

# Globus Compute Multi-User Endpoints



Reid Mello  
reid@globus.org



THE UNIVERSITY OF  
CHICAGO



 ParslFest 2024



Argonne  
NATIONAL LABORATORY 

globus



## Value-Add for Users

- No need to maintain multiple endpoints for different configurations
- Specify configuration **at task submission**
- No need to log into the target computer

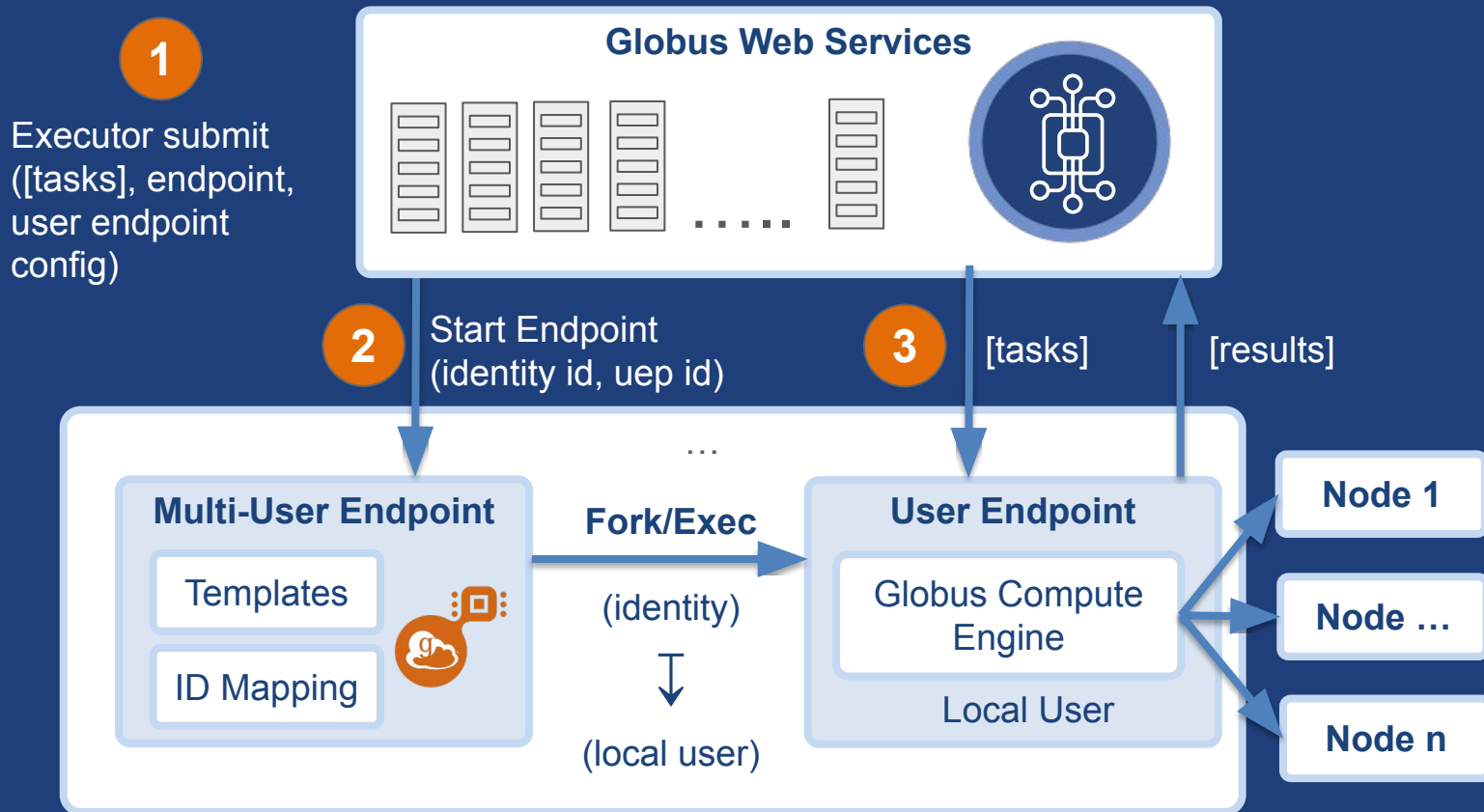


# Value-Add for Administrators

- Lower barrier for users
- Templatable (controllable) user endpoint configurations
  - E.g., pre-choose SlurmProvider, PBSProvider; enforce limits
- No orphaned user compute endpoints
  - Enforced process tree
  - Idle endpoints are shut down (per endpoint configuration)
- Standard Globus Identity Mapping



