**NGINX on the AWS Cloud**

***Deployment Steps***

**Step 1. Prepare an AWS Account**

1. If you don’t already have an AWS account, create one at https://aws.amazon.com by following the on-screen instructions.

2. Use the region selector in the navigation bar to choose the AWS Region where you want to deploy NGINX on AWS.

3. Create an IAM user, you will be given an Access key and Secret Access key as you need this to login to aws cli

**Step 2. Build Infrastructure**

Prerequisites

To follow this tutorial you will need:

1. The [Terraform CLI](https://learn.hashicorp.com/tutorials/terraform/install-cli?in=terraform/aws-get-started) (1.2.0+) installed.
2. The [AWS CLI](https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html) installed.
3. [AWS account](https://aws.amazon.com/free) and [associated credentials](https://docs.aws.amazon.com/general/latest/gr/aws-sec-cred-types.html) that allow you to create resources.
4. To use your IAM credentials to authenticate the Terraform AWS provider, set the AWS\_ACCESS\_KEY\_ID environment variable.
5. > aws configure
6. export AWS\_ACCESS\_KEY\_ID=
7. export AWS\_SECRET\_ACCESS\_KEY=
8. > mkdir learn-terraform-aws-instance
9. > cd learn-terraform-aws-instance
10. > touch main.tf
11. Open main.tf in your text editor, paste in the configuration below, and save the file.

terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "~> 4.16"

}

}

required\_version = ">= 1.2.0"

}

provider "aws" {

region = "us-west-2"

}

resource "aws\_instance" "app\_server" {

ami = "ami-830c94e3"

instance\_type = "t2.micro"

tags = {

Name = "ExampleAppServerInstance"

}

}

1. terraform init
2. terraform fmt
3. terraform validate
4. terraform apply

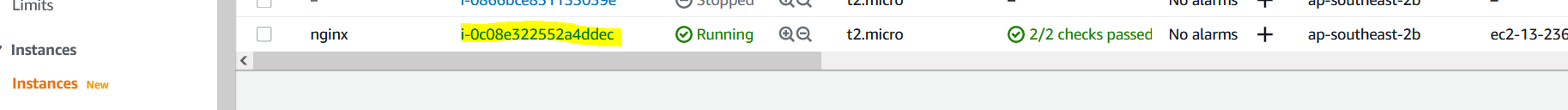
Inspect the current state using terraform show

1. terraform show

If terraform validate was successful

go the AWS Management Console.

1. Check if your EC2 instance was created



Step 3. Create your own VPC.

