Fakultet tehničkih nauka
Univerzitet u Novom Sadu

Arhitekture sistema velikih skupova podataka

Elastic Stack

Student
Stefan Aleksić

Broj indeksa
E2 42/2022

11. januar 2023.

Sažetak

Sadržaj

Spisak slika

Spisak tabela

Uvod

Glava 1

Elastic Stack

U ovom poglavlju biće opisan Elastik stek, kao primer postojećeg rešenja za prikupljanje i integraciju podataka. Bitno je napomenuti da se dalji tekst oslanja na dokumentaciju trenutno najnovije javno dostupne verzije alata [16], odnosno najnovije verzije komponenti koje čine alat, a koje će biti istaknute pri obrađivanju svake od komponenti. U ovom poglavlju će biti opisani delovi alata neophodni za implementaciju sistema koja će biti data u sledećem poglavlju.

1.1 Osnove *Elastic stack*-a

Elastik stek, takođe poznat kao ELK stek, predstavlja alat, koji postoji kao otvoreno rešenje (eng. open source) kompanije Elastik, a čije funkcionalnosti podrazumevaju: prikupljanje podataka u bilo kom formatu, njihovu obradu, nadogradnju, skladištenje, pretraživanje, analizu i vizuelizaciju u realnom vremenu. Akronim ELK identifikuje ključne komponente ovog steka, a to su: Elasticsearch, Logstash i Kibana [17].

Podaci mogu doći iz bilo kog izvora, što je na slici generalizovano komponentom Beats, koja predstavlja opcioni dodatak ELK steka i služi za slanje operativnih podataka na Logstash. Logstash komponenta predstavlja uređivač (eng. formatter) sistema, njena glavna funkcionalnost jeste prikupljanje ulaznih podataka bilo kog formata, parsiranje podataka i slanje istih u željenom obliku na Elasticsearch. Elasticseach uz pomoć indeksa vrši skladištenje podataka i olakšava njihovo pretraži-

vanje, takozvano pretraživanje celog teksta (eng. full-text search). Tako indeksirani podaci se uz pomoć Kibane mogu vizualno prikazivati i ujedno se vršiti njihova analiza. Neke od najčešćih vrsta podataka koji se obrađuju uz pomoć ELK steka podrazumevaju: evidentirane (eng. logged), metričke, bezbednosne i podatke biznis logike.

1.2 ElasticSearch

Elasticsearch je besplatan, distribuirani, otvoreni alat za skladištenje, pretragu i analizu (eng. search and analytics engine), raznih tipova podataka, uključujući tekstualne, numeričke, geo-prostorne (eng. geospatial), strukturne i nestrukturne. Manipulacija podataka je ostvarena REST API-jem, a alat omogućava izvanrednu horizontalnu skalabilnost. Elasticsearch je zasnovan na Apache Lucene biblioteci [18], tj. ideji indeksiranja podataka na osnovu samog sadržaja. Podaci se skladište kao dokumenti, koji pripadaju jednom indeksu, dok je uz pomoć distribuiranog modela, indekse moguće podeliti na manje komponente, odnosno krhotine (eng. shard), koji mogu biti rasprostranjeni kroz veći broj čvorova (eng. node). Trenutno aktuelna verzija na kojoj se zasniva opis alata jeste verzija 8.4 [17].

1.2.1 Osnove ElasticSearch-a

Elasticsearch arhitektura je projektovana za prikupljanje dokumenata, koji se skladište kao JSON objekti. Alat podržava ugnježdene strukture, što olakšava obradu kompleksnih podataka i upita. Za praćenje informacija o dokumentima dodeljuju se specijalni atributi, odnosno meta-podaci, koji počinju donjom crtom. Važni meta-podaci su:

- _index – predstavlja kom indeksu dokument pripada. U analogiji, npr. sa relacionim bazama, ovo bi predstavljalo jednu bazu podataka.
- _type – predstavlja klasu, odnosno mapiranje koje bi trebalo da ima svaki dokument koji pripada istom tipu. Ovo je analogno tabeli u relacionim bazama podataka. Trenutna verzija alata ovo polje trenira kao zastarelo (eng. deprecated), ostavljeno je samo radi kompatibilnosti sa starijim verzijama alata.
- _id – jedinstveni identifikator dokumenta u okviru tipa.

