

# Analiza zmogljivosti oblačnih in strežniških storitev

Uredil prof. dr. Miha Mraz

Maj 2019



# Kazalo

<b>Predgovor</b>	<b>iii</b>
<b>1 Testiranje zmogljivosti Amazon EC2 platforme (P. Matičič, J. Pelicon, B. Rojc)</b>	<b>1</b>
1.1 Opis problema . . . . .	1
1.2 Realizacija . . . . .	2
1.2.1 Aplikacija . . . . .	2
1.2.2 Tipi zahtev . . . . .	2
1.2.3 Amazon Web Services (AWS) . . . . .	3
1.2.4 Aplikacija . . . . .	4
1.3 Predvidevanje metrik . . . . .	4
1.4 Rezultati meritev . . . . .	4
1.5 Plan dela . . . . .	5



# Predgovor

Pričujoče delo je razdeljeno v deset poglavij, ki predstavljajo analize zmogljivosti nekaterih tipičnih strežniških in oblačnih izvedenk računalniških sistemov in njihovih storitev. Avtorji posameznih poglavij so slušatelji predmeta *Zanesljivost in zmogljivost računalniških sistemov*, ki se je v štud.letu 2018/2019 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 2019*



## Poglavje 1

# Testiranje zmogljivosti Amazon EC2 platforme

Peter Matičič, Jan Pelicon, Blaž Rojc

### 1.1 Opis problema

Med programerji je veliko takšnih, ki sanjajo o tem, bi bili naslednji Bill Gates, Mark Zuckerberg ali Steve Jobs. Imajo idejo, za katero verjamejo, da bo zavzela svet in jim prinesla milijone ter večno slavo. Ampak potrebujejo platformo, na kateri bo njihova storitev tekla. En sam prenosnik ne more vendar streči tisočem uporabnikom s celega sveta hkrati. Platforma mora biti cenovno dostopna, hkrati pa tudi poljubno razširljiva, da, ko se zgodi neizogiben naval uporabnikov, lahko programer enostavno in hitro aktivira dodatno procesno moč.

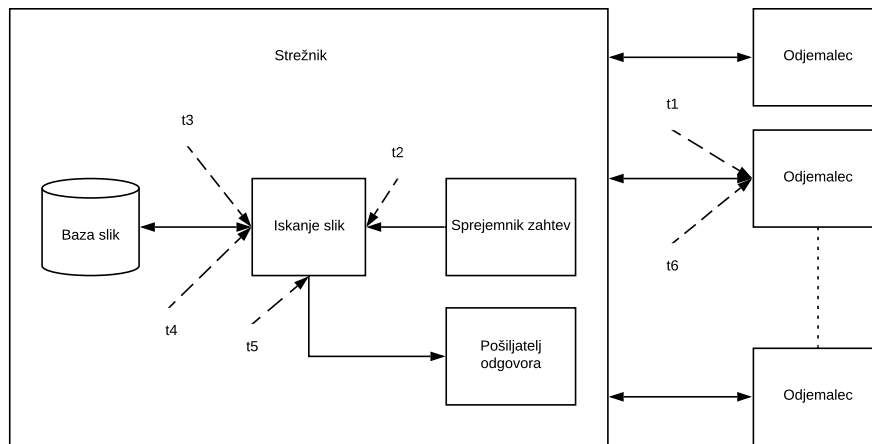
Tu vstopi Amazonov Elastic Cloud. Obljublja dostopne cene, fleksibilno alokacijo računskih virov in za nadobudnega podjetnika najpomembneje možnost uporabe določenih storitev brezplačno. Med temi storitvami je na voljo tudi najem tako imenovanih "mikro instanc". To so virtualni spletni strežniki z enim procesnim jedrom in 1 GB pomnilnika. Predstavljajo minimalno konfiguracijo, ki lahko gosti poljubno spletno storitev. Hkrati pa predstavlja procesno ozko grlo, katerega omejitve moramo upoštevati pri tvorbi storitve.

S tem v mislih želimo stestirati platformo Amazon EC2. Ustvarili bomo enostavno storitev, ki bo uporabniku omogočala iskanje vzorcev v večjem naboru slik, shranjenih v oblaku. Predstavljala bo generično spletno aplikacijo, ki potrebuje ravno dovolj računske moči, da se bodo pojavile slabosti mikro instanc, v obliki upočasnjene ali onemogočenega delovanja na strani uporabnika. Osnova storitve je iskanje vzorca v naboru slik. Uporabnik od storitve zahteva podatek o tem, v katerih slikah se ta vzorec nahaja, pričakuje pa hiter odgovor, ne več kot nekaj sekund zamika. Taka storitev nam bo omogočala re-

lativno enostavno merjenje odzivnosti platforme, modularnost nalaganja kode in morebitno razširljivost v primeru večjega števila hkratnih uporabnikov.

