Mascot Server Installation and Setup

MATRIX SCIENCE

Address:	Matrix Science Ltd. 64 Baker Street London W1U 7GB UK
Phone:	+44 (0)20 7486 1050
Fax:	+44 (0)20 7224 1344
Email:	info@matrixscience.com
WWW:	http://www.matrixscience.com

December 2016, Revision 2.6.0

Contents

1

1. Introduction

Mascot is a software system for protein identification by matching mass spectrometry (MS) data against FASTA format protein or nucleic acid sequence databases. This can be done in three different ways:

1. A Peptide Mass Fingerprint (PMF), in which the MS data are peptide molecular masses from the digestion of a protein by an enzyme.
2. A Sequence Query (SQ), also called a sequence tag, in which MS data are combined with amino acid sequence or composition data.
3. An MS/MS Ions Search (MIS), which uses MS/MS data from one or more peptides. MS/MS data can also be searched against spectral libraries.

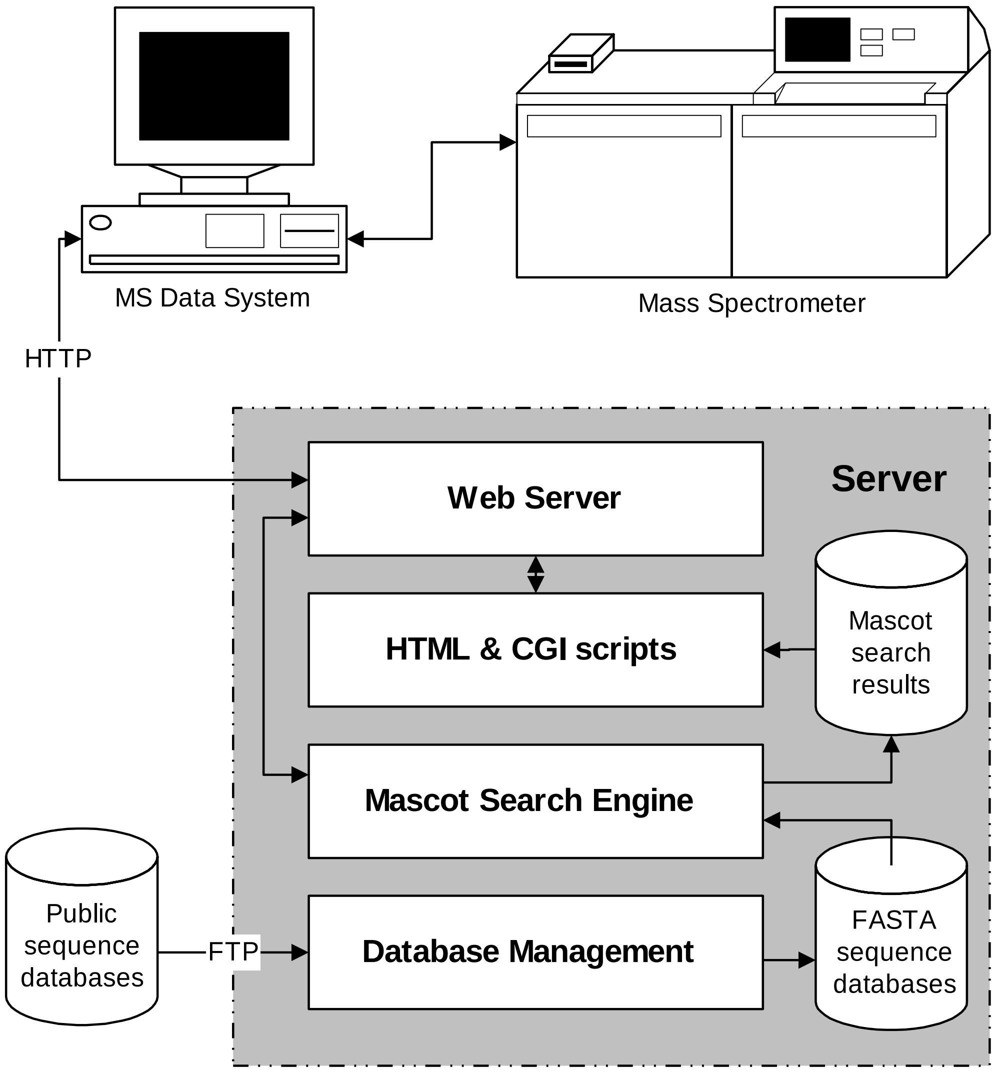
MS data are submitted to Mascot in the form of peak lists. That is, lists of centroided mass values, possibly with associated intensity values. The result of a search is a ranked list of the most closely matching proteins. Mascot uses a probability based scoring algorithm, so that it is possible to report whether a match is statistically significant. If an exact match is not present in the database, the highest scoring matches will be those entries which exhibit the greatest homology.

Overview

This manual describes how to install, configure and administer Mascot. It is not a User Guide. Mascot includes a linked collection of HTML help pages that provide guidance and application related reference material for end-users.

Mascot conforms to a client / server architecture, and the primary user interface is a JavaScript aware web browser. Searches can be submitted from web browser forms, customised for different types of searches, or from a variety of client software. Mascot Daemon is a client application, bundled with Mascot Server, for batch automation of search submission. Mascot Distiller is a powerful application, licensed separately, that can process a wide range of native file formats into peak lists, submit searches to a Mascot Server, and import the search results for examination or further processing. There are also a number of third party clients, including many mass spectrometry data systems that support search submission to Mascot.

In most cases, the Mascot search engine is executed as a CGI program. On completion of a search, it calls a Perl CGI script that reads the results file and returns an HTML report (or some other machine readable digest of the results) to the client. Links to additional CGI scripts provide more detailed views of the results.



Mascot Components

In this manual, "server" refers to the data system on which the Mascot search engine executes. The term "client" is used very loosely. It may refer to a data system attached to a mass spectrometer, or it may refer to any system at which a user interacts with the Mascot server via a web browser.

In a small laboratory, the server and client may be one and the same computer. This doesn’t affect installing or using Mascot, but it does introduce additional considerations, such as the need to adjust system priorities to ensure that the instrument control and data acquisition software is responsive to the real-time needs of both instrument and operator.

Configuration

Mascot configuration files are structured text files. Modifications can be made using a browser-based configuration editor and take effect without a system restart.

Search Engine

The Mascot search engine accepts data and parameters on STDIN in MIME format, executes a search of the specified FASTA format database, and outputs a structured text file containing the search results together with the input data and the complete set of search parameters.

The results file contains everything necessary to repeat the search at a later date, should the need arise. In the default configuration, a new directory is created on the server for each day's results files. If required, the contents of these results files can be parsed into an external database to be queried and analysed.

Monitor

Swapping databases without disrupting ongoing searches is handled by Mascot Monitor. The new database is compressed and tested by running a standard search. If errors are detected in the new database, the database exchange process is abandoned, and searches continue to use the earlier database

Assuming the test is successful, all new searches are performed against the new database, while searches that were in progress against the old database are allowed to continue. Once the final search against the old database is complete, the compressed files are deleted and the FASTA file is moved to an archive directory. If the database being exchanged is memory mapped, the mapping and un-mapping are also handled automatically.

Status

The Mascot package includes a CGI application that provides a live status display via a web browser. For each database, the Mascot job queue, the executing jobs, and the completed jobs are listed. The status lines for completed jobs contain hyperlinks to individual results reports.

Review

Review is a CGI application that provides easy access to the flat file database of search result files. Key search parameters, such as time and date, job number, user name, search type, etc. are displayed in a spreadsheet-like table. Columns can be hidden, sorted and filtered to facilitate locating a specific file or group of files. Each row includes hyperlinks, either to generate a Mascot results reports or to display the file contents as raw text.

2

2. Installation: Linux

Release Notes

Mascot 2.6 is compiled for 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: http://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 1.5 GB of Disk space, SwissProt requires 3.5 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 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.

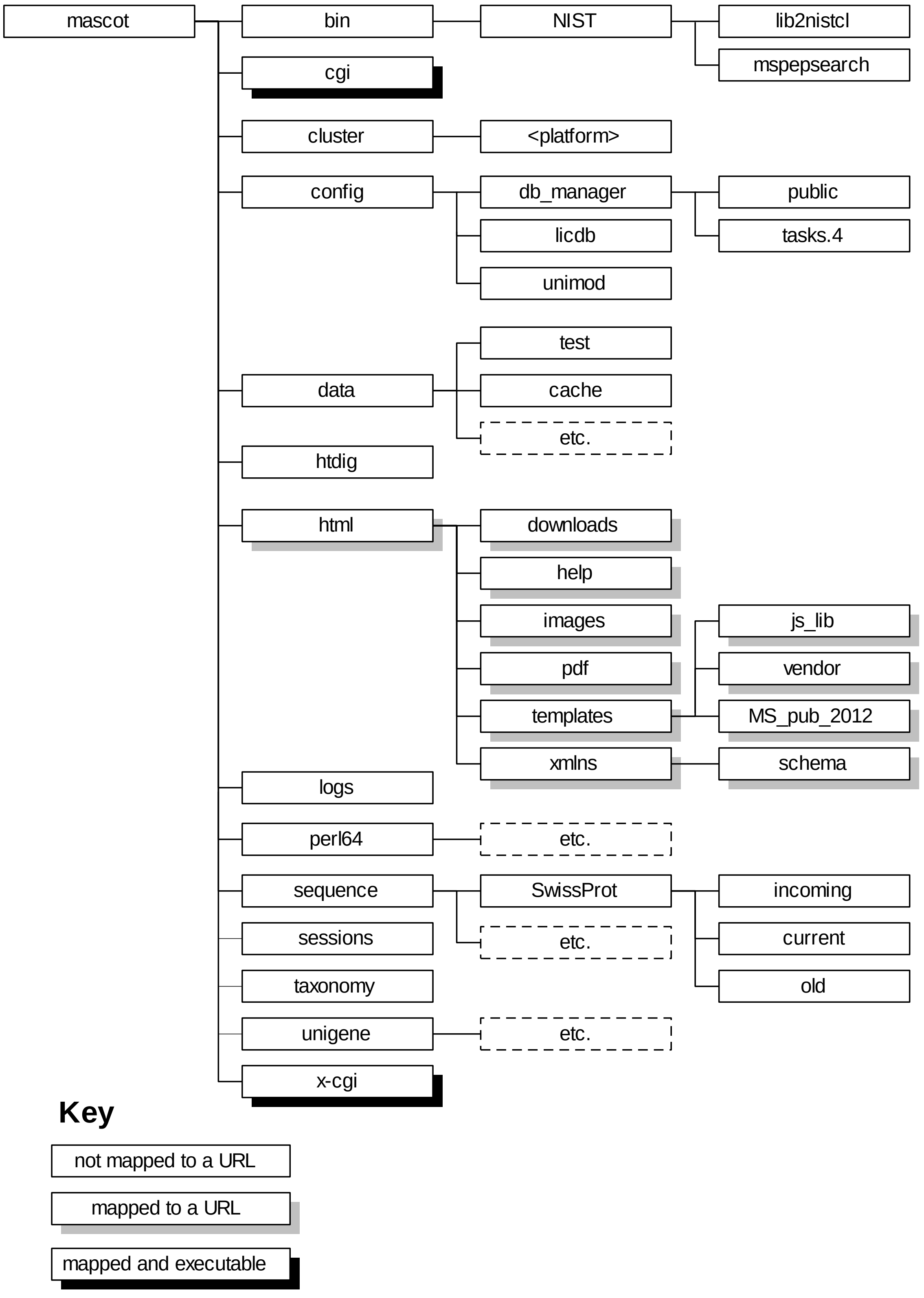


Figure 2.1 Mascot Directory Structure

Perl

Mascot will install a 'private' copy of Perl 5.18. 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.

Mascot Directory Structure

There are two directory structures to consider. One consists of the "real" paths to files on disk, the other consists of the "virtual" directories which define the web server URL's. The virtual directories are mapped to real directories. For example, the server URL

http://your.domain/mascot/home.html

might be mapped to the disk file

/usr/local/mascot/html/home.html

Any virtual directory that contains CGI executable programs (e.g. *nph-mascot.exe*) or scripts (e.g. *master_results.pl*) must have script execution enabled.

Under normal circumstances, if a directory is mapped to a URL, all of its subdirectories are also accessible as subdirectories of the URL. Figure 2.1 shows the recommended directory structure for Mascot. The root of this structure can be any convenient path.

Some of the directory paths can be changed by using a symbolic link or by modifying the configuration file, *mascot.dat*. For example, it may be desirable to have the sequence or data directories on a separate drive from the rest of the files. Care should be taken with any changes which affect a URL mapped directory or file, because this may require one or more HTML files to be edited to modify links.

In most cases, the contents of the directories can be deduced from their names:

> *bin* contains (non-CGI) executables.
>
> *cgi* contains CGI executables
>
> *cluster* contains a sub-directory for platform specific executables, for distribution to the nodes in a cluster
>
> *config* contains configuration files
>
> *data* contains Mascot results files. By default, a new sub-directory is created for each day's results files. The name of each sub-directory is that day's date in ISO format, yyyymmdd.
>
> *Htdig* contains templates for the HTML page text search facility
>
> *html* is the root directory for documents
>
> *logs* contains search and error logs, etc.
>
> *perl64* contains the 'private' copy of Perl 5.18
>
> *sequence* contains a sub-directory for each sequence database. As illustrated, for each database there are 3 sub-directories to organise the

FASTA files into new downloads (*incoming*), active databases (*current*) and the most recently replaced files (*old*).

sessions contains security session files

taxonomy contains taxonomy resources

unigene contains sub-directories for species specific UniGene indexes

x-cgi is a directory for administrative CGI executables, to which access may need to be restricted. This can be achieved using either Mascot security or web server security.

Installation

Clean Installation

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

Version upgrade

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

Kill the ms-monitor.exe process.

You might wish to make a backup of certain configuration files. Database Manager configuration files, *mascot.dat* and security settings will be retained. If you are upgrading from 2.5, 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, if you choose to unpack this).

Unpack the Mascot file system

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_2_6_0_linux.iso /mnt/mount_point
```

Decompress and unpack the files *mascot.tar.bz2, PRIDE_Contaminants.tar.bz2* and *swissprot.tar.bz2*. If this is an upgrade, and you already have an up-to-date copy of SwissProt, unpacking the swissprot archive should be skipped. For example, (your paths may be different):

```
cd /usr/local/mascot

tar xvf /mnt/mount_point/mascot.tar.bz2

tar xvf /mnt/mount_point/PRIDE_Contaminants.tar.bz2

tar xvf /mnt/mount_point/swissprot.tar.bz2
```

This will create the directory structure illustrated in Figure 2.1. Ensure that the ownership of the files matches the user ID that your web server is configured to use. The archives 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 -R apache:apache /usr/local/mascot/*
```

(If this is not acceptable, then the `logs`, `config`, `sessions`, and `data` directories, plus the file `logs/errorlog.txt` must be made writeable by the web server process).

Create a symbolic link for Perl

If you have installed Mascot in `/usr/local/mascot`, no link is required. Otherwise, create a symbolic link as follows, where the first path in the link is the path where Mascot has been installed:

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

sudo chmod 775 /usr/local/mascot

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

Create URL mappings

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

You may wish to restrict access to the administrative programs by setting a password or IP address restriction on /mascot/x-cgi.

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.

Note that many distros require you to enable CGI support in Apache:

```
a2enmod cgi
```

After modifying the Apache configuration in any way, Apache must be restarted.

Installation Script

Step 1: Web Server Operation

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

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.

Step 2: 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 corrrectly
- The `mascot/cgi` directory is not configured for CGI execution.
- CGI is not enabled in Apache
- JavaScript is disabled

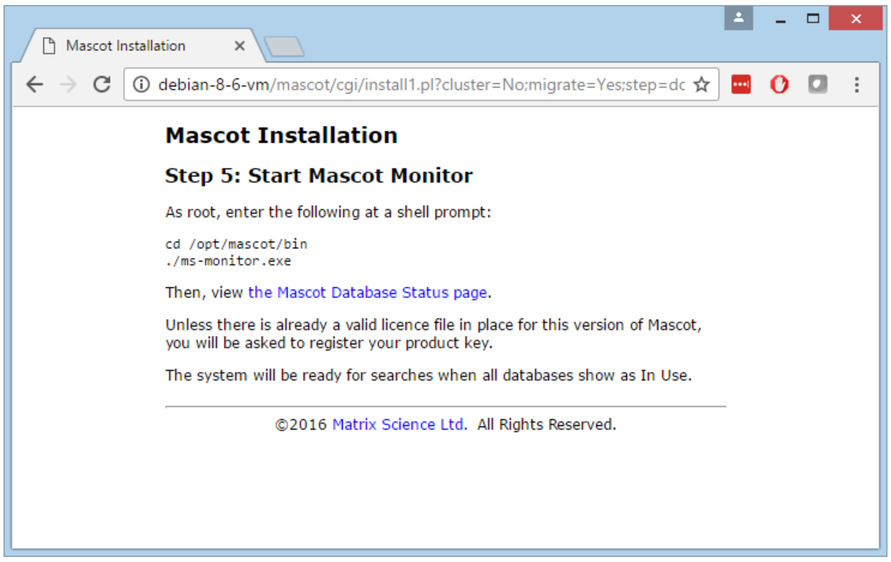
Step 3: Perl works correctly

If Perl is functioning correctly, this page confirms. Assuming there are no problems, choose 'Configure now'.

Step 4: 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 this is a version upgrade, the main configuration file, `mascot.dat`, will be updated. If it is a clean install, a new `mascot.dat` will be created.

Step 5: Start Mascot Monitor


Mascot Installation
debian-8-6-vm/mascot/cgi/install1.pl?cluster=No;migrate=Yes;step=dc

Mascot Installation

Step 5: Start Mascot Monitor

As root, enter the following at a shell prompt:

```
cd /opt/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.

©2016 Matrix Science Ltd. All Rights Reserved.

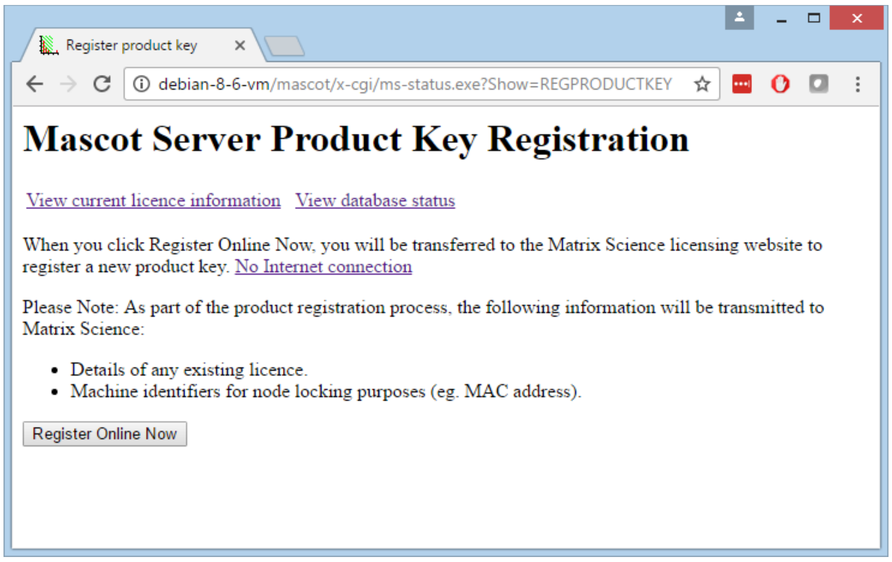
If you chose to configure Mascot as a single (SMP) server, you will see a screen similar to the one above, and can proceed to start Mascot Monitor. If you chose cluster mode, refer to Chapter 11 for additional configuration information.

Start Monitor at a shell prompt as root

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

Then follow the hyperlink to the Database Status page to register your product key.

Step 6: Licence Registration



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

A product key is required and must be registered online. The licence file can be saved directly to the Mascot Server. A copy of the licence file will also be sent by email.

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

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.

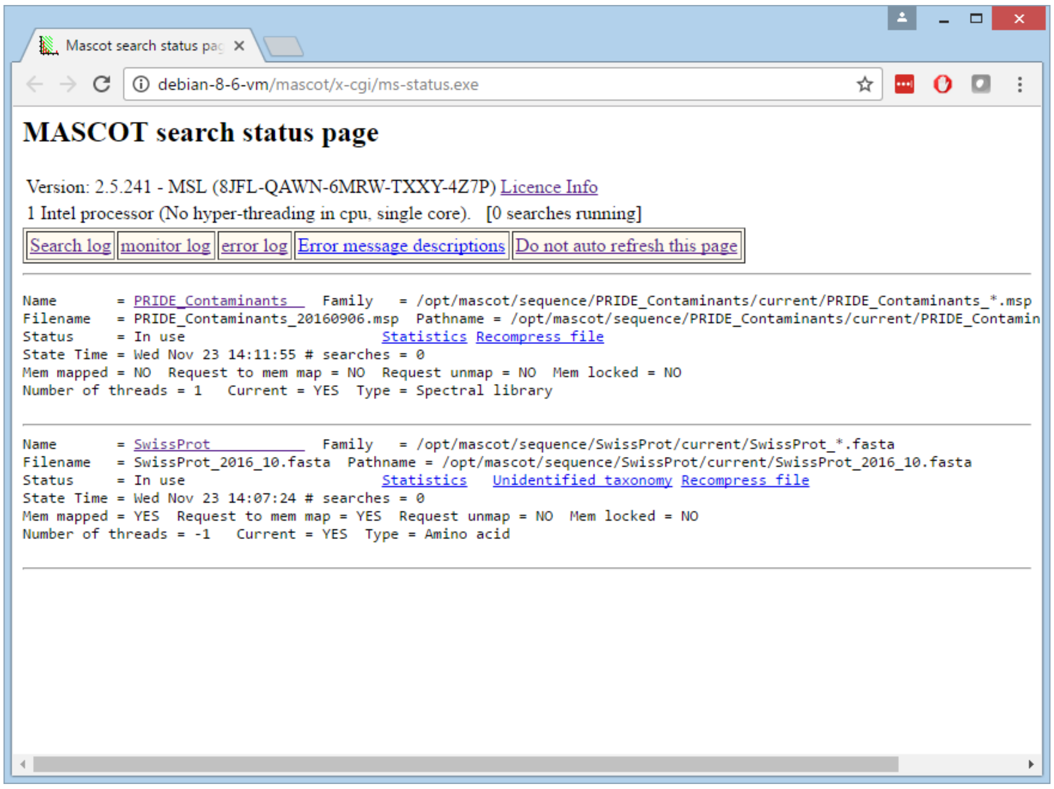
To be recognised, the licence file must be saved to the `config/licdb` directory as a file with the extension `.lic`.

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 pag

debian-8-6-vm/mascot/x-cgi/ms-status.exe

MASCOT search status page

Version: 2.5.241 - MSL (8JFL-QAWN-6MRW-TXXY-4Z7P) Licence Info

1 Intel processor (No hyper-threading in cpu, single core).   [0 searches running]

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

```
Name        = PRIDE_Contaminants     Family    = /opt/mascot/sequence/PRIDE_Contaminants/current/PRIDE_Contaminants_*.msp
Filename    = PRIDE_Contaminants_20160906.msp  Pathname = /opt/mascot/sequence/PRIDE_Contaminants/current/PRIDE_Contamin
Status      = In use                     Statistics Recompress file
State Time = Wed Nov 23 14:11:55 # 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    = /opt/mascot/sequence/SwissProt/current/SwissProt_*.fasta
Filename    = SwissProt_2016_10.fasta  Pathname = /opt/mascot/sequence/SwissProt/current/SwissProt_2016_10.fasta
Status      = In use                     Statistics   Unidentified taxonomy Recompress file
State Time = Wed Nov 23 14:07:24 # 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’.

Usually, you'll want to add *ms-monitor.exe* to the system boot process, so that it is started automatically. An example Linux init script called `mascot` can be found in the Mascot `bin` directory. Installation instructions can be found in the script header.

Security

Mascot security is disabled on installation. To enable Mascot security, refer to Chapter 12

Keyword Indexing

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 stable release (3.1.6).

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://www.htdig.org/files/binaries/

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

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
```

It is also necessary to add an alias to the Apache configuration. Add the following ScriptAlias entry, immediately before the ScriptAlias for /mascot/cgi:

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

On Red Hat/CentOS, `/usr/lib/cgi-bin` should be replaced with `/var/www/cgi-bin`

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
</Directory>
```

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.

Miscellaneous

Hyper-threading

Intel only: Hyper-threading is a technique used by Intel 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.

File System

The file system (NFS or a local file system) needs to support file locking and memory mapping. 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* so that if necessary they can be put on a local file system.

Fasta files greater than 2 GB are fully supported on ext2, ext3 and ext4 partitions.

System limits

Memory limits

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

1. Virtual address space. When files are memory mapped, the address space required can be large – the amount of physical RAM / swap space is not an issue here.
2. The amount of memory that can be locked. On most systems, memory can only be locked by root.
3. Physical memory. It is obviously not possible to lock more memory than is physically available!
4. Data segment size. The amount of memory that an executable or Perl script has access to. The default is sometimes too small to run master_results.pl, and big searches.

5. Swap space. May need to be increased for very large searches.
6. Stack space. Not normally an issue for executables or any of the perl scripts.
7. Thread stack space. Not normally an issue for executables. The perl scripts are not threaded

File size limits.

This is normally unlimited, but a limit may have been configured (e.g. /etc/security/limits.conf).

You should manually verify that your system can successfully FTP a file larger than 2 GB, as FTP doesn't necessarily report an error when it fails.

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.

Determining what the limits are.

Most systems have two sets of limits – the current limits and the hard limits.

```
$ 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. 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.

Changing the default limits

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

Detailed Information on each memory limit

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

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 frequently used 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 large master_results.pl script to fail and 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.

3

3. Installation: Windows

Release Notes

Mascot 2.6 is compiled for 64-bit Windows. Refer to the release notes for last-minute additions to documentation and the Matrix Science web site support page for patches and known issues: http://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, then please familiarise yourself with the material in Chapter 11, Cluster Mode, before proceeding with the installation.

Overview

To install or upgrade Mascot, the following steps need to be performed

1. Verify that the computer has sufficient memory and disk space
2. Verify that the computer has a suitable version of Microsoft Windows installed. Mascot requires 64-bit Windows Vista or later on Intel or AMD.
3. Virus scanning software or Microsoft Outlook should not be running during the installation
4. Install Web server software unless already installed.
5. Run setup64.exe off the Mascot program DVD

System Requirements

Disk Space

The Mascot Server program files require 1.5 GB of Disk space, SwissProt requires 3.5 GB and PRIDE Contaminants 0.3 GB.

The hard disk must be formatted for NTFS. FAT32 has a file size limit of 4GB, which would prevent the use of large sequence databases. It is advisable that NTFS file compression is *not* used for the compressed database files. There are reports that NTFS compression is not fully compatible with memory mapping. NTFS file compression can be used on the FASTA and reference files if you wish.

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.

Microsoft Windows versions

Vista SP2

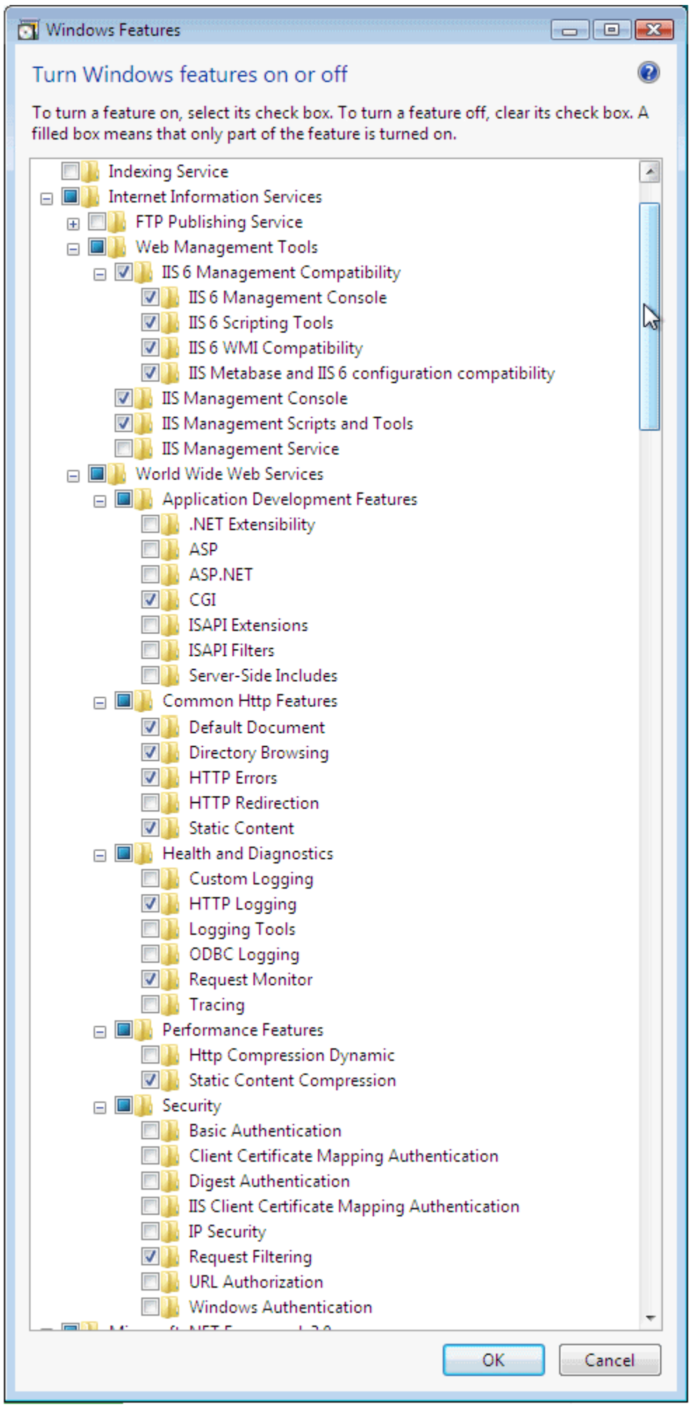
Mascot will run under all Windows Vista editions except for Starter and Home Basic.

Service pack SP2 must be installed. Check the following URL for current information:

https://support.microsoft.com/en-us/help/13858/

The Microsoft web server for Vista is IIS 7.0, which is provided as part of the standard distribution. A default installation of IIS 7.0 does not support running a CGI application such as Mascot. From the Control Panel, choose 'Programs and Features'. Choose 'Turn Windows features on or off'. Expand the node for Internet Information Services and ensure that all the checkboxes shown above are checked, in addition to any default selections. Then, choose OK.

In Vista Home Premium, the IIS 7 simultaneous request execution limit is 3. In Vista Business, Enterprise, and Ultimate Editions, the limit is 10. This will limit the number of simultaneous searches that can be run from a simple web browser form.



Windows Features

Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- [ ] Indexing Service
- [■] Internet Information Services
  - [ ] FTP Publishing Service
  - [■] Web Management Tools
    - [✓] IIS 6 Management Compatibility
      - [✓] IIS 6 Management Console
      - [✓] IIS 6 Scripting Tools
      - [✓] IIS 6 WMI Compatibility
      - [✓] IIS Metabase and IIS 6 configuration compatibility
    - [✓] IIS Management Console
    - [✓] IIS Management Scripts and Tools
    - [ ] IIS Management Service
  - [■] World Wide Web Services
    - [■] Application Development Features
      - [ ] .NET Extensibility
      - [ ] ASP
      - [ ] ASP.NET
      - [✓] CGI
      - [ ] ISAPI Extensions
      - [ ] ISAPI Filters
      - [ ] Server-Side Includes
    - [■] Common Http Features
      - [✓] Default Document
      - [✓] Directory Browsing
      - [✓] HTTP Errors
      - [ ] HTTP Redirection
      - [✓] Static Content
    - [■] Health and Diagnostics
      - [ ] Custom Logging
      - [✓] HTTP Logging
      - [ ] Logging Tools
      - [ ] ODBC Logging
      - [✓] Request Monitor
      - [ ] Tracing
    - [■] Performance Features
      - [ ] Http Compression Dynamic
      - [✓] Static Content Compression
    - [■] Security
      - [ ] Basic Authentication
      - [ ] Client Certificate Mapping Authentication
      - [ ] Digest Authentication
      - [ ] IIS Client Certificate Mapping Authentication
      - [ ] IP Security
      - [✓] Request Filtering
      - [ ] URL Authorization
      - [ ] Windows Authentication

OK Cancel

Server 2008 SP2 and Server 2008 R2

Mascot will run under all Server 2008 and 2008 R2 editions except for Core.

For Server 2008, service pack SP2 must be installed. Check the following URLs for current information:

https://support.microsoft.com/kb/968849

https://www.microsoft.com/en-gb/download/details.aspx?id=5842

The Microsoft web server for Server 2008 is IIS 7.0 and 7.5 for Server 2008 R2. From the Control Panel, choose *Turn Windows features on or off* to launch Server Manager. Select *Go to Roles*, scroll down to *Web Server (IIS)*, and choose *Add Role Services*. Then follow the configuration notes under the Windows Vista section, above

Windows 7

Mascot will run under all Windows 7 editions, but note that only Professional and Enterprise support remote desktop hosting.

It is advisable to ensure that the latest service pack has been installed. Check the following URL for current information:

https://www.microsoft.com/en-gb/download/details.aspx?id=5842

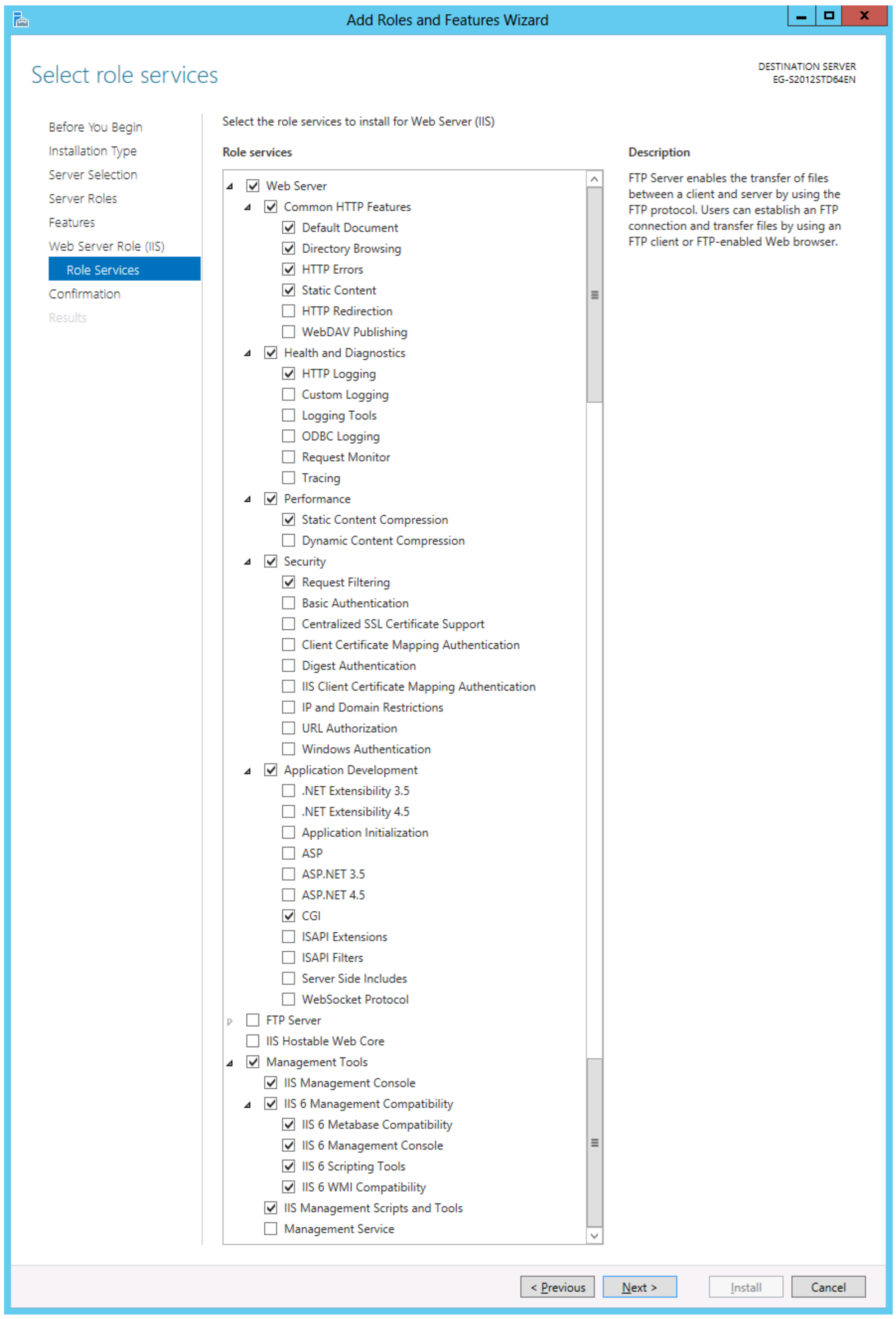
The Microsoft web server for Windows 7 is IIS 7.5. By default, this is not installed. To install IIS, from the Control Panel, choose *Programs and Features*, *Turn Windows features on or off*. Expand the node for Internet Information Services, then follow the configuration notes under the Windows Vista section, above

Server 2012 (including R2)

Mascot will run under all Server 2012 editions as long as they include the GUI. A 'Core' installation is not supported.

It is advisable to ensure that the latest updates have been installed.

The Microsoft web server is IIS 8.0 for Server 2012 and IIS 8.5 for Server 2012 R2. From the Control Panel, choose *Turn Windows features on or off* to launch Server Manager. Select *Add Roles and Features*. In the *Server Roles* page of the wizard, check *Web Server (IIS)*. In the *Role Services* page, configure as below.


Add Roles and Features Wizard

Select role services

DESTINATION SERVER
EG-S2012STD64EN

- Before You Begin
- Installation Type
- Server Selection
- Server Roles
- Features
- Web Server Role (IIS)
- **Role Services**
- Confirmation
- Results

Select the role services to install for Web Server (IIS)

Role services

- ☑ Web Server
  - ☑ Common HTTP Features
    - ☑ Default Document
    - ☑ Directory Browsing
    - ☑ HTTP Errors
    - ☑ Static Content
    - ☐ HTTP Redirection
    - ☐ WebDAV Publishing
  - ☑ Health and Diagnostics
    - ☑ HTTP Logging
    - ☐ Custom Logging
    - ☐ Logging Tools
    - ☐ ODBC Logging
    - ☐ Request Monitor
    - ☐ Tracing
  - ☑ Performance
    - ☑ Static Content Compression
    - ☐ Dynamic Content Compression
  - ☑ Security
    - ☑ Request Filtering
    - ☐ Basic Authentication
    - ☐ Centralized SSL Certificate Support
    - ☐ Client Certificate Mapping Authentication
    - ☐ Digest Authentication
    - ☐ IIS Client Certificate Mapping Authentication
    - ☐ IP and Domain Restrictions
    - ☐ URL Authorization
    - ☐ Windows Authentication
  - ☑ Application Development
    - ☐ .NET Extensibility 3.5
    - ☐ .NET Extensibility 4.5
    - ☐ Application Initialization
    - ☐ ASP
    - ☐ ASP.NET 3.5
    - ☐ ASP.NET 4.5
    - ☑ CGI
    - ☐ ISAPI Extensions
    - ☐ ISAPI Filters
    - ☐ Server Side Includes
    - ☐ WebSocket Protocol
- ☐ FTP Server
- ☐ IIS Hostable Web Core
- ☑ Management Tools
  - ☑ IIS Management Console
  - ☑ IIS 6 Management Compatibility
    - ☑ IIS 6 Metabase Compatibility
    - ☑ IIS 6 Management Console
    - ☑ IIS 6 Scripting Tools
    - ☑ IIS 6 WMI Compatibility
  - ☑ IIS Management Scripts and Tools
  - ☐ Management Service

Description

FTP Server enables the transfer of files between a client and server by using the FTP protocol. Users can establish an FTP connection and transfer files by using an FTP client or FTP-enabled Web browser.

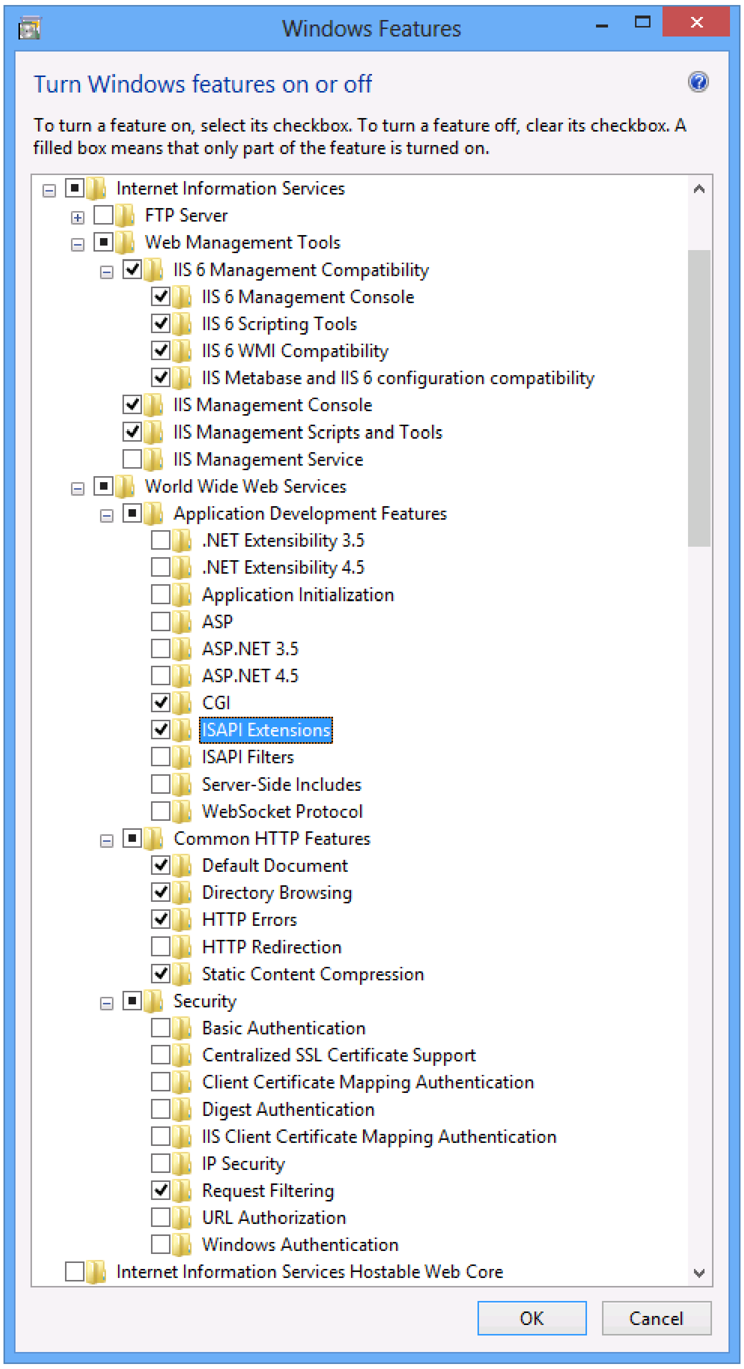
Windows 8

Mascot will run under all Windows 8 and 8.1 editions except RT, but note that only Professional and Enterprise support remote desktop hosting.

It is advisable to update to 8.1 and ensure that the latest updates have been installed.

https://support.microsoft.com/help/15288

The Microsoft web server is IIS 8.0 for Windows 8 and IIS 8.5 for Windows 8.1. By default, this is not installed. To install IIS, from the Control Panel, choose *Programs and Features*, *Turn Windows features on or off*. Expand the node for Internet Information Services, then configure as shown below.



Windows Features

Turn Windows features on or off

To turn a feature on, select its checkbox. To turn a feature off, clear its checkbox. A filled box means that only part of the feature is turned on.

- ■ Internet Information Services
  - ☐ FTP Server
  - ■ Web Management Tools
    - ✓ IIS 6 Management Compatibility
      - ✓ IIS 6 Management Console
      - ✓ IIS 6 Scripting Tools
      - ✓ IIS 6 WMI Compatibility
      - ✓ IIS Metabase and IIS 6 configuration compatibility
    - ✓ IIS Management Console
    - ✓ IIS Management Scripts and Tools
    - ☐ IIS Management Service
  - ■ World Wide Web Services
    - ■ Application Development Features
      - ☐ .NET Extensibility 3.5
      - ☐ .NET Extensibility 4.5
      - ☐ Application Initialization
      - ☐ ASP
      - ☐ ASP.NET 3.5
      - ☐ ASP.NET 4.5
      - ✓ CGI
      - ✓ ISAPI Extensions
      - ☐ ISAPI Filters
      - ☐ Server-Side Includes
      - ☐ WebSocket Protocol
    - ■ Common HTTP Features
      - ✓ Default Document
      - ✓ Directory Browsing
      - ✓ HTTP Errors
      - ☐ HTTP Redirection
      - ✓ Static Content Compression
    - ■ Security
      - ☐ Basic Authentication
      - ☐ Centralized SSL Certificate Support
      - ☐ Client Certificate Mapping Authentication
      - ☐ Digest Authentication
      - ☐ IIS Client Certificate Mapping Authentication
      - ☐ IP Security
      - ✓ Request Filtering
      - ☐ URL Authorization
      - ☐ Windows Authentication
- ☐ Internet Information Services Hostable Web Core

Server 2016

Mascot will run under all Server 2016 editions as long as they include the GUI or ‘Desktop Experience’. A non-GUI or ‘Core’ installation is not supported.

It is advisable to ensure that the latest updates have been installed.

The Microsoft web server is IIS 10. From the Control Panel, choose *Turn Windows features on or off* to launch Server Manager. Select *Add Roles and Features*. In the *Server Roles* page of the wizard, check *Web Server (IIS)*. In the *Role Services* page, configure as in the Windows 8 screen shot.

Windows 10

Mascot will run under all Windows 10 editions, but note that the Home edition does not support remote desktop hosting.

It is advisable to ensure that the latest updates have been installed.

The Microsoft web server is IIS 10. By default, this is not installed. To install IIS, from the Control Panel, choose *Programs and Features*, *Turn Windows features on or off*. Expand the node for Internet Information Services, and ensure all the options shown above, in the Windows 8 screen shot, are selected.

Web Server

Mascot for Windows is tested with IIS and Apache.

The Mascot installation has been fully automated for Microsoft Internet Information Server 7.0 and later. A good starting point for IIS support information is http://www.iis.net/

IMPORTANT: You **must** configure IIS as illustrated above **before** proceeding with the installation. Otherwise, the Mascot installation is likely to fail.

If IIS is configured as a secure server (SSL/TLS), you must change it temporarily to non-secure mode (http: on port 80). Once the installation is complete, you can change back to secure mode.

If you wish to use Apache as your web server, you will need to perform some manual configuration, as described in Appendix D.

Perl

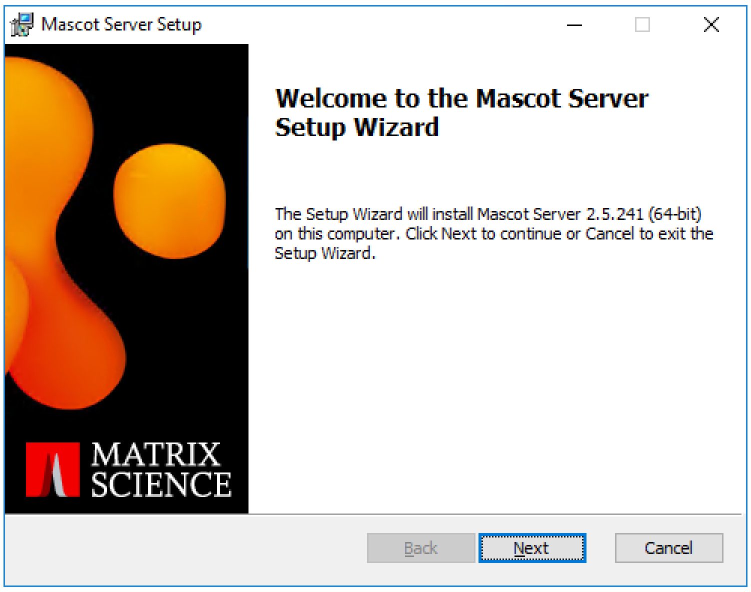
Mascot includes a ‘private’ copy of ActivePerl 5.16.3 (build 1603) from ActiveState Corporation. If a different version of Perl is already installed or is installed later, this will not affect Mascot and the Mascot copy of Perl will not be visible to other applications.

Mascot Installation

If you have downloaded the installer as a self-extracting executable, copy the file to a temporary location and double click to unpack. This will create a folder containing exactly the same files as on the Mascot Server DVD. In either case, double click on `setup64.exe`.

Before the installation of Mascot begins, required Microsoft Visual C++ libraries will be installed.

The following window will be displayed:


Mascot Server Setup

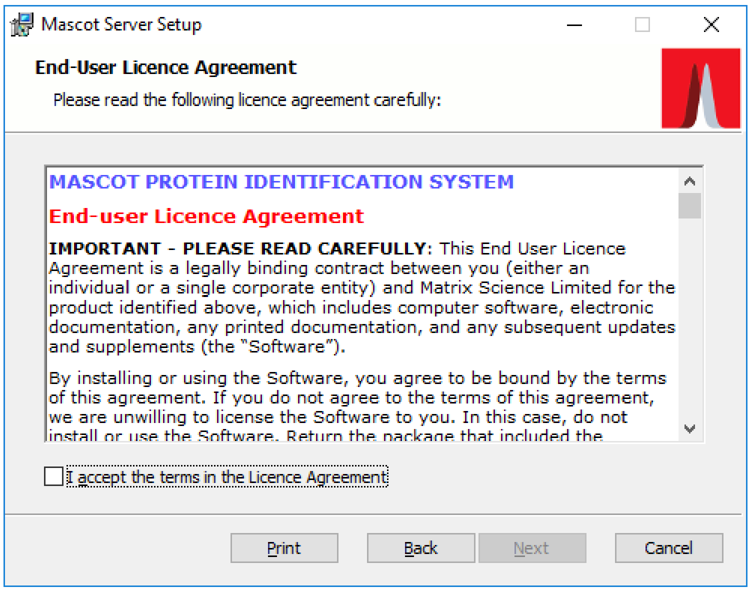
Welcome to the Mascot Server Setup Wizard

The Setup Wizard will install Mascot Server 2.5.241 (64-bit) on this computer. Click Next to continue or Cancel to exit the Setup Wizard.

If the installation cannot proceed, a message box will be displayed. Typical problems include:

> **You do not have Administrator privileges:** Log out and log in as a user with local Administrator rights
>
> **Unsupported Windows platform:** Refer to the system requirements at the beginning of this Chapter

Any problem(s) must be fixed before the installer will proceed. Pressing Next displays the Mascot End-User Licence Agreement:


Mascot Server Setup

End-User Licence Agreement

Please read the following licence agreement carefully:

MASCOT PROTEIN IDENTIFICATION SYSTEM

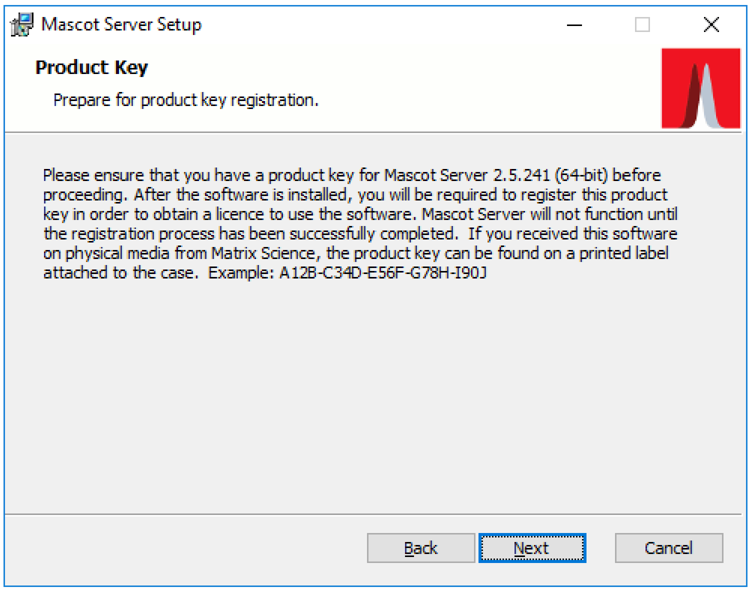
End-user Licence Agreement

IMPORTANT - PLEASE READ CAREFULLY: This End User Licence Agreement is a legally binding contract between you (either an individual or a single corporate entity) and Matrix Science Limited for the product identified above, which includes computer software, electronic documentation, any printed documentation, and any subsequent updates and supplements (the "Software").

By installing or using the Software, you agree to be bound by the terms of this agreement. If you do not agree to the terms of this agreement, we are unwilling to license the Software to you. In this case, do not install or use the Software. Return the package that included the

☐ I accept the terms in the Licence Agreement

If you do not consent, you cannot proceed with the installation.

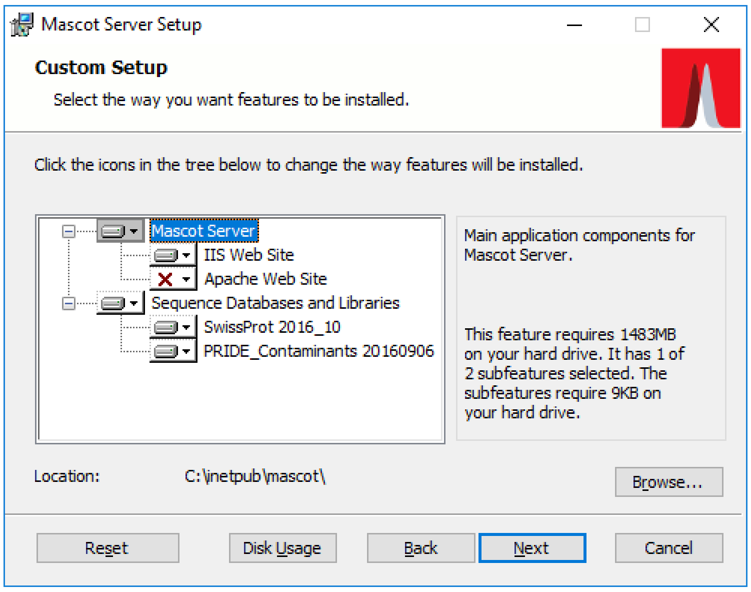


Product Key

Prepare for product key registration.

Please ensure that you have a product key for Mascot Server 2.5.241 (64-bit) before proceeding. After the software is installed, you will be required to register this product key in order to obtain a licence to use the software. Mascot Server will not function until the registration process has been successfully completed. If you received this software on physical media from Matrix Science, the product key can be found on a printed label attached to the case. Example: A12B-C34D-E56F-G78H-I90J

This is a reminder that you will need to register a product key to create a licence file. This product key may be printed on a sticker on the CD case or it may have been sent by email. If you cannot locate your product key, contact support@matrixscience.com for assistance. The next screen allows you to choose which components will be installed:



Custom Setup

Select the way you want features to be installed.

Click the icons in the tree below to change the way features will be installed.

- Mascot Server
  - IIS Web Site
  - Apache Web Site
- Sequence Databases and Libraries
  - SwissProt 2016_10
  - PRIDE_Contaminants 20160906

Main application components for Mascot Server.

This feature requires 1483MB on your hard drive. It has 1 of 2 subfeatures selected. The subfeatures require 9KB on your hard drive.

Location: C:\inetpub\mascot\

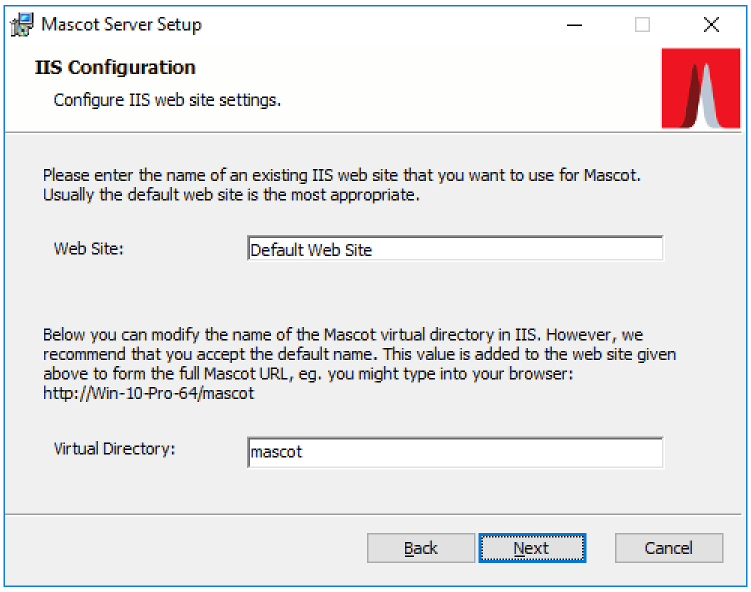
If IIS is installed and functional, the default selections will be as shown above, with IIS being configured automatically. If you don't have IIS installed, the Apache option will be selected instead. A test for whether Apache or some other web server is actually installed comes later.

You can de-select the Swiss-Prot Fasta database and PRIDE Contaminants spectral library, but if this is a clean install, you are advised not to do so. It is better to proceed with a full installation, so that correct installation of Mascot can be verified. If you don't want SwissProt to be available, you can easily remove it later.

The default location for the installation is `\inetpub\mascot` on the drive with most free space with the sequence databases in `\inetpub\mascot\sequence`.

You can change one or both of these by selecting the component then choosing Browse. If there is insufficient disk space on the selected drive(s), the installation will not be able to continue.

The next step depends on whether IIS or Apache was selected as the web server. For IIS, in most cases, you should leave the web site field set to 'Default Web Site'. If you have multiple web sites defined and want to use a non-default site for Mascot, enter the name as displayed in IIS Manager.



Please enter the name of an existing IIS web site that you want to use for Mascot. Usually the default web site is the most appropriate.

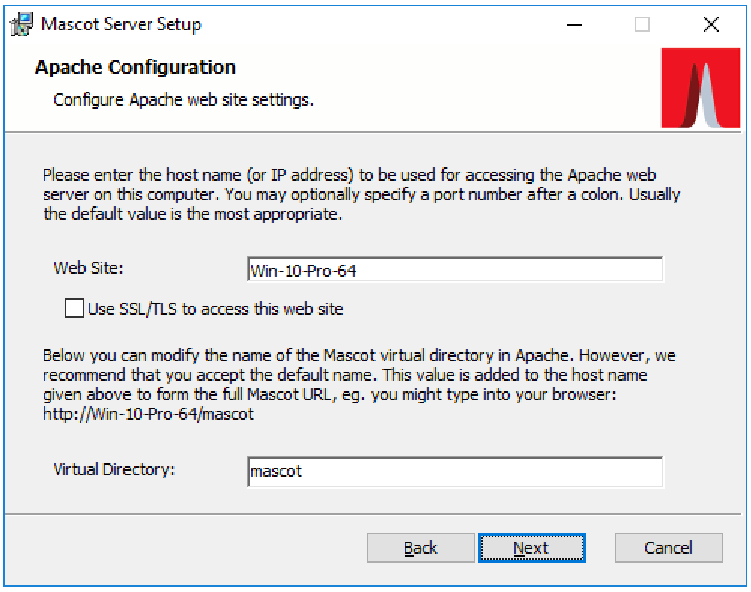
Web Site: Default Web Site

Below you can modify the name of the Mascot virtual directory in IIS. However, we recommend that you accept the default name. This value is added to the web site given above to form the full Mascot URL, eg. you might type into your browser: http://Win-10-Pro-64/mascot

Virtual Directory: mascot

For Apache, or any other web server, you need to confirm the local web server hostname and port. Do not enter localhost in the web site field if you wish to access your Mascot server from other computers on your LAN. If there are DNS problems, so that a hostname is not recognised across the LAN, then enter an IP address.

The default ports are 80 for http and 443 for https. The installer will test that the web server responds using the specified hostname and port number. If you have configured your Apache web server as a secure server (https), check the box for 'Use SSL/TLS to access this web site'.



Please enter the host name (or IP address) to be used for accessing the Apache web server on this computer. You may optionally specify a port number after a colon. Usually the default value is the most appropriate.

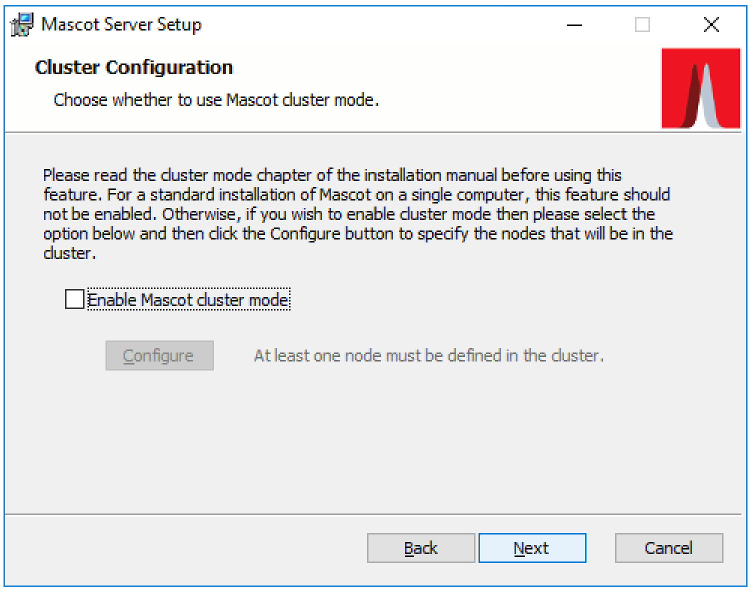
Web Site: Win-10-Pro-64

☐ Use SSL/TLS to access this web site

Below you can modify the name of the Mascot virtual directory in Apache. However, we recommend that you accept the default name. This value is added to the host name given above to form the full Mascot URL, eg. you might type into your browser: http://Win-10-Pro-64/mascot

Virtual Directory: mascot

The virtual directory name can be changed, but remember that users are more likely to guess the correct URL if you stick with mascot. Also, some third party software may incorrectly assume the directory name is always mascot.


Mascot Server Setup

Cluster Configuration

Choose whether to use Mascot cluster mode.

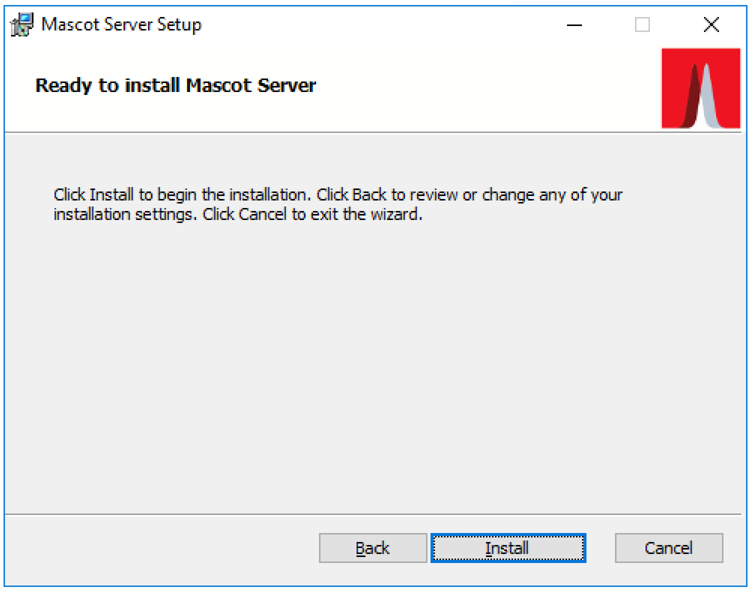
Please read the cluster mode chapter of the installation manual before using this feature. For a standard installation of Mascot on a single computer, this feature should not be enabled. Otherwise, if you wish to enable cluster mode then please select the option below and then click the Configure button to specify the nodes that will be in the cluster.

☐ Enable Mascot cluster mode

At least one node must be defined in the cluster.

If you have a multi-CPU licence, you can configure Mascot for execution on a networked cluster. If you intend to do this, refer to Chapter 11 for further details before proceeding. If you are installing Mascot on a single multiprocessor server, leave the Enable cluster mode checkbox clear.

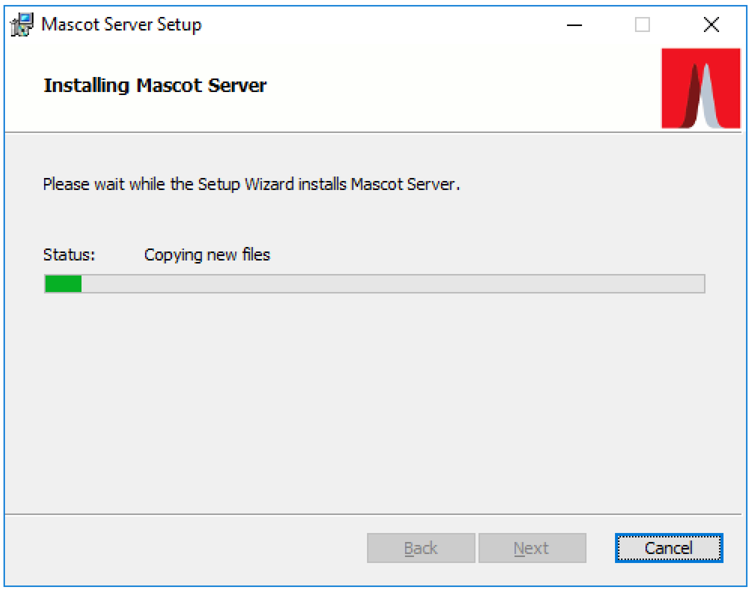
The next step is your last opportunity to cancel the installation!


Mascot Server Setup

Ready to install Mascot Server

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Copying the program files takes only a few minutes

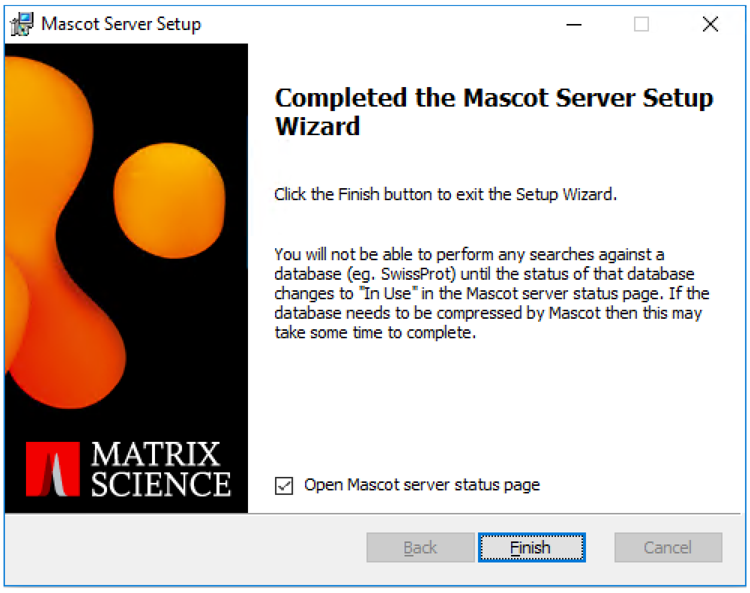

Mascot Server Setup

Installing Mascot Server

Please wait while the Setup Wizard installs Mascot Server.

Status: Copying new files

Unpacking the SwissProt files takes time, and a command window will be displayed at this point. Please be patient and don't try to close the command Window.


Mascot Server Setup

Completed the Mascot Server Setup Wizard

Click the Finish button to exit the Setup Wizard.

You will not be able to perform any searches against a database (eg. SwissProt) until the status of that database changes to "In Use" in the Mascot server status page. If the database needs to be compressed by Mascot then this may take some time to complete.

☑ Open Mascot server status page

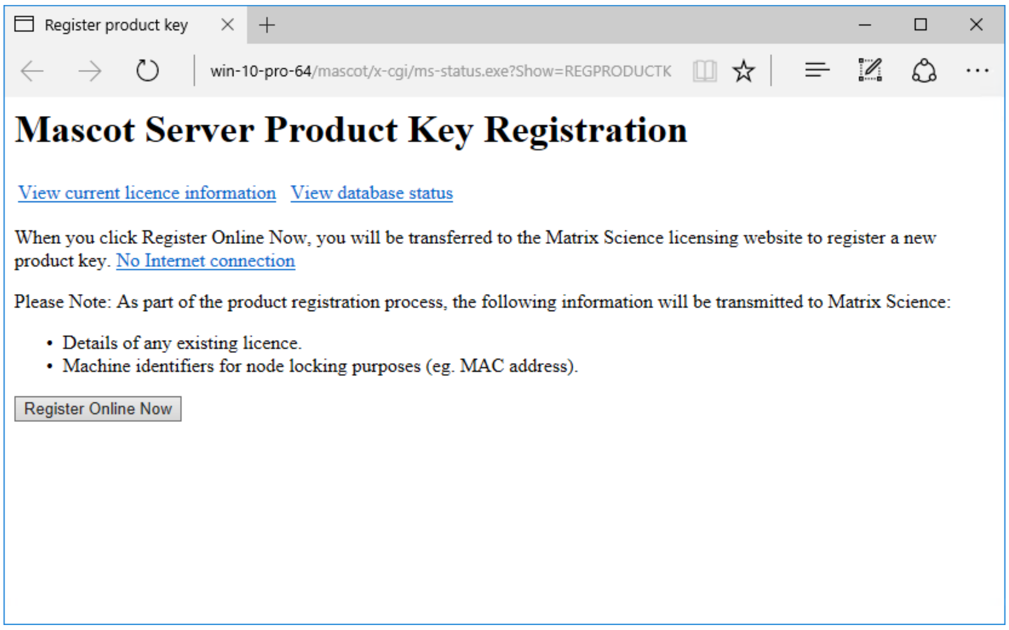
If you are using Apache, model entries for the Apache configuration file can be found in `httpd.conf` in the Mascot `config` directory. Further information on web server configuration can be found in Appendix D.

Installation is finished, but don't clear the checkbox!

Licence Registration

(If you cleared the checkbox at the end of the installation wizard, from the Windows Start menu, choose Programs; Mascot; Admin; Database Status. Then choose Register new product key.)

The following screen will be displayed in your default web browser



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

A product key is required and must be registered online. The licence file can be saved directly to the Mascot Server. A copy of the licence file will also be sent by email.

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

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.

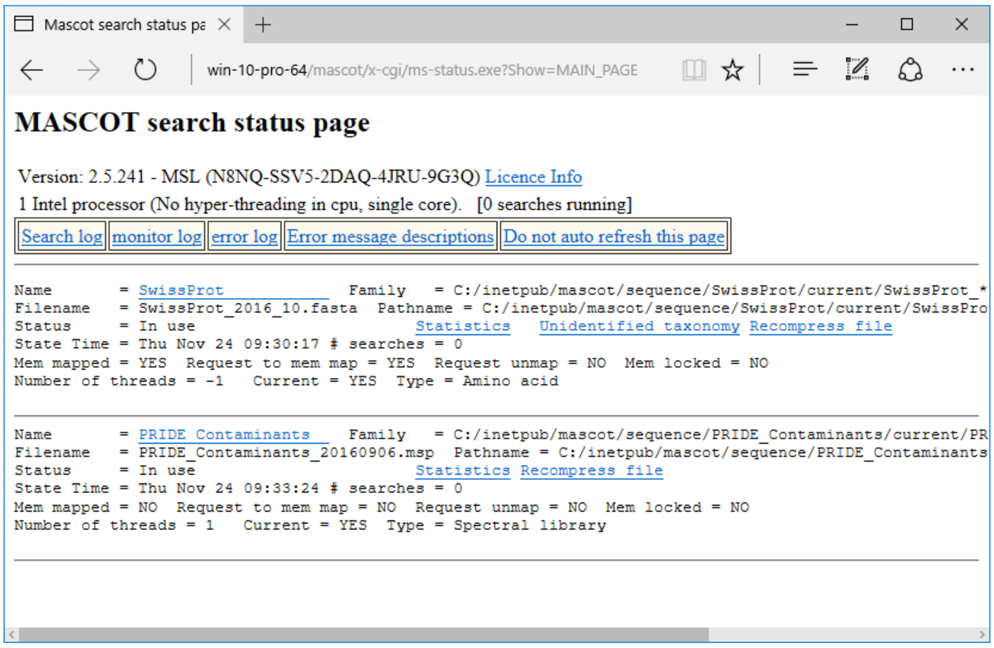
To be recognised, the licence file must be saved to the *config\licdb* directory as a file with the extension *.lic*.

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: 2.5.241 - MSL (N8NQ-SSV5-2DAQ-4JRU-9G3Q) Licence Info
1 Intel processor (No hyper-threading in cpu, single core).   [0 searches running]

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

