NFS around the world

Tigran Mkrtchyan for dCache Team

dCache User Workshop, Umeå, Sweden

The NFS community

History

- v1 – 1984, SUN Microsystems intern
  - 16 ops, 1:1 mapping to vfs
- 1986 – First Connectathon!
- v2 – 1989, rfc1094
  - 18 ops, vfs + placeholder
- v3 – 1994, rfc1813
  - 22 ops, vfs + weak cache control, 64bit file size

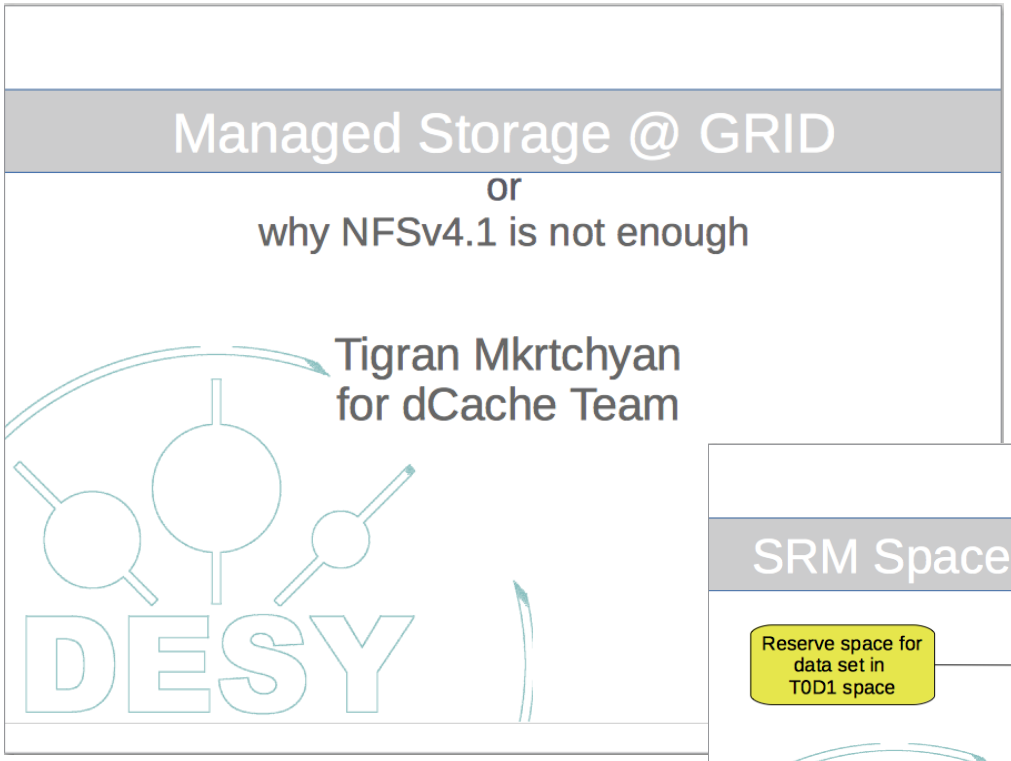
POSIX vs NFS

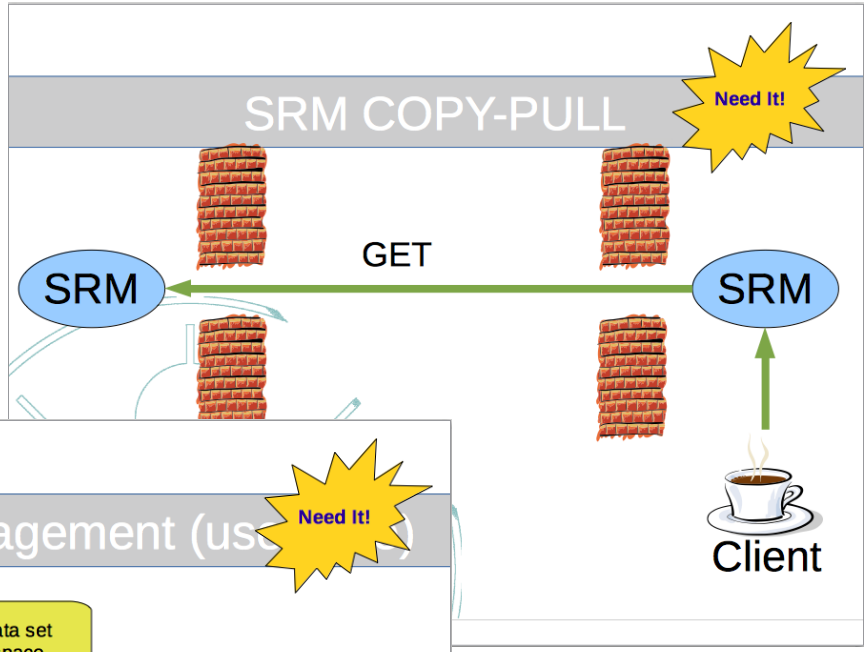
- POSIX is state full
  - open/close
  - lock/unlock
- NFS v{2,3} stateless
- You can’t (efficiently) map state full to stateless
  - (the same issue with posix-IO over HTTP)

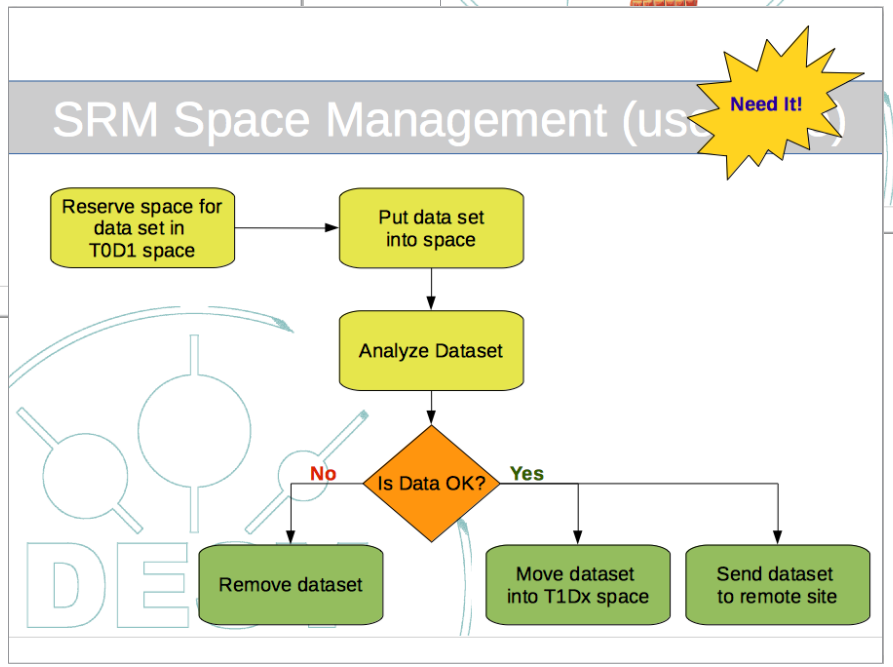
History (Cont)

