

ANALYSE DES DONNÉES DE RÉSEAU



Elasticsearch



Logstash



Kibana

Présenté par:

Wissal Boutayeb
Fatima Bouyarmane

Encadré par:

M. AIRAJ Mohammed

SOMMAIRE

Introduction	3
Problématique	4
objectifs	5
Architecture de la Solution	7
Étapes de Réalisation	9
Capture et analyse de trafic réseau	10
Défis Rencontrés	11
Conclusion	12

01

INTRODUCTION

Aujourd'hui, avec la croissance exponentielle des réseaux et l'augmentation des cybermenaces, il devient essentiel pour les organisations de surveiller et analyser leur trafic réseau. Cette tâche, cependant, est complexe en raison du volume massif de données générées quotidiennement.

C'est dans ce contexte que notre projet s'inscrit. Il exploite la puissance de la stack ELK (Elasticsearch, Logstash, Kibana), une solution moderne et robuste permettant de collecter, traiter et visualiser les données réseau en temps réel.

PROBLÉMATIQUE

Avec l'explosion des volumes de données générées par les réseaux informatiques, il devient de plus en plus difficile pour les entreprises de surveiller efficacement leur trafic, détecter les anomalies et répondre rapidement aux menaces de sécurité. Les outils classiques d'analyse sont souvent limités en termes de performances et de visualisation, rendant la prise de décision lente et peu efficace.

Comment concevoir une solution performante et accessible pour capturer, analyser et visualiser en temps réel le trafic réseau afin de renforcer la sécurité et optimiser les performances du système ?

03

Objectifs du Projet



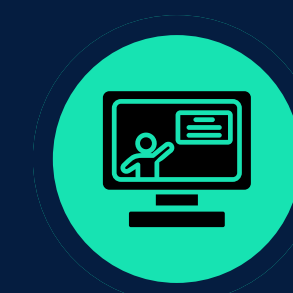
Installation et configuration de la pile ELK (Elasticsearch, Logstash, Kibana)



Capture et traitement des données réseau.



Automatisation l'indexation et l'analyse des données avec un script Python.



Création des tableaux de bord interactifs avec Kibana pour visualiser

- **Distribution des protocoles.**
- **IP sources principales.**
- **Chronologie du trafic réseau.**

09

ARCHITECTURE DE ELK



04

ÉTAPES DE RÉALISATION

01

Préparation de l'environnement.

02

Configuration de la stack ELK (Elasticsearch, Logstash, Kibana, Filebeat)

03

Capture et transformation des données.

04

Création de visualisations interactives dans Kibana.

Configuration de Elasticsearch

```
(kali@kali)-[~]  
$ sudo bash -c 'cat > /etc/elasticsearch/elasticsearch.yml << EOL  
cluster.name: kali-cluster  
node.name: node-1  
path.data: /var/lib/elasticsearch  
path.logs: /var/log/elasticsearch  
network.host: localhost  
http.port: 9200  
discovery.type: single-node  
EOL'
```

```
(kali@kali)-[~]  
$ curl http://localhost:9200  
{  
  "name" : "node-1",  
  "cluster_name" : "kali-cluster",  
  "cluster_uuid" : "fJqWMnNUQWGtA_J3lXSYgg",  
  "version" : {  
    "number" : "7.17.14",  
    "build_flavor" : "default",  
    "build_type" : "deb",  
    "build_hash" : "774e3bfa4d52e2834e4d9d8d669d77e4e5c1017f",  
    "build_date" : "2023-10-05T22:17:33.780167078Z",  
    "build_snapshot" : false,  
    "lucene_version" : "8.11.1",  
    "minimum_wire_compatibility_version" : "6.8.0",  
    "minimum_index_compatibility_version" : "6.0.0-beta1"  
  },  
  "tagline" : "You Know, for Search"  
}
```


