SCST Issues Discovered (Summary)

In the course of experimentation and study I found a few minor problems in the SCST implementation that resulted in some small changes to my copy of the SCST source code. Some of these issues were found by valgrind; one was found by dereferencing a NULL pointer (SEGV) [and rapidly debugged with gdb]; in other cases the bugs exposed themselves through assertions; some clues appeared by enabling more compiler warnings; and some I just happened to notice while reading code.

Running a large body of multi-threaded code like SCST using a new threading arrangement is a likely way to execute code sequences that rarely or never have occurred before — so it is not surprising at all to see a few bugs shake out.

The application is heavy on dynamic memory allocation, and there are many error paths, which together bring ample opportunity for memory leaks. So naturally the first time running a tool like valgrind(1) on code like this is another likely source of discovery.

Also, some of the issues coming to my attention appear in the obsolescent /proc support logic, which is not used by SCST when resident in a modern version of the Linux kernel, so it probably no longer gets regular testing.

> Not knowing either sysfs or procfs at the outset, their relative sizes of implementation in SCST led me to think it would be easier to figure out how to use fuse(8) to emulate procfs than sysfs.
>
> On the other hand, I have no doubt at all that implementing fuse(8) emulation of procfs was *far* easier than trying to figure out how to do the job of scstadmin some other way — it's really sweet having that script "*just work*" on the usermode process!

In fact, getting this 80,000 lines of SCST code to run in usermode was remarkably very little trouble, and the result seems quite solid, with the usermode server process staying up for weeks at a time running intermittent tests without problems, in the face of Initiator machine crashes and cable disconnections, and the Server machine also being used as my general-purpose workstation.

The Appendix contains a list of issues found, in the form of short diff(1) listing fragments, each followed by a (usually) brief explanation of why I thought it was a problem.

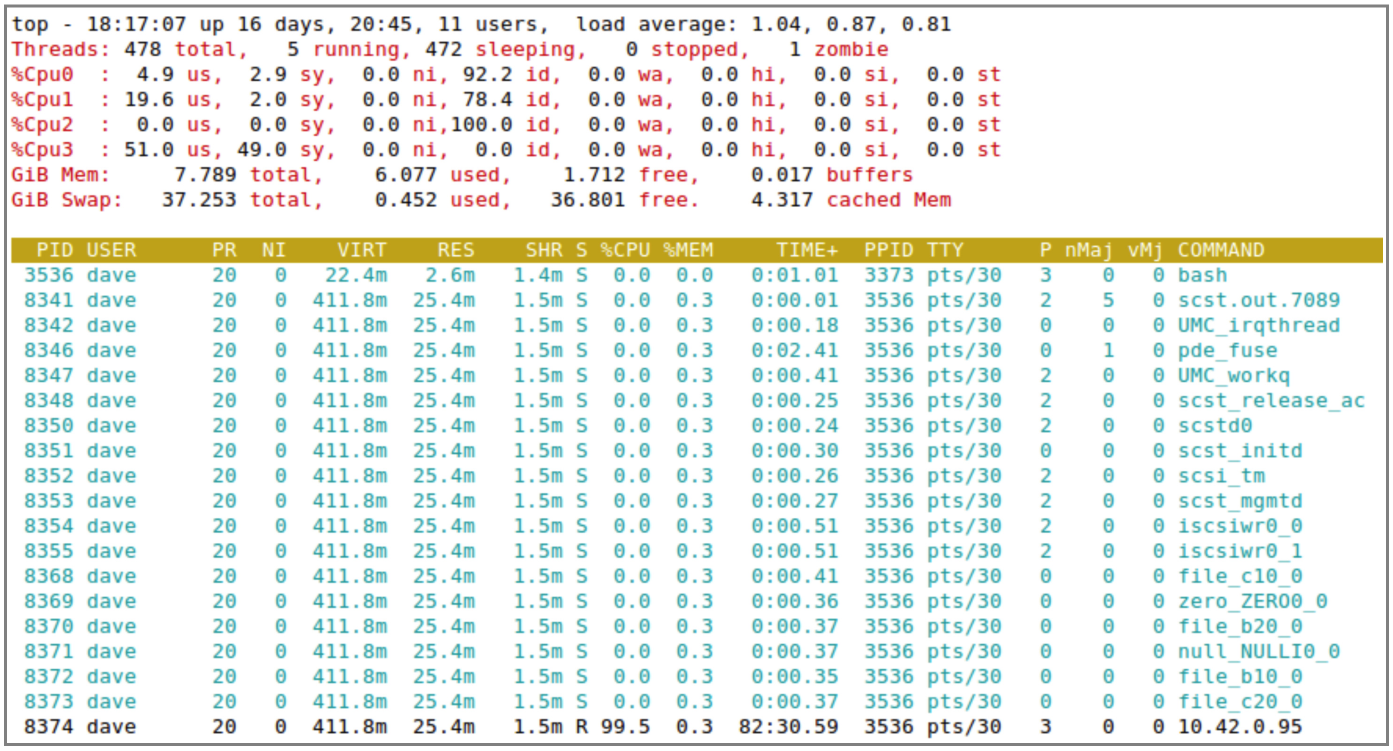
Sample top(1) Display

The top(1) display depicts a running 512-Byte Random Read workload from a single Linux Initiator using a single session over a single 1 Gb Ethernet connection.

All the threads seen here are in the Server process except the first "bash" thread. The Server process was started on (CPU0|CPU2). As the Initiator connected, the Session Thread 10.42.0.95 (bottom) was created, which I then manually taskset to CPU3. That is why in the top(1) display all the Server threads are seen to be on CPU0 or CPU2, except the 10.42.0.95 Session Thread on CPU3 ("P" column).

Notice that all the work (since the Server was last started) is being done by the Session Thread 10.42.0.95, with 82+ minutes of CPU time accumulated. The pde_fuse thread is the only other one with more than one second of CPU time accumulated. The two iSCSI Writer Threads iscsiwr0_0 and iscsiwr0_1 are not active in this workload.

The Session Thread is the thread that executes the epoll_wait(2) system call awaiting socket input and carries the operation all the way from reading the Request from the socket until writing the Reply to the socket.


```
top - 18:17:07 up 16 days, 20:45, 11 users,  load average: 1.04, 0.87, 0.81
Threads: 478 total,   5 running, 472 sleeping,   0 stopped,   1 zombie
%Cpu0  :  4.9 us,  2.9 sy,  0.0 ni, 92.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu1  : 19.6 us,  2.0 sy,  0.0 ni, 78.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu2  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu3  : 51.0 us, 49.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
GiB Mem:     7.789 total,    6.077 used,    1.712 free,    0.017 buffers
GiB Swap:   37.253 total,    0.452 used,   36.801 free.    4.317 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+  PPID TTY       P nMaj vMj COMMAND
 3536 dave      20   0   22.4m   2.6m   1.4m S  0.0  0.0   0:01.01  3373 pts/30    3    0   0 bash
 8341 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.01  3536 pts/30    2    5   0 scst.out.7089
 8342 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.18  3536 pts/30    0    0   0 UMC_irqthread
 8346 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:02.41  3536 pts/30    0    1   0 pde_fuse
 8347 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.41  3536 pts/30    2    0   0 UMC_workq
 8348 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.25  3536 pts/30    2    0   0 scst_release_ac
 8350 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.24  3536 pts/30    2    0   0 scstd0
 8351 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.30  3536 pts/30    0    0   0 scst_initd
 8352 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.26  3536 pts/30    2    0   0 scsi_tm
 8353 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.27  3536 pts/30    2    0   0 scst_mgmtd
 8354 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.51  3536 pts/30    2    0   0 iscsiwr0_0
 8355 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.51  3536 pts/30    2    0   0 iscsiwr0_1
 8368 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.41  3536 pts/30    0    0   0 file_c10_0
 8369 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.36  3536 pts/30    0    0   0 zero_ZERO0_0
 8370 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.37  3536 pts/30    0    0   0 file_b20_0
 8371 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.37  3536 pts/30    0    0   0 null_NULLI0_0
 8372 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.35  3536 pts/30    0    0   0 file_b10_0
 8373 dave      20   0  411.8m  25.4m   1.5m S  0.0  0.3   0:00.37  3536 pts/30    0    0   0 file_c20_0
 8374 dave      20   0  411.8m  25.4m   1.5m R 99.5  0.3  82:30.59  3536 pts/30    3    0   0 10.42.0.95
```

