GlideinWMS approach to token auth

Marco Mambelli, Dennis Box - Fermilab

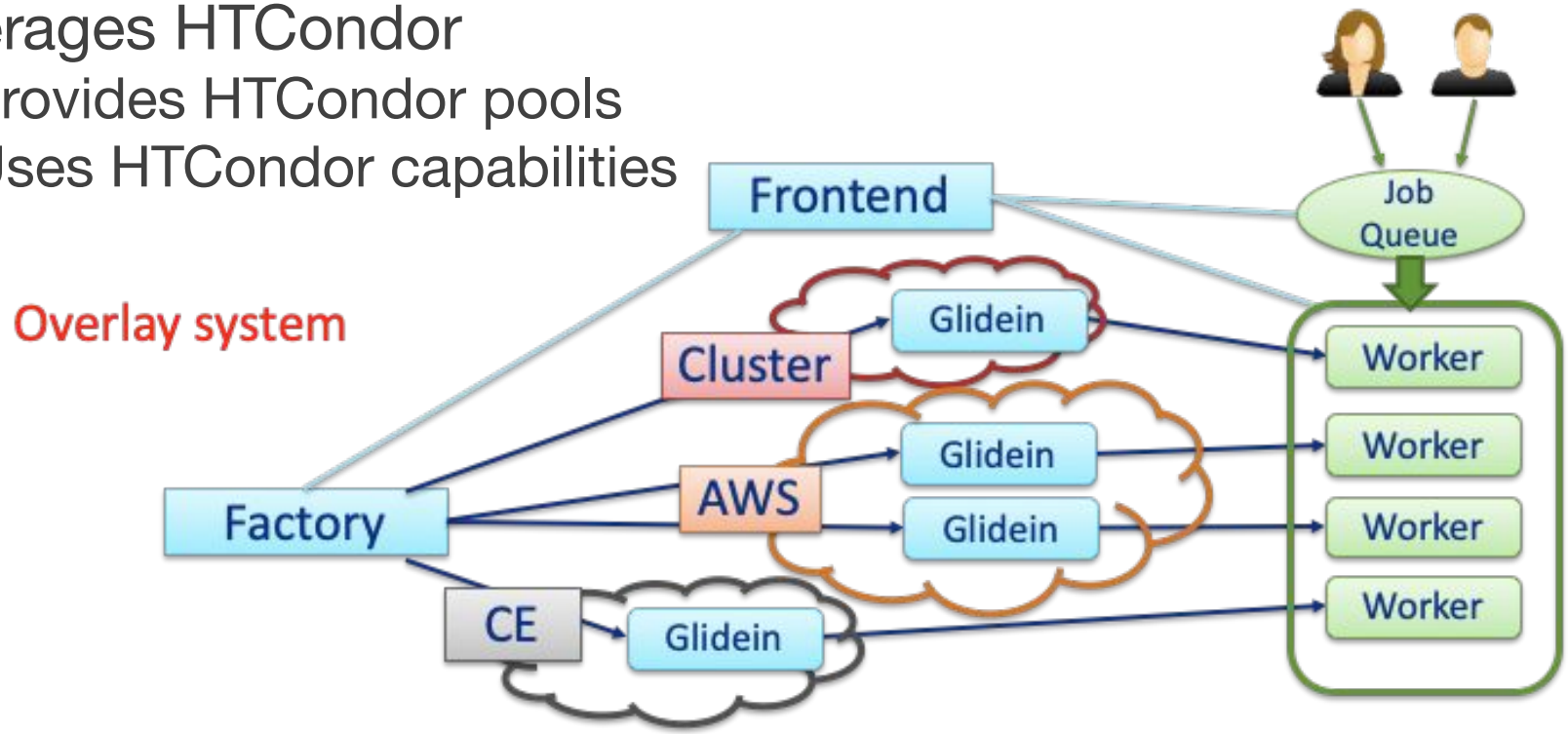
June 3, 2021

WLCG CE and Factory Token Hackathon

CERN

GlideinWMS

- GlideinWMS is a pilot based resource provisioning tool for distributed High Throughput Computing
- Provides reliable and uniform virtual clusters
- Submits Glideins to unreliable heterogeneous resources
- Leverages HTCondor
  - Provides HTCondor pools
  - Uses HTCondor capabilities



Frontend

- **Manages credentials and delegates them to the Factory**
  - **Managed by VO: stores and owns VO credentials**
  - **Long term credentials, forwarding short term ones**
  - **Manual input, interacts with IAMs**
- Monitors jobs to see how many Glideins are needed
- Compares what entries (sites) are available
- Requests Glideins from the Factory
- Requests Factory to kill Glideins if there are too many
- Pressure-based system
  - Works keeping a certain number of Glideins running or idle at the sites
  - Gradual Glideins requests to avoid spikes and overloads

Factory

- **Keeps a cache of credentials used or forwarded to Glideins**
- A Glidein Factory knows how to submit to sites
  - Sites are described in a local configuration
  - Only trusted and tested sites are included
- Each site entry in the configuration contains
  - Contact info (hostname, resource type, queue name)
  - Site configuration (startup dir, OS type, …)
  - VOs authorized/supported
  - Other attributes (Site name, core count, max memory, ...)
  - Glideins can also auto-detect resources
- Configuration can be auto-generated (e.g. from CRIC), admin curated, stored in VCS (e.g. GitHub)
- Condor does the heavy lifting of submissions.

