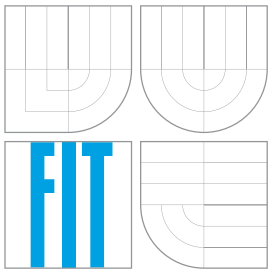
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

NÁSTROJE PRO PROFILOVÁNÍ APLIKACÍ V UNIXU

UNIX TOOLS FOR APPLICATION AND SYSTEM PROFILING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

DAVID DRESSLER

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. TOMÁŠ KAŠPÁREK

BRNO 2016

Zadání bakalářské práce/18178/2015/xdress00

Vysoké učení technické v Brně - Fakulta informačních technologií

Centrum výpočetní techniky Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel: **Dressler David**
Obor: Informační technologie
Téma: **Nástroje pro profilování aplikací v Unixu**
Unix Tools for Application and System Profiling
Kategorie: Operační systémy

Pokyny:

1. Nastudujte dostupné nástroje pro ladění aplikací a systému v zadaných unixových operačních systémech s primárním cílením na Linux.
2. Proveďte porovnání nalezených nástrojů z hlediska složitosti použití a invazivnosti (testovací vs. produkční prostředí).
3. Dle pokynů vedoucího navrhněte alespoň 2 modelové situace pro ladění chyb resp. výkonu.
4. Ukažte využití vhodných nástrojů pro nalezení, opravu a verifikaci odstranění problémů s výkonem na modelových situacích.

Literatura:

- DTrace, http://docs.oracle.com/cd/E19253-01/817-6223/, navštíveno 2015-10-27
- OProfile, http://oprofile.sourceforge.net, navštíveno 2015-10-27
- dle pokynů vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kašpárek Tomáš, Ing.**, CVT FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Cílem této práce bylo demonstrování využití nástrojů pro profilování aplikací a systémů. Nejprve byly tyto nástroje nalezeny a nastudovány. Také byly rozděleny do kategorií podle jejich účelu. Dále byly tyto nástroje porovnány z hlediska složitosti použití a invazivnosti. Výsledkem bylo rozdělení do tří skupin podle míry složitosti použití nebo invazivnosti. Jako technologie, využité při vytváření modelů, byly zvoleny Apache server a NFS server. Pro vytvoření modelů těchto serverů bylo využito virtualizace systémů pomocí technologie hyper-v. Vytvořeny byly čtyři virtuální stroje. Jeden pro Apache server, další pro NFS server, třetí pro zrcadlení obsahu Apache serveru a poslední pro generování zátěže. Poslední částí této práce je demonstrace využití nalezených nástrojů na vytvořených modelových situacích.

Abstract

The main goal of this thesis was to demonstrate usage of tools for application and system profiling. Initially, these tools was found and studied. They was also divided into categories according to their purporse. After that, these tools was compared according to their complexity of use and invasiveness. As the result of this comparation, these tools was divided into three groups, that express measure of complexity and invasiveness. As technology, used for creating models, was choosen Apache server and NFS server. Virtualzation by hyper-v technology was used for putting these models into operation. There was created four virtual machines. Fist one for Apache server, another one for NFS server. Third was for mirroring content of Apache server and the last one for load generation. The last part of this thesis was to demonstrate usage of found tools on the created models.

Klíčová slova

analýza výkonu, Apache server, NFS server, procfs, sysfs, Linux, invazivnost, generování zátěže

Keywords

performance analysis, Apache server, NFS sever, procfs, sysfs, Linux, invasiveness, load generation

Citace

DRESSLER, David. *Nástroje pro profilování aplikací v Unixu*. Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Kašpárek Tomáš.

Nástroje pro profilování aplikací v Unixu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Tomáše Kašpárka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
David Dressler
17. května 2016

Poděkování

Chtěl bych poděkovat svému vedoucímu práce Ing. Tomáši Kašpárkovi za pomoc a cenné rady při vypracování této práce.

Obsah

Kapitola 1

Úvod

Účelem profilování aplikací a systémů je optimalizace výkonu. Aby byla optimalizace výkonu možná, je nutné analyzovat činnost aplikací a jejich dopad na jednotlivé části systému. Jedná se zejména o využití procesoru, paměti, disku a sítě.

Musí tedy existovat nějaké možnosti, jak tyto části systému sledovat a získávat o nich informace. K tomuto účelu slouží různé nástroje. Každý nástroj se zaměřuje na nějakou část nebo části systému. Díky těmto nástrojům jsme poté schopni odhalit souvislosti, které nám pomohou identifikovat konkrétní problémy.

K odhalení problémů s výkoností nestačí pouze sledovat hodnoty celkového vytížení komponent systému. To můžeme brát spíše jako indikaci nějakého potenciálního problému, pokud jsou tyto hodnoty vytížení vysoké. Důležité je při odhalování problému postupovat systematicky a postupně jít hlouběji do systému. To zahrnuje například sledování akcí konkrétního procesu nebo sledování vstupního a výstupního provozu na disku.

Tato práce se zabývá zejména postupy pro nalezení příčin problémů prostřednictvím využití různých nástrojů, což vede k jejich odstranění a vylepšení využití výkonu systému.

V této práci jsou nejprve představeny různé nástroje, které se při profilování aplikací využívají a také je rozebrán obsah adresářů /proc a /sys. Uvedené nástroje jsou rozděleny podle jejich účelu. Každý nástroj je stručně popsán včetně uvedení některých jeho výhod a nevýhod. Další kapitola je věnována porovnání těchto nástrojů z hlediska složitosti použití a invazivnosti. Jedná se o dva důležité aspekty při využívání těchto nástrojů. Složitost použití je relevantní hlavně pro toho, kdo chce tyto nástroje využít. Invazivnost ovlivňuje celkový výkon systému a je důležité mít to na paměti, když se rozhodneme nástroj s vyšší invazivností využít. Samotná demonstrace využití nástrojů probíhá na vytvořených modelech, které jsou popsány v další kapitole. Při tvorbě modelů byly využity technologie webového serveru Apache a NFS serveru. V praxi se jedná o velice časté způsoby využití serveru a pomocí nich bude demonstrace využití nástrojů probíhat na simulovaném provozu. Poslední kapitola se věnuje samotným ukázkám využití nástrojů za účelem demonstrování postupů při ladění výkonu.

Kapitola 2

Přehled použitých nástrojů

V této kapitole budou představeny různé nástroje pro profilování aplikací na systému Linux. Každý nástroj bude stručně představen včetně uvedení jeho dobrých či špatných vlastností. Tyto nástroje budou nadále využívány k testování a ladění chyb a výkonu. Dále se v této kapitole budeme věnovat adresářům /proc a /sys. Na konci této kapitoly je zahrnuta tabulka, která shrne uvedené nástroje a uvede jejich rozhraní. Více informací k většině nástrojů lze nalézt na manuálových stránkách[8].

2.1 Adresáře /proc a /sys

Adresáře /proc (man. proc(5)) a /sys[11], které najdeme v kořenovém adresáři linuxových systémů, samozřejmě nejsou nástroje pro profilování aplikací. Jsou to virtuální souborové systémy, které jsou dostupné v těchto adresářích a slouží ke zpřístupnění informací poskytovaných jádrem systému. Tyto informace jsou poskytovány ve formě různých souborů. Většina těchto souborů je pouze pro čtení, některé ale umožňují editovat nastavení jádra systému a tím přímo ovlivňovat jeho chování. Pro zajímavost si lze všimnout, že všechny tyto soubory nezabírají na disku žádné místo. Je to tím, že obsah těchto souborů je generován z informací poskytovaných jádrem systému až při čtení jejich obsahu. Jelikož většina nástrojů pouze čte a formátuje informace získané pomocí těchto souborů, je důležité se o těchto virtuálních souborových systémech něco dozvědět. Obsah těchto adresářů se může mírně lišit pro různé verze jádra systému. V této práci využívám systém Linux Lite 2.6. [9] s verzí jádra 3.13.0-76-generic.

/proc

Původně byl tento virtuální souborový systém navržen pro přístup k informacím o procesech. Postupně se ale do něj začali přidávat i další informace. Obsah adresáře /proc tedy můžeme rozdělit na informace o procesech a informace poskytované jádrem systému.

V adresářích procesů (/proc/[pid]) nalezneme:

- fd/ - obsahuje odkazy na použité deskriptory souborů.
- task/ - obsahuje podadresář pro každé vlákno procesu.
- root - odkaz na kořenový adresář.

- cmdline - obsahuje název aplikace a případně její parametry.
- exe - symbolický odkaz na danou aplikaci.
- io - obsahuje vstupní a výstupní statistiky procesu.
- limits - obsahuje limity pro využívání systémových prostředků.
- maps - zobrazí aktuálně namapované oblasti paměti a jejich přístupová práva.
- stat (status) - zpřístupňuje informace o stavu daného procesu.
- statm - obsahuje informace o paměti využívané daným procesem.

Ostatní položky v adresáři /proc:

- bus/ - obsahuje informace o sběrnicích.
- sys/ - obsahuje soubory korespondující k proměnným jádra systému. K jejich modifikaci se využívá nástroj sysctl, který tedy slouží k modifikaci parametrů jádra za běhu systému.
- cpuinfo - poskytuje informace o procesoru.
- devices - zobrazuje seznam všech zařízení a jejich hlavního čísla.
- diskstats - zobrazuje vstupní a výstupní statistiky pro každé blokové zařízení.
- filesystems - poskytuje seznam podporovaných souborových systémů.
- interrupts - obsahuje informace o přerušeních.
- iomem - obsahuje aktuální paměťovou mapu pro každé fyzické zařízení.
- ioports - zobrazuje seznam registrovaných vstupních a výstupních portů.
- kcore - reprezentuje fyzickou paměť systému.
- loadavg - obsahuje informace o průměrném zatížení.
- locks - zobrazuje aktuálně uzamknuté soubory.
- modules - zobrazuje seznam modulů zavedených do jádra systému.
- meminfo - zobrazuje informace o využití paměti.
- partitions - obsahuje informace o diskových oddílech.
- slabinfo - poskytuje informace o cache paměti jádra systému.
- swaps - obsahuje informace o používaném swapu.
- timer_list - poskytuje seznam všech časovačů.
- version - obsahuje informace o verzi jádra systému.
- zoneinfo - zobrazuje informace o pamětových zónách.

Nebyly zde vyjmenovány a popsány všechny soubory, které v adresáři /proc najdeme. Vybral jsem některé zajímavé položky pro představu, jaké informace zde lze najít. Podrobnější rozbor lze naleznout na manuálových stránkách proc(5).

/sys

Tento virtuální souborový systém byl navržen, aby reprezentoval model zařízení tak, jak ho vidí jádro systému. Umožňuje exportovat objekty jádra, jejich atributy a propojení objektů do uživatelského prostředí. Obsahuje tedy informace o veškerém hardwaru, ovladačích, sběrnicích a jejich propojení. Má striktně definovanou strukturu.

Struktura /sys:

- block/ - obsahuje všechna bloková zařízení.
- bus/ - obsahuje všechny registrované sběrnice. Každá má defaultně dva podadresáře.
  - device/ - pro všechna zařízení na sběrnici.
  - driver/ - všechny ovladače pro konkrétní sběrnici.
- class/ - pro každou třídu zařízení je zde podadresář.
- dev/ - obsahuje tyto dva podadresáře, které odkazují na zařízení.
  - block/
  - char/
- devices/ - obsahuje informace pro všechna zařízení.
- firmware/ - poskytuje přístup pro manipulaci s objekty a atributy firmware.
- fs/ - obsahuje informace a nastavení pro každý připojený souborový systém, roztříděný podle typu souborového systému.
- hypervisor/ - tento podadresář využívá hypervizor.
- kernel/ - obsahuje objekty jádra, jejich atributy a vztahy mezi těmito objekty.
- module/ - každý nahraný modul má svůj adresář, který obsahuje informace a nastavení daného modulu.
- power/ - obsahuje soubory s informacemi o stavu napájení.

2.2 Nástroje pro analýzu síťového rozhraní

ifconfig

Nástroj ifconfig slouží ke konfiguraci síťového rozhraní. Využívá se k zobrazení nastavení síťových rozhraní a také k jejich modifikaci na daném systému. Například lze zobrazit informace o IP adresách, fyzických adresách či počtu příchozích a odchozích paketů. Tyto informace jsou nápomocné, pokud zjišťujeme, jestli je síť správně nastavená.

