

Analiza zmogljivosti oblačnih in strežniških storitev

Uredil prof. dr. Miha Mraz

Maj 2019

Kazalo

Predgovor	iii
1 Testiranje zmogljivosti Amazon EC2 platforme (P. Matičič, J. Pelicon, B. Rojc)	1
1.1 Opis problema	1
1.2 Realizacija	2
1.2.1 Opazovano okolje	2
1.2.2 Tipi zahtev	3
1.2.3 Amazon Web Services (AWS)	3
1.2.4 Aplikacija	4
1.3 Predvidevanje metrik	5
1.4 Rezultati meritev	5
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, da 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. [1] 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”. [2] To so virtualni spletni strežniki z enim procesnim jedrom in 1 GB pomnilnika. [3] 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

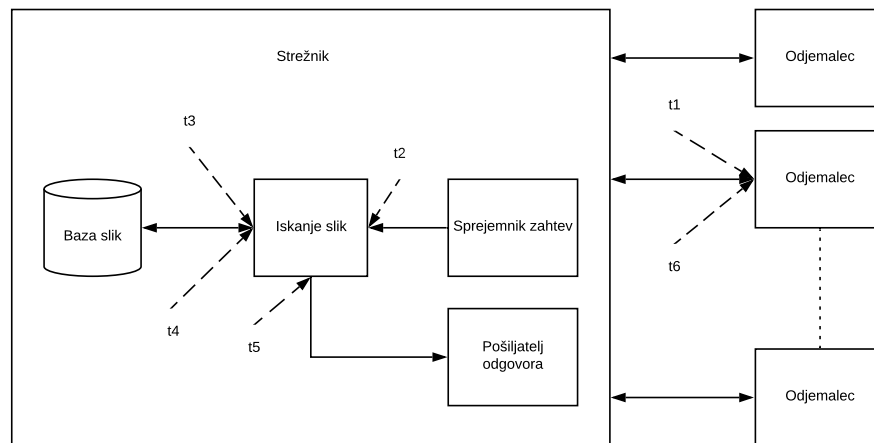
odgovor, z ne več kot nekaj sekund zamika. Taka storitev nam bo omogočala relativno enostavno merjenje odzivnosti platforme, modularnost nalaganja kode in morebitno razširljivost v primeru večjega števila hkratnih uporabnikov.

1.2 Realizacija

Storitev je sestavljena iz dveh delov, strežnika na storitvi EC2 in odjemalca na lokalnem računalniku. Opazujemo odzivnost strežnika, tako z meritvami na strežniku samem, kot pri odjemalcu.

1.2.1 Opazovano okolje

Aplikacija je realizirana kot spletna storitev, t.j. strežnik, dostopen na spletu, ki se odziva na zahteve uporabnikov. Zasnovana je po shemi v sliki 1.1. 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.



Slika 1.1: Shema opazovane storitve.

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. Storitve nudi tri tipe iskanj:

- iskanje specifične barve piksla,
- iskanje piksla, podobnega specifični barvi
- iskanje slikovnega izseka

Iskanje specifične barve piksla

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

Iskanje piksla, podobnega specifični barvi

Poleg iskane barve storitev prejme še največje dovoljeno odstopanje. Preišče vsako sliko in vrne prvi piksel, katerega barva se od iskane po komponentah razlikuje za največ toliko, kot določa odstopanje. Razlika se izračuna tako:

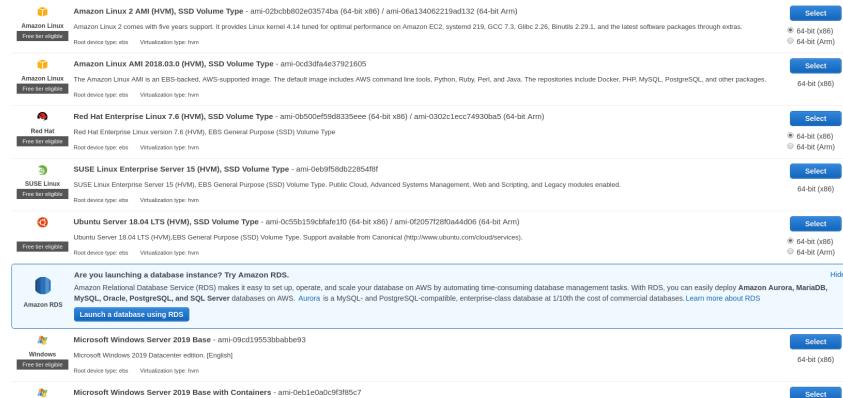
$$\begin{aligned} \text{Razlika}(RGB_{\text{piksel}}, RGB_{\text{iskan}}) &= \\ &= |R_{\text{piksel}} - R_{\text{iskan}}| + |G_{\text{piksel}} - G_{\text{iskan}}| + |B_{\text{piksel}} - B_{\text{iskan}}| \quad (1.1) \end{aligned}$$

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 [4] si moramo 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 lahko na mesec porabimo največ 750 ur delovanja ponujenih instanc. Obsežnejša navodila so na voljo tudi na Amazonovi spletni strani [5], mi pa to naredimo tako, da se postavimo v AWS Management Console [6], kjer lahko izberemo opcijo *Launch a virtual machine with EC2*. Takoj nam konzola ponudi izbiro *slike navideznega diska* - vnaprej pripravljenega nabora datotek, ki ga lahko neposredno zaženemo na instanci. [7] Izbor možnih slik diska je prikazan na sliki 1.2. Za naš projekt izberemo sliko Amazon Linux 2 AMI c.



