Using dHTC to Extend the HTC Cluster at IHEP

JIANG, Xiaowei
On behalf of Computing Team of IHEP CC
HTCondor Week 2022
2022/05/24

OUTLINE

- Background
- Status of Local HTC Cluster
- dHTC design and implement at IHEP
- Certification and Authentication
- User Interface and Job Environment
- Summary and Plan

Motivations

- IHEP is hosting or attending in >15 experiments around the field of high energy physics
  - Most of experiments do the data processing in the local cluster
  - Some domestic collaboration members are running small cluster
- Extending the htc local cluster is the better way to find more resources
  - The data processing has run for a long time and bounded with the local cluster
  - Most of users have been used to do data analysis in a local cluster
    - Closer to use a personal PC and would not like to change any habit
    - Access data and software in anytime and anywhere by the shared file system
  - Some experiments do not have official experiment software
    - Users write their own data processing program in different ways

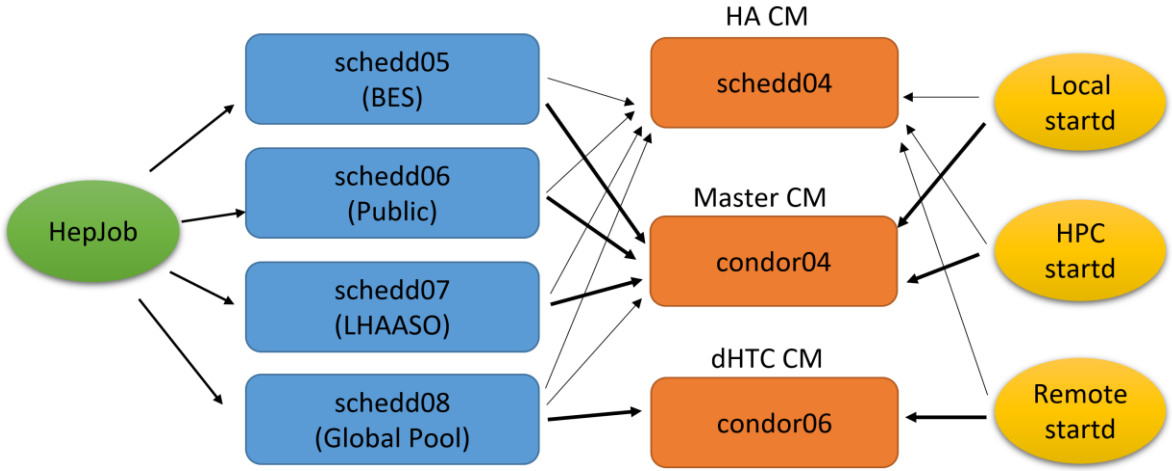
Clusters and Sites

- More and more sites are built for scientific research in domestic China
- Branches of IHEP
  - Long-term resources provided，regarded as part of htc pool
  - Limitation on network bandwidth and storage
- Collaboration members
  - Most are edge sites (not quite big)
  - Without a pledged storage
- Cooperation institutes (universities)
  - All are dynamic (opportunistic)
  - Nothing is pledged
- Diverse requirements according to the conditions
  - CPU & GPU & ARM & Cloud
  - Structures of computing system are different
  - Policies of job are different

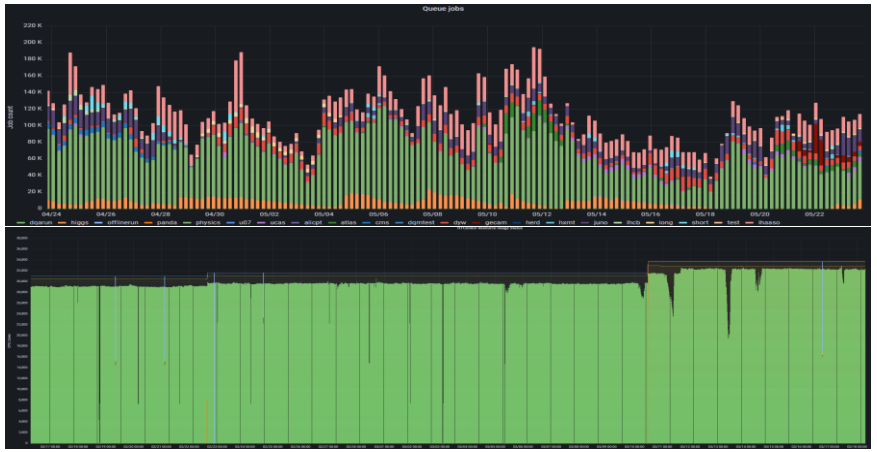
	CPU Cores	Disk Storage	Tape Storage	Network
IHEP-CC	46000	Lustre, >20 PB EOS, ~20 PB	30 PB	100G/10G/1G
IHEP-Dongguan Guangdong Province	~6000	Lustre 3 PB	1 PB	10G-100G to IHEP
IATAMS Shandong Province	~15000	Lustre 2.5 PB		1G-10G to IHEP
IHEP-Daocheng Sichuan Province	~3400	EOS 1.6 PB		2G to IHEP
USTC, Anhui Province	~2200	Lustre 4 PB		Internet
LZU Gansu Province	1636			Internet
PKU Beijing	288			Internet
Shandong Uni. Shandong Province	720			Internet
Dongguan DC Guangdong Province	~30000	6 PB		10G to Dongguan DC
IHEP Huairou Branch (ready by 2025)	~10000	Lustre 30 PB		100G to IHEP

