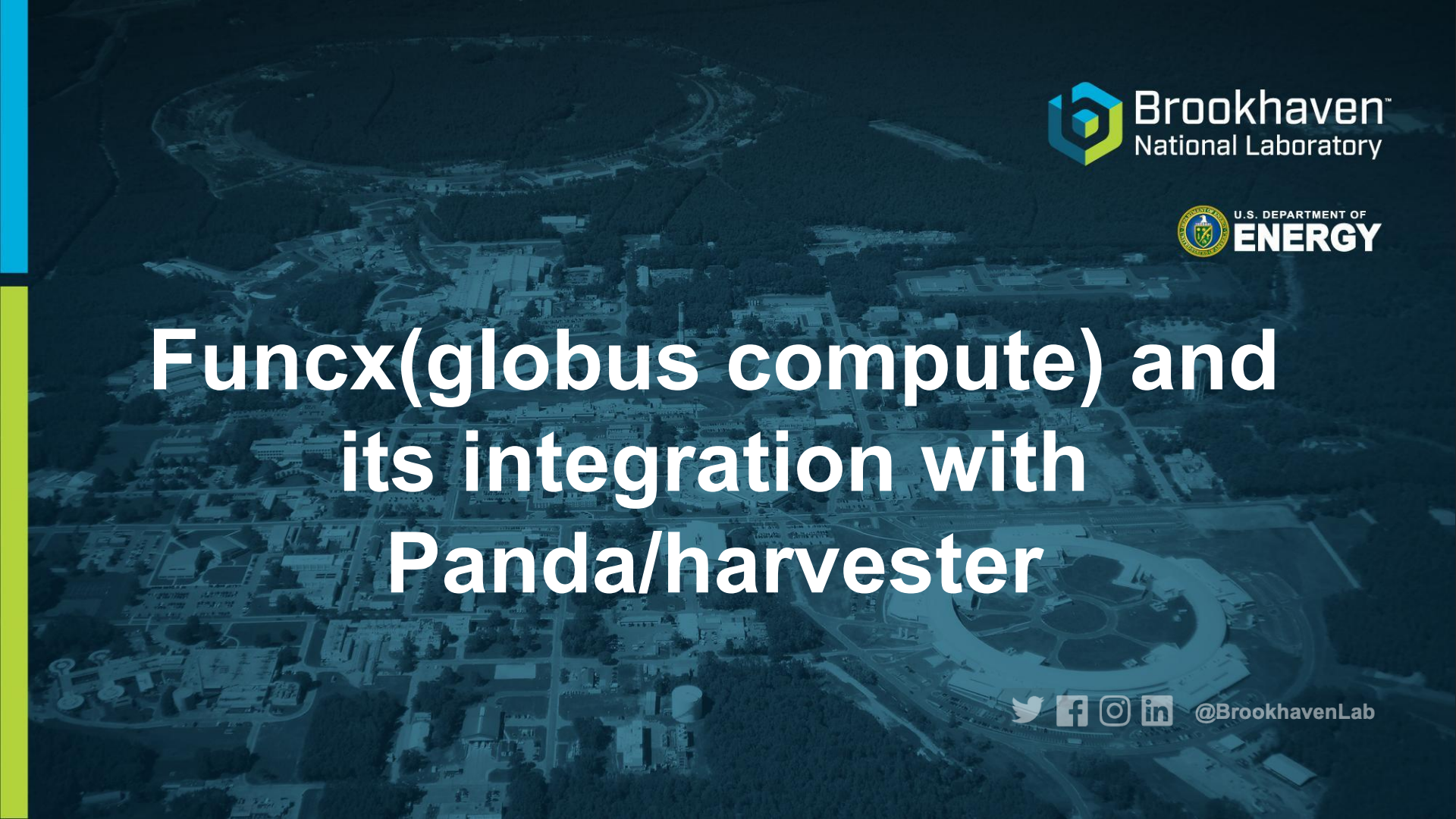
Brookhaven
National Laboratory
U.S. DEPARTMENT OF
ENERGY
Funcx(globus compute) and its integration with Panda/harvester
@BrookhavenLab

What is funcx

- A distributed Function as a Service (FaaS) platform that enables reliable, scalable, and high performance remote function execution.
- Enables execution functions on diverse remote systems, from laptops to campus clusters, clouds, and supercomputers.

All four endpoints runs successfully on local machine with

python main.py

```
from globus_compute_sdk import Executor

def add_func(a, b):
    return a + b

endpoint_id = '733ef4d2-4e13-409d-9e30-c6955b4d209d'    #Perlmutter, default
#endpoint_id = '94c6a085-9d80-48d0-9a92-a5c36e67b468'    #BNL lambda machine
#endpoint_id = 'd7036086-52b5-41f6-85a4-b8fbb306e232'    #Polaris
#endpoint_id = '733ef4d2-4e13-409d-9e30-c6955b4d209d'    #Localhost

with Executor(endpoint_id=tutorial_endpoint_id) as gce:
    future = gce.submit(add_func, 5, 10)
    print(future.result())                                #Should be 15
```

Advantage

- Single deploy and can be used in future without authentication needed.
- Easy to config on different platforms with different architecture and job scheduler
  - Different number of cores, number of GPU, GPU vendor
  - mpiexec, srun, pbspro
  - One config (endpoint) can be reused for different task/workflow
- Both blocking and non-blocking API allows for easy integration with workflow management system to handle workflow with complicated DAG

2-)Saadi, Aymen Al, et al. "Impeccable: Integrated modeling pipeline for covid cure by assessing better leads." *50th International Conference on Parallel Processing*. 2021.

Other Useful features

- Register function and container
  - Allows code/resource reuse
  - Decrease overhead
- Function sharing within group
- Batch submission

How to solve dependency (environment)

- Real task usually have complicated dependency
  - C++ program needs many library like ROOT, boost
  - Python script needs many packages like tensorflow, numpy
  - User have their own library
- Earlier work by Wen simply copy all source files using panda cache
- Latest funcx has two supports: Container and conda env
  - For python task, conda env is the easiest and prove working
  - For C++ task, container could be an option
- Currently funcx only support docker, singularity, apptainer. However, perlmutter uses shifter.
  - Temporary solution is to hack config file accordingly on perlmutter

Current issue: MPI

- Funcx does not officially support MPI
- The underlying executor is based on Parsl, which only support MPI recently
- Try to use subprocess to wrap srun job, failed on pm.
  - Even if it works, not extendable: For each MPI executor, need to wrap function accordingly.
- Develop team confirm a new MPI executor is WIP
- Another option might be to use the MPI executor from other service like Radical

Other issues

- Lack of detailed resource provision option
  - Heterogeneous asynchronous workflow requires heterogeneous resource binding with tasks
  - Even for homogeneous workflow, different binding options can change performance significantly
- Task specific configuration needed
  - Currently need to setup multiple config / endpoint
  - Can not config remotely
  - Develop team confirms that a multi-user endpoint feature will be ready soon which allows modifying config on a per-task basis

Current integration with panda/harvester

- Wen has integrate funcx with harvester with some ML example, where the dependency is solved by copying all files using panda cache
- With customized conda env, ML tasks with customized can be run without source file movement and reuse above code
- With the temporary solution of using shifter container, simulation task implemented using C++ can be run without much code rewritten
- The data dependency can either be accomplished by panda/harvester, or can be handled by funcx.
  - Intra-endpoint: In-memory data store
  - Inter-endpoint: Globus
- Need a real workflow to test if the above implementation and ideas works