Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informatiky a kvantitativních metod

Zpracování nestrukturovaných logů aplikace významného mobilního operátora

Diplomosvá práce

Autor: Bc. Monika Livarová

Studijní obor: Informační management

Vedoucí práce: Ing. Barbora Tesařová, Ph.D.

Odborní konzultanti: **Ing. Radek Brázda, Ph.D., Ing. Michal Charvát**

NTT Czech Republic

Hradec Králové **listopad 2019**

Prohlášení

Prohlašuji, že jsem diplomovou práci zpracovala samostatně a s použitím uvedené literatury.

V Hradci Králové dne 13.11.2019 Monika Livarová

Poděkování

Na tomto místě bych ráda poděkovala vedoucí diplomové práce, paní Ing. Barboře Tesařové, Ph.D., a odborným konzultantům, panu Ing. Radkovi Brázdovi, Ph.D., a panu Ing. Michalovi Charvátovi, za cenné rady a připomínky, které mi poskytli při tvorbě práce.

Název

Zpracování nestrukturovaných logů aplikace významného mobilního operátora

Anotace

Tato diplomová práce se zabývá zpracováním nestrukturovaných logů produkovaných mobilní aplikací významného českého mobilního operátora. Motivem bylo vybrat vhodnou náhradu existujícího řešení moderním interaktivním nástrojem pro správu logů na základě definovaných požadavků s ohledem na možnost výpočtu metrik v reálném čase a provedení ad-hoc analytických dotazů. Výsledkem je demonstrace zpracování a vizualizace anonymizovaných logů aplikace a srovnání navrženého řešení s tím dosavadním.

Title

Processing of unstructured logs of an application of a big mobile operator

Annotation

This diploma thesis deals with the processing of unstructured logs produced by a mobile application of a major Czech mobile operator. The motive was to select a suitable replacement of the existing solution with a modern interactive log management tool based on defined requirements with regard to the possibility of real-time metric calculation and performing ad-hoc analytical queries. The result is a demonstration of processing and visualization of anonymized application logs and comparison of the proposed solution with the current one.

Obsah

Úvod

Globální rozšíření internetu a rozvoj informačních a komunikačních technologií akceleruje tempo růstu objemu generovaných dat. Tato data jsou produkována v masivním množství v různé struktuře a formátu. Nestrukturovaná data tvoří běžně 80 % všech podnikových dat (1). Přestože informace ukryté v těchto datech by mohly pomoci zodpovědět kritické otázky, firmy je často nechávají bez povšimnutí.

S rychle rostoucím objemem dat se vyvíjejí nové nástroje pro jejich zpracování. Tradiční systémy jsou nahrazovány systémy, jejichž architektura je škálovatelná a schopná se vypořádat s velkými daty. Pomocí nástrojů a technik s přístupem Big Data je možné nestrukturovaná data přeměnit na cenné informace, které by se bez toho z dat jen těžko získávaly.

Právě nestrukturovaná strojově generovaná data vytvořená logováním podnikové infrastruktury představují ukrytou obchodní hodnotu a obrovský potenciál k odhalení aktuálních kritických míst. Logy už dávno neslouží pouze k pojmenování konkrétního evidentního problému, nýbrž mohou hrát klíčovou roli pro kvalifikované obchodní rozhodnutí.

Podle expertů je pro provedení kvalitní analýzy nezbytné nestrukturovaná data převést na data strukturovaná (2) (3) (4). Stěžejní částí tohoto procesu je definování výrazů či masek, které určují, jak má být log logicky rozdělen. Pravidla, jaká data budou jakým způsobem parsována, určují rozměr a kvalitu datové základny sloužící pro analýzu.

Využití nástroje s přístupem Big Data pro zpracování logů aplikace významného mobilního operátora umožní monitorování již v současnosti sledovaných ukazatelů, ale tentokrát v reálném čase s možností náhledu na data z různých perspektiv podle potřeby. Takový nástroj se tedy nebude zabývat pouhým zpracováním dat, ale nabídne i možnost logy dynamicky vizualizovat. Nové řešení by mělo, pokud možno, plně nahradit dosavadní řešení monitoringu logů mobilního operátora.

1 Úvod do problematiky

V této kapitole jsou popsány metody a software pro monitorování aplikací. Aplikační log je rozebrán z hlediska jeho formátu a účelu. Nakonec je vylíčen rozdíl mezi analýzou logů za pomocí tradičních systémů a systémů s přístupem Big Data.

1.1 Monitorovací metody a software

Monitorování aplikací na základě přístupu bíle skříňky (white box), je založená na znalosti vnitřní struktury aplikace (5). Přesně naopak tomu je při monitorování aplikací na přístupu černé skříňky (black box), kdy je analýza prováděná bez znalostí vnitřní struktury aplikace a pouze pomocí interakce s jejím externím prostředím (5), tedy skrze informace o volání služeb (13). Monitorování metodou černé skříňky lze využít k identifikaci příznaků problému, ale nikoliv k identifikaci hlavní příčiny (6). Ta se dá zjistit pomocí metody bílé skříňky.

1.1.1 Monitorovací metody bílé skříňky

Mezi monitorovací metody bílé skříňky podle Cindyho Sridharana (6) patří:

- logy
- metriky
- trasování požadavků.

Log je neměnný záznam diskrétních událostí, ke kterým došlo v některé době. Log obsahuje různá data, která se mohou proměnit v cenné informace. Záleží na tom, jaké otázky budou položeny, a takových otázek může být nekonečně mnoho. (7)

Metriky jsou kvantifikované měření, které se používá pro sledování a porovnávání výkonu nebo jiného aspektu systému (8). Podnikovými metrikami je sledován a vyhodnocován stav konkrétního obchodního procesu (9). Metriky jsou měřeny v časových intervalech a vznikají tzv. časové řady (7).

Tím, že metriky jsou reprezentovány pouze v podobě jednotlivých čísel, lze je ukládat a zpracovávat efektivněji než logy. Po určité době navíc mohou být metriky agregovány na denní nebo týdenní bázi, čímž se objem dat dále sníží. Výhodou metrik oproti logům je, že se zvýšenou systémovou aktivitou se cena metrik nezvyšuje. Ukládání a přenos metrik má tedy konstantní režijní náklady. Parametry jako jsou využití diskového prostoru, složitost zpracování nebo rychlost vizualizace, zůstávají relativně neměnné, což u monitorování logů neplatí. (6)

Distribuované trasování požadavků je metoda pro monitorování aplikací. Využívá se především u systémů, které jsou složeny z mikro služeb (10). Distribuované trasování požadavků pomáhá určit, kde se vyskytují poruchy, a co způsobuje nižší výkonnost systému (10). Základní myšlenkou trasování je určit konkrétní body, které leží na trase při vyřizování žádosti, a tyto body upravit tak, aby si informaci o trasování předávaly

dál (7). Tato metoda je důležitá v oblasti softwarového inženýrství, kde se sledují metriky o řadě důležitých chování v rámci kódu aplikace (10).

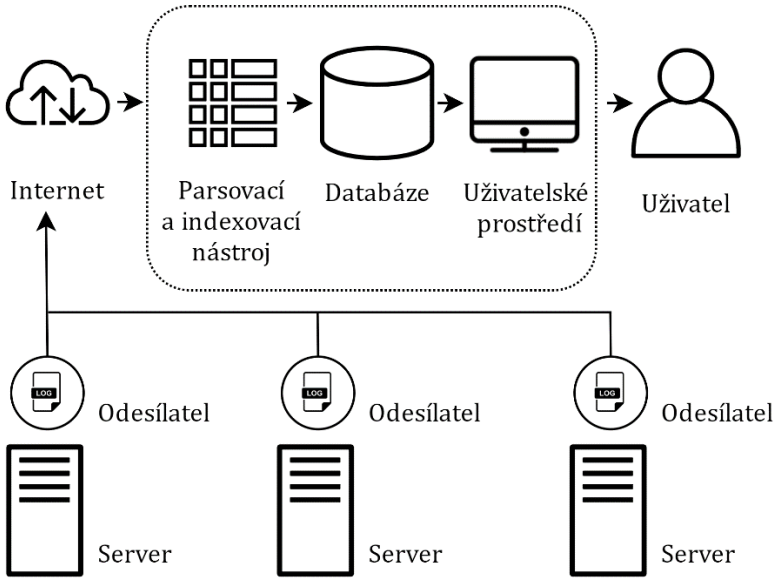
Trasování požadavků i metriky jsou abstraktně řečeno jakousi střechou postavenou nad logy. Trasování požadavků v podstatě podává obecnou informaci z hlediska životního cyklu požadavku. Metriky informují o stavu celého systému, který životní cyklus požadavku převyšuje. Pomocí takto předzpracovaných informací je možné se jednodušeji zaměřit na kritické časy a pomocí informací z logů vyčíst příčinu konkrétního problému. (7)

1.1.2 Monitorovací software bílé skříňky

Je k dispozici celá škála softwarů sloužící pro monitorování aplikací využívajících uvedených monitorovacích metod bílé skříňky. Takovými nástroji jsou:

- nástroj pro správu logů (LM – *Log Management*)
- nástroj pro správu bezpečnostních informací a událostí (SIEM – *Security Information and Event Management*)
- nástroj pro monitorování výkonu aplikací (APM – *Application Performance Monitoring*)
- nástroj pro monitorování systému.

Správa logů je zpracování dat z definovaného kontinuálního zdroje a může pomoci klasifikovat a používat data několika způsoby podle potřeb podniku. Data z logů se mohou indexovat do různých kategorií, z nichž lze vyhodnocovat získané informace. Nástroj pro správu logů také pomáhá logovaná data shromažďovat, ukládat, analyzovat a odstraňovat. (11)



Obrázek 1 Typická moderní architektura nástroje pro správu logů (autor; (4))

LM se vyznačuje tím, že se do něj ukládají všechny logy v surovém formátu, popřípadě jsou parsováním doplněné o některé informace pro dlouhodobější analýzu (12). Nástroje LM podporují fulltextové vyhledávání a události se odlišují pomocí tagy (12).

Analýza s využitím fulltextového vyhledávání v nestrukturovaném textu může být ovšem v případě tisíců a více logů za minutu velmi náročná (4).

Moderní nástroj pro správu logů se typicky sestává z odesílatelů dat na každém uzlu, které posílají logy do centralizovaného serveru (Obrázek 1). Centralizovaný server se poté často skládá z parsovacího a indexovacího nástroje, úložiště a uživatelského prostředí. (4)

Nástroje pro správu bezpečnostních informací a událostí (SIEM) shromažďují logy týkající se zabezpečení s využitím univerzálního formátu a logy jsou řazeny do různých kategorií (12). SIEM nástroje se zaměřují jednak na sběr a ukládání logů za dlouhé období a jednak na analýzu (podezřelých) vzorců chování (13). Tyto dva rozdílné účely vychází ze spojení SIM (Security Information Management) a SEM (Security Event Management) (13).

V SIEM nástrojích jsou informace z logů transformovány statisticky nebo na základě pravidel a informací ze souvisejících logů. Pohledy na data a pravidla pro výstrahy jsou detailnější než v nástrojích LM a jsou prováděny v reálném čase se zaměřením na zabezpečení sledovaného softwaru nebo hardwaru. (12)

Podle Jarno van de Moosdijk a Daan Wagenaar (13) jsou součástí SIEM nástrojů následující pilíře: LM, korelace[1], výstrahy a reakce. LM představuje sběr logů a jejich agregaci v centrálním umístění. K tomu, aby bylo možné zjistit podezřelé chování je někdy nutné korelovat různé logy z různých zdrojů a také je důležité o takové akci informovat – spustit výstrahu. Takové výstrahy mohou navíc vyústit ve vytvoření případu, na který reagují pověření pracovníci (reakce).

Každá firma sledující logy svého produktu potřebuje nástroj pro správu logů, nicméně pro zavedení SIEM je potřeba k tomu upravit produkty, jejich logy, detailně nastavit logovací nástroj, a tomu přizpůsobit chod podniku. Je rozdíl mezi tím se jednou za čas podívat na logy v případě nějakého problému nebo chyby, a kontrolou statistik každý den, případně spolu s procesy v podniku nastavenými tak, že lidé jsou schopní okamžitě reagovat na zprávy o chybách a výstrahách. (12)

Monitorováním se pouze zjistí, co se (právě teď) děje v aplikaci, pomocí LM se zjistí, co se přesně a detailně dělo (14). Většina LM nástrojů umožňuje zobrazení logů, které se zachycenou neobvyklou akcí souvisí, a to například z hlediska času, uživatele, modulu aplikace či typu nebo zdroje události (14).

Nástroje pro monitorování výkonu aplikací (APM) měří dobu odezvy a sledují další komponenty a zdroje, aby pomohly zachovat stabilitu a použitelnost softwaru (15). Nástroje APM poskytují metriky, které odrážejí zážitek koncového uživatele, jako například průměrnou dobu odezvy při špičkovém zatížení nebo metriky související s kapacitou a výkonem zdrojů (16). Většina metrik nástrojů APM není nutně založena na informacích z logů (16). Nástroje APM poskytují náhled na chod softwaru jiným způsobem a na jiné úrovni než logy (16). APM nástroje tak nejsou náhradou systému

[1] Korelace logů je spojení událostí stejné akce (obchodní případ) přes identifikátor akce (8).

pro analýzu problémů na základě agregovaných historických dat logů (16). se Nejčastěji se s APM porovnává distribuované trasování požadavků (10), protože distribuované trasování požadavků je součástí APM (17).

Nástroje monitorující metriky nemají jednotný používaný název. Nástroje tohoto typu monitorují metriky uložené v databázích časových řad[2]. Monitorující nástroje mohou sbírat a zobrazovat vlastní (podnikové) metriky. Největším rozdílem mezi APM nástroji a nástroji monitorující vlastní metriky je, že APM nástroje slouží spíše pro identifikování problémů souvisejícím s kódem, kdežto monitorovací nástroje mohou pomoci najít problém v infrastruktuře (17). Navíc monitorovací nástroje umožňují provést nad daty statistické výpočty (17).

Není podmínkou, že jeden nástroj nemůže zastávat funkci více uvedených nástrojů, takže je možné například narazit na nástroj primárně zaměřený na LM, a zároveň umožňující monitorovat metriky. Názvosloví a kategorizace nástrojů monitorující a analyzující aplikace není tak striktní a může na první pohled mást.

1.2 Formát a účel logů

Logy jsou zprávy, které můžou být generovány skoro jakýmkoliv hardwarovým nebo softwarovým výpočetním zařízením. Je to způsob, jak získávat informace o tom, co se děje se zařízeními a programy, nebo jak s nimi uživatel interaguje. Logy mohou pomoci při řešení problémů a chyb a hlídat software nebo hardware z hlediska zabezpečení.

V logu jsou zaznamenávány systémové a uživatelské akce a využívají se tak pro sledování stavu systému a včasné odhalení nežádoucího chování systému (18). Informace o varování systému a programů v lozích jsou důležité pro identifikování problémů, jako jsou například výpadky napájení nebo nedostatek paměti (18). Logy mohou obsahovat informace pro potřeby vývojářů pro možné řešení chyb a informace o výjimečných důležitých událostech především z hlediska zabezpečení systému (3), například podezřelé vzorce chování uživatele, jakož i neobvyklé akce ze strany systémových nebo aplikačních prostředků (16).

Logy se také dají využívat pro kontrolu dodržování určitých interních předpisů (18). Dalším využitím logů je statistická analýza dat, jejíž výsledky lze použít například pro stanovení vzorců chování při používání softwaru, úzkých míst aplikace nebo požadavků na hardware (18). Logy mohou být také důležité při identifikaci konkrétních transakcí a jiných událostí, které vyžadují ověření (16). Stejně tak jsou logy nepostradatelné při sledování historie a vývoje problémů ve fungování systému (16).

Nejpoužívanějšími typy logů pro analýzu jsou protokol webového serveru, síťový protokol, bezpečnostní protokol a systémový log. Většina aplikací generuje k těmto

[2] Databáze časových řad (*time series database*) je systém, který ukládá data časových řad jako dvojici hodnoty a času (176).

speciálním protokolům ještě minimálně jeden typ logu, který se obvykle vypisuje do konzole. (18)

V jakých přesně případech je log vygenerován záleží na zařízení a nastavení jeho protokolového subsystému. Především je při vývoji systému nutné určit, kam budou logy posílány a ukládány. Nejčastější způsob přenášení logů je pomocí Syslog protokolu, který se stal standardem pro zasílání logů a je k dispozici nejen na Unixových systémech. Na operačním systému Windows je možné využít protokolový systém Windows Event Log. Existuje i možnost ukládat logy přímo do databáze. Pro zařízení a aplikace třetích stran je běžné, že používají protokolový systém, který je k dispozici skrze API pro konkrétní vybraný programovací jazyk (jako např. C a Java), případně mají vytvořen protokolový systém vlastní. (3)

Tato práce se zabývá aplikačními logy, takže dále je pod pojmem log myšlen aplikační log, pokud není uvedeno jinak. *Aplikační logy jsou soubory událostí, které jsou zaznamenány softwarovou aplikací. Formát a obsah aplikačního logu určuje spíše vývojář softwarového programu než operační systém* (19). Log bývá ve (semi)strukturovaném formátu (typicky JSON), v binárním formátu nebo jako prostý text, což je nejčastější varianta (6).

Aplikační logy jsou používané převážně pro řešení problémů. Pokud firma neinvestuje do automatizace analýzy logů, je běžnou praxí, že vývojáři analyzují logy ručně. Testeři zase využívají logy k ověření funkčnosti softwaru vzhledem k její specifikaci. V některých případech jsou logy používané k profilování a benchmarkingu. (18)

„Formát a syntaxe určuje, jak jsou zprávy logů vytvořeny, posílány, ukládány, kontrolovány a analyzovány“ (3). Neexistuje žádný formát logů, který by se používal pro všechny typy aplikačních protokolů. Snahou o vytvoření univerzální formátu logu bylo například představení ULM (Universal Logger Message) formátu (20). Nicméně tento standard, který byl navržen Komisí pro technickou stránku internetu (IETF), vypršel v roce 1999 bez přirozeného nástupce (20).

Nejznámějšími formáty logů jsou: W3C ELF, Apache Access Log, Cisco SDEE/CIDEE, ArcSight CEF, Syslog nebo IDMEF (3). Syslog se stal standardním protokolem systémových událostí, nicméně není vhodný pro webové ani síťové protokoly (18). Syslog se nehodí ani pro aplikační logy, které vyžadují strukturální záznam informací (18).

Hojně se využívají nestrukturované logy a jsou nejlepším příkladem užitečných nestrukturovaných informací. Zápis logů v nestrukturovaným formátu je velmi běžný, protože tento způsob logování je pro vývojáře pohodlný a flexibilní (15). Vývojáři je často využívají pro základní ověření funkčnosti, nicméně jako takové bývají často zahozeny (21).

Jak formáty, tak i data ve zprávách logů, se liší a záleží na nastavení jednotlivých protokolových systémů. Nicméně tři informace logy obsahují vždy – datum a čas

vygenerování logu, zdroj systému, ze kterého byly vygenerovány a samotnou zprávu logu. (3)

1.3 Zpracování a analýza logů

V práci se často vyskytují termíny monitorování a analýza, které sice nejsou synonyma, ale jejich význam se velice prolíná. Zatímco monitorování aplikací je spíše spjaté s agregovanými metrikami a pravidly, na jejichž základě jsou spouštěny výstrahy, analýzou se za pomocí několika procesů zjišťují detailnější informace pro vylepšení aplikace či nalezení problému (22).

Obrázek 2 Pyramida pozorovatelnosti (autor; (23))

Dalším velice blízkým termínem je pozorovatelnost (*observability*), která je jakýmsi základním kamenem pro monitoring a analýzu aplikací (Obrázek 2). Pozorovatelnosti je dosaženo tak, že data zpřístupníme softwaru, v kterém je chceme monitorovat. Monitorování stojí pomyslně nad pozorovatelností, a je tím myšleno sbírání a zobrazovaní dat. Až samotná analýza aplikací tvoří onen pomyslný vrchol nad monitorováním a pozorovatelností aplikací. (23)

Někdy jsou nicméně v literatuře pojmy monitoring a analýza událostí logů brány jako synonyma (8). V této práci je význam těchto termínů rozlišován, ale v některých případech nejsou vyjmenované všechny termíny, které by pro daný případ byly použitelné, ale pouze ten nejvýše postavený v pyramidě pozorovatelnosti.

Největším problémem analýzy (ale i monitorování a pozorovatelnosti) logů je nejednotnost formátů logů. Z tohoto důvodu je vyžadována odlišná příprava analýzy logů pro každý jednotlivý typ logu. Časově nejvíce náročná implementace analytických nástrojů je v případě nestrukturovaných logů. (3) Při přípravě dat je nejdůležitější porozumět povaze událostí logů a rozlišit systémové a vlastní části logu (8).

Běžně se logy parsují pomocí regulárních výrazů (4). Rostoucí objem a rozmanitost logů nicméně vyžaduje pro ruční vytváření pravidel a regulárních výrazů vyvinout značné úsilí (4). Nejenže je nutné vynakládat úsilí před samotným zahájením analýzy nestrukturovaných logů, ale i po celou dobu, kdy je software vyvíjen (3). Formát protokolů a jejich obsah se totiž často mění spolu s produktem (18). Nejjednodušším způsobem je analyzovat data pomocí fulltextového vyhledávání, ale i pro tento způsob

je nutná jistá konzistence dat (3). Pro implementace spolehlivé automatizované analýzy je nicméně nutné rozumět syntaxi záznamu události (3).

Podle A. Chulvakin a spol. (3) jsou nejdůležitějšími předpoklady pro úspěšné dolování dat z logů centralizace a normalizace dat. Pro možnost filtrování a sumarizace všech logů zároveň je kriticky důležité mít všechna data na jednom místě, s čímž souvisí jednotný formát informací. Nemusí se jednat o žádný standardní formát, stačí když logy budou mít některá běžná pole společná, jako například: čas, zdroj, cílová destinace, port, typ události atd. I podle M. Du a F. Li (4) je nejdříve nutné logy parsovat, aby se z nich daly efektivně dolovat informace. Na druhou stranu podle P. Pasupuleti a B. S. Purra (21) by analytické rámce měly být navrženy pouze pro specifické otázky.

Odpovědí na to, jak efektivně dolovat informace z nestrukturovaných logů a zároveň neplýtvat zdroji pro nastavení parserů, je metoda automatizovaného parsování logů. Takovou metodou je například trénování statistických modelů na historických datech (24). Nicméně nástroje, které by automatizované parsování implementovaly, zatím chybí (4). Metody pro parsování nestrukturovaných logů do strukturovaných typů a parametrů však existují, příkladem může být například metoda Spell (Streaming Parser for Event Logs) (4). Integrováním automatického parsování Spell do nástroje pro správu logů by takový nástroj byl schopný poskytnout plnohodnotný datový sklad s mnohem sofistikovanějšími operacemi (4).

Další otázkou je určení správného množství informací, které má být zapsáno do logu (25). Pokud by se logovala každá operace, jednak by náklady na posílání, ukládání, indexování a třídění byly vysoké, a jednak by analýza byla náročnější (25). Také je nutné vhodně určit vážnost záznamů a podle toho je označit jako kritické či triviální (26). Z tohoto pohledu je nutná spolupráce mezi vývojáři a analytiky (18).

Navíc každý tým potřebuje získat z logu jiné informace. Bezpečnostní týmy se často zajímají pouze o události související s bezpečnostními událostmi. Informatický tým hledá v lozích informace pro vyřešení problémů. Auditorský tým se zase zajímá o používání citlivých informací. Jiné informace potřebují ke své činnosti manažeři, účetní nebo marketingový tým. (26)

Pokud jsou logovací nástroje vhodně nastaveny a logy správně transformovány, jsou logy připravené pro identifikaci událostí. Je důležité znát cíl, co má být zjištěno a jaké otázky mají být zodpovězeny. Například se dají položit následující typické otázky:

- jaké aktivity byly provedené,
- kdy byly provedené,
- jak dlouho trvaly,
- v jakém pořadí
- a kým nebo čím byly provedeny. (8)

V některých případech jsou místo individuálních odpovědí na otázky vyžadovány agregované nebo statistické informace. Běžně se také při analýze logů používají metody strojového učení, zejména při detekci anomálií. (25)

1.4 Logy a Big Data přístup

Pokud data z logů analyzujeme na unixovém systému, lze data prohledávat, parsovat, stripovat a zpracovávat pomocí příkazů jako jsou *cat*, *tail*, *grep* a programovacích jazyků, jako například *sed*, *awk* nebo *perl* (7). Příkaz *cat*, z anglického *catenate* tedy spojit, slouží ke spojení nebo vypsání obsahu souborů (8). Příkaz *tail* slouží k zobrazení posledních několik řádků (8), příkaz *grep* zase k hledání daných výrazů v textu (9). Nicméně tento způsob práce s logy lze provést pouze za předpokladu znalosti logu a je časově náročný (27). Navíc tato běžná metoda nestačí v případě potřeby více hostitelů nebo několika zdrojů logů a je nutné sáhnout po nástroji, který logy centralizuje (7).

Takové nástroje obvykle zpracování logů provádí pomocí OLAP (Online Analytical Processing) systémů, tzn. zpracování a analýza se děje mimo produkční server (27). Při centralizaci všech strojově generovaných dat je běžně nutné zpracovávat, ukládat a analyzovat velký objem dat, který se neustále a rychle navyšuje. Právě jedním z rapidně rostoucích zdrojů dat jsou informace o provozu strojů, serverů, síťových směrovačů, webových API a aplikací, tedy logy (28). Taková skutečnost vytváří potřebu nástrojů, které jsou určené pro práci s velkými daty (Big Data).

V tradičních řešeních jsou logy ukládány strukturovaně v relačních databázích. Takový přístup nicméně nedovoluje lineárně navyšovat množství ukládaných a analyzovaných různých typů dat. Tradiční datové sklady nejsou navržené tak, aby integrovaly, škálovaly a pracovaly s exponenciálně rostoucím objemem různě strukturovaných dat. Tradiční systémy postrádají schopnost integrovat data z různých zdrojů, což brání k nahlížení na data jako jeden celek. Řešení s přístupem Big Data se liší od toho tradičního především tím, že data jsou ukládána způsobem, který dovoluje ukládat nestrukturovaná data spolu se strukturovanými daty. Tomu odpovídají například NoSQL databáze. (21)

NoSQL databáze jsou novou generací databázových systémů, které se vyznačují tím, že jsou nerelační, distribuované, open-source a horizontálně škálovatelné (29). Hlavní výhody NoSQL databází jsou:

- flexibilní škálovatelnost,
- flexibilní datový model
- a žádné speciální požadavky na jednotlivé uzly (počítače) (28).

Zatímco relační databázové systémy spíše využívají vertikální škálovatelnost (využití výkonnějšího hardware), NoSQL databáze škálují horizontálně (distribuovaní úloh v rámci množiny uzlů – v rámci clusteru). U NoSQL databází je oproti relačním databázovým systémům databázové schéma volné a jeho změna neznamená pro databází významnou zátěž. (28)

Nejčastějším typem NoSQL databází jsou databáze dokumentové[3], které typicky využívají formáty JSON nebo XML (28). Tyto formáty jsou stromové datové struktury

[3] Dokumentové databáze jsou typem databází, které ukládají strukturované dokumenty (28).

obsahující dvojice názvů a hodnot (28). Nejpopulárnějšími dokumentovými databázemi jsou: MongoDB, Elasticsearch, Redis, Amazon DynamoDB, Couchbase nebo Microsoft Azure Cosmos DB (30).

