Lobber Deluge Plugin

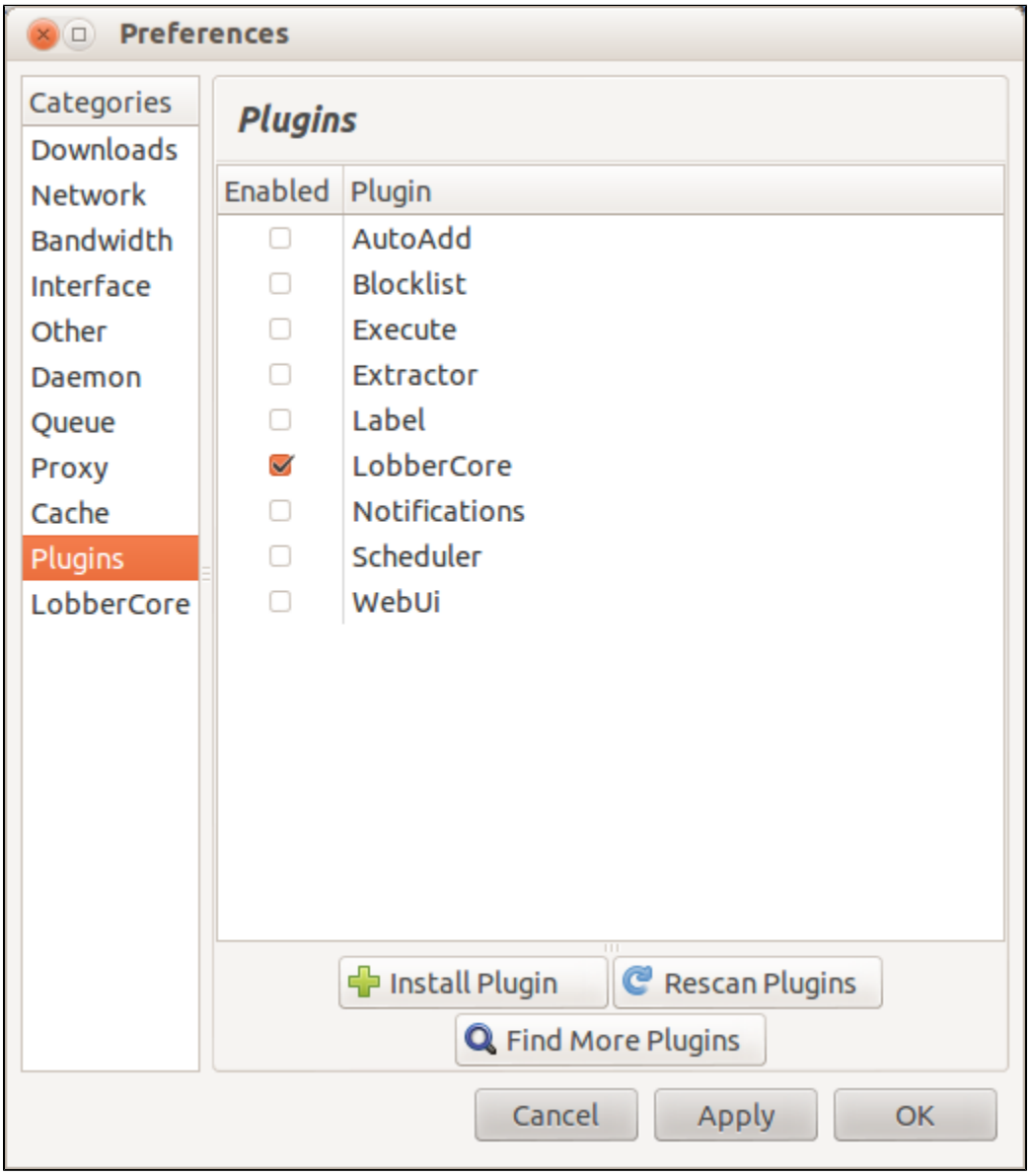
Page index

How to install

- What you need:
  - Deluge
  - Lobber-deluge-core

How you install Deluge differs from operating systems but it is available for Mac OSX, Windows and Linux. We recommend that you use the latest version available.

