**Getting Started with ECE on AWS**

A lot of organizations have recently been moving over to “the cloud” for a number of reasons. Some companies have a mandate to evaluate cloud options first; others are looking for easy ways to scale hardware to manage bursty data loads or simply for ways of having “someone else deal” with upgrading.

Regardless of the drive, we have seen an enormous amount of interest within our community to run the Elastic Stack on the cloud, and Elastic can certainly offer solution options. One of the great things about Elastic is that we are hardware agnostic: if it fits the use case and required read/write mix, our hats are off to you. Perhaps, a customer wants to move from Azure to on-prem and then port clusters over to GCP or maybe just some of the clusters. Your success is our success, so we are betting on your success with the Elastic Stack!

For a pure SaaS-based solution, we offer ElasticSearch Service (ESS). In this solution, we manage all the hardware for you and can provide top of the line SLA level support options, alongside all the great features our engineers have been rolling into commercial versions. Just a few clicks, and you are up and running! Suffice it to say, this is a premium service a lot of customers prefer to entrust their enterprise applications to run on. And, in the back end, the support provided is grade-A, top of the line, as would be expected by any organization betting their business on the use of well-engineered software.

That said, there are times when organizations would prefer to run their own service - often within their own privately secured network, but, at the same time, would still prefer this service to be managed, so that administrators do not have to deal with the intricacies of managing upgrades and scaling up and down nodes. And now for a drumroll……...

We have a solution for you!

Elastic Cloud Enterprise (ECE) takes what ESS provides in the cloud and enables you, as the user, to own your Elastic cloud. You can spin up clusters on the fly; upgrades take two clicks; scaling upwards and downwards is a mere matter of sliding controls, and you can be ensured to stay highly-available while scaling. Beyond that, with ECE, you can specialize your Elastic clusters to specific purposes - such as search or log use cases; for example, ECE can easily deploy a hot/warm architecture and allow users to optimize spinning up clusters with preset templates. There has recently been a great deal of work done in index lifecycle management, to the point where a lot of folks have decided to move away from curator, since the new console is so useful. Please check out this webinar {link here} to get a better idea of some of the amazing features which have been rolled out for ECE.

This guide can help you get started on AWS with ECE. Our goal is to get ECE up and running quickly so you can try out sparking up clusters and be able to quickly evaluate if ECE is going to be able to help your efforts. Beyond that, ECE 2.2 has released with sample ansible scripts which can help folks who want to roll out on-prem and are looking for ways to automate.

At the end of this guide, you will have three instances of AWS running with ECE and will be able to build clusters in your environment. This resulting deployment will not be a production-ready install and does not replace a planned for production rollout, but will help with understanding the process of how to install ECE and get started managing your own deployments. You will also have a trimmed down version of a [small baseline install](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-topology-example1.html).

For simplicity, a single NVMe disk (1900GB’s) is installed on each of 3 i3.2xlarge instances. If desiring more capacity, you can add more storage during step 7. Also, we will go with a very simple, semi-exposed VPC network provided by Amazon’s wizard.

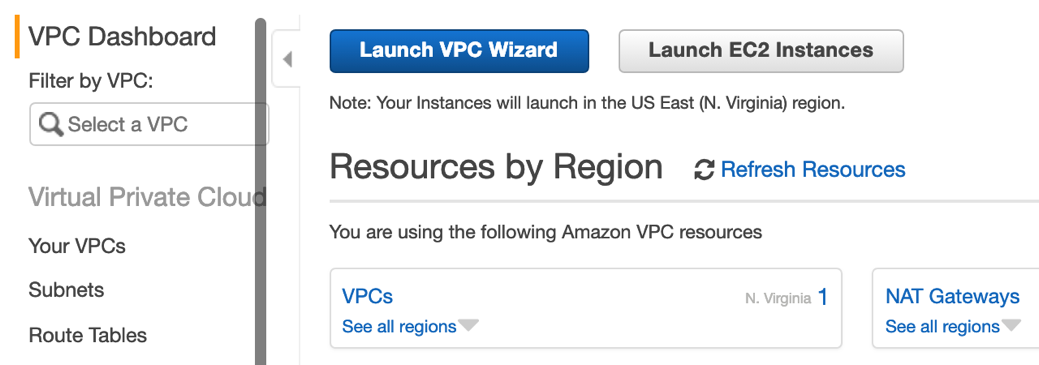
Just a couple of notes about an actual production rollouts before we get started through this below simple point and click/install process below. First, there are several next step references at the end of the blog if you’d like to go deeper. Second, this on-demand ECE class {link} will be helpful. Third, for hardware...**{Shawn here)**

So, without further ado, let’s get started. Here is **what you’ll need**:

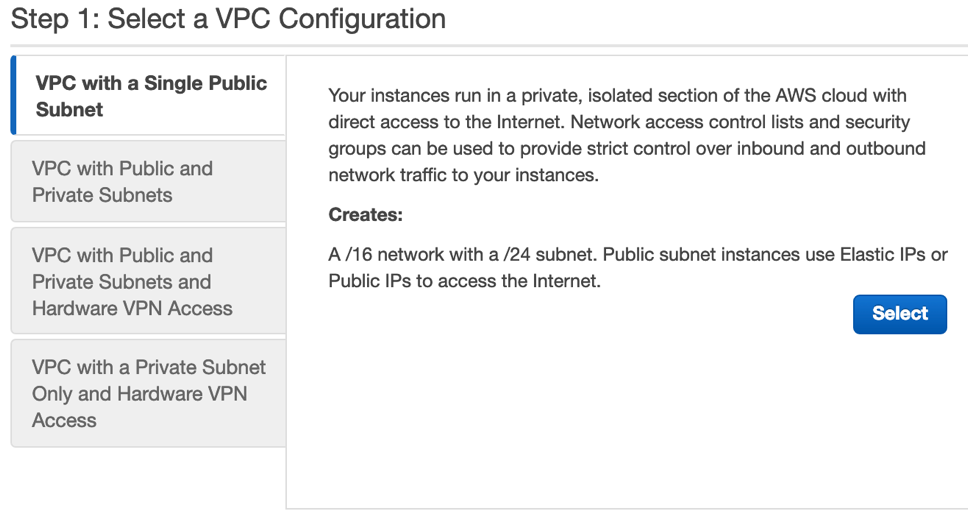
1. An **AWS account** which allows you to spark up i3 instances (we will do this for Azure and GCP in the near future perhaps with more scripts, but having gone through the below click-thru, you will get a great idea of all the working pieces of ECE).
2. **A client** (PC, Linux, Mac, or otherwise - Chromebook? We love Google, too!) which lets you tunnel into AWS

And here are the instructions to **get started on ECE on AWS from start to PoC-ready**:

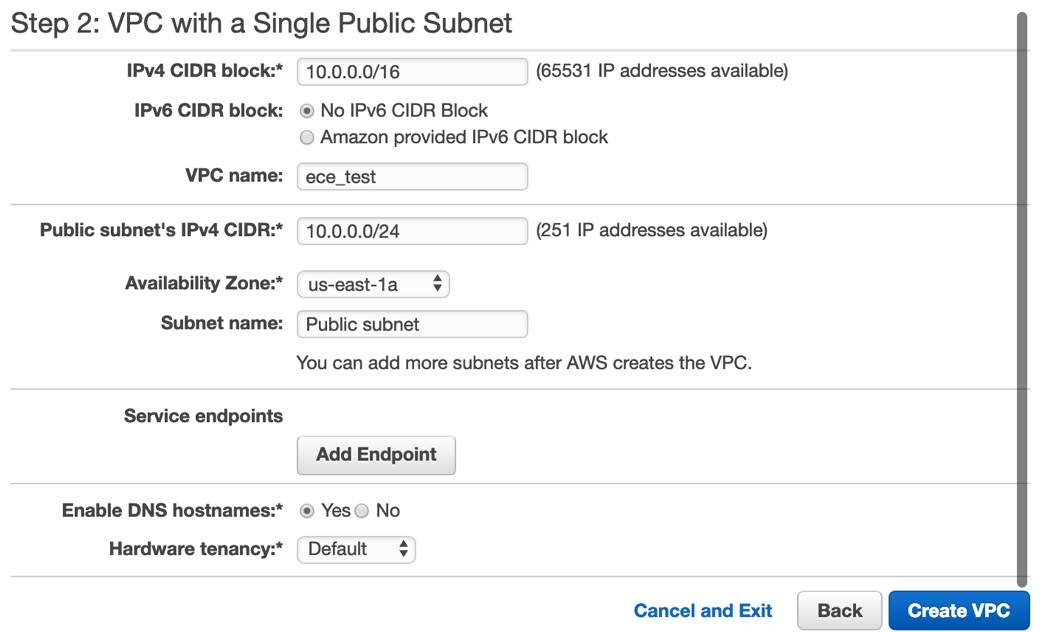
1. Go to VPC wizard to create a new vpc (VPC Dashboard ---> Launch VPC Wizard )