Glidein: node testing and customization

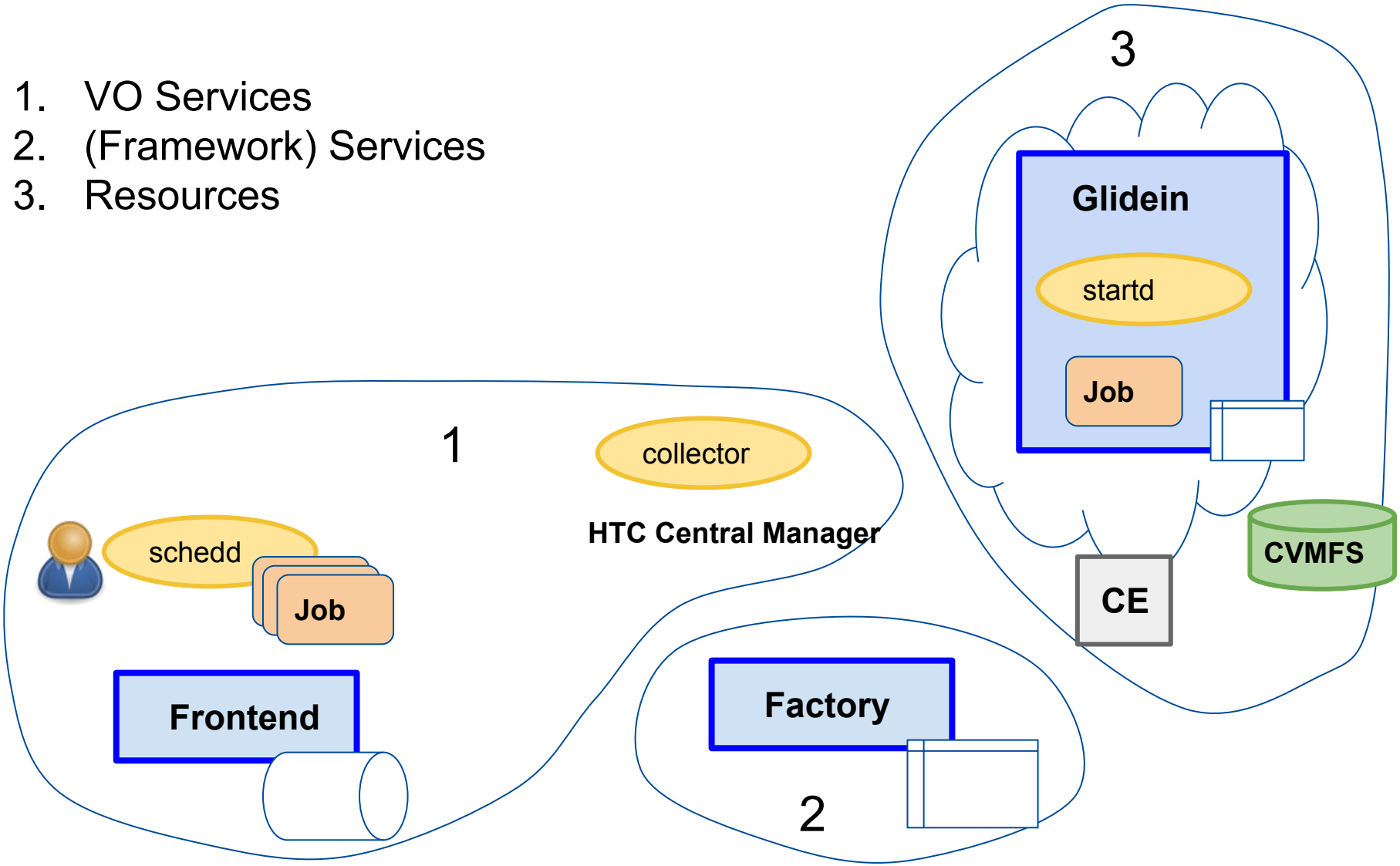
- Scouts for resources and validates the Worker node
  - Cores, memory, disk, GPU, …
  - OS, software installed
  - CVMFS
  - VO specific tests
- Customizes the Worker node
  - Environment, GPU libraries, …
  - Starting containers (Singularity, …)
  - VO specific setup
- Provides a reliable and customized execute node to HTCondor
- Reports back to the Factory
- **Pilot (e.g. VO Group) credentials storage and use**
- **Safely receive and store Job credentials**

GlideinWMS and IAM

- Carrier of different credentials, both identity or capability based
  - must support different resources
- Agnostic about the type
  - provider and service have to be compatible
- Internally using IDTOKENS (heavily reliant on HTCondor)

