Kapitola 4

Návrh nového monitoringu zátěže serverů

Tato kapitola popisuje problémy současného monitoringu zátěže serverů v KNOTIS a z toho plynoucí důvody, proč došlo k jeho nové implementaci. Dále je zde popsán návrh nového monitoringu zátěže, který řeší problémy toho stávajícího.

4.1 Problémy současného monitoringu zátěže serverů

Současný monitoring zátěže serverů využívá pro zaznamenávání jednotlivých monitorovaných hodnot programy *Nmon* a *Top*. *Nmon* je komplexní monitorovací nástroj, který zaznamenává kromě všech potřebných monitorovaných hodnot také hodnoty, které pro monitoring nejsou potřeba. Každý den tak *Nmon* na jednom serveru nasbírá přibližně 1,6 MB zbytečných dat, která pro monitoring nemají význam. Data, která jsou monitoringem skutečně využita, zabírají přibližně 0,2 MB.

Dalším problémem je příliš hrubé vzorkování sběru dat. Sběr dat, jak již bylo zmíněno, probíhá vždy pouze jedenkrát za hodinu. Může se tak snadno stát, že případná větší zátěž, která bude trvat např. 50 minut a bude probíhat právě mezi dvěma okamžiky sběru dat, nebude vůbec monitoringem zaznamenána a nikdo se o ní nedozví.

Soubor, který je jedenkrát za den odesílán na server *knot.fit.vutbr.cz*, obsahuje pro každý časový okamžik sběru dat všechny monitorované hodnoty. Někdy se stává, že po sobě následující hodnoty jsou stejné, a dochází tak zbytečně k několikanásobnému odesílání stejné informace. Tato situace je zachycena na výpisu 4.1.

```
MEM_RAM_FREE[MB],320,320,300,275,401,401,401,401,401,401,401
MEM_RAM_TOTAL[MB],3944,3944,3944,3944,3944,3944,3944,3944,3944,3944,3944
MEM_SWAP_FREE [MB],4093,4093,4093,4093,4080,4080,4080,4080,4080,4080,4080
MEM_SWAP_TOTAL[MB],4093,4093,4093,4093,4093,4093,4093,4093,4093,4093,4093
```

Výpis 4.1: Ukázka výpisu odesílaných dat se stejnou hodnotou v některých časových okamžicích sběru dat.

Dalším problémem je návrh databáze, do které jsou data ukládána, viz příloha A.1. Všechny tabulky, do kterých se nasbíraná data ukládají (`mon_zateze_cpu`, `mon_zateze_cpu_all`, `mon_zateze_disk`, `mon_zateze_disk_all`, `mon_zateze_mem`, `mon_zateze_top_num_processes` a `mon_zateze_top`), obsahují sloupec `server`, který by v těchto tabulkách nemusel být. Hodnota ve sloupci `server` je identifikátorem serveru, na kterém byla daná

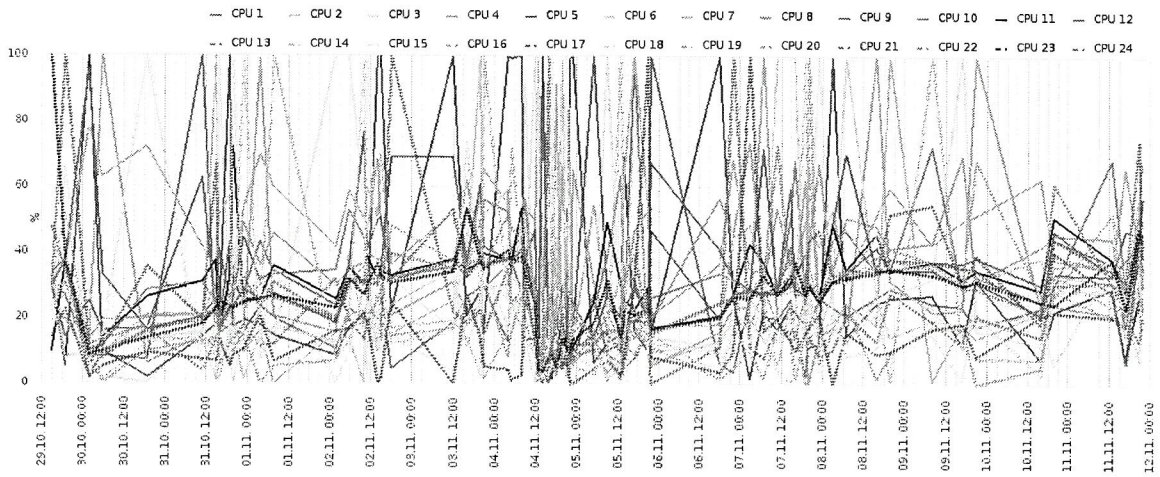
data nasbírána. Tato data jsou vždy spojována s nějakým konkrétním časovým okamžikem z tabulky `mon_zateze_time`. Stačilo by tedy, kdyby sloupec `server` byl přesunut do tabulky `mon_zateze_time` a ze všech ostatních tabulek, byl odstraněn.

V databázi také vzniká stejný problém, jako při odesílání dat. Některé po sobě nasbírané hodnoty mohou obsahovat tutéž informaci a do databáze se pak ukládají zbytečně. Ukázka uložení po sobě jdoucích stejných hodnot je na obrázku 4.1.

id	time	server	active	cached	ram_free	ram_free_percent	ram_total	swap_free	swap_free_percent	swap_total
1	1	1	1401.9	864.8	1761.4	44.7	3944.2	4093.0	100.0	4093.0
2	2	1	1449.3	870.9	1710.6	43.4	3944.2	4093.0	100.0	4093.0
3	3	1	1452.8	874.2	1703.6	43.2	3944.2	4093.0	100.0	4093.0
4	4	1	1474.6	875.1	1685.2	42.7	3944.2	4093.0	100.0	4093.0
5	5	1	1514.3	1114.2	1196.7	30.3	3944.2	4093.0	100.0	4093.0
6	6	1	1521.4	1137.8	1154.6	29.3	3944.2	4093.0	100.0	4093.0
7	7	1	1520.1	1140.0	1157.0	29.3	3944.2	4093.0	100.0	4093.0
8	8	1	1522.9	1800.3	138.6	3.5	3944.2	4093.0	100.0	4093.0
9	9	1	1582.2	1334.6	1198.5	30.4	3944.2	4093.0	100.0	4093.0

Obrázek 4.1: Ukázka uložení stejných hodnot z po sobě jdoucích časových okamžiků sběru dat.

Uživatelské rozhraní monitoringu, jak již bylo zmíněno, se skládá pouze z jedné stránky systému KNOTIS s názvem *Detail serveru*. Vedoucí tak má možnost zjistit zatížení vždy pouze jednoho konkrétního serveru bez dalších souvislostí. Grafy, které jsou na stránce zobrazeny, se generují na serveru a na stranu klienta se posílají jako obrázek ve formátu *png*. Z tohoto důvodu nelze s grafy provádět žádnou interakci. Nelze si zvolit libovolné časové období od kdy do kdy se grafy mají překreslit. Nelze rušit či zapínat zobrazení jednotlivých datových řad. Dále nelze měnit měřítko jednotlivých os grafu či zobrazovat detailní informace po najetí myši na graf nějaké datové řady. Grafy jsou také někdy dosti nepřehledné, viz obrázek 4.2.



Obrázek 4.2: Ukázka nepřehlednosti grafů stávajícího monitoringu zátěže.

Dalším problémem jak z pohledu zbytečného množství dat v databázi, tak i z pohledu uživatelského rozhraní, je ukládání informací o vytěžujících procesech do statistiky TOP. Pokud na některém serveru není během sběru dat generována žádná zátěž, tak se často stává, že všechny tři procesy, které se do statistiky TOP uloží, využívají procesor daného serveru na 0 %. Často se také stává, že se v této statistice objeví pouze jeden proces s využitím procesoru více než 0 %, a to právě program *Top*, který tuto statistiku vytvořil. Takovéto procesy jsou z pohledu monitorování zátěže nepodstatné, a vůbec by nemusely být ukládány a následně zobrazovány v uživatelském rozhraní. Jak vypadá zobrazení statistiky TOP, pokud server není zatížen, je zobrazeno na obrázku 4.3.

TOP

Datum	Pořadí	Příkaz	CPU	MEM	PID	User
11.11.2017 23:55	1	top	6 %	0.0 %	30440	root
	2	init	0 %	0.0 %	1	root
	3	kthreadd	0 %	0.0 %	2	root
11.11.2017 22:55	1	top	6 %	0.0 %	29942	root
	2	init	0 %	0.0 %	1	root
	3	kthreadd	0 %	0.0 %	2	root
11.11.2017 21:55	1	init	0 %	0.0 %	1	root
	2	kthreadd	0 %	0.0 %	2	root
	3	ksoftirqd/0	0 %	0.0 %	3	root

Obrázek 4.3: Zobrazení části statistiky TOP na nezatíženém serveru.

4.2 Nové monitorované hodnoty

Všechny hodnoty, které jsou zaznamenávány v současném monitoringu zátěže serverů, jsou nadále zaznamenávány i v nové implementaci tohoto monitoringu. Byly však přidány některé nové hodnoty, o kterých vedoucí v současném monitoringu neměl přehled.

V současném monitoringu zátěže nemá vedoucí žádné informace o síťovém provozu, který na daném serveru probíhá. Aby vedoucí měl přehled o tom, jak intenzivní síťová komunikace je na daném serveru generována, byl sběr dat rozšířen o nové monitorované hodnoty, které v současném monitoringu nebyly vůbec zaznamenávány. Jedná se o hodnoty, které popisují vytížení jednotlivých síťových rozhraní a počet soketů, které jsou na serveru vytvořeny.

4.2.1 Vytížení síťových rozhraní

V rámci monitorování vytížení síťových rozhraní jsou zaznamenávány následující hodnoty: *INTERFACE receive* a *INTERFACE transmit*. Tyto hodnoty jsou zaznamenávány pro jednotlivá síťová rozhraní, která se na daném serveru nacházejí.

- **INTERFACE receive** – Udává množství doručených dat na dané síťové rozhraní za daný časový interval.
- **INTERFACE transmit** – Udává množství odeslaných dat z daného síťového rozhraní za daný časový interval.

