Stitching Together Innovation with FABRIC Users

When Federated Learning Meets FABRIC

Zilinghan Li, Ze Yang, Ravi Madduri

12/17/2024

Welcome and Introduction

Zilinghan Li
Argonne National Laboratory |
Machine Learning Engineer

Ze Yang
University of Illinois at Urbana-Champaign |
Visiting Research Associate

Ravi Madduri
Argonne National Laboratory |
Senior Computer Scientist

Federated Learning

Motivations for Using Federated Learning

Federated learning (FL)'s Motivations:

- ➢ Sufficient data is essential to training high-quality and generalized AI models.
- ➢ Data within a single data silo are usually limited and biased.
- ➢ Data are likely distributed heterogeneously among various data silos.
- ➢ Directly collecting data from distributed silos are sometimes impossible due to privacy concerns, especially in sensitive domains such as finance and biomedicine.
- ➢ Federated Learning enables the training of more generalized model without direct data sharing.

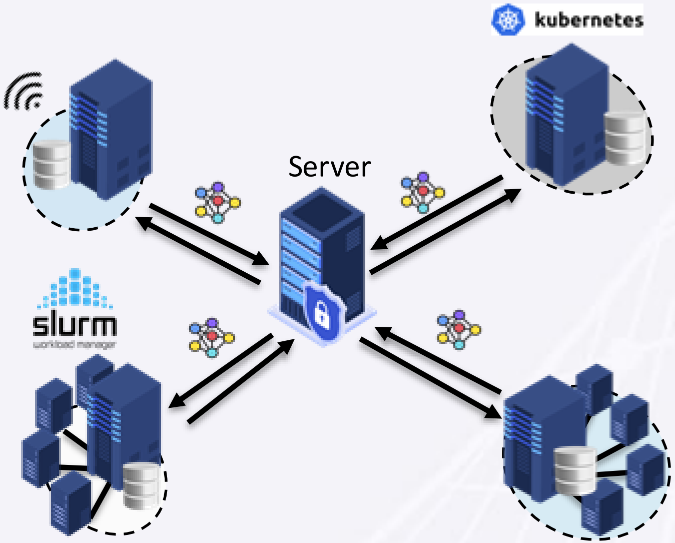


Fig. Federated Learning Illustration.

Federated Learning

Distributed Machine Learning Paradigm without Direct Data Sharing

Federated learning (FL)'s Workflow:

- ➢ Distributed machine learning paradigm
- ➢ Multiple *clients* with own computing resources and private local data
- ➢ One central orchestration *server*
- ➢ Each client trains a local model and shares the model with the server for aggregation
- ➢ The aggregated model leverages data from multiple clients to obtain more generalized model without direct data sharing

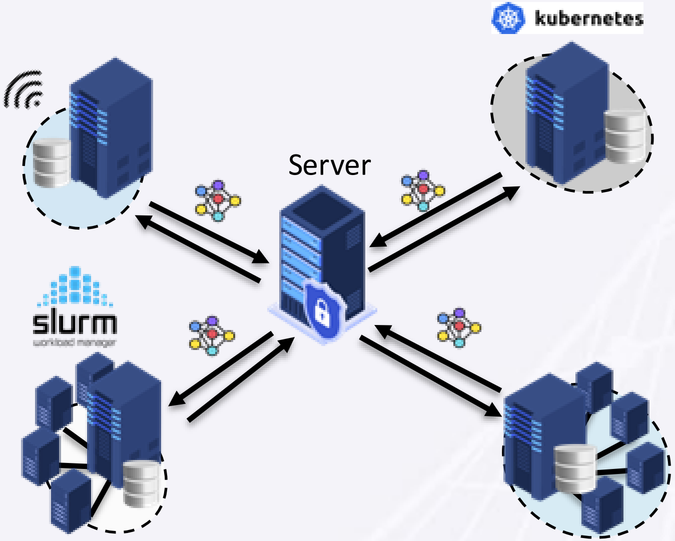
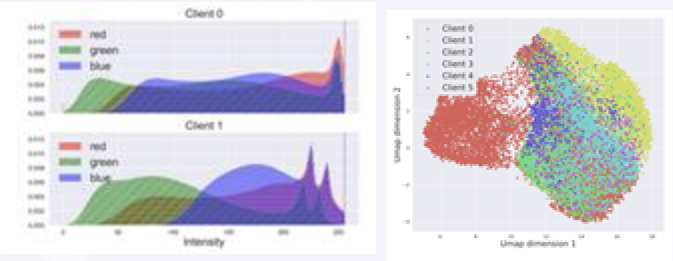


Fig. Federated Learning Illustration.