Status of IHEP Local HTC Cluster

- HTCondor Cluster is getting bigger
  - 4 SchedDs: mapping by specific groups
    - Maximum 10k running jobs for each schedd
  - 3 CMs: Main CM&HA CM
  - CPU: >35k cores
    - Most are single core slots

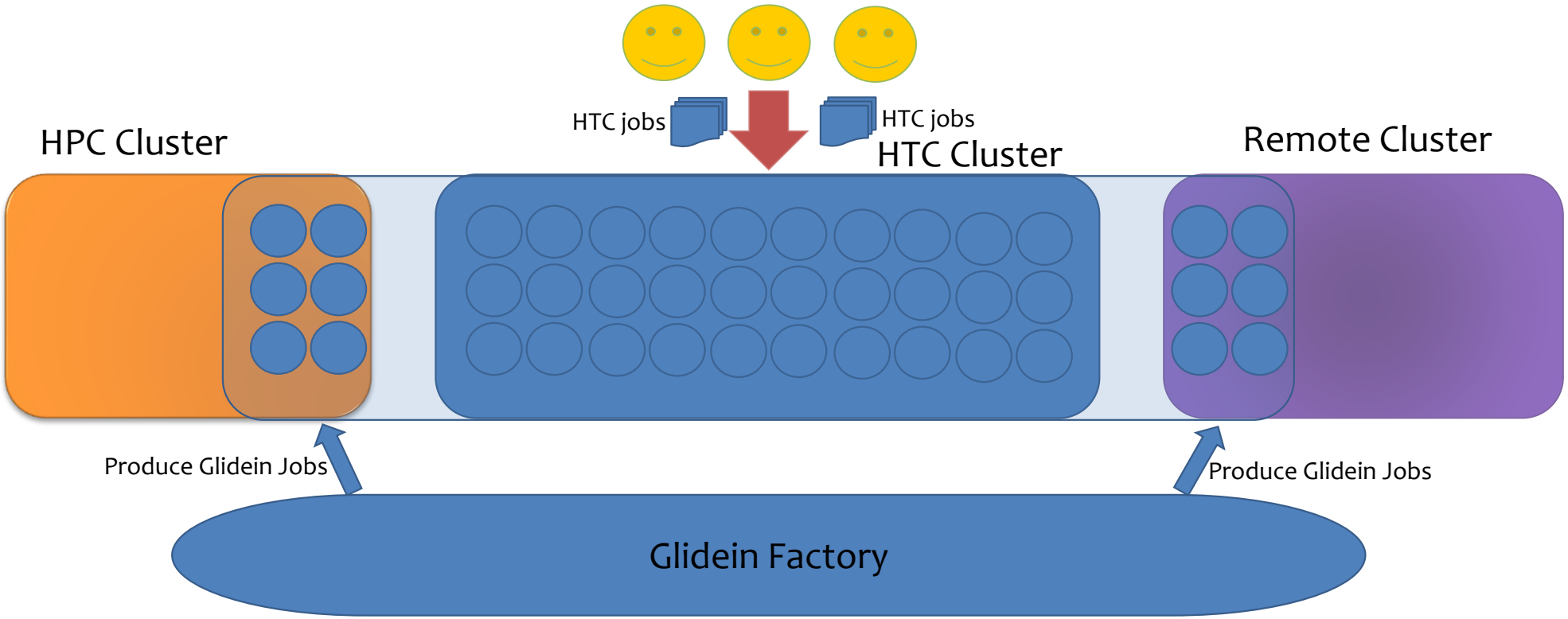


- Finding more resources for the local cluster is becoming an urgent task
  - Large amount of jobs are waiting for resources
    - >100k jobs sitting idle in the local job queue
  - The htc pool is quite busy
    - >90% resource utilization rate (>35,000 CPU cores)



