



Centralising Elasticsearch

- Project goals, challenges
- Implementation and Service status
- Access control (ACLs)
- Summary



Project goals/mandate

Setup a centralised Elasticsearch service

- New, centrally managed service
- Consolidate existing clusters



Challenges

Consolidation:

- Centralised management
- Resource sharing
- Use of standard hardware
- Use of virtualisation



Expectations:

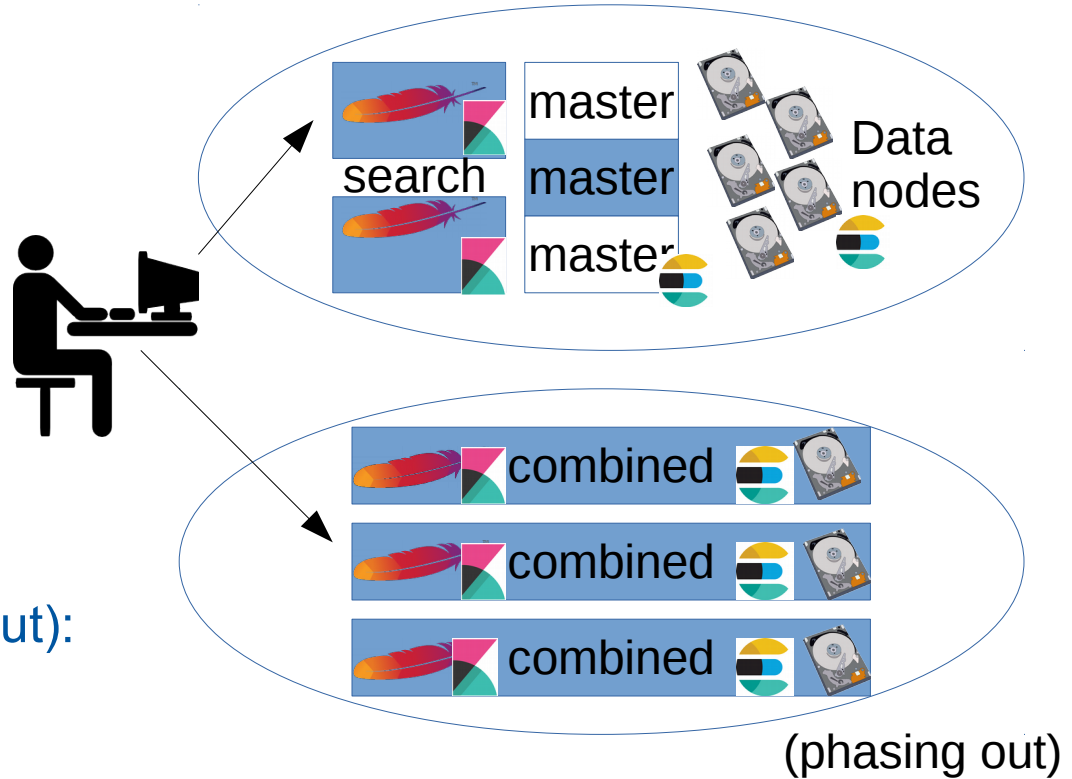
- Special requirements
- Privacy and security
- Performance
- Scalability

Solution

- Share resources where possible
 - Consolidate smaller use cases
 - Put users with similar needs on the same cluster
- Use dedicated clusters where needed
 - Special networking requirements (eg. Technical network (TN) trusted)
 - High demand use cases (eg. CERN IT monitoring)

Node organisation: Models in use

- Made to spare resources
- 4 node types:
 - ES Master nodes (3)
 - Search nodes (2+)
 - Http proxy (apache)
 - Kibana services
 - Data nodes (3+)
 - Combined nodes (phasing out):
 - Master, search and data node
 - For small installations



