Instructions for Free-Flow on Linux

26.3.2015, Pete Arnold

Contents

0. Supported Systems and System Requirements

Free-Flow is supported for the following Linux distributions:-

1. Debian 7.5 / kernel 3.2.0-4
2. Ubuntu 14.04 / kernel 3.13.0-24
3. CentOS 6.5 / kernel 2.6.32-431
4. Red Hat 6.5 / kernel 2.6.32-431
5. openSUSE 13.1 / kernel 3.11.6-4 / GDM /Gnome (There may be a drag issue with KDM / KDE)

There may be an outstanding issue in KDM/KDE in which the mouse can drag windows but not icons. This has only been seen on OpenSUSE/KDM but KDM has not been tested on any other distribution. OpenSUSE/GDM does not suffer from this problem.

In all cases, the scripts assume a clean, default installation. The scripts will install (or update) various tools required to process files and build the driver. Once installed, the scripts does not remove the tools in case they were already installed. If you wish to clean the system after installation, please record the packages that are installed and remove manually when required.

Free-Flow is expected to install and operate on most modern Linux distributions using techniques similar to those outlined for the supported variants. However, it is not possible to test and maintain code and scripts for all known Linuxes.

1. Preliminaries

The following steps must be peformed in order to set-up or remove any of the Free-Flow components on any Linux system.

(1) Copy the free-flow folder to a convenient place in the file system.
(2) Open a terminal application (e.g. Terminal or XTerm). Under Debian you can
open a Root Terminal combining this and the next step (skip to (4)).
(3) Become a superuser (or prefix all the following commands with sudo):

```
su
<enter superuser password>
```

To set a superuser password type:

```
sudo passwd root
<enter new superuser password>
<confirm new superuser password>
```

Note that by becoming a superuser, you give yourself permission to perform operations that can have catastrophic effects on your operating system and computer. If you have not been warned of the risks please take some time to familiarise yourself with warnings about being a root user you will find in many places on the internet.
(4) Change to the free-flow directory:

```
cd <path-to-free-flow>
```

2. Background Information

2.0 Single-monitor set-up

To use the Free-Flow mouse in a single-monitor system only requires the configuration of the device as a mouse. There is no need to install a driver. The correct configuration mechanism will depend on the distribution and whether it has been configured to use a particular mechanism. The 'simo' script will attempt to determine the correct method and provide a suitable configuration file.

2.1 Multiple-monitor set-up

To use the Free-Flow mouse across multiple-monitors, you will need to associate a new driver (hid_adder) with the Free-Flow mouse device. As the driver is not available (yet) in the linux kernel, it needs to be compiled as a module, registered with the kernel and, when required, associated (or bound) to the Free-Flow mouse.

2.1.1 Compilation

The compilation and registration of the driver has to be done once only and is done specifically for the kernel version of your system. The script needs to compile with the kernel headers (which it will download if needed) and the kernel library files already used by your kernel. These are referred to by a version number given by:

```
uname -r
```

However, some distributions provide header files with additional version numbers and may upgrade the headers over time, so you may need to specify the correct files to use (this can be done with the -s/--source parameter in makemumo). If you upgrade or change your kernel version you may need to rebuild the Free-Flow driver module.

2.1.2 Binding

The binding of the driver to the device needs to be done whenever the device is attached or the windowing system started. Ideally this will be done automatically e.g. by including/calling the binding script in a start-up script. However, this is not always successful and the binding script may need to called manually.

2.1.3 Use

Once the driver is installed and bound to the device, the position of the mouse is determined by the device and the layout of the screens configured on the device. The screen layout defined under X is irrelevant. Note that the config script uses xrandr to set and re-arrange the monitor parameters. If these settings are not suitable, please edit the scripts to either make them suitable or remove them altogether (they may not be necessary if you already have scripts setting up your monitors).

2.2 Multiple-monitor uninstall

When uninstalling, it is essential to remove the calls to the mumo and unmumo scripts as, if present but unexecutable, the system may refuse to log you into the window manager.

3. Configuration and Use for Single-Monitor Systems

The configuration for single-monitor systems consists of providing a udev rule
for most systems. Red Hat (and CentOS) still use HAL which requires a different
(but equivalent) file.

