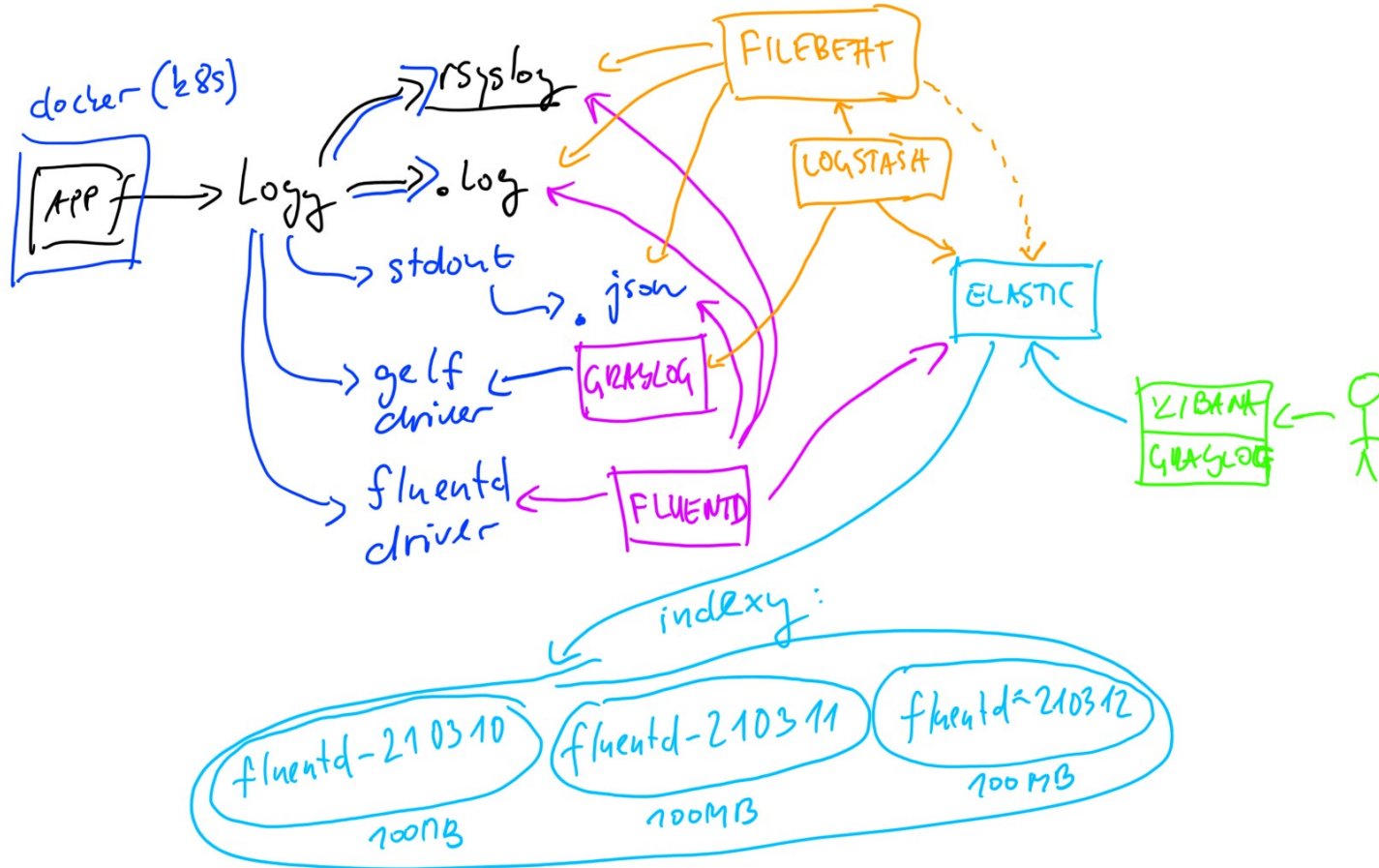
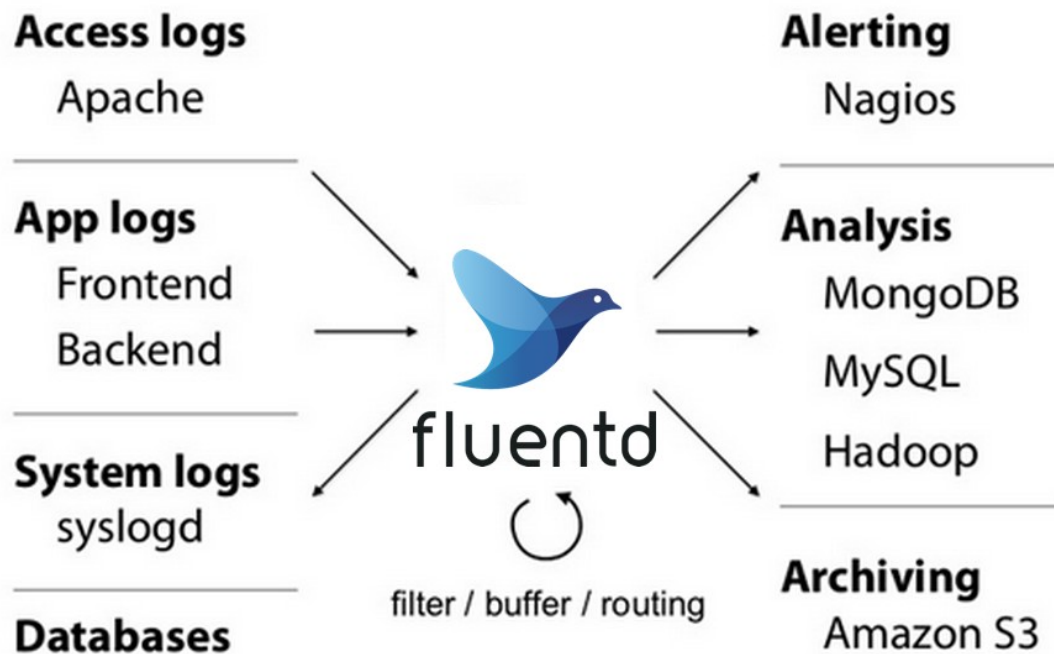


