UPDATE ON SECURE LUSTRE

James Beal, Principal Systems Administrator
Pavlos Antoniou, Principal Software Developer
V29 20/05/2021
AGENDA

INTRODUCTION
IMPLEMENTATION
ENABLING RESEARCH
CONCLUSIONS
INTRODUCTION
RESEARCH

The Sanger Centre was established in 1992.

The first draft of the human genome was announced in 2000 and the Sanger Centre was the largest single contributor.

The 1000 Genome project was started in 2008 and UK10K was started in 2010.

To celebrate 25 years, 25 species were sequenced in 2018.

In 2019 the institute was a founding member of the Darwin Tree of Life project, to sequence all known eukaryotes, and the institute plans to sequence all eukaryotes from UK and Ireland (60,000 species)
The Institute is a founding member of the COVID-19 Genomics UK consortium. The consortium has two inter-related tracks of activity.

The first is research on viral transmission and viral variants.

The second is the acquisition of positive samples from people with COVID-19 and the generation of viral genome data of high quality.

The Sanger Institute is providing a centralised service for large-scale genome sequencing of samples from the ‘Lighthouse Lab’ National Testing Centres and from other diagnostic services in parts of the UK that are not covered by the COG-UK regional sequencing labs. It also serves as a backup to take pressure off the regional sequencing labs during periods of high demand.

https://www.cogconsortium.uk/
https://www.sanger.ac.uk/collaboration/covid-19-genomics-uk-cog-uk-consortium
The Wellcome Sanger Institute has:

- 14,000 VCPU of compute in OpenStack
- 20,000 cores of compute using LSF
- 25 Petabytes of Lustre
- 5 Petabytes of Ceph (S3 and Block), triple replicated
- 14 Petabytes of data in iRODS, dual replicated
The institute is processing more personally identifiable data than it has in the past. It is quite common for data to be released to groups with a data access agreement that requires that access to data is only granted to specific users.

The purpose of our OpenStack is to give our developers the flexibility to create systems which solve problems for their researchers.

Users have root in OpenStack and our internal systems need to be protected from those instances. The institute has more data which is personally identifiable and this means that governance of data is becoming critical.

Before Secure Lustre we did not have a performant POSIX filesystem available inside OpenStack. Considerable effort was spent staging data to and from OpenStack.
This talk is an update to a talk given in June 2017 where we introduced using Lustre routers to allow access to Lustre from OpenStack.

A whitepaper was written and this has also been updated to show how to allow access to Lustre using only layer 3 switches.

https://hpc-news.sanger.ac.uk/
OpenStack is an open source system which allows users to create and manage virtual machines and their associated infrastructure using both APIs and a web interface.

The Sanger institute has two main OpenStack deployments:

A general purpose OpenStack which is based on a Kolla “Train” release with support from StackHPC. With approximately 10,000 threads, and 115TB of RAM, with access to a 5PB usable Ceph cluster providing block (Cinder) and S3-compatible (radosgw) access.

A dedicated production sequencing (currently used for Covid and UK Biobank) OpenStack, which is based on the upstream “Queens” release. With approximately 3,500 threads, and 36TB of RAM, with access to a 70TB usable Ceph cluster providing block (Cinder). Main storage is provided by a redundant pair of 3.1PB Lustre systems.
Using Secure Lustre

From a user's perspective, they boot a provided image in a prescribed way and they then have access to their Lustre share.

As an admin, a share is created by creating a nodemap entry; we have script to simplify this.

Networks and subnets are created for each supported project.

As an admin, a project is granted access to a share by adding an RBAC entry to allow access to the network.

ACL’s on the top of rack switches ensure that packets from each project’s allocated IP address range can only arrive at the correct Lustre server IP address – others are blocked.

OpenStack images are created via a continuous integration system.
LNET SPACE

Completely separate network spaces.
LNet protocol currently limits to 32 spaces
Exascaler limits to 17 spaces.
Originally mainly used for InfiniBand.
Ethernet implementation
    iproute2 on Lustre servers
    VRF could be used on layer 3 switches
Our secure Lustre implementation could use a single LNet space, multiple are used to clarify the configuration.
Two features are fundamental to secure Lustre

Subdirectory mounts: force a client to mount a subdirectory as the root of the file system.