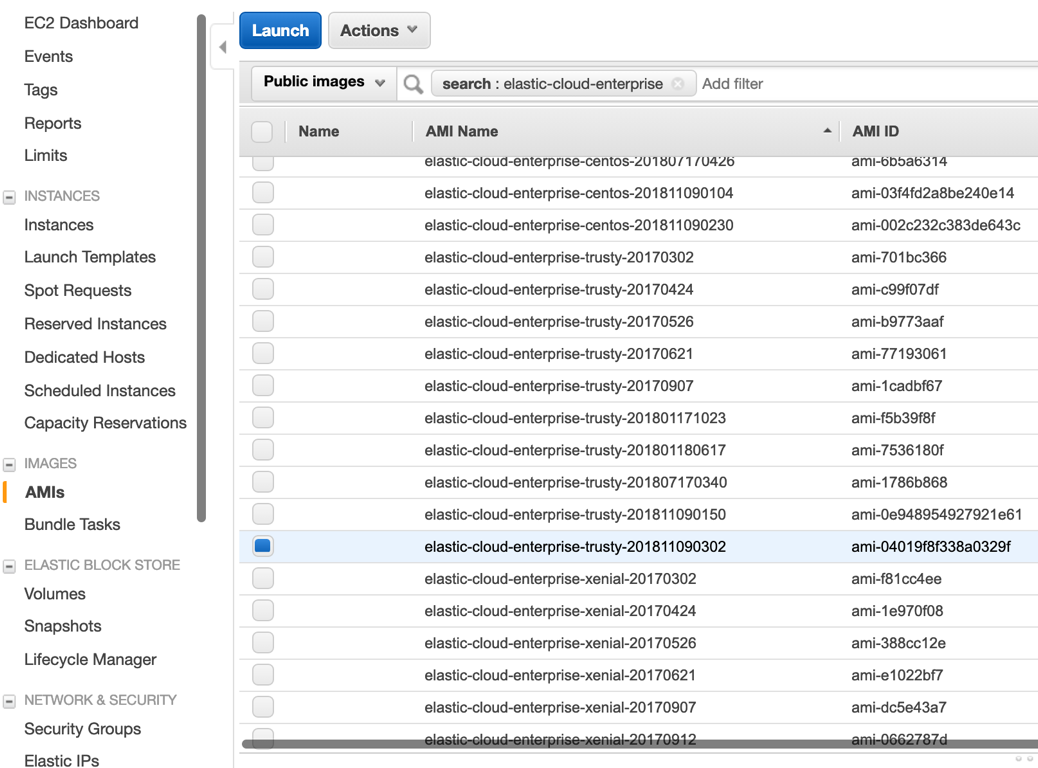
1. Select simple deployment – **“VPC with a Single Public Subnet**” for now.



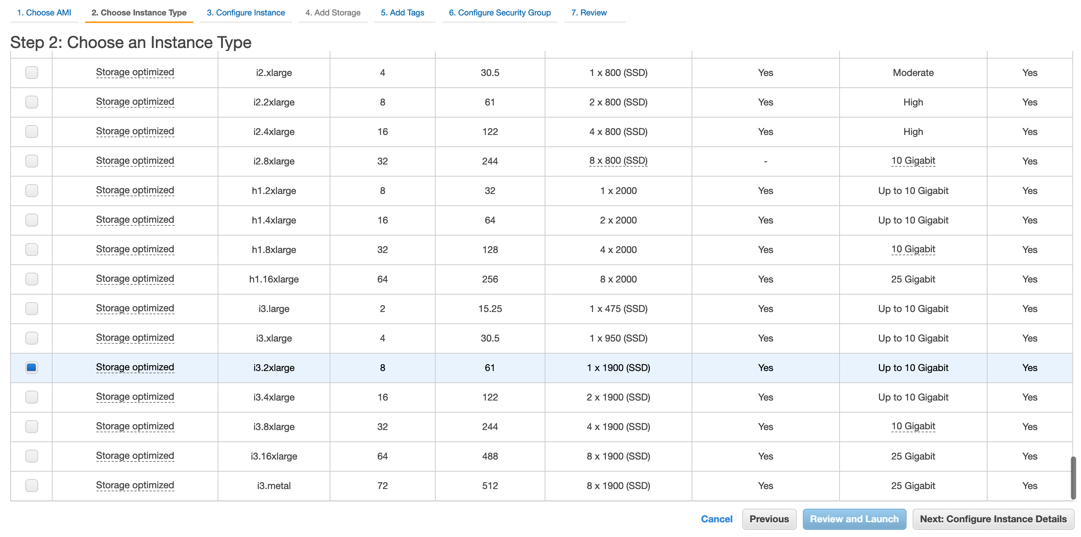
1. **Name your VPC, pick an AZ**, and take default CIDR settings for now and click **“Create VPC”**



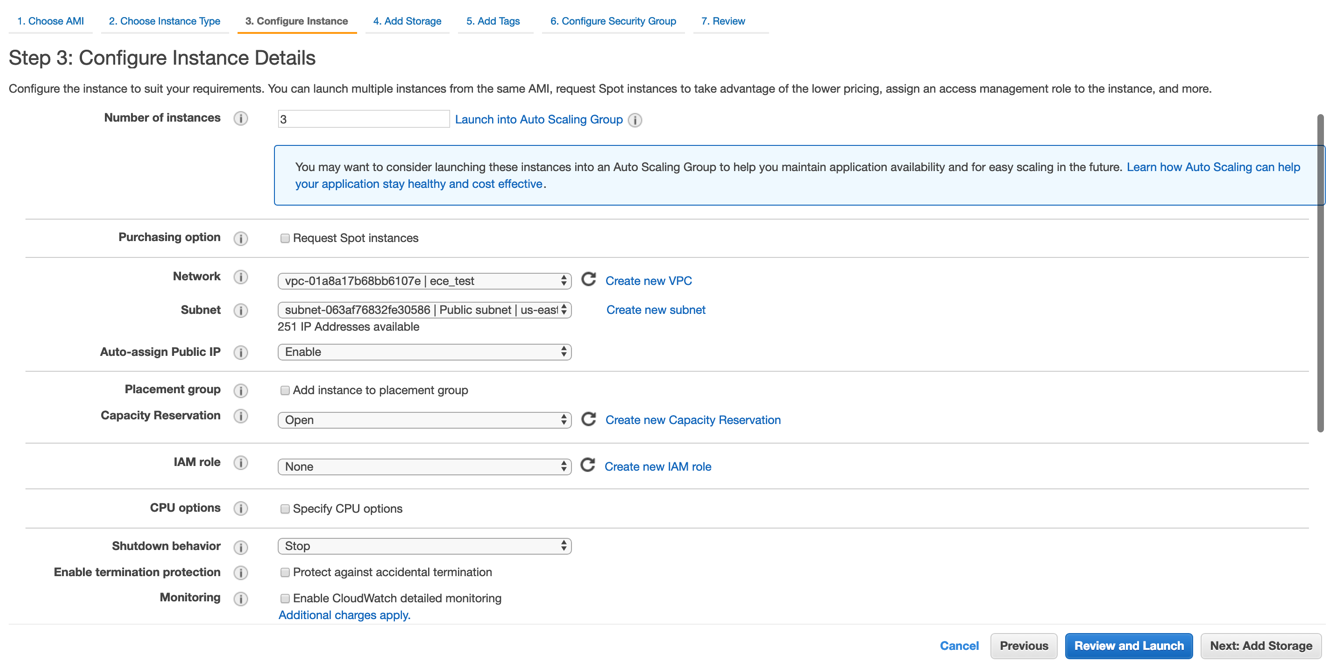
1. Now go to services ---> EC2 and click AMI’s. Modify filter to be public images and search for “elastic-cloud-enterprise”. Pick “**elastic-cloud-enterprise-trusty-201811090302**” and click launch button.



