Seneca POLYTECHNIC

Transactional Machine Learning (TML) Solution Studio (TSS) and Cisco Packet Tracer (CPT)

Professor: Sebastián Maurice

Course: CYT-160

Elaborate by:

Mit Jayesh Doshi-174482232

Darsh Jitendra Modessa-157829185

Hetkumar Thakkar-123765182

Leandro Delgado-114416241

December 5,2024

A. **What problem is your solution solving (be as detailed as possible)? How is your solution solving this problem (be as detailed as possible)?**

Real-time data processing and cybersecurity solutions typically need to be developed using the resources of egregious coding skills, a deep understanding of tools and frameworks, and time-consuming manual effort to setup infrastructure, machine learning pipelines, and integrations. Unfortunately, this makes the process expensive, yet time consuming and prone to errors, restricting accessibility to smaller teams or more inexperienced developers.

This complexity is eliminated with the TML Solution Studio (TSS). This is a no-code or low code approach to the process which TML transforms into a streamlined process where users can leverage minimal effort to build advanced, scalable. Users can deploy solutions with prewritten Directed Acrylic graphs (DAGs), by simply configuring, users can automate documentation, integrate AI models with PrivateGPT, and streamline workflow using Apache Kafka and air flow tools.

This solution enables cyber security and data processing in real time with versatile, scalable, and inexpensive tools paired with Ai and cutting-edge technologies. It uses TSS and its 10 prewritten Apache Airflow DAGs to provide an end to end, real time data processing and monitoring system, that is secure, scalable, and easy to configure.

Key Problems Solved by the Solution

1. **Real-Time Threat Detection and Network Monitoring**

In real time, TML processes data streams to find anomalies or abnormal network traffic behaviors. The system uses entity-based machine learning to continuously build models to predict future behaviors and flag IPs or devices at risk.

2. **Effortless Data Streaming and Integration**

According to the chosen method for real time data coming in from devices, including MQTT. Apache Kafka provides topics to store input and output data by creating new topics for data streaming and data flow.

3. **In-Memory Machine Learning**

While other streaming solutions require external databases, TML performs entity-based machine learning in memory. It speeds up the process, lowers costs, and increases scalability all the while keeping precision at the entity level.

4. **Seamless AI Integration**

Solution with PrivateGPT and Qdrant vector database integration for generative AI for deeper insight analysis, anomaly analysis and visualization.

5. **Real-Time Visualization and Automation**

A dashboard of real time threat indicators and at-risk IP addresses is visualized on dashboard watched by client browsers streamed via WebSocket. From preprocessing to visualization, deployment, and documentation, all these processes are done by automated processes.

B. **How is your solution solving this problem?**

i. **What is TML doing?**

The solution leverages TML and Confluent Kafka Cloud in conjunction with Kubernetes and Cisco Packet Tracer to efficiently process, visualize and manage real-time data securely and at scale. Kubernetes together with TML's JSON-centric approach allows for dynamically scaling microservices while keeping the processing of streaming data lightweight and cost effective. Kafka Cloud comes with solid data storage and streaming capabilities that guarantee high availability and throughput. Furthermore, Cisco Packet Tracer lets you do simulation and testing of secure network configurations to support real time data flow.

1. **How is it processing the JSON data?**

There are three core binaries included in TML; Viper, HPDE, and Viperviz, along with its Python library, all working together to parse and manage streams of JSON data efficiently. The Viper binary connects to Apache Kafka to preprocess data, stream it for machine learning, and generate training datasets for the HPDE binary, which does hyper predictions and entity level machine learning in real time. We preprocess JSON data using predefined paths of keys and values to extract key and value with filtering criteria of uid, datetime, and subtopics. This is controlled by JSONCRITERIA to schedule Kafka stream inputs for machine learning or visualization.

It's all in memory, no external databases, low latencies. TML recognizes the need to analyze data in real time but still be able to be flexible and scalable, so we focus on JSON processing (instead of traditional SQL query) to do the real time analysis faster and cost-effective.

2. **How is it dashboarding the data?**

For real time data visualization TML uses the Viperviz binary. Viperviz streams processed data directly to the client browser through websockets to provide responsive, dynamic dashboards without the need for third party tools like Tableau or Power BI. Customizable dashboards that provide trends, metrics and alerts based on processed JSON data are provided for users to interact with.

