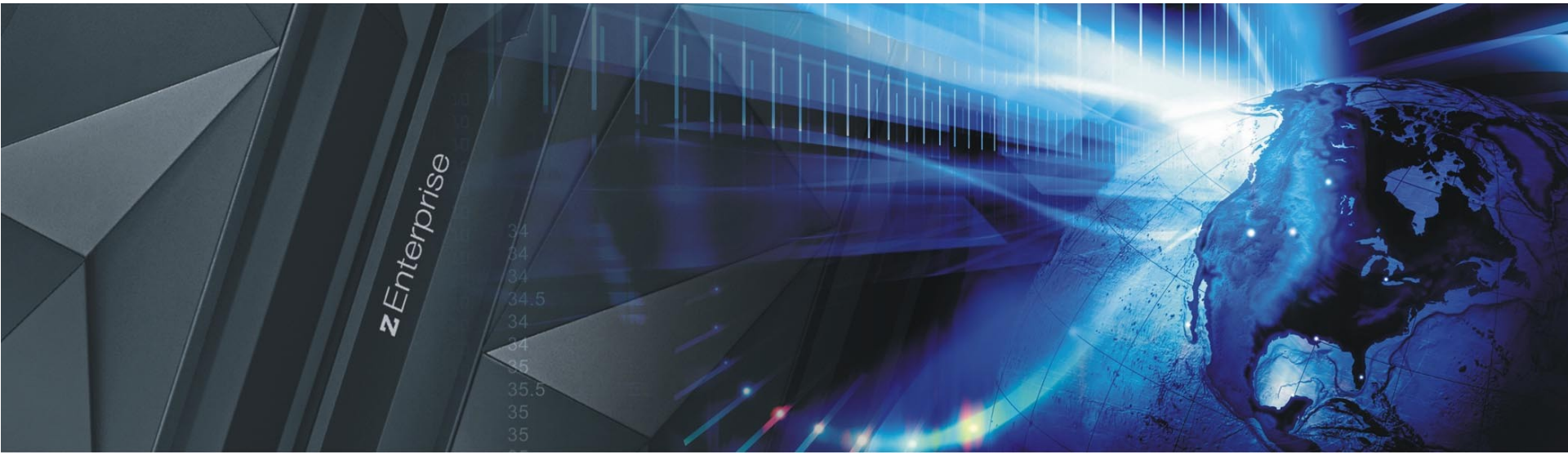
Linux on System z - Disk I/O Performance – Part 1

Mustafa Mešanović, IBM R&D Germany, System Performance Analyst



Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

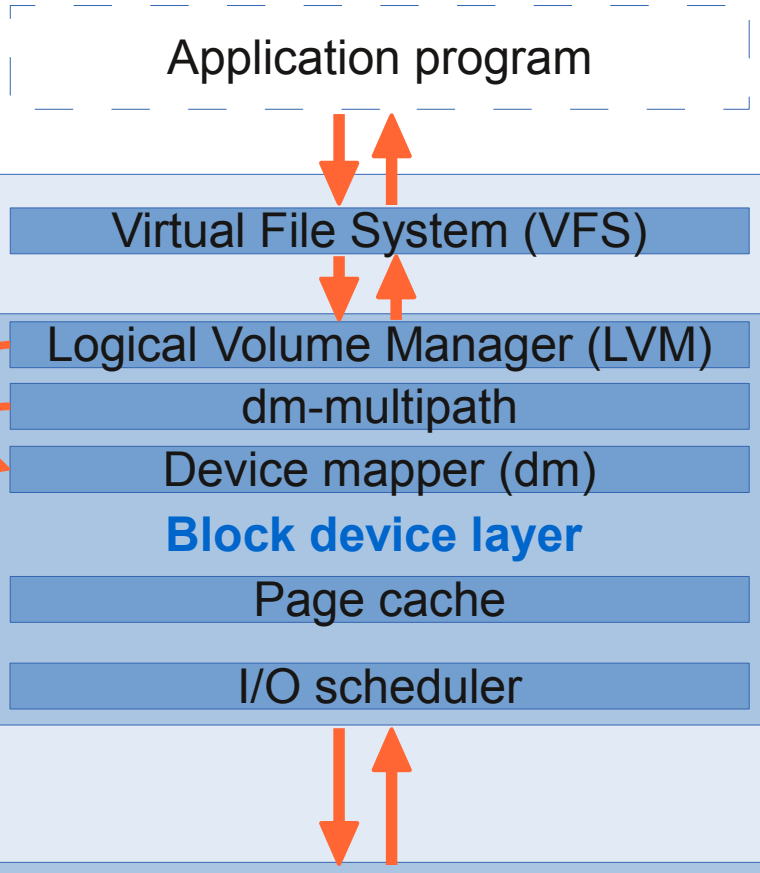
Oracle and Java are registered trademarks of Oracle and/or its affiliates in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies.

Agenda

- Terms and characteristics in the disk I/O area
  - I/O scheduler, page cache, direct I/O...
- Storage server setup
  - common parts, storage pool striping...
- Disk I/O configurations for FICON/ECKD and FCP/SCSI
  - and its possible advantages and bottlenecks
- Summary / Conclusion

Linux Kernel Components involved in Disk I/O



Issuing reads and writes

VFS dispatches requests to different devices and translates to sector addressing

LVM defines the physical to logical device relation

Multipath sets the multipath policies

dm holds the generic mapping of all block devices and performs 1:n mapping for logical volumes and/or multipath

Page cache contains all file I/O data, direct I/O bypasses the page cache
I/O schedulers merge, order and queue requests, start device drivers via (un)plug device

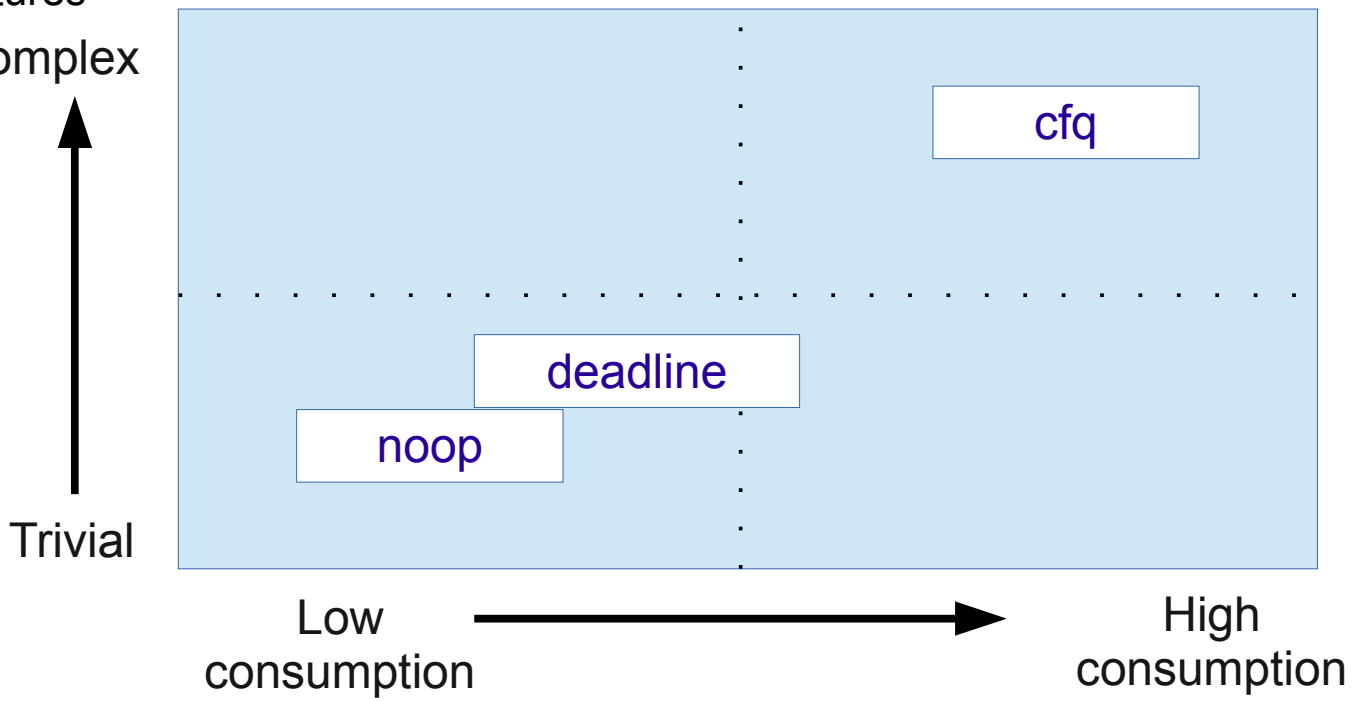
Data transfer handling

I/O Scheduler

- Three different I/O schedulers are available
  - noop scheduler
    - Only last request merging
  - deadline scheduler
    - Plus full search merge; avoids read request starvation
  - complete fair queuing (cfq) scheduler
    - Plus all users of a particular drive would be able to execute about the same number of I/O requests over a given time
- The default scheduler in current distributions is deadline
  - Do not change the default scheduler / default settings without a reason
    - It could reduce throughput or increase processor consumption

I/O Scheduler (cont.)

- deadline scheduler throughput and processor consumption moderate
- noop scheduler showed similar throughput like deadline scheduler
  - May save processor cycles in some constellations
- cfq had no advantage in all our measurements
  - Tends to high processor consumption
  - Offers many features



Page Cache

- Page cache helps Linux to economize I/O
  - Read requests can be made faster by adding a read ahead quantity, depending on the historical behavior of file system accesses by applications
  - Write requests are delayed and data in the page cache can have multiple updates before being written to disk.
  - Write requests in the page cache can be merged into larger I/O requests
- But page cache...
  - Requires Linux memory pages
  - Is not useful when cached data is not exploited
    - Data just only needed once
    - Application buffers data itself
  - In Linux does not know which data the application really needs next. It makes only a guess
- No alternatives if application cannot handle direct I/O

Consider to use...

