

Now when you create a VPC, that's not actually enough because services don't go directly into a VPC. Private services generally operate inside a network subnet, which is inside the VPC. **A subnet is an isolated local network and a VPC can have many subnets.** To compare this to the physical world, imagine a VPC is like a big data center, and a subnet might be one floor or a component of one floor. **Subnets have created inside a specific availability zone so they can't span availability zones** and in this case, this subnet is operating in the U.S. East 1a, it's not operating in 1b, and it can't span across to 1b so each individual subnet is homed to one specific availability zone. **Now an available zone can have many subnets, but a subnet can only ever be in one availability zone.** So this means that for this VPC that I've created in U.S. East 1 and let's say that I want to use two availability zones inside this VPC. Well, at a minimum, that means I'll need two subnets, one in each of the availability zones. If you recall earlier the default VPC that gets created, this creates a subnet inside each availability zone. So in this case, this region has six availability zones and so I've got six individual subnets inside this VPC. If, like on this diagram, I want to operating two availability zones then at minimum, I need to create one subnet in U.S. East 1a and one subnet in U.S. East 1b a single subnet cannot occupy both availability zones. One subnet for one availability zone.

Now, if you have different application tiers—so for example, different configurations or security levels. So **I've got a public subnet, a private subnet for our applications, and private subnet for our databases then you'll also need individual subnets for each of those tiers and if you operate that same stack of application tiers in multiple availability zones, then you'll need to mirror that inside that availability zone**. So in this example, we've got three tiers public, private, and database. So in two AZs that means a total of six subnets.

Now, you learned earlier in the course that subnets can only be broken in half, so a network can make two smaller networks, four even smaller networks, four even smaller still networks, and 16 tinier ones and so on. So you'll need to break a network down in that way, so break it down by two each time and so if you need six subnets, you'll need to have broken down your VPC CIDR in to eight or more smaller networks. So keep that in mind. You can only break things down into twos when it comes to network subnetting. **So if you've got a number of subnets in this case six that isn't aligned to that 2, 4, 8, 16, 32 and so on, then you'll need to add buffer.** So in this case will need to break our VPC down into eight and then essentially not use allocation for two and just configure these six subnets. **Now, if you wanted to add another availability zone you also need to leave spare capacity for expansion.** You might want to add another availability zone. So with this architecture, that would mean nine individual subnets. You might want to add another tier to this three tier application, so that would need another two subnets, or you might want to use both. So it'll be another application tier which would take us to eight and then another AZ which would take us to 12 and if you wanted to make 12 subnets because you can't break a subnet down into 12 you'd need to pick 16. So planning a VPC is essential.

Historically could only set the VPC CIDR once when you created it, but now you can add ranges to it so I could select this VPC and then click on the actions menu and then edit CIDRs and add additional CIDRs to this VPC. So while it's slightly less important to plan this in advance, it is always advisable to think about these things before you create the VPC. **Now subnets have the same size limits as this size on the VPC, so you've essentially got a maximum size of /16 and a minimum size of /28.** Which means in theory you could have one big subnet inside of EBC but now you know that that would have to occupy one availability zone, so you probably couldn't manage any form of high availability. Now, **subnet CIDR blocks cannot be bigger than the VPC. They cannot be created outside of the CIDR of the VPC and the subnet CIDRs cannot overlap with other subnets inside that same VPC**.

So what I'm going to do now is go ahead and create subnets inside this VPC. So I'm going to go to subnets. Now, in this diagram, we've got 10.0.0.0/16 for the VPC and we could since we need nine subnets, chop this in half to two seventeens again to four eighteens and again to eight nineteens if you wanted the buffer to expand, then we could go one further and chop it into 16 /20s but there's nothing to say that we have to pick the largest subnets we can. We could use /24 subnets and they're generally really easy for human beings to use and so, generally, if you don't have requirements that go past /24 I would always recommend using /24 for subnets and that's what I'm going to do. I'm going to go ahead and create a /24 subnet. I'm going to hit create subnet and I'll be creating this subnet. So it's the public subnet in U.S. East 1a. So for the subnet name, I'm going to pick sn for subnet, hyphen, and then public because it's going to be a public subnet and I'll explain how we configure that later in this topic and then hyphen and then a for availability zone A so we know this subnet is living inside availability zone a. Now, subnets belonged to a VPC. So I'm going to need to select the VPC demo that I created at the start of this lesson and because the subnet is in a particular availability zone I'll need to specify that. So I click on the dropdown. You'll note that we've got two different things in this dropdown. We've got the availability zone name and the availability zone ID. This is a useful point to introduce this. Each availability zone inside AWS has an availability zone name. So U.S. East 1a is a good example but what I see is U.S. East 1a in my account might not be the same as what you see as U.S. East 1a in your account because AWS randomized this to make sure that everybody uses all of the availability zones equally. So this name is specific only to my account. The ID, though, is consistent across accounts. So in this case, use1-az4 is the particular physical availability zone that U.S. East 1A maps to. Now it's rare that you'll need to use this ID. You'll only use it if you need to coordinate between different accounts, but it's worth knowing what this concept actually does. So I'm going to go ahead and pick one a for this particular subnet. So I'll go ahead and do that and for the CIDR range, I'm going to pick 10.0.1.0/24 which gives us 256 IP addresses, so I'll go ahead and enter that so 10.0.1.0/24 once I've entered that I'll hit create? It will create that subnet and I'll click on close and there we go.

