The CE Collector: Information Services for the HTCondor CE

Mátyás Selmeci
OSG Software Team

OSG All Hands Meeting • Northwestern University • 24 March 2015

Background

Purpose of Information Services

- Clusters have machines that vary in power, policy, etc.

- Need to know about these differences to send jobs

- Send Glideins from a central Factory to shield users from this complexity

- Collect machine information in a central location; Factory can query it to determine where to send Glideins

What Kind of Information?

- Information Services consists of the facts needed to provision machines by sending Glideins
  - Machine capabilities (RAM, CPU)
  - Machine policies (max run time, VO permissions)
  - Machine access methods (queues, required attributes)

Why Something New?

- Currently use GIP/BDII for info services, but:
- System too heavyweight; conflates site reporting with machine provisioning
- Fixed schema: not extensible with new attributes

CE Collector

- New mechanism to replace GIP for HTCondor CEs
    - Special configuration in HTCondor CE, not new software
- Work in progress
- Smaller and simpler: provisioning only
- No fixed schema

No Fixed Schema

- Attributes can be added without compatibility issues
- Easy to react to future capabilities
- Downside: effort needed to avoid proliferation of similar attributes
- For now, OSG Software Team will act as gatekeeper for new attributes

For Site Admins

Nothing New to Install

- Using existing software
  - OSG Configure to generate the information
  - HTCondor CE to send the information
- Few manual steps for existing sites

Resources
(aka Subclusters)

Uniform set of machines at a site

32GB, 12 core
Resource "gray"

32GB, 12 core
needs "WantRHEL7" set
Resource "blue"

128GB, 24 core
12 hour jobs
reserved for the glow VO
Resource "green"

Number of machines in each resource not important

Configuring a Resource 1

- Reuse existing GIP configuration as-is

```
/etc/osg/config.d/30-gip.ini:

[Subcluster Example_Gray]
name = Example_Gray
cores_per_node = 12
ram_mb = 32768
```

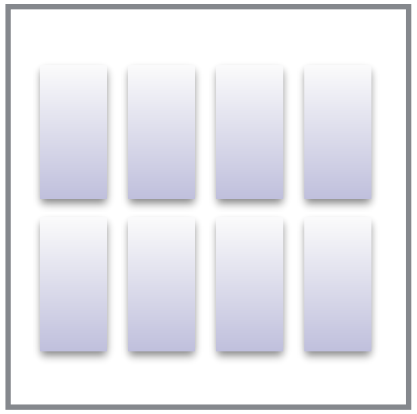
32GB, 12 core
Resource "gray"

Configuring a Resource 2

- Extend with access method if needed

```
/etc/osg/config.d/30-gip.ini:

[Subcluster Example_Blue]
name = Example_Blue
cores_per_node = 12
ram_mb = 32768
extra_transforms = set_WantRHEL7=1
```



32GB, 12 core
needs "WantRHEL7" set
Resource "blue"

Configuring a Resource 3

- Extend with policy if nonstandard

```
/etc/osg/config.d/30-gip.ini:

[Subcluster Example_Green]
name = Example_Green
cores_per_node = 24
ram_mb = 131072
max_wall_time = 720
allowed_vos = glow
```

