МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий (ИКБ)

КБ-2 «Информационно-аналитические системы кибербезопасности»

ОТЧЕТ О ВЫПОЛНЕНИИ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ №4

В РАМКАХ ДИСЦИПЛИНЫ «ТЕХНОЛОГИИ ХРАНЕНИЯ В СИСТЕМАХ КИБЕРБЕЗОПАСНОСТИ»

Выполнил:

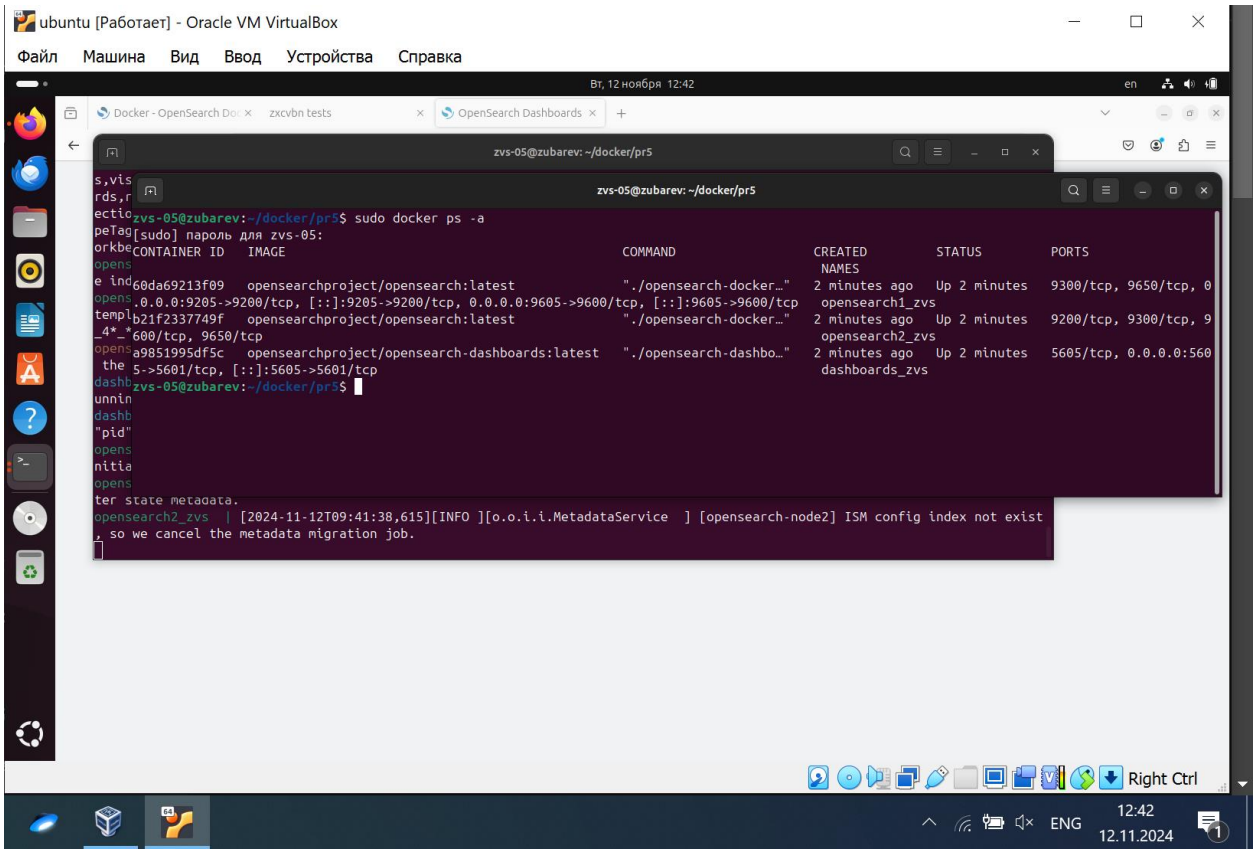
Студент 3-ого курса

Учебной группы БИСО-02-22

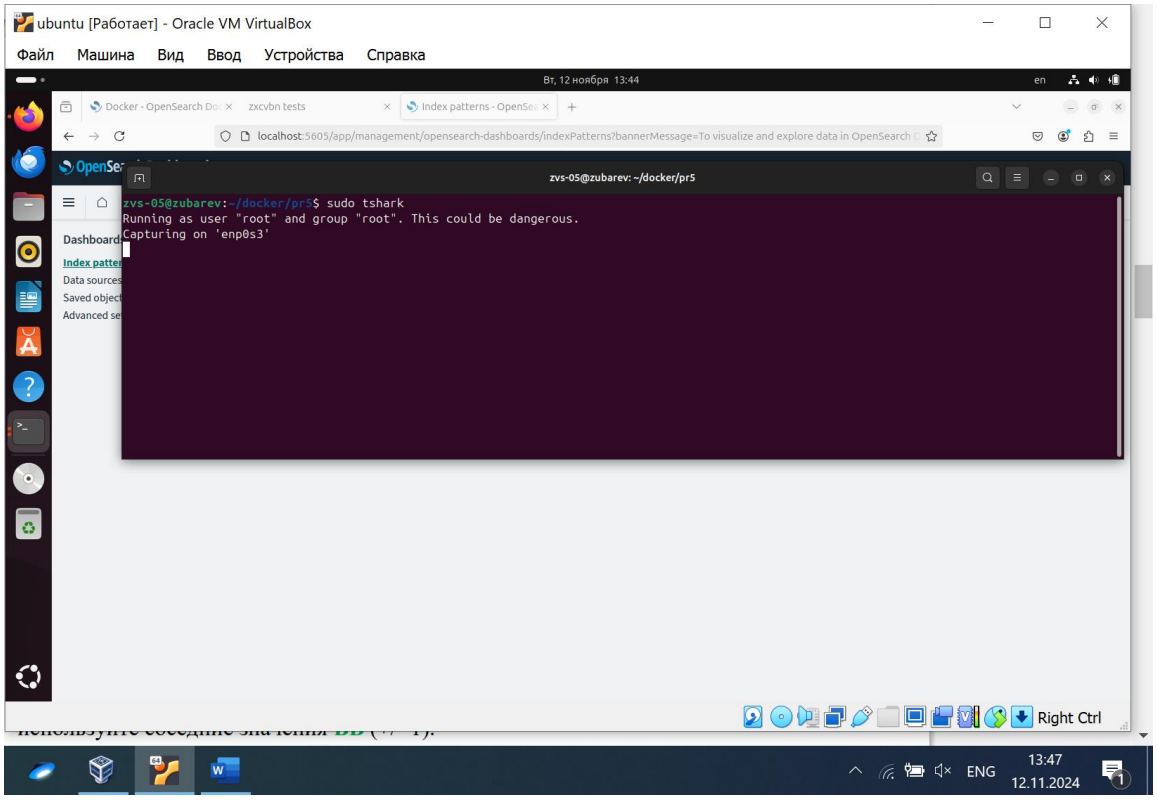
Зубарев В.С.

Москва 2024

Разверните OpenSearch и OpenSearch Dashboards с помощью Docker Compose



Установите ПО TShark (инструмент для сетевого анализа пакетов):

