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 Raspberry Pi v funkciji spletnega strežnika

Gašper Kolar, Matjaž Mav

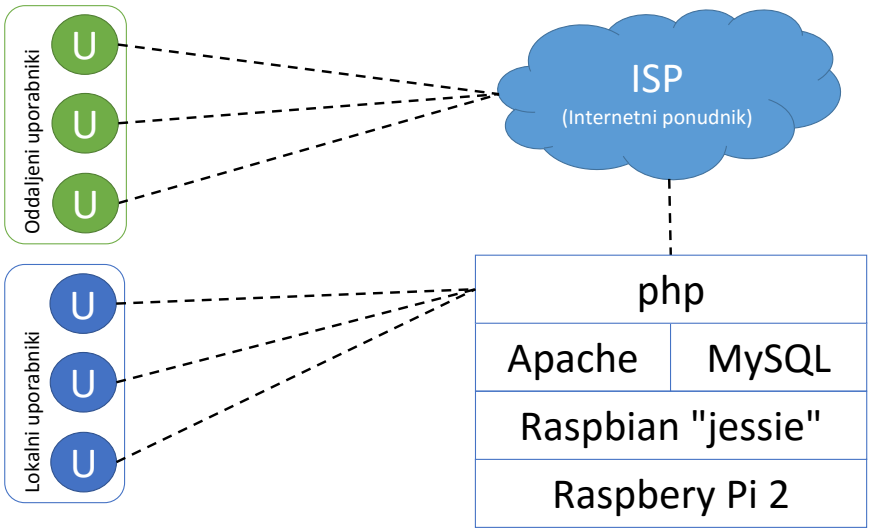
1.1 Predstavitev ideje

Raspberry Pi [1] predstavlja ugodno cenovno alternativo za gostovanje preprostih spletnih strani. V pričujočem prispevku sva ugotavljala kakšne so zmogljivosti preprostega LAMP spletnega strežnika na Raspberry Pi računalniku. Zanimal naju je predvsem povprečni odzivni čas ob različnih bremenih.

1.2 Definicija strežnika

Kot že rečeno sva uporabila računalnik Raspberry Pi, natančneje Raspberry Pi 2 model B [2]. Ta model uporablja štiri jedrni ARM Cortex-A7 procesor (Broadcom BCM2836 SoC) s taktom ure 900 MHz. Ima 1 GB delavnega pomnilnika tipa LPDDR [3]. Uporablja tudi 10/100 BaseT Ethernet priključek, kar pomeni, da lahko dosega hitrosti prenosa do 100 Mbit/s.

Kot OS je na Raspberry Pi nameščen Raspbian [4], ki bazira na Debian "jessie"distribuciji. Na ta Linux sva namestila LAMP spletni strežnik. Vlogo spletnega strežnika je prevzel Apache. Za podatkovno bazo sva uporabila MySQL. Uporabljeni programski jezik na strežniški strani je bil PHP. Shema strežnika



Slika 1.1: Osnovna shema strežnika.

