Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Diplomová práce

Monitorování mailserverů Kerio Connect webovými službami

Plzeň, 2012 Daniel Gruber

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne ....

Daniel Gruber

Poděkování

Děkuji svému vedoucímu diplomové práce Ing. Dušanovi Juhásovi za jeho podporu a mnoho cenných rad při vedení diplomové práce. Dále bych rád poděkoval svým rodičům a přátelům za morální i finanční podporu při studiu.

Abstract

Monitoring of mailservers Kerio Connect via web services

This diploma thesis gives an overview of monitoring network devices and system resources via web services. It analyses features and advantages of software Nagios and describes how to program user defined monitoring plugins. In the next part it presents mailserver Kerio Connect with focus to Administration API, which is used for server management and third-party products integration. It also provides specification of plugins features and their behaviour, these plugins which are monitoring the state of mailservers Kerio Connect using Administration API. Methods and results from plugins set testing are listed in the last part of thesis.

Obsah

1 Úvod

Cílem práce je seznámit čtenáře s problematikou monitorování síťových zařízení a systémových prostředků webovými službami. V úvodu práce nalezne čtenář popis a porovnání vybraných softwarových produktů, které jsou často využívány pro síťové monitorování.

Dále práce představuje systém Nagios, který se stal průmyslovým standardem v oblasti webového monitorování, jelikož přebírá výhody a důležité vlastnosti z několika jiných softwarových prostředků. Práce uvádí, jaká zařízení a jejich služby se mohou monitorovat, jaký má monitorování princip a jak Nagios vyhodnocuje získané výsledky. V práci čtenář nalezne informace potřebné pro naprogramování vlastních monitorovacích pluginů, dozví se jakou strukturu a výstupy musí plugin mít a jak jej úspěšně nasadit.

V další části analyzuje práce poštovní server Kerio Connect, jeho webová rozhraní, podporované technologie a síťové služby. Následně se zaměřuje na Administrační API pro Kerio Connect využívané při správě serveru a integraci produktů třetích stran.

Hlavním cílem práce je implementovat sadu pluginů do systému Nagios monitorujících stav několika poštovních serverů Kerio Connect za použití jejich administračního rozhraní. Pluginy mají napomoci ke spolehlivému provozu těchto serverů a usnadňovat jejich správu.

Po přečtení práce získá čtenář představu o vlastnostech a o principu činnosti softwaru Nagios v porovnání s dalšími webovými službami určenými pro monitorování. Bude vědět, jakým způsobem programovat vlastní pluginy a jak je v systému spouštět. Také bude mít informace o funkcích poštovního serveru Kerio Connect a především pak o jeho administrační rozhraní. Navíc bude znát princip činnosti implementovaných pluginů monitorujících stav poštovních serverů a bude seznámen s postupem a výsledky provedeného testování této sady.

2 Softwarové monitorovací nástroje

V této kapitole je detailně rozebráno několik vybraných softwarových monitorovacích nástrojů se zaměřením na ty, které nabízejí dohled nad síťovým provozem a síťovými službami. Vlastnosti popsané sady služeb budou v závěru kapitoly shrnuty a lze je porovnat se systémem monitorování Nagios, kterému se tato práce dále věnuje (viz kap. 3 – Základní popis systému Nagios).

2.1 RMON

Standard RMON [RMON] (Remote monitoring) je systémem navrženým tak, aby přenesl výpočetní zátěž při sběru dat ze serveru na klienta.

Komunikace mezi clientem (označován jako RMON sonda) a serverem (správcovským programem) je zprostředkována protokolem SNMP (Simple Network Management Protocol) [SNMP].

Client je inteligentní síťový agent obsažený v síťovém prvku (router, switch, bridge), který sám provádí sběr aktuálních informací o probíhající komunikaci na síťovém segmentu a také shromažďuje historickou statistiku. Veškerá data pak ukládá do lokální MIB (Management information base) databáze.

Sonda je konfigurovatelné zařízení, které lze SNMP příkazy Set nastavovat na měření konkrétních dat, vyvolání asynchronní zprávy při události, nebo nastavit periodu vzorkování historické statistiky. Data ze sondy server získává buď při výskytu asynchronně (metoda SNMP Trap), nebo na vyžádání (metody SNMP Get a GetBulk) dle hodnoty OID (Object identifier) z MIB databáze.

Sonda RMON umožňuje shromažďovat informace o protokolech Ethernet a Token Ring. Shromažďovaná data dělí do 9 skupin pro Ethernet/Token Ring a do 10 skupin pro Token Ring dle jejich charakteru. Pro protokol Ethernet lze například ukládat aktuální i historické síťové informace o jednotlivých uzlech síťového segmentu (identifikace MAC adresou), statistiku vzájemné komunikace každé dvojice uzlů v segmentu, nebo o komunikaci na jednotlivých síťových rozhraních sondy. Většina statistik obsahuje čítače pro počty přenesených octetů, počty chybových octetů, počty octetů dle velikosti (histogram) atd.

2.2 NetFlow

NetFlow [NetFlow] je proprietární protokol vyvinutý firmou Cisco Systems pro sběr informací o IP přenosu dat. Architektura protokolu se podobá architektruře RMON, kdy se v systému nacházejí exportéři neboli sondy (z angl. exporters) a kolektory (z angl. collectors). Komunikace mezi exportéry a kolektory

je ovšem možná i ve vztahu M:N, což znamená, že exportér může posílat data více kolektorům a kolektor přijímá data od více exportérů.

Exportéři v IP přenosu data monitorují tzv. IP tok. Jedním tokem jsou rozuměny pakety, které mají společnou zdrojovou a cílovou IP adresu, zdrojový a cílový TCP / UDP port, IP protokol a společné ToS (type of service).

Kolektor shromažďuje data, ukládá je do databáze a obvykle obsahuje software pro vizualizaci nasbíraných dat.

Uchovávaná informace o datovém toku zahrnuje také, kromě výše uvedených identifikátorů, dobu komunikace a počet paketů a oktetů a zdrojovou a cílovou masku. Pokud se jedná o TCP spojení pak také příznaky. Některé systémy uchovávají zdrojové a cílové číslo AS (autonomous system).

Firma Cisco Systems nasazovala nejprve exportéry protokolu Netflow do všech svých směrovačů, ale tato volba sebou nesla několik zásadních problémů. Kromě nejvýkonnějších směrovačů byly zařízení zahlceny sběrem dat a jejich rozesíláním ke kolektorům a neměly proto dostatečný směrovací výkon. Jedním z řešení bylo vzorkování dat exportérem, které ovšem mělo dopady na přesnost výsledků a na kvalitu odhalování bezpečnostních incidentů. Architektura protokolu byla proto rozšířena o tzv. pasivní sondy, které jsou umístěny v síti jako samostatné zařízení. Data přes ně procházejí bez nutnosti směrovat a tudíž beze změny. Pasivní exportéři bývají také často propojeni dedikovanou linkou přímo ke kolektorům, a proto nezatěžují ostatní komunikační kanály.

Komunikace mezi kolektory a exportéry probíhá protokoly SCTP (stream control transmission protocol) nebo UDP (universal datagram protocol). Po odeslání dat z exportérů je sonda smaže bez čekání na potvrzení, což může vést ke ztrátě dat.

Velcí výrobci síťové techniky nechtěli být závislí na proprietárním standardu NetFlow chtěli vytvořit společný průmyslový standard pro export informací o IP toku. Z tohoto důvodu vznikl IPFIX (Internet Protocol Flow Information eXport), nebo-li vylepšený protokol NetFlow verze 9. Funkčnost původního protokolu NetFlow společnosti převzaly a změnily jeho název (např. JFlow pro Juniper, NetStream pro 3Com a HP, RFlow pro Ericsson).

2.3 Munin

Munin [Munin] je softwarový monitorovací nástroj určení k monitorování velkého množství údajů převážně pro platformu Linux. Software je vybudován na architektuře klient – server.

Serverovou část zastupuje skript napsaný v perlu, který je periodicky spouštěn programem cron a který se dotazuje jednotlivých klientů. Získaná data se zobrazují do webového rozhraní v podobě grafů vykreslených nástrojem RRDTool (Round Robin Database Tool) [RRDTool].

Klientem je démon munin-node běžící na monitorovaném stroji. Démon spouští periodicky sadu definovaných pluginů, které získávají údaje. Odeslání dat probíhá buď přímou komunikací se serverem (na známém čísle portu), nebo navázáním SSH komunikace (přihlášení klíči).

Plugin měří jednu konkrétní vlastnost (hodnotu) a poskytuje výstup textové informace v syntaxy RRDTool. Munin obsahuje velkou sadu již vytvořených pluginů schopných získávat například systémové informace (využití procesoru, paměti, počet přihlášených uživatelů a další), data o serverových démonech (webový server, databázový systém a další), ale i pluginy pro monitorování síťového provozu, či monitorování přes SNMP. Munin umožňuje doprogramovat i vlastní pluginy.

2.4 PRTG Network Monitor

PRTG Network Monitor [PRTG] (Paessler Router Traffic Grapher) je proprietární software pro monitorování síťového provozu, serverových systémů a síťových zařízení vyvíjený firmou Paessler AG.

Server monitorovacího softwaru (označovaný jako core server) běží na operačním systému Microsoft Windows. Obsahuje úložiště dat ve vlastním datovém formátu, webový server a notifikační nástroj. Výstup monitorovacího softwaru zpřístupňuje webové rozhraní.

Server může získávat data odposloucháváním procházejících paketů, protokoly SSH, SNMP, nebo standardy NetFlow, JFlow a sFlow. K tomuto účelu je dostupné velké množství tzv. senzorů, (obdoba pluginů z předešlých systémů). Senzory umožňují monitorování všech běžných síťových služeb jako je protokol HTTP, SNMP, POP3, FTP a mnoho dalších.

Některé senzory vyžadují instalaci klienta na monitorovaném zařízení (označovaného probe), který se stará o získávání dat.

Každý senzor se nachází v jednom z definovaných stavů (Up, Down, Warning, Unusual a další) a dle tohoto stavu a chování je zalogován, případně spuštěn notifikační systém. Ten umožňuje při problému kontaktovat uživatele emailem, SMS nebo telefonním hovorem.

Software také poskytuje funkci autodiscover, nebo-li objevení síťových zařízení vhodných k monitorování (servery Windows, Linux a VMware/XEN a zařízení přístupná protokolem SNMP).

2.5 Zabbix

Zabbix [Zabbix] je propracovaným open-source řešením pro monitorování počítačové sítě, dostupnosti síťových zařízení, jejich služeb a HW prostředků. Pracuje na principu klient-server.

Na klientských zařízeních lze nainstalovat agenty (pro MS Windows, Linux / Unix, Novell), nebo provádět monitorování přes počítačovou síť protokolem SNMP. Do agentů se dodávají vlastní skripty (pluginy), za účelem monitorování v základu nepodporovaných vlastností.

Hlavní výkonná část je napsaná v programovacím jazyce C, datová část podporuje databáze MySQL, SQLite, Oracle a PostgreSQL a webové rozhraní obsluhují PHP skripty.

Webové rozhraní je přehledně strukturované s velkým množstvím různých funkcí a nastavení. Velkou výhodou Zabbixu je možnost veškeré konfigurační soubory spravovat z webového rozhraní a také funkce autodiscover sloužící k objevení aktivních zařízení na síťovém segmentu, či dle IP adresy a běžících služeb.

V případě problémů Zabbix kontaktuje uživatele emailem, nebo formou SMS a zároveň může spustit nakonfigurovanou událost za účelem proaktivního řešení (např. restart služby).

2.6 Torrus

Torrus [Torrus] se snaží být alternativním softwarovým produktem k MRTG (Multi Router Traffic Grapher) [MRTG] nebo Cacti [Cacti]. Umožňuje přizpůsobivost specifickým požadavkům a nabízí vysoký výkon.

Jedná se o serverový produkt napsaný v jazyce Perl, který periodicky shromažďuje data (pooling) protokolem SNMP. Výstup zprostředkovává přes webové rozhraní.

Software obsahuje sadu pluginů, která kromě základních informací o síťovém provozu, zatížení systému a diskovém prostoru zahrnuje také některé proprietární části MIB, jako je například stav UPS, údaje o Cisco QoS a jiné.

Použité pluginy, hierarchie zkoumaných zařízení či perioda vzorkování se konfigurují editací XML souborů.

Jako svou součást poskytuje nástroj devdiscover, který na zkoumaném zařízení dokáže zjistit rozsah poskytovaných funkcí a nabídnout vhodné pluginy pro monitorování.

2.7 Porovnání vlastností SW monitorovacích nástrojů

Vlastnosti jednotlivých softwarových monitorovacích nástrojů jsou srovnány v tabulce 2.1.

Nástroje / Vlastnosti	RMON	NetFlow	Munin	PRTG Network Monitor	Torrus	Zabbix
Architektura	Klient - server	Klient - server	Klient - server	Klient - server	Klient - server	Klient - server
Monitorování síťového provozu	Ano (pouze Eth)	Ano (IP tok)	Ano (ze syst. informací)	Ano (SNMP, NetFlow)	Ano (SNMP)	Ano (SNMP, syst. informací)
Monitorování systémových informací	Ne	Ne	Ano	Ne	Ano (SNMP)	Ano
Pluginy	Ne	Ne	Ano	Ano	Ano	Ano
Notifikace při problému	Ano (pouze SNMP Trap)	Ne	Ano	Ano	Ano (pouze SNMP Trap)	Ano
Programovací jazyk programu	-	-	Perl	-	Perl	C

Tabulka 2.1: Porovnání vlastností SW monitorovacích nástrojů

3 Základní popis systému Nagios

V této kapitole budou uvedeny údaje a principy fungování systému Nagios [Nagios], které byly čerpány ze zdrojů [NagiosDoc], [Barth], [Josephsen] a [Root].

Nagios je software určený pro monitorování počítačové sítě, služeb poskytovaných prvky sítě a zkoumání chování sítě a zařízení v ní. Software je primárně vyvíjen pro operační systém Linux, ale je dostupný i pro systémy s operačním systémem Unix. Obsahuje většinu důležitých vlastností a výhod, které mají softwarové monitorovací nástroje uvedené v kap. 2.

3.1 Verze

Software Nagios dělíme do několika verzí (balíků). Základní provozovanou verzí je Nagios Core (nyní ve verzi 3.3.1). Jedná se o open-source software vydávaný pod licencí GPL verze 2. Tento balík neumí sám monitorování síťových služeb a síťového prostředí, a proto se rozšiřuje o odděleně vyvíjenou část nazvanou Nagios Plugins. Pro enterprise monitorování je vyvíjeno řešení nazvané Nagios XI. Jedná se o komplexnější variantu softwaru postavenou na balíku Nagios Core obsahující řadu rozšíření. Tento software již není pod licencí GNU GPL, ale pro použití v komerční praxi se platí.

V další části této kapitoly se budu zabývat pouze balíkem Nagios Core (jádro), jeho rozšířením Nagios Plugins (pluginy) a popřípadě dalšími možnostmi vylepšení open-source verze softwaru.

3.2 Instalace

Nejmenší funkčí nutností je instalace verze Nagios Core a Nagios Plugins. Instalace se provádí ze zdrojových kódů, nebo balíčků a vyžaduje nejdříve nainstalování webového serveru Apache, interpretu programovacího jazyka PHP, překladače GCC a grafické knihovny GD. Samotný proces instalace krok po kroku popisuje instalační manuál k softwaru Nagios [NagiosInstall].

Před tím, než lze software spustit, vyžaduje editaci (opět dle manuálu) některých konfiguračních souborů (viz kap. 3.7 – Konfigurace). Ovládání (spuštění, restart, zastavení) provádíme dle zvyklostí operačního systému Linux, nejčastěji tedy init skriptem.

3.3 Objekty

Objekty, se kterými Nagios pracuje, jsou následující:

- Zařízení / Skupina zařízení (z angl. Host / Host group)
- Služba / Skupina služeb (z angl. Service / Service group)
- Příkaz (z angl. Command)
- Závislosti (z angl. Dependencies)
- Obeznamování / Skupina obeznamování (z angl. Contact / Contact group)
- Časový údaj (z angl. Time period)

Nagios má vnitřní hierarchické uspořádání. Základní jednotkou je **zařízení** (mohou být sdružována do skupin), reprezentující jeden fyzický stroj. Má jednu nebo několik **služeb** (možno sdružovat do skupin), které spouští **příkazy** se zadanými parametry. Příkaz reprezentuje jeden plugin, kterému tyto parametry předává. Jednotlivá zařízení v systému mohou mít rodiče a dědičnou stromovou strukturu dle propojení v počítačové síti. Ta se specifikuje **závislostmi**.

Dalšími objekty v systému jsou **oznámení** (a jejich skupiny), která určují kdo, jak a za jakých okolností bude Nagiosem informován.

Časový údaj se v systému využívá pro přesnou specifikaci času vhodnou pro intervaly mezi měřeními, nastavení času zasílání oznámení atd.

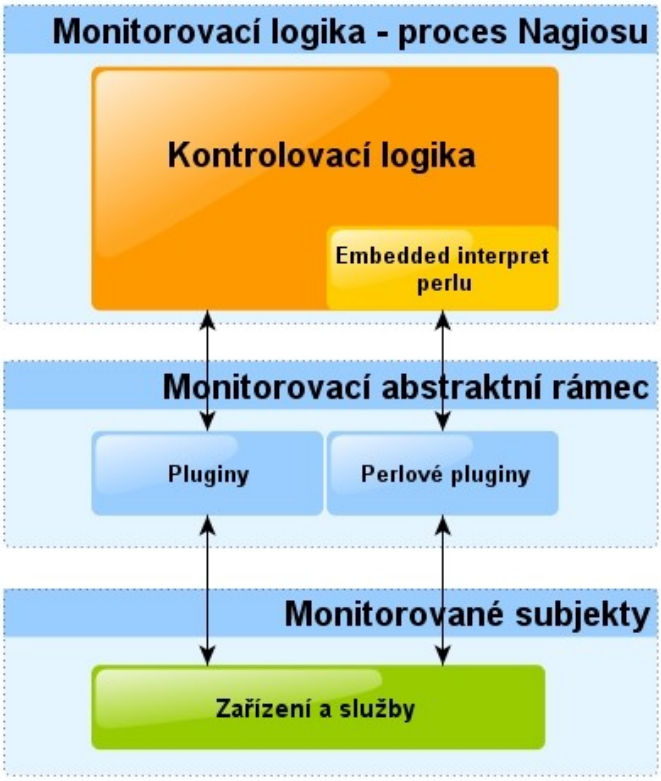
3.4 Monitorovaná zařízení

Software dokáže získávat data z velkého počtu různorodých typů zařízení identifikovatelných IP adresou prostřednictvím počítačové sítě.