Sample Content from /fuse/scst/proc/scsi_tgt

```
:::::::::::::: /fuse/scst/proc/scsi_tgt/iscsi/session ::::::::::::::
tid:1 name:iqn.2001-04.com.example.blackbox:1
        sid:10000043d0200 initiator:iqn.1993-08.org.debian:01:9778279657b7 (reinstating no)
                cid:0 ip:10.42.0.95 state:read_processing  hd:none dd:none
tid:2 name:iqn.2001-04.com.example.blackbox:2
tid:3 name:iqn.2001-04.com.example.blackbox:3
tid:4 name:iqn.2001-04.com.example.blackbox:4

:::::::::::::: /fuse/scst/proc/scsi_tgt/iscsi/version ::::::::::::::
3.3.0-pre1-procfs

:::::::::::::: /fuse/scst/proc/scsi_tgt/vcdrom/vcdrom ::::::::::::::
Name              Size(MB)  File name

:::::::::::::: /fuse/scst/proc/scsi_tgt/vcdrom/type ::::::::::::::
5 - CD-ROM device

:::::::::::::: /fuse/scst/proc/scsi_tgt/vdisk/vdisk ::::::::::::::
Name              Size(MB)    Block size  Options         File name                                     T10 device id
null_NULLIO       2621440     512         NIO             /dev/zero                                     N000
zero_ZERO         1073741824  512         NV              /dev/zero                                     Z000
disk_sda5         15280       4096                        /dev/sda5                                     sda5
file_b1           600         512         NV              /tmp/disk_b1                                  F_b1
file_b2           601         512         NV              /tmp/disk_b2                                  F_b2
file_c1           602         512         NV              /tmp/disk_c1                                  F_c1
file_c2           603         512         NV              /tmp/disk_c2                                  F_c2

:::::::::::::: /fuse/scst/proc/scsi_tgt/vdisk/type ::::::::::::::
0 - Direct-access device (e.g., magnetic disk)

:::::::::::::: /fuse/scst/proc/scsi_tgt/sgv ::::::::::::::
Inactive/active pages                       0/0
Hi/lo watermarks [pages]                    31888/0
Hi watermark releases/failures              0/0
Name                           Hit         Total       % merged    Cached (P/I/O)
sgv-0                          0           0           0           0/0/0
  ...
  big/other                                0/1450708   0/0
... [all others zero]

:::::::::::::: /fuse/scst/proc/scsi_tgt/groups/cian/names ::::::::::::::
iqn.1993-08.org.debian:01:9778279657b7
iqn.1991-05.com.microsoft:borgcube

:::::::::::::: /fuse/scst/proc/scsi_tgt/groups/cian/devices ::::::::::::::
Device (host:ch:id:lun or name)                             LUN          Options
file_c1                                                     31
zero_ZERO                                                   1
file_b2                                                     22
null_NULLIO                                                 0
file_b1                                                     21
file_c2                                                     32

:::::::::::::: /fuse/scst/proc/scsi_tgt/groups/Default/devices ::::::::::::::
Device (host:ch:id:lun or name)                             LUN          Options
null_NULLIO                                                 0
zero_ZERO                                                   1

:::::::::::::: /fuse/scst/proc/scsi_tgt/groups/Default/addr_method ::::::::::::::
PERIPHERAL

:::::::::::::: /fuse/scst/proc/scsi_tgt/threads ::::::::::::::
1

:::::::::::::: /fuse/scst/proc/scsi_tgt/sessions ::::::::::::::
Target name          Initiator name                          Group name                           Active/All Commands Count
iscsi                iqn.1993-08.org.debian:01:9778279657b7        cian                                 0/0

:::::::::::::: /fuse/scst/proc/scsi_tgt/version ::::::::::::::
3.3.0-pre1-procfs
TEST_IO_IN_SIRQ

:::::::::::::: /fuse/scst/proc/scsi_tgt/scsi_tgt ::::::::::::::
Device (host:ch:id:lun or name)                             Device handler
null_NULLIO                                                 vdisk_nullio
zero_ZERO                                                   vdisk_fileio
disk_sda5                                                   vdisk_fileio
file_b1                                                     vdisk_fileio
file_b2                                                     vdisk_fileio
file_c1                                                     vdisk_fileio
file_c2                                                     vdisk_fileio
```

Sample Logging (Startup, before scstadmin)

```
dave@blackbox:~/src$ ./scst.out.7089 -f                                         # UMC = Usermode Compatibility (kernel service simulati
E>1484778835.767186685 [8341]: DEBUG: Thread scst.out.7089 (8341) creates irqthread UMC_irqthread
E>1484778835.767258694 [8341]: MTE_EVENT:_eventfd_create:140: new event fd name='scst.out.7089' fd=6
E>1484778835.767280065 [8341]: MTE_EVENT:mte_event_task_alloc:908: sig_fd=7
E>1484778835.767359688 [8342]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread UMC_irqthread @0x1f5c700 starts up on tid=8342
E>1484778835.767667307 [8342]: MTE_EVENT:mte_event_task_run:804: NOTICE: event_task UMC_irqthread @0x1f5d200 starts up on tid=8342 epoll_
E>1484778835.777390310 [8341]: PDE_FUSE:pde_fuse_start:506: NOTICE: created /proc PDE_ROOT @0x1f5d600 -- starting fuse service
E>1484778835.777493048 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread UMC_workq
E>1484778835.777544209 [8347]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread UMC_workq @0x1f5e080 starts up on tid=8347
E>1484778835.777603718 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread scst_release_acg
E>1484778835.777655029 [8346]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread pde_fuse @0x1f5d840 starts up on tid=8346
E>1484778835.777689507 [8346]: PDE_FUSE:pde_fuse_run:462: NOTICE: pde_fuse thread @0x1f5d840 starts up on tid=8346
E>1484778835.777962999 [8348]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread scst_release_acg @0x1f5f2c0 starts up on tid=8348
E>1484778835.782574511 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread scstd0
E>1484778835.782738418 [8350]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread scstd0 @0x224bd00 starts up on tid=8350
E>1484778835.782839693 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread scst_initd
E>1484778835.783009235 [8351]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread scst_initd @0x224c700 starts up on tid=8351
E>1484778835.783048992 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread scsi_tm
E>1484778835.783111805 [8351]: INFO: scst: Init thread started
E>1484778835.783148138 [8352]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread scsi_tm @0x224d100 starts up on tid=8352
E>1484778835.783175417 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread scst_mgmtd
E>1484778835.783247131 [8352]: INFO: scst: Task management thread started
E>1484778835.783279390 [8353]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread scst_mgmtd @0x224db00 starts up on tid=8353
E>1484778835.783331958 [8341]: INFO: scst: SCST version 3.3.0-pre1-procfs loaded successfully (max mem for commands 1993MB, per device 79
E>1484778835.783354501 [8341]: INFO: scst: Enabled features: TEST_IO_IN_SIRQ
E>1484778835.783406642 [8341]: INFO: scst: Virtual device handler vdisk_fileio for type 0 registered successfully
E>1484778835.783424538 [8341]: INFO: scst: Virtual device handler vdisk_blockio for type 0 registered successfully
E>1484778835.783440989 [8341]: INFO: scst: Virtual device handler vdisk_nullio for type 0 registered successfully
E>1484778835.783454398 [8341]: INFO: scst: Virtual device handler vcdrom for type 5 registered successfully
E>1484778835.783477484 [8341]: INFO: iscsi-scst: iSCSI SCST Target - version 3.3.0-pre1-procfs
E>1484778835.783489883 [8341]: INFO: iscsi-scst: Registered iSCSI transport: iSCSI-TCP
E>1484778835.783518921 [8341]: INFO: scst: Target template iscsi registered successfully
E>1484778835.783549575 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread iscsiwr0_0
E>1484778835.783598315 [8353]: INFO: scst: Management thread started
E>1484778835.783621834 [8354]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread iscsiwr0_0 @0x2263880 starts up on tid=8354
E>1484778835.783649379 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread iscsiwr0_1
E>1484778835.784529892 [8354]: INFO: iscsi-scst: Write thread for pool 0x2262fc0 started
E>1484778835.784614308 [8355]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread iscsiwr0_1 @0x2264280 starts up on tid=8355

1484778835.784691: max_data_seg_len 268435456, max_queued_cmds 2048

E>1484778835.784958836 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:1 (relative target id 1) for template iscsi registered
E>1484778835.785005332 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:2 (relative target id 2) for template iscsi registered
E>1484778835.785036622 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:3 (relative target id 3) for template iscsi registered
E>1484778835.785073603 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:4 (relative target id 4) for template iscsi registered
E>1484778835.785272123 [8355]: INFO: iscsi-scst: Write thread for pool 0x2262fc0 started
cannot find isert_scst in /proc/devices - make sure the module is loaded
E>1484778835.789705236 [8346]: WARNING: [1/10] ((0 == 0)) 1/0x1 pde_lookup failed to find /.Trash under PDE_ROOT
E>1484778835.789800775 [8346]: WARNING: [2/10] ((0 == 0)) 1/0x1 pde_lookup failed to find /.Trash-1000 under PDE_ROOT

1484778836.586801: Connect from 10.42.0.95:44840 to 10.42.0.1:3260
E>1484778836.837013156 [8341]: scst_event_queue_negative_luns_inquiry:scst_event_queue_negative_luns_inquiry:313: WARNING: SKIP queuing e
1484778836.837035: Initiator iqn.1993-08.org.debian:01:9778279657b7 not allowed to connect to target iqn.2001-04.com.example.blackbox:1
```

