AWS

EC2--There can be various reasons why an EC2 (Amazon Elastic Compute Cloud) server is not working, including:

1. Instance connectivity issues: The instance may not be accessible over the network due to network configuration, firewall rules, or incorrect security group settings.
2. Operating system issues: The instance may have a misconfigured or corrupted operating system, or the instance may not have been started properly.
3. Insufficient resources: The instance may not have enough CPU, memory, or disk space to perform its intended tasks, leading to performance degradation or failure.
4. Application issues: The application running on the instance may have bugs or compatibility issues, or the application may not have been properly configured.
5. Storage issues: The instance's root volume or attached volumes may have issues, such as low disk space, or the instance may not have proper access to the storage.
6. Network connectivity issues: The instance may have issues with connecting to other resources in the network, such as an RDS database or S3 bucket.
7. Hardware failure: The instance's underlying hardware may have failed, resulting in instance termination

Difference between Public and private Subnet

Public subnets are subnets that have direct access to the Internet through an Internet Gateway attached to the VPC. They are used to host resources that need to be publicly accessible, such as web servers or application load balancers.

Private subnets, on the other hand, do not have direct access to the Internet and can only be accessed through a NAT Gateway or VPN connection. They are used to host resources that should not be directly accessible from the Internet, such as databases or backend servers.

In summary, the main differences between EBS and instance store are:

1. Persistence: EBS provides persistency, while instance store is ephemeral and the data is lost if the EC2 instance is terminated.
2. Availability: EBS is network-attached and provides high availability, while instance store is attached to the host computer and may be subject to host failures.
3. Use cases: EBS is ideal for use cases that require persistent storage, while instance store is ideal for use cases that require temporary storage with high disk I/O performance.

there are a few alternative options that can be used to access the instance:

1. Use a different key pair: If you have lost or deleted the original key pair, you can create a new key pair and use it to connect to the instance.
2. Use a bastion host: You can set up a bastion host in a public subnet in your VPC and use it to securely access the instances in private subnets.
3. Use AWS Systems Manager Session Manager: AWS Systems Manager Session Manager allows you to securely access the instance without having to use a key pair. Instead, you can use the AWS Management Console or the AWS CLI to initiate a session to the instance.
4. Use AWS Direct Connect: AWS Direct Connect is a network connection service that allows you to create private connections between your VPC and your own network. You can use AWS Direct Connect to access instances in your VPC without the need for a key pair.

Amazon EC2 (Amazon Elastic Compute Cloud) offers various types of instances, each optimized for specific use cases. Some of the most commonly used instance types are:

1. General Purpose:
   * t2: low to moderate CPU performance, burstable performance.
   * m5: balanced CPU to memory ratio, suitable for a wide range of applications.
2. Compute Optimized:
   * c5: optimized for compute-intensive applications, such as high-performance web servers and batch processing.
3. Memory Optimized:
   * r5: optimized for memory-intensive applications, such as high-performance databases, real-time big data analytics, and in-memory caches.
   * x1: optimized for large-scale, high-performance databases and in-memory analytics.
4. GPU instances:
   * p3: GPU instances for compute-intensive workloads and high-performance computing.
5. ARM instances:
   * a1: ARM instances for cost-effective, general-purpose workloads.
6. Storage Optimized:
   * h1: optimized for large, sequential read and write workloads, such as big data processing and log processing.
7. ARM instances:
   * a1: ARM instances for cost-effective, general-purpose workloads.

The main difference between NACLs and security groups is their scope and level of granularity.

1. NACLs:
   * NACLs are associated with subnets, not individual EC2 instances.
   * NACLs operate at the subnet level and provide more fine-grained control over network traffic compared to security groups.
   * NACLs control both inbound and outbound traffic.
   * NACLs have a stateful inspection of traffic, which means the reply traffic to a permitted incoming request is automatically allowed, regardless of outbound rules.
2. Security Groups:
   * Security groups are associated with individual EC2 instances.
   * Security groups operate at the instance level and provide a less fine-grained control over network traffic compared to NACLs.
   * Security groups control only inbound traffic.
   * Security groups are stateful, meaning that if you allow traffic inbound, the related response traffic will also be allowed, regardless of outbound rules.

Transit Gateway, you can:

* Connect VPCs from multiple AWS accounts to a single transit gateway.
* Connect VPCs across different regions to a single transit gateway.
* Connect on-premises networks to a VPC or multiple VPCs.
* Control the flow of traffic between connected networks using route tables and security groups.

AWS Direct Connect, you can:

* Connect your on-premises data center to AWS over a dedicated, private network connection.
* Use a single connection to access multiple AWS VPCs.
* Use multiple connections for redundancy and increased bandwidth.
* Transfer large amounts of data between your on-premises data center and AWS with lower network latency.
* Use Direct Connect as a network connection for Amazon Virtual Private Cloud (VPC) peering, AWS Transit Gateway, and AWS Direct Connect Gateway.

A Virtual Private Cloud (VPC) endpoint in Amazon Web Services (AWS) is a connection point within a VPC that enables communication between instances in the VPC and AWS services, without the need to traverse the public internet.

VPC endpoints allow you to securely connect your VPC to an AWS service, such as Amazon Simple Storage Service (S3), Amazon DynamoDB, or Amazon SNS, without exposing the service to the public internet. VPC endpoints use private IP addresses, so all communication between instances in your VPC and the AWS service is encrypted and remains within the AWS network.

Amazon Web Services (AWS) Storage Gateway is a hybrid cloud storage service that enables customers to securely store data in the AWS cloud while retaining the low latency and high bandwidth benefits of on-premises storage. It provides seamless integration with popular cloud-based storage services, such as Amazon S3, Amazon Glacier, and Amazon S3 Intelligent-Tiering, for both backup and archiving, as well as disaster recovery.

