SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Dokumentácia k inžnierskemu dielu

Tímový projekt II - Koniec LS

Číslo a názov tímu:	14. - SecMon
Členovia tímu:	Bc. Norbert Danišik, Bc. Matej Guráň, Bc. Matej Puk, Bc. Matúš Jurika, Bc. Roland Lang, Bc. Štefan Kadlic
Akademický rok:	2016/2017
Vedúci tímu:	Ing. Ján Laštinec, PhD.

Obsah

1 Úvod

V rámci predmetu Tímový projekt I a Tímový projekt II v akademickom roku 2016/2017 sme sa ako tím číslo 14 predstavili pod názvom Talented Otters s projektom SecMon. Tento dokument prináša celkový pohľad na inžinierske dielo, ktoré v rámci tímového projektu produkujeme ako tím a odráža náležitosti návrhu a vývoja jednotlivých častí projektu. Názov projektu SecMon vyplvýva z holej podstaty problému, ktorý riešime. SEC je open-source nástroj na analýzu zozbieraných logov zo siete a ich koreláciu v kontexte informačnej bezpečnosti. Druhá časť názvu Mon naznačuje, že sa podieľame na vytvorení monitorovacieho nástroja, ktorý prináša grafické používateľské rozhranie pre prácu s nástrojom SEC. Okrem toho prináša oveľa viacej výhod ako napríklad filtrovanie udalostí, ukladanie korelovaných udalostí, vytváranie videní a iné.

Jednotlivé kapitoly dokumentu budú obsahovať globálne ciele a celkový pohľad na systém ako taký, opis architektúry a dátový model systému.

Taktiež súčasťou tohto dokumentu je aj opis modulov, ktoré sú súčasťou vytvoreného softvéru. V druhom kontrolnom bode bola pridaná aj technická dokumentácia a inštalačná príručka pre server.

V letnom semestri sme doplnili časti dokumentácie, ktoré už boli vypracované a dodali sme ďalšie časti, ktoré súvisia s middlewarom a normalizátorom. Taktiež sme pridali správu o testovaní produktu treťou stranou a značne rozšírili používateľskú príručku pre prácu s frontendom.

2 Globálne ciele

Na začiatku pri počiatočnom určovaní cieľov je potrebné si uvedomiť, že tento dvojsemestrálny predmet je určený nielen na vyprodukovanie funkčného a správne integrovaného softvéru, ale aj na vyskúšanie si práce vo väčšom tíme a manažmente jednotlivých častí tímového a vývojového ekosystému. Preto okrem zamerania sa na vývoj produktu zadaného naším zákazníkom(vedúcim tímu) sa sústredíme aj na realizovanie sa v rámci tímu a úloh manažmentu tímu.

2.1 Globálne ciele pre zimný semester

Naším primárnym cieľom v rámci zimného semestra je verzia produktu systému SecMon, ktorá bude schopná prijímať vstupné logy, správne ich korelovať a kategorizovať a bude ponúkať základné náhľady na zanalyzované dáta. Tento hlavný cieľ bolo potrebné rozdeliť na ciele s menšou granularitou, aby bol vývoj lepšie sledovateľný a tie ciele sú:

1. **Nástroj SEC** - na začiatku vývoja bolo potrebné sa zoznámiť s nástrojom, ktorý tvorí srdce nášho systému. Tento nástroj je open-source, čiže máme k dispozícii aj jeho zdrojový kód, čo nám pomohlo ho lepšie pochopiť a pripraviť ekosystém schopný sa tomuto nástroju prispôsobiť.

2. **Návrh architektúry systému** – keďže SEC je nástroj, ktorý je vo svojej podstate samobežiaci na pozadí, je potrebné vyriešiť netriviálny problém kooperácie s webovým rozhraním, ktoré bude slúžiť ako ovládací panel pre používateľa, aby vedel nastavovať SEC nástroj a efektívne zobrazovať zozbierané údaje. Preto je kľúčové navrhnúť architektúru systému, ktorá bude spĺňať konkrétne náležitosti na správny chod celého ekosystému.

3. **Návrh a realizácia používateľského rozhrania** - problém, ktorý sa snažíme ako tím vyriešiť prináša široké vizuálne možnosti realizovania sa. Používateľovi systému je potrebné zobrazovať obrovské množstvo dát a je kľúčové zabezpečiť, aby jeho pozornosť bola upriamovaná na problém, ktorý vzniká/vznikol. Preto

aplikácia by mala poskytovať dostatočne prehľadné a jednoduché používateľské rozhranie, ktoré ale musí spĺňať podmienky vysokej efektivity pri práci. Toto sa budeme snažiť dosiahnuť čiastočným prototypovaním, pričom budeme zbierať spätnú väzbu od zákazníka a postupne zapracovávať jeho pripomienky do tvorby softvéru.

4. **Rozšírená funkcionalita softvéru** - tento systém by mal byť schopný okrem jednoduchého zobrazovania korelovaných a nekorelovaných udalostí v reálnom čase aj zobrazovať rôzne aplikované filtre a videnia na dané informácie.Jednoduchý graf a tabuľky s dátami budú aplikované ku koncu zimného semestra a tieto úlohy tejto kategórie sa budú prelínať aj do semestra letného.

2.2 Globálne ciele pre letný semester

Keď sa pozrieme na globálne ciele za uplynulý zimný semester, veľa z nich bolo oveľa náročnejšie dosiahnuť, ako sme si mysleli a preto niektoré z nich neboli naplnené, resp. boli naplnené iba z časti. Globálne ciele pre letný semester sa týkajú predovšetkým toho, aby sme boli schopní dodať funkčný produkt vo svojej holej podstate, aby ho bolo možné využívať v konkrétnom prostredí a aby vedel korelovať úspešne aspoň dva druhy udalostí. Preto ciele na letný semester sme si mohli rozdeliť na tieto menšie čiastkové ciele:

1. **Nástroj SEC** - po zoznámení s nástrojom SEC by sme v tomto semestri mali vyprodukovať pravidlá pre minimálne 2 typy odchytávaných bezpečnostných udalostí (napr. SSH alebo IPTables). Taktiež by bolo dobré, aby bolo možné SEC nastavovať priamo z používateľského rozhrania, no táto funkcionalita je skôr radená medzi "enhacementy", ktoré sa budú robiť, až keď všetko ostatné bude hotové.

2. **Implementácia Middlewaru** – middleware je jadrom celého nášho produktu a preto je kľúčové okolo neho postaviť veľmi dobrú architektúru, ktorú sme si navrhli a sprototypovali v zimnom semestri. Middleware by mal byť schopný čítať z viacerých named pipes a mal by byť schopný procesovať niekoľko desiatok logov za sekundu.

3. **Zveľaďovanie používateľského rozhrania** - používateľské rozhranie je vždy ako samostatne žijúca entita, ktorá sa veľmi dynamicky mení. Preto je veľmi tažké odhadhnúť, čo všetko je potrebné ešte spraviť, no v rámci globálnych cieľov chceme dosiahnuť čo najlepší zážitok pre používateľa v kombinácii s UX doladeným rozhraním, ktoré bude okrem vizuálne dobrej implementácie aj použiteľné z hľadiska UX.

4. **Zlepšovanie filtrov** - ako bolo vyššie spomenuté, implementácia filtrov a kombinujúce sa podmienky zobrazovania sa začali v istej miere implementovať už v zimnom semestri, pričom v letnom semestri sa v tomto cieli naďalej pokračuje.

5. **Dashboardy a videnia** - implementácia dashboardov, ktoré budú obsahovať jednotlivé videnia alebo komponenty, ktoré budú samostatne dragovateľné a pohyblivé po obrazovke, taktiež ako sa bude dať meniť aj ich veľkosť. Do týchto komponentov bude možné vkladať grafy alebo tabuľky, ktoré budú taktiež veľkosťou prispôsobiteľné.

3 Celkový pohľad na systém

Táto kapitola prináša pohľad na technickú stránku nášho projektu, predovšetkým na návrh architektúry a dátového modelu, ktorý predstavuje kľúčový odrazový mostík pre správny návrh entít používateľov, udalostí, logov, filtrov a videní. Tieto entity sú poprepájané vzťahmi, ktoré sú určujúce pre správny chod celého systému.

3.1 Architektúra systému

O správnom návrhu architektúry systému sa písalo aj v predošlých kapitolách, a teda je bolo potrebné navrhnúť prvky architektúry tak, že sústredenie bolo zamerané na faktor kooperácie s nástrojom SEC. Tento nástroj je funkčný iba na distribúciách operačného systému Linux. Projekt je vhodné realizovať ako webovú aplikáciu.

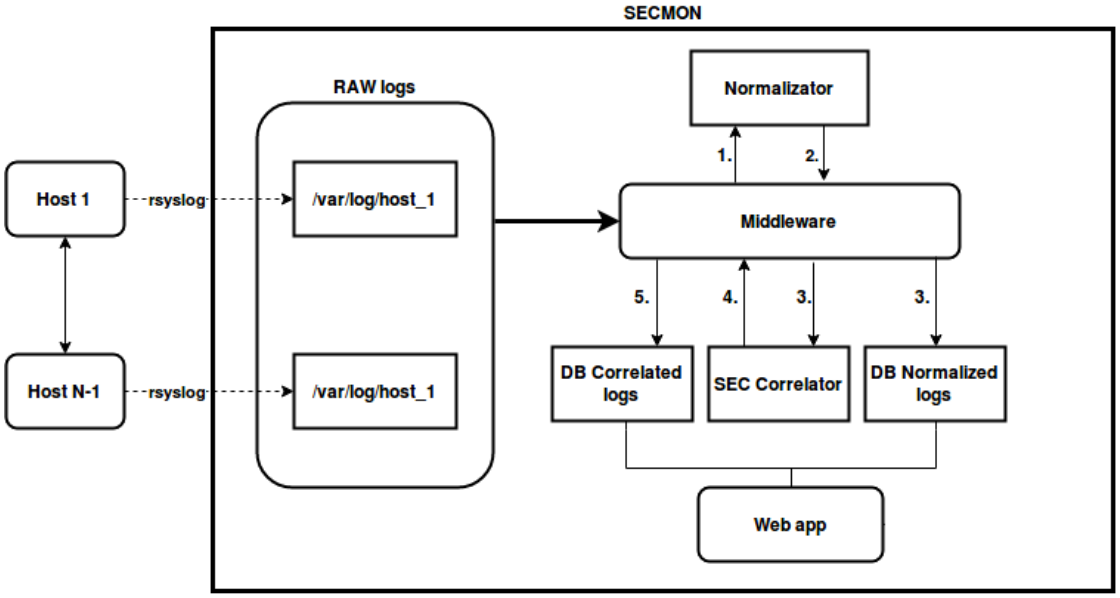
Ako operačný systém pre náš server sme zvoli linuxovú distribúciu s názvom CentOS, ktorá hostí Apache server. Funkcionalita nástroja SEC je na tomto operačnom systéme plne funkčná a bezproblémovo komunikuje aj s našou databázou, ktorá je typy PostgreSQL. Surové logy sa budú ukladať taktiež do PostgreSQL databázy pre potreby budúceho dohľadania.

V rámci vývoja a integrácie systému nástroj na správu verzií GIT a službu na manažovanie zmien v kóde s názvom GitHub.

