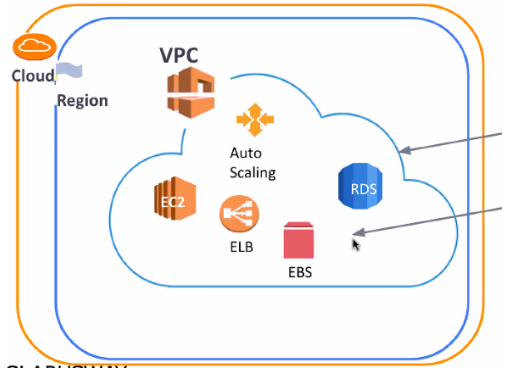
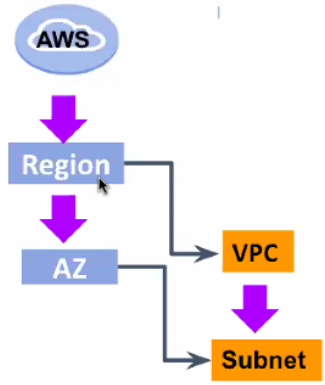
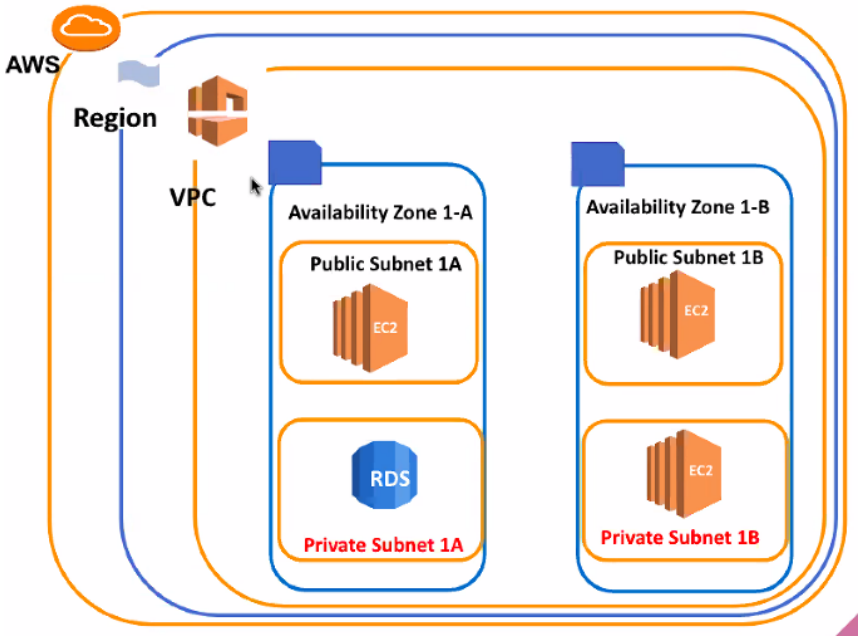
# VPC

Amazon bize default VPC veriyor ancak biz her sistem için ayrı VPC'ler kuracağız. RDS, EC2, S3 gibi servislerin farklı VPC'lerde olması daha güvenli. Her bir region için 5 VPC sınırımız var ancak AWS'den talep ederek bu sınırı arttırabiliriz.

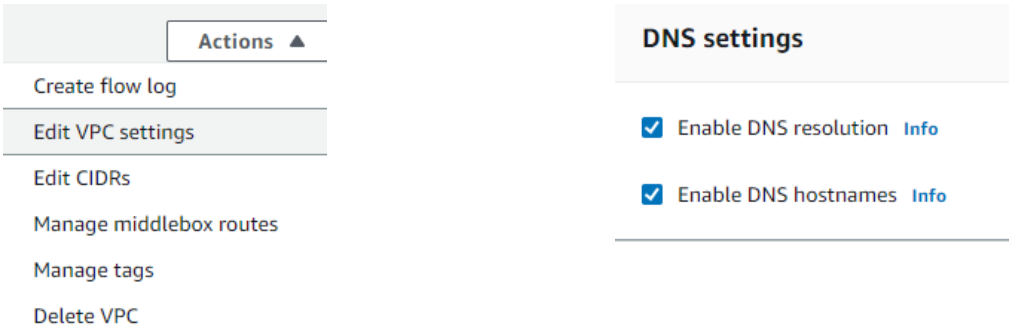


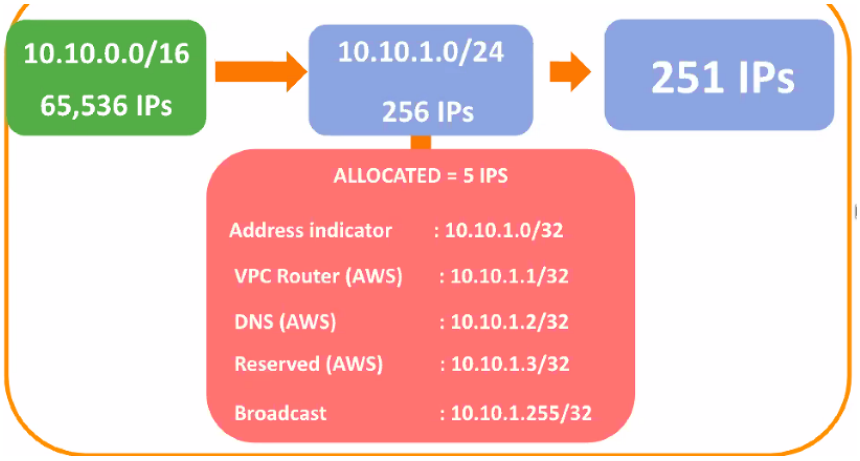

Nasıl ki AWS'de AZ'ler Regionların altında, Subnetler de VPC'nin altında bulunur.



⭐ VPC'yi kurduktan sonra ilk olarak DNS hostname'i enable yapmamız gerek. Yoksa VPC elemanlarının DNS name'i ile çağıramayız. Hep IP yazmamız gerekir. Bundan sonra bana public/private IP'den başka Public/Private DNS Name'de atayabilecek.

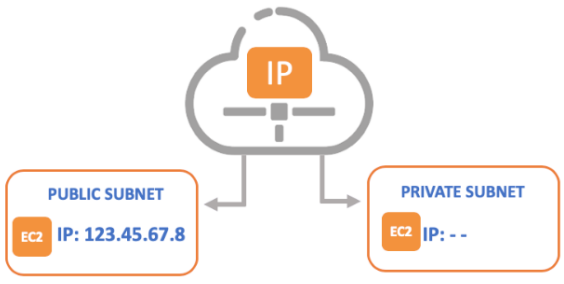



rfc-editor.org/rfc/rfc1918 ⇒ RFC 1918 Dökümanı Network için danışman kaynak

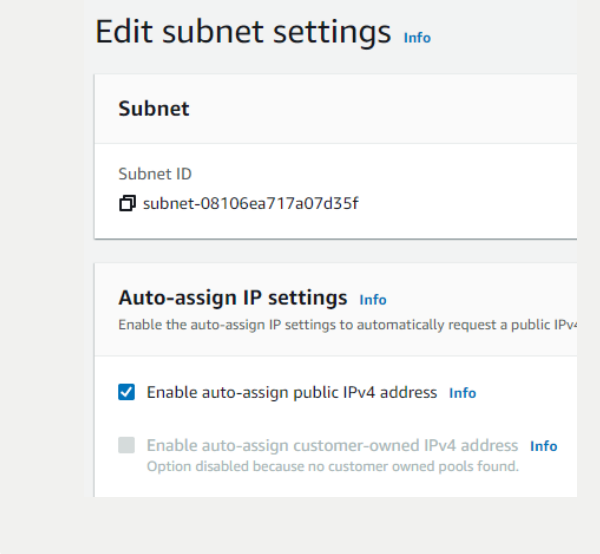



Subnetin içinde 256 parça var bunlardan 5ini kullanamıyoruz. Bu 5'ini AWS kendine tahsis etmiş. Kalanları kullanabiliriz.