# Multi-user Endpoints: Architecture





# Multi-user: Installation

```
# curl -LOs https://downloads.globus.org/globus-connect-server/stable/installers/repo/deb/globus-repo_latest_all.deb
# dpkg -i globus-repo_latest_all.deb
# apt-key add /usr/share/globus-repo/RPM-GPG-KEY-Globus

# apt update
# apt install globus-compute-agent
```

```
# globus-compute-endpoint configure my-ep --multi-user
Created multi-user profile for endpoint named <my-ep>
```

```
Configuration file: /root/.globus_compute/my-ep/config.yaml
```

```
Example identity mapping configuration: /root/.globus_compute/my-ep/example_identity_mapping_config.json
```

```
User endpoint configuration template: /root/.globus_compute/my-ep/user_config_template.yaml.j2
```

```
User endpoint configuration schema: /root/.globus_compute/my-ep/user_config_schema.json
```

```
User endpoint environment variables: /root/.globus_compute/my-ep/user_environment.yaml
```

Use the `start` subcommand to run it:

```
$ globus-compute-endpoint start my-ep
```



# Using the multi-user endpoint

```
# globus-compute-endpoint start my-ep
```

```
def hello_world():  
    return "Hello, World!"  
  
with Executor(endpoint_id="...") as gce:  
    future = gce.submit(hello_world)  
    print(future.result())
```

**GlobusAPIError:** ('POST',  
'https://compute.api.globus.org/v3/endpoints/07301555-e7c6-4f36-bbe7-c1963fc27909/submit', 'Bearer', 422,  
'SEMANTICALLY\_INVALID', 'Request payload failed validation: Identity failed to map to a local user name.  
(LookupError) \n Globus effective identity: 082d6a19-da16-4552-9944-e081cdaff7bc\n Globus username:  
082d6a19-da16-4552-9944-e081cdaff7bc@clients.auth.globus.org')



# Multi-user: Identity Mapping

*Same format as GCSv5*

/root/.globus\_compute/my-ep/example\_identity\_mapping\_config.json

```
[
  {
    "comment": "For more examples, see: https://docs.globus.org/globus-connect-server/v5.4/identity-mapping-guide/",
    "DATA_TYPE": "expression_identity_mapping#1.0.0",
    "mappings": [
      {
        "source": "{username}",
        "match": "(.*)@uchicago\\.edu",
        "output": "{0}"
      }
    ]
  }
]
```

[https://docs.globus.org/globus-connect-server/v5.4/identity-mapping-guide/#default\\_identity\\_to\\_username\\_mapping](https://docs.globus.org/globus-connect-server/v5.4/identity-mapping-guide/#default_identity_to_username_mapping)



# Using the multi-user endpoint

```
# globus-compute-endpoint start my-ep
```

```
def hello_world():  
    return "Hello, World!"  
  
with Executor(endpoint_id="...") as gce:  
    future = gce.submit(hello_world)  
    print(future.result())
```

```
$ python hello_world.py  
Hello, World!
```





# Multi-user: User Configuration Template

/root/.globus\_compute/my-ep/user\_config\_template.yaml.j2

```
engine:
  type: GlobusComputeEngine

provider:
  type: SlurmProvider
  partition: cpu
  account: {{ ACCOUNT_ID }}
  walltime: {{ WALLTIME|default("00:30:00") }}

launcher:
  type: SrunLauncher
```

```
from globus_compute_sdk import Executor

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

with Executor(endpoint_id="...") as gce:
    gce.user_endpoint_config = uep_conf
    fut = gce.submit(hello_world)
    res = fut.result()
```