Naša webová aplikácia je realizovaná pomocou klasického návrhového vzoru Model-View-Controller za réžie rámca(frameworku) s názvom YII, ktorý je napísaný v jazyku PHP. Využívame najnovšiu verziu jazyka PHP 7, ktorá podporuje všetky významné moderné princípy tvorby webov. Používateľské rozhranie je tvorené modulmi Views v rámci frameworku YII, ktoré sú dodefinované CSS štýlmi, ktoré vyvíjame pomocou technológie SASS, čo poskytuje elegantné a rýchle riešenie pre vizuál frontendu. Taktiež celá aplikácia bude ladená v štýle moderného a prehľadného *Material designu*. Interaktivita v prehliadači je zabezpečená okrem bežne dostupných Bootstrap javascriptových knižníc aj vďaka Packery.IO - knižnici na resizovanie a pohyb elementov po

gride. Tieto elementy môžu obsahovať grafy, ktoré sú realizované pomocou knižnice pre Javasript s názvom D3js. Správa dát vo webovej aplikácii je realizovaná pomocou technológie JSON, ktorá ponuka vysokú flexibilitu, ktorá je potrebná nakoľko pracujeme s dátami, ktoré môžu byť rôzneho typu, ale je potrebné ich z dôvodov ukladania a správy generalizovať.

Globálny pohľad na architektúru systému na obrázku 3.1 spočíva v tom, že by sa dala rozdeliť na viacero samostatných modulov, ktoré by ale mali spolu kooperovať. Nástroj SEC je spúšťaný skriptom, ktorý zabezpečuje odchytávanie výstupu z nástroja SEC a reportovanie výsledkov do databázy a ukladanie surových logov do PostgreSQL databázy. Backend webovej aplikácie tieto reporty spracúva z databázy a zobrazuje v konkrétnych moduloch, ktoré sú súčasťou frontendovej časti aplikácie.



Obr. 3.1: Architektúra systému

Používatelia sú celkovo rozdelení na dva druhy:

- Administrátor - tento typ používateľa môže upravovať nastavenia nástroja SEC a pridávať mu nové monitorovacie pravidlá, spravovať staré pravidlá alebo využívať funkcionalitu, na ktorú sú oprávnení aj bežní používatelia.
- Bežný používateľ - môže si vyberať filtre na zobrazenia logov, zobrazovať rôzne sledované videnia, ktoré si vyberie zo zoznamu, prípadne sledovať udalosti v rôznych typoch grafov. Tieto nastavenia si môže jednoducho uložiť a opäť sa k nim hocikedy vrátiť.

Funkcionalita a architektúra systému je založená na princípoch efektívneho prezentovania výsledkov používateľovi a intuitívnej práce s rozhraním. Rámec YII poskytuje veľa možností implementácie rôznych modulov, ktoré zabezpečujú plynulý vývoj a prudko stúpajúcu krivku učenia. Idea nášho projektu je taká, že softvér poskytne bežnému používateľovi vysoko efektívne, ale jednoduché možnosti práce s aplikáciou, ale na druhej strane, pokročilému používateľovi alebo administrátorovi možnosť plnohodnotne pracovať s nástrojom SEC a využívať všetky jeho výhody naplno.

3.2 Dátový model

Vizualizácia dátového modelu s poprepájaním jednotlivých elementov medzi sebou je priložená ako príloha A. Najdôležitejšie súčasti dátového modelu - entity sú:

- User – používateľ, ktorý je jednoznačne určený používateľským menom, heslom a rolou, ktorá predurčuje jeho funkciu a možnosti realizácie sa v systéme.
- Roles and Permissions – slúžia na odlíšenie používateľských kategórií a dovoľujú im určovať špecifické pravidlá aplikujúce sa na konkrétne roly.
- Videnie(View) – každý používateľ si môže personalizovať svoje používateľské videnie, čo predstavuje nejaké konkrétne a personalizované rozmiestnenie jednotlivých komponentov na obrazovke, pričom je ich možné spravovať a filtrovať pomocou filtrov zobrazení.
- Filters – filtre poskytujú možnosti realizácie používateľa v rámci vizualizácií udalostí v systéme. Filtrami je možné nastaviť jednotlivé komponenty a zamerať sa na konkrétne udalosti alebo to, čo ich vyvolalo.
- Filter Rules - pravidlá filtrov sú pridávané do databázy a priraďované jednotlivým entitám typu filter, ktoré sú potom následne priraďované ku komponentom.
- Komponenty - komponent je súčasťou videnia. Každé videnie môže mať n komponentov.
- Events_correlated – udalosť je základná dátová entita, ktorú sledujeme a vizualizujeme v rámci nášho projektu. Je dôležité aby udalosti boli jasne komunikované používateľovi, aby mohol flexibilne reagovať na výskyt neželanej aktivity v

systéme alebo sieti. Tieto udalosti sú generované nástrojom SEC z logov, ktoré zachytávajú sieťové prvky alebo iné komunikačné systémy.

- Events_normalized – log je základná informačná jednotka aktivity komunikačných systémov v rámci počítačovej siete. Logy sú generované konkrétnymi mechanizmami operačného systému a spracovávané nástrojom SEC, ktorý ich normalizuje (tento SEC = normalizátor) a potom putujú do SEC (tento SEC = korelátor).

- SEC_Rule – pravidlá pre nástroj SEC sú dôležitou súčasťou celého dátového modelu, pretože predstavujú spúšťač korelovania logov do udalostí na základe práve týchto pravidiel. SEC pravidlá je možné nastavovať, dopĺňať, ale aj mazať a upravovať.

Aktuálne zapracovanie a stav dátového modelu v databáze je možné sledovať v kapitole Technická dokumentácia, odsek Backend.

4 Moduly systému

V tejto kapitole dokumentu sú načrtnuté moduly systému a ku každému z nich je uvedený bližší popis ohľadom analýzy, návrhu, implementácie a testovania.

4.1 Práca nástroja SEC

Analýza

Srdcom a hnacím motorom nášho pripravovaného systému je korelačný nástroj s názvom SEC, ktorý je potrebné spravovať v príkazovom riadku. SEC analyzuje zozbierané logy v reálnom čase a produkuje výstup v tvare korelovanej udalosti.

Návrh

Do úvahy prichádzajú dve možnosti a to:

1. SEC bude bežať na každom stroji a logy bude analyzovať v rámci jednotlivých strojov a výsledky odosielať do centrálnej databázy.
2. SEC bude bežať v jedinej inštancii na centrálnom serveri, na ktorý budú odosielané informácie(logy) z jednotlivých strojov a tam sa budú hromadne analyzovať.

Rozhodli sme sa pre možnosť číslo 2, preto bolo potrebné zabezpečiť zber a ukladanie údajov.

Implementácia

Návrh sa pretavil do implementácie v podobe využitia vopred vstavaných linuxových možností odosielania logov ako napr. syslog. Pomocou takéhoto protokolu sa na centrálnom serveri zhromažďujú logy a udalosti, ktoré skript, ktorý manažuje nástroj SEC, následne po jeho spracovaní uloží do databázy, prípadne odošle konkrétnemu kontroleru backendu. Surové logy sa budú ukladať do PostgreSQL databázy pre búduce potreby dohľadávania súvisiacich udalostí.

Testovanie

Testovanie prebieha aplikovaním testovacích súborov sieťovej aktivity, ktoré sa podsúvajú SEC nástroju a analyzuje sa správna funkcionalita na základe zadaných obmedzení a filtrov.

4.2 Normalizátor logov

Analýza

Zariadenia, na ktoré sú uskutočňované útoky, sú často od rôznych posktyovateľov služieb a výrobcov, preto je potrebné zjednotiť formát spracovávaných logov pre efektívnejšiu prácu s nimi pomocou SECu a lepšou modularitou systému. Existuje viacero širšie používaných formátov, ktoré ale nie sú zjednoteným štandardom. Často každý výrobca preferuje vlastný tvar logov, ktoré sa navzájom nemusia zhodovať.

Návrh

Návrhom riešenia normalizátora je utilita, ktorá pre SEC predspracúva logy do jednotného formátu.V rámci návrhu sme dospeli k názoru, že budeme zbierať logy v tvare CEF - Common Event Format. V prípade, že zdrojové zariadenie nebude vedieť odosielať logy v tomto tvare, aplikuje sa normalizátor logov.

Implementácia

Normalizátor bude jedna inštancia SECu, ktorý predspracováva logy, ktoré nie sú v dohodnutom formáte a pomocou regexov ich parsuje na tento formát. Vstupom pre normalizátor bude typ vstupného logu a samotné logy, výstupom logy vo formáte CEF(prípadne inom vopred dohodnutom formáte).

Testovanie

Normalizátor sme testovali pomocou testovacích skriptov, ktorými sme tlačili dáta do normalizátora a potom sme sledovali výstup, či normalizátor správne pracuje.

4.3 Middleware a kontroléry

Webová aplikácia musí byť riadená logikou, ktorú zabezpečuje správne fungujúci backend aplikácie.

Analýza

SEC nástroj nie je natoľko sofistikovaný, aby ním bolo možné riadiť celkový prenos údajov v rámci celého backendu od spracovania až po zobrazenie údajov.

Návrh

Celý proces riadenia SEC nástroja je zabezpečený middlewarom, ktorý je realizovaný ako komponent v jazyku PHP. Zozbierané dáta o udalostiach sa ukladajú do databázy, pričom je potrebné ukladať hierarchicky aj jednotlivé logy, ktoré boli spúšťačmi, resp. zdrojom korelovaných udalostí. Tieto dáta je potrebné interpretovať frontendu a vykreslovačom dát v prehliadači.

Implementácia

Skript, ktorý riadi celý proces je napísaný v skriptovacom jazyku PHP, aby bola zabezpečená čo najlepšia kompatibilita s celým backendovým riešením, ktoré je realizované v YII. Udalosti a ich informácie sú ukladané do PostgreSQL databázy. Logy budú hierarchicky ukladané do textových súborov, ktorých referencie sa tiež budú ukladať do relačnej databázy. Tieto dáta sú interpretované vstavanými kontrolermi rámca YII, ktoré sa majú nakonfigurované konektory na relačnú databázu.

Testovanie

Testovanie tejto sekcie je veľmi náročné a vyžaduje si viacero stupňov testovania. Jednak sa testuje správnosť fungovania middlewaru ako samostatného komponentu. Taktie je treba jednoduchými validačnými queries kontrolovať nainsertované dáta do databázy a ich previazanie pomocou private a foreign kľúčov.

4.4 Správa používateľov a nastavovanie SEC nástroja

Do tohto modulu sme zaradili okrem správy používateľov aj nastavovanie SEC nástroja, ktorý je obmedzený na nastavovanie len administrátorom.

Analýza

Vzhľadom na to, že nastaviť nástroj SEC je netriviálna záležitosť, je potrebné odlíšiť jednotlivé skupiny používateľov a ich práva vykonávať jednotlivé aktivity v rámci celého systému.

Návrh

Používatelia sú rozdelení do dvoch kategórií - na bežných používateľov a administrátorov. Toto zabezpečí, že SEC nástroj bude nastavovaný iba skúseným používateľom s určitou úrovňou skúseností. Taktiež je potrebné zabezpečiť prihlasovanie do systému a zaraďovanie používateľov do jednotlivých skupín s odlíšiteľnými vlastnosťami a oprávneniami.

