SCST, a SCSI Target Framework

Bart Van Assche, Ph.D.

August 19, 2015

Overview

- What is SCST
  - Why SAN systems are useful
  - The SCSI protocol
  - Supported initiator operating systems
  - SCST architecture
- How to use SCST
  - Configuring SCST
  - Session information
  - SCST and the Linux storage stack
  - Building an H.A. system with SCST
  - Troubleshooting a SCSI setup
  - SCST Documentation
- How SCST is maintained
- SCST / LIO Unification Initiative

About myself

- Contributing to SCST since 2008.
- Kernel maintainer for SRP initiator driver.
- Added scsi-mq support to the Linux SRP initiator driver.
- Inside SanDisk, member of the ION Accelerator team.
- ION: H.A. SAN system that supports iSCSI, FC and SRP.

Why SAN systems are useful

- SAN = server that provides block storage
- From Webopedia:

  *Organizations often choose to deploy a storage area network because it offers better **flexibility**, **availability** and **performance** than direct-attached storage (DAS). Because a SAN removes storage from the servers and consolidates it in a place where it can be accessed by any application, it tends to improve storage utilization. Storage utilization improvements often allow organizations to defer purchases of additional storage hardware, which saves money and requires less space in the data center.*
- A SAN makes fast live migration possible, e.g. for virtual machines with VMware vMotion.

What is SCST

- SCST is a generic SCSI Target Subsystem for Linux.
- Software that converts any server into a SAN system.
- This approach is sometimes called *Software Based Storage* – a storage system that doesn't require any special hardware.
- Can be combined with other Linux storage layers, e.g.:
  - LVM – Logical Volume Manager.
  - RAID.
  - DRBD.

Adoption of SCST

- Several enterprise storage systems are based on SCST.
- See also http://scst.sourceforge.net/users.html.
- Many organizations use SCST in-house, including several large cloud computing and internet providers.
- Several open source projects are based on SCST:
  - Openfiler.
  - Enterprise Storage OS (ESOS).

The SCSI Protocol

- Well established standard.
- Supports reading and writing data.
- Supports multiple logical units (LUNs) per connection.
- Supports multiple transport protocols (iSCSI over TCP/IP, FC, SRP over InfiniBand, iSER over IB, RoCE or iWARP).
- Defines a protocol for coordination of multiple initiators a.k.a. persistent reservations.
- Allows a H.A. array to tell an initiator which path to use (ALUA = asymmetric logical unit access).
- A SCSI initiator driver is available for every major OS and for every storage HBA.