Sample Logging (Startup, during scstadmin)

```
E>1484778839.483695016 [8346]: WARNING: [3/10] ((0 == 0)) 1/0x1 pde_lookup failed to find /type under iscsi
...
E>1484778839.486697373 [8346]: WARNING: [10/10] ((0 == 0)) 1/0x1 pde_lookup failed to find /type under groups
E>1484778839.486972242 [8346]: INFO: dev_vdisk: Registering virtual vdisk_nullio device null_NULLIO (NULLIO, ROTATIONAL)
E>1484778839.487047375 [8346]: filp_open:filp_open:1961: NOTICE: name='/var/lib/scst/pr' fd=4 statbuf.st_size=4096 lseek_end_ofs=92233720
E>1484778839.487075478 [8346]: INFO: dev_vdisk: Attached SCSI target virtual disk null_NULLIO (file="/dev/zero", fs=2621440MB, bs=512, nb
E>1484778839.487167021 [8346]: INFO: scst: Attached to virtual device null_NULLIO (id 1)
E>1484778839.487958662 [8346]: INFO: dev_vdisk: T10 device id for device null_NULLIO changed to N000
...
E>1484778839.498675662 [8346]: INFO: dev_vdisk: Registering virtual vdisk_fileio device file_c2 (NV_CACHE, ROTATIONAL)
E>1484778839.498708142 [8346]: filp_open:filp_open:1961: NOTICE: name='/var/lib/scst/pr' fd=4 statbuf.st_size=4096 lseek_end_ofs=92233720
E>1484778839.498726879 [8346]: filp_open:filp_open:1961: NOTICE: name='/tmp/disk_c2' fd=4 statbuf.st_size=632291328 lseek_end_ofs=6322913
E>1484778839.498753440 [8346]: INFO: dev_vdisk: Attached SCSI target virtual disk file_c2 (file="/tmp/disk_c2", fs=603MB, bs=512, nblocks
E>1484778839.498787304 [8346]: INFO: scst: Attached to virtual device file_c2 (id 7)
E>1484778839.499847981 [8346]: INFO: dev_vdisk: T10 device id for device file_c2 changed to F_c2

E>1484778839.500915116 [8346]: INFO: scst: Added name iqn.1991-05.com.microsoft:borgcube to group cian (target ?)
E>1484778839.501439289 [8346]: INFO: scst: Added name iqn.1993-08.org.debian:01:9778279657b7 to group cian (target ?)
E>1484778839.502011327 [8346]: INFO: scst: Added device file_c1 to group cian (LUN 31, flags 0x0) to target ?
E>1484778839.502605577 [8346]: INFO: scst: Added device zero_ZERO to group cian (LUN 1, flags 0x0) to target ?
E>1484778839.503158346 [8346]: INFO: scst: Added device file_b2 to group cian (LUN 22, flags 0x0) to target ?
E>1484778839.503723189 [8346]: INFO: scst: Added device null_NULLIO to group cian (LUN 0, flags 0x0) to target ?
E>1484778839.504293554 [8346]: INFO: scst: Added device file_b1 to group cian (LUN 21, flags 0x0) to target ?
E>1484778839.504870811 [8346]: INFO: scst: Added device file_c2 to group cian (LUN 32, flags 0x0) to target ?
E>1484778839.505432073 [8346]: INFO: scst: Added device zero_ZERO to group Default (LUN 1, flags 0x0) to target ?
E>1484778839.505973262 [8346]: INFO: scst: Added device null_NULLIO to group Default (LUN 0, flags 0x0) to target ?
```