Slika 1.2: Slike navideznih diskov 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` – `suser` pa varnost pa uporabimo `.pem` datoteko (Privacy Enhanced Mail Security Certificate), 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 Uporabljene metrike

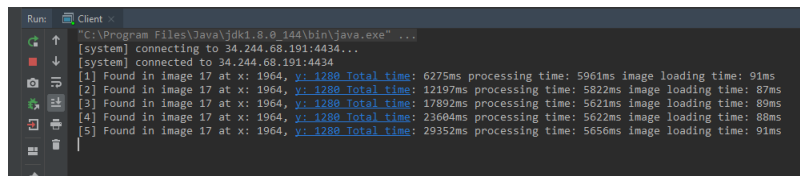
Kot glavno metriko bomo opazovali skupni čas zahteve in odgovora.

Podrobneje ga bomo razdelili na čas dostopa do datotečnega sistema (zbirke slik v mapi na datotečnem sistemu, recimo ji baza slik) ($t_{baza} = t_4 - t_3$) in celoten čas obdelave na strežniku ($t_{strenik} = t_5 - t_2$). Hkrati merimo celoten čas trajanja zahteve ($t_{zahteva} = t_6 - t_1$) (časi označeni na sliki 1.1). Z izmerjenimi časi lahko izračunamo tudi druge kot sta čas procesiranja na strežniku ($t_{procesiranje} = t_{strenik} - t_{baza}$) in čas paketa na mreži ($t_{mrea} = t_{zahteva} - t_{strenik}$). S tem smo se znebili problema sinhronizacije ur med odjemalcem in strežnikom. Če bi želeli meriti tudi čas potovanja paketa od odjemalca na strežni in čas potovanja paketa od strežnika na odjemalec, pa bi potrebovali tudi sinhronizacijo ur, ker pa naš cilj ni meriti čase potovanja paketov, saj je to namreč lastnost omrežja in ne same platforme, bomo zadovoljni s skupnim časom paketa na mreži.

Zanima nas, kako se sistem odziva na zahteve ob različnih urah. Za potrebe meritev bomo odziv sistema merili ob treh različih časih v dnevu. Želimo izvedeti tudi, 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
"C:\Program Files\Java\jdk1.8.0_144\bin\java.exe" ...
[system] connecting to 34.244.68.191:4434...
[system] connected to 34.244.68.191:4434
(1) Found in image 17 at x: 1964, y: 1280 Total time: 6275ms processing time: 5961ms image loading time: 91ms
(2) Found in image 17 at x: 1964, y: 1280 Total time: 12197ms processing time: 5822ms image loading time: 87ms
(3) Found in image 17 at x: 1964, y: 1280 Total time: 17892ms processing time: 5621ms image loading time: 89ms
(4) Found in image 17 at x: 1964, y: 1280 Total time: 23604ms processing time: 5622ms image loading time: 88ms
(5) Found in image 17 at x: 1964, y: 1280 Total time: 29352ms processing time: 5656ms image loading time: 91ms
```

Slika 1.3: Rezultati poskusnega testiranja

1.5 Plan dela

- določitev bremen - čas meritev, število hkratnih uporabnikov, ... ?
- natančnejša določitev metrik
- določitev orodij - avtomatizacija merjenja, zbiranje rezultatov
- premislek o skalabilnosti
- pravilno dodajanje literature

Literatura

- [1] “Amazon elastic compute cloud.” <https://aws.amazon.com/ec2/>, 2019. [Online; dostopano 8-April-2019].
- [2] “Aws free tier.” <https://aws.amazon.com/free/>, 2019. [Online; dostopano 8-April-2019].
- [3] “Amazon ec2 t2 instances.” <https://aws.amazon.com/ec2/instance-types/t2/>, 2019. [Online; dostopano 8-April-2019].
- [4] “Amazon web services.” <https://aws.amazon.com/>, 2019. [Online; dostopano 8-April-2019].
- [5] “Launch a linux virtual machine.” https://aws.amazon.com/getting-started/tutorials/launch-a-virtual-machine/?trk=gs_card, 2019. [Online; dostopano 8-April-2019].
- [6] “Aws management console.” <https://aws.amazon.com/console/>, 2019. [Online; dostopano 8-April-2019].
- [7] “Aws management console.” <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html>, 2019. [Online; dostopano 8-April-2019].