***Wojskowa Akademia Techniczna***

***im. Jarosława Dąbrowskiego***



**Wydział Cybernetyki, kierunek informatyka - inżynieria systemów**

Realizacja zadania laboratoryjnego w ramach przedmiotu:

*Systemy Baz Danych*

Temat laboratorium:

***Bazy Danych typu wyszukiwarki***

**Opracował:** Radosław Relidzyński, **Grupa:** WCY23IX3S4

Spis treści

[Wstęp teoretyczny 3](#_Toc169205898)

[Treść zadania 3](#_Toc169205899)

[Środowisko 4](#_Toc169205900)

[Implementacja środowiska 4](#_Toc169205901)

[Definiowanie środowiska docker-compose.yml 4](#_Toc169205902)

[Zarządzanie danymi (logami) logstash.conf 7](#_Toc169205903)

[Przetwarzanie wstępne logów 8](#_Toc169205904)

[Uruchomienie środowiska 9](#_Toc169205905)

[Wizualizacja danych 14](#_Toc169205906)

[Podsumowanie 16](#_Toc169205907)

# Wstęp teoretyczny

**Baza danych** – „uporządkowany zbiór danych określających wybrany fragment rzeczywistości lub problemu, które są przechowywane trwale w pamięci komputerowej do której może mieć dostęp wielu użytkowników w dowolnej chwili czasu.”

**System zarządzania bazami danych** – „zorganizowany zbiór narzędzi (programów komputerowych i bibliotek), które umożliwiają wykonanie podstawowych operacji na danych (CRUD) zawartych w jednej lub więcej bazach danych.”

System baz danych – jego definicja wyraża się wzorem:

Gdzie:

[źródło: materiały z wykładu „Temporalne bazy danych” dr inż. Jarosława Koszeli]

# Treść zadania

Wymagania:

* Implementacja środowiska
* 4 wykresy dashboard
* opisać proces tworzenia indeksu z wgrywaniem danych
* opisać skąd są logi (plik) i w jaki sposób są przetwarzane (parser, groki)
* stworzenie 3 węzłów elasticsearch
* przedstawić i opisać konfigurację z logstashem
* prezentacja liczby wstawionych dokumentów
* pokazać jak przeparsowano datę (format/mapping, dopasowanie)
* Stworzyć wizualizację danych – 4 wizualizacje (wykresy dashboard)

# Środowisko

**„Elasticsearch** jest silnikiem wyszukiwania i analizy danych pozwalającym również na składowanie danych. Cechuje się dużą szybkością oraz skalowalnością. Komunikacja z silnikiem odbywa się przy użyciu zapytań REST, dzięki czemu jest bardzo prosta w zaimplementowaniu do naszego systemu.”  
[źródło: <https://ermlab.com/blog/technicznie/elasticsearch-jako-narzedzie-do-przeszukiwania-danych/>]

Środowisko zostanie utworzone lokalnie przy pomocy narzędzia docker-compose oraz różnych kontenetów, głównie dla elasticsearch oraz kibany dostarczającej narzędzia wizualizacji danych.

# Implementacja środowiska

### Definiowanie środowiska docker-compose.yml

version: '3.8'  
services:  
 es01:  
 image: elasticsearch:7.16.2  
 container\_name: es01  
 restart: always  
 volumes:  
 - elastic\_data1:/usr/share/elasticsearch/data/  
 environment:  
 - node.name=es01  
 - cluster.name=es-docker-cluster  
 - discovery.seed\_hosts=es02,es03  
 - cluster.initial\_master\_nodes=es01,es02,es03  
 - xpack.ml.enabled=false  
 - xpack.monitoring.collection.enabled=true  
 - bootstrap.memory\_lock=true  
 - "ES\_JAVA\_OPTS=-Xmx256m -Xms256m"  
 ulimits:  
 memlock:  
 soft: -1  
 hard: -1  
 ports:  
 - '9200:9200'  
 networks:  
 - elastic  
  
 es02:  
 image: elasticsearch:7.16.2  
 container\_name: es02  
 restart: always  
 volumes:  
 - elastic\_data2:/usr/share/elasticsearch/data/  
 environment:  
 - node.name=es02  
 - cluster.name=es-docker-cluster  
 - discovery.seed\_hosts=es01,es03  
 - cluster.initial\_master\_nodes=es01,es02,es03  
 - xpack.ml.enabled=false  
 - xpack.monitoring.collection.enabled=true  
 - bootstrap.memory\_lock=true  
 - "ES\_JAVA\_OPTS=-Xmx256m -Xms256m"  
 ulimits:  
 memlock:  
 soft: -1  
 hard: -1  
 ports:  
 - '9201:9200'  
 networks:  
 - elastic  
  
