# Cahier des Charges Détaillé

## Projet : ****Système Big Data de Surveillance et d’Analyse de Logs Réseau pour la Cybersécurité****

## 1. Introduction

### 1.1 Contexte

Face à l’évolution des cybermenaces et à la croissance exponentielle du trafic réseau, les organisations doivent disposer d’un système capable de **capturer, stocker, analyser et visualiser** les événements réseau en **temps réel**.

Le projet propose la mise en place d’un **pipeline Big Data complet** pour la **surveillance, l’analyse et la détection d’anomalies réseau** à partir des données capturées en direct.

### 1.2 Objectifs

* Capturer le trafic réseau brut via le mode promiscuous.
* Convertir les paquets capturés (PCAP) en **données de flux analytiques (NetFlow/CSV)**.
* Intégrer les données collectées dans un pipeline Big Data (Kafka, Spark, Cassandra, Elasticsearch).
* Appliquer des traitements analytiques et de Machine Learning pour détecter les comportements anormaux.
* Visualiser les résultats et alertes via **Power BI**.

## 2. Architecture Globale du Système

### 2.1 Vue d’ensemble

[Trafic Réseau Réel]

↓

Mode Promisc / tcpdump (PCAP)

↓

nfpcapd / nfdump / CICFlowMeter

↓ CSV

Apache Kafka

↓

Apache Spark Streaming

↓ ↓

Cassandra Elasticsearch

↓ ↓

Data Lake Kibana / Power BI

### 2.2 Composants Principaux

| **Composant** | **Rôle** |
| --- | --- |
| **Interface Promisc** | Capture des paquets réseau circulant sur le réseau. |
| **tcpdump** | Enregistrement continu des paquets au format .pcap. |
| **nfpcapd / nfdump / CICFlowMeter** | Conversion des fichiers PCAP en flux NetFlow et extraction des statistiques CSV. |
| **Apache Kafka** | Transmission asynchrone et scalable des flux de logs vers Spark. |
| **Apache Spark** | Traitement temps réel et batch, analyse statistique et apprentissage automatique. |
| **Apache Cassandra** | Stockage distribué haute disponibilité des flux traités. |
| **Elasticsearch + Kibana** | Indexation rapide et visualisation en temps réel. |
| **Power BI** | Tableaux de bord analytiques et rapports décisionnels. |

## 3. Environnement Technique

| **Élément** | **Technologie** |
| --- | --- |
| Capture | tcpdump, nfpcapd, nfdump, CICFlowMeter |
| Traitement | Apache Spark, PySpark, Kafka |
| Stockage | Apache Cassandra |
| Indexation | Elasticsearch |
| Visualisation | Kibana, Power BI |
| Langage | Python / Scala |
| OS | Linux (Ubuntu/Debian) |

## 4. Étapes Techniques Détaillées

### 4.1 Capture du trafic réseau

#### Activation du mode promiscuité :

sudo ifconfig eth0 promisc

#### Mode monitor pour carte Wi-Fi :

sudo ip link set wlan0 down

sudo iw wlan0 set monitor control

sudo ip link set wlan0 up

#### Capture et rotation automatique :

sudo tcpdump -i eth0 -s 0 -W 10 -C 100 -w /var/log/pcap/capture-%Y%m%d%H%M.pcap

### 4.2 Conversion PCAP → NetFlow / CSV

#### Extraction via nfpcapd :

nfpcapd -r /tmp/capture.pcap -l /var/log/netflow/

#### Export CSV :

nfdump -r /var/log/netflow/nfcapd.YYYYMMDDHHMM -o csv > /tmp/flows.csv

#### Extraction des features avancées :

java -jar CICFlowMeter.jar -r /tmp/capture.pcap -f /tmp/flows.csv

### 4.3 Ingestion via Apache Kafka

Chaque CSV est lu et publié sur un topic Kafka (netflow-data) :

from kafka import KafkaProducer

import pandas as pd, json

producer = KafkaProducer(bootstrap\_servers=['localhost:9092'])

df = pd.read\_csv('/tmp/flows.csv')

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

producer.send('netflow-data', value=row.to\_json().encode('utf-8'))

