Hewlett Packard
Enterprise

SWARMLEARNING

Month DD, 20YY

AGENDA

INTRODUCTION

DIFFERENT APPROACH IN MACHINE LEARNING

MACHINE LEARNING LANDSCAPE

PROBLEM WITH CENTRALIZED AND TRADITIONAL TECHNIQUE

HPE SWARM LEARNING

HOW DOES IT WORK

AGENDA

PROGRAMMING VIEW

ARCHITECTURE-COMPONENT VIEW

DEPLOYMENT VIEW

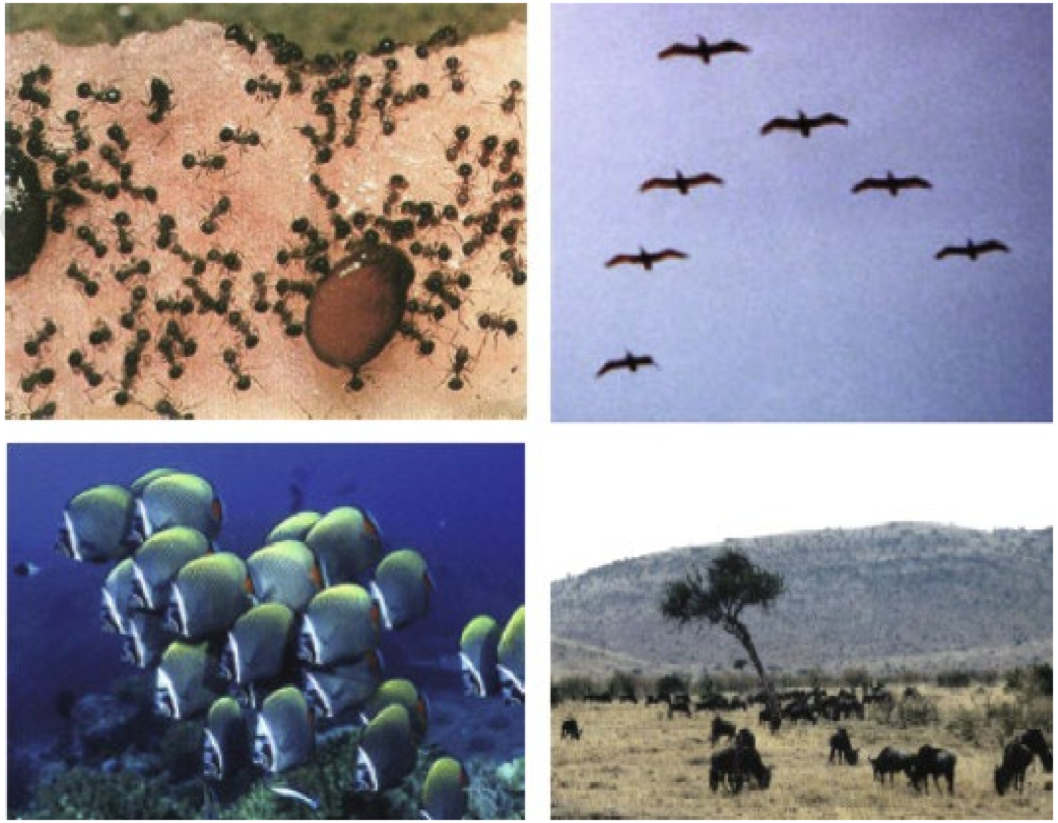
TROUBLESHOOTING

LEARNING CHECK

INTRODUCTION

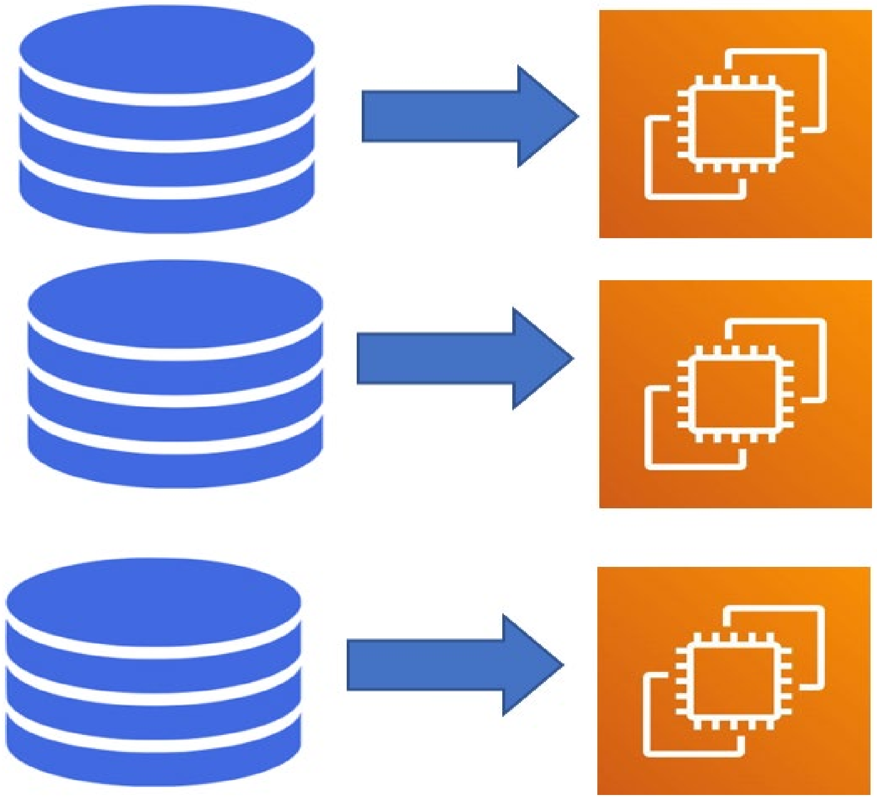
INTRODUCTION – UNDERSTANDING THE NEED

- Swarm intelligence is a subfield of Artificial Intelligence.
- The inspiration often comes from nature, especially biological systems
- It is the collective behavior of decentralized, self-organized systems.
- The trend of edge computing that is moving computing and intelligence closer to the data.
- The data privacy and security are important for an organization .
- Swarm Learning tries to overcome the limitation.



DIFFERENT APPROACH IN MACHINE LEARNING

DISCONNECTED LEARNING

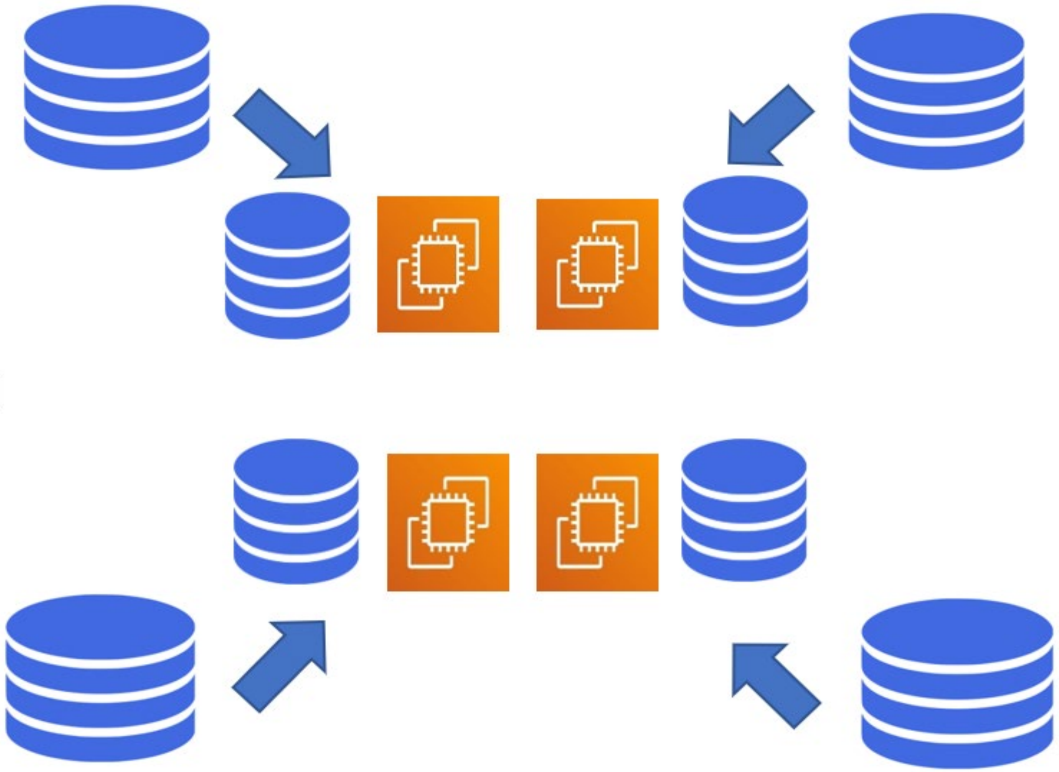


- Set of individual machines that are independent.

- Each machine is having their own set of data and are also trained independently.
- Same data can be trained using different machine learning technique at a same time.

- Drawback
  - Data Availability.
  - Cost in Data Migration to each local space.
  - Not Adequate Data
  - Can only be used for simple workloads.

CENTRALIZEDEARNING

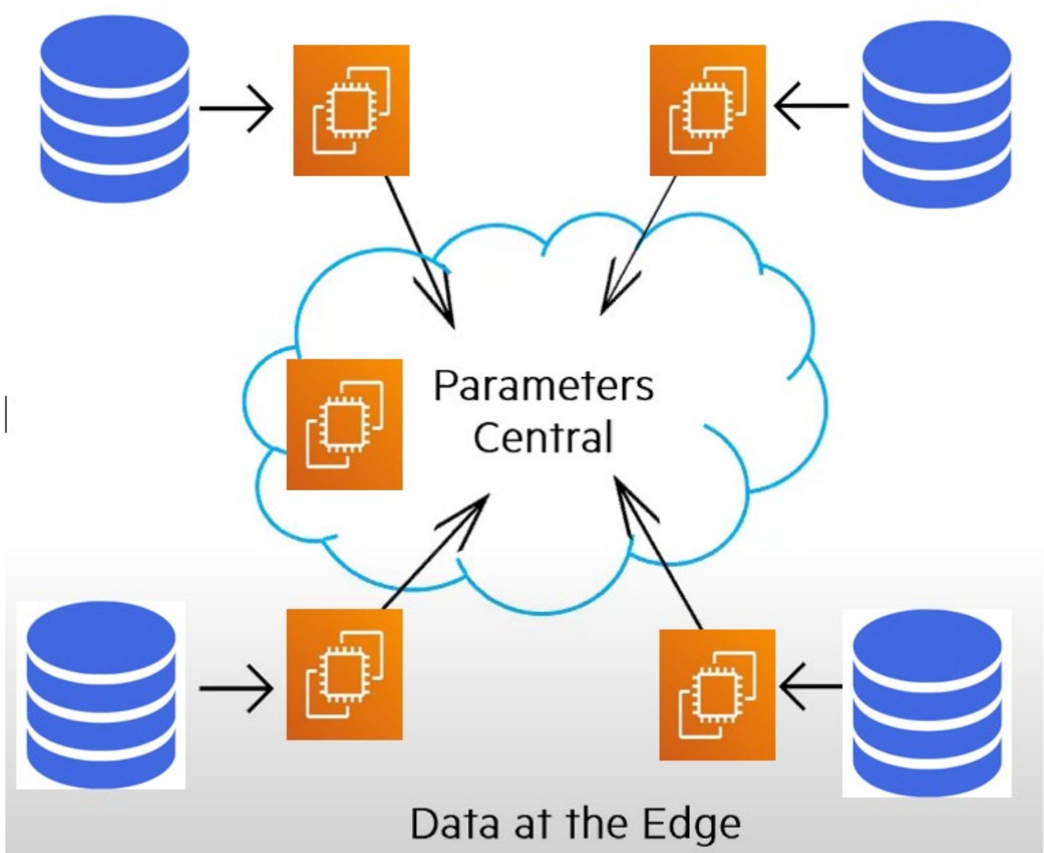


- There is a central repository where the data is available.