Implementácia

Implementácia prihlasovania je priamo možná v rámci rámca YII, ktorý poskytuje mnoho modulov na jednoduché pridanie. Prihlasovacia obrazovka je pre všetkých používateľov rovnaká, no po prihlásení sa zobrazia dve rôzne podoby menu – jedna pre bežného používateľa a druhá pre administrátora, ktorého úlohou je aj nastavovať pravidlá pre nástroj SEC a jeho odchytávanie a agregovanie udalostí. Administrátor je odvodeným používateľom od bežného používateľa a teda nestráca možnosti plnohodnotného využitia monitorovacích nástrojov našej aplikácie. Používatelia sa ukladajú do databázy so zahashovanými heslami a taktiež aj jednotlivé nastavenia SEC nástroja sú ukladané do databázy.

Testovanie

Pre testovanie prihlasovania sme si vygenerovali dummy používateľov, ktorým sme priradili heslá, a testovali sme rôzne scenáre prihlasovania a zmien v rámci skupín

používateľov a jednotlivých oprávnení. V čase písania tejto dokumentácie, netriviálne testovanie nastavovania nástrojov SEC nebolo zatiaľ implementované.

4.5 Dashboard a komponenty

Dashboard – v preklade „prístrojová doska“ je hlavným a úvodným oknom pre monitorovanie aktivity v rámci aplikácie SecMon. Je odrazovým mostíkom pre personalizáciu jednotlivých videní, ktoré môžu obsahovať rôzne grafy a tabuľky s informáciami o stave siete v reálnom čase.

Analýza

Po prihlásení musí mať používateľ nejakú základnú obrazovku, z ktorej sa vie odraziť smerom k nastaveniam toho, čo chce sledovať, kedy chce sledovať a akým spôsobom. Úvodné okno by malo byť prehľadné a malo by poskytovať možnosti modularity v rámci nižšej granularity.

Návrh

Dashboard by mal byť dostatočne jednoduchý a smerodajný na úvodné nastavenie a postupnú personalizáciu obsahu. Predpokladá sa, že každý používateľ zastáva inú úlohu a preto potrebuje iný pohľad na vec, rôzne videnia. To by malo byť jednoducho nastavovateľné v rámci dashboardu.

Implementácia

Po prihlásení sa používateľovi zobrazí plocha, ktorú môže vypĺňať pridávaním rôznych grafov, tabuliek, sledovačov a iných prídavných funkcionalít. Súčasťou dashboardu je aj menu pre používateľa, ktoré umožňuje meniť jednotlivé videnia, meniť časové obdobie prípadne ich inak parametrizovať. Taktiež je možné presmerovanie do používateľských nastavení alebo priame uloženie nastavení celkového dashboardu.

Testovanie

Testovacie dáta sa generujú z reálnych zariadení v sieti. Tieto dáta sa zatiaľ zbierajú iba na v developerskom prostredí a preto pre lokálne testovanie je potrebné si vyexportovať databázu, aby bolo možné testovať aj lokálne nie až na testovacom prostredí. Na začiatok uvažuje iba pár základných typov udalostí no na testovanie je to dostatočný počet. Tieto dáta sú rôznorodé, aby bolo možné testovať zobrazovanie a filtrovanie v rámci tabuľky v dashboarde.V rámci testovania sme odchytávali udalosti typu PortScan a Wrong SSH loginy. Tieto sme si zobrazovali v rôznych typoch komponentov s rôznymi pridanými filtrami.

4.6 Zobrazenie korelovaných udalostí a nekorelovaných logov

Zobrazenie udalostí je jednoduchý zoznam všetkých udalostí, ktoré sa dajú pomocou rôznych parametrov a filtrov prehľadne prehľadávať a analyzovať. Toto zobrazenie má slúžiť ako doplnkový nástroj na spätnú dôkladnejšiu analýzu udalostí.

Analýza

Výstupom zo SEC nástroja je entita – korelovaná udalosť, ktorá je práve tou informáciou, ktorá je zaujímavá v rámci monitorovania bezpečnosti. Preto je potrebné ju správnou a prehľadnou formou odkomunikovať používateľovi.

Návrh

Udalosti, ako sme sa dozvedeli v implementácii predošlých modulov, sú jednak zobrazované naživo a jednak ukladané do databázy. Preto je možné ich zobrazovať, filtrovať a inak spravovať.

Implementácia

Zobrazovanie korelovaných udalostí a nekorelovaných logov je možné pomocou tabuliek, ktoré zobrazujú potrebné informácie ako IP adresa zdroja, IP adresa cieľu,

porty komunikácie, názov eventu a iné. Používateľ môže aplikovať rôzne filtre ako napríklad typový filter alebo dátumový filter udalostí. Na základe týchto filtrov tak vie dohľadať historické údaje o dianí v sieti a tak analyzovať udalosti, ktoré vznikli.

Testovanie

Na testovanie používame rovnaké dáta ako pri dashboardoch a preto pre testovanie zobrazení udalostí platia rovnaké zásady.

5 Testovanie treťou stranou

V tejto kapitole sa nachádza opis produktu v stave, v akom bol daný tretej strane na testovanie počas 9teho šprintu. Treťou stranou bol samotný vedúci tímu ale aj jeho kolegovia s inštitúcie CSIRT.SK, ktorí nám poskytli skvelý feedback k nášmu produktu.

5.1 Zmrazená verzia produktu

Pri odovzdávaní nášho produktu na testovanie sme mali hotové bloky riešenia, ktoré zabezpečovali možnosť odchytávať určitý typ útokov a zobrazovať na frontende. Nasledujúci prehľad ukáže, o čom vieme, že by malo ísť a o čom vieme, že ešte stále nie je funkčné alebo odladené.

Čo by malo ísť

Tretej strane sme na testovanie dali túto funkcionalitu:

- Prihlásenie používateľov rôznych privilegovaných stupňov.
- Zobrazenie stĺpcového grafu
- Pridávanie filtrov
- Pridávanie a manipulácia s dashboardami
- Pridávanie komponentov do dashboardu
- Správa filtrov
- Správa komponentov a dashboardov
- Resizovateľnosť a draggovanie komponentov vo videní
- Refresh dát automaticky
- Zobrazovanie korelovaných eventov

- Zobrazovanie nekorelovaných logov
- Samotná korelácia pomocou nástroja SEC
- Zber a korelácia údajov z viacerých zariadení
- Zobrazenie detailu v tabuľkách eventov
- Responzívny web design

5.2 Čo nie je odladené

O tomto vieme, že nie je odladené a teda to netreba testovať.

- Zobrazenie grafu typu line.
- Jednoduchšie filtrovanie podľa dátumu
- Previazanie cez kliknutie medzi korelovanými a normalizovanými eventami

5.3 Feedback z testovania

Vedúci tímu nám priniesol niekoľko pripomienok na produkt v rámci jeho spätnej väzby a toto sú jeho pripomienky, ktoré je potrebné zapracovať:

- Graf nemá legendu.
- Používateľovi nie je možné za behu meniť privilégiá.
- Stránkovanie tabuľky je neohrabané
- Bolo by dobré doladiť pridávanie dátumových filtrov
- Variabilné zobrazovanie časového úseku v grafe
- Menu komponentov by sa malo zobraziť iba po kliknutí na tri bodky
- Správa SEC pravidiel - aktivácia/deaktivácia, vytvorenie migrácie
- Pri Datetime pickeroch, pri filtroch dať možnosť zvoliť aj minúty a hodiny

- Možnosť zvoliť pre tabuľku, ktoré stĺpce zobraziť (nech neukazuje stále všetky, aj tie nepotrebné)
- Uloženie pozícií komponentov v rámci dashboardu
- Bolo by dobré ujednotiť kategoriálny výstup zo SECu - aby napr.CEF vendor obsahovalo pri všektých pravidlách rovnakú kategóriu hodnôt
- Bolo by dobré nejak obmedziť samotné stránkovanie tabuliek, nech nejde donekonečna.
- Pri normalizovaní eventov určitého typu je vymenený SRC a DES port

5.4 TP CUP - Feedback

V rámci tohto testovacieho týždňa sme sa dostali aj na TP Cup, ktorý bol pre nás príležitosťou predviesť náš produkt aj širšej verejnosti. Porotcovia v rámci TP Cupu sa zastavovali pri našom stánku a my sme im vysvetľovali a predvádzali dané riešenie. V rámci TP Cupu sme dostali feedback, ktorý je zobrazený na obrázku číslo 5.1. Jedná sa o screenshot z mailu od pani profesorky Bielikovej.

****

Pozitívne aspekty:

- monitorovanie viacerých zariadení v sieti
- prakticky použiteľné v sieťovom prostredí
- Na kolene vyrobeny aplikacny SIEM z existujucich komponentov a vlastneho kodu.
- Pouzitelne v konkretnej oblasti, kde je ekonomicka rezerva na spravnu ceno tvorbu
- Prakticky zameraný projekt
- Vhodne zvolené GUI
- funkcny parser
- inovativne riesenie

Negatívne aspekty:

- Chybajuce push eventy
- Chybajuce UI na tvorbu sofistikovanych a statistickych pravidiel
- Chybajuce automaticke akcie systemunap. Pri jednoznacnych naruseniach pravidiel – suvisi s push eventami.
- Zrejme nie úplne premyslený ďalší osud projektu, najmä s ohľadom na potrebu (pred)definovania pravidiel
- zobrazenie dat iba na velmi zakladnej urovni
- prezentacia riesenia

Obr. 5.1: Feedback zo súťaže TP Cup

6 Technická dokumentácia

Táto kapitola obsahuje relevantnú technickú dokumentáciu jednotlivých častí systému.

6.1 SEC

Samotný nástroj SEC je open source a jeho dokumentáciu je možné nájsť na jeho oficálnej webovej stránke na odkaze :

```
https://simple-evcorr.github.io/
```

V tejto dokumentácii sa nachádza dokumentácia k našej implementácii. Dôležitý pojem, ktorý sa často vyskytuje v tejto časti dokumentácie:

Named pipe (tiež známa ako FIFO vzhľadom na jej správanie) - je rozšírenie tradičného konceptu pipe (rúry) v unixových systémoch. Je to jedna z metód používaných na medziprocesovú komunikáciu (IPC).

6.2 Normalizátor

Pre normalizovanie logov nám podobne, ako pre korelovanie, slúži nástroj SEC. Samozrejme pri vhodne upravených pravidlách. V rámci riešenia projektu sme pripravili dve vzorové sady pravidiel pre normalizáciu pomocou SECU. Obe sa nachádzajú v adresáry *rules/default/normalization*. Sú to súbory *iptables.rule* a *centos7_sshd.rule*. V súbore *iptables.rule* sa nachádzajú základné pravidlá pre normalizovanie logov z iptables. V súbore *centos7_sshd.rule* sa zasa nachádzajú základné pravidlá pre normalizáciu sshd logov.

