

**SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ
TECHNOLÓGIE**

**VIRTUALIZÁCIA V SYSTÉME SOLARIS 10
DIPLOMOVÁ PRÁCA**

FCHPT-5414-25600

**SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKÉJ A POTRAVINÁRSKEJ TECHNOLOGIE**

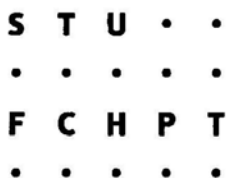
**VIRTUALIZÁCIA V SYSTÉME SOLARIS 10
DIPLOMOVÁ PRÁCA**

FCHPT-5414-25600

Študijný program:	Automatizácia a informatizácia v chémii a potravinárstve
Číslo a názov študijného odboru:	5.2.14 automatizácia
Školiace pracovisko:	Oddelenie informatizácie a riadenia procesov
Vedúci záverečnej práce/školiťel':	Prof. Ing. Miroslav Fikar, DrSc.

Bratislava 2010

Bc. Marián Harajdič



ZADANIE DIPLOMOVEJ PRÁCE

Študent: **Bc. Marián Harajdič**
ID študenta: 25600
Študijný program: automatizácia a informatizácia v chémii a potravinárstve
Študijný odbor: 5.2.14 automatizácia
Vedúci práce: prof. Dr. Ing. Miroslav Fikar
Miesto vypracovania: Bratislava

Názov práce: **Virtualizácia v systéme Solaris 10**

Špecifikácia zadania:

Cieľom práce je implementácia virtualizácie v operačnom systéme Solaris 10 pomocou kontajnerov a zón. Hlavným cieľom je vytvorenie viacerých web a sql serverov na jednom počítači, záťažové testy a porovnanie výkonnosti vzhľadom na nevvirtualizované prostredie.

Úlohy:

- prehľad virtualizačných stratégií
- virtualizácia v OS Solaris 10
- implementácia virtualizácie
- záťažové testy pre HTTP a SQL server

Rozsah práce: 40

Zoznam odbornej literatúry:

1. BAUER, K. *Automating UNIX and Linux Administration*. Berkeley: Apress, 2003. 574 s. ISBN 1-59059-212-3.
2. ŠVEC, P. – FIKAR, M. – DZIVÁK, J. – VOZÁR, L. *Základy práce v operačnom systéme Solaris : Fundamentals of Solaris Operating System*. 2005. ISBN 80-8050-881-X.
3. ROSEBROCK, E. – FILSON, E. *Linux, Apache, MySQL a PHP : Instalace a konfigurace prostředí pro pokročilé webové aplikace*. Praha: Grada, 2004. 191 s. ISBN 80-247-1260-1.

Riešenie zadania práce od: 15. 02. 2010

Dátum odovzdania práce: 21. 05. 2010



Harajdič

Bc. Marián Harajdič

študent

prof. Dr. Ing. Miroslav Fikar

vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc.

garant študijného programu

Pod'akovanie

Ďakujem svojmu školiteľovi prof. Ing. Miroslavovi Fikarovi, DrSc. za vedenie a cenné rady, ktoré mi poskytoval v priebehu vypracovávania diplomovej práce.

Čestné prehlásenie

Čestne prehlasujem, že som diplomovú prácu vypracoval samostatne, podľa pokynov vedúceho práce a s použitím zdrojov uvedených v literatúre.

V Bratislave, 25. apríla 2010

.....

Podpis

Súhrn

Diplomová práca sa zaoberá implementáciou virtualizácie v operačnom systéme Solaris 10, pomocou Solaris kontajnerov a zón. V práci nájdeme vysvetlenie niektorých Solaris technológií, ako sú zóna, kontajner, Fair Share Scheduler a Dynamic Resource Pools. Hlavným cieľom práce je vytvorenie niekoľkých virtuálnych počítačov na jednom fyzickom počítači pomocou kontajnerov a zón, inštalácia web a sql serverov v jednotlivých zónach, testovanie vytvorených serverov záťažovými testami a zhodnotenie výkonnosti serverov na základe získaných výsledkov záťažových testov. Práca slúži ako návod ako postupovať pri vytváraní zón priradením k predvolenému poolu, vytváraní zón priradením k novovytvorenému poolu, administrácií zón systému, klonovaní zón, vymazávaní zón, migrácií zón a inštalácií Apache servera a MySQL servera v jednotlivých zónach systému pomocou nástroja pkgutil. Súčasťou diplomovej práce je testovanie jednotlivých Apache serverov, pomocou nástrojov Apache JMeter a Ab, a MySQL serverov, pomocou nástrojov MySQL5test a The MySQL Benchmark. Získané výsledky potvrdili skutočnosť, že výkonnosť všetkých Apache serverov, respektíve MySQL serverov, je takmer rovnaká. To isté platí o vyťažovaní procesora jednotlivými servermi.

Kľúčové slová: virtualizácia, kontajner, zóna.

Abstract

A diploma thesis is aimed at implementation of virtualization in operating system Solaris 10, using Solaris containers and zones. In the thesis we find explanation of some Solaris technologies, as zone, container, Fair Share Scheduler and Dynamic Resource Pools. Main goal of the thesis is creation of few virtual computers in one physical computer using containers and zones, installation web and sql servers in given zones, testing of created servers by endurance tests and evaluation of servers efficiency based on results of endurance tests. The thesis advices how to proceed by creation of zones by associating to default pool, creation of zones associating to new created pool, administration of system zones, zones cloning, zones deleting, zones migration and installation of Apache server and MySQL server in respective zones of system using pkgutil tool. Part of the thesis is testing of respective Apache servers using Apache JMeter and Ab tools, and MySQL servers using MySQL5test and The MySQL Benchmark tools. Final results confirmed reality, that efficiency of all Apache servers, let us say MySQL servers, is almost equal. The same we can say about used capacity of CPU by servers.

Keywords: Virtualization, Container, Zone.

Obsah

Zoznam príloh.....	11
Zoznam symbolov, skratiek a značiek.....	12
Zoznam ilustrácií.....	13
Zoznam tabuliek.....	15
Slovník	16
Úvod	19
1 TEORETICKÁ ČASŤ.....	21
1.1 Virtualizácia	21
1.1.1 Dôvody prečo pristúpiť k virtualizácii	21
1.1.2 Metódy virtualizácie	23
1.1.3 Simulácia	23
1.1.4 Emulácia	24
1.1.5 Úplná simulácia	24
1.2 Virtualizácia v systéme Solaris 10	24
1.3 Solaris zóny.....	25
1.4 Rozdiel medzi zónami a kontajnermi.....	27
1.5 Fair Share Scheduler (FSS).....	28
1.6 Dynamic Resource Pools	29
1.7 Spôsoby vytvárania Solaris zón	30
1.8 Vytvorenie lokálnej zóny a jej priradenie k novému poolu	31
1.8.1 Vytvorenie poolu	31
1.8.2 Vytvorenie Solaris zóny priradením k novému poolu.....	33
2 PRAKTICKÁ ČASŤ	37
2.1 Vytváranie Solaris zón pridelením k predvolenému poolu.....	38
2.1.1 Aktivovanie resource pools	38
2.1.2 Modifikovanie predvoleného poolu s využitím technológie FSS.....	40
2.1.3 Vytvorenie lokálnej zóny Kirp141	41
2.1.3.1 Konfigurácia lokálnej zóny Kirp141	42
2.1.3.2 Inštalácia a naboťovanie Solaris zóny Kirp141	45
2.1.3.3 Prvé prihlásenie sa do zóny Kirp141	46

2.1.3.4	Rýchle nastavenie lokálnej zóny Kirp141	50
2.1.4	Vytvorenie lokálnej zóny Kirp140	53
2.1.4.1	Vytvorenie zóny Kirp140 priradením k predvolenému poolu.....	53
2.1.4.2	Vymazanie lokálnej zóny Kirp140	53
2.1.4.3	Vytvorenie zóny Kirp140 klonovaním lokálnej zóny Kirp141	55
2.1.5	Aktualizácia systému a inštalácia aplikácií	59
2.1.5.1	Inštalácia a konfigurácia Apache servera	60
2.1.5.2	Inštalácia a konfigurácia podpory PHP pre Apache server	62
2.1.5.3	Inštalácia a konfigurácia MySQL servera	64
2.1.6	Presun lokálnej zóny Kirp141 z globálnej zóny Kirp136 na Kirp138	66
2.2	Zaťažové testy MySQL a Apache servera	75
2.2.1	MySQL5test.....	75
2.2.1.1	Testovanie pomocou MySQL5test	76
2.2.1.2	Test Create	77
2.2.1.3	Test Connect	77
2.2.2	The MySQL Benchmark	78
2.2.2.1	Testovanie pomocou MySQL Benchmark	79
2.2.2.2	Test MySQL servera globálnej zóny pri pozastavení lokálnych zón ..	82
2.2.3	Apache JMeter	86
2.2.3.1	Tvorba testovacieho plánu	86
2.2.3.2	Test zameraný na malé zaťaženie Apache servera	87
2.2.3.3	Test zameraný na vyššie zaťaženie Apache servera.....	89
2.2.3.4	Test zameraný na zaťaženie Apache serverov jednotlivo	91
2.2.4	Ab – Apache HTTP server benchmarking tool	93
2.2.4.1	Testovanie pomocou ab	93
Záver		99
Zoznam použitej literatúry		103

Zoznam príloh

Elektronická verzia diplomovej práce sa nachádza na priloženom CD.

Zoznam symbolov, skratiek a značiek

OS - Operačný systém

VS - Virtuálny server

VM - Virtual machine, virtuálny počítač

MB - Megabit

FSS - Fair Share Scheduler

CPU - Central Processing Unit, procesor

IP - Internet Protocol

HTTP - Hypertext Transport Protocol

SQL - Structured Query Language

PHP - Hypertext Preprocessor

HTML - Hypertext Markup Language

DNS - Domain Name System

SSH - Secure Shell

SFTP - Secure FTP

kB – kilobajt

Zoznam ilustrácií

Obr. 1	Ukážka prerozdelenia zdrojov pomocou FSS	28
Obr. 2	Ukážka globálnej zóny a predvoleného poolu.....	29
Obr. 3	Ukážka pridelenia vytvorenej zóny k novému poolu	30
Obr. 4	Ukážka pridelenia zón k predvolenému poolu	30
Obr. 5	Ukážka dvoch poolov a globálnej zóny.....	32
Obr. 6	Ukážka dvoch poolov a dvoch zón.....	35
Obr. 7	Ukážka dvoch globálnych a troch lokálnych zón.....	37
Obr. 8	Ukážka simulácie.....	37
Obr. 9	Zadanie mena počítača pre zónu Kirp141	48
Obr. 10	Nastavenie bezpečnostnej politiky pre zónu Kirp141	48
Obr. 11	Konfigurácia DNS pre zónu Kirp141	49
Obr. 12	Ukážka lokálnej zóny Kirp141 a globálnej zóny.....	50
Obr. 13	Ukážka lokálnej zóny Kirp141, globálnej zóny a dvoch kontajnerov.....	50
Obr. 14	Ukážka globálnej zóny, lokálnych zón Kirp141 a Kirp140	58
Obr. 15	Ukážka globálnej zóny, 2 lokálnych zón a 3 kontajnerov	58
Obr. 16	Konfigurácia systému po nainštalovaní Apache serverov.....	61
Obr. 17	Test správnosti konfigurácie PHP Kirp136	62
Obr. 18	Test správnosti konfigurácie PHP Kirp140	63
Obr. 19	Test správnosti konfigurácie PHP Kirp141	63
Obr. 20	Konfigurácia systému po nainštalovaní podpory PHP	63
Obr. 21	Konfigurácia systému po nainštalovaní MySQL servera	66
Obr. 22	Verzia PHP v globálnej zóne Kirp138	74
Obr. 23	Verzia PHP v lokálnej zóne Kirp143	74
Obr. 24	Konfigurácia Kirp138, Kirp142, Kirp143	74
Obr. 25	Test výkonu MySQL servera pomocou MySQL5test.....	76
Obr. 26	Test Create.....	77
Obr. 27	Test Connect.....	78

Obr. 28	Porovnanie podľa jednotlivých testov [Kirp136/Kirp140] – dĺžka testu	80
Obr. 29	Porovnanie podľa testov [Kirp136/Kirp140] – vyťaženie CPU	80
Obr. 30	Test rýchlosti pripojenia [Kirp136/Kirp140] – rýchlosť pripojenia	81
Obr. 31	Test rýchlosti pripojenia [Kirp136/Kirp140] – vyťaženie CPU	81
Obr. 32	Test rýchlosti vkladania údajov [Kirp136/Kirp140] – rýchlosť vkladania	82
Obr. 33	Test rýchlosti vkladania údajov [Kirp136/Kirp140] – vyťaženie CPU	82
Obr. 34	Porovnanie výkonu podľa jednotlivých testov [Kirp136] – dĺžka testu	83
Obr. 35	Porovnanie výkonu podľa testov [Kirp136] – vyťaženie CPU	83
Obr. 36	Test rýchlosti pripojenia [Kirp136] – rýchlosť pripojenia	84
Obr. 37	Test rýchlosti pripojenia [Kirp136] – vyťaženie CPU	84
Obr. 38	Test rýchlosti vkladania údajov [Kirp136] – rýchlosť vkladania	85
Obr. 39	Test rýchlosti vkladania údajov [Kirp136] – vyťaženie CPU	85
Obr. 40	Zostavený plán testu	88
Obr. 41	Apache JMeter – malé zaťaženie	88
Obr. 42	Štandardná odchýlka – malé zaťaženie	89
Obr. 43	Odozva servera – malé zaťaženie	89
Obr. 44	Apache JMeter – vyššie zaťaženie	90
Obr. 45	Štandardná odchýlka – vyššie zaťaženie	90
Obr. 46	Odozva servera – vyššie zaťaženie	91
Obr. 47	Apache JMeter – jednotlivé servery samostatne	92
Obr. 48	Štandardná odchýlka – jednotlivé servery samostatne	92
Obr. 49	Odozva servera – jednotlivé servery samostatne	92
Obr. 50	Minimálny čas pripojenia nameraný pomocou ab	94
Obr. 51	Maximálny čas pripojenia nameraný pomocou ab	94
Obr. 52	Stredná hodnota času pripojenia nameraná pomocou ab	95
Obr. 53	Priemerný čas pripojenia nameraný pomocou ab	95
Obr. 54	Počet požiadaviek za sekundu získaný pomocou ab	96
Obr. 55	Súhrny čas súbežných požiadaviek (ab)	97
Obr. 56	Čas jednej požiadavky (ab)	97

Zoznam tabuliek

Tab. 1	Výsledky získané pomocou MySQL Benchmark [Kirp136/Kirp140]	79
Tab. 2	Test rýchlosti pripojenia [Kirp136/Kirp140]	80
Tab. 3	Test rýchlosti vkladania údajov [Kirp136/Kirp140]	81
Tab. 4	Test MySQL servera globálnej zóny pri pozastavení lokálnych zón	83
Tab. 5	Test rýchlosti pripojenia [Kirp136]	84
Tab. 6	Porovnanie rýchlosti vkladania údajov [Kirp136]	85
Tab. 7	Apache JMeter test – malé zaťaženie	88
Tab. 8	Apache JMeter test – vyššie zaťaženie	90
Tab. 9	Apache JMeter test – jednotlivé servery samostatne	91
Tab. 10	Minimálny/maximálny čas pripojenia nameraný pomocou (ab)	94
Tab. 11	Čas pripojenia nameraný pomocou (ab)	95
Tab. 12	Dĺžka testu/počet požiadaviek za sekundu (ab)	96
Tab. 13	Súhrny čas súbežných požiadaviek (ab)	96
Tab. 14	Čas jednej požiadavky (ab)	97

Slovník

Operačný systém - základný program každého počítača.

Fyzický počítač - zariadenie pozostávajúce z mechanických a elektronických častí, ide o hardvér počítača.

Virtualizačná vrstva - jej úlohou je zabezpečiť to, že softvér, ktorý beží v rámci virtuálneho počítača, funguje tak isto ako softvér bežiaci v rámci fyzického počítača.

Virtuálny počítač - replika fyzického počítača poskytujúca potrebné prostriedky na beh aplikácií.

Superužívateľ - je špeciálny užívateľský účet pre správu celého systému v praxi často označovaný ako root.

Jadro operačného systému (kernel) - základný komponent každého operačného systému, ktorého hlavnou úlohou je komunikácia s vyššími vrstvami a programami.

Solaris zóny - ich úlohou je vytvoriť systém, v ktorom sa aplikácie bežiacie v rámci virtuálnych počítačov správajú po spustení, ako keby boli spustené na rôznych fyzicky oddelených počítačoch.

Solaris kontajnery - ich podstatou je rozkúskovanie pôvodnej inštancie operačného systému Solaris 10 na veľké množstvo virtuálnych inštancií operačného systému.

Globálna zóna - jediná zóna systému, ktorá má prístup k hardvéru systému.

Lokálne zóny - všetky zóny, ktoré sú vytvorené vo vnútri globálnej zóny.

Fair Share Scheduler - technológia využívaná na pridelenie podielov z dostupných zdrojov medzi jednotlivé zóny.

Dynamic Resource Pools - technológia chápaná ako logická jednotka, ktorá sa stará o systémové zdroje.

Predvolený pool - jediný pool v systéme, ktorý vznikne po aktivovaní technológie resource pool.

Secure Shell - šifrovaná podoba telnetu. V preklade označuje bezpečný príkazový riadok.

Pkgadd - nástroj na aktualizáciu systému a inštaláciu aktualizácií, ktorý je súčasťou operačného systému Solaris 10.

Pkgutil - nástroj na aktualizáciu systému a inštaláciu aktualizácií, ktorý je potrebné nainštalovať pomocou nástroja pkgadd.

HTTP server - počítačový program, ktorý zodpovedá za vykonávanie HTTP príkazov od jednotlivých klientov.

SQL server - počítačový program, ktorý zodpovedá za vykonávanie SQL príkazov od jednotlivých klientov.

PHP - je skriptovací jazyk, využívaný najmä pri vývoji webových aplikácií, pričom ho môžeme použiť aj pri HTML.

SFTP - protokol pre prenos súborov, ktorý vznikol zdokonalením FTP.

Skript - počítačový program vytvorený pre uľahčenie operácií v počítači.

Úvod

Pod pojmom virtualizácia sa najčastejšie myslí virtualizácia behu operačného systému, podstatou ktorej je vsunutie novej abstrakčnej vrstvy medzi hardvér a operačný systém počítača. Ide o vytváranie virtuálnych počítačov nad fyzickým počítačom. Hlavným dôvodom prečo virtualizovať, je dôvod šetrenia finančných prostriedkov. Dôvodom prečo virtualizovať môže byť aj snaha zvýšiť izoláciu a bezpečnosť jednotlivých aplikácií, služieb a užívateľov. Hlavnou úlohou virtualizácie je zabezpečiť úplnú nezávislosť virtuálnych počítačov od fyzického počítača. V prípade dosiahnutého vysokého stupňa nezávislosti virtuálnych počítačov od fyzického počítača môžeme pristúpiť k migrácii virtuálnych počítačov z jedného fyzického počítača na iný fyzický počítač.

Základným virtualizačným nástrojom operačného systému Solaris 10 sú Solaris zóny. Úlohou zón je vytvoriť systém, v ktorom sa bežiace aplikácie virtuálnych počítačov správajú ako aplikácie rôznych fyzických počítačov. Rozlišujeme dva druhy zón. Prvým druhom je pôvodná inštancia operačného systému, nazývaná ako globálna zóna. Každá vytvorená zóna sa nazýva lokálna zóna. Technológia zón je založená na využívaní technológie Solaris kontajnerov, podstatou ktorej je rozkúskovanie pôvodnej inštancie operačného systému na niekoľko virtuálnych inštancií.

Pri vytváraní zón a kontajnerov môžeme využiť rôzne technológie ako napríklad Fair Share Scheduler, ktorá slúži na pridelenie podielov zo zdrojov systému. Ďalšou takou technológiou môže byť technológia Dynamic Resource Pools, ktorá slúži na dosiahnutie rôznych úrovni hardvérovej izolácie. Zóny môžeme vytvárať dvoma spôsobmi. Prvý spôsob sa zakladá na priradení vytvorenej zóny k novovytvorenému poolu. Druhý spôsob sa zakladá na priradení vytvorenej zóny k predvolenému poolu. Predtým, ako začneme vytvárať zóny, musíme zabezpečiť aktivovanie technológie resource pools. Rozdiel medzi jednotlivými spôsobmi vytvárania zón je v tom, že v prípade prvého spôsobu je nevyhnutné vytvoriť nový pól a v prípade druhého spôsobu je vhodné prerozdeliť dostupné zdroje systému technológiu Fair Share Scheduler.

Pred začatím vytvárania zóny, je nevyhnutné vytváranej zóne prideliť miesto na disku a prideliť mu prístupové práva. Následne je možné pristúpiť ku konfigurácii vytváranej zóny, ktorá pozostáva z pridelenia súborového systému, nastavenia možnosti bootovania, sieťovej konfigurácie a pridelenia poolu. Konfiguráciu ukončíme overením a zapísaním. Rozdiel v konfigurácii medzi jednotlivými spôsobmi vytvárania zón je v tom, že pri prvom spôsobe vytvárania zón priradíme vytváranú zónu k novovytvorenému poolu a pri druhom spôsobe vytváranú zónu priradíme k predvolenému poolu, pričom musíme v konfigurácii zóny priradiť vytváranej zóne vhodný počet podielov z dostupných zdrojov. Po nakonfigurovaní je potrebné vytváranú zónu nainštalovať a nabootovať. Po prvom prihlásení sa do vytvorenej zóny sme vyzvaný na nastavenie parametrov zóny.

Najlepším nástrojom na aktualizáciu a inštaláciu jednotlivých aplikácií je pkgutil, ktorý je potrebné nainštalovať pomocou nástroja pkgadd, ktorý je súčasťou operačného systému Solaris 10. Najvhodnejšími nástrojmi na záťažové testovanie MySQL serverov jednotlivých zón je MySQL5test a The MySQL Benchmark, ktoré sú súčasťou balíčkovacieho systému aktualizácie, preto ich inštalácia a nastavenie sú veľmi jednoduché. Ich hlavnou prednosťou je ich presnosť. Najvhodnejšími nástrojmi na záťažové testovanie nainštalovaných Apache serverov jednotlivých zón je Apache JMeter, vyznačujúci sa najmä svojou presnosťou, a Ab - Apache HTTP server benchmarking tool, ktorý je súčasťou balíčkovacieho systému aktualizácie.

Cieľom diplomovej práce je vysvetlenie pojmu virtualizácia, odôvodnenie potreby virtualizácie, oboznámenie s hlavnými virtualizačnými stratégiami a implementácia virtualizácie v operačnom systéme Solaris 10 pomocou Solaris kontajnerov a zón. Hlavným cieľom diplomovej práce je inštalovanie viacerých web a sql serverov na jednom počítači a ich testovaniu záťažovými testami. Teoretická časť je venovaná vysvetleniu základných pojmov, oboznámeniu sa s využívanými virtualizačnými technológiami a priradovaniu vytváraných zón k novovytvorenému poolu. Praktická časť pozostáva z dvoch častí. Prvá časť popisuje postup priradovania vytváraných zón k predvolenému poolu, klonovania zón, administrácie jednotlivých zón, vymazávania zón systému a migrácie zón. Druhá časť je venovaná záťažovým testom Apache a MySQL serverov jednotlivých zón.

1 TEORETICKÁ ČASŤ

1.1 Virtualizácia

Pod pojmom virtualizácia sa najčastejšie myslí virtualizácia behu operačného systému, ktorá predstavuje vsunutie novej abstrakčnej vrstvy, pričom nová abstrakčná vrstva sa vsunie medzi hardvér a operačný systém počítača, čo nám umožní to, že v jednom okamihu môžeme spúšťať naraz viac rôznych operačných systémov. Ide vlastne o vytváranie virtuálnych počítačov nad fyzickým počítačom s cieľom znížiť počet fyzických počítačov, čo sa nám často prejaví v efektívnejšom využívaní dostupných zdrojov a zjednodušenom spravovaní celého systému, s čím je často spojené šetrenie veľkého množstva financií.

Virtuálny počítač alebo virtuálny server je podľa definície izolovaný výpočtový systém, ktorý poskytuje všetky potrebné prostriedky na samostatný beh aplikácií, takže virtuálny stroj predstavuje takmer presnú repliku fyzického stroja.[1]

1.1.1 Dôvody prečo pristúpiť k virtualizácii

Človek, ktorý nie je zbehlý v tejto problematike, si môže položiť nasledujúcu otázku: Načo použiť pri navrhovaní počítačových sietí virtualizáciu? Ved' každý operačný systém počítača je prostriedok, ktorý sa má starať o správu všetkých zdrojov hardvéru v počítači.[2]

Hlavným dôvodom prečo virtualizovať, je dôvod šetrenia finančných prostriedkov, pretože zlé budovanie počítačových sietí sa môže negatívne prejsť vo zvýšených nákladoch na navrhovanie, budovanie, spravovanie, prevádzku a údržbu počítačových sietí. Príkladom môže byť počítač, na ktorom sú nainštalované aplikácie, respektíve služby, ktoré nevyťažujú úplne zdroje fyzického počítača. Vyťaženosť zdrojov počítača môžeme jednoducho zvýšiť pomocou virtualizácie, pretože

virtualizácia nám umožňuje vytvoriť systém založený na zdieľaní fyzických zdrojov, čím sa zdroje budú využívať efektívnejšie, zabezpečí sa takto ekonomickejšie priemerné zaťaženie fyzických zdrojov, čo sa nám následne prejaví v znížených nákladoch na navrhovanie, budovanie, spravovanie, prevádzku a údržbu počítačových sietí. Pri takomto zdieľaní fyzických zdrojov netreba zabúdať na zabezpečenie pred prílišným vyťažovaním zdrojov jedným z virtuálnych počítačov na úkor ostatných virtuálnych počítačov. Príkladom možnosti, pomocou ktorej predchádzame nadmernému vyťažovaniu zdrojov jedným virtuálnym počítačom, môže byť pridelenie presných podielov z celkových dostupných zdrojov fyzického počítača. Ďalšou z možností obmedzenia nadmerného vyťaženia zdrojov jedným virtuálnym počítačom môže byť obmedzenie maximálneho množstva využívaných zdrojov fyzického počítača pre jednotlivé virtuálne počítače.

