Insider Threat Detection using Splunk MLTK

1. Objective

This project focuses on detecting insider threats based on behavioral drift using machine learning in Splunk MLTK.

The aim is to identify anomalous user behavior that could indicate a potential security risk.

2. Tools & Technologies

- Splunk Enterprise (with Machine Learning Toolkit - MLTK)
- Python (for generating synthetic datasets)
- SPL (Search Processing Language)
- GitHub for version control and submission

3. Dataset

Synthetic dataset containing user activities such as login time, activity type, volume, location, and status.

Fields used for training include: activity_count, avg_volume, hour_of_day.

4. Data Ingestion

1. Created custom index in Splunk (e.g., insider_index)
2. Uploaded CSV file via Settings > Add Data
3. Verified upload by searching in the index.

5. Feature Engineering

SPL used to extract features:

```
index=insider_index
| eval hour_of_day = strftime(strptime(timestamp, "%Y-%m-%d %H:%M:%S"), "%H")
| stats count AS activity_count avg(volume) AS avg_volume BY username, activity_type, hour_of_day, location, status
```

6. Model Training

Three separate unsupervised models trained using DensityFunction algorithm on:

- activity_count

- avg_volume

- hour_of_day

Each model was trained with a threshold = 0.0001 using the MLTK Experiment interface.

7. Model Application

Each model was applied using SPL with the 'apply' command. Outlier scores were renamed and evaluated:

| apply "Threat Drift Detection"

| rename ...

The total threat score was computed by summing the outlier indicators.

8. Threat Classification

Threat levels were assigned based on total score:

- High (>=2 anomalies)

- Medium (=1 anomaly)

- Normal (=0 anomalies)

This logic was implemented in SPL using `eval` and `case` functions.

9. Dashboard

A simple dashboard was created showing high threat users with columns:

username, activity_type, location, status, total_score, threat_level

10. GitHub Repository Structure

- /models: Exported MLTK models (CSV)

- /data: Training & test datasets

- /dashboards: XML or JSON files for UI

- README.md: Complete usage instructions

- screenshots/: Optional UI images

11. Result & Next Steps

The prototype detects behavioral anomalies in real-time.

Future Scope:

- Add supervised models using historical incidents.

- Integrate alerting systems or automated blocking.

- Visualize drift trends with advanced dashboards.