```
Name       = SwissProt            Family   = C:/inetpub/mascot/sequence/SwissProt/current/SwissProt_*
Filename   = SwissProt_2016_10.fasta  Pathname = C:/inetpub/mascot/sequence/SwissProt/current/SwissPro
Status     = In use                            Statistics   Unidentified taxonomy Recompress file
State Time = Thu Nov 24 09:30:17 # 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

Name       = PRIDE Contaminants   Family   = C:/inetpub/mascot/sequence/PRIDE_Contaminants/current/PR
Filename   = PRIDE_Contaminants_20160906.msp  Pathname = C:/inetpub/mascot/sequence/PRIDE_Contaminants
Status     = In use                            Statistics Recompress file
State Time = Thu Nov 24 09:33:24 # 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
```

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'.

Security

Mascot security is disabled on installation. To enable Mascot security, refer to Chapter 12

Miscellaneous

Hyper-threading

Intel only: Hyper-threading is a technique used by Intel 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.

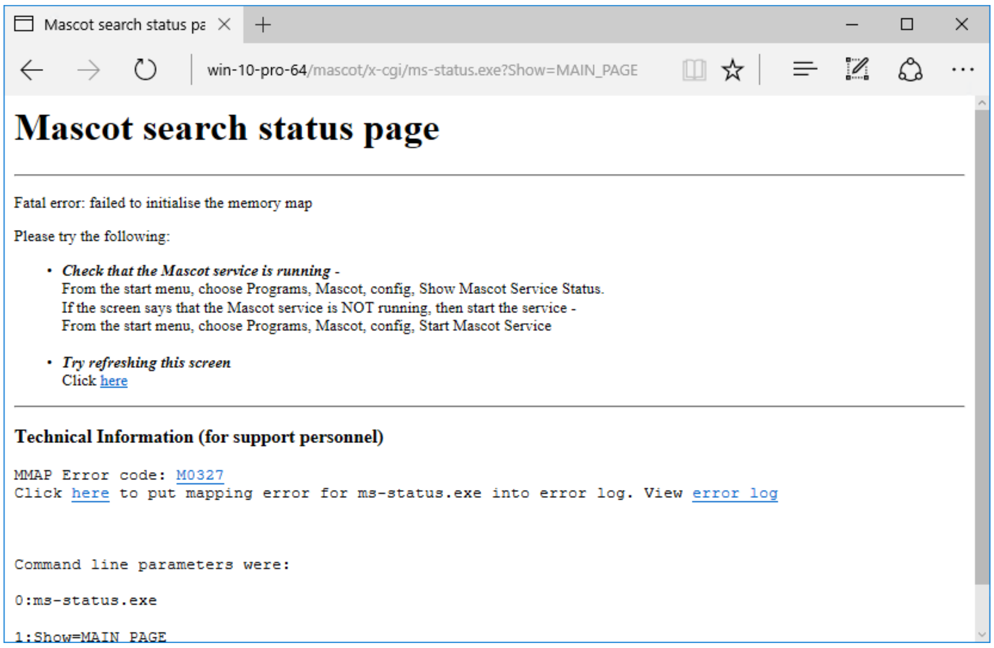
Troubleshooting

Check the Mascot Server Support Page

There may be a fix listed on the Matrix Science Web Site. From the menu, choose Support; Mascot Server and scan down to see if your problem is described.

The status screen shows an error

If the Mascot Monitor service fails to start, then the following text or something similar will be displayed in the status screen:



Mascot search status page

Fatal error: failed to initialise the memory map

Please try the following:

- ***Check that the Mascot service is running -***
  From the start menu, choose Programs, Mascot, config, Show Mascot Service Status.
  If the screen says that the Mascot service is NOT running, then start the service -
  From the start menu, choose Programs, Mascot, config, Start Mascot Service

- ***Try refreshing this screen***
  Click here

Technical Information (for support personnel)

```
MMAP Error code: M0327
Click here to put mapping error for ms-status.exe into error log. View error log


Command line parameters were:

0:ms-status.exe

1:Show=MAIN_PAGE
```

There are several possible causes:

Service not started

Since one of the first things that the Monitor service does is to create the memory mapped file, this could indicate that the service has not started. You can tell whether the service has started by choosing *Start; Programs; mascot; config; Show Mascot ms-monitor service status.*

If the service is not running, check the `monitor.log` and `errorlog.txt` file in the `logs` directory. If there is nothing in those files, then it may be necessary to try and run `ms-monitor.exe` as a command line executable. *You should only do this if the Mascot service is not running.* To do this, open a command prompt window, and change directory to the mascot `bin` directory. If your installation path was the default, you will need to type:

```
cd \Inetpub\mascot\bin
```

next start the monitor program:

```
ms-monitor DEBUG
```

Any error messages should be displayed on the screen. If possible, correct the faults, and then start the Mascot Service from the start menu. Note that the mascot service should never be running at the same time as `ms-monitor.exe` is being run from the command line.

International Versions of Windows

If Mascot is installed on a version of Windows that is not in the English language, then when the ms-status screen is displayed, it may have the error 'Failed to initialise memory map"

To correct this fault, the following procedure is required:

1. You will need to find the names of the 'groups' that your version of Windows uses for Administrators and Users. In German, for example, these names are "Administratoren" and "Benutzer" respectively. To see a list of User names, from the start menu, select Control panel, Administrative Tools, Computer Management. Expand Local Users and Groups.

2. From the start menu, select
   Programs | Mascot | Config | Stop Mascot Service

3. From the start menu, select
   Programs | Mascot | Config | Mascot Configuration File

4. Scroll down to near the bottom of the file and find the line:
   NTIUserGroup Users
   and change this to (for example, for German)
   NTIUserGroup Benutzer

5. Find the line
   NTMonitorGroup Administrators
   and change this to (for example, for German)
   NTMonitorGroup Administratoren

6. Save the mascot.dat file

7. From the start menu, select
   Programs | Mascot | Config | Start Mascot Service

8. Re-load the status page:
   Programs | Mascot | Search Status
   (You may need to refresh / reload the page)

9. For each active database, choose Recompress file

Wait until the files have been compressed and a test search has been done. Mascot is now ready for use.

The site search facility does not work

The local Mascot web pages are indexed using a product called ht://Dig. A log file is made as the indexes are built during the installation. The log file *mascot\htdig\build.log* may contain an error message indicating the nature of the problem.

If the web server was not operational during Mascot installation, it will not have been possible to build the keyword index. To build or rebuild it, open an administrator command window and enter the following commands. If Mascot was installed into a different path, you may have to modify the first two lines

```
C:
cd \inetpub\mascot\htdig
bin\htdig.exe -v
bin\htmerge.exe -v
```

Once the commands have completed, keyword search using the control at the top right of the web pages should be operational

Search status shows a failure to create compressed files

On the search screen, find out what caused the error by clicking on the *`Error log`* link, fix the fault, (possibly out of disk space), and then click on *`retry`*.

4

4. Validation

CGI Operation

To verify that the search engine is functioning correctly when executed as a CGI application, launch a JavaScript aware web browser and load the Mascot home page, (http://your_server/mascot/). Select Mascot from the main menu and then choose the "Peptide Mass Fingerprint" link near the top of the page. This will load the search form for a peptide mass fingerprint.

Enter your name and email address into the fields at the top of the form and type a number, say 1234, into the Query field. Then press the Start Search… button.

The search form will be replaced by the search progress screen. Once the search is complete, the Master Results page will appear. Unless you went to the trouble of entering some real mass values, the results will be meaningless!

Monitor Test

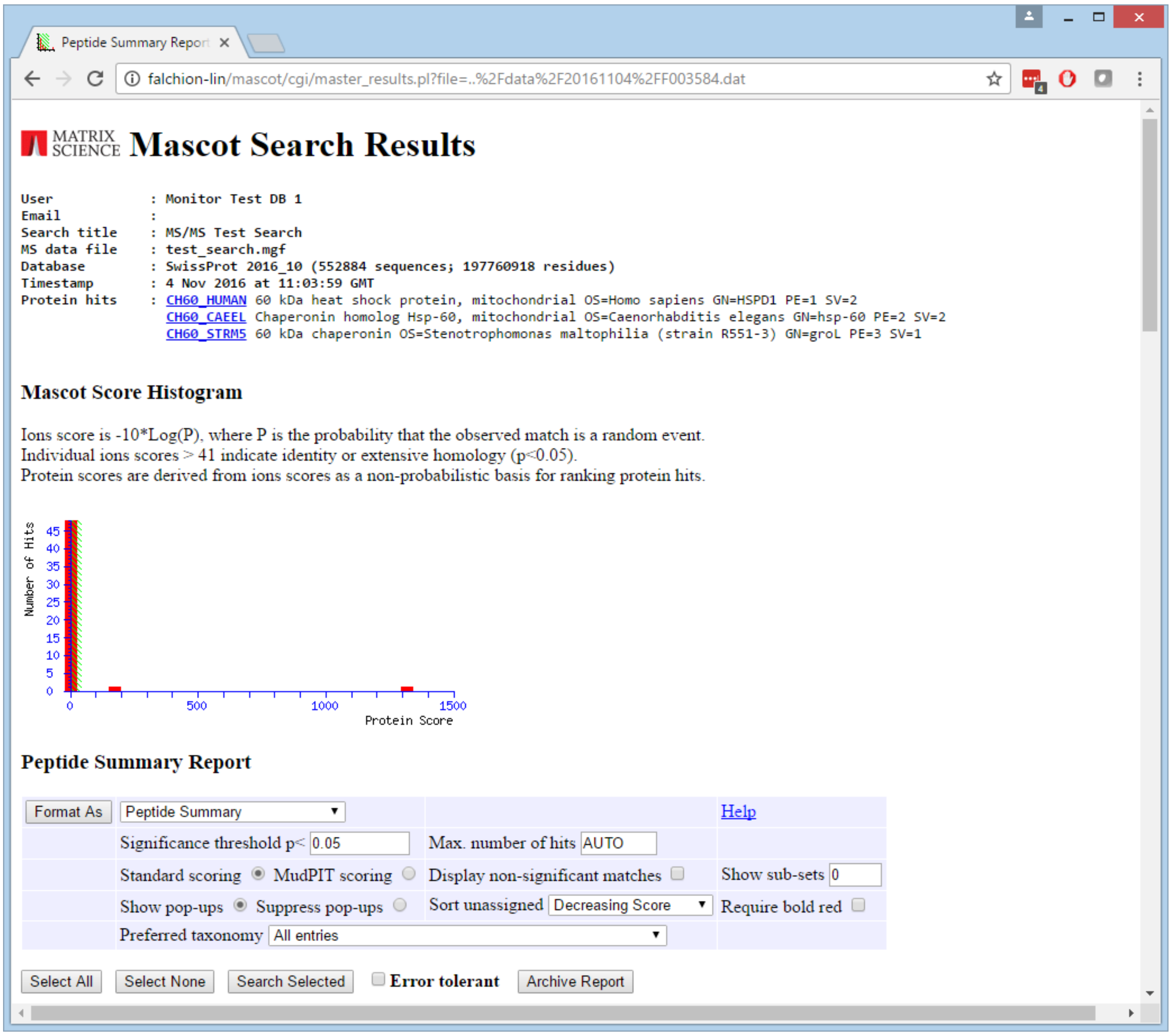
When Mascot Monitor is started, it runs a test search against each sequence database. It also runs this same test search against any update to the database as part of the exchange procedure. If the test search fails, an error message will be displayed in the Mascot Status screen and the database will not be available for searching. Error messages from Monitor are logged to *errorlog.txt* in the *mascot/logs* directory. Both this file and *monitor.log* can be viewed using links on the Mascot Status page.

The input file which defines a test search can be found in the *mascot/data/test* directory. The filename is constructed from the name of the database together with the extension .asc. For example, *SwissProt.asc*.

Note: Test files for new databases are generated by modifying a copy of *do_not_delete.asc*. Never delete this file.

The output of the test search may change slightly with each new update to a database. Sequences may be corrected or descriptions modified. Quite often, a new entry appears which is very homologous with one of the matched proteins so that it appears on the hit list.

Using SwissProt 2016_10, the report from running the standard test search is shown on the following pages.


Peptide Summary Report
falchion-lin/mascot/cgi/master_results.pl?file=..%2Fdata%2F20161104%2FF003584.dat
MATRIX SCIENCE

Mascot Search Results

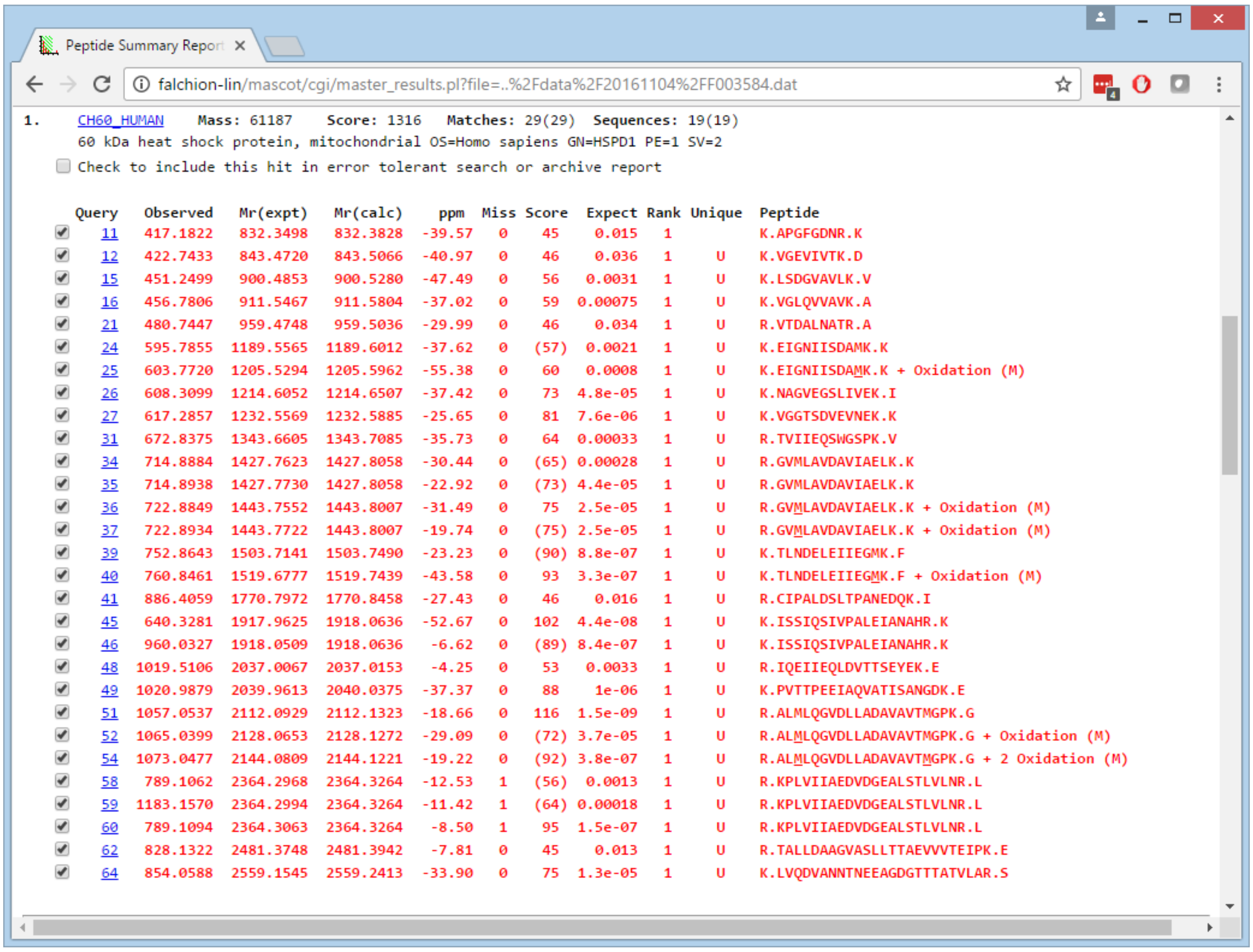
```
User            : Monitor Test DB 1
Email           :
Search title    : MS/MS Test Search
MS data file    : test_search.mgf
Database        : SwissProt 2016_10 (552884 sequences; 197760918 residues)
Timestamp       : 4 Nov 2016 at 11:03:59 GMT
Protein hits    : CH60_HUMAN 60 kDa heat shock protein, mitochondrial OS=Homo sapiens GN=HSPD1 PE=1 SV=2
                  CH60_CAEEL Chaperonin homolog Hsp-60, mitochondrial OS=Caenorhabditis elegans GN=hsp-60 PE=2 SV=2
                  CH60_STRM5 60 kDa chaperonin OS=Stenotrophomonas maltophilia (strain R551-3) GN=groL PE=3 SV=1
```

Mascot Score Histogram

Ions score is -10*Log(P), where P is the probability that the observed match is a random event.
Individual ions scores > 41 indicate identity or extensive homology (p<0.05).
Protein scores are derived from ions scores as a non-probabilistic basis for ranking protein hits.

Peptide Summary Report

Format As	Peptide Summary		Help
	Significance threshold p< 0.05	Max. number of hits AUTO	
	[✓ Standard scoring] MudPIT scoring	Display non-significant matches ☐	Show sub-sets 0
	[✓ Show pop-ups] Suppress pop-ups	Sort unassigned Decreasing Score	Require bold red ☐
	Preferred taxonomy All entries		

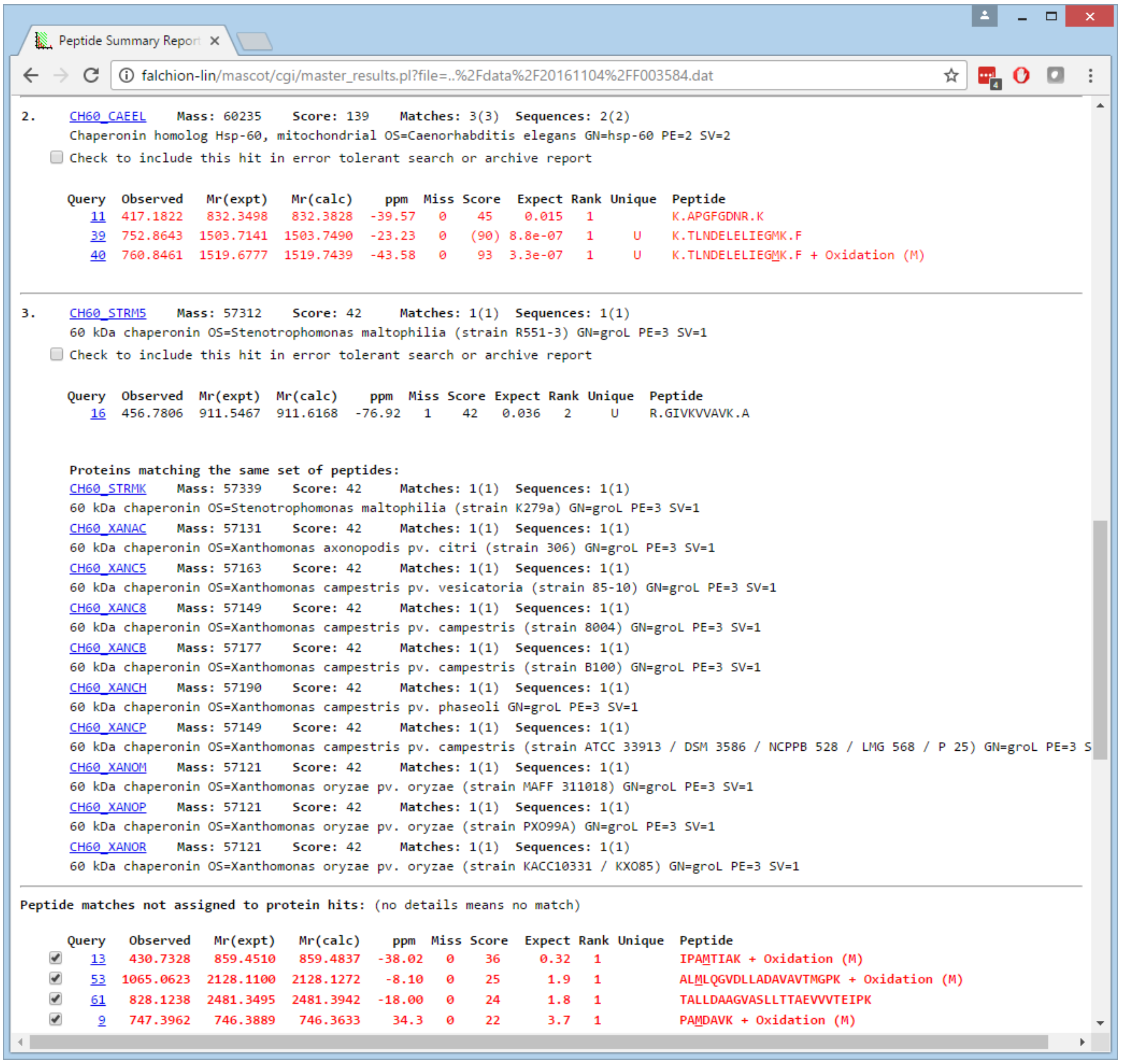


Peptide Summary Report

falchion-lin/mascot/cgi/master_results.pl?file=..%2Fdata%2F20161104%2FF003584.dat

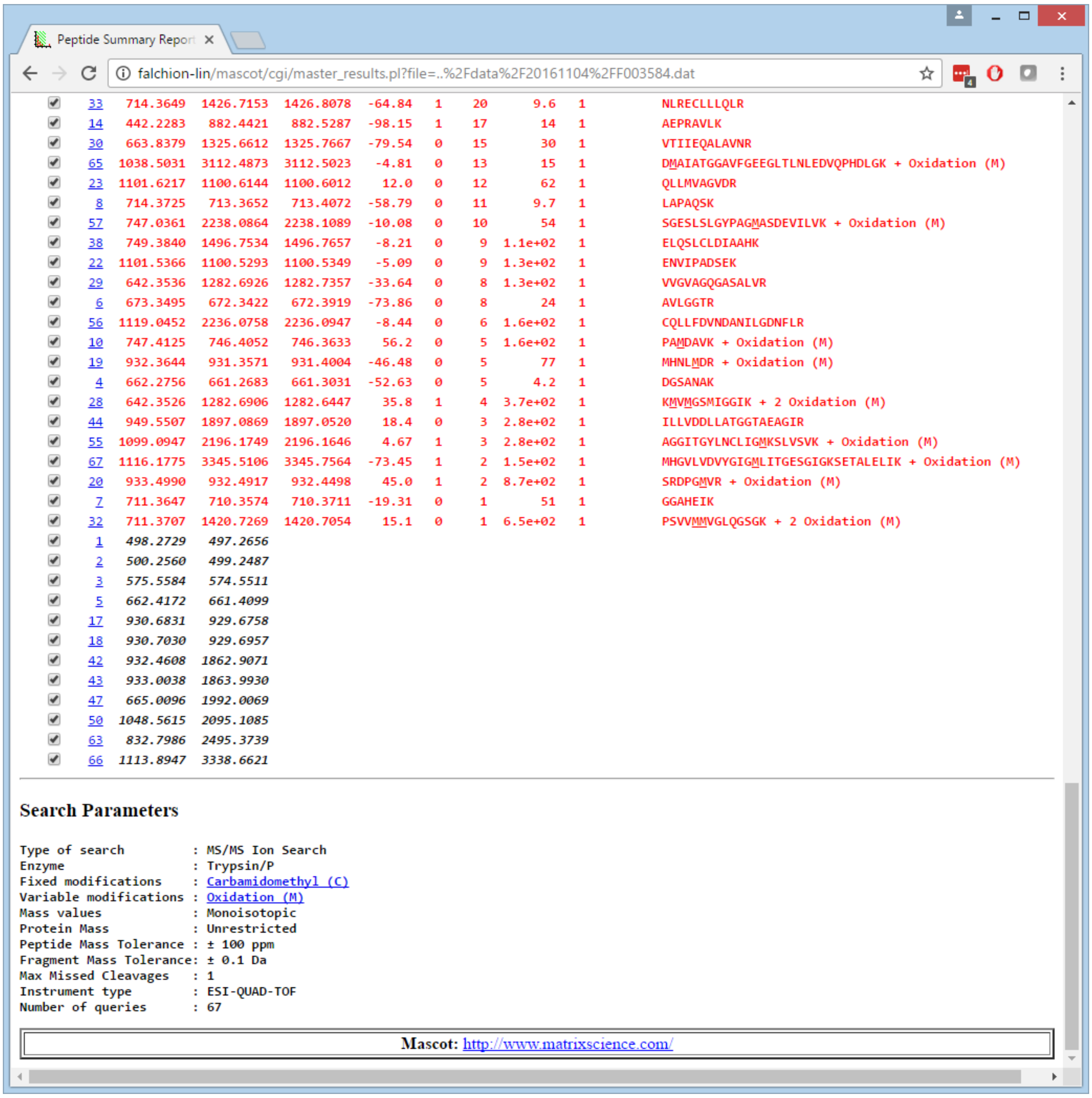
```
1.   CH60_HUMAN    Mass: 61187    Score: 1316    Matches: 29(29)   Sequences: 19(19)
     60 kDa heat shock protein, mitochondrial OS=Homo sapiens GN=HSPD1 PE=1 SV=2
     Check to include this hit in error tolerant search or archive report
```

	Query	Observed	Mr(expt)	Mr(calc)	ppm	Miss	Score	Expect	Rank	Unique	Peptide
✓	11	417.1822	832.3498	832.3828	-39.57	0	45	0.015	1		K.APGFGDNR.K
✓	12	422.7433	843.4720	843.5066	-40.97	0	46	0.036	1	U	K.VGEVIVTK.D
✓	15	451.2499	900.4853	900.5280	-47.49	0	56	0.0031	1	U	K.LSDGVAVLK.V
✓	16	456.7806	911.5467	911.5804	-37.02	0	59	0.00075	1	U	K.VGLQVVAVK.A
✓	21	480.7447	959.4748	959.5036	-29.99	0	46	0.034	1	U	R.VTDALNATR.A
✓	24	595.7855	1189.5565	1189.6012	-37.62	0	(57)	0.0021	1	U	K.EIGNIISDAMK.K
✓	25	603.7720	1205.5294	1205.5962	-55.38	0	60	0.0008	1	U	K.EIGNIISDA<u>M</u>K.K + Oxidation (M)
✓	26	608.3099	1214.6052	1214.6507	-37.42	0	73	4.8e-05	1	U	K.NAGVEGSLIVEK.I
✓	27	617.2857	1232.5569	1232.5885	-25.65	0	81	7.6e-06	1	U	K.VGGTSDVEVNEK.K
✓	31	672.8375	1343.6605	1343.7085	-35.73	0	64	0.00033	1	U	R.TVIIEQSWGSPK.V
✓	34	714.8884	1427.7623	1427.8058	-30.44	0	(65)	0.00028	1	U	R.GVMLAVDAVIAELK.K
✓	35	714.8938	1427.7730	1427.8058	-22.92	0	(73)	4.4e-05	1	U	R.GVMLAVDAVIAELK.K
✓	36	722.8849	1443.7552	1443.8007	-31.49	0	75	2.5e-05	1	U	R.GV<u>M</u>LAVDAVIAELK.K + Oxidation (M)
✓	37	722.8934	1443.7722	1443.8007	-19.74	0	(75)	2.5e-05	1	U	R.GV<u>M</u>LAVDAVIAELK.K + Oxidation (M)
✓	39	752.8643	1503.7141	1503.7490	-23.23	0	(90)	8.8e-07	1	U	K.TLNDELEIIEGMK.F
✓	40	760.8461	1519.6777	1519.7439	-43.58	0	93	3.3e-07	1	U	K.TLNDELEIIEG<u>M</u>K.F + Oxidation (M)
✓	41	886.4059	1770.7972	1770.8458	-27.43	0	46	0.016	1	U	R.CIPALDSLTPANEDQK.I
✓	45	640.3281	1917.9625	1918.0636	-52.67	0	102	4.4e-08	1	U	K.ISSIQSIVPALEIANAHR.K
✓	46	960.0327	1918.0509	1918.0636	-6.62	0	(89)	8.4e-07	1	U	K.ISSIQSIVPALEIANAHR.K
✓	48	1019.5106	2037.0067	2037.0153	-4.25	0	53	0.0033	1	U	R.IQEIIEQLDVTTSEYEK.E
✓	49	1020.9879	2039.9613	2040.0375	-37.37	0	88	1e-06	1	U	K.PVTTPEEIAQVATISANGDK.E
✓	51	1057.0537	2112.0929	2112.1323	-18.66	0	116	1.5e-09	1	U	R.ALMLQGVDLLADAVAVTMGPK.G
✓	52	1065.0399	2128.0653	2128.1272	-29.09	0	(72)	3.7e-05	1	U	R.AL<u>M</u>LQGVDLLADAVAVTMGPK.G + Oxidation (M)
✓	54	1073.0477	2144.0809	2144.1221	-19.22	0	(92)	3.8e-07	1	U	R.AL<u>M</u>LQGVDLLADAVAVT<u>M</u>GPK.G + 2 Oxidation (M)
✓	58	789.1062	2364.2968	2364.3264	-12.53	1	(56)	0.0013	1	U	R.KPLVIIAEDVDGEALSTLVLNR.L
✓	59	1183.1570	2364.2994	2364.3264	-11.42	1	(64)	0.00018	1	U	R.KPLVIIAEDVDGEALSTLVLNR.L
✓	60	789.1094	2364.3063	2364.3264	-8.50	1	95	1.5e-07	1	U	R.KPLVIIAEDVDGEALSTLVLNR.L
✓	62	828.1322	2481.3748	2481.3942	-7.81	0	45	0.013	1	U	R.TALLDAAGVASLLTTAEVVVTEIPK.E
✓	64	854.0588	2559.1545	2559.2413	-33.90	0	75	1.3e-05	1	U	K.LVQDVANNTNEEAGDGTTTATVLAR.S


Peptide Summary Report

falchion-lin/mascot/cgi/master_results.pl?file=..%2Fdata%2F20161104%2FF003584.dat

```
2.     CH60_CAEEL     Mass: 60235     Score: 139     Matches: 3(3)  Sequences: 2(2)
       Chaperonin homolog Hsp-60, mitochondrial OS=Caenorhabditis elegans GN=hsp-60 PE=2 SV=2
     ☐ Check to include this hit in error tolerant search or archive report

      Query  Observed   Mr(expt)   Mr(calc)    ppm  Miss Score   Expect Rank Unique   Peptide
         11  417.1822   832.3498   832.3828  -39.57   0    45     0.015   1           K.APGFGDNR.K
         39  752.8643  1503.7141  1503.7490  -23.23   0   (90) 8.8e-07   1      U     K.TLNDELELIEGMK.F
         40  760.8461  1519.6777  1519.7439  -43.58   0    93  3.3e-07   1      U     K.TLNDELELIEGMK.F + Oxidation (M)

3.     CH60_STRM5     Mass: 57312     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Stenotrophomonas maltophilia (strain R551-3) GN=groL PE=3 SV=1
     ☐ Check to include this hit in error tolerant search or archive report

      Query  Observed  Mr(expt)  Mr(calc)    ppm  Miss Score Expect Rank Unique  Peptide
         16  456.7806  911.5467  911.6168  -76.92   1    42   0.036   2     U    R.GIVKVVAVK.A


       Proteins matching the same set of peptides:
       CH60_STRMK     Mass: 57339     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Stenotrophomonas maltophilia (strain K279a) GN=groL PE=3 SV=1
       CH60_XANAC     Mass: 57131     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas axonopodis pv. citri (strain 306) GN=groL PE=3 SV=1
       CH60_XANC5     Mass: 57163     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas campestris pv. vesicatoria (strain 85-10) GN=groL PE=3 SV=1
       CH60_XANC8     Mass: 57149     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas campestris pv. campestris (strain 8004) GN=groL PE=3 SV=1
       CH60_XANCB     Mass: 57177     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas campestris pv. campestris (strain B100) GN=groL PE=3 SV=1
       CH60_XANCH     Mass: 57190     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas campestris pv. phaseoli GN=groL PE=3 SV=1
       CH60_XANCP     Mass: 57149     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas campestris pv. campestris (strain ATCC 33913 / DSM 3586 / NCPPB 528 / LMG 568 / P 25) GN=groL PE=3 S
       CH60_XANOM     Mass: 57121     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas oryzae pv. oryzae (strain MAFF 311018) GN=groL PE=3 SV=1
       CH60_XANOP     Mass: 57121     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas oryzae pv. oryzae (strain PXO99A) GN=groL PE=3 SV=1
       CH60_XANOR     Mass: 57121     Score: 42      Matches: 1(1)  Sequences: 1(1)
       60 kDa chaperonin OS=Xanthomonas oryzae pv. oryzae (strain KACC10331 / KXO85) GN=groL PE=3 SV=1

Peptide matches not assigned to protein hits: (no details means no match)

      Query   Observed   Mr(expt)   Mr(calc)    ppm  Miss Score   Expect Rank Unique   Peptide
  ✓     13   430.7328   859.4510   859.4837  -38.02   0    36     0.32   1            IPAMTIAK + Oxidation (M)
  ✓     53  1065.0623  2128.1100  2128.1272   -8.10   0    25      1.9   1            ALMLQGVDLLADAVAVTMGPK + Oxidation (M)
  ✓     61   828.1238  2481.3495  2481.3942  -18.00   0    24      1.8   1            TALLDAAGVASLLTTAEVVVTEIPK
  ✓      9   747.3962   746.3889   746.3633   34.3   0    22      3.7   1            PAMDAVK + Oxidation (M)
```


Peptide Summary Report
falchion-lin/mascot/cgi/master_results.pl?file=..%2Fdata%2F20161104%2FF003584.dat

```
✓  33   714.3649 1426.7153 1426.8078 -64.84  1  20     9.6  1   NLRECLLLQLR
✓  14   442.2283  882.4421  882.5287 -98.15  1  17      14  1   AEPRAVLK
✓  30   663.8379 1325.6612 1325.7667 -79.54  0  15      30  1   VTIIEQALAVNR
✓  65  1038.5031 3112.4873 3112.5023  -4.81  0  13      15  1   DMAIATGGAVFGEEGLTLNLEDVQPHDLGK + Oxidation (M)
✓  23  1101.6217 1100.6144 1100.6012   12.0  0  12      62  1   QLLMVAGVDR
✓   8   714.3725  713.3652  713.4072 -58.79  0  11     9.7  1   LAPAQSK
✓  57   747.0361 2238.0864 2238.1089 -10.08  0  10      54  1   SGESLSLGYPAGMASDEVILVK + Oxidation (M)
✓  38   749.3840 1496.7534 1496.7657  -8.21  0   9 1.1e+02  1   ELQSLCLDIAAHK
✓  22  1101.5366 1100.5293 1100.5349  -5.09  0   9 1.3e+02  1   ENVIPADSEK
✓  29   642.3536 1282.6926 1282.7357 -33.64  0   8 1.3e+02  1   VVGVAGQGASALVR
✓   6   673.3495  672.3422  672.3919 -73.86  0   8      24  1   AVLGGTR
✓  56  1119.0452 2236.0758 2236.0947  -8.44  0   6 1.6e+02  1   CQLLFDVNDANILGDNFLR
✓  10   747.4125  746.4052  746.3633   56.2  0   5 1.6e+02  1   PAMDAVK + Oxidation (M)
✓  19   932.3644  931.3571  931.4004 -46.48  0   5      77  1   MHNLMDR + Oxidation (M)
✓   4   662.2756  661.2683  661.3031 -52.63  0   5     4.2  1   DGSANAK
✓  28   642.3526 1282.6906 1282.6447   35.8  1   4 3.7e+02  1   KMVMGSMIGGIK + 2 Oxidation (M)
✓  44   949.5507 1897.0869 1897.0520   18.4  0   3 2.8e+02  1   ILLVDDLLATGGTAEAGIR
✓  55  1099.0947 2196.1749 2196.1646   4.67  1   3 2.8e+02  1   AGGITGYLNCLIGMKSLVSVK + Oxidation (M)
✓  67  1116.1775 3345.5106 3345.7564 -73.45  1   2 1.5e+02  1   MHGVLVDVYGIGMLITGESGIGKSETALELIK + Oxidation (M)
✓  20   933.4990  932.4917  932.4498   45.0  1   2 8.7e+02  1   SRDPGMVR + Oxidation (M)
✓   7   711.3647  710.3574  710.3711 -19.31  0   1      51  1   GGAHEIK
✓  32   711.3707 1420.7269 1420.7054   15.1  0   1 6.5e+02  1   PSVVMMVGLQGSGK + 2 Oxidation (M)
✓   1   498.2729  497.2656
✓   2   500.2560  499.2487
✓   3   575.5584  574.5511
✓   5   662.4172  661.4099
✓  17   930.6831  929.6758
✓  18   930.7030  929.6957
✓  42   932.4608 1862.9071
✓  43   933.0038 1863.9930
✓  47   665.0096 1992.0069
✓  50  1048.5615 2095.1085
✓  63   832.7986 2495.3739
✓  66  1113.8947 3338.6621
```

Search Parameters