1. Create VPC in your selected region, example mine is in Sydney

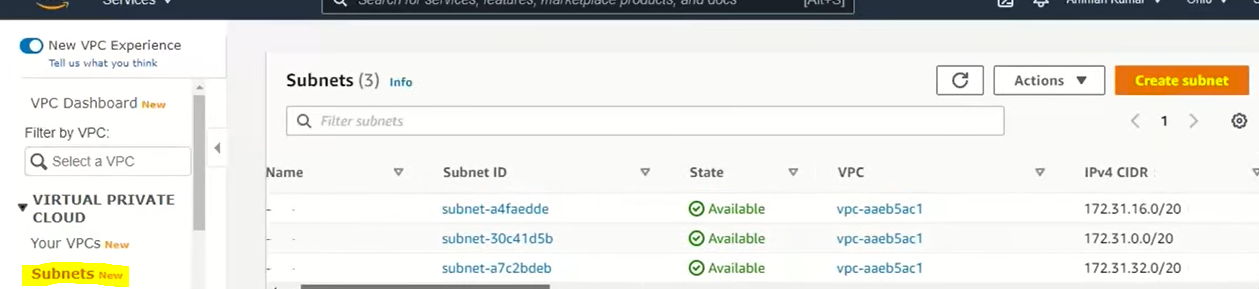
Graphical user interface, application

Description automatically generated

1. Go the search for VPC, then click Create VPC



1. After creating VPC, you have to create a Subnet



1. Then create an Internet Gateways.



1. Then attached your new Internet Gateway to the new VPC you created.
2. Create a Route Table

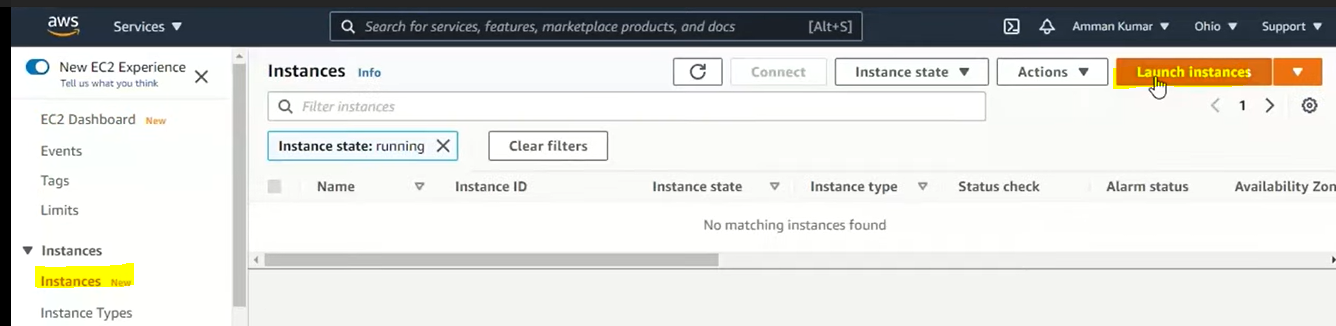
Graphical user interface, text, application

Description automatically generated

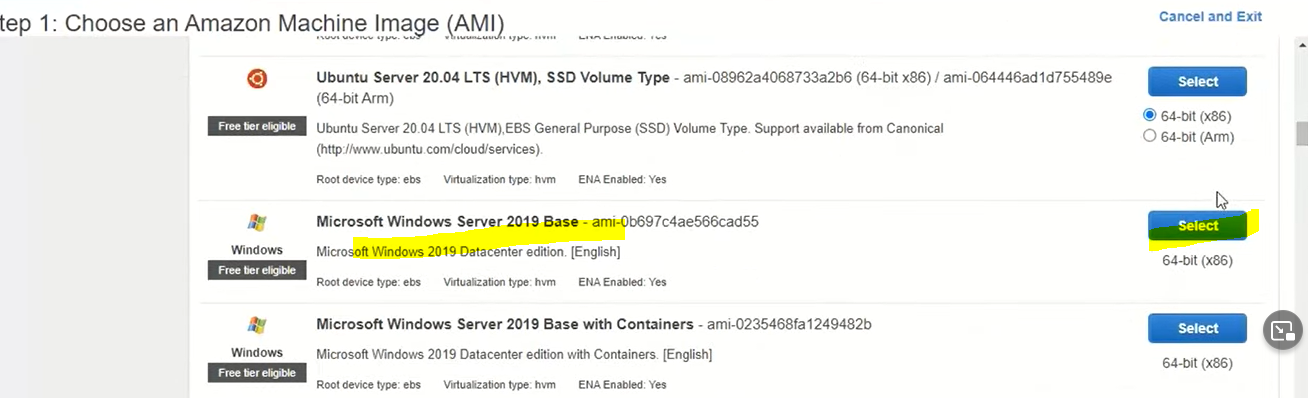
1. Associate the new route table to your subnet you created.

Step 4. Launch an instance to VPC.

