

Welcome back and in this next lesson I want to focus on a particular type of elastic load balancer and that's the first type of load balancer that was available inside AWS the classic load balancer. So before I get started on the theory, I want to start off a couple of tasks that I'll be using later in this lesson. So I'm going to go to the VPC and the initial thing I'm going to do is make sure I've got a VPC in this account. Now I don't so I want to go ahead and create a default VPC. As you learned earlier in the course that just creates a VPC with some standard IP addressing along with all of the networking infrastructure that I'll need for this lesson. So now that I've got this default VPC could move across to the EC2 console. Now to demonstrate how load balancing works. I'm going to go ahead and create three EC2 instances. So I'll click on launch instance, and I'm going to make this as quick as possible. So I'm going to accept most of the defaults. Now for this demonstration, small instances will be fine. So I can either use the T2.micro, which is free tier eligible. So this won't cost anything to use this instance or I could pick the newer type, which is T3.micro. Keep things simple. I'll go a T2.micro, go to next instance details. Now, I'm going to leave everything is default apart from for subnet, I'm going to change this to availability zone 1a. Now I'm going to create three EC2. So I want to make sure that there's one in 1a, one in1b, and one in 1c. I'll scroll down, expand advanced details and then in this user data box, I'm going to paste in some user data. Now, I've created this and its present inside this lesson's folder in the course GitHub repository. Now there are three files bootstrap one, two, and three, and I'll use the first one for server one, the second one for server two, and logically, the third one for server three. So what I'll do is I'll paste the contents of bootstrap one into this user data text box. Now, all this does is set this instance up to be a web server, so installs the Apache web server, downloads some assets also from the course GitHub repository, and then just performs some finishing configuration. So it configures the permissions on the folders and it make sure the web server is set to started. Once I've entered that I'm going to go to add storage. I'll accept the default go to add tags, click on Add Tag. The key will be name, and then the value will be Cat web one, and I'll call the other servers Cat web two and Cat web three. Now that I've got that set, I'll go to Configure Security Group. I'll create a brand new security group and I'll call it Cat web. I'll put the same for description. I'll leave the existing rule that allows SSH but allowed a new rule that allows HTTP. So I got both of those, go to review and launch. I go to launch. I'll create a new key pair. I'll call it ELB, download that key pair, and then launch the instance. So that's going to create a single EC2 instance called Cat web one and I'm going to go ahead and do the same for two additional servers making sure that I use the correct bootstrap file for each of these service, so bootstrapped two for the next one and then bootstrap three for the final one. So while they're creating I want to talk about classic load balancer as a product. So classic load balancer is actually the oldest type of load balancer, and that's available in AWS and it's actually not recommended for any new projects. So AWS recommend when you create load balancers that you don't use the classic load balancer. The only situation that you should be using the classic load balancer in is if you don't have access to VPC. So you need to deploy it into EC2 classic situations. Now I have not even mentioned EC2 classic anywhere in this course, because it's something that generally doesn't come up in the exam. **So before VPCs existed, there was a previous method of deploying EC2 instances, which was known as EC2 classic. It was a lot less configurable. You didn't have access to anywhere near the amount of network configurability that you have with VPCs.** So if you hear the term EC2 classic mentioned, remember that it's a legacy method for configuring EC2 instances and **so generally AWS now only recommend classic load balancers to be used in situations where you have to use EC2 classic, which at this point in time should be next to no deployments.** You really should not be in a situation where you need to use classic load balancers.

Now the reason why they're not recommended is that classic load balancers are layer three and four devices. They essentially support the TCP and SSL or TLS protocols. Now they do have some ability to understand certain elements of HTTPD and HTTPS but this is generally restricted to be able to do health checks, which I'll talk about later in this lesson. For the most part, you can only perform decisions on any incoming traffic based on elements of the layer three and layer four protocol. So it really is a limitation, as you'll learn when I talk about the application load balancer. A lot of the functionality that we take for granted today is not available within the classic load balancer. So, **for instance, one of these is that with a classic load balancer you can only have one SSL certificate associated with that classic load balancer. So generally that means that every single application that you deploy into your environment needs its own dedicated, classic load balancer. So if you had 10 websites that you were looking to host each with their own DNS name and thus their own SSL certificate, you'd need 10 individual classic load balancers. So that's a real limitation, because it means that in almost all cases, it is more expensive to use a classic load balancer than any of the other types**. So that's another reason why it's not recommended that you use classic load balancers. Now this is the architecture that a classic load balancer uses. **You go ahead and create a load balancer, and I talked about in the previous lesson. This creates a node inside each availability zone that you elect to use and it's that actual node that does the balancing connections across backend instances**. So let's look at how this works. So now we've got our three cat web web servers, so I just want to load each of those in a brand new tab just to demonstrate that they are unique websites. So first, I'll open cat web one. So here we go we've got cat web one. So it's got to cat picture and it's got an IP address and this IP address is actually the public IP address of the EC2 instance. So now if I load cat web two. So I'll copied that DNS name into my clipboard and open that in a new tab. So this is cat web two it's a unique cat picture, so I can tell it apart and it's also got the public IP address of this EC2 instance and the same logic is true for cat web three. I'll copy the DNS name of cat web three and open that in a new tab and again unique cat picture unique IP address. So we've got cat web one, cat web two, and cat web three so moving back to the EC2 console, we create load balancers from the load balancing section. So load balancers are actually a component of EC2. So I'm going to go to load balancer and then create a load balancer. So you've got the three different types that I'm going to be talking about over the next few lessons but this is the one I want to focus on in this lesson. It's the oldest on the least feature rich of all of the load balancers available and it's grayed out, and it's even called out that it's a previous generation. So again, it is not recommended to use a classic load balancer for any new deployments but I do want to demonstrate the functionality that it offers so I'm going to click on create. Now you need to start by providing the load balancer with the name, so I'm going to keep it simple and call it CLB for classic load balancer. Load balancers exist inside the VPC, so the first step is picking the VPC they want to use. I'll be using the default VPC that I created at start of this lesson.