Dokumenty kromě samotných dat obsahují i metadata popisující význam jednotlivých částí (28). Metadata jsou výborným pomocníkem pro znalost původu dat (21). U tradičních systémů je z důvodu neexistence metadat obtížné identifikovat, která data jsou validní pro odpověď na položenou otázku (21).

Charakteristické pro dokumentové databáze je, že

- *různé záznamy mohou mít různé sloupce,*
- *typy hodnot jednotlivých sloupců se mohou pro různé záznamy lišit,*
- *záznamy mohou mít vnořenou strukturu*. (31)

Důležitou součástí dokumentových databází jsou indexy, které slouží pro efektivní vyhodnocování dotazů. Pro fulltextové vyhledávání NoSQL databáze využívají invertovaného indexu. Invertovaný index ukládá prvky textu pro efektivní vyhledávání v nestrukturovaném textu.[4] (28)

Naopak tradiční systémy jsou založené na datovém modelu, který je navržený před tím, než jsou data přijímána, a neumožňují fulltextové vyhledávání dat. Využitím tradičního systému společnosti tak přichází o cenné skryté poznatky. (21)

Zásadní hodnotu, kterou přístup Big Data přináší, je nový pohled na nestrukturovaná data v kombinaci s historickými daty společnosti. Podle P. Pasupuleti a B. S. Purra společnost, která chce získat opravdu všechny výhody přístupu Big Data, potřebuje dvě základní vlastnosti:

- disponovat technologií, která umožňuje získávat, ukládat, kombinovat a obohacovat velké množství nestrukturovaných a strukturovaných dat v nezpracovaném formátu,
- schopnost provádět analýzy v (téměř) reálném čase na těchto obrovských objemech dat. (21)

Systémy s přístupem Big Data dokáží analyzovat data pomocí technik jako je metoda strojového učení (například pomocí algoritmu rozhodovacího stromu nebo algoritmu nejbližšího souseda), data mining nebo deep learning (8). Pomocí Big Data technik lze také vytvářet predikce, jako například predikci časových řad (8). Takové systémy dále nabízí pokročilejší vizualizaci například v podobě teplotních map, tag cloudu (mrak slov), treemapy nebo síťových diagramů (28).

Při využití tradičních systémů nebude společnost schopná najít schovanou hodnotu v datech, která buď neukládá, nebo je vědomě zahazuje. Problémem je, že nikdo neví, jakou hodnotu mají data v době jejich získání, a je možné, že některé otázky, na které by bylo dobré znát odpověď, vyvstanou až časem. (21)

[4] Velmi zjednodušeně je vytvořena tabulka jednotlivých slov textů, a ke každému slovu jsou přiřazeny identifikátory dokumentů, v kterých jsou obsaženy (28).

2 Cíl práce, volba metodologie

Hlavními cíli práce je zmapovat trh s nástroji pro správu logů, navrhnout vhodnou architekturu a implementovat řešení na praktickém příkladu v prostředí mobilního operátora s ohledem na možnost výpočtu metrik a ad-hoc analytických dotazů.

Jsou definovány požadavky na nástroj pro správu logů, které jsou nezbytné pro zpracování konkrétního nestrukturovaného logu mobilní aplikace, a to z pohledu jeho struktury, tak i povahy celého řešení. Tyto požadavky jsou rozděleny na dvě skupiny – základní a ostatní. Základní požadavky představují snadno a rychle identifikovatelné ukazatele. Na základě základních požadavků jsou navrženy běžně rozšířené nástroje pro správu logů. Tyto navržené nástroje jsou zanalyzovány z hlediska splnění ostatních definovaných požadavků. V případě, že existuje více nástrojů, které splňují všechny definované požadavky, je výběr nástrojů učiněn s ohledem na srovnání jejich výhod.

Vybraný nástroj je nainstalován a nakonfigurován na počítači autora a uzpůsoben na míru pro zpracování logů aplikace mobilního operátora. Vybraný nástroj je otestován zpracováním a vizualizací anonymizovaných logů, které autor obdržel od zadavatelské firmy tématu této práce NTT Czech Republic (dále NTT; původně Dimension Data Czech Republic s.r.o.). Rozsah funkcionalit navrženého řešení, které se skládá z otestovaných funkcionalit nástroje a jeho diskutovaného možného rozšíření, je srovnán s rozsahem funkcionalit dosavadního řešení.

3 Dosavadní řešení a definování požadavků

Ve spolupráci s odbornými konzultanty z NTT je popsáno dosavadní řešení správy logů, představena struktura logů z mobilní aplikace a definovány požadavky na nástroj, který logy má zpracovávat a vizualizovat.

3.1 Popis dosavadního řešení

Dosavadní řešení bylo vybudováno primárně za účelem kontroly kvality dodávaných služeb v rámci outsourcing kontraktu mezi NTT a T-Mobile Czech Republic, a.s. (dále TMCZ). Oproti běžnému monitoringu implementuje dohled podnikových procesů, protože služby v rámci outsourcingu měly definovány podnikové parametry, namísto obvyklých hodnot systémových metrik.

Řešení sestává z části sběru a parsování dat s následnou korelační a agregační vrstvou. Datovými vstupy jsou nestrukturované textové soubory, CSV soubory a databáze Oracle. Korelovány jsou záznamy logů z jednoho či více proudů dat[5].

Výstupem jsou podnikové metriky měření podnikových parametrů služeb téměř v reálném čase, jako je doba trvání procesu a jeho spolehlivost v granularitě 5/15 minut dle požadavku. Podnikové metriky jsou dostupné prostřednictvím webového portálu buď interaktivně, nebo formou generovaných reportů zasílaných emailem. Konkrétně se jedná tyto podnikové metriky:

- celkový počet transakcí,
- počet úspěšně dokončených transakcí,
- počet neúspěšně dokončených transakcí,
- počet úspěšně pomalu dokončených transakcí,
- poměr úspěšně dokončených transakcí k celkovému počtu transakcí,
- poměr úspěšně rychle dokončených transakcí k celkovému počtu transakcí,
- a devadesátý pátý percentil doby trvání transakce.

Nedílnou součástí řešení je zasílání výstrah o porušení požadovaných parametrů podnikových metrik do dohledového centra. Výstrahy jsou zasílány pomocí emailu, systémového příkazu nebo SNMP trapů[6].

Později byl systém doplněn o jednoduché open-source OLAP řešení na bázi produktu Pentaho, které poskytuje možnost dynamické analýzy měřených parametrů podle potřeb uživatele namísto klasického a nepružného reportingu. Výhoda dynamické analýzy je chápána především z hlediska snadného posunu v čase a možnosti provedení drill down [7] agregovaných dat do dat detailních. Data jsou agregovaná v granularitě 1 minuta, 5 minut, 15 minut, 1 hodina, 1 týden, 1 měsíc. Možné je

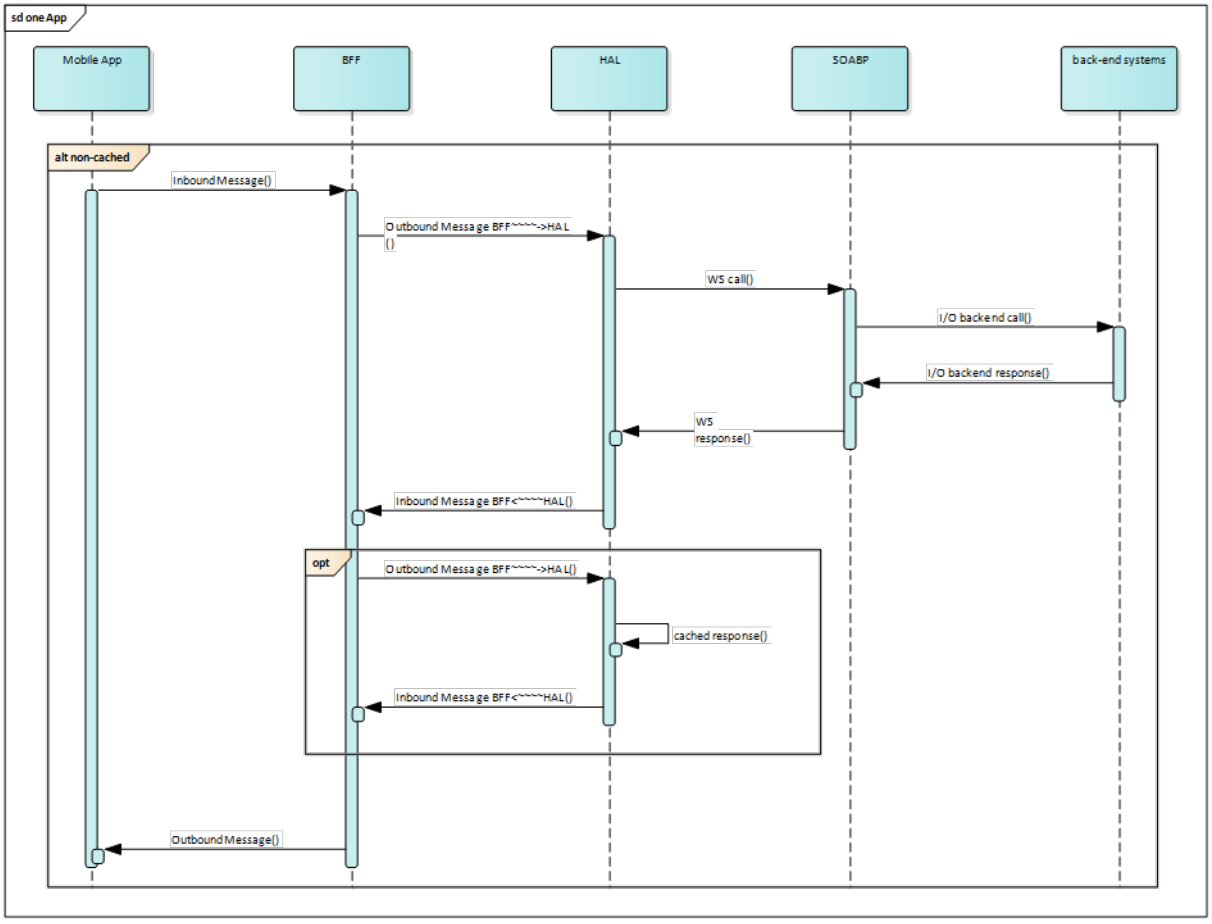
[5] Proud dat je zdroj dat, z něhož data přicházejí ve formě nikdy nekončícího rychlého toku dat (28).
[6] Simple Network Management Protocol (SNMP) je standardní internetový protokol, který poskytuje sadu operací pro sledování a správu síťových zařízení (173).
[7] Drill down je princip zpřístupnění agregovaných dat na vyšší úrovni detailu, s vyšší granularitou (174).

i zobrazit logy z více proudů dat najednou. Reporting je stále využíván a reporty jsou plánovaně odesílané na denní, týdenní a měsíční bázi.

Systém je provozován přes 10 let a tomu odpovídá architektura a povaha webového rozhraní. Zákazník z toho důvodu přemýšlí o jeho náhradě, pokud možno pomocí open-source technologií.

3.2 Popis a struktura logu

OneApp je nová mobilní aplikace samoobsluhy pro zákazníky TMCZ. Aplikační log z této infrastruktury je nestrukturovaný textový dokument. V logu jsou data o voláních a odpovědích front-endové vrstvy nazvané BFF a back-endové vrstvy nazvané HAL (Obrázek 3).



Obrázek 3 Sekvenční diagram monitorování aplikace oneApp (interní zdroj)

V následujícím textu jsou často využívány termíny, jako je záznam, událost, akce a log, a chápány jsou následovně:

- *záznam* je jednotlivý řádek v souboru;
- *událost* je jeden, či více záznamů logicky spojených a zároveň provedených ve stejný časový okamžik;
- *akce* je více událostí korelovaných přes identifikátor;
- a *log* je soubor záznamů.

V logu aplikace oneApp jsou události, které se skládají z více záznamů, ohraničené dlouhou řadou spojovníků (Část kódu 1). Krátká řada spojovníků odděluje některé záznamy uvnitř takové události.

```
Sep  4 22:00:00 hkvnode764.cz.tmo 1 2019-09-04T22:00:00.018+02:00 bff-server-
78fcdcffc9-mx8xt  bff-prod  -  Audit  [mdc@18060  RID="6202436"  UID=""
category="org.apache.cxf.interceptor.LoggingInInterceptor"          exception=""
priority="INFO" thread="http-nio-8080-exec-58"] APP~~~~>BFF - Inbound Message
---------------------------
ID: 18813869
Address: http://t-app.t-mobile.cz/oneapp-bff/api/customerBills/unpaid/summary/
--------------------------------------
Sep  4 22:00:00 hkvnode764.cz.tmo 1 2019-09-04T22:00:00.019+02:00 bff-server-
78fcdcffc9-mx8xt  bff-prod  -  Audit  [mdc@18060  RID="6202436"  UID=""
category="org.apache.cxf.interceptor.LoggingOutInterceptor"          exception=""
priority="INFO" thread="http-nio-8080-exec-58"] BFF~~~~>HAL - Outbound Message
---------------------------
ID: 18813870
--------------------------------------
Sep  4 22:00:00 hkvnode764.cz.tmo 1 2019-09-04T22:00:00.029+02:00 bff-server-
78fcdcffc9-mx8xt  bff-prod  -  Audit  [mdc@18060  RID="6202436"  UID=""
category="org.apache.cxf.interceptor.LoggingInInterceptor"          exception=""
priority="INFO" thread="http-nio-8080-exec-58"] BFF<~~~~HAL - Inbound Message
---------------------------
ID: 18813870
Response-Code: 200
--------------------------------------
Sep  4 22:00:00 hkvnode368.cz.tmo 1 2019-09-04T22:00:00.030+02:00 bff-server-
78fcdcffc9-gcr8t  bff-prod  -  Audit  [mdc@18060  RID="6197790"  UID=""
category="dt.oneapp.bff.config.audit.filters.AuditLoggingInInterceptor"
exception=""          priority="INFO"          thread="http-nio-8080-exec-77"]
{"uri":"http://product-inventory:8080/product-
inventory/v2/products/28717691?fields=id,categories%5Bid%5D,status,statusDescripti
on,suspensionReason,productOffering%5Bid,name%5D","path":"/product-
inventory/v2/products/28717691","httpMethod":"GET","requestHeader":{"Accept":["app
lication/json"],"Authorization":[],"productOffering":null,"suspensionReason":null}
,"responseCode":200,"source":"BFF2HAL","auditProcessingTime":0}
Sep  4 22:00:00 hkvnode764.cz.tmo 1 2019-09-04T22:00:00.031+02:00 bff-server-
78fcdcffc9-mx8xt  bff-prod  -  Audit  [mdc@18060  RID="6202436"  UID=""
category="org.apache.cxf.interceptor.LoggingOutInterceptor"          exception=""
priority="INFO" thread="http-nio-8080-exec-58"] APP<~~~~BFF - Outbound Message
---------------------------
ID: 18813869
Response-Code: 200
```

Část kódu 1 Ukázka anonymizovaného logu aplikace oneApp (interní zdroj)

Log obsahuje informace o tom, kdy a z jaké vrstvy se o požadavek zažádalo/vrátila se odpověď, na jakém serveru a na které instanci aplikace. K takové události se dále vztahují další řádky obsahující identifikační číslo (dále ID). V případě požadavku přímo z aplikace se k takové události vztahuje také řádek s adresou, z které uživatel žádá odpověď. Dále s k žádosti může pojit řádek *payload*, jehož obsah je ve formátu JSON. Na takovém řádku lze najít například informace o verzi aplikace, operačním systému, rozlišení displeje a nastaveném jazyku zařízení uživatele.

K událostem o odpovědi z dané vrstvy je připojen řádek s kódem odpovědi. Kódy 200 a 207 značí, že požadavek se provedl správně, zatímco kódy 500, 401, 400 nebo 429

vypovídají o různých chybách. Například kód 400 představuje nesprávné provedení platby a kód 401 nevalidní přístupový token.

Záznam volání má stejné ID jako jeho odpověď ze stejné vrstvy, takže je možné související události korelovat, a zjistit tím důležité informace jako například dobu, za kterou uživatel dostal odpověď. Konkrétně se jedná o výpočet času mezi požadavkem odeslaným z aplikace (APP~~~~>BFF) a odpovědí odeslanou do aplikace (APP<~~~~BFF). Hodnota pole ID je generována v sekvenci a může být resetována při restartování serveru.

Log obsahuje také řádky, které začínají časovým razítkem, ale nekončí informací o tom, z které a do které vrstvy je informace posílána. Většina takových událostí jsou označeny stupněm *trace* nebo *debug*, které jsou nejpoužívanější při ladění aplikace. Pro analýzu logů jsou cennější logy se stupněm *info*.

Představená struktura logu, je výsledkem anonymizování reálného logu aplikace oneApp. Úplná verze logu má navíc například řádek *headers*. Takový řádek v podstatě obsahuje HTTP parametry, které jsou součástí HTTP žádostí a odpovědí. Na řádku *headers* lze například najít token anebo identifikátor, díky němuž lze propojit data vrstvy BFF a HAL, a tím získat ještě detailnější informace o určité žádosti anebo odpovědi.

3.3 Definování požadavků

V této kapitole jsou definovány požadavky na vybíraný nástroj. Nástroj má umožnit provádět ad-hoc analytické požadavky nad všemi daty z logů a zároveň umět vypočítat a zobrazit vyparsované podnikové metriky z těchto dat. APM a monitorovací nástroje tomuto zadání nevyhovují, protože pracují pouze s metrikami. SIEM nástroj se pro tento případ také nehodí, protože se zaměřuje především na data související se zabezpečením. Zadání nejvíce odpovídá nástroj pro správu logů, který logy ukládá v textovém formátu a umožňuje v nich vyhledávat. Nicméně, musí být vybrán takový nástroj pro správu logu, který bude určen i pro práci s metrikami.

Požadavky na nástroj pro správu logů aplikace oneApp byly definovány na základě dosavadního řešení. Zjednodušeně, nově navržené řešení by mělo umožnit výpočet stejných podnikových metrik v (téměř) reálném čase a zobrazit je v moderním interaktivním uživatelském prostředí běžně rozšířených škálovatelných open-source technologií.

V první řadě by měl nástroj být zdarma. Firma TMCZ si řešení, které je výsledkem této práce neobjednala, a proto není prostor pro finanční náklady. Nicméně nabídnout řešení firmě T-Mobile a nahradit tím stávající řešení firmy NTT v plánu je. Z tohoto důvodu autor této práce při výběru musel zohlednit, aby nástroj byl stále bezplatný i při vysokých objemech dat, která aplikace oneApp generuje. Tímto se zúžil výběr pouze na bezplatné open-source nástroje, protože komerční nástroje mají bezplatnou verzi omezenou právě objemem dat.

Dalším požadavkem je, aby nástroj pracoval jak se systémovými logy tzv. syslogy, tak s logy uloženými v obyčejném textovém souboru. Nástroj by měl umět zpracovat také víceřádkové logy a umět propojit (korelovat) více záznamů (událostí) přes klíč. Požadavkem je také odečítání dvou časových polí od sebe. Nástroj by měl podporovat agregaci dat (výpočty spojené s časovými řady) pro možnost vizualizace podnikových metrik a také by měl umět logy zpracovávat na základě definovaných podmínek.

Nástroj pro správu logů by měl podporovat operační systém Linux, který se na serverech používá nejčastěji. Nástroj by měl mít schopnost pracovat s vysokým neustále rychle rostoucím objemem dat. Mělo by se tedy jednat o škálovatelný nástroj, nejlépe určený pro práci s Big Data. V neposlední řadě by nástroj měl umožňovat vizualizovat data v (téměř) reálném čase. Vizualizace dat by měla být interaktivní a mít moderní vzhled.

Disponováním možnosti zasílaní výstrah v reálném čase na základě určených pravidel by bylo velkým plusem nástroje stejně jako možnosti generování a posílání reportů, ale není podmínkou, aby těmito funkcionalitami byl vybraný nástroj vybaven.

V úvahu pro výběr byly brány pouze běžně rozšířené nástroje, protože open-source nástroje oproti komerčním nedisponují podporou, a tak je při konfiguraci a nastavení nesmírně cenná nejen dokumentace produktu, ale i podpora aktivní komunity na fórech.

Pro zařazení do výběru nástrojů ke srovnání byly definovány základní požadavky, jejichž splnění je snadno a rychle identifikovatelné. Nástroje splňující základní požadavky by měly být:

- open-source,
- je možné je naistalovat na operační systém Linux,
- umožňují analyzovat zpracovaná data téměř v reálném čase
- a mají interaktivní moderní uživatelské prostředí.

4 Výběr nástroje pro správu logů

V rámci této kapitoly je vybrán nástroj pro správu logů, který je schopen zpracovat nestrukturované logy aplikace oneApp. Podle předem definovaných základních požadavků jsou navrženy kandidující nástroje, které jsou srovnány podle všech ostatních určených kritérií. Nejvhodnější nástroj je implementován do testovacího prostředí a je ověřen na anonymizovaných datech aplikace oneApp. Nadto je diskutováno možné rozšíření navrženého řešení.

4.1 Představení navržených nástrojů

V této kapitole jsou představeny nástroje splňující základní požadavky, které jsou definovány v kapitole 3.3. Obecné poznatky o těchto nástrojích jsou shrnuty v následující tabulce (Tabulka 1).

Tabulka 1 Obecné srovnání navržených nástrojů pro správu logů (autor; kapitola 4.1.1.1–4.1.1.4)

Nástroj	*Fluentd*	*Graylog*	*Logstash*	*Syslog-ng*
Licence jádra projektu	Apache v2.0	GPL v3.0	Apache v2.0 (+ Elastic)	GPL v2.O + LGPL v2.1
OS	MacOS/Linux/ Windows	Linux	Linux/Windo ws	Linux /Windows[8] /(MacOS)
Implementace	On premise	On premise	On premise /SaaS[8]	On premise /SaaS[8]
Edice	Open Source	Open Source /Enterprise[8]	Open Source /Basic/Gold[8] /Platinum[8]	Open Source /Premium[8] /SSB[8]
Pluginy	500+	300+	200+	—
Uživatelé podle StackShare	80+	90+	700+	—
Hvězdičky na GitHubu	8 000+	5 000+	10 000+	1 000+
Největší výhoda	Výkonově nenáročný	Uživatelsky jednoduché	Velmi rozšířený	Výkonově méně náročný
Největší nevýhoda	Podmíněné vyhodnocení není plnohodnotné a žádné lokální proměnné	Zdarma jen základní funkcionality	Výkonově náročnější	Některé důležité funkcionality jen v Premium verzi

[8] Platí jen pro placené verze nástrojů.

Běžně rozšířené nástroje pro správu logů, které nebyly navrženy pro srovnání, jsou uvedeny na konci této kapitoly spolu s důvody, proč v úvahu brány nebyly.

4.1.1.1 Obecný popis nástroje Fluentd

Fluentd byl koncipován spoluzakladatelem společnosti Treasure Data v roce 2011, Inc., která je primárním sponzorem tohoto nástroje. Fluentd je plně otevřený software pod licencí Apache 2.0, který sbírá, filtruje a ukládá data do vyrovnávací paměti a data ve výsledném formátu JSON odesílá. (32) Tento nástroj je dostupný pouze v open-source verzi, a to pro Linux, macOS i Windows (33). Komunita Fluentd vytvořila více než 500 pluginů, které spojují několik vstupních datových zdrojů s cílovými destinacemi (32).

K dispozici jsou například vstupní pluginy pro posílání logů z aplikací napsaných v běžně používaných programovacích jazycích (JAVA, .NET, Python atd.), pluginy na získávání záznamů pomocí běžných síťových protokolů (Syslog, HTTP, UDP atd.) a pluginy na sbírání informací z ostatních zdrojů jako jsou Docker, PostgreSQL, nebo například i Twitter. Výstupní pluginy zahrnují nástroje pro správu logů (Elasticsearch + Kibana, Splunk, Sumo Logic), big data platformy (Hadoop DFS, MongoDB), monitorovací systémy (Datadog, Graphite), datové sklady (MySQL, PostgreSQL), notifikační systémy (e-mail, Slack) atd. (34)

Kromě vstupních a výstupních pluginů Fluentd disponuje pluginy pro parsování, úpravu dat a formátu logu (34). Pluginy, které byly vyvinuty předními vývojáři komunity Fluentd nebo společnostmi, které se zavázaly k projektu, jsou označeny jako certifikované. Toto označení má pomoci rozlišit plně funkční pluginy od těch potenciálně nefunkčních nebo těch ve vývoji. (35)

Fluentd je napsaný v jazyku C v kombinaci s jazykem Ruby (35). Právě díky využití jazyku C je tento nástroj nenáročný na systémové prostředky (35). Díky malé paměťové stopě, která je běžně mezi 30 a 40 MB, je možné při sbírání dat z tisíců zařízení v reálném čase najednou ušetřit velké množství paměti (36). Části kódů v jazyku Ruby fungují jako obal, který poskytuje flexibilitu celému řešení (35). K nástroji Fluentd nabízí Treasure Data také odlehčený nástroj – Fluent Bit – který je napsaný pouze v jazyku C, jehož primárním úkolem je posílat data do Fluentd nástrojů, kde se všechna data agregují a dále zpracovávají (35).

Navíc Fluentd podporuje ukládání do vyrovnávací paměti, aby se zabránilo ztrátě dat mezi uzly. Fluentd také podporuje převzetí služeb při selhání (failover) cílící na zajištění vysoké dostupnosti. (32)

Podle StackShare nástroj pro správu logů Fluentd implementovalo 89 firem včetně firmy 9GAG nebo Repro (37). Na GitHubu má Fluentd přes 8 000 hvězdiček a téměř tisíc odštěpených větví projektu (38). Na webové stránce Fluentd je uvedeno více než 5 tisíc společností, které tento nástroj využívá k lepšímu využití a porozumění jejich logů (36). Tyto společnosti nejčastěji vyzdvihávají rychlost implementace nástroje a jeho systémovou nenáročnost a spolehlivost.

Například Operation Management Suite od Microsoftu popisuje, že Fluentd má stovky existujících pluginů, které usnadňují přidání nového zdroje. Společnost Change.org zdůrazňuje, že při implementaci nástroje Fluentd dokázala rychle a snadno navázat několik vstupních a výstupních zdrojů. Spolehlivost a škálovatelnost nástroje Fluentd oceňuje Knowlarity Communications, která analyzuje a notifikuje v reálném čase na základě sbíraných záznamů o hovorech, kterých tato firma za den provede mezi dvěma a deseti milióny. (39)

4.1.1.2 Obecný popis nástroje Graylog

Graylog je open-source nástroj pro správu logů, který sbírá, centralizuje, upravuje, ukládá a vizualizuje data z logů interaktivním moderním způsobem v reálném čase (40). Tím, že Graylog obsahuje celé řešení pro analýzu logů od odesílání až po vizualizaci logů, se od ostatních představených nástrojů liší. Pro konfiguraci, administraci ani analýzu není vyžadována expertní zkušenost, protože vše je přístupné skrze jednoduché uživatelské prostředí (41).

