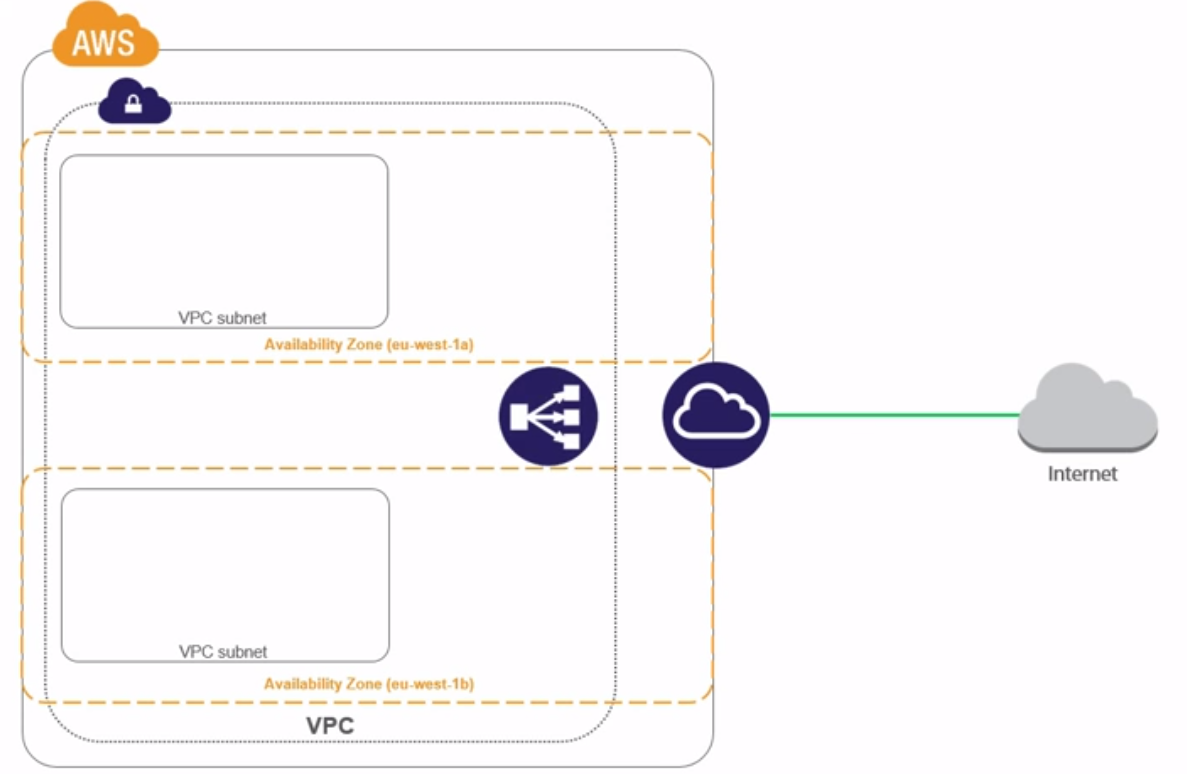
**Course Introduction**

Amazon Web Services-- coolest and most important technologies platforms in the world.

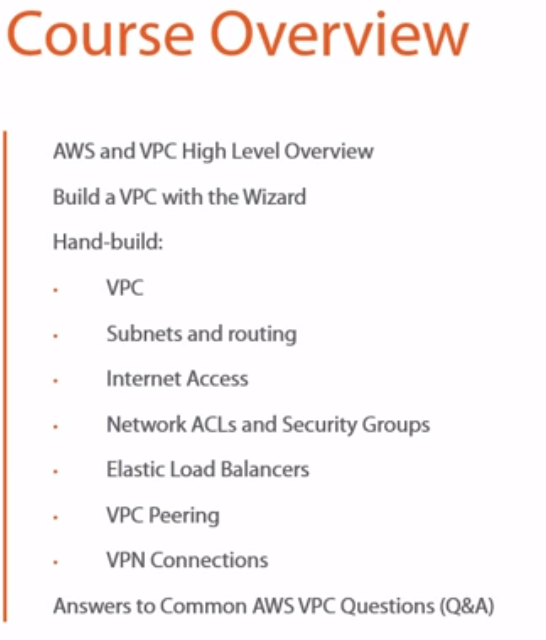
Using AWS and VPCs you can literally build yourself your own world class data center for virtual machines in the cloud

VPC🡪like that full working secured data center in the cloud

of highly available, high performance, running virtual machines.