Configuration de Logstash

```
(kali@kali)-[~/Project_ELK]
└─$ sudo nano /etc/logstash/conf.d/network.conf
GNU nano 8.2 /etc/logstash/conf.d/network.conf
# /etc/logstash/conf.d/network.conf
input {
  file {
    path => "/home/kali/Project_ELK/network_analysis/network_data.json"
    start_position => "beginning"
    sincedb_path => "/dev/null"
    codec => json
    type => "pcap"/Project_ELK
  }
}

filter {
  if [type] == "pcap" {
    date {
      match => [ "timestamp", "%Y-%m-%d %H:%M:%S" ]
      target => "@timestamp"
    }
    geoip {
      source => "source_ip"
      target => "source_location"
    }
    # Enrichissement supplémentaire
    mutate {
      add_field => {
        "processed_by" => "logstash"
        "analysis_type" => "enhanced"
      }
    }
  }
}

output {
  elasticsearch {
    hosts => ["localhost:9200"]
    index => "network-analysis-enriched-%{+YYYY.MM.dd}"
  }
  stdout { codec => rubydebug }
}
```

Configuration de Kibana

```
(kali㉿kali)-[~]
└─$ sudo apt --fix-broken install
```

Summary:

```
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 34
```

```
└─(kali@kali)-[~]
```

```
$ sudo bash -c 'cat > /etc/kibana/kibana.yml << EOF
```

```
server.port: 5601
```

```
server.host: "localhost"
```

```
elasticsearch.hosts: ["http://localhost:9200"]
```

EOL™

```
(kali㉿kali)-[/etc/logstash/conf.d] Kali Forums  Kali NetHunter  Exploit-DB  Google Hacking DB  OffSec
$ curl http://localhost:5601/app/home
```

```
<!DOCTYPE html><html lang="en"><head><meta charset="utf-8"/><meta http-equiv="X-UA-Compatible" content="IE=edge"><meta name="viewport" content="width=device-width"/><title>Elastic</title><style>
```

```
@font-face {
  font-family: 'Inter';
  font-style: normal;
  font-weight: 100;
  src: url('/ui/fonts/inter/Inter-Thin.woff2') format('woff2'), url('/ui/fonts/inter/Inter-Thin.woff'
```

```
@font-face {
  font-family: 'Inter';
  font-style: italic;
  font-weight: 100;
  src: url('/ui/fonts/inter/Inter-ThinItalic.woff2') format('woff2'), url('/ui/fonts/inter/Inter-Thin');
}
```

Capture et analyse de trafic réseau

```
(kali㉿kali)-[~/Project_ELK]
$ sudo tcpdump -i eth0 -w capture.pcap -G 60 -W 1
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C29 packets captured
29 packets received by filter
0 packets dropped by kernel
```

```
(kali@kali)-[~]
$ nmap localhost
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-12 11:51 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000070s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
9200/tcp  open  wap-wsp logstash

Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds

(kali@kali)-[~]
$ ping google.com
curl example.com
PING google.com (142.250.184.174) 56(84) bytes of data:
64 bytes from mad07s23-in-f14.1e100.net (142.250.184.174): icmp_seq=1 ttl=128 time=19.6 ms
64 bytes from mad07s23-in-f14.1e100.net (142.250.184.174): icmp_seq=2 ttl=128 time=20.6 ms
64 bytes from mad07s23-in-f14.1e100.net (142.250.184.174): icmp_seq=3 ttl=128 time=19.1 ms
64 bytes from mad07s23-in-f14.1e100.net (142.250.184.174): icmp_seq=4 ttl=128 time=19.0 ms
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 19.041/19.594/20.631/0.640 ms
<!doctype html>
<html>
<head>
<title>Example Domain</title>

<meta charset="utf-8" />
<meta http-equiv="Content-type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<style type="text/css">
body {
background-color: #f0f0f2;
margin: 0;
padding: 0;
font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;
}
div {
width: 600px;
margin: 5em auto;
padding: 2em;

```

Capture et analyse de trafic réseau

```
GNU nano 8.2 analyze_pcap.py *
from elasticsearch import Elasticsearch
import pyshark
from datetime import datetime, timezone

def analyze_pcap(pcap_file):
    es = Elasticsearch(['http://localhost:9200'])
    capture = pyshark.FileCapture(pcap_file)
    for packet in capture:
        try:
            current_time = datetime.now(timezone.utc)
            packet_data = {
                'timestamp': current_time.isoformat(),
                'protocol': str(packet.highest_layer),
                'length': int(packet.length)
            }

            if hasattr(packet, 'ip'):
                packet_data.update({
                    'source_ip': str(packet.ip.src),
                    'dest_ip': str(packet.ip.dst)
                })

            if hasattr(packet, 'tcp'):
                packet_data.update({
                    'source_port': int(packet.tcp.srcport),
                    'dest_port': int(packet.tcp.dstport)
                })

            es.index(index='network-analysis', document=packet_data)
            print(f"Paquet indexé: {packet.highest_layer}")

        except Exception as e:
            print(f"Erreur: {e}")
            continue

    capture.close()

if __name__ == "__main__":
    analyze_pcap('capture.pcap')
```


