```
VBS_FS
------

The vbsfs package consists of three utilities for dealing with scattered
recorded VLBI data on FlexBuff (and since version r45, July 2016, Mark6):

    vbs_ls
        lists all(some) vbs recordings on the system
    vbs_rm
        allows you to remove (a) recording(s)
    vbs_fs
        a FUSE virtual file system, presents each scattered recording as a
        single file, for use with standard UNIX tools

As of version r45 (27 July 2016) all utilities have been completely
rewritten for use on either FlexBuff or Mark6 hardware and deal
transparently with FlexBuff ("vbs") and/or Mark6 recordings ("dplane") even
when mixed recording formats are present on the disks.

Installation instructions can be found later on in this document. Since 29
May 2017 the changelog of vbs_fs is online as http://www.jive.eu/~verkout/flexbuff/vbs_fs.html


Basic operation (r45 and later)
-------------------------------

All three tools scan a set of disks for known recordings. The command line
arguments have been synchronized such that all tools share the disk
selection mechanism and some helpful options:

    --help       print detailed help for the utility
    --version    print the actual version of the utility
    -6           add Mark6 mountpoints to search path [/mnt/disks/*/*]
    -v           add FlexBuff mountpoints to search path [/mnt/disk*]
    -R <path>    add pattern <path> to search path
    -I <pattern> index recording(s) matching <pattern> (r53 and later)

If the search path is not set from the command line, the FlexBuff pattern will be used.

Also new since r45 is that in vbs_ls and vbs_fs the owner, group and
permissions metadata reflect the actual on disk owner, group and permissions.


vbs_ls and vbs_rm
-----------------


vbs_ls and vbs_rm try to mimic their famous UNIX equivalents ls(1) and
rm(1). As such vbs_ls and vbs_rm support many of the more pedestrian options
that ls(1) and rm(1) offer.

As an example:

    $> vbs_ls -lrth

produces output equivalent to

    $> ls -lrth


"vbs_ls" has an added "-T" option which yields "totals per pattern given on
the command line". Use this to e.g. find out the recorded size per
experiment:
    $> vbs_ls -lT gm082_ys_* n16l1_o8_*

or per station:
    $> vbs_ls -lT \*_ys_\* \*_o8_\*
```

```
Note that on some shells it is necessary to escape the asterisk or else
the shell will expand the file names (or set the "noglob" option in your
shell if it supports that)

Run  "vbs_{fs,ls,rm} --help" for detailed information on command line
options.

Because vbs_ls now supports both "dplane" and "vbs" recordings, the output
has been modified to reflect this. In the example below are shown some test
recordings made on one of JIVE's FlexBuffs in both "dplane" and "vbs" format:

    verkout@flexbuf1:~$ vbs_ls -l haavee*
    Found 4 recordings in 90 chunks,   57.68G
    -rw-r--r-- jops      flexbuf      16797931200 Jun 22 10:27 haavee_m6_no0001
    -rw-r--r-- jops      flexbuf      20981685600 Jun 22 10:28 haavee_m6_no0002
    drw-r--r-- jops      flexbuf      13689999360 Jun 22 10:27 haavee_vbs_no0001
    drw-r--r-- jops      flexbuf      10468823040 Jun 22 10:27 haavee_vbs_no0001a

Mark6/"dplane" recordings get type "-" (file) and EVN FlexBuff "vbs"
recordings are listed as directories. (On disk that is also the
distinction).

A file type of "+" means that the named recording was found in both vbs and
dplane format. This could happen if a recording by the exact same name was
made on two different disk packs and then mounted at the same time.

Or in this (synthetic) case - where vbs_ls is asked to aggregate ("-T") all
recordings matching the pattern 'haavee*' - which fits both the vbs and
dplane recordings shown previously:

    verkout@flexbuf1:~$ vbs_ls -lTh haavee*
    Found 3 recordings in 4503 chunks, 1.08T
    +rw-r--r-- jops flexbuf 55.43G Jun 22 10:28 haavee* 90.00% recovered [81/90]


vbs_fs
------

vbs_fs is the FUSE file system. It will in memory gather the scattered
recordings based on block sequence numbers. Basic usage:

    $> mkdir /path/to/mountpoint
    $> vbs_fs [options] /path/to/mountpoint

From this point on all recognized recordings found in the search path are presented as
regular files in "/path/to/mountpoint". Since r45 vbs_fs reassembles Mark6
"dplane" recordings as well.


Here follows a description of how to run and, after use, unmount/stop the
vbs_fs file system.