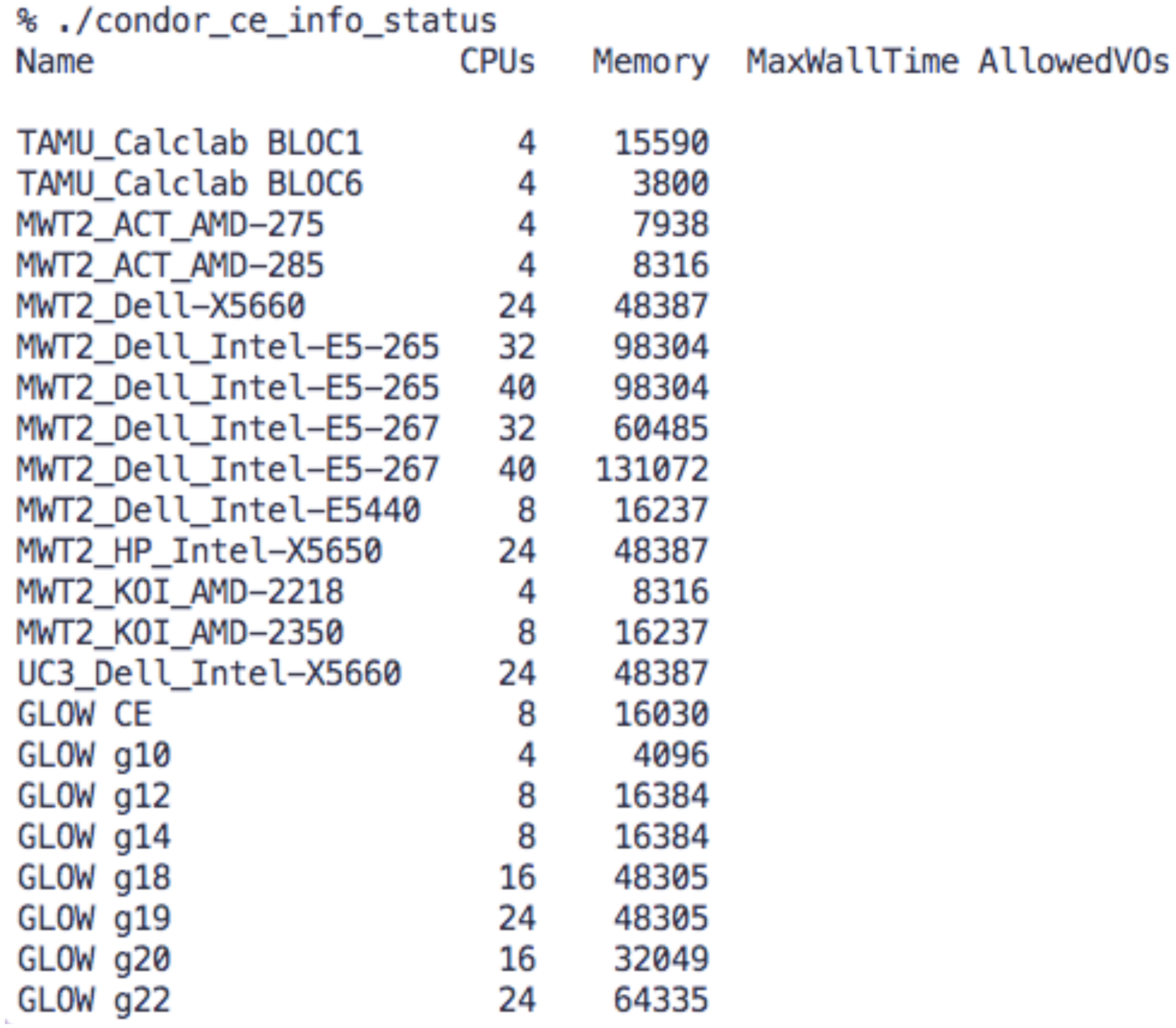
128GB, 24 core
12 hour jobs
reserved for the glow VO
Resource "green"

Applying Configuration

- Re-run osg-configure
    - Adds resource information to HTCondor CE config
    - Tells HTCondor CE daemons to reload config and start sending data to GOC

Querying 1

- Run condor_ce_info_status (found in the htcondor-ce-client package)

