Common errors in Amazon EMR

Sometimes, clusters fail or are slow to process data. The following sections list some common cluster issues With suggestions on how to fix them.

Topics

Error codes with ErrorDetail information

When an EMR cluster terminates with an error, the `DescribeCluster` and `ListClusters` APIs return an error code and an error message. For some cluster errors, the `ErrorDetail` data array can help you troubleshoot the failure.

Errors that include an `ErrorDetail` array provide the following details:

`ErrorCode`

A unique error code that you can use for programmatic access.

`ErrorData`

A list of identifiers in key-value pairs that you can use for programming or manual lookup. For descriptions of the `ErrorData` values that an error code includes, see the troubleshooting page for the error code.

`ErrorMessage`

Description of the error with a link to more information in the Amazon EMR documentation.

> **Note**
>
> We don't recommend that you parse the text from `ErrorMessage` because this text is subject to change.

Error codes by category

- Bootstrap failure error codes
- Internal error codes
- Validation failure error codes

Bootstrap failure error codes

The following sections provide troubleshooting information for bootstrap failure error codes.

Topics

- BOOTSTRAP_FAILURE_PRIMARY_WITH_NON_ZERO_CODE

- BOOTSTRAP_FAILURE_BA_DOWNLOAD_FAILED_PRIMARY
- BOOTSTRAP_FAILURE_FILE_NOT_FOUND_PRIMARY

BOOTSTRAP_FAILURE_PRIMARY_WITH_NON_ZERO_CODE

Overview

When a cluster terminates with a `BOOTSTRAP_FAILURE_PRIMARY_WITH_NON_ZERO_CODE` error, a bootstrap action has failed in the primary instance. For more information about bootstrap actions, see Create bootstrap actions to install additional software.

Resolution

To resolve this error, review the details returned in the API error, modify your bootstrap action script, and create a new cluster with the updated bootstrap action.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`primary-instance-id`

The ID of the primary instance where the bootstrap action failed.

`bootstrap-action`

The ordinal number for the bootstrap action that failed. A script with a `bootstrap-action` value of 1 is the first bootstrap action to run on the instance.

`return-code`

The return code for the bootstrap action that failed.

`amazon-s3-path`

The Amazon S3 location of the bootstrap action that failed.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the root cause of the bootstrap action error. Then launch a new cluster.

1. Review the bootstrap action log files in Amazon S3 to identify the root cause for the failure. To learn more on how to view Amazon EMR logs, see View log files.
2. If you turned on cluster logs when you created the instance, refer to the `stdout` log for more information. You can find the `stdout` log for the bootstrap action in this Amazon S3 location:

   ```
   s3://EXAMPLE-BUCKET/logs/Your_Cluster_Id/node/Primary_Instance_Id/bootstrap-actions/Failed_Bootstrap_Action_Number/stdout.gz
   ```

   For more information on cluster logs, see Configure cluster logging and debugging.
3. To determine the bootstrap action failure, review the exceptions in the `stdout` logs, and the `return-code` value in `ErrorData`.
4. Use your findings from the previous step to revise your bootstrap action so that it avoids exceptions or can gracefully handle exceptions when they occur.
5. Launch a new cluster with your updated bootstrap action.

BOOTSTRAP_FAILURE_BA_DOWNLOAD_FAILED_PRIMARY

Overview

A cluster terminates with the `BOOTSTRAP_FAILURE_BA_DOWNLOAD_FAILED_PRIMARY` error when the primary instance can't download a bootstrap action script from the Amazon S3 location that you specify. Potential causes include the following:

- The bootstrap action script file isn't in the specified Amazon S3 location.
- The service role for Amazon EC2 instances on the cluster (also called the *EC2 instance profile for Amazon EMR*) doesn't have permissions to access the Amazon S3 bucket where the bootstrap action script resides. For more information about service roles, see Service role for cluster EC2 instances (EC2 instance profile).

For more information about bootstrap actions, see Create bootstrap actions to install additional software.

Resolution

To resolve this error, ensure that your primary instance has appropriate access to the bootstrap action script.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`primary-instance-id`

The ID of the primary instance where the bootstrap action failed.

`bootstrap-action`

The ordinal number for the bootstrap action that failed. A script with a `bootstrap-action` value of 1 is the first bootstrap action to run on the instance.

`amazon-s3-path`

The Amazon S3 location of the bootstrap action that failed.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the root cause of the bootstrap action error. Then launch a new cluster.

Troubleshooting steps

1. Use the `amazon-s3-path` value from the `ErrorData` array to find the relevant bootstrap action script in Amazon S3.
2. If you turned on cluster logs when you created the instance, refer to the `stdout` log for more information. You can find the `stdout` log for the bootstrap action in this Amazon S3 location:

```
s3://EXAMPLE-BUCKET/logs/Your_Cluster_Id/node/Primary_Instance_Id/bootstrap-
actions/Failed_Bootstrap_Action_Number/stdout.gz
```

For more information on cluster logs, see Configure cluster logging and debugging.

