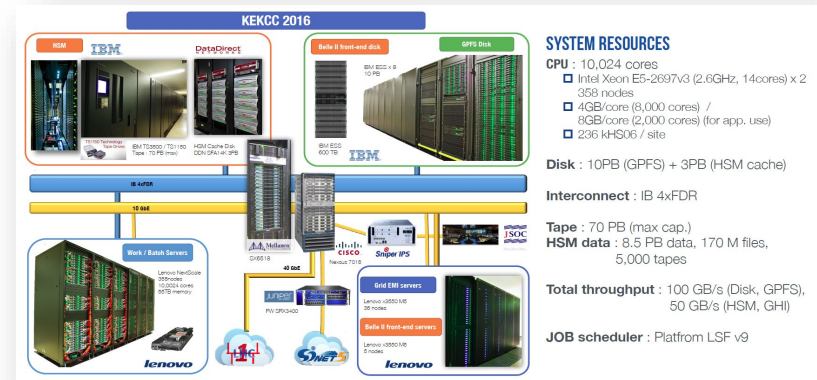


# TYL-FJPPL: Comp\_04

## Evolution of the computing environment for high-energy and astroparticle experiments



### Members (\*Leader)

CC-IN2P3: Renaud Vernet\* (renaud.vernet@cc.in2p3.fr)  
G. Rahal, F. Hernandez, F. Suter, B. Rigaud, V. Hamar

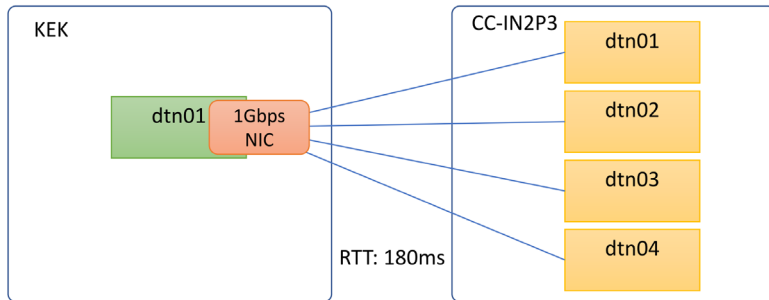
KEK-CRC: Tomoaki Nakamura\* (tomoaki.nakamura@kek.jp)  
W. Takase, S. Kaneko, T. Sasaki, S. Suzuki, K. Murakami, G. Iwai

### Focus of COMP\_04 project

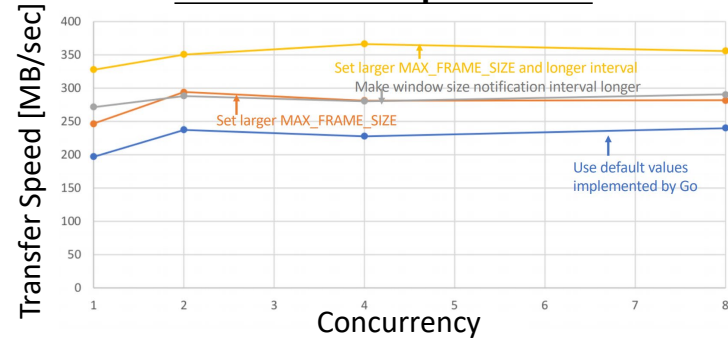
- Deployment of HTTP based data transfer system
- Study of hybrid computing system of on-premise resource and commercial cloud services
- Application and development of machine learning for computer system

## Deployment of HTTP based data transfer system

### Prototype of data transfer nodes



### Performance improvement



Improvement of transfer speed

- We have investigated the feature and performance of data transfer protocol, for example, the difference of plane HTTP1.1 and HTTP2 with some algorithm of transport layer security by constructing a prototype of data transfer nodes in both sites.
- We found some problems in the implementation of HTTP2 inside the Go language which is used for an application tool of our data transfer test.
- We updated the module and our transfer tools, then confirmed certain improvements to the long-range transfer speed.