- A central server is used to orchestrate the different steps of the algorithms and coordinate all the participating nodes during the learning process.

- Drawback
  - Bottleneck.
  - Data Security
  - Data sovereignty
  - Data Privacy

FEDERATEDLEARNING



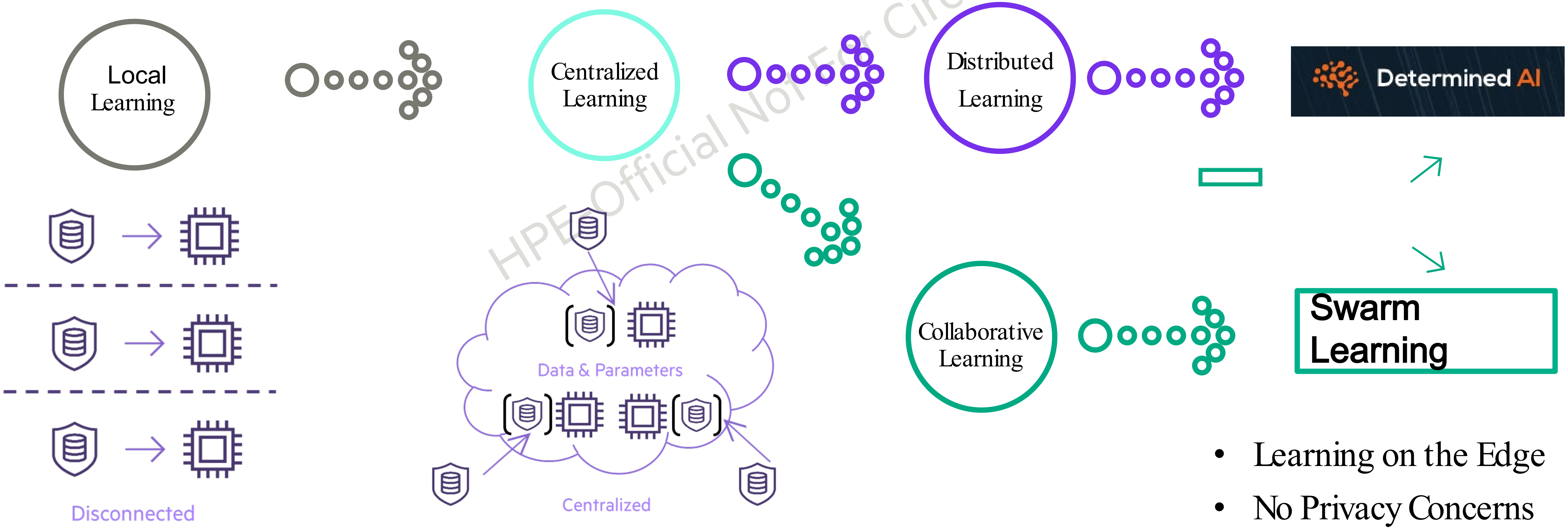
- Distributed Machine Learning Framework.
- Models are trained at the edges and parameters are learned.
- Parameters are merged by a central coordinator.

- Benefits
    - Lot of cost saving of transferring data to a central storage and Storage cost etc.
    - Ensures Data Privacy.

MACHINE LEARNING LANDSCAPE

- Performance ↓
- Learning in data center
- Scale ↓

- Learning in data center



Local Learning

Disconnected

Centralized Learning

Data & Parameters

Centralized

Distributed Learning

Collaborative Learning

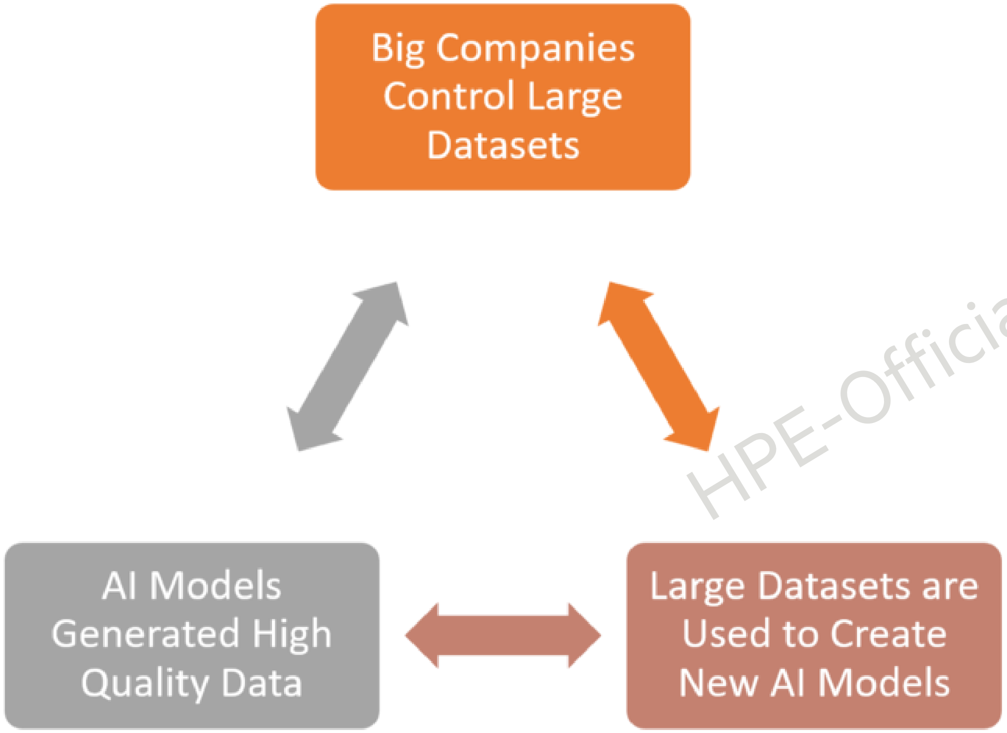
Determined AI

Swarm Learning

- Learning on the Edge
- No Privacy Concerns
- Scale ↑

PROBLEM WITH CENTRALIZED AND TRADITIONAL TECHNIQUE

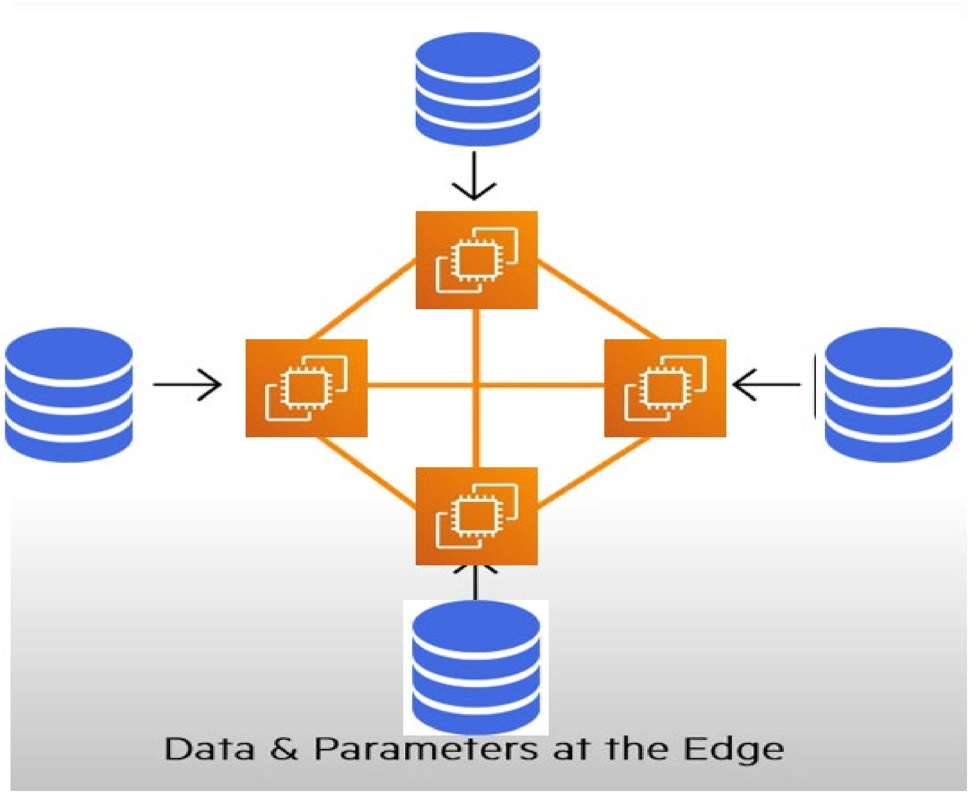
PROBLEMWITHCENTRALIZEDEARNING


Big Companies Control Large Datasets
Large Datasets are Used to Create New AI Models
AI Models Generated High Quality Data

- Model Centralization Problem
- Training Centralization Problem
- Model Optimization problem
- Lack of Standard Monetization

INTRODUCTION TO SWARM LEARNING

SWARMLEARNING



Swarm Learning: It is a framework designed for creating a set of nodes in which each node possessing some training data locally to train a common Machine Learning model without sharing data. Instead, the parameter learned by each node is shared and merged to obtain a global model

There is not a static central merger, but it may be any node in the swarm network that got elected for parameter merging.

