Linux Storage Area Network Topics & Linux Kernel Developers Summit Report

FUJITA Tomonori

NTT Cyber Solutions Laboratories

Storage Area Network topics

SCSI Client Server Model

- Initiator device (client)
  - Sending SCSI requests
- Target device (Server)
  - Performing SCSI requests and sending responses
- Transport
  - Connecting initiators with targets

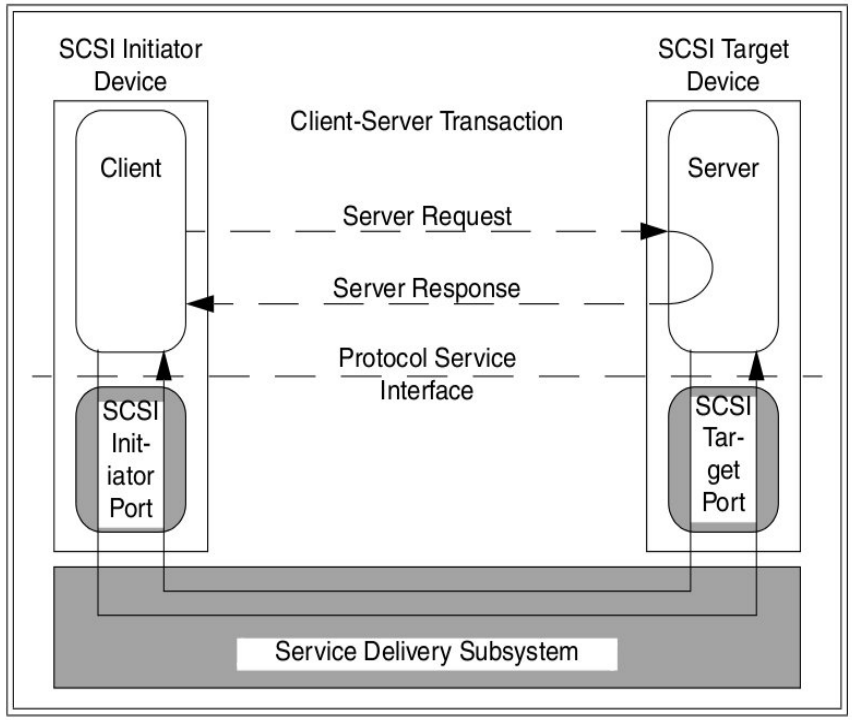
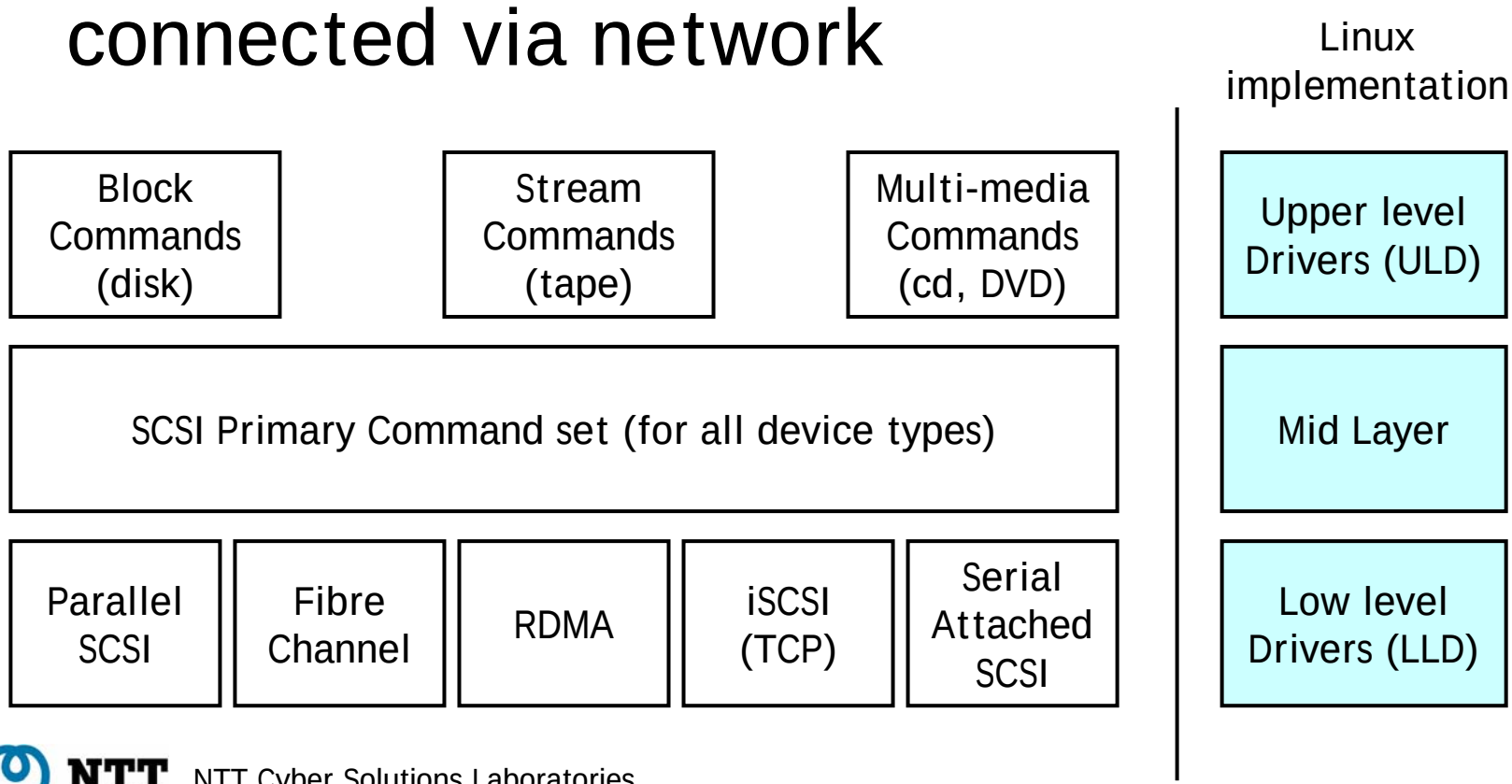


