DOI 10.7251/STED1902283K

Professional Paper – Stručni rad

UPOTREBA ELK STEKA ZA ANALIZIRANJE I PREZENTOVANJE DOGAĐAJA IZ IOT MREŽE

Nebojša Kuduz, Saša Salapura

Mtel, a.d. Banja Luka, Bosna i Hercegovina, Nebojsa.Kuduz@mtel.ba
PIM univerzitet, Fakultet računarskih nauka, Banja Luka, Bosna i Hercegovina

APSTRAKT

U radu je opisana implementaciju Elastisearch, Logstash i Kibana (ELK) steka, obrada, analiza i prikaz IoT podataka koji potiču iz nekoliko vrsta senzora. Na pripremljenoj Elasticsearch platformi prikupljeni su i integrisani podaci koje prikupljaju različiti senzori, izvršena je obrada i prikaz željenih informacija. Pohranjivanje ulaznih podataka različitog formata, pronalaženje i izdvajanje željenih informacija ključni je problem koji smo trebali riješiti u našem IoT okruženju. Koristili smo Grok filtar unutar Logstash-a za raščlanjivanje nestrukturiranih podataka i kao rezultat smo dobili strukturirane podatke u JSON formatu. Korišćenje Logstash-a omogućilo nam je usmjeravanje filtriranih podataka prema Elasticsearch-u. Elasticsearch pojednostavljuje skladištenja velike količine zapisa u nerelacionu bazu podataka koji se nakon skladištenja mogu jednostavno pretraživati. Zavisno od vrste podataka zaprimljenih od IoT senzora, primjenili smo prilagođene filtere za analizu primljenih podataka, njihovo indeksiranje i pohranjivanje u bazu. Željene informacije smo predstavili u Kibana okruženju i omogućili smo korisniku preged i manipulaciju statističkim podacima. Izborom Kibane postigli smo prikaz željenih analitičkih informacija iz IoT mreže u skoro realnom vremenu.

Predstavljeni koncept i platforma za analizu podataka mogu se koristiti u složenijim IoT mrežama radi automatizacije generisanja potrebne analitike što IoT sistemu pruža mogućnost da preduzima automatske radnje zasnovane na analizi prikupljenih IoT informacija.

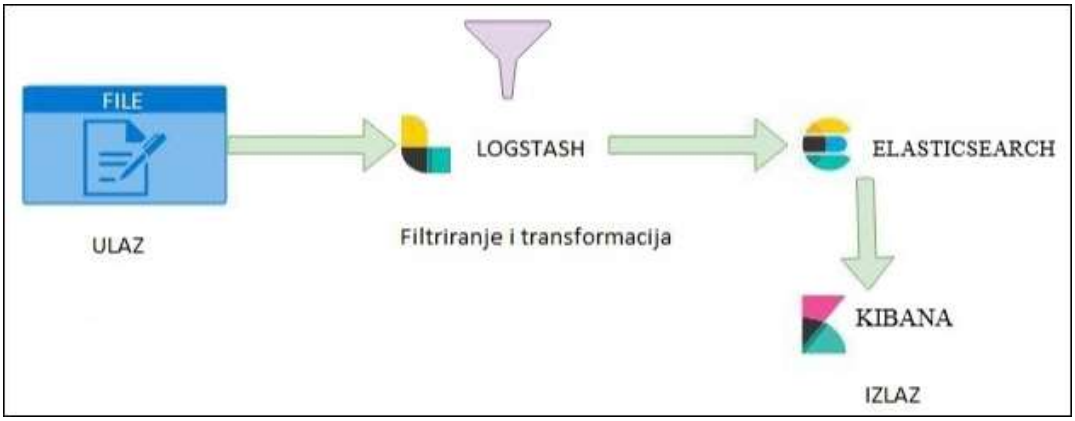
Ključne riječi: elastisearch, logstash, kibana, IoT, sensor.