+ Je vždy k dispozici.
+ Vhodný k získání základních informací o nastavení sítě.
- Je vhodnější pouze pro menší zásahy do síťového rozhraní.

Podobné nástroje: iproute2, který nahrazuje net-tools, mezi který patří i nástroj ifconfig. Narozdíl od net-tools, který používá pro změnu nastavení adresář /proc, iproute2 využívá pro komunikaci s jádrem netlink rozhraní. Více informací o netlink rozhraní najdeme v manuálu netlink(7).

ping

Pomocí nástroje ping lze posílat požadavky protokolu ICMP (ping6 pro ICMPv6) na síťového hosta. Tímto způsobem se testuje spojení mezi dvěma síťovými hosty.

+ Jednoduché použití.
+ Poskytuje informace o ztrátovosti a latenci v síti.

tcpdump

Nástroj tcpdump slouží k analýze síťových paketů. Pomocí tohoto nástroje lze zachycovat pakety a posléze je analyzovat. Pomocí parametrů lze nastavit různé možnosti zachycování paketů. Například na jakém zařízení potřebujeme pakety zachycovat či na jakém portu a mnoho dalších možností.

+ Je skoro v každé instalaci systému již připraven.
+ Možnost sledování provozu na síti.
- Složitější filtrování paketů podle pravidel.
- Pouze pro systémového uživatele.

Podobné nástroje: Wireshark

iptraf

Nástroj iptraf umožňuje monitorování sítě LAN. S jeho pomocí lze sledovat různé síťové statistiky. Nabízí jednoduché barevné konzolové rozhraní. Využívat ho může pouze uživatel s právy systémového uživatele.

+ Grafické rozhraní.

+ Přehledně poskytuje statistiky o provozu na síti.
- Pouze pro systémového uživatele.

traceroute

Pomocí nástroje traceroute můžeme zjistit, jakou "cestou" se vydá náš požadavek na vzdáleného hosta. Jinými slovy nám poskytne informace o směrovačích, kterými musí náš požadavek projít, než se dostane ke svému cíli. Tímto způsobem lze například zjistit, na kterém směrovači nastala případná chyba, pokud se náš požadavek nedostane do cíle.

+ Pomůže se zorientovat v síti.
+ Může objevit, na kterém místě v sití je problém.
- Zařízení mohou být maskovaná.

vnstat

Nástroj vnstat slouží k zobrazení statistik provozu sítě. Je závislý na informacích, které poskytuje jádro systému. Informace o provozu na síti získává z adresářů /proc a /sys.

+ Přehledné zobrazení statistik.
+ Informace získávány z adresářů /proc a /sys, tudíž nezatěžuje síť.

netstat

Pomocí nástroje netstat lze zobrazit informace o síťových připojeních. Mezi tyto informace patří například směrovací tabulky, statistiky rozhraní a členství v multicastových skupinách.

+ Více možností nastavení vypisovaných informací.
- Vypisuje pouze snímek aktuálního stavu.

Podobné nástroje: Socket Statistics

ntop

Nástroj ntop slouží k zobrazování statistik na síti. Zobrazuje seznam hostů, kteří aktuálně používají síť a k nim přidružené různé informace o jejich provozu na síti. Využívá grafického webového rozhraní.

+ Grafické rozhraní.
+ Přehledné zobrazení včetně grafů.
+ Možnost vidět statistiky provozu konkrétních uživatelů.

2.3 Nástroje pro diskové oddíly

df

Nástroj df je užitečný, pokud potřebujeme vědět informace o pevném disku. Nabízí nám přehledné informace o všech připojených discích a využití jejich kapacit.

+ Jednoduchost použití.
+ Rychlá cesta k zobrazení informací o pevných discích.

iostat

Nástroj iostat slouží k zobrazení statistik o procesoru a vstupních a výstupních statistikách diskových oddílů. Není nutné vypisovat statistiky o všech diskových oddílech připojených v systému, můžeme specifikovat, o kterých oddílech informace vyžadujeme.

+ Vypisuje informace nejen o diskových oddílech, ale také o procesoru.
- Vypisuje pouze snímek aktuálního stavu.

blktrace

Nástroj blktrace slouží k vystopování vstupního a výstupního provozu na zadaném blokovém zařízení. Například pevného disku připojeného v systému. Výstupem jsou informace o provedených operacích a celkové statistiky.

+ Možnost zobrazení veškerého vstupního i výstupního provozu na disku.
- Vysoká invazivnost.

lsof

Pomocí nástroje lsof můžeme vypsat všechny aktuálně otevřené soubory v systému a to včetně souborů zařízení a podobně. Vypisuje také otevřené adresáře. Mezi zobrazené informace patří například název procesu, který soubor využívá, jeho pid, jméno uživatele, velikost souboru a další.

+ Lze kombinovat s nástroji, jako je například grep.
- Vypisuje velké množství záznamů. Ideálně je potřeba vědět, co konkrétně hledáme.

iotop

Nástroj iotop slouží ke sledování vytížení disku v reálném čase. Vypisuje aktuální vytížení disku při čtení a zápisu dat jednotlivými procesy.

+ Zobrazuje periodicky aktuální stav.
+ Možnost zvolit sledované procesy.
- Pouze pro systémového uživatele.

2.4 Nástroje pro analýzu systému

ps

Nástroj ps vypíše všechny aktuální procesy na daném počítači. Nevypisuje je v reálném čase, pouze udělá snímek stavu, který je aktuální v době provedení příkazu. U každého procesu zobrazí informace dle zadaných parametrů, například o jeho pid a stavu.

- Vypisuje pouze snímek aktuálního stavu.
+ Různé možnosti výpisu informací.

top

Nástroj top slouží k zobrazení všech spuštěných procesů na daném počítači. Zobrazuje informace v reálném čase. Mezi tyto informace patří například stav paměti RAM, vytížení procesoru či status jednotlivých procesů. Zobrazování těchto údajů lze upravit.

+ Zobrazuje periodicky aktuální stav.
+ Možnost upravit, které informace jsou zobrazené.
+ Možnost zvolit sledované procesy.

Podobné nástroje: htop, atop

free

Pomocí nástroje free se zjišťují informace ohledně paměti RAM. Konrétně o využité a volné kapacitě této paměti. Také zobrazuje informace o paměti pro swap a vyrovnávací paměti jádra.

+ Přehledné zobrazení.
+ Jednoduchost použití.

pmap

Nástroj pmap slouží k zjištění informací ohledně paměťové mapy zadaného procesu. Konkrétní proces, na který chceme tento nástroj použít, identifikujeme pomocí čísla pid. Nejprve tedy musíme zjistit pid procesu například pomocí nástroje ps či nástroje top.

+ Možnost vidět přesnější informace ohledně využití paměti konkrétním procesem.
+ Možnost výpisu informací pro více procesů najednou.

vmstat

Nástroj vmstat slouží k zobrazování statistik virtuální paměti. Zobrazuje statistiky procesů, paměti, stránkování, vstupních a výstupních informací blokových zařízení, discích a procesoru.

+ Veškeré statistiky o částech systému v jednom nástroji.
+ Možnost upřesnit, které informace chceme vypsat.

mpstat

Pomocí nástroje mpstat lze zobrazit statistiky procesoru. Tyto statistiky lze získat i pomocí jiných nástrojů, ty však vypisují i další informace.

+ Přehledné informace o procesoru.
- Vypisuje pouze snímek aktuálního stavu.

strace

Nástroj strace se využívá pro sledování systémových volání a signálů. Monitoruje vždy specifikovaný proces, například proces mysql databáze, který se identifikuje jeho číslem pid.

+ Možnost vidět veškeré systémové volání a signály, které proces využívá.
- Vysoká invazivnost.

ltrace

Použití tohoto nástroje je podobné jako u nástroje strace. Slouží ovšem k jinému účelu a to k zobrazení volání funkcí dynamických knihoven daným procesem.

+ Možnost sledovat veškerá volání dynamických knihoven daným procesem.
- Vysoká invazivnost.

2.5 Nástroje pro monitorování systému

sar

Pomocí nástroje sar můžeme buď monitorovat systémovou činnost v reálném čase, nebo shromažďovat tyto údaje a poté je analyzovat ze sesbíraných dat. Můžeme monitorovat činnost procesoru, paměti, síťové statistiky či vstupní a výstupní aktivity systému.

+ Možnost dlouhodobého monitorování systému.
+ Je možné upřesnit, které informace chceme vypsat či sledovat.

conky

Conky je nástroj na monitorování systému v reálném čase. Nabízí informace o procesoru, paměti, síti, disku, o tom jak dlouho systém běží či využití swapu.

+ Grafické zobrazení informací.
+ Přehledné.
+ Aktuální informace.

Podobné nástroje: gkrellm

cacti

Cacti je nástroj, který sbírá informace o systému a následně z nich dokáže vygenerovat grafy. Má grafické webové rozhraní, které slouží k jeho ovládání. V tomto rozhraní najdeme nastavení a také vyhotovené grafy. Zároveň také můžeme vytvářet nové grafy, které budou obsahovat informace podle naší potřeby. Hodí se pro dlouhodobé sledování systému.

+ Dlouhodobé monitorování systému.
+ Grafické prostředí.
+ Různé možnosti nastavení.

2.6 Nástroje pro analýzu výkonu

perf[1]

Nástroj perf slouží k analýze výkonu. Využívá různé čitače k analýze jak hardware, tak i software zařízení. Například využití procesoru nebo softwarové tracepointy. Umožňuje velké množství možností sledování. Je tedy potřeba nastavit, co přesně chceme sledovat. Nástroj perf se používá s několika jeho podpříkazy.

- Složitější použití nástroje.
- Vysoká invazivnost.

Podobné nástroje: oprofile

oprofile[6]

Nástroj oprofile slouží stejně jako nástroj perf k analýze výkonu. Slouží například ke sledování obsluhy přerušení nebo využívání dynamických knihoven. Stejně jako s nástrojem perf i tady můžeme ovlivnit, co chceme sledovat.

- Složitější použití nástroje.
- Vysoká invazivnost.

Podobné nástroje: perf

systemtap [7]

Systemtap je nástroj, který také slouží ke sbírání dat o činnosti počítače. Od ostatních se liší tím, že je nutné psát skripty, abychom mohli nějaká data sesbírat. Celá ideologie tohoto nástroje je v tom, že se pojmenuje nějaká událost, a když nastene, tak se provede příslušná obsluha. Například tak můžeme sledovat, za jak dlouho proběhne nějaká funkce či vypršení nějakého časovače.

- Nutnost psaní skriptů.
- Vysoká invazivnost.

2.7 Generátory zátěže

JMeter [3]

Tento nástroj slouží na testování webového serveru Apache. Je s ním možné generovat zátěž a to budeme potřebovat k demonstraci postupů při hledání problémů s výkonností.

+ Grafické rozhraní.
+ Různé možnosti nastavení při generování zátěže.

iozone

Nástroj iozone (man. iozone(1)) slouží ke generování operací zápisu a čtení nad soubory. V této práci bude využit pro generování zátěže na NFS server.

+ Možnost generování různých operací pro práci se soubory.
+ Poskytuje statistiky.
+ Možnost statistiky zapsat do souboru typu xls.

2.8 Shrnutí

Linuxových nástrojů, které lze využít pro ladění aplikací je mnoho. Určitě zde v této kapitole nejsou uvedeny všechny, které existují. Liší se nejen konkrétním využitím pro různé části systému, ale také svým rozhraním, možnostmi úpravy jejich použití různými parametry a především také složitostí jejich použití a invazivnosti.

Nástroj	terminál	grafické rozhraní	skriptovací nástroje
Nástroje pro analýzu síťového rozhraní			
ifconfig	x		
ping	x		
tcpdump	x		
iptraf		x	
traceroute	x		
vnstat	x		
netstat	x		
ntop		x	
Nástroje pro diskové oddíly			
df	x		
iostat	x		
blktrace	x		
lsof	x		
iotop	x		
Nástroje pro analýzu systému			
ps	x		
top	x		
free	x		
pmap	x		
vmstat	x		
mpstat	x		
strace	x		
ltrace	x		
Nástroje pro monitorování systému			
sar	x		
conky		x	
cacti		x	
Nástroje pro analýzu výkonu			
perf	x		
oprofile	x		
systemtap		x	x
Generátory zátěže			
jmeter		x	
iozone	x		

Tabulka 2.1: Přehled nástrojů

Kapitola 3

Porovnání nástrojů

V této kapitole budou nástroje uvedené v kapitole 2 porovnány z hlediska obtížnosti použití a invazivností.