3. **Where is the data being stored in Confluent Kafka Cloud?**

Confluent Kafka Cloud is used to store and stream processed data streams. From the point of view of reliability and scalability, it stores different streams of processed data organized in Kafka topics. Kafka's partitioning of each topic maps to a specific type of processed output, and both the high throughput data stream and the guaranteed delivery of messages make Kafka perfect for topics.

ii. **What is happening in Cisco Packet Tracer?**

Using Cisco Packet Tracer, simulated network environments are configured in order to test transmission of data between TML components, Kafka Cloud, and external clients. They include the simulation of firewalls, routers and switches to make sure network paths are optimized for real time data flow and still maintain robust security policies. Packet Tracer gives you that verity that the underlying network infrastructure of which TML relies on to stream and process can function without bottlenecks or vulnerabilities.

C. **What are all the technology components? Provide screenshots (as needed)**

Technology Components of TML:

1. **Three Core Binaries**
   TML three binaries which work in micro services for purposes of scalable data processing.:

- **Viper**: Executes a number of significant operations like stream data to Kafka, stream data preprocessing, and training dataset construction. It interacts with REST APIs for real time data processing and ML operations.
- **HPDE** (Hyper-Prediction Engine): It executes real time machine learning and hyper predictions for every entity using sliding time windows to provide rapid predictions in seconds.
- **Viperviz**: It provides real time streaming visualizations on WebSockets so that users can build cost effective dashboards without needing third party tools like Tableau or PowerBI.

2. **MAADSTML Python Library**
   This library can be used as the base API for building TML solutions by linking it to the Viper binary and control the processing of any field in stream.

3. **Apache Kafka**
   Can be integrated for real time data streaming but can operate on premises or in cloud (AWS MSK, Confluent Cloud).

4. **Docker Containers**
   Insured portability and scalability is achieved on all TML solutions through containerization for seamless production deployment.

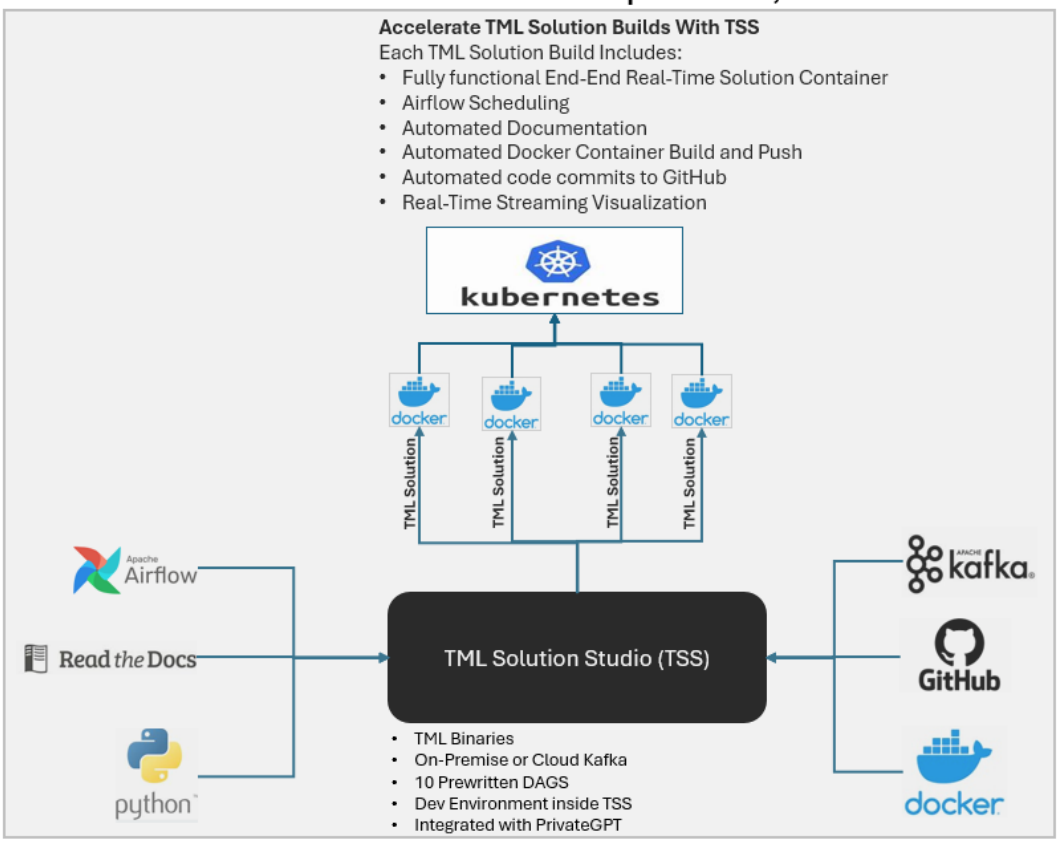
5. **Kubernetes**
   Scaling and unlimited real time data processing capabilities are provided by TML with the help of Kubernetes for the Docker based solutions.

