## Table of Contents

System Requirements ................................................................. 1

Supported Operating Systems .................................................. 2
  Supported for production use of Fusion services ....................... 2
  Also supported for trial and development use of Fusion services .... 2

Hardware requirements .............................................................. 3
  Fusion and Solr nodes .......................................................... 3
  ZooKeeper nodes .................................................................. 3
  Fusion components .............................................................. 3

Java requirements ..................................................................... 5
  Java 11 licensing requirements .............................................. 5
  Checking your Java version ................................................... 5
  JavaScript pipeline stages ..................................................... 5

Cluster Requirements ................................................................ 6
  Supported Solr Versions ....................................................... 6
  Supported ZooKeeper Versions ............................................. 6

Recommended HTTP Clients ..................................................... 7

Supported Components .............................................................. 8

Deployment Types ..................................................................... 9
  Deployment goals ................................................................. 10
  Cluster Arrangements ............................................................ 13
    Nodes running core Fusion services and Solr also run ZooKeeper 13
    Nodes running ZooKeeper aren’t running core Fusion services or Solr . 14

New Installation ....................................................................... 15
  Install Fusion on a Single Node .............................................. 16
    Ports ................................................................................. 16
    Unix installation ............................................................... 18
      Starting Fusion ............................................................... 18
      Running Fusion In The Foreground ................................... 18
      Stopping Fusion ............................................................ 19
      Using systemd to manage processes ................................. 19
      Using Ubuntu Upstart to manage processes ...................... 19
    Windows installation ......................................................... 19
      Starting Fusion ............................................................... 19
      Stopping Fusion ............................................................ 20
    Installation with an existing Solr instance or cluster ............. 20
    Troubleshooting ............................................................... 20

Install a Fusion Cluster ............................................................ 21
  Install Fusion Server with Bundled Solr and ZooKeeper .......... 21
  Supported Cluster Arrangements ........................................... 21
  Hardware and software requirements .................................... 21
  Next steps ............................................................................ 21
  Install a Fusion Cluster (Unix) ............................................... 22
## System Requirements

Requirements for Fusion installation are detailed below.

<table>
<thead>
<tr>
<th>Important</th>
<th>Lucidworks recommends <em>not</em> virus scanning the <code>fusion/data</code> folder. Virus scanning can cause slow performance, and it can cause downtime if it quarantines an index file identified as a possible virus.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tip</td>
<td>Fusion does not support installation under a directory whose name contains a space.</td>
</tr>
</tbody>
</table>
### Supported Operating Systems

<table>
<thead>
<tr>
<th>Note</th>
<th>Certain systems may place limits on open files or processes that may be used by a process or user (for example, Unix systems may use <code>ulimit</code> and the <code>limits.conf</code> file). Fusion may require increasing these limits. We recommend setting these to “unlimited” or the maximum allowed by the system. If “unlimited” is not an option, we suggest a minimum limit of 65535 on the number of open files, and a minimum limit of 16384 on the number of processes for the Fusion user.</th>
</tr>
</thead>
</table>

### Supported for production use of Fusion services

#### Operating system:

- Windows Server 2012, 2012 R2, and 2016 (x64 only)

<table>
<thead>
<tr>
<th>Note</th>
<th>Windows systems must have the Microsoft Visual C++ 2010 SP1 package installed.</th>
</tr>
</thead>
</table>

- Linux 64-bit (x86_64/amd64 only) with 2.6 or later kernel

<table>
<thead>
<tr>
<th>Note</th>
<th>RedHat based Linux, including CentOS, must be 6.6.x or later, or else the bug fix <a href="https://rhn.redhat.com/errata/RHSA-2013-1605.html">https://rhn.redhat.com/errata/RHSA-2013-1605.html</a> must have been applied. This fix remediates a RedHat bug that causes Fusion to hang. See <a href="http://www.infoq.com/news/2015/05/redhat-futex">http://www.infoq.com/news/2015/05/redhat-futex</a> for more information.</th>
</tr>
</thead>
</table>

#### JVM:

See Java requirements below.

### Also supported for trial and development use of Fusion services

#### Operating system:

- Windows 7, 8, 8.1, and 10 64-bit (x64 only)
- Mac OS X 10.8 and above

For information about the embedded versions of Solr, ZooKeeper, and so forth, see the [Release Notes](#).

#### JVM:

See Java requirements below.
Hardware requirements

Fusion and Solr nodes

Here are some minimum recommendations for different Fusion deployments. These are for the nodes that run Fusion (including ones that also run Solr and/or ZooKeeper, if those cluster arrangements are used), as well as for external (non-Fusion) nodes for SolrCloud nodes:

<table>
<thead>
<tr>
<th>Deployment type</th>
<th>Memory</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development/Testing</td>
<td>16 GB</td>
<td>4 cores</td>
</tr>
<tr>
<td>Small Production*</td>
<td>32 GB</td>
<td>8 cores</td>
</tr>
<tr>
<td>Large Production**</td>
<td>64+ GB</td>
<td>8+ cores</td>
</tr>
</tbody>
</table>

- Small production environments will have 2+ nodes with these specs, on which both Fusion and Solr are installed. (3+ nodes are recommended.)
- Large production environments will have 3+ nodes with these specs. Additional nodes are needed for an external SolrCloud cluster, if that is used. An external SolrCloud cluster is only necessary for Solr collections with very large numbers of documents (for example, 100 million documents or more), though a Fusion deployment can use it regardless. On these nodes, Fusion and Solr are installed if no external SolrCloud cluster is used. If an external SolrCloud cluster is used, these nodes might or might not have Solr installed.

ZooKeeper nodes

Here are some minimum recommendations for the nodes that run ZooKeeper. These nodes can be more lightweight than the nodes that run Fusion and/or Solr. With the exception of small development/testing deployments, ZooKeeper nodes should only run ZooKeeper (not Fusion or Solr). Because ZooKeeper synchronizes all its operations to disk, we recommend using a disk with high throughput and low latency for your ZooKeeper nodes.

<table>
<thead>
<tr>
<th>Deployment type</th>
<th>Memory</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development/Testing</td>
<td>4 GB</td>
<td>4 cores</td>
</tr>
<tr>
<td>Small Production</td>
<td>4+ GB</td>
<td>4+ cores</td>
</tr>
<tr>
<td>Large Production</td>
<td>4+ GB</td>
<td>4+ cores</td>
</tr>
</tbody>
</table>

Fusion components

You can run Fusion components on different nodes. Different Fusion components require different amounts of resources. Below are the minimum recommended memory requirements; consult Lucidworks for specific recommendations tailored to your unique use case, data load, and production needs.

<table>
<thead>
<tr>
<th>Fusion component</th>
<th>Minimum RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connectors</td>
<td>2 GB</td>
</tr>
<tr>
<td>API service</td>
<td>1 GB</td>
</tr>
<tr>
<td>Fusion component</td>
<td>Minimum RAM</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Fusion UI</td>
<td>512 MB</td>
</tr>
<tr>
<td>Solr</td>
<td>2 GB</td>
</tr>
<tr>
<td>Spark master</td>
<td>512 MB</td>
</tr>
<tr>
<td>Spark worker</td>
<td>1 GB</td>
</tr>
</tbody>
</table>
Java requirements

Fusion is a Java-based application and requires a pre-installed JDK to start. The JRE alone is not sufficient:

- **Oracle JDK 1.8**, 64-bit (x64 only)
- **OpenJDK JDK 1.8**, 64-bit (x64/x86_64 only)

| Note | Java 8 is recommended for Lucidworks Fusion 5.0. In some cases, Java 9 can be used without issue. Java 10 and later is not supported by Lucidworks Fusion 5.0. |

Future releases of Lucidworks Fusion will support and/or require the use of Java 11.

**Java 11 licensing requirements**

Starting with Java 11, Oracle has changed the license requirements for commercial or production use of the Oracle Java JDK. In particular, such use may require purchase of a subscription.

Oracle also offers a Java JDK under an open source license as the OpenJDK. This is available for commercial use at no additional cost. If you have been using the OpenJDK to run Lucidworks Fusion, then the licensing change will not affect you.

If you have been using the Oracle JDK at no cost to run Lucidworks Fusion, and wish to continue using a free Java runtime after moving to Java 11 or later, you should use the OpenJDK versions of Java.

If you have a commercial license for Oracle Java, you might be able to continue using Oracle JDK under the same license.

For additional details, see Oracle JDK Releases for Java 11 and Later on the Oracle blog.

**Checking your Java version**

Fusion scripts execute both the java and javac commands, which are usually aliases for the current Java installation. To check which version of Java is invoked, run the following commands from a shell or terminal window:

```
java -version
javac -version
echo $JAVA_HOME    // Unix
echo %JAVA_HOME%   // Windows
```

**JavaScript pipeline stages**

Fusion's JavaScript pipeline stages are application-specific custom JavaScript programs. The first time a pipeline is run, the JDK compiles the JavaScript using the Nashorn JavaScript engine.
Cluster Requirements

Supported Solr Versions

Fusion includes an embedded instance of Solr; see the Release Notes to find out which Solr version is included in each Fusion release.

Fusion Server 5.0 is certified with Solr

If your search requirements are very large (for example, 100 million documents or more), we recommend that you use an external SolrCloud cluster. (You can use an external SolrCloud cluster regardless.)

 Optionally, you can use both embedded Solr instances and an external SolrCloud cluster. In this case, only store logs in the embedded Solr instances. Store data in the external SolrCloud cluster.

If you use an external SolrCloud cluster, it has ZooKeeper bundled with it. However, Apache recommends that you not use the bundled ZooKeeper in production. Instead, create an external ZooKeeper cluster (external to both Fusion and SolrCloud) or use the ZooKeeper embedded with Fusion, depending on the circumstance. For more information, see Supported ZooKeeper Versions.

| Note | If you decide to set up an external SolrCloud cluster, check the requirements for this as well in the Solr documentation. |

We strongly recommend that you use Network Time Protocol (NTP) on a SolrCloud cluster to ensure that nodes use synchronized time. While this is not strictly required, reasoning about log contents and database entries becomes impossible without it. Information and instructions on how to install and run NTP are available at www.ntp.org.

Supported ZooKeeper Versions

Fusion includes an embedded instance of ZooKeeper; see the release history to find out which version of ZooKeeper is included in each Fusion release. For demo, trial, and some testing deployments, you can use the embedded ZooKeeper. For production deployments, we recommend using an external ZooKeeper cluster.

ZooKeeper 3.4.11 is supported for external ZooKeeper instances.

A ZooKeeper cluster must have 3+ nodes configured as an ensemble, and it must have an odd number of nodes. We recommend three ZooKeeper nodes in a ZooKeeper cluster/ensemble. Only for very large SolrCloud clusters (50+ nodes) might you need more ZooKeeper nodes.
Recommended HTTP Clients

The Fusion API can be accessed from any HTTP client, and allows you to build user interfaces and applications that work with any browser. However, the Fusion Administration UI, Dashboards, and built-in Search UI are supported only with:

- Chrome latest version
- Firefox latest version and latest Extended Support Release (ESR)
- Internet Explorer 11
## Supported Components

Fusion Server comes with a number of third-party components. The table below shows the supported versions of those components for each version of Fusion Server. Other versions are not guaranteed to be compatible with Fusion.

<table>
<thead>
<tr>
<th>Fusion</th>
<th>Solr</th>
<th>ZooKeeper</th>
<th>Spark</th>
<th>Jetty</th>
<th>Ignite</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0.1</td>
<td>8.2</td>
<td>3.4.13</td>
<td>2.4.3</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>5.0.0</td>
<td>8.2</td>
<td>3.4.13</td>
<td>2.4.3</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>4.2.5</td>
<td>7.7.2</td>
<td>3.4.13</td>
<td>2.3.2</td>
<td>9.4.19.v20190610</td>
<td>2.6.0</td>
</tr>
<tr>
<td>4.2.4</td>
<td>7.7.2</td>
<td>3.4.13</td>
<td>2.3.2</td>
<td>9.4.19.v20190610</td>
<td>2.6.0</td>
</tr>
<tr>
<td>4.2.3</td>
<td>7.7.2</td>
<td>3.4.13</td>
<td>2.3.2</td>
<td>9.4.12.v20180830</td>
<td>2.6.0</td>
</tr>
<tr>
<td>4.2.2</td>
<td>7.5</td>
<td>3.4.13</td>
<td>2.3.2</td>
<td>9.4.12.v20180830</td>
<td>2.6.0</td>
</tr>
<tr>
<td>4.2.1</td>
<td>7.5</td>
<td>3.4.13</td>
<td>2.3.2</td>
<td>9.4.12.v20180830</td>
<td>2.6.0</td>
</tr>
<tr>
<td>4.2.0</td>
<td>7.5</td>
<td>3.4.13</td>
<td>2.3.1</td>
<td>9.4.11.v20180605</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.1.3</td>
<td>7.4.0</td>
<td>3.4.13</td>
<td>2.3.1</td>
<td>9.3.25.v20180904</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.1.2</td>
<td>7.4.0</td>
<td>3.4.13</td>
<td>2.3.1</td>
<td>9.3.25.v20180904</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.1.1</td>
<td>7.4.0</td>
<td>3.4.13</td>
<td>2.3.1</td>
<td>9.3.25.v20180904</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.1.0</td>
<td>7.4.0</td>
<td>3.4.11</td>
<td>2.3.1</td>
<td>9.3.8.v20160314</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.0.2</td>
<td>7.2.1</td>
<td>3.4.10</td>
<td>2.2.1</td>
<td>3.8.10</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.0.1</td>
<td>7.2.1</td>
<td>3.4.10</td>
<td>2.2.1</td>
<td>3.8.10</td>
<td>2.3.0</td>
</tr>
<tr>
<td>4.0.0</td>
<td>7.2.1</td>
<td>3.4.10</td>
<td>2.2.1</td>
<td>3.8.10</td>
<td>2.3.0</td>
</tr>
</tbody>
</table>
Deployment Types

The Fusion platform is designed to support enterprise search applications at any scale. You can deploy Fusion across multiple nodes in order to store large amounts of data or to achieve high processing throughput or both.

Fusion consists of a number of Java processes that run in JVMs, including the api, connectors-classic, connectors-rpc, and admin-ui processes, and possibly others such as spark-master and spark-worker. When you start Fusion, the processes that start are listed. You might also see zookeeper and solr processes, depending on the cluster arrangement.

For more information about Fusion components, see Fusion Components. For a complete list of Fusion services, see Start and stop Fusion.
**Deployment goals**

- **Demo, trial, and development deployments** – The simplest possible architecture is the one you get out of the box, by unpacking the tar/zip file and running `fusion/5.0.x/bin/fusion start`, so that all components (including the bundled Solr and ZooKeeper instances) run on a single host in their default configurations.

You can quickly install and run Fusion on a computer (even on your laptop) to explore its features and work with sample data. See Quickstart for instructions. This diagram illustrates a single-node Fusion deployment:
Fusion node

Fusion services

Core services
admin-ui
api
proxy
webapps

Connector services
connectors-classic
connectors-rpc

Spark services *
spark-master
spark-worker
sql

Solr service
solr

ZooKeeper service
zookeeper

* Depends on features used
• **Onsite late-stage development and test deployments** – Ideally, an onsite deployment for late-stage development and testing should have the same architecture as the production deployment, though it doesn’t need to be scaled to provide the same level of service.

• **Production deployments** – Fusion is designed for flexible, distributed deployment. Any of its components can be distributed across your network, and some can be clustered. A production deployment requires multiple Fusion nodes, each of which runs some or all Fusion services (including Solr and ZooKeeper).
Cluster Arrangements

You can deploy Fusion across multiple nodes in a *Fusion cluster* and use a ZooKeeper cluster as the centralized, synchronized store for both application configurations and user data.

Regarding Solr, if you already have SolrCloud clusters managing your data, you can integrate them into a Fusion deployment.

To satisfy processing requirements, install Fusion, ZooKeeper, and Solr on specific nodes. These are the possibilities:

**Nodes running core Fusion services and Solr also run ZooKeeper**

In this cluster arrangement, a ZooKeeper cluster runs on the same nodes that run core Fusion services and Solr.

**Fusion cluster**

<table>
<thead>
<tr>
<th>Node 1</th>
<th>Node 2</th>
<th>Node 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion services</td>
<td>Fusion services</td>
<td>Fusion services</td>
</tr>
<tr>
<td>UI/API services</td>
<td>UI/API services</td>
<td>UI/API services</td>
</tr>
<tr>
<td>Connector services</td>
<td>Connector services</td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services <strong>•</strong></td>
<td>Spark services <strong>•</strong></td>
<td>Spark services <strong>•</strong></td>
</tr>
<tr>
<td>Solr service</td>
<td>Solr service</td>
<td>Solr service</td>
</tr>
<tr>
<td>ZooKeeper service</td>
<td>ZooKeeper service</td>
<td>ZooKeeper service</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 4</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion services</td>
<td><strong>ZooKeeper cluster</strong></td>
</tr>
<tr>
<td>UI/API services</td>
<td><strong>Solr cluster</strong></td>
</tr>
<tr>
<td>Connector services</td>
<td></td>
</tr>
<tr>
<td>Spark services <strong>•</strong></td>
<td></td>
</tr>
<tr>
<td>Solr service</td>
<td></td>
</tr>
</tbody>
</table>

--- Scale out nodes as needed.

* Depends on features used

Here are installation instructions for Unix and Windows.
Nodes running ZooKeeper aren’t running core Fusion services or Solr

In this cluster arrangement, the ZooKeeper cluster runs on nodes in the Fusion cluster on which core Fusion services and Solr aren’t running.

**Fusion cluster**

<table>
<thead>
<tr>
<th>Node 1</th>
<th>Node 2</th>
<th>Node 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion services</td>
<td>Fusion services</td>
<td>Fusion services</td>
</tr>
<tr>
<td>UI/API services</td>
<td>UI/API services</td>
<td>UI/API services</td>
</tr>
<tr>
<td>Connector services</td>
<td>Connector services</td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services *</td>
<td>Spark services *</td>
<td>Spark services *</td>
</tr>
<tr>
<td>Solr service</td>
<td>Solr service</td>
<td>Solr service</td>
</tr>
</tbody>
</table>

* Depends on features used

... Scale out nodes as needed...

<table>
<thead>
<tr>
<th>Node Z1</th>
<th>Node Z2</th>
<th>Node Z3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion service</td>
<td>Fusion service</td>
<td>Fusion service</td>
</tr>
<tr>
<td>ZooKeeper service</td>
<td>ZooKeeper service</td>
<td>ZooKeeper service</td>
</tr>
</tbody>
</table>

**Key**

- ZooKeeper cluster
- Solr cluster

Here are installation instructions for Unix and Windows.
New Installation

Install new instances of Fusion:

• Install Fusion on a Single Node

  Start here for the simplest deployment, which is useful for evaluation, demo, and development purposes. You can install Fusion on a server, desktop, or laptop.

• Install a Fusion Cluster (Introduction)

  For production deployments, install Fusion on multiple servers (a Fusion cluster). Start with this introduction. When you install Fusion Server, the software includes bundled versions of Solr and ZooKeeper. Install Fusion on every node. Then, configure Fusion so that it doesn’t start Solr and ZooKeeper on nodes where they aren’t needed.

• Install a Fusion Cluster (Unix)

  Install a Fusion Cluster on Unix nodes. When you install Fusion Server, the software includes bundled versions of Solr and ZooKeeper.

• Install a Fusion Cluster (Windows)

  Install a Fusion cluster on Windows nodes. When you install Fusion Server, the software includes bundled versions of Solr and ZooKeeper.

• Integrate Fusion with an Existing Solr Deployment

  Use these instructions when installing Fusion in an existing Solr environment.
Install Fusion on a Single Node

Note

These instructions are for an initial installation of Fusion on a single node (computer). To install Fusion on multiple nodes (a Fusion cluster), see Install a Fusion Cluster. If you already have a version of Fusion installed and want to upgrade it, see the Fusion upgrade instructions.

Out of the box, Fusion uses the instances of Solr, ZooKeeper, and Spark that are included in the Fusion distribution. See the Release Notes to find out which versions of Solr, Spark, and ZooKeeper are included in each Fusion release.

To use Fusion with an existing Solr deployment, see Integrating with existing Solr instances.

Ports

This table lists the default port numbers used by Fusion processes. Port settings are defined in the fusion.properties file in fusion/5.0.x/conf/ (on Unix or macOS) or fusion\5.0.x\conf\ (on Windows).

<table>
<thead>
<tr>
<th>Port</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>8091</td>
<td>Fusion agent</td>
</tr>
<tr>
<td>8763</td>
<td>Fusion UI service (use port 8764 to access the Fusion UI)</td>
</tr>
<tr>
<td>8764</td>
<td>Fusion proxy</td>
</tr>
<tr>
<td></td>
<td>This service includes the Fusion Authorization Proxy.</td>
</tr>
<tr>
<td>8765</td>
<td>Fusion API Services</td>
</tr>
<tr>
<td>8766</td>
<td>Spark Master</td>
</tr>
<tr>
<td>8769</td>
<td>Spark Worker</td>
</tr>
<tr>
<td>8771</td>
<td>Connectors RPC Service</td>
</tr>
<tr>
<td></td>
<td>This service can distribute connector jobs to as many Fusion nodes as you want. It uses HTTP/2 and has an SDK that you can use to build your own connectors.</td>
</tr>
<tr>
<td>8780</td>
<td>Web Apps</td>
</tr>
<tr>
<td></td>
<td>This service delivers the UIs of Fusion apps.</td>
</tr>
<tr>
<td>Port</td>
<td>Service</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>8781</td>
<td>Log shipper</td>
</tr>
<tr>
<td></td>
<td>Monitoring port that agent uses to check the health of the log shipper process. This port does not need to be accessible from other nodes.</td>
</tr>
<tr>
<td>8983</td>
<td>Solr</td>
</tr>
<tr>
<td></td>
<td>This is the embedded Solr instance included in the Fusion distribution.</td>
</tr>
<tr>
<td>8984</td>
<td>Connectors Classic Service</td>
</tr>
<tr>
<td></td>
<td>This service runs nondistributed connector jobs. It uses HTTP/1.1 and has no SDK.</td>
</tr>
<tr>
<td>9983</td>
<td>ZooKeeper</td>
</tr>
<tr>
<td></td>
<td>The embedded ZooKeeper used by Fusion services.</td>
</tr>
</tbody>
</table>

**Important**

The ZooKeeper port is also defined in the configuration file for the embedded ZooKeeper, `fusion/5.0.x/conf/zookeeper/zoo.cfg` (on Unix or macOS) or `fusion\5.0.x\conf\zookeeper\zoo.cfg` (on Windows). Look for `clientPort`. If you run Fusion with the embedded ZooKeeper, remember to change the port number in both places.