```
Type of search         : MS/MS Ion Search
Enzyme                 : Trypsin/P
Fixed modifications    : Carbamidomethyl (C)
Variable modifications : Oxidation (M)
Mass values            : Monoisotopic
Protein Mass           : Unrestricted
Peptide Mass Tolerance : ± 100 ppm
Fragment Mass Tolerance: ± 0.1 Da
Max Missed Cleavages   : 1
Instrument type        : ESI-QUAD-TOF
Number of queries      : 67
```

Mascot: http://www.matrixscience.com/

5

5. Sequence Database Setup

Sequence database URL's and formats change constantly. Provided your Mascot Server can connect to the Internet, Mascot Database Manager will keep database definitions up-to-date automatically for many popular public databases. For each database, you can configure a file update schedule, so that new releases are downloaded automatically. For more information about Database Manager, refer to the Mascot HTML help pages.

If you want to set up a custom database, such as the proteome or genome of a single organism, download and configuration information can also be found in the Mascot HTML help pages. Note that the HTML help pages for your in-house Server are only updated when you install a new version of Mascot, so for the latest information, go to the help pages on the Matrix Science public web site (http://www.matrixscience.com/help/seq_db_setup.html).

This chapter contains reference material, most of which is only important if you choose not to use Database Manager.

The Fasta Format

Mascot can search any Fasta format sequence database as long as it can parse a unique identifier (accession string) from each entry in a consistent fashion. The accession string can contain any US-ASCII printing characters except comma and double quotes.

The Fasta format is extremely simple. Each entry consists of a one line title followed by one or more lines containing the contiguous sequence string in 1 letter code. Fasta databases can contain either amino acid sequences or nucleic acid sequences, but not both. Nucleic acid databases are translated on the fly by Mascot in all six reading frames.

The Fasta title line begins with a "greater than" character, followed by one or more accession strings, and an optional text string describing the entry. Apart from the use of the "greater than" character, the precise syntax of the title line varies from database to database. The title line is delimited from the sequence that follows by a platform dependent new line character.

The title line is followed by lines of contiguous sequence characters. Line lengths vary between databases; anything from 60 characters to a thousand or more. Mascot can handle lines up to 50,000 characters long. The end of a sequence is indicated when the following line is either a new title line or the end of the file. For example:

```
.
.
.
VYEYVRKYAEHRMLVVAEQPLHAMRKGLLDVLPKNSLEDLTAEDFRLLVNGCGEVNVQML
ISFTSFNDESGENAEKLLQFKRWFWSIVERMSMTERQDLVYFWTSSPSLPASEEGFQPMP
SITIRPPDDQHLPTANTCISRLYVPLYSSKQILKQKLLLAIKTKNFGFV
>104K_THEPA (P15711) 104 KD MICRONEME-RHOPTRY ANTIGEN.
MKFLILLFNILCLFPVLAADNHGVGPQGASGVDPITFDINSNQTGPAFLTAVEMAGVKYL
QVQHGSNVNIHRLVEGNVVIWENASTPLYTGAIVTNNDGPYMAYVEVLGDPNLQFFIKSG
DAWVTLSEHEYLAKLQEIRQAVHIESVFSLNMAFQLENNKYEVETHAKNGANMVTFIPRN
.
.
.
```

Mascot doesn’t search the Fasta file directly. When a new database is recognised, Mascot Monitor uses the Fasta file to create a set of compressed files. One reason for doing this to separate the sequence string from the title line, because only the sequence string needs to be memory mapped. In the case of a database with predominantly short sequences, this greatly reduces the amount of memory required. In the case of a nucleic acid database, the limited character set allows Mascot to pack two base codes into each byte of memory. If a taxonomy filter is required, a taxonomy index is built at the same time as the file is compressed.

Spectral library formats

Spectral libraries are searched using NIST MSPepSearch. Libraries are downloaded or copied as MSP text files. The MSP format is described here:

http://chemdata.nist.gov/mass-spc/ftp/mass-spc/PepLib.pdf

Before a library can be searched, it must be converted to a binary NIST MS library using a utility called LIB2NIST. Mascot Monitor also creates CDB format look-up files that map library peptide sequences to protein accessions in a reference database.

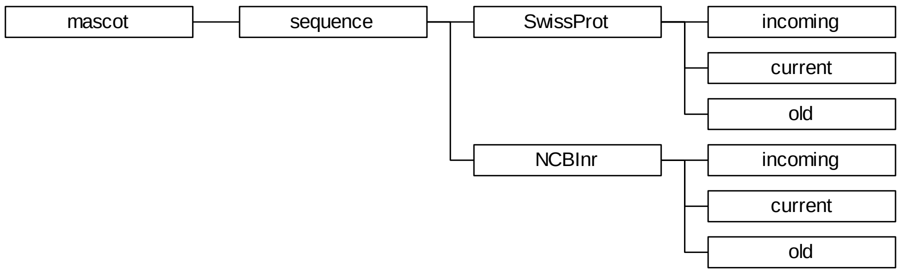
Naming Conventions and Directory Structure

Although Microsoft Windows permits file and directory names to include spaces, file and directory names to be used by Mascot, or to appear in a URL, cannot include spaces.

By following some simple conventions in database naming, Mascot Monitor enables sequence databases to be automatically updated without any disruption to on-going searches.

The procedure followed by Monitor is that the new database is compressed and tested by running a standard search. If errors are detected in the new database, the database exchange process is abandoned. Assuming the test is successful, all new searches are performed against the new database, while searches that were in progress against the old database are allowed to continue. Once the final search against the old database is complete, the disk file is moved into an archive directory. If the database being exchanged is memory mapped, the mapping and un-mapping are also handled automatically.

Assuming that the new database will be updated periodically, a directory structure similar to the one created for SwissProt during installation is recommended. For example:



For each database, the *incoming* directory provides a workspace for downloading and expanding a new database file. The *current* directory contains the active database, and this is where Mascot Monitor creates the memory mapped compressed files. The *old* directory is where the immediate past database file is archived … just in case.

In the Mascot configuration, the filename for each database **must** include a wild card. This is to enable the automatic recognition and exchange of an update file. For example, the filename for the SwissProt database might be defined as `SwissProt_*.fasta`. This would match to filenames that included a release number, e.g. *SwissProt_2012_03.fasta,* or a date stamp, e.g. *SwissProt_20120311.fasta*.

Whenever Monitor sees a file in that directory which matches to the database name and is *not* the current database, it will initiate the exchange process. This is why the wild card is important, even though you may not wish to track database dates or revision numbers.

Even if you never intend to swap a database, and have called it (say) *SwissProt.fasta*, you must still define it in the Mascot configuration using a wild card as `SwissProt*.fasta`.

Database File Update Procedure

Mascot Database Manager can update database files automatically to a specified schedule. This section describes how to update the files for a database if your Mascot Server is not connected to the Internet or if you choose not to use Database Manager.

When a new release of a database becomes available, it should be copied or downloaded into the *incoming* directory. In many cases, the downloaded file will have to be de-compressed. The filename may or may not be constant from release to release.

The Fasta database should be renamed to a name that includes a version or date stamp and matches the wild card path for the database, then **moved** to the current directory. Never copy a large file to the current directory under its final name because this will take time and the exchange process may be triggered prematurely.

If you are using a local reference file, rename and move this file first. Otherwise, the exchange process will be triggered by the appearance of the Fasta file, but will immediately fail because the new reference file is not yet available. Note that Fasta and reference files must have identical names apart from the filename extension.

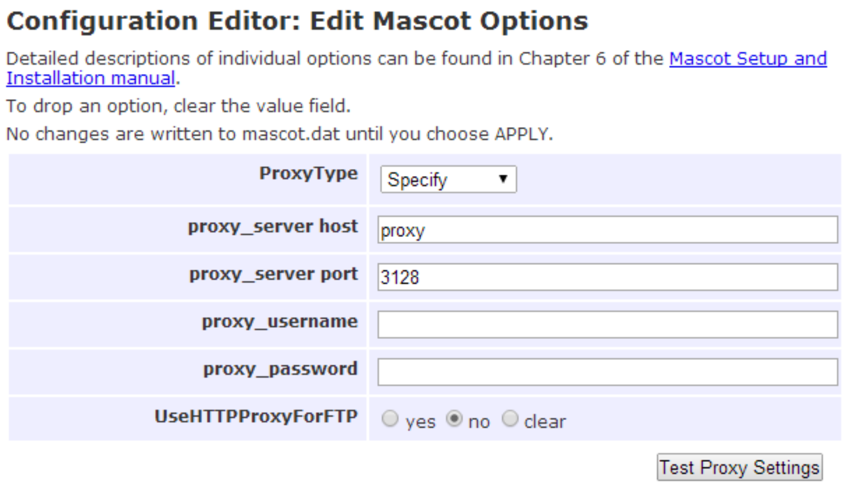
Once Mascot Monitor sees a new Fasta file that matches the wild card path for the database, it will begin the exchange process. Progress can be monitored from the Mascot Database Status page.

If your Mascot Server is not connected to the Internet, download the required files on a PC with Internet access and copy them to your Mascot Server. Download URLs and configuration information for popular databases can be found on the Matrix Science web site at http://www.matrixscience.com/help/seq_db_setup.html

Troubleshooting

Proxy Server

If there is a proxy server between your Mascot server and the Internet, downloads may fail unless you define your proxy server in the Options section of `mascot.dat`. Proxy settings can be modified and tested in Database Manager or, if you choose not to use Database Manager, in Configuration Options:


Configuration Editor: Edit Mascot Options

Detailed descriptions of individual options can be found in Chapter 6 of the Mascot Setup and Installation manual.

To drop an option, clear the value field.

No changes are written to mascot.dat until you choose APPLY.

ProxyType	Specify
proxy_server host	proxy
proxy_server port	3128
proxy_username	
proxy_password	
UseHTTPProxyForFTP	yes [✓ no] clear

Permissions / Security

Mascot monitor will need to create the compressed database files in the database `current` directory, and may need to move old database files to the `old` directory. Mascot searches, running as CGI processes with very restricted privileges, need to read the files. Make sure Linux permissions or Windows security settings don't prevent this.

Files not where they are supposed to be

When you enable the database, if nothing happens, double check that the sequence database files are exactly where the Path definition specifies. Note that the taxonomy files are shared, and go into the Mascot `taxonomy` directory, not a sequence database directory. Under Windows, remember that the directory separator in Database Manager and in mascot.dat must be a forward slash, not a back slash.

6

6. Configuration & Log Files

Configuration Files

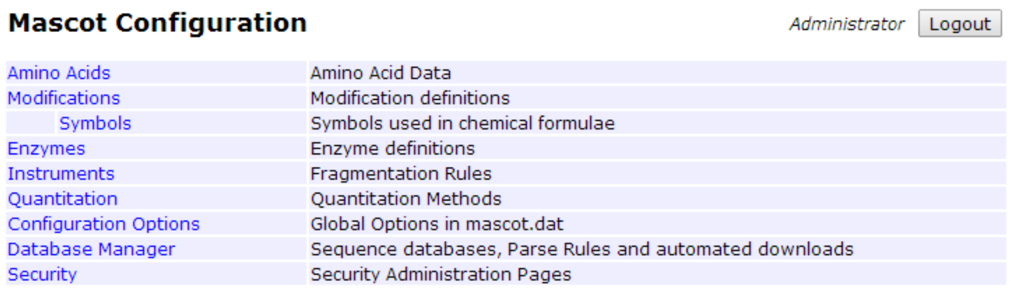
Mascot configuration files are located in the `mascot/config` directory:

> `unimod.xml` defines mass values and modifications, including substitutions
>
> `enzymes` defines enzyme cleavage specificity
>
> `fragmentation_rules` specifies which fragment ion series correspond to defined instrument types
>
> `mascot.dat` contains general configuration information. If you use Database Manager, do not modify the sequence database-related sections of mascot.dat because any changes will be lost when Database Manager is next used.
>
> `taxonomy` specifies the taxonomy filter choices for the search form (described in Chapter 9)
>
> `quantitation.xml` defines quantitation methods
>
> `nodelist.txt` configures the systems belonging to a Mascot cluster (described in Chapter 11)
>
> `user.xml, group.xml, security_options.xml, and security_tasks.xml` are the configuration files for Mascot security, described in Chapter 12
>
> `mod_file, masses, and substitutions` are obsolete configuration files that are created on the fly from `unimod.xml` to support third party applications that expect to find these files.
>
> Files in `config/dbmanager` are configuration files used by Database Manager. For descriptions, see the Database Manager HTML help page.

A browser-based Configuration Editor is provided to view and edit these files. These files are all text files, so can also be edited in any text editor. If you choose to edit the files, exercise care and always make a backup first, because seemingly small errors can render Mascot unusable.

Configuration Editor

The local Mascot home page contains a link to the Configuration Editor. The top-level page is a menu. If Mascot security is enabled, members of the Administrators group see the menu item for Mascot security



Mascot Configuration Administrator Logout

Amino Acids	Amino Acid Data
Modifications	Modification definitions
Symbols	Symbols used in chemical formulae
Enzymes	Enzyme definitions
Instruments	Fragmentation Rules
Quantitation	Quantitation Methods
Configuration Options	Global Options in mascot.dat
Database Manager	Sequence databases, Parse Rules and automated downloads
Security	Security Administration Pages

unimod.xml

Do not edit or update unimod.xml. This file is created by merging entries from master.xml and usermods.xml in the config/unimod directory, and your changes will be lost. If you wish to edit or update modifications outside of the configuration editor, the file to edit or update is `config/unimod/master.xml`.

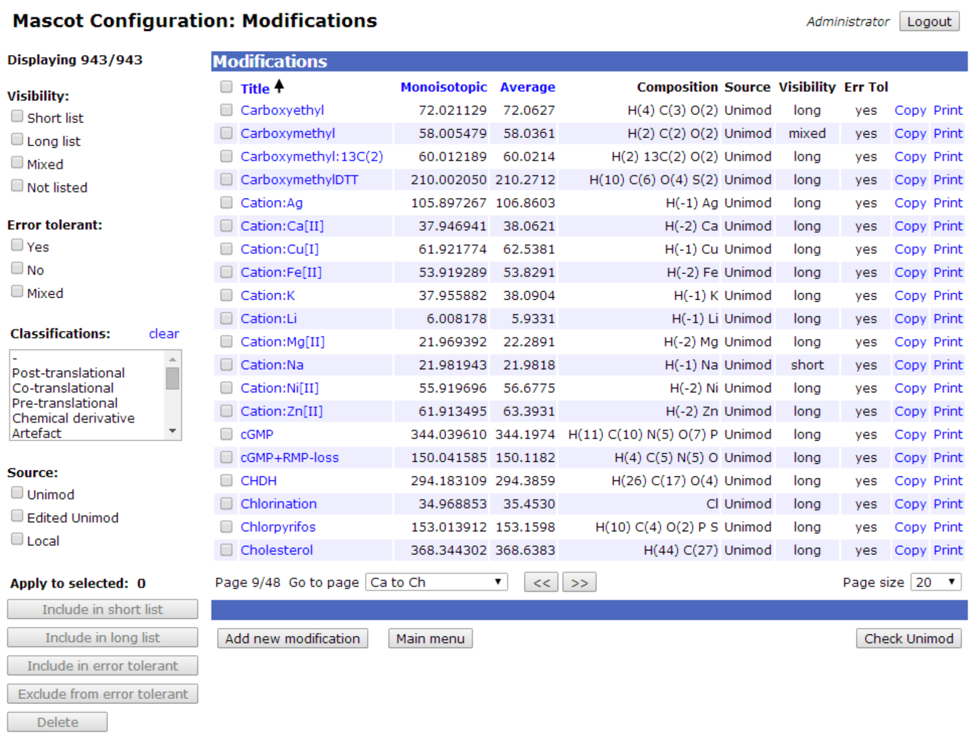
The first three menu items: Amino acids, Modifications, and Symbols, are interfaces to different aspects of the `unimod.xml` configuration file

In the Amino Acids module, the standard amino acids are read-only, but J, O, and U can be redefined.

The Symbols module displays the read-only masses and compositions of all the elements and molecules that are available for creating modifications.

The Modifications module can be used to browse, add, delete, and re-define modifications. Changes made locally using the configuration editor are stored separately from the definitions present in the unimod.xml file downloaded from www.unimod.org, referred to as the 'master' file. This allows the master file to be updated without losing local changes.

(Unless a modification is confidential or experimental, it is better to add it to the public Unimod database, www.unimod.org, and later download an updated unimod.xml file. By going this route, you share the new modification with others, and benefit in turn from other people's updates.)


Mascot Configuration: Modifications

Administrator Logout

Displaying 943/943

Visibility:
- ☐ Short list
- ☐ Long list
- ☐ Mixed
- ☐ Not listed

Error tolerant:
- ☐ Yes
- ☐ No
- ☐ Mixed

Classifications: clear

-
Post-translational
Co-translational
Pre-translational
Chemical derivative
Artefact

Source:
- ☐ Unimod
- ☐ Edited Unimod
- ☐ Local

Apply to selected: 0

Include in short list | Include in long list | Include in error tolerant | Exclude from error tolerant | Delete

Modifications

☐	Title	Monoisotopic	Average	Composition	Source	Visibility	Err Tol		
☐	Carboxyethyl	72.021129	72.0627	H(4) C(3) O(2)	Unimod	long	yes	Copy	Print
☐	Carboxymethyl	58.005479	58.0361	H(2) C(2) O(2)	Unimod	mixed	yes	Copy	Print
☐	Carboxymethyl:13C(2)	60.012189	60.0214	H(2) 13C(2) O(2)	Unimod	long	yes	Copy	Print
☐	CarboxymethylDTT	210.002050	210.2712	H(10) C(6) O(4) S(2)	Unimod	long	yes	Copy	Print
☐	Cation:Ag	105.897267	106.8603	H(-1) Ag	Unimod	long	yes	Copy	Print
☐	Cation:Ca[II]	37.946941	38.0621	H(-2) Ca	Unimod	long	yes	Copy	Print
☐	Cation:Cu[I]	61.921774	62.5381	H(-1) Cu	Unimod	long	yes	Copy	Print
☐	Cation:Fe[II]	53.919289	53.8291	H(-2) Fe	Unimod	long	yes	Copy	Print
☐	Cation:K	37.955882	38.0904	H(-1) K	Unimod	long	yes	Copy	Print
☐	Cation:Li	6.008178	5.9331	H(-1) Li	Unimod	long	yes	Copy	Print
☐	Cation:Mg[II]	21.969392	22.2891	H(-2) Mg	Unimod	long	yes	Copy	Print
☐	Cation:Na	21.981943	21.9818	H(-1) Na	Unimod	short	yes	Copy	Print
☐	Cation:Ni[II]	55.919696	56.6775	H(-2) Ni	Unimod	long	yes	Copy	Print
☐	Cation:Zn[II]	61.913495	63.3931	H(-2) Zn	Unimod	long	yes	Copy	Print
☐	cGMP	344.039610	344.1974	H(11) C(10) N(5) O(7) P	Unimod	long	yes	Copy	Print
☐	cGMP+RMP-loss	150.041585	150.1182	H(4) C(5) N(5) O	Unimod	long	yes	Copy	Print
☐	CHDH	294.183109	294.3859	H(26) C(17) O(4)	Unimod	long	yes	Copy	Print
☐	Chlorination	34.968853	35.4530	Cl	Unimod	long	yes	Copy	Print
☐	Chlorpyrifos	153.013912	153.1598	H(10) C(4) O(2) P S	Unimod	long	yes	Copy	Print
☐	Cholesterol	368.344302	368.6383	H(44) C(27)	Unimod	long	yes	Copy	Print

Page 9/48 Go to page Ca to Ch << >> Page size 20

Add new modification | Main menu | Check Unimod

The display can be filtered using the controls to the left.

Visibility defines where the modification appears in a search form. Short list is the list of common modifications and long list is the complete list. If a modification appears in the short list, this implies it also appears in the long list. Because visibility is defined at the specificity level, a modification can have mixed visibility, e.g. Oxidation (M) is in the short list but Oxidation (D) is not. If a specificity is in neither the short or long list, it is hidden from users.

Error tolerant defines whether the modification is included in the second pass of an error tolerant search. By default, all modifications are included.

Classifications have no effect in Mascot except that modifications classified as AA substitutions do not appear in the modifications list in a search form.

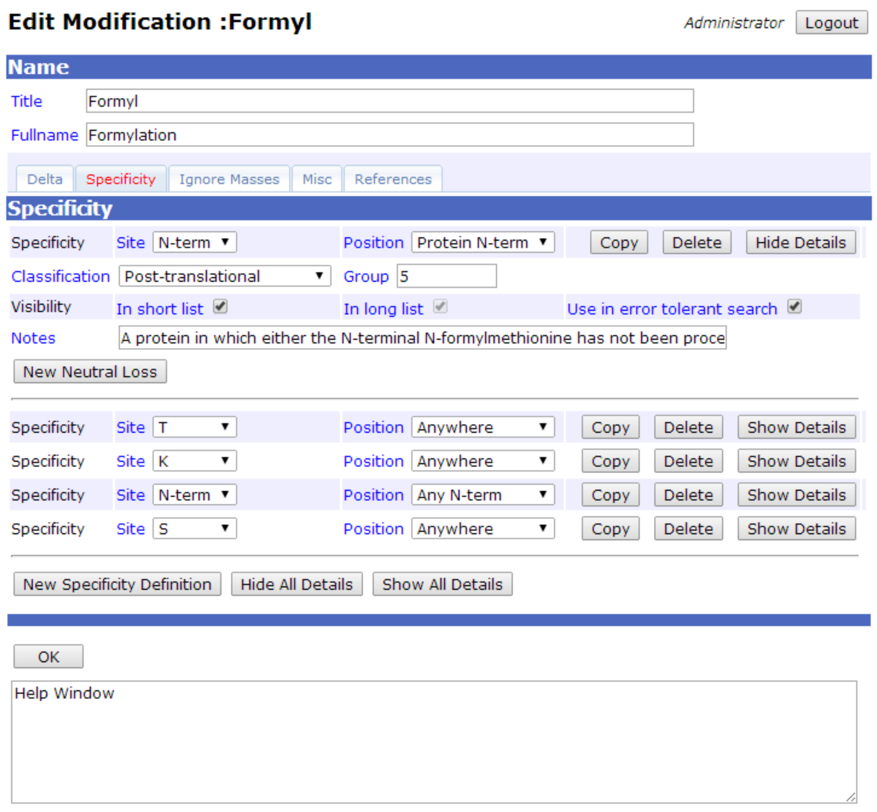
Source can be used to locate entries that have been modified or added locally.

The selection boxes can be used to change the visibility and error tolerant states of a set of modifications. Checking or clearing the checkbox in the header row changes the state of all checkboxes on the page. The buttons to the lower left then apply to the checked modifications.

Only locally defined modifications can be deleted. Modifications defined in Unimod cannot be deleted, but can be hidden by choosing not to display them in either the short or long lists.

To edit a modification, click on the name. If the modification is from the Unimod master list, it will be read-only unless you choose *Make editable*. This creates a copy of the modification that masks the original, and you can revert to the original by choosing *Revert to Unimod* and see how the local copy and the Unimod original differ by choosing *Show differences*.

You can add a new modification by starting from scratch (*Add new modification*) or by making a copy of an existing modification. Most of the controls associated with editing a modification are self-explanatory. Where necessary, help text is displayed when the mouse rolls over a hyperlink. Additional information about the Unimod modifications database can be found in the help pages on www.unimod.org. This is also the place to find details of the unimod.xml file format, which is defined by a schema called unimod_1.xsd.


Edit Modification :Formyl

Name

Title: Formyl

Fullname: Formylation

Specificity

Specificity	Site	N-term	Position	Protein N-term		
Classification	Post-translational		Group	5		
Visibility	In short list	✓	In long list	✓	Use in error tolerant search	✓
Notes	A protein in which either the N-terminal N-formylmethionine has not been proce					

Specificity	Site	T	Position	Anywhere
Specificity	Site	K	Position	Anywhere
Specificity	Site	N-term	Position	Any N-term
Specificity	Site	S	Position	Anywhere

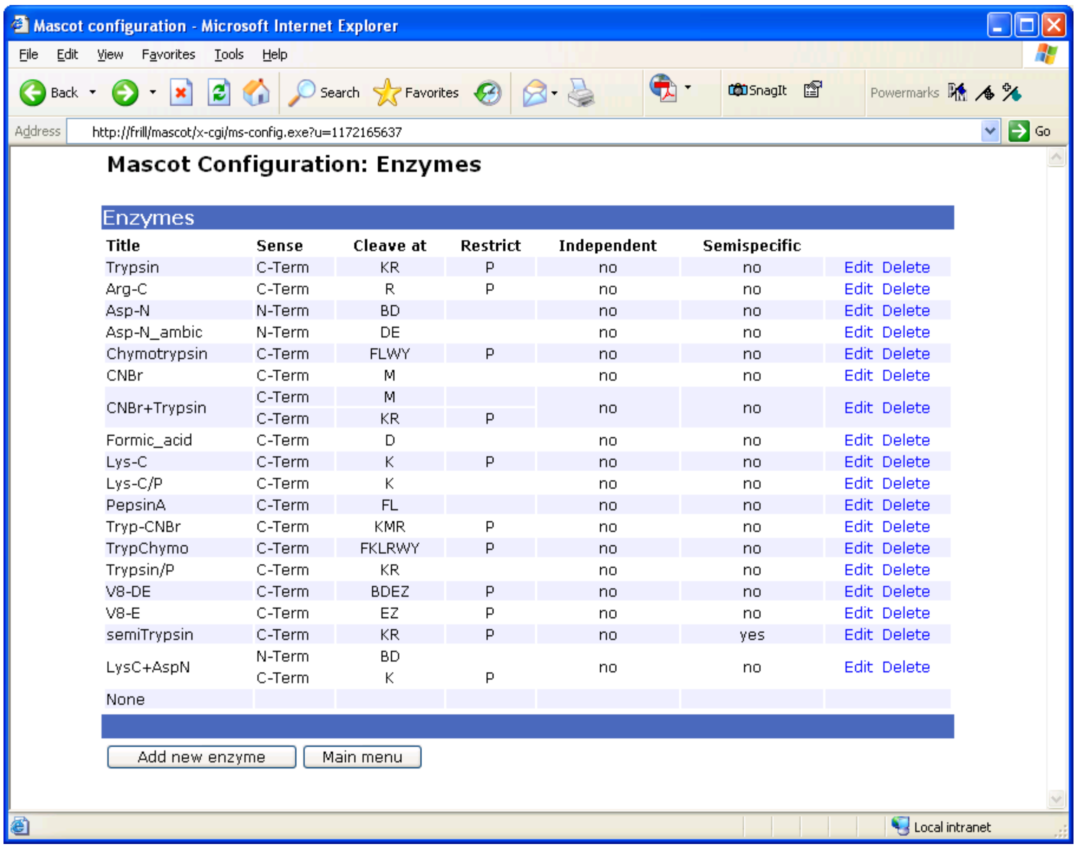
Help Window

Many properties of a modification are defined at the specificity level. For example, formyl at the protein N-term could be post-translational whereas formyl at any N-term would be an artefact. Each specificity must be different. Residue specificities that have identical neutral losses and visibility can be grouped together by giving them the same group number. For example, the search form list contains Phospho (ST) because both the S and T specificities have the same group number.

For more about grouping, neutral losses, and similar topics, see the blog articles Modifications round-up, parts 1 and 2 on the Matrix Science web site.

If Mascot security is enabled, when you add a new, local modification, you have the option to control which security groups can see it. On the privacy tab, check *Private*, and select which groups should have access to the entry. *Update* saves the changes while keeping the modification open in the editor. *Save changes* returns to the main display after saving.

Enzymes


Mascot configuration - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites SnagIt Powermarks
Address http://frill/mascot/x-cgi/ms-config.exe?u=1172165637
Go
Local intranet

Mascot Configuration: Enzymes

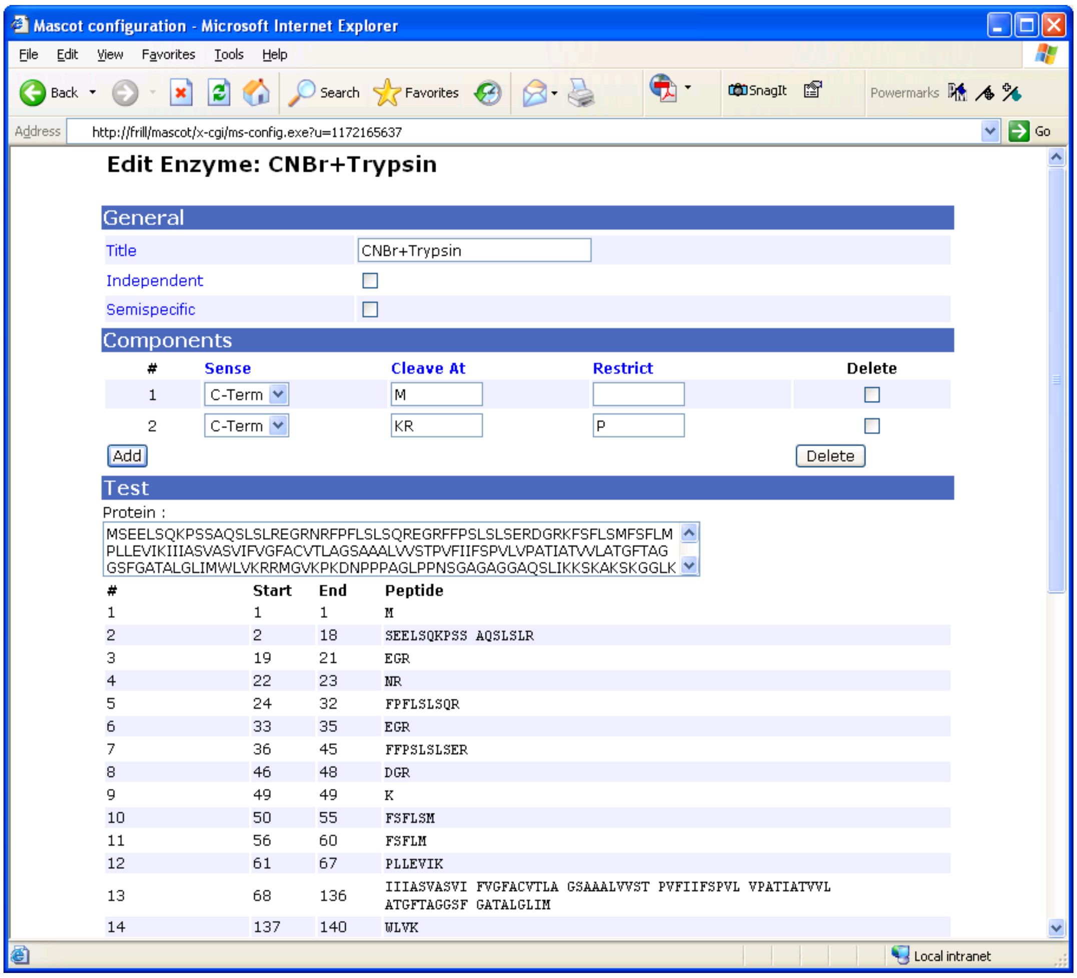
Enzymes

Title	Sense	Cleave at	Restrict	Independent	Semispecific	
Trypsin	C-Term	KR	P	no	no	Edit Delete
Arg-C	C-Term	R	P	no	no	Edit Delete
Asp-N	N-Term	BD		no	no	Edit Delete
Asp-N_ambic	N-Term	DE		no	no	Edit Delete
Chymotrypsin	C-Term	FLWY	P	no	no	Edit Delete
CNBr	C-Term	M		no	no	Edit Delete
CNBr+Trypsin	C-Term	M		no	no	Edit Delete
	C-Term	KR	P			
Formic_acid	C-Term	D		no	no	Edit Delete
Lys-C	C-Term	K	P	no	no	Edit Delete
Lys-C/P	C-Term	K		no	no	Edit Delete
PepsinA	C-Term	FL		no	no	Edit Delete
Tryp-CNBr	C-Term	KMR	P	no	no	Edit Delete
TrypChymo	C-Term	FKLRWY	P	no	no	Edit Delete
Trypsin/P	C-Term	KR		no	no	Edit Delete
V8-DE	C-Term	BDEZ	P	no	no	Edit Delete
V8-E	C-Term	EZ	P	no	no	Edit Delete
semiTrypsin	C-Term	KR	P	no	yes	Edit Delete
LysC+AspN	N-Term	BD		no	no	Edit Delete
	C-Term	K	P			
None						

Add new enzyme Main menu

Enzyme ‘None’ is a special case, which cannot be modified or deleted. All the other enzyme definitions can be edited or deleted, and new ones added.

The edit page allows you to test a new enzyme definition against a protein



Edit Enzyme: CNBr+Trypsin

General

Title	CNBr+Trypsin
Independent	☐
Semispecific	☐

Components

#	Sense	Cleave At	Restrict	Delete
1	C-Term	M		☐
2	C-Term	KR	P	☐

Test

Protein :

```
MSEELSQKPSSAQSLSLREGRNRFPFLSLSQREGRFFPSLSLSERDGRKFSFLSMFSFLM
PLLEVIKIIIASVASVIFVGFACVTLAGSAAALVVSTPVFIIFSPVLVPATIATVVLATGFTAG
GSFGATALGLIMWLVKRRMGVKPKDNPPPAGLPPNSGAGAGGAQSLIKKSKAKSKGGLK
```

#	Start	End	Peptide
1	1	1	M
2	2	18	SEELSQKPSS AQSLSLR
3	19	21	EGR
4	22	23	NR
5	24	32	FPFLSLSQR
6	33	35	EGR
7	36	45	FFPSLSLSER
8	46	48	DGR
9	49	49	K
10	50	55	FSFLSM
11	56	60	FSFLM
12	61	67	PLLEVIK
13	68	136	IIIASVASVI FVGFACVTLA GSAAALVVST PVFIIFSPVL VPATIATVVL ATGFTAGGSF GATALGLIM
14	137	140	WLVK

File format (enzymes)

Each cleavage agent is defined by a block of lines. Blocks are delimited from one another by a line containing an asterisk. Each line in a block starts with a keyword.

```
Title:Trypsin
Cleavage:KR
Restrict:P
Cterm
*
Title:Asp-N
Cleavage:DB
Nterm
*
```

The first line of each block must start with the `Title:` keyword, followed by a text string that is used to identify the cleavage agent in forms and reports. The definition should be short and self-explanatory. It should only include alphanumeric characters and spaces. Internal spaces are significant.

Each block must also include a line starting with the keyword `Cleavage:` followed by a list of the residues that identify the cleavage site.

Optionally, a block can include a line starting with the keyword `Restrict:` followed by a list of the residues which prevent cleavage if present adjacent to the potential cleavage site.

Finally, the block must include either the keyword `Cterm` or `Nterm` to define whether cleavage occurs on the C terminal or N terminal side of the specified residues.

This syntax can be extended to support multiple cleavage specificities, enabling enzyme mixtures to be modelled, or mixed C-term and N-term cutters. This is achieved by appending zero-based index numbers in square brackets to the keywords `Cleavage, Restrict, Cterm,` and `Nterm`. For example:

```
Title:CNBr+Trypsin
Cleavage[0]:M
Cterm[0]
Cleavage[1]:KR
Restrict[1]:P
Cterm[1]
Independent:0
*
```

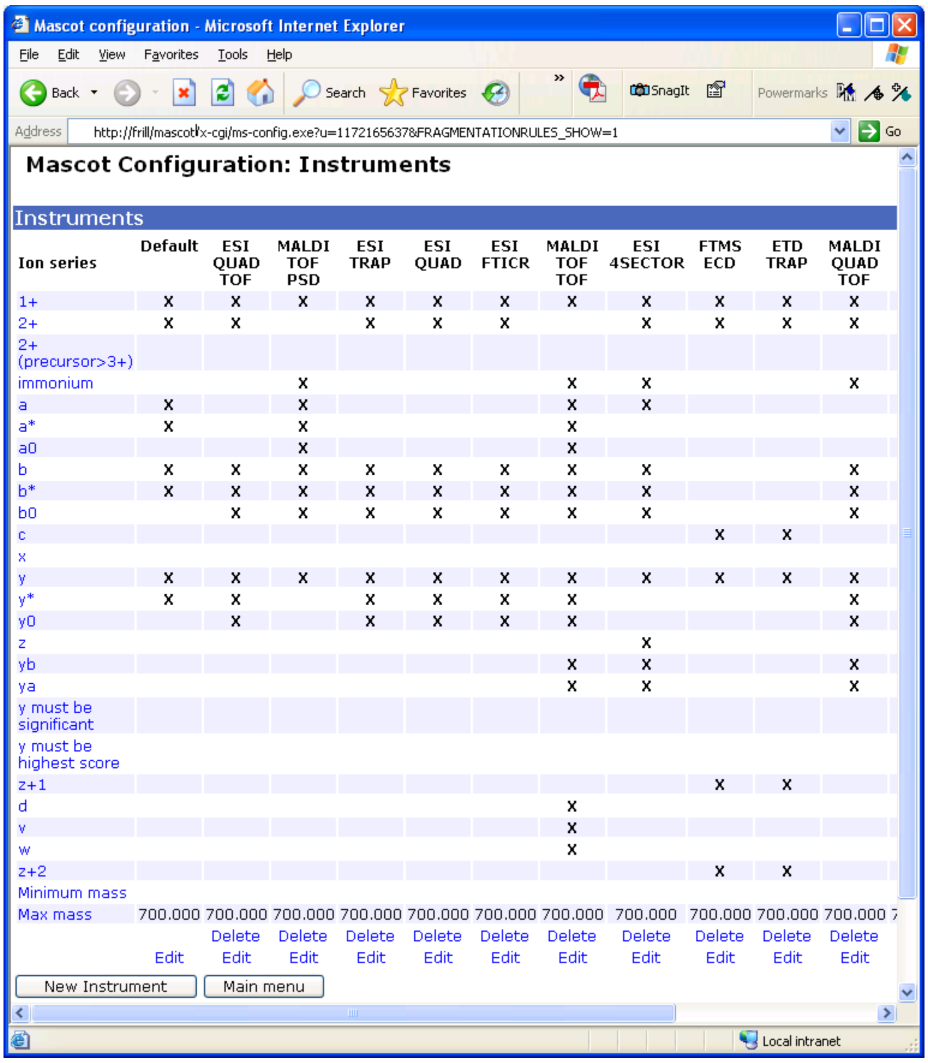
The use of index numbers is optional when only one specificity is defined, but required when there are multiple specificities, as in this example.

For a definition with multiple specificities, if the keyword `Independent` appears and is given a value of 1, this means that the specificities should be treated as if independent digests had been performed on separate sample aliquots and the resulting peptide mixtures combined. Thus, any given peptide will conform to the specificity of one cleavage type only. In the case of CNBr+Trypsin, if `Independent` was set to 1, you would not find any peptides resulting from cleavage after K or R at one end, and cleavage after M at the other. When `Independent` is omitted or given a value of 0, the specificities are combined, as if the reagents had been applied simultaneously or serially to a single sample aliquot. The keyword `Independent` does not take an index.

```
Title:semiTrypsin
Cleavage[0]:KR
Restrict[0]:P
Cterm[0]
SemiSpecific:1
*
```

If the keyword `SemiSpecific` appears and is given a value of 1, this means that any given peptide need only conform to the cleavage specificity at one end. The other end can result from non-specific cleavage. When `SemiSpecific` is omitted or given a value of 0, peptides are required to conform to the cleavage specificity at both ends. The keyword `SemiSpecific` does not take an index.

Instruments


Mascot configuration - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites SnagIt Powermarks
Address http://frill/mascot/x-cgi/ms-config.exe?u=1172165637&FRAGMENTATIONRULES_SHOW=1 Go

Mascot Configuration: Instruments

Instruments

Ion series	Default	ESI QUAD TOF	MALDI TOF PSD	ESI TRAP	ESI QUAD	ESI FTICR	MALDI TOF TOF	ESI 4SECTOR	FTMS ECD	ETD TRAP	MALDI QUAD TOF
1+	X	X	X	X	X	X	X	X	X	X	X
2+	X	X		X	X	X		X	X	X	X
2+ (precursor>3+)											
immonium			X				X	X			X
a	X		X				X	X			
a*	X		X				X				
a0			X				X				
b	X	X	X	X	X	X	X	X			X
b*	X	X	X	X	X	X	X	X			X
b0		X	X	X	X	X	X	X			X
c									X	X	
x											
y	X	X	X	X	X	X	X	X	X	X	X
y*	X	X		X	X	X	X				X
y0		X		X	X	X	X				X
z								X			
yb							X	X			X
ya							X	X			X
y must be significant											
y must be highest score											
z+1									X	X	
d							X				
v							X				
w							X				
z+2									X	X	
Minimum mass											
Max mass	700.000	700.000	700.000	700.000	700.000	700.000	700.000	700.000	700.000	700.000	700.000
		Delete	Delete	Delete	Delete	Delete	Delete	Delete	Delete	Delete	Delete
	Edit	Edit	Edit	Edit	Edit	Edit	Edit	Edit	Edit	Edit	Edit

The INSTRUMENT search parameter is used to select the set of ion series used for scoring MS/MS matches.

File format (fragmentation_rules)

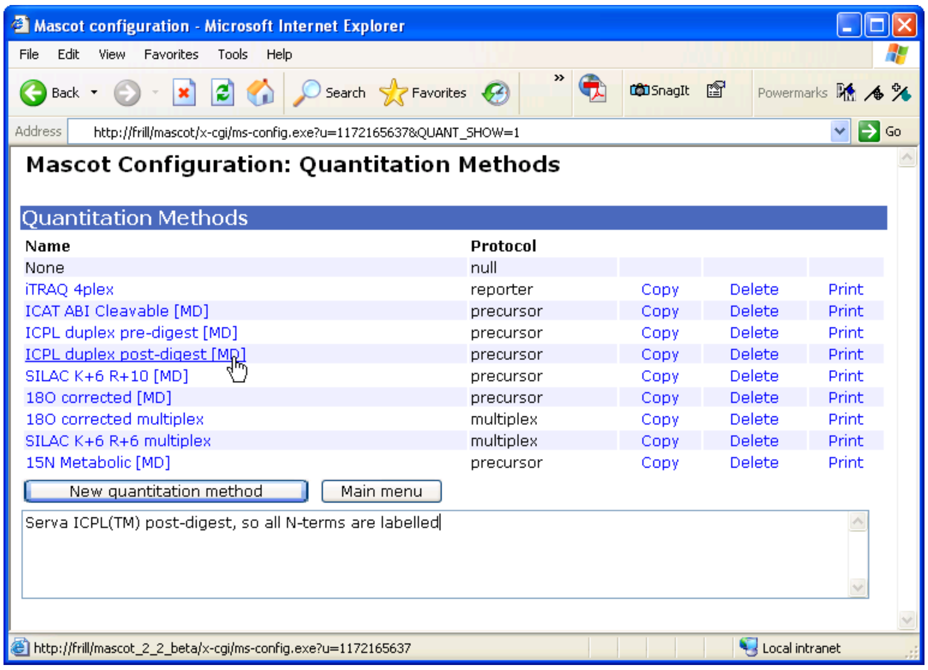
Each instrument is defined by a block of lines. Blocks are delimited from one another by a line containing an asterisk.

The first line of each block must start with the Title: keyword, followed by a text string that is used to identify the instrument in forms and reports. The definition should be short and self-explanatory. It should only include alphanumeric characters and hyphens. The following lines start with an integer, each of which represents an ion series or a rule to be included in the definition. Refer to the file header for a list of available integers. Anything following a hash (#) symbol is treated as a comment.

A block can also specify mass range limits for internal ions. The default range is 0 to 700 Da, and could be changed as in this example:

```
title:MALDI-QIT-TOF
1  # singly charged
4  # immonium
5  # a series
6  # a - NH3 if a significant and fragment includes RKNQ
7  # a - H2O if a significant and fragment includes STED
8  # b series
9  # b - NH3 if b significant and fragment includes RKNQ
10 # b - H2O if b significant and fragment includes STED
13 # y series
14 # y - NH3 if y significant and fragment includes RKNQ
15 # y - H2O if y significant and fragment includes STED
17 # internal yb < 700 Da
18 # internal ya < 700 Da
minInternalMass 200
maxInternalMass 1000
*
```

Quantitation


Mascot configuration - Microsoft Internet Explorer

Address: http://frill/mascot/x-cgi/ms-config.exe?u=1172165637&QUANT_SHOW=1

Mascot Configuration: Quantitation Methods

Quantitation Methods

Name	Protocol			
None	null			
iTRAQ 4plex	reporter	Copy	Delete	Print
ICAT ABI Cleavable [MD]	precursor	Copy	Delete	Print
ICPL duplex pre-digest [MD]	precursor	Copy	Delete	Print
ICPL duplex post-digest [MD]	precursor	Copy	Delete	Print
SILAC K+6 R+10 [MD]	precursor	Copy	Delete	Print
18O corrected [MD]	precursor	Copy	Delete	Print
18O corrected multiplex	multiplex	Copy	Delete	Print
SILAC K+6 R+6 multiplex	multiplex	Copy	Delete	Print
15N Metabolic [MD]	precursor	Copy	Delete	Print

New quantitation method | Main menu

Serva ICPL(TM) post-digest, so all N-terms are labelled

http://frill/mascot_2_2_beta/x-cgi/ms-config.exe?u=1172165637 — Local intranet

A detailed description of quantitation methods, the relevant Configuration Editor pages, and the underlying file, (*quantitation.xml*), is contained in the HTML help pages. Choose Help from the Mascot main menu bar and then choose Quantitation.

Database Manager

Database Manager is mainly described in the HTML help pages. Choose Help from the Mascot main menu bar and then choose Sequence Database Setup; Database Manager.

Configuration Options

This is a simple interface to the Options section of *mascot.dat*, which contains a variety of global settings. Reference material can be found below.

mascot.dat

Two sections of *mascot.dat*, Processors and Cluster, have no interface in either Database Manager or Configuration Options, and the only way to make changes is to edit *mascot.dat*.

Windows users should note that the path delimiters used in *mascot.dat* must always be forward slashes, never the backward slashes used at the command prompt. If sequence database files are not on a local disk drive, the remote drive must be mapped to a local drive letter. UNC path specifications cannot be used. Finally, spaces are not allowed in file or directory names. Hence:

`C:/InetPub/mascot/config/mascot.dat`	correct ✓
`C:\InetPub\mascot\config\mascot.dat`	wrong ✗
`\\matrix_nt_01\InetPub\mascot\config\mascot.dat`	wrong ✗
`//matrix_nt_01/InetPub/mascot/config/mascot.dat`	wrong ✗

General

`mascot.dat` is divided into sections. Each section starts with a unique keyword and ends with the keyword ‘`end`’.

Comments and blank lines can be used freely. A line which starts with the # character (pound in the US, hash in Europe) is a comment line.

Databases

Do not modify this section if you ever use Database Manager

```
Databases
.
.
.
# NCBInr c:/inetpub/mascot/sequence/NCBInr/current/
  NCBInr_*.fasta AA 1234 14 1 1 -1 0 0 6 7 0 8
SwissProt c:/inetpub/mascot/sequence/SwissProt/current/
  SwissProt_*.fasta AA 1234 15 1 1 -1 0 1 33 13 15 3
.
.
.
end
```

A line that is commented out with a # character at the start is an *inactive* database definition. Each line defines a database using the following 14 parameters:

1. Name: Each database must have a unique name. Ideally, the name should be short and descriptive. Note that these names are case sensitive, and much confusion can be caused by creating (say) Sprot *and* SPROT. The name does not need to be the same as or even similar to the filename of the actual FASTA file. Allowed characters are alphanumerics and _.-$%&()[]

2. Path: FASTA database files must be available locally. Mascot creates its compressed files in the same directory as the original FASTA file. The location of the FASTA file is defined in the Path field. This must be the fully qualified path to the FASTA file, with a wild card in the filename to allow incoming and outgoing database files with different version or date stamps to be present in the current directory simultaneously. The delimiters between directories must always be forward slashes, even if Mascot is running on a Windows system.

3. AA / NA: AA for an amino acid (protein) database and NA for a nucleic acid (DNA) database.

4. Obsolete: This parameter used to contain the approximate number of entries (sequences) in the database, used for progress reports during a search. The value is now just a place holder.

5. Obsolete: This parameter used to contain a unique identification number. The value is now just a place holder.

6. Mem map: Flag to indicate whether the database file should be memory mapped (1) or not (0). Database files should always be memory mapped. Unlike memory locking, this does not consume physical RAM.

7. Obsolete: This parameter (Blocks) must always be set to 1.

8. Threads: A Mascot search can use multiple threads. If you are running in cluster mode, 'Threads' is ignored. Otherwise, set to –1 to allow the number of threads to be determined automatically. To specify a fixed number of threads in non-cluster mode, set a value of 1 or more.

9. Mem lock: Flag to indicate whether a memory mapped database file should be locked in memory (1) or not (0). This setting is only relevant if column 6 contains a 1.

Memory mapped files can be locked in memory, but only if the computer has sufficient RAM. Having a database locked in memory means that it can never be swapped out to disk, ensuring there will never be a lag if the database files have to be read from disk. Of course, there also needs to be sufficient RAM for the operating system, (Windows consumes approximately 60 MB), anything from tens to hundreds of MB for each Mascot search, and space for any other applications which might be running.

If you try to lock databases into RAM when there isn't room, this will not be a major problem. The locking will fail, generate an error message, and Mascot will carry on regardless. A more serious problem is when there is just sufficient RAM to lock the databases, but none left over for searches or other applications. In this case, the whole system will slow down and the hard disk will be observed to be "thrashing". Eventually, the system is likely to hang or crash.

10. Local ref file: Flag to indicate whether a local reference file is available (1) or not (0). For certain databases, e.g. SwissProt, it is possible to have a local reference file, from which full text information can be taken for a 'Protein View' report.

11. AccessionParseRule: Index of the regular expression in the PARSE section that can be used to parse an accession string from a FASTA file title line.

12. DescriptionParseRule: Index of the regular expression in the PARSE section that can be used to parse a description string from a FASTA file title line.

13. AccessionRefParseRule: Index of the regular expression in the PARSE section that can be used to parse an accession string from a local full text reference file. If there is no local reference file, this value is ignored and can be set to 0.

14. Taxonomy: Index of the taxonomy rule block to be used to parse taxonomy information. If taxonomy information is not available, or is not to be used, this value should be set to 0.

PARSE

Do not modify this section if you ever use Database Manager

The PARSE section contains Basic Regular Expressions used to extract strings from various files.

```
PARSE
.
.
.
# For NCBI accession e.g.
RULE_6   ">\(gi|[0-9]*\)"
#
# For NCBI description - everything after the first space
RULE_7   ">[^ ]* \(.*\)"
#
.
.
.
end
```

The syntax of a standard Basic Regular Expression (BRE) is described in Appendix A. Rules defined in this section are referred to by means of their index number in two sections: Databases and WWW.

RULE_6, for example, looks for the “>” at the beginning of the title line. The string to be extracted is in backslashed parentheses: “gi|” then as many digits as possible. The match will stop when a non-digit is encountered, such as a pipe symbol or a space.

If you are not familiar with regular expressions, use the information in Appendix A to understand how the pre-defined rules in *mascot.dat* work.

A mistake in a rule called from the databases section may prevent Mascot from using the database concerned. Always use the Database Manager to configure and test new database definitions before they are brought on-line.

WWW

Do not modify this section if you ever use Database Manager

The WWW section defines where CGI scripts look for the information needed to compile a results report.

At least one line is required for each database, to define the source from which the sequence string of a database entry can be obtained. A second line can optionally define the source from which the full text report of an entry can be obtained. The syntax is very similar in both cases, independent of whether the information originates locally or on a remote system.

Sequence strings can always be retrieved locally, because the FASTA file must be present on a local disk. The Mascot utility *`ms-getseq.exe`* is normally used to retrieve a sequence string.

If full text for an entry is available locally and the database has been defined as including a ref file, (Column 10 in the Database section of *`mascot.dat`*), *`ms-getseq.exe`* can be used to retrieve the full text. Otherwise, a utility or URL must be identified which can accept an accession string and return the report text in a parseable format. An example of a suitable external URL for full annotation text is shown in the example for Trembl, below

Each line in the WWW section contains 5 columns:

```
WWW
.
.
.
Trembl_SEQ "8" "localhost" "80" "c:/inetpub/mascot/x-cgi/ms-
   getseq.exe Trembl #ACCESSION# seq"
Trembl_REP "23" "www.uniprot.org" "80"
   "/uniprot/#ACCESSION#.txt"
.
.
.
end
```

1. Identifier: An identifier constructed from the name of the database, an underscore character, and either the keyword SEQ or REP. Thus, `Trembl_SEQ` is the source for the sequence string of an entry in the database called Trembl.

2. Parse rule: The index of a rule in the PARSE section that can be used to extract the information required. Note that the rule for parsing a sequence string from `ms-getseq.exe` the same for all databases.

3. Host: The information source. For *`ms-getseq.exe`* or a similar local executable, this column should contain `localhost`. For a remote source, or a local source that will be queried as a CGI application, enter the hostname. (NB the word `localhost` is used to determine whether the application is a command line executable or a CGI application. If you want to specify a CGI application on the local server, just specify the hostname in some other way, for example `127.0.0.1`).

4. Port: The port number. This should be left at `80` unless another value is required to access a web server operating on a non-default port.

5. Path: A string containing the path to the executable and parameters, some of which are variables.

In the case of a command line executable, the parameters will generally be delimited by spaces. In the case of a CGI application, the parameters may be delimited from the executable by a question mark, and there must be no spaces within the parameter string. In general, spaces in URL's must be replaced by plus symbols, and non-alphanumeric characters should be URL encoded using the %nn notation.

A reminder to Windows users: Do not use backslashes as path delimiters, because these will be interpreted as escape characters.

Most parameters are entered as literal strings, with two exceptions: `#ACCESSION#` is a place holder that will be replaced by an actual accession string, `#FRAME#` is a place holder that will be replaced by the number of the reading frame used to translate a nucleic acid sequence. Obviously, this last parameter is only used with NA databases.

The syntax for calling *`ms-getseq.exe`* is described in Chapter 7. In the examples shown above, the full text report for Trembl is taken from an external URL because the full text file for Trembl is huge (40 GB). The default configuration for SwissProt uses a local full text reference file.

Processors

Mascot licensing is physical CPU or socket-based. For each CPU covered by the licence, Mascot will fully utilise up to 4 logical processors or cores.

If the number of processors available is the same as the number licensed, then it is best not to include a PROCESSORS section. You can include one, if you wish, but this may have a negative impact on system performance.

If the number of processors available is greater than the number licensed, you can use a PROCESSORS section to force specific cores to be used.

Logical processor (core) numbers generally start at 0, but see your computer documentation. The `ProcessorSet=` line specifies the complete set of logical processors (cores) to be used. Separate processor values with a comma The number in this list must be less than or equal to four times the number of physical CPU licensed, or the system will not run.

Following this, the processors to be used for each database are specified. These numbers must be a subset of the numbers in the ProcessorSet, and there must be the same number of values as the number of threads specified earlier in the database section. For example, if you had a 1 cpu licence and the physical processor had 6 cores, and you wanted to avoid using cores 0 and 1, you could specify this as follows:

```
PROCESSORS
ProcessorSet=2,3,4,5
SwissProt=2,3,4,5
end
```

The PROCESSORS section must be after the Databases section in *`mascot.dat`*, and `ProcessorSet=` must come before the other entries in this section.

Taxonomy

Do not modify this section if you ever use Database Manager

The syntax of the taxonomy blocks is fully described in Chapter 9.

Cluster

The syntax of the cluster block is fully described in Chapter 11.

UniGene

Do not modify this section if you ever use Database Manager

UniGene is an index created by automatically partitioning GenBank sequences into a non-redundant set of gene-oriented clusters, (http://www.ncbi.nlm.nih.gov/UniGene/). Each UniGene cluster is a list of the GenBank sequences, including EST's, which represent a unique gene. It is not an attempt to produce a consensus sequence. UniGene can be used to simplify the results of a Mascot search of dbEST.

An index file must be downloaded for each species of interest. For each species, the fully qualified path to the index file is associated with the species name:

```
UniGene
human C:/Inetpub/MASCOT/unigene/human/current/Hs.data
mouse C:/Inetpub/MASCOT/unigene/mouse/current/Mm.data
mosquito C:/Inetpub/MASCOT/unigene/mosquito/current/Aga.data
```

To add a UniGene report option to Mascot for a particular sequence database, add a line containing the name of the database followed by a list of the available species names:

```
EST_human human
EST_mouse mouse
EST_others mosquito
end
```

Options

The Options section is used for miscellaneous parameters, which are listed here in alphabetical order. If a parameter is shown with argument(s), these are the default(s) that apply if the parameter is missing.