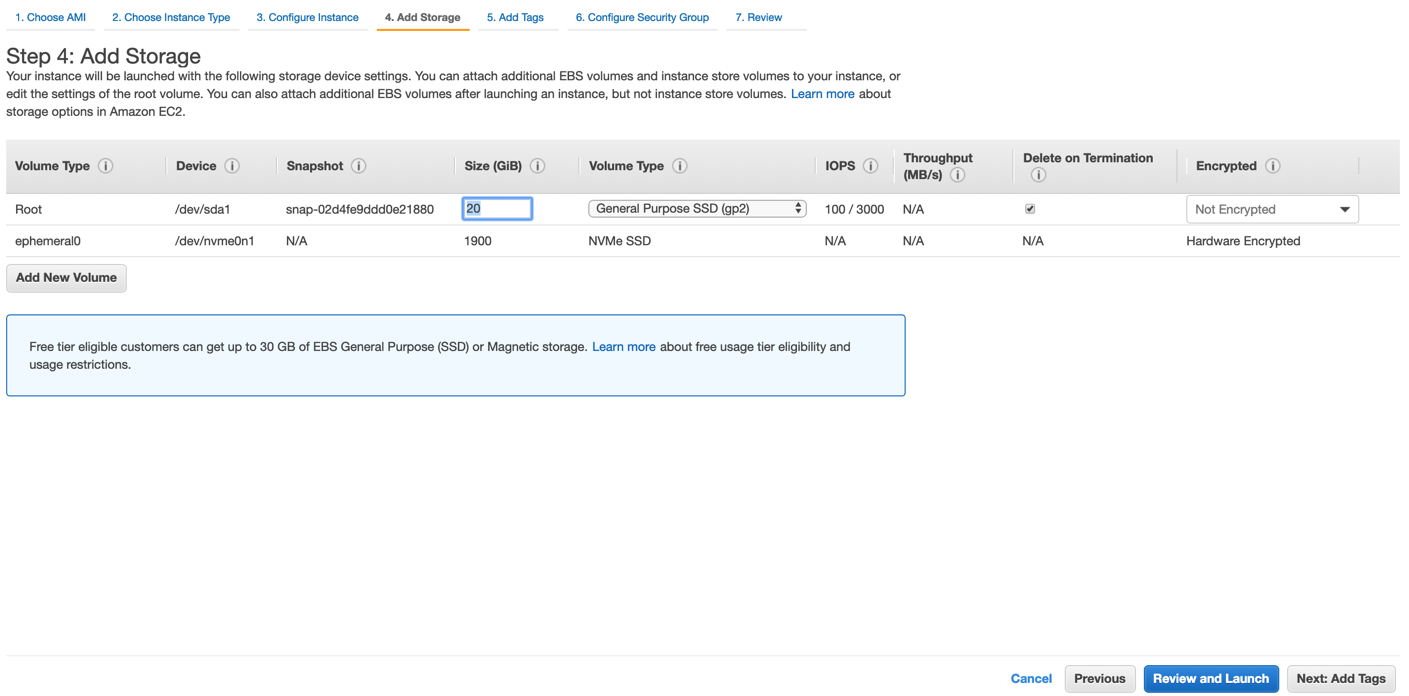
1. Pick the **i3.2xlarge** instance choice. In production, we would probably pick 8xlarge but this is a dev environment. Continue with wizard and select “**Next: Configure Instance Details**”



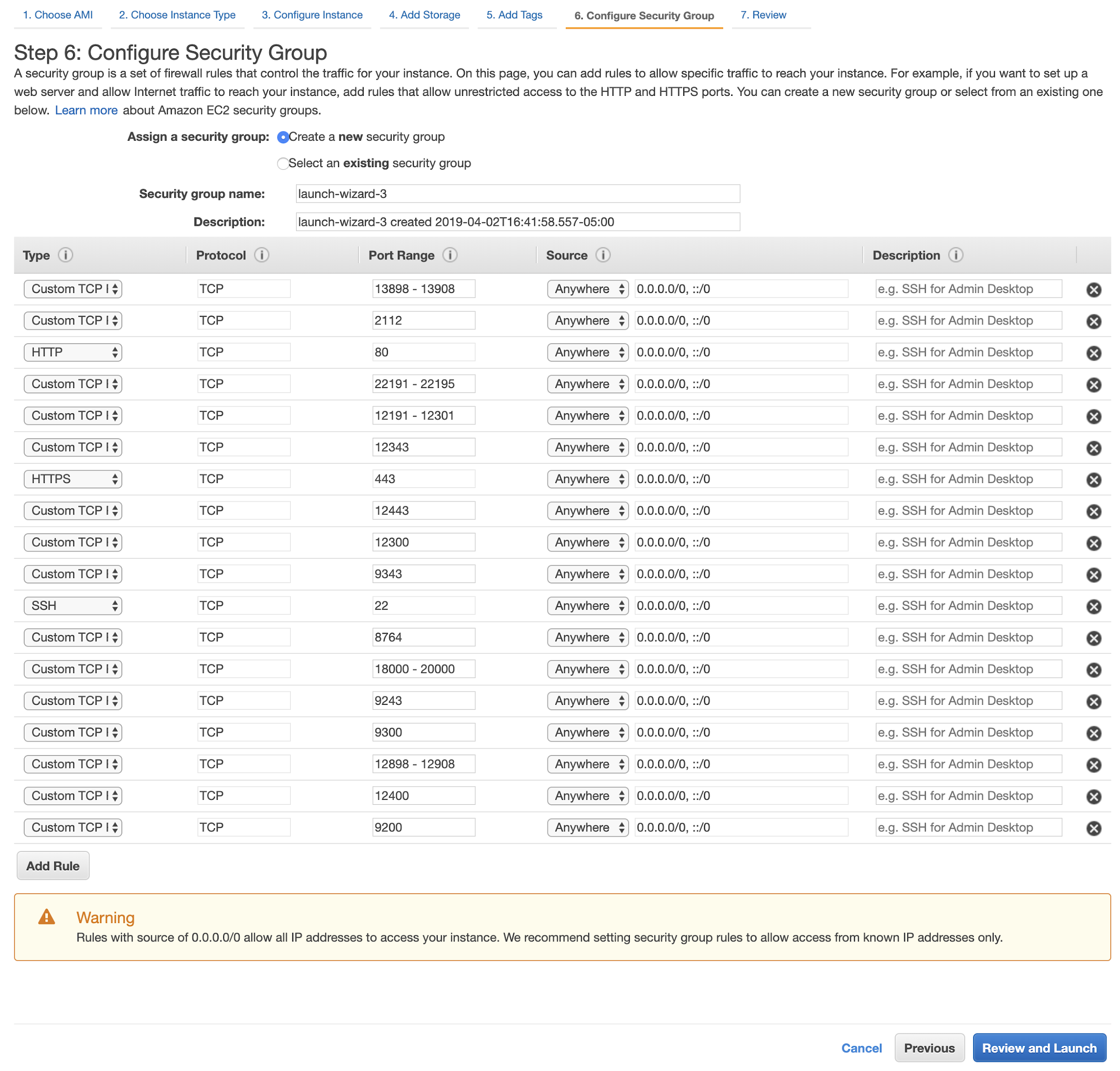
1. Proceed to configure instance. Pick 3 instances. Select the VPC created in steps 1-3. Be certain **to enable auto-assign public IP** . Click the “Next: Add Storage” button.