⭐ Subnetleri oluşturduktan sonra bir ayar daha yapmamız gerekiyor. Public subnetlerdeki elemanların internete çıkış yapabilmesi için public IP adresine sahip olması gerekiyor. Bu nedenle **Enable Auto-Assign Public IPv4 Address** diyoruz. Bunu tüm public subnetler için yapmamız lazım.

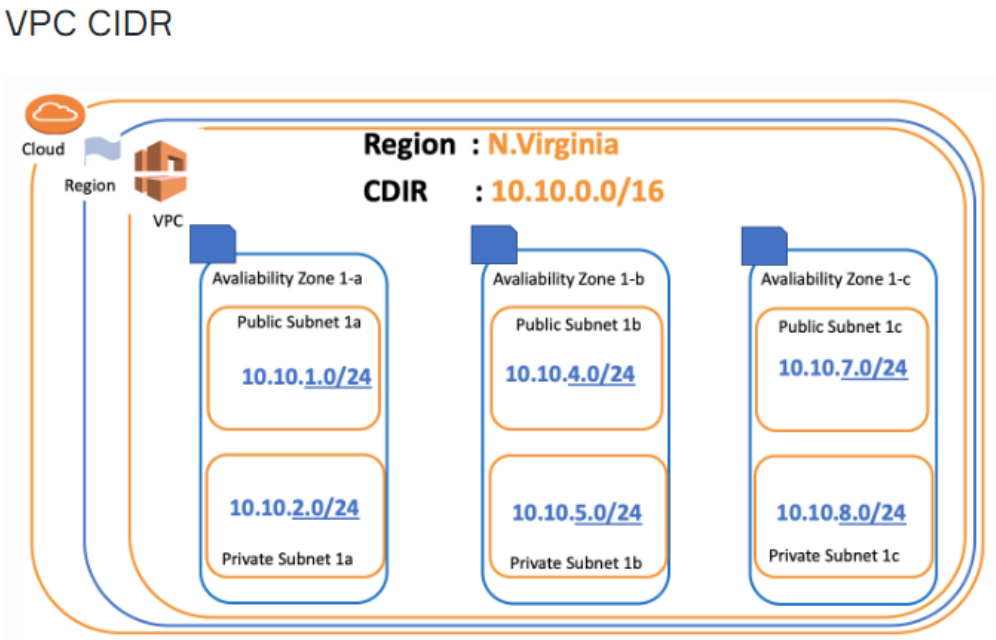



Bu sayede Public'teki EC2'lara Public IP ataması yapılırken Private'dekiler aynı kalır ve internet üzerinden




bağlantı sağlanamaz.

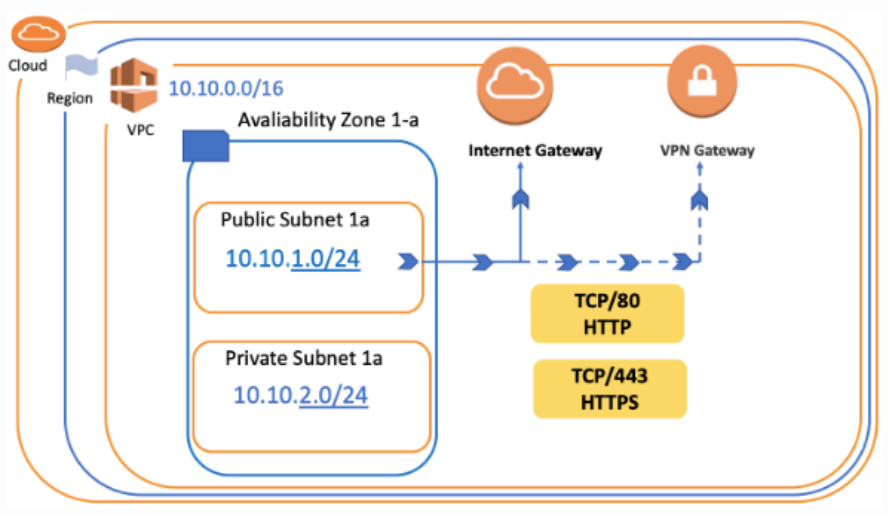
Subnetleri belirlerken aralarda boşluk bırakmak faydalı olabilir. Çünkü AWS subnet bloğunu genişletmeye/değiştirmeye izin vermiyor. İlerdide benzer bir amaç için kullanacağız subnet olursa biri 1'de biri 58'de olmasın diye mimaride boşluk bırakmak faydalı




## Internet Gateway

Internet Gateway bir VPC internete erişimini sağlar. NAT vazifesi de görür. Kendi şirket ağımızdan direkt olarak bu VPC ye bağlanmak istersek VPN Gateway'de kullanabiliriz. Bu daha güvenli. Direkt olarak bir kanal açıyor ve bağlantıyı oradan yapıyoruz.

Yani VPC, dış dünyaya Internet Gateway veya VPN Gateway olmak üzere iki temel yolla bağlanabilir.




İnternet Gateway'i oluşturduk ancak bu hemen internete çıkıyoruz anlamına gelmiyor. Bunu VPC'mize bağlamamız gerek.

## Route Table

VPC'nin giriş çıkış kurallarının yazılı olduğu tablodur. Bir nevi pasaporttur. Biraz sec.gr.'a benzer. VPC içine herhangi bir eleman eklersek bunu route table'a işlememiz gerek. Örneğin aşağıdada görüldüğü gibi dışarıya (internete) çıkmak isteyenler için yolu göstermemiz gerek. Bu nedenle 0.0.0.0 'a gidecekler igw'den gitsin diyoruz.

