ICPS Anomaly Detection

Project Configuration

Samruddhi Khairnar

Introduction :

This project is aimed at building a simulated industrial cyber physical system using the Electra dataset, training an SVM model on it for predicting anomalous points in the dataset, processing and predicting the labels of a simulated stream of incoming data points and showing the output in the form of a dashboard.

Technology Stack :

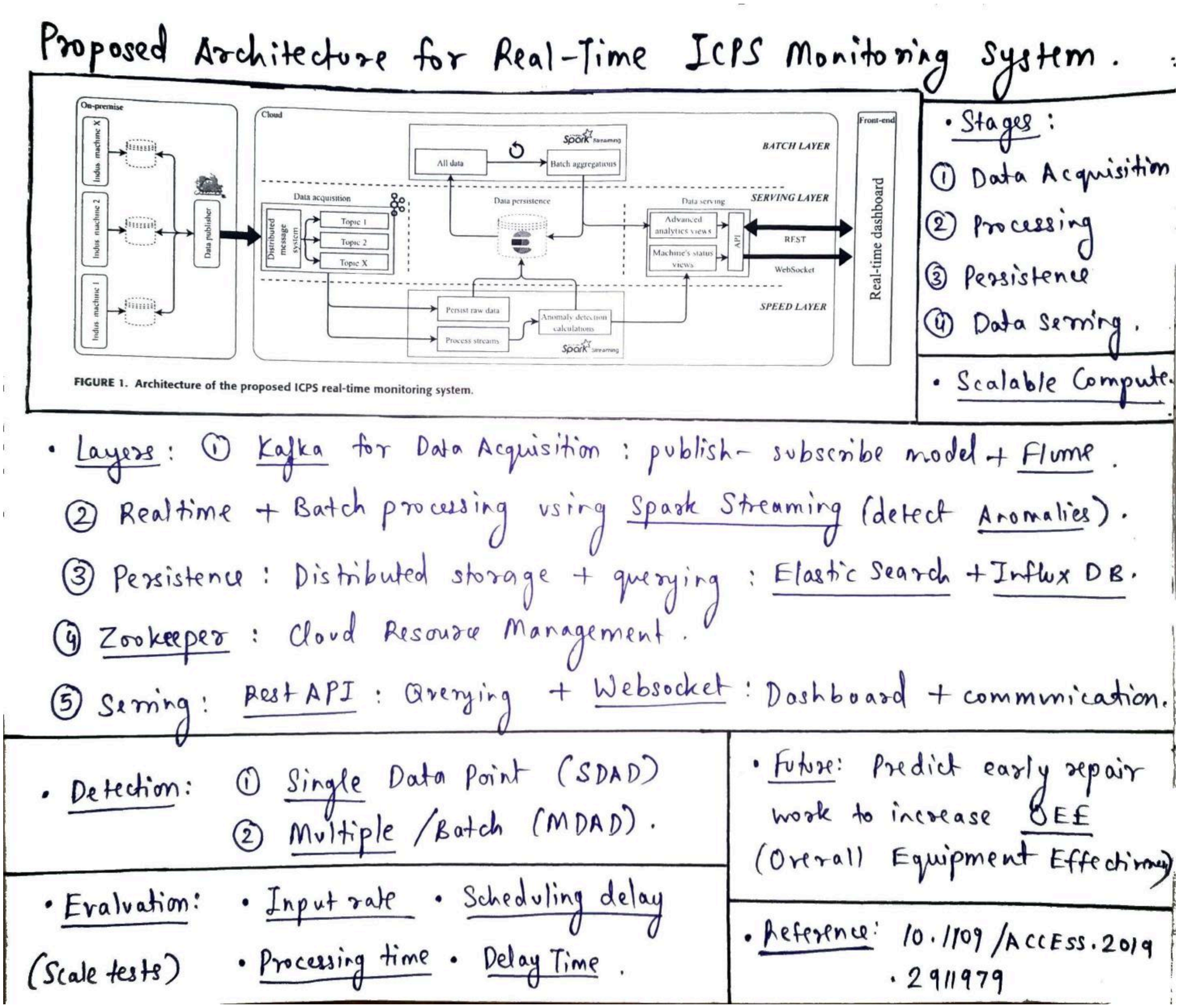
- Python3, JupyterLab, Colab.
- Dash library for Dashboard.
- Apache Spark for distributed ML model (SVM) training.
- Apache Hadoop for distributed storage.
- Apache Kafka for simulated pub-sub producer-consumer model for streaming data points in real time.

References :

1. **Project Architecture** :
   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8693711

2. **Electra Dataset** :
   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8926471

Proposed Architecture *(as mentioned in the base paper)* **:**



Proposed Architecture for Real-Time ICPS Monitoring System.

FIGURE 1. Architecture of the proposed ICPS real-time monitoring system.

- Stages :
  1. Data Acquisition
  2. Processing
  3. Persistence
  4. Data Serving.

- Scalable Compute.

- Layers : ① Kafka for Data Acquisition : publish- subscribe model + Flume.
  ② Realtime + Batch processing using Spark Streaming (detect Anomalies).
  ③ Persistence : Distributed storage + querying : Elastic Search + Influx DB.
  ④ Zookeeper : Cloud Resource Management.
  ⑤ Serving : Rest API : Querying + Websocket : Dashboard + communication.