- _version – predstavlja broj izmena dokumenta, odnosno koliko puta je dokument bio kreiran, ažuriran, obrisan.

Glavni elementi arhitekture Elasticseach-a su: klaster, čvor, krhotina i analizator.

Klaster predstavlja grupu čvorova koji skladište podatke. U okviru konfiguracionog fajla config/elasticsearch.yml se može precizirati broj čvorova koji bivaju startovani u klasteru, kao i fizička ili virtualna adresa svakog od čvorova. Čvorovi u okviru jednog klastera su logički povezani i mogu razmenjivati podatke jedni sa drugima. Pri pokretanju, sistem automatski kreira jedan klaster sa jednim čvorom u njemu, što je dovoljno za osnovne potrebe prosečnog korisnika.

Čvor ne predstavlja server, već jednu instancu Elasticsearch-a, odnosno proces. Svaka instanca pripada jednom klasteru i zajedno, sa ostalim čvorovima u klasteru radi na rešavanju istog zadatka. Svaki čvor se može konfigurisati tako da obavlja barem jednu, a može obavljati više uloga u klasteru. Uloge koje čvoru mogu biti dodeljene obuhvataju:

- Master čvor – vrši kontrolu nad klasterom u vidu koordinacije čvorova koji pripadaju klasteru. Ovi čvorovi su odgovorni za operacije nad klasterom (eng. clasterwide operations), poput kreiranja i brisanja indeksa.
- Data čvor – vrši skladištenje i odgovoran je za invertovani indeks podataka. Ovo je podrazumevana uloga čvora.
- Klijentski čvor – predstavlja raspoređivač opterećenja (eng. load balancer) koji vrši usmeravanje pristiglih zahteva različitim čvorovima u klasteru.

Za ostvarivanje komunikacije se koriste dva glavna porta:

- Port 9200 – koristi se za filtriranje zahteva koji dolaze izvan klastera. Ovo su zahtevi REST API-ja koji se koriste za izvršavanje upita, indeksiranje i slično.
- Port 9300 – koristi se za među-čvornu (eng. inter-node) komunikaciju.

Krhotina predstavlja podskup dokumenata u okviru jednog indeksa. Naime, indeks nema ograničen broj dokumenata, niti ograničen memorijski kapacitet koji može da skladišti. Ukoliko se desi prekoračenje memorije na jednom od čvorova, Elasticsearch prestaje sa radom i javlja grešku. Kako bi se ovaj problem rešio, indeksi se dele

na krhotine, koje označavaju skalabilnu jedinicu za indeksiranje i omogućavaju distribuciju jednog indeksa na više čvorova. Ovim se postiže horizontalna skalabilnost sistema. Svaka krhotina funkcioniše kao nezavisan Lucene [18] indeks, koji može biti skladišten bilo gde u klasteru.

Replika predstavlja kopiju krhotine indeksa, dok se originalna krhotina naziva primarnom (eng. primary shard). Redundantnost podataka je glavni mehanizam na koji se sistemi otporni na greške oslanjaju. S obzirom da se radi o distribuiranom sistemom, ukoliko bi neki od čvorova otkazao, krhotine indeksa koje sadrži bi bile nedostupne, a samim tim i svi dokumenti koji se nalaze u nedostupnim podskupovima. Iz tog razloga se formiraju replike i dodeljuju različitim čvorovima u sistemu. Pored otpornosti na otkaze, ovo omogućava i brže pretraživanje podataka, jer više čvorova koji sadrže istu kopiju krhotine indeksa mogu istovremeno da vrše njenu pretragu i time ubrzaju proces pretrage indeksa u celini. Broj replika nije ograničen i mogu se precizirati nakon kreiranja indeksa, međutim, treba voditi računa jer iako je ovim čitanje ubrzano, svaka modifikacija je time znatno složenija.

