

# IT of SPIM Data Storage and Compression

EMBO Course - August 27th

Jeff Oegema, Peter Steinbach, Oscar Gonzalez

# Talk Outline

- Introduction and the IT Team
- SPIM Data Flow
- Capture, Compression, and the Data Volume Problem
- Transfer, Network and Storage Infrastructure
- Planning for SPIM

# People Involved - IT Staff



**Peter Steinbach ([steinbac@mpi-cbg.de](mailto:steinbac@mpi-cbg.de))**  
**Scientific / HPC Software Development**  
Data Streaming Library  
Compression and HPC Algorithm Development



**Ian Henry ([henry@mpi-cbg.de](mailto:henry@mpi-cbg.de))**  
**Scientific Computing Leader**  
Scientific and Project Coordination  
Collaboration Management



**Jeff Oegema ([joegema@mpi-cbg.de](mailto:joegema@mpi-cbg.de))**  
**IT Coordinator**  
Overall Project Coordination  
External Collaboration Management

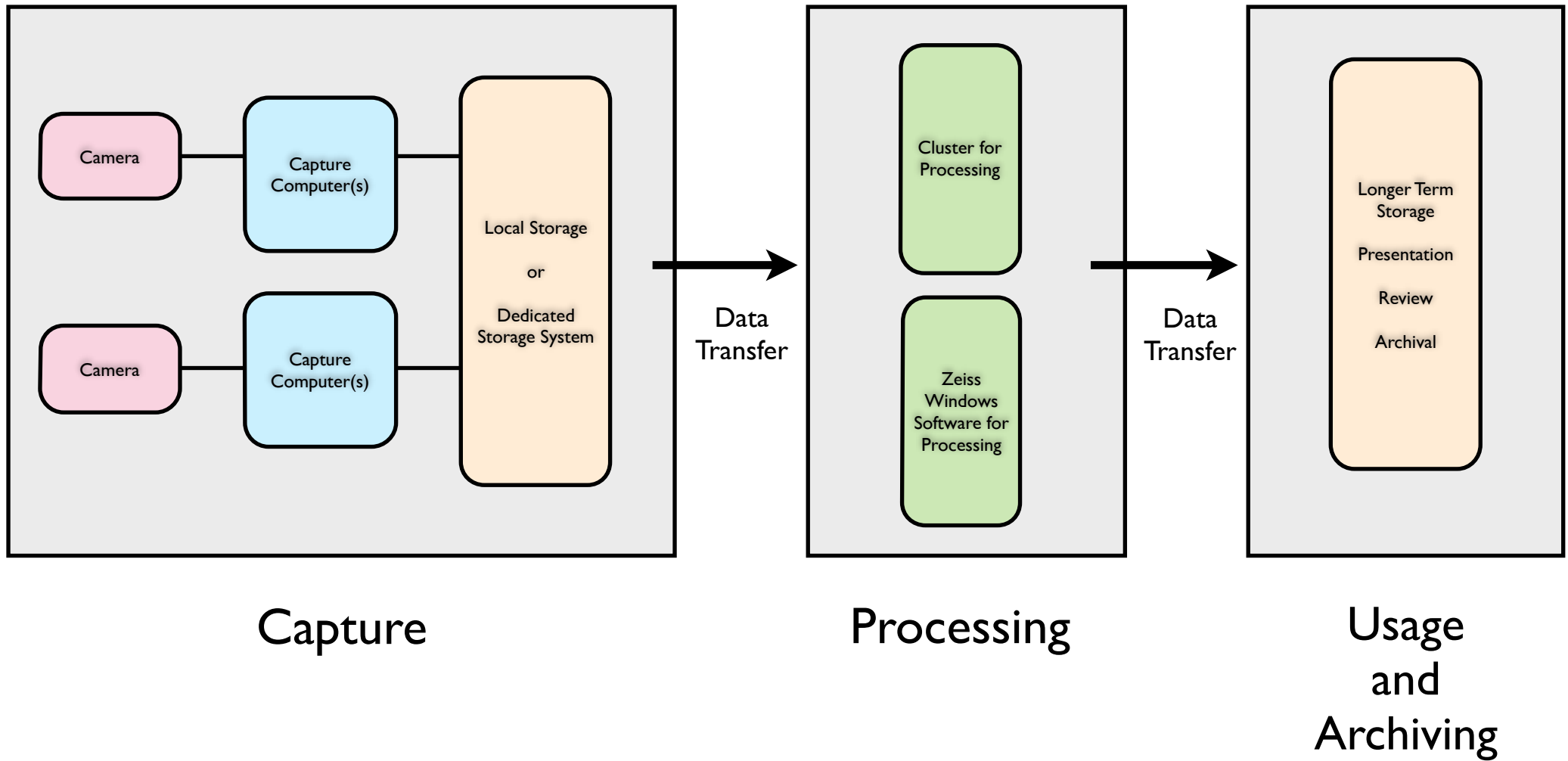
**Oscar Gonzalez ([ogonzale@mpi-cbg.de](mailto:ogonzale@mpi-cbg.de))**  
**HPC Administrator**  
Cluster Interaction and Queuing  
High-Performance Storage / Lustre  
Network Benchmarking and Performance Tuning



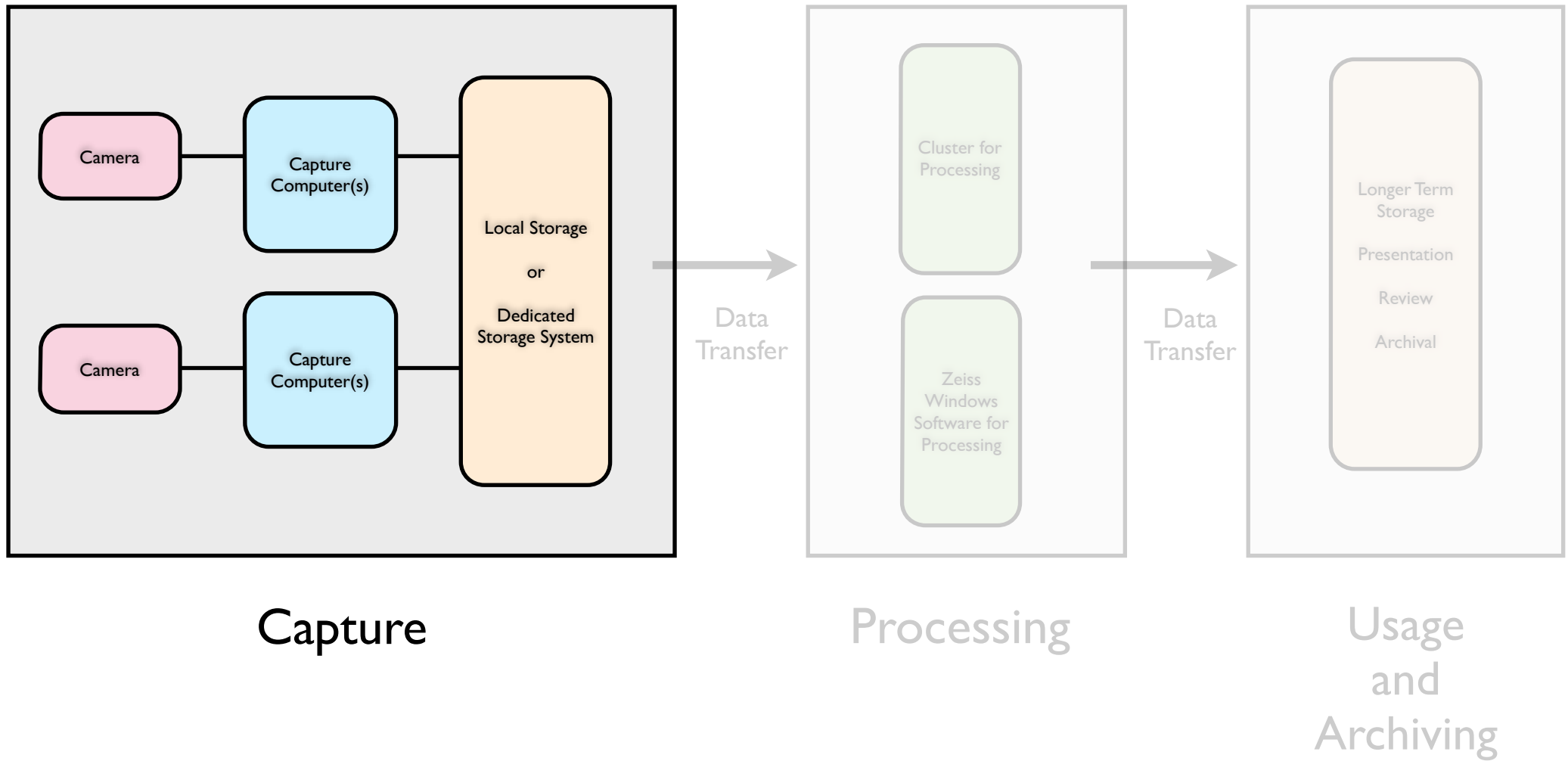
**Matt Boes ([boes@mpi-cbg.de](mailto:boes@mpi-cbg.de))**  
**Infrastructure Team Leader**  
Network Design and Development  
Fileserver Design and Development