Federated Learning

Various Challenges of Federated Learning Due to its Distributed Nature

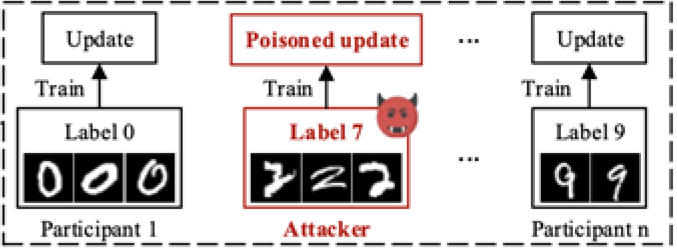
Advanced Aggregation Strategies



➢ Heterogeneous Data

➢ Heterogeneous Compute

Advanced Asynchronous Scheduling Algorithms



➢ Malicious Attack

Robust Authentication

➢ Data Reconstruction

Differential Privacy and Privacy Enhancing Techniques

➢ Cumbersome Setup

Simple Exp Configuration

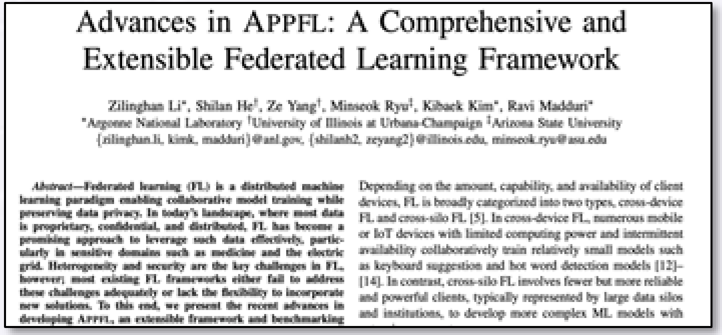
APPFL

APPFL (Advanced Privacy-Preserving Federated Learning) Framework is our framework-wise solutions to those challenges.

APPFL Framework

APPFL: Advanced Privacy-Preserving Federated Learning Framework

APPFL is an open-source FL framework which supports comprehensive solutions for various FL challenges.

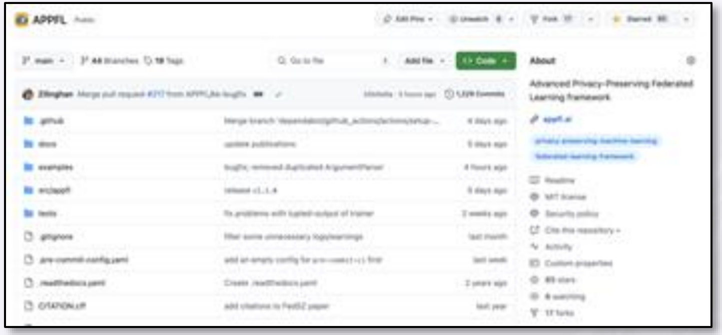


Advances in APPFL: A Comprehensive and Extensible Federated Learning Framework

Zilinghan Li[*], Shilan He[†], Ze Yang[†], Minseok Ryu[‡], Kibaek Kim[*], Ravi Madduri[*]
[*]Argonne National Laboratory [†]University of Illinois at Urbana-Champaign [‡]Arizona State University
{zilinghan.li, kimk, madduri}@anl.gov, {shilanh2, zeyang2}@illinois.edu, minseok.ryu@asu.edu