4.2.2 Počet soketů

Soket tvoří API[1] pro procesy komunikující jak po síti, tak i lokálně v rámci operačního systému. Vytváří tzv. koncový bod komunikace. Implementačně se jedná o abstraktní datovou strukturu, která obsahuje nezbytné údaje pro komunikaci a jednoznačnou identifikaci komunikujících uzlů. Detailní popis soketů je uveden v knize [4].

Poslední hodnotou, která je v rámci nového monitoringu zátěže zaznamenávána, je hodnota *OPEN SOCKETS* vyjadřující počet soketů. Popis této hodnoty je převzat z manuálové stránky *ss* [8].

- **OPEN SOCKETS** – Udává celkový počet všech vytvořených soketů na daném serveru v daný časový okamžik. Do tohoto součtu se počítají sokety všech typů, které jsou aktuálně v rámci operačního systému vytvořeny. Jsou to jak sokety typu TCP, UDP či RAW, které slouží pro síťovou komunikaci, tak i sokety, které slouží ke komunikaci meziprocesové v rámci operačního systému (tzv. *Unix domain sockets*[2]).

4.3 Návrh sběru a odesílání dat

Sběr dat je základní částí celého monitoringu zátěže. Celý sběr dat se skládá ze sběru monitorovaných hodnot současného monitoringu popsaných v kapitole 2.3 a ze sběru nových monitorovaných hodnot popsaných v kapitole 4.2. Není důvod proč by některé z hodnot současného monitoringu měly být v novém monitoringu vynechány. Všechny tyto hodnoty jsou pro monitoring důležité a mají potřebnou vypovídací hodnotu. Celkově se tedy v novém monitoringu sbírají data potřebná k zjištění vytížení procesoru, paměti, swapu, disků a síťových rozhraní daného serveru. Dále pak data potřebná ke zjištění, kterými uživateli a procesy je daný server nejvíce vytěžován. Případně kolik procesů je na daném serveru spuštěno či kolik je na něm vytvořeno soketů.

Sběr dat probíhá na každém serveru nezávisle. Může tak být spuštěn současně na libovolném počtu serverů, u kterých je potřeba, aby na nich monitoring probíhal.

Sběr dat se provádí v pravidelných časových intervalech. Pro každý časový okamžik, kdy dochází ke sběru dat, se vytvoří soubor, do kterého se uloží všechny sbírané hodnoty aktuální pro tento časový okamžik. Interval mezi jednotlivými okamžiky sběru dat není 1 hodina, jako u současného monitoringu, ale je zkrácen na interval 5 minut. Tento časový interval lze v určitém rozsahu nastavit na libovolnou hodnotu, viz kapitola 5.1. Data se tak sbírají častěji a tím se výrazně omezuje možnost, že by nějaká případná větší zátěž nebyla monitoringem zaznamenána. Každý tento soubor nese ve svém názvu informaci o přesném času, kdy byla data, která jsou v něm uložena, nasbírána. Soubor je poté uložen na daném serveru do dočasného adresáře, ve kterém se všechny soubory jednotlivých okamžiků sběru dat průběžně shromažďují.

Způsob, jakým jsou data sbírána, se liší oproti současnému monitoringu. Již se nevyužívá monitorovacího nástroje *Nmon*, ale využívá se více jednodušších programů, viz kapitola 5.1. Touto změnou bylo docíleno toho, že v dočasných souborech jsou uložena pouze data, která jsou pro monitoring skutečně potřeba, a ne i data, která pro monitoring nemají žádný význam a jejich uložení je tedy zbytečné.

Po provedení určitého počtu cyklů sběru dat, který lze také libovolně nastavit, viz kapitola 5.1, se ze všech dočasných souborů vytvoří jeden výsledný soubor, který je ná-

[1] API – Application programming interface

[2] Linux Programmer's Manual UNIX(7) http://man7.org/linux/man-pages/man7/unix.7.html

sledně odeslán na server *knot.fit.vutbr.cz*, kde je dále zpracován. Počet cyklů je v novém monitoringu nastaven tak, aby se výsledný soubor pro odeslání vytvářel vždy jedenkrát za den. Čili je-li interval mezi okamžiky sběru dat 5 minut, pak počet cyklů (vytvořených dočasných souborů) sběru dat než dojde k odeslání, je 288. Po provedení všech 288 cyklů sběru dat je obsah všech vytvořených souborů transformován do jednoho výsledného souboru ve formátu CSV, viz kapitola 3.2. Následně je k tomuto souboru přidána hlavička obsahující potřebné údaje k autentizaci serveru a poté, jak již bylo zmíněno, je odeslán na server *knot.fit.vutbr.cz*. Každý řádek odesílaného souboru obsahuje název jedné monitorované hodnoty a její nasbíraná data pro všechny časové okamžiky sběru dat. Přesný formát odesílaných dat je popsán v kapitole 5.2.

Během transformace dočasných souborů do jednoho výsledného je provedena důležitá optimalizace množství odesílaných dat. Do výsledného souboru se ve skutečnosti neukládají všechny hodnoty, ale pouze ty, které jsou odlišné vůči hodnotě předcházejícího časového okamžiku sběru dat. Odesílaná data současným monitoringem, která jsou zobrazena ve výpisu 4.1, jsou při použití nového monitoringu odesílána tak, jak je uvedeno ve výpisu 4.2.

```
MEM_RAM_FREE[MB],320,,300,275,401,,,,,,
MEM_RAM_TOTAL[MB],3944,,,,,,,,,,
MEM_SWAP_FREE [MB],4093,,,,4080,,,,,,
MEM_SWAP_TOTAL[MB],4093,,,,,,,,,,
```

Výpis 4.2: Ukázka odesílaných dat z výpisu 4.1, při použití nového monitoringu zátěže serverů.

Na těchto výpisech lze dobře vidět, že pokud se některé monitorované hodnoty v čase příliš nemění, tak je množství odesílaných dat při použití nového monitoringu výrazně menší, než v případě použití monitoringu současného.

Tato optimalizace se dá také libovolně nastavit pro každou monitorovanou hodnotu zvlášť, viz kapitola 5.1. Lze nastavit, jak moc se musí dvě po sobě nasbírané hodnoty lišit, aby se považovaly za odlišné a do výsledného souboru se uložily obě. Optimalizaci lze také vypnout, aby se data odesílala opravdu všechna jako při použití současného monitoringu.

Podobný princip je použit i u statistky TOP. Existuje příslušný limit, kterým se dá nastavit, jak moc daný proces musí zatěžovat procesor, aby se tento záznam uložil do výsledného souboru, viz kapitola 5.1. V základu je nastaven tak, aby se zaznamenávaly všechny procesy, které procesor vytěžují na více než 0 %. Dalším limitem se pak dá nastavit maximální počet procesů, který se má ve výsledku uložit. Ten je v základu nastaven stejně jako v současném monitoringu, a to na maximálně tři procesy.

4.4 Návrh schématu databáze

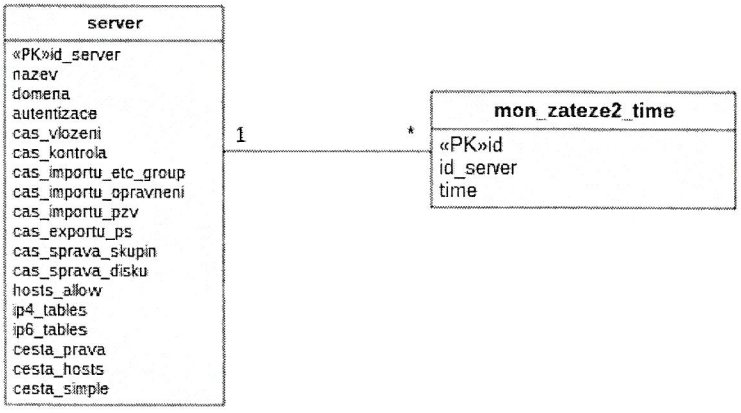
Aby nedocházelo k problémům s databází popsaným v kapitole 4.1, byla nová databáze navržena jinak, než ta původní. Schéma databáze současného monitoringu je znázorněno v příloze A.1 a schéma databáze nového monitoringu je znázorněno v příloze B.1.

V současném monitoringu je pro každou skupinu spolu souvisejících monitorovaných hodnot vytvořena jedna tabulka. Např. pro hodnoty, které se týkají vytížení procesoru, existuje tabulka `mon_zateze_cpu`, která obsahuje sloupce s hodnotami `idle`, `sys`, `user` a `wait`. Tyto hodnoty se do databáze ukládají vždy všechny pro daný časový okamžik sběru dat. Je tedy logické mít tyto hodnoty takto pohromadě v jedné tabulce s identifikátorem příslušného jádra procesoru.

U nového monitoringu je situace odlišná. Pro daný časový okamžik sběru dat se nemusejí ukládat všechny monitorované hodnoty, ale pouze některé, konkrétně ty, které jsou odlišné oproti hodnotě předchozího časového okamžiku sběru dat. Pokud by tedy sloupce `idle`, `sys`, `user` a `wait` zůstaly všechny v jedné tabulce, tak by v případech, kdy se mají uložit pouze některé z těchto hodnot, docházelo k tomu, že by se do ostatních sloupců uložila hodnota `null`. Možná úspora množství ukládaných dat do databáze by tak ve výsledku vůbec nefungovala. Pouze by se místo stejné hodnoty předchozího okamžiku sběru dat uložila do databáze hodnota `null`. Z tohoto důvodu má v databázi nového monitoringu každá monitorovaná hodnota vytvořenou vlastní tabulku, do které se ukládá pouze tato hodnota. Pokud se tedy např. mají vložit nové hodnoty `idle` a `sys`, tak se vloží pouze do příslušných tabulek a do tabulek s hodnotami `user` a `wait` se již nic nevloží. Dojde tak k požadovanému ušetření množství ukládaných dat do databáze. Ve výsledku se tedy do databáze ukládá nová hodnota pouze tehdy, je-li tato hodnota rozdílná oproti hodnotě v předcházejícím časovém okamžiku sběru dat. Pro některé časové okamžiky sběru dat tedy databáze neobsahuje některé monitorované hodnoty. Při následném načítání těchto hodnot z databáze je potřeba provést jejich doplnění hodnotami z předcházejících časových okamžiků, viz kapitola 5.3.