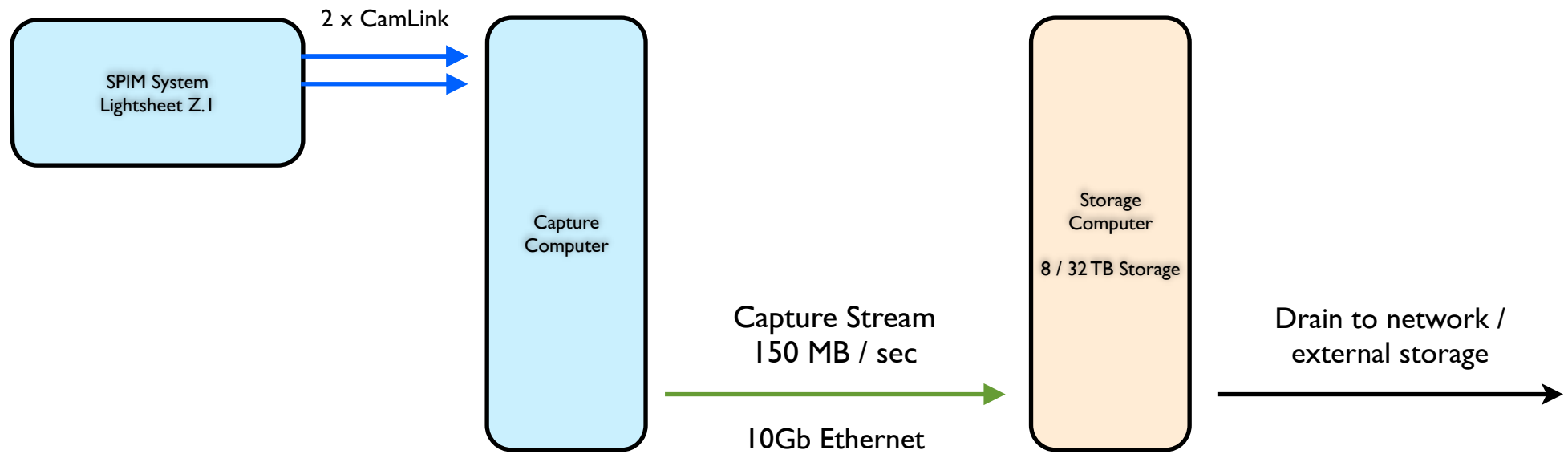
# SPIM Dataflow



# SPIM Dataflow



# Zeiss Lightsheet Z.I



# The Potential Deluge

Developmental SPIM - Camera Potential - 138 TB / Day

82 TB - Estimated CERN Data Production / Day

Single Lightsheet ZI Capture - 1 day

**13 TB**

5 50 GB - Confocal - 1 day

# The Potential Deluge - Multiple Lightsheet Z.I

$150 \text{ MB / sec} \times 4 = 600 \text{ MB / sec}$

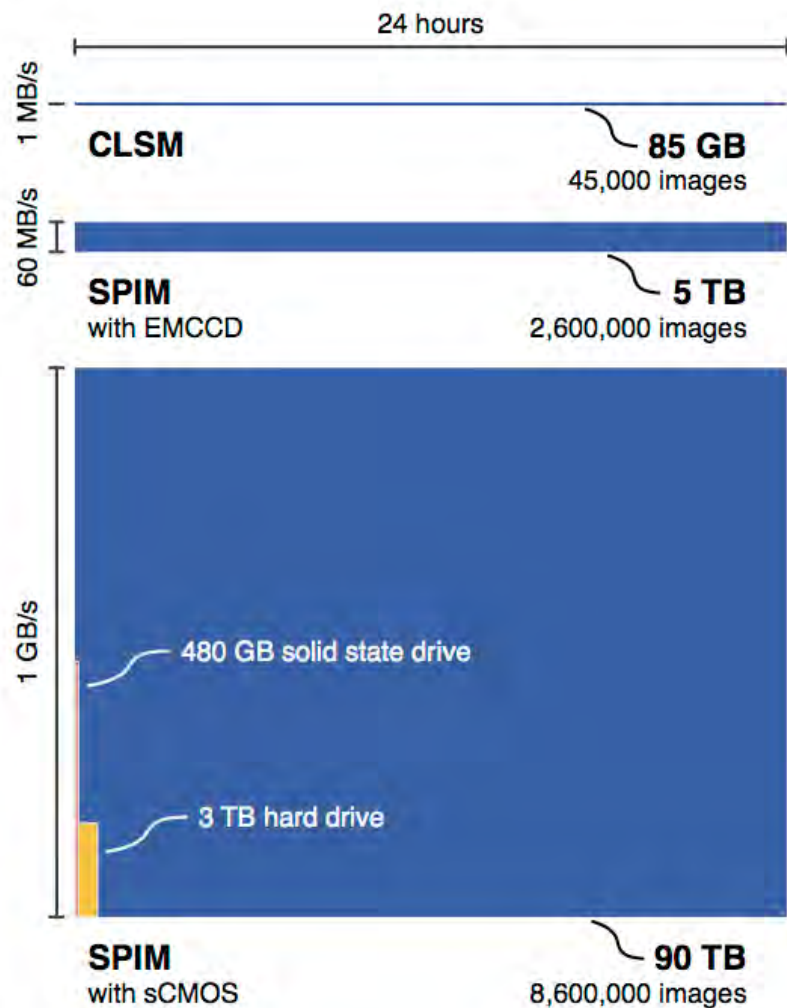
$\sim 52 \text{ TB / day}$

$\sim 364 \text{ TB / week}$

Our entire online disk storage (fileserver)  
currently is 700 TB



# The Potential Deluge - Future Tech



$800 \text{ MB / sec} \times 2 \text{ cameras} = 1.6 \text{ GB / sec}$

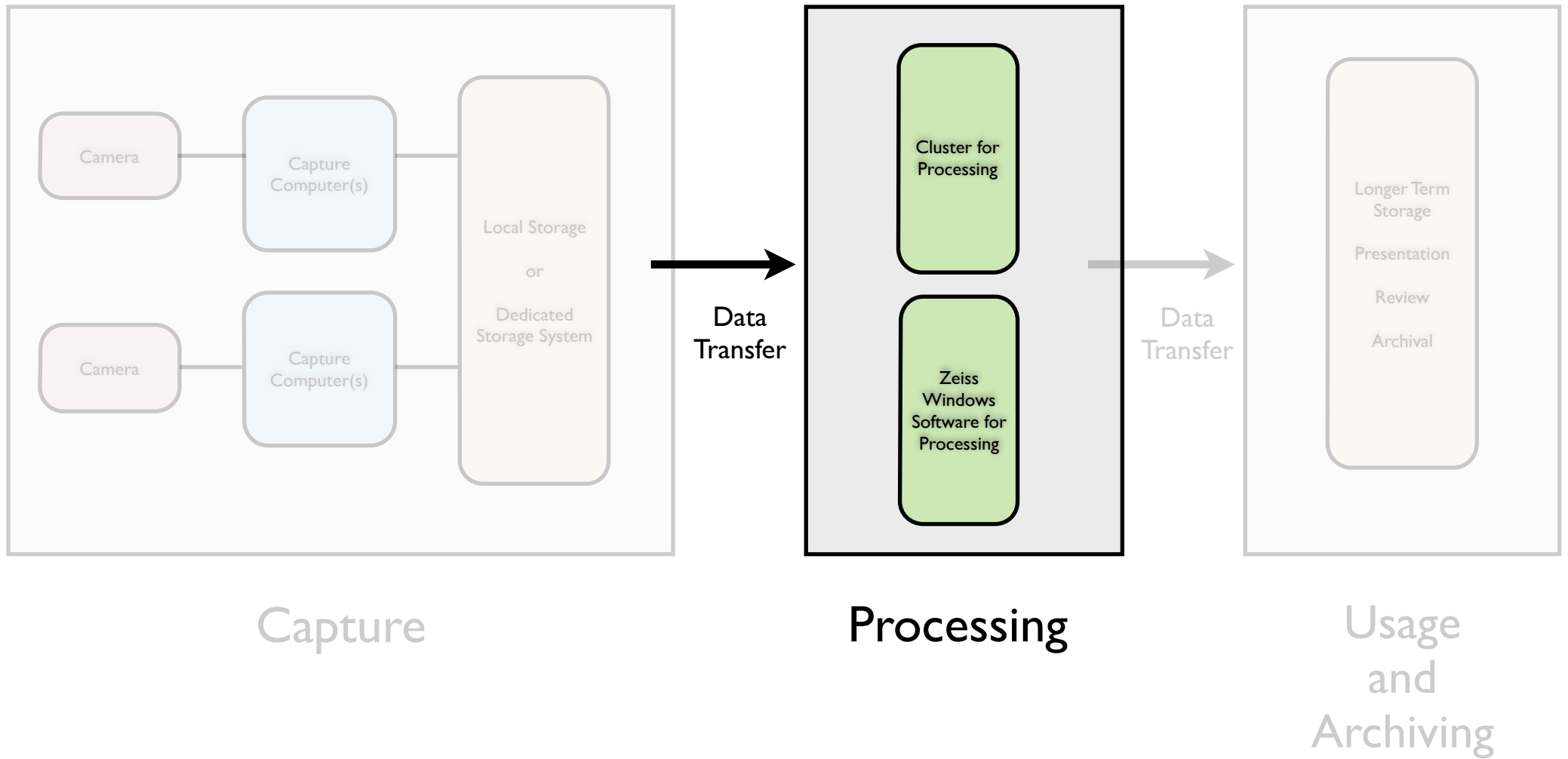
$\sim 138 \text{ TB / day}$

Almost a PB per week

CERN produces 30 PB of data annually from LHC experiments\*

\* <http://home.web.cern.ch/about/computing>

# SPIM Dataflow





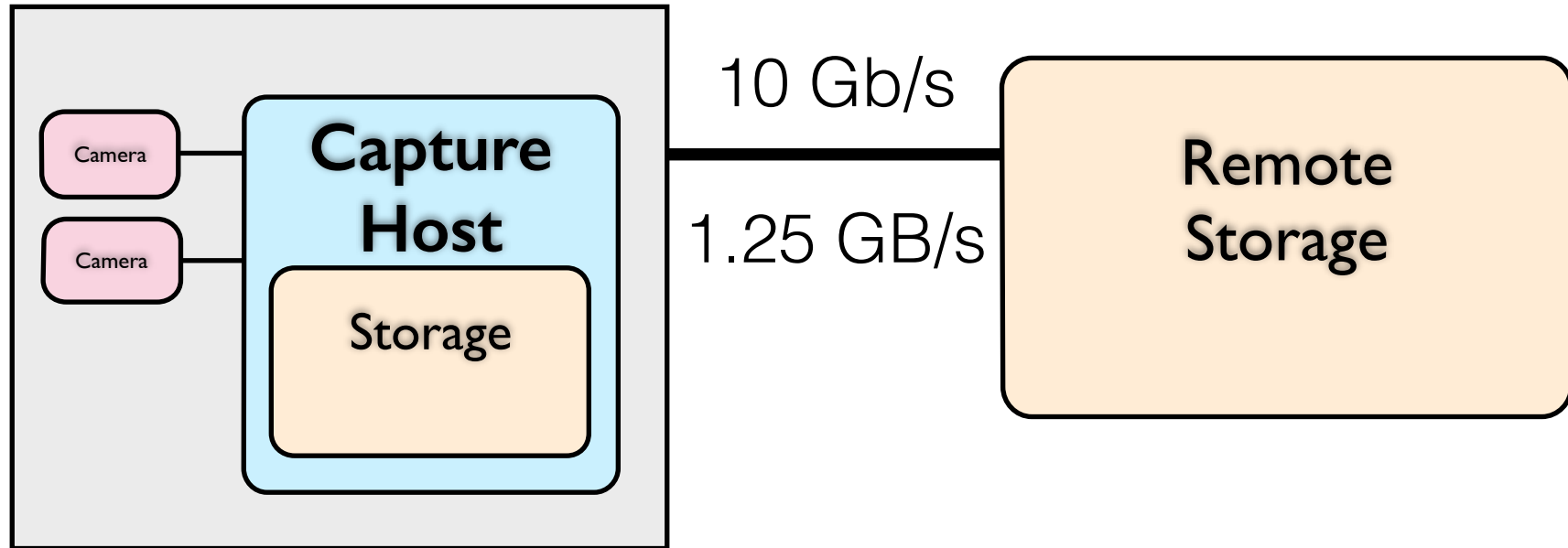
# Transfer Volumes and Times

Data Volume / Time	1 Gbit	10 Gbit
150 MB / sec	1.5 sec	.15 sec
9 GB / minute	90 sec	9 sec
540 GB / hour	1.5 hours	9 minutes
~13 TB / day	1.5 days	3.6 hours

This assumes approximately theoretical maximum line speed - which never happens. Typically we see 60%.