### Code modification of the transport module in Go language

```

140 - func configureTransport(t1 *http.Transport) (*Transport, error) {
141 + func configureTransport(t1 *http.Transport, fct *FlowControlTransport) (*Transport, error) {
142     connPool := new(ClientConnPool)
143     t2 := &Transport{
144         ConnPool: noDialClientConnPool{connPool},
145         t1: t1,
146         ConnPool: noDialClientConnPool{connPool},
147         MaxFrameSize: fct.MaxFrameSize,
148         TransportDefaultConnFlow: fct.TransportDefaultConnFlow,
149         TransportDefaultStreamFlow: fct.TransportDefaultStreamFlow,
150         TransportDefaultStreamMinRefresh: fct.TransportDefaultStreamMinRefresh,
151     }
152     connPool.t = t2
153 }
154
155 func (t *Transport) newClientConn(c net.Conn, single bool) (ClientConn, error) {
156     cc := &ClientConn{
157         t: t,
158         tconn: c,
159         maxStreamID: 0,
160         nextStreamID: 1,
161         maxFrameSize: t.MaxFrameSize, // spec default 9000
162         transportDefaultStreamMinRefresh: t.TransportDefaultStreamMinRefresh, // spec default 1024
163         initialWindowSize: 65535, // spec default
164         maxConcurrentStreams: 1000, // "softlimit" per spec, 1000 seems good enough.
165         peerNameAndAddrListSize: 8192, // "softlimit", per spec, use "8k" instead.
166     }
167     return &ClientConn{net.Conn, single bool} (ClientConn, error)
168 }
169
170 func (t *Transport) newClientConn(c net.Conn, single bool) (ClientConn, error) {
171     cc := &ClientConn{
172         ID: Settings.ID,
173         ID: Settings.ID,
174         ID: Settings.ID,
175         ID: Settings.ID,
176     }
177 }
178 }
179

```

Relax limit of maximum frame size

```

48 48 // transportDefaultStreamMinRefresh is the minimum number of bytes we'll send
49 49 // a stream-level WINDOW_UPDATE for at a time.
50 50 transportDefaultStreamMinRefresh = 4 << 10

```

```

140 - func configureTransport(t1 *http.Transport) (*Transport, error) {
141 + func configureTransport(t1 *http.Transport, fct *FlowControlTransport) (*Transport, error) {
142     connPool := new(ClientConnPool)
143     t2 := &Transport{
144         ConnPool: noDialClientConnPool{connPool},
145         t1: t1,
146         ConnPool: noDialClientConnPool{connPool},
147         MaxFrameSize: fct.MaxFrameSize,
148         TransportDefaultConnFlow: fct.TransportDefaultConnFlow,
149         TransportDefaultStreamFlow: fct.TransportDefaultStreamFlow,
150         TransportDefaultStreamMinRefresh: fct.TransportDefaultStreamMinRefresh,
151     }
152     connPool.t = t2
153 }
154

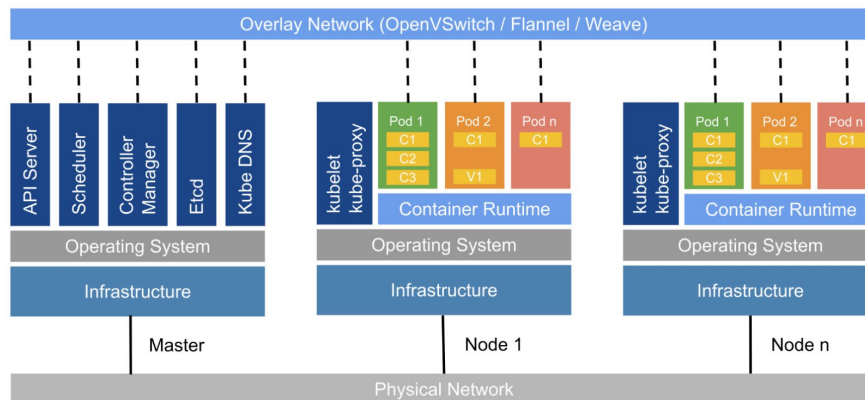
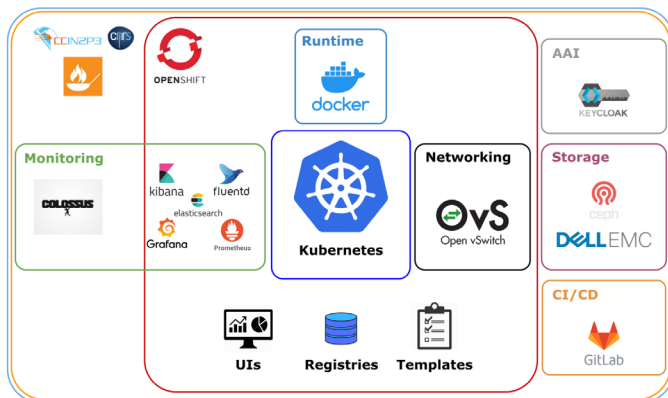
```