Identity mapping: 1:1 mapping of mapped users and groups to canonical users and groups.

A separate nodemap is required per share. The Institute has over 150 projects, CERN have over 4,000 projects.

Each request is checked against a list of ranges to determine which nodemap that it should have applied.
The ubuntu user in OpenStack is mapped to isgbot in HPC

The ubuntu group in OpenStack is mapped to ssg-isg in HPC

jb23 is a local user with the same uid number in OpenStack as the jb23 user in HPC
Should the whole file system be shared with traditional HPC?

Divisional support teams can have a share higher up the filesystem tree.

If the shared directory is owned by a mapped user or group then all the data can be deleted by any instance in OpenStack.
### Example Nodemaps

<table>
<thead>
<tr>
<th>Nodemap Name</th>
<th>Range Start</th>
<th>Range End</th>
<th>Fileset</th>
<th>Idmap uid</th>
<th>Idmap gid</th>
<th>Squash uid</th>
<th>Squash gid</th>
</tr>
</thead>
<tbody>
<tr>
<td>lustre_casm01</td>
<td>10.77.96.0@tcp1</td>
<td>10.77.97.255@tcp1</td>
<td>/casm/test1</td>
<td>1000:2325</td>
<td>1000:1323</td>
<td>2325</td>
<td>1323</td>
</tr>
<tr>
<td>lustre_hgi01</td>
<td>10.77.0.0@tcp3</td>
<td>10.77.1.255@tcp3</td>
<td>/hgi</td>
<td>1000:13245</td>
<td>1770:1770</td>
<td>13245</td>
<td>65534</td>
</tr>
<tr>
<td>lustre_hgi02</td>
<td>10.77.2.0@tcp3</td>
<td>10.77.3.255@tcp3</td>
<td>/hgi/project1</td>
<td>1000:1535</td>
<td>1000:1770</td>
<td>1535</td>
<td>1770</td>
</tr>
<tr>
<td>lustre_hgi03</td>
<td>10.77.4.0@tcp3</td>
<td>10.77.5.255@tcp3</td>
<td>/hgi/project2</td>
<td>1000:1793</td>
<td>1000:24003</td>
<td>1793</td>
<td>24003</td>
</tr>
<tr>
<td>lustre_manops01</td>
<td>10.77.128.0@tcp2</td>
<td>10.77.129.255@tcp2</td>
<td>/admin/test1</td>
<td>1000:17234</td>
<td>1000:18333</td>
<td>17234</td>
<td>18333</td>
</tr>
<tr>
<td>lustre_manops02</td>
<td>10.77.130.0@tcp2</td>
<td>10.77.131.255@tcp2</td>
<td>/test2</td>
<td>1000:24024</td>
<td>1000:19424</td>
<td>24024</td>
<td>19424</td>
</tr>
</tbody>
</table>
The scientific network is built from Arista Switches. Multiple 100 Gigabit/second links are used between switches and for high bandwidth systems.

Our OpenStack systems use a leaf and spine topology while our traditional batch systems have top of rack switches connected to a traditional core.

All devices in the data path have redundant connections.

Hypervisors and computation nodes are connected with active/active 25 Gigabits/second connections.
Each secure Lustre client has two interfaces

1. Standard self service network
2. Dedicated storage network

In this model all clients are connected to the Lustre top of rack router, which has all the ACL’s required to ensure that only access is correctly granted.
Here we show an OpenStack network router which is responsible for standard OpenStack network access.

The provider router is implemented as a distributed router on each of the top of rack switches that hypervisors are connected to.

BGP is used as our interior routing protocol and host routes are promoted to ensure that packets do not trombone.
Client IP addresses are associated with OpenStack projects.

As a Lustre filesystem could be used by any division we add an ACL per division per Lustre filesystem to ensure that only that division’s network has access to the appropriate IP address of the Lnet interface.

