18:46 # searches = 0
Mem mapped = NO  Request to mem map = NO  Request unmap = NO  Mem locked = NO
Number of threads = 1   Current = YES  Type = Spectral library

Name       = SwissProt             Family   = /usr/local/mascot/sequence/SwissProt/current/SwissProt_
Filename   = SwissProt_2024_04.fasta  Pathname = /usr/local/mascot/sequence/SwissProt/current/SwissPr
Status     = In use                          Statistics   Compression warnings   Unidentified taxonomy R
State Time = Fri Sep 13 12:18:46 # searches = 0
Mem mapped = YES  Request to mem map = YES  Request unmap = NO  Mem locked = NO
Number of threads = -1   Current = YES  Type = Amino acid
```

If an error occurs, use the links to the monitor log and the error log to investigate the cause. If all is well, you will see the following messages displayed on the status line for SwissProt:

```
Creating compressed files
Running 1st test
First test just run OK
Trying to memory map files
Just enabled memory mapping
In Use
```

You can begin exploring and using Mascot. However, do not try to run searches or view results reports until the relevant sequence database is 'In Use'.

Detailed Information on system limits

This section gives details about how Mascot reports errors, and tries to increase the limits where appropriate

How the errors are reported

If the Mascot executables report a memory error, the error can be found in the `errorlog.txt` file, including the error code returned by the operating system. For a Perl script running in CGI mode, the web server may just kill the job, and no error will be logged.

Memory limits

There are several types of memory limits that can stop Mascot from running:

1. The amount of memory that can be locked. On most systems, memory can only be locked by root.
2. Physical memory. It is obviously not possible to lock more memory than is physically available!
3. Data segment size. The amount of memory that an executable or Perl script has access to. Not normally an issue.
4. Swap space. Not normally an issue when the amount of physical RAM is 16GB or greater.
5. Stack space. Not normally an issue for executables or any of the Perl scripts.
6. Thread stack space. Not normally an issue for executables. The Perl scripts are not threaded.

Virtual address space

If memory cannot be mapped, the error M00048 "Failed to create memory map for [filename]. Error [detailed message]" will be displayed and put into the errorlog.txt file.

The amount of memory that can be locked

As well as the obvious limitation of physical memory, there is generally a limit set on the amount of memory that can be locked. Another term for locked is 'wired'.

On most systems, memory can only be locked by root. Before a “Failed to lock memory for file xxx” error is given, Mascot Monitor will try and increase the amount of RSS available by calling

```
setrlimit(RLIMIT_RSS, xxx)
```

with the current value plus the size of the file to be locked.

If the resource limit cannot be increased, then error M00114 “Error calling setrlimit(RLIMIT_RSS, [memory requested]) - error [detailed error message]” will be put into errorlog.txt

If the memory cannot be locked, then the error M00073 “Failed to lock memory for file [file name]. Error [detailed text]” will be put into the errorlog.txt file.

Physical memory

If the amount of memory locked gets close to the amount of physical memory, the system will grind to a halt! The error M00073 “Failed to lock memory for file [file name]. Error [detailed text]” will also probably be put into the errorlog.txt file.

Data segment size

This amount does not include the space used by memory-mapped files.

Insufficient data segment size will cause a Mascot search to fail with an error M00000 – “Out of memory (malloc) [number of] bytes requested”

Swap space

When all physical memory is exhausted, swap space is used. When all swap space is used, no more memory can be allocated and an error will be reported.

There is a different way of setting up swap space on each system – see system documentation.

Mascot shows free swap space for cluster nodes only.

Stack space

Has not been a problem yet.

Thread stack space

This is not normally an issue, since it is increased by all the binaries at run time to 128k.

Process limits

File size limit

Maximum file size is normally unlimited, but a limit may have been configured (e.g., *`/etc/security/limits.conf`*).

Limit on the maximum number of filehandles

Maximum number of filehandles is often limited to 1024. The Mascot monitor service keeps open six files for each active database, plus it needs to open and close the log files and certain control files as needed. When a database is being updated, compressing the new version requires up to 10 more files to be open. As soon as the open file limit is reached, the operating system will prevent the Monitor service from opening any new files, including log files.

Determining what the limits are

Most systems have two sets of limits – the current limits and the hard limits. To check the limits for the currently logged in user:

$ ulimit -a

```
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 1857
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 65536
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 1857
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

$ ulimit -aH

```
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 1857
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 65536
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) unlimited
cpu time               (seconds, -t) unlimited
max user processes              (-u) 1857
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

These values will be different for root and a normal user, and possibly different again for the owner of CGI processes and if Mascot Monitor (*`ms-monitor.exe`*) is launched as a system service by systemd. Since you may not be able to log in as the CGI user, it can be hard to find out what the real values are. If a script or binary is failing in the web browser, try running from the command line as both root and a normal user.

To check the limits of a running process, inspect the contents of *`/proc/[PID]/limits`*, where PID is the process ID.

Changing the default limits

There are different utilities / configuration files on every system. Refer to system documentation.

For example, if you launch *`ms-monitor.exe`* on the command line, then the limit is set in the current user's *`.bashrc`* or *`.profile`* by running ulimit before starting *`ms-monitor.exe`*. If you launch *`ms-monitor.exe`* as a system service, then the limit nofile (number of files) is typically set in *`/etc/security/limits.conf`*. On some systems, it may be set with `DefaultLimitNOFILE=` in *`/etc/systemd/system.conf`*.