UVOD

Obrada IoT događaja zahtijeva brzinu, prostor za skladištenje, ali i efikasne algoritme za obradu i iščitavanje podataka (Yeo, Chian, & Ng, 2014). Procesiranje i analiziranje događaja predstavlja veliki izazov u budućnosti s obzirom da će svi IoT uređaji biti povezani putem Interneta i buduće 5G mreže (Tulasi, Girish, 2016). Cilj ovog rada je da prikaže jedan način obrade, analize i prikaza informacija prikupljenih putem IoT senzora primjenom Elasticsearch-Logstash-Kibana (ELK) steka. Ovi alati su dizajnirani da rukuju velikim brojem podataka, ali se mogu uspješno primijeniti i za spremanje, pretraživanje i vizualizaciju podataka. Kako bi se pokazala praktična primjena ideje, korišteni su odabrani senzori koji očitavaju vodostaj, temperaturu, vlažnost, kvalitet vazduha i lokaciju. Pretpostavka je da sadržaj podataka mora biti brzo dostupan u realnom vremenu što nas je opredijelilo za izbor ELK platforme. Primjena ELK steka omogućila nam je korišćenje naprednih metoda obrade za dobijanje konkretnih vrijednosti iz događaja kao i prikupljanje, obradu i vizuelizaciju događaja. Ovako obrađene podatke mogu da koriste krajnji korisnici ili se mogu dalje upotrijebiti za analitiku, statistiku, alarmiranje ili u mašinskom učenju. Prikazani način obrade podataka je i svojevrsna primjena *edge computing-a* i usmjerena je na obradu podataka bliže izvoru nastanka događaja i na smanjenje kašnjenja dostupnosti informacija u aplikacijama koje obrađuju i prikazuju podatke u skoro realnom vremenu.

PRIKUPLJANJE I METODOLOGIJA OBRADE PODATAKA

Za prenos podataka koristimo LoRaWAN mrežu. Podaci od LoRa krajnjih tačaka, odnosno senzora, moraju proći kroz nekoliko uređaja prije nego što dođu do ELK steka.



Slika 1. Arhitektura ELK steka
Figure 1. ELK stack architecture

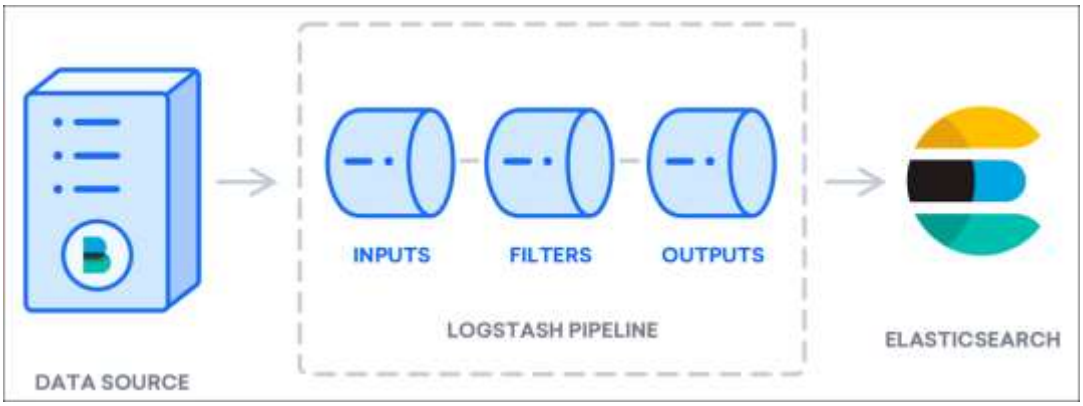
LoRaWAN (Long Range Wide Area Network) je protokol bežične mreže male snage dizajniran za jeftinu dvosmjernu sigurnu IoT komunikaciju. Tehnologija koja se koristi u LoRaWAN mreži dizajnirana je za povezivanje niskobudžetnih, akumulatorskih senzora na većim udaljenostima. Infrastrukturno, radi se o mreži koja obezbjeđuje upravljanje, kontrolu, automatizaciju i prenos senzorskih podataka u realnom vremenu.

Senzori koje koristimo za slanje podatke o vodostaju su Numeus UltraSonic, a senzor LibeliumAir mjeri temperaturu, vlažnost i kvalitet vazduha. Informacije koje prikupljamo sa senzora nisu produkcionog karaktera i potiču od uspostavljenog željenog testnog okruženja.

Sistem radi na standardan način, server prikuplja podatke sa senzora, odnosno sa mrežnog posrednika. Senzori i mrežna upravljačka aplikacija šalju *uplink* poruke periodično na način na koji su konfigurisani. Poruke, odnosno događaji koji su pristigli na server su različitog formata i nalaze se u strukturiranom ili nestrukturiranom obliku. Pohranjuju se u datoteke gdje ih prihvata i dalje obrađuje Logstash.

