

Saad Abdullah
2402262

IT00CE11-3005

Cloud Computing

Assignment # 1

September 07, 2024



Launching an AWS VM | STUDENT NO-2402262

Saad Abdullah

saad.abdullah@abo.fi

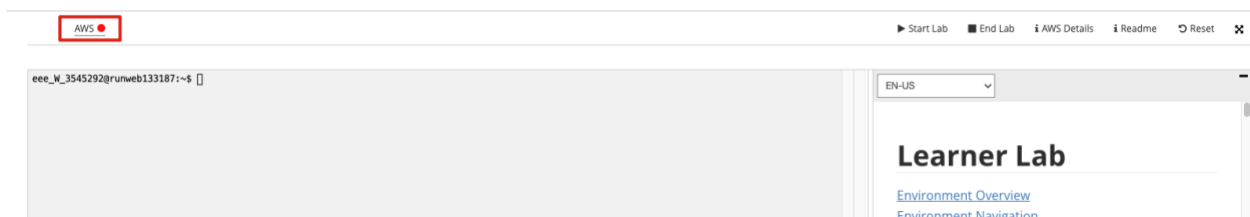
Table of Contents

Step 1: Log in to the AWS console.	3
Step 2: Launch an instance (Virtual machine-VM) via console.	4
Step 3: Check the public IP & DNS of the newly created VM also connect to it from the local machine using SSH.	6
Step 4: Check if the Linux distribution contains packages to be updated.	7
Step 5: Get the following information from VM.	8
Step 6: Execute the special command from VM	9
Step 7: Download log.dat to local machine using scp	10
Step 8: Kill the VM!	10
Step 9: Content for log.dat file:.....	11
What would happen if you lost the private key provided when you instantiated your VM?	11
Do you have any idea where was the physical server on which your VM was running?	12
How long was the “waiting time” (approximately) between requesting a VM and having it up and running?	12
Reflection:	12
Have you learned anything completely new?.....	12
Did anything surprise you?	12
Did you find anything challenging? Why?.....	12
Did you find anything satisfying? Why?.....	12

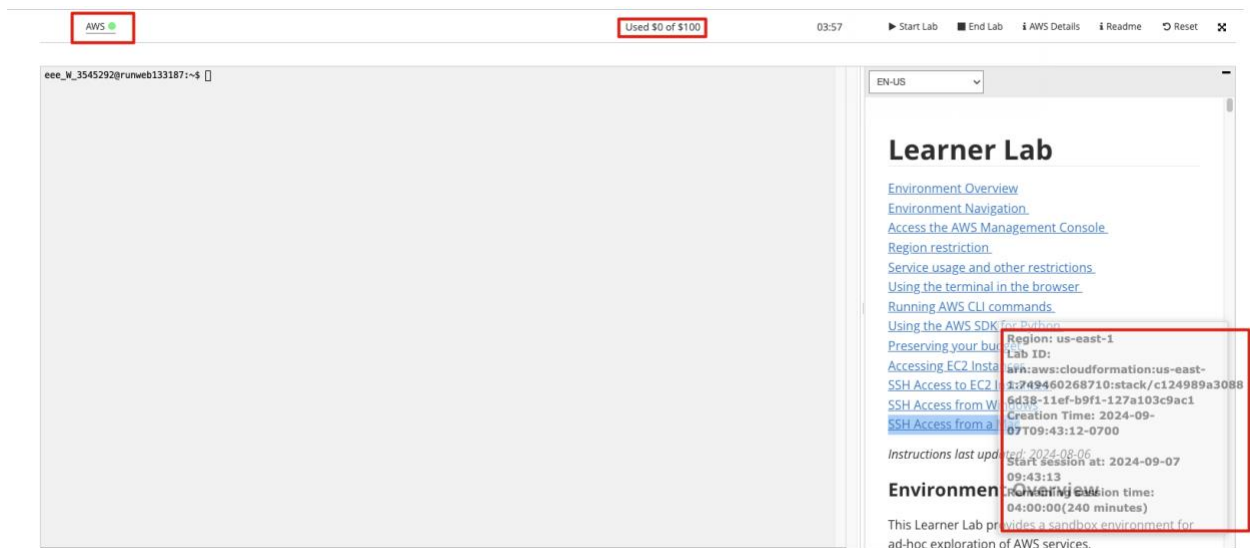
Launching an AWS VM

Step 1: Log in to the AWS console.