GlideinWMS system

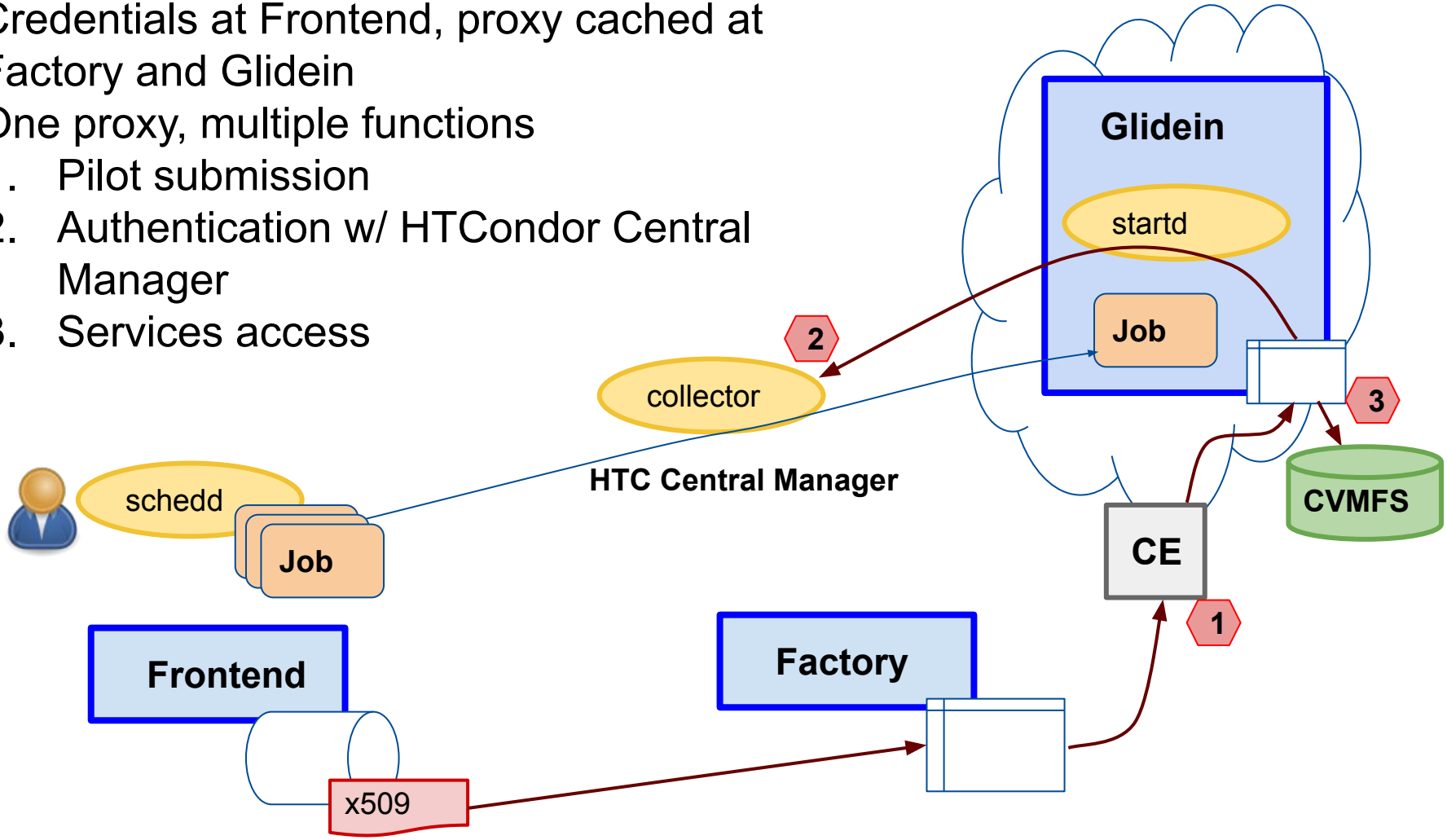
1. VO Services
2. (Framework) Services
3. Resources



Traditional x509 authentication - Pilot proxy

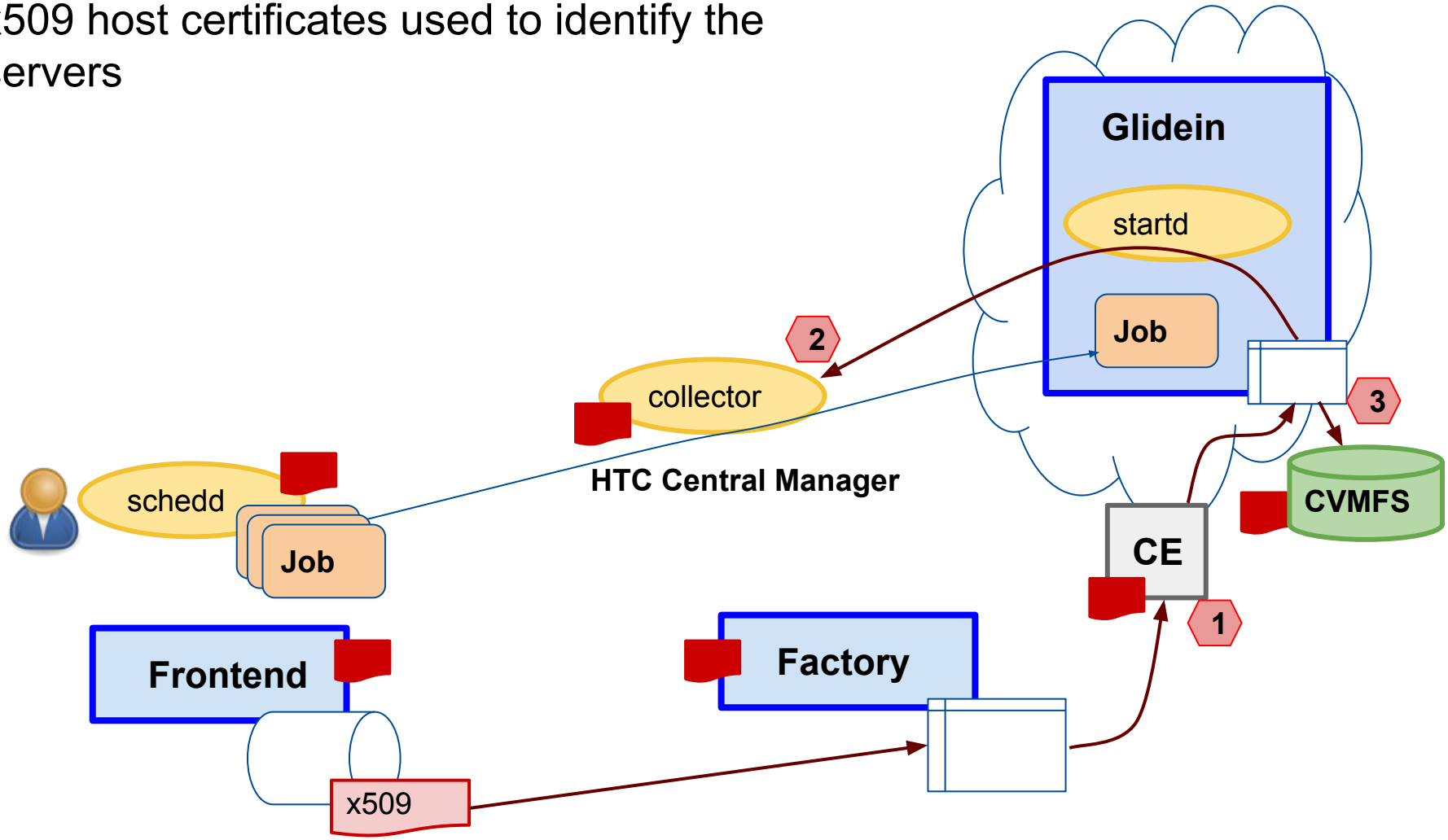
Credentials at Frontend, proxy cached at Factory and Glidein
One proxy, multiple functions