```
AutoSelectCharge 1
```

Controls how MS/MS queries are treated when the CHARGE parameter specifies more than one charge state (e.g. 1+. 2+, and 3+). This is usually because no charge information was available for a query, so the search form defaults applied.

If set to 0, a query is generated for each charge state and these queries are searched and reported independently. This is the default setting because this was the behaviour in earlier versions of Mascot.

If set to 1, each charge state will be searched, but only the charge state that gets the highest scoring match is saved to the result file and reported. This is the recommended setting

Note that this switch only applies to MS/MS queries, (including tags). Independent queries are always generated if multiple charge states are specified for molecular mass queries.

```
CacheDirectory ../data/cache/%Y/%m
```

Cache files are created and to improve performance when viewing large search results. This option specifies the relative path from the cgi directory to the location for saving report cache files. The actual directory will be, for example, ../data/cache/2010/02/uwcuxlsxx3s524f4vnnz3btmni/ where the lowest level directory is an md5sum of the .dat filename, the size and last-modified date of the .dat file. The tokens are % followed by any of the conversion specifiers supported by the strftime function (http://www.cplusplus.com/reference/clibrary/ctime/strftime/). For example, %Y gets converted to the year as a decimal number including the century, %m to the month as a decimal number (range 01 to 12) and %d to the day of the month as a decimal number (range 01 to 31). The date used will be the last modified date of the .dat file (rather than the time that the search started). See also ResfileCache and ResultsCache

```
CentroidWidth 0.25
CentroidWidthCount 1000
```

CentroidWidth is the width in Daltons of the sliding window used for re-centroiding profile data. Must be a floating point number between 0 and 10. Re-centroiding is applied whenever the number of peaks in a single scan exceeds CentroidWidthCount

```
CompressTool_AA ../bin/ms-compress.exe $dbname $dbpath
CompressTool_NA ../bin/ms-compress.exe $dbname $dbpath
CompressTool_SL ../bin/NIST/lib2nistcl/lib2nistcl.exe -log9
  $logfile -msp2peplib $inputfile $outputdirectory
```

The command lines used by Mascot Monitor to compress input files of type AA Fasta, NA Fasta, and SL MSP.

```
DecoyTypeNoEnzyme 3
DecoyTypeSpecific 1
```

These parameters determine how decoy sequences are created for Mascot Auto-decoy searches. DecoyTypeSpecific applies to MS/MS searches using fully specific or semi-specific enzymes. DecoyTypeNoEnzyme applies to MS/MS searches with no enzyme. For PMF, random protein sequences are used, whatever the settings. For NA databases, the sequences are randomized before translation. Classifications are based on G. Wang, et al. (2009), "Decoy Methods for Assessing

False Positives and False Discovery Rates in Shotgun Proteomics", Anal Chem. 81(1):146-159. Values supported in Mascot 2.4 and later are:

> 1 Reverse the sequence of each protein entry.
>
> 3 For each protein entry, generate a random sequence of the same length, with the composition based on the average composition of the whole database. This is the default in Mascot 2.3 and earlier.
>
> 4 Digest each protein sequence into peptides, then generate a random sequence for each peptide, but keep the same terminal residues and don't introduce new cutting sites.

```
DisplayNonSignificantMatches 0
```

Settings are 0 = Display checked check box in format controls section of summary reports, 1 = Display unchecked check box, 2 = Display edit box to allow user to input a custom expect or score value.

```
EmailErrorsEnabled 0
EmailFromTextName
EmailFromUser
EmailPassword
EmailProfile
EmailService
EmailTimeOutPeriod 120
EmailUsersEnabled 0
ErrMessageEmailTo
MailTempFile C:/TEMP/MXXXXXX
MailTransport 2
MonitorEmailCheckFreq 300
SendmailPath /usr/lib/sendmail
```

Mascot can be configured to use email for two purposes:

1. When the search engine executes as a CGI application, email can be used to send the results of a search to a user who accidentally or deliberately disconnected before the search was complete. This facility can be enabled by setting `EmailUsersEnabled` to 1 or disabled by setting it to 0.
2. Serious error messages can be emailed to an administrator. This facility can be enabled by setting `EmailErrorsEnabled` to 1 or disabled by setting it to 0. Error messages that are considered serious are identified in the file *`errors.html`*. This file can be found in the root directory of the installation CD-ROM, and is displayed by clicking on the link ‘Error message descriptions’ at the top of the database status page.

A number of parameters are used to define how email should be sent:

`MailTransport` should be set to one of the following values:

> 0 for CMC
>
> 1 for MAPI
>
> 2 for sendmail
>
> 3 for Blat

`EmailService` is the service name (CMC only)

`EmailPassword` is the password (if any) required to log onto MAPI or CMC

`EmailProfile` is the MAPI profile name

`sendmailPath` is the path to sendmail, or an equivalent.

`EmailFromUser` is the name which will appear in the 'From' field of the email message.

`EmailFromTextName` will appear in the 'Title' field of the message.

If `EmailUsersEnabled` is set to 1, search results will be emailed to a user if their web browser does not respond within the number of seconds specified in `EmailTimeOutPeriod` following the completion of a search.

Email messages can be sent in batches at intervals specified by `MonitorEmailCheckFreq` (in seconds). `MailTempFile` is the name of the temporary file used to store email messages until they can be sent.

If `EmailErrorsEnabled` is set to 1, serious error messages will be emailed to `ErrMessageEmailTo`.

MAPI Configuration (Windows Only)

Set `MailTransport` to 1.

Set the `EmailPassword` to the password (if any) that is required to log onto MAPI.

Set the `EmailProfile` to the profile name used by MAPI. This can be found by opening the Windows Control Panel and clicking on Mail. (Depending on whether you have an 'internet mail only' or a 'corporate or workgroup' installation of MS-Outlook, you will have a list of either account names or profile names to choose from).

Sendmail Configuration (Linux Only)

Set `MailTransport` to 2.

Set the `EmailFromUser` parameter to the name that is required in the 'From' field of the email messages.

Set `EmailFromTextName` as the name of the server that is running mascot. For example setting `EmailFromUser` to www and `EmailFromTextName` to Mascot Server will result in emails from www (Mascot Server). The From field of the email will be www@www.your_domain.com.

Set `sendmailPath` as the path for the sendmail program, e.g. /usr/lib/sendmail

Set `MailTempFile` as the name of the file used to store email messages until they can be sent (must be the path followed by a filename in the form: MXXXXXX). This will create temporary files that begin with M followed by a unique number. Typically this parameter will be /var/tmp/MXXXXXX.

Blat Configuration (Windows only)

Blat is a free, easily installed mail program for Windows. For more information, visit:

http://www.blat.net/

Set `MailTransport` to 3.

Set the `EmailUserFrom` parameter to the name that is required in the 'From' field of the email messages.

Set `EmailFromTextName` as the name of the server that is running mascot. For example setting `EmailUserFrom` to www and `EmailFromTextName` to Mascot Server will result in emails from www (Mascot Server). The From field of the email will be www@www.your_domain.com.

Set `sendmailPath` as the fully qualified path (including drive letter) for the Blat program.

Set `MailTempFile` as the name of the file used to store email messages until they can be sent (must be in the form path/MXXXXXX). This will create a new temp file where the first letter will be an M and the next 6 characters will make up a unique number. Typically this parameter will be c:/temp/MXXXXXX

```
ErrorLogFile ../logs/errorlog.txt
GetSeqJobIdFile ../data/getseq.job
InterFileBasePath c:/inetpub/mascot/data (Windows)
/usr/local/mascot/data (Linux)
InterFileRelPath ../data
MascotCmdLine ../cgi/nph-mascot.exe
MascotControlFile ../data/mascot.control
MascotJobIdFile ../data/mascot.job
MascotNodeControlFile ../data/mascotnode.control
MonitorLogFile ../logs/monitor.log
SearchLogFile ../logs/searches.log
TestDirectory ../data/test
UniqueJobStartNumber 001234
```

These entries determine local paths (not URL's). `ErrorLogFile, MascotCmdLine, MonitorLogFile, SearchLogFile,` and `TestDirectory` are self-explanatory.

`GetSeqJobIdFile` contains the next available job number for the `ms-getseq.exe` utility. These numbers wrap around at 999 and do not appear in the search logs. If this file is deleted, the next job number will be reset to 1 and a new `jobId` file created automatically

Mascot output files are written to a path given by:

`InterFileBasePath/InterFileRelPath/`*yyyymmdd*`/F`*nnnnnn*`.dat`

Where *yyyymmdd* is the current ISO date, and *nnnnnn* is a sequential job number with a minimum of 6 digits. The path is split into a base path and a relative path as seen by the CGI scripts so that the search engine can pass a file path to (say) *`master_results.pl`* as:

`InterFileRelPath/`*yyyymmdd*`/F`*nnnnnn*`.dat`

`TestDirectory` contains the input files used by Monitor to test new sequence databases.

`MascotControlFile` contains critical internal parameters. This file must be memory mapped and locked to provide interprocess communication between different Mascot components. `MascotNodeControlFile` is a similar, additional file used in cluster mode

`MascotJobIdFile` contains the next available job number. If this file is deleted, the next job number will be initialised to the value given by `UniqueJobStartNumber`, and a new `jobId` file created automatically. NB `UniqueJobStartNumber` must never be set lower than 1000.

```
ErrTolMaxAccessions 0
```

The maximum number of database entries allowed for a manual error tolerant search. Default is 0, meaning no limit.

```
ExecAfterSearch_n flag:num[flag:num], title string, command
  string
```

Defines a command to be run after a search is complete. N is one or two digits in the range 1 to 10. The Mascot installer creates the following two entries which provide Percolator integration:

```
ExecAfterSearch_1 waitfor:0;logging:0, Creating percolator
  input, ../bin/ms-createpip.exe -i %resultfilepath -o
  %percolator_pip
ExecAfterSearch_2 waitfor:1; logging:1, Percolating,
  ../bin/percolator.exe $PercolatorExeFlags
```

The following flags may be specified:

flag	**num**	**description**
waitfor	0..10	The command should wait for completion of the command specified by num. A value of 0 means don't wait, equivalent to omitting the flag
logging	0..3	Messages are put into errorlog.txt 0 – no logging 1 – log successful commands (return code 0) 2 – log unsuccessful commands (return code not 0) 3 – log successful and unsuccessful commands
percolator	0..1	0 – no dependency on Percolator 1 – command should only be run if search fulfills criteria for running Percolator

The title string will be displayed in the search progress while the process is running. This string must not contain a comma

The command string can include literals and also the following tags, which will be substituted at run time:

Tag	**Replaced with**

%resultfilepath	Relative path from the cgi directory to the results file
%resultfilename	File name part of %resultfilepath
%percolator_pip	Relative path from the cgi directory to the Percolator input file
%percolator_decoy_pop	Relative path from the cgi directory to the Percolator output file for the decoy matches
%percolator_target_pop	Relative path from the cgi directory to the Percolator output file for the target matches
%session_id	The session identifier of the logged in user when Mascot Security is enabled.
%task_id	The task identifier assigned using client.pl when called from client applications.
$PercolatorExeFlags	See below

If the executable string in ExecAfterSearch_2 includes $PercolatorExeFlags, this is expanded as follows

```
%PercolatorExeFlags -j %percolator_pip -B
  %percolator_decoy_pop -r %percolator_target_pop
```

where %PercolatorExeFlags is the value of PercolatorExeFlags in mascot.dat. Note that -D 14 will be suppressed automatically if PercolatorUseRT is set to 0 or if the peak list that has been searched doesn't contain any retention times.

Paths to executables and any paths included as arguments should use forward slashes and should not include spaces

```
FeatureTableLength 30000
```

If a nucleic acid sequence is longer than 30000 bases, the protein view report will automatically switch to feature table mode and output the matches as a GenBank feature table. The threshold for switching to feature table mode can be altered using the parameter FeatureTableLength in the Options section of mascot.dat or by appending _featuretablelength=X to the protein view URL, where X is the length in bases.

```
FeatureTableMinScore
```

By default, only matches with significant scores ($p < 0.05$) are output. A different score threshold can be specified using the parameter FeatureTableMinScore in the Options section of mascot.dat or by appending _featuretableminscore=X to the protein view URL, where X is the score threshold.

```
ForkForUnixApache 0
```

If a user presses 'Stop' or goes to another page in their browser when a search is running, the intended behaviour is that the search should continue, and the user be emailed with their results. Always set ForkForUnixApache to 1 for Apache on Linux, so that *nph-mascot.exe* ignores PIPE signals, does a fork, the parent exits, and the child then ignores HUP signals. This setting only applies to Linux, it is ignored under Windows.

```
FormVersion 1.01
```

Mascot users may save search forms off-line, or submit searches using scripts or private forms. When the search engine is upgraded, there is the possibility that old scripts or forms may contain invalid or obsolete parameters. If a search is submitted to Mascot without a version number, or if the version number is lower than that specified by `FormVersion`, a warning will be included in the results file and in the master results report.

`GetSeqJobIdFile` see `ErrorLogFile`

```
ICATQuantitationMethod ICAT
```

For backward compatibility, if a search is submitted from an old client with ICAT=ON, then the specified quantitation method will be used.

```
IgnoreDupeAccessions EST_others
```

A comma separated list of database names. For any database in this list, don't check for duplicate accession numbers when creating the compressed files. A database should only be added to this list if it has a very large number of sequence which may causes the system to run out of memory when creating the compressed files.

```
IgnoreIonsScoreBelow 0.0
```

When a report is generated, any ions score lower than this value will be set to zero and ignored. The parameter is a floating point number, default 0.0. Values greater than 0 and less than 1 act as an expect value threshold, and the scores for any peptide matches with higher expect values are set to 0. This global default can be over-ridden on an individual report URL by appending &_ignoreionsscorebelow=X, where X is the cut-off value.

This setting is ignored unless DisplayNonSignificantMatches is set to 2

```
IntensitySigFigs 2
```

The precision of intensity values written to the result file.

`InterFileBasePath` see `ErrorLogFile`

`InterFileRelPath` see `ErrorLogFile`

```
IonsDecimalPlaces 2
```

Mascot calculates all masses to an accuracy of 1/65535 Daltons. The number of decimal places used to display fragment ion masses in reports can be altered by changing this value.

```
IteratePMFIntensities 1
```

Set this option to 0 to prevent selection of PMF values on the basis of their intensity.

```
LabelAll 0
```

Set this option to 1 to make the initial display in Peptide View one in which all peaks that match a calculated mass value are labelled.

```
LastQueryAscFile ../logs/lastquery.asc
SaveEveryLastQueryAsc 1
SaveLastQueryAsc 0
```

`SaveLastQueryAsc` is a flag which controls whether the most recent input file to Mascot (i.e. the MIME format file containing MS data and search parameters) should be saved to disk (1) or not (0). This can be a useful debugging tool when writing scripts or forms to submit searches to Mascot. If `SaveLastQueryAsc` is set to 1, the name of the file is determined by `LastQueryAscFile`. Each new search over-writes this file. NB `LastQueryAscFile` is a disk path, not a URL.

An additional debugging tool is provided by `SaveEveryLastQueryAsc`. If set to 1, the Mascot input file will be saved for any search that fails to complete because it generates a fatal error. The name of the output file follows the same naming convention as a normal Mascot result file, except for the additional suffix `.inp`. If a search goes to completion, this file is deleted as soon as the normal output file has been written to disk.

```
LibrarySearch
```

The command line used for library searches. Default is:
`../bin/NIST/mspepsearch/MSPepSearch.exe m G $peptidetoleranceunit $peptidetolerance $iontoleranceunit $iontolerance /LIB $libname /INP $inputfile /OUTTAB $outputfile /HITS 100 /MinMF 0 /NumCompared /OutPrecursorMz /OutDeltaPrecursorMz /OutSpecNum`

```
LogoImageFile ../images/88x31_logo_white.gif
```

This is the URL of the Matrix Science logo, used at the top of a search progress report. You can customise this by substituting the URL of your own logo. For optimum appearance, the image should be 88 pixels wide and 31 pixels high.

`MailTempFile` see `EmailErrorsEnabled`

`MailTransport` see `EmailErrorsEnabled`

`MascotCmdLine` see `ErrorLogFile`

`MascotControlFile` see `ErrorLogFile`

`MascotJobIdFile` see `ErrorLogFile`

```
MascotMessage
```

A text string to be displayed ahead of the progress reports when a search is run

```
MassDecimalPlaces 2
```

Mascot calculates all masses to an accuracy of 1/65535 Daltons. The number of decimal places used to display peptide mass values in reports can be altered by changing this value.

```
MaxAccessionLen
```

Obsolete

```
MaxConcurrentSearches 10
```

This parameter limits the maximum number of concurrent searches so as to avoid overloading the Mascot server. Default is 10

```
MaxDatabases 64
```

The maximum number of concurrently active sequence databases. Increasing this value uses more RAM, so don't set unnecessarily high. There is no upper limit to this value. You need to restart the Mascot service after changing this value.

```
MaxDescriptionLen 100
```

Description text parsed from the FASTA title line will be truncated at this number of characters. (Note: There is no need to recompress a database if this parameter is changed).

```
MaxEtagMassDelta 1770
MinEtagMassDelta -130
```

In an error tolerant tag search with a fully specific enzyme, these values set the limits on the amount the mass is allowed to increase (MaxEtagMassDelta) or decrease (MinEtagMassDelta) in order to reach the first available cleavage point.

```
MaxEtVarMods 2
```

The maximum number of variable mods allowed in the first pass of an automated error tolerant search (global default, can be over-ridden for a group in security)

```
MaxNumPeptides
```

The maximum number of peptides that can be expected from the enzymatic digest of a single entry. The default is MaxSequenceLen/4. For Mascot 2.5 and later, this is a soft limit and more memory will be allocated if required.

```
MaxPepNumVarMods 5
```

The maximum number of different variable mods allowed in a single peptide match

```
MaxQueries 10000
```

The maximum number of MS/MS spectra allowed in a single search. Note that the maximum number of mass values in a PMF is hard-coded to 1000

```
MaxSearchesPerUser 0
```

Sets the maximum number of concurrent searches from a single IP address. A value of 0 means no limit. (global default, can be over-ridden for a group in security)

```
MaxSequenceLen 50000
```

The maximum length of a database entry in characters, (bases for NA or residues for AA). The default is 50,000. The length of the longest sequence in a database can be found in the *.stats file, created by Mascot Monitor when the database is compressed. The larger the value of MaxSequenceLen, the more memory mascot uses. So, if you need to increase it, make it just a little greater than the length of the longest sequence. On a 32 bit system, try not to exceed 3 million, because searches may run slower than normal. If you are trying to search an assembled genome, you might want to consider searching shorter sequences instead, such as a database of the contigs.

```
MaxVarMods 9
```

The maximum number of variable mods allowed for an MIS search (global default, can be over-ridden for a group in security). Value is an integer in the range 0 to 32

```
MinPeaksForHomology 6
```

For an MS/MS search, a homology threshold will not be reported if the number of peaks in a spectrum is less than this value

```
MinPepLenInPepSummary 7
```

In a Peptide Summary report, two proteins are reported as distinct matches if the peptide matches to one protein are not identical to or a sub-set of the peptide matches to the other protein. Since matches to very short peptides are usually random, peptides shorter than `MinPepLenInPepSummary` are not considered in this comparison.

```
MinPepLenInSearch 7
```

Peptides shorter than MinPepLenInSearch are rejected during the search. Matches to very short peptides are meaningless because a 2-mer or 3-mer can occur in almost every entry in a database. If such matches are allowed in the peptides section, it can cause serious bloating of the result file.

```
MonitorEmailCheckFreq see EmailErrorsEnabled
```

```
MonitorLogFile see ErrorLogFile
```

```
MonitorPidFile monitor.pid
```

The name for the file that holds the process ID number for *`ms-monitor.exe`*. Default is `monitor.pid`.

```
MonitorTestTimeout 1200
```

A time-out can be applied to the test searches used to validate a new database. If the test search on a new database does not produce a valid result within the number of seconds specified by `MonitorTestTimeout`, the problem is assumed to be with the new database, and the exchange process is halted.

```
MoveOldDbToOldDir 1
```

After a successful database swap, the old Fasta file and old reference file (if any) are moved to the *`../old`* directory unless this parameter is present and set to 0. Note that, if set to 0, the old files are not deleted. Some other application must take care of this or there will be problems next time Monitor starts up.

```
Mudpit 1000
```

Obsolete, see MudpitSwitch

```
MudpitSwitch 0.001
```

Mascot has two ways to calculate protein scores in a Peptide or Select summary report. Standard scoring is used when the ratio between the number of queries and the number of database entries, (after any taxonomy filter), is small. The standard score is the sum of the ion scores after excluding duplicate matches and applying a small correction. Protein score calculation switches to large search mode when the ratio between the number of queries and the number of database entries, (after any taxonomy filter), exceeds the value specified by MudpitSwitch. Only those ions scores that exceed one or both significance thresholds contribute to the score, so that low scoring, random matches have no effect. The global default can also be over-ridden on an individual report URL by appending &_server_mudpit_switch=X, where X is the ratio between the number of queries and the number of database entries, (after any taxonomy filter).

```
NoResultsScript ../cgi/master_results.pl
ProteinFamilySwitch 300
ResultsFullURL ###URL###/cgi/master_results.pl
ResultsFullURL_2 ###URL###/cgi/master_results_2.pl
ResultsPerlScript ../cgi/master_results.pl
ResultsPerlScript_2 ../cgi/master_results_2.pl
```

These are URL's (not disk paths) for the scripts to be called by the search engine at the completion of a search. A successful search calls `ResultsPerlScript` if the number of queries is less than `ProteinFamilySwitch` otherwise `ResultsPerlScript_2`. A search that didn't find any hits calls `NoResultsScript`.

The `ResultsFullURL` and `ResultsFullURL_2` are used when a link to the search results is emailed to a user. Since the email will probably be received on another system, the link needs to have the full URL including the Web server hostname. `###URL###` is replaced by the server URL during installation

```
NTIUserGroup Users
NTMonitorGroup Administrators
```

Under Windows, the Mascot service is generally run using the 'Local System' account. It has to create, write and read the memory mapped files. The CGI scripts (such as *nph-mascot.exe*) are run by the Web server, and will be run using a different user name with different permissions from the service. These programs also need to be able to read and write to these files. For example, with the Microsoft Web server (IIS), a new user with the name IUSR_<name_of_pc> is created when the server is installed, and the scripts are run using this user name. The installation program sets these values appropriately. Other Web servers may use different user names, with different permissions.

`NTIUserGroup` is the name of a group that the user name of the process to run CGI scripts belongs to. `NTMonitorGroup` is the name of the local Administrators group.

If not using IIS, check the documentation that comes with the server to find out which user name is used for running scripts, then from the start menu, choose, Programs, administrative tools (common), and User Manager. Double click on the user name, and press the groups button to find out which groups this user name belongs to. This is the name to put in *mascot.dat* for `NTIUserGroup`.

Failure to put the correct group name will generally result in one of two error messages:

Failed to open memory mapped file <filename>. Error: access denied

or

Failed to create memory map for <filename>. Error Access denied

After changing either of these entries, the Mascot service will need to be stopped, (from the start menu, choose *Programs; Mascot; config; Stop Mascot service)*. All compressed database files must be deleted. Then the Mascot service can be re-started (*Programs; Mascot; config; Start Mascot service*).

```
Percolator 0
PercolatorFeatures mScore, lgDScore, mrCalc, charge, dM,
  dMppm, absDM, absDMppm, isoDM, isoDMppm, mc, varmods,
  totInt, intMatchedTot, relIntMatchedTot
PercolatorMinQueries 100
PercolatorMinSequences 100
PercolatorUseProteins 0
```

```
PercolatorUseRT 0
PercolatorExeFlags -i 10 -D 14 -v 0
```

Set `Percolator` to 1 if percolated results should be opened by default, 0 otherwise. `PercolatorFeatures` specifies the list of features used by Percolator. To see the list of available features, run *`ms-createpip.exe -help`*. Percolator will only be run if the number of queries in the search is at least `PercolatorMinQueries` and the number of entries in the sequence database is at least `PercolatorMinSequences`. Percolator will use the assignment of proteins to peptides as a feature if `PercolatorUseProteins` is set to 1. This can have undesirable results and should be used with great care. This flag is not supported in the current release. Percolator will use the retention times of peptides as a feature if `PercolatorUseRT` is set to 1. `PercolatorExeFlags` is used to specify the Percolator command line arguments with the exception of the file path arguments –j –B –r. If the string includes the argument `-D num`, this will be removed unless `PercolatorUseRT` is set to 1

```
PrecursorCutOut -1,-1
```

The precursor peak can often have very high intensity relative to the fragment peaks, which may give rise to spurious fragment ion matches. It is usually best if the precursor is removed before the search.

With the default arguments of –1,–1, a smart filter is created. This removes peaks within the fragment ion tolerance window about each of the precursor isotope peaks. The number of isotopes is assumed to be as follows:

```
Mr     Number
< 10003
1000 - 1999 4
2000 - 2999 5
3000 - 3999 6
4000 - 4999 7
5000 - 5999 8
6000 - 6999 9
> 700010
```

So, if the precursor m/z was 800, the charge was 2, and fragment ion tolerance was +/– 0.1 Da, the filter would remove 4 notches of width

```
m/z 800.0 +/- 0.1
m/z 800.5 +/- 0.1
m/z 801.0 +/- 0.1
m/z 801.5 +/- 0.1
```

At first sight, this may seem a strange mix of m/z and Da. The reason is that we need to avoid matches from 1+ fragment ions, whatever the charge on the precursor.

If the arguments are anything other than –1,–1, a single notch is used where the first argument is the mass offset of the beginning of the notch and the second value is the mass offset of the end of the notch. For the precursor in the last example, if the arguments were –1,4 then the notch would run from m/z 799.5 to m/z 802.0. However, if the precursor charge was 1, then the notch would be from m/z 799 to m/z 804.

The mascot.dat setting can be over-ridden in a search by using the search parameter CUTOUT. Note that the peaks removed by this filter are not recorded in the result file, so cannot be recovered by changing this parameter in a repeat search.

```
ProteinFamilySwitch see NoResultsScript
```

```
ProteinsInResultsFile 2
```

Determines the number of protein title lines saved to each results file.

1. As in Mascot 1.7 and earlier, only proteins that appear in the Summary section will appear in the Proteins section
2. Include proteins with at least one top ranking peptide match to a peptide of length greater than or equal to `MinPepLengthInPepSummary`
3. Include all proteins

```
proxy_password
proxy_server
proxy_username
ProxyType Auto
```

These entries support a proxy server between the Mascot server and the outside world. A typical entry might be

```
proxy_server http://our-cache:3128
```

If there is no `proxy_server` entry, scripts will look for proxy information in the server environment. The `proxy_username` and `proxy_password` parameters are only required if the proxy server requires authentication. Remote host authentication should be included directly in the URLs specified in mascot.dat. e.g. `http://username:password@hostname/`

Allowed values for ProxyType are:

None	No proxy server will be used.
Registry	Windows only
Specify	The proxy server must be specified using proxy_server
WPAD	Windows only
Environment	Proxy settings will be loaded from the system environment
Auto	The proxy server will be discovered automatically. This is the default. On Windows the order is Registry, then WPAD and finally the magical system default proxy. On Linux, the order is Environment, Specify, and finally None.

```
RemoveOldIndexFiles 1
```

After a successful database swap, the compressed files in the current directory are deleted unless this parameter is present and set to 0

```
ReportBuilderColumnArrangement
```

Set the column arrangement at the given index. Column arrangements are used by Report Builder (introduced in Mascot 2.4) to provide a default list of columns to show. These can be selected from a dropdown list in the report. Each column arrangement is of the form “Name:[columns]” where Name is the column arrangement name (e.g. Standard) and [columns] is a comma-separated list of column names, as used by Report Builder. The following is the standard list of column names, available in every report:

```
family
member
db
acc
score
mass
matches
matches-sig
sequences
sequences-sig
empai
frame
desc
```

Frame will not be shown in the report if the search is against a proteindatabase. Quantitation methods add additional column names, but these are generated from the quantitation ratio names. The easiest way to create a column arrangement is to arrange the columns in Report Builder, then "export" the arrangement as a string.

```
ReportNumberChoices 5,10,20,30,40,50
```

If present, this list will define the choices provided in the search form ‘Report top’ drop down list.

```
RequireBoldRed 0
```

If this flag is set to 1, only protein matches which have one or more ‘bold red’ peptide matches will be listed in a peptide summary report. That is, proteins that include at least one top ranking peptide match that has not already appeared in the report. This global default can be overridden on an individual report URL by appending &_requireboldred=X, where X is 0 or 1.

```
ResfileCache master_results.pl, master_results_2.pl,
  peptide_view.pl, protein_view.pl, export_dat.pl,
  export_dat_2.pl, ms-createpip.exe, MSAnatomiser.class,
  mi_getpeaklist.pl, msms_gif.pl, nph-mascot.exe, ms-
  searchcontrol.exe

ResultsCache master_results.pl, master_results_2.pl,
  protein_view.pl, export_dat.pl, export_dat_2.pl, ms-
  createpip.exe, MSAnatomiser.class, mi_getpeaklist.pl, nph-
  mascot.exe, ms-searchcontrol.exe
```

Comma, space or tab delimited string of scripts and applications that will use cache files to speed up access to the results files. To prevent the use of the cache for a particular script, remove it from this list. There are two sets of cache files, one for the results file, independent of any particular report format, controlled by ResfileCache, and one for each combination of summary report format settings, controlled by ResultsCache. See also CacheDirectory.

```
ResultsFileFormatVersion
```

If present, and the argument is 2.1, the result file format will be “2.1 compatible”. That is, no xml sections. No other arguments are supported at this time.

`ResultsFullURL` see `NoResultsScript`

`ResultsFullURL_2` see `NoResultsScript`

`ResultsPerlScript` see `NoResultsScript`

`ResultsPerlScript_2` see `NoResultsScript`

```
ReviewColDisplay 1,1,1,1,1,0,0,1,1,1,1,1,1,0,1,1
```

Sets whether a column should be displayed (1) or collapsed (0) in ms-review.exe.

```
ReviewColWidths 7,8,8,27,30,120,32,25,6,13,4,4,6,16,7,150
```

This sets the widths of the columns in ms-review.exe.

`SaveEveryLastQueryAsc` see `LastQueryAscFile`

`SaveLastQueryAsc` see `LastQueryAscFile`

```
SaveSpectralLibraryFiles 0
```

If set to 1, then temporary files used for the spectral library search are left on the server for debugging. Input files have extension MGF and output files have extension TSV.

```
ScoreThresholdForAuto
```

Deprecated, use `SigThreshold`.

```
SearchControlLifetime 7200
SearchControlSaveE 0
```

Obsolete.

`SearchLogFile` see `ErrorLogFile`

`SendmailPath` see `EmailErrorsEnabled`

```
SelectSwitch 1000
```

If the number of queries in an MS/MS search is less than or equal to this number, the default report is the Peptide Summary. If it is greater than this number, the default report is the Select Summary.

```
   SeparateLockMem 0
```

Only required for 32-bit versions if the total amount of memory to be locked is greater than 2 GB (or lower if some system limit is set). Setting this value to 1 indicates that *ms-monitor.exe* will run a separate program (*ms-lockmem.exe*) that will lock the memory blocks. A value greater than 1 specifies the block size in Mb. For example, if there is a 1.5 GB `*.s00` file, and this parameter is set to 750, then two instances of *ms-lockmem.exe* will be run.

```
   ShowAllFromErrorTolerant 0
```

Standard behaviour for the result report of a manual error tolerant search is to show only those matches that satisfy two criteria: (i) the score must be at least as high as the match for the same query in the original 'parent' search, (ii) the score equals or exceeds the identity threshold for the same query in the original 'parent' search. Setting `ShowAllFromErrorTolerant` to 1 causes all matches to be displayed. This global default can be overridden on an individual report URL by appending &_showallfromerrortolerant=X, where X is 0 or 1.

```
   ShowSubSets 0
```

If this is set to 1, under each protein match in a peptide summary report, matches to proteins that contain a sub-set of the same peptides will also be listed. This was the default behaviour in version 1.6 and earlier. If this flag is set to 0, which is now the default, the sub-set matches will not be shown. Values between 0 and 1 represent the fraction of the protein score of the primary hit that a subset hit can lose and still be listed. For example, if ShowSubSets is 0.2, and the primary hit has a protein score of 200, sub-set hits with scores of 160 or more will be listed.

If multiple entries contain the full set of peptides, they are all displayed, whatever the setting of this parameter. This global default can be overridden on an individual report URL by appending &_showsubsets=X, where X is 0 or 1.

```
   SigThreshold 0.05
```

Significance threshold used in result reports, default 0.05. Valid range is 1 to 1E-18. This global default can be overridden on an individual report URL by appending &_sigthreshold=X, where X is the significance threshold.

```
   SiteAnalysisMD10Prob 0.1
```

Used to calculate relative probabilities of modification assignments in Peptide View. It defines the factor in probability that a peptide score difference of 10 corresponds to. The default is 0.1, which means a score difference of 10 corresponds to a factor of 10 in probability. Similarly, 0.05 corresponds to a factor of 20.

```
   SortUnassigned scoredown
```

In a peptide summary report, peptide matches that are not assigned to protein hits are initially sorted by descending score (scoredown). Alternatives for `SortUnassigned` are ascending query order (queryup) and descending intensity order (intdown). This global default can be overridden on an individual report URL by appending &_sortunassigned=X, where X is scoredown, queryup, or intdown.

```
   SpectrumViewerDefaultColourScheme screen
```

This would specify that the colour scheme for the SVG spectrum viewer in Peptide View is specified in another option called SpectrumViewerColourScheme_screen

```
   SplitDataFileSize 10000000
```

Large searches are divided into 'chunks', and no single chunk can exceed this number of bytes – default 10 Mb. When a search is divided into chunks, protein and peptide match data are no longer written to the summary section of the result file. This means that a Protein summary report cannot be generated.

```
   SplitNumberOfQueries 1000
```

Large searches are divided into 'chunks', and no single chunk can exceed this number of queries – default 1000. When a search is divided into chunks, protein and peptide match data are no longer written to the summary section of the result file. This means that a Protein summary report cannot be generated.

In cluster mode, this value is also used to determine how the search should be distributed among the nodes. If the search contains more queries (ms-ms spectra) than this value, the queries are distributed among the nodes and each node searches the complete database. If the search contains less queries than this value, then all queries are searched on every node, but each node searches just a part of the database.

```
   StoreModPermutations 1
```

If set to 0, only the highest scoring permutation of variable modifications for each unique peptide sequence is retained in the list of the top 10 ions scores. If set to 1, then different permutations of variable modifications are treated as independent matches, creating the possibility that all 10 top ions scores correspond to the same primary sequence. Default is 1.

```
   SVGSpectrumSwitch 2000
```

If the number of peaks for the query is greater than this value, it will switch from the interactive SVG spectrum to the old GIF image. The SVG based viewer is not very responsive with a huge number of peaks, so the default value of 2000 is suitable for most systems. Maximum setting 100000.

```
   TargetFDRPercentages 0.1, 0.2, 0.5, 1+, 2, 5
```

Choices available for the FDR drop down list in the Protein Family Summary report of an auto-decoy search. Each item in the list is a percentage. The + symbol specifies the default setting of the control, 1% in this case.

```
TaxBrowserURL
```

(No default). The URL used in reports to retrieve taxonomy information for a Protein View report. By default, this points to the NCBI. If you don't want to send such queries out to the internet, the URL can be replaced by a call to the ms-gettaxonomy.exe utility:

```
TaxBrowserUrl ../x-cgi/ms-
  gettaxonomy.exe?4+#DATABASE#+#ACCESSION#
```

`TestDirectory` see `ErrorLogFile`

`UniqueJobStartNumber` see `ErrorLogFile`

```
UnixDirPerm 777
```

Specify the Linux permissions for the 'daily' result file directories. For example, 775 makes each directory world readable but not writeable. This option provides more fine grained control than UnixWebUserGroup

```
UnixWebUserGroup
```

This entry, if present, will restrict access to the files created by *`ms-monitor.exe`*, and hence improve system security. The `UnixWebUserGroup` is the number of the group used by the web server to run CGI programs. With Apache, the group name will generally be nobody, and you will need to ascertain the group number from the group file. For other Web servers, check the documentation that comes with the server to find out which user name is used for running CGI programs.

A value of `-2` can be used if the same user name is used to run Web server scripts as runs *`ms-monitor.exe`*. In this case, The files created by *`ms-monitor.exe`* will not be world accessible, and 'chown' is not used on the files to change ownership.

Failure to put the correct group name will generally result in one of two error messages:

Failed to open memory mapped file <filename>. Error: access denied

or

Failed to create memory map for <filename>. Error Access denied

```
UseHTTPProxyForFTP 0
```

Set to 1 to allow making FTP requests through the HTTP proxy. Note that the HTTP proxy server must explicitly support such "tunnelling"

```
Vmemory -1
```

Obsolete.

Cron

Do not modify entries in this section used by Database Manager (dbman_*.pl)

Database Manager uses the information in this section to schedule database updates.

```
Cron
CronEnable 1
Logfile ../logs/cron.log
Logging 3
0-59 * 1-31 * * /usr/local/mascot/bin/dbman_process_tasks.pl
end
```

`CronEnable` is set to 1 to enable cron functionality, 0 to disable.

`Logfile` specifies the path to the log for recording cron events, `Logging` controls the verbosity:

```
0 - No logging
1 - Log successful commands (return code 0)
2 - Log unsuccessful commands (return code not 0)
3 - Log successful and unsuccessful commands
```

The remaining lines in this section simulate a crontab file. Each line contains six fields, separated by spaces or tabs. The first five are integer patterns that specify the following: minute (0-59), hour (0-23), day of the month (1-31), month of the year (1-12), day of the week (0-6 with 0=Sunday). Each of these patterns may be an asterisk (meaning all legal values), a range of integers or a list of comma separated integers.

An element is either a number or two numbers separated by a minus sign (meaning an inclusive range). Note that days may be specified in two different ways (day of the month and day of the week). If both are specified as a list of elements, both are adhered to. For example,

```
0 0 1,15 * 1
```

would run a command on the first and fifteenth of each month, as well as on every Monday. To specify days by only one field, the other field should be set to * (for example, `0 0 * * 1` would run a command only on Mondays).

The sixth field is a string that is executed by the shell (command prompt) at the specified times. The string must be on a single line. The entire string, up to the end of the line, is passed to the command prompt for execution. The part of the string up to the first space must be the fully qualified path to an executable. The remainder of the line will be passed to the command as parameters.

Log files

Mascot maintains several log files, which are described below. When trouble-shooting, it can be useful to inspect the web server log files, also. Errors in Perl scripts, for example, will be appear in the web server error log, not the Mascot error log.

Error Log

All errors are logged to *logs/errorlog.txt*. This is may be the only place to find a fatal error message resulting from a major configuration problem.

Examples of typical error messages are shown below. A comprehensive list of all Mascot error messages can be found in the file *errors.html*, in the root directory of the Mascot CD-ROM.

```
Error [M00088 - Job 2636 - X00123:file-upload]
   - Thu Mar 11 10:59:30 2009
   - Invalid command/mass at line 1 of your query.
  Line is where am I?

Error [M00034 - Job 2638 - X00251:modifications]
   - Thu Mar 11 10:59:59 2009
   - Modification conflict: Both Carbamidomethyl (C) and
  Carboxymethyl (C) modify the same residue

Error [M00133 - Job 2639 - X00938:www]
   - Thu Mar 11 11:00:21 2009
   - Peptide mass of -1234 is too small. The minimum mass
  allowed is 30
```

Searches Log

Every Mascot search is listed in *logs/searches.log*. The Mascot Review utility provides a web browser interface to this file, displaying filtered and sorted listings of searches. Mascot Review is described in Chapter 7.

Alternatively, the file can be opened in a spreadsheet program. The file consists of 14 columns, delimited by tabs. Row 1 contains column titles. An example of a single entry is shown below:

```
2633 \t 185 \t NCBInr \t JSC \t JSC@gmail.com \t  \t
../data/20090311/F002633.dat \t Thu Mar 11 09:10:36 2009 \t 17 \t User
read res \t 1 \t PMF \t Yes \t 192.168.42.4
```

(Tabs indicated by \t for clarity). The individual columns contain the following information:

Column 1: Mascot job number. Job numbers are allocated sequentially, but will appear in the log in the order in which searches are completed. If the submitted search contained an error which prevented the search starting, there will be no entry in `searches.log`, but there should be an entry in `errorlog.txt`.

Column 2: Process ID

Column 3: Sequence Database searched

Column 4: User name. User names are required by the (JavaScript) search forms, but not by the search engine, so this field may be empty. If an entry logs utility program activity, rather than a search, this field contains the name of the utility, e.g. TESTPARSE or GETSEQ.

Column 5: User email address. User email addresses are required by the (JavaScript) search forms, but not by the search engine, so this field may be empty.

Column 6: Search title. Empty if none supplied.

Column 7: Relative path to Mascot search results file

Column 8: Start time in the format illustrated in the example above.

Column 9: Duration in seconds

Column 10: Completion Status, normally "`User read res`". If `EmailUsersEnabled` is set to 1, and the user disconnected before the search was complete, this entry would read "`user emailed`".

Column 11: Job Priority. Not currently implemented

Column 12: Type of search: PMF, SQ, or MIS

Column 13: Enzyme: Either `yes` (if user selected an enzyme) or `no` (if user selected enzyme type None).

Column 14: User IP address

Monitor Log

Mascot Monitor activity, such as sequence database exchange, is logged to *`logs/monitor.log`*. The following extract shows a typical example of the contents:

```
Fri Apr 20 17:21:28 2012 - ---------------------------------- ms-monitor
   2.4.0 started
Fri Apr 20 17:21:28 2012 - Locked memory for file ../data/mascot.control
Fri Apr 20 17:21:28 2012 - Waiting for valid licence
Fri Apr 20 17:30:28 2012 - Licensed to: Edman University  (XQ5P-TFRR-3APW-
   FB33-7H6X)
Fri Apr 20 17:30:28 2012 - Starting up                          to Checking
   that Mascot Nodes exist
Fri Apr 20 17:30:28 2012 - Checking that Mascot Nodes exist      to Loading DB
   information
Fri Apr 20 17:30:28 2012 - Loading DB information               to Started up
   successfully
Fri Apr 20 17:30:29 2012 - SwissProt0 Not in use                 to Preparing
   to run 1st test
Fri Apr 20 17:30:29 2012 - SwissProt0 Preparing to run 1st test   to Waiting
Fri Apr 20 17:30:30 2012 - SwissProt0 Waiting                    to About to
   compress files
Fri Apr 20 17:30:30 2012 - SwissProt0 About to compress files     to Creating
   compressed files
Fri Apr 20 17:30:33 2012 - Creating compressed files from
   /usr/local/mascot/sequence/SwissProt/current/SwissProt_2012_03.fasta
Fri Apr 20 17:30:33 2012 - Creating compress file
   /usr/local/mascot/sequence/SwissProt/current/SwissProt_2012_03.i00
Fri Apr 20 17:30:33 2012 - Creating compress file
   /usr/local/mascot/sequence/SwissProt/current/SwissProt_2012_03.s00
Fri Apr 20 17:30:33 2012 - Creating compress file
   /usr/local/mascot/sequence/SwissProt/current/SwissProt_2012_03.a00
Fri Apr 20 17:30:33 2012 - Creating compress file
   /usr/local/mascot/sequence/SwissProt/current/SwissProt_2012_03.t00
Fri Apr 20 17:30:33 2012 - Creating compress file
   /usr/local/mascot/sequence/SwissProt/current/SwissProt_2012_03.stats
```

```
Fri Apr 20 17:32:26 2012 - SwissProt0 Creating compressed files   to Finished
   compressing files
Fri Apr 20 17:32:26 2012 - SwissProt0 Finished compressing files  to Running
   1st test
Fri Apr 20 17:32:33 2012 - SwissProt0 Running 1st test            to First
   test just run OK
Fri Apr 20 17:32:33 2012 - SwissProt0 First test just run OK      to Waiting
   for other DB to end
Fri Apr 20 17:32:33 2012 - SwissProt0 Waiting for other DB to end to Trying to
   memory map files
Fri Apr 20 17:32:33 2012 - SwissProt0 Trying to memory map files  to Just
   enabled memory mapping
Fri Apr 20 17:32:33 2012 - SwissProt0 Just enabled memory mapping to In use
.
.
.
```

IPC Log

In cluster mode (only) an interprocess communication log can be enabled by setting IPCLogging (in the cluster section of *mascot.dat*) to 1 or 2. This log can be used to investigate communications errors at the socket level.

7

7. Program Reference

Mascot implements a client-server architecture using the HTTP protocol, (web server / web browser). In this mode, the search engine is run by the web server as a CGI application.

It is also possible to execute the search engine as a 'console' or 'command line' application. This Chapter provides the information that is required to write scripts or applications which interface to the Mascot search engine and associated programs.

Mascot Search Engine

The Mascot search engine, *`cgi/nph-mascot.exe`*, accepts command line arguments and a MIME format ASCII text file on standard input (STDIN) containing search data and parameters.

