What is a Multi-User Compute Endpoint?
What is a Multi-User Compute Endpoint?

● Aside … hereafter:
  ○ MEP → Multi-User Endpoint
  ○ UEP → User Endpoint (“normal endpoint”)
What is a Multi-User Compute Endpoint?

- Aside … hereafter:
  - MEP → Multi-User Endpoint
  - UEP → User Endpoint ("normal endpoint")
- In contrast to a “normal” compute endpoint, an MEP does not run tasks.
What is a Multi-User Compute Endpoint?

- Aside ... hereafter:
  - MEP → Multi-User Endpoint
  - UEP → User Endpoint (“normal endpoint”)

- In contrast to a “normal” compute endpoint, an MEP does not run tasks.

- Instead, an MEP
  - starts UEPs
  - (Slightly more precisely, fork, drop privileges, exec)
  - Manages their lifecycle (okay, `os.fork()` and `os.waitpid()`)

What is a Multi-User Compute Endpoint?

- Aside … hereafter:
  - MEP → Multi-User Endpoint
  - UEP → User Endpoint (“normal endpoint”)

- In contrast to a “normal” compute endpoint, an MEP does not run tasks.

- Instead, an MEP
  - starts UEPs
  - (Slightly more precisely, fork, drop privileges, exec)
  - Manages their lifecycle (okay, `os.fork()` and `os.waitpid()`)

- Receives start UEP commands from the web-service
PDF NOTE: Original presentation had a live screen recording, showing the values updating in real time as “presentation-proof” that the software exists (if not yet released). See speaker notes.

Video of original presentation linked via the ParslFest 2023 list of presentations.  
(https://parsl-project.org/parslfest/parslfest2023.html)

<table>
<thead>
<tr>
<th>PID/USER</th>
<th>RES</th>
<th>S</th>
<th>CPU%</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>1923814</td>
<td>root</td>
<td>126M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924147</td>
<td>kevin</td>
<td>142M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924217</td>
<td>kevin</td>
<td>124M</td>
<td>S</td>
<td>1.3</td>
</tr>
<tr>
<td>1924233</td>
<td>kevin</td>
<td>123M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924243</td>
<td>kyle</td>
<td>142M</td>
<td>S</td>
<td>0.6</td>
</tr>
<tr>
<td>1924266</td>
<td>kyle</td>
<td>124M</td>
<td>S</td>
<td>0.6</td>
</tr>
<tr>
<td>1924282</td>
<td>kyle</td>
<td>123M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924303</td>
<td>harper</td>
<td>142M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924368</td>
<td>harper</td>
<td>125M</td>
<td>S</td>
<td>1.3</td>
</tr>
<tr>
<td>1924384</td>
<td>harper</td>
<td>123M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924399</td>
<td>jessica</td>
<td>142M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924419</td>
<td>jessica</td>
<td>124M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924435</td>
<td>jessica</td>
<td>123M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924445</td>
<td>rowan</td>
<td>142M</td>
<td>S</td>
<td>0.0</td>
</tr>
<tr>
<td>1924472</td>
<td>rowan</td>
<td>124M</td>
<td>S</td>
<td>1.3</td>
</tr>
<tr>
<td>1924488</td>
<td>rowan</td>
<td>123M</td>
<td>S</td>
<td>0.0</td>
</tr>
</tbody>
</table>
How do we *do* it?
Admin Writes/Controls

```yaml
engine:
  type: GlobusComputeEngine

provider:
  type: SlurmProvider
  partition: cpu
  account: {{ ACCOUNT_ID }}

launcher:
  type: SrunLauncher

walltime: {{ walltime|default("00:30:00") }}
```

`user_config_template.yaml`
Admin Writes/Controls

```yaml
engine:
  type: GlobusComputeEngine

provider:
  type: SlurmProvider
  partition: cpu
  account: {{ ACCOUNT_ID }}

launcher:
  type: SrunLauncher

walltime: {{ walltime|default("00:30:00") }}
```

User Script

```python
import globus_compute_sdk as GC

uep_conf = {
  "ACCOUNT_ID": "314159265",
  "walltime": "00:02:00"
}

with GC.Executor(
  endpoint_id=mep_id,
  user_endpoint_config=uep_conf
) as gce:
  fut = gce.submit(some_func)
  res = fut.result()
```

`user_config_template.yaml`
Admin Writes/Controls

```
engine:
  type: GlobusComputeEngine

provider:
  type: SlurmProvider
  partition: cpu
  account: {{ ACCOUNT_ID }}

launcher:
  type: SrunLauncher

walltime: {{ walltime|default("00:30:00") }}
```

User Script

```
import globus_compute_sdk as GC

uep_conf = {
    "ACCOUNT_ID": "543126688"
}

with GC.Executor(
    endpoint_id=mep_id,
    user_endpoint_config=uep_conf
) as gce:
    fut = gce.submit(some_func)
    res = fut.result()
```

`user_config_template.yaml`
Two different configurations; same user!
Value-Add for Users
Value-Add for Users

- No need to maintain multiple endpoints for different configurations
Value-Add for Users

- No need to maintain multiple endpoints for different configurations
- Specify needs *at task submission*
Value-Add for Users

- No need to maintain multiple endpoints for different configurations
- Specify needs **at task submission**
- No need to log in to the terminal
Value-Add for Site Administrators
Value-Add for Site Administrators

- Templatable User Endpoint Configurations (Jinja)
  - e.g., pre-choose SlurmProvider, PBSProvider; enforce limits
Value-Add for Site Administrators

- Templatable User Endpoint Configurations (Jinja)
  - e.g., pre-choose SlurmProvider, PBSProvider; enforce limits

- No orphaned user compute endpoints
  - Enforced process tree
  - Idle-endpoints are shutdown (per template configuration)
Value-Add for Site Administrators

- Templatable User Endpoint Configurations (Jinja)
  - e.g., pre-choose SlurmProvider, PBSProvider; enforce limits

- No orphaned user compute endpoints
  - Enforced process tree
  - Idle-endpoints are shutdown (per template configuration)

- Standard Globus Identity Mapping
Value-Add for Site Administrators

- Templatable User Endpoint Configurations (Jinja)
  - e.g., pre-choose SlurmProvider, PBSProvider; enforce limits
- No orphaned user compute endpoints
  - Enforced process tree
  - Idle-endpoints are shutdown (per template configuration)
- Standard Globus Identity Mapping
- Lower barrier for users
Current status

- We’re buttoning up a few details
- Have not yet written any documentation
- Looking for brave volunteers to give it go
Thank You!

- Questions?
- Comments?
- Synergistic thoughts?