Abstract—Federated learning (FL) is a distributed machine learning paradigm enabling collaborative model training while preserving data privacy. In today's landscape, where most data is proprietary, confidential, and distributed, FL has become a promising approach to leverage such data effectively, particularly in sensitive domains such as medicine and the electric grid. Heterogeneity and security are the key challenges in FL, however, most existing FL frameworks either fail to address these challenges adequately or lack the flexibility to incorporate new solutions. To this end, we present the recent advances in developing APPFL, an extensible framework and benchmarking

Depending on the amount, capability, and availability of client devices, FL is broadly categorized into two types, cross-device FL and cross-silo FL [5]. In cross-device FL, numerous mobile or IoT devices with limited computing power and intermittent availability collaboratively train relatively small models such as keyboard suggestion and hot word detection models [12]–[14]. In contrast, cross-silo FL involves fewer but more reliable and powerful clients, typically represented by large data silos and institutions, to develop more complex ML models with



APPFL Public

Zilinghan Merge pull request #[illegible] from APPFL/[illegible]-bugfix

.github	Merge branch '[illegible]/github_actions/actions/setup-…	4 days ago
docs	update publications	5 days ago
examples	bugfix: removed duplicated ArgumentParser	4 hours ago
src/appfl	release v1.1.4	5 days ago
tests	fix problems with [illegible] output of trainer	2 weeks ago
.gitignore	filter some unnecessary logs/warnings	last month
.pre-commit-config.yaml	add an empty config for pre-commit-ci first	last week
.readthedocs.yaml	Create .readthedocs.yaml	2 years ago
CITATION.cff	add citations to FedS2 paper	last year

About

Advanced Privacy-Preserving Federated Learning framework

appfl.ai

[Manuscript](#)

[Open-source Code](#)

Framework Design Description 📄

- ➢ Framework overview
- ➢ Addressed challenges
- ➢ Evaluations
- ➢ Additional case studies
- ➢ …

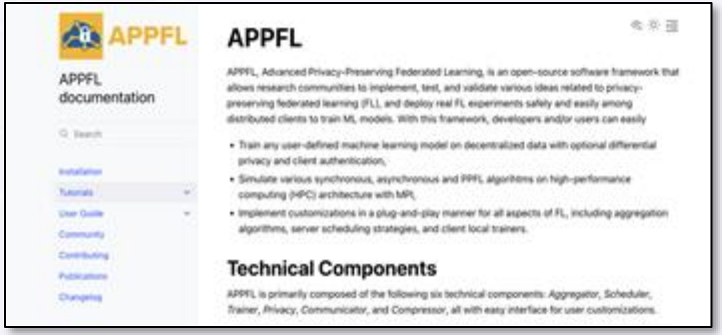
Source code on Github

- ➢ Fully open-source
- ➢ Welcome issues
- ➢ Welcome contributions
- ➢ …

APPFL Framework

APPFL: Advanced Privacy-Preserving Federated Learning Framework

APPFL is an open-source FL framework which supports comprehensive solutions for various FL challenges.



APPFL

APPFL, Advanced Privacy-Preserving Federated Learning, is an open-source software framework that allows research communities to implement, test, and validate various ideas related to privacy-preserving federated learning (FL), and deploy real FL experiments safely and easily among distributed clients to train ML models. With this framework, developers and/or users can easily

- Train any user-defined machine learning model on decentralized data with optional differential privacy and client authentication,
- Simulate various synchronous, asynchronous and PPFL algorithms on high-performance computing (HPC) architecture with MPI,
- Implement customizations in a plug-and-play manner for all aspects of FL, including aggregation algorithms, server scheduling strategies, and client local trainers.

Technical Components

APPFL is primarily composed of the following six technical components: *Aggregator*, *Scheduler*, *Trainer*, *Privacy*, *Communicator*, and *Compressor*, all with easy interface for user customizations.

appfl.ai

Detailed Documentation 📜
- ➢ Installation
- ➢ Launching FL experiments
- ➢ Advanced Developer Guides
- ➢ …

Building AI Models For Science

Privacy-Preserving Federated Learning for Science: Building Sustainable and Trustworthy Foundation Models.