3. To determine the bootstrap action failure, review the exceptions in the `stdout` logs, and the `return-code` value in `ErrorData`.
4. Use your findings from the previous step to revise your bootstrap action so that it avoids exceptions or can gracefully handle exceptions when they occur.
5. Launch a new cluster with your updated bootstrap action.

BOOTSTRAP_FAILURE_FILE_NOT_FOUND_PRIMARY

Overview

The `BOOTSTRAP_FAILURE_FILE_NOT_FOUND_PRIMARY` error indicates that the primary instance can't find the bootstrap action script that the instance just downloaded from the specified Amazon S3 bucket.

Resolution

To resolve this error, confirm that your primary instance has appropriate access to the bootstrap action script.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`primary-instance-id`

The ID of the primary instance where the bootstrap action failed.

`bootstrap-action`

The ordinal number for the bootstrap action that failed. A script with a `bootstrap-action` value of `1` is the first bootstrap action to run on the instance.

`amazon-s3-path`

The Amazon S3 location of the bootstrap action that failed.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the root cause of the bootstrap action error. Then launch a new cluster.

1. To find the relevant bootstrap action script in Amazon S3, use the `amazon-s3-path` value from the `ErrorData` array.
2. Review the bootstrap action log files in Amazon S3 to identify the root cause for the failure. To learn more on how to view Amazon EMR logs, see View log files.

   > **Note**
   >
   > If you didn't turn on logs for your cluster, you must create a new cluster with the same configurations and bootstrap actions. To ensure the cluster logs are turned on, see Configure cluster logging and debugging.

3. Review the `stdout` log for your bootstrap actions and confirm that there are no custom processes that delete files in the `/emr/instance-controller/lib/bootstrap-actions` folder on your primary instances. You can find the `stdout` log for the bootstrap action in this Amazon S3 location:

   ```
   s3://EXAMPLE-BUCKET/logs/Your_Cluster_Id/node/Primary_Instance_Id/bootstrap-actions/Failed_Bootstrap_Action_Number/stdout.gz
   ```

4. Launch a new cluster with your updated bootstrap action.

Internal error codes

The following sections provide troubleshooting information for internal error codes.

Topics

- INTERNAL_ERROR_EC2_INSUFFICIENT_CAPACITY_AZ
- INTERNAL_ERROR_SPOT_PRICE_INCREASE_PRIMARY
- INTERNAL_ERROR_SPOT_NO_CAPACITY_PRIMARY

INTERNAL_ERROR_EC2_INSUFFICIENT_CAPACITY_AZ

Overview

A cluster terminates with an `INTERNAL_ERROR_EC2_INSUFFICIENT_CAPACITY_AZ` error when the selected Availability Zone doesn't have enough capacity to fulfill your Amazon EC2 instance type request. The subnet that you select for a cluster determines the Availability Zone. For more information about subnets for Amazon EMR, see Configure networking.

Resolution

To resolve this error, modify your instance type configurations and create a new cluster with your updated request.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`instance-type`

The instance type that is out of capacity.

`availability-zone`

The Availability Zone that your subnet resolves to.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the root cause of the cluster configuration error:

- Review the best practices for cluster configuration. See Best practices for cluster configuration in the *Amazon EMR Management Guide*.
- Troubleshoot the launch issues and review your configuration. See Troubleshoot instance launch issues in the *Amazon EC2 User Guide for Linux Instances*.
- Launch a new cluster with your updated cluster configuration.

INTERNAL_ERROR_SPOT_PRICE_INCREASE_PRIMARY

Overview

A cluster terminates with an `INTERNAL_ERROR_SPOT_PRICE_INCREASE_PRIMARY` error when Amazon EMR can't fulfill your Spot Instance request for the primary node because instances aren't available at or below your maximum Spot price. For more information, see Spot Instances in the *Amazon EC2 User Guide for Linux Instances*.

Resolution

To resolve this error, specify instance types for your cluster that are within your price target, or increase your price limit for the same instance type.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`primary-instance-id`

The ID for the primary instance of the cluster that failed.

`instance-type`

The instance type that is out of capacity.

`availability-zone`

The Availability Zone where your subnet resides.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to troubleshoot your cluster configuration strategy, and then launch a new cluster:

1. Review the best practices for Amazon EC2 Spot Instances and review your cluster configuration strategy. For more information, see Best practices for EC2 Spot in the *Amazon EC2 User Guide for Linux Instances* and Best practices for cluster configuration.

2. Modify your instance type configurations or Availability Zone and create a new cluster with your updated request.
3. If the issue persists, use On-Demand capacity for your primary instance.

