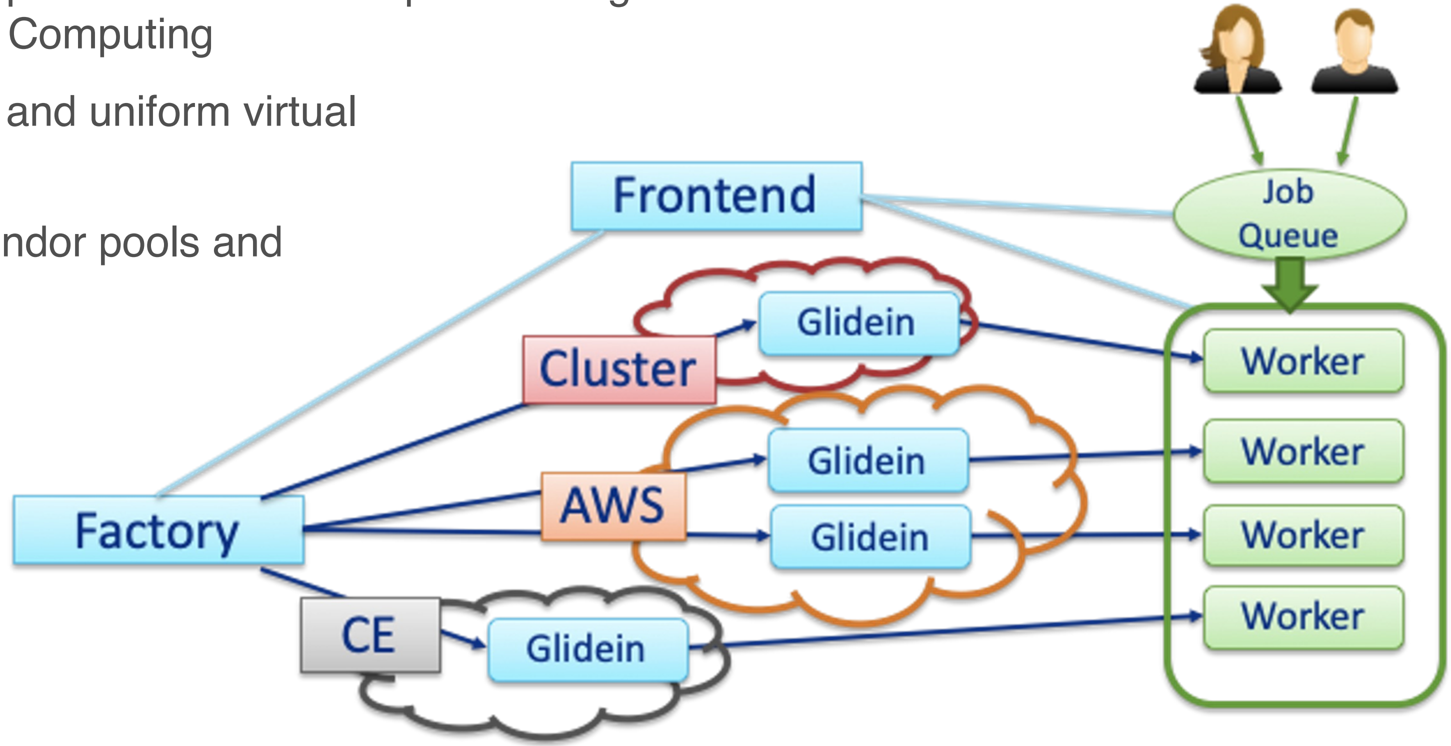
The new GlideinWMS credentials model and the new challenges it presents