Basic Idea

- The extended resources are likely added as direct worker nodes into the local htcondor
- The users still keep the previous usage mode and interact with the computing system in a unified entrance
- The solution is still based on HTCondor

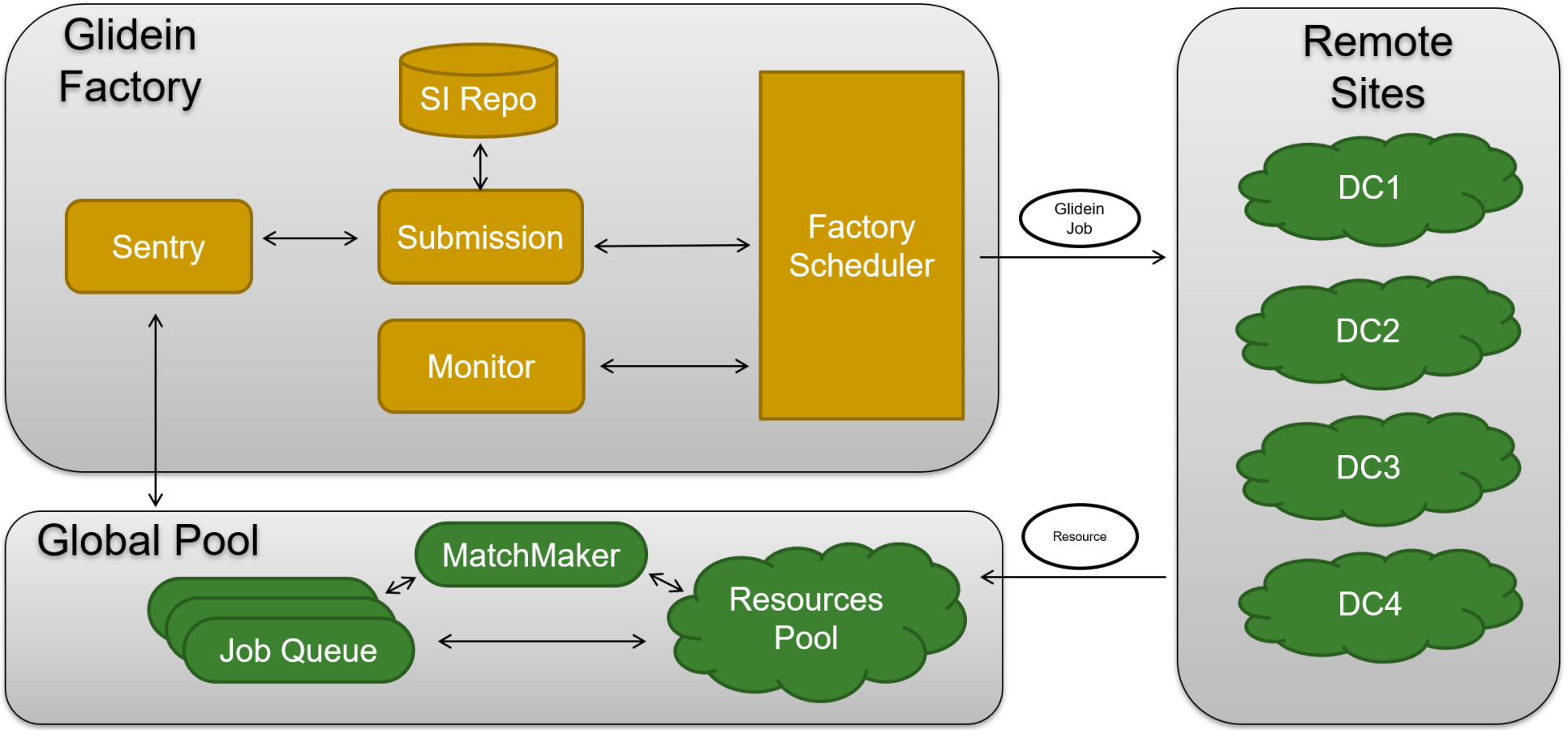


Computing Structure

- A typical dHTC computing structure but with the customized design
- The design and implement is based on htcondor glidein
  - Glidein Factory
    - Produce the glidein jobs according to the defined policies
  - Glidein Job
    - Start the htcondor startd and report the worker node to the local htc pool
  - Certification and Authentication
    - IDTokens and Kerberos (Kerberos has served for many years in the local cluster)
  - User Interface（Workflow & HepJob）
    - Job and data workflow (some experiments have no standard software for data processing)
    - Unified job interface (by extending the job submission tool - HepJob)

