



OpenStack deployment and related development at KEK-CRC

Wataru Takase

Computing Research Center, KEK

14th February, 2017

Agenda

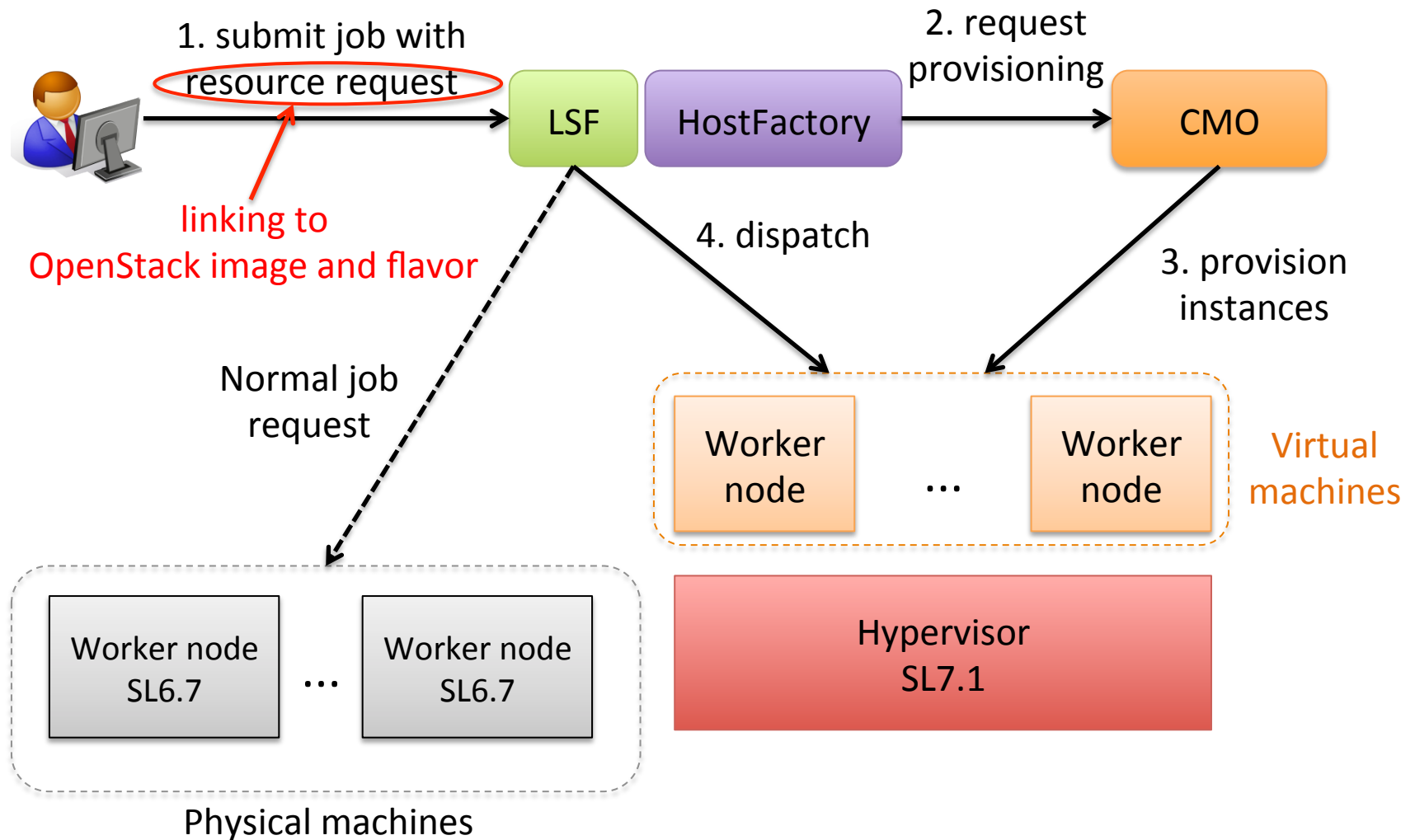
1. OpenStack deployment at KEK/CRC
2. Research on Docker
3. Monitoring related development
 - not only for cloud

1. WIP: OpenStack deployment at KEK/CRC

Cover 2 use cases

- We use **IBM Cloud Manager with OpenStack (CMO)** to provide private cloud service to users.
 - CMO: IBM cloud software based on OpenStack
- 1. Batch integration
 - LSF + OpenStack
 - User just submits job with resource request, then OpenStack prepares required environment.
- 2. Self-service provisioning with minimum privileges
 - Provides experimental groups to customizable machines for development/interactive job/test.

Batch integration



Self-service provisioning with minimum privileges

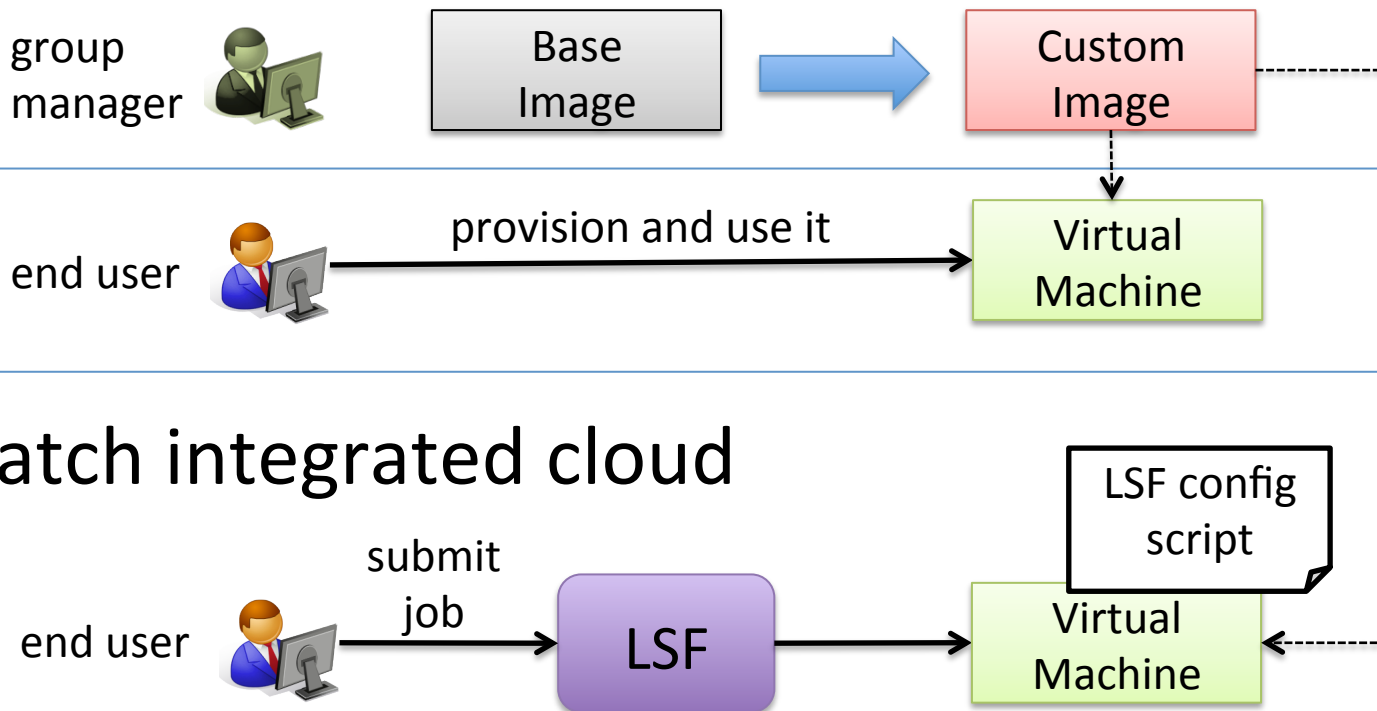
- A user (non group-manager) only can provision instances and has no root privilege.
 - CMO provides self-service portal which is simplified Web interface.
 - Allowed actions are limited by OpenStack role.

