PROJEKT

Multitenantné operačné centrum kybernetickej bezpečnosti riešené
ako otvorená cloudová služba s prvkami strojového učenia

Previazané s balíkom KPB1 – Návrh virt. riešenia

Previazané s výstupom V1 výsledok analýzy dostupných riešení

Financované
Európskou úniou
NextGenerationEU

PLÁN OBNOVY

ÚRAD VLÁDY
SLOVENSKEJ REPUBLIKY

Obsah

MOŽNOSTI RIEŠENIA 3

1 ELK 3

2 Canonical Observability Stack (COS) 4

3 Monasca 5

4 Ceilometer 7

5 Grafana Loki 8

6 Prometheus 11

MOŽNOSTI RIEŠENIA

Boli analyzované viaceré možnosti riešenia. Každému bola venovaná jedna podkapitola v tomto dokumente. Medzi vybrané patrí: ELK, COS, Monasca, Ceilometer, Grafana Loki, Prometheus.

1 ELK

Jedným z populárnych riešení pre centralizované logovanie a monitorovanie je ELK Stack. Jedná sa o kombináciu viacerých systémov zlúčených dokopy. Týmito komponentami sú Elasticsearch, Logstash a Kibana. Dlhú dobu bolo toto riešenie jedným z preferovaných a vybudovalo si komunitu, avšak v poslednej dobe sa dostávajú do popredia aj iné riešenia.

Použité jednotlivé komponenty:

- **Elasticsearch** slúži ako nástroj na vyhľadávanie, analýzu a uchovávanie logov, vyniká rýchlosťou vyhľadávania a schopnosti analyzovania veľkému objemu logov.
- **Logstash** slúži na agregáciu, filtráciu a transformáciu logov z rôznych zdrojov (ako aj OpenStack komponentov) a ich následné odoslanie do Elasticsearch na ukladanie a analýzu. Transformácia logov sa vykonáva takzvaným mapovaním vzorov za pomoci Regex a Grok filtrov.
- **Kibana** poskytuje rozhranie pre vizualizáciu dát uložených v Elasticsearch.

Pri integrácia OpenStack s ELK sú logy odosielané do Logstash. Toto odosielanie sa zabezpečuje buď použitím exportéra ako napríklad filebeat alebo sa logy posielajú cez sieťové protokoly ako syslog. Pre efektívne prijímanie logov sa používajú vstupné pluginy pre Logstash. Ďalej sa logy filtrujú a transformujú za pomoci Regex a Grok vzorov, ktoré majú za úlohu odstránenie nežiaducich informácií. Dokážu pridávať dodatočné metadáta, štítky a celkovú zmenu logov do štruktúrovaného formátu ako napríklad JSON, ktorý je vhodný na uchovanie a analýzu. ElasticSearch sa stará o ukladanie, indexovanie a analýzu logov, pre rýchle vyhľadávanie a efektívne uchovávanie. Dáta sú následne vizualizované v Kibane, kde sa môžu vytvárať nástenky, sledovať výkon, zabezpečenie a dostupnosť služieb.

Výhody integrácie logovania OpenStack s ELK Stack-om:

- **Rýchla analýza a vyhľadávanie:** Elasticsearch umožňuje rýchle vyhľadávanie a analýzu logov, čo uľahčuje identifikáciu problémov a optimalizáciu výkonu sledovanej infraštruktúry.
- **Flexibilná vizualizácia:** Kibana poskytuje širokú škálu nástrojov pre vizualizáciu logov a metrík a zároveň umožňuje prispôsobiť dashboard-y špecifickým potrebám sledovania.
- **Rozšíriteľnosť a škálovateľnosť:** ELK Stack môže byť škálovaný a prispôsobený na spracovanie veľkého množstva dát, čo umožňuje jeho použitie v rozsiahlych cloudových prostrediach.
- **Parsovanie logov:** Transformácia logov pomocou Logstash nám umožňuje pretvoriť zasielané dáta do podoby lepšie čitateľnej pre počítače, čo prináša efektívnejšie vyhľadávanie a analýzu. Mapovací nástroj Grok nám pomáha pridať dodatočné dáta alebo naopak zbaviť sa nadbytočných informácií z logov.

Niektoré z nevýhod:

- **Náročnosť na zdroje:** ELK Stack vyžaduje významné množstvo výpočtových a úložných zdrojov, najmä pri spracovaní veľkých objemov dát.
- **Komplexnosť nasadenia a správy:** Správne nastavenie a optimalizácia ELK Stack-u môžu byť zložité a vyžadujú si pokročilé pochopenie celého systému.
- **Latencia:** Pri spracovaní veľkého množstva logov a nedostatočných zdrojoch a pre vysoké požiadavky tohto systému môže dôjsť k zvýšenej latencii pri vyhľadávaní dát v Kibane.