# First : Get the Data Off

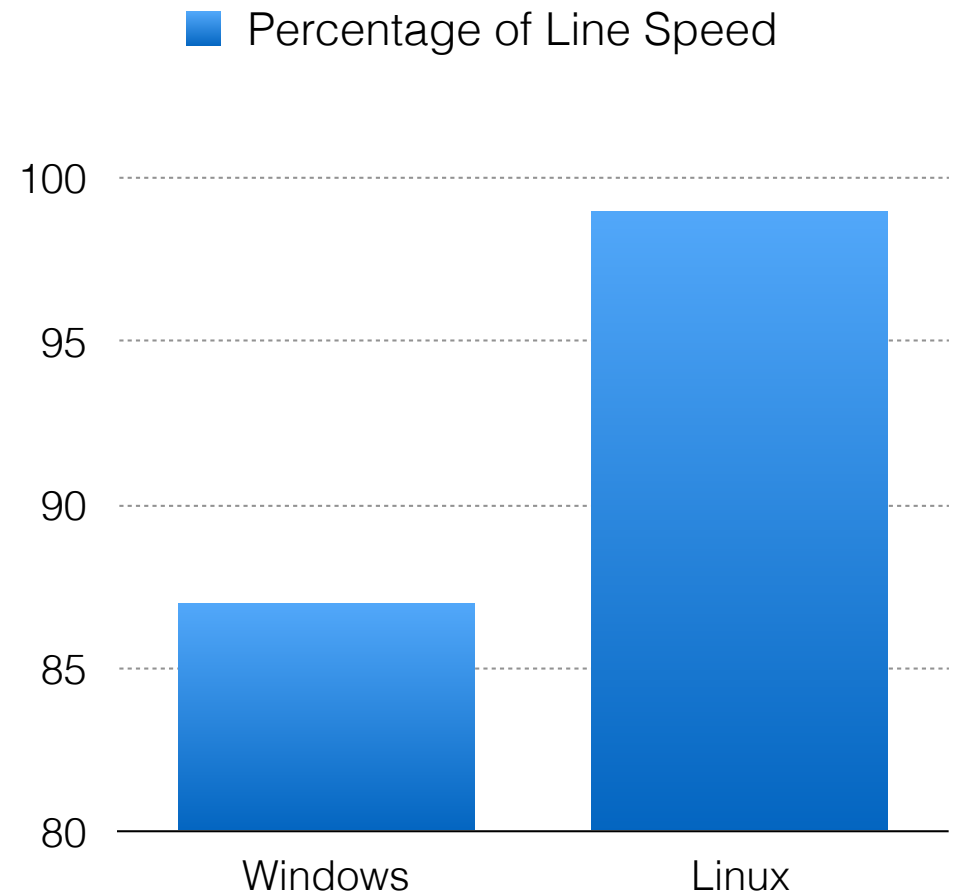


	pro	con
<b>network mounted drives</b> (ex. SMB)	simple	OS dependent
<b>secure network file transfer</b> (scp/sftp/rsync)	secure	encryption may slow transfer
<b>unencrypted file transfer</b> (ex. ftp)	fast	insecure



# Operating Systems & Networking

- Extensive Network Streaming Tests
- Win7, Windows Server 2008 R2
- 10 Gbit/s fiber network
- same hardware
- No disk i/o involved





# Networks are ...

5



a  
shared  
resource!





# Network File Transfer

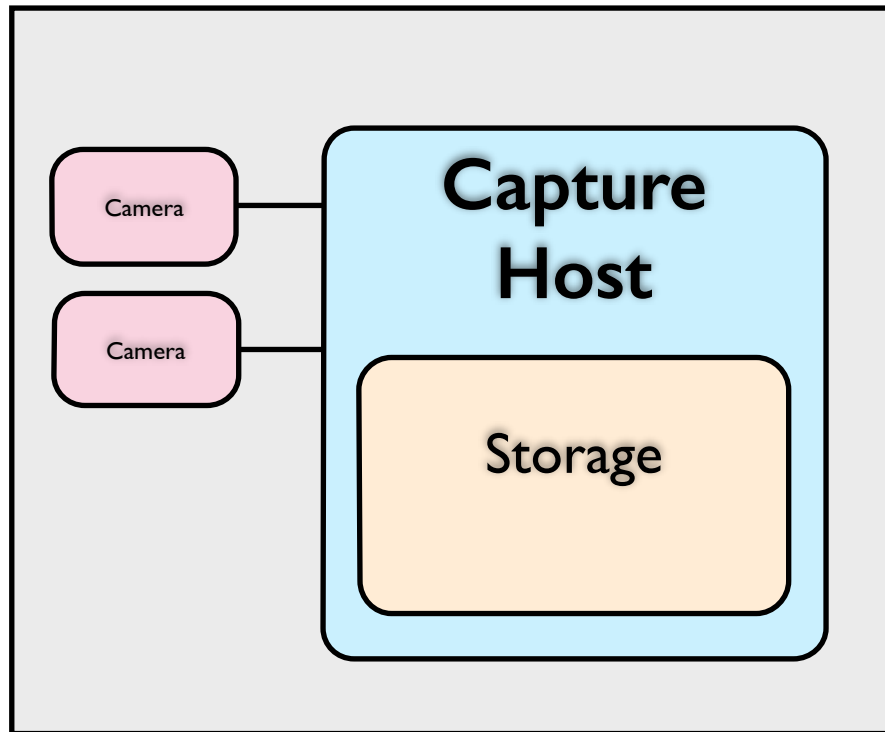
- is **necessary**  
( capture host becomes unusable / full )
- **protocols are important** to keep in mind
- **network is a shared resource**



<http://erindriver.travellerspoint.com/148/>



## Second : Bottlenecks again



### Spinning disk based storage

- Large Volume
- Comparatively cheap

### SSD based storage

- Fast
- Small Volume
- Expensive

### SPIM Operation

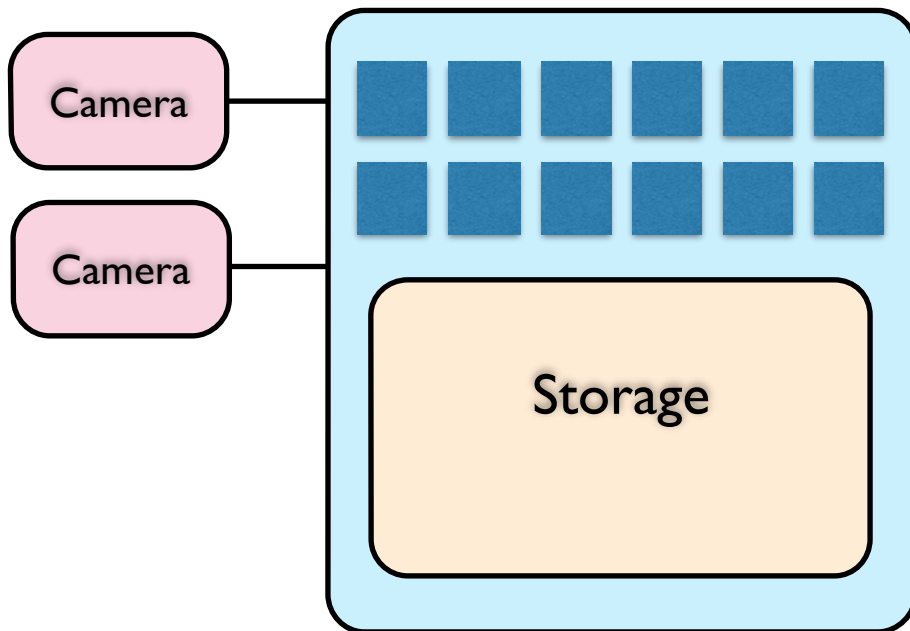






## Second : Get it Small

### Capture Host



- **cropping**  
(only keep what you need)
- **fusion + deconvolution**  
(n stacks become 1)
- **compression**
- ...

Reduce data volume **before** any network or disk!



## Compression : Demonstration

**Please zip a SPIM dataset of  
your choice!**

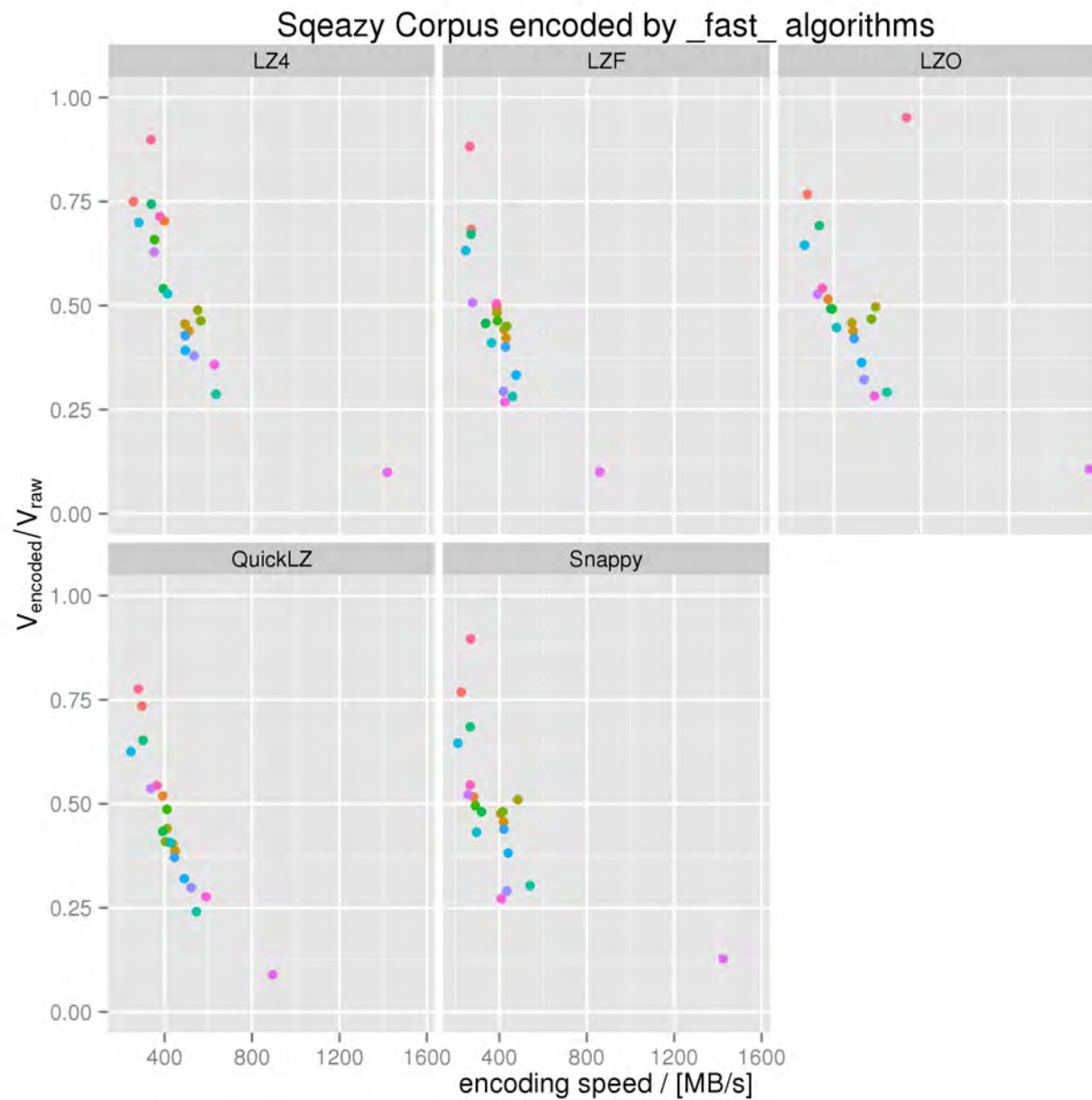
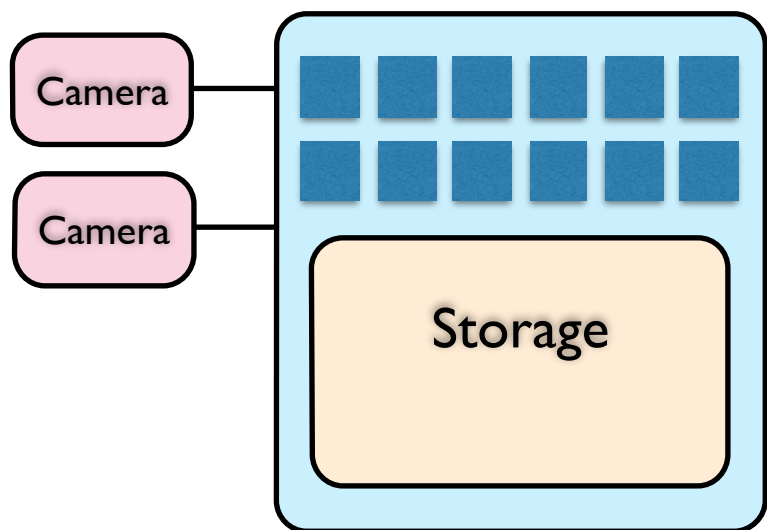
How long did it take?

