Bearer Token Usage in the OSPool

MORGRIDGE
INSTITUTE FOR RESEARCH
RESEARCH COMPUTING

FEARLESS SCIENCE

Bearer Tokens in the OSPool

The Open Science Pool (OSPool) is a HTCondor-based resource pool.

- Historically, resources were added to the pool exclusively using GlideinWMS.
- Every component authenticated with others using GSI.
- Quite coarse-grained permissions; authorization was based on identity mapping.
  - With tokens, **we are moving to capabilities**. What you can do, not who you are!

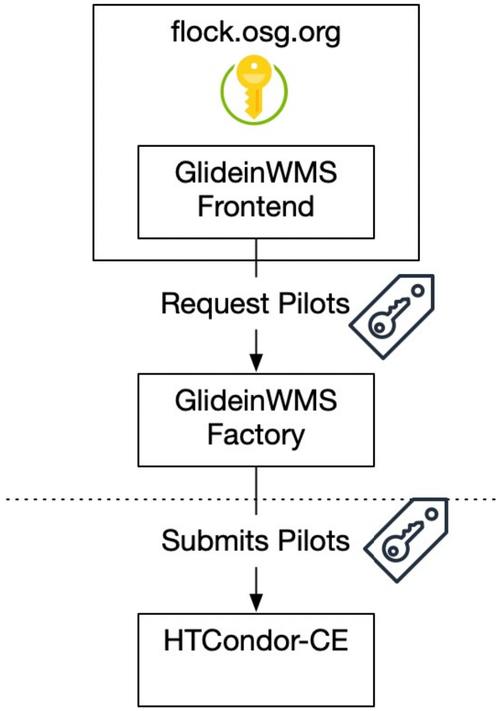
There are three use cases I want to highlight today:

- How tokens are used to submit pilot jobs to HTCondor-CE.
- How tokens are used to add resources to the pool.
- How tokens are used to provide storage access.

Use Case 1: Submitting pilots to HTCondor-CE

On our GlideinWMS frontend, we use the OSG signing key to generate a **SciToken per CE**.

- The frontend measures the job pressure in the OSPool and requests resources from clusters across the OSG.
- If pilots are needed, then the frontend sends a pilot request and encrypted token to the factory.
- The factory submits individual pilots to the CEs using the SciToken.
  - The factory->HTCondor-CE connection uses the CEDAR protocol with the SCITOKENS authentication method.
  - The SciToken identifies the client; a traditional X.509 host certificate identifies the server.



Use Case 1: Submitting pilots to HTCondor-CE

We purposely generate a **SciToken per CE** instead of one per VO:

- The CE SciToken is only accepted at a single host. All others reject it. Minimizes the "blast radius" of a stolen CE token.
- Each CE token also receives a unique subject and a unique token identifier.
- Scopes limit the token to being used for job submission.
- 1 hour expiration: token only needs to create the session with the CE. Continuously renewed; does not need to travel with the pilot.


```
{
  "sub": "vofrontend-SLATE_US_NMSU_DISCOVERY",
  "scope": "compute.read compute.modify compute.create
compute.cancel",
  "wlcg.ver": "1.0",
  "aud": "osg-ce.nmsu.edu:9619",
  "jti": "3f776538-e4c6-4781-86f2-b7c734604ae6",
  "iss": "https://scitokens.org/osg-connect",
  "exp": 1634180606,
  "iat": 1634177006,
  "nbf": 1634177006
}
```

Status: 52 Yes, 85 No

Every hour we poll all 137 HTCondor-CE's that report to the OSG Collector. **52 currently accept SciTokens from OSG**.

- Many of the missing CEs don't support OSG at all (e.g., purely ATLAS).
- 97% of the 38 hosted CEs run by OSG accept SciTokens.
  - Don't yet have the ability to track percent of pilots submitted with tokens.
- Would love to work with sites to close the gap tomorrow.

