Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

Analýza, návrh a implementace dohledového systému ve firemním prostředí

Bc. Martin Brož

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Martin Brož

Systémové inženýrství a informatika
Informatika

Název práce

Analýza, návrh a implementace dohledového systému ve firemním prostředí

Název anglicky

Analysis, design and implementation of monitoring system in a corporate environment

---

Cíle práce

Hlavním cílem práce je analýza, návrh a implementace vybraného dohledového systému ve zvoleném firemním prostředí

Dílčí cíle
- Přehled řešené problematiky
- Analýza firemního IT prostředí
- Výběr a návrh dohledového systému
- Implementace systému a jeho přizpůsobení pro potřeby firmy
- Posouzení přínosu nového systému ve zvoleném prostředí
- Diskuse, závěry a doporučení

Metodika

Metodika řešení teoretické části diplomové práce bude založena na studiu a analýze odborných informačních zdrojů a na praktických zkušenostech autora práce. Na základě těchto informací dojde k implementaci navrženého modelu dohledového systému, a to v reálném firemním prostředí. Funkčnost celého systému bude poté vyhodnocena a budou formulovány závěry práce.

Doporučený rozsah práce

50 – 60 stran

Klíčová slova

zabbix dohled SIEM SNMP proxy trigger agent bezpečnost

Doporučené zdroje informací

Handoro Semayat Fikre. Sledování výkonu vysoce složitých síťových systémů. Praha, 2021. Diplomová práce. Česká zemědělská univerzita v Praze. Katedra informačních technologií. Vedoucí práce Jiří Vaněk

OLUPS, Richard. Zabbix Network Monitoring Second edition. Birmingham: 2016. ISBN 1782161287

Schwartz, Baron. High Performance MySQL – 3rd. Edition. USA: O'Reilly Media, Inc, 2012. ISBN 1449314287

ZABBIX SIA. ZABBIX MANUAL [online]. Dostupné z Zabbix: https://www.zabbix.com/documentation/current/manual

---

Předběžný termín obhajoby

2021/22 LS – PEF

Vedoucí práce

doc. Ing. Jiří Vaněk, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 25. 6. 2021

doc. Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 10. 2021

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 24. 11. 2021

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Analýza, návrh a implementace dohledového systému ve firemním prostředí " jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 31. března 2022

Poděkování

Rád bych touto cestou poděkoval vedoucímu práce doc. Ing. Jiřímu Vaňkovi, Ph.D., za jeho cenné rady, připomínky a věnovaný čas. Dále bych rád poděkoval členům mé rodiny a omluvil se jim za čas, který jsem jim nemohl pro tuto práci věnovat.

Analýza, návrh a implementace dohledového systému ve firemním prostředí

Abstrakt

Diplomová práce se zabývá problematikou monitoringu firemní počítačové sítě a jejích komponent. V této práci se autor bude věnovat implementaci vybraného monitorovacího systému do reálného prostředí firmy a následnému přizpůsobení jejím potřebám.

Teoretická východiska analyzují různé důvody pro monitoring sítě a možnosti, jak toho dosáhnout. Jsou zde specifikovány základní síťové protokoly, které jsou určeny k monitoringu a jejich možnosti využití. Práce také přibližuje základní charakteristiky dohledových systémů a analyzuje, jakým způsobem je sběr dat až po jejich vizualizaci řešen. Součástí teoretického východiska je i analýza a shrnutí vlastností některých systémů určených primárně pro monitoring.

V praktické části je provedena analýza firemního prostředí, jejíž součástí je i sběr požadavků, které by měl navrhovaný systém splňovat. Na základě těchto informací je vybrán dohledový systém a navrhnuta jeho topologie. Následně je popsána instalace systému a jeho přizpůsobení dle požadavků firmy a to včetně přidání dalších požadovaných funkcí. V závěru práce dochází k posouzení jednotlivých kroků implementace včetně přínosů dohledového systému. Na základě získaných zkušeností jsou pak formulovány doporučení.

Klíčová slova: zabbix, dohled, SNMP, proxy, trigger, agent, bezpečnost

Analysis, design and implementation of monitoring system in a corporate environment

Abstract

The diploma thesis deals with the issue of monitoring the corporate computer network and its components. In this work, the author will focus on the implementation of the selected monitoring system into a real environment of the company and subsequent adaptation to its needs.

Theoretical background analyzes the various reasons for monitoring the network and how to achieve this. The basic network protocols that are intended for monitoring and their possibilities of use are specified here. The work also introduces the basic characteristics of surveillance systems and analyzes how data collection is solved up to their visualization. Part of the theoretical basis is also the analysis and a summary of the properties of some systems intended primarily for monitoring.

The practical part is an analysis of the corporate environment, which includes the collection of requirements that should comply with the proposed system. Based on this information, a surveillance system is chosen and its topology is designed. Subsequently, the installation of the system and its adaptation according to requirements is described including additional required features. At the end of the work there is an assessment of individual implementation steps, including the benefits of the supervisory system. Based on the obtained experience, recommendations are then formulated.

Keywords: zabbix, monitoring, SNMP, proxy, trigger, agent, security

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Monitoring informačních a komunikačních technologií (dále jen ICT) je v dnešní době nedílnou součástí každé sítě. V menších sítích a to zejména v domácích je sledování hardware či software možno realizovat svépomocí. Pokud se ale síť rozrůstá, přestává být v lidských silách v rozumné době zaregistrovat a následně zareagovat na vzniklé problémy. Proto téměř každá společnost dojde do bodu, kdy je potřeba centrálně dohlížet svěřené zdroje za pomocí automatizovaných systémů.

Systémy, které jsou dedikovány na monitoring ICT, nabízejí i další funkcionality, jako jsou např. vizuální zobrazení nasbíraných hodnot, jejich predikci vývoje a následné upozornění obsluhy na vzniklý problém. Nedílnou součástí bývají i automatizované úkony, které systém provádí na základě definovaných pravidel. Nejen z těchto důvodů se dohled nad ICT stává efektivním a je možno vzniklé problémy řešit skutečně rychle a automatizovaně s vynaložením méně času a úsilí.

Monitorovací systémy dokáží kontrolovat mimo samotný hardware i software. Díky tomu lze vyvíjet programy a v součinnosti s dohledovým systémem kontrolovat jejich správný chod. Tyto vlastnosti dávají společnosti se správně nastaveným dohledovým systémem možnosti a schopnosti, které může následně také využít k efektivnějšímu plánování rozvoje nákladů na ICT. Netřeba dodávat, že snižování nákladů na „nadbytečné“ prvky v ICT by měla být snahou každého, ať již z důvodů finančních či ekologických.

Dalším důležitým aspektem používání monitorovacích systémů je také zvýšení bezpečnosti celé ICT infrastruktury. Nad nasbíranými daty může být totiž prováděna další analýza, která může odhalit nebezpečí v podobě např. krádeže dat, neoprávněného přístupu apod.

Správné nastavení dohledového systému a jeho udržování v ideálním stavu samozřejmě s sebou nese nároky na kvalifikovanost obsluhy a tedy i cenu. Přes tyto aspekty jsou dohledové systémy ve firemních prostředích nasazovány a nelze si bez nich představit správné fungování sítě. Obdobně tomu bylo u firmy, která požádala autora této práce, aby jim pomohl s nasazením dohledového systému.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem práce je analýza, návrh a implementace dohledového systému ve zvoleném firemním prostředí.

Dílčí cíle:
- Přehled řešené problematiky
- Analýza firemního IT prostředí
- Výběr a návrh dohledového systému
- Implementace systému a jeho přizpůsobení pro potřeby firmy
- Posouzení přínosu nového systému ve zvoleném prostředí
- Diskuse, závěry a doporučení

2.2 Metodika

Metodika řešení teoretické části diplomové práce bude založena na studiu a analýze odborných informačních zdrojů a na praktických zkušenostech autora práce. Díky těmto informacím dojde k výběru a ke zprovoznění dohledového systému ve firemním prostředí dle požadavků skutečné firmy.
Teoretická východiska analyzují různé možnosti monitoringu sítě. Práce se bude věnovat základním síťovým protokolům, které jsou určeny pro monitoring síťových prvků a jejich možnosti využití. Součástí teoretického východiska bude i analýza trhu a shrnutí nabízených řešení.
V praktické části autor provede analýzu IT infrastruktury společnosti, včetně sběru požadavků na systém. Na základě těchto zjištění bude systém vybrán a navržena jeho topologie. Poté dojde k samotné instalaci systému a k jeho modifikaci dle potřeb firmy. Na konci práce budou shrnuty poznatky a popsána doporučení.

3 Teoretická východiska

3.1 Monitoring

Podniky a organizace, které při poskytování svých produktů a služeb závisí na informačních technologiích, musí vybudovat a udržovat IT infrastrukturu. Infrastruktura IT zahrnuje všechna aktiva nezbytná k poskytování a podpoře služeb IT: datová centra, servery, sítě, počítačový hardware a software, úložiště a další vybavení. Zatímco IT infrastruktura zahrnuje jak fyzická aktiva, tak virtuální aktiva (software, virtuální stroje, virtuální servery atd.), IT politiky a procesy spolu s lidskými zdroji nejsou považovány za součást IT infrastruktury.

Monitorování infrastruktury IT je obchodní proces, který vlastní a provozuje organizace IT. Jeho účelem je shromažďovat a analyzovat data z IT infrastruktury a využívat je ke zlepšení obchodních výsledků a vytváření hodnot pro organizaci.

IT organizace implementují specializované softwarové nástroje, které agregují data ve formě protokolů událostí z celé IT infrastruktury organizace. Protokoly událostí jsou automaticky generovány aplikacemi nebo zařízeními v síti, které tak reagují na síťový provoz nebo aktivitu uživatelů. Tyto soubory protokolu obsahují informace, jako je čas a datum, kdy k události došlo, uživatel, který byl přihlášen k počítači, název počítače, jedinečný identifikátor, zdroj události a popis typu události. Některé soubory protokolu mohou obsahovat další informace v závislosti na aplikaci, odkud pocházejí.

Softwarové nástroje pro monitorování infrastruktury IT zachycují soubory protokolů z celé sítě a agregují je do jediné databáze, kde je lze řadit, vyhledávat a analyzovat pomocí lidských nebo strojových algoritmů. Pomocí tohoto typu monitorování infrastruktury mohou IT organizace detekovat provozní problémy, identifikovat možná narušení zabezpečení nebo škodlivé útoky a identifikovat nové oblasti obchodních příležitostí. (1)

Monitorování prvků v sítě může být rozděleno:

Monitoring hardware - zachycují se data ze senzorů, které lze najít v počítačích a jiných strojích. Tyto informace poskytuje sám hardware. Mezi ně mohou patřit data o výdrži baterie, stavu disků, data ze senzorů výkonu a zátěže, proudu a napětí, data o rychlosti ventilátoru a další. Některé informace jsou shromažďována i operačním systémem, který je hostován na daném hardware.

Monitoring sítě pomáhá ověřit, zda síť funguje správně a poskytuje očekávané úrovně rychlosti a výkonu. Pomocí nástrojů pro monitorování infrastruktury IT lze sledovat přenosové rychlosti, latence a případné výpadky na síti. Z pohledu bezpečnosti lze v síti monitorovat také příchozí a odchozí připojení, počet posílaných paketů a jejich velikost. Na základě těchto informací lze v případě bezpečnostního incidentu zasáhnout proti neoprávněnému využití sítě.

Monitoring aplikací je kritickým aspektem monitorování IT infrastruktury. Softwarové aplikace a jejich bezchybný chod je důležitý pro správné fungování celého firemního řetězce. Při monitoringu jsou kontrolovány aplikace samotné, jejich chování či jejich logy. Při správném dohledu lze rychle na problémy reagovat a lze díky tomu zabránit velkým škodám, které mohou být vytvořeny nesprávným chodem aplikace či úmyslnou snahou útočníka poškodit IT infrastrukturu a to i třeba krádeží dat. (1)

3.1.1 Požadavky na monitoring

Zvyšující nároky na bezvadný chod počítačových sítí současně zvyšují i nároky na složitost jejich dohledu. Incidenty, které se na síti mohou stát, jsou těžko predikovatelné. Proto v roce 1989 Mezinárodní organizace pro normalizaci (ISO) vydala model síťového managementu[1], kterým definuje pět oblastí v IT, kterým pak přidává jednotlivé role v rámci monitoringu celé infrastruktury. Jednotlivé oblasti jsou známé také pod zkratkou FCAPS. (2)

- F = Fault (chybový management sítě),
- C = Configuration (konfigurační management sítě),
- A = Accounting (management účtování sítě),
- P = Performance (výkonnostní management sítě),
- S = Security (management bezpečnosti sítě).

Chybový management sítě
Hlavním úkolem této oblasti je vyhledávání chyb, reakce na nahlášené problémy, provádění diagnostických testů a samozřejmě samotná oprava chyb.

Konfigurační management sítě
Tato oblast má za úkol identifikovat, kontrolovat a shromažďovat data za účelem vytváření funkčních služeb, které budou kooperovat se systémem. Reaguje na nové skutečnosti v systému, vytváří metodiky postupů, jak monitoring provádět. Postupy je nutno udržovat aktuální, je tedy nutno postupy inovovat či je nahrazovat novými.

Management účtování sítě
Management účtování sítě je zodpovědný za analyzování využívání prostředků sítě, přidělování kvót uživatelům, za které je oprávněný vybírat poplatky.

Výkonnostní management sítě
Tento management je zodpovědný za měření, analýzu a řízení jednotlivých částí systému. Pomocníkem pro výkon této činnosti jsou také historická data nasbíraná z jednotlivých komponent a znalosti minulých skutečností.

Management bezpečnosti sítě
Hlavním úkolem tohoto managementu je zajištění bezpečnosti pro veškeré součásti systému. Nedílnou součástí je tedy vytváření, modifikace či rušení bezpečnostních politik v návaznosti na aktuální situaci. Hlášení bezpečnostních incidentů a pořádání

---

[1] ISO/IEC 7498-4

bezpečnostních školení pro uživatele či správce systému je činností tohoto managementu také.

3.1.2 Způsoby monitoringu

Aktivní
Jedná se o základní způsob monitoringu. Dohledový systém sám posílá dotazy na jednotlivé prvky sítě, které mu na jeho dotaz odpovídají. Nevýhodou tohoto typu monitoringu je to, že vše řídí samotný dohledový server, což zvyšuje nároky na jeho hardware. Každý dotaz otevírá samostatné spojení ke klientovi, což může při mnoha dotazech server dosti zatížit a to jak z hlediska využití CPU tak i vytížení síťové vrstvy. (3)

Pasivní
Oproti předešlému způsobu se pasivní model liší tím, že samo monitorované zařízení odesílá dohledovému serveru potřebná data- sledované hodnoty. Tento způsob šetří prostředky dohledového serveru, protože zodpovědnost za posílání dat má dohledované zařízení.
Pasivní monitoring je vhodný zejména při dohledu nad hodnotami, které se mění nepravidelně. Samotnému serveru může být tedy poslána nová hodnota až tehdy, když dojde k její změně či v případě, pokud dosáhne nějaké přednastavené hodnoty.

S agentem
Agent je malý program, který je spouštěn na dohlíženém zařízení a to většinou jako služba. On pak dle svého nastavení monitoruje potřebné položky a zasílá získané informace samotnému serveru, který s daty dále nakládá. Většinou bývá dodáván společně s dohledovým serverem jako celek. Nelze ho nainstalovat a spouštět všude, bývá tedy většinou v modifikacích pro systémy Windows či Linux. Komunikace se serverem může probíhat pasivně či aktivně.
Velká výhoda v použití agenta je ta, že dokáže získávat další informace, které by bez přístupu do operačního systému nebylo možno získat. Někteří agenti umožňují spouštět uživatelský kód vůči samotnému hostitelskému systému a tímto způsobem získávat data dle individuálních požadavků.
V dřívějších dobách v neprospěch agentů byla skutečnost, že pro svůj chod konzumovali mnoho systémových zdrojů, někdy dokonce způsobovali nestabilitu hostovaného operačního systému či sami byli bezpečnostní dírou v systému. Nutno dodat, že dnešní sofistikované dohledové systémy mají již výborně fungující agenty, kteří velmi rozšiřují dohledové možnosti, které by za pomocí ostatních tradičních protokolů nebylo možno jinak dosáhnout.

Bez agenta
Tento způsob získávání informací lze nazvat tradičním. Samotný dohledový systém získává informace dotazy, které směřuje přímo na dohlížené zařízení. Nutností je, aby server a dohlížený prvek společně podporovali protokol, kterým spolu komunikují. Veškeré úkony s daty pak dělá server sám, není zde tedy možná úprava či filtrace dat před přijetím, tak jak by to mohl udělat agent. (4)

3.2 Dohledový systém

K zajištění požadavků, kterým se věnovala kapitola výše, mohou pomoci dohledové systémy. Jejich hlavní funkcionalitou je právě sběr dat z monitorovaných prvků a zobrazení v čitelné podobě jeho uživatelům. Na základě těchto získaných hodnot je také možno:

- Ladit výkon systému - identifikace požadavků na zdroje a jejich balancování dle potřeby
- Řešení problémů – identifikace, diagnostika a oprava chyb
- Plánování – na základě sesbíraných dat lze lépe predikovat potřebu navýšení kapacit
- Vývoj a design – díky ucelenosti informací ze systému lze lépe analyzovat a tedy dále navrhovat vývoj systému.
- Simulace a výzkum – získané informace mohou být použity pro modelace chování systému či k jeho zkoumání (5)

3.2.1 Členění

Dohledové systémy lze dělit na základní, rozšířené a proaktivní. (6)

- Základní monitorovací systémy – jedná se spíše o jednoduché programy, které většinou využívají jeden protokol k monitorování. Z naprosté většiny případů se jedná o protokol ICMP[2], který podává jen základní informace o síťové dostupnosti. Tyto systémy jsou většinou jedno-uživatelské a pro svoji omezenou funkcionalitu se hodí spíše do menších sítí LAN.
- Rozšířené monitorovací systémy – to jsou dohledové systémy, které již využívají širokou paletu protokolů, kterými získávají informace. Díky variabilitě protokolů, jsou schopny získávat různé informace o zařízeních v síti. Některé systémy využívají i vlastních agentů, čímž rozšiřují možnosti dohledu o další hodnoty, které klasické protokoly nedokáží.
- Proaktivní dohledové systémy – jedná se prakticky o nadstavbu rozšířených monitorovacích systémů, kdy jsou tyto systémy na základě nadefinovaných skriptů schopny plnit automatizované úkoly a tedy i opravy. Tyto systémy se používají zejména ve velkých datacentrech, kde jsou schopny velkou část různých konfliktů vyřešit samostatně.

3.2.2 Složení

Dohledový systém lze charakterizovat jako ucelený systém, který pro potřeby této práce bude posuzován jako třívrstvý.

---

[2] ICMP – Internet Control Message Protocol

Nejnižší vrstva je zodpovědná za samotný monitoring prvků sítě a samozřejmě i tedy za sběr získaných dat. Druhá vrstva pak získaná data analyzuje a tyto již strukturovaná data předává nejvyšší vrstvě systému, která je zodpovědná za jejich intepretaci. Interpretace může být pomocí GUI [3], různých textových výstupů, grafů apod. Často bývá v této poslední vrstvě implementována část, která různými způsoby jako je email, SMS upozorňuje uživatele systému na vzniklý problém či na možný předpoklad vzniku problému. V této vrstvě může být dále implementováno automatizované chování systému, čímž může být například oprava vzniklého problému zjištěného z hodnot nasbíraných dat.
Výše uvedené vrstvy jsou pro tyto systémy typické, avšak jejich implementace včetně její šíře je již pro každý systém jedinečná.
Bližšímu popisu jednotlivých vrstev budou v této práci věnovány samostatné kapitoly.

3.2.3 Architektura

Jednotlivé vrstvy systému (vrstvy zodpovědné za sběr, analýzu a interpretaci dat) mohou být umístěny společně na stejném serveru - jedná se o architekturu centralizovanou. Pokud jsou tyto vrstvy oddělené - hovoříme o architektuře decentralizované neboli federativní / distribuované. (7)

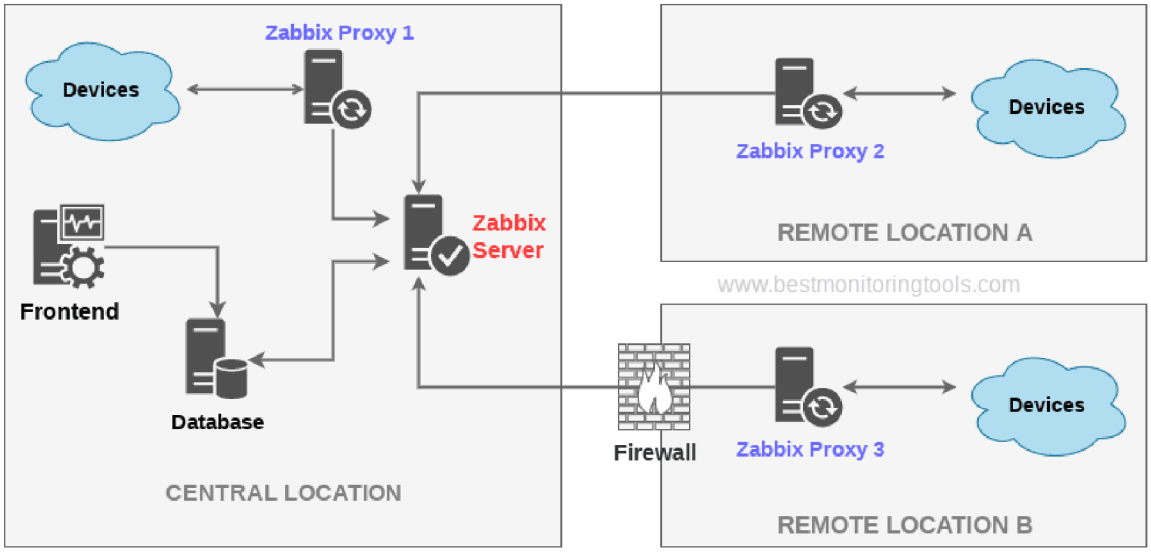
Centralizovaná architektura
Jedná se o méně složitou architekturu, než je architektura federativní. Veškeré vrstvy systému jsou instalovány společně na jedno zařízení - server. Z tohoto místa jsou pak žádosti o dotazy na prvky odesílány a přijímány.
Výhodou je jednodušší instalace, což je většinou implementováno jako instalace jediného instalačního balíčku, který veškeré potřebné moduly systému nainstaluje. Nevýhodou je naopak potřeba vyšších nároků na hostitelský hardware. Velkou nevýhodou je řešení vzniklých problémů. Většinou totiž při poruše některé z částí dojde k zastavení celého monitorovacího systému. Při monitoringu složitějších sítí je i z důvodu lepší propustnosti sítě a následné dosažitelnosti monitorovaných prvků vhodné uvažovat o architektuře federativní.

Federativní architektura
Pomocí této architektury lze dohledový systém rozdělit na více částí, které mohou být rozmístěny i v geograficky odlišných lokalitách. Této možnosti je využíváno většinou u složitějších systémů.
Rozdělením na více částí lze dosáhnout vyšší dostupnosti celého systému či jen jeho některých krizových komponent. Některé části mohou být v pasivním režimu, což znamená, že jsou pouze připravené převzít práci komponenty, která se dostane do chybového stavu. V tomto případě hovoříme o HA[4] systému. (8)
V případě monitorovacích systémů dochází často k decentralizaci části, která je zodpovědná za sběr dat. Díky této schopnosti může být „sběrná část“ přesunuta geograficky blíže k monitorovaným prvkům, což je využíváno ke snížení nároků

[3] GUI - Graphical User Interface (grafické uživatelské rozhranní)

[4] HA – High Availability

na kvalitu spoje či k rozdělení počtu úloh dohledového systému zodpovědných za monitoring.



Obrázek 1 - federativní architektura v systému Zabbix, převzato z (9)

Výše uvedený obrázek znázorňuje možnou variantu použití federativní architektury v systému Zabbix. Levá část popisuje rozdělení jednotlivých částí (frontend, database, server) systému včetně rozdělení vrstvy, která je zodpovědná za samotný sběr dat (pravá část obrázku). V případě Zabbix je to řešeno pomocí „Zabbix proxy“, která je schopna sama sbírat data a posílat je k vyhodnocení na server. Více bude rozebráno v dalších částí práce.

3.2.4 Sběr dat

Pro získání dat je pro monitorovací systém důležité znát „kontaktní“ informace monitorovaného prvku (IP adresa, DNS název) a komunikační protokol, pomocí něhož bude možno žádané informace získat. (10)
Způsoby, jakým monitorovací server získáván informace jsou pooling a trapping:

Pooling
Jedná se o častější způsob získávání informací. Server se aktivně na každou hodnotu ptá klienta, který mu po té odpovídá. Tento způsob je založen tedy na cyklickém dotazování. Počet dotazů a tedy i odpovědí si řídí samotný server na základě nastavení.
Výhody:

- Předvídatelná zátěž na monitorovací systém a dobrá predikovatelnost obsazenosti místa nasbíraných dat
- Data z pravidelných cyklů jsou vhodná pro predikci vývoje a další zobrazení v grafech
- Při nedostupnosti dat z cyklu lze usoudit na vznik problému

Trapping

Tento způsob dotazování oproti poolingu sbírá data od klientů tak, že je pouze shromažďuje. Nedochází tedy ke klasickému dotazování a čekání na odpověď. V první fázi dojde k nastavení klienta včetně cílové IP adresy monitorovacího systému a požadované závažnosti posílaných dat. Většinou je nabízeno několik úrovní závažnosti, jak je zobrazeno v tabulce níže:

Severity Number	Severity Name	System Response
0	Emergency	System unusable
1	Alert	Immediate action needed
2	Critical	Critical conditions exist
3	Error	Error conditions exist
4	Warning	Warning conditions exist
5	Notice	Normal but significant conditions exist
6	Informational	Informational messages
7	Debug	Debug messages

Obrázek 2 - klasifikace závažností zpráv, převzato z (11)

Výhody:

- Server se dozvídá požadované informace a to jen při jejich změnách
- Dohledovaný prvek sám upozorní na vznik možného problému

Porovnáme-li oba způsoby dotazování, dojdeme k závěru, že velkou nevýhodou pro trapping je nemožnost dohledového systému získat informaci o problému v případě síťové nedostupnosti prvku. Pokud server nezíská informaci z prvku, sám se totiž o ní nijak nedozví.
Autor by dále rád upozornil na skutečnost, že jednotlivá zařízení mohou posílat trapové informace s pro ně typickými zprávami - lze tedy těžko používat globální nastavení monitorování trapových dat komplexně na celý systém.
Pro výše uvedené důvody, jako je zejména nepředvídatelnost dat získaných z trappingu, je vhodné trapping používat společně spolu s poolingem.

Adresace

Jak již bylo výše napsáno, k samotné komunikaci je potřeba znát i adresu klienta popřípadě serveru samotného. Adresace se provádí na bázi IP adres či doménových jmen (DNS). V lokální síti LAN by se dalo uvažovat i o adresování za pomocí MAC

adres, přestože se takováto funkcionalita nepodařila autorovi v běžných dohledových systémech dohledat.
Některé systémy mají integrovány služby, které jsou schopny aktivně prohledávat síť a v případě nalezení nového prvku sítě jsou schopny upozornit obsluhu či automaticky nové zařízení dle předefinovaných pravidel začít monitorovat. V případě Zabbix se jedná o službu LLD (Low-level discovery).

