

02-video-notes-thursday

Uvodni dio

- Vezano za ALB (Application Load Balancer) koji proslijedjuje requestove nasim EC2 instancama.
- Nas problem je u tome sto ovaj koncept radi super kad imamo minimum jednu healthy instancu, medjutim kad ta instanca ima problem ALB nema opciju da je zamijeni i iz tog razloga koristimo `Auto Scaling grupu`.
- Auto Scaling Grupa ima mehanizam koji nam omogucava da automatski zamijenimo instancu koja je u stanju unhealthy. Prije nego pocnemo sa ASG trebali bi dotaci se Elastic Block Storage-a (EBS).

EBS (Elastic Block Storage) - osnove

- EBS predstavlja disk koji koristi nasa EC2 instanca i ima svoj zivot neovisno od EC2 instance - komuniciraju preko mreze.
- Njihova neovisnost znaci da 1 Elastic Block Storage mozemo da zakacimo na vise instanci - npr. kad terminiramo EC2 instancu mi mozemo taj EBC da zakacimo na neku drugu EC2 instancu.
- Drugi feature je da mozemo raditi `snapshots` tog EBS-a, odnosno radimo backup i pravimo novi EBS koji mozemo zakaciti nekoj nasoj EC2 instanci.

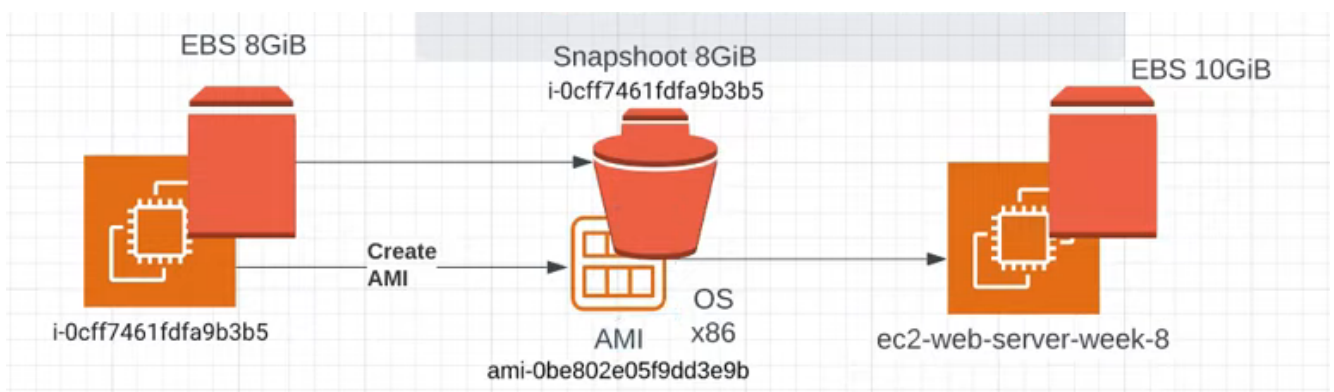
Praktican primjer rada sa EBS

- Nalazimo se u regiji `N. Virginia` na glavnoj konzoli, idemo u EC2 Dashboard - Images - AMIs, gdje pristupamo nasem AMI imageu koji smo napravili u utorak.
- Kad kliknemo na taj AMI image, vidimo njegov ID i ispod na `Storage` vidimo da on uz sebe ima zakacen snapshot `/dev/xvda`.
- Kako pristupamo diskovima i snapshotu? Sa lijeve strane na management konzoli imamo tab `Elastic Block Store` i `Volumes` gdje vidimo nase diskove, trenutno ih nemamo nijedan.
- `Snapshots` - kad kliknemo vidimo da imamo jedan snapshot koji je kreiran u trenutku kad je kreiran nas AMI image.
- Sta smo prije uradili? Kad smo imali prethodno kreiranu EC2 instancu, ona je imala svoj EBS. Zatim kreirali smo AMI od te instance, u tom trenutku se kreirao AMI image koji pored podataka o OS-u sadrzi i snapshot EBS-a koji je bio zakacen za nasu EC2 instancu; Kada smo rekli da zelimo podignuti novu EC2 instancu od naseg AMI imagea, ovaj snapshot se pretvorio u EBS.

Kreiranje nove instance od postojećeg AMI-a.

- Prethodno spomenut proces u praksi - trenutno imamo svoj kreiran AMI image - kad pogledamo u `Volumes` vidimo da nemamo nijedan disk, medjutim kad odemo u `Images - AMIs`, kliknemo na svoj AMI i kliknemo na `Launch Instance from AMI`:

- Name: ec2-web-server-week-8
- Key pair (login): odabrati ključ koji smo koristili prije - week-8
- Select existing security group - web-server-week-8
- Configure storage - 8 GiB, gp3.
- Možemo primijetiti da imamo unaprijed upisanu veličinu našeg diska - koja je upisana na osnovu veličine od koje je kreiran naš snapshot. Znači kad smo kreirali AMI image ovaj EBS je bio velik 8 GiB pa samim tim i ovaj snapshot je velik toliko. I šta sad mi možemo uraditi?
- Kad kreiramo novi disk od ovog snapshot-a, mi možemo odabrati da kreiramo novi Elastic Block storage koji će biti iste te veličine Ili možemo povećati ovaj disk jer želimo veći EBS npr. od 10 GiB, i kliknuti ćemo na `Launch Instance`.
- Možemo otvoriti EC2 instancu i kliknuti dole u dio Storage, zatim vidimo da se pojavio `Volume ID` koji ima veličinu 10 i u statusu je Attaching.
- Ako sa lijeve strane u konzoli kliknemo na dio `Volumes` vidjet ćemo da imamo taj jedan volume i vidimo od kojeg snapshota je kreiran taj volume (pogledati Snapshots - Snapshot ID). Snapshot u opisu sadrži od koje instance je kreiran.



Spajanje na novokreiranu EC2 instancu

- `ssh -i "week-8.pem" ec2-user@nova-javna-ip-adresa`
- `sudo su -` - da se prebacimo na root
- `lsblk` - da provjerimo particije zakacene na naš disk
 - Vidimo da se veličina našeg diska podudara sa zadanom veličinom u konzoli.
- EBS ima opciju koja nam omogućava da mijenjamo veličinu našeg diska. Recimo da smo popunili ovih 10 GB, u konzoli `Elastic Block Store - Volumes` označimo naš volume i kliknemo na `Actions - Modify volume` i size: 25 GB. Kliknemo na Modify. Primijetite sad Volume state - In-use-modifying.

Pricing EBS servisa

- [Documentation](#)
- Kako znamo koliko će nas AWS naplaćivati za disk? Izaberemo Region: N. Virginia u kojem se trenutno nalazimo.

- IOPS se naplacuje, citanje i pisanje sa diska itd.
- Pravljenje snapshota nije besplatno. Snapshote koje smo arhivirali cemo placati manje.

Dok se nas disk modifikuje

- mi cemo pripremiti nesto drugo.
- U nasem terminalu, u SSHiranoj EC2 instanci kada ponovo unesemo `lsblk` vidimo da je povecan storage na 25 GB medjutim mi trenutno koristimo samo 10 GB zato sto nasa particija ovdje nije povecana na max. velicinu diska. Da bismo to odradili moramo da izvorsimo par komandi.
- Proguglati: Extend a Linux file system after resizing a volume - <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/recognize-expanded-volume-linux.html>
- Primijetimo prvu komandu koja koristi AWS CLI da bismo vidjeli Hypervisor koji koristimo za nasu EC2 instancu, odnosno tip virtualizacije koji koristimo za nasu EC2 instancu.
 - `aws ec2 describe-instance-types --instance-type instance_type --query "InstanceTypes[].Hypervisor"`
- Taj tip virtualizacije mozemo pogledati i ovdje u dijelu `Instances` - kada selektujemo nasu instancu - u opisu i detaljima ispod mozemo vidjeti `Virtualization type - hvm`.
- Ako zelimo da vidimo isto kroz CLI, mozemo da otvorimo cloud shell (ikonica pored menija gdje biramo regiju), i unosimo gore navedenu komandu sa tim da je modifikujemo - proslijedimo tip instance koji koristimo pod `instance_type`:
 - `aws ec2 describe-instance-types --instance-type t2.micro --query "InstanceTypes[].Hypervisor"`
 - output je `xen` virtualizacija - koja je generalno virtualizacija za EC2 instance, medjutim prilikom kreiranja instance mi smo odabrali drugi vid virtualizacije - tako da zanima nas konkretno za ovu nasu instancu, i moci cemo ovdje da izmijenimo ovu komandu:
 - `aws ec2 describe-instance-types --instance-ids x-xxxxxxxxxxxxxxxxx --query "Reservations[*].Instances[*].VirtualizationType"`
 - `x-xxxxxxxxxxxxxxxxx` je instance ID koji unosimo.
 - Output ce biti `hvm`.

xen virtualizacija - digresija

- AWS u svom datacentru kad ima neki fizicki server na njemu se nalazi sloj virtualizacija - xen, i onda na tom sloju virtualizacije imamo svoju instancu koja na njemu sjedi.
- HVM - hardware virtual machine.

kako napraviti query koristeći jq za parsiranje

- Nas gore prethodno navedeni query koristi `jq` za parsiranje.