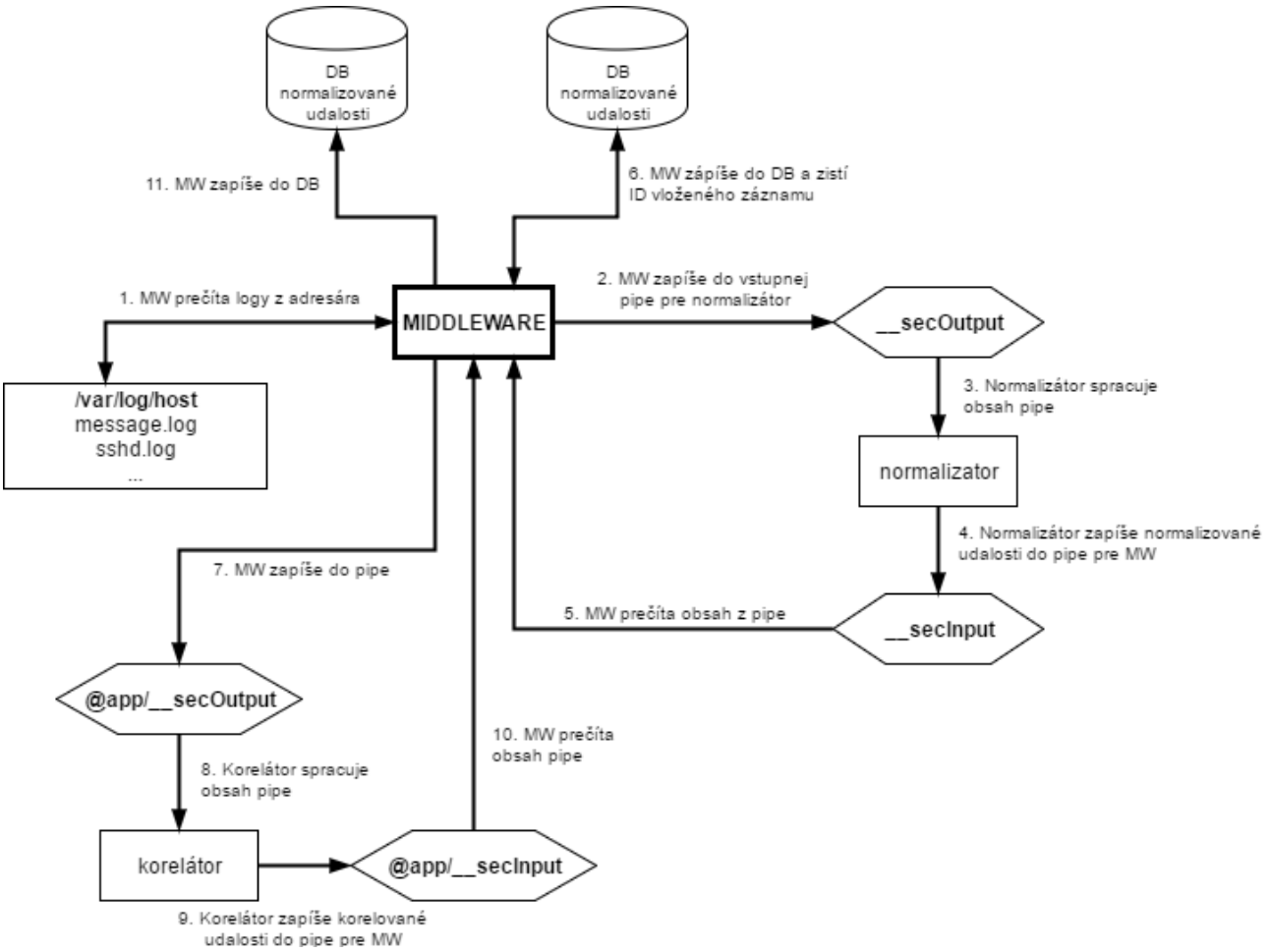
Ako to celé spustiť je podrobnejšie spomínané v sekcii Normalizátor

6.3 Middleware

Dôležitý pojem, ktorý sa často vyskytuje v tejto časti dokumentácie:

Named pipe (tiež známa ako FIFO vzhľadom na jej správanie) - je rozšírenie tradičného konceptu pipe (rúry) v unixových systémoch. Je to jedna z metód používaných na medziprocesovú komunikáciu (IPC).

Middleware vznikol ako prostriedok na prepojenie normalizátora, korelátora a databázy. V skratke funguje nasledovne. Neustále číta adresár, v ktorom sa ukladajú logy. Akonáhle nejaké logy pribudnú, posunie ich normalizárotu. Následne zoberie dáta, ktoré vyprodukoval normalizátor. Tie vloží do databázy normalizovaných udalostí a zároveň ich posunie korelátoru. Dáta vyprodukované korelátorom vezme a vloží do databázy korelovaných udalostí. Všetky tieto prepojenia jednotlivých častí realizujeme prostredníctvom *named pipes*. Celkovo middleware používa štyri *named pipes*.



Obr. 6.1: Činnosť middlewaru

Na obrázku 6.1 sú znázornené jednotlivé kroky, ako middleware beží aj s poradovým číslom. *Named pipes* môžme rozdeliť na tie pre normalizátor a tie pre korelátor. Majú rovnaké názvy ale ich umiestnenie je rozdielne. *Named pipes* pre normalizátor sa nachádzajú v adresári s logmi. Tie pre korelátor sa nachádzajú v hlavnom adresári aplikácie. *Named pipes* sa vytváraju pri prvom spustení middlewaru. Po načítaní logov z adresára posúva teda cez *pipe* logy normalizátoru. MW potom z *pipe* logy spracované normalizátorom prečíta a zapíše ich do DB pričom zisťuje ID záznamu. Po zistení ID záznamu prepošle spolu s IDčkom normalizované udalosti cez *pipe* korelátoru. Znovu sa odohrá podobný scenár ako pri normalizátore. Rozdiel je len v tom, že tu už sa po zápise do DB na nič nečaká.

Spustenie middlewaru

V tejto časti je stručný návod, ako middleware spustiť aj spolu s normalizátorom a korelátorom. Middleware je písaný v PHP. Konkrétne ako *command* v YII frameworku. Zdrojový kód k nemu sa nachádza v */commands/CorrelatorController.php*. Pred spustením samotného middlewaru si najskôr ukážeme ako spustiť normalizátor a korelátor.

Na spustenie normalizátora nám poslúži nasledujúci príkaz:

```
sudo /usr/local/bin/sec --conf=rules/default/normalization/* --input=/
    var/log/host/__secOutput --bufsize=1 --detach
```

Podobne aj korelátor tu však treba pre argument –input zvoliť *pipe* umiestnenú v hlavnom adresári aplikácie.

```
sudo /usr/local/bin/sec --conf=rules/default/correlation/* --input=
    __secOutput --bufsize=1 --detach
```

Middleware sa spúšťa prostredíctvom konzolovej aplikácie YII. Púšta sa nasledovným príkazom:

```
sudo ./yii correlator /var/log/secmon secmon
```

Pre spustenie na pozadí stačí dať na konci znak &. V prípade, že by ste chceli spustit middleware na pozadí a logovať aj prípadné chybové hlásenia, môžte použiť nasledujúci príkaz:

```
sudo ./yii correlator /var/log/secmon secmon 2> cesta/error.log &
```

6.4 Front-End

Táto podkapitola obsahuje dokumentováciu Front-Endovej časti našej aplikácie, ktorá je úspšne zabehnutá.

Rozloženie prvkov

Aplikácia sa spúšťa vo webovom prehliadači a rozloženie prvkov je možné vidieť na obrázku 6.2, nachádza sa tu:

- **Hlavné menu (č. 1)** - obsahuje informácie o prihlásenom používateľovi a menu jednotlivých stránok, ktoré je rozdelené do dvoch sekcií, pričom druhá je zobrazená iba administrátorovi.
- **Hlavička (č. 2)** - obsahuje názov stránky a na jej strede sú pripnuté tlačidlá obsluhujúce hlavnú funkcionalitu na stránke.
- **Hlavné okno (č. 3)** - zobrazuje sa tu obsah aktuálnej stránky.
- **Pätička (č. 4)** - obsahuje kontakt, odkaz na webové sídlo autorov aplikácie a copyright text.

Pre každú stránku je vytvorený .php súbor v priečinku:

```
\views\
```

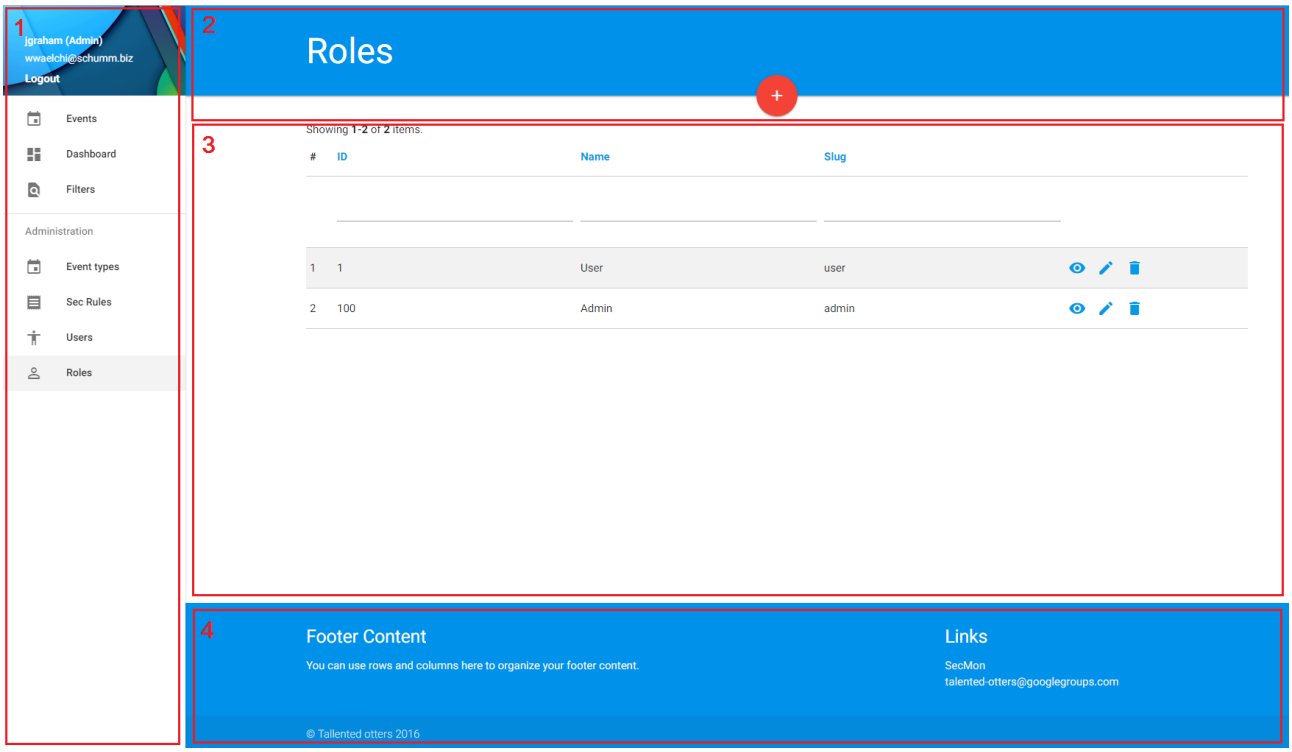
Tu sa nachádza aj hlavné view:

```
\views\layouts\main.php
```

V ňom je zadefinované zobrazenie menu, headera, footera a hlavného okna, v ktorom sa zobrazujú jednotlivé views podľa aktuálnej stránky.