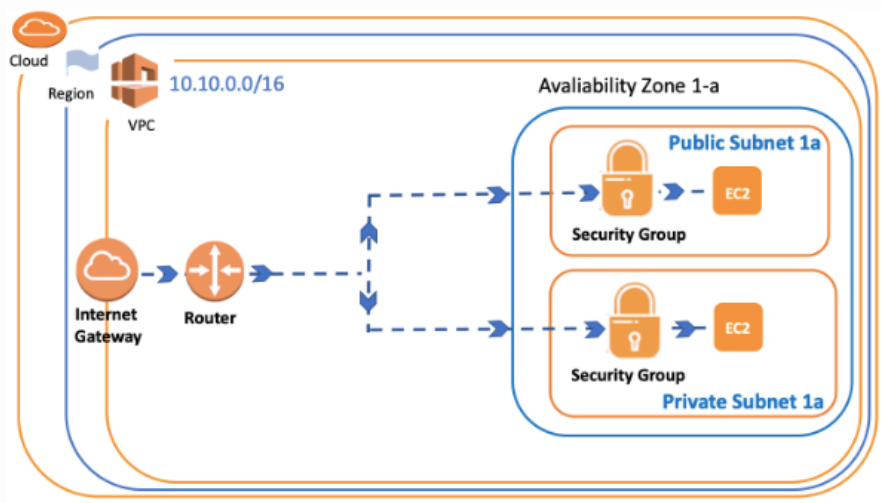
**Routes** (2) — Edit routes

Filter routes — Both — 1

| Destination | Target | Status | Propagated |
|---|---|---|---|
| 0.0.0.0/0 | igw-050ee703a22fc2c39 | Active | No |
| 172.31.0.0/16 | local | Active | No |

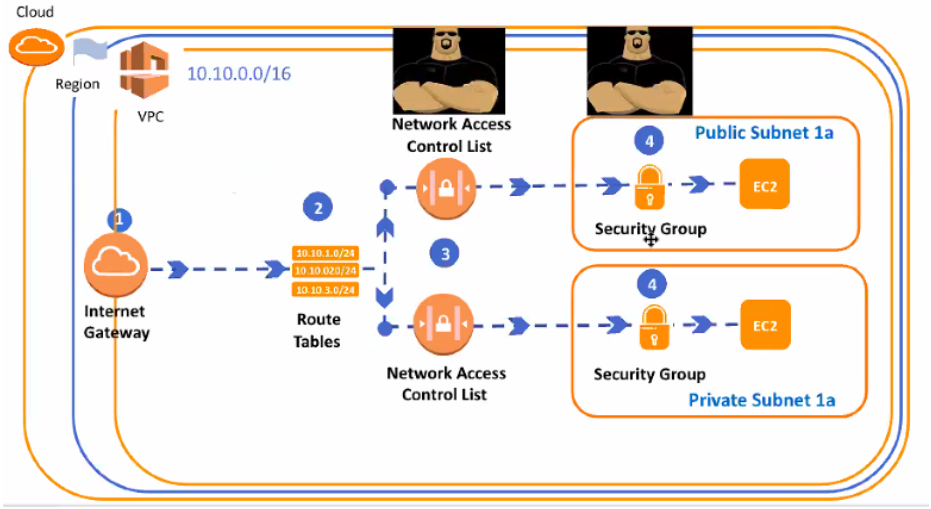
## Security Group

Statefulldur. Giriş çıkış birbiri ile haberleşiyor. Yani girişteki secgrup şartlarını sağlarsan rahatlıkla çıkarsın. EC2ların önlerine konulur. "Hangi portlardan kim erişebilir" izni verilir. EC2'nun Firewall'u diyebiliriz.




## Network Access Control List (NACL)

Sec.Gr. gibi bir firewalldur. Farkı, subnetin önünde yer alır. Sec group EC2 önünde yer alır. Stateless'dır. Yani giriş çıkış kapısı arasında bir haberleşme yok. giriş için bir kontrol, çıkış için ayrı bir kontrol. Her bir subnet bir NACL'a bağlanmalıdır. Eğer biz seçmezsek default NACL'a bağlanır. Sadece tek bir NACL'a bağlanabilir.




⭐ VPC oluşturduğumuz zaman otomatik olarak Default bir NACL oluşur. Oluşan bu Default NACL **All Traffic Allow**'dur.
Ancak sonradan oluşturulan (default olmayan) NACL'lar en başta **All Traffic Deny**'dır. Oluşturduktan sonra inbound ve outbound rule belirlememiz gerek

**Inbound rules (1)**

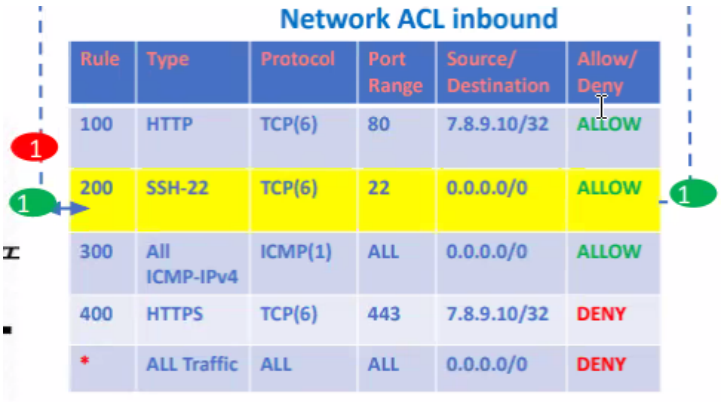
Edit inbound rules

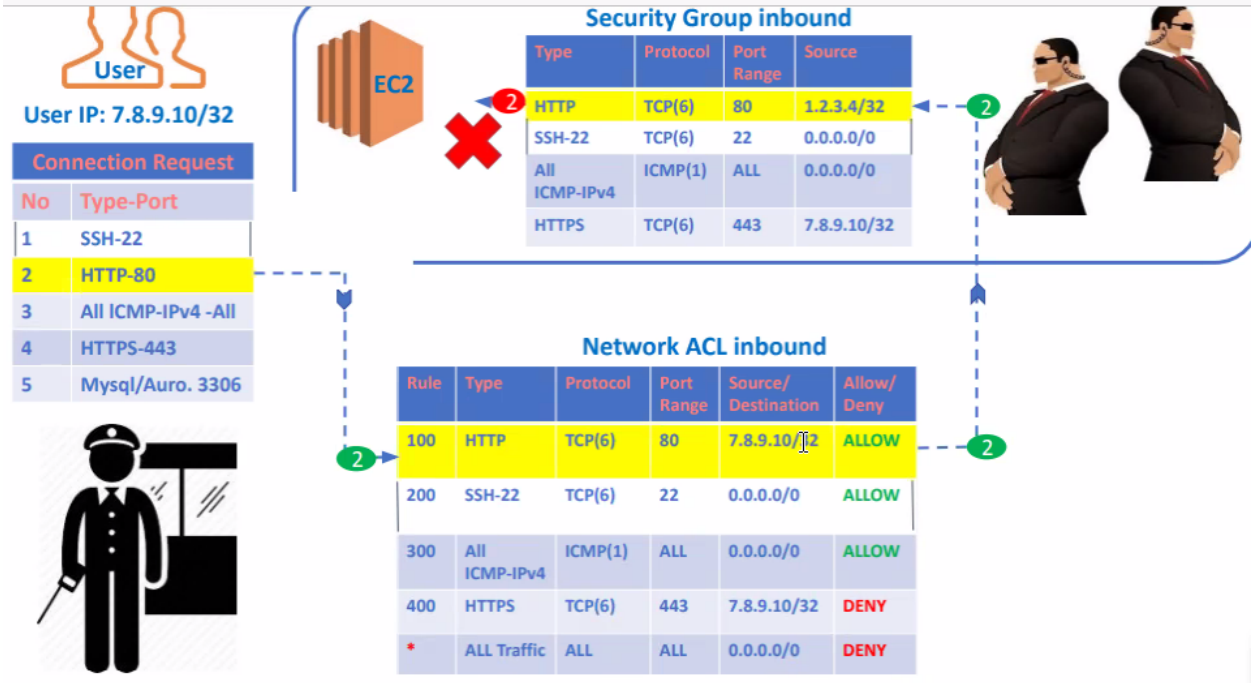
Filter inbound rules

1

| Rule number | Type | Protocol | Port range | Source | Allow/Deny |
|---|---|---|---|---|---|
| * | All traffic | All | All | 0.0.0.0/0 | Deny |