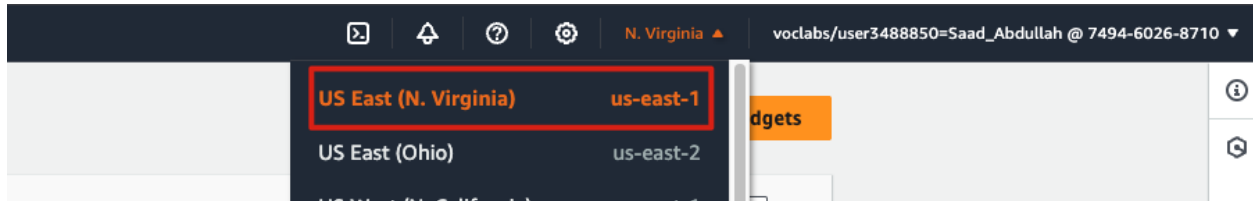
To log into the AWS console, I first accepted the AWS Academy course invitation and then headed over to the AWS Learner LAB.



Then I started the LAB after clicking the Start Lab button.

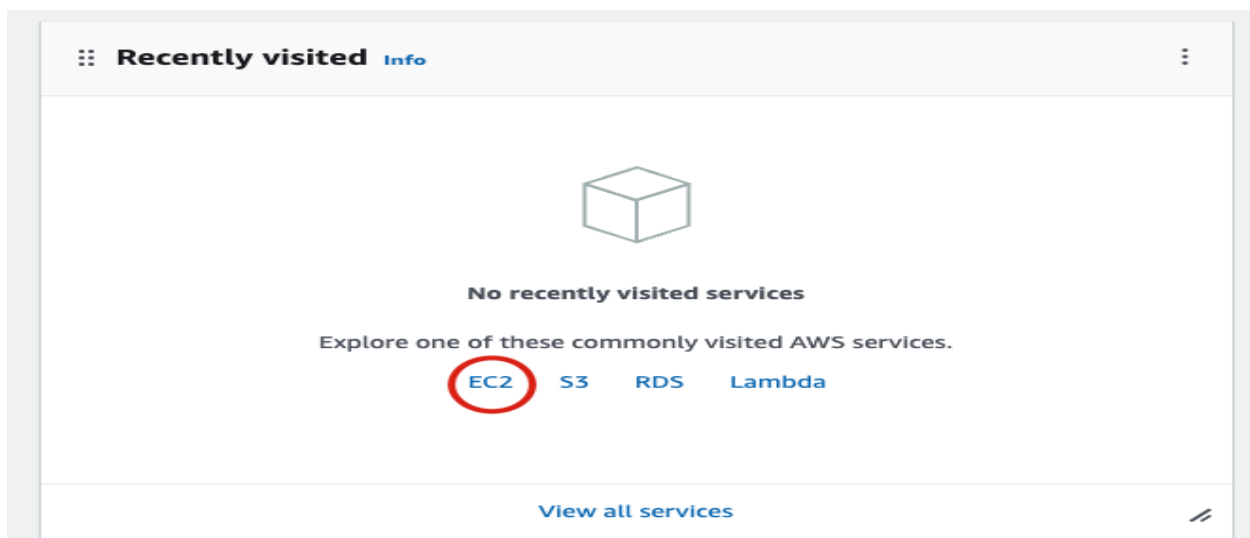


It took around 4 minutes to start the LAB session, afterwards, I clicked the AWS link with the green circle to access the AWS console.






Step 2: Launch an instance (Virtual machine-VM) via console.

I clicked on EC2 to explore more options like launching instances with it.



Then I clicked on the launch instance button for EC2.

Resources [Info](#)


EC2 Global View   

You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:



Instances (running)	0	Auto Scaling Groups	0	Capacity Reservations	0
Dedicated Hosts	0	Elastic IPs	0	Instances	0
Key pairs	1	Load balancers	0	Placement groups	0
Security groups	1	Snapshots	0	Volumes	0

Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.


