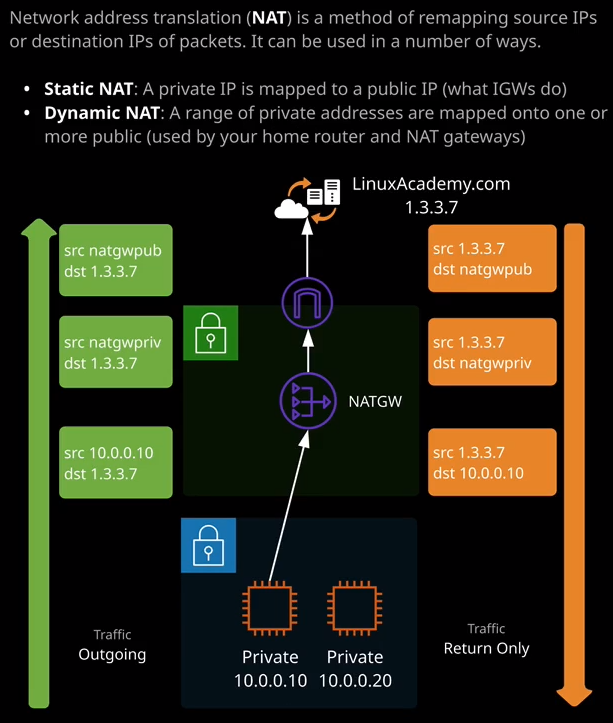
***NAT***



There are going to be situations where all resources inside a VPC will need some level of outgoing internet access and an example of this is running software updates. It's not desirable, though, to give all resources inside a VPC the ability to accept incoming connections. You might not want this for risk reasons or be prevented from doing so by your security team. **It's best practice in everything AWS related to work to the concept of least privilege.** If instances or resources inside a VPC don't need incoming internet access, then don't give them incoming internet access and that's not even involving the fact that we are running out of public IP version four addresses and so **it's always best practice where possible to avoid giving resources publicly routable IP addresses**.

Now in the routing lesson, I told you how an internet gateway works that uses a process called static NAT or static network address translation and network address translation changes the source IP address of traffic, which is leaving an EC2 instance, and it changes it from its private IP address to its allocated public address. In effect, it makes the EC2 instance, a true public machine. With an internet gateway, the same process works in reverse. A public EC2 instance can also be contacted from the public internet but what if you just want outgoing access? **Well, that's where dynamic NAT comes in or dynamic network address translation. It's similar to static NAT but instead of a 1 to 1 relationship where you're changing the private IP address of an instance, which has an associated public IP to that public IP. With dynamic NAT, you're essentially having a large number of private IP addresses that can use a single or a small group of public IPs. Dynamic NAT is actually how your home internet connection likely works. So you've probably got a single public IP address for your internet connection, and no matter how many devices you add to your home network or your home WiFi they all use that single IP address, and that's called dynamic NAT and there's a product that does that inside AWS and it's called the NAT gateway.**

So let's have a look at how this works. Using an example say that we've got a number of private EC2 instances. So instances with private IP addresses. Let's assume that one of these wants to talk to the linuxacademy.com web server, and it's using an IP address of 1.3.3.7. Now, these instances don't have an associated public IP address so they can't use the internet gateway. So our packet is created and it's sent out of the EC2 instance. Got a source address of 10.0.0.10 and a destination address of 1.3.3.7. Now note this is a private IP address, and it cannot be used on the public internet. It's private only IP**. So what occurs is that as soon as this packet hits the NAT gateway, the NAT gateway adjusts the source IP address, and it changes it from the private IP of the EC2 instance to its IP address and I've got that indicated here as NAT gateway priv for private the NAT gateway because it lives in a public subnet it then sends that packet through to the internet gateway and in NAT gateway is given an elastic, static IP address. So when the packet hits the internet gateway, it converts the source IP address from the private IP address of the NAT gateway through to the public elastic IP that we've allocated the NAT gateway and it sends out on its way to the public internet.** **The same process happens in reverse when linuxacademy.com responds the destination IP addresses is this public IP. It hits the internet gateway that gets converted through to the private IP address of the NAT gateway. It hits the NAT gateway and then the destination IP address, which the NAT gateway has kept a record of it's changed from itself to the instance IP address, and then it hits the instance.**

Now how this differs from using an internet gateway is because we got multiple EC2 instances all sharing this IP address linuxacademy.com is unable to initiate a connection to this instance. It doesn't have a solo IP address a public IP that it can connect to and gain access to this instance. So in effect, we're giving these EC2 instances outgoing access only.

Now in AWS, this process is handled by a NAT gateway, and it's probably easier to show you rather than tell you. So I'm going to go across to the VPC console and I'm going to go to NAT gateways because this is where we create NAT gateways. Now AWS has actually got two different products that can fulfill this need: NAT gateways which are the newer product and NAT instances All the instances were, were just EC2 instances with the public IPs allocated in a public subnet which were configured to do dynamic NAT. Now this comes with an admin overhead. You needed to manage these and so **NAT gateways are the as a service response from AWS. So NAT gateways do the same process but you don't need to manage the actual instances that they run on. NAT gateways need to be provisioned in a public subnet and they need to be given a static, elastic IP address because they're public services they'll also need all of the public configuration set up inside the VPC.** So all the work I've done so far configuring this VPC to allow public access that is required for a NAT gateway to work. I'm going to go ahead and create one so I'll create NAT gateway, it needs to be placed inside a public subnet. So I'm going to pick public subnet A so sn-public-a and it needs to be given an elastic IP address. So this will be used and it communicates through the internet gateway with any public IP addresses. Now if you don't already have one defined you need to create a new elastic AP address. So I'll do that and then I'll go ahead and create the NAT gateway. Now, this will take a few moments to finish creating.

While it does, I just want to mention a few points about NAT gateways. Now **they're not highly available by design.** **You have to put them in a single subnet. So it's inside a single availability zone. So while they can tolerate failure of specific components of the product and they do scale really well with load. So while you are able to place additional load on these NAT gateways and they do scale in response to that load, if you do have an entire AZ fail or the subnet fails that the NAT gateway is inside, then the NAT gateway itself will fail.** Now, why this matters is that if you do **need to implement something which is highly available so in my case, I've created this VPC and I've got all these different subnets spread across different availability zones. If I wanted to create a NAT gateway infrastructure that could cope with any AZ in my VPC failing then I need to create a separate NAT gateway in each public subnet that's inside that VPC, so a particular NAT gateway should be used from a best practice perspective by any resources in that AZ. Then you'd place another NAT gateway in a different AZ and you'd have all of the resources in that other AZ using that other NAT gateway.** **So for true high availability, you need as many NAT gateways as you've got availability zones inside the VPC.**