Ďalším dôvodom prečo virtualizovať, je snaha posilniť izoláciu a bezpečnosť, pretože úlohou virtualizácie je zabezpečiť izoláciu jednotlivých aplikácií, služieb a užívateľov, čo sa prejavuje aj vo zvýšenej bezpečnosti. Príkladom môže byť počítač, na ktorom máme viac aplikácií, respektíve služieb. Potom na tomto počítači môže nastať reálna situácia založená na tom, že pomocou slabého miesta v určitej aplikácii, respektíve chybe v bezpečnostných nastaveniach aplikácie, môže neoprávnená osoba preťažiť celý systém, poprípade sa môže podpísať aj pod pád celého systému. Takýmto situáciám môžeme predísť virtualizáciou a to tak, že každá aplikácia bude nainštalovaná na samostatný virtuálny počítač, čiže v prípade, ak neoprávnená osoba využije slabé miesto aplikácie, môže dosiahnuť maximálne preťaženie, respektíve pád iba virtuálneho počítača a nie celého systému, pretože každý virtuálny počítač má svojich vlastných superužívateľov, ktorí nemajú právo zasahovať do činnosti iných virtuálnych počítačov ani do činnosti fyzického počítača. Takto pomocou virtualizácie potom vytvárame bezpečné prostredie pre jednotlivé bežiacie aplikácie.

Absolútnu nezávislosť vytvorených virtuálnych počítačov môžeme využiť pre množstvo účelov, napríklad ju môžeme využiť pri vývoji a ladení nestabilných jadier operačných systémov, pri testovaní kompatibility jednotlivých verzií programov s rôznymi operačnými systémami. V prípade dosiahnutého vysokého stupňa nezávislosti vytvorených virtuálnych počítačov od fyzického počítača môžeme pristúpiť

k migrácii virtuálneho počítača, čiže k presunu virtuálneho počítača z jedného fyzického počítača na iný fyzický počítač.

Migrácia nám umožňuje jednoducho uskutočňovať zmeny hardvéru na fyzickom počítači bez veľkých výpadkov poskytovaných služieb. Okrem toho nám migrácia slúži aj ako nástroj na zvýšenie spoľahlivosti jednotlivých služieb. Umožní nám v prípade akéhokoľvek hardvérového výpadku presunúť funkčný stav virtuálnych počítačov na vopred pripravený záložný počítač.[2]

1.1.2 Metódy virtualizácie

Jednotlivé metódy virtualizácie sa líšia predovšetkým v rôznych možnostiach ovplyvňovať virtuálne rozhranie. Základný rozdiel medzi jednotlivými metódami virtualizácie je spôsob realizácie virtualizačnej vrstvy.

Virtualizácia nám ponúka tieto dve základné riešenia [2]:

- Virtuálny server (VS): Pod jedným operačným systémom je spustených viac od seba izolovaných serverov.[2]
- Virtuálny počítač (virtual machine - VM): Na jednom počítači je spustených viac operačných systémov naraz.[3]

1.1.3 Simulácia

Patrí medzi najviac využívané metódy virtualizácie. Základ simulácie, ako virtualizačnej metódy, je založený na simulovaní správania jedného počítačového systému pomocou druhého počítačového systému, pričom simulácia sa snaží vytvoriť pre bežiacie programy také prostredie, ktoré úplne simuluje prostredie pôvodného reálneho systému. Základným nástrojom simulácie ako virtualizačnej metódy je aplikačný program simulátor, ktorého hlavnou úlohou je interpretovanie zdrojového kódu.

1.1.4 Emulácia

Je to jedna z najpomalších metód virtualizácie, ktorá je veľmi podobná simulácii, preto sa často zvyknú pojmy emulácia a simulácia považovať za to isté. Výhodou emulácie oproti simulácii je skutočnosť, že okrem softvéru dokáže emulovať aj hardvér, pričom emulovaný hardvér nemusí byť súčasťou fyzického počítača. Hlavným rozdielom medzi emuláciou a simuláciou je skutočnosť, že emulácia sa v podstate nesnaží vytvárať identickú kópiu fyzického počítača. Pri emulácii dochádza iba k vytvoreniu jednoduchej podoby fyzického počítača, nie k identickej kópii fyzického počítača, pretože úlohou emulácie je simulovanie iba najdôležitejších znakov fyzického počítača. Emulácia sa najčastejšie využíva pri testovaní aplikácií a služieb na rôznych procesoch, ktoré za bežných podmienok nie sú veľmi jednoducho dostupné.

1.1.5 Úplná simulácia

Je to metóda virtualizácie, ktorá je veľmi blízka simulácii a v určitých črtách aj emulácii. Jej základná myšlienka je založená na tom istom základe ako emulácia, čiže podstatou úplnej simulácie nie je simulovanie úplne rovnakého prostredia, ktoré prislúcha pôvodnému fyzickému počítaču. Na rozdiel od bežnej simulácie sa zakladá na predpoklade, že čím viac časti je možné vykonať bez simulácie, čiže vytvárania identickej kópie fyzického počítača, tým virtuálny počítač beží efektívnejšie, z toho vyplýva skutočnosť, že úplnú virtualizáciu nie je za bežných podmienok možné vykonať na každom fyzickom počítači.

1.2 Virtualizácia v systéme Solaris 10

Základným virtualizačným nástrojom v operačnom systéme Solaris 10 sú Solaris zóny. Jednotlivé Solaris zóny sa vyznačujú vlastným súborovým systémom, vlastným administrátorským účtom a vlastným sieťovým rozhraním. Výhodou je, že jednotlivé dostupné voľné zdroje systému sa využívajú maximálne ako je to možné.[2]

Technológia Solaris zón je založená na využívaní technológie nazvanej ako Solaris kontajnery. Podstatou Solaris kontajnerov je rozdelenie pôvodnej inštancie operačného systému Solaris 10 na niekoľko virtuálnych inštancií operačného systému.[4]

Základnou úlohou Solaris kontajnerov je lepšie riadenie zdrojov a izolovanie vytvorených virtuálnych počítačov od fyzického počítača. Solaris kontajnery umožňujú administrátorovi rozdeliť zdroje medzi jednotlivé aplikácie a služby. Umožňujú zabezpečiť prostredie, v ktorom nebude hroziť vzájomné rušenie sa jednotlivých aplikácií a služieb. Vytvorí sa tak systém, kde chybná aplikácia nemôže ovplyvňovať žiadnu inú aplikáciu, ktorá sa nachádza v inej zóne.[2]

1.3 Solaris zóny

Skôr, ako vytvoríme v systéme akúkoľvek Solaris zónu, máme systém pozostávajúci iba z pôvodnej inštancie operačného systému Solaris 10. Pôvodná inštancia operačného systému je hlavnou časťou systému, ktorú nazývame ako globálna zóna.[4]

Globálna zóna sa vyznačuje tým, že má prístup k fyzickému počítaču, čiže k reálnemu fyzickému hardvéru počítača. Úlohou každej globálnej zóny je kontrola všetkých procesov bežiacich v systéme. Administrátor globálnej zóny môže jednoducho spravovať systém ako celok. Globálna zóna má oprávnenie na vytváranie a kontrolu ďalších zón, ktoré už nie sú globálne, preto ich nazývame ako lokálne zóny. V praxi sa zvyknú nazývať lokálne zóny iba ako zóny a všetky ostatné zóny sa zvyknú nazývať ako miestne zóny.[2]

Maximálny počet lokálnych zón, ktoré môžeme vytvoriť v rámci vnútra jednej globálnej zóny je obmedzený najmä hardvérom fyzického počítača, pretože každá z lokálnych zón systému potrebuje pre svoje správne fungovanie minimálne 40 MB operačnej pamäte a minimálne 320 MB voľného miesta na pevnom disku. V skutočnosti

je však možné vytvoriť vo vnútri jednej globálnej zóny maximálne 8192 lokálnych zón, väčšie množstvo lokálnych zón tento operačný systém nepovoľuje.

Výhodou virtualizácie pomocou Solaris zón je to, že napriek tomu, že všetky lokálne zóny existujú vo vnútri jednej globálnej zóny, majú svojich vlastných užívateľov, vrátane superužívateľa root. Jednotliví užívatelia lokálnych zón, bez rozdielu či ide o bežného užívateľa alebo užívateľa root, môžu zasahovať iba do činnosti zóny, v ktorej boli vytvorení, čím je zabezpečená izolácia a nezávislosť jednotlivých aplikácií jednotlivých zón. To isté platí aj o užívateľoch globálnej zóny, okrem užívateľa root globálnej zóny, ktorý má ako jediný užívateľ právo vytvárať, klonovať a vymazávať jednotlivé lokálne zóny globálnej zóny.

Výhodou pre celý systém je, že jednotlivé lokálne zóny sú izolované od fyzického hardvéru. Na izolovanie lokálnych zón od fyzického hardvéru sa používa vrstva virtuálnej platformy, ktorej úlohou je poskytovať každej zóne vlastné virtuálne sieťové rozhranie, vlastné súborové systémy a v neposlednom rade vlastnú virtuálnu konzolu.
[2]

Virtualizácia Solaris zón sa uskutočňuje na úrovni operačného systému. Technológia Solaris zón poskytuje virtuálne prostredie operačného systému v rámci jednej inštancie operačného systému Solaris 10. Každé virtuálne prostredie, nazývané Solaris zóna, má svoju vlastnú identitu, ktorá je oddelená od hardvéru. Každá Solaris zóna sa aplikáciám a užívateľom javí ako celkom samostatný systém.[5]

Charakteristickým znakom všetkých lokálnych zón systému je, že obsahujú kópiu softvérového vybavenia globálnej zóny. Výhodou je, že jednotlivé lokálne zóny majú, napriek tomu, že obsahujú kópiu programového vybavenia globálnej zóny, vlastnú softvérovú konfiguráciu, čiže jednotlivé aplikácie, respektíve služby, nepreberajú nastavenia z globálnej zóny.

Vo vnútri globálnej zóny môžeme mať rôzne druhy lokálnych zón. Prvým druhom sú predvolené lokálne zóny, ktoré sa v praxi zvyknú nazývať aj ako natívne. Ich

charakteristickou črtou je, že majú rovnaké vlastnosti ako pôvodná inštancia operačného systému, čiže ako globálna zóna, to znamená, že obsahujú aj rovnakú verziu operačného systému Solaris ako globálna zóna.

Druhým druhom lokálnych zón sú značkové zóny. V praxi sa rozlišujú dva typy značkových zón. Prvým typom sú značkové zóny typu 1x. Ich charakteristickou črtou je, že slúžia na virtualizáciu linuxového prostredia, čiže sú to lokálne zóny, ktoré majú v sebe namiesto Solarisu Linux. Druhým typom značkových zón sú zóny slúžiace na virtualizáciu inej verzie operačného systému Solaris, aká je v globálnej zóne, čiže v našom prípade na virtualizáciu operačného systému Solaris 9 a Solaris 8.

1.4 Rozdiel medzi zónami a kontajnermi

Vo všeobecnosti platí pravidlo, že Solaris kontajnery nie sú totožné s Solaris zónami. Technológia zón môže byť jednoducho použitá na vytvorenie Solaris kontajnerov. Solaris zóna je istý druh kontajnera, ale na druhej strane treba povedať, že Solaris kontajner nemusí byť druhom Solaris zóny. Solaris zóna je virtuálny operačný systém, ktorý poskytuje chránené prostredie, v ktorom aplikácia beží. Solaris zóny umožňujú veľmi dobré oddelenie jednotlivých aplikácií, a to až na úrovni jadra operačného systému. Toto oddelenie je vhodné najmä z hľadiska bezpečnosti, pretože v prípade, ak sa útočníkovi podarí pomocou zraniteľnosti v niektorej službe alebo aplikácii získať nadštandardný prístup k systému, dokáže ohroziť len súčasti zóny, v ktorej sa nachádza. Aplikácie sú chránené od seba vzájomne, čo nám zabezpečuje izolácia jednotlivých zón.[2]

Solaris kontajner je tiež zóna, ktorá tiež používa operačný systém pre riadenie zdrojov počítača. Veľa ľudí si zamieňa dve slová kontajner a zóna.[6]

Solaris kontajnery zabezpečujú prostredie, ktoré podporuje bezpečné usporiadanie aplikácií do jediného fyzického servera.[7]

1.5 Fair Share Scheduler (FSS)

Solaris technológia využívaná pri vytváraní jednotlivých Solaris zón, respektíve Solaris kontajnerov.

Operačný systém Solaris 10 spracúva každú požiadavku na dostupné zdroje s rovnakou prioritou, bez ohľadu na dôležitosť jednotlivých aplikácií, respektíve procesov. Tento spôsob pridelovania dostupných voľných zdrojov nemusí byť výhodný v prípade, ak požiadavky na dostupné zdroje prekračujú kapacitu dostupných voľných zdrojov. V takomto prípade Solaris 10 sa prispôbuje vzniknutej situácii a obmedzuje prístup k zvyšným voľným zdrojom. Technológia FSS sa môže jednoducho použiť na pridelovanie nevyhnutných požiadaviek na zdroje, ktoré potrebujú jednotlivé aplikácie, respektíve procesy systému.[4]

Táto technológia umožňuje prideliť každej aplikácii, za predpokladu, že sa tie nachádzajú v samostatných zónach, určitý podiel z dostupných voľných zdrojov. Tak ako vidno na Obr. 1, sme pridelili prvej aplikácii 1 diel, druhej aplikácii 3 diely, tretej aplikácii 2 diely, štvrtej aplikácii 1 diel [2]:



Obr. 1 Ukážka prerozdelenia zdrojov pomocou FSS

Charakteristickým znakom technológie FSS je, že pridelené podiely nepredstavujú percentuálne podiely z dostupných zdrojov CPU, ale relatívne podiely z dostupných zdrojov CPU. To znamená, ak v systéme nastane situácia, že niektorá zóna nevyužíva pridelené zdroje naplno, dôjde k automatickému prerozdeleniu nevyužívaných zdrojov medzi ostatné zóny, ktoré využívajú svoje pridelené zdroje naplno. V prípade, ak zdroje

CPU boli prerozdelené pomocou technológie FSS, a ak v systéme nastane situácia, že je aktívna iba jedna zóna, čiže globálna zóna, nedochádza k žiadnemu deleniu dostupných zdrojov CPU. To znamená, že globálna zóna využíva 100 % z dostupných zdrojov CPU.

FSS sa zakladá na priradovaní jednotlivých podielov pre jednotlivé zóny. Priradené množstvo podielov je priamo úmerné s priradením dostupných voľných zdrojov CPU z predvoleného poolu. To znamená, že priradením väčšieho množstva podielov priradíme väčší podiel z dostupných voľných zdrojov a opačne priradením menšieho množstva podielov sa zabezpečí menší podiel z dostupných voľných zdrojov.[4]

1.6 Dynamic Resource Pools

Táto Solaris technológia je využívaná pri vytváraní jednotlivých Solaris zón a Solaris kontajnerov. Jej základnou úlohou je dosiahnutie rôznych úrovni izolácie v prípade, ak to vyžadujú podmienky kladené na aplikácie a procesy bežiacie v jednotlivých zónach, respektíve kontajneroch.

Pred vytvorením akéhokoľvek poolu a Solaris zóny budeme mať v systéme iba predvolený pool a globálnu zónu, tak ako to je znázornené na Obr. 2 [2]:



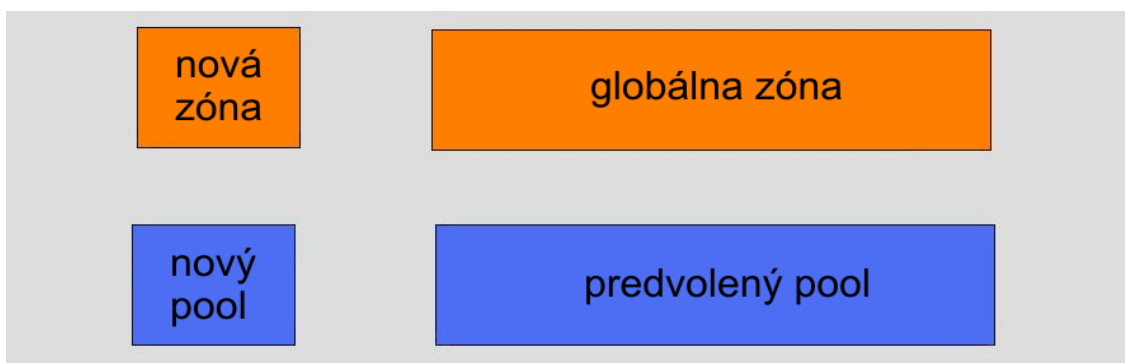
Obr. 2 Ukážka globálnej zóny a predvoleného poolu

V operačnom systéme Solaris 10 je resource pool logická jednotka, ktorá sa stará o systémové zdroje, ako napríklad procesor a pamäť. Pred vytvorením poolov existuje v systéme jeden hlavný pool, ktorý sa nazýva ako predvolený pool. Každý ďalší vytvorený pool je prevzatý z predvoleného poolu.[2]

1.7 Spôsoby vytvárania Solaris zón

Pri vytváraní jednotlivých Solaris zón a Solaris kontajnerov môžeme postupovať dvoma základnými spôsobmi:

- priradenie vytvorenej Solaris zóny k novovytvorenému poolu,
- priradenie vytvorenej Solaris zóny k predvolenému poolu.



Obr. 3 Ukážka pridelenia vytvorenej zóny k novému poolu



Obr. 4 Ukážka pridelenia zón k predvolenému poolu

1.8 Vytvorenie lokálnej zóny a jej priradenie k novému poolu

K tomuto spôsobu vytvárania Solaris zón môžeme pristúpiť iba v prípade, ak máme v hardvérovej konfigurácii fyzického počítača viac ako jeden procesor, pretože pri tomto spôsobe vytvárania zón je nevyhnutné najprv vytvoriť nový pool a vytvorenie nového poolu bez viac procesorov v hardvérovej konfigurácii fyzického počítača nie je možné. Po vytvorení nového poolu môžeme pristúpiť k vytváraniu novej zóny a následne k jej priradeniu k novému poolu.

1.8.1 Vytvorenie poolu

Predtým, ako začneme vytvárať nový pool, musíme zabezpečiť aktivovanie technológie resource pools:

pooladm -e.

Po aktivovaní technológie resource pools je nevyhnutné zapísať predvolenú konfiguráciu do konfiguračného súboru /etc/pooladm.conf:

pooladm -s,

pričom bez zapísania konfigurácie by sme nemohli pokračovať ďalej vo vytváraní nového poolu.

K aktuálnemu nastaveniu globálnej zóny, kde uvidíme aj zoznam jednotlivých poolov, sa môžeme dostať zadaním:

pooladm.

Pred samostatným vytvorením nového poolu je nevyhnutné vytvoriť nový pset, v našom prípade pset s názvom test-pset, ktorý bude disponovať s výkonom minimálne jedného procesora a maximálne troch procesorov:

poolcfg -c 'create pset test-pset (uint pset.min=1; uint pset.max=3)',

tento príkaz hovorí zmeniť konfiguráciu poolu, pričom vytvorí procesor set pset s názvom test-pset, pričom mu pridelí minimálne 1 CPU a maximálne 3 CPU.

Samostatné vytvorenie nového poolu test-pool zabezpečíme:

poolcfg -c 'create pool test-pool'.

Priradenie poolu test-pool k psetu test-pset:

poolcfg -c 'associate pool test-pool (pset test-pset)'.

Nevyhnutné je uskutočniť aktualizáciu týchto zmien:

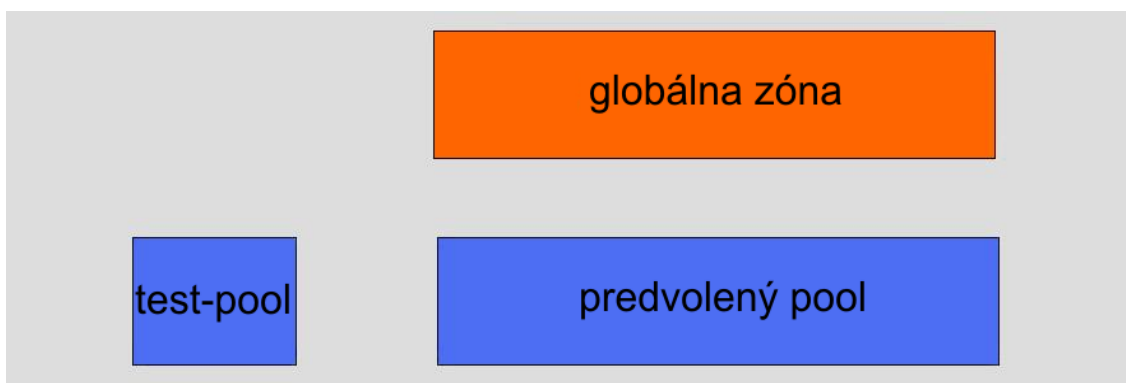
pooladm -c.

Overenie správnosti vytvorenia poolu test-pool uskutočníme:

pooladm,

pričom pomocou tohto príkazu si overíme okrem existencie poolu test-pool aj existenciu psetu test-pset.

Vytvorením poolu test-pool budeme mať v systéme globálnu zónu a dva pooly:



Obr. 5 Ukážka dvoch poolov a globálnej zóny

K takejto hardvérovej izolácii vytváraných zón sme pri plnení diplomovej práce nemohli pristúpiť, z dôvodu hardvérovej konfigurácie počítača, na ktorom sa vykonávala diplomová práca, pričom dôvodom bola skutočnosť, že na počítači, na ktorom sa vykonávala diplomová práca, bol iba jeden procesor.

1.8.2 Vytvorenie Solaris zóny priradením k novému poolu

Po úspešnom vytvorení nového poolu, v našom prípade test-pool, môžeme pristúpiť k vytváraniu novej zóny, v našom prípade test-zone, a následne k jej priradeniu k vytvorenému poolu, pričom vytvorenie každej novej zóny pozostáva z týchto krokov:

- konfigurácia (definovanie zóny, definovanie súborového priestoru zóny, konfigurovanie sieťových adaptérov),
- inštalácia (samostatné vytvorenie zóny),
- virtuálna platforma správy (zavedenie, pozastavenie, reštart zóny),
- prihlásenie sa do zóny (prihlásenie sa do zóny, odhlásenie sa zo zóny).

Pred vytvorením každej zóny, to isté platí aj o našej vytváranej zóne test-zone, je nevyhnutné vytváranej zóne vyčleniť miesto na disku:

```
# mkdir -p /export/home/zones/test-zone/local.
```

Nevyhnutným krokom je priradenie prístupových práv k vyčlenenému miestu:

```
# chmod 700 /export/home/zones/test-zone,
```

pričom bez tohto kroku nemôžeme pokračovať v konfigurácii a inštalácii zóny.

Prvým krokom konfigurácie zóny je priradenie názvu vytváranej zóne:

```
# zonecfg -z test-zone.
```

Pretože chceme vytvárať novú zónu:

```
zonecfg:test-zone> create.
```

Pre správne fungovanie zóny je nevyhnutné zónu priradiť do súborového systému, pomocou príkazu set zonepath:

```
zonecfg:test-zone> set zonepath=/export/home/zones/test-zone.
```

Automatické nabootovanie zóny po zapnutí systému, respektíve reštarte systému zabezpečíme pomocou:

```
zonecfg:test-zone> set autoboot=true.
```

Nevyhnutným krokom, ak chceme používať zónu, je nakonfigurovanie siete. V našom prípade ide iba o priradenie internej IP adresy, ktoré znemožní pripojenie sa do zóny cez internet:

```
zonecfg:test-zone> add net  
zonecfg:test-zone:net>set address=10.0.0.1  
zonecfg:test-zone:net>set physical=ngel0  
zonecfg:test-zone:net> end.
```

Priradenie poolu test-pool k vytváranej zóne:

```
zonecfg:test-zone> set pool=test-pool.
```

Pred zapísaním konfigurácie je vhodné overiť zapisovanú konfiguráciu:

```
zonecfg:test-zone> verify.
```

Zapísanie konfigurácie uskutočníme:

```
zonecfg:test-zone> commit.
```

Konfiguráciu ukončíme:

```
zonecfg:test-zone> exit.
```

Po ukončení konfigurácie môžeme pristúpiť k jej inštalácii:

```
# zoneadm -z test-zone install,
```

pričom predpokladom úplného nainštalovania zóny je správna konfigurácia zóny.

Správne nainštalovanú zónu môžeme nabootovať:

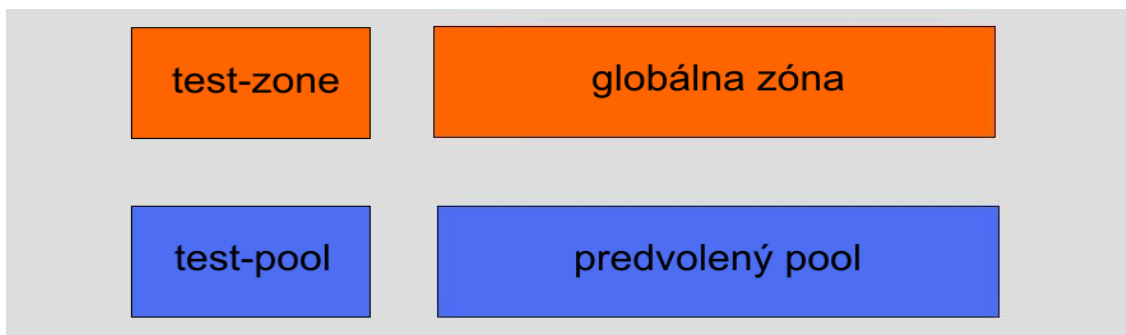
```
# zoneadm -z test-zone boot.
```

Prvé prihlásenie sa do zóny uskutočníme:

zlogin -C test-zone.

Po prvom prihlásení odpovedáme na položené otázky, čiže vytváranej zóne test-zone priradíme príslušajúce parametre. Z prihlásenej zóny test-zone sa odhlásime pomocou ~.