Hardware

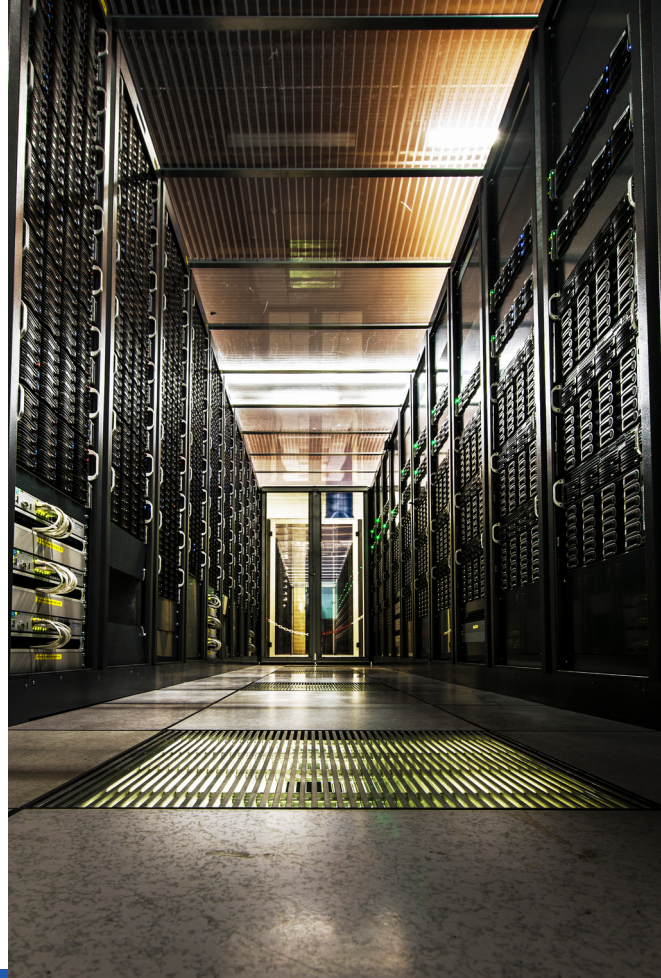
Virtualized hardware openstack™ CLOUD SOFTWARE

- Higher flexibility and better resource consolidation
- Allows to give smaller nodes to small use cases
- Allows for rapid access and replacement
- **116** VMs for search, master and combined nodes

Work horse for data nodes:

- Virtualised data nodes on special hypervisors
- Spinning disks with SSD cache (b-cache)
- ca. 1.2 TB space and 60GB RAM per full node
- Use full or half nodes, **100** nodes in use

Special case: ES for computer security



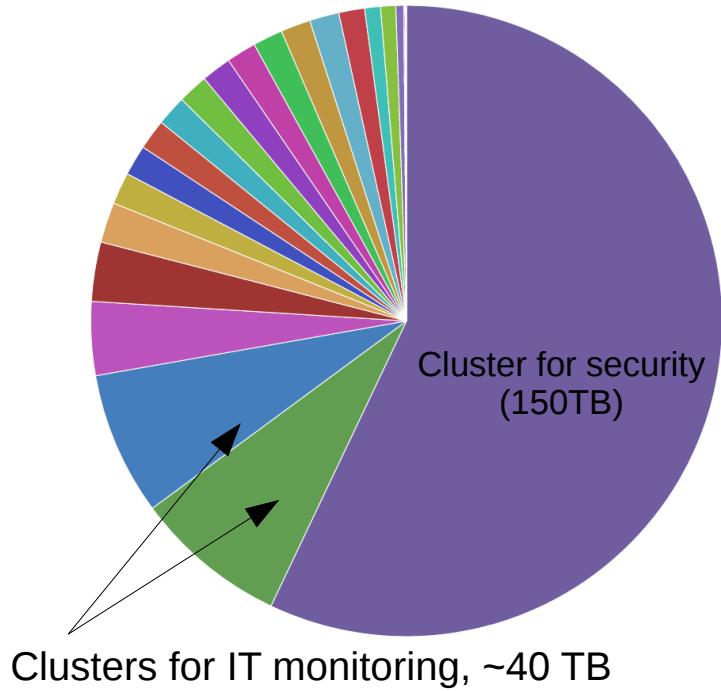
Description	Details	Link	Status
Service status monitoring	SLS	https://cern.service-now.com/service-portal/sls?showall=true	OK
Internal service monitoring	Accesses, errors, disk usage, ...	https://es-perfmon.cern.ch/	OK
Integration into AI infrastructure	Puppet, lemon monitoring, CI, ...		OK
Support structures	Functional Elements, SNOW integration		OK
Documentation	Enduser documentation Service rota person documentation Service manager documentation	https://cern.ch/esdocs https://cern.ch/esops	OK ACLs to be documented
Work flow automation	Removing/adding machines, ES restarts, upgrades, reboots	https://itesrundeck.cern.ch , https://gitlab.cern.ch/it-elasticsearch-project/itestools	OK
Accounting	Aggregation by Accounting group		Per cluster OK, per accounting group ongoing
Index level security	ACL settings on index level		OK, being deployed