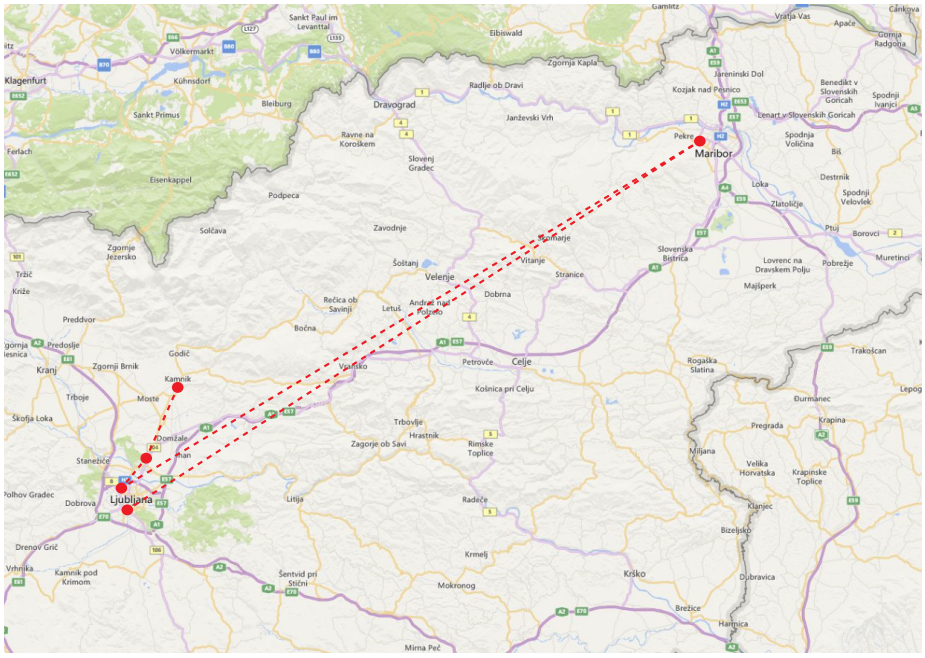
je predstavljena na Sliki 1.1

1.3 Definicija odjemalca

Ločila sva dva tipa odjemalcev. Prvi tip odjemalca je bil na istem lokalnem omrežju kot strežnik. Drugi tip ne bo na istem območju, tako, da bo promet tekel preko IP ponudnika (ISP). Strežnik se je nahajal v Kamniku (download 20 Mb/s, upload 1 Mb/s), ne lokalni odjemalci, tisti ki zahtevke pošiljajo preko ISP pa so se nahajali v Ljubljani (download 60 Mb/s, upload 2 Mb/s). Med odjemalcem in strežnikom je tako 21 km zračne razdalje, vendar je dejanska razdalja bistveno daljša. Orodje **tracert** je pokazalo, da komunikacija poteka po naslednjih korakih Ljubljana -> Maribor -> Ljubljana -> Trzin -> Kamnik, kar je vidno na sliki 1.2. V obeh primerih je odjemalec pošiljal preproste HTTP zahtevke s katerimi je zahteval spletno stran.

1.4 Breme

Celotni Raspberry Pi je bil namenjen LAMP strežniku tako, da so glavni vir bremena predstavljali odjemalci s svojimi HTTP zahtevki. Odjemalce sva si-



Slika 1.2: Grafični prikaz orodja **tracert**

mulirala z **ab (Apache Benchmark)** [5, 6]. To je orodje za ocenjevanje in primerjanje zmogljivosti HTTP strežnikov. Glavni namen tega orodja je prikazati, kako se obnaša strežnik pod različnimi bremeni. Poleg tega sva uporabila še **JMeter** [7]. To je orodje za testiranje zmogljivosti aplikacij, prvotno namenjeno za testiranje spletnih aplikacij. Z njim lahko testiramo tako statične, kot tudi dinamične vire pod raznimi obremenitvami.

1.5 Metrike

Vsako metriko sva merila za oba tipa odjemalcev (lokalnega in oddaljenega):

- **Število zahtevkov na sekundo**: Ocenjuje koliko zahtevkov je bilo poslanih ciljnemu strežniku v eni sekundi (ang. *Request per Second* oziroma *Average Load*). Povprečje se po navadi izračuna v intervalu od 1 do 5 minut.
- **Intenzivnost napak**: Ob velikih bremenih so napake bolj pogoste. Sem spadajo HTTP napake s statusnimi številkami 4xx ter 5xx. Ta metrika se meri v procentih, cilj pa je, da je procent čim manjši.
- **Povprečni odzivni čas**: Je ena glavnih metrik, ki meri povprečni čas od trenutka ko odjemalec pošlje zahtevek do prejetja odgovora (vključuje

tudi čas, ki ga strežnik potrebuje za pripravo odgovora). Meri se v milisekundah.

