Amazon EC2 Overview and Networking Introduction for Telecom Companies

Implementation Guide

September 2019

Notices

Contents

About this Guide

Many telecom providers are considering the AWS Cloud for their core networking workloads. This paper describes the Amazon EC2 options that are available and highlights important performance considerations. Networking capabilities and connectivity options available between on-premises telecom environments and the AWS Cloud, such as Amazon Virtual Private Cloud (VPC), AWS Direct Connect (DX), and VPNs, are also discussed.

Overview

Many telecom providers are in the process of building out 5G network infrastructure, assessing their mobile edge computing (MEC) strategy, and moving more of their IT workloads to the cloud. Similarly, AWS has announced AWS Outposts[1], a new service that runs AWS services on-premises and can be used to provide network functions virtualization infrastructure (NFVI) and MEC. With these trends, there is a need for telecom networking engineers to understand AWS elastic computing and its performance characteristics as well as AWS networking services, such as Amazon Virtual Private Cloud (Amazon VPC), AWS Transit Gateway, and AWS Direct Connect (DX). These services allow telecom providers to securely connect their on-premises environments to the cloud and achieve the high availability and performance they require.

In considering NFVI deployments, telecom providers have specific demands and require specific features, such as single root I/O virtualization (SR-IOV), Data Plane Development Kit (DPDK), Anti-affinity group support, Non-Uniform Memory Access (NUMA), and CPU pinning. They also require packet per second (pps) performance that can extend to 100 Gbps+. This whitepaper explains the performance characteristics and evolution of these features across the different elastic compute instance families. This paper assumes a basic understanding of networking concepts, such as virtual private networks (VPNs), and explains how AWS networking relates to what networking engineers do daily in running internal IT and large-scale WAN infrastructures.

Amazon VPC is a logically isolated environment in the AWS Cloud that gives telecom providers complete control over how they allocate their subnets, configure routing, and implement security through access control lists (ACLs) and security groups. AWS Transit Gateway allows inter-VPC and VPC to on-premises environments connectivity at scale.

Finally, services such as DX and VPNs allow telecom providers to connect their environments to the AWS Cloud in a secure and scalable manner, without compromising on availability. This paper also provides an example of an OSS workload running in Amazon VPC and communicating with the telecom provider's network using DX.

Mapping AWS Services to the NFV Framework

To begin, it’s important to understand how AWS services relate to the European Telecommunications Standards Institute (ETSI) network functions virtualization (NFV) framework. It’s impossible to relate all services and the roles that they could play in building the entire stack, as this would be implementation-dependent. Instead, the roles of key services and how they map to the framework will be explained. A high-level mapping of AWS services to the ETSI NFV framework is depicted in the following figure.

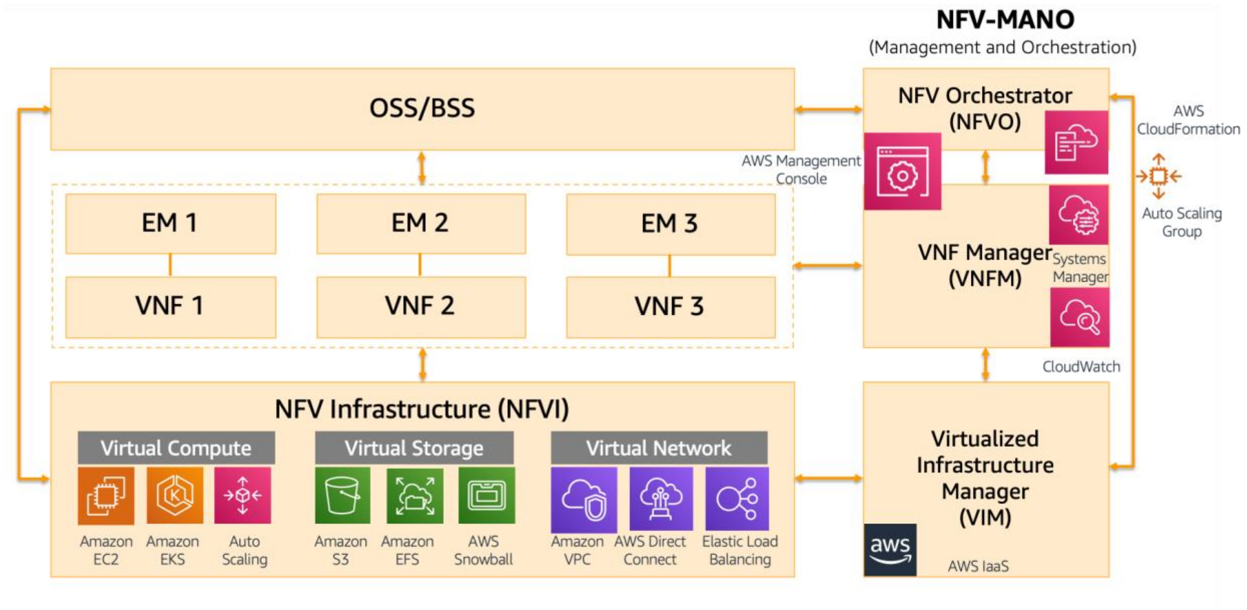


Figure 1 - AWS services mapping to the ETSI NFV framework

The NFVI layer is built using Amazon EC2, Amazon S3, Amazon EBS, instance storage, Amazon VPC, AWS Direct Connect, and AWS Transit Gateway. The Virtualized Infrastructure Manager (VIM) layer in traditional implementations is typically OpenStack,[2] however, in AWS, VIM is represented by AWS native APIs. VIM can also be based on VMware. However, for most core telecom workloads, AWS native APIs represent the most relevant, cloud native approach.

Virtual network functions (VNFs) can run as either VMs or containers on top of the compute and storage infrastructure. The VNF Manager function can be fulfilled by using tools, such as AWS CloudFormation[3], to provision the entire infrastructure stack and then leveraging Elastic Load Balancing and dynamic scaling to elastically spin-up or spin-down the compute environment. In on-premises environments, you must purchase or develop dedicated VNFM software modules. But with AWS Cloud, the VNFM function is performed by AWS services such as AWS CloudFormation and Amazon EC2 Auto Scaling.[4] Amazon CloudWatch[5] provides appropriate alarm triggers to scale up or down the entire environment. CloudFormation allows you to use a simple text file to model and provision, in an automated and secure manner, all the resources needed for your applications across all Regions and accounts. This file serves as the single source of truth for your cloud environment.

The NFV Orchestrator function is provided by the application vendor in partnership with AWS.

Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2[6]) provides a virtual server for running applications, which can scale up or down as your computing requirements change. EC2 instance types are grouped based on target application profiles and include the following: general purpose, compute-optimized, memory-optimized, storage-optimized (high I/O), dense storage, GPU compute, and graphics intensive. Today, there are more than 175 instance types available for a variety of virtual workloads and business needs. In addition to these broad categories, capability choices can be made based on the type of processor (for example, Intel, AMD, or AWS), memory footprint, networking, size, etc. If necessary, each EC2 instance can be associated with a specific choice of Amazon Elastic Block Storage (Amazon EBS[7]), Amazon Elastic Graphics[8], and Amazon Elastic Inference.[9]

The breadth of the options available is shown in the following diagram:

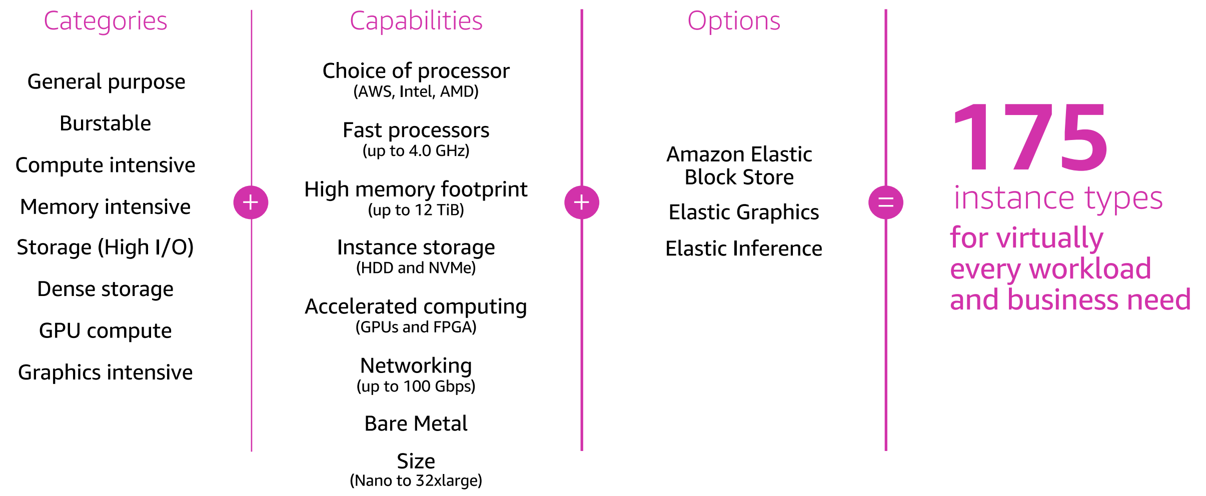

Categories
General purpose
Burstable
Compute intensive
Memory intensive
Storage (High I/O)
Dense storage
GPU compute
Graphics intensive
Capabilities
Choice of processor (AWS, Intel, AMD)
Fast processors (up to 4.0 GHz)
High memory footprint (up to 12 TiB)
Instance storage (HDD and NVMe)
Accelerated computing (GPUs and FPGA)
Networking (up to 100 Gbps)
Bare Metal
Size (Nano to 32xlarge)
Options
Amazon Elastic Block Store
Elastic Graphics
Elastic Inference
175 instance types
for virtually every workload and business need

Figure 2 – Overview of Amazon EC2 instance types

Telecom providers require several performance accelerating features to be supported in their computing infrastructure and this paper will show how AWS supports those features. First, an overview of the different performance and optimization options available in AWS for virtualized environments is provided. Next, a brief history of EC2 performance is given, followed by how that evolution has affected the different instance types. Finally, guidance is provided on what you can expect to achieve with the different instance families in regard to performance.

Overview of Performance and Optimization Options

Single-Root Input/Output Virtualization (SR-IOV) is a mechanism that virtualizes a single PCIe Ethernet controller to make it appear as multiple PCIe devices. Telecom providers have been deploying SR-IOV for their virtualized Evolved Packet Core (vEPC) VNFs to obtain the required performance from their applications and to share a physical NIC among multiple VMs. One of the biggest drawbacks of using SR-IOV is the lack of support for live migration.

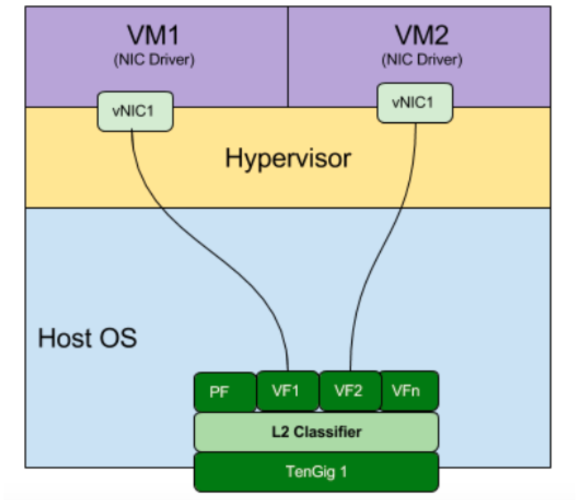


Figure 3 – Illustration of SR-IOV

AWS enhanced networking uses SR-IOV to provide high performance networking capabilities on supported instance types. Support of additional technologies, such as DPDK, is described in Amazon EC2 Performance Evolution and Implementation.

The **Data Plane Development Kit (DPDK)** consists of a set of libraries and user-space drivers to accelerate packet processing on any CPU. Designed to run in user-space, DPDK enables applications to perform their own packet processing operations directly to and from the NIC. By enabling fast packet processing, DPDK makes it possible for the telecom providers to move performance sensitive applications, such as virtualized mobile packet core and voice, to the cloud. DPDK was also identified as a key enabling technology for network functions virtualization (NFV) by ETSI. The main benefits provided by DPDK are lower latency due to kernel and TCP stack bypass, more control of packet processing, and lower CPU overhead. The DPDK libraries provide only minimal packet operations within the application, but enable receiving and sending packets with a minimum number of CPU cycles. It does not provide any networking stack and instead helps to bypass the kernel network stack to deliver high performance.

When it comes to EC2 instance support, DPDK is supported on Enhanced Networking instances, both Intel-based ixgbevf and AWS Elastic Network Adapter (ENA). All Nitro-based instances, such as C5, M5, I3, and T3, as well as Intel-based instances, such as C4, M4, and T2, provide DPDK support. The Amazon drivers, including the DPDK driver for ENA, are available on GitHub.[10] DPDK support for ENA has been available since version 16.04. The ENA Poll Mod Driver (PMD) is a DPDK poll-mode driver for the ENA family. The ENA driver exposes a lightweight management interface with a minimal set of memory mapped registers and an extendable command set through an admin queue.

DPDK and SR-IOV are not mutually exclusive and can be used together. An SR-IOV NIC can write data on a specific VM that hosts a virtual function. The data is then consumed by a DPDK-based application. The following figure illustrates the difference in packet flow between a non-DPDK and a DPDK-optimized application:

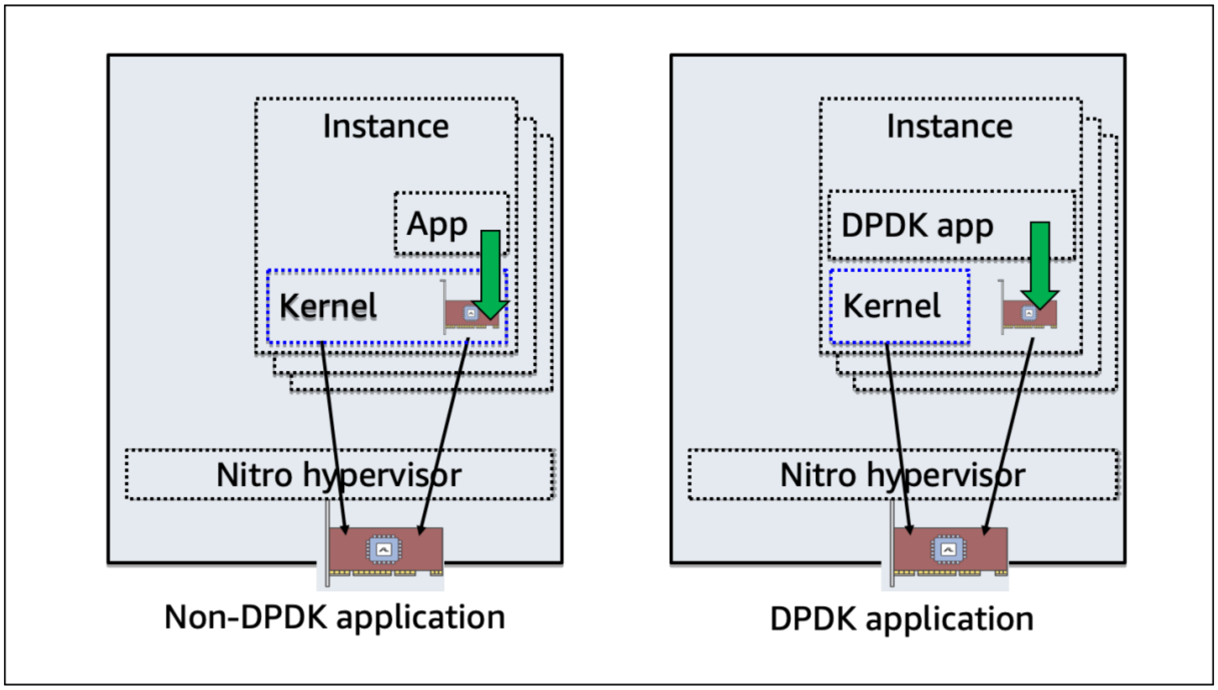


Figure 4 – Non-DPDK vs DPDK packet path

Non-Uniform Memory Access (NUMA) is a shared memory architecture where a cluster of microprocessors in a multiprocessing system is configured so that they can share memory locally, thus improving performance and the ability of the system to be expanded. The memory access time varies with the location of the data to be accessed. If the data resides in local memory, access is fast. If the data resides in remote memory, access is slower. The advantage of the NUMA architecture as a hierarchical shared memory scheme is its potential to improve average case access time through the introduction of fast, local memory. For more information, see Optimizing Applications for NUMA.[11]

In Amazon EC2, all instances that support more than one CPU also support NUMA. These include i3.8xlarge, r5.8xlarge, c5.8xlarge, and above.