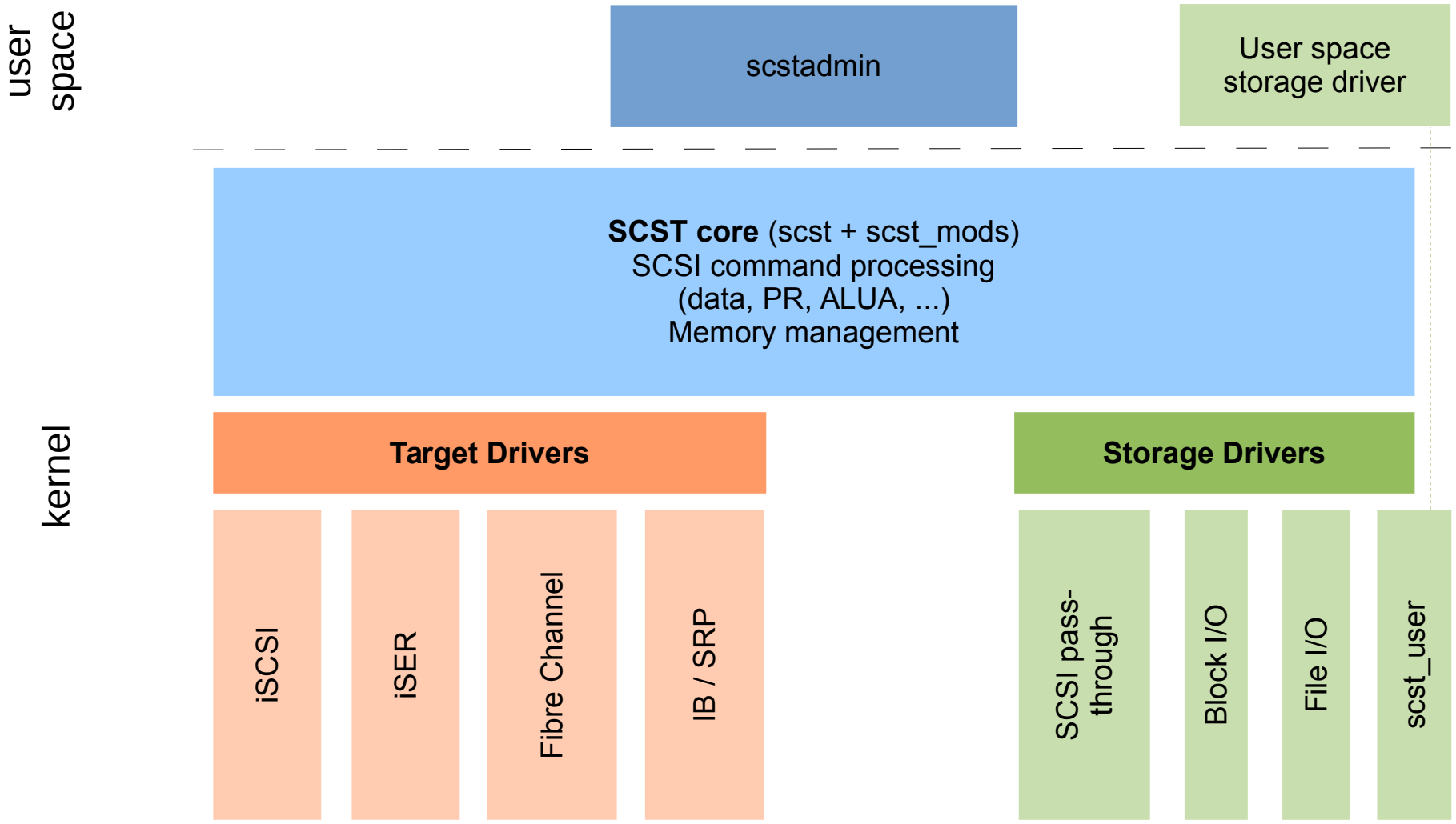
Supported Initiator Operating Systems

- Linux.
- Windows.
- VMware ESX and ESXi.
- Solaris.
- IBM AIX.
- HP/UX.

Supported Linux Kernel Versions

- SCST builds against the following kernel versions:
  - Greg KH's stable kernels, version v2.6.30.x .. v4.1.x.
  - RHEL 6.x and RHEL 7.x.
  - SLES 11 and SLES 12.
  - The RDMA drivers build against the in-tree RDMA stack, OpenFabrics OFED and Mellanox OFED.

SCST Architecture (1/2)



SCST Architecture (2/2)

- SCST core processes SCSI commands and routes these between target drivers and storage drivers.
- Target drivers implement a SCSI protocol and communicate with the initiator system.
- Storage drivers present local storage to the SCST core as a SCSI device. This can be a SCSI device, block device, file, or RAID controller.
- scst_user driver allows to implement a SCSI storage device in a user space process.

Configuring SCST

- Either via sysfs or via scstadmin.
- sysfs API exists under /sys/kernel/scst_tgt/.
- Allows to add a LUN, remove a LUN, export a LUN, enable/disable target ports, create initiator groups, …
- scstadmin makes the sysfs API easier to use.
- scstadmin also allows to save or restore the entire SCST configuration in text format:
  - scstadmin -write_config /etc/scst.conf
  - Scstadmin -config /etc/scst.conf

Presenting Storage Devices to SCST

- Examples using the scst.conf syntax:

```
HANDLER vdisk_blockio {
    DEVICE disk1 {
        filename /dev/sda5
    }
    DEVICE disk2 {
        filename /dev/disk/by-id/scsi-SATA_SanDisk_SD6SF1M143310400839
    }
}
HANDLER vdisk_fileio {
    DEVICE disk3 {
        filename /disk3
        blocksize 512
    }
    DEVICE disk3 {
        filename /disk4
        blocksize 4096
    }
}
```

Storage Port Identification

- Each initiator port and each target port has a unique name.
- FC: World Wide Name (WWN), e.g. 25:00:00:f0:98:87:92:f3.
- IB/SRP: IB HCA port GUID, e.g. fe80:0000:0000:0000:24be:05ff:ffa9:cbb1.
- iSCSI over TCP/IP and iSER: iSCSI Qualified Name (IQN), e.g. iqn.1992-01.com.example:storage.disk2.sys1.xyz.
- Multiple IQNs can be associated with the same Ethernet port.
- Storage port identifiers are globally unique. An organizationally unique identifier (OUI) is embedded in each WWN and GUID. A domain name is embedded in each IQN.