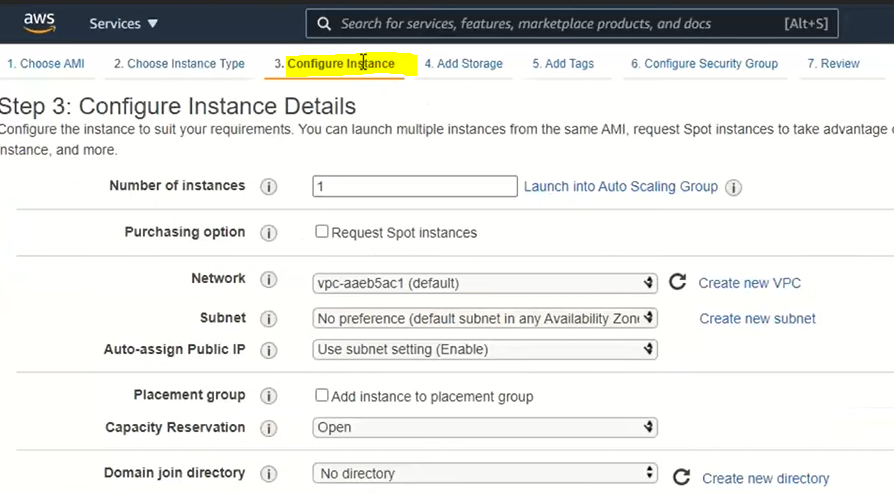
1. Go to Instances and click Launch



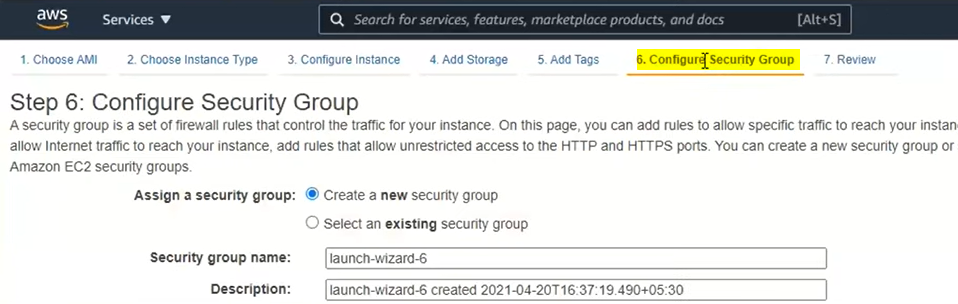
1. Click Windows instance



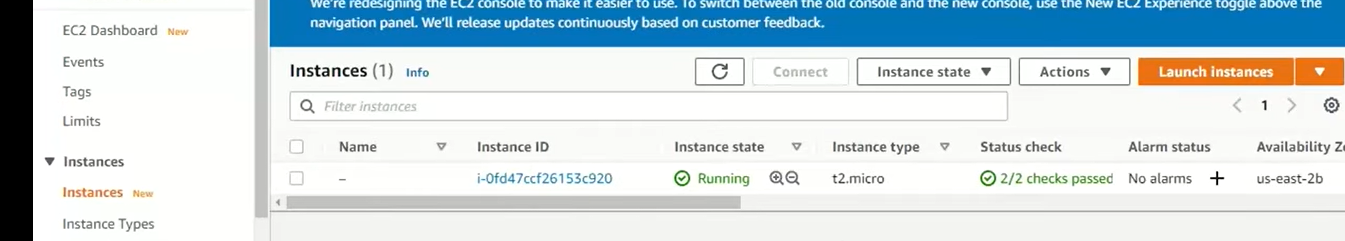
1. Click configure tab and choose In the dropdown your Network, new subnet and auto assign IP to enable.