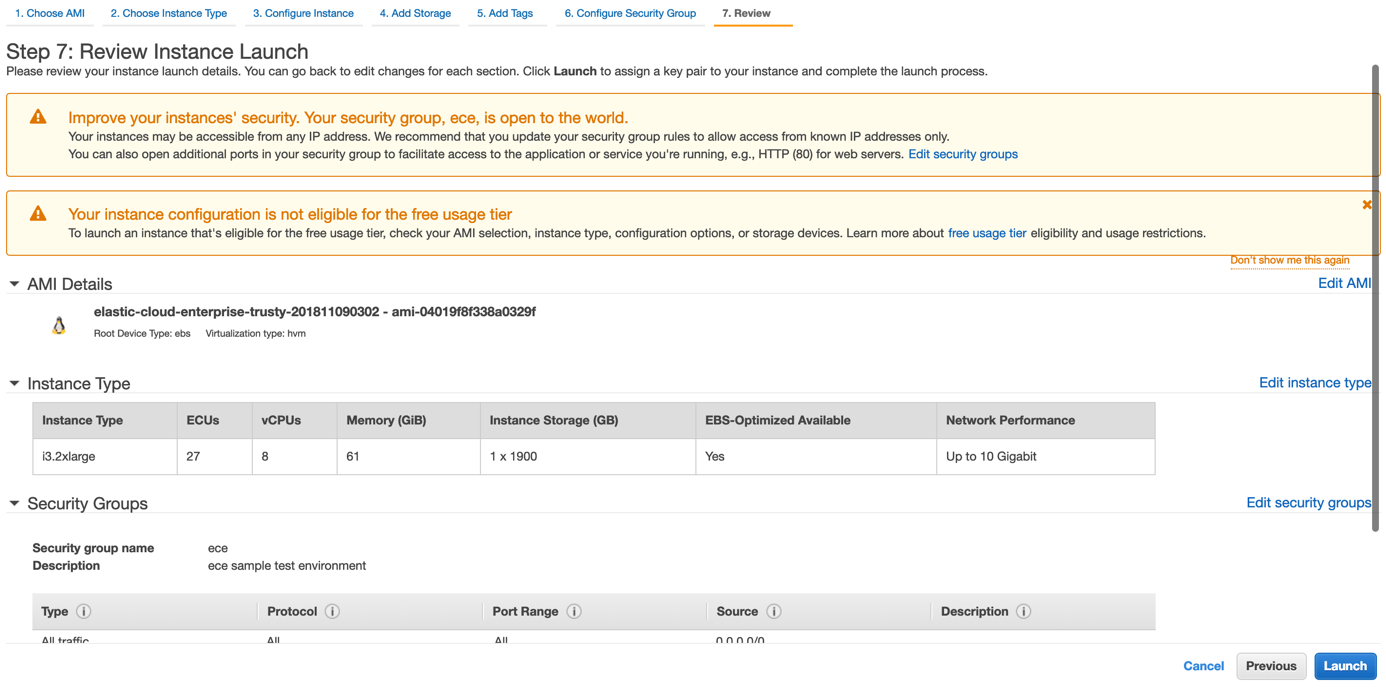
1. Delete the EBS storage. We will be leveraging built-in NMVe storage. Click “Next: Add Tags” to continue, optionally tag your new instances, and then click configure security group to continue.



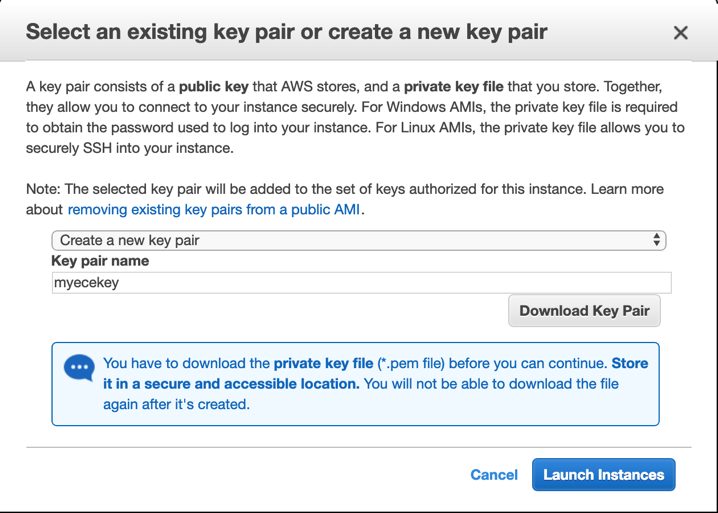
1. Create a new security group - give it a name and modify the single firewall rule type to “All traffic”. You can use the defaults that we have below. Keep in mind these defaults are entirely open to the internet. Click “review and launch” button.



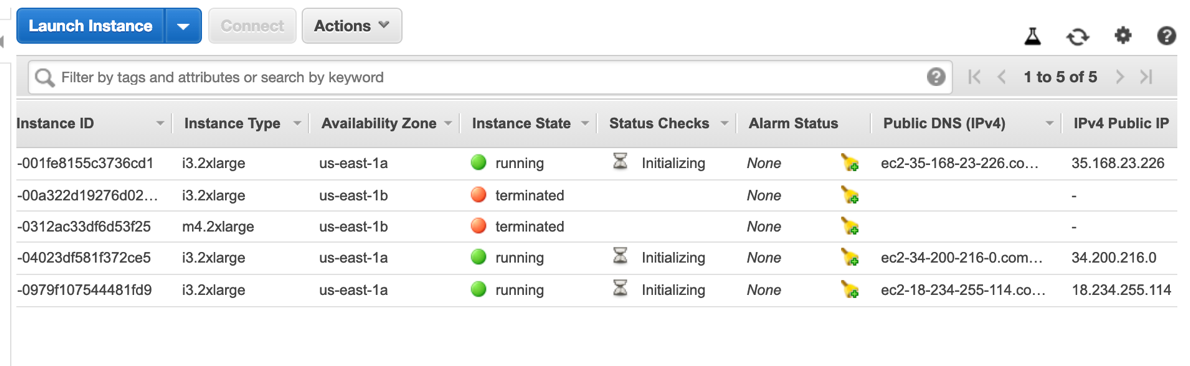
1. Review and click launch.



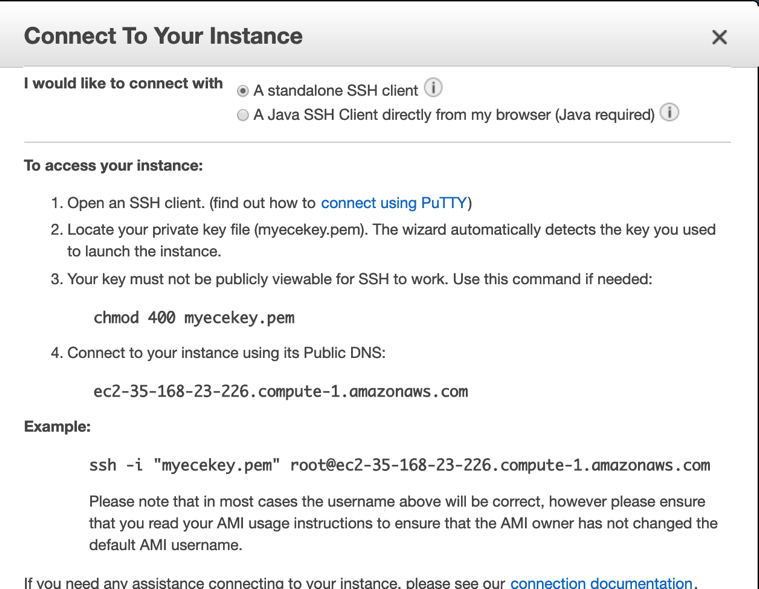
1. Create a new key/pair and download it. Then click “launch instances”. You can click on “view instances” to watch your instances start up.



1. Note the ip addresses of your instances when they spin up. **Keep in mind these instances are exposed to the internet for this simple lab experiment.** In a real-world environment, we would be much more careful with security.



1. Select an instance and click connect to collect instructions on ssh’ing in.



1. Launch terminal, enable the key/pair, and shell into the environment (if using windows consider using putty to tunnel into your instance with key enabled). If all goes well, you should be ready to proceed.