- **Čas neprekinjenega delovanja**: Meri se v procentih. Pove nam kakšen odstotek vsega časa v nekem časovnem intervalu je strežnik deloval. Pričakuje se vsaj 99% dostopnost.
- **Poraba procesorskih virov**: Meri se odstotek procesorskega časa, ki ga potrebuje strežnik za serviranje zahtevkov.
- **Poraba pomnilnika**: Meri se odstotek celotnega pomnilnika, ki ga uporablja strežnik za serviranje zahtevkov.
- **Število niti**: Spletna stran generira več niti, te pa so omejene s strojno opremo ter operacijskim sistemom.
- **Delež odprtih deskriptorjev**: Datoteke so lahko odprte za branje ali pa za pisanje. Sem spadajo tudi mrežni vtičniki. Ker ima operacijski sistem omejeno število odprtih deskriptorjev, nam to predstavlja problem. Metrika se meri v procentih (število odprtih deskriptorjev glede na število vseh možnih deskriptorjev, ki jih OS dovoljuje).

1.6 Rezultati meritev

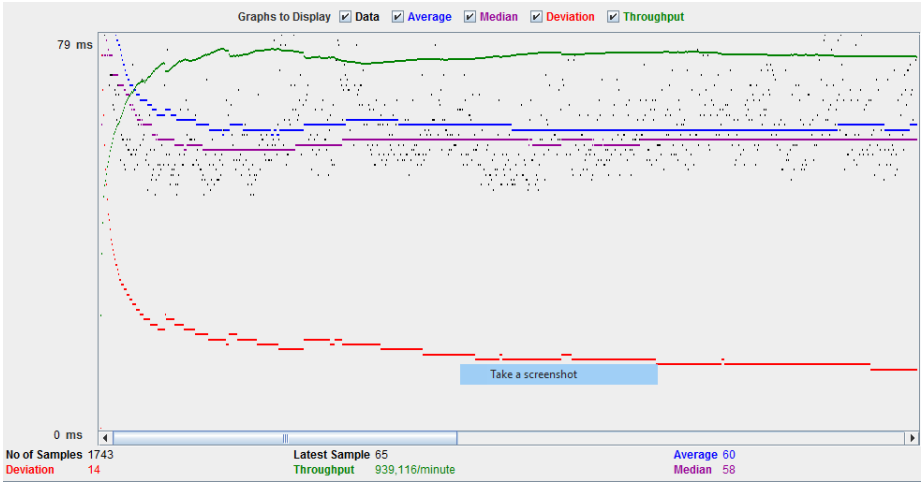
Vse meritve sva izvedla preko lokalnega omrežja ter preko internetnega ponudnika. Strežnik se je nahajal v Kamniku z naslednjimi internetnimi hitrostmi: download 20 Mb/s, upload 1 Mb/s, ping med ponudnikom in strežnikom pa je 36 ms (izmerjeno s pomočjo Speedtest). Uporabila sva tri različna orodja: ukaz **ab**, **JMeter** in **loader.io**, ki je spletna storitev za testiranje zmogljivosti spletnih aplikacij. Vsa orodja so dala zelo podoben rezultat. Opazi pa se velika razlika med lokalnimi in oddaljenimi zahtevki. V nadaljevanju so predstavljeni rezultati posameznih testov.

1.6.1 Poraba procesorja in pomnilnika

Poraba procesorja in pomnilnika je bila vseskozi zelo nizka. Procesorska poraba se je gibala okoli 0.7% na eno nit apache strežnika. Pomnilniška poraba je bil povsem zanemarljiva.

1.6.2 JMeter v lokalnem omrežju

Najprej sva uporabila orodje **JMeter** znotraj lokalnega omrežja (prvi tip odjemalca). Rezultati so vizualizirani na sliki 1.3. Vidi se, da se je povprečni odzivni čas ustalil pri 60 ms. To je čas od trenutka ko zahtevek zapusti računalnik pa do prejetja odgovora.



