Analiza zmogljivosti oblačnih in strežniških storitev

Uredil prof. dr. Miha Mraz

Maj 2016

Kazalo

Predgovor

Pričujoče delo je razdeljeno v devet poglavij, ki predstavljajo različne analize zmogljivosti nekaterih oblačnih izvedenk računalniških sistemov in njihovih storitev. Avtorji posameznih poglavij so slušatelji predmeta *Zanesljivost in zmogljivost računalniških sistemov*, ki se je v štud.letu 2015/2016 predaval na 1. stopnji univerzitetnega študija računalništva in informatike na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Vsem študentom se zahvaljujem za izkazani trud, ki so ga vložili v svoje prispevke.

prof. dr. Miha Mraz, Ljubljana, v maju 2016

Poglavje 1

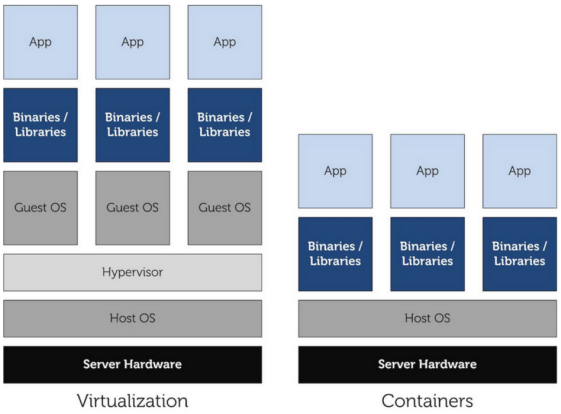
Zmogljivostna analiza virtualnih implementacij

Nejc Nadižar, Eva Križman, Klemen Klanjšček

1.1 Predstavitev ideje

V okviru seminarske naloge pri predmetu Zanesljivost in zmogljivost računalniških sistemov, smo se v skupini odločili narediti primerjavo med dvema realizacijama strežnika za virtualno okolje.
Prva implementacija (levo na sliki 1.1) je realizirana na najbolj pogost način, kar pomeni da na strojni opremi teče namenski operacijski sistem ("hypervisor"), nad katerim se potem kreirajo virtualne delovne postaje s svojim nespremenjenim operacijskim sistemom, do katerih lahko dostopamo preko oddaljenega dostopa.
Druga implementacija (desno na sliki 1.1) je realizirana s programskimi zabojniki ("software containers"), kjer direktno na strojni opremi teče nespremenjen operacijski sistem, znotraj katerega so zabojniki realizirani na nivoju posameznih aplikacij. Iz zabojnika navzven se vidi samo sliko sistema, torej znotraj zabojnika ne moremo direktno vplivati na sam sistem.

Za testiranje obeh implementacij smo uporabili že obstoječa orodja za testiranje zmogljivosti sistema. Omejili smo se na testiranje zmogljivosti samega jedra operacijskega sistema, diskovja ter omrežnih storitev. Za testiranje mrežne povezave smo napisali klienta, ki ustvarja uporabnike, kateri bodo obremenjevali naš testni sistem.



Slika 1.1: Strukturi sistemov [1].

1.2 Predstavitev rešitve

1.2.1 Strojne specifikacije testnega sistema

Testni sistem predstavlja zmogljivejša delovna postaja ("workstation") s sledečimi specifikacijami

- Procesor: Intel i7-2600 (3.40GHz),
- Pomnilnik: 16 GB,
- Diskovje: 250 GB SDD,
- Mreža: 100MB/1GB.

1.2.2 Programske specifikacije testnih sistemov

Sistem, ki uporablja navadno virtualizacijo [2] (levo na sliki 1.1), uporablja kot osnovni operacijski sistem "VMWare ESXi 5.5.0 Hypervisor", z brezplačno licenco in z verzijo jedra (kernel) "VMKernel release 2403361". Virtualne delovne postaje, ki tečejo na prej omenjenem sistemu, uporabljajo operacijski sistem "CentOS 6.7 (Final)", z verzijo jedra "Linux 2.6.32-042stab113.21 x86-64".

Sistem z virtualizacijo na nivoju operacijskega sistema [3] (desno na sliki 1.1) uporablja kot osnovni operacijski sistem "CentOS 6.7 (Final)", z verzijo jedra "Linux 2.6.32-042stab113.21 x86-64". Aplikacije tečejo v svojem virtualnem okolju ("software containers") s sistemom "OpenVZ 2.6.32-042stab113.21 i686" s predlogo sistema "CentOS 6.7 template x86".