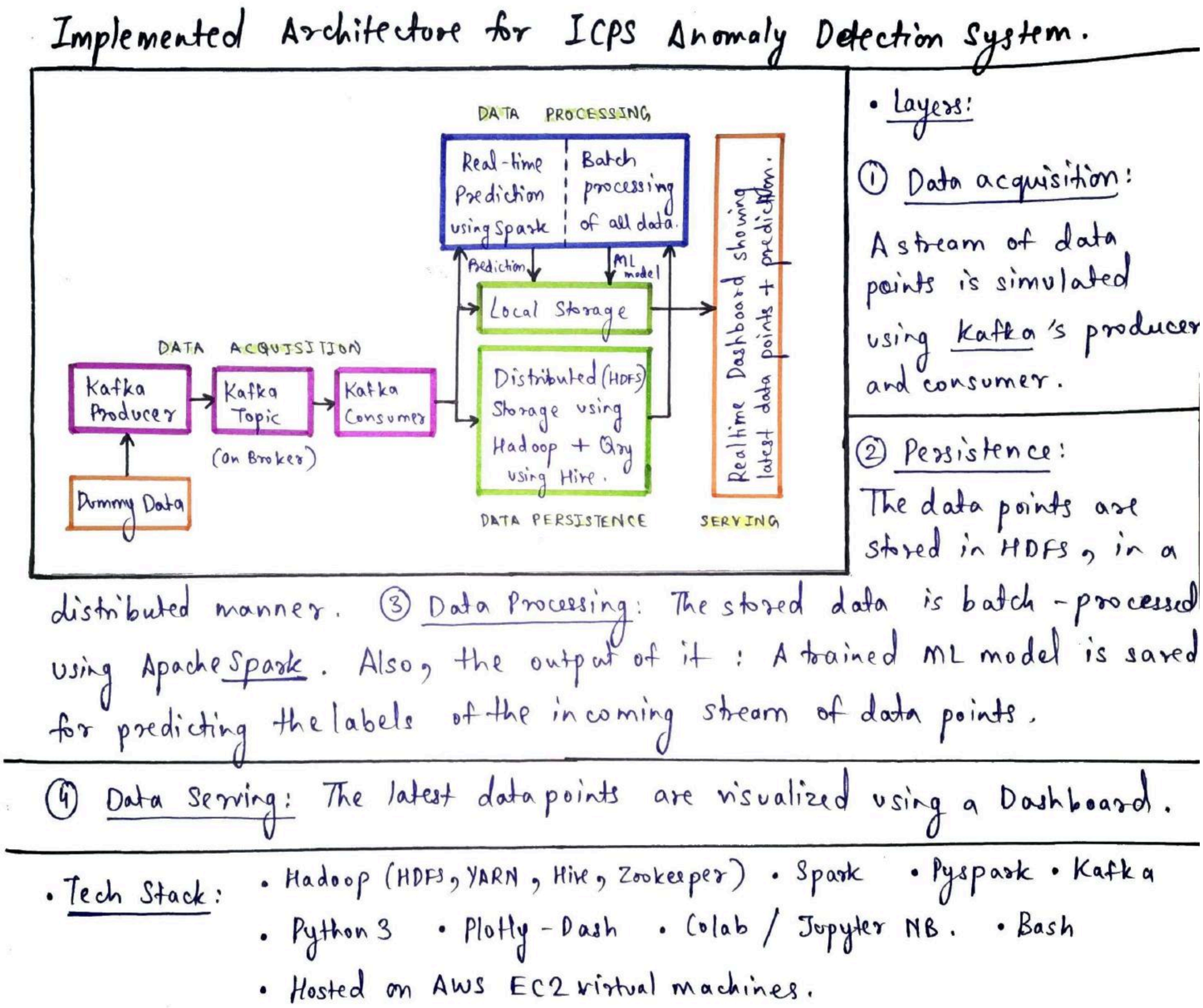
- Detection : ① Single Data Point (SDAD)
  ② Multiple /Batch (MDAD).

- Future : Predict early repair work to increase OEE (Overall Equipment Effectivness)

- Evaluation : (Scale tests) • Input rate • Scheduling delay • Processing time • Delay Time.

- Reference : 10.1109/ACCESS.2019.2911979

Implemented Architecture *(using a 3-node cluster on AWS EC2)* :



Implemented Architecture for ICPS Anomaly Detection System.

DATA PROCESSING
Real-time Prediction using Spark
Batch processing of all data.
Prediction
ML model
Local Storage
Realtime Dashboard showing latest data points + prediction.
DATA ACQUISITION
Kafka Producer
Kafka Topic
(On Broker)
Kafka Consumer
Dummy Data
Distributed (HDFS) Storage using Hadoop + Qry using Hive.
DATA PERSISTENCE
SERVING

- Layers:

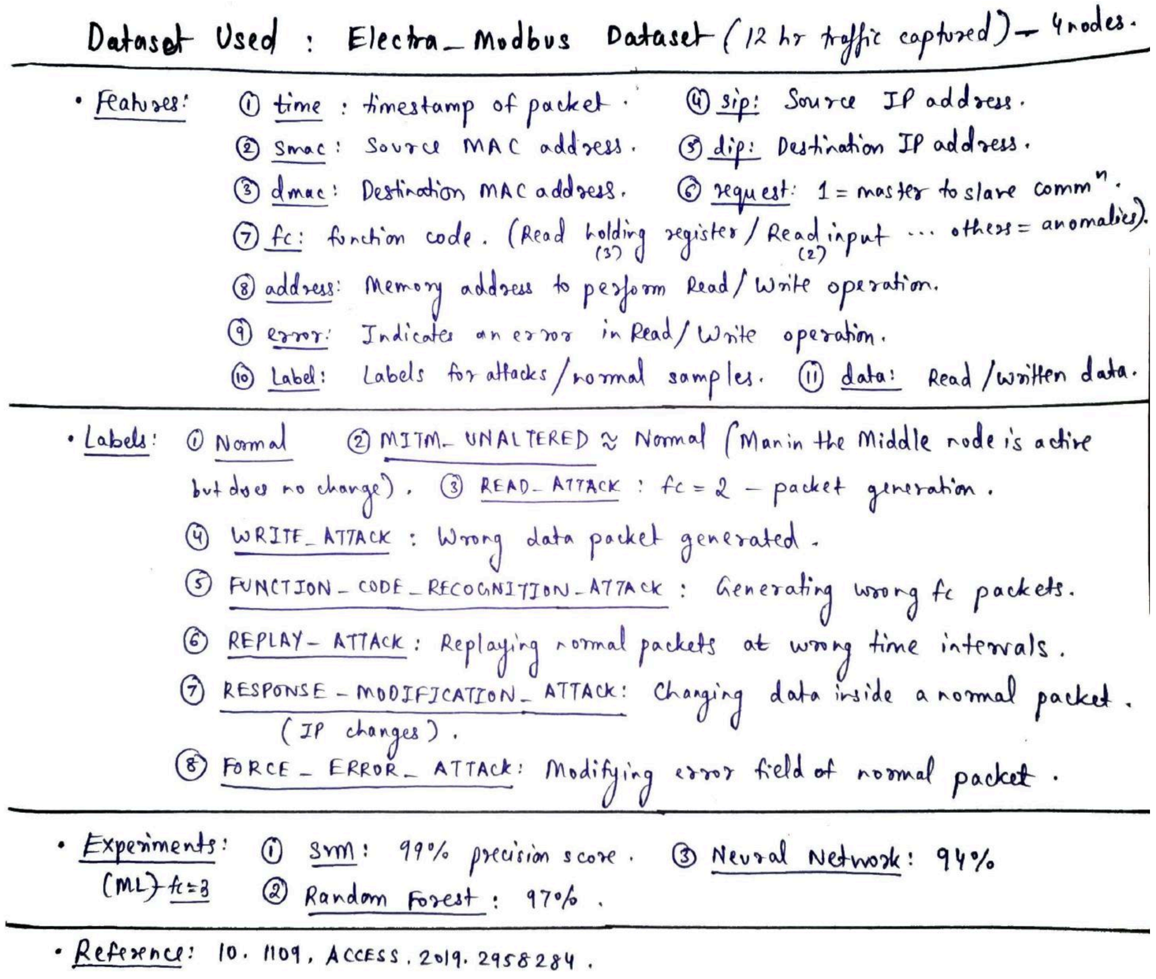
① Data acquisition: A stream of data points is simulated using Kafka's producer and consumer.

② Persistence: The data points are stored in HDFS, in a distributed manner. ③ Data Processing: The stored data is batch-processed using Apache Spark. Also, the output of it : A trained ML model is saved for predicting the labels of the incoming stream of data points.

④ Data Serving: The latest data points are visualized using a Dashboard.

- Tech Stack: • Hadoop (HDFS, YARN, Hive, Zookeeper) • Spark • Pyspark • Kafka
  • Python 3 • Plotly-Dash • Colab / Jupyter NB. • Bash
  • Hosted on AWS EC2 virtual machines.

Electra Dataset Description *(as mentioned in the reference paper)* :



Dataset Used : Electra_Modbus Dataset (12 hr traffic captured) – 4 nodes.