3.1 Set-up

3.1.1. Debian 7.5 / 3.2.0-4
Ubuntu 14.04 / 3.13.0-24
openSUSE 13.1 / 3.11.6-4

The following steps are required to use the Free-Flow mouse on a single-monitor system:

(1) The single-monitor-system script is 'simo'. You need to make this executable by:

```
chmod +x simo
```

(2) Execute the script:

```
./simo
```

This should inform you that it has found a udev directory and installed a suitable udev rule for the free-flow mouse.
You will need to unplug and replug the device to trigger the rule.
The free-flow mouse should now be operational and will be available whenever you login to the X windows system.

3.1.2. CentOS 6.5 / 2.6.32-431
Red Hat 6.5 / 2.6.32-431

Redhat distributions seem to give copied files executable privileges. If you find that this is not the case, please refer to the chmod step listed for other distributions.

The following steps are required to use the Free-Flow mouse on a single-monitor system:
(1) Execute the script:

```
./simo
```

This should inform you that it has forced the use of HAL rules for redhat. This is necessary as the redhat distribution contains udev, HAL and Xorg directories, but uses only HAL for the free-flow mouse.
You will need to unplug and replug the device to trigger the rule.
The free-flow mouse should now be operational and will be available whenever you login to the X windows system.

3.2 Removal

3.2.1. Debian 7.5 / 3.2.0-4
Ubuntu 14.04 / 3.13.0-24
openSUSE 13.1 / 3.11.6-4

The following steps are required to remove the Free-Flow mouse on a single-monitor system.

(1) The single-monitor-system script is 'desimo'. You need to make this executable by:

```
chmod +x desimo
```

(2) Execute the script:

```
./desimo
```

This will remove all Free-Flow configuration or rule files in any of the expected directories.
You will need to unplug and replug the device to trigger the rule.
[Note that on Debian 6, the unplugging of the device often causes the user to be logged-out and returned to the logon screen.]

3.2.2. CentOS 6.5 / 2.6.32-431
Red Hat 6.5 / 2.6.32-431

The following steps are required to remove the Free-Flow mouse on a single-monitor system.

(1) Execute the script:

```
./desimo
```

This will remove all Free-Flow configuration or rule files in any of the expected directories.
You will need to unplug and replug the device to trigger the rule.

4. Installation and Use for Multiple-Monitor Systems

Using the Free-Flow mouse on a multi-monitor system requires a simple driver. The driver needs to be compiled once for each system and then bound and configured when used. Binding associates the driver with the device and configuration copies the co-ordinates of the monitors to the configfs file system where the driver can access them. In order to make the Free-Flow mouse available automatically, the bind and configure scripts can be added to the system's autostart files.

Each of the supported distributions do this with varying degrees of simplicity.

4.1. Debian 7.5 / 3.2.0-4

4.1.1 Driver compilation

The following steps are required to build the Free-Flow mouse on a multiple-monitor system:

(1) The multi-monitor-system build script is 'makemumo'. You need to make this executable by:

```
chmod +x makemumo
```

(2) Execute the script:

```
./makemumo
```

This will execute the 'simo' script and then check and, where necessary, install any tools it needs that aren't already available. It uses the standard installation mechanism which will attempt to install either from the installation media or via the internet. You will be asked to insert any media and to confirm that you wish to install the packages. If you do not install the packages, the build will not complete and multi-monitor Free-Flow won't install. The driver will then be compiled against the current kernel and the resulting module installed in the system.
[Note that on Debian 6, the usbhid driver quirk parameters are not necessary and, if the stop/start of usbhid is attempted, will cause the current user to be logged out.]

(3) [Note: this step isn't needed on Debian 6: skip to step 4.]
There is a problem with the xrandr configuration which forces the screen to be a maximum of 1920 x 1920 pixels. This can be remedied by a new xorg.conf. A sample file is included which, since it includes hardware dependent definitions, may require modification for your system. Save any existing file:

```
mv /etc/X11/xorg.conf /etc/X11/xorg.conf.old
```

Replace with the new file:

```
cp xorg.conf /etc/X11/
```

(4) Remove the DVD from the drive if still present.

(5) Reboot the computer.