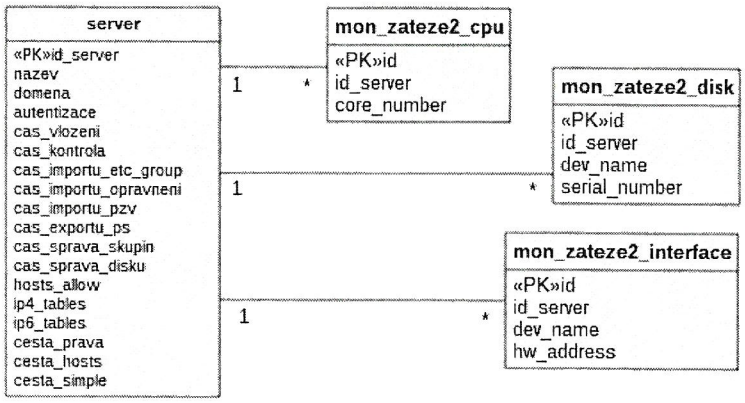
4.4.1 Schéma databáze

Sloupec `server`, který je v tabulkách současného monitoringu, které obsahují hodnoty nasbíraných dat, byl přesunut do tabulky `mon_zateze2_time`, přejmenován na `id_server` a z ostatních tabulek odstraněn. Tabulka `mon_zateze2_time` tak obsahuje sloupec `id_server` (identifikátor serveru, na kterém byl sběr dat proveden) a sloupec `time` (přesný čas sběru dat). Tabulka `server` je již v databázi vytvořena. Obsahuje záznamy o všech serverech, které jsou v rámci skupiny KNOT využívány. Pro monitoring zátěže jsou důležité pouze sloupce `nazev`, který obsahuje jméno daného serveru, `domena`, který obsahuje doménové jméno daného serveru a `autentizace`, který obsahuje autentizační řetězec daného serveru. Všechny tyto hodnoty jsou využívány při přijímání nových dat s následným ukládáním do databáze, viz kapitola 5.2. Tabulky `mon_zateze2_time` a `server` jsou znázorněny na obrázku 4.4.



Obrázek 4.4: Databázové tabulky `server` a `mon_zateze2_time`.

Pro všechna monitorovaná zařízení na serveru jsou vytvořeny patřičné tabulky. Zařízeními se rozumí jádra procesoru, disky a síťová rozhraní. Každý server může mít jiný počet těchto zařízení, proto jsou pro ně vytvořeny jednotlivé tabulky s vazbou na daný server. Tabulka `mon_zateze2_cpu` obsahuje sloupec `id_server` (identifikátor serveru, na kterém se dané jádro procesoru nachází) a sloupec `core_number`, který obsahuje čísla jednotlivých jader. Tabulka `mon_zateze2_disk` obsahuje shodně sloupec `id_server` a dále obsahuje sloupce `dev_name` a `serial_number`. Sloupec `dev_name` obsahuje názvy jednotlivých disků či jejich oddílů tak, jak jsou uvedeny v systémovém adresáři */dev*. Sloupec `serial_number` pak obsahuje sériová čísla těchto disků. Další tabulkou je `mon_zateze2_interface`, která obsahuje informace o jednotlivých síťových rozhraních na daném serveru. Tato tabulka obsahuje shodně jako u předchozích tabulek sloupec `id_server` a dále pak sloupce `dev_name` a `hw_address`. Sloupec `dev_name` obsahuje názvy jednotlivých síťových rozhraní tak, jak jsou uvedeny v systémovém adresáři */proc/net* v souboru *dev*. Sloupec `hw_address` obsahuje fyzické adresy jednotlivých síťových rozhraní. Tyto tabulky s vazbami na tabulku `server` jsou znázorněny na obrázku 4.5.

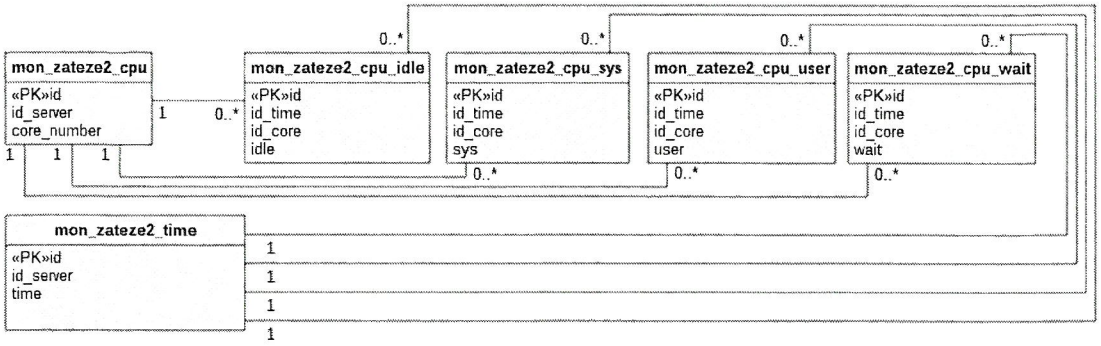


Obrázek 4.5: Databázové tabulky `server`, `mon_zateze2_cpu`, `mon_zateze2_disk` a `mon_zateze2_interface`.

Dalšími tabulkami jsou tabulky, které obsahují jednotlivé monitorované hodnoty týkající se vytížení jader procesoru. Jak již bylo popsáno v úvodu této kapitoly, pro jednotlivé hodnoty `idle`, `sys`, `user` a `wait` jsou vytvořeny zvláštní tabulky, konkrétně tabulky `mon_zateze2_cpu_idle`, `mon_zateze2_cpu_sys`, `mon_zateze2_cpu_user` a `mon_zateze2_cpu_wait`. Každá z těchto tabulek obsahuje sloupec `id_time` a sloupec `id_core`. Sloupec `id_time` obsahuje identifikátor konkrétního časového okamžiku z tabulky `mon_zateze2_time`, při kterém došlo ke sběru dat. Sloupec `id_core` pak obsahuje identifikátor příslušného jádra procesoru z tabulky `mon_zateze2_cpu`, na kterém tato data byla nasbírána. Poslední sloupec pak podle názvu tabulky obsahuje danou monitorovanou hodnotu (`idle`, `sys`, `user` nebo `wait`). Tyto tabulky s vazbami na tabulku `mon_zateze2_cpu` a tabulku `mon_zateze2_time` jsou znázorněny na obrázku 4.6.

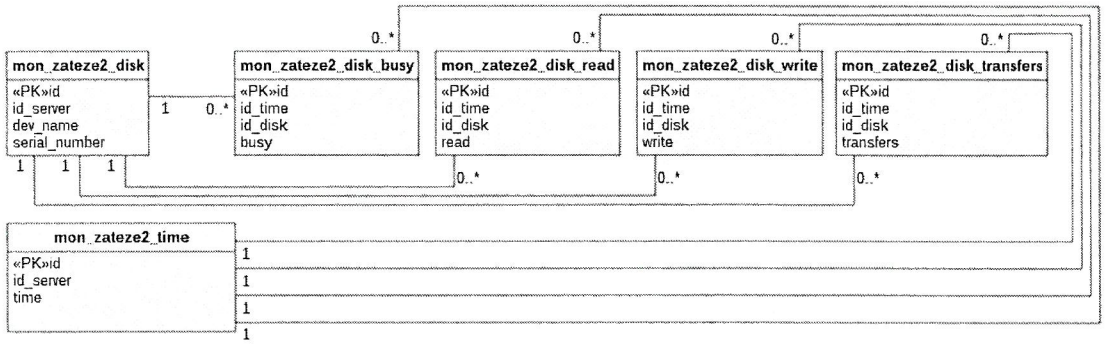
Stejným způsobem jako tabulky, které obsahují jednotlivé monitorované hodnoty týkající se vytížení jader procesoru jsou vytvořeny i tabulky, které obsahují monitorované hodnoty týkající se vytížení disků a síťových rozhraní.

Pro ukládání hodnot týkajících se vytížení disků jsou vytvořeny tabulky `mon_zateze2_disk_busy`, `mon_zateze2_disk_read`, `mon_zateze2_disk_write` a `mon_zateze2_disk_-`



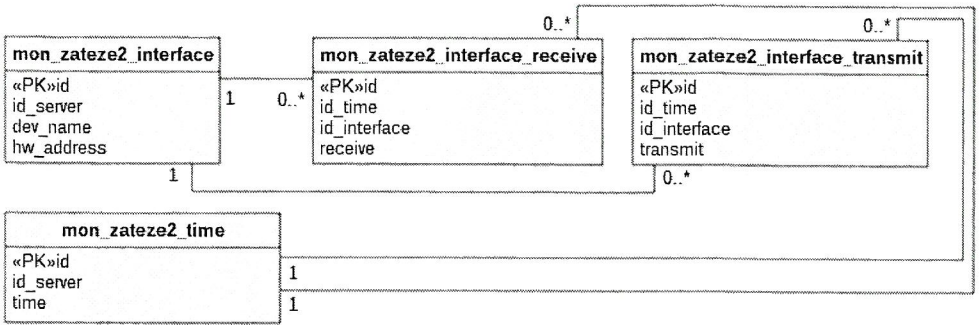
Obrázek 4.6: Databázové tabulky `mon_zateze2_time`, `mon_zateze2_cpu`, `mon_zateze2_cpu_idle`, `mon_zateze2_cpu_sys`, `mon_zateze2_cpu_user` a `mon_zateze2_cpu_wait`.

`transfers`. Stejně jako u tabulek týkajících se vytížení jader procesoru, obsahují i tyto tabulky sloupec `id_time`, který obsahuje identifikátor konkrétního času z tabulky `mon_zateze2_time`, při kterém došlo ke sběru dat. Dalším sloupcem je sloupec `id_disk`. Tento sloupec obsahuje identifikátor příslušného disku z tabulky `mon_zateze2_disk`, na kterém příslušná data byla nasbírána. Poslední sloupec pak opět podle názvu tabulky obsahuje danou monitorovanou hodnotu (`busy`, `read`, `write` nebo `transfers`). Tyto tabulky s vazbami na tabulky `mon_zateze2_disk` a `mon_zateze2_time` jsou znázorněny na obrázku 4.7.