- v4 – 2000, rfc3010, IETF joined effort
  - 35 ops.
  - state full
  - mandatory strong security
  - compound requests
- pNFS Problem Statement – 2004, IETF memo
- dCache.ORG join v4.1/pNFS development – 2006
  - 2008, dcache-1.9.3 first publicly available NFSv4.1/pNFS server
- v4.1 – 2010, rfc5661
  - 29 iterrations
  - 617 pages (v2 – 27 pages)
  - 2012, NetApp ONTAP-8.1 with pNFS

pNFS is not enough.... (2008)







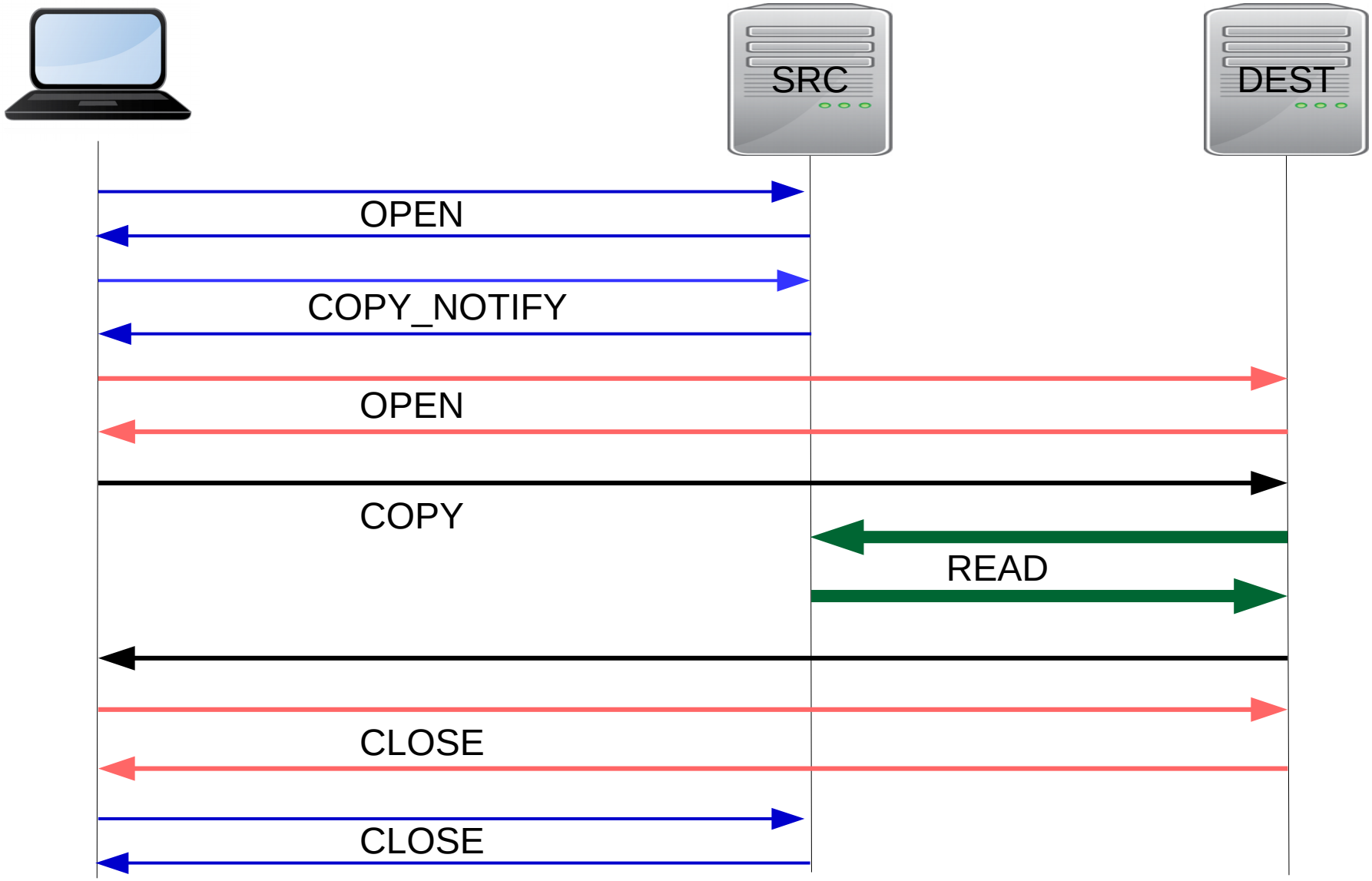
NFSv4.2

- v4.2 - 2016, rfc7862
  - Server-to-server copy and file initialization
    - 3rd party copy
    - space reservation
  - Sparse files: hole seeking and punching, sparse read
    - do not send zeros over the wire
  - Security labels
    - delegate access control to 3-rd party

Server to Server Copy (SSC)

- CLONE
  - Atomic COPY-like operation
  - possible with in same server
- COPY
  - possible between two server
  - possible within server
  - both server must support v4.2

Well, nothing new…



Server Side Copy implementation status

- Working prototype for Linux client and server
  - Sponsored by NetAapp
  - matures with each kernel release
- Exposed as copy_file_range syscall
  - Only for x86_32 and x86_64.
  - Uses 4MB chunks
  - Fall back to read+write if not supported (v3 mount)
- Error recovery is a challenge
  - Handle source server reboot
  - Handle destination server reboot

Other NFS development

- RPCGSS_SEC v3 (rfc7861)
  - multi-principal authentication, krb5 delegation
- Multi-domain Namespace Deployment
  - federated namespace deployment
  - federated user identity deployment
- XATTRS over NFS
- NFS over RDMA
  - pushed by Oracle
  - Linux client/server, Solaris client/server
- NFS4 MIGRATION
  - uninterruptedly moving data volume to an other server

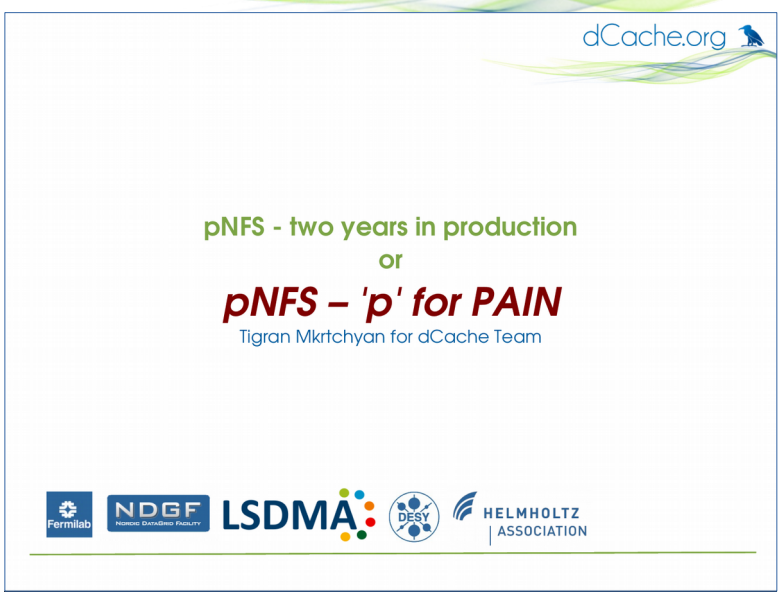
NFS: decode_first_pnfs_layout_type: Warning: Multiple pNFS layout drivers per filesystem not supported