When you have installed Deluge you should download our plugin and then use the Deluge settings to add a new plugin. In Deluge preferences, under plugin, you should be able to click install
plugin and locate the .egg file that you have downloaded. Make sure that the downloaded file is named LobberCore-0.1-py2.7.egg or LobberCore-0.1-py2.6.egg matching your installed version of Python.



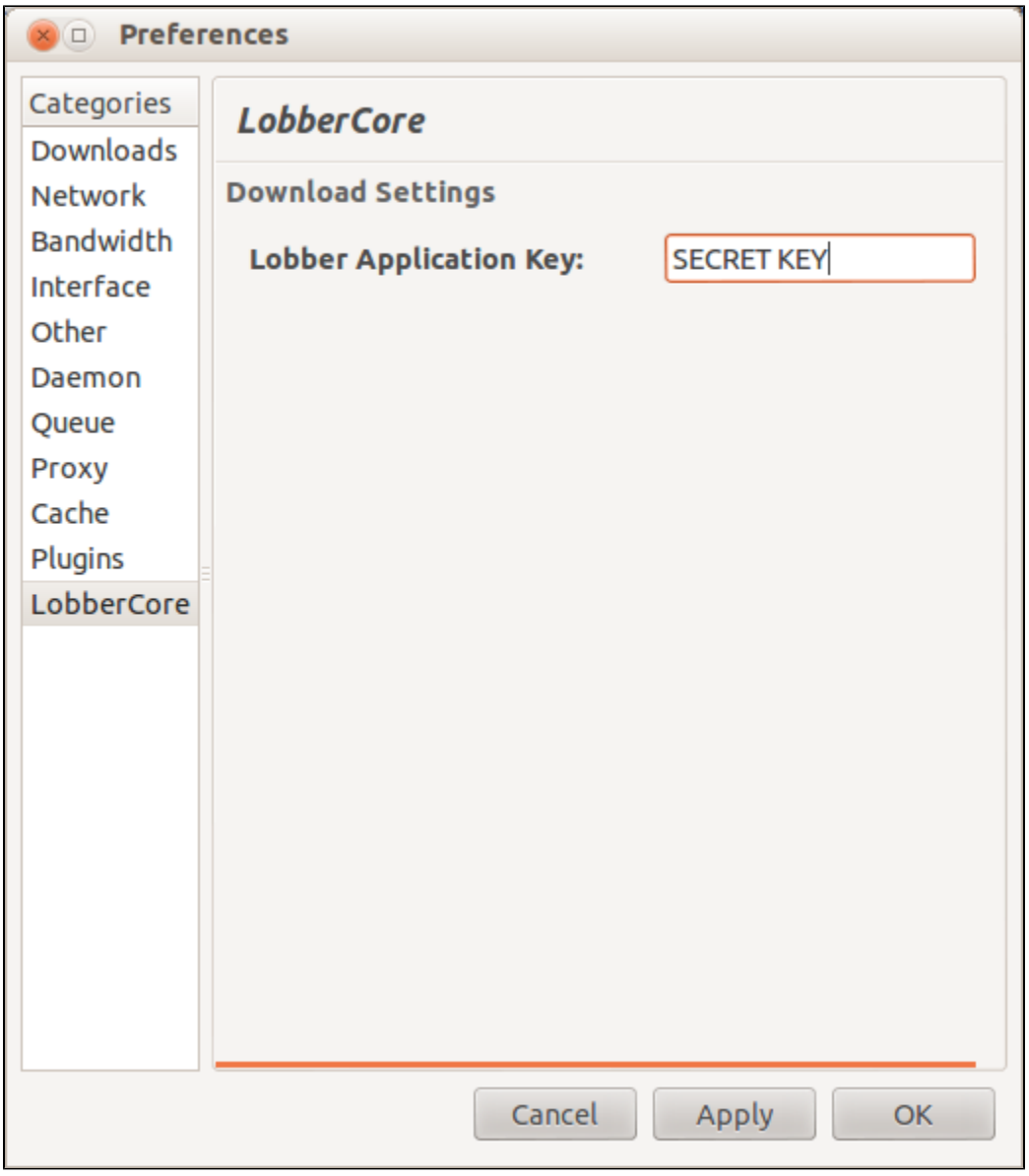
After you installed the plugin you should be able to check the box beside LobberCore and a new choice (LobberCore) should be visiable at the bottom of the left column.