Launch Instance ▼

[Migrate a server](#) 

Note: Your instances will launch in the US East (N. Virginia) Region


Service health [AWS Health Dashboard](#)  

Region
US East (N. Virginia)

Status
 This service is operating normally.

Zones

Then I created an RSA key pair to access this instance from the local machine.

Create key pair 

Key pair name
Key pairs allow you to connect to your instance securely.

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type



☒ **RSA**
RSA encrypted private and public key pair

☐ **ED25519**
ED25519 encrypted private and public key pair

Private key file format

☒ **.pem**
For use with OpenSSH

☐ **.ppk**
For use with PuTTY

 When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#) 

Cancel

Create key pair

Launched this instance with the configurations recommended in the LAB manual. Allow HTTP for the curl request for the LAB task to work properly.

Firewall (security groups) [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-1' with the following rules:

☒ Allow SSH traffic from Anywhere
Helps you connect to your instance

☐ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

☒ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

▼ Summary

Number of instances [Info](#)
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.5.2...[read more](#)
ami-0182f373e66f89c85

Virtual server type (Instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Cancel **Launch instance**
[Review commands](#)

▼ Configure storage [Info](#) [Advanced](#)

1x GiB Root volume (Not encrypted)

Instance is running now.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elas
saads-first-ec2-instance	i-040e069bd2ff2df68	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-23-20-58-217.com...	23.20.58.217	-

Step 3: Check the public IP & DNS of the newly created VM also connect to it from the local machine using SSH.

Below are the public IP and DNS for the newly created VM.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 address
saads-first-ec2-instance	i-040e069bd2ff2df68	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-23-20-58-217.compute-1.amazonaws.com	23.20.58.217

Then I used the following command from the local machine terminal to access the VM.

Page 7 of 12

```
sudo yum update
```

```
~/Downloads — ec2-user@ip-172-31-25-18:~ — ssh -i saads-aws-ec2.pem ec2-user@ec2-  
[ec2-user@ip-172-31-25-18 ~]$ sudo yum update  
Last metadata expiration check: 17:28:31 ago on Sat Sep  7 17:02:06 2024.  
Dependencies resolved.  
Nothing to do.  
Complete!  
[ec2-user@ip-172-31-25-18 ~]$
```

Everything was already updated.

Step 5: Get the following information from VM.

I used the following command for CPU model name, clock frequency, and cache size.

```
cat /proc/cpuinfo
```

```
[ec2-user@ip-172-31-25-18 ~]$ cat /proc/cpuinfo  
processor       : 0  
vendor_id      : GenuineIntel  
cpu family     : 6  
model          : 79  
model name     : Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz  
stepping       : 1  
microcode      : 0xb000040  
cpu MHz        : 2300.108  
cache size     : 46080 KB  
physical id    : 0  
siblings       : 1  
core id        : 0  
cpu cores      : 1  
apicid         : 0  
initial apicid : 0  
fpu            : yes  
fpu_exception  : yes  
cpuid level    : 13  
wp             : yes  
flags          : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov  
good nopl xtopology cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse  
erervisor lahf_lm abm cpuid_fault invpcid_single pti fsgsbase bmi1 avx2 smep bmi  
bugs           : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf md  
bogomips       : 4600.01  
clflush size   : 64  
cache_alignment : 64  
address sizes  : 46 bits physical, 48 bits virtual  
power management:  
  
[ec2-user@ip-172-31-25-18 ~]$
```


For the CPU vendor and hypervisor vendor, I used the following command:

```
lscpu
```

```
[[ec2-user@ip-172-31-25-18 ~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Address sizes:          46 bits physical, 48 bits virtual
Byte Order:             Little Endian
CPU(s):                 1
On-line CPU(s) list:    0
Vendor ID:              GenuineIntel
Model name:             Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
CPU family:             6
Model:                 79
Thread(s) per core:     1
Core(s) per socket:     1
Socket(s):              1
Stepping:               1
BogoMIPS:               4600.01
Flags:                  fpu vme de pse tsc msr pae mce cx8 apic sep mt
                        tsc rep_good nopl xtopology cpuid tsc_known_fr
                        imer aes xsave avx f16c rdrand hypervisor lahf
                        eopt
Virtualization features:
Hypervisor vendor:      Xen
virtualization type:    full
Caches (sum of all):
L1d:                    32 KiB (1 instance)
L1i:                    32 KiB (1 instance)
L2:                     256 KiB (1 instance)
L3:                     45 MiB (1 instance)
NUMA:
NUMA node(s):           1
NUMA node0 CPU(s):      0
Vulnerabilities:
```