Napriek týmto výzvam ponúka integrácia ELK Stack-u silný nástroj pre logovanie v OpenStack, pomáha zlepšovať výkon, bezpečnosť a dostupnosť cloudových služieb.

2 Canonical Observability Stack (COS)

COS predstavuje integrované riešenie pre monitorovanie, logovanie a sledovanie služieb v cloudových, ale aj on-premise prostrediach. Je vyvinutý priamo spoločnosťou Canonical, tvorcami MaaS, Juju a charms. COS je navrhnutý tak, aby poskytoval hlboký prehľad o infraštruktúre a aplikáciách s dôrazom na jednoduchosť, automatizáciu a otvorenosť štandardov. Toto riešenie je tvorené niekoľkými otvorenými technológiami spojenými do viacerých charms. COS tvorí monitorovacie riešenie presne pre infraštruktúru nasadenú pomocou Juju a MaaS aby zastrešili širokú škálu monitorovacích technológií.

COS sa skladá z:

Prometheus: zber a ukladanie metrík o výkone a dostupnosti služieb.

Grafana: vizualizačný nástroj pre tvorbu dashboard-ov a alertov z dát zhromaždených Prometheus-om a ďalšími zdrojmi.

Loki: pre agregáciu a efektívne vyhľadávanie v logoch aplikácií a infraštruktúry.

Jaeger a OpenTelemetry: distribuované sledovanie a analýza latencií a výkonnosti služieb.

Grafana agent: exportér logov a metrík.

Tieto komponenty sú integrované do jednotného riešenia charms, ktoré je možné nasadiť ako súčasť Kubernetes clusteru alebo v tradičnejších serverových prostrediach.

Pre integráciu a nasadenie tohto riešenia je potrebná infraštruktúra v Juju s pripraveným Juju cloud, kde budú služby nasadzované. COS podporuje obe možnosti nasadenia v Juju, buď do a bootstrap-ovaného k8s cluster-a, s Juju kontrolórom alebo do machine cloudu. COS Juju charms nasadia všetky potrebné aplikácie ako sú Prometheus, Grafana, Loki a Jaeger. Potom sa vykoná potrebná konfigurácia v Juju charms, vytvoria sa relácie, nasadia sa exportéri pre odosielania metrík a logov. Taktiež po vytvorení relácií s Grafanou, sa v nej vytvoria špecifické dashboard-y pre vizualizáciu metrík a logov, a sú nakonfigurujú sa alerting pravidlá. Pre monitorovanie aplikácií je potrebné COS stack integrovať pomocou Juju add-relation a to konkrétne prepojením grafana-agent charm-u s aplikáciami ktoré chceme monitorovať. Grafana-agent charm sa po vytvorení vzťahu s danou aplikáciu nainštaluje k aplikácií ako subordinate. Subordinate charm/aplikácia má za úlohu bežať po boku druhej aplikácie. Je to

typ charm-u presne stvorený pre logovanie a monitorovanie. Grafana-agent sa nachádza pri aplikácií počas jej celého životného cyklu, pri škálovaní, zmenšovaní atď.

Výhody integrácie COS:

- **Použitie charms:** Podporenie celistvosti a jednotnosti riešenia, keďže celá doterajšia infraštruktúra OpenStacku je už vybudovaná za pomoci Juju a charm.
- **Otvorené štandardy a technológie:** Využíva populárne a overené open-source riešenia, čo zabezpečuje flexibilitu, rozšíriteľnosť a komunitu.
- **Integrácia:** Prirodzená, vstavaná integrácia pre Kubernetes alebo machine cloud a podpora pre cloud-natívne aplikácie.
- **Jednoduchá škálovateľnosť:** COS komponenty podporujú jednoduché horizontálne škálovanie.

Nevýhody:

- **Komplexnosť:** Integrácia viacerých komponentov vyžaduje pokročilé technické znalosti, plus obsahuje technológie navyše ako napríklad Jaeger a OpenTelemetry.
- **Slabá dokumentácia:** Predstavuje dosť závažný problém pre nových používateľov. Juju a charms sú peknými nástrojmi pre správu a nasadenie aplikácií, avšak nie moc otestovanými respektíve zdokumentovanými, čím znemožňujú využitie ich plného potenciálu.