Initiator Groups

- A sample configuration that provides each initiator access to one LUN:

```
TARGET_DRIVER qla2x00t {
    TARGET 25:00:00:f0:98:87:92:f3 {
        enabled 1
        LUN 0 dummy
        GROUP ini1 {
            LUN 0 vol1
            INITIATOR 25:00:00:f0:98:87:92:f4
        }
        GROUP ini2 {
            LUN 0 vol2
            INITIATOR 25:00:00:f0:98:87:92:f5
        }
    }
}
```

SCST Initiator Session Information

- Information is available in sysfs about active sessions.
- Identity of initiator ports: targets/${target_driver}/${target_port}/sessions/${initiator_port}

  # (cd /sys/kernel/scst_tgt/targets && ls -d */*/sessions/*) | head -n1
  ib_srpt/fe80:0000:0000:0000:24be:05ff:ffa9:cbb1/sessions/fe80:0000:0000:0000:e41d:2d03:000a:6d51
- Exported LUNs per initiator:

  # (cd /sys/kernel/scst_tgt/targets && for d in */*/sessions/*/luns/*/device; do echo $(dirname $d) $(readlink $d | sed 's,.*/,,'); done)
  ib_srpt/fe80:0000:0000:0000:e41d:2d03:000a:8091/sessions/fe80:0000:0000:0000:e41d:2d03:000a:85b2/luns/0 vol0
  ib_srpt/fe80:0000:0000:0000:e41d:2d03:000a:8091/sessions/fe80:0000:0000:0000:e41d:2d03:000a:85b2/luns/1 vol1
- Statistics per session:

  # cat /sys/kernel/scst_tgt/targets/ib_srpt/.../sessions/.../active_commands
  3
- Tell an initiator to log out:

  # echo 1 > /sys/kernel/scst_tgt/targets/ib_srpt/.../sessions/.../force_close

SCST and the Linux Storage Stack

- SCST can export any file, block device or SCSI device.
- This includes:
  - md-raid arrays.
  - LVM volumes.
  - DRBD device nodes.
  - /dev/sd* nodes created by an iSCSI, FC or SRP initiator. A possible application of this is to use SCST as a SCSI protocol converter.
  - Regular files. Filesystems like BTRFS and ZFS support checksums, snapshots and deduplication.

Building an H.A. System with SCST

- Most popular configurations:
  - Two servers with shared storage, e.g. dual-port SAS drives.
  - Replication between storage servers with DRBD.
  - Initiator-side mirroring with md-raid.
- How to make a choice:
  - Lower storage cost with dual-ported drives but only one data copy.
  - Lowest write latency with initiator side mirroring because initiator submits write requests to the two storage servers concurrently.
  - Replication with DRBD works also for other initiator operating systems than Linux.
  - Rebuild time after a power failure is lower with DRBD because of the DRBD activity log. DRBD tracks recent writes in the activity log and resynchronizes replicas by replaying the activity log.
- SCST does not yet support persistent reservations for clusters.
- More about persistent reservation support for clusters during Linux Plumbers.

Troubleshooting a Storage Configuration

- The error messages that report a configuration error can be cryptic.
- Observing and analyzing SCSI communication can be a big help.
- For iSCSI, capture TCP/IP traffic with Wireshark.
- For SRP and iSER, capture IB traffic with ibdump (Mellanox OFED).
- For any protocol (including FC), enable SCSI packet logging in SCST:

```
echo add scsi > /sys/kernel/scst_tgt/trace_level
sleep 60
echo del scsi > /sys/kernel/scst_tgt/trace_level
```

- Result:

```
scst: op_name <TEST UNIT READY> (cmd ffff8807949bc7d8), direction=4 (expected 4,
  bufflen=0, out_bufflen=0, (expected len data 0, expected len DIF 0, out expecte
scst: cmd ffff8807949bc7d8, status 0, msg_status 0, host_status 0, driver_status
scst: tag=131072, lun=1, CDB len=16, queue_type=1 (cmd ffff8807949bca48, sess fff
Received CDB:
 (h)___0__1__2__3__4__5__6__7__8__9__A__B__C__D__E__F
   0: 12 00 00 00 60 00 00 00 00 00 00 00 00 00 00 00   ....`...........
```

What if a LUN does not show up (1/2) ?

