VPC- Virtual Private Cloud

* Amazon Virtual Private Cloud (Amazon VPC) - A *virtual private cloud* (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. It lets you launch AWS resources such as EC2 in your virtual network. This virtual network closely resembles a traditional network that you'd operate in your own data center; with the benefits of using the scalable infrastructure of AWS.AWS VPC is the networking layer of EC2.
* Amazon VPC supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see [PCI DSS Level 1](https://aws.amazon.com/compliance/pci-dss-level-1-faqs/).
* Amazon VPC integrates with many other AWS services; furthermore, some services require a VPC in your account to carry out certain functions. Below are examples of services that use Amazon VPC.

|  |  |
| --- | --- |
| **Service** | **Relevant Topic** |
| Amazon EC2 | [Amazon EC2 and Amazon VPC](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-vpc.html) |
| Auto Scaling | [Auto Scaling and Amazon VPC](http://docs.aws.amazon.com/autoscaling/latest/userguide/autoscalingsubnets.html) |
| Elastic Load Balancing | [Setting Up Elastic Load Balancing](http://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/setting-up-elb.html) |
| Amazon RDS | [Amazon RDS and Amazon VPC](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.RDSVPC.html) |
| Route 53 | [Working with Private Hosted Zones](http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/hosted-zones-private.html) |

* Concepts of VPC:
* Subnets
* IP addressing
* Security Groups and NACLs
* VPC Flow Logs
* Route Tables
* Internet Gateways & Egress Only Gateways
* NAT- gateways and instances
* DHCP options
* DNS
* VPC peering
* VPC endpoints (AWS PrivateLink)
* VPN connections

**Subnets**

* VPC and Subnets – Subnet is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet and a subnet must completely reside in one AZ. Each subnet must be associated with one and only one (at a time) route table. Multiple subnets can be associated with a route table. If you do not associate your subnet with any route table, the subnet is implicitly associated with the main route table (which cannot be deleted but can be modified).
* Public Subnet - If a subnet's traffic is routed to an internet gateway, the subnet is known as a public subnetIf you want your instance in a public subnet to communicate with the internet over IPv4, it must have a public IPv4 address or an Elastic IP address (IPv4)
* Private Subnet – If a subnet’s traffic is not directed towards IG it is called private subnet.
* VPN-only subnet - If a subnet doesn't have a route to the internet gateway, but has its traffic routed to a virtual private gateway for a VPN connection, the subnet is known as a VPN-only subnet.
* Note: Regardless of the type of subnet, the internal IPv4 address range of the subnet is always private—we do not announce the address block to the internet.
* Understanding the Subnet Creation –
* To create only a single subnet in the VPC we can specify the CIDR block of VPC exactly same as CIDR block of the VPC
* Other wise to create multiple subnets specify non overlapping subnet CIDR blocks. The allowed block size is between /28 netmask and /16 netmask.
* First four and last IP is a subnet is reserved. For eg: in a subnet of CIDR block 10.0.0.0/24

1. 10.0.0.0 : Network address
2. 10.0.0.1 : Reserved by AWS for the VPC router
3. 10.0.0.2: The second address of the VPC (primary CIDR block’s 2nd IP) is reserved for Amazon DNS server. In each subnet the 2nd Ip is reserved for local DNS
4. 10.0.0.3 – reserved by AWS for future use
5. 10.0.0.255 – Network broadcast address. Broadcast in a VPC is not supported so this address is reserved.

* Secondary CIDR blocks can also be associated to a VPC. **See : IPv4 CIDR Block Association Restrictions**
* While creating a VPC you can also specify a CIDR block (allowed size - a/16 or smaller like a/12 or a/8) **for PRIVATE IPS.** This private IP block may be inside or outside the public IP block.
* When you add or remove a CIDR block, it can go through various states: associating | associated |disassociating | disassociated | failing | failed. The CIDR block is ready for you to use when it's in the associated stat
* Subnet Routing –
* Each Subnet must be associated with a route table, which specifies the allowed routes for outbound.
* Every subnet on creation is associated with the main route table for the VPC. If the primary CIDR block chosen for the VPC was 10.0.0.0/16 , the main table by default has one entry and looks like this:

|  |  |
| --- | --- |
| Destination | Target |
| 10.0.0.0/16 | local |

* You can allow an instance in your VPC to initiate outbound connections to the internet over IPv4 but prevent unsolicited inbound connections from the internet using a network address translation (NAT) gateway or instance.
* To make a number of instances available to the internet at static IP ,it is recommended not to use individual Elastic IPs for each (since no of Elastic IPs you can use are limited). Instead use a NAT and attach elastic IP to it, keep the instances at private IPs.
* Elastic IPs and public IPs are accessed through IGs. In a VPN connection a virtual private gateway is used which uses private IPs
* Subnet Security is provided by two features: Security Groups and Access Control Lists-Security groups control inbound and outbound traffic for your instances, and network ACLs control inbound and outbound traffic for your subnets.
* Connections with other LANs and VPCs – you can connect with hosts in other VPCs or LANs using an IPsec AWS managed VPN connection, making the AWS Cloud an extension of your data center, as long as their IP is outside the CIDR block(s) of your VPC(not just outside your subnet).
* However to connect to other VPCs we can also set up a VPC peering connection provided that the CIDR blocks of the networks should be different. For now VPC peering connection enables you to route traffic using private IPs.
* What can you do in a VPC (related to subnets): GUIDE - <https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/working-with-vpcs.html>
* Create a VPC
* Creating a subnet in your VPC
* Associating a secondary CIDR block with your VPC
* Launching instances in your subnet
* Disassociating CIDR block from VPC
* Deleting your subnet
* Deleting your VPC
* NAT - NAT device enables instances in a private subnet to connect to the Internet (for example, for software updates) or other AWS services, but prevent the Internet from initiating connections with the instances

**IP Addressing**

* Private IP
* Public IP
* Elastic IP
* If VPC is enabled to support DNS hostnames, then each instance with a public/elastic IP receives a DNS hostname. DNS is resolved to private or public IP depending on whether the DNS hostname has been hit from outside or inside the network.
* Elastic Network Interface – a logical NIC

An ENI can have following components:

* Primary private IP
* One or more secondary IP
* Elastic IP/Public IP
* Security Groups
* MAC address
* Source/Destination Check Flag
* Description
* There is a limit on how many network interfaces can be assigned and how many IPv4 addresses can be assigned per network interface, on each instance type. For e.g. T2 micro can have 2 ENI and 2 IPv4 per ENI.
* Scenarios for using Network Interfaces
* Create a Management Network – e.g. : attaching a web server (instance) to a public and a private subnet by attaching two interfaces to it. The public subnet (may be behind a load balancer) has inbound SG rules for 80 and 443 from all and the private subnet has rules allowing SSH only from some IPs (in this case the instance IP).In case of an instance failure….
* Use network and security appliances – The same network appliance eg NAT, firewall etc can be used in multiple networks and can interact with each network through a separate ENI.
* Create dual-homed instance with workloads/roles on distinct subnet
* Create a low-budget , high availability solution

**VPC Security**

* VPC security:
* IAM
* Security Groups – acts as Instance level firewalls, can specify allowed inbound and outbound traffic.
* NACLs - A *network access control list (ACL)* is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets. You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC.A subnet must be associated and can be associated to only one ACL at a time. Multiple Subnets can be associated with the same ACL The default ACL contains rule that allows all (not IPv6) inbound and outbound traffic. When we proceed to create a custom ACL, if we do not specify any rules in it all traffic (I/O) is blocked. Each rule has a number (rank) and is evaluated bottom to top. So it’s a good practice to add rules in multiples of 10/100.Each rules can allow or deny traffic. Unlike SG, ACL is stateless so to allow responses to incoming traffic, an outbound rule should be specified in ACL.