(kernel version < 4.9)

pNFS layout:

- Describes how file's data spread over the data servers
- Which protocol data servers supports (layout types)
  - Block-
  - File-
  - Object-
- Server may offer multiple layout types
  - dCache 3.0
- Client can support multiple layout types
  - Linux kernel staring from 4.9

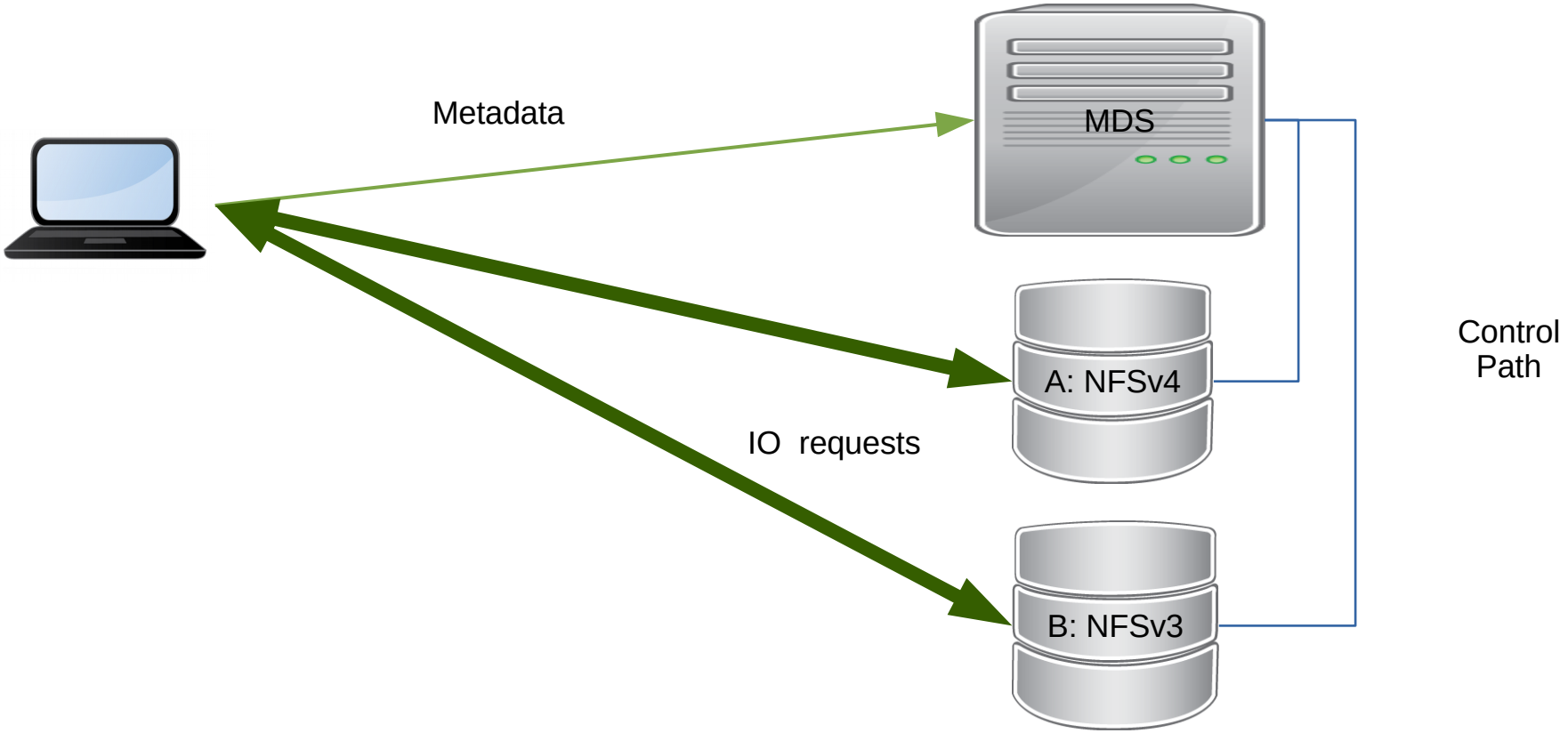
Flexfile layout



- New layout type
  - Allowed by NFSv4.1 spec
  - Pushed by PrimaryData
- Attempt to make pNFS right
  - Lesson learned from existing implementations (dCache)
- Supports nfsv3 and v4 on the data servers
- Client side mirroring
- Ability to propagate DS errors to MDS
  - Introduces in v4.2