GET STARTED

Learn how to start an FL experiment using APPFLx.

service.appfl.ai

APPFL-based Service Platform 🚀
- ➢ Fully based on APPFL
- ➢ User-friendly for domain experts
- ➢ Comprehensive report generation
- ➢ …

APPFL x FABRIC

Using FABRIC as the Testbed for APPFL

Benefits provided by FABRIC:

➢ *Heterogeneous and Distributed Testbed*

FABRIC offers a heterogeneous and geographically distributed testbed for testing and benchmarking distributed computing applications.

➢ *Enhanced Understanding of Network Role in FL*

FABRIC enables exploration of the role of networks in training models across geographically distributed sites, especially when working with large datasets at each location.

➢ *Inbound Connection Support*

Unlike many supercomputers (e.g., Polaris), FABRIC supports servers which allows inbound connections from clients, making it ideal for hosting FL servers.

APPFL × FABRIC

APPFL x FABRIC

Using FABRIC as the Testbed for APPFL

Benefits provided by FABRIC:

➢ *Comprehensive Monitoring Tools*

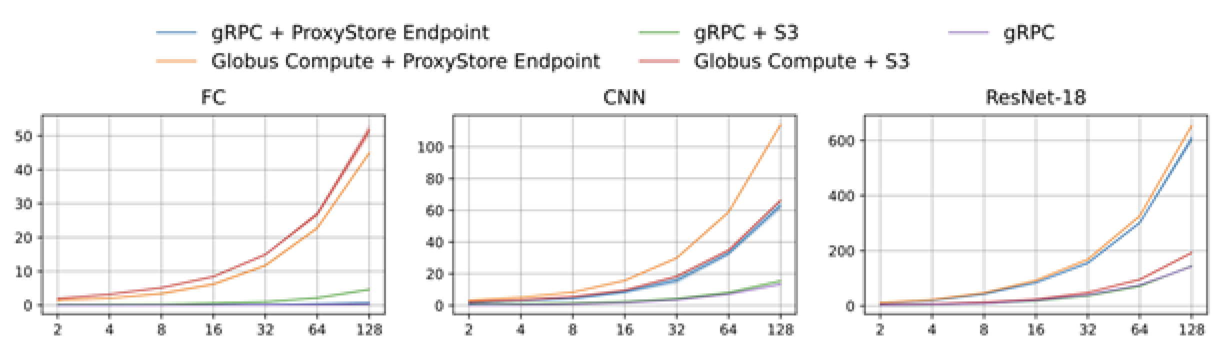
FABRIC provides advanced tools for measuring latency and monitoring resource utilization.

➢ *Connectivity to External Facilities*

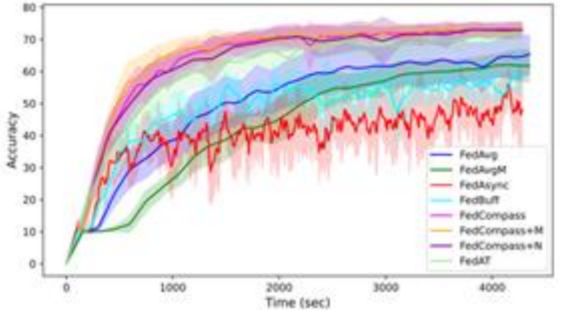
FABRIC allows seamless connection to external facilities for additional computing resources, for example, Chameleon Facility Ports help access additional computing resources for compute-intensive applications.

APPFL x FABRIC

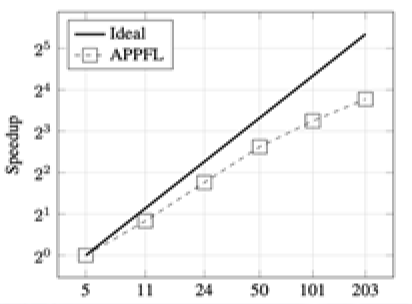
Experiments Enabled by FABRIC Testbed



Benchmark the *communication latency* of the various communication methods and protocols



Measure the *training efficiency* of several synchronous and asynchronous algorithms on *distributed and heterogeneous* machines



Evaluate the framework *scalability* as the client number grows

APPFL × FABRIC

APPFL x FABRIC

Demo Setup

- ➢ *Configure FABRIC Environment*

  Set up and customize the configuration for the FABRIC platform to ensure smooth execution of tasks [Following FABIRC’s Tutorial Jupyter Notebook].