- Znaci Dzenan objasnjava JSON kod koji se sastoji od niza u kojem imamo odredjene atribute i na koji nacin da imamo output za taj atribut, tj. ulazi u kompletan niz preko tacke:

```
Reservations[*].Instances[*].VirtualizationType
```

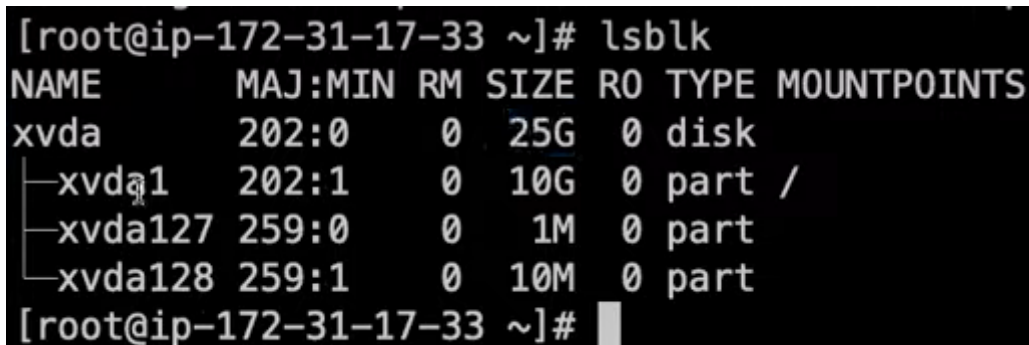
- Kada radimo ovaj `jq` query i kad imamo neki JSON kod, ovaj Reservations obuhvata sve unutar tog JSON-a, Instances obuhvata sve ono sto se nalazi unutar dijela za instancu, pa onda imamo parametar koji zelimo da izvucemo iz toga - VirtualizationType.

drugi primjer na kodu preko koristenja pipe-a

- Drugi primjer kako odraditi query na kodu je od 27:30m - 29:49m
- Koristeci pipe preusmjeravamo kompletan output komande aws cli preko jq-a u nasem terminalu - ne AWS konzoli:
 - `aws ec2 describe-instance-types --instance-type t2.micro --output json | jq "InstanceTypes[].FreeTierEligible"`
 - Ova komanda gore nije radila jer moguće da Dzenan nije imao instaliranu jq biblioteku koja se koristi za parsiranje JSON odgovora na svom Macu. Mozemo provjeriti je li biblioteka instalirana pokretanjem naredbe `jq -V` u terminalu. Za instalaciju jq mozemo provjeriti sluzbenu stranicu jq.
- Uglavnom `jq` alat mozemo koristiti za parsiranje JSON fajla da iz jednog JSON-a uzmemo vrijednost kljuca koji nam treba.

Vracamo se nazad na nas disk

- U nasoj AWS konzoli u `Volumes` - vidimo da je nas Volume state in-use, sto znaci da sad ga mozemo koristiti.
- Ono sto je zanimljivo, dok smo radili prosirenje naseg diska, u pozadini smo mogli koristiti nas disk u kapacitetu u kojem smo ga imali na pocetku, odnosno, u kapacitetu onih 10 GiB, medjutim nismo mogli da koristimo ovih 25 GiB dok on nije u statusu in-use.
- Sad, SSHirani u terminalu na nasu instancu, treba da odradimo povecanje storage-a na nasoj instanci:
 - `sudo su -` za root.



```
[root@ip-172-31-17-33 ~]# lsblk
NAME        MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
xvda        202:0    0  25G  0 disk
├─xvda1      202:1    0  10G  0 part /
├─xvda127    259:0    0   1M  0 part
└─xvda128    259:1    0  10M  0 part
[root@ip-172-31-17-33 ~]#
```

- `lsblk` - vidimo da je 25 GB medjutim mi moramo ovu particiju `xvda` koja je zakacena za nas disk da je prosirimo na taj kapacitet

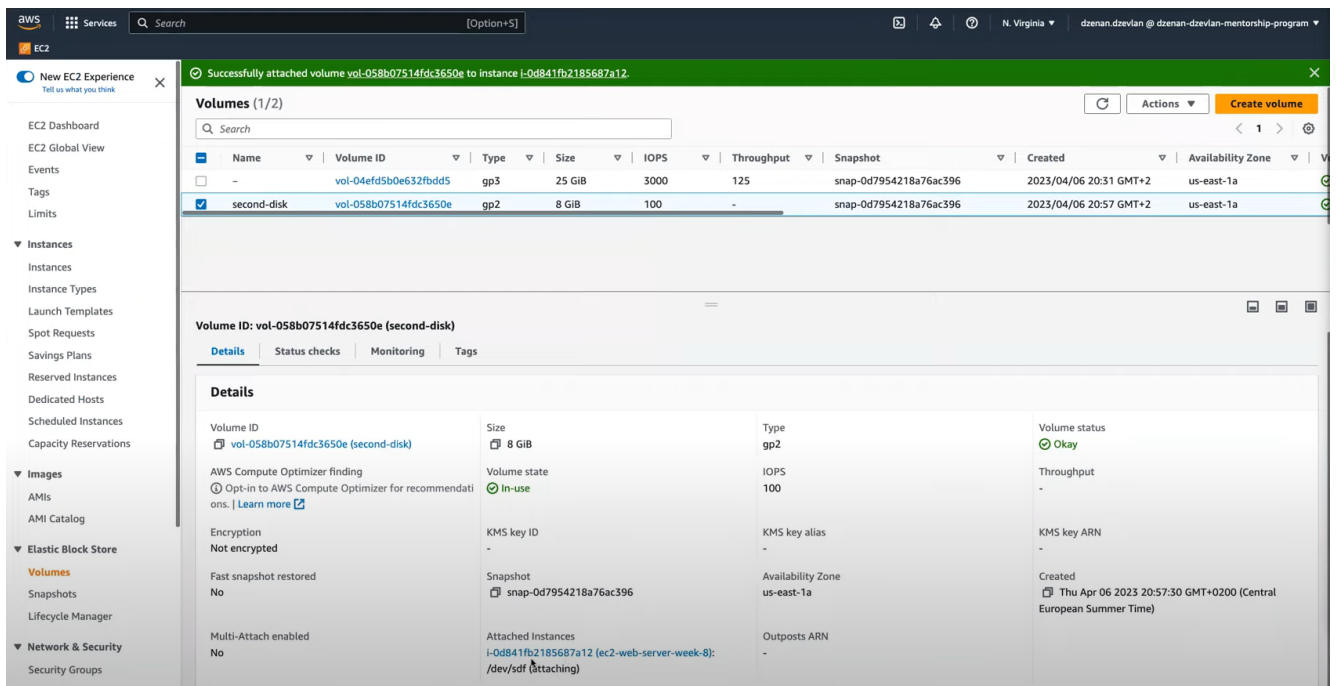
- Ponovo se vracamo na dokumentaciju "Extend the file system of EBS volumes" da vidimo sta kaze dokumentacija. Znaci mi koristimo `xen instance example` a ne Nitro instance example.
- U dokumentaciji stoji da odradimo `growpart`
- `growpart /dev/xvda1` - bez sudo jer vec gore smo usli u sudo mode.

```
[root@ip-172-31-17-33 ~]# sudo growpart /dev/xvda 1
CHANGED: partition=1 start=24576 old: size=20946911 end=20971487 new: size=52404191 end=52428767
[root@ip-172-31-17-33 ~]#
```

- Kad odradimo `lsblk` da potvrdimo, vidimo da je nasa particija 25GB.