Factory Workflow

- A central glidein factory can cover most sites or clusters
- Glidein job will be submitted when the sentry finds new user jobs in the global job queue



Factory - Schedule Policies

- A heterogeneous policy of resource share
- Static-resource policy (for self-owned and the pledged sites)
  - Resource provider report the promised number of long-term resources to the pool
  - All the resources have been ready and glidein job can be submitted at anytime.
- Dynamic-resource policy (for cooperation sites and HPC clusters)
  - Resource provider report a specific number of short-term resources as the workload
  - Site decides when and what resources will be reported
    - A client on site side to take charge of glidein submission and removal
- Job policies depend on the job requirements
  - All the job can run in the local HPC cluster due to the same worker node environment
  - The job with small data transfer can run in the edge sites without pledged storage
  - The job with large data output can run in the big site with large storage and good network

Extending to Local HPC (1)

- Several solutions were investigated and tested to share resources between HPC and HTC
  - Overlap & Flocking & HTCondor-C

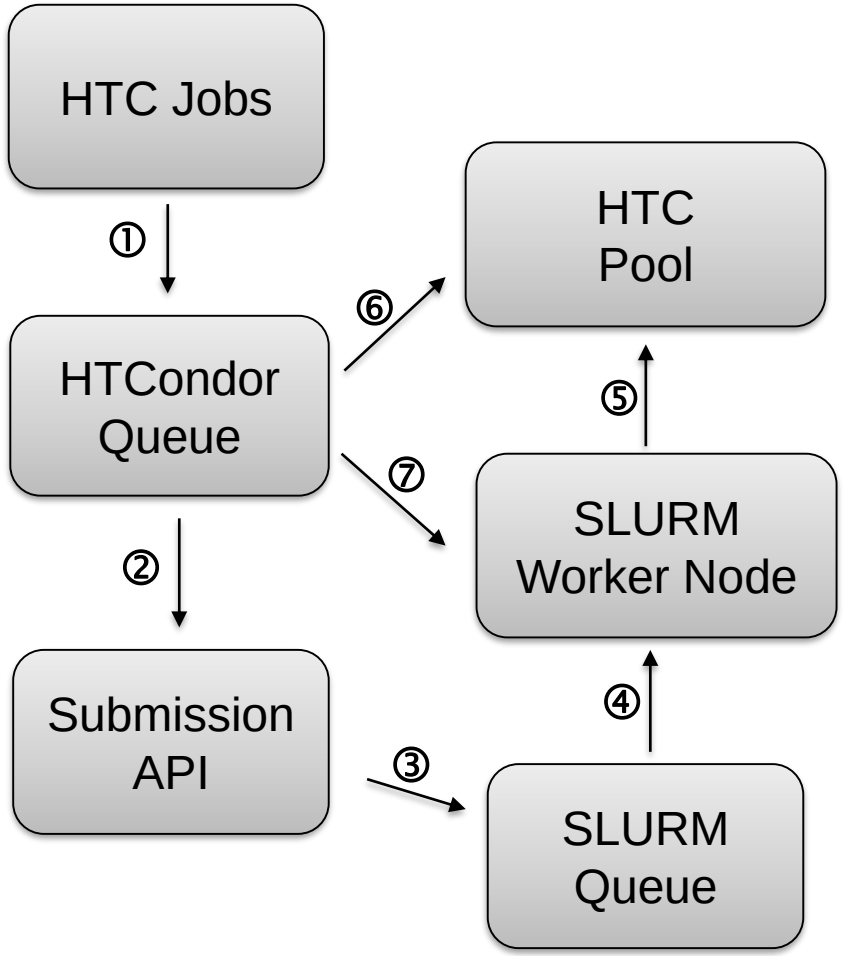
Metrics	Overlap	Flocking	HTCondor-C
Configuration	Scale up with worker nodes number.	Scale up with pools number.	Scale up with cluster number.
Customized job scheduling	**Almost none**, depends on HTCondor configuration.	**Almost none**, depends on the job queue status.	**Rich** : Job Router, Slurm spank plugins, blahpd

- The problem is to bring too large amount of jobs to SLURM
- Similarly as the overlap, glidein way can avoid submitting too many htc jobs to SLURM queue.