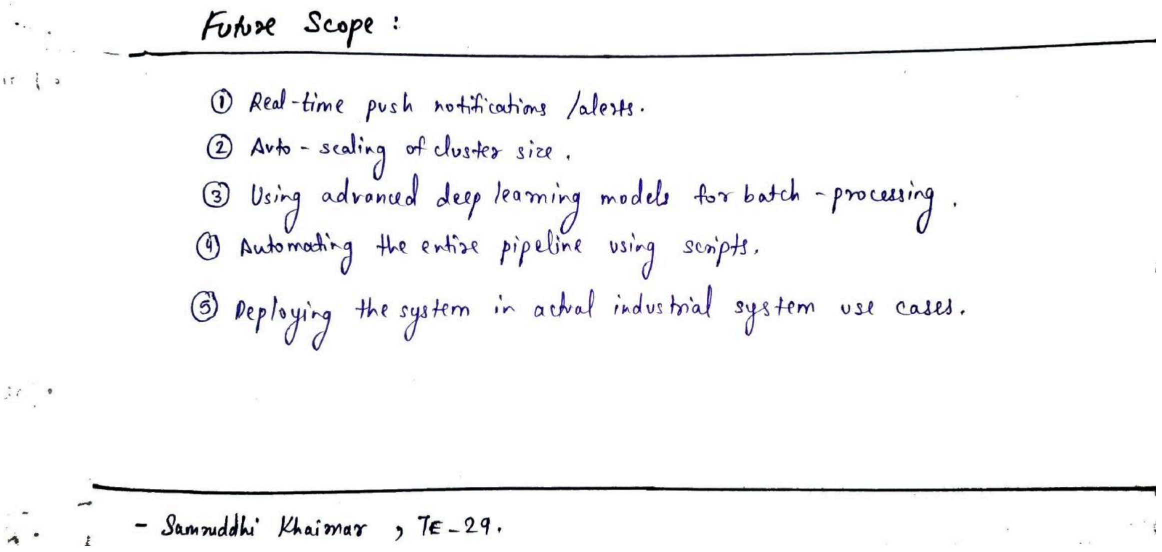
- Features:
  ① time : timestamp of packet.
  ② smac : Source MAC address.
  ③ dmac : Destination MAC address.
  ④ sip: Source IP address.
  ⑤ dip: Destination IP address.
  ⑥ request: 1 = master to slave comm^{n}.
  ⑦ fc: function code. (Read holding (3) register / Read (2) input ... others = anomalies).
  ⑧ address: Memory address to perform Read / Write operation.
  ⑨ error: Indicates an error in Read / Write operation.
  ⑩ Label: Labels for attacks / normal samples.
  ⑪ data: Read / written data.

- Labels:
  ① Normal
  ② MITM_UNALTERED ≈ Normal (Man in the Middle node is active but does no change).
  ③ READ_ATTACK : fc = 2 – packet generation.
  ④ WRITE_ATTACK : Wrong data packet generated.
  ⑤ FUNCTION_CODE_RECOGNITION_ATTACK : Generating wrong fc packets.
  ⑥ REPLAY_ATTACK : Replaying normal packets at wrong time intervals.
  ⑦ RESPONSE_MODIFICATION_ATTACK: Changing data inside a normal packet. (IP changes).
  ⑧ FORCE_ERROR_ATTACK: Modifying error field of normal packet.

- Experiments: (ML) → fc = 3
  ① svm: 99% precision score.
  ② Random Forest : 97%.
  ③ Neural Network: 94%

- Reference: 10.1109, ACCESS.2019.2958284.

Future Scope :

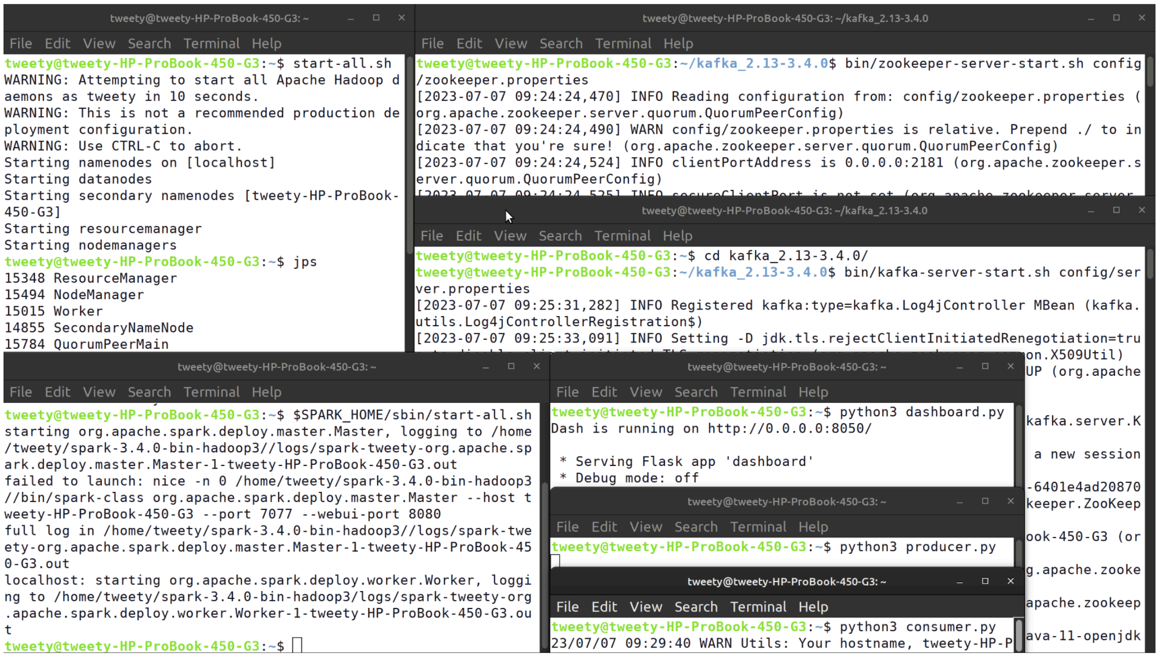


Future Scope :

① Real-time push notifications / alerts.
② Auto - scaling of cluster size.
③ Using advanced deep learning models for batch - processing.
④ Automating the entire pipeline using scripts.
⑤ Deploying the system in actual industrial system use cases.