Storage Gateway provides a virtual machine (VM) appliance that you can install on-premises, in a virtualized environment, or in the cloud, which provides a file-based interface to your data in the cloud. There are three types of Storage Gateway offerings: File Gateway, Volume Gateway, and Tape Gateway.

File Gateway: Allows you to store files in the cloud as objects, while retaining the file system interface that applications and users are familiar with.

Volume Gateway: Provides block-based storage, where data is stored in the cloud as Amazon EBS snapshots, while retaining the iSCSI block interface that applications and users are familiar with. There are two options for the Volume Gateway: Stored Volumes and Cached Volumes.

Tape Gateway: Emulates a virtual tape library, allowing you to archive data to the cloud using the same processes and tools you use for tape backup today.

AWS Storage Gateway provides customers with a cost-effective, flexible, and secure solution for storing, managing, and protecting their data in the cloud.

S3—

Amazon S3 offers a range of storage classes that cater to different data storage and retrieval requirements, each with its own set of trade-offs between performance, cost, and durability. The available storage classes in S3 are:

1. Standard: This is the default storage class and provides high durability, low latency, and high throughput performance. Standard storage is suitable for frequently accessed data that requires rapid access times and high reliability.
2. Intelligent-Tiering: This storage class provides cost optimization for data with unknown or changing access patterns, as it automatically moves data to the most cost-effective tier.
3. S3 One Zone: This storage class stores data in a single availability zone and is suitable for infrequently accessed, non-critical data. It provides lower durability than Standard, but at a lower cost.
4. S3 Glacier: This storage class provides long-term storage for data that is infrequently accessed and for which retrieval times of several hours are acceptable. S3 Glacier is ideal for archives, backups, and disaster recovery.
5. S3 Glacier Deep Archive: This storage class provides the lowest-cost storage for data that is infrequently accessed and for which retrieval times of 12 hours or more are acceptable. S3 Glacier Deep Archive is ideal for long-term archives, backups, and disaster recovery.

Amazon Simple Storage Service (S3) versioning is a feature that allows you to store multiple versions of an object in the same S3 bucket. With versioning, you can preserve, retrieve, and restore every version of every object in your bucket, which makes it easier to recover from both unintended user actions and application failures.

Encryption is an important security control for protecting sensitive data in AWS. AWS provides several options for encrypting data at rest and in transit, including:

1. Server-side encryption: This encryption method uses encryption keys managed by AWS Key Management Service (KMS) or by Amazon S3 itself to encrypt data before it is written to disk, and to decrypt the data when it is read. Server-side encryption with KMS provides an extra layer of security, as the keys are managed by a separate service with its own set of access controls and audit trails.
2. Client-side encryption: This encryption method allows you to encrypt data on your own computer before uploading it to AWS, using encryption keys that you manage. The encrypted data is then stored in an S3 bucket, and is decrypted when you download it.
3. SSL/TLS encryption: This encryption method protects data in transit over the network, by encrypting data as it is transmitted between the client and the server. AWS provides support for SSL/TLS encryption for several services, including Amazon S3, Amazon Elastic Compute Cloud (EC2), and Amazon Simple Queue Service (SQS).

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) service. You can use Route 53 to route traffic to different regions for improved performance and resiliency. Here's how you can do this:

1. Create a record set: Start by creating a record set in Route 53. This is essentially a mapping between a domain name and the IP address of the resource that you want to receive traffic.
2. Create a health check: Create a health check to monitor the health of your resources. This helps Route 53 to determine whether a resource is available to receive traffic.
3. Create a geographic routing policy: Create a routing policy that specifies the geographic locations from which you want Route 53 to route traffic.
4. Associate the routing policy with the record set: Associate the routing policy with the record set you created in step 1. This will determine the traffic flow based on the geographic locations specified in the routing policy.
5. Test the routing policy: Verify that traffic is being routed as expected by testing the routing policy.

Record sets

1. A (Address) record: Maps a domain name to an IPv4 address. An A record is the most basic type of record and is used to route traffic to a web server or other internet-connected resource.
2. AAAA (Address) record: Maps a domain name to an IPv6 address. An AAAA record is similar to an A record, but it's used for IPv6 addresses.
3. MX (Mail exchange) record: Maps a domain name to an email exchange server. An MX record is used to route email traffic to your email server.
4. CNAME (Canonical name) record: Maps a domain name to another domain name. A CNAME record is used to create an alias for a domain name, so that multiple domain names can resolve to the same resource.
5. ALIAS record: Maps a domain name to another AWS resource, such as an Amazon S3 bucket or a CloudFront distribution. An ALIAS record is similar to a CNAME record, but it provides a more efficient routing mechanism for AWS resources.
6. Health Checks: You can use Route 53 health checks to monitor the health of your resources and route traffic to healthy resources.
7. Traffic Flow: You can use Route 53 Traffic Flow to manage and monitor the routing of traffic to your resources. With Traffic Flow, you can create policies for routing traffic based on criteria such as geographic location, endpoint health, and latency.
8. Alias Records: You can use Route 53 alias records to route traffic to Amazon Web Services resources such as Amazon S3 buckets, Amazon CloudFront distributions, and AWS Elastic Beanstalk environments.
9. Geo-Location Routing: Route 53 provides the ability to route traffic based on the geographic location of your users, which can help you deliver a faster and more relevant experience to your users.
10. Latency-Based Routing: You can use Route 53 to route traffic based on the lowest latency between your users and your resources, which can help you deliver a faster and more reliable experience to your users.
11. Simple Routing Policies: Route 53 provides simple routing policies such as weight-based routing, failover routing, and multivalue answer routing, which you can use to route traffic to multiple resources in a flexible and scalable manner.