3.1 Složitost použití

Většina nástrojů, které byly použity v této práci, využívá rozhraní příkazové řádky (CLI). Vyjímkou jsou pouze nástroje iptraf, ntop, conky, cacti a jmeter, které mají grafické uživatelské rozhraní (GUI). Nástroj iptraf je jediným z uvedených nástrojů, který využívá grafického uživatelského rozhraní v prostředí terminálu.

Při práci s nástroji, které využívají rozhraní příkazové řádky, se k úpravě jejich chování využívá vstupních a výstupních parametrů. Význam parametrů se pro každý nástroj liší. U většiny nástrojů jsou jejich definované parametry nepovinné. Tyto nástroje tedy mají nastavené nějaké defaultní chování. Některé parametry ale mohou být definovány jako povinné a nástroj je pak při spuštění vyžaduje. Důležité však ale je pochopení výsledku, který nám je těmito nástroji poskytnut. Mnoho nástrojů vyžaduje znalost operací a různých souvislostí, které v systému probíhají, aby pro nás byly užitečné. Nástroje systemtap, perf a oprofile například vyžadují širší znalosti jádra systému. U nástroje systemtap se dokonce vyžaduje psaní skriptů, které obsahují specifické funkce pro tento nástroj. V kontrastu k těmto nástrojům můžeme uvést například nástroj vnstat, který nám poskytuje statistiky počtu přenesených dat na síti v přehledném výpisu. Výhodou nástrojů využivajících rozhraní příkazové řádky je také to, že můžeme přizpůsobit a také kombinovat jejich použití. Můžeme tedy například vytvořit vlastní skript, který bude kombinovat využití více nástrojů a plnit úlohu, kterou potřebujeme. K tomuto účelu se využívá například jazyku bash. A díky službě cron můžeme tyto skripty nebo jiné úlohy spouštět periodicky.

Složitost použití nástrojů s grafickým uživatelským rozhraním se odvijí od složitosti prováděných úkonů. Chování nástroje conky (2.5) lze ovlivnit jeho parametry nebo přímo v konfiguračním souboru /etc/conky/conky.conf. Jedná se však ve většině případů pouze o nastavení vzhledu, pozice na ploše a intervalů výpisu informací. Nástroj iptraf (2.2) však již nabízí různé možnosti sledování a přehledu statistik. Nástroj ntop (2.2) nabízí komplexnější grafické rozhraní. Ke statistikám přidává například grafy a prezentování informací je přehlednější než u nástroje iptraf. Nicméně, obsahuje složitější nastavení. Totéž lze říci o nástroji cacti (2.5). Ten jelikož slouží k monitorování více parametrů systému, obsahuje také komplexnější možnosti nastavení. Nástroj Jmeter (2.7) nabízí mnoho možností, jak si upravit generovanou zátěž. To souvisí s typy požadavků na server Apache. Pro pochopení

fungování a nastavení je vhodné využít nějakého návodu[1]. Nevýhodou oproti nástrojům, které využívají rozhraní příkazové řádky je to, že můžeme provádět pouze změny nastavení a operace, které nám jsou nabízeny. Tím přicházíme o jistou variabilitu možností použití, kterou nástroje využívající rozhraní příkazové řádky disponují.

3.2 Invazivnost

Všechny nástroje, které jsou použity v této práci, nám mají nějakým způsobem pomáhat při zjišťování problémů v systému. Poskytují nám informace, které od nich očekáváme podle jejich účelu. Nicméně, každý nástroj musí nějakým způsobem tyto informace získávat, aby nám je poté mohl zprostředkovat.

Způsoby získávání informací ze systému jsou:

- Čtení obsahu adresářů /proc a /sys.
- Systémová volání (syscall).

Invazivnost nástrojů při čtení obsahu adresářů /proc a /sys závisí na způsobu využití, jelikož se obsah souborů v těchto adresářích generuje až při jejich čtení. Záleží tedy na tom, kolik souborů daný nástroj z těchto adresářů využívá a také jestli tyto soubory používá periodicky nebo je čte pouze jednou. Také pokud vypisujeme pouze část souboru například v kombinaci s nástrojem grep, musí se tento soubor prohledat celý, aby se našly všechny odpovídající hledané řádky. To platí také pro nástroje. Například příkaz *ps ax |grep "apache"* vypisuje pouze procesy Apache, musí však nejprve projít všechny procesy, aby nám pak poskytnul pouze hledané výsledky.

Nástroje, které nevyužívají pouze obsahu adresářů /proc a /sys musí k získávání informací využít systémových volání prostřednictvým knihovních funkcí, které tuto komunikaci s jádrem systému umožňují, což má větší vliv na probíhající operace. Důvodem je, že tyto požadavky jsou vyřizovány na úkor jiných operací, které by mohly v tu chvíli probíhat. To vede k pomalejší odezvě systému. Nástroje s vysokou invazivností tak v důsledku prezentují více či méně zkreslené výsledky. Jejich použití je tedy vhodnější v případě, kdy potřebujeme zjistit, co přesně probíhá za operace nebo komunikaci. To znamená například analýzu komunikace na síti nebo vystopování vstupního a výstupního provozu na disku.

Je tedy třeba si uvědomit, že i když tyto nástroje slouží k tomu, aby nám poskytovaly informace, které nám pomohou při hledání problému, samy spotřebovávají nějaké prostředky systému a tím ho také zatěžují. Z kategorií nástrojů uvedených v kapitole 2 mají všechny kromě nástrojů pro analýzu výkonu a generátorů zátěže zástupce méně i více invazivních nástrojů. Nástroje pro analýzu výkonu se vyznačují nejvyšší invazivností z uvedených nástrojů. U generátorů zátěže bych se invazivností nezabýval. Důvodem je, že nám mají pomáhat při simulování reálného provozu, ne poskytováním informací.

[1] https://www.digitalocean.com/community/tutorials/how-to-use-apache-jmeter-to-perform-load-testing-on-a-web-server

3.3 Shrnutí

Při porovnávání nástrojů byly zohledněny skutečnosti popsané v této kapitole. Při porovnávání z hlediska složitosti použití byly nástroje rozděleny podle uživatelského rozhraní. U nástrojů využívajících rozhraní příkazové řádky rozhodovala zejména složitost využití různých parametrů, forma poskytovaných informací a také potřebné znalosti systému nutné k využití těchto nástrojů. U nástrojů s grafickým uživatelským rozhraním zejména složitost prováděných úkonů. Při porovnávání invazivnosti nezáleželo na složitosti použití nástrojů. Rozhodujícími faktory byly způsob získávání požadovaných informací, typ těchto informací a také způsob jejich využívání. Nižší invazivnost vykazovaly nástroje, které využívaly obsahu adresářů /proc a /sys.

V následující tabulce je u využitých nástrojů uvedena složitost použití a invazivnost. Vyjádřena je počtem hvězdiček v rozsahu 1 - 3, jedná se o rozdělení nástrojů uvedených v tabulce 2.1 v kapitole 2 do tří skupin.

Tyto skupiny podle počtu hvězdiček vyjadřují:

- Nízkou složitost nebo invazivnost (*).
- Střední složitost nebo invazivnost (**).
- Vysokou složitost nebo invazivnost (***).

Nástroj	Složitost použití	Invazivnost	Nástroj	Složitost použití	Invazivnost
Nástroje pro analýzu síťového rozhraní			Nástroje pro analýzu systému		
ifconfig	*	*	ps	**	**
ping	*	*	top	**	**
tcpdump	***	***	free	*	*
iptraf	**	***	pmap	**	**
traceroute	*	*	vmstat	*	*
vnstat	*	**	mpstat	*	**
netstat	*	**	strace	**	***
ntop	**	***	ltrace	**	***
Nástroje pro diskové oddíly			Nástroje pro monitorování systému		
df	*	*	sar	***	**
iostat	**	**	conky	*	**
blktrace	***	***	cacti	**	***
lsof	**	***			
iotop	**	***			
Nástroje pro analýzu výkonu			Generátory zátěže		
perf	**	***	jmeter	***	-
oprofile	***	***	iozone	**	-
systemtap	***	***			

Tabulka 3.1: Porovnání nástrojů

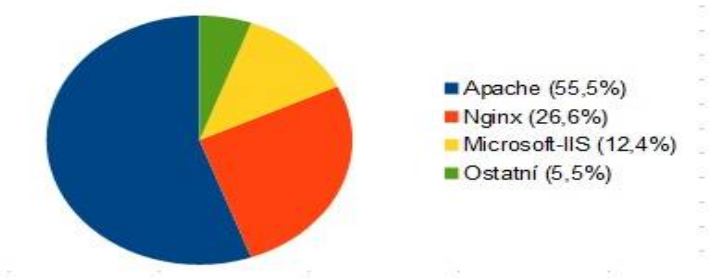
Kapitola 4

Modelové situace

V této kapitole budou představeny technologie, které budou využity při tvorbě modelových situací. Jedná se zejména o webový server Apache a NFS server. Také budou popsány vytvořené modely. Na těchto vytvořených modelech bude probíhat demonstrace ladění chyb a výkonu pomocí nástrojů uvedených v kapitole 2.

4.1 Webový server Apache

Webový server je obecně hojně využívaná technologie. Jedním z nejpoužívanějších je server Apache [2]. Momentálně má největší podíl využití ze všech webových serverů. Proto jsem zvolil právě Apache pro vytvoření modelové situace ladění výkonu a chyb webového serveru.



Obrázek 4.1: Podíl využití webových serverů [19]

Webový portál

Jako běžné využití webového serveru bychom mohli označit spojení s nějakou databází (například MYSQL [12]) a vytvoření webového portálu. Příkladem takového využití je Wikipedia.

Webový server obsluhuje příchozí požadavky. Ty se týkají přístupu k webovým stránkám či různým jiným skriptům. Poskytuje tedy také podporu pro různé programovací jazyky (například php), pomocí kterých se řídí komunikace s databází nebo také například zpracování formulářů či autentizace.

Databáze se v tomto případě využívá pro uchovávání dat, které jsou prezentovány na webových stránkách. S využitím databáze se však pojí nejen prezentování uchovávaných dat

ale také další úkony. Práci s databází můžeme rozdělit na vyhledávání, vkládání, mazání a úpravu dat.

V databázi se neuchovávají pouze data, která mají být prezentována na webových stránkách, ale také přístupové údaje uživatelů a administrátorů či další řídící prvky jako nastavení databáze a podobně.

Zrcadlení webového serveru

Jako méně běžné využití webového serveru si dovolím uvést zrcadlení serverů [17] (server mirroring). Účelem zrcadlení serverů je vytvoření přesné kopie obsahu serveru na jiném serveru. Tím se pro webový server docílí toho, že webová stránka bude dostupná i z jíného místa.

Hlavní důvod tohoto využití je zrychlení přístupu na webovou stránku, která je geograficky vzdálená. Například kdyby originální stránka byla na serveru v USA a chtěli bychom zrychlit přístup třeba z České republiky, tak by se vytvořil zrcadlený server v České republice.

Zrcadlením se také mohou vytvářet zálohy serveru. V případě výpadku hlavního serveru se pak mohou požadavky přesměrovat na záložní server a stránky budou pořád dostupné a funkční.

4.2 NFS server

NFS [16] (Network File System) je síťový souborový systém určený pro přístup k souborům na síti. Je postavený na modelu klient-server. Server řídí přístup klientů a vyřizuje jejich příchozí požadavky.

Pro komunikaci mezi NFS serverem a klientem se využívá systém volání vzdálených procedur RPC [18]. Jedná se o synchronní komunikaci mezi dvěma procesy běžícími na vzdálených počítačích. Pro kódování zpráv se využívá standard XDR [10], který umožňuje přenos dat mezi různými počítačovými architekturami. Ke svému správnému fungování NFS využívá další programy pro zamykání souborů a monitorování stavu na straně serveru a také pro připojení k serveru.