Self-service portal for end users

OpenStack dashboard for admins

Planned workflow of using the cloud

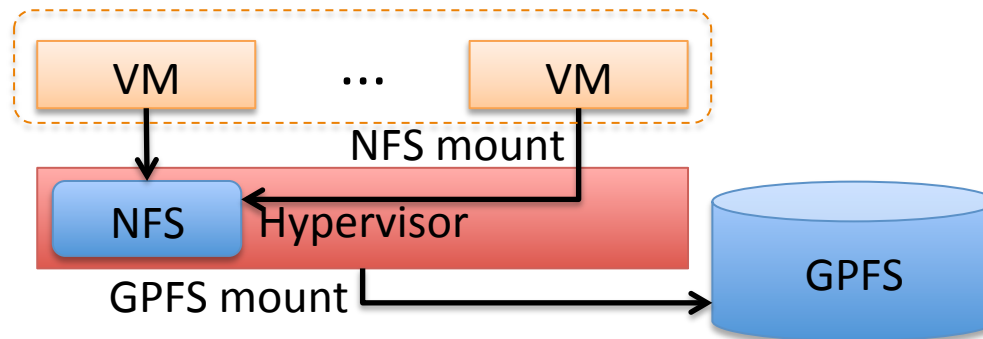
- Cloud admin prepares base images
 - SL6, CentOS7, Ubuntu16
- Self-service



Integration with existing systems

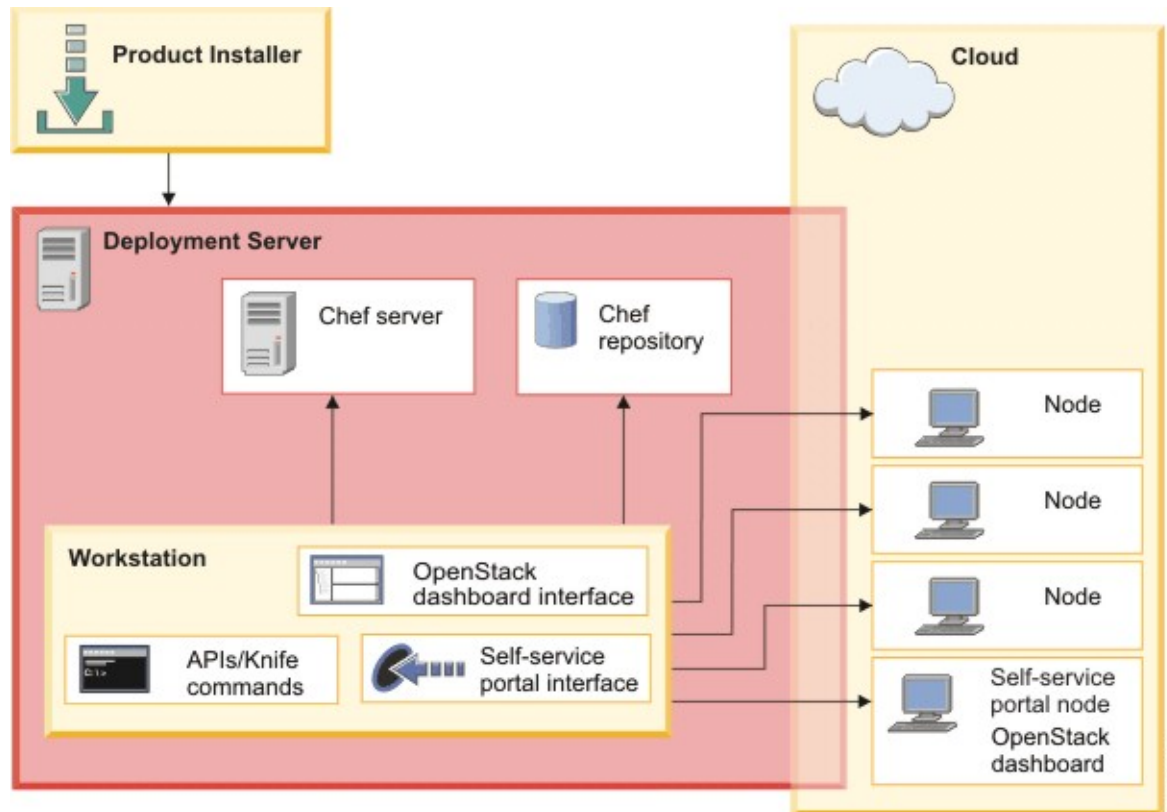
- Integrate with LDAP
 - Use our LDAP as OpenStack authentication backend.
 - Linux accounts inside a VM are also managed by the LDAP.
- Avoid GPFS mount from VM because of Additional GPFS operation
 - GPFS mount requires a node registration to cluster.
 - At the time of VM termination, deletion from the cluster is also required.

➡ Host server mounts GPFS and exposes the directories to virtual machine via NFS.