Analizatori imaju zadatak da vrše parsiranje fraza i izraza u konzistentne termine. Ovo se odvija u procesu indeksiranja. Svaki analizator je sačinjen od jednog tokenizatora (eng. tokenizer) i većeg broja filtera tokena. Kada se naiđe na određeni izraz, tokenizator može da podeli izraz u prethodno definisane termine. U ovome se krije još jedna tajna brze pretrage podataka.

1.3 Logstash

Logstash je alat otvorenog koda za prikupljanje podataka sa mogućnostima nadovezivanja (eng. pipelining) u realnom vremenu. Alat može dinamičnim putem objediniti podatke iz različitih izvora, obogatiti ih, normalizovati i na kraju dostaviti podatke na unapred definisana odredišta. Ovim se pruža mogućnost prečišćavanja podataka za različite slučajeve upotrebe, naprednu analizu ili vizuelizaciju [21].

Na početku je Logstash bio korišćen za prikupljanje podataka evidencije (eng. log), međutim, mogućnosti koje pruža su prevazišle taj slučaj korišćenja. Naime, bilo koja vrsta događaja može biti prosleđena alatu, uz pomoć širokog spektra ulaznih dodataka (eng. input plugins), primljeni događaji mogu biti integrisani i transformisani uz pomoć dodataka za filtriranje (eng. filter plugins) i na kraju tako formatirani podaci mogu biti prosleđeni na razne vrste izlaza, uz pomoć izlazih dodataka (eng. output plugins). Sam proces obrade je podržan velikim brojem često korišćenih kode-

ra, a pritom je alat dizajniran za obradu velike količine podataka u realnom vremenu. Alat je razvijen u programskom jeziku JRuby i izvršava se na javinoj virtuelnoj mašini (eng. Java Virtual Machine), skraćeno JVM.

1.3.1 Kako *Logstash* funkcioniše?

Logstash za sebe ima vezane cevovode, ili tokove podataka (eng. pipeline) definisane ulaznim, transformacionim i izlaznim etapama. Ulazi očitavaju događaje, filteri ih modifikuju, a izlazi tako modifikovane događaje dostavljaju na definisana odredišta. Ulazi i izlazi imaju unapred podržane kodeke, koji omogućavaju kodiranje i dekodiranje podataka pri ulazu i izlazu iz toka obrade, bez korišćenja transformacionih filtera.

Svaki dodatak koji se koristi u okviru ulazne etape se izvršava u sopstvenoj niti i svaki od njih upisuje događaje u centralizovani red čekanja koji je ili u radnoj memoriji pokrenutog procesa, ili može biti na disku. Svaki radnik toka podataka (eng. pipeline worker) uzima blok događaja iz reda čekanja, sprovodi blok kroz definisane dodatke u okviru filter etape i na kraju šalje obrađene događaje na svaki od definisanih dodataka izlazne etape. Veličina bloka, kao i broj radnika toka podataka, odnosno niti, je podesiv kroz konfiguracione fajlove Logstash-a.

Logstash podrazumevano koristi redove čekanja koji se nalaze u radnoj memoriji za baferovanje događaja između etapa, ulaz i filter etapa, odnosno filter i izlaz etapa, što je brže rešenje. Međutim pri nastanku greške, odnosno obustavi procesa, baferovani podaci bivaju izgubljeni.

Ulaznim dodacima se definiše izvor podataka koji ulaze u sistem. Najčešće korišćeni ulazni dodaci podrazumevaju:

- file – vrši čitanje fajla sa fajl sistema nalik Unix komandi tail -0F
- syslog – osluškuje na podrazumevanom portu 514, koji predstavlja port za syslog poruke u Unix-u i parsira ih na osnovu RFC3164 formata [22].
- redis – vrši čitanje sa Redis servera, konkretno sa Redis kanala i listi. Redis je često korišćen kao berzijanac poruka (eng. message broker) u centralizovanoj instalaciji Logstash-a i služi za skladištenje događaja u okviru reda čekanja.
- beats – vrši obradu podataka koji se dobijaju uz pomoć Beats alata.