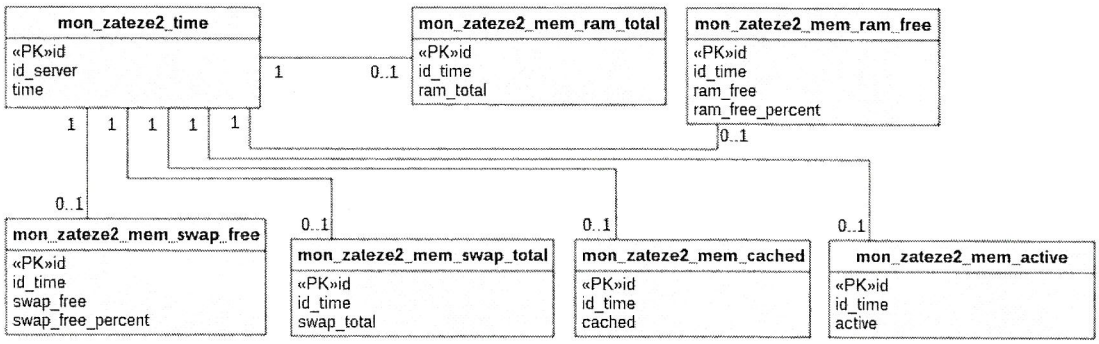
Obrázek 4.7: Databázové tabulky `mon_zateze2_time`, `mon_zateze2_disk`, `mon_zateze2_disk_busy`, `mon_zateze2_disk_read`, `mon_zateze2_disk_write` a `mon_zateze2_disk_transfers`.

Pro ukládání hodnot týkajících se vytížení síťových rozhraní jsou podobně jako v předchozích případech vytvořeny tabulky `mon_zateze2_interface_receive` a `mon_zateze2_interface_transmit`. Obdobně také tyto tabulky obsahují sloupec `id_time`, který obsahuje identifikátor času z tabulky `mon_zateze2_time` a sloupec `id_interface`. Sloupec `id_interface` obsahuje jednoznačný identifikátor příslušného síťového rozhraní z tabulky `mon_zateze2_interface`, na kterém příslušná data byla nasbírána. Poslední sloupec pak podle názvu tabulky obsahuje danou monitorovanou hodnotu (`receive` nebo `transmit`). Tyto tabulky s vazbami na tabulku `mon_zateze2_interface` a tabulku `mon_zateze2_time` jsou znázorněny na obrázku 4.8.



Obrázek 4.8: Databázové tabulky `mon_zateze2_time`, `mon_zateze2_interface`, `mon_zateze2_interface_receive` a `mon_zateze2_interface_transmit`.

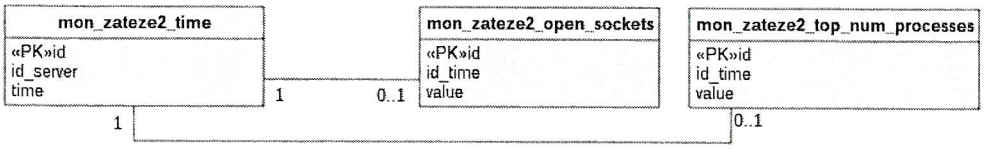
Další monitorované hodnoty se již neváží ke konkrétnímu zařízení na serveru, ale váží se přímo k nějakému serveru. Nasbírané hodnoty týkající se vytížení fyzické paměti jsou ukládány do tabulek s názvy `mon_zateze2_mem_ram_total`, `mon_zateze2_mem_ram_free`, `mon_zateze2_mem_active` a `mon_zateze2_mem_cached`. Hodnoty týkající se vytížení odkládacího prostoru SWAP pak do tabulek `mon_zateze2_mem_swap_total` a `mon_zateze2_mem_swap_free`. Všechny tyto tabulky, stejně jako v předešlých případech, obsahují sloupec `id_time`, který obsahuje identifikátor konkrétního času z tabulky `mon_zateze2_time`, při kterém došlo ke sběru dat. Díky tomuto identifikátoru lze i jednoznačně určit, na kterém serveru k tomuto sběru dat došlo, a není tedy v těchto tabulkách potřeba dalšího sloupce s identifikátorem serveru, jak tomu bylo v případě současného monitoringu. Další sloupec pak podle názvu tabulky obsahuje danou monitorovanou hodnotu (`ram_total`, `ram_free`, `active`, `cached`, `swap_total` a `swap_free`). Tyto tabulky s vazbami na tabulku `mon_zateze2_time` jsou znázorněny na obrázku 4.9.



Obrázek 4.9: Databázové tabulky `mon_zateze2_time`, `mon_zateze2_mem_ram_total`, `mon_zateze2_mem_ram_free`, `mon_zateze2_mem_active`, `mon_zateze2_mem_cached`, `mon_zateze2_mem_swap_total` a `mon_zateze2_mem_swap_free`.

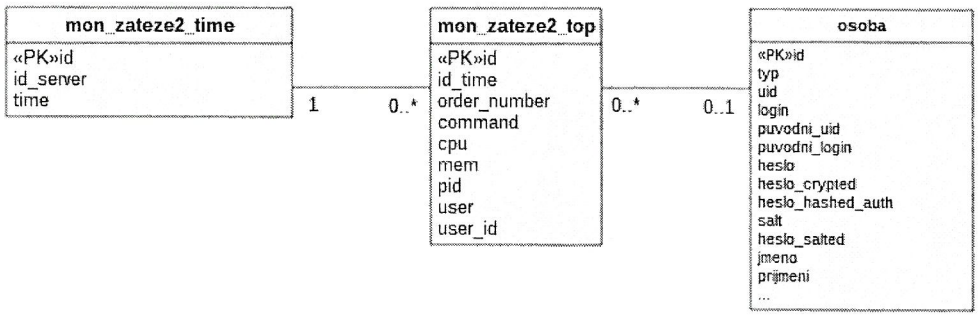
Stejně jako tabulky týkající se vytížení fyzické paměti jsou vytvořeny i tabulky pro počet vytvořených soketů a počet spuštěných procesů na daném serveru. Jsou to tabulky `mon_zateze2_open_sockets` a `mon_zateze2_top_num_processes`. Stejně tak tyto tabulky obsahují sloupec `id_time`. Dalším sloupcem, který tyto tabulky obsahují je sloupec `value`.

Tento sloupec obsahuje danou nasbíranou hodnotu (`open_sockets` nebo `top_num_processes`). Tyto tabulky s vazbami na tabulku `mon_zateze2_time` jsou znázorněny na obrázku 4.10.



Obrázek 4.10: Databázové tabulky `mon_zateze2_time`, `mon_zateze2_open_sockets` a `mon_zateze2_top_num_processes`.

Poslední tabulkou potřebnou pro ukládání dat monitoringu zátěže je tabulka obsahující statistiky TOP. Tato tabulka má název `mon_zateze2_top` a obsahuje sloupce `id_time`, `order_number`, `command`, `cpu`, `mem`, `pid`, `user` a `user_id`. Sloupec `id_time` má stejný význam jako v předchozích tabulkách. Sloupec `user_id` je identifikátor konkrétního uživatele z tabulky `osoba`, který je v KNOTIS registrován. Ostatní sloupce obsahují monitorované hodnoty popsané v kapitole 2.3.4. Tabulka `osoba` je již v databázi vytvořena a obsahuje záznamy o všech uživatelích, kteří jsou v rámci KNOTIS registrováni. Pro monitoring zátěže je důležitý pouze sloupec `id`, který obsahuje identifikátor tohoto uživatele a sloupec `login`, který obsahuje uživatelské jméno tohoto uživatele v rámci operačního systému. Většina sloupců tabulky `osoba` není v obrázku 4.11 zobrazena, protože nemají pro monitoring význam. Celá tabulka `osoba` je znázorněna v přílohách A.1 a B.1. Důvodem proč tabulka `mon_zateze2_top` obsahuje současně sloupec `user` se jménem uživatele ze statistiky TOP a sloupec `user_id` obsahující identifikátor uživatele z tabulky `osoba` je to, že ve statistice TOP se mohou objevit i uživatelé, kteří nejsou v KNOTIS registrováni (např. uživatel *root*). Pro tohoto uživatele tedy neexistuje vazba na tabulku `osoba` a jeho jméno je tedy uchováno ve sloupci `user` v tabulce `mon_zateze2_top`. Tabulka `mon_zateze2_top` s vazbami na tabulky `mon_zateze2_time` a `osoba` je znázorněna na obrázku 4.11.



Obrázek 4.11: Databázové tabulky `mon_zateze2_time`, `mon_zateze2_top` a `osoba`.

V celkovém výsledku návrh databáze nového monitoringu zátěže obsahuje mnohem více tabulek než je potřeba u monitoringu současného. Je však díky tomuto návrhu umožněno to, aby se do databáze ukládaly jen ty hodnoty, které jsou odlišné oproti hodnotám před-

cházejícího časového okamžiku sběru dat. Množství dat, které se ve výsledku do databáze uloží je menší, než množství dat, které se uloží do databáze monitoringu současného.

4.5 Návrh kontroly a ukládání dat

Kontrola dat je důležitou součástí monitoringu zátěže a je nezbytné aby byla provedena před tím než dojde k uložení dat do databáze. Zajistí, aby data byla uložena ve správném formátu a aby poté mohla být správně interpretována.

Nasbíraná data na jakémkoliv serveru jsou jedenkrát za den uložena ve formátu CSV, viz kapitola 3.2, do výsledného souboru, který je následně odeslán na server *knot.fit.vutbr.cz*. Kromě nasbíraných dat ve formátu CSV musí také tento soubor obsahovat hlavičku s patřičnými údaji k autentizaci serveru. Ta je těsně před tím, než dojde k odeslání, k tomuto souboru přidána. Přesný formát, který je nezbytný pro úspěšný průběh zpracování, je popsán v kapitole 5.2.

Na začátku je serverem provedena kontrola, zda přijatá data neobsahují nedovolené znaky. Pokud tato kontrola proběhne v pořádku a data tedy obsahují pouze znaky, které jsou přípustné, tak je provedena autentizace serveru, ze kterého data pocházejí. V hlavičce přijatých dat musí být sekce oddělená `--hostname--`, která obsahuje položky s názvem serveru, doménovým jménem serveru a autentizačním řetězcem daného serveru. Na základě těchto položek je provedena kontrola, zda daný server skutečně existuje.

Pokud se autentizace serveru povede, tak se dále provede kontrola, zda přijatá data obsahují další sekci oddělenou `--zatez--`. V této sekci se předpokládá obsažení všech monitorovaných hodnot, které mají být uloženy do databáze. Každý řádek v této sekci musí být ve formátu CSV a musí jako první položku obsahovat název jedné z monitorovaných hodnot a následně dané nasbírané hodnoty.

Nejdůležitějším řádkem je řádek s hodnotami `TIME`. Řádek s hodnotami `TIME` musí existovat vždy, aby ke všem ostatním nasbíraným hodnotám mohla být přiřazena správná hodnota času, kdy došlo k jejich sběru. Následně je zkontrolováno jestli hodnoty `TIME` jsou v požadovaném formátu času a zjistí se jejich počet.