We allocated 16 networks each of /23 in to a supernet of /19.

We allow all established and icmp traffic.

TCAM which is used to implement ACL’s on switches is a scarce resource. Our top of rack switches Arista 7060 have a relatively small amount of TCAM and we are using 7% of one bank of TCAM with this ACL for 5 divisions (Lnets).

```
ip access-list secure-lustre
  10 remark Established traffic
  20 permit tcp 10.177.0.0/17 10.160.32.0/19 established
  30 remark ICMP Traffic
  40 permit icmp any any
  50 remark Human Genetics
  60 permit tcp 10.177.0.0/19 10.160.32.96/27 eq 988
  70 permit tcp 10.177.0.0/19 10.177.252.96/27 eq 988
  80 remark Human Genetics Deny
  90 deny ip 10.177.0.0/19 any
 100 remark Tree Of Life
 110 permit tcp 10.177.32.0/19 10.160.32.128/27 eq 988
 120 permit tcp 10.177.32.0/19 10.177.252.128/27 eq 988
130 remark Tree Of Life Deny
140 deny ip 10.177.32.0/19 any
...
 300 remark Catchall Deny
 310 deny ip any any log
```
**OPEN VSWITCH**

Open vSwitch connects each of the software NICs and hardware NICs together and ensures packets flow from one to another.

The diagram to the right shows that a Lustre packet traverses two Linux bridges and two OVS switches.

Each bridge and switch introduces latency.
We use Packer to create our images that we provide to our internal customers.

The virtual machine inspects network routes on boot, and configures LNet appropriately.

The machine Inet pings each potential server and attempts to mount those that reply.

Users base their images on ours.

https://github.com/wtsi-ssg/simple-image-builder/tree/lustre
CREATING INSTANCES

The standard OpenStack VM creation command line does not allow for a machine to be created with two ports, one with standard security groups and one with security disabled. We have an example bash script which we provide which first creates both ports and then creates a new machine with both ports attached.

We have created terraform systems to create machines which are then used as kubernetes machines.

Our pipeline runner wr has support for secure Lustre.

https://github.com/HelenCousins/createserver
https://github.com/VertebrateResequencing/wr
Later versions of OpenStack have made the configuration significantly simpler.

Firstly we create all the networks and subnets as an admin in advance per division.

```
openstack network create --provider-network-type vlan --provider-physical-network physnet1
    --provider-segment 75 --no-share --internal --mtu 9000 --disable-port-security lustre-npg01

openstack subnet create --dhcp --host-route destination=10.177.252.24/27,gateway=10.177.64.1
    --network lustre-npg01 --allocation-pool start=10.177.64.10,end=10.177.65.240
    --subnet-range 10.177.64.0/23 lustre-npg01

Then when we wish to grant access to a share we as an admin create an rbac entry

openstack network rbac create --target-project npg-es4a --action access_as_shared
    --type network lustre-npg01
LUSTRE CONFIGURATION

Adding shares piecemeal is prone to errors.

Multiple secure Lustres require common configuration.

There is an issue **LU-14657** which means that once the nodemap is created then if the fileset and idmapping are set before the configuration is stable on all the Lustre targets then the nodemap configuration is inconsistent on the servers and stays inconsistent until the nodemap is recreated.

We have a simple set of scripts which allow the Lustre servers to be configured consistently, this also allows us to store our configuration in git.

We ask that all nodes using a nodemap unmount the share before we make any changes to the nodemap.

https://github.com/wtsi-ssg/secure-lustre
A simple dd test was used as this gave the most consistent results.