1. Pilot submission
2. Authentication w/ HTCondor Central Manager
3. Services access



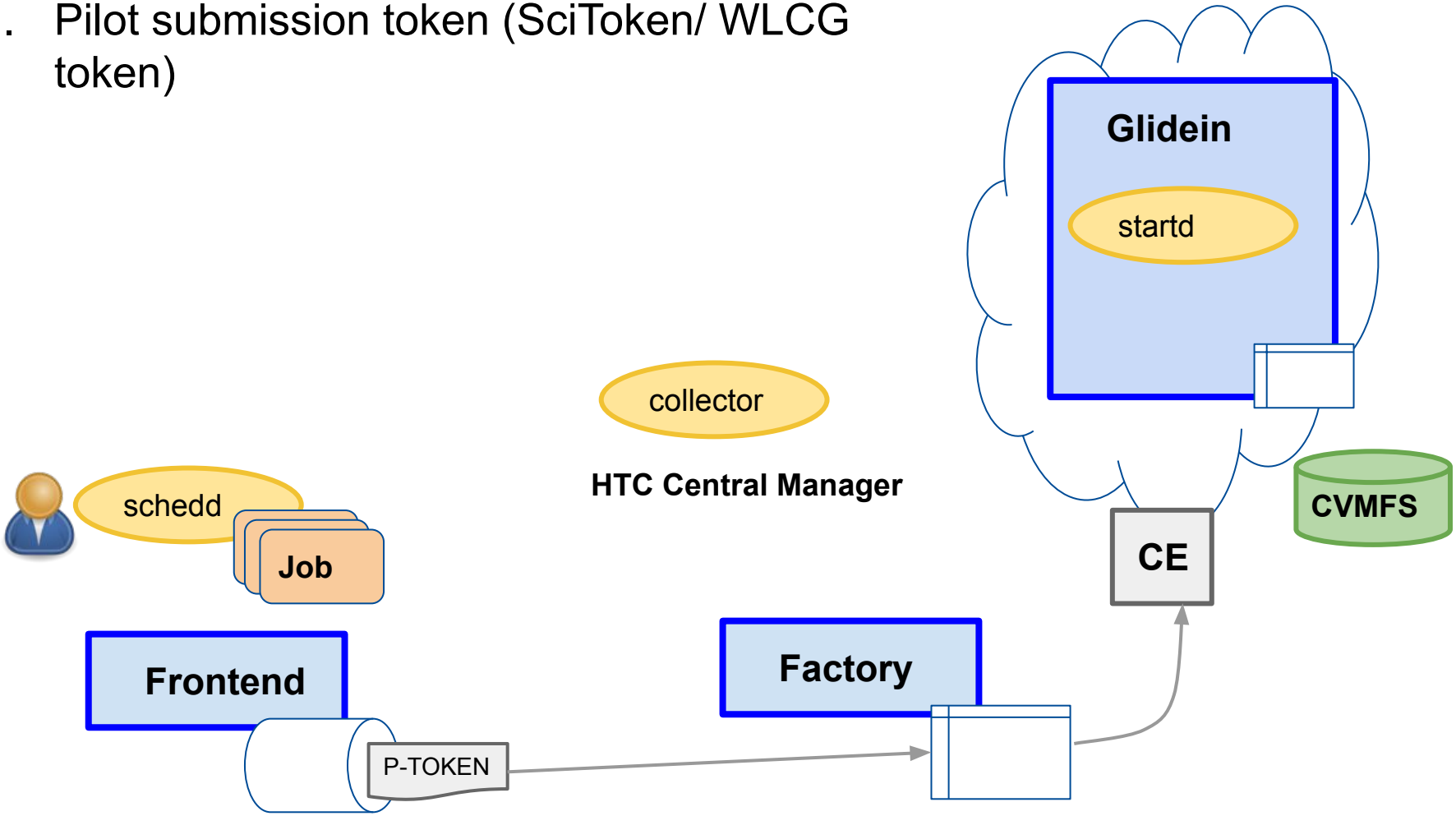
Traditional x509 authentication - Host certificates