47100-48099

Apache Ignite TCP communication port range (used by the API, Connectors Classic, Connectors RPC, and Proxy services)

48100-48199

Apache Ignite shared memory port range (used by the API, Connectors Classic, Connectors RPC, and Proxy services)

49200-49299

Apache Ignite discovery port range (used by API, Connectors Classic, Connectors RPC, and Proxy services)

Additional ports might be required. See Port configuration for more information or to modify the default ports before starting Fusion.
Unix installation

Fusion for Unix is distributed as a gzipped tar file.

How to install Fusion on Linux or Mac

1. Verify that the node on which you plan to install Fusion meets hardware and software requirements.

2. Download the Fusion tar/zip file for the latest version of Fusion and move it to where you would like it to reside in your filesystem (if you would like to use Upstart for process management, you must install Fusion in /opt/lucidworks).

3. Become the user that will run Fusion.

<table>
<thead>
<tr>
<th>Important</th>
<th>Do not run Fusion as the root user.</th>
</tr>
</thead>
</table>

4. Change your working directory to the directory in which you placed the fusion-version.x.tar.gz file, for example:

```
$ cd /opt/lucidworks
```

| Tip | Fusion does not support installation under a directory whose name contains a space. |

5. Unpack the archive with tar -xf (or tar -xvf), for example:

```
$ tar -xf fusion-version.x.tar.gz
```

The resulting directory is named fusion/5.0.x. You can rename this if you wish. This directory is considered your Fusion home directory. See Directories, Files, and Ports for the contents of the fusion/5.0.x directory.

Starting Fusion

All Fusion start scripts must be executed by a user who has permissions to read and write to the directories where Fusion is installed. These scripts don’t need to be run as root (or sudo), nor should they be. Use a suitable user, or create a new one, and then ensure that it owns the directory where Fusion resides, (for example, C:\lucidworks).

Give the commands that follow from the directory fusion/5.0.x/bin.

Start the required services that are defined in the group.default property.

How to start all required services

```
./fusion start
```

| Tip | This is equivalent to ./fusion start default. You can omit the group name default. |

For information about starting groups of services or individual services, see Start and Stop Fusion.

Running Fusion In The Foreground

To run Fusion or any of its services in the foreground, use the run command-line argument in place of start.
Stopping Fusion

To stop Fusion or any of its services, use the `stop` command-line argument in place of `start`.

Using systemd to manage processes

On Red Hat Enterprise Linux, CentOS 7 and newer, and Ubuntu 15.04 LTS and newer, we support using the operating system-provided `systemd` for process management.

For more information about using `systemd`, see Using systemd to manage processes.

Using Ubuntu Upstart to manage processes

Under Ubuntu 12.04 LTS through Ubuntu 14.10, we support using Upstart for process management. This requires Fusion to be installed in the `/opt/lucidworks/` directory.

For more information about using Upstart, see Using Ubuntu Upstart to manage processes.

Windows installation

Fusion for Windows is distributed as a compressed zip file. To unpack the Fusion zip file on Windows, you can use a native compression utility or the freely available 7zip file archiver. Visit the 7zip download page for the latest version.

How to install Fusion on Windows

1. Verify that the node on which you plan to install Fusion meets hardware and software requirements.
2. Download the zip file for the latest version of Fusion and move it to where you would like Fusion to reside in your filesystem. It will appear as a compressed folder.
3. Unpack the archive. In most cases, you need only right-click and choose “Extract all...”. If you don’t see this option, check that you have permissions to extract folders on your system.
   
   The resulting directory is named `fusion\5.0.x`. This directory is considered your Fusion home directory. See Directories, Files, and Ports for the contents of the `fusion\5.0.x` directory.

To install Fusion as a set of Windows services

1. Run `bin\install-services.cmd`.
2. Enter the name of the windows user that is used to launch this service.
   
   Remember the username is `COMPUTERNAME\username` or `DOMAIN\username` (if your computer is part of a Windows domain).
3. Enter the user's password.
4. Enter the path to the directory containing the JDK to use for running the services.

Starting Fusion

All Fusion start scripts must be executed by a user who has permissions to read and write to the directories where Fusion is installed. Ensure that the user owns the directory where Fusion resides (for example, `C:\lucidworks`).
Give the commands that follow from the directory `fusion5.0.x\bin`.

How to start all required Fusion services as Java processes

```
fusion.cmd start
```

How to start all required Fusion services as Windows services

```
start-services.cmd
```

For information about starting groups of services or individual services, see Start and Stop Fusion.

**Stopping Fusion**

To stop Fusion or any of its services, use the `stop` command-line argument in place of `start`.

**Installation with an existing Solr instance or cluster**

Before you install Fusion with an existing Solr instance or cluster, confirm that the Solr version is supported by Fusion.

If installing Fusion to work with an existing Solr instance, either in SolrCloud mode or standalone, you should install Fusion as described above. You should start each of the services as described above.

Once Fusion installation is complete, you can register your existing Solr installation with Fusion to be able to use the two systems together. For details on how to do that, see the section Integrate Fusion with an Existing Solr Deployment.

**Troubleshooting**

For information about problems you might encounter when installing Fusion, and solutions, see Troubleshoot When Installing Fusion.
Install a Fusion Cluster

To support enterprise search applications at any scale, install Fusion on multiple nodes (a Fusion cluster). Deployed in this manner, Fusion can store large amounts of data and achieve high processing throughput.

| Note | Historically, “node” has also had a meaning at Lucidworks related to licensing and to the number of CPU cores. We don’t mean that here. In the context of this documentation “node” is a computer, usually a physical server but also possibly a virtual machine (especially for test environments). |

See also this video tutorial for building a multi-mode Fusion cluster:

To install Fusion on a single node, see Install Fusion on a Single Node.

Install Fusion Server with Bundled Solr and ZooKeeper

When you install Fusion Server, the software includes bundled versions of Solr and ZooKeeper. Install Fusion Server on every node. Then, configure Fusion Server so that it doesn’t start Solr and ZooKeeper on nodes where they aren’t needed.

Supported Cluster Arrangements

To satisfy processing requirements, install Fusion Server, ZooKeeper, and Solr on specific nodes.

Before proceeding, review the supported cluster arrangements.

Hardware and software requirements

Verify that the nodes on which you plan to install Fusion meet hardware and software requirements.

Next steps

Perform the installation steps in one of these articles:

- Install a Fusion Cluster (Unix)
- Install a Fusion Cluster (Windows)

If you already have a Solr deployment, use the instructions Integrate Fusion with an Existing Solr Deployment.
Install a Fusion Cluster (Unix)

This article describes how to install a Fusion cluster on multiple Unix nodes. Instructions are given for each of the cluster arrangements described in Deployment Types.

Preliminary steps

Before proceeding to one of the sections that follow, perform these steps:

How to prepare for setting up a Fusion cluster

1. Prepare your firewall so that the Fusion nodes can communicate with each other. The default ports list contains a list of all ports used by Fusion. From this list, it is important that the ZooKeeper ports, Apache Ignite ports, and the Spark ports (if you are using Spark) are open between the different nodes for cross-cluster communication.

   If you plan to use an external SolrCloud cluster and/or an external ZooKeeper cluster, then also prepare your firewall so that Fusion nodes can communicate with the SolrCloud and ZooKeeper nodes.

2. Fusion for Unix is distributed as a compressed archive file (.tar.gz) Download the Fusion compressed archive file to each node that will run Fusion.

<table>
<thead>
<tr>
<th>Note</th>
<th>To leverage the copies of Solr and/or ZooKeeper that are distributed with Fusion on nodes that will not run Fusion (as a simple means of obtaining compatible versions of the other software), also download the Fusion compressed archive file to each of those nodes. Below, you will edit configuration files so that Fusion doesn’t run on those nodes.</th>
</tr>
</thead>
</table>

3. On each node, change your working directory to the directory in which you placed the Fusion tar/zip file and unpack the archive, for example:

   ```
   $ cd /opt/lucidworks
   $ tar -xf fusion-version.x.tar.gz
   ```

<table>
<thead>
<tr>
<th>Tip</th>
<th>Fusion does not support installation under a directory whose name contains a space.</th>
</tr>
</thead>
</table>

   The resulting directory is named fusion/5.0.x. You can rename this if you wish. This directory is considered your Fusion home directory. See Directories, files, and ports for the contents of the fusion/5.0.x directory.

<table>
<thead>
<tr>
<th>Important</th>
<th>In the sections that follow, for every step on multiple nodes, complete the step on all nodes before going to the next step. It is especially important that you don’t start Fusion on any node until the instructions say to do so.</th>
</tr>
</thead>
</table>

In the steps below, the port numbers reflect default port numbers and one common choice (port 2181 for nodes in an external ZooKeeper cluster). Port numbers for your nodes might differ.
Nodes running core Fusion services and Solr also run ZooKeeper

In this cluster arrangement, a ZooKeeper cluster runs on the same nodes that run core Fusion services and Solr.

**Fusion cluster**

 NODE 1

<table>
<thead>
<tr>
<th>Fusion services</th>
<th>Core services</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services</td>
<td>Solr service</td>
</tr>
<tr>
<td>ZooKeeper service</td>
<td></td>
</tr>
</tbody>
</table>

 NODE 2

<table>
<thead>
<tr>
<th>Fusion services</th>
<th>Core services</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services</td>
<td>Solr service</td>
</tr>
<tr>
<td>ZooKeeper service</td>
<td></td>
</tr>
</tbody>
</table>

 NODE 3

<table>
<thead>
<tr>
<th>Fusion services</th>
<th>Core services</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services</td>
<td>Solr service</td>
</tr>
<tr>
<td>ZooKeeper service</td>
<td></td>
</tr>
</tbody>
</table>

 NODE 4

<table>
<thead>
<tr>
<th>Fusion services</th>
<th>Core services</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services</td>
<td>Solr service</td>
</tr>
</tbody>
</table>

**Key**

- ZooKeeper cluster
- Solr cluster

---

* Scale out nodes as needed.

* Depends on features used

---

How to set up a Fusion cluster

Perform the steps in the section Preliminary steps, and then perform these steps:

1. Assign a number to each Fusion node, starting at 1. We refer to the number we assign to each node as the ZooKeeper myid.

2. On each Fusion node, create a fusion/5.0.x/data/zookeeper directory, and a file called myid in that directory. Edit the file and save the ZooKeeper myid assigned for this node as the only contents.

3. On each Fusion node, open the fusion/5.0.x/conf/zookeeper/zoo.cfg file in a text editor and add the following after the clientPort line (change the hostnames or IP addresses to the correct ones for your servers):

   ```
   server.1=[Hostname or IP for ZooKeeper with myid 1]:2888:3888
   server.2=[Hostname or IP for ZooKeeper with myid 2]:2888:3888
   server.3=[Hostname or IP for ZooKeeper with myid 3]:2888:3888
   ```
For example:

| server.1=10.10.31.130:2888:3888 |
| server.2=10.10.31.178:2888:3888 |
| server.3=10.10.31.166:2888:3888 |

+ NOTE: Don't use localhost or 127.0.0.1 as the hostname/IP. Specify the hostname/IP that other nodes will use when communicating with the current node.

1. On each Fusion node, edit default.zk.connect in fusion/5.0.x/conf/fusion.properties to point to the ZooKeeper hosts:

   default.zk.connect=[ZK host 1]:9983,[ZK host 2]:9983,[ZK host 3]:9983

2. On each node, start ZooKeeper with bin/zookeeper start. Zookeeper should start without errors. If a ZooKeeper instance fails to start, check the log at fusion/5.0.x/var/log/zookeeper/zookeeper.log.

3. On each node, start the rest of Fusion using bin/fusion start.

4. Create an admin password and log in to Fusion at http://FIRST_NODE_IP:8764, where FIRST_NODE_IP is the IP address of your first Fusion node.

5. Verify the Solr cluster is healthy by looking at http://ANY_NODE_IP:8983/solr/#/cloud, where ANY_NODE_IP is the IP address of a Solr node. All of the nodes should appear green.

6. If necessary, prepare high availability by setting up a load balancer in front of Fusion so that it load balances between the Fusion UI URL's at http://NODE_IP:8764.

   Consult your load balancer's documentation for instructions.

### Nodes running ZooKeeper aren’t running core Fusion services or Solr

In this cluster arrangement, the ZooKeeper cluster runs on nodes in the Fusion cluster on which core Fusion services and Solr aren’t running.

Each node in the Fusion cluster has Fusion and Solr installed. ZooKeeper runs on Fusion cluster nodes on which neither Fusion nor Solr is running.
How to set up a Fusion cluster

Perform the steps in the section Preliminary steps, and then perform these steps:

1. Edit `conf/fusion.properties` and remove `zookeeper` from the `group.default` list. This will make it so that ZooKeeper doesn't start when you start Fusion.

2. On each Fusion node, edit `default.zk.connect` in `fusion/5.0.x/conf/fusion.properties` to point to the ZooKeeper hosts:

   ```
   default.zk.connect=[ZK host 1]:2181,[ZK host 2]:2181,[ZK host 3]:2181
   ```

3. On each node, start ZooKeeper with `bin/zookeeper start`. Zookeeper should start without errors. If a ZooKeeper instance fails to start, check the log at `fusion/5.0.x/var/log/zookeeper/zookeeper.log`.

4. On each node, start the rest of Fusion using `bin/fusion start`.

5. Create an admin password and log in to Fusion at `http://FIRST_NODE_IP:8764`, where `FIRST_NODE_IP` is the IP address of your first Fusion node.

6. Verify the Solr cluster is healthy by looking at `http://ANY_NODE_IP:8983/solr/#/cloud`, where `ANY_NODE_IP` is the IP address of a Solr node. All of the nodes should appear green.

7. If necessary, prepare high availability by setting up a load balancer in front of Fusion so that it load balances between the Fusion UI URL's at `http://NODE_IP:8764`. 

---

**Key**

- **ZooKeeper cluster**
- **Solr cluster**
Consult your load balancer's documentation for instructions.
Install a Fusion Cluster (Windows)

This article describes how to install a Fusion cluster on multiple Windows nodes. Instructions are given for each of the cluster arrangements described in Deployment Types.

Preliminary steps

Before proceeding to one of the sections that follow, perform these steps:

How to prepare for setting up a Fusion cluster

1. Prepare your firewall so that the Fusion nodes can communicate with each other. The default ports list contains a list of all ports used by Fusion. From this list, it is important that the ZooKeeper ports, Apache Ignite ports, and the Spark ports (if you are using Spark) are open between the different nodes for cross-cluster communication.

   If you plan to use an external SolrCloud cluster and/or an external ZooKeeper cluster, then also prepare your firewall so that Fusion nodes can communicate with the SolrCloud and ZooKeeper nodes.

2. Fusion for Windows is distributed as a compressed archive file (.zip). Download the Fusion zip file for the latest version of Fusion to each node that will run Fusion, and move the file to where you would like Fusion to reside in your filesystem. It will appear as a compressed folder.

   Note
   To leverage the copies of Solr and/or ZooKeeper that are distributed with Fusion on nodes that will not run Fusion (as a simple means of obtaining compatible versions of the other software), also download the Fusion zip file to each of those nodes. Below, you will edit configuration files so that Fusion doesn’t run on those nodes.

3. Unpack the archive. In most cases, you need only right-click and choose “Extract all...”. If you don’t see this option, check that you have permissions to extract folders on your system.

   The resulting directory is named fusion\5.0.x. This directory is considered your Fusion home directory. You can rename it if you wish. See Directories, Files, and Ports for the contents of the Fusion home directory.

   Tip
   Fusion does not support installation under a directory whose name contains a space.

   Important
   In the sections that follow, for every step on multiple nodes, complete the step on all nodes before going to the next step. It is especially important that you don’t start Fusion on any node until the instructions say to do so.

In the steps below, the port numbers reflect default port numbers and one common choice (port 2181 for nodes in an external ZooKeeper cluster). Port numbers for your nodes might differ.

Nodes running core Fusion services and Solr also run ZooKeeper

In this cluster arrangement, a ZooKeeper cluster runs on the same nodes that run core Fusion services and Solr.
How to set up a Fusion cluster

Perform the steps in the section Preliminary steps, and then perform these steps:

1. Assign a number to each Fusion node, starting at 1. We refer to the number we assign to each node as the ZooKeeper myid.

2. On each Fusion node, create a `fusion\5.0.x\data\zookeeper` directory, and a file called `myid` in that directory. Edit the file and save the ZooKeeper myid assigned for this node as the only contents.

3. On each Fusion node, open the `fusion\5.0.x\conf\zookeeper\zoo.cfg` file in a text editor and add the following after the `clientPort` line (change the hostnames or IP addresses to the correct ones for your servers):

   ```
   server.1=[Hostname or IP for ZooKeeper with myid 1]:2888:3888
   server.2=[Hostname or IP for ZooKeeper with myid 2]:2888:3888
   server.3=[Hostname or IP for ZooKeeper with myid 3]:2888:3888
   ```

   For example:
+ NOTE: Don’t use localhost or 127.0.0.1 as the hostname/IP. Specify the hostname/IP that other nodes will use when communicating with the current node.

1. On each Fusion node, edit default.zk.connect in fusion\5.0.x\conf\fusion.properties to point to the ZooKeeper hosts:

```plaintext
default.zk.connect=[ZK host 1]:9983,[ZK host 2]:9983,[ZK host 3]:9983
```

2. On each node, start ZooKeeper with `bin\zookeeper start`. Zookeeper should start without errors. If a ZooKeeper instance fails to start, check the log at fusion\5.0.x\var\log\zookeeper\zookeeper.log.

3. On each node, start the rest of Fusion using `bin\fusion start`.

4. Create an admin password and log in to Fusion at http://FIRST_NODE_IP:8764, where FIRST_NODE_IP is the IP address of your first Fusion node.

5. Verify the Solr cluster is healthy by looking at http://ANY_NODE_IP:8983/solr/#/-cloud, where ANY_NODE_IP is the IP address of a Solr node. All of the nodes should appear green.

6. If necessary, prepare high availability by setting up a load balancer in front of Fusion so that it load balances between the Fusion UI URL’s at http://NODE_IP:8764.

Consult your load balancer’s documentation for instructions.

**Nodes running ZooKeeper aren’t running core Fusion services or Solr**

In this cluster arrangement, the ZooKeeper cluster runs on nodes in the Fusion cluster on which core Fusion services and Solr aren’t running.

Each node in the Fusion cluster has Fusion and Solr installed. ZooKeeper runs on Fusion cluster nodes on which neither Fusion nor Solr is running.
How to set up a Fusion cluster

Perform the steps in the section Preliminary steps, and then perform these steps:

1. Edit `conf\fusion.properties` and remove `zookeeper` from the `group.default` list. This will make it so that ZooKeeper doesn't start when you start Fusion.

2. On each Fusion node, edit `default.zk.connect` in `fusion\5.0.x\conf\fusion.properties` to point to the ZooKeeper hosts:

   ```
   default.zk.connect=[ZK host 1]:2181,[ZK host 2]:2181,[ZK host 3]:2181
   ```

3. On each node, start ZooKeeper with `bin\zookeeper start`. Zookeeper should start without errors. If a ZooKeeper instance fails to start, check the log at `fusion\5.0.x\var\log\zookeeper\zookeeper.log`.

4. On each node, start the rest of Fusion using `bin\fusion start`.

5. Create an admin password and log in to Fusion at `http://FIRST_NODE_IP:8764`, where `FIRST_NODE_IP` is the IP address of your first Fusion node.

6. Verify the Solr cluster is healthy by looking at `http://ANY_NODE_IP:8983/solr/#/cloud`, where `ANY_NODE_IP` is the IP address of a Solr node. All of the nodes should appear green.

7. If necessary, prepare high availability by setting up a load balancer in front of Fusion so that it load balances between the Fusion UI URL's at `http://NODE_IP:8764`.  

---

**Key**

- **ZooKeeper cluster**
- **Solr cluster**

---

**Fusion cluster**

<table>
<thead>
<tr>
<th>Node 1</th>
<th>Node 2</th>
<th>Node 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fusion services</strong></td>
<td><strong>Fusion services</strong></td>
<td><strong>Fusion services</strong></td>
</tr>
<tr>
<td>Core services</td>
<td>Core services</td>
<td>Core services</td>
</tr>
<tr>
<td>Connector services</td>
<td>Connector services</td>
<td>Connector services</td>
</tr>
<tr>
<td>Spark services *</td>
<td>Spark services *</td>
<td>Spark services *</td>
</tr>
<tr>
<td>Solr service</td>
<td>Solr service</td>
<td>Solr service</td>
</tr>
</tbody>
</table>

* Depends on features used

---

**Scale out nodes as needed.**
Consult your load balancer’s documentation for instructions.
Integrate Fusion with an Existing Solr Deployment

If you have already implemented Solr as a standalone instance or as a SolrCloud cluster, you can add Fusion to your existing Solr deployment (if the Solr version is supported) and import your Solr collections into Fusion. Each Fusion collection can import one Solr collection.

- If your existing Solr instance is running in SolrCloud mode, you can use Fusion’s UI to modify configuration files (such as schema.xml or solrconfig.xml) and create Solr collections.
- If your existing Solr instance is running in standalone mode, you can still connect it to Fusion. Fusion can send documents to a standalone Solr instance and query the instance. But you won’t be able to use Fusion’s UI to create Solr collections (Solr cores) or to modify Solr configuration files.

Prerequisites

- You have already installed Fusion.
- You have already installed Solr, which must meet these Solr requirements.
- You have already installed ZooKeeper, which must meet these ZooKeeper requirements.

<table>
<thead>
<tr>
<th>Note</th>
<th>We recommend that you create an external ZooKeeper cluster (external to both Fusion and SolrCloud).</th>
</tr>
</thead>
</table>

- Your Solr deployment must contain one or more collections (cores).
- In SolrCloud mode, Solr must be configured to use ZooKeeper.

Configure Fusion to use an existing Solr deployment

Use the Fusion UI or the Fusion API to integrate Fusion with an existing Solr deployment.

Use the Fusion UI

1. Create a Fusion search cluster:
   a. In the Fusion UI, navigate to System > Solr Clusters and click New Solr Cluster.
   b. Enter this information:
      - A cluster ID of your choice
      - Whether SolrCloud is enabled
      - The connect string (to tell Fusion how to connect to the SolrCloud cluster or Solr instance)
        - For SolrCloud, this is the ZooKeeper connect string.
        - For a standalone Solr instance, this is the URL of the Solr instance.
   c. Verify that the connection is working by clicking Cores in the new cluster and inspecting the contents.
2. Create a Fusion collection that points to your Solr cluster and collection:
   a. In the UI, navigate to Collections and click Add a Collection.
   b. Enter a name for the new collection.
c. Click **Advanced**.
d. Select your SolrCloud cluster or Solr instance from the dropdown.
e. Enter the name of the Solr collection to import.