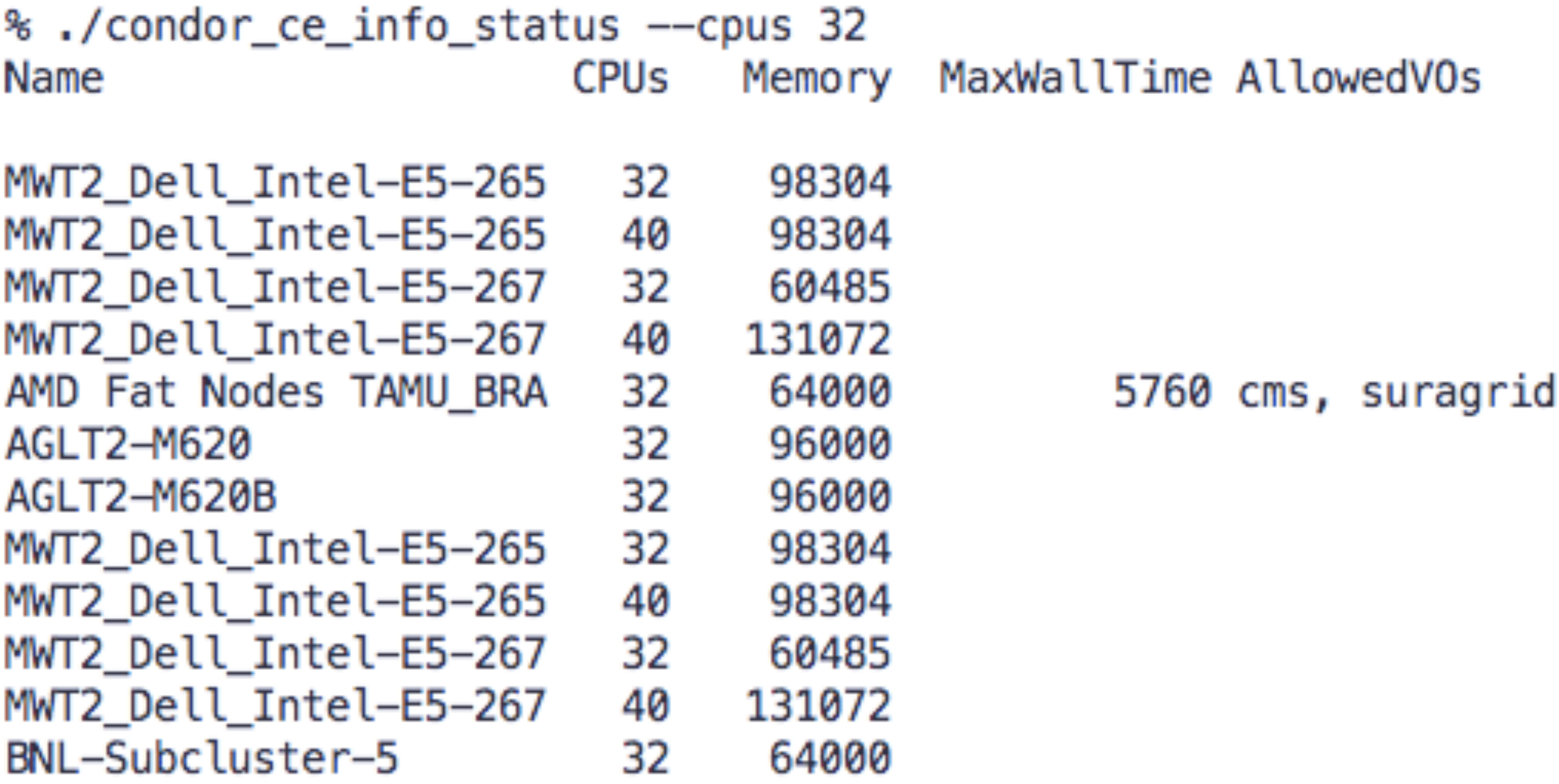


```
% ./condor_ce_info_status
Name                     CPUs   Memory  MaxWallTime AllowedVOs

TAMU_Calclab BLOC1          4    15590
TAMU_Calclab BLOC6          4     3800
MWT2_ACT_AMD-275            4     7938
MWT2_ACT_AMD-285            4     8316
MWT2_Dell-X5660            24    48387
MWT2_Dell_Intel-E5-265     32    98304
MWT2_Dell_Intel-E5-265     40    98304
MWT2_Dell_Intel-E5-267     32    60485
MWT2_Dell_Intel-E5-267     40   131072
MWT2_Dell_Intel-E5440       8    16237
MWT2_HP_Intel-X5650        24    48387
MWT2_KOI_AMD-2218           4     8316
MWT2_KOI_AMD-2350           8    16237
UC3_Dell_Intel-X5660       24    48387
GLOW CE                     8    16030
GLOW g10                    4     4096
GLOW g12                    8    16384
GLOW g14                    8    16384
GLOW g18                   16    48305
GLOW g19                   24    48305
GLOW g20                   16    32049
GLOW g22                   24    64335
```