INTERNAL_ERROR_SPOT_NO_CAPACITY_PRIMARY

Overview

A cluster terminates with a `INTERNAL_ERROR_SPOT_NO_CAPACITY_PRIMARY` error when there isn't enough capacity to fulfill a Spot Instance request for your primary node. For more information, see Spot Instances in the *Amazon EC2 User Guide for Linux Instances*.

Resolution

To resolve this error, specify instance types for your cluster that are within your price target, or increase your price limit for the same instance type.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`primary-instance-id`

The ID for the primary instance of the cluster that failed.

`instance-type`

The instance type that is out of capacity.

`availability-zone`

The Availability Zone that your subnet resolves to.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to troubleshoot your cluster configuration strategy, and then launch a new cluster:

1. Review the best practices for Amazon EC2 Spot Instances and review your cluster configuration strategy. For more information, see Best practices for EC2 Spot in the *Amazon EC2 User Guide for Linux Instances* and Best practices for cluster configuration.
2. Modify your instance type configurations and create a new cluster with your updated request.
3. If the issue persists, use On-Demand capacity for your primary instance.

Validation failure error codes

The following sections provide troubleshooting information for validation failure error codes.

Topics

- VALIDATION_ERROR_SUBNET_NOT_FROM_ONE_VPC
- VALIDATION_ERROR_SECURITY_GROUP_NOT_FROM_ONE_VPC
- VALIDATION_ERROR_INVALID_SSH_KEY_NAME
- VALIDATION_ERROR_INSTANCE_TYPE_NOT_SUPPORTED

VALIDATION_ERROR_SUBNET_NOT_FROM_ONE_VPC

Overview

When your cluster and the subnets that you reference for your cluster belong to different virtual private clouds (VPCs), the cluster terminates with a `VALIDATION_ERROR_SUBNET_NOT_FROM_ONE_VPC` error. You can launch clusters with Amazon EMR with the instance fleets configuration across subnets in a VPC. For more information about instance fleets, see Configure instance fleets in the *Amazon EMR Management Guide*.

Resolution

To resolve this error, use subnets that belong to the same VPC as the cluster.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

vpc

For each subnet:VPC pair, the ID for the VPC that the subnet belongs to.

`subnet`

For each subnet:VPC pair, the ID for the subnet.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the error:

1. Review the subnet IDs that are listed in the `ErrorData` array and confirm that they belong to the VPC where you want to launch the EMR cluster.
2. Modify your subnet configurations. You can use one of the following methods to find all available public and private subnets in a VPC.
   - Navigate to the Amazon VPC Console. Choose **Subnets** and list all of the subnets that reside within the AWS Region for your cluster. To find only public or private subnets, apply the **Auto-assign public IPv4 address** filter. To find and select subnets in the VPC that your cluster uses, use the **Filter by VPC** option. For more information on how to create subnets, see Create a subnet in the *Amazon Virtual Private Cloud User Guide*.
   - Use the AWS CLI to find all available public and private subnets in the VPC that your cluster uses. For more information, see the describe-subnets API. To create new subnets in a VPC, see the create-subnet API.
3. Launch a new cluster with subnets from the same VPC as the cluster.

VALIDATION_ERROR_SECURITY_GROUP_NOT_FROM_ONE_VPC

Overview

When your cluster and the security groups that you assign to your cluster belong to different virtual private clouds (VPCs), the cluster terminates with a `VALIDATION_ERROR_SECURITY_GROUP_NOT_FROM_ONE_VPC` error. For more information about security groups, see Specifying Amazon EMR-managed and additional security groups and Control network traffic with security groups.

Resolution

To resolve this error, use security groups that belong to the same VPC as the cluster.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`vpc`

For each security-group:VPC pair, the ID for the VPC that the security group belongs to.

`security-group`

For each security-group:VPC pair, the ID for the security group.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the error:

1. Review the security group IDs that are listed in the `ErrorData` array and confirm that they belong to the VPC where you want to launch the EMR cluster.
2. Navigate to the Amazon VPC Console. Choose **Security groups** to list all of the security groups within the Region that you select. Find the security groups from the same VPC as your cluster, and then modify your security group configuration.
3. Launch a new cluster with security groups from the same VPC as the cluster.

VALIDATION_ERROR_INVALID_SSH_KEY_NAME

Overview

A cluster terminates with a `VALIDATION_ERROR_INVALID_SSH_KEY_NAME` error when you use an Amazon EC2 key pair that isn't valid to SSH into the primary instance. The key pair name might be incorrect, or the key pair might not exist in the requested AWS Region. For more information about key pairs, see Amazon EC2 key pairs and Linux instances in the *Amazon EC2 User Guide for Linux Instances*.