Pro-tip: you can remotely query the CE to see info about the token used!

```
[root@flock ~]# condor_q -pool collector.opensciencegrid.org:9619 -name osg-gk.mwt2.org -all osg -l | grep Auth
AuthTokenId = "570f7b0b-00af-4442-82eb-bb83b611037f"
AuthTokenIssuer = "https://scitokens.org/osg-connect"
AuthTokenScopes = "compute.read,compute.modify,compute.create,compute.cancel"
AuthTokenSubject = "vofrontend-OSG_US_MWT2_gk"
```

Use Case 2: Connecting the “execution point” to the central manager.

HTCondor has three major components:

- The Access Point (AP): where workloads are placed; includes the condor_schedd.
- The Execution Point (EP): where jobs are executed; includes the condor_startd.
- The Central Manager (CM): where daemons register and matchmaking occurs; condor_collector and condor_negotiator.

At the minimum, the AP and EP must authenticate with the CM and can do so using tokens.

- We use HTCondor’s IDTOKENS format for this. This uses symmetric encryption; only the issuer key can validate the token.
- The IDTOKEN is useless for anything but registering an EP or an AP.

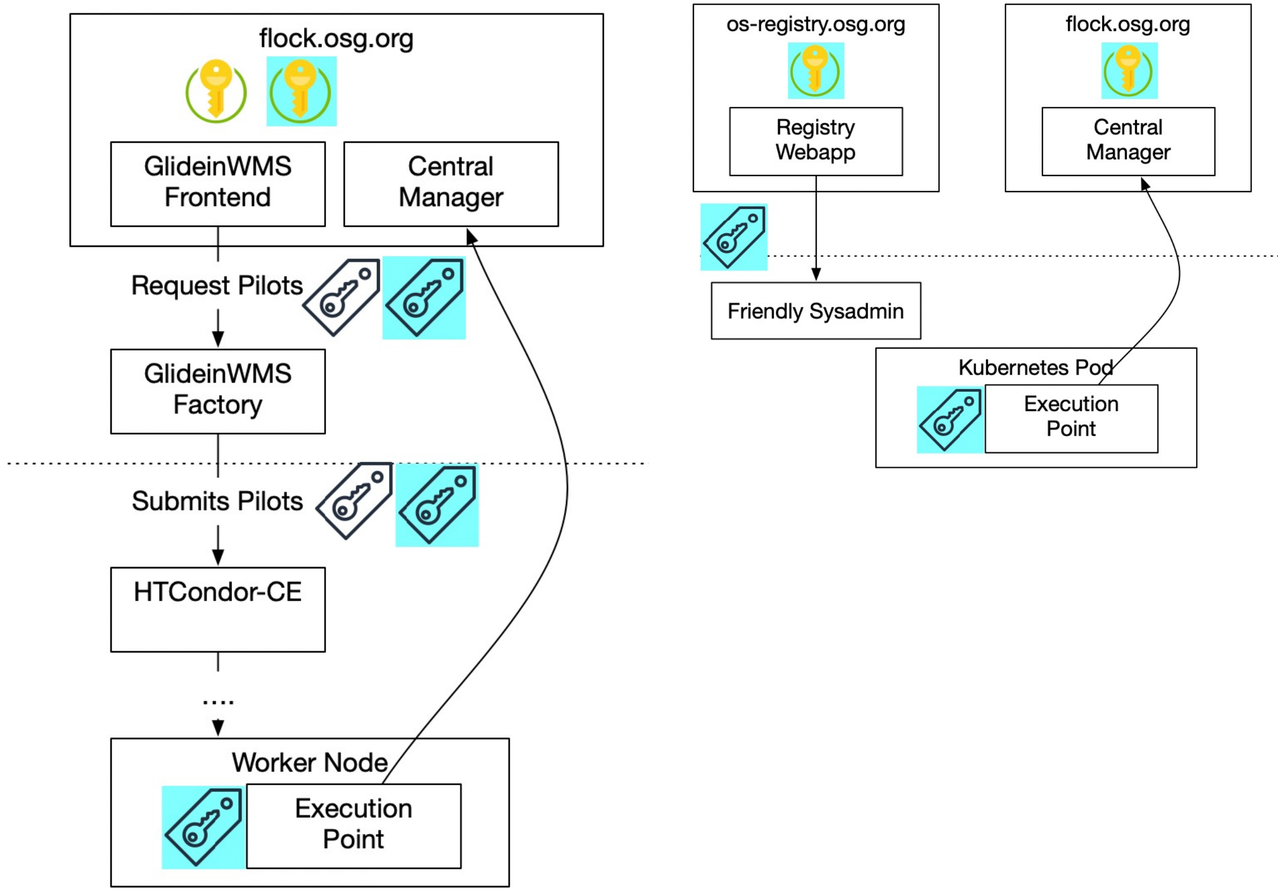
Tokens for EP's in two ways

GlideinWMS can generate a token and send it along with the pilot.

- The IDTOKEN, not the SciToken, travels all the way to the worker node.
- Note the IDTOKEN must remain valid for **however long the pilot is in queue**.

Alternately, trusted administrators can request a token for their site and approve it through the webapp using their CILogon identities.

- These tokens can be inserted as a secret into a Kubernetes pod.
- The site admin then makes the decision on how many "backfill" containers to launch.
- [See the OSPool Containers documentation](#).



Status: 95% converted

