Mass Storage Gadget (MSG)

Overview

Mass Storage Gadget (or MSG) acts as a USB Mass Storage device, appearing to the host as a disk or a CD-ROM drive. It supports multiple logical units (LUNs). Backing storage for each LUN is provided by a regular file or a block device, access can be limited to read-only, and gadget can indicate that it is removable and/or CD-ROM (the latter implies read-only access).

Its requirements are modest; only a bulk-in and a bulk-out endpoint are needed. The memory requirement amounts to two 16K buffers. Support is included for full-speed, high-speed and SuperSpeed operation.

Note that the driver is slightly non-portable in that it assumes a single memory/DMA buffer will be usable for bulk-in and bulk-out endpoints. With most device controllers this is not an issue, but there may be some with hardware restrictions that prevent a buffer from being used by more than one endpoint.

This document describes how to use the gadget from user space, its relation to mass storage function (or MSF) and different gadgets using it, and how it differs from File Storage Gadget (or FSG) (which is no longer included in Linux). It will talk only briefly about how to use MSF within composite gadgets.

Module parameters

The mass storage gadget accepts the following mass storage specific module parameters:

- file=filename[,filename...]

  This parameter lists paths to files or block devices used for backing storage for each logical unit. There may be at most FSG_MAX_LUNS (8) LUNs set. If more files are specified, they will be silently ignored. See also "luns" parameter.

  BEWARE that if a file is used as a backing storage, it may not be modified by any other process. This is because the host assumes the data does not change without its knowledge. It may be read, but (if the logical unit is writable) due to buffering on the host side, the contents are not well defined.

  The size of the logical unit will be rounded down to a full logical block. The logical block size is 2048 bytes for LUNs simulating CD-ROM, block size of the device if the backing file is a block device, or 512 bytes otherwise.

- removable=b[,b...]

  This parameter specifies whether each logical unit should be removable. "b" here is either "y", "Y" or "1" for true or "n", "N" or "0" for false.

  If this option is set for a logical unit, gadget will accept an "eject" SCSI request (Start/Stop Unit). When it is sent, the backing file will be closed to simulate ejection and the logical unit will not be mountable by the host until a new backing file is specified by userspace on the device (see "sysfs entries" section).

  If a logical unit is not removable (the default), a backing file must be specified for it with the "file" parameter as the module is loaded. The same applies if the module is built in, no exceptions.

  The default value of the flag is false, *HOWEVER* it used to be true. This has been changed to better match File Storage Gadget and because it seems like a saner default after all. Thus to maintain compatibility with older kernels, it's best to specify the default values. Also, if one relied on old default, explicit "n" needs to be specified now.

  Note that "removable" means the logical unit's media can be ejected or removed (as is true for a CD-ROM drive or a card reader). It does *not* mean that the entire gadget can be unplugged from the host; the proper term for that is "hot-unpluggable".

- cdrom=b[,b...]

  This parameter specifies whether each logical unit should simulate CD-ROM. The default is false.

- ro=b[,b...]

  This parameter specifies whether each logical unit should be reported as read only. This will prevent host from modifying the backing files.

  Note that if this flag for given logical unit is false but the backing file could not be opened in read/write mode, the gadget will fall back to read only mode anyway.

  The default value for non-CD-ROM logical units is false; for logical units simulating CD-ROM it is forced to true.

- nofua=b[,b...]

  This parameter specifies whether FUA flag should be ignored in SCSI Write10 and Write12 commands sent to given logical units.

MS Windows mounts removable storage in "Removal optimised mode" by default. All the writes to the media are synchronous, which is achieved by setting the FUA (Force Unit Access) bit in SCSI Write(10,12) commands. This forces each write to wait until the data has actually been written out and prevents I/O requests aggregation in block layer dramatically decreasing performance.

Note that this may mean that if the device is powered from USB and the user unplugs the device without unmounting it first (which at least some Windows users do), the data may be lost.

The default value is false.

- luns=N

This parameter specifies number of logical units the gadget will have. It is limited by FSG_MAX_LUNS (8) and higher value will be capped.

If this parameter is provided, and the number of files specified in "file" argument is greater then the value of "luns", all excess files will be ignored.

If this parameter is not present, the number of logical units will be deduced from the number of files specified in the "file" parameter. If the file parameter is missing as well, one is assumed.

- stall=b

Specifies whether the gadget is allowed to halt bulk endpoints. The default is determined according to the type of USB device controller, but usually true.

In addition to the above, the gadget also accepts the following parameters defined by the composite framework (they are common to all composite gadgets so just a quick listing):

- idVendor -- USB Vendor ID (16 bit integer)
- idProduct -- USB Product ID (16 bit integer)
- bcdDevice -- USB Device version (BCD) (16 bit integer)
- iManufacturer -- USB Manufacturer string (string)
- iProduct -- USB Product string (string)
- iSerialNumber -- SerialNumber string (sting)

sysfs entries

For each logical unit, the gadget creates a directory in the sysfs hierarchy. Inside of it the following three files are created:

- file

When read it returns the path to the backing file for the given logical unit. If there is no backing file (possible only if the logical unit is removable), the content is empty.

When written into, it changes the backing file for given logical unit. This change can be performed even if given logical unit is not specified as removable (but that may look strange to the host). It may fail, however, if host disallowed medium removal with the Prevent-Allow Medium Removal SCSI command.

- ro

Reflects the state of ro flag for the given logical unit. It can be read any time, and written to when there is no backing file open for given logical unit.

- nofua

Reflects the state of nofua flag for given logical unit. It can be read and written.

Other then those, as usual, the values of module parameters can be read from /sys/module/g_mass_storage/parameters/* files.

Other gadgets using mass storage function

The Mass Storage Gadget uses the Mass Storage Function to handle mass storage protocol. As a composite function, MSF may be used by other gadgets as well (eg. g_multi and acm_ms).

All of the information in previous sections are valid for other gadgets using MSF, except that support for mass storage related module parameters may be missing, or the parameters may have a prefix. To figure out whether any of this is true one needs to consult the gadget's documentation or its source code.

For examples of how to include mass storage function in gadgets, one may take a look at mass_storage.c, acm_ms.c and multi.c (sorted by complexity).

Relation to file storage gadget

The Mass Storage Function and thus the Mass Storage Gadget has been based on the File Storage Gadget. The difference

between the two is that MSG is a composite gadget (ie. uses the composite framework) while file storage gadget was a traditional gadget. From userspace point of view this distinction does not really matter, but from kernel hacker's point of view, this means that (i) MSG does not duplicate code needed for handling basic USB protocol commands and (ii) MSF can be used in any other composite gadget.

Because of that, File Storage Gadget has been removed in Linux 3.8. All users need to transition to the Mass Storage Gadget. The two gadgets behave mostly the same from the outside except:

1. In FSG the "removable" and "cdrom" module parameters set the flag for all logical units whereas in MSG they accept a list of y/n values for each logical unit. If one uses only a single logical unit this does not matter, but if there are more, the y/n value needs to be repeated for each logical unit.
2. FSG's "serial", "vendor", "product" and "release" module parameters are handled in MSG by the composite layer's parameters named respectively: "iSerialnumber", "idVendor", "idProduct" and "bcdDevice".
3. MSG does not support FSG's test mode, thus "transport", "protocol" and "buflen" FSG's module parameters are not supported. MSG always uses SCSI protocol with bulk only transport mode and 16 KiB buffers.