- direct I/O:
  - bypasses the page cache
  - is a good choice in all cases where the application does not want Linux to economize I/O and/or where the application buffers larger amount of file contents
- async I/O:
  - prevents the application from being blocked in the I/O system call until the I/O completes
  - allows read merging by Linux in case of using page cache
  - can be combined with direct I/O
- temporary files:
  - should not reside on real disks, a ram disk or tmpfs allows fastest access to these files
  - they don't need to survive a crash, don't place them on a journaling file system
- file system:
  - use ext3 and select the appropriate journaling mode (journal, ordered, writeback)
  - turning off atime is only suitable if no application makes decisions on "last read" time, consider relatime instead

Direct I/O versus Page cache

- Direct I/O
  - Preferable if application caches itself
    - Application knows best which data is needed again
    - Application knows which data is most likely needed next

      Example database base management systems DBMS
  - Preferable if caching makes no sense
    - Data only needed once

      Backup and restore
- Page cache
  - Optimizes re-read / write but can be critical
    - Data written to the page cache but not to disk yet can get lost if data loss cannot easily be handled
  - If application cannot handle direct I/O
    - Typical example is a file server

Linux multipath (High Availability Feature)

- Connects a volume over multiple independent paths and / or ports
- Failover mode
  - Keeps the volume accessible in case of a single failure in the connecting hardware
    - should one path fail, the operating system will route I/O through one of the remaining paths, with no changes visible to the applications
    - Will not improve performance
- Multibus mode
  - Uses all paths in a priority group in a round-robin manner
    - switches the path after a configurable amount of I/Os

      See rr_min_io (or rr_min_io_rq) parameter in multipath.conf
    - Check the actual used value by the device mapper with “dmsetup table“
  - To overcome bandwidth limitations of a single path
  - Increases throughput for
    - SCSI volumes in all Linux distributions
    - ECKD volumes with static PAV devices in Linux older than SLES11 and RHEL6

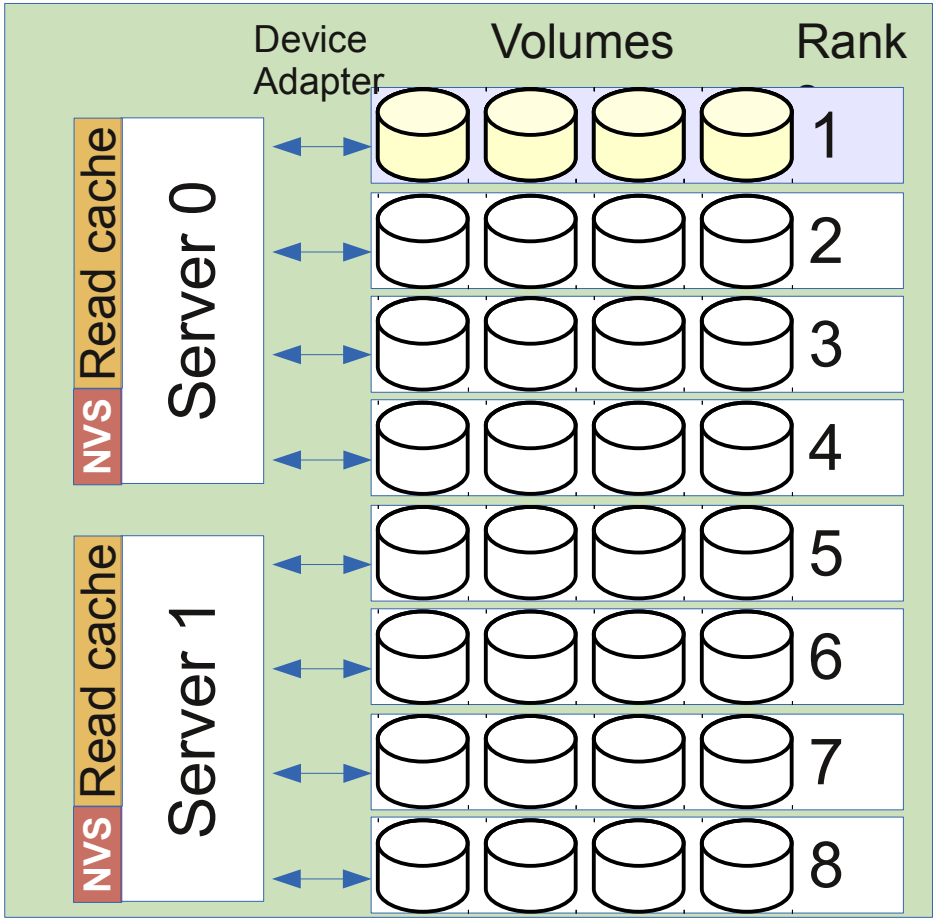
      But now HyperPAV with RHEL5.9 compatible (Jan 2013)
  - For load balancing
-

Logical Volumes

- Linear logical volumes allow an easy extension of the file system
- Striped logical volumes
  - Provide the capability to perform simultaneous I/O on different stripes
  - Allow load balancing
  - Are extendable
  - May lead to smaller I/O request sizes compared to linear logical volumes or normal volumes
- Don't use logical volumes for “/” or “/usr”
  - In case the logical volume gets corrupted your system is gone
- Logical volumes require more processor cycles than physical disks
  - Consumption increases with the number of physical disks used for the logical volume
- LVM / Logical Volume Manager is the tool to manage logical volumes

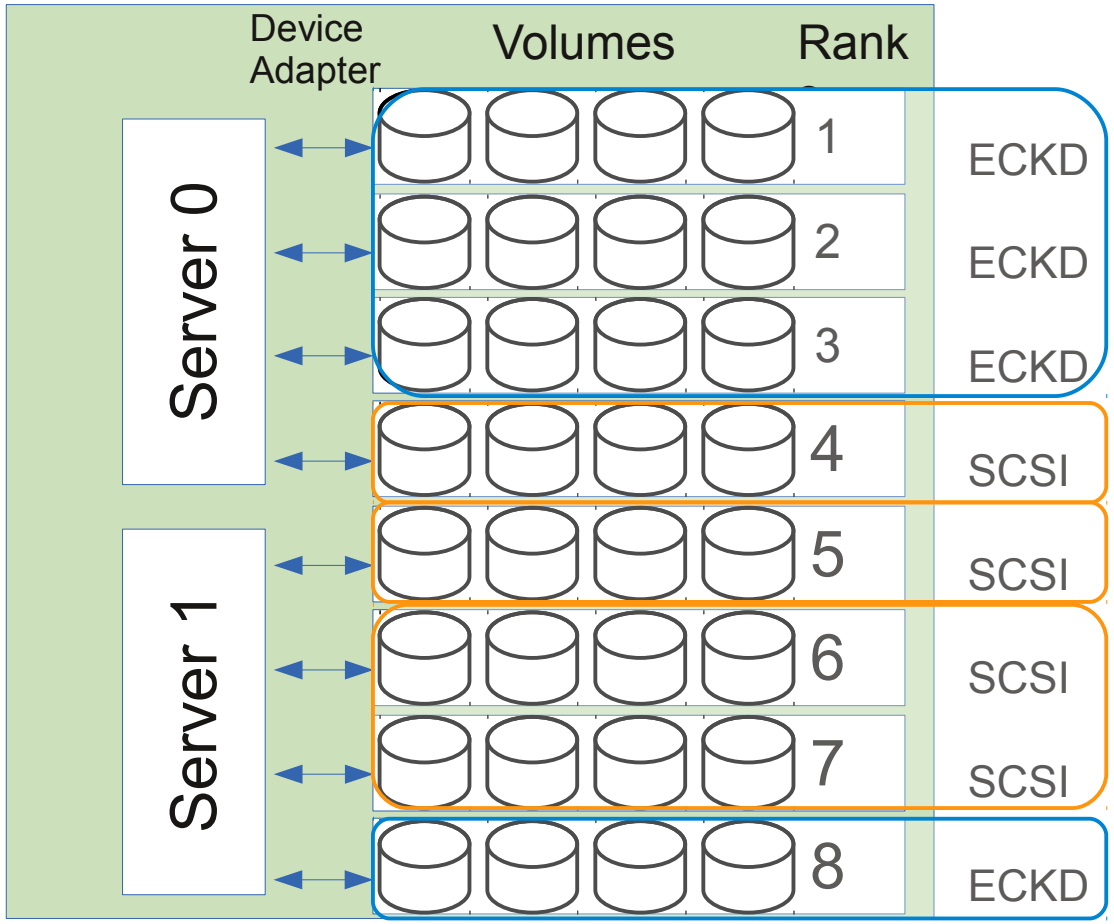
DS8000 Storage Server – Components

- Rank
  - Represents a raid array of disk drives
  - Belongs to one server for primary access
- A device adapter connects ranks to servers
- Each server controls half of the storage servers disk drives, read cache and non-volatile storage (NVS / write cache)