Service status: ES clusters

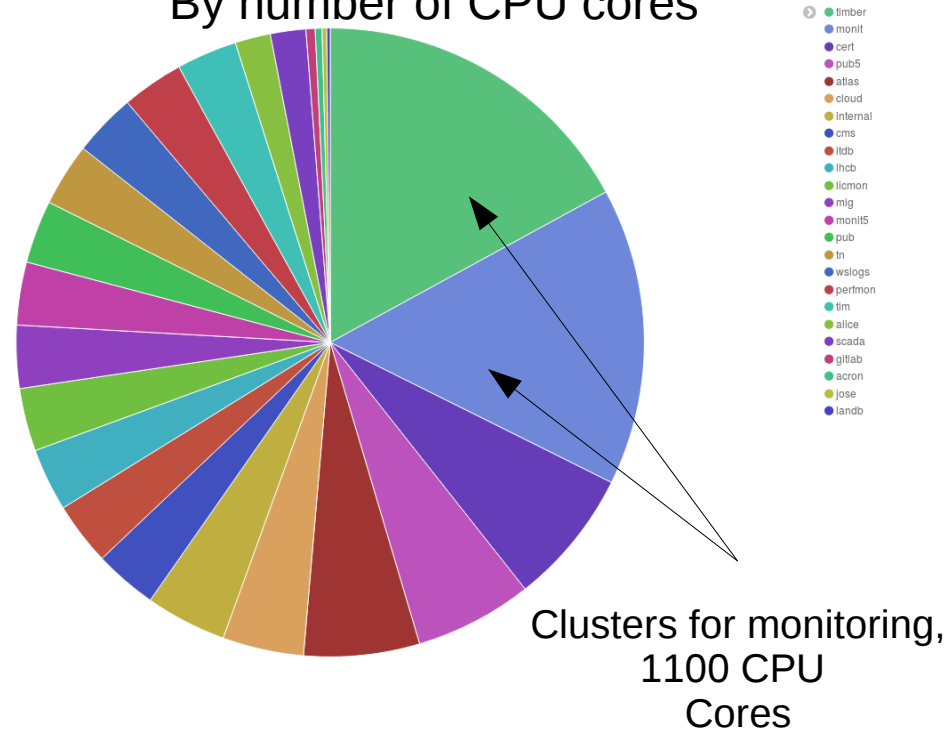
- ~21 Clusters up and running:
 - ~39 use cases (entry-points) supported
 - Currently up to 6 entry points on a single cluster
- Elasticsearch 2.4.1 or 5.2.1
- Kibana 4.6.1 or 5.2.1
- Planning upgrades to 5.2.1 (or newer) with our users