1. Review and launch and create a Security Group

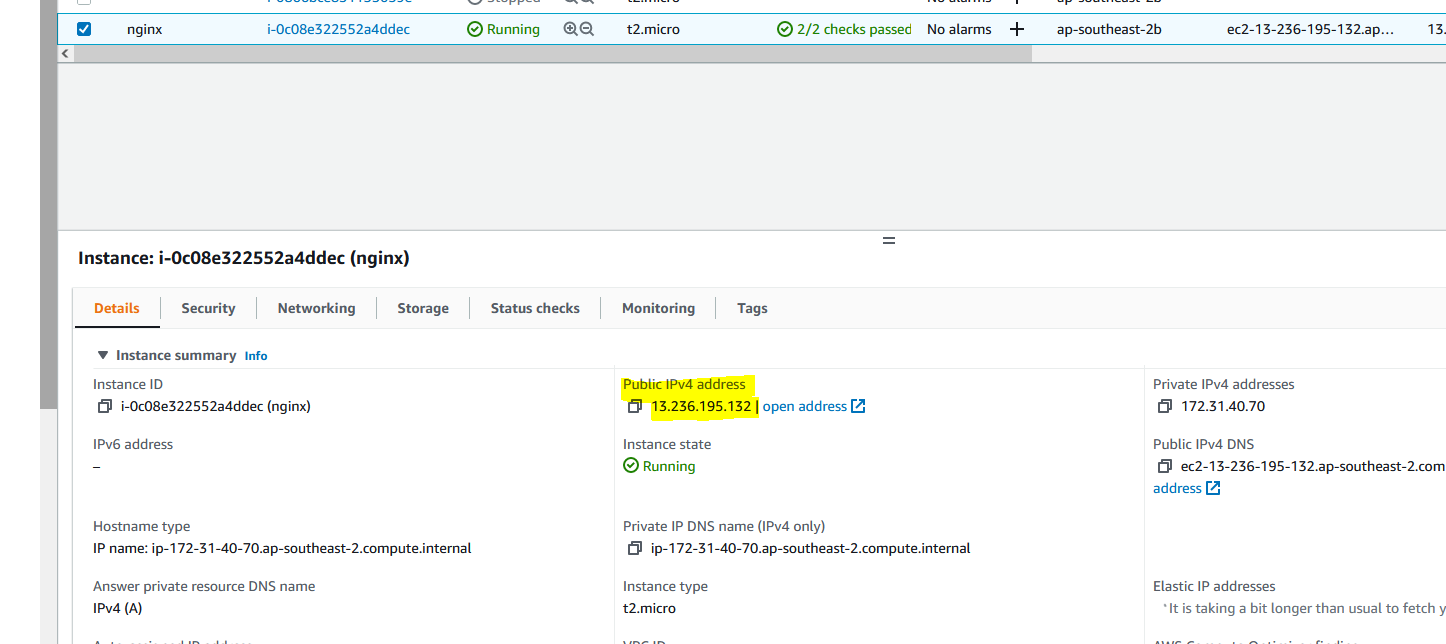


1. And Launch your instance
2. And view instance



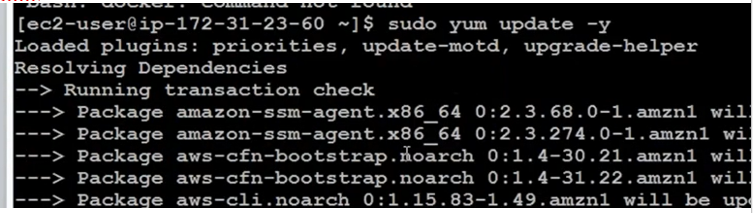
Step 5. Install Docker and install NGINX and start the service

1. Ssh to your EC2 to the public IP, I use putty.



1. Install docker

* Sudo yum update

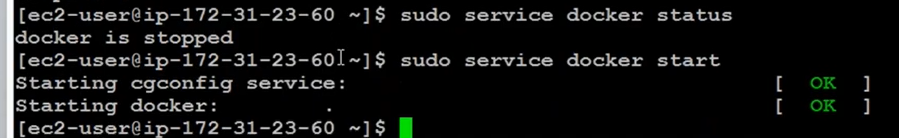


1. > Sudo yum install docker -y



1. Check docker status

* Sudo service docker status
* Sudo service docker start



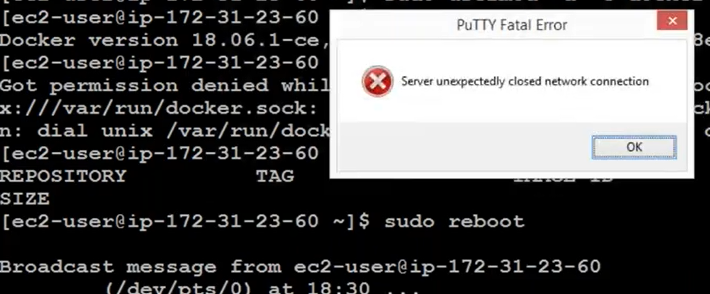
1. Check docker version

* Docker -v



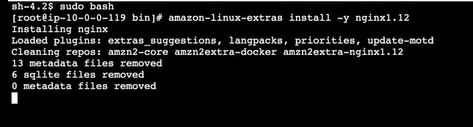
1. Then reboot

< sudo reboot



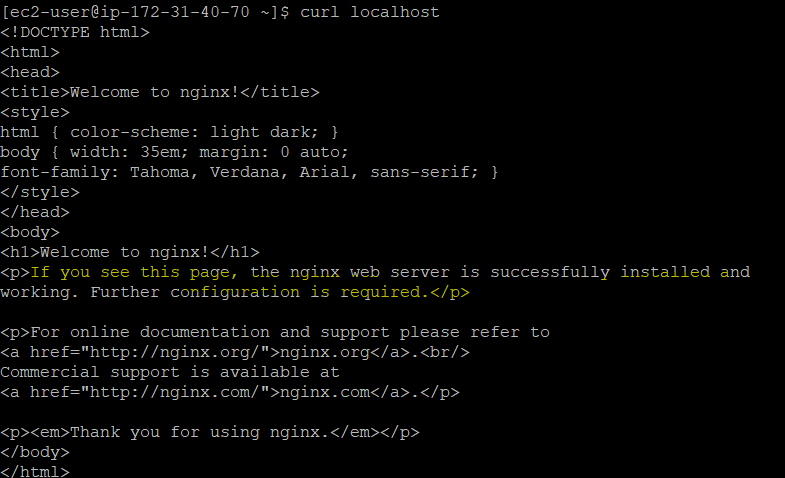
1. Install nginx

* Amazon-linux-extras install -y nginx1.12



1. Check if your nginx is running on your machine by typing

* Curl localhost



And you see html!