4.1.2 Driver use

Both Debian 6.0 and 7.5 seem to autostart once the user logs in. No further action is required to use Free-Flow over multiple monitors. If manual start/stop of the driver is required, it can be started using:

```
./mumo
```

(use the -d option display the output rather than sending it to ~/mumo.log) and stopped using:

```
./unmumo
```

4.1.3 Multiple monitor de-set-up

The driver can be removed by:

(1) Run:

```
./unmumo
./demumo
```

(2) If required, delete the directory containing the driver files and scripts.

4.2. Ubuntu 14.04 / 3.13.0-24

4.2.1 Driver compilation

The following steps are required to build the Free-Flow mouse on a multiple-monitor system:

(1) The multi-monitor-system build script is 'makemumo'. You need to make this executable by:

```
chmod +x makemumo
```

(2) Execute the script:

```
./makemumo
```

This will execute the simo script and then check and, where necessary, install any tools it needs that aren't already available. It uses the standard installation mechanism which will attempt to install either from the installation media or via the internet. You will be asked to insert any media and to confirm that you wish to install the packages. If you do not install the packages, the build will not complete and multi-monitor Free-Flow won't install. The driver will then be compiled against the current kernel and the resulting module installed in the system.

4.2.2 Driver use

If the driver doesn't automatically start, it can be started using:

```
./mumo
```

(using the -d option display the output rather than sending it to ~/mumo.log) and stopped using:

```
./unmumo
```

4.2.3 Multiple monitor de-set-up

The driver can be removed by:

(1) Run:

```
./unmumo
./demumo
```

(2) If required, delete the directory containing the driver files and scripts.

4.3. CentOS 6.5 / 2.6.32-431
Red Hat 6.5 / 2.6.32-431

4.3.0 General

Redhat distributions seem to give copied files executable privileges. If you find that this is not the case, please refer to the chmod step listed for other distributions.

If the installation does not enable the network connections automatically, you may find that the 'makemumo' script fails (cannot resolve repository host) - in this case, you will need to enable a network connection in the Gnome desktop menus: System - Preferences - Network Connections (select connect automatically).

Since CentOS requires that the config script is run as a user, there is the possibility that it may fail when it tries to write to the log file is that log file was created (and so is owned) by root. If the multi-monitor mouse fails to autostart when logging in, check whether ~/mumo.log is owned by the user or by root. If it is owned by root either chown it to the user or delete it.

Note that Red Hat will require a registered subscription to enable downloading of additional software.

4.3.1 Driver compilation

The following steps are required to build the Free-Flow mouse on a multiple-

monitor system:

(1) Execute the script:

```
./makemumo
```

This will execuate the simo script and then check and, where necessary, install any tools it needs that aren't already available. It uses the standard installation mechanism which will attempt to install either from the installation media or via the internet. You will be asked to insert any media and to confirm that you wish to install the packages. If you do not install the packages, the build will not complete and multi-monitor Free-Flow won't install. The driver will then be compiled against the current kernel and the resulting module installed in the system.

(2) Reboot the computer.

4.3.2 Driver use

If manual start/stop of the driver is required, it can be started using:

```
./mumo
```

(using the -d option display the output rather than sending it to ~/mumo.log) and stopped using:

```
./unmumo
```

4.3.3 Multiple monitor de-set-up

The driver can be removed by:
(1) Run:

```
./unmumo
./demumo
```

(2) If required, delete the directory containing the driver files and scripts.

4.4. openSUSE 13.1 / 3.11.6-4

4.4.1 Driver compilation

The following steps are required to build the Free-Flow mouse on a multiple-monitor system:

(1) The multi-monitor-system build script is 'makemumo'. You need to make this executable by:

```
chmod +x makemumo
```

(2) Execute the script:

```
./makemumo
```

This will execuate the simo script and then check and, where necessary, install any tools it needs that aren't already available. It uses the standard installation mechanism which will attempt to install either from the installation media or via the internet. You will be asked to insert any media and to confirm that you wish to install the packages. If you do not install the packages, the build will not complete and multi-monitor Free-Flow won't install. The driver will then be compiled against the current kernel and the resulting module installed in the system.