NFS se postupně vylepšovalo a vyšlo několik verzí, nejnovější je verze NFSv4 [14] z roku 2003. Ta ještě dostala vylepšení uvedené v roce 2010 jako verze NFSv4.1 [15]. V dnešní době se lze stále setkat s využitím i starší verze NFSv3 [5]. Důvodem je, že oproti verzi NFSv4 je verze NFSv3 bezestavová. Neuchovává tedy historii požadavků, což usnadňuje implementaci, protože není zapotřebí zajišťovat zotavování po pádu systému. Od verze NFSv4 je tedy NFS stavové a došlo k navýšení výkonu a vylepšení zabezpečení.

4.3 Vytvořené modely

Pro vytvoření modelů jsem využil virtualizace systémů pomocí technologie hyper-v[1]. Díky virtualizaci si vystačíme pouze s jedním fyzickým strojem. Nevýhodou však je nižší výkon virtuálních strojů díky přerozdělení prostředků tohoto stroje. Nástroje pro virtualizaci systémů (hypervisory) jsou dvojího typu[2]. Rozdíl mezi těmito dvěma typy je v tom, že u

[1] https://hyperv.veeam.com/blog/what-is-hyper-v-technology/

[2] http://searchservervirtualization.techtarget.com/tip/Virtualization-hypervisor-comparison-Type-1-vs-Type-2-hypervisors

typu 1 hypervisor funguje přímo na hardwaru stroje. U typu 2 hypervisor funguje jako další aplikace na již nainstalovaném operačním systému. Hyper-v spadá do typu 1. Zástupcem typu 2 je například Oracle Virtualbox.

Použitým operačním systémem pro virtualizované stroje je Linux Lite 2.6[9] s verzí jádra Linux 3.13.0-76-generic (x86_64).

Hardware využitého notebooku Acer Aspire 5750G:

- Procesor[1] - Intel(R) Core(TM) i5-2430M CPU @ 2.40GHz (2 jádra, 4 logické procesory) Intel Virtualization Technology (VT-x), Intel VT-x with Extended Page Tables.
- Paměť - 2x Kingston 4GB 1333MHz RAM DDR3.
- Disk[2] - 750GB WDC WD7500BPVT 5400RPM 8Mb cache SATA 3Gb/s.

Přidělené zdroje každému virtuálnímu systému:

- Procesor - Přidělen 1 virtuální procesor.
- Paměť - Přidělena paměť o velikosti 2048MB.
- Disk - Využivá se virtuálního disku formátu vhdx o velikosti 80GB.

Virtuální stroje bylo potřeba vytvořit čtyři. Důvodem je, že krom stroje, kde je provozován webový portál a druhého, kde je provozován NFS server, je zapotřebí další stroj na generování požadavků. Ten už není zapotřebí vytvořit pro oba případy, ale vystačíme si s jedním. Poslední virtuální stroj bude využit pro zrcadlený server. Při práci s každým modelem budou zapnuty pouze dva jeho příslušné virtuální stroje a ostatní budou vypnuté.

Model webového portálu

Pro vytvoření modelu webového portálu jsou využity:

- server Apache - verze 2.4.7, APR 1.5.1-dev, APR-UTIL 1.5.3, Server MPM: prefork, forked.
- php5 - verze 5.5.9-1ubuntu4.14, PHP API: 20121113, PHP Extension: 20121212.
- databáze mysql - verze 5.5.46-0ubuntu0.14.04.2, innodb_version: 5.5.46.
- wikimedia[20]- verze 1.26.2

Kvůli celkovému výkonu systému a nárokům databáze na diskové operace by mohl být přidán jeden disk pouze pro její účely. Jelikož by se ale v našem případě jednalo pouze o další virtuální disk, na výkon by to mělo spíše negativní vliv. Důvodem je, že by se stále využívalo pouze jediného fyzického disku. Dalším a nejspíše i výhodnějším řešením by mohlo být pro databázi vyhradit místo samostatného disku další server. Opět by se ale v našem

[1] http://ark.intel.com/products/53450/Intel-Core-i5-2430M-Processor-3M-Cache-up-to-3_00-GHz
[2] http://www.wdc.com/wdproducts/library/SpecSheet/ENG/2879-701278.pdf

případě jednalo pouze o další virtualizovaný stroj, který bychom provozovali na jediném hostitelském počítači. Proto tedy budeme využívat pouze jednoho virtualizovaného stroje s jedním virtuálním diskem.

Na jediném disku tedy bude nainstalovaný operační systém včetně Apache, php5, mysql a také wikimedia. Také na něm bude uložen obsah databáze. Disk je naformátovaný souborovým systém ext4. Pro Apache byly ponechány defaultní porty. Tedy port 80 a pro ssl komunikaci port 443[1]. Zároveň musí být povolena příchozí a odchozí komunikace na těchto portech. To se dá zařídit například pomocí nástroje iptables[2]. Pomocí druhého virtuálního stroje, na kterém bude nainstalovaný nástroj JMeter (viz. podkapitola 2.7), bude probíhat generování požadavků.

Při generování požadavků se zaměříme na:

- Přístup k jednoduchým statickým stránkám.
- Vyhledávání dat v databázi.

Php skripty, které se budou při testování spouštět přes webové rozhraní, budou uloženy ve složce /var/www/html. Tyto skripty budou obsahovat dotazy pro vyhledávání v databázi a podobně. Podrobněji budou popsány při demonstraci testování na modelech v následující kapitole. Co se týče dat pro databázi, není potřeba je nějakým způsobem generovat, můžeme totiž využít volně dostupné databázové dumpy Wikipedie[3]. Nejprve musíme nainstalovat wikimedii, abychom mohli tento databázový dump naimportovat. Wikimedie bude nainstalována ve složce /var/www/html/mediawiki-1.26.2.

Model zrcadlení webového portálu

Pro vytvoření zrcadleného serveru jsou využity:

- nástroj rsync[4] - verze 3.1.0, protocol version 31.
- komunikační protokol ssh - verze OpenSSH_6.6.1p1, OpenSSL 1.0.1.
- plánovač úloh cron

Jako server, který bude zrcadlen, využijeme již vytvořený server pro model webového serveru Apache. Zrcadlený server se bude využívat jako záložní server. Synchronizaci dat mezi těmito dvěma servery bude spouštět záložní server pomocí nástroje rsync. Výhodou nástroje rsync je, že se při synchronizaci těchto dvou serverů budou přenášet pouze změněná data. Také je schopen rozpoznat soubory a adresáře, které byly na hlavním serveru smazány a vymaže je i na záložním serveru. Pro komunikaci při přenosu změněných dat se využívá ssh protokolu. Máme tedy zajištěný bezpečný datový přenos. Pro ssh máme více možností autentizace. Jelikož budeme chtít, aby synchronizace mezi těmito dvěma servery probíhala automaticky, využijeme možnost autentizace pomocí klíčů. Díky tomu nebude potřeba zadávat přihlašovací jméno a heslo při každém spuštění synchronizace. Pro zajištění

[1] Změnit nastavení portů pro Apache lze v souboru /etc/apache2/ports.conf
[2] http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-linuxovy-firewall-zaklady-iptables
[3] česká wikipedie ke stažení https://dumps.wikimedia.org/cswiki/20160407/
[4] Návod pro využití nástroje rsync http://www.tecmint.com/sync-two-apache-websites-using-rsync/

synchronizace v daných časových intervalech využijeme plánovač úloh cron. Sychronizaci budeme chtít provádět každých pět minut. Abychom toho docílili, je potřeba editovat soubor /etc/crontab.

Přidání záznamu do tabulky crontab záložního serveru:

- */5 * * * * user rsync -avze ssh root@ip_hlavniho_serveru:/var/www/html /var/www/

Model NFS serveru

Pro vytvoření modelu[1] NFS serveru jsou využity:

- NFSv3 a NFSv4
- soubor /etc/exports
- soubor /etc/fstab

Při vytváření NFS serveru využijeme disku naformátovaného souborovým systém ext4. Bude na něm nainstalovaný operační systém a nástroje pro NFS. Bude také obsahovat soubory a adresáře, které chceme sdílet na síti. V tomto modelu bude využita verze NFSv3 i NFSv4. Druhý virtuální stroj bude sloužit jako NFS client.

Aby se NFS client mohl připojit na náš NFS server, musíme mu to nejdříve povolit. To zařídíme editací souboru /etc/exports. Po editaci tohoto souboru bude potřeba restartovat službu pro NFS (nfs-kernel-server).

Povolení přístupu klienta k adresáři /sdileny v souboru /etc/export:

- /sdileny ip_klienta(rw,root_squash,sync)

Klientovi povolíme číst i zapisovat. Možnost "root_squash" zařídí, že připojeným root uživatelům nebudou poskytnuta root práva. Komunikace bude probíhat synchronně.

Nyní ještě zbývá vyřešit připojení klienta. K tomu se využívá nástroje mount. Máme tři možnosti, jak tento nástroj využít. První možností je zadat příkaz mount v terminálu. Nevýhodou tohoto řešení je, že se při restartu systému zruší jím vytvořené připojení. Další možností by bylo využití jednoho z automatic mount daemonů (například amd). Pro naše účely však postačí poslední možnost a tou je přidání záznamu do souboru /etc/fstab. Ten uchovává informace o všech připojených souborových systémech, včetně síťových disků, které budou připojeny po restartu systému.

Vložený záznam do souboru /etc/fstab (man. nfs(5)):

- ip_serveru:/sdileny /home/user/sitovydisk nfs4 timeo=15,rsize=8192,wsize=8192

Obsah adresáře /sdileny bude tedy dostupný na klientovi v adresáři /home/user/sitovydisk. Ten musí být na klientovi vytvořený. Možnost "nfs4" specifikuje využití verze NFSv4.

[1] https://help.ubuntu.com/community/SettingUpNFSHowTo

Timeout pro připojení je nastaven na 15 sekund. Možnosti ”wsize” a ”rsize” specifikují maximální velikost části dat, které si navzájem NFS server a klient posílají při požadavcích na čtení a zápis dat. Tím zařídíme změnu oproti defaultnímu chování, kdy jsou tyto hodnoty nastavené na hodnotu 262144. Je možné nastavit i další parametry. Pokud nejsou explicitně nastaveny, využije se defaultních hodnot. Jedná se například o paramer ”nointr”, který by zabránil přerušení diskových operací. Dále také parametr ”proto”, který specifikuje využití tcp nebo udp protokolu. Také parametry ”hard” a ”soft”, které specifikují chování pokud server neodpovídá (timeout). Při možnosti ”hard” se client bude snažit dokončit NFS operaci dokud server neodpoví. Při možnosti ”soft” využije omezeného počtu pokusů a poté při neúspěchu vrátí chybu. V našem případě jsou defaultně nastaveny možnosti povolení přesušení diskových operací (intr), tcp protokol (při využití verze NFSv4) a možnost ”hard”.

Pro generování požadavků klienta na zápis a čtení souboru na připojený disk využijeme nástroje iozone (viz. podkapitola 2.7). Použití tohoto nástroje bude demonstrováno v následující kapitole.

Kapitola 5

Demonstrace ladění výkonu a chyb na modelových situacích

V této kapitole bude demonstrováno využití uvedených nástrojů. Tyto nástroje budou používány při ladění výkonu a chyb na připravených modelech popsaných v kapitole 4. Pro každý model bude uvedeno více ukázek. Ty budou vždy podrobně popsány. Uvedené hodnoty vytížení procesoru, paměti, disku, atd. v ukázkách odpovídají pouze namodelovaným situacím v této bakalářské práci. Závisí na parametrech stroje, který je využíván pro virtualizaci, způsobu virtualizace a dalších parametrech.

5.1 Využití nástrojů na modelu webového serveru Apache

Ukázky pro webový server Apache budou obsahovat popis generování zátěže pomocí nástroje jmeter a popis prováděných úkonů pomocí uvedených nástrojů. Skripty test1.html a test2.php používané v ukázkách jsou k dispozici na přiřazeném CD. Na CD jsou obsaženy také vytvořené testovací plány Test1.jmx a Test2.jmx pro nástroj JMeter.