**Use the Fusion API**

Use the Search Cluster API to create a Solr cluster.

Then use the Collections API to create and configure a collection.

**Sending Documents to Solr through Fusion**

You can use the Fusion connectors to crawl documents and index them to your existing Solr deployment.

1. Follow the steps above to create and configure a search cluster and a collection that points to Solr.
2. Define an index pipeline that ends with a Solr Indexer stage that sends the documents to Solr.
3. Use one of these methods to ingest your data:
   - In the collection that points to your Solr collection, define a datasource using the connector of choice.
   - Send prepared documents directly to the index pipeline for processing. See Importing Data with the REST API.
   - It’s also possible to use a different indexing process besides a connector, such as a script that sends documents through the index pipeline.

When documents are sent to Solr, a buffering SolrServer is used. Buffering the updates reduces the number of HTTP requests made from Fusion to Solr, which can significantly affect processing time. For example, when processing simple documents, you should always try to buffer as many documents as possible to increase throughput. When processing complex documents, you should use small batch sizes. You should only turn buffering off if you are using an older version of Solr and you want Fusion to catch and document indexing errors.

**Querying Solr via Fusion requests**

Indexed documents are stored in Solr indexes. You can query for these documents by using query pipelines. The query pipelines let you define your query parameters – such as how many records to return, the fields you’d like, how to structure facets, and so on. You also have the ability to add JavaScript to the response processing, and define landing pages or specific boost levels depending on the user’s query. See Query Pipelines.

If you prefer, you can also use the Solr API and SolrAdmin API to query Solr directly.
Troubleshoot When Installing Fusion

This topic explains how to troubleshoot difficulties that occur when installing or upgrading Fusion.

Fusion run script failures

Common problems that cause Fusion run scripts to fail:

- Wrong Java version
- Users have insufficient privileges for the installation directory.
- Java bin directory not in the PATH environment variable.
- Some Fusion services may already be running, or registered as running.
- Roaming IP address; try uncommenting this line in fusion/5.0.x/conf/fusion.properties:

  ```text
  default.address = 127.0.0.1
  ```

Check the Java version

Fusion runs on JDK 1.8. See System Requirements.

Fusion scripts use the environment variable JAVA_HOME. To check the setting of this variable, log in to the account used to run Fusion, and check that this variable is set to the proper value. On a linux, Mac, or other Unix system, use the following command:

```
echo $JAVA_HOME
```

On Windows, the command is:

```
echo %JAVA_HOME%
```

Fusion scripts execute both the java and javac commands. To check the Java version invoked by these commands, run the following commands from a shell or terminal window:

```
java -version
javac -version
```

Clear browser cache

If a previous version of Fusion was accessed in the browser with the same URL as that of the newly installed version of Fusion, then there may be old pages and/or cookies in the browser cache. A hard page refresh will clear old pages from the browser cache. If clearing the page cache doesn’t solve this problem, clear session cookies as well.
Stop/Clean up/Start

If the script `fusion/5.0.x/bin/fusion start` completes without reporting an error, but the Fusion UI displays a message that it can’t find Collections or Datasources, this may be due to Fusion services not being able to communicate properly (via ZooKeeper). This can happen with developer deployments running on a laptop if the network connection changes or is interrupted, especially when using the embedded ZooKeeper instance that is bundled with Fusion.

In this situation, you should stop Fusion, inspect the system processes and if necessary, manually terminate running processes and cleanup `.pid` files to bring the system back to a clean state, then start Fusion once again.

Although the Fusion run script `bin/fusion` provides a restart option, the restart option assumes a correctly functioning system and can’t always recover from system failure.

To stop Fusion:

Run the script `fusion/5.0.x/bin/fusion` with the argument `stop`:

```
$ cd /path/to/fusion/5.0.x
$ ./bin/fusion stop
Successfully stopped ui (process ID 41524)
Successfully stopped connectors (process ID 41328)
Successfully stopped api (process ID 41159)
Successfully stopped solr (process ID 41153)
Successfully stopped zookeeper (process ID 41151)
```

After stopping Fusion, you should make sure that no Fusion services are running. When the Fusion scripts start a Fusion service, they record the process id in a `.pid` file in the directory `fusion/5.0.x/var`. For a Fusion instance that is up and running, we see the following set of `.pid` files:

```
> find /path/to/fusion/5.0.x/var -name "*.pid" -print

fusion/var/api/api.pid
fusion/var/connectors/connectors.pid
fusion/var/solr/solr.pid
fusion/var/spark-master/spark-master.pid
fusion/var/spark-worker/spark-worker.pid
fusion/var/ui/ui.pid
fusion/var/zookeeper/zookeeper.pid
```

The above output shows the set of `.pid` created by a single Fusion instance running with embedded ZooKeeper and Solr.

But if no Fusion services are running, there shouldn’t be any `.pid` files. In the case that all services have been stopped, but there are still some `.pid` files found, these files should be deleted before starting Fusion.

**Inspect the log files**

If none of the above help, inspect the Fusion log files in directory `fusion/5.0.x/var/log`.

Log file names that start with "oom" indicate out-of-memory problems. You might need to increase the amount of memory allotted to that service. The amount of memory allotted to each kind of Fusion service is controlled by environment variables that are set in the `fusion.properties` file.
More Help and Support

• Lucidworks Fusion Knowledge Base: https://support.lucidworks.com/s/
• Contact Lucidworks: https://lucidworks.com/company/contact/
Start and Stop Fusion

This topic explains how to start and stop Fusion Server and its services using the scripts in the bin directory below the Fusion home directory:

- /opt/fusion/5.0.x/bin (Unix)
- C:\lucidworks\fusion\5.0.x\bin (Windows)
Command summary

You can control all Fusion services at once under the management of the Fusion agent, or you can control services individually.

How to control all services using the Fusion agent:

- **Unix:** /opt/fusion/5.0.x/bin/fusion <command>
- **Windows:** C:\lucidworks\fusion\5.0.x\bin\fusion.cmd <command>

How to control individual services:

- **Unix:** /opt/fusion/5.0.x/bin/<servicename> <command>
  
  For example: /opt/fusion/5.0.x/bin/proxy restart

- **Windows:** C:\lucidworks\fusion\5.0.x\bin\<servicename>.cmd <command>
  
  For example: C:\lucidworks\fusion\5.0.x\bin\proxy.cmd restart

<table>
<thead>
<tr>
<th>Tip</th>
<th>When starting services individually, start Zookeeper first.</th>
</tr>
</thead>
</table>

The commands below can be issued to the fusion/fusion.cmd script to issue the command to all services in the correct sequence, or they can be issued to an individual service.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>start</strong></td>
<td>Start one or all Fusion services.</td>
</tr>
<tr>
<td><strong>status</strong></td>
<td>Display the status of one or all Fusion services.</td>
</tr>
<tr>
<td><strong>restart</strong></td>
<td>Restart one or all Fusion services.</td>
</tr>
<tr>
<td><strong>stop</strong></td>
<td>Stop one or all Fusion services.</td>
</tr>
<tr>
<td><strong>run</strong></td>
<td>Start one or all Fusion services in the foreground.</td>
</tr>
<tr>
<td><strong>run-in-shell</strong> (Unix only)</td>
<td>Start an individual service using Bash’s exec function, which allows the service to assume the shell process’s PID. See Run Fusion in shell mode below.</td>
</tr>
</tbody>
</table>
Define groups of services

The `fusion.properties` file includes the property definition `group.default`. This property defines the Fusion services to start and stop by default (if no property is named in the start or stop command).

The default list of services out-of-the-box is also the *minimum* set of services, with the exception of the `log-shipper` service, which you can remove if you don't use it.

Here is the `group.default` definition in `fusion.properties`:

```properties
group.default = zookeeper, solr, api, connectors-classic, connectors-rpc, proxy, webapps, admin-ui, log-shipper
```

With the exception of the `log-shipper` service, these are all required services. Even if only using RPC connectors, the `connectors-classic` service is required. The `log-shipper` service is required to use the Log Shipper.

How to modify the default list of services

Edit the `group.default` property, for example, to include Spark and SQL related services:

```properties
group.default = zookeeper, solr, api, connectors-classic, connectors-rpc, proxy, webapps, admin-ui, log-shipper, spark-master, spark-worker, sql
```

How to define other lists of services (Unix)

You can define other lists of services by defining other `group` properties. For example, define this group property to start and stop services for Spark and SQL together:

```properties
group.spark-only = spark-master, spark-worker, sql
```

Define this group property to start and stop services for classic and RPC connectors together:

```properties
group.connectors = connectors-classic, connectors-rpc
```
Start and stop Fusion on Unix.

**Start Fusion**

All Fusion start scripts must be executed by a user who has permissions to read and write to the directories where Fusion is installed. These scripts don't need to be run as root (or sudo), nor should they be. Use a suitable user, or create a new one, and then ensure that it owns the directory where Fusion resides, (for example, C:\lucidworks).

Give the commands that follow from the directory `fusion/5.0.x/bin`.

**Start required services**

Start the required services that are defined in the `group.default` property.

How to start all required services

```bash
./fusion start
```

**Tip**

This is equivalent to `./fusion start default`. You can omit the group name `default`.

**Start a group of services**

You can start a group of services together. Reference the property in `fusion.properties` that defines the group.

Examples of when this is useful are:

- **Spark and SQL** – The `spark-master`, `spark-worker`, and `sql` services are interdependent and should be started and stopped together.

  ```bash
  ./fusion start spark-master spark-worker sql
  ```

- **Classic and RPC connectors** – RPC connectors require both the `connectors-classic` and `connectors-rpc` services to be running.

  ```bash
  ./fusion start connectors-classic connectors-rpc
  ```

**Start services individually**

You can start services individually.

How to start services individually

- Fusion UI service:

  ```bash
  ./admin-ui start
  ```
• API services:

  ./api start

• Classic Connectors services:

  ./connectors-classic start

• RPC Connectors services:

  ./connectors-rpc start

• Log shipper service:

  ./log-shipper start

• Proxy:

  ./proxy start

• Solr:

  ./solr start

• Spark Master:

  ./spark-master start

• Spark Worker:

  ./spark-worker start

• SQL service:

  ./sql start

• Web Apps:

  ./webapps start

• ZooKeeper:

  ./zookeeper start

For information about default ports, see Default Ports.

**Run Fusion in the foreground**

To run Fusion or any of its services in the foreground, use the *run* command-line argument in place of *start*.

**Run Fusion in shell mode**

To start any of Fusion’s services using Bash’s *exec* function, which allows the service to assume the shell process’s PID, use the *run-in-shell* command-line argument in place of *start* or *run*. The *run-in-shell* argument can only be used to start one service at a time.

Examples
./fusion run-in-shell zookeeper

or

./zookeeper run-in-shell

Shell mode is particularly useful in containerized environments, which generally assume that only one process runs per container and that process is “process 0”, that is, the initial process invoked within the container, not a separate spawned process.

Stop Fusion

How to stop Fusion services

To stop Fusion or any of its services, use the command above with the stop command-line argument in place of start, for example:

./solr stop

Using systemd to manage processes

On Red Hat Enterprise Linux, CentOS 7 and newer, and Ubuntu 15.04 LTS and newer, we support using the operating system-provided systemd for process management.

Launching Fusion at system start

You can configure systemd to launch Fusion when your system starts.

How to launch Fusion at system start:

1. Change your working directory to Fusion’s systemd directory, for example:

   cd {fusion-parent-unix}/fusion/{version}{sub-version}/init/systemd

2. Edit fusion.service to provide correct values for the FUSION_HOME and JAVA_HOME environment variables.

3. Stop Fusion if it is already running:

   {fusion-parent-unix}/fusion/{version}{sub-version}/bin/fusion stop

4. Create the systemd management file, which launches Fusion under systemd management:

   sudo bash install.sh

Starting and stopping Fusion

You can use the systemctl command to start and stop Fusion:
Log files for Fusion services are found in directories under `fusion/5.0.x/var/log`.

**Using Ubuntu Upstart to manage processes**

Under Ubuntu 12.04 LTS through Ubuntu 14.10, we support using Upstart for process management. This requires Fusion to be installed in the `/opt/lucidworks/` directory.

To configure upstart, run the following commands:

```bash
$ cd /opt/lucidworks/fusion/{version}/init/upstart
$ sudo bash install.sh
```

If this complains with `no JAVA_HOME set`, replace `sudo` with `sudo -E`. Then you can use the `service` command to control the server:

```bash
$ sudo service fusion-solr start
$ sudo service fusion-api start
$ sudo service fusion-connectors start
$ sudo service fusion-ui start
```

and similarly use `stop` and `status`.

Log files for Fusion services are found in directories under `fusion/5.0.x/var/log`. 
Windows

Start and stop Fusion on Windows.

Start Fusion

All Fusion start scripts must be executed by a user who has permissions to read and write to the directories where Fusion is installed. Ensure that the user owns the directory where Fusion resides (for example, \lucidworks).

Give the commands that follow from the directory fusion5.0.x\bin.

Start required services

How to start all required Fusion services as Java processes

```
fusion.cmd start
```

How to start all required Fusion services as Windows services

```
start-services.cmd
```

Start services individually

How to start specific services as Java processes

- UI service:
  ```
  admin-ui.cmd start
  ```

- API services:
  ```
  api.cmd start
  ```

- Classic Connectors services:
  ```
  connectors-classic.cmd start
  ```

- RPC Connectors services:
  ```
  connectors-rpc.cmd start
  ```

- Log shipper service:
  ```
  log-shipper.cmd start
  ```

- Proxy:
  ```
  proxy.cmd start
  ```

- Solr:
  ```
  solr.cmd start
  ```
• Spark Master:
  
  `spark-master.cmd start`

• Spark Worker:

  `spark-worker.cmd start`

• SQL service:

  `sql.cmd start`

• Web Apps:

  `webapps.cmd start`

• ZooKeeper:

  `zookeeper.cmd start`

For information about default ports, see Default Ports.

**Run Fusion in the foreground**

To run Fusion or any of its services in the foreground, use the `run` command-line argument in place of `start`, for example:

`connectors.cmd run`

**Stop Fusion**

How to stop all Fusion services

- `fusion.cmd stop` (Stop all Fusion services, if they are running as Java processes)
- `stop-services.cmd` (Stop all Fusion services, if they are running as Windows services)

How to stop a specific service running as a Java process

To stop a specific Fusion service that is running as a Java process, use the command above with the `stop` command-line argument in place of `start`, for example:

`connectors.cmd stop`

**Run Fusion with a service account**

This example assumes the following:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account</td>
<td>FUSION_SVC</td>
</tr>
<tr>
<td>Domain</td>
<td>qe</td>
</tr>
<tr>
<td>Installation directory</td>
<td>C:\fusion&lt;version&gt;</td>
</tr>
</tbody>
</table>
1. As an administrator, create the service account, and install it to the server you want to use for Fusion:

```powershell
C:\Users\Administrator> New-ADServiceAccount -Name "FUSION_SVC" -RestrictToSingleComputer
C:\Users\Administrator> Add-ADComputerServiceAccount -Identity EC2AMAZ-79FD9JL -ServiceAccount "FUSION_SVC"
C:\Users\Administrator> Install-ADServiceAccount -Identity "FUSION_SVC"
C:\Users\Administrator> Test-ADServiceAccount "FUSION_SVC"
```

2. Run `install-services.cmd` as a local administrator:

```powershell
C:\Users\Administrator> C:\fusion\4.2.2\bin\install-services.cmd
ECHO is off.
Thank you for choosing

```

3. Grant the service account full access to Fusion:

```powershell
C:\Users\Administrator>$path = "C:\fusion"
$Acl = Get-Acl "$path"
$permission = "qe\FUSION_SVC"", "FullControl", "ContainerInherit, ObjectInherit", "None", "Allow"
$Acl.SetAccessRule($accessRule)
Set-Acl "$path" $Acl
Get-Acl "$path" | fl
```

4. Modify the **Lucidworks Fusion** service to use the service account.
   a. Open **Administrative Tools > Services** on your Windows server.
   b. Select the **Lucidworks Fusion** service.
   c. Open the **Properties > Log On** dialog.
   d. Change the service user account to the **FUSION_SVC** user account.
| Note           | Only the account name is required. The password is managed by Windows. |
Access Fusion after startup

After Fusion services have started, you can open the Fusion UI in a web browser at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).

The first time you access Fusion, you must set the password for the user admin and agree to the Fusion Licensing Agreement (which contains terms of service). This is followed by an optional registration step. After this, Fusion displays the Fusion launcher (the page from which you can open apps).

If you can't access the system, see the Troubleshooting topic. Checking System State shows how to inspect Fusion services.
Fusion Server Directories, Files, and Ports
## Configuration files

Fusion configuration files are stored in `fusion/5.0.x/conf/` (on Unix or macOS) or `fusion\5.0.x\conf\` (on Windows). The contents of this directory are as follows:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fusion.properties</td>
<td>Fusion's main configuration file, which defines the common environment variables used by the Fusion run scripts. Many of the values in this file can be set using environment variables, enabling you to set them using <code>systemd</code>, Docker, and so on. Default values are also provided. For example, in <code>api.port = ${API_PORT:-8765}</code>, the value is 8765 unless <code>API_PORT</code> is defined. <strong>NOTE</strong> <code>ZOOKEEPER_PORT</code> cannot not be used, and value of <code>zookeeper.port</code> in <code>fusion.properties</code> must be the same as the value of <code>clientPort</code> in <code>conf/zookeeper/zoo.cfg</code>.</td>
</tr>
<tr>
<td>hive-site.xml</td>
<td>Configuration for Fusion's Serializer/Deserializer (SerDe) for Hive.</td>
</tr>
<tr>
<td>zookeeper/commons-logging.properties zookeeper/zoo.cfg</td>
<td>ZooKeeper configuration files.</td>
</tr>
</tbody>
</table>
Port configuration

Fusion services run in their own JVM and listen for requests on a number of ports. Environment variables, set in a common configuration file, are used to specify the port a service uses. To change the port(s) a service uses, you must change the settings in the configuration file.

Default ports

This table lists the default port numbers used by Fusion processes. Port settings are defined in the `fusion.properties` file in `fusion/5.0.x/conf/` (on Unix or macOS) or `fusion\5.0.x\conf\` (on Windows).

<table>
<thead>
<tr>
<th>Port</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>8091</td>
<td>Fusion agent</td>
</tr>
<tr>
<td>8763</td>
<td>Fusion UI service (use port 8764 to access the Fusion UI)</td>
</tr>
<tr>
<td>8764</td>
<td>Fusion proxy</td>
</tr>
<tr>
<td></td>
<td>This service includes the Fusion Authorization Proxy.</td>
</tr>
<tr>
<td>8765</td>
<td>Fusion API Services</td>
</tr>
<tr>
<td>8766</td>
<td>Spark Master</td>
</tr>
<tr>
<td>8769</td>
<td>Spark Worker</td>
</tr>
<tr>
<td>8771</td>
<td>Connectors RPC Service</td>
</tr>
<tr>
<td></td>
<td>This service can distribute connector jobs to as many Fusion nodes as you</td>
</tr>
<tr>
<td></td>
<td>want. It uses HTTP/2 and has an SDK that you can use to build your own</td>
</tr>
<tr>
<td></td>
<td>connectors.</td>
</tr>
<tr>
<td>8780</td>
<td>Web Apps</td>
</tr>
<tr>
<td></td>
<td>This service delivers the UIs of Fusion apps.</td>
</tr>
<tr>
<td>8781</td>
<td>Log shipper</td>
</tr>
<tr>
<td></td>
<td>Monitoring port that agent uses to check the health of the log shipper</td>
</tr>
<tr>
<td></td>
<td>process. This port does not need to be accessible from other nodes.</td>
</tr>
<tr>
<td>Port</td>
<td>Service</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>8983</td>
<td>Solr</td>
</tr>
<tr>
<td></td>
<td>This is the embedded Solr instance included in the Fusion distribution.</td>
</tr>
<tr>
<td>8984</td>
<td>Connectors Classic Service</td>
</tr>
<tr>
<td></td>
<td>This service runs nondistributed connector jobs. It uses HTTP/1.1 and has no SDK.</td>
</tr>
<tr>
<td>9983</td>
<td>ZooKeeper</td>
</tr>
<tr>
<td></td>
<td>The embedded ZooKeeper used by Fusion services.</td>
</tr>
</tbody>
</table>

a | Important

| The ZooKeeper port is also defined in the configuration file for the embedded ZooKeeper, 'fusion/5.0.x/conf/zookeeper/zoo.cfg' (on Unix or macOS) or 'fusion\5.0.x\conf\zookeeper\zoo.cfg' (on Windows). Look for 'clientPort'. If you run Fusion with the embedded ZooKeeper, remember to change the port number in both places. |

47100-48099

Apache Ignite TCP communication port range (used by the API, Connectors Classic, Connectors RPC, and Proxy services)

48100-48199

Apache Ignite shared memory port range (used by the API, Connectors Classic, Connectors RPC, and Proxy services)

49200-49299

Apache Ignite discovery port range (used by API, Connectors Classic, Connectors RPC, and Proxy services)

### Jetty ports

Jetty is used to run the Admin UI, API, Connectors Classic, Proxy, Solr, and Web Apps services. For each of these services, Jetty runs the service on the assigned port and listens on a second port for shutdown requests. Therefore, `fusion.properties` defines pairs of ports for components running on Jetty, such as:

```properties
api.port = ${API_PORT:-8765}
api.stopPort = 7765
```
## Spark ports

This table lists the default port numbers used by Spark processes in Fusion.

<table>
<thead>
<tr>
<th>Port number</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>4040</td>
<td>SparkContext web UI</td>
</tr>
<tr>
<td>7337</td>
<td>Shuffle port for Apache Spark worker</td>
</tr>
<tr>
<td>8767</td>
<td>Spark master web UI</td>
</tr>
<tr>
<td>8770</td>
<td>Spark worker web UI</td>
</tr>
<tr>
<td>8766</td>
<td>Spark master listening port</td>
</tr>
<tr>
<td>8769</td>
<td>Spark worker listening port</td>
</tr>
<tr>
<td>8772 (spark.driver.port)</td>
<td>Spark driver listening port</td>
</tr>
<tr>
<td>8788 (spark.blockManager.port)</td>
<td>Spark BlockManager port</td>
</tr>
</tbody>
</table>

If a port is not available, Spark uses the next available port by adding 1 to the assigned port number. For example, if 4040 is not available, Spark uses 4041 (if available, or 4042, and so forth).