OBRADA PODATAKA - LOGSTASH

Logstash je popularan izbor zbog svoje stabilne integracije, moćnih mogućnosti obrade ulaznih podataka i preko 200 unaprijed ugrađenih dodataka koji nam pomažu da naše podatke indeksiramo na željeni način. Logstash je agregator i filtar na strani servera koji omogućuje prikupljanje i obradu podataka iz različitih izvora, transformaciju podataka i slanje do željenog odredišta (slika 2). U našem slučaju prikupljanje se vrši putem prilagodljivih ulaznih datoteka, koje koristimo kao sabirnicu poruka. Nakon što ulazni modul prikupi podatke, oni se mogu obraditi mnogobrojnim filterima koji mijenjaju i filtriraju ulazne događaje. U našem slučaju prikupljanje se vrši putem prilagodljivih ulaznih datoteka, koje koristimo kao sabirnicu poruka. Nakon što ulazni modul prikupi podatke, oni se mogu obraditi mnogobrojnim filterima koji mijenjaju i filtriraju ulazne događaje. Nakon željenih transformacija, Logstash usmjerava događaje na standardni izlaz ili ka različitim servisima i aplikacijama, a u našem slučaju je to ElasticSearch. Logstash arhitekturu čine tri komponente: Ulaz, Filter i Izlaz.



Slika 2. Logstash arhitektura
Figure 2. Logstash pipeline

Unašem slučaju prikupljanje se vrši putem prilagodljivih ulaznih datoteka, koje koristimo kao sabirnicu poruka. Nakon što ulazni modul prikupi podatke, oni se mogu obraditi mnogobrojnim filterima koji mijenjaju i filtriraju ulazne događaje. Nakon željenih transformacija, Logstash usmjerava događaje na standardni izlaz ili ka različitim servisima i aplikacijama, a u našem slučaju je to ElasticSearch. Logstash arhitekturu čine tri komponente: Ulaz, Filter i Izlaz.

Ulaz je posebno interesantan zbog sposobnosti obrade logova i događaja iz različitih izvora. Logstash može da procesira podatke sa više od 50 različitih ulaza i formate koji su nastali primjenom različitih tehnologija (Chhajed, 2015). Pomoću ovih ulaza podaci se mogu uvesti iz više izvora i njima manipulišemo po želji i na kraju ih spremamo u ElasticSearch za analizu i dalju obradu. Ulazni podaci su početna tačka za konfiguraciju Logstash.

Izlaz, slično kao i Ulaz, dolazi sa velikim brojem mogućnosti koje prosljeđuju događaje na različite lokacije, ka drugim uslugama i tehnologijama. Moguće je pohraniti događaje pomoću izlaza kao što su File ili CSV, pretvoriti ih u poruke ili ih poslati raznim servisima poput ElasticSearch-a. Broj kombinacija ulaza i izlaza čini Logstash svestranim transformatorom događaja, pa kao takav i predstavlja naš logičan izbor.

Filteri vrše posredničku obradu događaja. Filteri i transformacije se često primjenjuju uslovno u zavisnosti od karakteristika događaja. Logstash ima nekoliko vrlo moćnih filtera pomoću kojih možemo manipulisati, transformisati ili stvarati nove događaje. Moć ovih filtera čini Logstash svestranim i korisnim alatom. Logstash je u našoj predloženoj arhitekturi prijemnik podataka koje šalju senzori preko LoRaWAN mreže.

INDEKSIRANJE I PRIKAZ PODATAKA - ELASTICSEARCH I KIBANA

ElasticSearch (ES) je otvorena, distributivna i skalabilana platforma za pretragu u realnom vremenu koja je namijenjena za rad sa velikom količinom podataka. Odlikuje je visoka dostupnost i mogućnost distribucije na više čvorova, pa se uglavnom konfiguriše kao klaster rješenje koje je skalabilno i otporno na greške i otkaze čvorova (Paro, 2017).

ES možemo posmatrati kao neralacionu bazu podataka, indeksi sadrže dokumente, a dokumenti su određenog tipa. Npr. možemo kreirati indeks *senzori* i tip *senzor*. U tom slučaju indeks će sadržati dokumente koji definišu *senzori* i koji odgovaraju tipu *senzor*. Svaki indeks u ES ima mapiranje koje možemo uporediti sa definicijom šeme za sve tipove unutar nekog indeksa.