Capture et analyse de trafic réseau

```
(myenv)-(kali@kali)-[~/Project_ELK/network_analysis]
$ python3 analyze_pcap.py
/home/kali/Project_ELK/network_analysis/analyze_pcap.py:31: ElasticsearchWarning: Elasticsearch built-in security features are not enabled
See https://www.elastic.co/guide/en/elasticsearch/reference/7.17/security-minimal-setup.html to enable security.
  es.index(index='network-analysis', document=packet_data)
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: ICMP
Paquet indexé: ICMP
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: ICMP
Paquet indexé: ICMP
Paquet indexé: ICMP
Paquet indexé: ICMP
Paquet indexé: ICMP
Paquet indexé: ICMP
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: DNS
Paquet indexé: TCP
Paquet indexé: TCP
Paquet indexé: TCP
Paquet indexé: HTTP
Paquet indexé: TCP
Paquet indexé: DATA-TEXT-LINES
Paquet indexé: TCP
Paquet indexé: TCP
Paquet indexé: TCP
Paquet indexé: TCP
Paquet indexé: TCP
```

Capture et analyse de trafic réseau

```
(myenv)-(kali@kali)-[~/Project_ELK/network_analysis]
$ cat network_data.json
{"timestamp": "2024-11-12T12:39:12.838397", "protocol": "DNS", "length": 70, "source_ip": "192.168.145.128", "dest_ip": "192.168.145.2"}
{"timestamp": "2024-11-12T12:39:12.840613", "protocol": "DNS", "length": 70, "source_ip": "192.168.145.128", "dest_ip": "192.168.145.2"}
{"timestamp": "2024-11-12T12:39:12.842126", "protocol": "DNS", "length": 98, "source_ip": "192.168.145.2", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.843370", "protocol": "DNS", "length": 86, "source_ip": "192.168.145.2", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.844190", "protocol": "ICMP", "length": 98, "source_ip": "192.168.145.128", "dest_ip": "142.250.184.174"}
{"timestamp": "2024-11-12T12:39:12.845010", "protocol": "ICMP", "length": 98, "source_ip": "142.250.184.174", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.845911", "protocol": "DNS", "length": 88, "source_ip": "192.168.145.128", "dest_ip": "192.168.145.2"}
{"timestamp": "2024-11-12T12:39:12.846810", "protocol": "DNS", "length": 127, "source_ip": "192.168.145.2", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.847567", "protocol": "ICMP", "length": 98, "source_ip": "192.168.145.128", "dest_ip": "142.250.184.174"}
{"timestamp": "2024-11-12T12:39:12.848230", "protocol": "ICMP", "length": 98, "source_ip": "142.250.184.174", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.849349", "protocol": "ICMP", "length": 98, "source_ip": "192.168.145.128", "dest_ip": "142.250.184.174"}
{"timestamp": "2024-11-12T12:39:12.850108", "protocol": "ICMP", "length": 98, "source_ip": "142.250.184.174", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.851302", "protocol": "ICMP", "length": 98, "source_ip": "192.168.145.128", "dest_ip": "142.250.184.174"}
{"timestamp": "2024-11-12T12:39:12.852793", "protocol": "ICMP", "length": 98, "source_ip": "142.250.184.174", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.854233", "protocol": "DNS", "length": 71, "source_ip": "192.168.145.128", "dest_ip": "192.168.145.2"}
{"timestamp": "2024-11-12T12:39:12.855355", "protocol": "DNS", "length": 71, "source_ip": "192.168.145.128", "dest_ip": "192.168.145.2"}
{"timestamp": "2024-11-12T12:39:12.856486", "protocol": "DNS", "length": 99, "source_ip": "192.168.145.2", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.857670", "protocol": "DNS", "length": 87, "source_ip": "192.168.145.2", "dest_ip": "192.168.145.128"}
{"timestamp": "2024-11-12T12:39:12.860459", "protocol": "TCP", "length": 74, "source_ip": "192.168.145.128", "dest_ip": "93.184.215.14", "source_port": 37408, "dest_port": 80}
{"timestamp": "2024-11-12T12:39:12.865836", "protocol": "TCP", "length": 60, "source_ip": "93.184.215.14", "dest_ip": "192.168.145.128", "source_port": 80, "dest_port": 37408}
{"timestamp": "2024-11-12T12:39:12.868755", "protocol": "TCP", "length": 54, "source_ip": "192.168.145.128", "dest_ip": "93.184.215.14", "source_port": 37408, "dest_port": 80}
{"timestamp": "2024-11-12T12:39:12.871318", "protocol": "HTTP", "length": 129, "source_ip": "192.168.145.128", "dest_ip": "93.184.215.14", "source_port": 37408, "dest_port": 80}
{"timestamp": "2024-11-12T12:39:12.874737", "protocol": "TCP", "length": 60, "source_ip": "93.184.215.14", "dest_ip": "192.168.145.128", "source_port": 80, "dest_port": 37408}
{"timestamp": "2024-11-12T12:39:12.880544", "protocol": "DATA-TEXT-LINES", "length": 1652, "source_ip": "93.184.215.14", "dest_ip": "192.168.145.128", "source_port": 80, "dest_port": 37408}
{"timestamp": "2024-11-12T12:39:12.883345", "protocol": "TCP", "length": 54, "source_ip": "192.168.145.128", "dest_ip": "93.184.215.14", "source_port": 37408, "dest_port": 80}
{"timestamp": "2024-11-12T12:39:12.886491", "protocol": "TCP", "length": 54, "source_ip": "192.168.145.128", "dest_ip": "93.184.215.14", "source_port": 37408, "dest_port": 80}
{"timestamp": "2024-11-12T12:39:12.888774", "protocol": "TCP", "length": 60, "source_ip": "93.184.215.14", "dest_ip": "192.168.145.128", "source_port": 80, "dest_port": 37408}
{"timestamp": "2024-11-12T12:39:12.891146", "protocol": "TCP", "length": 60, "source_ip": "93.184.215.14", "dest_ip": "192.168.145.128", "source_port": 80, "dest_port": 37408}
{"timestamp": "2024-11-12T12:39:12.893385", "protocol": "TCP", "length": 54, "source_ip": "192.168.145.128", "dest_ip": "93.184.215.14", "source_port": 37408, "dest_port": 80}
```