– Samruddhi Khairnar , TE-29.

Dashboard :



tweety@tweety-HP-ProBook-450-G3: ~
File Edit View Search Terminal Help

```
tweety@tweety-HP-ProBook-450-G3:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop d
aemons as tweety in 10 seconds.
WARNING: This is not a recommended production de
ployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [tweety-HP-ProBook-
450-G3]
Starting resourcemanager
Starting nodemanagers
tweety@tweety-HP-ProBook-450-G3:~$ jps
15348 ResourceManager
15494 NodeManager
15015 Worker
14855 SecondaryNameNode
15784 QuorumPeerMain
```

tweety@tweety-HP-ProBook-450-G3: ~/kafka_2.13-3.4.0
File Edit View Search Terminal Help

```
tweety@tweety-HP-ProBook-450-G3:~/kafka_2.13-3.4.0$ bin/zookeeper-server-start.sh config
/zookeeper.properties
[2023-07-07 09:24:24,470] INFO Reading configuration from: config/zookeeper.properties (
org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2023-07-07 09:24:24,490] WARN config/zookeeper.properties is relative. Prepend ./ to in
dicate that you're sure! (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2023-07-07 09:24:24,524] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.s
erver.quorum.QuorumPeerConfig)
[2023-07-07 09:24:24,525] INFO secureClientPort is not set (org.apache.zookeeper.server
```

tweety@tweety-HP-ProBook-450-G3: ~/kafka_2.13-3.4.0
File Edit View Search Terminal Help

```
tweety@tweety-HP-ProBook-450-G3:~$ cd kafka_2.13-3.4.0/
tweety@tweety-HP-ProBook-450-G3:~/kafka_2.13-3.4.0$ bin/kafka-server-start.sh config/ser
ver.properties
[2023-07-07 09:25:31,282] INFO Registered kafka:type=kafka.Log4jController MBean (kafka.
utils.Log4jControllerRegistration$)
[2023-07-07 09:25:33,091] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=tru
                                                                              on.X509Util)
                                                                              UP (org.apache

                                                                              kafka.server.K

                                                                              a new session

                                                                              -6401e4ad20870
                                                                              keeper.ZooKeep

                                                                              ook-450-G3 (or

                                                                              g.apache.zooke

                                                                              apache.zookeep

                                                                              ava-11-openjdk
```

tweety@tweety-HP-ProBook-450-G3: ~
File Edit View Search Terminal Help

```
tweety@tweety-HP-ProBook-450-G3:~$ $SPARK_HOME/sbin/start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /home
/tweety/spark-3.4.0-bin-hadoop3//logs/spark-tweety-org.apache.sp
ark.deploy.master.Master-1-tweety-HP-ProBook-450-G3.out
failed to launch: nice -n 0 /home/tweety/spark-3.4.0-bin-hadoop3
//bin/spark-class org.apache.spark.deploy.master.Master --host t
weety-HP-ProBook-450-G3 --port 7077 --webui-port 8080
full log in /home/tweety/spark-3.4.0-bin-hadoop3//logs/spark-twe
ety-org.apache.spark.deploy.master.Master-1-tweety-HP-ProBook-45
0-G3.out
localhost: starting org.apache.spark.deploy.worker.Worker, loggi
ng to /home/tweety/spark-3.4.0-bin-hadoop3//logs/spark-tweety-org
.apache.spark.deploy.worker.Worker-1-tweety-HP-ProBook-450-G3.ou
t
tweety@tweety-HP-ProBook-450-G3:~$
```

tweety@tweety-HP-ProBook-450-G3: ~
File Edit View Search Terminal Help

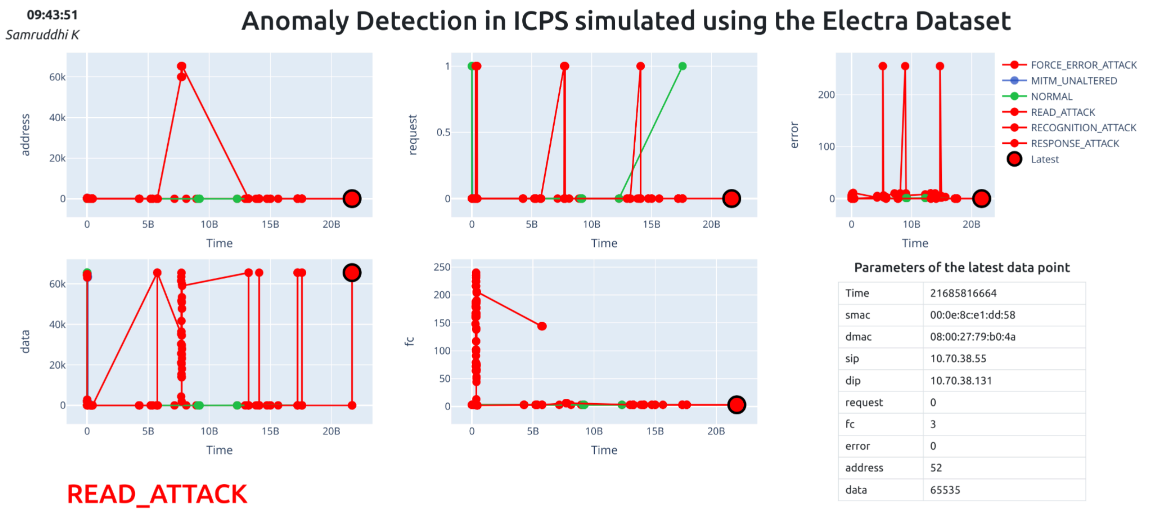
```
tweety@tweety-HP-ProBook-450-G3:~$ python3 dashboard.py
Dash is running on http://0.0.0.0:8050/

 * Serving Flask app 'dashboard'
 * Debug mode: off
```