Slika 1.3: Rezultati testiranje z JMeter-om v lokalnem omrežju.

1.6.3 JMeter preko spletnega ponudnika

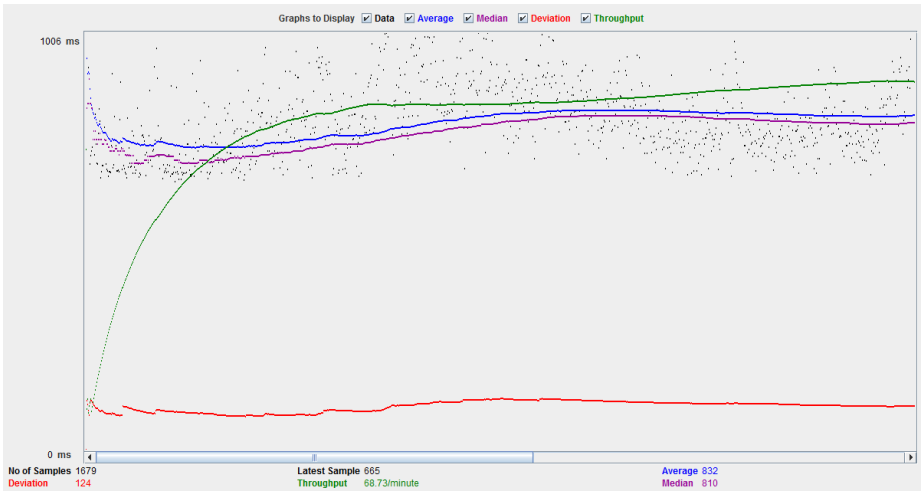
Nato sva orodje **JMeter** uporabila še preko spletnega ponudnika (drugi tip odjemalca). Rezultati so prikazani na sliki 1.4 in so grafično zelo podobni rezultatom **JMeter-a** v lokalnem omrežju. Glavna razlika je, da se je tu povprečni odzivni čas ustalil pri 832 ms, kar je približno štirinajstkrat daljši odzivni čas. Čas se spet mero od odpošiljanja zahtevka pa do prejetja odgovora. Sklepava da je razlog za tako večji odzivni čas slaba internetna hitrost na strani spletnega strežnika.

1.6.4 Primerjava povprečnih odzivnih časov

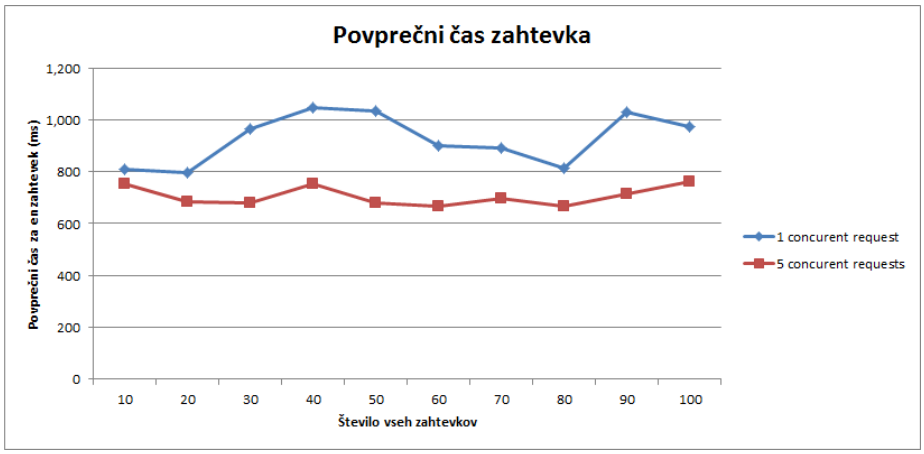
Nato sva uporabila **ab** orodja, da sva primerjala povprečne odzivne čase zahtevkov, če sva pošiljala le po en zahtevek na enkrat, ali pa 5 zahtevkov na enkrat. Rezultati so grafično prikazani na sliki 1.5. Razvidno je, da so povprečni odzivni časi manjši, če pošiljamo po 5 sočasnih zahtevkov, saj je takrat pride do boljše izkoriščenosti procesorja Izmerjeni čas se še vedno meri od trenutka odpošiljanja zahtevka pa do prejetja odgovora.