From a snapshot this week, 95% of ~30k slots in the OSPool were advertised via IDTOKENS.

- The remaining GSI usage appears to be misconfigurations on our side we need to track down. No site involvement needed.


```
{
  "sub":
"vofrontend_service@flock.opensciencegrid.org",
  "iat": 1634176259,
  "nbf": 1634176259,
  "jti": "d2abdcf1f3e6414988d82cac19b3e9b5",
  "iss": "flock.opensciencegrid.org",
  "exp": 1634262659,
  "scope": "condor:/READ condor:/ADVERTISE_STARTD
condor:/ADVERTISE_MASTER"
}
```

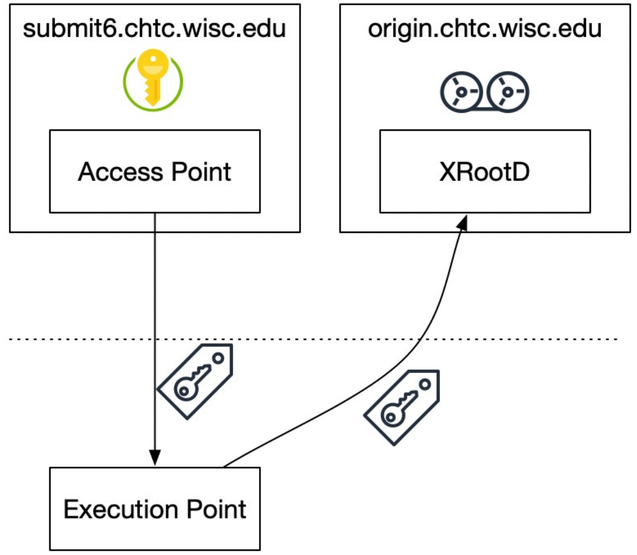

Pro-tip: you can remotely query the collector to see how a condor daemon authenticated!

```
[root@flock ~]# condor_status -pool flock.opensciencegrid.org:9800 -af AuthenticationMethod AuthenticatedIdentity | sort | uniq -c
      1 GSI pilot@flock.opensciencegrid.org
     67 IDTOKENS vofrontend_service@flock.opensciencegrid.org
```

Use Case 3: Job access to storage

The IDTOKEN only connects the EP to the pool and enables the EP to launch user jobs.

- By default, user jobs are marked as needing storage tokens.
- The AP runs the condor_credmon which automatically generates a token before a job is started.
  - A renewed token is periodically requested by the EP; ensures a valid token is always available.
- The token is usable at the (XRootD-based) origin server associated with the AP for writing…
  - Or any of the XRootD-based cache servers for reading.




```
{
  "ver": "scitokens:2.0",
  "sub": "brian.bockelman.1",
  "iss": "https://osg-htc.org/ospool",
  "jti": "9b57013d-9fc5-4fbe-a674-4844563177e3",
  "exp": 1634182518,
  "iat": 1634181318,
  "scope": "read:/brian.bockelman.1
write:/brian.bockelman.1",
  "nbf": 1634181318,
  "aud": [
    "ANY"
  ]
}
```

Status: Available everywhere!