Kreiranje novog volume-a

- Sta jos mi mozemo odraditi? Mozemo otici u dio `Elastic Block Store - Snapshots` i kliknuti na nas snapshot tj. oznaciti ga pored Name-a, kliknuti na `Actions - Create Volume from snapshot` da kreiramo jos jedan volume iz vec postojeceg snapshot.
 - Volume type: gp2
 - Size: 8
 - Availability zone: us-east-1a (mora biti ista u kojoj se nalazi nasa EC2 instanca za koju planiramo da zakacimo ovaj volume)
- Snapshot i volume moraju biti u istoj availability zoni u kojoj se nalazi EC2 instanca da bi mogli da je zakacimo.
 - Tag key - name; value: second-disk
- Kliknemo na `Create volume`.
- Ili mozemo kreirati image iz ovog snapshot-a - `Create image from snapshot` - Ako izaberemo ovu opciju automatski ce da se kreira AML image od ovog snapshot-a ali mi cemo ici na prvu opciju gore - Create volume from snapshot.
- Sada se kreira novi volume i kada odemo u dio `Elastic Block Store - Volumes` vidimo da imamo prvi volume od 25 GB koji je root volume nase EC2 instance i drugi ovaj disk koji nije zakacen niti za jednu instancu. Mozemo kliknuti da oznacimo ovaj nas volume i ici na `Actions - Attach volume`:
 - Instance: izaberemo instancu za koju zelimo zakaciti ovaj volume
 - Device name: /dev/sdf po defaultu jer je to data volume, root volume ima naziv /dev/sda1
 - Kliknemo na `Attach volume`
- Sad mozemo ovdje vidjeti za koju EC2 instancu je zakacen taj volume pod `Attached Instances` u `Details` section ispod Volumes.

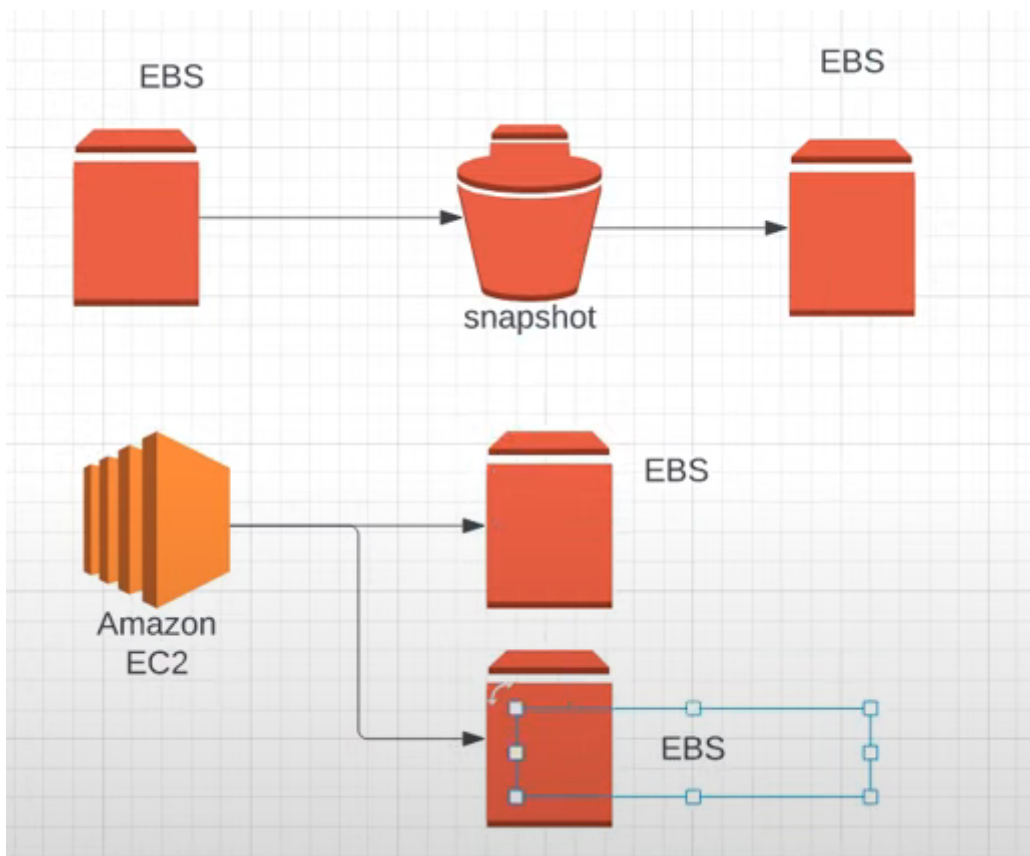


dalje u terminalu

- U terminalu, ako se vratimo nazad na svoju SSHiranu instancu i kazemo:
 - `lsblk`
 - vidimo da imamo zakacena dva diska za ovu instancu `xvda` i `xvdf`. Sad mi mozemo napraviti neki zapis na disku i otkaciti taj disk, te zakaciti ga na neku drugu instancu.

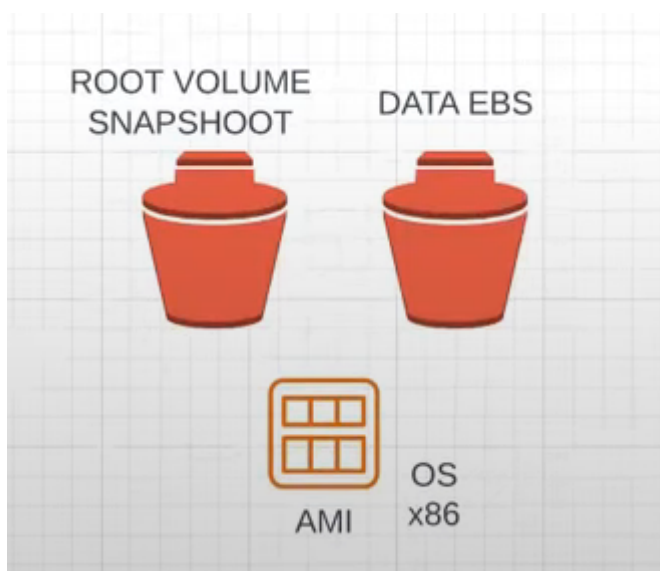
summary - sta smo prosli do sada:

- Pokazali smo da kada imamo jednu EC2 instancu, kako mozemo kreirati Elastic Block Storage (EBS) iz volume-a iz snapshot-a i zakaciti volume 1 i volume 2 - sto znaci da sada imamo dva Elastic Block Storage-a zakacena za nasu instancu.



Kreiranje AMI od instance koja ima zakacena 2 diska

- Mozemo vidjeti sta ce se desiti kada kreiramo AMI image od instance koja ima zakacena dva diska.
- Idemo na `Instances` - kliknemo na nasu instancu i vidimo da ima zakacena dva diska, kliknemo na `Actions - Image and templates - Create image`:
 - Image name: ami-ec2-tier-1-group-2
 - Mozemo izabrati da li zelimo da kreiramo nas AMI koji ce imati oba ova diska ili zelimo da ovaj drugi disk jednostavno otkacimo. Uvijek mora imati zakacen root volume - to je onaj EBS koji sadrzi OS i sve one podatke potrebne da bi instanca radila.
- Kako bi izgledao taj novi image sa dva snapshot-a:



- Ovo je veoma korisno da se zna jer mozemo doci u situaciju da imamo problem sa diskom - da napravimo snapshot tog diska i da onda mozemo da podignemo EC2 instancu kompletnog tog diska.

Sta smo prosli do sada:

```
You, 1 second ago | 1 author (You)
## Week-8

You, 10 hours ago | 1 author (You)
#### Group-1-Tier-1 | 04.04.2023.
- ALB i Target Groups - kako se podesava, kako se povezuje sa EC2 instancama

You, 1 second ago | 1 author (You)
#### Group-1-Tier-2 | 06.04.2023.
- Nastavak predavanja o AWS servisima od Utorka gdje smo podigli EC2 instance sa jednostavnom html stranicom i postavili ih iza Application Load Balancer (ALB) servisa. Demonstrirali smo HA izmedju razlicith AZ. Veceras cemo obraditi:
- [ ] Elastic Block Storage (EBS)
  - [ X ] Snapshots
  - [ X ] Create Volume from Snapshot
  - [ X ] Create AMI from Volume
  - [ X ] Expand Volume
  - [ X ] Attach Volume to EC2 instance
  - [ x ] Attach Multiple Volumes to EC2 instance
- [ ] AutoScaling Groups (ASG) - kako se podesava, kako se povezuje sa ALB-om, kako se povezuje sa EC2 instancama
- [ ] ELB and AutoScaling Groups Pricing
- [ ] Podesavanje SSL sertifikata na nasu EC2 instancu gdje koristimo SSL let's encrypt sertifikat
- [ ] Prebacivanje SSL sertifikata sa EC2 instance na ALB servis koristeći AWS Certificate Manager (ACM) servis
```

Auto Scaling Groups

- Kad se vratimo nazad, znamo da nas ELB nema mehanizam da bi automatski dodao novu EC2 instancu ukoliko obje ove EC2 instance imaju problem.
- Digresija - when expanding volume we also need to resize it - Dzenan je spomenuo da je to vec odradjeno osobi koja je postavila pitanje, medjutim provjerio je i disk je i dalje bio 10 GB sto znaci da nismo dovrшили taj dio.
 - Izvrsili smo: `df -hT` da provjerimo podatke za diskove.
 - dalje: `sudo xfs_growfs -d/` to grow file system.
- Poslije ovog se povecao size sa 10 na 25 GB.
- What is Amazon EC2 Auto Scaling? - proguglati aws docs.