Vytvorením novej zóny test-zone a nového poolu test-pool budeme mať v systéme dve zóny a dva pooly:



Obr. 6 Ukážka dvoch poolov a dvoch zón

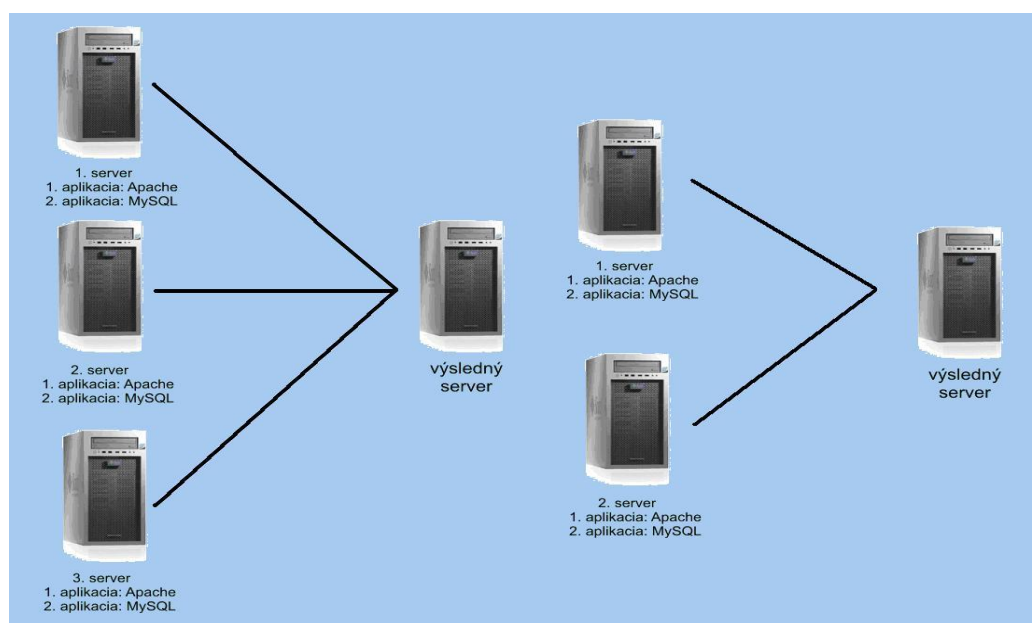
K takémuto spôsobu vytvárania zón sme v diplomovej práci nemohli pristúpiť z dôvodu, že hardvérová izolácia zón nebola možná z dôvodu, že na počítači, na ktorom sa uskutočňovala diplomová práca, bol iba jeden procesor.

2 PRAKTICKÁ ČASŤ

Naším cieľom je vytvoriť na jednom fyzickom počítači systém pozostávajúci z troch Solaris zón, pričom jedna zóna bude globálna zóna a dve zóny budú lokálne, pomocou Solaris technológií ako sú Solaris zóny a Solaris kontajnery. Programovým vybavením všetkých troch zón je HTTP a SQL server. Následne jednu z lokálnych zón premiestniť do inej hostujúcej globálnej zóny. Presunutú zónu upraviť tak, aby mohla správne fungovať v pôvodnej globálnej zóne a súčasne aj v novej hostujúcej globálnej zóne. Výkonnosť jednotlivých HTTP a SQL serverov pôvodného fyzického počítača otestovať záťažovými testami.



Obr. 7 Ukážka dvoch globálnych a troch lokálnych zón



Obr. 8 Ukážka simulácie

2.1 Vytváranie Solaris zón pridelením k predvolenému poolu

Pri tomto spôsobe vytvárania Solaris zón priradujeme vytvorené zóny k predvolenému poolu, to znamená, že pri tomto spôsobe vytvárania Solaris zón nie je potrebné vytvárať nové pooly. K takémuto spôsobu vytvárania zón pristupujeme:

- v prípade, ak nepovažujeme za dôležité využiť hardvérovú izoláciu, čiže vytvárať nové pooly
- v prípade, ak nám vytvorenie poolu nepovoľuje hardvérová konfigurácia počítača, čiže počítač má iba jeden procesor.

Pri tomto spôsobe vytvárania Solaris zón, čiže pri priradovaní vytvorených zón k predvolenému poolu, je dôležité si uvedomiť, že všetky lokálne zóny a globálna zóna sa budú spoločne deliť o všetky dostupné voľné zdroje CPU. Z tohto dôvodu je vhodné zabezpečiť, aby žiadna zo zón, či už globálna alebo lokálna zóna, neúmerne neuberala zdroje ostatným zónam, čo by sa mohlo prejaviť v spomalení celého systému. Vhodným nástrojom na prerozdelenie dostupných voľných zdrojov systému je nástroj FSS.

2.1.1 Aktivovanie resource pools

Napriek tomu, že tento spôsob vytvárania Solaris zón nie je založený na vytváraní nového poolu, je nevyhnutné zabezpečiť aktivovanie resource pools. Bez aktivovania resource pools by sme nemohli pristúpiť k prerozdeleniu zdrojov CPU pomocou technológie FSS. Pri aktivovaní postupujeme tak isto ako pri aktivovaní resource pools pri vytváraní Solaris zón priradením k novému poolu:

pooladm -e.

Po aktivovaní resource pools je nevyhnutné zapísať konfiguráciu do konfiguračného súboru:

pooladm -s,

pričom bez zapísania konfigurácie by sme nemohli prerozdeliť zdroje CPU pomocou technológie FSS.

K aktuálnemu nastaveniu globálnej zóny, kde uvidíme aj informácie o predvolenom poole, sa dostaneme pomocou:

pooladm,

pričom po zadaní získame odpoveď:

system default

string system.comment

int system.version 1

boolean system.bind-default true

string system.pooldefault.objectives wt-load

pool pool_default

int pool.sys_id 0

boolean pool.active true

boolean pool.default true

int pool.importance 1

string pool.comment

pset pset_default

pset pset_default

int pset.sys_id -1

boolean pset.default true

uint pset.min 1

uint pset.max 65536

string pset.units population

uint pset.load 831

uint pset.size 1

string pset.comment

cpu

int cpu.sys_id 0

string cpu.comment

string cpu.status on-line.

2.1.2 Modifikovanie predvoleného poolu s využitím technológie FSS

Po bezchybnom aktivovaní resource pools môžeme pristúpiť k modifikácii predvoleného poolu pomocou technológie FSS, pričom prvým krokom modifikácie predvoleného poolu s využitím FSS je:

poolcfg -c 'modify pool pool_default (string pool.scheduler="FSS")',

pričom túto zmenenú konfiguráciu predvoleného poolu je nevyhnutné aktualizovať v konfiguračnom súbore:

pooladm -c.

Po aktualizovaní novej konfigurácie je nevyhnutné zabezpečiť presun všetkých procesov predvoleného poolu a súvisiacich oblastí v rámci FSS:

priocntl -s -c FSS -i class TS

priocntl -s -c FSS -i pid 1.

Informácie o počte podielov, ktoré sú pridelené globálnej zóne získame:

prctl -n zone.cpu-shares -i zone global

zone: 0: global

<i>NAME</i>	<i>PRIVILEGE</i>	<i>VALUE</i>	<i>FLAG</i>	<i>ACTION</i>	<i>RECIPIENT</i>
<i>zone.cpu-shares</i>					
<i>privileged</i>	<i>1</i>	<i>-</i>	<i>none</i>		<i>-</i>
<i>system</i>	<i>65,5K</i>	<i>max</i>	<i>none</i>		<i>-.</i>

Pridelenie 2 podielov globálnej zóne:

prctl -n zone.cpu-shares -v 2 -r -i zone global.

Pridelenie X podielov zóne Y:

prctl -n zone.cpu-shares -v X -r -i zone Y.

Informácie o počte podielov, ktoré sú pridelené zóne Y získame:

prctl -n zone.cpu-shares -i zone Y.

2.1.3 Vytvorenie lokálnej zóny Kirp141

V prípade, ak nepovažujeme za dôležité vytvoriť zóne Kirp141 nový pool vyčlenením z predvoleného poolu, môžeme vytvoriť novú zónu a následne ju priradiť priamo k predvolenému poolu. K takejto možnosti môžeme byť prinútený aj v prípade, ak na pôvodnom fyzickom počítači máme iba jeden procesor, čiže vytvorenie nového poolu nie je možné, čo je aj prípad lokálnych Solaris zón Kirp141 a Kirp140.[4]

V prípade, ak priraďujeme novú zónu k predvolenému poolu, je nevyhnutné uvedomiť si, že všetky lokálne zóny a globálna zóna sa budú spoločne deliť o dostupné voľné prostriedky CPU. Z tohto dôvodu je dôležité zabezpečiť vhodné prerozdelenie zdrojov CPU a to tak, aby žiadna zo zón nevyťažovala nadmerne CPU na úkor ostatných zón. Vhodným nástrojom pre prerozdelenie dostupných voľných zdrojov systému je nástroj FSS.

Pred vytvorením novej zóny Kirp141 je potrebné vyčleniť zóne miesto na disku:

mkdir -p /export/home/zones/kirp141/local.

K vyčlenenému miestu na disku by mal mať prístup iba superužívateľ root globálnej zóny. Je to veľmi dôležité najmä z hľadiska bezpečnosti novo vytváranej zóny Kirp141. Zamedzíme takto neoprávneným prístupom iných užívateľov globálnej zóny k zóne Kirp141. Neoprávnené zásahy ostatných užívateľov zamedzíme odobratím prístupových práv k tomuto adresáru:

chmod 700 /export/home/zones/kirp141.

Pre správne fungovanie lokálnej zóny je nevyhnutné vytvoriť prepojujacie body k súborovému systému globálnej zóny. Vhodným prepojujacím bodom, za predpokladu, že v budúcnosti plánujeme použiť na aktualizáciu pôvodný baličkový systém pomocou nástroja pkdadd, je adresár /usr/local, pretože do tohto adresára sa automaticky umiestňujú jednotlivé nainštalované aplikácie pomocou pôvodného spôsobu aktualizácie systému. Keďže sa do tohto kroku neinštalovali žiadne aplikácie, je nevyhnutné tento adresár vytvoriť manuálne, aby sme mohli potom bezproblémovo pokračovať v konfigurácii vytváratej zóny Kirp141:

mkdir /usr/local.

2.1.3.1 Konfigurácia lokálnej zóny Kirp141

Po prípravných krokoch vytvárania lokálnej zóny Kirp141 môžeme pristúpiť k prideleniu názvu vytváratej zóny, ktoré sa uskutoční tak isto ako pri vytváraní zóny priradením k novému poolu pomocou príkazu:

zonecfg -z kirp141

kirp141: No such zone configured

Use 'create' to begin configuring a new zone.

zonecfg:kirp141>.

Proces vytvárania a konfigurácie zóny Kirp141 začneme:

zonecfg:kirp141> create.

Priradenie vytváratej lokálnej zóny Kirp141 do súborového systému uskutočnime pomocou set zonepath:

zonecfg:kirp141> set zonepath=/export/home/zones/kirp141.

Ďalším krokom konfigurácie lokálnej zóny Kirp141 je nastavenie bootovania zóny pri spustení, respektíve reštarte systému. Ak chceme zabezpečiť automatické nabootovanie zóny použijeme možnosť true, pretože predvolenou možnosťou je false:

zonecfg:kirp141> set autoboot=true.

V prípade, ak chceme, aby sme mohli so zónou Kirp141 pracovať tak isto ako s globálnou zónou, čiže sa k nej pripájať aj na diaľku, je nevyhnutné nakonfigurovať sieťové pripojenie, pričom konfigurovanie siete pre túto zónu začneme:

```
zonecfg:kirp141> add net.
```

Priradenie IP adresy 147.175.79.141 pre túto zónu uskutočníme:

```
zonecfg:kirp141:net> set address=147.175.79.141
```

```
zonecfg:kirp141:net> set physical=nge0.
```

Konfiguráciu sieťového pripojenia ukončíme:

```
zonecfg:kirp141:net> end.
```

V prípade, ak by sme chceli zóne Kirp141 prideliť internú adresu:

```
zonecfg:kirp141> add net
```

```
zonecfg:kirp141:net> set address=10.0.0.2
```

```
zonecfg:kirp141:net> set physical=nge0
```

```
zonecfg:kirp141:net> end.
```

Pričom takéto pridelenie internej IP adresy znemožní pripojenie sa zóny do internetu. Táto zóna by mala svoje opodstatnenie iba v rámci daného počítača, pretože by sa k nej nedalo priamo pripojiť cez internet.

Priradenie predvoleného poolu zóne Kirp141:

```
zonecfg:kirp141> set pool=pool_default.
```

Keďže v každej zóne budú dve aktívne aplikácie Apache server s podporou PHP a MySQL server, preto je nevyhnutné zabezpečiť, aby všetky zóny mali dostatok dostupných zdrojov. Všetky zóny sa budú deliť iba o jeden procesor, preto je nevyhnutné rozhodnúť podiely v správnom pomere medzi všetkými zónami. Pretože chceme, aby všetky zóny mali rovnaký prístup k zdrojom musíme im prideliť pomocou nástroja

FSS aj rovnaký počet podielov, bez ohľadu na to, či ide o globálnu zónu alebo lokálnu zónu, preto sme lokálnej zóne Kirp141 prideliť jeden podiel:

```
zonecfg:kirp141> add rctl
```

```
zonecfg:kirp141:rctl> set name=zone.cpu-shares
```

```
zonecfg:kirp141:rctl> add value (priv=privileged,limit=1,action=none).
```

Proces priradenia podielov ukončíme:

```
zonecfg:kirp141:rctl> end.
```

Nevyhnutným krokom konfigurácie je pridelenie konfigurovanej zóny Kirp141 do súborového systému pomocou `add fs`:

```
zonecfg:kirp141> add fs.
```

Kde pomocou:

```
zonecfg:kirp141:fs> set dir=/usr/local
```

označíme správny adresár.

Exportovanie adresára z globálnej zóny do novo vytváranej zóny sa uskutoční:

```
zonecfg:kirp141:fs> set special=/export/home/zones/kirp141/local.
```

Pomocou týchto dvoch príkazov zaistíme, že `/export/home/zones/kirp141/local` z globálnej zóny bude nastavený ako adresár `/usr/local/` v zóne Kirp141.

Dôležité je nastavenie oprávnení pre adresár, v našom prípade konkrétne pridelenie práv pre čítanie a písanie:

```
zonecfg:kirp141:fs> set options=[rw,nodevices].
```

Nastavenie typu súborového systému pre zónu Kirp141:

```
zonecfg:kirp141:fs> set type=lofs.
```

Súborový systém lofs je virtuálny súborový systém, ktorý poskytuje alternatívnu cestu do existujúceho súborového systému, pre adresáre ako: /bin, /platform, /sbin, /usr.

Proces prideľovania súborového systému zóne Kirp141 ukončíme:

zonecfg:kirp141:fs > **end**.

Pridelenie súborového systému zóne Kirp141 je posledný krok konfigurácie. Pred zapísaním konfigurácie zóny Kirp141 je vhodné overiť správnosť konfigurácie pomocou príkazu:

zonecfg:kirp141 > **verify**.

Správnu konfiguráciu zapíšeme:

zonecfg:kirp141 > **commit**.

Konfiguráciu ukončíme:

zonecfg:kirp141 > **exit**,

alebo:

zonecfg:kirp141 > **Ctrl + D**.

2.1.3.2 Inštalácia a naboťovanie Solaris zóny Kirp141

Ďalším krokom po konfigurácii, kontrole správnosti konfigurácie a zapísaní konfigurácie, je inštalácia zóny Kirp141. V tomto kroku sa otestuje, či všetky zdroje, ako napríklad sieťové rozhranie, sú správne stanovené v konfigurácii a či sú k dispozícii.[4]

Samostatná inštalácia zóny pozostáva z [4]:

- prípravy inštalácie,
- vytvorenia zoznamu súborov, ktoré budú prekopírované z globálnej zóny,
- inicializácie zóny,

-
- stanovenia potrebných balíčkov pre vytváranú zónu,
 - prípravy inicializácie balíčkov,
 - inicializácie balíčkov do zóny.

Inštalácia lokálnej zóny Kirp141:

zoneadm -z kirp141 install

Preparing to install zone <kirp141>.

Creating list of files to copy from the global zone.

Initializing zone product registry.

Determining zone package initialization order.

Preparing to initialize <1473> packages on the zone.

Initialized <1473> packages on zone.

Zone <kirp141> is initialized.

The file </export/home/zones/kirp141/root/var/sadm/system/logs/install_log> contains a log of the zone installation.

Zone <kirp141> is initialized.

Nabootovanie zóny Kirp141 uskutočníme:

zoneadm -z kirp141 boot.

2.1.3.3 Prvé prihlásenie sa do zóny Kirp141

Na prvé prihlásenie sa do zóny Kirp141 použijeme:

zlogin -C kirp141

[Connected to zone 'kirp141' console].

Po prvom prihlásení do zóny Kirp141 sme vyzvaní na výber:

- jazyka zóny,
- lokality zóny,
- typu využívaného terminálu.

Select a Language

- 0. English*
- 1. French*
- 2. German*
- 3. Italian*
- 4. Spanish*

Please make a choice (0 - 4), or press h or ? for help: 0.

Select a Locale

- 0. English (C - 7-bit ASCII)*
- 1. Albania (ISO8859-2)*
- 2. Albania (UTF-8)*
- 3. Belgium-Flemish (ISO8859-1)*

Press Return to show more choices.

Please make a choice (0 - 68), or press h or ? for help: 0.

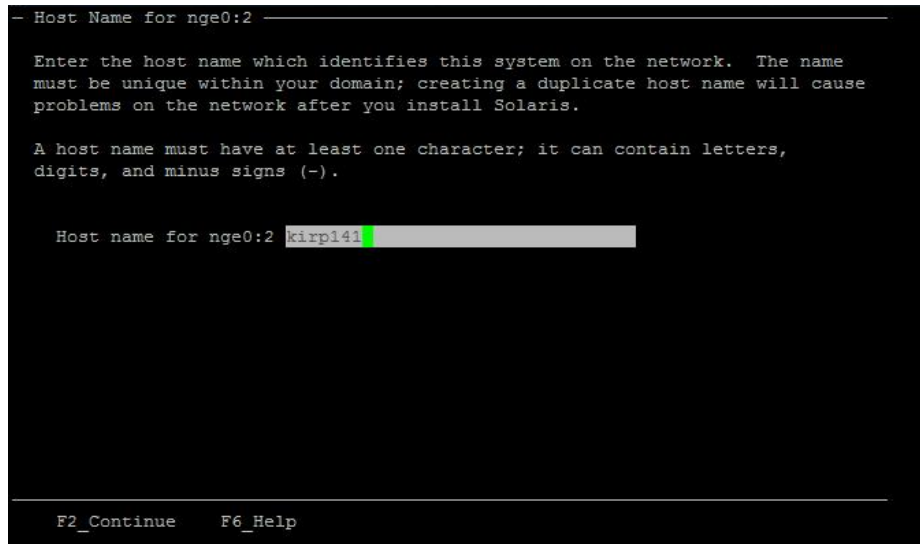
What type of terminal are you using?

- 1) ANSI Standard CRT*
- 2) DEC VT52*
- 3) DEC VT100*
- 4) Heathkit 19*
- 5) Lear Siegler ADM31*
- 6) PC Console*
- 7) Sun Command Tool*
- 8) Sun Workstation*
- 9) Televideo 910*
- 10) Televideo 925*
- 11) Wyse Model 50*
- 12) X Terminal Emulator (xterms)*
- 13) CDE Terminal Emulator (dtterm)*

14) Other

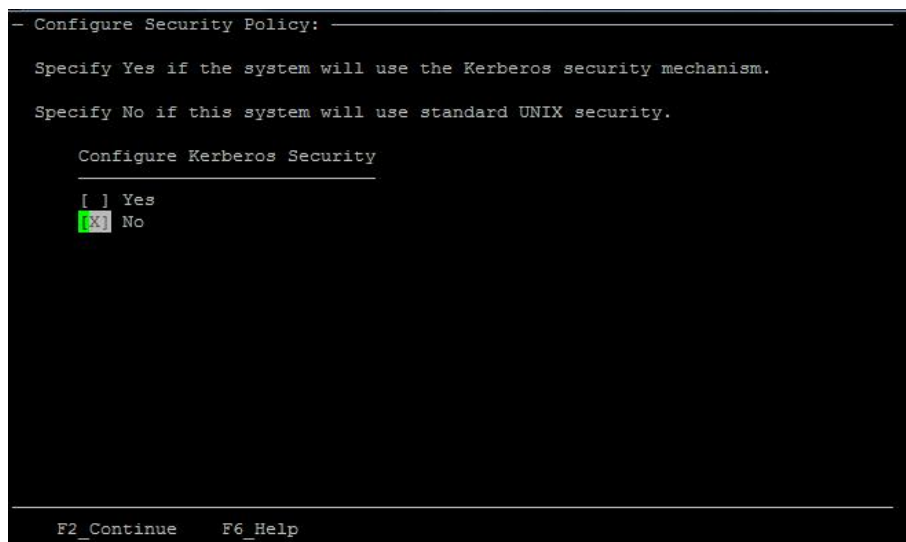
Type the number of your choice and press Return: **12**.

Nevyhnutné je zadanie mena počítača pre zónu Kirp141:



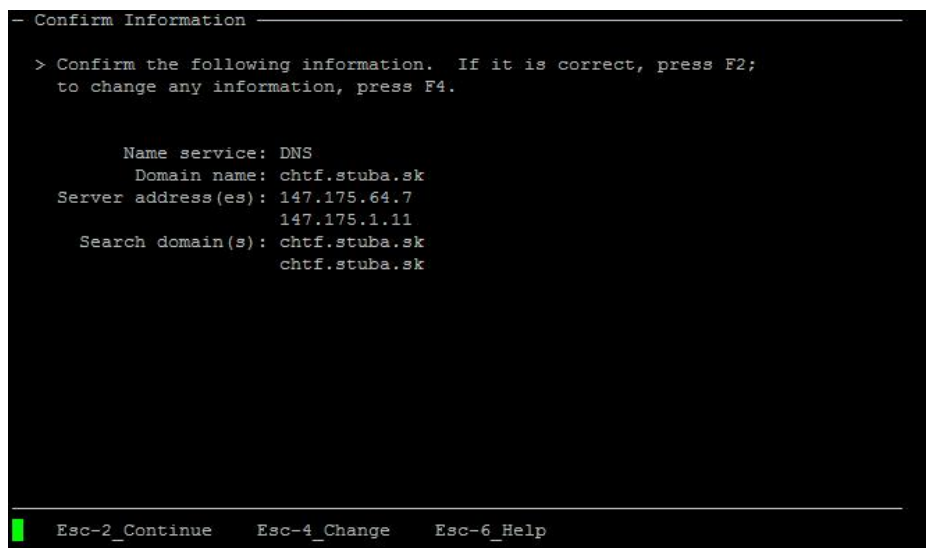
Obr. 9 Zadanie mena počítača pre zónu Kirp141

Zadané meno počítača potvrdíme pomocou F2, následne musíme venovať pozornosť nastaveniu bezpečnostnej politiky:



Obr. 10 Nastavenie bezpečnostnej politiky pre zónu Kirp141

Ďalším krokom nastavovania zóny Kirp141 je výber voliteľných služieb, kde je nevyhnutne nakonfigurovať DNS:



```
- Confirm Information -  
  
> Confirm the following information.  If it is correct, press F2;  
   to change any information, press F4.  
  
      Name service: DNS  
      Domain name:  chtf.stuba.sk  
Server address(es): 147.175.64.7  
                  147.175.1.11  
      Search domain(s): chtf.stuba.sk  
                      chtf.stuba.sk  
  
Esc-2_Continue  Esc-4_Change  Esc-6_Help
```

Obr. 11 Konfigurácia DNS pre zónu Kirp141

Po nastavení voliteľných služieb je nevyhnutné nastavenie NFSv4 domény, nastavenie časovej zóny, nastavenie regiónu a nastavenie hesla pre superužívateľa root.

Po nastavení jednotlivých parametrov zóny Kirp141 čakáme na odpoveď systému v podobe:

Creating new rsa public/private host key pair

Creating new dsa public/private host key pair

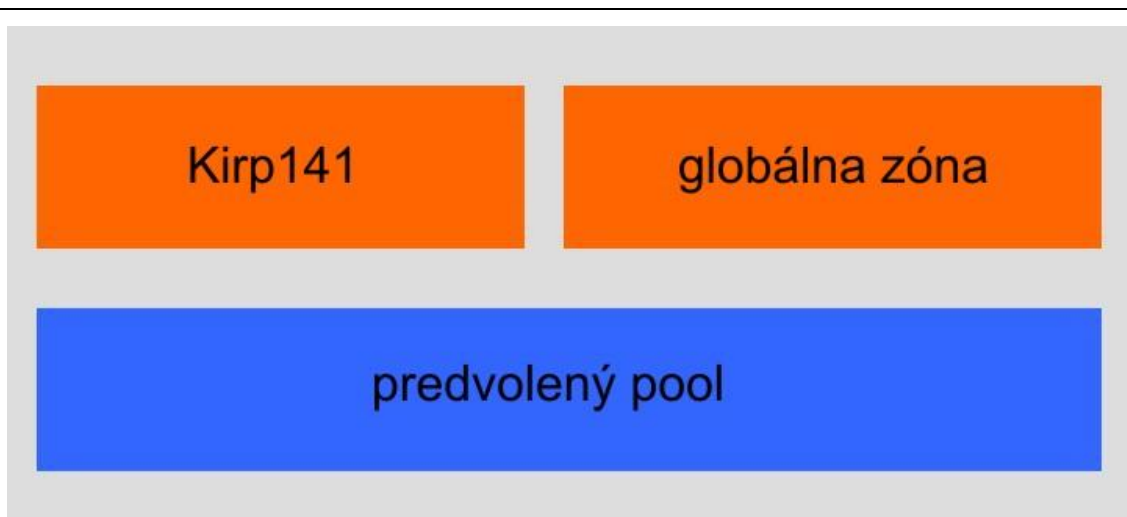
Configuring network interface addresses: nge0.

System identification is completed.

rebooting system due to change(s) in /etc/default/init.

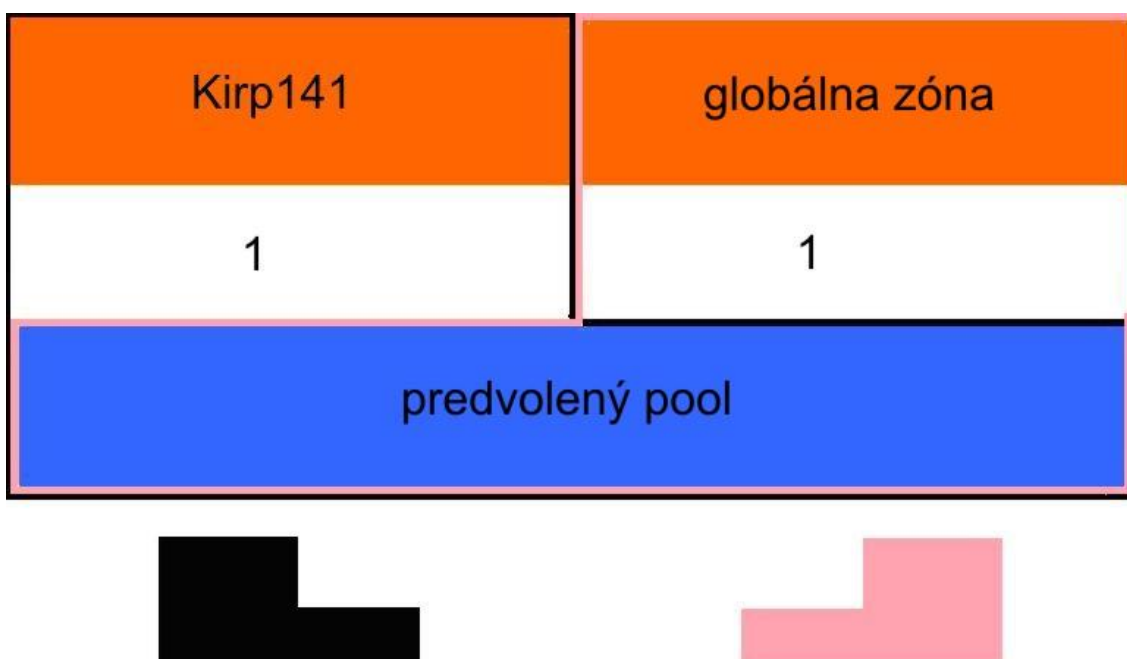
Z prihlásenej zóny Kirp141 sa môžeme odhlásiť zadaním ~, kde po zadaní ~ sme automaticky prihlásení v globálnej zóne Kirp136.