Decrease interval of window update notification

Modified transfer tool (Chasqui): <https://github.com/wtakase/chasqui/>

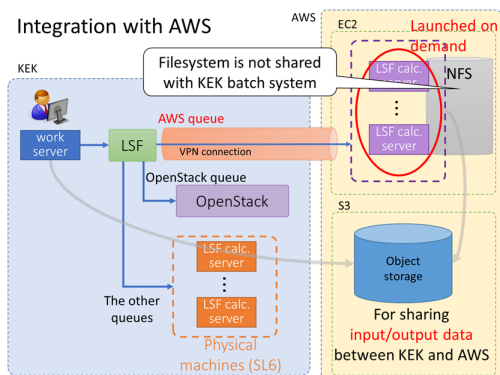
Study of hybrid computing system of on-premise resource and commercial cloud services

## Software modules and configurations for virtualization developing at CC-IN2P3



- Toward studying the effectiveness of this hybrid-type system, we discussed a lot of things related to the virtualization technology and the method of its deployment often used in the cloud environment.
- We focused on the usability of Containers (docker, Singularity) and deployment method using Kubernetes at the workshop held at Lyon in last year for the future collaborative R&D.

## Cloud On-premise hybrid system at KEK (available at the next KEKCC in 2020)



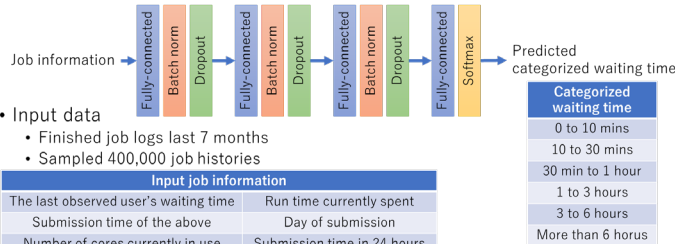
## Providing user friendly environment by Singularity container and Kubernetes deployment tools



## Application and development of machine learning for computer system

### Waiting time estimation based on Deep Learning at KEK-CRC

- Set up Fully-Connected Neural Network model



- Input data
  - Finished job logs last 7 months
  - Sampled 400,000 job histories

Input job information	
The last observed user's waiting time	Run time currently spent
Submission time of the above	Day of submission
Number of cores currently in use	Submission time in 24 hours
Current CPU utilization efficiency	Queue
Number of current waiting jobs	

Categorized waiting time
0 to 10 mins
10 to 30 mins
30 min to 1 hour
1 to 3 hours
3 to 6 hours
More than 6 hours

### Command line tool for end user

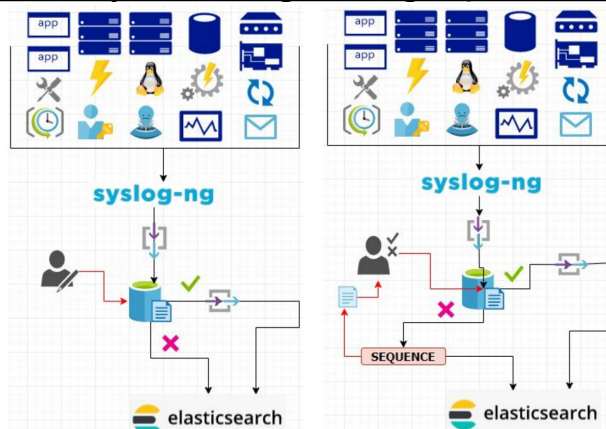
- The last observed user's waiting time
- Submission time of the above
- Run time currently spent
- Day of submission
- Number of cores currently in use
- Number of current waiting jobs
- Submission time in 24 hours

```

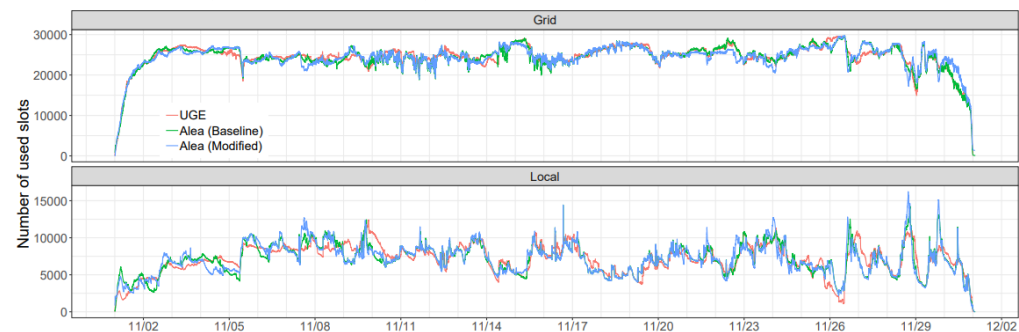
takase@cw14 ~]$ bpredict -q 1 -u takase
Expected waiting time: 0~599 sec (prob. 92.1123 %)
takase@cw14 ~]$
    
```

- The effectiveness of Deep Learning for the estimation of waiting time after the submission of user's jobs have been studied. A command line tool to provide the current estimation for users was also developed at KEK-CRC.
- We confirmed the applicability of the tools developed even for the batch job system utilized in CC-IN2P3.
- In addition to the batch job information, we discussed the possibility to extend this kind of log analysis method for the computing system, storage system, and network system by using the further brandnew tools utilized at CC-IN2P3.