NACL'larda sıralı kural oluşturuluyor ve bunları allow ya da deny olarak ayarlıyoruz. Yukarıdan aşağıya kuralları sırayla tarar



(Stateful) **Security Group inbound**

| Type | Protocol | Port Range | Source |
|---|---|---|---|
| HTTP | TCP(6) | 80 | 1.2.3.4/32 |
| SSH-22 | TCP(6) | 22 | 0.0.0.0/0 |
| All ICMP-IPv4 | ICMP(1) | ALL | 0.0.0.0/0 |
| HTTPS | TCP(6) | 443 | 7.8.9.10/32 |

ALLOW Only

**Network ACL inbound** (Stateless)

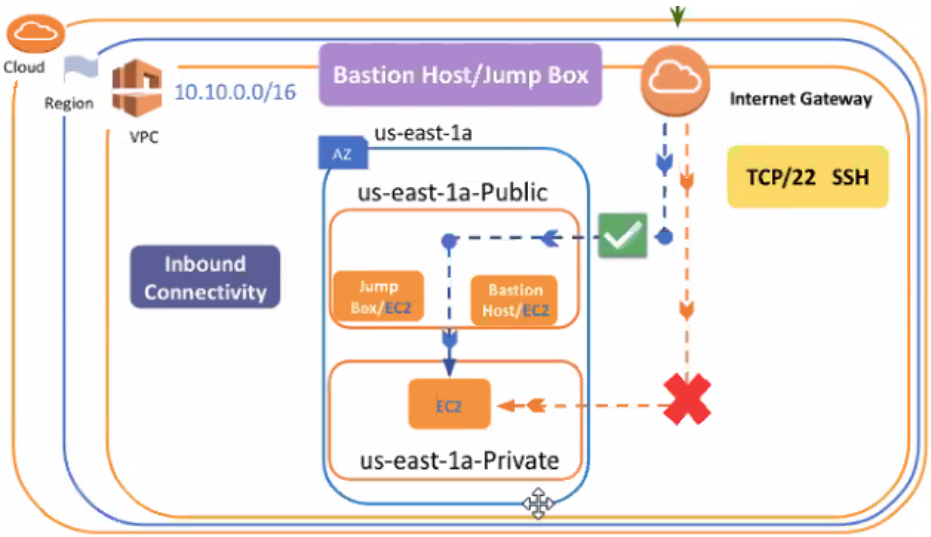
| Rule | Type | Protocol | Port Range | Source | Allow/ Deny |
|---|---|---|---|---|---|
| 100 | HTTP | TCP(6) | 80 | 7.8.9.10/32 | ALLOW |
| 200 | SSH-22 | TCP(6) | 22 | 0.0.0.0/0 | ALLOW |
| 300 | All ICMP-IPv4 | ICMP(1) | ALL | 0.0.0.0/0 | ALLOW |
| 400 | HTTPS | TCP(6) | 443 | 7.8.9.10/32 | DENY |
| * | ALL Traffic | ALL | ALL | 0.0.0.0/0 | DENY |

(Stateless) **Network ACL outbound**

| Rule | Type | Protocol | Port Range | Destination | Allow/ Deny |
|---|---|---|---|---|---|
| 100 | HTTP | TCP(6) | 80 | 7.8.9.10/32 | ALLOW |
| 200 | Custom TCP | TCP(6) | 32768 - 65535 | 0.0.0.0/0 | ALLOW |
| 300 | All ICMP-IPv4 | ICMP(1) | ALL | 0.0.0.0/0 | ALLOW |
| 400 | HTTPS | TCP(6) | 443 | 7.8.9.10/32 | DENY |
| * | ALL Traffic | ALL | ALL | 0.0.0.0/0 | DENY |

## Bastion Host/Jump Box

Bunlar birer instance. Bu yüzden free-Tier kapsamında. Oluşturduğumuz private subnetteki bir instance'ın Public IP'si olmadığı için ona ulaşmak için Bastion Host kullanıyoruz. Private ile aynı VPC'de olmalı. Mutlaka public subnette olmalı. Inbound connectivity için. Genel bir Cloud kavramıdır. On-Premis, Azure ya da Goodle cloudda da kullanılır.




Private'a bağlanmak için Bastion instance'a geldik. Ancak bir sorunumuz var. Yeni oluşan bu instance'da .pem dosyamız yok. Dolasıyla bu şekilde private'ye bağlanamayız. 2 yöntem var.

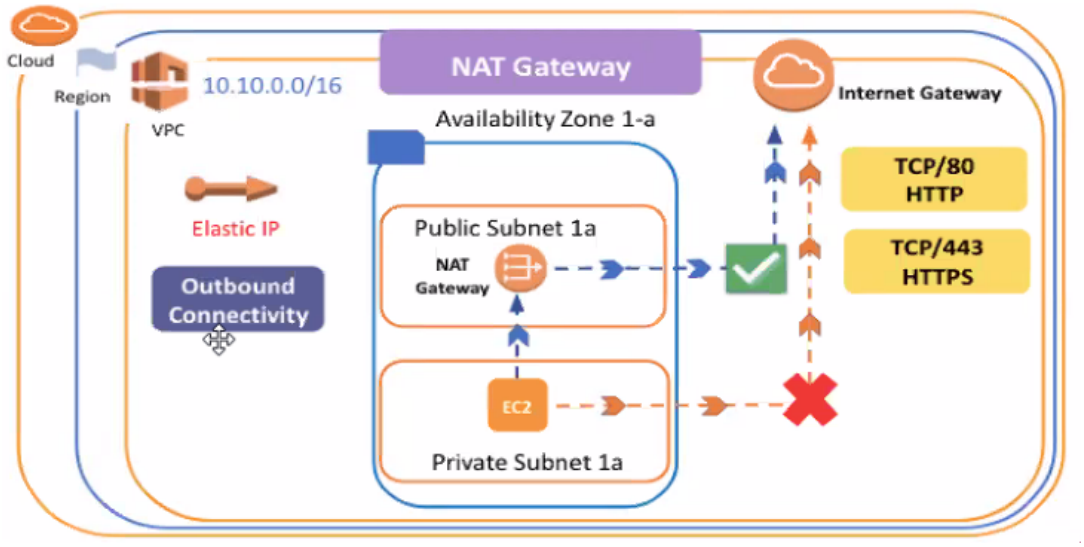
1. Zahmetli yol. localimizdeki pem dosyasını açıp içindeki key'i kopyalar, Bastionda aynı isimde pem dosyası oluşturup chmod 400 yapar ardından ssh ile Private'a bağlanırız. ssh ile Bastiondan Privateye giderken mecburen private IP adresini kullanırız. Çünkü public yok.
   *Burada şöyle bir soru gelebilir aklımıza. Ben kendi evimden direkt private IP ile bağlanamıyorum elin Bastion Hostunun ne ayrıcalığı var da ssh'tan Private IP ile bağlanıyor. Çünkü Bastion host ile Private Instance aynı VPC'de. Aynı LAN'daki cihazlar birbiri ile private üzerinden haberleşir.*
2. Agent kullanma. Bastion'a bağlanırken yanımıza agent alıyoruz. pem dosyasını o taşıyor.

```
1. eval "$(ssh-agent)" ya da eval $(ssh-agent -s)=> Burada ajanı çağırıyoruz.
2. ssh-add ./[your pem file name] => pem dosyasının pathini yazıyoruz.
ya da ssh-add -K key.pem
3. ssh -A ec2-user@ec2-3-88-199-43.compute-1.amazonaws.com => komutu ile Bastiona bağlanırken ajan yanında pemi de götürüyor.
Bastiona bağlandıktan sonra
4. ssh ec2-user@[Your private EC2 private IP] => ile private ye bağlanıyoruz.
```

Lokale döndükten belli bir süre sonra bile ajan çalışmaya devam ediyor. Yani Bastiona ssh ec2-user@[Bastion Public IP] ile bağlanabiliriz.

