External Resources: Clouds and HPCs for the expansion of the ATLAS production system at the Tokyo regional analysis center

Michiru Kaneda

(ICEPP, The University of Tokyo)

On behalf of the ATLAS Collaboration

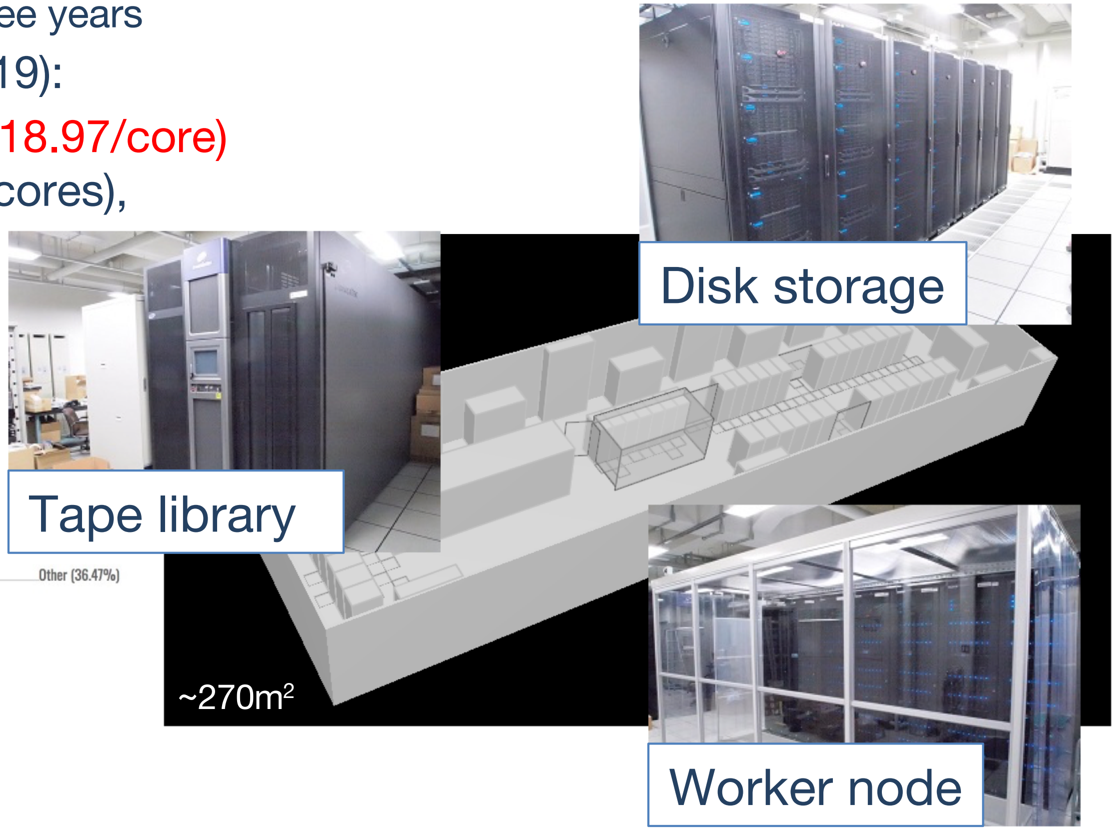
07/Nov/2019

24th International Conference on Computing in High-Energy and Nuclear Physics

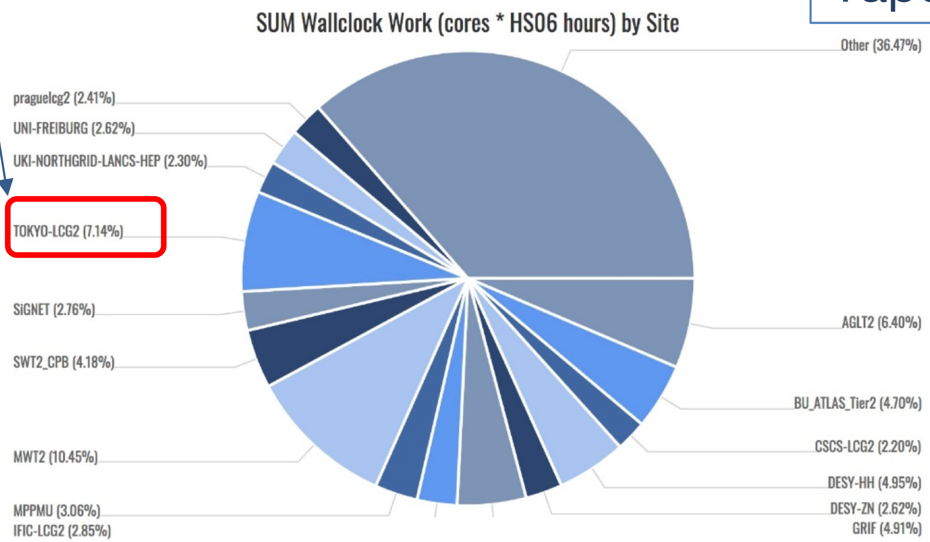
Adelaide, Australia

The Tokyo regional analysis center

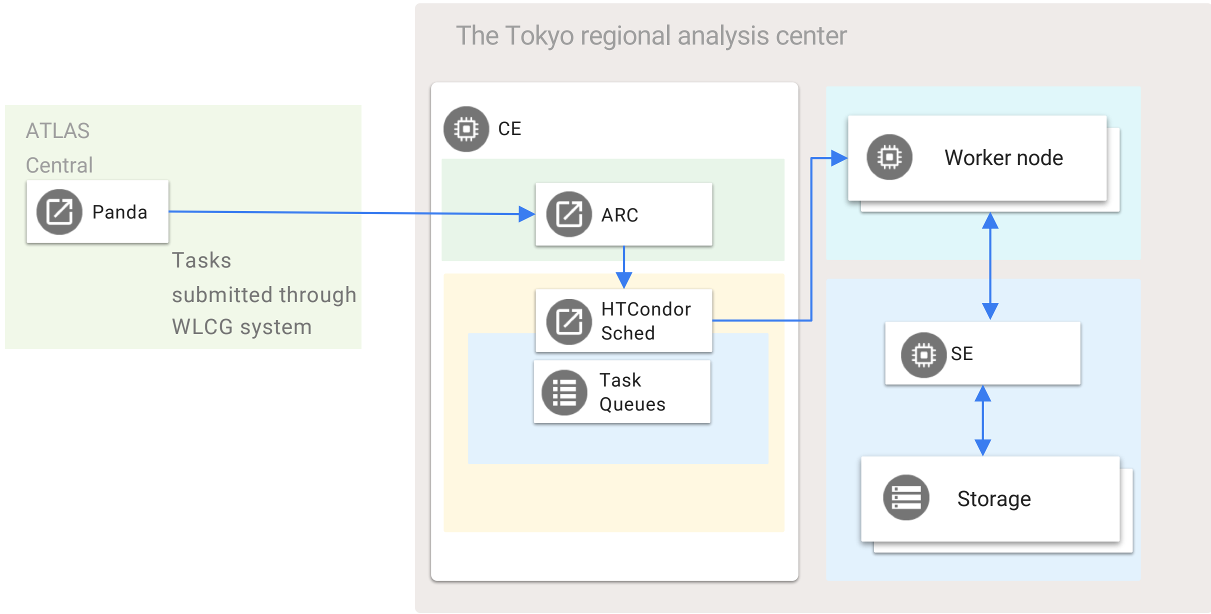
- The computing center at ICEPP, the University of Tokyo
- Supports ATLAS VO as one of the WLCG Tier2 sites
  → Provides local resources to the ATLAS Japan group, too
- All hardware devices are supplied by the three years rental
  → All hardware devices are renewed in three years
- Current system (Starting from Jan/2019):
  → Worker node: 10,752cores (HS06: 18.97/core) (7,680 for WLCG, 145689.6 HS06*cores), 3.0GB/core
  → File server: 15,840PB, (10,560TB for WLCG)



7% of Tier2 sites of ATLAS



Our Local System

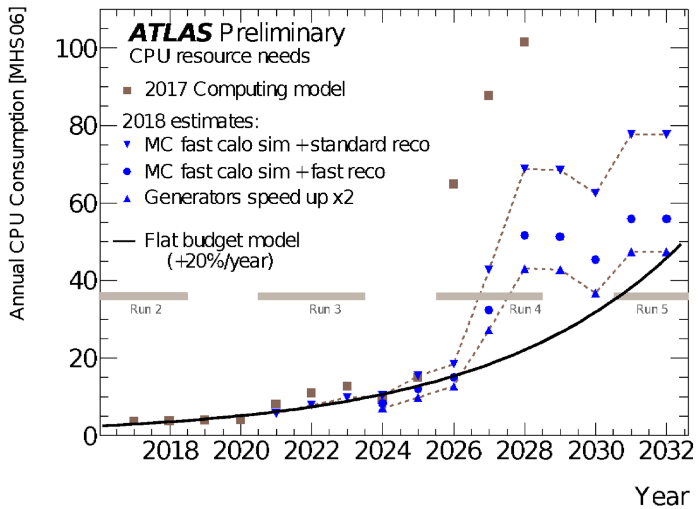


- Panda: ATLAS job management system, using WLCG framework
- ARC-CE: Grid front-end
- HTCondor: Job scheduler

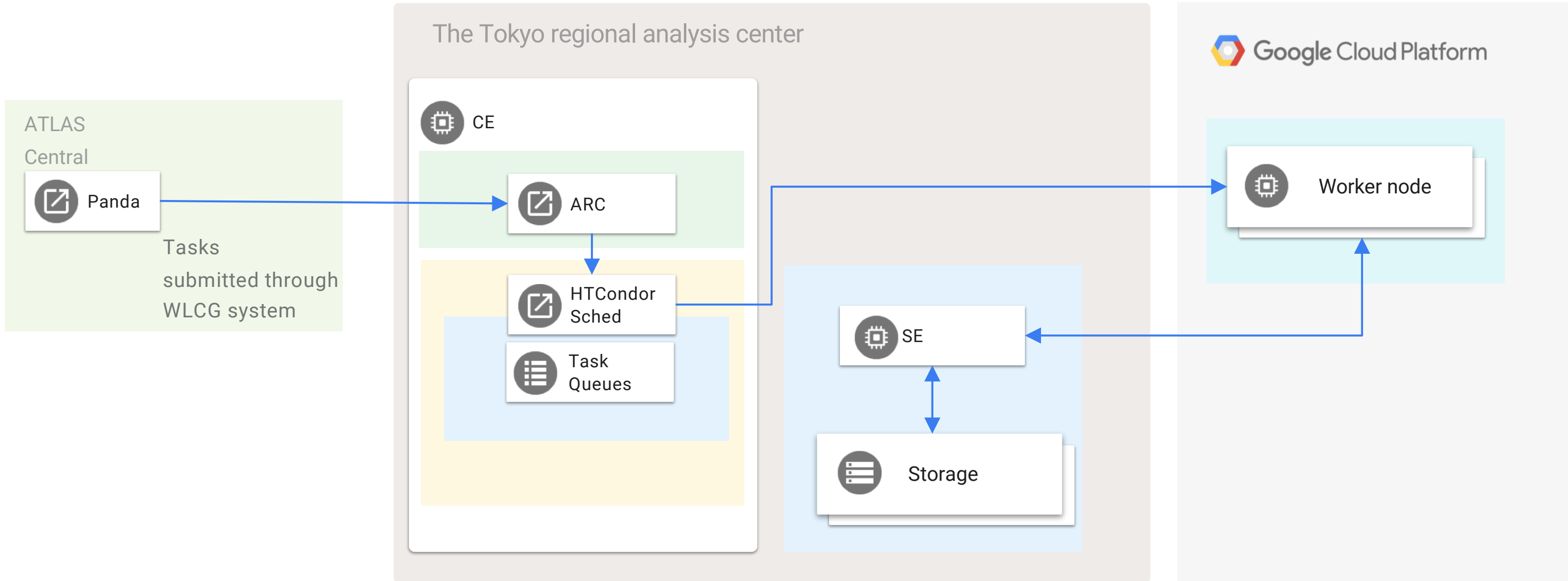
Future Computing Resources

- WLCG have provided enormous computing resources and made it possible to give a lot of results by the LHC experiments
  →But we will need more resources for the future experiments

- CERN plans High-Luminosity LHC in 2026
  →The peak luminosity: x 5
  →The current system cannot provide enough resources with expected budgets
  →More improvements or new ideas are necessary
    →Software update
    →New devices: GPGPU, FPGA, (QC)
    →New grid structure: Data Cloud
    →External resources: HPC, Commercial cloud



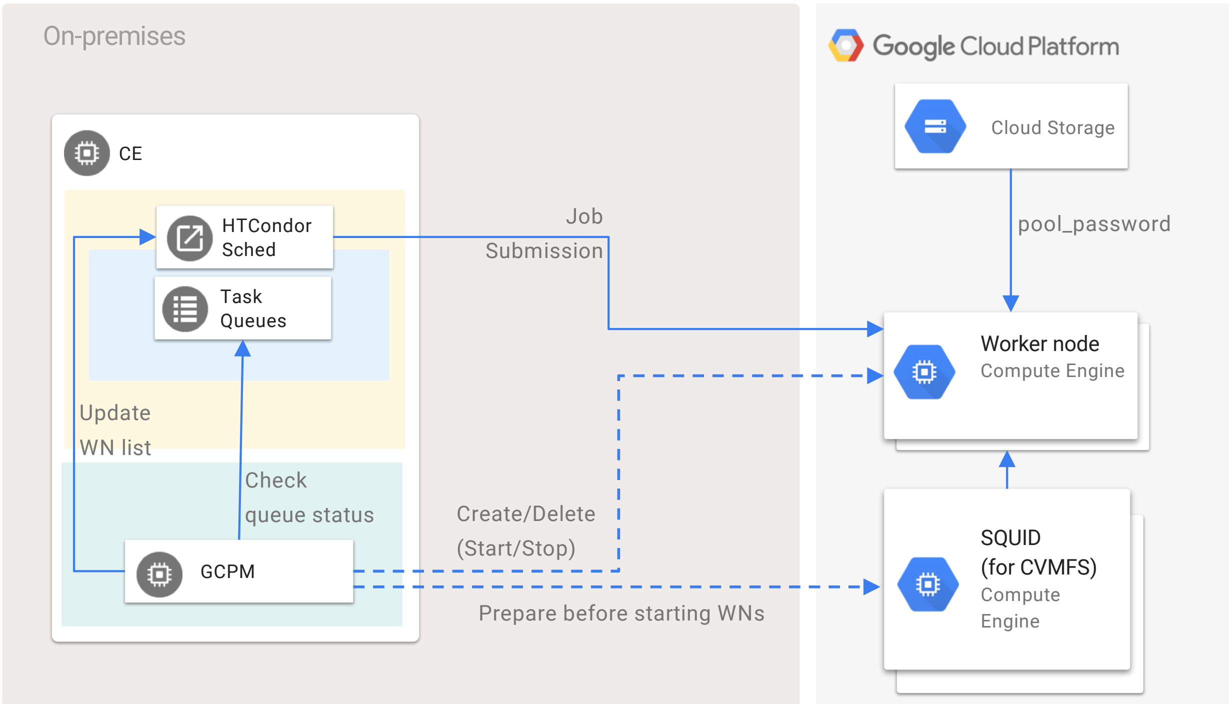
Hybrid System with Google Cloud Platform