Sample Logging (Startup, after scstadmin)

```
1484778839.591132: Connect from 10.42.0.95:44842 to 10.42.0.1:3260
E>1484778839.841439575 [8341]: INFO: scst: Using security group "cian" for initiator "iqn.1993-08.org.debian:01:9778279657b7" (target iqn
E>1484778839.841490706 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread file_c10_0
E>1484778839.841568608 [8368]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread file_c10_0 @0x7f2f3c027900 starts up on tid=8368
E>1484778839.841662654 [8341]: filp_open:filp_open:1961: NOTICE: name='/tmp/disk_c1' fd=14 statbuf.st_size=631242752 lseek_end_ofs=631242
E>1484778839.841691520 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread zero_ZERO0_0
E>1484778839.841733223 [8369]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread zero_ZERO0_0 @0x2270880 starts up on tid=8369
E>1484778839.841783856 [8341]: filp_open:filp_open:1961: NOTICE: name='/dev/zero' fd=15 statbuf.st_size=1125899906842624 lseek_end_ofs=0
E>1484778839.841811312 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread file_b20_0
E>1484778839.841857583 [8370]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread file_b20_0 @0x2271e80 starts up on tid=8370
E>1484778839.841903195 [8341]: filp_open:filp_open:1961: NOTICE: name='/tmp/disk_b2' fd=16 statbuf.st_size=630194176 lseek_end_ofs=630194
E>1484778839.841926445 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread null_NULLI0_0
E>1484778839.841968634 [8371]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread null_NULLI0_0 @0x2273480 starts up on tid=8371
E>1484778839.842007228 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread file_b10_0
E>1484778839.842047845 [8372]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread file_b10_0 @0x2274980 starts up on tid=8372
E>1484778839.842095291 [8341]: filp_open:filp_open:1961: NOTICE: name='/tmp/disk_b1' fd=17 statbuf.st_size=629145600 lseek_end_ofs=629145
E>1484778839.842121553 [8341]: DEBUG: Thread scst.out.7089 (8341) creates kthread file_c20_0
E>1484778839.842164359 [8373]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread file_c20_0 @0x2275f80 starts up on tid=8373
E>1484778839.842196776 [8341]: filp_open:filp_open:1961: NOTICE: name='/tmp/disk_c2' fd=18 statbuf.st_size=632291328 lseek_end_ofs=632291
E>1484778839.842221058 [8341]: INFO: iscsi-scst: Negotiated parameters: InitialR2T No, ImmediateData Yes, MaxConnections 1, MaxRecvDataSe
E>1484778839.842233790 [8341]: INFO: iscsi-scst:     MaxBurstLength 16773120, FirstBurstLength 262144, DefaultTime2Wait 0, DefaultTime2Re
E>1484778839.842243698 [8341]: INFO: iscsi-scst:     MaxOutstandingR2T 1, DataPDUInOrder Yes, DataSequenceInOrder Yes, ErrorRecoveryLevel
E>1484778839.842254271 [8341]: INFO: iscsi-scst:     HeaderDigest None, DataDigest None, OFMarker No, IFMarker No, OFMarkInt 2048, IFMark
E>1484778839.842265544 [8341]: INFO: iscsi-scst: Target parameters set for session 10000023d0200: QueuedCommands 2048, Response timeout 9
E>1484778839.842354008 [8341]: DEBUG: Thread scst.out.7089 (8341) creates irqthread 10.42.0.95
E>1484778839.842393941 [8341]: MTE_EVENT:_eventfd_create:140: new event fd name='scst.out.7089' fd=21
E>1484778839.842428041 [8341]: MTE_EVENT:mte_event_task_alloc:908: sig_fd=22
E>1484778839.842478735 [8374]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread 10.42.0.95 @0x226d180 starts up on tid=8374
E>1484778839.842513269 [8374]: MTE_EVENT:mte_event_task_run:804: NOTICE: event_task 10.42.0.95 @0x226dc00 starts up on tid=8374 epoll_fd=
E>1484778839.842586404 [8374]: INFO: iscsi-scst: nagle_threshold=9999999999
```

Sample Logging (Shutdown, SIGINT-->exit)

```
^C
E>1484791455.406755951 [8374]: MTE_EVENT:sigfd_handler:737: sigfd_handler
E>1484791455.406755999 [8342]: MTE_EVENT:sigfd_handler:737: sigfd_handler
E>1484791456.401631044 [8346]: PDE_FUSE:pde_fuse_run:467: NOTICE: fuse_main returned 0 -- FUSE thread exits
1484791456.418229: kernel module shutdown -- daemon exits

E>1484791456.418316615 [8374]: DEBUG: Thread 10.42.0.95 (8374) creates kthread iscsi_conn_cleanup
E>1484791456.418399448 [15455]: MTE_SERVICE:sys_thread_fn:110: NOTICE: thread iscsi_conn_cleanup @0x7f2f30003dc0 starts up on tid=15455
E>1484791456.418507413 [8352]: scst_event_queue_tm_fn_received:scst_event_queue_tm_fn_received:315: WARNING: SKIP queuing event scst_even
E>1484791456.447158099 [8374]: MTE_EVENT:event_task_stop_onthread:854: event_task[8374:10.42.0.95] stopping
E>1484791456.447179205 [8374]: MTE_EVENT:event_task_loop:713: exit event_task event loop
E>1484791456.447184406 [8374]: MTE_EVENT:mte_event_task_run:839: NOTICE: event_task 10.42.0.95 @0x226dc00 disengaged -- tid=8374 epoll_fd
E>1484791456.447287600 [15455]: MTE_EVENT:mte_event_task_free:945: mte_event_task[8374:10.42.0.95] freeing -- stats:
  8374      10.42.0.95 steps=2159405 age_sec=12616 last_hb_ms_ago=0 DISENGAGED  AWAKE active_polls=0 pending_alarms=0 queued_work=0 queue

E>1484791456.449531436 [8352]: scst_event_queue_tm_fn_received:scst_event_queue_tm_fn_received:315: WARNING: SKIP queuing event scst_even
E>1484791458.021281139 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:1 for template iscsi unregistered successfully
E>1484791458.021357110 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:2 for template iscsi unregistered successfully
E>1484791458.021383069 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:3 for template iscsi unregistered successfully
E>1484791458.021407684 [8341]: INFO: scst: Target iqn.2001-04.com.example.blackbox:4 for template iscsi unregistered successfully
E>1484791458.384895644 [8354]: INFO: iscsi-scst: Write thread for pool 0x2262fc0 finished
E>1484791458.628928859 [8355]: INFO: iscsi-scst: Write thread for pool 0x2262fc0 finished
E>1484791458.632574964 [8341]: INFO: scst: Target template iscsi unregistered successfully
E>1484791458.632623072 [8341]: INFO: iscsi-scst: Unregistered iSCSI transport: iSCSI-TCP
E>1484791458.632666005 [8341]: INFO: dev_vdisk: Detached virtual device null_NULLIO ("/dev/zero")
E>1484791458.632683729 [8341]: INFO: scst: Removed LUN 0 from group cian (target ?)
E>1484791458.632698452 [8341]: INFO: scst: Removed LUN 0 from group Default (target ?)
E>1484791458.632745003 [8341]: INFO: scst: Detached from virtual device null_NULLIO (id 1)
E>1484791458.632772215 [8341]: INFO: dev_vdisk: Virtual device null_NULLIO unregistered
...
E>1484791458.633157114 [8341]: INFO: dev_vdisk: Detached virtual device file_c2 ("/tmp/disk_c2")
E>1484791458.633167617 [8341]: INFO: scst: Removed LUN 32 from group cian (target ?)
E>1484791458.633188326 [8341]: INFO: scst: Detached from virtual device file_c2 (id 7)
E>1484791458.633205226 [8341]: INFO: dev_vdisk: Virtual device file_c2 unregistered
E>1484791458.633226968 [8341]: INFO: scst: Device handler "vdisk_nullio" unloaded
E>1484791458.633237467 [8341]: INFO: scst: Device handler "vdisk_blockio" unloaded
E>1484791458.633252186 [8341]: INFO: scst: Device handler "vdisk_fileio" unloaded
E>1484791458.633264784 [8341]: INFO: scst: Device handler "vcdrom" unloaded
E>1484791458.900269724 [8352]: INFO: scst: Task management thread finished
E>1484791459.071553130 [8353]: INFO: scst: Management thread finished
E>1484791459.207855071 [8351]: INFO: scst: Init thread finished
E>1484791459.562597258 [8341]: INFO: scst: SCST unloaded
E>1484791459.562679765 [8342]: MTE_EVENT:event_task_stop_onthread:854: event_task[8342:UMC_irqthread] stopping
E>1484791459.562695266 [8342]: MTE_EVENT:event_task_loop:713: exit event_task event loop
E>1484791459.562708870 [8342]: MTE_EVENT:mte_event_task_run:839: NOTICE: event_task UMC_irqthread @0x1f5d200 disengaged -- tid=8342 epoll
E>1484791459.562768652 [8341]: MTE_EVENT:mte_event_task_free:945: mte_event_task[8342:UMC_irqthread] freeing -- stats:
  8342   UMC_irqthread steps=295 age_sec=12623 last_hb_ms_ago=0 DISENGAGED  AWAKE active_polls=0 pending_alarms=0 queued_work=0 queued_wo
E>1484791459.918698132 [8341]: MTE_MEM:mem_arena_destroy:384: NOTICE: Successfully destroyed arena
dave@blackbox:~/src$                                        # ^ means all memory allocations were freed
```