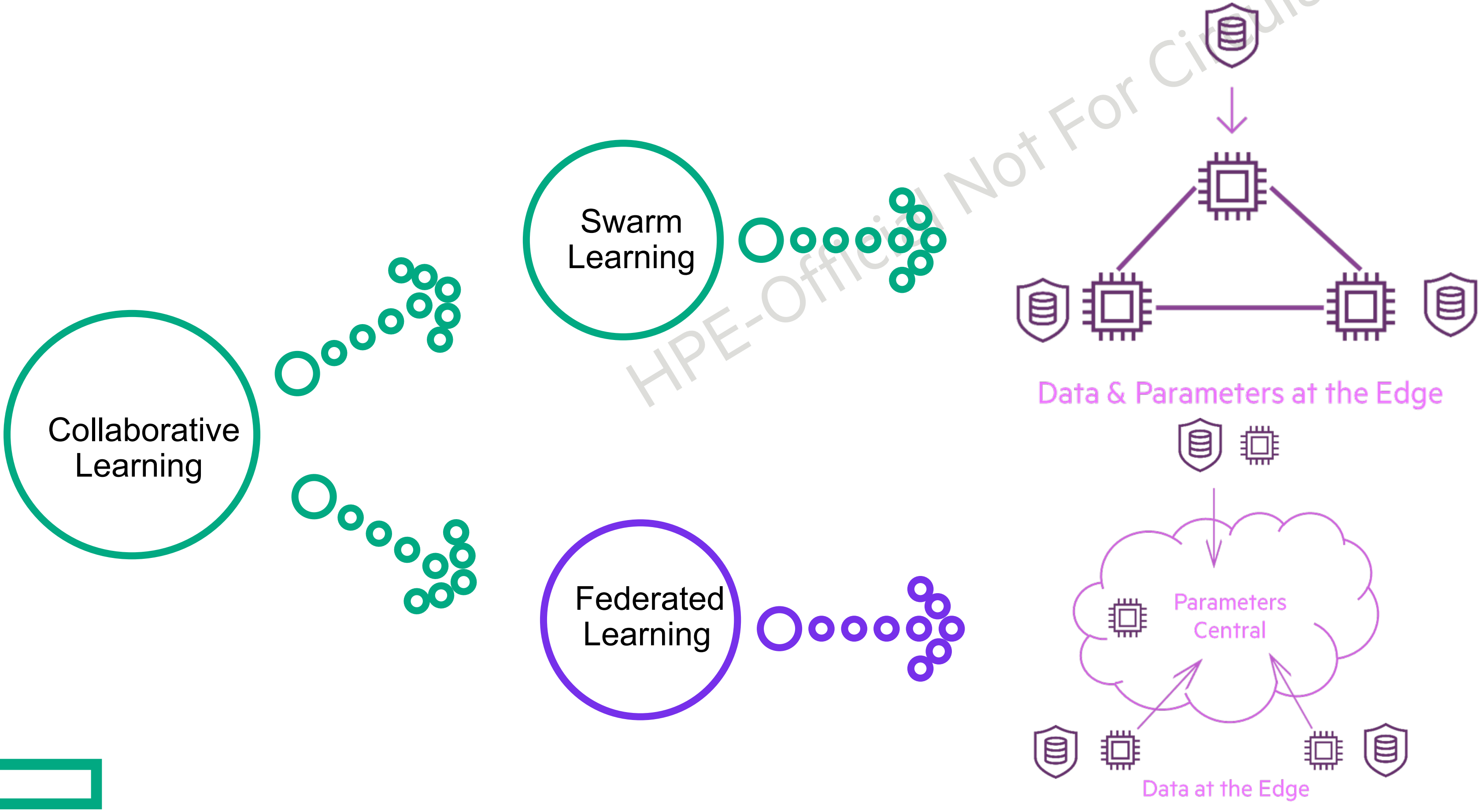
Idea is to achieve super intelligence combining brains.

In short Swarm Intelligence is a “Brain of Brains”.

COLLABORATIVE MACHINE LEARNING APPROACHES …

Swarm Learning can do what Federated Learning (our competition!) does and more …

It enables privacy-preserving, collaborative machine learning by treating all participants equally



NEED FOR DECENTRALIZED ML

- Rise of Edge AI
- Need for privacy preserving computation
- Adoption of swarm intelligence in IoT
- Expanding AI use cases and shift to more responsible AI

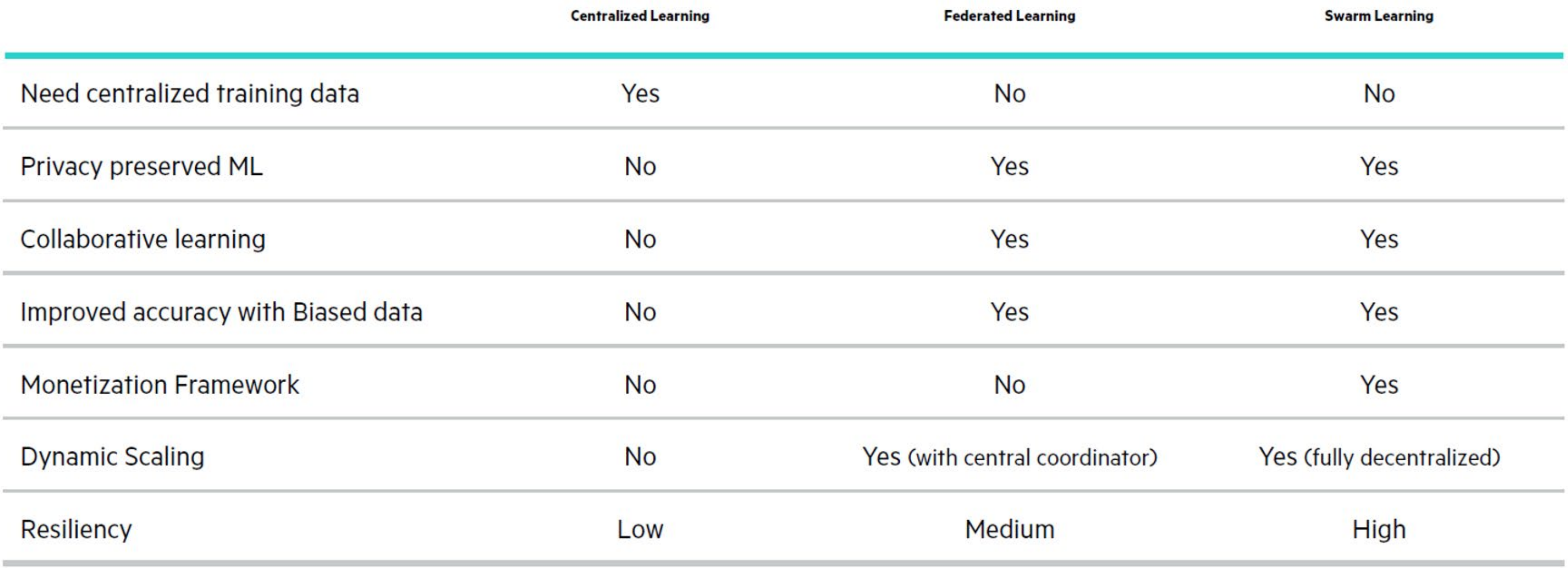
CHALLENGES IN ML

Low efficiency	Multiple sites send **raw data** over the network; need high bandwidth
Lack of Data Privacy	Privacy acts like GDPR prevent moving data to a central datacenter/cloud
Lack of Collaboration	Data generated in silos (e.g. data centers, sensors, vehicles)
Biased Data	Data biases due to demographic distribution
Lack of Monetization Framework	Data is new currency – owners look for ways to monetize the data

SWARM LEARNING – VALUE PROPOSITION

Improved efficiency	Saving in network bandwidth; Near real-time learning & inferencing
Privacy Preserved ML	Data does not leave the origin
Collaborative Learning	Decentralized network of swarm nodes based on blockchain; Dynamic consortium membership
Learning with Biased Data	Localized learning with global state merge
Monetization Framework	Enable data owners to monetize the insights – Framework to enable incentives for sharing the model parameters

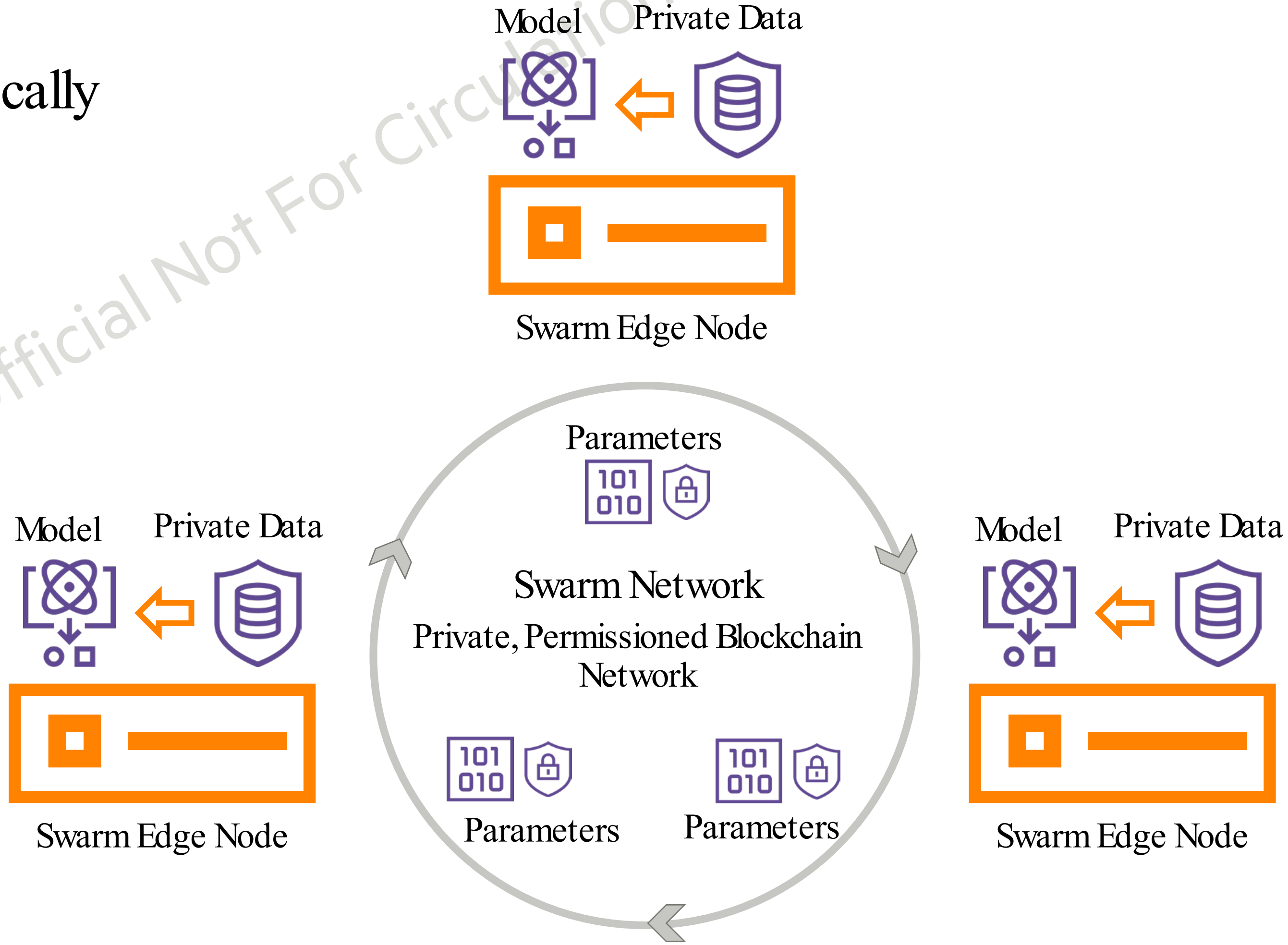
COMPARISON OF SWARM LEARNING WITH FEDERATED AND CENTRALIZED



	Centralized Learning	Federated Learning	Swarm Learning
Need centralized training data	Yes	No	No
Privacy preserved ML	No	Yes	Yes
Collaborative learning	No	Yes	Yes
Improved accuracy with Biased data	No	Yes	Yes
Monetization Framework	No	No	Yes
Dynamic Scaling	No	Yes (with central coordinator)	Yes (fully decentralized)
Resiliency	Low	Medium	High

WHAT ARE THE GUIDING PRINCIPLES OF SWARM LEARNING

- Equal and like-minded partners in the network
- Ownership of the data remains local
- Data protection and data security solved locally
- Less susceptible to biased datasets
- Easy to develop, deploy and consume



BLOCKCHAIN(WITH SN AND CONTROL LAYER)

A blockchain is a growing list of records, called blocks, that are linked together using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data

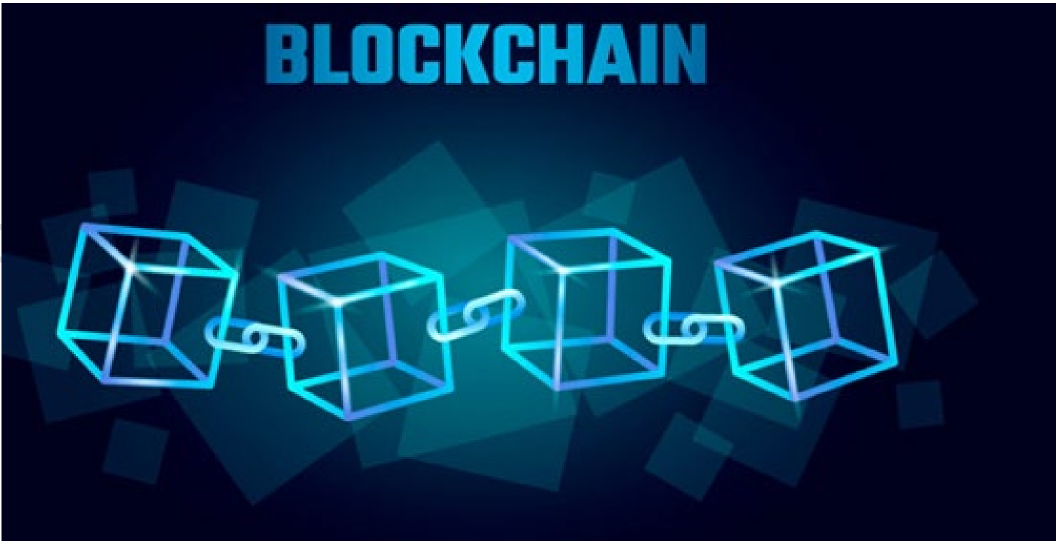
It is difficult or impossible to change or hack the system

- ➢ It brings Security into the network
- ➢ Monetize for parameters sharing using crypto tokens(in future)

BLOCKCHAIN

- One key difference between a typical database and a blockchain is how the data is structured.