### 4.4 Traitement en Temps Réel avec Apache Spark

from pyspark.sql import SparkSession

from pyspark.sql.functions import \*

spark = SparkSession.builder \

.appName("NetFlowPipeline") \

.config("spark.cassandra.connection.host", "127.0.0.1") \

.getOrCreate()

df = spark.readStream \

.format("kafka") \

.option("kafka.bootstrap.servers", "localhost:9092") \

.option("subscribe", "netflow-data") \

.load()

cleaned = df.selectExpr("CAST(value AS STRING)")

query = cleaned.writeStream \

.format("org.apache.spark.sql.cassandra") \

.options(table="flows", keyspace="netflow") \

.outputMode("append") \

.start()

query.awaitTermination()

### 4.5 Stockage et Indexation

#### Cassandra

Structure de table :

CREATE TABLE netflow.flows (

id UUID PRIMARY KEY,

src\_ip text,

dest\_ip text,

src\_port int,

dest\_port int,

protocol int,

bytes\_in bigint,

bytes\_out bigint,

num\_pkts\_in bigint,

num\_pkts\_out bigint,

start\_time timestamp,

end\_time timestamp

);

#### Elasticsearch

Connexion via Spark :

df.writeStream \

.format("es") \

.option("es.nodes", "localhost") \

.option("es.resource", "netflow/\_doc") \

.outputMode("append") \

.start()

### 4.6 Visualisation avec Power BI

* Connexion directe à Elasticsearch via le connecteur officiel.
* Dashboards dynamiques :
  + 🔹 Volume de trafic par protocole
  + 🔹 Top IP sources/destinations
  + 🔹 Activité anormale détectée par Spark ML
  + 🔹 Statistiques temporelles (trafic horaire, pics, incidents)

## 5. Spécifications Fonctionnelles

| **Module** | **Fonctionnalités clés** |
| --- | --- |
| **Capture** | Mode promisc, rotation PCAP, export NetFlow |
| **Kafka** | Collecte des logs en streaming |
| **Spark** | Nettoyage, transformation, ML |
| **Cassandra** | Stockage distribué des données de flux |
| **Elasticsearch** | Indexation et recherche |
| **Power BI** | Visualisation et alertes décisionnelles |

## 6. Spécifications Non Fonctionnelles

| **Critère** | **Exigence** |
| --- | --- |
| Performance | ≥ 10 000 paquets/s |
| Scalabilité | Cluster Kafka/Spark/Cassandra extensible |
| Sécurité | Authentification, TLS, RBAC Power BI |
| Disponibilité | 24/7 avec reprise automatique |
| Interopérabilité | Formats PCAP, NetFlow, CSV, JSON |
| Monitoring | Logs centralisés, alertes automatiques |

## 7. Cas d’Utilisation

| **Acteur** | **Cas d’utilisation** | **Description** |
| --- | --- | --- |
| **Administrateur Réseau** | Lancer capture réseau | Activation manuelle de tcpdump ou script automatique |
| **Système** | Transformation PCAP → CSV | Conversion périodique via nfpcapd |
| **Pipeline Kafka/Spark** | Ingestion & traitement temps réel | Nettoyage et analyse de flux réseau |
| **Analyste SOC** | Visualiser et filtrer alertes | Power BI et Kibana |
| **Moteur ML** | Détection d’anomalies | Algorithmes Spark MLlib |

## 8. Livrables

1. Scripts Bash pour capture et conversion PCAP/NetFlow.
2. Pipeline Kafka & Spark Streaming complet.
3. Base de données Cassandra structurée.
4. Index Elasticsearch configuré.
5. Dashboard Power BI et Kibana.
6. Documentation technique et guide d’installation.

## 9. Conclusion

Ce système combine **l’ingénierie réseau (tcpdump, nfpcapd, NetFlow)** avec la **puissance du Big Data (Kafka, Spark, Cassandra, Elasticsearch)**.  
L’ajout de **Power BI** permet une **visualisation décisionnelle complète**, facilitant la **détection proactive des anomalies réseau** et la **surveillance continue** de la cybersécurité.