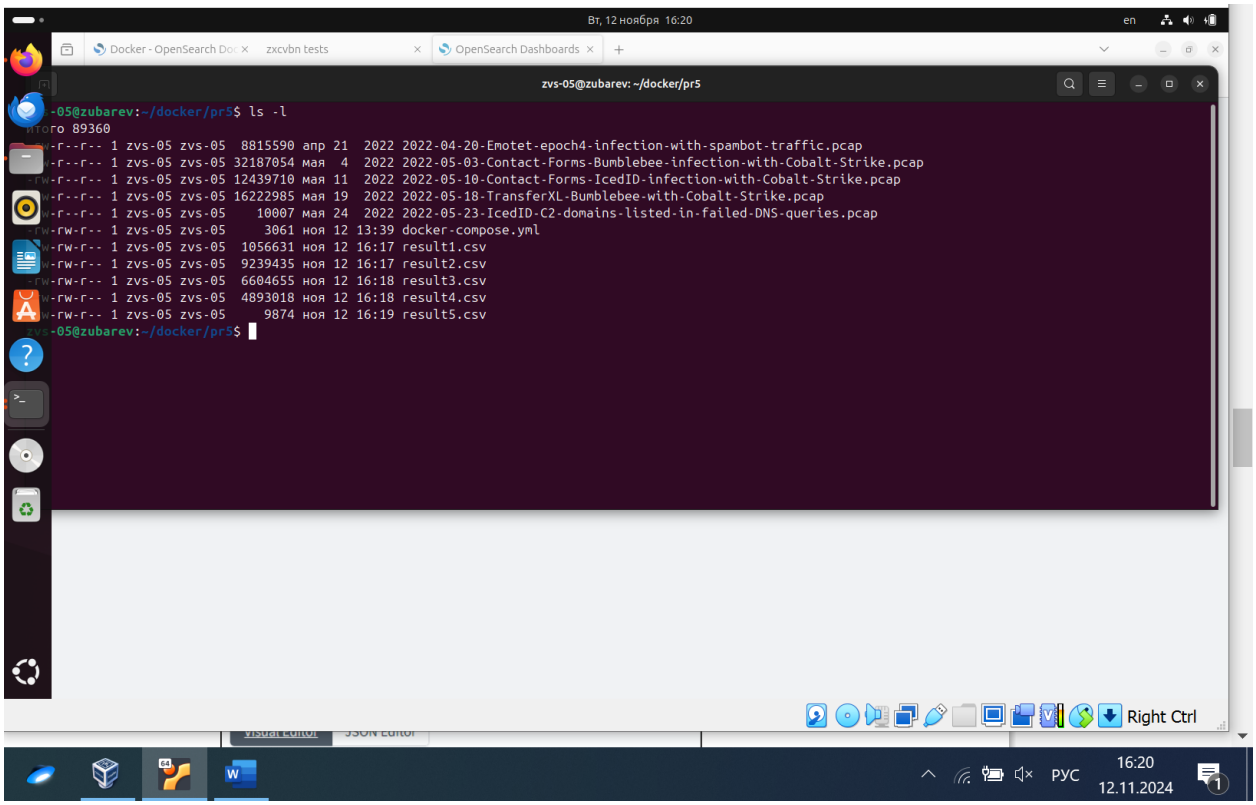


Соберите статистические данные от скачанных PCAP-файлов с помощью TShark.

Я добавил для поля tcp.stream (для связи пакетов в одной tcp-сессии) и tcp.falgs (для идентификации пакетов, что бы можно было понять кое действие совершается)

Команда(для первого файла)

sudo tshark -r 2022-04-20-Emotet-epoch4-infection-with-spambot-traffic.pcap -T fields -e frame.time_epoch -e ip.src -e ip.dst -e frame.protocols -e http.host -e tcp.srcport -e tcp.dstport -e udp.srcport -e udp.dstport -e tls.handshake.extensions_server_name -e frame.len -e tcp.stream -e tcp.flags -E header=y -E separator=\; -E aggregator=, > result1.csv