We'll build multiple networks or subnets in different locations for high availability, we'll configure high speed, highly available internet access, and we'll build load balancers and VPN connections

**Introducing AWS and AWS VPCs**



Amazon was running probably the biggest online retail website in the world and the website itself was the crops of their business...

It was high performance, highly available, all of that stuff, and it had spare capacity. 

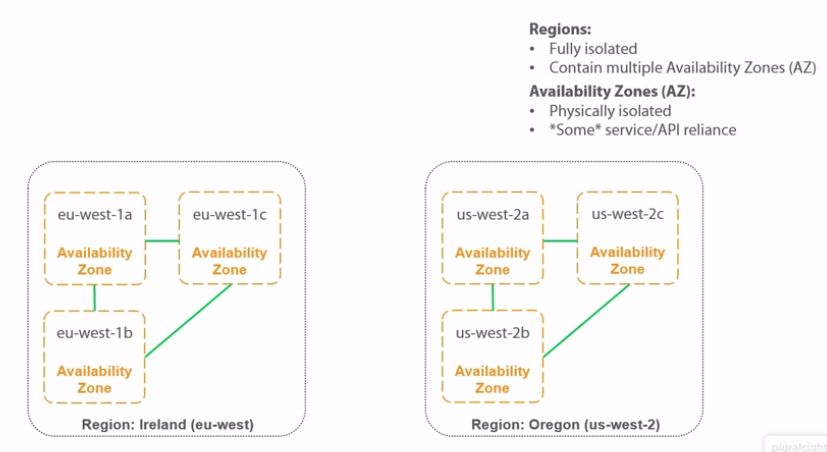
Why don't we sell some of this spare capacity to customers??

Infra. On rent

* Any business that needs to grow, but you don't want to lay down a bunch of capital to build your own data center and buy your own servers and storage, instead you can just go and rent some of it from Amazon. They probably build and manage it better than you do.
* Okay, well let's say business is good and you're expecting a spike in business next month and you think maybe you need to double or triple or quadruple your needs, but then you're projecting that maybe a month or so after that you don't need it anymore. No sweat. That's what this stuff is all about. **It's elastic.**

You can just spin up more virtual machines or consume more storage space or whatever and pay for what you use and then if you stop using it, you stop paying.

* AWS as a cloud platform, architected into regions and availability zones (AZ). So, at a global level AWS has got data centers in strategic locations around the world. These locations then get called regions.
* And now to every region Amazon offers certain AWS services. Regions closer to where you are going to be faster. Every region is fully independent, so fully isolated from all other regions. Meaning, if a service in a region or in fact if the entire region itself goes down, other regions shouldn't be impacted.
* Each region we've got availability zones. If we use Ireland, EU-West as an example, it's got three availability zones. Each connected, yeah, by a high speed, low latency network links.

 Now then, each availability zone is physically isolated from the rest, meaning if one of them goes down, I don't know a power issue or rock falls from space onto it, whatever, other availability zones in that region won't be affected.

But, and even AWS has buts, there have been plenty of services outages, okay, where issues have started in one availability zone and rippled over into other availability zones, causing the effected service to choke and become unavailable across the entire region.