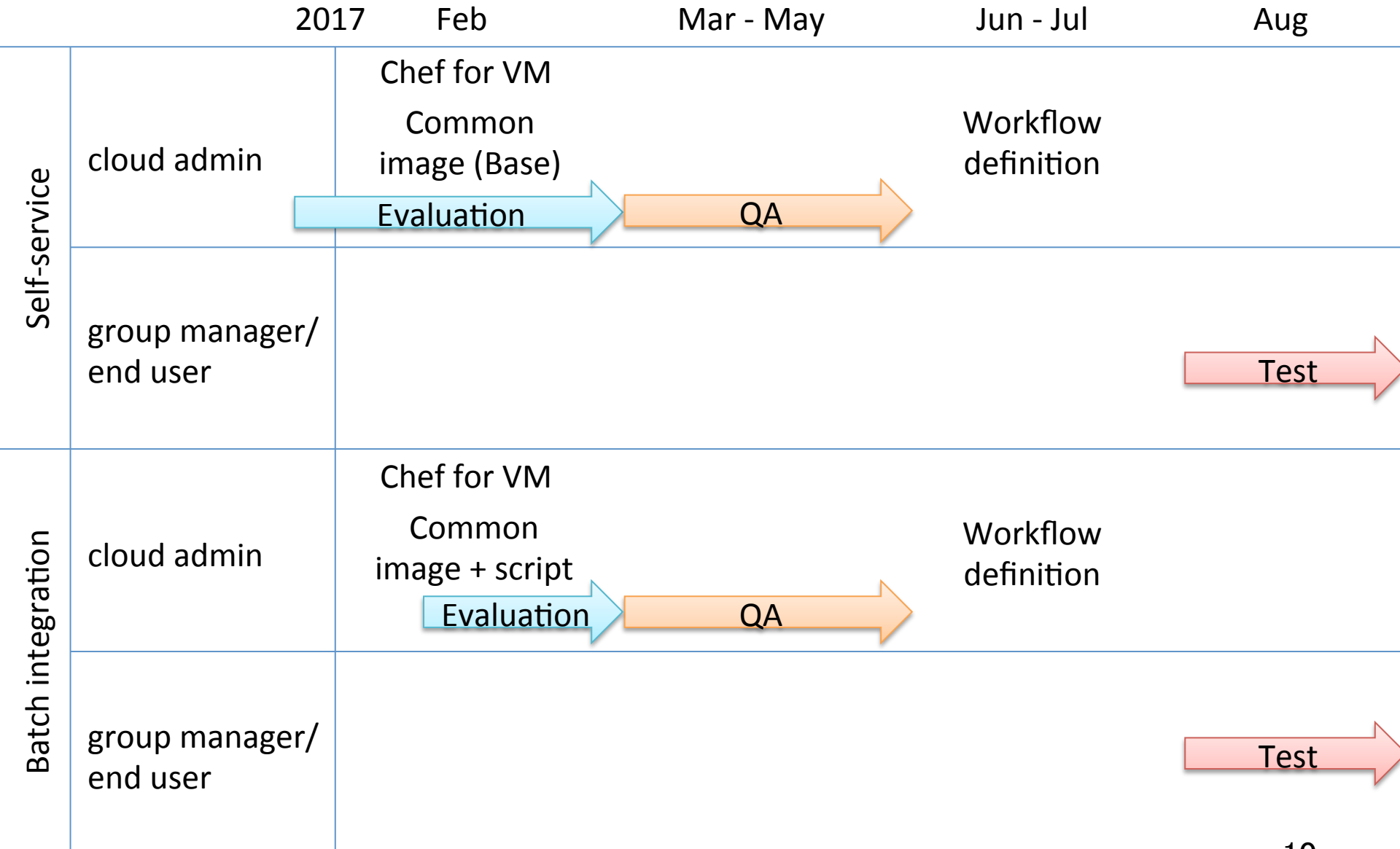


Deployment by Chef

- CMO provides Chef deployment server for automate deployment of cloud infrastructure.
- Investigating a way of VM management by the Chef.



Schedule



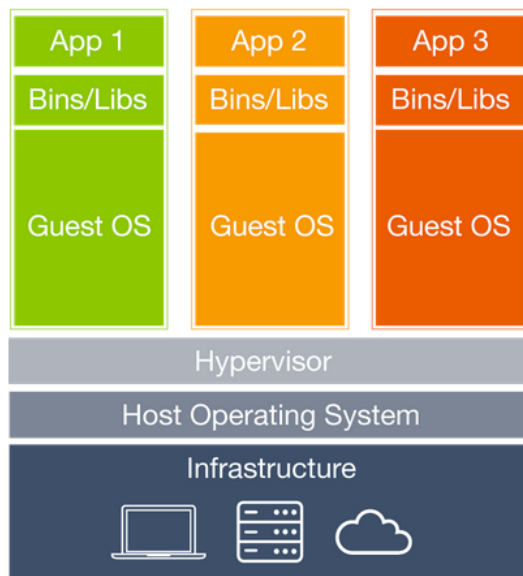
2. Research on Docker

Docker in OpenStack

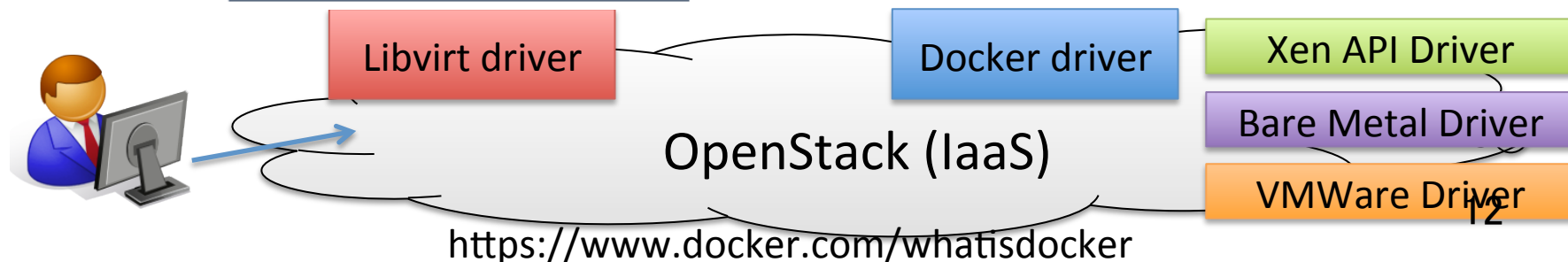
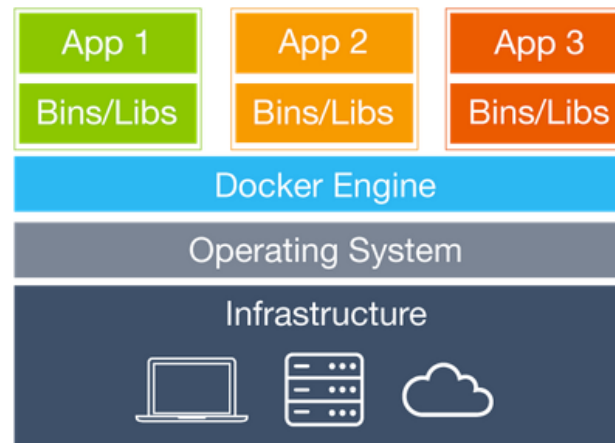
- OpenStack provides 2 ways to handle Docker:
 - 1. **Nova Docker driver**: boot a container as an instance like a VM
 - 2. **Magnum**: Deploy Docker cluster management tool (Kubernetes/Swarm/Mesos) as instances and user can deploy containers on top of the tool