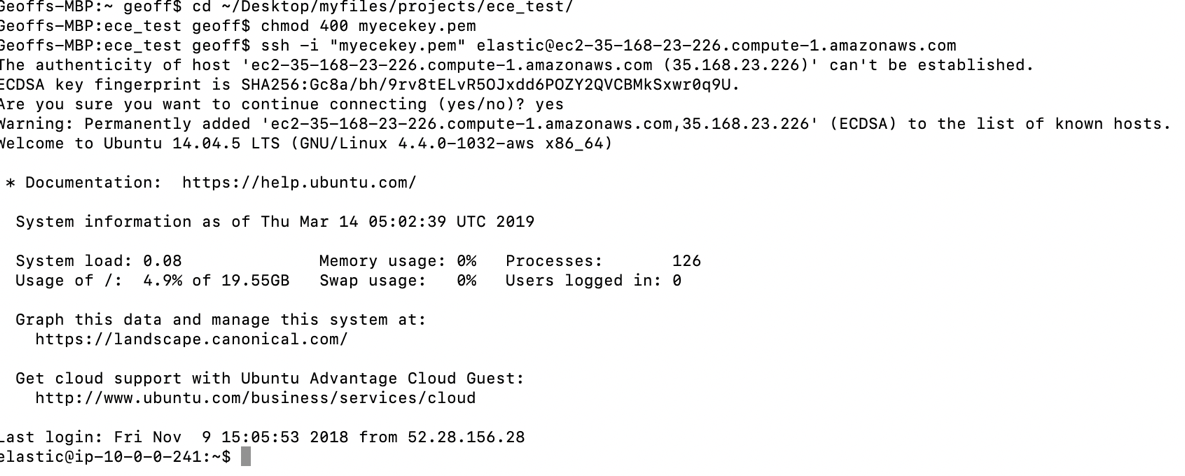
**For Mac:**

Commands: (note the user change to **elastic** as pre-built into AMI)

cd directoryofpem

chmod 400 nameofpem.pem

ssh -i "nameofpem.pem" elastic@ec2-35-168-23-226.compute-1.amazonaws.com



For windows: (**TODO**)

1. (**TODO**: make this script work for trusty – no systemctl ; **skip this step for now**) At this stage, we could make the filesystem xfs. This step could be skipped, but we want to be able to track storage space with ECE. Here is how we set up XFS for our single disk setup (we’ve commented out three other drives, so the script is handy for more disks when ready). Just run each of these commands line-by-line (**TODO**: put this on git and clone):

sudo pvcreate /dev/nvme0n1

#sudo pvcreate /dev/nvme1n1

#sudo pvcreate /dev/nvme2n1

#sudo pvcreate /dev/nvme3n1

#sudo vgcreate eceVG /dev/nvme0n1 /dev/nvme1n1 /dev/nvme2n1 /dev/nvme3n1

sudo vgcreate eceVG /dev/nvme0n1

#sudo lvcreate -i 4 -I 8 -l 100%FREE eceVG -n eceLV

sudo lvcreate -i 1 -I 8 -l 100%FREE eceVG -n eceLV

sudo mkfs.xfs /dev/eceVG/eceLV

sudo mount /dev/eceVG/eceLV /mnt/data

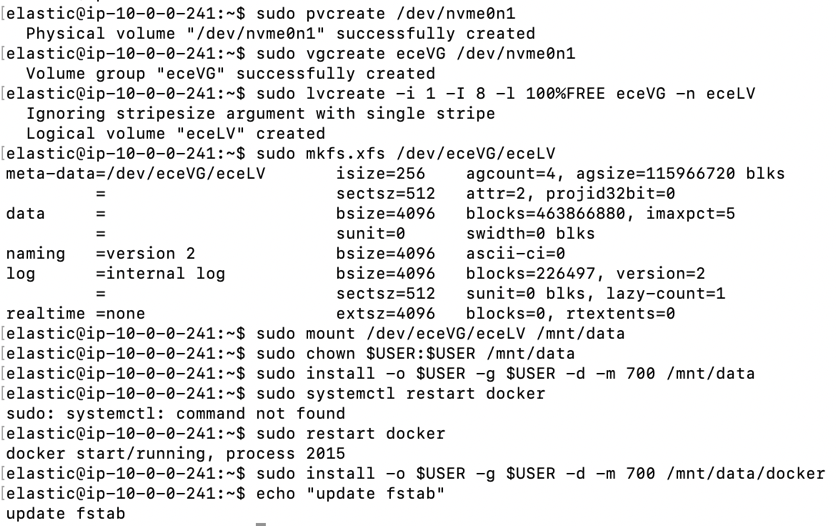
sudo chown $USER:$USER /mnt/data

sudo install -o $USER -g $USER -d -m 700 /mnt/data

sudo systemctl restart docker

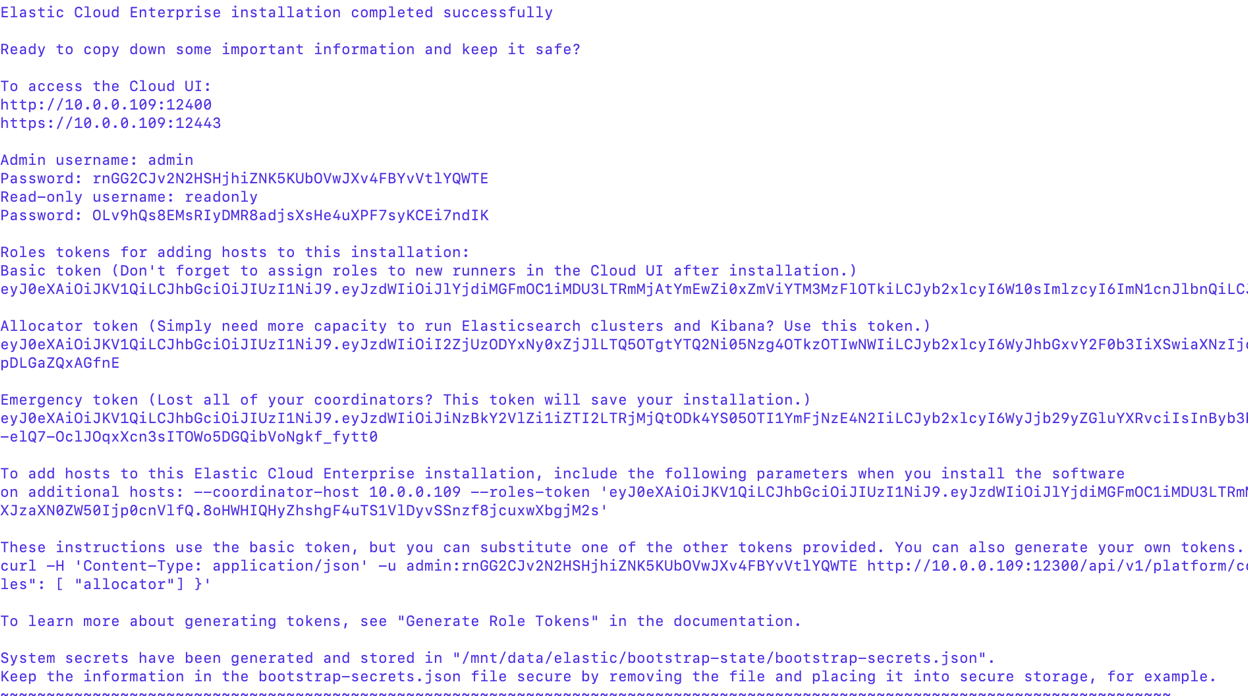
sudo install -o $USER -g $USER -d -m 700 /mnt/data/docker

echo "update fstab"