Graylog je vydáván pod GNU v3.0 licencí, je napsaný v Javě, metadata ukládá do MongoDB databáze a logy ukládá do Elasticsearche (42). Na tržišti Graylogu je k dispozici více než 300 pluginů, 80 balíčků, 70 knihoven GELF (Graylog Extended Log Format) a 90 návodů, jak integrovat Graylog s externími systémy a zařízeními (43).

Pluginy slouží pro nastavení vstupů, notifikací, zpracování logů atd. a jsou dostupné na GitHubu. Balíčky obsahují předpřipravenou konfiguraci pro konkrétní účel použití v podobě JSON formátu. Balíček může obsahovat všechno nastavení od zdroje přes pipelinu až po dashboard. Formát GELF je v podstatě komprimovaný JSON (44). GELF je využíván mnoha dostupnými knihovnami, které slouží pro zasílaní strukturovaných logů z aplikací programovaných v běžně používaných jazycích. (45)

Vstupním zdrojem Graylogu může být např. Beats, Logstash, Syslog, JSON nebo textový soubor. V případě, že shromažďování dat z více různých zdrojů je zapotřebí využít tzv. Graylog Sidecar, který slouží pro centralizovanou konfiguraci a správu zdrojů. Nestrukturované logy lze parsovat, upravovat, podle podmínek řadit do kategorií a nepotřebné události ignorovat. Informace z logů se dají vyhledávat a vizualizovat. Data uložená v Graylogu jsou přístupná pomocí REST API pro nástroje třetích stran, takže je možné je zobrazit v jiném nástroji. (46)

Graylog je možné naistalovat na linuxové distribuce (balíčky jsou k dispozici pro Ubuntu, Debian, CentOS a SLES), případně se dá stáhnout jako Docker image nebo jako virtuální zařízení ve formátu OVA (47). Graylog nabízí Open source a Enterprise verzi. Enterprise verze je balíček obsahující Open source verzi spolu s nainstalovaným pluginem Enterprise. Tato verze přináší navíc technickou podporu, uživatelské auditorské logy, nástroj pro přeposílání logů nebo nastavení pravidel výstrah pro neobvyklé shluky událostí, jako například nastavení výstrahy pro situaci, kdy nepřijde žádná úspěšná událost po dobu jedné minuty. Navíc umožňuje vytváření dashboardů, nastavení odesílání reportů v určitém čase nebo uložení starých logů do složek,

z kterých je lze jednoduše zase zpětně importovat do Graylogu. Tato *Enterprise* verze je zdarma, pokud bude Graylog přijímat logy ve velikosti do 5 GB. (48)

Podle StackShare Graylog používá více než 90 firem, včetně firmy Stockopedia (49). Na GitHubu má více než 5 tisíc hvězdiček a více než 700 odštěpených větví (50). Podle oficiální stránky nástroje Graylog je na trhu od roku 2009 a od té doby má na svém kontě už více než 40 tisíc instalací po celém světě (41).

4.1.1.3 Obecný popis nástroje Logstash

Logstash je open-source software, který podporuje vstupy z mnoha běžných zdrojů, a to současně. Logstash data parsuje a transformuje je tak, aby měla jednodušší a čitelnější formát pro další analýzu. Tento nástroj disponuje více než 200 pluginy, které pomáhají s procesy zajišťující vše od integrace různých vstupů, přes filtrování až ke zformátování výstupu do cílové destinace. Logstash umožňuje také vytváření vlastních pluginů nebo upravování těch stávajících. (51) Logstash oficiálně podporuje operační systémy Linux a Windows (52).

Logstash umožňuje jeho výstupy napojit na různé aplikace, nejčastější varianta je Elasticsearch a Kibana, jejichž spojení je nazýváno akronymem ELK. Nyní se už spíše používá název Elastic Stack, protože řešení často obsahují i další nástroje od firmy Elastic. Elastic Stack umožňuje vzít různá data z různých zdrojů a zpracovat, uložit, vyhledat, zanalyzovat a vizualizovat je v reálném čase (53).

Elasticsearch je srdce Elastic Stacku, centrálně ukládá data a obstarává distribuované vyhledávání. Elasticsearch používá standardní REST API a formát JSON, nicméně je možné využit i dedikované API pro některé jazyky, jako je například Java, Python, .NET nebo SQL (54). Kibana dává celému Elastic Stacku uživatelské rozhraní. Je to vizualizační nástroj, který nabízí klasické grafy, geo mapy, časové řady a umí analyzovat závislosti nebo anomálie (55).

Další komponentou ze skupiny Elastic Stack může být nástroj ze skupiny Beats. Ten je jakýmsi odlehčeným odesílatelem dat, který odesílá data z různých zdrojů do Logstash nebo Elasticsearch (56). Konkrétně Filebeat je určený pro přenos textových souborů, Metricbeat pro přenos metrik, Packetbeat pro přenos síťových dat a Winlogbeat přenáší události systému Windows (57).

Logstash je velmi rozšířený mezi firmami a vývojáři. 750 firem na platformě StackShare udalo, že Logstash využívají (58). Mezi ty nejznámější patří například Airbnb, Reddit nebo Bitbucket Na GitHubu má Logstash více než 10 tisíc hvězdiček a téměř 3 tisíce vytvořených vlastních odštěpených větví projektu (59). Logstash má tu výhodu, že na něj lze bez problému navázat další nástroje z rodiny Elastic. Například Fitbit uvádí, že zpracovává až 225 tisíc logů za sekundu a využívá nástroje Elastic Stacku od nasměrování logů až po mapování polí (60).

Logstash a vlastně celý Elastic Stack má více možností předplatného, a to jak na on-premise, tak ve variantě SAAS. Pro on-premise využití jsou k dispozici dvě varianty, které jsou úplně zdarma na neomezenou dobu – Open Source a Basic. Automaticky se

instaluje předplatné s názvem Basic, které má více funkcí než nejzákladnější předplatné Open Source. Funkce, které jsou v základním balíčku Open Source jsou pod Apache licencí verze 2.0 a ostatní funkce, které jsou navíc v předplatném Basic, jsou pod licencí Elastic. (61) Tyto funkce pod licencí Elastic jsou součástí tzv. X-packu a dříve (před verzí 6.3 všech komponent Elastic Stacku) byl X-pack pouze v soukromém repozitáři (62).

Další dvě varianty (Gold a Platinum) jsou placené a umožňují například posílat notifikace do emailu nebo Slacku, mají více možností zabezpečení, jako je vlastní nastavení autentifikace a autentizace, zabezpečení polí nebo šifrovanou komunikaci a řízení přístupu na základě rolí. Dále je v těchto vyšších variantách v Kibaně dostupný modul strojového učení, je možné centralizovaně spravovat procesy v Logstashi a v Beats a k tomu všemu je k dispozici 24/7 podpora. Cloudové varianty jsou samozřejmě placené všechny. (61)

4.1.1.4 Obecný popis nástroje Syslog-ng

Syslog-ng je švýcarský open-source nástroj pro správu logů, který spadá pod organizaci One Identity (63). Syslog-ng sbírá logy z jakéhokoliv zdroje a odesílá je paralelně do určených destinací v téměř reálném čase (64). Nestrukturované logy lze parsovat, formátovat a klasifikovat (64). Z nástroje Syslog-ng je možné upravené či neupravené logy uložit například v Elasticsearch, PostgreSQL, Oracle, MongoDB, Redis atd (63).

Syslog-ng je dostupný ve třech verzích: Open Source, Premium a tzv. SSB. Pouze Open Source verze je zdarma a je vydaná pod kombinací licencí GPL a LGPL (63). Premium verze obsahuje navíc některé funkcionality, jako je posílání a ukládání logů v zašifrované formě (65). SSB verze představuje celé řešení včetně uživatelského rozhraní pro konfiguraci, vyhledávání a vytváření reportů (65).

Syslog-ng v open-source verzi je možné naistalovat z oficiálního repozitáře na linuxové distribuce a jako Docker image (66). Pro instalaci na další operační systémy (např.: macOS) je možné využít instalačních balíčků třetích stran (67). Placené verze lze pak naistalovat z oficiální distribuce i na operační systém Windows (67).

Případové studie na webových stránkách nástroje Syslog-ng poukazují na výhody placených verzí. Například společnosti Tecnocom a Magyar Telekom zvolily Premium verzi a chválí spolehlivost a bezpečnost přenosu logů. Univerzitě Exeter vyhovuje možnost konfigurace v uživatelském prostředí, které nabízí SSB verze nástroje Syslog-ng. Bohužel se žádná případová studie nevěnovala Open Source verzi. (68)

Syslog-ng není uveden na platformě StackShare, takže není možné srovnat jeho počet uživatelů s ostatními nástroji pro správu logů. Stejně tak počet pluginů, kterým Syslog-ng disponuje není z dostupných zdrojů znám. Na GitHubu má Syslog-ng přes 12 tisíc hvězdiček a přes 300 odštěpených větví (66).

4.1.1.5 Ostatní nástroje pro správu logů

R-syslog a Syslog-ng jsou odlišné open-source projekty s odlišnými funkcionalitami a účely, ale vychází se stejného základního démona Syslog. Syslog-ng například oproti R-syslogu umožňuje logy klasifikovat, označovat a korelovat v reálném čase (69). Navíc konfigurační soubor nástroje Syslog-ng je čitelnější, a i jeho dokumentace je úplnější a přehlednější (69).

Během psaní této práce se na trhu objevil back-end inspirovaný nástrojem Prometheus, který je určen pro ukládání logů a jejich vizualizaci v Grafaně. Je nazván Loki a v době psaní práce byl stále v beta verzi. Loki ovšem neumožňuje logy parsovat, ani nepodporuje fulltextové vyhledávání, a nesplňuje tak definované požadavky pro zpracování a analýzu logů aplikace oneApp. (70)

Splunk (71) je nejznámějším komerčním nástrojem pro správu logů a přináší celé řešení pro analýzu logů včetně vizualizace stejně jako Graylog. Celé řešení přináší také například komerční nástroje: Papertrail (72), Logentries (73), Sumo Logic (74), Loggly (75) a Stackify (76). Komerční nástroj OverOps navíc nabízí náhled problémového kódu sledované aplikace přímo z nástroje pro správu logů, konkrétně z nástrojů Kibana, Sumo Logic a Splunk, a umožňuje tak vývojářům zachycené chyby rychle a snadno debugovat (77).

DataDog je také komerční nástroj a umožňuje integraci s více než 350 nástroji včetně datových úložišť, sběračů logů, monitoringu, zabezpečení atd (78). DataDog zaujmul především možností zobrazení souvisejících logů po kliknutí na určitou část vizualizace podnikových metrik.

4.2 Splnění ostatních požadavků nástroji

Jak už bylo uvedeno v předchozí kapitole, všechny detailněji popsané nástroje jsou open-source, dají se nainstalovat na operační systém Linux, umožňují analyzovat zpracovaná data téměř v reálném čase, a to v interaktivním moderním uživatelském prostředí. Splnění či nesplnění ostatních požadavků, jako je možnost

- syslogu a textového souboru jako vstupu,
- zpracování víceřádkového logu,
- propojení více záznamů do jedné události,
- agregování událostí pro zobrazení podnikových metrik,
- podmíněného vyhodnocování při zpracování událostí
- a možnosti odečítání dvou časových polí od sebe,

je uvedeno v následujících podkapitolách.

4.2.1.1 Syslog a textový soubor jako vstup

Syslog je protokol, který se používá k předávání zpráv o systémových událostech. Syslog protokol definuje standardní strukturovaný formát zprávy. Specifikace syslog protokolu i formátu syslog zprávy jsou definované komisí IETF v dokumentech RFC

(Request for Comments). Původní specifikace RFC 3164 byla nahrazená specifikací RFC 5424. (79)

Vstupní data nástroje Graylog můžou být nestrukturované i strukturované. Syslogy můžou být ve formátu RFC 5424 nebo RFC 3164. Nestrukturované logy jsou zpracovávány parserem, který musí být nadefinovaný. Čtení logů ze souborů je možné za pomocí tzv. Graylog Sidecar (80). Sidecar je back-end Graylogu, který je odlehčeným systémem správy konfigurací pro různé sběrače logů (81) a dohlíží na přenos logů do Graylogu (44).

Fluentd také podporuje jako vstupní zdroj syslog ve formátu RFC 3164 i RFC 5424, který je načten přes protokol UDP nebo TCP (82) a také čte textové soubory (83). Rovněž Syslog-ng podporuje jak starší standard syslogů RFC 3164, tak nový standard RFC 5424 a také umožňuje sbírat zprávy logů z textových souborů (84).

Logstash umožňuje mít syslog jako vstupní zdroj a ve výchozím nastavení je podporován pouze syslog RFC 3164 (85). Nicméně za pomocí Grok[9] masek je možné číst a parsovat i nestandardní syslogy (85). Jako vstupní zdroj může být i soubor a logika čtení funguje tak, že program hledá nové soubory a v cyklu je zpracovává (86). Soubory si označuje stavy „sledovaný“, „ignorovaný“, „aktivní“, „uzavřený“ nebo „nezobrazený“ (86).

4.2.1.2 Zpracování víceřádkového logu

Událost, jež je předmětem zájmu, může být rozepsána na více řádcích logu. Nástroj by měl být dostatečně flexibilní na to, aby šlo takové víceřádkové záznamy spojit do jedné ucelené události. Typickým příkladem je, že v logu je první řádek události uveden s časovým razítkem, načež další řádky pouze doplňují tu samou událost o další informace.

Fluentd umožňuje zpracovávat víceřádkové textové soubory (87). Při nastavení parsování víceřádkového logu se nastaví formát počátečního řádku (87), což je možné provést pomocí Grok masek (88).

Samotný Graylog pracovat s víceřádkovými logy neumí, ale při posílání dat do Graylogu z Filebeat (odlehčený odesílatel z Elastic Stacku), je možné nastavit sjednocování více řádků jedné události do jedné zprávy přímo ve Filebeat. Stojí za zmínku, že konfigurace některých programů – například nástroje Filebeat – se provádí pomocí webového uživatelského prostředí již zmiňovaného konfiguračního systému Graylog Sidecar. (81)

V Logstash se ve výchozím nastavení bere jeden řádek jako jedna událost. Pokud by jedna událost měla být přes více řádků, použije se kodek (86). Kodek dekóduje data před jejich dalším zpracováním (86) za pomocí nadefinovaných regulárních výrazů nebo Grok masek (89).

[9] Grok masky jsou sada užitečných výrazů (jako např. používané formáty dat) složených z regulárních výrazů sloužící pro parsování logů v Logstashi (172).

Syslog-ng může pracovat s víceřádkovými logy třemi způsoby. Buď se logy spojí podle odsazení nebo podle prvního řádku anebo podle prvního a posledního řádku. V případě prvních dvou způsobů spojování řádků je každý řádek připojen k nějaké události. Při zvolení třetí metody se může stát, že záznamy, které se budou nacházet mezi řádky, které odpovídají regulárnímu výrazu posledního a prvního řádku, budou zahozeny. (90)

4.2.1.3 Propojení více událostí do jedné události

I mezi ucelenými událostmi lze vytvářet další propojení. Typickým příkladem takového propojení je korelace žádosti a odpovědi na žádost podle stejného identifikátoru akce.

Fluentd umožňuje seskupovat více událostí do jedné události podle klíče události *request_id*. Je nezbytné nastavit pole (nebo více polí), jejichž hodnoty mají být seskupeny. Povinností také je určit pole a jeho hodnotu, která poukáže na ukončující událost. Fluentd uzavře skupinu událostí po určitém čase, pokud neobdrží ukončující událost. (38) Fluentd disponuje dokonce pluginem, který umí zpracovávat víceřádkové logy a zároveň je korelovat do jedné události z více událostí se stejným identifikátorem (91).

Slučování více logů podle identifikátoru není v Graylogu ani ve Filebeat podle zdrojů dostupných v době psaní této práce možné. Na GitHubu je dokonce otevřený námět k vytvoření zmíněné funkcionality v Graylogu z roku 2014 (92).

Logstash umožňuje agregovat události jednoho požadavku podle jakéhokoliv identifikátoru (v dokumentaci je označován jako *task id*). U první události s definovaném *task_id* se vytvoří mapa. Do této mapy se ukládají hodnoty různých polí, která se mohou uložit do další události se stejným identifikátorem. Po přečtení a zpracování události, která je označena jako ukončující, se mapa vymaže. V případě, že se nenalezne ukončující událost s požadovaným *task_id*, mapa se vymaže a nic se nezaznamená. Případně lze vytvořit novou událost s daty z dosavadní mapy. Maximální čas (timeout) mapy lze ručně nastavit, což má smysl hlavně u nedokončených požadavků. (93)

Syslog-ng zvládá propojovat více událostí dvojím způsobem – pomocí identifikátoru nebo pomocí masky. Pomocí identifikátoru se propojení provede obdobně jako u nástrojů Logstash a Fluentd. Identifikátor určí, které události se mají agregovat (94). Také je nutné určit, jaké hodnoty se mají agregovat a timeout (94). Propojování více událostí pomocí masek je velmi podobné první metodě, pouze je tu navíc ještě určená maska pro daný typ události (95). V podstatě se maskou vyfiltrují (určené) stejné události a pak pomocí identifikátoru (klíče) se spojují (95). I v tomto případě je nutné nastavit timeout (95).

4.2.1.4 Agregace dat a podnikové metriky

Agregací dat za určitý interval lze vypočítat různé podnikové metriky. Jsou k tomu využívány základní statistiky, jako například minimum, maximum, průměr, součet

a percentily. Díky vizualizaci podnikových metrik je možné se v datech rychleji zorientovat a snadněji tak pomocí nich identifikovat problémy.

Fluentd umí spočítat všechny jmenované základní statistiky za určitý interval (96). K definovanému intervalu je možné nastavit i maximální počet událostí, z kterých se statistiky mají vypočítat (96). Fluentd lze napojit na Graphite nebo Prometheus (34), což jsou open-source monitorovací nástroje sloužící pro ukládání časových řad.

Podle dostupných zdrojů v době psaní práce Graylog ani Filebeat, který by mohl už zpracovaná data posílat do nástroje Graylog, nepodporují agregaci dat během zpracování. Ovšem data je možné agregovat při jejich vizualizaci v Graylogu nebo pro účely zasílání notifikací (97). Popřípadě pro pokročilé monitorování metrik je možné data z Graylogu posílat do nástrojů monitorování metrik jako jsou Graphite, Prometheus nebo Influxdb (98).

V Logstashi lze agregace, jako jsou počet, minimum, maximum, průměr, odchylka, percentil, spočítat za určitý interval (99). Interval musí vždy být násobkem pěti (99). Logstash také podporuje odesílání metrik do nástrojů, jako jsou Prometheus, Graphite, Influxdb, Riemann nebo DataDog Metrics (100).

Syslog-ng také umožňuje vypočítat minimum, maximum, průměr, součet numerických polí, ale děje se tak při propojování událostí (101). Vzhledem k nemožnosti výpočtu percentilů Syslog-ng nemusí však být uspokojivý při výpočtu metrik. Pro pokročilé monitorování umí i Syslog-ng posílat data do monitoracích nástrojů metrik jako jsou Graphite a Riemann (102).

Elasticsearch, jakožto další úroveň řetězce pro zpracování logů, také umožňuje nad daty provádět agregační operace, jako je výpočet maxima, minima, průměru, součtu nebo percentilů numerických polí. Agregaci dat lze aplikovat pouze pro určitou skupinu událostí dle hodnoty zvoleného pole. (103) V potaz je nicméně nutné brát, že výpočty v Elasticsearchi jsou výkonově náročnější než ty provedené před uložením dat do něj.

I v další úrovni, v Kibaně, je možné data agregovat a vytvářet tak různé grafy vykreslující podnikové metriky v reálném čase (104). Také v tomto případě je však nutné počítat s tím, že se jedná o výkonově náročnější operaci, především při agregaci velkého množství dat.

Několikrát byly zmíněny nástroje pro monitorování metrik jako Graphite (105), Prometheus (106) nebo InfluxDB (107), které slouží primárně pro monitorování systémových metrik a jsou open-source. Do těchto nástrojů lze posílat i podnikové metriky a vizualizovat je buď ve vestavěném uživatelském zobrazení (v případě Graphite a InfluxDB), nebo je zobrazit ve vizualizačním open-source nástroji Grafana (108).

4.2.1.5 Podmíněné vyhodnocování

V mnoha případech je žádoucí, aby při zpracování bylo s daty z logů zacházeno podle pravidel, která si správce systému sám nastaví. Může se jednat o podmíněné akce v podobě transformace, zahození, změnu cílové destinace apod. Podmínky, podle kterých se určuje akce, mohou být vztahovány na zdroj záznamu, jeho označení, nebo na jeho samotnou hodnotu.

Pravidla v Graylogu, která určují, co ze surového logu se před odesláním změní, upraví, zahodí a kam se co pošle, se nastavují v tzv. pipelině. V ní je možné proměnným přiřadit datový typ nebo využít běžných operátorů ke srovnání hodnot. (109) I v nástroji Logstash se dají napsat podmínky, podle kterých se určuje, které události budou zpracovány, a do jakého výstupu budou poslány (110). Také Syslog-ng umí podmínečně vyhodnocovat, používá klasický zápis *if*, *else* a *elif* (111).

Fluentd nedisponuje plnohodnotným podmíněným vyhodnocováním a definováním lokálních proměnných, jako tomu je u ostatních srovnaných nástrojů. Pro odfiltrování hodnot je možné využít plugin, kde se velmi jednoduchým a striktním způsobem definuje podmínka, a pokud je splněna, hodnoty budou zahozeny (112). Dalším pluginem je možné podle zadaných podmínek otagovat události (113) nebo je označit nálepkou (113). Pouze podle těchto tagů a nálepek lze podmíněně zpracovávat události v pipelině (114).

4.2.1.6 Odečítání časových polí

Pokud jsou dvě události sémanticky propojené a doba uplynulá mezi jejich časovými razítky je zajímavá, je nutné data od sebe odečíst.

Podle dostupných zdrojů se zdá, že Fluentd nedisponuje pluginem, který by sloužil pro odečítání časových polí. Pravděpodobně by tedy bylo nutné napsat plugin vlastní nebo odečítání časů provést až na úrovni Kibany. Kibana umožňuje vytvoření tzv. *scripted fields*, což jsou pole nadefinované v Kibaně, s kterými se dá dále pracovat jako s jakýmikoliv jinými poli (115). Samozřejmě je však potřeba počítat s tím, že vizualizace těchto polí je výkonnostně náročnější než vizualizace polí, které jsou do Elasticsearche posílány rovnou z nástroje pro sběr dat (115).

Graylog ve své dokumentaci nepopisuje, jak odečíst dvě časová pole, ale podle příspěvků na komunitním fóru by to mělo být možné (116).

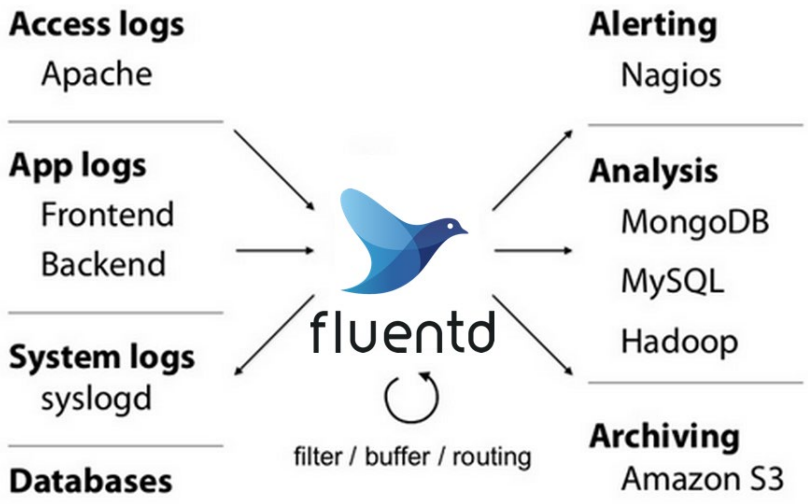
V Logstashi je možné vypočítat dobu mezi dvěma událostmi v případě, že počáteční a konečná událost jsou označeny tagy, a tyto události mají stejné identifikační číslo. Pro otagování událostí je možné použít masky Grok. (117) Doba mezi dvěma událostmi se dá spočítat i tak, že se datum z počáteční události vloží do ukončující události a vypočítá se doba mezi těmito daty pouze v rámci ukončující události (118). Nejjednodušším způsobem je vypočítat rozdíl mezi dvěma časovými poli pomocí pluginu Ruby. Tento plugin umožňuje použít vložený kód v jazyku Ruby a provést tak nespočet možných operací (119).

Stejně jako u nástroje Fluentd, Syslog-ng zřejmě nedisponuje funkcionalitou na odečítání dvou časových polí, jejichž rozdíl by byl schopný uložit do pole nového. V nejaktuálnější dokumentaci k nástroji Syslog-ng ani v jiných dostupných zdrojích v době psaní této práce není taková možnost zmíněna. Avšak s numerickým polem by taková operace možná byla (84).

4.2.1.7 Škálovatelnost

Škálovatelnost se v tomto požadavku chápe jako možnost dodatečné integrace dalších nástrojů, které se dají použít při zpracování rostoucího vysokého objemu dat z více zdrojů.

Fluentd je škálovatelný (Obrázek 4). Tento nástroj ve své odlehčené verzi Fluent Bit pouze data sbírá a zpracovává, ale na rozdíl od plnohodnotného Fluentd je neshromažďuje (120). Tyto dva nástroje je možné kombinovat tak, že Fluent Bit může posílat data z různých zdrojů právě do Fluentd, který data shromažďuje a dále zpracovává (120). Data zpracovaná nástrojem Fluentd se mohou dále ukládat v již zmíněném Elasticsearchi nebo v MongoDB, což je open-source databáze na ukládání semistrukturovaných dat formátem podobných formátu JSON (121). Pro ukládání velkého množství dat je možné použít HDFS (Hadoop Distributed File System) (122) – systém pro ukládání tzv. Big Data. Fluentd je navíc možné pomocí pluginu nastavit tak, aby používal více jader procesoru, což teoreticky umožňuje zpracovat i více než miliardu záznamů za den (123).