Generování požadavků

Generování zátěže pomocí nástroje JMeter bude probíhat následovně. Nejdříve je nutné vytvořit nový testovací plán. Pomocí pravého tlačítka myši na "Test Plan", který se nachází v levém panelu, se zvolí "Add ->Threads (Users) ->Thread Group. V "Thread Properties" nastavíme "Number of Threads (users)". Tato hodnota nám určuje počet simulovaných uživatelů. "Ramp-Up Period" určí, za jakou dobu budou spuštěni všichni simulovaní uživatelé. Například kdybychom chtěli simulovat 10 uživatelů a hodnota "Ramp-Up Period" bude 100, tak se každý další uživatel spustí po 10s. "Loop Count" znamená počet opakování požadavku každým uživatelem. Dále pomocí pravého tlačítka myši na naši vytvořenou Thread Group přidáme "Add ->Sampler ->HTTP Request". V "Server Name or Ip" nastavíme ip adresu našeho stroje s Apache serverem. A nakonec v "HTTP Request ->Path" nastavíme cestu k našemu skriptu. Poté si tento testovací plán můžeme uložit. Nástroj JMeter obsahuje spoustu dalších možností, ale pro generování našich požadavků postačí pouze toto nastavení.

Ukázka 1

V této ukázce budeme generovat zátěž pomocí požadavků na zobrazení statické stránky. Cestu ke skriptu, který budeme využívat nastavíme na /test1.html. V tomto skriptu se nachází pouze výpis nadpisu v tagu h1 s textem "ahoj svete!". Pro generování zátěže nastavíme počet simulovaných uživatelů na 5 a budou se postupně spouštět v rozmezí 20s (hodnota Ramp-Up Period bude 100). Počet opakování požadavků nastavíme na "Forever". Simulovaná zátěž v této ukázce bude minimální. Budeme se věnovat analýze příchozích požadavků a zjišťování, jaké operace jsou v systému prováděny za účelem jejich obsluhy.

Ještě před spuštěním generování zátěže se pomocí ssh připojíme na náš Apache server. Pomocí příkazu *ps ax /grep "apache2"* se nám vypíšou všechny existující procesy Apache. Jejich status bude začínat na *S*, to znamená uspaný, jelikož v tuto chvíli ještě nevysíláme žádné požadavky. Jeden z těchto procesů bude mít status *Ss*. Takto rozeznáme hlavní proces. Důvod, proč je Apache procesů více najdeme pomocí příkazu *apachectl -V*, který nám zobrazí informace o nainstalovaném apachi. V tomto výpisu najdeme nastavení "Server MPM: prefork[4], threaded: no, forked: yes (variable process count)". Při obsluze příchozích požadavků se tedy vytvářejí kopie procesu Apache, které tyto příchozí požadavky obsluhují. Dále spustíme nástroj top, abychom mohli průběžně sledovat vytížení Apache serveru.

Nyní spustíme generování zátěže. V popředí výpisu nástroje top se okamžitě objevily procesy Apache. Jejich status se průběžně mění mezi *S* a *R* (běžící proces). S každým dalším simulovaným uživatelem, který začne vysílat požadavky, také vzrůstá vytížení procesoru těmito procesy a také počet těchto procesů. V době, kdy byl aktivní pouze jeden simulovaný uživatel se vytížení procesoru každým Apache procesem pohybovalo od 1,7 % do 4,0 %. V případě, kdy už je aktivních všech pět simulovaných uživatelů se toto vytížení pohybuje od 4,9 % do 8,3 %. Celkové vytížení procesoru se v tomto případě skládá z 6,7 % až 9,2 % z času stráveného v uživatelském prostoru, 30 % až 33 % z času stráveného v jádře, 0,3 % až 1,4 % z čekání na vstupní a výstupní operace disku a z 25,3 % až 33 % z softwarového přerušení. Zbytek času procesor stráví v nečinném stavu (idle).

Také si lze ve výpisu nástroje top všimnout vytížení paměti, které není vysoké a také toho, že je dostatek volné paměti. Každý proces Apache využívá pouze 0,1 % paměti v obou případech, ať už je aktivní pouze jeden uživatel nebo všech pět. Nástroj top nám ale neposkytuje informace o vytížení disku. Využijeme tedy nástroje iotop. Můžeme si všimnout, že procesy Apache vytěžují disk požadavky na zápis dat. Hodnota rychlosti zápisu dat každým Apache procesem se pro jednoho uživatele pohybovala od 3,5 kB/s do 11,2 kB/s a celkový zápis na disku okolo 44 kB/s. Pro všech 5 uživatelů se rychlost zápisu dat každým Apache procesem pohybovala od 3,5 kB/s do 20,4 kB/s. Celkový zápis na disku byl od 95,4 kB/s do 109,6 kB/s. Důvodem těchto zápisů dat na disk je například využívání logů. Ty se pro Apache nelézají v umístění /var/log/apache2. Zaznamenali jsme tedy nárůst zátěže pro procesor a disk, který roste s počtem simulovaných uživatelů.

Nyní se pokusíme zjistit nějaké informace o těchto příchozích požadavcích. IP adresu, ze které jsou vysílány požadavky na Apache server můžeme zjistit buď nástrojem netstat, iptraf nebo tcpdump. Nástroj netstat je nejméně invazivní variantou, nicméně poskytuje nejméně informací o přenosu z těchto tří nástrojů. Jelikož chceme IP adresu připojeného stroje, zadáme příkaz *netstat -n*. Ve sloupci " Foreign Address" požadovanou IP adresu najdeme.

U nástroje iptraf po spuštení vybereme možnost "IP traffic monitor" a monitorovat budeme chtít rozhraní eth0. Nyní vidíme všechny probíhající komunikace. IP adresy stroje, který výsílá i stroje, který přijímá pakety jsou zobrazovány ve tvaru *HOST:PORT*, přičemž

IP adresa s portem 80 je adresou našeho Apache serveru. Druhá IP adresa je adresou stroje, který vysílá požadavky. Také vidíme komunikaci ssh, přes kterou jsme připojení na Apache server (port 22).

Pokud bychom neměli nástroj iptraf k dispozici, pomocí nástroje tcpdump bychom IP adresu stroje vysílajícího požadavky zjistili následovně. Jelikož nástroj tcpdump vypisuje zachycené pakety do terminálu, objevuje se velké množství vypsaných údajů, ze kterých je složité něco v rychlosti zjistit. Proto je lepší jeho výpisy ukládat do souboru místo vypisování do terminálu a následovně informace analyzovat z tohoto souboru. To zajistíme příkazem *tcpdump -n -i eth0 -w tcpdumpOutput*. Po několika sekundách můžeme zachycování paketů přerušit a začít analyzovat zachycené pakety z vytvořeného souboru pomocí příkazu *tcpdump -n -r tcpdumpOutput port 80*. Tím také zajistíme zobrazení komunikace pouze pro komunikaci, kde se vyskytuje port 80. IP adresy strojů, které se podílejí na komunikaci jsou zobrazeny ve tvaru *IP.port >IP.port* včetně času zachycení a dalších informací. Pokud bychom nepoužili parametr *-n*, místo IP adres bychom zjistili jména těchto strojů a místo portu typ protokolu (http, ssh, ...).

Nástroje iptraf a tcpdump, narozdíl od nástroje netstat, poskytují statistiky přenesených dat a další informace. Pokud potřebujeme pouze zjistit IP adresu a nepotřebujeme tyto statistiky, je nejvýhodnější využít nástroje netstat kvůli nižší invazivnosti.

Nyní se pokusíme zjistit, jaké úkony jsou v systému prováděny při obsluze těchto generovaných požadavků. Využijeme k tomu nástroj strace, ten vyžaduje jako parametr číslo pid daného procesu. Zvolíme si jeden z Apache procesů a pomocí příkazu *strace -p pid -o straceOutput* spustíme jeho sledování. Na začátku souboru s výsledkem sledování najdeme informaci o číslu pid procesu, který byl sledován. Poté již vidíme samotné operace. Jako první je operace "accept4". Jedná se o systémové volání, kterým přijmeme síťové připojení. Poté a na více místech se objevuje systémové volání "gettimeofday" pro získávání času. Dále systémové volání "getsockname", sloužící pro získání IP adresy, na kterou je vázaný socket. Dvojce systémových volání "fcntl" s parametry "F_GETFl" a "F_SETFL" slouží pro přečtení a nastavení file descriptoru a následující systémové volání "read" pro přečtění dat. V těle read vidíme příchozí požadavek začínající "GET /test1.html", což je požadavek, který generujeme jako žátěž. Následně jsou provedeny systémové volání "stat" a "open", nejprve se tedy zjistí status souboru (například jeho přístupová práva) a poté se otevře. Následuje systémové volání "mmap", které vytvoří nové paměťové mapování v prostoru virtuálních adres volajícího procesu. Následuje systémové volání "writev", které na socket zapíše výsledek zpracování požadavku. V jeho těle vidíme část "HTTP/1.1 200 OK", což znamená úspěch a také vidíme obsah souboru /test1.html. Klient, který poslal požadavek tedy obdržel kladnou odpověď a také obdržel požadovaná data. Poté se uvolní vytvořené paměťové mapování pomocí systémového volání "munmap". Dále proběhne systémové volání "write", které obsahuje ip klienta, který poslal požadavek. Nakonec proběhne systémové volání "times", které poskytne hodnotu počtu provedených cyklů procesoru a obsluha požadavku se následně ukončí systémovým voláním "close". Jelikož se naše generované požadavky opakují, dokud je nezastavíme, tato systémová volání probíhají opakovaně. V dalších požadavcích se však již neotevírá socket, ale využívá se systémového volání "poll", díky kterému se čeká, dokud nebude socket volný pro využívání. Neukončuje se tedy v tomto případě připojení s klientem, pouze se vyřižují jeho další požadavky. Pokud proces nemá ve frontě další požadavky k obsluze, proběhne systémové volání "shutdown". Na konci souboru se vyskytuje řádek "<detached ...>", to znamená konec sledování nástrojem strace.

Nástroj strace vypisoval všechny provedené systémové volání při obsluze našeho požadavku k přístupu ke skriptu /test1.html. Musel je tedy všechny sledovat a tím zpomalil jejich

vykonávání. Jak již bylo uvedeno, tento nástroj vykazuje vysokou invazivnost. Nicméně, díky němu jsme se mohli dozvědět o všech provedených systémových voláních, což nám může pomoci při hledání problému.

Ukázka 2

V této ukázce budeme generovat zátěž stejným způsobem jako v ukázce 1. Navýšíme však počet simulovaných uživatelů na 50, spustíme je všechny okamžitě ("Ramp-Up Period" nastavíme na 1) a počet opakování necháme na "Forever". Cesta ke skriptu v požadavku bude nastavena na /test2.php. Tento skript již v sobě obsahuje jednoduchou komunikaci s databází a je napsán ve skriptovacím jazyku php. Tento skript nejprve připojí soubor, který slouží pro připojení k mysql databázi, poté provede výběrový dotaz *SELECT * FROM page* a vypíše celý sloupec *page_title* z výsledku poslaného z databáze. Můžeme tedy očekávat, že se tentokrát bude na vytížení systému podílet krom samotného Apache také mysql databáze. Tentokrát však již nebudeme pouze zkoumat příchozí požadavky, ale vžijeme se do role administrátora, který musí řešit neočekávaný problém. Představte si, že Vás ráno vzbudí vyzvánění na mobilu a volá Vám šéf, že si nějaký klient zrovna stěžoval, že jeho server "nestíhá" a "nedá se s webovou stránkou vůbec pracovat" a vy to musíte vyřešit. Tak tedy vstanete, uděláte si kávu a začnete hledat problém.

První kroky se nebudou od ukázky 1 lišit. Musíme se nejprve připojit na náš Apache server pomocí ssh. Jelikož nevíme, co se na serveru děje, zkusíme nejprve nástroj top, abychom to zjistili. Se zděšením zjistíme, že proces databáze vytěžuje procesor až na 85 %. Vidíme také aktivní procesy Apache a také, že je využita téměř celá kapacita paměti.