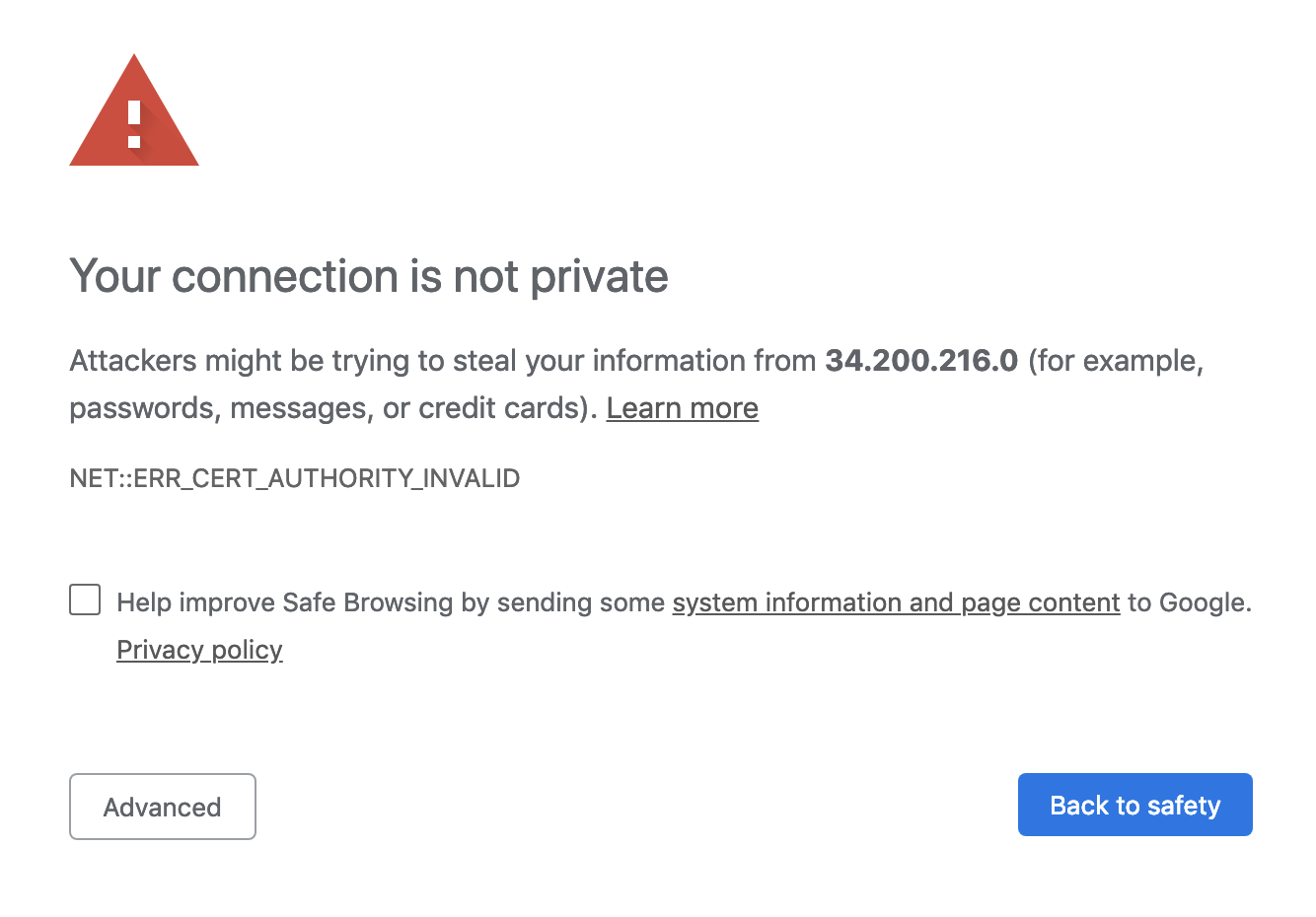


1. Now, we will actually install Elastic Cloud Enterprise (ECE). ECE will check the environment to see if it’s ready to go, including loading docker images. Next, ECE will proceed to install. Here is the [install](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-installing-online.html) script (it’s a one-liner!):

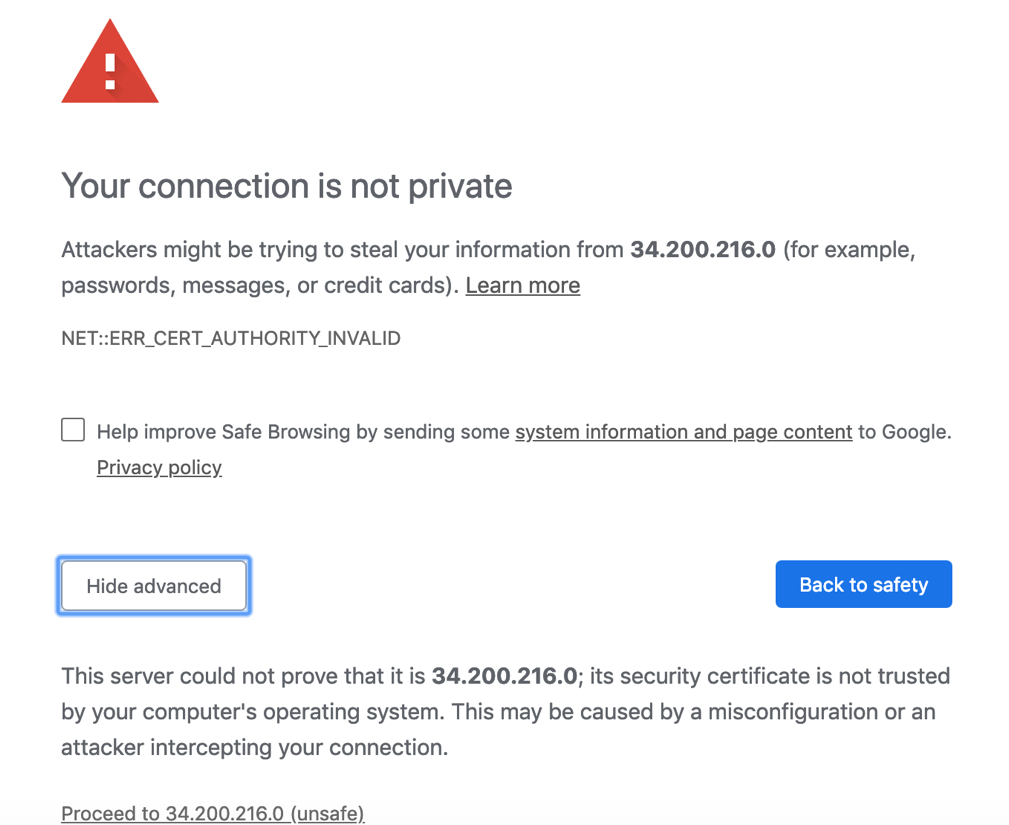
**bash <(curl -fsSL https://download.elastic.co/cloud/elastic-cloud-enterprise.sh) install**

P**lease sure to make a note of the provided URLs, usernames, passwords, and all tokens**; it is a good practice to copy that information and paste it somewhere safe (textpad or otherwise). 

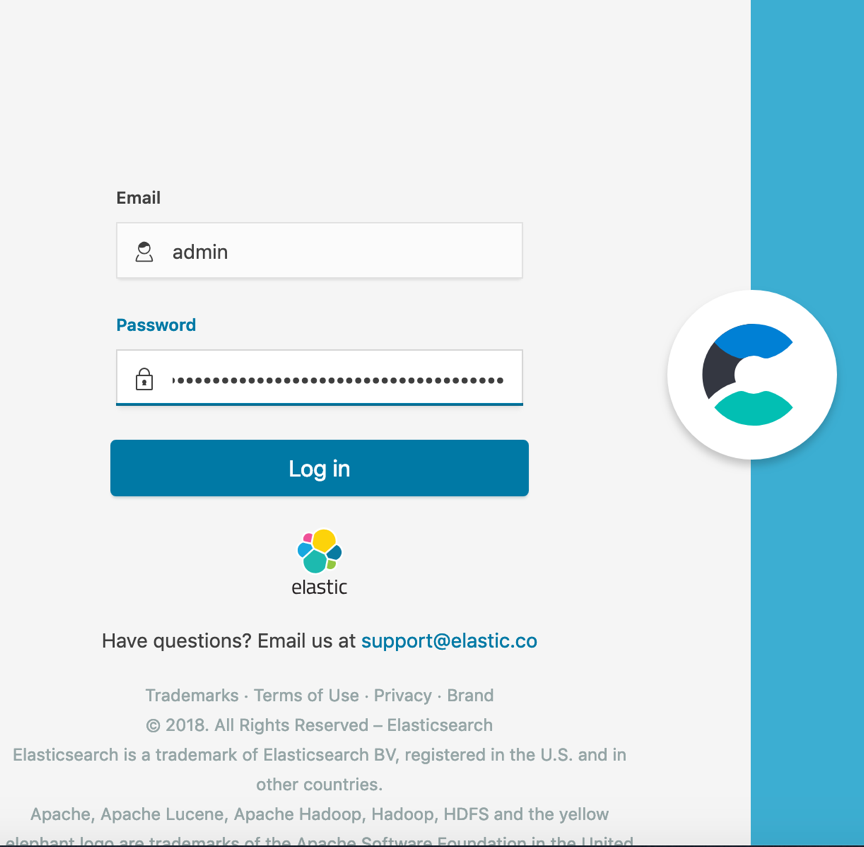
1. At this point, you should be able to go to the URL and pull up the ECE console. Please note the URL provided is the private IP address. The 10.0.0.x address needs to be replaced with the **external IP address of the instance**. In our scenario here, we took the IPv4 Public IP and pointed it at port 12443 (“<https://34.200.216.0:12443>”) Click on advanced on the warning browser: (taken from the AWS EC2 Console)