Huge Pages can improve performance for workloads that execute large amounts of memory access. This feature of the Linux kernel enables processes to allocate memory pages of size 2MB/1GB (instead of 4K). Additionally, memory allocated using huge pages is pinned in physical memory and cannot be swapped out. Huge page support is configurable on supported instance types. The important thing to note is that huge pages make memory access faster, however you cannot overcommit memory.

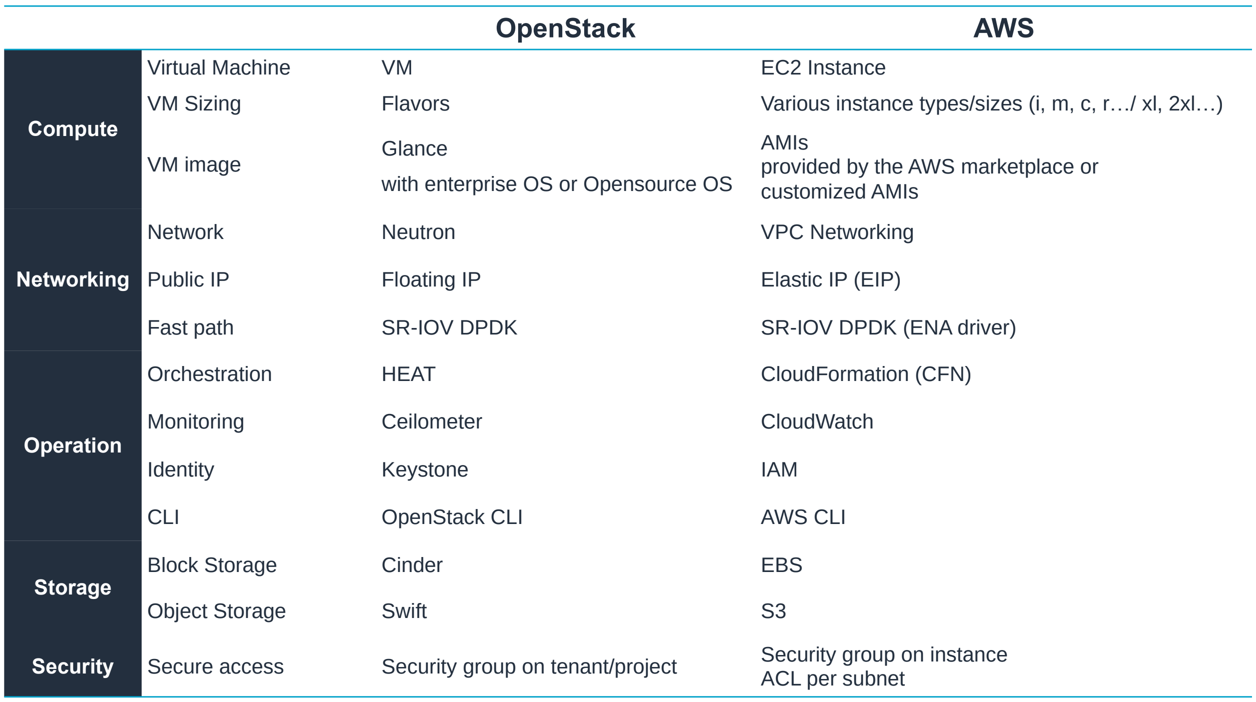
CPU Pinning (CPU Affinity)

CPU Pinning is a technique that enables the binding and unbinding of a process or a thread to a CPU, or a range of CPUs, so that the process or thread will execute only on the designated CPU or CPUs rather than any CPU. This is useful when you want to dedicate vCPU to VNF and avoid sharing and dynamic rescheduling of CPUs. AWS provides this functionality through Placement Groups. Placement groups determine how are instances placed on the underlying hardware and there are two flavors:

- **Cluster** – instances can be clustered into a low latency group in a single Availability Zone. This strategy enables workloads to achieve the low-latency network performance necessary for tightly coupled node-to-node communication that is typical of high performance computing applications and latency sensitive VNFs.
- **Spread** – instances can be spread across the underlying hardware to reduce correlated failures.

For more information, see Amazon EC2 Placement Groups.[12]

Finally, to make it easier to understand AWS performance and networking capabilities, below diagram provides high-level translation of key concepts between OpenStack terms and their equivalent mapping in AWS environment:



		OpenStack	AWS
Compute	Virtual Machine	VM	EC2 Instance
	VM Sizing	Flavors	Various instance types/sizes (i, m, c, r…/ xl, 2xl…)
	VM image	Glance with enterprise OS or Opensource OS	AMIs provided by the AWS marketplace or customized AMIs
Networking	Network	Neutron	VPC Networking
	Public IP	Floating IP	Elastic IP (EIP)
	Fast path	SR-IOV DPDK	SR-IOV DPDK (ENA driver)
Operation	Orchestration	HEAT	CloudFormation (CFN)
	Monitoring	Ceilometer	CloudWatch
	Identity	Keystone	IAM
	CLI	OpenStack CLI	AWS CLI
Storage	Block Storage	Cinder	EBS
	Object Storage	Swift	S3
Security	Secure access	Security group on tenant/project	Security group on instance ACL per subnet

Figure 5 – OpenStack and AWS terminology comparison

Amazon EC2 Performance Evolution and Implementation

AWS has evolved its EC2 platform from the early days of cc2 instances, which used the Xen hypervisor and paravirtualization with up to 10 Gbps of throughput, to the current Nitro-based family, which scale up to 100 Gbps (and millions of pps) for the largest instance types, such as c5n.

In order to improve performance in virtualized environment, SR-IOV technology has been used to bypass the hypervisor, resulting in the first version of enhanced networking, which provided improved performance and lowered jitter and latency. C3 instance family was the first to introduce Enhanced Networking concept and more than halve the latency of its predecessor, CC2. The first release of Enhanced Networking used Intel-based chipsets (ixgbevf) and the later release was based on an in-house based solution called Enhanced Network Adapter (ENA). This is the reason why the references are made to two variants of enhanced networking:

- Enhanced networking using Intel-based chipsets

- Enhanced networking using AWS ENA, fully in-house developed Network Interface Card (NIC).

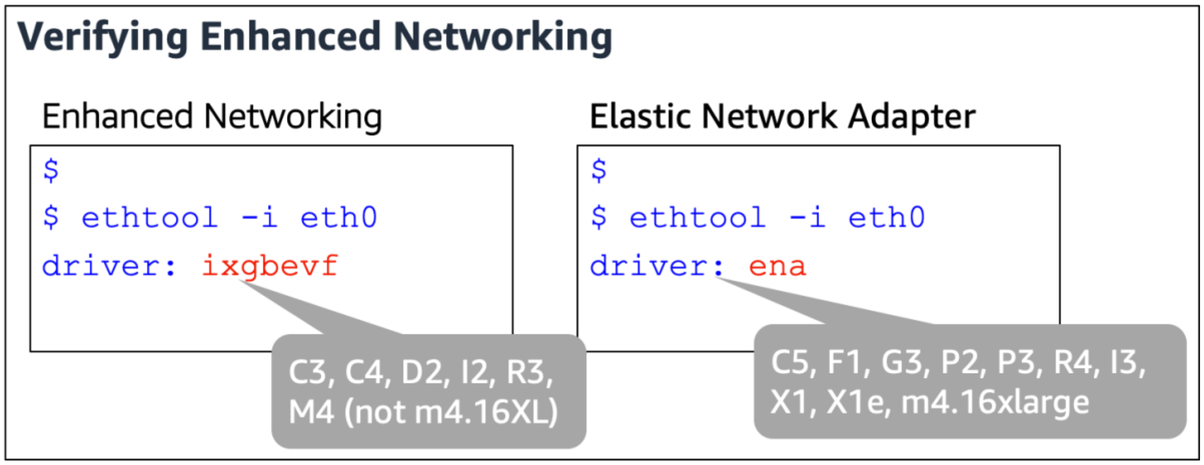
The C4 generation saw the introduction of the Annapurna Labs-based chipset, which replaced Intel, and this instance family provides both networking and storage-optimized performance. The overall performance limit is 10 Gbps, however, workloads requiring both storage and network optimized performance, were able to take advantage of this type of optimization and architecture.