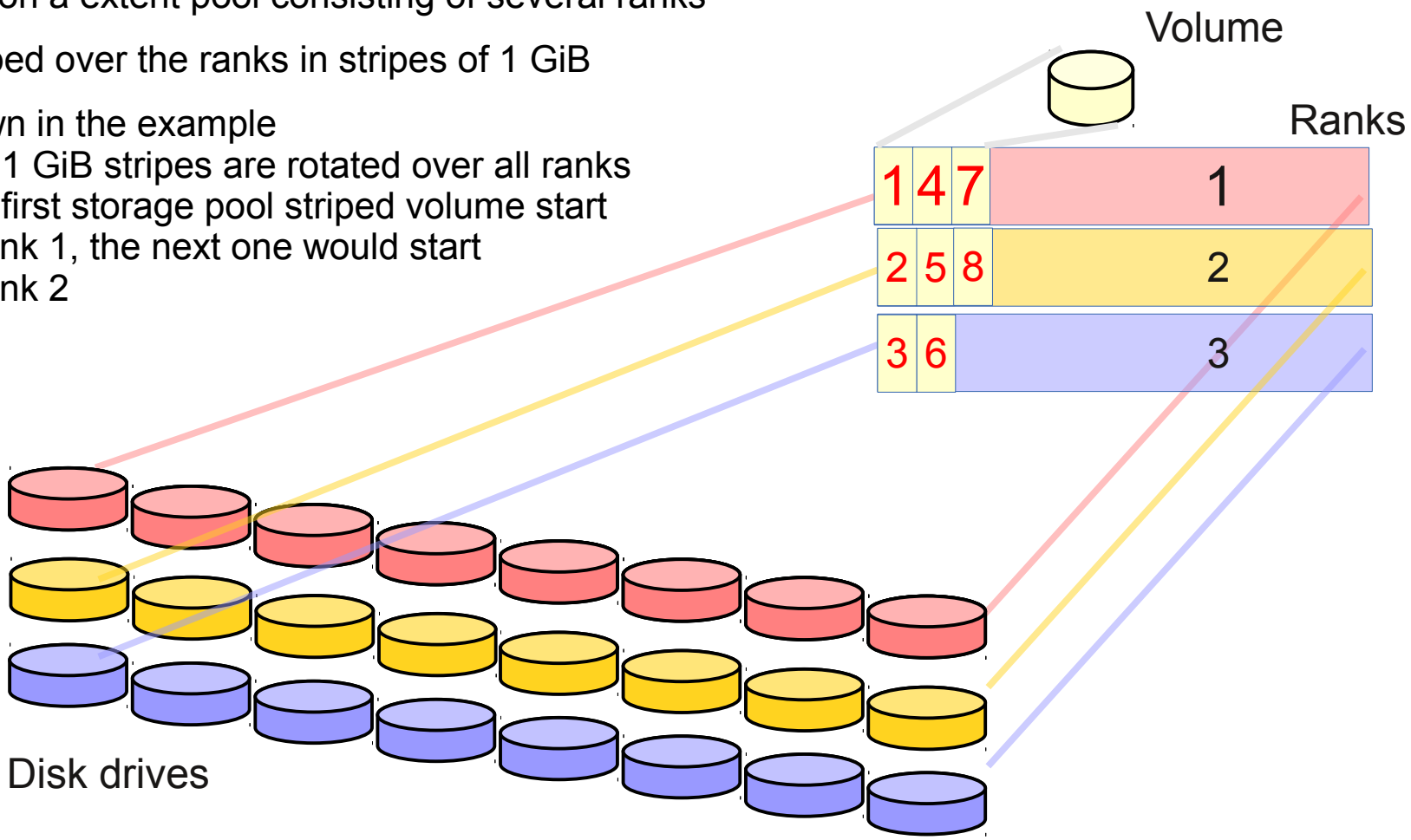
DS8000 Storage Server – Extent Pool

- Extent pools
  - consist of one or several ranks
  - can be configured for one type of volumes only: either ECKD DASDs or SCSI LUNs
  - are assigned to one server
- Some possible extent pool definitions for ECKD and SCSI are shown in the example



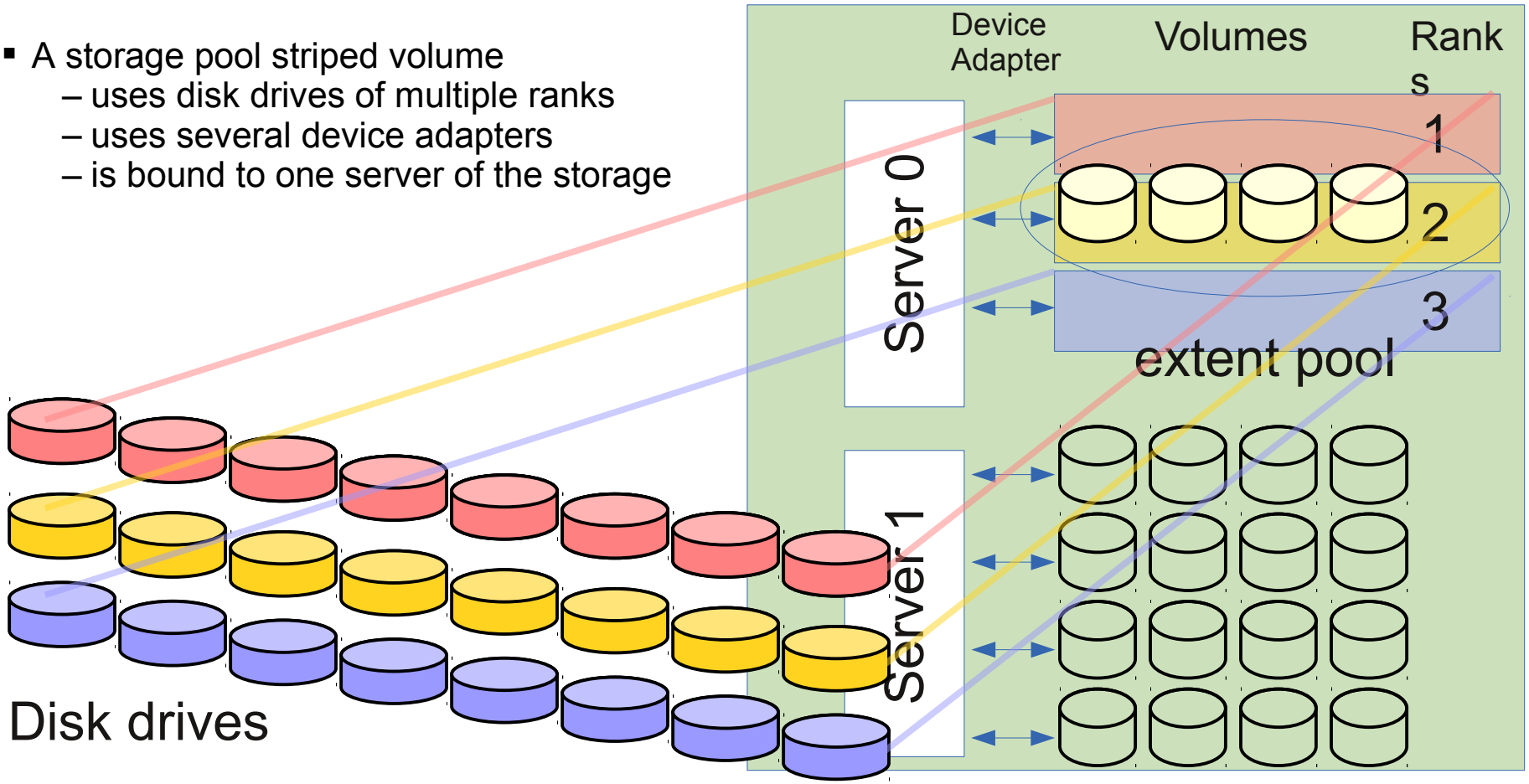
DS8000 Storage Pool Striped Volume

- A storage pool striped volume (rotate extents) is defined on a extent pool consisting of several ranks
- It is striped over the ranks in stripes of 1 GiB
- As shown in the example
  - The 1 GiB stripes are rotated over all ranks
  - The first storage pool striped volume start in rank 1, the next one would start in rank 2



DS8000 Storage Pool Striped Volume (cont.)

- A storage pool striped volume
  - uses disk drives of multiple ranks
  - uses several device adapters
  - is bound to one server of the storage

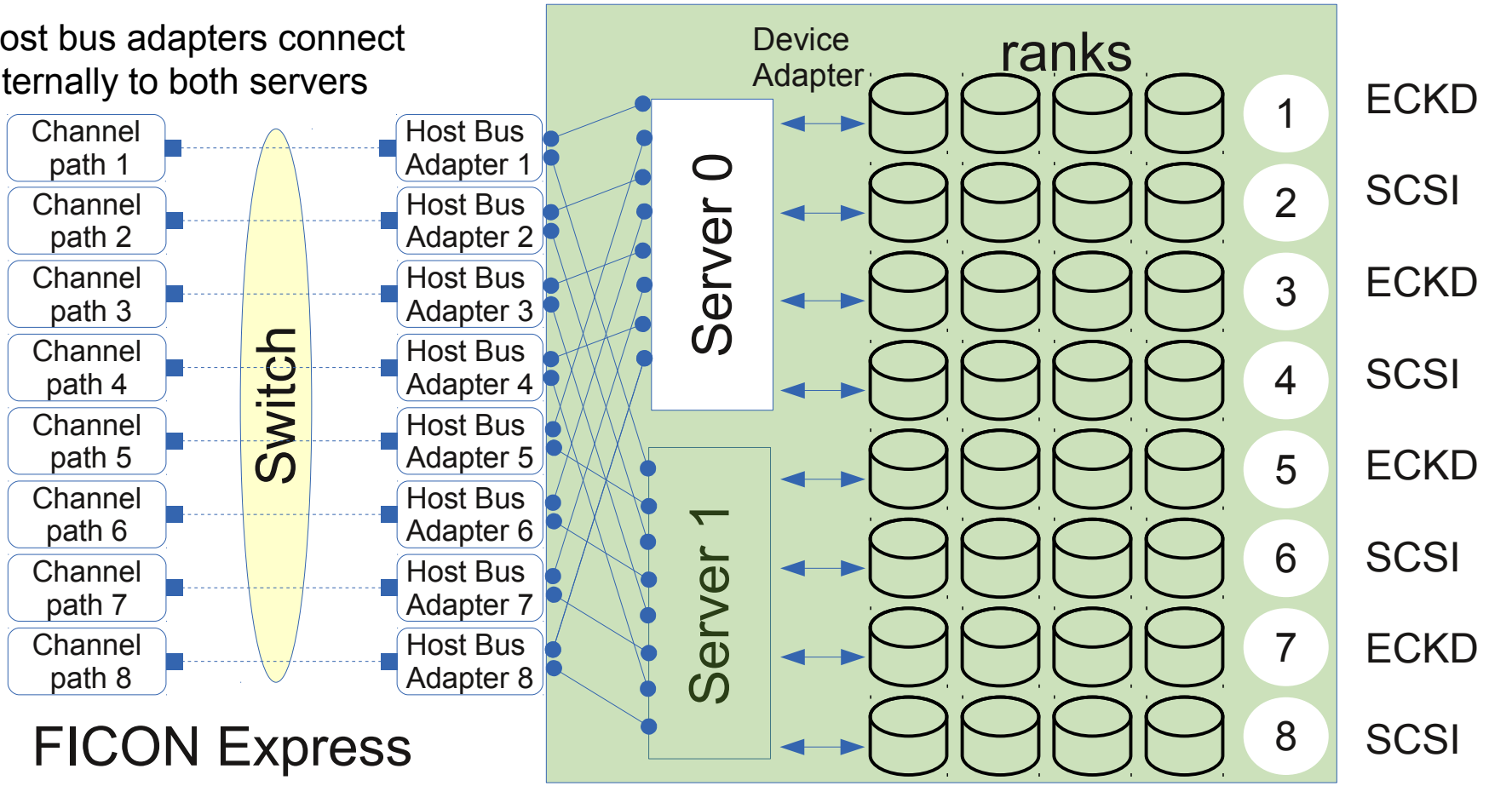


Striped Volumes Comparison - Best Practices for best Performance

	LVM striped logical volumes	DS8000 storage pool striped volumes
Striping is done by...	Linux (device-mapper)	Storage server
Which disks to choose...	plan carefully	don't care
Disks from one extent pool...	per rank / extent pool, alternating over servers	out of multiple ranks
Administrating disks is...	complex	simple
Extendable...	yes	no, but “gluing” disks together as linear LV can be a workaround
Stripe size...	variable, to suit your workload (64KiB, default)	1GiB