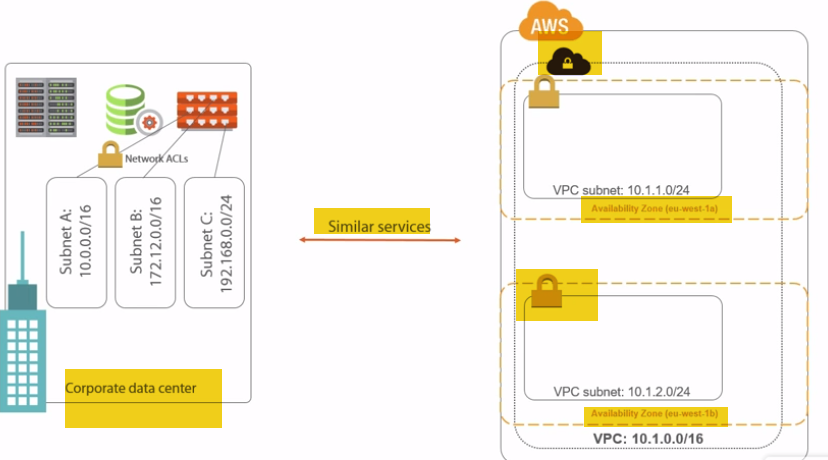
Availability zones are highly available and isolated from a physical perspective.

**What is VPC?**

When Amazon was first launched, like back in the mid 2000s, There was just the S3 object store and the EC2 virtual machine platform.

Most established companies were used to their own data centers where they owned the service, storage, and networking and they were in full control of the lot. Then enter EC2 as a virtual machine computing platform and suddenly not only were the servers and the likes not theirs, **but they had precious little control over them too, especially in the networking space.**

So it was like, spin up an EC2 instance, a virtual machine, yeah, question, what's its IP address going to be?



Lots of shortcomings also, yeah. Well here enter VPCs.

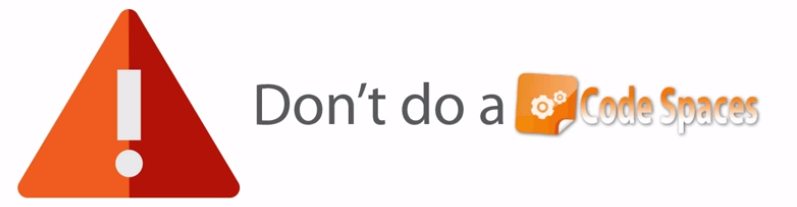
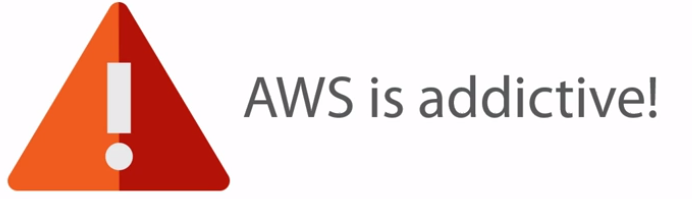
1.) We get to choose the network range for our entire VPC and then we're free to carve that up into as many subnets as we want. But not only that, we can put these subnets in whichever availability zones we want and that is massive, right.

Ex: I want a couple of web servers for performance and availability and then choose to deploy one into a subnet that let's say in availability zone 1A and the other into a subnet that's in availability zone 1B. Then if 1A goes away, so like the whole availability zone or maybe just the EC2 service, yeah, who cares, we're cool. 1B is still up and we're still in business.

2.) But it's not just that, on each of these subnets, we can lash **network access control lists** just like we do on routers and firewalls in our corporate data center networks and before long..

This is starting to look a whole lot like a regular corporate or data center network, which is a huge driver for adoption. Now bunch more customers are going to deploy it**.**

**A Couple of Warnings**

 They were hosting everything on AWS, so their production environment and all its backups. Okay, I get that, it makes life easy, yeah.

A single AWS account and you've got access to everything, makes day to day ops nice and easy, right.