ES ima jednostavan i moćan API koji omogućuje pristup aplikacijama napravljenim u bilo kom programskom jeziku. ES je horizontalno skalabilan i izgrađen na Apache Lucene koji putem API-ja nudi napredne mogućnosti pretraživanja terabajta podataka. Ova mogućnost brze pretrage je uslovila intenzivan razvoj ES i primjenu u mnogim kompanijama i aplikacijama (Gormley and Tong, 2015).

ES preuzima i smješta objekte putem REST API metoda kao što su PUT, POST, GET i DELETE. U elastičnom pretraživanju svaki podatak ima definisan tip i indeks. ES je prilično fleksibilan pa možemo lako kreirati, ažurirati, pregledati i brisati indekse. U ES notaciji dokument je

osnovna jedinica podataka koji se može indeksirati. Dokumente određenog indeksa pohranjujemo u ES pomoću dinamičkog mapiranja ili putem mapiranja koje sami kreiramo. U formi klastera, ES je raširen na više čvorova koji zajedno drže podatke i pružaju objedinjeno indeksiranje i mogućnost pretrage na svim čvorovima. Indeks može pohraniti veliku količinu podataka koja može premašiti hardverske granice jednog čvora. Da bi riješio ovaj problem, ES pruža mogućnost podjele indeksa na više dijelova i stvaranje jedne ili više kopija indeksa ili dijelova indeksa koje zovemo replikama. Replikacija je važna za očuvanje dostupnosti i performansi u slučaju nestanka fragmenata ili pada čvora.

Kibana je platforma za analitiku i vizualizaciju namijenjena radu sa ES. Ona nam omogućuje vizualizaciju ES podataka. To čini ogromne i složene tokove podataka bržim i lakšim za razumijevanje preko grafičke prezentacije u realnom vremenu.

Kibana dashboard pruža fleksibilno dinamičko upravljačko okruženje za generisanje izvještaja. Generisani izvještaji mogu predstaviti podatke u tabelama, kružnim ili linijskim crtežima, grafikonima s prilagodljivim bojama, ili mape sa georeferenciranim događajima. Kibana uključuje i alate za dijeljenje vizualiziranih podataka. Slika 3. prikazuje Kibanu nakon prvog pristupa nekom od indeksiranih sadržaja. U gornjem dijelu nalazi se traka za pretraživanje, ispod nje se nalazi vremenska traka sa podacima u grafičkom prikazu. Poruke se prikazuju za odabrani vremenski period. Vremenski okvir prikaza se može mijenjati prema želji korisnika. U donjem dijelu stranice nalaze se poruke koje je Logstash primio i indeksirao u ES. Klasifikacija svake primljene poruke zasnovana je na Logstash filterima. Siva kolona oivičena plavim okvirom prikazuje sve klasifikacije ili oznake koje su prikupljene iz podataka primljenih na temelju Logstash filtera. Kako smo odabrali dvije vrste izvora podataka koji potiču od uplink poruka IoT senzora, označili smo primljene podatke posebnim tagovima kako bismo ih razlikovali prilikom korišćenja u Kibani.