Použité knižnice

V aplikácii je vyvíjaná v YII framework vrámci ktorého je možné importovať rôzne balíky. Hlavným balíkom je Materialize CSS. Je to moderný responzívny frontend framework založený na Material Designe. Balík je importnutý pomocou Composera a jeho verzia je zadefinovaná v súbore composer.json, ktorý sa nachádza v koreni projektu. Balík obsahuje javascript, css a widgety. Viac o balíku na oficiálnej githubovej stránke:



Obr. 6.2: Rozloženie prvkov v aplikácii

```
https://github.com/MacGyer/yii2-materializecss
```

Ďalej sú použité javascriptové knižnice pri vývoji frontendu aplikácie:

- **jQuery.js** - knižnica pre zlepšenie interakcie javascriptu a html.
- **Packery.js a Draggabilly.js** - knižnice použité na posúvateľné komponenty na dashboarde.
- **d3.js** - knižnica pre vytváranie grafov.

Tieto knižnice sú pridané ako súbory do priečinka:

```
\web\js
```

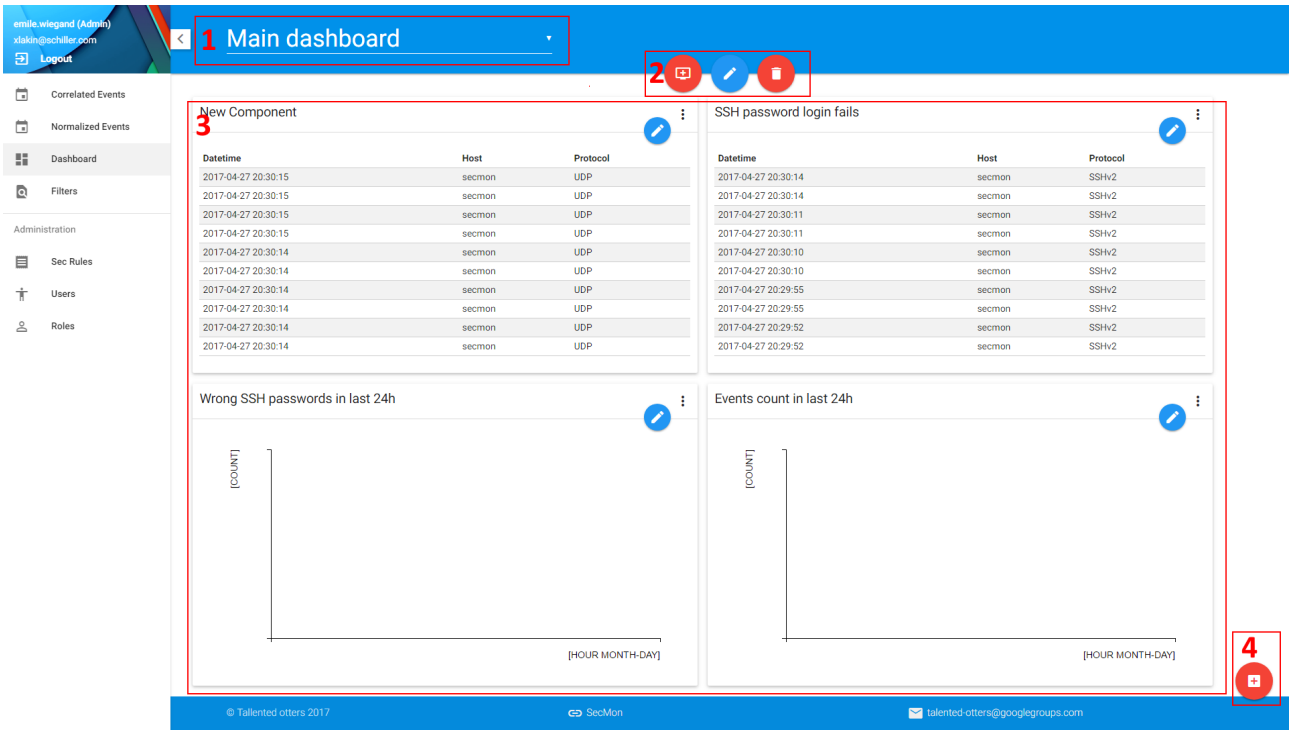
Importované do projektu sú definovaním v súbore:

```
\assets\AppAsset.php
```

Dashboard

Dashboard slúži na zobrazenie a monitorovanie vybraných udalostí. Dashboard sa dá vytvoriť viackrát, pričom každý má svoj názov a unikátny obsah, ktorý si nastaví používateľ. Práca s dashboardom prebieha nasledovne 6.3:

- **Výber dashboardu (č. 1)** - zobrazuje názov aktuálneho dashboardu a umožňuje výber iného.
- **Akcie dashboardu (č. 2)** - umožňujú používateľovi pridať nový dashboard, zmeniť názov alebo vymazať aktuálny dashboard. V prípade vymazania dashboardu budú vymazané aj všetky komponenty.
- **Obsah dashboardu (č. 3)** - tu sa nachádzajú komponenty na ktorých sú zobrazované vybrané udalosti.
- **Pridanie komponentu (č. 4)** - tlačidlo slúži na pridanie nového komponentu na aktuálny dashboard.



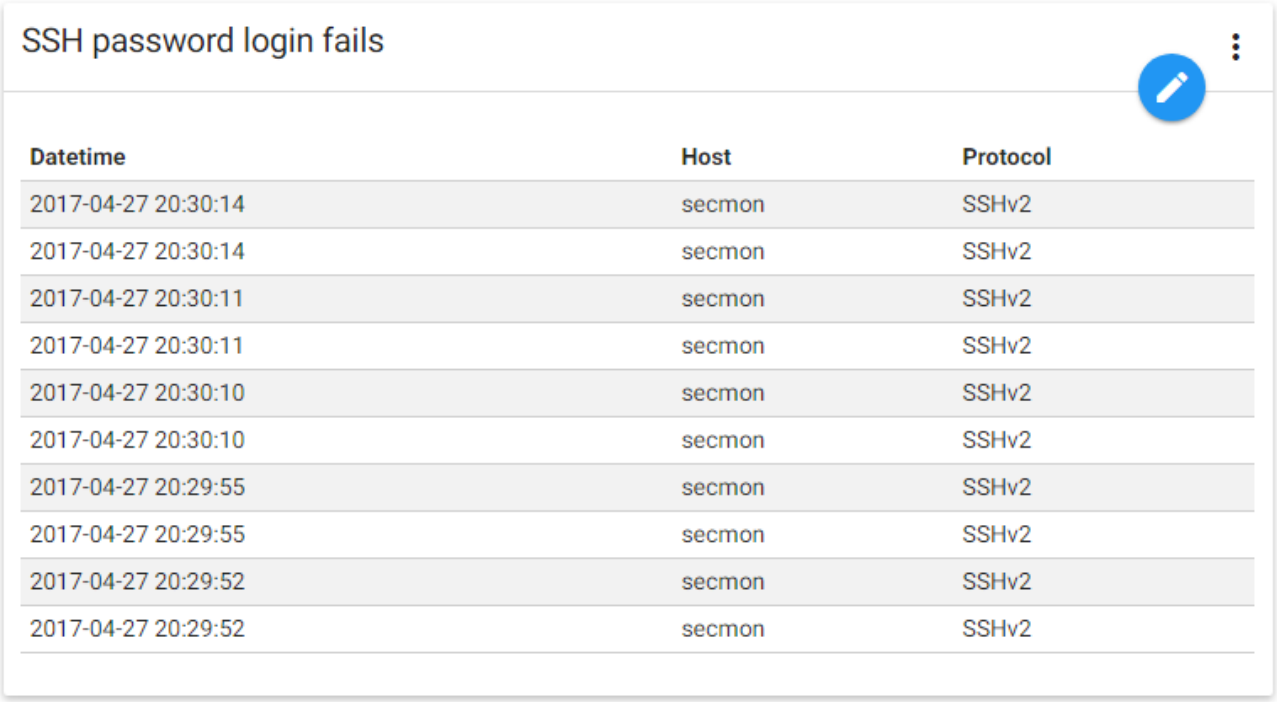
Obr. 6.3: Dashboard

Dashboardy sú ukladané v databáze *secmon* v tabuľke *views* a uchovávajú sa nasledovné informácie:

- **id** - Id dashboardu,
- **name** - názov dashboardu,
- **user_id** - id používateľa, ktorý vytvoril dashboard,
- **active** - určuje, či je dashboard aktuálne vybraný a zobrazený.

Komponenty a ich nastavenia

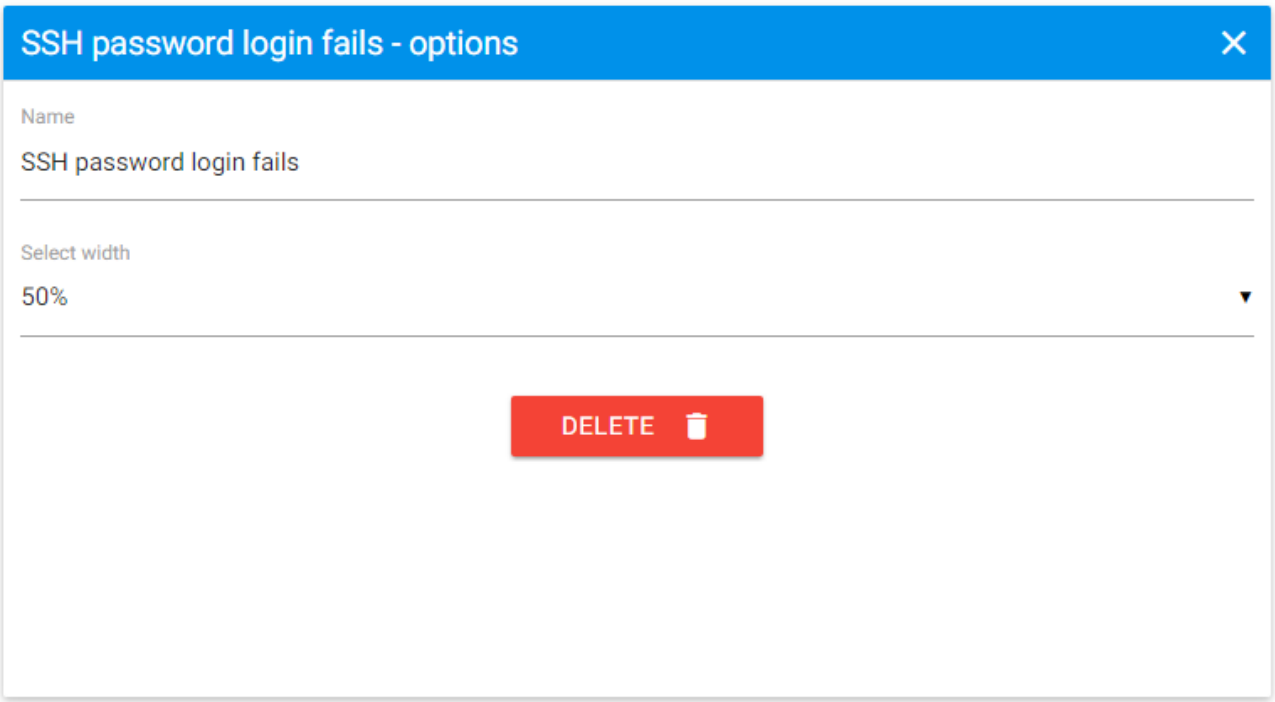
Komponenty slúžia na zobrazenie vybraných udalostí. Komponenty je možné voľne presúvať vrámci dashboardu a meniť ich názov, veľkosť alebo ich vymazať. Príklad komponentu s tabuľkou je na obrázku 6.4. Nastavenia komponentu sú priístupné po kliknutí na tri bodky v pravom hornom rohu, tie je vidieť na obrázku 6.5. Akákoľvek zmena komponentu je automaticky uložená a aplikovaná.