Low-level discovery
Jedná se o zjišťování na nízké úrovni, které poskytuje způsob, jak automaticky vytvářet položky, spouštěče a grafy pro různé entity v počítači. Například Zabbix může automaticky spustit monitorování souborových systémů nebo síťových rozhraní na vašem počítači, aniž by musela být ručně vytvářena položka pro každý souborový systém nebo síťové rozhraní. Kromě toho je možné nakonfigurovat Zabbix tak, aby automaticky odstraňoval nepotřebné entity na základě skutečných výsledků periodicky prováděného zjišťování. (12)

3.2.5 Analýza a uložení dat

Dohledové systémy mohou nasbíraná data ukládat v nezměněné formě do svých datových úložišť anebo před uložením provézt jejich úpravu dle potřeby. Jednoduché systémy ukládají data do souborů, složitější do databází.
Výrobci dohledových systémů preferují dedikovanou SQL databázi, která bude schopna vyřizovat požadavky co nejrychleji. Například pro systém Zabbix jsou podporovány níže uvedené databázové systémy:

Software	Supported versions*	Recommended version	Comments
MySQL/Percona	5.7.28-8.0.X	8.0.X	Required if MySQL (or Percona) is used as Zabbix backend database. InnoDB engine is required. We recommend using the MariaDB Connector/C library for building server/proxy.

Software	Supported versions*	Recommended version	Comments
MariaDB	10.0.37-10.5.X	10.5.X	InnoDB engine is required. We recommend using the MariaDB Connector/C library for building server/proxy.
Oracle	12.1.0.2 - 19c	19c	Required if Oracle is used as Zabbix backend database.
PostgreSQL without partitioning	10.9-13.X	12.X.X or newer	Required if PostgreSQL is used as Zabbix backend database.
PostgreSQL with partitioning	10.9-13.X	13.X.X	Required if PostgreSQL is used as Zabbix backend database. Please note, that PostgreSQL versions below 13.0.0 may have issues handling operations that involve many partitions. Though some of the issues have been resolved in version 12,

Software	Supported versions*	Recommended version	Comments
			version 13 offers better performance.
PostgreSQL with *TimescaleDB*	TimescaleDB 1.5-2.1	PostgreSQL 12 with TimescaleDB 1.7/2.0 or PostgreSQL 13.2 or newer with TimescaleDB 2.1	Required if TimescaleDB is used as Zabbix backend database. Make sure to install the distribution of TimescaleDB with the compression supported.
SQLite	3.3.5-3.34.X	3.3X.X	SQLite is only supported with Zabbix proxies. Required if SQLite is used as Zabbix proxy database.

Tabulka 1 - podporované databázové systémy v Zabbix , převzato z (13)

Jelikož databáze bude neustále s přibývanými monitorovanými hosty a jejich položkami přibývat na velikosti, je vhodné si vypočítat předpokládanou velikost budoucí databáze a dle toho uzpůsobit velikost úložiště. Většina výrobců udává informace, jak jsou data ukládána do databáze a kolik j potřeba pro ně místa. Dobré je si uvědomit, že mimo klasických dat jsou ukládána i data jiná potřebná pro chod samotného systému jako jsou data o uživatelích, historie událostí apod. (12)

Parameter	Formula for required disk space (in bytes)
Zabbix configuration	Fixed size. Normally 10MB or less.

Parameter	Formula for required disk space (in bytes)
History	days*(items/refresh rate)*24*3600*bytes items : number of items days : number of days to keep history refresh rate : average refresh rate of items bytes : number of bytes required to keep single value, depends on database engine, normally ~90 bytes.
Trends	days*(items/3600)*24*3600*bytes items : number of items days : number of days to keep history bytes : number of bytes required to keep single trend, depends on the database engine, normally ~90 bytes.
Events	days*events*24*3600*bytes events : number of event per second. One (1) event per second in worst-case scenario. days : number of days to keep history bytes : number of bytes required to keep single trend, depends on the database engine, normally ~330 + average number of tags per event * 100 bytes.

Tabulka 2 - nastavení, které ovlivňuje velikost databáze, převzato z (13)

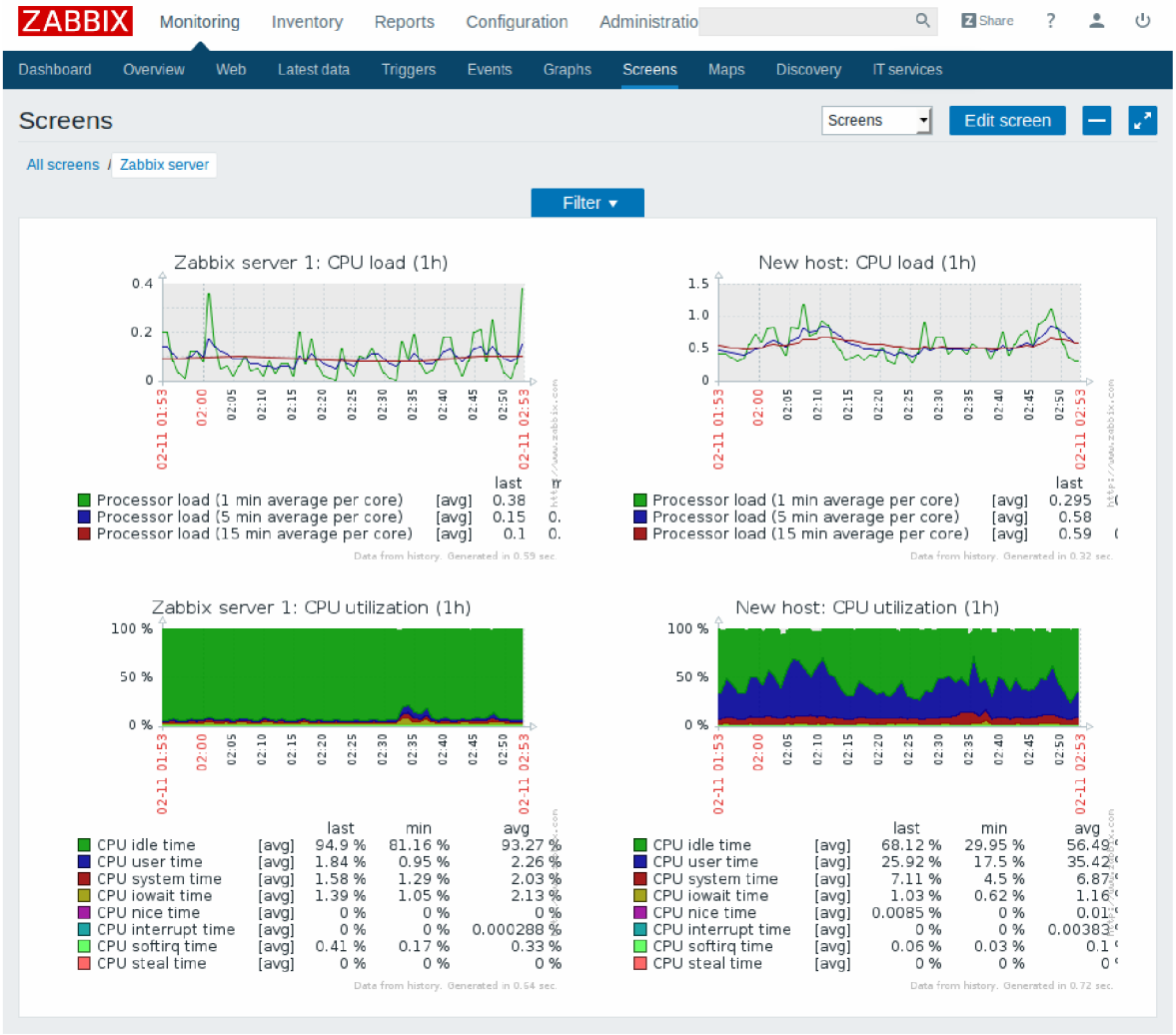
Jak je patrno z tabulky výše, samotná velikost databáze je z velké části determinována vlastním nastavením zodpovědným za ukládání dat. Proto je vhodné, před samotným nastavením položky ke hlídání důsledně určovat možnosti jako jsou délka uložení, historie dat, periodicita dotazů, typ ukládaných dat apod.
Většina systémů má dobře zpracované procesy, které se starají o úklid starých již nepotřebných dat. V případě Zabbix je tím procesem HouseKeeper - periodický proces, spouštěný Zabbix serverem. Tento proces z databáze v pravidelných cyklech odstraňuje staré informace a informace smazané uživatelem.
Některé systémy umožňují s daty provézt tzv. preprocessing – předem definovatelnou transformaci dat před uložením do samotné databáze. Touto transformací lze za pomocí vhodné metody (jako je např. regex) zkrátit ukládanou hodnotu o nadbytečná data.

Součástí každého systému má být scénář s možnostmi obnovení a záloh dat. Databázi je vhodné zálohovat co nejčastěji, avšak i samotná činnost zálohy má dopady na výkon a odezvu databáze. Uvědomíme-li si, že obnova databáze v případě nenadálé poruchy může trvat i několik hodin, je vhodné použít i databázovou replikaci. Ta umožňuje v případě potřeby rychlé přepnutí na záložní databázi (často i automatizované), což způsobí pouze krátkodobý výpadek v dostupnosti systému, což je právě u dohledových systémů žádoucí.

3.2.6 Interpretace dat

Přístup do dohledových systémů nutný pro samotné zobrazení dat či ke změnám nastavení systému bývá převážně realizován pomocí webových služeb (web-frontend) či pomocí API[5]. U menších dohledových systémů, které jsou většinou publikovány jako celistvé řešení, tato možnost většinou chybí a k systému se přistupuje napřímo rozhraním, které je naimplementováno již v samotném systému.

Web-frontend
Ve většině systémů jsou získaná data zobrazována pomocí webového frontendu, který má přístup do databáze, ze které čerpá data. Následně jsou data zobrazena v internetovém prohlížeči, kde jsou k vizualizaci používány grafy, tabulky a další komponenty, které zpřehledňují a zvyšují uživatelský komfort. Obrázek níže zobrazuje grafické znázornění nasbíraných dat v systému Zabbix.



Obrázek 3 - GUI systému Zabbix, převzato z (13)

[5] Application Programming Interface – rozhranní pro programování aplikací

Funkcionality webového rozhrání jsou koncipovány tak, aby zejména splňovaly (13):

- Aktuálnost zobrazovaných dat

Jednotlivé části stránek, které zobrazují získaná data, jsou načítány nezávisle na sobě a jsou pravidelně obnovovány. Tímto je docíleno co nejaktuálnějšího zobrazení hodnot. (Jelikož samotné načítání může vytěžovat DB, je četnost načítání individuálně nastavitelná).

- Vše na jednom místě

Snahou je, aby veškeré nasbírané hodnoty bylo možné sledovat z jednoho místa a tedy pomocí webového rozhraní. Zároveň je patrná snaha o začlenění možnosti nastavit systém také z webových stránek.

- Žádné restarty

Veškeré změny v nastavení (jak samotného zobrazení, tak i služeb systému) je provedeno ihned a není k tomu potřeba restartu služeb samotných.

- Vícejazyčnost

Snaha o možnosti volby jazykových mutací

- Auditování

Interakce se systémem je zaznamenávána a zpřístupněna administrátorům k analýze chování uživatelů či pro případné řešení bezpečnostních konfliktů

- Podpora více browserů

Snaha o podporu všech základních webových prohlížečů a snaha o reakci na jejich aktualizace

- Globální zobrazení upozornění

Centralizace všech upozornění ze systému na jediné stránce. V případě opravdu závažných problémů jsou závažné zprávy upřednostňovány a zobrazovány v popředí

Jelikož přístup přes webový frontend je součástí zabezpečení celého systému, doporučuje se nastavení webového serveru provézt následovně:

- Oprávnění procesu frontendu
    - Zajištění, že samotný proces webové služby bude používat co nejmenší možné oprávnění, aby při kompromitaci služby nebylo možno ovlivnit zbytek systém
- Použití SSL
    - Zajistit přístupnost serveru jen pomocí HTTPS protokolu. Toto opatření zajistí, že komunikaci mezi serverem a klientem nebude možno odposlouchávat či jinak modifikovat.
- Kořenový adresář služby jen pro systém
    - Zajistit přesměrování komunikace z jiné domény na kořenový adresář služby, kde bude umístěna základní stránka programu. Součástí přesměrování je i přesměrování právě přes protokol HTTPS.
- Využití HSTS
    - Mechanismus, který zabrání přechodu ze šifrované komunikace HTTPS na HTTP

- Využití CSP [6]
    - Technika, která zabezpečuje webové stránky zejména proti zneužití obsahu tím, že do hlavičky v odpovědi webového serveru ukládá informaci, která zakazuje manipulaci s obsahem
- Zakázání nadbytečných informací
    - Pomocí nastavení webového serveru se doporučuje zakázat poskytování nadbytečných informací o webovém serveru samotném. Servery obvykle v hlavičce odpovědi zasílají defaultně klientovi informace o svém názvu, verzi, času systému apod. Znalost těchto informací může potenciálním útočníkům pomoci lépe analyzovat cíl, přestože pro chod webových služeb nejsou potřebné.

API

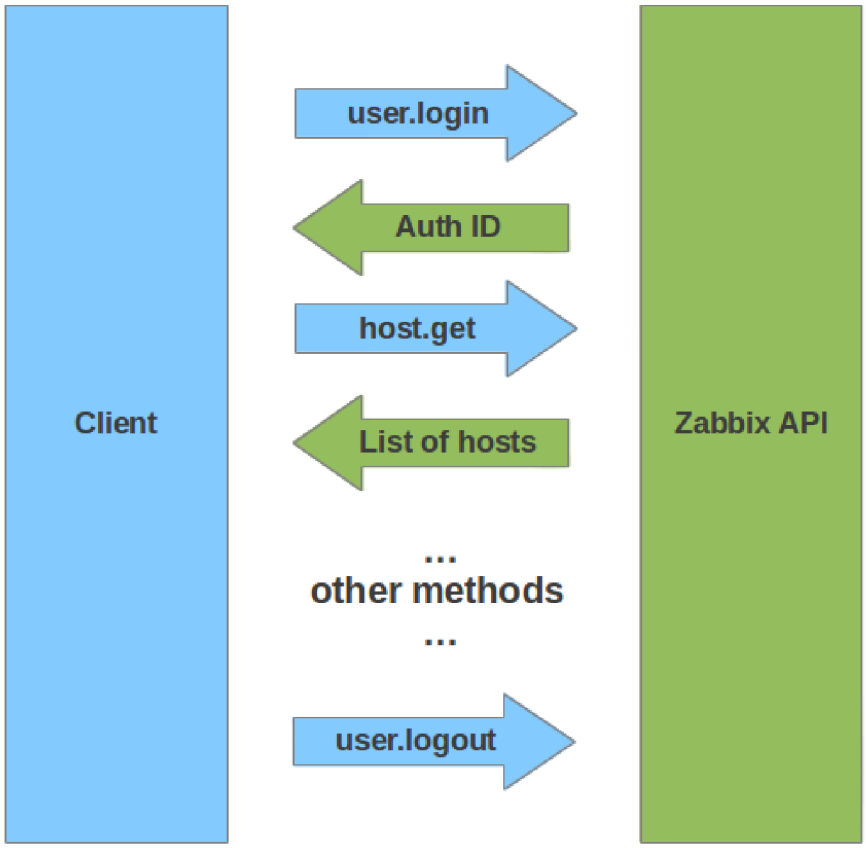
V počítačovém slovníku Foldoc (14) je API popisováno jako:

„Rozhraní, pomocí kterého aplikační program přistupuje k operačnímu systému a dalším službám. API je definováno na úrovni zdrojového kódu a poskytuje úroveň abstrakce mezi aplikací a jádrem (nebo jinými privilegovanými nástroji), aby byla zajištěna přenositelnost kódu.

Rozhraní API může také poskytovat rozhraní mezi jazykem vysoké úrovně a nástroji a službami nižší úrovně, které byly napsány bez ohledu na konvence volání podporované kompilovanými jazyky. V tomto případě může být hlavním úkolem API překlad seznamů parametrů z jednoho formátu do druhého a interpretace argumentů podle hodnoty a volání podle odkazu v jednom nebo obou směrech.“

V případě monitorovacích systému je většinou používáno API využívající JSON formátu. V prvé řadě dojde k autentizaci uživatele a k autorizaci, zda má právo k systému přistoupit. Pokud je uživatel autorizován, odpovědí je mu jedinečný klíč-token. Tento token musí být pak v každém dalším dotazu použit, aby systém věděl, že se jedná o daného uživatele. Každý dotaz je pak serverem autorizován, zda uživatel může takovýto dotaz vykonávat.
Schéma dotazování je na obrázku níže

[6] Content Security Policy (Zásady zabezpečení obsahu)



Obrázek 4 - komunikace pomocí API (systém Zabbix), převzato z (15)

V případě systému Zabbix, dotazování vypadá následovně:

- Autentizace (dotaz od klienta)

```
{
   "jsonrpc": "2.0",
   "method": "user.login",
   "params": {
      "user": "Admin",
      "password": "zabbix"
   },
   "id": 1,
   "auth": null
}
```

- Odpověď ze strany serveru

```
{
   "jsonrpc": "2.0",
   "result": "0424bd59b807674191e7d77572075f33",
   "id": 1
}
```

  - Zde je v poli result uveden token – autentizace proběhla v pořádku

- Dotaz na informace k uloženému hostovi

```
{
   "jsonrpc": "2.0",
   "method": "host.get",
   "params": {
      "output": [
         "hostid",
         "host"
      ],
      "selectInterfaces": [
         "interfaceid",
         "ip"
      ]
   },
   "id": 2,
   "auth": "0424bd59b807674191e7d77572075f33"
}
```

  - V parametru auth je uveden token, který byl získán metodou user.login
  - V dalších parametrech je možné samotnou funkci omezit nebo jinak upravit – v příkladu je uveden výčet požadovaných vrácených parametrů

- Výsledná odpověď od serveru


```
{
   "jsonrpc": "2.0",
   "result": [
      {
         "hostid": "10084",
         "host": "Zabbix server",
         "interfaces": [
            {
               "interfaceid": "1",
               "ip": "127.0.0.1"
            }
         ]
      }
   ],
   "id": 2
}
```


- Zde již odpověď od serveru, kde jsou vráceny požadované informace

K dispozici bývají i API pro ostatní programovací jazyky např. PHP, C#, ale to se jedná spíše o nadstavby, které využívají základní API systému (v případě Zabbixu je to JSON).

Komunikace se systémem pomocí Api je využívána zejména k automatizovanému dotazování či jiné automatizované činnosti se serverem. Další možností je také využití API k přístupu do systému aplikacemi třetích stran, což přidává další možnosti jak využít potenciál monitorovacího systému či další možnosti využití nasbíraných dat.

Moduly
Rozšíření frontendu o další funkcionality lze u některých systémů docílit přidáním modulů.
Dle zdroje IT slovník.cz (16) jsou moduly vysvětleny jako pluginy - jsou to nějaké kusy, které se připojí k běžící aplikaci a rozšíří se jimi funkčnost dané aplikace. Můžeme mít například platební modul, modul pro odesílání newsletterů, moduly pro přidání CAPTCHA ověřovacích kódů do formulářů.
Použitím modulů dáváme plný přístup softwaru třetích stran do monitorovacího systému. Proto je potřeba novému modulu respektive jeho výrobci důvěřovat. Problémy mohou nastat také při aktualizaci systému, po které může být modul nekompatibilní.

3.3 Monitorovací protokoly

Úlohou protokolů monitorování sítě je poskytovat základní statistiky a důležité informace týkající se různých síťových aktivit a informace o stavu prvků samotných. Jsou navrženy tak, aby usnadnily sledování dat a provozu proudícího do a ze síťových odkazů (hostitel a klient). Data shromážděná nástroji pro monitorování sítě pomocí

standardních protokolů se zobrazují graficky, aby správcům pomohla používat informace při správě činnosti sítě. (17)

3.3.1 ICMP

ICMP je síťový protokol ze skupiny protokolů využívající TCP/IP (18). Princip činnosti je takový, že odesílatel odešle příjemci požadavek na odpověď (echo request). Ten, pokud je dostupný, odesílá zpět informaci o své dostupnosti (echo response). Pokud však dotazovaný není dostupný, přichází zpět zpráva upřesňující důvod nedostupnosti. Aby dotaz neputoval sítěmi nekonečně dlouho, je do hlavičky paketu přidán údaj TTL[7] – číselný údaj, který je při každém přesměrování paketu (tzv. hop) síťovým prvkem snížen o 1. Pokud dojde ke snížení TTL na číslo 0, pak paket není dál přesměrováván a je zahozen. (Unixové systémy mají defaultně nastaveno TTL hodnotu na 64, Windows systém na 128).

Možné chybové zprávy zobrazuje tabulka níže:

ICMP message type	Description	Codes	IP Version
3	Destination Unreachable	0 - 15	IPv4
5	Redirect	0 - 3	IPv4
11	Time Exceeded	0 - 1	IPv4
12	Parameter Problem	0 -2	IPv4
4	Source Quench (Deprecated)	NA	IPv4
1	Destination Unreachable	0 - 8	IPv6
2	Packet Too Big	0	IPv6
3	Time Exceeded	0 - 1	IPv6
4	Parameter Problem	0 - 10	IPv6

Tabulka 3- chybové zprávy protokolu ICMP (zdroj: vlastní tvorba)

Jak je vidět z tabulky, každá zpráva obsahuje také kód (Codes), který blíže specifikuje problém.

[7] TTL – Time To Live

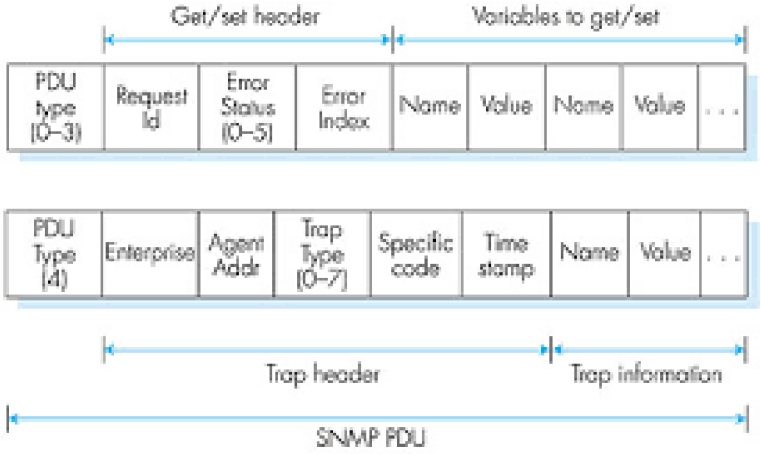
ICMP je důležitým protokolem, který je často používán jako první nástroj k ověření správného chodu sítě. Využíván je například příkazem PING či TRACERT, který zároveň podává informace o jednotlivých uzlech sítě, který paket (dotaz) směrovali dál.
Níže viz příkaz TRACERT na adresu google.com.

```
Tracing route to google.com [172.217.23.238]
over a maximum of 30 hops:

  1    17 ms    39 ms    59 ms  10.20.0.1
  2    56 ms    10 ms    20 ms  10.77.230.129
  3    12 ms    12 ms    13 ms  10.77.48.73
  4    20 ms    18 ms    14 ms  10.77.48.65
  5    12 ms    19 ms    14 ms  10.111.1.41
  6    20 ms    14 ms    13 ms  81.201.60.252
  7    17 ms    14 ms    16 ms  78.108.106.24
  8    22 ms    14 ms    56 ms  81.201.48.70
  9    26 ms    15 ms    33 ms  72.14.194.20
 10    16 ms    16 ms    23 ms  108.170.245.49
 11    23 ms    23 ms    33 ms  108.170.238.155
 12    16 ms    15 ms    24 ms  172.217.23.238

Trace complete.
```

Obrázek 5 - výstup z příkazu TRACERT (zdroj: vlastní tvorba)

3.3.2 SNMP

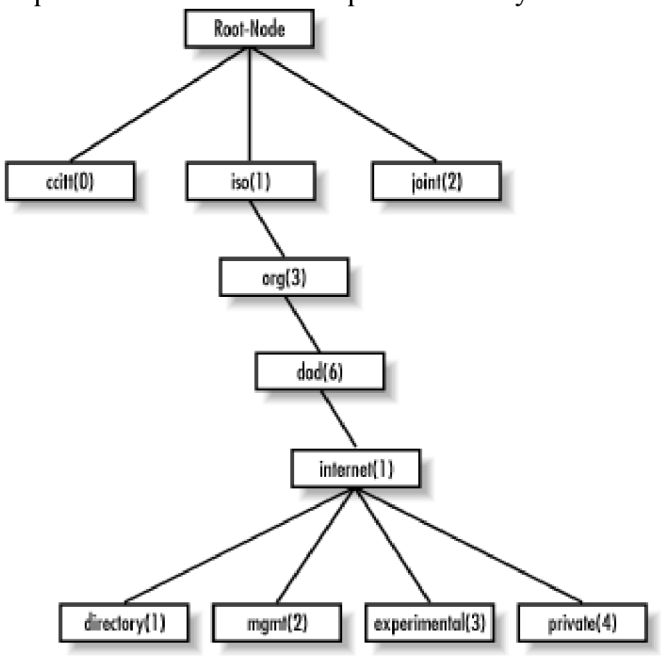
Protokol SNMP je asynchronní, transakčně orientovaný protokol založený na modelu klient/server (19). Ke komunikaci je využíván protokol UDP. Odesílatel (manažer) odesílá příjemci (agentovi) požadavky, na které agent odpovídá. Výjimkou jsou zprávy typu Trap, kdy sám agent zasílá zprávy manažerovi a to asynchronně. Tento protokol je velmi často užívaným protokolem pro správu a monitoring zařízeních. SNMP je ale i možno využít ke změnám nastavení zařízení.
SNMP má definovánu datovou jednotku PDU (protocol data unit), což je struktura zprávy, která je mezi zařízeními posílána.
PDU SNMP



Obrázek 6- PDU protokolu SNMP , převzato z (20)

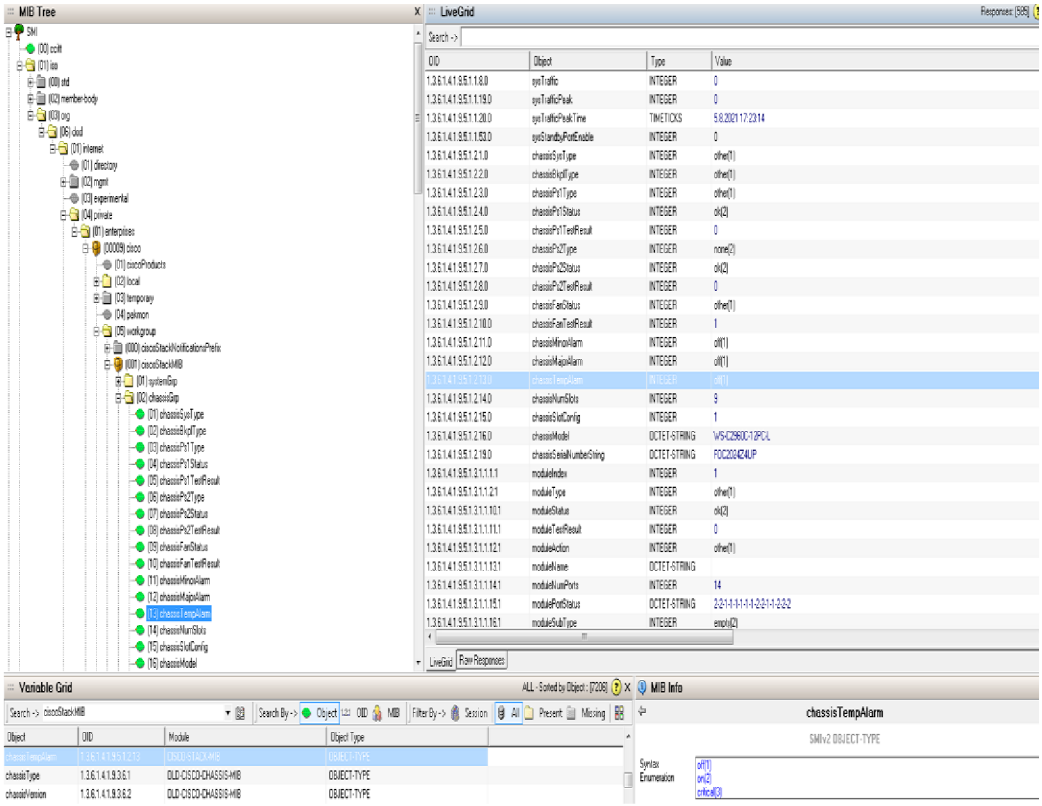
MIB databáze – každý příjemce SNMP dotazů má svoji MIB databázi. MIB databáze je strukturovaný seznam dat (obrázek č. 8), který je možno využít při dotazování. Jednotlivé záznamy obsahuji jedinečný identifikátor objektu (OID), který přímo

identifikuje hodnoty (proměnné), které lze číst či měnit. OID má strukturu čísel, které jsou odděleny tečkou.
Základní struktura databáze MIB je pevně daná. Pokud je potřeba do databáze přidat specifickou proměnnou, přidává se většinou do větve zvané *experimental* (nestandartní a nezařazené objekty) či *private* (specifické objekty pro dané řešení). K procházení databáze lze použít některý z MIB browserů či příkaz SNMPWALK.