How small is the compressed  
data?



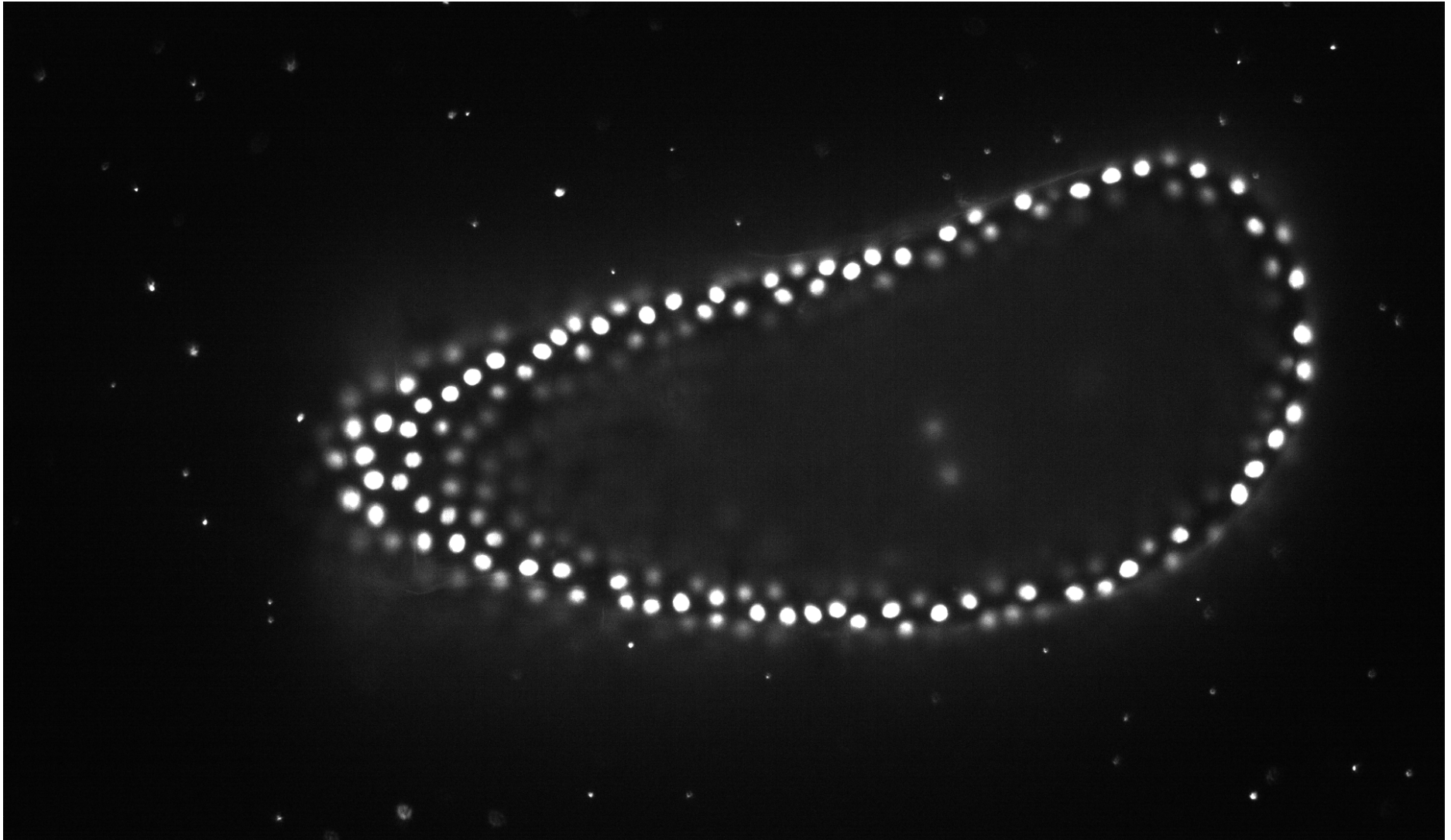
# Compression : Fast

## Capture



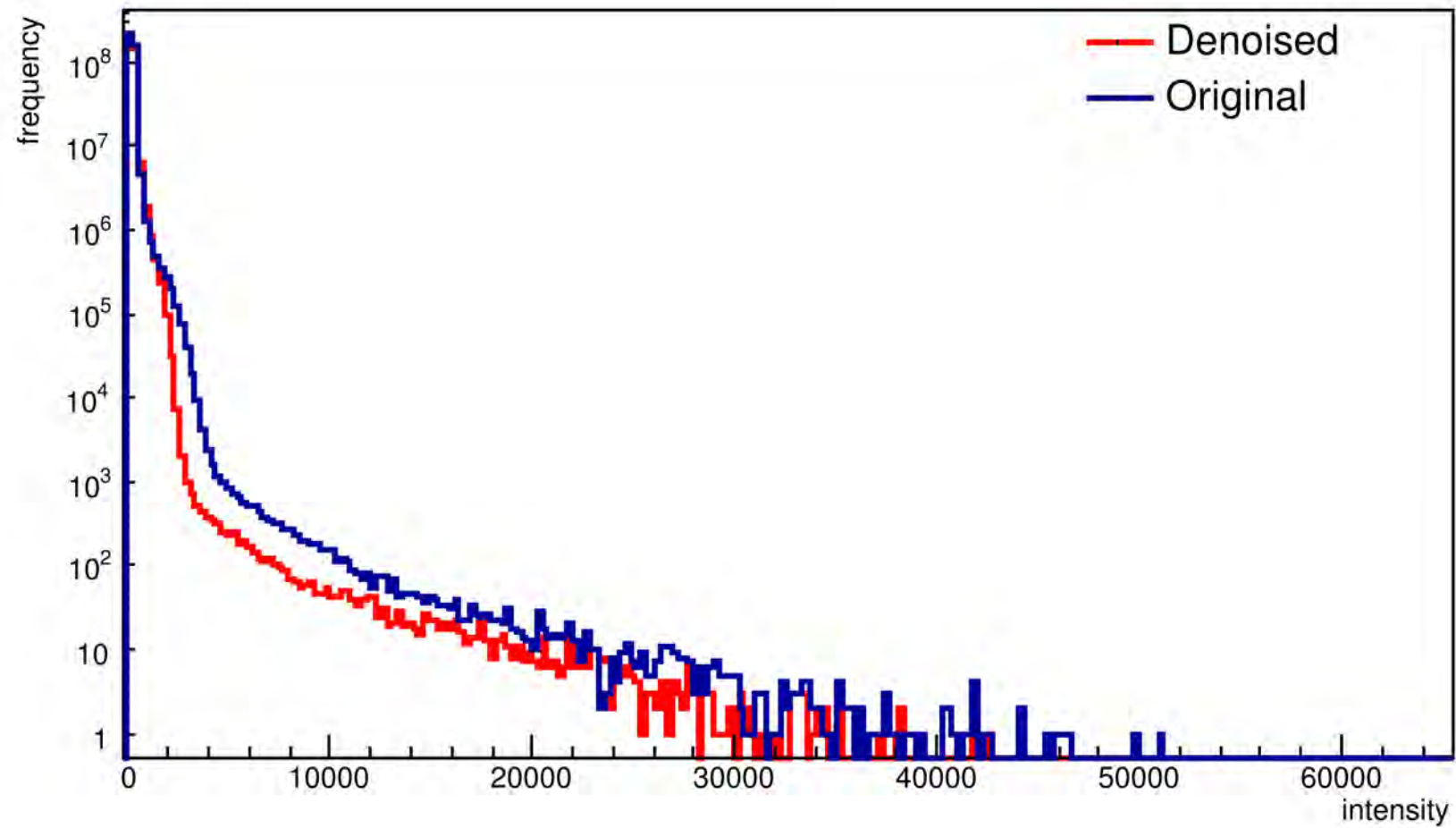


# Compression : Noise?





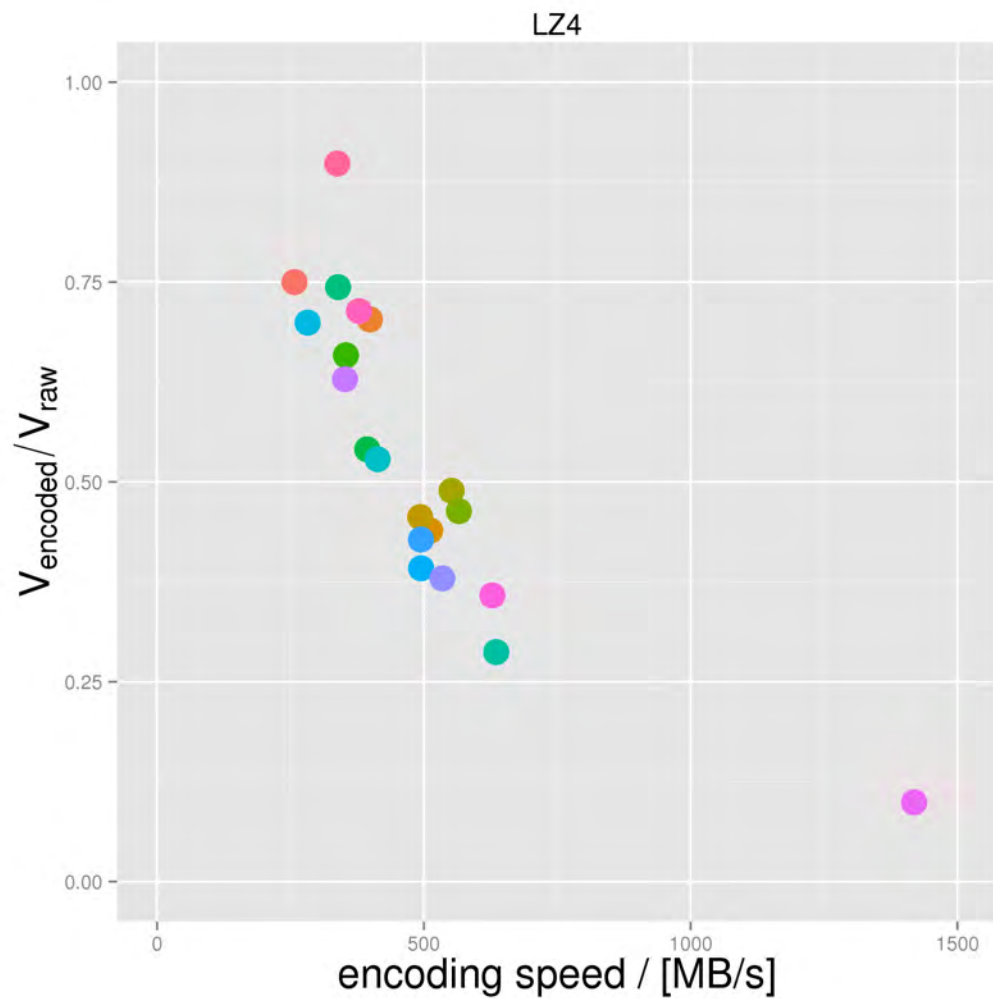
# Compression : Denoised



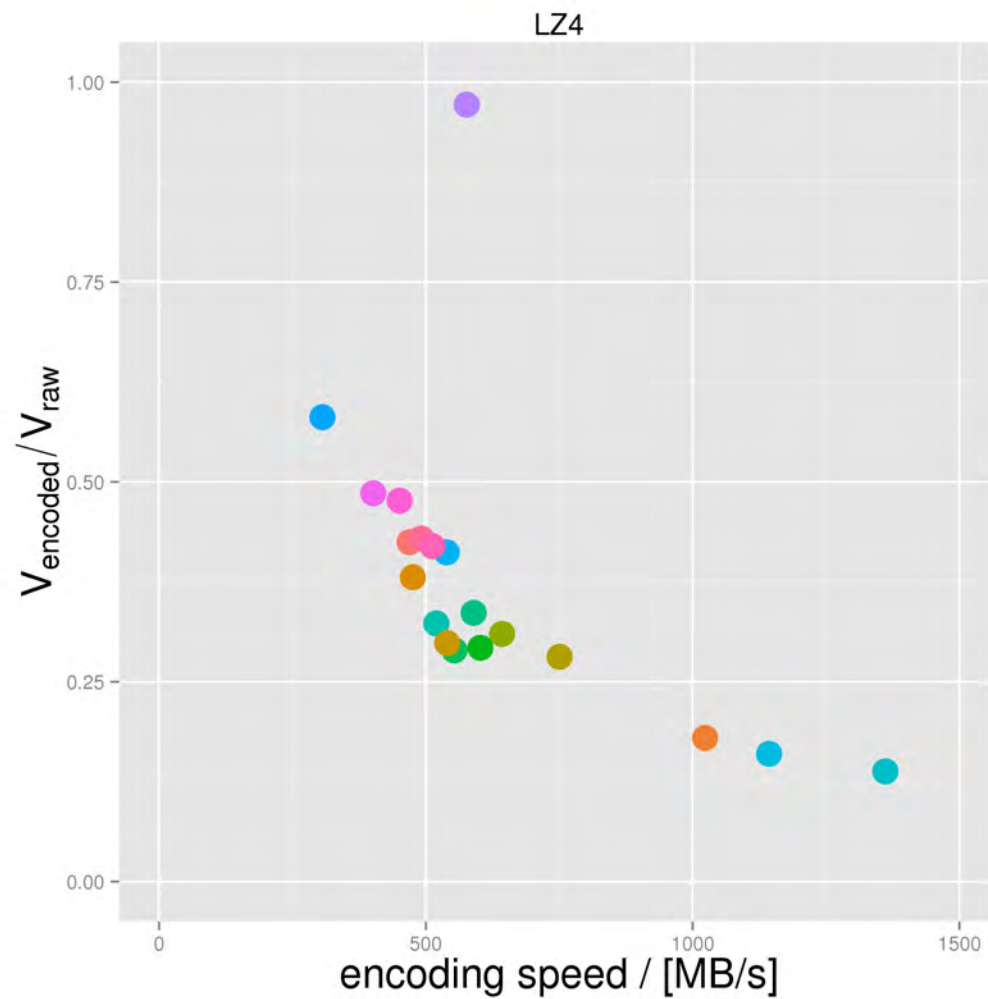




# Compression : Denoised



Original



Denoised



# Compression : Squeazy

The screenshot shows the Bitbucket interface for a repository named 'Squeazy' by user 'psteinb'. The repository is a fork of 'royerloic/Squeazy' and is written in the C language. It has an 'Admin' access level. The repository statistics show 1 branch, 0 tags, 0 forks, and 1 watcher. The last update was on 2014-08-22. The left sidebar lists actions: Clone, Create branch, Create pull request, and Compare.

Overview		1 Branch	0 Tags
Last updated	2014-08-22		
Fork of	royerloic/Squeazy		
Language	C		
Access level	Admin	0 Forks	1 Watcher

- pipeline standard
- compression algorithms fast
- soon to be open-sourced
- currently:
  - 3x lossless compression
  - 10x lossy compression

initiated by:

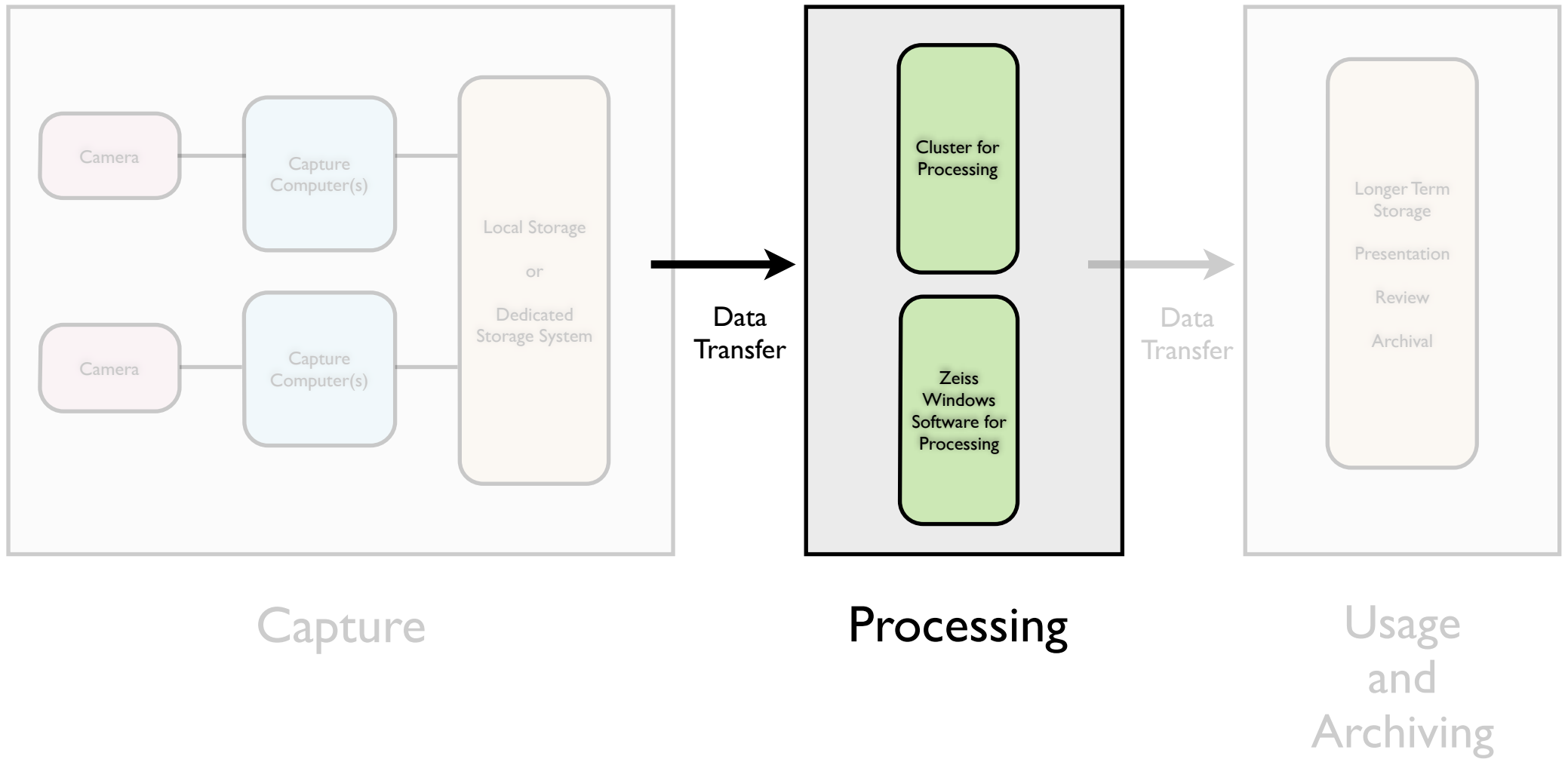


Loic Royer



Martin Weigert

# SPIM Dataflow





# HPC for SPIM



Lots of image data

# HPC for SPIM



Lots of



CPU intensive



# HPC for SPIM



Lot

High memory footprint

# HPC for SPIM



Lot

otprint

High I/O



# HPC for SPIM



Lots of



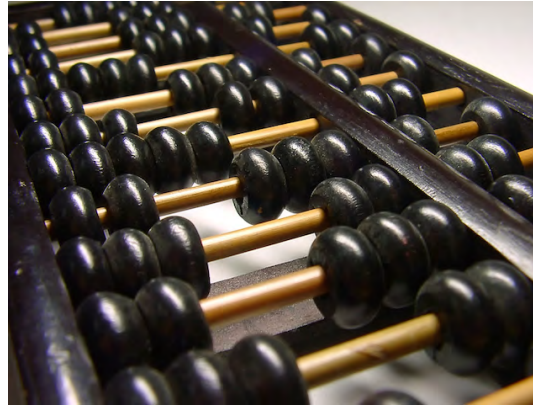
footprint

GPUs are promising

# HPC for SPIM



Lots of image data



CPU intensive



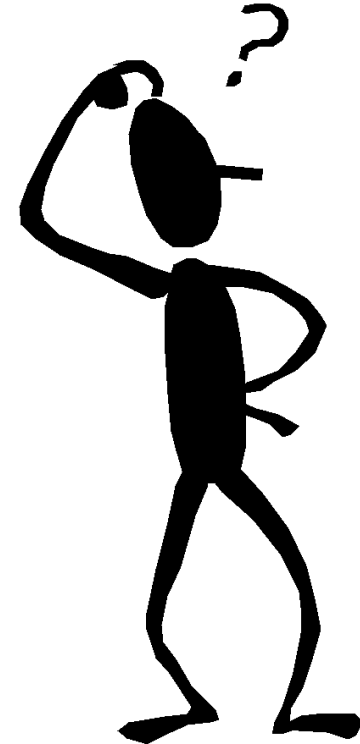
High memory footprint