Lokacija strežnika je v Ljubljani, medtem ko se nekateri testi izvajajo dislocirano, drugi pa preko oddaljenega dostopa direktno na strežniku (v kolikor ob

testu ni navedena lokacija se test izvaja lokalno).

Pri ocenjevanju zmogljivosti obeh sistemov pri različnih obremenitvah smo se pri testih odločili za testiranje "enouporabniške" (en "container" v enem primeru in ena virtualna delovna postaja čez celoten sistem v drugem) in "večuporabniške" (več uporabnikov na eni virtualni delovni postaji ali na več virtualnih delovnih postajah) implementacije.

Pri testiranju "enouporabniške" implementacije, smo se osredotočili na sledeče teste:

- Test I/O operacij: Testirali smo zmogljivost pri ustvarjanju, kopiranju datotek (različnih velikosti, različnega števila datotek).
- Test mreže: Testiramo učinkovitost in zmogljivost operacij, ki uporabljajo mrežno opremo (dostop do mrežne opreme, diskovnih enot, interneta, drugih sistemov,...).
- Test CPU: Testiramo zmogljivost samega procesorja, v eni in drugi implementaciji (veliko benchmarkov je že implementiranih).
- Test preobremenitve: Testiramo odzivnost sistema, ko zahtevnost bremena presega same specifikacije sistema (enormna količina operacij, preobremenitev sistema z že obstoječimi programi).

Nekatera obstoječa orodja za testiranje že zajemajo več kot en tip testov in so na kratko opisani kasneje. Večina testov, ki smo jih morali napisati sami, je realizirana preko BASH skript ali preprostejših programov (C), razen kjer je navedeno drugače.

Testiranje "večuporabniške" implementacije poteka na naslednje načine:

- Test odzivnosti: testirali smo čas izvedbe ukaza(programa) in čas do pridobitev rezultata (pri različnih obremenitvah sistema; več uporabnikov naenkrat, obremenitev gostitelja,...);
- Test komunikacije: testiramo vpliv komunikacije med uporabniki na delovanje in zmogljivost sistema (prenos podatkov med uporabniki na sistemu); testiralo se je komunikacijo med več uporabniki znotraj programskih zabojnikov in virtualnega okolja, kot tudi komunikacijo med več zabojniki in virtualnimi delovnimi postajami;
- Vpliv infrastrukture: merili smo zmogljivost sistema pri dostopu uporabnikov iz različnih lokacij (implicirana različna infrastruktura povezav);

1.3 Definicija orodij in metrik

Trenutna izbira orodij za testiranje vsebuje:

- **UnixBench** [4]: orodje za testiranje operacijskega sistema (eno/več opravil, paralelno izvajanje); vsebuje teste za delo z nizi ("Dhrystone"), delo v plavajoči vejici ("Whetstone"), število sistemskih klicev ("execl throughput"), kopiranje datotek preko predpomnilnika, komunikacijo med procesi na sistemu, količina "overhead"-a pri sistemskih klicih ("system call overhead");
- **Interbench** [5]: orodje, ki testira interakcijo sistema (CPU, I/O, "scheduler", datotečni sistem); beleži hitrost in učinkovitost dodeljevanja procesorja programom, pri različnih obremenitvah sistema;
- **Ttcp**: meri zmogljivost komunikacije preko omrežne povezave;
- **Iperf3** [6]: je uporabljen kot alternativa predhodnem testu (po vsej verjetnosti uporabljen bolj uporaben za naše potrebe); meri zmogljivost in zanesljivost povezave (količina prenosa, hitrost povezave, število ponovnih pošiljanj);
- **Time**: je standardni linux program, ki meri čas izvajanja določenega ukaza (procesorski čas, uporabljen (ali izpeljanke) v večini programov, ki bi jih implementirali sami);
- **Hdparam**: je orodje za testiranje diskov in ostalih medijev za hranjenje podatkov (tako HDD kot SSD);