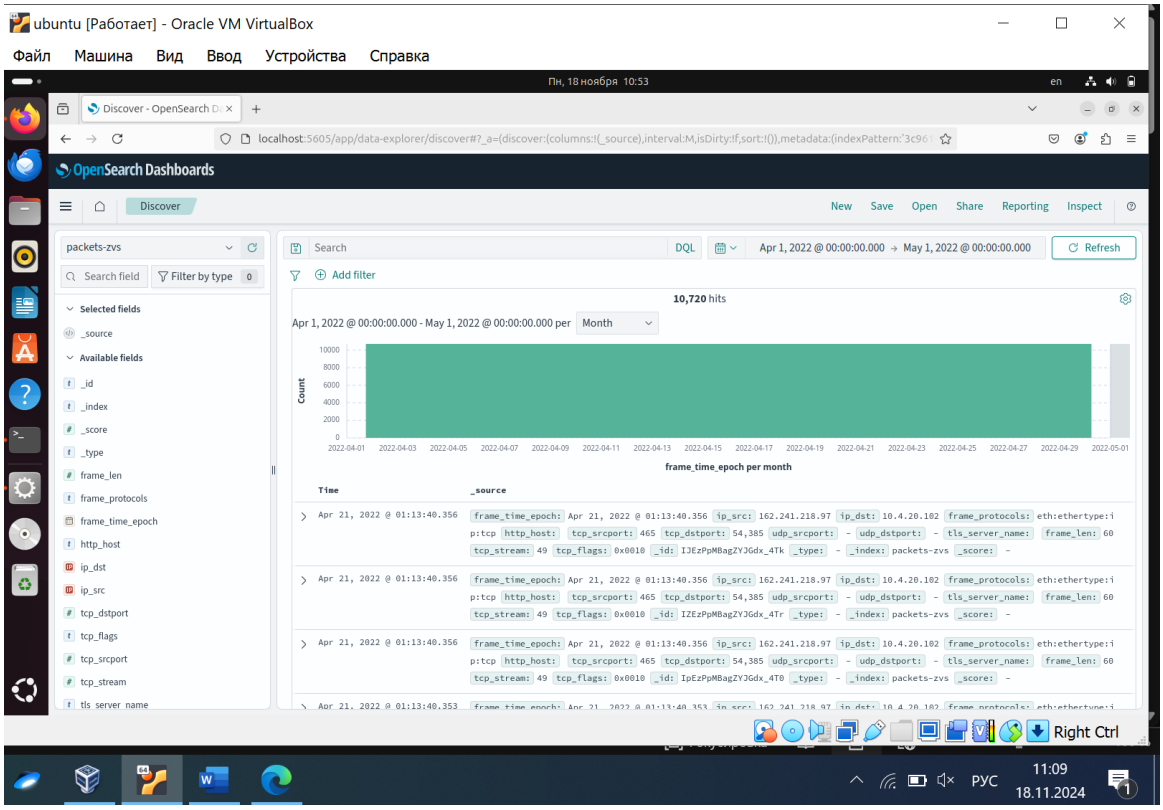


Загрузите полученные статистические данные в OpenSearch любым удобным способом. Ниже представлен пример на языке Python.