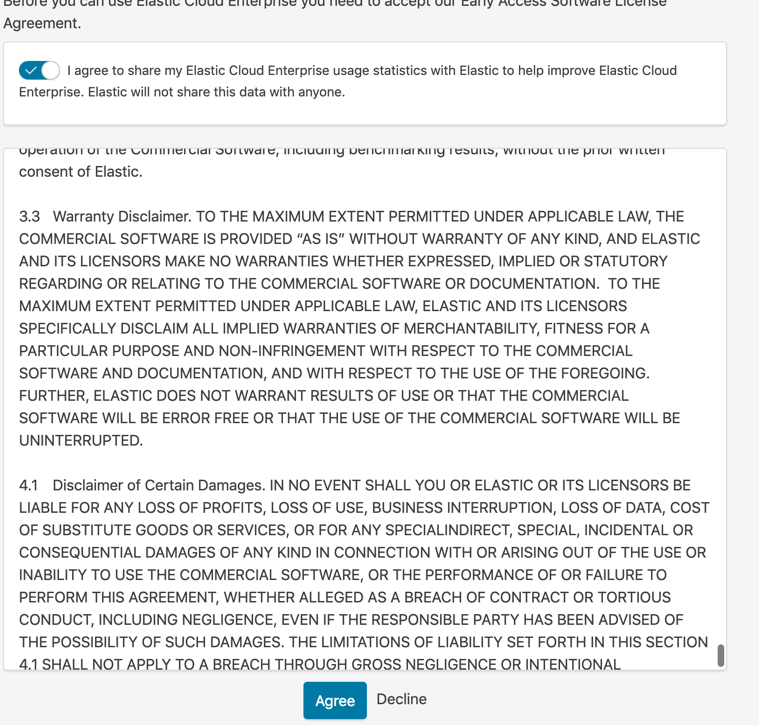
1. Good news: ECE is installed. We can take care of the ”connection not private” message by working with certificates and adding a load balancer. Go ahead and click on “advanced” and then click on the link provided, which should say “**proceed to….unsafe**”:



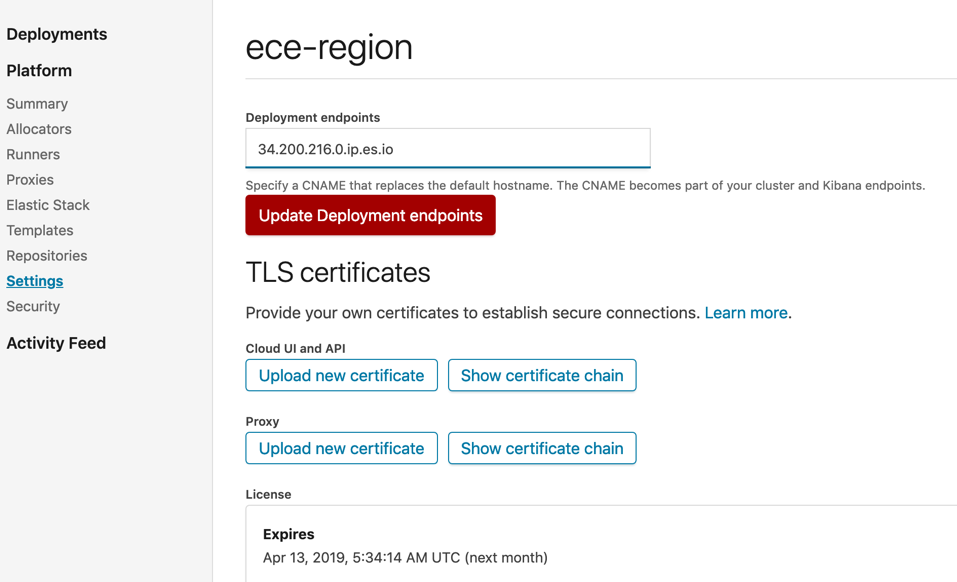
1. And wallah – the login page. Login with the admin credentials provided after the install of ECE; you will need the **admin username and the password** provided during the installation.



1. Agree to the license terms. You will need to slide to check that you agree and then click the **Agree** button at the bottom. You will now be in.



1. We can very easily set the default URL of clusters to the assigned external IP address by going to **Platform ---> Settings** and modifying the first four numbers in the deployment endpoints to reflect the external IP address. In a production scenario, we would be putting a load balancer up front of these instances and locking down networking, as previously mentioned; we might also add our own certificates on this screen. After updating deployment, click the “Update Deployment endpoints” button to proceed.



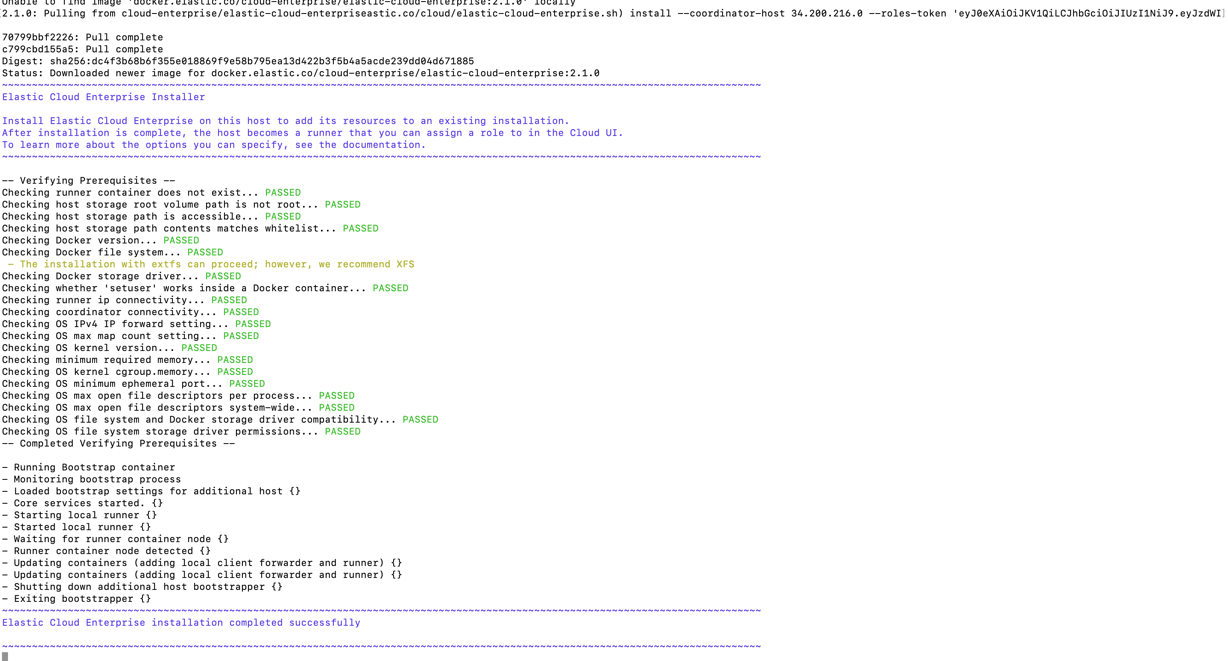
1. We could start building clusters now. However, in our case, we are going to add the other two AWS instances to the ECE install. In order to do that, we will tunnel into the other images,as in step 13 and run the same install script but with reference to the original host IP and the roles token provided during the first installation (to learn more about options when installing ECE, go [here](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-installing-online.html); to learn more about adding hosts, go [here](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-installing-online.html#ece-installing-additional)). For this simple install, we can use the **basic token** provided during the installation.