 es03:  
 image: elasticsearch:7.16.2  
 container\_name: es03  
 restart: always  
 volumes:  
 - elastic\_data3:/usr/share/elasticsearch/data/  
 environment:  
 - node.name=es03  
 - cluster.name=es-docker-cluster  
 - discovery.seed\_hosts=es01,es02  
 - cluster.initial\_master\_nodes=es01,es02,es03  
 - xpack.ml.enabled=false  
 - xpack.monitoring.collection.enabled=true  
 - bootstrap.memory\_lock=true  
 - "ES\_JAVA\_OPTS=-Xmx256m -Xms256m"  
 ulimits:  
 memlock:  
 soft: -1  
 hard: -1  
 ports:  
 - '9202:9200'  
 networks:  
 - elastic  
  
 logstash:  
 image: logstash:7.16.2  
 container\_name: logstash  
 restart: always  
 volumes:  
 - ./logstash/:/logstash\_dir  
 command: logstash -f /logstash\_dir/logstash.conf  
 depends\_on:  
 - es01  
 ports:  
 - '9600:9600'  
 environment:  
 LS\_JAVA\_OPTS: "-Xmx256m -Xms256m"  
 networks:  
 - elastic  
  
 kibana:  
 image: kibana:7.16.2  
 container\_name: kibana  
 restart: always  
 ports:  
 - '5601:5601'  
 environment:  
 ELASTICSEARCH\_URL: http://es01:9200  
 ELASTICSEARCH\_HOSTS: '["http://es01:9200","http://es02:9200","http://es03:9200"]'  
 networks:  
 - elastic  
  
volumes:  
 elastic\_data1:  
 elastic\_data2:  
 elastic\_data3:  
  
networks:  
 elastic:

Zastosowane kontenery:

* es1, es2, es3 – 3 węzły Elasticsearch
  + obraz: elasticsearch:7.16.2
  + wykorzystywane porty: 9200, 9201, 9202 (wszystkie przekierowywane na 9200)
  + sieć: elastic (jedyna dostępna)
  + paczka kolekcji monitoringu ustawiona na true. Jest to automatyzcne włączenie monitoringu danych tak, żeby przy uruchomieniu nie wyświetlała się taka wiadomość:A screenshot of a computer

    Description automatically generated
* logstash
  + obraz: logstash:7.16.2
  + Wykorzystywany port: 9600
  + ścieżka do folderu z plikiem logstasha: ./logstash/
  + instrukcja wywołania logstasha wraz z podaniem nazwy pliku: logstash -f /logstash\_dir/logstash.conf
* kibana
  + obraz: kibana:7.16.2
  + Wykorzystywany port: 5601
  + adres url do elasticsearch: <http://es01:9200>
  + adresy dla hostów elasticsearch: '["http://es01:9200","http://es02:9200","http://es03:9200"]'

Uruchomienie polecenia „docker-compose up -d” tworzy 5 kontenerów, 3 węzły elasticsearch, interfejs do wizualizacji kibana oraz narzędzie przetwarzania danych logstash.

### Zarządzanie danymi (logami) logstash.conf

input {  
 file {  
 path => "/logstash\_dir/logs\_source.csv"  
 start\_position => "beginning"  
 sincedb\_path => "/dev/null"  
 }  
}  
  
filter {  
 csv {  
 separator => ","  
 columns => [  
 "error\_level",  
 "error\_message",  
 "raw",  
 "remote\_host",  
 "remote\_logname",  
 "remote\_user",  
 "request\_header\_referer",  
 "request\_header\_user\_agent",  
 "request\_header\_user\_agent\_\_browser\_\_family",  
 "request\_header\_user\_agent\_\_os\_\_family",  
 "request\_header\_user\_agent\_\_os\_\_version\_string",  
 "request\_http\_ver",  
 "request\_method",  
 "request\_url",  
 "response\_bytes\_clf",  
 "status",  
 "time\_received"]  
 skip\_header => true  
 }  
  