6. **PrivateGPT**
   Provides generative AI capabilities for advanced analytics and fast, real time ai processing.

7. **TML Solution Studio (TSS) Container**
   It brings an easy to use platform for developing TML solutions, aiming for a rapid deployment and management.

Combining these components together cohesively, each of these components works to enable scalable, cost efficient, and real time data processing, machine learning and visualization in TML solutions.

TSS: Build Advanced Real-Time Solutions For the Enterprise Faster, Easier and within Compliance



D. **How are all the technology components linked or integrated (be as detailed as possible)? Provide screenshots (as needed)**

MAADSTML Python library is used to build TML solutions and establish a connection to the TML binaries via REST APIs to allow real time data streaming to Apache Kafka. It processes data, and provides a way to do machine learning, all in a seamless way with Kafka for handling large scale data.

TML solutions are later containerized via Docker for quick deployment and then scaled with Kubernetes to guarantee high availability and high performance. It enables TML to manage real time workloads in a scalable and reliable manner.

E. **Explain how you built the solution in the TSS.**

To create the (TSS) solution with TML and Docker, the process began with the configuration of Docker, ensuring that it was installed and working correctly. Then, we proceeded to download the TML image, creating a configuration file in YAML or JSON format to define the source of the data to be processed and its subsequent destination. After having made this

configuration, we proceeded to save the file in a folder where docker can access it when required. Once these parameters were established, we proceeded to run the TML container using a docker run command to start the container, ensuring the linking of the configuration file and the data directory. By running this command, we are ensuring that the container knows the location of its configuration.

Once it was confirmed that the container is running, the logs were reviewed with docker logs to ensure that everything is working. It was also checked if the telemetry data was being processed and sent to the correct destination. When you create a docker-compose yml file, you have the versatility to set up the services you need, including configurations, data directories, and ports. The start of the process is super simple and requires the use of the command to be started. Finally, following the procedure shown in tmlreadthedocs, this configuration was able to be carried out reliably, although facing challenges in certain errors, it was possible to achieve its configuration.

F. **What are the answers to the 16 questions?**

a. **Which machines are HOSTS AT RISK?**

To determine the host at risk we must identify abnormal behavior like ping status = -1.

For given analysis Hosts at risk: 192.168.6.101, 192.168.5.23, 192.168.5.24, 192.168.5.25, 192.168.5.26, 192.168.5.27, 192.168.5.28, 192.168.5.29, 192.168.5.30, 192.168.5.31, 192.168.5.100, 192.168.6.17, 192.168.6.23, 192.168.6.18, 192.168.5.21, 192.168.5.22

b. **Which HOSTS AT RISK cannot be pinged?**

N/A

c. **What is the avg outbound packets for HOSTS AT RISK?**

Hosts at Risk and Average Outbound Packets: {'192.168.5.100': 301.0, '192.168.5.21': 245.15, '192.168.5.22': 302.45, '192.168.5.23': 363.82, '192.168.5.24': 266.93, '192.168.5.25': 246.158, '192.168.5.26': 275.63, '192.168.5.27': 323.26, '192.168.5.28': 324.78, '192.168.5.29': 303.31, '192.168.5.30': 325.63, '192.168.5.31': 271.68, '192.168.6.17': 326.68, '192.168.6.18': 328.47, '192.168.6.23': 246.68}

d. **What is the avg inbound packets for HOSTS AT RISK?**

Hosts at Risk and Average Inbound Packets: {'192.168.5.100': 274.526, '192.168.5.21': 258.75, '192.168.5.22': 268.85, '192.168.5.23': 330.9, '192.168.5.24': 313.3025, '192.168.5.25': 293.316, '192.168.5.26': 305.0, '192.168.5.27': 340.0, '192.168.5.28': 274.526, '192.168.5.29': 328.158, '192.168.5.30': 325.579, '192.168.5.31': 284.579, '192.168.6.17': 268.263, '192.168.6.18': 289.0, '192.168.6.23': 365.263}