4.4.2 Build problems

The build procedure above should handle the following issues. However, since the openSUSE build is particularly complex, should you encounter any problems, these may help:

4.4.2.1 Kernel version

openSUSE 13.1 originally included kernel 3.11.6 which has a bug preventing the use of Free-Flow. This has been rectified in later kernels (tested at 3.15.6). It may, therefore be necessary to upgrade your kernel. The 'makemumo' script
will offer you this choice. It can also be done manually as follows:

Check the kernel version:

```
uname -r
```

and, if necessary, upgrade the kernel to 3.15.6 or later. This can be done as follows:
(1) Add a suitable zypper repository.

```
zypper ar -f http://download.opensuse.org/repositories/Kernel:/stable/standard/ kernel
```

(2) Refresh the repositories:

```
zypper ref
```

and accept the key as you see fit.

(3) Perform a distribution update using the kernel repository:

```
zypper dup -r kernel
```

You may have to answer some questions about dependencies. Despite the unpleasant-looking nature of 'breaking' the kernel, there seems no choice but to go ahead anyway. The new kernel hasn't shown any faulty behaviour to date but hasn't been tested other than by normal use. This can take some time but should complete without error.

(4) Reboot the computer.

4.4.2.2 Build errors

It is likely that there will be a problem with the build caused by a missing file used by some parts of the kernel build scripts. These scripts aren't needed for the driver build so can be commented out. If, after following the build steps in the next section, the output shows errors such as:

```
scripts/selinux/genheaders/genheaders.c:13:22: fatal error: classmap.h: No such file or directory
```

please make the following edits and try again:
File:

```
/usr/src/linux-<new version number>.<hash>/scripts/Makefile
```

Comment out (place a '#' at the start of the line):
the line containing `+= sortextable`
the line containing `+= selinux.`
It might be preferable to set the kernel configuration to exclude these items.

4.4.3 Driver use

If the driver doesn't automatically start, it can be started using:

```
./mumo
```

(using the -d option display the output rather than sending it to ~/mumo.log) and stopped using:

```
./unmumo
```

4.4.4 Multiple monitor de-set-up

The driver can be removed by:

(1) Run:

```
./unmumo
./demumo
```

(2) If required, delete the directory containing the driver files and scripts.

5. Troubleshooting

5.1. General

CentOS 6.4 complained that it could not resolve a host "mirrorlist.centos.org". In this case, this was due to there being no active network connections which was fixed using:

```
ifup eth1
```

(`ifdown eth1; ifup eth1` might be useful to reset an interface; eth0, the built-in ethernet card was non-functional already on the test PC).

5.2. Single-monitor set-up

The single-monitor set-up will look for directories that contain set-up files for several device configuration methods and will use the first method it finds unless we have previously determined in testing that this is incorrect.

If you have already set-up one of the other methods for device set-up, simo may select the wrong one. This won't cause any harm, but nor will it work. However, you can force simo to select one of the alternative set-up that methods as detailed in Appendix 1.

5.3. Single-monitor de-set-up

It shouldn't be possible for desimo to fail as it merely looks for the Free-Flow configuration files and deletes any it finds in the expected places. If your configuration is such that these files are located in non-standard folders, you will need to delete the files manually.

5.4 Package availability

Occasionally the required build tools are not available (for example, having been moved, retired or mirrors being down). In this case the build will fail. You my need to locate and install these tools manually. The output from the makemumo script will indicate what it was trying to download. You may need to find the package (e.g. an rpm file) yourself, download it and install it. Once you have done so, makemumo should see that the tool is up-to-date and continue. Please make sure that you get the correct package (especially for the kernel-devel headers which need to match the kernel version you have (uname -r) and the platform (uname -m).

5.5 Kernel building

5.5.1 CentOS 5
See http://wiki.centos.org/HowTos/Custom_Kernel and http://wiki.centos.org/HowTos/I_need_the_Kernel_Source

Install the following packages:

```
yum groupinstall "Development Tools"
yum install ncurses-devel
yum install qt-devel (This is only necessary if you wish to use make xconfig instead of make gconfig or make menuconfig.)
yum install hmaccalc zlib-devel binutils-devel elfutils-libelf-devel
```

Get the source:
As an ordinary user, **not root**, create a build tree based on a ~/rpmbuild/ directory:

```
[user@host]$ mkdir -p ~/rpmbuild/{BUILD,BUILDROOT,RPMS,SOURCES,SPECS,SRPMS}
[user@host]$ echo '%_topdir %(echo $HOME)/rpmbuild' > ~/.rpmmacros
```

Install the source package and tools:
As root, install the rpm-build, redhat-rpm-config and unifdef packages:

```
[root@host]# yum install rpm-build redhat-rpm-config unifdef
```