Canonical Observability Stack predstavuje silné a flexibilné riešenie pre monitorovanie, logovanie infraštruktúry a aplikácií. Jeho integrácia open-source nástrojov ako Prometheus, Grafana, Loki a Jaeger ponúka celistvý prehľad o výkone, bezpečnosti a dostupnosti služieb. Napriek istej zložitosti nasadenia a potrebe škálovania zdrojov podľa požiadaviek, COS poskytuje dobrý základ pre efektívne monitorovanie v moderných cloud-natívnych aplikáciách ako aj v tradičných IT prostrediach. Veľkou medzerou však ostáva slabšia komunita a dokumentácia.

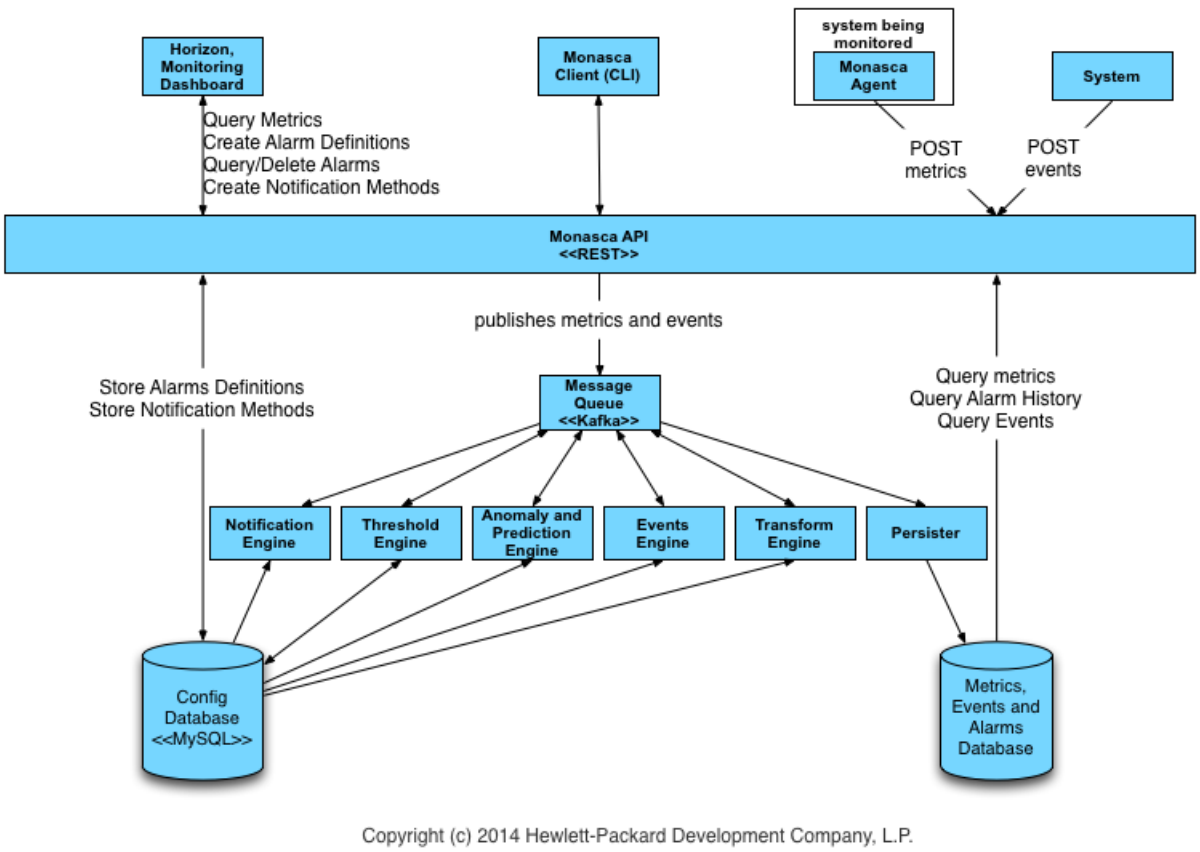
3 Monasca

Monasca je open-source projekt navrhnutý na monitorovanie a logovanie predovšetkým pre OpenStack ale aj iné cloudové prostredia. Ponúka vysokú škálovateľnosť a výkon, je navrhnutý tak, aby efektívne spracovával veľké objemy dát v reálnom čase. Monasca umožňuje užívateľom sledovať zdravie, výkon a dostupnosť cloudových zdrojov a aplikácií, a to všetko prostredníctvom MonascaAPI. Monasca architektúra sa skladá z viacerých hlavných komponentov, ktoré spoločne poskytujú komplexné monitorovacie služby.

Monasca komponenty:

- **Monasca API:** Slúži ako brána pre všetky požiadavky na monitorovanie, umožňuje konfiguráciu alarmov, metrík a notifikácií.