Initiator and Server Machine Settings

Note: These scripts won't work as they are seen here — they are only here to indicate what settings I changed from Linux-distribution defaults. Also, some of these settings are probably unnecessary or unhelpful — I have not carefully determined which ones are useful.

```
#!/bin/bash
# SCST_config_common.bash -- iSCSI performance testing Initiator/Server common configuration script

if [ -d /sys/devices/system/cpu/cpu0/cpufreq ]; then
    for f in /sys/devices/system/cpu/cpu*/cpufreq ; do cat $f/cpuinfo_max_freq > $f/scaling_max_freq ; done
    for f in /sys/devices/system/cpu/cpu*/cpufreq ; do cat $f/scaling_max_freq > $f/scaling_min_freq ; done
    sleep 1
    echo "Set CPU frequency to " `cat /sys/devices/system/cpu/cpu*/cpufreq/scaling_cur_freq`
else echo "No cpufreq"; fi

if [ -d /sys/devices/system/cpu/cpufreq/policy0/scaling_governor ]; then
    for f in /sys/devices/system/cpu/cpufreq/policy*/scaling_governor ; do echo performance > $f ; done
    echo "Set scaling_governor to " `cat /sys/devices/system/cpu/cpufreq/policy*/scaling_governor`
else echo "No scaling_governor"; fi

if [ -d /sys/devices/system/cpu/intel_pstate ]; then
    echo 100 > /sys/devices/system/cpu/intel_pstate/max_perf_pct
    echo 100 > /sys/devices/system/cpu/intel_pstate/min_perf_pct
    echo   1 > /sys/devices/system/cpu/intel_pstate/no_turbo    # want consistent more than speedy
    echo "Set intel_pstate"
else echo "No intel_pstate"; fi

NETSIZE=16777216
/bin/echo ${NETSIZE} > /proc/sys/net/core/rmem_max
/bin/echo ${NETSIZE} > /proc/sys/net/core/wmem_max
/bin/cat /proc/sys/net/core/rmem_max > /proc/sys/net/core/rmem_default  # unneeded for TCP?
/bin/cat /proc/sys/net/core/wmem_max > /proc/sys/net/core/wmem_default  # unneeded for TCP?

/bin/echo -e "4096\t${NETSIZE}\t${NETSIZE}" > /proc/sys/net/ipv4/tcp_rmem
/bin/echo -e "4096\t${NETSIZE}\t${NETSIZE}" > /proc/sys/net/ipv4/tcp_wmem

/bin/echo ${NETSIZE} > /proc/sys/net/ipv4/tcp_limit_output_bytes
/bin/echo       4096 > /proc/sys/net/core/netdev_max_backlog
/bin/echo          0 > /proc/sys/net/ipv4/tcp_moderate_rcvbuf

NETS=`ifconfig -s | sed -e "1d" -e "s/ .*//" | egrep -v "^lo$"`
for n in $NETS; do
    ifconfig         $n mtu     9000                # jumbo frame ethernet
    /sbin/ethtool -G $n rx      4095
    /sbin/ethtool -G $n tx      4095
done
echo "Set network MTU: " ${NETS}
```

```
#!/bin/bash
# SCST_config_server -- iSCSI Server configuration script
# /proc/version_signature: Ubuntu 3.13.0-101.148-generic 3.13.11-ckt39
echo "iSCSI Server configuration"

. SCST_config_common.bash
for f in /sys/block/sd[abc]/queue/scheduler; do echo cfq > $f ; done
for f in /dev/sd[abc]; do blockdev --setra 1024 $f; done
```

```
#!/bin/bash
# SCST_config_initiator -- iSCSI Initiator configuration script
# /proc/version_signature: Ubuntu 4.4.0-45.66-generic 4.4.21
echo "iSCSI Initiator configuration"

iscsiadm --mode node --logout ; sleep 3

. SCST_config_common.bash

for n in $NETS; do
    /sbin/ethtool -C $n rx-usecs       4
done

# Rediscover session parameters that may be cached (assumes Initiator is 10.42.x.y and Server is 10.42.x.1)
sleep 1
IP=`ifconfig | grep 10.42 | sed -e "s/.*inet addr://" -e "s/ .*//" | head -1 | sed -e "s/\(10\.42\.[0-9]*\.\).*/\11/"`
if [ -z $IP ] ; then echo "Unable to determine address for discovery" ; exit -1 ; fi

DISC=`iscsiadm --mode discoverydb --type sendtargets --portal $IP --discover | grep $IP | head -1`
if [ -z "$DISC" ] ; then echo "Unable to determine target at $IP" ; exit -2 ; fi
PORT=`echo $DISC | sed -e "s/ .*//"`
TGT=`echo $DISC | sed -e "s/.* //"`
echo PORT: $PORT TGT: $TGT

sleep 1
iscsiadm --mode node --login --portal $PORT --target $TGT
sleep 5
### Now our block devices should exist

# Increase number of requests that can be outstanding to the device at one time (at some layer or another)
for f in /sys/devices/platform/host*/session*/target*/*/block/sd[b-z]/queue/nr_requests ; do /bin/echo 1280 > $f; done

# Decrease max initiator-side coalescing in sectors from default 32768 to cap I/O size
for f in /sys/devices/platform/host*/session*/target*/*/block/sd[b-z]/queue/max_sectors_kb ; do /bin/echo 8192 > $f; done

# Readahead in sectors -- unchecked rumor says this must be set *after* max_sectors_kb
for f in /dev/sd[b-z]; do blockdev --setra 1024 $f; done

echo 0 > /proc/sys/net/ipv4/tcp_autocorking         # disable autocorking

# Coalescing disabled for random I/O testing, enabled for sequential
for f in /sys/devices/platform/host*/*/*/*/block/*/queue/nomerges ; do echo 2 > $f ; done # disables coalescing

# Not sure this helped -- retest
for f in /sys/class/net/*/queues/tx-0/byte_queue_limits/limit_min ; do /bin/echo 9000 > $f ; done
```

iSCSI-SCST Usermode Adaptation — Appendix: Issues Discovered

David A. Butterfield — February 2017

See also the *Issues Summary* section in the main paper. These diffs are against sourceforge.net/projects/scst **scst-svn-7089-trunk**

+++ scst/iscsi-scst/kernel/nthread.c
```
close_conn_thr()...
        close_conn(conn);
        TRACE_EXIT();
-       return 0;
+       do_exit(0);         //XXX Right?  No one does kthread_stop() on this one
 }
```

This comment in kthread.c kthread_create_on_node() seems to imply that a kernel thread that exits on its own (without anyone calling kthread_stop on it) is supposed to call do_exit() rather than returning from the initial thread function. "*threadfn() can either call do_exit() directly if it is a standalone thread for which no one will call kthread_stop(), or return when 'kthread_should_stop()' is true (which means kthread_stop() has been called).*" However this does not 100% unambiguously rule out a standalone thread returning instead of calling do_exit, so I'm not sure the "return" is actually wrong.

+++ scst/iscsi-scst/kernel/session.c
```
 #ifdef CONFIG_SCST_PROC
        name = kmalloc(strlen(info->user_name) + strlen(info->initiator_name) +
-                       1, GFP_KERNEL);
+                       2, GFP_KERNEL); // +1 (for '\0') +1 (for '@')  XXX Right?
        if (name == NULL) {
                err = -ENOMEM;
                goto err;
         }
         if (info->user_name[0] != '\0')
                 sprintf(name, "%s@%s", info->user_name, info->initiator_name);
         else
                 sprintf(name, "%s", info->initiator_name);
```

This appears to be a potential buffer overflow due to short computation of needed string size.