Here is the Linux command:

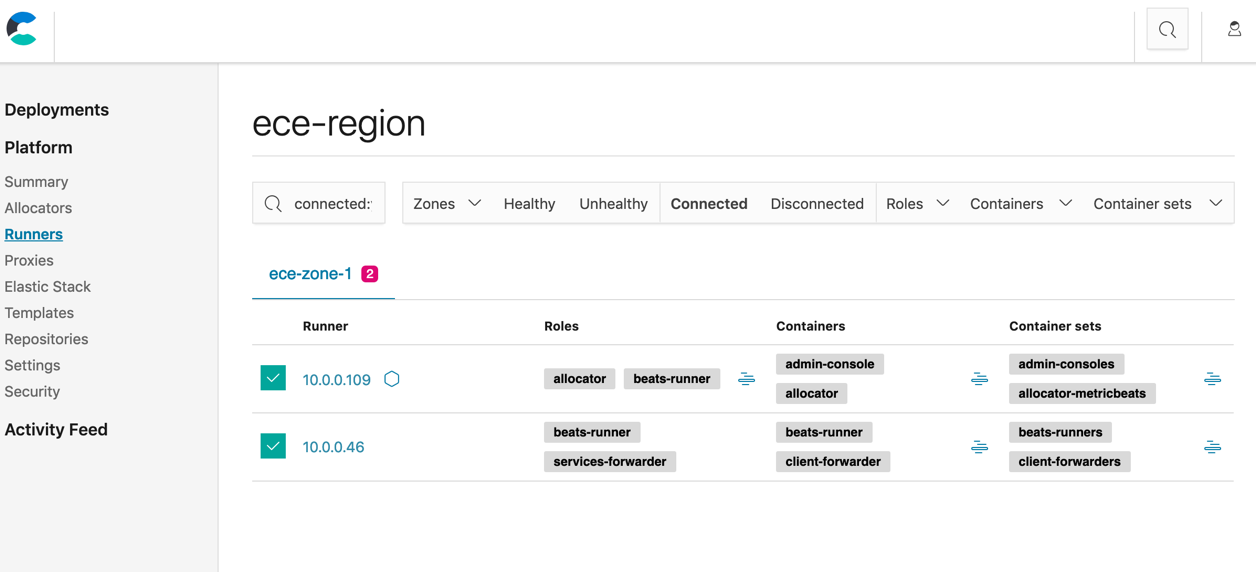
**bash <(curl -fsSL https://download.elastic.co/cloud/elastic-cloud-enterprise.sh) install --coordinator-host HOST\_IP --roles-token 'TOKEN'**

(to )

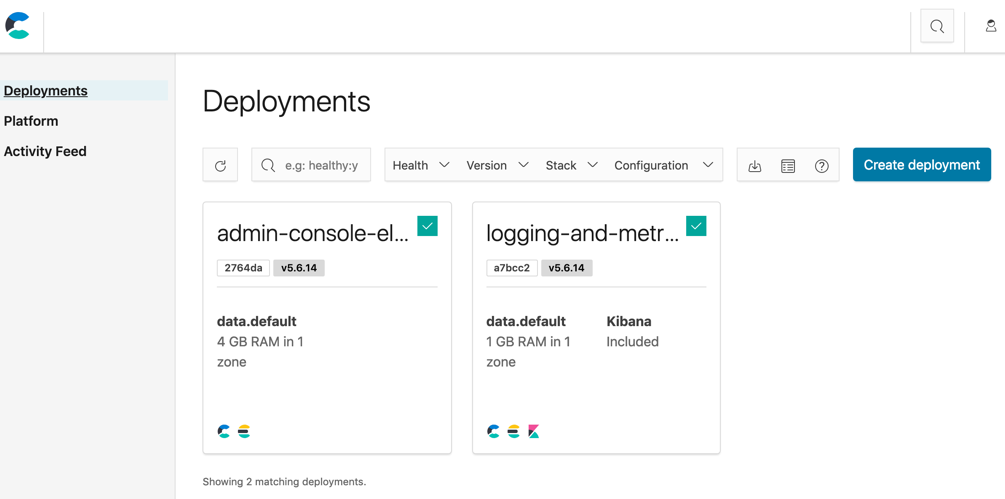
If you see the message “Elastic Cloud Enterprise installation completed successfully”, the install succeeded.

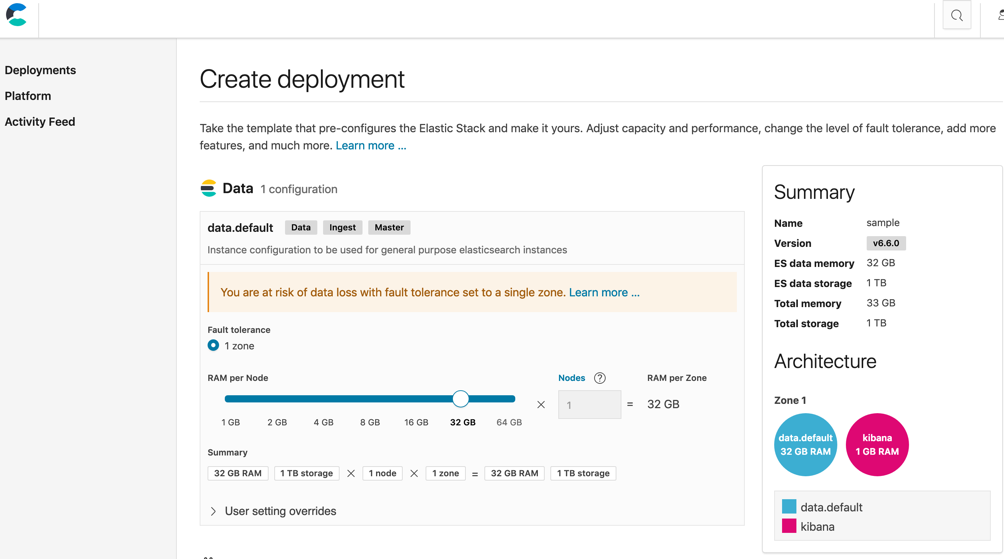


1. Next if you click to **Platform ---> Runners** inside the admin console, you will be able to see your second and third IP address. Keep in mind that this environment is not spread across availability zones but could be by adding subnets to the VPC, installing ECE in those new subnets, and taking advantage of high availability. Here is what the install we are doing looks like after two “runners” have been added:



1. Finally you can build out multiple clusters with your very own cloud managed service environment! You are in the driver’s seat and in full control of your managed service. Simply go to the deployments section and click on the **create deployments** button.





1. From here, there are other tasks which could be completed:

* [start ingesting, searching, visualizing data](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-stack-getting-started.html)
* [add hardware/hosts](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-installing-online.html#ece-installing-additional) (same as step 21 above)
* [specialize the role of hosts](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-change-roles.html) by updating the runner roles.
* [create a separate cluster and send it monitoring data](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-enable-monitoring.html#ece-monitoring_for_production_use) from all deployed clusters
* add servers for intense computation using Elastic’s [machine learning](https://www.elastic.co/guide/en/elastic-stack-overview/current/xpack-ml.html) capability (link)
* [configure new deployments options](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-administering-deployments.html) and specialize hardware, for search, hot/warm/cold logs, compute-heavy use cases
* deploy your organizations own [security certificates](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-manage-certificates.html)
* set up index lifecycle management
* add a load balancer up front, add a bastio host, and improve upon security groups, inbound/outbound traffic rules
* add subnets to the VPC in other availability zones and expand the ECE install for high availability

If you give the Elastic team a call, we’d be happy to help support the planning of what makes sense for your use case/s for a production rollout.

In the meantime, you might want to learn more about typical [playbooks](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-playbook.html) and [deployment templates](https://www.elastic.co/guide/en/cloud-enterprise/current/ece-configuring-ece-templates.html), which can help to cater to your architectural needs with the ultimate goal being to fully and optimally scale in production. There is a also a great on-demand course which can help to guide further learning.