## NAT (Network Address Translation) Gateway

Private Subnetten dışarıya çıkmak için kullanılır. Outbound connectivity sağlar. Yani subnetten çıkan requestler engellidir. Bunun sayesinde çıkış yapılır. Ancak subnete giriş için kullanılmaz. AWS Fully-managed ediyor. Bakım tutum onda. Sen bırak gerisini ben hallederim modunda. Cüz'i bir ücrete mukabbil kullanabiliyoruz. Kullanmak için Static IP gerekiyor. Çünkü benim private Ip'yi alır Elastic IP'ye çevirir ve o elastic IP ile internete bağlanır. Bu yüzden önceden Static IP almamız gerek.




## Elastic IP

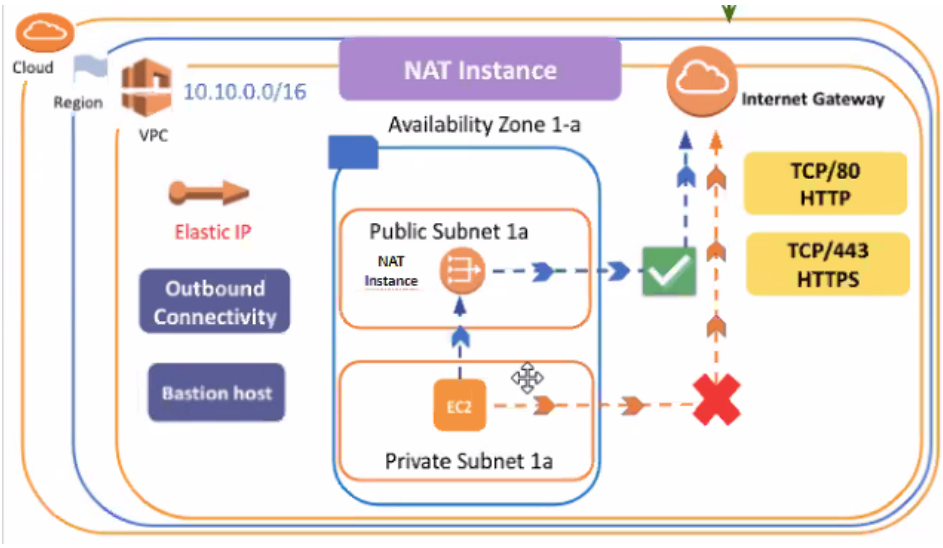
Standart Static IP'dir. Sizin instance'a tanımlı kalıcı IP'dir. Statik IP'yi almak, kullanmak ücretsiz. Ancak aldık ve kullanmıyorsak para öderiz. Bunun için release işlemi yapmamız gerek. Mantıken boşu boşuna IP adresi işgal edersen parayı kesiyor. Kullanmadığın saat başı 0,01 USD

Public IP'ler dinamiktir. Yani Instance'ı durdurur/başlatırsanız yeni bir IP atanır. Ancak Static IP kalıcıdır, değişmez.

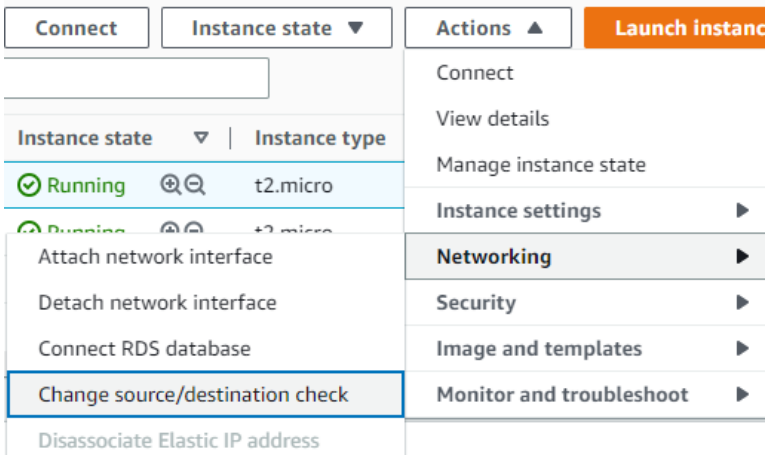
NAT Gateway oluşturmak için Elastic IP'ye sahip olmak zorunludur.

## NAT Instance

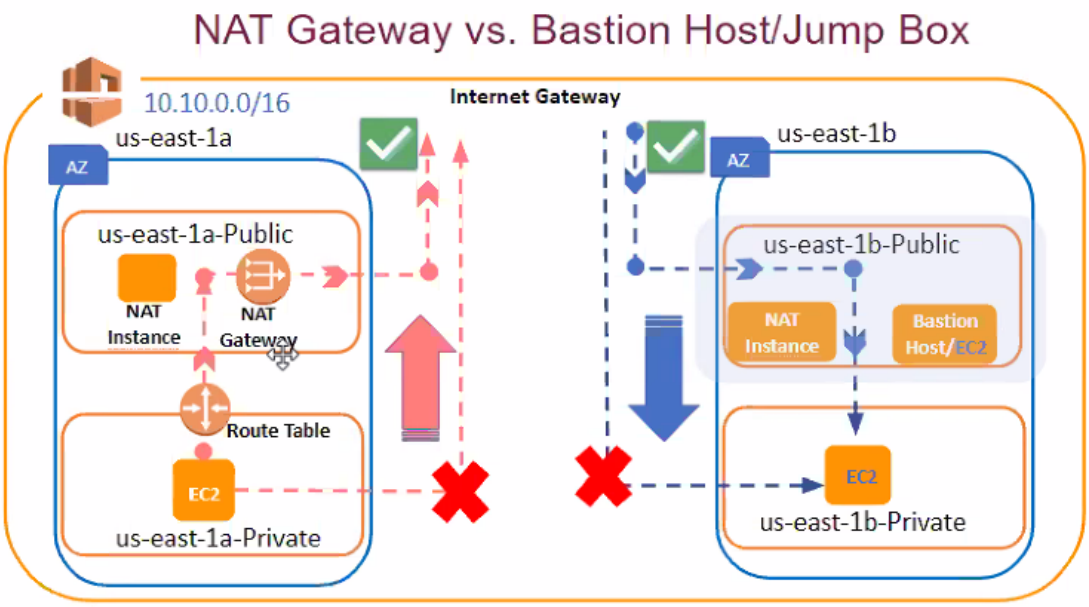
Natting (Public - Private IP değiştirme şeysi) işlemini yapan özel bir instance. Yani Static IP'ye ihtiyacımız yok. Ona göre AMI'si var. Instance'ı ayağa kaldırırken o AMI'yi seçmeliyiz. (Community AMI kısımından arama çubuğuna NAT yazıyoruz. İlk  çıkan AMI'yi seçiyoruz) Natting işlemini yapabilen bir instance olduğu için SSH ile bağlanabilirim. Doğal olarak tersten gelerek Private subnete giriş yapabiliriz. Tabi private'ın ssh'ı açık ise. Ancak NAT Gatewayde bu yok sadece çıkış yapabiliriz.