Finally, the culmination of the performance evolution resulted in release of Nitro powered C5 instances. The switch from Intel to ENA has allowed us to deliver much better performance due to increased number of queues (8 instead of 6 with Intel-based chipsets). C5 family delivers performance of up to 25 Gbps and this limit goes to millions of pps and ~100 Gbps with the largest C5n, network optimized instances. It is important to note that chipsets are future proof to deliver performance of up to 400 Gbps.

Nitro system delivers high-speed networking with hardware offload, high-speed EBS storage with hardware offload, NVMe local storage, hardware protection/firmware verification for bare metal instances and all business logic required to control EC2 instances. In more simplified terms, Nitro system is a lightweight hypervisor combined with Nitro security chip and Nitro card for storage and networking.

Enabling Enhanced Networking

As covered in the previous section, Enhanced Networking can be based on Intel ixgbevf or EC2 ENA adaptor. The first step in enabling ENA is to check and verify what type of driver you have. Following commands can be run to determine driver type:


Verifying Enhanced Networking
Enhanced Networking
Elastic Network Adapter
C3, C4, D2, I2, R3, M4 (not m4.16XL)
C5, F1, G3, P2, P3, R4, I3, X1, X1e, m4.16xlarge

```
$
$ ethtool -i eth0
driver: ixgbevf
```

```
$
$ ethtool -i eth0
driver: ena
```

Figure 6 – Verifying Enhanced Networking

With the driver type determined, the following commands can be used to determine if an instance has ixgbevf or ENA enhanced networking enabled:

ixgbevf enhanced networking:

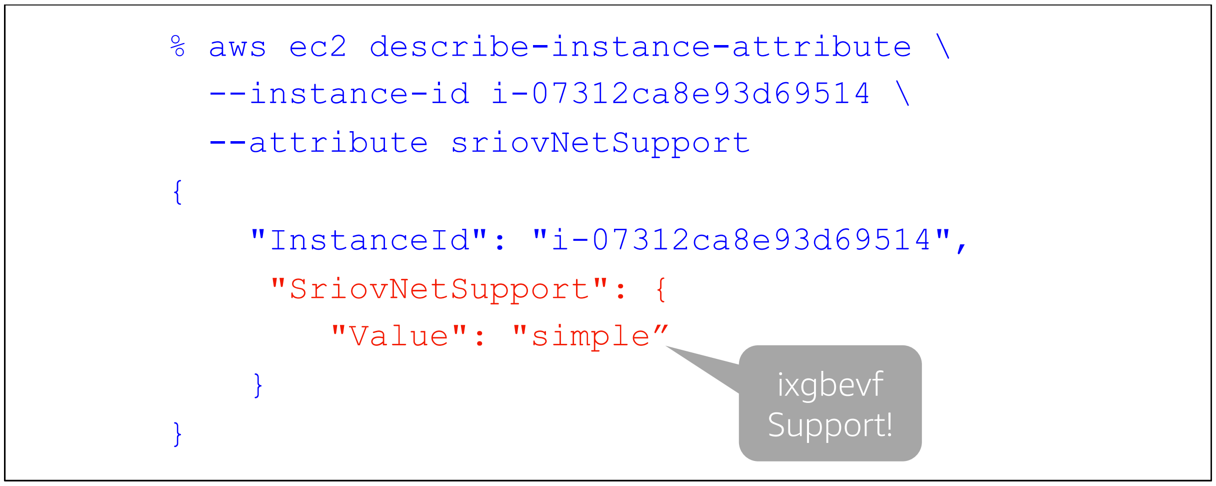
```
 aws ec2 describe-image-attribute --image-id ami_id \
   --attribute sriovNetSupport
```

ENA enhanced networking:

```
 aws ec2 describe-image-attribute --image-id ami_id \
   --attribute enaSupport
```

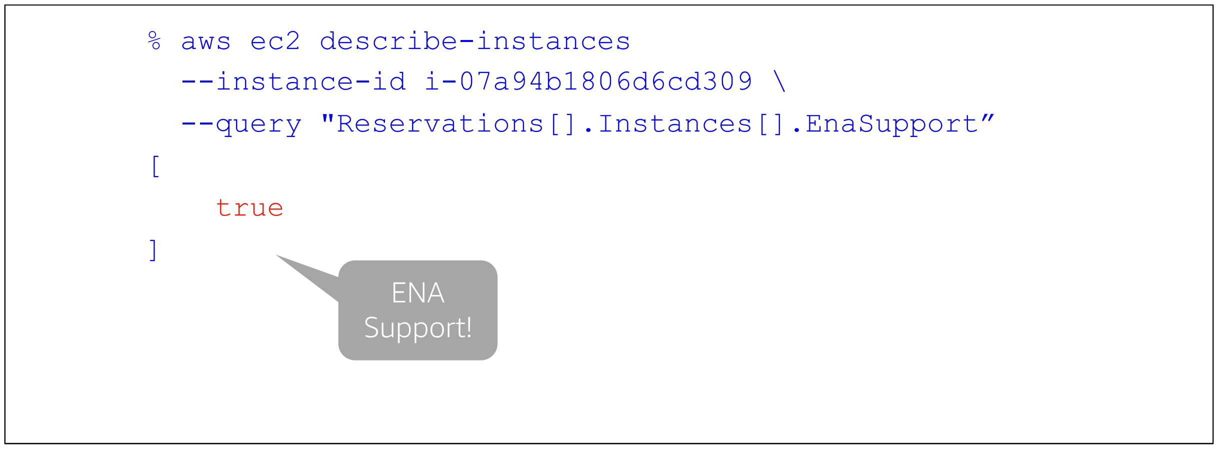
Figure 7 – Verifying Enhanced Networking, continued

Finally, from the following sample output, it can be seen what the output looks like with ixgbevf support and ENA support enabled, respectively:



```
% aws ec2 describe-instance-attribute \
  --instance-id i-07312ca8e93d69514 \
  --attribute sriovNetSupport
{
    "InstanceId": "i-07312ca8e93d69514",
     "SriovNetSupport": {
        "Value": "simple”
    }
}
```

Figure 8 – Verifying Enhanced Networking, continued



```
% aws ec2 describe-instances
  --instance-id i-07a94b1806d6cd309 \
  --query "Reservations[].Instances[].EnaSupport”
[
    true
]
```

Figure 9 – Verifying Enhanced Networking, continued

If an instance has been launched without enhanced networking enabled, the following process can be used to enable it:

1. Connect to the instance that does not have ENA enabled
2. Download the driver
3. Enable ENA support on the instance and verify that it has been enabled

   At this point, a new AMI can be built with ENA enabled so that it can be reused in the future
4. Restart the instance to continue operating with enhanced networking support enabled.