CMO supports it

Each VM runs on virtual hardware

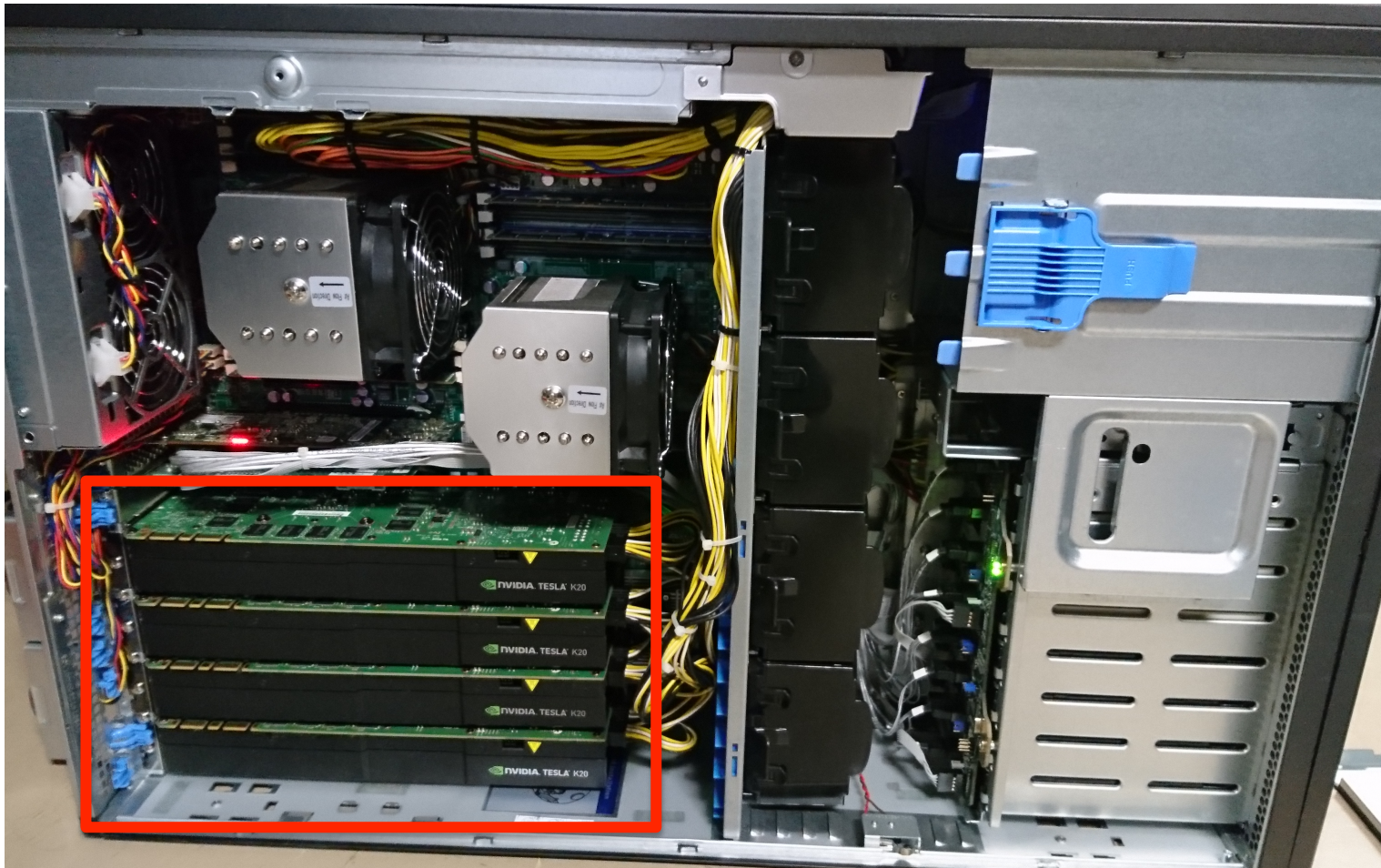


Containers share host kernel and hardware




Investigation of Docker + GPU

- Supermicro SuperWorkstation 7047GR-TPRF
 - NVIDIA Tesla K20 x4



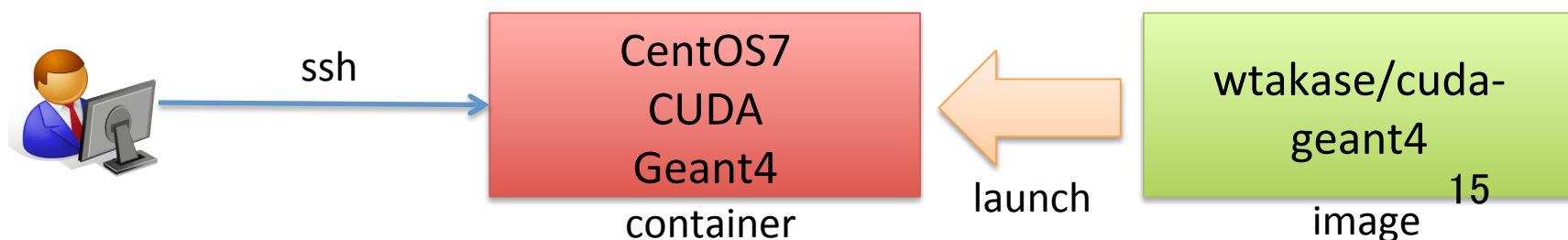
Motivation

- Some users wanted to play with GPUs/CUDA.
- I offered them my machine for it.
- They preferred to use CUDA 8 on CentOS7.
 - My environment is CUDA 7.5 and SL6.

- 
- It's good to use Docker containers:
- Environment isolation from the host
 - Easy resource limitation by cgroups
 - Good portability: Write once, run anywhere
 - Container can be launched as a OpenStack instance by using Nova Docker driver

Create Docker images

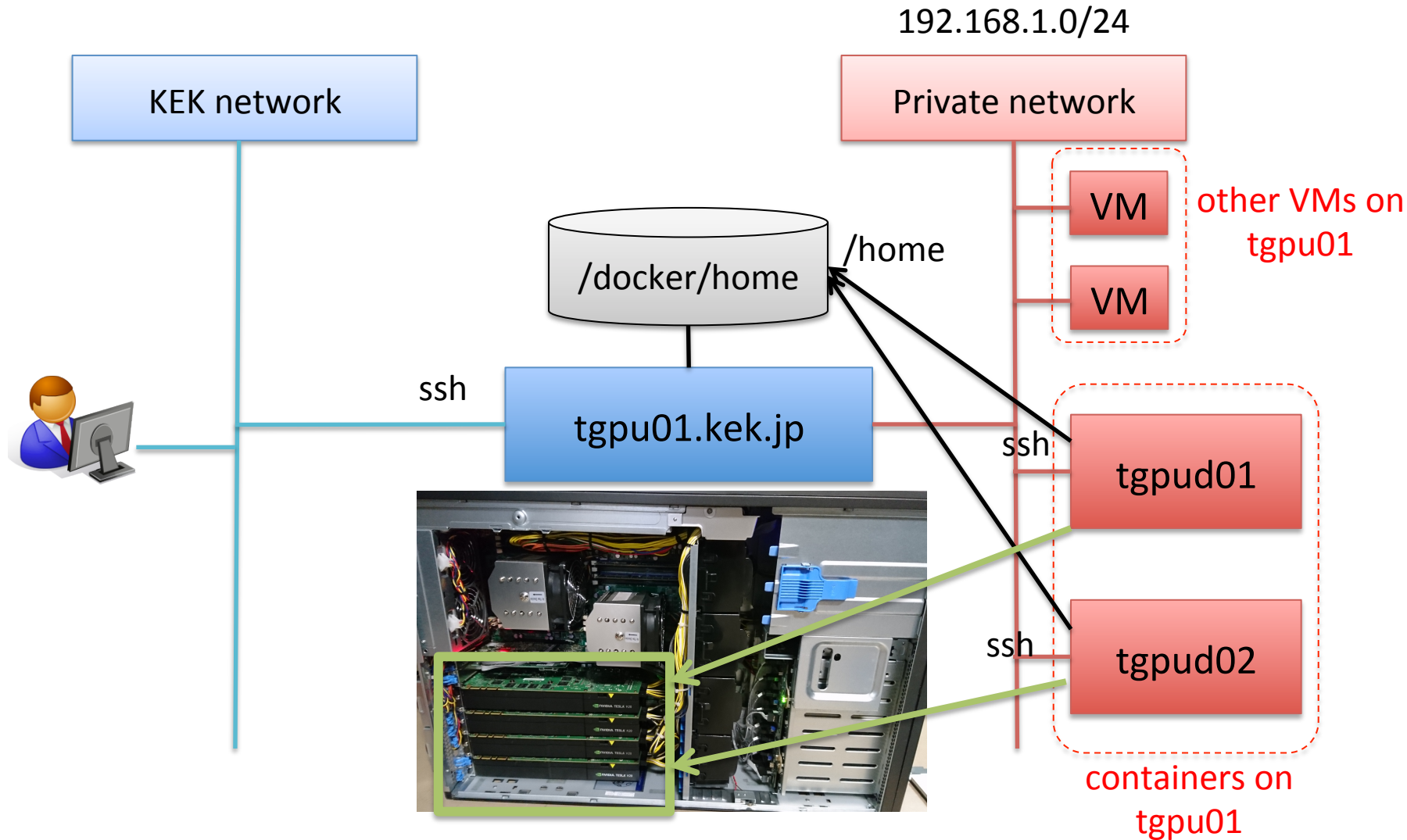
- wtakase/cuda:8.0-centos7
 - installed packages:
 - CUDA 8.0, OpenSSH server
 - <https://github.com/wtakase/docker-cuda>
- wtakase/cuda-geant4
 - extends wtakase/cuda:8.0-centos7
 - additionally installed packages:
 - Development tools
 - ROOT 5.34
 - Geant 4.10
 - <https://github.com/wtakase/docker-cuda-geant4>