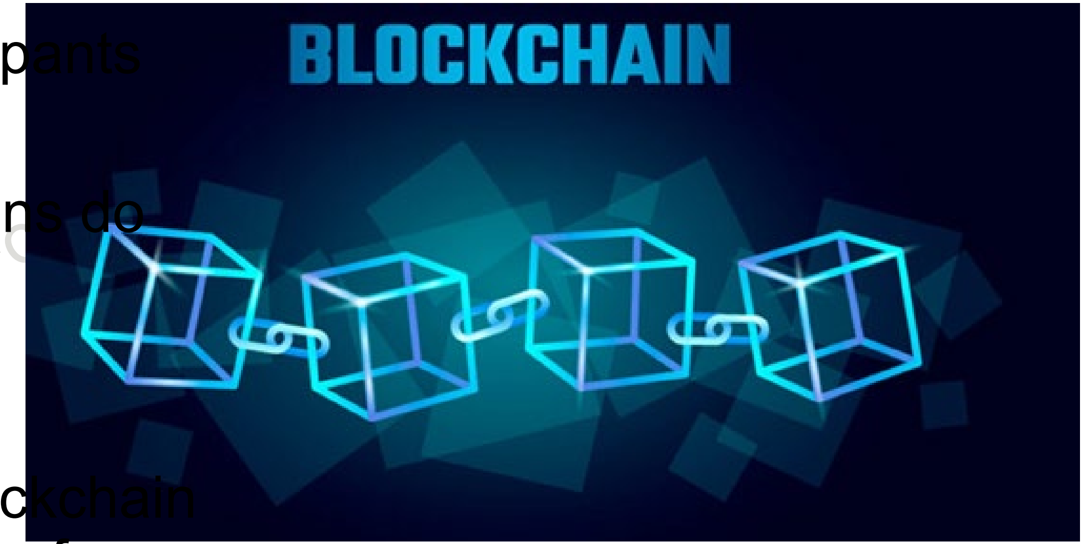
- A blockchain collects information together in groups, known as blocks, that hold sets of information. Blocks have certain storage capacitie and, when filled, are closed and linked to the previously filled block, forming a chain of data known as the blockchain.



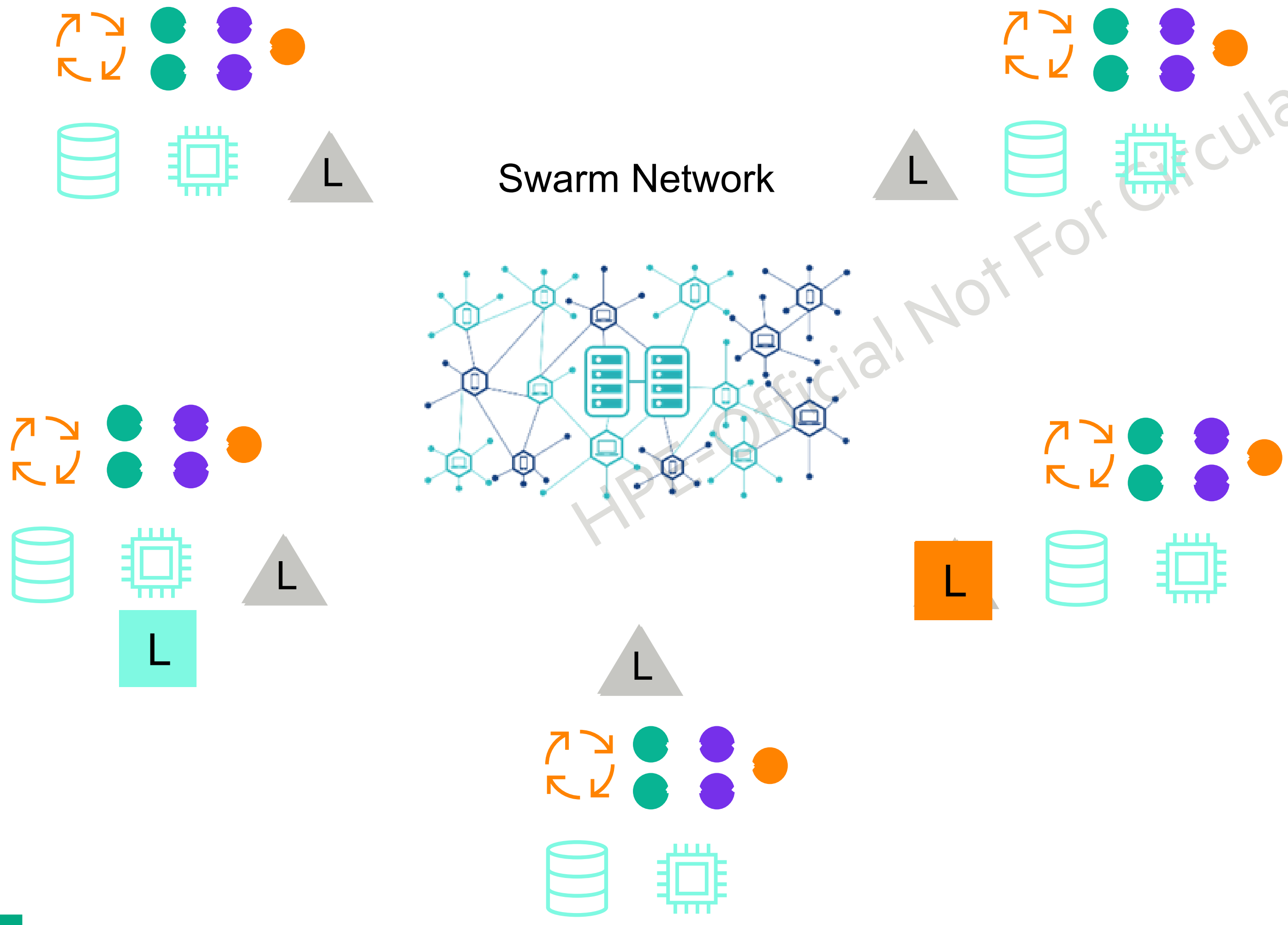
- All new information that follows that freshly added block is compiled into a newly formed block that will then also be added to the chain once filled.

BLOCKCHAIN

- Public blockchains use compute-intensive consensus algorithms, like Proof of Work, to ensure the transactional guarantee across untrusted participants
- Private, permissioned blockchain implementations do not need such consensus algorithms as the participants are validated through identity.
- Swarm Learning uses private, permissioned blockchain that does not require such a consensus and therefore does not suffer from compute intensive consensus algorithms.



HOW DOES IT WORK



0. On-boarding (Done Offline)

1. **Register**
   Nodes register to Swarm Network and receive ML model

2. **Train**
   Nodes train the model on local data for a time-window (batch-size)

3. **Merge**
   Nodes share and merge the trained models

4. **Repeat**
   Repeat 2 & 3 till desired accuracy is achieved

REGISTER

- Nodes register to Swarm Network. The process begins with enrollment, or registration, in the Swarm smart contract by each node. This is a one-time process.

- Each node subsequently records its relevant attributes in the contract such as the uniform resource identifier (URI) from which its own set of trained parameters can be downloaded by other nodes.

TRAIN

- Nodes train a model on local data for certain epochs. In this node proceed to train the local copy of the model iteratively over multiple epochs.

- During each epoch, every node trains its local model using one or more data batches after which it exports the parameter values in a file and uploads it to a shared file system for other nodes to access.

- Node signals other nodes that it is ready for the parameter-sharing step.

PARAMETER SHARING

- Once the number of nodes that are ready for parameter sharing step.

- It begins with the process of electing the epoch leader, whose role is to merge the parameters derived after local training on all nodes. It's a star topology, where a single leader performs the merge.

PARAMETER MERGE

- Nodes share and merge the trained models at elected Node.
- Currently The node which completes the local training first will be a Leader
- The framework supports multiple merge algorithms such as mean, weighted mean, m and so on.

PROGRAMMING VIEW

Integrate using Swarm callback API (keras)

```
# 1. IMPORT SWARM CALLBACK #
from swarmlearning.tf import SwarmCallback
…
# 2. DEFINE SWARM CALLBACK #
swarmCallback = SwarmCallback(syncFrequency=100, min_peers=2, useAdaptiveSync=true, adsValData=(x_test,
y_test), adsValBatch_size=VALID_BATCH_SIZE)
…
# 3. ADD SWARM CALLBACK IN THE CALLBACK LIST FOR TRAINING #
callbacks_list = [swarmCallback,…]
```

PROGRAMMING VIEW

```
 (x_train, y_train),(x_test, y_test) = load_data(dataDir)    # Pre process and load Data
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([                          # define the keras model and add layers ( a simple NN )
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(512, activation=tf.nn.relu),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation=tf.nn.softmax)
])

model.compile (optimizer='adam',   loss='sparse_categorical_crossentropy',  metrics=['accuracy']) # Define metric

# Create Swarm callback
swarmCallback = SwarmCallback ( syncFrequency=100,  minPeers=2,   adsValData=(x_test, y_test),  adsValBatchSize=32 )

model.fit (x_train, y_train,                          # Model Training. Assume Dataset X train has 100000  ( 0.1 million ) elements
       batch_size = 100,                              #  100000 / 100  = 1000 iterations for each epoch
       epochs=10,                                     # 1000 iterations *10 epochs= 10000 iterations (local model update) for this training run
       verbose=1,
       callbacks=[swarmCallback])

# Save model and weights
model_path = os.path.join(modelDir, model_name)
model.save (model_path)
print('Saved the trained model!')
```

No of batches of local training, between 2 swarm syncs

Overall : 100 Swarm "global" trainings (G), 9900 local training (L) in this example training…

. ...