Po nainštalovaní lokálnej zóny Kirp141 máme v systéme 2 zóny - jednu globálnu zónu Kirp136, druhú lokálnu zónu Kirp141, ktoré zdieľajú spoločne zdroje predvoleného poolu:



Obr. 12 Ukážka lokálnej zóny Kirp141 a globálnej zóny

V systéme nám vzniknú dva Solaris kontajnery, tak ako je to znázornené na obr. 13:



Obr. 13 Ukážka lokálnej zóny Kirp141, globálnej zóny a dvoch kontajnerov

2.1.3.4 Rýchle nastavenie lokálnej zóny Kirp141

Po vytvorení lokálnej zóny Kirp141 je nevyhnutné upraviť parametre zóny cieľom, ktorých je urýchliť a zjednodušiť si ďalšiu prácu vo vytvorenej zóne.

Informácie o systéme získame:

\$ uname -a

SunOS kirp141 5.10 Generic_139556-08 i86pc i386 i86pc.

To, či vytvorená lokálna zóna obsahuje všetky adresáre ako globálna zóna, zistíme pomocou:

ls

*core etc home mnt opt proc system usr
bin dev export lib net platform sbin tmp var.*

Neustálym prihlasovaním sa ako root užívateľ sa vystavujeme množstvu bezpečnostných rizík, preto je vhodné, ak chceme pracovať v zóne na bežných veciach, ktoré si nevyžadujú prihlásenie sa ako root, prihlasovať sa ako bežný užívateľ. Za predpokladu, že sa chceme prihlasovať do zóny priamo cez vzdialené prihlasovanie pomocou SSH je vhodné si vytvoriť nového užívateľa, ktorý nebude mať práva užívateľa root. Takto sa vyhneme množstvu bezpečnostných rizík. Takéto priame prihlasovanie, prostredníctvom bežného užívateľa, je nevyhnutné za predpokladu, že v konfiguračnom súbore SSH je zakázané priame prihlasovanie sa ako superužívateľ. [4]

Nového užívateľa pokus vytvoríme:

useradd -d /export/home/pokus -m -s /bin/bash pokus,

pričom okrem samostatného vytvorenia užívateľa pokus zabezpečíme aby sa nám automaticky po prihlásení na užívateľa pokus zapol bash, a následne vytvoríme a pridelieme užívateľovi pokus domovský adresár /export/home/pokus.

Heslo novovytvorenému užívateľovi pokus pridelieme:

passwd pokus

*New Password:******

*Re-enter new Password:******

passwd: password successfully changed for pokus.

Automatické spustenie bashu pre užívateľa root zabezpečíme:

usermod -s /bin/bash root.

Po správnom vytvorení užívateľa pokus môžeme otestovať možnosť vzdialeného pripojenia sa do zóny Kirp141 pomocou SSH:

```
-bash-3.00$ ssh pokus@147.175.79.141
```

```
The authenticity of host '147.175.79.141 (147.175.79.141)' can't be established.
```

```
RSA key fingerprint is ba:fb:2f:b5:1f:6f:36:a9:68:25:7g:19:1e:56:91:1f.
```

```
Are you sure you want to continue connecting (yes/no)? yes
```

```
Password: *****
```

```
Last login: Sun Mar 21 21:20:16 2010 from 147.175.221.215
```

```
Sun Microsystems Inc. SunOS 5.10 Generic January 2005
```

```
-bash-3.00$.
```

V prípade, ak by sme mali záujem používať v lokálnej zóne Kirp141 na aktualizáciu, inštaláciu a odinštalovanie jednotlivých aplikácií nástroj pkgutil je vhodné zabezpečiť, aby PATH cesta bola automaticky nastavená ako [4]:

```
/usr/sbin:/usr/bin:/opt/csw/bin.
```

Informácie o nastavenej PATH ceste systému získame:

echo \$PATH

```
/usr/sbin:/usr/bin.
```

V prípade, ak by sme mali záujem zabezpečiť automatické nastavenie PATH cesty na /usr/sbin:/usr/bin:/opt/csw/bin po prihlásení sa hociktorého užívateľa v systéme je nevyhnutné v súbore /etc/default/login a v súbore /etc/default/su pridať na správne miesto nasledujúce dva riadky [4]:

```
PATH=/usr/bin:/opt/csw/bin
```

```
SUPATH=/usr/sbin:/usr/bin:/opt/csw/bin.
```

Po reštartovaní lokálnej Solaris zóny Kirp141 môžeme overiť správnosť nastavenej cesty:

echo \$PATH

/usr/sbin:/usr/bin:/opt/csw/bin.

Tieto kroky zjednodušujúce a urýchľujúce prácu je vhodné analogicky aplikovať aj v globálnej zóne.

2.1.4 Vytvorenie lokálnej zóny Kirp140

Na vytvorenie lokálnej Solaris zóny Kirp140 môžeme použiť dva spôsoby vytvárania Solaris zón:

- vytvorenie lokálnej zóny Kirp140 a jej priradenie k predvolenému poolu,
- vytvorenie lokálnej zóny Kirp140 klonovaním lokálnej zóny Kirp141.

2.1.4.1 Vytvorenie zóny Kirp140 priradením k predvolenému poolu

Tento spôsob vytvorenia lokálnej zóny Kirp140 je oproti spôsobu vytvorenia klonovaním lokálnej zóny Kirp141 zdĺhavejší, navyše po vytvorení tejto zóny je v nej nevyhnutné uskutočniť kroky, ktoré urýchľujú a zjednodušujú prácu v zóne nanovo.

Pri konfigurácii, inštalácii, naboťovaní a prvom prihlásení sa do zóny Kirp140 postupujeme analogicky ako v prípade vytvárania lokálnej zóny Kirp141 pridelením k predvolenému poolu.

2.1.4.2 Vymazanie lokálnej zóny Kirp140

Skôr ako začneme vytvárať lokálnu zónu Kirp140 klonovaním lokálnej zóny Kirp141, je vhodné nazrieť do zoznamu už existujúcich zón systému:

zoneadm list -vc

<i>ID NAME</i>	<i>STATUS</i>	<i>PATH</i>	<i>BRAND</i>	<i>IP</i>
0 global	running	/	native	shared
1 kirp140	running	/export/home/zones/kirp140	native	shared
2 kirp141	running	/export/home/zones/kirp141	native	shared.

Z výpisu je jasné, že v systéme máme 3 zóny, globálnu zónu Kirp136 a dve lokálne zóny Kirp140 a Kirp141, ktoré sme vytvorili a následne priradili k predvolenému poolu. Pretože sme už lokálnu zónu Kirp140 vytvorili, musíme ju vymazať. Prvým krokom vymazania zóny je zastavenie zóny Kirp140:

zoneadm -z kirp140 halt.

Po pozastavení zóny Kirp140 môžeme pristúpiť k jej odinštalovaniu:

zoneadm -z kirp140 uninstall

Are you sure you want to uninstall zone kirp140 (y/[n])? Y.

To, či sme lokálnu zónu Kirp140 správne vymazali overíme pomocou:

zoneadm list -vc

<i>ID NAME</i>	<i>STATUS</i>	<i>PATH</i>	<i>BRAND</i>	<i>IP</i>
0 global	running	/	native	shared
- kirp140	configured	/export/home/zones/kirp140	native	shared
1 kirp141	running	/export/home/zones/kirp141	native	shared.

Zo získaného výpisu je zreteľné, že lokálna zóna je ešte v režime konfigurácie, preto je nevyhnutné zabezpečiť vymazanie konfigurácie zóny:

zonecfg -z kirp140 delete.

Are you sure you want to delete zone kirp140 (y/[n])? Y.

Po vymazaní konfigurácie lokálnej zóny Kirp140 je vhodné vymazať pridelené miesto na disku, ktoré sme zóne Kirp140 prideliť:

rm -r /export/home/zones/kirp140.

Úplnosť vymazanie zóny Kirp140 overíme:

zoneadm list -vc

<i>ID NAME</i>	<i>STATUS</i>	<i>PATH</i>	<i>BRAND</i>	<i>IP</i>
<i>0 global</i>	<i>running</i>	<i>/</i>	<i>native</i>	<i>shared</i>
<i>1 kirp141</i>	<i>running</i>	<i>/export/home/zones/kirp141</i>	<i>native</i>	<i>shared.</i>

2.1.4.3 Vytvorenie zóny Kirp140 klonovaním lokálnej zóny Kirp141

Rýchlejším spôsobom vytvorenia lokálnej zóny Kirp140 je vytvorenie pomocou klonovania lokálnej zóny Kirp141. Hlavná výhoda tohto spôsobu vytvárania je v tom, že vzniknuté zóny sú takmer identické. Rozdiel medzi oboma zónami je iba v parametroch, ktoré lokálnej zóne Kirp140 priradíme po prvom prihlásení sa do zóny. To znamená, že v lokálnej zóne Kirp141 nebudeme musieť znovu vykonať kroky na urýchlenie a zjednodušenie práce.

Z dôvodu, že ideme vytvárať lokálnu zónu Kirp140 klonovaním lokálnej zóny Kirp141, musíme lokálnu zónu Kirp141 zastaviť:

```
# zoneadm -z kirp141 halt.
```

Prípravným krokom konfigurácie zóny Kirp140 je vyčlenenie miesta pre túto zónu a pridelenie prístupových práv pre toto miesto:

```
# mkdir -p /export/home/zones/kirp140/local
```

```
# chmod 700 /export/home/zones/kirp140.
```

Prvým krokom konfigurácie lokálnej zóny Kirp140 je exportovanie konfigurácie lokálnej zóny Kirp141 do súboru nazvaného ako master:

```
# zonecfg -z kirp141 export -f /export/zones/master.
```

Vytvorený súbor master je nevyhnutné upraviť tak, aby sme pomocou tohto súboru mohli bezchybne vytvoriť lokálnu zónu Kirp140 podľa stanovených požiadaviek na lokálnu zónu Kirp140:

```
create -b
```

```
set zonepath=/export/home/zones/kirp140
```

```
set autoboot=true
set pool=pool_default
set ip-type=shared
add inherit-pkg-dir
set dir=/lib
end
add inherit-pkg-dir
set dir=/platform
end
add inherit-pkg-dir
set dir=/sbin
end
add inherit-pkg-dir
set dir=/usr
end
add fs
set dir=/usr/local
set special=/export/home/zones/kirp140local
set type=lofs
add options rw
add options nodevices
end
add net
set address=147.175.79.140
set physical=nge0
end
add rctl
set name=zone.cpu-shares
add value (priv=privileged,limit=1action=none)
```

end.

Vytvorenie lokálnej zóny Kirp140 pomocou príkazov v súbore master:

zonecfg -z kirp140 -f /export/zones/master.

Inštalácia lokálnej zóny Kirp140 klonovaním lokálnej zóny Kirp141:

zoneadm -z kirp140 clone kirp141.

Nabootovanie lokálnej zóny Kirp140:

zoneadm -z kirp140 boot.

Prvé prihlásenie sa do zóny uskutočníme:

zlogin -C kirp140.

Po prvom prihlásení sa do lokálnej zóny Kirp140 sme automaticky vyzvaný na nastavenie parametrov lokálnej zóny Kirp140, čiže na:

- výber jazyka zóny,
- lokality zóny,
- typu využívaného terminálu,
- zadanie mena počítača,
- nastavenie bezpečnostnej politiky,
- výber voliteľných služieb, kde je nevyhnutné nakonfigurovať DNS,
- nastavenie NFSv4 domény,
- nastavenie časovej zóny,
- nastavenie regiónu,
- nastavenie hesla pre superužívateľa root.

Pre správne fungovanie lokálnej zóny Kirp141 je potrebné zónu Kirp141 nabootovať:

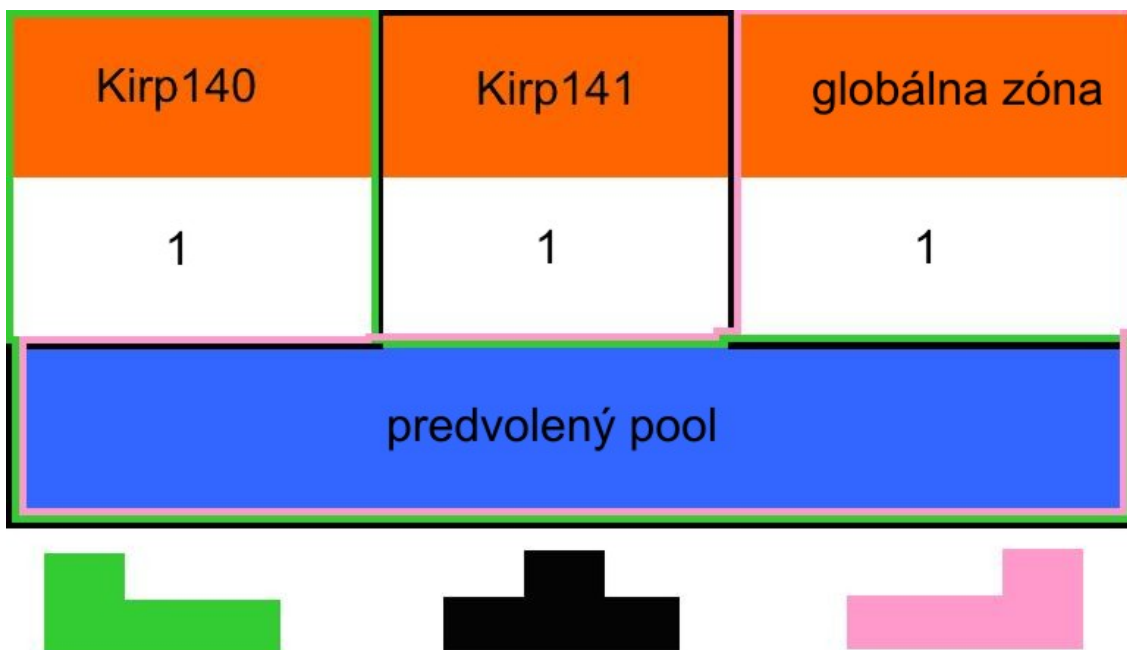
zoneadm -z kirp141 boot.

Klonovaním lokálnej zóny Kirp141 sme zabezpečili aj automatické vytvorenie všetkých užívateľov lokálnej zóny Kirp141 v lokálnej zóne Kirp140.

Po vytvorení lokálnej zóny Kirp140 klonovaním lokálnej zóny Kirp141 máme v systéme 3 zóny - jednu globálnu zónu a dve lokálne zóny, pričom všetky zóny sa delia o zdroje z predvoleného poolu:



Obr. 14 Ukážka globálnej zóny, lokálnych zón Kirp141 a Kirp140



Obr. 15 Ukážka globálnej zóny, 2 lokálnych zón a 3 kontajnerov

2.1.5 Aktualizácia systému a inštalácia aplikácií

Aktualizácia a inštalácia jednotlivých aplikácií v OS Solaris 10 sú založené na balíčkovacom systéme. Takýto balíčkovací systém aktualizácie systému je veľmi jednoduchý, pretože sa tu vykonáva automatická kontrola závislosti jednotlivých balíčkov. Výhoda takéhoto systému je v tom, že ak nastane situácia, že niektoré balíčky potrebné pre fungovanie práve inštalovanej aplikácie nie sú nainštalované, dôjde k automatickému nainštalovaniu potrebných balíčkov.

Jednotlivé balíčky môžu pochádzať z troch rôznych zdrojov:

- spoločnosť SUN Microsystems,
- komunita SunFreeware,
- komunita Blastwave.

Pri inštalácii jednotlivých aplikácií sme využili dva nástroje:

- pkgadd,
- pkgutil.

Pkgadd je súčasťou operačného systému, preto jeho inštalácia užívateľom root nie je potrebná. Nástroj pkgutil nie je súčasťou operačného systému, preto je nevyhnutné tento nástroj nainštalovať prostredníctvom root užívateľa globálnej zóny.

Pred nainštalovaním je potrebné pkgutil najprv stiahnuť:

http://download.blastwave.org/csw/pkgutil_i386.pkg.

Po stiahnutí tohto nástroja a nastavení sa do adresára, v ktorom je uložený stiahnutý balíček, môžeme prísť k inštalácii tohto nástroja pomocou pkgadd:

pkgadd -d pkgutil_i386.pkg.

Po nainštalovaní je vhodné zistiť či nami nainštalovaná verzia pkgutil je najnovšia, ak nami nainštalovaná verzia nie je najnovšia, je vhodné ju aktualizovať na najnovšiu verziu:

pkgutil -u pkgutil.

Po aktualizovaní nástroja pkgutil môžeme pristúpiť k inštalácii Apache servera, podpory PHP pre Apache server a MySQL servera vo všetkých zónach. Základnou výhodou je, že ak budeme chcieť inštalovať jednotlivé aplikácie vo všetkých zónach naraz, stačí sa nastaviť do globálnej zóny, kde potom pomocou nástroja pkgutil môžeme inštalovať vybrané aplikácie naraz vo všetkých zónach. V prípade, ak budeme chcieť zabezpečiť, aby sme vybrané aplikácie inštalovali iba v globálnej zóne, budeme musieť v konfiguračnom súbore pkgutil /etc/opt/csw/pkgutil.conf odpoznamkovať riadok:

```
# pkgaddopts=-G.
```

2.1.5.1 Inštalácia a konfigurácia Apache servera

Apache server je softvérový webový server. Jeho základnou výhodou je, že má Open source licenciu, čo znamená, že jeho šírenie je voľné. V internetovom svete mu patrí prvenstvo v rozšírenosti HTTP serverov, pretože ho používa takmer polovica serverov. Tento HTTP server bol pôvodne vyvinutý pre UNIXové operačné systémy, ale pre svoju popularitu sa vytvorili aj mutácie pre iné platformy, najmä pre Windows.

Najjednoduchším spôsobom ako nainštalovať Apache server a všetky nevyhnutné balíčky potrebné pre správne a bezchybné fungovanie Apache servera vo všetkých zónach naraz je inštalácia pomocou pkgutil v globálnej zóne:

```
# pkgutil -i -y apache2.
```

Pretože na inštaláciu Apache2 servera a všetkých ostatných balíčkov potrebných pre správne a bezchybné fungovanie Apache servera bol použitý nástroj pkgutil, nie je potrebné vytvárať konfiguračný súbor httpd.conf. Ten sa vytvorí automaticky pri inštalácii Apache servera pomocou pkgutil.

Test správnosti konfigurácie Apache servera v jednotlivých zónach samostatne uskutočníme:

```
# /opt/csw/apache2/sbin/apachectl configtest
```

Syntax OK.

Spustenie, respektíve vypnutie, servera vo všetkých zónach samostatne:

/opt/csw/apache2/sbin/apachectl start

/opt/csw/apache2/sbin/apachectl stop.

Apache servery nachádzajúce sa v jednotlivých zónach majú svoje vlastné logy. V logu **/opt/csw/apache2/var/log/access_log** v jednotlivých zónach nájdeme informácie o úspešných prihláseniach. V logu **/opt/csw/apache2/var/log/error_log** v jednotlivých zónach nájdeme informácie o neúspešných prihláseniach.

Keďže sme na inštaláciu a konfiguráciu Apache servera použili nástroj pkgutil, nemusíme manuálne nastaviť automatické spustenie Apache servera po znovu spustení zón, pretože nástroj pkgutil nám zabezpečí automatické spustenie Apache servera po spustení zón, bez ohľadu na to, či ide o globálnu alebo lokálnu zónu. Automatické spustenie Apache servera v jednotlivých zónach môžeme overiť:

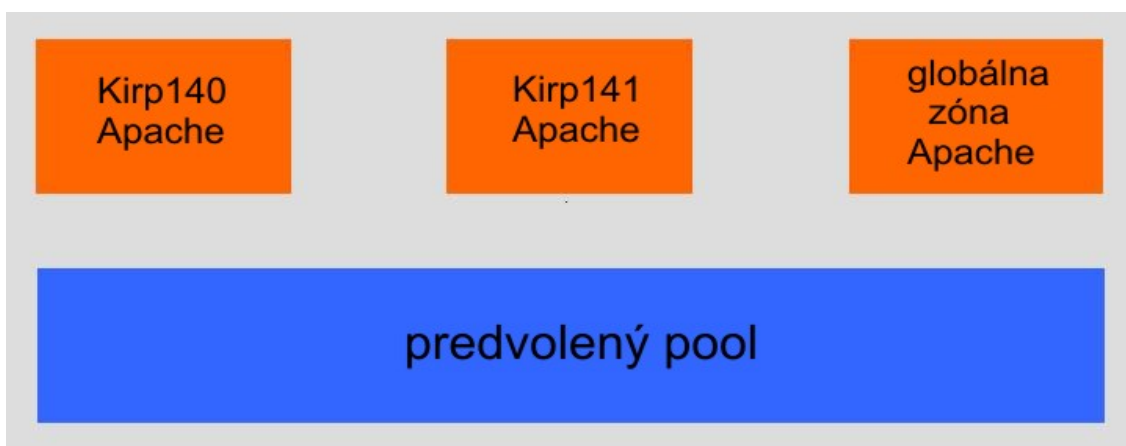
ps -ef | grep apache,

nobody 926 830 0 mar 19 ? 0:00 /opt/csw/apache2/sbin/httpd -k start

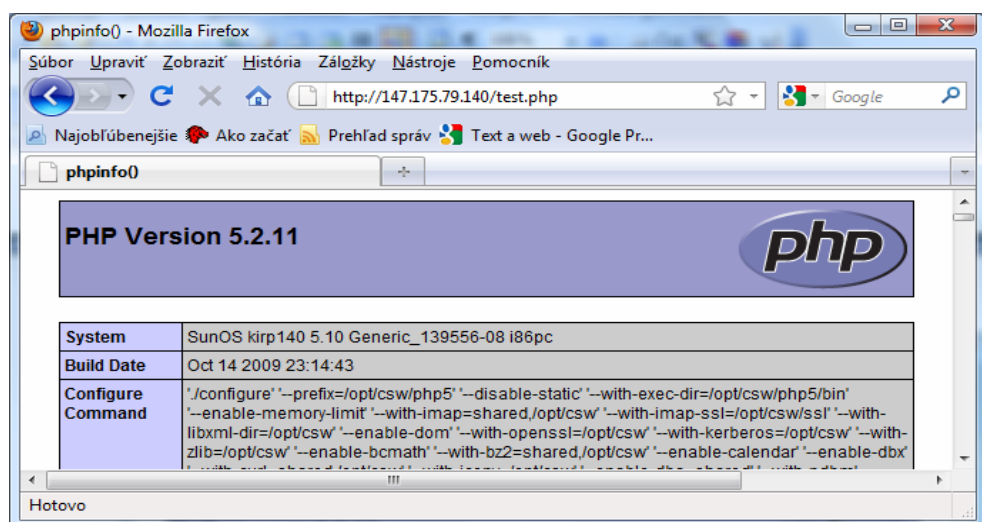
nobody 2238 2108 0 mar 19 ? 0:00 /opt/csw/apache2/sbin/httpd -k start

root 830 1 0 mar 19 ? 0:07 /opt/csw/apache2/sbin/httpd -k start.

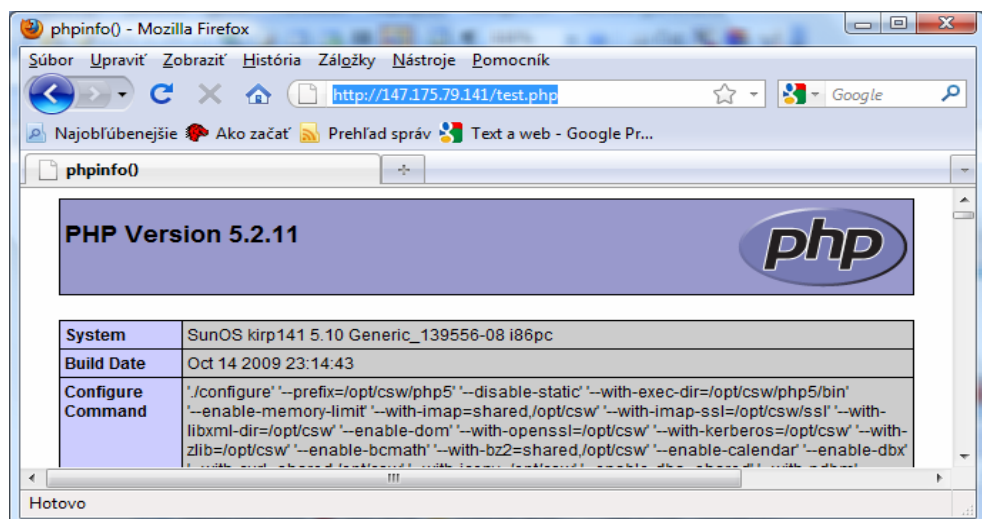
Správnosť inštalácie a konfigurácie Apache servera môžeme otestovať aj pomocou internetového prehliadača, do ktorého napíšeme prislúchajúce IP adresy jednotlivých Apache serverov jednotlivých zón.