- Cost of storage is high
  →Additional cost to extract data
- Only worker nodes (and some supporting servers) were deployed on cloud, and other services are in on-premises
  →Hybrid system

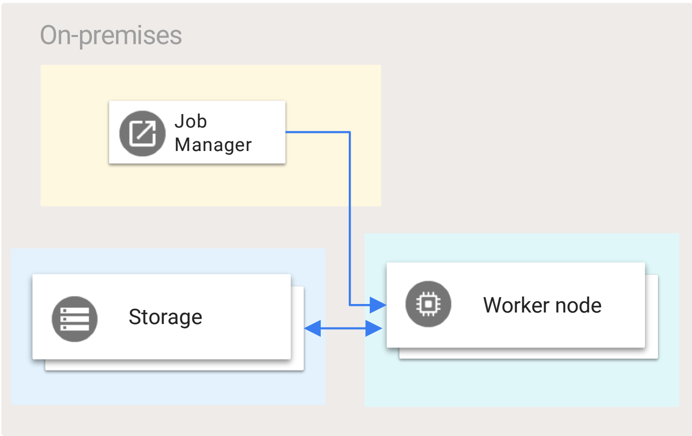
Google Cloud Platform Condor Pool Manager

- Google Cloud Platform Condor Pool Manager (GCPM)
  → https://github.com/mickaneda/gcpm
    → Can be installed by pip:
      → ***$ pip install gcpm***
- Manage GCP resources and HTCondor's worker node list
  → On-demand instance preparation
- Can be used for any of HTCondor systems
  → Useful for high-peak needs of CPUs, GPGPU instances, many cores instances, or high-memory instances which are needed once in a while

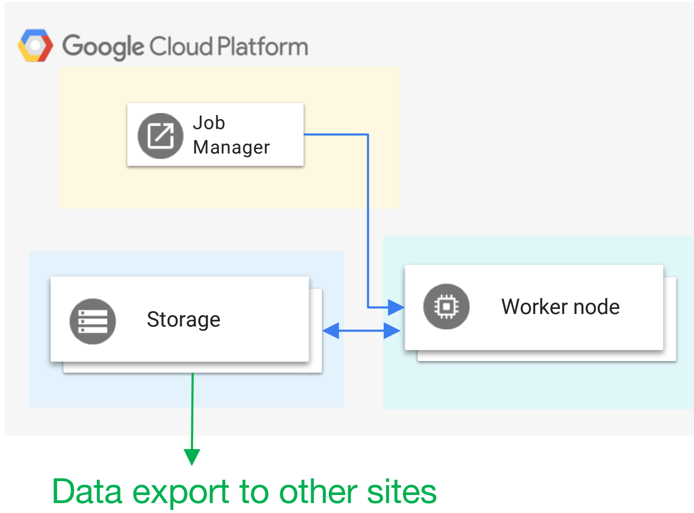


Cost Estimation

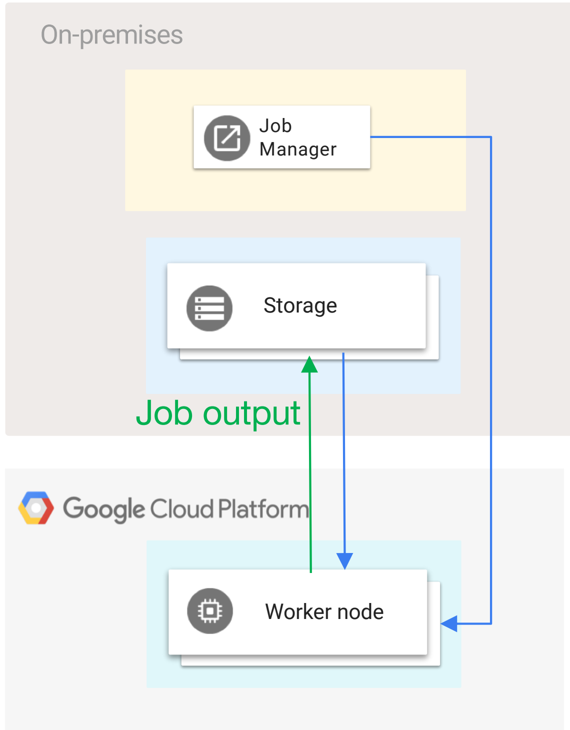
Full on-premises system



Full cloud system



Hybrid System



- Estimated with Dell machines
- 10k cores, 3GB/core memory, 35GB/core disk: $5M
- 16PB storage: $1M
- Power cost: $20k/month

  → For 3 years usage: ~$200k/month (+Facility/Infrastructure cost, Hardware Maintenance cost, etc…)

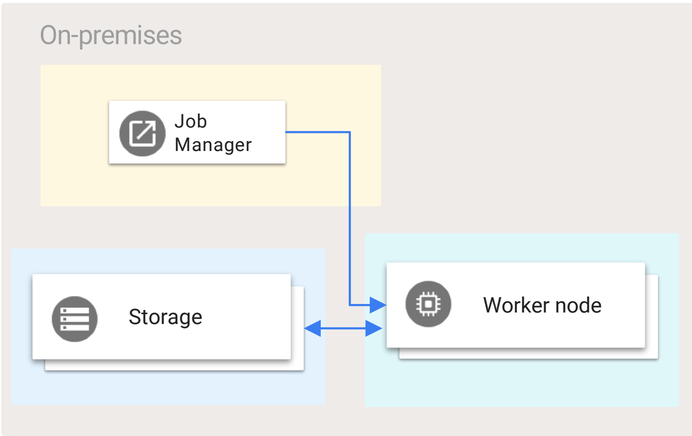
- For GCP, use 20k to have comparable spec

  → Use Preemptible Instance (Hyperthreading On, half)
- 8PB storage which is used at ICEPP for now
- Cost to export data from GCP

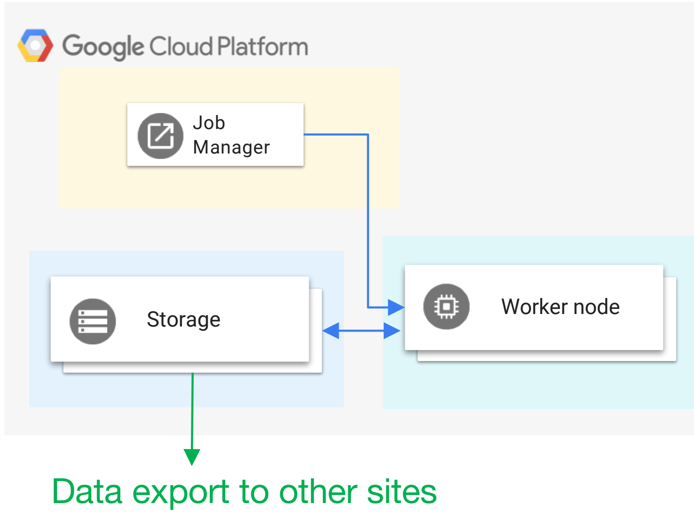
https://cloud.google.com/compute/pricing
https://cloud.google.com/storage/pricing

Cost Estimation

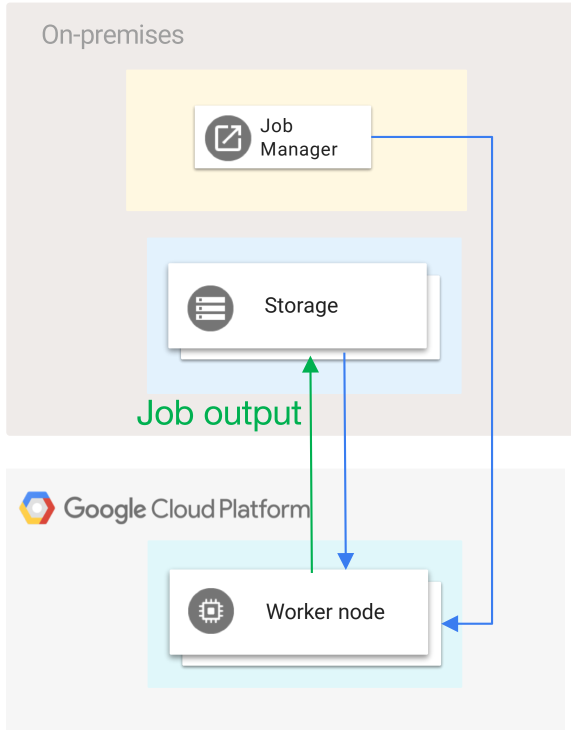
Full on-premises system



Full cloud system



Hybrid System



- Estimated with Dell machines
- 10k cores, 3GB/core memory, 35GB/core disk: $5M
- 16PB storage: $1M
- Power cost: $20k/month
  - → For 3 years usage: ~$200k/month (+Facility/Infrastructure cost, Hardware Maintenance cost, etc…)

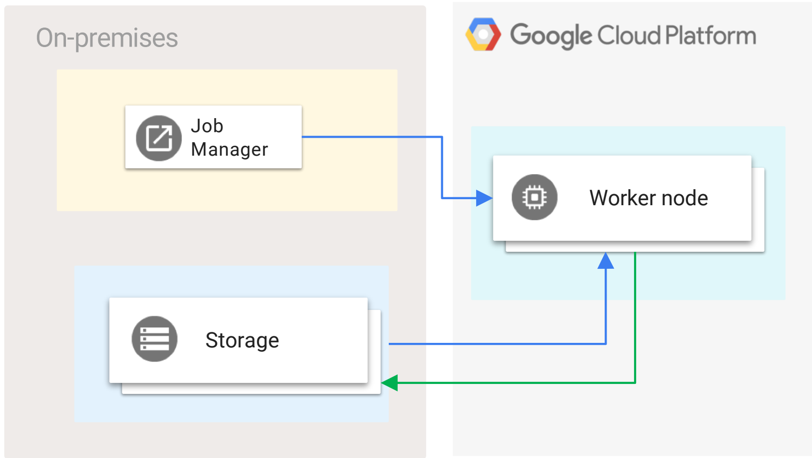
Resource	Cost/month
vCPU x20k	$130k
3GB x20k	$52k
Local Disk 35GBx20k	$28k
Storage 8PB	$184k
Network Storage to Outside 600 TB	$86k

Total cost: $480k/month

Resource	Cost/month
vCPU x20k	$130k
3GB x20k	$52k
Local Disk 35GBx20k	$28k
Network GCP WN to ICEPP Storage 300 TB	$43k

Total cost: $243k/month + on-premises costs (storage $30k/month + others)

1 Day Real Cost



Reduced Hybrid system: 1k cores, 2.4GB/core memory

1 Day Real Cost for 1k cores

	Usage	Cost/day	x30x20
vCPU (vCPU*hours)	20046	$177	$106k
Memory (GB*hours)	47581	$56	$34k
Disk (GB*hours)	644898	$50	$30k
Network (GB)	559	$78	$47k
Other services		$30	$18k
Total		$391	$236k

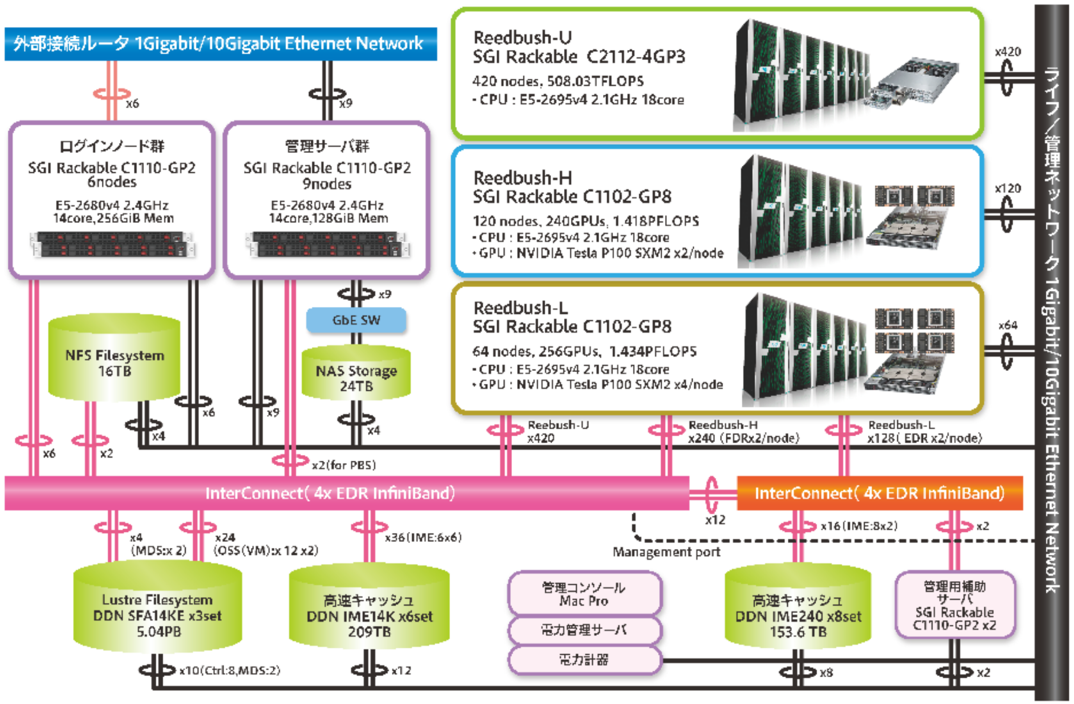
vCPU: 1vCPU instances max 200, 8 vCPUs instances max 100
Memory: 2.4 GB/vCPU
Disk: 50GB for 1vCPU instance, 150 GB for 8 vCPUs instance