e. **What is the maximum outbound packet for HOSTS AT RISK?**

Hosts at Risk and Maximum Outbound Packets: {'192.168.5.100': 516, '192.168.5.21': 505, '192.168.5.22': 499, '192.168.5.23': 522, '192.168.5.24': 439, '192.168.5.25': 483, '192.168.5.26': 470, '192.168.5.27': 496, '192.168.5.28': 509, '192.168.5.29': 461, '192.168.5.30': 522, '192.168.5.31': 494, '192.168.6.17': 513, '192.168.6.18': 523, '192.168.6.23': 485}

f. **What is the maximum inbound packets for HOSTS AT RISK?**

Hosts at Risk and Maximum Inbound Packets: {'192.168.5.100': 520, '192.168.5.21': 493, '192.168.5.22': 511, '192.168.5.23': 505, '192.168.5.24': 510, '192.168.5.25': 476, '192.168.5.26': 491, '192.168.5.27': 521, '192.168.5.28': 523, '192.168.5.29': 480, '192.168.5.30': 495, '192.168.5.31': 515, '192.168.6.17': 486, '192.168.6.18': 506, '192.168.6.23': 482}

g. **Which HOSTS AT RISK have increasing outbound packets in bytes?**

Hosts at Risk with Increasing Outbound Packets in Bytes: ['192.168.5.23' '192.168.5.24' '192.168.5.25' '192.168.5.26' '192.168.5.28' '192.168.5.30' '192.168.5.31' '192.168.5.100' '192.168.6.17' '192.168.6.23' '192.168.6.18']

h. **Which HOSTS AT RISK have decreasing outbound packets in bytes?**

Hosts at Risk with Decreasing Outbound Packets in Bytes: ['192.168.5.27' '192.168.5.29' '192.168.5.21' '192.168.5.22' '192.168.5.23' '192.168.5.25']

i. **Which HOSTS AT RISK have increasing inbound packets in bytes?**

Hosts at Risk with Increasing Inbound Packets in Bytes: ['192.168.5.24' '192.168.5.26' '192.168.5.29' '192.168.5.30' '192.168.5.31' '192.168.6.17' '192.168.6.18' '192.168.5.22' '192.168.5.23' '192.168.5.25']

j. **Which HOSTS AT RISK have decreasing inbound packets?**

Hosts at Risk with Decreasing Inbound Packets in Bytes: ['192.168.5.25' '192.168.5.27' '192.168.5.28' '192.168.5.100' '192.168.6.23' '192.168.5.21']

k. **Which machines are down? on switch 1.**

Machines Down on Switch 1:

['192.168.5.23' '192.168.5.24' '192.168.5.25' '192.168.5.26'

'192.168.5.27' '192.168.5.28' '192.168.5.29' '192.168.5.30'

'192.168.5.31' '192.168.5.100' '192.168.5.21' '192.168.5.22']

l. **Which machines are down? on switch 2.**

Machines Down on Switch 2:

['192.168.6.101' '192.168.6.17' '192.168.6.23' '192.168.6.18']

m. **Which machine has the highest variance in inbound packets?**

Machine with Highest Inbound Packet Variance: 192.168.5.31 (Variance: 22351.72)

n. **Which machine has the highest variance in outbound packets?**

Machine with Highest Outbound Packet Variance: 192.168.6.18 (Variance: 24147.3)

o. **Which machine has an anomaly probability in outbound packets above 80%**

Machines with Anomaly Probability in Outbound Packets Above 80%: ['192.168.5.31' '192.168.6.17']

p. **Which machine has an anomaly probability in inbound packets above 80%**

Machines with Anomaly Probability in Inbound Packets Above 80%: ['192.168.5.27' '192.168.5.29' '192.168.5.22']

G. **How can the solution be improved (be as detailed as possible)?**

This upgrade includes fixing vulnerabilities, improving current features, and providing some additional features for greater efficiency, scalability, usability, and security. Here are detailed proposals to enhance Transactional Machine Learning (TML) Solution Studio (TSS) and its integration with Cisco Packet Tracer (CPT):

Features and Additional Integrations

Integration into the Cloud Services:

Comprehensive support of AWS IoT, Google Cloud Pub/Sub, or even Azure Event Hub for true cloud operational integration.

Provide possibilities to deploy TML into a cloud environment for simple scalability.

AI-Driven Recommendations:

Intelligent recommendations for configurations in the area of optimizing the network and security, according to the user's historical data.

Gamification in CPT:

Gamification in CPT would help the learner to be involved in this difficult subject as they configure complex networks.