- **Agent Monasca:** Nasadzuje sa na monitorovaných serveroch a zbiera metriky, ktoré potom odosiela do systému prostredníctvom API.
- **Kafka:** Ako sprostredkovateľ správ sa používa na ukladanie a prenos metrík medzi komponentmi. Možná integrácia aj s existujúcim message broker použitým.
- **Monasca Persister:** Prijíma metriky zo systému Kafka a ukladá ich do databázy pre neskoršie spracovanie, do tzv. time series database, ako je napríklad InfluxDB.
- **Keystone:** Vyžaduje autentifikáciu a autorizáciu v OpenStack prostredí.
- **Grafana s pluginom Monasca:** Môže byť použitý na vyhľadávanie a vizualizáciu metrík a alarmov.



Obrázok 1 Monasca architektúra

Monasca podporuje 4 rôzne spôsoby nasadenia pomocou Docker, k8s, ansible a pupet. Pre jej vytvorenie je potrebné prepojiť MonascaAPI s keystone a Apache Kafka, keďže sa jedná o systém priamo vytvorený pre monitorovanie OpenStack. Monasca podporuje prevzatie konfigurácie priamo z databázy akou je napríklad MySql. Ďalej potrebuje nasadenie time series database, tzv. Monasca Persister ako napríklad InfluxDB. Následné prepojenie na grafanu, buď pomocou Grafana pluginu alebo pripojením Grafany priamo na InfluxDB, v ktorej sa nachádzajú monitorovacie dáta pre vizualizáciu a ďalšie spracovanie. Na záver inštalácia Monasca agent na uzloch a VM, z ktorých chceme zasielať monitorovacie dáta.

Výhody:

- **Škálovateľnosť a výkon:** Niektoré spôsoby nasadenia Monasca podporujú aj škálovanie.
- **Integrácia s OpenStack:** Monasca je primárne vyvinutá pre monitorovanie OpenStacku a preto sa ľahko integruje s ostatnými službami a modulmi OpenStack.

Nevýhody:

- **Potreba správy:** Vyžaduje neustálu správu a udržiavanie pre zabezpečenie výkonu a spoľahlivosti.
- **Komunita:** Monasca nepatrí k často náročným riešeniam a od toho sa vyvíja kvalita jej dokumentácie.
- **Neexistujúce charms:** Ako projekt určený priamo pre OpenStack je nevýhodou že pre daný softvér nie je vybudované nasadenie v podobe Juju charms.

Monasca je robustné a škálovateľné riešenie, určené pre monitorovanie v cloudových prostrediach, ktoré je ideálne pre platformu OpenStack. Jej architektúra a komponenty poskytujú detailný prehľad o výkone a zdravotnom stave cloudových zdrojov. Hoci jej nasadenie a správa môžu byť náročnejšie a chýba jej nasadenie pomocou charm-u, v niektorých prípadoch môže byť vhodným nástrojom pre monitorovanie.

4 Ceilometer

Ceilometer je komponent OpenStacku, ktorý poskytuje funkcie zberu údajov o využití zdrojov a monitorovania pre cloudové prostredia. Bol navrhnutý s cieľom umožniť užívateľom a administrátorom získavať detailné informácie o využití zdrojov ich infraštruktúry v cloudových prostrediach. Je navrhnutý na uľahčenie účtovania, benchmarking a plánovanie kapacity. Často sa používa v spojení ostatných modulov, ako napríklad CloudKitty, pre overenie výkonu celej cloudovej infraštruktúry. [12] Ceilometer zaznamenáva informácie o virtuálnych strojoch, úložiskách, sieťových službách a iných zdrojoch, poskytujúc tak bohatý zdroj dát pre analýzu a správu cloudového prostredia.

Ceilometer-u sa skladá:

- **Ceilometer Agent:** Zbiera metriky a údaje o využití z rôznych zdrojov v rámci OpenStacku. Agenti môžu byť nasadení na hypervízoroch, sieťových zariadeniach alebo iných uzloch.
- **Ceilometer Collector:** Prijíma údaje od agentov a preposiela ich do úložiska dát.
- **Datastore:** Úložisko pre metriky a udalosti zozbierané Ceilometer-om. Môže využívať rôzne databázy, napríklad MongoDB alebo MySQL, na uchovávanie údajov. Pri použití novších verzií OpenStacku je odporúčané použitie ďalšieho OpenStack modulu Gnocchi(TSDB).
- **Ceilometer API:** Poskytuje rozhranie pre získavanie údajov a metrík uložených v Ceilometer-i. Umožňuje prístup k zozbieraným dátam pomocou API.