Reference: A Feasibility Study on workload integration between HTCondor and Slurm Clusters

Extending to local HPC (2)

- The system software of local HPC WN is same as that of HTC WN
  - With shared file system and same user namespace
- The workflow is:
  - ① submit htc jobs to htcondor queue
  - ② submission API receives the job requirements
  - ③ submission API submits glidein jobs to slurm queue
  - ④ slurm schedules glidein jobs to slurm worker nodes
  - ⑤ glidein startups startd and report to htc pool
  - ⑥ matchmaker
  - ⑦ htc jobs are sent to slurm worker node
- The policy in submission API is quite easy
  - Specific idle user jobs & empty worker nodes in slurm



Extending to Remote Sites

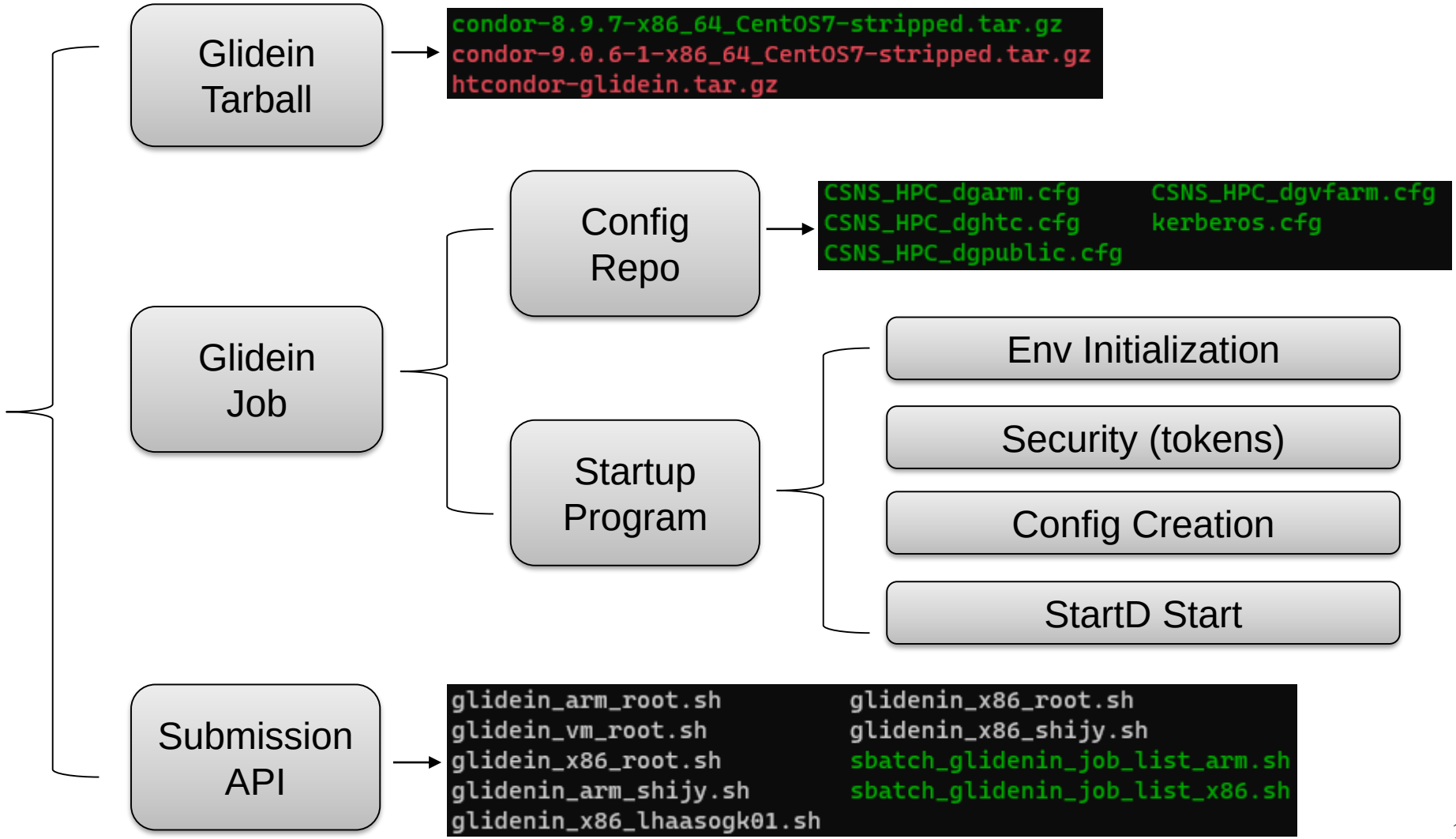
- Currently, we are focusing on one of the remote sites – Dongguan DC
- Dongguan Data Center is a typical remote site
  - A HPC pool built with Slurm
  - Without shared file system & different user namespace