Proces monitorování zařízení se v systému Nagios dělí na tzv. **aktivní** a **pasivní**.

3.4.1 Aktivní monitorování

Aktivní kontrola je činnost, kterou vyvolá a provádí přímo proces Nagiosu. Ten spouští skripty označované jako pluginy, které se periodicky dotazují na data. Každý plugin je přímočarý a monitoruje jednu specifickou vlastnost nebo službu. Pluginy nebo-li skripty v libovolném programovacím jazyce, který má nainstalovanou podporu v operačním systému, se spouští z příkazové řádky, kdykoliv se potřebuje zjistit stav služby. Jak můžete vidět na obrázku 3.1, plugin zastupuje vrstvu mezi monitorovací logikou a samotnými zařízeními a jejich službami (více viz kap 3.9 – Princip činnosti).

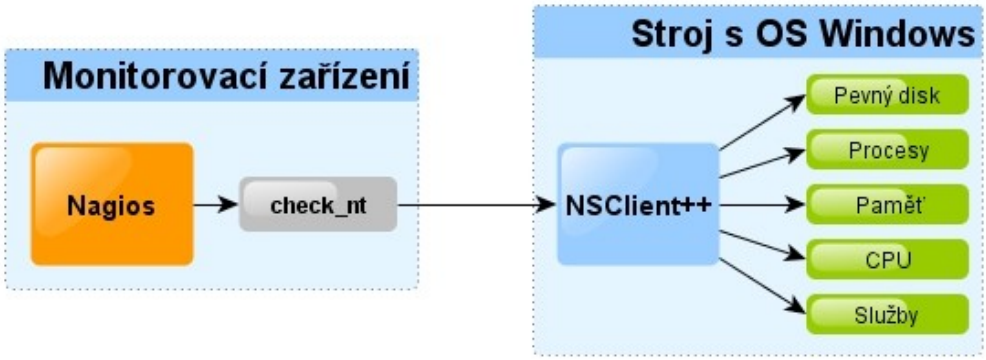


Obrázek 3.1: Vrstvy aktivní monitorovací logiky softwaru Nagios

Nagios dokáže aktivně monitorovat následující zařízení a jejich služby.

Stroje s operačním systémem Microsoft Windows

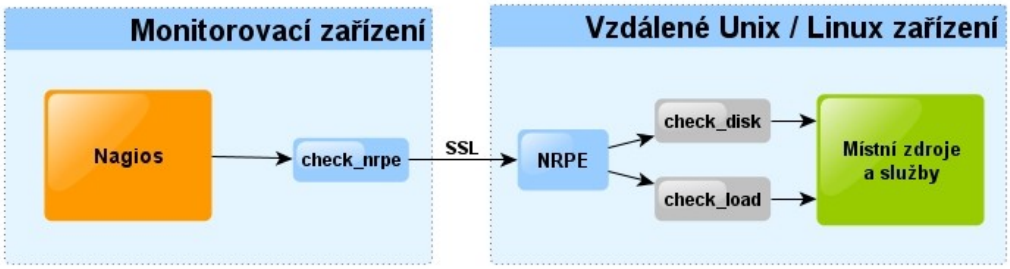
K monitorování informací poskytovaných operačním systémem Windows využívá Nagios program **NSClient++**, který se nainstaluje a nastaví na měřeném stroji a plugin **check_nt**, který je součástí balíku Nagios Plugins. Princip propojení mezi měřícím softwarem Nagios a strojem naznačuje obrázek 3.2.



Obrázek 3.2: Schéma aktivního monitorování stroje s OS Microsoft Windows

Stroje s operačním systémem Linux (Unix)

Monitorování informací poskytovaných stroji s operačním systémem Linux probíhá velice podobně jako s operačním systémem Windows. Na monitorovaném stroji se instaluje rozšíření **NRPE**, ale na rozdíl od softwaru NSClient++ data nezískává přímo, ale spouští na vzdáleném zařízení sadu pluginů, prostřednictvým kterých data shromažďuje. Na stroj s Nagiosem se pak dle obrázku 3.3 data přenesou kanálem zabezpečeným protokolem SSL a zpracují se pluginem **check_nrpe**.

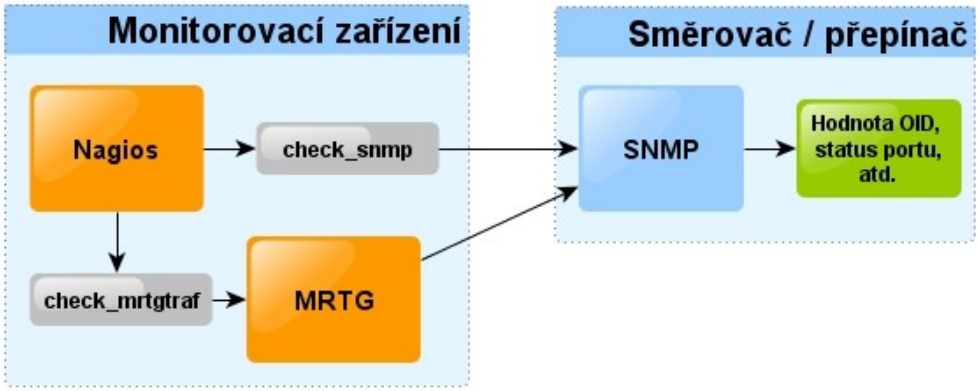


Obrázek 3.3: Schéma aktivního monitorování stroje s OS Linux / Unix

Routery a switche

Informace ze síťových zařízení jako jsou routery a switche se získávají dle obrázku 3.4 protokolem SNMP, který zařízení musí podporovat. Pro tuto činnost balík Nagios Plugins obsahuje plugin **check_snmp**.

Některé získané hodnoty o síťové komunikaci se mohou vykreslovat v podobě grafů MRTG. K tomu slouží plugin **check_mrtgtraf**, který získaná data vyexportuje do grafů.



Obrázek 3.4: Schéma aktivního monitorování routeru / switche

Síťové tiskárny

Nagios umí monitorovat i síťové tiskárny, které jsou dosažitelné buď protokolem SNMP a výše zmíněným pluginem **check_snmp**, nebo specializovanými pluginy pro konkrétní typ tiskárny (například plugin **check_hpdj**). Poskytují informaci například o stavu zásobníků papíru, množství toneru a podobně.

Síťové služby poskytované zařízeními

Jednou z nejdůležitějších vlastností Nagiosu je schopnost monitorovat síťové služby poskytované zařízeními. Kromě základního pluginu **check_ping**, který se používá na všechna síťová zařízení s IP adresou, software nabízí pluginy pro monitorování stavu webového serveru (**check_http**), FTP serveru (**check_ftp**), SSH serveru (**check_ssh**) a mnoho dalších. U většiny těchto pluginů se nemusí kontrolovat pouze funkčnost služby, ale například plugin check_http dovoluje zkontrolovat, zda se na webové stránce vyskytuje zadaný text.

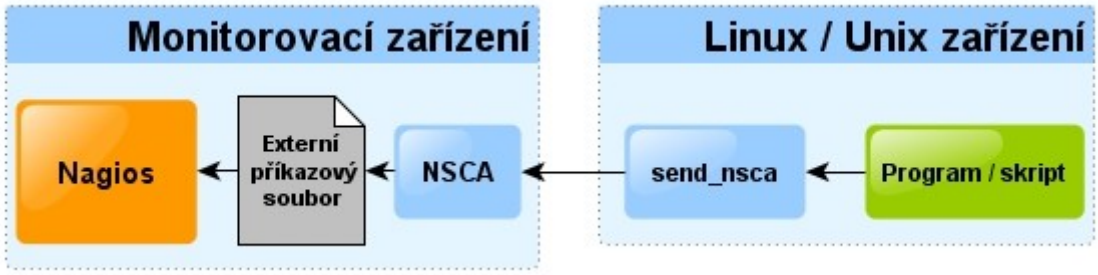
3.4.2 Pasivní monitorování

Pasivní systém kontroly je asynchronní. Hodí se například tam, kde není známa přesná periodicita událostí, nebo kam se aktivně nelze dostat (například kvůli firewallu). Podporu pasivního monitorování je nutno explicitně povolit v konfiguračních souborech softwaru Nagios.

Jedna z možností pasivní kontroly probíhá dle obrázku 3.5. Monitorování je iniciováno externí aplikací s periodou určenou zvnějšku, kterou monitorovací proces Nagiosu neovlivní. Aplikace získaná data uloží do externího souboru, jenž je pravidelně Nagiosem kontrolován a při jehož změně dojde ke zpracování dat klasickou metodou aktivní monitorovací logiky.

Obrázek 3.5: Základní princip pasivní kontroly

Další možností pasivní kontroly je využití rozšíření **NSCA** se strukturou dle obrázku 3.6. Na monitorujícím serveru běží démon a na Linuxové / Unixové stanici běží klient. Démon naslouchá příchozím spojením od klientů, provede validaci dat a zapíše je do externího souboru pravidelně kontrolovaného Nagiosem.



Obrázek 3.6: Pasivní kontrola s využitím rozšíření NSCA

3.5 Vlastnostnosti Nagiosu

Nagios využívá **webové rozhraní** umožňující pohled na aktuální stav sítě, historii stavů prvků sítě a jejich služeb. Dále umožňuje ovládání a nastavení monitorování.

Nagios jako svoji základní součást **umožňuje informovat** správce o nastalém problému. Podporuje notifikaci emailem, pagerem, SMS, IM komunikací, audio výstupem a dalšími způsoby. Umožňuje přesně specifikovat notifikované administrátory dle aktuálního času, čímž lze například rozdělit služby mezi administrátory, nebo informovat pouze toho, který server nebo službu spravuje.

Obsluha událostí (z angl. event handler) je vlastnost určená pro proaktivní řešení vzniklého problému. Jedná se o příkazy (skripty nebo programy) spuštěné při změně stavu služby, dříve než dojde k upozorňování uživatelů notifikačním mechanismem. Skripty nejčastěji provádí restart služby, restart systému, zalogování události do databáze a podobně. Běží s právy uživatele, pod kterým je spuštěn Nagios, a proto by se měl klást velký důraz na bezpečnost (restart systému a služeb je obvykle povolován pouze uživatelům s právy roota).

Nagios dovoluje pro rozložení zatížení uzlů a zvýšení spolehlivosti také několik možností **distribuovaného monitorování**. Populárním systémem pro vyrovnávání zátěže (load-balancing) je DNX (Distributed Nagios eXchange) [DNX]. Ten pracuje na principu klient-server. Server shromažďuje údaje monitorováních klientů, kteří je provádějí. Každý klient pak může paralelně pracovat a serveru se vrací pouze získaná data. Dále lze pro distribuované monitorování použít rozšíření Nagios Fusion [NagiosFusion] nebo MNTOS (Multi-Nagios Tactical Overview System) [MNTOS], které zajišťují sběr dat od klientů na serverovou stanici, jejich agregaci a zobrazování klientům.

Nagios umí vygenerovat přehledné grafické **výstupy** (z angl. reports) s historickou statistikou dostupnosti přímo z webového rozhraní. Prostřednictvím jednoduchého několikakrokového průvodce si uživatel vybere, které zařízení, skupiny zařízení, služby, nebo skupiny služeb zahrnout a jaké časové období zpracovat. Vygenerovaný výstup obsahuje graf s průběhem dostupnosti v čase, s počtem oznámení v čase, nebo dalšími veličinami.

V systému se pamatuje i na **plánování odstávek** (z angl. scheduled downtime). Nastavením časového úseku, po který nemá být dané zařízení monitorováno, lze jednoduše zohlednit odstávku systému, aniž by docházelo k upozorňování nastavených kontaktů nebo ke spouštění obsluhy událostí.

Optimalizaci systému Nagios lze zjišťovat programem **nagiostats**. Umožňuje získávání informací o běžícím procesu Nagios, které zobrazuje v textové podobě nebo v MRTG kompatibilním formátu. Takto získané informace se využívají například pro ověření provedených zlepšení běhu procesu Nagios.

Kromě vzájemného vztahu jednotlivých zařízení daných umístěním v počítačové síti (nejčastěji stromovou strukturou), Nagios umožňuje definovat **vztah mezi službami** (z angl. host / service dependencies) jednoho zařízení nebo mezi službami několika zařízení. Příkladem může být závislost AFS serveru na serveru Kerberos, bez kterého neposkytuje kompletní služby.

Nagios dovoluje vytvářet **vlastní pluginy** monitorující specifickou vlastnost, kterou pluginy v balíku Nagios Plugins nepokrývají. Tato možnost nabízí obrovskou flexibilitu monitorování (více viz kap. 4 – Tvorba pluginů).

Nagios poskytuje také obrovské množství rozšířeních základních softwarových balíků, které lze nalézt v knihovně **Nagios Exchange** [NagiosExchange]. Nachází se zde různé pluginy, sady pluginů, vzhledy webového rozhraní, programy operačních systémů pro správu Nagiosu, konfigurační a vizualizační prostředky a mnoho dalšího.

3.6 Výstup pluginů

Každé zařízení může být dle logiky buď živé (UP), neživé (DOWN), nebo nedostupné (UNREACHABLE).

Každý plugin ale jako svůj výsledek dává čtyři možné hodnoty:

- V pořádku (OK)
- Varování (WARNING)
- Neznámí (UNKNOWN)
- Kritické (CRITICAL)

Mapování mezi stavem zařízení a výstupem pluginu znázorňuje tabulka 3.1.

Výstup pluginu	Stav zařízení
OK	UP
WARNING	UP / DOWN (dle nastavení Nagiosu)
UNKNOWN	DOWN
CRITICAL	DOWN

Tabulka 3.1: Mapování výstupu pluginu na stav zařízení

Nezáleží na tom, jaká veličina je monitorována, protože pluginy do softwaru Nagios vrací jednu z výše uvedených čtyř hodnot a záleží na pluginu, jak stav měřené služby vyhodnotí. Plugin dále také vrací textový řetězec, který upřesňuje stav, v jakém se služba nebo zařízení nachází.

3.7 Konfigurace

Konfigurace softwaru Nagios se nachází v několika textových souborech. Hlavní konfigurační soubor obsahuje obecnou konfiguraci systému, lze v něm specifikovat umístění ostatních konfiguračních a logovacích souborů, jméno uživatele a skupiny, pod kterými systém běží a některé další nastavení. Další konfigurační soubory je pak možno pojmenovávat a vkládat do nich konfiguraci dle osobních zvyklostí.

Jednotlivé soubory obsahují tzv. **konfigurační direktivy** určující objekty systému (zařízení, skupiny zařízení, služby, skupiny služeb, příkazy, kontakty, skupiny kontaktů, časové údaje a další).

Konfigurační systém funguje na principu **šablon**, které jsou děděny a dále rozšiřovány. Proto se například vytváří obecný linuxový stroj, na kterém se monitoruje zatížení CPU a volná disková kapacita a v hierarchii ho rozšířit o podporované služby (HTTP, SSH a další).

Velkou flexibilitu konfigurace zajišťuje také možnost použít tzv. **konfigurační makra** (z angl. macros) v definici některých objektů (kontrola zařízení, služby, upozornění, obsluha události a další). Dovolují odkazovat na informace z nadřazeného zařízení a služby. Systém také kromě velkého množství **předdefinovaných maker** dovoluje definovat **vlastní makra** pro specifické účely. Přehled maker se nalézá na stránce [NagiosMacro].

Konfigurace pro zařízení s jednou službou a využitím maker může vypadat například takto:

```
define host {
      use             linux-server            //využití šablony
      host_name       linuxbox                //název zařízení
      address         192.168.1.2             //IP adresa
}
```

```
define service {
        host_name               linuxbox                //služba patří tomuto zařízení
        service_description     PING                    //popisek služby
        check_command           check_ping!200.0,80%!400.0,40%          //volání příkazu
}

define command{
        command_name            check_ping              //název příkazu
        command_line            $USER1$/check_ping -H $HOSTADDRESS$ -w $ARG1$ -c
                                $ARG2$                  //příkaz a parametry makry
}
```

Veškerá konfigurační nastavení jsou dostupná na stránce [NagiosConfig].

3.8 Adresářová struktura

Software Nagios se vyvíjí pouze pro instalaci na operačních systémech Linux/Unix. Po instalaci jej nalezneme nejčastěji v adresáři **/usr/local/nagios**.

Samotný program (deamon) operačního systému je umístěn v adresáři **/bin**. Nachází se zde také program **nagiostats** (viz kap. 3.5 – Vlastnosti Nagiosu).

V adresáři **/etc** je uložena konfigurace softwaru, kde nalezneme hlavní konfigurační soubor, soubor s uživatelským jménem a zakódovaným heslem pro přístup do webového rozhraní a také konfiguraci CGI skriptů. Dále je zde adresář **/objects**, kam se ukládají konfigurační soubory objektů (zařízení, služby, příkazy, …) potřebných k monitorování.

Monitorovací pluginy se nacházejí v adresáři **/libexec**. V balíku Nagios Plugins jsou pluginy jako binární programy, skripty v jazyce PHP, nebo Perl. Tento adresář je také místem, kam by se měly ukládat další vlastní pluginy.

Aplikační logiku, starající se o činnost software, nalezneme v adresáři **/sbin**. Zde jsou uloženy CGI skripty webového rozhraní například pro tvorbu histogramů, práci s historickými daty, nebo pro zasílání upozornění.

Soubory webového rozhraní systému Nagios jsou uloženy v adresáři **/share**. Jedná se o HTML výstup skriptů napsaných v programovacím jazyce PHP. Vzhled stránky je formátován jazykem CSS.