Pro všechny monitorované hodnoty, které jsou v této sekci očekávány, je provedena kontrola, zda existuje řádek s jejich názvem, zda jsou všechny hodnoty ve správném formátu vzhledem k jejich názvu a zda jejich počet je stejný jako počet hodnot `TIME`. Pokud by počet některých hodnot byl jiný, než je počet hodnot `TIME`, pak by tyto hodnoty nemohly být správně interpretovány, protože by nemohly být přiřazeny ke správnému časovému okamžiku sběru dat.

Dalším krokem, který se provede, je uložení všech hodnot do databáze. Nejprve se uloží hodnoty `TIME`. Následně se postupně provede uložení všech monitorovaných hodnot do patřičných tabulek podle jejich názvu. Pokud se mezi ukládanými hodnotami vyskytne název nového zdroje (jádro procesoru, disk nebo síťové rozhraní), který v databázi ještě není, tak se také provede jeho uložení do patřičné tabulky.

4.6 Návrh uživatelského rozhraní

Uživatelské rozhraní má za úkol vhodným způsobem vedoucímu zobrazovat všechna data, která jsou v rámci monitoringu zátěže nasbírána. Vhodnou grafickou reprezentací nasbíraných dat v závislosti na čase jsou grafy, ty jsou v rámci nového monitoringu ve velké míře využívány.

Uživatelské rozhraní bylo navrženo tak, aby vedoucí měl k dispozici několik různých pohledů na nasbíraná data. Před tím než byly jednotlivé pohledy vytvořeny, došlo k definování nejčastějších otázek, na které by vedoucí měl pomocí těchto pohledů nového monitoringu zátěže jednoduše získat odpověď. Mezi tyto otázky patří:

- Který server je nejvíce/nejméně zatížen?
- Kterým serverům dochází paměť?
- Které servery využívají odkládací prostor swap?
- Na kterém serveru je největší síťový provoz?
- Ze kterého serveru se nejvíce čte či na který se nejvíce zapisuje?
- Který uživatel celkově servery nejvíce zatěžuje?
- Které servery daný uživatel zatěžuje?
- Jak moc je zatížen jeden konkrétní server?
- Kteří uživatelé a jakými procesy případnou zátěž na serveru generují?
- Která jádra procesoru na daném serveru jsou zatížena na 100 %?
- Ze kterého disku se na daném serveru nejvíce čte, či na který se nejvíce zapisuje?

Pomocí jednotlivých pohledů, které byly v rámci nového monitoringu vytvořeny, může vedoucí snadno získat odpovědi na všechny tyto otázky.

Jeden pohled se zabývá vytížením konkrétního serveru jako celku, další pohledy pak vytížením jednotlivých zdrojů, které se na daném serveru nacházejí a uživateli, kteří s tímto serverem pracují. Další pohledy zobrazují statistiky všech uživatelů, kteří jsou v informačním systému KNOTIS registrovaní a statistiky všech serverů, které jsou v rámci KNOT využívány.

Nový monitoring zátěže pro každý z těchto pohledů v rámci informačního systému KNOTIS vytváří jednu novou stránku tohoto systému, na které jsou nejčastěji zobrazovány grafy jednotlivých monitorovaných hodnot v závislosti na čase. U všech grafů, které jsou v rámci nového monitoringu vytvořeny lze nastavovat hodnotu – datum od kdy a datum do kdy se mají grafy překreslit. Tato hodnota je společná a je vždy nastavena pro všechny grafy stejně, protože by nebylo vhodné mít pod sebou např. dva grafy, kde by každý z nich byl zobrazen pro jiné časové období.

Novou funkcionalitou u těchto grafů je také možnost rušit a zapínat zobrazení jednotlivý datových řad, které se v daném grafu nacházejí. Pokud například graf obsahuje datové řady pro všechny disky na serveru, tak pomocí tohoto rušení zobrazení si může vedoucí zobrazit pouze některé disky, které v danou chvíli potřebuje vidět. Další novou funkcionalitou je zobrazení vytěžujících uživatelů a procesů přímo v grafu. Po najetí myši na jednu z datových řad, která se v grafu nachází, se vedoucímu vždy zobrazí informace o uživatelích a procesech, kteří v tu danou chvíli server nejvíce vytěžovali. Nemusí tak složitě tyto informace vyhledávat v tabulce se statistikami TOP, jako tomu bylo v případě současného monitoringu.

4.6.1 Detail serveru 2

Tato stránka je v KNOTIS vytvořena jako náhrada za stránku *Detail serveru*, která je jedinou stránkou v KNOTIS v rámci současného monitoringu zátěže. Nová stránka má stejnou funkcionalitu jako stránka původní a to zobrazovat informace o vytížení jednoho konkrétního serveru. Způsob, jakým jsou tyto informace zobrazovány je však odlišný a je popsán v kapitole 5.4.

Na této stránce jsou v osmi grafech znázorněny hodnoty pro získání přehledu o všech hlavních zařízeních, které se na serveru nacházejí. Pohledem do těchto grafů tak vedoucí získá přehled o celkovém dění na daném serveru. Grafy, které jsou na této stránce zobrazeny jsou pojmenovány:

- **Procesor – celkově** – Zobrazuje hodnoty *CPU idle*, *CPU sys*, *CPU user* a *CPU wait* pro procesor jako celek v jednotkách % v závislosti na čase.
- **Paměť RAM, SWAP – volné GB** – Zobrazuje hodnoty *MEM free* a *SWAP free* v jednotkách GB i % v závislosti na čase.
- **Paměť RAM – active, cached** – Zobrazuje hodnoty *MEM active* a *MEM cached* v jednotkách GB i % v závislosti na čase.
- **Spuštěné procesy** – Zobrazuje hodnotu *TOP NUM PROCESSES* v závislosti na čase.
- **Disky – zaneprázdněnost** – Zobrazuje hodnotu *DISK busy* pro jednotlivé disky v jednotkách % v závislosti na čase.
- **Disky – čtení, zápis** – Zobrazuje hodnoty *DISK read* a *DISK write* pro všechny disky na serveru jako celek v jednotkách MB/s v závislosti na čase.
- **Síťová rozhraní – přijatá, odeslaná data** – Zobrazuje hodnoty *INTERFACE receive* a *INTERFACE transmit* pro všechna síťová rozhraní na serveru jako celek v jednotkách MB/s v závislosti na čase.
- **Otevřené sokety** – Zobrazuje hodnotu *OPEN SOCKETS* v závislosti na čase.

Pod těmito grafy se nachází tabulka se statistikami TOP, stejně jako v současném monitoringu. Je však přehlednější, protože neobsahuje záznamy procesů, které využívají procesor na 0 % a jsou tak z pohledu monitorování zátěže nepodstatné.

4.6.2 Detail serveru 2 – detail jader procesoru

Na této stránce jsou zobrazeny grafy všech jader procesoru, které procesor daného serveru obsahuje. Všechny tyto grafy zobrazují stejné hodnoty, vždy pro jiné jádro. Vedoucí tak má možnost zjistit, zda případné generovaná zátěž zatěžuje pouze jedno jádro, či jestli je rozložena mezi více jader. Počet grafů na této stránce se liší v závislosti na počtu jader, které procesor obsahuje. Grafy, které jsou na této stránce zobrazeny jsou vždy pojmenovány:

- **Procesor – core *N*** – Zobrazuje hodnoty *CPU idle*, *CPU sys*, *CPU user* a *CPU wait* pro dané jádro procesoru v jednotkách % v závislosti na čase. Kde *N* značí číslo tohoto jádra.

4.6.3 Detail serveru 2 – detail pamětí

Na této stránce je zobrazeno pět grafů, které se týkají vytížení fyzické paměti a odkládacího prostoru SWAP daného serveru. Tyto grafy poskytují detailnější pohled na tato vytížení, než grafy, které jsou zobrazeny na stránce *Detail serveru 2*. Grafy, které jsou na této stránce zobrazeny jsou pojmenovány:

- **Paměť RAM – active, cached** – Zobrazuje hodnoty *MEM active* a *MEM cached* v jednotkách GB i % v závislosti na čase.
- **Paměť RAM – volné %** – Zobrazuje hodnotu *MEM free* v jednotkách % v závislosti na čase.
- **Paměť RAM – volné GB** – Zobrazuje hodnotu *MEM free* v jednotkách GB i % v závislosti na čase.
- **Paměť SWAP – volné %** – Zobrazuje hodnotu *SWAP free* v jednotkách % v závislosti na čase.
- **Paměť SWAP – volné GB** – Zobrazuje hodnotu *SWAP free* v jednotkách GB i % v závislosti na čase.

Na první pohled by se mohlo zdát, že grafy, které zobrazují volné % jsou stejné jako grafy, které zobrazují volné GB i %. Je v nich však rozdíl v tom, že grafy s volnými % jsou zobrazeny s pevnou y-ovou osou od 0 % do 100 % a jsou tak vhodné ke zjištění celkového volného množství dané zobrazované hodnoty. Namísto toho, u grafů, které zobrazují volné GB i %, je y-ová osa omezena nejmenší a největší hodnotu dané datové řady, která je v grafu zobrazena. Grafy, které zobrazují volné GB i % jsou tedy detailnější.

4.6.4 Detail serveru 2 – detail disků

Na této stránce jsou zobrazeny čtyři grafy, které se týkají vytížení disků, které se na daném serveru nacházejí. Tyto grafy poskytují detailnější pohled na jednotlivé disky, než grafy, které jsou zobrazeny na stránce *Detail serveru 2*. Grafy, které jsou na této stránce zobrazeny jsou pojmenovány:

- **Disky – přenosy/s** – Zobrazuje hodnotu *DISK transfers* pro všechny disky na serveru jako celek v závislosti na čase.
- **Disky – zaneprázdněnost** – Zobrazuje hodnotu *DISK busy* pro jednotlivé disky v jednotkách % v závislosti na čase.
- **Disky – čtení** – Zobrazuje hodnotu *DISK read* pro jednotlivé disky v jednotkách MB/s v závislosti na čase.
- **Disky – zápis** – Zobrazuje hodnotu *DISK write* pro jednotlivé disky v jednotkách MB/s v závislosti na čase.