Slika 3. Prikaz događaja u Kibana
Figure 3. Kibana dashboard layout

OBRADA I VIZUALIZACIJA PODATAKA IOT UREĐAJA

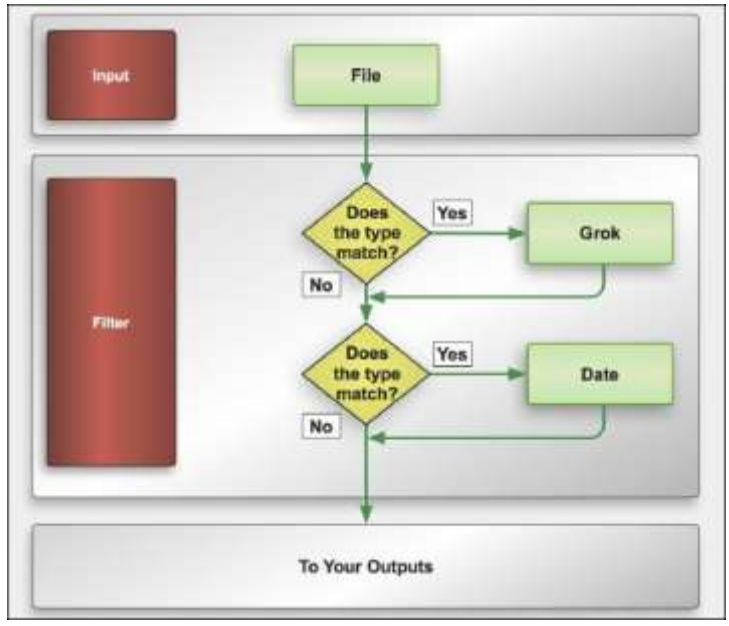
Podaci o mjerenjima na ulazu u logstash dolaze i u strukturiranom i u nestrukturiranom obliku, a korisne informacije koje sadrže očitane vrijednosti u kriptovanom i nekriptovanom obliku. U oba slučaja je potrebno izvršiti neophodno kodiranje i transformaciju unutar Logstash da bi format zapisa bio JSON prilagođen za ES. Podaci pristigli sa mrežnih uređaja smještaju se na disk prostor servera u odgovarajuće fajlove gdje ih u realnom vremenu dohvata Logstash na ulazu i dalje vrši njihovu obradu.

Na slici 4. prikazan je format zapisa na Ulazu. U pitanju je složen višelinijski JSON zapis, a Logstash-u je potrebno eksplicitno navesti da se radi o višelinijskom formatu. Da bi upravljali zapisima ovoga tipa u ulaznoj deklaraciji koristimo *multiline* kodek. Opcije u ovom kodeku određuju na koji način se zapis sa više linija objedinjuje u jedan događaj. Kada se ovaj kodek koristi na ulazu, on se pokreće u jednoj niti, što mu omogućuje uspješno spajanje višelinijskih događaja.

```
{
 "type": "uplink",
  "meta": {
    "network": "xxx", "app": "xxx","device_addr": "xxx", "gateway": "xxx",
  },
  ...
  "params": {
      "modulation": {"type": "LORA",
      },
      "hardware": {
        ...
        "gps": {
            "lat": 44.77605819702148, "lng": 17.19153022766113, "alt": 221
        }
        ...
      },
      ...
      "time": 1571122368.352331,
    },
    "payload": "...crypted...",
}
```


Slika 4. Pojednostavljen izgled zapisa uplink poruke
Figure 4. Simplified view of uplink event

Sljedeći korak primijenjen u procesiranju je filtriranje i transformaciji zapisa. Pojednostavljen izgled algoritma koji smo koristili za filtriranje i transformaciju zapisa prikazan je na slici 5. Prvo koristimo *Grok* filtar za kreiranje polja i filtriranje podataka, konverziju tipova podataka i nakon toga nastavljamo da procesiramo samo željene podatke. *Grok* je jedan od mnogih ES dodataka koji nam olakšava analizu podataka sa programskim jezikom koji je razumljiviji od upotrebe regularnih izraza. *Date* filtar koristimo za formatiranja vremenskog zapisa kako bismo ga kasnije koristili za indeksiranje u ES.



Slika 5. Algoritam filtriranja događaja
Figure 5. Event filtering algorithm

Slika 6. prikazuje pojednostavljenu konfiguraciju Filtara.

```
input{
filter{
    grok {
        match => ["message" ,
        %{GREEDYDATA},
        "deviceid":%{INT:uredjaj:double},
        "zadnje_mjerenje_podataka":
        {"value":%{NUMBER:ocitanje:int},"unit":"mm"},
        "coordinates":\[%{SPACE}%{DATA:longitude:float},
        "zadnje_mjerenje_vrijeme":
        "%{TIMESTAMP_ISO8601:vrijeme}",
        "Battery voltage":
        {"value":%{DATA:napon},"unit":"V"},
        %{GREEDYDATA}]
    }
    if " grokparsefailure" in [tags] {
    mutate {
            add_field => [ "[mjerenje][location]", "%{longitude}" ]
            add_field => [ "[mjerenje][location]", "%{latitude}" ]
            convert => [ "[mjerenje][location]", "float" ]
            copy => { "[mjerenje][location]" => "pozicija" }
    }
    date {
        match => [ "vrijeme", "YYYY-MM-dd HH:mm:ss", "ISO8601" ]
        timezone => "Europe/Sarajevo"
        target => "@timestamp"
    }
}
output{
```