Service status: Current cluster sizes

By storage



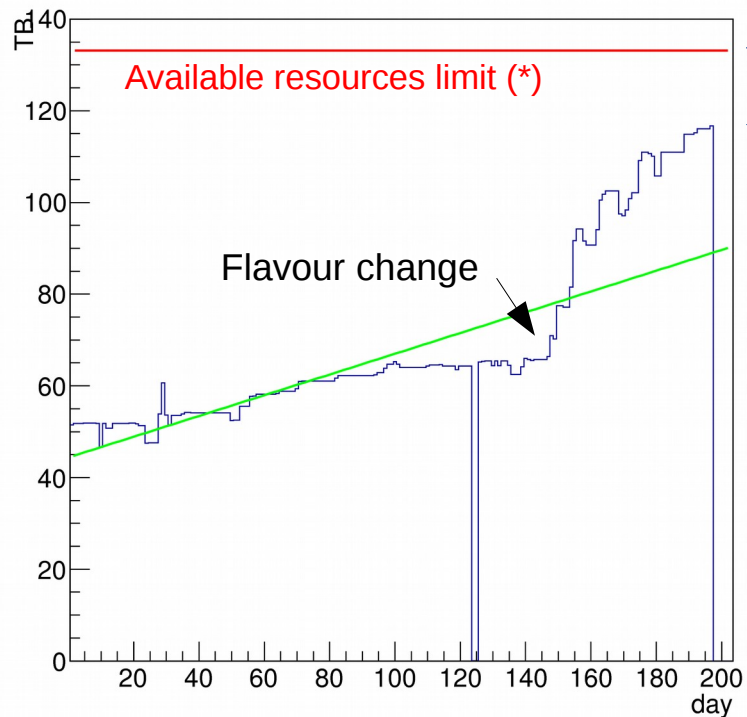
By number of CPU cores



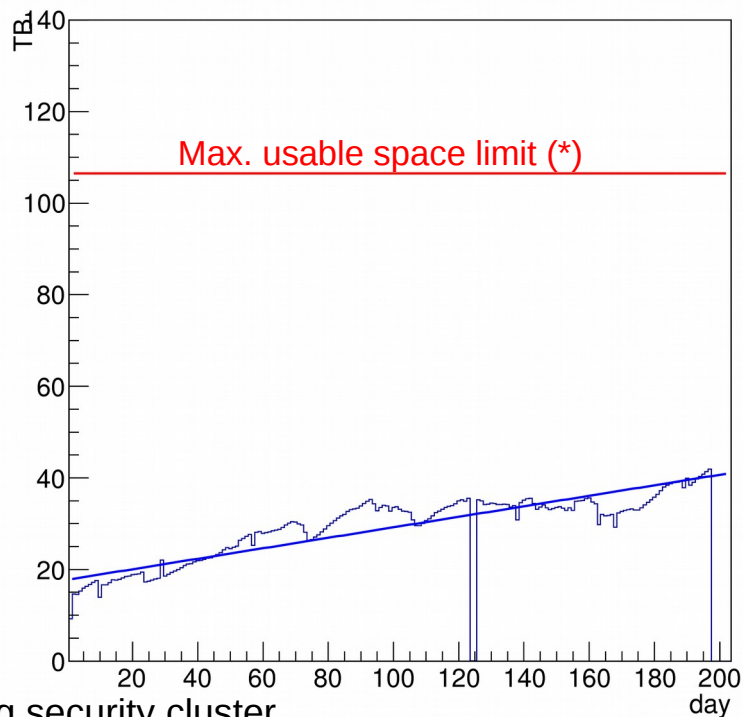
- timber
- monit
- cert
- pub5
- atlas
- cloud
- internal
- cms
- ltdb
- lhcb
- licmon
- mlg
- monit5
- pub
- in
- wslogs
- perfmom
- tim
- alice
- scada
- gitlab
- acron
- lpsa
- landb

Service status: resources since 10/2016

Total space (all clusters)




Total usage (all clusters)



* excluding security cluster

Access control (ACL) implementation (1)

- Why ?
 - Privacy and security requirements
 - Needed for efficient consolidation of resources
- Commercial plugins: 
 - Offer for XPACK from Elastic (shield) too expensive
 - SearchGuard: concerns about performance and integration
- Desired model
 - Pure OpenSource solution (Apache2 license or similar)
 - Index-level security is enough



ACL Implementation (2)



1) Apache proxies and virtual hosts

2) Readonlyrest Elasticsearch plugin (from Simone Scarduzio)



<https://readonlyrest.com/>

<https://github.com/sscarduzio/elasticsearch-readonlyrest-plugin>

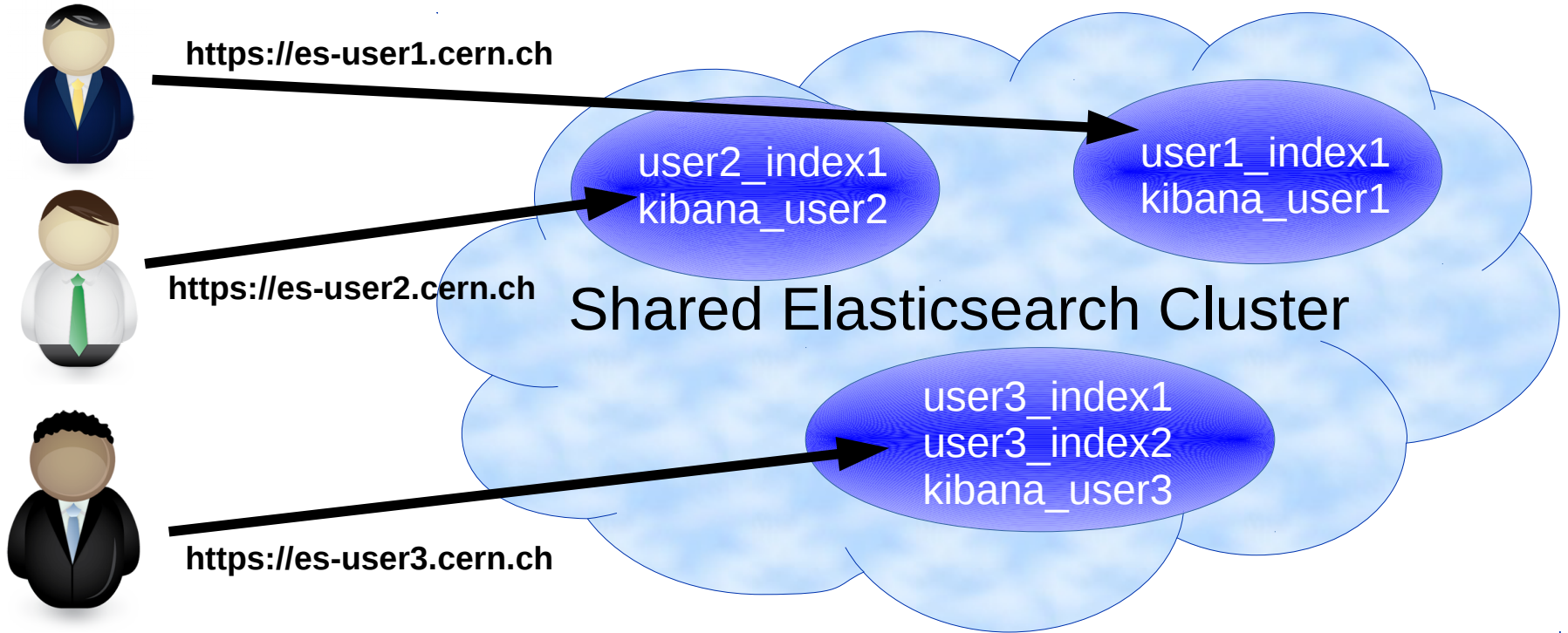
3) Kibana ownhome plugin (from Wataru Takase)