Obr. 16 Konfigurácia systému po nainštalovaní Apache serverov



Obr. 18 Test správnosti konfigurácie PHP Kirp140



Obr. 19 Test správnosti konfigurácie PHP Kirp141



Obr. 20 Konfigurácia systému po nainštalovaní podpory PHP

2.1.5.3 Inštalácia a konfigurácia MySQL servera

MySQL je v súčasnej dobe najpoužívanější databázový server. Najjednoduchším spôsobom ako nainštalovať, nakonfigurovať a zabezpečiť automatické spustenie MySQL servera po znovu spustení konkrétnych zón vo všetkých zónach naraz je inštalácia pomocou nástroja pkgutil v globálnej zóne:

```
# pkgutil -i -y mysql5.
```

Dôležitým krokom je inicializáciu MySQL databáz a nastavenie oprávnení:

```
cd /opt/csw/mysql5
```

```
./bin/mysql_install_db
```

```
chown -R mysql:mysql ./var.
```

Spustiteľnosť MySQL servera v jednotlivých zónach samostatne môžeme overiť:

```
# cd /opt/csw/mysql5; /opt/csw/mysql5/bin/mysqld_safe &.
```

Po reštartovaní jednotlivých zón môžeme otestovať automatické spustenie MySQL serverov:

```
# ps -ef | grep mysql
```

```
root      321          1      0      mar 19  ?                0:00 /bin/sh
/opt/csw/mysql5/bin/amd64/mysqld_safe --pid-file=/opt/csw/mysql5/var/my
mysql  549  321  0  mar 19  ?                4:09 /opt/csw/mysql5/libexec/amd64/mysqld --
basedir=/opt/csw/mysql5 --datadir=/opt/c
root      1277          1      0      mar 19  ?                0:00 /bin/sh
/opt/csw/mysql5/bin/amd64/mysqld_safe --pid-file=/opt/csw/mysql5/var/my.
```

Prvému prihláseniu na MySQL server predchádza pridelenie hesla pre MySQL užívateľov:

```
# /opt/csw/mysql5/bin/amd64/mysqladmin -u root password 'new-password'
```

```
# /opt/csw/mysql5/bin/amd64/mysqladmin -u root -h kirp136 password 'new-
password'.
```

Proces pridelenia hesla opakujeme v každej zóne samostatne, aby sme mohli pridelit', z bezpečnostných dôvodov, každému MySQL užívateľovi iné heslo.

Spustiteľnosť MySQL serverov v jednotlivých zónach môžeme otestovať:

```
# cd /opt/csw/mysql5
```

```
# ./bin/mysql -p.
```

Po zadaní **./bin/mysql -p** a zadaní správneho hesla sa úspešne prihlásime do MySQL:

```
mysql> STATUS
```

```
-----
```

```
./bin/mysql Ver 14.12 Distrib 5.0.75, for pc-solaris2.10 (i386) using readline 5.2
```

```
Connection id: 1
```

```
Current database:
```

```
Current user: root@localhost
```

```
SSL: Not in use
```

```
Current pager: stdout
```

```
Using outfile: "
```

```
Using delimiter: ;
```

```
Server version: 5.0.75 Source distribution
```

```
Protocol version: 10
```

```
Connection: Localhost via UNIX socket
```

```
Server characterset: latin1
```

```
Db characterset: latin1
```

```
Client characterset: latin1
```

```
Conn. characterset: latin1
```

```
UNIX socket: /tmp/mysql.sock
```

```
Uptime: 5 days 9 hours 9 min 19 sec
```

```
Threads: 1 Questions: 4 Slow queries: 0 Opens: 12 Flush tables: 1 Open tables: 6
```

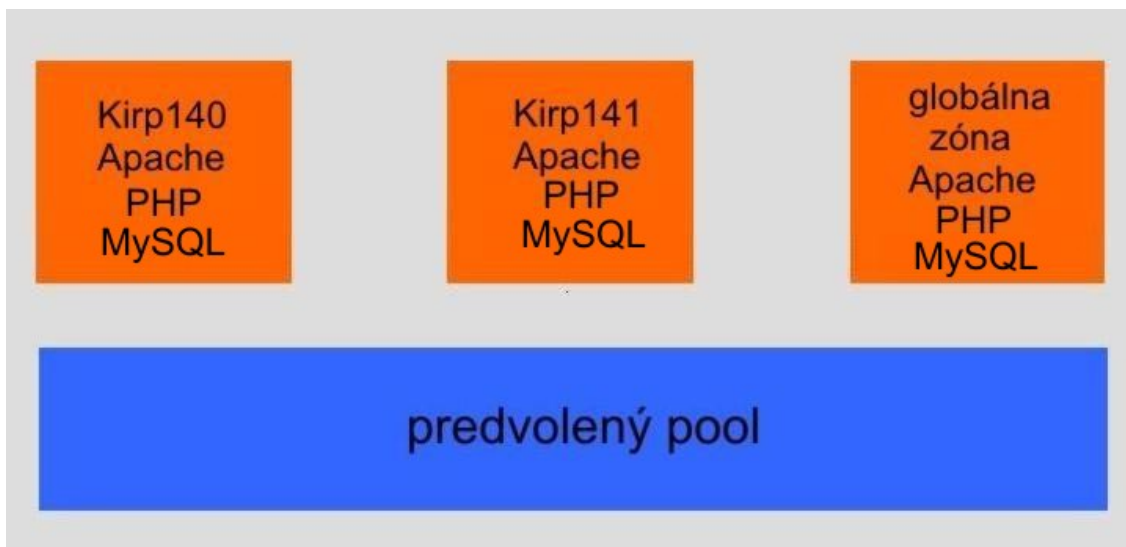
```
Queries per second avg: 0.000
```

```
-----.
```

Prácu s MySQL serverom ukončíme:

mysql> quit

Bye.



Obr. 21 Konfigurácia systému po nainštalovaní MySQL servera

2.1.6 Presun lokálnej zóny Kirp141 z globálnej zóny Kirp136 na Kirp138

Keďže chceme presunúť lokálnu zónu Kirp141 na iný hostujúci počítač, musíme zónu Kirp141 najprv podržať:

zoneadm -z kirp141 halt.

Po podržaní lokálnej zóny Kirp141 je nevyhnutné zónu Kirp141 odpojiť:

zoneadm -z kirp141 detach.

Je dôležité presunúť sa do adresára /export/home/zones a zabaliť všetky súbory súvisiace s zónou Kirp141 do jedného balíčka:

cd /export/home/zones

tar cf kirp141.tar kirp141.

Vytvorený balíček je nevyhnutné prekopírovať na hostujúci počítač, na ktorý budeme pomocou migrácie prenášať lokálnu zónu Kirp141. Najvhodnejším spôsobom na kopírovanie je kopírovanie pomocou SFTP:

```
# sftp pokus@147.175.79.138
```

```
sftp> cd /pomocna
```

```
sftp> put kirp141.tar
```

```
Uploading kirp141.tar to /pomocna/ kirp141.tar.
```

Potrebuje baliček kirp141.tar prekopírovať, poprípade presunúť, z adresára /pomocna do adresára /export/home/zones:

```
# cp -r kirp141.tar /export/home/zones/kirp141.tar.
```

Balíček kirp141.tar je potrebné rozbaľiť:

```
# cd /export/home/zones
```

```
# tar xf kirp141.tar.
```

Pre lepšiu orientáciu v zónach je výhodné premenovať kirp141 na kirp143:

```
# mv kirp141 kirp143.
```

Naším cieľom je mať v prevádzke migrovanú zónu Kirp141 na oboch hostujúcich počítačoch, preto musíme zónu na novom hostujúcom počítači premenovať, napríklad na Kirp143, po premenovaní môžeme prísť ku konfigurácii lokálnej zóny Kirp143:

```
# zonecfg -z kirp143
```

```
kirp143: No such zone configured
```

```
Use 'create' to begin configuring a new zone.
```

Priradenie vytváranej zóny do súborového systému:

```
zonecfg:kirp143> create -a /export/home/zones/kirp143.
```

Informácie o konfigurácii vytváranej zóny Kirp143 získame:

zonecfg:kirp143> info

zonename: kirp143

zonepath: /export/home/zones/kirp143

brand: native

autoboot: true

bootargs:

pool: pool_default

limitpriv:

scheduling-class:

ip-type: shared

[cpu-shares: 1]

inherit-pkg-dir:

dir: /lib

inherit-pkg-dir:

dir: /platform

inherit-pkg-dir:

dir: /sbin

inherit-pkg-dir:

dir: /usr

fs:

dir: /usr/local

special: /export/home/zones/kirp141/local

raw not specified

type: lofs

options: [rw,nodevices]

net:

address: 147.175.79.141

physical: nge0

defrouter not specified

rctl:

name: zone.cpu-shares

value: (priv=privileged,limit=1,action=none).

Pretože sme zmenili súborový systém lokálnej zóny Kirp143 musíme vymazať celú časť konfigurácie zameranú na priradenie súborového systému:

zonecfg:kirp143> remove fs.

Pretože naším cieľom je mať súbežné bežiacu pôvodnú lokálnu zónu Kirp141 a presunutú lokálnu zónu Kirp143, ktorá nám vznikla migráciou lokálnej zóny Kirp141 na iný hostujúci počítač, musíme vymazať v konfigurácii lokálnej zóny Kirp143 pridelenú IP adresu, aby sme predišli skutočnosti, že obe zóny budú mať pridelenú tú istú IP adresu:

zonecfg:kirp143> remove net.

Úspešnosť vymazania sieťového pripojenia a priradenia zóny do súborového systému v konfigurácii zóny Kirp143 overíme:

zonecfg:kirp143> info

zonename: kirp143

zonepath: /export/home/zones/kirp143

brand: native

autoboot: true

bootargs:

pool: pool_default

limitpriv:

scheduling-class:

ip-type: shared

[cpu-shares: 1]

inherit-pkg-dir:

```
dir: /lib
inherit-pkg-dir:
dir: /platform
inherit-pkg-dir:
dir: /sbin
inherit-pkg-dir:
dir: /usr
rctl:
name: zone.cpu-shares
value: (priv=privileged,limit=1,action=none).
```

Pre správne fungovanie vytváranej lokálnej zóny Kirp143 je nevyhnutné nanovo uskutočniť sieťovú konfiguráciu zóny:

```
zonecfg:kirp143> add net
zonecfg:kirp143:net> set address=147.175.79.143
zonecfg:kirp143:net> set physical=ng0
zonecfg:kirp143:net> end.
```

Pre fungovanie zóny je potrebné jej pridelenie do súborového systému:

```
zonecfg:kirp143> add fs
zonecfg:kirp143:fs> set dir=/usr/local
zonecfg:kirp143:fs> set special=/export/home/zones/kirp143/local
zonecfg:kirp143:fs> set options=[rw,nodevices]
zonecfg:kirp143:fs> set type=lofs
zonecfg:kirp143:fs> end.
```

Úspešnosť zapísania sieťovej konfigurácie a priradenia lokálnej zóny Kirp143 do súborového systému otestujeme:

```
zonecfg:kirp143> info
zonename: kirp143
```

zonepath: /export/home/zones/kirp143

brand: native

autoboot: true

bootargs:

pool: pool_default

limitpriv:

scheduling-class:

ip-type: shared

[cpu-shares: 1]

inherit-pkg-dir:

dir: /lib

inherit-pkg-dir:

dir: /platform

inherit-pkg-dir:

dir: /sbin

inherit-pkg-dir:

dir: /usr

fs:

dir: /usr/local

special: /export/home/zones/kirp143/local

raw not specified

type: lofs

options: [rw,nodevices]

net:

address: 147.175.79.143

physical: nge0

defrouter not specified

rctl:

name: zone.cpu-shares

value: (priv=privileged,limit=1,action=none).

Po správnom zapísaní sieťovej konfigurácie a pridelení zóny do súborového systému môžeme prísť ku overeniu, zapísaniu a ukončeniu konfigurácie:

zonecfg:kirp143> verify

zonecfg:kirp143> commit

zonecfg:kirp143> exit.

Po nakonfigurovaní je nevyhnutné nakonfigurovanú zónu Kirp143 pripojiť:

zoneadm -z kirp143 attach -u

Getting the list of files to remove

Removing 269 files

Remove 77 of 77 packages

Installing 1175 files

Add 26 of 26 packages

Updating editable files

The file </var/sadm/system/logs/update_log> within the zone contains a log of the zone update.

Po pripojení zóny Kirp143 je nevyhnutné nazrieť do zoznamu existujúcich zón:

zoneadm list -vc

<i>ID</i>	<i>NAME</i>	<i>STATUS</i>	<i>PATH</i>	<i>BRAND</i>	<i>IP</i>
<i>0</i>	<i>global</i>	<i>running</i>	<i>/</i>	<i>native</i>	<i>shared</i>
<i>1</i>	<i>kirp142</i>	<i>running</i>	<i>/export/home/zones/kirp142</i>	<i>native</i>	<i>shared</i>
<i>-</i>	<i>kirp143</i>	<i>installed</i>	<i>/export/home/zones/kirp143</i>	<i>native</i>	<i>shared.</i>

Zo zoznamu je zreteľné, že je nevyhnutné pripojiť zónu naboťovať:

zoneadm -z kirp143 boot.

Potom bude zo zoznamu zón zreteľné, že novo vytvorená lokálna zóna Kirp143 je v prevádzke:

zoneadm list -vc

<i>ID</i>	<i>NAME</i>	<i>STATUS</i>	<i>PATH</i>	<i>BRAND</i>	<i>IP</i>
0	global	running	/	native	shared
1	kirp142	running	/export/home/zones/kirp142	native	shared
5	kirp143	running	/export/home/zones/kirp143	native	shared.

Základné informácie o vytvorenej zóne Kirp143 získame pomocou:

uname -a

SunOS kirp141 5.10 Generic_139556-08 i86pc i386 i86pc.

Z výpisu je zreteľné, že vytvorená zóna má meno počítača Kirp141, preto po prihlásení sa do zóny Kirp143 je pre bezchybné fungovanie:

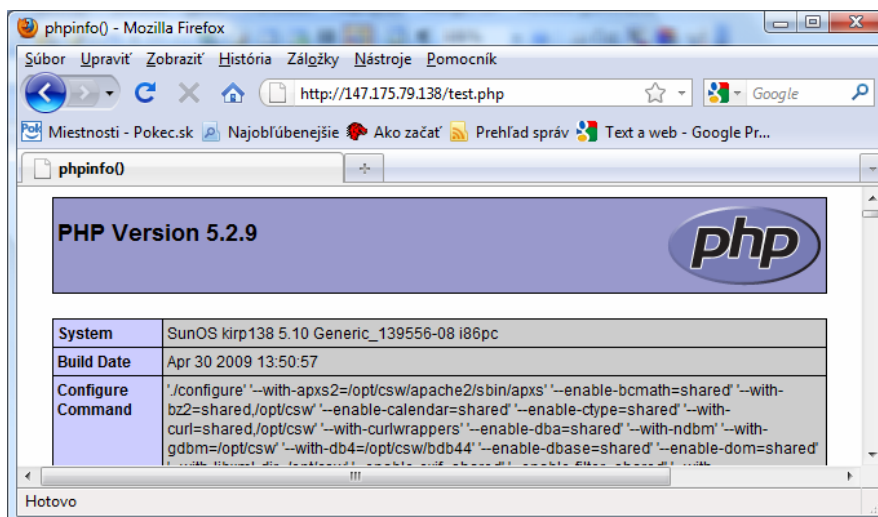
- v /etc/nodename upraviť meno počítača z Kirp141 na Kirp143,
- v /etc/hosts upraviť IP adresu a meno počítača,
- v /etc/inet/ipnodes skontrolovať, či sa zmenou údajov v /etc/hosts automaticky prepísali aj údaje v tomto súbore.

Zmena údajov sa prejaví aj v základných informáciách získaných pomocou:

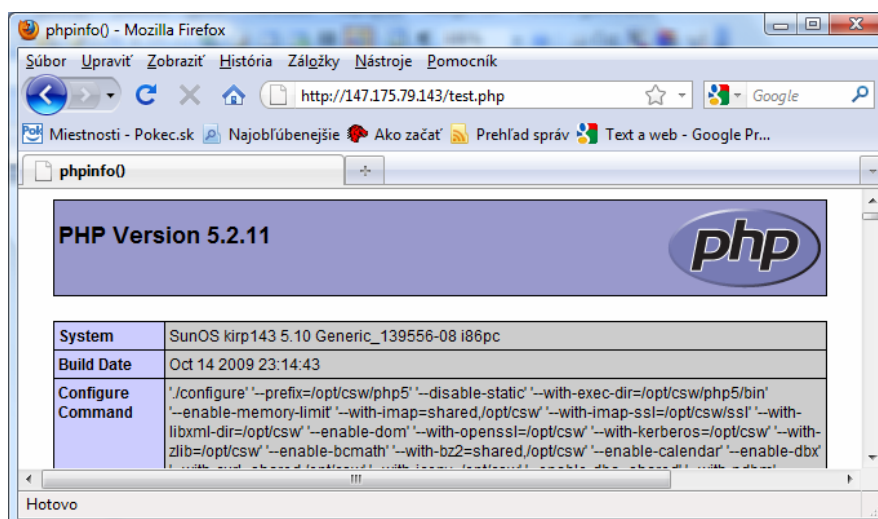
uname -a

SunOS kirp143 5.10 Generic_139556-08 i86pc i386 i86pc.

Vytvorená lokálna zóna Kirp143 je úplne nezávislá, čo nám dokazuje aj skutočnosť, že v globálnej zóne Kirp138 je nainštalovaná verzia PHP 5.2.9 a v lokálnej zóne Kirp143 verzia PHP 5.2.11.



Obr. 22 Verzia PHP v globálnej zóne Kirp138



Obr. 23 Verzia PHP v lokálnej zóne Kirp143



Obr. 24 Konfigurácia Kirp138, Kirp142, Kirp143

Našou úlohou je mať v prevádzke aj lokálnu zónu Kirp141, preto je nevyhnutné zónu Kirp141 znovu pripojiť:

zoneadm -z kirp141 attach -u.

Nevyhnutné je lokálnu zónu Kirp141 znovu nabootovať:

zoneadm -z kirp141 boot.

2.2 Zát'azové testy MySQL a Apache servera

Pomocou zát'azových testov sme sledovali správanie MySQL a Apache serverov v kritických situáciách. Na testovanie MySQL serverov vo všetkých zónach sme použili MySQL5test a MySQL Benchmark, ktoré sú súčasťou balíčkovacieho systému určeného pre aktualizáciu systému. Základnou úlohou pri testovaní bolo sledovanie rýchlosti pripojenia sa k serverom, rýchlosti vytvárania tabuliek, respektíve databáz, rýchlosti zapisovania dát do vytvorených tabuliek, a rýchlosti výberu dát z tabuliek. Výkon a správnosť fungovania Apache serverov vo všetkých zónach sme overili pomocou nástroja Apache JMeter a nástroja pomenovaného ako ab - Apache HTTP server benchmarking tool, ktorý je obsiahnutý v zdrojovom kóde distribúcie Apache servera. Našou úlohou bolo počas testov Apache serverov sledovať odozvu, priepustnosť a chybovosť serverov pri veľkom počte požiadaviek na server.

2.2.1 MySQL5test

Je testovací nástroj, ktorý je obsiahnutý v zdrojovom kóde distribúcie MySQL. Umožňuje užívateľom a vývojárom vykonávať testovanie MySQL servera. V prípade potreby je možné, aby si jednotliví vývojári vytvárali svoje vlastné testy, prostredníctvom ktorých môžu testovať MySQL server na požiadavky, ktoré považujú za dôležité. Základnou úlohou tohto testovacieho nástroja je otestovať správanie systému v kritických situáciách, čiže pri veľkom počte požiadaviek na server. Tento testovací nástroj pozostáva z interpretra mysql-test a per skriptu mysql-test.pl

slúžiaceho na spúšťanie všetkých testov. Domovská stránka tohto testovacieho nástroja je <http://dev.mysql.com/doc/mysql/en/mysql-test-suite.html>.

Inštaláciu tohto nástroja a všetkých potrebných balíčkov vo všetkých zónach naraz môžeme uskutočniť pomocou pkgutil v globálnej zóne:

pkgutil -i -y mysql5test,

pričom sa tento testovací nástroj spúšťa z adresára /opt/csw/mysql5/mysql-test v každej zóne samostatne.

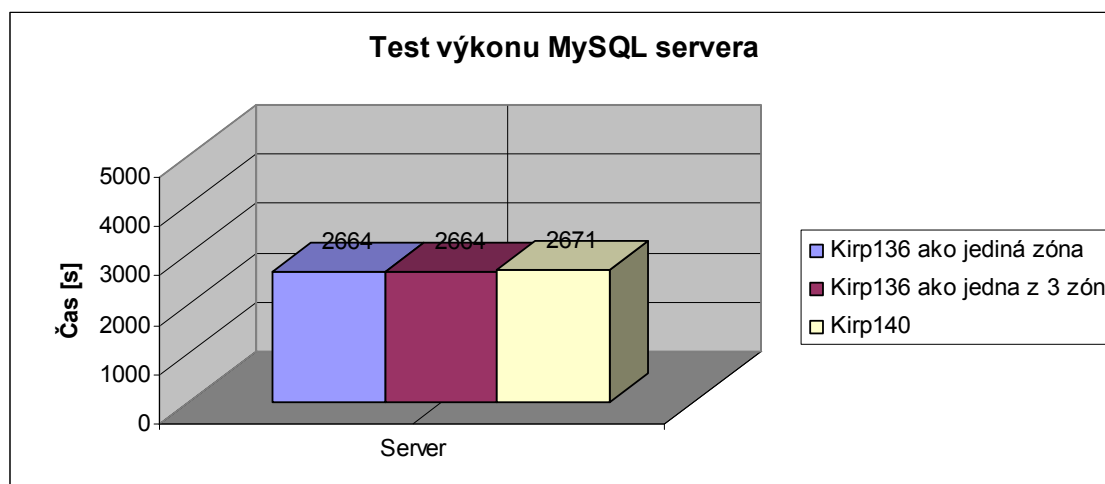
Uvádzame iba výsledky z jednej lokálnej zóny, pretože výsledky oboch lokálnych zón boli takmer totožné.

2.2.1.1 Testovanie pomocou MySQL5test

Najprv sme pristúpili k testovaniu MySQL servera na Kirp136, ako jedinej zóny v systéme, následne sa testovalo správanie MySQL servera na Kirp136 ako jednej z troch zón. Ako posledné sa testovalo správanie MySQL servera na Kirp140, čiže na lokálnej zóne. Spustenie všetkých testov v jednotlivých zónach samostatne sa uskutoční pomocou:

./mysql-test-run --force,

pričom po zadaní tohto príkazu sa automaticky spustí 402 testov.

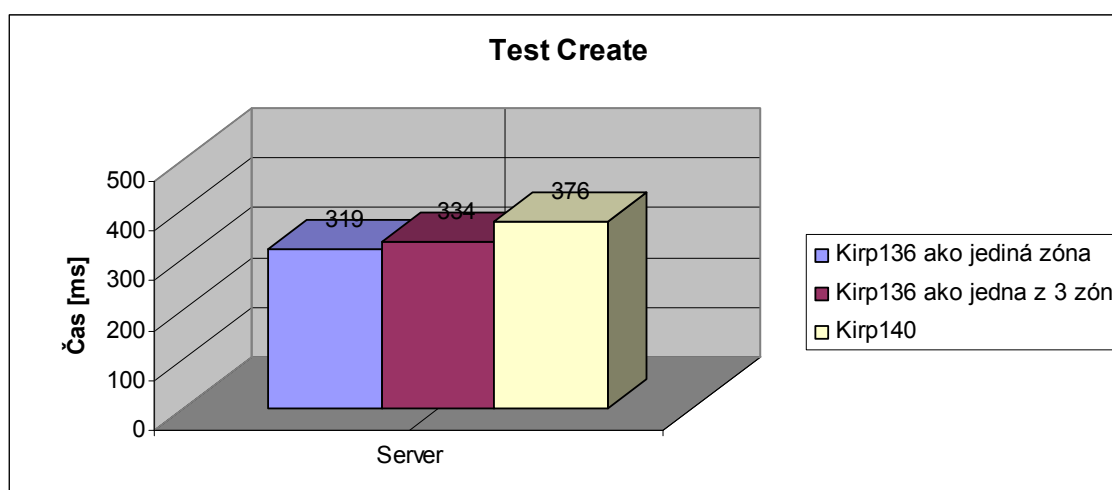


Obr. 25 Test výkonu MySQL servera pomocou MySQL5test

Z výsledkov tohto testu je zreteľné, že v prípade, ak budeme mať súbežne spustenú globálnu zónu a dve lokálne zóny nedôjde takmer k žiadnemu zhoršeniu výkonnosti MySQL servera. Výkonnosť MySQL servera lokálnej zóny je porovnateľná s výkonnosťou MySQL servera globálnej zóny.

2.2.1.2 Test Create

Tento test je zameraný na neustále vytváranie testovacích tabuliek, pričom počet vytvorených tabuliek závisí od úspešnosti vytvárania tabuliek, zapisovania údajov do vytvorených tabuliek a následnom vyberaní zapísaných údajov z týchto tabuliek. Testovacie tabuľky sa neustále vytvárajú a vymazávajú, aby sa šetrilo miesto na disku.



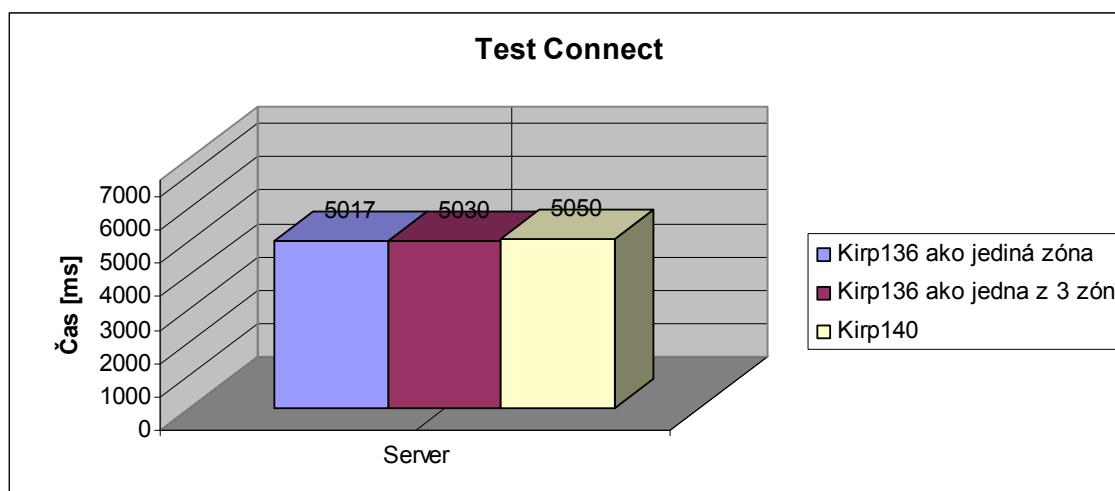
Obr. 26 Test Create

Z výsledkov testu vyplýva, že MySQL server na Kirp136 ako samostatnej zóny a server na Kirp136 v prípade, ak súbežne bežia dve lokálne zóny, sú výkonnostne podobné. Server lokálnej zóny Kirp140 je o niečo menej výkonný.