If the instance type supports the Elastic Network Adapter for enhanced networking, the detailed procedures to enable it are outlined in Enabling Enhanced Networking with the Elastic Network Adapter (ENA) on Linux Instances.[13]

If the instance type supports the Intel 82599 Virtual Function interface for enhanced networking, the detailed procedures to enable it are outlined in Enabling Enhanced Networking with the Intel 82599 VF Interface on Linux Instances.[14]

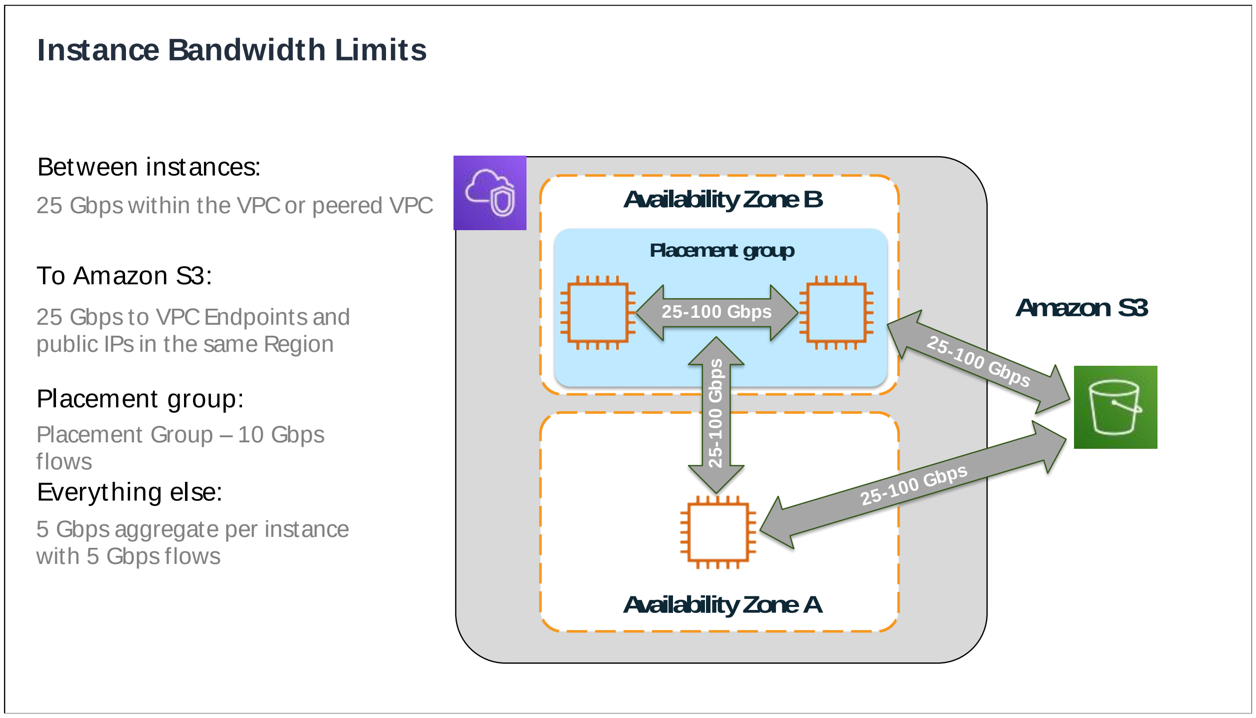
Overall Instance Bandwidth Limitations

As a general guide, the smaller sizes of C5, M5, and R5 instance types can sustain up to 10-Gbps network performance. Larger instance sizes can sustain between 10–25 Gbps. Smaller sizes of C5N provide up to 25 Gbps with the largest C5n instances scaling up to 100 Gbps. Some examples of instance type, configuration, and associated limits are provided in the following table:

Model	vCPU	Mem (GiB)	Network Performance (Gbps)	Model	vCPU	Mem (GiB)	Network Performance (Gbps)
c5.large	2	4	Up to 10	c5n.large	2	4	Up to 25
c5.xlarge	4	8	Up to 10	c5n.xlarge	4	8	Up to 25
c5.2xlarge	8	16	Up to 10	c5n.2xlarge	8	16	Up to 25
c5.4xlarge	16	32	Up to 10	c5n.4xlarge	16	32	Up to 25
c5.9xlarge	36	72	10	c5n.9xlarge	36	72	50
c5.18xlarge	72	144	25	c5n.18xlarge	72	144	100

Table 1 – c5 and c5n instance family configuration and performance comparison

Aggregate bandwidth throughput for instances between Availability Zones (within a VPC) or between instances in a peered VPC scenario is 25–100 Gbps, depending on instance type (see Table 1). Similarly, aggregate bandwidth to VPC endpoints, such as Amazon S3, is 25-100 Gbps. Single TCP flow is limited to 10 Gbps for instances in the same placement group and 5 Gbps between instances anywhere else. (TCP flow is defined as traffic going through a single TCP port.) A placement group is a logical grouping, or cluster, of instances within a single Availability Zone, that allows applications to use low latency 10-Gbps network. These limitations are depicted in the following diagram:



Instance Bandwidth Limits

Between instances:
25 Gbps within the VPC or peered VPC

To Amazon S3:
25 Gbps to VPC Endpoints and public IPs in the same Region

Placement group:
Placement Group – 10 Gbps flows

Everything else:
5 Gbps aggregate per instance with 5 Gbps flows

Figure 10 – Various instance bandwidth limits

Amazon Virtual Private Cloud

Amazon Virtual Private Cloud (Amazon VPC[15]) is the virtual data center in the AWS Cloud. A VPC closely resembles the traditional network that an organization might operate in their own data center, but with all the benefits of elastic and on-demand scaling. Like a traditional data center, VPCs can have public and/or private subnets. Private subnets do not have routes to the internet gateway, but public subnets do. You have complete control over your virtual networking environment, including the selection of your IP address range, creation of subnets, and configuration of route tables and network gateways. You can use both IPv4 and IPv6 in your VPC for secure and easy access to resources and applications.

As in traditional data centers, you can control the flow of inbound and outbound traffic by using network access control lists (NACLs). NACLs act as a firewall for associated subnets and are stateless. To control traffic flow at the instance level, use security groups. Security groups act as firewalls for associated EC2 instances and are stateful, which automatically allows return traffic without needing to define special rules.

In addition to public and private IP addresses, it’s important to understand concept of an Elastic IP address and elastic network interface (ENI). An ENI is analogous to a virtual network interface card (NIC) and you can apply multiple ENIs to an instance. You can also move an ENI to another instance in the same subnet. An Elastic IP address is a static public IP address that is applied to an ENI and it can be associated to another instance after an instance is terminated. The main reason why we have Elastic IP addresses is so that rules such as ACLs, DNS entries and similar do not have to change if an instance fails. Multiple EIPs can be applied to an ENI. The concept of Elastic IP addresses is particularly useful when designing high availability workloads, where an Elastic IP address gets assigned as a secondary IP address of an active instance. That instance is then continuously monitored through CloudWatch tools and that Elastic IP address can be switched through a script or API call to another instances, should failure occur.

External connectivity options for VPCs include the following.

An **internet gateway** is a horizontally scaled, highly available VPC component that allows communication between your instances in a VPC and the internet.

A **NAT gateway** enables instances in a private subnet to connect to the internet or other AWS services, but prevents an internet request from initiating a connection with those instances.

A **virtual private gateway** represents the anchor of the AWS side of a VPN connection between Amazon VPC and the customer environment. In case of a VPN connection between VPC and on-premises environment, VGW connects to the customer gateway, which can be a hardware or software appliance.

All of these building blocks have been represented in the following figure to assist in illustrating how they relate to traditional networking constructs and connectivity, which you are already familiar with.