Querying 2

- Constraining results from condor_ce_info_status



```
% ./condor_ce_info_status --cpus 32
Name                   CPUs   Memory  MaxWallTime AllowedVOs

MWT2_Dell_Intel-E5-265   32    98304
MWT2_Dell_Intel-E5-265   40    98304
MWT2_Dell_Intel-E5-267   32    60485
MWT2_Dell_Intel-E5-267   40   131072
AMD Fat Nodes TAMU_BRA   32    64000         5760 cms, suragrid
AGLT2-M620               32    96000
AGLT2-M620B              32    96000
MWT2_Dell_Intel-E5-265   32    98304
MWT2_Dell_Intel-E5-265   40    98304
MWT2_Dell_Intel-E5-267   32    60485
MWT2_Dell_Intel-E5-267   40   131072
BNL-Subcluster-5         32    64000
```

The Internals

Resource Catalog 1

- An HTCondor CE classad attribute
- Generated by OSG Configure
- One entry per subcluster

```
OSG_ResourceCatalog = { \
   [ (Entry for blue) ]; \
   [ (Entry for green) ]; \
   [ (Entry for gray) ]; \
}
```

Resource Catalog 2

```
[Subcluster Example_Gray]
name = Example_Gray
cores_per_node = 12
ram_mb = 32768
```

```
[ \
 Name          = "Example_Gray"; \
 CPUs          = 12; \
 Memory        = 32768; \
 Requirements = \
  TARGET.RequestCPUs   <= CPUs && \
  TARGET.RequestMemory <= Memory && \
 Transform = [ \
  set_MaxMemory    = RequestMemory; \
  set_xcount       = RequestCPUs; \
 ]; \
] \
```

32GB, 12 core
Resource "gray"

Resource Catalog 3

```
[Subcluster Example_Blue]
name = Example_Blue
cores_per_node = 12
ram_mb = 32768
extra_transforms =
  set_WantRHEL7=1
```

```
[ \
  Name            = "Example_Blue"; \
  CPUs            = 12; \
  Memory          = 32768; \
  Requirements = \
   TARGET.RequestCPUs    <= CPUs && \
   TARGET.RequestMemory <= Memory && \
  Transform = [ \
   set_MaxMemory     = RequestMemory; \
   set_xcount        = RequestCPUs; \
   set_WantRHEL7     = 1; \
  ]; \
] \
```

32GB, 12 core
needs "WantRHEL7" set
Resource "blue"

Resource Catalog 4

```
[Subcluster Example_Green]
name = Example_Green
cores_per_node = 24
ram_mb = 131072
max_wall_time = 720
allowed_vos = glow
```