1. Collecting Nginx error logs from file

With the following directive, Nginx will log all messages of "warn" severity or higher to the specified log file.

nginx.conf

error\_log /var/log/nginx/error.log warn;

Following is a log message generated by Nginx, an NXLog configuration for parsing it, and the resulting JSON.

Log sample

2022/08/12 10:37:16 [emerg] 17479#17479: epoll\_create() failed (24: Too many open files)

nxlog.conf

<Input nginx\_error>

Module im\_file

File '/var/log/nginx/error.log'

<Exec>

if $raw\_event =~ /^(\S+ \S+) \[(\S+)\] (\d+)\#(\d+): (\\*(\d+) )?(.+)$/

{

$EventTime = strptime($1, '%Y/%m/%d %H:%M:%S');

$NginxLogLevel = $2;

$NginxPID = $3;

$NginxTID = $4;

if $6 != '' $NginxCID = $6;

$Message = $7;

}

</Exec>

</Input>

Output sample

{

"EventReceivedTime": "2022-08-12T10:37:16.245375+02:00",

"SourceModuleName": "nginx\_error",

"SourceModuleType": "im\_file",

"EventTime": "2022-08-12T10:29:16.000000+02:00",

"NginxLogLevel": "emerg",

"NginxPID": "17479",

"NginxTID": "17479",

"Message": "epoll\_create() failed (24: Too many open files)"

}

# [Installing syslog-ng in Amazon Linux](https://www.syslog-ng.com/community/b/blog/posts/installing-syslog-ng-in-amazon-linux-2-including-graviton2)

# Installing syslog-ng from EPEL

# The following command enables the EPEL repository on Amazon Linux

amazon-linux-extras install epel

yum install syslog-ng

systemctl enable syslog-ng

systemctl start syslog-ng

yum erase rsyslog

[root@ip-172-31-40-70 ~]# cat healthcheck.sh

#!/bin/bash

/usr/bin/logger File descriptor: `/sbin/ss -s`

/usr/bin/logger NGINX Status:  `/sbin/service nginx status|grep Active`

/usr/bin/logger NGINX connections: `/bin/curl http://127.0.0.1:8080/nginx\_status`

[root@ip-172-31-40-70 ~]# crontab -l

MAILTO=""

\* \* \* \* \* /root/healthcheck.sh 🡪 check health status every 1 minute

Sample logs to get nginx status, file descriptor and number active of connections:

message arrived successfully in /var/log/messages:

[root@ip-172-31-40-70 log]# tail -f messages

Aug 14 01:05:01 ip-172-31-40-70 root: NGINX Status:   Active: active (running) since Sat 2022-08-13 23:48:53 UTC; 1h 16min ago

Aug 14 01:05:01 ip-172-31-40-70 root: NGINX connections: Active connections: 1 server accepts handled requests 90 90 90 Reading: 0 Writing: 1 Waiting: 0

Aug 14 01:05:01 ip-172-31-40-70 systemd[1]: Removed slice User Slice of root.

Aug 14 01:05:15 ip-172-31-40-70 dhclient[2866]: XMT: Solicit on eth0, interval 130710ms.

Aug 14 01:06:01 ip-172-31-40-70 systemd[1]: Created slice User Slice of root.

Aug 14 01:06:01 ip-172-31-40-70 systemd[1]: Started Session 936 of user root.

Aug 14 01:06:01 ip-172-31-40-70 root: File descriptor: Total: 216 TCP: 15 (estab 2, closed 3, orphaned 0, timewait 1) Transport Total IP IPv6 RAW 0 0 0 UDP 8 4 4 TCP 12 8 4 INET 20 12 8 FRAG 0 0 0

Rest API

Vi /etc/nginx/conf.d/api.gateway.conf

|  |
| --- |
| error\_page 400 = @400; |
|  |

|  |
| --- |
| location @400 { return 400 '{"status":400,"message":"Bad request"}\n'; } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| error\_page 401 = @401; |
|  |

|  |
| --- |
| location @401 { return 401 '{"status":401,"message":"Unauthorized"}\n'; } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| error\_page 403 = @403; |
|  |

|  |
| --- |
| location @403 { return 403 '{"status":403,"message":"Forbidden"}\n'; } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| error\_page 404 = @404; |
|  |

location @404 { return 404 '{"status":404,"message":"Resource not found"}\n'; }

Response

$ **curl -i https://** **127.0.0.1:8080/foo**

HTTP/1.1 400 Bad Request

Server: nginx/1.19.5

Content-Type: application/json

Content-Length: 39

Connection: keep-alive

{"status":400,"message":"Bad request"}