2.2.1.3 Test Connect

Hlavnou náplňou tohto testu je overiť rôzne prípady spojenia s MySQL serverom. Zisťuje sa tu rýchlosť odozvy MySQL servera pri zadaní správneho hesla a rýchlosť

odzvy pri zadaní nesprávneho hesla. Testuje sa tu rýchlosť pri priamom pripojení k databáze a rýchlosť bez priameho pripojenia k databáze.



Obr. 27 Test Connect

Z výsledkov testu Connect je zreteľné, že výkonnosť MySQL servera globálnej zóny Kirp136 je porovnateľná s výkonnosťou MySQL servera lokálnej zóny Kirp140. Výkonnosť MySQL servera globálnej zóny Kirp136 sa v prípade súbežného behu dvoch lokálnych zón takmer nemení.

2.2.2 The MySQL Benchmark

MySQL Benchmark je testovací nástroj, ktorý slúži na testovanie MySQL servera. Podstata tohto testovacieho nástroja spočíva v tom, že postupne zaznamenáva implementáciu SQL príkazov, sleduje či implementácia príkazov prebieha správne alebo nesprávne. Výhodou tohto nástroja je, že umožňuje automatické spustenie všetkých testov naraz, poprípade v prípade potreby môžeme pristúpiť k použitiu iba jednotlivých testov samostatne.[8]

Tento testovací nástroj pre MySQL server sa po nainštalovaní nachádza v zložke /opt/csw/mysql5/sql-bench, kde v súbore README nájdeme informácie o tomto testovacom nástroji.

Inštalácia vo všetkých zónach naraz v globálnej zóne pomocou pkgutil:

pkgutil -i -y mysql5bench.

2.2.2.1 Testovanie pomocou MySQL Benchmark


Pri testovaní MySQL serverov v jednotlivých zónach boli použité všetky testy naraz pomocou príkazu:

./run-all-tests --server=mysql --cmp=mysql,pg,solid --user=root --password=* --force --log,**

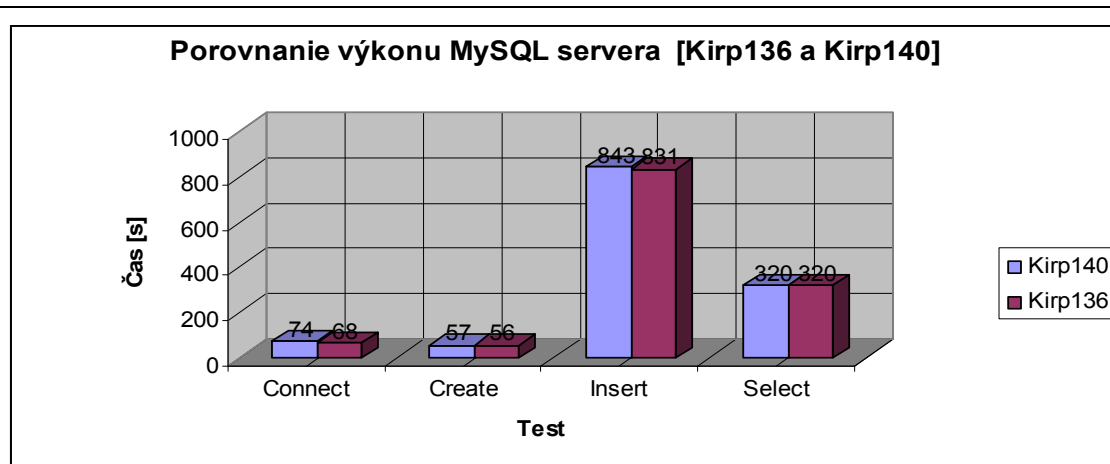
pričom po zadaní tohto príkazu získame výsledky testu v podobe šesť stĺpcovej tabuľky. Prvý stĺpec predstavuje názov testu, druhý stĺpec predstavuje dĺžku testu v sekundách. Tretí stĺpec predstavuje percentuálne vytťaženie CPU, užívateľom MySQL servera, pomocou ktorého sa prihlasujeme na server pri vykonávaní testu. Štvrtý stĺpec predstavuje percentuálne vytťaženie CPU systémom pri vykonávaní testu, piaty stĺpec je súčtom tretieho a štvrtého stĺpca. Posledný stĺpec predstavuje počet dát, ktoré sa preniesli pri testovaní.

Uvádzame iba výsledky z jednej lokálnej zóny, pretože výsledky oboch lokálnych zón boli takmer totožné. Najprv sme otestovali MySQL server, bežiaci v lokálnej zóne Kirp140, potom sme testovali MySQL server v globálnej zóne, pričom obe lokálne zóny boli spustené. Následne sme test v globálnej zóne opakovali, pričom obe lokálne zóny boli pozastavené.

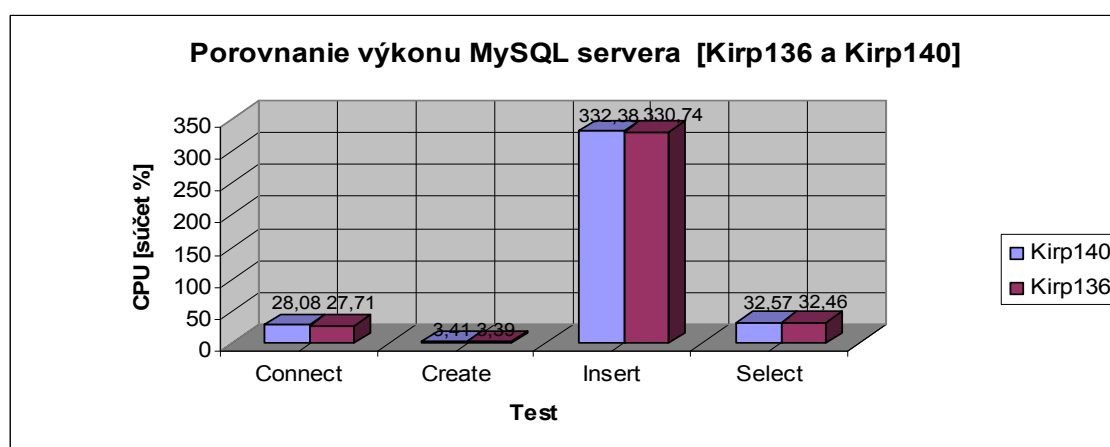
Tab. 1 Výsledky získané pomocou MySQL Benchmark [Kirp136/Kirp140]

	seconds	usr	sys	cpu	tests
Kirp140	1419,00	369,97	35,83	405,80	3161841
Kirp136	1408,00	369,62	33,91	403,53	3161841

Z výsledkov testu je zreteľné, že MySQL server globálnej zóny, pri behu dvoch lokálnych zón, má porovnateľnú výkonnosť ako MySQL server lokálnej zóny, pričom oba MySQL servery vytťažujú CPU takmer rovnako.



Obr. 28 Porovnanie podľa jednotlivých testov [Kirp136/Kirp140] – dĺžka testu

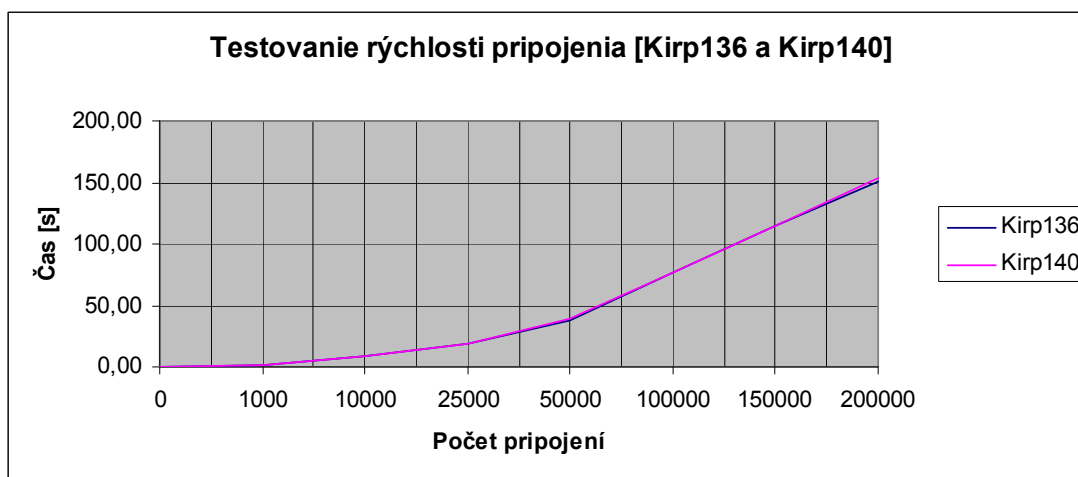


Obr. 29 Porovnanie podľa testov [Kirp136/Kirp140] – vytáženie CPU

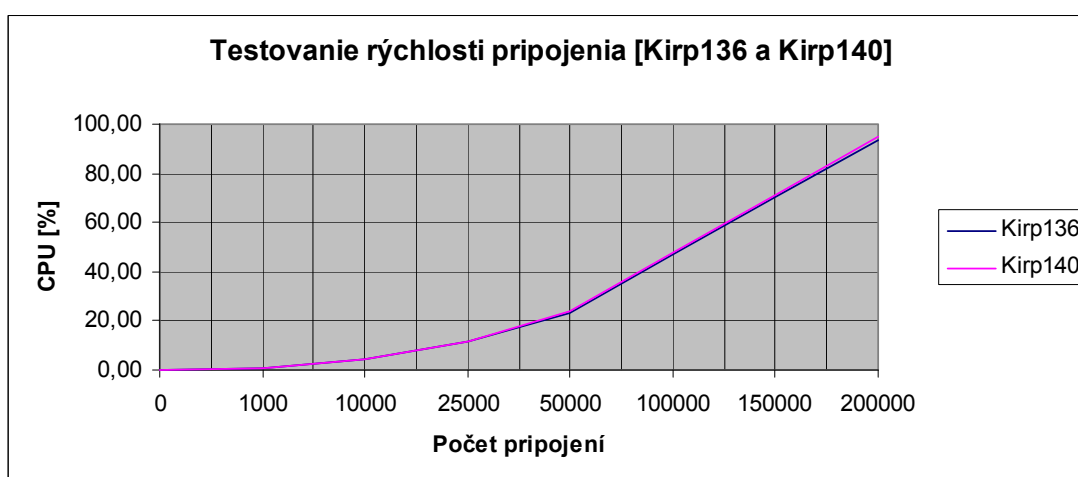
Úlohou testu rýchlosti pripojenia, respektíve testu rýchlosti vkladania údajov, je sledovanie rýchlosti pripojenia, respektíve rýchlosti vkladania údajov, a sledovanie vytáženía CPU pri testovaní pri postupnom zvyšovaní počtu požiadaviek na server.

Tab. 2 Test rýchlosti pripojenia [Kirp136/Kirp140]

Testovanie rýchlosti pripojenia [Kirp136/Kirp140]					
Počet pripojení	Server	Čas [s]		Využitie CPU [%]	
		Kirp136	Kirp140	Kirp136	Kirp140
1 000		1	1	0,47	0,47
10 000		8	8	4,67	4,67
25 000		19	19	11,69	11,85
50 000		38	39	23,39	23,87
100 000		77	77	46,84	47,60
150 000		114	114	70,08	71,07
200 000		151	154	93,26	94,84



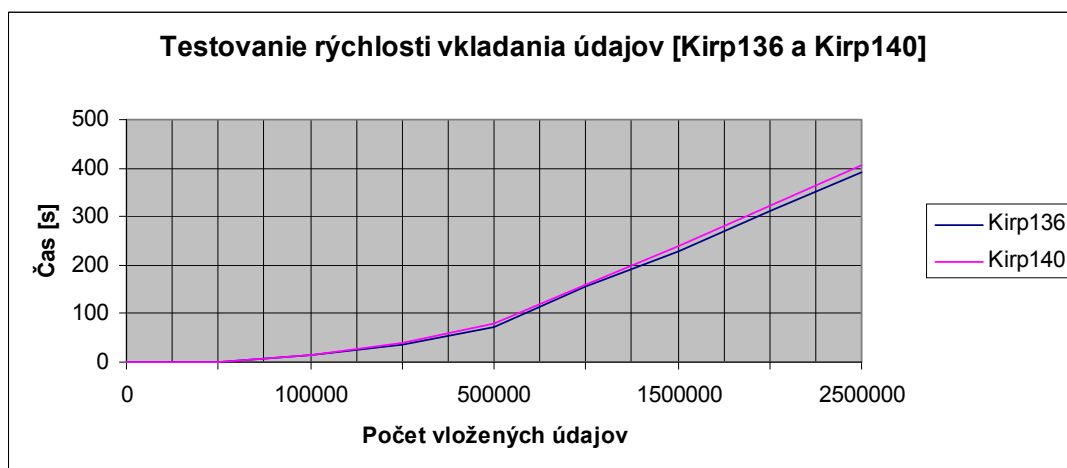
Obr. 30 Test rýchlosti pripojenia [Kirp136/Kirp140] – rýchlosť pripojenia



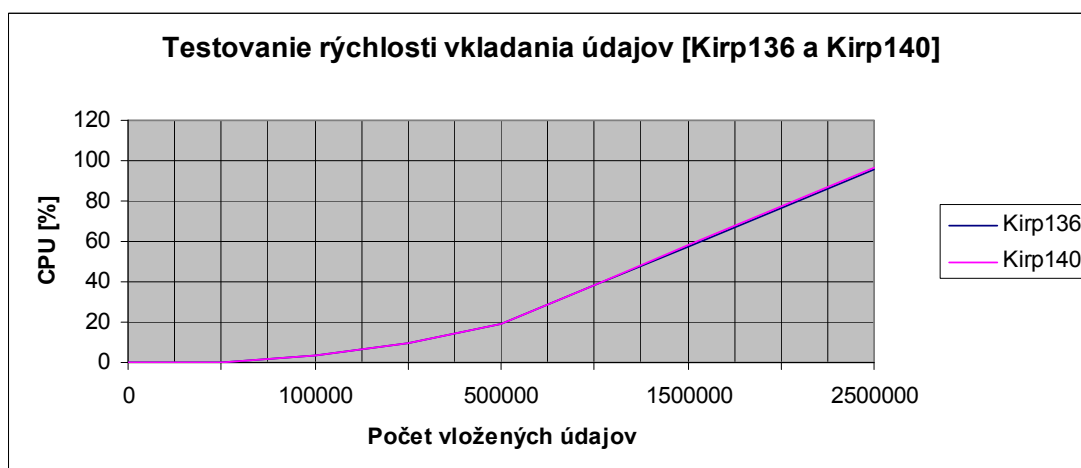
Obr. 31 Test rýchlosti pripojenia [Kirp136/Kirp140] – vyťaženie CPU

Tab. 3 Test rýchlosti vkladania údajov [Kirp136/Kirp140]

Testovanie rýchlosti vkladania údajov [Kirp136/Kirp140]					
Počet údajov	Server	Čas [s]		Využitie CPU [%]	
		Kirp136	Kirp140	Kirp136	Kirp140
10 000		1	1	0,38	0,38
100 000		15	15	3,78	3,82
250 000		36	39	9,50	9,60
500 000		74	79	18,99	19,22
1 000 000		155	160	18,27	38,63
1 500 000		230	240	57,31	57,92
2 000 000		310	322	76,44	77,34
2 500 000		390	404	95,25	96,30



Obr. 32 Test rýchlosti vkladania údajov [Kirp136/Kirp140] – rýchlosť vkladania



Obr. 33 Test rýchlosti vkladania údajov [Kirp136/Kirp140] – vyťaženie CPU

Z testov je zreteľné, že súhrnný čas pripojení, respektíve súhrnný čas zapisovania údajov, oboch MySQL serverov je podobný a vyťaženie CPU oboch MySQL serverov je takmer rovnaké.

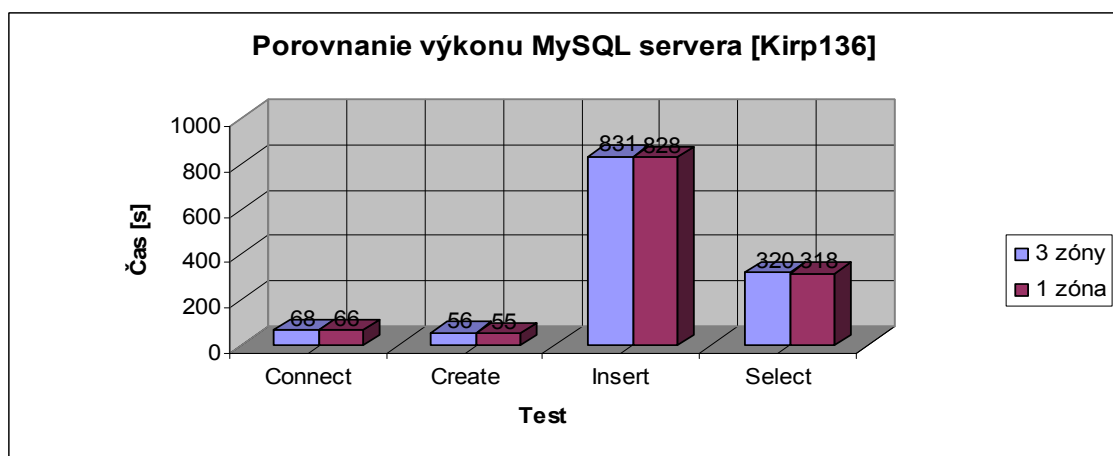
2.2.2.2 Test MySQL servera globálnej zóny pri pozastavení lokálnych zón

Tento test bol zameraný na sledovanie výkonu MySQL servera globálnej zóny a vyťaženia CPU pri pozastavení všetkých lokálnych zón.

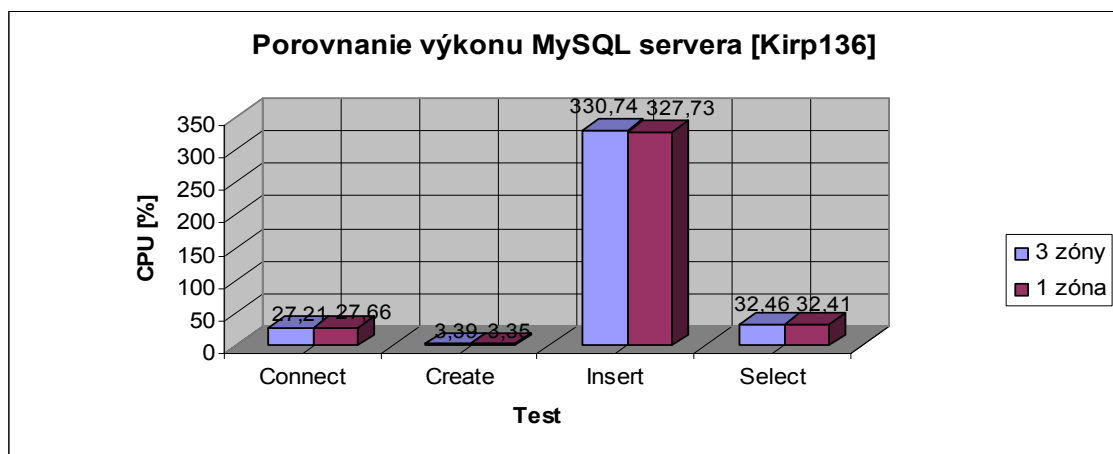
Tab. 4 Test MySQL servera globálnej zóny pri pozastavení lokálnych zón

	seconds	usr	sys	cpu	tests
Kirp136 (3 Z)	1408,00	369,62	33,91	403,53	3161841
Kirp136 (1 Z)	1378,00	368,43	33,78	402,21	3161841

Z testu je zreteľné, že MySQL server globálnej zóny, v prípade ak sú pozastavené lokálne zóny a v prípade ak sú obe lokálne zóny v prevádzke, podáva takmer rovnaké výsledky a vyťažuje CPU takmer na rovnakej úrovni.



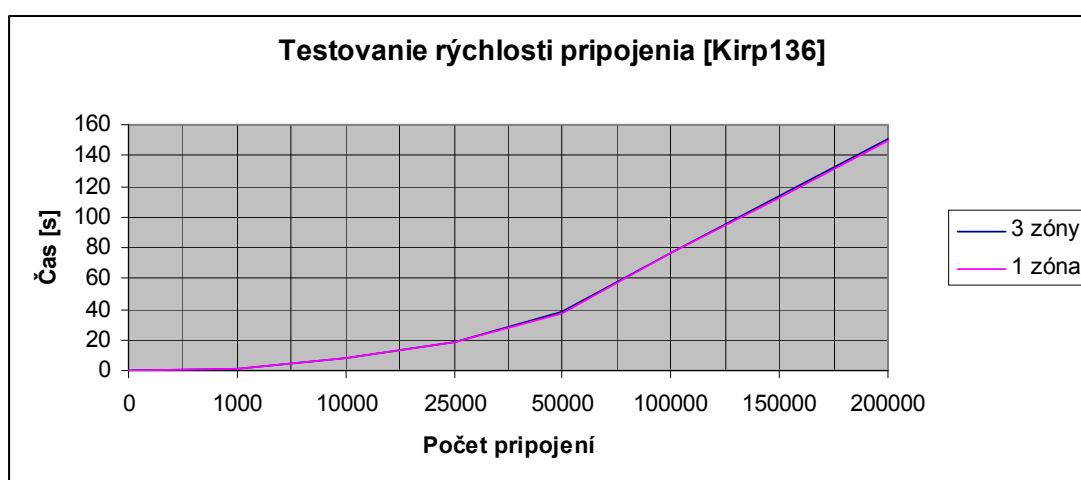
Obr. 34 Porovnanie výkonu podľa jednotlivých testov [Kirp136] – dĺžka testu



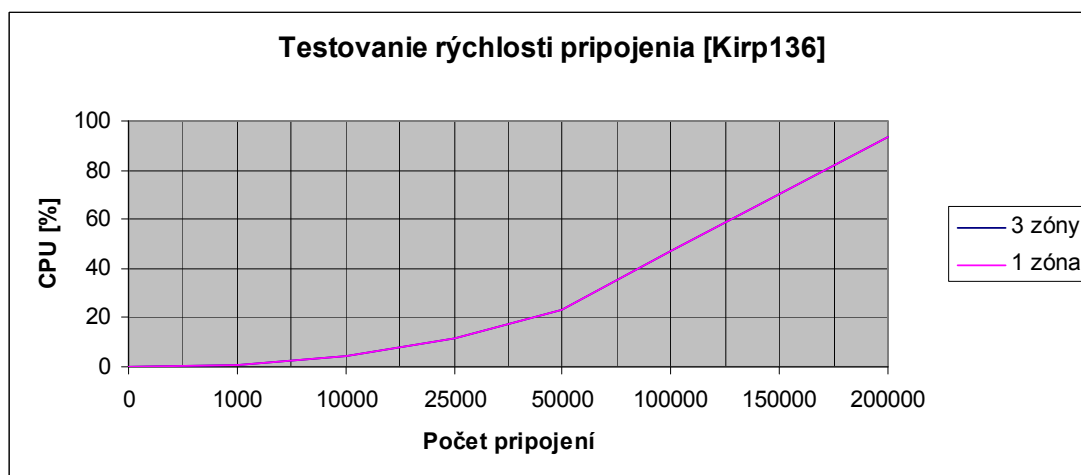
Obr. 35 Porovnanie výkonu podľa testov [Kirp136] – vyťaženie CPU

Tab. 5 Test rýchlosti pripojenia [Kirp136]

Testovanie rýchlosti pripojenia [Kirp136]					
Počet pripojení	Server	čas [s]		využitie CPU [%]	
		3 zóny	1 zóna	3 zóny	1 zóna
1 000		1	1	0,47	0,47
10 000		8	8	4,67	4,67
25 000		19	19	11,69	11,67
50 000		38	37	23,39	23,37
100 000		77	76	46,84	46,80
150 000		114	112	70,08	70,00
200 000		151	150	93,26	93,20



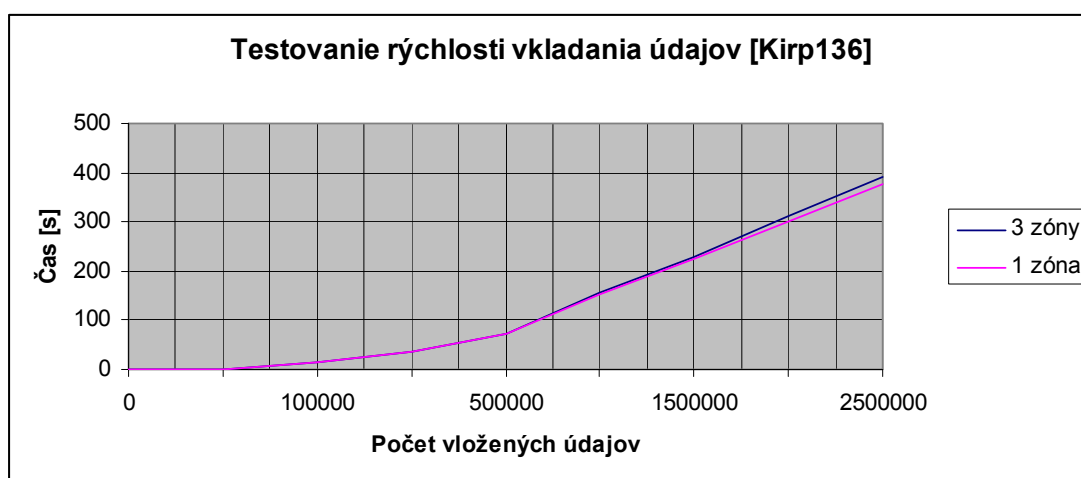
Obr. 36 Test rýchlosti pripojenia [Kirp136] – rýchlosť pripojenia



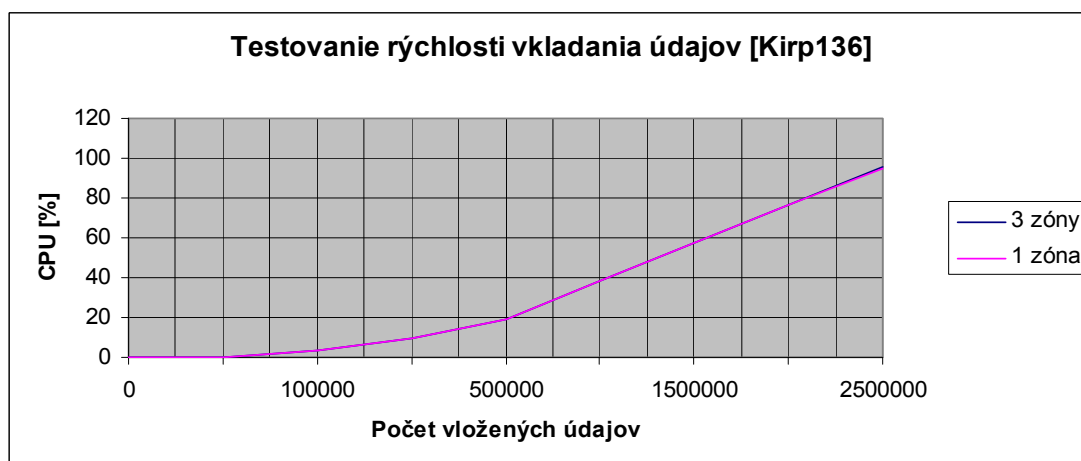
Obr. 37 Test rýchlosti pripojenia [Kirp136] – vyťaženie CPU

Tab. 6 Porovnanie rýchlosti vkladania údajov [Kirp136]

Testovanie rýchlosti vkladania údajov [Kirp136]					
Počet údajov	Server	čas [s]		využitie CPU [%]	
		3 zóny	1 zóna	3 zóny	1 zóna
10 000		1	1	0,38	0,38
100 000		15	15	3,78	3,78
250 000		36	35	9,50	9,50
500 000		74	72	18,99	18,90
1 000 000		155	152	38,27	38,20
1 500 000		230	225	57,31	57,20
2 000 000		310	302	76,44	76,11
2 500 000		390	377	95,25	94,80



Obr. 38 Test rýchlosti vkladania údajov [Kirp136] – rýchlosť vkladania



Obr. 39 Test rýchlosti vkladania údajov [Kirp136] – vyťaženie CPU

Testy rýchlosti pripojenia a rýchlosti vkladania údajov potvrdili, že výkon oboch MySQL serverov je takmer rovnaký a oba servery vyťažujú CPU rovnako.