***Now load balancers come in two different types. We've got an internet facing load balancer and an internal load balancer. The difference between those two is that with an internet facing load balancer, the nodes of the load balancer that get created have publicly addressable IP addresses. With an internal only load balancer, the nodes have private only addresses.*** Now that's an important distinction to keep in mind. You generally use an internal load balancer **when you wanted to position it between application tiers**. So if you had a frontend tier of web servers and behind those web servers, you had a tier of application servers. You could position the load balancer in between those tiers to abstract them away from each other. So a web server could talk to the internal load balancer and be directed towards a specific application server and it wouldn't care which application server it's talking to. It wouldn't care if it was changed if it talked to a new application server. So load balancers are generally used between tiers of an application.   
**Internet facing load balancers generally the things that are presented at the front of your application stack. So between your application and your customers** and as I mentioned in the last lesson, you do that because it abstracts away from your underlying infrastructure, so your customer doesn't need to care how many EC2 instances you have providing your application, you can put a load balancer in front of them, and then load balance any incoming connection across your instances which provide that service. So that's the architecture that all load balancers provide. Whether you pick an internal only or an internet facing just depends on whether the load balancer is going to offer services to the public or not.

Now, for this example, I'm going to keep it simple, So I'm going to create an internet facing load balancer but what I am going to do is enable the advanced VPC configuration because I want to talk about exactly how the subnet selection works. **Now, there are two really important parts that you need to understand about these load balancers. The first is the listener configuration. All load balancers are essentially configured by defining what ports and protocols that you'll listen to**. So in this particular case, I could configure this load balancer to accept connections on HTTP using port 80. I could also change this and have the load balancer accept connections using secure HTTP on port 443. Now, if I do that, I'll need to apply an SSL certificate so that load balancer can actually talk using this encrypted connection. So, for this lesson, I won't do that I'll just select HTTP but it's worth keeping in mind that the load balancer frontend protocol and the protocol that's used to communicate with the instances can be different. I want you to consider the scenario with this diagram on the right of my screen, where you've got four independent instances, A1 and A2, B1 and B2. Now let's say that you want to provide a secure HTTPS connection from your customers to these web servers. Well**, if you don't use a load balancer, each of these instances will need an SSL certificate installed and configured on the web server and that's quite a lot of admin overhead. It also means that each of these servers needs to perform encryption and decryption operations to communicate using HTTPS. Now, one option that you do have is a process called SSL offloading and that's something that most load balancers support. Now an example of SSL offloading is where we could set this to be HTTPS. So the connections between our customers and this load balancer are using secure HTTP but then the instance protocols so the protocol that's used for the load balancer to talk to these backend instances is HTTP, that means that all of the encryption and decryption is happening on the load balancer. We only need the load balancer to have the SSL certificate and these backend instances could just talk over normal HTTP. They don't need the SSL certificate and they don't need any of the admin overhead or any of the encryption and decryption CPU cycles that will be used to perform those operations.**

Now, this is important to understand for exam it often comes up as an exam question. **If you need complete encryption from start to finish, then you do not use this process but if you want to implement a system where encryption is used to the edge point of your network, the perimeter of your network, then you can configure HTTPS here and then configure HTTPD for the backend and by doing that, you'll reduce the CPU cycles required on these instances so you can use smaller instances and they can serve more customers. You don't have to worry about any of the admin overhead, and the configuration of these instances is massively reduced.** Now, I won't be doing that in this lesson. I want to keep it simple. So I'll select HTTP for the frontend or listener protocol and port 80 and then for the backend also the same. So HTTPD and port 80. Now, the subnets that you define in this part the configuration are actually the subnets where the load balancer nodes live and so, if I wanted to provision a highly available load balancer I need to select multiple availability zones or subnets. So I'm going to do that. I'm going to select U.S. East 1a, U.S. East 1b, and U.S. East 1c and **that means that this load balancer that will be created with three load balancer nodes.** So now that I've said that I'm going to move to security groups.