Prvo

- terminiramo instancu koju smo maloprije kreirali sa ova njena dva diska.
 - `Instance State - Terminate`
- I dalje imamo onaj AMI image koji smo prije kreirali i kada odemo u `Volumes` vidimo da smo kreirali i `second-disk` ali on nam vise ne treba i onog trenutka kad se EC2 instanca terminira ovdje ce pisati da nije u upotrebi i onda cemo ocistiti ovaj volume. Vrlo je vazno da cistite resurse koje smo kreirali koji nam vise ne trebaju.

Drugo

- Dok se ovo gore ne terminira, vracamo se na kreiranje Auto Scaling grupe - u AWS management konzoli na lijevoj strani idemo ispod na `Auto Scaling` grupe, i kazemo `Launch Configurations` i da bismo kreirali Auto Scaling grupu prvo cemo da napravimo template kako ce ta nasa ASG da izgleda.
- Prvi dio je kreiranje template-a i kliknemo na `Create launch configuration` te zatim kliknemo na `Create Launch Template` da kreiramo template (45:30 za vise detalja). Unosimo:
 - Name: asg-template-web-server
 - description: Template for ASG used for Web Server
 - Template tags: Key - CreatedBy; Value - Ime Prezime; Key - Environment; Value: Development
 - Application and OS Images (AMI image): Owned by me: ami-week-8
 - Instance type: t2.micro
 - Key pair name: week-8
 - Security groups: web-server-week-8
 - EBS Volumes: imamo onaj volume od ranije - svi podaci su unijeti po defaultu.
 - Zatim kliknemo na `Create launch template`
- Sada vidimo da se kreirao ovaj nas launch template.

kreiranje ASG iz prethodno kreiranog template-a

- Preduslov za kreiranje Auto Scaling grupe je da imamo prethodno kreiran template.
- Sta nam treba da bismo kreirali ASG? Idemo na `Auto Scaling` i zatim kliknemo na `Create Auto Scaling group`:
 - Name: asg-web-servers
 - launch template: asg-template-web-server
 - Version - biramo verziju templatea jer mozemo da ga editujemo i da ima vise verzija sto cemo kasnije pokazati.
 - Sljedeci korak je `Next`.
 - Biramo defaultni VPC i biramo subnet - tj. selektujemo 1a i 1d availability zone.
- Digresija - Sta je ideja za ASG? Mi hocemo da nasa Auto Scaling Grupa dinamicki kreira nase resurse u vise availability zona kako bismo osigurali multi AZ availability.
- Nase instance moraju biti u minimalno dvije availability zone - mi cemo izabrati us-east-1a i us-east-1d zone.
 - Instance type requirements - mozemo izabrati override ali necemo trenutno, kliknemo na `Next`.
 - Sljedeća stvar je da li nasa Auto Scaling Grupa ima Load Balancer? Mi mozemo da je zakacimo na postojeći load balancer ili mozemo da kreiramo novi load balancer. U ovom

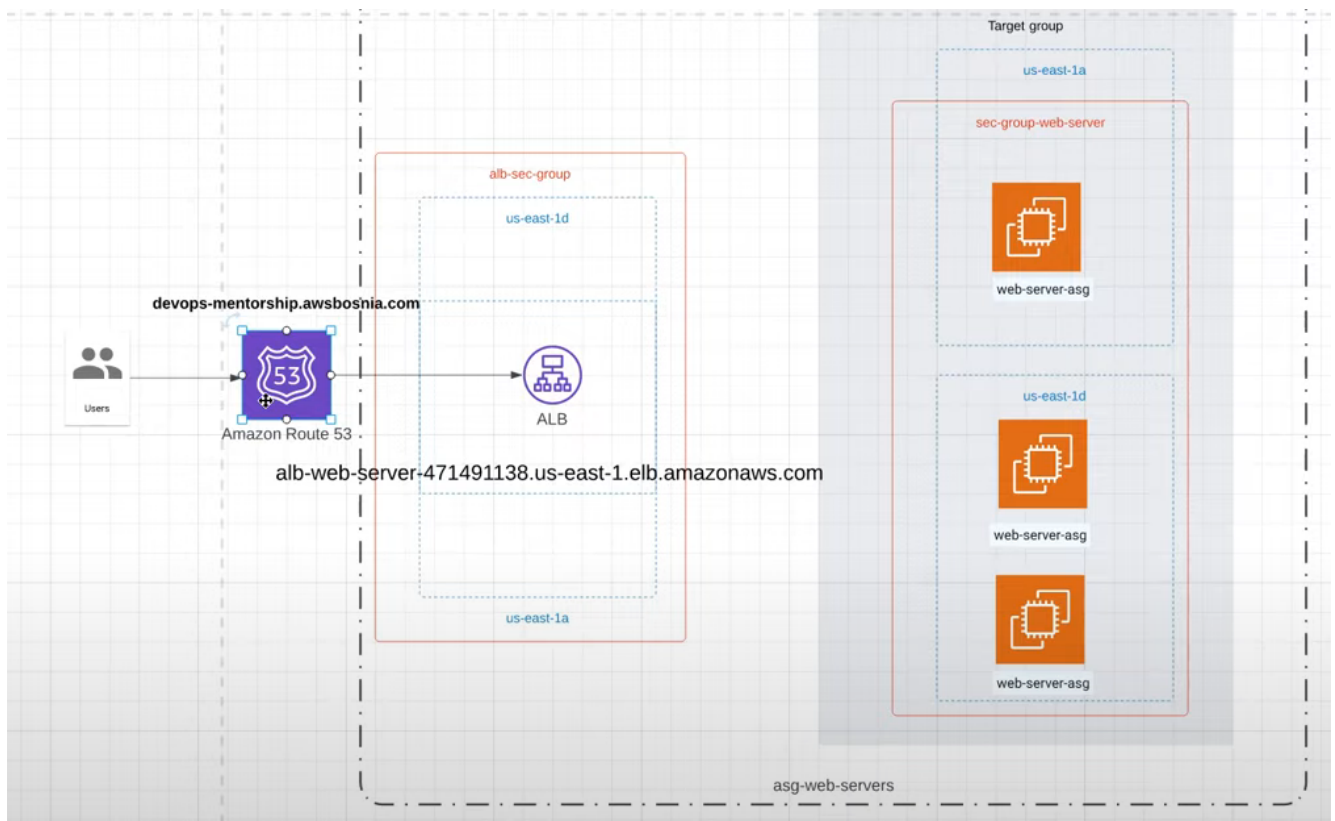
slucaju cemo zakaciti nas postojeći Application Load Balancer.