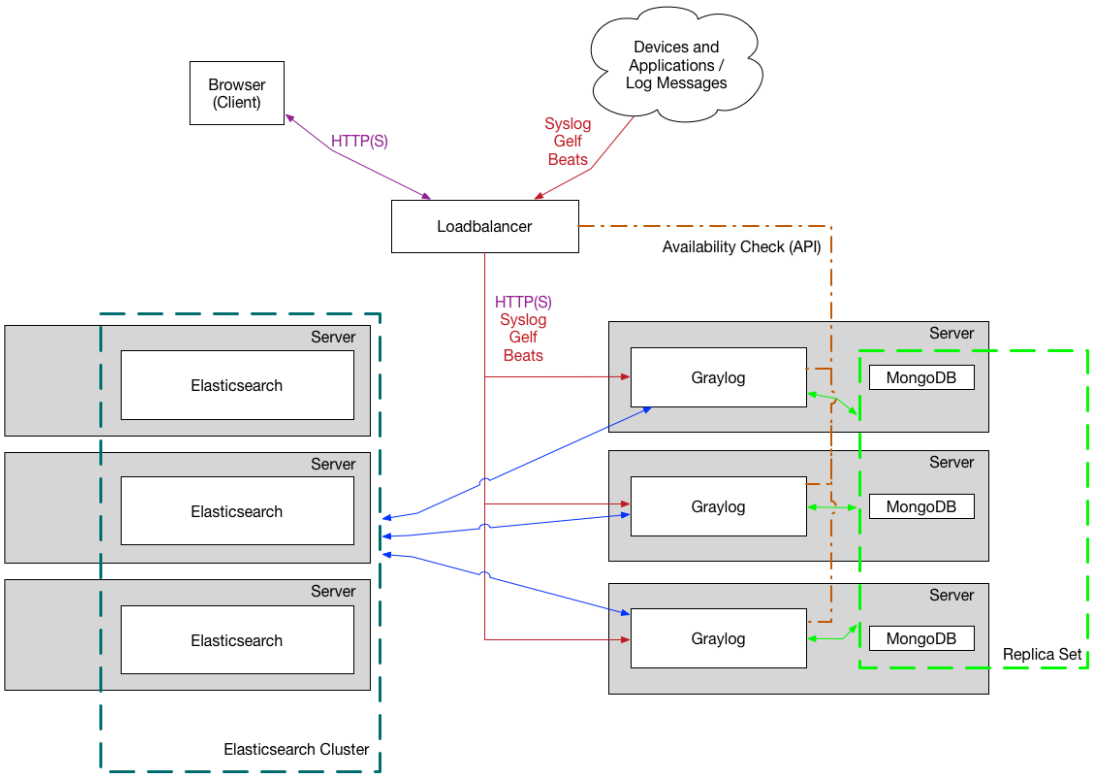


Obrázek 4 Architektura nástroje Fluentd (124)

Každý systém používající Graylog musí sestávat alespoň z minimálně jedné instance Graylog serveru, MongoDB a Elasticsearche a žádná z těchto komponent nesmí být nahrazena jinou technologií. V případě, že je vyžadována vysoká dostupnost pro vysoký objem událostí, měly by se počet instancí jak Graylogu, tak Elasticsearche, navyšovat (Obrázek 5). V případě, že by vyvažovač zátěže zjistil, že některý z uzlů je mrtvý, odstranil by jej z clusteru. V komunitních průvodcích lze najít návod, jak přidat

také funkci zprostředkování zpráv pomocí implementace Apache Kafka nebo RabbitMQ. (125)

Architektura Graylogu sice umožňuje zpracovávat velké množství logů, ale rozsah a flexibilita integrace s ostatními nástroji je v porovnání s ostatními nástroji pro správu logů omezenější.

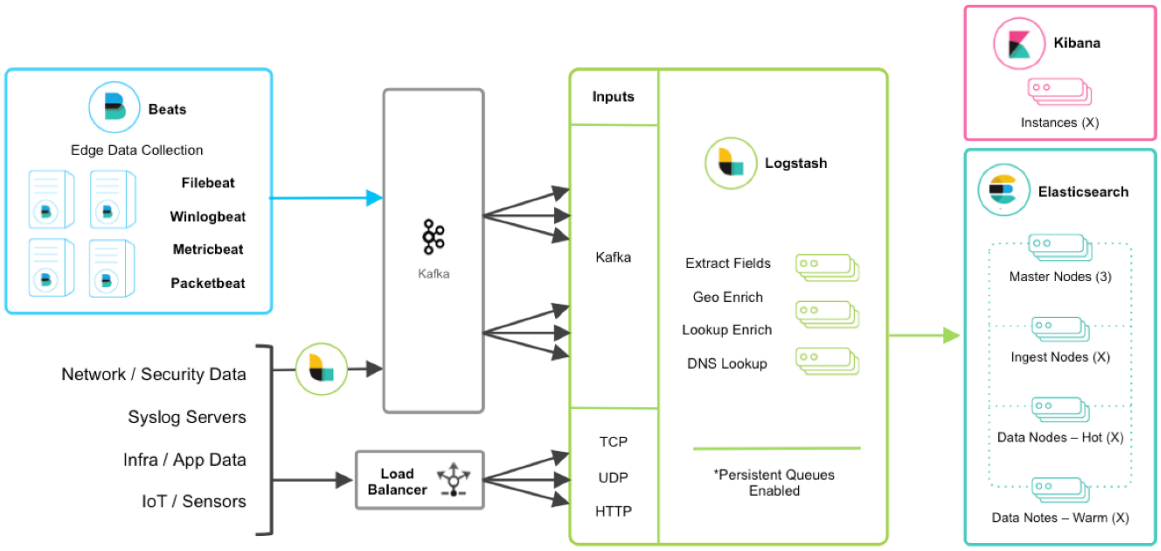


Obrázek 5 Architektura nástroje Graylog (125)

Integrování dalších nástrojů s Logstash je vzhledem k existenci již zmíněného Elastic Stacku přímočaré. Elastic Stack je tím, jak se dá poskládat z více či méně komponent, vysoce škálovatelný (Obrázek 6). K odesílání dat se může využít speciálních odesílačů dat Beats, kteří data sbírají a posílají do Logstashe. Konkrétně Filebeat je určený pro přenos textových souborů, Metricbeat pro přenos metrik, Packetbeat pro přenos síťových dat a Winlogbeat přenáší události systému Windows. Logstash také může poslouchat síťový provoz probíhající skrze různé protokoly, jako například TCP, UDP nebo HTTP. (57)

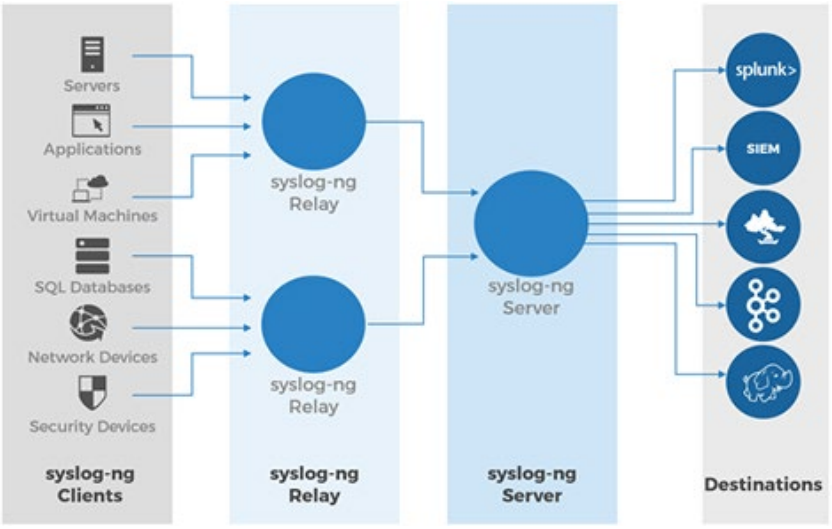
Pokud není potřeba data složitě transformovat, je možné je sbírat nástrojem Beats a rovnou je posílat do Elasticsearche, kde se data indexují a ukládají. Data je také možné sbírat a posílat přes Elasticsearch Ingest – součást Elasticsearch – což je zjednodušená verze Logstashe. Naopak zpracování velkého množství dat je možné distribuovat na více instancí Logstashe, přičemž zpracovaná data lze následně odesílat do jednoho centrálního Logstashe. Do infrastruktury Elastic Stacku lze také přidat nástroje pro zprostředkování zpráv Kafka, Redis nebo RabbitMQ. (57)

Kromě základních komponent architektury Elastic Stacku je možné Logstash integrovat s nástroji pro Big Data HDFS a MongoDB. Dalšími výstupními pluginy pro Logstash jsou například DataDog nebo Google BigQuery. (100)



Obrázek 6 Architektura Elastic Stacku (57)

Syslog-ng může shromažďovat logy téměř v reálném čase na základě prakticky nekonečného počtu pravidel založených na typu zdroje, zdrojové adrese a obsahu zprávy a posílat logy do několika destinací najednou (64). Syslog-ng disponuje odesílatelem nazvaném relé, který odesílá data do centrálního serveru (Obrázek 7). Syslong-ng zaručuje, že všechny zprávy, které dorazily do komponenty zvané relé, budou doručeny do centrálního serveru (126). Syslog-ng může sbírat data z webových serverů, SQL databází a přímo z aplikací nebo zařízení, a posílat je do distribuovaného systému Hadoop nebo do dalších systémů MongoDB, Apache Kafka či Elasticsearch (127).



Obrázek 7 Architektura nástroje Syslog-ng (64)

4.3 Zdůvodnění výběru nástroje

Graylog je jednoduchý uživatelsky přívětivý nástroj. Jeho výhodou je, že ho mohou spravovat i méně technicky zdatní administrátoři. Avšak jeho integrace s ostatními nástroji je omezená a mnoho užitečných funkcionalit je dostupných pouze v placené verzi.

Fluentd a Logstash jsou si nástroje velmi podobné. Oba jsou naprogramované v jazyku Ruby, který je v případě Fluentd doplňován částmi kódu v jazyce C a v případě Logstashe je to Java. Právě implementace jazyka C v nástroji Fluentd je jeho největší výhodou, a to z hlediska rychlosti a nízké náročnosti na paměť. Z dostupných zdrojů však nebylo zjištěno, zda nástroj Fluentd umožňuje při zpracování logu odečíst od sebe dvě časová pole a uložit jejich rozdíl. Fluentd navíc nedisponuje podporou plnohodnotného podmíněného vyhodnocování a lokálních proměnných v nastavení pipeliny, takže se spíše hodí pro analýzu strukturovaných a semistrukturovaných logů.

Stejně tak jako Fluentd i Syslog-ng patrně neumí odečíst od sebe dvě časová pole bez nutnosti tvorby vlastního pluginu. Tato skutečnost byla při rozhodování kritická, a proto byl vybrán pro správu logů aplikace oneApp nástroj Logstash (Tabulka 2).

Tabulka 2 Srovnání požadovaných vlastností nástrojů pro správu logu (autor; kapitola 4.2) [10]

Nástroj	*Fluentd*	*Graylog*	*Logstash*	*Syslog-ng*
Open Source	✓	✓	✓	✓
Linux	✓	✓	✓	✓
Analýza reálném čase	✓	✓	✓	✓
Syslog a textový soubor	✓	✓	✓	✓
Podmínečné vyhodnocení	✓	✓	✓	✓
Agregace dat/metriky	✓	—	✓	✓
Zpracování víceřádkového logu	✓	✓	✓	✓
Propojení událostí (korelace logů)	✓	×	✓	✓
Odečítání času	—	×	✓	—
Škálovatelnost	✓	✓	✓	✓

Všechny popisované nástroje kromě Graylogu nejsou kompletním řešením pro správu logů. Logy pouze zpracovávají, ale neukládají je, ani je nevizualizují. Tudíž je nutné k Logstashi vybrat vhodnou databázi a vizualizační nástroj. Rozšířené open-source vizualizační nástroje určené pro monitoring a analýzu logů existují pouze tři:

[10] Černá fajfka znázorňuje, že nástroj splňuje daný požadavek. Šedivá fajfka znamená, že nástroj splňuje požadavek pouze z části. Vodorovná čárka vyznačuje, že nebylo prokázáno, zda nástroj může splnit daný požadavek a křížek znázorňuje nemožnost požadavek daným nástrojem splnit.

- Grafana,
- Kibana
- a Redash.

Nejznámější je Kibana, která vizualizuje data uložené pouze v Elasticsearchi. Výsledkem spojení Elasticsearche a Kibany je vizualizační nástroj umožňující provedení dynamické analýzy dat pomocí fulltextového vyhledávání. Přestože primárním účelem Elasticsearche je ukládat logy, ukládání a monitorování metrik je také běžnou praxí.

Grafana je primárně určena pro vizualizaci numerických časových řad, tedy dat uložených v databázích časových řad. Grafana podporuje nicméně i jiné typy databází, jako je Elasticsearch nebo novou vlastní databázi určenou pro ukládání logů – Loki. Loki, jak již bylo zmíněno v kapitole 4.1.1.5, byl v době psaní práce v beta verzi. Grafana i Loki jsou vydávány pod licencí Apache 2.0.

Redash je vizualizační nástroj, který vyhledává data v hojně používaných datových úložišť, jako jsou Oracle, InfluxDB, Google BigQuery, MongoDB, Elasticsearch atd (128). Jazyk vyhledávání uzpůsobuje zvolené databázi. Redash je vydáván pod licencí BSD-2, která povoluje komerční využití produktu.

Při výběru konkrétní databáze pro toto řešení musí být zvážen i fakt, že ukládány mají být logy i metriky a řešení by mělo umožnit provedení ad-hoc analýzy, respektive fulltextové vyhledávání. S ohledem na to, že byl vybrán Logstash pro zpracování logů, se přímo nabízí nástroje Elastic Stacku. Kibana i Elasticsearch nejenže vyhovují požadavkům, ale přináší i zaručenou kompatibilitu, protože všechny tři nástroje jsou vydávány stejnou firmou a jsou pro společné využití navržené.

5 Nastavení Elastic Stacku na míru a zpracování logů

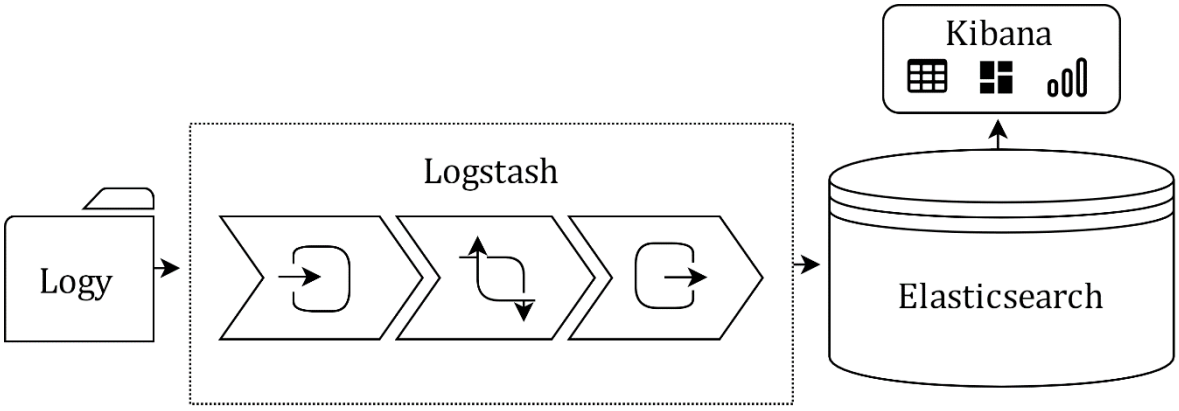
Jak již bylo zdůvodněno v předchozí kapitole, pro zpracování logů z aplikace oneApp, byl vybrán nástroj Logstash. Protože Logstash sám o sobě neumí vizualizovat data pro potřeby analýzy logů, je nezbytné navázat na něj ještě vizualizační nástroj. Pro tento účel byla vybrána Kibana, což je nástroj z rodiny Elastic. Pro ukládání, indexování a rychlé vyhledávání dat Elastic nabízí Elasticsearch, který je důležitým back-endem právě pro Kibanu.

Do virtuálního prostředí autorova počítače bylo naistalováno Ubuntu ve verzi 16.04, jelikož vyšší verze Ubuntu nejsou u většiny nástrojů Elastic Stack prozatím podporovány. V této kapitole budou všechny tři nástroje nastíněny z technického hlediska a následně bude popsána jejich instalace a konfigurace spolu s kroky pro uzpůsobení na míru pipeliny podle logu aplikace oneApp. Hlavními zdroji této kapitoly je převážně oficiální stránka Elasticu (129), stránka Elasticu na GitHubu (130) a kniha The Logstash book (131).

Konfigurace Elastic Stacku by mohla být také provedena přes nástroje pro správu konfigurací, jako je Puppet (132) nebo Chef (133). Díky těmto nástrojům je možné lehce přejít na předchozí verzi konfigurace v případě nehody a umožňují přehlednou správu konfigurace na více strojích. Avšak pro účely této práce není správa konfigurace stěžejní.

5.1 Technický popis Elastic Stacku

Pro plynulejší a jasnější vysvětlení konfigurace a nastavení na míru celého Elastic Stacku je nejdříve věnována pozornost detailnějšímu popisu jeho fungování, potažmo jeho technickému popisu.



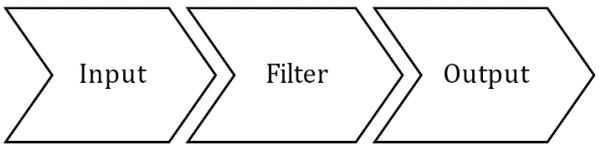
Obrázek 8 Komponenty Elastic Stacku (autor)

Logstash přijímá data z různých zdrojů, například logy v podobě textových souborů uložených ve složce, zpracovává je a posílá je do NoSQL databáze Elasticsearch (Obrázek 8). Data uložená v Elasticsearchi jsou vizualizována v podobě grafů a tabulek pomocí Kibany.

5.1.1 Technický popis Logstashe

Při instalaci přes Debianové a RPM balíčky je vše potřebné k funkci Logstashe nainstalováno do třech adresářů. Domovským adresářem instalace Logstashe se spouštěcími skripty a nainstalovanými pluginy je */usr/share/logstash,* zatímco adresář se všemi konfiguračními soubory je */etc/logstash*. V adresáři */var/logstash* se nachází soubory logů a datové soubory, které Logstash používá. (134)

Hlavní konfigurační soubory jsou *logstash.yml*, *jvm.options* a vlastní konfigurační soubor pro zpracování dat s příponou *conf*. V souboru *logstash.yml* se nastavuje například spouštění Logstashe, nastavení pipeliny (které je rozebráno v kapitole 5.3.1) nebo umístění vlastních konfiguračních souborů (135). Logstash běží v Java Virtual Machine (dále JVM) a parametry souboru *jvm.options* konfigurují JVM pro spuštění Logstashe. Soubor *jvm.options* se upravuje jen výjimečně, a to především v případě, kdy by nestačila velikost paměti (136). Vlastní konfigurační soubor musí být uložený v adresáři, který je definovaný v souboru *logstash.yml* a musí mít příponu *.conf*.



Obrázek 9 Pipelina Logstashe (autor)

Každý vlastní konfigurační soubor (pipelina) má tři základní části (Obrázek 9):

- definice vstupních dat (*input*),
- transformace dat (*filter*)
- a definice výstupních dat (*output*).

Vstupní data jsou definovaná v části *input*. Pokud vstupem mají být soubory, zadává se cesta k těmto souborům. Určuje se jejich typ, což je v podstatě označení dat sloužící pro další práci s nimi. V případě potřeby slučování více řádků do jedné události se také v této části nastaví, podle jakých pravidel mají být data spojena. Jako vstupní plugin je například k dispozici Twitter, Kafka, GitHub, nebo JDBC plugin (137), který umožňuje posílat data do Logstashe z jakékoliv databáze podporující rozhraní JDBC[11], což je například Oracle DB.

Transformace dat se nastavuje v části *filter* daného vlastního konfiguračního souboru. Existuje řada pluginů, které se dají doinstalovat do Logstashe, popřípadě je možné nějaký plugin upravit, nebo si vytvořit svůj vlastní. Strukturovaná data se jednoduše dají zpracovat Logstashem pomocí k tomu určených pluginů. Takový plugin je k dispozici například pro soubory CSV, JSON a XML (138).

Nestrukturovaná data se nejčastěji vyparsují na jednotlivá pole pomocí masek Grok pro snadnější vyhledávání, porovnávání a výpočet ukazatelů, metrik atd. Masky Grok

[11] Java Database Connectivity (JDBC) je API Javy, které určuje, jak přistupovat do databáze (175).

mají syntaxi *%{syntaxe:sémantika}*, kde syntaxe je jméno masky a sémantika je vlastní označení pole. Masky Grok mohou být použité z knihovny Grok (139), kde jsou k dispozici předdefinované masky. Vlastní masky je možné vytvořit pomocí regulárních výrazů, případně lze jako část výrazu využít předdefinované masky Grok. Při vytváření masek Grok je vhodné využít debugovací nástroje, jako je například Grok Debugger v Kibaně nebo Grok Constructor (140), který nabízí i debugování víceřádkového logu.

V pluginu *filter* se dají pole přejmenovávat, mazat nebo přidávat, ať už fixní, nebo vypočítaná na základě jiných údajů. V tomto pluginu je data možné upravovat pomocí podmínek a také přidávat tagy, které slouží k označení nějaké skupině událostí. Neméně používaný je také plugin, který umožňuje definovat typ pole. Ten se používá převážně pro datová pole.

V části *output* se používají pluginy, které určují, kam mají data dále plynout. Jednou z možností je vypsání dat do konzole, nejčastěji se však data odesílají do Elasticsearche. V takovém případě je minimálně uveden port a název indexu. Další možností je například zpracovaná data posílat do jiného Logstashe nebo je vypisovat do souboru, emailu, Google Cloudu, Influxu, databáze MongoDB atd.

Tento celý proces od vstupu po výstup se nazývá pipelina. Pipelin, nebo v podstatě konfiguračních souborů, lze mít několik. Nicméně je vhodné je definovat v dalším konfiguračním souboru *pipelines.yml*. Takový postup je možný v bezplatné verzi až od verze 6.3, a je součástí zmiňovaného X-packu. Jeho výhodou je, že pipeliny běží paralelně a nejsou závislé na chodu ostatních procesů. V případě, že by bylo definováno více pipelin bez použití výše uvedeného konfiguračního souboru, by v případě problému a zaseknutí jedné pipeliny přestaly fungovat všechny ostatní. Logstash by totiž nepřijal další dávku dat na vstupu, dokud by neodeslal na výstup všechny dosavadně zpracovávané dávky (141).

5.1.2 Technický popis Elasticsearche

Elasticsearch indexuje data a umožnuje jejich vyhledávání (54). Vyhledávání v Elasticsearchi funguje v téměř reálném čase (142). Data jsou dostupná přes standardní REST API a JSON (54).

Každá událost je tvořena poli, které dohromady tvoří dokument, a ten je uvnitř daného indexu. Pojmy v Elasticsearch se dají srovnat s pojmy v relační databázi (131):

- *„index je tabulka,*
- *dokument je řádek tabulky*
- *a pole je sloupec tabulky“*.

Index je soubor dokumentů s podobnou charakteristikou (138). Dokument je základní jednotka informace, která může být indexována, musí jí být přiřazen typ a je vyjádřená ve formátu JSON (143). Běžně se dokument po jeho zaindexování dá během jedné

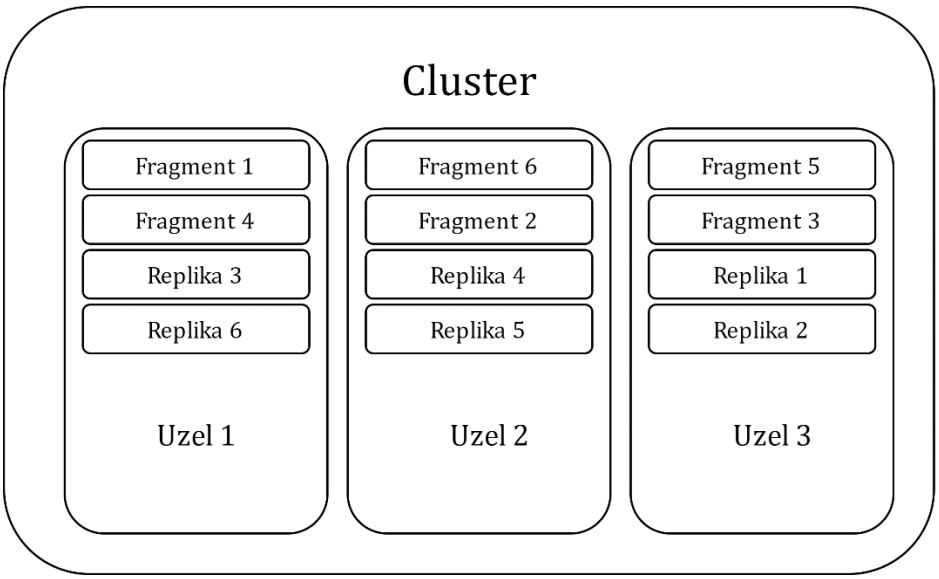
sekundy vyhledat (142). Elasticsearch používá na vytváření invertovaných indexů Apache Lucene[12] (dále Lucene) (131).

V předchozích verzích bylo navíc možné definovat tzv. mapování, které odpovídá schématu v relační databázi, takže bylo možné v jednom indexu mít více typů dokumentů. Od verze 6.0.0. taková možnost není. Důvodem je fakt, že pokud byl název polí stejný v různých mapováních v rámci jednoho indexu, musela tato pole mít stejný typ pro všechna mapování. Navíc ukládání entit, které měly málo společných polí, vedlo k narušení schopnosti Lucene dokumenty efektivně komprimovat. Z těchto důvodů je lepší dokumenty rozdílných typů ukládat do rozdílných indexů. Komprimování dokumentů je v tomto případě efektivnější a pravděpodobně bude přesnější fulltextové vyhledávání, protože dokumenty v jednom indexu představují jednu entitu. (143)

Architektura Elasticsearche je škálovatelná a umožňuje data horizontálně distribuovat a zajistit jejich vysokou dostupnost (144). Pro jednotlivé úrovně hierarchie jsou použity následující pojmy:

- cluster,
- uzel (node),
- fragment (primary shard)
- a replika (replica).

Každý dokument v indexu v Elasticsearchi patří do jednoho primárního fragmentu (144). Ke každému primárnímu fragmentu může být vytvořeno několik kopií tohoto fragmentu, kterým je nazýváno repliky (144). Fragment je instancí invertovaných indexů Lucene, který indexuje a zpracovává dotazy nad podmnožinou dat (145).



Obrázek 10 Architektura Elasticsearche (autor; (146))

[12] Open-source indexovací a vyhledávací software v jazyce Java pod licencí Apache Software Licence, který je podporován neziskovou organizací The Apache Software Foundation (168).

Fragmenty jsou uloženy v uzlech (Obrázek 10), které jsou automaticky součástí clusteru. Uzly jsou jednotlivé servery, které ukládají data a pomáhají clusterům indexovat a vyhledávat data. Cluster je soubor jednoho nebo více propojených uzlů, který umožňuje indexovat a vyhledávat ve všech uzlech. (142)

Fragmenty slouží převážně pro případ, kdy jeden index představuje obrovské množství dat, které převyšuje kapacitu hardwaru serveru (uzlu). Index je tak rozdělený na jednotlivé části, tzv. fragmenty. To umožňuje horizontálně rozdělit obsah a distribuovat operace, čímž se zvýší výkon/propustnost. Pro případ selhání nějakého uzlu se dají fragmenty duplikovat, protože v případě disponování několika uzly v rámci clusteru repliky nikdy nejsou na stejném uzlu jako primární fragmenty. Tím se také zrychlí vyhledávání, protože lze činnost paralelizovat. (142)

Elasticsearch má konfigurační soubor *elasticsearch.yml* pro konfiguraci samotného Elasticsearche, soubor *jvm.options* sloužící pro konfiguraci JVM a soubor *log4j2.properties* pro konfiguraci logování. Při instalaci přes repozitáře jsou tyto soubory v umístění */etc/elasticsearch.* (147)

5.1.3 Technický popis Kibany

Většina nastavení je možné provést přes uživatelské prostředí Kibany. Pro zobrazení dat uložených v Elasticsearchi je nutné vytvořit indexovou masku. Tato definuje, které indexy, respektive data z nich, se zahrnou do vizualizace. Název indexu, který se definuje v pipelině Logstashe je často složený nejen z fixních slov, ale také z dynamických proměnných. Běžně se název indexu udává jako název a datum. Pro definici indexové masky se tak v Kibaně užívá regulárních výrazů.