4.6.5 Detail serveru 2 – detail síťových rozhraní

Na této stránce jsou zobrazeny tři grafy, které se týkají vytížení síťových rozhraní, která se na daném serveru nacházejí. Tyto grafy poskytují detailnější pohled na jednotlivá síťová rozhraní, než grafy, které jsou zobrazeny na stránce *Detail serveru 2*. Grafy, které jsou na této stránce zobrazeny jsou pojmenovány:

- **Síťová rozhraní – receive** – Zobrazuje hodnotu *INTERFACE receive* pro jednotlivá síťová rozhraní v jednotkách MB/s v závislosti na čase.
- **Síťová rozhraní – transmit** – Zobrazuje hodnotu *INTERFACE transmit* pro jednotlivá síťová rozhraní v jednotkách MB/s v závislosti na čase.
- **Otevřené sokety** – Zobrazuje hodnotu *OPEN SOCKETS* v závislosti na čase.

4.6.6 Detail serveru 2 – uživatelé

Na této stránce se nachází tabulka, ve které jsou vypsáni jednotliví uživatelé, kteří se na daném serveru ve vybraném časovém období vyskytli ve statistice TOP, a tedy na serveru generovali nějakou zátěž. V jednotlivých sloupcích této tabulky je zobrazeno kolikrát se daný uživatel ve statistice TOP vyskytoval, kolika různými procesy server zatěžoval a na kolik procent tyto jeho procesy zatěžovaly procesor a fyzickou paměť. Pohledem do této tabulky tak vedoucí získá přehled o tom, kteří uživatelé daný server nejvíce zatěžují.

Pod touto tabulkou si pak vedoucí má možnost vybrat jednoto z uživatelů, kteří jsou v této tabulce uvedeni a nechat si pro něj vykreslit následující grafy:

- **Zatížení CPU – *username*** – Zobrazuje hodnoty *CPU user* daného serveru a *%CPU* ze statistiky TOP pro vybraného uživatele v jednotkách % v závislosti na čase. Kde *username* značí jméno vybraného uživatele.
- **Zatížení RAM – *username*** – Zobrazuje hodnoty *MEM active* daného serveru a *%MEM* ze statistiky TOP pro vybraného uživatele v jednotkách % v závislosti na čase. Kde *username* opět značí jméno vybraného uživatele.

Pohledem na tyto grafy tak má vedoucí možnost snadno porovnat celkové vytížení serveru s vytížením, které generoval vybraný uživatel.

4.6.7 Monitoring – statistiky uživatelé

Na této stránce se nachází podobná tabulka, jako je na stránce *Detail serveru 2 – uživatelé*. Její odlišnost spočívá v tom, že nezobrazuje uživatele a informace o nich pro jeden vybraný server, ale zobrazuje je jako souhrn přes všechny servery, na kterých monitoring zátěže probíhá. Navíc je v této tabulce přidán sloupec, ve kterém jsou uvedeny jména všech serverů, na kterých daný uživatel generuje zátěž a počet výskytů tohoto uživatele ve statistice TOP pro tyto servery. Pohledem do této tabulky tak vedoucí získá informace nejenom o tom, který uživatel generuje největší zátěž, ale také na kterých serverech a hlavně zda je tato zátěž generována pouze na jednom serveru, či je rozložena mezi více serverů.

Pomocí odkazů pak vedoucí může snadno přejít buď na stránku s detailem konkrétního uživatele nebo na stránku s detailem konkrétního serveru.

4.6.8 Monitoring – detail uživatele

Tato stránka zobrazuje zátěž, kterou daný uživatel generuje na jednotlivých serverech. Opět se na ní nachází tabulka, která je vytvořena ze statistiky TOP. V prvním sloupci této tabulky jsou uvedena jména jednotlivých serverů, na kterých se daný uživatel ve vybraném časovém období vyskytl ve statistice TOP, a tedy na nich generoval nějakou zátěž. V dalších sloupcích této tabulky je opět zobrazeno kolikrát se daný uživatel pro daný server ve statistice TOP vyskytoval, kolika různými procesy daný server zatěžoval a na kolik procent jeho

procesy zatěžovaly procesor a fyzickou paměť tohoto serveru. Pohledem do této tabulky tak vedoucí získá přehled o tom, které servery daný uživatel nejvíce vytěžuje.

Pod touto tabulkou si pak vedoucí má možnost vybrat jeden ze serverů, který je v této tabulce uveden a nechat si vykreslit grafy zátěže daného uživatele pro tento vybraný server. Tyto grafy jsou shodné s grafy, které se nachází na stránce *Detail serveru 2 – uživatelé*, viz kapitola 4.6.6.

4.6.9 Monitoring – statistiky servery

Tato stránka zobrazuje informace o vytížení všech serverů, na kterých monitoring zátěže probíhá formou jedné velké tabulky, která lze opět překreslit pro vybrané časové období. Hodnoty, které jsou v ní zobrazovány pro jednotlivé servery jsou následující: 100 - *CPU idle*, *MEM active*, 100 - *SWAP free*, *TOP NUM PROCESSES*, *DISK busy*, *DISK transfers*, *DISK read*, *DISK write*, *INTERFACE receive*, *INTERFACE transmit* a *OPEN SOCKETS*. Všechny tyto hodnoty ukazují průměrné zatížení daných zdrojů za vybrané časové období.

Pohledem na tuto tabulku tak má vedoucí možnost snadno zjistit, které servery jsou nejvíce vytěžovány z pohledu těch nejdůležitějších monitorovaných hodnot. Příslušnými odkazy pak může jednoduše přejít na stránku s detailem zvoleného serveru, na kterém potřebuje zjistit zdroj případné zátěže. Nebo naopak, pokud vedoucí potřebuje nalézt server, který není zatížen, aby na něm mohl být spuštěn nějaký náročný výpočet, provede to též pomocí této tabulky.

V celkovém výsledku lze říci, že uživatelské rozhraní nového monitoringu zátěže obsahuje více různých a detailnějších pohledů na zátěž než uživatelské rozhraní monitoringu současného. Nabízí tak vedoucímu snadnější a rychlejší orientaci při hledání případné zátěže.

Kapitola 5

Implementace nového monitoringu zátěže serverů

V této kapitole je popsána implementace jednotlivých částí nového monitoringu zátěže serverů. Ta je složena s několika skriptů, souborů funkcí a nových stránek informačního systému KNOTIS. Zasazení zdrojových souborů nového monitoringu zátěže do adresářové struktury zdrojových souborů informačního systému KNOTIS je zobrazeno v příloze C.1.

5.1 Implementace sběru a odesílání dat

Sběr dat je v novém monitoringu zátěže implementován pomocí skriptu `busymon2.sh`, který je spouštěn pomocí systémového nástroje *cron*[1]. Příkaz, kterým je skript `busymon2.sh` pravidelně spouštěn každý den v 1:00 hodinu vypadá takto:

```
00 1 * * * /bin/bash /root/busymon2.sh
```

Skript po svém spuštění vytvoří v adresáři, ze kterého je spuštěn dočasnou složku `busymon2-data`. Tato složka lze změnit pomocí konfigurační proměnné `WORK_DIR`. Dále skript vždy po uplynutí pravidelného časového intervalu 5 minut (lze nastavit pomocí proměnné `SEC_BETWEEN_SNAPSHOTS`), vytvoří soubor (dále jen snímek), do kterého uloží všechny monitorované hodnoty pro daný časový okamžik. Tento snímek je následně uložen do složky `busymon2-data`. Celkem tento skript vytvoří 288 snímků zátěže (lze nastavit pomocí proměnné `MAX_SNAPSHOTS`), než dojde k jejich dalšímu zpracování. Vše je nastaveno tak, aby se snímky vytvářely v intervalech 5 minut a po uplynutí 24 hodin došlo k jejich zpracování, odeslání a ukončení běhu skriptu. Ve chvíli, kdy dojde k ukončení běhu skriptu, také dojde k jeho opětovnému spuštění pomocí nástroje cron. Tento cyklus lze podle potřeby vedoucího pomocí konfiguračních proměnných a příkazu nástroje cron libovolně upravovat, tak aby se data sbírala či odesílala častěji nebo naopak.

Jednotlivé konfigurační proměnné, které lze na začátku skriptu `busymon2.sh` nastavovat jsou popsány zde:

- `WORK_DIR` – Slouží k nastavení dočasného adresáře, do kterého se mají ukládat jednotlivé snímky zátěže.
- `SEC_BETWEEN_SNAPSHOTS` – Je hodnota v sekundách, která složí k nastavení časového intervalu mezi vytvářením jednotlivých snímků zátěže.

[1] cron(8) – Linux manual page http://man7.org/linux/man-pages/man8/cron.8.html

- `MAX_SNAPSHOTS` – Složí k nastavení počtů snímků, které se mají vytvořit, než dojde k jejich odeslání.
- `TOP_MAX_PROCESSES` – Složí k nastavení maximálního počtu procesů, které se mají ukládat do statistiky TOP.

K samotnému sběru dat slouží funkce `take_snapshot()`, která vytváří jednotlivé snímky zátěže. Snímek, který je vytvořen touto funkcí se vždy jmenuje `out_time`, kde `time` je přesný čas snímku ve formátu `'+%Y-%m-%dT%H:%M:%S'`. V implementaci této funkce je ke sběru dat využito několika jednoduchých programů:

- `date '+%Y-%m-%dT%H:%M:%S'` – Slouží k získání hodnoty časového okamžiku sběru dat.
- `mpstat -P ALL 1 1` – Slouží k získání všech hodnot potřebných k monitorování vytížení procesoru.
- `cat /proc/meminfo` – Slouží k získání všech hodnot potřebných k monitorování vytížení fyzické paměti a odkládacího prostoru SWAP.
- `iostat -dpmxy -g ALL 1 1` – Slouží k získání všech hodnot potřebných k monitorování vytížení disků.
- `top -b -n 1` – Slouží k získání statistik TOP a počtu existujících procesů.
- `cat /proc/net/dev` – Slouží k získání všech hodnot potřebných k monitorování vytížení síťových rozhraní.
- `ss -an` – Slouží k získání počtu vytvořených soketů.

Po vytvoření všech snímků zátěže pomocí funkce `take_snapshot()` je ve složce `busymon2-data`, která tyto snímky obsahuje, spuštěn skript `busymon2.py`. Tento skript pomocí regulárních výrazů ze všech snímků v této složce vytvoří výsledný soubor ve formátu CSV, který je vždy pojmenován jako `busymon2_time`, kde `time` je opět přesný čas ve formátu `'+%Y-%m-%dT%H:%M:%S'`.