⭐ Burada önemli bir adım daha var. O da source/destination check'i tiklemek gerekiyor. Aşağıdaki gibi networkün altında. Bunun anlamı şu: Normalde bizim kullandığımız standart instance'lar sadece kendisi ile ilgili bir paket varsa işlem yapıyor. Yani pakete source/destination checki yapıyor kendisine ait değilse işlem yapmaz. Ancak bizim burada kullanacağımız NAT İnstance bir nevi köprü görecek. Bu yüzden paket kendisi ile alakalı olmasa bile iletilmesi gerek. Bu nedenle alıcı/gönderici kontrolünün kapalı olması gerekiyor.




⭐ NAT instance'ı kurup gerekli bağlamaları yaptıktan sonra Private instance internete erişebilir. Tabi bunun için de NAT instance'ın security grubundan belirli izinlerin verilmesi gerek. Mesela ping atmak istiyor ICMP protocolüne izin verilmeli. yum update yapacaksak bunların çalıştığı HTTPS portunun açılması gerek. Ya da curl ile bir siteye gideceksek HTTP'nin açılması gerek.



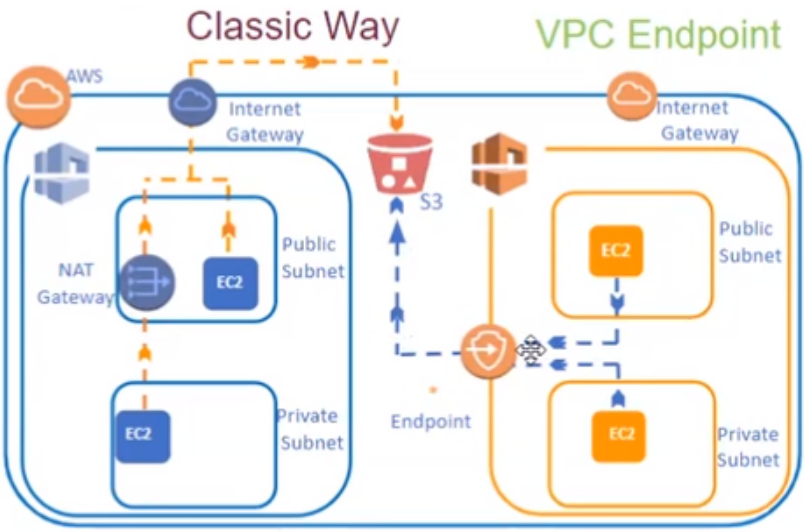

Bastion host'ta sadece dışarıdan içeriye giriş vardır.

İnternet gateway de çift yönlü trafik var. Yani içeriden çıkan bir bağlantı tekrar girer. Bu 1. Ayrıca dışarıdan bir bağlantı da içeri girip çıkabilir. Bu da 2

Ancak NAT Gateway tek yönlüdür. Sadece içeriden çıkmış bir bağlantı geri gelebilir. Dışarıdan ilk defa gelen bir bağlantı içeri giremez.

## VPC EndPoint

AWS'nin non-VPC servisleri var. Örneğin S3, DynamoDB. Bunlar herhangi bir VPC içinde bulunmaz AWS'nin kendi sınırları içinde bulunur. Aynı VPC'de bulunmadığımız için bu servislere bağlanmak için internete çıkıp oradan S3, DynamoDB'ye bağlanmamız gerekiyor. Endpoint ise internete çıkmadan bunara bağlanabilmeyi sağlar. Route table'a işliyoruz.