1.6.5 Primerjava števila neuspelih zahtevkov

Na sliki 1.6 je prikazana primerjava števila neuspešno serviranih zahtevkov med testom, ki sočasno pošlje en zahtevek in testom, ki sočasno pošlje pet zahtevkov. Testi so bili izvedeni z **ab** orodjem. Z slike se vidi, da ima test s petimi sočasnimi zahtevki nekoliko več neuspešnih zahtevkov, kar je normalno, saj je strežnik pod večjo obremenitvijo.



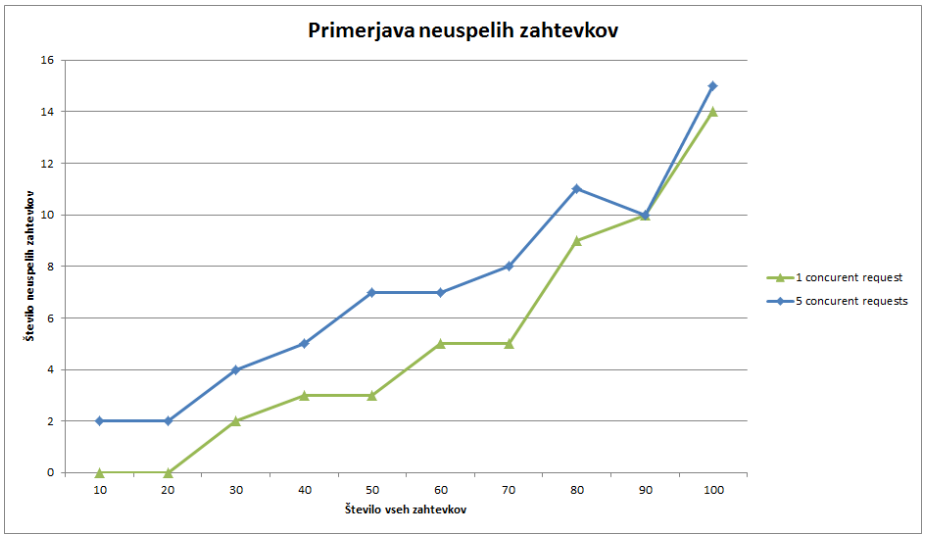
Slika 1.4: Rezultati testiranje z JMeter-om preko ISP ponudnika.



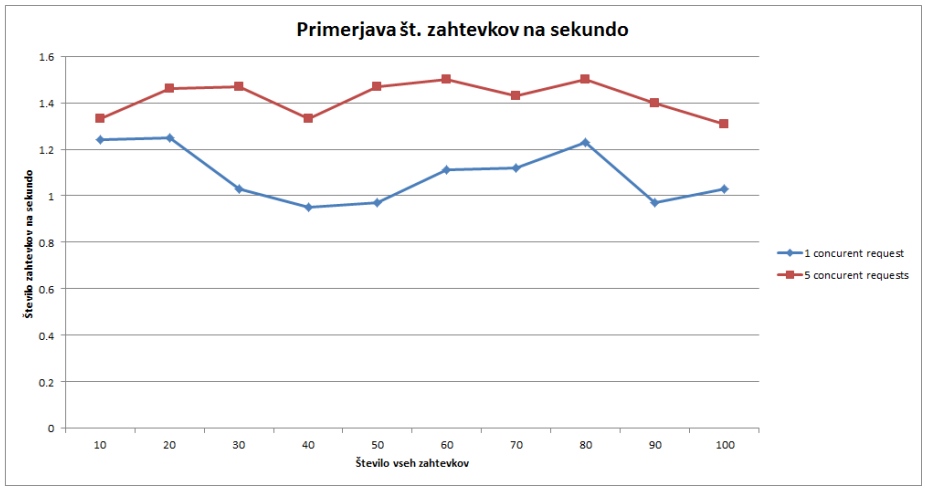
Slika 1.5: Primerjava povprečnih odzivnih časov zahtevkov.