Resolution

To resolve this error, create a new cluster with a valid SSH key pair name.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`ssh-key`

The SSH key pair name that you provided when you created the cluster.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the error:

1. Check your *`keypair`*.pem file and confirm that it matches the name of the SSH key that you see in the Amazon EMR console.
2. Navigate to the Amazon EC2 console. Verify that the SSH key name that you used is available in the AWS Region that your cluster uses. You can find your AWS Region next to your account ID at the top of the AWS Management Console.
3. Launch a new cluster with a valid SSH key name.

VALIDATION_ERROR_INSTANCE_TYPE_NOT_SUPPORTED

Overview

A cluster terminates with a `VALIDATION_ERROR_INSTANCE_TYPE_NOT_SUPPORTED` error when the AWS Region and Availability Zones for your cluster don't support the specified instance type for one or more instance groups. Amazon EMR might support an instance type in one Availability Zone within a Region but not another. The subnet that you select for a cluster determines the Availability Zone within the Region. For a list of instance types and Regions that Amazon EMR supports, see Supported instance types.

Resolution

To resolve this error, specify instance types for your cluster that Amazon EMR supports in the Region and Availability Zone where you request the cluster.

To troubleshoot the failed EMR cluster, refer to the `ErrorDetail` information returned from the `DescribeCluster` and `ListClusters` APIs. For more information, see Error codes with ErrorDetail information. The `ErrorData` array within `ErrorDetail` returns the following information for this error code:

`instance-types`

The list of unsupported instance types.

`availability-zones`

The list of Availability Zones that your subnet resolves to.

`public-doc`

The public URL of the documentation for the error code.

Steps to complete

Perform the following steps to identify and fix the error:

1. Use the AWS CLI to retrieve the available instance types in an Availability Zone. To do this, you can use the `ec2 describe-instance-type-offerings` command to filter available instance types by location (AWS Region or Availability Zone). For example, the following command returns the instance types that are offered in the specified AZ, *`us-east-2a`*.

   ```
   aws ec2 describe-instance-type-offerings --location-type "availability-
   zone" --filters Name=location,Values=us-east-2a --region us-east-2 --query
    "InstanceTypeOfferings[*].[InstanceType]" --output text | sort
   ```

   To learn more about how to discover available instance types, see Find an Amazon EC2 instance type.

2. After you determine the instance types that are available in the same Region and Availability Zone as the cluster, choose one of the following resolutions to continue:

   a. Create a new cluster, and choose a subnet for the cluster that is in an Availability Zone where the instance type that you selected is available and supported by Amazon EMR.

   b. Create a new cluster in the same Region and Amazon EC2 subnet as the cluster that failed, but with an instance type that is supported in that location by Amazon EMR.

For a list of instance types and Regions that Amazon EMR supports, see Supported instance types. To compare the capabilities of the instance types, see Amazon EC2 instance types.

Resource errors

The following errors are commonly caused by constrained resources on the cluster.

Topics

Cluster terminates with NO_SLAVE_LEFT and core nodes FAILED_BY_MASTER

Usually, this happens because termination protection is disabled, and all core nodes exceed disk storage capacity as specified by a maximum utilization threshold in the `yarn-site` configuration classification, which corresponds to the `yarn-site.xml` file. This value is 90% by default. When disk utilization for a core node exceeds the utilization threshold, the YARN NodeManager health service reports the node as `UNHEALTHY`. While it's in this state, Amazon EMR deny lists the node and does not allocate YARN containers to it. If the node remains unhealthy for 45 minutes, Amazon EMR marks the associated Amazon EC2 instance for termination as `FAILED_BY_MASTER`. When all Amazon EC2 instances associated with core nodes are marked for termination, the cluster terminates with the status `NO_SLAVE_LEFT` because there are no resources to execute jobs.

Exceeding disk utilization on one core node might lead to a chain reaction. If a single node exceeds the disk utilization threshold because of HDFS, other nodes are likely to be near the threshold as well. The first node exceeds the disk utilization threshold, so Amazon EMR deny lists it. This increases the burden of disk utilization for remaining nodes because they begin to replicate HDFS data among themselves that they lost on the deny-listed node. Each node subsequently goes `UNHEALTHY` in the same way, and the cluster eventually terminates.

Best practices and recommendations

Configure cluster hardware with adequate storage

When you create a cluster, make sure that there are enough core nodes and that each has an adequate instance store and EBS storage volumes for HDFS. For more information, see Calculating the required HDFS capacity of a cluster. You can also add core instances to existing instance groups manually or by using auto-scaling. The new instances have the same storage configuration as other instances in the instance group. For more information, see Use cluster scaling.

Enable termination protection

Enable termination protection. This way, if a core node is deny listed, you can connect to the associated Amazon EC2 instance using SSH to troubleshoot and recover data. If you enable termination protection, be aware that Amazon EMR does not replace the Amazon EC2 instance with a new instance. For more information, see Using termination protection.