Tabulky, grafy a ostatní vizualizace jsou vždy vázané k té určité indexové masce. Pro malé množství dat by indexová maska nebyla ani zapotřebí, stačilo by zobrazit všechna data daného indexu. Nicméně data velkých objemů jsou rozdělena na více indexů (tabulek v názvosloví relačních databází).

Dalším krokem je nastavení datového pole, podle kterého budou data filtrována podle času. Zvolit se dá jakékoliv časové pole daného indexu včetně základního pole *@timestamp*, které udává čas, kdy data byla do Elasticsearche uložena. Kibana povoluje datové pole nevyplnit, ale varuje, že data nebudou moci být předfiltrována podle časového období, což by jinak snížilo objem dat pro další filtrování.

Právě nastavení časového období hraje v Kibaně velkou roli. Je možné nastavit fixní časové období, ale i relativní časové období ohraničené z obou stran. Nejvíce používané je nastavení časového období na nedávnou dobu, jako je posledních 15 minut, poslední hodina, dnes, tento týden, tento měsíc nebo poslední rok. Podle tohoto nastavení se v Kibaně zobrazí pouze data, která tomuto časovému rámci odpovídají a přizpůsobí se tomu i časová osa v grafech.

V grafech je data možné zobrazit pouze agregovaně, například jako počet, průměr, percentil, medián, součet atd. Kibana omezuje velikost intervalu pro agregaci podle

nastaveného časového období. Například tomu, aby se data zobrazila po sekundách, odpovídá pětiminutový úsek. Data za minutu se dají agregovaně zobrazit minimálně po 100 milisekundách. Je možné nastavit i automatický interval, který zřejmě pro přehlednost volí spíše vyšší intervaly.

V uživatelském rozhraní Kibany je několik modulů v postranním panelu. Kromě zmiňovaného nastavení (modul *Management*) a vizualizací (modul *Visualize*) je tam k dispozici modul pro základní vyhledávání *Discover*. Vyhledává se pomocí jazyka Kibana Query Language a syntaxe vyhledávání je *název pole: hodnota pole*. Je možné vyhledávat na základě složitějších podmínek s využitím logických výrazů *and* a *or*, podmínky je možné i negovat. V tabulce výsledků lze upravit zobrazované sloupce. Kibana umožňuje vytvoření vlastního dashboardu, kam se mohou vkládat vytvořené tabulky a grafy.

V modulu *Timelion* se vizualizují data časových řad z nezávislých datových zdrojů pomocí jednoduchého výrazového jazyka (148). Data se dají různými způsoby filtrovat, transformovat nebo agregovat (148). V modulu *Canvas* se data dají zobrazit do barevných rámečků s obrázky (149). *Canvas* je vhodný pro jasné a graficky vzhledné zobrazení čísel, především pro marketingové účely (149).

Dalším modulem Kibany je modul *Infrastructure*, ve kterém mohou být zobrazeny metriky z různých nástrojů, jako jsou Apache, Kubernetes, Docker nebo MongoDB. V tomto modulu se Metricbeat postará o celou konfiguraci Elasticsearche i Kibany pro vybraný nástroj z kterého je žádoucí metriky vizualizovat. (150)

Modul *Logs* je obdoba modulu *Infrastructure*, ale pro logy. Tento modul umožňuje poslat logy běžných formátu například z Kubernetes, MySQL nebo Apache pomocí Filebeat do Elasticsearche konfigurací z uživatelského prostředí Kibany (151). Tento modul slouží pro zobrazení logů ze všech systémů najednou v reálném čase, a dokonce přidat běžná pole logů do zobrazení (151). Modul *APM* slouží pro monitoring výkonu softwarových služeb a aplikací v reálném čase (152).

V modulu *Machine learning* je možné data určitého indexu nebo data ze souboru strojově analyzovat (153). Všechny funkce strojového učení jsou dostupné pouze v placené verzi nebo v měsíční trial verzi (61). Ostatní zmíněné modulu jsou součástí bezplatného předplatného Basic (61).

5.2 Instalace a konfigurace Elastic Stacku

Všechny komponenty Elastic Stacku byly naistalovány do virtuálního prostředí s operačním systémem Ubuntu 16.04 LTS. Popsány jsou jen základní kroky instalace a konfigurace, které bylo pro umožnění zpracování a vizualizaci logů aplikace oneApp potřeba splnit.

5.2.1 Instalace a konfigurace Logstashe

Logstash běží v JVM, proto je nutné nejdříve naistalovat Javu. Ve virtuálním prostředí byla nainstalována Java 1.8.0_191. Minimální podporovaná verze Javy je Java 8. Přestože je Logstash psaný v JRuby, není třeba JRuby nebo další závislosti instalovat, protože Elastic požadované závislosti přidává do poskytovaných balíčků a tarballů[13] (131).

Komponenty Elastic Stacku nejsou dostupné v základním repozitáři Ubuntu, proto je nejprve nutné do seznamu zdrojů přidat repozitář Elastic. Stejně jako Java, i komponenty Elastic Stacku se instalují pomocí správce balíčků APT nebo YUM. Tyto komponenty se dají také nainstalovat pomocí Docker kontejnerů. Všechny balíčky od Elastic Stack jsou podepsány klíčem PGP, aby byl systém chráněný před spoofingem[14]. (154)

Do testového prostředí byla naistalována nejnovější dostupná verze v době psaní práce – verze 6.6.2. Byl vytvořen vlastní konfigurační soubor *oneapp_bff.conf* v umístění */etc/logstash*. Vybrané umístění je výchozí umístění vlastních konfiguračních souborů, takže v hlavním konfiguračním souboru *logstash.yml* nebylo potřeba nic upravovat. Nastavení pipeliny v souboru *oneapp_bff.conf* je detailně popsáno v kapitole 5.3.1.

5.2.2 Instalace a konfigurace Elasticsearche

Elasticsearch pro jeho instalaci také vyžaduje minimálně Javu 8, nainstalovaný PGP klíč a Elastic repozitář. V souboru *elasticsearch.yml* byl změněn název clusteru a uzlu na oneApp. V tomto souboru bylo také nastaveno, aby se ke každému indexu vytvořila jedna primární fragmentace a žádná replika. Některé z údajů je možné vyčíst přímo z JSONu, který vrací Elasticsearch na HTTP požadavek na adresu, na které poslouchá, defaultně *localhost:9200*. Ostatní konfigurační soubory změněny nebyly.

Další konfigurace Elasticsearche se dají provést přímo v rozhraní Kibany. Dá se tu pracovat především s indexy. V tabulce správy indexů je zobrazeno, kolik má index primárních fragmentů, replik, dokumentů, a kolik místa zabírá. Je možné také upravit některá nastavení indexu v Kibaně, a to v JSONu. Indexy se dají zavřít a tím znepřístupnit dokumenty daného indexu, anebo je možné indexy vymazat či sloučit.

Ke clusteru Elasticsearche se dá připojit i dvěma dalšími způsoby, a to pomocí REST API a klientských SDK v několika jazycích, jako jsou Java, Python, .NET, SQL atd. (155).

5.2.3 Instalace a konfigurace Kibany

Stejně tak Kibana vyžaduje mít nainstalovanou Javu, PGP klíč i repositář Elasticu před instalací samotné Kibany. Kibana se dá instalovat stejně jako ostatní části Elastic Stacku pomocí tarballů, zip souborů, RPM balíčků atd.

[13] Souborový archiv vytvořený programem tar (171).
[14] *„Činnost v počítačové síti, kdy se útočník vydává za někoho jiného“* (169).

Konfigurační soubor Kibany je *kibana.yml*, ten je defaultně umístěn při instalaci přes debianovský balíček v adresáři */etc/kibana* (156). Ve výchozím nastavení Kibana běží na *localhost:5601*, tento port byl pro účely této práce ponechán, takže nebylo potřeba v konfiguračním souboru provádět žádné změny.

5.3 Zpracování a odeslání logů

Tato kapitola se věnuje nastavení pipeliny pro zpracování logů produkovaných z aplikace oneApp, procesu jejich odeslání a výsledné strukturované podobě těchto logů. Pro účely práce byly poskytnuty firmou NTT anonymizované logy za 4 hodiny od 4.9.2019 22:00 do 5.9.2019 02:00 ve velikosti 1,52 GB.

5.3.1 Nastavení pipeliny Logstashe

Jak již bylo uvedeno v kapitole 5.1.1, vlastní konfigurační soubor Logstashe, tzv. pipelina, kde se definuje zpracování logů od vstupu až po výstup má tři hlavní části: *input*, *filter* a *output*.

V této kapitole je nastavení této pipeliny, která byla uložena pod názvem *oneapp_bff.conf*, podrobně přestaveno.

V části *input* pipeliny Logstashe v pluginu *file* byla definována cesta k souborům s logy (Část kódu 2). Plugin *file* slouží právě pro nastavení vstupních dat do Logstashe ze souboru. Cesta k souboru obsahuje zástupné symboly pro možnost číst více souborů v jedné složce. Do pole *sincedb_path* se nastavuje cesta k souborům, které mají být sledovány a zpracovány po přidání nových dat do těchto souborů. V případě restartu nebo vypnutí Logstashe si program pamatuje, kterým řádkem naposledy skončil a kterým má spuštění začínat. Pokud je však žádoucí přečíst všechny řádky v daném souboru i v případě, že už někdy byly načteny, nastaví se pole *sincedb_path* na hodnotu */dev/null*.

Plugin *file* běží ve dvou módech – *read* nebo *tail*. V režimu *tail* jsou soubory považovány za nekonečný tok obsahu a plugin sleduje nově připojený obsah do souboru. V režimu *read* je každý soubor považován za úplný obsah, kde je konečný počet řádků. Z toho důvodu, že se nepředpokládá, že se soubor bude upravovat, je v tomto módu možné zpracovávat data i z komprimovaných souborů. Pokud Logstash vyhodnotí, že je soubor přečten celý až do konce, je soubor vyloučen z aktivně sledovaných a tím se ušetří paměť a další zdroje.

Logy z aplikace oneApp je požadováno číst v reálném čase, s tím, jak postupně do souborů přibývají, proto je nutné využít mód *tail*. Dále je nastaveno, aby se soubor četl od začátku a aby se uzavřel po hodině, kdy do souboru nepřibydou další data.

```
input
{
    file
    {
        path => "/home/monika/logs/*.log.gz.anonymized"
        sincedb_path => "/dev/null"
        mode  => "tail"
        start_position => "beginning"
        close_older => "1 hour"
        codec => multiline
        {
            patterns_dir => ["/etc/logstash/patterns"]
            pattern =>"(^%{DOTS_SHORT1}(\r)?$|^%{DOTS_SHORT2}(\r)?$|^%{ID}(\r)?$|^
            %{ADDRESS}(\r)?$|^%{ADDRESS2}(\r)?$|^%{RESPONSE_CODE}(\r)?$|^%{PAYLOAD
            1}(\r)?$|^%{PAYLOAD2}(\r)?$|^%{DOTS_LONG}(\r)?$)"
            negate => "false"
            what => "previous"
        }
    }
}
```

Část kódu 2 Input pipeliny (autor)

V pluginu *file* se také dá nastavit oddělovač, který označuje konec řádku. Ve výchozím nastavení je to znak konce řádku. Pokud je událost přes více řádků, dají se řádky spojit přes plugin *codec multiline*. V něm se nastavuje maska, která je zapsaná buď regulárními výrazy, nebo pomocí masek Grok. V případě, že je maska rozsáhlejší, dá se pro přehlednost využít definování vlastních masek v odděleném souboru, ke kterému je v pluginu *multiline* definována cesta.

```
DIRECTION (APP~~~~>BFF|BFF~~~~>HAL|BFF<~~~~HAL|APP<~~~~BFF)
LOG_TYPE (Outbound|Inbound)
BEGGINING %{CISCOTIMESTAMP}%{SPACE}%{NOTSPACE:server}%{SPACE}%{INT}%{SPACE}%{TIMES
TAMP_ISO8601:timestamp}
MAIN %{BEGGINING}%{SPACE}%{NOTSPACE:app_instance}%{SPACE}%{NOTSPACE}%{SPACE}-
%{SPACE}Audit%{SPACE}%{SYSLOG5424SD}%{SPACE}%{DIRECTION:direction}%{SPACE}-
%{SPACE}%{LOG_TYPE:log_type}%{SPACE}Message
DOTS_SHORT1 ----------------------------
DOTS_SHORT2 ----------------------------
DOTS_LONG ------------------------------------
ID ID:%{SPACE}%{INT:ID}
RESPONSE_CODE Response-Code:%{SPACE}%{INT:response_code}
ADDRESS Address:%{SPACE}http://t-app.t-mobile.cz/oneapp-bff/api/%{WORD:module}(//)
?(%{NOTSPACE})?
ADDRESS2 Address:%{SPACE}%{URIPROTO}://(?:%{USER}(?::[^@]*)?@)?(?:((?:%{HOSTNAME:m
odule}|%{IP})(?::%{POSINT:port})?))?(?:%{URIPATHPARAM})?
APP_VERSION %{INT}(.%{INT})?(.%{INT})?
PAYLOAD1 Payload:{"campaignParameters":{"global":{"langCode":"%{WORD:lang_code}",“
density":"%{WORD:density}","os":"%{WORD:os}","appVersion":"%{APP_VERSION:app_versi
on}
PAYLOAD2 Payload:{"global":{"langCode":"%{WORD:lang_code}","density":"%{WORD:densi
ty}","os":"%{WORD:os}","appVersion":"%{APP_VERSION:app_version}
FILENAME_DATE [0-9]{8}
```

Část kódu 3 Nadefinované masky (autor)

Tento přístup byl zvolen a v odděleném souboru byly definovány vlastní masky s využitím masek knihovny Grok (Část kódu 3). V nastavení pluginu *codec multiline* se v parametru *patterns_dir* byla zadána cesta k souboru, kde jsou tyto vlastní masky

uvedeny. V tomto případě plugin funguje tak, že všechny řádky, které odpovídají nějaké z definovaných masek, se připojí k předchozímu řádku. Konkrétně se jedná o řádky, kde jsou pouze spojovníky a řádky, které začínají na slovo *response-code*, *address, payload* nebo *ID*.

Chování, že se tyto vyjmenované řádky připojují k těm, co jim předchází, se docílilo nastavením hodnoty *previous* do parametru *what* a definováním hodnoty *negate* na *fals*e. Díky těmto dvěma parametrům je umožněno nastavit logiku zpracování v několika různých kombinacích.

Nastavení části *filter* v pipelině je nejrozsáhlejší, a proto je jeho kód rozdělen na šest částí a vysvětlen odděleně. Proto první část začíná definováním, že se jedná o část *filter* v pipelině a v šesté části je ukončující závorka části *filter*.

Nejdříve je pomocí pluginu *grok* vyparsován datum z názvu textového souboru obsahující logy (Část kódu 4). Tento krok je nezbytný pro vytvoření dynamického názvu indexu v části *output*.

```
filter
{

    grok {
        patterns_dir => ["/etc/logstash/patterns"]
        match =>  {"path"    =>    "/bff%{FILENAME_DATE:filename_date}[0-9]{2}.[0-
                  9]{1,2}.log.gz.anonymized$"}
    }
```

Část kódu 4 Filter pipeliny – 1.část (autor)

Poté se událost porovná s maskou (Část kódu 5), které odpovídá například první událost (prvních 8 řádků) v ukázce logu (Část kódu 1). Konkrétně se událost musí skládat z řádku, který začíná časovým razítkem a končí informací o směru události (například *APP~~~~>BFF*). Dalším řádkem je několik spojovníků v řadě a za nimi je řádek s identifikátorem, za kterým následuje řádek s adresou, anebo s kódem odezvy, anebo řádek začínajícím slovem *payload*. Posledním řádkem je opět několik spojovníků v řadě.

V případě, že událost odpovídá hlavní masce, přidá se k ní označení *main_grok_ok*, v opačném případě se označí tagem *main_grok_fail*. Ty události, které neodpovídají hlavní masce, se porovnají s další maskou v pluginu *grok*. Této masce by měly odpovídat všechny události, které začínají časovým razítkem. Tento krok je v pipelině proto, aby se do pole *timestamp* vložil čas zapsaný v logu. Toto pole je velmi důležité pro zobrazení dat v Kibaně, ačkoliv je možné si takové pole nevytvářet a všechno seřazování, agregaci i vizualizaci dat provádět podle času, kdy událost byla uložena do Elasticsearch. Vhodnější a přesnější je však mít data zobrazena podle data uvedeného v logu, o to více v případě zpracování historických dat.

Pokud by nějaká událost neodpovídala ani druhé masce, byla by označena tagem *date_grok_fail*. Události s takovým označením by se neměly vyskytovat, pokud je tomu naopak, mělo by se zkontrolovat, zda je vše nastaveno správně. Zkontrolovat lze i počty

událostí s tagem *main_grok_fail* a *date_grok_ok*, které by se měly v případě plné funkčnosti rovnat. Jinými slovy všechny události by měly mít vyparsované pole *timestamp*.

Aby se pole *timestamp* u všech událostí správně zobrazovalo jako datumové pole, je vhodné ho ještě jednou definovat jako datumové pole v daném formátu pomocí pluginu *date*.

```
grok
{
    patterns_dir => ["/etc/logstash/patterns"]
    match =>
    {
        "message" => "^%{MAIN}(\r)?\n((%{DOTS_SHORT1})|(%{DOTS_SHORT2}))(\r)?\
                    n%{ID}(\r)?(\n)?((%{ADDRESS})|(%{ADDRESS2}))?(\r)?(\n)?(%
                    {RESPONSE_CODE})?(\r)?(\n)?((%{PAYLOAD1})|(%{PAYLOAD2}))?
                    (\r)?(\n)?(%{DOTS_LONG})?(\r)?$"
    }
    add_tag => ["main_grok_ok"]
    tag_on_failure => ["main_grok_fail"]
}
    if "main_grok_fail" in [tags]
{
    grok
    {
        patterns_dir => ["/etc/logstash/patterns"]
        match =>
        {
            "message" => "^%{BEGINNING}"
        }
        add_tag => ["date_grok_ok"]
        tag_on_failure => ["date_grok_fail"]
    }
}
date
        {
            match => ["timestamp","ISO8601"]
            target => "timestamp"
        }
```

Část kódu 5 Filter pipeliny – 2. část (autor)

Ty události, které bylo možné rozparsovat pomocí hlavní masky (s označením *main_grok_ok*), se dále zpracovávají (Část kódu 6). Zjednodušeně se události, které prošly hlavní maskou rozdělí na dva typy událostí, události označené tagem

- *completed*
- a *timeouted*,

kde události označené tagem *completed* jsou žádosti a odpovědi na žádost z vrstvy APP do vrstvy BFF a události označené tagem *timeouted* jsou žádost z vrstvy APP do BFF, na které nebylo zodpovězeno do stanoveného času.

Podrobněji, u událostí označenými tagem *main_grok_ok* je definován číselný datový typ poli s kódem odpovědi. Tento krok se v pipelině využije později, nicméně je nutné definovat datový typ v této části, aby se u všech událostí typ pole rovnal.

```
if "main_grok_ok" in [tags]
{
    mutate
        {
            convert =>
            {
                "response_code" => "integer"
            }
        }
    if [direction] == "APP~~~~>BFF"
    {
        aggregate
        {
             task_id => "%{ID}"
             code =>
             "
             map['message_start'] = event.get('message')
             map['module'] = event.get('module')
             map['timestamp'] = event.get('timestamp')
             map['lang_code'] = event.get('lang_code')
             map['density'] = event.get('density')
             map['os'] = event.get('os')
             map['app_version'] = event.get('app_version')
             map['filename_date'] = event.get('filename_date')
             "
             map_action => "create"
             push_map_as_event_on_timeout => true
             timeout_task_id_field => "ID"
             timeout_timestamp_field => "@timestamp"
             timeout => 60
             timeout_tags => ['timeouted']
             timeout_code =>
             "
             event.set('success',0);
             event.set('status','timeouted');
             event.set('direction','APP<~~~~BFF')
             "
        }
    }
```

Část kódu 6 Filter pipeliny – 3. část (autor)

Pro události, které jsou označeny tagem *main_grok_ok* a zároveň žádostmi z vrstvy APP do vrstvy BFF se pomocí pluginu *aggregate* vytvoří mapa s daným identifikátorem události (pole ID) a uloží se do ní pole *message*, *module, lang_code, density, os, app_version, filename_date* a *timestamp*.

V poli *message* je celá zpráva logu a v poli *module* je uložený modul adresy, z které uživatel žádal odpověď. Pole *lang_code* vypovídá o nastaveném jazyku používaného zařízení, *density* o jeho rozlišení displeje a pole *os* o jeho operačním systému. Pole *app_version* obsahuje verzi nainstalované aplikace. Pole *filename_date* obsahuje vyparsovaný datum z názvu souboru, z kterého jsou logy čteny.

V případě, že by odpověď nepřišla do minuty, vytvoří se událost s identifikátorem žádosti, se všemi poli z map a dalšími třemi definovanými vlastními poli, a označí se tagem *timeouted* (Část kódu 6). Do události se uloží všechna pole z mapy v případě vypršelého požadavku díky nastavení *push_map_as_event_on_timeout* na *true*. Všechny ostatní parametry týkající se nastavení vypršelých odpovědí začínají slovem *timeout*.

Odpovědím, které jsou označeny tagem *main_grok_ok* a zároveň přišly do vrstvy APP z vrstvy BFF, jsou pluginem *aggregate* přidána pole z mapy uložené ze žádosti a dále jsou označené tagem *completed* (Část kódu 7). Název pole se může lišit od názvu v mapě, takže hodnota z mapy *timestamp* může být vložena do pole *timestamp_start*, aby bylo odlišeno, že se jedná o časové razítko žádosti. Ostatní názvy polí zůstávají totožné s názvy map.

Důležité je nastavení parametru *map_action* na *update*, protože jinak by Logstash měl tendenci vytvořit mapu novou a pokud by narazil na stejné ID, vyhodil by chybu, protože každá mapa musí vždy jedinečné ID. Mapa se vymaže, jakmile Logstash zpracuje událost, která má nastavený parametr *end_of_task* na *true,* nebo vyprší časový limit, který je nastavený v tomto případě na minutu.

```
if [direction] == "APP<~~~~BFF"
{
    aggregate
    {
         task_id => "%{ID}"
         code => "
         event.set('message_start', map['message_start'])
         event.set('timestamp_start', map['timestamp'])
         event.set('module', map['module'])
         event.set('lang_code', map['lang_code'])
         event.set('density', map['density'])
         event.set('os', map['os'])
         event.set('app_version', map['app_version'])
         "
         map_action => "update"
         add_tag => "completed"
         end_of_task => true
    }
}
```

Část kódu 7 Filter Pipeliny – 4. část (autor)

V páté části *filter* pipeliny je vytvořeno nové pole *success*, které vypovídá o tom, zda odpověď na žádost z vrstvy APP do vrstvy BFF, která byla označena tagem *completed*, byla úspěšná (Část kódu 8). Pole *success* se vyplňuje na základě pole *response_code*. Pokud je kód odpovědi menší než 300, žádost byla úspěšně zodpovězena.

```
if "completed" in [tags]
{
    ruby
     {
        code => "event.get('response_code') < 300 ? event.set('success',1)
        : event.set('success',0)“
     }
```

Část kódu 8 Filter pipeliny – 5. část (autor)

Pokud odpověď úspěšná byla, jsou vytvořeny další dvě pole (Část kódu 9):

- *duration*
- a *prompt*.

V pluginu ruby je vypočítána doba mezi žádostí a odpovědí z/do vrstvy *APP*. V podstatě se jedná o časový úsek, za který server uživateli odpoví. V pipelině je tato doba ukládaná do pole *duration* a následně je pluginem *mutate* tomuto poli přiřazený číselný datový typ. V případě, že doba, za kterou dostal zákazník odpověď byla do tří sekund, dá se žádost považovat za rychle zodpovězenou. Pak je pole *prompt* nastaveno na hodnotu 1, jinak na hodnotu 0.

```
if [success] == 1
{
        ruby
        {
            code => "event.set('duration',  event.get('timestamp')
            - event.get('timestamp_start'))"
        }
        mutate
        {
            convert =>
            {
                "duration" => "float"
            }
        }
        ruby
        {
             code => "(event.get('duration') < 3)  ?
             event.set('prompt',1) : event.set('prompt',0)"
        }
}
```

Část kódu 9 Filter pipeliny – 6. část (autor)

Nakonec se ještě vyplní pole *status*, jehož hodnota záleží právě na tom, jestli odpověď byla úspěšná a rychlá (Část kódu 10). Pole *status* může nabývat následujících hodnot:

- *OK*,
- *slow*,
- *ERR*,
- a *timeouted.*

Odpověď je označena statusem *OK*, pokud je úspěšně a zároveň rychle odeslána. Statusem *slow* jsou označeny odpovědí úspěšné, ale pomalé. Neúspěšným odpovědím je do pole status přiřazena hodnota *ERR*. Odpovědi, které nebyly odeslány, tzn. události s označením *timeouted,* mají v poli status také hodnotu *timeouted*. Nastavení pro takové události je v třetí části *filter* pipeliny (Část kódu 6).

Pole status bylo přidáno kvůli nemožnosti nastavení podmínky na základě hodnot vícero polí u některých vizualizačních prvků v Kibaně, a také kvůli větší přehlednosti při vizualizaci.

```
            ruby
            {
                    code => "((event.get('success') == 1) and (event.get('prompt')
                    == 1)) ? event.set('status','OK') : (((event.get('success')
                    == 1) and (event.get('prompt') == 0))
                    ? event.set('status','slow') : (event.get('success') == 0
                    ? event.set('status','ERR') : event.set('status','non')))"
            }
        }
    }
}
```

Část kódu 10 Filter pipeliny – 7. část (autor)

V poslední části *output* pipeliny se definuje, kam se mají zpracovaná data odesílat (Část kódu 11). Pomocí pluginu *elasticsearch* se data posílají do Elasticsearche poslouchajícím na dané adrese a portu, kde jsou uložena pod daným indexem. Data jsou odesílána na *localhost* na port 9200 do indexu nazvaného *bff- a* vyparsovanou částí názvu souboru, která odpovídá datu událostí.

```
output
{
    elasticsearch
    {
        hosts => ["localhost:9200"]
        index => "bff-%{filename_date}"
    }
}
```

Část kódu 11 Output pipeliny (autor)

Indexy běžně obsahují datum v názvu, aby se daly lépe mazat. V uživatelském rozhraní Kibany je možné v nastavení Elasticsearche nastavit životní cyklus indexů, na jehož základě se indexy mažou automaticky. Do názvu indexu je možné datum vkládat také automaticky, a to např. pomocí masky *%{+dd.MM.YYYY }*. Nicméně tímto řešením je možné datum vložit pouze z pole *@timestamp*.

5.3.2 Odeslání logů a jejich ukládání ve strukturovaném formátu

Po instalaci, konfiguraci a customizaci Logstashe, Elasticsearche i Kibany je konečně možné celý Elastic Stack uvést do provozu. Logstash lze zapnout s danou konfigurací příkazem

```
$ sudo /usr/share/logstash/bin/logstash -f /etc/logstash/oneapp_bff.conf.
```

Logstash začne logy číst, zpracovávat a odesílat, a to až do té doby, než se příkaz neukončí, nebo více než hodinu nepřibyde další události v lozích v definované složce.