## 1.2 Realizacija

### 1.2.1 Aplikacija



Slika 1.1: Shema aplikacije.

Aplikacija je realizirana kot spletna storitev, t.j. strežnik, dostopen na spletu, ki se odziva na zahteve uporabnikov. Uporabnik strežniku pošlje zahtevo v obliki JSON niza, ki vsebuje zaporedno številko zahteve, podatek o tipu zahteve in potrebne parametre. Strežnik zahtevo primerno obdela in vrne rezultat v obliki JSON niza, katerega oblika je odvisna od tipa zahteve.

### 1.2.2 Tipi zahtev

V osnovi vsi tipi zahtev vključujejo iskanje vzorca v naboru slik. Razlikujejo se v tipu vzorca, ki ga uporabnik želi najti v sliki. Storitev nudi tri tipe iskanj: iskanje specifične barve piksla, iskanje vzorca s prosojnostno masko in iskanje slikovnega izseka.



## Iskanje specifične barve piksla

Storitev prejme podatke o iskani barvi piksla, preišče vsako sliko in vrne prvo pojavitev iskanega piksla.

## Iskanje vzorca s prosojnostno masko

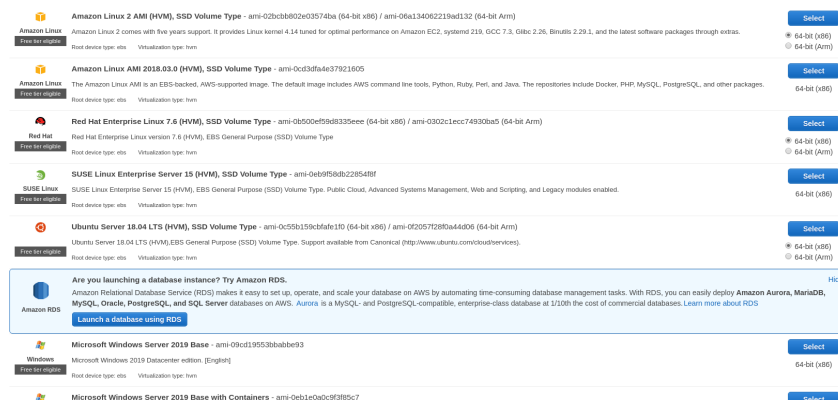
Storitev prejme slikovni izsek in pripadajočo prosojnostno masko. Med slikami najde lokacijo, ki se z izsekom najboljše ujema, in jo vrne uporabniku.

## Iskanje slikovnega izseka

Storitev prejme slikovni izsek. Preišče slike in, ko najde prvo ujemanje, rezultat vrne uporabniku.

### 1.2.3 Amazon Web Services (AWS)

Za delo z Amazon Web Services si mora uporabnik ustvariti račun v njihovem sistemu. Po uspešni registraciji si lahko ustvarimo svojo instanco EC2 storitve, ki nam jo Amazon ponuja zastoj za eno leto pri čemer na mesec porabimo največ 750 ur delovanja. Obsežnejša navodila so na voljo tudi na Amazonovi spletni strani [1], mi pa smo to naredili tako, da se postavimo v AWS Management Console, kjer lahko izberemo opcijo *Launch a virtual machine with EC2*. Takoj nam konzola ponudi izbiro slike, katero bi uporabljali na novi virtualki kot prikazano na sliki 1.2. Za naš projekt smo izbrali sliko Amazon Linux 2 AMI c.