Create an alarm for the MRUnhealthyNodes CloudWatch metric

This metric reports the number of nodes reporting an `UNHEALTHY` status. It's equivalent to the YARN metric `mapred.resourcemanager.NoOfUnhealthyNodes`. You can set up a notification for this alarm to warn you of unhealthy nodes before the 45-minute timeout is reached. For more information, see Monitoring Amazon EMR metrics with CloudWatch.

Tweak settings using yarn-site

The settings below can be adjusted according to your application requirements. For example, you may want to increase the disk utilization threshold where a node reports `UNHEALTHY` by increasing the value of `yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage`.

You can set these values when you create a cluster using the `yarn-site` configuration classification. For more information see Configuring applications in the *Amazon EMR Release Guide*. You can also connect to the Amazon EC2 instances associated with core nodes using SSH, and then add the values in `/etc/hadoop/conf.empty/yarn-site.xml` using a text editor. After making the change, you must restart hadoop-yarn-nodemanager as shown below.

> **Important**
>
> When you restart the NodeManager service, active YARN containers are killed unless `yarn.nodemanager.recovery.enabled` is set to `true` using the `yarn-site`

configuration classification when you create the cluster. You must also specify the directory in which to store container state using the `yarn.nodemanager.recovery.dir` property.

```
sudo /sbin/stop hadoop-yarn-nodemanager
sudo /sbin/start hadoop-yarn-nodemanager
```

For more information about current `yarn-site` properties and default values, see YARN default settings in Apache Hadoop documentation.

Property	Default value	Description
yarn.nodemanager.disk-health-checker.interval-ms	120000	The frequency (in seconds) that the disk health checker runs.
yarn.nodemanager.disk-health-checker.min-healthy-disks	0.25	The minimum fraction of the number of disks that must be healthy for NodeManager to launch new containers. This corresponds to both yarn.nodemanager.local-dirs (by default, `/mnt/yarn` in Amazon EMR) and yarn.nodemanager.log-dirs (by default `/var/log/hadoop-yarn/containers`, which is symlinked to `mnt/var/log/hadoop-yarn/containers` in Amazon EMR).
`yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage`	90.0	The maximum percentage of disk space utilization allowed after which a disk is marked as bad. Values can range from 0.0 to 100.0. If the value is greater than or

Property	Default value	Description
		equal to 100, the NodeManager checks for a full disk. This applies to `yarn-nodemanager.local-dirs` and `yarn.nodemanager.log-dirs .`
`yarn.nodemanager.disk-health-checker.min-free-space-per-disk-mb`	0	The minimum space that must be available on a disk for it to be used. This applies to `yarn-nodemanager.local-dirs` and `yarn.nodemanager.log-dirs .`

Cannot replicate block, only managed to replicate to zero nodes.

The error, "Cannot replicate block, only managed to replicate to zero nodes." typically occurs when a cluster does not have enough HDFS storage. This error occurs when you generate more data in your cluster than can be stored in HDFS. You see this error only while the cluster is running, because when the job ends it releases the HDFS space it was using.

The amount of HDFS space available to a cluster depends on the number and type of Amazon EC2 instances that are used as core nodes. Task nodes are not used for HDFS storage. All of the disk space on each Amazon EC2 instance, including attached EBS storage volumes, is available to HDFS. For more information about the amount of local storage for each EC2 instance type, see Instance types and families in the *Amazon EC2 User Guide for Linux Instances*.

The other factor that can affect the amount of HDFS space available is the replication factor, which is the number of copies of each data block that are stored in HDFS for redundancy. The replication factor increases with the number of nodes in the cluster: there are 3 copies of each data block for a cluster with 10 or more nodes, 2 copies of each block for a cluster with 4 to 9 nodes, and 1 copy (no redundancy) for clusters with 3 or fewer nodes. The total HDFS space available is divided by the replication factor. In some cases, such as increasing the number of nodes from 9 to 10, the increase in replication factor can actually cause the amount of available HDFS space to decrease.

For example, a cluster with ten core nodes of type m1.large would have 2833 GB of space available to HDFS ((10 nodes X 850 GB per node)/replication factor of 3).

If your cluster exceeds the amount of space available to HDFS, you can add additional core nodes to your cluster or use data compression to create more HDFS space. If your cluster is one that can be stopped and restarted, you may consider using core nodes of a larger Amazon EC2 instance type. You might also consider adjusting the replication factor. Be aware, though, that decreasing the replication factor reduces the redundancy of HDFS data and your cluster's ability to recover from lost or corrupted HDFS blocks.

EC2 QUOTA EXCEEDED

If you get an `EC2 QUOTA EXCEEDED` message, there may be several causes. Depending on configuration differences, it may take up to 5-20 minutes for previous clusters to terminate and release allocated resources. If you are getting an `EC2 QUOTA EXCEEDED` error when you attempt to launch a cluster, it may be because resources from a recently terminated cluster have not yet been released. This message can also be caused by the resizing of an instance group or instance fleet to a target size that is greater than the current instance quota for the account. This can happen manually or automatically through automatic scaling.