Useful Docker options: data volume and device options

- Data volume option
 - enables a container to mount a host directory as a data volume
 - Share the same directory among containers
 - designed to persist data even if the container itself is deleted
 - # docker run -d -v /docker/home:/home wtakase/cuda-geant4
 - mounts the host directory /docker/home into the container at /home.
- Device option
 - allows you to run host devices inside the container
 - # docker run -d --device /dev/nvidiactl --device /dev/nvidia-vm --device /dev/nvidia0 wtakase/cuda-geant4
 - allows to run 1 GPU(/dev/nvidia0) inside the container
 - cf.) There is a *--privileged* option which allows you to access all host devices inside the container

Overview of the environment

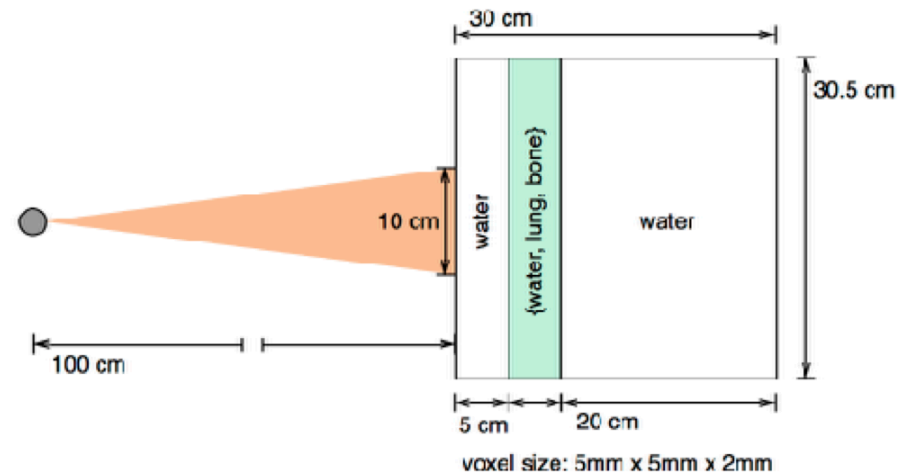


Shogo confirmed a MPEXS application works fine on the container

MPEXS on Docker Container

- Compared the MPEXS computing performances (process time/event) between Native and Docker container.
- Run a simulation with 1 million events of a 20MeV electron travelling through a water phantom 2000 times.
 - Number of threads: 4096 blocks x 256 threads/block
 - Total number of threads = Total number of events
 - Process time/event = Total process time / Total number of events
 - Fit the results by Gaussian distribution
- Confirmed there is no performance difference.

	Process time (us/event)	
	Native	Docker
CUDA 7.5	17.63±0.49	17.62±0.49
CUDA 8.0	17.46±0.48	17.46±0.48

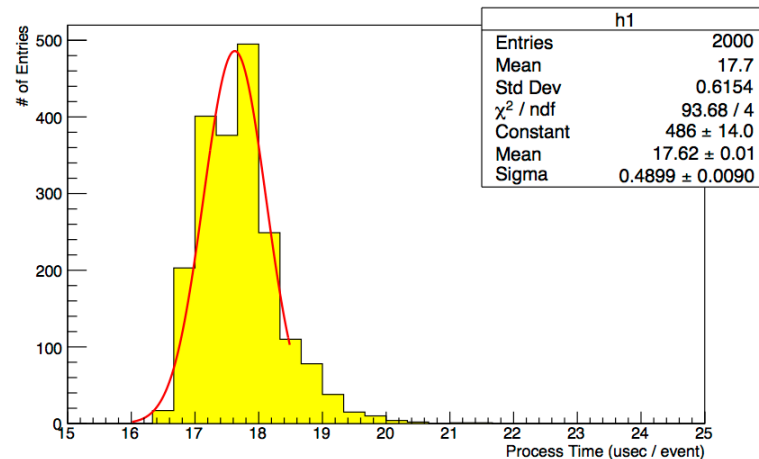
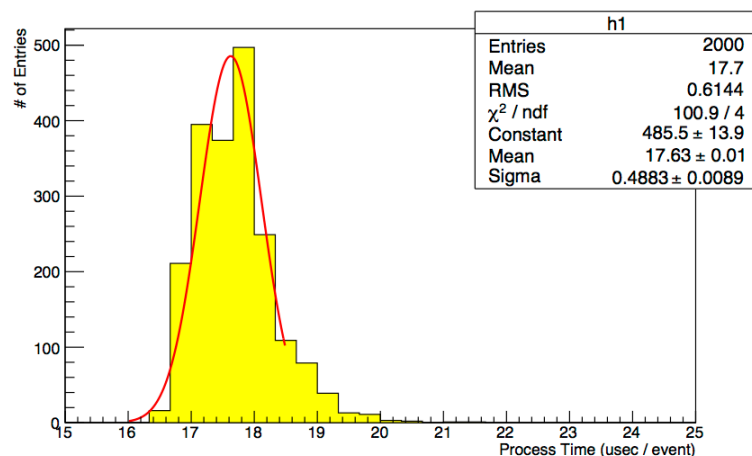


MPEXS on Docker Container

Native

Docker

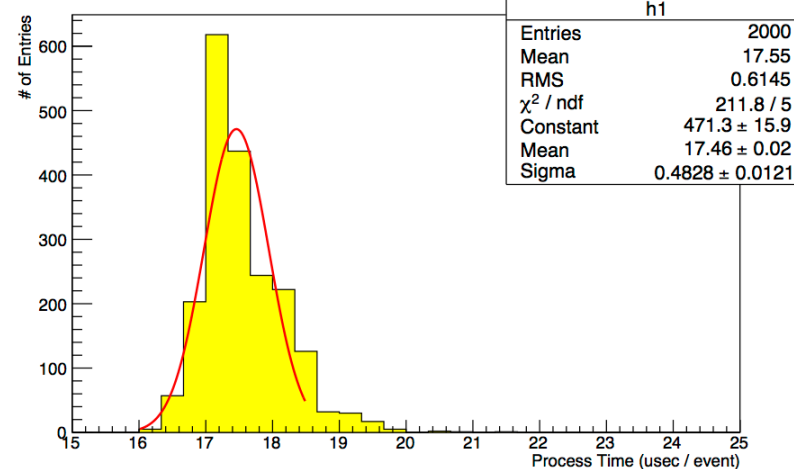
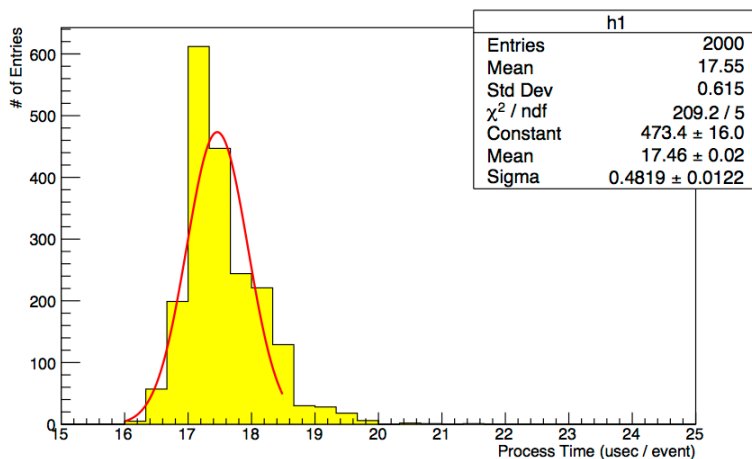
CUDA 7.5