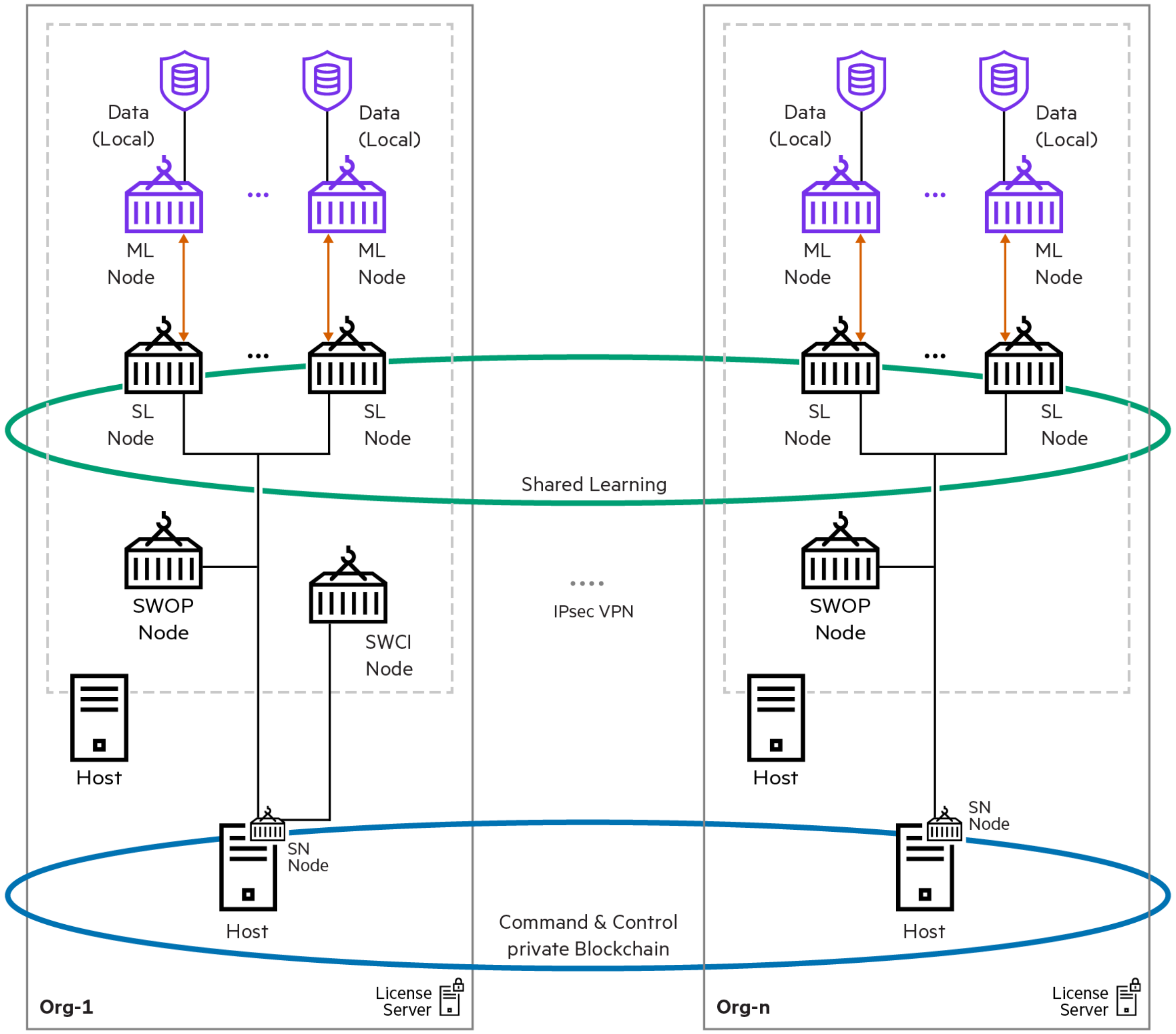
Dataset:<1,2..100>, L, <101,102..200 >,L,..<9901,9902,.. 10000>, G, <10001,10002..10100>,L ..20000>, G… and it keeps repeating..

Example Timeline (for syncFrequency = 3) ; L -> Local Training model update; G → Swarm Global Merge & model update

Start------L------L-----L----- **G**------L------L-----L-----**G**----L-----L-----L---**G**------L-----L-----L-----**G** →Training is complete

ARCHITECTURE – COMPONENT VIEW

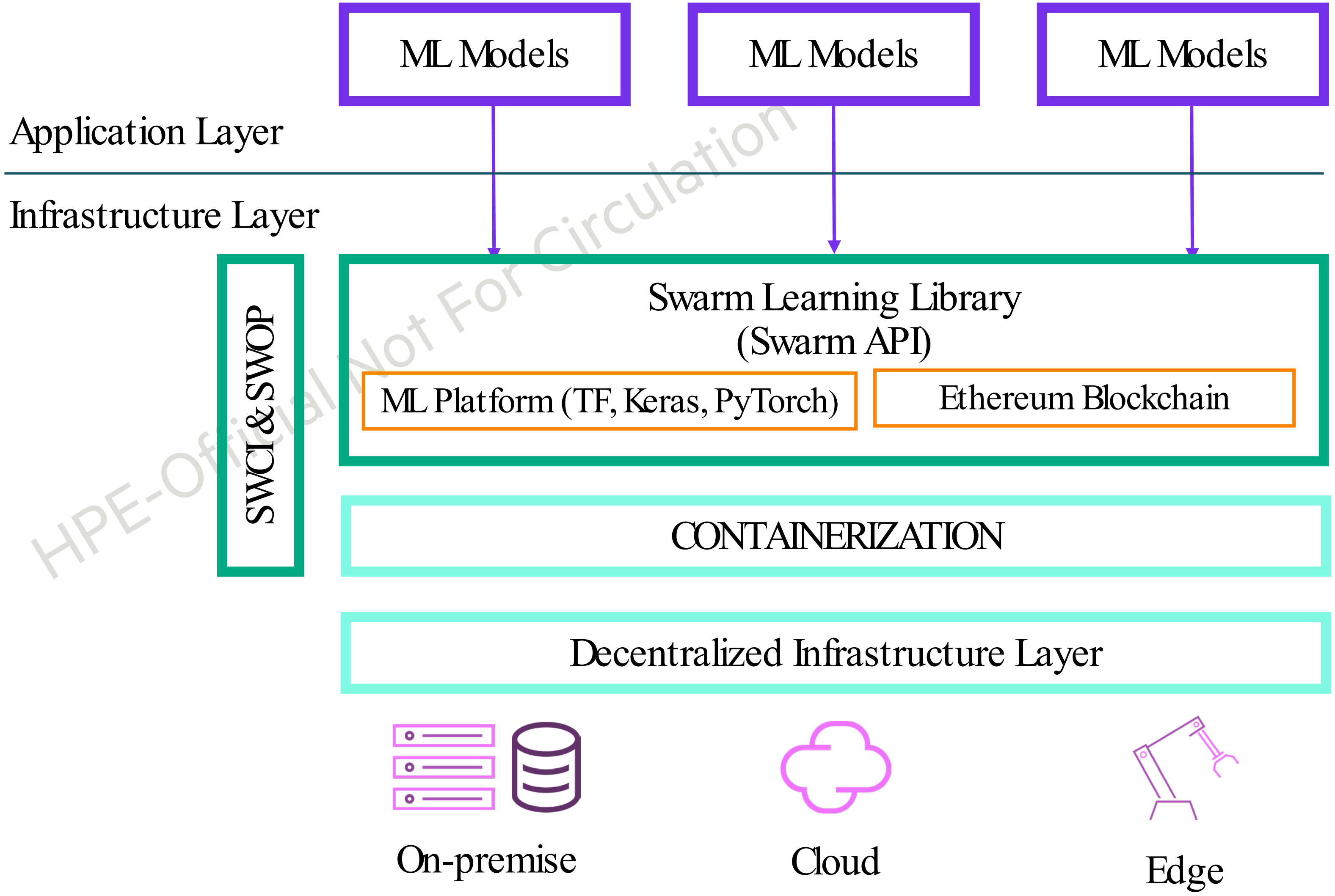
- Swarm Learning Software is provided as Containers with APIs for easy integration
- ML Node/container – Swarmf'ied User ML program that ONLY talks to its "twin" SL node. Uses the Swarm API Client Library
- 4 components / containers
  - Swarm Learning (SL) – Interfacing user-defined Machine Learning programs, performs core "global" merge.
  - Swarm Network (SN) - the blockchain network - "global state" of the all operations
  - SWCI – command interface (console)
  - SWOP – Command and Control Agent (pull image/ build image / make user content/ run image)
- License server – Autopass license server
- Security and digital identity aspects are handled by X509 certificates.
- Enable starting small Swarm Networks and then growing big by combining them



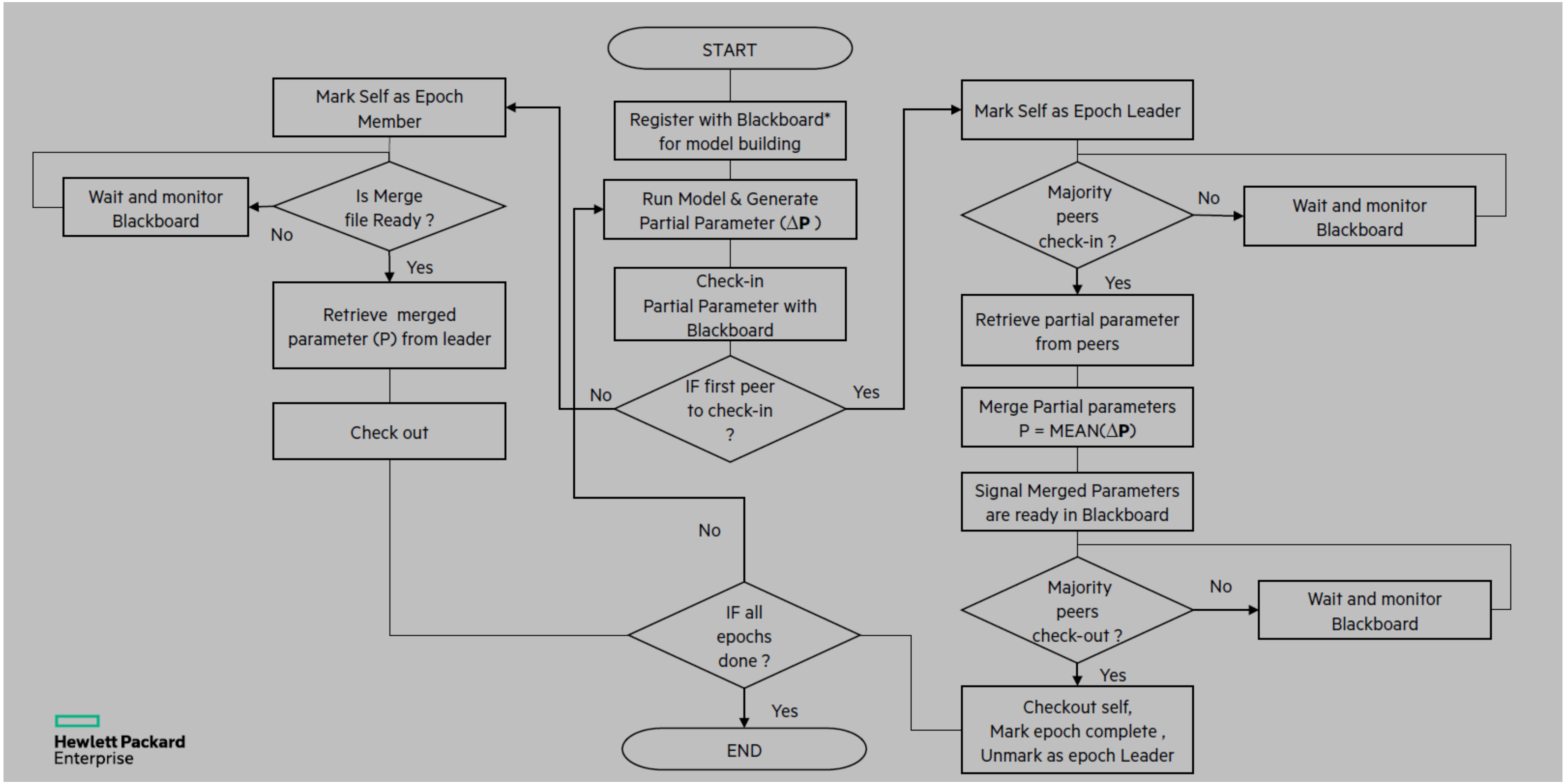
*All nodes are containers

SOFTWARE STACK VIEW

- Swarm Learning Library provided as containers
  - Simplify deployment
  - Simplify Scaling
- Simple callback API for integration with ML models
  - Simplify Programmability
  - Reduces Re-Skilling
- Management commands to control the network
  - Packaging
  - Deployment
  - Command and Control