Consider the following options to resolve the issue:

- Follow the instructions in AWS service quotas in the *Amazon Web Services General Reference* to request a service limit increase. For some APIs, setting up a CloudWatch event might be a better option than increasing limits. For more details, see When to set up EMR events in CloudWatch.
- If one or more running clusters are not at capacity, resize instance groups or reduce target capacities on instance fleets for running clusters.
- Create clusters with fewer EC2 instances or reduced target capacity.

Too many fetch-failures

The presence of "**Too many fetch-failures**" or "**Error reading task output**" error messages in step or task attempt logs indicates the running task is dependent on the output of another task. This often occurs when a reduce task is queued to execute and requires the output of one or more map tasks and the output is not yet available.

There are several reasons the output may not be available:

- The prerequisite task is still processing. This is often a map task.

- The data may be unavailable due to poor network connectivity if the data is located on a different instance.
- If HDFS is used to retrieve the output, there may be an issue with HDFS.

The most common cause of this error is that the previous task is still processing. This is especially likely if the errors are occurring when the reduce tasks are first trying to run. You can check whether this is the case by reviewing the syslog log for the cluster step that is returning the error. If the syslog shows both map and reduce tasks making progress, this indicates that the reduce phase has started while there are map tasks that have not yet completed.

One thing to look for in the logs is a map progress percentage that goes to 100% and then drops back to a lower value. When the map percentage is at 100%, this does not mean that all map tasks are completed. It simply means that Hadoop is executing all the map tasks. If this value drops back below 100%, it means that a map task has failed and, depending on the configuration, Hadoop may try to reschedule the task. If the map percentage stays at 100% in the logs, look at the CloudWatch metrics, specifically `RunningMapTasks`, to check whether the map task is still processing. You can also find this information using the Hadoop web interface on the master node.

If you are seeing this issue, there are several things you can try:

- Instruct the reduce phase to wait longer before starting. You can do this by altering the Hadoop configuration setting mapred.reduce.slowstart.completed.maps to a longer time. For more information, see Create bootstrap actions to install additional software.
- Match the reducer count to the total reducer capability of the cluster. You do this by adjusting the Hadoop configuration setting mapred.reduce.tasks for the job.
- Use a combiner class code to minimize the number of outputs that need to be fetched.
- Check that there are no issues with the Amazon EC2 service that are affecting the network performance of the cluster. You can do this using the Service Health Dashboard.
- Review the CPU and memory resources of the instances in your cluster to make sure that your data processing is not overwhelming the resources of your nodes. For more information, see Configure cluster hardware and networking.
- Check the version of the Amazon Machine Image (AMI) used in your Amazon EMR cluster. If the version is 2.3.0 through 2.4.4 inclusive, update to a later version. AMI versions in the specified range use a version of Jetty that may fail to deliver output from the map phase. The fetch error occurs when the reducers cannot obtain output from the map phase.

Jetty is an open-source HTTP server that is used for machine to machine communications within a Hadoop cluster.

File could only be replicated to 0 nodes instead of 1

When a file is written to HDFS, it is replicated to multiple core nodes. When you see this error, it means that the NameNode daemon does not have any available DataNode instances to write data to in HDFS. In other words, block replication is not taking place. This error can be caused by a number of issues:

- The HDFS filesystem may have run out of space. This is the most likely cause.
- DataNode instances may not have been available when the job was run.
- DataNode instances may have been blocked from communication with the master node.
- Instances in the core instance group might not be available.
- Permissions may be missing. For example, the JobTracker daemon may not have permissions to create job tracker information.
- The reserved space setting for a DataNode instance may be insufficient. Check whether this is the case by checking the dfs.datanode.du.reserved configuration setting.

To check whether this issue is caused by HDFS running out of disk space, look at the `HDFSUtilization` metric in CloudWatch. If this value is too high, you can add additional core nodes to the cluster. If you have a cluster that you think might run out of HDFS disk space, you can set an alarm in CloudWatch to alert you when the value of `HDFSUtilization` rises above a certain level. For more information, see Manually resizing a running cluster and Monitoring Amazon EMR metrics with CloudWatch.

If HDFS running out of space was not the issue, check the DataNode logs, the NameNode logs and network connectivity for other issues that could have prevented HDFS from replicating data. For more information, see View log files.

Deny-listed nodes

The NodeManager daemon is responsible for launching and managing containers on core and task nodes. The containers are allocated to the NodeManager daemon by the ResourceManager daemon that runs on the master node. The ResourceManager monitors the NodeManager node through a heartbeat.