The OSPool's use of tokens to authorize storage access has been in production for about 4 years; was one of the earliest users of the condor_credmon.

Recent activities:

- The command line tool (stashcp) is getting improved integration as a HTCondor file transfer plugin.
- Additional origin servers (at Wisconsin).
- Use of tokens to read through authenticated caches.

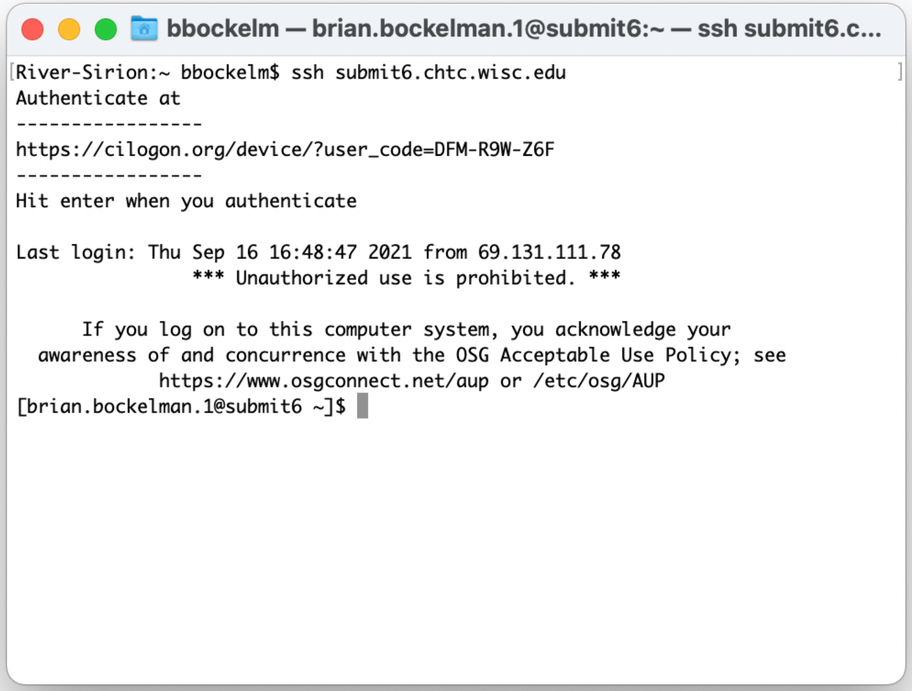
Bonus topic #1: CHTC pool

While this talk is about the OSPool, we often preview new functionality at smaller pools.

The **CHTC glidein pool** for UW-Madison researchers and collaborators:

- Modest number of patches to GlideinWMS to improve token generation.
- Recently upgraded to a HTCondor 9.3.0 release candidate which has GSI support disabled at compile time.
  - AP's and EP's can join our pool via IDTOKENS or SSL but not GSI.
- Removed all use of GSI authentication from the GlideinWMS frontend.
  - Proxy delegation is still used so the factory can submit pilots to CEs without token support.
    - Proxy delegation in HTCondor is implemented with pure OpenSSL.
  - Frontend<->Factory communication done with IDTOKENS.
- Hopefully can start reaching out to sites to enable SciToken-based pilot submits.

Bonus Use Case: JWT's for SSH



We use pam_oauth2_device and the OAuth2 device flow for SSH logins.

- Instead of a password at the terminal, they are given a CILogon URL to visit.
- The user opens the URL in their browser and authenticates with CILogon.
- If the browser login is successful, the SSH server receives a token from CILogon.
  - Once validated, the username is extracted from the JWT and the user can login.

morgridge.org

This material is based upon work supported by the National Science Foundation under Grant No. 1836650, 2030508, and 2114989. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

FEARLESS SCIENCE