Filter dodaci predstavljaju procesne uređaje koji posreduju tokom podataka. Filteri mogu biti kombinovani uslovima koji diriguju da li se određena faza obrade izvršava ili ne, u zavisnosti od kriterijuma koji se ispituje. Najčešće korišćeni dodaci za filtriranje su:

- grok – parsira i gradi proizvoljan tekst. Ovaj dodatak je trenutno najpogodnije rešenje za parsiranje nestrukturnih tokova podataka u podatke koji imaju strukturu i moguće je nad njima izvršavati upite.
- mutate – izvršava generalne transformacije nad poljima događaja, poput: preimenovanja, uklanjanja, zamene, modifikacije i slično.
- drop – izvršava obustavu događaja u potpunosti. Korisno pri događajima koji imaju debug nivo ozbiljnosti .
- clone – vrši replikaciju događaja, sa mogućnošću za dalje dodavanje i uklanjanje pojedinih polja.
- geoip – nadograđuje IP adresu geo-prostornim koordinatama, kao i dostupnim informacijama za iste.

Izlazni dodaci predstavljaju destinacije konačne etape u izvršavanju toka podataka Logstash-a. Događaj može proći kroz više izlaza, međutim, njegova obrada je završena kada prođe kroz sve navedene izlaze. Najčešće korišćeni dodaci za izlaz podrazumevaju:

- elasticsearch – šalje događaj na Elasticsearch, samim tim se vrši indeksiranje podataka koji dalje mogu lako da se pretražuju i vizualizuju u Kibani.
- file – skladišti formatiran događaj na fajl u fajl sistemu.
- graphite – šalje događaj na graphite, koji predstavlja popularan alat otvorenog koda za skladištenje podataka metrike. [23]
- statsd – šalje podatke na statsd. Ovo je pozadinski servis (eng. deamon), koji se izvršava na Node.js platformi, a čija je funkcija osluškivanje podataka statistike poslate UDP transportnim protokolom i prosleđivanje primljenih podataka na jedan ili više priključivih (eng. pluggable) pozadinskih (eng. backend) servisa, poput gore navedenog graphite servisa. [24]

1.3.2 Logstash konfiguracioni fajlovi

Logstash sadrži dve vrste konfiguracionih fajlova, a to su:

- Pipeline konfiguracioni fajlovi, koji definišu sam tok podataka. Izgled ovih fajlova će biti detaljnije objašnjen u nastavku.
- Settings konfiguracioni fajlovi, koji definišu inicijalizaciju i tok rada samog Logstash procesa. Ovi fajlovi su generisani pri instaliranju samog Logstash servisa i podrazumevaju fajlove:
    - Logstash.yml – sadrži konfiguracione indikatore (eng. flags) za instancu programa. Ovde se mogu definisati: tip i lokacija bafera koji se koristi za red čekanja između etapa, veličina bloka događaja koju obrađuje jedan radnik, nivo evidentiranja (eng. logging level) i mnoga druga podešavanja.
    - Pipelines.yml – sadrži konfiguraciju većeg broja tokova podataka koji će se izvršavati u okviru jedne instance Logstash-a. Ovde se između ostalog navodi putanja do same konfiguracije toka podataka, kao i broj radnika u okviru toka, identifikator toka i slično.
    - Jvm.options – sadrži podešavanja za JVM indikatore. Ovde se između ostalog može definisati minimalna i maksimalna veličina radne memorije dodeljene procesu.
    - Log4j2.properties – sadrži podrazumevana podešavanja za log4j biblioteku [25].