Nasadenie Ceilometer-u do OpenStack charms:

Pre nasadenie Ceilometer-u je potrebné mať k dispozícii Gnocchi TSDB modul ako úložisko dát. Ceilometer je potrebné integrovať, vytvoriť reláciu s keystone, rabbitmq a vyššie spomínaný Gnocchi. Následne je potrebné nasadiť Ceilometer-agent na každom nova-compute uzli a tak isto vytvoriť relácie s message broker, pre interakciu s ostatnými časťami tohto softvéru.

Výhody:

- **Integrácia s OpenStack:** Ako súčasť ekosystému OpenStack, Ceilometer integruje s ostatnými službami ako keystone, rabbitmq a ostatnými nástrojmi tretích strán.
- **Flexibilita:** Podporuje viacero databáz ako úložisko dát, čo umožňuje flexibilné spracovanie údajov. TSDB sa dajú priamo vložiť ako zdroj dát do Grafana dashboard-u pomocou Grafana plugin.
- **Rozšíriteľnosť:** vďaka jeho modulárnej architektúre a možnosti nasadenia pomocou charms je Ceilometer možné jednoducho rozšíriť o nové zdroje dát a metrík.

Nevýhody:

- **Výkon:** Zber a spracovanie veľkého množstva údajov môže zaťažiť systémové zdroje, čo môže ovplyvniť celkový výkon systému a je potrebné myslieť na voľnosť zdrojov ktoré sú poskytované používateľom CC.

Ceilometer je súčasťou monitorovacieho rámca v ekosystéme OpenStack, ktorá poskytuje prehľad o výkonnosti celého cloudu a vytvorených používateľských inštancií vo vnútri cloudu. Dá sa integrovať s ostatnými nástrojmi ako ManageIQ pre zobrazenie použitia zdrojov používateľom. Ceilometer je modulom ktorý sa oplatí nasadiť pre zobrazenie výkonu, spotreby a využitia zdrojov pre koncových používateľov. [13]

5 Grafana Loki

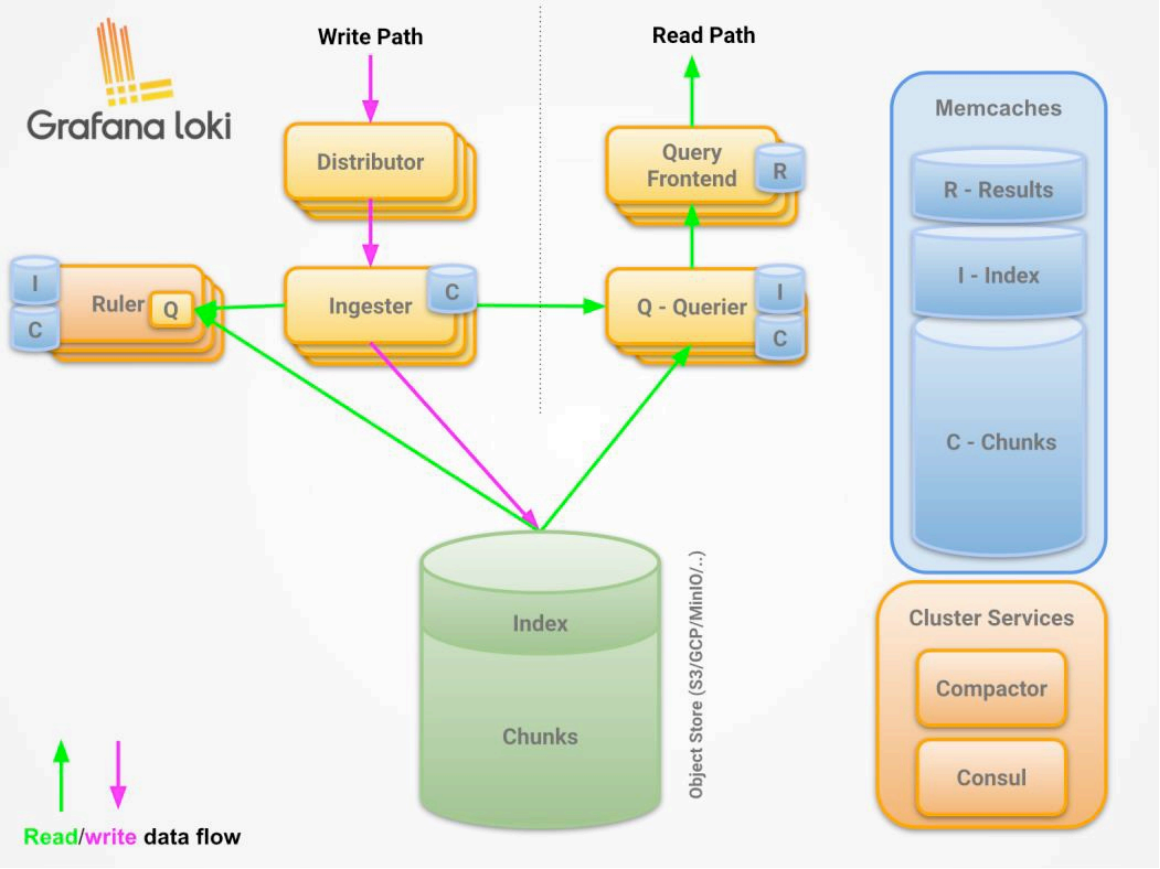
Ako už názov napovedá, Grafana Loki je produkt od spoločnosti, ktorá stojí za vývojom Grafana dashboard-u. Tento logovací systém je škálovateľný, efektívny a jednoducho integrovateľný, čo ho robí ideálnym pre monitorovanie prostredí cloudu a mikroslužieb. Loki sa sústreďuje na efektívne ukladanie a vyhľadávanie logov, pričom kladie dôraz na jednoduchosť, nízke náklady na úložisko a výkon. Inšpirovaný Prometheus-om, Loki využíva rovnaký prístup k značkovaniu dát (labels), čo zjednodušuje prácu s metrikami a logmi v rovnakom ekosystéme.