Obrázek 7 - struktura databáze MIB , převzato z (21)

Níže je zobrazena část struktury MIB zařízení Cisco WS-C2960C-12PC-L, získaná programem OiDViEW. Zvýrazněna je položka *chasissTempAlarm*, která může nabývat hodnot 1 (off), 2 (on) a 3 (critical). V tomto případě vrací hodnotu 1 tedy teplotní sensor v šasi zařízení je ve stavu vypnuto.



Obrázek 8 - výstup z programu OidVieW (zdroj: vlastní tvorba)

Protokol SNMP prošel historickým vývojem a nyní je ve třech verzí.

SNMPv1

Jedná se o první verzi protokolu definovanou standardem RFC 1157 v roce 1988. Nadefinovány byly tyto PDU:

- GetRequest – manažer žádá informace od agenta
- GetNextRequest – manažer požaduje další OID v MIB
- Response – agent posílá odpověď na dotaz manažeru
- SetRequest – manažer posílá požadavek agentovi, aby nastavil určitou hodnotu
- Trap – agent posílá manažeru informaci nezávisle na vyžádání

K zabezpečení komunikace se používá tajné slovo „community string“, které musí znát jak manager, tak agent. Většinou jsou používány dvě slova - pro ověření na úrovni read a pro úroveň write.

Přestože se tato verze pro své slabé zabezpečení nedoporučuje používat, stále je mnoho sítí, kde lze prvky se SNMPv1 najít.

SNMPv2

Protokol známý i pod označením SNMPv2c vyšel v roce 1993 v RFC 1441. Oproti předchozí verzi obsahuje více kódů na zpracování chyb. Další změnou bylo přidání dalších PDU a to:

- GetBulkRequest – manažer požaduje velký počet dat ve formátu „tabulka“. K provedení příkazu je využíván příkaz GetNextRequest a to několikrát za sebou.
- InformRequest – používáno k potvrzení příjmu zprávy – obvykle Trapu

Jelikož zabezpečení v této verzi je ponecháno stále na „community stringu“, který je posílán jako plain text, lze považovat využívání této verze stále jako nebezpečné. Další nevýhodou je chybějící zpětná kompatibilita s verzí 1.

SNMPv3
Jedná se o nejnovější verzi protokolu SNMP, která byla představena v RFC 3410. Oproti předchozí verzi byl kladen důraz na zvýšení bezpečnosti. K zabezpečení se již používá uživatelské jméno (Username - obdoba „Community string“) a heslo pro autorizaci (Authentisation password) s klíčem (Privacy password). Samotná autorizace může být šifrována pomocí MD5 či SHA a komunikace se šifruje pomocí DES či AES.
Díky této verzi a zejména pro zlepšení její bezpečnosti začal být protokol SNMP plnohodnotným dohledovým protokolem. Předchozí verze totiž pro svoje nižší zabezpečení nejsou vhodná k použití pro nastavování prvků ale jen ke čtení informací z nich.
Zlepšení zabezpečení umožnilo vícero-uživatelského přístupu, kdy jsou jednotliví uživatelé identifikováni dle Username, Authentisation password a Privacy password.

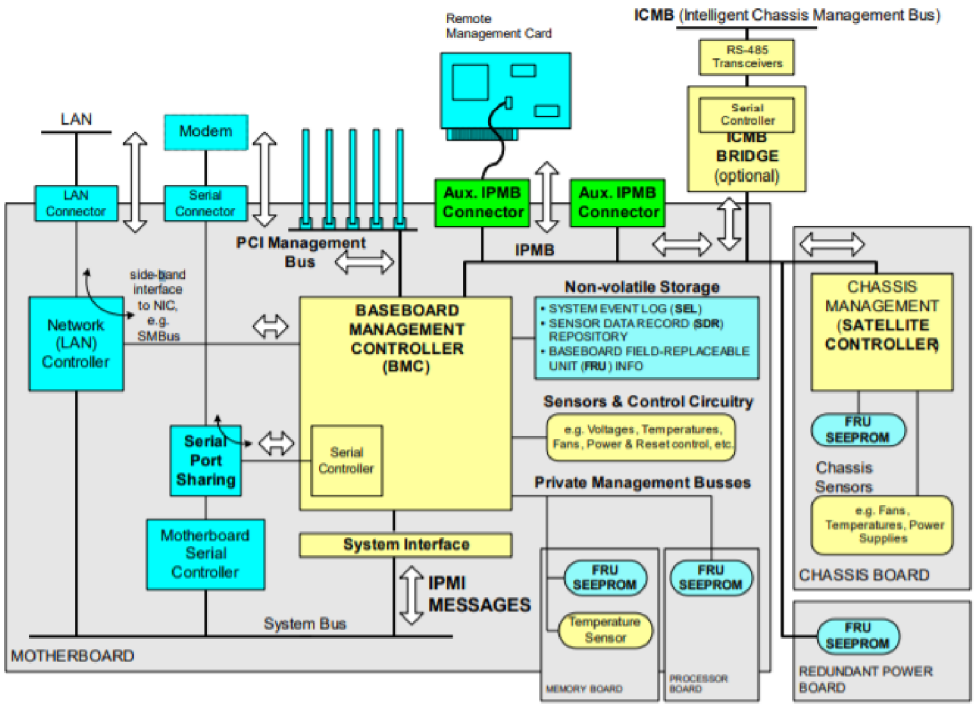
3.3.3 IPMI

Rozhraní IPMI (Intelligent Platform Management Interface) je specifikace hardwaru pro správu hardwaru s otevřeným standardem, která definuje specifický způsob komunikace mezi integrovanými subsystémy správy (22). Informace o IPMI jsou vyměňovány prostřednictvím řadičů pro správu základní desky (BMC), které jsou umístěny na hardwarových komponentách kompatibilních s IPMI, jako je servisní procesor (SP). Použití nízkoúrovňového přístupu k hardware oproti přístupu přímo k operačnímu systému má dvě hlavní výhody:

- Umožní správu serveru bez nutnosti mít spuštěný operační systém. Pro správu je potřeba mít jen funkční síťové připojení a napájení (server může být jen ve standby módu)
- Samotný operační systém není vytěžován dotazy klienta, jako je tomu u přístupu pomocí jiných protokolů (ssh, http apod.)

Obrázek níže zobrazuje, jak jsou jednotlivé serverové komponenty napojeny na BMC, který s nimi komunikuje. Interakce s okolím je umožněna pomocí síťové karty, kterou je vhodné dedikovat jen pro účely IPMI komunikace. Častou praxí bývá vytvoření vlastních VLAN [8]právě pro správu serverů pomocí tohoto protokolu.

[8] VLAN – virtuální LAN (logická síť)



Obrázek 9 - napojení serverových komponent na BMC, převzato z (23)

Výrobci serverů využívají IPMI pro webové nadstavby (např. u HP je to iLO, u Dell je to iDrac), které uživatelům grafickým způsobem umožňují monitoring a správu serveru.

Níže některé možnosti využití:

- Monitoring HW součástí – lze dohlížet jak aktuální stav zařízení, tak i jeho aktuálního hodnoty. Monitorují se např. disky, teplota, paměti, stav napájení apod.
- Systémové informace – stavy firmware, sériová čísla, licenční možnosti serveru a další, dávají lepší možnosti, jak udržovat informace o serverech aktuální
- Možnost aktualizace serveru bez nutnosti mít spuštěný OS. Některé aktualizace avšak potřebují restart serveru ke své instalaci
- Vzdálenou správu serveru pomocí virtuální konzole. Jedná se o přístup k serveru via KVM IP.
- Vzdálené připojení ISO obrazu – využíváno zejména ke vzdálené instalaci operačního systému
- Měření aktuální elektrické spotřeby – umožňuje balancování požadavků na servery ve vztahu ke spotřebě elektrické energie.
- Vzdálené vypnutí/zapnutí či restart severu
- Management přístupu – přístup k IPMI je multi-uživatelský a zabezpečený
- Nastavení proměnných a funkcionalit serveru jako jsou možnosti alertování, boot možnosti, NTP server apod.

IPMI je důležitý protokol, který je ve firemní infrastruktuře velmi využíván. Jelikož se jedná o protokol, který má možnost ovlivňovat hardware např. vypnutí, zapnutí apod., bývá zároveň bezpečnostním rizikem. Výrobci serverů doporučují aktualizaci

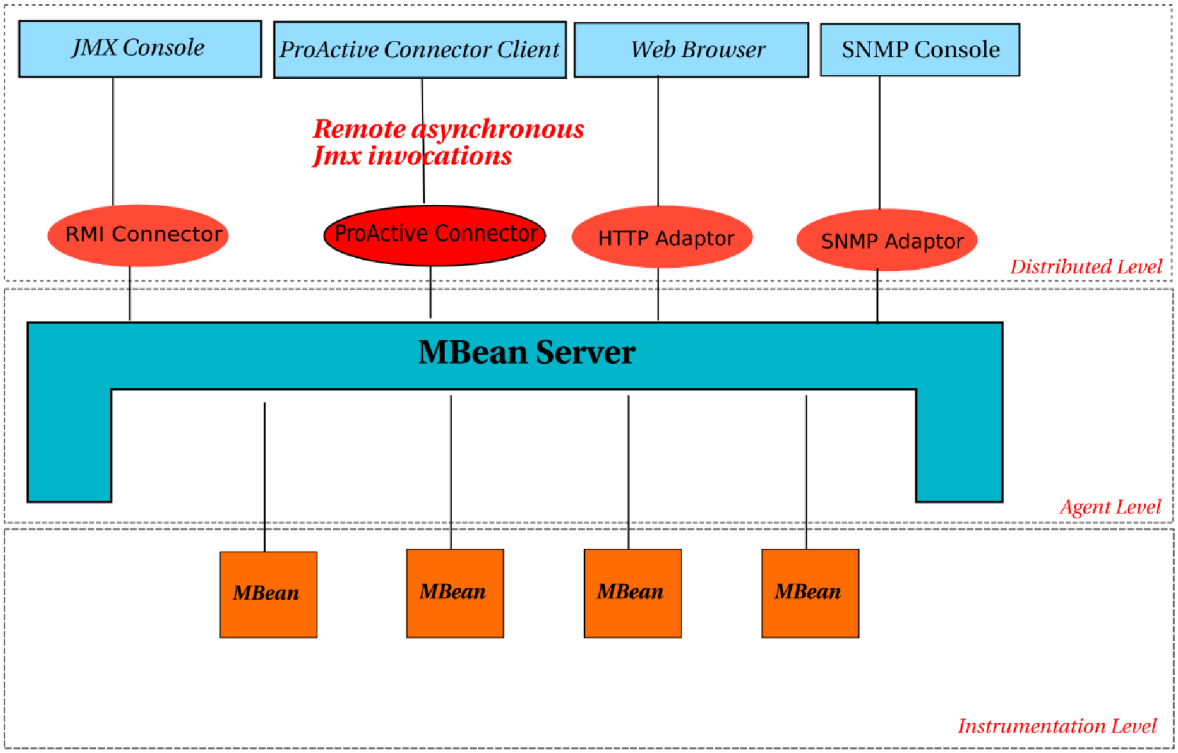
jejich webových nadstaveb na nejaktuálnější verzi, čímž opravují bezpečnostní rizika, ale i zvyšují stabilitu samotného produktu.

3.3.4 JMX

JMX (Java Management Extension) je rozhraní, které je psáno v jazyce Java. JMX umožňuje vytvářet aplikace s webovým rozhraním, které slouží k monitorování a správě různých zařízení, aplikací nebo sítí se službami. (24) Aplikace v rámci JMX jsou
vytvářeny pomocí Managed Bean (MBean) komponent, které zajišťují spojení se sledovaným objektem. Tyto MBean jsou zpracovány serverem – Mbean Server.

JMX je složeno ze tří vrstev – vrtsva instrumentace, agenta a vzdálené správy.

- Instrumentační vrstva – dohledované zdroje jsou převedeny na objekty – Mbean. Po té je možno k těmto objektům přistupovat pomocí agentů. Tato vrstva obsahuje informace o managementovatelných atributech objektu, operacích, kterých je možno s objektem dělat. Dále jsou v této vrstvě specifikovány způsoby jak reagovat na změny stavu objektu.
- Vrstva agenta – agent komunikuje s MBean a informace předává aplikacím vzdálené správy. Agenti nepotřebují předem znát, jaké zdroje budou spravovat. Zapotřebí je pouze splnění specifikace pro vzájemnou komunikaci. Samotný agent bývá spouštěn na dohledovaném zařízení, ale k zařízení může přistupovat i vzdáleně. Agent je součástí MBean serveru, který udržuje informace o MBean objektech. Tato vrstva je centrálním bodem architektury
- Vrstva vzdálené správy- umožňuje vzdálený přístup java aplikacím. Specifikace JMX definuje dva typy vzdáleného přístupu: konektory protokolu a adaptéry protokolu.
    - Konektory umožňují správci provádět volání metod na vzdáleném agentovi pomocí MBeanServeru (například Java Remote Method Invocation - Java RMI).
    - Adaptéry jsou komponenty, které zajišťují vazbu mezi konkrétním protokolem (například pro SNMP nebo HTTP) a spravovanými prostředky. Umožňují přístup k Mbeans pomocí stávajících prostředků.



Obrázek 10 - interakce s MBean serverem , převzato z (25)

Managed Beans (MBeans)
MBean je objekt Java, který implementuje konkrétní rozhraní za použití návrhových vzorů, které zodpovídá za monitoring a správu zdrojů aplikací. Rozhraní musí implementovat:

- Název a typ atributů, ke kterým je přístup (pomocí metody get či set)
- Název a typ operace, které lze vyvolat
- Oznámení, která lze odesílat
- Konstruktory pro třídu Java MBean

Princip činnosti je takový, že MBeans zapouzdří atributy a operace prostřednictvím svých veřejných metod a postupů dle návrhových vzorů a poté je vystaví aplikacím pro správu.
Libovolné objekty, které jsou implementovány jako MBean a registrovány u agenta, mohou být spravovány i mimo samotný stroj s agentem (24).
Architektura JMX neklade žádná omezení na to, kde jsou kompilované MBean třídy uloženy. Mohou být uloženy na libovolném místě s uvedením cesty ke třídě agenta JVM nebo na vzdáleném místě, pokud se používá dynamické načítání třídy

Specifikace JMX definuje čtyři typy MBean:

- Standardní
    - Jedná se o nejjednodušší typ, kdy vlastní rozhraní popisují názvy metod. Přístupy k parametrům jsou řešeny pomocí metod get a set. Samotné zjednodušení kódování na základě konceptu Open MBeans umožňuje efektivitu kódování.
- Dynamický

    - Tento typ také musí implementovat rozhraní, ale k jeho načtení dochází až v průběhu chodu aplikace.
- otevřený
    - Jedná se také o dynamický typ, který však k základním datovým typům přidává vlastní pro vyšší uživatelskou přívětivost. Základní typy jsou zachovány z důvodu univerzálnosti.
- model MBeans.
    - Jedná se o podobný typ jako v předchozím případě, avšak jeho vlastnost je rozšířena o možnost za běhu měnit konfiguraci.

Každý z nich odpovídá jiné potřebě instrumentace:

MBean server
MBean server je registrační místo pro MBeany, které mají být vystaveny pomocí JMX. Každý objekt, který je zde zaregistrovaný, se po registraci stává viditelným pro klientské aplikace. MBean server zpřístupní definované rozhraní jednotlivých instancí, ale nikdy nevystavuje celé instance.

Při registraci je instanci MBean přiřazeno unikátní jméno (unique object name), které pak aplikace pro správu používá k identifikaci objektu, na kterém má provést operaci správy. Operace dostupné v MBeans zahrnují:

- Objevování rozhraní pro správu MBeans
- Čtení a zápis hodnot jejich atributů
- Provádění operací definovaných MBeans
- Získávání oznámení vyslaných pomocí MBeans
- Dotazování MBeanů na základě názvu jejich objektu nebo jejich hodnot atributů

Služby agentů
Služby agentů jsou objekty, které mohou provádět operace správy na MBeans, které jsou
registrovány na serveru MBean. Služby agenta jsou často také MBeans, což umožňuje jejich ovládání a kontrolu jejich funkčnosti prostřednictvím serveru MBean. Specifikace JMX definuje následující služby agentů (24):
služby:

1. Dynamické načítání –
    - vytváření nových instancí z různých síťových umístění
2. Monitoring
    - Sledování číselné nebo slovní hodnoty atributu několika MBeanů. Při překročení může upozorňovat ostatní objekty na změnu monitorovaného attributu.
3. Časování
    - Umožňuje pravidelné nebo v předem nastaveném čase odesílání oznámení
4. Služba relací
    - Definuje vztah na základě předem definovaných relací a udržuje jejich konzistenci

3.3.5 Ostatní protokoly

K dispozici jsou i další protokoly, které nebyly vyvinuty primárně za účelem monitoringu, ale lze je takto použít. Níže budou zmíněny některé z nich.

SSH protokol[9]
Protokol, který slouží k bezpečné komunikaci se síťovými prvky přes nezabezpečenou síť. Pomocí příkazů zaslaných tímto protokolem, může monitorovací program posílat dotazy, které lze pak následně vyhodnocovat. Přidáním veřejného klíče monitorovacího serveru do seznamu autorizovaných klíčů monitorovaného hosta lze docílit možnosti dotazování bez následné nutnosti zadávat přihlašovací údaje.
Dotazy pak využívají schopnosti systému odpovídat na dotazy, jak je zobrazeno v obrázku níže, kde je zobrazen výstup ze systému Zabbix, kdy pomocí ssh dotazů je získáván na dotaz *date –t* lokální čas systému Redat. (získaná hodnota je ve formátu unix time)

Timestamp	lokalni cas
2021-07-21 23:58:12	1626904692
2021-07-21 23:55:13	1626904513
2021-07-21 23:52:13	1626904333
2021-07-21 23:49:12	1626904153
2021-07-21 23:46:12	1626903972
2021-07-21 23:43:13	1626903793

Obrázek 11 -odpověď na dotaz pomocí SSH příkazu (zdroj: vlastní tvorba)

Telnet[10]
Tento protokol lze využít stejným způsobem jako ssh, jen jeho použití není z bezpečnostních důvodů doporučováno. Komunikace není totiž šifrována a data jsou tedy posílána v čistém textu.

HTTP(S)[11]
Tento protokol je určen pro komunikaci s webovými servery. Pokud budou jeho odpovědi uzpůsobeny tak, aby reagovaly na dotazy zaslané monitorovacím serverem, lze jejich odpovědi použít ke sběrům požadovaných dat. Součástí dotazů mohou být i měnitelné parametry, které lze posílat pomocí parametrů v samotném dotazu v adrese URL.

[9] Blíže specifikuje norma RFC 4250
[10] Blíže specifikuje norma RFC 854
[11] Blíže specifikuje norma RFC 2818

3.4 Dohledový systém Zabbix

3.4.1 O systému

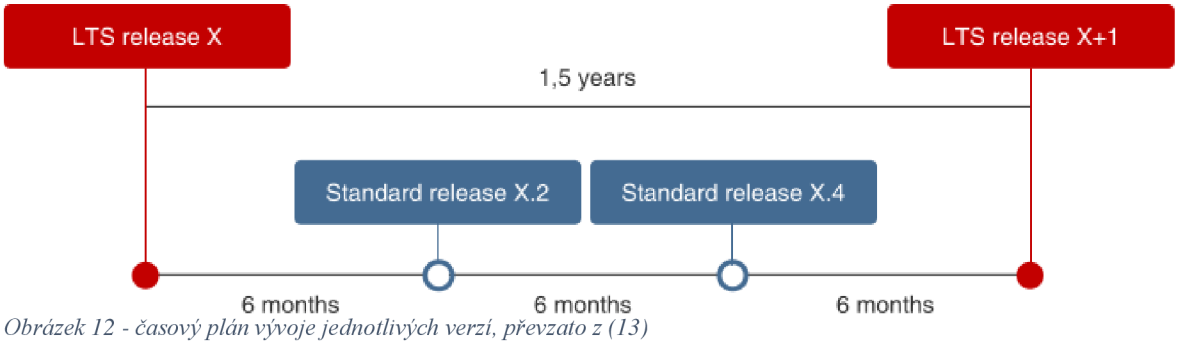
Dohledový systém Zabbix je ucelený enterprise-level software určen pro sledování dostupnosti a výkonnosti komponent IT infrastruktury. Zabbix je open source produkt a je k dispozici zdarma v plné verzi. Je to komplexní řešení, které je schopné sbírat, ukládat, řídit a analyzovat informace z IT systému, které pak následně zobrazuje na obrazovce. V případě problémů zasílá uživatelům zprávy pomocí například e-mailu či SMS zprávy.

3.4.2 Historie a současnost

V roce 1998 Alexei Vladishev vyvinul pro potřeby banky monitorovací nástroj. Ten byl pak po úpravách v roce 2001 nabídnut pod GPL licencí[12] - využití zdarma pro širokou veřejnost. V roce 2004 vyšla první verze Zabbix, která je neustále vyvíjena a nyní je ve verzi 5. Od roku 2005 je poskytována také komerční podpora- Zabbix SIA, která je poskytována v několika úrovních[13] dle výše poplatku. Dle informací ze Zabbix.com je nyní ve světě přes 300.000 instalací tohoto produktu. (13)
V současné době jsou nabízeny dvě verze:

- Standardní verze
    - Podporována 6 měsíců. Následující měsíc je podpora zúžena pouze na kritické a bezpečnostní problémy
    - Nová verze je každý 6. měsíc
    - Implementuje novinky, rychlejší vývoj
    - Pro rychlé stárnutí verzí není vhodná pro velké systémy, kdy je update komplikovaný
- LTS verze (Long Time Support)
    - Celková podpora je po dobu pěti let (první tři roky plná podpora, zbylé dva roky pouze pro kritické a bezpečnostní problémy)
    - Tato verze klade velký důraz na bezproblémový chod.
    - Novinky jsou implementovány až po ozkoušení na standardní verzi)
    - Pro vyšší spolehlivost a menší periodicitu aktualizací je vhodná do firemních prostředí

[12] GPL licence – obecná veřejná licence, licence pro svobodný software. Odvozená díla musí být pod stejnou licencí.
[13] Jednotlivé úrovně podpory Zabbix SIA - Silver, Gold, Platinum, Enterprise, Global I



Obrázek 12 - časový plán vývoje jednotlivých verzí, převzato z (13)

3.4.3 Složení

Zabbix je složen z několika komponent, které nemusí být instalovány na jednom místě – jedná se o distribuovaný systém. (26)

- Základní komponenty¨
  - Zabbix server - jádro a logika celého systému.
  - Zabbix databáze – databáze, kam se ukládají veškerá data.
  - Zabbix front-end – webové rozhraní, které umožňuje zobrazení dat a konfiguraci systému
- Volitelné komponenty
  - Zabbix proxy – Jedná se o odlehčenou verzi Zabbix serveru. Obsahuje databázi, ale nemá webové rozhraní. Její funkcí je sběr dat a zasílání je do centrálního bodu – Zabbix serveru. Používána je pro rozmělnění zátěže Zabbix serveru, nebo v případech horší konektivity, kdy data mohou být při nedostupnosti sítě po nějakou dobu cachována na proxy.
  - Zabbix agent – agent, který je zodpovědný za vykonávání dotazů, které jsou řízeny pomocí serveru či proxy.
  - Zabbix-get – jedná se o utilitu, díky které lze z příkazového řádku komunikovat s agenty a kontrolovat tak jejich činnost. Využívána je jak ke kontrole činnosti agentů, tak i ke kontrole samotného nastavení kontrolované položky.
  - Zabbix-sender – je to utilita, která komunikuje se serverem a posílá mu nastavené hodnoty. Sender může být použit jak ke kontrole komunikace (opačným směrem nežli je tomu u get), ale i jako doplněk k aplikacím, které ji mohou využít pro zasílání hodnot do serveru.

3.4.4 Funkcionality

Níže jsou funkcionality, které jsou publikovány na webu Zabbix.com

- Sběr dat
  - kontrola dostupnosti a výkonu
  - podpora monitorování SNMP (trapping i polling), IPMI, JMX, VMware
  - možnosti vlastní kontroly
  - shromažďování požadovaných dat ve vlastních intervalech
  - provádí server/proxy a agenti

- Flexibilní definice prahu problému
  - možnost definování spouštěčů (trigger, který je spuštěn při překročení nastavené meze)
- Vysoce konfigurovatelné upozornění
  - zasílání oznámení lze přizpůsobit plánu eskalace, příjemci, typu média
  - oznámení mohou být smysluplná a užitečná pomocí proměnných maker
  - automatické akce zahrnují vzdálené příkazy
- Grafy v reálném čase
  - sledované položky jsou okamžitě zobrazovány pomocí vestavěné funkce grafů
- Možnosti webového monitorování
  - Zabbix může sledovat způsobem simulovaných kliknutí myší na webových stránkách a kontrolovat funkčnost a dobu odezvy
- Rozsáhlé možnosti vizualizace
  - schopnost vytvářet vlastní grafy, které mohou kombinovat více položek do jednoho zobrazení
  - síťové mapy - prezentace v přehledu ve stylu dashboardu
  - reporty
  - pohled na monitorované zdroje na business úrovni
- Historické ukládání dat
  - data uložená v databázi
  - konfigurovatelná historie
  - vestavěný úklid nepotřebných dat (housekeeper)
- Snadná konfigurace
  - Jednoduché přidání sledovaných zařízení
  - Jakmile je zařízení přidáno, ihned dochází k monitoringu
- Použití šablon
  - seskupování pravidel pro monitoring do šablon
  - šablony mohou dědit vlastnosti z jiné šablony
- Prohledávání sítě
  - automatické zjišťování síťových zařízení
  - automatická registrace/odregistrace zařízení na základě prohledávání sítě
  - automatické objevování souborových systémů, síťových rozhraní a SNMP OID
- Webové rozhraní
  - Rychlé odezvy
  - Psáno v PHP
  - Jednotlivé položky jsou lehce přístupny
  - Auditování přístupu a provedených akcí
- Zabbix API
  - Poskytuje programovatelné rozhraní pro hromadné manipulace, integraci softwaru třetích stran a další účely.
- Systém oprávnění
  - zabezpečené ověřování uživatelů
  - integrace s LDAP
  - někteří uživatelé mohou být omezeni na určitá zobrazení
- Zabbix agent

    - Nasazují se na monitorovací cíle
    - lze nasadit na operační systém Linux i Windows
    - mnoho funkcí
- Binární démoni
    - napsané v C, pro výkon a malou paměťovou stopu
    - snadno přenosný
- Připraveno pro složitá prostředí
    - snadné vzdálené monitorování pomocí proxy serveru Zabbix

Zabbix prezentuje také roadmapu, což je strategický plán, který definuje cíle, kterých by mělo být v nějakém časovém horizontu dosaženo. Z této mapy je zřejmé, že funkcionality se budou nadále rozvíjet a přibývat, což jen potvrzuje, že Zabbix má velké ambice a je i velkým hráčem v oboru IT monitoringu.