Could CPU：9600 cores
X86 CPU：10000 cores
ARM CPU：9600 cores
GPU：80 Tesla V100
Storage：6PB

- The advantage of Dongguan site is that belong to the IHEP internal network
  - Network bandwidth is 10Gpbs (dedicated)
  - Data access and transfer has more possibilities
    - Access data by EOS commands
    - Read/write files in Lustre by XRootD gateway

Glidein Job

- Glidein Client
  - Glidein Job
  - Submission

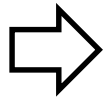


Cert&Auth with Tokens

- The aim is to migrate from CLAIMTOBE to Tokens
  - CLAIMTOBE is used in the local cluster for years (not safe)
- IDTokens for Daemon Certification
  - All works are following htcondor manual (configuration level)
- Kerberos tokens for jobs and other services
  - First try based on htcondor configuration was failed
    - It seems the same user namespace between submit side and execute side is necessary
  - Current solution is to transfer token credential as a normal input file
    - The token credential will be initialized and akloged in job wrapper

Submit Side

```
transfer_input_files = /tmp/krb5cc_10634
+HepJob_KRB5CCNAME = "krb5cc_10634"
```

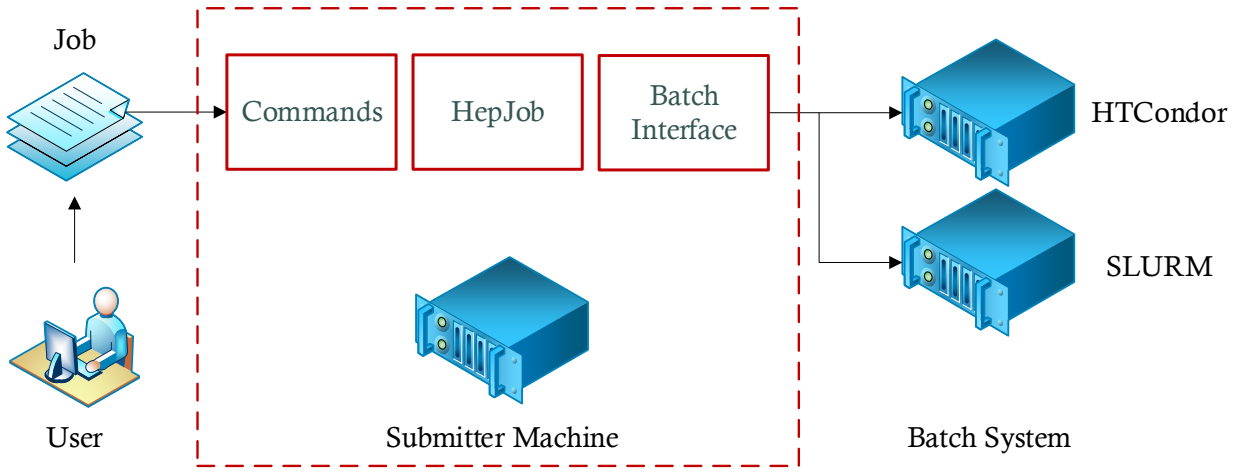


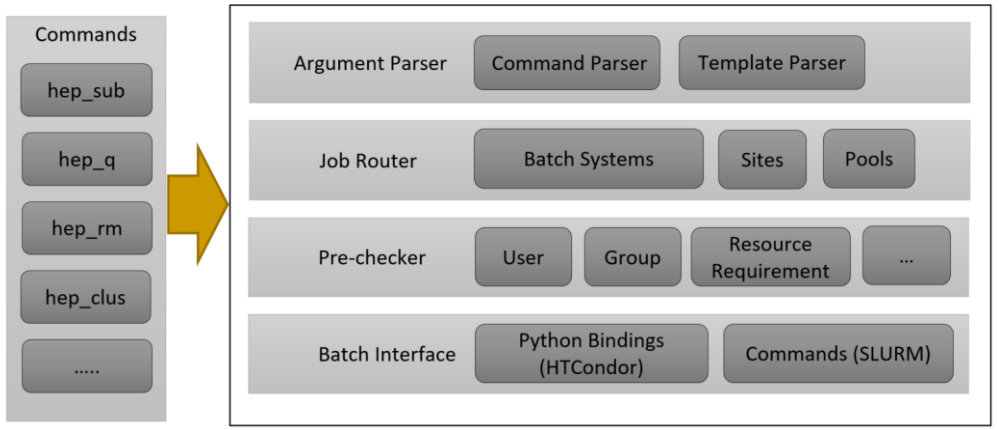
Execute Side

```
$ ls /var/lib/condor/execute/dir_6412/
condor_exec.exe  _condor_stderr  _condor_stdout  krb5cc_10634  tmp  var
```