Struktura konfiguracionih fajlova

Logstash konfiguracioni fajlovi za tokove podataka su pisani u specijalnom jeziku [26] koji je razvijen od strane samih osnivača programa. Jezik podseća na JSON format sa par izuzetaka, ali je vrlo jednostavan za razumevanje. Tok podataka, pipeline ima odvojene segmente za ulaznu, filter i izlaznu etapu, čija je oblast važenja uokvirena vitičastim zagradama. U okviru svake od etapa se mogu navesti dodaci koji se koriste. Bitno je naglasiti da se dodaci u okviru ulazne etape izvršavaju konkurentno, dok se u filter i izlaznoj etapi izvršavaju sekvencijalno.

Primer jednog konfiguracionog fajla je dat u okviru slike 7.

```
input {
    http {
        port => 3333
        tags => gateway
    }
}

filter {
    . . .
}

output {
    . . .
}
```

1: Primer konfiguracije toka podataka

Dodaci mogu zahtevati vrednosti za određena podešavanja. Tipovi vrednosti koje postoje u okviru konfiguracije su:

- Lista – definiše se uglastim zagradama, dok su elementi unutar zagrada odvojeni zarezima.
- Logička vrednost – definiše se vrednostima true i false.
- Bajt – definiše se kao string koji sadrži broj bajtova sa mernom jedinicom, bilo da je u pitanju SI (osnova 1000), ili binarna (osnova 1024) merna jedinica. Polje nije osetljivo na velika slova (eng. case-insensitive) i prepoznaje razmak između vrednosti i jedinice. Ukoliko se ne napiše jedinica, već samo celobrojna vrednost, podrazumeva se da vrednost predstavlja egzaktan broj bajtova.
- Kodek – definiše ime jednog od Logstash kodeka, koji se može naznačiti i u ulaznim i u izlaznim etapama. Kada se nađe u ulaznoj etapi, predstavlja način dekodiranja podataka koji ulaze u tok, dok na izlazu predstavljaju način kodiranja u određeni format.
- Heš mapa – definiše kolekciju ključ-vrednost elemenata u formatu “polje“ => “vrednost“, gde se elementi razdvajaju blanko znacima, ne zapetama kao kod listi i nizova.

- Broj – obuhvata realne brojeve, odnosno integer i float vrednosti.
- Lozinka – string vrednost koja ne biva evidentirana.
- URI – string vrednost koja predstavlja identifikator resursa, može biti potpun ili parcijalni URL. Ukoliko sadrži korisničko ime i lozinku, takođe se neće evidentirati.
- Putanja – definiše validnu putanju do resursa na sistemu.
- String – sekvenca karaktera, ukoliko sadrži blanko znake, neophodno je ograditi ga jednostrukim ili dvostrukim znakom navoda.
- Referenca na polje – predstavlja putanju do polja u događaju, ukoliko se radi o polju koje je u korenu događaja, onda se može koristiti notacija [polje] ili se izostaviti uglaste zagrade, polje. Ukoliko se referencira ugnježđeno polje, onda je neophodno koristiti notaciju sa uglastim zagradama i to po ugledu na oblik [polje prvog nivoa][polje drugog nivoa]… [polje n-tog nivoa].
- Komentari se označavaju kao u programskim jezicima Perl, Ruby i Python, počinju znakom taraba (#).

Uslovne strukture (eng. conditionals) su podržane, imaju poznatu strukturu kao i u ostalim programskim jezicima if, else if, else, a koriste se za dodatno kontrolisanje filter i izlazne etape toka podataka. Primer strukture je dat u nastavku (slika 8).

```
1 if EXPRESSION {
2     ...
3 } else if EXPRESSION {
4     ...
5 } else {
6     ...
7 }
```

2: Uslovne strukture u konfiguracionom fajlu toka podataka

Izraz (eng. expression) predstavlja logički izraz koji se svodi na jednu od logičkih vrednosti true, ili false. Ukoliko izraz sadrži referencu na polje, vrednost izraza će biti false ukoliko: polje ne postoji, polje ima nedefinisanu vrednost null ili postoji i ima vrednost false. Podržani operatori u okviru jezika podrazumevaju:

- Operatore poređenja:
  - Jednakost: ==, !=, <, >, <=, >=
  - Regularni izraz: = , !
  - Provera sadržaja: in, not in
  - Logičke operatore: and, or, nand, xor
- Unarni operator: !