 date {  
 match => ["time\_received", "[dd/MMM/yyyy:HH:mm:ss Z]", "[EEE MMM dd HH:mm:ss yyyy]", "[dd/MMM/yyyy:HH:mm:ss]", "[dd/MMM/yyyy HH:mm:ss]"]  
 target => "@timestamp"  
 timezone => "UTC"  
 }  
}  
  
output {  
 stdout {  
 codec => rubydebug  
 }  
  
 elasticsearch {  
 hosts => ["http://es01:9200","http://es02:9200","http://es03:9200"]  
 index => "new\_logs"  
 }  
  
 file {  
 path => "/logstash\_dir/log\_output.log"  
 }  
}

1. Sekcja ‘input’:
   * file – definiowanie źródła w postaci ścieżki do pliku, miejsca z którego ma zacząć przetwarzać (w tym przypadku od poczatku) oraz określa ścieżkę do pliku „sincedb” śledzącą progres przetwarzania pliku. Przekierowywanie do „/dev/null” oznacza, że nie będzie on śledzony, wszystkie przekierowywane tam dane zostaną utracone.
2. Sekcja „filter”:
   * csv – deklaracja separatora i listy kolumn do przetworzenia pliku csv, wraz z pominięciem pierwszego wiersza (nagłówków)
   * date – definicja formatów dat do dopasowania, strefy czasowej oraz tego, która data ma zostać dodana do „@timestamp”. Są one różne w zależności od rekordu, stąd wymagane jest stworzenie różnych formatów.
3. Sekcja „output”:
   * stdout – przetwarzanie standardowego wyjścia w formacie „rubydebug”
   * elasticsearch – definicja hostów wyjściowych dla klastra elasticsearch oraz nazwy indeksu, do którego będą zapisywane przetworzone dane
   * file – podanie ścieżki i nazwy pliku, w którym zapisane mają być przetworzone logi.

# Przetwarzanie wstępne logów

Analiza logów prezentuje się następująco:

* Pierwszy wiersz to zbiór kolumn
* Logi są zapisane w jednej lub dwóch liniach
* Logi posiadające „error\_lever” oraz „error\_message” są zapisane w jednej linii
* Logi nie posiadające „error\_lever” ani „error\_message” są zapisane w dwóch liniach

Aby ułatwić ich przetwarzanie, logi zapisane w dwóch liniach przetworzone zostały tak, aby być w jednej linii. Dzięki temu logstash nie wymaga konfiguracji przy pomocy instrukcji „multiline”, będzie mógł bez tego trafnie je przeparsować

# Uruchomienie środowiska

#### Wywołanie polecenia „docker-compose up -d”

A screenshot of a computer program

Description automatically generated

#### Podgląd utworzonych kontenerów w aplikacji Docker Desktop

A screenshot of a computer

Description automatically generated

#### Widok początkowy Kibany

A screenshot of a computer

Description automatically generated

#### Widok indeksów, prezentacja liczby wstawionych dokumentów

A screenshot of a computer

Description automatically generated

#### Tworzenie patternu dla indeksów

A screenshot of a computer

Description automatically generated

#### Przegląd wszystkich logów

A screenshot of a computer

Description automatically generated

# Wizualizacja danych

### Procentowy udział kodów odpowiedzi w logach

A pie chart with numbers and a circle

Description automatically generated

### Top 5 najbardziej aktywnych zdalnych hostów

A graph of a number of green rectangular objects

Description automatically generated with medium confidence

### Top 5 najczęściej wykorzystywanych przeglądarek z wyszczególnieniem udziału wersji http

A graph with green and blue bars

Description automatically generated

### Podsumowanie metod dla zapytań w zależności od wersji http

A graph with a bar

Description automatically generated with medium confidence

# Podsumowanie

W ramach przeprowadzonego zadania wykonany został pełny proces implementacji środowiska baz danych typu wyszukiwarki w ramach technologii „Elasticsearch”, obejmujący konfigurację oraz uruchomienie kontenerów Elasticsearch, Logstash oraz Kibana. Udało się przetworzyć logi. Wizualizacja wyników w Kibana pozwoliła na wizualizację wybranych aspektów logów.

Elasticsearch to bardzo efektywne narzędzie do przetwarzania i przeglądania logów. Jak pokazało przygotowywanie wizualizacji potrafi przetworzyć bardzo duże ilości rekordów w bardzo krótkim czasie. Możliwości narzędzia w zakresie zarządzania formatami oraz sposoby przechowywania rekordów wskazują na duży jego potencjał przy bardzo dużych systemach.