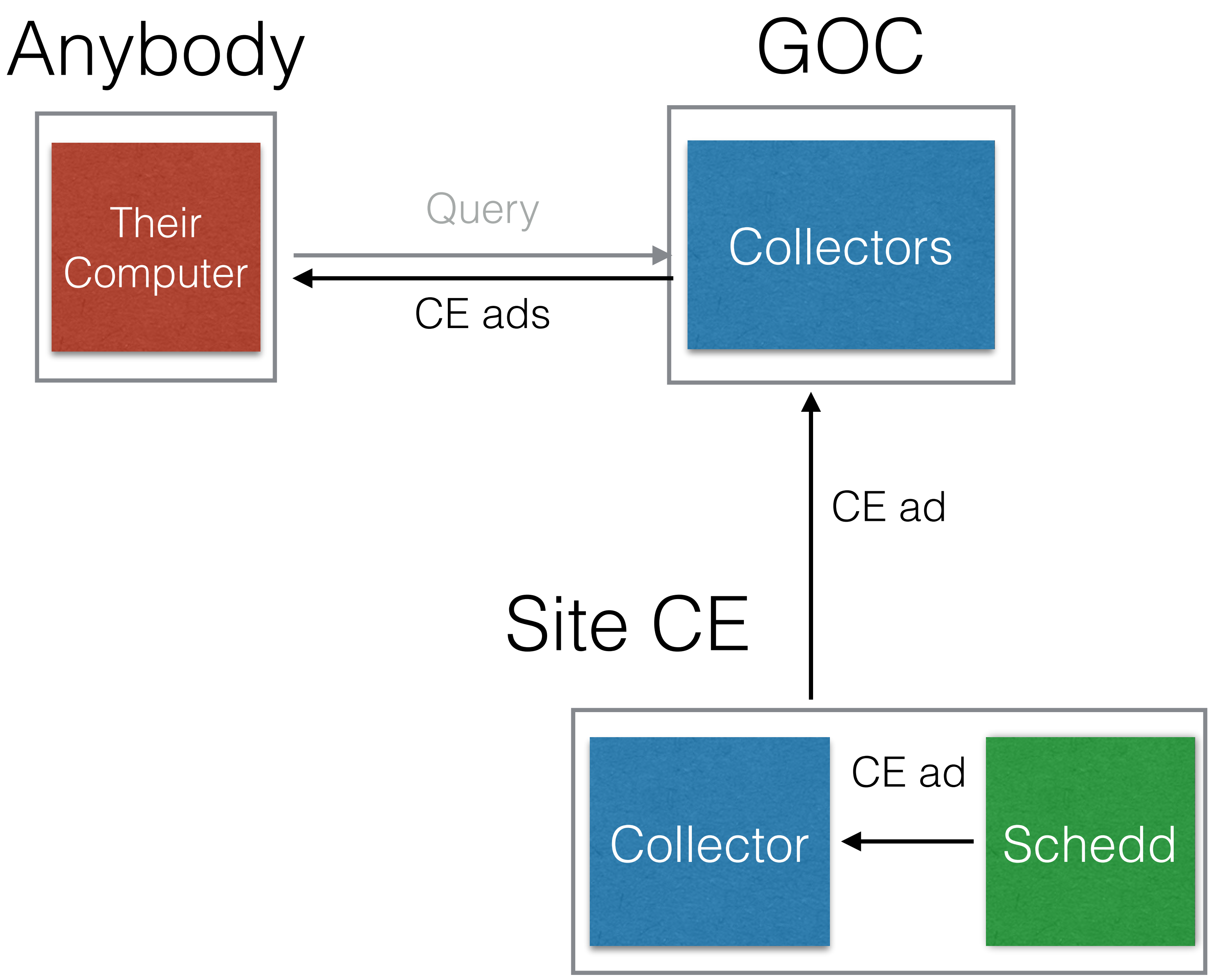
```
[ \
 Name          = "Example_Green"; \
 CPUs          = 24; \
 Memory        = 131072; \
 MaxWallTime   = 720; \
 AllowedVOs    = { "glow" }; \
 Requirements = \
  TARGET.RequestCPUs    <= CPUs && \
  TARGET.RequestMemory <= Memory && \
  member(TARGET.VO, AllowedVOs); \
 Transform = [ \
  set_MaxMemory     = RequestMemory; \
  set_xcount        = RequestCPUs; \
 ]; \
] \
```

128GB, 24 core
12 hour jobs
reserved for the glow VO
Resource "green"