Konkretan primer uslovne strukture sa izrazima je dat na slici 9.

```
filter {
    if [fooVal] in ["test", "debug"] and !([logLevel] == "debug") {
        mutate { add_tag => "testing" }
    }
^^I
^^Iif [foo] =~ "[0-9]+" {
        mutate { add_tag => "contains numbers" }
    }


    if "%{+HH}" < "16" {
      mutate { add_tag => "Before 16h" }
    }
}

output {
    if "testing" not in [tags] {
        elasticsearch(...)
    }
}
```

3: Primer izraza u okviru uslovnih struktura

Svakom događaju koji se generiše u ulaznoj etapi obrade se pridružuju polja koja bliže opisuju događaj. Ova polja su rezervisana, njihov tip zavisi od implementacije samog ulaznog dodatka koji ih generiše, a namenjeni su boljoj kontroli toka obrade:

- @metadata – heš mapa namenjena za skladištenje pomoćnih podataka u toku obrade. Ovo polje je dostupno u filter i izlaznoj etapi, međutim, podrazumevano se ne šalje na izlazne dodatke.
- @timestamp – predstavlja vremenski trenutak u kom je generisan događaj. Iz ovog polja se čitaju podaci koje koristi sprintf format, koji omogućava referenciranje vrednosti iz ostalih polja događaja.
- @version – predstavlja verziju dodatka koji je generisao događaj.
- tags – niz vrednosti koji se koriste radi bližeg opisivanja događaja.

1.3.3 Dodaci toka obrade

Dodaci toka obrade (eng. pipeline plugins) su programi napisani u programskom jeziku Java ili Ruby, koji predstavljaju alat za olakšavanje kontrole samog toka obrade, od ulaza, preko obrade, do izlaza samih podataka. Ove programe može napisati bilo ko u programskom jeziku Java ili Ruby i publikovati ih na javni Github repozitorijum sa kog ostali korisnici mogu da ih povlače i koriste u svojim sistemima. Samim tim, Logstash ima jako veliki skup dodataka koji mogu biti korišćeni, dok će ovaj rad nadalje obraditi samo one koji su neophodni za razumevanje sistema koji se implementira.

Syslog ulazni dodatak [27] vrši čitanje syslog [28] poruka preko mreže, tako što osluškuje poruke TCP i UDP protokola, generišući odgovarajuće događaje za tok podataka u kom je konfigurisan.

Grok filter dodatak [30] vrši parsiranje proizvoljnog teksta i njegovo prevođenje u strukturni podatak, nad kojim se mogu pisati upiti. Funkcioniše tako što koristi regularne izraze za pronalaženje šablona u okviru tekstualnih polja. Sintaksa koju prati ovaj dodatak je %SINTAKSA:SEMANTIKA. Sintaksa predstavlja ime unapred definisanog šablona regularnog izraza koji se koristi za identifikovanje dela teksta, dok semantika predstavlja identifikator pronađenog teksta koji se dalje može koristiti u toku podataka. Pored unapred definisanih šablona, mogu se direktno pisati regularni izrazi za pronalaženje šablona. Sintaksa regularnih izraza koja se koristi je Onigurama [31], koja ima izgled (?<identifikator polja>šablon). Još jedan način jeste kreiranje sopstvenih šablona u odvojenim fajlovima (putanje do fajlova se definišu u konfiguraciji dodatka), dok novo-definisani šablon ima izgled IME_ŠABLONA ŠABLON.

Mutate filter dodatak [32] omogućava generalizaciju mutacija nad poljima događaja. Moguće su operacije: preimenovanja, zamene, modifikacije polja u okviru događaja i slično. Redosled operacija u okviru dodatka je unapred ustanovljen, odnosno ne prati redosled navođenja operacija u okviru konfiguracije. Ukoliko je neophodno izvršiti više operacija koje ne prate ovakav redosled, neophodno je definisati više mutate filter dodataka u odgovarajućem redosledu.