```
nph-mascot.exe 1 [-commandline] [-batch] [-f path] [--taskID
  number] [--sessionID string] < in.asc
```

The first argument is required, and is a digit, between 1 and 4, which determines the mode of operation:

1: Normal search; MS/MS data, if any, form part of the MIME format input file

2: Monitor test mode 0

3: Monitor test mode 1

4: Repeat search; the MIME format input file contains a reference to a Mascot results file which may contain MS/MS data

Optional argument `-commandline` is a flag. If present, HTML formatted output is not written to STDOUT. This flag is now deprecated.

Optional argument `-batch` is a flag. If present:

- create a task ID if one is not given on the command line

- print out an HTTP redirect header to search_status.pl (with the task ID as parameter)
- all output to STDOUT (e.g. progress reports) is redirected to F*.dat.log.

Optional argument –f allows a result file path to be specified. In the absence of this argument, the result file will be written to a daily sub-directory of *mascot/data* and have the filename *F123456.dat*, where 123456 is an auto-incremented job number.

Optional argument --taskID is used to specify a unique numeric identifier. This identifier should be obtained from the SearchControl utility, described later in this chapter. By specifying an identifier, progress reports and search results can be obtained asynchronously from SearchControl.

Optional argument --sessionID is used to specify a Mascot security session identifier, (see Chapter 12).

The file piped to STDIN must be a MIME format file containing the search parameters and mass spectrometry data.

Monitor test mode has a different syntax:

nph-mascot.exe 2|3 path [number] < in.asc

Required argument *path* is the path to a flag file, e.g. *../data/test/SwissProt_2011_06.fasta.bu253neb5rcnpqtv2jiiannc2y .testedOk* and optional argument *number* is the cluster number. The input file, e.g. *../data/test/SwissProt.asc*, is created automatically from the *do_not_delete.asc* template. (Hash string bu253neb5rcnpqtv2jiiannc2y is system generated from the size and date of the Fasta file.)

The Monitor application must be running before the search engine can be invoked.

Unless the –batch flag is used, during search execution, warnings, errors, progress reports, etc. are written to standard output (STDOUT). This output is formatted as HTML text for viewing on a web browser. When a search is complete, an HTML string is written to STDOUT, which causes the client browser to invoke the script defined in *mascot.dat* for displaying a results report, (*master_results.pl or master_results_2.pl*). If the search engine is not being executed as a CGI application, the name of the results file can be parsed directly from this string. The output to STDOUT from a successful search will resemble the following:

```
(null) 200 OK
Server: (null)
Content-type: text/html
Pragma: no-cache

<HTML>
<HEAD><TITLE>Mascot searching...</TITLE>
<META HTTP-EQUIV="Expires" CONTENT="0">
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
</HEAD><BODY BGCOLOR='#FFFFFF'>
<!-- comment here -->
<!-- comment here -->
<!-- comment here -->
<!-- comment here -->
<!-- comment here -->
<!-- comment here -->
<!-- comment here -->
```

```
<!-- comment here -->
<!-- comment here -->
<!-- comment here -->
<H1><IMG SRC="../images/88x31_logo_white.gif" WIDTH="88" HEIGHT= ➘
➥ "31" ALIGN="TOP" BORDER="0" NATURALSIZEFLAG="3"> Mascot
   Search</H1>
Licensed to: Matrix Science In-house test system.<BR>Not a real form
Finished uploading search details...<BR>
<B>IMPORTANT:</B> If you get disconnected or choose not to wait ➘
➥ for your search results<BR>DO NOT RESUBMIT THE SEARCH. Your ➘
➥ results will be sent by email when the search is complete<BR>


Searching....<BR>
.10% complete<BR>
..20% complete<BR>
...30% complete<BR>
....50% complete<BR>
.....60% complete<BR>
......70% complete<BR>
.......90% complete<BR>
271397 sequences and 86500527 residues checked.<BR>
<SCRIPT LANGUAGE="JavaScript">
<!-- Begin hiding Javascript from old browsers.
if(window.navigator.userAgent.indexOf("MSIE") != -1){
  window.location.replace("../cgi/master_results.pl?file= ➘
➥ ../data/20090312/F002642.dat");
} else if (window.location.replace == null){
  window.location.assign("../cgi/master_results.pl?file= ➘
➥ ../data/20090312/F002642.dat");
} else {
  window.location.replace("../cgi/master_results.pl?file= ➘
➥ ../data/20090312/F002642.dat");
}


// End hiding Javascript from old browsers. -->
</SCRIPT>
<NOSCRIPT>
<A
   HREF="../cgi/master_results.pl?file=../data/20090312/F002642.dat"
   > ➘
➥ Click here to see Search Report</A>
</NOSCRIPT>
</BODY></HTML>
```

The executable called nph-mascot1.exe is for Mascot TD (“BIG” Mascot), where the precursor mass limit of 16 kDa has been removed. It will only be used for searches if enabled in the licence.

Monitor

The primary function of Mascot Monitor, *bin/ms-monitor.exe*, is to manage the sequence databases. Monitor must be running in order for the search engine to execute. Under Linux this runs as a daemon, and under Windows this runs as a service.

Monitor does the following:

1. Creates compressed files from the databases, checking that the FASTA database files are valid; minor errors in the files are reported as warnings, more serious errors stop the databases from being used
2. These files can then be mapped into memory to improve search times
3. Allows swapping and updating of databases without interruption to executing searches. This means that Mascot can be available for running searches 24/7.
4. Deletes old copies of the FASTA databases to stop the disk becoming full; only the most recent copy is kept.
5. Optionally email a system administrator with serious errors requiring immediate attention. Configuration of email settings in the options section of `mascot.dat` is described in Chapter 6.
6. Optionally email users with their results if they didn't wait for them. Configuration of email settings in the options section of `mascot.dat` is described in Chapter 6.

Sequence of events when a new database is added

When a new or updated database is added to a directory, the following sequence of events takes place:

1. If the entry in the `mascot.dat` file indicates that there should also be a reference file containing full text entries, Monitor looks for a file with the same name as the new file but with a `.ref` or `.dat` extension instead of `.fasta`. If there is no such file, the swap to the new database stops.

2. Compressed index files are made from the `.fasta` and reference files. For example, the following files would be created for the database SwissProt_2014_03:

```
SwissProt_2014_03.a00
SwissProt_2014_03.i00
SwissProt_2014_03.s00
SwissProt_2014_03.stats
SwissProt_2014_03.NoTaxonomyMatch.txt
SwissProt_2014_03.t00
```

The final two files are only created if taxonomy is specified in the database configuration. Compressed files are a proprietary format, which is unlikely be useful for other applications.

3. If a serious error occurs while creating these files, then the conversion to the new database stops, an error is put into the error log and (optionally) the error message is emailed to the administrator. Also, if the status screen is shown, the existence of the error is shown on that screen. Searches on the existing database will continue until the problem is resolved.

4. A test search is performed on the new database. The test uses the appropriate file in the `../data/test` directory. If the test is successful, then a file with the name <database_name>.<unique hash key>.fasta.testedOk is put into the `../data/test` directory. If the test fails, then an error is put into the error log and (optionally) the error message is emailed to the administrator. Also, if the status screen is shown, the existence of the error is shown on that screen. Searches on the existing database will continue until the problem is resolved.

5. Any new searches submitted by users will now use the new database.

6. When there are no more searches running that use the old database, the files for the old database will be unmapped from memory, and the new files are then mapped into memory.

7. Any files in the *old* directory for the database, which have the same base name as the current files, are deleted.

8. The `.fasta` and `.ref` files for the outgoing database are moved to the *old* directory

9. The compressed index files are for the outgoing database are deleted.

Why memory map and lock the FASTA files?

To speed up the processing of the FASTA files, they should be mapped into memory. Databases can be configured in three operational modes:

1. Without memory mapping. Do not choose this option, it will make searches very slow.
2. Memory mapping the database files, but not locking the memory. This gives the best performance in most cases. When the system gets low on memory, the files are swapped out of memory to disk. On most platforms, this will give better performance than simply relying on the system file cache.
3. Memory mapping the files, and locking the memory. This gives the best possible performance, but does require sufficient RAM for the databases, the operating system, searches, and any other applications that are to run concurrently with Mascot.

In order to reduce the amount of memory required, and to prevent memory fragmentation, the sequence strings from the FASTA database are saved separately in a number of files that are then memory mapped. The description line(s) are not memory mapped, only an index to the description in the original database. Compared with mapping the original FASTA database, this can reduce memory requirements by more than 30%. Furthermore, the savings for a nucleic acid database are even greater because the files are compressed with a 2:1 ratio.

For trouble-shooting purposes, Monitor can be started from a command or shell prompt with the argument DEBUG. Under Windows, *ms-monitor.exe* must not be started from the command line if it is already running as a service.

Status and error messages from Monitor can be viewed from a web browser using the Mascot Status application, described below.

GetSeq

GetSeq is a utility for retrieving the sequence, title, or full text of an entry in a database configured for use by Mascot. The utility can be used to retrieve information for a single entry, or in batch mode

Single entry mode

The executable, *x-cgi/ms-getseq.exe*, accepts the following command line parameters:

1. The name of the database, e.g. NCBInr This argument is required.

2. An accession string, e.g. 100K_RAT. This argument is required.
3. One of five keywords: seq, all, len, title or pI. This argument is required, and is explained further below.
4. (Nucleic acid databases only) Frame number between 1 and 6 to retrieve a sequence translated into protein or 0 for the original nucleic acid sequence.
5. (Optionally, if Mascot sercurity enabled) --sessionID followed by a space and then the security session identifier

If the keyword `seq` is supplied, the output from GetSeq has the following format:

```
Content-type: text/plain

*MMSARGDFLNYALSLMRSHNDEHSDVLPRLY ...
  PLYSSKQILKQKLLLAIKTKNFGFV
>100K_RAT 100 KD PROTEIN (EC 6.3.2.-). - RATTUS NORVEGICUS
  (RAT).
```

The keyword, all, is only applicable if a local, full text database is available and configured in *`mascot.dat`*. In which case, the returned text has a format similar to the following:

```
Content-type: text/plain

*MMSARGDFLNYALSLMRSHNDEHSDVLPRLY ...
  PLYSSKQILKQKLLLAIKTKNFGFV
>100K_RAT 100 KD PROTEIN (EC 6.3.2.-). - RATTUS NORVEGICUS
  (RAT).
>P1;100K_RAT
100 KD PROTEIN (EC 6.3.2.-). - RATTUS NORVEGICUS (RAT).
.
.
.
C;DOMAIN 827 847 PRO-RICH.
C;BINDING 858 858 UBIQUITIN (BY SIMILARITY).
C;Keywords: UBIQUITIN CONJUGATION; LIGASE.
```

In all cases, the first line is a content-type specifier, followed by a blank line.

For `seq` and `all` there is then an asterisk followed by the unformatted sequence in one letter code. The next line is identical to the FASTA title line, beginning with a right angle bracket.

In the case of a full text report, this is followed by the raw text entry, as retrieved from the sequence database full text file.

If the keyword len is supplied, then the length of the sequence is returned as ascii text. If the database is a nucleic acid database, then the length returned will depend on the translation frame number specified.

If the keyword title is supplied, the FASTA title line is returned, beginning with a right angle bracket.

If the keyword pI is supplied, the calculated iso-electric point is returned.

Batch mode

Request format

GET-request always means single entry mode. POST-request automatically means batch mode. A batch mode request should use UTF-8 encoding and be of "multipart/form-data"-enctype, for example:

```
-----------------------------41184676334
Content-Disposition: form-data; name="db"

SwissProt
-----------------------------41184676334
Content-Disposition: form-data; name="accession"

"RL19_YEAST"
"G3P2_YEAST", "ERROR_YEAST"
-----------------------------41184676334
Content-Disposition: form-data; name="accession"

"TRY1_BOVIN"
-----------------------------41184676334
Content-Disposition: form-data; name="showpi"

on
-----------------------------41184676334
Content-Disposition: form-data; name="showtitle"

on
-----------------------------41184676334
Content-Disposition: form-data; name="showlen"

on
-----------------------------41184676334
Content-Disposition: form-data; name="showsequence"

on
-----------------------------41184676334
Content-Disposition: form-data; name="showreference"

off
-----------------------------41184676334
Content-Disposition: form-data; name="sessionID"

123456
-----------------------------41184676334--
```

Maximum number of accession strings submitted at once shouldn't be more than 100 000 and the total size of request shouldn't be more than 10 Mb.

All request parameter names are case-insensitive. Any parameter value can be optionally quoted.

DB – mandatory parameter and can only appear once. If several databases are searched than ms-getseq must be called separately for each database.

ACCESSION – must appear at least once and consist of entries in the format "accession_string"[:frameNo]

Quotes around accession strings are mandatory, Frame number can be integer from 0 to 6 and can only be specified for NA-databases. Otherwise, an error will be reported. Accessions can be delimited with commas, spaces, tabs or new-line characters. Several ACCESSION fields will be merged by ms-getseq.exe into one internally.

SHOWPI – can appear only once and if set to TRUE pi-values will have to be calculated for each sequence and output.

SHOWTITLE – can appear only once and if set to TRUE a description for each db-entry has to be output.

SHOWLEN – can appear only once and if set to TRUE a length of sequence string is output for each db-entry.

SHOWSEQUENCE – can appear only once and if set to TRUE a sequence string should be output for every db-entry.

SHOWREFERENCE – can appear only once and if set to TRUE reference lines should be output for each db-entry.

SESSIONID – an optional parameter and can appear at most once. If no session ID is supplied then ms-getseq can either process the request when security is disabled or try to retrieve the ID from cookies.

Boolean values can be coded in different ways:

> true = TRUE = True = on = any number except 0 = any string except an empty string
>
> false = FALSE = False = 0 = ""

All missing parameters are defaulted to "false" value. Missing frame-parameter by default is equal to 0.

Output format

In response to any POST-request, XML format output is returned. Encoding UTF-8 is to be used for output. XML output is schema-validated and schema-versioned. All XML output must be XML escaped using the following substitutions:

>	&gt;
<	&lt;
&	&amp;
'	&apos;
"	&quot;

Proteins are returned in the order requested. A <msgs:frame> element will only be output for an NA database.

The example input file would produce output similar to this (edited for brevity):

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```

```
<msgs:ms_getseq_out
  xmlns:msgs="http://www.matrixscience.com/xmlns/schema/msget
  seq_1"
                    majorVersion="1" minorVersion="0"

  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  xsi:schemaLocation="http://www.matrixscience.com/xmlns/sche
  ma/msgetseq_1 msgetseq_1.xsd">
  <msgs:all_errors>
    <msgs:error code="461">
      <msgs:err_description>Sequence not
  found</msgs:err_description>
      <msgs:err_param
  name="accession">ERROR_YEAST</msgs:err_param>
    </msgs:error>
  </msgs:all_errors>
  <msgs:all_proteins jobid="873">
    <msgs:protein>
      <msgs:accession>RL19_YEAST</msgs:accession>
      <msgs:db>SwissProt</msgs:db>
      <msgs:prot_title>&gt;sp|P05735|RL19_YEAST 60S ribosomal
  protein L19 OS=Saccharomyces cerevisiae GN=RPL19A PE=1 SV=
5</msgs:prot_title>
      <msgs:prot_len>189</msgs:prot_len>
      <msgs:prot_pi>11.35</msgs:prot_pi>
      <msgs:prot_sequence>MANLRT ...
  ALLKEDA</msgs:prot_sequence>
    </msgs:protein>
    <msgs:protein>
      <msgs:accession>G3P2_YEAST</msgs:accession>
      .
      .
      .
    </msgs:protein>
    <msgs:protein>
      <msgs:accession>TRY1_BOVIN</msgs:accession>
      .
      .
      .
    </msgs:protein>
  </msgs:all_proteins>
</msgs:ms_getseq_out>
```

Error messages

All errors have unique codes and are logged to both the XML output and the Mascot error log, (but only the first 10 instances of any particular error number). The XML output contains a full set of error messages in a structured format that can be processed automatically.

Fatal Errors (no database entry is going be retrieved)

403	"Error while reading mascot.dat" Parameters: errstring – error message as generated by ms-parser
463	"'db' parameter is missing"

464 "'accession' parameter is missing"
440 "Invalid session or session ID"
Parameters:
errstring – error message as returned by security objects
443 "Not allowed to search the database"
Parameters:
db – database name that was requested
462 "One or more errors happened while loading taxonomy nodes"
Parameters:
Messages – more detailed error information
460 "Failed to register job. Please inspect mascot error log."
270 "A POST-request is submitted with zero content length"
55 "Cannot find boundary string"
56 "First line was not a boundary"
259 "Corrupted input - possibly a binary file is submitted"
72 "Corrupted input or incompatible browser"
458 "Invalid accession format for ms-getseq.exe"
459 "Too large POST-request"
54 "Standard input stream error"
Parameters:
bytesread – number of bytes already read
lengthofdata – total size of input data in the stream

Non-fatal Errors:

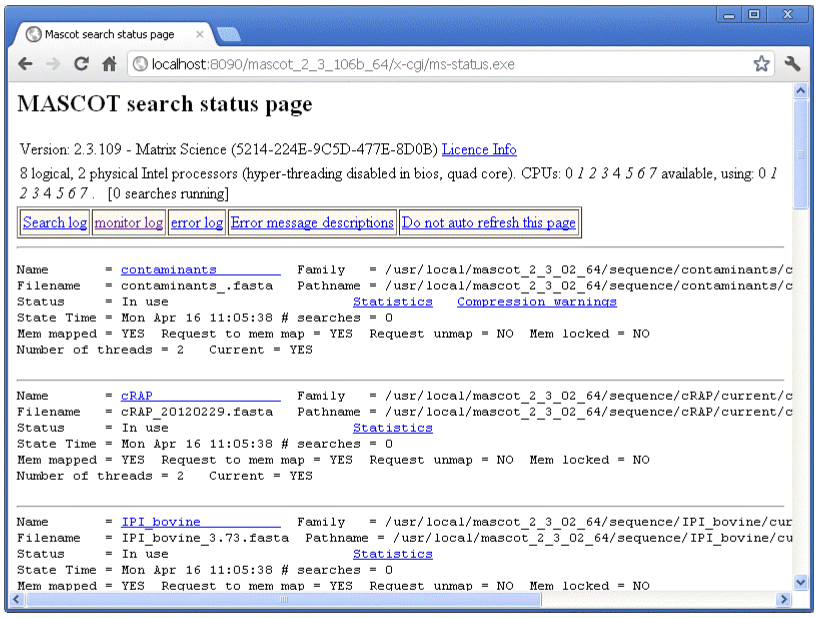
461 "Sequence not found"
Parameters:
accession – accession string
frame – frame number (0 if not supplied in the input or missing if AA-database)

Warnings that are only reported in the end of the XML document:

400 "Missing or invalid gencode id. Table 1 is used for translation"
Parameters:
accession – accession string
frame – frame number (0 if not supplied or missing if AA-database)
470 "Cannot find taxonomy id"
Parameters:
accession – accession string
frame – frame number (0 if not supplied or missing if AA-database)
104 "Sequence is too long for translation"
Parameters:
accession – accession string
frame – frame number (0 if not supplied in the input or missing if AA-database)

Status

The Database Status utility, *x-cgi/ms-status.exe*, provides an overview of the active and recent searches on all of the configured databases. The top level display will resemble this:



Mascot search status page
localhost:8090/mascot_2_3_106b_64/x-cgi/ms-status.exe

MASCOT search status page

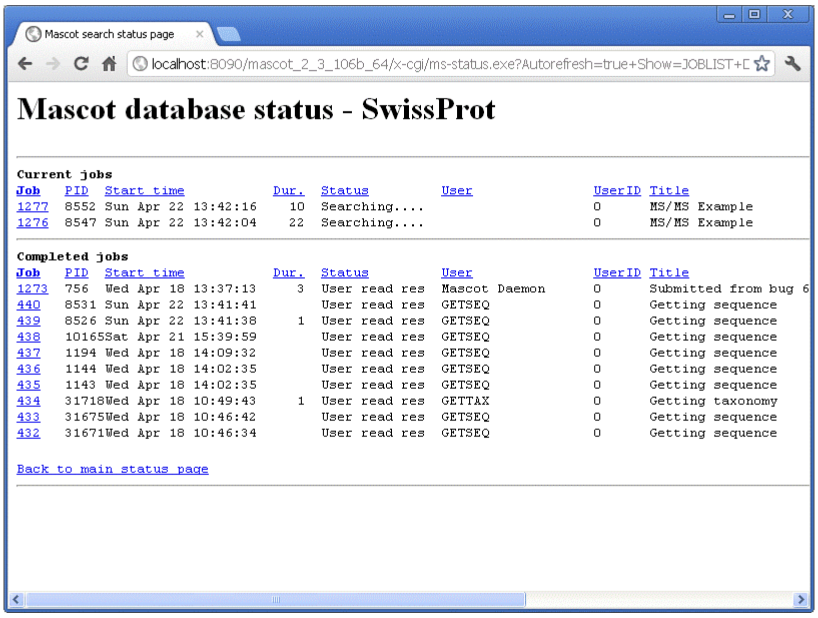
Version: 2.3.109 - Matrix Science (5214-224E-9C5D-477E-8D0B) Licence Info

8 logical, 2 physical Intel processors (hyper-threading disabled in bios, quad core). CPUs: 0 *1 2 3* 4 *5 6 7* available, using: 0 *1 2 3* 4 *5 6 7* . [0 searches running]

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

```
Name       = contaminants           Family   = /usr/local/mascot_2_3_02_64/sequence/contaminants/c
Filename   = contaminants_.fasta    Pathname = /usr/local/mascot_2_3_02_64/sequence/contaminants/c
Status     = In use                           Statistics   Compression warnings
State Time = Mon Apr 16 11:05:38 # searches = 0
Mem mapped = YES  Request to mem map = YES  Request unmap = NO  Mem locked = NO
Number of threads = 2   Current = YES

Name       = cRAP                   Family   = /usr/local/mascot_2_3_02_64/sequence/cRAP/current/c
Filename   = cRAP_20120229.fasta    Pathname = /usr/local/mascot_2_3_02_64/sequence/cRAP/current/c
Status     = In use                           Statistics
State Time = Mon Apr 16 11:05:38 # searches = 0
Mem mapped = YES  Request to mem map = YES  Request unmap = NO  Mem locked = NO
Number of threads = 2   Current = YES

Name       = IPI_bovine             Family   = /usr/local/mascot_2_3_02_64/sequence/IPI_bovine/cur
Filename   = IPI_bovine_3.73.fasta  Pathname = /usr/local/mascot_2_3_02_64/sequence/IPI_bovine/cu
Status     = In use                           Statistics
State Time = Mon Apr 16 11:05:38 # searches = 0
Mem mapped = YES  Request to mem map = YES  Request unmap = NO  Mem locked = NO
```

By clicking on a database hypertext link, a page is displayed showing the activity on that particular database:

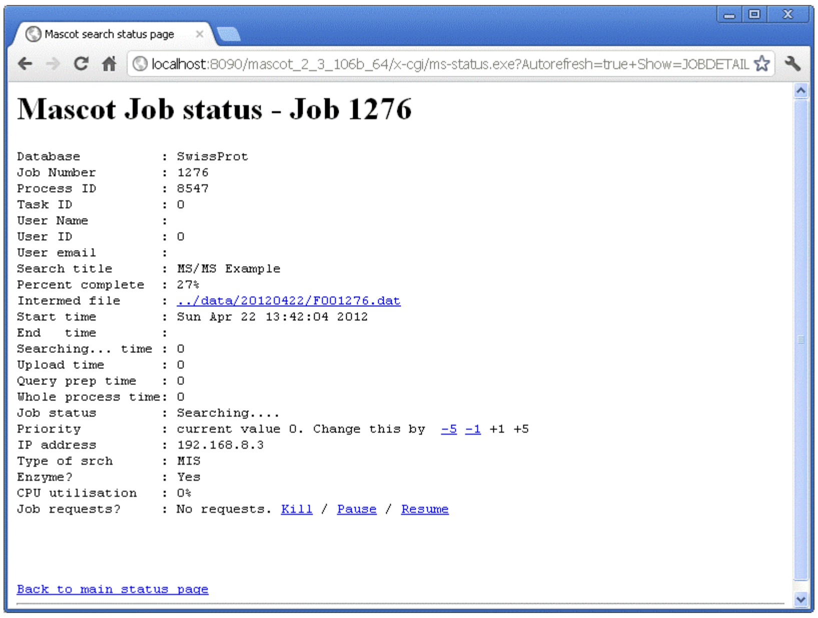


Mascot search status page
localhost:8090/mascot_2_3_106b_64/x-cgi/ms-status.exe?Autorefresh=true+Show=JOBLIST+D

Mascot database status - SwissProt

```
Current jobs
Job   PID  Start time           Dur.  Status        User                UserID  Title
1277  8552 Sun Apr 22 13:42:16    10  Searching....                     0       MS/MS Example
1276  8547 Sun Apr 22 13:42:04    22  Searching....                     0       MS/MS Example

Completed jobs
Job   PID  Start time           Dur.  Status        User                UserID  Title
1273  756  Wed Apr 18 13:37:13     3  User read res Mascot Daemon       0       Submitted from bug 6
440   8531 Sun Apr 22 13:41:41        User read res GETSEQ              0       Getting sequence
439   8526 Sun Apr 22 13:41:38     1  User read res GETSEQ              0       Getting sequence
438   10165Sat Apr 21 15:39:59        User read res GETSEQ              0       Getting sequence
437   1194 Wed Apr 18 14:09:32        User read res GETSEQ              0       Getting sequence
436   1144 Wed Apr 18 14:02:35        User read res GETSEQ              0       Getting sequence
435   1143 Wed Apr 18 14:02:35        User read res GETSEQ              0       Getting sequence
434   31718Wed Apr 18 10:49:43     1  User read res GETTAX              0       Getting taxonomy
433   31675Wed Apr 18 10:46:42        User read res GETSEQ              0       Getting sequence
432   31671Wed Apr 18 10:46:34        User read res GETSEQ              0       Getting sequence

Back to main status page
```

From which, links allow details of any specific search to be displayed:


Mascot search status page
localhost:8090/mascot_2_3_106b_64/x-cgi/ms-status.exe?Autorefresh=true+Show=JOBDETAIL

Mascot Job status - Job 1276

```
Database          : SwissProt
Job Number        : 1276
Process ID        : 8547
Task ID           : 0
User Name         :
User ID           : 0
User email        :
Search title      : MS/MS Example
Percent complete  : 27%
Intermed file     : ../data/20120422/F001276.dat
Start time        : Sun Apr 22 13:42:04 2012
End   time        :
Searching... time : 0
Upload time       : 0
Query prep time   : 0
Whole process time: 0
Job status        : Searching....
Priority          : current value 0. Change this by  -5 -1 +1 +5
IP address        : 192.168.8.3
Type of srch      : MIS
Enzyme?           : Yes
CPU utilisation   : 0%
Job requests?     : No requests. Kill / Pause / Resume


Back to main status page
```

Status can also be used to print Mascot configuration and result files to STDOUT. This provides a method to display these files in a browser. For example:

http://your_server/mascot/x-cgi/ms-status.exe?Show=MS_ENZYMES

where the argument to Show determines the file to be displayed:

MS_ENZYMES	enzymes
MS_FRAGMENTATION_RULES	fragmentation_rules
MS_MASCOT_DAT	mascot.dat
MS_MASSES	masses
MS_MOD_FILE	mod_file
MS_QUANTITATIONXML	quantitation.xml
MS_SUBSTITUTIONS	substitutions
MS_TAXONOMY	taxonomy
MS_UNIMODXML	unimod.xml

The above files are all displayed as plain text, without any formatting. If Show=RESULTFILE, then a results file from any directory under mascot/data can be returned, with HTML formatting. For example:

http://your_server/mascot/x-cgi/ms-status.exe?Show=RESULTFILE&DateDir=20031231&ResJob=F006983.dat

For security reasons, the following characters are not allowed in the DateDir or ResJob: ~ / \ :

The argument MS_USERS returns a list of users that can be spoofed by the user whose session ID was supplied. This may be an empty list. Output format is: “username”,”user id”,”user type”,”full name”,”email address”. E.g.:

```
"guest","1","1","Guest user","guest@localhost"
"admin","2","1","Administrator","admin@localhost"
"daemon","4","1","Mascot Daemon","daemon@localhost"
```

MS_STATUSXML returns an XML formatted document equivalent to the main, top-level status page. The schema is html\xmlns\schema\msstatus_1\msstatus_1.xsd

JOBDETAILS_XML returns an XML formatted document equivalent to the individual Mascot Job status page. The schema is html\xmlns\schema\msstatus_1\msstatus_1.xsd. The job number for the search must be specified as a second argument. For example

```
http://your_server/mascot/x-cgi/ms-
  status.exe?Show=JOBDETAILS_XML&Job=15993017
```

RECOMPRESS_PLAIN causes the specified database to be re-compressed. The database name must be specified as a second argument. For example

```
http://your_server/mascot/x-cgi/ms-
  status.exe?Show=RECOMPRESS_PLAIN&DBName=SwissProt
```

Review

Mascot Review, `x-cgi/ms-review.exe`, provides similar functionality to Status, but takes its input from `searches.log`. The tabular display can be filtered and sorted to locate specific searches by title, user name, or any one of the following log fields:

1. Mascot job number.
2. Process ID
3. Sequence Database
4. User name
5. User email address
6. Search title
7. Results file path
8. Start time and date
9. Duration in seconds
10. Completion Status
11. Job priority
12. Type of search: PMF, SQ, or MIS
13. Enzyme: Either `yes` (if user selected an enzyme) or `no` (if user selected enzyme type None).
14. User IP address

At the top of each column is a checkbox and a radio button. Select the radio button to sort the display on that column. Uncheck the checkbox to hide that column.

Along the top of the screen are a series of controls:

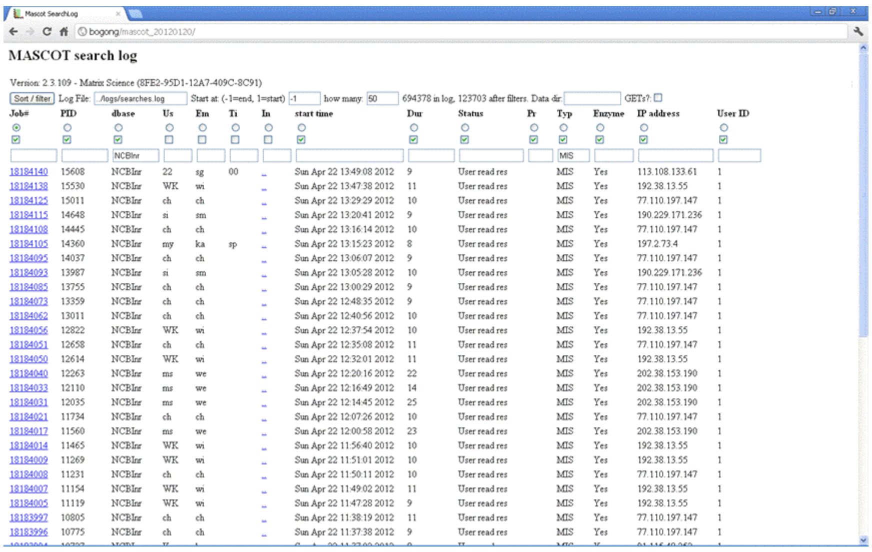
The Sort/filter button updates the display to reflect changes in parameters.

If you have multiple log files, a specific file can be displayed by entering its path into the Log File text field.

Start can be used to page through a long listing in blocks of entries specified by the number in the following field. Setting start to -1 displays the list starting from the last entry in the log file rather than the first

Finally, there is a field to specify a path to the data files. The log file only contains a relative path. If the data files have been moved, possibly to an archive directory or CD-ROM, the path to the new location can be specified here so as to restore the validity of the relative path.

An example of the Status display, filtered to show MS/MS searches of NCBInr, is shown below:



Mascot SearchLog

bogong/mascot_20120120/

MASCOT search log

Version: 2.3.109 - Matrix Science (8FE2-95D1-12A7-409C-8C91)

Sort / filter Log File: ../logs/searches.log Start at: (-1=end, 1=start) -1 how many: 50 694378 in log, 123703 after filters. Data dir: GETs?: ☐

Job#	PID	dbase	Us	Em	Ti	In	start time	Dur	Status	Pr	Typ	Enzyme	IP address	User ID
●	○	○	○	○	○	○	○	○	○	○	○	○	○	○
✓	✓	✓	☐	☐	☐	☐	✓	✓	✓	✓	✓	✓	✓	✓
		NCBInr									MIS			
18184140	15608	NCBInr	22	sg	00	..	Sun Apr 22 13:49:08 2012	9	User read res		MIS	Yes	113.108.133.61	1
18184138	15530	NCBInr	WK	wi		..	Sun Apr 22 13:47:38 2012	11	User read res		MIS	Yes	192.38.13.55	1
18184125	15011	NCBInr	ch	ch		..	Sun Apr 22 13:29:29 2012	10	User read res		MIS	Yes	77.110.197.147	1
18184115	14648	NCBInr	si	sm		..	Sun Apr 22 13:20:41 2012	9	User read res		MIS	Yes	190.229.171.236	1
18184108	14445	NCBInr	ch	ch		..	Sun Apr 22 13:16:14 2012	10	User read res		MIS	Yes	77.110.197.147	1
18184105	14360	NCBInr	my	ka	sp	..	Sun Apr 22 13:15:23 2012	8	User read res		MIS	Yes	197.2.73.4	1
18184095	14037	NCBInr	ch	ch		..	Sun Apr 22 13:06:07 2012	9	User read res		MIS	Yes	77.110.197.147	1
18184093	13987	NCBInr	si	sm		..	Sun Apr 22 13:05:28 2012	10	User read res		MIS	Yes	190.229.171.236	1
18184085	13755	NCBInr	ch	ch		..	Sun Apr 22 13:00:29 2012	9	User read res		MIS	Yes	77.110.197.147	1
18184073	13359	NCBInr	ch	ch		..	Sun Apr 22 12:48:35 2012	9	User read res		MIS	Yes	77.110.197.147	1
18184062	13011	NCBInr	ch	ch		..	Sun Apr 22 12:40:56 2012	10	User read res		MIS	Yes	77.110.197.147	1
18184056	12822	NCBInr	WK	wi		..	Sun Apr 22 12:37:54 2012	10	User read res		MIS	Yes	192.38.13.55	1
18184051	12658	NCBInr	ch	ch		..	Sun Apr 22 12:35:08 2012	11	User read res		MIS	Yes	77.110.197.147	1
18184050	12614	NCBInr	WK	wi		..	Sun Apr 22 12:32:01 2012	11	User read res		MIS	Yes	192.38.13.55	1
18184040	12263	NCBInr	ms	we		..	Sun Apr 22 12:20:16 2012	22	User read res		MIS	Yes	202.38.153.190	1
18184033	12110	NCBInr	ms	we		..	Sun Apr 22 12:16:49 2012	14	User read res		MIS	Yes	202.38.153.190	1
18184031	12035	NCBInr	ms	we		..	Sun Apr 22 12:14:45 2012	25	User read res		MIS	Yes	202.38.153.190	1
18184021	11734	NCBInr	ch	ch		..	Sun Apr 22 12:07:26 2012	10	User read res		MIS	Yes	77.110.197.147	1
18184017	11560	NCBInr	ms	we		..	Sun Apr 22 12:00:58 2012	23	User read res		MIS	Yes	202.38.153.190	1
18184014	11465	NCBInr	WK	wi		..	Sun Apr 22 11:56:40 2012	10	User read res		MIS	Yes	192.38.13.55	1
18184009	11269	NCBInr	WK	wi		..	Sun Apr 22 11:51:01 2012	10	User read res		MIS	Yes	192.38.13.55	1
18184008	11231	NCBInr	ch	ch		..	Sun Apr 22 11:50:11 2012	10	User read res		MIS	Yes	77.110.197.147	1
18184007	11154	NCBInr	WK	wi		..	Sun Apr 22 11:49:02 2012	11	User read res		MIS	Yes	192.38.13.55	1
18184005	11119	NCBInr	WK	wi		..	Sun Apr 22 11:47:28 2012	9	User read res		MIS	Yes	192.38.13.55	1
18183997	10805	NCBInr	ch	ch		..	Sun Apr 22 11:38:19 2012	11	User read res		MIS	Yes	77.110.197.147	1
18183996	10775	NCBInr	ch	ch		..	Sun Apr 22 11:37:38 2012	9	User read res		MIS	Yes	77.110.197.147	1

GetTaxonomy

GetTaxonomy is a utility for retrieving taxonomy details for an entry in a database configured for use by Mascot. The utility can be used to retrieve information for a single entry, ot in batch mode

Single entry mode

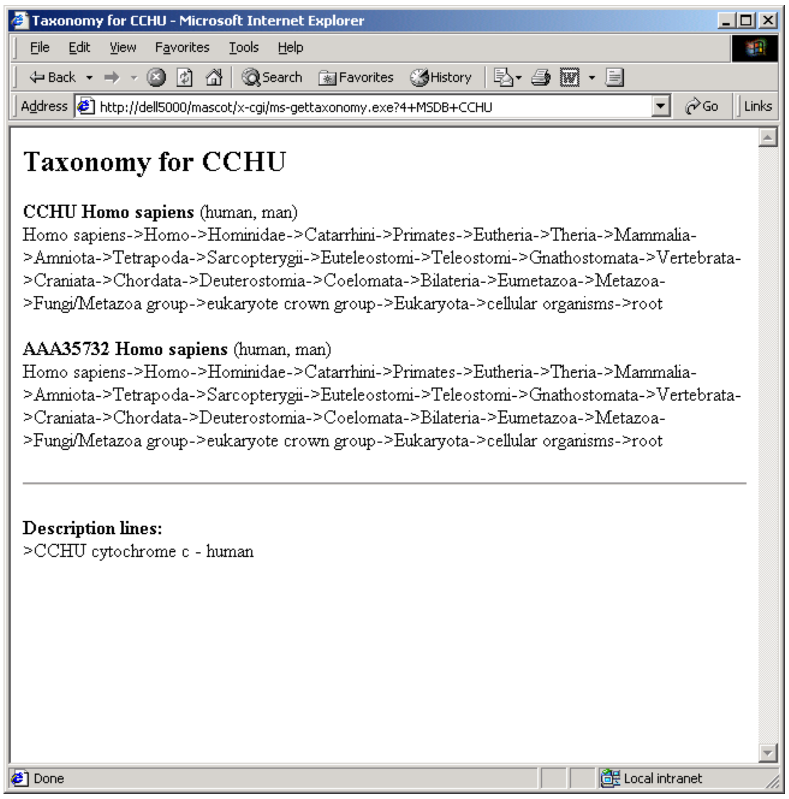
The executable, *`x-cgi/ms-gettaxonomy.exe`*, can be called from the command line, or via a URL as a CGI application.

When calling as a CGI application, with arguments appended to the URL, the parameter list must be URL escaped. (Spaces replaced by '+' and characters other than letters or numbers replaced by a '%xx' where xx is the ASCII code for the character as a hexadecimal number).

When running from a command line, the accession string should be enclosed in single or double quotes. This is essential for accession strings beginning `gi|` , because the pipe character has special meaning in Linux and Windows.

In the table below, the first argument supplied to `ms-gettaxonomy.exe` is an integer to specify the mode. The remaining arguments are selected from:

database	Mascot database name, e.g. NCBInr
accession	accession string, e.g. gi|7633482
tax_ID	taxonomy ID number, e.g. 9606
species	name of species, e.g. homo sapiens


Taxonomy for CCHU - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://dell5000/mascot/x-cgi/ms-gettaxonomy.exe?4+MSDB+CCHU
Go Links
Done
Local intranet

Taxonomy for CCHU

CCHU Homo sapiens (human, man)
Homo sapiens->Homo->Hominidae->Catarrhini->Primates->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Coelomata->Bilateria->Eumetazoa->Metazoa->Fungi/Metazoa group->eukaryote crown group->Eukaryota->cellular organisms->root

AAA35732 Homo sapiens (human, man)
Homo sapiens->Homo->Hominidae->Catarrhini->Primates->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Coelomata->Bilateria->Eumetazoa->Metazoa->Fungi/Metazoa group->eukaryote crown group->Eukaryota->cellular organisms->root

Description lines:
>CCHU cytochrome c - human

Mode	Parameters	Returns
1	database accession	Space separated list of accession string, tax_ID number, and scientific species name. Where a database entry represents multiple accessions, this information is repeated for each accession. Plain formatted.
2	database accession	Space separated pair of accession string and scientific species name. Where a database entry represents multiple accessions, this information is repeated for each accession. Followed by the FASTA title line for the accession supplied as an argument. Pretty formatted.
3	database accession	Same as mode 2, plus a list of common species names in parentheses.
4	database accession	Same as mode 3, plus complete taxonomy tree
5	database tax_ID	The scientific species name as a string. Pretty formatted.
6	database tax_ID	Same as mode 5, plus a list of common species names in parentheses.

7	database tax_ID	Same as mode 6, plus complete taxonomy tree
8	database species	verbose tax_ID information
9	database accession	genetic code number

Batch mode

Request format

GET-request always means single entry mode. POST-request automatically means batch mode. A batch mode request should use UTF-8 encoding and be of “multipart/form-data”-enctype, for example:

```
-----------------------------41184676334
Content-Disposition: form-data; name="db"

SwissProt
-----------------------------41184676334
Content-Disposition: form-data; name="accession"

"RL19_YEAST"
-----------------------------41184676334
Content-Disposition: form-data; name="taxID"

1061
-----------------------------41184676334
Content-Disposition: form-data; name="showtitle"

on
-----------------------------41184676334
Content-Disposition: form-data; name="showSynonyms"

on
-----------------------------41184676334
Content-Disposition: form-data; name="showTaxTree"

on
-----------------------------41184676334
Content-Disposition: form-data; name="sessionID"

123456
-----------------------------41184676334—
```

The batch format aggregatesboth “find taxonomy from accession” and “find taxonomy from id” requests.

Maximum number of accessions / taxIDs submitted at once must not exceed 100000 and the total size of request should be no more than 10 MB.

All request parameter names are case-insensitive. Any parameter value can be in quotes.

DB – mandatory parameter and can only appear once. If several databases are searched than ms-getseq must be called separately for each database.

ACCESSION – can appear any number of times. Quotes are mandatory. Can have a list of accessions delimited by commas, spaces, tabs or new line characters. All ACCESSION-fields are merged into one list of accession strings internally.

TAXID – can appear any number of times and contains a list of taxonomy ids delimited by commas, spaces or new line characters. All such fields are merged into one list internally.

SHOWTITLE – can appear only once and if set to TRUE a description for each db-entry has to be output.

SHOWSYNONYMS – can appear only once and if set to TRUE a list of common names should be output for each taxonomy.

SHOWTAXTREE – can appear only once and if set to TRUE taxonomy tree should be output for each taxonomy.

SESSIONID – an optional parameter and can appear at most once. If no session ID is supplied then ms-gettaxonomy can either process the request when security is disabled or try to retrieve the ID from cookies.

Boolean values can be coded in different ways:

true = TRUE = True = on = any number except 0 = any string except an empty string

false = FALSE = False = 0 = ""= off

All missing parameters are defaulted to "false" value.

Translation table number is always output as well as taxonomy Id and scientific name.

Output format

In response to any POST-request, XML format output is returned. Encoding UTF-8 is to be used for output. XML output is schema-validated and schema-versioned. All XML output must be XML escaped using the following substitutions:

>	&gt;
<	&lt;
&	&amp;
'	&apos;
"	&quot;

Taxonomy information is returned in the order requested. A <msgs:frame> element will only be output for an NA database.

The example input file would produce output similar to this (edited for brevity):

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<msgt:ms_gettaxonomy_out
  xmlns:msgt="http://www.matrixscience.com/xmlns/schema/msget
  taxonomy_1"
                     majorVersion="1" minorVersion="0"

  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.matrixscience.com/xmlns/sche
  ma/msgettaxonomy_1 msgettaxonomy_1.xsd">
  <msgt:results jobid="874">
    <msgt:db_entry>
      <msgt:db>SwissProt</msgt:db>
      <msgt:accession_str>RL19_YEAST</msgt:accession_str>
      <msgt:title>&gt;sp|P05735|RL19_YEAST 60S ribosomal
  protein L19 OS=Saccharomyces cerevisiae GN=RPL19A PE=1
  SV=5</ms
gt:title>
      <msgt:all_accessions>
        <msgt:accession>
          <msgt:accession_str>RL19_YEAST</msgt:accession_str>
          <msgt:taxonomy>
            <msgt:db>SwissProt</msgt:db>
            <msgt:taxonomy_id>4932</msgt:taxonomy_id>
            <msgt:scientific_name>Saccharomyces
  cerevisiae</msgt:scientific_name>

  <msgt:translation_table_id>1</msgt:translation_table_id>
            <msgt:common_names>
              <msgt:synonym>Candida robusta</msgt:synonym>
              <msgt:synonym>Saccaromyces
  cerevisiae</msgt:synonym>
              <msgt:synonym>Saccharomyces
  capensis</msgt:synonym>
              <msgt:synonym>Saccharomyces
  italicus</msgt:synonym>
              <msgt:synonym>Saccharomyces
  oviformis</msgt:synonym>
              <msgt:synonym>Saccharomyces uvarum var.
  melibiosus</msgt:synonym>
              <msgt:synonym>Saccharomyes
  cerevisiae</msgt:synonym>
              <msgt:synonym>Sccharomyces
  cerevisiae</msgt:synonym>
              <msgt:synonym>YEAST</msgt:synonym>
              <msgt:synonym>baker&apos;s yeast</msgt:synonym>
              <msgt:synonym>brewer&apos;s
  yeast</msgt:synonym>
              <msgt:synonym>lager beer yeast</msgt:synonym>
              <msgt:synonym>yeast</msgt:synonym>
            </msgt:common_names>
            <msgt:tree>
              <msgt:node level="12">Saccharomyces
  cerevisiae</msgt:node>
              <msgt:node level="11">Saccharomyces</msgt:node>
              <msgt:node
  level="10">Saccharomycetaceae</msgt:node>
              <msgt:node
  level="9">Saccharomycetales</msgt:node>
              <msgt:node
  level="8">Saccharomycetes</msgt:node>
              <msgt:node
  level="7">Saccharomycotina</msgt:node>
              <msgt:node level="6">Ascomycota</msgt:node>
              <msgt:node level="5">Dikarya</msgt:node>
              <msgt:node level="4">Fungi</msgt:node>
            <msgt:node level="3">Fungi/Metazoa
group</msgt:node>
            <msgt:node level="2">Eukaryota</msgt:node>
            <msgt:node level="1">cellular
organisms</msgt:node>
          </msgt:tree>
        </msgt:taxonomy>
      </msgt:accession>
    </msgt:all_accessions>
  </msgt:db_entry>
  <msgt:tax_from_id>
    <msgt:taxonomy>
      <msgt:db>SwissProt</msgt:db>
      <msgt:taxonomy_id>1061</msgt:taxonomy_id>
      <msgt:scientific_name>Rhodobacter
capsulatus</msgt:scientific_name>

<msgt:translation_table_id>11</msgt:translation_table_id>
      .
      .
      .
    </msgt:taxonomy>
  </msgt:tax_from_id>
</msgt:results>
</msgt:ms_gettaxonomy_out>
```

The way information is represented in the XML output will be clearer if a few rules are kept in mind:

- msgt:title element will only appear in the output if showTitle=true,
- msgt:common_names element will only appear in the output if showSynonyms=true,
- msgt:tree element will only appear in the output is showTaxTree=true,
- order of elements within msgt:tree is essential,
- in msgt:tree "root" element is not listed but always assumed,
- msgt:translation_table_id element may not be available,
- Any of the elements msgt:db_entry, msgt:tax_from_id can be missing or repeated several times depending on request.

Error messages

All errors have unique codes and are logged to both the XML output and the Mascot error log, (but only the first 10 instances of any particular error number). The XML output contains a full set of error messages in a structured format that can be processed automatically.

Fatal Errors (no database entry is going be retrieved)

403	"Error while reading mascot.dat" Parameters: errstring – error message as generated by ms-parser
463	"'db' parameter is missing"
465	"POST-request to ms-gettaxonomy is empty"
440	"Invalid session or session ID"

Parameters:
errstring – error message as returned by security objects

443 "Not allowed to search the database"
Parameters:
db – database name that was requested

27 "Database is not available or not active"
Parameters:
db – database name that was requested

251 "No taxonomy indexes for this database"
Parameters:
db – database name that was requested

469 "Failed to load species file"
Parameters:
messages – more detailed error message

462 "One or more errors happened while loading taxonomy nodes"
Parameters:
messages – more detailed error information

460 "Failed to register job. Please inspect mascot error log."

270 "A POST-request is submitted with zero content length"

55 "Cannot find boundary string"

56 "First line was not a boundary"

259 "Corrupted input - possibly a binary file is submitted"

72 "Corrupted input or incompatible browser"

466 "Invalid accession format for ms-gettaxonomy.exe"

468 "Too large POST-request"

467 "Invalid taxID format for ms-gettaxonomy.exe"

54 "Standard input stream error"
Parameters:
bytesread – number of bytes already read
lengthofdata – total size of input data in the stream

Non-fatal errors:

461 "Sequence not found"
Parameters:
accession – accession string

470 "Cannot find taxonomy id"
Parameters:
accession – accession string (empty if non-fatal error, can be non-empty only in warning-section for accession-requests)
taxid – taxonomy id

Warnings that are only reported in the end of XML document:

400 "Missing or invalid gencode id. Table 1 is used for translation"
Parameters:
accession – accession string (empty if non-fatal error, can be non-empty only in warning-section for accession-requests)
taxid – taxonomy id

470 "Cannot find taxonomy id"
Parameters:
accession – accession string (empty if non-fatal error, can be non-empty only in warning-section for accession-requests)
taxid – taxonomy id

SearchControl

Any helper application can call bin/ms-searchcontrol.exe to implement asynchronous automation of search submission. Available commands are:

```
--status
--result_file_name
--result_file_mime
--result_file_ini
--results
--xmlresults
--create_task_id
--mascot_job_number
--kill_job
--pause_job
--resume_job
--nice_job
--set_to_queued
--version
--create_results_cache
--results_cache_status
```

ms-searchcontrol.exe --status --taskID <number> [--sessionID <string>]

The 'status' command will return one of the following:

```
unknown_id (referring to task ID)
id_assigned (referring to task ID)
error=nnnn
running=yy%
complete
queued
searchcontrol-error=nnn
```

where `error` indicates an error in the search, and will be the Mascot error number or one of:

```
TASK_ERROR_NO_ERROR              =  0
TASK_ERROR_JOB_CRASHED           = -1
TASK_ERROR_JOB_KILLED            = -2
```

And searchcontrol-error indicates a problem with the ms-searchcontrol.exe program. Values will be one of:

```
ERR_TASKID_NOERROR               = 0
ERR_TASKID_FAILOPEN              = 1
ERR_TASKID_FAILCREATE            = 2
ERR_TASKID_FAILREAD              = 3
ERR_TASKID_FAILWRITE             = 4
ERR_TASKID_FAILCLOSE             = 5
ERR_TASKID_CHANGEDRECORD         = 6
ERR_TASKID_INVALIDMASCOTDAT      = 7
ERR_TASKID_MISSINGRESULTSFILE    = 8
ERR_TASKID_FILENAMETOOLONG       = 9
```

```
ERR_TASKID_SESSIONTIMEDOUT       = 10
ERR_TASKID_PERMISSIONDENIED      = 11
```

ms-searchcontrol.exe --result_file_name --taskID <number> [--sessionID <string>]

This will return either the results file name:

filename=<filename>

or

searchcontrol-error=nnn

with values of 'nnn' as for --status.

Note that <filename> may be empty for some states - this is not an error.

This may then be used from the command line for other applications to provide functionality that is not in ms-searchcontrol.exe For example, a client application needs the USER name from a search. In this case, a perl script 'getusername.pl' could be written that takes the passed unique task ID, finds the results file name using:

ms-searchcontrol.exe --result_file_name

and then looks for the user name in the results file.

ms-searchcontrol.exe --result_file_mime --taskID <number> [--sessionID <string>]

This will return the results file as a mime format file or

searchcontrol-error=nnn

with values of 'nnn' as for --status.

ms-searchcontrol.exe --result_file_ini --taskID <number> [--sessionID <string>]

This will return the results file as a windows '.ini' format file or

searchcontrol-error=nnn

with values of 'nnn' as for --status.

ms-searchcontrol,exe --results --taskID <number> [--sessionID <string>]

If the job is complete, then this will return the search results in a format recognised by Mascot Daemon:

For a peptide mass fingerprint, the output is of the form:

```
###daemon###file=..\data\F981122.dat
###daemon###release=MSDB_20020121.fasta
###daemon###queries=8
###daemon###num_hits=6
###daemon###h1=1A6K,103,1.00,17004.1
###daemon###h1_text=myoglobin - sperm whale
###daemon###reptype=concise
###daemon###sigscoreprot=72
###daemon###ionquery1=734.992175 from(736.000000,1+)
```

```
###daemon###ionquery2=746.992175 from(748.000000,1+)
###daemon###ionquery3=939.992175 from(941.000000,1+)
###daemon###ionquery4=1515.992175 from(1517.000000,1+)
###daemon###ionquery5=1591.992175 from(1593.000000,1+)
###daemon###ionquery6=1853.992175 from(1855.000000,1+)
###daemon###ionquery7=1980.992175 from(1982.000000,1+)
###daemon###ionquery8=2111.992175 from(2113.000000,1+)
###daemon###selectpeptides=0
```

For an MS/MS ions search, the output is of the form:

```
###daemon###file=..\data\F981123.dat
###daemon###release=MSDB_20020121.fasta
###daemon###queries=4
###daemon###num_hits=1
###daemon###h1=Q9XZJ2,286.477,1.00,79480.1
###daemon###h1_text=HEAT SHOCK PROTEIN 70.- Crassostrea gigas
  (Pacific oyster).
###daemon###reptype=peptide
###daemon###sigscoreprot=72
###daemon###ionquery1=1341.784350 from(671.900000,2+)
  query(1)
###daemon###score1=95.12
###daemon###sigscore1=49
###daemon###ionquery2=1614.584350 from(808.300000,2+)
  query(2)
###daemon###score2=74.55
###daemon###sigscore2=48
###daemon###ionquery3=1945.784350 from(973.900000,2+)
  query(3)
###daemon###score3=89.84
###daemon###sigscore3=47
###daemon###ionquery4=2167.784350 from(1084.900000,2+)
  query(4)
###daemon###score4=39.11
###daemon###sigscore4=47
###daemon###selectpeptides=1
```

If the job is incomplete, or has failed, then an error will be returned:

unknown_id

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

ms-searchcontrol,exe --xmlresults --taskID <number> --reporttop [FILE|AUTO|num_hits] [--sessionID <string>]

If the job is complete, then this will return the results formatted as an XML instance document that conforms to the schema

mascot/html/xmlns/schema/DistillerMascotSearch_1/DistillerMascotSearch_1.xsd

If the job is incomplete, or has failed, then an error will be returned:

unknown_id

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

ms-searchcontrol.exe --create_task_id [--sessionID <string>]

On failure, this will return

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

And on success it will return:

taskID=nnn

ms-searchcontrol.exe --mascot_job_number --taskID <number> [--sessionID <string>]

This will return either the job number:

mascotjobnumber=nnnn

or

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

ms-searchcontrol.exe --kill_job --taskID <number> [--sessionID <string>]

If the task is successful, this will return the text:

job_killed

If there is an error, one of the following will be returned:

unknown_id

job_not_running

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

The ‘kill’ is implemented by setting a flag in the mascot.control memory mapped file. The nph-mascot.exe task is responsible for ‘killing’ itself.

ms-searchcontrol.exe --pause_job --taskID <number> [--sessionID <string>]

If the task is successful, this will return the text:

job_paused

If there is an error, one of the following will be returned:

unknown_id

job_not_running

job_already_paused

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

The ‘pause’ is implemented by setting a flag in the mascot.control memory mapped file. The nph-mascot.exe task is responsible for ‘pausing’ itself.

ms-searchcontrol.exe --resume_job --taskID <number> [--sessionID <string>]

If the task is successful, this will return the text:

job_resumed

If there is an error, one of the following will be returned:

unknown_id

job_not_running

job_not_paused

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

The ‘resume’ is implemented by setting a flag in the mascot.control memory mapped file. The nph-mascot.exe task is responsible for ‘resuming’ itself.

ms-searchcontrol.exe --nice_job --taskID <number> [--nice <integer>] [--sessionID <string>]

The task ID need to be supplied. and an optional nice value

If a valid new nice value is supplied, this will return the text:

job_niced

If a nice value is not supplied, the program will return the current nice value:

nice=xxx

If there is an error, one of the following will be returned:

unknown_id

job_not_running

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

The ‘nice’ is implemented by setting a flag in the mascot.control memory mapped file. The nph-mascot.exe task is responsible for ‘resuming’ itself. Nice values range from –20 to +20. A value of +20 will set the task to a very low priority. The Mascot status screen shows the ‘nice’ value as a priority, which is simply -1 * the nice value. Microsoft Windows does not allow such a fine grained control of priorities, so, for example +20 and +19 will map to the same priority.

ms-searchcontrol.exe --set_to_queued --taskID <number> [--sessionID <string>]

If the task is successful, this will return the text:

queued

If there is an error, one of the following will be returned:

unknown_id

already_running

already_complete

searchcontrol-error=nnn

with values of ‘nnn’ as for --status.

A batch processing client can make queued jobs visible to the Mascot system by getting a taskID and using this call to set the status to ‘queued’. When the search is eventually submitted, nph-mascot.exe will set the status ‘running’. A queued job will return ‘queued’ when ms-searchcontrol.exe is called with the --status argument.

```
ms-searchcontrol.exe --version [--sessionID <string>]
```

If the task is successful, this will return the version number

```
ms-searchcontrol.exe --create_results_cache --taskID <number>
   [--sessionID <string>]
```

Creates the cache files, so subsequent calls to get results will be faster. On completion, outputs the same values as the --results_cache_status.

When called from client.pl, this command returns instantly and the --results_cache_status command should be used to determine the progress and status.

```
ms-searchcontrol.exe --results_cache_status --taskID <number>
   [--sessionID <string>]
```

If there are no cache files, then this will return the text:

not_created

If the cache files have been created and are ready for use, then this will return the text:

created

If the cache files are being created by a previous call using the --create_results_cache command, then this will return the text:

creating=xx%

where xx the percentage value between 0 and 100 of the process to create the cache files. If the cache files are being created by another process, for example someone loading the results report, then this will return the text:

creating=

with no percentage value. If caching is disabled for ms-searchcontrol.exe in the Options section of mascot.dat, then this will return the text:

caching_disabled

If the results file is missing, then this will return the text:

missing_resultsfile=[filename]

If there is no results file associated with the taskID, then this will return

the text:

missing_results_filename

If there was an error creating the cache files, for example a corrupt or incomplete file or insufficient disk space, then this will return the text:

error=xxx:[error_text]

where xxx is a Mascot Parser error number. If the taskID specified is invalid, then this will return the text:

unknown_id

If there is a search control error, then this will return the text:

searchcontrol-error=nnn

with values of 'nnn' as for --status.

CreatePIP

Usage

ms-createpip.exe [OPTION] -i filename

Options

-h, --help	display this help page and exit
-f, --features	display list of features defined in mascot.dat and exit
--sessionID <id>	not normally used because this is run from command line
-o output_file	default is filename.pip
-q <#queries>	override minimum number of queries set in mascot.dat
-s <#sequences>	override minimum number of sequences set in mascot.dat
-a <feature>	add a feature to the list specified in mascot.dat
-r <feature>	remove a feature to the list specified in mascot.dat
-c	use cached results

-p <interval>	progress reports of Process:X% every <interval> seconds
--nocache	do not use cached results
--version	display version number and exit

If neither -c or --nocache are specified, then cache usage depends on whether ms-createpip.exe is specifed in the ResfileCache and ResultsCache lines in the options section of mascot.dat

Features

retentionTime	Retention time in seconds if available
dM	Calculated minus observed peptide mass in Da
mScore	Mascot score (always on)
lgDScore	Mascot score minus Mascot score of next best non-isobaric peptide hit
mrCalc	Calculated Mr
charge	Charge
dMppm	Calculated minus observed peptide mass in ppm
absDM	Absolute value of calculated minus observed peptide mass in Da
absDMppm	Absolute value of calculated minus observed peptide mass in ppm
isoDM	Calculated minus observed peptide mass, after eliminating possible isotope errors up to 2 Da, in Da
isoDMppm	Calculated minus observed peptide mass, after eliminating possible isotope errors up to 2 Da, in ppm
mc	Number of missed cleavages (always 0 if no enzyme)
varmods	Number of modified sites divided by number of modifiable sites

varmodsCount	The number of variable mods used in the peptide. That is, if there are 10 Met and 5 of these are oxidised, this counts as 1. A peptide with Met-OX, phosphoS, deamidation, and acetylation, would count as 5.
modifiable	Total number of modifiable sites
modified	Total number of modified residues and terminii
totInt	Log total ion intensity. The 20 most intense peaks in each 100 Da bin are used for all features, and totInt reports this value
intMatchedTot	Log total matched ion intensity
relIntMatchedTot	Total matched ion intensity divided by total ion intensity as a percentage (no logs involved)
fragDeltaMed	Median value of all matched fragment errors in Da
fragDeltaIqr	Interquartile range value of all matched fragment errors in Da
fragDeltaMedPPM	Median value of all matched fragment errors in ppm
fragDeltaIqrPPM	Interquartile range value of all matched fragment errors in ppm
fragDeltaPolyFit	2nd order polynomial fit to m/z vs delta. Result is Rsquared multiplied by the number of points divided by 100
longest	Longest sequence matched ions, reported separately for each ion series (backbone only), as with fracIonsMatched
fracIonsMatched	Fraction of calculated ions matched, reported separately for each ion series, with NLs lumped together (e.g. fracIonsMatchedB1, fracIonsMatchedB1deriv, fracIonsMatchedB2, fracIonsMatchedB2deriv)
matchedIntensity	Matched ion intensity, reported separately for each ion series, as with fracIonsMatched

numUniqPeps	The excess of the number of unique peptide matches (unique primary sequence) over the number of matches expected by chance (not implemented)
qmatch	The number of peptide matches for which an ms-ms match was attempted
peptide	The peptide string that was matched
proteins	A tab separated list of accessions of proteins that contain this peptide. Must be last

Error codes

Return	Description
-1	Invalid parameters. Use --help for help
-2	Missing or invalid mascot.dat. Error:
-3	No MS-MS spectra in results file
-4	Automatic decoy search not enabled
-5	Insufficient number of queries.
-6	Insufficient number of sequences searched.
-7	Cannot read the results file. Error:
-8	Failed to create output file:
-9	Invalid feature in mascot.dat options:
-10	Invalid feature for -a option:
-11	Invalid feature for -r option:

Miscellaneous Utilities

Service

Supplied for Windows only. This application shows the status of the Matrix Science Mascot Service, and allows it to be stopped and started. It is normally accessed from the start menu -

```
Programs; Mascot; config; Show Mascot Service Status

Programs; Mascot; config; Start Mascot Service

Programs; Mascot; config; Stop Mascot Service
```

These options run the program *`x-cgi/ms-service.exe`* with the first parameter set to the service name (`MatrixScienceMascotService`) and the second parameter being `0, 1, or 2` respectively.