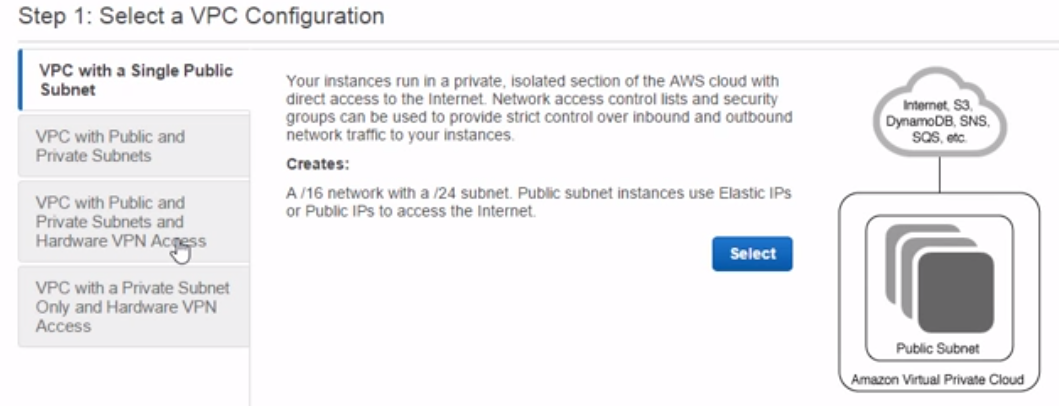
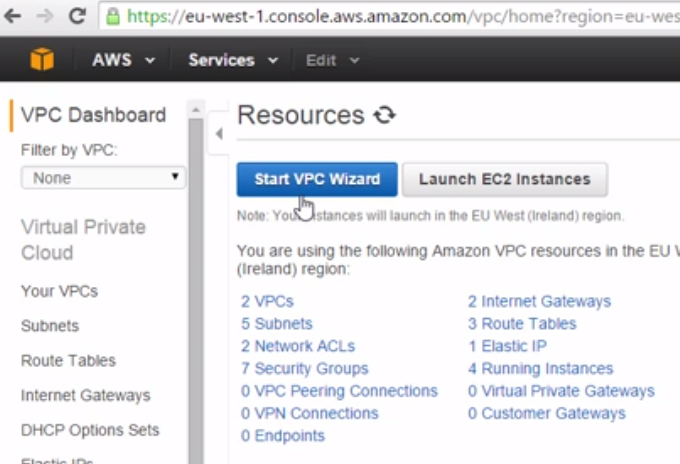
Though, I'm sure you've probably twigged, it also makes it dead, dead simple to trash the entire thing.

gained access to the Code Spaces AWS account and well, basically deleted the company production services and all of its backups.

multifactor authentication, separate accounts for different roles, proper identity and access management, maybe even hosting backups or copies of backups elsewhere. There are a ton of things that can be done to mitigate disasters like this and in truth...

B**uilding Our First VPC using wizard..**

Network🡪VPC, four options, only it's pretty much two options with just a slight variation on each.



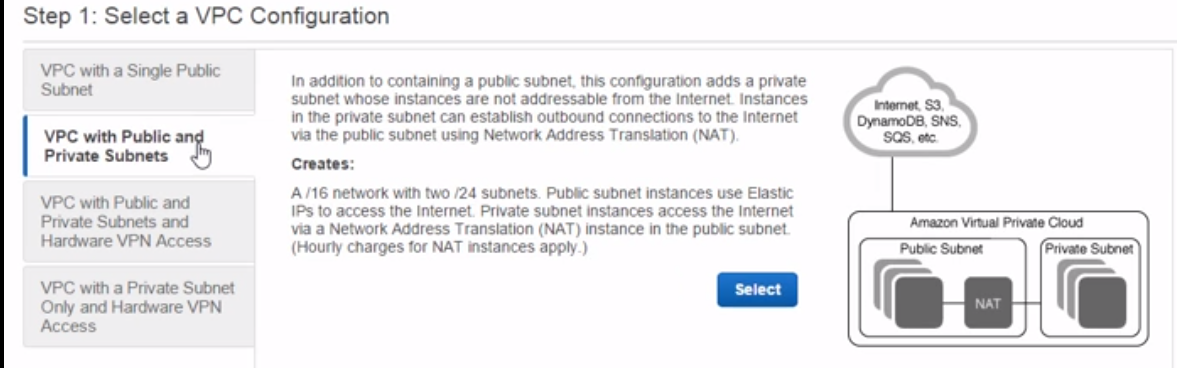
What am I talking about? Well, we've got VPC with a Single Subnet up here, then underneath we've got VPC with Public and Private Subnets. And then the two options below down here, these are just the same, but with a **VPN bolted on.**

(option 2) We'll have a public and a private subnet..