1.4 UnixBench

1.4.1 Opis testa

Unixbench orodje preverja zmogljivost operacijskega sistema. Njegovi rezultati so bolj odvisni od jedra ("kernel") operacijskega sistema in verzije orodij s katerimi so bili programi prevedeni, tako da strojna oprema nima velikega vpliva na sam test. Orodje uporablja več različnih testov, vsako z drugačno najprimernejšo enoto.
Meritve v tabeli 1.1 predstavljajo sekvenčne teste, v tabeli 1.3 pa so predstavljeni paralelni testi. Obe meritvi uporabljata naslednje enote: lps[1] prestavlja iteracijo zanke na sekundo ("loop iterations per second"), posamezen test se izvaja 10 sekund in se ponovi 7-krat; lpm[2] predstavlja iteracijo zanke na milisekundo ("loop iterations per milisecond"), posamezen test se izvaja 60 sekund in se ponovi 2-krat; lps[3] enako iteracija zanke na sekundo, razlika je v trajanju (20 sekund) in številu ponovitev (2-krat); MWIPS[4] predstavlja milijon operacij v plavajoči vejici na sekundo, posamezen test traja 10 skund in se ponovi 7-krat; KBps[5] predstavlja "KiloBytes" na sekundo, test se izvaja 30 sekund in se ponovi 2-krat.

Tabeli 1.2 in 1.4 sta samo povzetek v obliki indeksov, da se rezultati lažje primerjajo z zmogljivostjo ostalih sistemov (določen sistem se da primerjati s sistemi po svetu [7]).

1.4.2 Hipoteza

Predvidevamo, da je klasična virtualizacija hitrejša pri kopiranju datotek, katere je možno v celoti naložiti v pomnilnik, zaradi boljše implementacije samega dostopa do njega. Virtualizacija na nivoju operacijskega sistema ima prednost pri paralelnem izvajanju, saj uporablja večnitenje strojne opreme in ne programske emulacije le te. Večkratna ponovitev testov ne bi spremenila rezultatov, saj že test sam po sebi izvede vsak test večkrat.

1.4.3 Rezultati

"Benchmark name"	ESXi-VM	OpenVZ
Dhrystone 2 using register variables (lps[1])	40958278.5	26378260.0
Double-Precision Whetstone (MWIPS[4])	4388.1	3347.8
Execl Throughput (lps[3])	3273.1	5331.2
File Copy 1024 bufsize 2000 maxblocks (KBps[5])	1030616.5	870075.2
File Copy 256 bufsize 500 maxblocks (KBps[5])	283165.4	229052.6
File Copy 4096 bufsize 8000 maxblocks (KBps[5])	2482963.3	2327838.4
Pipe Throughput (lps[1])	1953764.8	1674236.4
Pipe-based Context Switching (lps[1])	60294.2	207629.1
Process Creation (lps[3])	14316.2	17607.7
Shell Scripts (1 concurrent) (lpm[2])	7809.2	7478.7
Shell Scripts (8 concurrent) (lpm[2])	2733.3	3256.6
System Call Overhead (lps[1])	2908854.9	2115694.5

Tabela 1.1: Rezultati meritev na enem procesorju

"Benchmark index results"	ESXi-VM	OpenVZ
Dhrystone 2 using register variables	3509.7	2260.3
Double-Precision Whetstone	797.8	608.7
Execl Throughput	761.2	1239.8
File Copy 1024 bufsize 2000 maxblocks	2602.6	2197.2
File Copy 256 bufsize 500 maxblocks	1711.0	1384.0
File Copy 4096 bufsize 8000 maxblocks	4281.0	4013.5
Pipe Throughput	1570.6	1345.8
Pipe-based Context Switching	150.7	519.1
Process Creation	1136.2	1397.4
Shell Scripts (1 concurrent)	1841.8	1763.8
Shell Scripts (8 concurrent)	4555.5	5427.7
System Call Overhead	1939.2	1410.5
System Benchmarks Index Score	1539.9	1596.6

Tabela 1.2: Primerjave indeksov rezultatov na enem procesorju

"Benchmark name"	ESXi-VM	OpenVZ
Dhrystone 2 using register variables (lps[1])	153975510.8	109049113.5
Double-Precision Whetstone (MWIPS[4])	16404.3	22967.9
Execl Throughput (lps[3])	7272.1	31693.3
File Copy 1024 bufsize 2000 maxblocks (KBps[5])	1153151.1	937811.1
File Copy 256 bufsize 500 maxblocks (KBps[5])	318685.8	246762.0
File Copy 4096 bufsize 8000 maxblocks (KBps[5])	3354213.7	3027486.1
Pipe Throughput (lps[1])	7194671.0	7646409.9
Pipe-based Context Switching (lps[1])	1438292.1	1345339.3
Process Creation (lps[3])	64165.7	83359.0
Shell Scripts (1 concurrent) (lpm[2])	15011.1	29866.7
Shell Scripts (8 concurrent) (lpm[2])	2133.0	4085.3
System Call Overhead (lps[1])	8868482.1	9620146.3