Création de visualisations interactives dans Kibana.

Create index pattern

Name

Use an asterisk (*) to match multiple characters. Spaces and the characters , / , ? , " , < , > , | are not allowed.

Timestamp field

Select a timestamp field for use with the global time filter.

[Show advanced settings](#)

✕ CloseCreate index pattern


✓ Your index pattern matches 1 source.


network-analysis	Index
------------------	-------


Rows per page: 10 ▾


Création de visualisations interactives dans Kibana.


New visualization

 **Lens**
Create visualizations with our drag and drop editor. Switch between visualization types at any time. *Recommended for most users.*


 **Maps**
Create and style maps with multiple layers and indices.


 **TSVB**
Perform advanced analysis of your time series data.

 **Custom visualization**
Use Vega to create new types of visualizations. *Requires knowledge of Vega syntax.*


 **Aggregation based**
Use our classic visualize library to create charts based on aggregations.
[Explore options →](#)


Tools


 **Text**
Add text and images to your dashboard.


 **Controls**
Add dropdown menus and range sliders to your


New visualization


 **Area**
Emphasize the data between an axis and a line.


 **Data table**
Display data in rows and columns.


 **Gauge**
Show the status of a metric.


 **Goal**
Track how a metric progresses to a goal.


 **Heat map**
Shade data in cells in a matrix.


 **Horizontal bar**
Present data in horizontal bars on an axis.


 **Line**
Display data as a series of points.

 **Metric**
Show a calculation as a single number.

 **Pie**
Compare data in proportion to a whole.

 **Tag cloud**

 **Timelion**

 **Vertical bar**

Visualize Library

Create visualization

Building a dashboard? Create and add your visualizations right from the [Dashboard application](#).

Search...