H. **How does this solution compare to other commercial solutions (be as detailed as possible)?**

This solution is unique in the implementation of scalable and flexible cloud services such as AWS IoT, Google Cloud Pub/Sub, and Azure Event Hub. AI-driven recommendations further enhance network optimization and security to compete with other tools such as Splunk and Palo Alto Cortex. The gamification feature for learning network configurations in Cisco Packet Tracer is one of a kind over most commercial platforms. While it grants robust modularity and cost-effectiveness, the addition of advanced features like hybrid-cloud orchestration and threat intelligence would easily place it at par or even beyond other market-leading services like AWS CloudWatch and IBM QRadar. In that case, this platform could outperform many other solutions on scalability, security, and user experience.

I. **What are the advantages and disadvantages of this solution (be as detailed as possible)?**

Advantages

- **Fast Deployment:** Solutions can be built in under 2 minutes, with automated documentation, Docker builds, and GitHub commits.
- **Cost Efficiency:** Unlike traditional stream processing that relies on external databases, TML uses in-memory processing and operates with lightweight binaries, reducing storage, compute, and network costs.
- **Advanced Machine Learning:** TML enables entity-level, in-memory machine learning for real-time data. This provides deeper insights into individual device behaviors and enhances predictive accuracy.
- **Ease of Use:** TML is a low-code/no-code platform using TML Solution Studio (TSS), allowing users to quickly design, deploy, and manage solutions with minimal coding.
- **JSON Processing Over SQL:** Unlike conventional technologies that rely on SQL, TML processes data in-memory with JSON, offering faster, more efficient, and easier data management.
- **GenAI Integration:** TML integrates with GenAI tools like PrivateGPT and Qdrant vector databases, making it the first solution to combine fast AI capabilities with real-time data processing and machine learning.

- **Pre-Built Data Ingestion:** TML provides pre-built Python clients for.

Disadvantages

To configure TML and be able to handle telemetry data effectively, it is necessary to have knowledge of its syntax and capabilities because it requires a learning process to be able to manage containers, networks and volumes. Another disadvantage can be its performance. Even though Dockers is lightweight, compared to virtual machines, running too many containers can cause problems for applications that require real-time data processing. Also, monitoring plays an important role due to the docker does not being integrated with tools to monitor its performance, so it is advisable to add additional systems such as Prometheus or Grafana, which could cause greater complexity when using it.

Debugging problems within containers can be a bit more complex on the normal server. Network and data management brings its own complications. Establishing secure and efficient communication between containers can be a challenge, but if data is stored persistently, volumes to external storage systems should be carefully managed. This will lead to the risk of losing non-persistent data if a container fails or restarts unexpectedly. Lastly, security is a point to consider. Using outdated Docker images or incorrectly configuring access controls can expose your data to risks by causing Docker to fail and take all containers out of service at once

J. **Why is this solution important (be as detailed as possible)?**

This solution is essential because **TML** is the only technology that can do entity-based machine learning, in-memory, on real-time data while integrating seamlessly with Apache Kafka. If you need to process real-time data, **you need TML.** Its versatility makes it valuable in any industry around the world. Combined with Docker, it ensures smooth deployment, scalability, and efficient management, helping organizations stay ahead with real-time insights and smarter decision-making.

K. **How can this solution be extended to other business/industry areas (apart from cybersecurity and networking)?**

This solution's real-time data processing and machine learning capabilities make it adaptable across industries. In healthcare, it predicts diseases and optimizes resources. Finance benefits from fraud detection and personalized services, while retail and logistics enhance inventory, pricing, and tracking. Manufacturing improves quality control and predictive maintenance, and energy sectors optimize grids and renewable resources. In agriculture, it enables precision farming, while media and automotive sectors use it for personalization and smart systems. Its versatility drives efficiency, innovation, and smarter decision-making across all industries.

L. **What is the importance of artificial intelligence and machine learning?**

Artificial intelligence (AI) and machine learning (ML) are crucial because they enable real-time data processing, automate complex tasks, and drive data-driven decision-making. They enhance personalization, scalability, and efficiency across industries while solving complex problems like climate modeling and disease prediction. AI and ML fuel innovation, provide a competitive edge, and complement human capabilities, making them essential for shaping a smarter, more efficient future.

M. **How would you scale this solution (be as detailed as possible)?**

To scale this solution, the focus should be on cloud deployment and orchestration using AWS, Azure, or Google Cloud. Use Kubernetes to handle container orchestration, making sure fault tolerance and efficient scaling across distributed environments are guaranteed. Apply auto-scaling techniques for real-time handling of fluctuating workloads and employ load balancing for consistent performance. Integrate with a wider range of third-party tools and enterprise CRM and ERP systems for wider adoption. Enhance security frameworks by adding automated compliance checks and vulnerability scanning. Finally, leverage edge computing to process data closer to the source for latency-sensitive applications, ensuring scalability across diverse use cases like IoT and smart cities.