x509 host certificates used to identify the servers



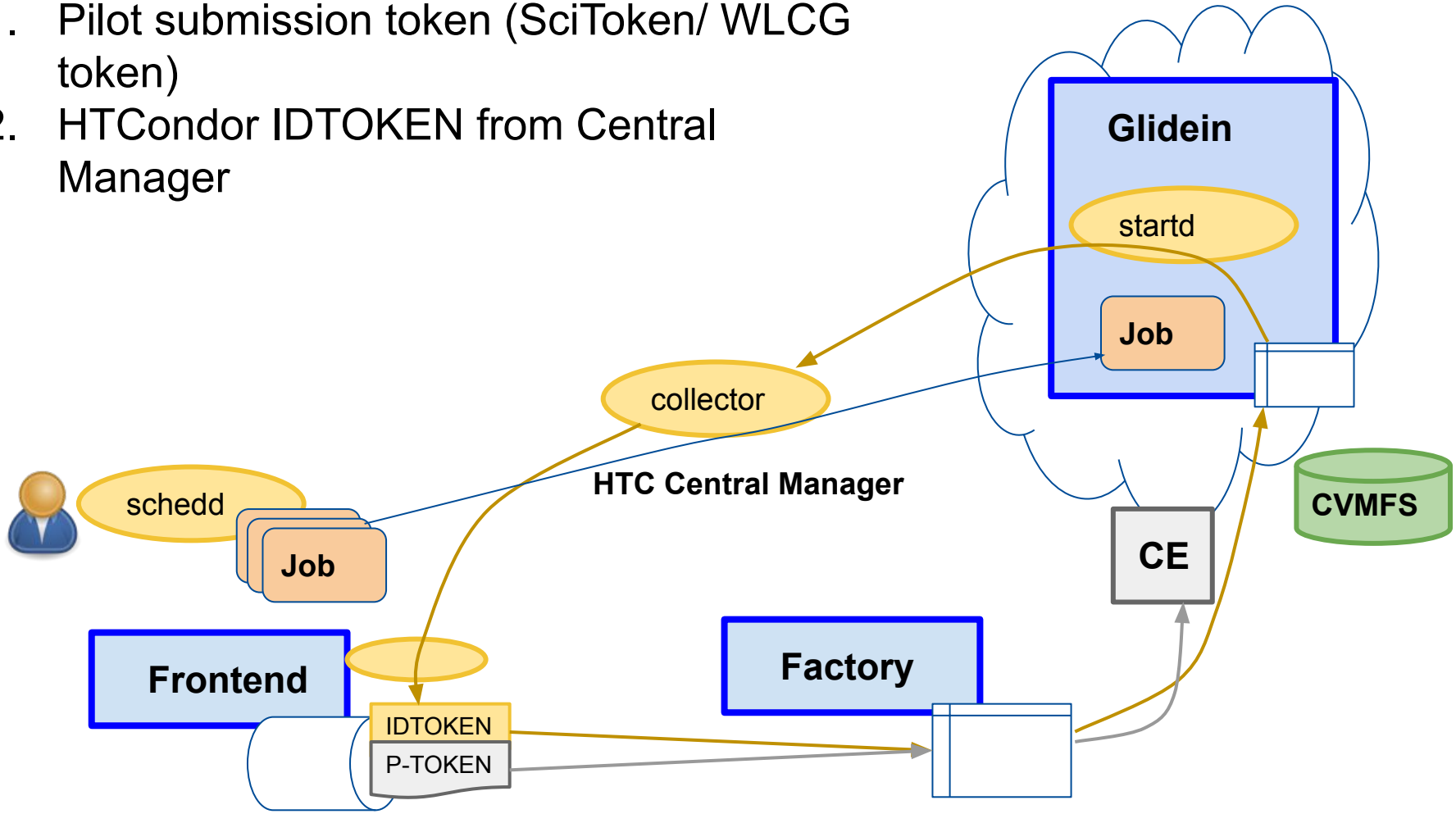
Token authentications

1. Pilot submission token (SciToken/ WLCG token)



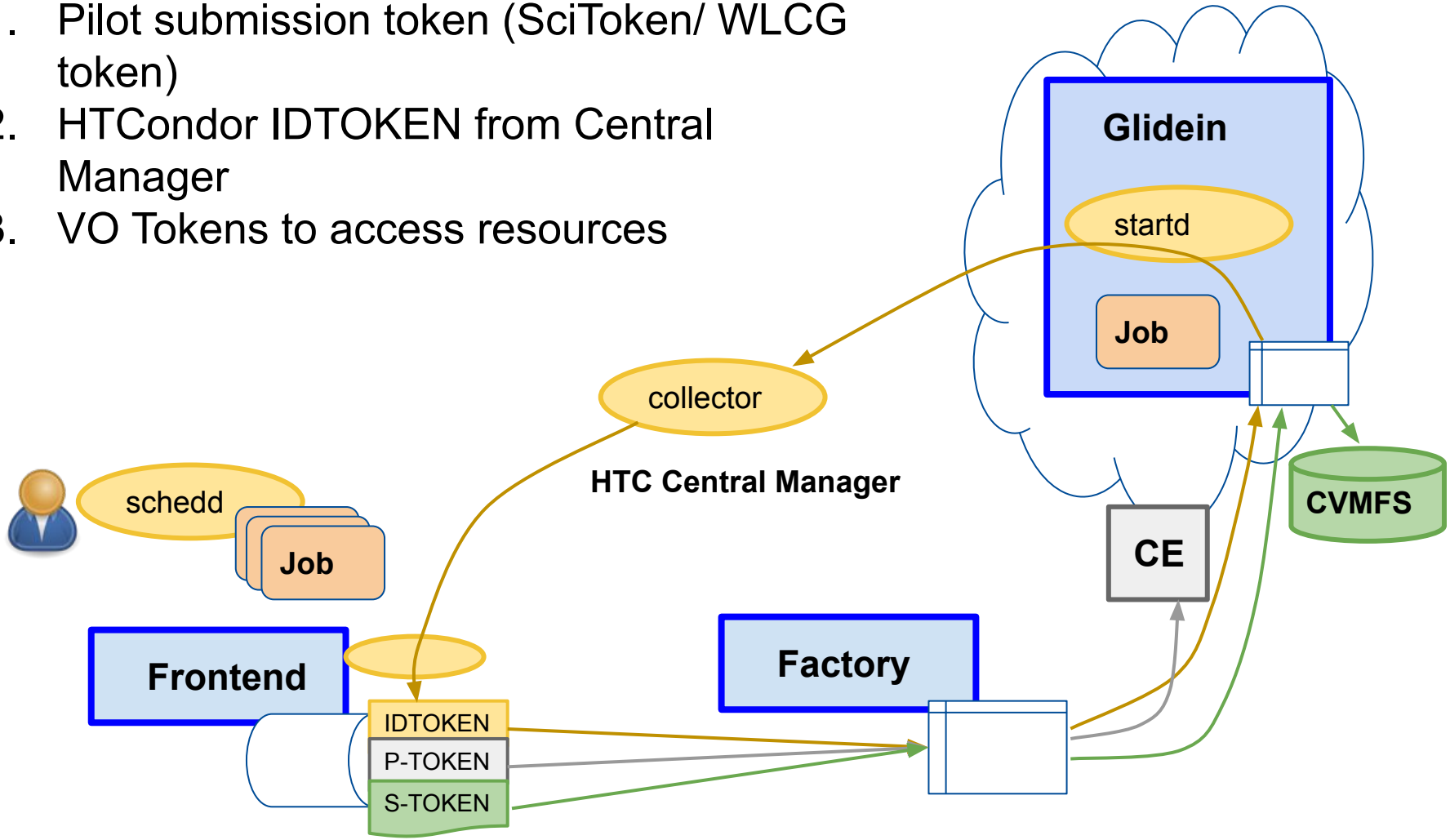
Token authentications