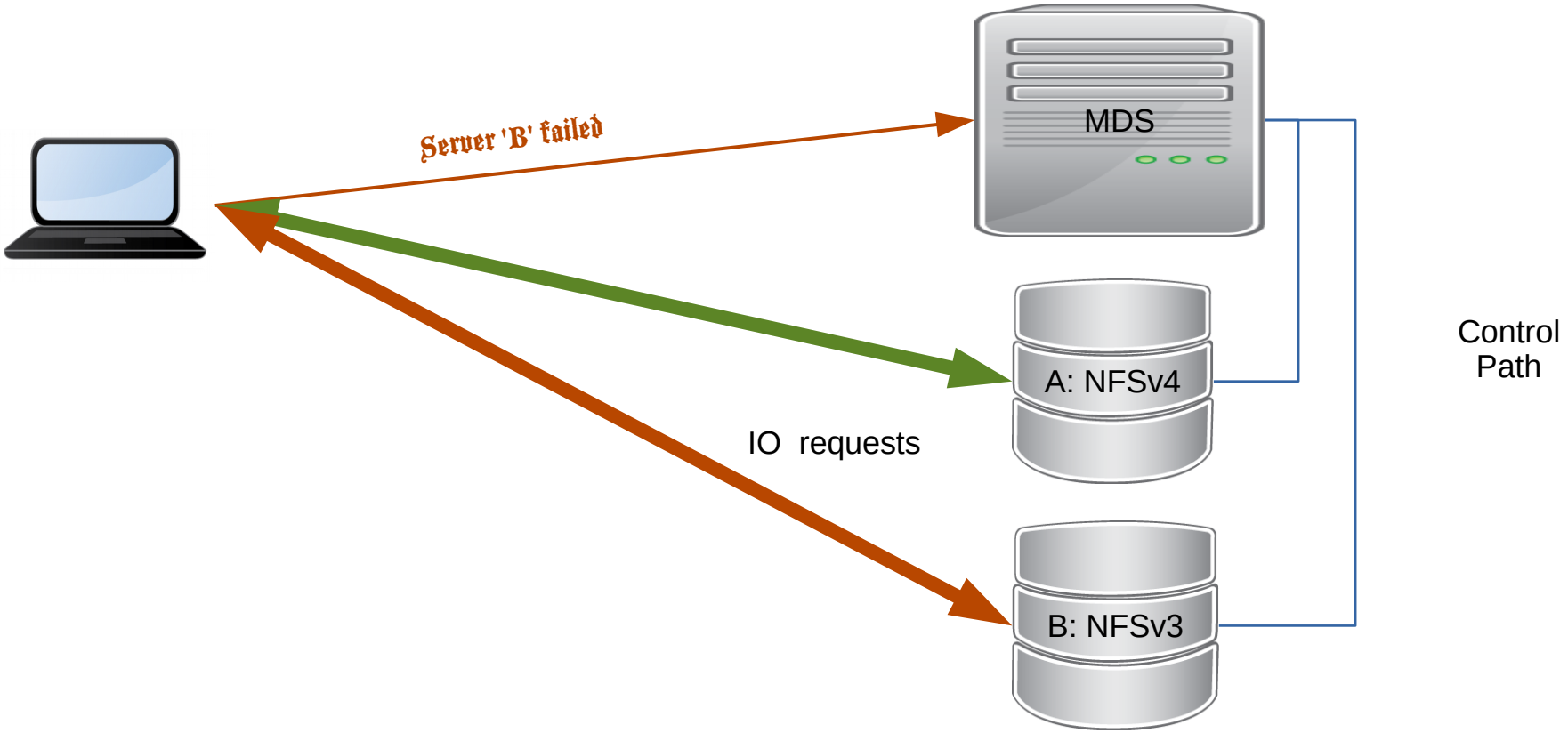
Flexfiles layout

- Business model: consolidate existing servers



Flexfiles layout

- Business model: consolidate existing servers

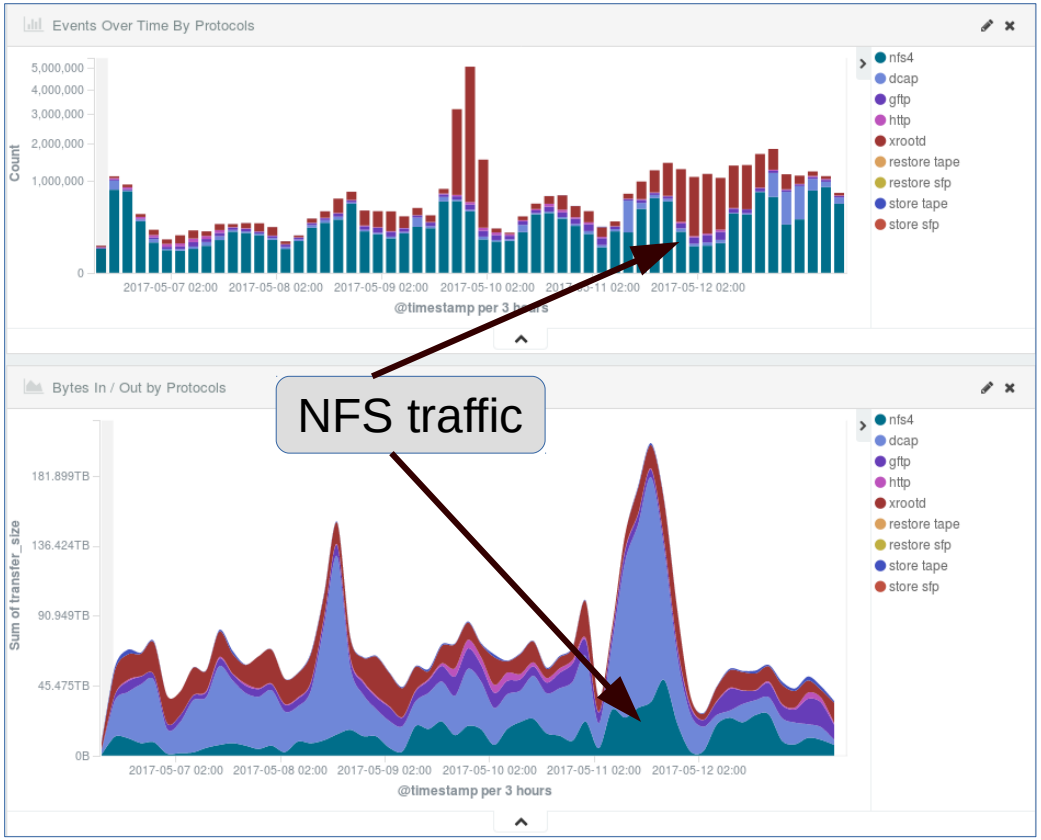


Flexfiles status

- In upstream kernel
  - supported by RHEL 7.2
- Provided dCache 3.0
- DataSphere from PrimaryData
  - like a ‘nfs only dCache with nfs3 servers instead of pools’

Summary

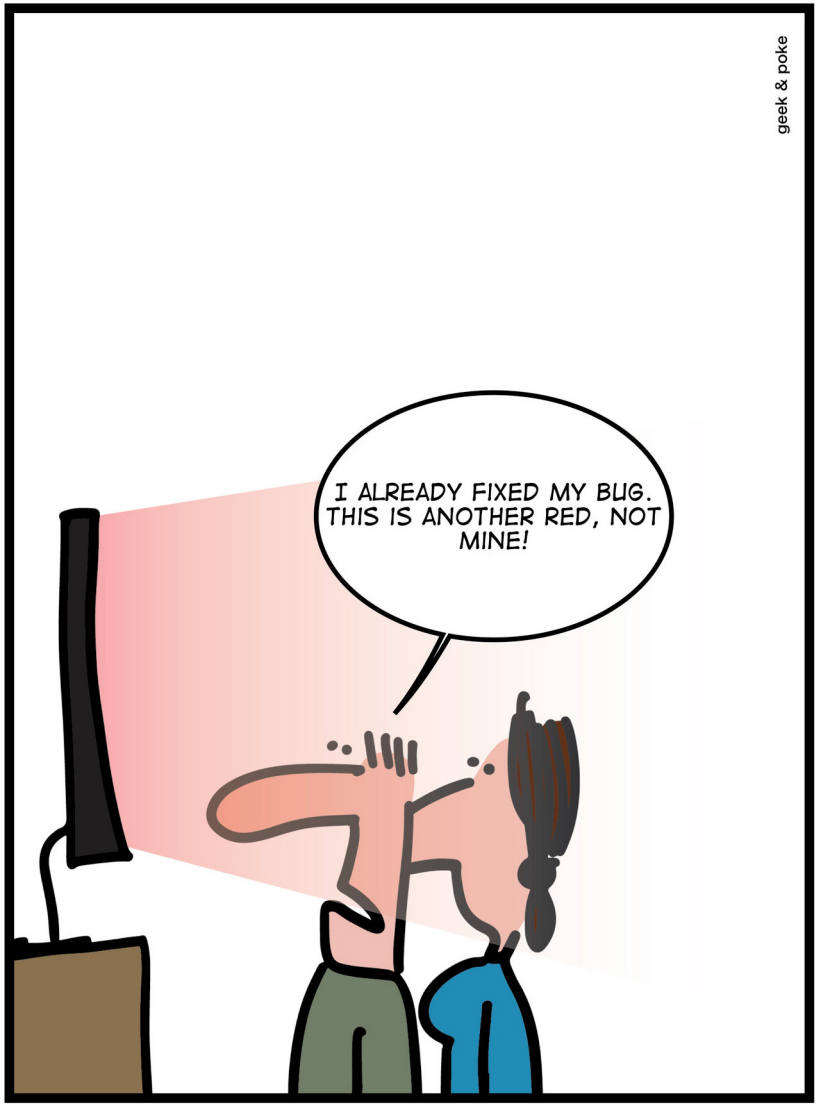
- DESY heavily relays on NFS
  - Local and interactive users
  - xxxCloud
  - XFEL
  - Photon-science
- NFS is an active evolving protocol
  - new requirements
  - new functionality
  - new players
- dCache.ORG is involved in all phases of development
  - protocol definition
  - server development
  - client testing



