FSProtect - Filesystem Protection

Harhalakis Stefanos <v13@v13.gr>

May 23, 2009

Contents

1 Introduction

FSProtect is a set of scripts that make immutable the root and other filesystems. This means that whatever is written to the disk will not be actually written and will be "lost" after the system is rebooted.

FSProtect typical usage is for:

- Public computers like those in libraries and labs
- Testing purposes (i.e. Test the upgrade of KDE3 -> KDE4 without actually changing anything to the system)
- Security

FSProtect is written for Debian but can be also be ported to other distributions. It is 100% dependent on the distribution so it cannot be made as a common program for all distributions.

2 How it works

FSProtect uses the "Another UnionFS" (aufs) filesystem and the tmpfs filesystem. Aufs is available as additional modules for debian and tmpfs is included in the kernel. FSProtect uses aufs to combine an existing filesystem with a tmpfs, putting tmpfs first in order. This results in changes been written to the tmpfs instead of the actual filesystem. Since tmpfs is a virtual filesystem that stores changes to VM (RAM or swap), everything is lost after a reboot.

The big benefits of using fsprotect, except from filesystem protection are:

- Immediate filesystem recovery - Since nothing is written on the disks, nothing needs to be restored. After a reboot everything is already in place.
- Protection from violent shutdowns - Since existing filesystems are not modified any more, they are mounted as read-only. This means that the computer can be safely powered off without going through the shutdown procedure. For public computers, this saves the administrator from checking and repairing destroyed filesystems.

FSProtect works in two phases:

1. Protect the root filesystem
2. Protect other filesystems

This is required since the root filesystem cannot be "protected" after it becomes /. Other filesystems can be "protected" only before they start being used.

2.1 Root filesystem

The root filesystem must be "protected" at the very first stages of the boot procedure. This is done during the initramfs boot stage where the root filesystem is mounted but hasn't become / yet (no pivot_root done). This is accomplished by adding a local-bottom script to the initramfs.

The script:

1. Creates /fsprotect, /fsprotect/system, /fsprotect/tmp, /fsprotect/aufs. Those are directories under the initial ram-based / which is available during the initramfs boot procedure. Those are not the directories of the real root filesystem.
2. Binds the root filesystem to /fsprotect/system (mount -o bind)
3. Mounts a tmpfs to /fsprotect/tmp
4. Creates a aufs of /fsprotect/system and /fsprotect/tmp
5. Umounts old root filesystem (from ${rootmnt})
6. Binds the aufs to ${rootmnt}
7. Umounts /fsprotect/aufs
8. Moves /fsprotect/system and /fsprotect/tmp inside ${rootmnt}
9. Create ${rootmnt}/fastboot to prevent attempts of fsck'ing the root filesystem

At this point, ${rootmnt} points to a aufs filesystem that combines a read-only physical root file-system and a tmpfs. All changes that are written to ${rootmnt} are stored in the tmpfs and are lost when the system reboots

The part of the script that accomplishes this is:

```
BASE=/fsprotect

log_begin_msg "Setting␣up␣fsprotect␣(aufs):"
[ -d $BASE ] || ( mkdir -m 700 $BASE || mkdir $BASE )
[ -d $BASE/system ] || mkdir $BASE/system
[ -d $BASE/tmp ] || mkdir $BASE/tmp
[ -d $BASE/aufs ] || mkdir $BASE/aufs

mount -n -o bind ${rootmnt} $BASE/system
mount -n -t tmpfs -o size=$SZ none $BASE/tmp
mount -n -t aufs -o dirs=$BASE/tmp=rw:$BASE/system=ro none $BASE/aufs
umount ${rootmnt}
mount -n -o bind $BASE/aufs ${rootmnt}
umount $BASE/aufs
mount -n -o move $BASE/system ${rootmnt}$BASE/system
mount -n -o move $BASE/tmp ${rootmnt}$BASE/tmp

touch ${rootmnt}/fastboot
```

2.2 Other filesystems

Other filesystems are protected during the boot procedure using an init script. This script needs to run just after the filesystems are mounted but before anything else is done. The init script is installed into single-user init procedure (S) with priority 37. This script reads /etc/default/fsprotect and protects the filesystems that are specified there. It relies on a helper script which it calls for each specified filesystem.

2.2.1 Helper script

The helper script is installed as /lib/fsprotect/fsprotect-protect. It is called as:

```
# /lib/fsprotect/fsprotect-protect <filesystem> <tmpfs-size>
```

This script does the same thing the initramfs script did. This is what it does:

```
BASE0="/fsprotect"
BASE="/fsprotect/fs"
MP="${1%/}"

DST0=$(echo "${MP#/}" | sed 's,/,-,g')

DST="$BASE/$DST0"

DSTORIG="$DST/orig"
DSTTMP="$DST/tmp"
DSTAUFS="$DST/aufs"

mount -o bind "${MP}" "$DSTORIG"
mount -t aufs -o "dirs=$DSTTMP=rw:$DSTORIG=ro" none "$DSTAUFS"
umount "$MP"
mount -o bind "$DSTAUFS" "$MP"
umount "$DSTAUFS" mount -o remount,ro "$DSTORIG"
```

3 Usage

Installation is as easy as installing a debian package. FSProtect can be enabled or disabled on demand. Configuration is done differently for the root filesystem and for other filesystems

3.1 Installation

Ti install fsprotect, just install the debian package with dpkg:

```
# dpkg -i fsprotect_1.0.1_all.deb
```

This will put everything in place.

3.2 Enabling and root filesystem

To enable fsprotect you need to add the fsprotect kernel option. The fsprotect parameter takes one optional argument, which it passes to mount(8) as the size option of the tmpfs. It is the size in bytes, and K/M/G suffixes can be used as multiplicators. If no argument is specified, mount(8) will create a tmpfs with size half the system's memory.

The syntax is fsprotect=size, where size can be any size that can be understood by mount. For example:

```
fsprotect=1024M

or just:

fsprotect
```

This will protect the root filesystem by using a 1024M tmpfs. This limits the changes that can be performed to the root filesystem to 1GB.

3.3 Other filesystems

For non-root filesystems to be protected, fsprotect you need to:

- Enable fsprotect with the kernel parameter
- Specify the filesystems to be protected and their tmpfs size in /etc/default/fsprotect

A sample /etc/default/fsprotect can have the following line:

```
PROTECT="/var=2048M /home=4096M"
```

This will protect /var and /home filesystems using 2GB and 4GB tmpfs respectively.

4 Copyright, License and Contact Information

4.1 Copyright

4.2 License

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You can find the full license text at: http://www.gnu.org/licenses/

4.3 Contact

For comments, request, bug reports and suggestions, contact Stefanos Harhalakis via e-mail at v13@v13.gr

5 Changes

23 May 2009: Better documentation for enabling fsprotect (Thanks to Martin T. Krafft)