tweety@tweety-HP-ProBook-450-G3: ~
File Edit View Search Terminal Help

```
tweety@tweety-HP-ProBook-450-G3:~$ python3 producer.py
```

tweety@tweety-HP-ProBook-450-G3: ~
File Edit View Search Terminal Help

```
tweety@tweety-HP-ProBook-450-G3:~$ python3 consumer.py
23/07/07 09:29:40 WARN Utils: Your hostname, tweety-HP-P
```



09:43:51

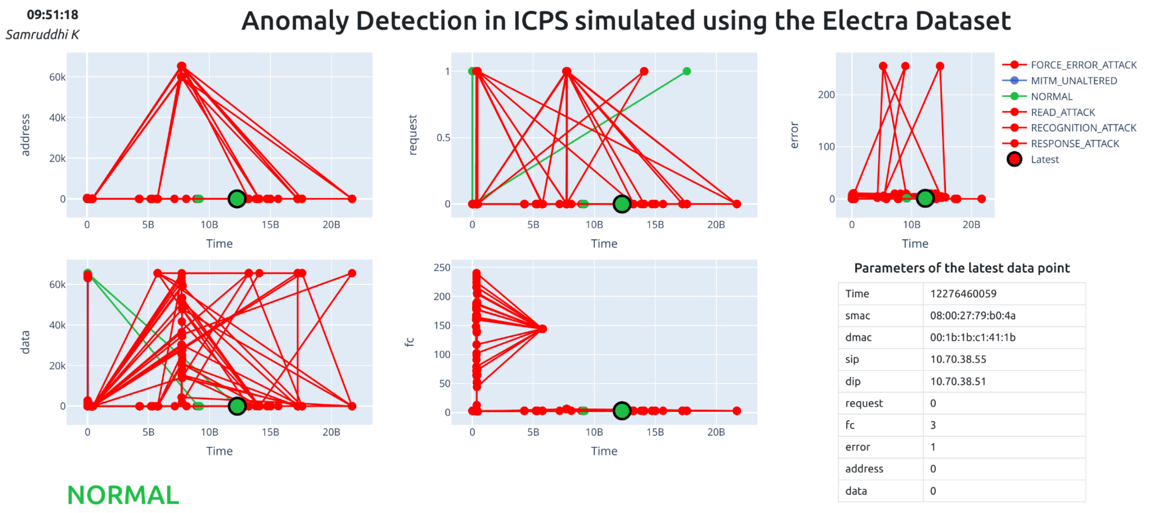
Samruddhi K

Anomaly Detection in ICPS simulated using the Electra Dataset

Parameters of the latest data point

Time	21685816664
smac	00:0e:8c:e1:dd:58
dmac	08:00:27:79:b0:4a
sip	10.70.38.55
dip	10.70.38.131
request	0
fc	3
error	0
address	52
data	65535

READ_ATTACK



09:51:18

Samruddhi K

Anomaly Detection in ICPS simulated using the Electra Dataset

Parameters of the latest data point

Time	12276460059
smac	08:00:27:79:b0:4a
dmac	00:1b:1b:c1:41:1b
sip	10.70.38.55
dip	10.70.38.51
request	0
fc	3
error	1
address	0
data	0

NORMAL

Original Source Code for Distributed 3-node cluster :

1. **Configuration** : *bit.ly/icps-1*
2. **Exploratory Data Analysis** : *bit.ly/icps-2*
3. **Model Training** : *bit.ly/icps-3*

Setup for Standalone single-node cluster :

Prerequisites

1. Install Ubuntu.
2. Install curl :

```
tweety@tweety-HP-ProBook-450-G3:~$ sudo apt-get update
tweety@tweety-HP-ProBook-450-G3:~$ sudo apt install curl
```

3. Install Java 11 :

```
tweety@tweety-HP-ProBook-450-G3:~$ sudo apt install openjdk-11-jdk
```

4. Check whether Python3 is installed :

```
tweety@tweety-HP-ProBook-450-G3:~$ python3 --version
```

5. Install gdown + add PATH :

```
tweety@tweety-HP-ProBook-450-G3:~$ pip install gdown
tweety@tweety-HP-ProBook-450-G3:~$ export PATH=/home/<username>/.local/bin
# Replace with your username
```