- U ovom slucaju Dzenan ima uklonjen ALB tako da ce praviti novi. Ostavio je otvoren tab sa kreiranjem ASG-a i vratio se nazad na EC2 Dashboard gdje je kreirao ALB isti kakav je imao prosli put. Ista procedura za kreiranje load balancera kao prije.
- Digresija - sta se desava unutar AWS racuna - `CloudTrail logovi`, na event history imamo kompletnu historiju o tome sta je neki korisnik uradio.
 - Dalje, izaberemo target grupu i automatski kad smo izabrali target grupu ovdje ne trebamo da biramo load balancer zato sto je target grupa zakacena za load balancer jer postoji veza 1:1 izmedju ta dva servisa.
 - Mi mozemo koristiti istu ASG (Auto Scaling Grupu) za vise load balancer-a odnosno vise target grupa, tako da smo mogli ovdje dodati jos target grupa. To je van danasnjeg scope-a ali samo trebamo imati na umu da postoji ova opcija.
- Digresija - `VPC Lattice` - taj servis cemo obraditi kad budemo pricali o VPC-u detaljnije.
 - Health checks - mozemo aktivirati health checkove da nasa scaling grupa koristi health checkove za load balancere (Turn on Elastic Load Balancing health checks).
 - Additional Settings - kliknuti Enable group metrics collection within CloudWatch.
 - kliknuti `Next`
- Configure group size and scaling policies - optional
 - Desired capacity je broj instanci koji ce biti kreiran u trenutku kada mi kreiramo nasu ASG.
 - Minimum capacity je kapacitet koliko minimalno nasa ASG moze skalirati prema dole.
 - Maximum capacity je kapacitet koliko maksimalno nasa ASG moze skalirati prema gore.
 - Setujemo desired: 3, Minimum: 2; Maximum: 4.
 - Minimum od 1 nije preporuceno jer to znaci da nasa ASG moze spasti na 1 instancu i ako se nesto desi sa tom instancom mi cemo u jednom trenutku biti bez instanci i nasa app ce biti nedostupna. Sto znaci da bismo osigurali visoku dostupnost mi u svakom trenutku moramo imati minimum 2 instance.
 - Scaling policies - Target tracking scaling policy
 - Target value: 50 - svaki put kad se CPU poveca iznad 50 mi zelimo da podignemo novu EC2 instancu.
 - Instanced need - preskacemo za sad.
 - Kliknemo na `Next`.
- Add notifications - optional
 - Kliknemo na Add notification
 - Create a topic
 - Send a notification to: asg-sns-notifications
 - with these recipients: dzenan.dzevlan@gmail.com

- Event types - selektujemo sve.
- Kliknemo na `Next`.
- Tagovi
 - Add tag - Key: Environment; Value: development; Key: Name; Value: web-server-asg
 - Kliknemo na `Next`
- Review
 - Sada mozemo da pregledamo sta smo napravili ovdje za ASG.
 - Kliknemo na `Create Auto Scaling group`

Sta se desava u pozadini - dijagram ASG

- U pozadini imamo sljedecu situaciju, prvo na `EC2 - Auto Scaling groups` imamo `asg-web-servers` grupu gdje pise Updating capacity - odnosno kreiraju se resursi u pozadini.
- Otvoriti cemo nas EC2 Dashboard i vidjeti da se kreiraju tri EC2 instance koje su bile desired capacity.
- Znaci, pored ALB-a, target grupe i instanci, mi sad imamo nesto sto se zove Auto Scaling group. Ova ASG obuhvata i ovaj ALB. Zasto? Zato sto ovaj ALB pripada ovoj ASG.
 - Dvije instance imamo u availability zoni 1d, jednu u 1a.
 - Update DNS name-a preko Route 53 - Dzenan odradio na svom drugom accountu gdje ima domain name.



Auto Scaling policy u praksi (od 1:17:00)

- Imamo pokrenute EC2 tri instance gdje se nalazi nasa aplikacija. Ako odemo u dio za ALB i pogledamo target group tog ALB vidimo da imamo 3 healthy instance.
- Na EC2 Dashboard - Auto Scaling groups - asg-web-servers vidimo da imamo 3 podignute EC2 instance.
- Sad je neophodno da u nasem email-u potvrdimo subskripciju na sve mejlove koje smo dobili jer u startu smo odradili pretplatu na notifikacije od nase EC2 instance.
- Na `EC2 - Auto Scaling groups - asg-web-servers` mozemo vidjeti detalje nase Auto Scaling grupe.

asg-web-servers

Details | Activity | Automatic scaling | Instance management | Monitoring | Instance refresh

Group details Edit

Auto Scaling group name asg-web-servers	Desired capacity 3	Status -	Amazon Resource Name (ARN) <code>arn:aws:autoscaling:us-east-1:769312778453:autoScalingGroup:e3a18ea9-121a-49d2-b117-25e7a9d7208f:autoScalingGroupName/asg-web-servers</code>
Date created Thu Apr 06 2023 21:29:09 GMT+0200 (Central European Summer Time)	Minimum capacity 2		
	Maximum capacity 4		

Launch template Edit

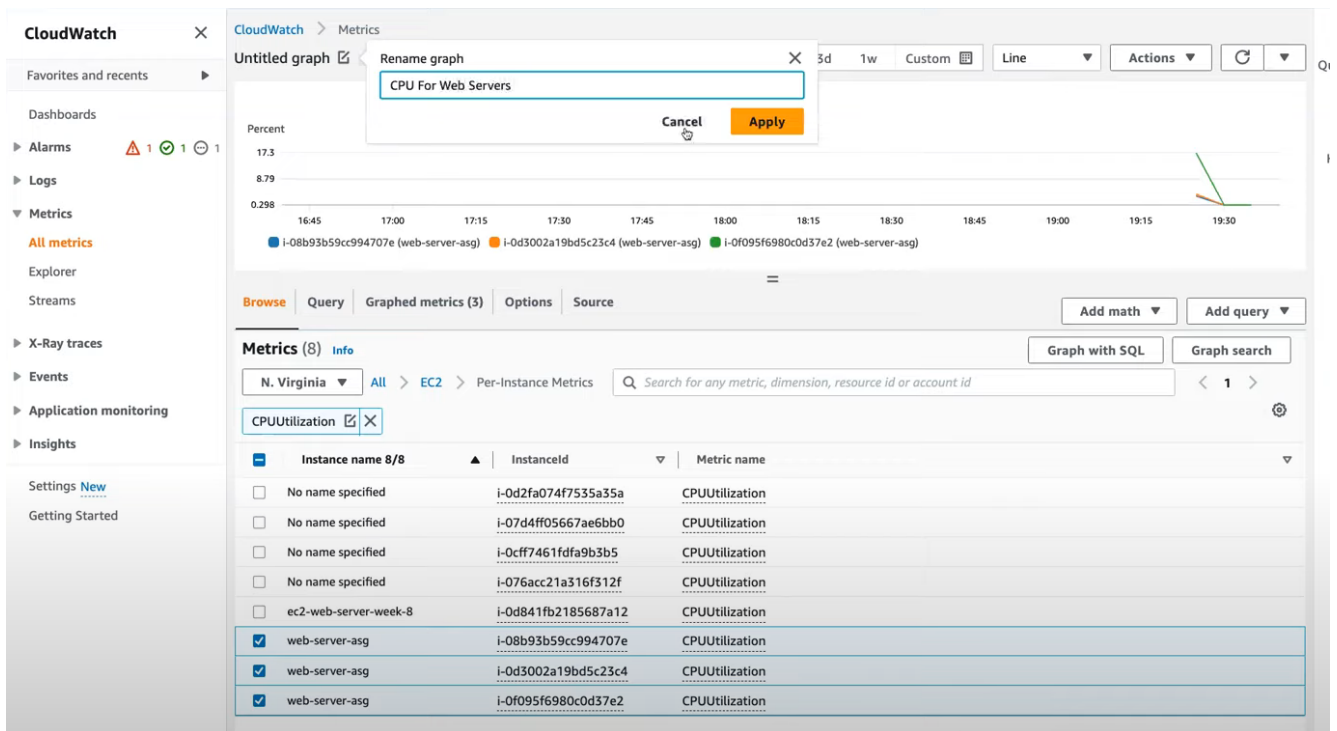
Launch template <code>lt-009bf8e18e35b8a1c</code> asg-template-web-server	AMI ID <code>ami-0be802e05f9dd3e9b</code>	Instance type t2.micro	Owner <code>arn:aws:iam::769312778453:user/dzenan.dzevla</code>
Version Default	Security groups -	Security group IDs <code>sg-0300ca0c384d5aa00</code>	Create time Thu Apr 06 2023 21:11:28 GMT+0200 (Central European Summer Time)
Description Template for ASG used for Web Server	Storage (volumes) -	Key pair name week-8	Request Spot Instances No

[View details in the launch template console](#)

- Desired 3, Minimum 2, Maximum 4 kad su u pitanju kapaciteti.
 - Kada ce porasti Max. kapacitet? Onda kada nas Auto Scaling policy registrira da se dogodilo povecanje diska. (Automatic Scaling - Dynamic scaling policies) - As required to maintain Average CPU utilization at 50 - sto znaci kad je preko 50 nas Auto Scaling ce porasti.