+++ scst/iscsi-scst/kernel/iscsi.c
```
        req->scst_state = ISCSI_CMD_STATE_RX_CMD;
        conn->rx_task = current;
        scst_cmd_init_stage1_done(scst_cmd, SCST_CONTEXT_DIRECT, 0);
+       conn->rx_task = NULL;       //XXX Right?

        if (req->scst_state != ISCSI_CMD_STATE_RX_CMD)
                res = req->conn->transport->iscsit_receive_cmnd_data(req);
```

It looks like conn->rx_task is intended to be set only during the call to _stage1_done, but there was no code to reset it. I think it probably doesn't actually matter in execution, but it's a little confusing to wonder and try to understand why it wasn't reset.

+++ scst/iscsi-scst/kernel/conn.c
```
@@ -397
         */
        list_for_each_entry_reverse(conn, &session->conn_list,
                                    conn_list_entry) {
-               if (conn->cid == cid)
+               if (conn->cid == cid && !conn->closing) //XXX Right?
                        return conn;
        }
        return NULL;
 }
```

The above change helped a secondary problem I had under valgrind (due to some other bug) with initiators timing out and reconnecting their sessions faster than the threads were getting cleaned up (for one thing, valgrind only ever runs one thread at a time).

It seems logical that the above code in **conn_lookup()** would want to skip any connection that's already known closing in this case, for the same reason it searches the list in reverse. And the dropping connections don't have to drop in order, so searching in reverse doesn't seem sufficient to avoid finding a wrong (stale, closing) connection structure. *[This should be reviewed by someone more familiar with the code.]*

+++ scst/scst/src/scst_main.c
```
 #ifdef CONFIG_SCST_PROC
+       mutex_lock(&scst_mutex);        //XXX Right?
        scst_del_free_acg(scst_default_acg, false);
+       mutex_unlock(&scst_mutex);
 #endif
```

scst_del_free_acg() does lockdep_assert_held(&scst_mutex), so we'd better take the lock first. *[Yes, I simulated lockdep_assert_held, and it triggered]*

+++ scst/scst/src/scst_no_dlm.c
```
static bool scst_no_dlm_reserved(struct scst_device *dev)
 {
-       return dev->reserved_by;
+       return (dev->reserved_by != NULL);
 }
```

The return value is expected to be boolean (value zero or one), but what was being returned was (value zero or nonzero). (I think this was from an extra compiler warning I enabled.)

```
+++ scst/iscsi-scst/kernel/session.c
@@ -101
        if (!session->sess_params.rdma_extensions) {
                err = iscsi_threads_pool_get(
-                       (bool)scst_get_acg_tgt_priv(session->scst_sess->acg),
+                       scst_get_acg_tgt_priv(session->scst_sess->acg) != NULL,
                        &session->scst_sess->acg->acg_cpu_mask,
                        &session->sess_thr_pool);
                if (err != 0)
```

With some extra warnings enabled gcc would produce

```
 session.c:103:4: warning: cast from pointer to integer of different size [-Wpointer-to-int-cast]
 session.c:103:4: warning: cast from function call of type 'void *' to non-matching type 'unsigned char' [-Wbad-function-cast]
```

Another abuse of boolean, but I just realized subtly much nastier than the one above. Here the cast of the pointer to a boolean will return "false" for any pointer aligned to a 256-byte boundary, and "true" for all others (which, without checking, I boldly assume was not the intended semantic).

```
+++ scst/scst/src/scst_proc.c
 static int scst_proc_group_add(const char *p, unsigned int addr_method)
 {
-       int res = 0, len = strlen(p) + 1;
+       int res = 0;
        struct scst_acg *acg;
-       char *name = NULL;
        TRACE_ENTRY();

-       name = kmalloc(len, GFP_KERNEL);
-       if (name == NULL) {
-               PRINT_ERROR("Allocation of new name (size %d) failed", len);
-               goto out_nomem;
-       }
-       strlcpy(name, p, len);
-
-       res = scst_alloc_add_acg(NULL, name, false, &acg);
+       res = scst_alloc_add_acg(NULL, p, false, &acg);
        if (res != 0) {
-               PRINT_ERROR("scst_alloc_add_acg() (name %s) failed", name);
-               goto out_free;
+               PRINT_ERROR("scst_alloc_add_acg() (name %s) failed", p);
+               goto out;
        }
@@ -968
-out_free:
-       kfree(name);
-       goto out;
-out_nomem:
        res = -ENOMEM;
        goto out;
 }
```

Valgrind noticed that the "name" allocated here in scst_proc_group_add() was leaking — it turns out that scst_alloc_add_acg makes its own copy of the name passed to it from this code, making the string duplication done here redundant (and leaky). The change eliminates the string duplication (along with all its associated error handling logic) and simply passes the (unowned) incoming string down.

```
+++ scst/scst/src/scst_proc.c
        /* We may not bother about locks here */
        scst_proc_cleanup_sgv();
+
+       //XXX the lockdep_assert_held() in scst_del_free_acg disagrees with the comment above
+       mutex_lock(&scst_mutex);
        scst_proc_cleanup_groups();
+       mutex_unlock(&scst_mutex);
```

Another call that ends up at the lockdep_assert_held() in scst_del_free_acg() without locking. Also see code comment.

```
+++ scst/scst/src/dev_handlers/scst_vdisk.c
 #define SCST_FIO_VENDOR                        "SCST_FIO"
 #define SCST_BIO_VENDOR                        "SCST_BIO"
 /* 4 byte ASCII Product Revision Level - left aligned */
+                  //XXX Is SCST_FIO_REV really "left aligned" ?
 #define SCST_FIO_REV                   " 320"
```

Looks "right aligned" to me, in contradiction to the comment just above it.

```
+++ scst/scst/src/scst_lib.c
static void scst_del_acg(struct scst_acg *acg)
...
        list_for_each_entry_safe(acn, acnt, &acg->acn_list, acn_list_entry)
-               scst_del_acn(acn);
+               scst_del_free_acn(acn, false);      //XXX Right?
```

I think this was another one valgrind turned up as a leak. *[Fix needs review]*

```
+++ scst/scst/src/scst_lib.c
@@ -4428
        } else
                acg_dev->acg_dev_dif_guard_format =
+                   acg->tgt &&     /* sometimes NULL */
                        acg->tgt->tgt_hw_dif_ip_supported && !dev->dev_dif_ip_not_supported ?
                                                        SCST_DIF_GUARD_FORMAT_IP :
                                                        SCST_DIF_GUARD_FORMAT_CRC;
```

The dereference of acg->tgt gave me a SEGV a few times at one point (though I had other bugs at the time so possibly this case "cannot happen" now that those are fixed). I didn't really analyze this one much, just added the check that acg->tgt is nonzero before dereferencing it.

```
+++ scst/scst/src/dev_handlers/scst_vdisk.c
                        PRINT_ERROR("File path \"%s\" is not "
                                "absolute", filename);
                        res = -EINVAL;
-                       goto out_up;
+                       goto out_free_vdev;     //XXX Right?
                }

                virt_dev->filename = kstrdup(filename, GFP_KERNEL);
```

Another leak valgrind popped out.

```
+++ scst/scst/src/dev_handlers/scst_vdisk.c
static int vdisk_attach(struct scst_device *dev)
...
        dev->dev_rd_only = virt_dev->rd_only;

+       //XXX Is this right?  Should it be under #ifdef CONFIG_SCST_PROC?
+       if (virt_dev->nullio && !virt_dev->file_size) {
+           virt_dev->file_size = VDISK_NULLIO_SIZE;
+       }
+
        res = vdisk_reexamine(virt_dev);
        if (res < 0)
                goto out;
```