Slika 6. Filtriranje i transformacija poruke
Figure 6. Filtering and Transforming

Ulazni zapis u posebnom polju sadrži korisni sadržaj, odnosno vrijednosti koje su očitali senzori. Budući da su informacije o korisnom sadržaju prenose u kriptovanom obliku, potrebno ih je dekriptovati i raščlaniti prije indeksiranja u ES. Logstash je pisan u JRuby i posjeduje moćan *ruby* filtar, pa smo dekripciju uradili upotrebom *ruby* filtra. Nakon *ruby* transformacije vrijednosti očitanja dobijamo u csv formatu kao u (1).

КСКСКС # ГТГ # 139 # ТЦ: 21.36 # ХУМ: 55.0 # ПРЕС: 99635.88 # ЦО2: 215.565 # ПМ1: 6.5900 # ПМ2_5: 8.2700 # ПМ10: 9.6700 # (1)

Из овог формата једноставна трансформација употребом Грок филтра је могла да рашчланимо податке у одговарајућим пољима погодним за индексирање у ЕС што је приказано на Слици 7.

Na kraju izlaznu sekciju Logstash filtra jednostavno konfigurišemo tako da zapise šalje u ES prema odgovarajućem indeksu.

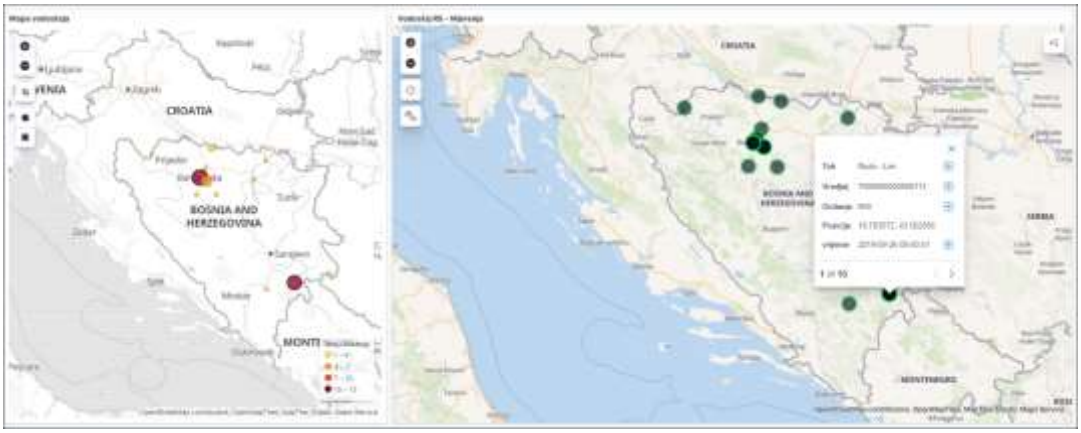
Da bismo uspješno indeksirali podatke i prikazali ih u Kibani, potrebno je da imamo i odgovarajuće mapiranje za tipove podataka koje indeksiramo. U ulaznom zapisu dobijamo koordinate lokacije senzora i da bismo dobili odgovarajući prikaz na mapi konvertovali smo dobijene koordinate u *geo_point* tip podatka koji će uz odgovarajuće mapiranje biti prepoznat u ES. ES za razliku od Logstash prepoznaje geo-referencirane tačke kao *geo_point* tip podatka. Postoji pet načina na koje se može odrediti geo-tačka, a mi smo formirali geo-tačke kao niz s formatom: "latitude, longitude". ES će automatski kreirati preslikavanje tipa dokumenta prilikom indeksiranja. Tokom mapiranja, ES zaključuje koga su tipa podaci za sva polja iz dokumenta. U našem slučaju za geo zapis potrebno je kreirati statičko mapiranje polja koja su *geo_point* tipa što postižemo jednostavnom *curl* direktivom (2).

```
ruby {
    init => "require 'base64'"
    code => "event.set( '[decoded]', Base64.decode64( event.get('[podatak]')))"
    }
grok {
    match => ["decoded" ,'#%{GREEDYDATA:Senzor_id}#%{DATA:NazivSenzora}
    #%{INT:OznakaSenzora:int}#TC:%{DATA:Temperatura:double}
    #HUM:%{DATA:Vlaznost:double}#PRES:%{DATA:Pritisak:double}
    #CO2:%{DATA:CO2:double}
    #%{GREEDYDATA:PM1}#%{GREEDYDATA:PM2.5}#%{GREEDYDATA:PM10}#']
    }
}
```