Cost Estimation (20k cores/month)

Resource	Cost/month
vCPU x20k	$130k
3GB x20k	$42k
Local Disk 35GBx20k	$28k
Network GCP WN to ICEPP Storage 300 TB	$43k
Total	$243k

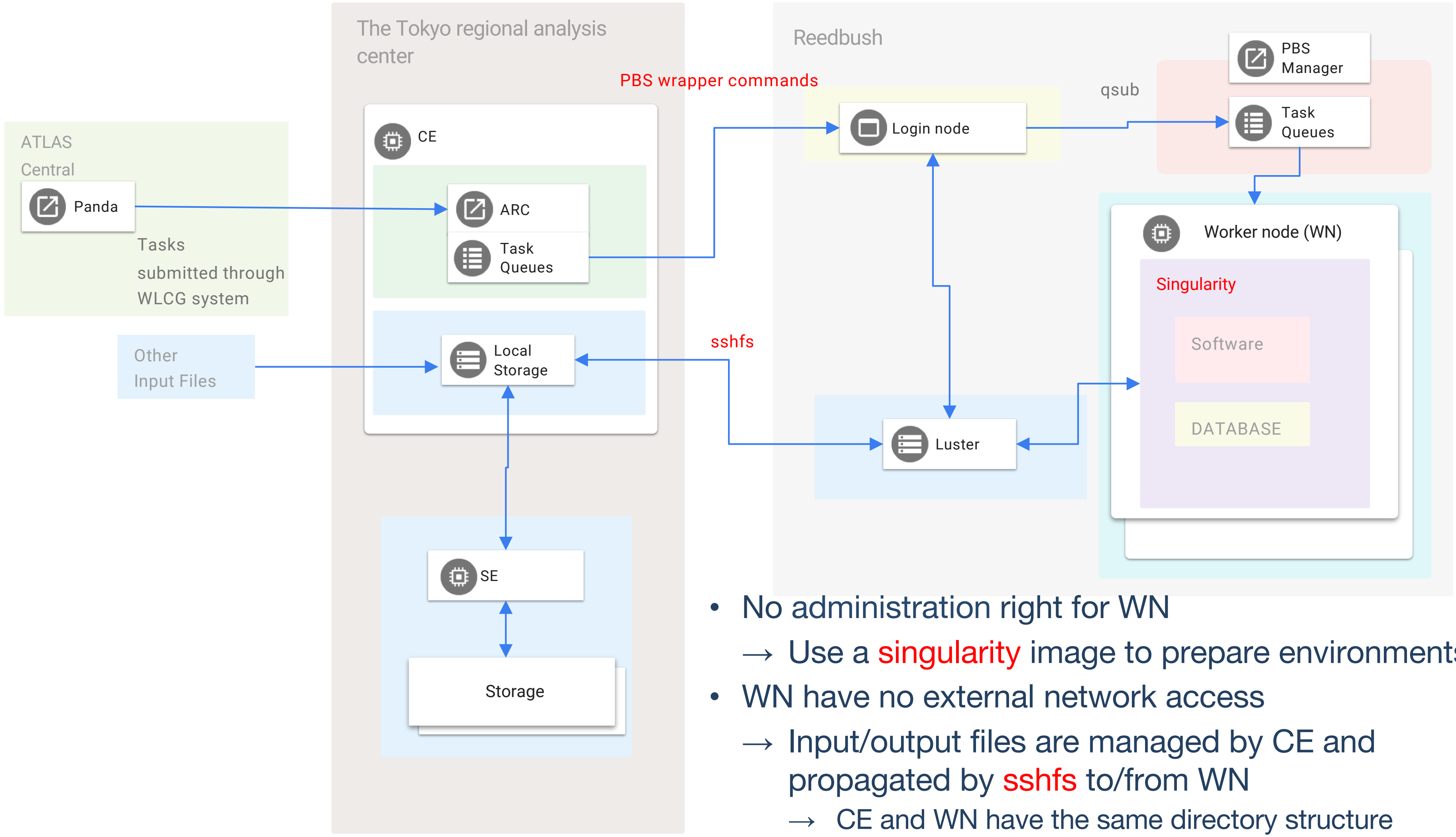
Reedbush

- Supercomputer system @Information Technology Center, The University of Tokyo
  →CPU:Intel Xeon (2CPUs/node (36cores/node))
  →GPU: NVIDIA Tesla P100
- CPU only nodes and GPU nodes
- OS: Read Hat Enterprise Linux 7



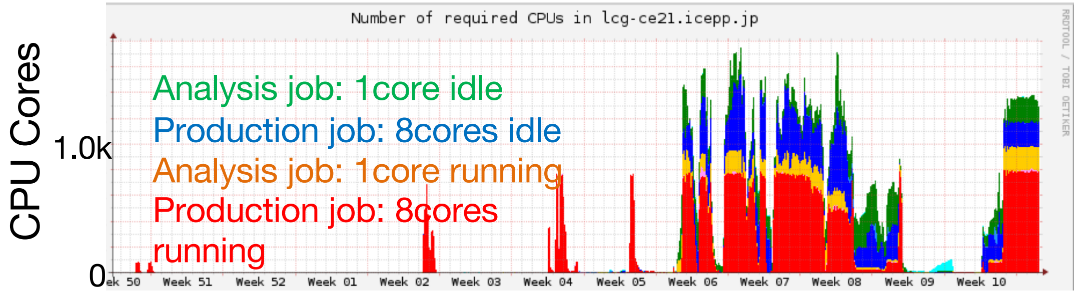
- PBS for the job management
- Lustre file system
- No external network access from each WN

System with Reedbush



- No administration right for WN
  → Use a singularity image to prepare environments
- WN have no external network access
  → Input/output files are managed by CE and propagated by sshfs to/from WN
    → CE and WN have the same directory structure
- Reedbush uses PBS for the job management
  → Available only on the login node
  → To manage jobs from CE, PBS wrapper commands are used
    → qsub:
    `ssh user@reedbush “cd $work_dir && qsub job.sh”`

ATLAS jobs on GCP and Reedbush

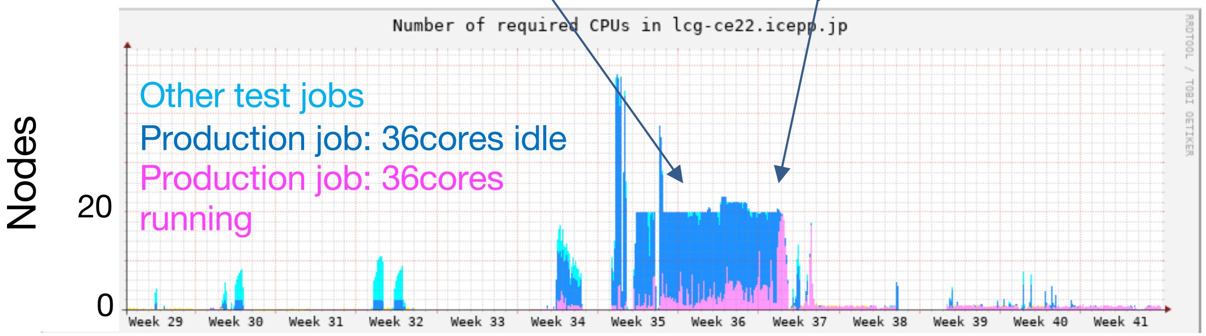


HTCondor status monitor for GCP

Max CPU Cores = 1k

Tested with the small queue (Only a few nodes are available)

Tested with the large queue



PBS status monitor for Reedbush
Max nodes = 20 (=720 CPU cores)

Summary

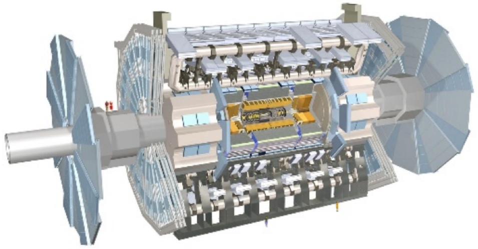
- The Tokyo regional analysis center introduced new systems using external resources
  →Commercial cloud: GCP
    → The hybrid system with cloud WN
    → GCPM has been developed to manage HTCondor job and instances of GCP
      → Can be used not only for WLCG, but also for any of HTCondor systems
  →HPC: Reedbush (@the Univ. Tokyo)
    → WNs of the system have no external network access
    → The system covers such a special situation by using singularity, sshfs and PBS wrapper commands are used
- ATLAS production jobs ran successfully

Future Plan

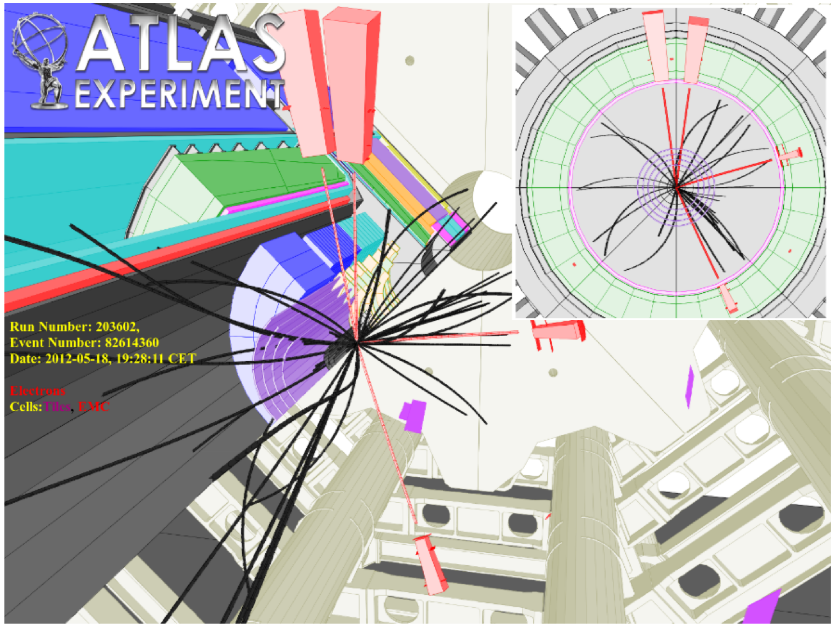
- Deploy other clouds (AWS, Azure, Oracle, …) and other HPCs in Japan
  → And other architectures (GPGPU, FPGA, …)
- Implement MPI for the HPC system

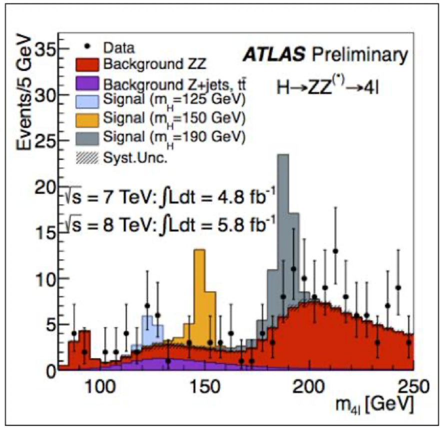
Backup

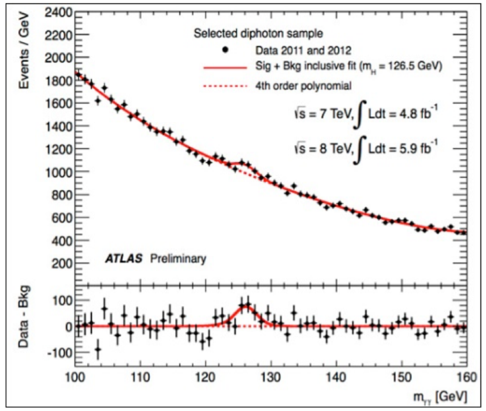
The ATLAS Experiment



Raw data: ~1GB/s

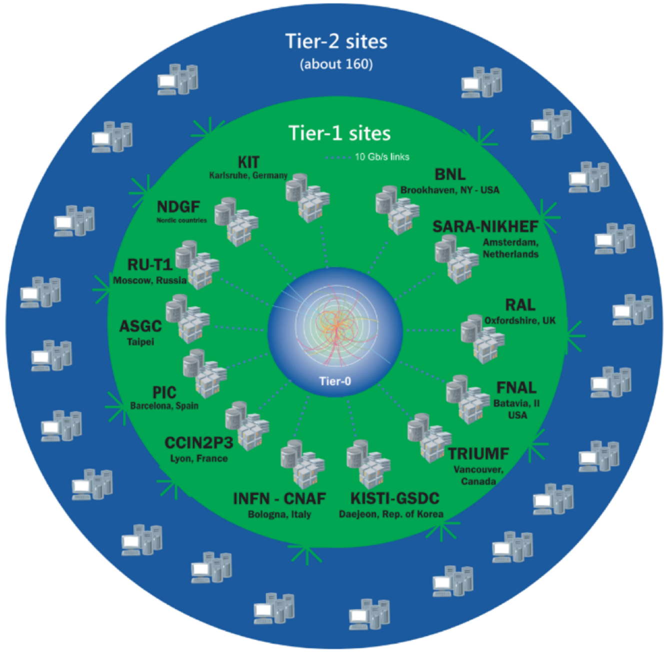






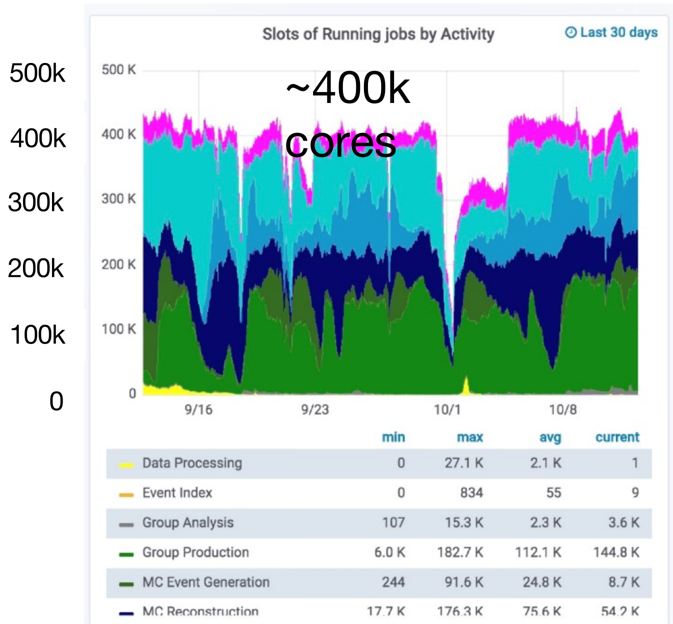
The Higgs Boson Discovery in 2012

Worldwide LHC Computing Grid (WLCG)