The file_size wasn't getting set for NULLIO; maybe only a problem with /proc support.

```
+++ scst/scst/include/scst.h
@@ -4065
static inline enum scst_exec_context __scst_estimate_context(bool atomic)
{
        if (in_irq())
                return SCST_CONTEXT_TASKLET;
/*
 * We come here from many non reliable places, like the block layer, and don't
 * have any reliable way to detect if we called under atomic context or not
 * (in_atomic() isn't reliable), so let's be safe and disable this section
 * for now to unconditionally return thread context.
 */
#if 0
        else if (irqs_disabled())
                return SCST_CONTEXT_THREAD;
        else if (in_atomic())
                return SCST_CONTEXT_DIRECT_ATOMIC;
        else
+               //XXX Isn't this backwards?  Could it explain "non-reliability"?
                return atomic ? SCST_CONTEXT_DIRECT :
                                SCST_CONTEXT_DIRECT_ATOMIC;
#else
        return SCST_CONTEXT_THREAD;
#endif
}
```

Looking no further than the local code and symbol names, it seems like the final return statement has the test backwards, returning ATOMIC in the non-atomic case. (But I haven't tried tracking down callers to see what they actually expect, and the Usermode Adaptation doesn't use this logic — I didn't change anything here.)

```
+++ scst/iscsi-scst/usr/chap.c
static u8 decode_base64_digit(char base64)
@@ -105
                else if ((base64 >= '0') && (base64 <= '9'))
                        return 52 + (base64 - '0');
                else
+                       //XXX This return value should be unsigned; and anyway
+                       //XXX in case of a bad character in the string, our
+                       //XXX caller (sometimes) checks for 65, not 255 or -1
                        return -1;
```

I didn't fix this one — but the caller checks the return value for a different error value than the function returns.

```
+++ scst/iscsi-scst/Makefile
-all: include/iscsi_scst_itf_ver.h progs mods
+all: include/iscsi_scst_itf_ver.h
+	@$(MAKE) -C . progs mods
```

The Makefile change is to ensure that iscsi_scst_itf_ver.h exists before mods tries to #include it — otherwise they race in parallel and the #include can fail. (There must be a better way to fix this.)

When that "make" race intermittently failed after "make extraclean" it was obvious, because the missing include file would cause a compile error. More subtly, I suppose that if there happened to be an old version of iscsi_scst_itf_ver.h lying around from a prior build, the same race would result not in an overt failure, but in an older, stale version number appearing in the executable.

```
+++ scst/scst/src/dev_handlers/scst_vdisk.c
@@ -1157
	} else if (S_ISBLK(inode->i_mode)) {
		inode = inode->i_bdev->bd_inode;
	} else {
+		PRINT_ERROR("File %s smells bad: mode=0%o\n",
+				filename, inode->i_mode);
		res = -EINVAL;
		goto out_close;
	}
@@ -8116
		} else if (!strcasecmp("blocksize", p)) {
			virt_dev->blk_shift = scst_calc_block_shift(val);
			if (virt_dev->blk_shift < 9) {
+				PRINT_ERROR("blocksize %u too small", 1<<virt_dev->blk_shift);
				res = -EINVAL;
				goto out;
			}
@@ -10344
			block_shift = scst_calc_block_shift(block_size);
			if (block_shift < 9) {
+				PRINT_ERROR("blocksize %u too small", 1<<block_shift);
				res = -EINVAL;
				goto out_free_vdev;
			}
```

That is some error logging I added so I could see what error path was returning me an EINVAL.

```
+++ scst/iscsi-scst/usr/iscsi_scstd.c
	iser_fd = create_and_open_dev("isert_scst", 1);

-	poll_array[POLL_ISER_LISTEN].fd = iser_fd;
-	if (iser_fd != -1) {
+	if (iser_fd >= 0) {
+		poll_array[POLL_ISER_LISTEN].fd = iser_fd;
		poll_array[POLL_ISER_LISTEN].events = POLLIN;

		/* RDMAExtensions */
		session_keys[key_rdma_extensions].max = 1;
		session_keys[key_rdma_extensions].local_def = 1;
	} else {
+		poll_array[POLL_ISER_LISTEN].fd = -1;
		poll_array[POLL_ISER_LISTEN].events = 0;
		return;
	}
```

create_and_open_dev() returns a (-errno), so the "if (iser_fd...)" check should detect *any* negative return value as a case when fd should set to -1.

```
+++ scst/iscsi-scst/usr/config.c
@@ -856
	if (type == key_session) {
		for (i = 0; i < session_key_last; i++) {
+			uint32_t in_val = params[i].val;
			if (partial & (1 << i)) {
				err = params_check_val(session_keys, i, &params[i].val);
				if (err < 0) {
-					log_error("Wrong value %u for parameter %s\n",
-						params[i].val, session_keys[i].name);
+					log_error("%s: Wrong value %u->%u for session parameter %s\n",
+						__func__, in_val, params[i].val, session_keys[i].name);
					goto out;
				}
			}
```

Above is one of four similar instances (all near each other in the source file) of a very misleading and confusing logging statement, which would print a "Wrong value" that had already been corrected by the check function.

```
+++ scst/iscsi-scst/usr/target.c
@@ -415
-       memcpy(target->name, name, sizeof(target->name) - 1);
+       strncpy(target->name, name, sizeof(target->name) - 1);
```

The source name string is not guaranteed to exist as valid addressable memory beyond the NUL byte. Valgrind probably said something that led me to this one.

```
+++ scst/usr/stpgd/stpgd_main.c
-               sleep(0.1);
+               usleep(100*1000);
```

The argument to sleep() gets "promoted" to an integer type with value zero.

```
+++ scst/scst/src/scst_targ.c   2017-01-10 18:24:25.715336806 -0700
int scst_cmd_thread(void *arg)
{
        ...
        spin_lock_irq(&p_cmd_threads->cmd_list_lock);
        spin_lock(&thr->thr_cmd_list_lock);
        while (!kthread_should_stop()) {
                if (!test_cmd_threads(thr)) {
                        DEFINE_WAIT(wait);

                        do {
                                prepare_to_wait_exclusive_head(
                                        &p_cmd_threads->cmd_list_waitQ,
                                        &wait, TASK_INTERRUPTIBLE);
                                if (test_cmd_threads(thr))
                                        break;
                                spin_unlock(&thr->thr_cmd_list_lock);
                                spin_unlock_irq(&p_cmd_threads->cmd_list_lock);
                                schedule();
                                spin_lock_irq(&p_cmd_threads->cmd_list_lock);
                                spin_lock(&thr->thr_cmd_list_lock);
                        } while (!test_cmd_threads(thr));
                        finish_wait(&p_cmd_threads->cmd_list_waitQ, &wait);
                }

                if (tm_dbg_is_release()) {
                        spin_unlock_irq(&p_cmd_threads->cmd_list_lock);
                        tm_dbg_check_released_cmds();
+                       /* XXX    acquires p while holding thr    XXX */
                        spin_lock_irq(&p_cmd_threads->cmd_list_lock);
                }
```

It looks like the "spin_lock_irq()" call shown highlighted in the listing of scst_cmd_thread() above is attempting to acquire p_cmd_threads->cmd_list_lock while holding thr->thr_cmd_list_lock; but at the top of the function and elsewhere the two were locked in the opposite order (a recipe for deadlock).

To me the locking in this function seems terribly convoluted and difficult to prove correct, particularly in the code that immediately follows what's quoted just above. Other than the lock reversal just mentioned I haven't identified any additional suspects, but it isn't easy to gain confidence in its correctness either. (If I'm missing some mental model that makes the existing implementation straightforward to reason about, I'd suggest adding a comment nearby to explain it.)