3.5 Ostatní dohledové systémy

V současné době je na trhu mnoho dohledových systémů, které mají různé funkcionality. Mnoho z nich je šířeno volně, tedy bez povinnosti platit licenční poplatky. Zpoplatněny bývají pouze přidané služby, které pomáhají s údržbou a integrací systému.

3.5.1 Nagios

Nagios je vyvíjen od roku 1996 (27). V současnosti má přes 5000 pluginů, které tento systém rozšiřují o jeho další možnosti. Nagios se opírá o silnou komunitu, která čítá přes 1 mil. uživatelů. Během vývoje získal mnoho ocenění např. v roce 2013 to byla cena *Network Monitoring Application of the Year* a to po osmé za sebou.
Nagios-Core, což je základ pro další nadstavby uvedené níže, je volně šiřitelný dle licence GNU GPL2. Tato verze se pyšní více než 8 mil. instalacemi po celém světě. Její nevýhodou jsou nároky na administrátory systému, protože samotné nastavení a práce se systémem není tak jednoduchá jako je tomu u placené verze Nagios XI. Primárně je počítáno, že core edice bude ovládána z příkazové řádky.

Nagios XI

Nagios XI poskytuje monitorování všech důležitých infrastrukturních komponent včetně aplikací, služeb, operačních systémů, síťových protokolů, metrik systému a síťové infrastruktury. Stovky doplňků třetích stran zajišťují monitorování prakticky všech interních a externích aplikací, služeb a systémů. Instalován může být na serverech s Windows či Linux operačním systémem. Možností je instalace předpřipraveného obrazu virtuálního serveru pro VMWare.
Protože hlavní výhodou této verze oproti core verzi je uživatelská přívětivost a snaha o co nejjednodušší způsob ovládání celého systému, je tato verze ve firemních prostředích upřednostňována.
Funkce:

- Komplexní monitorování

    - Schopnosti monitorovat aplikace, služby, operační systémy, síťové protokoly, metriky systému a komponenty infrastruktury pomocí jediného nástroje
    - Výkonná skriptová API umožňují snadné monitorování interních i vlastních aplikací, služeb a systémů
- Zobrazení a upozornění
    - Centralizovaný pohled na celou monitorovanou IT infrastrukturu
    - Podrobné informace o stavu dostupné prostřednictvím webového rozhraní
    - Rychlá detekce výpadků infrastruktury
    - Výstrahy lze doručovat technickému personálu prostřednictvím e -mailu nebo SMS
    - Možnosti eskalace zajišťují, že se výstražná upozornění dostanou ke správným lidem
- Náprava problému
    - Potvrzení výstrah poskytují komunikaci o známých problémech a reakci na problémy
    - Obslužné rutiny událostí umožňují automatické restartování neúspěšných aplikací a služeb
- Proaktivní plánování
    - Trendy a doplňky (jako je plánování kapacity) zajišťují, že jste si vědomi stárnoucí infrastruktury
    - Plánované odstávky umožňují potlačení výstrah během upgradů infrastruktury
- Reporty
    - Zprávy o dostupnosti zajišťují dodržování SLA
    - Historické zprávy poskytují uložené záznamy výstrah, oznámení, výpadků a reakcí na výstrahy
    - Doplňky třetích stran rozšiřují možnosti hlášení
- Možnosti více uživatelů
    - Víceuživatelský přístup k webovému rozhraní umožňuje zúčastněným stranám sledovat stav infrastruktury
    - Pohledy specifické pro uživatele zajišťují, že klienti vidí pouze jejich součásti infrastruktury
- Rozšiřitelná architektura
    - Integrace s vlastními aplikacemi a aplikacemi třetích stran je snadná díky API rozhraní
    - Stovky doplňků vyvinutých komunitou rozšiřují základní funkce Nagiosu

Nagios Log Server

Jedná se o monitorovací nástroj, který je dedikován pro práci s logy. Dokáže logy ze systémů shromažďovat, analyzovat a popřípadě upozorňovat na problémy, které zjistil právě z logů.

Nagios Fusion

Jedná se o řešení, které centralizuje výsledky monitoringu na jedno místo. Fusion je doporučováno používat při potřebě monitorovat stav IT služeb, které jsou rozděleny do více geolokací.

3.5.2 Checkmk

Jedná se o produkt, jehož začátek vývoje se datuje od roku 2008 (28). Jako svůj základ používal volně šiřitelné jádro Nagios-core. Postupem času docházelo k obměnám jádra, až vznikl vlastní produkt, který nahradil celý základ Nagios.
Checkmk je nyní dostupné v několika edicí:
Checkmk Raw

- Stále využívá jádro Nagios
- Zdarma dostupné
- Jedná se o kompletní monitorovací systém
- Neobsahuje přidané hodnoty jako další edice

Checkmk Enterprise

- Jedná se o komerční řešení
- Jádrem je již Checkmk Microcore

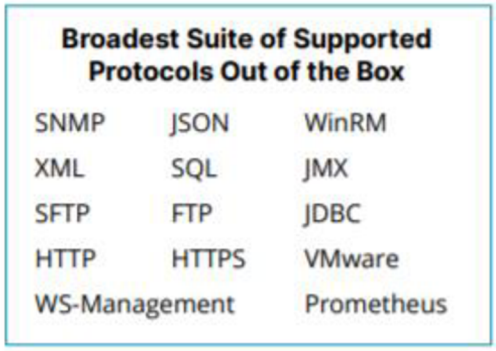
- Free – nabízí téměř všechny funkcionality jako vyšší verze, je však limitován na monitoring pouze 25 hostů
- Standard – plná verze systému; možno dohlížet přes 100.000 hostů
- Managed services – obsahuje vše co plná verze, je však přizpůsoben pro multi uživatelský přístup, vhodný pro firmy, kteří by chtěli provozovat Checkmk jako placenou službu pro ostatní

Jako hlavní přednost Checkmk uvádí svoji automatizovanost a tedy uživatelskou nenáročnost. Jako hlavní funkce jsou tedy vyzdvihovány:

- Přidávání nových prvků a jejich konfigurace je zjednodušeno. Checkmk sám pozná, že se jedná např. o firewall a dle toho vše nastaví automatizovaně a to včetně dohledovaných položek, metrik a prahových hodnot pro spuštění upozornění.
- Automatizace monitorování dynamické, měnící se infrastruktury. Síťové prvky, počítače, servery a ostatní jsou přidávány a odebírány z monitoringu dle skutečnosti a automatizovaně.
- Umožňuje používání moderní konfigurace založenou na pravidlech 1 až N, která zůstává intuitivní i ve složitých prostředích a má za následek snížení úsilí při konfiguraci, než je tomu u jiných monitorovacích řešení.
- Automatizace konfigurace a provoz pomocí rozhraní Checkmk REST-API.
- Centrální správa agentů a automatizace jejich aktualizace pomocí Agent Bakery.
- Možná integrace dalších systémů pomocí výkonných rozhraní API k automatizaci téměř čehokoli, co si lze představit.
- Integrace dat z celé řady zdrojů dat a formátů pro metriky (data JSON, XML, SNMP a další).

3.5.3 OpenNMS

OpenNMS je vyvíjen od roku 2000 a je nabízen ve dvou distribucích – Meridian a Horizon (29). Obě distribuce jsou postaveny na stejných základech a jsou open-source. Vzájemně je odlišuje rozdílný životní cyklus a možnost podpory. Podporované protokoly jsou u obou platforem stejné (viz obrázek níže)



Obrázek 13 - přehled podporovaných protokolů systémem OpenMMS, převzato z (29)

Produkt je vyvíjen v programovacím jazyku Java, instalován může být i na serverech s operačním systémem Windows, což ho odlišuje od ostatních zmiňovaných dohledových systémů. Jako jedinou možnou databázi pro uložení dat využívá PostgreSQL.

OpenNMS Horizon

- Časté aktualizace – měsíční bezpečnostní záplaty, čtvrtletní update funkcí
- Využívání nejnovějších funkcí platformy
- Nabízen zdarma
- Vhodný pro ty, kteří chtějí mít nejnovější funkcionality a neomezuje je potřeba časté aktualizace

OpenNMS Meridian

- Vyšší stabilita než předchozí distribuce
- Bezpečnostní aktualizace jednou měsíčně, aktualizace funkcí jednou ročně
- Placená verze, různé úrovně podpory
- Tato verze je doporučována do produkčního prostředí

OpenNMS je využíván mimo jiné k monitoringu síťových prvků a celkově k monitoringu datového provozu, kde společně s propracovanými grafy je cenným pomocníkem nejen síťových administrátorů.

3.6 Shrnutí

Teoretická část této práce byla věnována způsobům monitoringu firemní infrastruktury. Samotný monitoring se provádí s využitím standardizovaných protokolů, které jsou uzpůsobeny k vytěžování dat z dohledovaných prvků. Některými protokoly lze dokonce i zařízení ovládat čili měnit mu parametry, které pak ovlivňují jeho chod.

Firemní infrastruktura často čítá stovky i tisíce zařízení, která není v lidských silách v reálném čase kontrolovat a reagovat na jejich chybové stavy. Proto je nezbytné, v takových prostředích provozovat dohledový systém, který je schopen sám provádět monitoring a případně reagovat na chybové hlášení automatickými procedurami či jen upozorněním kompetentních osob.

Dohledové systémy jsou využívány nejen pro svoji hlavní činnost jakou je dohled, ale i pro svoji vlastnost strukturálně uložit získaná data, která mohou být pak využita pro jiné potřeby jako je např. evidence majetku, kontrola dodržení SLA a jiné informace, které lze díky nim získat.

V současné době lze těžko prohlásit nějaký z dohledových systémů za „nejlepší“. Každý produkt bude vyhovovat někomu jinému a to ať třeba z důvodu osobních preferencí nebo pro individuální účely, ke kterým má být provozován.

Sledujeme-li vývoj monitorovacích systémů, lze prohlásit, že snahou jejich vývojářů je co nejvíce systém přiblížit k uživatelům a snaha o co největší míru automatizace za účelem zjednodušení celé již takto dost složité problematiky jakým monitorování je. Nutno dodat, že velký pokrok lze zaznamenat i u dohlížených prvků, kdy je patrný důraz na implementaci a bezproblémový chod protokolů, které monitoring umožňují.

4 Vlastní práce

Problematika řešená v teoretické části práce bude ověřena v rámci praktické části na reálném firemním prostředí. Jedná se o 4 pobočky, které jsou geograficky rozmístěny po celém území České republiky. Společnost má několik odděleních, které se zabývají různými činnostmi. Dohledový systém má být nainstalován pro potřeby IT oddělení, které má na správu veškeré IT firmy a to včetně dohledu nad síťovou infrastrukturou firmy. Autor práce je ve zmiňované firmě již několik let zaměstnán a to právě na oddělení IT.
Součástí této práce bude nejenom zprovoznění dohledového systému, ale i provedení vlastních úprav pro potřeby společnosti.
Jelikož popisovaná společnost nechce být blíže specifikována, nebude se autor dále k její charakteristice vyjadřovat. Některé údaje týkající se společnosti budou proto záměrně pozměněny, avšak tyto změny nebudou mít vliv na popsanou implementaci systému v této práci.

4.1 Analýza firemního IT prostředí

Společnost, kde má být zprovozněn dohledový systém, se skládá z několika poboček. Jednotlivé pobočky mají vlastní IT oddělení, které se starají o chod IT prvků v dané lokalitě. Součástí povinnosti těchto zaměstnanců je také dohled nad technologiemi, které společnost provozuje. Jedná se o různé softwarové či hardwarové technologie. Pokud jsou pracovníci IT oddělení nepřítomni, dohled přebírá IT oddělení z pobočky v Brně, které je v provozu non-stop.
Pro potřeby implementace dohledového systému bylo IT prostředí rozděleno do kategorií, u kterých byly zjišťovány možnosti dohledu.

4.1.1 Síťová infrastruktura

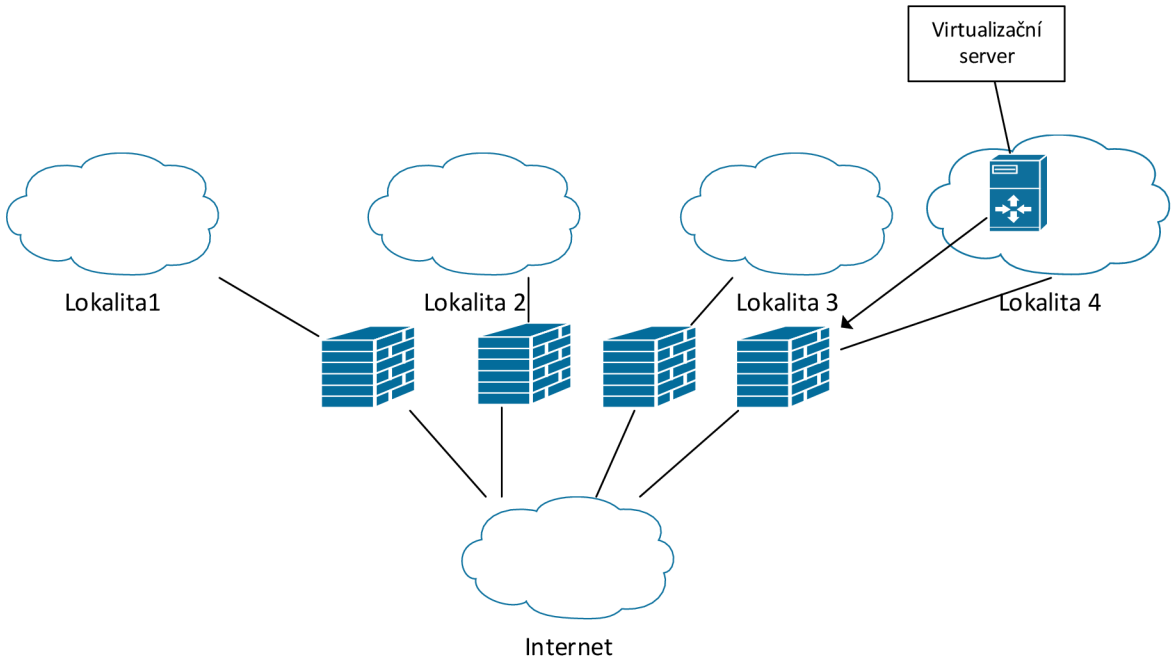
Pobočky jsou mezi sebou propojeny sítí internet, kdy jednotlivé sítě jsou ve společném VPN tunelu, který zajišťuje hraniční router. Samotný rozvod sítě v jednotlivých lokalitách je pak přes soustavu switchů dál rozveden do jednotlivých kanceláří a prostor.

Zjištěné poznatky potřebné pro návrh systému:

- Jednotlivé lokality jsou mezi sebou dostupné
- V jednom případě je politikou firewallu blokován vstup do jedné sítě. Směr ven je však povolen.
- Většina (všechny páteřní) síťových prvků využívá management interface. Lze tedy využít monitoringu pomocí různých protokolů – např. SNMP.

Položky potřebné monitorovat:

- Dostupnosti jednotlivých lokalit
- Aktuální parametry sítí
- Stavy hardwarových komponent – SW, FW



Obrázek 14 - síťová infrastruktura firmy (zdroj: vlastní tvorba)

4.1.2 Koncová zařízení

Zaměstnanci používají osobní počítače či notebooky. Nejčastěji používaným operačním systémem je OS Windows a to v různých verzích. Menší počet zařízení (zejména technologická zařízení) využívá operační systém na bázi Linuxu jako je CentOS či Debian.

Pro společné užití jsou využívány tiskárny, skenery a další zařízení. Je zde patrná snaha o využívání společných zařízeních za využití síťové konektivity.

Zjištěné poznatky potřebné pro návrh systému:

- Každá pobočka má cca. 40 uživatelských počítačů se systémem Windows
- Používají se sdílená zařízení – tiskárny
- Používaný uživatelský hardware je od mnoha výrobců, kteří se mění - nelze tedy spoléhat na proprietární techniky dohledu nabízených jednotlivými výrobci

Položky potřebné monitorovat:

- Hardware koncových stanic a síťových zařízení
- Stavy operačních systémů

4.1.3 Provozované služby

Nezbytné služby zajišťující chod technologie

Jedná se o služby, které jsou nezbytné pro správný chod celé sítě (např. NTP server, DHCP server apod.). Jsou to služby standardizované a běžné.

Služby využívané pro vlastní podnikání

Jedná se o služby, které jsou potřebné pro vlastní činnost firmy. Ať už se jedná o vlastní software či software třetích stran, je potřeba bezvadný chod těchto služeb také monitorovat.

Služby jsou provozovány na serverech. Některé servery jsou pro takovou činnost dedikovány. Potřeba je tedy dohlížet jak hardware, který hostí služby, tak i bezvadný chod služby samotné.
U služeb „nezbytných" se k monitoringu dají využít standardizované metody. U služeb „pro vlastní podnikání" jsou ale některé vyvinuty pro specifické účely firmy a nejsou tedy nijak standardizovány. Navíc činnosti těchto služeb jsou různorodé.

Zjištěné poznatky potřebné pro návrh systému:

- Společnost využívá klasických služeb sítě
- Společnost má také specifické služby a programy
- Nelze předem určit, jakým způsobem bude možno budoucí službu kontrolovat
- Služby využívají i dedikovaný hardware

Položky potřebné monitorovat:

- Standardní síťové služby jako je NTP či DHCP server
- Standardní služby typu databázový server, webový server
- Specifické služby zákazníka – požadované parametry dohledu je potřeba specifikovat pro každou službu zvlášť

4.1.4 Virtualizace

Společnost provozuje vlastní virtuální prostředí na technologii VMware a XenCenter, kde jsou hostovány virtuální servery či stanice. Ve virtualizaci jsou některé stroje připojeny do páteřní sítě tak, že veškerá komunikace směrem ke strojům je na firewallu blokována. Virtuální stroje tak mohou komunikovat pouze směrem ven a mezi sebou.

Z pohledu dohledového systému není dohled virtuální stanice odlišný od dohledu stanice klasické, tedy hostované přímo na hardwaru. Naopak, odpadá potřeba kontrolovat pro každou stanici její hardware.

Zjištěné poznatky potřebné pro návrh systému:

- Společnost využívá virtualizaci stanic
- Z důvodu použití firewallu a potažmo blokace komunikace směrem ke klientovi, bude potřeba pro monitoring aktivní účasti klienta (klient bude sám posílat informace, nebude dotazován)

Položky potřebné monitorovat:

- Instalaci agentů na virtuálních stanicích
- Správný chod virtuální platformy

4.1.5 Ostatní síťové zařízení

Společnost provozuje i další zařízení, která jsou síťově dostupná. Jedná se např. o datová úložiště, teploměry, různá jiná čidla apod.
K monitoringu těchto prvků je potřeba přistoupit k jako kterékoliv jinému síťovému prvku. Pokud má však dostupné API [14]či lze jiným způsobem se zařízením komunikovat, pak je možné ze zařízení získat různé jiné informace, které může být dobré také monitorovat.

Zjištěné poznatky potřebné pro návrh systému:

- Společnost provozuje různá síťová zařízení, jejichž bezproblémový stav a chod je také potřeba monitorovat
- Síťová zařízení mohou podporovat standardizované ale i proprietární možnosti dohledu

Položky potřebné monitorovat:

- Potřeby jsou obdobné jako u koncových stanic
- U každého zařízení potřeba zjistit specifické požadavky na monitoring

4.2 Analýza současného dohledového systému

Dohledový systém není zatím ve společnosti nainstalován. Dohled nad prvky v síti a jinými síťovými zařízeními je řešen pomocí různých menších programů a skriptů, které byly vytvořeny pro tyto specifické požadavky.
Zaměstnanci IT oddělení nemají jiné možnosti dohledu, než je manuální rutinní kontrola, což je nedostatečné. Často se stává, že se o problému dozvídají v době, kdy už problému nelze předejít. Možnosti, jak predikovat problém na základě vývoje nasbíraných hodnot nejsou. Informace o chodu zařízeních totiž nejsou nikde ukládány, a tedy není možné dělat ani zpětné analýzy nad daty.
Současný stav se jeví jako nežádoucí a společnost nutně potřebuje dohledový systém.

4.3 Analýza požadavků na nový dohledový systém

Součástí přípravy implementace dohledového systému musí být důkladné vytěžení zadavatele o jeho požadavky. Přestože autor práce zná firemní prostředí velmi dobře, naplánoval několik schůzek se zaměstnanci a vedením, kde zjišťoval jejich požadavky na systém.

Jednotlivé požadavky byly rozděleny do třech dimenzí - viz tabulka níže.

---

[14] API - Application Programming Interface

CSF - kritické faktory úspěchu			
kritérium	dimenze		
	řízení	technická	podniková
včasné splnění	x		
dodržení ceny implementace	x		
rychlá návratnost vložených prostředků	x		
splnění funkčních požadavků	x	x	
rychlost systému		x	
zvýšení bezpečnosti		x	
spokojenost zákazníků	x		x
přijetí systému zaměstnanci	x		
vysoká dostupnost		x	x
rychlost integrace	x	x	
možnost další rozšiřitelnosti	x	x	
školení	x	x	
monitoring různých zařízení		x	

Tabulka 4 - Požadavky na systém (zdroj: vlastní tvorba)

4.4 Navrhované řešení

Z předešlých analýz a požadavků na nový dohledový systém vyplynuly základní charakteristiky a požadavky, dle kterých bylo rozhodnuto o výběru systému.

4.4.1 Požadavky na systém

- Architektura nového dohledového systému by měla být federativní. Bude využito skutečnosti, že sama společnost je geograficky rozdělena a je potřeba nasbíraná data centralizovat a společně archivovat.
- Do některých sítí je blokován přístup, bude tedy nutno využít i pasivní kontroly
- Společnost klade důraz na dostupnost systému. Kritické části systému by tedy měly být redundantní či lehce vyměnitelné. Chybový stav kterékoliv části systému by neměl ovlivnit chod systému jako celku.

Požadované funkce

- Sběr, uložení a zpětná analýza dat
- Multi uživatelský přístup
- Bezpečnost a přístup na úrovni uživatelských rolí
- Eskalace problémů
- Upozornění na problémy
- Rozšiřitelnost o vlastní způsoby monitoringu – využití modifikovatelného agenta

Základní požadované způsoby monitoringu

- ICMP Ping – bude využit u každého interface, ke kontrole základních parametrů sítě
- IPMI – k získávání dat z Idrac u Dell serverů popř. iLO u HP serverů

- SNMP – získávání informací z různých zařízení pomocí standardu nezávislého na výrobci. Využití trapů při oznámeních vyvolané druhou stranou.
- Vytváření vlastních dotazů vůči specifickým zařízením či aplikacím (možné využití agenta)

4.4.2 Výběr systému

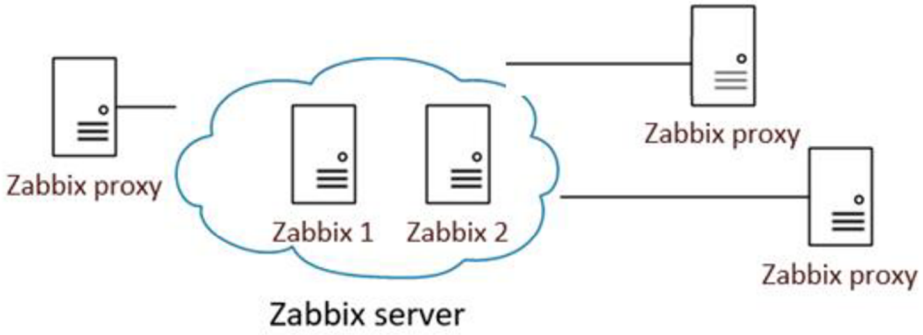
Ze vzniklých požadavků na systém bylo patrné, že nový systém musí být robustní a zároveň upravitelný dle požadavků firmy. Velký důraz byl kladen na možnost systém ve firmě vlastními silami zprovoznit a ten dále udržovat. Podpora ze strany výrobce systému by se měla využívat jen pro řešení problémů, které nelze vlastními silami vyřešit.

Výše uvedené požadavky splňoval Zabbix, který byl také vybrán pro následnou implementaci. Tento systém však není jediný, který by požadavky splňoval, ale autor práce měl se Zabbixem již nějaké zkušenosti, které chtěl využít. Zároveň autor velmi ocenil, že Zabbix má velmi dobře zpracovanou dokumentaci a jeho členská základna je velká a schopná na diskuzních fórech vyřešit případné problémy, což je u takových projektů velkou výhodou.
Nicméně výběr systému na „zelené louce“ je svým způsobem subjektivním rozhodnutím. Pokud systém dokáže splnit kladené požadavky, je to často o osobních preferencích.

4.5 Instalace systému

4.5.1 Topologie systému

Před samotnou instalací systému, byl vypracován návrh systému, který zohledňoval firemní infrastrukturu a zároveň splňoval požadavek na jeho vysokou dostupnost. Uspořádání jednotlivých komponent zobrazuje obrázek viz níže.



Obrázek 15 - plánovaná topologie systému Zabbix (zdroj: vlastní tvorba)

- **Zabbix server** – použity dva fyzické servery (Zabbix 1, Zabbix 2), které využívají vysoké dostupnosti. Jeden server bude vždy aktivní, druhý bude připraven převzít jeho funkci v případě problému či pro potřeby technologické odstávky. Lze tedy říci, že se jedná o Zabbix server cloud, jenž se okolí jeví jako server jediný. Nasbíraná data budou ukládána na těchto serverech a přístupna pro oba dva servery.

- **Zabbix proxy** – z jednotlivých geografických lokalit budou monitorovaná data shromažďována pomocí proxy. Výhodou proxy je cachování dat, které v případě problému s konektivitou se Zabbix serverem umožní data shromáždit na proxy a odeslat je až bude server dostupný. V případě lokality s omezeným přístupem do vnitřní sítě je využito módu pasivní proxy, kdy server komunikuje s proxy. Oproti ostatním proxy, které aktivně komunikují se serverem (aktivní proxy).

4.5.2 Nákup HW

Zabbix nemá vysoké požadavky na hostitelský HW. Na oficiálních stránkách Zabbix.com lze nalézt doporučovanou minimální konfiguraci, jež zobrazuje obrázek níže.