- ➢ *Allocate Resources and Provision Nodes*

  Efficiently allocate computational resources and provision the necessary nodes to support the experiment setup. [One node for server and two more nodes for clients]

- ➢ *Establish SSH Connection to Remote Nodes*

  Securely connect to remote nodes using SSH for seamless interaction and experiment configuration.

APPFL × FABRIC

APPFL x FABRIC

Demo Setup

➢ *Install APPFL and Configure Experiment*

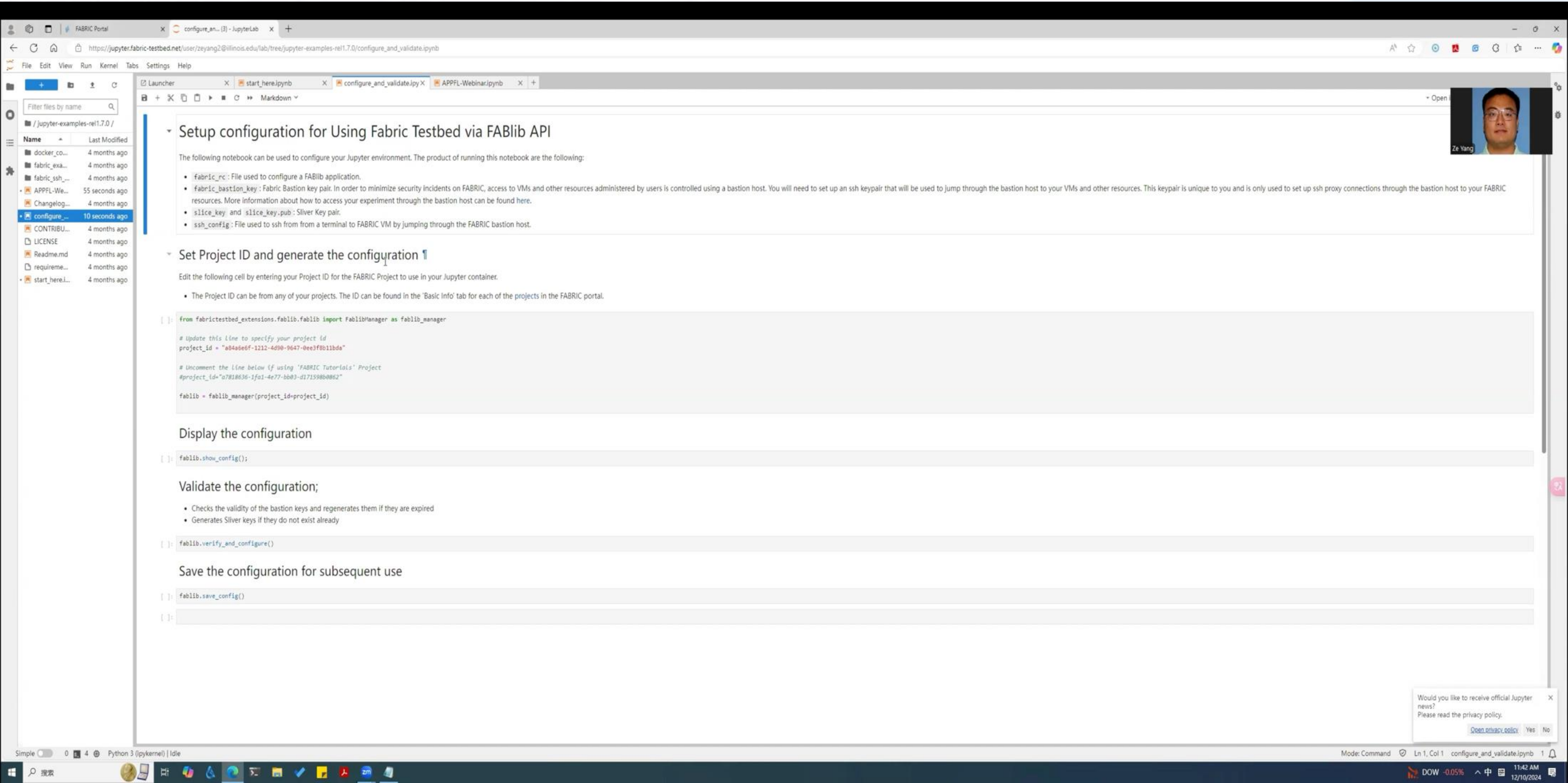
Install the APPFL framework and modify its for the experiments.