Slika 7. Pojednostavljen izgled filtra korisnog sadržaja
Figure 7. Simplified layout for payload filters

curl -XPUT "http://localhost:9200/airsenzor" -H 'Content-Type: application/json'
-d'{"mapping":{"properties": {"lokacija": {"type": "geo_point"}}}}' (2)

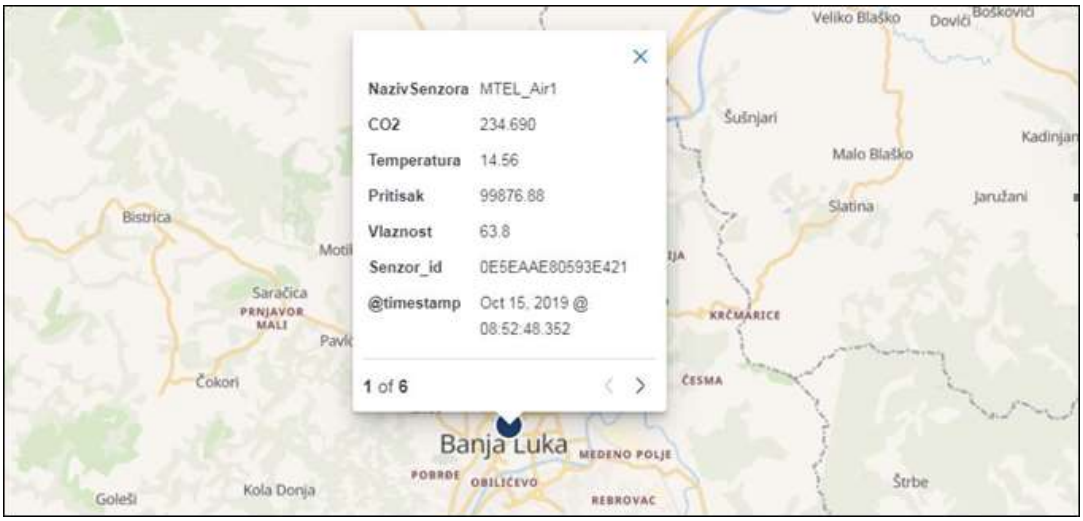
Dobijene podatke uvozimo u ES i indeksiramo ih po vremenskom zapisu koji smo ranije formirali u filter sekciji, a koji predstavljaju realna vremena očitanja mjernih vrijednosti. Podatke koje smo dobili potrebno je prikazati u Kibana Dashboard na odgovarajući način, što nam i jeste bio krajnji cilj.



Slika 8. Prikaz mape sa vrijednostima očitanja senzora mjerenja vodostoja
Figure 8. Map of the water level sensor measuring points and readings

Kibana Dashboard je popularna aplikacija za vizualizaciju podataka radi moćnih komponenti prikaza kao što su histogrami, više vrsta grafikona, geografskih mapa itd. Kibana nudi interaktivno kretanje po prikazu podataka, dinamičke radnje nad grafikonima (povlačenje, zumiranje), razne vrste filtera i još mnogo toga (Mohan, 2017). Aplikaciji Kibana pristupamo preko ES domena.

Kibana filterima se odredi što želimo vidjeti na dijagramima panela. U našem slučaju glavni filter je odabir vremenskog okvira za koji želimo vidjeti podatke. Moguće je filtrirati prikaz prema bilo kojem polju ES dokumenta. Na slikama 8. i 9. prikazani su željeni podaci očitanja vrijednosti dvije vrste senzora za odabrani vremenski interval. Vizualizacija podataka urađena je na geografskoj mapi unutar Kibana okruženja.