Cloudwatch

- Kad se vratimo na `CloudWatch` servis pod `All metrics` - u search kucamo `cpu` i selektujemo `EC2 > Per-Instance Metrics` i dalje u search kucamo `CPUUtilization` i selektujemo nase tri instance (3 x web-server-asg).
- Rename-at cemo graph na vrhu - "CPU For Web Servers"



- kliknuti cemo na vrijeme `1h` da vidimo kako izgleda u prethodnih 1 sat.
- Demo: podizanje CPU upotrebe na jednoj od instanci.
 - Iskoristimo njenu javnu IP adresu da se SSHiramo i pokrenemo `top` gdje vidimo da je CPU na nuli. Treba nam neki alat koji ce izazvati povecanje CPU-a na instanci tako da mozemo kucati `yum install stress` da instaliramo taj stress utility za instancu.
 - https://wellarchitectedlabs.com/performance-efficiency/100_labs/100_monitoring_linux_ec2_cloudwatch/5_generating_load/
 - ```
sudo stress --cpu 8 --vm-bytes $(awk '/MemAvailable/{printf "%d\n", $2 * 0.9;}' < /proc/meminfo)k --vm-keep -m 1
```
  - Vracamo se nazad na CloudWatch da vidimo kako load utice na CPU. Auto Scaling bi trebao da se prilagodi i kreira novu instancu.
- Vracamo se na Auto Scaling groups (EC2 - Auto Scaling groups - asg-web-servers) i vidimo ispod na aktivnostima na Activity history da prvo smo se vratili sa 3 (Desired) instance na 2 zato sto nije bilo potrebe zbog opterecenja da imamo 3.

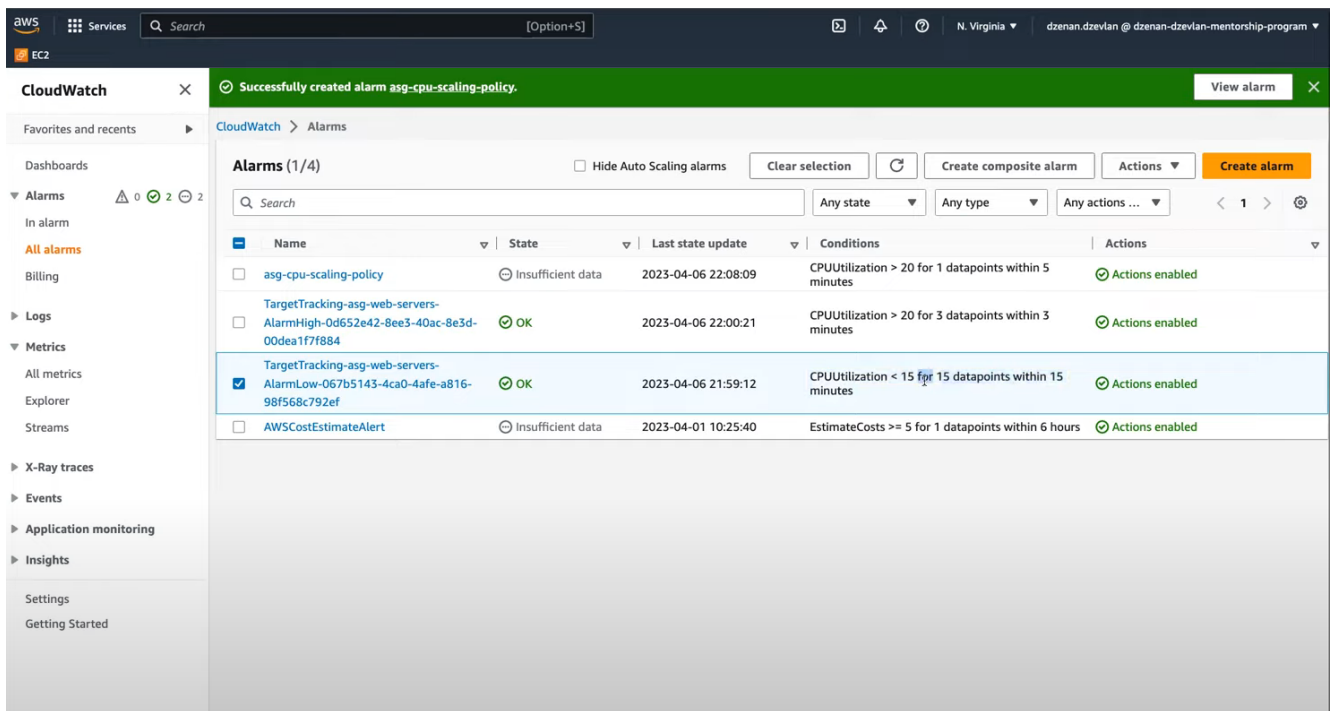
## Editovanje Target Policy-a u ASG

- Zbog `rustable` CPU-a, 50 nam je puno sto se tice opterecenja.
- U dijelu `EC2 - Auto Scaling groups - asg-web-servers` trebamo da izmijenimo nas Target Tracking Policy, oznaciti cemo ga i kliknuti na Actions - Edit gdje cemo reci da je Target value: 20 i kliknuti cemo na `Update`.
- Sta to znaci? Kad opteretimo nasu EC2 instancu sa CPU loadom preko 20, ASG ce kreirati novu EC2 instancu tj. ASG ce skalirati prema gore.
- Ispod na Activity history vidimo da se automatski pocela lansirati nova EC2 instanca. Mozemo provjeriti notifikacije unutar naseg mejla za termination i launch.

- U `Auto Scaling Documentation` na AWSu - kliknemo na User Guide - PDF da vidimo PDF dokument.
- Vratimo se nazad u target grupu (tg-alb-web-server) i vidimo da imamo 4 instance - dosegli smo max. kapacitet nase ASG.

## Dijagram samo za ASG (1:38:00)

- Objasnjenje vezano za Desired, Minimum i Max. capacity sa CPU Autoscaling policy od 20.
- Bitno da znamo da mozemo raditi fine tuning ove nase Auto Scaling grupe (EC2 - Auto Scaling group - asg-web-servers). Da bismo mogli pravilno da podesimo nasu ASG moramo da znamo kako se nasa aplikacija ponasa, u smislu kakav je load na nasoj aplikaciji i na osnovu kojih parametara cemo da skaliramo nasu aplikaciju.
- U `Automatic Scaling` dijelu - ako nasa aplikacija koristi samo mrezne resurse, onda mi necemo ovdje koristiti policy za CPU utilization nego network utilization. Ili ako je memorija problem onda cemo koristiti memory utilization instead of CPU ili network utilization.
- Da iskljucimo warm-up period: Actions - Edit - Instances need `0` seconds warm up before including in metric.
- Na `Create dynamic scaling policy` mozemo izabrati policy type - simple scaling, policy name unosimo `asg-policy` pa onda mozemo kreirati CloudWatch alarm - kliknemo na CreateCloud Watch alarm. Select Metric - CPUUtilization - By Auto Scaling Group.
- kliknemo na CPUUtilization - `Select metric` button - Specify metric and conditions - Greater than: 20 i kliknemo na `Next button`.
- Configure actions - Notification - Send a notification to `asg-sns-notifications` - email endpoint bi trebao da se pojavi nas mejl.
- Auto Scaling action - add auto scaling action (bitno da se prethodno kreira simple policy - sto smo odradili gore u prethodnim koracima).
  - Resource type: EC2 Auto Scaling group
  - Select a group: asg-web-servers
  - Take the following action - asg-policy
  - kliknemo na `Next`.
- Add name and description
  - Alarm name: asg-cpu-scaling-policy
  - Kliknemo na `Next` - `Create alarm`.



- Kada pogledamo gore - vidimo da nismo dobro podesili vrijeme vezano za downscaling, trebalo bi manje vremena da prodje da bi nas Auto Scaling policy radio.
  - Vracamo se na nasu ASG (asg-web-servers)
  - selektujemo asg-policy i idemo na Actions i Delete.

## umjesto dinamičkog scaling-a da koristimo nas alarm

- Vracamo se na EC2 - Auto Scaling groups - asg-web-servers i idemo na Automatic Scaling - Create dynamic scaling policy
  - Selectujemo policy type - simple scaling
  - Scaling policy name: cpu-simple-scaling
  - Cloud watch alarm - asg-cpu-scaling-policy
  - Take the action:
    - Add - 1 - percent of group
    - 1 capacity units
    - and then wait 0 seconds
    - Create
  - Sad bi trebalo da radi poprilično brzo ali vidimo i dalje da je vrijeme 300 seconds sto je 5 minuta, a to mozemo da smanjimo na nivou CloudWatch servisa.