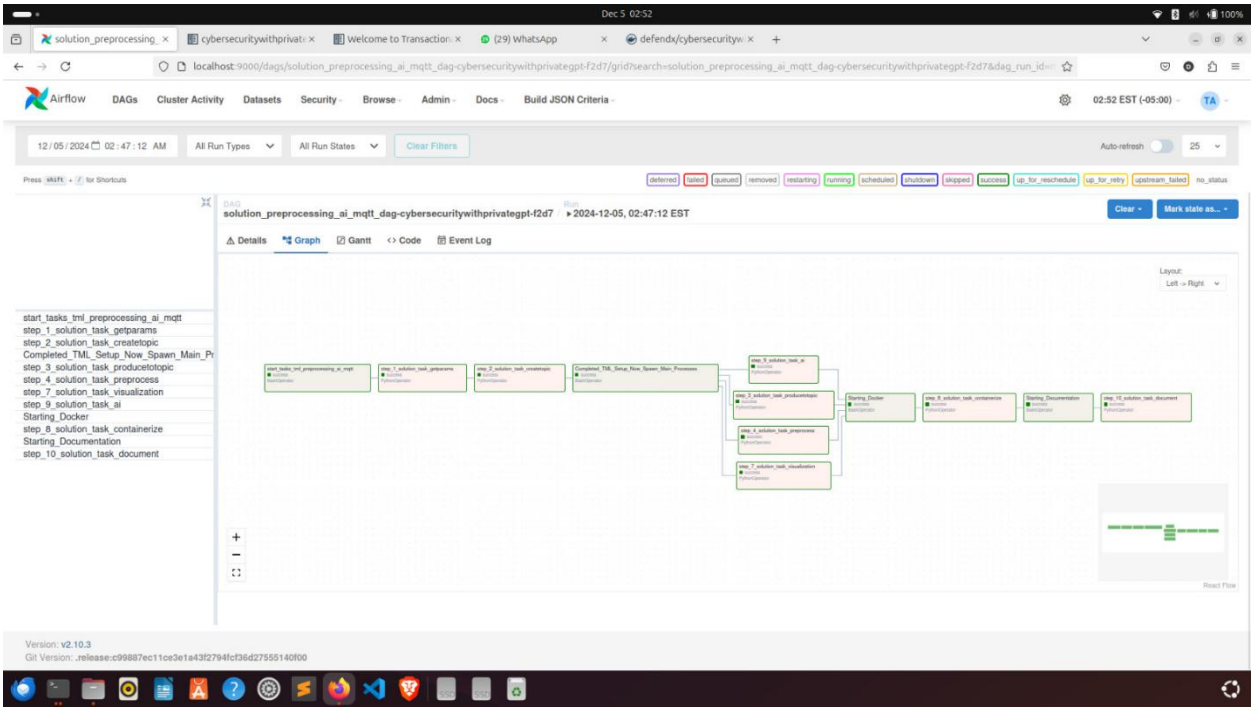
N. **What are your concluding thoughts about this solution (be as detailed as possible)?**

This solution represents a huge advancement in real-time data processing and machine learning technology, unique in the capabilities provided. Its fast deployment, cost efficiency, and low-code/no-code interface make it accessible for a broad user base, from novice developers to experienced data scientists. By using only in-memory entity-based machine learning and JSON processing instead of traditional database-reliant methods, the solution achieves superior speed, efficiency, and scalability. This makes it even more attractive, with integrations to state-of-the-art GenAI tools that unlock advanced AI-driven insights and automation. All these features make it particularly well-suited for industries where real-time decision-making and predictive analytics are critical, such as finance, healthcare, and manufacturing.