Figure 5 — Client-Server model

From t10.org

What is Storage Area Network (SAN)?

- hosts and storage devices are connected via network



What is the point in using virtual machines with SAN?

- Shared external storage with VMM
  - Enhancing manageability and availability
- Booting a host from a LUN on a SAN
  - VMM runs on a diskless box (maybe blade)
- Virtual machine cluster with SAN and multiple hosts
  - failover

SAN storage systems
(target devices)
are
expensive!

Can we convert a Linux box to a SAN storage system?

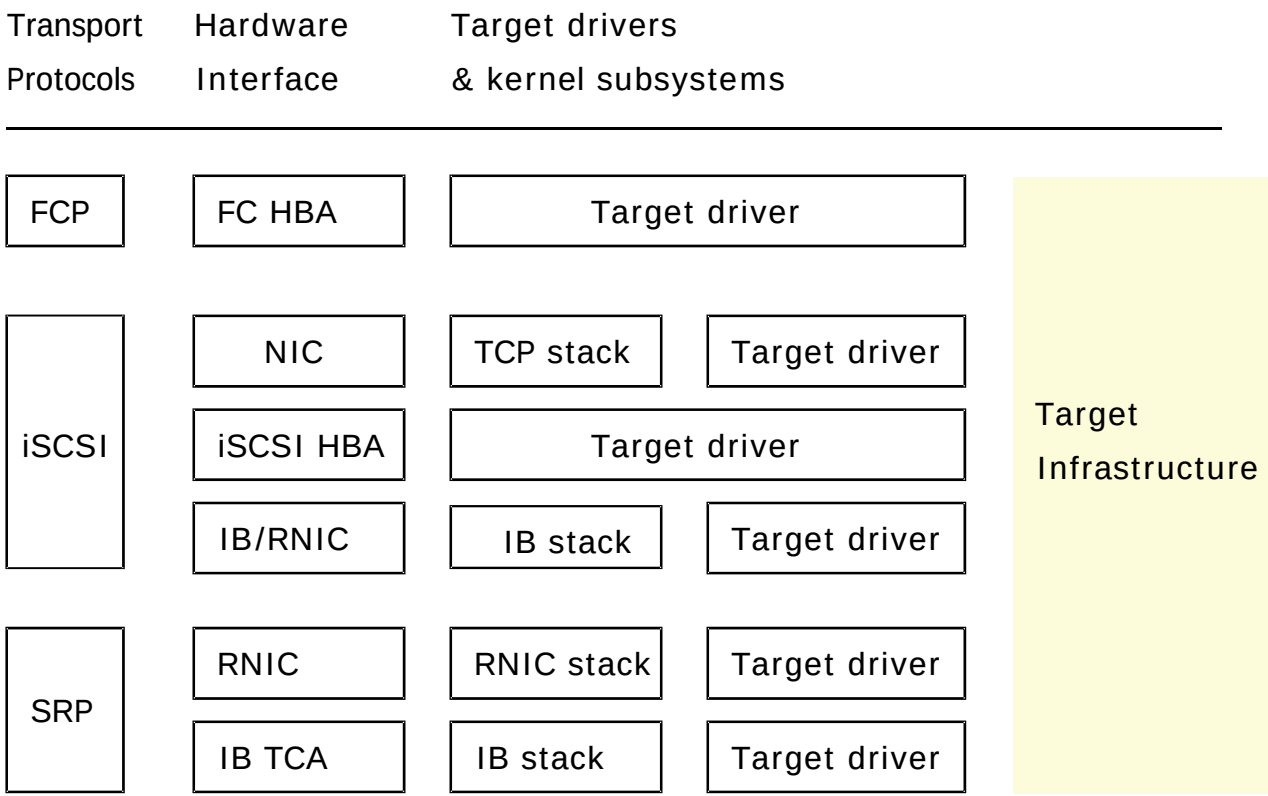
Target mode support

- Building a SCSI target device by using a Linux box to provide device services
  - Interpreting SCSI requests and sending back the results to initiators
- Why do we want it?
  - Poor man's SAN target devices
  - For debugging initiator drivers

How to design target mode?