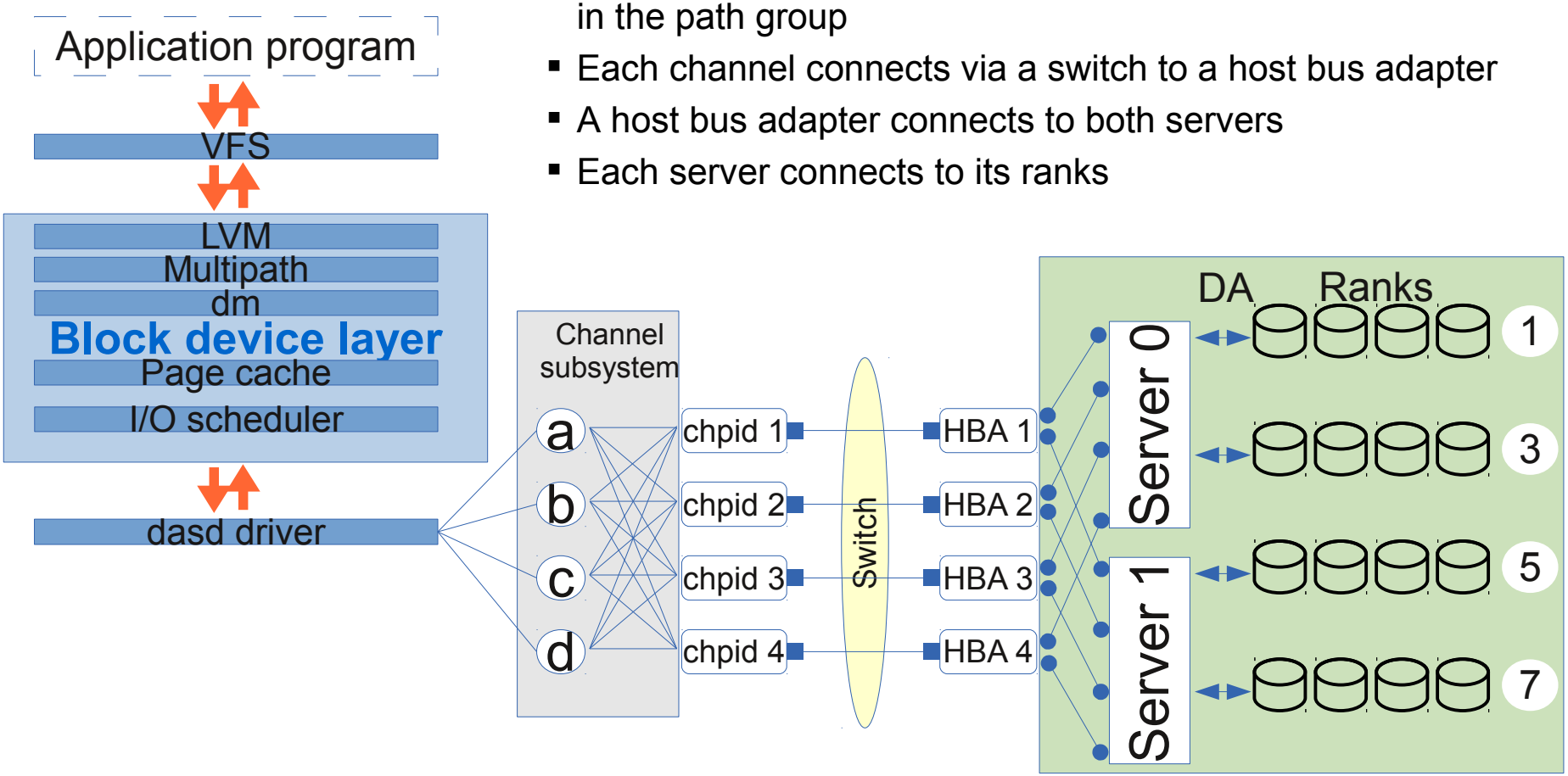
Disk I/O Attachment and the DS8000 Storage Subsystem

- Each disk is a configured volume in a extent pool (here extent pool = rank)
- Host bus adapters connect internally to both servers



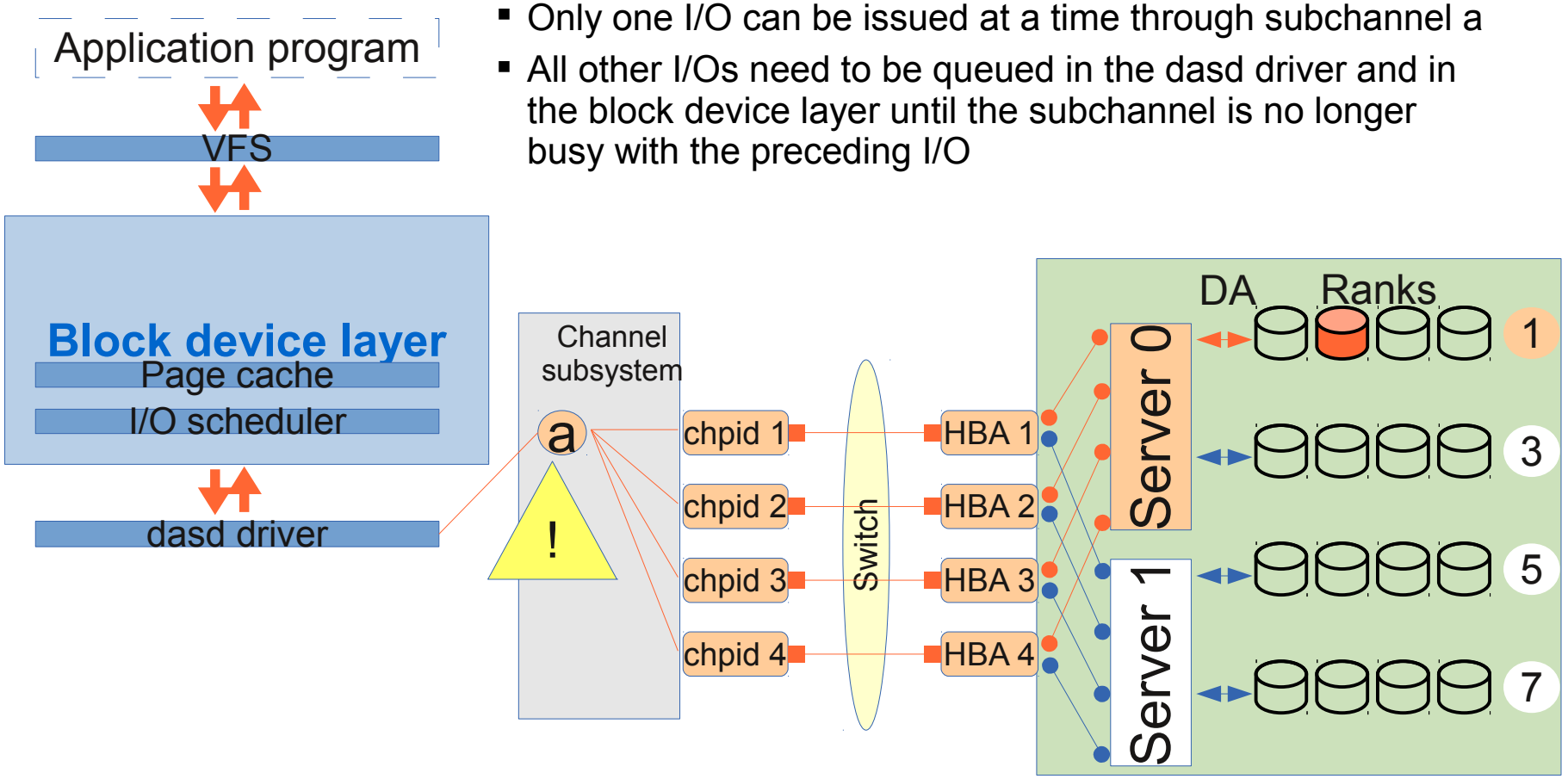
FICON/ECKD Layout

- The DASD driver starts the I/O on a subchannel
- Each subchannel connects to all channel paths in the path group
- Each channel connects via a switch to a host bus adapter
- A host bus adapter connects to both servers
- Each server connects to its ranks