Nejprve zjistíme, jestli opravdu probíhá nějaká komunikace klientů s naším Apache serverem. Využít k tomu můžeme nástroj iptraf. V jeho menu vybereme funkci "IP traffic monitor". Zjistíme, že počet komunikací je 51. Jedna z nich je naše ssh připojení (port 22) a zbytek požadavky na port 80, tudíž na náš Apache. Také vidíme, že všechny požadavky na Apache server přicházejí ze stejného zdroje. Tuto skutečnost budeme v tomto případě ignorovat, jelikož se jedná o naši generovanou zátěž, u které nemůžeme zařídit, aby každý požadavek putoval z odlišných IP adres. Budeme to tedy brát tak, že tyto požadavky jsou regulérní a přicházejí z různých zdrojů. Náš server tedy opravdu obsluhuje nějaké požadavky.

Pomocí příkazu *free -h* zjistíme, že krom dat v paměti jsou nějaká data také ve swapu. Tyto informace můžeme zjistit také ve výpisu souboru /proc/meminfo. Pokusíme se tedy zjistit, jaké procesy využívají swap. Pomocí příkazu *swapon -s* zjistíme, že je swap namountován jako zařízení /dev/sda5. Využijeme tedy nástroje blktrace pomocí příkazu *blktrace -d /dev/sda5 -o - |blkparse -i - >blktraceOutput*. Každý řádek ve výpisu obsahuje 8 sloupců. První sloupec obsahuje major a minor číslo zařízení. Druhý sloupec obsahuje číslo procesoru. Třetí sloupec je pořadí operace. Ve čtvrtém sloupci nalezneme časové razítko (time stamp). V pátém sloupci nalezneme číslo pid procesu. V šestém sloupci typ prováděné události. Sedmý sloupec značí, jestli se jedná o operaci čtení (R), zápisu (W), vyřazení (D) nebo ani jednu z těchto tří možností (N). V posledním, osmém sloupci nalezneme v hranatých závorkách název procesu. Zjistíme tedy, že swap využívají procesy Apache, které odlišíme podle čísla pid a také proces databáze. Jejich operace jsou buď typu čtení nebo typu N. Na konci výpisu také vidíme souhrné statistiky počtu čtení a zápisu.

Abychom zjistili, co přesně tyto procesy Apache a databáze vykonávají, využijeme nástroj strace. Nejprve ho použijeme na proces Apache. Využijeme ten, který byl jako první v našem výpisu nástroje blktrace. Zadáme tedy příkaz *strace -p pid -o straceOutput*, pomocí kterého zjistíme, že oproti použití tohoto nástroje v ukáze 1, probíhá velké množství sys-

témových volání ”read” a ”writev”. I přes mnoho znaků, které jsou zapsané v escapované podobě můžeme rozeznat, že se jedná o smysluplné informace, získané pravděpodobně z databáze (např. ”Amazonsk\303\251_de\305\241tn\303\251_pralesy</br>”). Také odhalíme, že se volaný skript v požadavku nachází v umístění /var/www/html/test2.php. Dále nástroj strace spustíme na proces databáze. Ve výpisech objevíme systémové volání ”futex”, které se u procesů Apache nevyskytovalo. Jedná se o systémové volání pro synchronizaci vláken procesu. Databáze narozdíl od Apache v našem případě tedy pracuje s vlákny procesu. Také vidíme, že databáze pracuje se svým socketem, nacházejícím se v umístění ”/var/run/mysqld/mysqld.sock” (”getsockname(106, sa_family=AF_LOCAL, sun_path=/var/run/mysqld/mysqld.sock”).

Když databáze využívá tento soubor, pravděpodobně může využívat i další soubory. To zjistíme pomocí nástroje lsof. Zadáme tedy příkaz *lsof |grep ”mysql”*. Na začátku výpisu toho nástroje nalezneme informaci, že proces databáze využívá adresáře /var/lib/mysql (ve čtvrtém sloupci ”cwd”) a kořenového adresáře (ve čtvrtém sloupci ”rtd”). Také, že binární kód mysql nalezneme v souboru /usr/sbin/mysqld (ve čtvrtém sloupci ”txt”). Adresář /var/lib/mysql využívá pro uložení potřebných dat. Nalezneme v něm další adresáře, které korespondují s názvy databází. Ve výpisu nástroje lsof jsme nalezli, že využívá soubory ”searchindex.MYI” a ”searchindex.MYD” ve složce /var/lib/mysql/enWiki, což je databáze pro naši wikipedii. Soubor ”searchindex.MYI” obsahuje indexy našich tabulek a soubor ”searchindex.MYD” obsahuje data. V adresáři /var/lib/mysql/enWiki také nalezneme soubory s příponou ”.frm”, ve kterých jsou uloženy definice tabulek této konkrétní databáze. Přenášená data v odpovědích na požadavky na Apache server tedy pocházejí z této databáze. Také si můžeme v druhém sloupci výpisu nástroje lsof všimnout, že nejprve jsou tyto soubory přiřazeny k pid procesu databáze a poté se se objevují výpisy stejných souborů, ale k číslu pid přibude další číslo (tid), které značí identifikaci vlákna procesu databáze. Na konci našeho výpisu nástroje lsof také nalezneme procesy Apache, protože využívají soubory dynamických knihoven souvisejících s mysql. Konkrétně se jedná o dynamické knihovny ”mysqli.so”, ”pdo_mysql.so” a ”mysql.so” z adresáře ”/usr/lib/php5/” a ”libmysqlclient.so.18.0.0” z adresáře ”/usr/lib/x86_64-linux-gnu”. U těchto souborů dynamických knihoven nalezneme ve čtvrtém sloupci výpisu informaci ”mem”, to znamená, že soubory jsou namapovány do paměti.

Zjistili jsme tedy, že swap využívají procesy Apache a proces databáze. Také jaký skript je volán v požadavcích na náš Apache server a jaká data se přenáší jako odpověď na tyto požadavky klienta. A zjistili jsme také, jaká databáze se využivá, aniž bychom zkoumali zdrojový kód volaného skriptu. Pro zefektivnění práce s databází by mohlo pomoci upravení jejího nastavení. Toto nastavení se nachází v konfiguračním souboru ”my.cnf”, který je umístěn ve složce ”/etc/mysql”. Uvažovat bychom mohli například o navýšení ”query_cache_limit”, ”query_cache_size” a ”key_buffer”[13]. Další možností je také úprava indexace této databáze.

5.2 Využití nástrojů na modelu zrcadleného serveru

Model zrcadleného serveru využijeme pro demonstraci nástrojů pro analýzu systému.

perf

Pomocí příkazu *perf list* zjistíme, jaké události můžeme na našem serveru sledovat. Rozděleny jsou podle typu události do skupin "Software event", "Hardware event" a "Tracepoint event". Nejprve zapoznámkujeme záznam pro příkaz rsync v souboru /etc/crontab a smažeme obsah adresáře /var/www na zrcadleném serveru. Důvodem je, že budeme nástroj rsync spouštět v terminálu společně s nástrojem perf. Vymazáním obsahu adresáře /var/www docílíme toho, že se bude muset přenést celý obsah adresáře /var/www/html z Apache serveru.

Nyní se pokusíme zjistit, jaké informace získáme od nástroje perf bez specifikování událostí ke sledování při spuštění nástroj rsync. Docílíme toho příkazem *perf stat rsync -avze ssh user@ip_apache_serveru:/var/www/html /var/www*. Ve výpisu výsledku vidíme informace o celkovém počtu přenesených dat a průměrné rychlosti přenosu. Dále vidíme hodnoty počtu "task-clock", přepnutí kontextu a výpadku stránek. Poté ještě celkový čas (17s), za který byl příkaz rsync dokončen.

Počet jednotlivých systémových volání můžeme zjistit příkazem *perf stat -e syscalls:* rsync -avze ssh user@ip_apache_serveru:/var/www/html /var/www*. Nejčastějším systémovým voláním bylo volání pro zjištení času. Poté systémová volání související se zápisem přenášených dat na disk. Celkový čas vykonávání se zvýšil na 44s, což je nárůst o 27s oproti předchozímu využití.

Důsledkem spouštění nástroje rsync se sledováním pomocí nástroje perf bylo značné zpomalení jeho provádění. Celkový čas provádění se odvijí podle událostí, které sledujeme.

oprofile

Nástroj oprofile sbírá informace na pozadí. Tyto informace (vzorky) ukládá do souboru /var/lib/oprofile/samples. K zobrazení těchto informací slouží příkaz *opreport*. Narozdíl od nástroje perf, tento nástroj poskytuje informace o přerušení časovačů jednotlivými procesy. V sloupci "samples" také vidíme počet jejich výskytů. Také jsou zobrazeny dynamické knihovny, které procesy využívají. Pokud bychom chtěli sledovat pouze využití dynamických knihoven jedno konkrétního procesu, můžeme také využít nástroje ltrace.

Díky tomuto nástroji můžeme odhalit, které procesy vytěžují náš server nejvíce v dlouhodobém horizontu. Narozdíl od jiných nástrojů, které nám zobrazují pouze aktuální vytížení.

systemtap

Prostřednictvým tohoto nástroje můžeme také sledovat různé části systému. Můžeme například sledovat veškerá systémová volání v daném časovém intervalu. Příslušný vytvořený skript (syscall.stp), který k tomuto účelu slouží, spustíme příkazem *stap syscall.stp*. Je v něm nastavený časový interval 4s, po kterém se provádění ukončí. Zároveň spustíme příkaz *rsync -avze ssh user@ip_apache_serveru:/var/www/html /var/www*. Následně se vypíší všechna zachycená systémová volání. V nich zjistíme PID procesů, které tato systémová volání využívaly. Je v nich obsažen také nástroj rsync. V jeho systémových voláních také vidíme přenášené soubory.

5.3 Využití nástrojů na modelu NFS serveru

Ukázky pro NFS server budou obsahovat popis generování zátěže pomocí nástroje iozone a popis prováděných úkonů pomocí uvedených nástrojů.

Generování požadavků

Pro generování požadavků nástrojem iozone můžeme využít různé typy testů. Je jich celkem 13 (test 0 - 12). Například test typu 0 generuje pouze požadavky na zápis a přepsání. Co obsahují tyto typy testů je popsáno v manuálových stránkách iozone(1). U každé ukázky bude uvedeno, který typ testu využíváme. Můžeme také například specifikovat velikost generovaných souborů a také po jak velkých částech se budou generovat. Můžeme tedy například nastavit zápis generovaných souborů o velikosti 8MB na disk a systémová volání, která tento zápis vykonávají, budou tento soubor zapisovat po 1MB. Proběhlo by tedy 8 systémových volání pro zápis. Tyto generované soubory jsou pouze dočasné. Nástroj iozone budeme spouštět v adresáři našeho sdíleného umístění. Tím zajístíme generovanou zátěž pro náš NFS server.

Ukázka 1

V této ukázce budeme využívat NFSv4. Připojení klienta je provedeno s parametry uvedenými v Modelu NFS serveru v podkapitole 4.3. Nástroj iozone nejprve spustíme v automatickém módu, kdy bude generovat náhodné diskové operace s náhodně velkými soubory (od 64kB do 512MB) a náhodně velkými částmi k zápisu (od 4KB do 16MB). To zařídíme příkazem *iozone -a*. Tento příkaz musíme zadat v umístění našeho připojeného síťového disku. Pokud bychom se chtěli ujistit, že nástroj iozone opravdu něco v našem sdíleném umístění generuje, ve výpisu obsahu tohoto adresáře musíme vidět soubor "iozone.tmp". To platí jak pro stranu klienta, tak pro stranu serveru.

Spustíme tedy generování zátěže a připojíme se pomocí ssh na náš NFS server. Ve výpisu nástroje top uvidíme procesy od NFS (nfsd). Jejich defaultní počet je 8 procesů. Využití procesoru jednotlivými procesy od NFS se pohybuje od 2,6 % do 14,2 %. Celkové vytížení procesoru se skládá z 11 % až 18 % z času stráveného v jádře, 15,8 % až 45 % z čekání na vstupní a výstupní operace disku a z 30 % až 60 % z softwarového přerušení. Zbytek času procesor stráví v nečinném stavu (idle). Tyto hodnoty se průběžně mění podle typu operací a souborů, které zrovna nástroj iozone generuje. Využití paměti procesy NFS je na 0,0 %. Když však porovnáme výpis nástroje free před spuštěním nástroje iozone a po něm, zjistíme, že se změnila hodnota množství dat v bufferech a vyrovnávací paměti cache a také se snížila kapacita celkové volné paměti. Sice jde o malou změnu v rámci několika desítek kB, znamená to ale, že nějaké operace s pamětí probíhaly. Využití procesoru je tedy v pořádku i přes vyšší zatížení. Využití paměti je minimální.