Ensure that the ports in the above table are accessible, as well as a range of up to 16 subsequent ports. For example, open ports 8772 through 8787, and 8788 through 8804, because a single node can have more than one Spark driver and Spark BlockManager.
Directories

The directory where the Fusion files go for a specific version of Fusion is the *Fusion home directory*. The Fusion home directory is a version-numbered directory (for example, 4.1.0) below the directory `fusion`. This installation strategy lets you install multiple versions of Fusion and switch between them.

The directories found in the Fusion home directory in `\fusion/5.0.x/` (on Unix or macOS) or `fusion\5.0\` (on Windows) are:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apps</td>
<td>Fusion components 3rd-party distributions used by Fusion, including jar files and plugins</td>
</tr>
<tr>
<td>bin</td>
<td>Master script to run Fusion, and per-component run scripts</td>
</tr>
<tr>
<td>conf</td>
<td>Configuration files for Fusion and ZooKeeper that contain parameters settings tuned for common use cases</td>
</tr>
<tr>
<td>data</td>
<td>Default location of data stores used by Fusion apps</td>
</tr>
<tr>
<td>docs</td>
<td>License information</td>
</tr>
<tr>
<td>examples</td>
<td>Fusion signals example</td>
</tr>
<tr>
<td>init</td>
<td><em>systemd</em> and <em>upstart</em> scripts and configurations for Linux</td>
</tr>
<tr>
<td>scripts</td>
<td>Developer utilities, including diagnostic scripts, for Linux and Windows. See <em>scripts/diag/linux/README</em> and <em>scripts/diag/win64/README.txt</em> for details.</td>
</tr>
<tr>
<td>var</td>
<td>Log files and system files created by Fusion components, as well as .pid files for each running process</td>
</tr>
</tbody>
</table>

Symbolic links on UNIX

To simplify access to the latest version of Fusion and to files in the `bin`, `conf`, and `var` directories, Fusion creates a symbolic link `latest` to the latest version and symbolic links `bin`, `conf`, and `var` to `latest/bin`, `latest/conf`, and `latest/var` respectively.

For example, if `latest` is 4.1.1, then instead of entering this command to change to the `bin` directory:

```
$ cd /path/to/fusion/4.1.1/bin
```

You could just type:

```
$ cd /path/to/fusion/bin
```
To avoid possible confusion in the documentation, we spell out the path below the fusion directory.

From the fusion directory, you can view the symbolic links by typing:

```bash
$ find . -maxdepth 1 -type l -ls
```

To change the version of Fusion to which the symbolic links refer, unlink and relink latest. For example:

```bash
$ cd /path/to/fusion
$ unlink latest
$ link -s 4.1.1 latest
```
Log files

Log files are found in directories under `fusion/5.0.x/var/log/` (on Unix or macOS) or `fusion\5.0.x\var\log\` (on Windows).

Because the Fusion components run in separate JVMs, each component has its own set of log files and files that monitor all garbage-collection events for that process.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin-ui,webapps</td>
<td>Fusion UI messages. Messages are logged to <code>jetty-&lt;date&gt;.stderrout.log</code>.</td>
</tr>
<tr>
<td>agent</td>
<td>Fusion agent logging and error messages</td>
</tr>
<tr>
<td>api</td>
<td>Fusion REST API services logging and error messages. This log shows the result of service requests submitted to the REST API directly via HTTP and indirectly via the Fusion UI.</td>
</tr>
<tr>
<td>connectors</td>
<td>Fusion connector services logging and error messages. Fusion index pipeline logging stages write to this file.</td>
</tr>
<tr>
<td>log-shipper</td>
<td>See Configure Fusion logging</td>
</tr>
<tr>
<td>proxy</td>
<td>Messages from the Fusion proxy, responsible for authentication and HTTP load balancing.</td>
</tr>
<tr>
<td>solr</td>
<td>Messages from Solr</td>
</tr>
<tr>
<td>spark-master</td>
<td>Spark-master logs</td>
</tr>
<tr>
<td>spark-worker</td>
<td>Spark-worker logs</td>
</tr>
<tr>
<td>sql</td>
<td>SQL logs</td>
</tr>
<tr>
<td>zookeeper</td>
<td>ZooKeeper messages</td>
</tr>
</tbody>
</table>

Every component logs all messages to a log file named `<component>.log`. For example, the full path to the log file for the connectors services is `fusion/5.0.x/var/log/connectors/connectors.log` (on Unix or macOS) or `fusion\5.0.x\var\log\connectors\connectors.log` (on Windows).

In addition to component log files, every component maintains a set of garbage-collection log files that are used for resource tuning. The garbage-collection log files are named `gc_<YYYYMMDD>_<PID>.log.<CT>`. In addition, the current garbage-collection log file has suffix `.current`.

The Fusion API, Fusion UI, Connectors Classic, Proxy, Web Apps, and Solr services all run inside a Jetty server. The Jetty server logs are also written to each component’s log file directory. The Jetty server logs are named:

- `jetty-YYYY_MM_DD.request.log`
- `jetty-YYYY_MM_DD.stderrout.log`

Configure Fusion logging

Fusion uses the Apache Log4j 2 logging framework with Jetty to log each of the Fusion components.
Logging is configured with an XML configuration file named `log4j2.xml`. Log levels, frequencies, and log rotation policy can be configured by changing these configuration files:

<table>
<thead>
<tr>
<th>Service</th>
<th>Configuration File</th>
</tr>
</thead>
<tbody>
<tr>
<td>API service</td>
<td><code>fusion/5.0.x/conf/api-log4j2.xml</code></td>
</tr>
<tr>
<td>Connectors</td>
<td><code>fusion/5.0.x/conf/connectors-classic-log4j2.xml</code></td>
</tr>
<tr>
<td></td>
<td><code>fusion/5.0.x/conf/connectors-rpc-log4j2.xml</code></td>
</tr>
<tr>
<td>Solr</td>
<td><code>fusion/5.0.x/conf/solr-log4j2.xml</code></td>
</tr>
<tr>
<td>Spark</td>
<td><code>fusion/5.0.x/conf/spark-driver-launcher-log4j2.xml</code></td>
</tr>
<tr>
<td></td>
<td><code>fusion/5.0.x/conf/spark-driver-log4j2.xml</code></td>
</tr>
<tr>
<td></td>
<td><code>fusion/5.0.x/conf/spark-driver-scripted-log4j2.xml</code></td>
</tr>
<tr>
<td></td>
<td><code>fusion/5.0.x/conf/spark-executor-log4j2.xml</code></td>
</tr>
<tr>
<td></td>
<td><code>fusion/5.0.x/conf/spark-master-log4j2.xml</code></td>
</tr>
<tr>
<td></td>
<td><code>fusion/5.0.x/conf/spark-worker-log4j2.xml</code></td>
</tr>
<tr>
<td>SQL</td>
<td><code>fusion/5.0.x/conf/sql-log4j2.xml</code></td>
</tr>
<tr>
<td>ZooKeeper</td>
<td><code>fusion/5.0.x/conf/zookeeper-log4j2.xml</code></td>
</tr>
<tr>
<td>Other Fusion services</td>
<td><code>fusion/5.0.x/conf/proxy-log4j2.xml</code></td>
</tr>
</tbody>
</table>

The [Log4j2 Configuration](#) guide provides documentation and examples of all logging configuration options.

The Fusion log shipper parses Fusion Java, HTTP, and garbage collector logs and sends them to a system collection. This system collection is used for dashboards in the Fusion UI.

Log shipping is enabled by default. You can disable log shipping, adjust which logs are parsed, or send log files to an external Solr instance or cluster, or to a custom collection name. For the configuration and details, see `fusion/5.0.x/conf/fusion.properties` (on Unix or macOS) or `fusion\5.0.x\conf\fusion.properties` (on Windows).

### View and analyze log files

Fusion has several features that make analysis of log files easier:

- **View log file dashboards** – The Service Logs, Access Logs, and Combined Logs dashboards provide graphical user interfaces for viewing and analyzing log files.
To open the default dashboard from the Fusion workspace, click Analytics > Dashboards or System > Log Viewer.

- **Assign Fusion request IDs** – To make it easier to follow requests through the Fusion system, you can assign Fusion request IDs. If you don’t, Fusion assigns request IDs automatically.

- **Filter log file dashboards by the Fusion request ID** – In the log file dashboards, you can filter by Fusion request ID.
• Click through from API errors in the Fusion UI to the Service Logs dashboard filtered by the Fusion request ID of the request that resulted in the error.
On-Prem Licensing

Fusion Server requires a valid license. Depending on the details of your contract, your license may also enable optional connectors or Fusion AI.

When you download Fusion Server, it comes with a 30-day trial license. Contact Lucidworks to obtain a permanent license.

Fusion Server provides a license management UI and a license API for installing and managing licenses. When you upload a license, Fusion Server stores it in ZooKeeper, so you only need to upload it to one node per cluster.

The Fusion UI notifies you when your trial license is about to expire. When your license has expired, Fusion Server accepts no configuration changes until you upload a valid license.
Uploading a license using the UI

1. Log in to the Fusion UI.
2. In the upper right, open the profile menu and select **License Details**:

The License Details window appears:

![License Details Window](image)
3. Click **Choose License** and select your license file.

4. Click **Upload**.
Upgrade Fusion

When you have a Fusion-based search application running, at some point it might be necessary to upgrade to a later version of Fusion. We provide a migrator tool to simplify the upgrade process.

Tip

See the release history to find out what’s new, including which versions of Solr, Spark, and ZooKeeper are bundled with each Fusion release.

The migrator transfers over most of the objects that make up your search application, all configurations and customizations for your application, and all data in collections in the application.

Note

In some cases, manual steps are required for objects that the migrator can’t handle automatically. We give you instructions and guidance about what might be required. You should also review the log of the upgrade in /opt/fusion/x.y.z/var/upgrade/tmp/migrator.log (on Unix) or C:\lucidworks\var\fusion\x.y.z\upgrade\tmp\migrator.log (on Windows). The x.y.z directory is for the Fusion version that you are migrating from.
Key points

Following are some key points about upgrading Fusion:

- **Migration involves down time** – The upgrade process involves multiple starts and stops of Fusion services. Please plan accordingly, especially in terms of disabling external load balancers or monitors that might react adversely to the starts and stops.

- **Current deployment is preserved** – Upgrades preserve the current Fusion deployment, copying information over from the current deployment to the new one. This provides a rapid roll-back option if you encounter problems during the upgrade process.

- **If the upgrade fails** – If an upgrade fails, there’s a procedure for dealing with that.
Supported upgrade sequences

Only specific version-to-version upgrade sequences are supported. Some upgrades require multiple steps.

These upgrade sequences are supported.

Upgrades to the current version

- **3.1.x to 4.2.y** – From any 3.1.x version to 4.2.5 (one step, using the migrator)
- **4.0.x to 4.2.y** – From any 4.0.x version to 4.2.5 (one step, using the migrator)
- **4.1.x to 4.2.y** – From any 4.1.x version to 4.2.5 (one step, using the migrator)

For links to these procedures, see Per-version instruction sets.

Upgrades to prior versions

Manual:

- **2.4.5 to 3.0.0** (one step)
  
  For more information, see Upgrade Fusion 2.1.4 or 2.4.x to Fusion 3.0.0.

- **3.0.0 to 3.0.1** (one step)
  
  For more information, see Upgrade Fusion 3.0.0 to Fusion 3.0.1.

Using the migrator:

- **3.0.x to 3.1.0** – From 3.0.0 or 3.0.1 directly to 3.1.5 (one step)
  
  For more information, see Upgrade Fusion 3.0.x to 3.1.y.

- **3.1.x to 3.1.y** – From 3.1.0, 3.1.1, 3.1.2, 3.1.3, or 3.1.4 to 3.1.5 (one step)
  
  For more information, see Upgrade Fusion 3.1.x to 3.1.y.

- **3.1.x to 4.0.y** – From 3.1.5 directly to 4.0.2 (one step)
  
  For more information, see Upgrade Fusion 3.1.x to 4.0.y.

- **4.0.x to 4.0.y** – From 4.0.0 or 4.0.1 to 4.0.2 (one step)
  
  For more information, see Upgrade Fusion Server 4.0.x to 4.0.y.

- **3.1.x to 4.1.y** – From any 3.1.x version to 4.1.3 (one step, using the migrator)
  
  For more information, see Upgrade Fusion Server 3.1.x to 4.1.y.

- **4.0.x to 4.1.y** – From 4.0.2 to 4.1.3 (one step, using the migrator)
For more information, see Upgrade Fusion Server 4.0.x to 4.1.y.

• **4.1.x to 4.1.y** – From 4.1.0 to 4.1.3 (one step, using the migrator)

For more information, see Upgrade Fusion Server 4.1.x to 4.1.y.

**Example**

For example, to upgrade from Fusion 3.0.1 to Fusion Server 4.2.5, you would perform the following upgrades (both of them using the migrator):

1. Upgrade from Fusion 3.0.1 to Fusion 3.1.5
2. Upgrade from Fusion 3.1.5 to Fusion Server 4.2.5
Per-version instruction sets

To upgrade to a later version of Fusion from an existing installation requires transferring over all configurations and data from your existing Fusion installation to the new version.

How to upgrade from Fusion 3.1.x to Fusion Server 4.2.y

Perform the steps in this article:

**Upgrade from Fusion Server 3.1.x to 4.2.y** – Run a migrator to upgrade from Fusion Server 3.1.x to 4.2.y.

How to upgrade from Fusion 4.0.x to Fusion Server 4.2.y

Perform the steps in this article:

**Upgrade from Fusion Server 4.0.x to 4.2.y** – Run a migrator to upgrade from Fusion Server 4.0.x to 4.2.y.

How to upgrade from Fusion 4.1.x to Fusion Server 4.2.y

Perform the steps in this article:

**Upgrade from Fusion Server 4.1.x to 4.2.y** – Run a migrator to upgrade from Fusion Server 4.1.x to 4.2.y.

How to upgrade from Fusion 4.2.x to Fusion Server 4.2.y

Perform the steps in this article:

**Upgrade from Fusion Server 4.2.x to 4.2.y** – Run a migrator to upgrade from Fusion Server 4.2.x to 4.2.y.
Upgrade Fusion Server 3.1.x to 4.2.y

Introduction

This article describes how to perform the following upgrade:

- From version: Fusion 3.1.x
- To version: Fusion 4.2.y

| Important | Only specific version-to-version upgrade sequences are supported. Some upgrades require multiple steps. For more information, see the supported upgrade sequences. |

For Fusion 3.1 and later releases, a migrator is available for upgrading Fusion.

During the upgrade process, the migrator uses a properties file. After downloading and installing the migrator, the properties files is in the /opt/lucidworks/fusion/3.1.x/var/upgrade directory (on Unix or MacOS) or the C:\lucidworks\fusion\3.1.x\var\upgrade directory (on Windows). The file names reference the versions you are upgrading from and to. For example:

- To upgrade 3.1.5 to 4.2.5, the migrator uses the 3.1.x-4.2.x.properties file.

Migration entails down time and multiple starts and stops of Fusion services. Plan accordingly, especially in terms of disabling external load balancers or monitors that might react adversely to the starts and stops.

Download the latest migrator immediately before upgrading. This helps ensure that the upgrade goes smoothly.

| Important | The newer Fusion instance must be newly untarred and never started. |

About the upgrade

This section describes how connectors, object migrations, and signals are migrated during an upgrade.

Connectors

In Fusion 3.1.0 and above, only a subset of connectors are included by default.

The migrator detects which connectors were used in the older version of Fusion, and installs them automatically in Fusion 4.2.y. This step requires an Internet connection. If no connection is available, then you must download connectors at http://lucidworks.com/connectors/ and install them as bootstrap plugins.

If a connector to be upgraded wasn’t available during the upgrade, then a message in /opt/lucidworks/fusion/3.1.x/var/upgrade/tmp/migrator.log (on Unix) or C:\lucidworks\fusion\3.1.x\var\upgrade\tmp\migrator.log (on Windows) indicates this.

Only datasources for connectors that are supported in the new Fusion version are upgraded. Datasources for custom connectors aren’t upgraded.
If no Internet connection is available

If no Internet connection is available during an upgrade, the migrator can’t automatically download the connectors it needs. Using the Fusion UI or API later to install the connectors also might not be an option.

In this case, download the connector zip files for all existing connectors and any connectors that you are adding from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new deployment (on all Fusion nodes). Do so at the time indicated in the procedures that follow.

Adding connectors during an upgrade

You can add connectors during an upgrade (that is, add connectors that aren't in the old deployment).

Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new version (on all Fusion nodes).

Object migrations and transformations

The migrator upgrades all Fusion 3.1 object types:

- Collections
- Index pipelines
- Query pipelines
- Search cluster configurations
- Schedules
- Aggregations
- Datasources
- Dashboards
- Parsing configurations
- Object groups
- Links
- Tasks
- Jobs
- Spark objects

The migrator adds these Fusion Server object types:

- Apps
- Appkit apps
- Index profiles
- Query profiles
- Blobs
Important

In Fusion Server 4.0 and later, most objects exist in the context of apps. When you upgrade from Fusion 3.1.x to Fusion Server 4.2.y, the migrator creates the app `default` and places object in that app or links them to it, as needed. Some objects are not linked to apps. You can explore objects in Object Explorer.

---

### Noteworthy transformations

While migrating objects, the migrator makes these transformations:

- It adds the new `log-shipper` service to `fusion.properties`.
- It removes obsolete log-cleanup tasks.
- It converts the format of job history records.

### Migration of index profiles and query profiles

In Fusion Server 4.0, index profiles and query profiles are objects. They have capabilities that exceed those of index profiles and query profiles in prior releases.

- **Prior to Fusion 4.0** – A single index profile could reference multiple index pipelines. A single query profile could reference multiple query pipelines.
- **In Fusion 4.0 and later** – A single index profile can reference a single index pipeline. A single query profile can reference a single query pipeline.

During migration from 3.1.x to 4.0.y, the migrator upgrades index profiles and query profiles as follows:

- **Index profiles:**
  - If a 3.1.x index profile contains a reference to the index pipeline `default`, then the migrated profile retains that single reference.
  - If a 3.1.x index profile doesn't contain a reference to the index pipeline `default`, then the migrated profile references an index pipeline that has the same name as the index profile.

- **Query profiles:**
  - If a 3.1.x query profile contains a reference to the query pipeline `default`, then the migrated profile retains that single reference.
  - If a 3.1.x query profile doesn't contain a reference to the query pipeline `default`, then the migrated profile references a query pipeline that has the same name as the query profile.

After migration, you might need to adjust the pipeline references of index profiles and query profiles by hand, and/or create new index profiles and query profiles.

### Access control migration

The migrator upgrades all access control configurations:

- **Security realms** – Security realms don’t require adjustments after migration.
- **Roles** – System-created roles don't require adjustments after migration. User-created roles probably require adjustments after migration.
• **Users** – System-created users don’t require adjustments after migration. User-created users probably require adjustments after migration.

### Migrations that shouldn’t require adjustments

This section describes fully automatic parts of migration. You shouldn’t have to adjust these items after migration.

The migrator performs these migration tasks:

- It migrates encrypted passwords, so users can log in with the same credentials they used on the older version of Fusion or Fusion Server.
- It adds the user `webapps-system-account`. Fusion Server uses this account. You don’t need to do anything with it.
- It adds these permissions to the role `developer`:

```
GET:/license
GET,POST,PUT:/appkit
GET,POST,PUT:/apps/**
```

- Updates these permissions for the role `search`:

```
POST:/apps/*/signals/**
GET:/query/**
GET:/apps/*/query/**
POST:/signals/**
```

- Adds the role `webapps-role` with these permissions:

```
GET,HEAD:/webapps/**
GET,HEAD:/license
```

### Migrations that might require adjustments

This section describes parts of migration for which you might need to make adjustments after migration.

<table>
<thead>
<tr>
<th>Important</th>
<th>Following migration, we recommend that you review the API and UI permissions for roles, and the roles and API permissions for users.</th>
</tr>
</thead>
</table>

- If you have created your own developer roles, add these permissions to the roles:

```
GET:/license
GET,POST,PUT:/appkit
GET,POST,PUT:/apps/**
```

- If you have created your own search roles, add these permissions to the roles:
Signals and Spark jobs

Fusion Server 4.2.y indexes search log events into the `<collection>_signals` collections as `response` signals, instead of indexing them into the `<collection>_logs` collections.

Roughly 90 new fields used by App Insights are added to existing `<collection>_signals` collections.

You are encouraged to adopt the new signal fields, but you can continue using the old dynamic field names, such as `user_id_s`, until Fusion 5.0. If you adopt the new signals schema, then you must update any Spark jobs that rely on the old field names.

Fusion Server 4.2.y replaces the `_signals_ingest` index pipeline with a new version that works with the new signals schema.

If you made changes to the `_signals_ingest` pipeline, then you'll need to manually add those changes to the new configuration after migration.

The migrator preserves legacy-style aggregation jobs. Optionally, you can manually convert these to SQL-based aggregation jobs.

Fusion Server 4.2.y adds a new session rollup job for each collection with signals enabled. The session rollup job creates session signals that contain aggregated information about user activity in a session.

Fusion Server 4.2.y adds a new head/tail analysis job for each collection with signals enabled. The head/tail analysis job uses signals to compute interesting metrics for head-and-tail queries.

Review known issues

Before upgrading, review the known issues to see whether any of them apply to the circumstances of your upgrade. Some known issues might require actions before upgrading.

That article also contains instructions regarding what to do if an upgrade step fails.

Upgrade on Unix

Use this procedure to upgrade Fusion on a single Unix node or on multiple Unix nodes.

Perform the steps in this procedure on the indicated nodes on which Fusion is running ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (api)
- Fusion UI service (ui)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don’t need to do anything with the external Solr and/or ZooKeeper instances.
Important

For every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. There is the notion of a "main node" during the migration process. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the "main node"; there's no special requirement as to which one you pick.

Install the Java JDK

Before upgrading to Fusion Server 4.2 or later, you must install the Java JDK on all nodes that will run Fusion Server. Prior versions of Fusion Server could use either the JRE or the JDK.

For more information, see Java requirements.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.
2. Extract the newer version of Fusion:

   ```
   cd /opt/lucidworks
   mv ~/Downloads/fusion-4.2.y.tar.gz ./
   tar -xf fusion-4.2.y.tar.gz
   ```

   For example, if Fusion is currently installed in `/opt/lucidworks/fusion/3.1.x`, then change your working directory to `/opt/lucidworks/` and extract the file there. Don't run the new version of Fusion yet.

3. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid `license.properties` file in the `/opt/lucidworks/fusion/4.2.y/conf` directory.