- A global computing collaboration for the LHC experiments

- The Tokyo regional analysis center is one of Tier2 for ATLAS



	min	max	avg	current
Data Processing	0	27.1 K	2.1 K	1
Event Index	0	834	55	9
Group Analysis	107	15.3 K	2.3 K	3.6 K
Group Production	6.0 K	182.7 K	112.1 K	144.8 K
MC Event Generation	244	91.6 K	24.8 K	8.7 K
MC Reconstruction	17.7 K	176.3 K	75.6 K	54.2 K

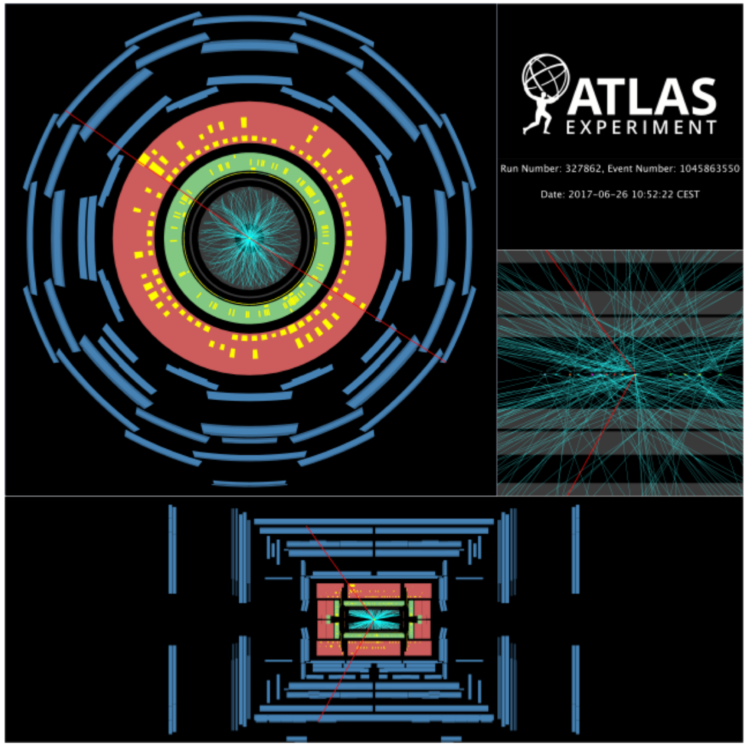
Number of cores used by ATLAS

Commercial Clouds and HPCs

- Commercial Clouds:
  →Many companies have introduced commercial clouds system
    → A lot of providers: AWS, Azure, Google Cloud Platform (GCP), IBM or Oracle…
  →A hybrid system of on-premises and cloud is also on trend
  →A hybrid system using GCP was constructed

- High Performance Computing (HPC):
  →HEP experiments have used "high throughput computing" (HTC) system
    → WLCG
  →But CPU clock did not get faster recently
  →HPC developments are strongly pushed in many countries
    → Need to adopt to these different architectures, MPI and special environments to obtain further computing power
  →Reedbush (HPC@The Univ. Tokyo) was used
    → Intel Xeon CPU (36cores/node)

Data Size



- LHC provides
  →40MHz proton-proton collision
- ATLAS filters events: 1kHz
- Raw event size: 1MB/events
  →1GB/seconds
- 150 days data taking/year:
  →10PB/year

- Current total data size: 200PB

(including Reconstructed data, Monte Carlo simulation)

Commercial Cloud

- Google Cloud Platform (GCP)
  →Number of vCPU, Memory are customizable
  →CPU is almost uniform:
    → At TOKYO region, only Intel Broadwell (2.20GHz) or Skylake (2.00GHZ) can be selected (they show almost same performances)
  →Hyper threading on
- Amazon Web Service (AWS)
  →Different types (CPU/Memory) of machines are available
  →Hyper threading on
  →HTCondor supports AWS resource management from 8.8
- Microsoft Azure
  →Different types (CPU/Memory) of machines are available
  →Hyper threading off machines are available

Performance Comparison

System	Hyper Threading	Core(vCPU)	Memory	CPU	HEPSPEC/ core	ATLAS simulation 1000events (hours)	Walltime*cores/Events
ICEPP local system	Off	32	96GiB	Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz	18.97	(8core job) 5.19	0.042
Google Cloud Platform	On	8	24GiB	Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz	12.62	(8core job) 9.27	0.074
Reedbush	Off	36	256GB	Intel(R) Xeon(R) CPU E5-2695 v4 @ 2.10GHz	16.78	(36 core job) 1.1	0.040

HEPSPEC (06): Benchmark for HEP

- The ATLAS production jobs can run with multi-processing mode
  →Normally 8 cores are used at WLCG sites
  →Will be multi-threading
- All GCP's instances are set as hyper-threading on
  →~half performance of other systems
- Reedbush nodes have 36 cores
  →Each job occupies all cores in the node: Run 36 processes mode

Google Computing Element

- **HT On**
  - → All Google Computing Element (GCE) at GCP are HT On
  - → TOKYO system is HT off

System	Core(vCPU)	CPU	SPECInt/core	HEPSPEC	ATLAS simulation 1000events (hours)
TOKYO system: HT off	32	Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz	46.25	18.97	5.19
TOKYO system: HT on	64	Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz	N/A	11.58	8.64
GCE (Broadwell)	8	Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz	(39.75)	12.31	9.32
GCE (Broadwell)	1	Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz	(39.75)	22.73	N/A
GCE (Skylake)	8	Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz	(43.25)	12.62	9.27

- SPECInt (SPECint_rate2006):
  - Local system: Dell Inc. PowerEdge M640
  - GCE(Google Compute Engine)'s value were taken from Dell system with same corresponding CPU
    - GCE (Broadwell): Dell Inc PowerEdge R630
    - GCE (Skylake): Dell Inc. PowerEdge M640
- ATLAS simulation: Multi process job 8 processes
  - For 32 and 64 core machine, 4 and 8 parallel jobs were run to fill cores, respectively

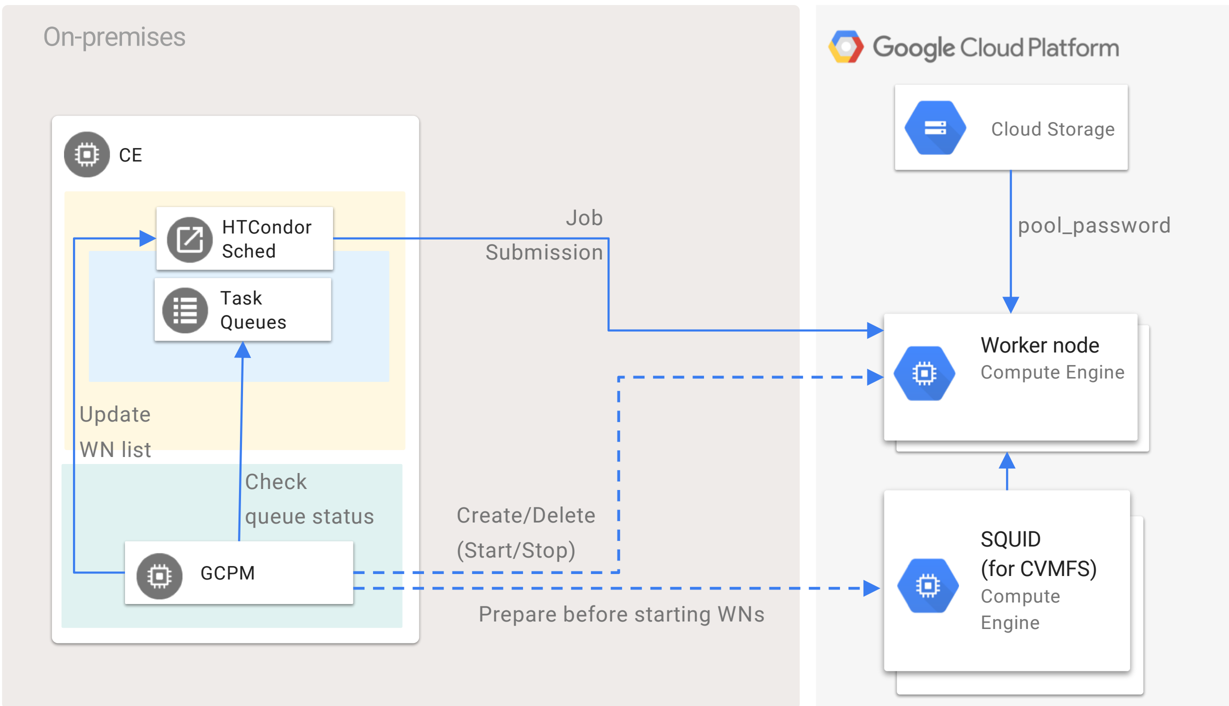
→ Broadwell and Skylake show similar specs
  - → Costs are same. But if instances are restricted to Skylake, instances will be preempted more
  - → Better not to restrict CPU generation for preemptible instances

→ GCE spec is ~half of TOKYO system

- **Preemptible Instance**
  - → Shut down every 24 hours
  - → Could be shut down before 24 hours depending on the system condition
  - → The cost is ~1/3

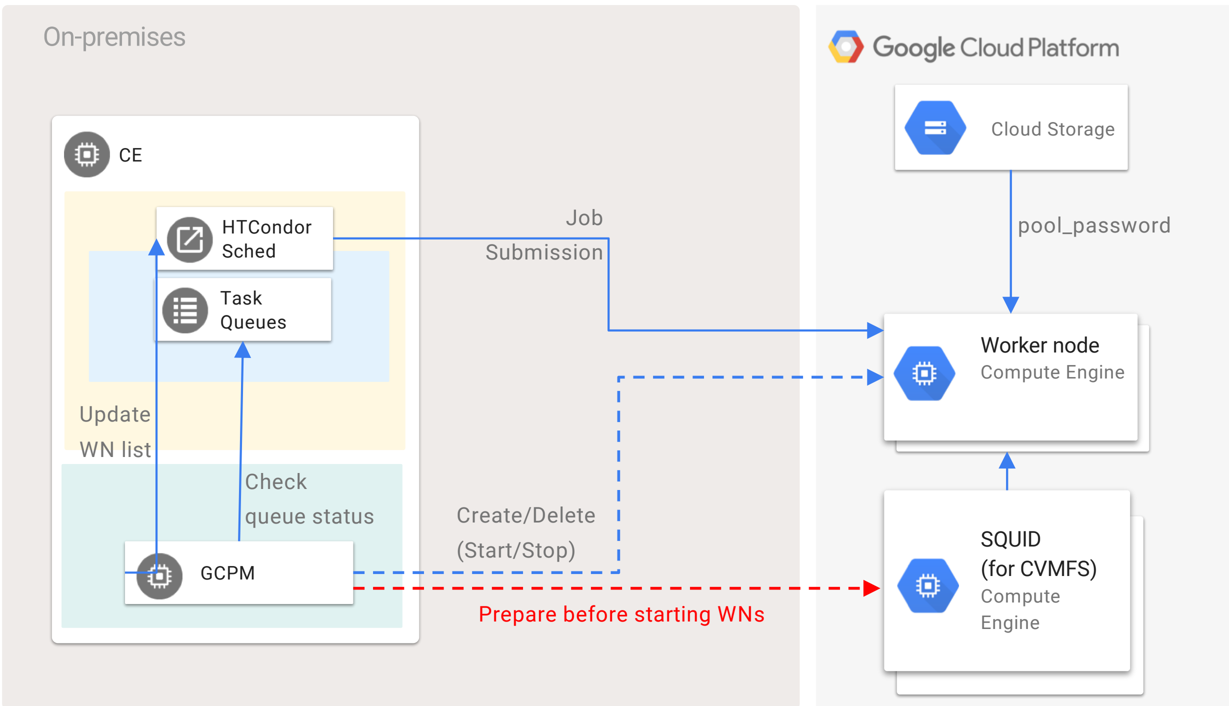
Google Cloud Platform Condor Pool Manager

- https://github.com/mickaneda/gcpm
  → Can be installed by pip:
    → ***$ pip install gcpm***
- Manage GCP resources and HTCondor's worker node list



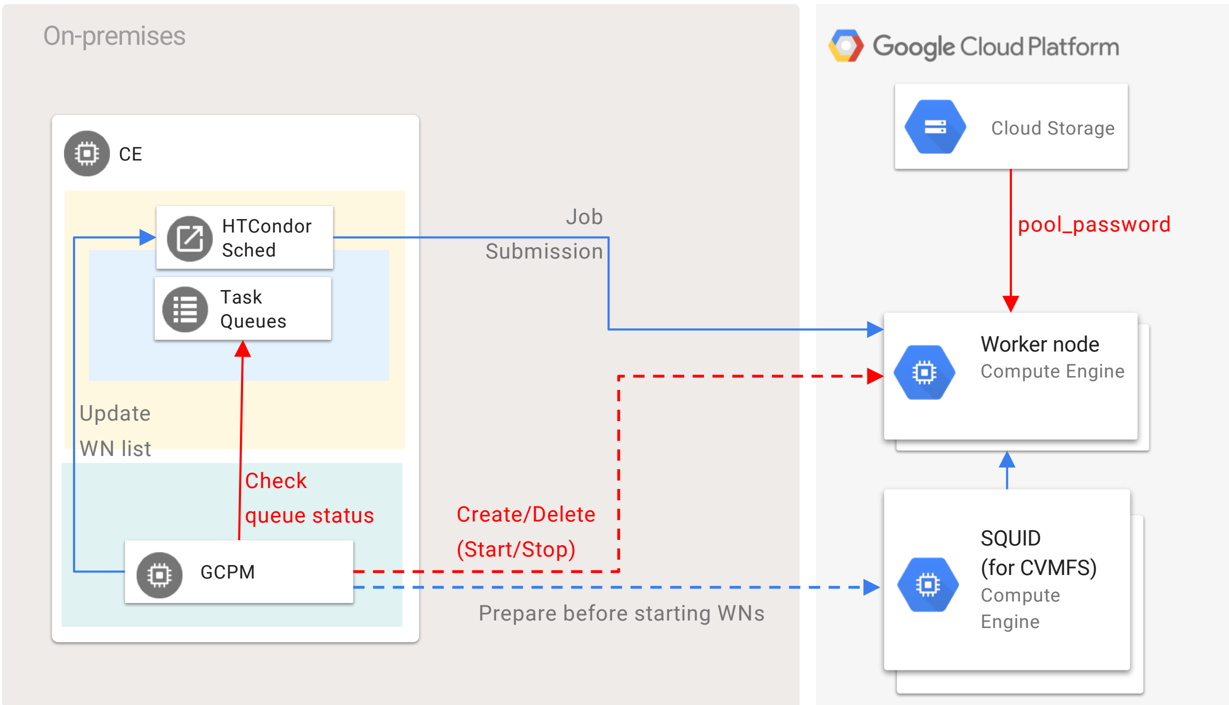
Google Cloud Platform Condor Pool Manager