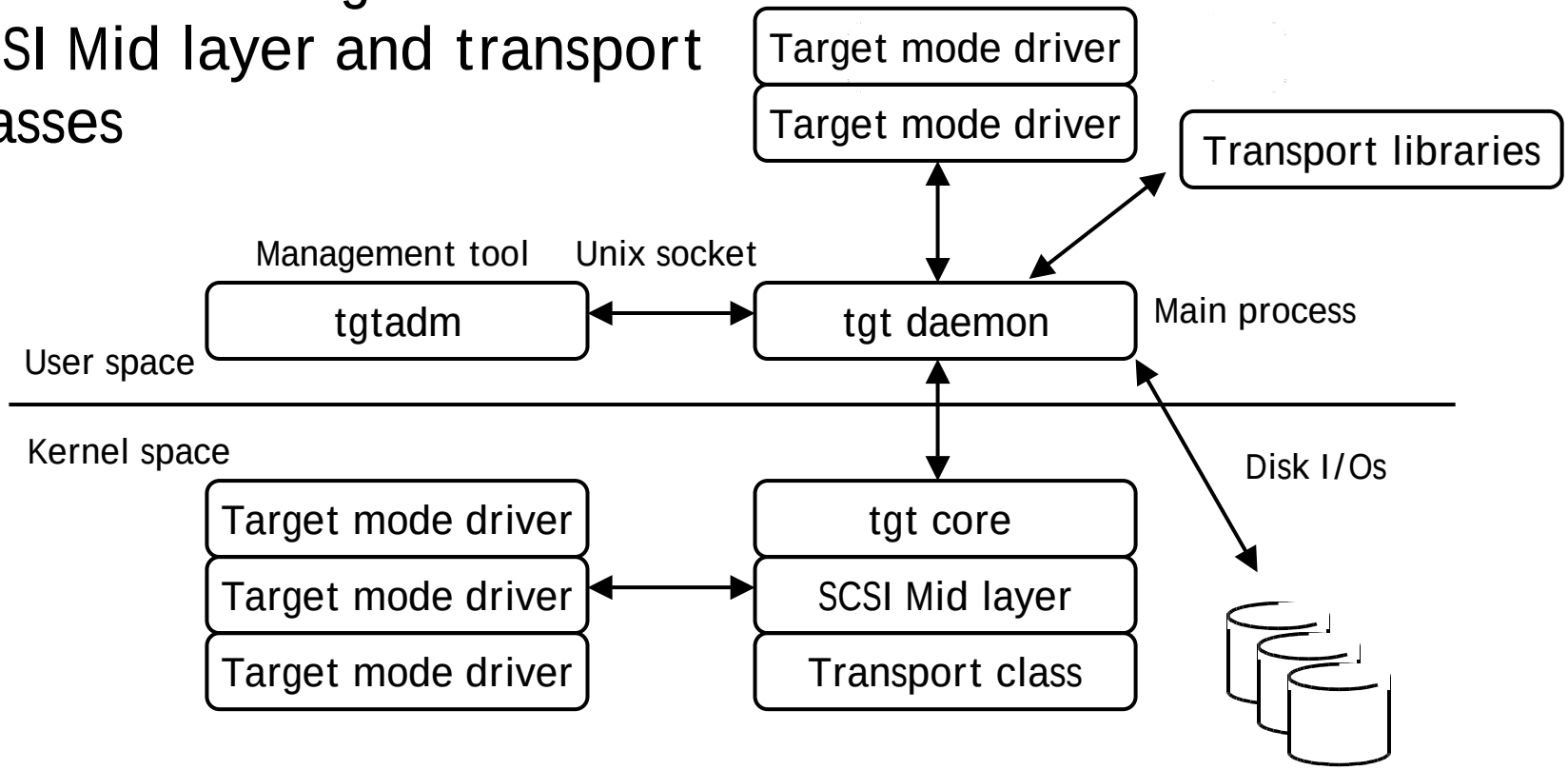
- Monolithic target mode design is wrong
  - e.g. iSCSI target software, FC target software
- Stop reinventing the wheel
  - Executing SCSI commands
  - SCSI state machine
  - logical unit management

Target infrastructure and target mode drivers



SCSI target infrastructure (aka tgt) design

tgt core is integrated with SCSI Mid layer and transport classes



Device type code

- Tgt can
  - provide a file as disk
  - provide an iso image file as cdrom
- Tgt will
  - provide a file as tape (Virtual Tape library)
  - provide a file as object storage

Supported tgt drivers

- iSCSI software driver (user space)
  - In RHEL 5.1 beta, openSUSE and debian (sid)
  - Trying to beat iSCSI enterprise target (aka IET)
- iSCSI Extension for RDMA driver (user space)
  - The code exists, but I've not merged it yet
- Qla2xxx FC driver
  - Under Qlogic review
- IBM System p Virtual target driver
  - SRP target devices (one VM provides the rest device service, like Xen blkback dirver)

bsg (block layer scsi generic driver or sg v4)