Geoip filter dodatak [33] nadograđuje događaje informacijama o geo-prostornoj lokaciji prosleđene IP adrese. Ovaj dodatak je zasnovan na podacima iz MaxMind GeoLite2 baze podataka [34], koja dolazi kao podrazumevana baza dodatka. Moguće je kroz opcije konfigurisati bazu podataka, odnosno promeniti podrazumevanu. Primer odgovora koji se dobija za prosleđenu IP adresu dat je u okviru slike 10.

```
{
    "ip": "12.34.56.78",
    "geo": {
        "city_name": "Seattle",
        "country_name": "United States",
        "cotinent_code": "NA",
        "continent_name": "North America",
        "country_iso_code": "US",
        "postal_code": 98106,
        "region_name": "Washington",
        "region_code": "WA",
        "region_iso_code": "US-WA",
        "timezone": "America/Los_Angeles",
        "location": {
            "lat": 47.6062,
            "lon": -122.3321,
        }
    },
    "domain": "example.com",
    "asn": {
        "number": 98765,
        "organization": {
            "name": "Elastic, NV"
        }
    },
    "mmdb": {
        "isp": "InterLi Supra LLC",
        "dma_code": 819,
        "organization": "Elastic, NV"
    }
}
```

4: Primer podataka koje pruža geoip filter dodatak za prosleđenu IP adresu

Http filter dodatak [35] omogućava integraciju sa eksternim Veb servisima ili REST API-jima, tako što osposobljava Logstash da šalje zahteve podesive strukture na odgovarajuće krajnje tačke (eng. endpoint), kako bi obogatio događaj koji se obrađuje dodatnim informacijama. Parametar koji definiše zaglavlja zahteva (eng.

headers) koji se šalju bivaju definisana u okviru @metadata polja, kako ne bi okupirali destinacije na izlazu. Ovo je vrlo podesiv dodatak i ima veliki broj opcija za konfiguraciju.

Dns filter dodatak [36] pruža pretragu stavki, kanoničkih imena (eng. CNAME), ili stavki adresa (eng. A record), odnosno ukoliko je u pitanju inverzni DNS, pretražuju se pointer stavke (eng. pointer record, PTR). Bitno je napomenuti da ovaj dodatak može da obradi samo jednu stavku, dakle za veći broj stavki je potrebno više puta iskoristiti ovaj dodatak, s tim što ovakvo podešavanje može drastično da uspori sistem.

Elasticseach izlazni dodatak [37] omogućava skladištenje vremenskih serija podataka, poput podataka evidencije, događaja i metrike, kao i podataka čije skladištenje nije bazirano na vremenskim intervalima, direktno u Elasticsearch. S obzirom da se radi o dodatku koji radi sa osnovnim proizvodom kompanije, ovaj dodatak je jako fleksibilan i može kontrolisati razne segmente vezane za operacije u Elasticsearch-u. Najbitnije opcije u okviru dodatka, koje se vezuju za temu ovog rada podrazumevaju:

- action – enumeracija(index, delete, create, update) koja definiše koja vrsta operacije biva izvršavana. Podrazumevana vrednost za vremenske serije podataka, odnosno tokove podataka je create, dok je index podrazumevana vrednost za podatke koji se ne vezuju za vremenske intervale.
- index – string koji predstavlja indeks u koji se upisuju podaci obrađeni u okviru toka podataka
- hosts – uri tip vrednosti, koji predstavlja jednu ili više adresa čvorova na kojima je pokrenuta instanca Elasticsearch-a. Podrazumevana vrednost je [//127.0.0.1].

