# Table of Contents

# Dirac for Administrators

# Tagging and distributing new releases

NewDiracTagAndRelease

## Finding what Dirac version are available

Look at https://raw.github.com/DIRACGrid/DIRAC/integration/releases.cfg

# Updating or patching the server installations

- ILCDiracUpdateServers

# Administration Scripts

Scripts that do not really belong into the ilcdirac code repository can be found here ILCDiracOps. Accessible only for ilcdirac-admin e-group members at the moment.

# Accessing the machines

To access machine, simple ssh from inside cern to any of the voilcdirac* machines. It might be that this access will be restricted to aiadm machines in the future. Then one has to join ai-admin e-group and log in to aiadm first.

## Granting access to someone

The list of users allowed to log on to these machines is specified in the puppet manifests for voilcdirac. If you can edit these manifests you can add users to log on to these machines.

# Migrating services to new machines

For most services, it's very simple: simply install an instance of the service on the new machine, and you're done. For the services below, there are a few precautions.

## Proxy Manager

Nothing particular but the fact that the DB must be installed on the localhost, and not on the DBOD service. This is for security as the proxies are stored in there.

## Configuration Service

- Copy the ILC-Prod.cfg that's in `/opt/dirac/etc/` to the new host. Also copy the `/opt/dirac/etc/csbackup` directory.
- Edit the new host's `/opt/dirac/etc/dirac.cfg` and change the following sections:

```
DIRAC
{
  Configuration
  {
    Servers = dips://volcd05.cern.ch:9135/Configuration/Server, dips://volcd01.cern.ch:9135/Confi
    MasterServer = dips://volcd05.cern.ch:9135/Configuration/Server
    Master = yes
    Name = ILC-Prod
  }
  Setups
  {
    ILC-Production
    {
      Configuration = Production
    }
  }
}
```

- Make sure the ILC-Prod.cfg file also contains the same info for the MasterServer.
- Restart the Configuration_Server and check that in its log it starts with `Starting configuration service as master`
- Update all the other machines.

Make sure the `/afs/cern.ch/eng/clic/data/ILCDIRACTars/defaults/ilc.cfg` contains the link to the new CS instance too. Potentially update the client's dirac.cfg (if the former host(s) is(are) killed)

## Request Management

This cannot be moved easily at all: jobs need the service to be available (for failover requests), and it's not possible to move to a new system from scratch: jobs would never be cleaned as their requests would never be marked as Done. The new version of the system being completely different, it will be possible to have both running in parallel.

## Sandbox Store and associated DB

This service is a bit tricky. It defines not only a service for the Job WMS, but also a storage element. To migrate, the recommended recipe is the following (A. Casajus):

1. Install a new sandbox store in your new host

2. Define a new SE for that sandbox (I'll call it NewSBSE)
    - Here you have two SB services running
3. Define the SB SE to be your new one (NewSBSE)
    - Here both of your SB services should have configured the new SE
    - Old jobs will still retrieve their SB from the old one since the SE is embedded in the SB URL
    - New jobs will use the New SB for their sandboxes
4. As soon as your old SB gracefully stops being used, remove it

The jobs know about the SE to use, as in their JDL, there is something like `"SB:ProductionSandboxSE|/SandBox/i/ilc_prod/03f/e5e/03fe5e4cb9889b87bb437adcc310337f.tar.bz2"` where the sandbox storage element is defined as `ProductionSandboxSE`. So when the job gets its input sandbox, it will connect to the `ProductionSandboxSE` which has as en end point a specific SandboxStore.

# gLite job matching

Understanding the way the job matching is done is necessary to understand why jobs get killed sometimes.

When a job is submitted to DIRAC, it is inserted in a certain TaskQueue that has a set of properties among which the CPU time needed in seconds. That's normally a "Wall clock" time. This CPUtime is not the one specified (as then one would have thousands of TaskQueues), but they are grouped per "segments". The different segments are in

```
DIRAC/WorkloadManagementSystem/Private/Queues.py
```

For example, if the CPU time is set to 300000 (case in the production), the closest segment is 4*86400 = 345600. So all the jobs that require 300000 CPU seconds will in fact require 345600 seconds.