**NOTE: An ELB won’t be able to carry out health checks of instances in a private subnet if the subnet’s ACL does not allow any traffic from 0.0.0.0/0. For ACL for load balancer**

[**http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-security-groups.html#elb-vpc-nacl**](http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-security-groups.html%23elb-vpc-nacl)**.**

**NOTE: If unsure (although it’s secure to decide the ephemeral port range precisely) you can choose to open all ephemeral port range (1024 - 65535) for inbound traffic in NACL because it will ultimately be filtered by SGs.**

* Flow Logs - You can monitor the accepted and rejected IP traffic going to and from your instances by creating a flow log for a VPC, subnet, or individual network interface. Flow log data is published to CloudWatch Logs, and can help you diagnose overly restrictive or overly permissive security group and network ACL rules. You can create a flow log for a VPC, a subnet, or a network interface. If you create a flow log for a subnet or VPC, each network interface in the VPC or subnet is monitored. After you've created a flow log, it can take several minutes to begin collecting data and publishing to CloudWatch Logs. Flow logs do not capture real-time log streams for your network interfaces. If you launch more instances into your subnet after you've created a flow log for your subnet or VPC, then a new log stream is created for each new network interface as soon as any network traffic is recorded for that network interface. If you no longer require a flow log, you can delete it. Deleting a flow log disables the flow log service for the resource, and no new flow log records or log streams are created. It does not delete any existing flow log records or log streams for a network interface. To delete an existing log stream, you can use the CloudWatch Logs console. After you've deleted a flow log, it can take several minutes to stop collecting data.

The capture window is a duration of time during which the flow logs service aggregates data before publishing flow log records. The capture window is approximately 10 minutes, but can take up to 15 minutes. A flow log record is a space-separated string that has the following format:

version account-id interface-id srcaddr dstaddr srcport dstport protocol packets bytes start end action log-status

The fields represent flow Logs version , aws account ID,ID of ENI on which log stream is being captured, source IP , destination IP , source port, destination port, IANA protocol number , number of packet and bytes,capture window start and end time in seconds, action(accept or reject) and log status(OK-data is logging normally in cloudwatch, NODATA- there was no traffic during the capture window, SKIPDATA – some data was skipped during capture window due to some capacity constraint or an internal error)

* Limitations of Flow Logs:
* We cannot enable flow logs for VPC peered without VPC unless the peered VPC is in our account.
* If your network interface has multiple IPv4 addresses and traffic is sent to a secondary private IPv4 address, the flow log displays the primary private IPv4 address in the destination IP address field.
* Traffic such as: Amazon DNS server communication, DHCP communication, communication between default router and reserved IPs (5 in each subnet) , endpoint ENI and Load balancer ENI communication…..such traffic is not monitored by flow logs.
* Comparison between SG and NACL

|  |  |
| --- | --- |
| **Security Group** | **Network ACL** |
| Operates at the instance level (first layer of defense) | Operates at the subnet level (second layer of defense) |
| Supports allow rules only | Supports allow rules and deny rules |
| Is stateful: Return traffic is automatically allowed, regardless of any rules | Is stateless: Return traffic must be explicitly allowed by rules |
| We evaluate all rules before deciding whether to allow traffic | We process rules in number order when deciding whether to allow traffic |
| Applies to an instance only if someone specifies the security group when launching the instance, or associates the security group with the instance later on | Automatically applies to all instances in the subnets it's associated with (backup layer of defense, so you don't have to rely on someone specifying the security group) |

**Route Tables**

* A set of rules called routes that determine where the network traffic is directed. A route is at its basic level an outbound only rule for forwarding packets.
* Each subnet must be associated with a route table.
* VPC has an implicit router and comes with a main table that you can modify but cannot delete.
* Each route has a destination CIDR (where it says it wants to go ultimately) and a target (where should it be forwarded).
* Every route table has a local route that is for communication within the VPC. If more than one IPv4. IPv6 blocks are present in a VPC, a local route for each block must be specified.
* When you add an IG, egress-only gateway, virtual private gateway, NAT, peering connection, or a VPC endpoint, route tables should be updated for the subnet/VPC so that the devices are utilized correctly.
* There is a limit on route tables per VPC, and routes per route table.
* We use the most specific route in your route table that matches the traffic to determine how to route the traffic (longest prefix match). For example, the following route table has a route for IPv4 Internet traffic (0.0.0.0/0) that points to an Internet gateway, and a route for 172.31.0.0/16 IPv4 traffic that points to a peering connection (pcx-1a2b3c4d). Any traffic from the subnet that's destined for the 172.31.0.0/16 IP address range uses the peering connection, because this route is more specific than the route for Internet gateway.
* Some basic examples regarding routes:
* For an IG – to make a subnet public, add a route for all traffic destinations for IPv4(0.0.0.0/0) or IPv6(::/0) and set the Target as an IG
* For a NAT – to enable the instances of private subnet to initiate communications over the internet, launch a NAT in a public subnet and add a route of all IPv4 destination (0.0.0.0/0) or IPv6 (::/0) to Target=NAT-id(also the local route should be present in private subnet route table). FYI NAT here refers to any NAT device-gateway or instance.
* For a Virtual Private Gateway – same as NAT
* For a VPC Peering Connection – While creating a peering connection you get a peering ID. For eg : if you are create a connection between two VPCs (A & B) with CIDR blocks 10.0.0.0/16 and 172.31.0.0/16 and the connection ID is pcx-xxxx.