# Multi-user: User Configuration Schema

/root/.globus\_compute/my-ep/user\_config\_schema.json

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "type": "object",
  "properties": {
    "ACCOUNT_ID": { "type": "string" },
    "WALLTIME": { "type": "string" }
  },
  "additionalProperties": false
}
```



# Restricting access to endpoints

## **Cloud-enforced: Authentication policies**

- Cloud gate keeps submission to the endpoint
- E.g., domain restrictions, high assurance policies

## **Endpoint-enforced: Identity Mappings**

- Map user identities to local accounts



# Multi-user: Authentication Policies

```
globus-compute-endpoint configure my-ep \  
  --auth-policy-project-id 8236ad07-2801-468a-b262-9f1814988cc5 \  
  --auth-policy-display-name "Globus Staff Only" \  
  --allowed-domains "*.globus.org" \  
  --auth-timeout 60 \  
  --subscription-id 964be8f5-5f9b-11e4-b64e-12313940394d \  
  --multi-user
```

### Edit Policy Details

Display Name\*

Description\*

High Assurance  User's identity must be authenticated within current browser session.

Authentication Timeout

Time allowed before reauthentication is required.

Included Domains

One domain per line - may include wildcards, e.g. "\*.edu". If left blank, any domain will satisfy this policy.

Excluded Domains

One domain per line - may include wildcards, e.g. "\*.edu".



# Multi-user: Authentication Policies

/root/.globus\_compute/my-ep/config.yaml

```
amqp_port: 443
display_name: Demo Endpoint
identity_mapping_config_path: /root/.globus_compute/my-ep/example_identity_mapping_config.json
multi_user: true
authentication_policy: d6071efc-c182-432d-a757-0fd8d975146c
```






# Multi-user: Restricting Functions

/root/.globus\_compute/my-ep/config.yaml

```
amqp_port: 443
display_name: Demo Endpoint
identity_mapping_config_path: /root/.globus_compute/my-ep/example_identity_mapping_config.json
multi_user: true
allowed_functions:
  - 94c7ce20-7a5a-4eb3-ac07-6fc0aabcd50c
  - aab66753-4b02-4162-a9d3-47374dabcdc4
```



```
def safe_hello_world():
    return "Hello, Safe World!"

with Executor(endpoint_id="...") as gce:
    function_id = gce.register_function(safe_hello_world)
    fut = gce.submit_to_registered_function(function_id=function_id)
    res = fut.result()
```



# Multi-user: Enable on boot

```
# globus-compute-endpoint enable-on-boot my-ep
Systemd service installed at /etc/systemd/system/globus-compute-endpoint-my-ep.service. Run
    sudo systemctl enable globus-compute-endpoint-my-ep --now
to enable the service and start the endpoint.
```


```
[Unit]
Description=Globus Compute Endpoint "my-ep"
After=network.target
StartLimitIntervalSec=0



[Service]
ExecStart=/opt/globus-compute-agent/venv-py39/bin/globus-compute-endpoint start my-ep
User=root
Type=simple
Restart=always
RestartSec=1

[Install]
WantedBy=multi-user.target
```




# Globus Compute Web App

 my-ep

Refresh in 17  

STATUS	OUTSTANDING TASKS	PENDING TASKS	TOTAL WORKERS	IDLE WORKERS	MANAGERS
<span style="color: green;">●</span> online	0	-	-	-	-

**Display Name** my-ep

**ID** 78b56e0e-5d41-4cd9-8eae-4cb815a74bbd 

**Description** (not set)


**IP Address** 72.36.119.43

**Hostname** 269c6034de95

**Compute Endpoint Version** 2.271

User Config Schema User Config Template Config

```
1 {
2   "type": "object",
3   "$schema": "https://json-schema.org/draft/2020-12/schema",
4   "properties": {
5     "worker_init": {
6       "type": "string"
7     },
8     "endpoint_setup": {
9       "type": "string"
10    }
11  },
12  "additionalProperties": true
13 }
```



- FILE MANAGER
- ACTIVITY
- COLLECTIONS
- GROUPS
- CONSOLE
- FLAWS
- COMPUTE**
- SETTINGS
- LOGOUT
- HELP & SITEMAP



# Any questions?

Docs: <https://globus-compute.readthedocs.io/en/latest/>

GitHub: <https://github.com/globus/globus-compute>

Slack: <https://funcx.slack.com/>