Ke zkontrolování činnosti disku využijeme nástroj iotop. V jeho výpisu také uvidíme procesy NFS. Jejich hodnoty rychlosti zápisu dat na disk míří až k 850 kB/s. Hodnoty "TOTAL DISK WRITE" i "ACTUAL DISK WRITE" mířily až k 2000 kB/s. Můžeme také vidět proces s názvem "jbd2/sda1-8", což je proces žurnálu pro souborový systém ext4. Jeho hodnota celkového zápisu a čtení (IO) se pohybuje od 0 % do 67 % podle typu operací a souborů, které zrovna iozone generuje. Prováděné operace na disku budeme sledovat nástrojem blktrace. Nejprve však musíme zjistit, které zařízení je náš disk. To zjistíme nástrojem df. V našem případě to bude zařízení /dev/sda1. Spustíme tedy nástroj blktrace

příkazem *blktrace -d /dev/sda1 -o - |blkparse -i - >blktraceOutput*. Ve výsledném výpisu uvidíme spoustu operací jak od NFS, tak od žurnálu.

Činnost procesu NFS na disku se skládala ze sledu akcí "AQGPIUDC". Nejdříve tedy proběhne přemapování na zařízení, které bude vstupní a výstupní požadavky obsluhovat (A). Poté se požadavek zařadí do fronty (Q), není však v tuto chvíli ještě plně vytvořen. Z této fronty se následně vyjme (G) a vykoná se akce "plug" (P), po které už je požadavek odeslán plánovači pro přidání do vnitřní fronty k pozdějšímu obsloužení (I), v tuto chvíli už je požadavek plně vytvořen. Následně se vykoná akce "unplug" (U), která začne odesílat požadavek ovladači zařízení. Poté následuje informace o dokončení odesílání požadavku ovladači zařízení (D). Informace o dokončení obsluhy požadavku (C) však již není přiřazena k procesu "nfsd" ale k procesu s číslem pid 0. Jedná se o proces s názvem "swapper/0". Ten se využije jako prázdná operace, kterou se konec obsluhy požadavku značí.

Činnost žurnálu se od činnosti NFS liší. Sled jeho akcí je "AQGP AQMIUDC AQGI DCC". První čtyři operace jsou stejné. Poté však následuje úsek "AQM", který se opakuje vícekrát. Akce "back merge" (M) slouží ke sloučení požadavků, pokud předchozí požadavek končí tam, kde další požadavek začíná, což nastává například při zápisu jednoho souboru po menších částech. Následně se vykoná sled akcí "IUDC" stejně jako u činnosti NFS. Poté ještě proběhnou akce "AQGI DCC". Tentokrát je k procesu "swapper/0" přiřazena i informace o dokončení odesílání požadavku ovladači zařízení (D).

K analýze komunikace NFS serveru a NFS klienta využijeme nástroje tcpdump. Zadáme tedy příkaz *tcpdump -n -i eth0 -w tcpdumpOutput*. Příkazem *tcpdump -nvvX -r tcpdumpOutput* poté budeme číst výsledky. Ve výpisu výsledků vidíme IP adresu ze které požadavky přicházejí a také z jakého portu. U IP adresy našeho serveru vidíme port 2049, což je defaultní port pro NFS server. Komunikace probíhá pomocí protokolu TCP. Velikost odeslaných i přijatých paketů se pohybovala od 52 B do 1500 B. K fragmentaci paketů nedocházelo. Celkovou rychlost přenosu dat po síti zjistíme nástrojem iptraf. Její hodnota se pohybovala okolo 18255,5 kbits/sec, z toho pouze 780 kbits/sec byly odchozí pakety.

Díky příchozím požadavkům od NFS klienta jsme zaznamenali nárůst počtu diskových operací a také přenášených dat po síti. Nicméně, náš server tyto požadavky zvládal obsloužit a zvládl by i vyšší zátěž.

Ukázka 2

V této ukázce budeme využívat postupy uvedené v ukázce 1. Změníme ale způsob generování požadavků. Generovat budeme pouze požadavky na zápis souborů. Velikost těchto souborů nastavíme na 500MB a velikost zápisu 2MB. Příkaz pro generování bude tedy *iozone -i 0 -s 512000 -r 2048*. Tentokrát budeme v roli administrátora, který si všimne, že načítání obsahu sdíleného adresáře probíhá neúměrně dlouhou dobu a bude se snažit zjistit, proč se tak děje.

Po připojení na náš NFS server pomocí ssh využijeme nástroje top. V jeho výpisech uvidíme aktivní procesy NFS (nfsd). Hodnoty využití paměti jsou nízké. Procesor tráví přibližně 25 % času v jádře, 60 % obsluhou softwarových přerušení a pouze do 2 % čekáním na vstupní a výstupní operace disku. Pomocí nástroje iotop zjistíme, že probíhá zápis dat procesy NFS. Hodnota celkového zápisu se pohybovala až k 3 MB/s. Hodnota zápisu procesu žurnálu je ale narozdíl hodnoty v ukázce 1 nízká, pouze do 6 % z celkového zápisu dat. Jelikož procesor netráví mnoho času čekáním na vstupní a výstupní operace disku a žurnál se také tolik nevyužívá, mnoho souborů se zřejmě nezapisuje.

Ke zjištění probíhajících diskových operací využijeme nástroje blktrace pomocí příkazu

blktrace -d /dev/sda1 -o - |blkparse -i - >blktraceOutput. Z výpisu tohoto nástroje zjistíme, že probíhá velké množství operací od NFS procesů (oproti ukázce 1) než proběhnou operace žurnálu. Konkrétně se jedná o operace zápisu dat. Operace mají stejný sled jako v ukázce 1 ("AQGPIUDC"), ale sekvence operací "AQG" probíhají vícekrát za sebou. Jedná se o práci s frontou požadavků. Stejně tak operace vložení do vnitřní fronty pro pozdější obsluhu ovladačem zařízení (I) a operace odeslání požadavku ovladači zařízení (D) probíhá vícekrát za sebou. Až poté se objeví více operací dokončení (C). Můžeme vidět, že tyto operace dokončení zápisu na sebe podle čísel bloků navazují. Navazující bloky jsou požadovány po 1024 (např. 14534656 + 1024 [0]). Můžeme se tedy domnívat, že se zapisuje jeden velký soubor.

O jaký soubor se jedná se pokusíme zjistit pomocí analýzy komunikace NFS serveru s NFS klientem. K tomu využijeme nástroje tcpdump. Zadáme tedy příkaz *tcpdump -n -i eth0 -w tcpdumpOutput2*. Příkazem *tcpdump -nvvX -r tcpdumpOutput2* poté budeme číst výsledky (vytvořený soubor tcpdumpOutput2 můžeme otevřít v nástroji Wireshark, který nám poskytne přehlednější výpis výsledku). Zjistíme tedy, že jako první poslal požadavek NFS klient na náš NFS server. Jednalo se o požadavek na otevření souboru iozone.tmp. Následovaly další požadavky, které vedly k úspěšnému otevření tohoto souboru. Požadavky na zápis dat probíhaly tak, že nejprve poslal NFS klient požadavek na zápis dat o velikosti 8192B, což odpovídá naší hodnotě "wsize" pro připojení NFS klienta. Server mu vrátil kladnou odpověď na tento požadavek. Obě tyto části komunikace probíhaly pomocí protokolu NFS. Poté NFS klient pošle určtěná data k zápisu pomocí protokolu TCP. Následně NFS server odpoví, že tyto pakety přijal (ACK). Tato komunikace pro zápis dat se opakuje, dokud NFS klient chce nějaká data zapisovat. Velikost těchto TCP paketů nepřekročí 1514B. To souvisí s hodnotou MTU na straně NFS klienta. Tato hodnota určuje maximální velikost paketu pro odesílání.

Zjistili jsme tedy, že se v našem sdíleném umístění zapisuje soubor iozone.tmp, a že tento zápis stále probíhá, což je důvod pomalé odezvy. Tento přenos samozřejmě můžeme nechat dokončit nebo ho můžeme přerušit či zakázat danému NFS klientovi přístup. Také je ale možnost nějakým způsobem optimalizovat tuto komunikaci mezi NFS serverem a NFS klientem. Nabízí se zredukovat režii, která při jejich komunikaci vzniká, nebo zredukovat velikost přenášených souborů. Velikost přenášených souborů můžeme zredukovat kompresí těchto souborů před jejich přenosem (při reálném nasazení, ne při využívání nástroje iozone). Režii při této komunikaci na síti můžeme zredukovat zvětšením hodnoty "wsize". Tím se zredukuje počet požadavků na zápis dat, jelikož každý požadavek bude na zápis většího množství dat. Počet a velikost TCP paketů s daty to neovlivní.

Ukázka 3

V této ukázce si ukážeme některé rozdíly při využití verze NFSv3. Upravíme připojení klienta v modelu NFS serveru v podkapitole 4.3 tak, aby využíval NFSv3.

Upravený záznam v souboru /etc/fstab pro NFSv3:

- ip_serveru:/sdileny /home/user/sitovydisk nfs vers=3,proto=udp,timeo=15, rsize=8192,wsize=8192

Nástroj iozone spustíme stejně jako v případě ukázky 2 příkazem *iozone -i 0 -s 512000 -r 2048*. Ve výpisu nástroje top uvidíme stejný počet procesů NFS (nfsd) jako v ukázce

2. Využití procesoru se liší v hodnotě času stráveného obsluhou softwarových přerušení, je o 10 % nižší. Využití paměti je na stejné úrovni jako v případě použití NFSv4. Neliší se ani diskové operace zjištěné pomocí nástroje blktrace. Rozdíl najdeme pomocí nástroje tcpdump v síťové komunikaci.

Všimnout si můžeme záměny protokolu tcp za protokol udp u paketů odesílaných klientem. Tím se značně zjednodušila komunikace mezi klientem a serverem, jelikož nedochází k potvrzování přijetí paketů s daty k zápisu (ACK). Po otevření souboru k zápisu dat už následují pouze požadavky na zápis dat, odpověd serveru, že klient může začít posílat tato data a posílání samotných dat. Pomocí nástroje iptraf jsme zjistili, že se liší také hodnota celkové rychlosti přenosu dat. U NFSv3 došlo k navýšení o 4000 kB/s.

5.4 Využití nástrojů na stroji generujícím požadavky

Ukázky využití nástrojů na stroji generujícím požadavky jsem zařadil pro demonstraci využití nástrojů pro monitorování systému. Zařazen byl i nástroj ntop, který nebyl využitý v předchozích ukázkách.

Conky

Spustíme nástroj conky a zobrazíme si naši plochu. Zobrazují se nám informace o aktuálním využití procesoru, paměti, swapu, počtu procesů, využití kapacity disku, statistiky sítě, uptime serveru a využití procesoru a paměti jednotlivými procesy. Můžeme tedy pohodlně sledovat využití našeho stroje i při práci na něčem jiném. Otázka u tohoto nástroje a nástrojů jemu podobných však je, jestli toto opravdu chceme a potřebujeme. Přímo ve výpisech nástroje conky vidíme, že tento nástroj spotřebuje 1 % až 2 % procesoru a také 0,4 % až 3 % paměti. Znamená to tedy, že jenom proto abychom viděli nějaké údaje, které většinou potřebujeme vidět až nastane nějaký problém, spotřebujeme část výkonu našeho serveru neužitečnými operacemi. Pokud tento nástroj opravdu využít chceme, můžeme aspoň zmírnit jeho dopady na výkon serveru tím, že zpomalíme interval výpisu informací. Toho docílíme buď spuštěním s parametrem u, například *conky -u 5*. Místo každou sekundu (což je defaultní hodnota) se budou informace vypisovat co 5 sekund. Druhou možností je tuto hodnotu změnit v config souboru. Ten se nachází ve složce /etc/conky a jeho název je conky.conf. Interval výpisu změníme úpravou hodnoty "update_interval".

Cacti

Instalace nástroje cacti je složitější a je vhodné vyhledat nějaký návod[1]. Na našem serveru je tento nástroj přístupný a funkční na adrese http://localhost/cacti/.