- Run on HTCondor head machine
  →Prepare necessary machines before starting worker nodes
  →Create (start) new instance if idle jobs exist
  →Update WN list of HTCondor
  →Job submitted by HTCondor
  →Instance's HTCondor startd will be stopped at 10min after starting
    → ~ only 1 job runs on instance, and it is deleted by GCPM
      → Effective usage of preemptible machine



Google Cloud Platform Condor Pool Manager

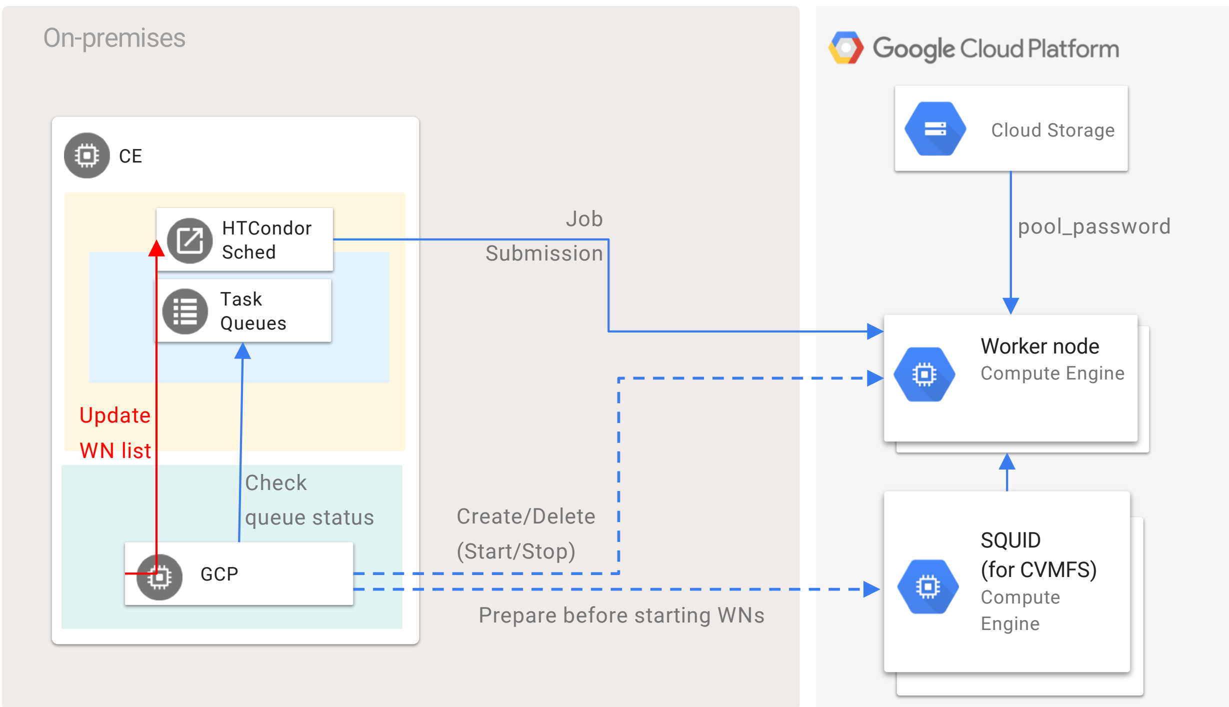
- Run on HTCondor head machine
  - → Prepare necessary machines before starting worker nodes
  - → Create (start) new instance if idle jobs exist
  - → Update WN list of HTCondor
  - → Job submitted by HTCondor
  - → Instance's HTCondor startd will be stopped at 10min after starting
    - → ~ only 1 job runs on instance, and it is deleted by GCPM
      - → Effective usage of preemptible machine



pool_password file for the authentication is taken from storage by startup script

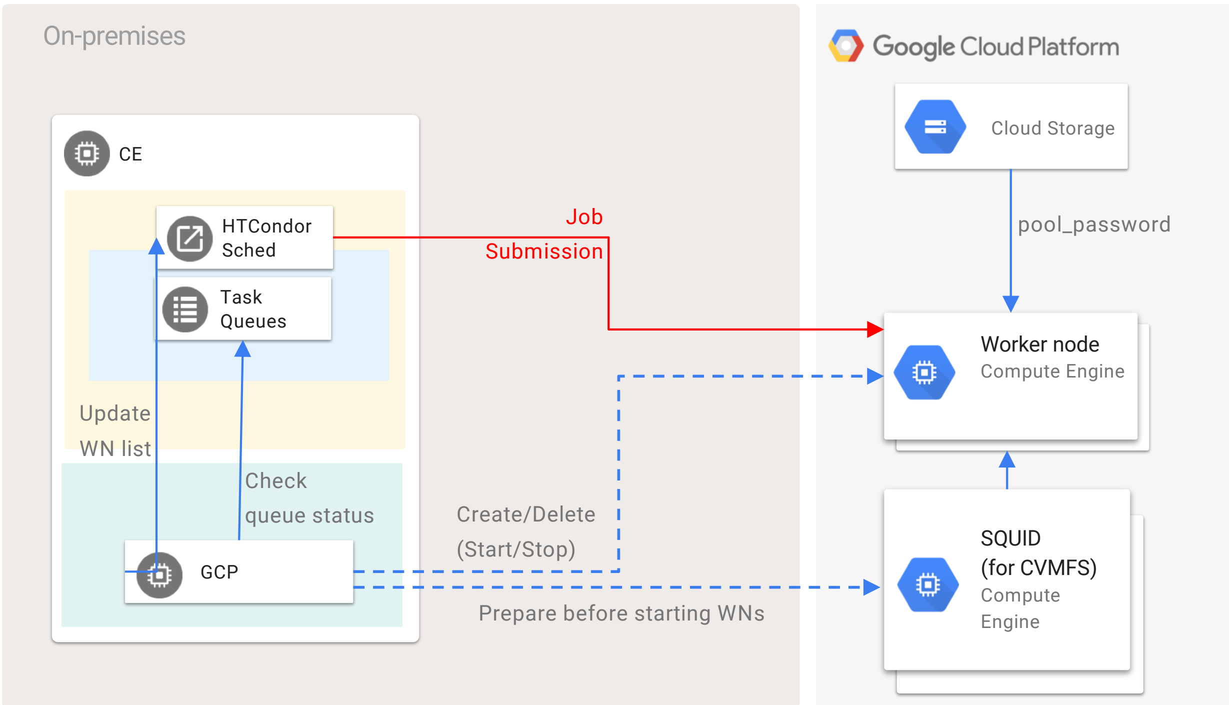
Google Cloud Platform Condor Pool Manager

- Run on HTCondor head machine
  - → Prepare necessary machines before starting worker nodes
  - → Create (start) new instance if idle jobs exist
  - → Update WN list of HTCondor
  - → Job submitted by HTCondor
  - → Instance's HTCondor startd will be stopped at 10min after starting
    - → ~ only 1 job runs on instance, and it is deleted by GCPM
      - → Effective usage of preemptible machine



Google Cloud Platform Condor Pool Manager

- Run on HTCondor head machine
  - → Prepare necessary machines before starting worker nodes
  - → Create (start) new instance if idle jobs exist
  - → Update WN list of HTCondor
  - → Job submitted by HTCondor
  - → Instance's HTCondor startd will be stopped at 10min after starting
    - → ~ only 1 job runs on instance, and it is deleted by GCPM
      - → Effective usage of preemptible machine



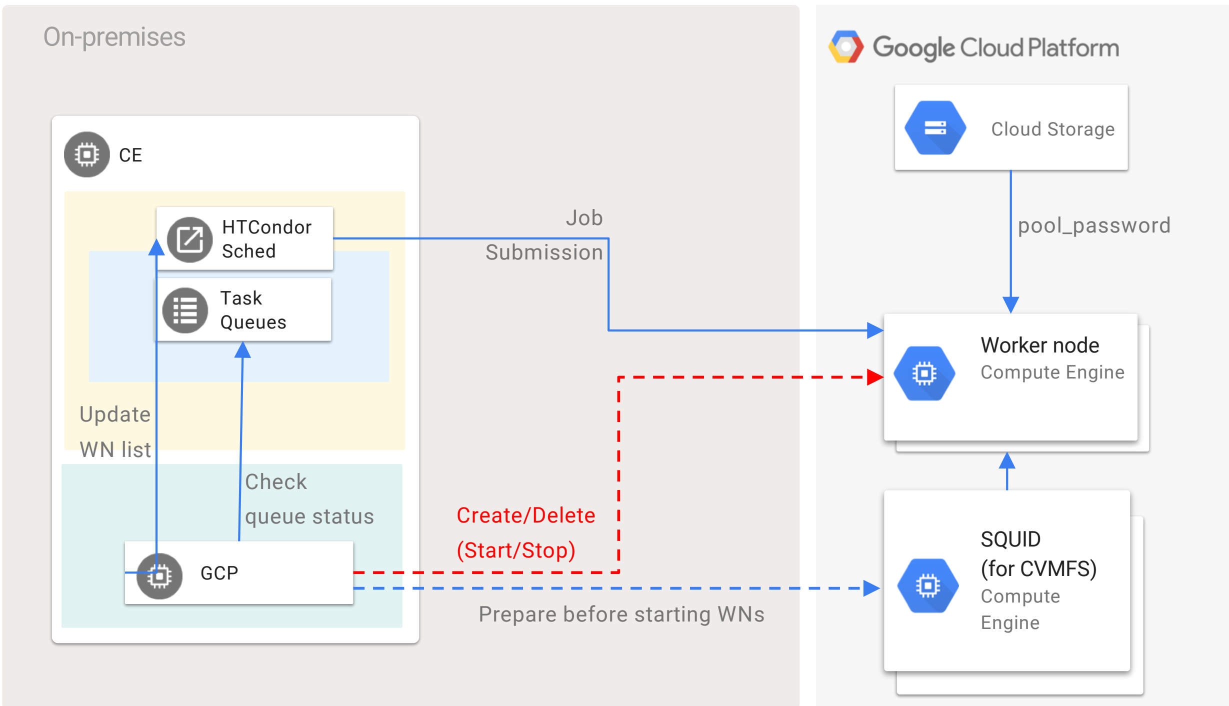
Google Cloud Platform Condor Pool Manager

- Set to execute `*condor_off -peaceful –startd*` after 10min by the startup script for GCE instance
- When a job finished, the instance is removed from `*condor_status*` list
- Then GCPM deletes (sotps) the instance