SWARM LEARNING FLOW CHART


START
Register with Blackboard* for model building
Run Model & Generate Partial Parameter (ΔP)
Check-in Partial Parameter with Blackboard
IF first peer to check-in ?
No
Yes
Mark Self as Epoch Member
Is Merge file Ready ?
No
Yes
Wait and monitor Blackboard
Retrieve merged parameter (P) from leader
Check out
Mark Self as Epoch Leader
Majority peers check-in ?
No
Yes
Wait and monitor Blackboard
Retrieve partial parameter from peers
Merge Partial parameters P = MEAN(ΔP)
Signal Merged Parameters are ready in Blackboard
Majority peers check-out ?
No
Yes
Wait and monitor Blackboard
Checkout self, Mark epoch complete , Unmark as epoch Leader
IF all epochs done ?
No
Yes
END
Hewlett Packard Enterprise

DEPLOYMENT VIEW - CONCEPTS

SWOP

- De-centralized task management framework
- Agent to manage Swarm Learning operations (ML OPS)
- Executes "tasks" assigned to it.
- Resource profile – profile schema YAML

Task – A "reusable" unit of work.

- Defined by user.
- Has Pre-requisite / Body/ Outcome.
- Chained together to create linear workflows.
- Schema specified in the **TASK SCHEMA YAML**
- 4 types:

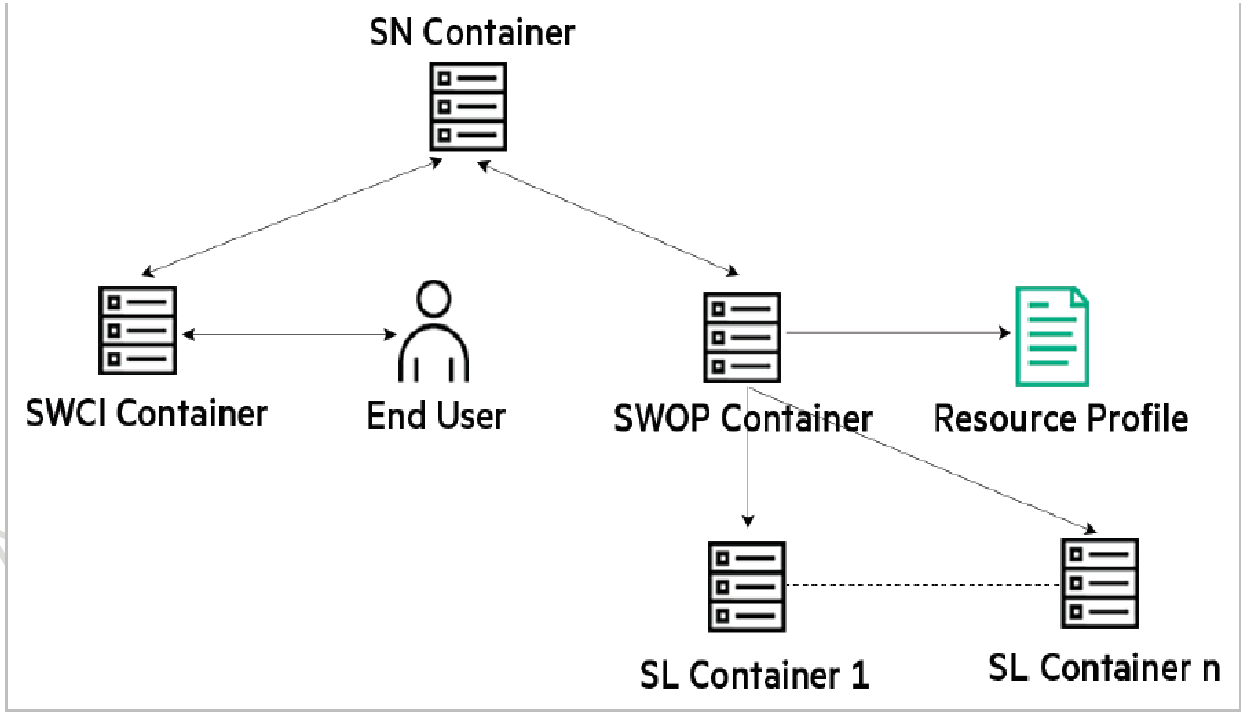
Pull_Image / Make_Swarm_User_Content / Make_User_Container / Run_Swarm

Taskrunner

- Entity that "executes" an assigned task
- Used to coordinate execution of tasks by SWOP nodes
- An instance of a smart contract

Swarm Learning training contracts:

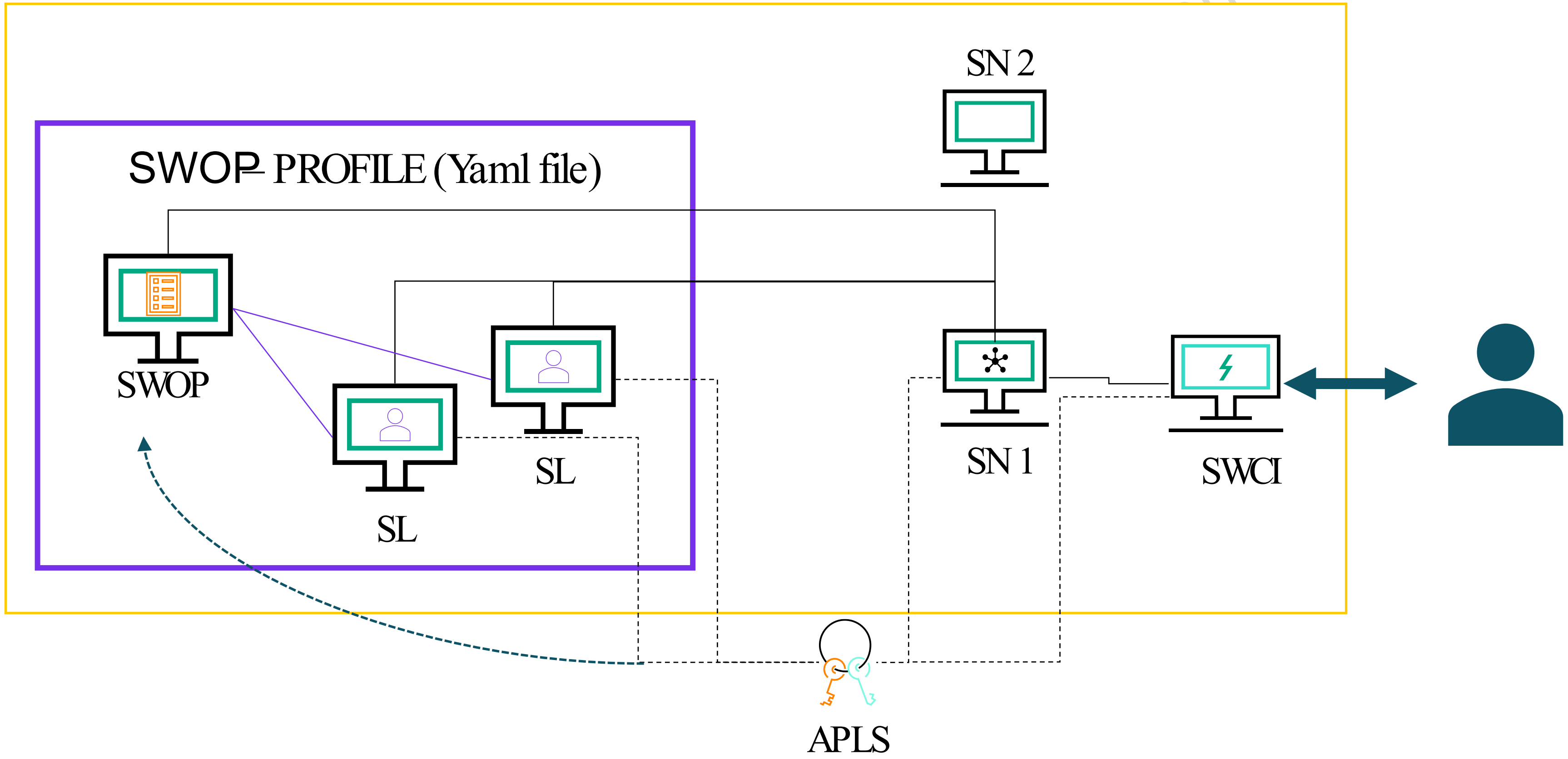
- Used to control the swarm learning training process.
- An instance of an Ethereum smart contract.
- Registered into SN using CREATE CONTRACT command.



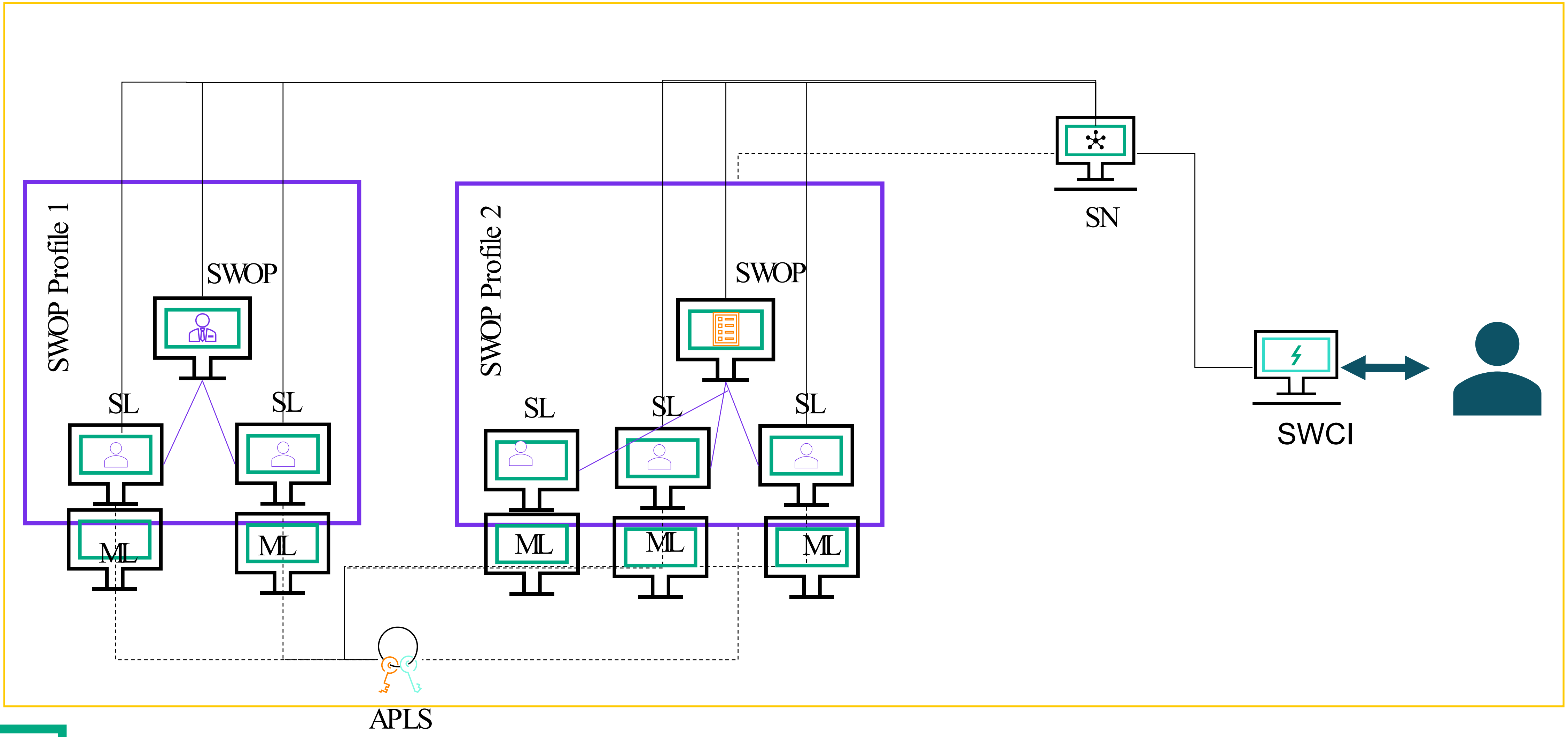
SWOP Resource profile (YAML) Typical elements:

- Name of the group (string)
- The No of SL nodes in the group (that would be spawned)
- Taskrunner contract of the group
- APLS, SN that this SWOP has to talk to
- Environmental variables (optional)

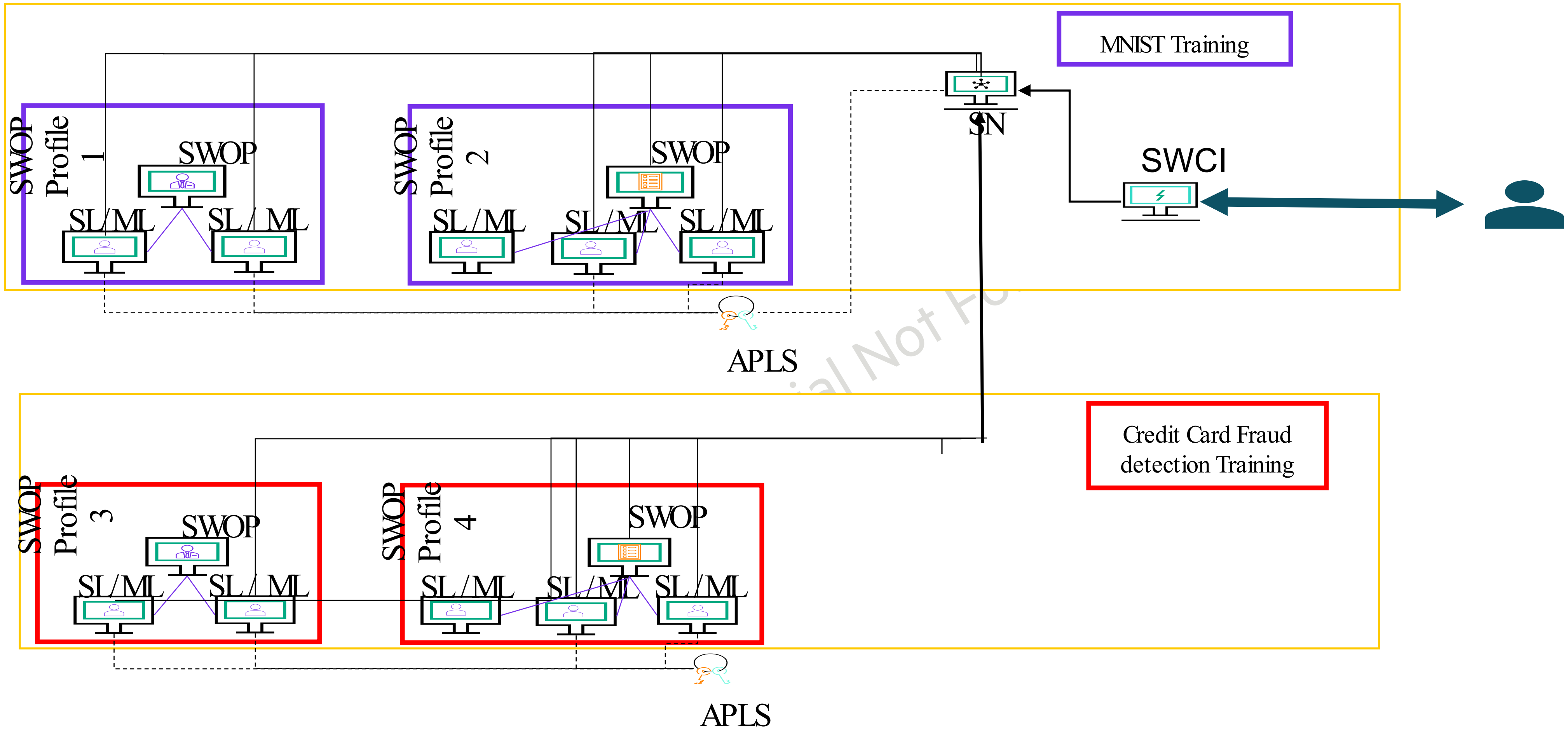
TASKRUNNER ARCHITECTURE (COMPONENTS)



SWOP DEPLOYMENT - 2 SWOPS LISTENING ON 1 TASK RUNNER (ILLUSTRATION)



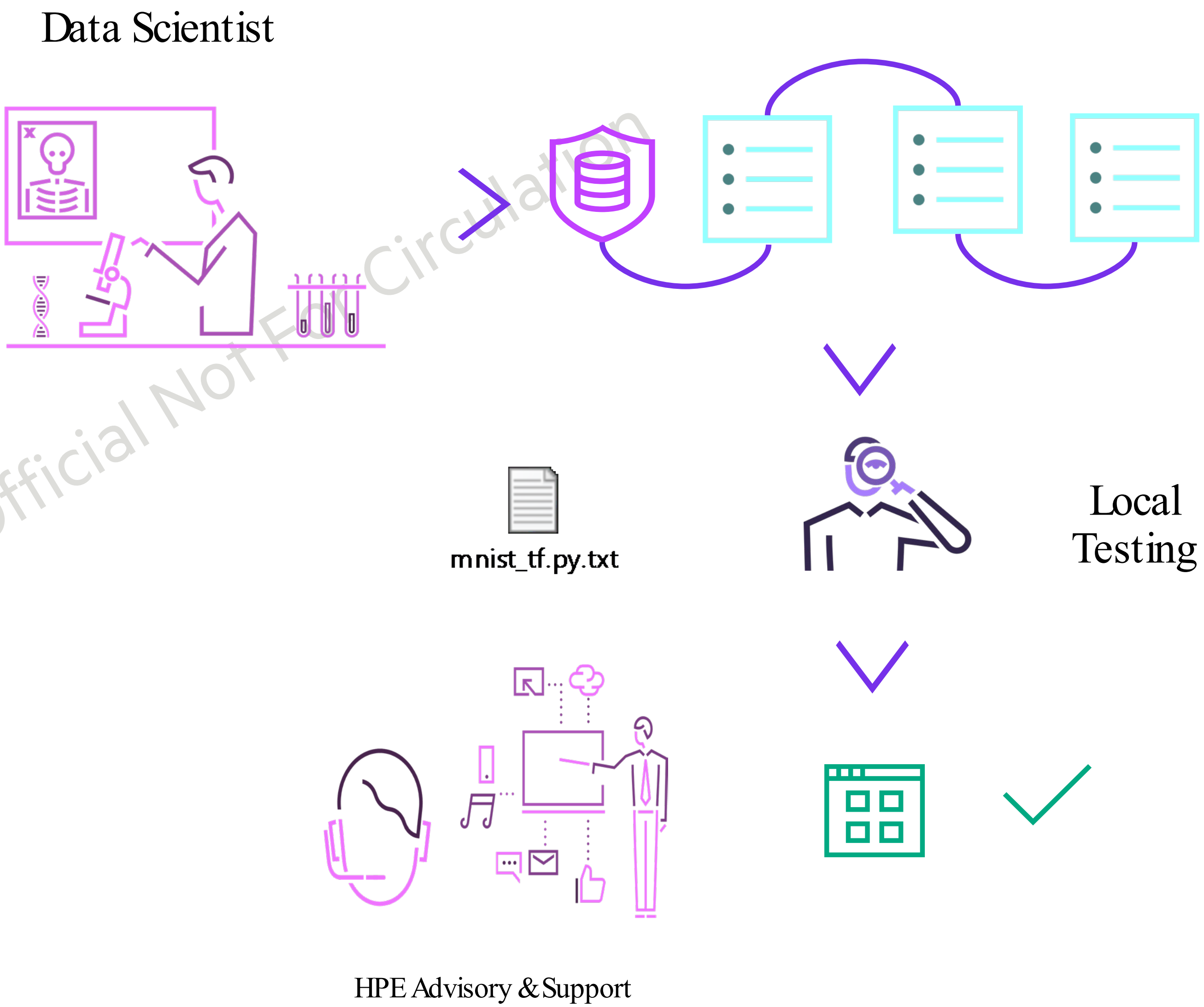
TASKRUNNER DEPLOYMENT (CONCURRENT MODEL TRAINING WITHIN THE SAME SN)



- Multiple Task runner contracts are created , one for MNIST and one for Credit card Fraud detection
- SWOP profile defines which Taskrunner its listening to