➢ *Launch and Monitor Experiments*

Run the experiments and continuously monitor their performance to ensure accuracy and efficiency.

APPFL × FABRIC

APPFL x FABRIC Demo



FABRIC Portal
configure_an... (3) - JupyterLab
https://jupyter.fabric-testbed.net/user/zeyang2@illinois.edu/lab/tree/jupyter-examples-rel1.7.0/configure_and_validate.ipynb
File Edit View Run Kernel Tabs Settings Help
Launcher
start_here.ipynb
configure_and_validate.ipy
APPFL-Webinar.ipynb
Markdown
Filter files by name
/ jupyter-examples-rel1.7.0 /
Ze Yang
Simple
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
configure_and_validate.ipynb

Name	Last Modified
docker_co...	4 months ago
fabric_exa...	4 months ago
fabric_ssh_...	4 months ago
APPFL-We...	55 seconds ago
Changelog...	4 months ago
configure_...	10 seconds ago
CONTRIBU...	4 months ago
LICENSE	4 months ago
Readme.md	4 months ago
requireme...	4 months ago
start_here.i...	4 months ago

Setup configuration for Using Fabric Testbed via FABlib API

The following notebook can be used to configure your Jupyter environment. The product of running this notebook are the following:

- `fabric_rc` : File used to configure a FABlib application.
- `fabric_bastion_key` : Fabric Bastion key pair. In order to minimize security incidents on FABRIC, access to VMs and other resources administered by users is controlled using a bastion host. You will need to set up an ssh keypair that will be used to jump through the bastion host to your VMs and other resources. This keypair is unique to you and is only used to set up ssh proxy connections through the bastion host to your FABRIC resources. More information about how to access your experiment through the bastion host can be found here.
- `slice_key` and `slice_key.pub` : Sliver Key pair.
- `ssh_config` : File used to ssh from from a terminal to FABRIC VM by jumping through the FABRIC bastion host.

Set Project ID and generate the configuration ¶

Edit the following cell by entering your Project ID for the FABRIC Project to use in your Jupyter container.

- The Project ID can be from any of your projects. The ID can be found in the 'Basic Info' tab for each of the projects in the FABRIC portal.

```
[ ]: from fabrictestbed_extensions.fablib.fablib import FablibManager as fablib_manager

     # Update this line to specify your project id
     project_id = "a84a6e6f-1212-4d90-9647-0ee3f8b11bda"

     # Uncomment the line below if using 'FABRIC Tutorials' Project
     #project_id="a7818636-1fa1-4e77-bb03-d171598b0862"

     fablib = fablib_manager(project_id=project_id)
```

Display the configuration

```
[ ]: fablib.show_config();
```

Validate the configuration;

- Checks the validity of the bastion keys and regenerates them if they are expired
- Generates Sliver keys if they do not exist already

```
[ ]: fablib.verify_and_configure()
```