1. Pilot submission token (SciToken/ WLCG token)
2. HTCondor IDTOKEN from Central Manager



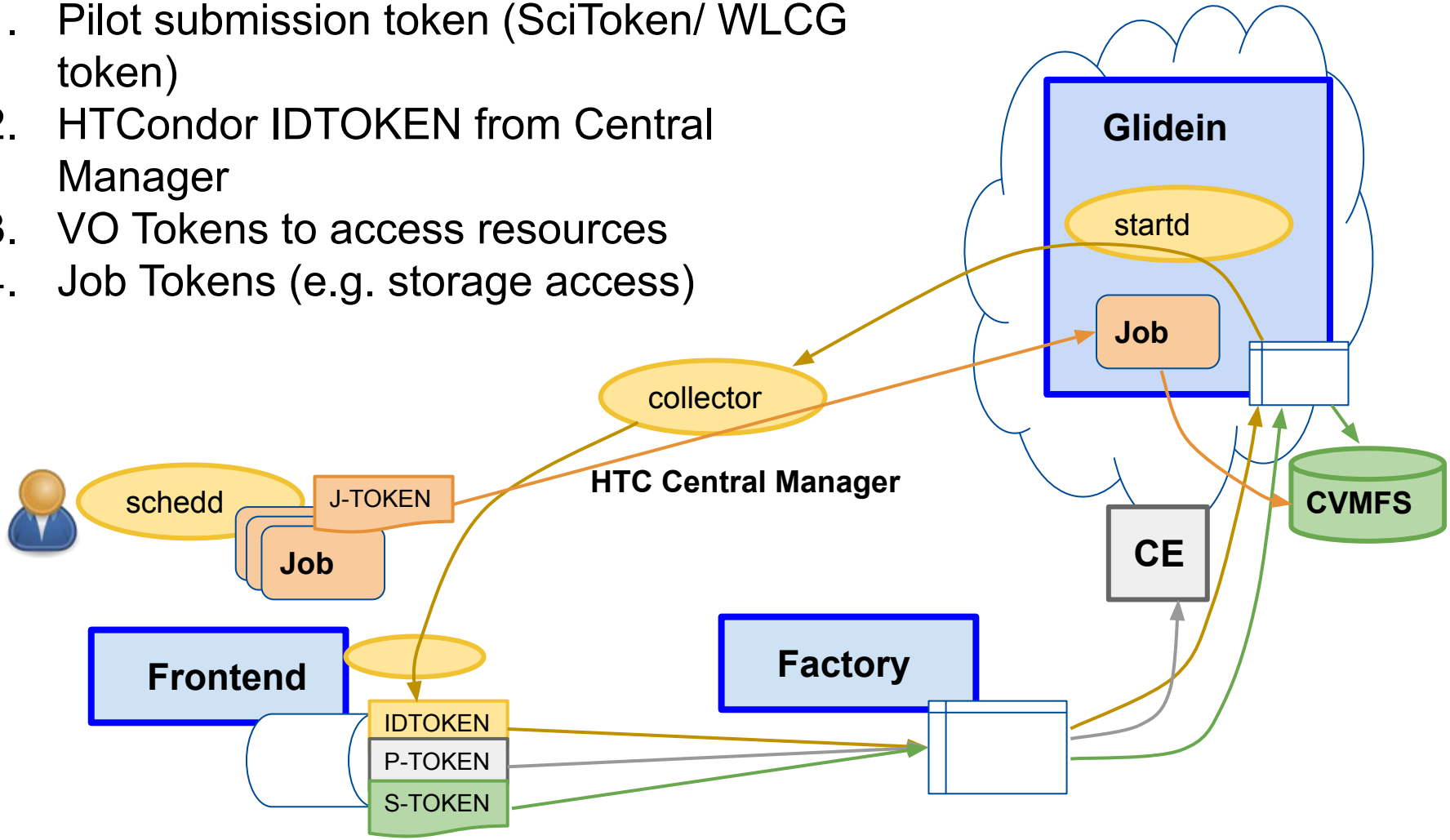
Token authentications

1. Pilot submission token (SciToken/ WLCG token)
2. HTCondor IDTOKEN from Central Manager
3. VO Tokens to access resources