Proměnné pluginů a historii měření Nagios ukládá do adresáře **/var**. Je to také místo, kam se ukládají logovací soubory a to jak aktuální tak historické. Sem mohou pluginy ukládat dočasné soubory potřebné k jednotlivým měřenín.

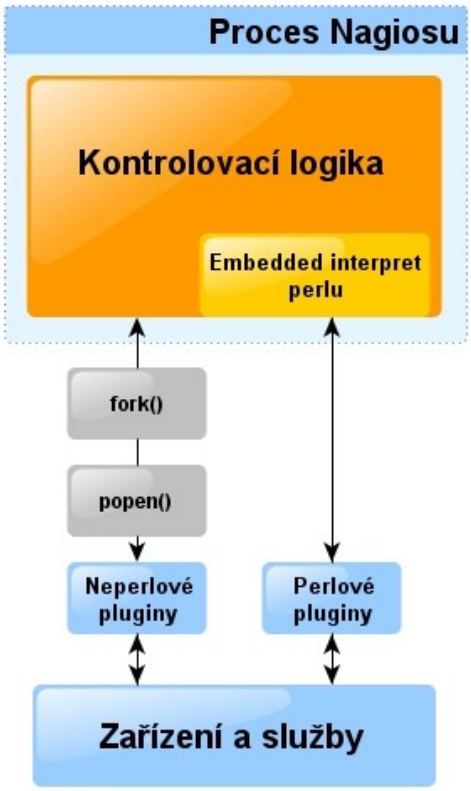
3.9 Princip činnosti

Metoda monitorování softwaru Nagios pracuje na principu **periodického dotazování** dle zadané konfigurace, nebo se kontrola provádí tzv. na vyžádání, když se stroj stane součástí logiky vyhodnocení dostupnosti stroje (viz kap. 3.10 – Vyhodnocování dostupnosti služby a stroje). O správu intervalů mezi dotazováními a o vhodné rozprostření dotazování v čase se stará **plánovač**.

Základní **frekvence dotazování** je jednou za 5 minut (24x7). Tento údaj lze samozřejmě změnit dle potřeby služby a zahrnout do něj plánované odstávky. Plánovač vytváří frontu monitorování, kde je uloženo, která služba se bude kontrolovat a v jakém čase.

Naplánované kontroly zařízení a služeb a kontroly na vyžádání jsou **prováděny paralelně**. To zvyšuje průchodnost systému monitorování. Jak je možno vidět na obrázku 3.7, kontrola (daný plugin) se spouští v novém procesu potomka, který se vytvoří (fork) z hlavního procesu Nagiosu (démona). Když plugin dokončí kontrolu, proces potomka předá rodiči výsledek a ten pak zajistí odpovídající akce (logování, upozornění, obsluha události, …).

Pluginy napsané v programovacím jazyce Perl běží rychleji, pokud využívají Nagios zkompilovaný s **Embedded Interpretem Perlu** (viz kap. 4.4 – Tvorba pluginů s využitím Embedded Perlu).



Obrázek 3.7: Možnosti spouštění pluginů

3.10 Vyhodnocování dostupnosti služby a stroje

Kromě stavů, které vrací pluginy pro každou službu, udržuje Nagios stav, ve kterém se služba aktuálně nachází. Označuje je **HARD / SOFT stav**. Tento stav je důležitou vlastností monitorovací logiky, dle které se určuje zda a která akce bude provedena.

Aby se předešlo falešnému alarmu, Nagios dovoluje definovat kolikrát bude **služba opětovně zkontrolována**, než bude shledán vážný problém. Dle nastavení u dané služby se provede v kratším časovém intervalu několik opakovaných kontrol, dokud se stav nezmění opět na OK, nebo dokud se nevyčerpá počet pokusů.

Služba se po inicializaci nachází ve stavu hard. Pokud poslední kontrola skončila jinak než OK, po dobu opakovaných pokusů se služba nachází ve stavu soft. Pokud se v průběhu opakovaných pokusů získá opět stav OK, je zařízení (služba) ve stavu soft (označováno jako soft recovery). V soft stavu je provedeno logování a obsluha události (event handler), pokud je nadefinována.

Do stavu hard služba přejde, když všechny opakované pokusy skončí jiným stavem než OK, když se vrácená hodnota z pluginu z jednoho chybového stavu změní na jiný (WARNING → CRITICAL), nebo když se probere z chyby ve stavu hard (z angl. hard recovery). V hard stavu je provedeno logování, obsluha události (pokud je nadefinována) a notifikace kontaktů o problému, nebo o probrání.

Nagios ukládá posledních 21 stavů získaných z dané služby. Pokud služba mění stavy příliš často, mluvíme o **skákání stavů** (z angl. state flapping) a zastavují se další notifikace, aby nedošlo k velkému množství posílaných oznámení. Skákání stavů může znamenat špatnou konfiguraci pro danou službu například malým časovým intervalem mezi kontrolami, chybu sítě, nebo jiný problém. Určování, zda-li se změna stavu objevuje příliš často, se provádí dle vzorce:

$$Z = (\text{počet změn} / \text{počet získaných vzorků}) * 100$$

kde: Z ... procentuální poměr změn.

V globální konfiguraci, nebo u každé služby lze nastavit horní a dolní mez, nebo-li kdy už se jedná o skákání stavů a když už ne.

Vyhodnocování dostupnosti stroje pracuje tak, že pokud jsou všechny služby daného zařízení neživé (DOWN), určuje se, zda je zařízení neživé (DOWN) nebo nedosažitelné (UNREACHABLE). To se provádí procházením prvků na cestě od serveru k danému zařízení ve stromové struktuře (dle nastavencých závislostí strojů). Pokud se na cestě narazí na neživé zařízení, jedná se o stav nedosažitelný. Pokud naopak všechny zařízení na cestě žijí, jedná se o stav neživý.

3.11 Nedostatky softwaru Nagios

Webové rozhraní nabízí velkou škálu možností a vysokou provázanost jednotlivých stránek, avšak na úkor **přehlednosti**. Limituje to především začínající uživatele, kteří se snaží naučit s rozhraním pracovat.

Dalším limitujícím faktorem u mnou používané verze je absence možnosti monitorování zařízení s **IPv6 adresou**. Tento nedostatek bude ale jistě s dalšími verzemi opraven.

Omezením pro nasazení Nagiosu může také být nutnost použití pouze **MySQL databáze**, na rozdíl od jiných SW monitorovacích systémů, které nabízejí možnost volby z rozsáhlejší škály databázových systémů.

I přes to, že Nagios nabízí tvorbu shrnujících výstupů, je jeho slabou stránkou, že neposkytuje možnost přímého **exportu dat** z webového rozhraní. Tento export lze realizovat pouze z MySQL databáze.

Při ladění systému Nagios by jistě velice pomohla možnost hledat a **procházet logovací soubory** prostřednictvím webového rozhraní. V současné verzi se činnost realizuje pouze přes textový soubor uložený v adresářové struktuře.

4 Tvorba pluginů

Následující kapitola řeší oblast tvorby pluginů dle zdrojů [PluginAPI] a [PluginDevel]. Uvádí informace pro výběr programovacího jazyka, přesný popis návratových kódů a formátu textové informace. Navíc uvádí postup pro vývoj pluginů s využitím Embedded Perlu a na závěr také správné zásady programování.

4.1 Programovací jazyk

Plugin je skript v libovolném programovacím jazyce umožňujícím vypsat text na standardní výstup. Lze vybrat například C, shell, Perl, Python, PHP a mnoho dalších. Důležité je, aby byla nainstalována podpora tohoto jazyka v operačním systému serveru s Nagiosem.

4.2 Návratový kód pluginu

Plugin jako svůj výstup poskytuje jeden ze čtveřice návratových kódů (částečný popis viz kap. 3.6 – Výstup pluginů). V tabulce 4.1 se nachází podrobný popis návratových stavů.

Číselná hodnota	Stav služby	Popis stavu
0	OK	Plugin byl schopen zkontrolovat službu a zdá se, že funguje správně.
1	WARNING	Plugin byl schopen zkontrolovat službu, ale zdá se, že překročila mez pro varování, nebo že nepracuje správně.
2	CRITICAL	Plugin byl schopen zkontrolovat službu, ale zdá se, že překročila mez pro kritický stav, nebo že nepracuje vůbec.
3	UNKNOWN	Plugin přijal neplatné argumenty příkazové řádky, nebo vznikla nízkoúrovňová chyba (nemožnost otevřít socket, nelze provést fork procesu, ...), což brání spuštění správné operace.

Tabulka 4.1: Podrobný popis návratových stavů

4.3 Textový výstup

Plugin dále poskytuje jako svůj výstup textovou informaci popisující stav dané služby. Jako minimum by měl plugin vypsat jednu řádku textu. S verzí Nagios 3 je možno rozšířit výstup na více řádek a použít tzv. **výkonná data** (z angl. PERFDATA).

Základní formát výstupu:

```
TEXTOVÝ VÝSTUP | VÝKONNÁ DATA
DALŠÍ TEXT – ŘÁDKA 1
DALŠÍ TEXT – ŘÁDKA 2
…
DALŠÍ TEXT – ŘÁDKA N | VÝKONNÁ DATA – ŘÁDKA 2
VÝKONNÁ DATA – ŘÁDKA 3
…
VÝKONNÁ DATA – ŘÁDKA N
```

Povinně se uvádí TEXTOVÝ VÝSTUP. Volitelným výstupem jsou VÝKONNÁ DATA. Zde se nacházejí informace pro zpracování externím programem (například vykreslení dat v podobě grafu MRTG). TEXTOVÝ VÝSTUP rozšiřuje volitelný DALŠÍ TEXT, kam lze uvést podrobné informací.

Příklad využíti dalšího textu a výkonných dat ukazuje výstup obsazenosti souborového systému.

```
DISK OK - free space: / 3326 MB (56%); | /=2643MB;5948;5958;0;5968
/ 15272 MB (77%);
/boot 68 MB (69%);
/home 69357 MB (27%);
/var/log 819 MB (84%); | /boot=68MB;88;93;0;98
/home=69357MB;253404;253409;0;253414
/var/log=818MB;970;975;0;980
```

Základní text se nachází v první řádce po znak '|' a uvádí pouze nejnutnější informaci. Pro další využití tuto část obsahuje makro v systému Nagios nazvané $SERVICEOUTPUT$. Text ve druhé až páté řádce rozšiřuje sdělení o podrobnosti. Tuto část obsahuje makro nazvané $LONGSERVICEOUTPUT$. Výkonný text je uveden v první a od páté řádky za znakem '|' a využívá se jako vstup tvorby grafů programem MRTG. Je uložen v makru $SERVICEPERFDATA$.

Nagios omezuje textový výstup pluginu na 4KB dat. Děje se tak proto, aby se zabránilo MB nebo GB zasílaných dat od pluginů do Nagiosu.

4.4 Tvorba pluginů s použitím embedded perlu

Nagios nabízí vylepšení pro spouštění pluginů napsaných v programovacím jazyce Perl v podobě zkompilovaného Nagiosu s interpretem Embedded Perlu [EPI]. Bez něj jsou spouštěny pluginy jako externí programy, s ním se provádí Perlové pluginy jako knihovní volání. Embedded Perl zavádí některá vylepšení, především pak snížení výpočetní zátěže během monitorování a snížení času stráveného programováním. Pro tvorbu pluginů v Embedded Perlu je však potřeba velmi dobrá znalost tohoto programovacího jazyka, protože nelze použít některé programové konstrukce (jejich seznam s podrobným vysvětlením na [EmbPerlDevel]).

4.5 Zásady programování

Při programování pluginů je vhodné dodržovat několik zásad, které napomáhají správné funkci pluginu a částečně zajišťují bezpečnost.

Pluginy by měly vytvářet dočasné (temp) soubory, pouze pokud se to jeví bezpodmínečně nutným a nelze ukládanou hodnotu získat jinak. Soubory ukládáme vždy do adresáře /var.

Pokud okolnosti donutí ladit funkci Nagiosu a jeho pluginů v konzoli (nejčastěji formát 80x25 znaků), dbáme na to, aby se textová informace z pluginu dala do tohoto prostoru vepsat.

Jako důležité bezpečnostní opatření je nutno validovat všechny vstupy a to jak parametry z příkazové řádky, tak údaje získané pluginem od monitorovaných služeb.

Pluginy využívající programy a příkazy operačního systému musí uvádět z důvodu bezpečnosti při spouštění absolutní cestu k danému binárnímu souboru. Tím se docílí toho, že daný binární soubor nemůže být zaměněn za nebezpečný kód.

4.6 Timeout běhu pluginu

Plugin má velice omezenou dobu běhu, s čímž se musí při programování počítat. Typicky plugin běží maximálně po dobu 10 sekund. Pro pluginy je vhodné, aby při překročení časového limitu reagoval návratovým kódem UNKNOWN s odpovídajícím textovým popisem. Činnost pluginu by tedy neměla zahrnovat dlouhé čekání na data ze síťového spojení nebo zpracování velkého objemu dat.

4.7 Volby příkazové řádky

Každý plugin by měl podporovat alespoň tyto volby příkazové řádky:

- Verbose – Režim, kdy plugin vypisuje podrobně činnost, kterou právě provádí.
- Help – Nápověda jak plugin spustit a jaké jsou jeho parametry a výstupy.
- Version – Vypsání názvu pluginu a jeho verze.
- Timeout – Určení doby čekání například na síťová data.
- Warning – Mez určující stav warning (varování).
- Critical – Mez určující stav critical (kritický).

4.8 Zadávání mezí a rozmezí

Zadávání a parsování mezí a rozmezí pro parametry příkazové řádky warning a critical by mělo být formátováno dle tabulky 4.2.

Definice mezí a rozmezí	Reakce pokud je zkoumaná hodnota
x	< 0 nebo > x (mimo interval <0 ; x>)
x:	< x (mimo interval <x ; ∞>)
~:x	> x (mimo interval <-∞ ; x>)
x:y	< x nebo > y (mimo interval <x ; y>)
@x:y	≥ x a ≤ y (uvnitř intervalu <x ; y>)

Tabulka 4.2: Definice mezí a rozmezí a jejich vysvětlení

5 Kerio Connect

Jak uvádí [AdminGuide]: „Kerio Connect je následovníkem uspěšné aplikace Kerio MailServer. Kerio Connect představuje multiplatformní moderní poštovni server, který podporuje širokou škálu komunikačních protokolů. Tyto protokoly umožňují využívat libovolné poštovní klienty včetně těch, které jsou součástí mobilních zařízení. Dále přináší možnost přímého přístupu k poštovni schránce přes webové rozhraní.

Kerio Connect ukládá do poštovních schránek různé typy dat. Kromě e-mailových zpráv lze do schránky uložit také kalendáře, poznámky, kontakty a úkoly. S kalendáři a úkoly lze navíc dále pracovat díky možnosti plánování schůzek a úkolů.“

Multiplatformní server (současně ve verzi 7.3.0) můžeme nainstalovat na operační systémy:

- **Mac OS X**
- **Linux / Unix**
- **Microsoft Windows**

Samotná instalace se provádí z instalačních balíčků dle zvyklostí konkrétního systému, nebo ve formě VMware Appliance (Debian Linux s předinstalovaným Kerio Connect).

5.1 Komponenty

Server Kerio Connect se skládá z těchto komponent:

- **Kerio Connect Engine** – Hlavní program (**mailserver**) vykonávající všechny služby a funkce aplikace Kerio Connect. Běží jako služba (MS Windows), nebo deamon (Unix) skrytě na pozadí. Součástí jsou samostatně spouštěné procesy **avserver** a **spamserver**, které obsluhují antivirový plugin resp. antispamový modul SpamAssassin.
- **Kerio Connect Monitor** – Utilita sloužící k ovládání (zastavení / spuštění) a monitorování stavu komponenty Connect Engine. Tato komponenta je dostupná pouze pro OS MS Windows a Mac OS X. Operační systém Linux využívá init skriptu pro zastavení a spuštění daemona.
- **Performance Monitor** – Modul (plugin) dostupný pouze pro MS Windows, který je určen pro sledování výkonu (zatížení) jednotlivých součástí aplikace Kerio Connect.

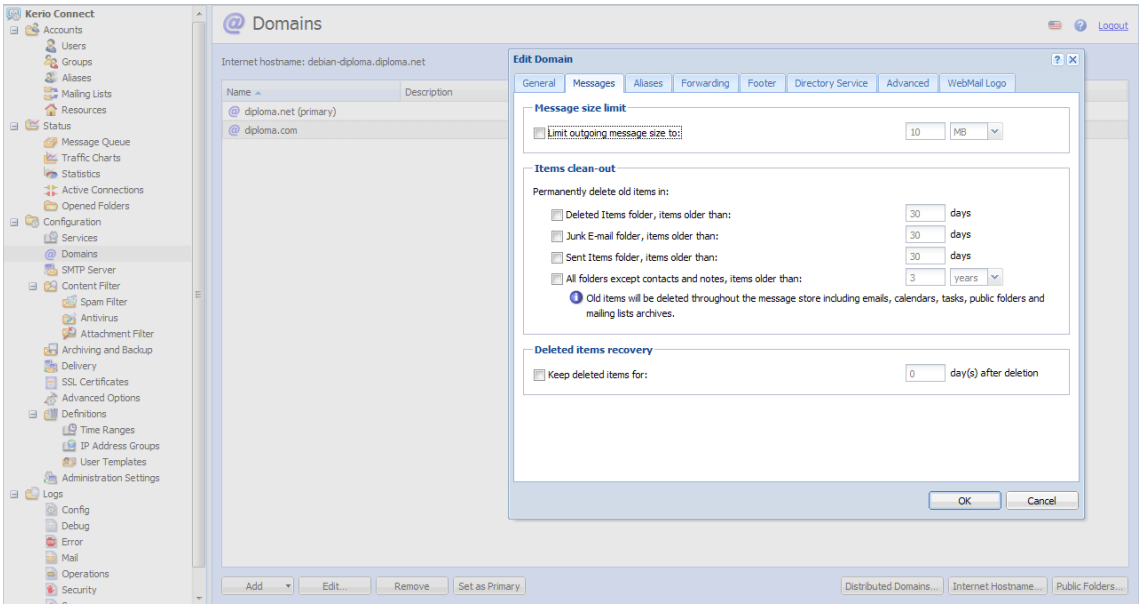
5.2 Webová rozhraní

5.2.1 Kerio WebAdmin

Server Kerio Connect pro svou správu poskytuje webové rozhraní Kerio Connect Administration (WebAdmin).