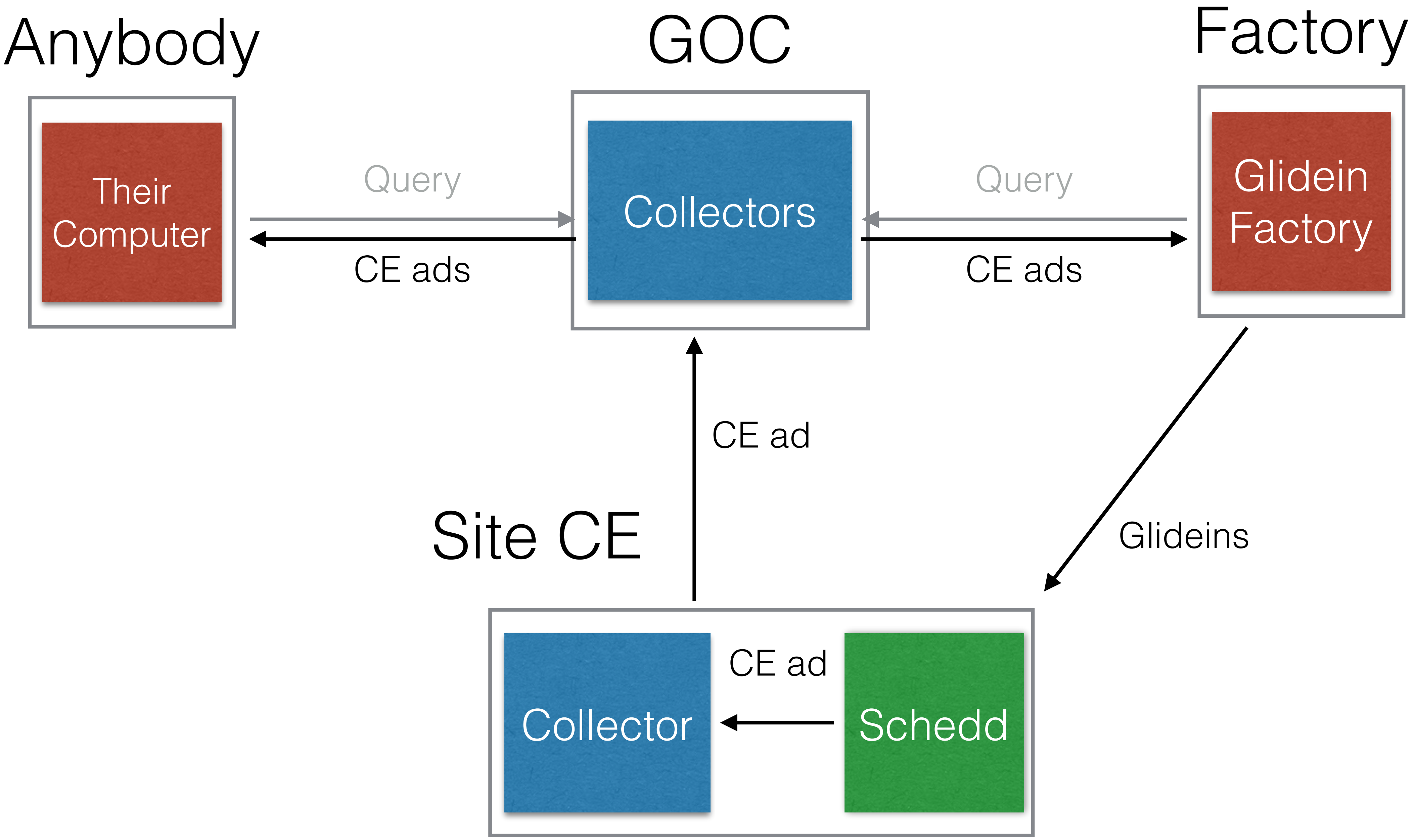
The CE Ad

- Created by an HTCondor CE daemon (condor_schedd)
- Resource Catalog packaged up with other necessary information:
  - Address needed to submit jobs
  - Batch system in use on the CE

Architecture (current)



Architecture (planned)



For Developers

Querying in Python 1

- condor_ce_info_query library found in htcondor-ce-client

```
import condor_ce_info_query as info
```

- Download all CE ads from the collector

```
ce_ads = info.fetchCEAds(
  'collector.opensciencegrid.org:9619')
```