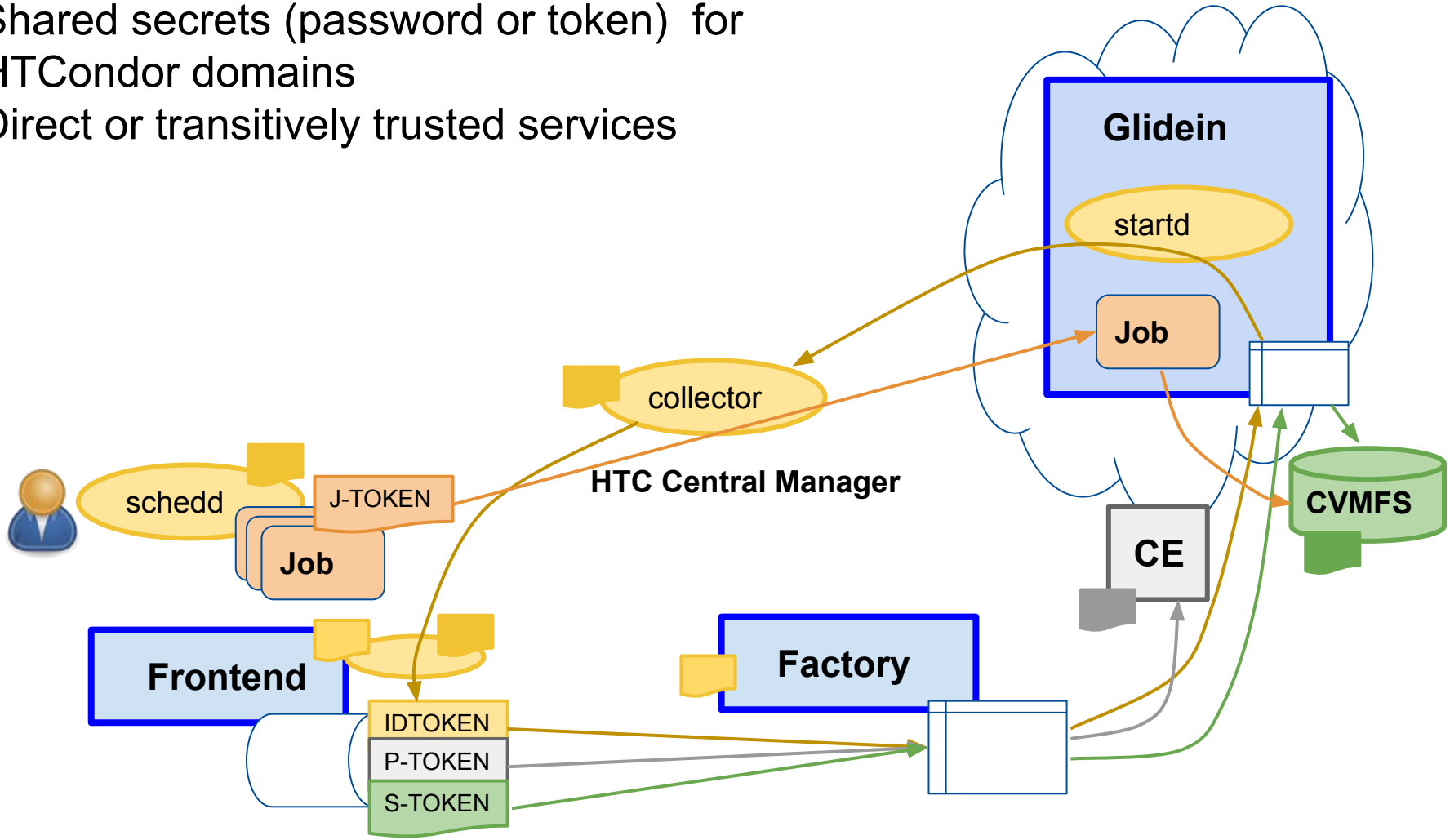
Token authentications

1. Pilot submission token (SciToken/ WLCG token)
2. HTCondor IDTOKEN from Central Manager
3. VO Tokens to access resources
4. Job Tokens (e.g. storage access)



Token authentications - services

Shared secrets (password or token) for HTCondor domains
Direct or transitively trusted services



VO credentials

- Currently treated as files
- Created independently
  - Request from WLCG INDIGO IAM via oidc-agent
  - Requested from CILogon or WLCG INDIGO IAM via htgettoken
  - Create a self-signed SciToken via scitoken-admins-... commands
- Renewal works between the Frontend and the Factory
- Will add some tool to ease creation, handling and renewal
  - Integration with INDIGO IAM, Vault

Framework credentials use HTCondor (mostly)

- IDTOKEN created on the Factory (condor_token_create) copied to the Frontend
- Frontend owns a copy of the VO pool password to create per Site tokens for the Glideins
  - tokens can be invalidated changing the generating password
  - renewal mechanism will allow short lived tokens
- Monitoring servers issue JWT token for the Glideins

Example 1: VO Credential

```
+ sudo -u dbox htgettoken --minsecs=3590 -v -a fermicloud346.fnal.gov -o /tmp/foo.token
Credkey from /cloud/login/dbox/.config/htgettoken/credkey-default-default:dbox@fnal.gov
Attempting to get bearer token from https://fermicloud346.fnal.gov:8200
  using vault token from /tmp/vt_u8531
  at path secret/oauth-default/creds/dbox@fnal.gov:default
Storing bearer token in /tmp/foo.token
+ cp /tmp/foo.token /var/lib/gwms-frontend/tokens.d/el7_osg35.scitoken
+ chown -R frontend:frontend /var/lib/gwms-frontend/tokens.d
+ chmod 600 /var/lib/gwms-frontend/tokens.d/el7_osg35.scitoken
```