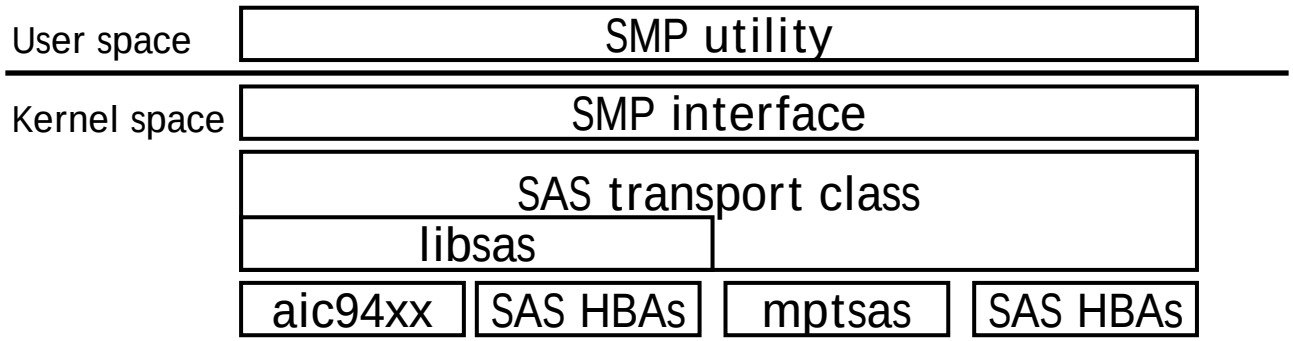
- Sg v3 (drivers/scsi/sg.c) shortcomings
  - Can handle only SCSI request/response for only scsi devices
    - We need transport level commands for management
    - We need SCSI task management commands
  - Can't commands with bidirectional data
  - Can't handle commands with task tag or attribute

bsg (cont.)

- Generic request/response protocol interface
  - You can create a bsg device for any object in kernel (bsg can be bind with request_queue)
  - You can handle any protocol (define your own request and response handlers)
  - User space can send requests and receive response via read/write system calls

bsg: SAN management

- SAS Management Protocol (SMP)
  - Talking with SAS devices and expanders via HBA
  - bsg devices are bind with every SAS object
- bsg provides an unified interface
  - Stop vendors from inventing various interfaces
  - aic94xx and mptsas are supported



SMP example

- REPORT_MANUFACTURE
  - Ask who you are

```
# smp_test /sys/class/bsg/expander-2:0
        SAS-1.1 format: 0
        vendor identification: LSILOGIC
        product identification: SASx12 A.0
        product revision level:
```

bsg: bidirectional data transfer (bidi)

- Object Storage Devices (aka OSD)
  - variable-length objects and object ids
  - bidi support is a must
- Bidi support status
  - Block layer is in upstream (needed for SMP)
  - Scsi mid layer is not yet (hopefully 2.6.25)

scsi data accessors

- Insulating LLDs from data transfer information
  - We plan to make lots of changes to scsi_cmnd structure support sg chaining and bidirectional data transfer
  - LLDs directly accessed to the values in scsi_cmnd
  - We rewrite LLDs to access scsi_cmnd via new accessors

scsi data accessors (cont.) Too simple sg setup examples

How a LLD tell addresses for I/Os for the HBA

Old way

```
struct scsi_cmnd *sc
struct scatterlist *sg =
         sc->request_buffer;

for(i = 0; i < nseg; i++) {
   paddr = sg_dma_address(sg[i]);
   …
```