It is also possible to run this program as a CGI script by entering the following URL in the browser:

`http://your.host/mascot/x-cgi/ms-service.exe?MatrixScienceMascotService+0`

Where *`your.host`* is replaced by the host name of the Mascot server. This CGI script can be run from any computer on the network. However, it is not usually possible to start and stop the service from another computer using the default access rights.

There is a final option, which will allow removal of the service. This may be required for a manual de-installation and will not normally be required. If this option is used, Mascot will not run again without re- running the installation program. The command to enter is:

`ms-service MatrixScienceMascotService remove`

Compress

Compress is a utility for compressing FASTA files independently of Mascot monitor.

The executable, *`bin/ms-compress.exe`* is executed from a shell or command prompt.

`ms-compress.exe db_name fasta`

where

`db_name` is the database family name from mascot.dat - e.g. MSDB

`fasta` is the fully qualified path to the FASTA file

`ms-compress.exe` compresses the fasta file using the rules specifed in `mascot.dat` and must be run so that it's current working directory is *`mascot/bin`*.

Return value of 0 for success, > 0 for failure

MakeSearchLog

The bin/ms-makesearchlog.exe utility is used to rebuild the search log by scanning all the result files located in sub-directories of *mascot/data*. This can be useful if the original search log has been damaged or if result files have been pruned after archiving. There are no arguments.

LockMem

On 32 bit platforms, the 2GB address space limit can quickly be reached by having several large databases locked into memory. To work around this limit, the *bin/ms-lockmem.exe* utility is provided.

LockMem is enabled by adding the parameter 'SeparateLockMem 1' to the options section of mascot.dat. Specifying a value greater than 1 specifies the block size in MB. For example, if there is a 1.5 GB `*.s00` file, and this parameter is set to 750, two instances of *ms-lockmem.exe* will be run.

GetError

The utility *cgi/ms-geterror.exe* takes an error number as an argument and returns the corresponding text string. For example:

```
C:\Inetpub\MASCOT\cgi>ms-geterror.exe 23
You specified enzyme %s which is not available. Choose
  another.
```

8

8. I/O File Formats

Both Mascot search input files and results output files are in MIME format. This is a text file which can be viewed easily for inspection or debugging purposes.

The MIME format is defined in various "request for comment" documents. The following are the most relevant:

ftp://ftp.isi.edu/in-notes/rfc2045.txt

ftp://ftp.isi.edu/in-notes/rfc2046.txt

ftp://ftp.isi.edu/in-notes/rfc2388.txt

Very briefly, a unique boundary string is used to divide the file into sections, each of which contains data in a format defined by a Content-type.

Each section begins with two hyphens followed by the boundary string. The next line contains the content definition and name, followed by a blank line. Then data, until the beginning of the next section. For example:

```
MIME-Version: 1.0 (Generated by Mascot version 1.0)
Content-Type: multipart/mixed; boundary=gc0p4Jq0M2Yt08jU534c0p

--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="first_name"

first_value
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="another_name"

another_value
--gc0p4Jq0M2Yt08jU534c0p
.
.
.
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="final_name"

final_value
--gc0p4Jq0M2Yt08jU534c0p--
```

Search Input File

The search input file is normally generated by the web browser. If another application is used to generate an input file, simply ensure that it conforms to the MIME format standard.

The Mascot Monitor test searches use “captured” input files. Hence, an example of a file can be seen by opening `mascot/data/test/SwissProt.asc` in any text editor.

```
-----------------------------243501029130836
Content-Disposition: form-data; name="INTERMEDIATE"


-----------------------------243501029130836
Content-Disposition: form-data; name="FORMVER"

1.01
-----------------------------243501029130836
Content-Disposition: form-data; name="SEARCH"

MIS
-----------------------------243501029130836
Content-Disposition: form-data; name="PEAK"

AUTO
-----------------------------243501029130836
Content-Disposition: form-data; name="REPTYPE"

peptide
-----------------------------243501029130836
Content-Disposition: form-data; name="ErrTolRepeat"

0
-----------------------------243501029130836
Content-Disposition: form-data; name="SHOWALLMODS"


-----------------------------243501029130836
Content-Disposition: form-data; name="USERNAME"

Monitor Program Test
-----------------------------243501029130836
Content-Disposition: form-data; name="COM"

MS/MS Test Search
-----------------------------243501029130836
Content-Disposition: form-data; name="DB"

SwissProt
-----------------------------243501029130836
Content-Disposition: form-data; name="CLE"

Trypsin/P
-----------------------------243501029130836
Content-Disposition: form-data; name="PFA"

1
-----------------------------243501029130836
Content-Disposition: form-data; name="QUANTITATION"

None
```

```
-----------------------------243501029130836
Content-Disposition: form-data; name="TAXONOMY"

All entries
-----------------------------243501029130836
Content-Disposition: form-data; name="MODS"

Carbamidomethyl (C)
-----------------------------243501029130836
Content-Disposition: form-data; name="IT_MODS"

Oxidation (M)
-----------------------------243501029130836
Content-Disposition: form-data; name="TOL"

100
-----------------------------243501029130836
Content-Disposition: form-data; name="TOLU"

ppm
-----------------------------243501029130836
Content-Disposition: form-data; name="PEP_ISOTOPE_ERROR"

0
-----------------------------243501029130836
Content-Disposition: form-data; name="ITOL"

0.1
-----------------------------243501029130836
Content-Disposition: form-data; name="ITOLU"

Da
-----------------------------243501029130836
Content-Disposition: form-data; name="CHARGE"

1+
-----------------------------243501029130836
Content-Disposition: form-data; name="MASS"

Monoisotopic
-----------------------------243501029130836
Content-Disposition: form-data; name="FILE"; filename="test_search.mgf"
Content-Type: application/octet-stream

BEGIN IONS
PEPMASS=498.272888
CHARGE=1+
157.096962 23.72
185.160000 26.69
286.134951 80.7
385.210000 13.49
.
.
.
2000.120000 3.142
2000.568167 4.108
2001.020697 2.098
2001.820000 1.103
END IONS


-----------------------------243501029130836
Content-Disposition: form-data; name="FORMAT"

Mascot generic
```

```
-----------------------------243501029130836
Content-Disposition: form-data; name="PRECURSOR"


-----------------------------243501029130836
Content-Disposition: form-data; name="INSTRUMENT"

ESI-QUAD-TOF
-----------------------------243501029130836
Content-Disposition: form-data; name="REPORT"

AUTO
-----------------------------243501029130836--
```

Results File

The results file contains the search results together with the search input parameters and peak list data. This means that a results file contains everything necessary to generate a report, repeat the search at a later date, or act as the self-contained input file to a project database or LIMS.

Mascot Parser provides an object-oriented Application Programmer Interface (API) to Mascot result files and configuration files, making it easy for programs written in C++, Java, Perl or Python to access Mascot results.

We strongly recommend that anyone writing software to process Mascot results uses Mascot Parser, just like all of the Mascot result report scripts:

- It makes application development much faster
- It makes your code simpler and easier to debug
- You don't have to worry about updating your code every time a new version of Mascot is released

The Mascot Parser package, which includes object libraries, header files, binary executables, extensive documentation, and example code for many functions, is available as a free download. For more information, go to http://www.matrixscience.com/msparser.html

For reference, the result file contents are divided into logical sections:

1. Search parameters
2. Mass values
3. Quantitation method (if used)
4. Unimod extract
5. Enzyme definition
6. Taxonomy (if a taxonomy filter was used)
7. Spectral library data (if a library was searched)
8. Misc. header information
9. Summary results (if PMF or small MS/MS)
10. Mixtures (if PMF)

11. Summary of decoy results (if automatic decoy)
12. Summary of error tolerant results (if automatic ET)
13. Summary of library results (if library search)
14. Mixtures in decoy results (if automatic decoy PMF)
15. Peptides (if SQ or MIS)
16. Decoy peptides (if SQ or MIS and automatic ET)
17. Error tolerant peptides (if SQ or MIS and automatic ET)
18. Library peptides (if library search)
19. Proteins (if SQ or MIS)
20. Query data, one block for each query
21. Index

General Notes

1. Values are shown in italics
2. Label case doesn't matter.
3. Labels are used to assist readability, but kept short to minimise file size
4. Parameters are grouped logically
5. Order of blocks is not important except that the index block must be the last block. Presence of blank lines within the index block may cause a problem.
6. Because the MIME type is defined as an unknown application, if this file passes through a mail agent, it will be treated as an "octet stream" and encoded "base64" for transmission.

Search parameters

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="parameters"

USERNAME=user name in plain text
USEREMAIL=email address in plain text
SEARCH=PMF
COM=search title text
DB=SwissProt
CLE=Trypsin
MASS=Monoisotopic
MODS=Mod 1,Mod 2
.
.
.
RULES=1,2,5,6,8,9,13,14
--gc0p4Jq0M2Yt08jU534c0p
```

The Parameters section contains the complete set of parameter values from the search form apart from the contents of the uploaded data file or the query window. Labels must be unique, independent of case. Where a parameter can be multivalued (e.g. mods) the values are listed on one line separated by commas.

RULES contains a list of the rule numbers that define the instrument type in the configuration file `fragmentation_rules`. The rule numbers are listed explicitly because the contents of the configuration file may have changed since the search was run.

Masses

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="masses"

A=71.037110
B=114.534940
C=160.030649
D=115.026940
E=129.042590
F=147.068410
G=57.021460
H=137.058910
I=113.084060
J=0.000000
K=128.094960
L=113.084060
M=131.040480
N=114.042930
O=0.000000
P=97.052760
Q=128.058580
R=156.101110
S=87.032030
T=101.047680
U=150.953630
V=99.068410
W=186.079310
X=111.000000
Y=163.063330
Z=128.550590
Hydrogen=1.007825
Carbon=12.000000
Nitrogen=14.003074
Oxygen=15.994915
Electron=0.000549
C_term=17.002740
N_term=1.007825
delta1=15.994919,Oxidation (M)
NeutralLoss1=0.000000
FixedMod1=57.021469, Carbamidomethyl (C)
FixedModResidues1=C
--gc0p4Jq0M2Yt08jU534c0p
```

This block contains “actual” mass values. That is, average or monisotopic residue masses, including any fixed modifications; C and N terminus groups also include any fixed modifications.

FixedMod1, FixedMod2, etc., records the delta mass and name for each fixed modification as comma separated values. FixedModResidues1 gives the site specificity. If multiple residues are affected, they are listed as a string, e.g. STY. If there was a neutral loss, the delta mass is given by the value of FixedModNeutralLoss1.

```
FixedModn=delta, Name
FixedModResiduesn=[A-Z]|C_term|N_term
FixedModNeutralLossn=mass
```

Fixed modifications cannot have peptide neutral losses, multiple neutral losses and cannot be protein-terminal or residue-terminal. In all these cases, fixed modifications are automatically converted into variable ones.

Variable modifications are reported in delta1, delta2, etc. Each entry defines the difference in mass introduced by the modification together with the name of the modification, separated by a comma. If a variable modification suffers a neutral loss on fragmentation, the delta is specified by a NeutralLoss*n* entry. By definition, this is always a master neutral loss. If there are multiple neutral losses, then two more lines appear:

```
NeutralLossn_master=mass[[,mass] ...]
NeutralLossn_slave=mass[[,mass] ...]
```

The first neutral loss (defined by NeutralLoss*n*) has an implicit index number of 1. Any additional neutral losses (defined by NeutralLoss*n*_master or followed by NeutralLoss*n*_slave) have implicit index numbers of 2 and up.

If a modification includes a required or optional neutral loss from the precursor, this is recorded as follows:

```
ReqPepNeutralLossn=mass[[,mass] ...]
PepNeutralLossn=mass[[,mass] ...]
```

Error-tolerant modifications are not listed in masses section.

Quantitation

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="quantitation"

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<quantitation majorVersion="1" minorVersion="0"
   xmlns="http://www.matrixscience.com/xmlns/schema/quantitation_1"
   xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.matrixscience.com/xmlns/schema/quantitatio
   n_1 qu
antitation_1.xsd">
  <method constrain_search="false" description="15N metabolic labelling"
   min_num_peptides="2" name="15N Metabolic [MD]" pro
t_score_type="mudpit" protein_ratio_type="weighted" report_detail="true"
   require_bold_red="true" show_sub_sets="0.5" sig_th
reshold_value="0.05">
    <component name="light">
      <isotope/>
  </component>
    <component name="heavy">
      <isotope>
        <old>N</old>
        <new>15N</new>
```

```
 </isotope>
```

This section is an extract from quantitation.xml containing the quantitation method specified for the search. For more details and a link to the schema, refer to the Mascot HTML help pages for quantitation.

Unimod

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="unimod"

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<umod:unimod xmlns:umod="http://www.unimod.org/xmlns/schema/unimod_2"
   majorVersion="2" minorVersion="0" xmlns:xsi="http://w
ww.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.unimod.org/xmlns/schema/unimod_2
   unimod_2.xsd">
  <umod:elements>
    <umod:elem avge_mass="1.00794" full_name="Hydrogen"
   mono_mass="1.007825035" title="H"/>
    <umod:elem avge_mass="2.014101779" full_name="Deuterium"
   mono_mass="2.014101779" title="2H"/>
    <umod:elem avge_mass="6.941" full_name="Lithium" mono_mass="7.016003"
   title="Li"/>
    <umod:elem avge_mass="12.0107" full_name="Carbon" mono_mass="12"
   title="C"/>
```

This section is an extract from unimod.xml containing data for the elements, amino_acids, and any modifications specified in the search form. For more details and a link to the schema, refer to the help pages at www.unimod.org

Enzyme

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="enzyme"

Title:Trypsin
Cleavage:KR
Restrict:P
Cterm
*
```

This section is simply an extract from the enzyme file. Syntax details can be found in Chapter 6

Taxonomy

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="taxonomy"

Title:. . . . . . . . . . . . . . . . Homo sapiens (human)
Include: 9606
Exclude:
*
```

This section is simply an extract from the taxonomy file. Syntax details can be found in Chapter 9

Spectral library

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="spectral_library"

Mod1=Acetyl,42.010565
Mod2=Carbamidomethyl,57.021464
Mod3=Gln->pyro-Glu,-17.026549
```

An entry is added for each modification in all the spectral libraries included in the search. If the name in the spectral library is identical to the name in the local unimod.xml file, then a delta is added to the line. The delta will be the average or monoisotopic delta, depending on the search condition, and not what was used when creating the spectral library.

Header

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="header"

sequences=551705
sequences_after_tax=7904
residues=197114987
distribution=7831,0,0,0,2,10,21,18,10,6,2,4
exec_time=69
date=1478689313
time=11:01:53
queries=8675
min_peaks_for_homology=6
max_hits=50
version=2.5.101
fastafile=C:/inetpub/…/NIST_S.cerevesiae_IonTrap_20120614.msp
release=NIST_S.cerevesiae_IonTrap_20120614.msp
db_type1=SL
sequences1=0
sequences_after_tax1=0
residues1=0
library_entries1=92609
sl_exec_command1=../bin/NIST/mspepsearch/MSPepSearch.exe …
library_reference_DB2=SwissProt
library_reference_fastafile2=C:/inetpub/…/SwissProt_2016_07.fasta
library_reference_release2=SwissProt_2016_07.fasta
library_reference_id=2
fastafile2=C:/inetpub/…/SwissProt_2016_07.fasta
release2=SwissProt_2016_07.fasta
db_type2=AA
sequences2=551705
sequences_after_tax2=7904
residues2=197114987
taskid=147868924401--gc0p4Jq0M2Yt08jU534c0p
```

The Header section contains general values, used in the master results page header paragraph.

The example is for a search of a Fasta file (SwissProt_2016_07.fasta) and a spectral library (NIST_S.cerevesiae_IonTrap_20120614.msp). `db_type` can be AA, NA, or SL. For a library, `sequences` is always 0 and `library_entries` gives the number of entries in the library msp file. This is not guaranteed to be the same as the number of entries actually searched. Each spectral library has an optional Reference Fasta Database. Peptide matches from the spectral library are looked up in the reference database and the accession from the refererence fasta is added to the peptide entry in the results file. All the reference fastas are listed in the header section and the 'id' is the database index in the library_peptides section. The command line used for the search is recorded in `sl_exec_command`.

`Distribution` is a comma separated list of values that represent a histogram of the complete protein score distribution. The first value is the number of entries with score 0, the second is the number of entries with score 1, and so on, up to the maximum score for the search. Scores are converted to integers by truncation. This distribution is only meaningful for a peptide mass fingerprint search.

If intensity values are supplied for a peptide mass fingerprint, Mascot iterates the experimental peaks to find the set that gives the best score. The number of values selected is reported in `pmf_num_queries_used` and the selected queries listed in `pmf_queries_used`.

Summary results

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="summary"

qmass1=Mr
qexp1=m/z for query 1,
charge
qintensity1=intensity value for query1 (if available)
qmatch1=Total number of peptide mass matches for query1 in database
qplughole1=Threshold score for homologous peptide match (MIS only)
qmass2=…
qexp2=…
qintensity1=
qmatch2=…
qplughole2=…
.
.
.
qmassn=…
qexpn=…
qintensityn=
qmatchn=…
qplugholen=…
num_hits=number of hits in the summary block (<= max_hits)
h1_db=index
h1=accession string,
total protein score,
obsolete,
intact protein mass
h1_text=title text
h1_frame=frame_number (between 1 and 6, for nucleic acid only)
h1_q1=missed cleavages, (-1 indicates no match)
peptide Mr,
delta,
start,
end,
number of ions matched,
peptide string,
peaks used from Ions1,
variable modifications string,
ions score,
multiplicity,
ion series found,
peaks used from Ions2,
peaks used from Ions3,
total area of matched peaks
h1_q1_et_mods=modification mass,
neutral loss mass,
modification description
h1_q1_et_mods_master=neutral loss mass[[,neutral loss mass] … ]
h1_q1_et_mods_slave=neutral loss mass[[,neutral loss mass] … ]
h1_q1_primary_nl=neutral loss string
h1_q1_na_diff=original NA sequence,
modified NA sequence
h1_q1_tag=tagNum:startPos:endPos:seriesID,…
h1_q1_drange=startPos:endPos
h1_q1_terms=residue,residue
h1_q1_subst=pos1,ambig1,matched1 … ,posn,ambign,matchedn
h1_p1_summed_mods=variable modifications string
h1_q2=…
.
.
.
h1_qm=…
h2=…
.
.
.
hn_qm=…
--gc0p4Jq0M2Yt08jU534c0p
```

Where a parameter has multiple values, these are shown on separate lines for clarity. In the actual result file, all values for a parameter are on a single line and there are no spaces or tabs between values.

Variable modifications is a string of digits, one digit for the N terminus, one for each residue and one for the C terminus. Each digit specifies the modification used to obtain the match: 0 indicates no modification, 1 indicates delta1, 2 indicates delta2 etc., in the masses section. If the number of modifications exceeds 9, the letters A to W are used to represent modifications 10 to 32. X is used to indicate a modification found in error tolerant mode.

neutral loss string is the same concept as the variable mod string, except each character represents the index of the primary neutral loss (one of the master NL). Any position that is not modified, or where the mod has no neutral loss, is set to 0. h*n*_q*m*_primary_nl will only be output if the string contains at least one non-zero character.

If a new modification is found in an error tolerant search, its position is marked by X, and details are recorded in an additional entry, h*n*_q*m*_et_mods. If the error tolerant search is of a nucleic acid database, and the modification is a single base change in the primary sequence, the two mass fields will be set to zero, and one of the keywords NA_INSERTION, NA_DELETION, or NA_SUBSTITUTION will appear in the description field. The additional parameter h*n*_q*m*_na_diff is then used to record the 'before' and 'after' nucleic acid sequences.

If the search includes a quantitation method and the search parameter MULTI_SITE_MODS is set to 1 then a single site can carry two modifications. When this occurs, a second modifications string, e.g. h1_p1_summed_mods, is used to record the additional modification(s).

Ion series is a string of 19 digits representing the ion series:

a

place holder
a++
b
place holder
b++
y
place holder
y++
c
c++
x
x++
z
z++
z+H
z+H++
z+2H
z+2H++

A digit is set to 1 if the corresponding series contains more than just random matches and 2 if the series contributes to the score.

Multiplicity means number of peptide mass matches for a query in a protein

For each sequence tag, four colon separated values are output: 1-based tag number, 1-based residue position where tag starts, 1-based residue position where tag ends, ion series into which the tag was matched:

-1 means no matches for the tag
0 "a" series (single charge)
1 "a-NH3" series (single charge)
2 "a" series (double charge)
3 "b" series (single charge)
4 "b-NH3" series (single charge)
5 "b" series (double charge)
6 "y" series (single charge)
7 "y-NH3" series (single charge)
8 "y" series (double charge)
9 "c" series (single charge)
10 "c" series (double charge)
11 "x" series (single charge)
12 "x" series (double charge)
13 "z" series (single charge)
14 "z" series (double charge)
15 "a-H2O" series (single charge)
16 "a-H2O" series (double charge)
17 "b-H2O" series (single charge)
18 "b-H2O" series (double charge)
19 "y-H2O" series (single charge)
20 "y-H2O" series (double charge)
21 "a-NH3" series (double charge)
22 "b-NH3" series (double charge)
23 "y-NH3" series (double charge)
25 "internal yb" series (single charge)
26 "internal ya" series (single charge)
27 "z+H" series (single charge)

28 "z+H" series (double charge)
29 high-energy "d" and "d'" series (single charge)
31 high-energy "v" series (single charge)
32 high-energy "w" and "w'" series (single charge)
33 "z+2H" series (single charge)
34 "z+2H" series (double charge)

If there are multiple tags for a query, comma separated groups of these numbers are output for each tag.

h*n*_q*m*_drange is output for a query that includes an error tolerant sequence tag. It defines the range of positions within which an unsuspected modification has been located. For a peptide of 10 residues, position 0 would indicate the amino terminus and position 11 would indicate the carboxy terminus. If there is no location information, the range is output as 0,256

h*n*_q*m*_terms shows the residues the bracket the peptide in the protein. If the peptide forms the terminus of the protein, then a hyphen is used instead.

h*n*_q*m*_subst is output when the matched peptide contained an ambiguous residue, (B, X, or Z). The argument is one or more triplets of comma separated values. For each triplet, the first value is the residue position, the second is the ambiguous residue, and the third is the residue that has been substituted to obtain the reported match.

For a large MS/MS search, `num_hits` is set to zero, and the summary block only contains entries for `qmass`*n*, `qexp`*n*, `qmatch`*n*, `qplughole`*n*. The threshold for switching to this mode is specified using two parameters in the Options section of *mascot.dat*. SplitDataFileSize is the size of the search process in bytes, (default 10000000), and SplitNumberOfQueries is the size of the search in queries, (default 1000).

If this is a library search, an automatic decoy database search, or an automatic error tolerant search, a second summary block appears, containing the second set of results. The section name is either library_summary, et_summary or decoy_summary. The syntax of the contents is identical

Mixture

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="mixture"

num_hits=number of mixtures found
h1_score=total score for mixture 1
h1_numprot=number of proteins in mixture 1
h1_nummatch=number of queries matched
h1_m1=accession string for protein component 1
h1_m2=accession string for protein component 2
.
.
.
h1_mm=accession string for protein component m
h2_score=
.
.
.
hn_mm=
--gc0p4Jq0M2Yt08jU534c0p
```

The Mixture section is only output for a peptide mass fingerprint. If any statistically significant protein mixtures are found, the mixture components are summarised. For details of individual components, use the accession strings to refer back to the Summary section.

If this is an automatic decoy database search, a second mixture block appears, containing the second set of results. The section name is decoy_mixture. The syntax of the contents is identical

Peptides

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="peptides"

q1_p1=missed cleavages, (-1 indicates no match)
peptide Mr,
delta,
number of ions matched,
peptide string,
peaks used from Ions1,
variable modifications string,
ions score,
ion series found,
peaks used from Ions2,
peaks used from Ions3;
"accession string": data for first protein
frame number:
start:
end:
multiplicity,
"accession string": data for second protein
frame number:
start:
end:
multiplicity,
   etc.
q1_p1_et_mods=modification mass,
neutral loss mass,
modification description
q1_p1_et_mods_master=neutral loss mass[[,neutral loss mass] … ]
q1_p1_et_mods_slave=neutral loss mass[[,neutral loss mass] … ]
q1_p1_primary_nl=neutral loss string
q1_p1_na_diff=original NA sequence,
modified NA sequence
q1_p1_tag=tagNum:startPos:endPos:seriesID,…
q1_p1_drange=startPos:endPos
q1_p1_terms=residue,residue[[:residue,residue] … ]
q1_p1_db=index[[index] … ]
q1_p1_subst=pos1,ambig1,matched1 … ,posn,ambign,matchedn
q1_p1_comp=quantitation component name
q1_p1_summed_mods=variable modifications string
q1_p2=…
.
.
.
qn_pm=…
--gc0p4Jq0M2Yt08jU534c0p
```

Each line contains the data for a peptide match followed by data for at least one protein in which the peptide was found.

If there multiple entries in the database containing the matched peptide, there will be a corresponding number of pairs of bracketing residues listed in q*n*_p*m*_terms.

Otherwise, individual field descriptions are identical to those for the Summary section

If this is a library search, an automatic decoy database search, or an automatic error tolerant search, a second peptides block appears, containing the second set of results. The section name is either library_peptides, et_peptides or decoy_peptides. The syntax of the contents is identical although many of the items on the q*n*_p*m* line are not applicable to library results, so set to 0.

A 'Y' is used in the variable modifications string to indicate a spectral library modification. An additional line of the form:

```
q1_p1_SLmod=position:mod,position:mod,...
```

is used to describe what the modification is. The name and offset of the modification can be found from the `ModX` line in the spectral_library section.

If flanking residues are not specified in a spectral library, the q1_p1_terms will be ?,?

Proteins

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="proteins"

index::"accession string"=protein mass,"title text"
.
.
.
index::"accession string"=protein mass,"title text"
--gc0p4Jq0M2Yt08jU534c0p
```

This block contains reference data for the proteins listed in the peptides block.

Input data for query n

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="queryn"

title=query title
index=query index
seq1=sequence qualifier (e.g. N-ABCDEF)
seq2=…
.
.
.
seqn=
comp1=composition qualifier (e.g. 0[P]2[W])
comp2=…
.
.
.
compn=…
PepTol=peptide tolerance qualifier (e.g. 2.000000,Da)
IT_MODS=Mod 1[,Mod 2[,…]]
```

```
INSTRUMENT=instrument identifier, (e.g. ESI-TRAP)
RULES=1,2,5,6,8,9,13,14
INTERNALS=min mass,max mass
CHARGE=charge state (e.g. 2+)
RTINSECONDS=a[[-b][,c[-d]]]
SCANS=a[[-b][,c[-d]]]
tag1=sequence tag (e.g. t,889.4,[QK]S,1104.54)
.
.
.
tagn=…
mass_min=lowest mass
mass_max=highest mass
int_min=lowest intensity
int_max=highest intensity
num_vals=number of mass values
num_used1=-1 (obsolete)
ions1=1344.65:34.3,1365.41:13.2
ions2=y-1344.65:34.3,1365.41:13.2
ions3=b-1344.65:34.3,1365.41:13.2

--gc0p4Jq0M2Yt08jU534c0p
```

Value “query*n*” runs from “query1” (no leading zeros). ions*n* values are sorted in the order that they were selected for scoring.

Most searches will only require a few of these fields. For example, a peptide mass fingerprint would only include the charge field.

The index is a 0 based record of the original query order before sorting by Mr

ions2 and ions3 are only required when fragment ions are specified in a sequence query as being N-terminal or C-terminal series.

The first field in a tagn value is t for a standard sequence tag and e for an error tolerant sequence tag

Some search parameters can be define in the local scope of a query. These are CHARGE, COMP, INSTRUMENT, IT_MODS, TOL, TOLU. Any that are used are listed here. If the MGF file contained scan range information in terms of seconds or scans, this is written to RTINSECONDS and/or SCANS.

Index

```
--gc0p4Jq0M2Yt08jU534c0p
Content-Type: application/x-Mascot; name="index"

parameters=4
masses=78
unimod=116
enzyme=322
taxonomy=329
header=336
summary=351
et_summary=6059
peptides=6473
et_peptides=7143
proteins=7292
query1=7362
query2=7374.
.
.
```

```
.
query81=8322
query82=8334
--gc0p4Jq0M2Yt08jU534c0p—
```

Values in index are the line number offsets of the section “Content-Type:” lines (starting from 0 for the first line of the file).

9

9. Taxonomy

Mascot supports the use of a taxonomy filter to select the database entries to be searched. This is useful because it speeds up the search, and can reduce the proteins in the results list to those expected in the sample being analysed.

Some databases record taxonomy in a manner that makes it difficult to extract the information reliably. The major problems are:

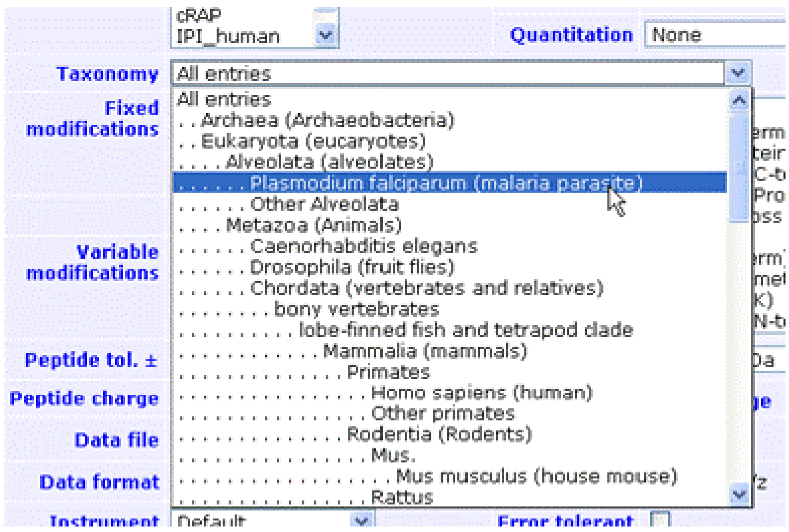
1. The location of the text containing the species identifier is undefined, and can vary within a database
2. There are often many names for one particular species - e.g. homo sapiens, human, man.
3. Names are sometimes misspelled - e.g. homo sapeins.
4. Continual re-classification of species is taking place
5. Some non-redundant databases only reliably give one species when several submissions from different species have identical sequences.
6. There are differences of opinion regarding the taxonomy 'tree' structure.

This section describes how the Mascot taxonomy filter works and how to configure it. Most of the configuration that will be required should be simple to change - for example the list of species displayed in the search form can be modified easily, and it is fairly simple to download updated taxonomy data files. On the other hand, modifying the configuration to extract taxonomy information in a novel way is a complex task that may take some time.

The NCBI keeps a list of taxonomy ID's up to date, and guarantees that the ID for a given species will not change (although some of the names used for that ID may change). Mascot configurations all use the NCBI IDs, although it would be possible to configure Mascot to use a different system.

Modifying the list in the search form

The list displayed in the search form is taken from the *taxonomy* file in the mascot *config* directory.



cRAP
IPI_human
Quantitation None
Taxonomy All entries
Fixed modifications
Variable modifications
Peptide tol. ±
Peptide charge
Data file
Data format
Instrument Default
Error tolerant

All entries
. . Archaea (Archaeobacteria)
. . Eukaryota (eucaryotes)
. . . . Alveolata (alveolates)
. . . . . . Plasmodium falciparum (malaria parasite)
. . . . . . Other Alveolata
. . . . Metazoa (Animals)
. . . . . . Caenorhabditis elegans
. . . . . . Drosophila (fruit flies)
. . . . . . Chordata (vertebrates and relatives)
. . . . bony vertebrates
. . . . lobe-finned fish and tetrapod clade
. . . . Mammalia (mammals)
. . . . Primates
. . . . Homo sapiens (human)
. . . . Other primates
. . . . Rodentia (Rodents)
. . . . Mus.
. . . . Mus musculus (house mouse)
. . . . Rattus

This file can be edited using any text editor. (Under Windows, from the start menu, choose Programs, Mascot, Config, Mascot taxonomy file).

The following is an extract from the supplied file:

```
Title:All entries
Include: 1
Exclude: 0
*
Title:. . Archaea (Archaeobacteria)
Include: 2157
Exclude:
*
Title:. . Eukaryota (eucaryotes)
Include: 2759
Exclude:
*
Title:. . . . Alveolata (alveolates)
Include: 33630
Exclude:
*
Title:. . . . . . . . . . . . . . Primates
Include: 9443
Exclude:
*
Title:. . . . . . . . . . . . . . . . Homo sapiens (human)
Include: 9606
Exclude:
*
Title:. . . . . . . . . . . . . . . . Other primates
Include: 9443
Exclude: 9606
*
```

The first line of each block must start with the `Title:` keyword, followed by a text string that is used to identify the species in forms and reports. The definition should be short and self-explanatory. To show the tree structure, indentation can be used. Unfortunately, it is not possible to use tabs or multiple spaces for indentation in an html form, so a full stop (period) and a space are used to indent the list. Internal spaces are significant, and there should never be two or more spaces together.

This should be followed with a definition line starting with the `Include:` keyword, followed by one or more NCBI taxonomy IDs separated with commas.

This should be followed with a definition line starting with the `Exclude:` keyword, followed by one or more NCBI taxonomy IDs separated with commas. Any sequence with a taxonomy ID that passes the 'include' test, may then be rejected by any entry in the exclude list.

Finally, each entry must end with a *

The NCBI taxonomy browser can be used to find the NCBI taxonomy ID for a given species:

http://www.ncbi.nlm.nih.gov/Taxonomy/

For example, to add a choice of Ferns or human, add the following to the taxonomy file:

```
Title:. Ferns or human
Include: 9606, 3263
Exclude:
*
```

And to add the choice of 'Not human or mice' add the following to the taxonomy file:

```
Title:. Not human or mice
Include: 1
Exclude: 9606, 10088
*
```

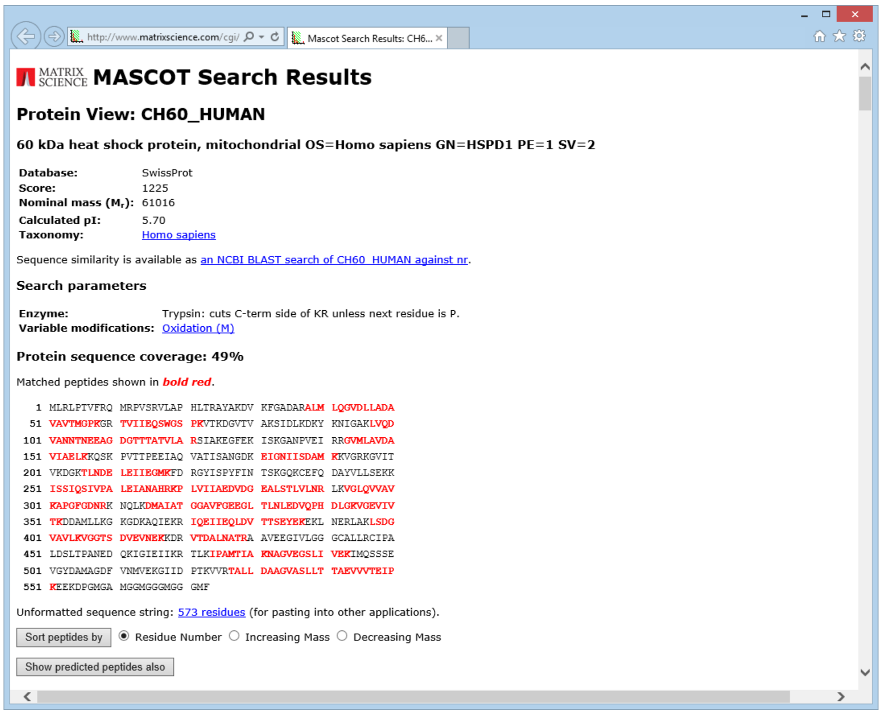
Note that 'all species' or root has the ID '1'.

It is, of course, possible to accidentally specify a selection that will result in no species matching - for example include humans, and exclude animals.

If you wish to include species in the taxonomy file without having them appear on the search form, the keyword 'Hidden' should appear on the line following the title line.

Modifying the "Taxonomy lineage' link

In the protein view, a link to taxonomy lineage is shown:



http://www.matrixscience.com/cgi/ | Mascot Search Results: CH6...

MATRIX SCIENCE **MASCOT Search Results**

Protein View: CH60_HUMAN

60 kDa heat shock protein, mitochondrial OS=Homo sapiens GN=HSPD1 PE=1 SV=2

Database:	SwissProt
Score:	1225
Nominal mass (M_r):	61016
Calculated pI:	5.70
Taxonomy:	Homo sapiens

Sequence similarity is available as an NCBI BLAST search of CH60_HUMAN against nr.

Search parameters

Enzyme:	Trypsin: cuts C-term side of KR unless next residue is P.
Variable modifications:	Oxidation (M)

Protein sequence coverage: 49%

Matched peptides shown in ***bold red***.

```
  1 MLRLPTVFRQ MRPVSRVLAP HLTRAYAKDV KFGADARALM LQGVDLLADA
 51 VAVTMGPKGR TVIIEQSWGS PKVTKDGVTV AKSIDLKDKY KNIGAKLVQD
101 VANNTNEEAG DGTTTATVLA RSIAKEGFEK ISKGANPVEI RRGVMLAVDA
151 VIAELKKQSK PVTTPEEIAQ VATISANGDK EIGNIISDAM KKVGRKGVIT
201 VKDGKTLNDE LEIIEGMKFD RGYISPYFIN TSKGQKCEFQ DAYVLLSEKK
251 ISSIQSIVPA LEIANAHRKP LVIIAEDVDG EALSTLVLNR LKVGLQVVAV
301 KAPGFGDNRK NQLKDMAIAT GGAVFGEEGL TLNLEDVQPH DLGKVGEVIV
351 TKDDAMLLKG KGDKAQIEKR IQEIIEQLDV TTSEYEKEKL NERLAKLSDG
401 VAVLKVGGTS DVEVNEKKDR VTDALNATRA AVEEGIVLGG GCALLRCIPA
451 LDSLTPANED QKIGIEIIKR TLKIPAMTIA KNAGVEGSLI VEKIMQSSSE
501 VGYDAMAGDF VNMVEKGIID PTKVVRTALL DAAGVASLLT TAEVVVTEIP
551 KEEKDPGMGA MGGMGGGMGG GMF
```

Unformatted sequence string: 573 residues (for pasting into other applications).

Sort peptides by
[● Residue Number] ○ Increasing Mass ○ Decreasing Mass
Show predicted peptides also

The default behaviour is for this to link to the NCBI taxonomy browser. For non-redundant databases with more than one species source per sequence, there will be a list of the species, each with a link. For the NCBI nr database, a separate accession will be shown for each database entry, with a link to Entrez and the NCBI Taxonomy browser for each entry.

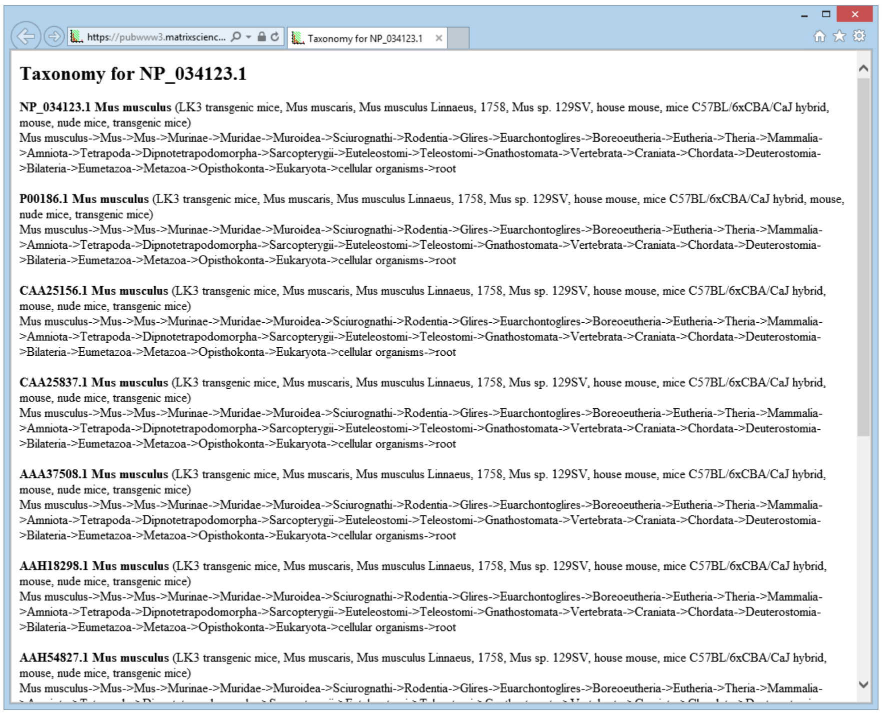
If security and confidentiality protocols make this unacceptable for your installation, then change the entry in the Options section of the *mascot.dat* file from:

```
TaxBrowserUrl https://www.ncbi.nlm.nih.gov/htbin-
  post/Taxonomy/wgetorg?lvl=0&lin=f&id=#TAXID#
```

to

```
TaxBrowserUrl ../x-cgi/ms-
  gettaxonomy.exe?4+#DATABASE#+#ACCESSION#
```

In this case, the link will display the information in the following format



Taxonomy for NP_034123.1

NP_034123.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Dipnotetrapodomorpha->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Bilateria->Eumetazoa->Metazoa->Opisthokonta->Eukaryota->cellular organisms->root

P00186.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Dipnotetrapodomorpha->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Bilateria->Eumetazoa->Metazoa->Opisthokonta->Eukaryota->cellular organisms->root

CAA25156.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Dipnotetrapodomorpha->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Bilateria->Eumetazoa->Metazoa->Opisthokonta->Eukaryota->cellular organisms->root

CAA25837.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Dipnotetrapodomorpha->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Bilateria->Eumetazoa->Metazoa->Opisthokonta->Eukaryota->cellular organisms->root

AAA37508.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Dipnotetrapodomorpha->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Bilateria->Eumetazoa->Metazoa->Opisthokonta->Eukaryota->cellular organisms->root

AAH18298.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia->Amniota->Tetrapoda->Dipnotetrapodomorpha->Sarcopterygii->Euteleostomi->Teleostomi->Gnathostomata->Vertebrata->Craniata->Chordata->Deuterostomia->Bilateria->Eumetazoa->Metazoa->Opisthokonta->Eukaryota->cellular organisms->root

AAH54827.1 Mus musculus (LK3 transgenic mice, Mus muscaris, Mus musculus Linnaeus, 1758, Mus sp. 129SV, house mouse, mice C57BL/6xCBA/CaJ hybrid, mouse, nude mice, transgenic mice)
Mus musculus->Mus->Mus->Murinae->Muridae->Muroidea->Sciurognathi->Rodentia->Glires->Euarchontoglires->Boreoeutheria->Eutheria->Theria->Mammalia-

Other parameters are possible for ms-gettaxonomy.exe - see the reference section ‘ms-gettaxonomy’ in Chapter 7.

Common Questions

Why do I sometimes get results for a species I didn’t specify?

Sometime, when specifying for example ‘Human’ species, the results may appear at first sight to be from for example a Mouse sample. The most common reason for this is that, for a non-redundant database, exactly the same sequence has been found in many species. To check this, look at the protein view, where you should see at least one entry for the species you selected.

What is the “unclassified” and “other” species?

The NCBI cannot always classify every sequence - either because no species information was supplied with the data or because it doesn’t fit into any current classification.

“Other” species include plasmids, and artificial sequences.

How do I see which sequences Mascot couldn’t assign a taxonomy ID?

In the status screen, click on the “Unidentified taxonomy” link. This will show sequences where one or more of the species names were not identified by Mascot.

Why do I get the message “Taxonomy 'xxx' ignored. No taxonomy indexes for this database”

Taxonomy is not necessary for a database where all the entries come from the same species. When searching such a database, leave the taxonomy filter setting at ‘All entries’. If you see this message for a database where taxonomy would be useful, change the configuration in Database Manager then choose ‘Recompress file’ in Database Status.

Taxonomy data files

NCBI format

NCBI creates two files that list all the species for which they have one or more sequences. *names.dmp* is a list of scientific names, synonyms and misspellings for the species. From this list, you can easily find the ID for the given species. For example:

```
3701 | Arabidopsis          | | scientific name       |
3701 | Cardaminopsis        | | synonym               |
3702 | Arabidopsis thaliana | | scientific name       |
3702 | Arbisopsis thaliana  | | misspelling           |
3702 | thale cress          | | preferred common name |
3702 | thale-cress          | | common name           |
3702 | mouse-ear cress      | | common name           |
```

The file *nodes.dmp* specifies the tree structure. The first column is a taxonomy ID and the second column is the parent taxonomy ID. Note that the ‘parent’ of Arabidopsis thaliana (3702) is Arabidopsis (3701).

```
3700 | 3699 | family  |    | 4 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
3701 | 3700 | genus   |    | 4 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
3702 | 3701 | species | AT | 4 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
```

Both files can be obtained from the NCBI ftp site by downloading this archive:

ftp://ftp.ncbi.nih.gov/pub/taxonomy/taxdump.tar.gz

You should not modify the *names.dmp* and *nodes.dmp* file in the taxonomy directory because these files will be replaced the next time a database is updated. If you wish to add more entries, a new file should be made with just the new entries, and listed in the taxonomy definition block, as illustrated below.

In the same archive, *merged.dmp* is supplementary to *nodes.dmp* and contains “late additions”. File *est.dmp* contains a small number of species in the same format as *names.dmp* and is used for reverse lookups.

GI2TAXID format

These two files allowed taxonomy IDs to be looked up using gi numbers. They became obsolete in 2016 when NCBI dropped using gi numbers

ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/gi_taxid_prot.dmp.gz

ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/gi_taxid_nucl.dmp.gz

PDB format

A lookup for entries derived from the Brookhaven Protein databank (PDB)

ftp://ftp.ncbi.nlm.nih.gov/mmdb/pdbeast/tax.table

ACC2TAXID format

EBI builds a file that links EMBL accession to taxonomy ID

ftp://ftp.ebi.ac.uk/pub/databases/embl/misc/acc_to_taxid.mapping.txt.gz

This file has two values per line, accession string and taxonomy ID, separated by white space. For example:

```
A00001  10641
A00002  9913
```

where A00001 and A00002 are accessions and 10641 is the NCBI taxonomy ID for "Cauliflower mosaic virus" and 9913 is the NCBI taxonomy ID for "Bos taurus"

Performance when creating the compressed files is faster if the order of entries in the taxonomy file is the same as the order of sequences in the Fasta file. When the ACC2TAXID file is first used or is updated, lookup files (.cdb) are created in the taxonomy directory. These files are only used when compressing the database and are not required when running a search.

NCBI builds two files that allow a taxonomy ID to be obtained direct from an accession.version string

ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/accession2taxid/prot.accession2taxid.gz

ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/accession2taxid/nucl_est.accession2taxid.gz

These files have a four column format that is not compatible with older versions of Mascot, so the latest prot.accession2taxid.gz file is re-formatted with two columns and made available for use with NCBI protein databases on this URL

http://www.matrixscience.com/downloads/prot.av2taxid.gz

SWISSPROT format

UniProt creates a file, `speclist.txt`, that is similar to the NCBI `names.dmp` file, except that it gives the NCBI taxonomy ID for the UniProt organism code.

```
Code    Taxon   N=Official name
        Node    C=Common name
                S=Synonym
_____ _ _______ ___________________________________________
AAV2  V 010804: N=Adeno-associated virus 2
                C=AAV2
ABDS2 B 056673: N=Antarctic bacterium DS2-3R
ABIAL E 045372: N=Abies alba
                C=Edeltanne
```

This file is available at:

ftp://ftp.ebi.ac.uk/pub/databases/uniprot/knowledgebase/docs/speclist.txt

If you wish to add more entries, a new file should be made with just the new entries. Mascot will load multiple files as described below.

Genetic code selection

During a search of a nucleic acid database, Mascot can use the taxonomy of each entry to choose the correct genetic code for translation. The genetic codes are defined in the NCBI file *gencode.dmp*, which is included in the archive *taxdump.tar*, mentioned above.

Nodes.dmp is used as a lookup table to obtain a genetic code number from a taxonomy ID.

For many species, the genetic code is different for mitochondrial and nuclear proteins. Although Mascot could try to determine whether a database entry is mitochondrial by performing a keyword search of the FASTA description, this is unreliable. In any case, mitochondrial proteins will usually represent only a very small fraction of the entries in any comprehensive database. The most important requirement is to use the correct code for a database that is specifically mitochondrial proteins. The solution is to include a flag in each *mascot.dat* taxonomy definition to specify, at a database level, which code is to be used.

For further information on genetic codes, see

http://www.ncbi.nlm.nih.gov/Taxonomy/Utils/wprintgc.cgi?mode=c

How Mascot gets a taxonomy ID for a database entry

This section contains detailed configuration information. It is only necessary to read and understand this section if you need to extract taxonomy information from a database in a novel way.

When Mascot Monitor creates the compressed files for a database, it optionally creates an index file containing the taxonomy ID(s) for each entry. The rules for how to do this are specified in numbered taxonomy blocks in mascot.dat. The taxonomy block number to be used for a database is specified as the 14th parameter in the *databases* section of the mascot.dat file.

Each block is a series of lines, starting with a keyword and optionally followed arguments. A block begins with the keyword Taxonomy_n, where n is an integer, and ends with the keyword End. Within a block, the order of the lines is unimportant. You can add comment lines and append comments after arguments - everything between a # character and the end of the line is ignored

All text searches and comparisons are case insensitive, except where stated. Taxonomy definition keywords in the mascot.dat file are also case insensitive. Regex means a Basic Regular Expression, as described in Appendix A.

If you wish to modify an existing block or add a new one ***and have never used Database Manager***, you can edit mascot.dat directly. This is not possible once Database Manager has been used because mascot.dat is re-written every time there are configuration changes. The procedure to make changes to taxonomy blocks when Database Manager is in use is described below in the section 'Taxonomy in Database Manager'.

Keywords

AccFromSpeciesLine

Regex used to extract an accession string from a species line in an annotations file or from the second and subsequent titles in multi-title Fasta. This is required because the 'main' accession parse rule only captures the primary accession for each entry. A different rule is needed to capture the secondary accessions

ConcatRefFileLines

0 - do not concatenate lines from the annotations file that begin with the same keyword

1 - concatenate lines from the annotations file that begin with the same keyword

(Only applies to QuickRefSearch)

DefaultRule

Regex used to extract the information that can be used to determine taxonomy ID when no other RULE_xxx applies

DescriptionLineSep

If taxonomy is taken from the Fasta and the database has entries with multiple concatenated title lines, this specifies the character used to delimit them by its ASCII code. For example, CTRL+A is ASCII code 1, which is the delimiter used in NCBI nr

If taxonomy is taken from the annotations file and there are entries associated with multiple species, this specifies the character used to delimit them by its ASCII code. For example, 44 is a comma.

DoThisRuleFirst

A regex that pre-processes the line matched by QuickRefSearch and tries to match the extracted string in NCBI:names.dmp. If this fails, then the extracted string is passed to DefaultRule

Enabled

0 - definition disabled

1 - definition active

End

Defines the end of a taxonomy block

ErrorLevel

0 - a single entry is added to the 'NoTaxonomyMatch.txt' file for every sequence that has no taxonomy information.

1 - an entry is added to the 'NoTaxonomyMatch.txt' file for every title line that has no taxonomy information. Since sequences in NCBInr can have hundreds or even thousands of title lines, this can make the 'NoTaxonomyMatch.txt' file very large.

FromRefFile

0 - taxonomy is taken from the title lines in the Fasta file

1 - taxonomy is taken from the annotations file (e.g. SwissProt *.dat file)

GencodeFiles

The file specifying genetic codes to be used to translate NA entries to AA. The filename should not include a path, and the specified file should be saved in the taxonomy directory. If no file is specified, and one is required, the default 'gencode.dmp' will be loaded and used. Up to 10 files may be specified.

Identifier

Free text that identifies the taxonomy block. Displayed in a drop down list in Database Manager. Must be unique.

MitochondrialTranslation

0 - translate using the genetic code for nuclear proteins

1 - translate using the genetic code for mitochondrial proteins

NoBreakDescLineIf

Only used when taxonomy is taken from an annotations file and a single entry may have multiple species. If the separator is a common character like a comma, it may not be desirable to break on this character in all cases. For example:

```
C;Species: A.hydrophila DNA, clone pPH4
```

NoBreakDescLineIf defines a comma separated list of words that inhibit breaking of the line when they follow the separator or the separator followed by a space. This entry can be repeated as often as required to accommodate large numbers of words.

NodesFiles

Lists the file(s) that define the hierarchy of taxonomy IDs. The supported file is described earlier in this chapter. Filenames are prefixed with a format identifier. The filename should not include a path, and the specified file should be saved in the taxonomy directory. Up to 10 files may be specified.

PrefixRemoves

List of space separated words used if CHOP:WP is specified in Rule_xxx or DefaultRule

QuickRefSearch

Rather than use a regular expression for each line in an annotations file, looking for taxonomy information, this text is used to select the line that begins with the specified string. Other lines are ignored. Note that this is a case sensitive compare, so it must be specified using the correct upper and lower case letters.

Rule_xxx

Regex selected by SrcDatabaseRule to extract the information that can be used to determine taxonomy ID. There are three parts to the argument.

First a keyword that represents the format of the file(s) listed in SpeciesFiles used to translate the taxonomy information into a taxonomy ID:

ACC2TAXID - use the lookup file labelled ACC2TAXID:
EXPLICIT - the regex returns a taxonomy ID directly
GI2TAXID - obsolete
NCBI - use the names.dmp format file(s) labelled NCBI:
PDB - use the tax.table format names file labelled PDB:
SWISSPROT - use the speclist.txt format file labelled SWISSPROT:

Second, after a comma, a term that controls processing of the string returned by the regex in the third part of the argument:

CHOP: Use the complete string
CHOP:W If necessary, remove one word at a time from the end of the string until a match is found
CHOP:WP All words listed in PrefixRemoves (plus any trailing space) should be removed before trying to find a match. Then, if necessary, remove one word at a time from the end of the string until a match is found
CHOP:S If any of the words in SuffixRemoves are found, then this word and all of the following text is removed before trying to find a match.

Third, after whitespace, a regex used to extract a string from the Fasta title or the string in the annotations file extracted by DoThisRuleFirst

SpeciesFiles

Lists the file(s) to be used to convert a taxonomy string into a taxonomy ID. Supported files are described earlier in this chapter. Each filename is prefixed with a format identifier.

Note that NCBI:names.dmp must always be listed, even if not used to obtain a taxonomy ID from the Fasta, because it is required by the ms-gettaxonomy utility

SrcDatabaseRule

If titles come from multiple source databases, it may be necessary to use a variety of regular expressions to parse out the taxonomy information. SrcDatabaseRule specifies a regex that can be used to select a specific rule according to the database source. For example, maybe there are two source databases called chalk and cheese, and typical title lines look like this

```
>chalk|A49190 corticosteroid-binding globulin - sheep
>cheese|NT_1299634 coat protein [East African cassava mosaic Malawi virus]
```

The regex ">\([^|]*\)" would return either chalk or cheese. The rules used for retrieving the taxonomy information could then be identified by Rule_chalk and Rule_cheese. In addition, it would be wise to include a DefaultRule to handle titles from other sources, even if this is not expected.

StrStrFiles

The file(s) used for the reverse search lookup specified by StrStrRule. Filenames are prefixed with a format identifier.

StrStrRule

If all rules fail, including the DefaultRule, this allows a reverse lookup to be performed. Each entry from the file(s) listed in StrStrFiles is searched for in the text returned by the regex. For example, if the title was

```
>gi|2148255|gb|D11734.1|D11734 HUM000HE06 Liver HepG2 cell line. Homo
   sapiens cDNA clone he06, mRNA sequence
```

And the taxonomy definition was

```
StrStrRule     NCBI, CHOP:S "gi\|[^ ]* \(.*\)"
```

And the file listed in StrStrFiles contained Homo sapiens this would give a match and return taxonomy ID 9606. If multiple entries match, e.g. 'Rattus' and 'Rattus norvegicus', then the longest match is taken, 'Rattus norvegicus'.

This should only be used as rule of last resort, and can easily give a misleading result. For example, if the title contained 'sequence from pig but similar to homo sapiens' it could return 'homo sapiens' unless SuffixRemoves included 'similar'.

SuffixRemoves

List of space separated words used if CHOP:S is specified in Rule_xxx or DefaultRule

SwissProtRegex

A regular expression used to extract the "Code" and "Taxon Node" from the speclist.txt file. This should be defined as follows in any taxonomy block that lists speclist.txt on the SpeciesFiles line:

```
SWISSPROTRegex      "^\([A-Z0-9]*\) *[ABEV] *\([0-9]*\):"
```

TaxID

Specify a single taxonomy ID that applies to all entries in the database. Mainly used for single species NA databases where taxonomy is only required to select the correct genetic code

Taxonomy_n

Defines the beginning of a taxonomy block

NCBIprot

This is the comprehensive, non-identical protein database from NCBI.