DEVELOP MODEL LOCALLY

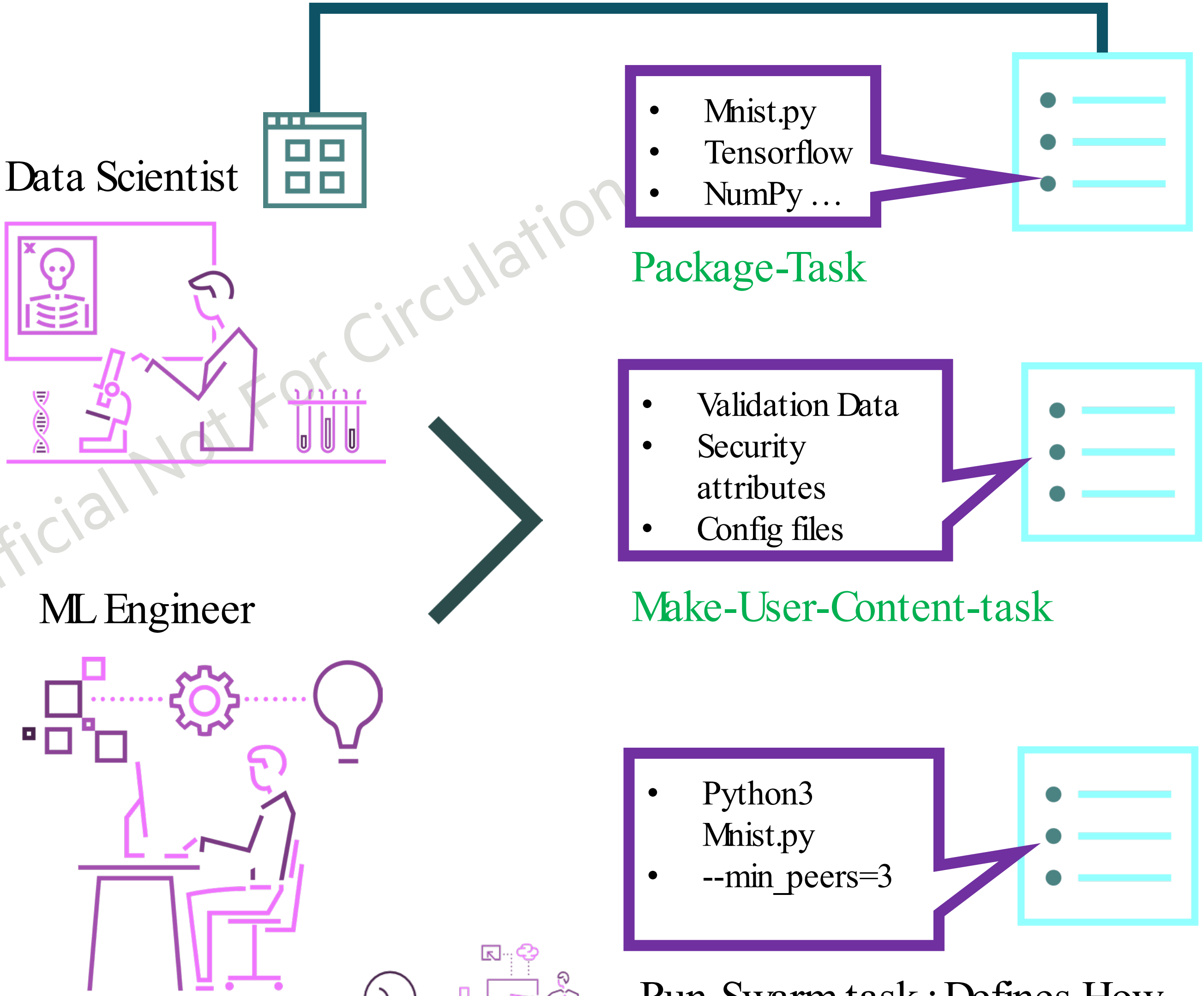
- Data scientist Builds the ML pipeline
- Locally test them
- Make the programs swarm enabled by integrating with swarm client library



PACKAGE , DEPLOY AND SETUP SWARM LEARNING

- ML Engineer with the help of the Data Scientist
  - Identifies dependencies
  - Breaks the application deployment down into Tasks
  - Registers them with the Swarm Learning Network using SWCI Tool

- For example : Running a training session on MNIST data set.
  - Task1 : Package-task : Setup necessary s/w environment on edge nodes to execute training session
  - Task2 : Make-User-Content-Task : Setup Common Validation data set, config files , etc...
  - Task3 : Run-Task : Execute the Training session on Edge Nodes



Data Scientist

ML Engineer

- Mnist.py
- Tensorflow
- NumPy …

Package-Task

- Validation Data
- Security attributes
- Config files

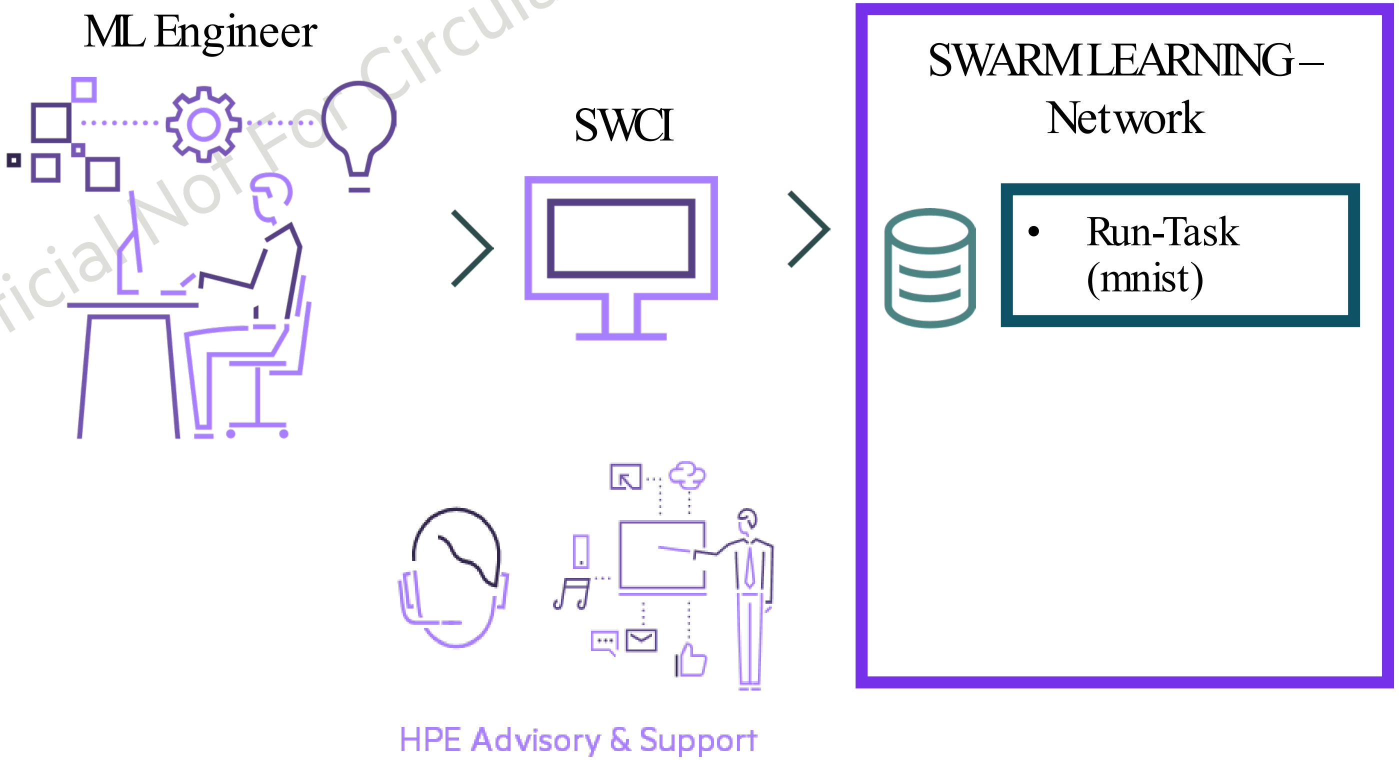
Make-User-Content-task

- Python3 Mnist.py
- --min_peers=3

Run-Swarm task : Defines How to execute the software

DECENTRALIZED TRAINING – USING SWOP

- ML Engineer now can train/re-train the model as needed
- The trained model can be passed on the next set of operations.



TROUBLESHOOTING

TROUBLESHOOTING – TYPICAL ISSUES

Could fall into any one of these:

- x.509 certificates are not configured correctly - https://www.linuxjournal.com/content/understanding-public-key-infrastructure-and-x509-certificates

- License server is not running or Swarm licenses are NOT installed - See chapter "HPE AutoPass License Server License Management" in AutoPass License Server User Guide for details of the web GUI management interface and how to install license.

- Swarm core components (docker containers) are not started/Errrors while starting . (SN, SWOP) - Check Swarm Install & Config guide for details of how to start Swarm.

- Swarm components are NOT able to see each other: Are the required ports exposed ?

- User is not using the Swarm APIs correctly - Check Swarm User Guide for details of API.

- Errors related to SWOP task defn / profile schema / SWCI init script – These are user defined artifacts. Check User Guide.

TROUBLESHOOTING – EXAMPLE ISSUE

Error code: 6002, as shown in shown screenshot happens when Swarm Learning components are not able to connect to the APLS server

Fix for above issue requires to make sure following two things.

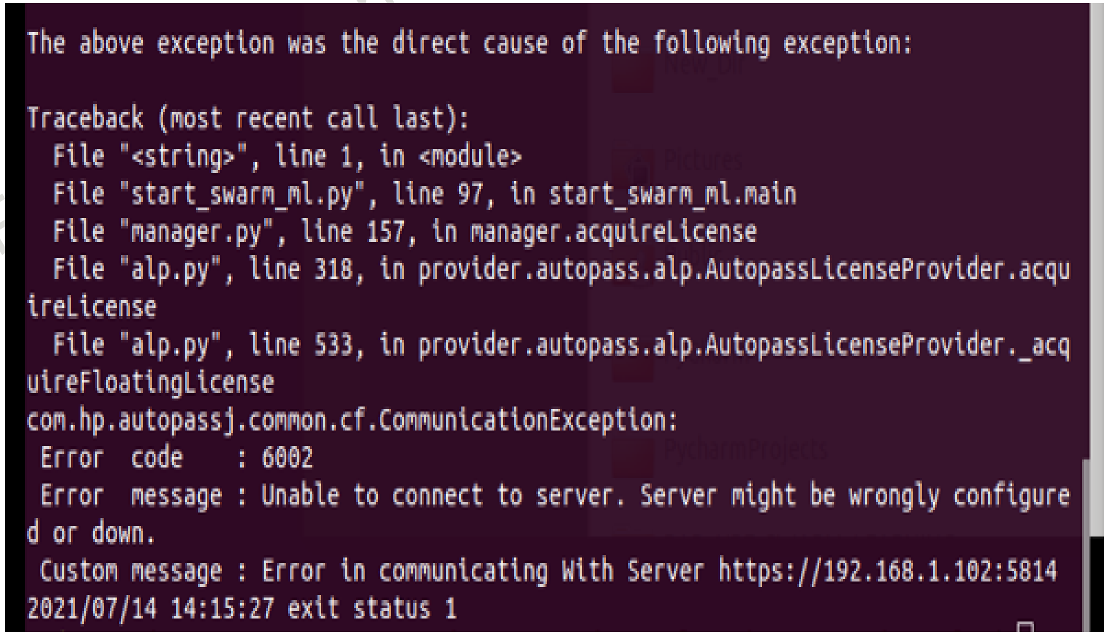
1. Verify APLS is running:

a. On License host : Check whether APLS is running. If not running, then license server needs to be started first.

b. Access APLS web management console. If the browser cannot connect, verify the network proxy settings, firewall policies, etc. that are in effect.

2. Setting up Swarm License

Download the Swarm License (MSC page). Install the license using APLS management console. For details refer APLS User Guide.



```
The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "<string>", line 1, in <module>
  File "start_swarm_ml.py", line 97, in start_swarm_ml.main
  File "manager.py", line 157, in manager.acquireLicense
  File "alp.py", line 318, in provider.autopass.alp.AutopassLicenseProvider.acqu
ireLicense
  File "alp.py", line 533, in provider.autopass.alp.AutopassLicenseProvider._acq
uireFloatingLicense
com.hp.autopassj.common.cf.CommunicationException:
 Error  code    : 6002
 Error  message : Unable to connect to server. Server might be wrongly configure
d or down.
 Custom message : Error in communicating With Server https://192.168.1.102:5814
2021/07/14 14:15:27 exit status 1
```

THANK YOU

LEARNING CHECK

What are the core components of Swarm Learning?

How do you know if SN node started successfully?

What ML models work with Swarm Learning?

Can each Swarm Learning run a different ML program and parameters?

What network ports does Swarm Learning use? Can they be customized?

What are the IP address of host systems used by Swarm containers ?