2.2.3 Apache JMeter

Je open source aplikácia naprogramovaná v Jave určená pre záťažové testy, skúmanie funkčnosti správania Apache serverov v kritických podmienkach a meranie výkonu. Tento testovací nástroj bol pôvodne naprogramovaný pre testovanie iba webových aplikácií, ale s postupom času sa rozšíril o ďalšie testovacie funkcie. Základnou výhodou tohto nástroja je, že je 100 % Java aplikáciou, čo zabezpečuje to, že by mal správne fungovať na ľubovoľnom počítači, ktorý je kompatibilný s Javou bez ohľadu či ide o UNIXový operačný systém alebo o Windows. Ďalšou výhodou je, že JMeter môžeme spúšťať z príkazového riadku alebo v grafickom móde.[9]

2.2.3.1 Tvorba testovacieho plánu

Prvým krokom testovania je tvorba testovacieho plánu, kde je nevyhnutné nastaviť parametre Thread Group, kde je nevyhnutné nastaviť počet vlákien, ramp-up čas a počet opakovaní testu. Veľkú pozornosť tu treba venovať ramp-up času, ktorý má predísť nadmernému pracovnému zaťaženiu na začiatku testu. Napríklad, ak si zvolíme 60 užívateľov a ramp-up čas 240 sekúnd zabezpečíme, že po štarte testu bude aktívny iba jeden užívateľ a druhý užívateľ začne simulovať svoju činnosť až o 4 sekundy neskôr, tretí po 8 sekundách od štartu testu.

Na jednotlivé servery sme umiestnili obrázky veľké 243,51 kB, s cieľom zvýšiť zaťaženie servera pri jednotlivých požiadavkách, ktoré boli pomocou vzorkovača HTTP Request zamerané na sťahovanie daných obrázkov.

Regulátor

Existujú dva druhy regulátorov. Vzorkovače (Samplers) zasielajú požiadavky na server a čakajú na odpoveď zo servera. Ich podstata je v tom, že požiadavky sa zasielajú v takom poradí, v akom sú zaradené do stromovej štruktúry, čiže podľa vopred stanoveného poradia. Automaty (Logic Controllers) umožňujú logické zasielanie požiadaviek, a to tak, že môžu zmeniť poradie žiadostí, čiže sa poruší vopred stanovená stromová štruktúra, to znamená, že detské elementy môžu predbehnúť rodičovské.

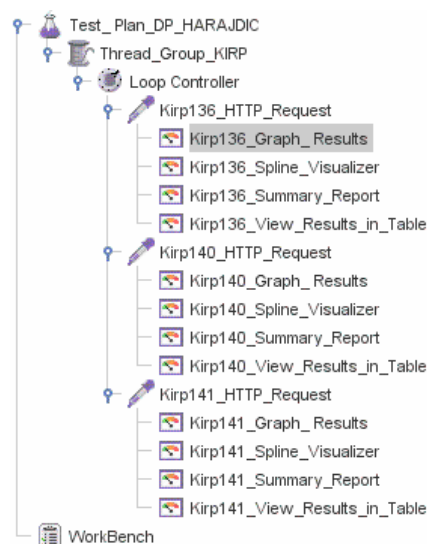
Naším cieľom je zasielanie veľkého množstva požiadaviek na tri Apache servery súčasne, preto sme použili automat Loop Controller a tri vzorkovače HTTP Request. Pomocou automatu Loop Controller sme zabezpečili to, aby do každého vzorkovača išlo 3000 požiadaviek. Vzorkovač HTTP Request umožňuje posielat' HTTP / HTTPS požiadavky na webový server, pričom je možné zároveň analyzovať jednotlivé HTML súbory.

Poslucháče (Listeners)

Ich základnou úlohou je poskytovať prístup k výsledkom získaným pomocou JMetra. Ďalšou možnosťou, ktorú poskytujú je možnosť zápisu výsledkov do súborov. Poslucháče sa umiestňujú pod plán skúšok, z ktorého budú zaznamenávať údaje. Pri testovaní sme využili poslucháč View Results in Table a poslucháč Summary Report. Poslucháč View Results in Table zabezpečí spracovanie nameraných údajov v podobe prehľadnej tabuľky. Nevýhodou takéhoto spracovania výsledkov je skutočnosť, že je potrebná veľká voľná pamäť. Poslucháč Summary Report je veľmi blízky predchádzajúcemu, ale má výhodu v tom, že nemá také veľké nároky na pamäť, pretože nezachytáva všetky hodnoty, ale iba priemerné hodnoty.

2.2.3.2 Test zameraný na malé zaťaženie Apache servera

Tento test bol zameraný na sledovanie správania Apache servera v prípade ak bude nastavených 3000 požiadaviek a ramp-up čas na 300. Toto nastavenie bolo zvolené tak, aby všetky tri sledované Apache servery stíhali reagovať na všetky požiadavky. Naším cieľom bolo sledovať správanie sa serverov, ak budú zasielané požiadavky naraz na všetky tri servery, preto bol zostavený takýto plán testu:

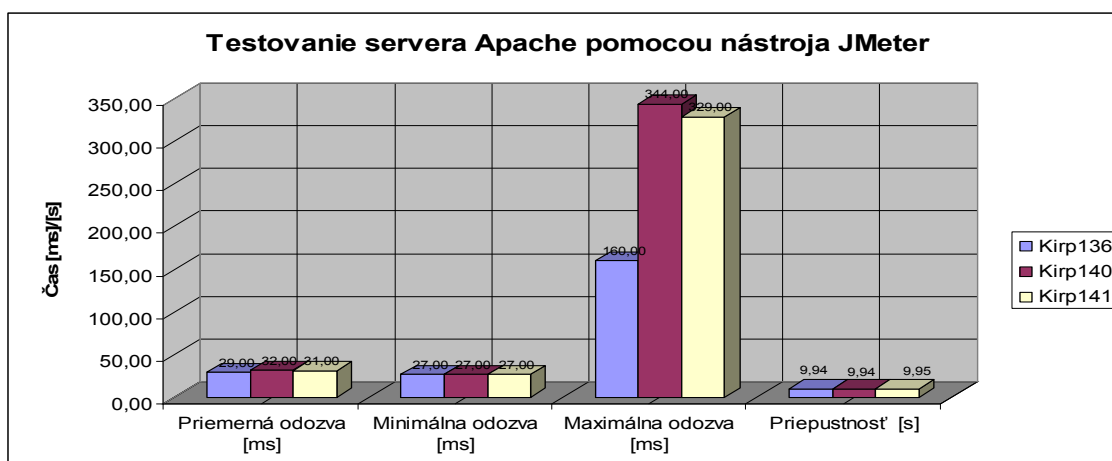


Obr. 40 Zostavený plán testu

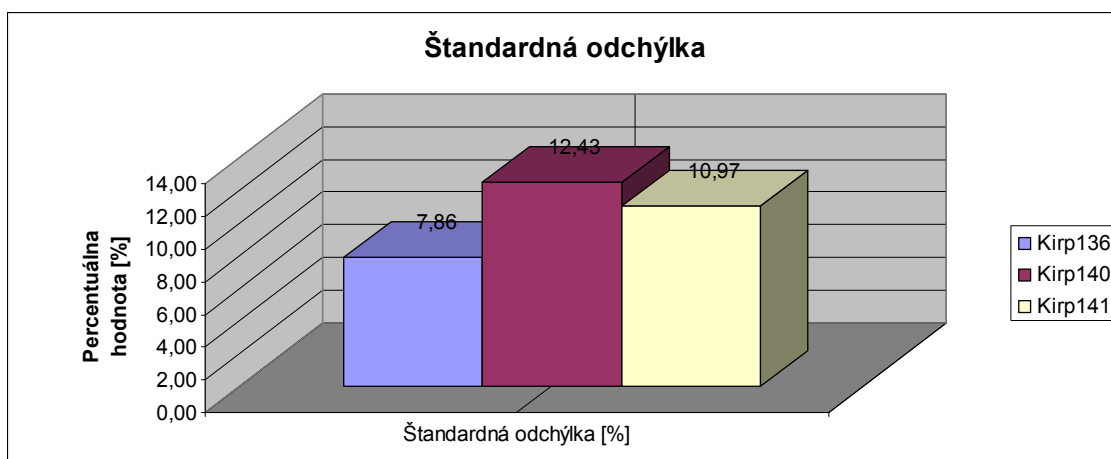
Tab. 7 Apache JMeter test – malé zaťaženie

Server	Počet požiadaviek	Priemerná odozva [ms]	Minimálna odozva [ms]	Maximálna odozva [ms]	Štandardná odchýlka [%]	Chybovosť [%]	Priepustnosť [s]	Priepustnosť [kb/s]	Priemerná reakcia [byt]
Kirp136	3000	29	27	160	7,86	0,00	9,94	2421,54	249359,00
Kirp140	3000	32	27	344	12,43	0,00	9,94	2421,34	249359,00
Kirp141	3000	31	27	329	10,97	0,00	9,95	2421,82	249359,00

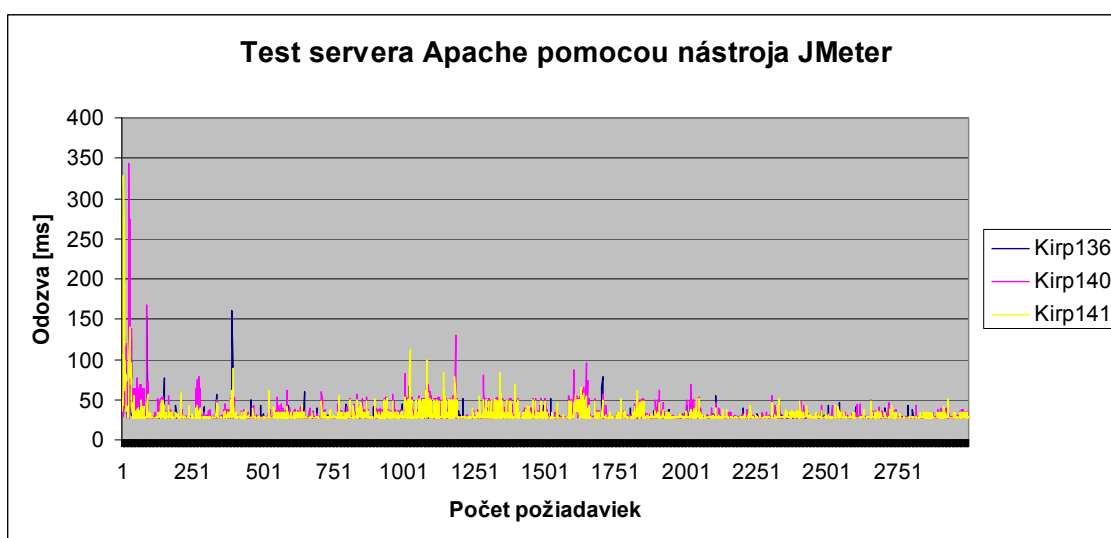
Z výsledkov tohto testu vyplýva skutočnosť, že Apache servery jednotlivých zón majú takmer rovnakú priemernú odozvu. Priepustnosť všetkých serverov je takmer rovnaká.



Obr. 41 Apache JMeter – malé zaťaženie



Obr. 42 Štandardná odchýlka – malé zaťaženie



Obr. 43 Odozva servera – malé zaťaženie

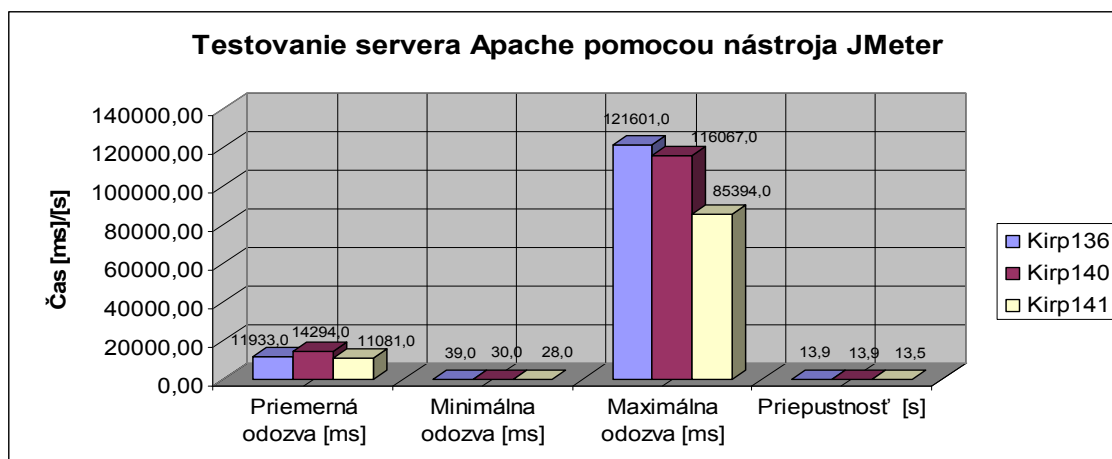
2.2.3.3 Test zameraný na vyššie zaťaženie Apache servera

Tento test sa zameriaval na sledovanie Apache serverov v prípade, ak bude nastavených 2000 požiadaviek a ramp-up čas na 100. Toto vyššie vyťaženie servera malo za dôsledok skutočnosť, že niektoré požiadavky na server boli neúspešné.

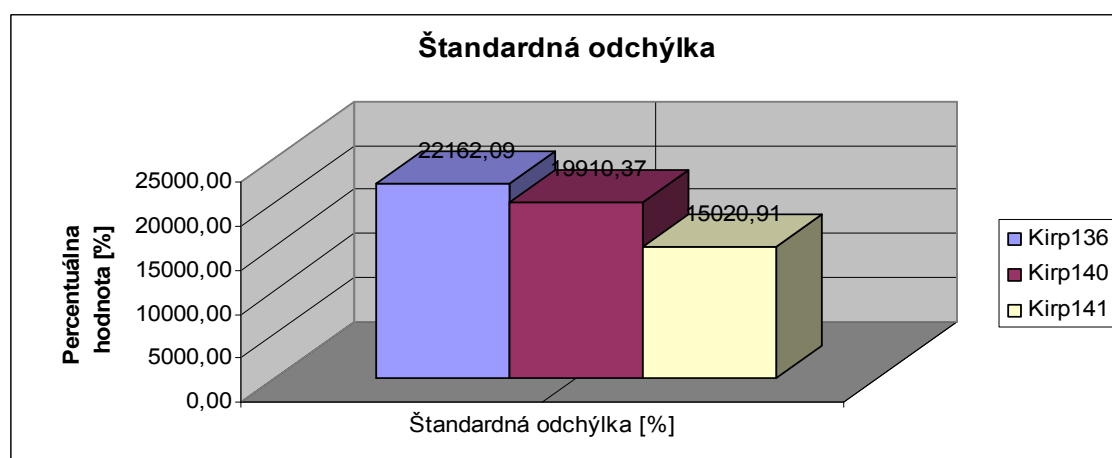
Tab. 8 Apache JMeter test – vyššie zaťaženie

Server	Počet požiadaviek	Priemerná odozva [ms]	Minimálna odozva [ms]	Maximálna odozva [ms]	Štandardná odchýlka [%]	Chybovosť [%]	Priepustnosť [s]	Priepustnosť [kb/s]	Priemerná reakcia [byt]
Kirp136	2000	11933	39	121601	22162,09	0,06	13,89	3181,18	234488,05
Kirp140	2000	14294	30	116067	19910,37	0,27	13,89	2461,66	181441,52
Kirp141	2000	11081	28	85394	15020,91	0,12	13,47	2873,89	218498,61

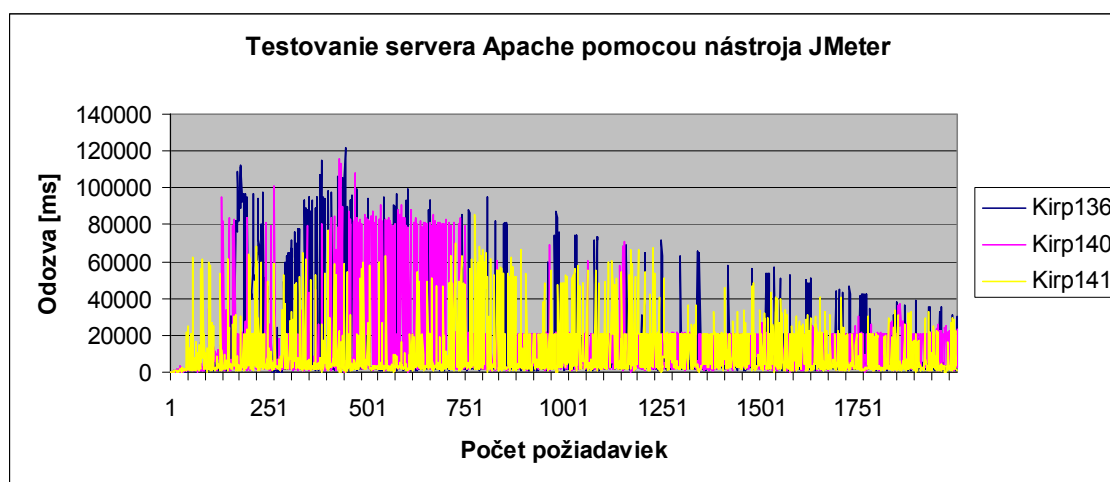
Z tohto testu je zreteľné, že pri vyššom zaťažení pracuje najlepšie Apache server globálnej zóny, vyplýva to z najnižšej priemernej odozvy servera, najnižšej chybovosti servera a z najvyššej priepustnosti servera.



Obr. 44 Apache JMeter – vyššie zaťaženie



Obr. 45 Štandardná odchýlka – vyššie zaťaženie



Obr. 46 Odozva servera – vyššie zaťaženie

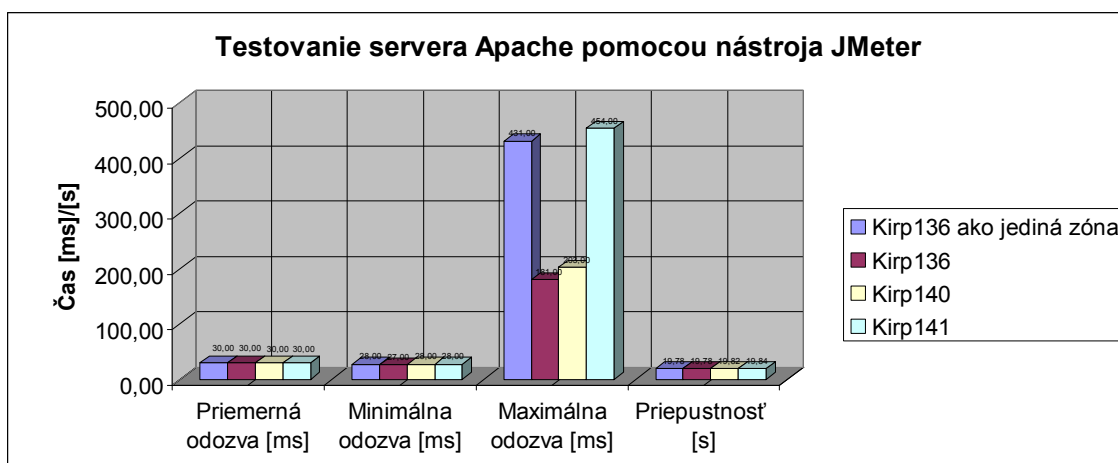
2.2.3.4 Test zameraný na zaťaženie Apache serverov jednotlivo

Tento test bol zameraný na to, že sme nezaťažovali všetky servery naraz, ale postupne Kirp136, Kirp140, Kirp141 a na záver boli pozastavené všetky lokálne zóny, aby sme otestovali aj výkon servera Apache v prípade, ak Kirp136 bude jediná zóna systému. Tento test sa zameriaval na sledovanie Apache serverov v prípade, ak bude nastavených 2000 požiadaviek a ramp-up čas na 100.

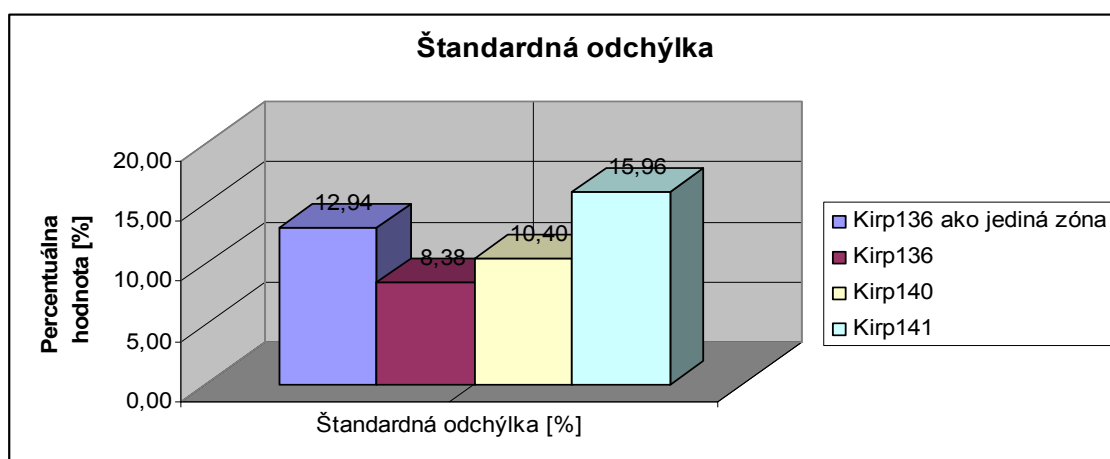
Tab. 9 Apache JMeter test – jednotlivé servery samostatne

Server	Počet požiadaviek	Priemerná odozva [ms]	Minimálna odozva [ms]	Maximálna odozva [ms]	Štandardná odchýlka [%]	Chybovosť [%]	Priepustnosť [s]	Priepustnosť [kb/s]	Priemerná reakcia [byť]
Kirp136 ako jediná zóna	2000	30	28	431	12,94	0,00	19,78	4816,92	249359,00
Kirp136	2000	30	27	181	8,38	0,00	19,78	4817,45	249359,00
Kirp140	2000	30	28	203	10,40	0,00	19,82	4825,89	249359,00
Kirp141	2000	30	28	454	15,96	0,00	19,84	4830,15	249359,00

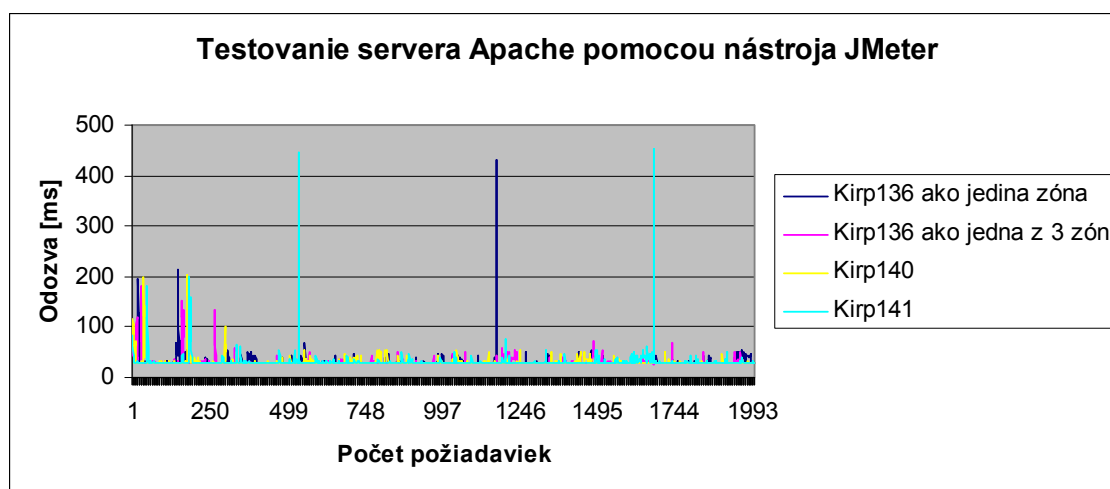
Z výsledkov testu vyplýva, že všetky Apache servery majú rovnakú výkonnosť.



Obr. 47 Apache JMeter – jednotlivé servery samostatne



Obr. 48 Štandardná odchýlka – jednotlivé servery samostatne



Obr. 49 Odozva servera – jednotlivé servery samostatne

2.2.4 Ab – Apache HTTP server benchmarking tool

Je testovací nástroj, ktorý je obsiahnutý v zdrojovom kóde distribúcie Apache servera. Jeho základnou úlohou je porovnávanie výkonu Apache servera a zaznamenávanie reakcií Apache servera pri obsluhovaní jednotlivých požiadaviek. Kvalita servera sa podľa tohto nástroja posudzuje podľa počtu obslužených požiadaviek za sekundu.