Find the kernel source rpm package in:

- http://vault.centos.org/5.N/os/SRPMS/
- http://vault.centos.org/5.N/updates/SRPMS/

(Replace the "N" with the relevant sub-version number.)
As an ordinary user, **not root**, install the source package by executing:

```
[user@host]$ rpm -i http://vault.centos.org/5.11/os/SRPMS/kernel-2.6.18-398.el5.src.rpm 2>&1 | grep
-v exist
```

Now that the source package and tools are installed, unpack and prepare the source files:

```
[user@host]$ cd ~/rpmbuild/SPECS
[user@host SPECS]$ rpmbuild -bp --target=$(uname -m) kernel.spec
```

The value of $(uname -m) sets the target to the architecture of your current kernel. This is generally accepted, as most people will need either i686 or x86_64 as the target. The kernel source tree will now be found under the ~/rpmbuild/BUILD/kernel*/linux*/ directory.

Configure the kernel:

If you you do not intend to modify the distributed kernel configuration file you may omit this section, as long as you perform the equivalent of the final step:

```
[user@host] $ cp ~/rpmbuild/BUILD/kernel-*/linux-*/configs/* ~/rpmbuild/SOURCES/
```

Modify the kernel specification file:

The kernel specification file should now be modified. The line numbers mentioned in this section relate to the **current** CentOS kernel specification files only.

```
[user@host]$ cd ~/rpmbuild/SPECS/
[user@host SPECS]$ cp kernel.spec kernel.spec.distro
[user@host SPECS]$ vi kernel.spec
```

At line 18, the definition of *buildid* is commented out. Line 74, for CentOS-5. This must be uncommented and given a value to avoid a conflict with your currently installed kernel. Change the line in a similar manner to the example below:

```
%define buildid .your_identifier
```

There should be no space between the "%" and the word "define".

If you have any patches to apply, you need to make reference to them in two places.

Firstly, just before line 613 (just before line 428, for CentOS-5)(which reads "# empty final patch file to facilitate testing of kernel patches"), add your declaration starting with the number 40000, so that *your* patch is not in any danger of conflicting with the RHEL/CentOS kernel patch space. For example:

```
Patch40000: my-custom-kernel.patch
```

Secondly, just before line 935 (which reads, "ApplyOptionalPatch linux-kernel-test.patch"), add a line to apply your patch. For example:

```
ApplyOptionalPatch my-custom-kernel.patch
```

For CentOS-5, just before line 685, add a line to apply your patch. For example:

```
%patch40000 -p1
```

Replace line 929 (not applicable for CentOS-5):

```
cp $RPM_SOURCE_DIR/config-* .
```

with:

```
cp $RPM_SOURCE_DIR/kernel-*.config .
```

Comment out line 933 (not applicable for CentOS-5):

```
#make -f %{SOURCE20} VERSION=%{version} configs
```

The following modification is necessary for CentOS-5. After line 698, there is a block of 25 lines of code that needs to be commented out:

```
#if a rhel kernel, apply the rhel config options
#%if 0%{?rhel}
#  for i in %{all_arch_configs}
#  do
#    mv $i $i.tmp
#    $RPM_SOURCE_DIR/merge.pl $RPM_SOURCE_DIR/config-rhel-generic $i.tmp > $i
#    rm $i.tmp
#  done
```

```
#%ifarch x86_64 noarch
#  for i in kernel-%{kversion}-x86_64*.config
#  do
#    mv $i $i.tmp
#    $RPM_SOURCE_DIR/merge.pl $RPM_SOURCE_DIR/config-rhel-x86_64-generic $i.tmp > $i
#    rm $i.tmp
#  done
#%endif
#%ifarch ppc64 noarch
#  #CONFIG_FB_MATROX is disabled for rhel generic but needed for ppc64 rhel
#  for i in kernel-%{kversion}-ppc64.config
#  do
#    mv $i $i.tmp
#    $RPM_SOURCE_DIR/merge.pl $RPM_SOURCE_DIR/config-rhel-ppc64-generic $i.tmp > $i
#    rm $i.tmp
#  done
#%endif
#%endif
```

Building the new kernel

Start the build:

```
[user@host SPECS]$ rpmbuild -bb --target=`uname -m` kernel.spec 2> build-err.log | tee build-out.log
```