Step 6: Execute the special command from VM

I used:

```
curl "vm4460.kaj.pouta.csc.fi/logs.php?name=saad_abdullah" > log.dat
```

```
[[ec2-user@ip-172-31-25-18 ~]$ curl "vm4460.kaj.pouta.csc.fi/logs.php?name=saad_abdullah" > log.dat
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %         %         Dload  Upload  Total    Spent    Left    Speed
100  128  100  128    0    0   313      0 --:--:-- --:--:-- --:--:--   312
[[ec2-user@ip-172-31-25-18 ~]$ cat log.dat

[[ec2-user@ip-172-31-25-18 ~]$ cat log.dat

Name: saad_abdullah -- 54.147.150.141 -- ec2-54-147-150-141.compute-1.amazonaws.com -- 192.168.1.14 -- 1725795776 -- curl/8.5.0 ec2-user@ip-172-31-25-18 ~]$
```

Step 7: Download log.dat to local machine using scp

First, I checked the location of the log.dat file in my VM using pwd command.

```
[ec2-user@ip-172-31-25-18 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-25-18 ~]$ ls
log.dat
[ec2-user@ip-172-31-25-18 ~]$
```

Then I googled about scp and made the following command to download the log.dat file, and it worked for me: (the . in the end means to download the file in the current directory which was Downloads for me)

```
scp -i saads-aws-ec2.pem ec2-user@ec2-54-147-150-141.compute-1.amazonaws.com:/home/ec2-user/log.dat .
```

```
(base) ~/Downloads
14:57:11 → scp -i saads-aws-ec2.pem ec2-user@ec2-54-147-150-141.compute-1.amazonaws.com:/home/ec2-user/log.dat .
log.dat
100% 128 0.5KB/s 00:00
005 → saadabdullah saads-MBP:s
14:57:24 → cat log.dat
(Sun, Sep08)
Name: saad_abdullah -- 54.147.150.141 -- ec2-54-147-150-141.compute-1.amazonaws.com -- 192.168.1.14 -- 1725795776 -- curl/8.5.0
(base) ~/Downloads
14:57:29 →
saadabdullah saads-MBP:s
(Sun, Sep08)
```

Step 8: Kill the VM!

✓ Successfully initiated termination (deletion) of I-040e069bd2ff2df68

I-040e069bd2ff2df68 (saads-first-ec2-instance) running

Instance state settings

☒ Start
Available when the instance is stopped

☐ Stop

☐ Hibernate
This instance did not have Stop - Hibernate enabled at launch

☐ Reboot

☒ Terminate

Cancel

Change state

Step 9: Content for log.dat file:

Name: saad_abdullah -- 54.147.150.141 -- ec2-54-147-150-141.compute-1.amazonaws.com --
192.168.1.14 -- 1725795776 -- curl/8.5.0

What would happen if you lost the private key provided when you instantiated your VM?

So, for security purposes, the private key is not stored in AWS. We can only have it during the instantiation of the VM and in case of losing it, we won't be able to do ssh into our VM which ultimately means we can't access our VM. But there are some workarounds to recover the instance, I found these workarounds in [AWS Forum](#) and in a [medium article](#).

Do you have any idea where was the physical server on which your VM was running?

AWS keeps this information private and we can't know the exact location of the physical server where our VM is running, but we can make a guess about the region where it is running like my VM was in us-east-1 which means it was somewhere in Virginia.

How long was the "waiting time" (approximately) between requesting a VM and having it up and running?

It took me around 2-3 minutes for my VM to be up and running using Amazon Linux AMI.

Reflection:

Have you learned anything completely new?

Some commands were familiar to me as I have been using Linux for quite some time now, but I had never made an AWS instance from scratch. So, instantiating the instance was new for me.

Did anything surprise you?

No, there wasn't anything surprising.

Did you find anything challenging? Why?

No.

Did you find anything satisfying? Why?

Given that it was our first assignment, the flow was quite good to get some confidence at the start of the course.