Common problems
Nothing happens when I check the box beside LobberCore - The downloaded egg-file probably has the wrong name, it should be named LobberCore-0.1-py2.6.egg or LobberCore-0.1-py2.7.egg depending on the installed version of Python on your system.

Running a storage node

Basic settings

- First create a Lobber API key for the groups you want to download torrents for.
- Add the key to the LobberCore settings in the Deluge settings dialog.



Done. The plugin will now check Lobber every 15 minutes to see if there are any torrents the key have read access to, if any new torrents are found the plugin will download them and start them in Deluge.

Advanced settings

If you want to make some advanced settings you have to locate the file lobbercore.conf, it should be in $HOME/.config/deluge/ for the user that is running Deluge.

Standard configuration file

```
{
  "file": 1,
  "format": 1
}{
  "lobber_key": "YOUR SECRET KEY",
  "tracker_host": "https://dev.lobber.se"
  "feed_url": "https://dev.lobber.se/torrent/all.json",
  "download_dir": "",
  "unique_path": false,
  "proxy_port": 7001,
  "minutes_delay": 15,
  "remove_data": false,
  "monitor_torrents": false,
  "torrent_evaluator": "total_seeders",
  "min_seeders": 1,
  "max_seeders": 2,
  "removed_torrents": [],
}
```

Setting	Value	Description
lobber_key	Key string	Your Lobber API key
tracker_host	https://url[:port]	The URL to the Lobber tracker you want to use.
feed_url	https://url/json-feed.json	The URL to the Lobber json feed you want to monitor for new torrents.
download_dir	Directory path	Path to the directory you want to save your downloaded data in. If this is empty ("") the standard Deluge setting will be used. **Ending slash (/) is important.**
unique_path	true/false	If set to true the download directory will be appended by a new directory with the torrents hash in which the data will be downloaded.
proxy_port	An unused local port	This is the port that will be used for the localhost proxy.
minutes_delay	Number of minutes	This is the delay between the plugins checks for new torrents.
remove_data	true/false	If set to true the torrents data will be removed if a torrent is removed by the plugin.
torrent_evaluator	Name of the torrent evaluator to use.	There is only one torrent evaluator right now, total_seeders. This evaluator will check how many seeders a torrent has every 15 minutes. If the torrent has more than min_seeders the torrent will be paused, equal or lower than min_seeders resumed and over or equal to max_seeders removed.
removed_torrents	List of removed torrents	This setting is used internally to remember which torrents that should not be added again.

Running a Lobber web app storage node (Ubuntu)

If you set up Deluge to support the Lobber web application data http upload functionality you are probably want to run it headless.

Install deluged.

```
sudo apt-get install deluged
```

Create a Deluge user:

```
sudo adduser --disabled-password --system --home /var/lib/deluge/ --gecos "Deluge daemon" --group deluge
```

Create /etc/init.d/deluged with the following content.

deluged

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          deluged
# Required-Start:    $local_fs $remote_fs
# Required-Stop:     $local_fs $remote_fs
# Should-Start:      $network
# Should-Stop:       $network
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start or stop the Deluge BitTorrent daemon.
# Description:       Start or stop the Deluge BitTorrent daemon.
### END INIT INFO

# Authors: Tanguy Ortolo <tanguy+debian@ortolo.eu>,
# Cristian Greco <cristian@regolo.cc>

PATH=/sbin:/usr/sbin:/bin:/usr/bin
DESC="Deluge BitTorrent Daemon"
NAME="deluged"
DAEMON=/usr/bin/$NAME
DAEMON_ARGS="-d -c /var/lib/deluge/.config/deluge -l /var/log/deluged.log -L info"
USER=deluge
MASK=0027
PIDFILE=/var/run/$NAME.pid
SCRIPTNAME=/etc/init.d/$NAME