4. (If there are custom jar files) If your deployment has custom jar files, add them to the new Fusion deployment.

5. (If you are performing an upgrade without Internet access) Without Internet access, the migrator can't download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in `apps/connectors/bootstrap-plugins` for the new deployment.

6. (If you are adding new connectors) If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in `apps/connectors/bootstrap-plugins` for the new deployment.

7. Verify that there is sufficient disk space for a second copy of the Solr index directory, `fusion/3.1.x/data/solr`. If there isn't sufficient disc space, free up space before proceeding.

Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. Download the latest migrator zip file for Unix. (Do this now, even if you have downloaded the migrator before, to
ensure that you have the latest version.)

2. Create `FUSION_OLD` and `FUSION_NEW` environment variables that point to the old and new Fusion installation directories respectively (using the full path).

   ```bash
   export FUSION_OLD="/opt/lucidworks/fusion/3.1.x"
   export FUSION_NEW="/opt/lucidworks/fusion/4.2.y"
   ```

   For example, when upgrading from Fusion 3.1.5 to 4.2.5:

   ```bash
   export FUSION_OLD="/opt/lucidworks/fusion/3.1.5"
   export FUSION_NEW="/opt/lucidworks/fusion/4.2.5"
   ```

3. Create this directory:

   ```bash
   mkdir $FUSION_OLD/var/upgrade
   ```

4. Install the migrator:

   ```bash
   tar -zxv -C $FUSION_OLD/var/upgrade --strip-components=1 -f fusion-migrator.tar.gz
   ```

**Run the migrator**

Perform these tasks on the indicated nodes:

1. **(On all Fusion nodes)** Start all Fusion services for the old version of Fusion:

   ```bash
   $FUSION_OLD/bin/fusion start
   ```

2. **(Only on the main Fusion node)** Run the migrator to export the configuration data from the old version of Fusion:

   ```bash
   java -jar $FUSION_OLD/var/upgrade/migrator.jar --export
   ```

   This message indicates that the step finished successfully:

   ```bash
   Old Fusion configuration export (--export) finished successfully.
   ```

3. **(On all Fusion nodes)** Stop the old versions of Fusion services and Solr; but not ZooKeeper:

   ```bash
   $FUSION_OLD/bin/ui stop
   $FUSION_OLD/bin/connectors stop
   $FUSION_OLD/bin/api stop
   $FUSION_OLD/bin/solr stop
   ```
Tip
You can see what is running with \$FUSION_OLD/bin/fusion status.

4. **(Only on secondary Fusion nodes)** Prepare secondary nodes:

```java
java -jar \$FUSION_OLD/var/upgrade/migrator.jar --prepare-secondary
```

This message indicates that the step finished successfully:

Prepare secondary nodes (--prepare-secondary) finished successfully.

5. **(On all Fusion nodes)** Stop ZooKeeper for the old version of Fusion (*unless you are using an external ZooKeeper instance, in which case you can ignore this step*):

```
\$FUSION_OLD/bin/zookeeper stop
```

6. **(Only on the main Fusion node)** Transform configuration data on the main Fusion node:

```java
java -jar \$FUSION_OLD/var/upgrade/migrator.jar --main-transform
```

**Note**
Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

Fusion data transformations on main node (--main-transform) finished successfully.

7. **(On all Fusion nodes)** Start ZooKeeper for the new version of Fusion (*unless you are using an external ZooKeeper instance, in which case you can ignore this step*):

```
\$FUSION_NEW/bin/zookeeper start
```

8. **(Only on the main Fusion node)** Import the first part of configuration data into the new version of Fusion:

```java
java -jar \$FUSION_OLD/var/upgrade/migrator.jar --zookeeper-import
```

This message indicates that the step finished successfully:

New Fusion Zookeeper import (--zookeeper-import) finished successfully.

9. **(If upgrading from Fusion 3.1.0 through 3.1.3; On all Fusion nodes)** Start Solr for the new Fusion version:

```
\$FUSION_NEW/bin/solr start
```
10. *(If upgrading from Fusion 3.1.0 through 3.1.3; Only on the main Fusion node)* Run a script to remove all old plugins from the blob store. Replace `solr-address` and `solr-port` as appropriate (as shown in the example):

```
java -cp "$FUSION_OLD/var/upgrade/jython-standalone-2.7.1.jar:$FUSION_OLD/var/upgrade/migrator.jar"
org.python.util.jython "$FUSION_OLD/var/upgrade/transformations/manual_delete_old_plugin_blobs.py" --solr-address solr-address --solr-port solr-port
```

For example:

```
java -cp "$FUSION_OLD/var/upgrade/jython-standalone-2.7.1.jar:$FUSION_OLD/var/upgrade/migrator.jar"
org.python.util.jython "$FUSION_OLD/var/upgrade/transformations/manual_delete_old_plugin_blobs.py" --solr-address localhost --solr-port 8983
```

This message indicates that plugins were deleted successfully:

```
Deleted old plugin blobs from solr
<?xml version="1.0" encoding="UTF-8"?>
<response>
<lst name="responseHeader">
  <int name="status">0</int>
  <int name="QTime">246</int>
</lst>
</response>
Old connector plugin blobs were deleted successfully.
```

11. *(On all Fusion nodes)* Start all Fusion services for the new version of Fusion:

```
$FUSION_NEW/bin/fusion start
```

12. *(Only on the main Fusion node)* Import the second part of configuration data into the new version of Fusion:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```

**Tip**

After migration, you can find details about the results in the `fusion/3.1.x/var/upgrade/tmp` directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting.

**Validate the new version of Fusion**

How to validate the new version of Fusion

1. *(Only on the main Fusion node)* Restart the new version of Fusion (all services defined in `fusion.properties`):
2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser’s cache.
      Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.
   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).
   c. Log in.
   d. Navigate to Admin > About Fusion.
      The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. If you migrated from version 3.1.x to version 4.2.0 or later, confirm associations between blobs and apps and make adjustments if necessary.

6. When you are satisfied with the migration and you have backed up the fusion/3.1.x/ directory, you can rm -fr fusion/3.1.x/ to remove the older version of Fusion (on all Fusion nodes).

**Add support for business rules to existing apps**

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The add-rule-objects-xyz.zip file (where xyz is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is $FUSION_OLD/var/upgrade/import-files.

| Note | Adding support for business rules has a costs. Additional collections and objects are created. Only add support for business rules to apps in which you plan to use them. |

You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

**How to import business rule objects**
1. In the Fusion launcher, click the app into which you want to import objects.
   
   The Fusion workspace appears.

2. In the upper left, click System > Import Fusion Objects.
   
   The Import Fusion Objects window opens.

3. For the data file, select add-rule-objects-xyz.zip from your local filesystem. The location in the extracted migrator files is $FUSION_OLD/var/upgrade/import-files.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, Lucene Revolution would become Lucene_Revolution.

6. Click Import.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click Merge to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click Close to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

How to import business rule objects

1. Create an app-name.txt file with the following content:

   
   ```
   {"app.id" : "_name-of-migrated-app_"}
   ```

   For example, for the app Lucene Revolution:

   ```
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the $HOME environment variable is defined.

2. Import the business rule objects:

   ```
   ```

   For example:
Upgrade on Windows

Use this procedure to upgrade Fusion on a single Windows node or multiple Windows nodes.

Perform the steps in this procedure on the indicated nodes on which Fusion is running ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (api)
- Fusion UI service (ui)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don’t need to do anything with the external Solr and/or ZooKeeper instances.

Important

If you are upgrading Fusion on multiple nodes, then, for every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the "main node"; there’s no special requirement as to which one you pick.

Install the Java JDK

Before upgrading to Fusion Server 4.2 or later, you must install the Java JDK on all nodes that will run Fusion Server. Prior versions of Fusion Server could use either the JRE or the JDK.

For more information, see Java requirements.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.
2. Move the fusion-4.2.y.zip file to the directory that contains the fusion\ directory.
   For example, if Fusion is installed in C:\lucidworks\fusion\3.1.x, then move the file to C:\lucidworks.
3. Unzip the fusion-4.2.y.zip file. Don’t run the new version of Fusion yet.
4. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the C:\lucidworks\fusion\4.2.y\conf directory.
5. (If there are custom jar files) If your deployment has custom jar files, add them to the new Fusion deployment.
6. (If you are performing an upgrade without Internet access) Without Internet access, the migrator can’t download new
versions of connectors automatically. Download the new versions of connector zip files for your current connectors from http://lucidworks.com/connectors/ and place them in apps\connectors\bootstrap-plugins for the new deployment.

7. (If you are adding new connectors) If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps\connectors\bootstrap-plugins for the new deployment.

8. Verify that there is sufficient disk space for a second copy of the Solr index directory, fusion\3.1.x\data\solr. If there isn't sufficient disc space, free up space before proceeding.

Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. Download the latest migrator zip file for Windows. (Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)

2. Open a Command Prompt window and create FUSION_OLD and FUSION_NEW environment variables that point to the old and new Fusion installation directories respectively. For example:

   ```
   set FUSION_OLD=C:\lucidworks\fusion\3.1.5
   set FUSION_NEW=C:\lucidworks\fusion\4.2.5
   ```

3. Create a fusion\3.1.x\var\upgrade directory.

4. Unzip the migrator zip file, and move the contents of the extracted folder to fusion\3.1.x\var\upgrade.

Run the migrator

Perform these tasks on the indicated nodes:

1. (On all Fusion nodes) Start all Fusion services for the old version of Fusion:

   ```
   %FUSION_OLD%\bin\fusion.cmd start
   ```

2. (Only on the main Fusion node) Run the migrator to export the configuration data from the old version of Fusion:

   ```
   java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --export
   ```

   This message indicates that the step finished successfully:

   ```
   Old Fusion configuration export (--export) finished successfully.
   ```

3. (On all Fusion nodes) Stop the old versions of Fusion services and Solr; but not ZooKeeper:

   ```
   %FUSION_OLD%\bin\ui.cmd stop
   %FUSION_OLD%\bin\connectors.cmd stop
   %FUSION_OLD%\bin\api.cmd stop
   %FUSION_OLD%\bin\solr.cmd stop
   ```
4. (Only on secondary Fusion nodes) Prepare secondary nodes:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --prepare-secondary
```

This message indicates that the step finished successfully:

```
Prepare secondary nodes (--prepare-secondary) finished successfully.
```

5. (On all Fusion nodes) Stop ZooKeeper for the old version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):

```
%FUSION_OLD%\bin\zookeeper.cmd stop
```

6. (Only on the main Fusion node) Transform configuration data on the main Fusion node:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --main-transform
```

Note

Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

```
Fusion data transformations on main node (--main-transform) finished successfully.
```

7. (On all Fusion nodes) Start ZooKeeper for the new version of Fusion (unless you are only using an external ZooKeeper instance, in which case you can ignore this step):

```
%FUSION_NEW%\bin\zookeeper.cmd start
```

8. (Only on the main Fusion node) Import the first part of configuration data into the new version of Fusion:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --zookeeper-import
```

This message indicates that the step finished successfully:

```
New Fusion Zookeeper import (--zookeeper-import) finished successfully.
```

9. (If upgrading from Fusion 3.1.0 through 3.1.3; On all Fusion nodes) Start Solr for the new Fusion version:

```
%FUSION_NEW%\bin\solr.cmd start
```
10. (If upgrading from Fusion 3.1.0 through 3.1.3; Only on the main Fusion node) Run a script to remove all old plugins from the blob store. Replace **solr-address** and **solr-port** as appropriate (as shown in the example):

```java
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar"
org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address solr-address --solr-port solr-port
```

For example:

```java
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar"
org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address localhost --solr-port 8983
```

This message indicates that plugins were deleted successfully:

```xml
<response>
    <lst name="responseHeader">
        <int name="status">0</int>
        <int name="QTime">246</int>
    </lst>
    Old connector plugin blobs were deleted successfully.
</response>
```

11. (On all Fusion nodes) Start all Fusion services for the new version of Fusion:

```
%FUSION_NEW%\bin\fusion.cmd start
```

12. (Only on the main Fusion node) Import the second part of configuration data into the new version of Fusion:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```

| Tip | After migration, you can find details about the results in the **fusion\3.1.x\var\upgrade\tmp** directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting. |

**Validate the new version of Fusion**

How to validate the new version of Fusion

1. (On all Fusion nodes) Restart all Fusion services for the new version of Fusion:
2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser’s cache. Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.
   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).
   c. Log in.
   d. Navigate to Admin > About Fusion.
      The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. If you migrated from version 3.1.x to version 4.2.0 or later, confirm associations between blobs and apps and make adjustments if necessary.

6. When you are satisfied with the migration and you have backed up the fusion\3.1.x directory, you can remove the older version of Fusion by removing that directory (on all Fusion nodes).

**Add support for business rules to existing apps**

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The add-rule-objects-xyz.zip file (where xyz is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is %FUSION_OLD%\var\upgrade\import-files\.

You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

**How to import business rule objects**

1. In the Fusion launcher, click the app into which you want to import objects.
   The Fusion workspace appears.
2. In the upper left, click System > Import Fusion Objects.
   The Import Fusion Objects window opens.

3. For the data file, select add-rule-objects-xyz.zip from your local filesystem. The location in the extracted migrator files is %FUSION_OLD%/var/upgrade/import-files.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, Lucene Revolution would become Lucene_Solution.

6. Click Import.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click Merge to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click Close to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

**How to import business rule objects**

1. Create an app-name.txt file with the following content:

   ```json
   {"app.id" : "_name-of-migrated-app_"}
   ```

   For example, for the app Lucene Revolution:

   ```json
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the %HOMEPATH% environment variable is defined.

2. Import the business rule objects:

   ```bash
   ```

   For example:

   ```bash
   ```
Upgrade Fusion Server 4.0.x to 4.2.y

Introduction

This article describes how to perform the following upgrade:

- From version: Fusion 4.0.x
- To version: Fusion 4.2.y

Important

Only specific version-to-version upgrade sequences are supported. Some upgrades require multiple steps. For more information, see the supported upgrade sequences.

For Fusion 3.1 and later releases, a migrator is available for upgrading Fusion.

During the upgrade process, the migrator uses a properties file. After downloading and installing the migrator, the properties files is in the /opt/lucidworks/fusion/4.0.x/var/upgrade directory (on Unix or MacOS) or the C:\lucidworks\fusion\4.0.x\var\upgrade\ directory (on Windows). The file names reference the versions you are upgrading from and to. For example:

- To upgrade 4.0.2 to 4.2.5, the migrator uses the 4.0.x-4.2.x.properties file.

Migration entails down time and multiple starts and stops of Fusion services. Plan accordingly, especially in terms of disabling external load balancers or monitors that might react adversely to the starts and stops.

Download the latest migrator immediately before upgrading. This helps ensure that the upgrade goes smoothly.

Important

The newer Fusion instance must be newly untarred and never started.

About the upgrade

This section describes how connectors, object migrations, and signals are migrated during an upgrade.

Connectors

In Fusion 3.1.0 and above, only a subset of connectors are included by default.

The migrator detects which connectors were used in the older version of Fusion, and installs them automatically in Fusion 4.2.y. This step requires an Internet connection. If no connection is available, then you must download connectors at http://lucidworks.com/connectors/ and install them as bootstrap plugins.

If a connector to be upgraded wasn't available during the upgrade, then a message in /opt/lucidworks/fusion/4.0.x/var/upgrade/tmp/migrator.log (on Unix) or C:\lucidworks\fusion\4.0.x\var\upgrade\tmp\migrator.log (on Windows) indicates this.

Only datasources for connectors that are supported in the new Fusion version are upgraded. Datasources for custom connectors aren't upgraded.
If no Internet connection is available

If no Internet connection is available during an upgrade, the migrator can’t automatically download the connectors it needs. Using the Fusion UI or API later to install the connectors also might not be an option.

In this case, download the connector zip files for all existing connectors and any connectors that you are adding from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new deployment (on all Fusion nodes). Do so at the time indicated in the procedures that follow.

Adding connectors during an upgrade

You can add connectors during an upgrade (that is, add connectors that aren't in the old deployment).

Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new version (on all Fusion nodes).

Object migrations and transformations

The migrator automatically migrates these Fusion 4.0 object types, transforming them as needed:

- Collections
- Index pipelines
- Query pipelines
- Search cluster configurations
- Datasources
- Parsing configurations
- Object groups
- Links
- Tasks
- Jobs
- Spark objects
- Apps
- Appkit apps
- Index profiles
- Query profiles
- Blobs

Important

In Fusion Server 4.0 and later, most objects exist in the context of apps. When you upgrade from Fusion Server 4.0.x to 4.2.y, the migrator upgrades app objects, all objects in or linked to objects in apps, and objects that are not linked to apps. You can explore the objects in Object Explorer.
Noteworthy transformations

While migrating objects, the migrator makes these transformations:

- It adds the new log-shipper service to fusion.properties.
- It removes obsolete log-cleanup tasks.
- It converts the format of job history records.

Access control migration

The migrator upgrades all access control configurations:

- Security realms
- Roles
- Users

None of these should require adjustments after migration.

Association of blobs with apps

The association of blobs with apps becomes stronger in Fusion Server 4.1.1. Newly created blobs are only linked to the app to which they are uploaded, and are only shared with other apps when the blobs are directly linked to those apps (added to the apps in Object Explorer or linked using the Links API).

When migrating from pre-4.1.1 versions to 4.1.1 or later, the resulting apps might not contain all of the blobs that they contained prior to migration.

<table>
<thead>
<tr>
<th>Note</th>
<th>It's important to stress that this change doesn't break the apps.</th>
</tr>
</thead>
</table>

What might change is the blobs that are linked to an app. This can be important if you export and re-import apps.

**Recommendation:** After migration from a pre-4.1.1 version to version 4.1.1 or later, we recommend that you use either Object Explorer or the Links API to manually link all blobs to the apps in which they are used.

Review known issues

Before upgrading, review the known issues to see whether any of them apply to the circumstances of your upgrade. Some known issues might require actions before upgrading.

That article also contains instructions regarding what to do if an upgrade step fails.

Upgrade on Unix

Use this procedure to upgrade Fusion on a single Unix node or on multiple Unix nodes.

Perform the steps in this procedure on the indicated nodes on which Fusion is running ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (api)
• Proxy service (*proxy*)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don’t need to do anything with the external Solr and/or ZooKeeper instances.

| Important | Important for every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. There is the notion of a “main node” during the migration process. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the “main node”; there’s no special requirement as to which one you pick. |

Ensure that your current version of Fusion has a valid license

Ensure that your current version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid `license.properties` file in the `/opt/lucidworks/fusion/4.0.x/conf` directory.

Install the Java JDK

Before upgrading to Fusion Server 4.2 or later, you must install the Java JDK on all nodes that will run Fusion Server. Prior versions of Fusion Server could use either the JRE or the JDK.

For more information, see Java requirements.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.

2. Extract the newer version of Fusion:

   ```
   cd /opt/lucidworks
   mv ~Downloads/fusion-4.2.y.tar.gz ./
   tar -xf fusion-4.2.y.tar.gz
   ```

   For example, if Fusion is currently installed in `/opt/lucidworks/fusion/4.0.x`, then change your working directory to `/opt/lucidworks/` and extract the file there. *Don’t run the new version of Fusion yet.*

3. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid `license.properties` file in the `/opt/lucidworks/fusion/4.2.y/conf` directory.

4. *(If there are custom jar files)* If your deployment has custom jar files, add them to the new Fusion deployment.

5. *(If you are performing an upgrade without Internet access)* Without Internet access, the migrator can’t download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from http://lucidworks.com/connectors/ and place them in `apps/connectors/bootstrap-plugins` for the new deployment.

6. *(If you are adding new connectors)* If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from http://lucidworks.com/connectors/ and
place them in `apps/connectors/bootstrap-plugins` for the new deployment.

7. Verify that there is sufficient disk space for a second copy of the Solr index directory, `fusion/4.0.x/data/solr`. If there isn’t sufficient disc space, free up space before proceeding.

**Download and install the Fusion migrator**

Perform these tasks on all Fusion nodes:

1. Download the latest migrator zip file for Unix. (Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)

2. Create `FUSION_OLD` and `FUSION_NEW` environment variables that point to the old and new Fusion installation directories respectively (using the full path).

   ```bash
   export FUSION_OLD="/opt/lucidworks/fusion/4.0.x"
   export FUSION_NEW="/opt/lucidworks/fusion/4.2.y"
   ```

   For example, when upgrading from Fusion 4.0.2 to 4.2.5:

   ```bash
   export FUSION_OLD="/opt/lucidworks/fusion/4.0.2"
   export FUSION_NEW="/opt/lucidworks/fusion/4.2.5"
   ```

3. Create this directory:

   ```bash
   mkdir $FUSION_OLD/var/upgrade
   ```

4. Install the migrator:

   ```bash
   tar -zxv -C $FUSION_OLD/var/upgrade --strip-components=1 -f fusion-migrator.tar.gz
   ```

**Run the migrator**

Perform these tasks on the indicated nodes:

1. (On all Fusion nodes) Start all Fusion services for the old version of Fusion:

   ```bash
   $FUSION_OLD/bin/fusion start
   ```

2. (Only on the main Fusion node) Run the migrator to export the configuration data from the old version of Fusion:

   ```bash
   java -jar $FUSION_OLD/var/upgrade/migrator.jar --export
   ```

   This message indicates that the step finished successfully:

   ```bash
   Old Fusion configuration export (--export) finished successfully.
   ```
3. (On all Fusion nodes) Stop the old versions of Fusion services and Solr; but not ZooKeeper:

```
$FUSION_OLD/bin/webapps stop
$FUSION_OLD/bin/admin-ui stop
$FUSION_OLD/bin/proxy stop
$FUSION_OLD/bin/connectors-classic stop
$FUSION_OLD/bin/connectors-rpc stop
$FUSION_OLD/bin/api stop
$FUSION_OLD/bin/solr stop
```

If Spark and SQL services are running, also stop those:

```
$FUSION_OLD/bin/spark-master stop
$FUSION_OLD/bin/spark-worker stop
$FUSION_OLD/bin/sql stop
```

**Tip**

You can see what is running with `$FUSION_OLD/bin/fusion status`.

4. (Only on secondary Fusion nodes) Prepare secondary nodes:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --prepare-secondary
```

This message indicates that the step finished successfully:

```
Prepare secondary nodes (--prepare-secondary) finished successfully.
```

5. (On all Fusion nodes) Stop ZooKeeper for the old version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):

```
$FUSION_OLD/bin/zookeeper stop
```

6. (Only on the main Fusion node) Transform configuration data on the main Fusion node:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --main-transform
```

**Note**

Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

```
Fusion data transformations on main node (--main-transform) finished successfully.
```

7. (On all Fusion nodes) Start ZooKeeper for the new version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):
8. (Only on the main Fusion node) Import the first part of configuration data into the new version of Fusion:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --zookeeper-import
```

This message indicates that the step finished successfully:

```
New Fusion Zookeeper import (--zookeeper-import) finished successfully.
```

9. (On all Fusion nodes) Start all Fusion services for the new version of Fusion:

```
$FUSION_NEW/bin/fusion start
```

10. (Only on the main Fusion node) Import the second part of configuration data into the new version of Fusion:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```

### Tip

After migration, you can find details about the results in the `fusion/4.0.x/var/upgrade/tmp` directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting.