Hadoop, Spark, Kafka Installation

1. Download the configured setups + untar it (*setup for single node cluster*) :

```
tweety@tweety-HP-ProBook-450-G3:~$ gdown 10byFo7JAgABKUUD0kcGBZ-RgoPQ5lq_D
tweety@tweety-HP-ProBook-450-G3:~$ gdown 1UJQDaSWbAXThG4jp9eVjbESI7gISd0Dh
tweety@tweety-HP-ProBook-450-G3:~$ gdown 1QoMG2H4EBVtM09rQtY4GZFz65FT-XeKh
tweety@tweety-HP-ProBook-450-G3:~$ gdown 1hbUdWQso2xYTt7OA4-PWFIDiJOfXdS9q
tweety@tweety-HP-ProBook-450-G3:~$ gdown 1P7k3LLtlQHrIdewdtYghfFeQdviwg-gw
tweety@tweety-HP-ProBook-450-G3:~$ gdown 1S5snu1xmB4Arcyf36ICu4n-HqID-oa7a
tweety@tweety-HP-ProBook-450-G3:~$ tar -xzvf hadoop.tar.gz
tweety@tweety-HP-ProBook-450-G3:~$ tar -xzvf spark.tar.gz
tweety@tweety-HP-ProBook-450-G3:~$ tar -xzvf kafka.tar.gz
```

2. Make these temp directories (*as configured in the setup ./hadoop-3.3.5/etc/hadoop/core-site and hdfs-site.xml + required for kafka, zookeeper logs)* + give them sudo privileges :

```
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /hadoop
tweety@tweety-HP-ProBook-450-G3:~$ sudo chmod -R 777 /hadoop/
tweety@tweety-HP-ProBook-450-G3:~$ sudo chown -R <username> /hadoop
# Replace with your username

tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /hadoop/tmp
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /hadoop/dfs
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /hadoop/dfs/data
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /hadoop/dfs/name
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /app
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /app/zookeeper
tweety@tweety-HP-ProBook-450-G3:~$ sudo chmod 777 /app/zookeeper
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /app/kafka-logs
tweety@tweety-HP-ProBook-450-G3:~$ sudo mkdir /app/kafka-logs/logs
tweety@tweety-HP-ProBook-450-G3:~$ sudo chmod 777 /app/kafka-logs/logs
```

3. Get the present working directory and copy the path (*ctrl+shift+C*) :

```
tweety@tweety-HP-ProBook-450-G3:~$ pwd
```

4. Add the installation directories to the PATH, open bashrc :

```
tweety@tweety-HP-ProBook-450-G3:~$ nano .bashrc
```

Type these lines to end of the file then, add pwd copied from previous step and (***save*** : ctrl+S -> ***exit*** : ctrl+X) :

```
export HADOOP_HOME=<pwd>/hadoop-3.3.5/
export JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64/"
export SPARK_HOME=<pwd>/spark-3.4.0-bin-hadoop3/
export KAFKA_HOME=<pwd>/kafka_2.13-3.4.0/
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
```

5. Source the changes :

```
tweety@tweety-HP-ProBook-450-G3:~$ source .bashrc
```

Source Code

1. Unzip and move the source code files :

```
tweety@tweety-HP-ProBook-450-G3:~$ sudo apt install unzip
tweety@tweety-HP-ProBook-450-G3:~$ export PATH=/usr/bin:/bin
```

```
tweety@tweety-HP-ProBook-450-G3:~$ unzip electra-model.zip
tweety@tweety-HP-ProBook-450-G3:~$ unzip files.zip
tweety@tweety-HP-ProBook-450-G3:~$ unzip files2.zip
tweety@tweety-HP-ProBook-450-G3:~$ mv files/* . -f
tweety@tweety-HP-ProBook-450-G3:~$ mv files2/* . -f
```

Installation of Dependencies

1. Install the dependencies :

```
tweety@tweety-HP-ProBook-450-G3:~$ pip3 install pandas pyspark confluent_kafka dash dash_bootstrap_components
```

Start Hadoop, Spark, Kafka

1. Setup passwordless SSH :

```
 tweety@tweety-HP-ProBook-450-G3:~$ sudo apt install openssh-client
openssh-server

# Add the following lines to /etc/ssh/sshd_config
PubkeyAuthentication yes
PasswordAuthentication no
ChallengeResponseAuthentication no

tweety@tweety-HP-ProBook-450-G3:~$ chmod 700 .ssh | chmod 644
.ssh/id_rsa.pub | chmod 600 .ssh/authorized_keys | chmod 600 .ssh/id_rsa
| chmod 755 ~

# Init ssh each time using :
sudo service ssh restart (or start)

 tweety@tweety-HP-ProBook-450-G3:~$ ssh-keygen -t rsa; cp
~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
 # Press enter for each line

# Verify the passwordless setup using :
ssh localhost
```

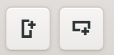
2. Start Hadoop and put the initial dataset in the HDFS :

```
tweety@tweety-HP-ProBook-450-G3:~$ source .bashrc
tweety@tweety-HP-ProBook-450-G3:~$ hdfs namenode -format
tweety@tweety-HP-ProBook-450-G3:~$ start-all.sh
```