→ Instance's HTCondor startd will be stopped at 10min after starting

→ ~ only 1 job runs on instance, and it is deleted by GCPM

→ Effective usage of preemptible machine

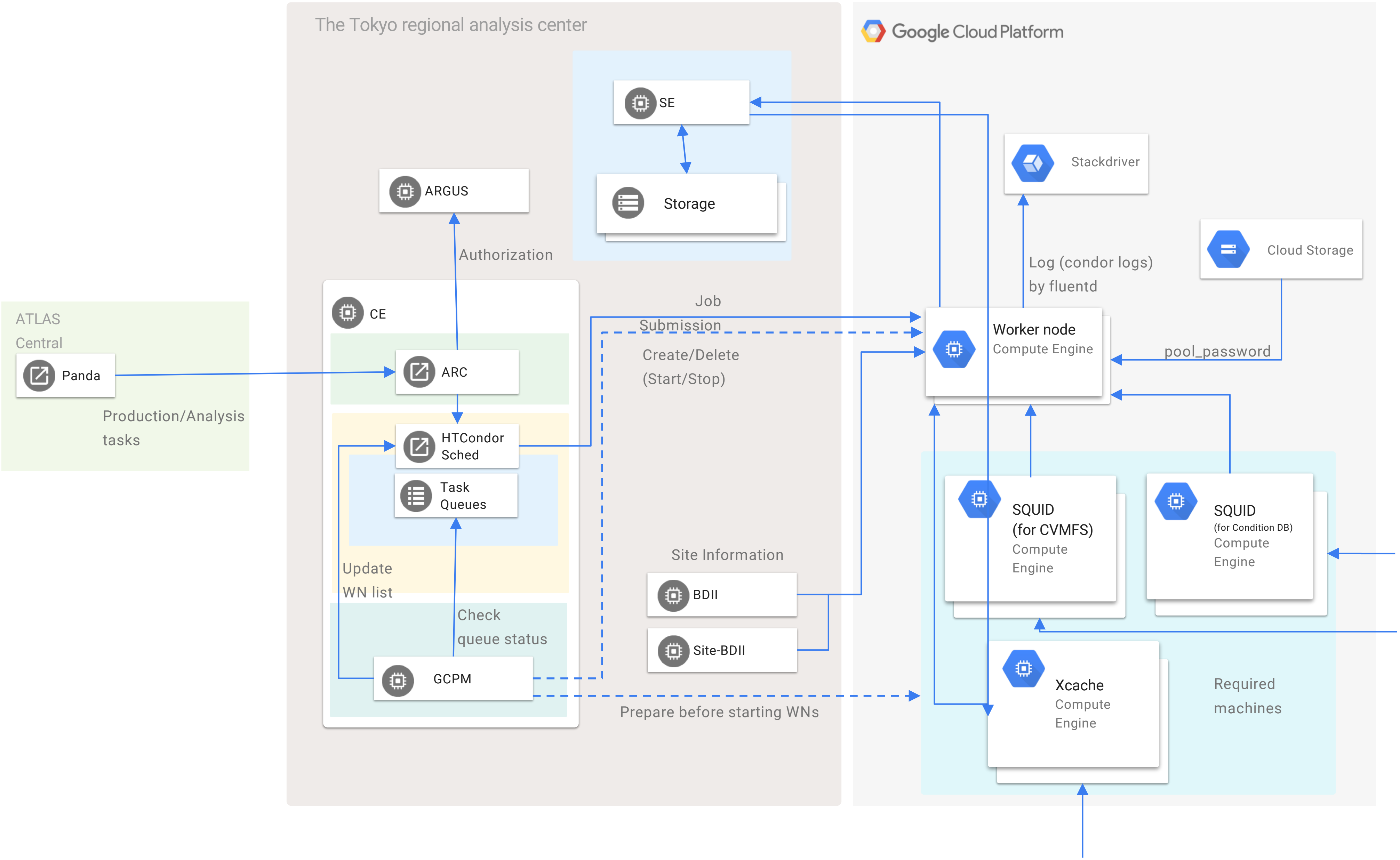


ARC CE Hacking

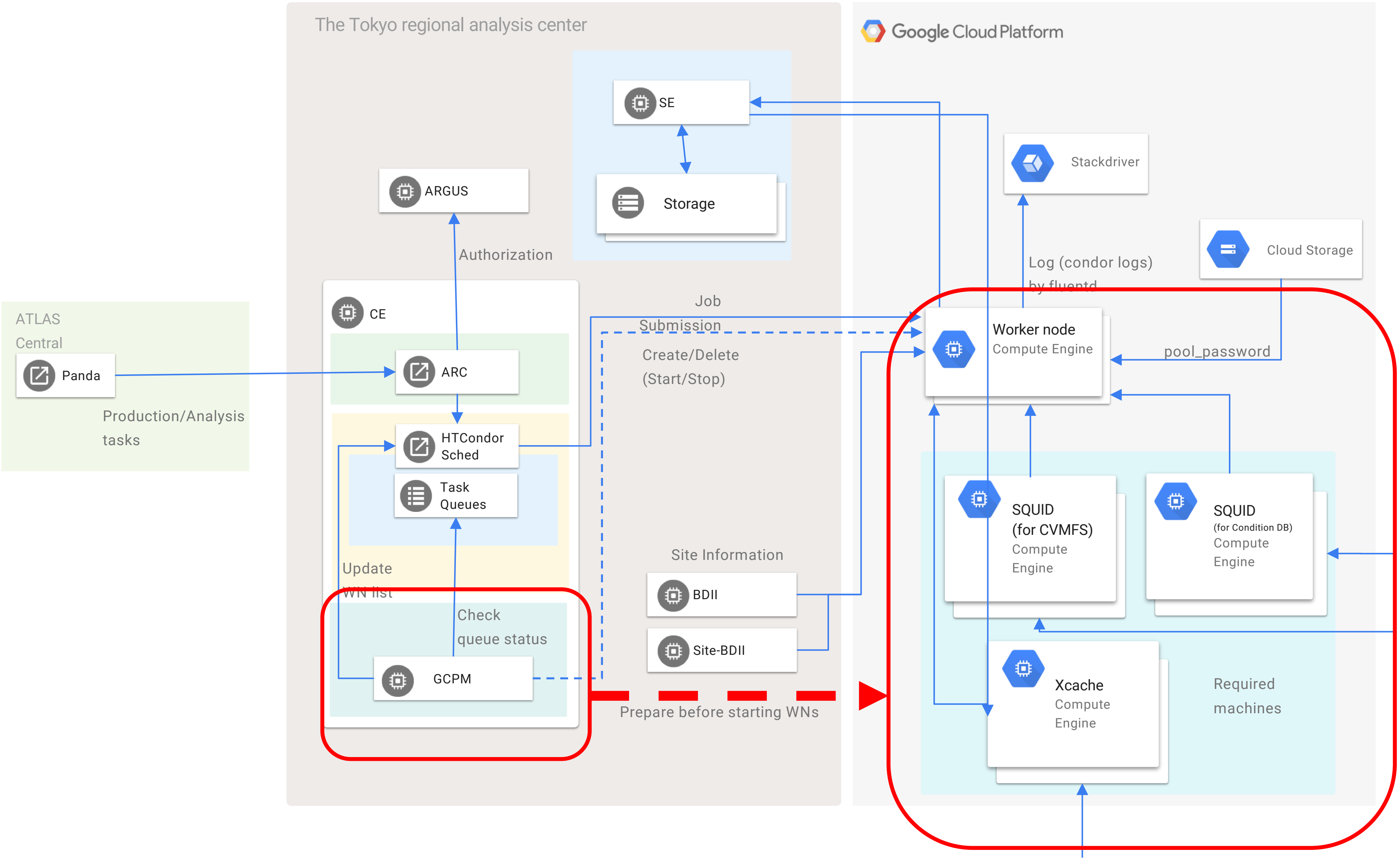
- ARC checks a number of available slots before submitting jobs
  → If a job specifies a number of CPUs and there are not enough slots, job submission fails
  → GCP pool has no slot at the start, jobs cannot be submitted
  → Hack /usr/share/arc/Condor.pm to return non-zero cpus if it is zero

```
#
# returns the total number of nodes in the cluster
#
sub condor_cluster_totalcpus() {
    # List all machines in the pool. Create a hash specifying the
TotalCpus
    # for each machine.
    my %machines;
    $machines{$$_{machine}} = $$_{totalcpus} for @allnodedata;

    my $totalcpus = 0;
    for (keys %machines) {
        $totalcpus += $machines{$_};
    }

    # Give non-zero cpus for dynamic pool
    $totalcpus ||= 100;
    return $totalcpus;
}
```

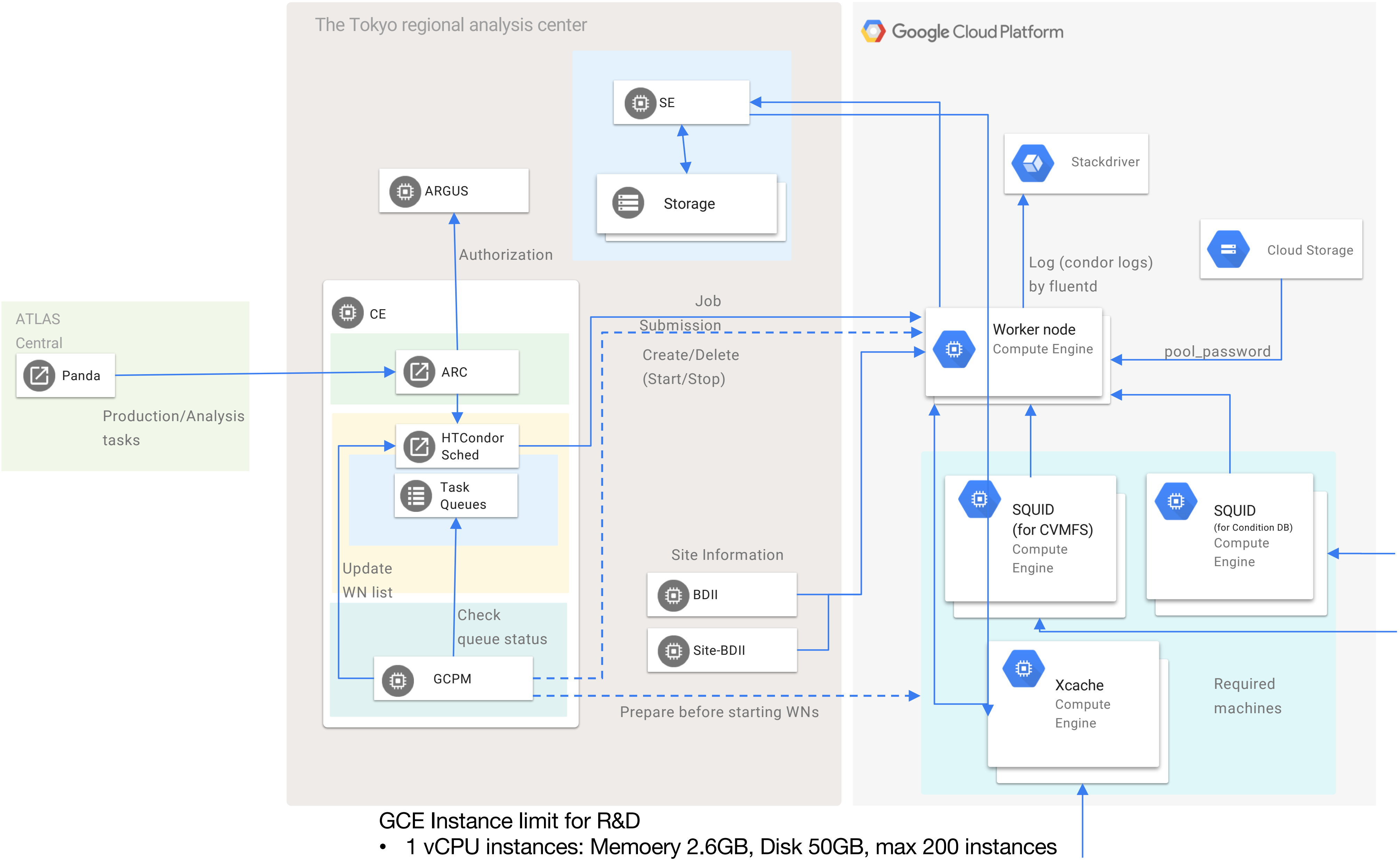
System for R&D



System for R&D



System for R&D



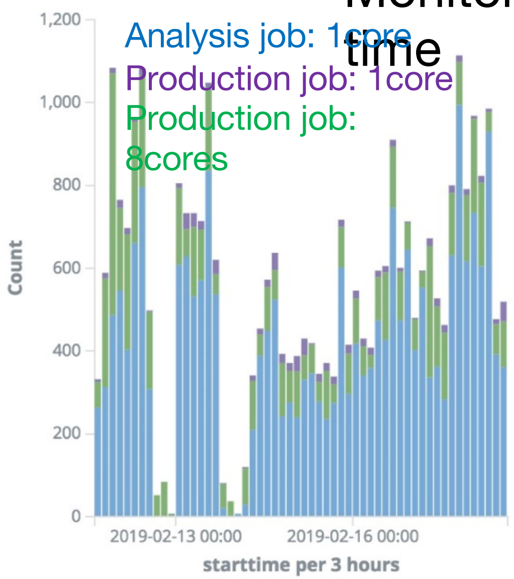
GCE Instance limit for R&D

- 1 vCPU instances: Memoery 2.6GB, Disk 50GB, max 200 instances
- 8 vCPU instances: Memory 19.2GB, Disk 150GB, max 100 instances

→ Total vCPU max: 1000

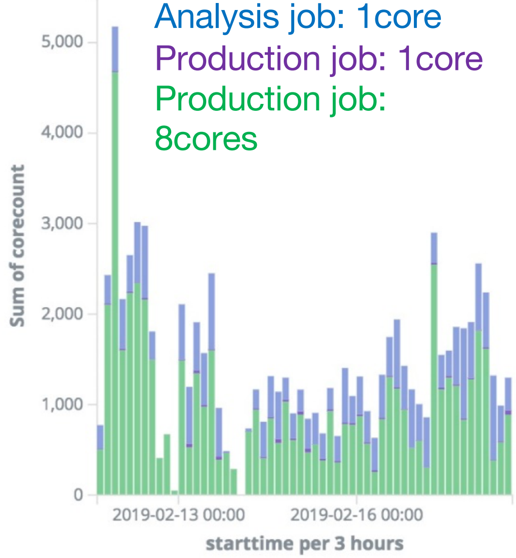
Jobs Running on GCP

Monitors of job starting time



Analysis job: 1core
Production job: 1core
Production job: 8cores

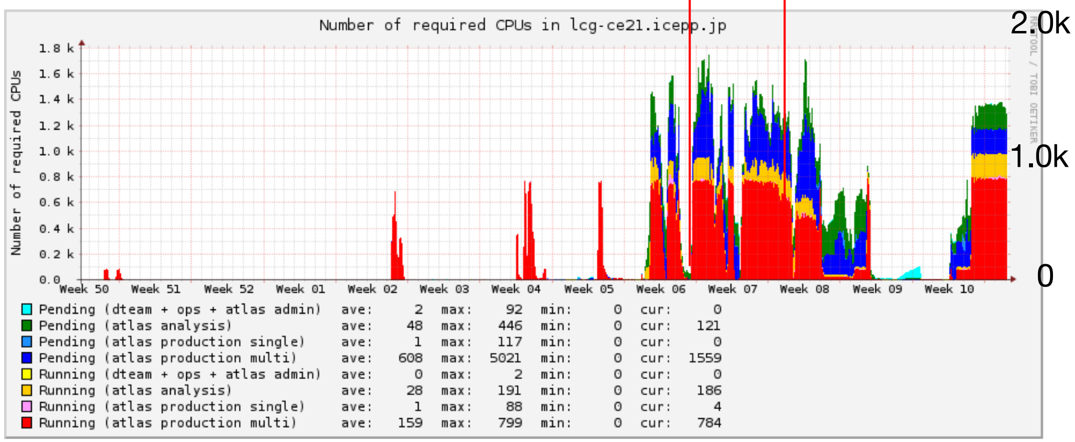
Number of jobs



Analysis job: 1core
Production job: 1core
Production job: 8cores

Number of vCPUs

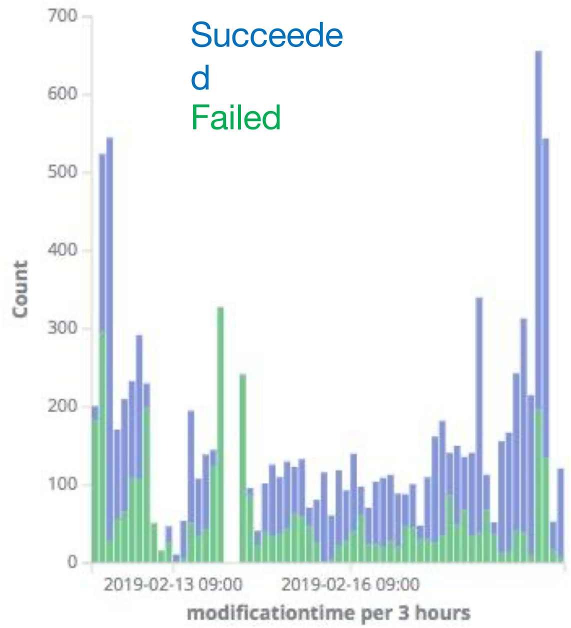
vCPUs



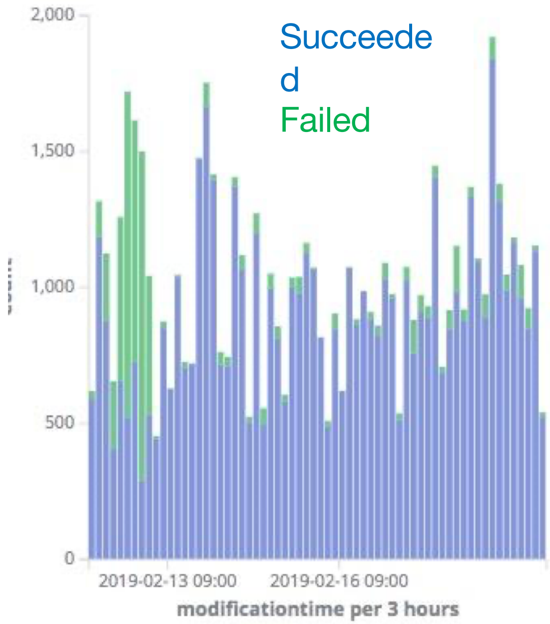
HTCondor status monitor

Analysis job: 1core idle
Production job: 8cores idle
Analysis job: 1core running
Production job: 8cores running