Administrace serveru se provádí protokolem HTTPS na portu číslo 4040 zadáním URL **https://hostname:4040/admin**. Webový prohlížeč komunikuje se serverem Kerio Connect přes jeho veřejné rozhraní protokolem **JSON RPC** (viz kap. 6.1 – Kerio Connect API). Pro správnou funkci WebAdmin rozhraní musí prohlížeč podporovat JavaScript a CSS.

Jak je možno vidět na obrázku 5.1, webová aplikace se skládá ze sloupce v levé části obsahujícího hlavní nabídku spravovaných vlastností a z části s daty vpravo. Některé údaje se navíc nastavují v samostatných modálních dialogových oknech.



Obrázek 5.1: Rozvržení rozhraní Kerio Connect Administration

Webové rozhraní nabízí veškeré možnosti pro snadnou a úplnou správu serveru. **Spravované vlastnosti** jsou v hlavní nabídce členěny do skupin dle jejich významu. Nastavovat lze domény, uživatelské účty, skupiny, aliasy, veřejné složky, antispamovou a antivirovou kontrolu obsahu, zálohování a mnoho dalšího. Serverem poskytované služby (SMTP, POP3, IMAP a další) lze z administračního rozhraní spravovat a také spouštět a zastavovat jejich běh.

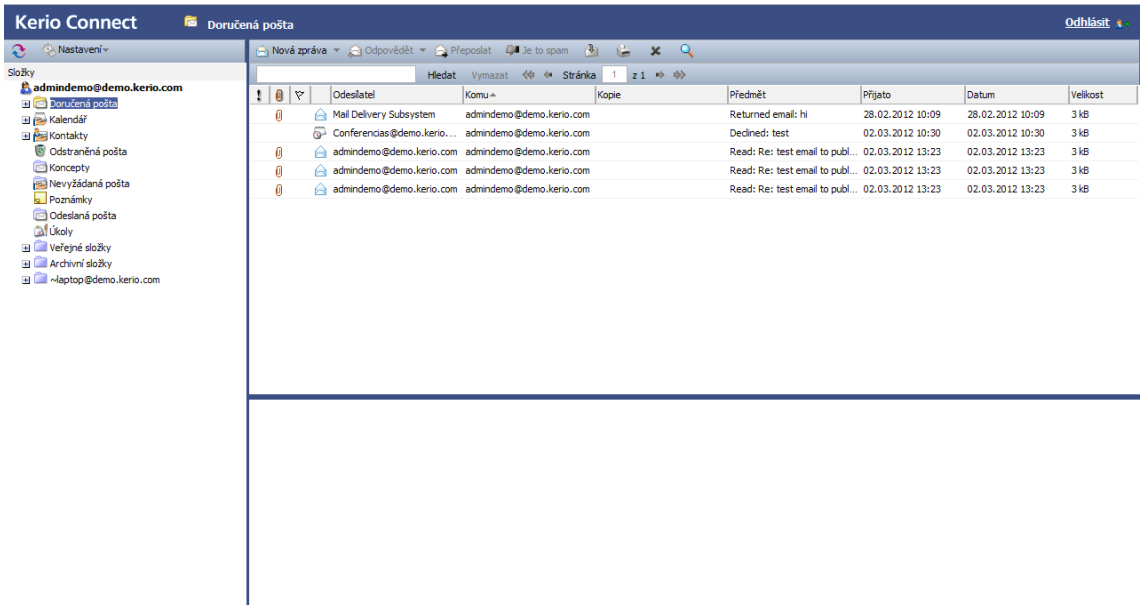
Rozhraní Kerio Connect Administration nabízí také možnosti pro **kontrolu správného chodu serveru a jeho služeb**. Můžeme v něm nalézt grafické i textové výstupy historických statistik serveru (zatížení služeb v čase, počet přijatých emailů obsahujících spam/vir a další), nebo výstupy logovacích souborů (chyby serveru, emailová komunikace, spam, bezpečnost a další).

Na rozdíl od jiných poštovních systémů (zahrnujících server + služby + WebMail), které mají nastavení uloženo v konfiguračních souborech, umožňuje webové administrační rozhraní WebAdmin veškerou správu z jednoho místa a tak usnadňuje administrátorům serverů Kerio Connect jejich činnost.

5.2.2 Kerio WebMail

Častým řešením při správě emailů a groupwarových služeb bývá využití webového prohlížeče. Aplikace Kerio Connect pro tuto činnost nabízí integrované řešení Kerio Webmail. Open-source řešení, založené na samostatném mailovém serveru (např. Postfix) provázaném s externí aplikací (např. SquirrelMail), nemůže nabídnout tak silné provázání a často ani úpravu webového rozhraní v reakci na nové vlastnosti serveru. I proto se většina poskytovalů komplexních poštovních serverů (MS Exchange a další) řídí trendem intergrace a webové rozhraní mají přímo v aplikaci (stejně jako server Kerio Connect).

Kerio Webmail poskytuje uživateli veškeré možnosti pro správu svých poštovních účtů a samozřejmě také využívání groupwarových služeb (poznámky, kontakty, úkoly). Ukázku rozvržení rozhraní Kerio WebMailu můžete vidět na obrázku 5.2.



Obrázek 5.2: Rozvržení rozhraní Kerio WebMail

Při webové správě nezapomíná Kerio Connect ani na starší prohlížeče bez podpory JavaScriptu a CSS. Kromě plnohodnotného WebMailu existuje také verze mini. Ta je určená například pro prohlížeče starších mobilních zařízení nebo pro textové webové prohlížeče. Verze mini nabízí stejnou funkcionalitu jako plnohodnotná verze s možností přizpůsobení dle požadavků uživatele.

5.3 Podporované technologie

Poštovní server Kerio Connect nabízí širokou podporu pro propojení s poštovními klienty a technologiemi různých výrobců a značek.

Lze jej propojit s klienty:

- **Microsoft Outlook** – Server využívá rozhraní **MAPI** (Messaging Apllication Programming Interface) vyvinuté společností Microsoft pro přenos zpráv. Aby mohl klient využít i groupwarových dat, vyžaduje se instalace softwarového rozšíření Kerio Outlook Connector.
- **WebDAV** – [RFC4918] Web Distribution Authoring and Versioning je rozhraní rozšiřující protokol HTTP o možnosti skupinově spravovat a editovat soubory umístěné na serverech. Využívá se pro připojení poštovního klienta MS Entourage (klient sady MS Office 2004 for Mac).
- **Kerio Sync Connector for Mac** – Program instalovaný na klientu sloužící pro obousměrnou synchronizaci kalendářů a kontaktů s programy Apple iCal (program pro správu kalendářů) a Apple Address Book (program pro správu kontaktů). Pro komunikaci využívá rozhraní WebDAV.
- **CalDAV** – [RFC4791] Protokol byl vyvinut jako rozšíření protokolu WebDAV a slouží pro výměnu kalendářových dat. Nejčastěji se používá jako nativní řešení pro program Apple iCal.
- **CardDAV** - [RFC6352] Protokol byl také vyvinut jako rozšíření protokolu WebDAV a má sloužit speciálně pro výměnu kontaktů. Server jím může synchronizovat seznam kontaktů s programem Apple Address Book.
- **ActiveSync** – Protokol slouží pro synchronici emailů, kalendářů a kontaktů. Je založen na protokolu HTTP(S) a hodí se pro komunikaci s mobilními zařízeními na platformě Windows Mobile, Palm OS, Symbian a OS X (např. Apple iPad, Apple iPhone, Samsung Galaxy a další).
- **Kerio Connector for BlackBerry** – Synchronizaci poštovních složek, kalendářů, kontaktů, úkolů a poznámek s mobilními zařízeními BlackBerry zajišťuje modul Kerio Connector for BlackBerry. Ten se instaluje na BlackBerry Enterprise Server a zajišťuje spojení se server Kerio Connect.

5.4 Síťové služby

Kerio Connect v sobě zahrnuje následující síťové služby:

- **SMTP** – Protokol využívaný v Kerio Connect pro odesílání zpráv, pro zpracování příchozích zpráv a pro zprávy doručenované e-mailovými konferencemi. Umožňuje posílat data nešifrovaným, nebo šifrovaným spojením.

- **IMAP** – Poskytuje uživatelům přístup ke zprávám a datům, ale zároveň je ponechává ve složkách na serveru. Toto umožňuje přístup k poště z více míst současně. Také dovoluje s poštou manipulovat ve více složkách.
- **LDAP** – Server umožňující přístup k uživatelským a veřejným adresářům kontaktů. Tento server dovoluje pouze čtení dat, zápis a editace jsou zakázány.
- **HTTP** – Tento protokol se v aplikaci Kerio Connect využívá například pro přístup k uživatelským schránkám přes webové rozhraní (Kerio WebMail), při synchronizaci protokolem ActiveSync, při publikaci kalendářů ve formátu iCal a při dalších činnostech. Jeho zabezpečená verze HTTPS se především používá pro přístup ke správě uživatelských účtů přes webové rozhraní a pro zabezpečený přístup k rozhraní Kerio WebMail.
- **POP3** – Umožňuje uživatelům vybírat (stahovat) zprávy a data ze svých schránek, takže se dále na serveru nenacházejí. Umožňuje práci s poštou pouze v jedné složce.
- **NNTP** – Jedná se o přenosový protokol diskuzních skupin v internetu. Server Kerio Connect umožňuje uživatelům číst zprávy tohoto protokolu a zobrazovat obsah veřejných složek tímto protokolem v internetu. Tento protokol je zastaralý a Kerio Connect jej podporuje především z důvodu zpětné kompatibility.

Každý výše uvedený protokol je serverem Kerio Connect podporován i v jeho zabezpečené verzi (protokoly SSL, nebo TLS).

6 Administrační API pro Kerio Connect

Administrační API pro Kerio Connect je programové rozhraní, kterým lze přistupovat ke správě serveru Kerio Connect. Dovoluje integrovat server s produkty třetích stran, nebo psát skripty k usnadnění rozsáhlých nebo pokročilých administrátorských činností.

API je napsáno tak, aby bylo s minimální námahou přístupné z jakéhokoliv programovacího jazyka. Nosným protokolem je **HTTPS** (SSL zabezpečení) s komunikací probíhající protokolem **JSON RPC** [JSONRPC] s daty ve formátu **JSON** (JavaScript Object Notation) [JSON].

6.1 JSON RPC

Jedná se o bezstavový protokol využívající architekturu klient-server a technologii Remote Procedure Call (vzdálené volání procedur). Je inspirován protokolem XML-RPC [XMLRPC] a vychází z některých jeho vlastností.

Jako formát pro výměnu dat používá **JSON** ve verzi 2.0. Datové typy tohoto formátu jsou:

- **primitivní** (řetězce, čísla, boolean a prázdná hodnota)
- **strukturované** (objekty, pole)

Strukturovaný datový typ objekt představuje neuspořádanou množinu primitivních typů ve tvaru **název:hodnota** (např. {“jmeno“:“jan“, “vek“:42}). Datový typ pole je uspořádaná množina jednoho primitivního datového typu (např. [“karel“, “petr“, “josef“]).

6.1.1 Princip komunikace

Každý klient vysílá na server požadavky (žádost o zpracování metody) a server odpoví zpracovanými daty nebo oznámením o chybě.

Požadavek klienta zahrnuje:

- Textovou specifikaci protokolu (jsonrpc) – verze 2.0 – povinná část
- Název spouštěné metody (method) – povinná část
- Parametry dané metody (params) – u některých metod povinné – ve strukturovaném datovém typu
- Identifikátor (id) – povinná část – slouží pro jednoznačné spárování s odpovědí

Odpověď serveru obsahuje:

- Textovou specifikaci protokolu (jsonrpc) – verze 2.0 – povinná část
- Výsledek metody (result) – data ve formátu strukturovaného typu
- Chyba (error) – vysvětlení vzniklé chyby
- Identifikátor (id) – povinná část - slouží pro jednoznačné spárování s požadavkem

6.1.2 Chyby

Při vzniku chyby se část odpovědi označená výsledek metody nevyskytuje a odesílá se pouze identifikace dané chyby. Ta obsahuje číselný kód chyby, její krátký textový popis a případně další data o chybě. Kódy chyb mohou být systémově předdefinované (celá čísla od -32768 do -32000 určující například neexistenci dané metody, chybně zadané parametry a podobně), nebo určené serverem zpracovávajícím vzdálené volání (ostatní čísla v rozsahu integer).

6.1.3 Ukázka syntaxe

Následují příklady obecné komunikace klienta a serveru protokolem JSON RPC.

Požadavek s **pojmenovanými parametry**:

```
{
     "jsonrpc": "2.0",
     "method": "subtract",
     "params": {
          "subtrahend": 23,
          "minuend": 42
     },
     "id": 3
}
```

Odpověď serveru:

```
{
     "jsonrpc": "2.0",
     "result": 19,
     "id": 3
}
```

Požadavek s **volání neexistující metody**:

```
{
     "jsonrpc": "2.0",
     "method": "foobar",
     "id": "1"
}
```

Odpověď serveru:

```
{
     "jsonrpc": "2.0",
     "error": {
          "code": -32601,
          "message": "Procedure not found."
     },
     "id": "1"
}
```

6.2 Sada funkcí

Rozhraní poskytuje všechny funkce, které jsou dostupné přes webové rozhraní pro správu Kerio Connect Administration (správa uživatelů, domén, nastavení služeb a další). Veškeré funkce dle jednotlivých spravovaných tříd lze nalézt na [KerioClass].

6.3 Činnost API

6.3.1 HTTP komunikace

Komunikace protokolem JSON RPC probíhá požadavky typu POST protokolu HTTPS. URL pro připojení a komunikaci s Kerio Connect přes API je ve tvaru **https://hostname:4040/admin/api/jsonrpc**.

Hlavička zasílaného požadavku musí obsahovat:

```
Content-Type: application/json-rpc
Accept: application/json-rpc
```

To platí při všech činnostech kromě uploadu souboru na server, kdy obsahuje:

```
Content-type: Multipart/form-data.
```

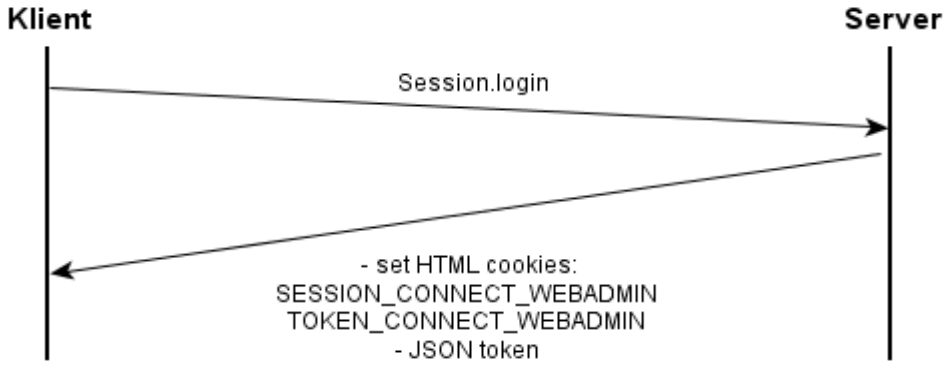
Server odpovídá daty s hlavičkou, která obsahuje:

```
Content-Type: application/json-rpc
```

Server Kerio Connect odpovídá na všechny požadavky standardně HTTP stavovým kódem 200 OK. Jiný kód (např. 4xx, 5xx) se vyskytuje při chybě způsobené na straně serveru (například špatné parsování).

6.3.2 Připojení

HTTP komunikace při připojení probíhá dle obrázku 6.1. Po každém požadavku je spojení uzavřeno (Connection: Close), a proto odpověď po přihlášení obsahuje v HTTP hlavičce příkazy pro nastavení 2 **cookies** a v JSON těle řetězec **token**. Tyto řetězce slouží pro jednoznačné identifikování klienta (více v kap. 6.6 – Bezpečnost) a udržení spojení. Přijatý řetězec token se při výměně dat odesílá od klienta v HTTP hlavičce v položce X-Token společně s nastavenými řetězci cookies.



Obrázek 6.1: Schéma komunikace při připojení klienta

Požadavek s daty ve formátu JSON pro připojení k administračnímu rozhraní volá metodu **Session.login** a vypadá například následovně:

```
POST /admin/api/jsonrpc/ HTTP/1.1
Accept: application/json-rpc
Content-Type: application/json-rpc
Content-Length: 189
Connection: close
Host: 192.168.1.1:4040

{
     "jsonrpc":"2.0",
     "id":1,
     "method":"Session.login",
     "params":{
          "userName":"Admin",
          "password":"pass",
          "application":{
               "name":"check_kerio_storage_occupied",
               "vendor":"Nagios",
               "version":"1.0"
          }
     }
}
```

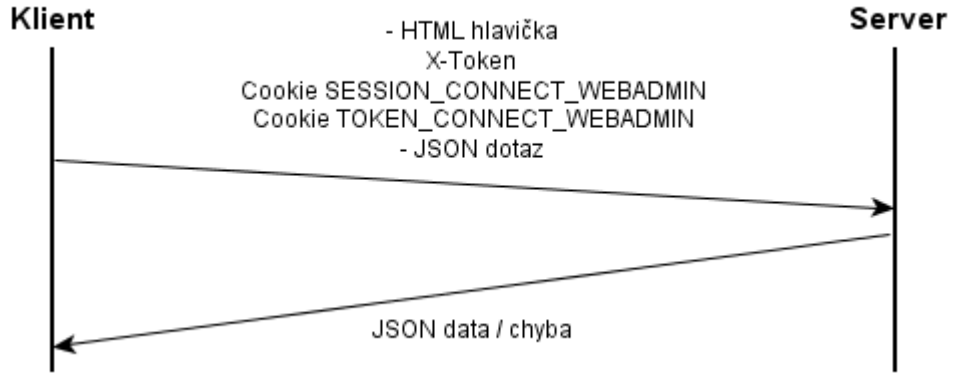
Odpověď serveru vypadá takto:

```
HTTP/1.1 200 OK
Connection: Close
Content-Type: application/json-rpc; charset=utf-8
Date: Wed, 2 May 2012 12:33:00 GMT
Server: Kerio Connect 7.3.2
X-UA-Compatible: IE=edge
Set-Cookie:
SESSION_CONNECT_WEBADMIN=e889081111f8c2dc4111fea6e312217a5fda5c0018304e47
95f32f1b6239fa01; path=/admin/; secure; HttpOnly
Set-Cookie:
TOKEN_CONNECT_WEBADMIN=f8113bf81b50c2d45c3e9392d52812ea767bee19e8d351c270
c2ca8add33f9a5; path=/admin/; secure
```

```
{
     "jsonrpc":"2.0",
     "id":1,
     "result":{
          "token":"f8113bf81b50c2d45c3e9392d52812ea767bee19e8d351c270c2ca
8add33f9a5"
     }
}
```

6.3.3 Výměna dat

Komunikace při výměně dat probíhá dle obrázku 6.2. Požadavky od klienta v HTTP hlavičce obsahují výše zmíněné cookies a token po celou dobu komunikace.