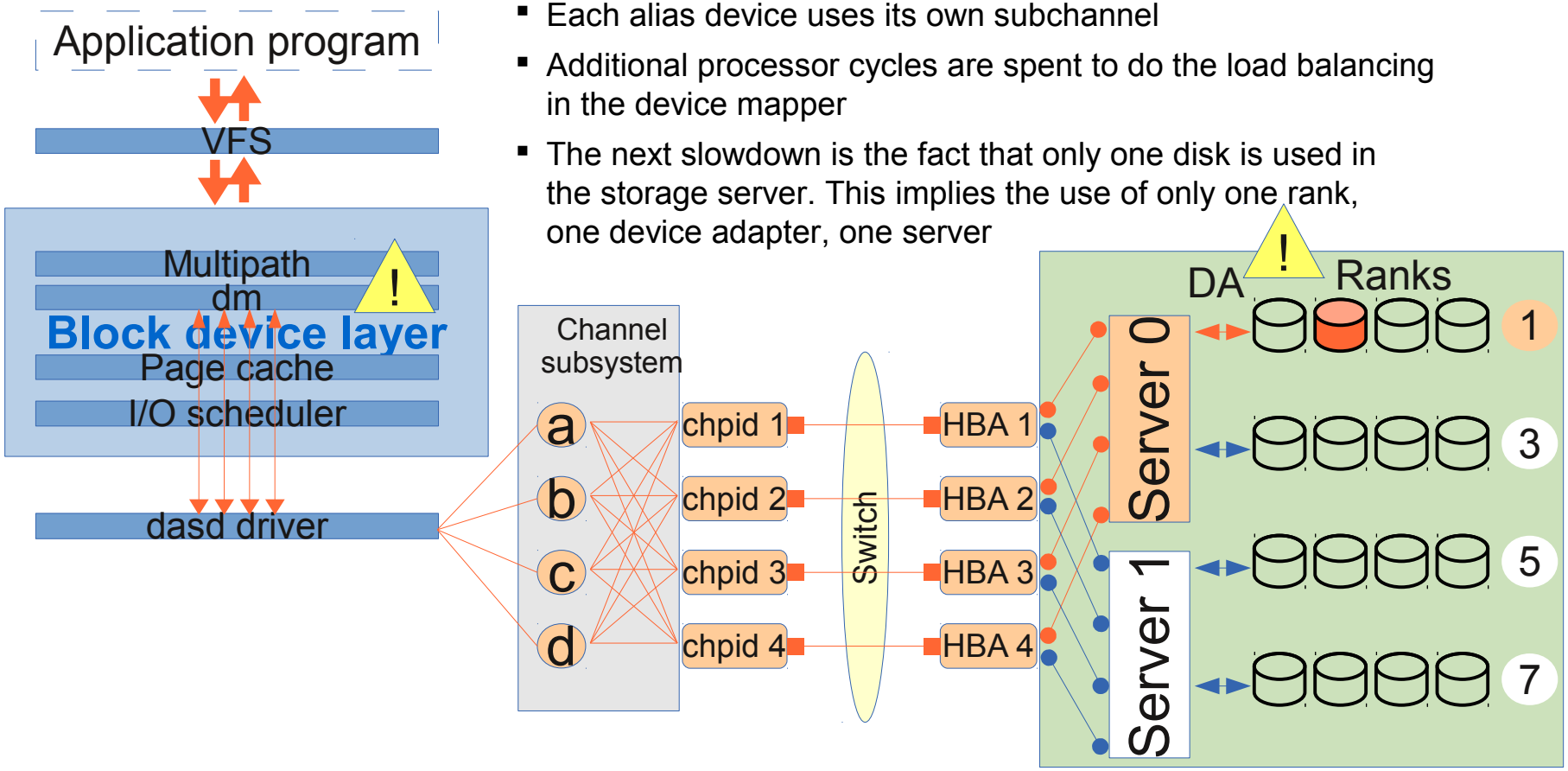
FICON/ECKD Single Disk I/O

- Assume that subchannel a corresponds to disk 2 in rank 1
- The full choice of host adapters can be used
- Only one I/O can be issued at a time through subchannel a
- All other I/Os need to be queued in the dasd driver and in the block device layer until the subchannel is no longer busy with the preceding I/O



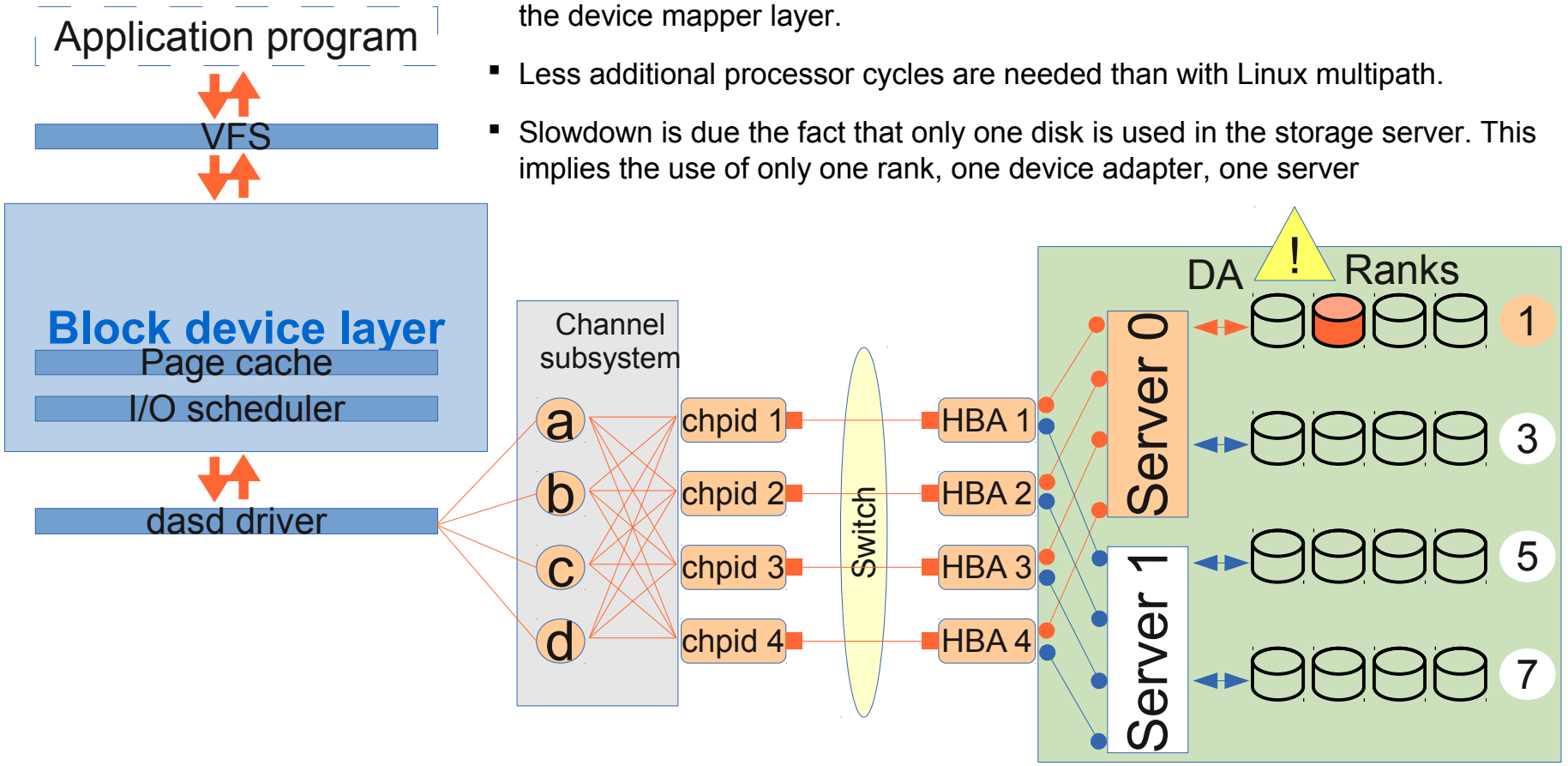
FICON/ECKD Single Disk I/O with PAV

- VFS sees one device
- The device mapper sees the real device and all alias devices
- Parallel Access Volumes solve the I/O queuing in front of the subchannel
- Each alias device uses its own subchannel
- Additional processor cycles are spent to do the load balancing in the device mapper
- The next slowdown is the fact that only one disk is used in the storage server. This implies the use of only one rank, one device adapter, one server



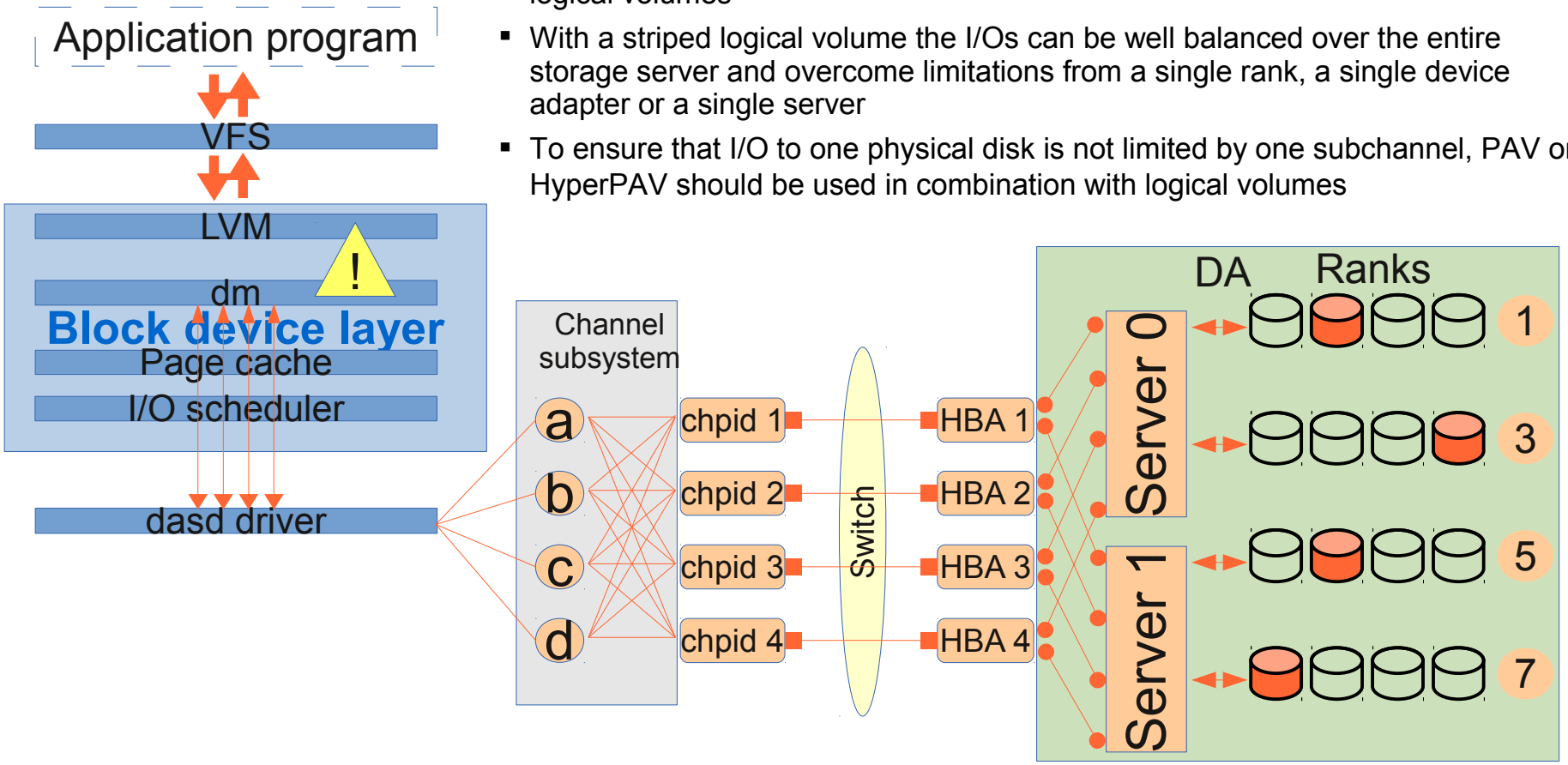
FICON/ECKD Single Disk I/O with HyperPAV (SLES11/RHEL6)