It's created and it's inside the VPC demo VPC but note that it's only got 251 available IP addresses. So why is this? Well, every subnet that you create inside AWS has a number of reserved IP addresses. To start with, we've got the first and last IP addresses that can't be used. This isn't specific to AWS. The first IP address so .0 in this case is known as the network address. So 10.0.1.0 in this case and then we've got the last address, which is the broadcast address, and that's .255 in this case. **So the first address in a subnet and the last address in this subnet are always reserved.** That's not specific to AWS. The broadcast address could be used to send IP packets to all devices in a subnet. This actually doesn't work in AWS, so that's important to know for the exam but the address is reserved anyway, so that you don't use it. In addition**, we've got three other reserved IP addresses. We've got what's known as the network plus one address. So this is the IP immediately following the network address. In this case 10.0.1.1 and this is used for the VPC router.** So this is the router that routes traffic between the different subnets inside a VPC and if you configure any external connectivity, it routes traffic in and out of the VPC. **The next one that reserved is the .2 address. So in this case, it's 10.0.1.2 and this is used for DNS within the VPC.** So if you want to resolve a DNS name to an IP inside the VPC, that's what's used and finally, w**e've got the network plus three address so .3. In this case it's 10.0.1.3 and that's reserved for future use**. **So that's a total of five reserved addresses in every single subnet.** So whatever size your subnet is, you've got to account for these five addresses that you won't be able to use. So in the case of this /24 network, we should have 256 we've got the five addresses we can't use, so that gives us an available pool of 251. Now, why this matters to you as an architect is you need to make sure that whatever you decide on for a VPC and then carve up for your subnets has enough room to grow and also that your subnets, when you remove these reserved addresses, has enough capacity left over to use for ever services you want to deploy into those subnets. Now, this matters more for smaller subnets because certain AWS products I'll be explaining later in the course do have a minimum number of addresses that they'll need to be able to use before they'll even deploy. So it's important to keep this in mind. At this point what I'm going to do is go ahead and quickly create these other five subnets inside this VPC. So that's the public subnet in 1b, the private app subnet in 1a and 1b, and then the database subnet in 1a and 1b. So I'm going to go ahead and create these on going edit out the gaps as I do it, and then we'll resume the video normally once I have completed that, and we're ready to move on we've finished creating these subnets. So we've got public A public B application A application B and database A and database B. I'm mirroring the structure that I've got on this diagram on the right of my screen.

I want to point out one specific thing, and that is that I've started my addressing using 10.0.1.0/24. That's not actually the first range available in the VPC. I could have used 10.0.0/24 but sometimes it pays to be less efficient. In some cases, VPCs and services inside AWS are all created using automation, so human understanding is less relevant. In some cases, though, humans do need to understand the networking structure that you use inside a VPC. So what I've done here is match the number to an availability zone and to an application tier. So 10.0.1.0 is AZ A and then two is AZ B, and then I've incremented by 10. So we've got 11.0, which is in A and12.0 which is in B and then of incremental by 10 again. So 21.0 is in AZ A 22.0 is an AZ B. So by creating logical rules like this, it can help humans, especially people other than you understand exactly how your VPC is structured and this is especially valuable for people with less IT experience.

first one is **a DHCP option set**. So this is essentially a configuration that sets various things that are provided to resources inside this VPC when they used DHCP. **So DHCP is a protocol that allows resource is inside a network to auto configure their IP addresses and it's this that allows EC2 instances, for example, to get their own private IP address.** Now you're able to create a DHCP option set that allows you to specify a lot of the granular configuration that machines or instances will receive when they do a DHCP request and then **you're able to associate one DHCP option set with a VPC. You can't edit these options sets, so the only thing you can do is create an additional one and then associate that with a particular VPC and a VPC can only have one option so associated with it**. So remember for the exam, **you can't update one of these options sets. You have to create a new one and then change which one is currently applied to a VPC and when you do that, the next time any resources attempt to auto configure the network card, they'll get these new options**. Now lastly, I'll be covering this in a lot more detail later in the course but it used to be that both the VPC and this subnets were isolated to one specific AWS account. That's no longer the case. You're able to select a subnet and actually share that subnet with other AWS accounts and at the time of creating this lesson, it was limited to only other AWS accounts, which were inside an AWS organization but **when you share a subnet with another AWS account, you own that subnet. So no other AWS accounts can make any changes, but they can deploy resources into that subnet.** **Now the account that deploys the resources owns those resources so you can't update their resources and they can't update your subnet but it does become possible to essentially share a VPC and it's subnets across different AWS accounts within an AWS organization.**