Lightning talk

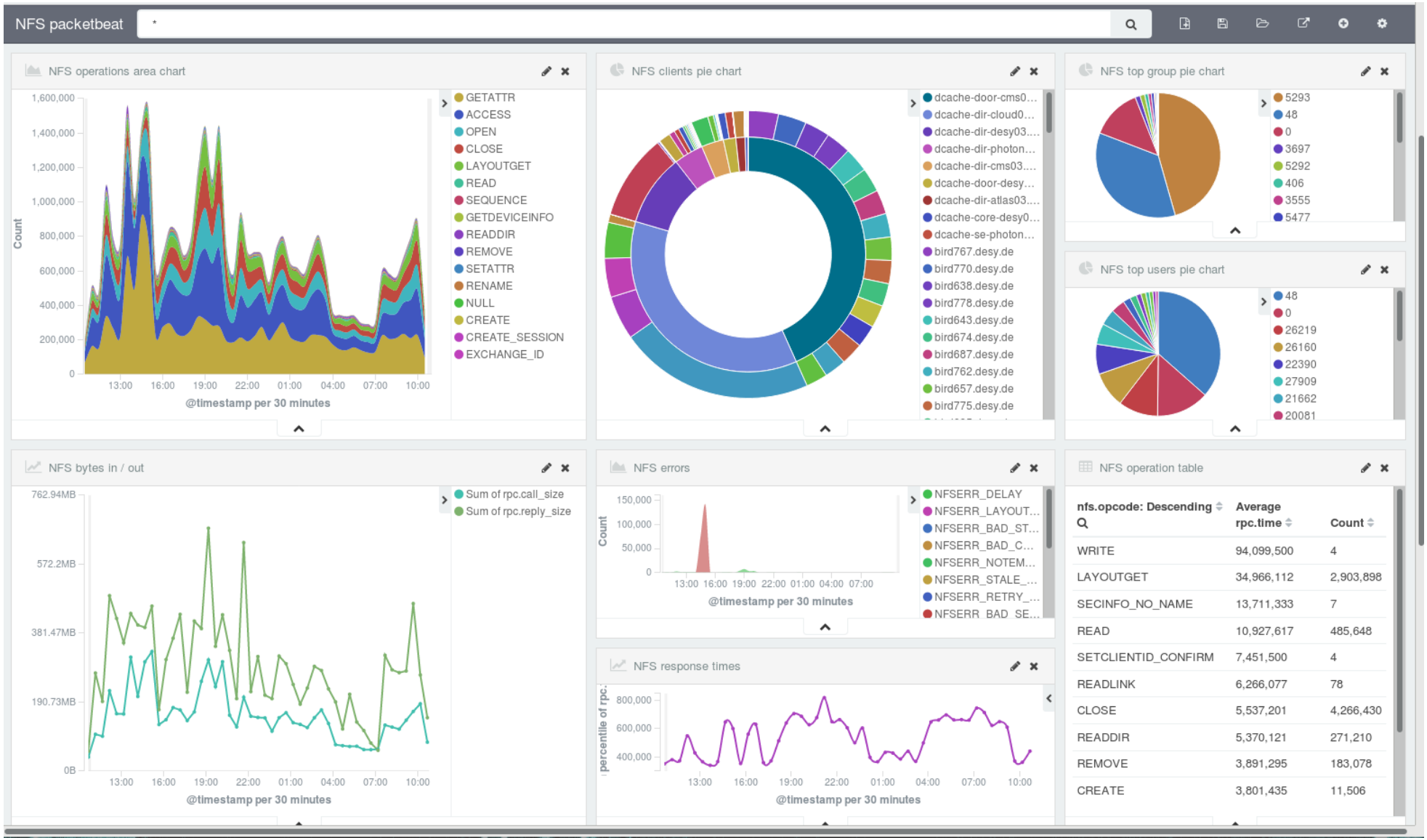
(Monitoring NFS in production)