- VFS sees one device
- The dasd driver sees the real device and all alias devices
- Load balancing with HyperPAV and static PAV is done in the dasd driver. The aliases need only to be added to Linux. The load balancing works better than on the device mapper layer.
- Less additional processor cycles are needed than with Linux multipath.
- Slowdown is due the fact that only one disk is used in the storage server. This implies the use of only one rank, one device adapter, one server



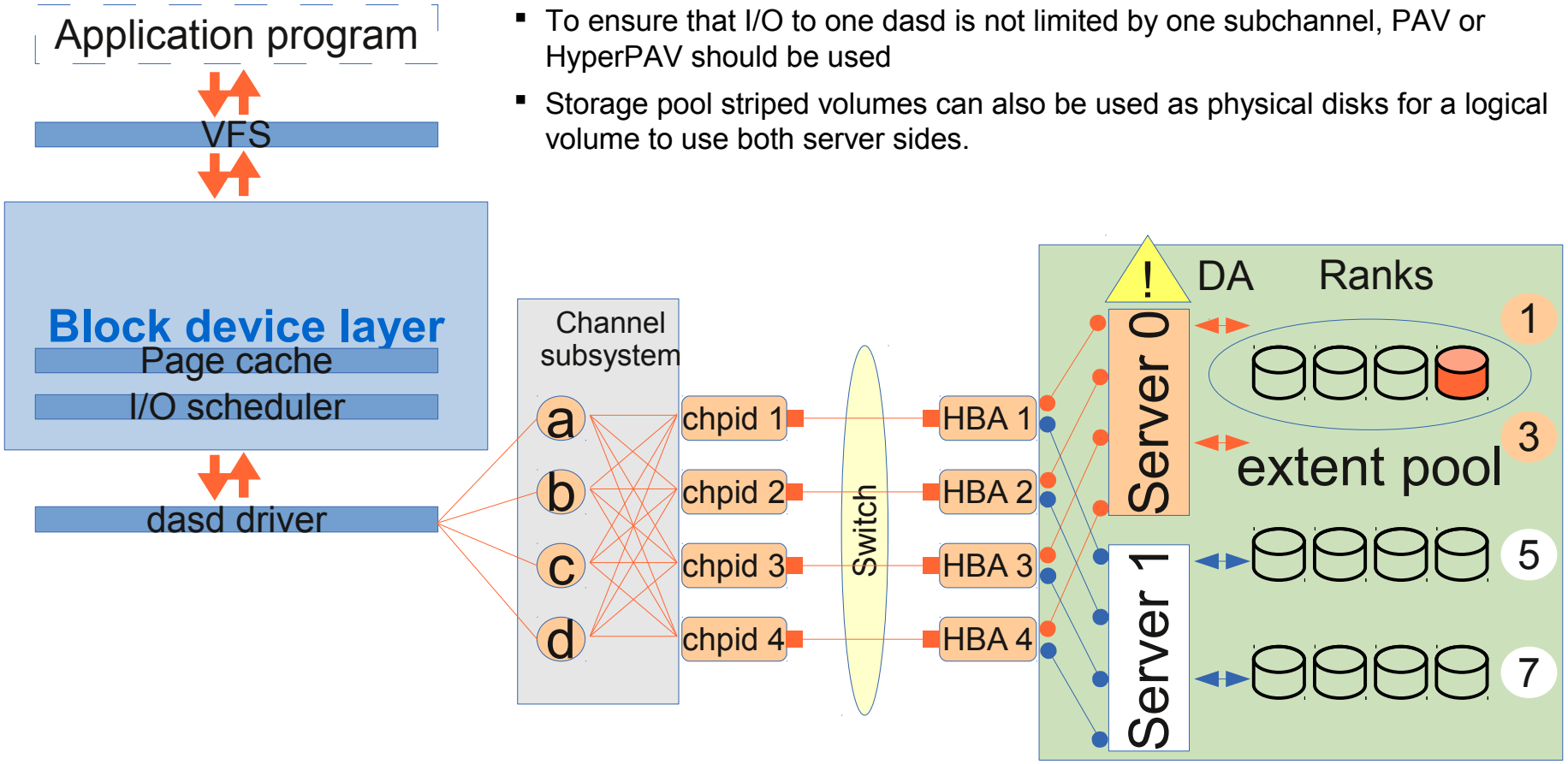
FICON/ECKD DASD I/O to a Linear / Striped Logical Volume

- VFS sees one device (logical volume)
- The device mapper sees the logical volume and the physical volumes
- Additional processor cycles are spent to map the I/Os to the physical volumes.
- Striped logical volumes require more additional processor cycles than linear logical volumes
- With a striped logical volume the I/Os can be well balanced over the entire storage server and overcome limitations from a single rank, a single device adapter or a single server
- To ensure that I/O to one physical disk is not limited by one subchannel, PAV or HyperPAV should be used in combination with logical volumes



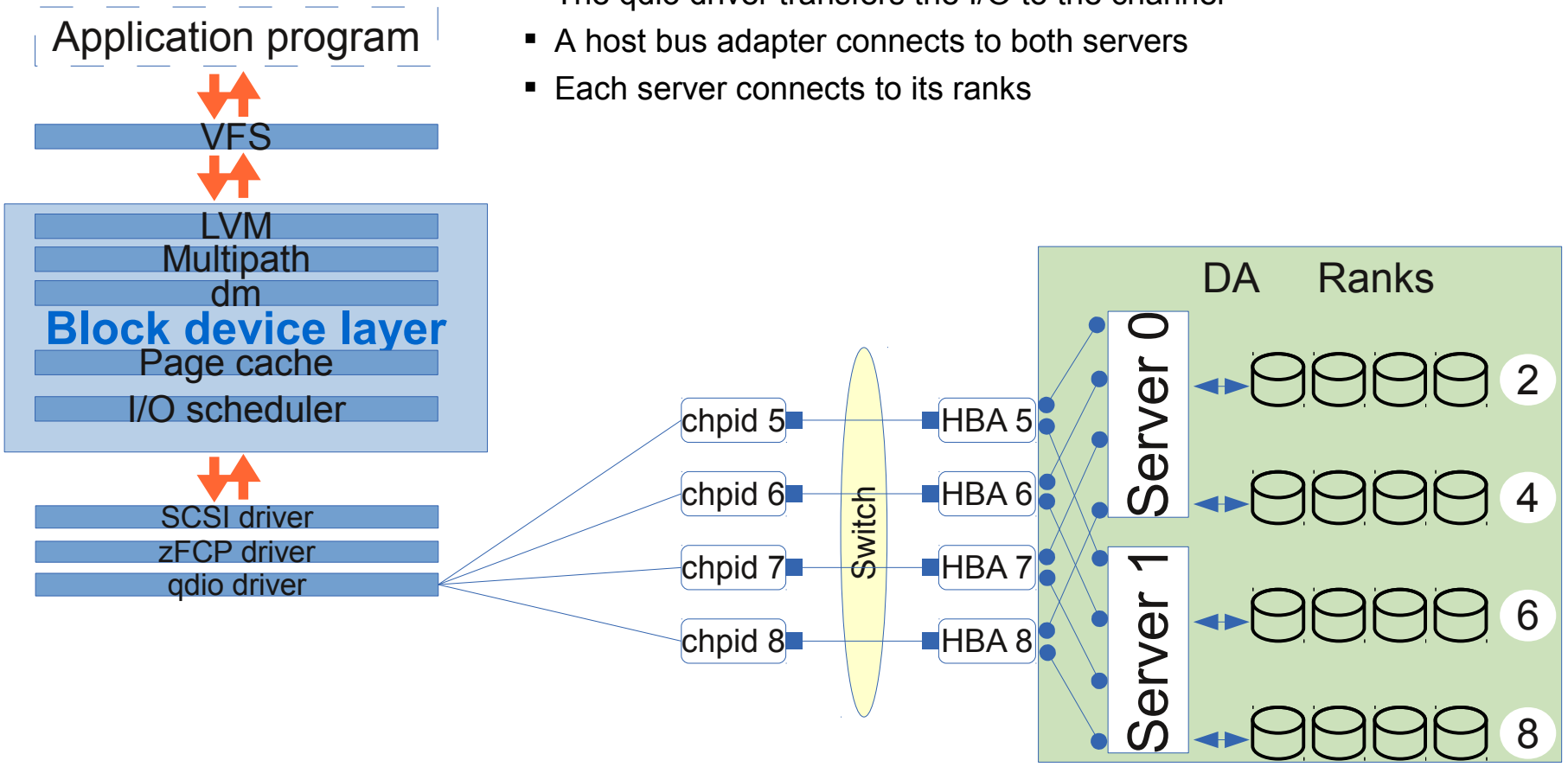
FICON/ECKD DASD I/O to a Storage Pool Striped Volume with HyperPAV

- VFS sees one device
- The dasd driver sees the real device and all alias devices
- The storage pool striped volume spans over several ranks of one server and overcomes the limitations of a single rank and / or a single device adapter
- To ensure that I/O to one dasd is not limited by one subchannel, PAV or HyperPAV should be used
- Storage pool striped volumes can also be used as physical disks for a logical volume to use both server sides.