Tabela 1.3: Rezultati meritev na več procesorjih

"Benchmark index results"	ESXi-VM	OpenVZ
Dhrystone 2 using register variables	13194.1	9344.4
Double-Precision Whetstone	2982.6	4176.0
Execl Throughput	1691.2	7370.5
File Copy 1024 bufsize 2000 maxblocks	2912.0	2368.2
File Copy 256 bufsize 500 maxblocks	1925.6	1491.0
File Copy 4096 bufsize 8000 maxblocks	5783.1	5219.8
Pipe Throughput	5783.5	6146.6
Pipe-based Context Switching	3595.7	3363.3
Process Creation	5092.5	6615.8
Shell Scripts (1 concurrent)	3540.3	7044.0
Shell Scripts (8 concurrent)	3554.9	6808.8
System Call Overhead	5912.3	6413.4
System Benchmarks Index Score	4004.5	4962.4

Tabela 1.4: Primerjave indeksov rezultatov na več procesorjih

1.4.4 Komentar

Na podlagi rezultatov lahko potrdimo našo predhodno hipotezo, ker v primeru kopiranja datotek preko pomnilnika, je virtualni sistem hitrejši (testi "File Copy bufsize"). Medtem, ko pa virtualizacija operacijskega sistema bolje izkorišča večnitnost procesorja in so zaradi tega boljši rezultati pri ostalih testih. Sekvenčni (v tabeli 1.1) in paralelni (v tabeli 1.3) testi imajo enak zaključek, se pa pri paralelnih testih vidi večja razlika med sistemoma, kar kaže na boljšo implementacijo v primeru virtualizacije na nivoju operacijskega sistema.

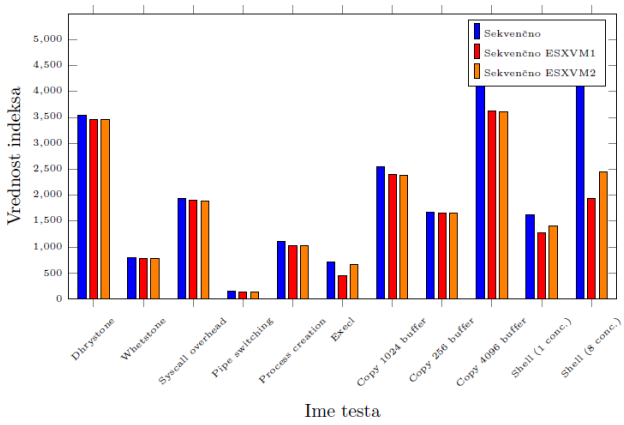
1.5 UnixBench - primerjava med sistemoma

1.5.1 Opis testa

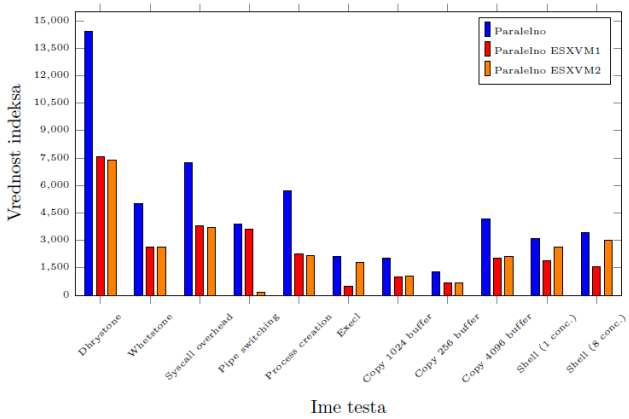
V naslednjem testu z orodjem **UnixBench** naredimo primerjavo med sekvečno in paralelno zmogljivostjo obeh sistemov. V primeru sekvenčnih testov dve virtualni delovni postaji ali dva programska zabojnika uporabljata vsaka polovico vseh sistemih sredstev. Slika 1.2 predstavlja sekvenčne (vsaka delovna postaja poganja samo eno kopijo testov) teste, medtem ko slika 1.3 prestavlja paralelne (vsaka delovna postja pognaja osem kopij testov naenkrat) teste.
Na sliki 1.4 imamo sekvenčne teste "OpenVZ" in na 1.5 paralelne (razlaga enako kot pri "ESX", samo da vsak programski zabojnik zaganja teste).