Komponenty Grafana Loki:

Promtail: Agent nasadený na cieľových zariadeniach na zber logov a ich odosielanie do Loki. Promtail zberá logy a priraďuje im štítky (labels), ktoré pomáhajú pri ich organizácii a vyhľadávaní.

Grafana Loki: Centrálny komponent, ktorý prijíma, ukladá a indexuje logy. Loki je navrhnutý tak, aby bol hlavne efektívny z hľadiska prístupu na zdroje, pri ukladaní a rýchly pri vyhľadávaní.

Grafana: Používa sa na vizualizáciu a vyhľadávanie logov uložených v Loki. Grafana umožňuje užívateľom vyhľadávať logy podľa štítkov a zobraziť ich v kontexte s metrikami z Prometheus-om.



Obrázok 2 Architektúra Grafana Loki

Distributor

Distribútor prijíma logové dáta od klientov a distribuuje ich do viacerých Ingesterov. Distribúcia je založená na konzistentnom hešovaní, čo umožňuje škálovateľnosť a odolnosť systému.

Ingester

Ingester je zodpovedný za spracovanie prichádzajúcich logových dát, ich dočasné ukladanie v pamäti a periodické zapisovanie týchto "chunkov" dát do dlhodobého úložiska, ako sú index a chunks storage.

Ruler

Ruler vyhodnocuje pravidlá definované používateľmi a vytvára upozornenia alebo záznamy na základe týchto pravidiel. Môže tiež priamo vykonávať dopyty na Ingestery alebo na dlhodobé úložisko.

Query Frontend

Query Frontend je vrstva ktorá vykonáva caching a sharding, ktorá optimalizuje vykonávanie dopytov. Do internej pamäti ukladá (kešuje) dopyty a ich výsledky a tiež umožňuje paralelné spracovanie veľkých dopytov rozdelením ich na menšie časti.

Querier

Querier spracováva dopyty na logové dáta. Komunikuje s Ingestermi pre získanie dát z vyrovnávacej pamäte a z úložiska pre staršie dáta. Výsledky dopytov môžu byť použité priamo alebo zobrazené v Grafana.

Memcaches

Memcaches sú použité na kešovanie rôznych častí dát - výsledkov (R), indexov (I) a chunkov (C). Tieto cache zvyšujú výkon systému znížením opakovaného čítania dát z úložiska.

Index

Index ukladá metadata o chunkoch, ako sú identifikátory a časové štítky. Umožňuje rýchle vyhľadávanie relevantných dát pri odpovedaní na dopyty.

Chunks

Chunks obsahujú samotné logované dáta rozdelené buď v segmentoch alebo chunk-och. Tieto chun-ky sú uložené v objektovom úložisku, ako je Amazon S3, Google Cloud Storage, alebo lokálne kompatibilné riešenie ako MinIO.