Slika 1.2: Slike za izdelavo virtualke

V naslednjem koraku izberemo tip instance slike, to je Amazonov način izbire paketov, ki vključujejo različne funkcionalnosti. V našem primeru ker izbiramo brezplačno različico, nam ponujajo tip t2.micro, ki vsebuje 1 jedro, 1GB pomnilnika, samo začasno hranjenje podatkov na disku in nižjo hitrost povezave. Za tem lahko nadaljujemo z nastavljanjem različnih konfiguracij naše virtualke

ali pa preprosto kliknemo Review and Launch, ki nam ponudi še en pregled čez izbrane nastavitve in zažene virtualko. Po zagonu virtualke nam sistem ponudi opcijo generiranja para ključev za varno SSH povezavo do nje. Ko zaključimo z ustvarjanjem, se premaknemo v EC2 management console kjer kliknemo na *instances*. Od tam lahko opazujemo status naše storitve in pridobimo tudi naslov, na katerem se nahaja. Za povezavo uporabimo javni naslov storitve, uporabnika *ec2* – *suser* za varnost pa uporabimo *.pem* datoteko, ki smo jo v prejšnjem koraku prenesli. Ko se uspešno povežemo na storitev, lahko pričnemo z razvojem naše aplikacije.

### 1.2.4 Aplikacija

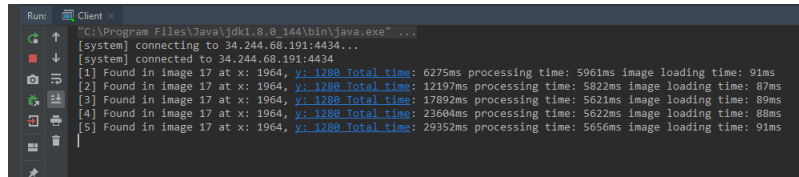
Aplikacija je napisana v programskem jeziku Java. Sestavljata jo odjemalska in strežniška komponenta, ki uporabljata skupne enumeratorje za določanje tipa zahteve. Zahteve sestavlja odjemalec in jih pošlje strežniku, ki to zahtevo obdelava - začne iskanje v slikah in pripravi prvi najden rezultat. Povezava med strežnikom in odjemalcem je trajna, dokler je eden od njiju ne prekine, kar nam omogoči, da pri meritvah ne upoštevamo časa vzpostavitve povezave. Podatki se prenašajo v obliki JSON, ki vsebuje sekvenčno številko zahteve, tip zahteve in morebitne dodatne podatke, ki so potrebni za obdelavo. Strežnik ob prejemu podatkov začne z delom na ustrezni zahtevi ter nato pošlje odgovor klientu z številko zahteve in rezultatom. Strežniški del je napisan tako, da lahko paralelno obdeluje več zahtev.

## 1.3 Predvidevanje metrik

Kot glavno metriko bomo opazovali skupni čas zahteve in odgovora. Podrobneje ga bomo razdelili na čas potovanja zahteve od nas do strežnika ( $t_2 - t_1$ ), čas obdelave na strežniku ( $t_3 - t_2$ ) in čas potovanja odgovora od strežnika do nas ( $t_4 - t_3$ ) (časi označeni na sliki 1.1). Zanima nas, kako se sistem odziva na zahteve ob različnih urah in kako se časi odgovorov podaljšajo glede na število hkratnih uporabnikov.

## 1.4 Rezultati meritev

Poskusno smo testirali iskanje specifičnega piksla. Ko en uporabnik naenkrat dostopa do storitve, je odzivni čas med 6 in 7 sekund, od tega je čas obdelave zahteve približno 5,5 do 6 sekund.



```
Run: Client
[system] connecting to 34.244.68.191:4424...
[system] connected to 34.244.68.191:4434...
[1] Found in image 17 at x: 1964, y: 1288 Total time: 6275ms processing time: 5961ms image loading time: 91ms
[2] Found in image 17 at x: 1964, y: 1288 Total time: 12197ms processing time: 5822ms image loading time: 87ms
[3] Found in image 17 at x: 1964, y: 1288 Total time: 17892ms processing time: 5621ms image loading time: 89ms
[4] Found in image 17 at x: 1964, y: 1288 Total time: 23604ms processing time: 5622ms image loading time: 88ms
[5] Found in image 17 at x: 1964, y: 1288 Total time: 29352ms processing time: 5656ms image loading time: 91ms
```

Slika 1.3: Rezultati poskusnega testiranja

## 1.5 Plan dela

- implementacija iskalnih algoritmov (1 / 3 narejenih)
- določitev bremen - čas meritev, število hkratnih uporabnikov, ... ?
- natančnejša določitev metrik
- določitev orodij - avtomatizacija merjenja, zbiranje rezultatov
- premislek o skalabilnosti
- pravilno dodajanje literature