Po přístupu na tuto adresu jsme nejprve vyzváni k zadání přihlašovacích údajů. Po úspěšném přihlášení se nám zpřístupní console nástroje cacti. Když se v menu přepneme do položky "graphs" vidíme defaultně vytvořené grafy. Tyto grafy se týkají využívání paměti, průměrné zátěži, počtu přihlášených uživatelů a počtu procesů. Můžeme nastavit, v jakém intervalu (datum, čas) od, do má vykreslený graf zobrazovat údaje. V položce "console" můžeme provádět různé změny v nastavení, vytvářet nové grafy a podobně. Těchto možností je spousty a není důvod je všechny v této bakalářské práci vyjmenovávat. Ve zkratce si tento nástroj můžeme přizpůsobit podle toho, jaké chceme mít grafy, neboli co chceme

[1] http://www.unixmen.com/install-cacti-ubuntu-14-04/

sledovat. Opět tedy existuje otázka, jestli toto opravdu potřebujeme. Nástroj cacti poskytuje vhodnou variantu pro dlouhodobé sledování, co se týče prezentování využívání serveru. Ne však pro výkon našeho serveru.

Ntop

Na našem serveru je nástroj ntop přístupný na url http://localhost:3000/. Poskytuje nám různé statistiky formou tabulek a grafů. Narozdíl od nástroje cacti slouží pouze pro shromažďování informací o provozu na síti. Jeho tabulky a grafy se tedy týkají například statistik o paketech, vytížení sítě, síťových protokolech a informací o aktuálně připojených hostech.

Pro demonstraci si otevřeme v menu položku "All Protocols ->Traffic". Zobrazí se nám tabulka se statistikami přenosu dat k různým hostům. V terminálu zadáme příkaz *ping IPapacheServeru -c 5*. Přepneme se zpátky do nástroj ntop, aktualizujeme stránku a uvidíme, že nám v hostech přibude náš Apache server. Je u něj zaznamenán počet přenesených dat a také typy protokolů. V tomto případě jsou všechna data přiřazena k protokolu ICMP (naše požadavky odeslané nástrojem ping). Pomocí nástroje jmeter spustíme generování zátěže na náš Apache server stejným způsobem jako v ukázce 1 u podkapitoly 5.1 "Využití nástrojů na modelu webového serveru Apache" a v ntop budeme sledovat statistiky přenosu. V tabulce uvidíme zvyšující se počet přenesených dat, jak pro náš server generující požadavky, tak i pro Apache server. Také zjistíme, že data jsou přenášena pomocí TCP protokolu. Po rozkliknutí hosta můžeme vidět další podrobnější informace.

Pomocí nástroje ntop tedy můžeme v přehledném grafickém uživatelském rozhraní sledovat různé statistiky vytížení sítě. Jak nyní zjistíme, pro tento způsob sledování sítě musíme také něco obětovat. Pomocí nástroje top zjistíme, že pokud nástroj ntop zrovna používáme, proces ntop využívá 12 % až 13 % procesoru a 1,5 % paměti.

sar

Nástroj sar, ale také ostatní uvedené nástroje, které spadají pod balíček sysstat (iostat, mpstat a vmstat), mohou být využity pro zobrazení informací o komponentách systému. Nástroj sar uchovává údaje od posledního vypnutí serveru. Tyto údaje jsou zaznamenány každých 10 minut (lze změnit). Pomocí parametrů můžeme zvolit, o jaké části systému chceme zobrazit informace. Díky tomu můžeme například zjistit v jakých časech a v kterých dnech byla zvýšená zátěž na náš server a jakých částí systému se to týkalo. Oproti nástroji cacti jde o méně invazivní variantu, která ovšem neposkytuje grafické uživatelské rozhraní a grafy. Nástroje iostat, mpstat a vmstat mohly být využity v předchozích ukázkách na Apache modelu a NFS modelu. Nicméně, informace, které by nám poskytly byly zjištěny pomocí jiných nástrojů.

Kapitola 6

Závěr

Cílem této práce bylo vyhledat a nastudovat nástroje pro profilování aplikací a systémů. Poté tyto vyhledané nástroje porovnat z hlediska složitosti použití a invazivnosti. Následně navrhnout a zprovoznit modelové situace, které mají simulovat reálný provoz systému. Na těchto modelech poté demonstrovat využití nalezených nástrojů za účelem nalezení a odstranění problémů s výkonem systému.

Nejprve jsem tedy musel vyhledat a nastudovat vhodné nástroje pro demonstraci. Nalezené nástroje svým účelem zahrnují všechny komponenty systému. Podle svého účelu jsou také roztříděné do skupin. Každý uvedený nástroj jsem stručně popsal a uvedl některé jeho výhody a nevýhody. U některých nástrojů jsem také uvedl podobný nástroj, který by plnil stejnou funkci. Proto nebyl důvod takový nástroj využít, byl uveden pouze jako alternativa. Velké množství těchto nástrojů využívá informace, které poskytuje jádro systému prostřednictvím souborových systému procfs a sysfs. Udělal jsem proto také rozbor obsahu těchto souborových systému dostupných v adresářích /proc a /sys.

Dále jsem provedl porovnání využitých nástrojů z hlediska složitosti použití a invazivnosti. U porovnávání složitosti použití nástrojů s rozhraním příkazové řádky (CLI) byly rozhodující zejména potřebné znalosti systému k pochopení výsledku poskytnutého těmito nástroji. V menší míře také složitost využití parametrů jednotlivých nástrojů. U nástrojů využívajících grafického uživatelského rozhraní (GUI) byla rozhodující zejména složitost provádění nabízených úkonů. U porovnávání invazivnosti byl rozhodující způsob získávání informací jednotlivými nástroji. Nástroje, které využívaly obsahu adresářů /proc a /sys, vykazovaly nižší invazivnost. Složitost použití a invazivnost spolu navzájem příliš nesouvisí. Porovnával jsem je individuálně a roztřídil jsem nástroje do tří skupin, které vyjadřují míru složitosti či invazivnosti.

Poté přišlo na řadu vytvoření modelů, na kterých bude demonstrace využití nástrojů probíhat. Nejprve byly po konzultaci s panem Ing. Tomášem Kašpárkem zvoleny technologie, které byly následně využity. Jedná se o technologie webového serveru Apache a NFS serveru. Pro vytvoření modelů těchto serverů bylo využito virtualizace systémů pomocí technologie hyper-v. Celkově jsem vytvořil čtyři virtuální stroje a přidělil jim totožné prostředky. Jeden virtuální stroj sloužil jako Apache server, další jako NFS server. Třetí byl využit pro zrcadlení obsahu Apache serveru. Poslední virtuální stroj sloužil pro generování požadavků (zátěže), čímž jsem simuloval reálný provoz těchto serverů. Jako operační systém, pro tyto virtuální stroje, jsem zvolil Linux Lite 2.6. Vytvořené modely byly popsány.

Poslední část práce jsem věnoval demonstraci využití uvedených nástrojů na vytvořených modelech. U každého modelu jsem popsal způsob generování požadavků a poté samotné postupy pro hledání příčin problémů s výkonem. Tyto postupy jsem rozdělil do jednotlivých

ukázek, které vždy náleží svému modelu. Tyto ukázky jsou buď názorné pro pochopení a popis probíhajících operací, nebo poskytují pohled administrátora, který má řešit neočekávaný problém či sám nějaký problém zaregistruje. V těchto ukázkách jsem se zaměřil na pochopení probíhajících operací v systému, které vedly k odhalení příčin problémů s výkonem. Na jejich základě jsem byl také schopen navrhnout jejich řešení.

Tato práce mimo jiné slouží jako návod pro nalezení problémů s výkonem systému. Využít lze také pro seznámení s různými nástroji, které se v praxi využívají a také pro zvolení vhodného nástroje k použití. Dále také pro seznámení s informacemi, které jsou poskytovány jádrem systému prostřednictvím souborových systémů procfs a sysfs.

Na této práci by se dalo pokračovat vytvořením dalších modelů s využitím jiných technologií jako je například samba. Také existuje možnost různých úprav generované zátěže a nastavení na stávajících modelech, které by mohlo přinést další zajímavé výsledky. Další experimentování by také mohlo přinést odhalení dalších užitečných nástrojů.

Literatura

[1] *Linux kernel profiling with perf [online]*. 2015 [cit. 2016-05-11].
URL https://perf.wiki.kernel.org/index.php/Tutorial

[2] Apache Software Foundation: *Apache HTTP SERVER PROJECT [online]*. 2016 [cit. 2016-01-28].
URL https://httpd.apache.org/

[3] Apache Software Foundation: *Apache JMeter [online]*. 2016 [cit. 2016-01-28].
URL http://jmeter.apache.org/

[4] Apache Software Foundation: *Apache MPM prefork [online]*. 2016 [cit. 2016-04-15].
URL https://httpd.apache.org/docs/2.4/mod/prefork.html

[5] Callaghan, B.; Pawlowski, B.; Staubach, P.: NFS Version 3 Protocol Specification RFC 1813. Sun Microsystems, Inc., June 1995 [cit. 2016-01-19].

[6] Cohen, W.; Suthikulpanit, S.; Deacon, W.; aj.: *OProfile [online]*. 2015 [cit. 2016-05-11].
URL http://oprofile.sourceforge.net/news/

[7] Eigler, F. C.: *Systemtap tutorial [online]*. 2016 [cit. 2016-05-15].
URL https://sourceware.org/systemtap/tutorial/tutorial.html

[8] Kerrisk, M.: Linux man pages online [online]. 2016 [cit. 2016-01-27].
URL http://man7.org/linux/man-pages/index.html

[9] Linux Lite Free Operating System: *Linux Lite [online]*. 2016 [cit. 2016-01-29].
URL https://www.linuxliteos.com/

[10] M. Eisler, E.: XDR: External Data Representation Standard RFC 4506. Network Appliance, Inc., May 2006 [cit. 2016-01-19].

[11] Mochel, P.: *The sysfs Filesystem [online]*. Linux Kernel Organization, Inc., 2005 [cit. 2015-12-26].
URL https://sourceware.org/systemtap/tutorial/tutorial.html

[12] Oracle Corporation: *Mysql [online]*. 2016 [cit. 2016-01-29].
URL http://www.mysql.com/

[13] Oracle Corporation: *5.1.4 Server System Variables [online]*. 2016 [cit. 2016-04-15].
URL
https://dev.mysql.com/doc/refman/5.6/en/server-system-variables.html

[14] Shepler, S.; Callaghan, B.; Robinson, D.; aj.: Network File System (NFS) version 4 Protocol RFC 3530. Sun Microsystems, Inc. and Network Appliance, Inc. and Hummingbird Ltd., April 2003 [cit. 2016-01-19].

[15] Shepler, S.; Eisler, M.; Noveck, D.: Network File System (NFS) Version 4 Minor Version 1 Protocol RFC 5661. NetApp, Storspeed, Inc., January 2010 [cit. 2016-01-19].

[16] Sun Microsystems, I.: NFS: Network File System Protocol Specification RFC 1094. Sun Microsystems, Inc., March 1989 [cit. 2016-01-19].

[17] TechTarget: Definition mirror [online]. 2016 [cit. 2016-01-30].
URL http://searchstorage.techtarget.com/definition/mirror/

[18] Thurlow, R.: RPC: Remote Procedure Call Protocol Specification Version 2 RFC 5531. Sun Microsystems, May 2009 [cit. 2016-01-19].

[19] W3Techs.com: *Usage of web servers for websites [online]*. 2015-12-28 [cit. 2015-12-28].
URL http://w3techs.com/technologies/overview/web_server/all

[20] Wikimedia Foundation, Inc.: *MediaWiki 1.26 [online]*. 2016 [cit. 2016-04-15].
URL https://www.mediawiki.org/wiki/MediaWiki_1.26

Příloha A

Obsah CD

Na přiřazeném CD nalezneme:

- /apache - konfigurační soubory Apache.
- /nfs - konfigurační soubory NFS.
- /jmeter - uložené testovací plány.
- /systemtap - skript využitý v ukázce modelu zrcadleného serveru.
- /database - využitý dump v modelu Apache serveru.
- /wikimedia - použité zdrojové kódy wikimedie verze 1.26.2.
- /xdress00_system_profiling.pdf - pdf verze této bakalářské práce.