Листинг программы

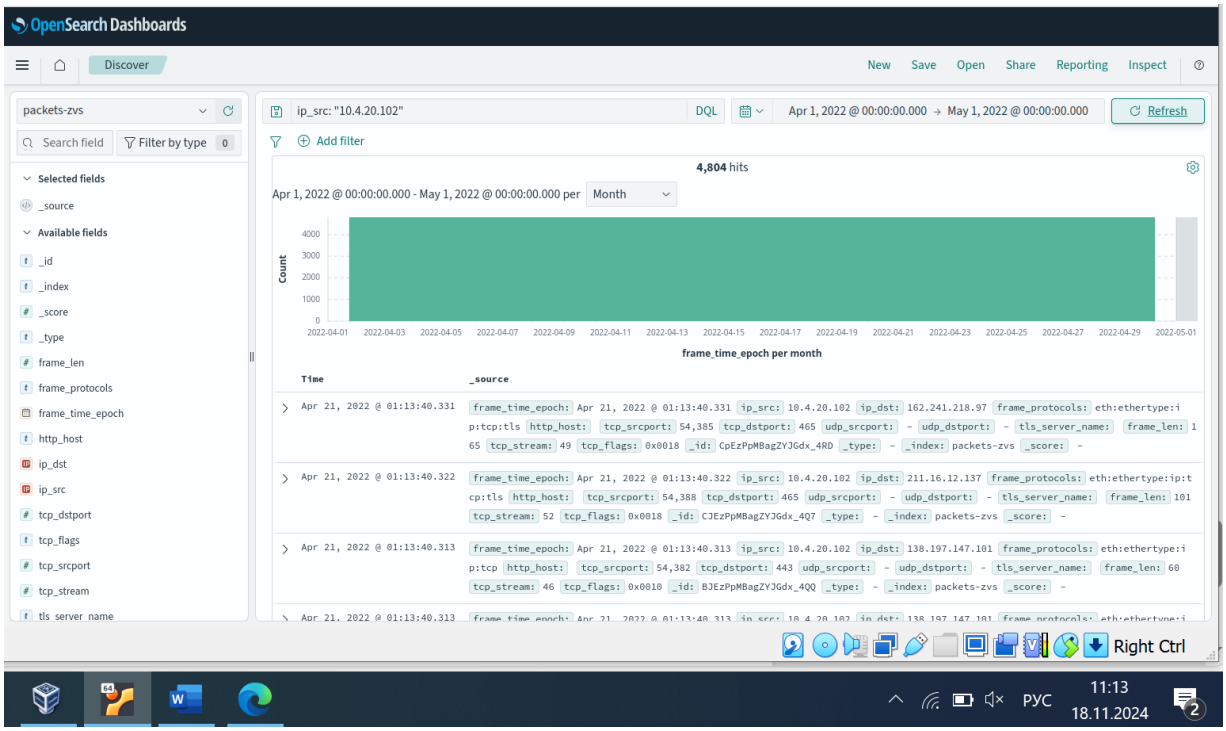
```
import csv
from opensearchpy import OpenSearch

# Настройки подключения к OpenSearch
OPENSEARCH_HOST = "http://localhost:9205"  # Порт 9205 соответствует вашему
docker-compose
USERNAME = "admin"
PASSWORD = "Vszub@2409"
INDEX_NAME = "packets-zvs"  # Имя индекса в OpenSearch

# Инициализация клиента OpenSearch
client = OpenSearch(
    hosts=[OPENSEARCH_HOST],
    http_auth=(USERNAME, PASSWORD),
    use_ssl=False,
    verify_certs=False
)

# Функция создания индекса
def create_index(index_name):
    index_body = {
        "settings": {
            "number_of_shards": 1,
            "number_of_replicas": 0
        },
        "mappings": {
            "properties": {
                "frame_time_epoch": {"type": "date", "format":
"epoch_second"},
                "ip_src": {"type": "ip"},
                "ip_dst": {"type": "ip"},
                "frame_protocols": {"type": "text"},
                "http_host": {"type": "keyword"},
                "tcp_srcport": {"type": "integer"},
                "tcp_dstport": {"type": "integer"},
                "udp_srcport": {"type": "integer"},
                "udp_dstport": {"type": "integer"},
                "tls_server_name": {"type": "keyword"},
                "frame_len": {"type": "integer"},
                "tcp_stream": {"type": "integer"},
                "tcp_flags": {"type": "text"}
            }
        }
    }
    if not client.indices.exists(index=index_name):
        response = client.indices.create(index=index_name, body=index_body)
        print(f"Index '{index_name}' created.")
    else:
        print(f"Index '{index_name}' already exists.")

# Функция загрузки данных из CSV в OpenSearch
def load_data_to_opensearch(csv_file, index_name):
    with open(csv_file, mode='r') as file:
        reader = csv.DictReader(file, delimiter=';')
        for i, row in enumerate(reader):
            doc = {
                "frame_time_epoch": float(row.get("frame.time_epoch", 0)),
                "ip_src": row.get("ip.src"),
                "ip_dst": row.get("ip.dst"),
                "frame_protocols": row.get("frame.protocols"),
                "http_host": row.get("http.host"),
                "tcp_srcport": int(row.get("tcp.srcport", 0)) if
row.get("tcp.srcport") else None,
                "tcp_dstport": int(row.get("tcp.dstport", 0)) if
```

```
row.get("tcp.dstport") else None,
                "udp_srcport": int(row.get("udp.srcport", 0)) if
row.get("udp.srcport") else None,
                "udp_dstport": int(row.get("udp.dstport", 0)) if
row.get("udp.dstport") else None,
                "tls_server_name":
row.get("tls.handshake.extensions_server_name"),
                "frame_len": int(row.get("frame.len", 0)),
                "tcp_stream": int(row.get("tcp.stream", 0)) if
row.get("tcp.stream") else None,
                "tcp_flags": row.get("tcp.flags")
            }
            response = client.index(index=index_name, body=doc)
            if response.get("result") == "created":
                print(f"Document {i + 1} indexed successfully.")
            else:
                print(f"Failed to index document {i + 1}: {response}")

# Основной поток
if __name__ == "__main__":
    CSV_FILE = "result1.csv"  # Замените на путь к вашему CSV файлу
    create_index(INDEX_NAME)
    load_data_to_opensearch(CSV_FILE, INDEX_NAME)
```