Save the configuration for subsequent use

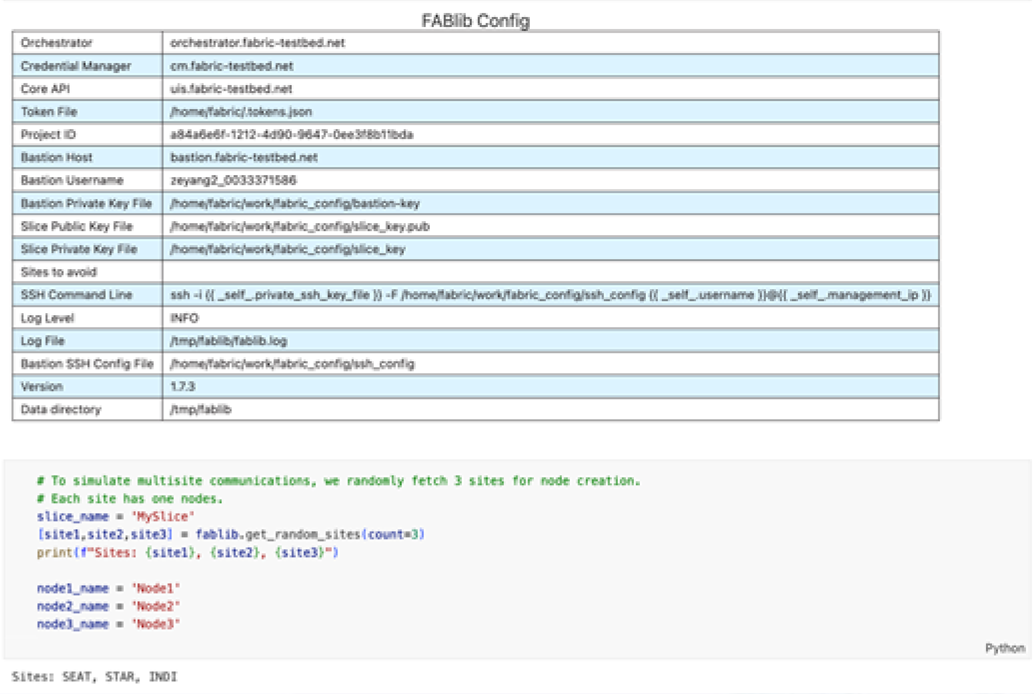
```
[ ]: fablib.save_config()
```

```
[ ]:
```

Would you like to receive official Jupyter news?
Please read the privacy policy.
Open privacy policy Yes No

APPFL x FABRIC

Demo Recipes



FABlib Config

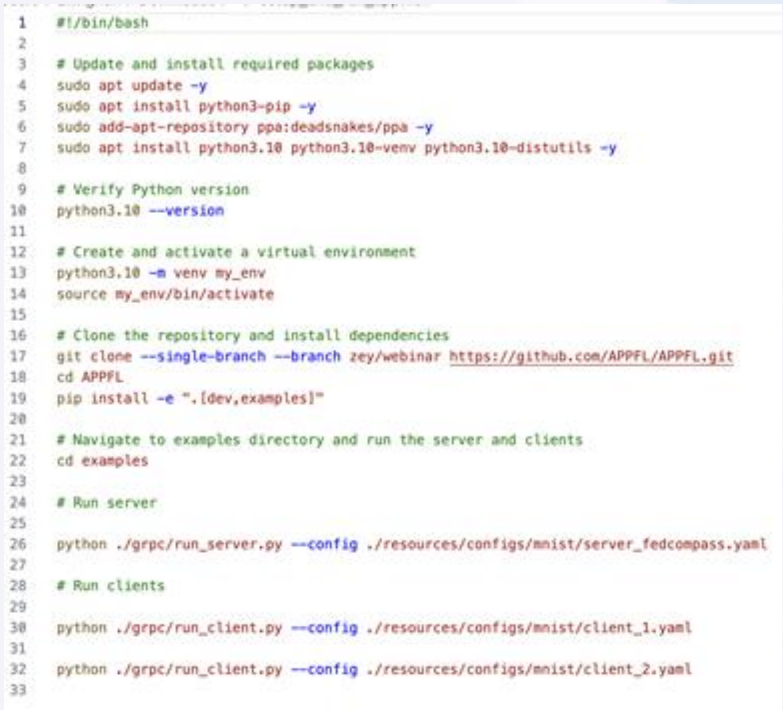
Orchestrator	orchestrator.fabric-testbed.net
Credential Manager	cm.fabric-testbed.net
Core API	uis.fabric-testbed.net
Token File	/home/fabric/.tokens.json
Project ID	a84a6e6f-1212-4d90-9647-0ee3f8b11bda
Bastion Host	bastion.fabric-testbed.net
Bastion Username	zeyang2_0033371586
Bastion Private Key File	/home/fabric/work/fabric_config/bastion-key
Slice Public Key File	/home/fabric/work/fabric_config/slice_key.pub
Slice Private Key File	/home/fabric/work/fabric_config/slice_key
Sites to avoid	
SSH Command Line	ssh -i {{ _self_.private_ssh_key_file }} -F /home/fabric/work/fabric_config/ssh_config {{ _self_.username }}@{{ _self_.management_ip }}
Log Level	INFO
Log File	/tmp/fablib/fablib.log
Bastion SSH Config File	/home/fabric/work/fabric_config/ssh_config
Version	1.7.3
Data directory	/tmp/fablib

```python
# To simulate multisite communications, we randomly fetch 3 sites for node creation.
# Each site has one nodes.
slice_name = 'MySlice'
[site1,site2,site3] = fablib.get_random_sites(count=3)
print(f"Sites: {site1}, {site2}, {site3}")

node1_name = 'Node1'
node2_name = 'Node2'
node3_name = 'Node3'
```