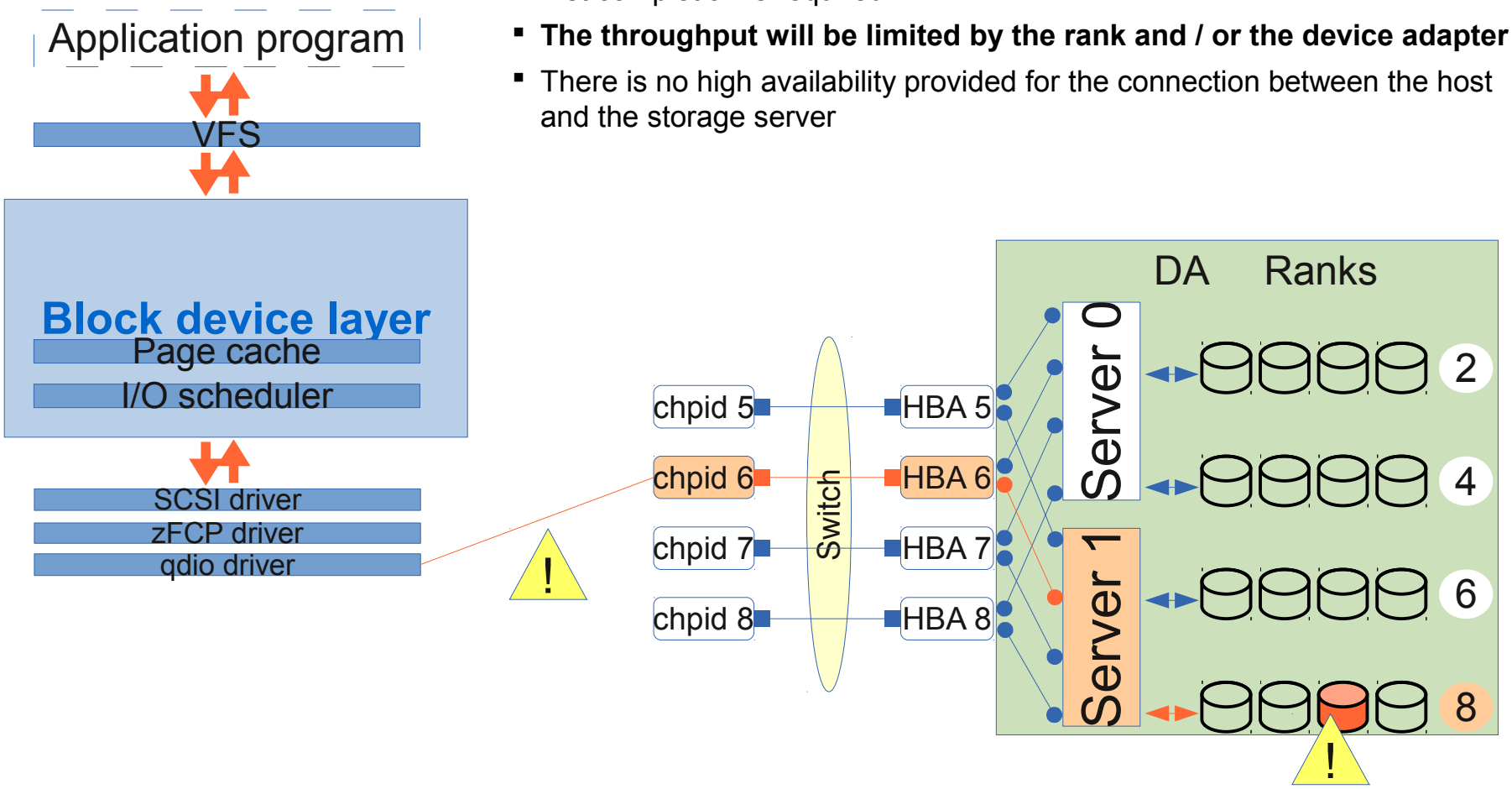
FCP/SCSI Layout

- The SCSI driver finalizes the I/O requests
- The zFCP driver adds the FCP protocol to the requests
- The qdio driver transfers the I/O to the channel
- A host bus adapter connects to both servers
- Each server connects to its ranks



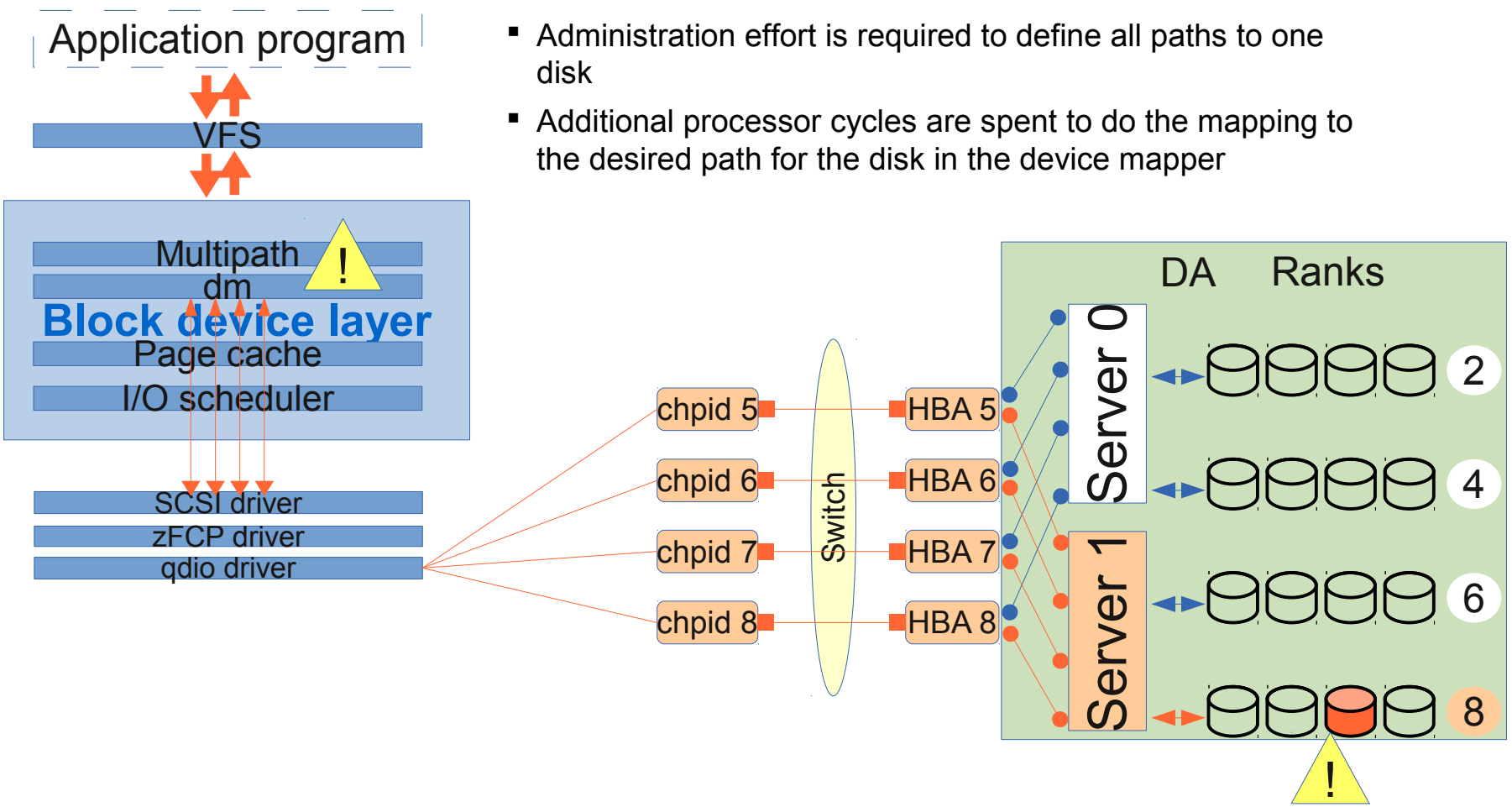
FCP/SCSI LUN I/O to a Single Disk

- Assume that disk 3 in rank 8 is reachable via channel 6 and host bus adapter 6
- Up to 32 (default value) I/O requests can be sent out to disk 3 before the first completion is required
- **The throughput will be limited by the rank and / or the device adapter**
- There is no high availability provided for the connection between the host and the storage server