Slika 9. Prikaz mape sa vrijednostima očitanja senzora mjerenja kvaliteta vazduha
Figure 9. Map of the air quality measurement sensor readings

ZAKLJUČCI

Nestrukturirani podaci i nedostatak obrade u realnom vremenu su neki su od problema kod analize podataka mjernih IoT uređaja. Neki servisi nude preuzimanje i pregled podataka, ali su ograničeni količinom podataka koju mogu da obrade i to uglavnom uz ograničeni broj izvora i formata. Uvidjeli smo da je potreban standardan i pouzdan način za obradu IoT podataka, kao i njihov prikaz u realnom vremenu. Obrada nestrukturiranih podataka zahtjeva korišćenje naprednih algoritama za analizu i prepoznavanje uzoraka u ulaznim podacima, odnosno uplink porukama IoT uređaja. Ako se odlučimo da obogatimo podatke novim informacijama ili kombinujemo podatke iz više izvora kompleksnost se povećava jer je potrebno napraviti različita prilagođenja ulaznih formata podataka. Predstavljena ELK arhitektura efikasno rješava navedene probleme. Sistem je konfigurisan da pronalazi potrebne vrijednosti, vrši indeksiranja, analize i vizualizacije nad događajima u realnom vremenu. ELK predstavlja alternativu u odnosu na konvencionalna rješenja koja po pravilu imaju navedena ograničenja. Prednost ELK je i ta što je sa njim lako raditi i pristupačan je i onima koji nemaju mnogo tehničke stručnosti.

Predložena magistrala za obradu podataka je osnova za budući razvoj sistema u oblaku koji će procesnom snagom, skladišnim kapacitetom, visokim performansama i niskom cijenom postati osnova za IoT. Da bi tako nešto bilo moguće, preduslov je dovoljno velika primjena i razmjera IoT. Naša buduća istraživanja su usmjerena ka razvoju ELK arhitekture na infrastrukturi oblaka, upotrebi Apache Kafka tehnologije za izvor i baferovanje podataka i razvoj novih filtara za obradu podataka iz različitih izvora.

LITERATURA

Tulasi, B., Girish, J.V. (2016). Blending Iot And Big Data Analytics. International Journal Of Engineering Sciences & Research Technology.

Yeo, K. S., Chian, M. C., & Ng, T.C.W. (2014). Internet of Things: Trends, challenges and applications. In *2014 International Symposium on Integrated Circuits (ISIC)* (pp. 568-571). IEEE.

Chhajed, S. (2015). Learning ELK stack: Build mesmerizing visualizations, and analytics from your logs and data using Elasticsearch, Logstash, and Kibana. Birmingham: Packt Publishing Ltd.

Paro, A. (2017). Elasticsearch 5.x Cookbook: Over 170 advanced recipes to search, analyze, deploy, manage, and monitor data effectively with Elastic stack. Birmingham: Packt Publishing Ltd.

Gormley, C., & Tong, Z. (2015). *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.".

Mohan, V. (2017), Elasticsearch Blueprints: A practical project-based guide to generating compelling search solutions using the dynamic and powerful features of Elasticsearch. Birmingham: Published by Packt Publishing Ltd.

APPLICATION OF ELK STACK FOR ANALYSYS AND VISUALIZATION OF EVENTS FROM THE IOT NETWORK

Nebojša Kuduz[1*], Saša Salapura[2]

[1]Mtel, a.d., Banja Luka, Bosnia and Herzegovina, Nebojsa.Kuduz@mtel.ba
[2]University PIM, Faculty of Computer Science, Banja Luka, Bosnia and Herzegovina

ABSTRACT

The paper describes the implementation of an Elastisearch, Logstash and Kibana (ELK) stack to process, analyze and display of IoT data originating from several types of sensors. The prepared Elasticsearch platform collects and integrates real data generated by various sensors for further processing and publishing the desired information. Storing different format inputs, finding and extracting the desired information is a key issue that we needed to solve in our IoT environment. We used Grok filters within Logstash to parse unstructured data and as a result we got structured data in JSON format. Using Logstash allowed us to route filtered data to Elasticsearch. Elasticsearch simplifies the storage of large amounts of records in a database that can be easily searched. Depending on the type of data received from the IoT sensors, we have applied custom filters to analyze the received data, index it and store it in the database. We have presented the relevant information and introduce it to the user using a Kibana interface. By using Kibana, we achieve the near-real-time display of desired analytical information from the IoT network.

The presented concept and data analytics system can be used in more sophisticated IoT networks to automate the generation of required analytics, giving the IoT system the ability to take automated actions based on analyses of collected IoT information.

Key words: elastisearch, logstash, kibana, IoT, sensor.