1.6.6 Primerjava števila uspešno serviranih zahtevkov na sekundo

Slika 1.7 prikazuje koliko uspešnih zahtevkov, servira strežnik na sekundo. Med uspešno servirane zahtevke se štejejo samo tisti, ki jih je strežnik uspešno obdelal in poslal odgovor. Test s petimi sočasnimi zahtevki se tukaj obnaša dosti boljše, kot pa test z enim sočasnim zahtevkom. Test je bil opravljen z **ab** orodjem.



Slika 1.6: Primerjava števila neuspelih zahtevkov.

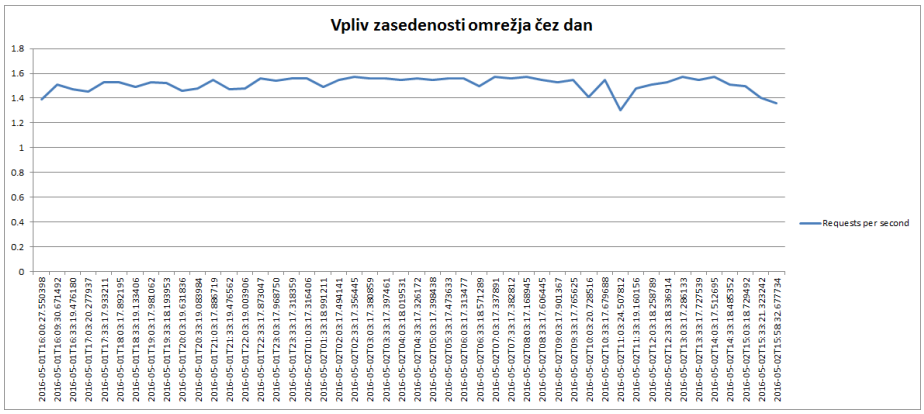


Slika 1.7: Primerjava števila uspešno serviranih zahtevkov na sekundo

1.6.7 Prikaz povprečnega števila zahtevkov v obdobju enega dne

Na sliki 1.8 je prikazano povprečno število izvedenih zahtevkov na sekundo. Meritve so bile izvedene med prvim in drugim majem. Vidi se, da tekom dneva zasedenost omrežja ni imela velikega vpliva. Meritve sva izvedla s pomočjo

skripte 1.8.1, ki se nahaja v prilogi.