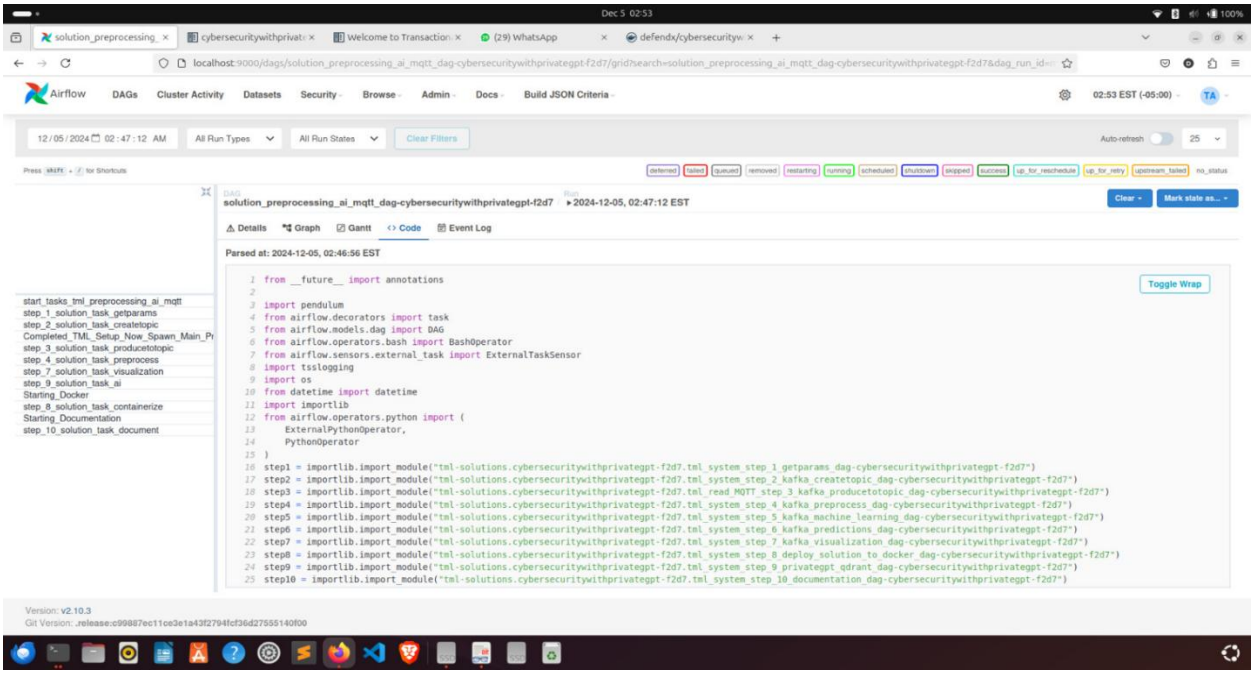
However, it is not without its own set of challenges. The solution requires users to have or acquire skills in managing containerized environments, understanding TML syntax, and configuring external monitoring systems like Prometheus or Grafana. Besides that,

potential performance bottlenecks with overconsumption of containers and security risks due to poorly configured Docker environments have to be proactively prevented. These can be minimized or reduced with proper training, robust documentation, and improved tooling for monitoring and securing deployments.

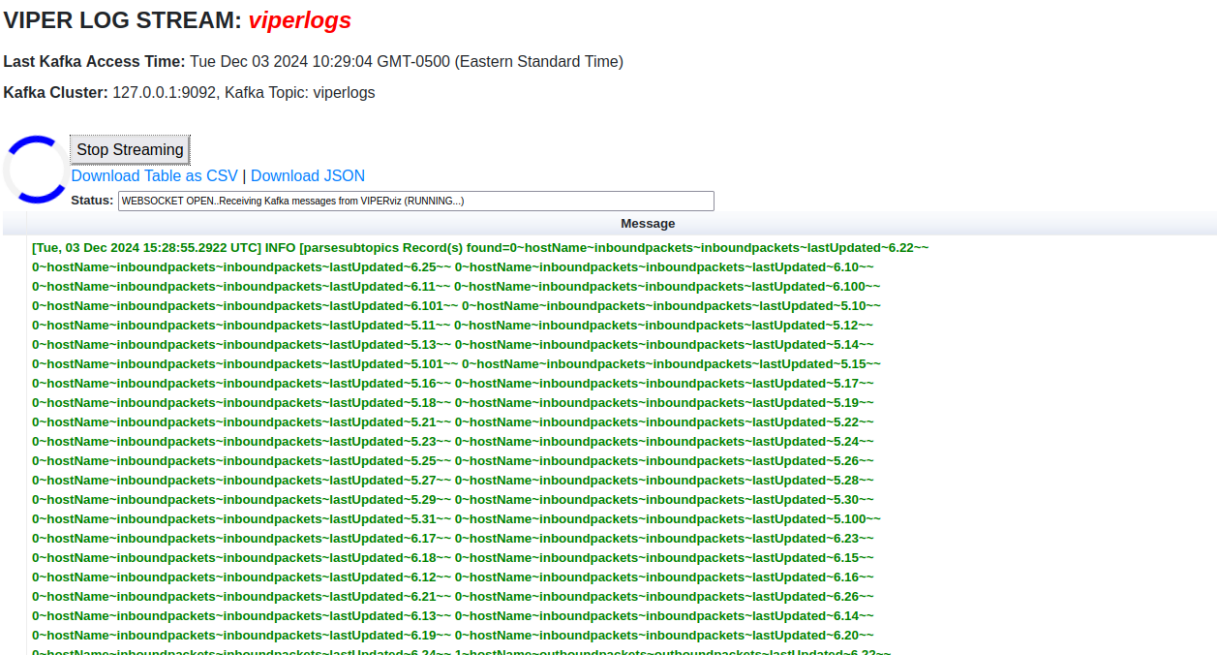
Appendix A: Example of Screenshot of Solution containing the TML Dags