Name	Platform	CPU/Memory	Database	Monitored hosts
Small	CentOS	Virtual Appliance	MySQL InnoDB	100
Medium	CentOS	2 CPU cores/2GB	MySQL InnoDB	500
Large	RedHat Enterprise Linux	4 CPU cores/8GB	RAID10 MySQL InnoDB or PostgreSQL	>1000
Very large	RedHat Enterprise Linux	8 CPU cores/16GB	Fast RAID10 MySQL InnoDB or PostgreSQL	>10000

Obrázek 16 - minimální konfigurace , převzato z (12)

Na základě doporučené minimální konfigurace a vlastních zkušeností byly vybrány tyto sestavy:

Zabbix server			
Název	Cena	ks	Cena
HPE DL160 Gen10 4208 1P 16G 8SFF Svr	44 804	1	44 804
HP Ethernet 1Gb 2P 332T Adptr	4 671	1	4 671
HPE 500W FS Plat Ht Plg LH Pwr Sply Kit	4 081	1	4 081
HPE 600GB SAS 10K SFF SC DS HDD	5 093	6	30 558
			84 114

Zabbix proxy			
Název	Cena	ks	Cena
HPE DL20 Gen10 E-2224 1P 16G 4SFF Svr	32 759	1	32 759
HP Ethernet 1Gb 2P 332T Adptr	4 671	1	4 671
HPE 500W FS Plat Ht Plg LH Pwr Sply Kit	4 081	1	4 081
HPE 600GB SAS 10K SFF SC DS HDD	5 093	3	15 279
			56 790

Tabulka 5 - rozpis jednotlivých sestav (cena v Kč včetně DPH); (zdroj: vlastní tvorba)

Celkové náklady na pořízení hardware byly:

Ks	Položka	Cena
2	server	168 228
4	proxy	227 160
		395 388

Tabulka 6- vyčíslení nákladů na pořízení HW (cena v Kč včetně DPH); (zdroj: vlastní tvorba))

4.5.3 Příprava HW

Nastavení RAID
Aby jednotlivé servery byly imunní vůči výpadku disku, byla na nich nastavena funkcionalita RAID, která umožňuje mj. spojení více fyzických disků do jednoho logického disku. Spojením je možno docílit zrychlení r/w [15]operací, ale i toho, že při poruše jednoho disku je dál logický disk dostupný a nedojde tedy ke ztrátě dat a systém pak může dále fungovat i v degradovaném stavu.
Pro naše účely byly vytvořeny na serverech tyto raidové skupiny.

Zabbix server – 5 x disk (velikost disku 600GB) v RAID 5 a 1 x disk hot spare

- Výsledná velikost logického disku je 2400 GB

Zabbix proxy – 2 x disk v RAID (velikost disku 600GB) 1 a 1 x disk hot spare

- Výsledná velikost logického disku je 1200 GB

Vlastnosti použitých svazků jsou charakterizováni níže.

RAID 1 - je označováno také jako zrcadlení (anglicky „mirroring“). Vyžaduje pro svůj chod sudý počet pevných disků. Data jsou v tomto případě ukládána duplicitně na dva disky současně (stejná data se nachází na obou pevných discích).

Výhodou tohoto řešení je ochrana proti chybám – při poruše jednoho z pevných disků je k dispozici záložní kopie, takže stačí nahradit chybný disk a uživatel nepřichází o svá data. Nevýhodou je fakt, že je využíváno maximálně 50 % skutečné diskové kapacity (v případě rozdílné kapacity použitých disků lze využít velikost odpovídající menšímu ze dvojice disků). Navíc toto diskové pole nenabízí žádné zvýšení výkonu při ukládání dat. (30)

RAID 5 - vyžaduje alespoň 3 členy, přičemž kapacitu jednoho členu zabírají samoopravné kódy (neboli parity), které jsou uloženy na členech střídavě (a ne pouze na jednom, čímž byla odstraněna nevýhoda RAID 4). Výhodou je, že lze

[15] Read/write – čtení/zápis

využít paralelního přístupu k datům, protože delší úsek dat je rozprostřen mezi více disků, takže čtení je rychlejší. Nevýhodou je pomalejší zápis (nutnost výpočtu samoopravného kódu). Je odolný vůči výpadku jednoho disku. (31)

HOT Spare – jedná se o disk, který je připraven automaticky zastoupit vadný disk v poli. Po výměně dojde k rebuildu pole a k obnovení bezvadného stavu. Výhodou je okamžitá obnova poškozeného svazku za cenu nevyužití tohoto disku pro jiné účely.

Tímto nastavením bylo docíleno toho, že chybový stav jednoho disku nebude znamenat pád systému pro nečitelnost dat, ale jen potřebu vadný resp. degradovaný svazek opětovně sestavit. Celý proces je zcela automatizovaný.
Jak server tak i proxy jsou imunní oproti výpadku jednoho disku. Dokonce, po úspěšném sestavení, kdy by byl vadný disk nahrazen spare diskem, může dojít k selhání dalšího disku. Raidové pole bude degradované, ale funkční. Lze tedy říci, že selhání dvou disků neohrozí systém – pokud nedojde k poruše disků současně.

Síťová konektivita
Aby mohly být jak servery tak i proxy monitorovány a umožněna jejich hardwarová vzdálená správa, bylo potřeba zapojit jejich iLO [16]interface do sítě.
První ethernetový port byl použit pro samotný chod systému.
V případě Zabbix Serverů byly použity i další ethernetové porty a to tak, že eth2 byl využit k dedikovanému spoji pro databázovou replikaci a eth3 byl využit k přímému propojení serverů pro potřeby HA. V další části práce budou tyto funkcionality osvětleny.

4.5.4 Příprava operačního systému

Před samotnou instalací Zabbixu, bylo potřeba nainstalovat operační systém. Zabbix je možno provozovat na mnoha linuxových platformách, autor si vybral CentOS 7. Tento operační systém lze stáhnout zdarma z oficiálních stránek na webové adrese www.centos.org.
Po startu systému z USB disku, kde byl uložen obraz s instalací, se spustil grafický průvodce. Ten je velmi intuitivní a jednotlivé parametry lze snadno vložit do předpřipravených formulářů.
V instalaci byla provedena následující konfigurace síťových rozhraní:

	server1	server2	proxy	popis
název	Zabbix1	Zabbix2	proxyN [17]	hostname

[16] iLO – integrated Lights-Out - patentovaná technologie pro správu vestavěných serverů od společnosti Hewlett-Packard Enterprise
[17] proxyN – N označuje číslo, které slouží k logickému rozlišení proxy (proxy1, proxy2 apod.)

eth0	DHCP	DHCP	DHCP	přístup do firemní sítě
eth1				nevyužitá záloha
eth2	10.0.1.1/30	10.0.1.2/30		replikace DB
eth3	10.0.2.1/30	10.0.2.2/30		heartbeat

Tabulka 7 - síťové nastavení (zdroj: vlastní tvorba)

Jak je z tabulky patrné, pouze Zabbix servery budou využívat více rozhraních.

Samotný disk (který je v raidové skupině) nebyl již nijak dál pomocí instalátoru dělen, aby mohlo být efektivně využito veškeré volné místo. Nastavení synchronizace času pomocí NTP (při instalaci položka network time) bylo záměrně necháno v defaultním nastavení – disabled. Jako jedna z posledních položek je výběr software, který má být nainstalován. Autor je zvyklý vybírat možnost „minimal install“, kdy se do systému nainstalují nejzbytnější komponenty nutné pro chod systému. Další součásti je možno doinstalovat v případě potřeby později. Ostatní položky lze nastavit dle osobních preferencí a nemají vliv na chod zprovozňovaného systému Zabbix (např. lokalizace, heslo apod.)
Po instalaci operačního systému byl proveden restart a systém již nastartoval z lokálních disků.
Po přihlášení se byl proveden následující příkaz:

- yum update
  - příkaz provede celkový update systému (po instalaci se jej doporučuje provézt)

a pro instalaci démona *chronyd*, který se stará o synchronizaci času s NTP serverem následující příkazy (s tímto démonem má totiž autor lepší zkušenosti, než s defaultním *ntpd* démonem)

- yum install chrony
  - příkaz nainstaluje démona chronyd
- vi /etc/chrony.conf
  - pomocí editoru vi byl přidán záznam s adresou ntp serveru
- systemctl enable chronyd; service chronyd restart;
  - tyto příkazy zajistí spouštění demona po startu resp. demona ihned spustí

Aby mohl být systém modifikován a rozšířen na zamýšlené využití, byly doinstalovány a nakonfigurovány následující balíčky:

- Samba
  - Jedná se o balíček, který umožní využít přistupovat k datům pomocí SMB protokolu. Jedná se o jednoduchý způsob, jak data na linuxových systémech zpřístupnit klientům windows.

- V tomto případě budou takto sdíleny různé instalační soubory, návody apod. potřebné pro chod Zabbix.
- *yum install samba samba-client* //instalace potřebných balíčků

systemctl start smb.service //spuštění potřebných služeb
systemctl start nmb.service //spuštění potřebných služeb
systemctl enable smb.service //aktivace spouštění po startu
systemctl enable nmb.service //aktivace spouštění po startu
firewall-cmd --permanent --zone=public --add-service=samba
//trvalé přidání pravidla do firewallu
firewall-cmd –reload //restart firewallu s novým nastavením

- Dále byly vytvořeny potřebné adresáře ke sdílení
  - *cd /home; mkdir share; mkdir share/Agent; mkdir share/Udelatka;* //vytvoření adresářů
  - *chown nobody:nobody public/; chmod -R 777 public/*

//nastavení oprávnění

- následujícími příkazy, dojde k povolení k přístupu do adresáře pomocí smb protokolu

setsebool -P samba_export_all_ro=1 samba_export_all_rw=1
getsebool –a | grep samba_export
yum install policycoreutils-python
semanage fcontext –at samba_share_t "/home/share(/.)?"*
restorecon /home/share

- úprava konfiguračního souboru /etc/samba/smb.conf dle obrázku níže

```
[global]
        workgroup = SAMBA
        security = user
        passdb backend = tdbsam
        guest account = nobody
        map to guest = bad user
        printing = cups
        printcap name = cups
        load printers = yes
        cups options = raw


[Agent]
path = /home/Samba/Agent
browseable = yes
guest ok = yes
read only=yes


[Udelatka]
path = /home/Samba/Udelatka
browseable = yes
guest ok = yes
read only=yes
```

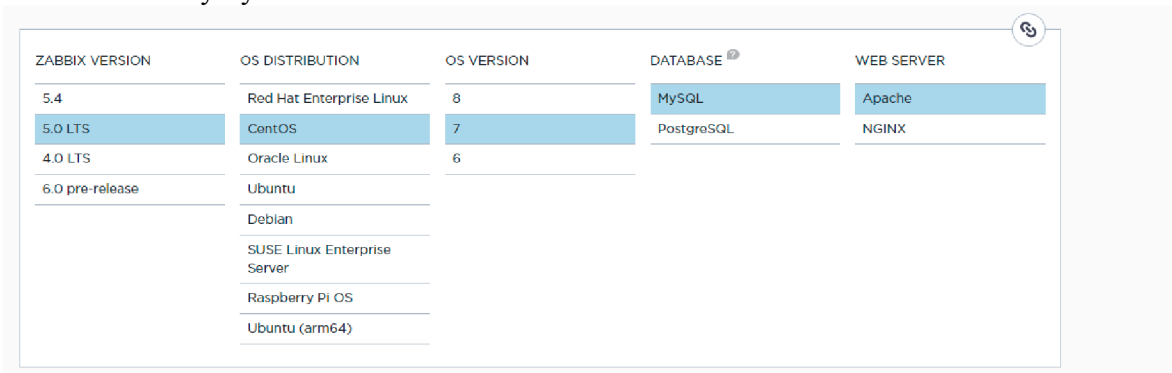
Obrázek 17- zeditovaný konfigurační soubor (zdroj: vlastní tvorba)

Na obrázku je vidět úprava globálního nastavení služby [global] a dále nastavení oprávnění pro jednotlivé cesty

- spuštěním příkazu *systemctl restart smb*.service dojde k restartu služby, která by již měla nabízet nasdílený adresář dostupný na adrese např. \\zabbix1\shared

4.5.5 Instalace Zabbix

Soubory potřebné k instalaci jsou dostupné zdarma, jako tomu bylo u instalovaného operačního systému. Na webových stránkách *www.zabbix.com/download* je dostupný formulář, kde lze přesně specifikovat požadovanou edici Zabbixu. Na obrázku níže je zobrazen zadaný výběr.



Obrázek 18- Výběr požadované verze instalace, převzato z (13)

Nejnovější verze 5.4 nebyla vybrána záměrně. Jelikož se jedná o firemní řešení, kde je kladen důraz na stabilitu, byla zvolena verze LTS[18]. Tato verze se od klasické verze liší delší dobou podpory a více ověřenou stabilitou. Klasické verze rychleji podporují nové funkcionality, což je určitě velká výhoda, ale je to na možný úkor stability. Klasická verze je také častěji pro své funkcionality aktualizována, což se ve firemním prostředí nejeví jako optimální.

Na základě výběru požadované instalace je i na stránkách doporučovaná sada příkazů, která nainstaluje systém Zabbix na hostící server. V našem případě je potřeba tyto kroky udělat na každém ze dvou serverů a všech proxy. Instalační kroky u proxy se mírně liší, bude na to upozorněno. Nejdříve je však nutné nainstalovat samotnou databázi.

1. *wget https://dev.mysql.com/get/mysql80-community-release-el7-3.noarch.rpm*
   - stáhne repositář databáze
2. *rpm -Uvh mysql80-community-release-el7-3.noarch.rpm*
   - dojde k update/instalaci RPM balíčku mysql
3. *install mysql-server*
   - samotná instalace serveru
4. *systemctl enable mysqld; systemctl start mysqld*

[18] LTS – Long Time Support

- těmito příkazy zajistíme spuštění mysql po startu a okamžité spuštění

5. *mysql_secure_installation*
   - příkaz pro prvotní nastavení databáze včetně bezpečnostních politik

Nyní je DB připravena a můžeme pokračovat s instalací Zabbix.

6. *rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/7/x86_64/zabbix-release-5.0-1.el7.noarch.rpm*
   - dojde k update/instalaci RPM balíčku Zabbix
7. *yum install zabbix-server-mysql zabbix-agent*
   - dojde k instalaci Zabbix serveru
   - v případě proxy se zamění slovo *server* za *proxy*
8. *yum install centos-release-scl; yum install zabbix-web-mysql-scl zabbix-apache-conf-scl*
   - instalace nezbytných prerequisit potřebných pro chod webového frontendu
   - u proxy není potřeba
9. *vi /etc/yum.repos.d/zabbix.repo*
   - v souboru zabbix.repo je potřeba zeditovat položku [zabbix-frontend] na enabled=1
   - není potřeba u proxy
10. *mysql -uroot –p*
    - po zadání hesla vytvořeného v kroku 5 se přihlásíme do konzoly databáze
11. *create database zabbix character set utf8 collate utf8_bin;*
    create user zabbix@localhost identified by 'password';
    grant all privileges on zabbix. to zabbix@localhost;*
    quit;
    - Výše uvedenými příkazy vytvoříme novou databázi Zabbix a uživatele s heslem, které si zvolíme. Po vytvoření bezpečnostní politiky se příkazem *quit* vrátíme zpět na příkazovou řádku systému.
12. *zcat /usr/share/doc/zabbix-server-mysql*/create.sql.gz | mysql -uzabbix -p Zabbix*
    - tímto dojde k importu schématu DB

Výše uvedenými kroky došlo k instalaci Zabbix. Nyní je potřeba upravit konfigurační soubory dle skutečnosti.

13. *vi /etc/zabbix/zabbix_server.conf*
    - upravit skutečné heslo v položce DBPassword=password
14. *vi /etc/opt/rh/rh-php72/php-fpm.d/zabbix.conf*
    - vložit položku *php_value[date.timezone] = Europe/Prague*

Nakonec příkazy, kterými se služby spustí a zajistí se jejich spuštění po startu systému.

15. *systemctl restart zabbix-server*
16. *zabbix-agent httpd rh-php72-php-fpm*
    systemctl enable zabbix-server zabbix-agent httpd rh-php72-php-fpm
    - v případě proxy se použije - *systemctl restart Zabbix_proxy; systemctl enable Zabbix_proxy*

Následně je vhodné bezproblémový chod služeb otestovat

17. *service zabbix-server status*
    - v případě proxy: *service zabbix_proxy status*

Nyní jsou jak proxy tak i Zabbix servery nainstalovány a připraveny plnit svoji základní funkci. Pro účely zajištění vysoké dostupnosti celého systému a to zejména serverů Zabbix, bude ještě základní konfigurace modifikována pro zajištění těchto potřeb.

4.5.6 Zajištění vysoké dostupnosti

Vysoká dostupnost (HA[19]) bude zajištěna na Zabbix serverech pro klíčové procesy, které jsou nezbytné pro chod systému Zabbix.
Před instalací služeb zodpovědných za chod HA je potřeba vypnout na serverech proces zabbix-server příkazem
systemctl stop zabbix-server a provézt následující příkazy (na každém serveru)

Instalace služby zajišťující HA a přidání procesů
Pro zabezpečení chodu funkcionality HA byla vybrána služba corosync. Pro komunikaci s clusterem pomocí shellu byl použit balíček pacemaker.

- Úprava současného nastavení

Protože k přístupu zabbix clusteru bude používána virtuální adresa, je potřeba upravit proces apache, zabbix_server a zabbix_proxy, aby se spojení navazovala přes tuto novou adresu.

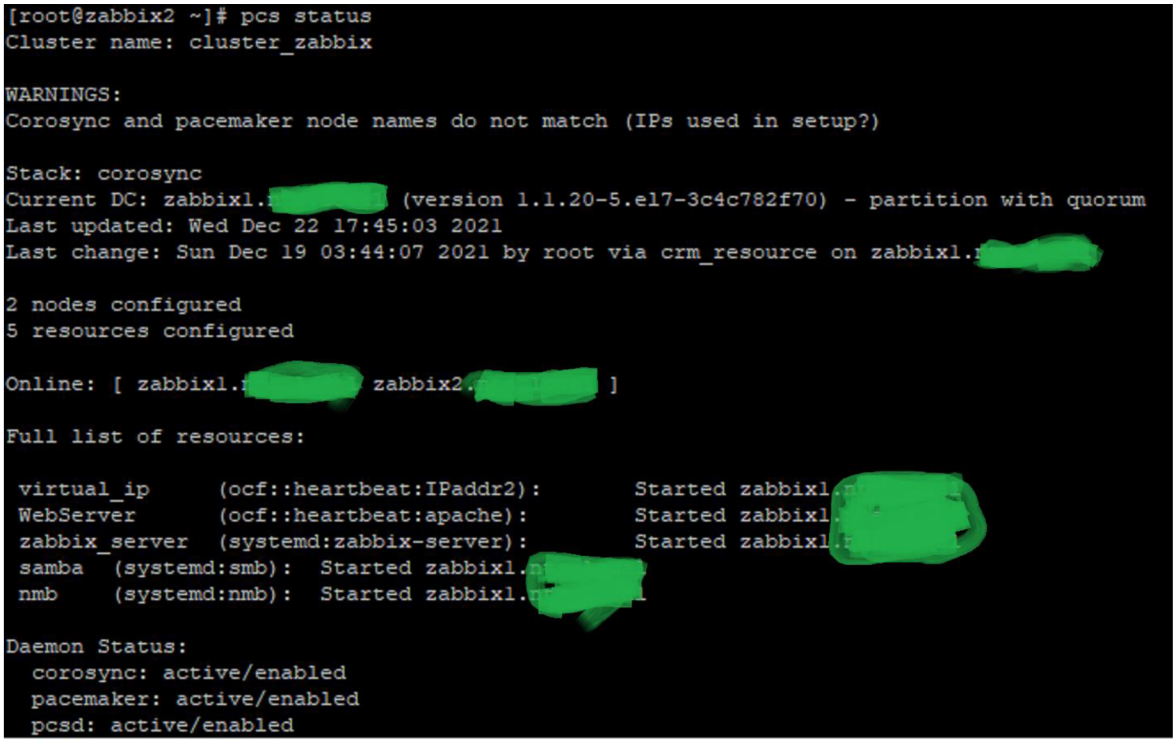
- *vi /etc/zabbix/zabbix_server.conf* //změnit položku „SourceIP“ na adresu 192.168.1.100
- *vi /etc/httpd/conf/httpd.conf* //přidat položku „Listen“ 192.168.1.100:80
- *vi /etc/zabbix/zabbix_proxy.conf* //přidat položku „Server=192.168.1.100“

- Instalace potřebných komponent
- *yum install pacemaker pcs* //instalace
- *passwd hacluster* //vytvoření hesla nově vytvořenému uživateli (stejné pro oba servery)
- *systemctl start pcsd.service* //spuštění služby
- *systemctl enable pcsd.service* //spouštění služby po startu
- *firewall-cmd --permanent --add-service=high-availability* //přidá pravidlo do firewallu
- *firewall-cmd –reload* //vynutí načtení pravidel firewallu

- Konfigurace

[19] HA – High Availability

- *pcs cluster auth zabbix1 zabbix2* //výstupem je dotaz na username, zde vložíme hacluster
- *pcs cluster setup --name zabbix_server zabbix1 zabbix2* //vytvoření clusteru
- *pcs cluster start –all* //zapnutí clusteru
- *systemctl enable corosync.service* //zapnutí služby po startu
- *systemctl enable pacemaker.*service //zapnutí služby po startu
- *pcs property set stonith-enabled=false* //v případě dvou nodů, není potřeba
- *pcs property set no-quorum-policy=ignore* //v případě dvou nodů, není potřeba
- *pcs resource create zabbix_server systemd:zabbix-server op monitor interval=10s* //vytvoření položky (zdroje) zabbix_server, který bude v intervalech kontrolován, zda je spuštěn
- *pcs resource create virtual_ip ocf:heartbeat:IPaddr2 ip=192.168.1.100 cidr_netmask=24 op monitor interval=20s* //vytvoření virtuálního interface, kde bude pod zadanou virtuální adresou přístupné webové rozhraní Zabbbix
- *pcs resource create WebServer ocf:heartbeat:apache configfile=/etc/httpd/conf/httpd.conf statusurl="http://127.0.0.1/server-status" op monitor interval=20s* //přidání služby apache do clusteru
- *pcs resource create samba systemd:smb op monitor interval=60s* //přidání služby smb
- *pcs resource create nmb systemd:nmb op monitor interval=60s* //přidání služby nmb – potřeba pro chod samba služby
- *pcs constraint colocation add zabbix_server virtual_*ip //tímto se zajistí, že zdroj Zabbix_server a virtual_ip budou spouštěny společně, tedy na stejném serveru
- *pcs constraint order virtual_ip then zabbix_server* //pořadí v kterém budou zdroje spouštěny
- *pcs constraint colocation add WebServer virtual_ip*
- *pcs constraint order virtual_ip then WebServer*
- *pcs constraint colocation add samba virtual_ip*
- *pcs constraint colocation add nmb samba*



Obrázek 19- výstup z příkazu pcs status (zdraj: vlastní tvorba)

Obrázek výše zobrazuje výstup z příkazu *pcs status*. Tímto příkazem lze zjistit, v jakém stavu se monitorované služby cloudu nacházejí. Zde je vidět, že cluster jménem cluster_zabbix má celkem 2 nody, které jsou oba online. Cluster hostuje 5 zdrojů, které aktuálně běží na serveru zabbix1.
Pokud tedy dojde k problémům s výše uvedenými procesy, budou automaticky pomocí HA funkcionality spuštěny na druhém serveru.

Konfigurace mirroringu databáze

V předchozí části byla na Zabbix serverech zprovozněna vysoká dostupnost klíčových služeb včetně procesu zabbix_server. Nyní je tedy potřeba zařídit, aby databáze, se kterou tento proces pracuje, byla vždy jemu dostupná a obsahovala aktuální data. Zároveň je však potřeba zajistit, aby při možné poruše databázového severu mysql, nedošlo k nedostupnosti dat pro proces zabbix_server. Jako možné řešení, zvolil autor replikaci typu master x master, kdy jsou data udržována synchronně na všech databází a z kterékoliv lze data číst či dokonce do ní i zapisovat.
Bohužel, u replikace master x master je mnohdy složité udržet konzistenci dat a to zejména v případech, kdy dochází k problémům při replikaci a data jsou měněna v obou databázích současně. Tento problém se většinou řeší přidáním třetího replikačního serveru, který bývá označován jako witness[20], což si lze představit jako jakéhosi arbitra, který zjišťuje stavy databázových enginů a na základě toho se rozhoduje, kam se data budou ukládat. V této práci bude vycházeno z toho, že je požadavek, aby databázi hostily Zabbix servery, které jsou jen dva. Dále se bude

[20] Witness – ang. překlad je svědek

vycházet z toho, že v databázi bude měnit data jen proces zabbix_server, který z principu fungování HA nebude spuštěn na obou strojích současně.

Před samotnou přípravou funkcionality, je potřeba vypnout proces zabbix_server na obou serverech. Jelikož je však již proces spravován službou corosync, je potřeba pro jeho vypnutí využít peacemaker. Autor vypl všechny prostředky cloudu příkazem *pcs cluster standby –all*, čímž se všechny nody přepnou do stavu offline – maintance mode a jejich resources jsou vypnuty.
Pro zajištění konzistence dat, byla data z databáze zabbix na serveru zabbix1 vyexportována příkazem dump *mysqldump -u root -p zabbix > /var/zabbix.sql* a zkopírována na zabbix2 , kde byla naimportována do DB příkazem *mysqldump -u root -p zabbix < /var/zabbix.sql.*
Dále je potřeba vypnout proces mysql příkazem *service mysql stop.*

Konfigurační soubory mysql procesu jsou uloženy v souboru v cestě /etc/my.cnf. Tento soubor je potřeba zeditovat dle obrázku níže.

```
[mysqld]

# master x master
server-id = 1
binlog-do-db = zabbix
log-bin = /var/lib/mysql/zabbix
expire_logs_days=30
slave-skip-errors=1062

# zajisteni spravneho chodu se Zabbix
collation_server = utf8mb4_unicode_ci
character_set_server = utf8mb4
innodb_file_format = barracuda
innodb_file_per_table = 1
innodb_large_prefix = 1
max_connections=1000
innodb_buffer_pool_size = 3968M

datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock

# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0

# Recommended in standard MySQL setup
sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

[client]
default-character-set = utf8mb4

[mysqld]


[mysql]
default-character-set = utf8mb4
```

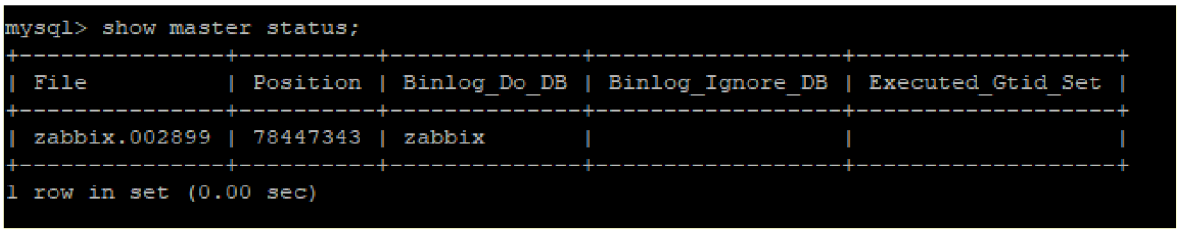
Obrázek 20 - soubor my.cnf (zdroj: vlastní tvorba)

Na obrázku jsou vidět komentáře, které upozorňují na nastavení, které se týká samotné replikace. Položka „server-id“ je identifikátorem nodu databáze, je tedy potřeba na nodu zabbix2 tuto položku změnit na např. *server_id = 2*.
Při replikaci a zejména při zátěžových testech funkcionality HA docházelo občas k chybám no. 1062, což odpovídá pokusu uložit duplicitní záznam. Jelikož k duplicitnímu uložení nedojde a replikace se zastaví, usoudil autor, že bude lepší tuto chybu ignorovat.
Druhý komentář upozorňuje na nastavení, které bylo přidáno na základě zkušeností s využitím databáze procesem zabbix_server.