Slika 1.8: Prikaz povprečnega števila zahtevkov v obdobju enega dne

1.6.8 Prikaz povprečnega števila zahtevkov v obdobju enega tedna

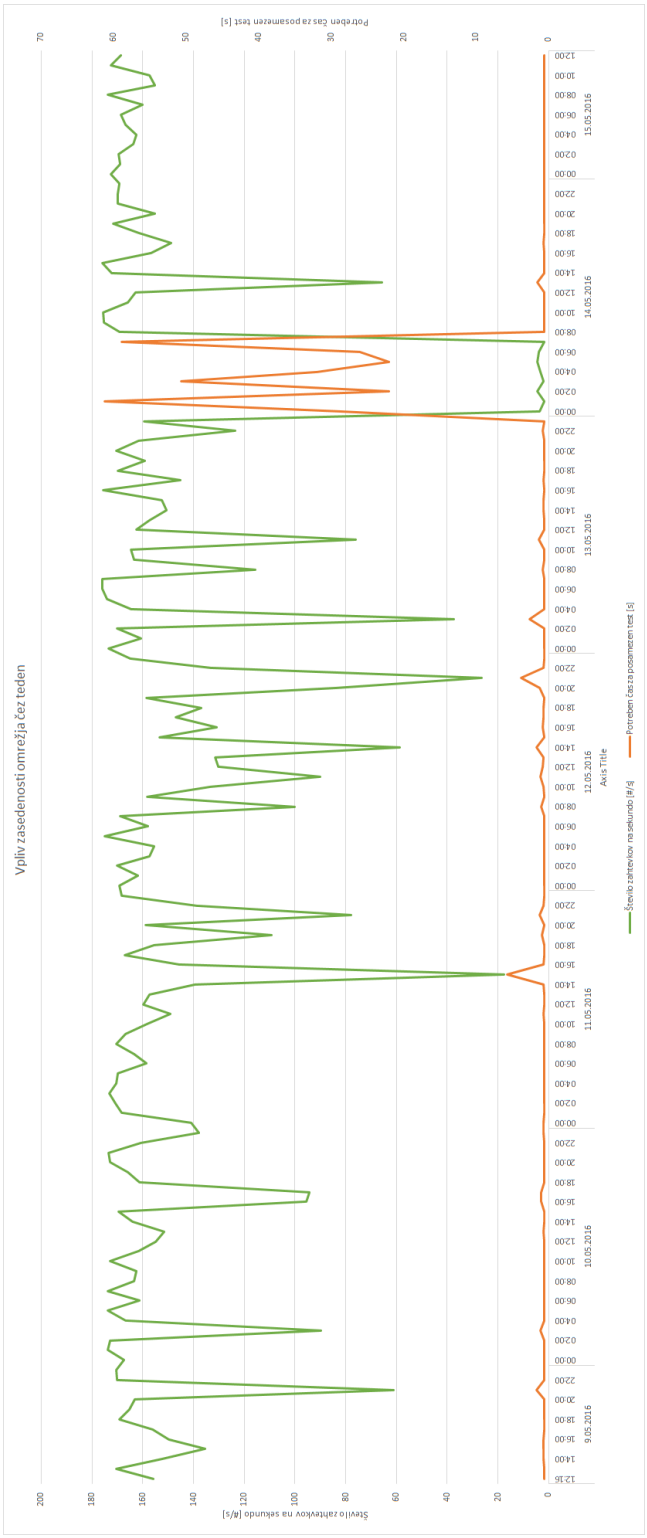
Ker v času enega dne nisva pridobila nobenih relevantnih podatkov sva test ponovila v času enega tedna. Meritve sva izvajala med 9.05.2016 in 15.05.2016. Rezultati so prikazani na sliki 1.9. Zasedenost omrežja spet ni imela nobenega vpliva na število poprečnih zahtevkov na sekundo. Sva pa tekom testa uporabljala izboljšane nastavitve, ki so izboljšale zmogljivost strežnika. Povprečni odzivni čas se je tako spustil na 280 ms kar je približno trikrat hitreje kot z starimi nastavitvami. Meritve sva izvedla s pomočjo skripte 1.8.1, ki se nahaja med prilogami.

V datoteki */etc/security/limits.conf* je bilo potrebno ročno nastaviti maksimalno število procesov in maksimalno število odprtih datotek.

```
#         /etc/security/limits.conf
#         ...
#         <domain>        <type>  <item>          <value>
          *               soft    nofile          10240
          *               soft    nproc           10240
#         ...
```