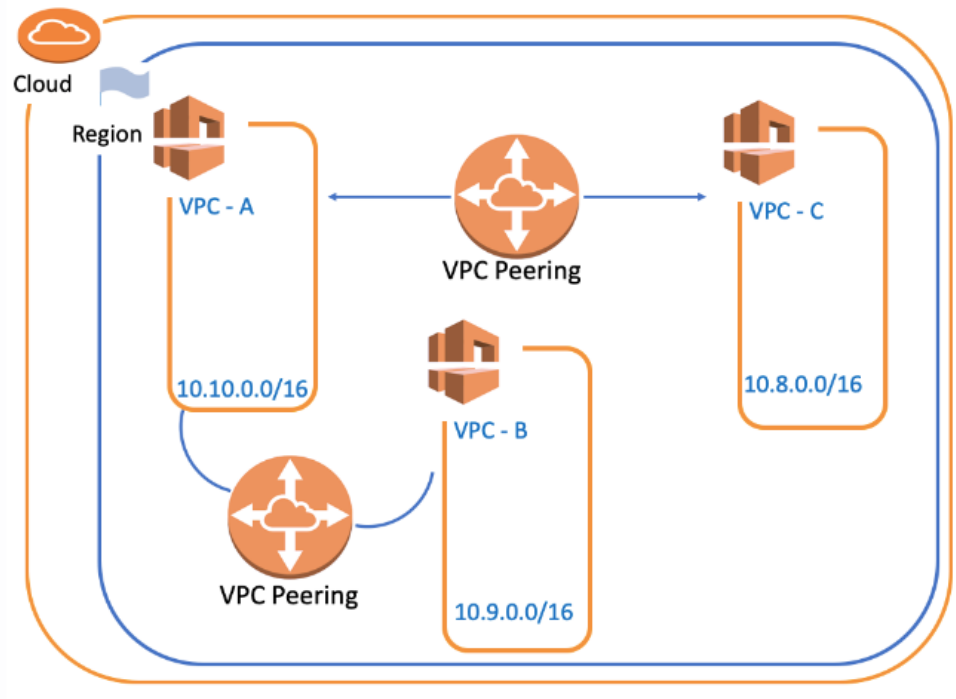



## VPC Peering (Peering Connection)

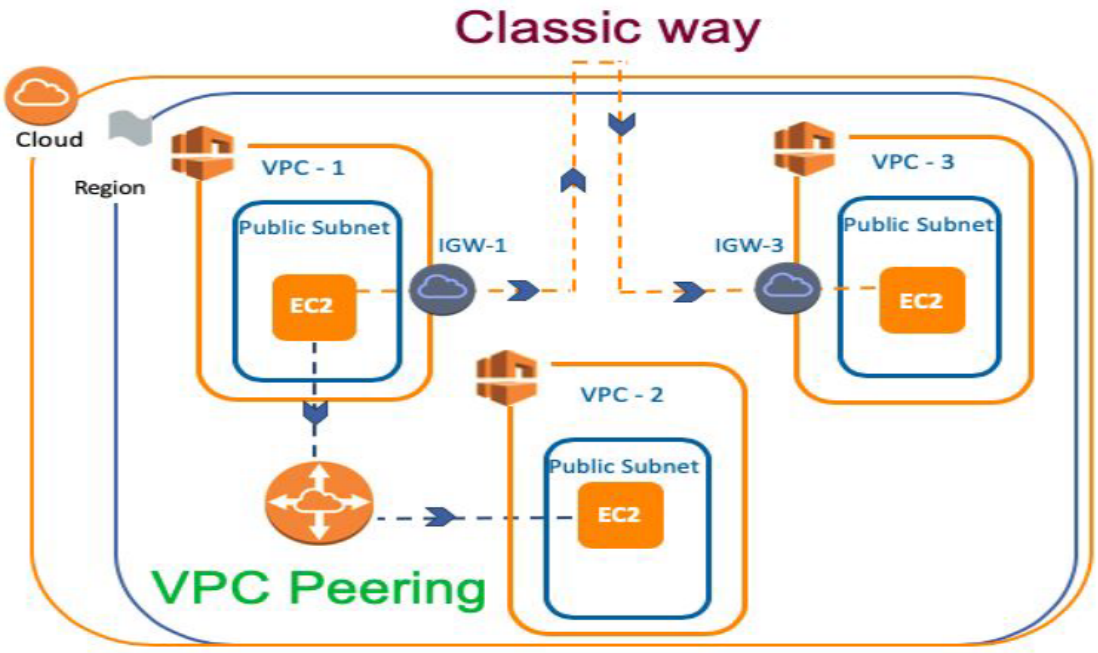
VPC Peering, iki VPC arasındaki bir ağ bağlantısıdır. Clasic yöntemde bir VPC'den diğerine bağlanılırken internete çıkılır oradan diğerine geçilir. Ancak peering işlemi sayesinde iki VPC sanki aynı VPC içindeymiş gibi davranır.

EndPointten farkı: EndPointte bir VPC servisi ile bir non-VPC servisi arasında bağlantı olur. Ancak Peering işleminde iki VPC arası bağlantı sağlanır. Bu sayede farklı VPC'lerdeki iki servis arasında sanki aynı VPC de kurulmuş gibi private IP ile bağlantı sağlanır.

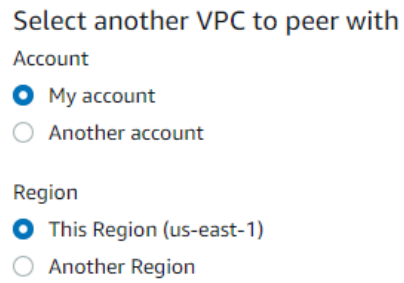
**Önemli husus,** bağlanılan iki VPC'nin CIDR bloklarının farklı olması lazım. Çünkü aynı VPC'deymiş gibi hareket edecekleri için gelen paketin karışma ihtimali var.




Yukarıda A ile B ve A ile C peering yaptı diye otomatik olarak B ile C aynı VPC'deymiş gibi **davranmaz.**

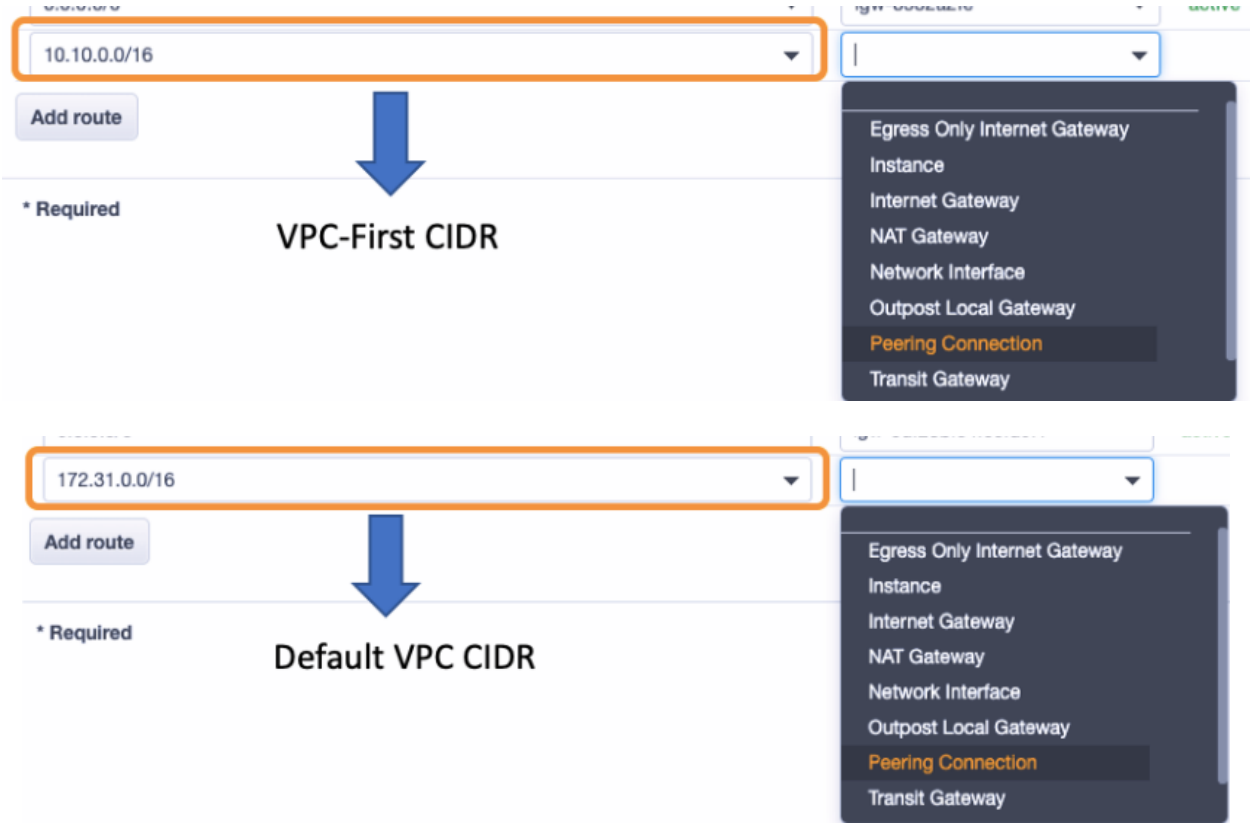



VPC Peering menüsündeki Başka hesap ve başka region seçeneklerinden farklı hesaplar arası ya da farklı regionlar arası VPC'leri birbiri ile bağlayabileceğimizi anlıyoruz.




Peering işleminde karşı taraftan onay gelmesi gerekiyor. Kendi hesabımızda yaparsak da biz onaylamalıyız. Bunu başka bir hesap ile yaparsak karşı taraftan onay gelmesini bekliyoruz. Action menüsü altında **Accept Request.**

Ardından route table'ı düzenlemeliyiz. Her iki VPC'nin CIDR ını tabloya yazıyoruz, gidiş yöntemini VPC peering olarak seçiyoruz.