When the TaskQueues are examined to create a grid job, the requirements are built using the following complicated procedure:

```
Rank = ( other.GlueCEStateWaitingJobs == 0 ? ( other.GlueCEStateFreeCPUs * 10 / other.GlueCEInfoT

Lookup = "CPUScalingReferenceSI00=*";
cap = isList(other.GlueCECapability) ? other.GlueCECapability : { "dummy" };
i0 = regexp(Lookup,cap[0]) ? 0 : undefined;
i1 = isString(cap[1]) && regexp(Lookup,cap[1]) ? 1 : i0;
i2 = isString(cap[2]) && regexp(Lookup,cap[2]) ? 2 : i1;
i3 = isString(cap[3]) && regexp(Lookup,cap[3]) ? 3 : i2;
i4 = isString(cap[4]) && regexp(Lookup,cap[4]) ? 4 : i3;
i5 = isString(cap[5]) && regexp(Lookup,cap[5]) ? 5 : i4;
index = isString(cap[6]) && regexp(Lookup,cap[6]) ? 6 : i5;
i = isUndefined(index) ? 0 : index;

QueuePowerRef = real( !isUndefined(index) ? int(substr(cap[i],size(Lookup) - 1)) : other.GlueHost
#This is the content of CPUScalingReferenceSI00 (ex. 44162 for kek)

QueueTimeRef = real(other.GlueCEPolicyMaxCPUTime * 60);
QueueWorkRef = QueuePowerRef * QueueTimeRef;

CPUWorkRef = real(345600 * 250);# 250 SpecInt 2000 or 1 HepSpec 2006

requirements = Rank >  -2 && QueueWorkRef > CPUWorkRef ;
```

The `Rank` is used to sort the sites according to the request.

The next block is used to obtain from the BDII the `CPUScalingReferenceSI00` parameter of the CE, or, if not defined, the `GlueHostBenchmarkSI00` parameter is used. This is the normalization factor of the CPU to HepSpec (jn principle). The `GlueCEPolicyMaxCPUTime` (minutes) is also obtained and converted to seconds, then multiplied by the scaling to have the maxCPUTime in HepSpec seconds units. The job's CPUtime (already in seconds) is converted also to HepSpec seconds units (factor 250), and put in the requirement which are used by the resource brokers.

It should be made clear that the `GlueCEPolicyMaxCPUTime` is not the `GlueCEPolicyMaxWallTime` parameter.

The relation between HEP-spec and kSI2K is value_kSI2K = value_HEP-SPEC / 4. This second relation is needed to understand the factor 250 above: value_SI00 = value_HEP-SPEC * 250

## How to get `GlueCEPolicyMaxCPUTime` and `CPUScalingReferenceSI00` from the BDII

You need to run

```
ldapsearch -x -LLL -h lcg-bdii.cern.ch -p 2170 -b o=grid '(&(objectclass=GlueCE)(GlueCEUniqueID=*
```

where `*kek*.jp*` should be replaced by a proper CE.

See http://glueschema.forge.cnaf.infn.it/Spec/V13 for the GLUE specification document. There is also a v2.0 that was designed by OGF. I don't know when it will be used nor how. It will certainly be a big mess.

# How to get the available CEs?

This should list the available sites for the ILC VO. The problem is that this depends on the good will of the sites to publish their info in the BDII, so ti's not necessarily correct...

```
ldapsearch -x -LLL -h lcg-bdii.cern.ch -p 2170 -b o=grid '(&(objectclass=GlueCE)(GlueCEAccessCont
```

# CE Maintenance

- ILCDiracCEMaintenance
- Debugging CEs directly

# Registering New Users

RegisteringNewUsersToDirac

# FAQ

## What if the Configuration service starts to be very slow?

That can be due to many things. The first check is to look at the Monitoring plots of the Configuration service. This will tell you the load of the system. If you see sudden rise in the number of active queries and/or Max file descriptors, this indicates something like a DOS attack. It's most likely due to a site that has a router issue: when packets are transmitted, something is lost and DIRAC tries again and again. To identify if this is a real issue, contact the LHCb people (Joel and/or Philippe Charpentier) and ask them if they see something similar. As we share a few of our sites, but not all, that's not necessarily a good indication. To see a bit further, you need to log onto the vobox hosting the configuration services and check the `netstat` output, grepping for 9135 as that's the CS port number. If you see many hosts of the same site, then the hypothesis is validated, and the site can be treated the following way:

1. ) Ban the site with `dirac-admin-ban-site` so that no new jobs are attempted there
2. ) Add the hosts to the iptables. You'll need to log as root and run `/sbin/iptables -I INPUT -s 193.62.143.66  -j DROP` where the IP can be the host name. Given the host, finding the IP is possible on many sites (GIYF).
3. ) Add a JIRA issue mentioning the problem.
4. ) If LHCb does not see the problem, put a ggus ticket against the culprit.

When the problem is resolved (either when the site replies or when LHCb says it's fixed), you can close the JIRA issue, unban the site (`dirac-admin-allow-site`), and reallow the IP (`/sbin/iptables -D INPUT -s 193.62.143.66  -j DROP`). Hopefully you should start to see things running smoothly again.

# Setting up development installation

- ILCDiracDevInstallation

# Sites

Comments and notes

This topic: CLIC > DiracForAdmins
Topic revision: r43 - 2015-08-21 - AndreSailer