<https://github.com/wtakase/kibana-own-home>

ACL Implementation (3): advantages

- Based on OpenSource solutions only
- Minimal performance impact
- Deployment only on search nodes:
 - No cluster restart needed if ACL configuration changes

ACL implementation (4)



ACL Implementation (5): Services

- SSL only access
- ES access via REST only (JAVA API disabled)
- for each endpoint we offer by default:
 - /kibana Kibana with SSO authentication, ldap-group, read-only
 - /kibana_rw Kibana with SSO authentication, ldap-group, read-write
 - /es ES, one (or more) agreed user/passwords, ro or rw
 - /krb ES, Kerberos5, ldap-group, read-only
 - /krb_rw ES, Kerberos5, ldap-group, read-write
- Customisations are possible

ACL implementation (6): Limitations

- Policy: try to avoid to have to patch the upstream code
- Need to run 2 instances of kibana per search node
 - One for read-write, one for read-only
 - Limitation of kibana which has the path in the configuration file
 - Would need a patch to the code to avoid that
- Cannot have different default dashboards in Kibana per entry-point
 - Would require another kibana patch to allow this

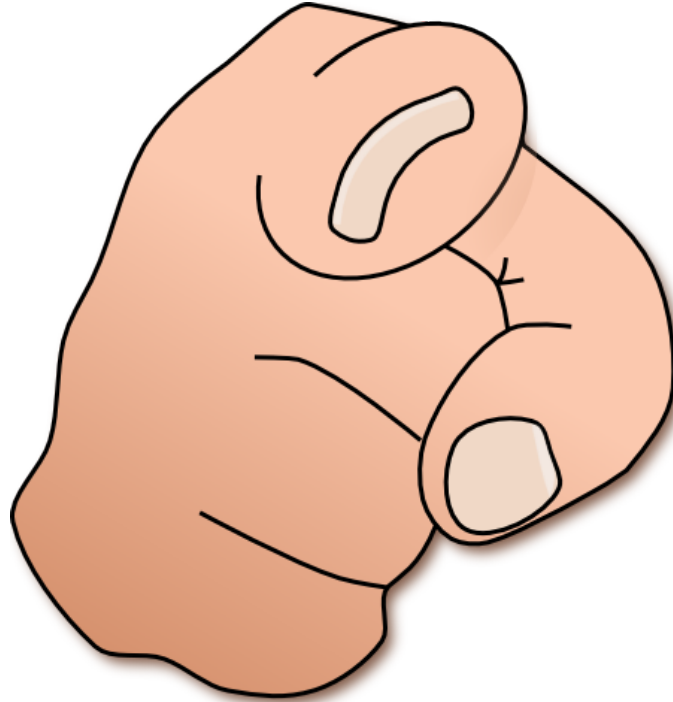
Work in progress

- Deployment of ACLs
 - Less clusters
 - More entrypoints
 - Free unused resources
- Further automation of work flows
 - Rundeck and script based
- Improve service stability
- Anomaly detection
 - Initially for service monitoring

Summary

- Setup a centralised Elasticsearch service at CERN
 - Chosen OpenSource solutions only
 - Fully centrally managed with puppet
- Consolidation ongoing
 - Exploiting ACL schema based on Readonlyrest
- Lessons learned (so far)
 - Met more challenges on the way than initially expected
 - A single cluster is not enough to meet all requirements
 - Very different use cases and requirements

Discussion





www.cern.ch