Pro další zpracování jsou nejdůležitější odpovědi z vrstvy BFF do vrstvy APP, kde jsou kromě vyparsovaných informací z odpovědi a informací přenesených ze žádosti také přidané informace o tom, za jak dlouho byla odpověď obdržena, jestli byla úspěšná nebo rychlá a podle těchto parametrů je také nastaveno pole status (Část kódu 12). Hodnoty polí *message* a *message_start*, které obsahují celou zprávu z odpovědi, respektive ze žádosti, byla zkrácena pro úsporu místa v práci.

```
{
"prompt" => 1,
"status" => "OK",
"tags" => [[0] "multiline",[1] "main_grok_ok",[2] "completed"],
"log_type" => "Outbound",
"direction" => "APP<~~~~BFF",
"response_code" => 200,
"duration" => 0.003,
"@timestamp" => 2019-10-14T02:37:08.666Z,
"path" => "/home/monika/logs/bff2019090501.11.log.gz.anonymized ",
"filename_date": "20190905",
"timestamp_start" => 2019-09-04T23:59:59.611Z,
"message" => "Sep  5 01:59:59 hkvnode366.cz …
"app_instance" => "bff-server-78fcdcffc9-nnbr6",
"success" => 1,
"timestamp" => 2019-09-04T23:59:59.614Z,
"ID" => "19081675",
"app_version": "15",
"os": "ios",
"density": "2X",
"lang_code": "cs",
"module" => "banners",
"message_start" => "Sep  5 01:59:59 hkvnode366.cz …
}
```

Část kódu 12 Vyparsovaná odpověď z vrstvy BFF (autor)

Událost vypovídající o neobdržené odpovědi je také velmi podstatná pro analýzu logů. Tato událost má na sobě přenesené informace ze žádosti a nová pole informují o neúspěšné vypršelé odpovědi (Část kódu 13). I v této ukázce je hodnota pole *message_start* zkrácena.

```
{
"success" => 0,
"tags" => [[0] "timeouted"],
"ID" => "18451157",
"filename_date": "20190905",
"message_start" => "Sep  5 01:59:59 …
"os": null,
"density": null,
"app_version": null,
"module" => "dashboard ",
"lang_code": null,
"timestamp" => 019-09-04T23:59:59.653Z,
"@timestamp" => 2019-10-14T02:38:23.188Z,
"direction" => "APP<~~~~BFF",
"status" => "timeouted"
}
```

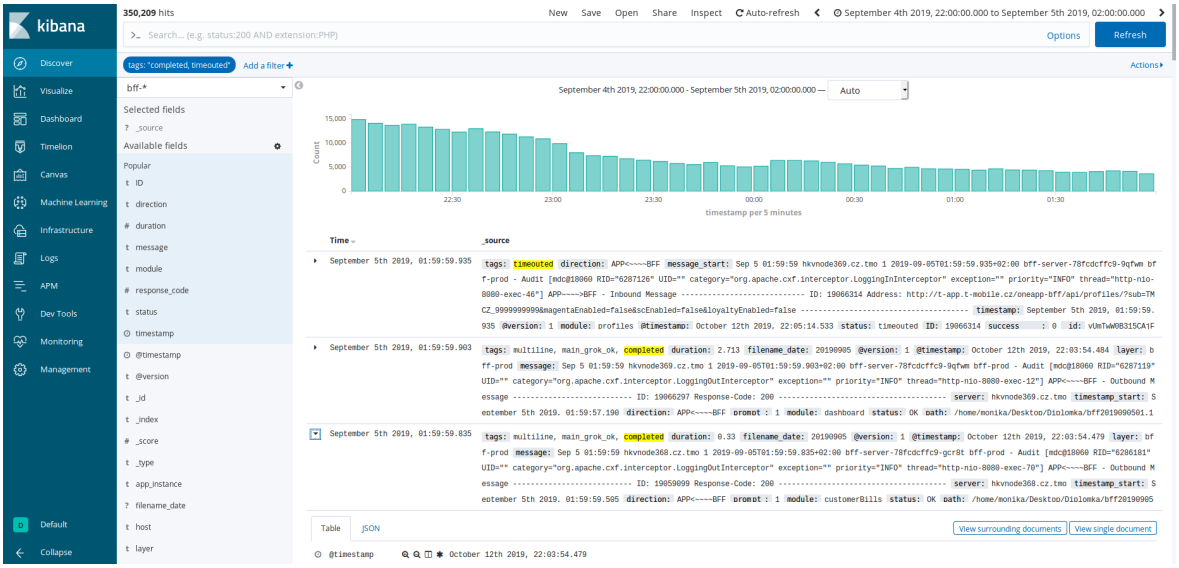
Část kódu 13 Vytvořená událost vypovídající o neobdržené odpovědi (autor)

Události, které nejsou označeny tagem *completed* ani *timeouted* nebudou brány v následující vizualizaci v potaz. Nicméně v Elasticsearchi uložené jsou, aby bylo v případě potřeby možné dohledat detailnější informace.

5.4 Vizualizace událostí logů

V uživatelském prostředí Kibany byla vytvořena indexová maska *bff-** s datový polem *timestamp*. Této masce odpovídají všechny dokumenty (události), jejichž pole *timestamp* obsahuje datum a hodnota v poli index začíná na *bff-*. Časový údaj v poli *timestamp* je převzatý přímo z logu a může se lišit od pole *@timestamp*, které odpovídá okamžiku uložení dokumentu do Elasticsearche.

Vizualizace logů aplikace oneApp je zaměřena jak na analýzu logů v podobě podnikových metrik, tak na analýzu podrobnějších informací z logů a ad-hoc dotazů. Časové rozmezí v Kibaně je nastaveno na 4.9.2019 22:00 – 5.9.2019 02:00, tedy na úsek, pro který jsou logy k dispozici.

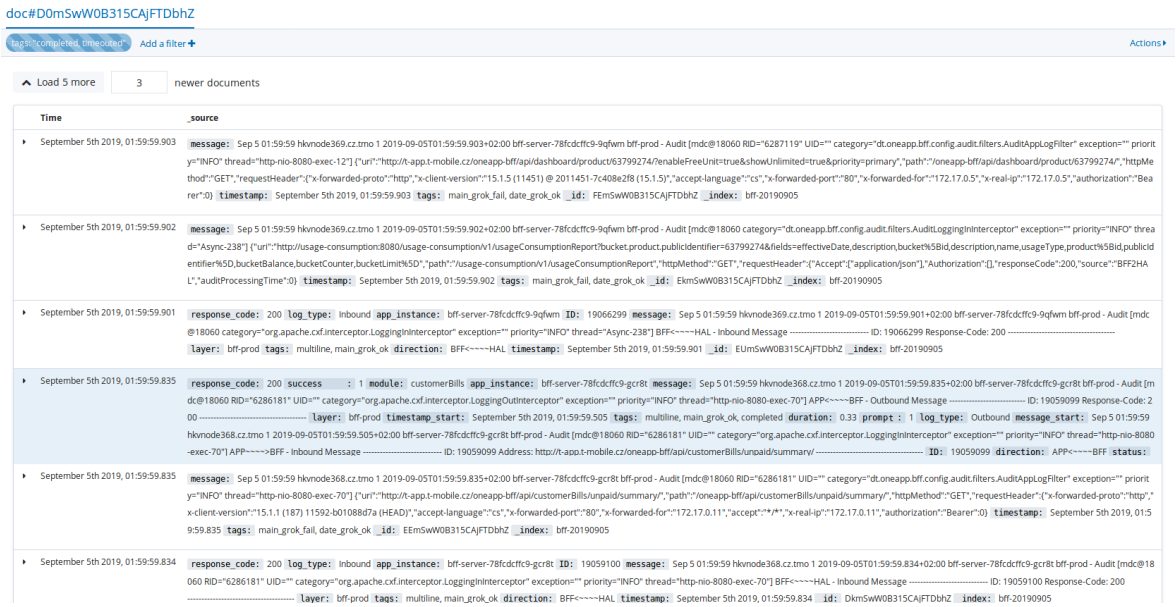


Obrázek 11 Základní zobrazení logů (autor)

V základním výchozím zobrazení jsou logy zobrazeny v tabulce se všemi disponujícími poli (Obrázek 11). Pokud by některá pole měla pro úsporu místa být skryta, je toho možné docílit v nastavení masky indexu. Je možné zobrazit jen vybrané pole, které se uspořádají do jednotlivých sloupců. V tomto zobrazení byly vyfiltrovány události označené tagem *completed* nebo *timeouted*. Celkově se jedná o 350 209 dokumentů (událostí). Pouze tyto události budou sloužit pro veškerou další vizualizaci informací.

Ostatní události nejsou pro vizualizaci v Kibaně tolik klíčové, ovšem jsou k dispozici především pro znalost návaznosti událostí. V Kibaně je k tomuto účelu možné si ke konkrétní jedné události jedním kliknutím zobrazit libovolný počet předchozích a následujících událostí (Obrázek 12). Tato funkce se hodí převážně pro debugování chyb.

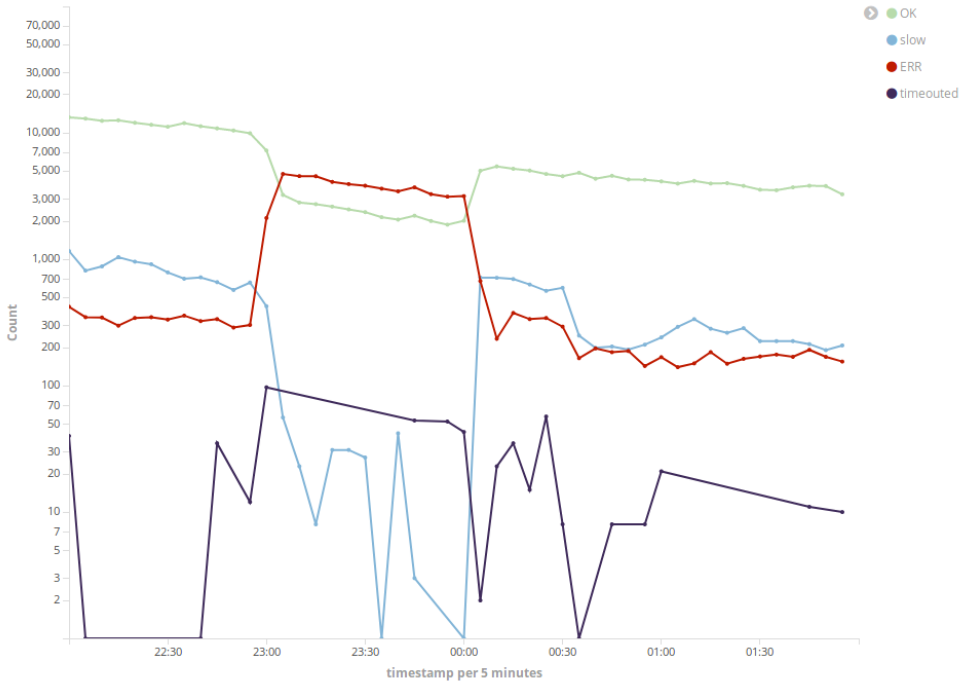
V zobrazení obklopujících událostí zvoleného dokumentu je možné přidat či smazat filtr dat. Vzhled zobrazení obklopujících událostí se dědí ze zvoleného zobrazení, v kterém bylo kliknuto na volbu zobrazit obklopující události. Tudíž pokud původní zobrazení vizualizovalo celé dokumenty, jsou zobrazeny celé dokumenty, pokud byly vybrány jen některé pole, jsou i v tomto zobrazení pouze tyto pole.



Obrázek 12 Zobrazení obklopující událostí (autor)

5.4.1 Vizualizace podnikových metrik

Události BFF vrstvy aplikace oneApp byly pomocí Logstashe rozděleny na čtyři skupiny: úspěšné a rychlé (*OK*), úspěšné a pomalé (*slow*), neúspěšné a obdržené (*ERR*), neúspěšné a neobdržené (*timeouted*), jak bylo podrobněji vysvětleno v kapitole 5.2.1. Pomocí spojnicového grafu s logaritmickou škálou na ose y jsou zobrazeny počty událostí v čase (Obrázek 13). Je zřejmé, že dne 4.9.2019 zhruba hodinu od jedenácti hodin večerních měla aplikace oneApp výpadek.



Obrázek 13 Počet událostí dle statusu (autor)

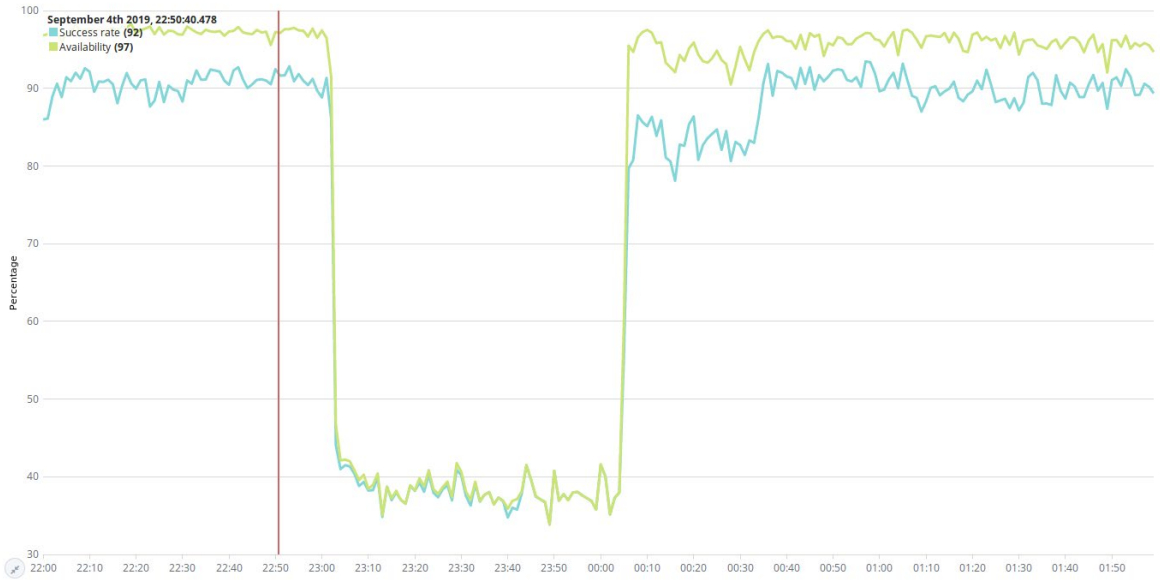
Interval agregace je automatický a pro tento časový rozsah byl Kibanou zvolen na pět minut. Je možné nastavit i fixní interval, ale je nutné počítat s tím, že pokud by byl interval pro daný časový rozsah moc nízký (například v rámci sekund), Kibana ho automaticky zvýší. Vytváření grafů v modulu Kibany *Visualize* je velmi jednoduché – vše je rychlé a intuitivní – avšak má svoje omezení.

Naopak u vytváření grafů časových řad – modul *Timelion* – je možné kombinovat různé zdroje a filtry a provádět nad nimi matematické operace. Navíc je možné si zvolit jakoukoliv vlastní barvu, názvy os a popisky. Výhodou také je, že při posouvání myší se zobrazuje vertikála s hodnotami všech proměnných v čase. Největším mínusem této komponenty je, že prozatím neumožňuje nastavení škály osy y.

Tento graf (Obrázek 14) vytvořený komponentou *Timelion* (Část kódu 14) zobrazuje poměr úspěšně dokončených událostí ku všem událostem (*availability*) a poměr úspěšně rychle dokončených události ku všem událostem (*success rate*). Tento graf zobrazil výpadek ještě zřetelněji.

```
.es(index='bff-*', q='status:OK').divide(.es(index='bff-*', q='tags:completed
and tags: timeouted')).multiply(100).label('Success rate').color(#83d6d9),
.es(index='bff-*', q='status:OK and status:slow').divide(.es(index='bff-*',
q='tags:completed and tags:
timeouted')).multiply(100).label('Availability').color(#cce476).yaxis(label="Pe
rcentage")
```

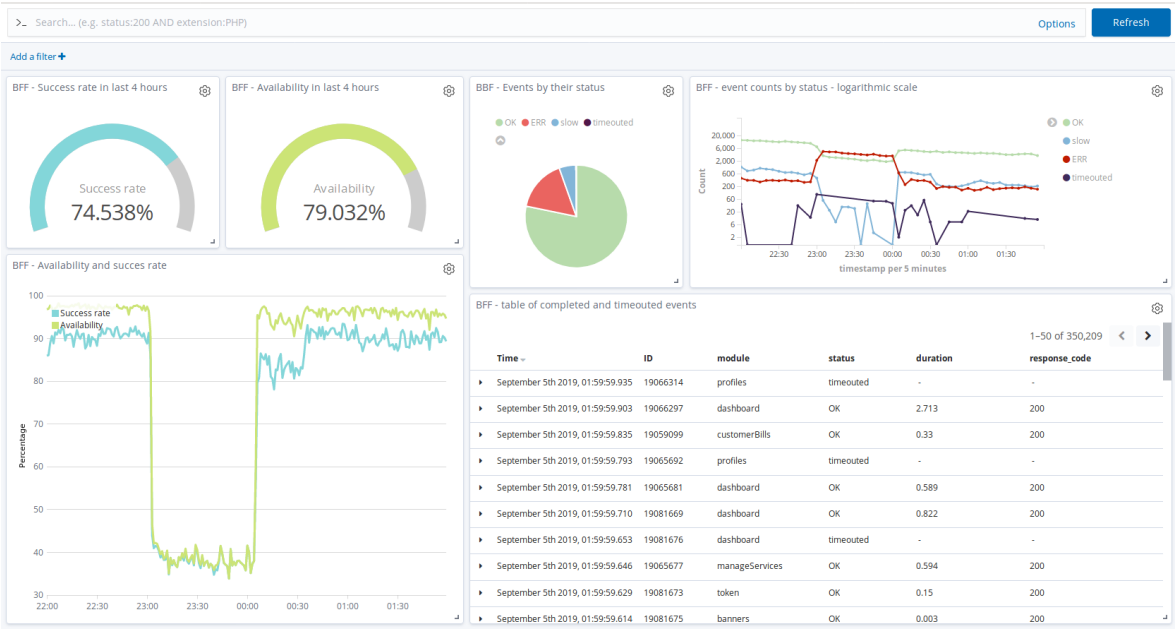
Část kódu 14 Definování grafu časové řady v Timelion (autor)



Obrázek 14 Dostupnost služby v čase (autor)

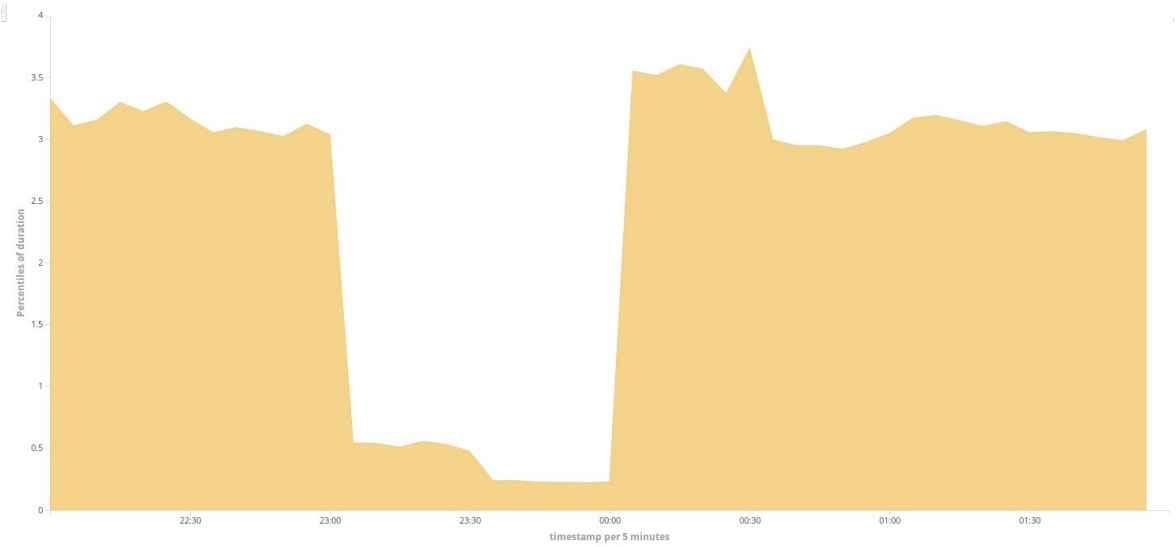
V dashboardu (Obrázek 15) kromě již představených grafů, přibyly ještě ukazatelé úspěšnosti (*success rate*) a dostupnosti (*availability*) za poslední čtyři hodiny. Tyto komponenty byly vytvořeny pomocí komponenty *Visual Builder*. Stejně jako *Timelion* je *Visual Builder* robustním a flexibilním nástrojem, ale jeho nevýhodou je, že nerespektuje globální nastavení časového rozsahu. Proto je nutné časový rozsah

komponentám fixně zadat. Na dashboardu je dále koláčový graf rozdělený podle statusu událostí. Při najetí myší na výseč koláčového grafu se zobrazí počet událostí a procentuální zastoupení statusu. Pro detailnější pohled na logy je také připnutá tabulka s nadefinovanými sloupci.



Obrázek 15 Dashboard podnikových metrik (autor)

Celý dashboard automaticky reflektuje nastavení filtru. Filtrovat lze klikem na části jednotlivých komponent, např. na body v liniovém grafu, na výseč koláčového grafu apod. Tabulka pak zobrazí detailnější informace o vyfiltrovaném jevu. Například pokud bychom klikli na zelenou část koláčového grafu, zobrazily by se pouze úspěšné a rychlé události, a to jak v grafech, tak v tabulce. Dashboardy jsou tak nejenom výborným přehledem informací, ale i nástrojem pro rychlé filtrování dat.

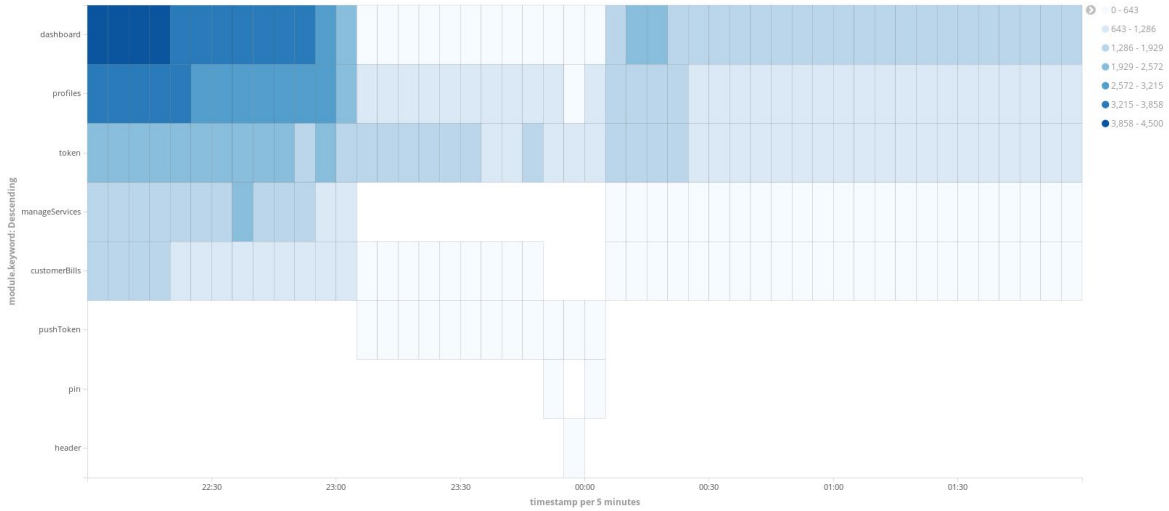


Obrázek 16 95 percentil doby trvání úspěšných odpovědí (autor)

Poslední důležitou podnikovou metrikou sledovanou v dosavadním řešení, která nebyla představena v navrhovaném řešení, je devadesátý pátý percentil doby trvání úspěšné odpovědi. Přesně tento ukazatel vykresluje plošný graf (Obrázek 16). V době výpadku je doba trvání úspěšných odpovědí výrazně nižší. Je tomu tak možná z toho důvodu, že služby, které byly stále v provozu, měly kratší dobu odezvy než ty, které nebyly.

5.4.2 Vizualizace detailnějších informací z logů

Kromě základních grafů Kibana nabízí například teplotní mapu, shrnující tabulky nebo vytvoření ovladačů k ještě rychlejšímu filtrování. Z teplotní mapy je výpadek také zřejmý (Obrázek 17). Jak je patrné, v době výpadku od jedenácté večerní do půlnoci, kdy běžně používané moduly (adresy) nebyly dostupné, se objevují dotazy na jinak opomenuté moduly.