Failure Rate (Production Jobs)



GCP Worker Nodes
(Production Job)

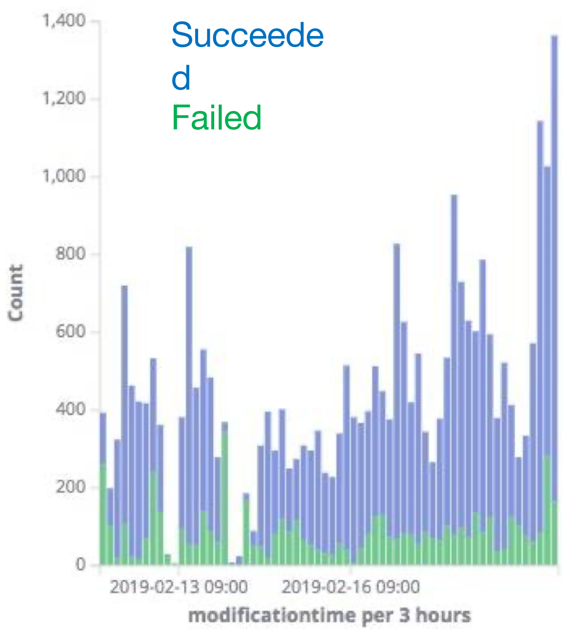


ICEPP Worker Nodes
(Production Job)

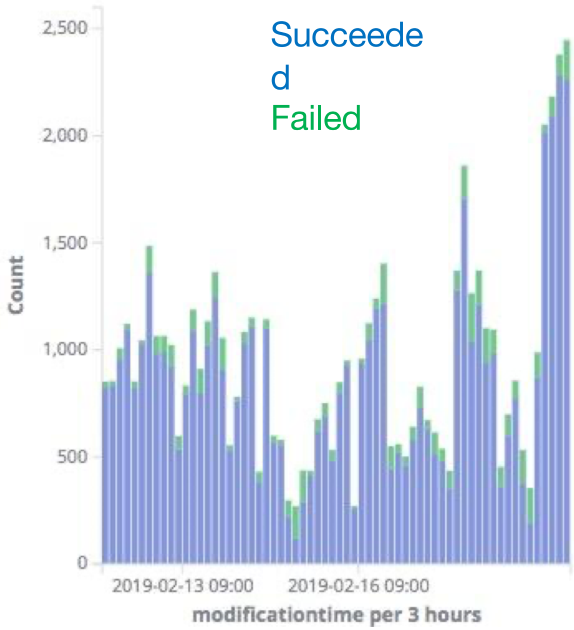
Job Type	Error rate
GCP Production (Preemptible)	35%
GCP Production (Non-Preemptible)	6%
Local Production	11%

Mainly 8 core jobs, long jobs (~10 hours/job)

Failure Rate (Analysis Jobs)



GCP Worker Nodes
(Analysis Job)



ICEPP Worker Nodes
(Analysis Job)

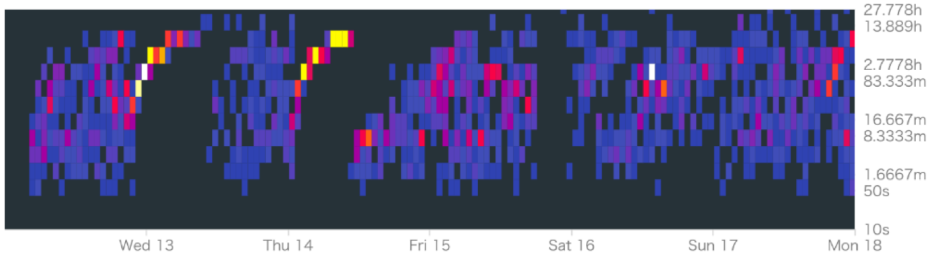
Job Type	Error rate
GCP Analysis (Preemptible)	19%
GCP Analysis (Non-Preemptible)	14%
Local Analysis	8%

Only 1 core job, shorter jobs

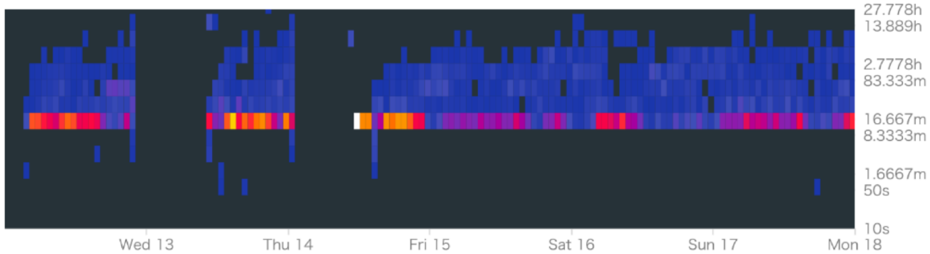
Preemption

1 core instances

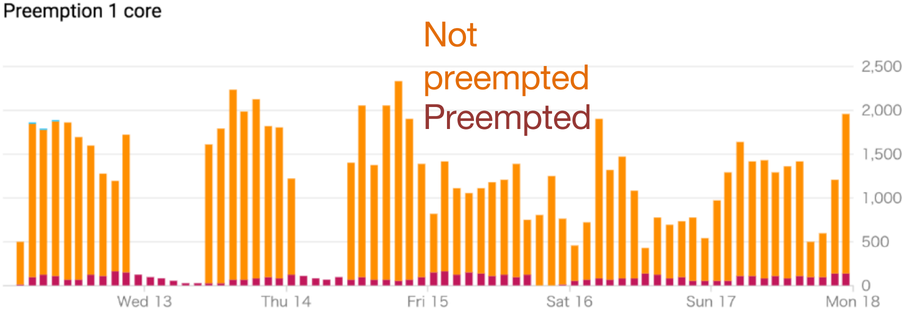
Uptime: 1 core, Preempted



Uptime: 1 core, Not preempted

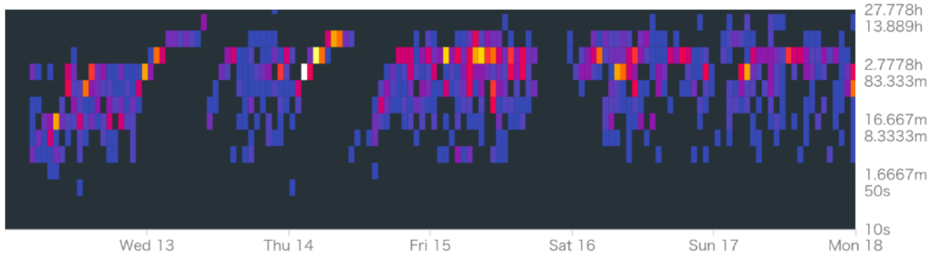


Preemption 1 core

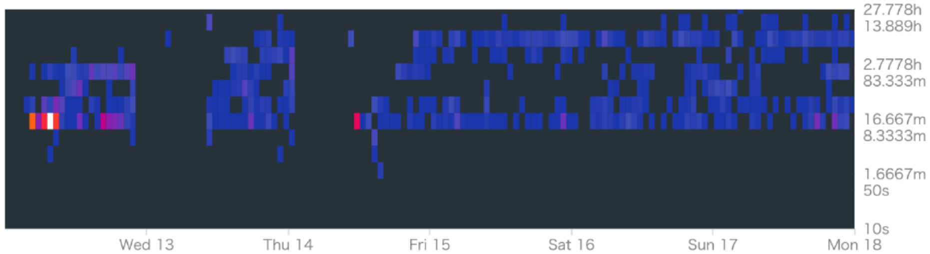


8 core instances

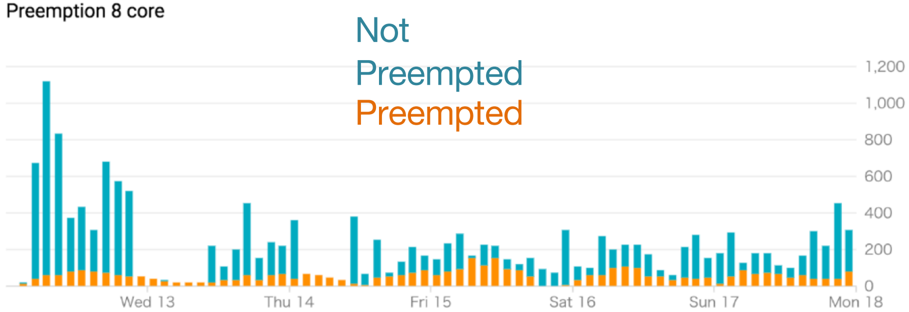
Uptime: 8 cores, Preempted



Uptime: 8 cores, Not preempted

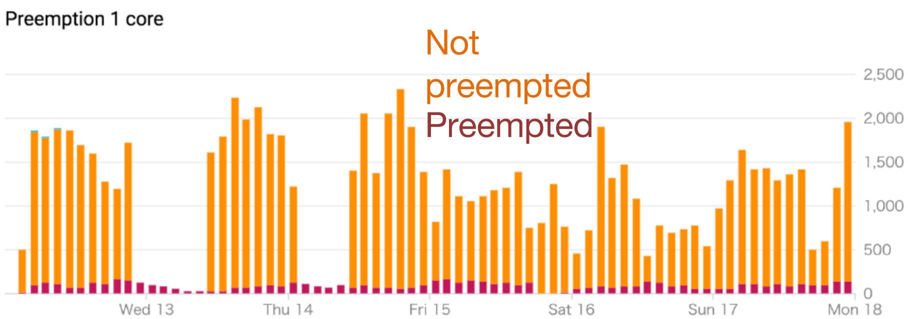


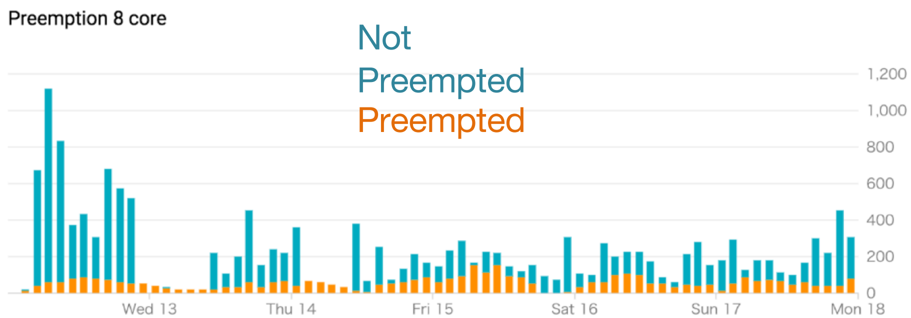
Preemption 8 core



Preemption v.s. Failure jobs

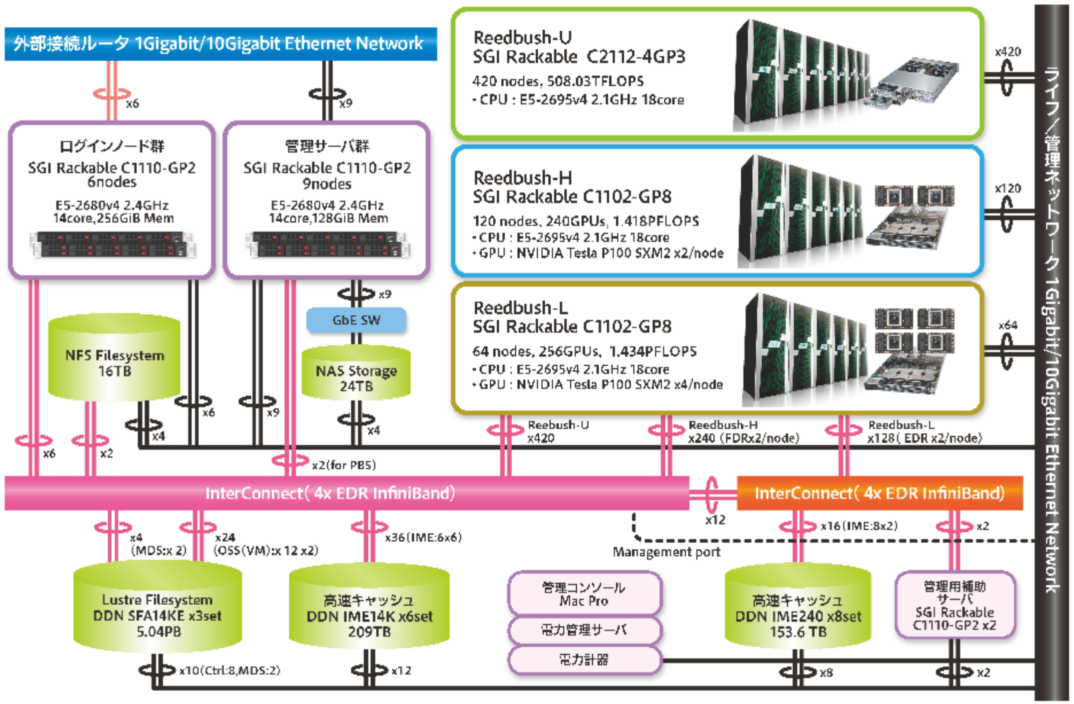
- 5~30 % instances were shut down by Preemption
  →Made failure jobs
- Typically shut down around 3~10 hours
  →Some instances were shutdown before 1 hours running
- More preemptions in 8 core jobs (production: reco/sim) because job running times are longer