Obrázek 6.2: Schéma komunikace při získávání dat

Zjištění verze produktu (metoda **Server.getProductInfo**) může vypadat takto:

```
POST /admin/api/jsonrpc/ HTTP/1.1
Accept: application/json-rpc
Content-Type: application/json-rpc
Host: 192.168.1.1:4040
Content-Length: 73
Connection: close
X-Token: 48a2a7b4f346f16829c2d13772293cf9e747ba74f75a2c446256080d6e481e87
Cookie:
SESSION_CONNECT_WEBADMIN=bd9479728297a2d0814eec473e110c5ae18f16bf8815a233
f5db3615ea661442;
TOKEN_CONNECT_WEBADMIN=48a2a7b4f346f16829c2d13772293cf9e747ba74f75a2c4462
56080d6e481e87

{
     "jsonrpc": "2.0",
     "id": 1,
     "method": "Server.getProductInfo"
}
```

A odpověď serveru bez chyby (viz kap. 6.4 – Ošetřování chyb) pak následovně:

```
HTTP/1.1 200 OK
Connection: Close
Content-Type: application/json-rpc; charset=utf-8
Date: Wed, 2 May 2012 11:17:27 GMT
Server: Kerio Connect 7.3.2
X-UA-Compatible: IE=edge

{
     "jsonrpc":"2.0",
     "id":1,
     "result":{
          "info":{
               "productName":"Kerio Connect",
               "version":"7.3.2",
               "buildNumber":"6388",
               "osName":"Debian GNU/Linux 6.0.4, x86",
               "os":"Linux",
               "releaseType":"Final",
               "newVersionInfoUrl":"",
               "allowKerioDirectory":false
          }
     }
}
```

6.3.4 Parametry

Některé typy funkcí (jako třeba Users.get) povolují zadání zpřesňujících parametrů metod. Parametry se zapisují strukturovaně ve zjednodušené syntaxy jazyka SQL.

Rozšíření dotazu může mít tuto strukturu:

```
{
     "jsonrpc":"2.0",
     "id":1,
     "method":"Session.login",
     "params":{
          "query": {
               "fields": [
                  "id", "loginName", "fullName", "description"
               ],
               "conditions": [
                  {
                        "fieldName":"QUICKSEARCH",
                        "comparator":"Like",
                        "value":"doe"
                  }
               ],
               "combining":"And",
```

```
                "orderBy": [
                    {
                          "columnName":"loginName",
                          "direction":"Asc"
                    }
                ],
                "start":0,
                "limit":2
            }
        }
}
```

Význam jednotlivých částí:

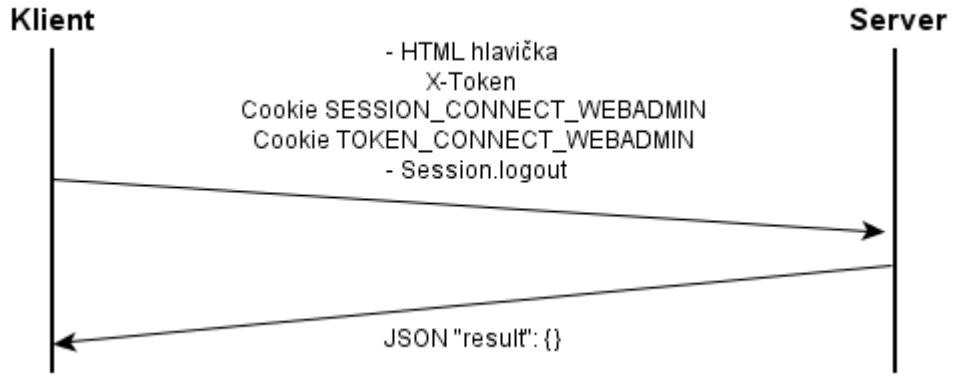
- **Fields** – Omezuje odpověď pouze na vyjmenované sloupce.
- **Conditions** – Budou vráceny pouze objekty vyhovující podmínce (nebo více podmínkám).
- **Combining** – Určuje, zda postačuje platnost jedné ("Or"), nebo všech ("And") podmínek.
- **orderBy** – Vyjmenovává sloupce, dle kterých se budou data v odpovědi řadit a jakým způsobem.
- **start** – Kolik počátečních záznamů bude přeskočeno.
- **limit** – Kolik záznamů bude vráceno. Limit roven -1 znamená bez omezení.

6.3.5 Odpojení

Odpojení od serveru může provést:

- server automaticky po uplynutí doby neaktivity (dle administrace),
- klient vysláním příslušného požadavku (metoda **Session.logout**).

Klientem vyslaný požadavek na odpojení obsahuje v HTTP hlavičce identifikaci spojení (2 cookie, X-Token). Server ukončí sezenení a odpovídá JSON daty s prázdným polem výsledku (z angl. result).



Obrázek 6.3: Schéma komunikace při odpojení klienta

Požadavek může mít následující strukturu:

```
POST /admin/api/jsonrpc/ HTTP/1.1
Accept: application/json-rpc
Content-Type: application/json-rpc
Content-Length: 62
Connection: close
X-Token: f8113bf81b50c2d45c3e9392d52812ea767bee19e8d351c270c2ca8add33f9a5
Cookie:
SESSION_CONNECT_WEBADMIN=e889081111f8c2dc4111fea6e312217a5fda5c0018304e47
95f32f1b6239fa01;
TOKEN_CONNECT_WEBADMIN=f8113bf81b50c2d45c3e9392d52812ea767bee19e8d351c270
c2ca8add33f9a5
Host: 192.168.1.1:4040

{"jsonrpc":"2.0","id":1,"method":"Session.logout","params":[]}
```

Server odpovídá zprávou:

```
HTTP/1.1 200 OK
Connection: Close
Content-Type: application/json-rpc; charset=utf-8
Date: Wed, 2 May 2012 12:33:00 GMT
Server: Kerio Connect 7.3.2
X-UA-Compatible: IE=edge

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

6.4 Ošetřování chyb

Kromě chyb serveru zmíněných výše (kódy od -32768 do -32000) definuje administrační rozhraní také chyby při práci s API (kódy od 1000 do 1999).

Chyba odeslaná v odpovědi serveru může vypadat například takto:

```
"error": {
     "code": 1001,
     "message": "User %1 (%2) already exists .",
     "data": {
          "messageParameters": {
               "positionalParameters": ["jdoe", "Jane Doe"],
               "plurality": 1
          }
     }
}
```

Na programátorovi zpracovávajícím odpověď pak spočívá, aby do zprávy správně vložil parametry, dle jejich pozice. Některé jazyky (např. čeština) mají více tvarů slova podle číslovky, která k nim náleží (např. 1 doména / 2 domény / 10 domén). K rozlišení ve výstupu slouží hodnota **"plurality"**. Nastavení "plurality": 2 bychom využili například u zprávy "If you want to add more than %1 [user|users], contact your Kerio Connect administrator.". Pokud zde poziční parametr %1 bude větší než 1, bude vhodné použít množného čísla a doplnit hodnotu users.

6.5 Částečný úspěch požadavku

Hromadné operace jako je například odebírání, nebo přidávání více uživatelů mohou také skončit chybou. Tuto situaci lze řešit transakčním způsobem a všechny změny odvolat, nebo provést všechny operace a vypsat chybové oznámení o těch nezdařených. Administrační API pro Kerio Connect využívá druhou možnost.

Výstup po hromadném odstranění uživatelů pak může být:

```
{
     "jsonrpc": "2.0",
     "id": "9",
     "result": {
     "errors": [{
          "inputIndex": 3,
          "code": 1002,
          "message": "Failed to delete %1, user was not found.",
          "messageParameters": {
               "positionalParameters": ["jdou"],
               "plurality": 1
          }
     ]}
}
```

Nejdůležitějším prvkem tohoto výpisuje je údaj **"inputIndex"**. Ten určuje, který prvek se nepodařilo smazat. Index je počítán od nuly, což v tomto případě znamená, že ze zamýšlených uživatelů se čtvrtého (jménem "jdou") nepodařilo odstranit.

6.6 Bezpečnost

Při komunikaci aplikací třetích stran s administračním API pro Kerio Connect je třeba dbát její zabezpečení. Při chybném způsobu psaní skriptů a programů využívajících tohoto rozhraním je možno ohrozit server i klienta.

Jedním z nebezpečí je **Cross-site scripting** (XSS) [XSS]. Metoda spočívá v narušení skriptů (nejčastěji webových) chybnou interpretací získaných dat. Data navrácená v odpovědi serveru mohou obsahovat speciální znaky (např. '<'), které musí programátor před zobrazením uživateli ošetřit tak, aby nenarušila bezpečnost skriptu ani vzhled výstupu (např. u webových stránek).

Dalším očekávaným typem ohrožením při práci s rozhraním je metoda **Cross-site request forgery** (CSRF) [CSRF]. Princip spočívá v nechtěném spuštění nebezpečného příkazu na serveru z autorizované klienské aplikace, aniž by o tom uživatel věděl.

Příkladem může být tento kód:

```
<img src="https://victim.example.com:4040/admin/api/deleteUsers">
```

Server poskytující administrační API pro Kerio Connect je proti tomutu typu útoku zabezpečen dvojím způsobem. Jednak povoluje pouze HTTP požadavky typu POST a především pak se při komunikaci vyžaduje v HTTP hlavičce korektní položka X-Token a správně nastavené cookies, které se získávají při přihlašování k serveru (viz kap. 6.3.2 – Připojení).

6.7 Download souborů

Činnost **stahování souboru** probíhá ve dvou krocích. Nejdříve server na základě požadavku (např. metoda Users.exportToCsv) vygeneruje soubor.

Požadavek může vypadat:

```
{
     "jsonrpc": "2.0",
     "id": 1,
     "method": "Users.exportToCsv",
     "params": {
          "query": {
               "fields": [
                  "loginName"
               ],
               "start": 0,
               "limit": "-1",
               "orderBy": [{
                  "columnName": "loginName",
                  "direction": "Asc"
               }]
          },
          "domainId": "keriodb://domain/4baa0a76-1a8c-4887-b56e-
6fe4ff1f4f53",
          "filename": ""
     }
}
```

Server v odpovědi vrátí URL, jméno a velikost souboru:

```
{
     "jsonrpc": "2.0",
     "id": 1,
     "result": {
          "fileDownload": {
               "url":
"http://localhost:4040/admin/api/jsonrpc/download/admin@localhost/4203959aa
d/users_localhost_2010-09-27.csv",
               "name": "users_localhost_2010-09-27.csv",
               "length": 1487
          }
     }
}
```

Poté klient stahuje soubor HTTP požadavkem GET:

```
GET
/admin/api/jsonrpc/download/admin@localhost/4203959aad/users_localhost_20
10-09-27.csv HTTP/1.0
Accept: */*
```

6.8 Upload souborů

Ukládání souboru probíhá HTTP požadavkem POST, kterým se soubor odešle na server:

```
Content-Type: multipart/form-data; boundary=----------------------
1655174106359
Content-Length: 1712 ----------------------------1655174106359

Content-Disposition: form-data; name="csvFile"; filename="users.csv"
Content-Type: application/vnd.ms-excel

Name;FullName;Description;Enable;DataSource;Authentication;Role;Groups;Mail
Address;EmailForwarding;ItemLimit;...
```

Server soubor uloží a vygeneruje mu jedinečné identifikační číslo (id). To vloží do odpovědi a odešle ji zpět klientovi například v této podobě:

```
{
     "jsonrpc": "2.0",
     "id": "1",
     "result": {
          "fileUpload": {
               "id": "73af7c5b28",
               "length": 1487,
               "name": "users.csv"
          }
     }
}
```

Se souborem se na serveru pracuje na základě získaného id. Některé metody (např. Users.parseFromCsv) podporují jako parametr toto identifikační číslo a daný soubor zpracovávají.

7 Sada pluginů

V této kapitole popíši vytvořenou sadu pluginů aplikace Nagios monitorujících chování a stav serveru Kerio Connect.

Sada pluginů se skládá z hlavní třídy a z jednotlivých pluginů. Hlavní třída obsahuje metody usnadňující návrh a programování pluginů.

7.1 Konfigurace

Systém Nagios spouští jednotlivé pluginy na základě jednoho či více konfiguračních souborů (viz kap. 3.7 – Konfigurace). V nich se určuje spouštěný **příkaz** (command - plugin s nastavením argumentů příkazové řádky), **služba** (service – jednu službu může kontrolovat více příkazů) a **zařízení** (host – na jednom zařízení může běžet více služeb).

Pro jeden z pluginů vypadají konfigurační direktivy takto:

```
define host {
     use          linux-server
     host_name    kerio-server
     alias        Kerio connect
     address      192.168.1.1
     _USERNAME    Admin
     _PASS        pass
}
define service {
     use                      local-service
     host_name                kerio-server
     service_description      Check storage percent usage
     check_command            check-kerio-storage-occupied!80!90
}
define command {
     command_name       check-kerio-storage-occupied
     command_line       $USER1$/check_kerio_storage_occupied.php
                        -H=$HOSTADDRESS$ -p=4040 -U=$_HOSTUSERNAME$
                        -P=$_HOSTPASS$ -w=$ARG1$ -c=$ARG2$
}
```

Nastavení příkazu využívá předdefinovaná makra systému Nagios k určení adresářové struktury pluginu ($USER1$), k doplnění adresy zařízení ($HOSTADDRESS$) a pro argumenty příkazu ($ARG1$, $ARG2$). V konfiguraci se vyskytují i uživatelská makra pro zadání jména ($_HOSTUSERNAME$) a hesla ($_HOSTPASS$). Z konfigurace je patrné, že definice služby a zařízení využívá co nejvíce předpřipravné šablony (parametr use) a nastavují se pouze měnící se údaje.

7.2 Hlavní třída

Hlavní řída obstarává tyto činnosti:

- SSL komunikace se serverem Kerio Connect přes jeho veřejné administrační API (přihlášení, odhlášení, výměna dat),
- parsování a zpracování argumentů z příkazové řádky pluginu,
- korektní ukončování pluginu s odpovídajícím výstupem,
- výpis nápovědy a způsobu použití.

7.2.1 Datové struktury

Třída uchovává v asociativních polích následující datové struktury:

- hlavičku a tělo HTTP požadavku,
- identifikaci serveru (jméno nebo IP adresa, číslo portu, uživatelské jméno, heslo),
- informace o pluginu (jméno, verze),
- přepínače a nastavení z příkazové řádky,
- ukončovací status (návratový kód a zpráva).

Dále třída uchovává:

- řetězec s odpovědí serveru,
- povolení upovídaného (verbose) režimu (zapnuto / vypnuto),
- handler pro SSL komunikaci se serverem.

V souboru jsou také definovány konstanty:

- návratové kódy pluginu (OK, WARNING, CRITICAL, UNKNOWN),
- délka timeoutu SSL komunikace,
- velikost bufferu pro příjem odpovědi.

7.2.2 Abstraktní metody

Třída obsahuje abstraktní metody, které pluginy zdědí:

- **setPluginInfo** – Metoda nastavujicí informace o pluginu (jméno a verzi).
- **processSpecificArguments** – Metoda zpracovávající argumenty příkazové řádky nutné pro nastavení běhu konkrétního pluginu (více viz kap. 7.2.4 – Zpracování příkazové řádky).
- **executeCommands** – Metoda obsahující volání metod pluginu, které provedou hlavní činnost.

- **showSpecificUsage** – Metoda sloužící pro vypsání informací o argumentech nutných pro nastavení běhu konkrétního pluginu.
- **showReturnStatus** – Metoda vypisující informaci o významu jednotlivých návratových kódů v daném pluginu.

7.2.3 Spuštění skriptu

Každý plugin se stává potomkem hlavní třídy a k jeho spuštění je třeba vytvořit instanci a zavolat metodu **executePlugin**.

Plugin kontroly využití úložiště dat spouštíme takto:

```
$plugin = new CheckKerioConnectStorageOccupied();
$plugin->executePlugin();
```

Činnost metody probíhá v pěti krocích:

- Volání abstraktní metody pro nastavení jména a verze pluginu,
- spuštění metody zpracovávající parametry zadané z příkazové řádky
- metoda pro přihlášení,
- abstraktní metoda obsahující příkazy (volání metod), které se v pluginu provedou,
- metoda pro odhlášení.

Programový kód této metody vypadá následovně:

```
public function start() {
     $this->setPluginInfo();
     $this->processArguments();
     $this->login();
     $this->executeCommands();
     $this->logout();
}
```

7.2.4 Zpracování příkazové řádky

Každý plugin pro svoji činnost potřebuje alespoň jméno a číslo portu serveru a přihlašovací údaje. Ty získává z příkazové řádky (dle nastavení direktivy command v konfiguraci aplikace Nagios).

Zpracování argumentů z příkazové řádky řídí metoda **processArguments**:

```
private function processArguments() {
     $this->parseArguments();
     $this->processMainArguments();
     $this->processSpecificArguments();
}
```

Všechny zadané údaje (viz kap. 7.2.7 – Přepínače a parametry) se rozparsují do asociativního pole (klíč – hodnota) v metodě **parseArguments**.

Následně se do datové struktury pro identifikaci serveru (asociativní pole server) nastaví výše uvedené nutné údaje v metodě **processMainArguments**. Pokud plugin potřebuje další nastavené údaje z příkazové řádky, využije abstraktní metodu **processSpecificArguments** a hodnoty z pole získá.

7.2.5 Odeslání a příjem dat

Před každým odesláním požadavku na server je nutno zadat volanou metodu a případné její parametry do asociativního pole pro tělo požadavku.

Například pro přihlášení (metoda login):

```
$this->requestBody['method'] = 'Session.login';
$this->requestBody['params'] = array(
          'userName' => $this->server["username"],
          'password' => $this->server["password"],
          'application' => array(
               'name' => $this->plugin["name"],
               'vendor' => "Nagios",
               'version' => $this->plugin["version"]
          )
);
```

Poté je možno zavolat metodu **getDataFromServer**, jejíž tělo vypadá následovně:

```
protected function getDataFromServer() {
     $this->sendRequest();
     $this->receiveResponse();
     return $this->getJSONArrayFromResponse();
}
```

Metoda **sendRequest** funkcí **fsockopen** naváže SSL spojení se serverem Kerio Connect přes jeho administrační API. Poté převede asociativní pole (requestHeader - HTML hlavička, requestBody - tělo se stukturou JSON) na řetězec (funkcí **json_encode**) a odešle ho navázaným spojením funkcí **fwrite**.

Metoda **receiveResponse** čte z navázaného spojení přes přijímací buffer data funkcí **fgets** do řetězce (proměnná response). Po přečtení všech dat metoda ukončí navázané spojení (funkce **fclose**).

V metodě **getJSONArrayFromString** se v přijatém řetězci nalezne pozice začátku dat ve formátu JSON, data se převedou funkcí **json_decode** do asociativního pole a vrátí se k dalšímu zpracování v pluginu.

7.2.6 Ukládání identifikátorů spojení

Server po odeslání dat z metody **login** navrátí v odpovědi identifikátory spojení. Ty je třeba z odpovědi získat a uložit pro další použití. K tomu slouží metody **storeTokenFromResponse** (token ve výsledku JSON dat) a **storeCookiesFromResponse** (požadavek nastavit v HTTP hlavičce cookies SESSION_CONNECT_WEBADMIN a TOKEN_CONNECT_WEBADMIN).

Obě metody naleznou v přijaté odpovědi řetězec určující začátek tokenu, resp. Cookies, a uloží získané identifikátory do datové struktury vytvořené pro HTTP hlavičku požadavku:

```
$this->requestHeader['X-Token:']
$this->requestHeader['Cookie:']
```

Výše uvedený postup umožní pluginu provádět více dotazů bez nutnosti opětovného přihlášení, i když je SSL spojení po přijetí odpovědi vždy ukončeno. Po úspěšném odhlášení od serveru (metoda **logout**) dojde k nastavení těchto položek na prázdný řetězec.

7.2.7 Přepínače a parametry

Každý plugin podporuje přepínače, které podávají informace o jeho použití, nebo mění jeho chování.

Přepínače umožňují:

- Vypsání názvu a verze pluginu (-V, --version) – Po vypsání těchto údajů je plugin ukončen s návratovým kódem UNKNOWN.
- Spuštění verbose ("upovídaného") režimu běhu pluginu (-v, --verbose). Více viz kapitola Verbose režim (viz kap. 7.2.8 – Verbose režim).
- Vypsání způsobu použití (-h, -?-, --help) – Po vypsání těcho údajů je plugin ukončen s návratovým kódem UNKNOWN.

Dále všechny pluginy **vyžadují parametry**:

- Jméno serveru nebo IP adresa (-H, --hostname).
- Číslo portu serveru (-p, --port).
- Uživatelské jméno pro přihlášení k serveru Kerio Connect (-U, --username).
- Heslo pro přihlášení k serveru Kerio Connect (-P, --password).

Volitelným parametrem pak je:

- Nastavení délky čekání na síťové spojení (-t, --timeout) – Pokud tento údaje není zadán, je nastavena předdefinovaná hodnota.

Navíc hlavní třída umožňuje každému pluginu použití dalších parametrů, které pluginy mohou zpracovávat v metodě processSpecificArguments (viz kap. 7.2.4 – Zpracování příkazové řádky).

7.2.8 Verbose režim

Za účelem ladění obsahuje hlavní třída také metody pro tzv. "upovídaný režim". V tomto režimu plugin vypisuje na standardní výstup řetězce předané jako parametr metodě **verbosePrint**. Pokud je verbose režim neaktivní, výstup se po zavolání metody neprovede.

Příklad volání metody verbosePrint:

```
$this->verbosePrint("JSON string parsed to an associative array.");
```

Hlavní třída při zapnutém verbose režimu vypisuje především odesílaná a přijímaná data a také zprávy o činnostech, které provedla (např. zpracování argumentů, odhlášení a další).

7.2.9 Způsob použití

Vypsání způsobu použití (vyvoláno při zadání daného přepínače, špatném formátu vstupů apod.) pluginu obstarává metoda **showUsage**. Ve svém těle volá další 3 metody:

```
private function showUsage() {
     $this->showMainUsage();
     $this->showSpecificUsage();
     $this->showReturnStatus();
}
```

Metoda **showMainUsage** vypisuje text se společným způsobem použití pro všechny pluginy. Zobrazí se tedy informace o tom, jak plugin spustit, jaké parametry nutně zadat (identifikace serveru a uživatele) a jakými přepínači lze výstup pluginu ovlivnit.

Následně se vyvolá abstraktní metoda **showSpecificUsage**. Každý plugin ji zdědí a může ji využít pro vypsání informací o parametrech vhodných pouze pro potřeby daného pluginu.

Nakonec se spouští abstraktní metoda **showReturnStatus**. Plugin vypíše informace o svých návratových kódech. Zobrazí se okolnosti nutné pro ukončení pluginu s určitým kódem (OK, WARNING, CRITICAL, UNKNOWN).

7.2.10 Chybové ukončení

Hlavní třída i všechny pluginy mohou využít metodu **errorExit**. Ta jako svůj parametr očekává textový řetězec popisující chybu zapříčiňující ukončení. Text vypíše na standardní výstup a poté ukončí skript s návratovým kódem UNKNOWN.

7.3 Jednotlivé pluginy

V následující části jsou rozebrány jednotlivé pluginy, jaké vstupy očekávají, jakou činnost provádějí a s jakými návratovými kódy končí.

7.3.1 Plugin Check_kerio_is_alive

Princip činnosti

Přihlášení a odhlášení k a od serveru Kerio Connect přes jeho veřejné rozhraní.

Metody administračního API pro Kerio Connect

- Session.login
- Session.logout

Vstupy

Plugin vyžaduje pouze údaje nutné pro navázání spojení (jméno nebo IP adresa serveru a číslo portu) a přihlášení (uživatelské jméno a heslo).

Návratový kód

- OK – Přihlášení a odhlášení se úspěšně provede.
- UNKNOWN – Došlo k chybě v průběhu skriptu a přihlášení a odhlášení se nezdařilo.

7.3.2 Plugin Check_kerio_alert

Princip činnosti

Plugin získá ze serveru seznam upozornění (z angl. alert list), jehož položky spočítá.

Metody administračního API pro Kerio Connect

- Session.login
- Server.getAlertList
- Session.logout

Vstupy

Plugin vyžaduje pouze údaje nutné pro přihlášení (jméno nebo IP adresa serveru, číslo portu, uživatelské jméno a heslo).

Návratový kód

- OK – Počet položek v seznamu upozornění je roven 0.
- WARNING – Počet položek v seznamu upozornění je větší než 0.
- UNKNOWN – Došlo k chybě v průběhu skriptu.

7.3.3 Plugin Check_kerio_error_log

Princip činnosti

Plugin získá poslední řádek z chybového logovacího souboru (error log) a zkontroluje, zda-li od posledního měření nepřibily nové řádky. Čas posledního řádku porovná s uloženým časem v dočasném textovém souboru v adresáři /usr/local/nagios/var. Poté čas posledního záznamu do tohoto souboru uloží.

Pokud plugin nalezne nový záznam, ukončí svou činnost chybovým stavem WARNING. Nagios naplánuje v průběhu následující periody další kontroly (v základním nastavení 4 kontroly po 1 minutě), ve kterých se snaží chybu ověřit. Při dalších pokusech už ale v chybovém logovacím souboru nové řádky nemusí přibýt, a proto je nutno do dočasného souboru, kam se ukládá čas, přidat také příznak, který zde vydrží po dobu všech opětovných kontrol a ukončí plugin stále se stavem WARNING. Tím se služba po jedné periodě převede do stavu HARD (viz kap. 3.10 – Vyhodnocování dostupnosti stroje) a mohou začít notifikační postupy.

Metody administračního API pro Kerio Connect

- Session.login
- Logs.get (parametry dotazu "logName" = "error", "fromLine" = -1, "countLines" = 1)
- Session.logout

Vstupy

Plugin vyžaduje nutné údaje pro navázání spojení a přihlášení k serveru Kerio Connect a navíc je možno zadat kolikrát Nagios opakuje během jedné periody ověření chybového výstupu (parametr příkazové řádky -a, --attempts). Tento údaj lze vložit do konfigurace makrem Nagiosu s názvem $MAXSERVICEATTEMPTS$.

Návratový kód

- OK – Žádné řádky v logovacím souboru nepřibyly.
- WARNING – V logovacím souboru se objevily nové řádky.
- UNKNOWN – Došlo k chybě v průběhu skriptu.

7.3.4 Plugin Check_kerio_queue_total_items

Princip činnosti

Plugin získá počet položek ve frontě a porovná je s limity pro chybové stavy (přednastavené, nebo z příkazové řádky).

Metody administračního API pro Kerio Connect

- Session.login
- Queue.get (položka odpovědi totalItems)
- Session.logout

Vstupy

Plugin vyžaduje nutné údaje pro navázání spojení a přihlášení k serveru Kerio Connect. Dále je třeba zadat číselné limity počtu zpráv, které se musí nacházet ve frontě, aby plugin ukončil svou činnost s návratovým kódem WARNING (parametr příkazové řádky -w, --warning) resp. CRITICAL (parametr příkazové řádky -c, --critical).

Návratový kód

- OK – Počet zpráv ve frontě nedosahuje limitu pro ukončení ve stavu WARNING.
- WARNING – Počet zpráv ve frontě překročil limit pro ukončení ve stavu WARNING.
- CRITICAL – Počet zpráv ve frontě překročil limit pro ukončení ve stavu CRITICAL.
- UNKNOWN – Došlo k chybě v průběhu skriptu.

7.3.5 Plugin Check_kerio_services_not_running

Princip činnosti

Plugin získá ze serveru informace o všech dostupných, nebo pouze o zadaných službách a zjistí, zda-li v daný okamžik běží.

Metody administračního API pro Kerio Connect

- Session.login
- Services.get (položka odpovědi isAlive)
- Session.logout

Vstupy

Plugin vyžaduje nutné údaje pro navázání spojení a přihlášení k serveru. Volitelně lze vyjmenovat názvy služeb, u kterých se bude monitorovat, zda běží (parametr příkazové řádky -s, --service).

Návratový kód

- OK – Všechny procesy běží.
- CRITICAL – Některý proces, nebo více procesů neběží.
- UNKNOWN – Došlo k chybě v průběhu skriptu.

7.3.6 Plugin Check_kerio_storage_occupied

Princip činnosti

Plugin získá ze serveru jeho statistiku a z ní vybere údaj o procentuálním využití diskového úložiště. Tento údaj porovná s limity pro chybové stavy (přednastavené, nebo zadané z příkazové řádky).

Metody administračního API pro Kerio Connect

- Session.login
- Statistics.get (pole odpovědi storage – položka percentage)
- Session.logout

Vstupy

Plugin vyžaduje nutné údaje pro navázání spojení a přihlášení k serveru Kerio Connect. Dále je třeba zadat číselné limity procentuálního využití úložiště, které jsou nutné pro ukončení pluginu s návratovým kódem WARNING (parametr příkazové řádky -w, --warning) resp. CRITICAL (parametr příkazové řádky -c, --critical).

Návratový kód

- OK – Procentuální využití diskového úložiště je pod limitem pro stav WARNING.
- WARNING – Procentuální využití diskového úložiště překročilo limit pro stav WARNING.
- CRITICAL – Procentuální využití diskového úložiště překročilo limit pro stav CRITICAL.
- UNKNOWN – Došlo k chybě v průběhu skriptu.

8 Ověření řešení

V rámci ověření funkčnosti naprogramované sady pluginů bylo provedeno testování. Pluginy monitorovaly ze serveru se systémem Nagios veličiny tří serverů (jeden ve stejné síti, dva v laboratoři společnosti Kerio Technologies) při měnícím se zatížení procesoru. Jejich parametry obsahuje tabulka B.1 v příloze. Regulaci zatížení obstarával program cpuburn [CPUBURN], který se dle nastavených parametrů snaží maximálně využít procesor.

8.1 Měřená data

Systém Nagios si naměřené údaje z jednotlivých strojů uchovává pro zobrazení ve webovém rozhraní v textovém souboru **status.dat**.

Z něj byla pro každý plugin ze sady a také pro testovací plugin získána **reakční doba (z angl. latency)** udávající rozdíl mezi časem, na který bylo naplánovo spuštění pluginu a časem, kdy byl plugin skutečně spuštěn. Tato hodnota by se dle předpokladů s rostoucím zatížením stroje s Nagiosem měla zvyšovat, protože systém nestihne spustit skript dle určení.

Dalším údajem získávaným ze souboru byla **doba zpracování (z angl. execution time)**. Ta určuje, jak dlouho trval běh skriptu od jeho skutečného spuštění do návratu stavového kódu a textové informace. Opět lze předpokládat, že doba zpracování bude s rostoucím zatížením stroje s Nagiosem stoupat a bude také přímo úměrná množství síťové komunikace strojů.

Zatížení stroje s Nagiosem bylo měřeno jednoduchým přečtením hodnot o loadu ze souboru **/proc/loadavg**. Zde se nachází **průměrné zatížení systému** za poslední 1 minutu, 5 minut a 15 minut a také **počet spuštěných procesů**.

Zatížení strojů se serverem Kerio Connect bylo z bezpečnostních důvodů problém měřit přímo čtením souboru loadavg. Proto jsem využil softwarového rozšíření **NRPE** (viz kap. 3.4.1 – Stroje s operačním systémem Linux (Unix)) a měřil zatížení přes něj. Bohužel však na strojích v laboratořích společnosti Kerio nebylo možno kvůli problému s firewallem uskutečnit síťovou komunikaci se strojem s Nagiosem, tudíž byl tímto způsobem zaznamenáván údaj pouze stroje v lokální síti.

8.2 Měřící plugin

Vlastní sběr dat zajišťoval plugin do systému Nagios (odděděný od Hlavní třídy – viz kap. 7.2). Ten byl spuštěn na každý měřený stroj a sbíral pouze jeho údaje, což zajišťovalo, že se měření bude opakovat s periodou pěti minut. Tato doba odpovídá periodě průměrného zatížení systému ze souboru /proc/loadavg a také času, během kterého postupně proběhne činnost všech pluginů daného zařízení.

Očekávané vstupy

Plugin vyžaduje nutné údaje pro přihlášení k serveru Kerio Connect, i když z něj přímo žádné údaje nezískává. Dále plugin očekává jméno zařízení odpovídající jménu zadanému v konfiguraci Nagiosu. Tento údaj slouží pro vyhledání textových informací o pluginech daného zařízení v logovacím souboru Nagiosu. Zadání argumentu usnadňuje makro systému s názvem $HOSTNAME$.

Princip činnosti

Plugin po spuštění přečte hodnoty zatížení stroje s Nagiosem ze souboru /proc/loadavg a uloží je do objektu. Dále čte ze souboru /usr/local/nagios/var/status.dat reakční dobu a dobu zpracování každého pluginu spuštěného nad zadaným jménem stroje (název pluginu začíná check_kerio_) a ukládá je do struktury objektu. Nakonec skript přečte a uloží do objektu výstup pluginu check_nrpe, který na stroji se serverem Kerio Connect měří zatížení.

Veškeré získané údaje, spolu s časem pořízení, plugin ukládá inkrementálně do CSV souboru uloženého v adresáři /usr/local/nagios/var pojmenovaného dle názvu stroje.

Návratový kód

- OK – Data se podařilo získat a uložit do logovacího souboru.
- UNKNOWN – Během zpracování skriptu se vyskytla chyba.

8.3 Podmínky měření

Bylo provedeno osm měření s různými podmínkami zatížení systémů na stroji se softwarem Nagios a na strojích s poštovním serverem Kerio Connect umístěnými ve stejné síti a v laboratoři společnosti Kerio Technologies.

- **Měření číslo 1** – Probíhalo při nízkém zatížení systémů strojů s Nagiosem i se serverem Kerio Connect – server #1 (provedeno 84 měření).
- **Měření číslo 2** – Bylo prováděno s téměř stoprocentním zatížením systému stroje s Nagiosem programem cpuburn a při slabém vytížení systému stroje se serverem Kerio Connect – server #1 (provedeno 37 měření).

- **Měření číslo 3** – Měřeno bylo při slabém vytížení systému stroje s Nagiosem a téměř stoprocentním vytížení systému stroje se serverem Kerio Connect – server #1 (provedeno 63 měření).
- **Měření číslo 4** - Měření bylo prováděno při současném téměř stoprocentním vytížení systémů strojů s Nagiosem a se serverem Kerio Connect – server #1 (provedeno 118 měření).
- **Měření číslo 5** – Prováděno při nízkém zatížení stroje se softwarem Nagios a nezměřeném zatížení stroje se serverem Kerio Connect – server #2 (provedeno 194 měření).
- **Měření číslo 6** – Měřeno při vysokém zatížení stroje se softwarem Nagios a nezměřeném zatížení stroje se serverem Kerio Connect – server #2 (provedeno 173 měření).
- **Měření číslo 7** - Probíhalo při nízkém zatížení stroje se softwarem Nagios a nezměřeném zatížení stroje se serverem Kerio Connect – server #3 (provedeno 195 měření).
- **Měření číslo 8** – Bylo prováděno při vysokém zatížení stroje se softwarem Nagios a nezměřeném zatížení stroje se serverem Kerio Connect – server #3 (provedeno 174 měření).

8.4 Výsledky měření

Jako statistický ukazatel byl zvolen medián, který má na daném vzorku výhodu v tom, že není ovlivněn extrémními hodnotami.

Porovnání mediánu zatížení systému (parametr systému load) při jednotlivých měřeních uvádí tabulka 8.2. Hodnoty nejsou pro měření nijak zásadní, spíše dokládají, který systém byl, nebo nebyl zatížen. Chybějící údaje u stroje se serverem Kerio Connect jsou způsobeny nemožností uskutečnit síťové spojení rozšíření NRPE s Nagiosem.

Měření	**# 1**	**# 2**	**# 3**	**# 4**	**# 5**	**# 6**	**# 7**	**# 8**
Stroj se softwarem Nagiosem	0,03	1,09	0,03	1,13	0,04	1,12	0,03	1,12
Stroj se serverem Kerio Connect	0	0	1,12	1,12	-	-	-	-

Tabulka 8.2: Medián zatížení systémů při jednotlivých měřeních [bezrozměrná jednotka]

Naměřené velikosti mediánů dob zpracování pluginů v tabulce 8.3 a z nich vytvořené grafy C.1, C.3 a C.5 v příloze dokládají předpoklad, že doba zpracování poroste se zvyšujícím se zatížením systému. Nejmenší je při malém zatížení obou systémů, nejvyšší pak při jejich souběžném zatížení. Vyšší hodnoty dob zpracování u měření číslo 5 až 8 jsou způsobeny síťovým zpožděním při komunikaci s Administračním API pro Kerio Connect.

Měření / Plugin	**# 1**	**# 2**	**# 3**	**# 4**	**# 5**	**# 6**	**# 7**	**# 8**
alert	0,2	0,29	0,25	0,37	0,38	0,52	0,38	0,52
queue-total-items	0,2	0,29	0,25	0,36	0,38	0,51	0,38	0,51
services-not-running	0,2	0,32	0,26	0,4	0,39	0,55	0,39	0,55
error-log-new-lines	0,21	0,31	0,25	0,38	0,39	0,53	0,39	0,53
storage-occupied	0,2	0,29	0,25	0,4	0,39	0,53	0,39	0,53
connect-alive	0,17	0,26	0,2	0,31	0,31	0,42	0,31	0,42
plugin-throughput	0,2	0,3	0,24	0,36	0,34	0,46	0,34	0,46

Tabulka 8.3: Mediány dob zpracování pluginů při jednotlivých měřeních [v sekundách]

Naměřené velikosti mediánů reakčních dob pluginů v tabulce 8.4 a z nich vytvořené grafy C.2, C.4 a C.6 v příloze dokládají, že reakční doba oproti předpokladům neroste se zvyšujícím se zatížením systému, ale zůstává přibližně na stejné hodnotě. Jednotlivé reakční doby pluginů ve všech měřeních se od sebe liší pouze v řádu setin sekundy.

Měření / Plugin	# 1	# 2	# 3	# 4	# 5	# 6	# 7	# 8
alert	0,11	0,09	0,1	0,12	0,12	0,15	0,13	0,15
queue-total-items	0,11	0,13	0,11	0,13	0,13	0,15	0,13	0,15
services-not-running	0,1	0,12	0,14	0,13	0,12	0,14	0,12	0,14
error-log-new-lines	0,13	0,12	0,15	0,11	0,12	0,11	0,12	0,11
storage-occupied	0,13	0,1	0,08	0,13	0,12	0,13	0,12	0,13
connect-alive	0,1	0,09	0,09	0,1	0,12	0,14	0,12	0,14
plugin-throughput	0,14	0,15	0,16	0,12	0,14	0,12	0,14	0,12

Tabulka 8.4: Mediány reakčních dob pluginů při jednotlivých měřeních [v sekundách]

8.5 Zhodnocení testování

Předpoklad roustoucí doby zpracování při zvyšujícím se zatížení systémů strojů se softwarem Nagios a se serverem Kerio Connect se potvrdil. Tento jev přisuzuji delšímu čekání pluginů na vstupně / výstupní operace při síťové výměnně dat. Naopak se nepotvrdil předpoklad zvyšující se reakční doby jednotlivých pluginů při zvyšující se zátěži systémů. Pro dosažení očekávaných hodnot by bylo zřejmě nutno zatížit proces Nagiosu dohledem nad větším množstvím zařízení a služeb.

I přes dosažené výsledky nebylo měření doby zpracování pluginů optimální, protože nezohledněňuje aktuální síťové zatížení, které mohlo mít vliv na délku čekání pluginu na data ze serveru. Měření zatížení (load) probíhalo pouze na jednom stroji se serverem Kerio Connect (server #1), ale lze předpokládat, že při měření obou zbývajících strojů by zjištěné výsledky byly potvrzeny.

9 Závěr

Práce je rozdělena do pěti částí. První část seznamuje čtenáře s přehledem softwarových monitorovacích služeb a zdůrazňuje některé jejich zajímavé vlastnosti. Ve druhé části představuje autor softwarový produkt Nagios, který se stal průmyslovým standardem v oblasti monitorování webovými službami a který přebírá většinu výhod systémů z přehledu. Dále jsou objasněny principy programování vlastních pluginů do tohoto systému. Třetí část se věnuje vysvětlení vlastností a služeb poštovního serveru Kerio Connect a především pak popisuje nutné činnosti pro komunikaci s jeho Administračním API. Ve čtvrté části čtenář získá informace o vytvořených pluginech do systému Nagios, které za pomoci výše uvedeného rozhraní monitorují stav poštovního serveru Kerio Connect. V závěrečné části práce autor navrhuje prinicipy ověření vytvořené sady na poštovních serverech a hodnotí výsledky provedených měření.

Sada pluginů napsaná v programovacím jazyce PHP je plně funkční a otestovaná při běžném provozu. Se serverem Kerio Connect komunikuje přes jeho veřejné administrační rozhraní a dle vnitřní logiky vrací údaje do systému Nagios. Pluginy ze sady umožňují změnu konfigurace, takže jejich chování a výstupy může administrátor nastavit a ovlivňovat. Mezi vytvořenými pluginy není v konfiguraci nastavena žádná závislost, i když by podle názoru autora bylo vhodné, aby se nejdříve spouštěl plugin pro otestování fungujícího přihlášení a následně teprve ostatní.

Práce pro mě byla velkým přínosem, jelikož jsem měl měl možnost podrobně prozkoumat podstatu činnosti v praxi velmi používané webové monitorovací služby Nagios a měl jsem možnost zhodnotit fungování poštovního serveru Kerio Connect v reálném prostředí. Z těchto zkušeností jsem si vytvořil vlastní závěry o tom, které hodnoty poštovního serveru jsou vhodné k monitorování a jaké by měly jednotlivé pluginy poskytovat výstupy v závislosti na získaných údajích.

Seznam zkratek

API – Application Programming Interface

AS – Autonomous System

CSRF – Cross-site Request Forgery

CSV – Comma-separated values

GNU GPL – GNU's Not Unix General Public Licence

IMAP – Internet Message Access Protocol

LDAP – Lightweight Directory Access Protocol

MAPI – Messaging Application Programming Interface

MIB – Management Information Base

NNTP – Network News Transfer Protocol

POP3 – Post Office Protocol

RMON – Remote Monitoring

RPC – Remote Procedure Call

SCTP – Stream Control Transmission Protocl

SMTP – Simple Mail Transfer Protocol

SNMP – Simple Network Management Protocol

SPF – Sender Policy Framework

SQL – Stuctured Query Language

SSH – Secure Shell

SSL – Secure Socket Layer

TLS – Transport Layer Security

ToS – Type of Service

XSS – Cross-site Scripting

Literatura

[AdminGuide] *Kerio Connect 7.3 – Příručka administrátora* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <**http://manuals.kerio.com/connect/adminguide/cz/>.**

[Barth] BARTH Wolfgang. *Nagios: System and Network Monitoring.* No Strarch Press, 2006. ISBN 1593270704.

[Cacti] *Cacti - The Complete RRDTool-based Graphing Solution* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.cacti.net/>.

[CPUBURN] *Debian – Details of package cpuburn in squeeze* [online]. 2012, [cit. 2012-05-02]. Dostupné z: <http://packages.debian.org/squeeze/cpuburn>.

[CSRF] *Cross-site request forgery - Wikipedia – the free encyklopedia* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://en.wikipedia.org/wiki/Cross-site_request_forgery>.

[DNX] *Distributed Nagios Executor (DNX)* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://dnx.sourceforge.net/>.

[**EmbPerlDevel**] *Developing Plugins For Use With Embedded Perl* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/3_0/epnplugins.html>.

[**EPI**] *Using The Embedded Perl Interpreter* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/3_0/embeddedperl.html>.

[Josephsen] JOSEPHSEN David. *Building a Monitoring Infrastructure with Nagios.* Prentice Hall, 2007. ISBN 0132236931.

[JSON] *The application/json Media Type for JavaScript Object Notation (JSON)* [online]. 2006, [cit. 2012-04-17]. Dostupné z: <http://www.ietf.org/rfc/rfc4627.txt>.

[JSONRPC] *JSON-RPC 2.0 Specification* [online]. 2010, [cit. 2012-04-17]. Dostupné z: <http://jsonrpc.org/specification>.

[KerioClass] *Administration API for Kerio Connect 7.2 Reference* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://manuals.kerio.com/connect/api/en/reference/index.html>.

[MNTOS] *Nagios Exchange – MNTOS* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://exchange.nagios.org/directory/Addons/Frontends-(GUIs-and-CLIs)/Web-Interfaces/MNTOS/details>.

[MRTG] *MRTG – Tobi Oetiker's MRTG – The Multi Router Traffic Grapher* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://oss.oetiker.ch/mrtg/>.

[Munin] *Munin – Trac* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://munin-

monitoring.org/>.

[Nagios] *Nagios - The Industry Standard in IT Infrastructure Monitoring* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.nagios.org/>.

[NagiosConfig] *Configuration Overview* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/nagioscore/3/en/config.html>.

[NagiosDoc] *Nagios Core Documentation* [online]. 2009, poslední změna: 2010-08-28, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/nagioscore/3/en>.

[NagiosExchange] *Nagios Exchange - The official site for Nagios projects of all kinds - Nagios plugins, addons, documentation, extension, and more* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://exchange.nagios.org/>.

[NagiosFusion] *Nagios – Nagios Fusion* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.nagios.com/products/nagiosfusion>.

[NagiosInstall] *Nagios Quickstart Installation Guide* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/nagioscore/3/en/quickstart.html>.

[NagiosMacro] *Standard Macros in Nagios* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/nagioscore/3/en/macrolist.html>.

[NetFlow] *Cisco IOS NetFlow – Cisco Systems* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html>.

[**PluginAPI**] *Nagios Plugin API* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagios.sourceforge.net/docs/3_0/pluginapi.html>.

[**PluginDevel**] *Nagios plug-in development guidelines* [online]. 2009, [cit. 2012-04-17]. Dostupné z: <http://nagiosplug.sourceforge.net/developer-guidelines.html>.

[PRTG] *PRTG Network Monitor – intuitive network monitor system* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.paessler.com/prtg>.

[RFC4791] *Calendaring Extensions to WebDAV (CalDAV)* [online]. 2007, [cit. 2012-04-17]. Dostupné z: <http://www.ietf.org/rfc/rfc4791.txt >.

[RFC4918] *HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)* [online]. 2007, [cit. 2012-04-17]. Dostupné z: <http://tools.ietf.org/html/rfc4918>.

[RFC6352] *CardDAV: vCard Extensions to Web Distributed Authoring and Versioning (WebDAV)* [online]. 2011, [cit. 2012-04-17]. Dostupné z: <http://www.rfc-editor.org/rfc/rfc6352.txt>.

[RMON] Klaška Luboš. *Svět sítí: Správa počítačových sítí - Principy RMON* [seriál online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.svetsiti.cz/clanek.asp?cid=Principy-RMON-1662000>.

[Root] *Nagios: monitorovanie počítačovej siete – Root.cz* [online]. 2010, [cit. 2012-04-

17]. Dostupné z: <http://www.root.cz/clanky/nagios-monitorovanie-pocitacovej-siete/>.

[RRDTool] *RRDTool – About RRDTool* [online]. 2012, [cit. 2012-05-04]. Dostupné z: <http://oss.oetiker.ch/rrdtool/>.

[SNMP] *Simple Network Management Protocol – Wikipedia, the free encyclopedia* [online]. 2012, [cit. 2012-05-04]. Dostupné z: <http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol>.

[Sophos] *Antivirus, Endpoint, Disk Encryption, Email and Web Security | Sophos* [online]. 2012, [cit. 2012-05-05]. Dostupné z: <**http://www.sophos.com/en-us/**>.

[Torrus] *Torrus project* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://torrus.org/>.

[XMLRPC] *Using Extensible Markup Language-Remote Procedure Calling (XML-RPC) in Blocks Extensible Exchange Protocol (BEEP)* [online]. 2003, [cit. 2012-04-17]. Dostupné z: <http://tools.ietf.org/rfc/rfc3529.txt>.

[XSS] *Cross-site scripting – Wikipedia – the free encyklopedia* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://en.wikipedia.org/wiki/Cross-site_scripting>.

[Zabbix] *Homepage of Zabbix :: An Enterprise-Class Open Source Distributed Monitoring Solution* [online]. 2012, [cit. 2012-04-17]. Dostupné z: <http://www.zabbix.com/>.

A Obsah přiloženého média

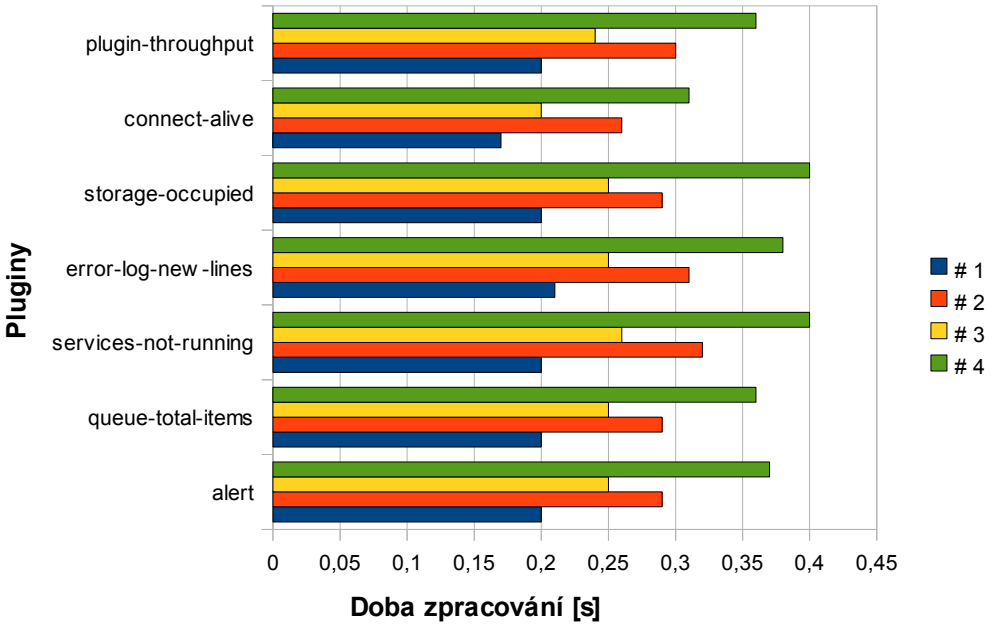
- **pluginy**
  - config – konfigurace softwaru Nagios pro pluginy
    - kerio_connect.cfg
    - line_to_nagios.cfg
  - exec – jednotlivé spustitelné skripty v jazyce PHP
    - check_kerio_is_alive.php
    - check_kerio_storage_occupied.php
    - check_kerio_error_log.php
    - check_kerio_queue_total_items.php
    - check_kerio_alert.php
    - check_kerio_services_not_running.php
  - usage – textové soubory se způsobem užití jednotlivých pluginů
- **testování**
  - plugin
    - config – konfigurace pluginu
      - kerio_connect_plugins_throughput.cfg
      - line_to_nagios.cfg
    - exec – spustitelný skript
      - check_kerio_plugin_throughput.php
    - usage – textový soubor se způsobem užití pluginu
  - výsledky – výsledky provedeného testování
    - csv – výsledky testování ve formátu csv
    - ods – výsledky testování ve formátu ods se statistickým zpracováním dat
  - Kerio Connect config – konfigurace serverů Kerio Connect
- **text práce**
  - dpGruber.odt – zdrojový text diplomové práce
  - dpGruber.pdf – tisknutelná verze diplomové práce

B Parametry testovacích prostředí

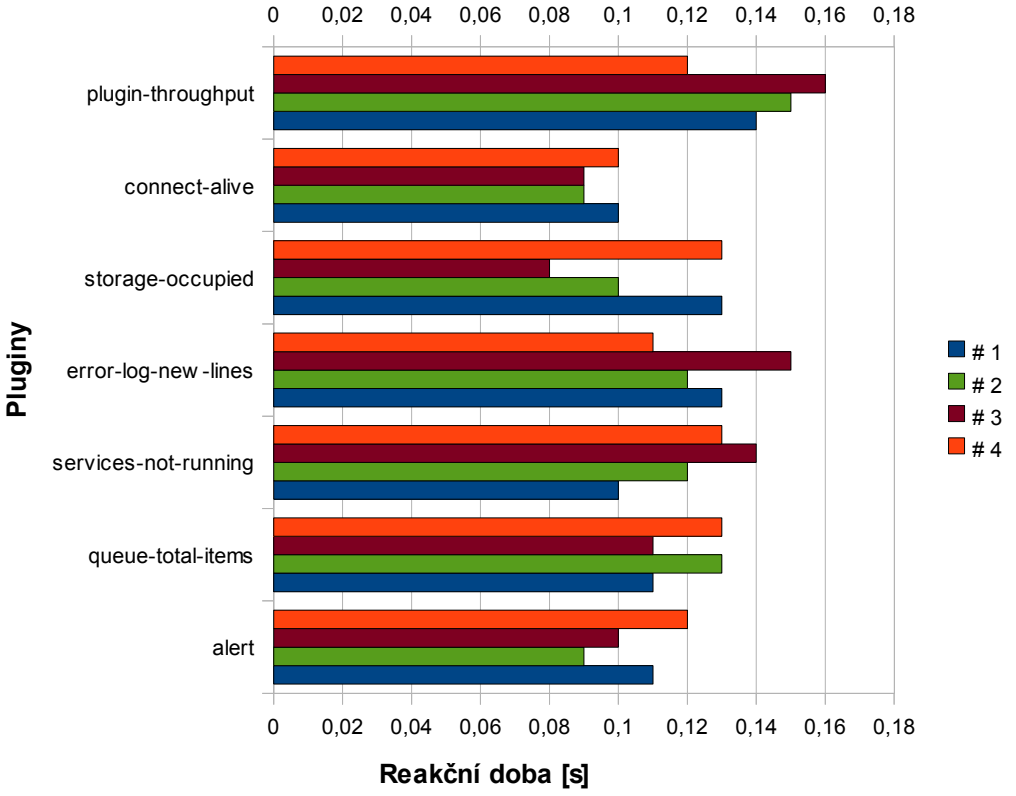
Stroj / Parametr	Server s Nagiosem	Server #1 s Kerio Connect	Server #2 s Kerio Connect	Server #3 s Kerio Connect
Procesor	Intel Core 2 Duo 1,83 GHz (2 jádra)	Intel Core 2 Duo 1,83 GHz (2 jádra)	Intel Xeon 3,73 GHz (2 jádra)	Intel Xeon 3,73 GHz (2 jádra)
Paměť RAM	512 MB	512 MB	1024 MB	1024 MB
Operační systém	Debian Linux 2.6.32-5-686	Debian Linux 2.6.32-5-686	Debian Linux 2.6.32-5-686	Debian Linux 2.6.32-5-686
Verze Kerio Connect	-	7.3.2 build 6388	7.3.0 build 5623	7.3.0 build 5623
Průměrné síťové zpoždění od stroje s Nagiosem	-	1,2 ms	13 ms	13,5 ms

Tabulka B.1: Parametry testovacích prostředí

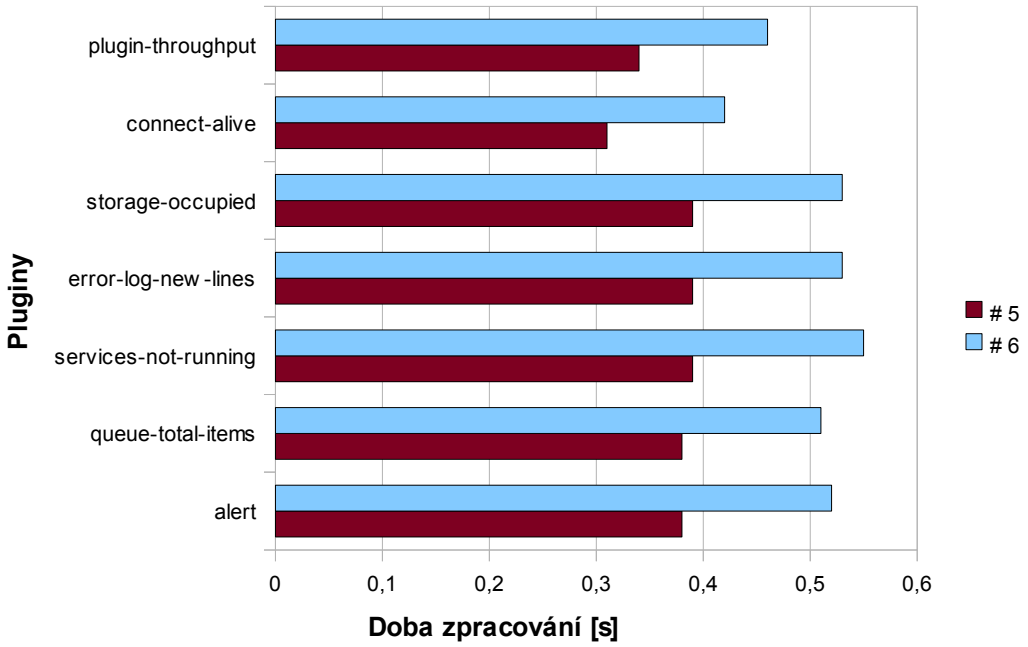
C Grafy výsledků testování



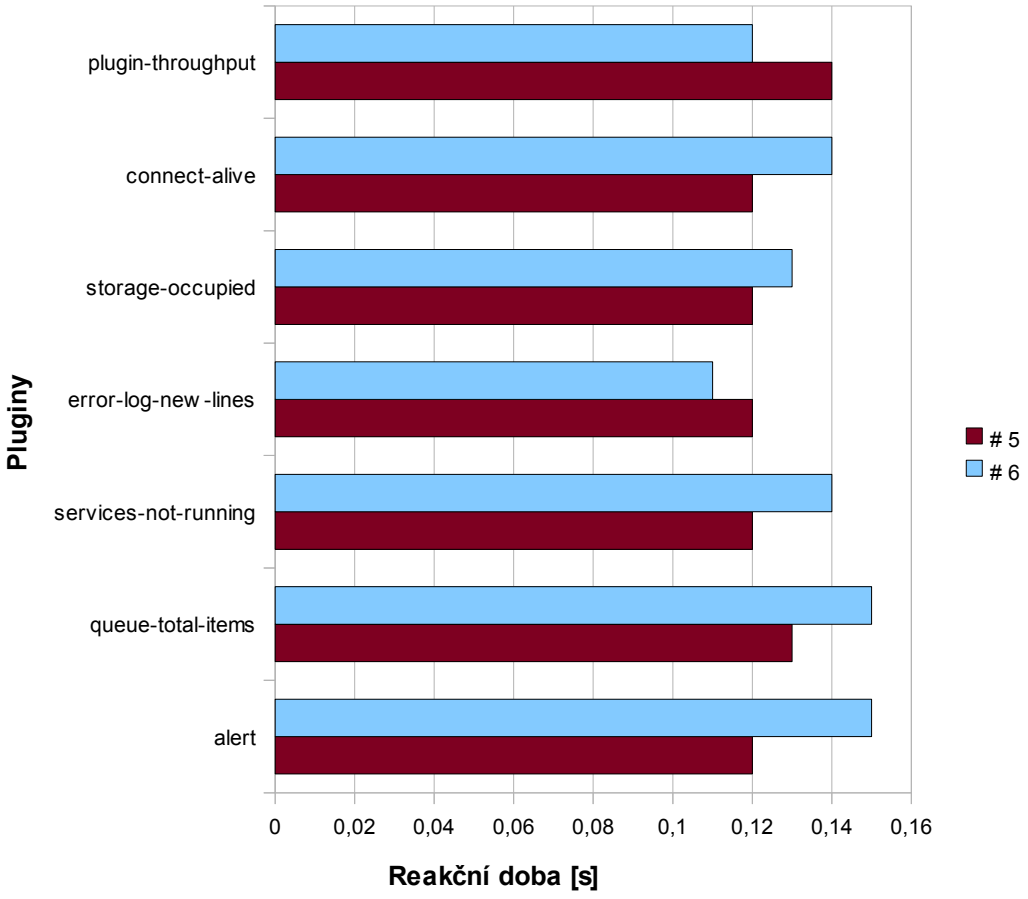
Graf C.1: Porovnání dob zpracování pluginů při měřeních serveru # 1



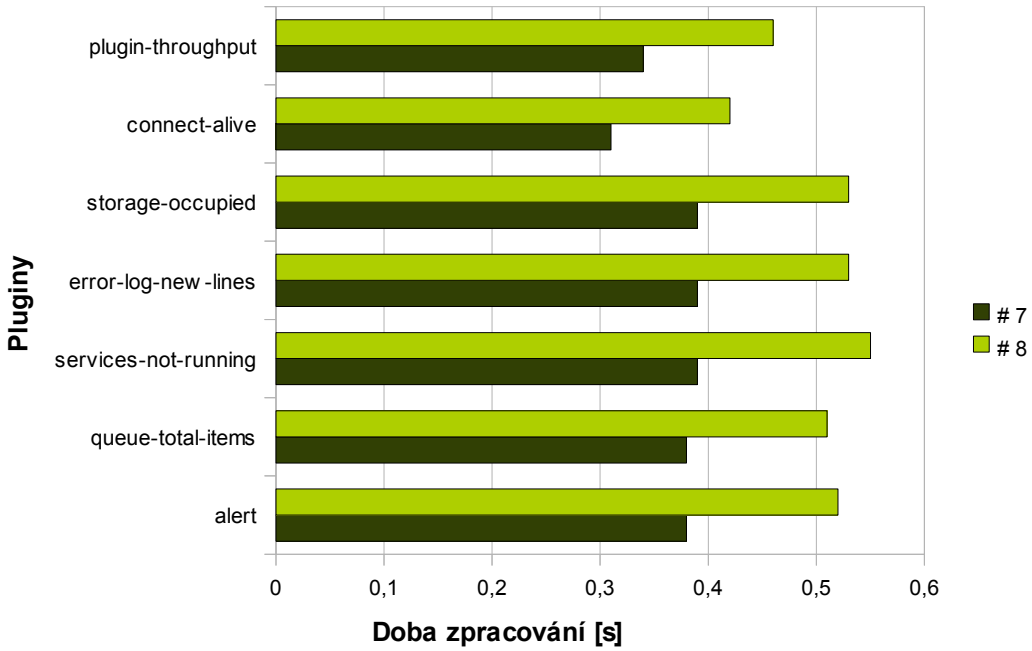
Graf C.2: Porovnání reakčních dob pluginů při měřeních serveru #1



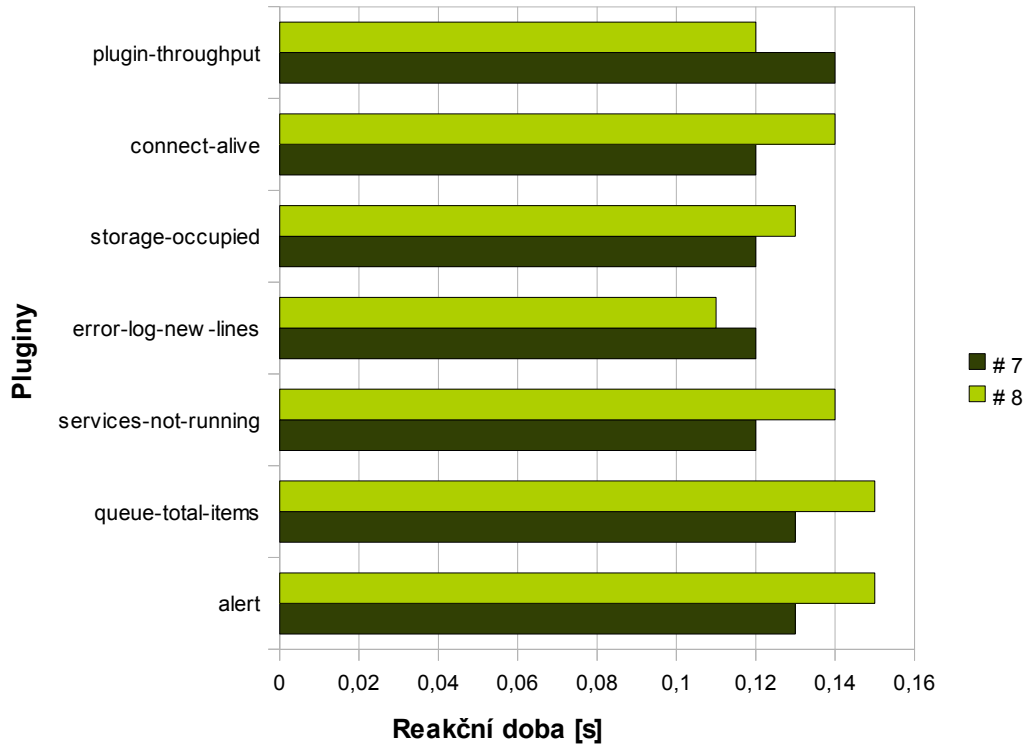
Graf C.3: Porovnání dob zpracování pluginů při měřeních serveru #2



Graf C.4: Porovnání reakčních dob pluginů při měřeních serveru #2



Graf C.5: Porovnání dob zpracování pluginů při měřeních serveru #3



Graf C.6: Porovnání reakčních dob pluginů při měřeních serveru #3

D Postup nasazení

Postup nasazení předpokládá stroj s operačním systémem Debian bez nainstalovaného softwaru Nagios, který má připojení k počítačové síti, ve které se nachází stroj s poštovním serverem Kerio Connect.

D.1 Instalace softwaru

Nejdříve nainstalujeme vyžadované balíčky programem apt-get:

```
apt-get install apache2
apt-get install libapache2-mod-php5
apt-get install build-essential
apt-get install libgd2-xpm-dev
```

Vytvoříme uživatele nagios a přiřadíme mu heslo:

```
/usr/bin/useradd -m -s /bin/bash nagios
passwd nagios
```

Vytvoříme skupinu nagios a přiřadíme do ní uživatele nagios:

```
/usr/bin/groupadd nagios
/usr/bin/usermod -G nagios nagios
```

Stáhneme zdrojový kód Nagios Core a Nagios Plugins (pozn.: URL se může s novými verzemi lišit):

```
wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-
3.2.3.tar.gz
wget http://prdownloads.sourceforge.net/sourceforge/nagiosplug/nagios-
plugins-1.4.11.tar.gz
```

Rozbalíme a nainstalujeme Nagios Core:

```
tar xzf nagios-3.2.3-tar.gz
cd nagios-3.2.3
./configure
make all
make install
make install-init
make install-config
make install-commandmode
```

Nainstalujeme webové rozhraní a nastavíme do něj heslo uživateli nagiosadmin:

```
make install-webconf
htpasswd -c /usr/local/nagios/etc/htpass.user nagiosadmin
/etc/init.d/apache2 reload
```

Rozbalíme a nainstalujeme Nagios Plugins:

```
tar xzf nagios-plugins-1.4.11.-tar.gz
cd nagios-plugins-1.4.11
```

```
./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
make install
```

Po instalaci se soubory softwaru Nagios nachází v adresáři /usr/local/nagios (více viz kap. 3.8 – Adresářová struktura).

D.2 Kopírování souborů

Z přiloženého média zkopírujeme celý obsah adresáře **pluginy/exec** (všechny pluginy) do adresáře **/usr/local/nagios/libexec/**.

Pluginům nastavíme vlastníka a práva:

```
chown nagios.nagios check_kerio_*
chmod 755 check_kerio_*
```

Dále zkopírujeme konfiguraci pluginů **kerio_connect.cfg** z adresáře pluginy/config do adresáře **/usr/local/nagios/etc/objects/**.

D.3 Konfigurace

Zkopírovanou konfiguraci je třeba upravit, aby pluginy komunikovaly s Administračním API serveru Kerio Connect.

V souboru **/usr/local/nagios/etc/objects/kerio_connect.cfg** upravíme komentované řádky:

```
define host {
    use          linux-server
    host_name    kerio-server
    alias        mailserver Kerio Connect      #alias zařízení
    address      192.168.1.1                   #IP adresa zařízení
    _USERNAME    Admin                         #jméno pro připojení k API
    _PASS        pass                          #heslo pro připojení k API
}
```

Dále můžeme nastavit některé parametry (např. warning a critical limity) pluginů:

```
define service {
    use   local-service
    host_name  kerio-server
    service_description Check storage percent usage
    check_command  check-kerio-storage-occupied!80!90
    #vykřičníky oddělují parametr warning a critical limitu
}
```

Bližší informace o parametrech lze získat z textových souborů na médiu v adresáři **pluginy/usage**, nebo spuštěním pluginu s parametrem -h:

```
php check_kerio_storage_occupied.php -h
```

Na závěr do souboru **/usr/local/nagios/etc/nagios.cfg** přidáme řádek:

```
cfg_file=/usr/local/nagios/etc/objects/kerio_connect.cfg
```

D.4 Spuštění Nagiosu

Před spuštěním je třeba zkontrolovat správnost konfiguračních souborů:

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

Pokud se nevyskytne žádná chyba spustíme Nagios:

```
/etc/init.d/nagios start
```

D.5 Provedení testování

Pro provedení testování je třeba zkopírovat skript **check_kerio_plugins_throughput.php** z média z adresáře **testování/plugin/exec** do adresáře **/usr/local/nagios/libexec**.

Pluginu nastavíme vlastníka a práva:

```
chown nagios.nagios check_kerio_plugins_throughput.php
chmod 755 check_kerio_plugins_throughput.php
```

Zkopírujeme soubor **kerio_connect_plugins_throughput.cfg** z adresáře **testování/plugin/config** do adresáře **/usr/local/nagios/etc/objects** a upravíme konfiguraci zařízení obdobně jako v Konfigurace – kap. C.3.

Do souboru **/usr/local/nagios/etc/nagios.cfg** přidáme řádku:

```
cfg_file=/usr/local/nagios/etc/objects/kerio_connect_plugins_throughput.c
fg
```

Zkontrolujeme správnost konfigurace a restartujeme Nagios:

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
/etc/init.d/nagios restart
```

Testovací plugin začne zjištěné údaje ukládat do adresáře **/usr/local/nagios/var** do CSV souboru nazvaného dle položky host_name z konfigurace zařízení.