Obrázek 17 Počet dotazovaných modulů v čase (autor)

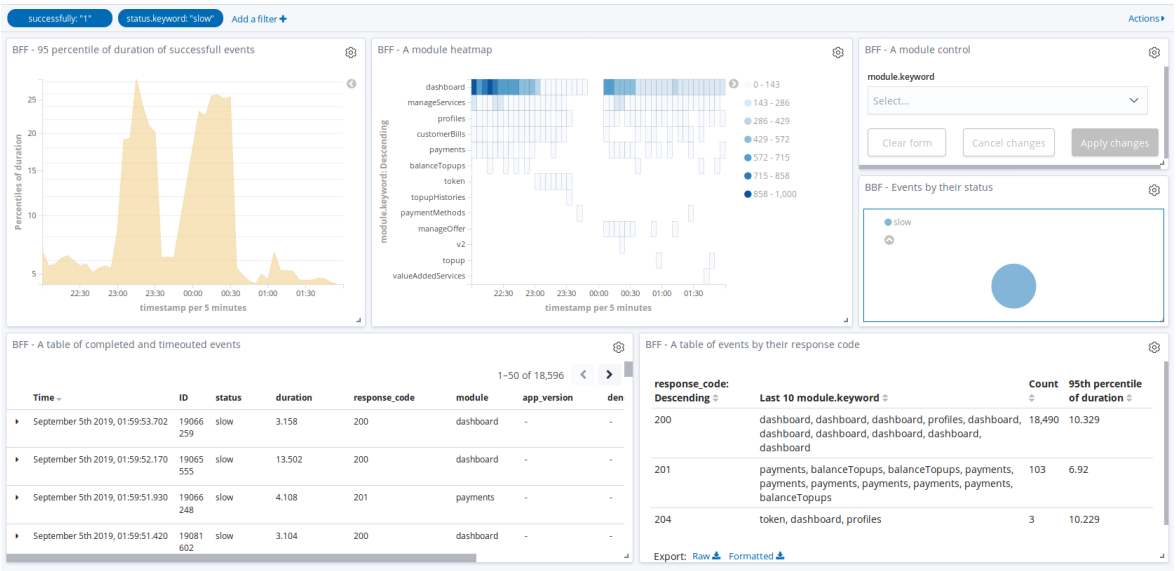
Tabulka četností kódů odpovědí udává také posledních deset modulů, které se k tomuto kódu vážou a pro úspěšné odpovědi je zobrazen i devadesátý pátý percentil doby trvání odpovědi (Obrázek 18). Je evidentní, že doba trvání odezvy událostí s kódem odpovědi 204 je velmi nízká.

response_code: Descending	Last 10 module.keyword	Count	95th percentile of duration
200	dashboard, customerBills, dashboard, dashboard, manageServices, token, banners, profiles, token, dashboard	290,252	3.134
500	manageServices, profiles, manageServices, manageServices, manageServices, manageServices, manageServices, manageServices, manageServices, manageServices	46,513	
401	dashboard, dashboard, dashboard, dashboard, dashboard, dashboard, dashboard, dashboard, dashboard, dashboard	9,790	
204	pushToken, pushToken, pushToken, pushToken, pushToken, pushToken, pushToken, pushToken, pushToken, pushToken	2,189	0.007
422	pin, manageServices, header, manageServices, manageServices, login, balanceTopups, v2, servicePin, pin	526	

Obrázek 18 Události dle kódu odpovědi (autor)

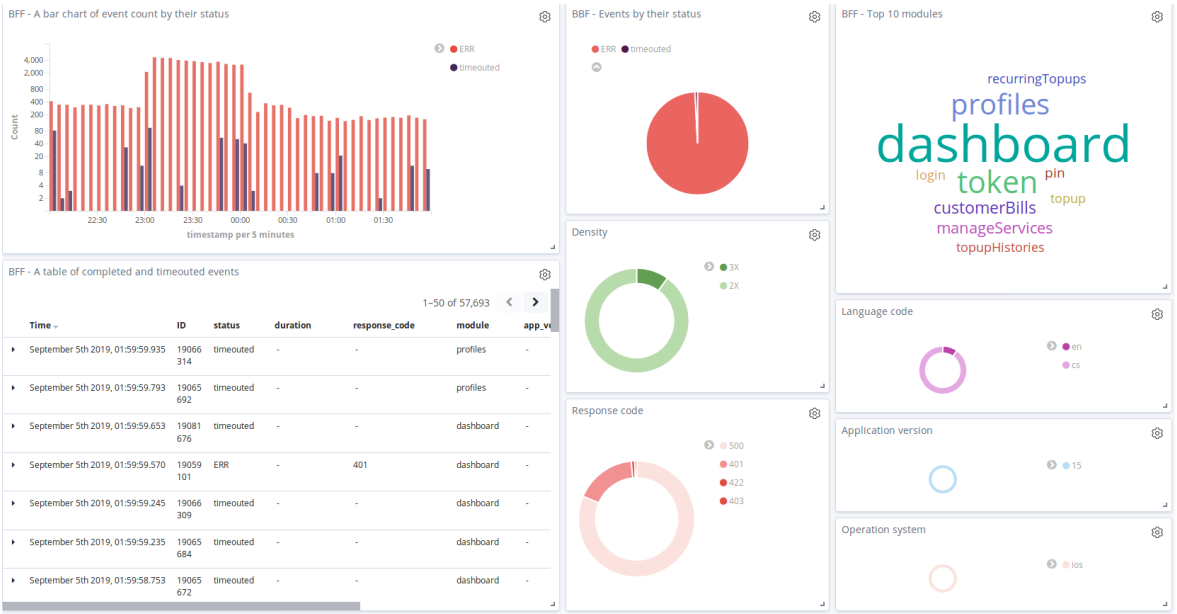
Při vytváření jednotlivých komponent (grafů/tabulek) je vhodné aplikovat pouze základní filtr, aby dané komponenty byly schopny zobrazit data dle nastavení globálního filtru pro celý dashboard. Dashboard (Obrázek 19) byl vytvořen primárně pouze pro úspěšné odpovědi, ale tento filtr je možné odstranit a zvolené komponenty

aplikovat na data jiná. Dashboard obsahuje již představený graf zobrazující devadesátý pátý percentil doby trvání v čase a teplotní mapu modulů v čase. Součásti dashboardu je i již známá tabulka dle kódů odpovědí, základní tabulka logů a koláčový graf dle statusu událostí pro usnadnění filtrování. Novinkou je komponenta, která také usnadňuje filtrování pomocí rozbalení možných hodnot daného pole. V tomto případě je možné jednoduše filtrovat podle jednotlivých modulů bez znalosti syntaxe vyhledávání Elasticsearche.



Obrázek 19 Dashboard úspěšných odpovědí (autor)

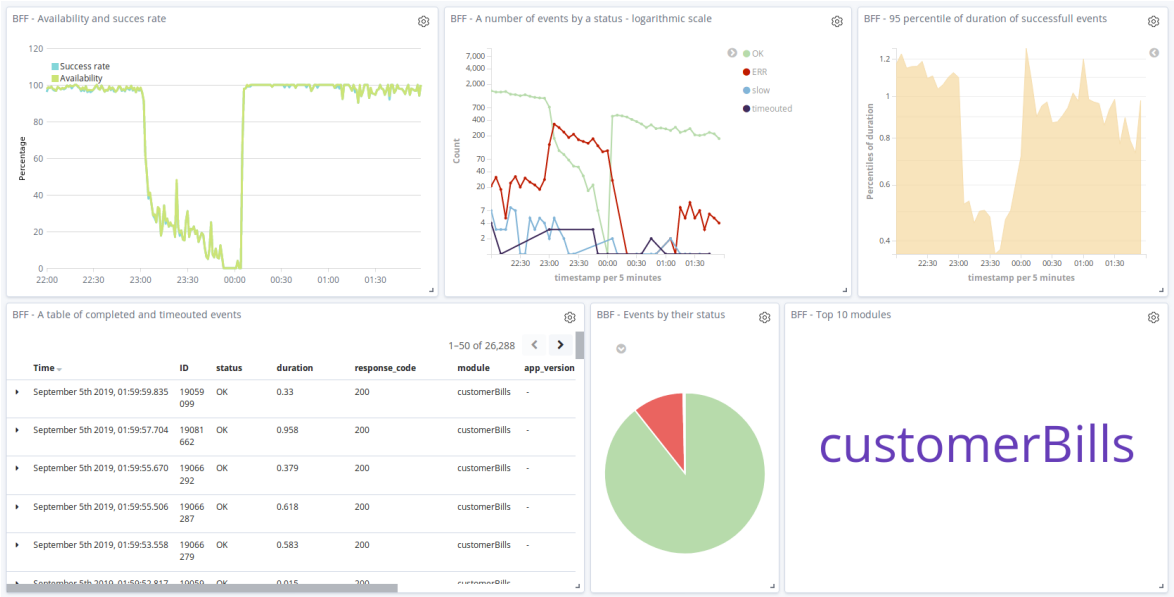
Tento dashboard byl vyfiltrován pouze na úspěšné a pomalé odpovědi. Výpadek je evidentní i na těchto datech. Doba trvání odpovědi je výrazně vyšší než v ostatních časech a počet dotazů na moduly je naopak výrazně nižší.



Obrázek 20 Dashboard neúspěšných odpovědí (autor)

Další dashboard (Obrázek 20) se zaměřuje na neúspěšné odpovědi. Součástí je graf zobrazující počet událostí v čase, který je graficky rozlišen podle jejich statusu. Také je připojen tag cloud nejčastěji nefunkčních modulů. Koláčové grafy mají sloužit pro snadnější filtrování, konkrétně podle statusu události, kódu odpovědi, verze aplikace, jazyka, operačního systému a rozlišení obrazovky zařízení. Samotné základní zobrazení logů v tabulce je také součástí dashboardu.

Poslední dashboard (Obrázek 21) sestavený z již představených komponent je vyfiltrovaný pouze podle modulu, na který bylo dotazováno – konkrétně na dotazy na vyúčtování. Je zřetelné, že výpadek takové dotazy vážně postihl. Dokonce pár minut před půlnocí nebyl žádný dotaz proveden úspěšně. Doba odpovědi úspěšných dotazů je velmi krátká – v řádu jedné sekundy.



Obrázek 21 Dashboard výpisů účtování (autor)

Všechny komponenty dashboardů lze maximalizovat na celou obrazovku a také zobrazit a exportovat samotná data skrývající se pod vizualizací. Data lze filtrovat ze všech různých pohledů. Jednak pomocí předpřipravených komponent, tak pomocí filtrovacího nástroje Kibany, která vyhledává hodnoty konkrétních (vyparsovaných) polí a také pomocí fulltextového vyhledávání. Neméně důležitý je časový filtr. Samozřejmě vždy záleží na tom, jakou otázku by měla data zodpovědět, a podle toho filtry nastavit.

5.5 Možné rozšíření navrženého řešení

K Elastic Stacku existuje nespočet open-source doplňků tvořených komunitou, které funkcionality Elastic Stacku dále rozšiřují (157). Například Datasweet Formula od Datasweet dovoluje flexibilnější nastavení vizualizace v základním modulu *Vizualize* v Kibaně (158). V uživatelském prostředí Kibany se po instalaci tohoto doplňku zobrazí při vytváření grafů pole *Formula input*, do kterého dají zapsat běžné statistické operace a i podmínky (158).

Vizualizace logů je užitečná především pro manažerské účely. Naopak výstrahy na nežádoucí události nebo sled událostí jsou nezbytné pro rychlý technický zásah v případě nějakého problému nebo výpadku. Nástrojů, které v Elastic Stacku lze využít pro zasílání notifikací, je více.

Modul dostupný přímo v Elastic Stacku s uživatelským rozhraním v Kibaně umožňuje nastavení výstrah na události různých povah a je určen pouze pro placené verze (viz kapitola 4.1.1.3). Může notifikovat například o přihlášení uživatelů z neznámých lokalit, o zobrazení citlivých informací, nebo o výrazném snížení počtu dokumentů v Elasticsearchi (159).

Pro neplacené verze je nutné využít alternativní řešení, které ovšem nebude znát informace ohledně zabezpečení, protože modul zabezpečení je v Elastic Stacku také dostupný pouze v placených verzích. Nicméně i tak lze dostávat výstrahy o informacích získaných z logů.

Jednou variantou je plugin *email*, který se nastavuje v pipelině Logstashe v části *output*. Díky tomuto pluginu je možné odesílat informace na základě nastavených pravidel v tomto konfiguračním souboru s využitím předpřipravené šablony na zadaný email. (160) Flexibilní možností, jak odesílat výstrahy přímo z Logstashe, je výstupní plugin podporující systémové příkazy (161).

Další bezplatnou možností, jak notifikovat, je využití rozhraní ElasticAlert (162)od Yelpu spolu s pluginem ElasticAlert Kibana (163) od BitSensoru. ElasticAlert je vydáván pod Apache licencí 2.0 (164) a plugin ElasticAlert Kibana (163) je vydáván pod modifikovanou licencí BSD[15]. Obě licence jsou velmi volné a dovolují komerční využití.

Po instalaci těchto doplňků je možné přímo v Kibaně nastavit pravidla, kdy a kam se budou výstrahy odesílat. Pravidla se vztahují například k frekvenci nebo tempu růstu/poklesu událostí, hlídají práh určité hodnoty pole nebo kontrolují, zda hodnota daného pole se rovná těm definovaným v blacklistu nebo whitelistu. Případně je možné si vytvořit vlastní pravidlo v jazyku Python. Výstrahy je možné odesílat například na email, Slack, GoogleChat, Telegram, JIRI nebo pomocí systémových příkazů. (164) Nativní podpora pro odesílání výstrah přes SNPM protokol neexistuje, ale je možné skrze systémový příkaz volat vlastní skripty pro zasílání SNMP trapů.

Další důležitou součástí monitoringu logů jsou reporty. Generování reportů ad-hoc, či plánovaně v nějaký čas a den nebo na základě pravidel, přináší cenný náhled na funkčnost softwaru. Kibana umožnuje všechny zmíněné způsoby tvorby a zasílání reportů, a to buď ve formátu PDF nebo CSV (165). Monitoring v Kibaně je ovšem dostupný pouze pro placené verze (61).

Bezplatnou alternativou podobné funkcionality jako monitoring v Kibaně přináší modul Sentinl od Sirensolutions vydávaný pod licencí Apache 2.0. Reporty v PDF nebo

[15] Modifikovaná BSD je originální BSD bez reklamní doložky, která je kompatibilní s GNU GPL licencí (170).

PNG je možné zasílat jako přílohu v emailu v určitém čase nebo na základě pravidel. Tento modul navíc umožňuje zasílání výstrah. Veškerá konfigurace probíhá v uživatelském prostředí Kibany v samostatném modulu nazvaném Sentinl. (166)

Poslední podstatnou částí monitoringu logů je analýza agregovaných historických metrik za dlouhé období, například v řádů měsíců a roků. Detailní informace z logů jsou nezbytné především pro hledání příčiny problémů a chyb, kdežto vizualizace informací z logů za dlouhé období slouží spíše pro přehled dostupnosti a funkčnosti softwaru obecně v čase. Často z důvodu nepotřebnosti detailních informací z historických logů, ještě častěji z důvodu úspory místa, jsou logy po nějakém čase zahozeny. Jediná data, která jsou ponechána, jsou agregované metriky. Metriky jakožto jednotlivá čísla za delší období zabírají minimálně místa.

Navíc i v případě, že by historická data nebyla zahazována, bylo by velmi výkonově náročně zobrazit agregované metriky za dlouhý časový interval, protože Elaticsearch by byl nucen vyhledávat možná až v petabajtech uložených dokumentů. Rychlost vyhledávání a vizualizace by záležela na rychlosti jednotlivých uzlů Elasticsearche a také na množství replik každého primárního fragmentu. Čím více replik by bylo k dispozici, tím rychleji by Elasticsearch byl schopný vyhledávat.

Agregované metriky za určený interval lze posílat přímo z Logstashe (viz kapitola 4.2.1.4). Možností, kde metriky ukládat a jak je vizualizovat, je více. Jednou z alternativ je také samotný Elasticsearch. Skutečnost, že by stejné nástroje byly používány, jak pro LM, tak pro monitorování systému, by byla určitě výhodou, především z hlediska jednotné správy takového systému. James Bloomer (167) dokonce Elasticsearch doporučuje pro využití jako databázi časových řad.

Druhou možností, jak ukládat (agregované) metriky, je databáze přímo určená pro ukládání časových řad. Takovými databázemi jsou například již zmiňované nástroje Prometheus, Graphite nebo InfluxDB. Vizualizace metrik by následně bylo možné provést například v Kibaně nebo v Grafaně.

6 Shrnutí výsledků se srovnáním dosavadního řešení

Mobilní aplikace oneApp produkuje log obsahující velké množství různých dat. Obsahuje jak data, která se běžně v komunikačních lozích vyskytují, jako jsou informace o zdrojovém a cílovém serveru či úroveň závažnosti, tak i citlivé informace, jako například token nebo jméno uživatele. Pro proměnu surového logu v užitečné informace je nicméně žádoucí data zobrazovat a vizualizovat. Kritickými sledovanými informacemi pro TMCZ jsou podnikové metriky, které je potřeba z logu vyparsovat.

Kroky, které jsou nutné provést pro vyparsování kritických podnikových metrik, nejsou triviální. V rámci získávání podnikových metrik je nutné identifikovat a vyhodnotit celkovou provedenou akci, která je představována žádostí uživatele a odpovědí serveru. Základní postup je takový, že jednotlivé záznamy v logu jsou pospojovány do událostí a ty jsou následně korelovány přes identifikátor k získání dat celé akce. Nakonec je pomocí statistických operací a podmínečných vyhodnocení vypočítána požadovaná podniková metrika.

Všem definovaným požadavkům pro zpracování nestrukturovaného logu aplikace oneApp vyhověl z běžně používaných škálovatelných open-source nástrojů pro správu logů pouze Logstash. Ostatní srovnatelné nástroje nedisponují tak komplexní sadou pluginů pro transformaci logů jako Logstash.

Logstash logy zpracovává, ale neukládá je, ani je nevizualizuje. Pro vizualizaci logů byla zvolena Kibana, která je open-source nástrojem vydávaným od stejné organizace (firma Elastic) jako Logstash, čímž je zaručena vysoká kompatibilita. Do navrženého řešení byl také zahrnut Elasticsearch, který je neodmyslitelnou součástí, protože přijaté logy od Logstashe ukládá, indexuje a vyhledává v nich na základě příkazů provedených v Kibaně. I Elasticsearch je vydáván firmou Elastic.

Mimo prokázaní schopnosti navrhovaného řešení zpracovat log aplikace oneApp na reálných anonymizovaných datech s následnou vizualizací žádaných podnikových metrik bylo navrženo další rozšíření, které by toto řešení zplnohodnotnilo ve srovnání s tím dosavadním.

Logstash podporuje stejné vstupy, které byly využity v dosavadním řešení. Kibana umožňuje vizualizovat logy i z více datových streamů. V případě optimálního nazvání indexů lze tak učinit přímo v klasických modulech Kibany, v jiném případě lze využít flexibilnějších modulů podporujících zobrazení dat z vícero indexů. Data jsou agregována v Kibaně a detailní informace logu lze jednoduše zobrazit využitím základního zobrazení (tabulky) výčtu jednotlivých událostí spadajícího do zadaného časového či i jiného filtru.

Posílání výstrah a reportů není v bezplatné verzi v Kibaně k dispozici, ale je možné využít alternativních open-source řešení třetích stran, která přinášejí takřka stejnou funkcionalitu. Výstrahy je možné odesílat na základě obdobných pravidel a způsobem

jako v dosavadním řešení. Stejně jako v dosavadním řešení je možné zajistit generování reportů a jejich plánované odesílání na e-mail.

Agregace historických dat za dlouhé období je v navrženém řešení možné také, ovšem vyhledávání a vizualizace velkého množství dat může být výkonově náročné. V tomto řešení navíc není zohledněna možnost zahazování detailních informací a ponechání pouze agregovaných podnikových metrik.

Vhodnější by bylo agregované metriky ukládat odděleně, což by umožnilo také oddělenou správu životního cyklu detailních informací, respektive agregovaných metrik. Logstash podporuje zpracování agregovaných metrik v určité požadované granularitě a jejich odesílání. Zvolení konkrétní destinace, kde by byly metriky uloženy, by záleželo na preferencích zákazníka.

Kompletní navrhované řešení by ve výsledku přineslo obdobnou funkcionalitu jako dosavadní řešení. Přidanou hodnotou navrhovaného řešení je monitoring logů v reálném čase v rozšířeném moderním interaktivním prostředí a následně především možnost ad-hoc analýzy pomocí fulltextového vyhledávání.

Závěry a doporučení

Tato diplomová práce se zabývá výběrem open-source nástroje vhodného pro zpracování nestrukturovaných logů produkovaných konkrétní mobilní aplikací. Zpracovaná data je nástroj schopen dynamicky vizualizovat v reálném čase pomocí moderního uživatelského prostředí. Nástroj umožňuje provádět ad-hoc analytické dotazy i zobrazovat agregované podnikové metriky v čase. Výběr nástroje byl proveden na základě požadavků, které mají zajistit minimálně stejné funkcionality a výstupy, jako přináší řešení dosavadní.

Definovaným požadavkům, které především souvisely s transformací dat z prostého textu na podnikové metriky, vyhověl pouze jediný běžně rozšířený open-source nástroj pro správu logů – Logstash. Jako vizualizační nástroj byla vybrána Kibana spolu s podpůrnou databází Elasticsearch, která logy ukládá, indexuje a vyhledává v nich. S využitím těchto tří nástrojů, které jsou společně přezdívány akronymem ELK nebo pojmenováním Elastic Stack, bylo demonstrováno zpracování a vizualizace anonymizovaných logů za čtyři hodiny provozu aplikace zahrnující výpadek.

Dosavadní řešení monitoringu aplikace kromě zpracování a vizualizace logů podporuje také zasílání výstrah a reportů, které jsou při monitorování aplikací nesmírně důležité. Výstrahy hlídající hodnoty podnikových metrik upozorní na problém v reálném čase, načež jsou logy zásahovým technikem využity pro nalezení konkrétní příčiny daného problému. Reporty nabízí celkový náhled na funkčnost systému v čase, například pro management nebo pro oddělení řízení kvality. I navržené řešení využívající Elastic Stack dokáže zasílání výstrah a reportů implementovat, a to skrze doplňující open-source moduly třetích stran.

Protože Elastic Stack je velmi rozšířený a disponuje aktivní komunitou, zpravidla se nestává, že by neexistoval plugin či jiné řešení, jak přidat další zdroj, transformační funkcionalitu či výstupní destinaci. Další výhodou Elastic Stacku je jeho škálovatelnost, tudíž se při rostoucím objemu dat do jeho architektury mohou snadno zapojit nové uzly a nástroje. Navrhované řešení spolu s dalšími komponentami ať už přímo od Elasticu, nebo od třetích stran, dosavadní řešení plně nahrazuje či dokonce funkcionalitou převyšuje. Je ovšem nutné připomenout, že srdcem Elastic Stacku je Elasticsearch – databáze – kterou Kibana vyžaduje. Všechna data, která jsou v Kibaně vizualizována, je tak nutné mít v Elasticsearchi uložené. Záleží na zákazníkovi, zda takovou skutečnost považuje za výhodu, či nikoliv.

V případě potřeby je nicméně možné jakoukoliv komponentu Elastic Stacku částečně či úplně nahradit. Například Kibanu by bylo možné nahradit za jiný vizualizační nástroj, který umožňuje zobrazovat data i z dalších typů databází. Nebo pokud by největší nevýhoda Logstashe – výkonová náročnost – byla vyhodnocena pro podnik jako kritická, mohl by být nahrazen i ten. V tomto případě by ale bylo potřeba najít jiný vhodný komerční nástroj, nebo doplnit open-source nástroje o požadovanou transformační funkcionalitu.

Ve výsledku není tak důležité, jakým konkrétním řešením bude aplikace monitorována, ale že monitorována je. Logy jsou produkovány v masivním objemu a s rozšířením internetů věcí jich bude ještě více. Využití nástrojů pro správu, monitoring a analýzu logů je alfou a omegou kontroly a řízení hardwaru i softwaru. Bylo by vhodné, aby společnosti těchto nástrojů využily, a logy, které mohou být nositeli cenných informací, vědomě či nevědomě nezahazovaly.

Citovaná literatura

1. **Sint, Rolf, a další.** *Combining Unstructured, Fully Structured and Semi-Structured Information in Semantic Wikis.* Hersonissos, Greece : ESWC 2009, 2009.

2. **Arora, Yojna a Goyal, Dinesh.** *Review of Data Analysis Framework for Variety of Big Data. Emerging Trends in Expert Applications and Security.* 2018. stránky 55–62. DOI: 10.1007/978-981-13-2285-3_7.

3. **Chuvakin, Anton, Kevin J. Schmidt, Chris Phillips a Patricia Moulder.** *Logging and log management: the authoritative guide to understanding the concepts surrounding logging and log management.* Amsterdam : Elsevier/Syngress, 2013. ISBN 978-1-59749-635-3.

4. **Du, Min a Li, Feifei.** *Spell: Online Streaming Parsing of Large Unstructured System Logs.* [online] místo neznámé : IEEE Transactions on Knowledge and Data Engineering, 2018. DOI: 10.1109/TKDE.2018.2875442.

5. **Patel, Vimal Ashwinkumar.** *A system for analyzing applications. PCT/N2013/000532* Indie/Ahmedabad, 29. 10. 2015.

6. **Sridharan, Cindy.** Logs and Metrics. *Medium.com.* [Online] 30. 04. 2017. [Citace: 25. 02. 2019.] https://medium.com/@copyconstruct/logs-and-metrics-6d34d3026e38.

7. —. Monitoring in the time of Cloud Native. *Medium.com.* [Online] 04. 10. 2017. [Citace: 20. 10. 2019.] https://medium.com/@copyconstruct/monitoring-in-the-time-of-cloud-native-c87c7a5bfa3e.

8. **Sakr, Sherif a Zomaya, Albert Y.** *Encyclopedia of Big Data Technologies.* [online] Cham : Springer International Publishing, 2019. DOI: 10.1007/978-3-319-77525-8.

9. **Klipfolio Inc.** What are Business Metrics? *Klipfolio.com.* [Online] © 2019. [Citace: 25. 02. 2019.] https://www.klipfolio.com/resources/articles/what-are-business-metrics.

10. **Rouse, Margaret.** Distributed tracing. *Searchitoperations.techtarget.com.* [Online] 28. 04. 2018. [Citace: 25. 02. 2019.] https://searchitoperations.techtarget.com/definition/distributed-tracing.

11. **Shingala, Amit.** Why Use Log Management Tools? *Dzone.com.* [Online] 07. 09. 2017. [Citace: 25. 02 2019.] https://dzone.com/articles/why-use-log-management-tools.

12. **Chuvakin, Anton.** The Complete Guide to Log and Event Management. [Online] Květen 2016. [Citace: 15. 09. 2019.] https://www.netiq.com/en-au/docrep/documents/m47h82fbmy/the_complete_guide_to_log_and_event_management_wp_ap.pdf.

13. **van de Moosdijk, Jarno a Wagenaar, Daan.** Addressing SIEM. *Vurore.nl.* [Online] 30. 08. 2015. [Citace: 27. 10. 2019.] http://www.vurore.nl/images/vurore/downloads/scripties/2030-Def-scriptie-Jarno-van-de-Moosdijk---daan-Wagenaar.pdf.

14. **Williams, B. R., Chuvakin, A. A., & Milroy, D.** *PCI Compliance: Logging events and monitoring the cardholder data environment.* 5. Waltham : Syngress, 2015. stránky 197-234. ISBN 9780128015797.

15. **Sydor, Michael J.** *APM Best Practices.* [online] Berkeley, CA : Apress, 2011. ISBN 978-1-4302-3141-7.

16. **Churchman, Michael.** Log Aggregation vs. APM: No, They're Not the Same Thing. *Sumologic.com.* [Online] 27. 06. 2017. [Citace: 21. 10. 2019.] https://sumologic.com/blog/log-aggregation-vs-apm/.

17. **Sanz, Jorge Salamero.** How to instrument code: Custom metrics vs APM vs OpenTracing. [Online] 19. 09. 2018. [Citace: 09. 11. 2019.] https://sysdig.com/blog/how-to-instrument-code-custom-metrics-vs-apm-vs-opentracing/.

18. **Jayathilake, Dileepa.** *Towards structured log analysis.* [online] Colombo, Sri Lanka : 2012 Ninth International Conference on Computer Science and Software Engineering (JCSSE), 05 2012. ISBN 978-1-4673-1920-1.

19. **Mishra, Kundan Kumar a Kaul, Rahul.** Audit Trail Based on Process Mining and Log. *International Journal of Recent Development in Engineering and Technology.* [Online] 10. 2013. [Citace: 12. 11. 2019.] https://pdfs.semanticscholar.org/c45f/7df513b8f007f2557fe270c4932c8e209b50.pdf. ISSN 2347 – 6435.

20. **Abela, J. a Debeaupuis, T.** Universal Format for Logger Message. [Online] 31. 05 1999. [Citace: 14. 09 2019.] https://tools.ietf.org/html/draft-abela-ulm-05.

21. **Pasupuleti, Pradeep a Purra, Beulah Salome.** *Data Lake Development with Big Data.* Birmingham : Packt Publishing, 2015. Community Experience Distilled. ISBN 9781785888083.

22. **Nguyen, Thu.** Log Monitoring vs Log Analysis. [Online] 02. 11. 2018. [Citace: 09. 11. 2019.] https://logdna.com/log-monitoring-and-analysis/.

23. **Waterworth, Steve.** Observability vs. Monitoring. [Online] 19. 02. 2019. [Citace: 09. 11. 2019.] https://www.instana.com/blog/observability-vs-monitoring/.