VO Credential Example (continued)

```
+[root@fermicloud081 ~]# ~dbox/bin/decode_jwt /var/lib/gwms-frontend/tokens.d/el7_osg35.scitoken
{
  "kid": "rsa1",
  "alg": "RS256"
}
{
  "wlcg.ver": "1.0",
  "sub": "a75aeb7a-04d6-44c3-8e71-ac39cf70d103",
  "aud": "https://wlcg.cern.ch/jwt/v1/any",
  "nbf": "2021-06-02T19:54:57Z",
  "scope": "storage.create:/ openid offline_access profile storage.read:/ storage.modify:/ email wlcg wlcg.groups",
  "iss": "https://wlcg.cloud.cnaf.infn.it/",
  "exp": "2021-06-02T20:54:57Z",
  "iat": "2021-06-02T19:54:57Z",
  "jti": "b0ae8ab3-80ea-476b-a186-5eecb8bbbd00",
  "client_id": "cd5623e3-16da-4156-a35f-c0b9e4190e04",
  "wlcg.groups": [
          "/wlcg",
          "/wlcg/pilots",
          "/wlcg/xfers"
  ]
}
```

VO credential config in frontend.xml

```
 <credential
absfname="/var/lib/gwms-frontend/tokens.d/el7_osg35.scitoken"
security_class="frontend" trust_domain="OSG" type="scitoken"/>
```

- Scitoken will be encrypted and passed to factory, which will use it to authenticate on behalf of VO to CE
- Matches to factory entries using trust_domain and security_class the same as other credential types

VO credential in Factory

Decrypted SCITOKEN in factory lives at

/var/lib/gwms-factory/client-proxies/<frontend uid name>
/glidein_gfactory_instance/credential_<frontend_name>.<fe_entry_name>.scitoken

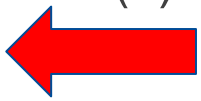
Factory Entry Configuration:

<entry name="el7_osg35" auth_method="scitoken" enabled="True"
gatekeeper="fermicloud127.fnal.gov fermicloud127.fnal.gov:9619"
gridtype="condor" rsl="(queue=default)(jobtype=single)"
schedd_name="fermicloud173.fnal.gov" trust_domain="OSG" verbosity="fast"
work_dir="OSG"/>

If auth_method = ‘grid_proxy’ and a scitoken passed in will try that first

VO credential mapping at CE

```
[root@fermicloud127 ~]# cat /etc/condor-ce/condor_mapfile
SCITOKENS "https://jobsub.fnal.gov(.*)nova(.*)" nova
SCITOKENS "https://jobsub.fnal.gov(.*)fermilab(.*)" fermilab
SCITOKENS "https://jobsub.fnal.gov(.*)osg(.*)" osg
SCITOKENS "https://test.cilogon.org/fermilab(.*)dbox(.*)" dbox
SCITOKENS https://wlcg.cloud.cnaf.infn.it osg
SCITOKENS https://jobsub.fnal.gov/minerva/Analysis minervaana
```



This is htcondor-ce 4.3.1. The new htcondor-ce 5 series has much more flexible mapping features.

Example 2: IDTOKEN generation

The Frontend automatically generates IDTOKENS that are used by the glideins to authenticate back to the frontend

```
[root@fermicloud044 ~]# ls /var/lib/gwms-frontend/tokens.d
el7_osg35.idtoken  el7_osg35.scitoken
```

auto generated

VO generated (Example 1)

No config changes are needed to frontend or factory for IDTOKENS

Example 2: IDTOKEN (cont)

```
[root@fermicloud044 ~]# ~dbox/bin/decode_jwt_human
/var/lib/gwms-frontend/tokens.d/el7_osg35.idtoken
{
  "alg": "HS256",
  "typ": "JWT",
  "kid": "FRONTEND"
}
{
  "sub": "vofrontend_service@fermicloud044.fnal.gov",
  "iss": "fermicloud044.fnal.gov:9618",
  "jti": "012005216f4a4398a3bc17c148fa34e7",
  "exp": "Thu Jun 3 15:47:53 CDT 2021",
  "scope": "condor:/READ condor:/ADVERTISE_STARTD condor:/ADVERTISE_MASTER",
  "iat": "Wed Jun 2 15:47:53 CDT 2021",
  "nbf": "Wed Jun 2 15:47:53 CDT 2021"
}
```

IDTOKEN on Factory

The IDTOKEN is encrypted and passed to the factory.

When decrypted, it lives at /var/lib/gwms-factory/client-proxies/user_<frontend_uid>/<glidein_entry_name>.idtoken

[root@fermicloud361]# ls -la /var/lib/gwms-factory/client-proxies/user_frontend044/glidein_gfactory_instance/el7_osg35.idtoken

-rw-------. 1 gfactory e875 440 Jun 2 16:01 /var/lib/gwms-factory/client-proxies/user_frontend044/glidein_gfactory_instance/el7_osg35.idtoken

IDTOKEN in glidein

The IDTOKEN is encrypted becomes part of the glide payload.

The glidein attempts to authenticate to the Frontend Collector using the IDTOKEN.

The glidein can be configured to attempt to fallback to GSI authentication if IDTOKEN auth fails

Acknowledgements

This work was done under the GlideinWMS project

This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

References

https://github.com/glideinWMS/glideinwms (branch_v3_7)