Nasadenia Grafana Loki:

Grafana Loki je multifunkčne nesaditeľný softvér, podporuje nasadenie pomocou rôznych nástrojov do cloudovej infraštruktúry v kontajnerovej podobe. Grafana Loki je možné nasadiť v rôznych módoch a to v monolitickom, jednoducho škálovateľnom a v mikroservisnom móde. Podporuje konfiguráciu backendového úložiska do S3, do súborov, TSDB a ďalšie. Na základe nasadenie sa líšia aj požiadavky pre backendové úložisko. Aj keď sa líši konfigurácia pri rôznych typoch nasadenia môže líšiť, základ zostáva rovnaký, že Grafana Loki príma, uchováva a vyhľadáva oštítkované logy, ktoré sa vizualizujú najčastejšie za pomoci Grafana dashboard-u. Exportovanie logov z cieľových zariadení zabezpečuje Promtail odosielateľ logov, ktorý tiež zabezpečuje aj ich oštítkovanie. V Grafane sa dá následne Loki pridať ako datasource a pristúpiť k dátam priamo z Grafany pre ich zobrazenie a ďalšie analyzovanie.

Výhody:

- **Efektívnosť nákladov:** Rovnako ako Prometheus Loki minimalizuje náklady na ukladanie logov tým, že ukladá indexy a už komprimované logy oddelene.
- **Integrácia s Grafana:** Jednoduchá integrácia s Grafanou umožňuje jednoduché vyhľadávanie a vizualizáciu logov spolu s metrikami.
- **Jednoduchosť a rýchlosť:** Loki je navrhnutý pre jednoduchosť, rýchlosť a efektívnosť, čo z neho robí atraktívnu voľbu pre riešenia kde sa dbá na efektivitu využitia zdrojov.

- **Komunita a rôznorodosť:** Grafana Loki sa čoraz stáva populárnejším čím sa urýchľuje jeho vývoj, rozšíriteľnosť a pluginy.

Nevýhody:

- **Obmedzené vyhľadávanie logov:** Loki sa zameriava na vyhľadávanie založené na štítkoch, ktoré môže byť pre niektoré prípady použitia obmedzujúce v porovnaní s plne textovým vyhľadávaním.
- **Pochopiteľnosť konfigurácie:** Zložité porozumieť niektorým typom nasadenia, samotný Loki sa skladá z viacerých menších služieb, ktoré sme si vymenovali vyššie.

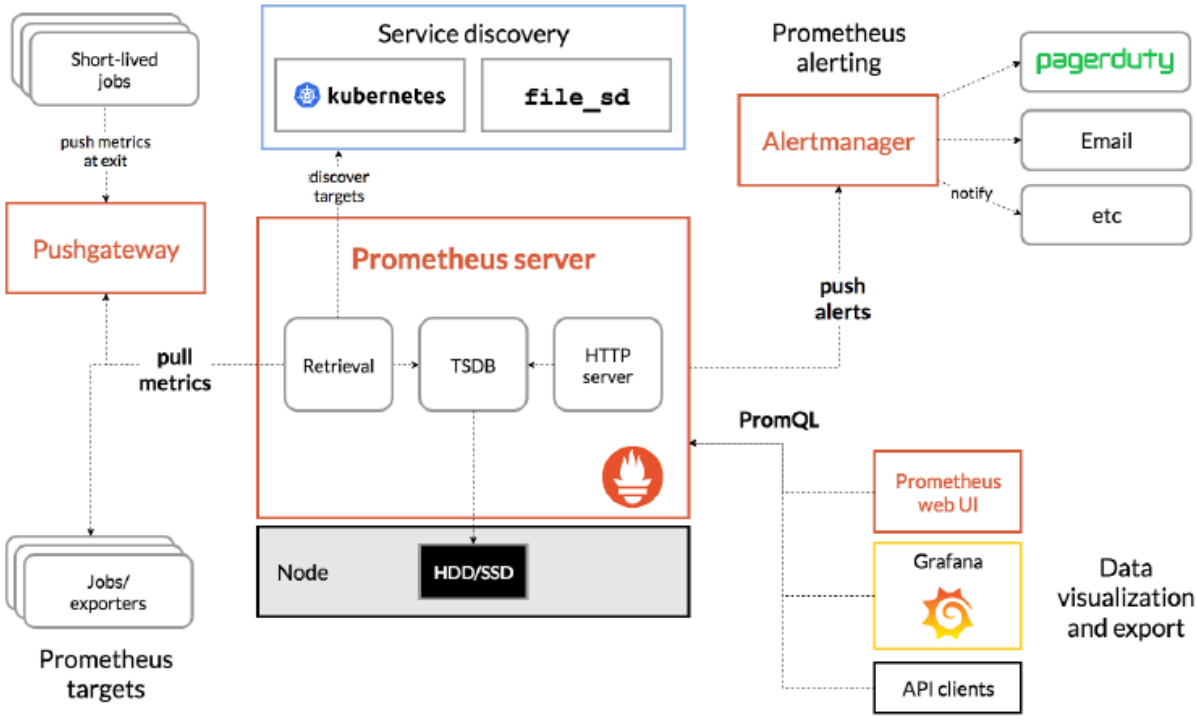
Grafana Loki teda predstavuje vynikajúcu voľbu pre riešenia, ktoré potrebujú efektívny logovací systém, nenáročné na využitie zdrojov a zároveň využívajú Grafanu pre monitorovanie a vizualizáciu metrík. Jeho integrácia s ekosystémom Prometheus, poskytuje silný nástroj pre správu logov pomocou štítkov. Aj keď vyhľadávanie založené výhradne na štítkoch môže byť v niektorých prípadoch obmedzujúce, avšak výhody ktoré Loki ponúka v oblasti jednoduchosti, rýchlosti a integrácie s Grafanou, z neho robia skvelú voľbu pre mnohé riešenia.