There are a couple of situations in which the ResourceManager daemon deny lists a NodeManager, removing it from the pool of nodes available to process tasks:

- If the NodeManager has not sent a heartbeat to the ResourceManager daemon in the past 10 minutes (600,000 milliseconds). This time period can be configured using the `yarn.nm.liveness-monitor.expiry-interval-ms` configuration setting. For more information about changing Yarn configuration settings, see Configuring applications in the *Amazon EMR Release Guide*.
- NodeManager checks the health of the disks determined by `yarn.nodemanager.local-dirs` and `yarn.nodemanager.log-dirs`. The checks include permissions and free disk space (< 90%). If a disk fails the check, the NodeManager stops using that particular disk but still reports the node status as healthy. If a number of disks fail the check, the node is reported as unhealthy to the ResourceManager and new containers are not assigned to the node.

The application master can also deny list a NodeManager node if it has more than three failed tasks. You can change this to a higher value using the `mapreduce.job.maxtaskfailures.per.tracker` configuration parameter. Other configuration settings you might change control how many times to attempt a task before marking it as failed: `mapreduce.map.max.attempts` for map tasks and `mapreduce.reduce.maxattempts` for reduce tasks. For more information about changing configuration settings, see Configuring applications in the *Amazon EMR Release Guide*.

Throttling errors

The errors "Throttled from *Amazon EC2* while launching cluster" and "Failed to provision instances due to throttling from *Amazon EC2*" occur when Amazon EMR cannot complete a request because another service has throttled the activity. Amazon EC2 is the most common source of throttling errors, but other services may be the cause of throttling errors. AWS service limits apply on a per-Region basis to improve performance, and a throttling error indicates that you have exceeded the service limit for your account in that Region.

Possible causes

The most common source of Amazon EC2 throttling errors is a large number of cluster instances launching so that your service limit for EC2 instances is exceeded. Cluster instances may launch for the following reasons:

- New clusters are created.

- Clusters are resized manually. For more information, see Manually resizing a running cluster.
- Instance groups in a cluster add instances (scale out) as a result of an automatic scaling rule. For more information, see Understanding automatic scaling rules.
- Instance fleets in a cluster add instances to meet an increased target capacity. For more information, see Configure instance fleets.

It is also possible that the frequency or type of API request being made to Amazon EC2 causes throttling errors. For more information about how Amazon EC2 throttles API requests, see Query API request rate in the *Amazon EC2 API Reference*.

Solutions

Consider the following solutions:

- Follow the instructions in AWS service quotas in the *Amazon Web Services General Reference* to request a service limit increase. For some APIs, setting up a CloudWatch event might be a better option than increasing limits. For more details, see When to set up EMR events in CloudWatch.
- If you have clusters that launch on the same schedule—for example, at the top of the hour—consider staggering start times.
- If you have clusters that are sized for peak demand, and you periodically have instance capacity, consider specifying automatic scaling to add and remove instances on-demand. In this way, instances are used more efficiently, and depending on the demand profile, fewer instances may be requested at a given time across an account. For more information, see Using automatic scaling with a custom policy for instance groups.

Instance type not supported

If you create a cluster, and it fails with the error message "The requested instance type *`InstanceType`* is not supported in the requested Availability Zone," it means that you created the cluster and specified an instance type for one or more instance groups that is not supported by Amazon EMR in the Region and Availability Zone where the cluster was created. Amazon EMR may support an instance type in one Availability Zone within a Region and not another. The subnet you select for a cluster determines the Availability Zone within the Region.

Solution

Determine available instance types in an Availability Zone using the AWS CLI

- Use the `ec2 run-instances` command with the `--dry-run` option. In the example below, replace *`m5.xlarge`* with the instance type you want to use, *`ami-035be7bafff33b6b6`* with the AMI associated with that instance type, and *`subnet-12ab3c45`* with a subnet in the Availability Zone you want to query.

  ```
  aws ec2 run-instances --instance-type m5.xlarge --dry-run --image-
  id ami-035be7bafff33b6b6 --subnet-id subnet-12ab3c45
  ```

  For instructions on finding an AMI ID, see Find a Linux AMI. To find a subnet ID, you can use the describe-subnets command.

To learn more about how to discover available instance types, see Find an Amazon EC2 instance type.

After you determine the instance types available, you can do any of the following:

- Create the cluster in the same Region and EC2 Subnet, and choose a different instance type with similar capabilities as your initial choice. For a list of supported instance types, see Supported instance types. To compare capabilities of EC2 instance types, see Amazon EC2 instance types.
- Choose a subnet for the cluster in an Availability Zone where the instance type is available and supported by Amazon EMR.

EC2 is out of capacity

An "EC2 is out of capacity for *`InstanceType`*" error occurs when you attempt to create a cluster, or add instances to a cluster, in an Availability Zone which has no more of the specified EC2 instance type. The subnet that you select for a cluster determines the Availability Zone.

To create a cluster, do one of the following:

- Specify a different instance type with similar capabilities
- Create the cluster in a different Region
- Select a subnet in an Availability Zone where the instance type you want might be available.