Native

Docker

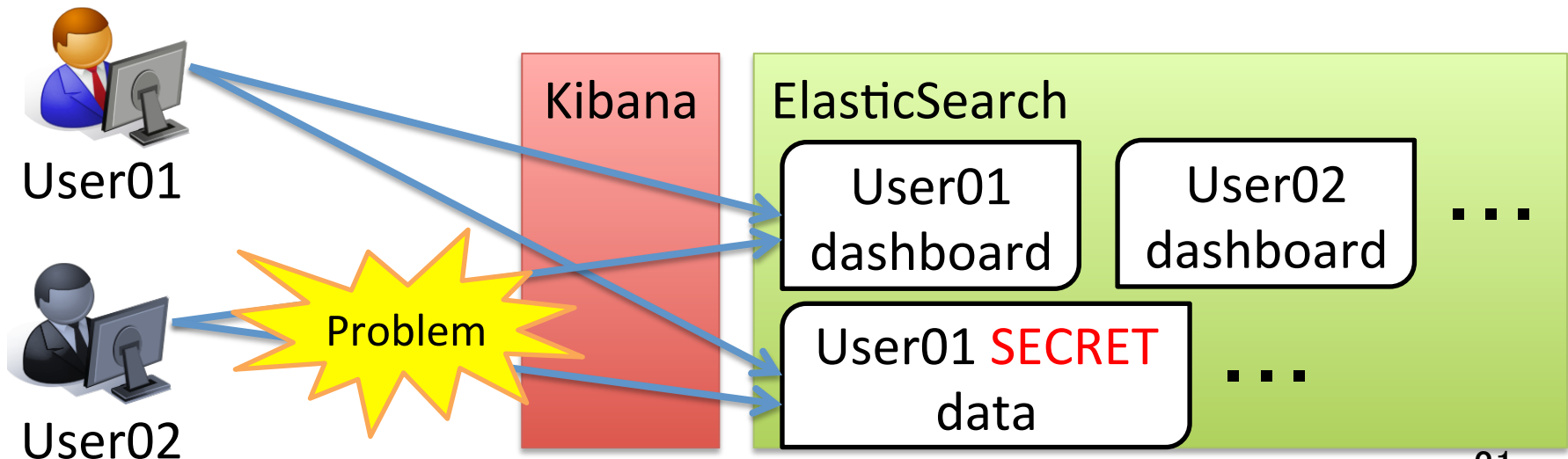
CUDA 8.0



3. Monitoring related development for secure use of Elasticsearch and Kibana

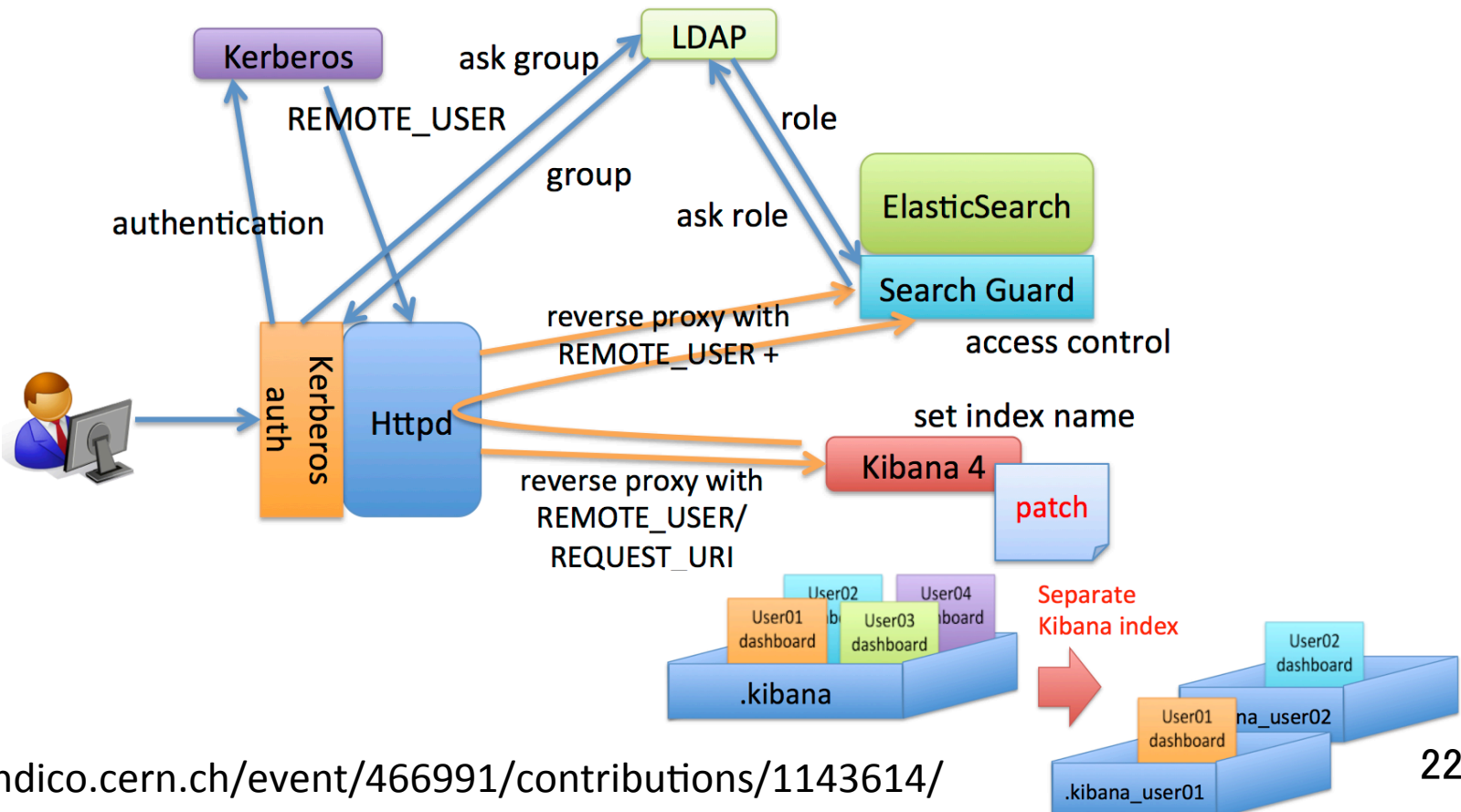
Motivation

- Kibana + Elasticsearch **lack access control feature**
- Multiple users/groups use single Kibana + Elasticsearch
 - Any user can access to all Elasticsearch data
 - Need access control



We worked together and provided a solution

- **Kerberos** authentication integration
- **Kibana patch** enabled to user/group based dashboard separation
- **Search Guard** enabled user/group based Elasticsearch access control