Skript `busymon2.py` má na svém začátku opět několik konfiguračních proměnných pomocí nichž lze nastavovat optimalizaci množství dat, které se odesílají a následně i ukládají do databáze, viz kapitola 4.3. Těmito konfiguračními proměnnými se vždy nastavuje limit, jak moc se dvě po sobě nasbírané hodnoty musí lišit, aby se obě uložili do výsledného souboru `busymon2_time`. Pokud daný limit není překročen, tak se hodnota považuje za stejnou jako hodnota předchozí a do výsledného souboru se neukládá.

- `cpu_idle_limit` – Limit pro hodnotu *CPU idle* v jednotkách % pro konkrétní jádro.
- `cpu_sys_limit` – Limit pro hodnotu *CPU sys* v jednotkách % pro konkrétní jádro.
- `cpu_user_limit` – Limit pro hodnotu *CPU user* v jednotkách % pro konkrétní jádro.
- `cpu_wait_limit` – Limit pro hodnotu *CPU wait* v jednotkách % pro konkrétní jádro.
- `cpu_all_idle_limit` – Limit pro hodnotu *CPU idle* v jednotkách % pro pro všechna jádra na serveru jako celek.

- `cpu_all_sys_limit` – Limit pro hodnotu *CPU sys* v jednotkách % pro pro všechna jádra na serveru jako celek.
- `cpu_all_user_limit` – Limit pro hodnotu *CPU user* v jednotkách % pro pro všechna jádra na serveru jako celek.
- `cpu_all_wait_limit` – Limit pro hodnotu *CPU wait* v jednotkách % pro pro všechna jádra na serveru jako celek.
- `mem_limit` – Limit pro hodnoty *MEM total*, *MEM free*, *MEM active*, *MEM cached*, *SWAP total* a *SWAP free* v jednotkách %.
- `device_limit_read` – Limit pro hodnotu *DISK read* v jednotkách MB/s pro konkrétní disk.
- `device_limit_write` – Limit pro hodnotu *DISK write* v jednotkách MB/s pro konkrétní disk.
- `device_limit_busy` – Limit pro hodnotu *DISK busy* v jednotkách % pro konkrétní disk.
- `device_limit_all_read` – Limit pro hodnotu *DISK read* v jednotkách MB/s pro všechny disky na serveru jako celek.
- `device_limit_all_write` – Limit pro hodnotu *DISK write* v jednotkách MB/s pro všechny disky na serveru jako celek.
- `device_limit_all_transfers` – Limit pro hodnotu *DISK transfers* pro všechny disky na serveru jako celek.
- `receive_limit` – Limit pro hodnotu *INTERFACE receive* v jednotkách MB/s pro konkrétní síťové rozhraní.
- `transmit_limit` – Limit pro hodnotu *INTERFACE transmit* v jednotkách MB/s pro konkrétní síťové rozhraní.
- `socket_limit` – Limit pro hodnotu *OPEN SOCKETS*.
- `top_num_processes_limit` – Limit pro hodnotu *TOP NUM PROCESSES*.
- `top_cpu_limit` – Limit pro hodnotu *%CPU* ze statistiky TOP v jednotkách %. Tento limit funguje jinak než všechny ostatní. Slouží totiž k tomu, aby se do výsledného souboru ukládaly pouze ty procesy, které vytěžují procesor na více procent než je hodnota tohoto limitu.

Pro jakýkoliv z těchto limitů shodně platí, že pokud je jeho hodnota -1, tak se do výsledného souboru ukládají všechny hodnoty, stejně jako v současném monitoringu zátěže.

Po ukončení skriptu `busymon2.py` dále skript `busymon2.sh` provede vytvoření souboru s názvem `busymon2Send.txt`. Do tohoto souboru je zapsána hlavička, která je potřebná k autentizaci daného serveru. Přesný formát této hlavičky je popsán v kapitole 5.2. Poté je do tohoto souboru zapsán obsah souboru `busymon2_time`, který obsahuje samotná data. Na závěr je soubor `busymon2Send.txt` pomocí nástroje `curl` odeslán na server `knot.fit.vutbr.cz`, dočasný adresář `busymon2-data` smazán a běh sktiptu `busymon2.sh` ukončen.

5.2 Formát odesílaných dat a implementace jeho kontroly

Data je potřeba odesílat ve správném formátu, aby poté mohla být správně interpretována. Celý formát odesílaných dat je zobrazen ve výpisu 5.1.

Hlavička odesílaných dat obsahuje části oddělené `--hostname--` a `--ip--`. První řádek `host name` obsahuje název serveru, ze kterého data pocházejí. Řádek `domain name` obsahuje doménové jméno daného serveru a řádek `authorization string` obsahuje autentizační řetězec daného serveru. Následuje sekce, ve které se nachází výstupy programů `ip route`, `ip -6 route` a `ip addr`. Oddělovačem `--ip--` hlavička končí. Za oddělovačem `--ip--` následuje oddělovač `--zatez--`, za nímž se již nachází samotná nasbíraná data ve formátu CSV.

Ke kontrole a uložení přijatých dat do databáze slouží skript `importZateze2.php`. Ten nejprve provede autentizaci serveru pomocí již existující funkce `autentizaceServeru($vstupniSoubor, $kontrolniNadpis = NULL)`, která je definována v souboru `pomFunkce.php`. Pokud autentizace proběhne úspěšně, tak následuje parsování jednotlivých řádků s daty, které se nacházejí za oddělovačem `--zatez--`. Aby parsování bylo úspěšné, tak všechny řádky musí být ve formátu CSV. Pro každý řádek je provedena kontrola, zda obsahuje stejný počet hodnot jako je počet hodnot s názvem `TIME`. K této kontrole slouží funkce `checkNumbersOfVals($time, $values, $name)`. Pro každý řádek je dále provedena kontrola, zda obsahuje hodnoty správného datového typu podle názvu dané hodnoty. K tomu slouží funkce `kontrolaRetezce($retezec, $typPolicka, $prazdny)` ze souboru `pomFunkce.php`. Řádky s hodnotami *CPU idle*, *CPU sys*, *CPU user*, *CPU wait*, *DISK read*, *DISK write*, *INTERFACE receive*, *INTERFACE transmit* a *%MEM* ze statistiky TOP musí obsahovat pouze kladná čísla s jedním desetinným místem a řádky s hodnotami *DISK busy*, *DISK transfers*, *MEM total*, *MEM free*, *MEM active*, *MEM cached*, *SWAP total*, *SWAP free*, *TOP NUM PROCESSES*, *OPEN SOCKETS*, *PID* a *%CPU* ze statistiky TOP musí obsahovat pouze celá kladná čísla. Řádky *COMMAND* a *USER* ze statistiky TOP musí obsahovat řetězce bez speciálních znaků a uvozovek.

Pokud jsou všechna data z pohledu kontroly v pořádku, tak dojde k jejich uložení do databáze vykonáním příslušných databázových dotazů pomocí funkce `query($dotaz)`, která je definována v souboru `queryFunkce.php`.

```
host name
---!!!---
domain name
---!!!---
authorization string
---hostname---
output of ip route
output of ip -6 route
output of ip addr
---ip---
---zatez---
TIME,
CPU_[0-9]{3}_Idle[%],
CPU_[0-9]{3}_Sys[%],
CPU_[0-9]{3}_User[%],
CPU_[0-9]{3}_Wait[%],
CPU_ALL_Idle[%],
CPU_ALL_Sys[%],
CPU_ALL_User[%],
CPU_ALL_Wait[%],
DISK_ALL_Read[MB/s],
DISK_ALL_Transfers/sec,
DISK_ALL_Write[MB/s],
DISK_.+_Busy[%],
DISK_.+_Read[MB/s],
DISK_.+_Write[MB/s],
INTERFACE_.+_Receive[MB/s],
INTERFACE_.+_Transmit[MB/s],
MEM_ACTIVE[MB],
MEM_CACHED[MB],
MEM_RAM_FREE[MB],
MEM_RAM_FREE_PERCET[%],
MEM_RAM_TOTAL[MB],
MEM_SWAP_FREE[MB],
MEM_SWAP_FREE_PERCET[%],
MEM_SWAP_TOTAL[MB],
OPEN_SOCKETS,
TOP_[0-9]{3}_COMMAND,
TOP_[0-9]{3}_CPU[%],
TOP_[0-9]{3}_MEM[%],
TOP_[0-9]{3}_PID,
TOP_[0-9]{3}_USER,
TOP_NUM_PROCESSES,
DISKS_Serial,
INTERFACES_hw_adr,
```

Výpis 5.1: Formát odesílaných dat. Za názvem na každém řádku vždy musí následovat jednotlivé nasbírané hodnoty oddělené čárkami.

5.3 Implementace načítání dat z databáze

Při načítání dat z databáze je využívána technologie AJAX, viz kapitola 3.7. Pomocí této technologie jsou vytvářeny požadavky typu GET[2], které jsou odesílány na server. Tyto požadavky slouží jako požadavky na načtení monitorovaných hodnot z databáze, které jsou následně využívány pro vykreslování jednotlivých grafů. Požadavky jsou vytvářeny ve funkcích `printChartsUserTop(from, to, idSession, userName, serverId)` a `printChart(statType, from, to, canvasId, titleText, labelString, idSession, coreNumber, userName)`, které jsou definovány v souboru `monitoringZatezeFunkce.js`.

Tyto požadavky jsou na serveru zpracovávány skriptem `monitoringZateze.php`. Tento skript má za úkol příchozí požadavky zkontrolovat a zprostředkovat načtení příslušných dat z databáze. Načítání dat z databáze mají za úkol funkce, které jsou definovány v souboru `monitoringZatezeFunkce.php`.

Každý z požadavků GET, který je odeslán na server musí obsahovat tyto položky:

- `ids` – Identifikátor sezení daného uživatele.
- `type` – Určuje data, která mají být z databáze načtena.
- `from` – Datum a čas prvního časového okamžiku, pro který mají být daná data načtena.
- `to` – Datum a čas posledního časového okamžiku, pro který mají být daná data načtena.

- využití technologie AJAX
- vytváření požadavků javascript/monitoringZatezeFunkce.js
- příchozí požadavky na data do ajax/monitoringZateze.php
- hlavní funkce, tahání dat z DB monitoringZatezeFunkce.php
- funkce doplnění vzorků

5.4 Implementace uživatelského rozhraní

- jednotlivé stránky server, uživatel, detaily ...