NFS in production

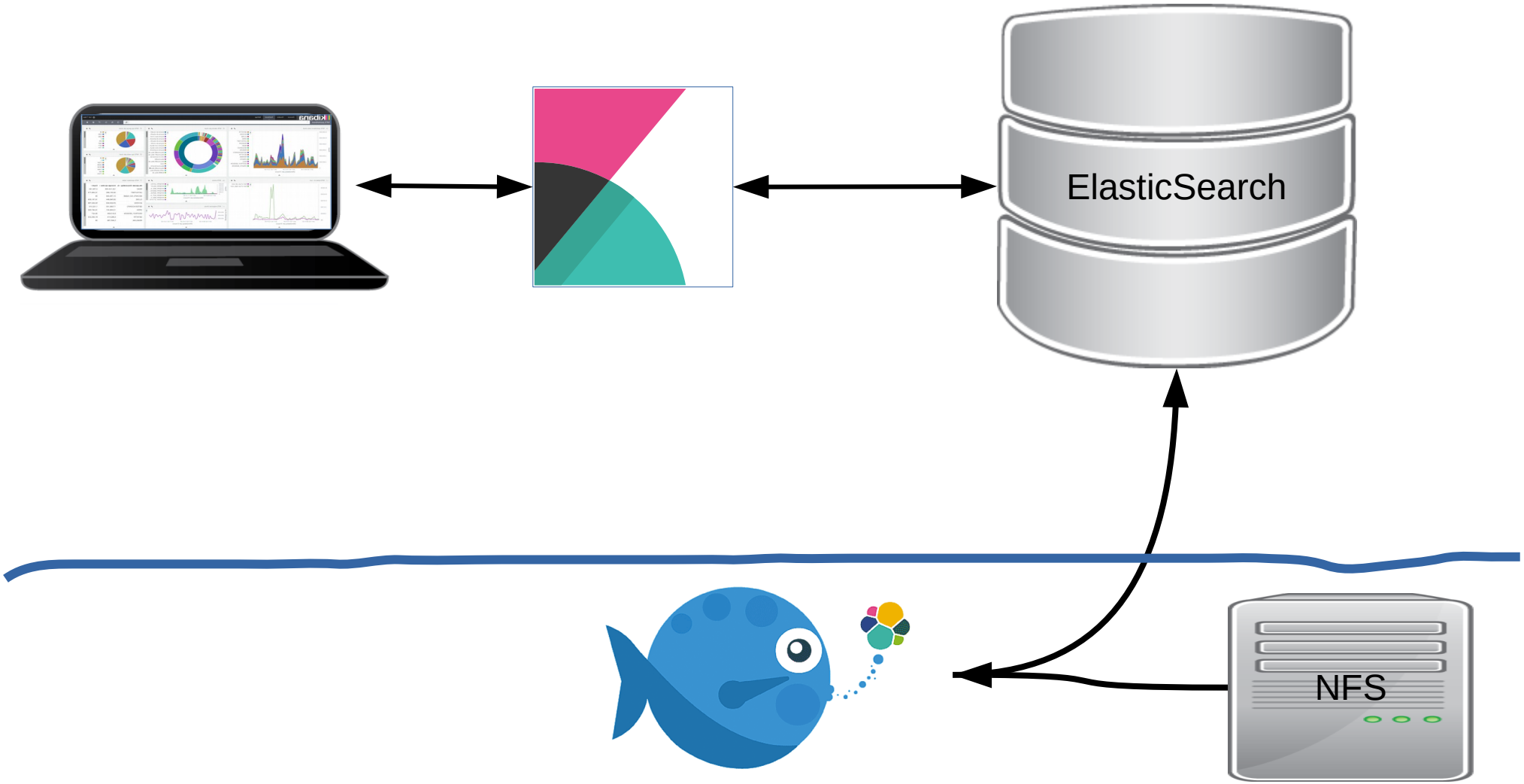


FIFTY SHADES OF RED

NFS traffic visualization



How it works?



Packetbeat

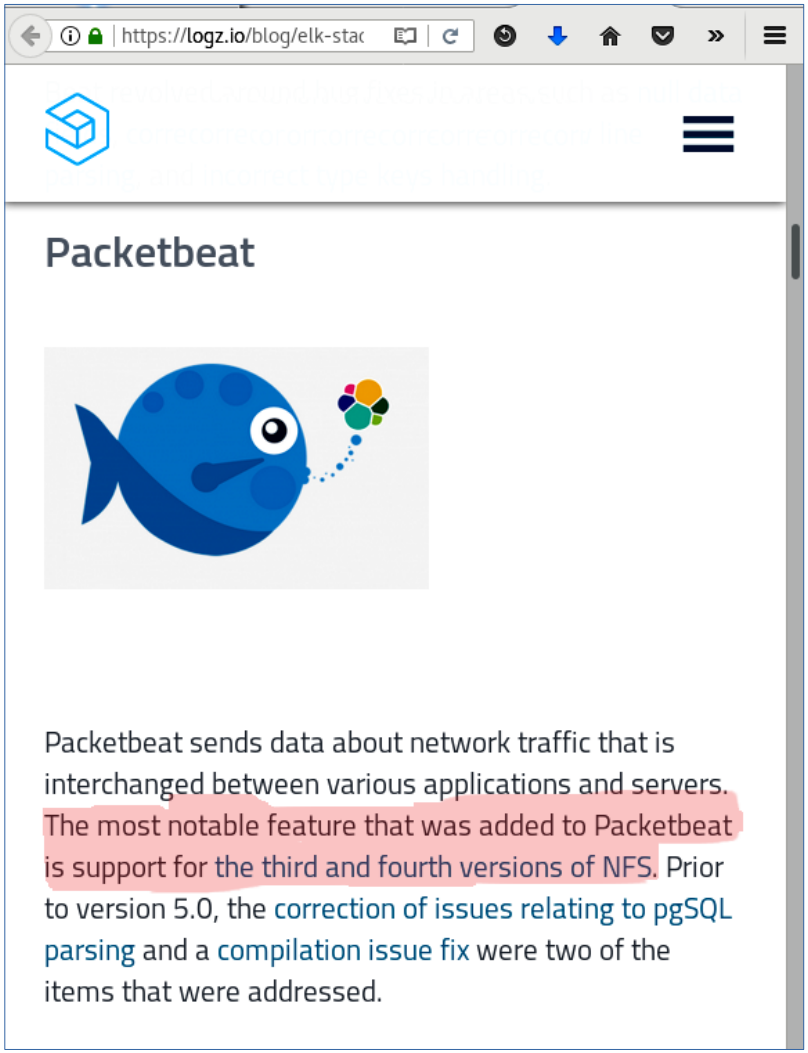
- Lightweight Shipper for Network Data
  - Ship to Elasticsearch or Logstash
- Based on libpcap
- Understands bunch of protocols
  - DNS
  - ICMP
  - HTTP
- Can be extended with your own protocol

NFS beat

- Developed at DESY for NFS traffic visualization
- Added to packetbeat base functionality
- Understands NFS v3, 4.0, 4.1
- Not dCache specific
  - any nfs traffic will work
  - can be configured to monitor appliances with port replication

Where to get it?

- Part of packetbeat-5.0
- Ready-to-use dashboard
- Available from elastic.co download page
- Ready-to-use docker container



Thank You!

to be continued…

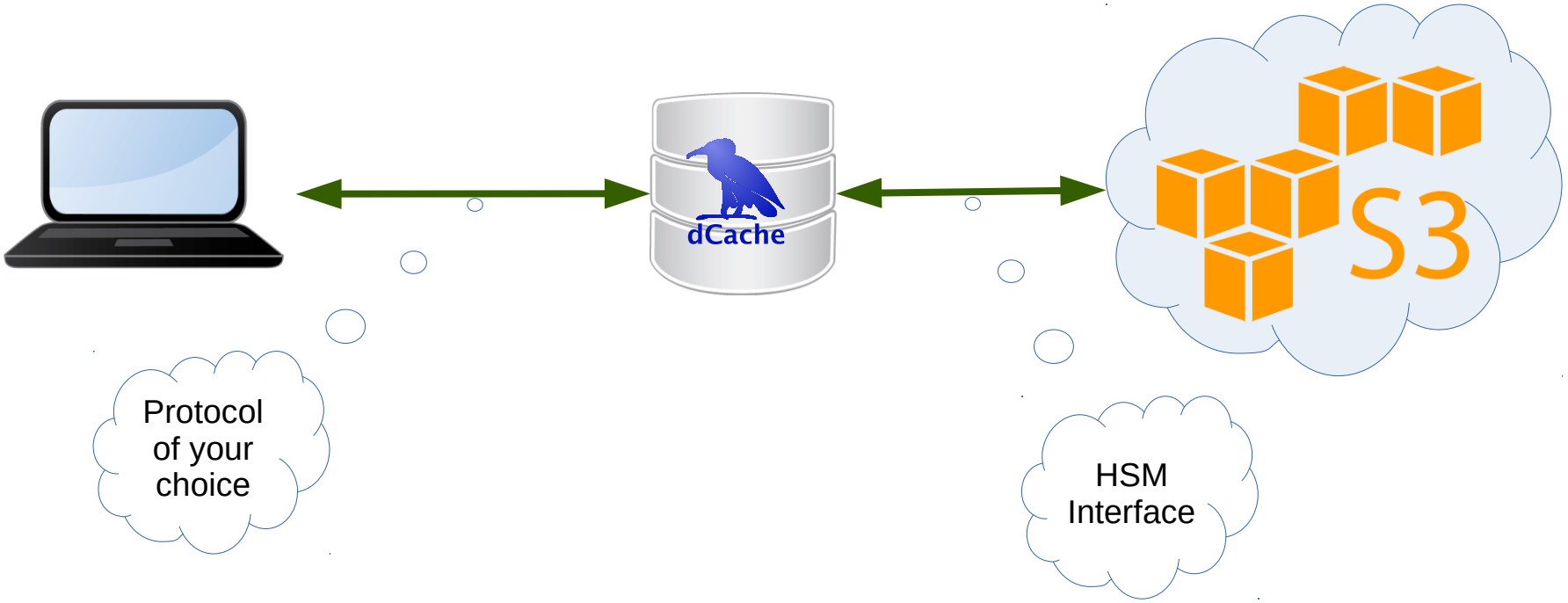
Lightning talk
(fun with dCache)

"Our all-flash systems can be connected to the cloud and provide low-latency access to the hot data and unlimited cheap storage."