---

## Validate the new version of Fusion

### How to validate the new version of Fusion

1. (Only on the main Fusion node) Restart the new version of Fusion (all services defined in `fusion.properties`):

```
$FUSION_NEW/bin/fusion restart
```

2. Log into the Fusion UI (your `admin` password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser's cache.
      Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.
   b. In a browser, open the Fusion UI at [http://localhost:8764/](http://localhost:8764/) (replace `localhost` with your server name or IP address if necessary).
   c. Log in.
d. Navigate to **Admin > About Fusion**.

The **About Fusion** panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click **System > Blobs**. If any connectors are missing from the list, click **Add > Connector Plugin** and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. If you migrated from version 4.0.x to version 4.2.0 or later, confirm associations between blobs and apps and make adjustments if necessary.

6. When you are satisfied with the migration and you have backed up the `fusion/4.0.x/` directory, you can `rm -fr fusion/4.0.x/` to remove the older version of Fusion (**on all Fusion nodes**).

### Add support for business rules to existing apps

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The `add-rule-objects-xyz.zip` file (where `xyz` is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is `$FUSION_OLD/var/upgrade/import-files`.

| Note | Adding support for business rules has a cost. Additional collections and objects are created. Only add support for business rules to apps in which you plan to use them. |

You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

**How to import business rule objects**

1. In the Fusion launcher, click the app into which you want to import objects.
   
   The Fusion workspace appears.

2. In the upper left, click **System > Import Fusion Objects**.
   
   The Import Fusion Objects window opens.

3. For the data file, select `add-rule-objects-xyz.zip` from your local filesystem. The location in the extracted migrator files is `$FUSION_OLD/var/upgrade/import-files`.

4. Click **Import**.

5. Edit the **Application ID** parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, `Lucene Revolution` would become `Lucene_Revolution`. 
6. Click **Import**.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click **Merge** to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click **Close** to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

How to import business rule objects

1. Create an **app-name.txt** file with the following content:

   ```json
   {"app.id": "_name-of-migrated-app_"}
   ```

   For example, for the app **Lucene Revolution**:

   ```json
   {"app.id": "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the `$HOME` environment variable is defined.

2. Import the business rule objects:

   ```bash
   ```

   For example:

   ```bash
   ```

**Upgrade on Windows**

Use this procedure to upgrade Fusion on a single Windows node or multiple Windows nodes.

Perform the steps in this procedure on the indicated nodes on which Fusion is running ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (**api**)
- Proxy service (**proxy**)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don’t need to do anything with the external Solr and/or ZooKeeper instances.

### Important

If you are upgrading Fusion on multiple nodes, then, for every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. There is the notion of a “main node” during the migration process. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the “main node”; there’s no special requirement as to which one you pick.

### Ensure that your current version of Fusion has a valid license

Ensure that your current version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the C:\lucidworks\fusion\4.0.x\conf directory.

### Install the Java JDK

Before upgrading to Fusion Server 4.2 or later, you must install the Java JDK on all nodes that will run Fusion Server. Prior versions of Fusion Server could use either the JRE or the JDK.

For more information, see Java requirements.

### Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.
2. Move the fusion-4.2.y.zip file to the directory that contains the fusion\ directory.
   
   For example, if Fusion is installed in C:\lucidworks\fusion\4.0.x, then move the file to C:\lucidworks.
3. Unzip the fusion-4.2.y.zip file. Don't run the new version of Fusion yet.
4. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the C:\lucidworks\fusion\4.2.y\conf directory.
5. (If there are custom jar files) If your deployment has custom jar files, add them to the new Fusion deployment.
6. (If you are performing an upgrade without Internet access) Without Internet access, the migrator can’t download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in apps\connectors\bootstrap-plugins for the new deployment.
7. (If you are adding new connectors) If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in apps\connectors\bootstrap-plugins for the new deployment.
8. Verify that there is sufficient disk space for a second copy of the Solr index directory, fusion\4.0.x\data\solr. If there isn’t sufficient disc space, free up space before proceeding.
Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. **Download the latest migrator zip file for Windows.** (Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)

2. Open a Command Prompt window and create `FUSION_OLD` and `FUSION_NEW` environment variables that point to the old and new Fusion installation directories respectively. For example:

   ```
   set FUSION_OLD=C:\lucidworks\fusion\4.0.2
   set FUSION_NEW=C:\lucidworks\fusion\4.2.5
   ```

3. Create a `fusion\4.0.x\var\upgrade` directory.

4. Unzip the migrator zip file, and move the contents of the extracted folder to `fusion\4.0.x\var\upgrade`.

Run the migrator

Perform these tasks on the indicated nodes:

1. *(On all Fusion nodes)* Start all Fusion services for the old version of Fusion:

   ```
   %FUSION_OLD%\bin\fusion.cmd start
   ```

2. *(Only on the main Fusion node)* Run the migrator to export the configuration data from the old version of Fusion:

   ```
   java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --export
   ```

   This message indicates that the step finished successfully:

   ```
   Old Fusion configuration export (--export) finished successfully.
   ```

3. *(On all Fusion nodes)* Stop the old versions of Fusion services and Solr; but not ZooKeeper:

   ```
   %FUSION_OLD%\bin\webapps.cmd stop
   %FUSION_OLD%\bin\admin-ui.cmd stop
   %FUSION_OLD%\bin\proxy.cmd stop
   %FUSION_OLD%\bin\connectors-classic.cmd stop
   %FUSION_OLD%\bin\connectors-rpc.cmd stop
   %FUSION_OLD%\bin\api.cmd stop
   %FUSION_OLD%\bin\solr.cmd stop
   ```

   If Spark and SQL services are running, also stop those:

   ```
   %FUSION_OLD%\bin\spark-master.cmd stop
   %FUSION_OLD%\bin\spark-worker.cmd stop
   %FUSION_OLD%\bin\sql.cmd stop
   ```
4. **(Only on secondary Fusion nodes)** Prepare secondary nodes:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --prepare-secondary
```

This message indicates that the step finished successfully:

```
Prepare secondary nodes (--prepare-secondary) finished successfully.
```

5. **(On all Fusion nodes)** Stop ZooKeeper for the old version of Fusion *(unless you are using an external ZooKeeper instance, in which case you can ignore this step)*:

```
%FUSION_OLD%\bin\zookeeper.cmd stop
```

6. **(Only on the main Fusion node)** Transform configuration data on the main Fusion node:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --main-transform
```

**Note**

Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

```
Fusion data transformations on main node (--main-transform) finished successfully.
```

7. **(On all Fusion nodes)** Start ZooKeeper for the new version of Fusion *(unless you are only using an external ZooKeeper instance, in which case you can ignore this step)*:

```
%FUSION_NEW%\bin\zookeeper.cmd start
```

8. **(Only on the main Fusion node)** Import the first part of configuration data into the new version of Fusion:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --zookeeper-import
```

This message indicates that the step finished successfully:

```
New Fusion Zookeeper import (--zookeeper-import) finished successfully.
```

9. **(On all Fusion nodes)** Start Solr for the new Fusion version:

```
%FUSION_NEW%\bin\solr.cmd start
```
10. *(Only on the main Fusion node)* Run a script to remove all old plugins from the blob store. Replace `solr-address` and `solr-port` as appropriate (as shown in the example):

```java
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar"
org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address solr-address --solr-port solr-port
```

For example:

```java
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar"
org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address localhost --solr-port 8983
```

This message indicates that plugins were deleted successfully:

```
Deleted old plugin blobs from solr
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <lst name="responseHeader">
    <int name="status">0</int>
    <int name="QTime">246</int>
  </lst>
  <response>
Old connector plugin blobs were deleted successfully.
```

11. *(On all Fusion nodes)* Start all Fusion services for the new version of Fusion:

```
%FUSION_NEW%\bin\fusion.cmd start
```

12. *(Only on the main Fusion node)* Import the second part of configuration data into the new version of Fusion:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```

**Tip**

After migration, you can find details about the results in the `fusion\4.0.x\var\upgrade\tmp` directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting.

---

**Validate the new version of Fusion**

How to validate the new version of Fusion

1. *(On all Fusion nodes)* Restart all Fusion services for the new version of Fusion:
2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser’s cache.
      Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.
   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).
   c. Log in.
   d. Navigate to Admin > About Fusion.
      The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. If you migrated from version 4.0.x to version 4.2.0 or later, confirm associations between blobs and apps and make adjustments if necessary.

6. When you are satisfied with the migration and you have backed up the fusion\4.0.x directory, you can remove the older version of Fusion by removing that directory (on all Fusion nodes).

Add support for business rules to existing apps

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The add-rule-objects-xyz.zip file (where xyz is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is %FUSION_OLD%\var\upgrade\import-files\.

You have a choice. You can update each app using the Fusion UI or the Fusion API.

Fusion UI

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

How to import business rule objects

1. In the Fusion launcher, click the app into which you want to import objects.

   The Fusion workspace appears.
2. In the upper left, click System > Import Fusion Objects.
   
   The Import Fusion Objects window opens.

3. For the data file, select add-rule-objects-xyz.zip from your local filesystem. The location in the extracted migrator files is %FUSION_OLD%\var\upgrade\import-files\.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, Lucene Revolution would become Lucene_Revolution.

6. Click Import.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click Merge to skip all conflicting objects and import only the non-conflicting objects.
   
   Fusion confirms that the import was successful.

8. Click Close to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

**How to import business rule objects**

1. Create an app-name.txt file with the following content:

   ```json
   {"app.id" : "_name-of-migrated-app_"}
   ```

   For example, for the app Lucene Revolution:

   ```json
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the %HOMEPATH% environment variable is defined.

2. Import the business rule objects:

   ```
   curl -u admin:{password} -H "Content-Type:multipart/form-data" -X POST -F 'variableValues=path-to-txt-file\app-name.txt' -F "importData=path-to-zip-file\add-rule-objects-xyz.zip" 
   ```

   For example:

   ```
   curl -u admin:Very-SecurePW8 -H "Content-Type:multipart/form-data" -X POST -F 'variableValues=%HOMEPATH%\Lucene_Revolution.txt' -F "importData=%FUSION_OLD%\import-files\add-rule-objects-001.zip" 
   ```
Upgrade Fusion Server 4.1.x to 4.2.y

Introduction

This article describes how to perform the following upgrade:

• From version: Fusion 4.1.x
• To version: Fusion 4.2.y

Important

Only specific version-to-version upgrade sequences are supported. Some upgrades require multiple steps. For more information, see the supported upgrade sequences.

For Fusion 3.1 and later releases, a migrator is available for upgrading Fusion.

During the upgrade process, the migrator uses a properties file. After downloading and installing the migrator, the properties file is in the /opt/lucidworks/fusion/4.1.x/var/upgrade directory (on Unix or MacOS) or the C:\lucidworks\fusion\4.1.x\var\upgrade\ directory (on Windows). The file names reference the versions you are upgrading from and to. For example:

• To upgrade 4.1.3 to 4.2.5, the migrator uses the 4.1.x-4.2.x.properties file.

Migration entails down time and multiple starts and stops of Fusion services. Plan accordingly, especially in terms of disabling external load balancers or monitors that might react adversely to the starts and stops.

Download the latest migrator immediately before upgrading. This helps ensure that the upgrade goes smoothly.

Important

The newer Fusion instance must be newly untarred and never started.

About the upgrade

This section describes how connectors, object migrations, and signals are migrated during an upgrade.

Connectors

In Fusion 3.1.0 and above, only a subset of connectors are included by default.

The migrator detects which connectors were used in the older version of Fusion, and installs them automatically in Fusion 4.2.y. This step requires an Internet connection. If no connection is available, then you must download connectors at http://lucidworks.com/connectors/ and install them as bootstrap plugins.

If a connector to be upgraded wasn’t available during the upgrade, then a message in /opt/lucidworks/fusion/4.1.x/var/upgrade/tmp/migrator.log (on Unix) or C:\lucidworks\fusion\4.1.x\var\upgrade\tmp\migrator.log (on Windows) indicates this.

Only datasources for connectors that are supported in the new Fusion version are upgraded. Datasources for custom connectors aren’t upgraded.
If no Internet connection is available during an upgrade, the migrator can’t automatically download the connectors it needs. Using the Fusion UI or API later to install the connectors also might not be an option.

In this case, download the connector zip files for all existing connectors and any connectors that you are adding from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new deployment (on all Fusion nodes). Do so at the time indicated in the procedures that follow.

Adding connectors during an upgrade

You can add connectors during an upgrade (that is, add connectors that aren't in the old deployment).

Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new version (on all Fusion nodes).

Object migrations and transformations

The migrator automatically migrates these Fusion 4.1 object types, transforming them as needed:

- Collections
- Index pipelines
- Query pipelines
- Search cluster configurations
- Datasources
- Parsing configurations
- Object groups
- Links
- Tasks
- Jobs
- Spark objects
- Apps
- Appkit apps
- Index profiles
- Query profiles
- Blobs

**Important**

In Fusion Server 4.0 and later, most objects exist in the context of apps. When you upgrade from Fusion Server 4.1.x to 4.2.y, the migrator upgrades app objects, all objects in or linked to objects in apps, and objects that are not linked to apps. You can explore the objects in Object Explorer.
Access control migration

The migrator upgrades all access control configurations:

- Security realms
- Roles
- Users

None of these should require adjustments after migration.

Association of blobs with apps

The association of blobs with apps becomes stronger in Fusion Server 4.1.1. Newly created blobs are only linked to the app to which they are uploaded, and are only shared with other apps when the blobs are directly linked to those apps (added to the apps in Object Explorer or linked using the Links API).

When migrating from pre-4.1.1 versions to 4.1.1 or later, the resulting apps might not contain all of the blobs that they contained prior to migration.

| Note | It's important to stress that this change doesn't break the apps. |

What might change is the blobs that are linked to an app. This can be important if you export and re-import apps.

**Recommendation:** After migration from a pre-4.1.1 version to version 4.1.1 or later, we recommend that you use either Object Explorer or the Links API to manually link all blobs to the apps in which they are used.

Review known issues

Before upgrading, review the known issues to see whether any of them apply to the circumstances of your upgrade. Some known issues might require actions *before* upgrading.

That article also contains instructions regarding what to do if an upgrade step fails.

Upgrade on Unix

Use this procedure to upgrade Fusion on a single Unix node or on multiple Unix nodes.

Perform the steps in this procedure on the indicated nodes *on which Fusion is running* ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (*api*)
- Proxy service (*proxy*)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don’t need to do anything with the external Solr and/or ZooKeeper instances.
Important

For every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. There is the notion of a “main node” during the migration process. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the “main node”; there’s no special requirement as to which one you pick.

Ensure that your current version of Fusion has a valid license

Ensure that your current version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the /opt/lucidworks/fusion/4.1.x/conf directory.

Install the Java JDK

Before upgrading to Fusion Server 4.2 or later, you must install the Java JDK on all nodes that will run Fusion Server. Prior versions of Fusion Server could use either the JRE or the JDK.

For more information, see Java requirements.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.

2. Extract the newer version of Fusion:

   cd /opt/lucidworks
   mv ~/Downloads/fusion-4.2.y.tar.gz ./
   tar -xf fusion-4.2.y.tar.gz

   For example, if Fusion is currently installed in /opt/lucidworks/fusion/4.1.x, then change your working directory to /opt/lucidworks/ and extract the file there. Don’t run the new version of Fusion yet.

3. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the /opt/lucidworks/fusion/4.2.y/conf directory.

4. (If there are custom jar files) If your deployment has custom jar files, add them to the new Fusion deployment.

5. (If you are performing an upgrade without Internet access) Without Internet access, the migrator can’t download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new deployment.

6. (If you are adding new connectors) If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new deployment.

7. Verify that there is sufficient disk space for a second copy of the Solr index directory, fusion/4.1.x/data/solr. If there isn’t sufficient disc space, free up space before proceeding.
Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. **Download the latest migrator zip file for Unix.** (Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)

2. Create `FUSION_OLD` and `FUSION_NEW` environment variables that point to the old and new Fusion installation directories respectively (using the full path).

   ```sh
   export FUSION_OLD="/opt/lucidworks/fusion/4.1.x"
   export FUSION_NEW="/opt/lucidworks/fusion/4.2.y"
   ```

   For example, when upgrading from Fusion 4.1.3 to 4.2.5:

   ```sh
   export FUSION_OLD="/opt/lucidworks/fusion/4.1.3"
   export FUSION_NEW="/opt/lucidworks/fusion/4.2.5"
   ```

3. Create this directory:

   ```sh
   mkdir $FUSION_OLD/var/upgrade
   ```

4. Install the migrator:

   ```sh
   tar -zxv -C $FUSION_OLD/var/upgrade --strip-components=1 -f fusion-migrator.tar.gz
   ```

Run the migrator

Perform these tasks on the indicated nodes:

1. **(On all Fusion nodes) Start all Fusion services for the old version of Fusion:**

   ```sh
   $FUSION_OLD/bin/fusion start
   ```

2. **(Only on the main Fusion node) Run the migrator to export the configuration data from the old version of Fusion:**

   ```sh
   java -jar $FUSION_OLD/var/upgrade/migrator.jar --export
   ```

   This message indicates that the step finished successfully:

   ```
   Old Fusion configuration export (--export) finished successfully.
   ```

3. **(On all Fusion nodes) Stop the old versions of Fusion services and Solr; but not ZooKeeper:**
$FUSION_OLD/bin/log-shipper stop
$FUSION_OLD/bin/admin-ui stop
$FUSION_OLD/bin/webapps stop
$FUSION_OLD/bin/proxy stop
$FUSION_OLD/bin/connectors-rpc stop
$FUSION_OLD/bin/connectors-classic stop
$FUSION_OLD/bin/api stop
$FUSION_OLD/bin/solr stop

If Spark and SQL services are running, also stop those:

$FUSION_OLD/bin/spark-master stop
$FUSION_OLD/bin/spark-worker stop
$FUSION_OLD/bin/sql stop

Tip You can see what is running with $FUSION_OLD/bin/fusion status.

4. (Only on secondary Fusion nodes) Prepare secondary nodes:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --prepare-secondary
```

This message indicates that the step finished successfully:

Prepare secondary nodes (--prepare-secondary) finished successfully.

5. (On all Fusion nodes) Stop ZooKeeper for the old version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):

```
$FUSION_OLD/bin/zookeeper stop
```

6. (Only on the main Fusion node) Transform configuration data on the main Fusion node:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --main-transform
```

Note Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

Fusion data transformations on main node (--main-transform) finished successfully.

7. (On all Fusion nodes) Start ZooKeeper for the new version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):
8. *(Only on the main Fusion node)* Import the first part of configuration data into the new version of Fusion:

```bash
java -jar $FUSION_OLD/var/upgrade/migrator.jar --zookeeper-import
```

This message indicates that the step finished successfully:

```
New Fusion Zookeeper import (--zookeeper-import) finished successfully.
```

9. *(On all Fusion nodes)* Start all Fusion services for the new version of Fusion:

```bash
$FUSION_NEW/bin/fusion start
```

10. *(Only on the main Fusion node)* Import the second part of configuration data into the new version of Fusion:

```bash
java -jar $FUSION_OLD/var/upgrade/migrator.jar --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```

| Tip | After migration, you can find details about the results in the fusion/4.1.x/var/upgrade/tmp directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting. |

**Validate the new version of Fusion**

How to validate the new version of Fusion

1. *(Only on the main Fusion node)* Restart the new version of Fusion (all services defined in fusion.properties):

```bash
$FUSION_NEW/bin/fusion restart
```

2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser’s cache.

   Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.

   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).

   c. Log in.
d. Navigate to Admin > About Fusion.

The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. If you migrated from version 4.1.0 to version 4.2.0 or later, confirm associations between blobs and apps and make adjustments if necessary.

6. When you are satisfied with the migration and you have backed up the fusion/4.1.x/ directory, you can `rm -fr fusion/4.1.x/` to remove the older version of Fusion (on all Fusion nodes).

**Add support for business rules to existing apps**

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The `add-rule-objects-xyz.zip` file (where `xyz` is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is `$FUSION_OLD/var/upgrade/import-files`.

| Note | Adding support for business rules has a cost. Additional collections and objects are created. Only add support for business rules to apps in which you plan to use them. |

You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

**How to import business rule objects**

1. In the Fusion launcher, click the app into which you want to import objects.

   The Fusion workspace appears.

2. In the upper left, click System > Import Fusion Objects.

   The Import Fusion Objects window opens.