1.5.2 Hipoteza

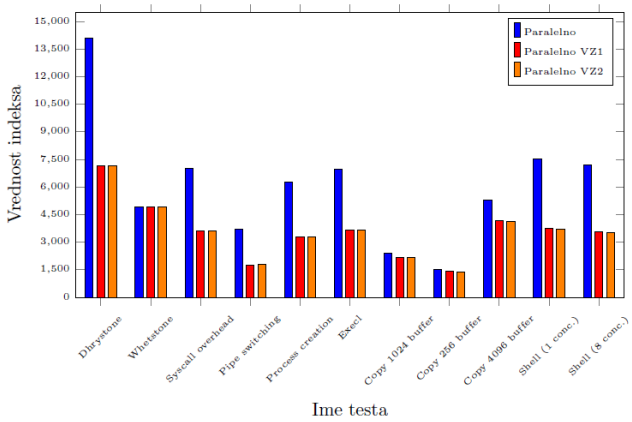
Kot v prejšnjem testu predvidevamo da bodo paralelni testi "OpenVZ" boljši od paralelnih testov sistema "ESX".



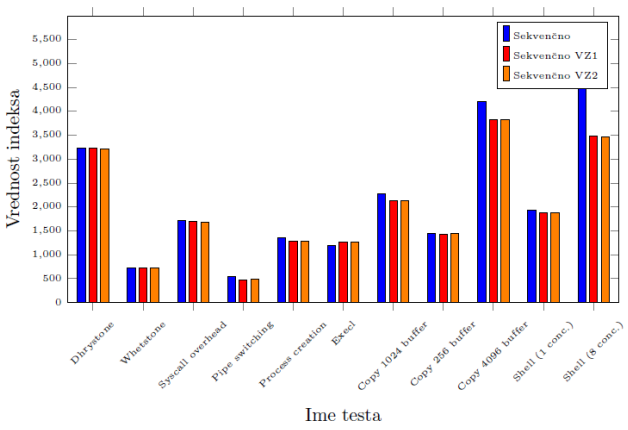
Slika 1.2: Primerjava sekvenčnih testov na ESX sistemu.



Slika 1.3: Primerjava paralelnih testov na ESX sistemu.



Slika 1.4: Primerjava sekvenčnih testov na OpenVZ sistemu.



Slika 1.5: Primerjava paralelnih testov na OpenVZ sistemu.

1.5.3 Komentar

Opazili smo, da se sistemska sredstva razporedijo bolj enakomerno v primeru "OpenVZ".

1.6 Iperf3

1.6.1 Opis testa

Orodje **Iperf3** testira zmogljivost in zanesljivost mrežne opreme sistema (v našem primeru mrežnih kartic). Teste sestavljajo sekvenčno in paralelno pošiljanje podatkov v določenem časovnem okvirju. Rezultati so predstavljeni kot količina podatkov ("transfer";v enoti "GBytes") v prej omenjenem časovnem okvirju("interval";v sekundah), zmogljivost povezave("bandwidth";v "Gbits" na sekundo), ter število ponovnih pošiljanj ("retry").
Za naše potrebe smo uporabili sekvenčno pošiljanje (v tabeli 1.5), paralelno pošiljanje 10-ih podatkov (v tabeli 1.6) in paralelno pošiljanje 100-ih podatkov (v tabeli 1.7). Test je bil izveden na istem sistemu, kjer je ena virtualna delovna postaja ali en programski zabojnik služil kot strežnik, drugi pa kot odjemalec.

1.6.2 Hipoteza

Predvidevali smo, da je sistem "OpenVZ" zmogljivejši pri višjih stopnjah paralelne obremenitve. Testi zajemajo večkratno ponovitev, saj vsak test traja 120 sekund, med katerim se na vsake 2 sekunde beleži količina prenesenih podatkov in hitrost prenosa. Na koncu pa se količina sešteje, hitrost pa povpreči.