# Exit if the package is not installed
[ -x "$DAEMON" ] || exit 0

ENABLE_DELUGED=1

# Read configuration variable file if it is present
[ -r /etc/default/$NAME ] && . /etc/default/$NAME

# Load the VERBOSE setting and other rcS variables
[ -f /etc/default/rcS ] && . /etc/default/rcS

# Define LSB log_* functions.
# Depend on lsb-base (>= 3.0-6) to ensure that this file is present
# and status_of_proc is working.
. /lib/lsb/init-functions
#
# Function that starts the daemon/service
#
do_start()
{
  if [ $ENABLE_DELUGED != 1 ]; then
    log_progress_msg "Not starting ${DESC} ${NAME}, disabled in /etc/default/${NAME}"
  else
    # Return
    #   0 if daemon has been started
    #   1 if daemon was already running
    #   2 if daemon could not be started
    start-stop-daemon --start --background --quiet --pidfile $PIDFILE --exec $DAEMON \
      --chuid $USER --umask $MASK --test > /dev/null \
      || return 1

    start-stop-daemon --start --background --quiet --pidfile $PIDFILE --make-pidfile --exec $DAEMON \
      --chuid $USER --umask $MASK -- $DAEMON_ARGS \
      || return 2
  fi
}
#
# Function that stops the daemon/service
#
do_stop()
{
```

```
  # Return
  #   0 if daemon has been stopped
  #   1 if daemon was already stopped
  #   2 if daemon could not be stopped
  #   other if a failure occurred

  start-stop-daemon --stop --quiet --retry=TERM/30/KILL/5 --pidfile $PIDFILE
  RETVAL="$?"
  [ "$RETVAL" = "2" ] && return 2

  rm -f $PIDFILE
  return "$RETVAL"
}
case "$1" in
  start)
    [ "$VERBOSE" != no ] && log_daemon_msg "Starting $DESC" "$NAME"
    do_start
    case "$?" in
      0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
      2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
    esac
    ;;
  stop)
    [ "$VERBOSE" != no ] && log_daemon_msg "Stopping $DESC" "$NAME"
    do_stop
    case "$?" in
      0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
      2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
    esac
    ;;
  status)
    status_of_proc "$DAEMON" "$NAME" && exit 0 || exit $?
    ;;
  restart|force-reload)
    log_daemon_msg "Restarting $DESC" "$NAME"
    do_stop
    case "$?" in
      0|1)
        do_start
        case "$?" in
          0) log_end_msg 0 ;;
          1) log_end_msg 1 ;; # Old process is still running
          *) log_end_msg 1 ;; # Failed to start
        esac
        ;;
      *)
        # Failed to stop
        log_end_msg 1
        ;;
    esac
    ;;
  *)
    echo "Usage: $SCRIPTNAME {start|stop|status|restart|force-reload}" >&2
    exit 3
    ;;
esac
```

Create a log file.

```
touch /var/log/deluged.log
chown deluge:deluge /var/log/deluged.log
```

Make deluged star on boot.

```
sudo update-rc.d deluged defaults
```

Start deluged.

```
sudo /etc/init.d/deluged start
```

Download LobberCore-0.1-py2.7.egg or LobberCore-0.1-py2.6.egg and put the egg-file in /var/lib/deluge/.config/deluge/plugins/.

Edit /var/lib/deluge/.config/deluge/core.conf to start the plugin.

```
.
.
"enabled_plugins": [
    "LobberCore"
  ],
.
.
```

Restart deluged to load the plugin.

```
sudo /etc/init.d/deluged restart
```

Last but not least edit the newly created file /var/lib/deluge/.config/deluge/lobbercore.conf so it looks like this.

Web app storage node configuration file

```
{
  "file": 1,
  "format": 1
}{
  "torrent_evaluator": "total_seeders",
  "monitor_torrents": true,
  "min_seeders": 1,
  "download_dir": "/var/www/lobber/seeding/",
  "proxy_port": 7001,
  "removed_torrents": [],
  "unique_path": true,
  "feed_url": "https://dev.lobber.se/torrent/all.json",
  "max_seeders": 2,
  "remove_data": true,
  "lobber_key": "",
  "minutes_delay": 1,
  "tracker_host": "https://dev.lobber.se"
}
```

download_dir should match the SEEDING_DIR in Lobbers Django settings.py. The deluge user have to have permission to write in the directory.
Deluge should now seed any torrent until there is another client seeding it and then delete the torrent with data.

Common problems
Deluge or the plugin doesn't start - Check the configuration files permissions, do the deluge user have permission to read and write them?