Obr. 6.4: Komponent

Komponenty sú ukladané v databáze secmon v tabuľke view_components a uchovávajú sa nasledovné informácie:

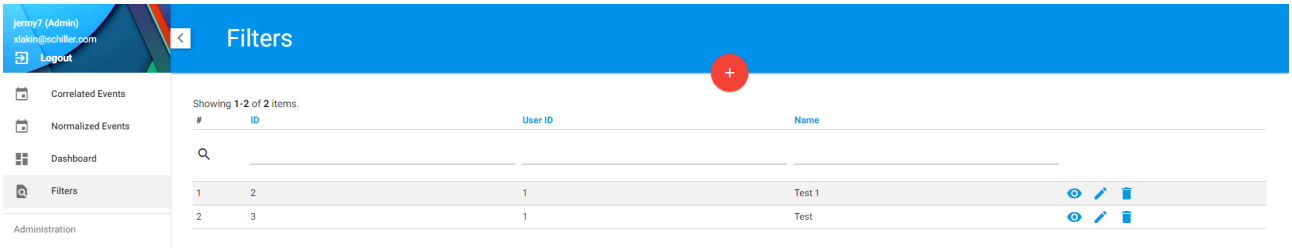
- **id** - Id komponentu,
- **view_id** - Id dashboardu, ktorému patrí komponent,
- **filter_id** - Id priradeného filtra,
- **config** - uchováva informácie v JSON formáte o názve, šírke a forme zobrazovaných údajov (napr.: tabuľka, graf) ,
- **order** - určuje poradie komponentu vrámci dashboardu.



Obr. 6.5: Nastavenia komponentu

Filtrovanie obsahu

Filtre na filtrovanie obsahu sa vytvárajú v hlavnej aplikácii v časti "Filters". Hlavná stánka danej časti je zoznam všetkých vytvorených filtrov pre daného používateľa. Každý používateľ má svoje vlastné pravidlá. Na obrázku 5.4 je zobrazená hlavná stránka.



Obr. 6.6: Hlavná stránka pre filtre

Na vytvorenie nového pravidla slúži obrazovka, ktorá je zobrazená na obrázku 6.6. Filter sa skladá z viacerých pravidiel. Tieto pravidlá sú aplikované za sebou s logickým operátorom, ktorý je zvolený pomocou hodnoty 5 na obrázku 6.7. Táto hodnota môže nadobúdať hodnoty ” AND ” alebo ” OR ”.

Každé pravidlo sa skladá z týchto štyroch parametrov:

1. **StĺpEc** - zvolený názov stĺpec, na ktorý sa má dáne pravidlo aplikovať,

2. **Typ** - typ pravidla,
3. **Operátor** - operátor, ktorý sa použije na porovnávanie,
4. **Hodnota** - hodnota, ktorá sa porovná s hodnotou daného stĺpca pomocou zvoleného, operátora a podľa typu pravidla.

Aktuálne sú podporované tieto typy pravidiel:

1. **Compare** - porovnanie hodnoty s hodnotou,
2. **Date** - dátumové porovnanie,
3. **Regular expression** - porovnanie pomocou regulárneho výrazu.

Jednotlivé typy pravidiel sa povolujú resp. zakazujú podľa zvoleného stĺpca. Táto definícia povolených typov pre jednotlivé stĺpce sa nachádza v triede modelu pre každý typ udalostí, konkrétne v statickej metóde *columns*. Toto je zadefinované v triede *BaseEvent*, ktorá zabezpečuje logiku práce s touto definíciou. Tieto modely sa nachádzjú v priečinku:

```
\models\
```

Každý typ pravidla ma vlastnú triedu, ktorá je zdedená z triedy *BaseFilterRule*. Táto trieda obsahuje logiku filtrovania. Každý typ pravidla musí byť vo formáte:

```
%typ_pravidla%FilterRule.php
```

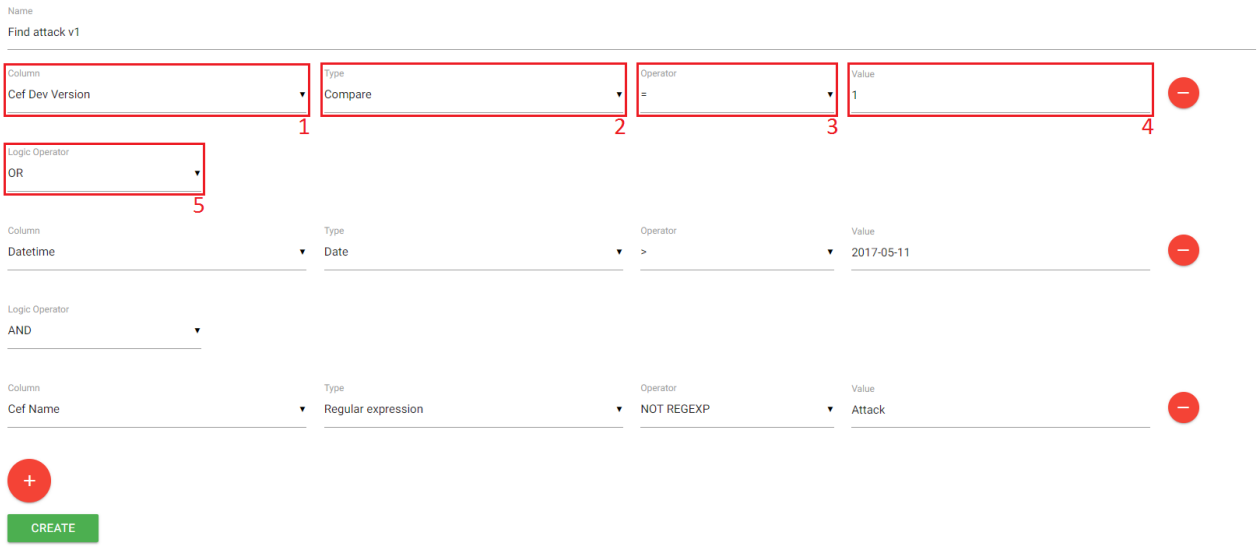
aby dané filtrovacie pravidlo fungovalo. Toto je spôsobené tým, že pravidlá sa vyhľadávajú na základe názvu triedy. Tieto pravidlá sa nachádzajú v priečinku:

```
\components\filter\
```

Filtre rozdeľujeme na *statické a dynamické*. Statické sú také, ktoré sa aplikujú až po zísakní dát z databázy. Naopak dynamické sú také, ktoré sa pridávajú do požiadavky na databázu a teda aplikujú sa už pri dotaze na dáta.

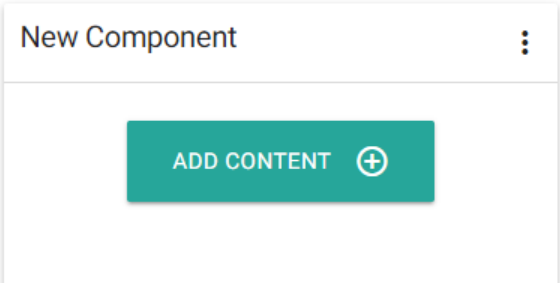
6.5 Pridávanie obsahu na komponent

Po pridaní komponentu do dashboardu, je potrebné nastaviť aký obsah má zobrazovať. Toto docielime tak, že klineme na tlačidlo ÄDD CONTENT", ktoré je znázor-

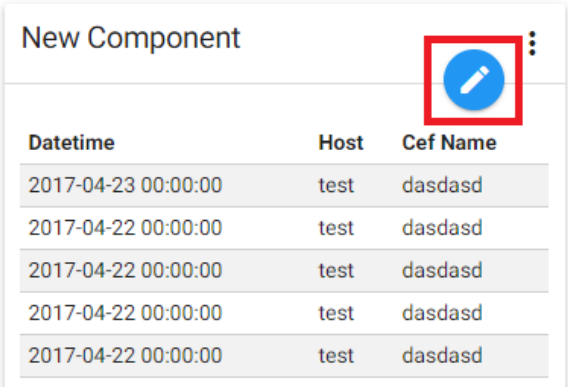


Obr. 6.7: Formulár na pridanie nového filtra

nené na obrázku 6.8. Keď už má komponent priradený obsah, je možné ho upraviť pomocou tlačidla ËDIT", ktoré je zvýraznené na obrázku 6.9.



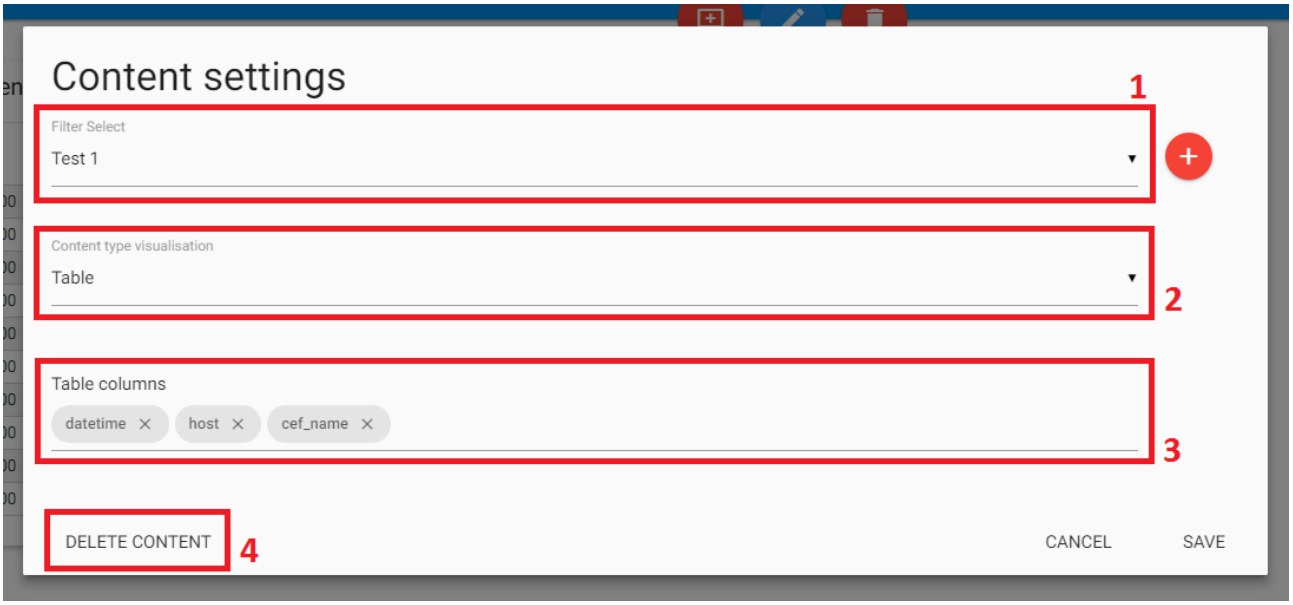
Obr. 6.8: Komponent bez obsahu



Obr. 6.9: Komponent s obsahom

Nastavenie obsahu alebo úprava sa robí v dialógu, ktorý je zobrazený na obrázku 6.10. V tomto dialógu sa vyberá, ktorý filter má byť aplikovaný na dáta (číslo 1

na obrázku 6.10). Ďalej sa tu vyberá typ zobrazovaného obsahu (číslo 2 na obrázku 6.10). Podľa toho aký je zvolený typ zobrazovaného obsahu, sa zobrazí posledné pole s doplnkovou informáciou pre dané zobrazenie obsahu (číslo 3 na obrázku 6.10). V prípade tabuľku sú to názvy stĺpcov a v prípade stĺpcového grafu to je časový interval, za ktorý sa majú zobraziť dáta. Momentálne sú podporované len tieto 2 typy zobrazovaného obsahu. Odobrať obsah z komponentu je možné pomocou tlačidla "DELETE CONTENT", ktoré je označené číslom 4 na obrázku 6.10.