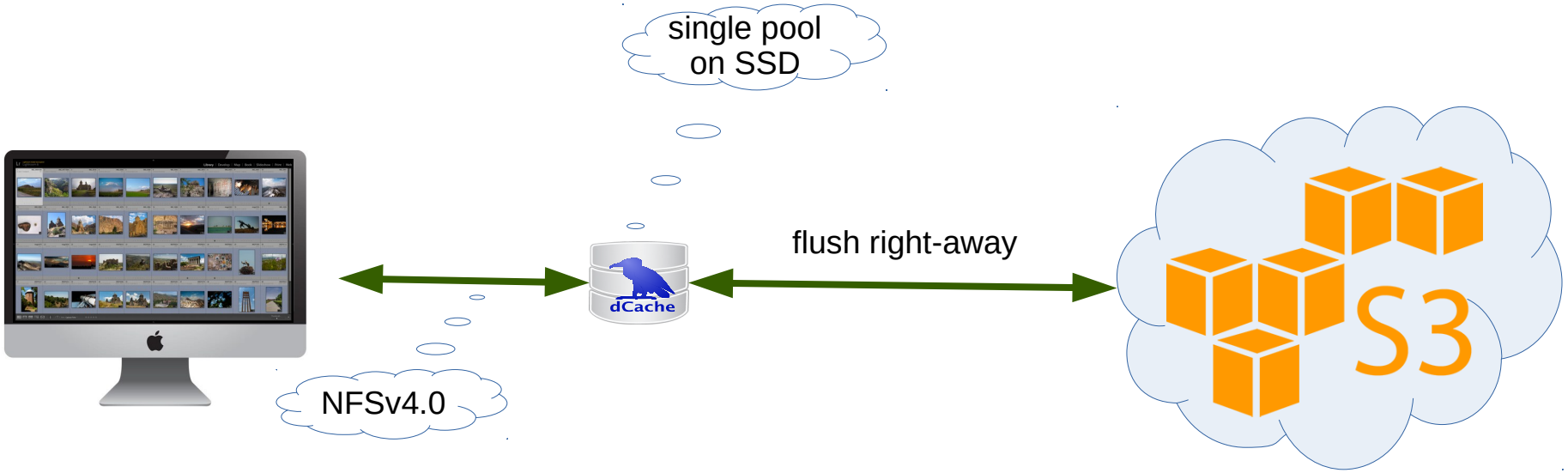
Vendor X.

Sounds familiar?

All-flash dCache



All-flash dCache@HOME



s3hsm

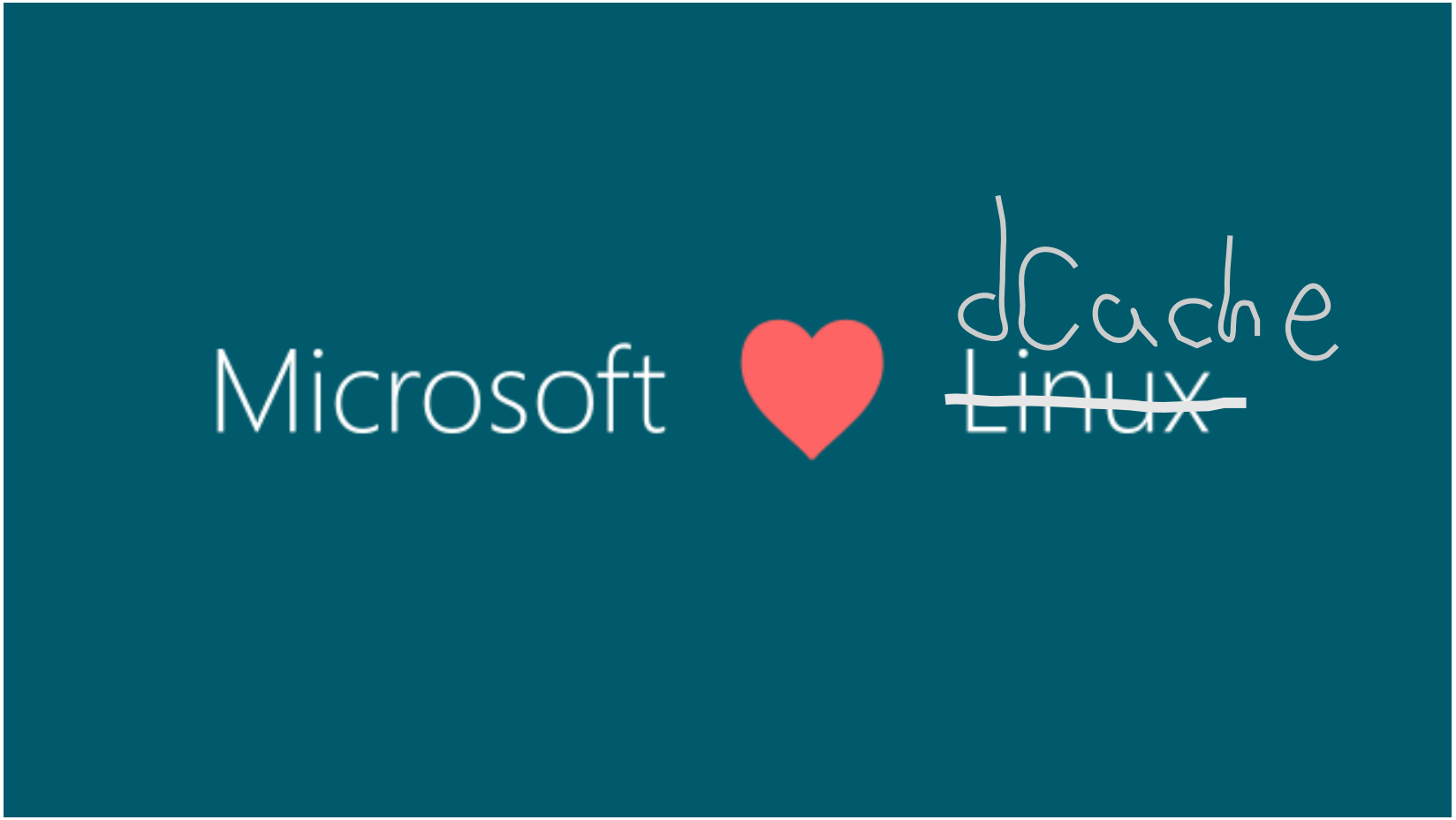
- Developed to store private data in s3
  - Each file encrypted with a unique key
- implements HSM script interface
- Keys stored as a part of HSM location
- Available on the GitHub