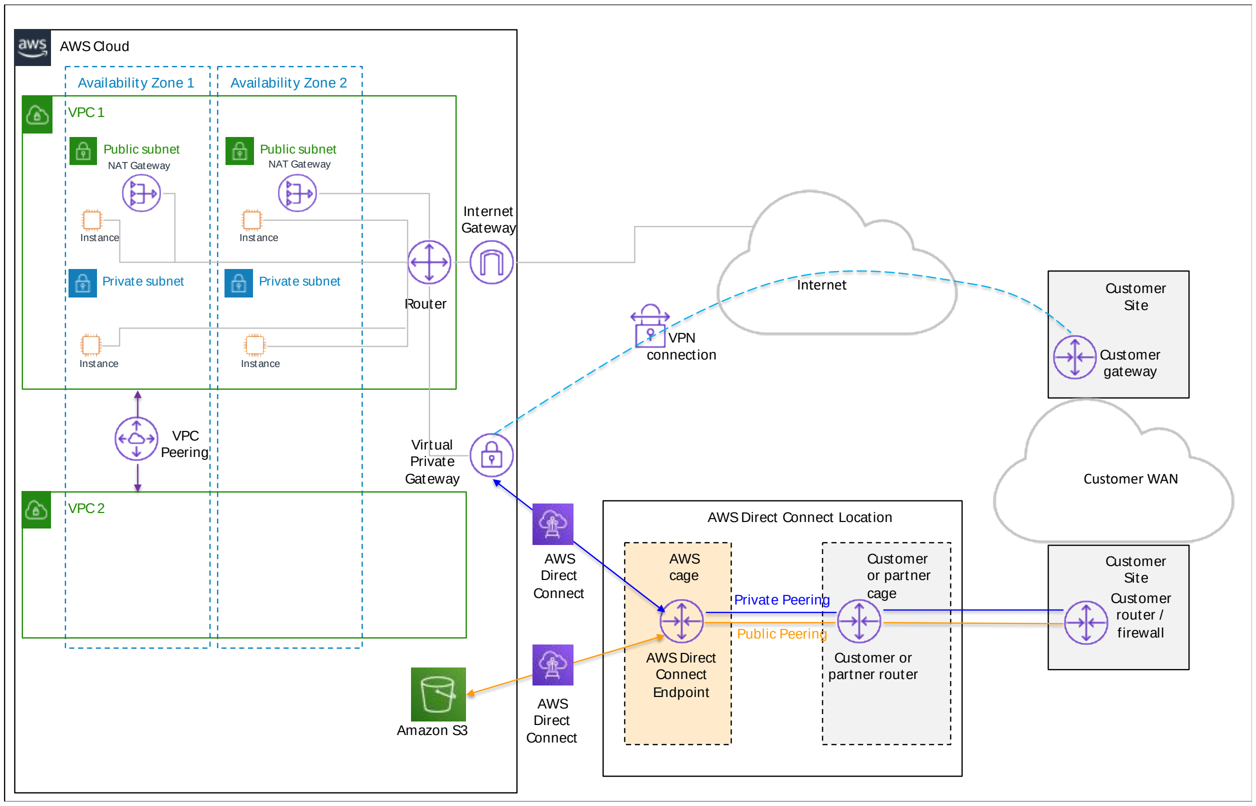


Figure 11 – Sample connectivity diagram between Amazon VPC and on-premises environment with DX and VPN connectivity

You can establish connectivity between two different VPCs by using a **VPC peering** connection. VPC peering allows instances in either VPC to communicate with each other as if they were within the same network. VPCs can be in different Regions and belong to different accounts. Since VPC peering is effectively point-to-point connectivity, it can be operationally costly and cumbersome to use without the ability to centrally manage the connectivity policies. That was the primary reason for introducing AWS Transit Gateway.

AWS Transit Gateway

As you grow the number of workloads running on AWS, you'll need to be able to scale your networks across multiple accounts and VPCs. Previously, you had to connect pairs of VPCs using VPC peering. Recently, AWS introduced AWS Transit Gateway[16], which provides a more scalable way for interconnecting multiple VPCs.

With AWS Transit Gateway, you only need to create and manage a single connection from the central gateway to each Amazon VPC, on-premises data center, or remote office across your network. AWS Transit Gateway acts as a hub that controls how traffic is routed among all the connected networks, which act like spokes. This hub and spoke model significantly simplifies management and reduces operational costs because each network only has to connect to AWS Transit Gateway and not to every other network. Any new VPC is simply connected to the gateway and is then automatically available to every other network that is connected. This ease of connectivity makes it easy to scale your network as you grow. The following before and after diagrams illustrate the benefit of using AWS Transit Gateway:

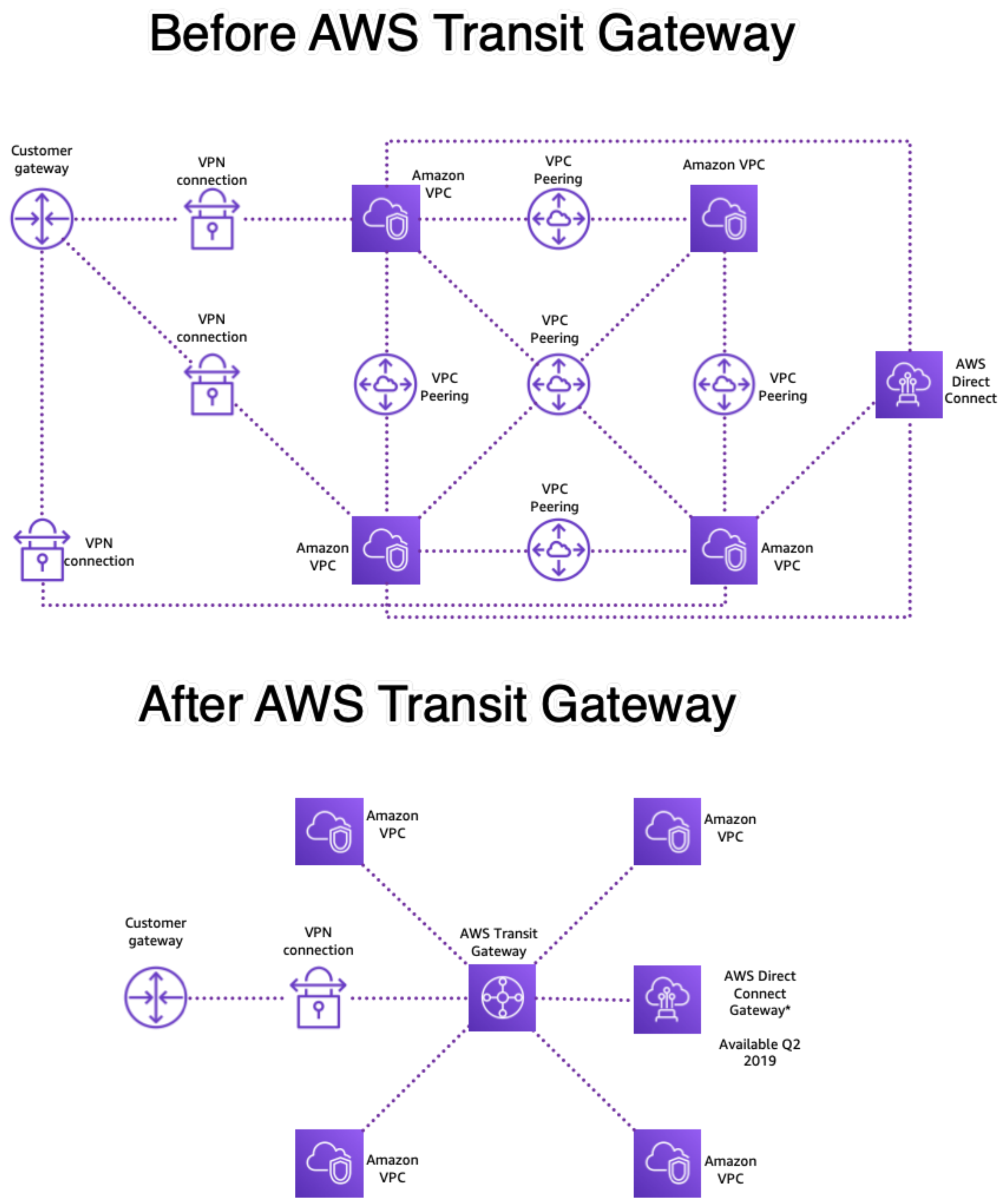


Figure 12 – Network connectivity before and after introducing AWS Transit Gateway

Finally, Elastic Load Balancing[17] allows incoming traffic to be equally distributed across multiple EC2 instances in a VPC and increases the availability of your application. While Elastic Load Balancing supports Application, Classic, and Network Load Balancers, typically only Network Load Balancers will be used for telecom workloads. Network Load Balancers function at Layer 4 of the OSI model, support both TCP and UDP traffic, and can handle millions of requests per second.