I'm suspecting the complexity is the result of an effort to eke out every last nanosecond of unnecessary lock contention — but unless it has been actually measured and demonstrated to improve macroscopically-measurable performance, I'm skeptical that it makes any difference, especially given the additional "if (locked)" conditional branches used to avoid (rarely?) unnecessary lock operations. In any case it's hard to believe it's worth the resulting code opacity. But maybe I'm just looking at it wrong.

Below is an attempt at fixing the lock-reversal problem and also improving the readability of the central portion of scst_cmd_thread() (hopefully without breaking it, after all that commentary). The diffs are too messy to read, so this lists the new code instead, with "..." indicating unchanged sections.

The "do { ... } while (someth_done)" in the bottom 2/3 of the listing encloses most of the change concerning readability; comparing it with the original by eye, the difference in the locking logic should be apparent — simplified enough to get rid of the variables tracking lock state. The top 1/3 is here to show the fix for the lock acquisition reversal, and the context of the rest of the lock manipulations in the function.

Note that the first (single-iteration) "for" loop is only there for looks (symmetry with the second "for" loop): gcc -O2 doesn't generate any instructions for looping logic in this case.

The revised code is shorter and easier to understand, but is it correct? It was running throughout all my performance testing... but on the other hand it ran for a long time with a bug I introduced there before I noticed it in reviewing my changes — a bug that would make no difference in the Usermode Adaptation. Obviously any change like this should receive very careful review before being integrated anywhere considered "stable".

```
int scst_cmd_thread(void *arg)
        ...
        /* Hold both locks for the test_cmd_threads() checks */
        /* Lock acquisition order is always:  First p_cmd_threads, Then thr */
        spin_lock_irq(&p_cmd_threads->cmd_list_lock);
        spin_lock(&thr->thr_cmd_list_lock);
        while (!kthread_should_stop()) {
                if (!test_cmd_threads(thr)) {
                        DEFINE_WAIT(wait);
                        do {
                                prepare_to_wait_exclusive_head(
                                        &p_cmd_threads->cmd_list_waitQ,
                                        &wait, TASK_INTERRUPTIBLE);
                                if (test_cmd_threads(thr))
                                        break;
                                spin_unlock(&thr->thr_cmd_list_lock);
                                spin_unlock_irq(&p_cmd_threads->cmd_list_lock);
                                schedule();
                                spin_lock_irq(&p_cmd_threads->cmd_list_lock);
                                spin_lock(&thr->thr_cmd_list_lock);
                        } while (!test_cmd_threads(thr));
                        finish_wait(&p_cmd_threads->cmd_list_waitQ, &wait);
                }

                /* Drop both locks now that we are through the test_cmd_threads() checks */
                spin_unlock(&thr->thr_cmd_list_lock);
                spin_unlock_irq(&p_cmd_threads->cmd_list_lock);

                if (tm_dbg_is_release()) {
                        tm_dbg_check_released_cmds();
                }
                ...
                do {
                        int thr_cnt;
                        struct scst_cmd *cmd;
                        someth_done = false;
                        for (thr_cnt = 0; thr_cnt < 1; thr_cnt++) {
                                cmd = NULL;
                                spin_lock_irq(&p_cmd_threads->cmd_list_lock);
                                if (!list_empty(&p_cmd_threads->active_cmd_list)) {
                                        cmd = list_first_entry(&p_cmd_threads->active_cmd_list,
                                                                typeof(*cmd), cmd_list_entry);
                                        TRACE_DBG("Deleting cmd %p from active cmd list", cmd);
                                        list_del(&cmd->cmd_list_entry);
                                }
                                spin_unlock_irq(&p_cmd_threads->cmd_list_lock);

                                if (!cmd) break;

                                if (cmd->cmd_thr == NULL) {
                                        TRACE_DBG("Assigning thread %p on cmd %p", thr, cmd);
                                        cmd->cmd_thr = thr;
                                }
                                scst_process_active_cmd(cmd, false);
                                someth_done = true;
                        }
                        for (thr_cnt = 0; thr_cnt < 2; thr_cnt++) {
                                cmd = NULL;
                                spin_lock(&thr->thr_cmd_list_lock);
                                if (!list_empty(&thr->thr_active_cmd_list)) {
                                        cmd = list_first_entry(&thr->thr_active_cmd_list,
                                                               typeof(*cmd), cmd_list_entry);
                                        TRACE_DBG("Deleting cmd %p from thr active cmd list", cmd);
                                        list_del(&cmd->cmd_list_entry);
                                }
                                spin_unlock(&thr->thr_cmd_list_lock);

                                if (!cmd) break;
                                scst_process_active_cmd(cmd, false);
                                someth_done = true;
                        }
                } while (someth_done);
                ...
                spin_lock_irq(&p_cmd_threads->cmd_list_lock);
                spin_lock(&thr->thr_cmd_list_lock);
        }
        spin_unlock(&thr->thr_cmd_list_lock);
        spin_unlock_irq(&p_cmd_threads->cmd_list_lock);
        ...
}
```

iSCSI-SCST Storage Server Usermode Adaptation

David A. Butterfield — January 2017

Abstract

This paper describes an adaptation of the iSCSI-SCST storage server software to run entirely in usermode on an unmodified Linux kernel; performance measurements and model; and an experimental algorithm to improve performance for small Read operations. The Appendix lists a few issues discovered in the SCST source code.

In a standard installation of SCST the iscsi-scstd daemon runs as a single-threaded Linux usermode process that cooperates with the kernel-resident SCST implementation using ioctl(2) and netlink(7) for communication.

In the iSCSI-SCST Usermode Adaptation the iscsi-scstd daemon runs on the main thread in a multi-threaded process in which other usermode threads are concurrently providing the services and executing the SCST code that would be running inside the kernel in a standard installation of SCST.

The subset of SCST used includes the SCST Core, the iSCSI daemon and kernel logic, the vdisk device, and the /proc interface; comprising about 80,000 lines of code. To support running in usermode, around 55 (fifty-five) lines of executable C code have been added or changed under #ifdef in SCST source files.

For a single session over 1 Gb Ethernet being serviced by a single 2.4 GHz CPU: the described *Adaptive Nagle* optimization improves peak throughput performance for 512-Byte Random Read of /dev/zero from around 63,000 IOPS to more than 100,000 IOPS, with no adverse impact below Queue Depth 17.

Contents

Diagrams (PDF)

About the Author

David Butterfield *[david.butterfield@acm.org]* began programming for usermode in 2008 after a prior gigasecond or so working on software in various versions of the Unix and Solaris kernels (or without any kernel). He holds an MSc in Computer Science from UCLA, where his undergraduate degree was in Mathematics and Computer Science.

One of the founders of Locus Computing Corporation, he designed the first Virtual Machine Monitor for x86, to "*Merge*" MS-DOS and its applications under Unix SVR2; and led an engineering team in its implementation. The *OS-Merge* product was first marketed by AT&T under the name "*Simultask*" on their 6300+ (IBM AT clone) model. *[Sometime after his involvement that product evolved into two descendant products known as NeTraverse Merge and Win4Lin]*

He joined Sun Microsystems to establish and lead the first Solaris x86 device driver development team, later accepting an international assignment to Dublin, Ireland to start another driver engineering team there.

Back in the U.S., at LeftHand Networks he contributed many performance improvements to the SAN/iQ event-driven distributed storage application, introducing application-transparent multi-threading into the existing single-threaded event framework and devising other optimizations amounting in total to a 2.5x increase in throughput (IOPS) capability. Some of his diagrams were said to inspire awe.

His most recent (unpaid) project is described in this paper. He is presently looking for an interesting (well-paid) project.