Obr. 6.10: Dialóg na nastavenie obsahu komponentu

Pri tabuľkovom zobrazení sa automaticky dopĺňajú názvy stĺpcov po začatí písania. Pri stĺpcovom grafe sa používa definovanie časového intervalu definovanom v PHP. Príklad definovania časového obdobia 2 dní:

```
P2D
```

Definícia začína veľkým písmenom 'P' a nesleduje definícia časového obdobia vo formáte:

```
[pocet][identifikator typu]
```

Identifikátor typu určuje či ide o dni, mesiace, hodine a pod. Detailnejšiu definíciu nájdete na adrese:

```
http://php.net/manual/en/dateinterval.format.php
```

6.6 Back-End

V tejto podkapitole je uvedená dokumentácia k backendu našej aplikácie.

Generovaná dokumentácia

Generovanú dokumentáciu sme získali pomocou nástroja *PHPDoc*, ktorý nám z našej hierarchie súborov v projekte vygeneroval k tomu HTML dokumentáciu podľa štandardov tohto nástroja. Táto dokumentácia je dostupná na elektronickom médiu a na webovej stránke nášho tímu.

Dokumentácia k frameworku

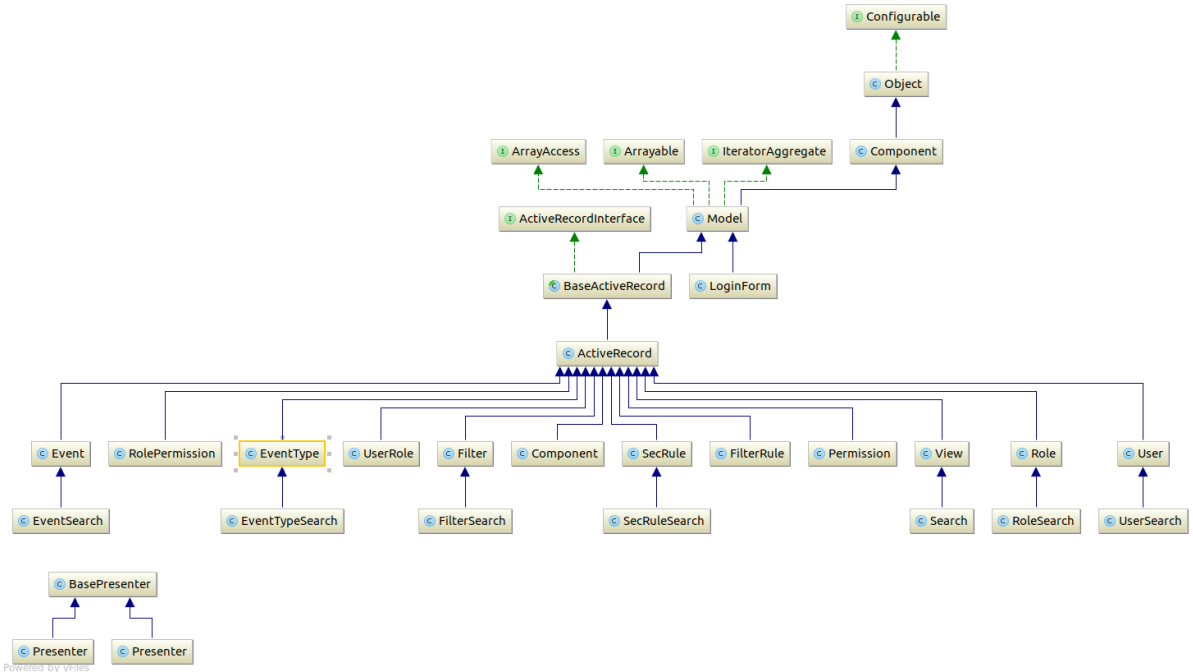
Pri vývoji tejto aplikácie sa riadime oficiálnou dokumentáciou *YII Framework version 2*. Možné je ju dohľadať na webovej stránke:

```
http://www.yiiframework.com/doc-2.0/
```

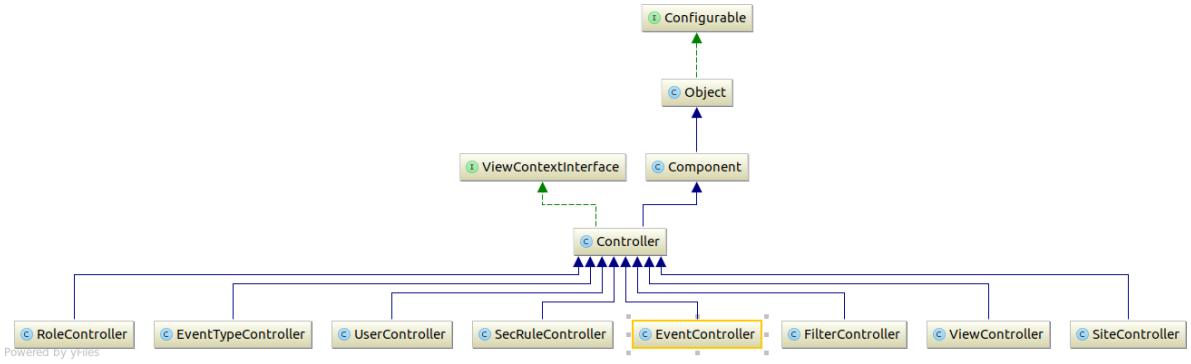
Diagramy

Počas vývoja sa vyvíja aj vnútorná štruktúra projektu a databázy. Aktuálny stav backendu aplikácie je možné vidieť na nasledujúcich obrázkoch:

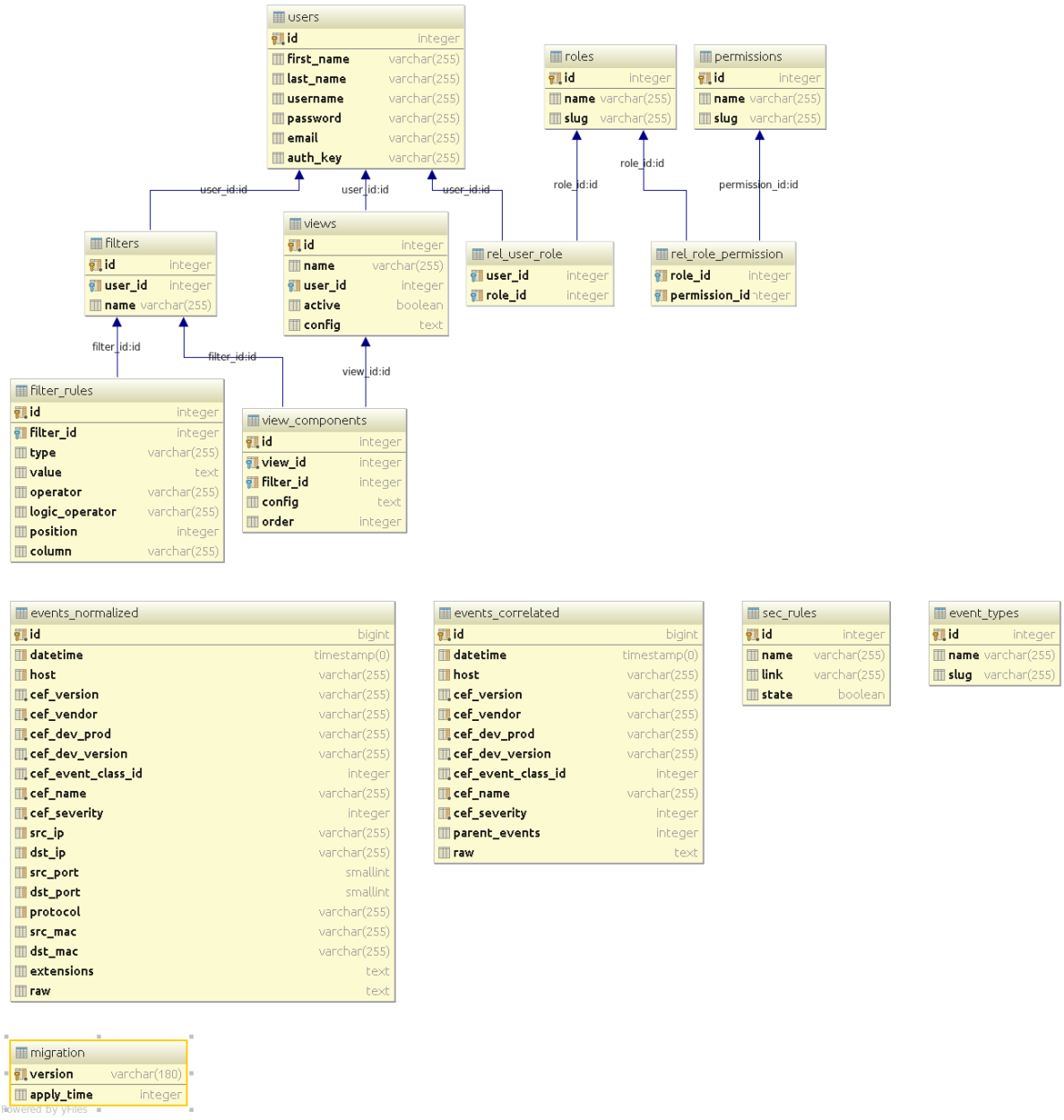
- **Diagram tried modelov** - je možné vidieť na obrázku 6.11. Diagram tried modelov aj s metódami sa nachádza v prílohe B-1.
- **Diagram tried kontrolérov** - je možné vidieť na obrázku 6.12. Diagram tried kontrolérov aj s metódami sa nachádza v prílohe B-2.
- **Dátový model** - aktuálny stav databázy s prepojeniami tabuliek pomocou foreign keys je možné vidieť na obrázku 6.13.



Obr. 6.11: Diagram tried modelov



Obr. 6.12: Diagram tried kontrolérov



Obr. 6.13: Aktuálny dátový model

7 | Príručky

V tejto kapitole sa nachádzajú relevantné príručky k nášmu produktu.