Nyní jsou databáze nakonfigurovány pro potřeby replikace a je možno tedy nastavení provézt i přímo v databázi.

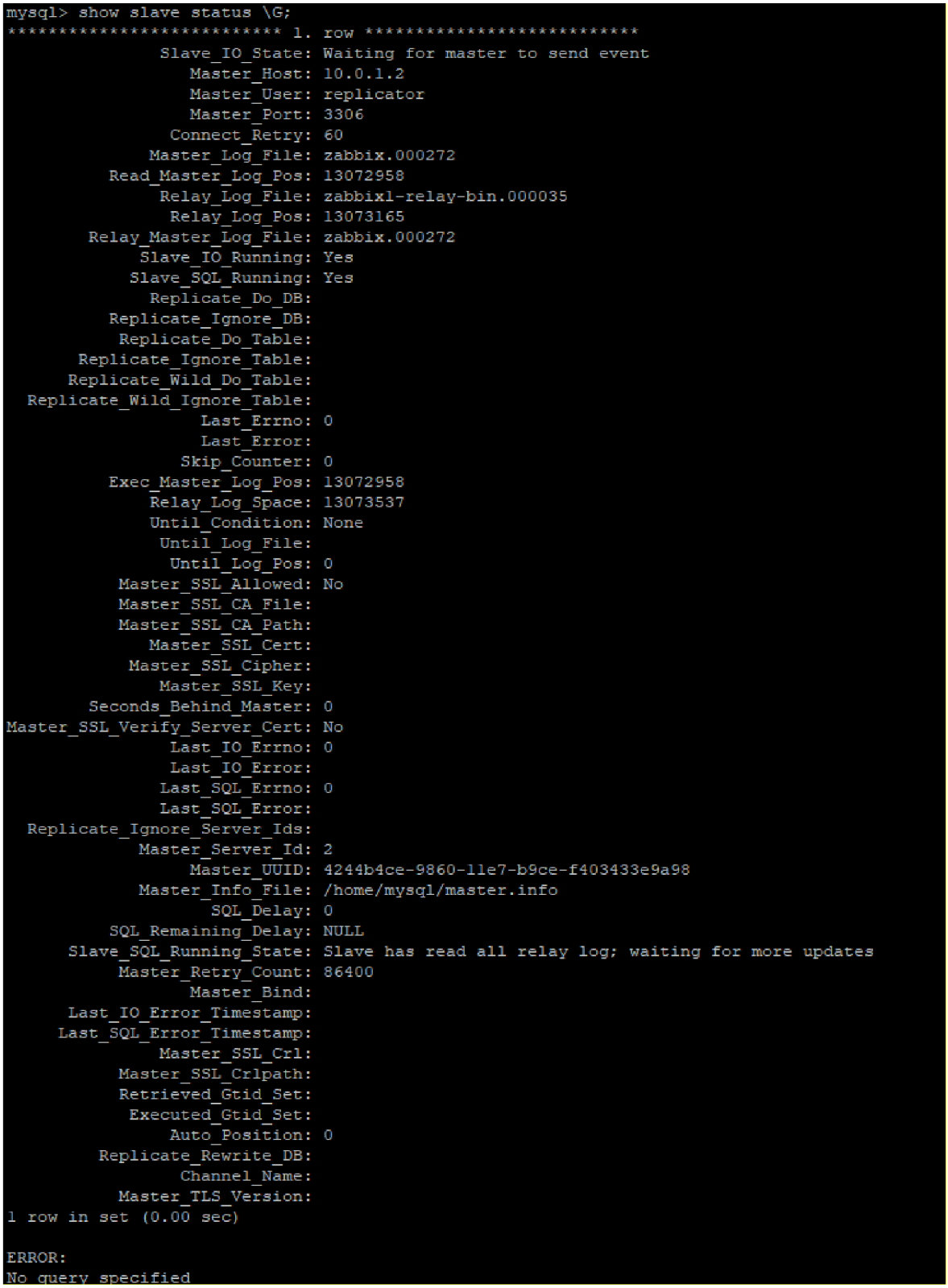
- Zabbix1
    1. *service mysql start* //opětovné spuštění služby
    2. *mysql -u root –p* //přihlášení se do DB
    3. *slave stop;* //vypnutí replikace (pro jistotu)
    4. *create user 'replicator'@'%' identified by 'heslo';* //vytvoření uživatele s heslem pro replikaci
    5. *grant replication slave on *.* to 'replicator'@'%';* //povolení replikace
    6. *change master to master_host = '10.0.1.2', master_user = 'replicator', master_password = 'heslo', master_log_file = 'údaj z bodu 13', master_log_pos = údaj z bodu 13;*
    7. *slave start;*
    8. *show master status;*
    9. *slave start;*
- Zabbix2
    10. *service mysql start //opětovné spuštění služby*
    11. *mysql -u root –p*
    12. *create user 'replicator'@'%' identified by 'heslo';*
    13. *grant replication slave on *.* to 'replicator'@'%';*
    14. *show master status;*
    15. *slave stop;*
    16. *change master to master_host = '10.0.1.1', master_user = 'replicator', master_password = 'heslo', master_log_file = 'údaj z bodu 8', master_log_pos = 'údaj z bodu 8';*
    17. *slave start;*

Na obrázku níže je ukázka výstupu z příkazu show master status, což vrací informace ke stavu master databázi.



Obrázek 21 - zobrazení příkazu show master status (zdroj: vlastní tvorba)

Obrázek níže ukazuje, co vrací příkaz ke zjištění stavu slave databáze. Zde je vidět, že databáze je úspěšně sesynchronizována s master databází a čeká na vznik další události.



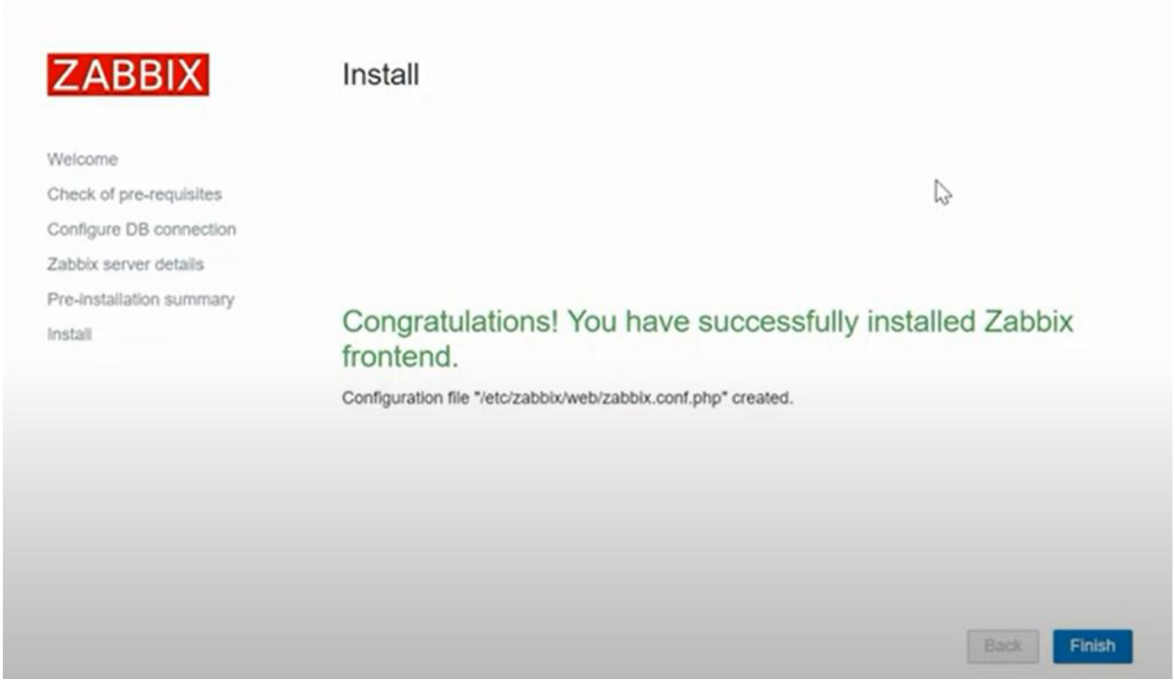
Obrázek 22 - výpis příkazu show slave status \G (zdroj: vlastní tvorba)

Nyní je zajištěna replikace databází, zapíšou-li se data do kterékoliv databáze, dojde k replikaci dat do databáze druhé.

4.5.7 Závěrečná konfigurace Zabbix

Před poslední konfigurací, která je prováděna pomocí webového rozhraní, autor doporučuje provézt restart obou zabbix serverů a následně zkontrolovat, zda jsou potřebné procesy automaticky spouštěny a zejména zda bezproblémově funguje HA a replikace DB. Velkým pomocníkem jsou i logy služeb či systému, které často zaznamenají problém, který není jednoduché jinak detekovat.

Webové rozhraní zabbix je dostupné z aktualizovaného webového prohlížeče, který splňuje nároky na bezpečnost, na adrese http://192.168.1.100/zabbix. Zobrazí se webová stránka, kde je instalační průvodce, který provede jednotlivými kroky instalace. Součástí instalace je i kontrola php komponent. Pokud byl postup dodržen, měla by se po vyplnění požadovaných údajů objevit tato finální stránka.

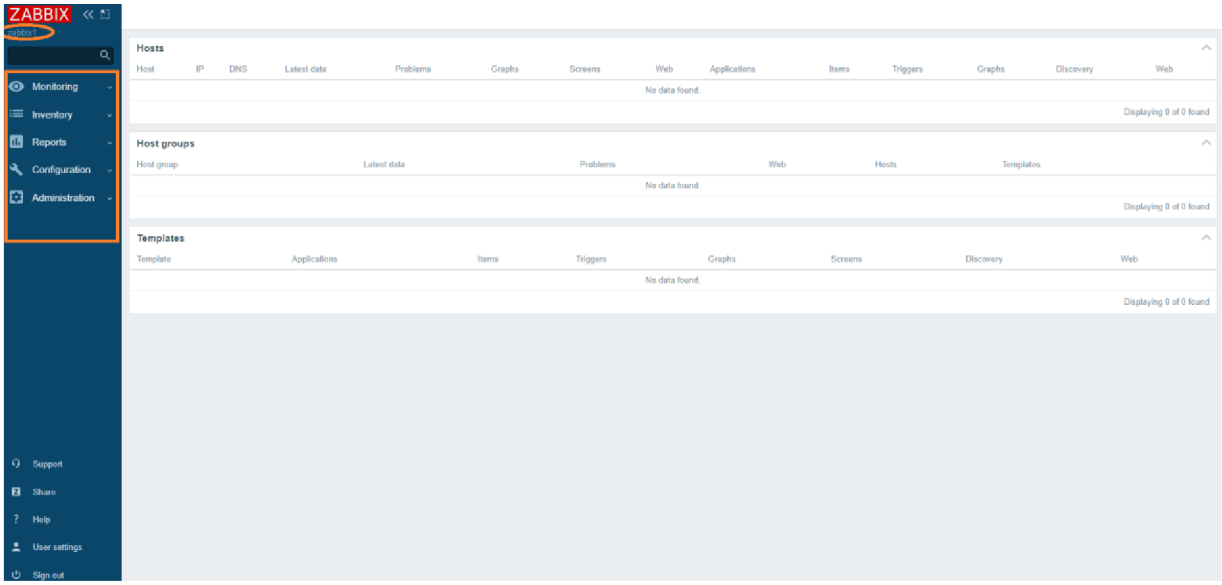


Obrázek 23 - okno zobrazující úspěšnou instalaci (zdroj: vlastní tvorba)

4.6 Obsluha systému Zabbix

V předchozí části byl přiblížen postup, kterým za použití vlastních úprav došlo k instalaci systému a k jeho zprovoznění. Nyní je Zabbix tedy nainstalován a nejsou v něm žádná uživatelská data.
Není záměrem této práce popisovat veškeré funkcionality systému Zabbix, ale přiblížit, jak je systém koncipován a ukázat, jak lze jeho funkcionality rozšiřovat o vlastní.

Pro prvotní přihlášení je potřeba použít defaultní účet *Admin* s heslem *zabbix*. Zobrazí se stránka podobná této.

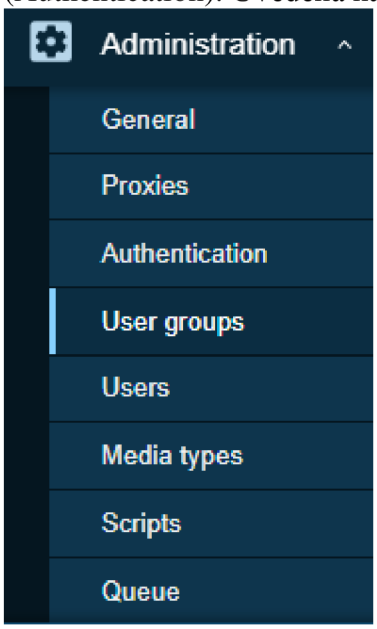


Obrázek 24 - Prvotní přihlášení do Zabbix (zdroj: vlastní tvorba)

Na obrázku výše bylo vyznačeno jméno hostitelského serveru – tato informace je využitelná zejména při použití funkce HA k jednoduchému ověření, který node je nyní aktivní. V našem případu by to měl být právě zabbix1.
Základním rozcestníkem pro ovládání je menu, které je umístěno po levé straně stránky. Na obrázku jsou zobrazeny všechny možné položky, které vidí jen uživatelé s nejvyšším oprávněním.

4.6.1 Uživatelé

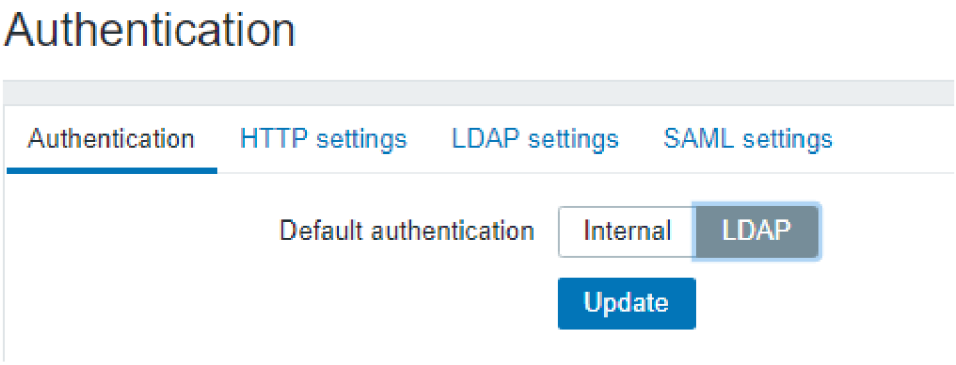
Každý uživatel systému (Users) musí mít vytvořený uživatelský účet, který je přidán do uživatelské skupiny. U každé uživatelské skupiny (User groups) je pak možnost nastavit oprávnění pro její uživatele a zvolit způsob přihlášení respektive autentizaci (Authentication). Uvedená nastavení jsou přístupná v záložce Administration.



Obrázek 25 - záložka "Administration" (zdroj: vlastní tvorba)

Autentizace (Authentication)

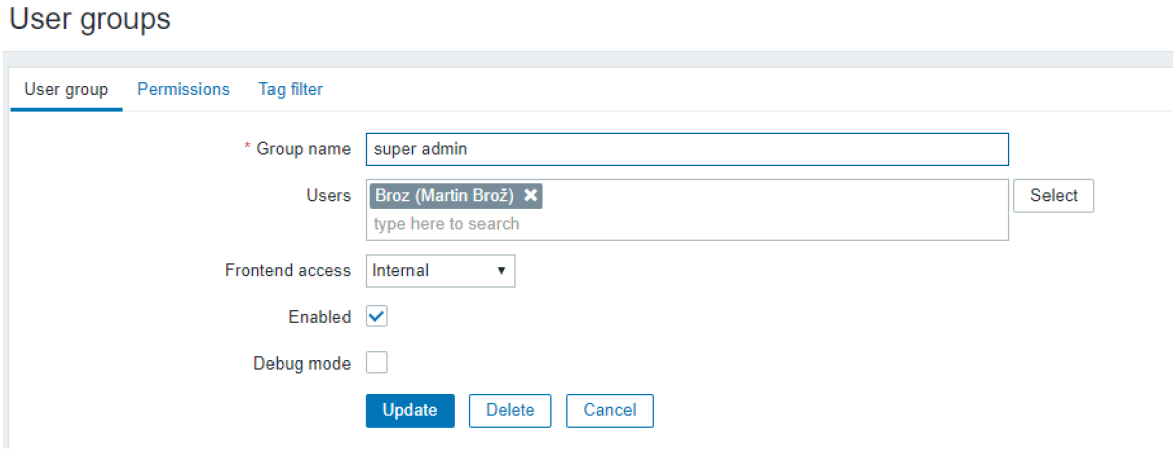
Defaultně je používána interní autentizace, kdy každý uživatelský účet má i vlastní heslo. Ve firemním prostředí, kde byla implementace prováděna, je do různých systémů používána autentizace pomocí LDAP, která se ověřuje vůči Microsoft Active Directory. Z těchto důvodů a pro praktičnost (zaměstnanec si pamatuje jen jedno heslo) byla defaultní politika změněna na LDAP. Dále bylo potřeba vyplnit požadované údaje v položce *LDAP settings.*



Obrázek 26 – nastavení autentizace (zdroj: vlastní tvorba)

Uživatelské skupiny (User groups)

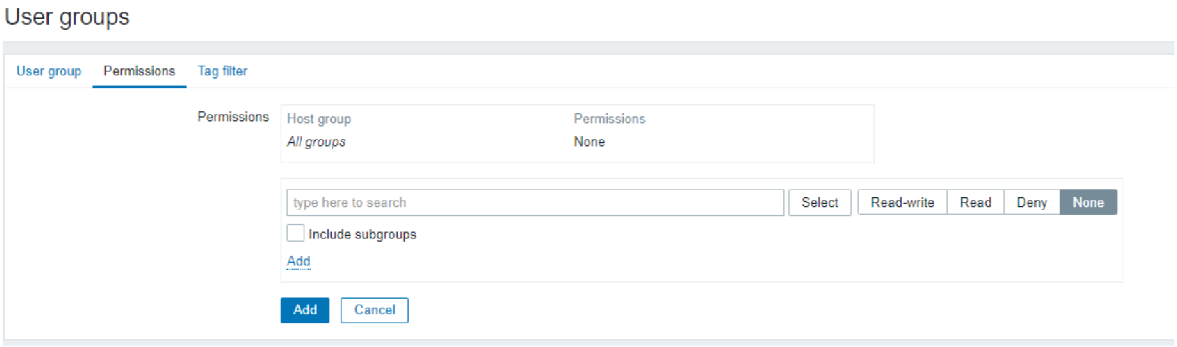
Jak již bylo napsáno dříve, uživatelské skupiny lze chápat jako množiny uživatelů, na která jsou aplikována pravidla (Permissions) omezující jejich přístup na vybrané skupiny dohlížených strojů (Hosts). Nad skupinami se také definuje, jak se uživatelé, kteří patří do dané skupiny, budou přihlašovat (viz Autentizace)



Obrázek 27 - vytvoření uživatelské skupiny (zdroj: vlastní tvorba)

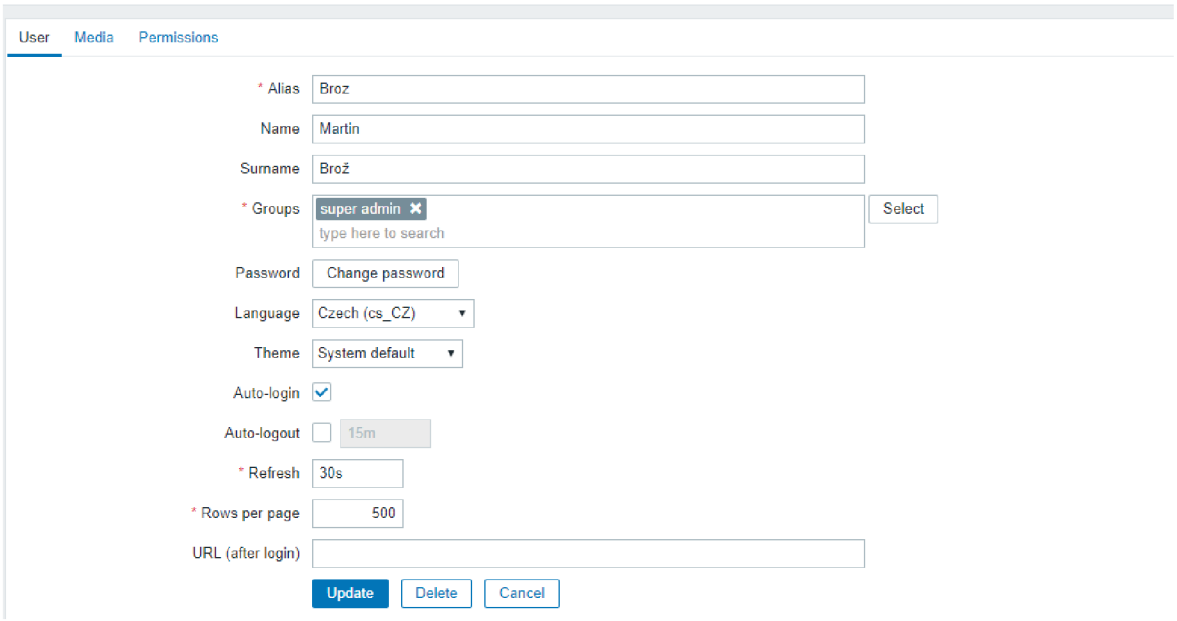
Na obrázku výše je znázorněno, jak autor vytvořil skupinu *super admin,* ve které použil interní autentizaci. Následně do ní přidal svůj uživatelský účet. *Použití vnitřní autentizace je vhodné u administrátorů a to pro potřeby se přihlásit i v případě problémů s LDAP ověřením.*

V záložce Permissions lze explicitně nastavit, oprávnění na skupiny hostů. V případě vytvářené skupiny super admin se automaticky zobrazí pravidlo All groups:None, což znamená, že nebudou pravidla aplikována. Uživatel Broz je totiž role super admin, což mu umožní přístup ke všem funkcionalitám zabbix.



Obrázek 28 - úprava oprvnění pro skupinu (zdroj: vlastní tvorba)

Uživatelé (Users)



Obrázek 29 - vytvoření uživatele (zdroj: vlastní tvorba)

Toto nastavení slouží k vytvoření účtu uživatele a k úpravám jeho profilu. Záložka *Media*, slouží k přidání media, které může pak uživatel využívat k informování se o problému (typicky se jedná o email). Další záložka *Permissions* slouží k zobrazení všech oprávnění na *Hosty* pro daného uživatele. Lze zde měnit také uživatelská úroveň (role) a to na Zabbix User/Admin/Super Admin.

- User – má oprávnění pouze zobrazit menu „Monitoring“ a jsou mu zpřístupněny jen explicitně nastavené skupiny hostů z User groups, kam patří.
- Admin – stejné oprávnění jako User + oprávnění konfigurovat zpřístupněné skupiny
- Super Admin – má oprávnění na všechny prostředky systému.

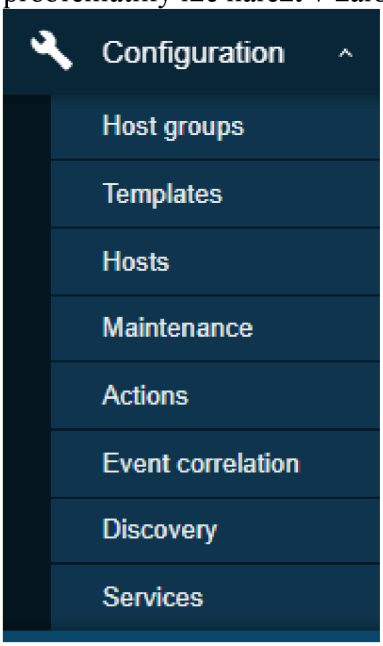
Přestože se použije ověřování pomocí LDAP, je potřeba, aby každý uživatel měl založený svůj účet v Zabbix. Jinak mu nebude umožněno se přihlásit.

Aplikace v praxi

V prostředí, kde byl systém aplikován, byly vytvořeny uživatelské skupiny, které byly pojmenovány dle lokalit. Jejich ověřování bylo ponecháno na defaultním nastavení čili pomocí LDAP.
Dále byli vytvořeni uživatelé a v záložce *Permissions* jim bylo přiřazeno oprávnění na úrovni *user* či *admin*.
Do skupiny *super admin* byl k uživateli Brož přidán další uživatel (administrátor), což je vhodné udělat a to z důvodu potřeby zastupitelnosti. Každý systém by měl mít právě minimálně dva uživatele, kteří mají nejvyšší oprávnění.
Uživatelům nebyly prozatím přiřazeny skupiny hostů (nebyly doposud vytvořeny).

4.6.2 Monitoring

V předchozí podkapitole došlo k vytvoření uživatelů v systému a přidání jejich rolí. Dalším krokem je zavedení monitorovaných položek. Nastavení týkající se této problematiky lze nalézt v záložce Configuration.



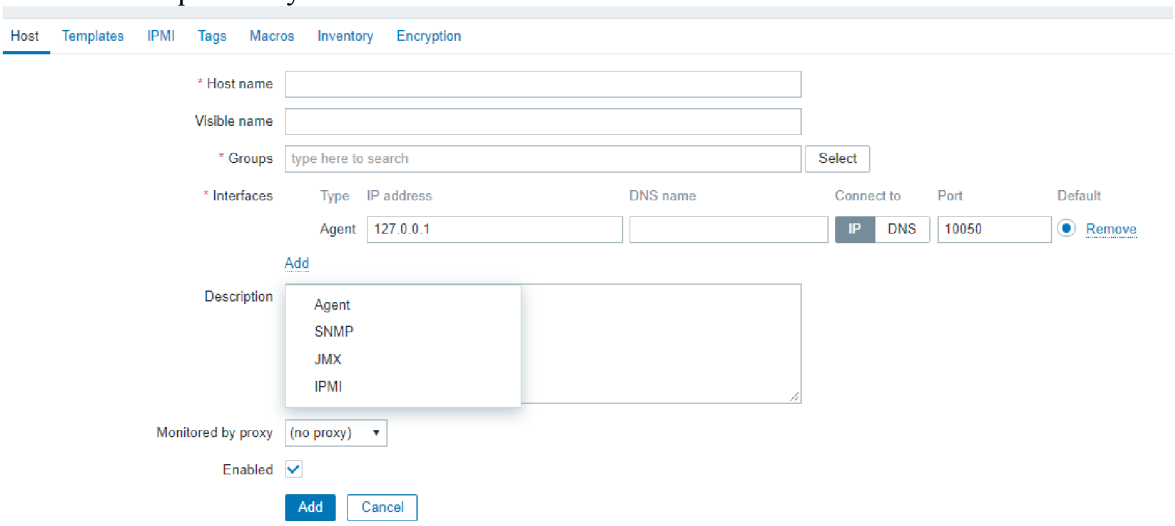
Obrázek 30 - záložka "Configuration" (zdroj: vlastní tvorba)

Skupiny hostů (Host groups)
Zde je patrná analogie se skupinami uživatelů viz výše. Jedná se tedy o množinu hostů, do které lze vkládat jednotlivé hosty na základě nějaké příslušnosti. Samotné skupiny nemají žádný speciální význam. Slouží pro zvýšení přehlednosti a lepší organizovanost hostů. Na druhou stranu jsou využívány na aplikaci politik, které lze s výhodou aplikovat nad celými skupinami.