User Interfaces

- User interface will fully respect the old usage mode
  - Users would like to run job remotely without much change of the previous usage mode (especially the commands)
  - All the new functions are wrapped and added by extending the job tool – HepJob

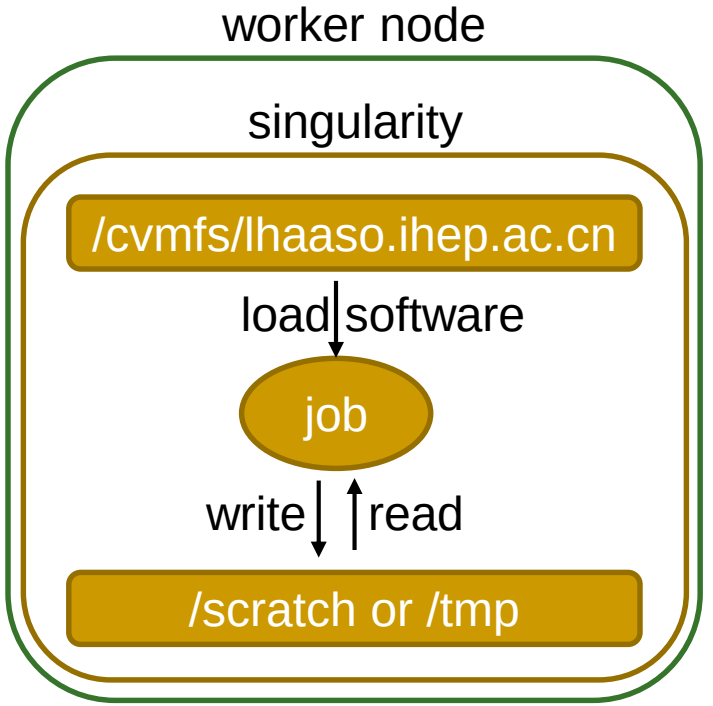




- Job workflow is helpful to introduce dHTC to experiments
  - Re-design and implement the process of job workflow and the structure of data storage

User Job Environment (Singularity)

- Job environment in all solutions are based on singularity
- Operating System
  - Singularity images are published into /cvmfs/container…
  - Glidein job starts up singularity as the given image
- Software
  - Managed and served by CVMFS (recommended)
  - Transferred with job, as part of job input
- Temporary storage
  - The local scratch on the worker node
  - The global storage shared in the whole distributed infrastructure



Application Cases

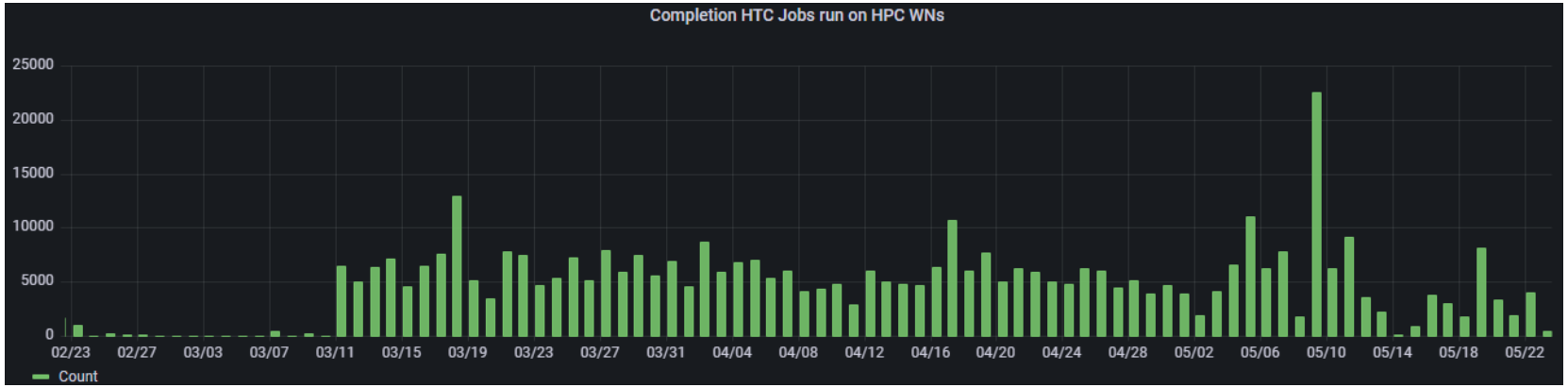
- LHAASO simulation
  - Characteristics：small input and not large output
  - Job submission policy is that user decides where the job will run
    - No mature common software and users have their own software in the personal directory
    - Still a simple command in HepJob: hep_sub job.sh -rmt