Network Performance Troubleshooting

For performance and troubleshooting purposes, you can take advantage of two features:

- VPC Flow Logs
- Traffic Mirroring

VPC Flow Logs[18] enable you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data can be published to Amazon S3 or Amazon CloudWatch Logs. In addition to using flow logs for troubleshooting purposes, such as determining why traffic is not reaching a particular instance, they also can be used as a security tool to monitor the traffic that is reaching your instance.

Traffic Mirroring[19] allows you to capture and inspect network traffic at scale for troubleshooting issues, gaining greater operational insights, implementation of security and compliance controls. Unlike VPC Flow Logs, the destination can be an enhanced network interface or a Network Load Balancer. Both instance traffic and mirroring traffic count towards the overall instance performance, therefore right-sizing both the source and destination instances is an important consideration.

AWS Direct Connect and VPNs

AWS Direct Connect (DX) provides a dedicated connection from your on-premises network to one or more Amazon VPCs. It’s possible to create a single sub-1 Gbps connection or use a link aggregation group (LAG) to aggregate multiple 1 GBps or 10-Gbps connections into a single managed connection. DX uses VLANs to access Amazon EC2 instances running within the VPC. DX supports both static and dynamic routing through BGP. One of the following virtual interfaces (VIFs) must be created in order to use a DX connection:

- **Private virtual interface** – used to access VPC resources using private IP addresses

- **Public virtual interface** – used to access all AWS public services using public IP addresses
- **Transit virtual interface** – used to access one or more AWS Transit Gateways associated with DX gateways.

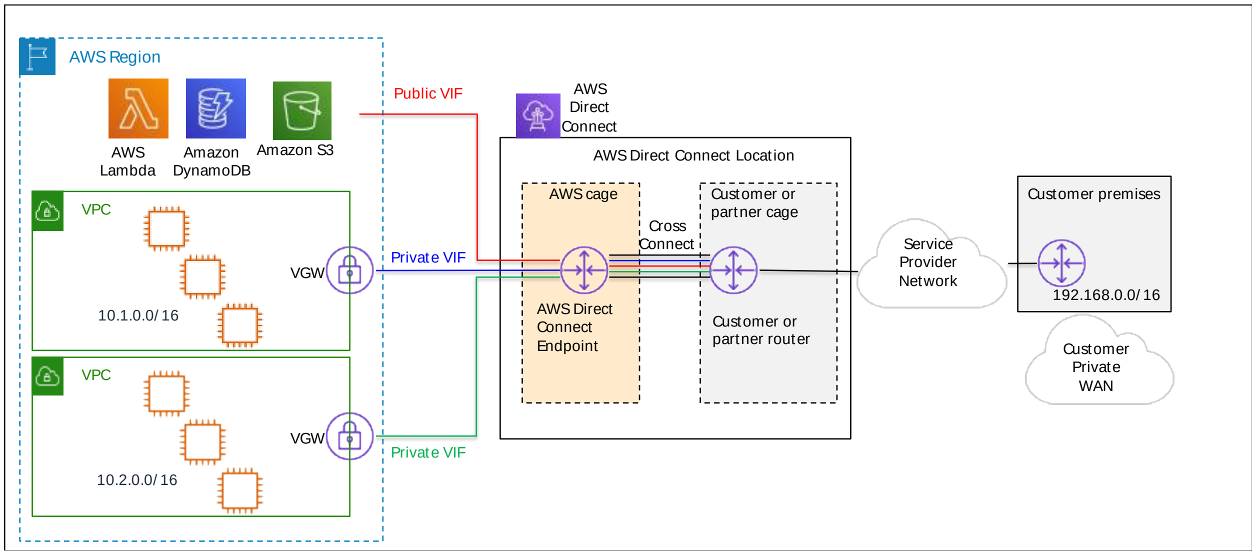


Figure 13 – AWS Direct Connect

Typically, DX is used for critical, latency sensitive workloads given the dedicated nature of the connectivity. AWS also offers a Service Level Agreement for AWS Direct Connect as per the following policy: https://aws.amazon.com/directconnect/sla/.

If the workload does not require the dedicated nature of DX, using an AWS managed VPN provides the option of creating an IPsec VPN connection over the internet between your on-premises environment and Amazon VPC. With an AWS managed VPN, you can take advantage of automated multi-data center redundancy and failover, which is built into the AWS side of VPN. Basically, a virtual private gateway will terminate two distinct VPN endpoints in two separate data centers. The redundancy can be further improved by also implementing redundancy at your side of connection and terminating VPN endpoints on two separate customer gateways at the on-premises environment. Finally, both dynamic and static routing options are supported to give you flexibility in setting your routing configuration. Dynamic routing uses BGP peering to exchange routing information between AWS and your on-premises environment. With dynamic routing, you can also specify routing priorities, policies, and weights (metrics) in your BGP advertisements and influence the network path taken between your networks and AWS.

The potential drawbacks of using an AWS managed VPN are that availability is dependent on the internet conditions, and the VPN adds complexity to implementing redundancy and failover (if necessary) at your end. DX, on the other hand, provides dedicated connectivity and minimal latency. However, it does require new network circuits to be provisioned through your hosting provider, unless you are the hosting provider.

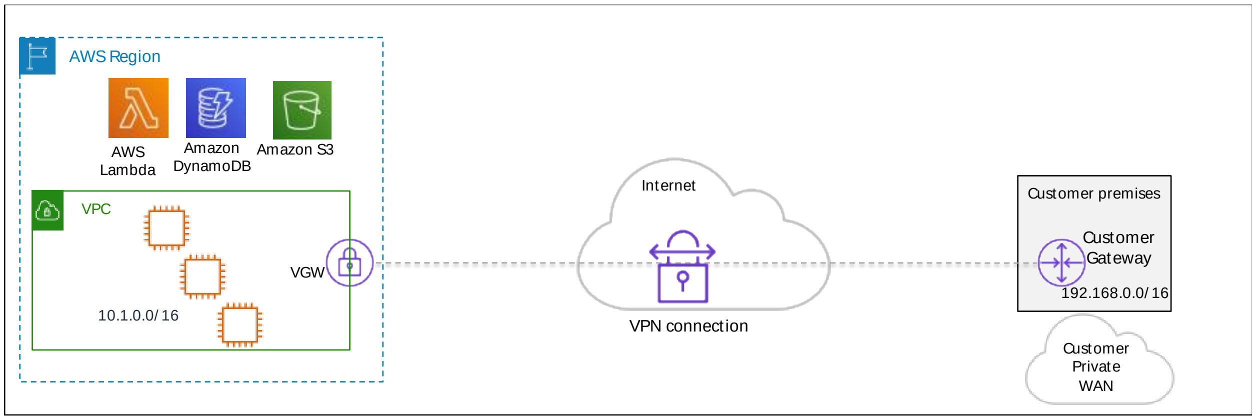


Figure 14 – VPN Connectivity

VPC Design Example with Telecom OSS Workload

This section provides an example of an OSS workload running in the AWS Cloud and communicating with a telecom's network via DX link. At a high level, the application is gathering performance data from a variety of network elements and this data is being correlated and presented through an OSS application running in Amazon VPC. In this example, the application is provided as a SaaS offering, and is managed by the SaaS provider in a dedicated VPC. The VPC is connected with both the telecom network and the operations network.