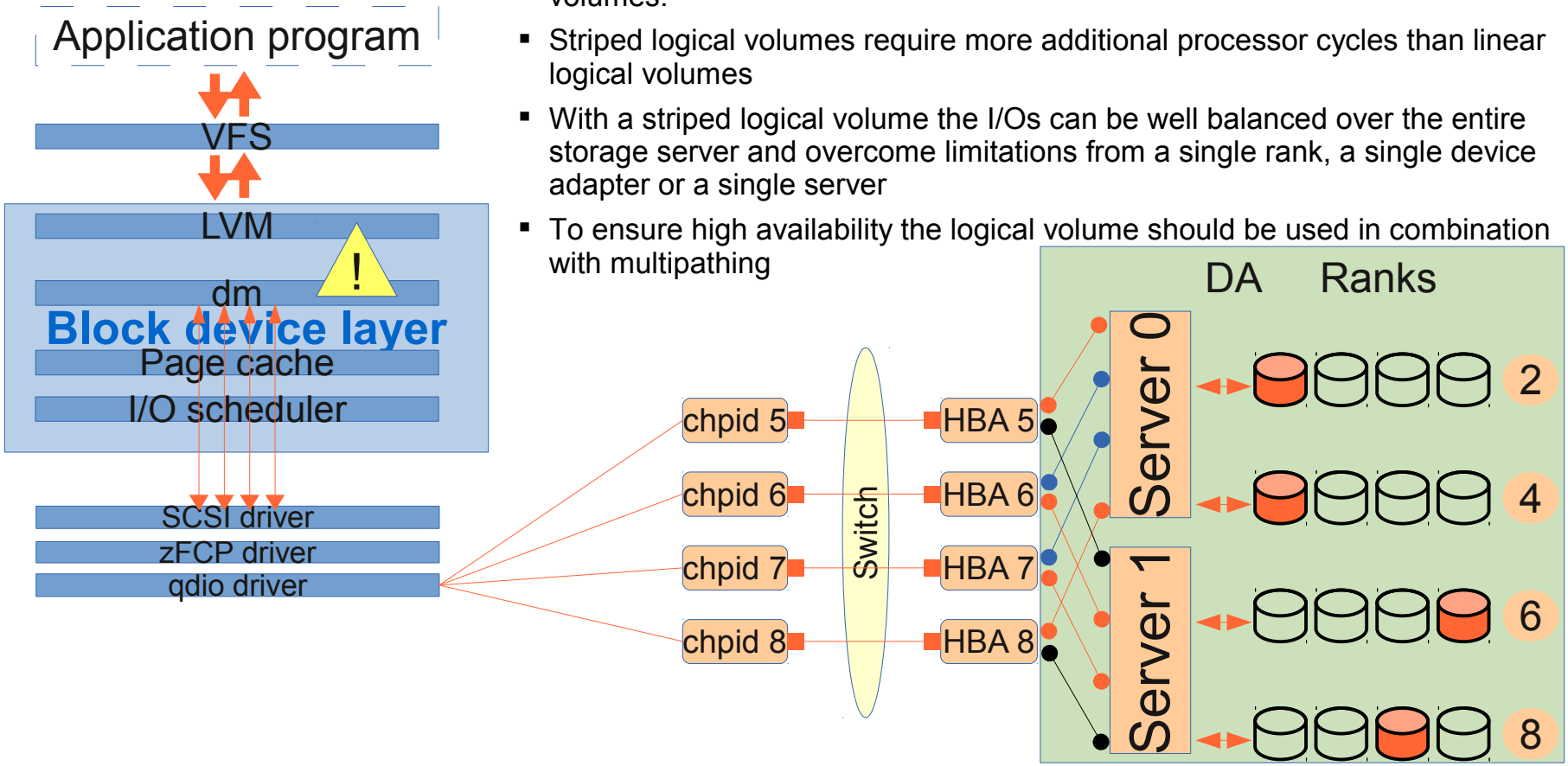
FCP/SCSI LUN I/O to a Single Disk with multipathing

- VFS sees one device
- The device mapper sees the multibus or failover alternatives to the same disk
- Administration effort is required to define all paths to one disk
- Additional processor cycles are spent to do the mapping to the desired path for the disk in the device mapper



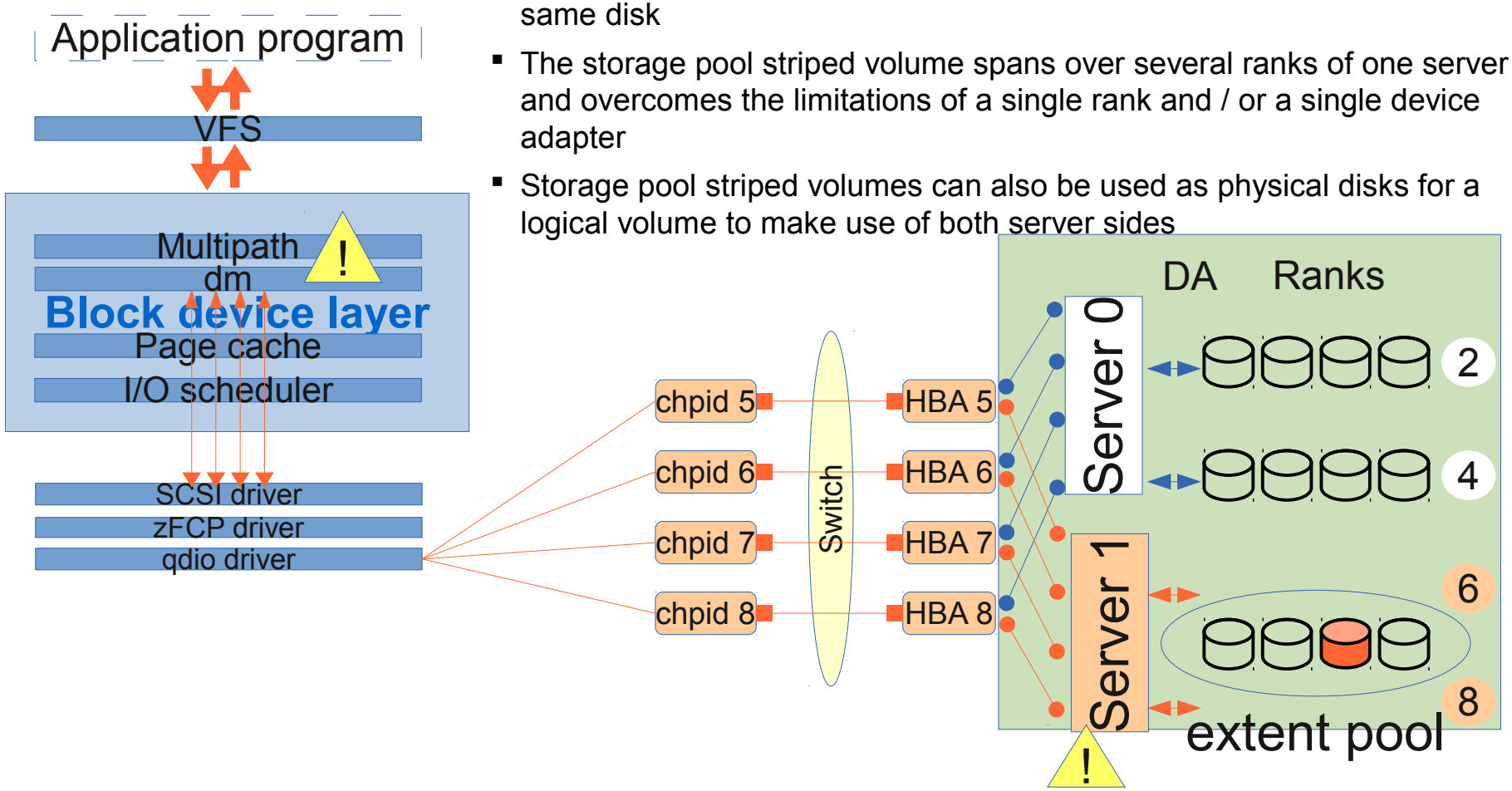
FCP/SCSI LUN I/O to a Linear / Striped Logical Volume

- VFS sees one device (logical volume)
- The device mapper sees the logical volume and the physical volumes
- Additional processor cycles are spent to map the I/Os to the physical volumes.
- Striped logical volumes require more additional processor cycles than linear logical volumes
- With a striped logical volume the I/Os can be well balanced over the entire storage server and overcome limitations from a single rank, a single device adapter or a single server
- To ensure high availability the logical volume should be used in combination with multipathing



FCP/SCSI LUN I/O to a Storage Pool Striped Volume with multipathing

- Storage pool striped volumes make no sense without high availability
- VFS sees one device
- The device mapper sees the multibus or failover alternatives to the same disk
- The storage pool striped volume spans over several ranks of one server and overcomes the limitations of a single rank and / or a single device adapter
- Storage pool striped volumes can also be used as physical disks for a logical volume to make use of both server sides



I/O Processing Characteristics

- FICON/ECKD:
  - 1:1 mapping host subchannel:dasd
  - Serialization of I/Os per subchannel
  - I/O request queue in Linux
  - Disk blocks are 4 KiB
  - High availability by FICON path groups
  - Load balancing by FICON path groups and Parallel Access Volumes (PAV) or HyperPAV (if supported by the distribution and a storage server feature)
- FCP/SCSI
  - Several I/Os can be issued against a LUN immediately
  - Queuing in the FICON Express card and / or in the storage server
  - Additional I/O request queue in Linux
  - Disk blocks are 512 Bytes
  - High availability by Linux multipathing, type failover or multibus
  - Load balancing by Linux multipathing, type multibus

Setup Performance Considerations

- Speed up techniques
  - Use more than one rank: Storage Pool Striping SPS, Linux striped Logical Volume
  - Use more channels for ECKD
    - Use more than 1 subchannel: PAV, HyperPAV
    - High Performance FICON and Read Write Track Data
  - Use more channels for SCSI
    - SCSI Linux multipath multibus
    - For newer distro's with FICON Express8S cards use the datarouter option (put into kernel command line “zfcp.datarouter=1”)

Processor Consumption

- Linux features, like page cache, PAV, striped logical volume or multipath consume additional processor cycles
- Processor consumption
  - grows with number of I/O requests and/or number of I/O processes
  - depends on the Linux distribution and versions of components like device mapper or device drivers
  - depends on customizable values as e.g. Linux memory size (and implicit page cache size), read ahead value, number of alias devices, number of paths, rr_min_io setting, I/O request size from the applications
- HyperPAV and static PAV in SLES11 consume less processor cycles compared to static PAV in older Linux distributions
- The processor consumption in the measured scenarios
  - has to be normalized to the amount of transferred data
  - differs between a simple and a complex setup
    - for ECKD up to 2x
    - for SCSI up to 2.5x

Linux and Hardware options

Linux options	Hardware options

▪ Choice of Linux distribution
▪ Appropriate number and size of I/Os
▪ File system
▪ Placement of temp files
▪ Direct I/O or page cache
▪ Use of striped logical volume
  • ECKD
  • HyperPAV
  • High Performance FICON for small I/O requests
  • SCSI
  • Single path configuration is not recommemded
  • Multipath multibus – choose rr_min_io value appropriate
	▪ FICON Express8 or 8S
	▪ Number of channel paths to the storage server
	▪ Port selection to exploit maximum link speed
	▪ No switch interconnects with small bandwidth
	▪ Storage server configuration
	  • Extent pool definitions
	  • Storage pool striped volumes

Questions ?

- Further information is located at
  - Linux on System z – Tuning hints and tips
    http://www.ibm.com/developerworks/linux/linux390/perf/index.html
  - Live Virtual Classes for z/VM and Linux
    http://www.vm.ibm.com/education/lvc/