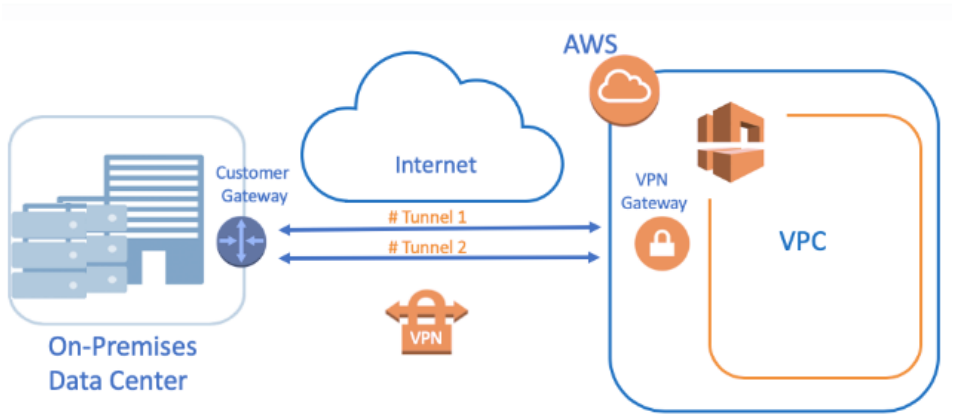

## VPN & Direct Connect

AWS dışında bulunan yerel ağınızdan Clouddaki VPC'nize güvenli bir şekilde bağlanmak için özel bir tünel oluşturmanıza yardımcı olur.

**Site-to-Site VPN**

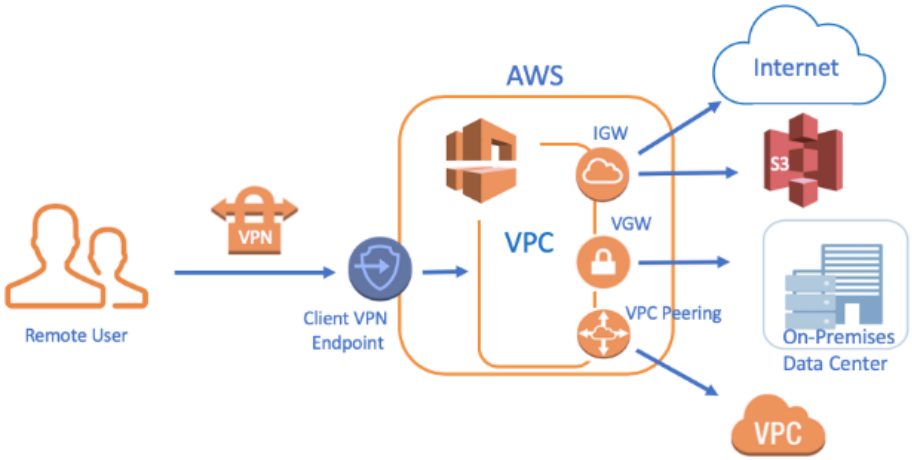
AWS'deki hesap ile on-premise ağın güvenli bir şekilde bağlanmasını sağlar. High-availability için 2 tünel açar. Güvenlik için (IPSec) Internet protocol Security kullanır. Customer Gateway diye bir program var. Bunu on-premise kuruyoruz. VPN gateway ile bağlantı sağlıyor.

Hem on-premise hem de AWS'de serverlarımız varsa ortak kullanım için faydalı.




**Client VPN**

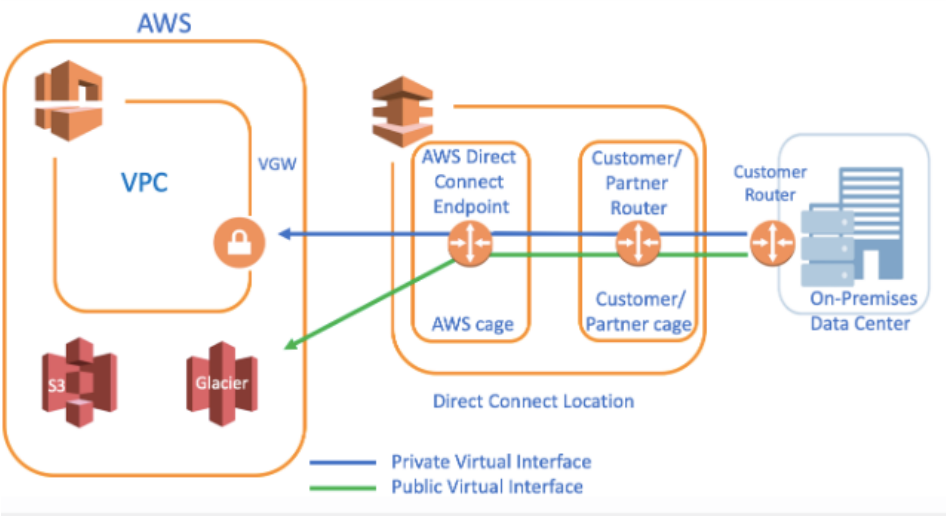
Uzaktan on-premise ağımıza bağlanmayı sağlar. Bunun için önce AWS bizim ağa bağlanır. Daha sonra kendi üzerinden bize bir kanal açıyor. Data Center'ımıza ulaşmak için.




**Direct Connect**

Burada belirli aracı firmalar var. Dünya genelinde 100 tane. AWS ile bunlar arasında fiber obtik kablolar döşenmiş. Biz de kendi data center'ımızı bunlara yakın bir yere kurup fiber optik bağlantı yapıyoruz. Bu sayede hem hızlı hem de internete çıkmadan güvenli bir bağlantı elde ediyoruz. Oldukça maliyetli bir işlem.

Galiba Pentagonun sistemi bu şekilde.



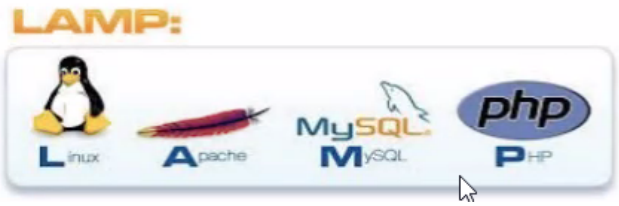

**VPC4**

Dynamic Web Site nedir?

Staticten farkı kullanıcı da bir girdi yapabiliyor bu sitede.

Dynamic website için genellikle wordpress kullanılır


Bir siteyi Dynamic olarak ayağa kaldırmak için bir LAMP stack'e ihtiyaç vardır. Bu da 4 şeyden oluşur. Yani bir işletim sistemi, Web Server, Database ve yazılım dili gerekli. Wordpress de php kullanır.




Kullanıcının yaptığı girdiler database'de saklanır.

VPC-4 Hands-on

**Ephemeral Port**

Bir porttan giren bir işlem yine aynı porttan çıkacak diye bir şey yok. Örneğin ssh girişte 22'den girer ancak çıkış için 22'yi kullanmaz. Kullanılan sisteme göre belirli aralıkta bir portu açık bırakıyoruz. Biz linuxtan girdiğimiz için NACL outbound rule kısmında Custom TCP seçip port numarasına 32768-61000 yazıyoruz.

- Many Linux kernels (including the Amazon Linux kernel) use ports 32768-61000.
- Requests originating from Elastic Load Balancing use ports 1024-65535.
- Windows operating systems through Windows Server 2003 use ports 1025-5000.
- Windows Server 2008 and later versions use ports 49152-65535.
- A NAT gateway uses ports 1024-65535.
- AWS Lambda functions use ports 1024-65535.

Detaylar ⇒ docs.aws.amazon.com/vpc/latest/userguide/vpc-network-acls.html