24. **He, Pinjia, a další.** *Towards Automated Log Parsing for Large-Scale Log Data Analysis. IEEE Transactions on Dependable and Secure Computing.* [online] 2018. DOI: 10.1109/TDSC.2017.2762673.

25. **Oliner, Adam, Ganapathi, Archana a Xu, Wei.** *Advances and challenges in log analysis. Communications of the ACM.* [online] 2012. DOI: 10.1145/2076450.2076466.

26. **Lemoudden, Mouad a Ouahidi, Bouabid.** *Managing cloud-generated logs using big data technologies.* [online] Rabat : 2015 International Conference on Wireless Networks and Mobile Communications (WINCOM), 01. 10. 2015. DOI: 10.1109/WINCOM.2015.7381334.

27. **QI, Guanqiu, Wei-Tek TSAI, Wu LI, Zhiqin ZHU a Yong LUO.** A cloud-based triage log analysis and recovery framework. *Simulation Modelling Practice and Theory.* 2017, 77, stránky 292-316.

28. **Holubová, Irena, a další.** *Big Data a NoSQL databáze.* Praha : Grada, 2015. ISBN 978-80-247-5466-6.

29. **Edlich, Stefan.** Your Ultimate Guide to the Non-Relational Universe. [Online] [2009-2011]. [Citace: 10. 11. 2019.] http://nosql-database.org/.

30. **solid IT gmbh.** DB-Engines Ranking. *Db-engines.com.* [Online] © 2019. [Citace: 10. 11. 2019.] https://db-engines.com/en/ranking.

31. —. Document Stores. *Db-engines.com.* [Online] © 2019. [Citace: 10. 11. 2019.] https://db-engines.com/en/article/Document+Stores.

32. **Fluentd Project.** What is Fluentd? *Fluentd.org.* [Online] ©2010-2019. [Citace: 24. 02 2019.] https://www.fluentd.org/architecture.

33. —. Download Fluentd. [Online] ©2010-2019. [Citace: 04. 10. 2019.] https://www.fluentd.org/download.

34. —. Plugins. *Fluentd.org.* [Online] ©2010-2019. [Citace: 04. 10. 2019.] https://www.fluentd.org/plugins/all.

35. —. Frequently Asked Questions. *Fluentd.org.* [Online] ©2010-2019. [Citace: 04. 10. 2019.] https://www.fluentd.org/faqs#certified.

36. —. Why Use Fluentd? *Fluentd.org.* [Online] ©2010-2019. [Citace: 22. 09. 2019.] https://www.fluentd.org/why.

37. **StackShare, Inc.** Fluentd. *Stackshare.io.* [Online] © 2019. [Citace: 22. 09. 2019.] https://stackshare.io/fluentd.

38. **GitHub, Inc.** Fluent plugin event collector. *Github.com.* [Online] © 2019. [Citace: 03. 03. 2019.] https://github.com/adam-hart/fluent-plugin-event-collector.

39. **Fluentd Project.** Testimonials. *Fluentd.org.* [Online] ©2010-2019. [Citace: 22. 09. 2019.] https://www.fluentd.org/testimonials.

40. **Graylog Inc.** Log management for all. *Graylog.org.* [Online] © 2015-2018. [Citace: 24. 2. 2019.] https://www.graylog.org/products/open-source.

41. —. About Graylog. *Graylog.org.* [Online] © 2015-2018. [Citace: 04. 10. 2019.] https://www.graylog.org/about.

42. —. Frequently asked question. *Graylog.org.* [Online] © 2015-2019. [Citace: 30. 10. 2019.] https://docs.graylog.org/en/3.1/pages/faq.html.

43. —. Graylog Marketplace. *Marketplace.graylog.org.* [Online] © 2019. [Citace: 06. 03. 2019.] https://marketplace.graylog.org.

44. —. Sending in log data. *Graylog.org.* [Online] © 2015-2019. [Citace: 06. 10. 2019.] https://docs.graylog.org/en/3.1/pages/sending_data.html.

45. —. Graylog Documentation. *Docs.Graylog.org.* [Online] © 2015-2019. [Citace: 04. 10. 2019.] https://docs.graylog.org.

46. —. Graylog Enterprise Features. *Graylog.org.* [Online] © 2015-2018. [Citace: 04. 10. 2019.] https://www.graylog.org/features.

47. —. Download & Install. *Graylog.org.* [Online] © 2015-2018. [Citace: 04. 10. 2019.] https://www.graylog.org/downloads.

48. —. Open Source vs. Enterprise. *Graylog.org.* [Online] © 2015-2018. [Citace: 04. 10. 2019.] https://www.graylog.org/products/open-source-vs-enterprise.

49. **StackShare, Inc.** Graylog. *Stackshare.io.* [Online] © 2019. [Citace: 06. 10. 2019.] https://stackshare.io/graylog#stacks.

50. **GitHub, Inc.** Graylog. *GitHub.com.* [Online] © 2019 . [Citace: 06. 10. 2019.] https://github.com/Graylog2/graylog2-server?ref=stackshare.

51. **Elasticsearch B.V.** Logstash. *Elastic.co.* [Online] © 2019. [Citace: 23. 02. 2019.] https://www.elastic.co/products/logstash.

52. —. Support Matrix. *Elastic.co.* [Online] © 2019. [Citace: 06. 10. 2019.] https://www.elastic.co/support/matrix.

53. —. The Elastic Stack. *Elastic.co.* [Online] © 2019. [Citace: 23. 02. 2019.] https://www.elastic.co/products.

54. —. Elasticsearch. *Elastic.co.* [Online] © 2019. [Citace: 23. 02. 2019.] https://www.elastic.co/products/elasticsearch.

55. —. Kibana. *Elastic.co.* [Online] © 2019. [Citace: 23. 02. 2019.] https://www.elastic.co/products/kibana.

56. —. Beats. *Elastic.co.* [Online] © 2019. [Citace: 23. 02. 2019.] https://www.elastic.co/products/beats.

57. —. Deploying and Scaling Logstash. *Elastic.co.* [Online] © 2019. [Citace: 05. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/deploying-and-scaling.html#_network_and_security_data.

58. **StackShare, Inc.** Logstash. *Stackshare.io.* [Online] © 2019 . [Citace: 07. 08. 2019.] https://stackshare.io/logstash#stats.

59. **GitHub, Inc.** Logstash. *Github.com.* [Online] © 2019 . [Citace: 07. 08. 2019.] https://github.com/elastic/logstash.

60. **Elasticsearch B.V.** Scaling Log Aggregation at Fitbit. *Elastic.co.* [Online] © 2019. [Citace: 07. 09. 2019.] https://www.elastic.co/elasticon/conf/2018/sf/scaling-log-aggregation-at-fitbit.

61. —. Elastic Stack subscriptions. *Elastic.co.* [Online] © 2019. [Citace: 02. 10. 2019.] https://www.elastic.co/subscriptions.

62. —. We opened X-Pack. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/what-is/open-x-pack.

63. **One Identity LLC.** Syslog-ng Open Source Edition. *Syslog-ng.com.* [Online] © 2019. [Citace: 24. 02. 2019.] https://www.syslog-ng.com/products/open-source-log-management/#.

64. —. Universal log collection and routing. *Syslog-ng.com.* [Online] © 2019. [Citace: 10. 10. 2019.] https://www.syslog-ng.com/universal-log-collection-and-routing/.

65. —. Syslog-ng at a glance. *Syslog-ng.com.* [Online] © 2019. [Citace: 10. 10. 2019.] https://www.syslog-ng.com/products/.

66. **GitHub, Inc.** Syslog-ng. *GitHub.com.* [Online] © 2019. [Citace: 10. 10. 2019.] https://github.com/syslog-ng/syslog-ng.

67. **One Identity LLC.** Syslog-ng Open Source Edition installation packages. *Syslog-ng.com.* [Online] © 2019. [Citace: 10. 10. 2019.] https://www.syslog-ng.com/products/open-source-log-management/3rd-party-binaries.aspx.

68. —. Customer Success: Here you can see public references and download case studies. *Syslog-ng.com.* [Online] © 2019. [Citace: 10. 10. 2019.] https://www.syslog-ng.com/customer-stories/#%20.

69. —. Syslog-ng vs. rsyslog comparison. *Syslong-ng.com.* [Online] © 2019. [Citace: 06. 10. 2019.] https://www.syslog-ng.com/products/open-source-log-management/syslog-ng-rsyslog-comparison.aspx.

70. **Grafana Labs.** Loki. Prometheus-inspired logging for cloud natives. *Grafana.com.* [Online] © 2019. [Citace: 02. 10. 2019.] https://grafana.com/oss/loki.

71. **Splunk Inc.** *Splunk.* [Online] © 2005-2019. [Citace: 05. 10. 2019.] https://www.splunk.com/.

72. **Papertrail Inc.** Papertrail. [Online] © 2019. [Citace: 06. 10. 2019.] https://papertrailapp.com/.

73. **Logentries.com, Inc.** *Logentries.* [Online] ©2019. [Citace: 06. 10. 2019.] https://logentries.com/.

74. **Sumo Logic.** *Sumo Logic.* [Online] ©2019. [Citace: 06. 10. 2019.] https://www.sumologic.com/.

75. **Loggly, Inc.** *Loggly.* [Online] ©2019. [Citace: 06. 10. 2019.] https://www.loggly.com/.

76. **Stackify.** Get More Insights with Integrated Logging & Code Profiling. *Stackify.* [Online] © 2019. [Citace: 06. 10. 2019.] https://stackify.com/retrace-log-management/.

77. **OverOps, Inc.** Works the Way You Work. *OverOps.* [Online] ©2019. [Citace: 06. 10. 2019.] https://www.overops.com/integrations#logs.

78. **Datadog.** *Datadoghq.com.* [Online] © 2019. [Citace: 11. 10. 2019.] https://www.datadoghq.com.

79. **Gerhards, Rainer.** The Syslog Protocol. [Online] 03 2009. [Citace: 09. 11. 2019.] https://tools.ietf.org/html/rfc5424.

80. **Graylog Inc.** Sending in log data. *Greylog.org.* [Online] © 2015-2018. [Citace: 06. 03. 2019.] http://docs.graylog.org/en/2.5/pages/sending_data.html.

81. —. Graylog Sidecar. *Graylog.org.* [Online] © 2015-2017. [Citace: 06. 03. 2019.] http://docs.graylog.org/en/3.0/pages/sidecar.html.

82. **Fluentd Project.** Syslog Input Plugin. *Fluentd.org.* [Online] ©2010-2019. [Citace: 03. 03. 2019.] https://docs.fluentd.org/v1.0/articles/in_syslog.

83. —. Tail Input Plugin. *Fluentd.org.* [Online] ©2010-2019. [Citace: 03. 03. 2019.] https://docs.fluentd.org/v1.0/articles/in_tail.

84. **One Identity LLC.** Syslog-ng Open Source Edition 3.20 - Administration Guide. *Syslog-ng.com.* [Online] © 2019. [Citace: 09. 04. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.20/administration-guide#TOPIC-1121767.

85. **Elasticsearch B.V.** Syslog input plugin. *Elastic.co.* [Online] © 2019. [Citace: 01. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-inputs-syslog.html#_getting_help_42.

86. —. File input plugin. *Elastic.co.* [Online] © 2019. [Citace: 01. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-inputs-file.html.

87. **Fluentd Project.** Multiline Parser Plugin. *Fluentd.org.* [Online] ©2010-2019. [Citace: 03. 03. 2019.] https://docs.fluentd.org/v0.12/articles/parser_multiline.

88. **GitHub, Inc.** Fluent plugin grok parser. *Github.com.* [Online] © 2019. [Citace: 04. 03. 2019.] https://github.com/fluent/fluent-plugin-grok-parser.

89. **Elasticsearch B.V.** Multiline code plugin. *Elastic.co.* [Online] © 2019. [Citace: 02. 08. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-codecs-multiline.html.

90. **One Identity LLC.** Multi-line-mode. *Syslog-ng.com.* [Online] © 2019. [Citace: 02. 08. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.16/administration-guide/multi-line-mode.

91. **GitHub, Inc.** Fluent plugin concat. *Github.com.* [Online] © 2019. [Citace: 04. 03. 2019.] https://github.com/fluent-plugins-nursery/fluent-plugin-concat.

92. **Graylog Inc.** Support roll-up/correlation of messages. *Github.com.* [Online] © 2019. [Citace: 06. 03. 2019.] https://github.com/Graylog2/graylog2-server/issues/679.

93. **Elasticsearch B.V.** Aggregate filter plugin. *Elastic.co.* [Online] © 2019. [Citace: 01. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-filters-aggregate.html.

94. **One Identity LLC.** Correlating log messages. *Syslog-ng.com.* [Online] © 2019. [Citace: 02. 08. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.16/administration-guide/76#TOPIC-956675.

95. —. Correlating log messages using pattern databases. *Syslog-ng.com.* [Online] © 2019. [Citace: 02. 08. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.16/administration-guide/70#TOPIC-956650.

96. **GitHub, Inc.** Fluent plugin numeric monitor. *Github.com.* [Online] © 2019. [Citace: 04. 03. 2019.] https://github.com/tagomoris/fluent-plugin-numeric-monitor.

97. **Graylog Inc.** Announcing Graylog 3.1. *Graylog.org.* [Online] © 2015-2019. [Citace: 10. 09. 2019.] https://www.graylog.org/post/announcing-graylog-3-1.

98. —. Metrics Reporter Plugins. *Greylog.org.* [Online] © 2019. [Citace: 15. 03. 2019.] https://marketplace.graylog.org/addons/6fef88c7-94f7-488e-a6c5-bd6b71d8343e.

99. **Elasticsearch B.V.** Metrics filter plugin. *Elastic.co.* [Online] © 2019. [Citace: 04. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-filters-metrics.html.

100. —. Output plugins. [Online] © 2019. [Citace: 11. 10. 2019.] https://www.elastic.co/guide/en/logstash/current/output-plugins.html.

101. **One Identity LLC.** Hard vs. soft macros. *Syslog-ng.com.* [Online] © 2019. [Citace: 09. 04. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.16/administration-guide/58.

102. —. Administration Guide. *Syslog-ng.com.* [Online] © 2019. [Citace: 30. 10. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.24/administration-guide.

103. **Elasticsearch B.V. .** Aggregations. *Elastic.co.* [Online] © 2019. [Citace: 10. 10. 2019.] https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html.

104. **Elasticsearch B.V.** Business Critical Metrics & Insights. *Elastic.co.* [Online] © 2019. [Citace: 11. 10. 2019.] https://www.elastic.co/what-is/elasticsearch-business-analytics.

105. **The Graphite Project .** *Graphiteapp.org.* [Online] © 2011-2017. [Citace: 30. 10. 2019.] https://graphiteapp.org/.

106. **Prometheus Authors.** *Prometheus.io.* [Online] © 2014-2019. [Citace: 30. 10. 2019.] https://prometheus.io/.

107. **InfluxData Inc.** *Influxdata.com.* [Online] © 2019. [Citace: 11. 10. 2019.] https://www.influxdata.com/.

108. **Grafana Labs.** Using Elasticsearch in Grafana. *Grafana.com.* [Online] © 2018. [Citace: 11. 10. 2019.] https://grafana.com/docs/features/datasources/elasticsearch/.

109. **Graylog Inc.** Rules. *Graylog.org.* [Online] © 2015-2019. [Citace: 02. 08. 2019.] http://docs.graylog.org/en/stable/pages/pipelines/rules.html.

110. **Elasticsearch B.V.** Logstash Configuration Examples. *Elastic.co.* [Online] © 2019. [Citace: 04. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/config-examples.html.

111. **One Identity LLC.** Conditional expressions. [Online] © 2019. [Citace: 11. 10. 2019.] https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.16/administration-guide/48#TOPIC-956568.

112. **GitHub, Inc.** ConditionalFilter, a plugin for Fluentd. [Online] © 2019. [Citace: 11. 10. 2019.] https://github.com/kentaro/fluent-plugin-conditional_filter#conditionalfilteroutput.

113. **Fluentd Project.** Rewrite_tag_filter. [Online] ©2010-2019. [Citace: 11. 10. 2019.] https://docs.fluentd.org/output/rewrite_tag_filter.

114. —. Routing Examples. [Online] ©2010-2019. [Citace: 11. 10. 2019.] https://docs.fluentd.org/configuration/routing-examples.

115. **Bragin, Tanya.** Using Painless in Kibana scripted fields. *Elastic.co.* [Online] 13. 12. 2016. [Citace: 04. 03. 2019.] https://www.elastic.co/blog/using-painless-kibana-scripted-fields.

116. **Graylog Inc.** Issues with date conversion in pipeline. *Graylog.org.* [Online] © 2015-2019. [Citace: 03. 08. 2019.] https://community.graylog.org/t/issues-with-date-conversion-in-pipeline/8627.

117. **Elasticsearch B.V.** Elapsed filter plugin. *Elastic.co.* [Online] © 2019. [Citace: 04. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-filters-elapsed.html.

118. —. Elasticsearch filter plugin. *Elastic.co.* [Online] © 2019. [Citace: 01. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-filters-elasticsearch.html.

119. —. Ruby filter plugin. *Elastic.co.* [Online] © 2019. [Citace: 10. 10. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-filters-ruby.html.

120. **Treasure Data.** Fluentd and Fluent Bit. *Fluentbit.io.* [Online] © 2015-2017. [Citace: 03. 03. 2019.] https://fluentbit.io/documentation/0.11/about/fluentd_and_fluentbit.html.

121. **Fluentd Project.** Store Apache Logs into MongoDB. *Fluentd.org.* [Online] ©2010-2019. [Citace: 03. 03. 2019.] https://docs.fluentd.org/v1.0/articles/apache-to-mongodb.

122. —. Fluentd + HDFS: Instant Big Data Collection. *Fluentd.org.* [Online] © 2010-2019. [Citace: 03. 03. 2019.] https://docs.fluentd.org/v1.0/articles/http-to-hdfs.

123. —. Multiprocess Input Plugin. *Fluentd.org.* [Online] ©2010-2019. [Citace: 03. 03. 2019.] https://docs.fluentd.org/v0.12/articles/in_multiprocess.

124. —. What is Fluentd? [Online] ©2010-2019. [Citace: 03. 08. 2019.] https://www.fluentd.org/architecture.

125. **Graylog Inc.** Architecture. [Online] © 2015-2018. [Citace: 11. 10. 2019.] http://docs.graylog.org/en/2.4/pages/architecture.html.

126. **One Identity LLC.** What syslog-ng relays are good for. [Online] © 2019. [Citace: 11. 10. 2019.] https://www.syslog-ng.com/community/b/blog/posts/what-syslog-ng-relays-are-good-for.

127. —. Big data ingestion. [Online] © 2019. [Citace: 11. 10. 2019.] https://www.syslog-ng.com/big-data-ingestion/.

128. **Redash.** *Redash.io.* [Online] © 2019. [Citace: 11. 10. 2019.] https://redash.io/.

129. **Elasticsearch B.V.** *Elastic.co.* [Online] © 2019. [Citace: 31. 10. 2019.] https://www.elastic.co/.

130. **GitHub, Inc.** Elastic. *Github.com.* [Online] © 2019. [Citace: 31. 10. 2019.] https://github.com/elastic.

131. **Turnbull, James.** *The Logstash Book.* [E-kniha] místo neznámé : Amazon Digital Services LLC, 2013. ASIN B00B9JQTCO.

132. **Puppet.** *Puppet.com.* [Online] © 2019. [Citace: 15. 03 2019.] https://puppet.com/.

133. **Chef Software, Inc.** *Chef.io.* [Online] © 2008 – 2019. [Citace: 15. 03. 2019.] https://www.chef.io/.

134. **Elasticsearch B.V.** Logstash Directory Layout. *Elastic.co.* [Online] © 2019. [Citace: 09. 08. 2019.] https://www.elastic.co/guide/en/logstash/current/dir-layout.html#dir-layout.

135. —. Logstash.yml. *Elastic.co.* [Online] © 2019. [Citace: 09. 08. 2019.] https://www.elastic.co/guide/en/logstash/current/logstash-settings-file.html.

136. —. Setting JVM Options. *Elastic.co.* [Online] © 2019. [Citace: 09. 08. 2019.] https://www.elastic.co/guide/en/elasticsearch/reference/current/jvm-options.html.

137. —. Input plugins. *Elastic.co.* [Online] © 2019. [Citace: 03. 11. 2019.] https://www.elastic.co/guide/en/logstash/current/input-plugins.html.

138. —. Filter plugins. [Online] © 2019. [Citace: 03. 11. 2019.] https://www.elastic.co/guide/en/logstash/current/filter-plugins.html.

139. **GitHub, Inc.** Grok patterns. *Github.com.* [Online] © 2019. [Citace: 14. 08. 2019.] https://github.com/elastic/logstash/blob/v1.4.2/patterns/grok-patterns.

140. **Störr, Hans-Peter.** Test grok patterns. *Grokconstructor.appspot.com.* [Online] nedatováno. [Citace: 16. 08. 2019.] http://grokconstructor.appspot.com/do/match#result.

141. **Elasticsearch B.V.** Introducing Multiple Pipelines in Logstash. *Elastic.co.* [Online] © 2019. [Citace: 09. 08. 2019.] https://www.elastic.co/blog/logstash-multiple-pipelines.

142. —. Basic Concepts. *Elastic.co.* [Online] © 2019. [Citace: 20. 03. 2019.] https://www.elastic.co/guide/en/elasticsearch/reference/6.2/_basic_concepts.html.

143. —. Removal of mapping types. *Elastic.co.* [Online] © 2019. [Citace: 20. 03. 2019.] https://www.elastic.co/guide/en/elasticsearch/reference/6.2/removal-of-types.html.

144. —. Scalability and resilience: clusters, nodes and shards. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/guide/en/elasticsearch/reference/current/scalability.html.

145. —. How many shards should I have in my Elasticsearch cluster? *Elastic.co.* [Online] © 2019. [Citace: 20. 03. 2019.] https://www.elastic.co/blog/how-many-shards-should-i-have-in-my-elasticsearch-cluster.

146. —. Every shard deserves a home. *Elastic.co.* [Online] © 2019. Elasticsearch B.V. [Citace: 30. 10. 2019.] https://www.elastic.co/blog/every-shard-deserves-a-home.

147. —. Configuring Elasticsearch. *Elastic.co.* [Online] © 2019. [Citace: 18. 03. 2019.] https://www.elastic.co/guide/en/elasticsearch/reference/current/settings.html.

148. —. Timelion. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/guide/en/kibana/current/timelion.html.

149. —. Canvas. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/guide/en/kibana/current/canvas.html.

150. —. Elastic metrics. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/products/infrastructure-monitoring#instructions.

151. —. Elastic logs. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/products/log-monitoring.

152. —. APM overview. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/guide/en/apm/get-started/current/overview.html.

153. —. Machine Learning. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/guide/en/kibana/current/xpack-ml.html.

154. —. Installing Logstash. *Elastic.co.* [Online] © 2019. [Citace: 13. 03. 2019.] https://www.elastic.co/guide/en/logstash/current/installing-logstash.html#package-repositories.

155. —. Connect to Elasticsearch. *Elastic.co.* [Online] © 2019. [Citace: 16. 08. 2019.] https://www.elastic.co/guide/en/cloud/current/ec-getting-started-connect.html#ec-getting-started-transport-client.

156. —. Install Kibana with Debian Package. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/guide/en/kibana/current/deb.html.

157. **GitHub, Inc.** Awesome Elasticsearch. *Github.com.* [Online] © 2019. [Citace: 02. 11. 2019.] https://github.com/dzharii/awesome-elasticsearch.

158. —. Datasweet Formula. *Github.com.* [Online] © 2019. [Citace: 02. 11. 2019.] https://github.com/datasweet/kibana-datasweet-formula.

159. **Elasticsearch B.V.** Alerting. *Elastic.co.* [Online] © 2019. [Citace: 21. 08. 2019.] https://www.elastic.co/what-is/elasticsearch-alerting.

160. —. Email plugin output. *Elastic.co.* [Online] © 2019. [Citace: 21. 08. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-outputs-email.html.

161. —. Exec ouput plugin. *Elastic.co.* [Online] © 2019. [Citace: 03. 11. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-outputs-exec.html.

162. **Yelp.** ElastAlert - Easy & Flexible Alerting With Elasticsearch. *Elastalert.readthedocs.io.* [Online] © 2014. [Citace: 21. 08. 2019.] https://elastalert.readthedocs.io/en/latest/elastalert.html#overview.

163. **GitHub, Inc.** ElastAlert Kibana Plugin . *Github.com.* [Online] © 2019. [Citace: 21. 08. 2019.] https://github.com/bitsensor/elastalert-kibana-plugin.

164. —. Elastalert. *Github.com.* [Online] © 2019. [Citace: 21. 08. 2019.] https://github.com/Yelp/elastalert.

165. **Elasticsearch B.V.** Reporting. *Elastic.co.* [Online] © 2019. [Citace: 01. 11. 2019.] https://www.elastic.co/what-is/kibana-reporting.

166. **GitHub, Inc.** Sentinl. *Github.com.* [Online] © 2019. [Citace: 02. 11. 2019.] https://github.com/sirensolutions/sentinl.

167. **Bloomer, James.** Choosing the Elastic stack as a time series database. *Medium.com.* [Online] 07. 08. 2018. [Citace: 02. 11 2019.] https://medium.com/kudos-engineering/choosing-the-elastic-stack-as-a-time-series-database-9fac202c53ba.

168. **The Apache Software Foundation.** Welcome to Apache Lucene. *Apache.org.* [Online] © 2011-2016. [Citace: 13. 03. 2019.] http://lucene.apache.org/.

169. **IT-Slovnik.cz team.** Spoofing. *It-slovnik.cz.* [Online] © 2008 - 2018. [Citace: 15. 03. 2019.] https://it-slovnik.cz/pojem/spoofing.

170. **Free Software Foundation, Inc.** Various Licenses and Comments about Them. *Gnu.org.* [Online] © 2019. [Citace: 21. 08. 2019.] https://www.gnu.org/licenses/license-list.html.en#ModifiedBSD.

171. **Nitemedia s. r. o.** Tarball. *Abclinuxu.cz.* [Online] © 1999-2015. [Citace: 13. 03. 2019.] http://www.abclinuxu.cz/slovnik/tarball.

172. **Elasticsearch B.V. .** Grok filter plugin. *Elastic.co.* [Online] © 2019. [Citace: 30. 10. 2019.] https://www.elastic.co/guide/en/logstash/current/plugins-filters-grok.html.

173. **Mauro, Douglas R. a Schmidt, Kevin J.** *Essential SNMP.* 2. Sebastopol, CA : O'Reilly, 2005. ISBN 978-0596008406.

174. **Gála, Libor, Pour, Jan a Šedivá, Zuzana.** *Podniková informatika: počítačové aplikace v podnikové a mezipodnikové praxi.* Management v informační společnosti. Praha : Grada Publishing, 2015. ISBN 978-80-247-5457-4.

175. **Oracle.** Java SE Technologies - Database. *Oracle.com.* [Online] © 2019. [Citace: 03. 11. 2019.] https://www.oracle.com/technetwork/java/javase/jdbc/index.html#corespec40.

176. **Mueen, Abdullah, a další.** Exact Discovery of Time Series Motifs. [Online] 25. 05 2010. [Citace: 09. 11. 2019.] https://web.archive.org/web/20100625200233/https://www.cs.ucr.edu/~eamonn/EM.pdf.