This outer rectangle🡪 this is our entire VPC, our entire virtual data center that we're building, then within it we're creating two subnets, a public one, and a private one.

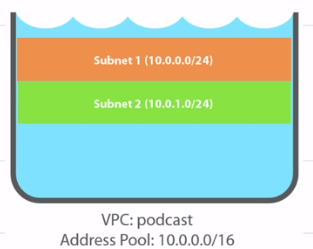
Now the major difference is that a public subnet is going to have direct access to the internet. We're going to be able to connect out to the wider world as well as accept unsolicited incoming connections. The private subnet on the other hand isn't..

**Configuring VPC and Public Subnet Options**



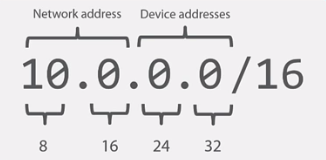
CIDR block(think as master n/w)--this is where we define the network or the network addresses from which all of our subnets will take their addresses from.

Master n/w, where pool of addresses that all of our subnets will then be a slice of... We choose here at the VPC definition level, our subnets have to draw from.



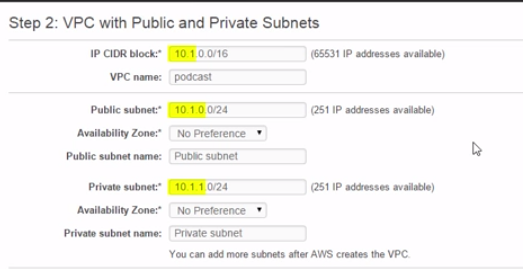
VPC name (say) podcast.

IPv4 addresses like these are broken down into 4 octets, 4 sets of 8 bits, 8, 16, 24, and 32.

And then by slapping /16 onto the end here, we're basically saying the first 16 bits or the first 2 octet’s🡪 network address.

I don't know, maybe like the city name in our postal address 10.0 city, then the remaining 2 octets are for street and house numbers of any devices that we put onto the network.

So the number after the /16 here is telling us how many bits constitute the network portion of the address, then we're free to play with any bits that are left over to the right of that, in this case the last 2 octets, yeah.



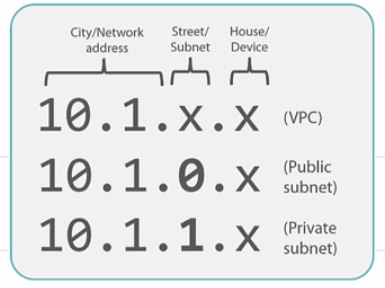
I'm going to go with 10.1 as the network for my VPC here.

Then down in our subnet we're going to have 10.1.0 and 10.1.1. Okay, still loads of red text. So, if we stick /24 onto the end of it, what we're basically saying okay is that the network portion of the addresses for our subnets is actually the first 24 bits...

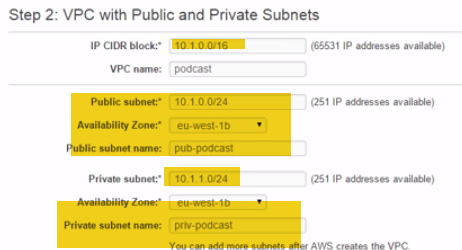
So 10.1.0 and 10.1.1 and then because they're both slices of the bigger 10.1 network that we've got up here in our VPC, the scary red text goes away and everything looks good again.

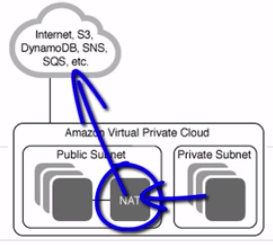
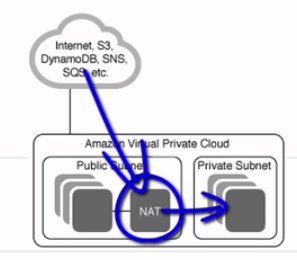
In proper layman's terms here, right, any internet router that sees packets with addresses like this is just going to drop those packets. Lets see how AWS manages it..