7.1 Inštalačná príručka Apache serveru pre produkčný server

Postup na inštalovanie APACHE2 webserveru:

```
sudo yum clean all
sudo yum -y update
sudo yum -y install httpd
```

Potom treba pridať výnimku na firewall:

```
sudo firewall-cmd --permanent --add-port=80/tcp
sudo firewall-cmd --permanent --add-port=443/tcp
sudo firewall-cmd -reload
```

Teraz nainštalujem Composer a pomocou Composeru aj framework Yii2:

```
curl -sS https://getcomposer.org/installer | php
mv composer.phar /usr/local/bin/composer
composer global require "fxp/composer-asset-plugin:^1.3.1"
composer create-project --prefer-dist yiisoft/yii2-app-basic basic
```

Následne naštartujeme Apache a nastavíme jeho zapínanie pri štarte[1]:

```
sudo systemctl start httpd
sudo systemctl enable httpd
```

Keď je všetko hotové, tak si do document rootu naklonujem git repozitár:

```
git clone git@github.com:mp205/secmon_backend.git /var/www/secmon/ -b
    master
```

V config sa nachádzajú konfiguračné vzorové súbory kde sa nastavuje prístup do databázy:

```
config/db.php
```

[1] Bližšie informácie na http://www.yiiframework.com/doc-2.0/guide-start-installation.html

```
<?php
return [
    'class' => 'yii\db\Connection',
    'dsn' => 'pgsql:host=localhost;dbname=meno\_databazy',
    'username' => 'meno\_pouzivatela',
    'password' => 'super\_bezpecne\_heslo',
    'charset' => 'utf8',
];
```

Treba samozrejme nainštalovať PostgreSQL a vytvoriť databázu a používateľa:

```
yum install postgresql, php-pgsql
```

Po pripojení do databázy:

```
CREATEUSER -U postgres -d -e -E -l -P -r -s meno_usera;
CREATE DATABASE meno_databazy OWNER meno\_usera;
```

config/config_secdb.ini sa nastavuje podľa technickej dokumentácie SECu - kapitola 5.1. Potom zostáva iba zmigrovať databázu a posťahovať balíky:

```
./yii migrate
composer update
```

Je podstatné nastaviť prístupy cez .htaccess a v /etc/httpd/conf/httpd.conf:

```
do /etc/httpd/conf/httpd.conf doplnit:
<Directory />
    AllowOverride none
    Require all denied
</Directory>
<Directory "/var/www">
    AllowOverride All
    Options Indexes FollowSymLinks
    Options All -Indexes
    Require all denied
</Directory>
<Directory "/var/www/secmon>
    Options Indexes FollowSymLinks
    Options All -Indexes
    AllowOverride all
    Require all granted
</Directory>
```

Do .htaccess doplniť a potom podľa ľubovôle dodať aj aliasy pre krajšiu adresu:

```
<Directory "/var/www/secmon/web">
    Options Indexes FollowSymLinks MultiViews
 AllowOverride all
 <ifDefine APACHE24>
    Require local
 </ifDefine>
 <ifDefine !APACHE24>
    Order Deny,Allow
\#        Deny from all
\#        Allow from localhost ::1 127.0.0.1
        Allow from all
 </ifDefine>
\# prevent httpd from serving dotfiles (.htaccess, .svn, .git, etc.)
RedirectMatch 403 /\..*\$
\# if a directory or a file exists, use it directly
RewriteCond \%{REQUEST_FILENAME} !-f
RewriteCond \%{REQUEST_FILENAME} !-d
\# otherwise forward it to index.php
RewriteRule . index.php
</Directory>
```

Teraz "nainštalujeme"SEC (perl je už zvyčajne súčasťou OS pri inštalácii[2]), tak že stiahneme najaktuálnejšiu verziu z http://sourceforge.net/project/showfiles.php?group_id=42089 a následne ju nakopírujeme aj s manuálovou stránkou na správne miesto, napríklad:

```
cp sec /usr/local/bin
cp sec.man /usr/local/share/man/man1/sec.1
```

A pridá sa cesta k PERLu do SEC súboru. Ak je cesta

```
/usr/local/bin/perl
cp sec /usr/local/bin
cp sec.man /usr/local/man/man1/sec.1
```

Tak potom prvý riadok vyzerá nasledovne:

```
\#!/usr/local/bin/perl -w
```

Spustenie SECu, aby bežal spustený neustále:

[2]ak náhodou nieje perl nainčtalovaný, tak nainštalujem: yum install perl

```
sec -conf=/cela/cesta/ku/konfigu.conf -input /cela/cesta/k/logom.log --
    bufsize=1 -detach
```

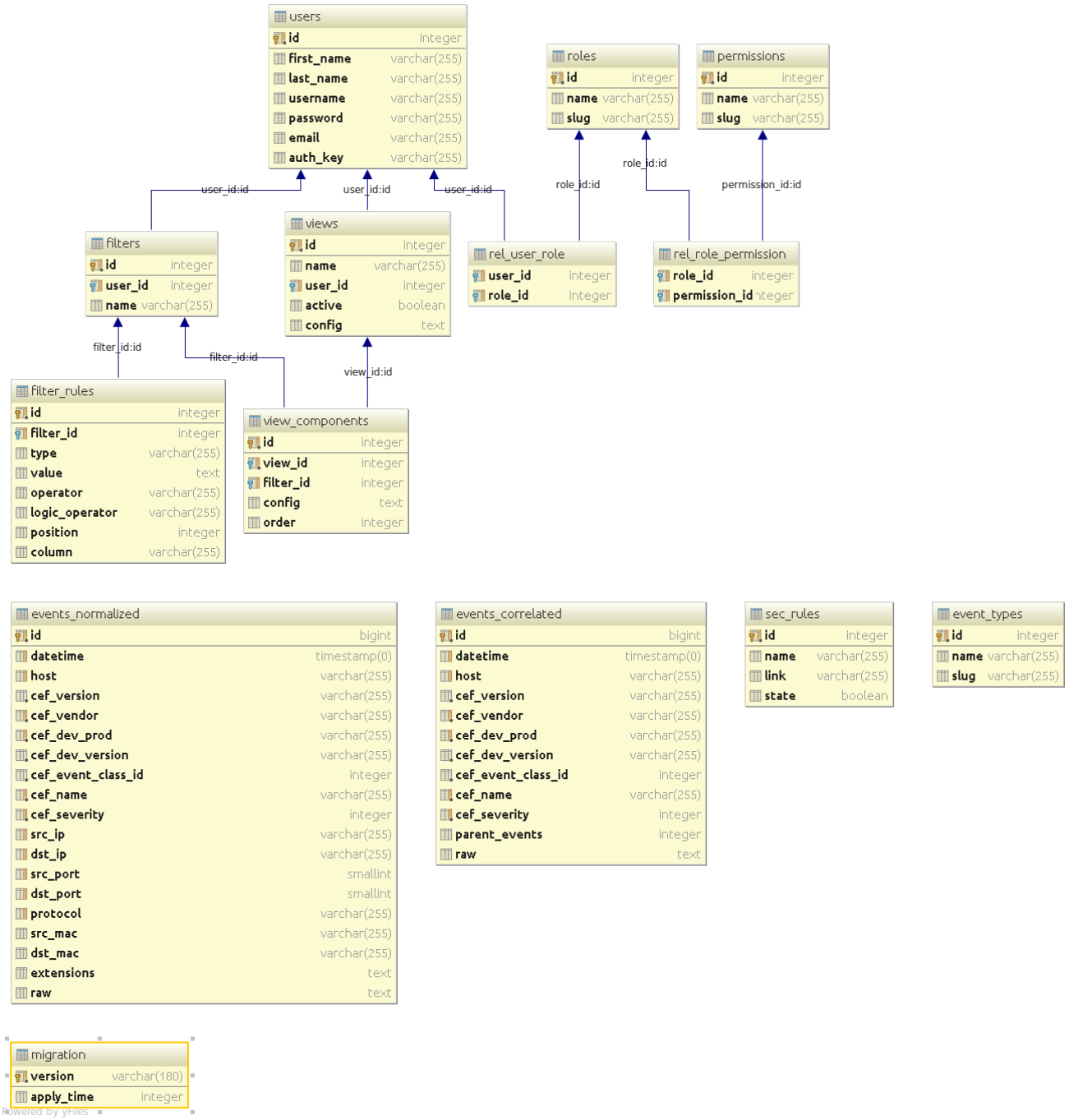
Nakoniec sa štandardne podľa potreby nainštaluje a nakonfiguruje *rsyslog*.

Prílohy

Zoznam príloh

Príloha A

Dátový model



Obr. 1: Dátový model

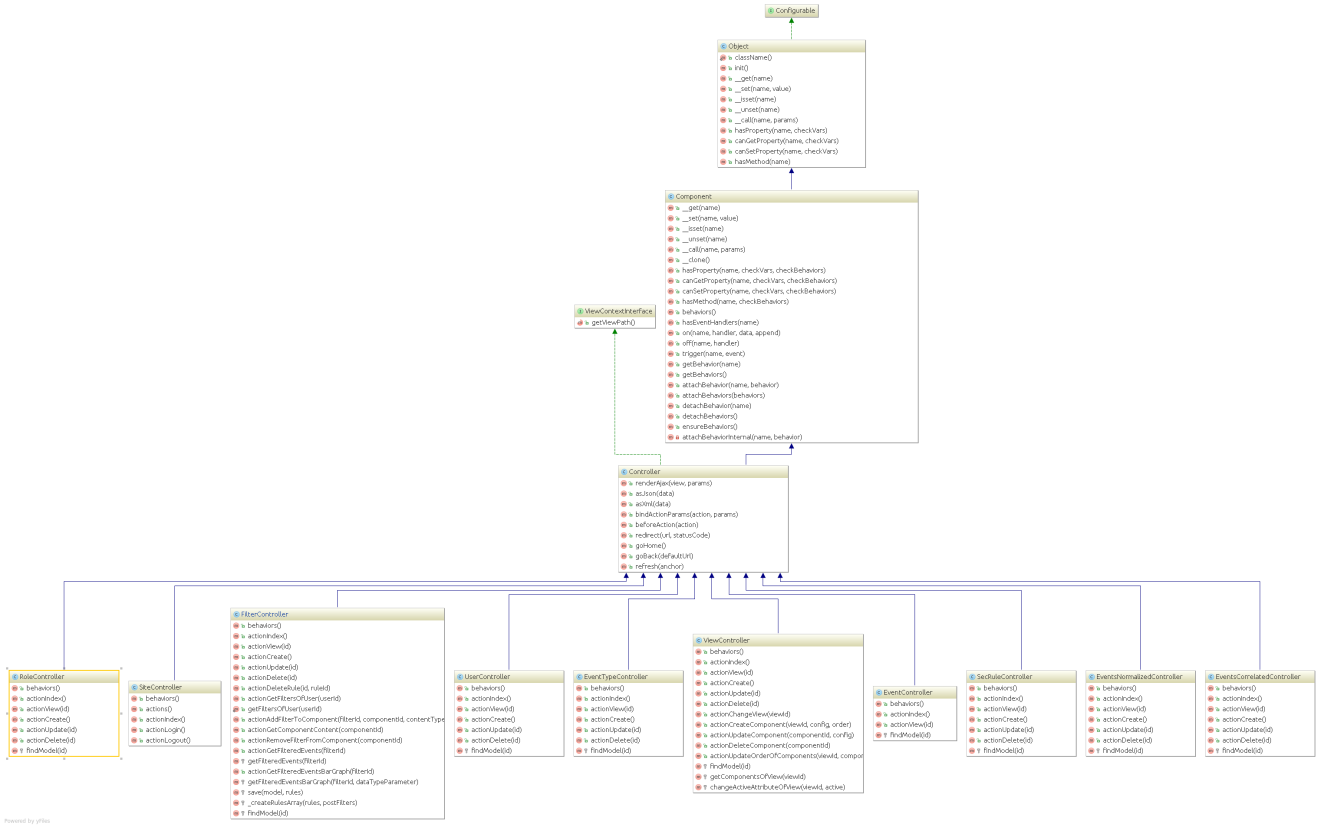
Príloha B-1

Diagram tried modelov

Obr. 2: Diagram tried modelov s metódami

Príloha B-2

Diagram tried kontrolérov



Obr. 3: Diagram tried kontrolérov s metódami