Modifications to the storage stack to better support tgt, sg, st and bsg.

- Scatterlist building and userspace <-> kernel data transfer.
- Userspace interface and SG v4 requirements.

Userspace<->Kernel Data Transfer

- sg, st, tgt, bsg and scsi_ioctl.c have implemented their own methods to copy or map data between userspace and the kernel. Reasons:
  - Block layer helpers provides basic functionality for sg v3, but sg and st have lots of features:
    - Reserved buffer support
    - mmap helper functions
    - Large requests using contiguous segments
  - tgt cannot control what size commands that will be sent to it.

Merging Data Transfer Code

- Move sg and st features to the block layer.
    - Do we need a new abstraction like sg_limits?
- Supporting real limits of hardware.
    - Modern Emulex and Qlogic hardware do not have scatter gather or IO size limits as they are defined by scsi_host_template and q limits today.
        - Increase SG_ALL and SCSI_MAX_PHYS_SEGMENTS.
            - Still have hard limit, but might be able to handle most case.
        - Is there a way to make scatterlist allocation not reliant on scsi_sg_pools for SG_IO and tgt requests?

Merging Data Transfer Code (cont)

- Pass through permission table
  - sg and SG_IO use the different permission tables
  - cmdfilter enables userspace to change permissions
  - Where should the permission tables be attached? gendisk, request queue, or something?
  - How to handle ATA passthrough?

sgv4: Basic Design Issues

- Who needs this?
  - Command (with task tag and attribute) and TMF
  - Transport level request & responses
  - Non request & response protocol?
    - requests from kernel & responses from user (tgt)
- How should sgv4 be implemented?
  - bsg, sg.c, or both
  - How to support bsg for non devices (like FC crasses)?
    - bsg-devicess attached to gendisk
    - Make bsg devices to everything via resest queue
- How should sgv4 be compatible with sgv3?
  - iovec -> 32/64 bit compat problem
  - mmap -> only one outstanding semantics

sgv4: Basic Design Issues (cont.)

- The interface between user and kernel
  - write/read system call
    - Writes buffer containing requests / reads buffer containing responses
    - A bit hacky & not effective
  - consumer/producer ring buffer
    - Use just two pointers
    - Not work well with multiple processes/threads
  - Ring buffer with new system calls
    - kevent is trying this
    - Common code for ring buffer (share with kevent, tgt, etc)

sgv4: Task Tag

- sgv4 needs task tag for task abort
  - TMF request to abort a task from user space
- User can sends a command with tag or sg returns a tag to user?
  - The block layer tagging might work in the latter case
  - The block layer tagging happens a bit later after user sends a command.
  - Only few LLDs use the block layer tagging
- Task tag collision
- Some LLDs simply can't support task tag

sgv4: Task Management Functions

- Where should hooks for tmf be added?
  - transport classes?