FileBeat / Logstash / Fluentd
+ Elasticsearch + Kibana

ELK stack



Fluentd



Fluentd struktura conf. souboru

- `source` = input (vstup), může být větší množství vstupů
 - `parse` = v jakém formátu jsou vstupní data
- `filter` = filtrace a transformace vstupu, může být větší množství filtrů, na pořadí filtrů záleží!
- `match` = output (výstup), může být větší množství výstupů, na pořadí záleží!
 - <https://docs.fluentd.org/configuration/config-file#note-on-match-order>
 - `buffer` = bufferování dat při výstupu
 - `format` = formátování výstupu
- <https://docs.fluentd.org/configuration/config-file>

Umístění souboru: <https://docs.fluentd.org/configuration/config-file#config-file-location>

Fluentd Tag & Label

- Každý vstup má:
 - Tag: String, ve kterém může být použita tečka, například: `myapp.access`
 - Time: Unix time
 - Record: JSON objekt
- Tagy se typicky používají v: `filter` & `match`, aby se příslušná konfigurace použila pouze pro konkrétní vstup.
- Label je podobný koncept jako tag a umožňuje “fine-grained” nastavení routování uvnitř konfigurace:
 - <https://docs.fluentd.org/configuration/config-file#5-group-filter-and-output-the-label-directive>
 - Existuje speciální label `@ERROR`, který umožňuje routovat Fluentd errorry někam jinam.

Fluentd @include

- Není zapotřebí mít jenom jeden fluent.conf soubor, ve kterém by byla veškerá konfigurace. Je možné mít konfiguraci ve více souborech a pomocí @include ji do fluent.conf připojit:
 - <https://docs.fluentd.org/configuration/config-file#6-reuse-your-config-the-include-directive>

Fluentd + Docker

- Docker obsahuje fluentd log driver, tudíž integrace je velice jednoduchá:
- `docker run --log-driver=fluentd --log-opt fluentd-tag=docker.appname`
 - <https://www.fluentd.org/guides/recipes/docker-logging>
- Pro transformaci log message se dá použít filter @type parser

Fluentd + Kubernetes

- <https://docs.fluentd.org/container-deployment/kubernetes>

Fluentd Grok parser

- Fluentd plugin pro parsování pomocí Grok formátu (nativní formát pro Logstash):
 - <https://github.com/fluent/fluent-plugin-grok-parser>
- Grok online debugger:
 - <https://grokdebug.herokuapp.com/>

Fluentd + multiline

- Fluentd + Docker / k8s + Java aplikace & multiline support:
 - Poznámka: Používá se fluent concat plugin
 - <https://arnoldgalovics.com/java-and-spring-boot-multiline-log-support-for-fluentd-efk-stack/>

Testování regulárních výrazů

- Užitečný online nástroj pro testování regulárních výrazů:
 - <https://regexr.com/>

Docker & Elastic & Kibana

docker-compose.yml:

```
version: '3.7'
services:
  elastic:
    image: elasticsearch:7.3.0
    ports:
      - "9200:9200"
      - "9300:9300"
    environment:
      discovery.type: single-node
  kibana:
    image: kibana:7.3.0
    ports:
      - "5601:5601"
    environment:
      ELASTICSEARCH_HOSTS: http://elastic:9200
```