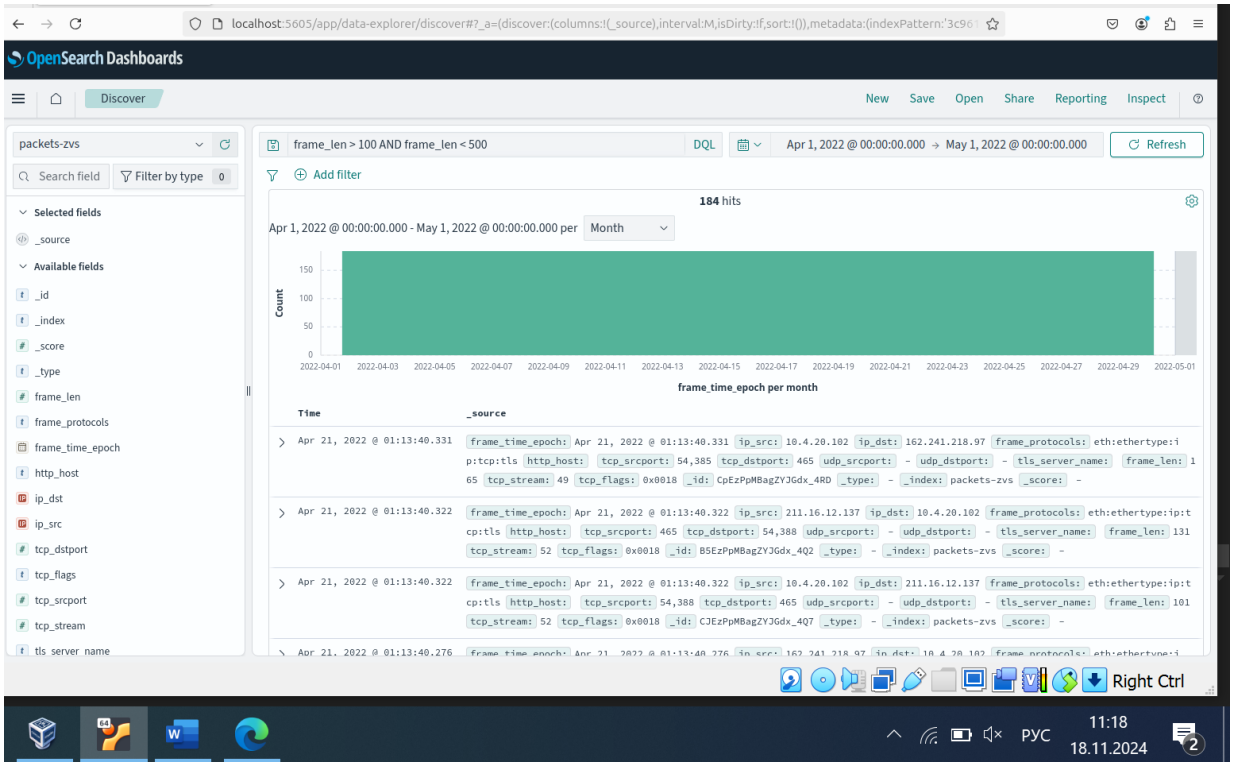


Изучите возможности поиска и анализа данных в разделе «Discover» (не менее 5 различных запросов).