Hosté (Hosts)
Každá monitorovaná položka musí patřit nějakému *hostovi*. Hosta lze tedy chápat jako zařízení, které má vlastní síťové rozhraní, se kterým může Zabbix komunikovat a získávat od něj informace (typicky se jedná o server).

K vytvoření hosta je potřeba zadat minimálně jeho jméno, skupinu, do které bude patřit, a jeho interface. U interface je potřeba zadat také komunikační protokol a adresu či DNS název, kde bude host k dosažení. Pokud daný host bude monitorován pomocí zabbix proxy, je nutné nastavit i proxy, která se bude o komunikaci starat.
Je nutné si uvědomit, že všechny protokoly, které budou uplatňovány pro monitoring jednotlivých položek, zde musí být přidány jako jednotlivé interfaces.
Nelze vytvořit hosta bez přidáním interfaces. Pokud nabízený protokol nevyhovuje, doporučuji použít protokol „agent“. V případě monitoringu za pomocí nenabízeného protokolu Zabbix použije právě tento- např. ověřování pomocí icmp dotazů (ping) apod.

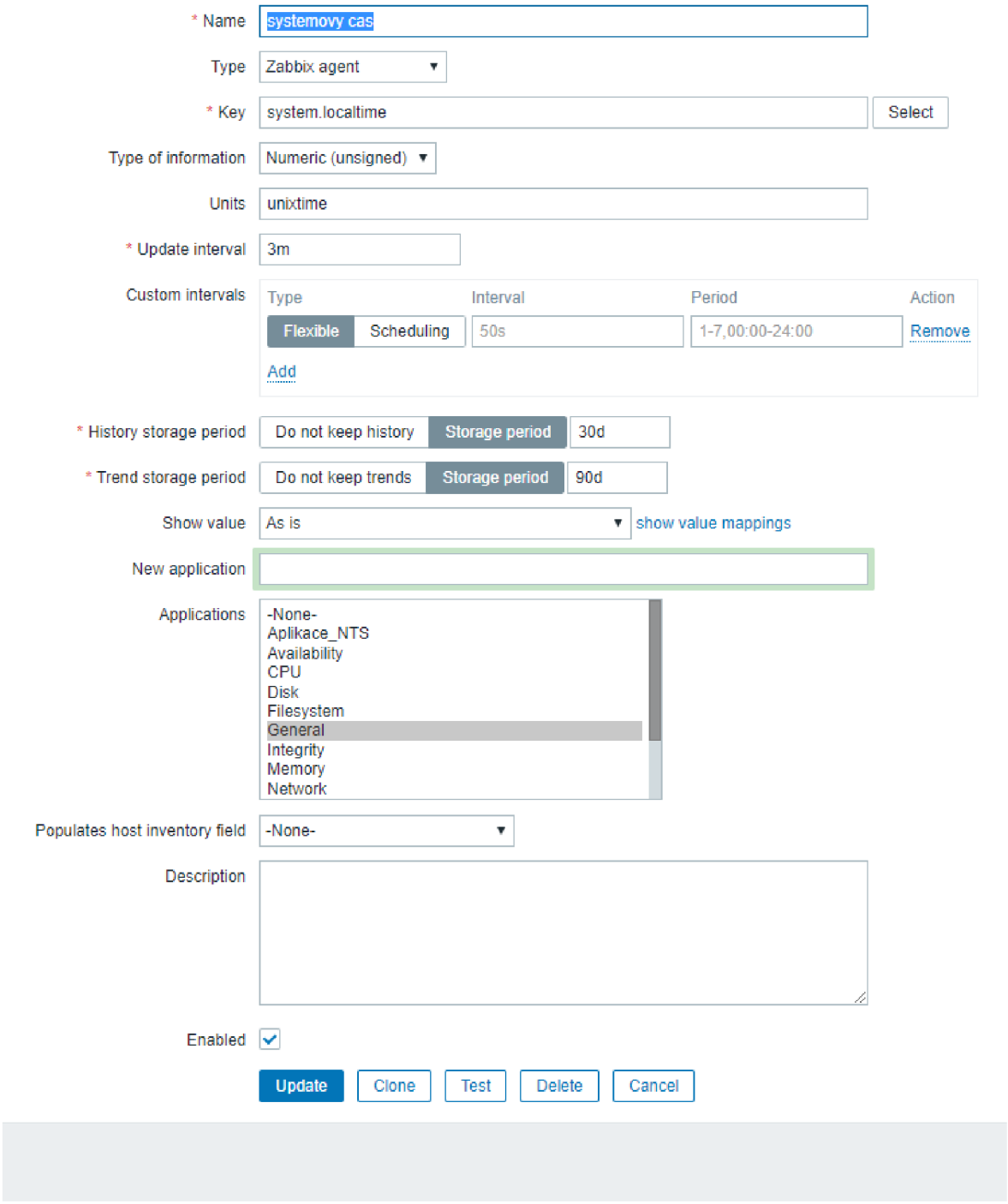
Na obrázku níže je zobrazen formulář pro zadání hosta. Zároveň je zobrazeno, jaké komunikační protokoly Zabbix nabízí.



Obrázek 31 - konfigurace hosta (zdroj: vlastní tvorba)

Položky (Items)
Každý údaj, který má být monitorován, je k *hostovi* přidáván jako položka. Možnost konfigurace položky je závislá na použitém protokolu, který byl k hostovi přidán (viz Hosté).
Protože správné nastavení položek a pochopení jejich významu je klíčové pro práci se systémem Zabbix, bude zde jejich nastavení blíže rozebráno.

Konfigurace položky může vypadat následovně:



Obrázek 32 - konfigurace položky (zdroj: vlastní tvorba)

- *Name* – jméno položky; je dobré, aby z jeho názvu bylo jasné, co položka monitoruje
- *Type* – způsob, jak bude položka sbírána resp. jakou metodou (SNMP, zabbix agent apod.)
- *Key* – samotný dotaz, který je vykonán. Z obrázku je patrné, že je dotaz vykonán na cílovém hostu pomocí zabbix agenta, který vykoná příkaz system.localtime. Což je přednastavený příkaz pro získání času.

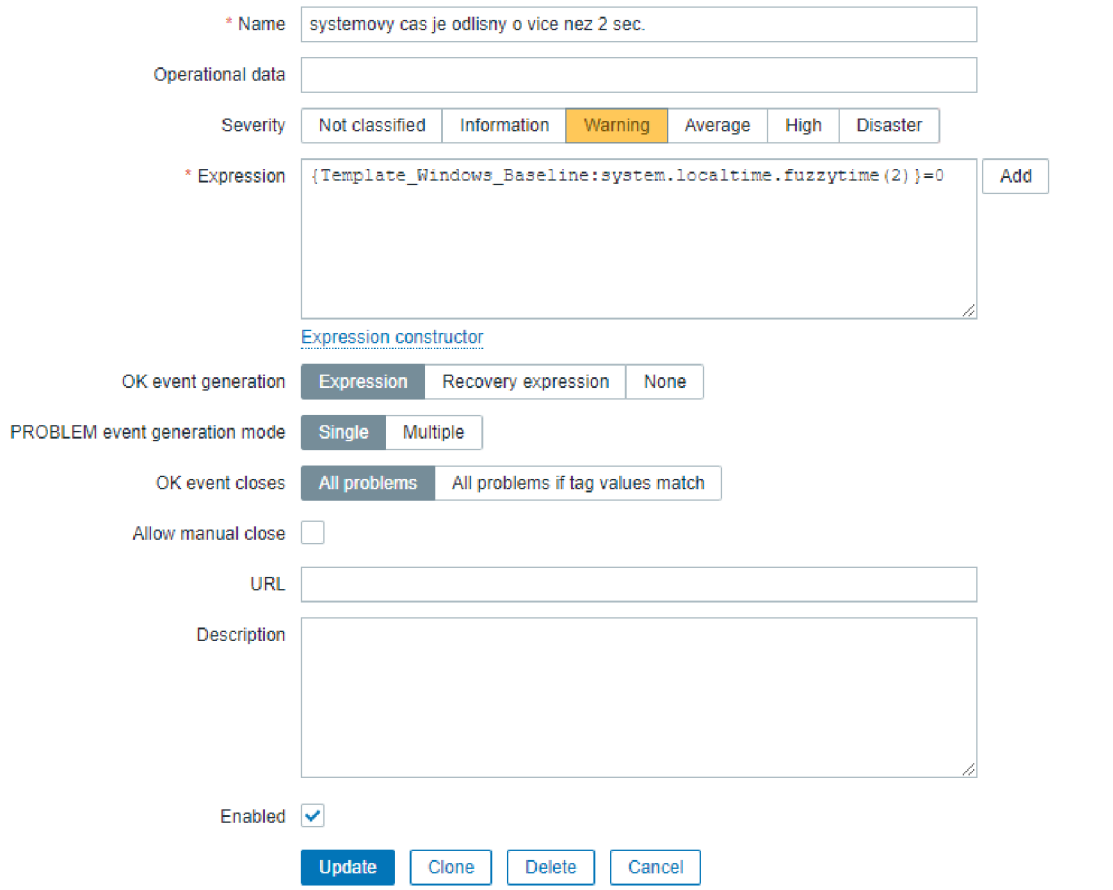
- *Type of information* – zde se upřesňuje typ návratové hodnoty. Jelikož je vracená hodnota u system.localtime číselné vyjádření unix time epoch[21], je možno nechat typ na Numeric (unsigned). Zde je potřeba upozornit, že špatně zvolený typ může vézt k chybě při konvertování, což způsobí nefunkčnost celé položky.
- *Units* – jednotky vrácené hodnoty. V našem případě se jedná o unixtime. Zabbix bude vědět, že při zobrazování těchto hodnot, lze hodnotu převézt na čas, což usnadní prohlížení nasbíraných hodnot. Pokud jednotku neuvedeme, bude interpretována bez dodatečných úprav, což by v případě unixtime znamenalo pouhé zobrazení čísla např. 1640967487
- *Update interval* – hodnota, která určuje, jak často se dotaz na položku bude provádět. Při volení intervalu je potřeba si uvědomit, že každý dotaz je dalším řádkem v databázi, což klade nároky na její výkon a velikost. Je dobré volit hodnoty, které nebudou zbytečně neúměrně zatěžovat systém (jak Zabbix tak i monitorovaného hosta) a budou přesto odpovídající. Autorovi se osvědčilo defaultně používat interval 3m (3 minuty).
- *Custom intervals* – někdy je potřeba intervaly monitoringu přizpůsobit dle času, což bývá použito na rozdílné periody o pracovních dnech či dnech mimopracovních. Flexible interval ignoruje hodnoty v *update interval* hodnotami nastavenými. Scheduling interval ignoruje *update interval* pouze v případě, že je pro něj interval nastaven.
- *History & Trend storage period* – nastavení, které upravuje, jak dlouho se mají nasbíraná data u položky držet v databázi. V případě trendů je u hodnot zobrazen i trend, který vizualizuje budoucí trend vývoje těchto hodnot.
- *Show value* – hodnoty mohou být převedeny dle určitého scénáře, který lze vytvořit v nastavení Administration/General/Value mapping. Zde se může nastavit, že pokud vrácená hodnota bude číslo 1, hodnota se převede na řetězec „OK“ apod.
- *Applications* – položka se může vztahovat k nějaké aplikaci. Proto vyplněním tohoto pole dojde ke sdružení hodnot pod aplikaci, což usnadní v budoucnu vyhledávání položek určených k aplikaci.
- *Populates host invertory field* – Zabbbix u Hostů vyplňuje karu Inventář, což umožní uložení informací potřebných pro potřeby inventarizace. Úpravou této položky upřesníme, že informace z položky budou propisovány do Inventáře k určité položce.
- *Description* – poznámky k položce, které slouží pro lepší přehled
- *Enabled* – zapnutí x vypnutí monitoringu této položky

Spouštěč (Trigger)

Aby byl Zabbixem vyvolán problém, musí být k položce přidán spouštěč. Pokud se bude získaná hodnota položky nacházet v rozmezí hodnot nastavených na spouštěči, bude spouštěčem vyvolána událost – Problém.

Na obrázku níže je zobrazeno vyvolání problému u položky system.localtime a to v případě, kdy bude získaná hodnota odlišná od času Zabbixe o více než 2 sekundy.

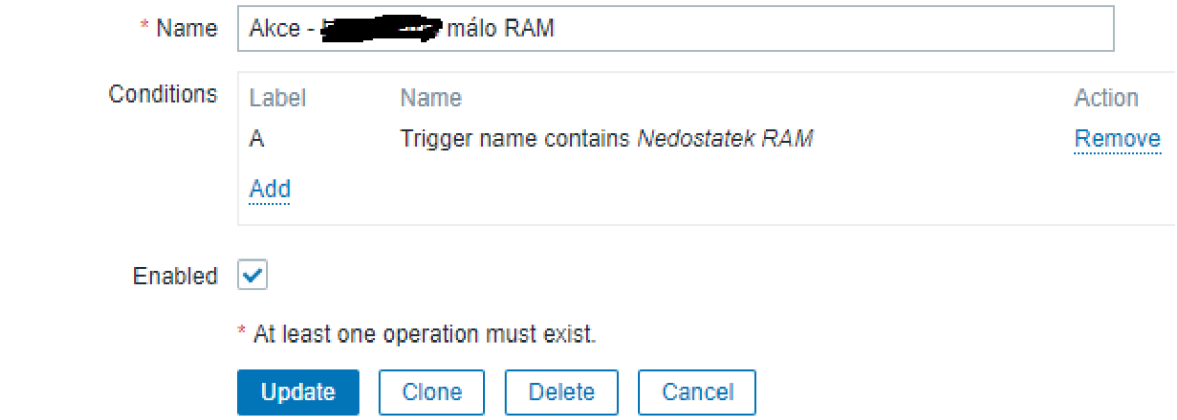
[21] Unix time epoch – počet sekund od data 1.1.1970 0:00:00 1



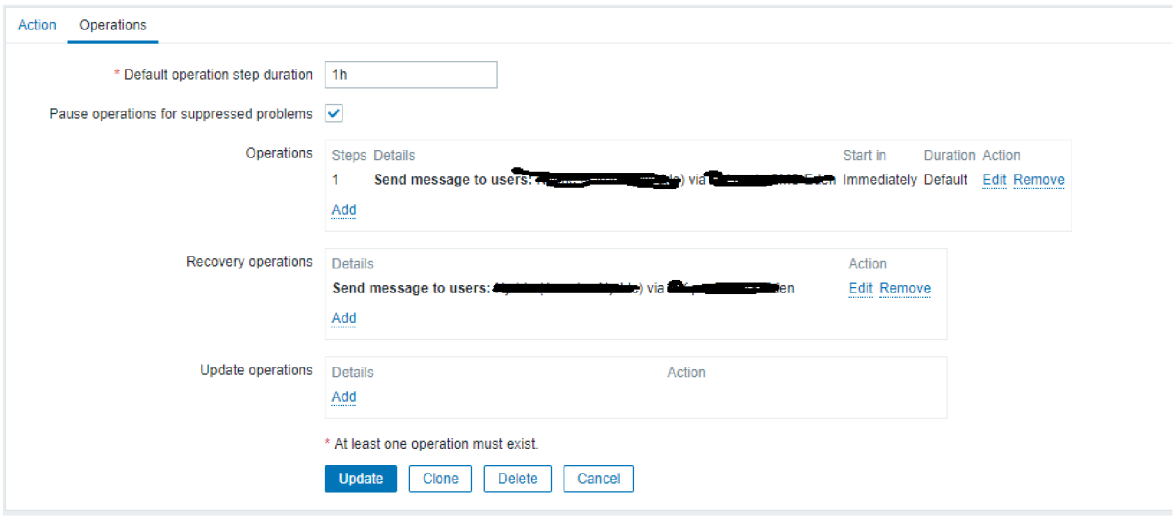
Obrázek 33 - nastavení spouštěče (zdroj: vlastní tvorba)

Jakmile dojde ke spuštění spouštěče, je na obrazovce Zabbixu v sekci Monitoring/Problems problém zaznamenám.
U spouštěče lze také nastavit důležitost problému (severity) na základě které lze pak vytvářet akce (Action).

Akce (Action)
Pokud dojde ke spuštění spouštěče (detekování problému), může vzniknout akce. Akce je rutina, která se provede automatizovaně, pokud jsou splněny vstupní podmínky. Typicky je akcí využíváno pro potřebu zaslat upozorňující email na výskyt problému. Zabbix však umožňuje mj. i spuštění skriptu, který může být proveden např. zabbix agentem u monitorovaného stroje, což může být použito pro automatickou opravu problému.
Na obrázku níže je zobrazena akce, která vznikne pouze při výskytu spouštěče jménem „nedostatek RAM“. Další obrázek zobrazuje operace, které se provedou. Nastavení akcí se provádí v menu Configuration/Action.

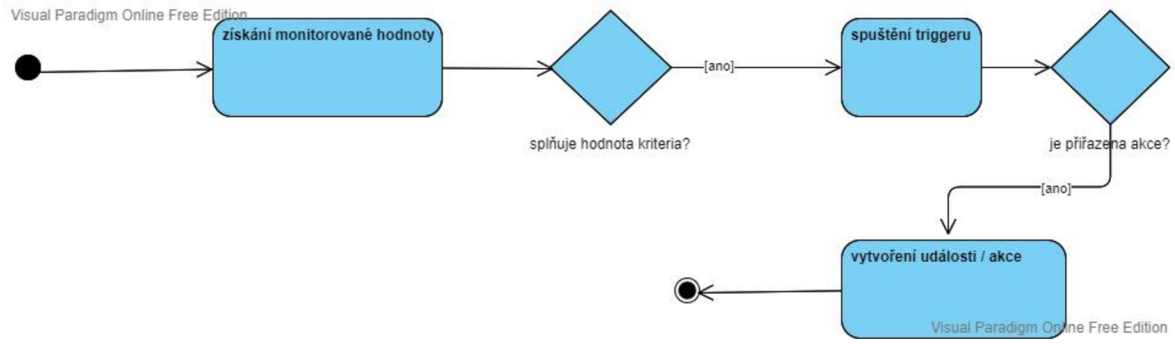


Obrázek 34 - nastavení akce (zdroj: vlastní tvorba)



Obrázek 35 - nastavení operace k akci (zdroj: vlastní tvorba)

Obrázek níže zobrazuje jednotlivé aktivity, které za splnění podmínek povedou ke spuštění akce respektive jejích operací.. Z obrázku je také patrné, jak jednotlivé prvky nastavení, které byly výše popsány, ovlivňují spuštění akce.



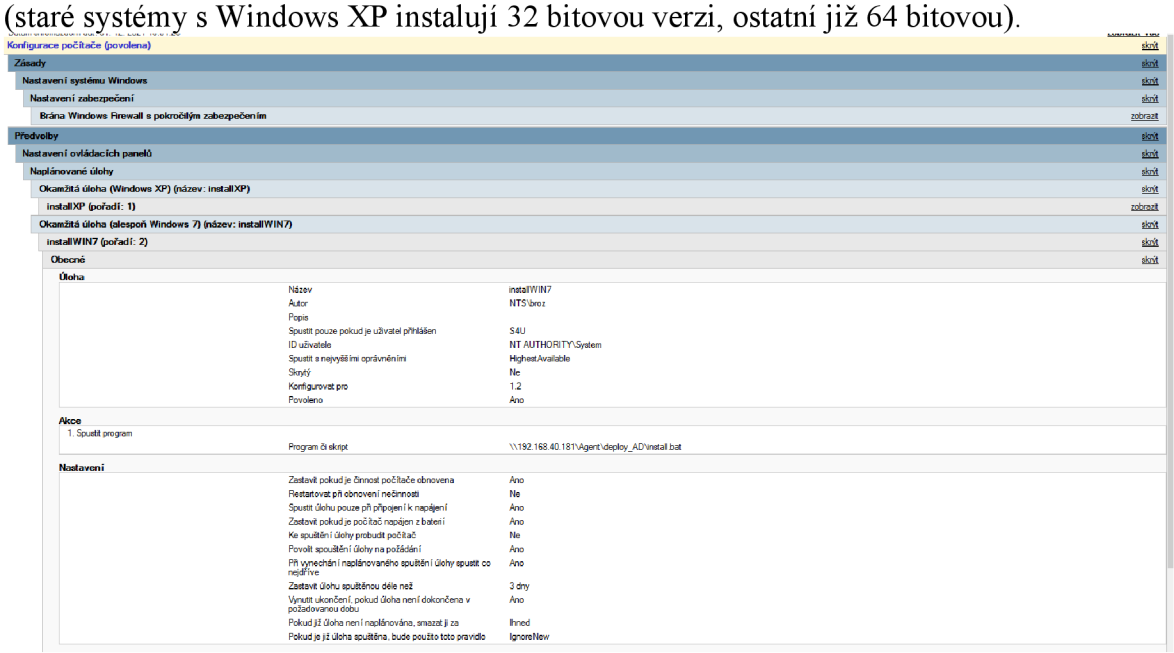
Obrázek 36 - diagram aktivit - vznik akce (zdroj: vlastní tvorba; on-line nástroj: https://online.visual-paradigm.com)

4.7 Vlastní přidaná hodnota

V předchozí kapitole bylo nastíněno, jak lze se systémem pracovat. Ve firemním prostředí, kde byla aplikace prováděna, vznikly specifické požadavky, které bylo potřeba zaimplementovat. Pro splnění těchto požadavků bylo potřeba systém rozšířit o vlastní přidanou hodnotu. Autor uvede některé případy, jak se s požadavky vypořádal.

4.7.1 Rozšíření zabbix agenta na klienty

Zabbix agent nabízí mnoho funkcí, jak z operačního systému windows či linux sbírat informace. Bohužel, v případě většího počtu stanic se instalace agenta stává časově náročnou operací. V případě použití agenta u OS Windows, bylo využito skutečnosti, že stroje jsou zařazeny do domény.
V doménové politice na úrovni Active directory (modul Správa zásad skupiny) byla vytvořena politika, která na podřízených stanicích otvírá porty potřebné pro komunikaci mezi agentem a serverem pomocí pravidel ve firewallu. Dále bylo přidáno pravidlo, které po spuštění počítače spustí skript, který je uložen ve sdíleném adresáři[22] na serveru Zabbix. Toto pravidlo bylo rozděleno dle operačních systémů (staré systémy s Windows XP instalují 32 bitovou verzi, ostatní již 64 bitovou).



Obrázek 37 - doménová politika pro šíření zabbix agenta (zdroj: vlastní tvorba)

Doménovou politikou skript při startu operačního systému otestuje, zda je služba zabbix_agent nainstalována. Pokud tomu tak není, vytvoří na klientské stanici potřebné adresáře, kam zkopíruje program zabbix_agent spolu s konfiguračním souborem. Následně agenta nainstaluje do systému a spustí jej.

[22] Postup vtvoření adresáře je blíže rozebrán v kapitole 4.5.4

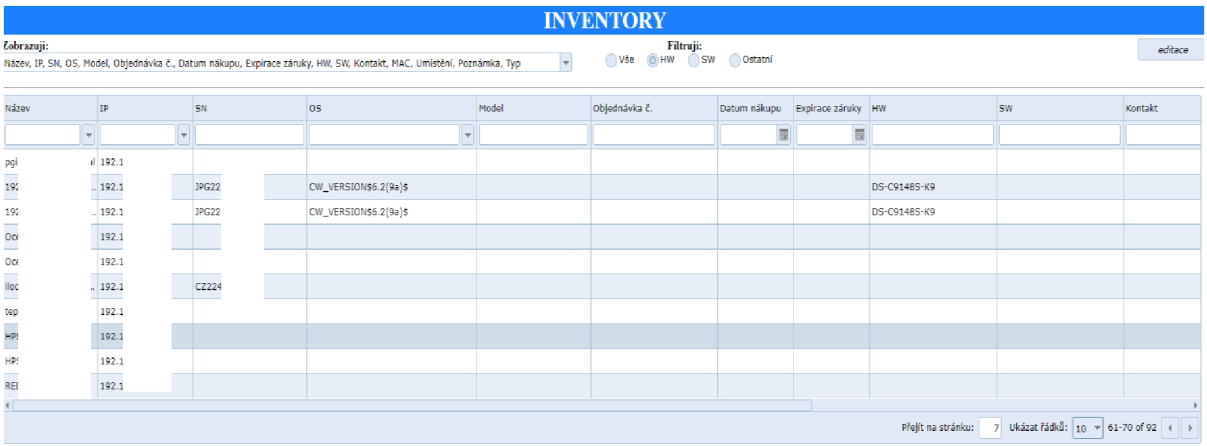
```
sc query | findstr "Zabbix"
if %ERRORLEVEL% EQU 0 GOTO END



:INSTALL
mkdir c:\zabbix_agent
mkdir c:\zabbix_agent\bin
mkdir c:\zabbix_agent\log
mkdir c:\zabbix_agent\conf
copy \\[redacted]\Agent\win\5.2\zabbix_agents_5.2.win\bin\win64\zabbix_agentd.exe c:\zabbix_agent\bin\zabbix_agentd.exe
copy \\[redacted]\Agent\deploy_AD\5.2\zabbix_agentd.win.conf c:\zabbix_agent\conf\zabbix_agentd.win.conf
c:\zabbix_agent\bin\zabbix_agentd.exe --config c:\zabbix_agent\conf\zabbix_agentd.win.conf  --install
c:\zabbix_agent\bin\zabbix_agentd.exe --start

:END
```

Obrázek 38 - skript spouštěný k instalaci zabbix agenta (zdroj: vlastní tvorba)

Pomocí výše uvedeného nastavení, lze docílit automatizovaného rozšíření zabbix agenta mezi klientské stroje s operačním systémem Windows.

4.7.2 Zpřístupnění Inventory pro skladové hospodářství

Z textu kapitoly 4.6.2 je patrné, jak lze automatizovaně vkládat položky do Inventáře. V praxi bylo však potřeba, umožnit editovat položky uživatelům z řad hospodářů. Jedná se o profese, které nemají přístup do systému Zabbix, avšak nasbíraná data jim mohou při práci pomoci.
Autor využil Zabbix API, díky kterému informace sbírá a zpřístupňuje je pomocí webové stránky i uživatelům mimo systém. Pokud uživatel má právo zápisu, může hodnoty přepisovat.



Obrázek 39 - webová stránka zobrazující INVENTORY pomocí ZabbixAPI (zdroj: vlastní tvorba)

4.7.3 Informace o běžících procesech

Vzhledem k tomu, že bylo potřeba získávat informace pro jiné systémy o stanicích windows, byl využit zabbix agent, který má vysokou úroveň oprávnění (local service) na tyto dotazy a je na všech stanicích automaticky instalován.

Na serveru Zabbix byl vytvořen stránka v PHP, při jejímž zavolání s příslušnými parametry došlo k dotazu na agenta, který provedl dotazy níže a vrátil stránku s jejich výsledky ve formátu json. Tyto data jsou pak dále jinými systémy zpracovány.