**VPC are per region. So an entire VPC exists inside of one region and only one region.**

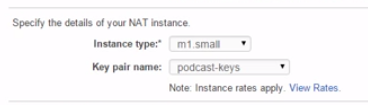
**Configuring Private Subnet, NAT, and DNS Options**

Any devices that we put into these subnets, so containers and EC2 instances, they get house numbers in the last octet after the final period…

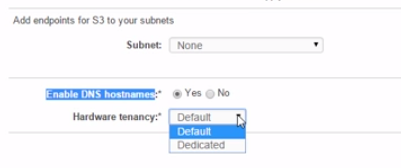


Now NAT instance: this is basically a specialized EC2 instance that's going to live in our public subnet, but it lets instances in our private subnet talk to the wider world, the internet.  

It works by getting itself a public IP address, so not a 10. Address, yeah, that means it can talk directly with the internet, and then we just create a route in our private network.

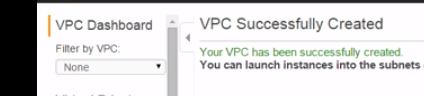
Hey you know what, for any traffic whose IP address we don't know where to send it to, send it over to the NAT instance. The NAT instance then does some networking magic and sent over to the internet. Any replies, well they get routed back to the appropriate host on the private subnet. 

Enable DNS host names: What this is going to do is just make sure that any instances that we spin up in this VPC, okay, are going to get DNS names that are resolvable internally within AWS and externally on the public internet. The latter of course only being if we're putting them in a public subnet and giving them public IPs, but the idea is, right, if we want name resolution rather than just working with IP addresses..

Hardware tenancy option, this lets up change from the default of where we share actual server hardware and the likes with other folks, to the other option, dedicated.

That can cost a ton however...

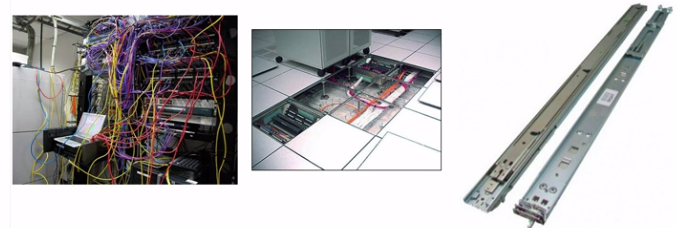
**A World Class Data Center - Building Our First VPC**



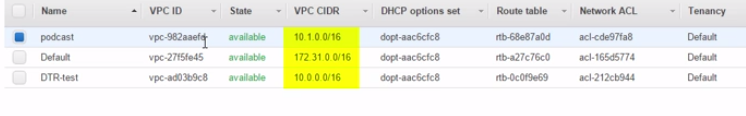
World class air con, world class power and power management, world class cabling, world class internet access, you name it, the whole shebang is backed by Amazon's world class data center infrastructures. 

we hardly had to know anything about data center design.

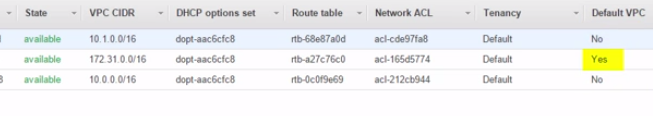
I remember back in the day, right, racking and stacking physical equipment, having contractors in to pull cabling, test AC equipment, pulling up floor tiles, cutting my fingers and hands on like ridiculous server rails, and patching cables behind a bunch of servers and switches that were pumping out hot air into my face like I stood beneath a space shuttle during a take-off.

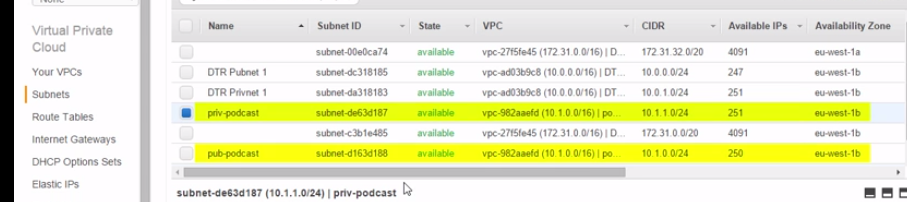
Oh yeah, those were the days, well no more. We just built ourselves a cloud based data center there that rivals, and you know what, no doubt surpasses anything that I have ever been involved with building with my own hands.