Elasticsearch & shardy

- Od ES 7 je out-of-the-box 1 primary shard a 1 replika pro každý index (dříve to bylo 5 shardů a také byla 1 replika pro každý index).
- Větší množství replik je pro zrychlení dotazů (dotazy se mohou vykonávat v jakékoli replice) a failover. Počet replik je možné po vytvoření indexu zvyšovat i snižovat.
- Větší množství Shardů je pro zrychlení zápisů. Nepříjemné je, že počet shardů se definuje při vytváření indexu a není možné ho za běhu lehce změnit.
- <https://qbox.io/blog/optimizing-elasticsearch-how-many-shards-per-index>

Elasticsearch

- Elasticsearch obsahuje indexy (v principu index ~ tabulka v relační databázi).
- V indexech se nacházejí dokumenty (dokument ~ řádek v tabulce v relační databázi). Dokumenty jsou obvykle JSON soubory.

Vytvoření indexu (create)

PUT <http://localhost:9200/movies>

```
{
  "mappings" : {
    "properties" : {
      "name" : {
        "type" : "text"
      }
    }
  }
}
```

Poznámka: Při vytváření indexu se mappings nemusí specifikovat, body může být prázdné. Anebo může obsahovat settings, kde se specifikuje počet shardů a replik:

```
{
  "settings": {
    "number_of_shards": 3,
    "number_of_replicas": 2
  }
}
```




Tady budou 3 shardy a každý z nich bude mít 2 repliky

Template

- Index může být také vytvořený na základě šablony (template):
 - <https://www.elastic.co/guide/en/elasticsearch/reference/current/index-templates.html>

Elasticsearch indexy

- Tímto způsobem lze lehce získat seznam všech indexů:
 - GET http://localhost:9200/_cat/indices
 - GET http://localhost:9200/_cat/indices?v  verbose
 - Plus názvy sloupců
- Nastavení indexu (vrací mimo jiné počet shardů a replik):
 - GET http://localhost:9200/movies/_settings

Přidání záznamu (insert)

POST http://localhost:9200/movies/_doc

```
{  
  "name" : "Batman"  
}
```

Poznámka: V odpovědi serveru se nachází atribut `_id`.
Dřív to bývalo číslo, nyní to je například: "mo71NngB_-oJ5tVYwmi0"

Nyní funguje: http://localhost:9200/movies/_doc/{HODNOTA_ID}

Poznámka: Také je k dispozici Bulk API (pomocí něj je možné dělat INSERT, UPDATE i DELETE:

<https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-bulk.html>

Změna záznamu (update)

- Každý dokument má atribut `_version`. Jakmile se provede update stávajícího záznamu, pak se vytvoří nový dokument s incrementovanou hodnotou `_version` a starý dokument je označen ke smazání. Elasticsearch ho někdy později automaticky smaže (je to podobný princip jako Gargage Collection).

```
PUT http://localhost:9200/movies/_doc/mo71NngB_-oJ5tVYwmi0/_update
```

```
{
  "doc" : {
    "name" : "Batman 2"
  }
}
```

Smazání záznamu (delete)

- Smazání záznamu je triviální:

```
DELETE http://localhost:9200/movies/_doc/mo71NngB_-oJ5tVYwmi0
```

Vyhledání záznamu (search)

GET http://localhost:9200/movies/_search

```
{
  "query": {
    "match": {
      "name": "Batman"
    }
  }
}
```

Nebo zjednodušeně (“URI Search”) tímto způsobem:

GET http://localhost:9200/movies/_search?q=name:Batman

Nevýhoda: Před posláním na server se musí provést URL encoding (browser to provede za nás, ale jindy na to nesmíme zapomenout), takže je zapotřebí poslat toto:

http://localhost:9200/movies/_search?q=name%3ABatman

Hodí se to zejména pro jednoduché experimentování

Docker & Filebeat & Elasticsearch

- Jak rozchodit Docker & Filebeat & Elasticsearch:
 - <https://medium.com/@bcoste/powerful-logging-with-docker-filebeat-and-elasticsearch-8ad021aecd87>
 - <https://www.sarulabs.com/post/5/2019-08-12/sending-docker-logs-to-elasticsearch-and-kibana-with-filebeat.html>

Konfigurace lifecycle management I.