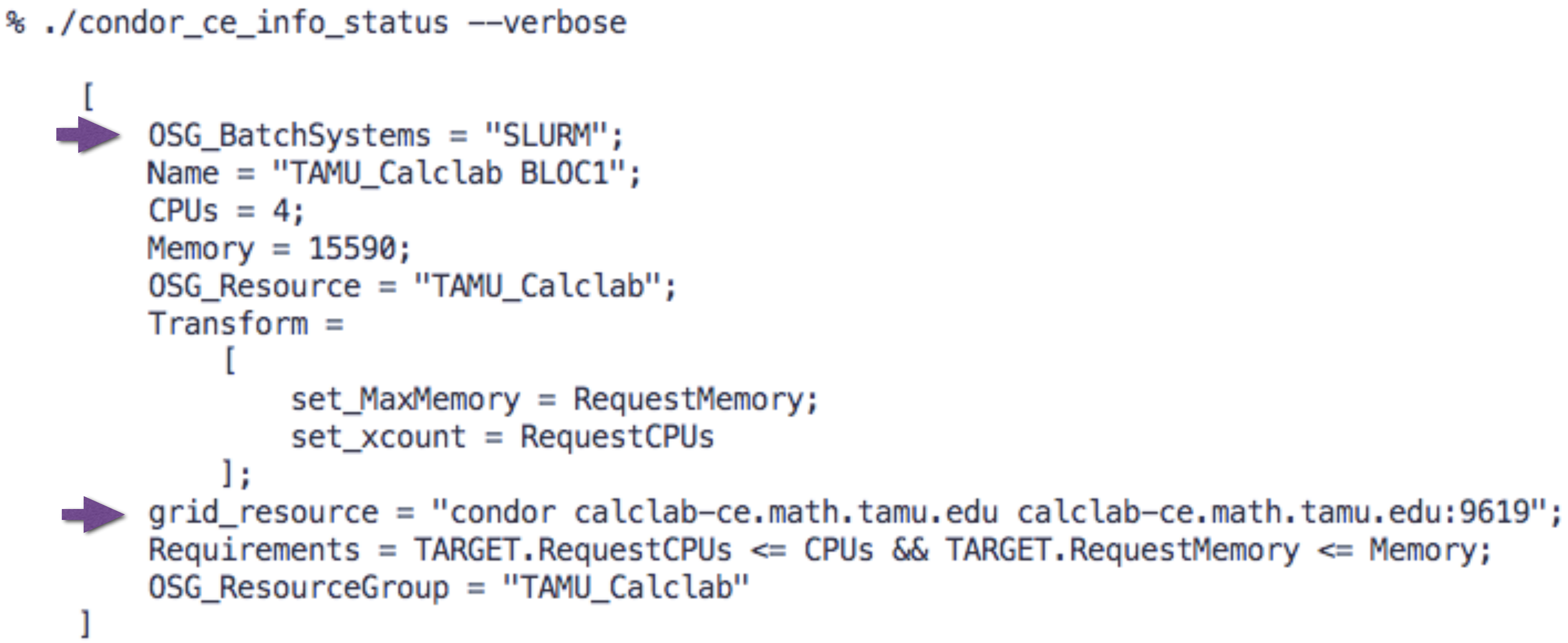
- Can work with CE ads directly, but easier to work with Resource Ads

Resource Ads 1

- Resource Catalog Entry plus attributes copied from the CE Ad
- All the necessary attributes to match against a resource and send jobs to it
- Created by query tool

Resource Ads 2

- Displaying Resource Ads with condor_ce_info_status



```
% ./condor_ce_info_status --verbose

    [
        OSG_BatchSystems = "SLURM";
        Name = "TAMU_Calclab BLOC1";
        CPUs = 4;
        Memory = 15590;
        OSG_Resource = "TAMU_Calclab";
        Transform =
            [
                set_MaxMemory = RequestMemory;
                set_xcount = RequestCPUs
            ];
        grid_resource = "condor calclab-ce.math.tamu.edu calclab-ce.math.tamu.edu:9619";
        Requirements = TARGET.RequestCPUs <= CPUs && TARGET.RequestMemory <= Memory;
        OSG_ResourceGroup = "TAMU_Calclab"
    ]
```

Querying in Python 2

- Split up list of CE ads into resource ads

```
resource_iter = info.getResourceAdsIter(ce_ads)
```

- Can iterate through ads and access them like dictionaries

```
for res_ad in resource_iter:
  print res_ad['Name'], res_ad['CPUs'], res_ad['grid_resource']
```

Submit File Generation (WIP)

```
grid_resource = "condor ce.example.net ce.example.net:9619"
Transform = [
 set_MaxMemory    = RequestMemory;
 set_xcount       = RequestCPUs;
 set_WantRHEL7    = 1;
];
```

```
info.getSubmitFileAdditions(resource_ad)
```

```
+GridResource = "condor ce.example.net ce.example.net:9619"
+MaxMemory    = RequestMemory
+xcount       = RequestCPUs
+WantRHEL7    = 1
```

Future Work

Future Work

- Integration with Glidein Factory
  - Submit file generation
- Better filtering in query tools
- More attributes
  - GPUs, etc.
  - Follow the needs of the community

Acknowledgements and Contact

- Thank you to the following people for their help with this presentation:
  - Brian Bockelman
  - Brian Lin
  - Jeff Dost
  - Tim Cartwright
- Questions, comments and feature requests should go to osg-software@opensciencegrid.org