3. For the data file, select `add-rule-objects-xyz.zip` from your local filesystem. The location in the extracted migrator files is `$FUSION_OLD/var/upgrade/import-files`.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, Lucene Revolution would become Lucene_Revolution.
6. Click **Import**.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click **Merge** to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click **Close** to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

**How to import business rule objects**

1. Create an **app-name.txt** file with the following content:

   ```json
   {"app.id" : "_name-of-migrated-app_
   }
   ```

   For example, for the app Lucene Revolution:

   ```json
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the `$HOME` environment variable is defined.

2. Import the business rule objects:

   ```bash
   ```

   For example:

   ```bash
   ```

**Upgrade on Windows**

Use this procedure to upgrade Fusion on a single Windows node or multiple Windows nodes.

Perform the steps in this procedure on the indicated nodes *on which Fusion is running* ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (api)
- Proxy service (proxy)
If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don’t need to do anything with the external Solr and/or ZooKeeper instances.

<table>
<thead>
<tr>
<th>Important</th>
<th>Important</th>
</tr>
</thead>
<tbody>
<tr>
<td>If you are upgrading Fusion on multiple nodes, then, for every step on</td>
<td>If you are upgrading Fusion on multiple nodes, then, for every step on</td>
</tr>
<tr>
<td>multiple nodes, ensure that the step completes on all Fusion nodes</td>
<td>multiple nodes, ensure that the step completes on all Fusion nodes</td>
</tr>
<tr>
<td>before going to the next step.</td>
<td>before going to the next step.</td>
</tr>
<tr>
<td>There is the notion of a “main node” during the migration process.</td>
<td>There is the notion of a “main node” during the migration process.</td>
</tr>
<tr>
<td>This node will be used for certain centralized migration activities that</td>
<td>This node will be used for certain centralized migration activities that</td>
</tr>
<tr>
<td>do not need to be done on every node, such as downloading connectors</td>
<td>do not need to be done on every node, such as downloading connectors</td>
</tr>
<tr>
<td>that are then uploaded to blob storage that is shared by all, etc.</td>
<td>that are then uploaded to blob storage that is shared by all, etc.</td>
</tr>
<tr>
<td>Just pick one of your Fusion nodes to be the “main node”; there's no</td>
<td>Just pick one of your Fusion nodes to be the “main node”; there's no</td>
</tr>
<tr>
<td>special requirement as to which one you pick.</td>
<td>special requirement as to which one you pick.</td>
</tr>
</tbody>
</table>

Ensure that your current version of Fusion has a valid license

Ensure that your current version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid `license.properties` file in the `C:\lucidworks\fusion\4.1.x\conf` directory.

Install the Java JDK

Before upgrading to Fusion Server 4.2 or later, you must install the Java JDK on all nodes that will run Fusion Server. Prior versions of Fusion Server could use either the JRE or the JDK.

For more information, see Java requirements.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.

2. Move the `fusion-4.2.y.zip` file to the directory that contains the `fusion\` directory.

   For example, if Fusion is installed in `C:\lucidworks\fusion\4.1.x`, then move the file to `C:\lucidworks`.

3. Unzip the `fusion-4.2.y.zip` file. Don't run the new version of Fusion yet.

4. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid `license.properties` file in the `C:\lucidworks\fusion\4.2.y\conf` directory.

5. (If there are custom jar files) If your deployment has custom jar files, add them to the new Fusion deployment.

6. (If you are performing an upgrade without Internet access) Without Internet access, the migrator can’t download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in `apps\connectors\bootstrap-plugins` for the new deployment.

7. (If you are adding new connectors) If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in `apps\connectors\bootstrap-plugins` for the new deployment.

8. Verify that there is sufficient disk space for a second copy of the Solr index directory, `fusion\4.1.x\data\solr`. If there isn’t sufficient disc space, free up space before proceeding.
## Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. **Download the latest migrator zip file for Windows.** (Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)

2. Open a Command Prompt window and create `FUSION_OLD` and `FUSION_NEW` environment variables that point to the old and new Fusion installation directories respectively. For example:

   ```
   set FUSION_OLD=C:\lucidworks\fusion\4.1.3
   set FUSION_NEW=C:\lucidworks\fusion\4.2.5
   ```

3. Create a `fusion\4.1.x\var\upgrade` directory.

4. Unzip the migrator zip file, and move the contents of the extracted folder to `fusion\4.1.x\var\upgrade`.

## Run the migrator

Perform these tasks on the indicated nodes:

1. **(On all Fusion nodes)** Start all Fusion services for the old version of Fusion:

   ```
   %FUSION_OLD%\bin\fusion.cmd start
   ```

2. **(Only on the main Fusion node)** Run the migrator to export the configuration data from the old version of Fusion:

   ```
   java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --export
   ```

   This message indicates that the step finished successfully:

   ```
   Old Fusion configuration export (--export) finished successfully.
   ```

3. **(On all Fusion nodes)** Stop the old versions of Fusion services and Solr; but not ZooKeeper:

   ```
   %FUSION_OLD%\bin\log-shipper.cmd stop
   %FUSION_OLD%\bin\admin-ui.cmd stop
   %FUSION_OLD%\bin\webapps.cmd stop
   %FUSION_OLD%\bin\proxy.cmd stop
   %FUSION_OLD%\bin\connectors-rpc.cmd stop
   %FUSION_OLD%\bin\connectors-classic.cmd stop
   %FUSION_OLD%\bin\api.cmd stop
   %FUSION_OLD%\bin\solr.cmd stop
   ```

   If Spark and SQL services are running, also stop those:

   ```
   %FUSION_OLD%\bin\spark-master.cmd stop
   %FUSION_OLD%\bin\spark-worker.cmd stop
   %FUSION_OLD%\bin\sql.cmd stop
   ```
<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tip</strong></td>
<td>You can see what is running with <code>%FUSION_OLD%\bin\fusion status</code>.</td>
</tr>
<tr>
<td><strong>4. (Only on secondary Fusion nodes) Prepare secondary nodes:</strong></td>
<td>java -jar &quot;%FUSION_OLD%\var\upgrade\migrator.jar&quot; --prepare-secondary</td>
</tr>
<tr>
<td></td>
<td>This message indicates that the step finished successfully:</td>
</tr>
<tr>
<td></td>
<td>Prepare secondary nodes (--prepare-secondary) finished successfully.</td>
</tr>
<tr>
<td><strong>5. (On all Fusion nodes) Stop ZooKeeper for the old version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):</strong></td>
<td>%FUSION_OLD%\bin\zookeeper.cmd stop</td>
</tr>
<tr>
<td><strong>6. (Only on the main Fusion node) Transform configuration data on the main Fusion node:</strong></td>
<td>java -jar &quot;%FUSION_OLD%\var\upgrade\migrator.jar&quot; --main-transform</td>
</tr>
<tr>
<td></td>
<td>Note: Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).</td>
</tr>
<tr>
<td></td>
<td>This message indicates that the step finished successfully:</td>
</tr>
<tr>
<td></td>
<td>Fusion data transformations on main node (--main-transform) finished successfully.</td>
</tr>
<tr>
<td><strong>7. (On all Fusion nodes) Start ZooKeeper for the new version of Fusion (unless you are only using an external ZooKeeper instance, in which case you can ignore this step):</strong></td>
<td>%FUSION_NEW%\bin\zookeeper.cmd start</td>
</tr>
<tr>
<td><strong>8. (Only on the main Fusion node) Import the first part of configuration data into the new version of Fusion:</strong></td>
<td>java -jar &quot;%FUSION_OLD%\var\upgrade\migrator.jar&quot; --zookeeper-import</td>
</tr>
<tr>
<td></td>
<td>This message indicates that the step finished successfully:</td>
</tr>
<tr>
<td></td>
<td>New Fusion Zookeeper import (--zookeeper-import) finished successfully.</td>
</tr>
<tr>
<td><strong>9. (On all Fusion nodes) Start Solr for the new Fusion version:</strong></td>
<td>%FUSION_NEW%\bin\solr.cmd start</td>
</tr>
</tbody>
</table>
10. **(Only on the main Fusion node)** Run a script to remove all old plugins from the blob store. Replace `solr-address` and `solr-port` as appropriate (as shown in the example):

```
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar"
org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address solr-address --solr-port solr-port
```

For example:

```
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar"
org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address localhost --solr-port 8983
```

This message indicates that plugins were deleted successfully:

```
Deleted old plugin blobs from solr 〈?xml version="1.0" encoding="UTF-8"?〉
<response>
  <lst name="responseHeader">
    <int name="status">0</int>
    <int name="QTime">246</int>
  </lst>
</response>
Old connector plugin blobs were deleted successfully.
```

11. **(On all Fusion nodes)** Start all Fusion services for the new version of Fusion:

```
%FUSION_NEW%\bin\fusion.cmd start
```

12. **(Only on the main Fusion node)** Import the second part of configuration data into the new version of Fusion:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```

**Tip**

After migration, you can find details about the results in the `fusion\4.1.x\var\upgrade\tmp` directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting.

### Validate the new version of Fusion

How to validate the new version of Fusion

1. **(On all Fusion nodes)** Restart all Fusion services for the new version of Fusion:
2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser’s cache.
      Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.
   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).
   c. Log in.
   d. Navigate to Admin > About Fusion.
      The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. If you migrated from version 4.1.0 to version 4.2.0 or later, confirm associations between blobs and apps and make adjustments if necessary.

6. When you are satisfied with the migration and you have backed up the fusion\4.1.x directory, you can remove the older version of Fusion by removing that directory (on all Fusion nodes).

**Add support for business rules to existing apps**

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The add-rule-objects-xyz.zip file (where xyz is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is %FUSION_OLD%\var\upgrade\import-files\.

You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

**How to import business rule objects**

1. In the Fusion launcher, click the app into which you want to import objects.
   The Fusion workspace appears.
2. In the upper left, click System > Import Fusion Objects.

   The Import Fusion Objects window opens.

3. For the data file, select add-rule-objects-xyz.zip from your local filesystem. The location in the extracted migrator files is %FUSION_OLD%\var\upgrade\import-files\.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, Lucene Revolution would become Lucene_Revolution.

6. Click Import.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click Merge to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click Close to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the add-rule-objects-xyz.zip file into the app.

How to import business rule objects

1. Create an app-name.txt file with the following content:

   ```json
   {"app.id" : "_name-of-migrated-app_"}
   ```

   For example, for the app Lucene Revolution:

   ```json
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the %HOMEPATH% environment variable is defined.

2. Import the business rule objects:

   ```bash
   ```

   For example:

   ```bash
   ```
Upgrade Fusion Server 4.2.x to 4.2.y

Introduction

This article describes how to perform the following upgrade:

- From version: Fusion 4.2.x
- To version: Fusion 4.2.y

Important

Only specific version-to-version upgrade sequences are supported. Some upgrades require multiple steps. For more information, see the supported upgrade sequences.

For Fusion 3.1 and later releases, a migrator is available for upgrading Fusion.

During the upgrade process, the migrator uses a properties file. After downloading and installing the migrator, the properties files is in the /opt/lucidworks/fusion/4.2.x/var/upgrade directory (on Unix or MacOS) or the C:\lucidworks\fusion\4.2.x\var\upgrade\ directory (on Windows). The file names reference the versions you are upgrading from and to. For example:

- To upgrade 4.2.3 to 4.2.5, the migrator uses the 4.2.x-4.2.x.properties file.

Migration entails down time and multiple starts and stops of Fusion services. Plan accordingly, especially in terms of disabling external load balancers or monitors that might react adversely to the starts and stops.

Download the latest migrator immediately before upgrading. This helps ensure that the upgrade goes smoothly.

Important

The newer Fusion instance must be newly untarred and never started.

About the upgrade

This section describes how connectors, object migrations, and signals are migrated during an upgrade.

Connectors

In Fusion 3.1.0 and above, only a subset of connectors are included by default.

The migrator detects which connectors were used in the older version of Fusion, and installs them automatically in Fusion 4.2.y. This step requires an Internet connection. If no connection is available, then you must download connectors at http://lucidworks.com/connectors/ and install them as bootstrap plugins.

If a connector to be upgraded wasn’t available during the upgrade, then a message in /opt/lucidworks/fusion/4.2.x/var/upgrade/tmp/migrator.log (on Unix) or C:\lucidworks\fusion\4.2.x\var\upgrade\tmp\migrator.log (on Windows) indicates this.

Only datasources for connectors that are supported in the new Fusion version are upgraded. Datasources for custom connectors aren’t upgraded.
If no Internet connection is available

If no Internet connection is available during an upgrade, the migrator can’t automatically download the connectors it needs. Using the Fusion UI or API later to install the connectors also might not be an option.

In this case, download the connector zip files for all existing connectors and any connectors that you are adding from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new deployment (on all Fusion nodes). Do so at the time indicated in the procedures that follow.

Adding connectors during an upgrade

You can add connectors during an upgrade (that is, add connectors that aren't in the old deployment).

Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps/connectors/bootstrap-plugins for the new version (on all Fusion nodes).

Object migrations and transformations

The migrator automatically migrates these Fusion 4.2 object types, transforming them as needed:

- Collections
- Index pipelines
- Query pipelines
- Search cluster configurations
- Datasources
- Parsing configurations
- Object groups
- Links
- Tasks
- Jobs
- Spark objects
- Apps
- Appkit apps
- Index profiles
- Query profiles
- Blobs

Important

In Fusion Server 4.0 and later, most objects exist in the context of apps. When you upgrade from Fusion Server 4.2.x to 4.2.y, the migrator upgrades app objects, all objects in or linked to objects in apps, and objects that are not linked to apps. You can explore the objects in Object Explorer.
Access control migration

The migrator upgrades all access control configurations:

- Security realms
- Roles
- Users

None of these should require adjustments after migration.

Review known issues

Before upgrading, review the known issues to see whether any of them apply to the circumstances of your upgrade. Some known issues might require actions before upgrading.

That article also contains instructions regarding what to do if an upgrade step fails.

Upgrade on Unix

Use this procedure to upgrade Fusion on a single Unix node or on multiple Unix nodes.

Perform the steps in this procedure on the indicated nodes on which Fusion is running ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (api)
- Proxy service (proxy)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don't need to do anything with the external Solr and/or ZooKeeper instances.

| Important | For every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. There is the notion of a "main node" during the migration process. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the "main node"; there’s no special requirement as to which one you pick. |

Ensure that your current version of Fusion has a valid license

Ensure that your current version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the /opt/lucidworks/fusion/4.2.x/conf directory.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.
2. Extract the newer version of Fusion:

```bash
cd /opt/lucidworks
mv ~/Downloads/fusion-4.2.y.tar.gz ./
tar -xf fusion-4.2.y.tar.gz
```

For example, if Fusion is currently installed in `/opt/lucidworks/fusion/4.2.x`, then change your working directory to `/opt/lucidworks/` and extract the file there. *Don’t run the new version of Fusion yet.*

3. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the `/opt/lucidworks/fusion/4.2.y/conf` directory.

4. *(If there are custom jar files)* If your deployment has custom jar files, add them to the new Fusion deployment.

5. *(If you are performing an upgrade without Internet access)* Without Internet access, the migrator can’t download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in `apps/connectors/bootstrap-plugins` for the new deployment.

6. *(If you are adding new connectors)* If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from [http://lucidworks.com/connectors/](http://lucidworks.com/connectors/) and place them in `apps/connectors/bootstrap-plugins` for the new deployment.

7. Verify that there is sufficient disk space for a second copy of the Solr index directory, `fusion/4.2.x/data/solr`. If there isn’t sufficient disc space, free up space before proceeding.

### Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. Download the latest migrator zip file for Unix. *(Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)*

2. Create FUSION_OLD and FUSION_NEW environment variables that point to the old and new Fusion installation directories respectively (using the full path).

```bash
export FUSION_OLD="/opt/lucidworks/fusion/4.2.x"
export FUSION_NEW="/opt/lucidworks/fusion/4.2.y"
```

For example, when upgrading from Fusion 4.2.0 to 4.2.5:

```bash
export FUSION_OLD="/opt/lucidworks/fusion/4.2.0"
export FUSION_NEW="/opt/lucidworks/fusion/4.2.5"
```

3. Create this directory:

```bash
mkdir $FUSION_OLD/var/upgrade
```

4. Install the migrator:
Run the migrator

Perform these tasks on the indicated nodes:

1. **(On all Fusion nodes)** Start all Fusion services for the old version of Fusion:
   
   ```
   $FUSION_OLD/bin/fusion start
   ```

2. **(Only on the main Fusion node)** Run the migrator to export the configuration data from the old version of Fusion:
   
   ```
   java -jar $FUSION_OLD/var/upgrade/migrator.jar --export
   ```

   This message indicates that the step finished successfully:
   
   Old Fusion configuration export (--export) finished successfully.

3. **(On all Fusion nodes)** Stop the old versions of Fusion services and Solr; but not ZooKeeper:
   
   ```
   $FUSION_OLD/bin/log-shipper stop
   $FUSION_OLD/bin/sql stop
   $FUSION_OLD/bin/admin-ui stop
   $FUSION_OLD/bin/webapps stop
   $FUSION_OLD/bin/proxy stop
   $FUSION_OLD/bin/connectors-rpc stop
   $FUSION_OLD/bin/connectors-classic stop
   $FUSION_OLD/bin/api stop
   $FUSION_OLD/bin/solr stop
   ```

   If Spark services are running, also stop those:
   
   ```
   $FUSION_OLD/bin/spark-master stop
   $FUSION_OLD/bin/spark-worker stop
   ```

   **Tip**
   
   You can see what is running with `$FUSION_OLD/bin/fusion status`.

4. **(Only on secondary Fusion nodes)** Prepare secondary nodes:
   
   ```
   java -jar $FUSION_OLD/var/upgrade/migrator.jar --prepare-secondary
   ```

   This message indicates that the step finished successfully:
   
   Prepare secondary nodes (--prepare-secondary) finished successfully.
5. (*On all Fusion nodes*) Stop ZooKeeper for the old version of Fusion (*unless you are using an external ZooKeeper instance, in which case you can ignore this step*):

```
$FUSION_OLD/bin/zookeeper stop
```

6. (*Only on the main Fusion node*) Transform configuration data on the main Fusion node:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --main-transform
```

**Note**

Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

```
Fusion data transformations on main node (--main-transform) finished successfully.
```

7. (*On all Fusion nodes*) Start ZooKeeper for the new version of Fusion (*unless you are using an external ZooKeeper instance, in which case you can ignore this step*):

```
$FUSION_NEW/bin/zookeeper start
```

8. (*Only on the main Fusion node*) Import the first part of configuration data into the new version of Fusion:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --zookeeper-import
```

This message indicates that the step finished successfully:

```
New Fusion Zookeeper import (--zookeeper-import) finished successfully.
```

9. (*On all Fusion nodes*) Start all Fusion services for the new version of Fusion:

```
$FUSION_NEW/bin/fusion start
```

10. (*Only on the main Fusion node*) Import the second part of configuration data into the new version of Fusion:

```
java -jar $FUSION_OLD/var/upgrade/migrator.jar --fusion-import
```

This message indicates that the step finished successfully:

```
New Fusion object import (--fusion-import) finished successfully.
```
Tip

After migration, you can find details about the results in the fusion/4.2.x/var/upgrade/tmp directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting.

Validate the new version of Fusion

How to validate the new version of Fusion

1. *(Only on the main Fusion node)* Restart the new version of Fusion (all services defined in fusion.properties):

   
   ```bash
   $FUSION_NEW/bin/fusion restart
   ```

2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   
   a. Clear your browser's cache.

   Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.

   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).

   c. Log in.

   d. Navigate to Admin > About Fusion.

   The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. When you are satisfied with the migration and you have backed up the fusion/4.2.x/ directory, you can `rm -fr fusion/4.2.x/` to remove the older version of Fusion (on all Fusion nodes).

Add support for business rules to existing apps

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.

The add-rule-objects-xyz.zip file (where xyz is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is $FUSION_OLD/var/upgrade/import-files.

Note

Adding support for business rules has a cost. Additional collections and objects are created. Only add support for business rules to apps in which you plan to use them.
You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

How to import business rule objects

1. In the Fusion launcher, click the app into which you want to import objects.

   The Fusion workspace appears.

2. In the upper left, click System > Import Fusion Objects.

   The Import Fusion Objects window opens.

3. For the data file, select `add-rule-objects-xyz.zip` from your local filesystem. The location in the extracted migrator files is `$FUSION_OLD/var/upgrade/import-files`.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, Lucene Revolution would become Lucene_Revolution.

6. Click Import.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click Merge to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click Close to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

How to import business rule objects

1. Create an `app-name.txt` file with the following content:

   ```json
   {"app.id" : "_name-of-migrated-app_"}
   ```

   For example, for the app Lucene Revolution:

   ```json
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the $HOME environment variable is defined.

2. Import the business rule objects:
Upgrade on Windows

Use this procedure to upgrade Fusion on a single Windows node or multiple Windows nodes.

Perform the steps in this procedure on the indicated nodes on which Fusion is running ("Fusion nodes"). To perform an upgrade, Fusion nodes must have at least these services running:

- API service (api)
- Proxy service (proxy)

If Solr and/or ZooKeeper instances are also running on other nodes (without Fusion), you don't need to do anything with the external Solr and/or ZooKeeper instances.

Important

If you are upgrading Fusion on multiple nodes, then, for every step on multiple nodes, ensure that the step completes on all Fusion nodes before going to the next step. There is the notion of a "main node" during the migration process. This node will be used for certain centralized migration activities that do not need to be done on every node, such as downloading connectors that are then uploaded to blob storage that is shared by all, etc. Just pick one of your Fusion nodes to be the "main node"; there's no special requirement as to which one you pick.

Ensure that your current version of Fusion has a valid license

Ensure that your current version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the C:\lucidworks\fusion\4.2.x\conf directory.

Download and install the newer version of Fusion

Perform these tasks on all Fusion nodes:

1. Download the version of Fusion to which you are upgrading.
2. Move the fusion-4.2.y.zip file to the directory that contains the fusion\ directory.
   
   For example, if Fusion is installed in C:\lucidworks\fusion\4.2.x, then move the file to C:\lucidworks.

3. Unzip the fusion-4.2.y.zip file. Don't run the new version of Fusion yet.
4. Ensure that the new version of Fusion has a valid permanent Fusion license before proceeding with the upgrade. Place a valid license.properties file in the C:\lucidworks\fusion\4.2.y\conf directory.

5. (If there are custom jar files) If your deployment has custom jar files, add them to the new Fusion deployment.

6. (If you are performing an upgrade without Internet access) Without Internet access, the migrator can’t download new versions of connectors automatically. Download the new versions of connector zip files for your current connectors from http://lucidworks.com/connectors/ and place them in apps\connectors\bootstrap-plugins for the new deployment.

7. (If you are adding new connectors) If you want your new deployment to use connectors that are not in the current deployment, you can add them now. Download the connector zip files from http://lucidworks.com/connectors/ and place them in apps\connectors\bootstrap-plugins for the new deployment.

8. Verify that there is sufficient disk space for a second copy of the Solr index directory, fusion\4.2.x\data\solr. If there isn’t sufficient disk space, free up space before proceeding.

Download and install the Fusion migrator

Perform these tasks on all Fusion nodes:

1. Download the latest migrator zip file for Windows. (Do this now, even if you have downloaded the migrator before, to ensure that you have the latest version.)

2. Open a Command Prompt window and create FUSION_OLD and FUSION_NEW environment variables that point to the old and new Fusion installation directories respectively. For example:

   ```
   set FUSION_OLD=C:\lucidworks\fusion\4.2.0
   set FUSION_NEW=C:\lucidworks\fusion\4.2.5
   ```

3. Create a fusion\4.2.x\var\upgrade directory.

4. Unzip the migrator zip file, and move the contents of the extracted folder to fusion\4.2.x\var\upgrade.

Run the migrator

Perform these tasks on the indicated nodes:

1. (On all Fusion nodes) Start all Fusion services for the old version of Fusion:

   ```
   %FUSION_OLD%\bin\fusion.cmd start
   ```

2. (Only on the main Fusion node) Run the migrator to export the configuration data from the old version of Fusion:

   ```
   java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --export
   ```

   This message indicates that the step finished successfully:

   ```
   Old Fusion configuration export (--export) finished successfully.
   ```

3. (On all Fusion nodes) Stop the old versions of Fusion services and Solr; but not ZooKeeper:
%FUSION_OLD%/bin/log-shipper.cmd stop
%FUSION_OLD%/bin\sql.cmd stop
%FUSION_OLD%/bin\admin-ui.cmd stop
%FUSION_OLD%/bin\webapps.cmd stop
%FUSION_OLD%/bin\proxy.cmd stop
%FUSION_OLD%/bin\connectors-rpc.cmd stop
%FUSION_OLD%/bin\connectors-classic.cmd stop
%FUSION_OLD%/bin\api.cmd stop
%FUSION_OLD%/bin\solr.cmd stop

If Spark and SQL services are running, also stop those:

%FUSION_OLD%/bin\spark-master.cmd stop
%FUSION_OLD%/bin\spark-worker.cmd stop

Tip  You can see what is running with %FUSION_OLD%/bin\fusion status.

4. (Only on secondary Fusion nodes) Prepare secondary nodes:

java -jar "%FUSION_OLD%/var\upgrade\migrator.jar" --prepare-secondary

This message indicates that the step finished successfully:

Prepare secondary nodes (--prepare-secondary) finished successfully.

5. (On all Fusion nodes) Stop ZooKeeper for the old version of Fusion (unless you are using an external ZooKeeper instance, in which case you can ignore this step):

%FUSION_OLD%/bin\zookeeper.cmd stop

6. (Only on the main Fusion node) Transform configuration data on the main Fusion node:

java -jar "%FUSION_OLD%/var\upgrade\migrator.jar" --main-transform

Note  Depending on the size of your Solr index, this step can take a long time (for example, multiple tens of minutes).

This message indicates that the step finished successfully:

Fusion data transformations on main node (--main-transform) finished successfully.

7. (On all Fusion nodes) Start ZooKeeper for the new version of Fusion (unless you are only using an external ZooKeeper instance, in which case you can ignore this step):
8. (Only on the main Fusion node) Import the first part of configuration data into the new version of Fusion:

```
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --zookeeper-import
```

This message indicates that the step finished successfully:

```
New Fusion Zookeeper import (--zookeeper-import) finished successfully.
```

9. (On all Fusion nodes) Start Solr for the new Fusion version:

```
%FUSION_NEW%\bin\solr.cmd start
```

10. (Only on the main Fusion node) Run a script to remove all old plugins from the blob store. Replace `solr-address` and `solr-port` as appropriate (as shown in the example):

```
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar" org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address solr-address --solr-port solr-port
```

For example:

```
java -cp "%FUSION_OLD%\var\upgrade\jython-standalone-2.7.1.jar;%FUSION_OLD%\var\upgrade\migrator.jar" org.python.util.jython "%FUSION_OLD%\var\upgrade\transformations\manual_delete_old_plugin_blobs.py" --solr-address localhost --solr-port 8983
```

This message indicates that plugins were deleted successfully:

```
Deleted old plugin blobs from solr
<?xml version="1.0" encoding="UTF-8"?>
<response>
<lst name="responseHeader">
  <int name="status">0</int>
  <int name="QTime">246</int>
</lst>
</response>
Old connector plugin blobs were deleted successfully.
```

11. (On all Fusion nodes) Start all Fusion services for the new version of Fusion:

```
%FUSION_NEW%\bin\fusion.cmd start
```

12. (Only on the main Fusion node) Import the second part of configuration data into the new version of Fusion:
java -jar "%FUSION_OLD%\var\upgrade\migrator.jar" --fusion-import

This message indicates that the step finished successfully:

New Fusion object import (--fusion-import) finished successfully.

Tip  | After migration, you can find details about the results in the fusion\4.2.x\var\upgrade\tmp directory. If the migration produces unexpected results, the files in this directory are helpful for troubleshooting.

Validate the new version of Fusion

How to validate the new version of Fusion

1. (On all Fusion nodes) Restart all Fusion services for the new version of Fusion:

   %FUSION_NEW%\bin\fusion.cmd restart

2. Log into the Fusion UI (your admin password is the same as for the old installation), and confirm the release number of the new version of Fusion:
   a. Clear your browser’s cache.
   Otherwise, you might inadvertently access a cached version of the old Fusion UI and see inconsistent behavior.
   b. In a browser, open the Fusion UI at http://localhost:8764/ (replace localhost with your server name or IP address if necessary).
   c. Log in.
   d. Navigate to Admin > About Fusion.
   The About Fusion panel should display the newer Fusion release number.

3. Ensure that all connectors were installed automatically during the upgrade. From the Fusion launcher, click the tile for a migrated app. Click System > Blobs. If any connectors are missing from the list, click Add > Connector Plugin and install them manually.

4. Ensure that all customizations you made in the former version of Fusion are present in the new one.

5. When you are satisfied with the migration and you have backed up the fusion\4.2.x directory, you can remove the older version of Fusion by removing that directory (on all Fusion nodes).

Add support for business rules to existing apps

Fusion AI 4.2 introduces functionality for using business rules, that is, manually created formulas for rewriting queries and responses.

You can add support for business rules to apps that were created in versions of Fusion AI prior to version 4.2. To do so, perform the steps in this section.
The `add-rule-objects-xyz.zip` file (where `xyz` is a version number) specifies the objects to add to an app. It is supplied in the Fusion migrator zip file at the top level. After installing the migrator, the location is `%FUSION_OLD%/var/upgrade/import-files`.

You have a choice. You can update each app using the Fusion UI or the Fusion API.

**Fusion UI**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

How to import business rule objects

1. In the Fusion launcher, click the app into which you want to import objects.
   
   The Fusion workspace appears.

2. In the upper left, click System > Import Fusion Objects.
   
   The Import Fusion Objects window opens.

3. For the data file, select `add-rule-objects-xyz.zip` from your local filesystem. The location in the extracted migrator files is `%FUSION_OLD%/var/upgrade/import-files`.

4. Click Import.

5. Edit the Application ID parameter value to use the app name. If the app name contains spaces, replace those with underscore characters. For example, `Lucene Revolution` would become `Lucene_Revolution`.

6. Click Import.

7. If there are conflicts, Fusion prompts you to specify an import policy. Click Merge to skip all conflicting objects and import only the non-conflicting objects.

   Fusion confirms that the import was successful.

8. Click Close to close the Import Fusion Objects window.

**Fusion API**

For each app in which you plan to use business rules, import the objects in the `add-rule-objects-xyz.zip` file into the app.

How to import business rule objects

1. Create an `app-name.txt` file with the following content:

   ```json
   {"app.id" : ".name-of-migrated-app."}
   ```

   For example, for the app `Lucene Revolution`:

   ```json
   {"app.id" : "Lucene_Revolution"}
   ```

   Here, we assume that you create the files in your home directory, for which the `%HOMEPATH%` environment variable is
defined.

2. Import the business rule objects:

```
```

For example:

```
```
Troubleshoot Upgrades

Beginning with Fusion 3.1, upgrading Fusion uses a migrator that migrates objects and data from your current version of Fusion to a new version of Fusion or Fusion Server.

The Fusion migrator log file will help you troubleshoot difficulties while upgrading. If the migrator fails to complete successfully, we explain how to correct problems and repeat an upgrade.

Known issues, and workarounds if available, are listed at the bottom of this topic.

Log file

The log file for the migrator is in /opt/lucidworks/fusion/x.y.z/var/upgrade/tmp/migrator.log (on Unix) or C:\lucidworks\var\fusion\x.y.z\upgrade\tmp\migrator.log (on Windows). The x.y.z directory is for the Fusion version that you are migrating from.

Completing an upgrade after an error

If the migrator script produces an error, you will need to correct the problem that the script encountered, and then finish the upgrade as explained here.

[[1-fix-the-error]]

1. Fix the error

If an upgrade step fails:

1. Examine the log file and determine what went wrong.
2. Fix what went wrong.

[[2-repeat-the-step-that-failed]]

2. Repeat the step that failed

After you have fixed the source of the error, repeat the step that failed.

| Note | You can repeat the step that failed as many times as needed to correct errors that occur. If the step continues to fail, restart the upgrade from the beginning.

[[3-complete-the-upgrade]]

3. Complete the upgrade

Complete the upgrade.

If a different step fails, fix that error, repeat that step, and then perform the remainder of the upgrade procedure.

If the upgrade steps produce no additional errors, then you are finished.

Restarting an upgrade

If an upgrade step continues to fail, then restart the upgrade process from the beginning:

1. Examine the log file and determine what went wrong.
2. Remove the directory for the new version of Fusion.
3. Reinstall the new version of Fusion, but don’t start it.
4. Fix what went wrong.
5. Perform the entire upgrade procedure from the beginning.

## Known issues

Following are known issues with the Fusion migrator.

<table>
<thead>
<tr>
<th>Upgrades affected</th>
<th>Issue</th>
<th>Workaround (if available)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.x to 4.0.y, 4.1.y, and 4.2.y</td>
<td>After migration, Query Workbench settings are reset for collections other than default.</td>
<td>For collections other than default, print or write down all Query Workbench settings prior to upgrading. After the upgrade, redo the Query Workbench configurations by hand for collections other than default.</td>
</tr>
<tr>
<td>3.1.x to 4.0.y, 4.1.y, and 4.2.y</td>
<td>Users can't log in after migration of a SAML realm.</td>
<td>This is being investigated. No workaround is available at this time.</td>
</tr>
<tr>
<td>3.1.x to 4.0.y, 4.1.y, and 4.2.y</td>
<td>After migration, the query pipeline default (created during migration for the app default) is associated with collections in the Query Workbench, instead of the query pipelines that were associated with the collections before migration (typically &lt;collection&gt;-default).</td>
<td>In Query Workbench, for each collection with which the default query pipeline is associated, load the correct query pipeline, and then click Save.</td>
</tr>
<tr>
<td>3.1.x to 4.0.y, 4.1.y, and 4.2.y</td>
<td>Migration of index profiles and query profiles modifies the profiles. For more information, see Migration of index profiles and query profiles.</td>
<td>After migration, you might need to adjust the pipeline references of index profiles and query profiles by hand, and/or create new index profiles and query profiles.</td>
</tr>
<tr>
<td>3.1.x to 4.2.x</td>
<td>Migrated JDBC blobs are’t linked to apps.</td>
<td></td>
</tr>
<tr>
<td>Note</td>
<td>Apps function fine without these links. You only need to create links if you want the JDBC blobs to be visible under the &quot;In Current App&quot; tab in Object Explorer.</td>
<td></td>
</tr>
</tbody>
</table>

Note: Apps function fine without these links. You only need to create links if you want the JDBC blobs to be visible under the "In Current App" tab in Object Explorer.
<table>
<thead>
<tr>
<th>Upgrades affected</th>
<th>Issue</th>
<th>Workaround (if available)</th>
</tr>
</thead>
<tbody>
<tr>
<td>After migration, link JDBC blobs to apps manually, using either the Fusion UI or the Fusion API.</td>
<td>All upgrades except for 3.1.x to 3.1.y</td>
<td>The managed-schema data in the Solr configuration can't exceed 1 MB in size.</td>
</tr>
<tr>
<td><strong>Fusion UI:</strong> <em>(repeat for all apps, and all JDBC blobs in each app)</em> In Object Explorer, locate a JDBC blob under &quot;In Any App&quot; or under &quot;In No Apps&quot;. Hover over the object, click the App icon, and then click <strong>Add to this app</strong>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Fusion API:</strong></td>
<td></td>
<td>This is an example of a failure message:</td>
</tr>
<tr>
<td>curl -u admin:password &quot;<a href="http://localhost:8764/api/blobs?resourceType=driver:jdbc">http://localhost:8764/api/blobs?resourceType=driver:jdbc</a>&quot;</td>
<td>2. <em>(Repeat for all JDBC blobs)</em> To link a JDBC blob to all apps, use this API command:</td>
<td></td>
</tr>
<tr>
<td>curl -u admin:password -H &quot;Content-Type:application/json&quot; -X PUT -d '{ &quot;subject&quot;: &quot;group:_shared&quot;, &quot;object&quot;: &quot;blob:JDBC_blob_ID&quot;, &quot;linkType&quot;: &quot;hasPart&quot;, &quot;originator&quot;: &quot;migration&quot; }' &quot;<a href="http://localhost:8764/api/links">http://localhost:8764/api/links</a>&quot;</td>
<td>Remove unneeded elements from the schema, and then restart the upgrade from the beginning.</td>
<td>Migrated signal-aggregation jobs retain their schedules (by design). To avoid job timeouts, the default schedule for signal-aggregation jobs was increased from every two minutes to every 15 minutes in Fusion 4.1.0.</td>
</tr>
<tr>
<td>3.1.x and 4.0.x to 4.1.y and 4.2.y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scale a Fusion Cluster

To scale a Fusion cluster, you can add new Fusion nodes, add new dedicated indexing nodes, or move Fusion to new nodes.
Adding a new Fusion node to an existing cluster

Follow these steps to add a new node to an existing Fusion cluster:

<table>
<thead>
<tr>
<th>Note</th>
<th>If you’re running embedded zookeepers from Fusion in an ensemble for your cluster, ensure that the you’re running an odd number of zookeepers for your environment after addition of the new node.</th>
</tr>
</thead>
</table>

1. Stop Fusion on all nodes in the cluster.

   This ensures that there is no data inconsistency between the instances when the new node comes up.

2. Decompress the new copy of Fusion and place it in the desired directory.

3. Configure the `fusion.properties` file to match your requirements.

   If you will also run the embedded ZooKeeper, add the new node’s IP/hostname and port to the `default.zk.connect` string and copy this change to all other instances in your cluster. Configure the memory and other JVM options for the Fusion modules, then save the file.

4. If embedded ZooKeepers are used in your cluster and you intend to start ZooKeeper on this node, then follow the additional steps below. If not, then you are ready to start all nodes in the cluster.

   a. Copy the `$FUSION_HOME/conf/zookeeper/zoo.cfg` file from one of the existing nodes to the new node, overwriting the default file.

   b. Add the entry for the new ZooKeeper to the server list in the `zoo.cfg` file.

      The entry format is `server.x=IP:port:port`. For example, if this is the 5th node, then the new entry in `zoo.cfg` is `server.5=IP:port:port`.

   c. Create a `zookeeper` folder under `$FUSION_HOME/data`.

   d. Create a new `myid` file in `$FUSION_HOME/data/zookeeper`.

      The contents of this file must be an integer equal to the number of the new ZooKeeper node in the ensemble. For example, if the new node will be the 5th node in your ZooKeeper ensemble, then the `myid` file should contain the value “5”.

   e. Copy the `$FUSION_HOME/data/zookeeper/version-2` directory from one of the existing nodes to the new node, overwriting the default directory.

   f. Modify the connect string for the default search cluster:

   g. Start ZooKeeper on all nodes.

   Next, you will need the `zkcli` script, located in `$FUSION_HOME/apps/solr-dist/server/scripts/cloud-scripts`. Use `zkcli.sh` for Unix or `zkcli.bat` for Windows. The examples below use the Unix script.

   h. Download the default search cluster file:

      ```
      ./zkcli.sh -z <zk1>::<port1>,<zk2>::<port2>,... -cmd getfile <path_to_default_cluster> <path_to_dump_file>.json
      ```
The path will differ depending on your Fusion version:

- **2.4.x**: `/lucid/search-clusters/default`
- **3.x**: `/lwfusion/<fusion_version>/core/search-clusters/default`

For example:

```
./zkcli.sh -z localhost:9983 -cmd getfile /lwfusion/3.1.2/core/search-clusters/default
default_search_cluster.json
```

i. In the downloaded JSON file, find the `connectString` key and replace the old IP value with the IP of the new Fusion node.

<table>
<thead>
<tr>
<th>Note</th>
<th>Be sure to specify the chroot if your cluster is configured to use it.</th>
</tr>
</thead>
</table>

For example:

```
{
    "id" : "default",
    "connectString" : "localhost:9983/lwfusion/3.1.2/solr",
    "zkClientTimeout" : 30000,
    "zkConnectTimeout" : 60000,
    "cloud" : true,
    "bufferFlushInterval" : 1000,
    "bufferSize" : 100,
    "concurrency" : 10,
    "authConfig" : {
        "authType" : "none"
    },
    "validateCluster" : true
}
```

j. Upload the modified search cluster file:

```
./zkcli.sh -z <zk1>:<port1>,<zk2>:<port2>,... -cmd putfile <path_to_default_cluster>
<path_to_dump_file>.json
```

For example:

```
./zkcli.sh -z localhost:9983 -cmd putfile /lwfusion/3.1.2/core/search-clusters/default
default_search_cluster.json
```

5. Start Fusion on all nodes in the cluster.
Adding an indexing node to a Fusion cluster

If you need more capacity for indexing, you can add nodes dedicated to indexing. To do this, you add a new Fusion node, configure it to only run the Solr service, then allocate replicas of your collections to the new node.

1. Install the Fusion package on the new node.

2. Edit `fusion.properties` as follows:
   a. Edit `group.default` to include only the Solr service.
      
      For example, change
      ```
      group.default = zookeeper, solr, api, connectors-rpc, connectors-classic, admin-ui, proxy, webapps
      ```
      to
      ```
      group.default = solr
      ```
   b. Uncomment `default.zk.connect` and point it to the cluster's ZooKeeper instances.
      
      For example, change
      ```
      # default.zk.connect = localhost:9983
      ```
      to
      ```
      default.zk.connect = 172.23.1.1:9983, 172.23.1.2:9983, 172.23.1.3:9983
      ```
   c. Save the file.

3. Start Fusion on the new node:
   ```
   bin/fusion start
   ```

   At this point, the new node is added to the cluster. No indexing takes place on the new node yet.

4. Allocate one or more collection replicas to this node:
   b. Click Collections
   c. Select a collection to replicate on the new indexing node.
   d. Click Shard: shard1 (or another shard if you have more than one for this collection).
   e. Click add replica.
   f. From the Node drop-down list, select the new node.
   g. Click Create Replica.
To verify that the collection is being replicated, you can click **Cloud** and view the replicas.

| Tip | Consider whether secondary collections should also be replicated. For example, consider adding replicas for the signals and aggregations collections associated with the main collections that you are replicating. |
Moving Fusion from one node to another

1. Stop Fusion on all nodes in the cluster.
   This ensures that there is no data inconsistency between the instances when the new node comes up.

2. Compress the Fusion node you wish to move.

3. Copy the compressed file to the destination.

4. Starting with step 2, follow the instructions above for adding a new node.
Migrate Fusion Objects

You can use the Fusion UI and the Objects API to migrate collections and related objects, including your searchable data, configuration data, pipelines, aggregations, and other objects on which your collections depend. You can also migrate entire apps.

| Important | When upgrading from one Fusion version to a later one, you must use the migrator to migrate objects. The migrator automates the process of translating relevant objects to the new version. Exporting an object from one Fusion version and importing it into a different Fusion version isn’t supported. |

You might need to migrate objects in the following circumstances:

- When migrating data from development environments into testing and production environments
- To back up data, so you can restore it after an incident of data loss
- When the migrator script wasn’t able to migrate an object automatically

| Caution | When you export and import objects to migrate them, make sure you that any data gets to where it is going before deleting the sources. |
## Migration approaches

Several approaches are available for migrating Fusion objects. This table summarizes the approaches.

<table>
<thead>
<tr>
<th></th>
<th>Export an app</th>
<th>Import an app</th>
<th>Export an object</th>
<th>Import an object</th>
<th>Add an object to an app</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fusion UI</strong></td>
<td>App configuration</td>
<td>Launcher (entire app)</td>
<td>–</td>
<td>–</td>
<td>Object Explorer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>App configuration (combine apps)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Objects API</strong> (endpoints)</td>
<td>GET from export endpoint</td>
<td>POST to import endpoint</td>
<td>GET from export endpoint</td>
<td>POST to import endpoint</td>
<td>–</td>
</tr>
</tbody>
</table>

For more information about using the Objects API to export and import objects, see Objects API.

The remainder of this topic describes approaches in the Fusion UI.

Use the parts of the Fusion UI indicated in the table to export and import apps and specific objects. Exporting creates a zip file. To import, you select a data file and possibly a variable file.

The approach with Object Explorer differs. With Object Explorer, you can add objects from other apps (or that are not linked to any apps) to the currently open app.
Export an app with the Fusion UI

How to export an app with the Fusion UI

1. Navigate to the launcher.
2. Hover over the app you want to export and click the Configure icon.

3. In the app config window, click Export app to zip:

This downloads a zip file that you can import into other instances of Fusion Server.
Import an app with the Fusion UI

How to import an app with the Fusion UI

1. Navigate to the launcher.
2. Click Import app.

3. Under Data File, click Choose File and select the zip file containing the app you want to import.
4. If your app has usernames and passwords in a separate file, select it under Variables File.

<table>
<thead>
<tr>
<th>Tip</th>
<th>If the variables file is included inside the zip file, then you don’t need to upload it separately.</th>
</tr>
</thead>
</table>

5. In some cases, you can edit parameter values to use the new values in the imported app. If this is the case, Fusion displays a dialog box that lets you edit the parameter values.
Make desired changes, and then click **Import**.
Copy an app

To copy an app from one deployment to a different one, export the app on the source deployment, and then import the app on the target deployment.
Import objects into an app

You can import objects into the currently open app.

How to import objects into an open app

1. In the Fusion launcher, click the app into which you want to import objects.
   
The Fusion workspace appears.

2. In the upper left, click System > Import Fusion Objects.
The Import Fusion Objects window opens.

3. Select the data file from your local filesystem.
   If you are importing usernames and passwords in a separate file, select it under Variables File.

4. Click Import.

   If there are conflicts, Fusion prompts you to specify an import policy:

   - Click **Overwrite** to overwrite the objects on the target system with the ones in the import file.
   - Click **Merge** to skip all conflicting objects and import only the non-conflicting objects.
   - Click **Start Over** to abort the import.

Fusion confirms that the import was successful:
5. Click **Close** to close the Import Fusion Objects window.
Add an object to an app

You can add objects present in other apps (or in no apps) to the open app. Some objects are linked to other apps. You can also add those directly to an app.

- **Add an object to an app** – While in the Fusion workspace for the app to which you want to add an object, open Object Explorer and click **In Any App**. Search for or browse to the object you want to add. Hover over the object, click the App icon, and then click **Add to this app**.

- **Add an object to an app directly** – In cases when an object is linked to an app, but is not linked directly to the app (it is linked via some dependency), you can add the object to an app directly.

While in the Fusion workspace for the app to which you want to add an object directly, open Object Explorer and click **In Any App**. Search for or browse to the object you want to add. Hover over the object, click the App icon, and then click **Add to this app directly**.