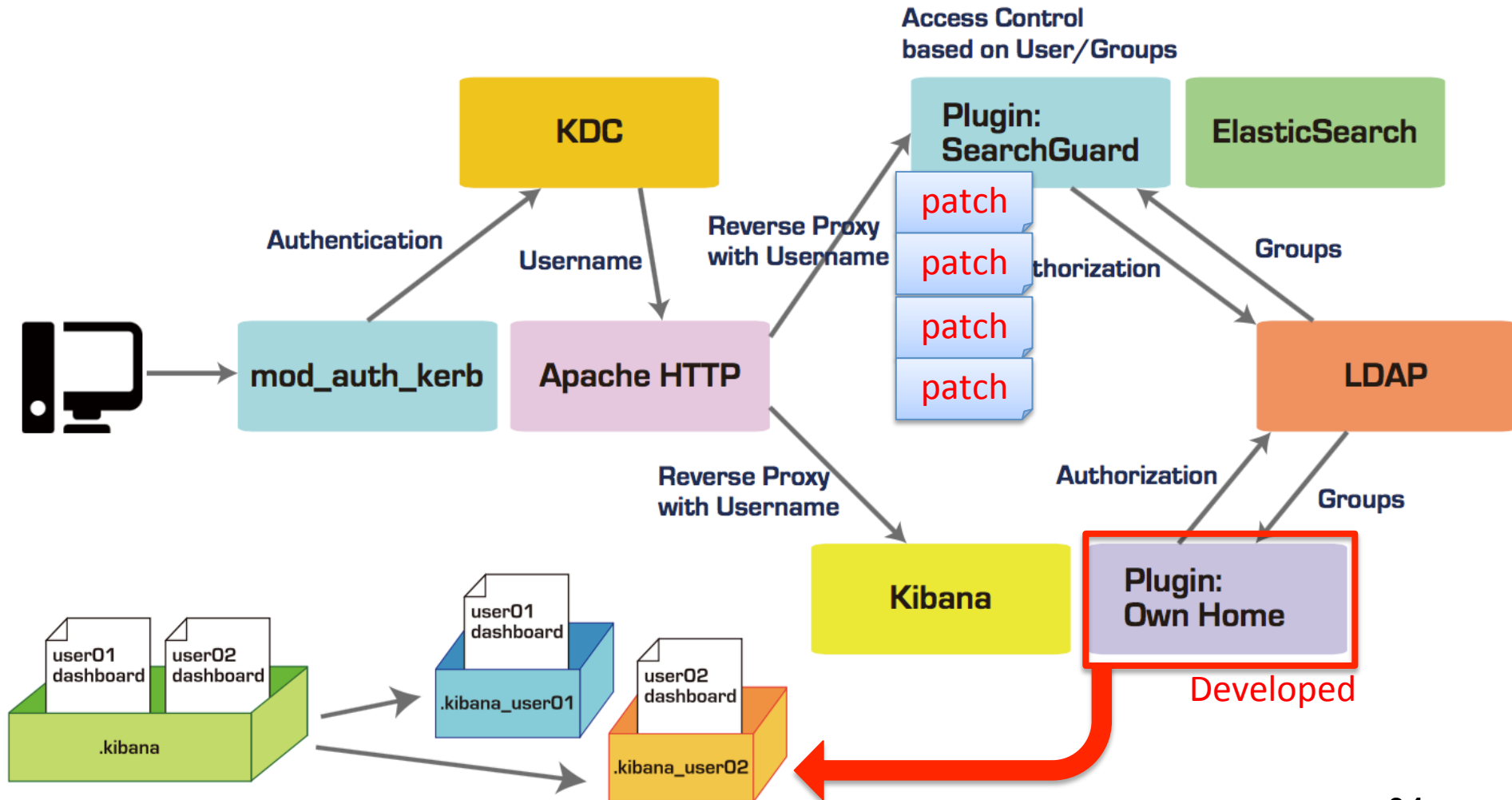
Catch up with the fast-paced developments

Our target versions

	Apr 2016	Current	Future
Kibana	4.1	4.6	5.x
ElasticSearch	1.5	2.4	5.x
Search Guard	0.5	2.4	5.x

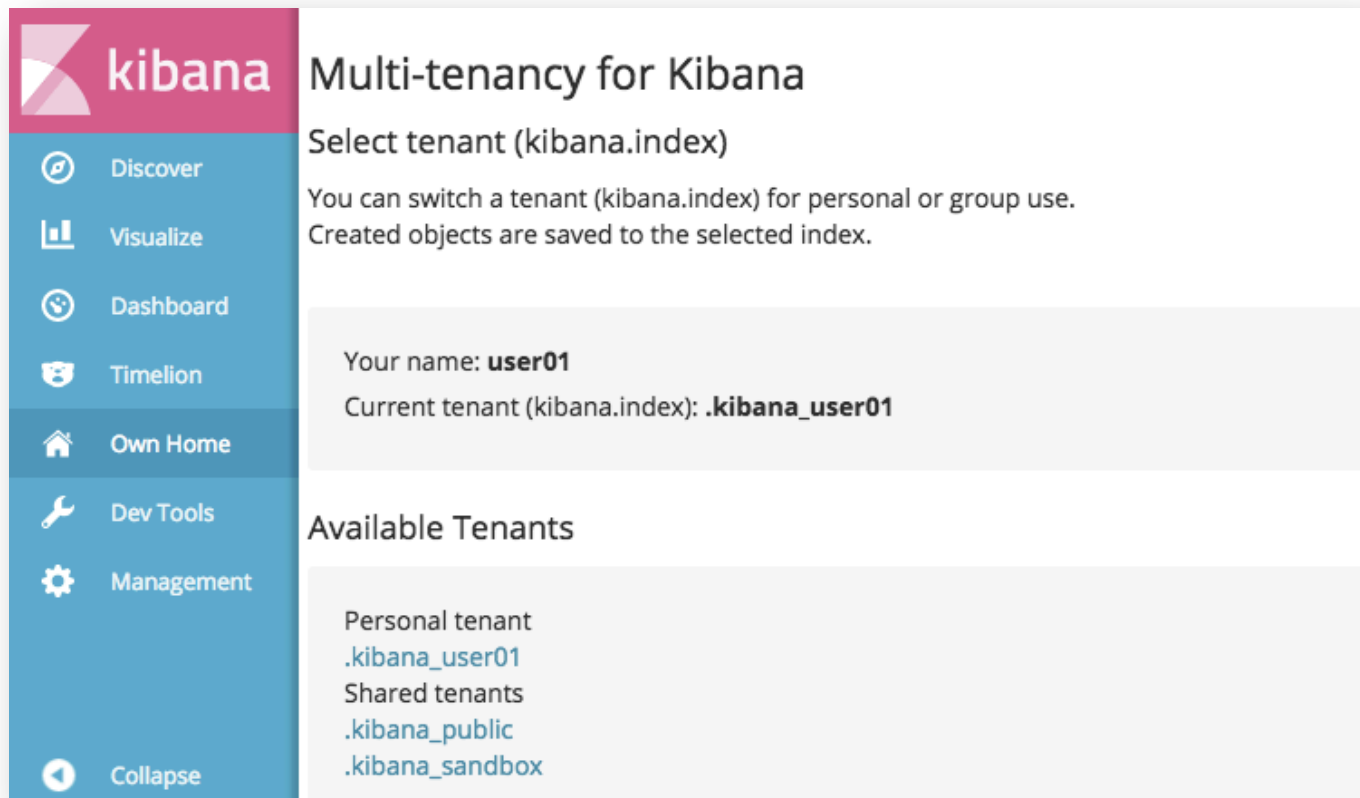
- The Kibana patch had been **no longer adaptable**
 - Needed to develop new one
- Current Search Guard configuration/usage has been completely **different from the old one**
 - Needed investigation