## smanjivanje vremena na CloudWatch-u

- CloudWatch - All alarms - selektujemo asg-cpu-scaling-policy
  - kliknemo na Actions - Edit - Metric - Period 1 minute - Update alarm

## nazad na Dynamic scaling policies



- kad refresujemo vidimo period of 60 seconds.
- Sad mozemo da izbrisemo Target Tracking policy - oznacimo ga - Actions - Delete.

## kreiranje dodatnog alarma na CloudWatch

- CloudWatch - Alarms - Create Alarm
  - Specify metric and conditions
  - CPUUtilization - oznacimo ispod asg-web-servers - Select metric
  - Period - 1 minute
  - Conditions - Static - Lower - bukvalno ako je ova vrijednost manja od 20 za jednu minutu
  - kliknemo Next
  - Configure actions:
    - In alarm
    - Select existing SNS topic
    - asg-sns-notifications
  - Auto Scaling action
    - OK
    - Select a group - asg-web-servers
  - Take the following action
    - asg-policy
  -
- Add name and description
  - Alarm name: cpu lower than 20
  - 
  -

## vracamo se na ASG - kako da napravimo autoscaling policy - scale down based on CPU

- i kliknemo na Create dynamic scaling policy.
  - Policy type - Step scaling
  - Scaling policy name - scale down
  - CloudWatch alarm - cpu lower than 20
  - Take the action - Remove - 0 - Percent of group when 20
  -
- Sad u ovom slucaju imamo scale down i scale up koji je cpu-simple-scaling

Dynamic scaling policy created or edited successfully.

Details | Activity | **Automatic scaling** | Instance management | Monitoring | Instance refresh

Scaling policies resize your Auto Scaling group to meet changes in demand. With reactive dynamic scaling policies, you can track specific CloudWatch metrics and take action when the CloudWatch alarm threshold is met. Use predictive scaling policies along with dynamic scaling policies in the following situations: when your application demand changes quickly, but with a recurring pattern, or when your EC2 instances require more time to initialize.

**Dynamic scaling policies (2)** [Info](#) Refresh Actions Create dynamic scaling policy < 1 >

**cpu-simple-scaling** ✕

Simple scaling

Enabled

asg-cpu-scaling-policy

breaches the alarm threshold: CPUUtilization > 20 for 1 consecutive periods of 60 seconds for the metric dimensions:

AutoScalingGroupName = asg-web-servers

Add 1 percent of group

Add capacity units in increments of at least 1 capacity units

0 seconds before allowing another scaling activity

**scale down** ✕

Step scaling

Enabled

cpu lower than 20

breaches the alarm threshold: CPUUtilization < 20 for 1 consecutive periods of 60 seconds for the metric dimensions:

AutoScalingGroupName = asg-web-servers

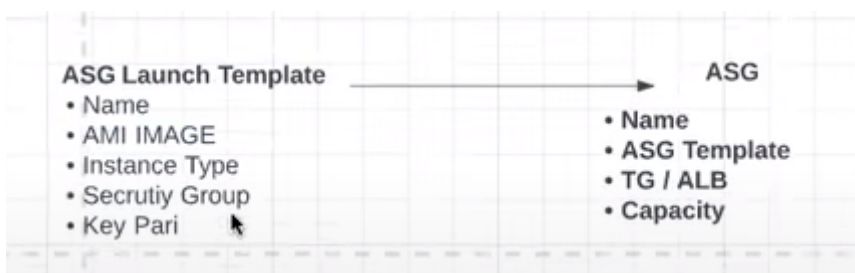
Add 0 percent of group when 20 >= CPUUtilization > -infinity

Add capacity units in increments of at least 1 capacity units

300 seconds to warm up after each step

## Rekapitulacija sta smo morali odraditi da kreiramo ASG

- ALB bi ponasao se na sljedeci nacin - ako neka EC2 instanca nije healthy, on bi preusmjerio saobraćaj prema narednoj instanci koja je healthy. Ukoliko se desi slicna situacija a imamo ASG, ona ce zahvaljujuci health checkovima zamijeniti instancu koja ne radi sa novom instancom.
- Na 1:59:00 Dzenan je ovo demonstrirao terminirajuci 3 instance i pali smo na samo 1 instancu, ASG minimal capacity je 2 instance tako da ASG ce prepoznati da nam fale ovdje instance i ona ce u pozadini krenuti da podize nove EC2 instance.
- Na osnovu cega ASG podize nove instance - na osnovu **ASG Launch Template** u kojem stoji koji AMI image da koristi, koji instance type, koji security group i koji key pair.



- Napomena - treba da prodje neko vrijeme kako bi ASG prepoznala da health status ostalih EC2 instanci nije u redu, trenutno pokazuje samo jednu.

aws Services Search [Option+S]

EC2

Instance Types  
Launch Templates  
Spot Requests  
Savings Plans  
Reserved Instances  
Dedicated Hosts  
Scheduled Instances  
Capacity Reservations

▼ Images  
AMIs  
AMI Catalog

▼ Elastic Block Store  
Volumes  
Snapshots  
Lifecycle Manager

▼ Network & Security  
Security Groups  
Elastic IPs  
Placement Groups  
Key Pairs  
Network Interfaces

▼ Load Balancing  
Load Balancers  
Target Groups

▼ Auto Scaling  
Launch Configurations  
Auto Scaling Groups

EC2 > Auto Scaling groups > asg-web-servers

### asg-web-servers

Details Activity Automatic scaling **Instance management** Monitoring Instance refresh

Instances (1/4)

Filter instances

|                                     | Instance ID         | Lifecycle   | Instance type | Weighted capa... | Launch templa...     | Availability Zone | Health stat |
|-------------------------------------|---------------------|-------------|---------------|------------------|----------------------|-------------------|-------------|
| <input checked="" type="checkbox"/> | i-0b2162acba02f95e5 | InService   | t2.micro      | -                | asg-template-web-ser | us-east-1a        | Healthy     |
| <input type="checkbox"/>            | i-0c0378e8798dc1cc3 | InService   | t2.micro      | -                | asg-template-web-ser | us-east-1d        | Healthy     |
| <input type="checkbox"/>            | i-0d660d79451afae4a | InService   | t2.micro      | -                | asg-template-web-ser | us-east-1a        | Healthy     |
| <input type="checkbox"/>            | i-0f095f6980cd37e2  | Terminating | t2.micro      | -                | asg-template-web-ser | us-east-1d        | Unhealthy   |

Actions: Detach, Set to Standby, Set to InService, Instance scale-in protection, Set scale-in protection, Remove scale-in protection

Lifecycle hooks (0) Info

Filter lifecycle hooks

| Name                                         | Lifecycle transition | Default result | Heartbeat timeout (seco... | Notification target ARN | Role ARN |
|----------------------------------------------|----------------------|----------------|----------------------------|-------------------------|----------|
| No lifecycle hooks are currently configured. |                      |                |                            |                         |          |

Lifecycle hooks help you perform custom actions on instances as they launch and before they terminate.