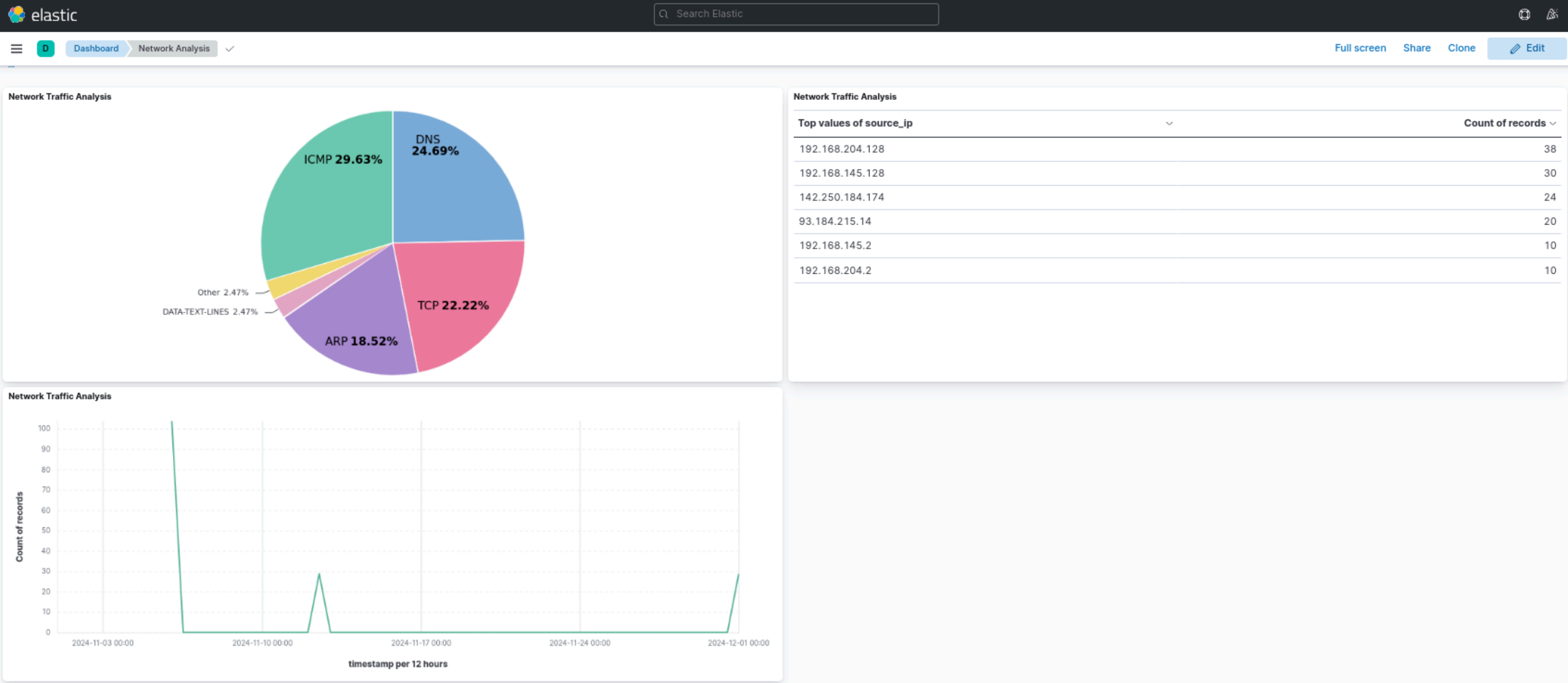
Tags

Title	Type	Description	Tags	Actions
Network Traffic Analysis	Lens			

Rows per page: 20

< 1 >

Création de visualisations interactives dans Kibana.



DÉFIS RENCONTRÉS

- Incompatibilités de Versions entre Elasticsearch, Logstash et Kibana
- Problèmes de Connexion sur le Port 9200 (Elasticsearch)
- Redémarrages Fréquents de la Service Elasticsearch

CONCLUSION

En conclusion, ce projet a permis de démontrer l'efficacité de la stack ELK pour l'analyse du trafic réseau en temps réel. Grâce à l'intégration de Elasticsearch, Logstash, Kibana, et Filebeat, nous avons pu collecter, traiter et visualiser les données réseau de manière fluide et performante. Bien que des défis techniques aient été rencontrés, comme les problèmes de compatibilité et de ressources, les solutions apportées ont permis d'optimiser le système. Ce projet montre l'importance des outils modernes pour une gestion proactive des réseaux, offrant ainsi une meilleure visibilité et une détection rapide des anomalies pour renforcer la sécurité et la gestion des infrastructures.

MERCI DE VOTRE ATTENTION!