	Corsika	Geant4(step1)	Geant4(step2)	Reconstruction	Analysis
input	tiny	middle	large	small	various
output	middle	large	small	small	small
persistent store?	No	Yes	No	Yes	No
CPU hours	long	Long	Short	Short	Various
users	Dedicate user	Dedicate user	Dedicate user	Dedicate user	Various user

- BES simulation
  - Characteristics: mature software framework (BOSS) ; tiny input but big output
  - Job policy is that job will be transparently scheduled to anywhere by scheduler policy
    - Job type and data position can be easy to judge with BOSS job option file
    - The simulation job will be selected and the data path will be remapped to the new path in the remote sites

Current Status of Application

- HTC jobs have been dispatched in the extended HPC resources for several months



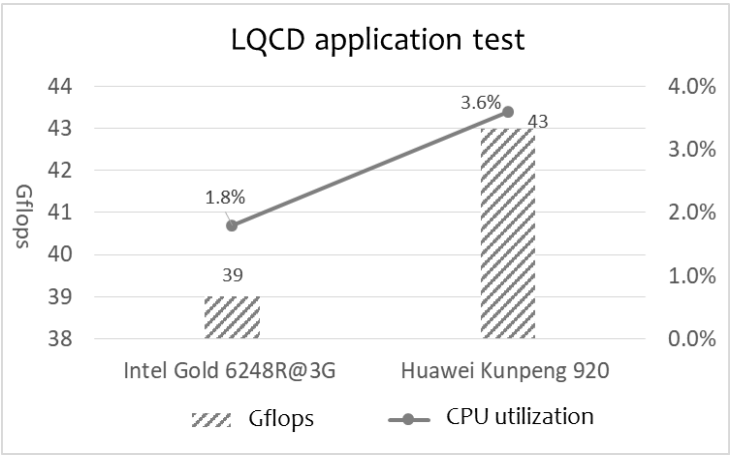
- Some specific jobs have been running at the Dongguan site
  - LHAASO experiment: WCDA simulation job and LHAASO WFCTA simulation job
  - BESIII experiment: official offline simulation job
    - Filter all the simulation jobs submitted by users according to the specific attributes (on going)
  - Other experiments: automatically schedule the jobs with small input/output data
    - Plan to finish the work by the end of this year

Trying to add ARM into dHTC

- ARM computing cluster at Dongguan Data Center
  - Huawei Taishan 200K server with Kunpeng 920 CPU
  - 100 worker nodes & ~10K CPU cores

Node Type	Num. of Jobs	Num. of servers	Job Run Time (s)
X86	5000	209 ↓ 199%	982.073 ↑ 169.2%
ARM	5000	105	1661.432

- Performance test (Corsika: Air Shower Simulation Program)
  - 5000 Corsika simulation jobs (htc jobs)
  - X86 takes less time to complete the same number of jobs, but requires more servers
- htcondor on arm machine
  - Compile the htcondor glidein tarball
  - Report arm resources to global pool via glidein
- Experiment job on arm machine
  - Help experiments to port and compile their software
  - Succeed to run large-scale corsika jobs in dHTC (the results are still under verification)



Summary and Plan

- Adding more resources by extending the local htc cluster is the better way currently
  - Experiments at IHEP have been used to processing data in the local cluster for a long time and the local htc cluster is getting quite busy
- With htcondor glidein, we have make a basic design of dHTC at IHEP
- Two parts of resources have been extended into the local htc cluster and served in production
  - The local HPC cluster and Dongguan site
- Future plan
  - Implement a complete glidein factory and make the whole process more automatic
  - Introduce the dHTC to more experiments and migrate their jobs to the dHTC resources
- Current Problems
  - Kerberos Tokens: renew and prolong
  - A better way to access and transfer data

Thanks
Q&A