### Pattern detection system of log messages (SEQUENCE) developing at CC-IN2P3



### Alea job scheduling simulator utilizes at CC-IN2P3



## Summarizing 2019's activities

- 16 registered participants
- 12 presentations
- 2 hands-on sessions

<https://indico.in2p3.fr/event/19919/>

## Agenda

### Dec. 2nd, 2019

- Welcome and introduction, Renaud Vernet (CC-IN2P3)
- Status report of KEK and KEK-CRC, Tomoaki Nakamura (KEK-CRC)
- CC-IN2P3 site report, Matthieu Puel (CC-IN2P3)
- ARM server: performance evaluation, Wataru Takase (KEK-CRC)
- Computing facility strategy, Vanessa Hamar (CC-IN2P3)
- Improving fairness in batch systems, Frederic Suter (CC-IN2P3)
- Machine learning on batch system, Wataru Takase (KEK-CRC)
- Data transfer test KEK-CCIN2P3, Wataru Takase (KEK-CRC)
- Storage analysis with SPARK, Antoine Dubois (CC-IN2P3)
- Log analysis: SEQUENCE, Fabien Wernli (CC-IN2P3)
- Kubernetes @ CC-IN2P3: what's cooking, Benjamin Guillon (CC-IN2P3)
- Containers for users, Sébastien Gadrat (CC-IN2P3)
- Conclusions and wrap-up, Renaud Vernet (CC-IN2P3)

### Dec. 3rd, 2019

- Hands-on session, Fabien Wernli and Wataru Takase
- Preparation of FJPPL report and proposals, Renaud Vernet and Tomoaki Nakamura

## FJPPL – Japan-France workshop on computing technologies

2 Dec 2019, 09:00 → 3 Dec 2019, 22:20 Europe/Paris

202 (CC-IN2P3)

Renaud Vernet (CCIN2P3)

**Description** The goal of this workshop is to explore relevant technologies, exchange experience and share ideas among experts of both Japan and France organisations in several scientific domains.

This is the 4th edition of this workshop, which is organized annually in the framework and with the sponsorship of the [France-Japan Particle Physics Laboratory](#). The agendas of previous editions are available:

- 2018
- 2017
- 2016
- 2015

Guidelines for speakers

Information requested:

• to provide the support of your presentation (slides, videos, documents, ...) on time, that is, not later than the time your presentation is scheduled to begin. After logging into Indico with your individual identifier, you will be able to fully manage your contribution, including uploading material to it.

• to upload the slides of your presentations in PDF format. Additional formats are also accepted but please make sure at least a PDF version is provided.

• to protect your presentation (for example, by password) if you want your presentation to be protected. The agenda of the meeting will be publicly available and eventually indexed by web search engines. If this is a problem for you, please let the chairman know for setting access protections

• to allow part of the time allocated in your slot for questions and comments from the audience

Information for participants

This meeting is organized so that participants can attend partially and should feel free to attend only their talks of interest. If you want to attend, you must register not later than November 15 by clicking in the link below.

Participants: Benjamin Guillon, Benoît DELAUNAY, David Bouvet, Fabien Wernli, Frederic Suter, Ghita Rahal, Gino Marchetti, Renaud Vernet, Sébastien Gadrat, Tomoaki Nakamura, Vanessa Hamar, Wataru Takase



## **Plan (a) Deployment of HTTP based data transfer system**

- We will start the real file transfer by setting up the dedicated data transfer node with high-speed storage in the production network environment for the systematic evaluation.
- We will try to compare the result with the other data transfer methods like XrootD transfer and cache mechanism (Xcache).

## **Plan (b) Study of hybrid computing system of on-premise resource and commercial cloud services**

- We continuously exchange the information on software for virtualization and experience to integrate which is well utilized in Europe and Japan in the existing Grid environment, for example, Singularity middleware.

## **Plan (c) Application and development of machine learning for computer system**

- We investigate the performance improvements of the learning process for that application by deploying this application to the GPU clusters located in CC-IN2P3.
- We check the correctness of the obtained estimation by the future job accounting data stored in the batch job scheduler in conjunction with Alea job scheduling simulator.
- We try to apply SEQUENCE software module, which is an open-source module of the text pattern recognition, to the job accounting data and the status information of the computer system.

## **Organizing a workshop on computing at Lyon in December 2020**