# Literatura

- [1] “Launch a linux virtual machine with amazon ec2.” [https://aws.amazon.com/getting-started/tutorials/launch-a-virtual-machine/?trk=gs\\_card](https://aws.amazon.com/getting-started/tutorials/launch-a-virtual-machine/?trk=gs_card), Marec 2019.
- [2] “Digitalocean: Simple cloud computing for developers.” <https://www.digitalocean.com/>, Marec 2017.
- [3] G. D. Greenwade, “The Comprehensive Tex Archive Network (CTAN),” *TUGBoat*, vol. 14, no. 3, pp. 342–351, 1993.
- [4] I. Drago, E. Bocchi, M. Mellia, H. Slatman, and A. Pras, “Benchmarking personal cloud storage,” in *Proceedings of the 2013 conference on Internet measurement conference*, pp. 205–212, ACM, 2013.
- [5] “Pythonanywhere.” <https://www.pythonanywhere.com/>, Maj 2018.
- [6] “Free ebooks - project gutenber.” <http://www.gutenberg.org/>, Maj 2018.
- [7] “Aws cloud9.” <https://aws.amazon.com/cloud9/?origin=c9io>, Maj 2018.
- [8] “Ip geolocation and threat data api.” <https://ipdata.co/>, Maj 2018.
- [9] “What are cpu seconds?.” <https://help.pythonanywhere.com/pages/WhatAreCPUSeconds>, Maj 2018.
- [10] “What is a tarpit and why are my processes in it?.” <https://www.pythonanywhere.com/tarpit/>, Maj 2018.
- [11] S. Stemler, “An overview of content analysis,” *Practical assessment, research & evaluation*, vol. 7, no. 17, pp. 137–146, 2001.
- [12] A. D. Booth, “A law of occurrences for words of low frequency,” *Information and control*, vol. 10, no. 4, pp. 386–393, 1967.
- [13] W. B. Paley, “Textarc: Showing word frequency and distribution in text,” in *Poster presented at IEEE Symposium on Information Visualization*, vol. 2002, 2002.

- [14] A.-H. Tan *et al.*, “Text mining: The state of the art and the challenges,” in *Proceedings of the PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases*, vol. 8, pp. 65–70, sn, 1999.
- [15] Wikipedia contributors, “Bellard’s formula — Wikipedia, the free encyclopedia,” 2018. [Online; dostopano 2-April-2018].
- [16] Wikipedia contributors, “Leibniz formula for pi — Wikipedia, the free encyclopedia,” 2018. [Online; dostopano 2-April-2018].
- [17] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, “A performance analysis of ec2 cloud computing services for scientific computing,” in *International Conference on Cloud Computing*, pp. 115–131, Springer, 2009.
- [18] “Amazon elastic compute cloud.” <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/t2-credits-baseline-concepts.html>. [Online; dostopano 30-Maj-2018].
- [19] “Amazon web services.” <https://aws.amazon.com/>. [Online; dostopano 2-April-2018].
- [20] “Microsoft azure.” <https://azure.microsoft.com/en-gb/>. [Online; dostopano 2-April-2018].
- [21] “Aws cloud9.” <https://aws.amazon.com/cloud9/?origin=c9io>. [Online; dostopano 2-April-2018].
- [22] M. Modž̃kon and M. Mraz, *Modeliranje radž̃unalnidž̃kih omredž̃ij*. Zalođ̃ba FE in FRI, 2012.
- [23] “Amazon web services.” <https://aws.amazon.com>, Maj 2018.
- [24] “Representational state transfer.” [https://en.wikipedia.org/wiki/Representational\\_state\\_transfer](https://en.wikipedia.org/wiki/Representational_state_transfer), Maj 2018.
- [25] “node.js.” <https://nodejs.org/en/>, Maj 2018.
- [26] “Javascript.” <https://www.javascript.com/>, Maj 2018.
- [27] “Mysql.” <https://www.mysql.com/>, Maj 2018.
- [28] “Amazon api gateway.” <https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html>, Maj 2018.
- [29] “Amazon lambda.” <https://docs.aws.amazon.com/lambda/latest/dg/welcome.html>, Maj 2018.
- [30] “Amazaon rds.” <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html>, Maj 2018.
- [31] “Ping.” <https://linux.die.net/man/8/ping>, Maj 2018.

- 
- [32] “Traceroute.” <https://linux.die.net/man/8/traceroute>, Maj 2018.
- [33] “Apache http server benchmarking tool.” <https://httpd.apache.org/docs/2.4/programs/ab.html>, Maj 2018.
- [34] “Easy live auction.” <https://www.easyliveauction.com/>, Maj 2018.