```bash
#!/bin/bash
rm -f deleteme
echo 3 > /proc/sys/vm/drop_caches
dd if=/dev/zero bs=1M count=4194304 status=progress conv=fdatasync of=deleteme
echo 3 > /proc/sys/vm/drop_caches
dd if=deleteme bs=1M status=progress of=/dev/null
```
Tests were performed against lus23 our latest research filesystem. It is composed of:

- 3 ES7990 each a SS9012 enclosure
  - 2 OSS per ES7990 each with dual active/active 100GBits/second Ethernet.
  - 164 NL-SAS.
- 1 SFA200NV
  - 10 NVMe.
  - 8 active/active fibre channel (FC8).
- 2 Dell R640
  - Single Intel Xeon Platinum 8260, 192GB RAM.
  - Dual active/active 100GBits/second Ethernet.
Here we compare performance between a full hypervisor instance and a small 4 VCPU instance.

A simple test with two collocated servers showed an increase in performance.
Here we compare performance between a full hypervisor instance accessing secure Lustre and equivalent physical host accessing the system with secure Lustre.

We believe that the performance drop off can be attributed to the overhead of the OVS/Linux bridge.
Here we compare a physical server first configured traditionally and then reconfigured with access to a secure Lustre VLan.

There is no significant performance difference.
ENABLING RESEARCH
Human Genetics Research

Projects

• Large cohorts of whole genome and exome sequencing
• Joint calls of local Sanger cohorts in combination with large external cohort frequencies like UKBB and gnomad
• QC -> Statistical analysis -> GWAS -> Phenotypes - Association analysis
• Identify new genes and variants likely to cause disease from association analysis

1. Interval

Whole Genome Sequencing
12,354 samples whole-genome sequencing
24 joined call VCF files
Tasks:
- Run QC pipeline for each chromosome
- Save filtered matrix tables after variant and sample QC
- Combine QC matrix tables to one WGS matrix table
- Run gwas analysis on the wgs matrix table
- Produce association analysis plots for each phenotype

2. MegaWES

Exome Sequencing
93600 samples
13651809 variants

Sample and Variant QC following established gnomad pipeline from Broad institute

- Remove samples fail QC
- Remove variants fail QC
- Run association analysis
HAIL

- Developed at the Broad Institute

Scalable
  software for genomic analysis
  can run on a laptop, on a cluster, on the cloud

Python computing
  Library exposed through python with a spark backend
  Jupyter notebook or python programming scripts
  Distributed computing of datasets with matrixtable data structures

- Allows statistical analysis of thousands of VCF samples
- Uses matrixtables as main data structure
- OpenStack Spark Cluster with python and hail installed

Multi-sample VCF as Hail matrixtable Structure

MatrixTable

https://hail.is/
https://github.com/wtsi-hgi/osdataproc
WHY HAIL

Hail as a scientific computing stack

- Read and write common formats
  - VCF
  - TSV
- Filter, group, aggregate
  - BGEN
  - PLINK
- Annotation
  - JSON
  - GEN
- Visualization
  - BED
  - GTF
WHY HAIL

Hail as a scientific computing stack

Data slinging

- Read and write common formats
- Filter, group, aggregate
- Annotation
- Visualization

Analytical toolbox

- Compute AF stratified by all combinations of (sub-)population and sex
- Counting number of loss-of-function alleles per sample per gene
# WHY HAIL

Hail as a scientific computing stack

<table>
<thead>
<tr>
<th>Data slinging</th>
<th>Analytical toolbox</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Read and write common formats</td>
<td>• Built-in wrappers for VEP, Nirvana</td>
</tr>
<tr>
<td>• Filter, group, aggregate</td>
<td>• Join with annotations by variant, locus, interval, gene</td>
</tr>
<tr>
<td>• Annotation</td>
<td>• ReferenceGenome is a first-class concept, for all our sanity</td>
</tr>
<tr>
<td>• Visualization</td>
<td></td>
</tr>
</tbody>
</table>
WHY HAIL

Hail as a scientific computing stack

- Data slinging
  - Read and write common formats