1.7 Zaključek

Cilj testiranja je bil preveriti, če lahko računalnik Raspberry Pi služi kot cenovno ugodna alternativa spletnega strežnika za majhna podjetja z malo spletnega prometa. Glede na rezultate meritev sva mnenja da ni ugodna alternativa. Cene spletnega gostovanja se začnejo že pri 12 evrih na leto medtem ko je najin Raspberry Pi stal okoli 40 evrov. Nato pa je potrebno imeti še hitro internetno povezavo, kar še dodatno poveča stroške, medtem ko nam pri spletnem gostovanju za to ni potrebno skrbeti. Poleg cenovnih razlogov pa so tu še tehnični. Na Raspberry Pi je potrebno ročno namestiti vso programsko opremo, ki jo potrebujemo (Apache, PHP, ...). Nato je po vsej verjetnosti potrebno nastaviti statičen IP naslove ter *port forwarding*. Poleg tega je potrebno konfigurirati spletni strežnik, da doseže optimalno zmogljivost.



Slika 1.9: Prikaz povprečnega števila zahtevkov v obdobju enega tedna

1.8 Priloge

1.8.1 Python 'ab' skripta

```
import subprocess
import datetime
import re

# UREDI - Pot do ukaza 'ab', pride z 'apache' streznikom
AB_PATH = "C:/wamp/bin/apache/apache2.4.17/bin/ab.exe"
# UREDI - Pot do root mape te skripte
ROOT_PATH = "E:/matja/Google Drive/Fri/Fri 2015_2016/S02/
    ZZRS/ab"

# UREDI
AB_N = "10" # <- koliko odjemalcev
AB_C = "2"  # <- koliko socasnih odjemalcev
AB_URL = "http://mavpi.ddns.net/"

AB_OUT = datetime.datetime.today().__format__("%Y%m%dT%H%
    M%S")
AB_OUT_PATH = ROOT_PATH + "/log/" + AB_OUT + ".log"
AB_CSV_PATH = ROOT_PATH + "/data.csv"

# IZVRSI AB UKAZ
AB = AB_PATH + " -n " + AB_N + " -c " + AB_C + " " +
    AB_URL + " > \"" + AB_OUT_PATH + "\""
subprocess.call(AB, shell=True)

# ODPRI DATOTEKO
ab_out = open(AB_OUT_PATH, "r+")
# PRIDOBI KLJUCNE VRSTICE
lines = ab_out.readlines()[14:25]
lines.pop(4)
ab_out.seek(0)
ab_out.truncate()
ab_out.writelines(lines)
# ZAPRI DATOTEKO
ab_out.close()

ab_csv = open(AB_CSV_PATH, "a")
# PRECISTI PODATKE
csv = []
for line in lines:
    match = re.search(r"[\d.]+", line)
    if match:
```

```
            csv.append(match.group())
csv = ",".join(csv)
ab_csv.write(datetime.datetime.today().isoformat()+","+
    csv+"\n")
ab_csv.close()

# PRIMER LOG DATOTEKE
# ===================
#
# Concurrency Level:      2
# Time taken for tests:   1.085 seconds
# Complete requests:      10
# Failed requests:        9
# Total transferred:      113696 bytes
# HTML transferred:       107606 bytes
# Requests per second:    9.21 [#/sec] (mean)
# Time per request:       217.059 [ms] (mean)
# Time per request:       108.530 [ms] (mean, across all
    concurrent requests)
# Transfer rate:          102.31 [Kbytes/sec] received
```

Literatura

[1] "Rasperry Pi." https://www.raspberrypi.org/, March 2016.

[2] "Rasperry Pi hardware." http://akizukidenshi.com/download/ds/rs/raspberrypi2b.pdf, March 2016.

[3] "LPDDR." https://en.wikipedia.org/wiki/Mobile_DDR, March 2016.

[4] "Raspbian OS." https://www.raspbian.org/, March 2016.

[5] "ab - Apache HTTP server benchmarking tool." https://httpd.apache.org/docs/2.4/programs/ab.html, April 2016.

[6] "Howto: Performance Benchmarks a Webserver." http://www.cyberciti.biz/tips/howto-performance-benchmarks-a-web-server.html, November 2008.

[7] "Apache JMeter." http://jmeter.apache.org/, April 2016.