**Reviewing**

Quick snapshot of what our VPC consists of 

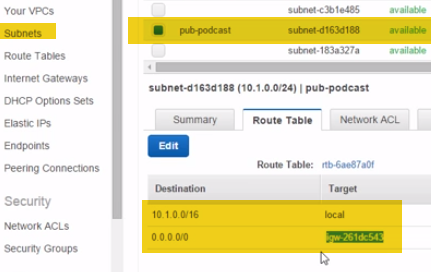
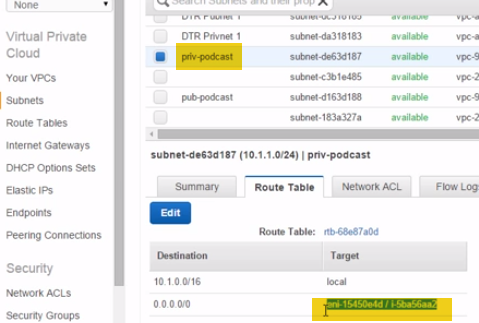
Three VPCs have Ids, all got different network addresses and this'll be really important later on when we try and connect them to each other. We've got DHCP stuff, routing stuff, access control lists, tenancy, the default, right, being we just want to share physical servers on the back end with everyone else, that's what default means. The other option, remember, being for those of us with money to burn who want their own dedicated physical servers on the back end.

default VPC, you can only have one per region, right..This is something that AWS creates automatically for you in each region. so you'll have one default VPC automatically created for you in each region.



subnets: but who the heck knows what those ones that I've not bothered to label are for.

**Moral of the story, remember label your stuff.**

look at the route table down here associated with our public subnet, right, this rule here, 0.0.0.0/0, well this is like the default route for the subnet, so the catchall route that gets used when there's no better matches. And this device here that that traffic gets sent to, starting with igw, is an internet gateway, a router that routes traffic to the internet.

If we compare it to our private subnet up here, yeah, see how that doesn't send traffic to an igw or to an internet gateway? Well, that there, at least at the time or recording, is the major difference between **public and private subnets.**

**A public subnet has got a route to the internet via an internet gateway and a**

**private subnet doesn't.** This other route(other) this one here, oop, well both subnets have got this. And what it does is it lets traffic flow locally between subnets without our VPC.

Okay, cool and all, but it doesn't throw any light onto the mystery of why our public subnet here, why it's got a private network that's not even routable on the internet. Well, let's unravel this mystery. You know what right? We're going to be coming back to all of this in more detail later, but this old boy here, okay, auto-assign public IP, this is where the magic happens or actually this is where we turn the magic on because the magic actually happens on the internet gateway. But this setting, right, it's a subnet wide setting, okay, and if we enable it, so we go up here and then we hit this modify auto-assign public IP option, what this is going to do is anytime we launch an instance into this subnet, it's going to get a public IP. So a proper, bonafide publically routable address, none of this RFC 1918 internal only, not publically routable stuff, no every instance is then going to get its very own public IP address, only the way it works, right, and let's say this first, but the way it works is that each instance still only gets a private IP address from the RFC 1918 CIDR block range down here. But then the internet gateway performs some NATting magic where it maps a public IP address to a private IP address. You get what I'm saying or at least I hope you do, right? Every instance will then a public IP address associated with its private IP address on the internet gateway. So the internet gateway holds a mapping, making it look and feel like the instances have their own public IPs. Anyway, we're going to cover this stuff properly later in the course and I'll draw pictures and diagrams and all of that kind of jazz, okay. For now, right, we know that public subnets 1) have a route to an internet gateway, meaning they can get out to the internet and if we want instances in that subnet to be directly addressable on the internet with public IPs, we can do that too, thanks to a bit of NAT magic done by the VPC's internet gateway. Wowzers.