  - Filter, group, aggregate
  - Annotation
  - Visualization
WHY HAIL

Hail as a scientific computing stack

- Data slinging
  - Analytical toolbox
    - Statistical methods for genetics
    - Scalable linear algebra
PREVIOUS HAIL SETTING
SECURE LUSTRE CONFIGURATION

• ISG has been developing solution for directly accessing secure lustre from OpenStack

• Users can create virtual machines in OpenStack that read and write to the high performance POSIX lustre filesystem

• Data from Sanger groups held in lustre for over 13 years. 13 PB of space

• Different tenants/users in OpenStack allow security of data

• Different teams and groups exist in /lustre
TESTING SCRIPT 1: INTERVAL WGS

- Sample and Variant QC pipeline for chrom 1
- **FAIL** to complete on initial hail configuration in OpenStack with nfs volume
- Migrated pipeline to google cloud
- 2 hours to complete pipeline for chr1 on Google Cloud
- **SUCCESS** hdfs in OpenStack: reading and writing to hdfs in 3 hours
- Data transfer overhead:
  - Copying results from hail volume to the farm to share with team.
  - 5 TB matrixtable for WGS takes more than 24 hours to copy from volume to the farm
  - Fastest and most reliable way I have found:
    - sync matrixtable from hail volume to s3 first and then copy from s3 to farm with **rclone**

Manhattan plot showing a chr4 region of significant SNPs
TESTING SCRIPT 2: MEGAWES

• Writing out final matrixtable after RF and exporting a VCF of all samples & variants (~93600 samples with 13651809 variants)

• OpenStack cluster 50 m2.medium workers:

  • **Matrixtable**: writing out the matrixtable to hdfs FAILS with out of memory issues

  • **VCF**: Exporting the VCF is possible in parallel mode (without saving the final matrixtable) which means that a repeat of its creation script next time it is required

• Additional time to copy data to farm for team to access

Ancestry analysis in MegaWES project
## RESULTS

<table>
<thead>
<tr>
<th>Cluster properties:</th>
<th>50 m2.medium 50 TB volume</th>
<th>50 m2.medium 50 TB volume</th>
<th>128 n1-standard-8 Google cloud storage europe-west2-a</th>
<th>20 m2.medium 100 GB volume</th>
<th>50 m2.medium 100 GB volume</th>
<th>100 m2.medium 100 GB volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERVAL WGS QC – chr1</td>
<td>✗  FAIL</td>
<td>✓ 3 hours</td>
<td>✓ 2 hours</td>
<td>✓ 6 hours</td>
<td>✓ 3 hours &amp; 20 min</td>
<td>✓ 1 hour &amp; 50 min</td>
</tr>
<tr>
<td>MegaWES matrixtable write and export to VCF</td>
<td>N/A</td>
<td>✗  FAIL</td>
<td>N/A</td>
<td>✓ 3 hours mt 4 hours VCF</td>
<td>✓ 2 hours mt 3 hours VCF</td>
<td>N/A</td>
</tr>
<tr>
<td>Additional overhead to copy data/results from/to farm</td>
<td>N/A</td>
<td>~6 hours (800GB)</td>
<td>~4 hours (800 GB)</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

m2.medium: 4 CPU - 46 GB RAM
n1-standard-8: 8 CPU - 30 GB RAM
CONCLUSIONS

• Secure Lustre with hail is a perfect match for genomic research
• Secure Lustre eliminates memory constraints writing hail matrixtables
• **Reliable** data storage for reading and writing
• Fast reading and writing files from hail even with modest memory clusters
• More OpenStack resources available for other users
• No longer required to have a huge volume attached to the cluster to save the results, we can write directly to farm
• Secure Lustre allows access to a fast POSIX filesystem from within OpenStack with no additional capital expenditure.
• Data is not isolated to volume and no need to transfer back to farm/S3
ACKNOWLEDGEMENTS

• Wellcome Sanger Institute
  • Peter Clapham
  • Helen Cousins
  • Tim Cutts
  • Christopher Harrison
  • Dave Holland
  • Vivek Iyer
  • Jonathan Nicholson
  • Matthew Vernon

• Data Direct Networks
  • Sébastien Buisson
  • James Coomer
  • Thomas Favre-Bulle
  • Richard Mansfield

• Arista Networking
  • David Murray

• StackHPC
  • Stig Telfer

• Image Credits
  • Simon Binley
  • Mathew Davies
  • Alex Gedny
THANK YOU