High I/O



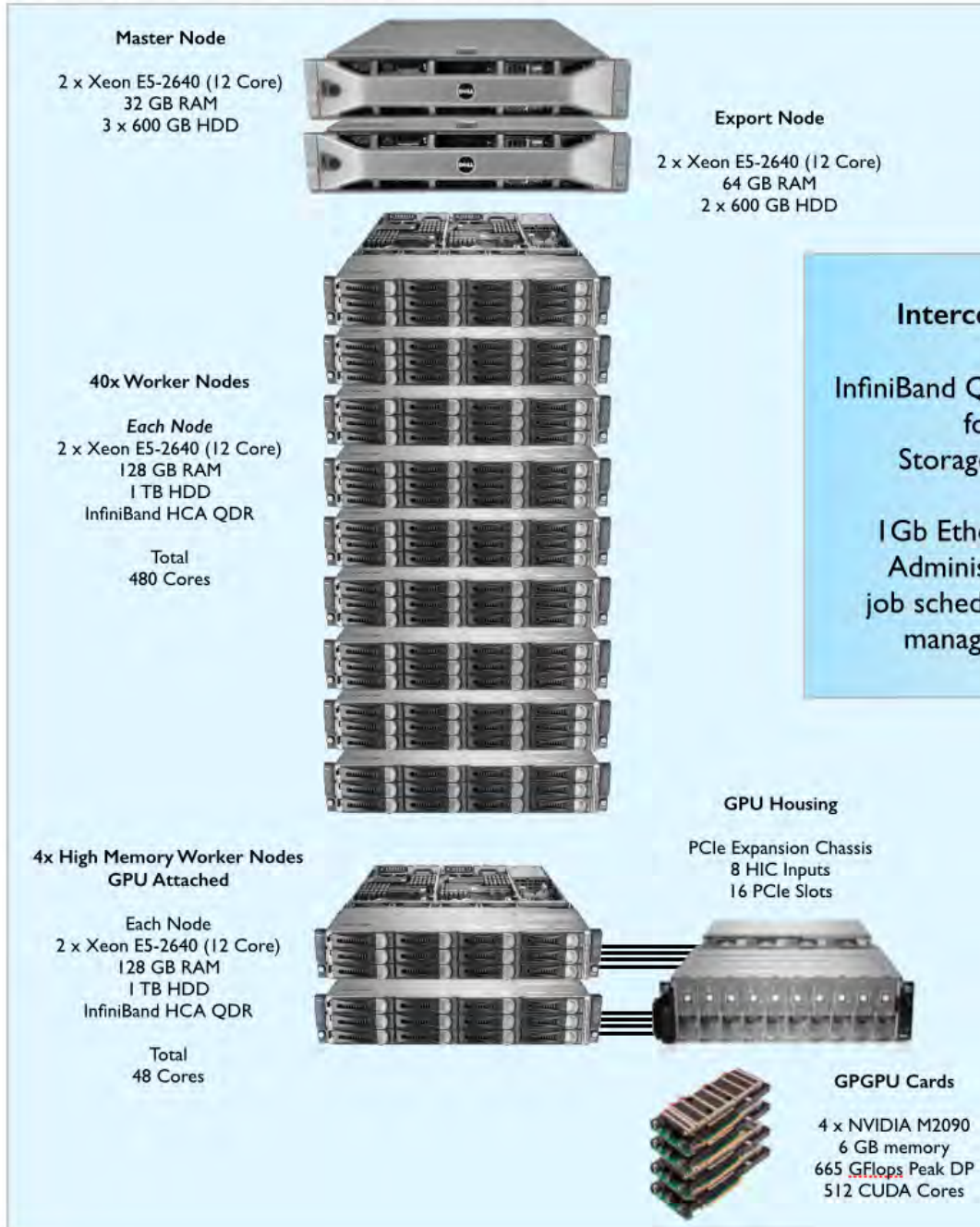
GPUs are promising



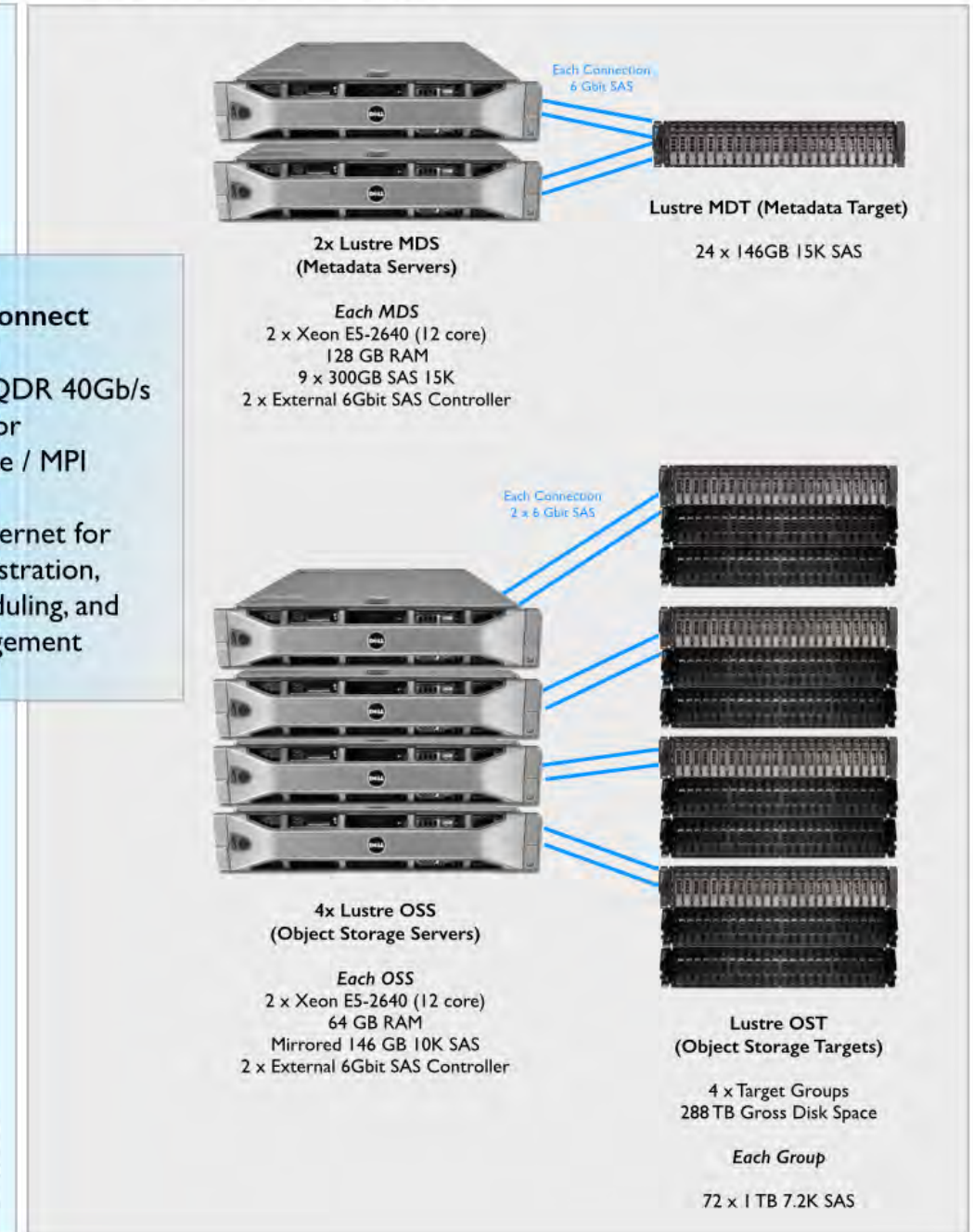


# Cluster Architecture

## Compute and Processing Architecture



## Lustre Storage Architecture



**Interconnect**  
InfiniBand QDR 40Gb/s  
for  
Storage / MPI  
  
1 Gb Ethernet for  
Administration,  
job scheduling, and  
management

# Cluster Architecture

## Head node

Job management  
Cluster monitoring

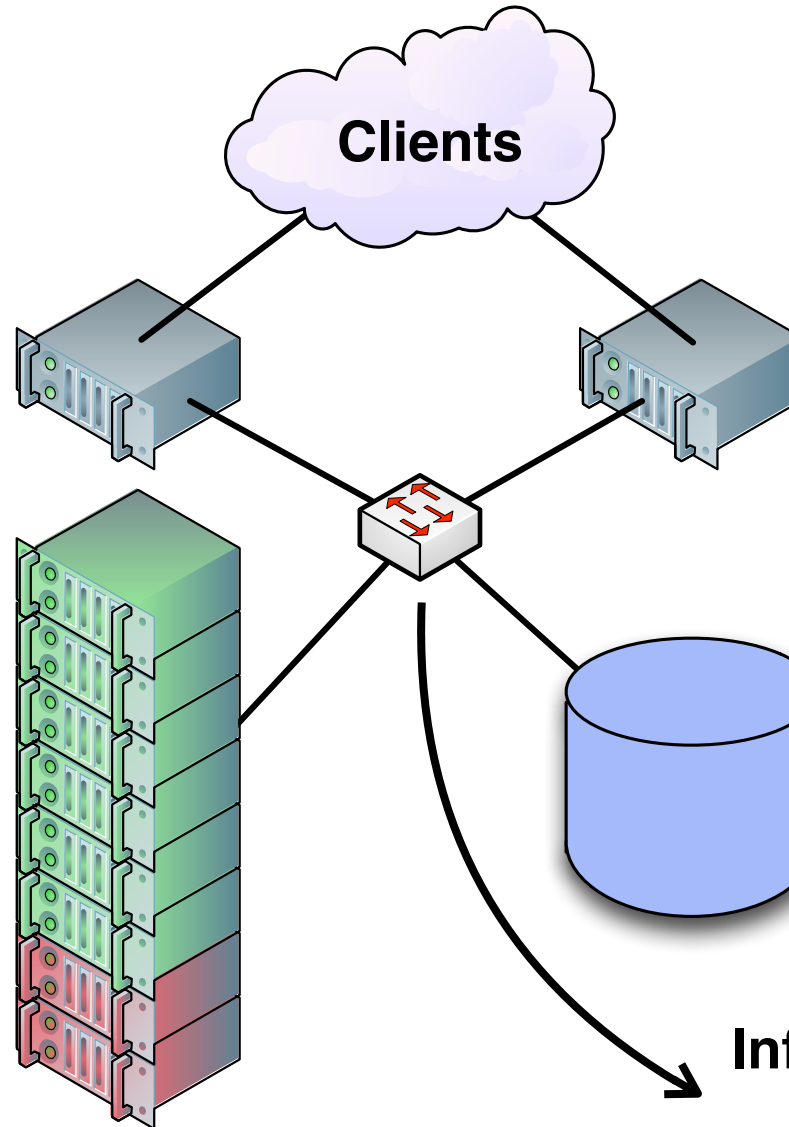
## Worker nodes

40x:

- \* 12 cores
- \* 128 GB RAM
- \* 1 TB HDD

4x:

- \* 12 cores
- \* 128 GB RAM
- \* 1 TB HDD
- \* GPU



## Export node

10 GbE

## Disk server

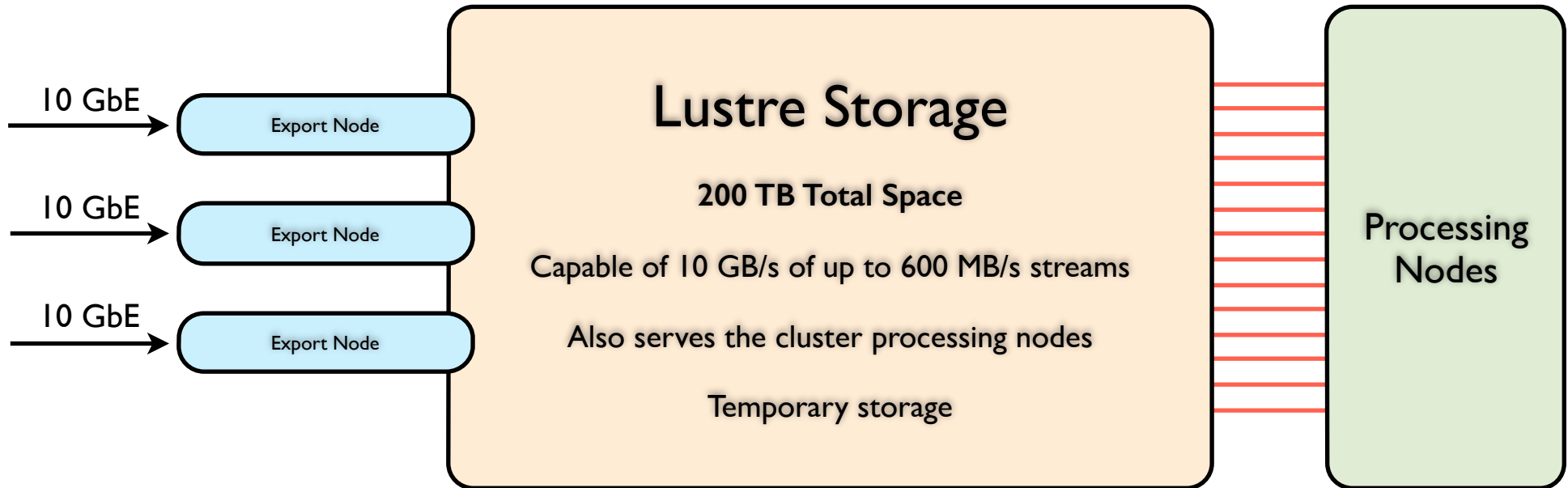
200 TB net space (RAID 6)  
10 GB/s

## InfiniBand

non-blocking  
400-600 MB/s (up to saturation)



# MPI-CBG Cluster Storage



# Resource Usage

The image shows handwritten mathematical derivations for the maximum likelihood estimation of the normal distribution parameters. The derivations include the log-likelihood function, its partial derivatives with respect to the mean  $\mu$  and variance  $\sigma^2$ , and the resulting maximum likelihood estimates.

$$\ln(L(\mu, \sigma^2)) = \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu)^2\right)$$
$$\ln(L(\mu, \sigma^2)) = \frac{n}{2} \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu)^2$$
$$\frac{\partial \ln(L(\mu, \sigma^2))}{\partial \mu} = \frac{1}{\sigma^2} \sum_{i=1}^n (x_i - \mu) = 0$$
$$\sum_{i=1}^n (x_i - \mu) = 0$$
$$\sum_{i=1}^n x_i - n\mu = 0$$
$$\mu = \frac{\sum_{i=1}^n x_i}{n} = \bar{x}$$
$$\frac{\partial \ln(L(\mu, \sigma^2))}{\partial \sigma^2} = \frac{n}{2} \left( -\frac{1}{\sigma^2} \right) + \frac{1}{2\sigma^4} \sum_{i=1}^n (x_i - \mu)^2 = 0$$
$$-\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4} \sum_{i=1}^n (x_i - \mu)^2 = 0$$
$$\frac{1}{2\sigma^4} \sum_{i=1}^n (x_i - \mu)^2 = \frac{n}{2\sigma^2}$$
$$\sum_{i=1}^n (x_i - \mu)^2 = n\sigma^2$$
$$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}$$
$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}}$$
$$\ln(L(\bar{x}, \sigma)) = \frac{n}{2} \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \bar{x})^2$$
$$\ln(L(\bar{x}, \sigma)) = \frac{n}{2} \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \bar{x})^2$$
$$\ln(L(\bar{x}, \sigma)) = \frac{n}{2} \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \bar{x})^2$$

- The cluster was made available on Feb 2013
- Total number of jobs done: 6,852,661
- Average throughput: 462 jobs/h
- CPU time consumed: 151y 46d 10h 59m 12s
- Average CPU time: 11m 35s

# Lessons Learned



- Cluster design is very important - think before you buy
- I/O is critical to move data in and out of the cluster
- I/O is VERY critical to access data from the cluster
- Storage requirements are huge, both inside and outside the cluster
- GPU resources might be useful but you need enough to make it practical

# Workstations

If/when a cluster is not an option, check what your WS can do.

## Example Data PC

- 12 cores
- 128 GB
- 4x 2TB (RAID 5)

### Pros:

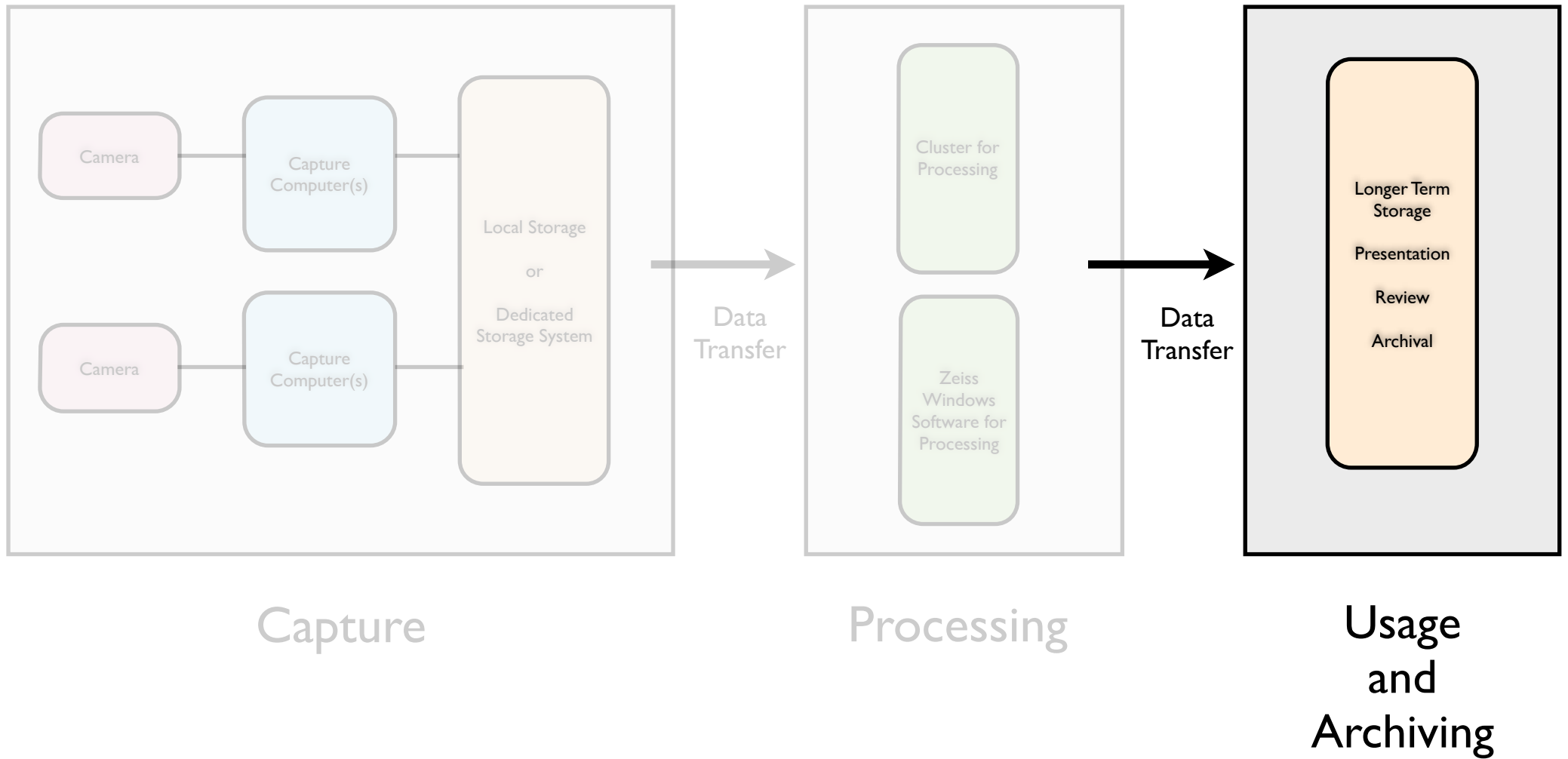
- Rather cheap
- Fine for small datasets
- Convenient for data visualisation