New way

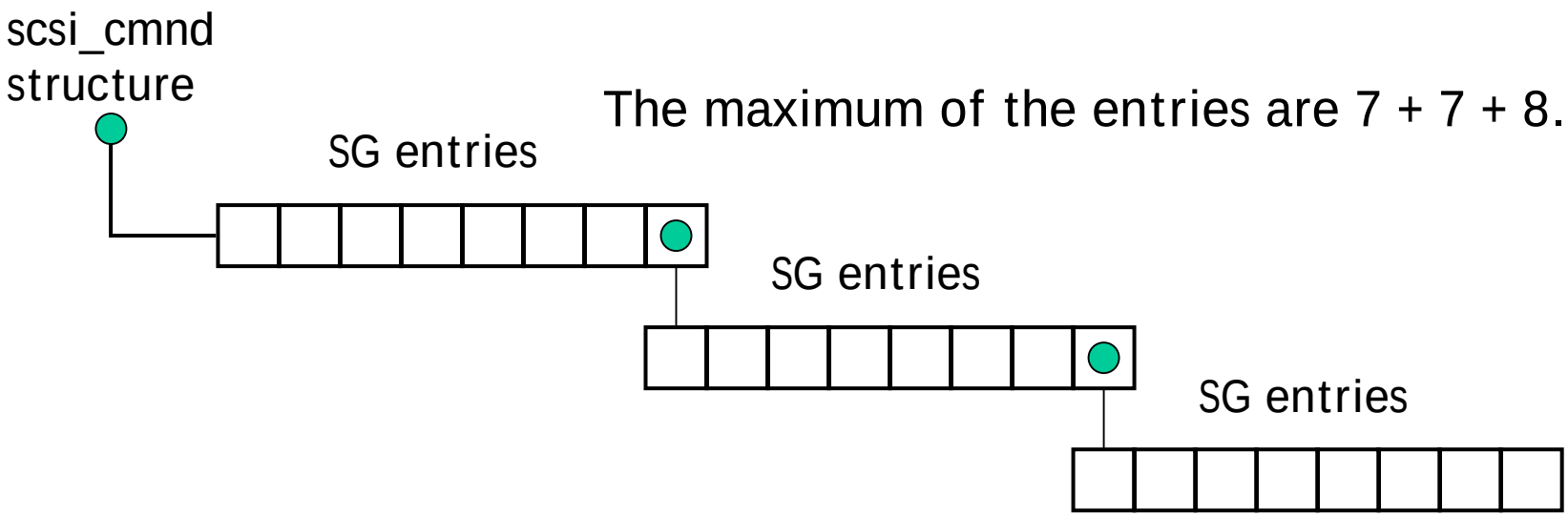
```
stcuct scsi_cmnd *sc
struct scatterlist *sg;

scsi_for_each_sg(sc, sg, nseg,i) {
   physaddr = sg_dma_address(sg);
   …
```

scatter gather chaining

- SCSI-ml couldn't handle Large data transfer
  - scsi-ml pools 8, 16, 32, 128 sg entries (the sg size is 32 bytes on x86_64)
  - People complains about scsi memory consumption so we can't have large sg entries
  - scsi_cmnd struct has a point to sg entries

scatter gather chaining (cont.)

- sg chaining
  - The last entry in a sg list points to the next sg list
  - sg entries aren't continuous any more!



How didi scsi data accessors help sg chaining?

- Before sg chaining

```
#define scsi_for_each_sg(sc, sg, nseg, i)
for(i = 0, sg = scsi_sglist(sc); i < nseg, i++, sg++)
```

sg entries must be continuous

- We changed it after sg chaining

```
#define scsi_for_each_sg(sc, sg, nseg, i)
for(i = 0, sg = scsi_sglist(sc); i < nseg, i++, sg =
   sg_next(sg))
```

sg_next macro takes care of discontinuous sg entries

LLDs can support sg chaining magically without modifications

Future SAN technologies

Fibre Channel over Ethernet (FCoE)

- Putting a FC frame into an Ethernet frame
  - iSCSI? Why do we need TCP overhead in data center?
  - TOE/RDMA? Who needs complicated hardware?
- Intel plans to release software FCoE initiator driver in December
  - Should be easy to add FCoE software target driver to tgt
- FCoE might be a good solution for VMs

Object Storage Device (OSD)

- New SCSI device type
  - No sector, initiators can access to variable-length objects with 64-bit id
- Who needs OSD?
  - pNFS (parallel NFS)
  - Parallel file systems (lustre, IBM storage tank, etc) prefer object-based storage
- Local file systems for OSD?

Linux Kernel Developers Summit

Agenda day1

- Distributor kernel maintainer panel
- Mini-summit reports
- The greater kernel ecosystem
- Documentation
- Kernel quality
- Hardware
- I386/x86_64 merger

Agenda day2

- Customer panel
- Real-time and scheduler
- Scalability
- Memory management
- Containers
- Developer relations
- Some of Andrew's process questions
- Future kernel summits