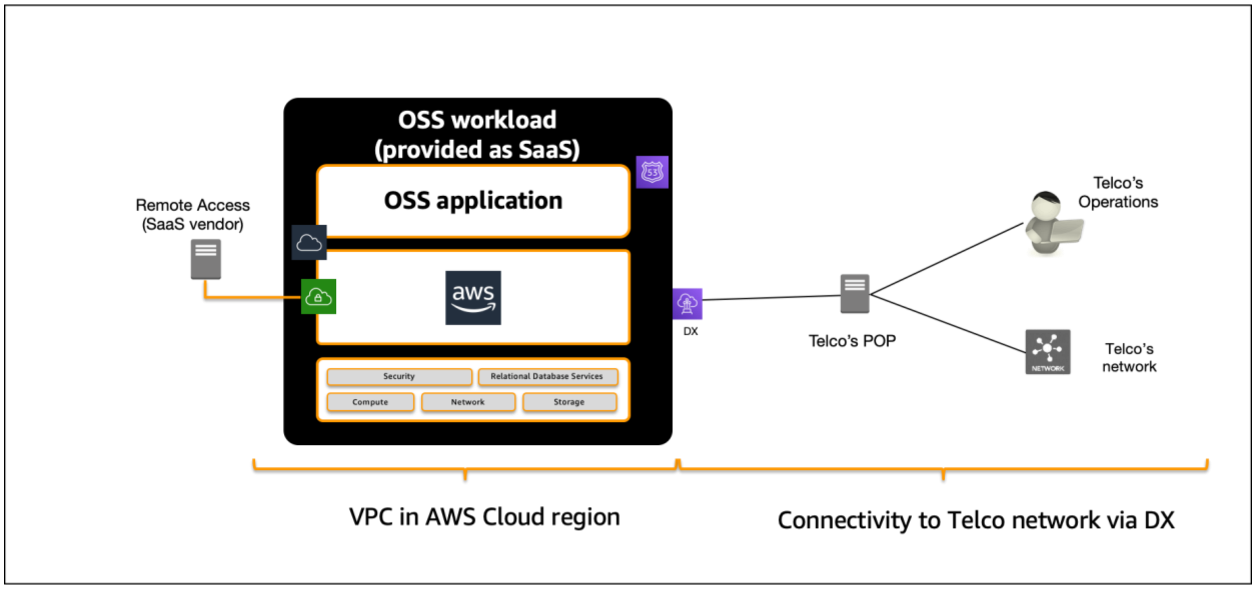


Figure 15 – OSS workload in Amazon VPC

Workloads can run as virtual machines (VMs) or containers. In this particular example, the OSS application is implemented as container workloads on Red Hat OpenShift, using a Multi-AZ deployment for high availability purposes. Amazon EBS, Amazon Elastic File System (Amazon EFS[20]) and Amazon Relational Database Service (Amazon RDS[21]) are used in the overall design.

The advantage of this cloud-based implementation for telecom providers is:

- Elastic scaling of the entire application using Elastic Load Balancing and automatic scaling
- Secure data handling as incoming data into the VPC is encrypted, data leaving the VPC is encrypted, and data held within the VPC is encrypted at both the storage and database level
- Secure access through ACLs, security groups, and multi-factor authentication (MFA)
- High availability implementation spanning three Availability Zones, with private and public subnet in each AZ. Internet gateway provides internet access to each subnet.
- AWS CloudFormation is used to deploy the entire infrastructure without the need for manual installation and stand-up.

The following diagram provides a logical representation of the key building blocks in the VPC and their connectivity to the telecom network through a DX connection:

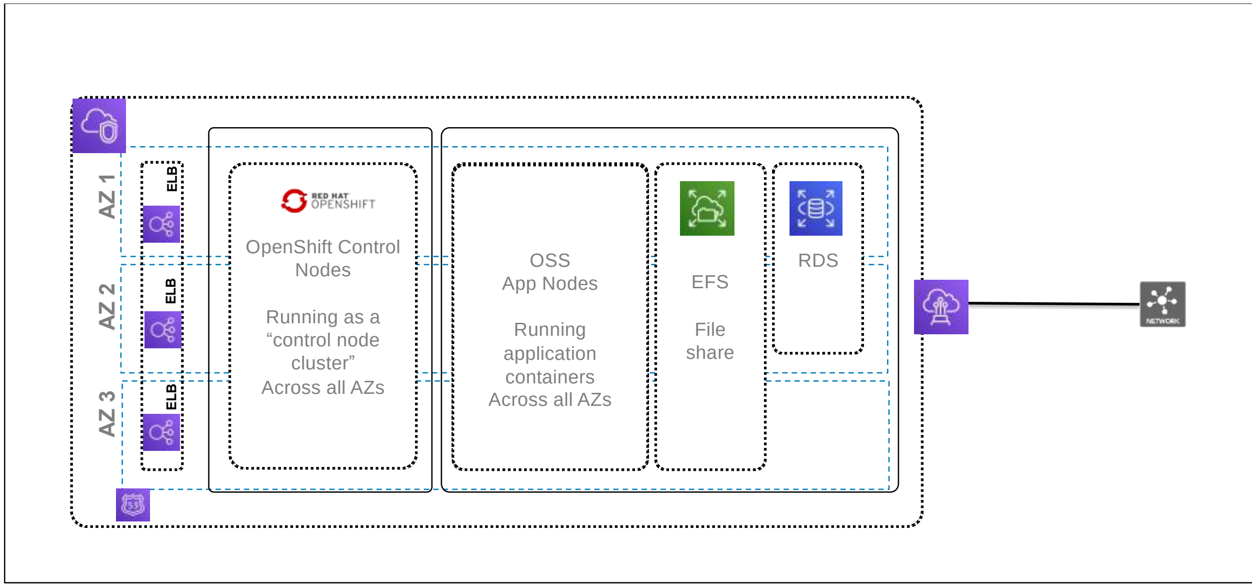


Figure 16 – Logical representation of the OSS workload architecture in the cloud

Conclusion

AWS Cloud offers a wide variety of elastic compute choices and provides a strong, cloud native alternative to traditional NFVI platforms, such as OpenStack. With 100 Gbps capable, compute-optimized instances, and the support of performance-enhancing tools, such as DPDK, CPU affinity, NUMA, and Hugepages, AWS Cloud provides a suitable environment for running mission critical telecom workloads. AWS networking services give telecom providers the ability to extend their on-premises networking to the cloud in a secure and reliable manner by using DX, Amazon VPC, VPNs, AWS Transit Gateway, and Elastic Load Balancing.

Contributors

Contributors to this document include:

- Rada Stanic, Principal Solutions Architect, APAC
- Dr. Young Jung, Senior Partner Solutions Architect, AWS Telecom Business Unit
- Tipu Qureshi, Principal Cloud Support Engineer, AWS Premium Support

Additional Resources

For additional information, see:

- Exploring NUMA on Amazon Cloud Instances[22]
- Enabling Enhanced Networking with the Elastic Network Adapter (ENA) on Windows Instances[23]
- Blog post: Amazon VPC for On-Premises Network Engineers – Part 1[24]
- Amazon Virtual Private Cloud Connectivity Options whitepaper[25]
- NFV reference architecture for deployment of mobile networks[26]

Document Revisions

Date	Description
September 2019	First publication

Notes

1 https://aws.amazon.com/outposts/

2 https://www.openstack.org/

3 https://aws.amazon.com/cloudformation/
4 https://aws.amazon.com/ec2/autoscaling/
5 https://aws.amazon.com/cloudwatch/

6 https://aws.amazon.com/ec2/

7 https://aws.amazon.com/ebs/
8 https://aws.amazon.com/ec2/elastic-graphics/
9 https://aws.amazon.com/machine-learning/elastic-inference/

10 https://github.com/amzn/amzn-drivers

11 https://software.intel.com/en-us/articles/optimizing-applications-for-numa

[12] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html

[13] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/enhanced-networking-ena.html

[14] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/sriov-networking.html

[15] https://aws.amazon.com/vpc/

[16] https://aws.amazon.com/transit-gateway/
[17] https://aws.amazon.com/elasticloadbalancing/
[18] https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs.html
[19] https://docs.aws.amazon.com/vpc/latest/mirroring/what-is-traffic-mirroring.html
[20] https://aws.amazon.com/efs/
[21] https://aws.amazon.com/rds/

[22] http://techblog.cloudperf.net/2016/09/exploring-numa-on-amazon-cloud-instances.html

[23] https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/enhanced-networking-ena.html

[24] https://aws.amazon.com/blogs/apn/amazon-vpc-for-on-premises-network-engineers-part-one/

[25] https://docs.aws.amazon.com/whitepapers/latest/aws-vpc-connectivity-options/introduction.html

[26] https://access.redhat.com/sites/default/files/attachments/nfvrefarch_v3.pdf