Python

```
Sites: SEAT, STAR, INDI
```



```bash
#!/bin/bash

# Update and install required packages
sudo apt update -y
sudo apt install python3-pip -y
sudo add-apt-repository ppa:deadsnakes/ppa -y
sudo apt install python3.10 python3.10-venv python3.10-distutils -y

# Verify Python version
python3.10 --version

# Create and activate a virtual environment
python3.10 -m venv my_env
source my_env/bin/activate

# Clone the repository and install dependencies
git clone --single-branch --branch zey/webinar https://github.com/APPFL/APPFL.git
cd APPFL
pip install -e ".[dev,examples]"

# Navigate to examples directory and run the server and clients
cd examples

# Run server

python ./grpc/run_server.py --config ./resources/configs/mnist/server_fedcompass.yaml

# Run clients

python ./grpc/run_client.py --config ./resources/configs/mnist/client_1.yaml

python ./grpc/run_client.py --config ./resources/configs/mnist/client_2.yaml

```

Links:

1. FABRIC Setup Notebook
2. FL Experiment Setup and Launching Scripts

APPFL × FABRIC

APPFL x FABRIC

Future Work

➢ *Chameleon Integration*

Integrate FABRIC with Chameleon (https://www.chameleoncloud.org) to get access to powerful machines to run more computational intensive experiments.

➢ *Running FL with FABRIC on Climate Data*

Leverage FABRIC's configurable network and seamless node connectivity to optimize federated learning for large-scale climate data, avoiding transferring large amount of data and improving model performance. [APPFL+FABRIC User Story]

Thank you!

FABRIC

FAB

FABRIC Across Borders

Q&A

FABRIC

FAB

FABRIC Across Borders

Acknowledgements

In addition to our presenter, we would like to acknowledge the behind-the-scenes team that diligently worked to bring this webinar to production:

- **KC Wang**, content
- **Chelsea Davis,** project manager
- **Jayasree Jaganatha,** social media specialist

Resources

Call to Action

FABRIC Matrix:
https://bit.ly/FABRICmatrix

Connect With Us

Newsletter Signup: bit.ly/FABRICnewsletter

Office Hour Sign Up: bit.ly/FABRIC-Office-Hours

Other Resources

Website: bit.ly/m/FABRICtestbed

YouTube: youtube.com/@fabrictestbed

FABRIC Account: portal.fabric-testbed.net

Ambassador Program: bit.ly/FABRIC-Ambassador-Program

FABRIC LinkedIn: linkedin.com/company/fabrictestbed

Citing FABRIC: bit.ly/citing-fabric

Thank You for Attending!

Join us for our upcoming webinars:

- **Date** - Stitching Together Innovation with FABRIC Users
- **Date** - Mastering FABRIC: Tips and Tricks Webinar

Visit our YouTube Channel: youtube.com/@fabrictestbed