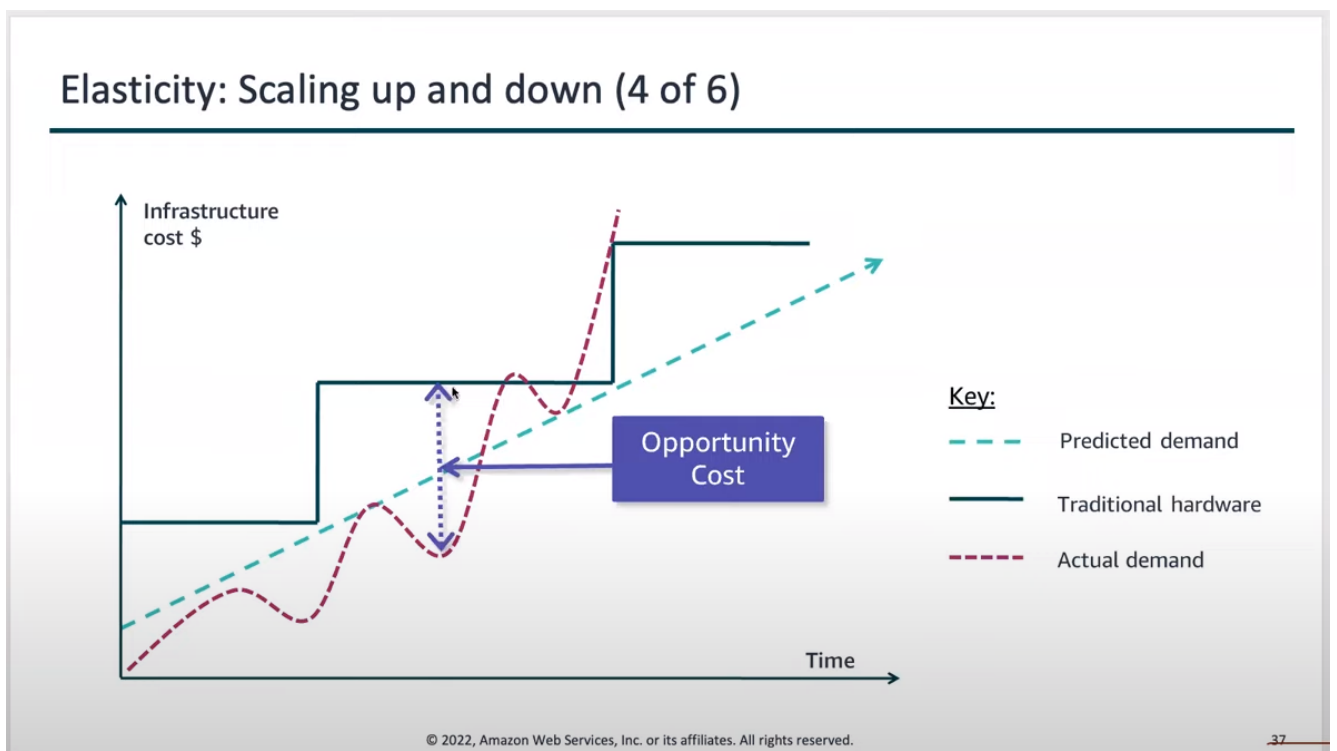
Create lifecycle hook

Warm pool Info

Create warm pool

- Mi mozemo raditi fine tuning vremena koje je potrebno ASG da prepozna unhealthy instance i isto mozemo ih otkaciti rukno ukoliko je potrebno.
- Sa ovim smo osigurali visoku dostupnost i ne samo visoku dostupnost i da se nase unhealthy instance zamijene automatski zahvaljujuci mehanizmu ASG u pozadini.

## Scaling up and down objasnjen na primjeru servera (2:12:00)

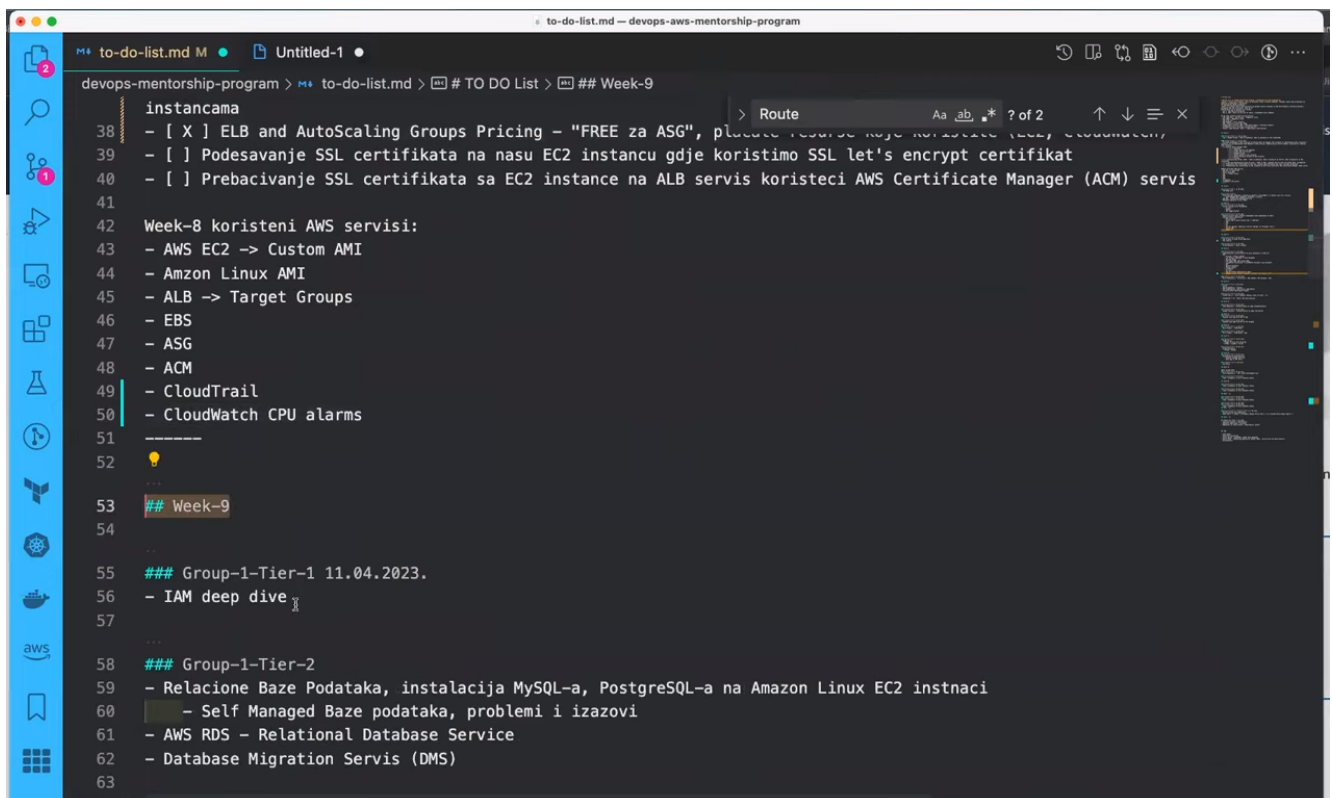


- Kad moramo raditi scaling up na primjerima fizickih servera, moramo unaprijed ulagati u nasu infrastrukturu, dok potreba nad nasim resursima je dosta manja i uvijek smo over-provisioned u odnosu na nase potrebe.
- Resursi stoje neiskoristeni i platili smo za infrastrukturu koju ne koristimo - sto se desava u tradicionalnim data centrima.

- To prevazilazimo nasim ASG - kada se automatski skalira gore i dole u zavisnosti od naseg load-a.
- Pored toga sto ASG automatski mijenja unhealthy instance, takodjer mozemo pratiti nas load dinamički.
- Jos jedna bitna stvar za ASG je da mozemo raditi manuelno skaliranje - mi unaprijed definisemo povecanje ili smanjivanje (primjer scheduled autoscalinga na odredjeni dan u odredjeno vrijeme kad imamo veci load, npr. Black Friday), to su Scheduled actions.
- U Predictive scaling policies - AWS moze na osnovu loada kakav je bio u posljednja dva dana da kreira predictive autoscaling policies - na osnovu toga se radi skaliranje.
- Dynamic scaling policies - sto smo mi radili - na osnovu alarma radimo scale up i scale down.
- Unutar AWS dokumentacije imamo objasnjenje razlicite tipove policy-a: AWS - Documentation - Amazon EC2 Auto Scaling - User Guide

## kako ide placanje ALB-a

- ELB pricing za Application Load Balancer: <https://aws.amazon.com/elasticloadbalancing/pricing/>
- Bitna napomena - precizni FAQ za svaki spomenuti servis
- Odradjeno:



```

devops-mentorship-program > to-do-list.md > # TO DO List > ## Week-9
38 instancama
39 - [X] ELB and AutoScaling Groups Pricing - "FREE za ASG", plus...
40 - [] Podesavanje SSL certifikata na nasu EC2 instancu gdje koristimo SSL let's encrypt certifikat
41 - [] Prebacivanje SSL certifikata sa EC2 instance na ALB servis koristeći AWS Certificate Manager (ACM) servis
42
43 Week-8 korišteni AWS servisi:
44 - AWS EC2 -> Custom AMI
45 - Amazon Linux AMI
46 - ALB -> Target Groups
47 - EBS
48 - ASG
49 - ACM
50 - CloudTrail
51 - CloudWatch CPU alarms
52 -----
53
54 ## Week-9
55
56 ### Group-1-Tier-1 11.04.2023.
57 - IAM deep dive
58
59 ### Group-1-Tier-2
60 - Relacije Baze Podataka, instalacija MySQL-a, PostgreSQL-a na Amazon Linux EC2 instanci
61 - Self Managed Baze podataka, problemi i izazovi
62 - AWS RDS - Relational Database Service
63 - Database Migration Service (DMS)

```