s3hsm

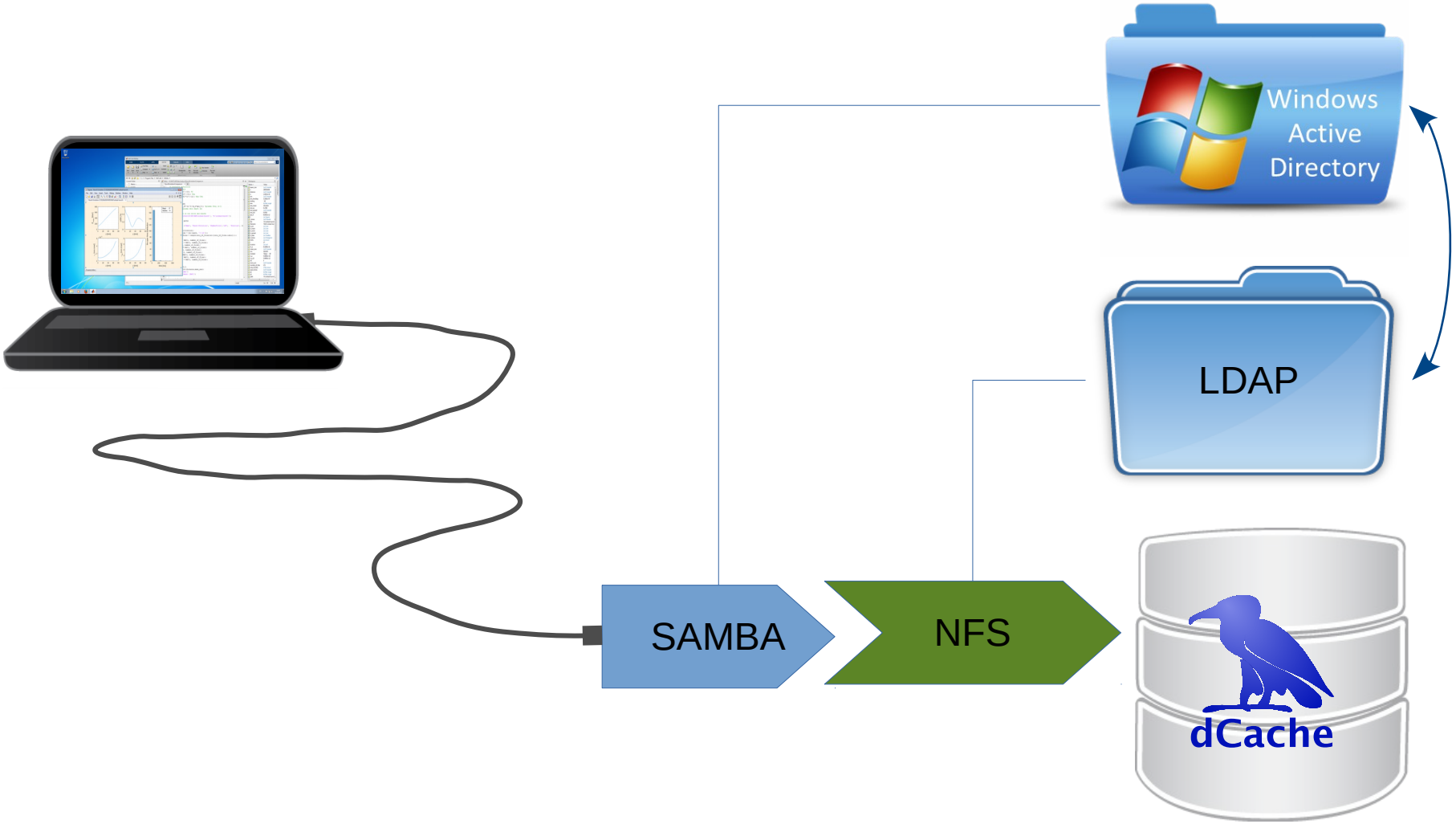
```
$ s3hsm put 0000000635D5968A4DD89E29C242185B2D82 \
      /pool-A/data/0000000635D5968A4DD89E29C242185B2D82 \
      -s3bucket=data -s3config=s3config.yml

osm://amazon-s3/data/0000000635D5968A4DD89E29C242185B2D82?
      ekey=2469f14e083e6b0e3914dc59537cafed1bff176b9a7f99d6d04b28549fdc9a7f
      &etype=aes
$
```

Lightning talk
(making new friends)


+1 Add Friend

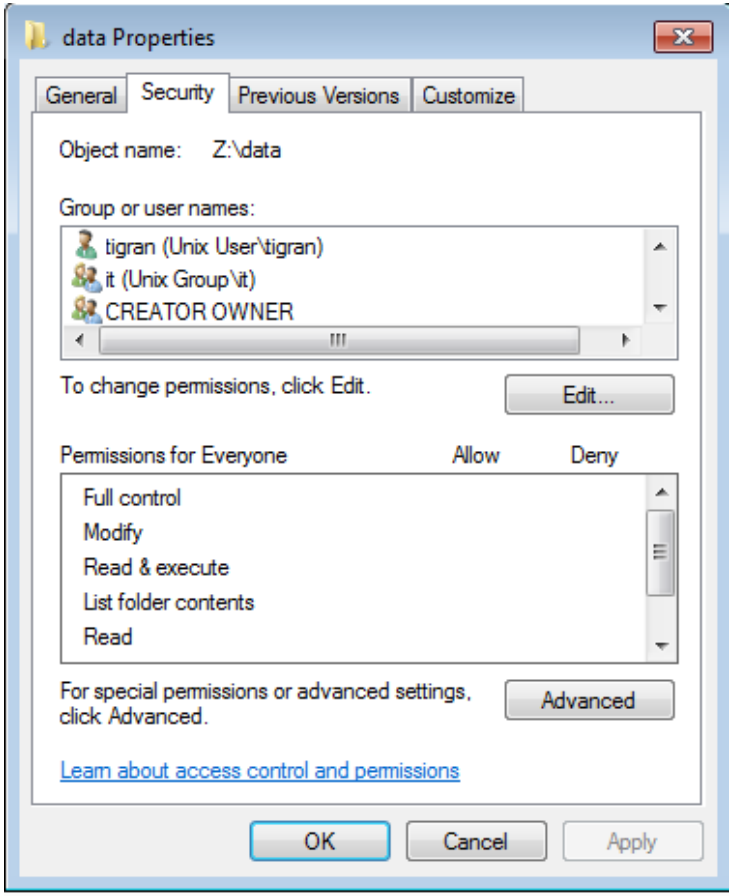

Microsoft
dCache
~~Linux~~

dCache+SAMBA



UNIX <=> Windows mapping

- Host running samba configured to use LDAP
  - no user login allowed!
- Samba as domain member
- Use tdb2 as user back-end
  - custom script for mapping
  - provides UID/GID <=> SID



Config files and more

https://github.com/dCache/dcache/wiki/Exporting-dCache-with-SAMBA

Thank You!