1) wmic OS get /format:list
    - zjistí verzi OS
2) wmic BIOS list /format:list
    - informace ke zjištění, zda je PC virtuální
3) wmic DISKDRIVE get /format:list
    - informace ke zjištění, zda je PC virtuální
4) reg QUERY "HKLM\Software\Eset\ESET Security\CurrentVersion\Info"
    - informace, zda je nainstalován antivir ESET
5) wmic path Win32_ReliabilityRecords get /format:list
    - informace o dalším SW
6) wmic qfe get /format:list
    - informace o dalším SW
7) wmic nicconfig where "IPEnabled=True" list ip /format:list
    - informace o IP adresách
8) reg QUERY HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters
    - informace, zda SMBv1 je zakázana v registrech
9) powershell -Command "& {Get-WindowsFeature FS-SMB1}"
    - informace, zda SMBv1 je zakázana v registrech (využití PowerShellu)

Část výstupu z dotazu:

{
"q1":" BootDevice=\Device\HarddiskVolume3 BuildNumber=18363 BuildType=Multiprocessor Free Caption=Microsoft Windows 10 Pro CodeSet=1250 CountryCode=420 CreationClassName=Win32_OperatingSystem CSCreationClassName=Win32_ComputerSystem CSDVersion=
CSName=PGTS21 CurrentTimeZone=60 DataExecutionPrevention_32BitApplications=TRUE DataExecutionPrevention_Available=TRUE DataExecutionPrevention_Drivers=TRUE DataExecutionPrevention_SupportPolicy=2 Debug=FALSE Description= Distributed=FALSE
EncryptionLevel=256 ForegroundApplicationBoost=2 FreePhysicalMemory=10124328 FreeSpaceInPagingFiles=2439848 FreeVirtualMemory=10480672 InstallDate=20191129091005.000000+060 LargeSystemCache= LastBootUpTime=20211223190200.500000+060
LocalDateTime=20211231201300.122000+060 Locale=0405 Manufacturer=Microsoft Corporation MaxNumberOfProcesses=4294967295 MaxProcessMemorySize=137438953344 MUILanguages={"cs-CZ"} Name=Microsoft Windows 10 Pro|C:\Windows\|Device\Harddisk1\Partition3
NumberOfLicensedUsers=0 NumberOfProcesses=190 NumberOfUsers=7 OperatingSystemSKU=48 Organization= OSArchitecture=64bitov OSLanguage=1029 OSProductSuite=256 OSType=18 OtherTypeDescription= PAEEnabled= PlusProductID= PlusVersionNumber=
PortableOperatingSystem=FALSE Primary=TRUE ProductType=1 RegisteredUser=syspg1 SerialNumber=00330-52336-89826-AAOEM ServicePackMajorVersion=0 ServicePackMinorVersion=0 SizeStoredInPagingFiles=2490368 Status=OK SuiteMask=272
SystemDevice=\Device\HarddiskVolume3 SystemDirectory=C:\Windows\system32 SystemDrive=C: TotalSwapSpaceSize= TotalVirtualMemorySize=19082056 TotalVisibleMemorySize=16591688 Version=10.0.18363 WindowsDirectory=C:\Windows ",
"q2":" BiosCharacteristics={7,9,11,12,15,16,19,23,24,25,26,27,28,29,32,33,40,41,42,43} BuildNumber= CodeSet= CurrentLanguage=en|US|iso8859-1 Description=1.0.3 IdentificationCode= InstallableLanguages=2 InstallDate= LanguageEdition= ListOfLanguages={"en|US|iso8859-1",""}
Manufacturer=Dell Inc. Name=1.0.3 OtherTargetOS= PrimaryBIOS=TRUE ReleaseDate=20190530000000.000000+000 SerialNumber=D75NY03 SMBIOSBIOSVersion=1.0.3 SMBIOSMajorVersion=3 SMBIOSMinorVersion=1 SMBIOSPresent=TRUE SoftwareElementID=1.0.3
SoftwareElementState=3 Status=OK TargetOperatingSystem=0 Version=DELL - 1072009 ",

Obrázek 40 - výstup z dotazu získaného z agenta (zdroj: vlastní tvorba)

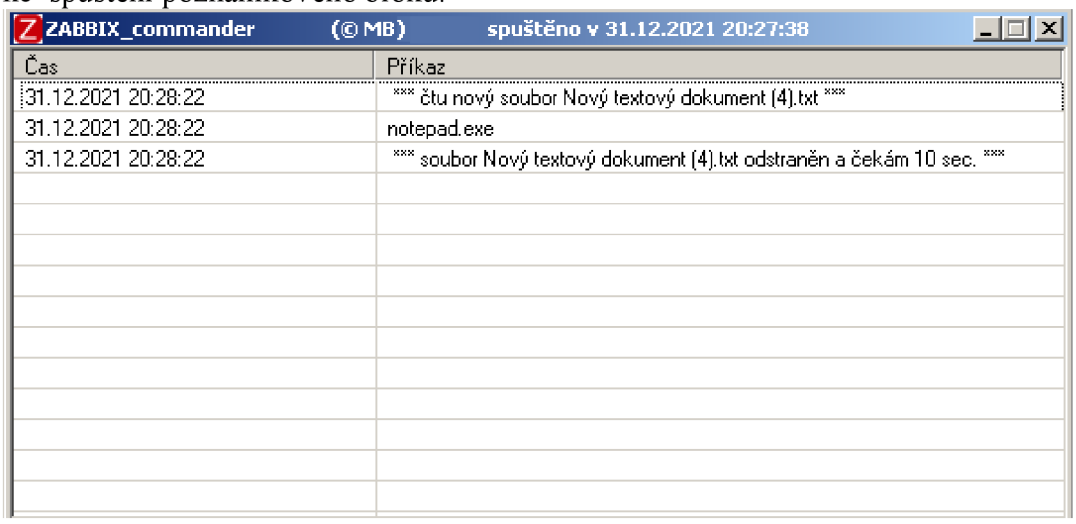
4.7.4 Kontrola počtu souborů

Ve firmě, kde autor pracuje, je potřeba také kontrolovat počet souborů v adresáři, a upozornit v případě překročení jejich počtu. Tento druh dotazu není Zabbixem implementován.
Autor tedy vytvořil konzolovou aplikaci, která po spuštění projde adresář a návratová hodnota je číslo s aktuálním počtem souborů. Součástí aplikace je konfigurační

soubor, kde lze specifikovat, zda se mají hledat soubory *starší než* a to dle data vytvoření či modifikace.
Položka v Zabbix serveru je upravena tak, aby daný soubor spouštěla, a její spouštěč kontroluje návratovou hodnotu, dle které vyvolává příslušnou akci.
Výše uvedený postup jednoduše rozšiřuje možnosti dohledu o vlastní často specifické požadavky.
Jako vhodné úložiště pro skripty, návody a různé jiné pomůcky se systémem Zabbix zvolil autor sdílený adresář na Zabbix cloudu, který vytvořil při instalaci systému. Zároveň je využito funkcionality HA, takže adresář bude vždy dostupný.

4.7.5 Spuštění aplikací s GUI

Zabbix při vykonání akce (viz oddíl 4.6.2) může pomocí zabbix agenta spustit jiný proces. Toho může být využito v situaci, kdy je zjištěno, že proces z nějakého důvodu neběží a je potřeba jej spustit. Nevýhodou ale je, že agent je spuštěn jako lokální služba a spustí-li jiný proces, tak proces bude spuštěn, ale nebude u něj spuštěno GUI (grafické okno aplikace).
Autor se s tímto problémem vyrovnal tak, že na potřebné stanice nainstaloval vlastní program zabbix_commander (WinForm – aplikace s vlastním GUI), který kontroluje soubory v adresáři na přítomnost textu. Pokud se tam text objeví, spustí příkaz resp. kód, který je tam vložený formou textu. Pomocí operace nastavené v *Akci* dojde tedy na cílové stanici k naplnění textového souboru zabbix_agentem a zabbix_commander příkaz provede. Příkazem může být spuštění nějaké aplikace s vlastním GUI. Níže obrázek zobrazuje provedení příkazu notepad.exe, které vede ke spuštění poznámkového bloku.



Obrázek 41 - zabbix_commander (zdroj: vlastní tvorba)

5 Zhodnocení výsledků a doporučení

Postupem, který byl v praktické části doložen, došlo ke splnění požadavků firmy, kde byla implementace dohledového systému aplikována.

5.1 Analýza společnosti

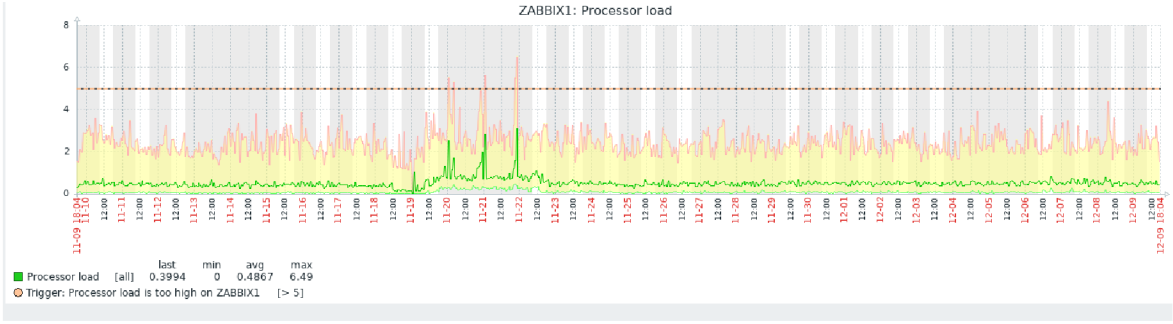
První činností, která musí předcházet implementaci kteréhokoliv systému, je důkladná analýza. Přestože je autor zaměstnancem společnosti, ve které systém implementoval, musely první činnosti spočívat v dokonalém průzkumu firemního prostředí a to zejména jeho IT technologií. Současně autor aktivně zjišťoval požadavky uživatelů a vedení, které měly na přizpůsobení systému a celkovou implementaci velký vliv.
Zajímavé bylo zjištění, že autorova představa o firemním IT nebyla úplná, jak předpokládal. Při zjišťování detailů o síťové infrastruktuře či používaných zařízeních autor získal cenné informace, které mu pak byly k užitku při samotné aplikaci systému. Lze říci, že bez těchto zkušeností by zřejmě topologie systému a jeho využití nebylo možno upravit dle specifických potřeb firmy.

5.2 Návrh řešení

Instalace systému Zabbix a jeho různé způsoby využití jsou v mnoha článcích a fórech už mnohokrát řešeny. Dokonce samotný webový portál Zabbix.com nabízí pěkně udělané návody (mnohdy ve formě videozáznamu), které postup instalace detailně řeší.
Různé požadavky firmy o další rozšíření systému včetně HA dostupnosti monitorovacího systému však znamenala, že autor nebude moci použít standardních postupů a bude muset návrh o tyto požadavky doplnit a tedy vymyslet vlastní postup implementace.
Postupy, které byly v této práci předloženy, nejsou určitě jediným možným řešením jak tyto cíle splnit. Autor však doposud nezpozoroval nevýhody tohoto způsobu řešení.

5.3 Implementace řešení

Implementace byla bezproblémová. Pokud se nějaký problém objevil, bylo v možnostech autora problémy vyřešit. Obavy z nedodržení termínu či nedodržení ceny implementace se nenaplnily.
Dimenzování použitého HW se ukázalo jako dostatečné. Níže je zobrazen graf využití CPU u Zabbix serveru.



Obrázek 42- využití CPU u Zabbix serveru (zdroj: vlastní tvorba)

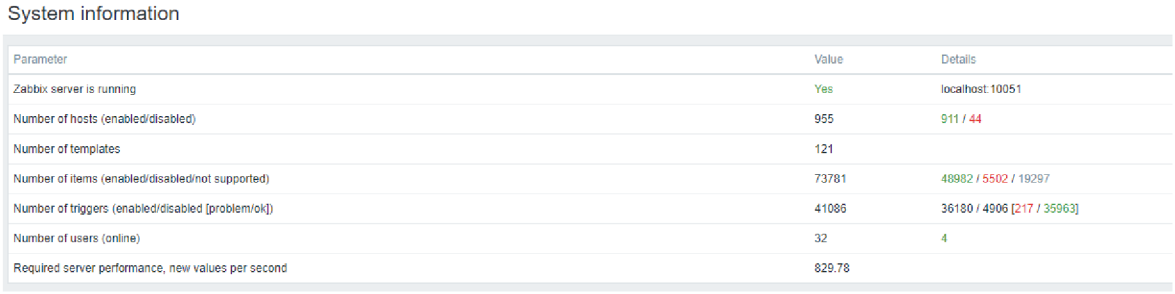
Dnešním trendem v IT odvětví je snaha o virtualizaci serverů ať už z důvodu úspory energie či z jiných důvodů. Autor připouští, že i ušetřené prostředky za fyzické servery použité v této implementaci (cca. 400.000 Kč včetně DPH) nejsou zanedbatelné. Výhodou tohoto řešení je ale fakt, že jde o systém s redundantními klíčovými prvky, který je nezávislý na ostatní firemní technologii. Skutečnost, že je systém hostován přímo na fyzických serverech, které jsou k tomuto účelu dedikovány, přispívá k rychlosti systému, kdy jsou např. operace I/O[23] prováděny přímo a ne přes prostředníka – hypervizor.

5.4 Využití systému

Zpočátku bylo potřeba proškolit uživatele a to zejména budoucí administrátory systému. Jelikož podobný druh systému nebyl ve firmě doposud používán, bylo využití samotnými uživateli bráno spíše skepticky. Postupem času se však počet dohledovaných položek rozšířil tak, že uživatelům- tedy IT technikům umožnil monitoring veškerých dohledovaných technologií. Zároveň byly udělány úpravy systému přesně dle požadavků uživatelů, díky čemuž se stal Zabbix nepostradatelným systémem společnosti.

Obrázek níže zobrazuje sumární počty dohledovaných položek ve společnosti systémem Zabbix. Je patrné, že tyto počty nelze lidskými zdroji kontrolovat. Autor této práce by také rád zdůraznil, že došlo ke změně struktury řešených problémů, kdy se řeší již více problémů s předstihem než v době, kdy už jsou „viditelné“ a mají negativní dopad na firemní business.

[23] I/O – vstupně/výstupní operace



Obrázek 43 - Systémové informace instalovaného systému Zabbix ve společnosti (zdroj: vlastni tvorba)

5.5 Ekonomické zhodnocení

Náklady na vznik a na udržení chodu systému byly kalkulovány z pohledu zaměstnavatele. Měrnou jednotkou jsou MD (man-day), což jsou náklady na jeden den pracovníka. Tyto náklady byly vypočteny na základě hodinové mzdy, která byla ve výši 400 Kč, tudíž 1 MD má hodnotu 3200,- Kč.
Data v první tabulce ukazují, jaké jsou náklady firmy na pracovníka, který 8 hodin denně pracuje na pozici IT dohledu.

zaměstnanec	
položka	**cena (Kč)**
hrubá mzda	45 000,00
sociální pojištění	11 160,00
zdravotní pojištění	4 050,00
celkem	**60 210,00**

Tabulka 8 – měsíční firemní náklady na pracovníka dohledu (zdroj: vlastní tvorba)

Druhá tabulka již zobrazuje náklady na dohledový systém. Tyto náklady lze rozdělit na pravidelné – měsíční a jednorázové. U jednorázových se jedná se o náklady, které bylo potřeba vynaložit na prvotní zprovoznění systému, nebo budou vznikat jen občas a to pro splnění konkrétního úkolu Nejde jen tedy o samotný hardware, ale i počty jednotlivých MD, které byly čerpány za účelem zprovoznění systému a pro jeho udržení. Tyto náklady, včetně nákladů na jednorázové aktualizace a jiné jednorázové úkony v podobě MD, byly rozpočítány do 72 měsíců (6 let plánované životnosti systému), aby byl patrný a srovnatelný měsíční náklad se zaměstnancem (viz tabulka 8).

Zabbix		
položka	**cena (Kč)**	**poznámka**
pořízení HW	5 491,50	investice 395 388,- Kč rozložena do 6-ti let
údržba / práce se systémem	3 200,00	1MD
instalace, zprovoznění	445,00	10 MD rozloženo do 6-ti let
aktualizace, řešení chyb, ostatní	667,00	15 MD rozloženo do 6-ti let
celkem	**9 803,50**	

Tabulka 9 - měsíční náklady na provoz systému Zabbix (zdroj: vlastní tvorba)

Z porovnání výsledných součtů je patrné, že dohled pomocí dohledové technologie (a to nejenom Zabbix) je výrazně levnější nežli pomocí lidského pracovníka. Uvědomíme-li si, že člověk pracuje pouze v pracovní dny a to 8 hodin oproti technologii, která pracuje non-stop, rozdíly nákladů jsou pak ještě markantnější. Nehledě na to, že maximální počet dohledovaných položek je u technologického dohledu oproti lidskému prakticky neomezený.
Na druhou stranu je nutno podotknout, že k technologickému dohledu je potřeba lidského, který s ním kooperuje.

5.6 Doporučení

Pro bezproblémový chod systému a zároveň pro jeho udržitelnost ve společnosti je potřeba systém neustále udržovat tak, aby jeho monitorované položky odpovídaly aktuálnímu stavu dozorované technologie. K zajištění výše uvedené potřeby je vhodné upravit řízení procesů v IT tak, aby administrátoři Zabbixu byli o každé změně informování a v případě potřeby mohli změnu do něj naimplementovat. Osvědčilo se také dělat pravidelné schůzky, kde administrátoři oznámili již udělané či chystané změny a od uživatelů se dozvěděli, jaké jsou nové požadavky na systém ze strany uživatelů.

Mimo standardní profylaxi serverů, která by měla určitě zahrnovat aktualizaci SW a údržbu HW, se osvědčilo dělat pravidelné statistiky zaznamenaných problémů Zabbixem. Na základě těchto informací byly pak upravovány vlastnosti dohlížených položek či spouštěčů, aby odrážely aktuální potřeby monitoringu. Stalo se běžnou praxí, že u častých problémů byla vytvořena akce (viz kapitola 4.6.2), díky které byl problém v budoucnu již automaticky Zabbixem řešen. Těchto zautomatizovaných procesů přibývá a citelně uvolňují IT technikům kapacity pro jejich další činnosti.

U spouštění akcí či vyhlašování problémů systémem se osvědčilo aplikování prodlev. V popsaném systému byla nastavena defaultní hodnota na 6 minut, což se také osvědčilo. V případě nenastavení této prodlevy docházelo často k situacím, kdy byli uživatelé informováni o problému a vzápětí zase o zrušení problému – typicky kontrola rozdílu systémového času. V případě nastavení prodlevy se zredukoval počet těchto zpráv a zvýšila se jejich vypovídající hodnota.

6 Závěr

V rámci této práce byly analyzovány možnosti různých protokolů, které se používají k monitoringu technologií v IT. Zároveň byly zjištěny možnosti současných systémů, které se touto problematikou zabývají. Na základě těchto zkušeností byl vybrán monitorovací systém Zabbix, který byl autorem této práce naimplementován ve skutečné společnosti.
Implementace obnášela dokonalé zmapování firemní IT struktury, včetně pochopení požadavků a potřeb společnosti. Dále byl vypracován návrh na nový dohledový systém a vybrány odpovídající servery nutné k realizaci. Po provedení nákupu došlo k fyzickému zapojení serverů a samotné instalaci systému. Autor této práce do systému přidává mimo jiné funkcionalitu vysoké dostupnosti, které se tato práce také věnuje.
V další části se autor již zaměřuje na vlastní použití systému Zabbix skrze jeho grafické rozhraní. Názorně je předvedeno, jak byly vytvářeny jednotlivé položky určené k monitoringu a jejich další použití, které může být například pouze upozornění, ale i automatizovaná oprava detekovaného problému.
Samostatná kapitola je věnována také rozšířením Zabbix, které autor vytvořil pro potřeby firmy. Jsou to některé ukázky možností, jak je možno dál se systémem pracovat a přidávat mu vlastní další funkcionality.

Zavedením systému Zabbix ve společnosti společně s úpravou procesů IT firmy došlo k požadovanému zlepšení v oblasti monitoringu IT zařízení. Samotný systém byl rozšířen o další funkcionality, které přinášejí užitek společnosti a to nejenom v IT odvětví. Přestože pořízení systému byl pro firmu nemalý náklad, již nyní je patrné, že to je investice, která se firmě vyplatí a zvýší úroveň kvality jejího IT a to nejenom v rychlosti odhalení a tedy i následných oprav zjištěných problémů.
Tabulka níže ukazuje, jak se změnily hodnoty jednotlivých položek v závislosti na systému Zabbix v jednotlivých ročních kvartálech, kdy bylo prováděno měření.

	Q4 2020 (bez Zabbix)	Q1 2021 (bez Zabbix)	Q2 2021	Q3 2021
počet dohledovaných položek	N/A	N/A	38458	47527
počet problémů s úrovní vyšší než varování trvajících déle než 6 minut	288	325	580	602
průměrná doba rakce na problém (v minutách)	152	112	24	19
zásahy mimo pracovní dobu	21	19	9	7

Tabulka 10 - naměřené hodnoty ve vztahu k implementaci Zabbix (zdroj: vlastní tvorba)

Hodnoty, které byly naměřeny před implementací technologie Zabbix, byly získány z elektronického deníku firmy. Do tohoto deníku mají zaměstnanci povinnost psát učiněné zásahy nad dohlíženou technologií.
Z tabulky lze usoudit, že mnoho problémů bylo díky zavedenému systému nově zjištěných a doba reakce na problémy se velmi snížila a to i včetně počtu zásahů provedených mimo pracovní dobu.

Naimplementovaný dohledový systém přinesl i přidané hodnoty jako jsou např. možnosti predikce vývoje sledovaných hodnot, zobrazení historických dat či využití

těchto dat k dalšímu využití jinými systémy či odděleními firmy, což bylo nastíněno v této práci.

Bez nadsázky lze konstatovat, že funkci současného dohledového systému ve společnosti již nelze nahradit lidským faktorem.

7 Seznam použitých zdrojů

(1) *DevOps and Security Glossary Terms* [online]. Sumo Logic [cit. 2021-08-10]. Dostupné z: https://www.sumologic.com/glossary/infrastructure-monitoring/

(2) *ISO / IEC 7498-4: Information processing systems - Open Systems Interconnection - Basic Reference Model*. Part 4. Švédsko, 1989.

(3) *Internet End-to-end Performance Monitoring: Passive vs. Active Monitoring* [online]. 2001. Stanford: SLAC National Accelerator Laboratory, 2001 [cit. 2021-08-10]. Dostupné z: https://www.slac.stanford.edu/comp/net/wan-mon/passive-vs-active.html

(4) KEREN, Emil. *It's Not about Agent vs. Agentless Monitoring Anymore* [online]. [cit. 2021-08-10]. Dostupné z: https://www.eginnovations.com/blog/agentless-vs-agent-based-monitoring/

(5) SEMAYAT FIKRE, Handoro. *Sledování výkonu vysoce složitých síťových systémů*. Praha, 2021. Diplomová práce. ČZU. Vedoucí práce Jiří Vaněk.

(6) MARIK, Ondrej a Stanislav ZITTA. Comparative analysis of monitoring system for data networks. *2014 International Conference on Multimedia Computing and Systems (ICMCS)*. Marrakech: IEEE, 2014. Dostupné z: doi:10.1109/ICMCS.2014.6911307

(7) ZAVORAL, Filip. *Distribuované operační systémy*. Praha, 2001. Dostupné také z: https://docplayer.cz/10213821-Distribuovane-operacni-systemy.html

(8) SCHMIDT, Klaus. *High Availability and Disaster Recovery: Concepts, Design, Implementation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. ISBN 978-3-540-34582-4.

(9) *Zabbix - Oerations in IT* [online]. [cit. 2021-04-27]. Dostupné z: https://blog.oper-init.eu/zabbix/

(10) CLARK, Martin. *Data networks, IP and the Internet: PROtocol, Design and Operation*. Chichester: Wiley, 2003. ISBN 0-470-84856-2.

(11) *SNMP* [online]. [cit. 2021-04-27]. Dostupné z: https://www.oreilly.com/library/view/snmp

(12) Zabbix documentation: 5.4. *Zabbix.com* [online]. [cit. 2021-08-10]. Dostupné z: https://www.zabbix.com/documentation/current/manual

(13) Zabbix: Web frontend. *Zabbix.com* [online]. [cit. 2021-08-10]. Dostupné z: https://www.zabbix.com/zabbix_web_frontend

(14) *Foldoc: Free online dictionary of computing* [online]. 2021 [cit. 2021-08-10]. Dostupné z: http://foldoc.org/Application+Program+Interface

(15) *Server monitoring* [online]. [cit. 2021-04-27]. Dostupné z: https://tecadmin.net/api-zabbix-server-monitor

(16) *IT slovník* [online]. [cit. 2021-08-10]. Dostupné z: https://it-slovnik.cz/pojem/modul

(17) *Network Monitoring: Protocols, Best Practices, and Tools* [online]. [cit. 2021-08-10]. Dostupné z: https://www.tek-tools.com/network/network-monitoring-guide-and-tools

(18) *RFC 972: INTERNET CONTROL MESSAGE PROTOCOL*. 1981. Dostupné také z: https://datatracker.ietf.org/doc/html/rfc792

(19) MARO, Douglas a Kevin SCHMIDT. *Essential SNMP*. 2. Gravenstein: O`Reilly media, 2005. ISBN 0-596-00840-6.

(20) *Cloud monitoring* [online]. [cit. 2021-05-12]. Dostupné z: https://www.sumologic.com/glossary/infrastructure-monitoring

(21) *MIB* [online]. [cit. 2021-05-29]. Dostupné z: http://www.regatta.cs.msu.su/doc/usr/share/man/info/ru_RU/a_doc_lib/aixprggd/proomc/mib.htm

(22) SUN MICROSYSTEM. *Embedded Lights Out Manager Administration Guide: For the Sun Firetrademark X2200 M2 and Sun Fire X2100 M2 Servers* [online]. 2009. [cit. 2021-08-10]. ISBN 819-6588-14. Dostupné z: https://docs.oracle.com/cd/E19121-01/sf.x2200m2/819-6588-14/ipmicom.html

(23) *BMC -Baseboard Management Controller* [online]. [cit. 2021-07-04]. Dostupné z: https://electronicdesk.com/80286.html

(24) QUSAY, Mahmoud. *Getting Started with Java Management Extensions (JMX): Developing Management and Monitoring Solutions* [online]. Oracle. 2004 [cit. 2021-08-10]. Dostupné z: https://www.oracle.com/technical-resources/articles/javase/jmx.html

(25) *Relationship between components of the JMX architecture* [online]. [cit. 2021-07-27]. Dostupné z: https://www.researchgate.net/figure/Relationship-between-components-of-the-JMX-architecture_fig1_46299401

(26) OLUPS, Richard. *Zabbix network monitoring: second edition*. 2. Birmingham: Packt publishing, 2016. ISBN 978-1-78216-128-8.

(27) NAGIOS. Nagios.org. *Nagios* [online]. [cit. 2021-08-10]. Dostupné z: www.nagios.org

(28) Checkmk. *Checkmk* [online]. [cit. 2021-08-10]. Dostupné z: www.checkmk.com

(29) OpenMMS. *OpenMMS* [online]. [cit. 2021-08-10]. Dostupné z: https://www.openmms.org/

(30) *Informatika a grafika* [online]. 1 [cit. 2021-12-7]. Dostupné z: https://www.gjszlin.cz/ivt/esf/ostatni-sin/raid.php

(31) *Zálohování a bezpečnost dat nemocničního informačního systému*. České Budějovice, 2012. Bakalářská práce. Jihočeská univerzita v Českých Budějovicích. Vedoucí práce Ing. Zdeněk Čuta.