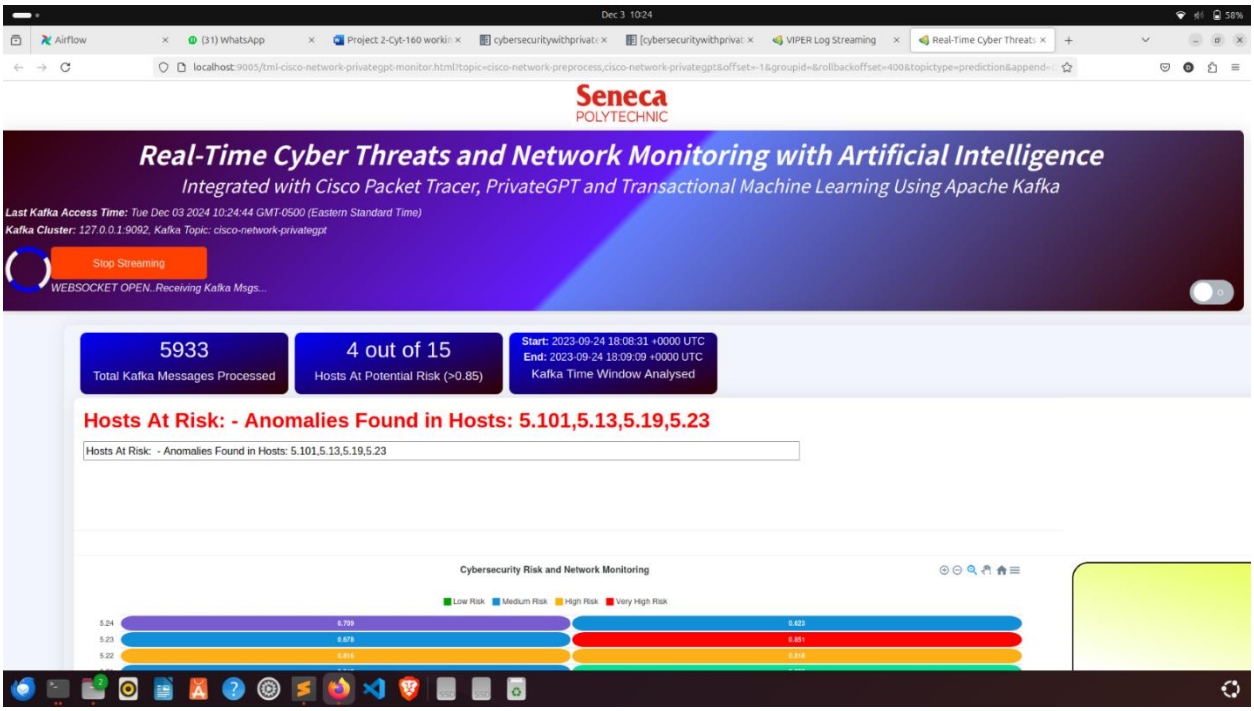
Appendix B: Copy of the Solution Code with Run Status



Appendix C: Copy of TML Viper logs



Appendix D: Solution Dashboard View



Links:

read the docs - https://cybersecuritywithprivategpt-f2d7.readthedocs.io/en/latest/ ,

DockerHub- https://hub.docker.com/repository/docker/defendx/cybersecuritywithprivategpt-f2d7-amd64/general,

Github - https://github.com/Defendx3/raspberrypi.git,

References

TML Solution Components — Transactional Machine Learning (TML) 0.1 documentation. (n.d.). https://tml.readthedocs.io/en/latest/usage.html

For additional information on APA Style formatting, please consult the APA Style Manual, 7th Edition.