Bruno Coimbra

FERMILAB-SLIDES-24-0167-CSAID

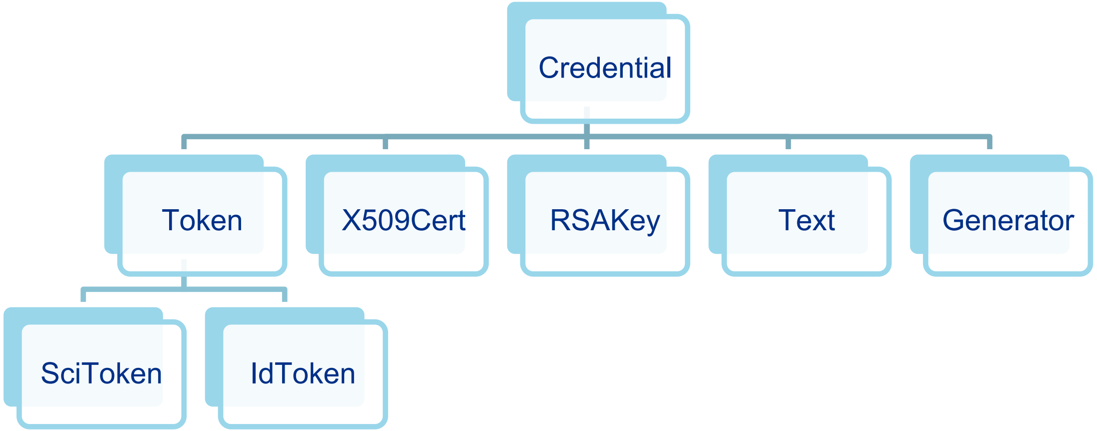
GlideinWMS

- GlideinWMS is a pilot-based resource provisioning tool for distributed High Throughput Computing
- Provides reliable and uniform virtual clusters
- Leverages HTCondor pools and capabilities



New GlideinWMS Credentials Model

- Hierarchical credential types
  - All credentials have a set of common methods and attributes
  - Factory and Frontend operations are agnostic to the credential type
  - Credential types can be inferred from a string or file



New GlideinWMS Credentials Model

- Credential Purposes
  - Determine how a credential is used (e.g. CE authentication)
  - Allow to create list of credentials to be sent along with the Glidein

- Credential Parameters
  - Credential qualifiers are now independent parameters (e.g. VMID, VMTYPE)

- New auth_method implementation

```
auth_method="scitoken,grid_proxy;vmid;vmtype"
```

New GlideinWMS Credentials Model

- Generators
  - A new generator framework extend the current functionality of credential generators
  - The same framework can be used to generate security parameters
  - Built-in generators can be used straight from a configuration file

```
<credential
      absfname="RoundRobinGenerator”
      context="{'items': ['cred1', 'cred2', 'cred3'], 'type': 'text'}"
      purpose="payload"
      security_class="frontend"
      trust_domain="grid"
      type="generator”
/>

<parameter
      name="VMId”
      value="RoundRobinGenerator"
      context="{'items': ['vm1', 'vm2', 'vm3'], 'type': 'string'}"
      type="generator"
/>
```

Challenge: identifying a token type

- Inferring credential types brings a new challenge when dealing with tokens
  - It’s easy to infer a credential is a token (JWT)
  - It’s not so easy to determine which token type (SciToken vs IDTOKEN)

- Some token attributes give us hints of its type
  - SciTokens and IDTOKENs use different encryption methods
  - “jti” along with the issuer might help us differentiate both formats
  - SciTokens usually define “ver” as “scitoken:2.0”

- It would be helpful to specify claims to help us identify these token types from generic JWTs
  - WLCG tokens require “wlcg.ver”

Challenge: providing fallback credentials to HTCondor CE

- The new auth_method allow for multiple credentials to be used to authenticate with CEs

- HTCondor CE accepts a list of authentication methods
  - Used only during the negotiation phase
  - The first accepted credential (and only the first) is used for authentication
  - The CE won’t try other credentials of the agreed type if the first one fails
  - The CE won’t try other authentication methods if the first one fails

- If some kind of authentication method fallback was implemented on the CE we could reduce the complexity of GlideinWMS submissions

Challenge: renewing tokens after submitting a job

- Ongoing discussion…

- Tokens live shorter than grid proxies
  - Payload tokens submitted with Glideins will often expire while the it's in queue

- A mechanism to renew tokens after Glideins are submitted is necessary
  - We could implement a solution on the GlideinWMS side
  - A native HTCondor implementation of this feature would be cleaner and address other use cases

Thank You!

- New challenges:
  - Identify token types
  - Implement credentials fallback on CEs
  - Renew SciTokens after jobs are submitted

This work was produced 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. Publisher acknowledges the U.S. Government license to provide public access under the DOE Public Access Plan DOE Public Access Plan