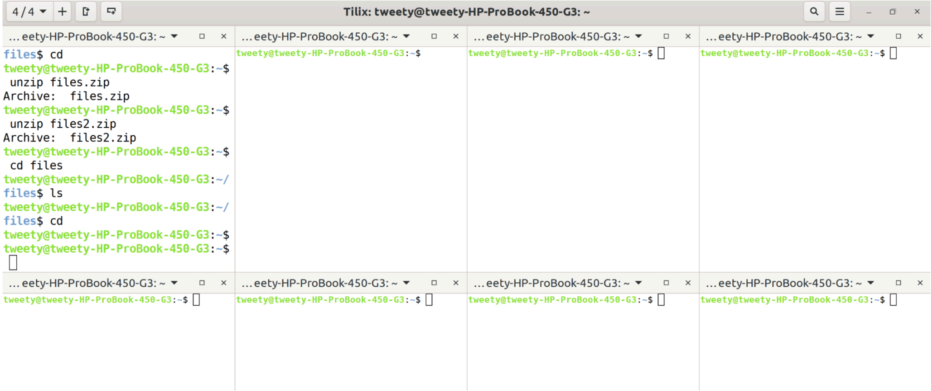
```
# Verify the running of Hadoop using :
tweety@tweety-HP-ProBook-450-G3:~$ jps

tweety@tweety-HP-ProBook-450-G3:~$ hdfs dfs -mkdir /electra
tweety@tweety-HP-ProBook-450-G3:~$ hdfs dfs -put ./d.csv /electra/electra_modbus.csv
```

3. Start Spark :

```
tweety@tweety-HP-ProBook-450-G3:~$ $SPARK_HOME/sbin/start-all.sh
```

4. Install and start Tilix - *will be required for multiple terminal windows simultaneously (example shown below) - use the*  *buttons to open multiple terminals :*

```
tweety@tweety-HP-ProBook-450-G3:~$ sudo apt install tilix
tweety@tweety-HP-ProBook-450-G3:~$ tilix
```


4 / 4
Tilix: tweety@tweety-HP-ProBook-450-G3: ~
... eety-HP-ProBook-450-G3: ~

```
files$ cd
tweety@tweety-HP-ProBook-450-G3:~$
 unzip files.zip
Archive:  files.zip
tweety@tweety-HP-ProBook-450-G3:~$
 unzip files2.zip
Archive:  files2.zip
tweety@tweety-HP-ProBook-450-G3:~$
 cd files
tweety@tweety-HP-ProBook-450-G3:~/
files$ ls
tweety@tweety-HP-ProBook-450-G3:~/
files$ cd
tweety@tweety-HP-ProBook-450-G3:~$
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

```
tweety@tweety-HP-ProBook-450-G3:~$
```

5. In one Tilix terminal, start Zookeeper (*keep it open*):

```
tweety@tweety-HP-ProBook-450-G3:~$ cd $KAFKA_HOME
tweety@tweety-HP-ProBook-450-G3:~$ bin/zookeeper-server-start.sh
config/zookeeper.properties
```

6. In another Tilix terminal and start Kafka server (*keep it open*) :

```
tweety@tweety-HP-ProBook-450-G3:~$ cd $KAFKA_HOME
tweety@tweety-HP-ProBook-450-G3:~$ bin/kafka-server-start.sh
config/server.properties
```

7. In the third Tilix terminal, create a new Kafka topic :

```
tweety@tweety-HP-ProBook-450-G3:~$ cd $KAFKA_HOME/bin
tweety@tweety-HP-ProBook-450-G3:~$ ./kafka-topics.sh --create --topic
electra --bootstrap-server localhost:9092
```

8. In the fourth Tilix terminal, start the Kafka consumer (*consumes from topic electra and stores in HDFS + locally in d.csv*) :

```
tweety@tweety-HP-ProBook-450-G3:~$ cd
tweety@tweety-HP-ProBook-450-G3:~$ mv ./content/electra-model/ .
tweety@tweety-HP-ProBook-450-G3:~$ python3 consumer.py
```

9. In the fifth Tilix terminal, start the Kafka producer (*pushes data into topic electra and from dummy.csv*) :

```
tweety@tweety-HP-ProBook-450-G3:~$ python3 producer.py
```

10. In the sixth Tilix terminal, start the dashboard process (*displays from d.csv - latest 200 data points and gives params + prediction of the latest data point*) :

```
tweety@tweety-HP-ProBook-450-G3:~$ python3 dashboard.py
```

11. Run ml.py once, every week, to train on the entire dataset :

Create new file to store the cron job command, make it executable and open it :

```
tweety@tweety-HP-ProBook-450-G3:~$ touch run.sh
tweety@tweety-HP-ProBook-450-G3:~$ chmod +x cron.sh
tweety@tweety-HP-ProBook-450-G3:~$ nano run.sh
```

*Enter this in to the file (**paste** :* ctrl+shift+V -> ***save*** : ctrl+S -> ***exit*** : ctrl+X) *:*

```
#!/bin/bash
while true; do
python3 ml.py
sleep 604800
done
```

Run this cronjob in the seventh Tilix Terminal *:*

```
tweety@tweety-HP-ProBook-450-G3:~$ ./run.sh
```

View the Dashboard using a Browser :

http://127.0.0.1:8050/

Stop all Processes :

Stop Hadoop :

```
tweety@tweety-HP-ProBook-450-G3:~$ stop-all.sh
```

Stop Spark :

```
tweety@tweety-HP-ProBook-450-G3:~$ $SPARK_HOME/sbin/stop-all.sh
```

Stop other processes running in terminals (Kafka, Zookeeper, Python3 codes) :

```
Press Ctrl + C
```