For kernels >= 2.6.18-53.el5, you can add some useful options to the rpmbuild command by using the --with and/or --without flags and associated arguments. The main options to note are:

```
--with baseonly
--with xenonly                <-- I didn't use this one.
--without up
--without xen
--without debug
--without debuginfo
--without fips
--without kabichk
```

For example, to build just the base kernel packages use:

```
--with baseonly --without debug --without debuginfo
```

CentOS-5 only. To build just the xen kernel packages use:

```
--with xenonly --without debug --without debuginfo
```

CentOS-5 only. To build just the PAE kernel packages use:

```
--without up --without xen --without debug --without debuginfo
```

When the build completes, your custom kernel rpm files will be found in the ~/rpmbuild/RPMS/`uname -m`/ directory. Make sure that you install those files, as **root**, using an *rpm -ivh kernel-*.rpm* command. Note: If you have built a kernel version that is older than a currently installed version you will also have to use the *--oldpackage* flag with the rpm command.

UNDER NO CIRCUMSTANCES use an *rpm -Uvh* command to install your kernel as this will update (overwrite) the currently installed version. Hence if you have a problem with your custom kernel, you will not be able to revert to the previous, working, version.

Note that I found I had to use --force to get this to work but apparently without side-effects.

The kernel files will be in /boot with the suffix .your_identifier. Modify (add a separate entry) to the grub.cfg file to allow this kernel to be booted.

Appendices: Script information

Appendix 1: simo

The simo script looks for pre-existing directories in which it can place configuration or rule files for one or more device configuration files. It will install the first one of the following files it finds a place for:

(1) udev rules [-u | --udev]
(2) Xorg InputClass snippets [-x | --xorg]
(3) HAL FDI files [-h | --hal]

If you wish to force a particular file, use the parameters should in brackets above. If you wish to install all of them (that can be found a location), use -a or --all.

Red Hat Enterprise Linux 6.x will be forced to HAL (even though a udev folder exists) as that has been the only method that has worked in our tests without further setting-up.

Appendix 2: desimo

The desimo script looks for pre-existing directories in which configuration or rule files may have been placed. It will delete any of the Free-Flow files in these directories.

Appendix 3: makemumo

The makemumo script checks that the required kernel and tools are available, gets those it needs and compiles and installs the driver for the host's kernel. It will then call the driver loading script (mumo).
Available options are:-

-l <linux-type> where linux-type is one of debian, redhat, suse or ubuntu.
Forces the script to behave as it would for the specified linux. This is only necessary if the script cannot determine which of these is most appropriate (e.g. if used on an unsupported distribution).

-n <device-name> (default is Free-Flow).
Allows the device name to be varied (note that this must match the name used in the hid-adder.c source file.

-ns
Do not bind & start the driver - the script stops before calling mumo.<dm>.

-s <source version>
Use the specified source version.
This is the name of the directory at /usr/src to use for the kernel headers. The format of the directory name is distribution dependent. It is only necessary to specify this if you have more than one set of headers available.

Appendix 4: mumo

The mumo script loads the driver module, quirks the usbhid module if necessary (i.e. tells usbhid that 21d1:0001 devices have a separate driver module) and organises the monitors to allow the monitor configuration to be saved to the configfs file system where the driver can access the data. The mumo script is set-up to stream its output to ~/mumo.log rather than stdout.
Available options are:-

-d Display the output to stdout (defaults to a log file ~/mumo.log).

-nm Don't change the monitor layout (using xrandr), just accept the layout as it stands (some distributions don't make additional monitors active by default, using this option in that case will result in those monitors not being available. However, the use of xrandr in other situations may break an existing set-up).

Appendix 5: unmumo

Available options are:-

-d Display the output to stdout (defaults to a log file ~/mumo.log).

Appendix 6: demumo

Available options are:-
-d Display the output to stdout (defaults to a log file ~/mumo.log).

Appendix 7: Useful commands and utilities

To determine whether the driver has been bound to the device, type:

```
dmesg | grep hid
```

A line containing Free-Flow at the left-hand end (where generic-usb or hid-generic is seen for the other USB devices) shows that the driver is bound to that device.

To determine if the driver is available, type:

```
lsmod | grep hid
```

If hid_adder is not listed, it has not been installed or registered correctly.