In VPC A -

|  |  |
| --- | --- |
| Destination | Target |
| 10.0.0.0./16 | Local |
| 172.31.0.0/16 | pcx-xxxx |

In VPC B –

|  |  |
| --- | --- |
| Destination | Target |
| 172.31.0.0/16 | Local |
| 10.0.0.0./16 | pcx-xxxx |

**Internet Gateways & Egress-Only Gateways**

* An Internet Gateway is a VPC component that serves two purposes: To provide target in route tables for internet-routable traffic and perform NAT for instances that have been assigned public IPv4 address. Internet gateway is present in VPC
* IG supports IPv6 and IPv4 both traffic
* Enabling Internet Access in a subnet
* Attach an IG to subnet
* Configure route table to forward traffic to IG
* Assign public IP instances you want to expose to the internet for hits
* Configure NACLs and SGs aptly
* When an instance sends a packet destined for any IP on internet, the instance is only aware of the private address space inside the subnet, so the packet goes to the route table so that the packet destined for an internet-address can be forwarded to a device inside the private address (subnet), which knows what to do with this packet, i.e. an IG. Since sending the packet from instance to IG is an internal communication process the source field in the packet contains private IP of the instance. The IG before forwarding the packet performs NAT so that the source field contains the public IP of the instance. Conversely when a packet is sent to an instance inside the subnet, it comes to the IG and it sends the packet to the instance, which is an internal communication so the IG NATs destination from the public IP to private IP.
* Egress-Only Gateway – NAT device for IPv6

**NAT**

* A NAT device enables instance in a private subnet to connect to internet or other AWS services but prevents internet from initiating the communication. When a packet is sent out from subnet the source address is replaced with NAT device’s address and when the response is received by NAT device the destination address is replaced with the private IP of the instance and response is forwarded to the instance. NAT devices ultimately forward the traffic to IG only but pretending as if the request was originated by them. NAT devices only work for IPv4.Nat actually performs Address & Port translation both.
* AWS offers two kinds of NAT devices: Nat Gateways and NAT Instances
* NAT gateways:
* Each NAT gateway is created in a specific Availability Zone and implemented with redundancy in that zone. You have a limit on the number of NAT gateways you can create in an Availability Zone.
* A NAT gateway supports 5 Gbps of bandwidth and automatically scales up to 45 Gbps. If you require more, you can distribute the workload by splitting your resources into multiple subnets, and creating a NAT gateway in each subnet.
* Once a NAT gateway is attached to an Elastic IP (it is done while creating) , the address cannot be disassociated without deleting the NAT gateway.
* The network ACL applies to the NAT gateway's traffic. A NAT gateway uses ports 1024–65535.
* A network interface comes attached to the NAT gateway and this interface comes assigned a private IP from the CIDR block of subnet. The attributes of this interface cannot be modified.
* You can monitor NAT gateway using Cloudwatch . The metric data is provided at a frequency of 1 minute and statistics are recorded for a period of 15 months.
* This is a managed NAT service that provides better availability, higher bandwidth, and requires less administrative effort than NAT instance.
* NAT Instances:
* Are EC2 instances configured with NAT AMIs.
* The performance metrics depend on the instance type
* Is not managed by AWS, instead by us so installing updates, OS patches is all on us. Unlike NAT gateways they are not automatically created redundantly in AZs so to manage failovers scripts should be used.
* IP addressing is in the way of a normal ec2 instance.
* Unlike gateways, Port forwarding can be configured in instances