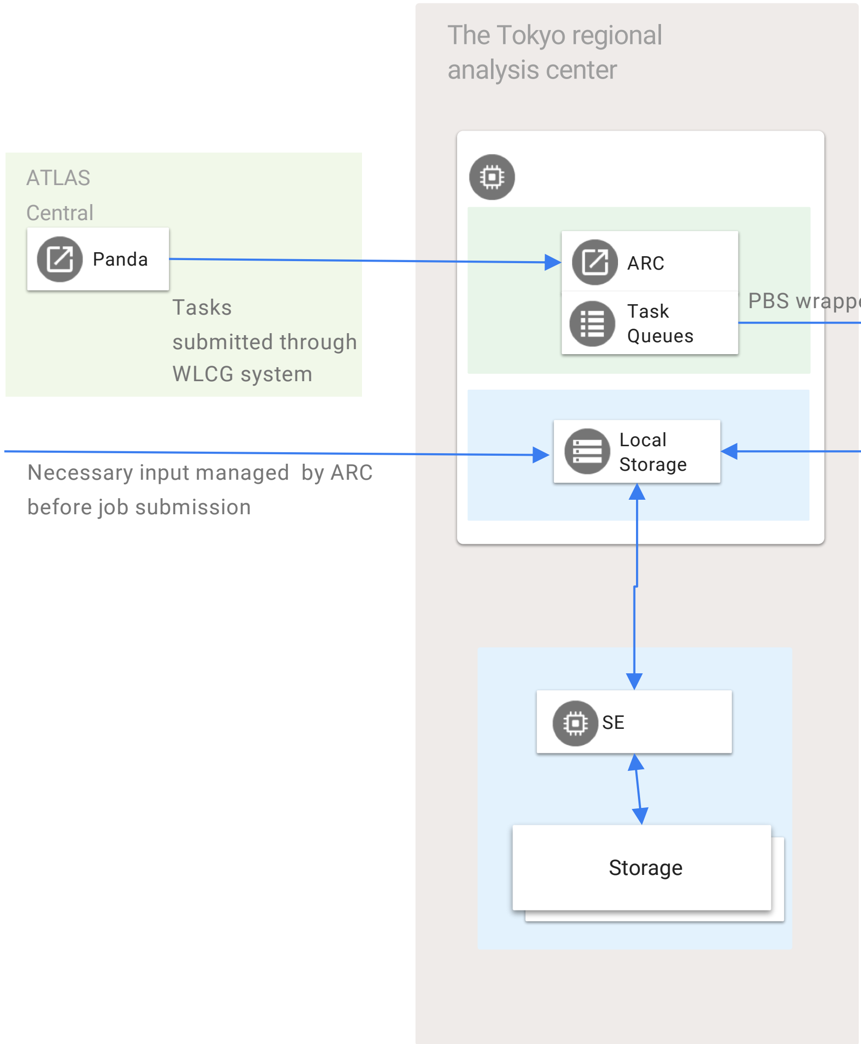
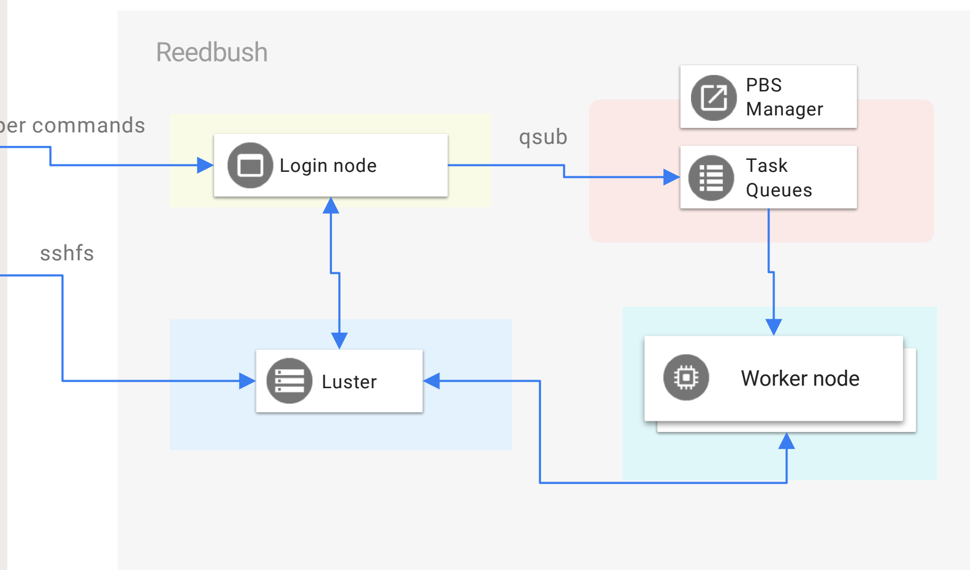
Reedbush

- Supercomputer system @Information Technology Center, The University of Tokyo
  →CPU:Intel Xeon (2CPUs/node (36cores/node))
  →GPU: NVIDIA Tesla P100
- CPU only nodes and GPU nodes
- OS: Read Hat Enterprise Linux 7

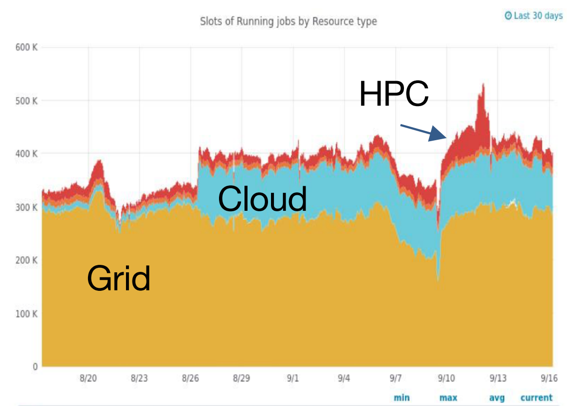


- PBS for the job management
- Lustre file system
- No external network access from each WN

Reedbush System

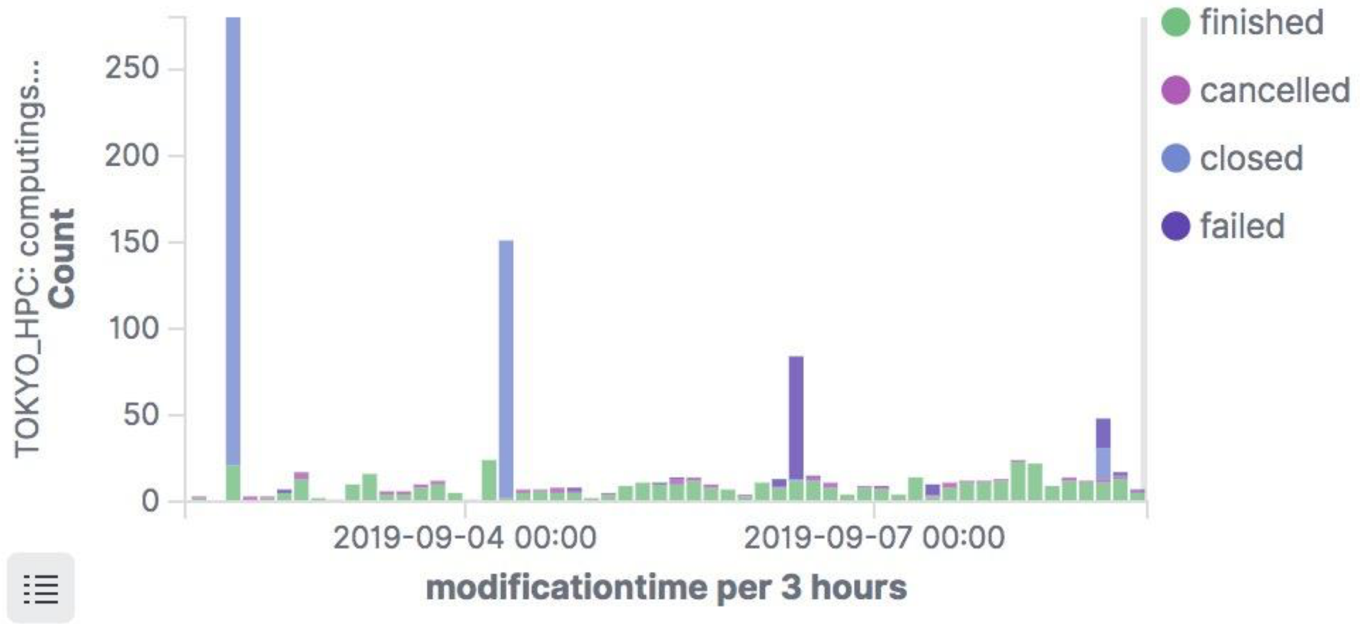



4484564357 Attempt 0 of 0	gangarbt		Sim_tf.py	finished	2019-09-15 06:59:30	0:0:07:16	0:0:12:00	2019-09-15 07:34:24	FR TOKYO_HPC test HC.Blacklist.set.test	10000	2 (1)
	Job name: 50691303-db54-485b-9b36-6e2cce1afb8f_79286#0										
	Datasets: **In:** hc_test:mc15_13TeV.361106.PowhegPythia8EvtGen_AZNLOCTEQ6L1_Zee.evgen.EVNT.e3601_tid04972714_00 || Rucio link **Out:** hc_test.gangarbt.hc20145990.tid957.TOKYO_HPC.166										

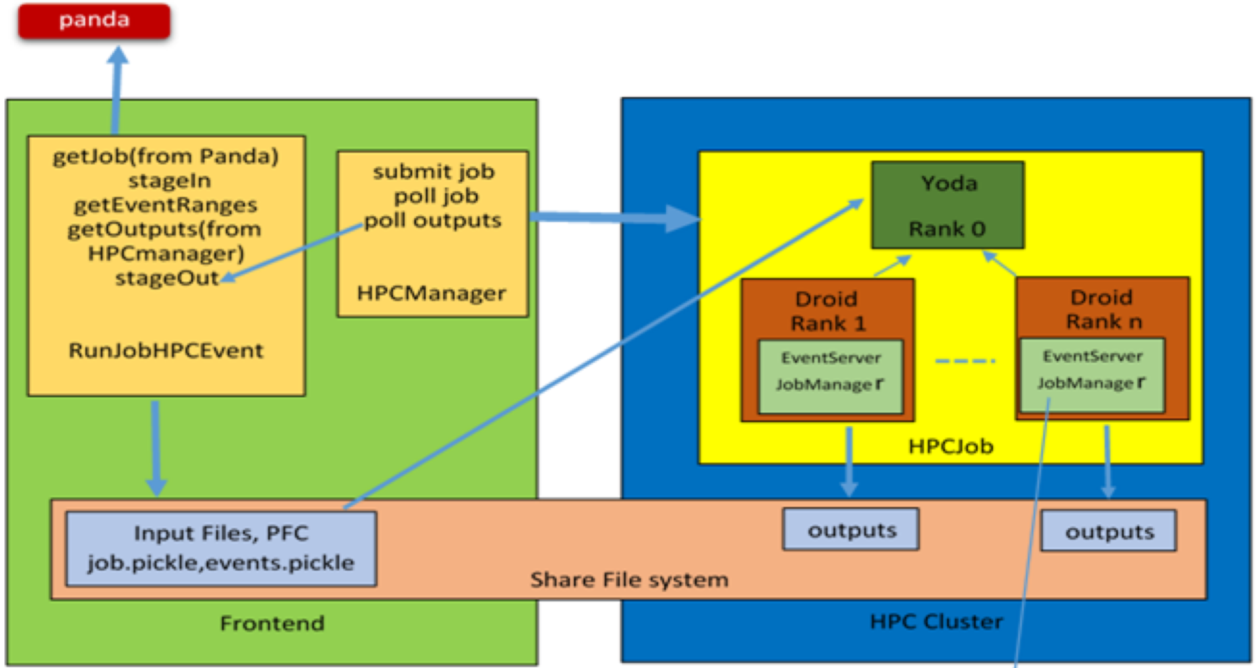


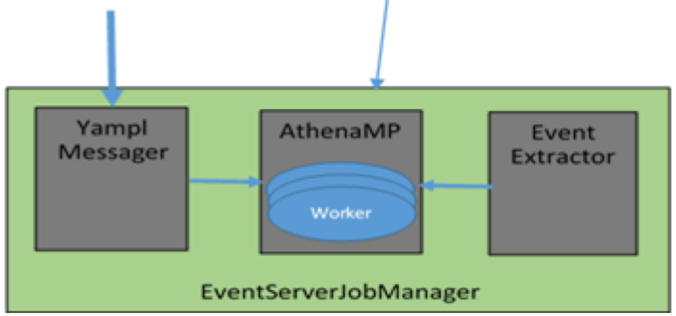
Usage of HPC by ATLAS

Reedbush Failure rate



MPI Job





- Yoda:
  - →A tool for Message Passing Inerface (MPI) developed by ATLAS
  - →We will try to deploy the tool in our system

Cost Comparison

System	Cost for 10k cores/Month
On-premises	$200k
Reedbush	$40k
Google Cloud Platform	$250k

- On-premises:
  - → Total server cost of 10k cpu cores, 16PB storage (Dell)/3 years
  - → Additional cost: infrastructure, maintenance
- Reedbush:
  - → Price as a user
  - → Non-university groups also can apply to use the system (price: x1.2)
  - → Only limited number of resources
    - → Currently max number of nodes is ~ 20 (~700cores)
  - → Additional cost: on-premises storage and other service components
  - → Cost of Reedbush itself is ~50k/month for 10k cores
- GCP:
  - → Hyper Threading On: Need double number of CPU cores (calculated by assuming 20k cores)
  - → Reduced cost by using preemptible instances
  - → Including network cost
  - → Additional cost: on-premises storage and other service components