6 Prometheus

Prometheus je open-source systém pre monitorovanie a upozorňovanie, ktorý bol pôvodne vyvinutý spoločnosťou SoundCloud. Dnes je široko rozšírený v cloudových prostrediach a mikroslužbách, vďaka svojej výkonnosti, flexibilite a spoľahlivosti. Prometheus umožňuje organizáciám zbierať a spracovať metriky zo širokej škály zdrojov, od operačných systémov po aplikácie bežiace v cloudových alebo kontajnerizovaných prostrediach. S jeho robustným modelom dát a silným jazykom na dopytovanie (PromQL), poskytuje Prometheus hlboký pohľad na výkon aplikácií a infraštruktúry.

Prometheus komponenty:

- **Prometheus server:** Zbiera metriky z konfigurovaných zdrojov v určených intervaloch, ukladá ich a vykonáva pravidlá na vyhodnotenie upozornení.
- **Klientské knižnice:** Poskytujú rozhrania pre aplikácie na odosielanie metrík do Prometheus-a.
- **Push gateway:** Umožňuje dočasné ukladanie metrík pre úlohy, ktoré existujú iba krátkodobo.
- **Exportéry:** Zbierajú metriky zo štandardných služieb alebo serverov a prekladajú ich do formátu, vhodným pre spracovanie Prometheus-om.
- **Alertmanager:** Spracováva upozornenia vygenerované definovanými pravidlami v Prometheus serveri a zabezpečuje ich doručenie prostredníctvom kanála tzv. web hook.



Obrázok 3 Prometheus architektúra

Nasadenie Prometheus:

Je možné nasadenie do kontajnerov alebo na bare metal a to inštaláciou a konfiguráciou Prometheus serveru. Ďalej inštaláciou potrebných exportérov pre monitorovanie špecifických informácií. Taktiež Prometheus zahŕňa použitie Alertmanager-a ktorý sa nakonfiguruje pre zozbierané špecifické metriky, nastavia sa kritické hodnoty a aplikácie, kde chceme odosielať notifikácie. Prometheus ponúka taktiež jednoduché webové rozhranie, kde je možné otestovať, zobraziť zozbierané metriky a overiť ostatné nastavenia, avšak toto rozhranie je jednoduché a na hlbšiu analýzu sa dáta zobrazujú za pomoci Grafany.

Výhody:

- **Multidimenzionálne dáta:** Prometheus ukladá metriky so štítkami, čiže umožňuje identifikovať metriky s názvom a pármi kľúč/hodnota.
- **Aktívny pull model:** Prometheus aktívne zbiera metriky zo svojich cieľov, čo zjednodušuje správu, zabezpečuje konzistentnosť dát a odľahčuje využitie zdrojov svojich cieľov.
- **Silný jazyk na dopytovanie (PromQL):** Umožňuje komplexné dopyty na metriky a ich agregáciu.
- **Vstavané upozorňovanie:** Integrácia upozorňovania priamo v Prometheus eliminuje potrebu externých služieb na monitorovanie a alarmovanie.
- **Automatické objavovanie cieľov:** Využitie v cloudových infraštruktúrach ako napríklad k8s.

Nevýhody:

- **Obmedzené dlhodobé ukladanie dát:** Bez integrácie s externým úložiskom môže byť dlhodobé ukladanie dát v Prometheus náročné.

Prometheus je vysoko výkonný a flexibilný nástroj pre monitorovanie a upozorňovanie, ktorý sa stal štandardom v mnohých cloudových a mikroslužbových architektúrach. Jeho silné stránky, ako sú multidimenzionálny dátový model, vlastný jazyk na dopytovanie a vstavaný systém upozorňovania z neho robia ideálne riešenie pre moderné monitorovacie požiadavky. Napriek niektorým výzvam, ako je dlhodobé ukladanie dát a zložitosť správy, ponúka Prometheus robustné riešenie pre organizácie hľadajúce hlboký náhľad do ich systémov a aplikácií.