Overview of our updated solution



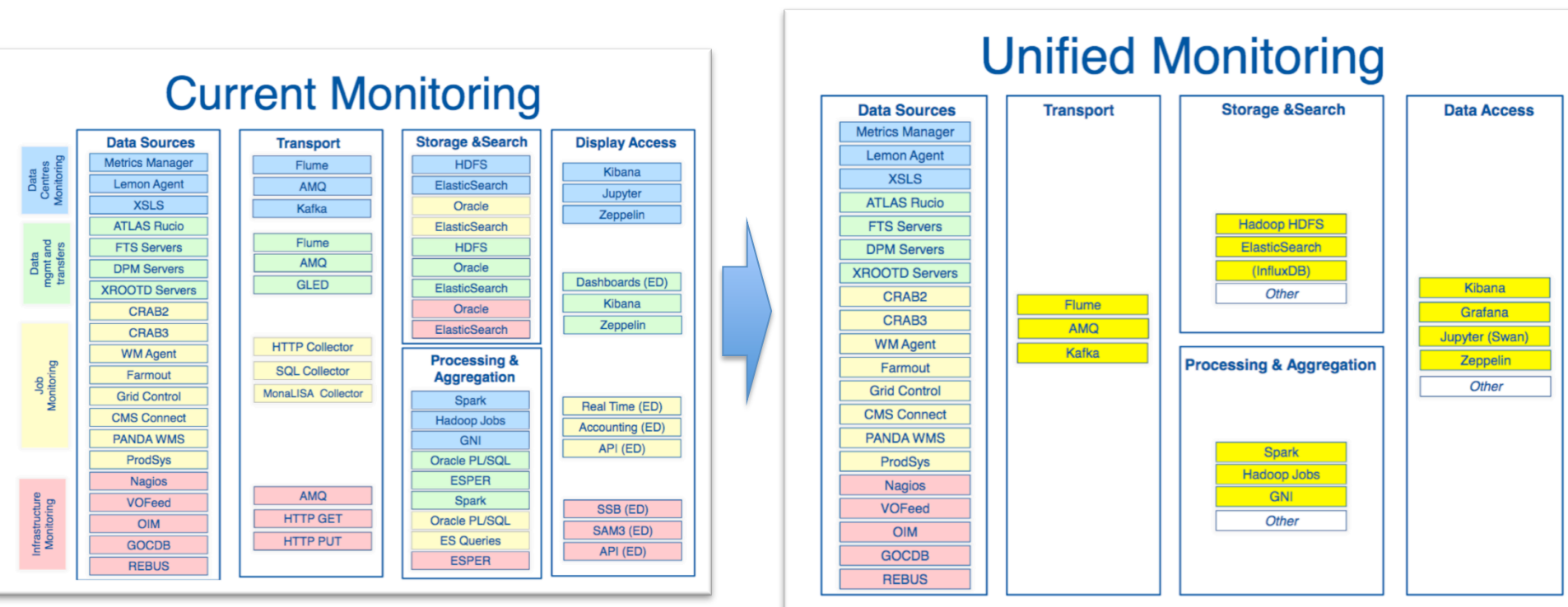
Development of a Kibana plugin

- Kibana plugin: Own Home
 - <https://github.com/wtakase/kibana-own-home>
 - Works **without** any Kibana patches
 - Supports Kibana 4.6, 5.0, 5.1, 5.2

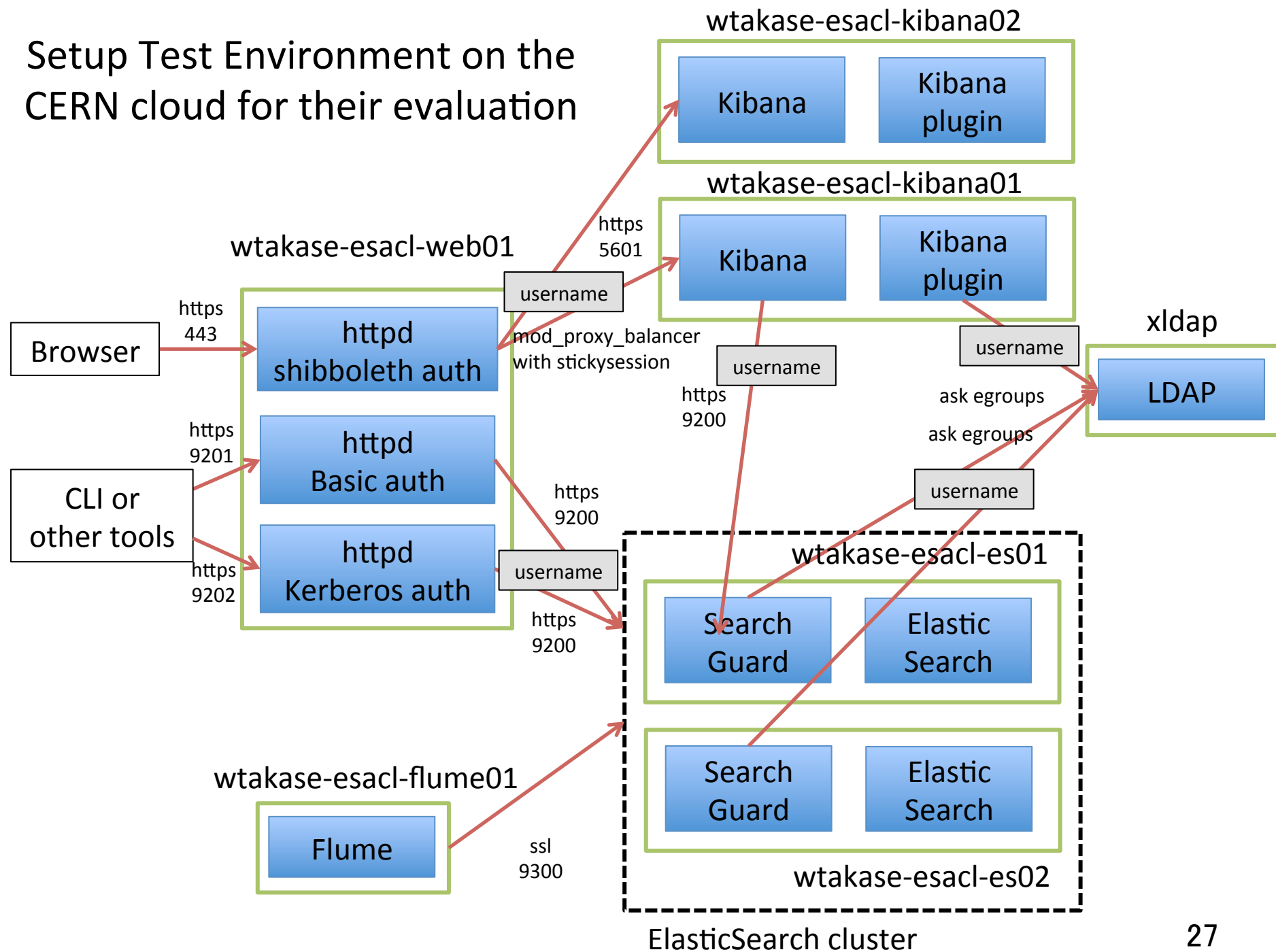


Collaboration with the CERN monitoring team

- They are trying to consolidate monitoring services in CERN.
 - After last HEPiX, we started to work together for the consolidation and also other ES/Kibana topics.



Setup Test Environment on the CERN cloud for their evaluation



Summary

- At KEK/CRC, cloud service for self-service and batch integration is under construction and the test service will be available in August.
- We created Docker images for GPU/CUDA and confirmed a MPEXS application works with no performance difference comparing to Native.
- For monitoring security, we updated our solution and developed Kibana plugin working without Kibana patches.
- CERN tries to unify monitoring services inside CERN and we started collaboration work.