- Check the session information in sysfs to verify whether login succeeded:

```
# (cd /sys/kernel/scst_tgt/targets && ls */*/sessions)
ib_srpt/fe80:0000:0000:0000:24be:05ff:ffa9:cbb1/sessions:
fe80:0000:0000:0000:0002:c903:00a0:2dc1
fe80:0000:0000:0000:0002:c903:00a0:2dc2
```

- Check whether the LUN has been exported:

```
# (cd /sys/kernel/scst_tgt/targets && for d in
*/*/sessions/*/luns/*/device; do echo $(dirname $d) $
(readlink $d | sed 's,.*/,,'); done)
ib_srpt/fe80:0000:0000:0000:e41d:2d03:000a:8091/sessions/fe8
0:0000:0000:0000:e41d:2d03:000a:6d51_1/luns/0 vol0
ib_srpt/fe80:0000:0000:0000:e41d:2d03:000a:8091/sessions/fe8
0:0000:0000:0000:e41d:2d03:000a:6d51_1/luns/1 vol1
```

What if a LUN does not show up (2/2) ?

- Use lsscsi and sg3_utils to verify the initiator side. Examples:

```
# sg_luns /dev/sdc
Lun list length = 32 which implies 4 lun entries
Report luns [select_report=0x0]:
    0000000000000000
    0001000000000000
    0002000000000000
    0003000000000000
# sg_persist --read-status /dev/sdc
  FUSIONIO  ION LUN           3243
  Peripheral device type: disk
  PR generation=0x0
  No full status descriptors
```

How SCST is Maintained

- User and developer feedback is posted on the scst-devel mailing list.
- Most questions are answered within a few days.
- Either by an SCST user or one of the SCST maintainers.
- Every night SCST is compile tested against 35 different kernel versions (2.6.27 – 4.1).
- Developed against upstream kernel and later on ported to enterprise kernels like RHEL and SLES.
- Periodically tested with libiscsi test suite. Analyzing test results not only resulted in SCST patches but also in libiscsi patches. See also https://github.com/sahlberg/libiscsi.
- Verification of the source code with checkpatch.pl, sparse, smatch and Coverity. See e.g. https://scan.coverity.com/projects/scst.

SCST - Defect Density Reported by Coverity

Component Name	Lines of Code	Defect Density
SCST core	47,420	0.00
SRP target driver	3,793	0.00
iSCSI target driver (user space)	8,452	0.47
iSCSI target driver (kernel)	13,319	0.15
FCoE target driver (fcst)	1,609	0.00
qla2x00t	42,160	0.28
scst_local	1,681	0.00

SCST Documentation

- For users:
  - SCST README (http://sourceforge.net/p/scst/svn/HEAD/tree/trunk/README).
  - iSCSI README (http://sourceforge.net/p/scst/svn/HEAD/tree/trunk/iscsi-scst/README).
  - qla2x00t README (http://sourceforge.net/p/scst/svn/HEAD/tree/trunk/qla2x00t/doc/qla2x00t-howto.html).
  - ib_srpt README (http://sourceforge.net/p/scst/svn/HEAD/tree/trunk/srpt/README).
- For developers:
  - The SCSI SAM, SPC, and SBC standard documents (http://www.t10.org/).
  - The SCST Technical Documentation (http://scst.sourceforge.net/scst_pg.pdf).

SCST / LIO Unification Initiative

- Purpose of SCST and LIO is to realize SCSI target functionality for Linux.
- Structure of SCST and LIO is similar:
  - Core that processes SCSI commands.
  - Device handlers that implement a SCSI device type.
  - Target drivers that implement a SCSI transport protocol.
- Much code is shared between the SCST and LIO versions of the QLogic FC, the IB/SRP and the FcoE target drivers.
- However, the internal API's between both projects are different.
- The goal of this initiative is to reduce code duplication and to enlarge the user base.

See Also

- SCST home page (http://scst.sourceforge.net/).
- Openfiler home page (https://www.openfiler.com/).
- Marc Smith, *Implementing Enterprise Disk Arrays Using Open Source Software*, Merit Member Conference, 2012 (http://www.merit.edu/mmc/pdf/2012_Smith.pdf).
- Bart Van Assche, *[LSF/MM TOPIC] Unifying the LIO and SCST target drivers*, target-devel mailing list, January 2015, (http://thread.gmane.org/gmane.linux.scsi.target.devel/7779).

Any questions or comments ?

SanDisk®

Legal Notice