- http://localhost:5601/app/kibana#/management/elasticsearch/index_lifecycle_management/policies
 - Nastavení rollover indexů, mazání starých indexů
 - Nastavení v Kibana
- Nebo se dá použít Curator (ale ILM je jednodušší):
 - <https://www.elastic.co/guide/en/elasticsearch/client/curator/current/index.html>

Poznámka: Fluentd elasticsearch plugin v současnosti obsahuje nativní podporu pro ILM:
https://github.com/ukem/fluent-plugin-elasticsearch#enable_ilm

Konfigurace lifecycle management II.

- Low-level způsob konfigurace ILM je následujícím způsobem:

```
PUT _ilm/policy/datastream_policy
{
  "policy": {
    "phases": {
      "hot": {
        "actions": {
          "rollover": {
            "max_size": "50GB",
            "max_age": "30d"
          }
        }
      },
      "delete": {
        "min_age": "90d",
        "actions": {
          "delete": {}
        }
      }
    }
  }
}
```

```
PUT _template/datastream_template
{
  "index_patterns": ["datastream-*"],
  "settings": {
    "number_of_shards": 1,
    "number_of_replicas": 1,
    "index.lifecycle.name": "datastream_policy",
    "index.lifecycle.rollover_alias": "datastream"
  }
}
```


Index Alias

- V Elasticsearch je možné mít nejenom samotné indexy, ale také alias indexů:
 - <https://www.elastic.co/guide/en/elasticsearch/reference/6.8/indices-aliases.html>

Nastavení filebeat & stacktrace (multiline)

- <https://www.elastic.co/guide/en/beats/filebeat/master/multiline-examples.html>

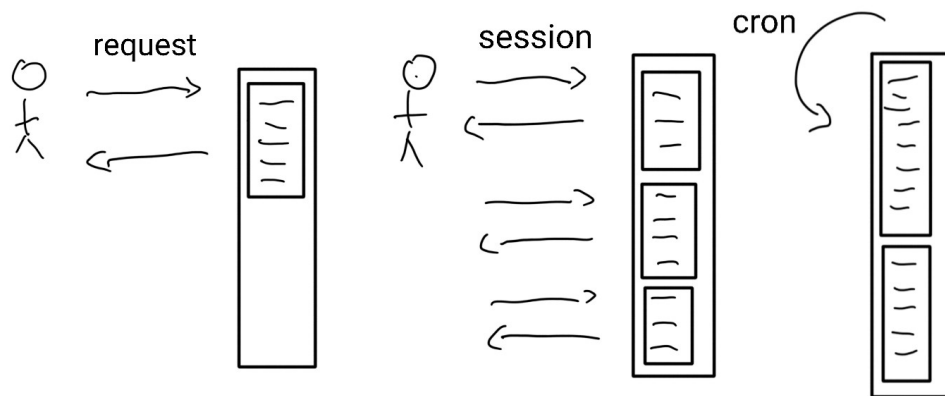
Tip: Kibana LIKE QUERY

- <https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-wildcard-query.html>

```
GET /_search
{
  "query": {
    "wildcard": {
      "user.id": {
        "value": "ki*y",
        "boost": 1.0,
        "rewrite": "constant_score"
      }
    }
  }
}
```

Tip: MDC (Mapped Diagnostic Context)

- Při logování více microservice nebo u reaktivních microservice je užitečné do logu přidat MDC (Mapped Diagnostic Context):
 - <https://dzone.com/articles/mdc-better-way-of-logging-1>
 - <https://spring.io/projects/spring-cloud-sleuth>
- Jinak pro analýzu logů obecně (včetně nebo i bez MDC) je velice užitečný například ELK stack (ElasticSearch & Kibana).
 - Nebo Graylog:
 - <https://www.graylog.org/>
 - Nebo Loki:
 - <https://grafana.com/oss/loki/>



Tip: Exclude Docker Container

- Když se čtou logy z Docker JSONu:
 - <https://www.sarulabs.com/post/5/2019-08-12/sending-docker-logs-to-elasticsearch-and-kibana-with-filebeat.html>
- Pak je možné provést exclude Docker kontejneru ve Filebeat tímto způsobem:
 - https://www.reddit.com/r/elastic/comments/9jt7a8/filebeat_docker_logs_how_can_you_exclude/

Tip: Low disk watermark exceeded on ...

- Na disku ve výchozím nastavení musí být minimálně 85% volného místa, jinak Elasticsearch nebude vytvářet shardy:
 - <https://www.elastic.co/guide/en/elasticsearch/reference/7.8/modules-cluster.html#disk-based-shard-allocation>
 - <https://stackoverflow.com/questions/33369955/low-disk-watermark-exceeded-on>

Elasticsearch cheatsheet

- <https://elasticsearch-cheatsheet.jolicode.com/>