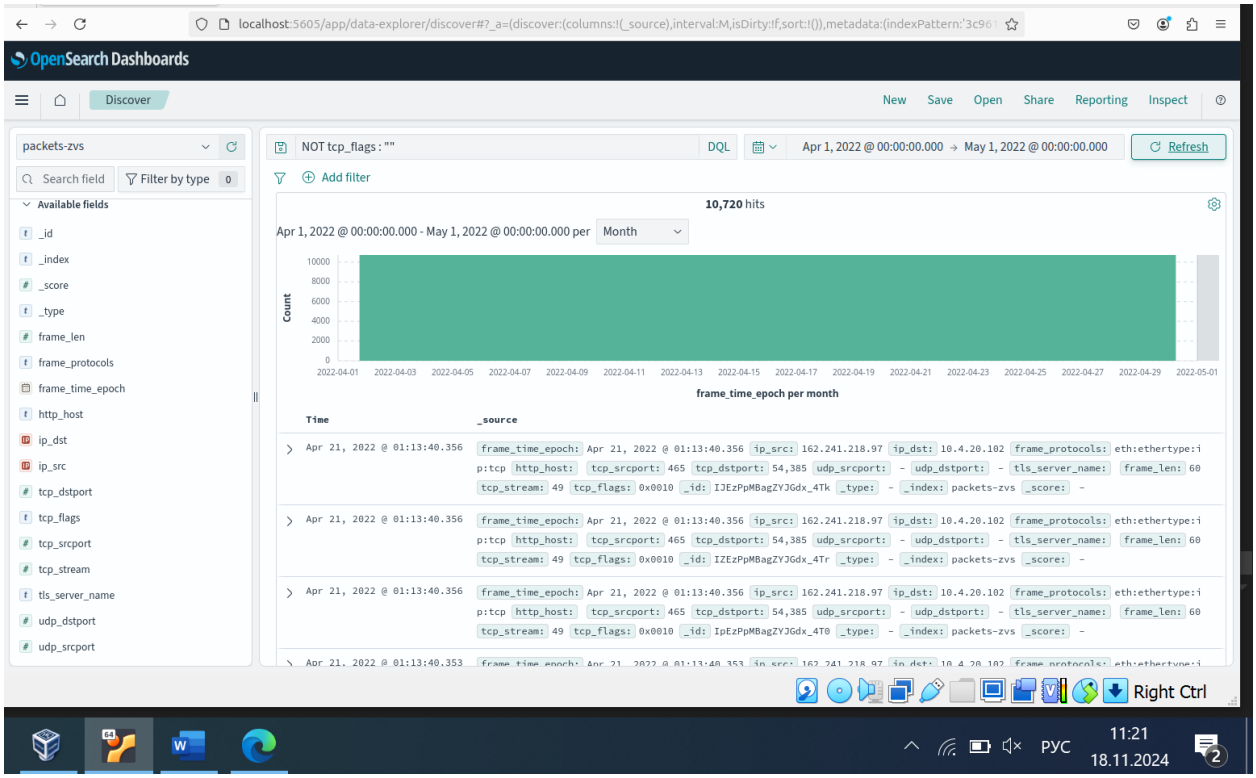
Ip src= 10.4.20.102



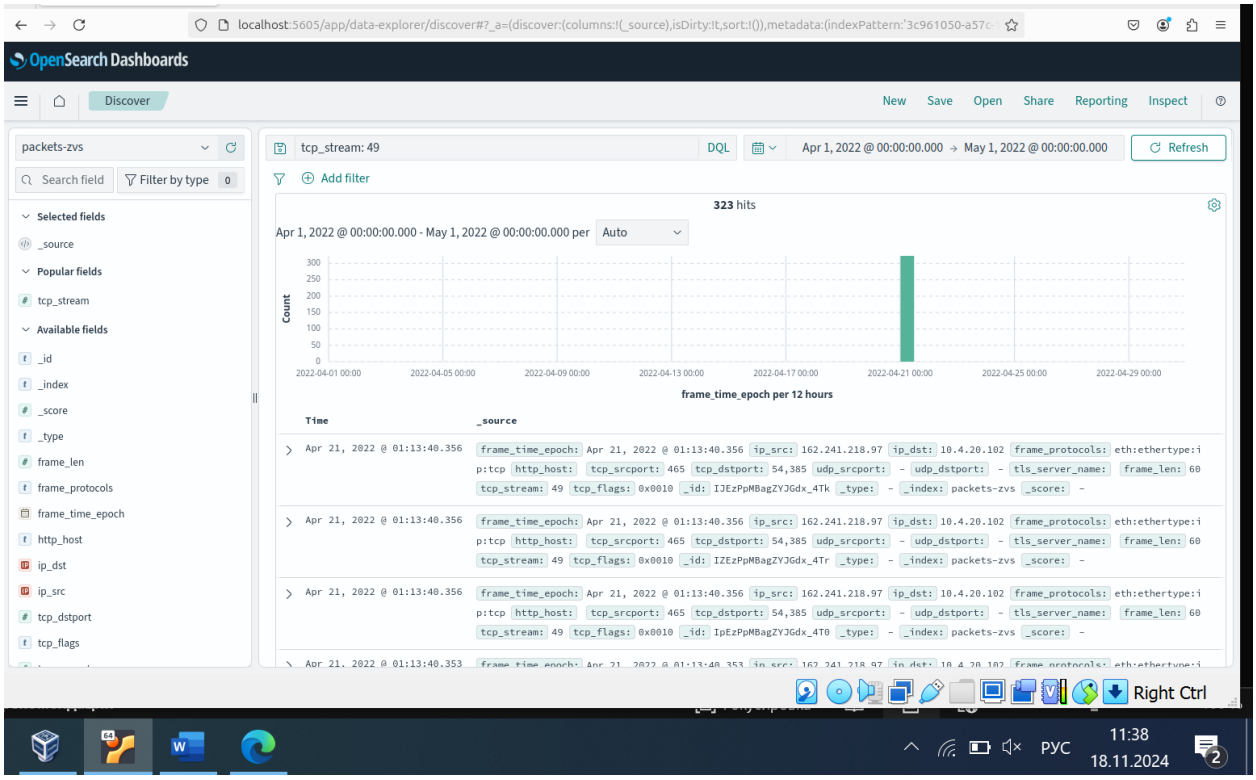
Поиск пакетов где 100<=frame_length<=500