Running vbs_fs:

        # Create a mountpoint for the FUSE file system
        $> mkdir /path/to/mountpoint

        $> vbs_fs [-f] [-6] [-o allow_other] /path/to/mountpoint

        Where:
            -f              run in foreground, do not daemonize (can ^C it)
            -6              monitor Mark6 mountpoints for vbs recordings
                            (Only available since 0.4.1)
            -o allow_other  other users may access the mounted files too

        Run "vbs_fs -h" to show all options.  Since r45 use "vbs_fs --help"
        to show all options and also refer to 'common command line options'
```

```
        above.

vbs_fs r45 and later gained extra command line options, affecting startup
time and allow the user to balance memory usage with performance:

    --quick         perform a quick scan of the disks. Not all metadata
                    (size, access time) will be correct; a full scan is
                    expensive, which is the backward compatible default.

                    This option is mostly useful if it is known beforehand
                    which recording is to be opened (for e.g. correlation)
                    and a full scan of the disks is a waste of everyone's
                    time.

                    The recording sizes will be set to 0 and a full scan of a
                    recording is done at the moment a recording is first
                    opened. This means that before a recording can be read
                    from, it must be opened at least once to trigger the
                    scan.

                    The use of touch(1) is useful for that.

    -n <int>        The readahead amount. vbs_fs is optimized for sequential
                    access and does readahead of <n> blocks. Use this option
                    to either limit the readahead (0 is a valid value and
                    also the default) or e.g. to the number of disks - this
                    keeps all disks spinning all the time.

                    Setting the read ahead amount to number of disks it
                    should be possible to reach read-back speeds of 1GB/s or
                    more; subject to number of disks, CPUs and memory
                    available - YMMV.


Unmounting of the mounted vbs_fs file system depends on how 'vbs_fs' was
run:

    * if started with the '-f' flag ('run in foreground') you can just ^C
      the program and it will shut down correctly and unmount

    * otherwise it was daemonized - vbs_fs runs in the background. In such
      case the FUSE file system has to be unmounted. Use:

        $> fusermount -u /path/to/mountpoint

       to unmount the file system. The vbs_fs daemon program will exit.



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                        Installation and compilation

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



Since version 0.4.2 the package is distributed in two formats:

    vbs_fs-X.Y[.Z].tar.gz   - the raw source code, can install anywhere

    vbsfs-X.Y[.Z].deb       - self-compiling and installing Debian package;
                              installs as /usr/local/bin/{vbs_fs, vbs_ls, vbs_rm}
```

```
    (Note the subtle difference in file names. The Debian Package system
    did not allow '_' in the package name).


See below for installation instructions for both formats. After having
compiled or installed the package, vbs_fs can be run as follows.



Questions, comments and suggestions can always be sent to verkouter at jive
dot eu.

(*) The FlexBuff software was formerly called 'VLBI Streamer' - hence the
tla 'vbs'.


#########################################################################
#####
#####
#####          Detailed installation instructions
#####
#####               vbs_fs-X.Y[.Z].tar.gz
#####
#####              installation from source
#####
#####
#########################################################################

0. Make sure libfuse-dev, make and g++ are installed, for the FUSE header file(s) and
libraries.

1. Download and un(zip+tar) the source code:
    $> wget http://www.jive.nl/.../vbs_fs-X.Y[.Z].tar.gz
    $> tar zxf vbs_fs-X.Y[.Z].tar.gz

2. Configure and build:
    $> cd vbs_fs-X.Y[.Z]
    # 'make install' will install vbs_fs, vbs_ls and vbs_rm
    # in PREFIX/bin. Use ./configure --help to see defaults.
    $> ./configure [--prefix=PREFIX]

    # build and optionally install, if you want to do that right now
    $> make [install]

3. After configuring, the Makefile (created by the configure step) has two
other useful make targets:

    - make clean
        cleans the object files and binary/ies locally, does not remove the
        installed binary/ies
    - make uninstall
        This target is only available since 0.4.2 and - as the name suggests -
        does an uninstall of the installed binary/ies and configuration
        file(s)


#########################################################################
#####
#####
#####          Detailed installation instructions
#####
#####               vbsfs-X.Y[.Z].deb
#####
#####           installation from Debian Package
#####
#####
#########################################################################
```

```
0. Make sure libfuse-dev, make and g++ are installed, for the FUSE header
file(s), C++ compiler and make utility:
    $> sudo apt-get install make g++ libfuse-dev

1. Download and install the .deb
    $> wget http://www.jive.nl/.../vbsfs-X.Y[.Z].deb
    $> sudo dpkg -i /path/to/vbsfs-X.Y[.Z].deb

   The installer will unpack vbs_fs's source code in
   /usr/src/vbs_fs-X.Y[.Z]. Following that, the source code will be
   configured for installation under "/usr/local/bin", then compiled and finally
   installed. The package will not install if the dependencies (step 0) are
   not met.

2. The package and all its files can easily be removed using this command:
    $> sudo dpkg -r vbsfs
```