Medzi najpoužívanéjšie voliteľné možnosti pri testovaní patria:

- c - počet viacnásobných žiadostí vykonávaných naraz, predvolená hodnota je 1
- e - umožňuje uložiť výsledky v CSV súbore, ktorý bude obsahovať dve hodnoty oddelené čiarkou, prvá hodnota bude percento žiadostí a druhá hodnota bude čas odozvy v milisekundách
- g - umožňuje uložiť výsledky testu ako TSV súbor, v ktorom budú výsledky oddelené tabulátorom
- w - uloží výsledky v HTML tabuľkách
- n - počet žiadostí o vykonanie
- t - maximálny čas čakania na odpoveď.

2.2.4.1 Testovanie pomocou ab

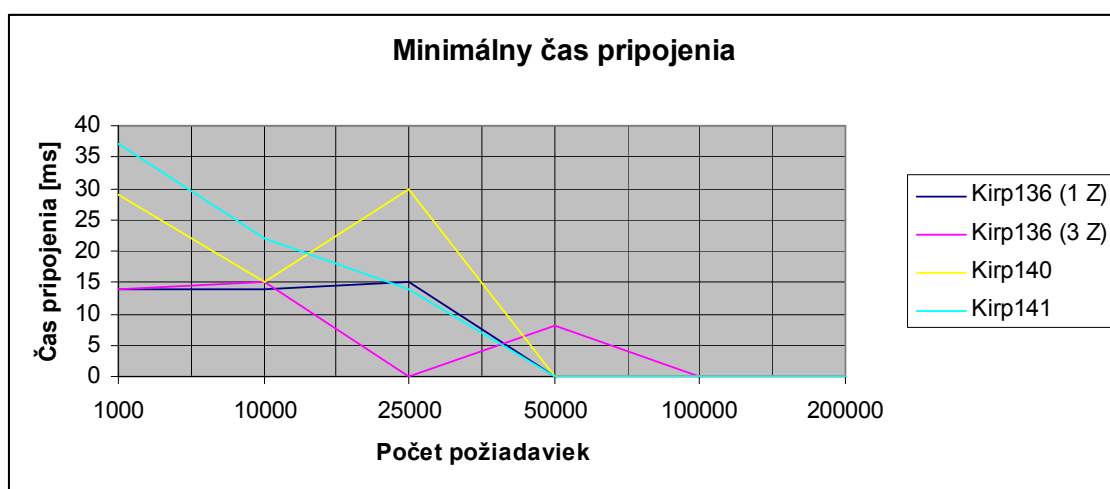
Cieľom tohto testu bolo sledovanie rýchlosti spracovávania rôzneho počtu žiadostí jednotlivými servermi za predpokladu, že na dané servery bude súčasne nasmerovaných 250 žiadostí. Test sme v jednotlivých zónach samostatne spustili pomocou:

./ab -n xxx -c 250 http://147.175.79.yyy/test.jpg,

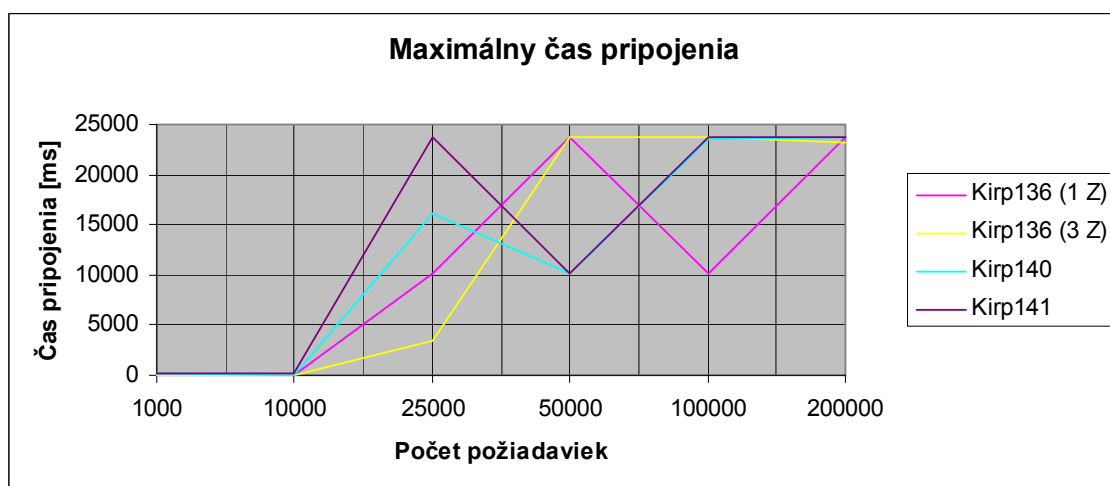
kde namiesto xxx sme zadávali požadovaný počet požiadaviek a namiesto yyy posledné tri čísla IP adresy konkrétnych serverov. Najprv sme testovali Apache server globálnej zóny, potom servery lokálnych zón a nakoniec znovu Apache server globálnej zóny pri pozastavení všetkých lokálnych zón.

Tab. 10 Minimálny/maximálny čas pripojenia nameraný pomocou (ab)

Počet požiadaviek	Minimálny čas pripojenia [ms]				Maximálny čas pripojenia [ms]			
	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141
1000	14	14	29	37	44	61	146	189
10000	14	15	15	22	52	45	52	266
25000	15	0	30	14	10202	3492	16143	23704
50000	0	8	0	0	23668	23704	10168	10172
100000	0	0	0	0	10163	23651	23635	23645
200000	0	0	0	0	23662	23233	23691	23654



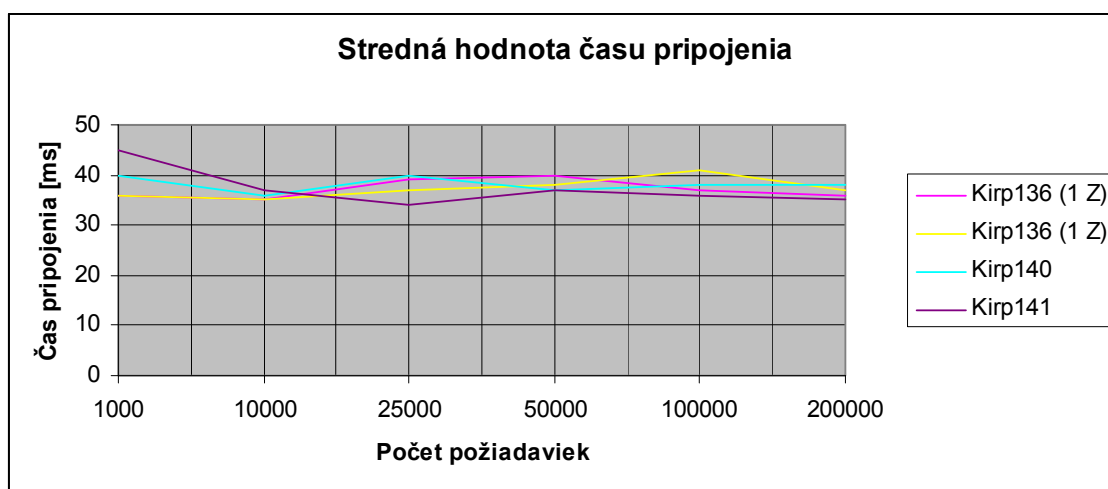
Obr. 50 Minimálny čas pripojenia nameraný pomocou ab



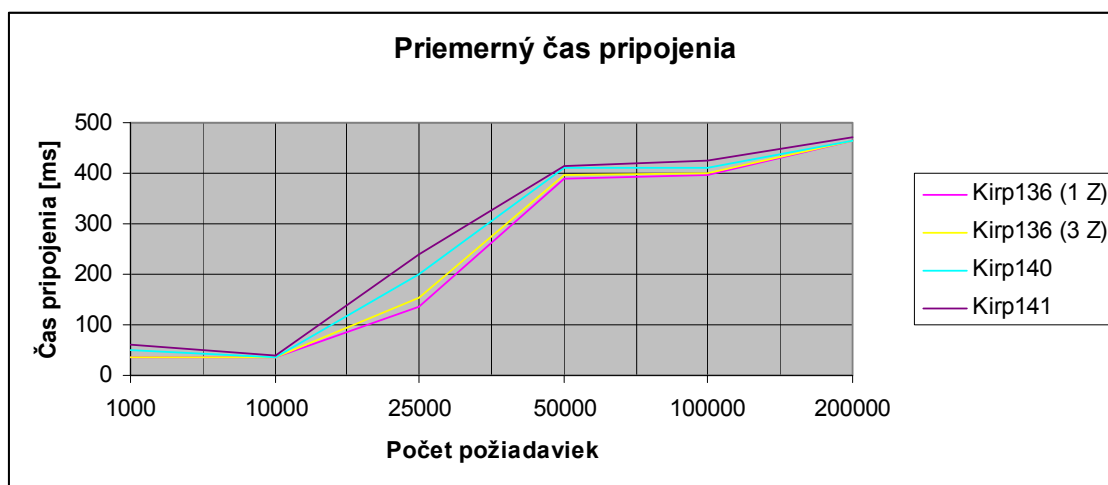
Obr. 51 Maximálny čas pripojenia nameraný pomocou ab

Tab. 11 Čas pripojenia nameraný pomocou (ab)

Počet požiadaviek	Stredná hodnota [ms]				Priemer [ms]			
	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141
1000	36	36	40	45	35	37	51	61
10000	35	35	36	37	35	35	36	40
25000	39	37	40	34	137	153	200	241
50000	40	38	37	37	391	396	409	414
100000	37	41	38	36	396	401	410	424
200000	36	37	38	35	463	466	466	470



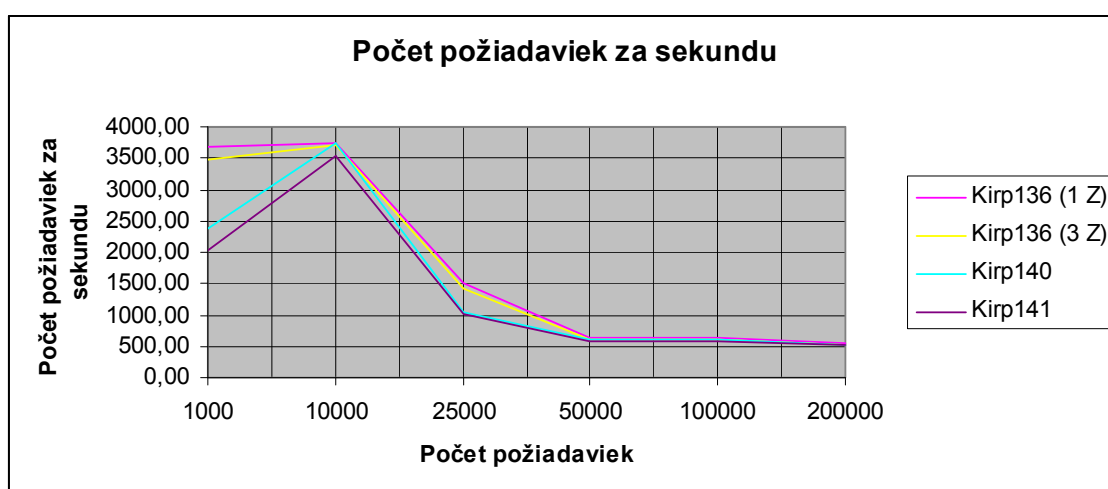
Obr. 52 Stredná hodnota času pripojenia nameraná pomocou ab



Obr. 53 Priemerný čas pripojenia nameraný pomocou ab

Tab. 12 Dĺžka testu/počet požiadaviek za sekundu (ab)

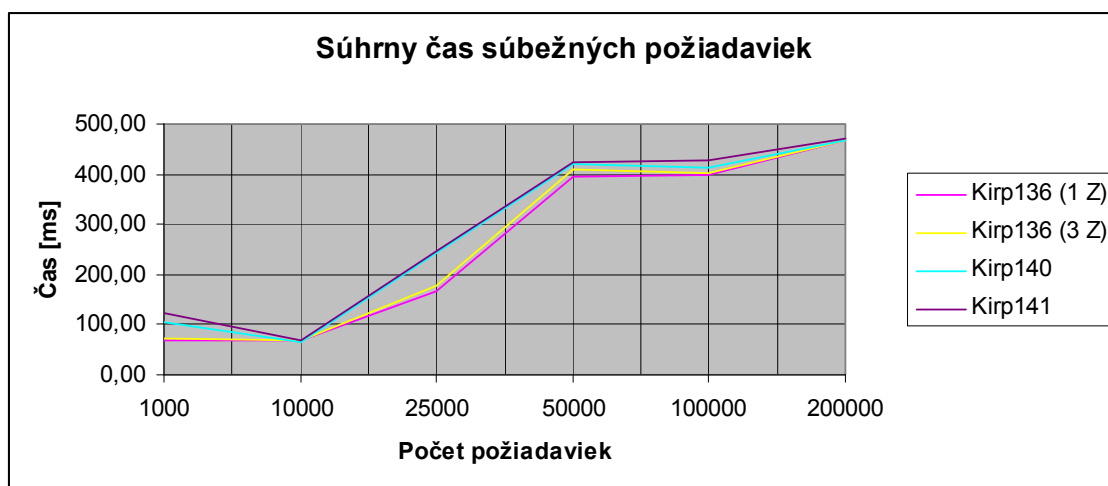
Počet požiadaviek	Dĺžka testu [s]				Počet požiadaviek za sekundu			
	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141
1000	0,27	0,29	0,42	0,49	3666,90	3472,10	2388,70	2027,50
10000	2,68	2,69	2,67	2,82	3728,20	3712,60	3744,20	3542,50
25000	16,64	17,77	24,24	24,80	1502,80	1406,90	1031,20	1007,90
50000	79,29	81,74	83,71	84,96	630,62	611,73	597,28	588,49
100000	159,45	161,59	165,71	171,38	627,18	618,85	603,46	583,49
200000	372,68	373,93	375,34	377,19	536,65	534,86	532,85	530,24



Obr. 54 Počet požiadaviek za sekundu získaný pomocou ab

Tab. 13 Súhrny čas súbežných požiadaviek (ab)

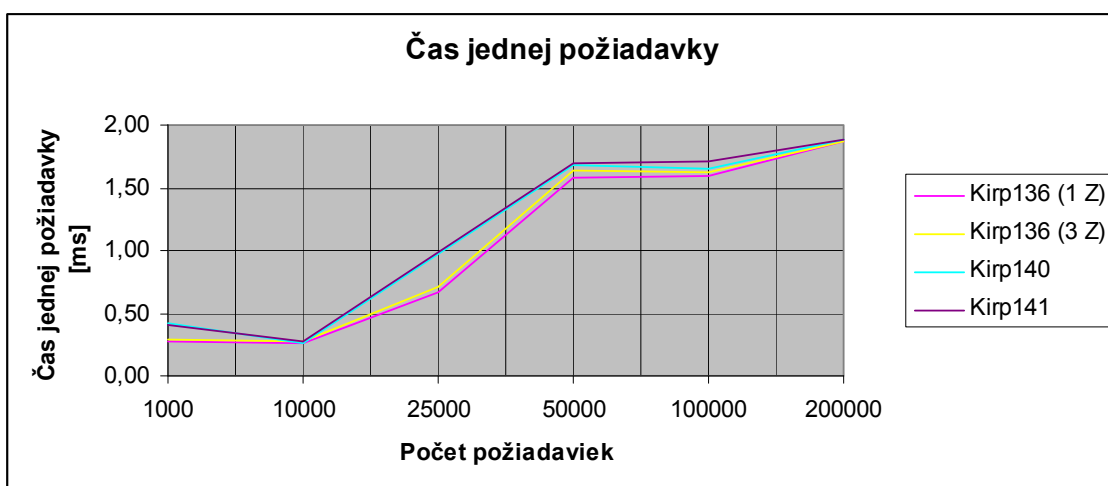
Počet požiadaviek	Súhrny čas súbežných požiadaviek [ms]			
	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141
1000	68,18	72,00	104,66	123,30
10000	67,06	67,34	66,77	70,57
25000	166,36	177,77	242,43	248,03
50000	396,44	408,67	418,56	424,81
100000	398,61	403,97	414,28	428,46
200000	465,85	467,41	469,17	471,49



Obr. 55 Súhrny čas súbežných požiadaviek (ab)

Tab. 14 Čas jednej požiadavky (ab)

Počet požiadaviek	Čas jednej požiadavky [ms]			
	Kirp136 (1 Z)	Kirp136 (3 Z)	Kirp140	Kirp141
1000	0,27	0,29	0,42	0,40
10000	0,27	0,27	0,27	0,28
25000	0,67	0,71	0,97	0,99
50000	1,59	1,64	1,67	1,70
100000	1,59	1,62	1,66	1,71
200000	1,86	1,87	1,88	1,89



Obr. 56 Čas jednej požiadavky (ab)

Z nameraných výsledkov, ktoré sme získali pomocou nástroja ab, je zreteľné, že Apache servery jednotlivých zón sú výkonnostne na rovnakej úrovni. Pri porovnaní výsledkov Apache servera globálnej zóny a Apache serverov lokálnych zón nie sú pozorovateľné výrazne zhoršenia výkonnosti v prípade Apache serverov lokálnych zón oproti Apache serveru globálnej zóny. Pretože tento testovací nástroj neumožňuje nasmerovať na server veľké množstvo súčasných požiadaviek, je chybovosť vybavenia požiadaviek všetkých serverov nulová.

Záver

Táto diplomová práca bola zameraná na oboznámenie sa so základnými pojmami virtualizácie, dôvodmi virtualizácie, jednotlivými virtualizačnými stratégiami a implementáciou virtualizácie v operačnom systéme Solaris 10 pomocou Solaris kontajnerov a zón. Hlavným cieľom diplomovej práce bolo inštalovanie viacerých web a sql serverov na jednom počítači a ich testovanie záťažovými testami.

Teoretická časť je zameraná na oboznámenie sa s virtualizáciou, hlavnými dôvodmi virtualizácie a s jednotlivými metódami virtualizácie. Pojmom virtualizácia sa najčastejšie myslí vytváranie virtuálnych počítačov nad fyzickým počítačom s cieľom znížiť počet fyzických počítačov. Medzi základné metódy virtualizácie patrí: simulácia, emulácia a úplná simulácia.

Základným virtualizačným nástrojom operačného systému Solaris 10 sú Solaris zóny, ktorých výhodou je, že všetky zóny majú vlastný súborový systém, vlastný administrátorský účet a vlastné sieťové rozhranie. Úlohou zón je poskytnúť prostriedky na vytváranie jedného alebo viacerých virtuálnych počítačov na jednej inštancii operačného systému. V jednom systéme môžeme mať dva druhy zón. Prvým druhom je globálna zóna, pričom ide o pôvodnú inštanciu operačného systému. Druhým druhom sú lokálne zóny, ktoré vznikajú oddelením z globálnej zóny. Každá zóna má svojich vlastných užívateľov, pričom užívatelia z jednej zóny nemôžu zasahovať do činnosti druhej zóny, bez ohľadu na to, či ide o užívateľov globálnej alebo lokálnej zóny. Jediný užívateľ, ktorý môže zasahovať do činnosti iných zón je root užívateľ globálnej zóny, ktorý ako jediný užívateľ má oprávnenie vytvárať, administrovať a vymazávať jednotlivé zóny.

Zóny využívajú pri svojej činnosti technológiu Solaris kontajnerov založenú na rozkúskovaní pôvodnej inštancie operačného systému na niekoľko virtuálnych inštancií. Výhodou je, že kontajnery využívajú pre svoju činnosť iba malú časť zdrojov systému. Kontajnery umožňujú lepšie riadenie zdrojov a zabezpečenie izolovaného prostredia pre vytvorené virtuálne počítače od fyzického počítača. Úlohou tohto izolovaného

prostredia je zamedziť vzniku situácie, v ktorej aplikácie z jednej zóny ovplyvňujú aplikácie inej zóny.

Na zvýšenie efektívnosti využívania dostupných zdrojov systému sa využíva Solaris technológia Fair Share Scheduler. Úlohou tejto technológie je pridelenie podielov zo zdrojov systému pre jednotlivé zóny. Na dosiahnutie rôznej úrovne hardvérovej izolácie nám slúži Solaris technológia Dynamic Resource Pools. Posledná časť teoretickej časti slúži ako návod na vytváranie zón priradením k novému poolu.

Prvá časť praktickej časti slúži ako návod na vytváranie zón priradením k predvolenému poolu, administráciu jednotlivých zón, mazanie zón systému, vytváranie zóny klonovaním inej zóny, migráciu zón a inštalovanie požadovaných aplikácií v jednotlivých zónach. K vytváraniu zón priradením k predvolenému poolu najčastejšie pristupujeme v prípade, ak nepovažujeme za dôležité zabezpečiť rôznu hardvérovú izoláciu zón. K tomuto spôsobu vytvárania zón môžeme byť prinútený aj v prípade, ak má fyzický počítač iba jeden procesor. Rozdiel v tomto spôsobe vytvárania zón oproti spôsobu vytvárania zón priradením k novému poolu je v konfigurácií, kde je nevyhnutné priradiť zónu k predvolenému poolu a priradiť konfigurovanej zóne vhodný počet podielov zo zdrojov systému.

Výhodou migrácie zón je, že v prípade, ak migrovaná zóna obsahuje určitú verziu niektorej aplikácie, a tá istá aplikácia je nainštalovaná v novej globálnej zóne, pričom verzia tejto aplikácie v novej globálnej zóne je novšia, respektíve staršia, ako verzia tej istej aplikácie v migrovanej zóne, nedôjde v migrovanej zóne počas migrácie k automatickému preinštalovaniu na verziu aplikácie, ktorá sa nachádza v globálnej zóne. Získame tak systém, v ktorom tá istá aplikácia bude mať v globálnej zóne inú verziu ako v lokálnej zóne, čo je ďalším dôkazom, že jednotlivé zóny systému sú úplne nezávisle od seba.

Najlepším nástrojom na aktualizáciu a inštaláciu jednotlivých aplikácií systému je pkgutil, ktorý nie je súčasťou operačného systému Solaris 10, preto je ho potrebné nainštalovať pomocou pkgadd. Výhodou pkgutil je to, že nám umožňuje inštalovať

aplikácie vo všetkých zónach naraz, za predpokladu, že sme prihlásení ako root užívateľ v globálnej zóne. V prípade potreby je možné zabezpečiť pomocou tohto nástroja inštaláciu aktualizácií a aplikácií iba v globálnej zóne pomocou úpravy konfiguračného súboru tohto nástroja. Výhodou Solaris zón je, že jednotlivé lokálne zóny, ktoré sú vytvorené po nainštalovaní aplikácií v globálnej zóne, obsahujú programové vybavenie globálnej zóny, ale s vlastnou konfiguráciou. To znamená, že aplikácie lokálnej zóny, ktoré boli prebrané z globálnej zóny, nepreberajú svoje nastavenia z globálnej zóny, preto je nutné nakonfigurovať prebrané aplikácie lokálnych zón, ktoré sú vytvorené po nainštalovaní aplikácií v globálnej zóne, nanovo.

Druhá časť praktickej časti bola zameraná na záťažové testovanie nainštalovaných web a sql serverov. Na testovanie sql serverov jednotlivých zón bol použitý nástroj MySQL5test a nástroj The MySQL Benchmark, ktoré sa vyznačujú svojou presnosťou. Na testovanie web serverov jednotlivých zón bol použitý nástroj Apache JMeter a nástroj ab - Apache HTTP server benchmarking tool. Podľa výsledkov získaných pomocou týchto nástrojov je jasné, že výkonnosť všetkých MySQL serverov, respektíve Apache serverov, bez rozdielu či ide o server globálnej zóny alebo server lokálnej zóny, je takmer rovnaká. To isté platí o vytážení procesora jednotlivými servermi.

Prínosom tejto diplomovej práce je, že nám poskytuje jednoduchý návod ako môžeme znížiť počet fyzických počítačov, pomocou virtualizácie. Zabezpečíme takto efektívnejšie využívanie dostupných zdrojov. Jednou z nevýhod vytvárania virtuálnych počítačov pomocou Solaris kontajnerov a zón je skutočnosť, že v prípade reštartovania globálnej zóny dôjde k automatickému reštartovaniu aj lokálnych zón, čiže k reštartovaniu všetkých počítačov. V prípade, ak budeme chcieť znížiť počet vynútených reštartov globálnej zóny a zvýšiť bezpečnosť všetkých zón, respektíve znížiť pravdepodobnosť zásahu neoprávnenej osoby do celého systému, čiže do všetkých zón, pomocou slabého miesta niektorej z aplikácií globálnej zóny, je vhodné používať globálnu zónu iba na konfigurovanie lokálnych zón, čiže napríklad, ak budeme mať požiadavku vytvoriť n web serverov, bude potrebné vytvoriť $n + 1$ zón. Takéto vytvorenie jednej zóny navyše by prispelo k zvýšeniu skutočného vytáżenia procesora, respektíve procesorov, fyzického počítača.

Zoznam použitej literatúry

- [1] IW: Virtualizácia – základné pojmy a princípy, Dostupné na internete: <<http://www.itnews.sk/tituly/infoware/free-clanky/2009-12-14/c130843-iw-virtualizacia-zakladne-pojmy-a-principy>>, Online, 12.4.2010.
- [2] HARAJDÍČ, M. *Virtualizácia v OS Solaris 10*: Semestrálny projekt 2. Bratislava: FCHPT STU v Bratislave, 2009. 37 s.
- [3] Klonovanie tučniaka alebo virtualizácia v Linuxe: úvod – Root.cz, Dostupné na internete: <<http://www.root.cz/clanky/klonovanie-tucniaka-alebo-virtualizacia-v-linuxe-uvod/>>, Online, 12.4.2010.
- [4] HARAJDÍČ M.: *Virtualizácia v OS Solaris 10*, Diplomový projekt. Bratislava: FCHPT STU v Bratislave, 2009. 55 s.
- [5] Náповѣда, Dostupné na internete: <http://publib.boulder.ibm.com/infocenter/tivihelp/v11r1/index.jsp?topic=/com.ibm.tivoli.tpm.scenario.doc/virtual/scom_introduction_zn.html>, Online, 12.4.2010.
- [6] Zones and Containers FAQ at OpenSolaris.org, Dostupné na internete: <<http://www.opensolaris.org/os/community/zones/faq/>>, Online, 12.4.2010.
- [7] Solaris kontajnery: zdieľanie systémových zdrojov – Root.cz, Dostupné na internete: <<http://www.root.cz/clanky/solaris-kontajnery-zdielanie-systemovych-zdrojov/>>, Online, 12.4.2010.
- [8] MySQL :: MySQL 5.0 Reference Manual : 7.1.3 The MySQL Benchmark Suite, Dostupné na internete: <<http://dev.mysql.com/doc/refman/5.0/en/mysql-benchmarks.html>>, Online, 12.4.2010.
- [9] JMeter – Apache JMeter, Dostupné na internete: <<http://jakarta.apache.org/jmeter/>>, Online, 12.4.2010.