1.6.3 Rezultati

System	Interval	Transfer	Bandwidth	Retry	CPU load
ESXi-VM					
sender	120 sec	467 GBytes	33.5 Gbits/sec	3789	69.8%
receiver	120 sec	467 GBytes	33.5 Gbits/sec	-	61.2%
OpenVZ					
sender	120 sec	25.6 GBytes	1.83 Gbits/sec	0	100.0%
receiver	120 sec	25.6 GBytes	1.83 Gbits/sec	-	56.7%

Tabela 1.5: Rezultati meritev sekvenčnega pošiljanja

System	Interval	Transfer	Bandwidth	Retry	CPU load
ESXi-VM					
sender	120 sec	187 GBytes	13.4 Gbits/sec	10424	69.2%
receiver	120 sec	187 GBytes	13.4 Gbits/sec	-	19.0%
OpenVZ					
sender	120 sec	28.1 GBytes	2.01 Gbits/sec	0	100.0%
receiver	120 sec	28.1 GBytes	2.01 Gbits/sec	-	76.6%

Tabela 1.6: Rezultati meritev paralelnega (10) pošiljanja

System	Interval	Transfer	Bandwidth	Retry	CPU load
ESXi-VM					
sender	120 sec	125 GBytes	8.98 Gbits/sec	168627	74.4%
receiver	120 sec	125 GBytes	8.98 Gbits/sec	-	70.1%
OpenVZ					
sender	120 sec	32.7 GBytes	2.34 Gbits/sec	0	100.0%
receiver	120 sec	32.7 GBytes	2.34 Gbits/sec	-	98.8%

Tabela 1.7: Rezultati meritev paralelnega (100) pošiljanja

1.6.4 Komentar

Glede na rezultate testov implementacije mrežne opreme na naših testnih sistemih, lahko povzamemo, da je bila naša predhodnja trditev napačna, saj virtualni strežnik "ESXi" bolje izrablja povezavo s strojno opremo, medtem ko sistem "OpenVZ" realizira mrežne kartice programsko. Pri slednjem sistemu, je iz testov razvidno, da je ozko grlo in omejevalni faktor procesor, saj je pri vseh testih maksimalno obremenjen. Da je implementacije programska, je razvidno tudi iz podatka, da pri "OpenVZ" ni bilo potrebnih ponovnih pošiljanj datagamov.

1.7 Zaključek

Tekom seminarske naloge smo testirali sistemsko in omrežno zmogljivost dveh različnih implementacij virtualnega strežnika. Poizkušali smo ugotoviti katera implementacija je bolj primerna za različne vrste aplikacij in storitev.
Za storitve, ki vključujejo komunikacijo znotraj enega sistema, je boljša izbira "VMware ESX", saj uporablja strojno opremo kot pomoč pri komunikaciji. Pri "OpenVZ" pa komunikacija med drugimi zabojniki na istem sistemu poteka preko programsko implementirane opreme.
Storitve, ki uporabljajo visoko stopnjo paralelnosti so bolj smiselne s sistemom "OpenVZ", saj uporablja paralelnost strojne opreme, medtem ko lahko imamo pri "ESX" več emulacij procesorjev kot jih je v dejanski strojni opremi (tej emulacije procesorjev si delijo vsa sredstva fizičnega sistema).
Glede varnosti obeh sistemov ima prednost "OpenVz". Program (aplikacija) znotraj svojega zabojnika vidi samo sliko sistema na katerega nima direktnega vpliva. "ESX" pa načeloma nima dostopa do strežniškega operacijskega sistema, vendar ima administratorski (seveda odvisno od uporabniških pravic) dostop do svoje virtualne delovne postaje.

Literatura

[1] "System comparison." http://cloudacademy.com/blog/container-virtualization/, April 2016.

[2] "Operating system virtualization." https://en.wikipedia.org/wiki/Operating-system-level_virtualization, April 2016.

[3] "OpenVZ." https://en.wikipedia.org/wiki/OpenVZ, April 2016.

[4] "UnixBench." https://github.com/kdlucas/byte-unixbench, April 2016.

[5] "interbench." http://users.tpg.com.au/ckolivas/interbench/, April 2016.

[6] "iperf." https://github.com/esnet/iperf, April 2016.

[7] "ServerBear." http://serverbear.com/benchmarks, April 2016.