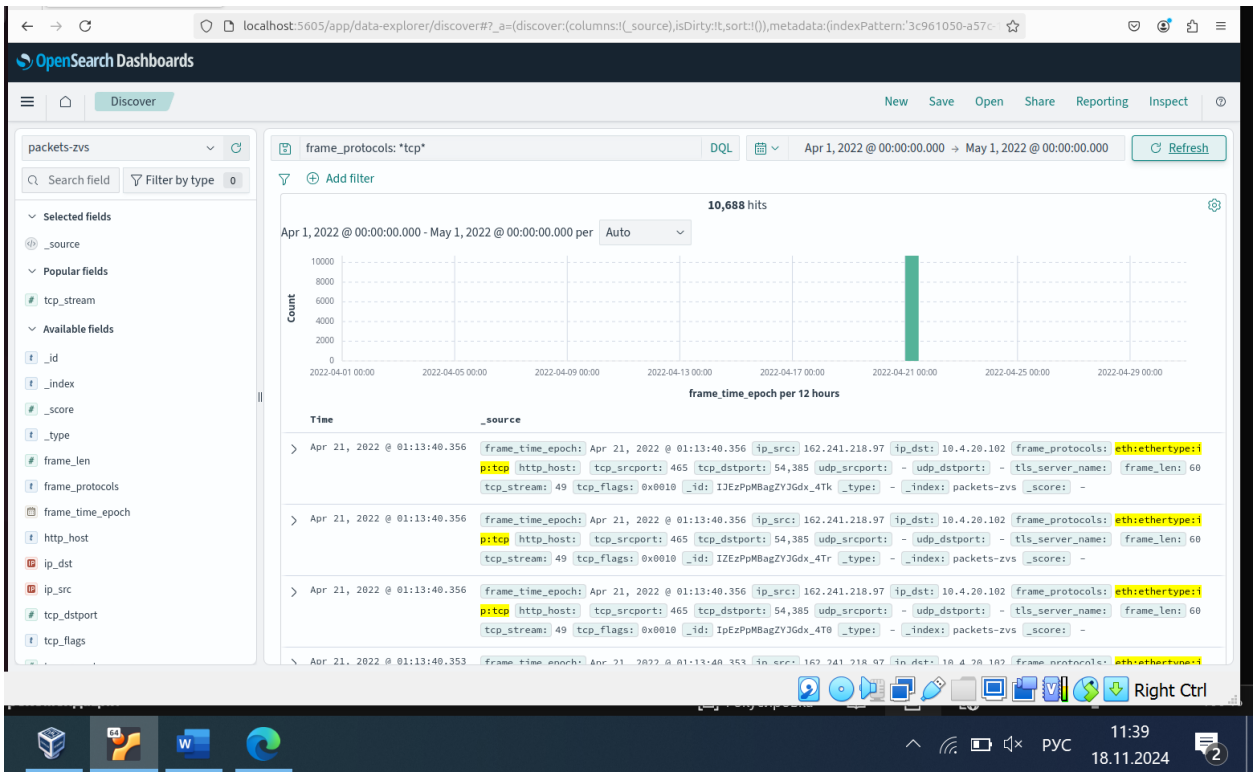
Поиск вывод записей действий (tcp_flags не путое)



Поиск всех пакетов сессии 49



Все пакеты, где использовался tcp



Сформируйте не менее 5 различных дашбордов (раздел «Dashboards»). Каждый должен иметь уникальное значение. Поясните их назначение и какие выводы вы сделали по результатам анализа данных.