### Cons:

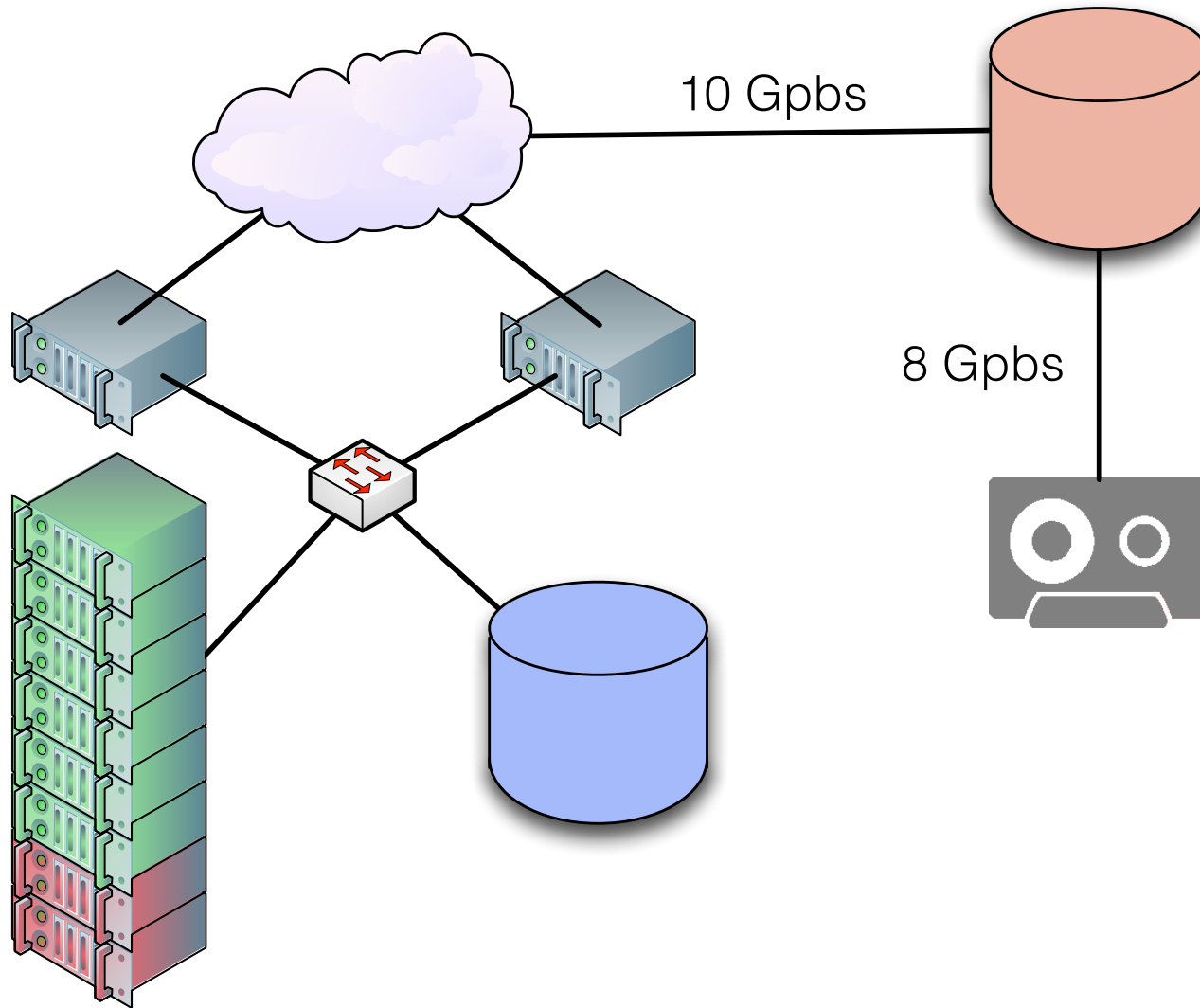
- Limited computing resources
- Limited storage capacity and bandwidth



# SPIM Dataflow



# Current Infrastructure



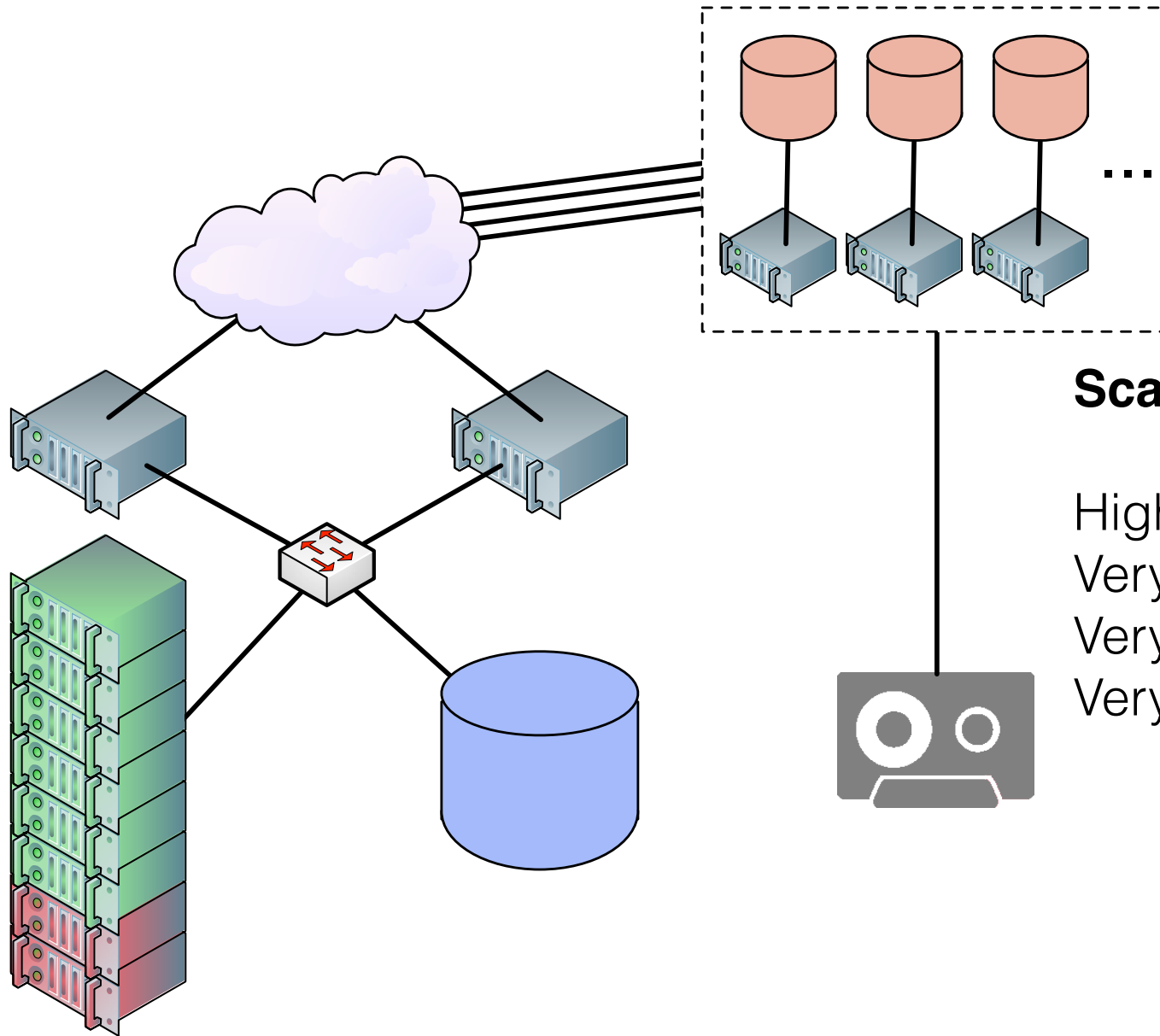
## Fileserver

Visible network-wide  
High capacity (900 TB)  
Robust (no SPOF)  
Backed up daily

## Tape library

Second copies  
Long term storage

# Future Plans



## Scale-out NAS

Highly scalable  
Very high capacity (10 PB)  
Very high bandwidth  
Very robust (no SPOF)



## Taking it Home - External Drives

### 4TB transfer

#### Protocol

USB 3.0 (600 MB / sec)	1.85 hours
USB 2.0 (60 MB / sec)	18.5 hours

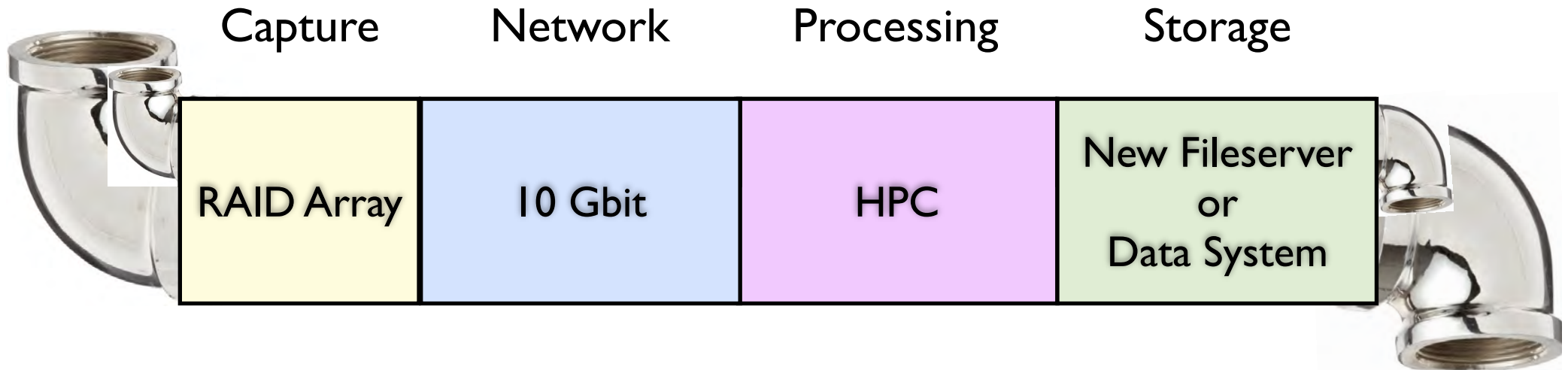
#### Drive Speed

WD Black (130.4 MB / sec)	8.52 hours
Hitachi Deskstar (102.95 / MB sec)	10.79 hours
Samsung SSD - 1 TB (550 MB /sec)	2.02 hours

The limitation is the slower of the two!



# Bottlenecks at Each Stage



Bottlenecks can be addressed but the pipeline can't  
be made infinitely wide

Experiment Design and Data Management become  
extremely important!

Compression can help but the issue remains

# IT Planning for SPIM

(or “things to think about before I capture”)

What is the practical output of your SPIM setup?

How long are you planning on capturing at a time?

What processing do you need to do on your data?

How fast do you need to complete the processing?

What is the data you will consider primary data for publication?

How will you present your data to the world or turn it into movies or results more easily shared?



## Discussion and Questions