[2]GET – požadavek protokolu HTTP definovaný v RFC 7231 https://tools.ietf.org/html/rfc7231

Kapitola 6

Nasazená, testování, zhodnocení...

Kapitola 7

Závěr

Závěrečná kapitola obsahuje zhodnocení dosažených výsledků se zvlášť vyznačeným vlastním přínosem studenta. Povinně se zde objeví i zhodnocení z pohledu dalšího vývoje projektu, student uvede náměty vycházející ze zkušeností s řešeným projektem a uvede rovněž návaznosti na právě dokončené projekty.

Literatura

[1] Edmonds, B.: *User Commands FREE(1).* [Online; navštíveno 25.04.2018].
URL http://man7.org/linux/man-pages/man1/free.1.html

[2] Godard, S.: *Linux User's Manual IOSTAT(1).* [Online; navštíveno 25.04.2018].
URL http://man7.org/linux/man-pages/man1/iostat.1.html

[3] Godard, S.: *Linux User's Manual MPSTAT(1).* [Online; navštíveno 25.04.2018].
URL http://man7.org/linux/man-pages/man1/mpstat.1.html

[4] Matoušek, P.: *Síťové aplikace a jejich architektura.* VUTIUM, 2014, ISBN 978-80-214-3766-1.

[5] Mozilla and individual contributors: *Web developer guides – Ajax.* [Online; navštíveno 05.05.2018].
URL https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX

[6] Oracle Corporation: *MySQL 5.6 Reference Manual – What is MySQL?* [Online; navštíveno 05.05.2018].
URL https://dev.mysql.com/doc/refman/5.6/en/what-is-mysql.html

[7] PHP Documentation Group: *PHP Manual – Preface.* [Online; navštíveno 05.05.2018].
URL http://php.net/manual/en/preface

[8] Prokop, M.: *System Manager's Manual SS(8).* [Online; navštíveno 27.04.2018].
URL http://man7.org/linux/man-pages/man8/ss.8.html

[9] Python Software Foundation: *The Python Tutorial.* [Online; navštíveno 04.05.2018].
URL https://docs.python.org/2/tutorial

[10] Raymond, E. S.: *The Art of Unix Programming.* Addison-Wesley, 2003, ISBN 9780131429017.

[11] van Rossum, G.: *A Brief Timeline of Python.* Leden 2009, [Online; navštíveno 04.05.2018].
URL
http://python-history.blogspot.com/2009/01/brief-timeline-of-python.html

[12] Rábová, Z.; Hanáček, P.; Peringer, P.; aj.: *Užitečné rady pro psaní odborného textu.* FIT VUT v Brně, Listopad 2008, [Online; navštíveno 12.05.2015].
URL http://www.fit.vutbr.cz/info/statnice/psani_textu.html

[13] Shafranovich, Y.: *RFC 4180 Common Format and MIME Type for Comma-Separated Values (CSV) Files.* [Online; navštíveno 04.05.2018].
URL https://tools.ietf.org/html/rfc4180

[14] Stenberg, D.: *curl(1) - Linux man page.* [Online; navštíveno 04.05.2018].
URL https://linux.die.net/man/1/curl

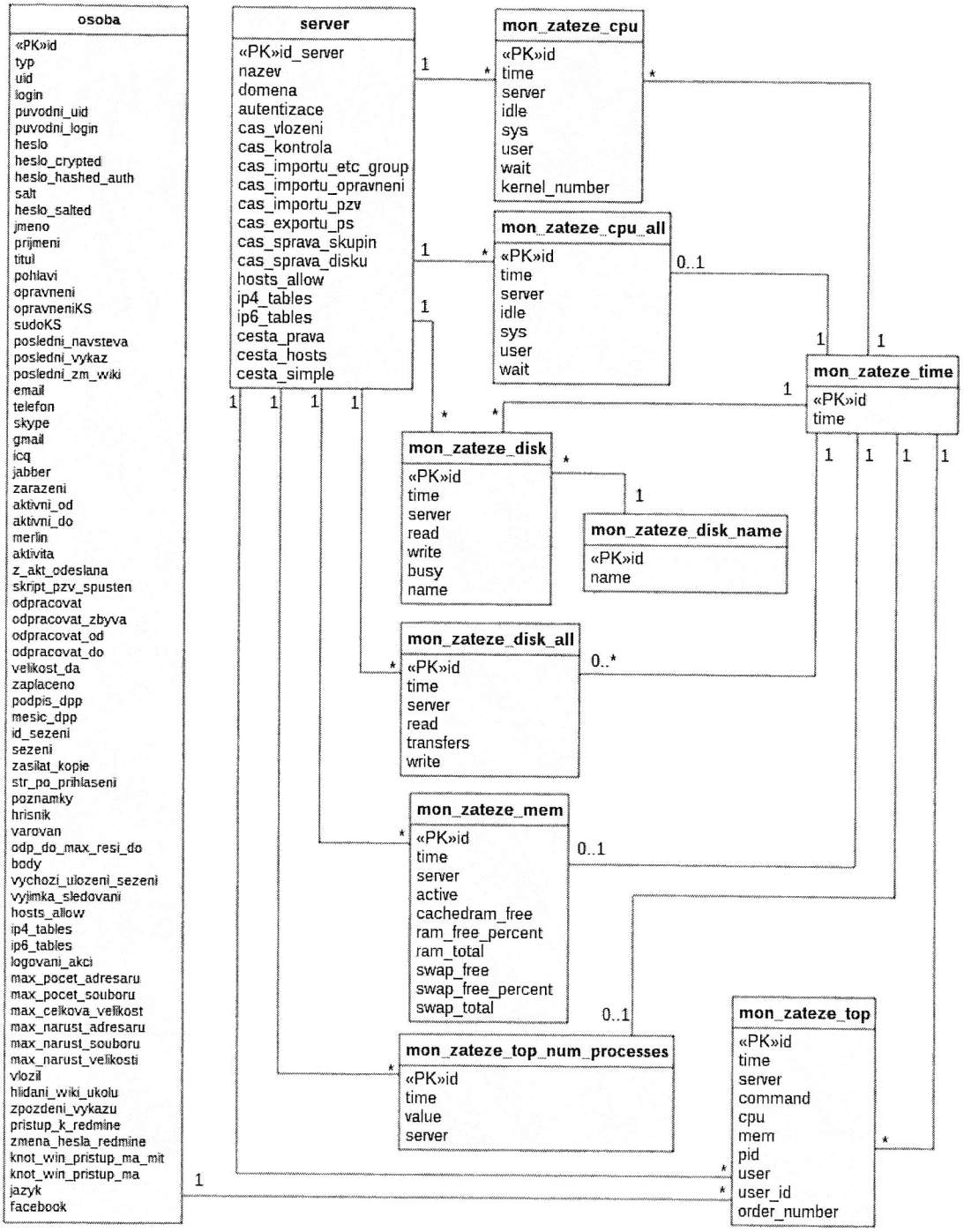
[15] Warner, J.: *User Commands TOP(1).* [Online; navštíveno 25.04.2018].
URL http://man7.org/linux/man-pages/man1/top.1.html

[16] Zendulka, J.; Rudolfová, I.: *Databázové systémy IDS Studijní opora.* FIT VUT v Brně, 2006.

Přílohy

Příloha A

Schéma databáze současného monitoringu zátěže serverů



Obrázek A.1: Schéma databáze současného monitoringu zátěže serverů v KNOTIS.

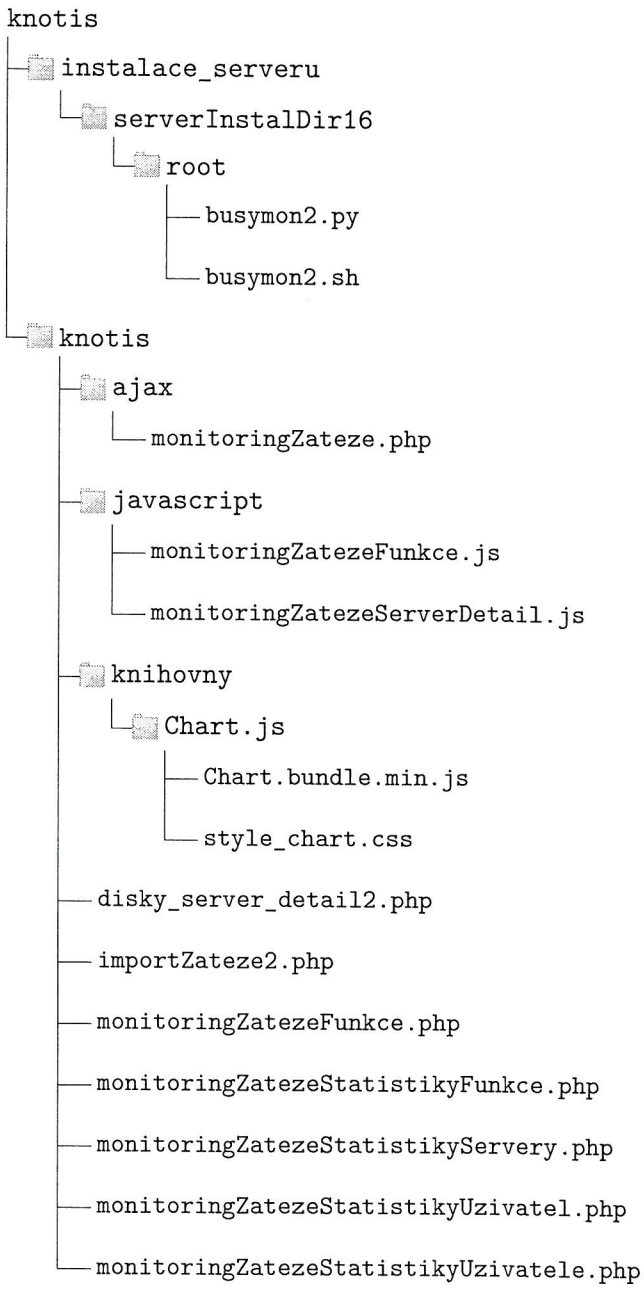
Příloha B

Schéma databáze nového monitoringu zátěže serverů

Obrázek B.1: Schéma databáze nového monitoringu zátěže serverů v KNOTIS.

Příloha C

Adresářová struktura zdrojových souborů nového monitoringu zátěže serverů



Obrázek C.1: Adresářová struktura zdrojových souborů nového monitoringu zátěže serverů v adresáři zdrojových souborů informačního systému KNOTIS.