```
Taxonomy_n
Enabled                 1
Identifier              NCBI nr using prot.accession2taxid
FromRefFile             0
DescriptionLineSep      1   # ctrl a - hex code 1
ErrorLevel              0
DefaultRule             ACC2TAXID, CHOP: "^>*\([^> ,]*\)"
NodesFiles              NCBI:nodes.dmp,NCBI:merged.dmp
SpeciesFiles            ACC2TAXID:prot.av2taxid,NCBI:names.dmp
AccFromSpeciesLine      "^>*\([^> ,]*\)"
End
```

FromRefFile is set to 0 because there is no annotations file for nr.

An entry may have multiple title lines separated by CTRL+A, so DescriptionLineSep is set to 1, the ASCII character code for CTRL+A.

There are two SpeciesFiles - prot.accession2taxid and names.dmp, and two NodesFiles - nodes.dmp and merged.dmp. All four files must be present and up-to-date.

The method of finding the taxonomy ID is particularly simple. DefaultRule is applied to each Fasta title line to extract the accession.version identifier. ACC2TAXID specifies that this can be looked up in prot.accession2taxid to return the taxonomy ID.

The regex in DefaultRule is a little more complicated than might be expected because only the first title for an entry starts with a 'greater than' character. This means that we need to match everything from the beginning of the title up to the first space but excluding the 'greater than' character if present.

AccFromSpeciesLine is used to extract accessions from the second and subsequent title lines when an entry has multiple titles. It is identical to the regex in DefaultRule because the accession.version string is also the information used to determine the taxonomy ID

SwissProt Fasta

The rules for Uniprot databases are fairly simple. This taxonomy block should always be selected for SwissProt and Trembl, even if you have a local annotations file.

```
Taxonomy_n
Identifier          SwissProt FASTA
Enabled             1        # 0 to disable it
ErrorLevel          0
FromRefFile         0
SpeciesFiles        NCBI:names.dmp, SWISSPROT:speclist.txt
NodesFiles          NCBI:nodes.dmp, NCBI:merged.dmp
DefaultRule         SWISSPROT, CHOP: ">[^_]*_\([^ ]*\) "
SWISSPROTRegex      "^\([A-Z0-9]*\) *[ABEV] *\([0-9]*\):"
End
```

There is only one database source, so the DefaultRule can be used. This takes everything after the first underscore to the next space. For example:

```
>sp|Q6GZX4|001R_FRG3G Putative transcription factor 001R OS=Frog virus 3
   (isolate Goorha) GN=FV3-001R PE=4 SV=1
```

Would find the text FRG3G, which it would look up in speclist.txt, returning a taxonomy ID of 654924.

EMBL EST

```
Taxonomy_n
Identifier          EMBL EST Fasta
Enabled             1   # 0 to disable it
FromRefFile         0
ErrorLevel          0
SpeciesFiles        ACC2TAXID:acc_to_taxid.mapping.txt, NCBI:names.dmp
NodesFiles          NCBI:nodes.dmp, NCBI:merged.dmp
DefaultRule         ACC2TAXID,      CHOP: ">EM_EST:\([A-Z0-9]*\)"
GencodeFiles        NCBI:gencode.dmp
MitochondrialTranslation 0
End
```

The ACC2TAXID identifier is used to identify the file that contains a simple mapping of accession to taxonomy ID.

The nodes.dmp file allows genetic code to be determined from taxonomy ID and MitochondrialTranslation is set to 0 specifying that the genetic code for nuclear proteins should be used.

Species specific nucleic acid databases

A nucleic acid database for a single species still needs taxonomy to be defined at the database level, so that the correct genetic code can be chosen.

```
Taxonomy_n
Identifier               All human with TaxID 9606
Enabled                  1   # 0 to disable it
SpeciesFiles             NCBI:names.dmp
NodesFiles               NCBI:nodes.dmp, NCBI:merged.dmp
GencodeFiles             NCBI:gencode.dmp
MitochondrialTranslation 0
TaxID                    9606
End
```

MitochondrialTranslation is set to 0 and TaxID is set to 9606, specifying that all database entries are homo sapiens. So, genetic code 1, (standard), will be selected for all entries.

HUPO PSI PEFF Format

The HUPO Proteomics Standards Initiative PEFF Fasta format is described in http://www.psidev.info/index.php?q=node/363

```
Taxonomy_n
Identifier          HUPO PSI PEFF Format
Enabled             1   # 0 to disable it
FromRefFile         0
ErrorLevel          0
SpeciesFiles        NCBI:names.dmp
NodesFiles          NCBI:nodes.dmp, NCBI:merged.dmp
DefaultRule         EXPLICIT, CHOP: "\\NcbiTaxId=\([0-9]*\)"
End
```

The NCBI taxonomy ID is parsed directly from the title line, e.g.

```
>nxp:NX_Q15029-1 \DbUniqueId=NX_Q15029-1 \Pname=116 kDa U5 small nuclear
   ribonucleoprotein component isoform Iso 1 \Gname=EFTUD2 \NcbiTaxId=9606
   \TaxName=Homo Sapiens \Length=972 …
```

Taxonomy in Database Manager

There is no user interface in Database Manager for taxonomy blocks. If you need to make changes to a block, or add a new one, the procedure is:

1. Ensure the task queue is empty, so that no Database Manager script will run behind your back.

2. Make the required changes by editing and saving mascot\config\db_manager\configuration.xml
3. In Database Manager, make some change that forces mascot.dat to be re-written. For example, deactivate then activate any database.
4. Choose 'Recompress file' in Database Status for any database that uses the changed or new taxonomy block

Database Manager configuration files

In mascot\config\db_manager\public, databases_1.xml and libraries_1.xml are the original definitions files from when Mascot was first installed.

If a later version of databases_1.xml or libraries_1.xml is available on the Matrix Science public web site, it is downloaded and saved to the public folder with the timestamp as the name, e.g. databases_20160902T140934.xml.

mascot\config\db_manager\configuration.xml is the local configuration file from which mascot.dat is generated. All editable Database Manager configuration information is stored in this file.

Database configuration entries in configuration.xml refer to individual definitions files in mascot\config\db_manager\public\ by means of the 'inherit_from' attribute.

As a simple example, we'll add a new taxonomy block for an e. coli nucleic acid database because this requires a non-standard genetic code for translation. In mascot.dat, it will look like this

```
Taxonomy_n
Enabled 1
Identifier e. coli with TaxID 562
FromRefFile 0
DescriptionLineSep 0
ConcatRefFileLines 1
TaxID 562
ErrorLevel 1
MitochondrialTranslation 0
GencodeFiles NCBI:gencode.dmp
NodesFiles NCBI:nodes.dmp,NCBI:merged.dmp
SpeciesFiles NCBI:names.dmp
End
```

Make a backup copy of configuration.xml then open the active file in a text editor. Search for the taxonomy_entries closing tag

```
</msgd:taxonomy_entries>
```

Unless you have already made changes, you'll see that all of the standard taxonomy blocks are inherited from the download of the latest databases_1.xml.

Our new entry will not be inherited, but will be explicitly defined in configuration.xml. Immediately before the taxonomy_entries closing tag, add this text:

```
<msgd:taxonomy_entry name="e. coli with TaxID 562">
  <msgd:ConcatRefFileLines>1</msgd:ConcatRefFileLines>
```

```
        <msgd:DBLevelTaxId>562</msgd:DBLevelTaxId>
        <msgd:DefaultRule>
          <msgd:FileTypeToSearch>NCBI</msgd:FileTypeToSearch>
        </msgd:DefaultRule>
        <msgd:DescriptionLineSep>0</msgd:DescriptionLineSep>
        <msgd:Enabled>1</msgd:Enabled>
        <msgd:ErrorLevel>1</msgd:ErrorLevel>
        <msgd:GencodeFiles>
          <msgd:GencodeFile
    remote_source="remote:taxdump.tar.gz_2">
            <msgd:FileName>gencode.dmp</msgd:FileName>
            <msgd:Format>GENCODE</msgd:Format>
          </msgd:GencodeFile>
        </msgd:GencodeFiles>

    <msgd:MitochondrialTranslation>0</msgd:MitochondrialTransla
    tion>
        <msgd:NodesFiles>
          <msgd:NodesFile
    remote_source="remote:taxdump.tar.gz_2">
            <msgd:FileName>nodes.dmp</msgd:FileName>
            <msgd:Format>NCBI</msgd:Format>
          </msgd:NodesFile>
          <msgd:NodesFile
    remote_source="remote:taxdump.tar.gz_2">
            <msgd:FileName>merged.dmp</msgd:FileName>
            <msgd:Format>NCBI</msgd:Format>
          </msgd:NodesFile>
        </msgd:NodesFiles>
        <msgd:SpeciesFiles>
          <msgd:SpeciesFile
    remote_source="remote:taxdump.tar.gz_2">
            <msgd:FileName>names.dmp</msgd:FileName>
            <msgd:Format>NCBI</msgd:Format>
          </msgd:SpeciesFile>
        </msgd:SpeciesFiles>
      </msgd:taxonomy_entry>
```

Search for the repositories closing tag

```
  </msgd:repositories>
```

And add the download information for taxdump.tar.gz immediately before

```
  <msgd:remote_source name="remote:taxdump.tar.gz_2">
    <msgd:uri>ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/taxdump.t
  ar.gz</msgd:uri>
  </msgd:remote_source>
```

(The name is remote:taxdump.tar.gz_2 to avoid a collision with the remote_source for the same file in databases_1.xml)

Save configuration.xml. You will find that can now select this taxonomy in Database Manager when configuring a custom database.

The easiest way to determine the XML syntax for a line in a taxonomy block is by comparing a taxonomy_entry in the download of the latest databases_1.xml with your current mascot.dat. You can also look at the XML file schema and its annotations, ms_mascot_configuration_1.xsd. A copy of this can be found on your Mascot Server under mascot/html/xmlns/schema/ms_mascot_configuration_1

10

10. Mascot Daemon

Overview

Mascot Daemon is a client application that automates the submission of searches to a Mascot Server. Functionality includes:

1. Batch mode, in which an arbitrary group of files can be defined for searching, either immediately or at some pre-set time.
2. Real-time monitor mode, in which new files on a pre-defined path are searched as they are created.
3. Data dependent follow-up tasks. For example, automatically repeating an unsuccessful search at a later date or against a different sequence database.

Tasks

The functional unit of Mascot Daemon is a task. A task can be created or modified in the Task Editor. A task is defined by:

1. The data source (e.g. a file list or a file path)
2. How the data are to be searched (an associated set of search parameters)
3. When the searches are to take place
4. Any follow-up activities, such as conditional repeat searches.

Tasks can be in one of four states: running, paused, completed, or cancelled. A paused task can be resumed. A paused or completed task can be cancelled or deleted.

Data Files

Data files can be any of the peak list formats supported by Mascot. Other types of file, such as binary data, can be specified if an appropriate data import filter is available:

1. A wide range of native file formats can be processed using Mascot Distiller, (requires an additional licence).
2. AB Sciex MS Data Converter supports both Wiff files and TOF/TOF data. Output can be an MGF peak list file or an mzML file containing a more complete representation of the data.
3. Thermo ExtractMsn.exe can be used to peak pick Xcalibur Raw files.
4. ProteoWizard msconvert is a command line tool for converting between various file formats.

Flexibility

Several Mascot Daemon clients can submit searches to a single Mascot Server, and can even share a common task database. If you have several mass spectrometers, you can choose whether to install separate copies of Daemon on each instrument data system or whether to have a single copy of Daemon somewhere on the LAN, marshalling searches for all instruments.

User Help

Mascot Daemon includes comprehensive, context sensitive on-line help. Press F1 at any time to jump to the relevant topic.

Installation

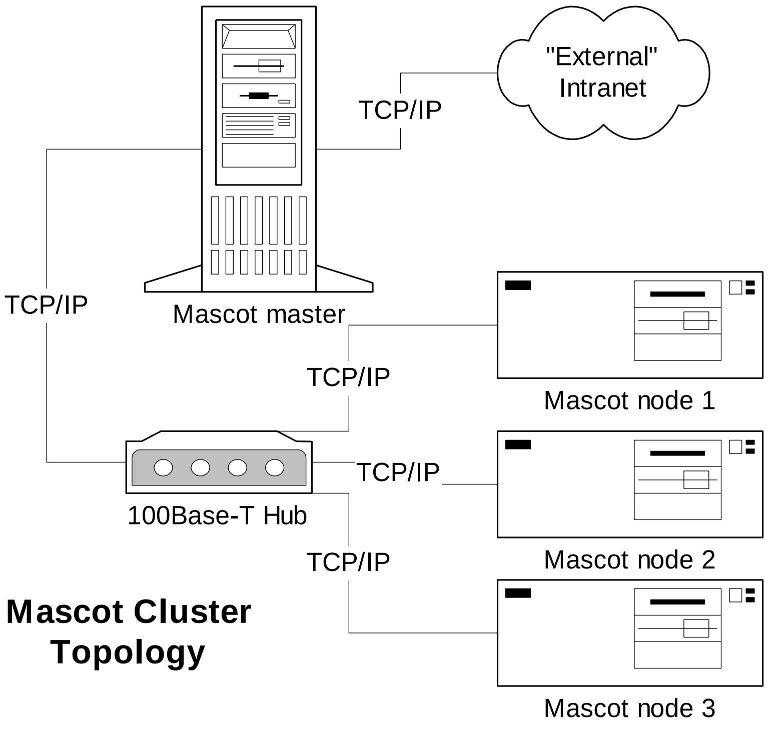
After Mascot Server has been installed, go to your local home page for links to a help page that describes how to install, upgrade or troubleshoot Mascot Daemon. All the required installation files are hyperlinked from this page.

11

11. Cluster Mode

Introduction

Mascot has been designed and implemented to work efficiently on a cluster of computers. A cluster of single or dual processor boxes provides a highly cost effective solution for high throughput protein identification. Mascot can be run in cluster mode on all supported hardware platforms and operating systems.



Hardware Requirements

All machines in a cluster should have processors of the same speed. Otherwise, the box with the slower processor(s) will become a bottleneck.

Two network ports are required on the master server; one for external access and one for communication on the private local area network (LAN) that connects the master to the nodes. The LAN for the cluster should run at least at 1000Mb/s.

The total amount of RAM required in a cluster is a function of how many sequence databases need to be held in memory simultaneously. Mascot supports an unlimited number of databases, but only those that are searched frequently justify being locked in memory. The others can be allowed to swap in and out of memory as needed. For example, a 5 node, 8 processor cluster (non-searching head node) might have 16 GB on the master, and 8 GB on each of the search nodes. Assuming memory requirements for the OS are negligible, this gives nearly 32 GB in total for searches and databases. Even though multiple searches might be running, this should be sufficient to allow the more popular databases to remain resident in memory, although not a huge database like NCBI nr.

Each search node requires sufficient free disk space for the Mascot application software and the compressed FASTA databases. The master also requires sufficient disk space for the original FASTA databases and the accumulating search result files. The amount of space required for the results files depends on how heavily the system is used and how often the files are backed-up and deleted.

For best performance, it is advisable for the nodes to have local hard disks. If you prefer to use shared storage, then each node must have its own dedicated directory structure. A single set of sequence database files can be shared between the master and search nodes by setting NodeSequenceDatabaseDir to an absolute path.

Mascot nodes may have any number of processors, but the number of cores in each node should be a multiple of 4 to make maximum use of the number of CPUs in the licence.

A search node does not require a keyboard, monitor or mouse. If you are running Windows on the nodes, and want to be able to "see" the individual desktops, you might consider using a KVM switch so that a single keyboard, monitor and mouse can be shared between all the nodes. Alternatively, Windows Remote Desktop or VNC can be used.

http://www.realvnc.com/

Operating System Requirements

For nodes running Windows, it is not necessary to use a 'Server' version of Windows on the search nodes.

For Linux clusters, it must be possible for the master to communicate with the search nodes using ssh or rsh without quoting a password or passphrase.

Search nodes do not require Perl or web server software.

The master detects that search nodes are responding by issuing an echo command to TCP on port 7 under Linux and ICMP echo under Windows. Hence, these services must not be disabled or blocked by firewalls

Overview of Implementation

Each search is distributed to all the cluster nodes, but each node searches just an allocated portion of the sequence database. Search results are returned to the master, which merges them, writes the result file to disk, and optionally generates HTML reports to be returned to a client web browser.

All master - node communication is via TCP/IP.

Configuration and program files are distributed and updated automatically from the Master node.

Mascot administration tools provide web browser based system status reports. These are continuously updated and show at a glance important parameters such as processor usage and free disk space for each of the nodes. As an option, critical alerts can also be sent to the system administrator by email.

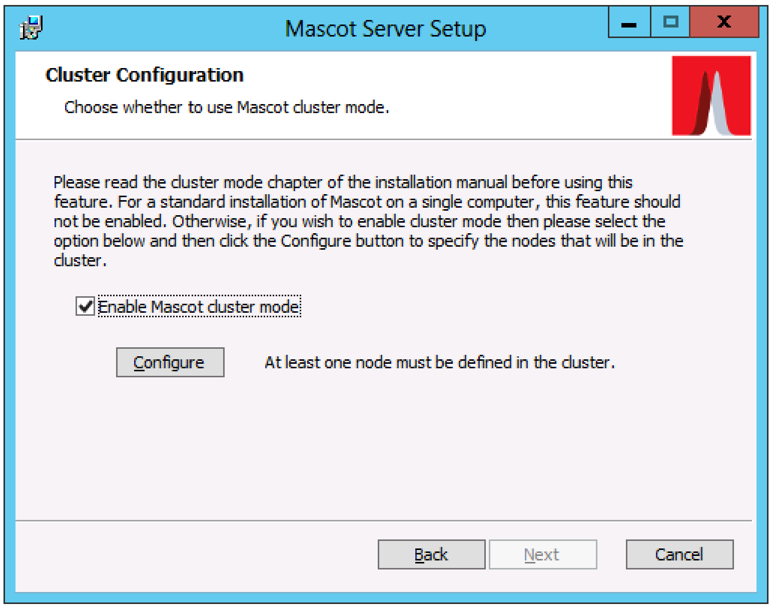
In cluster mode, Mascot is intended to run as a dedicated system. Trying to run other applications on the cluster simultaneously may have unpredictable effects on search speed.

Installation of Mascot

It is only necessary to install or upgrade Mascot on the master system. In fact, no files are copied to the Mascot nodes during installation. The distribution of files and executables is all handled when Mascot Monitor starts.

Windows

During Mascot installation on a Windows system, the following dialog will be displayed:


Mascot Server Setup

Cluster Configuration

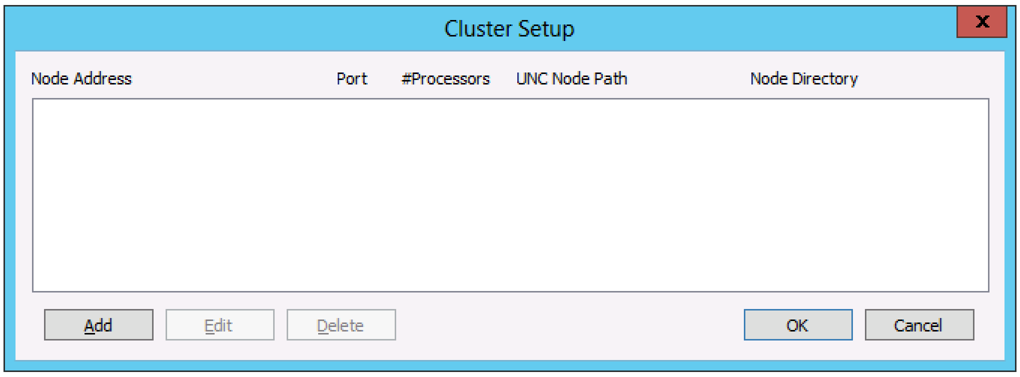
Choose whether to use Mascot cluster mode.

Please read the cluster mode chapter of the installation manual before using this feature. For a standard installation of Mascot on a single computer, this feature should not be enabled. Otherwise, if you wish to enable cluster mode then please select the option below and then click the Configure button to specify the nodes that will be in the cluster.

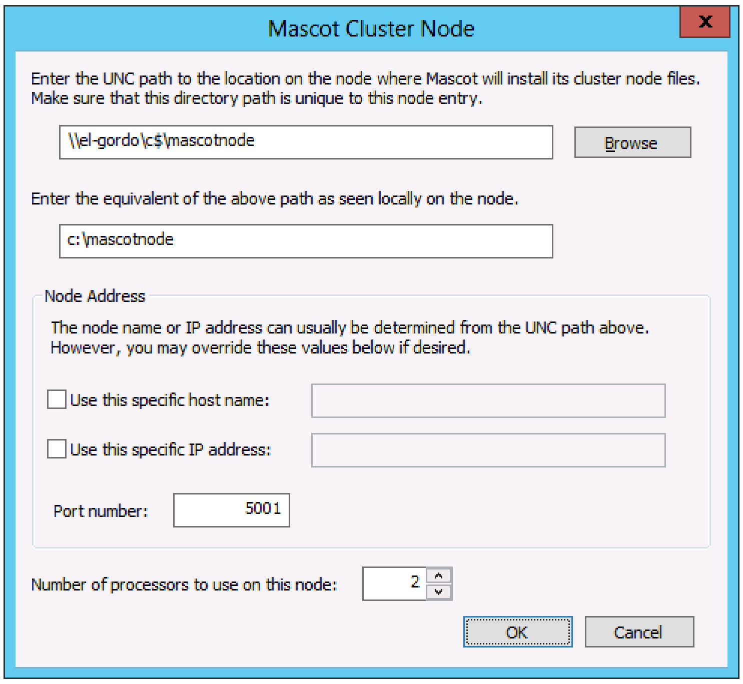
[✓] Enable Mascot cluster mode

Configure — At least one node must be defined in the cluster.

If you enable cluster mode, the configure button invokes the following dialog



Choose Add to configure each cluster node



Enter the UNC path to the location on the node where Mascot will install its cluster node files.
Make sure that this directory path is unique to this node entry.

\\el-gordo\c$\mascotnode

Enter the equivalent of the above path as seen locally on the node.

c:\mascotnode

The node name or IP address can usually be determined from the UNC path above.
However, you may override these values below if desired.

☐ Use this specific host name:

☐ Use this specific IP address:

Port number: 5001

Number of processors to use on this node: 2

Use the Browse button to ensure that the UNC path to the node is correct. If the machines are in a Windows domain, and the remote drive is not explicitly shared, you can enter C$ for drive C, etc., to use the administrative shares. If the base directory does not exist, create it using the 'Make New Folder' button. The recommended base folder name is mascotnode.

Ensure that the local path to the mascotnode directory matches the UNC path. This must be a local or mapped drive on the node so that the path can be specified using a drive letter. The dialog will try to guess the local path from the UNC path, but it may get it wrong. Ensure that this path is correct before pressing OK.

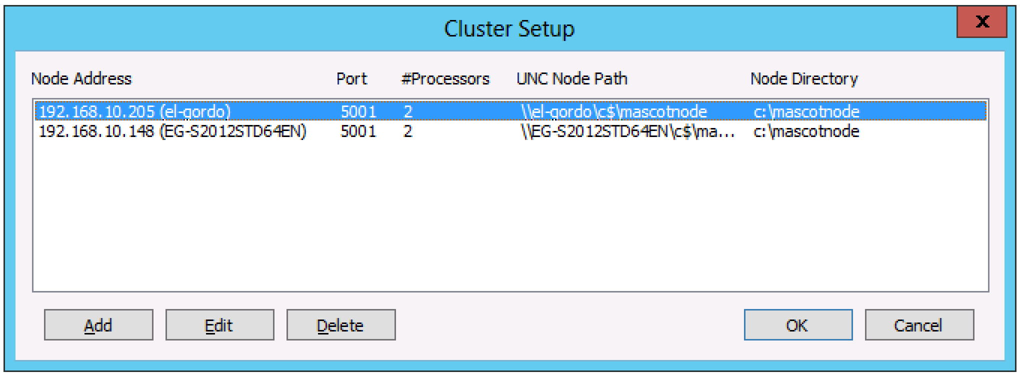
It is not necessary to fill in the Host name and IP Address fields unless the node is a multi-homed system and it is necessary to define which network interface will be used for communication with the Mascot master.

The default port number for cluster communication is 5001. If there are conflicts, this can be changed

The number of processors must be specified. The total number of processors specified for all nodes can be greater than the number of processors in the Mascot licence. The surplus processors will then behave as ‘hot-spares’, to be swapped into the cluster as required if there is a hardware problem on another node.

NOTE: If the master is also a search node, and will execute Mascot searches in addition to running Mascot Monitor and the web server, it must be added as a search node using this dialog.

Use the Add, Edit, and Delete buttons to specify the complete cluster.



Node Address	Port	#Processors	UNC Node Path	Node Directory
192.168.10.205 (el-gordo)	5001	2	\\el-gordo\c$\mascotnode	c:\mascotnode
192.168.10.148 (EG-S2012STD64EN)	5001	2	\\EG-S2012STD64EN\c$\ma...	c:\mascotnode

Press OK to return to the installation wizard, and file installation will begin. Copying the files and configuring the system may take some time.

Once complete, you will be presented with a message requesting that you configure and start the Mascot Monitor service. **Do not do this yet!**

First, you need to configure or disable Windows Firewall on the search nodes. Then, the Monitor service must be started manually, as described below in the section *Starting the Mascot service for the first time.*

Very large clusters

Defining a very large cluster using the Add node... dialog can be tedious. It is usually faster to define a small cluster, let the installation program run to completion, then edit the configuration files using a text editor.

From the Program menu, stop the Mascot service, and edit the cluster and sub-cluster configuration details into `mascot.dat` and `nodelist.txt`. A full description of these files can be found below in the ‘Reference’ section. Then, start the Mascot service.

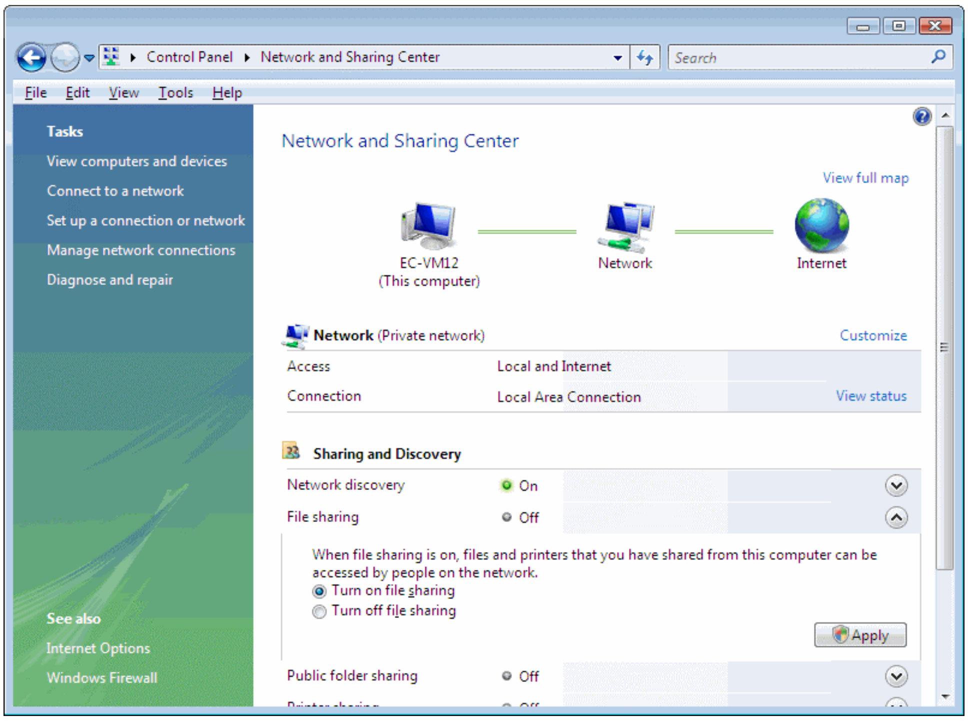
Windows Firewall on Search Nodes

Windows XP and later includes a software firewall called Windows Firewall. You can avoid the configuration steps in this section by turning off Windows Firewall on the search nodes. If the search nodes are on a separate subnet, that can only connect to the master node, having a firewall enabled on a search node is of little use. It is redundant until the master node has been compromised, by which time it is too late. If the search nodes are not on a separate subnet, or if you simply want to enable Windows Firewall because the operating system keeps nagging you to do so, it is necessary to run through the following steps on each search node.

Windows firewall configuration varies across the different editions of Windows and also according to whether it was part of the original installation or added in a service pack.

Vista and Server 2008:

On each search node, log in as a user with local administrator rights. Go to Control Panel, Network Status and ensure the network connection to the master node is described as Private or Domain. If it shows as Public, choose customise to change it. Under *Sharing and Discovery*, Enable *File Sharing*.



Network and Sharing Center

Network (Private network)

Access	Local and Internet
Connection	Local Area Connection

Sharing and Discovery

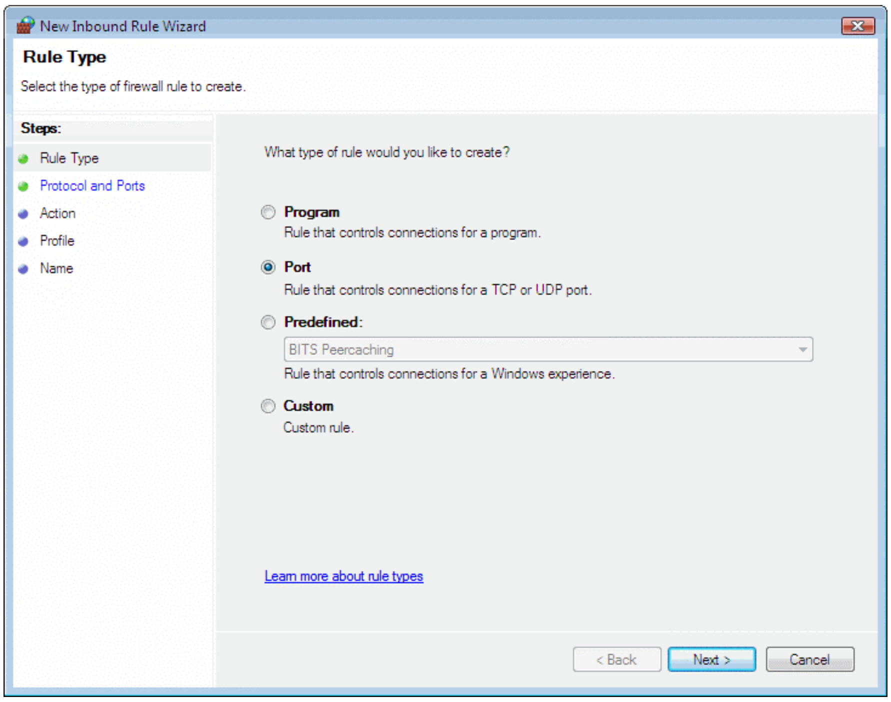
Network discovery	On
File sharing	Off

When file sharing is on, files and printers that you have shared from this computer can be accessed by people on the network.

[✓ Turn on file sharing] Turn off file sharing

Public folder sharing	Off

Select *Administrative Tools* and launch *Windows Firewall with Advanced Security*. Select *Inbound Rules* in the left hand panel and *New Rule* in the action panel.



Rule Type

Select the type of firewall rule to create.

What type of rule would you like to create?

Program
Rule that controls connections for a program.

[✓ Port]
Rule that controls connections for a TCP or UDP port.

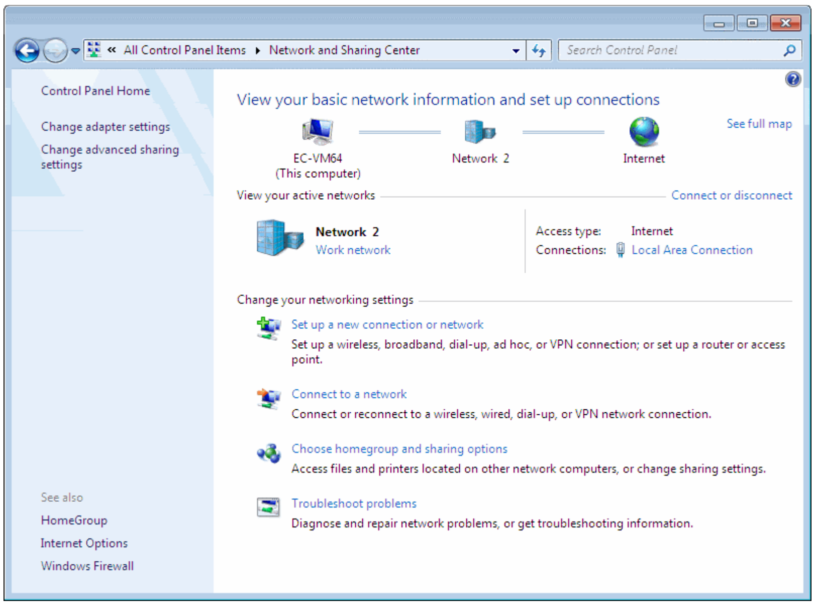
Predefined:
BITS Peercaching
Rule that controls connections for a Windows experience.

Custom
Custom rule.

In the wizard, choose Port, Next, TCP, Specific Local Ports, 5001, Next, Allow Connection, Next, Clear the checkbox for Domain and Public, Next, Enter the name as MascotNodePort5001, Finish. The new rule will be added to the list of Inbound Rules.

Windows 7, Server 2012, and later:

On each search node, log in as a user with local administrator rights. Go to Control Panel, Network & Sharing Center and ensure the network connection to the master node is described as Work or Private or Domain. If it shows as Public, click on the hyperlink to change it. Choose *Change Advanced sharing settings* and ensure *File and Printer Sharing* is enabled.



« All Control Panel Items ▸ Network and Sharing Center

Search Control Panel

Control Panel Home

Change adapter settings

Change advanced sharing settings

See also

HomeGroup

Internet Options

Windows Firewall

View your basic network information and set up connections

See full map

EC-VM64 (This computer) — Network 2 — Internet

View your active networks — Connect or disconnect

Network 2
Work network

Access type: Internet
Connections: Local Area Connection

Change your networking settings

Set up a new connection or network
Set up a wireless, broadband, dial-up, ad hoc, or VPN connection; or set up a router or access point.

Connect to a network
Connect or reconnect to a wireless, wired, dial-up, or VPN network connection.

Choose homegroup and sharing options
Access files and printers located on other network computers, or change sharing settings.

Troubleshoot problems
Diagnose and repair network problems, or get troubleshooting information.

Choose Windows Firewall then Advanced Settings. Select Inbound Rules in the left hand panel and New Rule in the action panel.

In the wizard, choose Port, Next, TCP, Specific Local Ports, 5001, Next, Allow Connection, Next, Clear the checkbox for Public, Next, Enter the name as MascotNodePort5001, Finish. The new rule will be added to the list of Inbound Rules.

Nodes belonging to a Workgroup

The steps in this section are <u>not</u> required if all the nodes belong to a Windows domain.

A registry change is required to allow administrator rights when logging in using a local (SAM) account. This procedure is taken from Microsoft KB article 951016

1. Click Start, click Run, type regedit, and then press ENTER. If the start menu does not have a Run… option, then open a Command Prompt window from the Accessories program folder and use this instead.

2. Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System

3. If the LocalAccountTokenFilterPolicy registry entry does not exist, follow these steps:

   A. On the Edit menu, point to New, and then click DWORD Value.
   B. Type LocalAccountTokenFilterPolicy, and then press ENTER.
4. Right-click LocalAccountTokenFilterPolicy, and then click Modify.
5. In the Value data box, type 1, and then click OK.
6. Exit Registry Editor.

Repeat this entire procedure on every search node.

Remote Registry

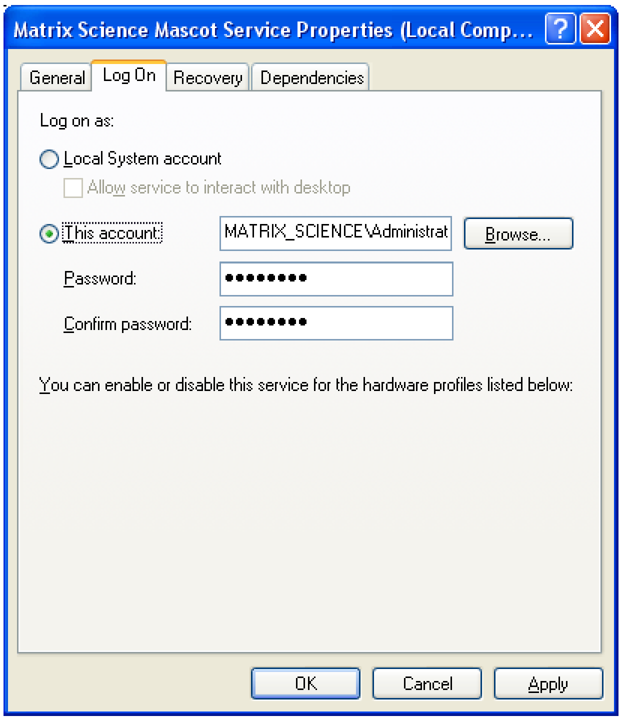
On each search node, from the Control panel, Administrative Tools, open the Services dialog and select *Remote Registry*. Unless already set to Automatic, right click and choose Properties. On the General tab, set *Startup type* to Automatic and also start the service. Choose OK.

Starting the Mascot service for the first time

The Monitor service on the master needs to be run under an account that has local Administrator rights on each node because it needs to write to the registry, install, start and stop services on each node. (If you later change the password for this account, remember to change it in the Logon tab of the Matrix Science Mascot Service properties.)

On the master node, from the Control panel, Administrative Tools, open the Services dialog and select Matrix Science Mascot Service. Right click and choose Properties. Go to the *Log On* tab and choose *This account*. Enter the user name and password for a domain account with local Administrator rights on each search node. (*not* the local administrator account on the master). You could use a domain administrator account, but this might be considered risky.

If the nodes do not belong to a domain, all nodes (including the master) must have a user defined with administrator rights and the *same user name and password*. The service must be set to log in as this user.


Matrix Science Mascot Service Properties (Local Comp...

General | Log On | Recovery | Dependencies

Log on as:

Local System account

Allow service to interact with desktop

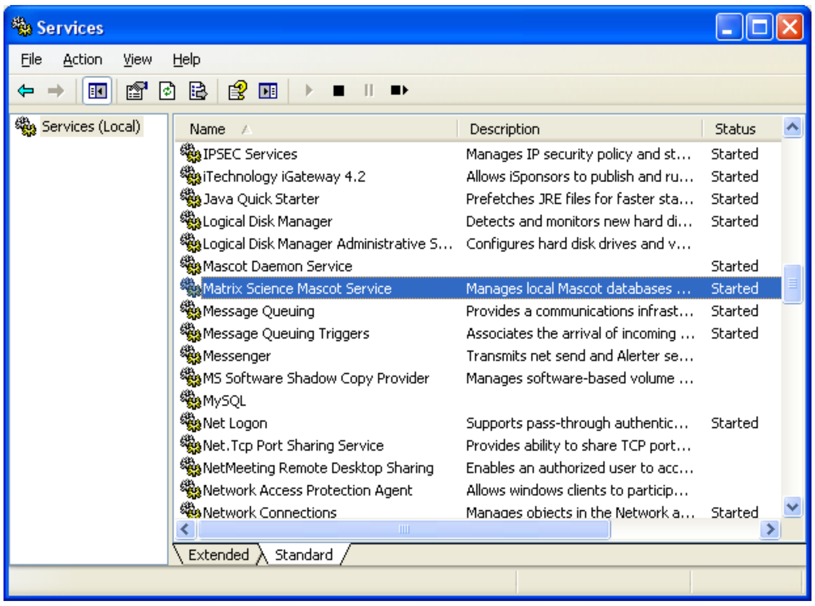
[✓ This account:] MATRIX_SCIENCE\Administrat

Password: ••••••••

Confirm password: ••••••••

You can enable or disable this service for the hardware profiles listed below:

Press OK, and you will be returned to the Services dialog:



Name	Description	Status
IPSEC Services	Manages IP security policy and st...	Started
iTechnology iGateway 4.2	Allows iSponsors to publish and ru...	Started
Java Quick Starter	Prefetches JRE files for faster sta...	Started
Logical Disk Manager	Detects and monitors new hard di...	Started
Logical Disk Manager Administrative S...	Configures hard disk drives and v...	
Mascot Daemon Service		Started
Matrix Science Mascot Service	Manages local Mascot databases ...	Started
Message Queuing	Provides a communications infrast...	Started
Message Queuing Triggers	Associates the arrival of incoming ...	Started
Messenger	Transmits net send and Alerter se...	
MS Software Shadow Copy Provider	Manages software-based volume ...	
MySQL		
Net Logon	Supports pass-through authentic...	Started
Net.Tcp Port Sharing Service	Provides ability to share TCP port...	
NetMeeting Remote Desktop Sharing	Enables an authorized user to acc...	
Network Access Protection Agent	Allows windows clients to particip...	
Network Connections	Manages objects in the Network a...	Started

Highlight the entry for Matrix Science Monitor Service and press Start. If the service fails to start, the cause must be investigated and the problem fixed before proceeding.

Monitor progress using the Database Status page on the master. Choose *Monitor log* and watch for error messages as the program files and database files are copied to the search nodes.

Completion

The installation is now complete. There will be a lot of disk activity while the Mascot service compresses the SwissProt sequence database. Searches on the database cannot be performed until the files have been compressed. You should open up the status screen in a web browser (Start menu; Programs; Mascot; Search Status) and verify the cluster status.

If this is a clean installation or a version update, you will need to follow the links to register a product key as described in Chapter 2 (Linux installation) or Chapter 3 (Windows installation). Once the licence file has been saved to *`config/licdb`* on the master node, you will be able to proceed to Database Status.

If all is well, you will see rows of happy faces and the status line will display the following messages:

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

Once the database is "In use", you can begin exploring and using Mascot. Clicking on the links in the cluster node table will display more detailed status information for individual nodes.

Linux

Communication

Under Linux, the master node communicates with the search nodes using either ssh (preferred), or rsh. If communication can be established using ssh, then scp is used for file copying. If rsh is used for communication, then rcp is used for file copying.

Whether ssh or rsh is used, it is essential that communication can be established without requiring passwords or passphrases. In the case of ssh, key based authentication is the preferred mechanism. A less secure alternative for rsh is provided by file based authentication using .rhosts or hosts.equiv.

A detailed description of the many ways to configure ssh or rsh is outside the scope of this manual. For key based authentication, read the man pages for: ssh, sshd, ssh-keygen, ssh-add, ssh-agent. For file based authentication, read the man pages for: rsh, rshd, rlogin, hosts.equiv.

The minimum procedure to set up key based authentication for ssh on a clean Linux system, where there are no pre-existing keys, is as follows:

1. Login to the master node as the user who will own the ms-monitor.exe process, (generally root), and generate a version 2 RSA key pair by executing:

   ssh-keygen –t rsa

2. When asked for a passphrase, press return to indicate no passphrase is required. Accept the default location for saving the key files, ($HOME/.ssh)

3. The contents of the public key, $HOME/.ssh/id_rsa.pub, must be added to a file called $HOME/.ssh/authorized_keys on each of the search nodes.
4. Test communication by logging in to each search node from the master node using ssh. The first time a connection is made, confirm that the new host should be added to the list of known hosts, $HOME/.ssh/known_hosts

Installation

Perform a standard installation of Mascot onto the master system according to the procedure in Chapter 2. Verify correct system operation as a single server by performing searches of SwissProt and familiarise yourself with administrative tools such as ms-review.exe and ms-status.exe (Chapter 7). Any problems need to be resolved before reconfiguring for cluster operation.

Cluster Configuration Procedure

1. Kill ms-monitor.exe

2. Open mascot/config/mascot.dat in a text editor. Move down to the "Cluster" section and enter configuration information for the cluster. The parameters are fully described below in the Reference section. In the databases section, verify that the threads and blocks parameters are set to 1 for all databases. If this is not the case, make the necessary changes, then save mascot.dat. For a 5 node, 10 cpu cluster, typical entries might be:

```
Cluster
#
# Enable (1) or disable (0) cluster mode
Enabled 1
#
# MasterComputerName must be the hostname
MasterComputerName zx80
#
# Node defaults
DefaultNodeOS Linux
DefaultNodeHomeDir /usr/local/mascotnode
#
# Following line must be commented out WHEN this is a homogeneous
MascotNodeScript        /usr/local/mascot/bin/load_node.pl
#
# Sub-cluster definition
# Syntax is SubClusterSet X Y where X is the sub-cluster number
# and Y is the maximum number of CPUs to use within the given sub-
#
SubClusterSet 0 -1
#
# Time outs, log files
IPCTimeout 5      # seconds with no response before timeout
IPCLogging 0      # no logging = 0, minimal = 1, verbose = 2
IPCLogfile ../logs/ipc.log     # relative path
CheckNodesAliveFreq     30     # seconds between node health checks
SecsToWaitForNodeAtStartup     20     # seconds to wait for node to
NodeSequenceDatabaseDir ../sequence # location of compressed databases
#
end
```

3. Open *mascot/config/not.nodelist.txt* in a text editor. Enter configuration information for the cluster. The parameters are fully described below in the Reference section. Save as *nodelist.txt*. For a 5 node, 10 cpu cluster, typical entries might be:

```
# Cluster node definitions
#
# Each line begins with the word Node, followed by a space and
# then a comma delimited list of configuration parameters:
#      ip address:port
#      computer (host) name
#      maximum number of node CPU's to be used
#      operating system
#      local path to home directory
#      home directory as seen from master (specify for NT master only)
#
Node 10.0.0.1:5001, search01, 2, Linux, /usr/local/mascotnode
Node 10.0.0.2:5001, search02, 2, Linux, /usr/local/mascotnode
Node 10.0.0.3:5001, search03, 2, Linux, /usr/local/mascotnode
Node 10.0.0.4:5001, search04, 2, Linux, /usr/local/mascotnode
Node 10.0.0.5:5001, search05, 2, Linux, /usr/local/mascotnode
```

4. Re-start *ms-monitor.exe*. Note that you must change directory to *mascot/bin* and have super user privileges to execute *ms-monitor.exe*.

Note: (Linux only) Under Redhat Linux 8.0, if *ms-monitor.exe* terminates immediately after launch, without any error messages, the problem may relate to a bug in gethostbyname_r(). In the cluster section of *mascot.dat*, try using the IP address for the master node, rather than the hostname, as the argument to MasterComputerName.

5. In a web browser, navigate to *ms-status.exe* and verify that the system starts up correctly.

Reference

The Cluster section in mascot.dat

```
Cluster
#
# Enable (1) or disable (0) cluster mode
Enabled 1
#
# MasterComputerName must be the hostname
MasterComputerName mascot-master
#
# Node defaults
DefaultNodeOS Windows_NT
DefaultNodeHomeDir c:/mascotnode
#
# Following line must be commented out UNLESS this is a
   DefaultNodeHomeDirFromMaster \\<host_name>\c$\mascotnode
#
# Following line must be commented out WHEN this is a
# MascotNodeScript        ###ROOT###/bin/load_node.pl
#
# Sub-cluster definition
# Syntax is SubClusterSet X Y where X is the sub-cluster number
# and Y is the maximum number of processors in the sub-cluster
SubClusterSet 0 -1
```

```
#
# Time outs, log files
IPCTimeout              5                    # seconds with no
IPCLogging              0                    # no logging = 0,
IPCLogfile              ../logs/ipc.log      # relative path
CheckNodesAliveFreq     30                   # seconds between node
SecsToWaitForNodeAtStartup  20               # seconds to wait for
NodeSequenceDatabaseDir ../sequence # location of compressed databases
#
end
```

Enabled

1 to enable cluster mode, 0 to enable single server mode

MasterComputerName

Enter the host name for the master computer and, optionally, the IP address separated by a comma. The IP address may need to be specified for a multi-homed master where it is necessary to define which network card is on the LAN and which is the gateway to the outside world.

DefaultNodeOS

If no OS is defined for a particular node, then this OS is assumed. Must be one of:

Windows_NT

Linux

Note that these names are case sensitive.

DefaultNodeHomeDir

If no specific home directory is specified for a particular node, then this default is used.

On a Linux system, this will typically be *`/usr/local/mascotnode`*. It is best not to use *`/usr/local/mascot`* as this is the directory mostly used for the master.

On a Windows system, this will typically be *`C:/MascotNode`* or *`D:/MascotNode`*.

To override this setting for a particular node, enter the directory on the node line

DefaultNodeHomeDirFromMaster

This is the directory on the node as seen from the master. For a Windows cluster, this must be present and specified as a UNC name.

The text <host_name> will be replaced by the host name as specified in the Node line.

For a Linux cluster, this parameter must be commented out.

MascotNodeScript

This script is run for each node with the following parameters:

-i ip address of node - required

-t The task to be performed - required, either

'StopNode' – the script will try and stop the Mascot Node daemon or service on the specified node.

or

'StartNode' – the script will unconditionally update *ms-mascotnode.exe* and *mascot.dat* on the specified node, then start the Mascot Node daemon or service.

-f Full path to the node's home directory - required

-r Port number of node - required

-o The operating system running on the node – required

For a Linux cluster, the master and search nodes must be able to communicate using either ssh (preferred), or rsh without requiring passwords or passphrases. In the case of ssh, key based authentication is the preferred mechanism. A less secure alternative for rsh is provided by file based authentication using .rhosts or hosts.equiv.

As shipped, load_node.pl executes *ms-mascotnode.exe* as root on each search node. If this is not acceptable, the script can be edited.

SubClusterSet X Y

Large clusters can be divided into sub-clusters. X is a unique integer value (0 based) used to identify the sub-cluster. A single cluster must have a single entry with X set to 0. Y is the maximum number of processors in the sub-cluster. The default for Y is -1, which corresponds to the number of processors in the licence.

IPCTimeout

The timeout in seconds for inter-process communication

IPCLogging

0 for no logging of inter-process communication
1 for minimal logging
2 for verbose logging

IPCLogfile

The relative path to the inter-process communication log file

CheckNodesAliveFreq

The interval in seconds between 'health checks' on the nodes

SecsToWaitForNodeAtStartup

At startup, if a node is not available within this time, the system will continue to startup without that node. If the value is set to 0, then the system will wait indefinitely. Default is 60 (seconds).

This timeout is also used if a node fails while the system is running. The system will wait for this number of seconds before re-initialising ms-monitor.exe. This means that a short-lived interruption in network communication doesn't create a major service interruption.

MascotNodeRebootScript

Path to an optional CGI script to re-boot a cluster node. If this parameter is defined, there will be a link at the bottom of each Mascot Cluster Node status page. Clicking on this link will execute the specified CGI script with the host name of the specified node as an argument.

DefaultPort

Sets the default port number to be used when this parameter is missing from *nodelist.txt*. Recommended default is 5001

UseCompleteDatabase

Not used. If specified, must be set to 1.

NodeSequenceDatabaseDir

The parent directory for the compressed sequence database files. Can be an absolute path or relative to the bin directory. If the cluster uses shared storage, making this an absolute path allows the master and search nodes to share a common set of sequence database files. Default is ../sequence

nodelist.txt

This file is used to define the nodes that belong to the cluster. For a very large cluster, it is advisable to define a few percent of additional nodes as 'spares'. For example, if 51 nodes with 102 processors were available, and Mascot was configured to use 2 sub-clusters, each of 50 processors, the node with the 2 spare processors could be used to replace a failed node automatically.