1.4 Kibana

Kibana je aplikacija otvorenog koda koja pruža pregledan korisnički interfejs ka Elastik steku, uz mogućnosti pretraživanja, vizuelizacije i analize podataka indeksiranih Elasticsearch-om. Takođe poznata i kao alat za kreiranje raznovrsnih grafikona (eng. charts), Kibana omogućava nadgledanje, upravljanje i održavanje bezbednosti klastera u ELK steku. Prvi put je postala dostupna javnosti 2013. godine, dok je

trenutna aktuelna verzija 8.4, na osnovu koje će biti opisani pojedini detalji same aplikacije [38].

Glavna primena Kibane podrazumeva: pretraživanje, vizualizaciju indeksiranih podataka u okviru Elasticsearch-a, kao i analizu podataka kroz kreiranje grafikona, poput: poluga, pita, tabela, histograma i mapa. Komandna tabla (eng. dashboard) kombinuje ove elemente na jedan pano i time omogućava analitički pregled podataka u realnom vremenu za razne slučajeve korišćenja, poput:

1. analize podataka evidencije,
2. nadgledanja metričkih podataka infrastrukture i kontejnera,
3. vizuelizaciju geo-prostornih podataka,
4. analizu sigurnosnih podataka,
5. analizu podataka biznis logike.

Upitni jezik Kibane

Kibana koristi specifični upitni jezik nazvan Kibana Query Language, ili skraćeno KQL, kojim je omogućeno jednostavno filtriranje podataka u svrhu vizualizacije. Ovaj jezik se razlikuje od standardnog Lucene upitnog jezika, jer ne omogućava pretragu uz pomoć regularnih izraza ili takozvanu pomućenu (eng. fuzzy) pretragu podataka, međutim omogućena je pretraga ugnježdenih polja kao i takozvanih skriptovanih polja (eng. scripted fields).

U okviru jezika se mogu identifikovati podvrste upitnog jezika: Upiti termina (eng. terms query), koji koriste egzaktan stil pretrage. Sintaksa ovog upita ima oblik <putanja do polja> : <list termina> Gde putanja do polja predstavlja ugnježdena polja počevši od korena dokumenta, sve do polja koje se pretražuje, nivoi su razdvojeni tačkom. Lista termina predstavlja prihvatljive vrednosti u okviru navedenog polja, dok se različiti termini odvajaju razmakom, a ukoliko se pretražuju egzaktne fraze, koriste se znaci navođenja. Putanja do polja i lista termina se odvajaju specijalnim karakterom dvotačka, na koji može da se gleda kao na operator in. Upiti Bulove algebre (eng. boolean queries) predstavljaju kombinovanje prethodno navedene podvrste upita logičkim operatorima: or, and i not. Po pravilu, and ima viši prioritet od operatora or, ukoliko je neophodna drugačija logika, koriste se zagrade.

Opsežni upiti (eng. range queries) predstavljaju upite koji numeričke i vrednosti datuma navedenih polja porede operatorima jednakosti: >, >=, < i <=. Moguće je koristiti i matematičke izraze pri poređenju vrednosti, što je jako pogodno za datume.

Upiti koji koriste džokere (eng. wildcard), koji je označen znakom * i može biti ili u delu putanje polja, čime se pokriva veći broj polja, ili u okviru vrednosti koje se traže u poljima i time pokriva veći spektar vrednosti, jer menja odgovarajući deo bilo kojom sekvencom bilo koje dužine. Upiti nad ugnježdenim poljima (eng. nested field queries) omogućavaju dva pristupa u filtriranju ugnježdenih dokumenata:

- Filtriranje jednog dokumenta na osnovu određenih delova upita <polje>: <ugnježdeno polje 1>:<izraz 1> and <ugnježdeno polje 2>:<izraz 2>
- Filtriranje više dokumenata na osnovu određenih delova upita <polje>: <ugnježdeno polje 1>:<izraz 1> and <polje>: <ugnježdeno polje 2>:<izraz 2>

Glava 2

Primer korišćenja

Glava 3

Zaključak

Literatura

[1] Daniel J. Rosenkrantz, Richard E. Stearns, and Philip M. Lewis, II. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing*, 6(3):563–581, 1977.

Dodatak A

. , ,
,
.

Dodatak B

[1]