Давайте разберемся, как визуализировать данные из **.log** файлов, сгенерированных Calnex CAT или аналогичными инструментами (например, **Open Time Instrument**), используя Python и Grafana .

**1. Визуализация на Python**

**Шаг 1: Подготовка данных**

Допустим, данные в **.log** имеют формат:

Копировать

<timestamp> <tie\_value>

Пример:

1628073600.123456 0.000001234

1628073601.234567 0.000002345

**Шаг 2: Установка зависимостей**

bash

Копировать

pip install pandas matplotlib plotly

**Шаг 3: Скрипт для визуализации**

python

⌄

# Для данных Clientgen (пример)

columns = ['timestamp', 'latency', 'packet\_type']

df = pd.read\_csv('clientgen.log', sep=' ', names=columns)

df['datetime'] = pd.to\_datetime(df['timestamp'], unit='s')

# График задержек

plt.figure(figsize=(12, 6))

for packet\_type in df['packet\_type'].unique():

subset = df[df['packet\_type'] == packet\_type]

plt.plot(subset['datetime'], subset['latency'], label=packet\_type)

plt.legend()

plt.title('Packet Latency Over Time')

plt.xlabel('Time')

plt.ylabel('Latency (ms)')

plt.grid(True)

plt.show()

**2. Визуализация в Grafana**

**Шаг 1: Установка инструментов**

1. InfluxDB (база данных для временных рядов):

bash

Копировать

1

docker run -d -p 8086:8086 influxdb

1. Grafana :

bash

Копировать

1

docker run -d -p 3000:3000 grafana/grafana

**Шаг 2: Импорт данных в InfluxDB**

1. Создайте базу данных в InfluxDB:

1

curl -G http://localhost:8086/query --data-urlencode "q=CREATE DATABASE mydb"

1. Запишите данные из **.log** в InfluxDB:

python

from influxdb import InfluxDBClient

client = InfluxDBClient(host='localhost', port=8086)

client.switch\_database('mydb')

# Преобразование данных

data = []

for \_, row in df.iterrows():

data.append({

"measurement": "tie\_data",

"time": row['datetime'],

"fields": {

"tie": row['tie']

}

})

client.write\_points(data)

**Шаг 3: Настройка Grafana**

1. Добавьте источник данных :
   * URL: **http://localhost:8086**
   * База данных: **mydb**.
2. Создайте дашборд :
   * Добавьте график с запросом:

sql

SELECT "tie" FROM "tie\_data" WHERE $timeFilter

* + Настройте оси и форматирование.

1. Пример дашборда :
   * График TIE с временной меткой.
   * Гистограмма распределения задержек.
   * Счетчики пакетов (если есть данные из Clientgen).

**3. Автоматизация**

Для автоматической записи данных в InfluxDB при генерации логов:

python

Копировать

⌄

# В скрипте, который генерирует логи (например, Open Time Instrument)

def log\_to\_influx(timestamp, tie\_value):

json\_body = [

{

"measurement": "tie\_data",

"time": timestamp,

"fields": {

"tie": tie\_value

}

}

]

client.write\_points(json\_body)

**4. Примеры визуализаций**

**Python (Matplotlib/Plotly)**

**Grafana**

**5. Рекомендации**

* Для реального времени используйте Grafana + InfluxDB.
* Для быстрого анализа — Python + Jupyter Notebook.
* Если данные содержат PTP-пакеты, добавьте их в визуализацию (например, задержки Sync/DelayReq).

Если уточните структуру ваших **.log** файлов, можно адаптировать скрипты под конкретные данные!