```
# Cluster node definitions
#
# Each line begins with the word Node, followed by a space and
# then a comma delimited list of configuration parameters:
#      ip address:port
#      computer (host) name
#      maximum number of node CPU's to be used
#      operating system
#      local path to home directory
#      home directory as seen from master (specify for NT master only)
#
Node 10.0.0.1:5001, search01, 2, Windows_NT, c:/MascotNode,
  \\search01\c$\MascotNode
```

```
Node 10.0.0.2:5001, search02, 2, Windows_NT, c:/MascotNode,
   \\search02\c$\MascotNode
Node 10.0.0.3:5001, search03, 2, Windows_NT, c:/MascotNode,
   \\search03\c$\MascotNode
Node 10.0.0.4:5001, search04, 2, Windows_NT, c:/MascotNode,
   \\search04\c$\MascotNode
Node 10.0.0.5:5001, search05, 2, Windows_NT, c:/MascotNode,
   \\search05\c$\MascotNode
```

Important: Because Mascot frequently writes status information to `nodelist.txt`, you should open the file in a text editor that puts a lock on the file (e.g. vi or wordpad). This will prevent Mascot from modifying the file while it is being edited. `nodelist.txt` can be viewed using Mascot Status.

Node

There must be one or more node entries. Items in square brackets are optional – but the commas must always be supplied.

IP address:Port, Host name, Number of processors, [OS], [Home dir], [Home dir from master]

IP address, port, and host name must always be specified.

The number of processors to be used on the node can be less than the number of processors available. If the total number of processors specified in all the nodes exceeds the number of licenses available, only the licensed number will be used at any one time.

If the OS is not specified, then the DefaultNodeOS is used. Must be one of the choices shown under DefaultNodeOS.

The home directory is the local path on the node to the root of the Mascot directory structure. If this is not specified, then DefaultNodeHomeDir is used.

Home directory from master is the home directory on the node as seen from the master. This parameter is only applicable to a Windows cluster and must be omitted for a Linux cluster.

Once a cluster has been started, an additional four status values will be written periodically to `nodelist.txt`. If you edit this file while Mascot is *not* running, these values can be deleted.

subcluster ID number (0 based)

node within subcluster (0 based)

status: 0 unknown status

1 attempting to bring into use

2 no response to ping

3 failed to start service

4 in use

number of CPU's actually being used

File Replication

The configuration files, such as `mascot.dat`, that are on the Mascot master are automatically replicated to the nodes. So, it is only necessary to update a file on the master. The `ms-monitor.exe` program (run as the Matrix Science Mascot Service under Windows), continually looks to see if a file has been updated, and will distribute new versions to the nodes as required. The dates, times and lengths of the distributed files should be identical on all systems.

The same process is used for updates to executable programs, except that these updates will only be made when the `ms-monitor.exe` service first starts.

The Status screen will indicate if any executable files need updating.

Files required on each Mascot Node

Target File name and directory relative to node home directory	**Notes**
./bin/ms-mascotnode.exe	Updated at start-up
./bin/nph-mascot.exe	
./config/enzymes	
./config/mascot.dat	Updated at start-up
./config/unimod.xml	
./config/taxonomy	
./config/fragmentation_rules	
./config/quantitation.xml	
./taxonomy/nodes.dmp	
./taxonomy/usernodes.dmp	Not required by most users. Note that names.dmp is not required on the Mascot Nodes.

Start-up of ms-monitor.exe

The following sequence occurs (for each node) when `ms-monitor.exe` starts for the first time on the master system. (items marked * are for Windows clusters only).

1. See if the computer is available by opening a socket to the ping port (port 7)
2. If there is an entry ‘StopMascotNodeCmd’ in the mascot.dat file, then run that command to stop the Mascot node daemon

   or

See if there is a MascotNodeService installed on the computer - if there is, then stop that service

3. If there is no ms-mascotnode.exe or if it is out of date on the Mascot node, then copy/update the file from the cluster/<OS> directory on the Mascot Master system to the specified directory on the Mascot Node.
4. If the service is not installed, then install the service, and add a registry entry for the directory to be changed to at start up
5. Make a logs and config directory and copy mascot.dat
6. Start the MascotNodeService on the Mascot Node computer. (With a Linux based system, the ms-mascotnode.exe daemon will be started).
7. Check that the service / daemon now communicates through TCP/IP sockets – if it fails, then a message indicating which Mascot node it is waiting for is displayed in the ms-status screen.
8. Initialise the MascotNodeService / daemon by sending the appropriate commands
9. See if any files are missing or out of date (see above), and if necessary, update them. This is done though the TCP/IP socket, so no directory mapping / NFS mounts are required.

Once all the Mascot nodes have been successfully initialised, then Mascot Monitor starts as normal.

Licensing

The number of processors that the search is permitted to run on is restricted by the number of mascot licenses. The Mascot master node is not included in this list, since it merely distributes the search and collates the results. The number of processors to be used for Mascot will never exceed the number specified in the licence.

Error messages and emails

In the single server version of Mascot, selected warning messages can optionally be emailed to the system administrator when something critical, such as a database update, fails on the server. The following additional messages, specific to a cluster, can also be emailed:

M00323 One or more cluster nodes has stopped responding

M00316 Dr. Watson log updated (indicating a software crash) on one of the cluster nodes.

Who Am I?

If the Mascot master is also being used as a node, when nph-mascot.exe is run, it needs to know whether it is running as a node task or as master task. Since the different `mascot.dat` files are identical, it determines this from a file *`mascot/config/iam.dat`* that is created by the Mascot node service when it starts up. Do not copy or replace this file.

Windows Manual Configuration

The following configuration steps on each search node are performed automatically as part of the Windows installation

MascotNodeService

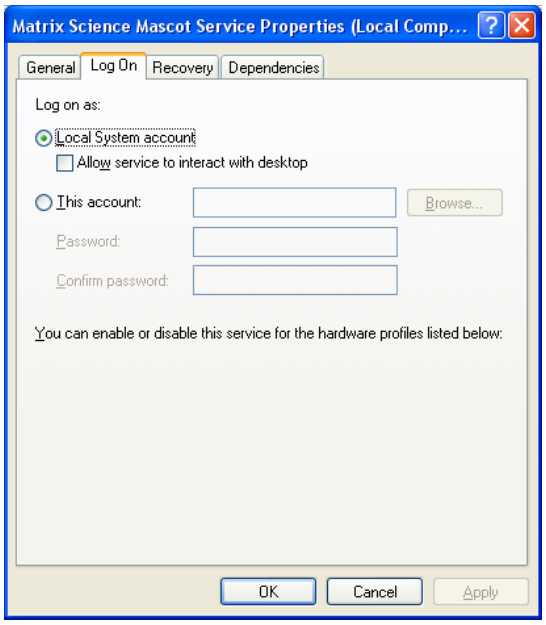
Under Windows, *ms-mascotnode.exe* is configured to run as a service. This should be taken care of automatically. If there are any problems, service creation or deletion requires the Microsoft utility *sc.exe*, which can be found in the *mascot/cluster/Windows_NT* directory.

The command to create the service is:

```
sc create MascotNodeService type= own  binpath=
   c:\mascotnode\bin\ms-mascotnode.exe  start= auto
```

You may need to change the path to the executable, and note that the spaces after the equals signs are significant.

To verify that the service has been created successfully, from the Control panel, open the Services control panel and choose MascotNodeService. Select Startup… and the following dialog should be displayed:


Matrix Science Mascot Service Properties (Local Comp…

General | Log On | Recovery | Dependencies

Log on as:

(●) Local System account

[ ] Allow service to interact with desktop

( ) This account:

Password:

Confirm password:

You can enable or disable this service for the hardware profiles listed below:

To delete the service, first stop it, close the services control panel, then enter:

```
sc delete MascotNodeService
```

Dr. Watson

To prevent (invisible) dialog boxes from being displayed if a fatal error occurs, edit the registry key

```
HKEY_LOCAL_MACHINE\Software\Microsoft\DrWatson
```

Set the value of `VisualNotification` to 0. When the Mascot node service starts on a Windows system, it sets a Dr. Watson registry entry to ensure that Dr. Watson log files are written to the node *logs* directory.

Registry Settings

Two registry entries are used on each search node to record the root directory of the mascot file structure and the port number used for communication. For example:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\MatrixScience\Mascot\1.00]
"MascotNodeFolder"="C:/mascotnode/bin"
"MascotNodePort"="5001"
```

Very large Mascot clusters

Very large clusters (> 30 nodes) pose certain special problems:

- Even with reliable hardware, node failures can be expected relatively frequently
- LAN communication can become a bottleneck
- Need to avoid mixing processors with different speeds, because the slower processors become a bottleneck

Mascot allows large clusters to be divided into sub-clusters. Each sub-cluster uses identical databases and configuration files, but operates independently of the other sub-clusters. An incoming search can be directed to a specific sub-cluster or the first available sub-cluster.

Should a node go down, only the sub-cluster is affected. Ideally, there will be one or more "spare" nodes defined. Mascot will reconfigure the sub-cluster using a spare node and re-start. If there are no spare nodes, Mascot will reconfigure the sub-cluster to exclude the faulty node and re-start.

Configuration

mascot.dat

SubClusterSet X Y

Large clusters can be divided into sub-clusters. X is a unique integer value (0 based) used to identify the sub-cluster. A single cluster must have a single entry with X set to 0. Y is the maximum number of processors in the sub-cluster. The default for Y is -1, which corresponds to the number of processors in the licence.

nodelist.txt

This file is used to define the nodes that belong to the cluster. For a very large cluster, it is advisable to define a few percent of additional nodes as "spares". For example, if 51 nodes with 102 processors were available, and Mascot was configured to use 2 sub-clusters, each of 50 processors, the node with the 2 spare processors could be used to replace a failed node automatically. At start-up, ms-monitor starts each sub-cluster in turn, taking the required number of nodes from nodelist.txt in the order specified in the file. If you wish to override this behaviour, specify a sub-cluster number in nodelist.txt:

```
# Each line begins with the word Node, followed by a space and
# then a comma delimited list of configuration parameters:
#      ip address:port
#      computer (host) name
#      maximum number of node CPU's to be used
#      operating system
#      local path to home directory
#      status: 0 = available
#      sub-cluster
```

Mascot system limits

The following system limits listed in Appendix C of manual are relevant to very large clusters

- Maximum number of processors per machine
- Maximum number of sub-clusters in a cluster
- Maximum number of machines in a sub-cluster
- Maximum number of processors in a sub-cluster
- Maximum number of nodes in nodelist.txt

Directing jobs to a sub-cluster

The SUBCLUSTER search parameter is used to direct jobs to a sub-cluster. This can be added to the web-browser search form as a hidden field by editing the Perl script.

To use the next free sub-cluster:

```
SUBCLUSTER=-1
```

If all of the sub-clusters have searches running, and the search has been submitted from a browser, then the following will be displayed in the browser until a sub-cluster becomes free:

Waiting for sub cluster to become available......

To use a specific sub-cluster, e.g sub-cluster 2

```
SUBCLUSTER=2
```

The default value is 0 so, if this parameter is not specified, a search will go to the first sub-cluster.

Specifying which sub-cluster a particular job goes to usually implies some third party job queuing system is being used. For example

- Job gets submitted to Portable Batch System (PBS) and PBS decides which sub-cluster to run the search on
- PBS adds a SUBCLUSTER=x to the search parameters

- PBS creates a task_id using ms-searchcontrol.exe --create_task_id
- PBS submits the search, passing the returned task_id
- You can monitor / control the running search using ms-searchcontrol.exe

A simpler, “static” system could be implemented by adding a SUBCLUSTER command to a Daemon parameter set. SwissProt-SC1.par might contain SUBCLUSTER=1, so selecting this for a task would direct searches to sub-cluster 1, etc.

Database Status

If multiple sub-clusters are defined, the database status screen (ms-status.exe) only shows one sub cluster at a time. An additional summary table is shown at the bottom of the page, with links for the other sub-clusters.

12

12. Mascot Security

Overview

The security model allows a Mascot administrator to:

- Prevent un-authorised changes of Mascot server configuration files using, for example, the database maintenance utility
- Restrict access to results files and sequence databases based on group and user definitions
- Provide standard 'session' support (with time-outs) so users do not need to continually re-enter passwords
- Restrict access to Mascot server based utilities that allow deletion of searches and other job control functions
- Provide read-only access to configuration files for third party applications without requiring login
- Optionally allow submission of searches etc. for 3rd party applications without a login
- Easily set-up Mascot Daemon to run searches as the 'customer' in a service or core lab environment

Some third party applications require helper scripts to be installed on the Mascot web server. If Mascot security is enabled, you should be aware that such scripts may create security holes.

Enabling security

When Mascot is first installed, the security system is disabled. To enable security, open a command prompt or shell on the Mascot server and change to the mascot/bin directory. Enter the command:

```
perl enable_security.pl
```

The Mascot service (ms-monitor.exe) must then be stopped and restarted.

Disabling security

To disable security, open a command prompt or shell on the Mascot server and change to the mascot/bin directory. Enter the command:

```
perl disable_security.pl
```

The Mascot service (ms-monitor.exe) must then be stopped and restarted.

Authentication

There are two different ways in which users can be authenticated:

1. Mascot authentication. The passwords are stored and maintained by Mascot security.
2. Web server authentication. Available with any web server that supports authentication. Refer to your web server documentation for details on how to set up authentication

The type of authentication is set up at the user level, and not as a global setting. Even if the server has web authentication switched on, it may be useful to set some users to be authenticated using the Mascot authentication. A typical case for this might be for a service lab manager running Windows and IIS with integrated authentication. This user would not typically want to create a separate Windows login account for the administrator, but would choose to login explicitly as administrator to update configuration files etc. For an Apache server, with authentication switched on, most users would want to be set to use the authenticated login

Users

New users are added using the Mascot security administration utility. There are 6 special "system" user accounts:

guest

The guest user is not enabled by default. If this account is enabled, then any user is automatically logged in as guest, and needs to explicitly login as another user to gain further access rights. The guest account cannot be deleted, but the account can be disabled. The userid is 1.

admin

This account should be used to perform administration on the Mascot server. It is recommended that you always log in as administrator to perform security and other administration rather than assign administrator rights to another user. The administrator account cannot be deleted or disabled and the admin user cannot be removed from the administrators group. By default, the administrator can access all the administrator screens, but cannot submit searches. The userid is 2. The initial password for admin is admin, but this must be changed on first login.

command line

This pseudo user is always used when running programs from the command line, and can perform any task without restriction. This 'user' doesn't appear in the security administration utility and hence the account cannot be deleted or disabled. The userid is 3.

daemon

This user should be used to run searches in Mascot Daemon. See the Mascot Daemon help for details. The user account is disabled by default, so it will need to be enabled and before use. The userid is 4.

public_searches

This is a pseudo user that is used for the example searches. This 'user' doesn't appear in the security administration utility and hence the account cannot be deleted or disabled. It isn't possible to login as this user. The userid is 5.

Types of user

Five 'types' of user are available, and the appropriate type should be selected using a the drop down list in the administration screen:

Standard Mascot User

The user name and password are stored by Mascot

IP address

This 'user' should only be used for third party legacy applications that do not support Mascot security. Instead of a user name, enter the static IP address of the computer that will access the Mascot server. Do not enter a password.

Computer name

Same as the IP address, but the computer name is used instead. A computer name is more practical where dynamic IP addresses are being used.

Agent string

Should only be used as a last resort for third party applications that haven't implemented Mascot security and where the computer name / IP address is not reliable. A case sensitive substring comparison will be made with the HTTP_USER_AGENT environment variable.

Use built in web server authentication

See description of 'authentication' above.

Mascot will never prompt these users for username and password, and hence passwords and password expiry will be ignored.

Mascot security session time-outs do not apply.

In Microsoft Internet Information Services, (IIS), if anonymous access and integrated authentication are both enabled, then users will generally be 'logged in' as anonymous until they try to access a file where permission is denied. This almost certainly means that anonymous login must be disabled to use this option.

IIS user names generally include the Domain name: e.g. matrix_science/charles. The comparison will be with everything after the last forward or back slash. So, in this case, you would enter 'charles' as the user name.

Groups

Access rights can are assigned to groups, not users. Therefore, a user has no effective rights unless they belong to one or more groups. If a user belongs to more than one group, then their rights are the combination of the rights in both groups.

There are 7 special built in groups:

Guests

By default, the guest user is the only member of this group and the guest group can only submit PMF searches against any database. This can easily be changed using the security administration utility.

Administrators

The admin user always belongs to this group. Members of the group can perform any administration task, but cannot submit searches.

PowerUsers

Members of this group can submit all types of searches and perform some administration. They cannot access the security administration utility.

Daemons

The daemon user belongs to this group by default.

MIUsers

Mascot Insight users with limited rights.

MIPowerUsers

Mascot Insight users with additional rights.

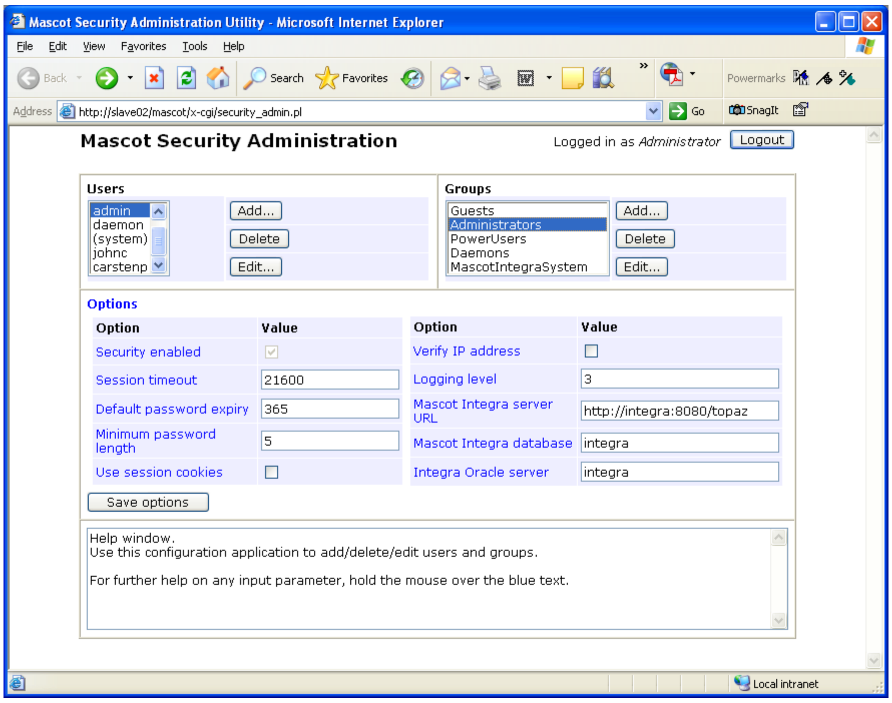
MIAdministrators

Mascot Insight users with administrator rights.

Using the security administration utility

When the security administration utility is started for the first time, you will need to login as admin/admin. You are then forced to change the password.

The main page lists the current users and groups and has buttons for deleting adding and editing users and groups. The global security options can also be modified from this page. On all the pages, there is a help window that gives details about specific options – just position the mouse over the relevant hyperlink to see the help.


Mascot Security Administration Utility - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Powermarks
Address http://slave02/mascot/x-cgi/security_admin.pl
Go SnagIt
Local intranet

Mascot Security Administration

Logged in as *Administrator* Logout

Users		Groups	
admin daemon (system) johnc carstenp	Add... Delete Edit...	Guests Administrators PowerUsers Daemons MascotIntegraSystem	Add... Delete Edit...

Options

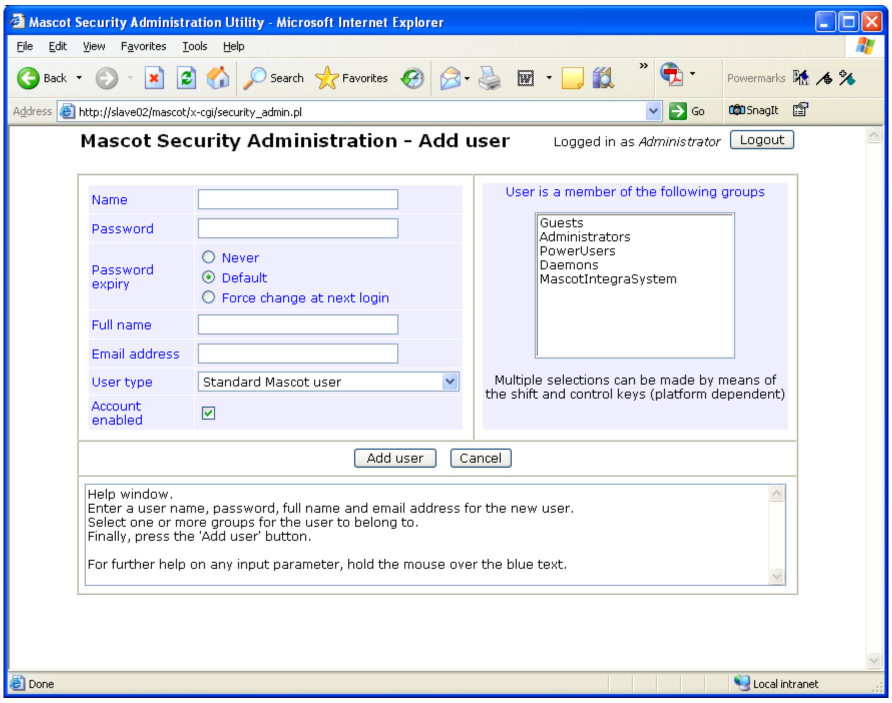
Option	Value	Option	Value
Security enabled	✓	Verify IP address	☐
Session timeout	21600	Logging level	3
Default password expiry	365	Mascot Integra server URL	http://integra:8080/topaz
Minimum password length	5	Mascot Integra database	integra
Use session cookies	☐	Integra Oracle server	integra

Save options

Help window.
Use this configuration application to add/delete/edit users and groups.

For further help on any input parameter, hold the mouse over the blue text.

To add a new user, click on the Add… button:


Mascot Security Administration Utility - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Powermarks
Address http://slave02/mascot/x-cgi/security_admin.pl
Go SnagIt
Done
Local intranet

Mascot Security Administration - Add user

Logged in as *Administrator* Logout

Name	
Password	
Password expiry	○ Never ● Default ○ Force change at next login
Full name	
Email address	
User type	Standard Mascot user
Account enabled	✓

User is a member of the following groups

Guests
Administrators
PowerUsers
Daemons
MascotIntegraSystem

Multiple selections can be made by means of the shift and control keys (platform dependent)

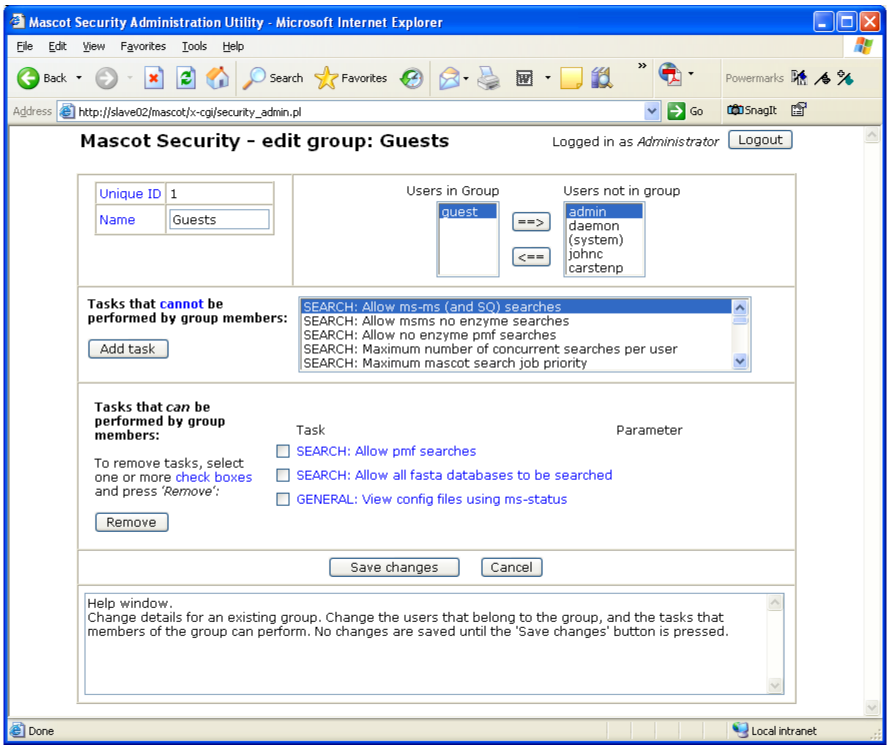
Add user Cancel

Help window.
Enter a user name, password, full name and email address for the new user.
Select one or more groups for the user to belong to.
Finally, press the 'Add user' button.

For further help on any input parameter, hold the mouse over the blue text.

The new user must be given a name, password, full name and email address. There is a description of the different user types of user earlier in this chapter. You should also select one or more groups that the user should belong to before pressing the ‘Add user’ button.

New groups may be added or edited:


Mascot Security Administration Utility - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Powermarks
Address http://slave02/mascot/x-cgi/security_admin.pl
Go
SnagIt
Logout
==>
<==
Add task
Remove
Save changes
Cancel
Done
Local intranet

Mascot Security - edit group: Guests

Logged in as *Administrator*

Unique ID	1
Name	Guests

Users in Group: guest

Users not in group: admin, daemon, (system), johnc, carstenp

Tasks that cannot be performed by group members:

SEARCH: Allow ms-ms (and SQ) searches
SEARCH: Allow msms no enzyme searches
SEARCH: Allow no enzyme pmf searches
SEARCH: Maximum number of concurrent searches per user
SEARCH: Maximum mascot search job priority

Tasks that *can* be performed by group members:

To remove tasks, select one or more check boxes and press *'Remove'*:

Task	Parameter
☐ SEARCH: Allow pmf searches	
☐ SEARCH: Allow all fasta databases to be searched	
☐ GENERAL: View config files using ms-status	

Help window.
Change details for an existing group. Change the users that belong to the group, and the tasks that members of the group can perform. No changes are saved until the 'Save changes' button is pressed.

Each group has a unique ID that cannot be changed.

Users can be added to or removed from groups either on this screen or from the edit/add user screens.

Mascot security is fine grained. There is a list of about 20 tasks that members of a group can (or cannot perform). The tasks that are not permitted are in the top list. To allow group members to perform one of these tasks, click on the task in the list, and then ‘Add task’. This task will then appear in the lower list. Similarly, to remove a task, click on the check box in the lower list, and click on the ‘Remove’ button. To get further information about any task, hold the mouse over the task in the lower window and further details will appear in the help box.

No changes to a group are saved until the ‘Save changes’ button is pressed.

Session files

Session files are created in the mascot/sessions directory. Sessions that have expired will be deleted automatically by ms-monitor.

Log file

The log file ‘security.log’, in the mascot/logs directory contains information about all security changes. The file is not available from any web based application for security reasons. The level of logging can be controlled from the security administration utility.

Configuration Files

Security information is saved in three configuration files in the mascot/config directory:

security_options.xml
security_tasks.xml
group.xml
user.xml

The schema for these files is mascot_security_1_0.xsd.

Use the security administration utility or Mascot Parser rather than editing these files manually.

Automating addition of new users

Mascot Parser users have access to all of the documentation for the lower level functions to administer Mascot security programmatically. The security administration utility uses some of these functions.

To simply to add a large number of users, then the add_user.pl script in the mascot/bin directory can be used:

```
Usage: add_user.pl -u username
                   -p password
                   -x password_expiry
                   -f fullname
                   -e email_address
                   -g group to which user should belong
```

The password expiry should be 0 for never expires or 1 to force the user to change the password when they first log in.

Resetting the administrator password

If the admin user password is lost, the easiest way to reset it is to re-run 'enable_security.pl' from the command line as described above. This will not affect any existing groups or users, but will just reset the password.

User ID

The user ID for each search is saved in the results file. If security is disabled, then the search ID will be set to zero. Special user IDs are listed above. Other users will have an automatically assigned IDs starting at 1000.

A

A. Basic Regular Expressions

Sequence database parsing in Mascot is defined using rules which conform to Basic Regular Expression (BRE) notation as defined in standard ISO/IEC 9945-2: 1993. BRE notation is widely used in Unix, e.g. in the grep command, but it may be less familiar to those from a DOS or Windows background. Man pages containing a rigorous definition of BRE notation can be found on most Unix systems.

The following description is much simplified, and is intended to provide just enough information to understand the existing rules in `mascot.dat`, and to enable someone without prior knowledge of regular expressions to write simple rules for new databases. Only the most basic aspects of BRE notation are touched on.

In `mascot.dat`, the PARSE section contains a number of rules. For each rule, the pattern in double quotes is a BRE which is used to identify a string so that it can be parsed from the surrounding text. For example:

```
#Report text from NCBI excluding sequence (used for AA entries)
RULE_10 "\(LOCUS .*\)ORIGIN "
```

The part of the BRE between the backslashed parentheses `\(` and `\)` is the string which we are trying to locate and extract. This rule looks for the word `LOCUS` followed by a space. It will extract all the text, including the word `LOCUS`, up to but excluding the word `ORIGIN` followed by a space.

BRE Rules

The rules for performing this match are as follows:

The BRE always looks for the longest, leftmost matching string.

Matching is case sensitive.

Newline characters (LF in Unix or CR+LF in Windows) are treated like any other character

The sub-expression to be extracted from the surrounding text is defined using backslashed parentheses `\( \)`. The parentheses are ignored for matching purposes.

Some characters are "Special":

`. [ \` The period, left-bracket and backslash are special except when used in a bracket expression.

`*` The asterisk is special except when used in a bracket expression, as the first character of an entire BRE (after an initial `^`, if any), as the first character of a subexpression (after an initial `^`, if any).

`^` The circumflex is special when used as an anchor, or as the first character of a bracket expression.

`$` The dollar sign is special when used as an anchor.

Matching Single Characters

Any character that is not a special character is an ordinary character. An ordinary character, or a special character preceded by a backslash, matches to itself.

A period, used outside a bracket expression, matches to any single character, including a newline character.

A bracket expression (a list of characters enclosed in square brackets, `[ ]`) matches any single character from the enclosed list. The following rules and definitions apply to bracket expressions:

A bracket expression is either a matching list expression or a non-matching list expression. The right-bracket `]` loses its special meaning and represents itself in a bracket expression if it occurs first in the list (after an initial circumflex `^`, if any). Otherwise, it terminates the bracket expression. The special characters: `. * [ \` (period, asterisk, left-bracket and backslash, respectively) lose their special meaning within a bracket expression.

A matching list expression matches any one of the characters in the list. The first character in the list must not be the circumflex. For example, `[abc]` matches any one of the characters `a`, `b` or `c`.

A non-matching list expression begins with a circumflex `^` and specifies a list that matches any character except for the characters in the list after the leading circumflex. For example, `[^abc]` matches any one character except the characters `a`, `b` or `c`. The circumflex will have this special meaning only when it occurs first in the list, immediately following the left-bracket.

A range expression represents the inclusive set of characters between two characters in the ASCII character set. The starting and ending characters are separated by a hyphen. For example, `[A-Z]` will match to any single upper case letter, while `[0-9_A-Za-z]` matches any single alphanumeric character.

Matching Multiple Characters

When a BRE matching a single character or a subexpression is followed by the special character asterisk `*`, together with that asterisk it matches what zero or more consecutive occurrences of the character. For example, `[ab]*` and `[ab][ab]` are equivalent when matching the string `ab`. The expression `ab*c` will match to `ac` or `abc` or `abbbbbbc`.

When a BRE matching a single character or a subexpression is followed by an interval expression of the format `\{m\}`, `\{m,\}` or `\{m,n\}`, together with that interval expression it matches what repeated consecutive occurrences of the BRE would match. The values of `m` and `n` will be decimal integers in the range `0 <= m <= n <= 255`, where `m` specifies the exact or minimum number of occurrences and `n` specifies the maximum number of occurrences. The expression `\{m\}` matches exactly `m` occurrences of the preceding BRE, `\{m,\}` matches at least `m` occurrences and `\{m,n\}` matches any number of occurrences between `m` and `n`, inclusive.

For example, in the string `abababccccccd` the BRE `c\{3\}` is matched by characters seven to nine, the BRE `\(ab\)\{4,\}` is not matched at all and the BRE `c\{1,3\}d` is matched by characters ten to thirteen.

The behaviour of multiple adjacent duplication symbols produces undefined results.

Expression Anchoring

A BRE can be limited to matching strings that begin or end a line; this is called anchoring. The circumflex and dollar sign special characters will be considered BRE anchors in the following contexts:

A circumflex `^` is an anchor when used as the first character of an entire BRE. The circumflex will anchor the expression to the beginning of a string; only sequences starting at the first character of a string will be matched by the BRE. For example, the BRE `^ab` matches `ab` in the string abcdef, but fails to match in the string `cdefab`.

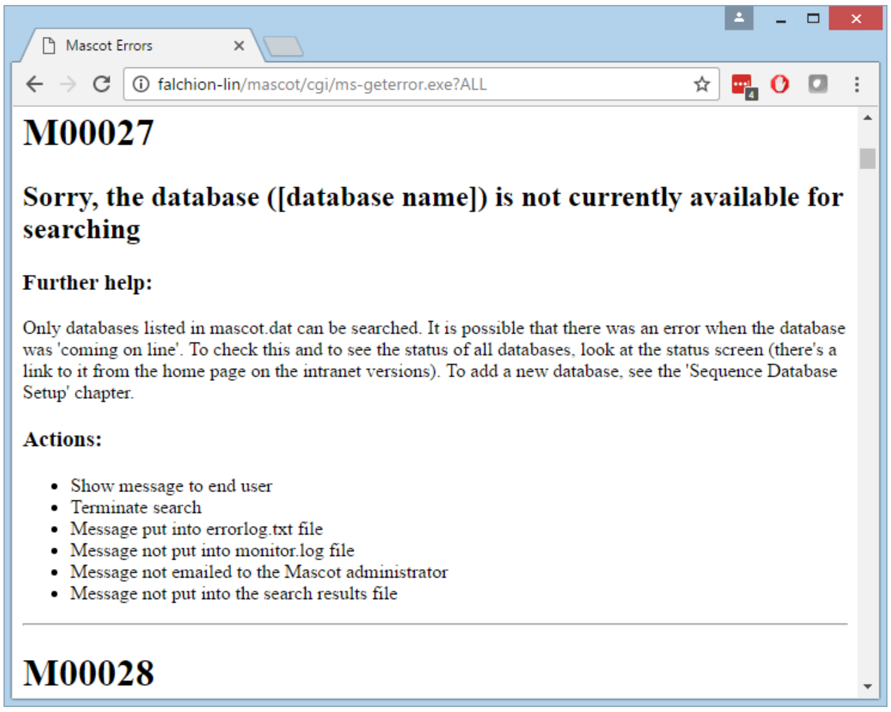
A dollar sign `$` is an anchor when used as the last character of an entire BRE. The dollar sign will anchor the expression to the end of the string being matched (not including a final newline character, if present).

A BRE anchored by both `^` and `$` matches only an entire string. For example, the BRE `^abcdef$` matches strings consisting only of `abcdef`.

B

B. Error Messages

A complete listing of Mascot error codes, messages, and explanations can be found at the URL mascot/cgi/ms-geterror.exe?ALL.



M00027

Sorry, the database ([database name]) is not currently available for searching

Further help:

Only databases listed in mascot.dat can be searched. It is possible that there was an error when the database was 'coming on line'. To check this and to see the status of all databases, look at the status screen (there's a link to it from the home page on the intranet versions). To add a new database, see the 'Sequence Database Setup' chapter.

Actions:

- Show message to end user
- Terminate search
- Message put into errorlog.txt file
- Message not put into monitor.log file
- Message not emailed to the Mascot administrator
- Message not put into the search results file

M00028

The same text can also be found in the file `errors.html` in the root directory of the Mascot CD-ROM.

C

C. System Limits

Number of different modifications in unimod.xml	unlimited
Number of enzymatic peptides per sequence	user definable (MaxNumPeptides)
Length of a sequence (number of residues)	user definable (MaxSequenceLen)
Number of seq(), comp(), and ions() type qualifiers per query	20
Maximum number of tags and etags in a search	100
Number of peptide masses (MS/MS search)	unlimited
Number of peptide masses (PMF search)	1000
Number of enzymes in the enzymes file	100
Number of protein hits saved in the results file summary section (PMF)	50
Number of peaks per MS/MS spectrum	10,000
Number of lines with name= in MIME format file	1,000,000
Maximum mass of any peptide in standard Mascot (Daltons)	16,000
Minimum mass of any peptide (Daltons)	100
Maximum mass of an unmodified amino acid residue	300
Length of any peptide in residues in standard Mascot	254
Length of name (TITLE=) for any query when ‘escaped’	30,000
Length of database name	159
Length of enzyme name	50
Length of modification name	50
Simultaneous variable modifications	32
Number of missed cleavage sites in a peptide	9
Maximum number of cleavage rules per enzyme	20

Number of active sequence databases	user definable (MaxDatabases)
Number of threads per search	1024
Number of concurrent jobs per database	100
Number of parse rules	256
Length of parse rule	128
Maximum length of an accession string	200
Maximum number of processors per server	64
Maximum number of sub-clusters in a cluster	50
Maximum number of machines in a sub-cluster	1024
Maximum number of processors in a sub-cluster	65536
Maximum number of nodes in nodelist.txt	4096
Widest precursor tolerance for PMF or MS/MS search	1%
	or 10,000 mmu
	or 10,000 ppm
	or 10 Da
Widest precursor tolerance for sequence query	25%
	or 250,000 mmu
	or 250,000 ppm
	or 250 Da

D

D. Web Server Configuration

Mascot Directory Structure

The Mascot directory structure is described in Chapter 2, Installation: Linux

Microsoft Internet Information Services

The Mascot installation program automatically configures Microsoft IIS 7.0 or later.

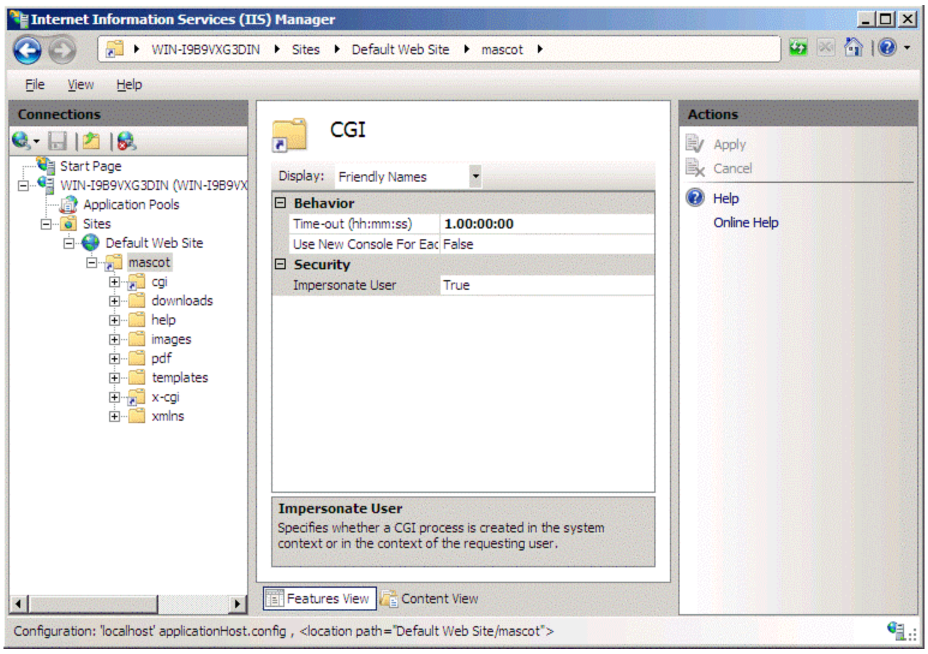
CGI Timeout

The CGI timeout is set to 1 day, and any searches running longer than this will be terminated. If you wish, you can increase this timeout.

The CGI timeout value is set only on the parent node /w3svc/1/root/mascot so that it is inherited by both the cgi and x-cgi nodes. If a different value is set at a lower level, it will override the inherited value.

IIS 7 and later

In the Windows Start menu, go to Control panel, Administrative Tools, Internet Information Services (IIS) Manager. On the connections tree, expand Sites and Default web site and select mascot. In the central pane, double click the CGI properties icon. The CGI time-out will be displayed and can be edited. If you make changes, choose Apply in the Action pane.



CGI

Display: Friendly Names

Behavior	
Time-out (hh:mm:ss)	**1.00:00:00**
Use New Console For Eac	False
Security	
Impersonate User	True

Impersonate User
Specifies whether a CGI process is created in the system context or in the context of the requesting user.

Configuration: 'localhost' applicationHost.config , <location path="Default Web Site/mascot">

If you have configured IIS 7 with multiple web sites, and the Mascot server is not installed in the default web site, you will need to browse to the appropriate location. You can also inspect the CGI timeout at other connection nodes, in case a different timeout has been set manually at the cgi node or even at the level of individual files (inadvisable).

Apache

Apache is a very rugged and popular server for Unix platforms. It is a less obvious choice for Windows, since the Mascot installation program will configure Microsoft IIS automatically.

If the URL /mascot is mapped to disk path *mascot/html*, then URL /mascot/images will correspond to disk path *mascot/html/images*. So, it is important that the entries for the *cgi* and *x-cgi* directories come before that for the *html* directory. Otherwise, the server will report that it cannot find the *cgi* and *x-cgi* paths, because it has assumed from the URL that they are sub-directories of *mascot/html*.

Linux configuration

The following lines illustrate typical mappings and permissions in Apache 2.4 and later for the Mascot directories:

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

<Directory /usr/local/mascot/cgi>
  AllowOverride None
  Options None
  Require all granted
</Directory>
ScriptAlias /mascot/cgi /usr/local/mascot/cgi
```

```
<Directory /usr/local/mascot/x-cgi>
  AllowOverride None
  Options None
  Require all granted
</Directory>
ScriptAlias /mascot/x-cgi /usr/local/mascot/x-cgi

<Directory /usr/local/mascot/html>
  AllowOverride None
  Options None
  Require all granted
</Directory>
Alias /mascot /usr/local/mascot/html
```

For Apache 2.2 and earlier, replace

```
Require all granted
```

with

```
Order allow,deny
Allow from all
```

Windows Installation

If you choose to use Apache under Windows, a good starting point for support information is:

http://httpd.apache.org/docs/2.4/platform/windows.html

Important: If IIS is installed, stop the IIS service before installing Apache, Perl and Mascot. IIS and Apache must be configured to listen on different ports.

Mascot 2.6 has been tested with Apache 2.4 on Windows 7 Ultimate x64.

After Mascot has been installed, edit the Apache configuration file (probably *C:\Apache24\conf\httpd.conf*). Copy the customised Apache configuration settings from the *httpd.conf* file in the Mascot *config* directory and paste them at the end of the Apache *httpd.conf* file.

```
<Directory C:/inetpub/mascot/cgi/>
  AllowOverride None
  Options None
  Require all granted
  ScriptInterpreterSource Script
</Directory>
ScriptAlias /mascot-apache/cgi C:/inetpub/mascot/cgi

<Directory C:/inetpub/mascot/x-cgi/>
  AllowOverride None
  Options None
  Require all granted
  ScriptInterpreterSource Script
</Directory>
ScriptAlias /mascot-apache/x-cgi C:/inetpub/mascot/x-cgi

<Directory C:/inetpub/mascot/html/>
```

```
    AllowOverride None
    Options None
    Require all granted
  </Directory>
  Alias /mascot-apache C:/inetpub/mascot/html

  TimeOut 86400
```

For Apache 2.2 and earlier, replace

```
Require all granted
```

with

```
Order allow,deny
Allow from all
```

Save the changes then stop and start Apache. You should now be able to view Mascot pages in a web browser and proceed with licence registration.

The Apache configuration has TimeOut set to 86400 (1 day) to prevent the web browser connection breaking during a search.

Mascot uses a 'private' copy of Perl located in the *mascot\perl64* directory. The Apache configuration specifies that Mascot scripts should use the shebang line (ScriptInterpreterSource Script). This is set to

```
#!/usr/local/mascot/perl64/bin/perl
```

The Mascot installer creates a Windows symbolic link between *C:\usr\local\mascot\perl64* and the actual location of the *mascot\perl64* directory. If this link is deleted, Mascot Perl scripts will no longer work.

The advantage of this arrangement is that you can have a second copy of Perl on the system, maybe a different version that is used with either Apache or IIS for other applications. Only the Mascot scripts 'see' the Mascot copy of Perl.

Keyword Indexing

If you use Apache, the keyword index required for site search will not have been built during Mascot installation because the web server mappings were not in place.

To build the keyword index, open a command window and enter the following commands. If Mascot was installed into a different path, you may have to modify the first two lines

```
C:
cd \inetpub\mascot\htdig
bin\htdig.exe -v
bin\htmerge.exe -v
```

Once the commands have completed, keyword search using the control at the top right of the web pages should be operational

User authentication.

Apache provides several ways to restrict access to directories or files. One method is to limit access to clients from a range of IP addresses or a particular domain. Another method is to require a username and password, which may be a convenient way for a system administrator to limit access to the `x-cgi` directory.

Setting up user authentication takes two steps: firstly, creating a file containing the usernames and passwords. Secondly, telling the server what resources are to be protected and which users are allowed (after entering a valid password) to access them.

Creating a User Database

A list of users and passwords needs to be created in a file. For security reasons, this file should not be under the document root. This example assumes the file is called */usr/local/mascot/config/.passwd*.

The file will consist of a list of usernames and a password for each. The format is similar to the standard Unix password file, with the username and password being separated by a colon. However you cannot just type in the usernames and passwords because the passwords are stored in an encrypted format.

The program *htpasswd* is used to add create a user file and to add or modify users. This can be found in the *bin* directory of the Apache distribution. To create a new user file and add the username "mickey", the command would be:

```
./htpasswd -c /usr/local/mascot/config/.passwd mickey
```

The *-c* argument tells *htpasswd* to create new users file. You will be prompted to enter a password for mickey, and confirm it by entering it again. Other users can be added to the existing file in the same way, except that the *-c* argument is not needed. The same command can also be used to modify the password of an existing user.

Specifying the password protected resources

Having created a password file, the next step is to modify the configuration mapping for the *x-cgi* directory. Instead of the mapping shown earlier, you would use a directive like this:

```
<Directory /usr/local/mascot/x-cgi>
  AllowOverride None
  Options None
  AuthType Basic
  AuthName Restricted
  AuthUserFile /usr/local/mascot/config/.passwd
  require valid-user
</Directory>
ScriptAlias /mascot/x-cgi /usr/local/mascot/x-cgi/
```

You will need to stop and restart Apache, or send a *kill -HUP* to the parent process, to activate the new configuration. For further information on restricting access to the server, see the “Authentication and Access Restrictions” section of the Apache FAQ documentation.

E

E. End User Licence Agreements

Mascot Server

END USER LICENCE AGREEMENT

IMPORTANT – PLEASE READ CAREFULLY: This End User Licence Agreement is a legally binding contract between you (either an individual or a single corporate entity) and Matrix Science Limited for the product identified above, which includes computer software, electronic documentation, printed documentation, and any subsequent updates and supplements (the “Software”).

By installing or using the Software, you agree to be bound by the terms of this agreement. If you do not agree to the terms of this agreement, we are unwilling to license the Software to you. In this case, do not install or use the Software. Return the Software to Matrix Science Limited or their authorised distributor within 30 days of receipt for a full refund.

1 **Licence**

Matrix Science Limited owns the copyright in the Software contained within this package and all other copies which you are authorised by this agreement to make.

This licence is personal to you (either an individual or a single corporate entity) as the purchaser of a licence to use the Software and the licence granted herein is for your benefit only.

You may not use the Software in any way that permits unlicensed access to the Software. In particular, individuals who are not party to this licence or the general public must not be permitted access to the Software through a public network such as the Internet.

2 **Permitted Users**

As purchaser of a licence to use the Software, you may, subject to the following conditions:

2.1 load the Software onto and use it on a single computer (of the type identified on the package) which is under your control; and

2.2 copy the Software for backup and archival purposes and make up to two copies of the documentation (if any) accompanying the Software provided that the original and each copy is kept in your possession and that your installation and use of the Software does not exceed that allowed by this agreement.

2.3 modify the HTML and Perl documents for sole use by yourself in connection with the Software.

3 Restrictions of Use

You may not:

3.1 load the Software into two or more computers at the same time. If you wish to transfer the Software from one computer to another, you must erase the Software from the first system before you install it onto a second system;

3.2 sub-license, assign, rent, lease or transfer the licence or the Software or make or distribute copies of the Software;

3.3 translate, reverse engineer, decompile, disassemble, modify or create derivative works based on the Software except as permitted by Law;

3.4 make copies of the Software except for backup or archival purposes as permitted hereunder;

3.5 use any backup copy of the Software (or allow anyone else to use such copies) for any purpose other than to replace the original copy in the event it is destroyed or becomes defective;

3.6 distribute copies of modified HTML or Perl documents; or

3.7 copy the written materials (except as provided by this agreement) accompanying the Software.

4 Title

As licensee, you own only the medium on which the Software is recorded. We shall at all times retain ownership of the Software.

5 Warranty

We warrant that for a period of ninety days (90) from the date of delivery ('the Warranty Period'):

5.1 the medium on which the Software is recorded will be free from defects in materials and workmanship under normal use. If the medium fails to conform to this warranty, you may, as your sole and exclusive remedy, obtain (at your option) either a replacement free of charge or a full refund if you return the defective medium to us or to your supplier during the Warranty Period; and

5.2 the copy of the Software in this package will materially conform to the documentation that accompanies the Software. If the Software fails to operate in accordance with this warranty, you may, as your sole and exclusive remedy, return all of the Software and the documentation to us or to your supplier during the Warranty Period, specifying the problem, and we will provide you either with a new version of the Software or a full refund (at your option).

6 Disclaimer

We do not warrant that this Software will meet your requirements or that its operation will be uninterrupted or error free. We exclude and hereby expressly disclaim all express and implied warranties or conditions not stated herein, so far as such exclusion is or disclaimer is permitted under the applicable law. THIS AGREEMENT DOES NOT AFFECT YOUR STATUTORY RIGHTS.

7 Liability

7.1 Our liability to you for any losses shall not exceed the amount you originally paid for the Software.

7.2 In no event will we be liable to you for any indirect or consequential damages even if we have been advised of the possibility of such damages. In particular, we accept no liability for any programs or data made or stored with the Software nor for the costs of recovering or replacing such program or data.

7.3 Nothing in this clause limits our liability to you in the event of death or personal injury resulting from our negligence.

8 Termination

8.1 The agreement and the licence hereby granted to use the Software automatically terminates if you:

8.1.1 fail to comply with any provisions of this agreement; or

8.1.2 voluntarily return the Software to us.

8.2 In the event of termination in accordance with clause 8.1 you must destroy or delete all copies of Software from all storage media in your possession.

9 Severability

In the event that any provision of this agreement is declared by any judicial or other competent authority to be void, voidable, illegal or otherwise unenforceable or indications of the same are received by either you or us from any relevant competent authority we shall amend that provision in such reasonable manner as achieves the intention of the parties without illegality, or at our discretion such provision may be severed from this agreement and the remaining provisions of this agreement shall remain in full force and effect.

10 Entire Agreement

You have read and understand this agreement and agree that it constitutes the complete and exclusive statement of the agreement between us with respect to the subject matter hereof and supersedes all proposals, representations, understandings and prior agreements, whether oral or written, and all other communications between us relating thereto.

11 Assignment

This agreement is personal to you (either an individual or a single corporate entity) and you may not assign, transfer, sub-contract or otherwise part with this agreement or any right or obligation under it without our prior written consent.

12 Law and Disputes

This agreement and all matters arising from it are governed by and construed in accordance with the laws of England and Wales, whose Courts shall have exclusive jurisdiction over all disputes arising in connection with this agreement.

If you have any questions about this agreement, write to us at Matrix Science Ltd., 64 Baker Street, London W1U 7GB, UK or call us at +44 (0)20 7486 1050 or email us at info@matrixscience.com

Xerces

Curl

gzip, ht://Dig, cksum, touch, libstdc++

```
        GNU GENERAL PUBLIC LICENSE
           Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.
                          675 Mass Ave, Cambridge, MA 02139, USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

               Preamble

  The licenses for most software are designed to take away your
freedom to share and change it.  By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users.  This
General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it.  (Some other Free Software Foundation software is covered by
the GNU Library General Public License instead.)  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.

  To protect your rights, we need to make restrictions that forbid
anyone to deny you these rights or to ask you to surrender the rights.
These restrictions translate to certain responsibilities for you if you
distribute copies of the software, or if you modify it.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must give the recipients all the rights that
you have.  You must make sure that they, too, receive or can get the
source code.  And you must show them these terms so they know their
rights.

  We protect your rights with two steps: (1) copyright the software, and
(2) offer you this license which gives you legal permission to copy,
distribute and/or modify the software.

  Also, for each author's protection and ours, we want to make certain
that everyone understands that there is no warranty for this free
software.  If the software is modified by someone else and passed on, we
want its recipients to know that what they have is not the original, so
that any problems introduced by others will not reflect on the original
authors' reputations.

  Finally, any free program is threatened constantly by software
patents.  We wish to avoid the danger that redistributors of a free
program will individually obtain patent licenses, in effect making the
program proprietary.  To prevent this, we have made it clear that any
patent must be licensed for everyone's free use or not licensed at all.

  The precise terms and conditions for copying, distribution and
modification follow.
```

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

bzip2

SWIG

Linux glibc (section 6b applies)

GNU Lesser General Public License

Version 2.1, February 1999

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence the
version number 2.1.]

G.0.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software—typically libraries—of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the *Lesser* General Public License because it does *Less* to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a “work based on the library” and a “work that uses the library”. The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called “this License”). Each licensee is addressed as “you”.

   A “library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a “work that uses the Library” with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

   You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

   a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable “work that uses the Library”, as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
   b. Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
   c. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
   d. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
   e. Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

   For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the

Regex

HWLOC

C Clustering Library