To add instances to a running cluster, do one of the following:

- Modify instance group configurations or instance fleet configurations to add available instance types with similar capabilities. For a list of supported instance types, see Supported instance types. To compare capabilities of EC2 instance types, see Amazon EC2 instance types.
- Terminate the cluster and recreate it in a Region and Availability Zone where the instance type is available.

Input and output errors

The following errors are common in cluster input and output operations.

Topics

Does your path to Amazon Simple Storage Service (Amazon S3) have at least three slashes?

When you specify an Amazon S3 bucket, you must include a terminating slash on the end of the URL. For example, instead of referencing a bucket as "s3n://*DOC-EXAMPLE-BUCKET1*", you should use "s3n://*DOC-EXAMPLE-BUCKET1*/", otherwise Hadoop fails your cluster in most cases.

Are you trying to recursively traverse input directories?

Hadoop does not recursively search input directories for files. If you have a directory structure such as /corpus/01/01.txt, /corpus/01/02.txt, /corpus/02/01.txt, etc. and you specify /corpus/ as the input parameter to your cluster, Hadoop does not find any input files because the /corpus/ directory is empty and Hadoop does not check the contents of the subdirectories. Similarly, Hadoop does not recursively check the subdirectories of Amazon S3 buckets.

The input files must be directly in the input directory or Amazon S3 bucket that you specify, not in subdirectories.

Does your output directory already exist?

If you specify an output path that already exists, Hadoop will fail the cluster in most cases. This means that if you run a cluster one time and then run it again with exactly the same parameters, it will likely work the first time and then never again; after the first run, the output path exists and thus causes all successive runs to fail.

Are you trying to specify a resource using an HTTP URL?

Hadoop does not accept resource locations specified using the http:// prefix. You cannot reference a resource using an HTTP URL. For example, passing in http://mysite/myjar.jar as the JAR parameter causes the cluster to fail.

Are you referencing an Amazon S3 bucket using an invalid name format?

If you attempt to use a bucket name such as "*DOC-EXAMPLE-BUCKET1*.1" with Amazon EMR, your cluster will fail because Amazon EMR requires that bucket names be valid RFC 2396 host names; the name cannot end with a number. In addition, because of the requirements of Hadoop, Amazon S3 bucket names used with Amazon EMR must contain only lowercase letters, numbers, periods (.), and hyphens (-). For more information about how to format Amazon S3 bucket names, see Bucket restrictions and limitations in the *Amazon Simple Storage Service User Guide*.

Are you experiencing trouble loading data to or from Amazon S3?

Amazon S3 is the most popular input and output source for Amazon EMR. A common mistake is to treat Amazon S3 as you would a typical file system. There are differences between Amazon S3 and a file system that you need to take into account when running your cluster.

- If an internal error occurs in Amazon S3, your application needs to handle this gracefully and re-try the operation.
- If calls to Amazon S3 take too long to return, your application may need to reduce the frequency at which it calls Amazon S3.
- Listing all the objects in an Amazon S3 bucket is an expensive call. Your application should minimize the number of times it does this.

There are several ways you can improve how your cluster interacts with Amazon S3.

- Launch your cluster using the most recent release version of Amazon EMR.

- Use S3DistCp to move objects in and out of Amazon S3. S3DistCp implements error handling, retries and back-offs to match the requirements of Amazon S3. For more information, see Distributed copy using S3DistCp.
- Design your application with eventual consistency in mind. Use HDFS for intermediate data storage while the cluster is running and Amazon S3 only to input the initial data and output the final results.
- If your clusters will commit 200 or more transactions per second to Amazon S3, contact support to prepare your bucket for greater transactions per second and consider using the key partition strategies described in Amazon S3 performance tips & tricks.
- Set the Hadoop configuration setting io.file.buffer.size to 65536. This causes Hadoop to spend less time seeking through Amazon S3 objects.
- Consider disabling Hadoop's speculative execution feature if your cluster is experiencing Amazon S3 concurrency issues. This is also useful when you are troubleshooting a slow cluster. You do this by setting the `mapreduce.map.speculative` and `mapreduce.reduce.speculative` properties to `false`. When you launch a cluster, you can set these values using the `mapred-env` configuration classification. For more information, see Configuring Applications in the *Amazon EMR Release Guide*.
- If you are running a Hive cluster, see Are you having trouble loading data to or from Amazon S3 into Hive?.

For additional information, see Amazon S3 error best practices in the *Amazon Simple Storage Service User Guide*.

Permissions errors

The following errors are common when using permissions or credentials.

Topics

- Are you passing the correct credentials into SSH?
- If you are using IAM, do you have the proper Amazon EC2 policies set?

Are you passing the correct credentials into SSH?

If you are unable to use SSH to connect to the master node, it is most likely an issue with your security credentials.

First, check that the .pem file containing your SSH key has the proper permissions. You can use chmod to change the permissions on your .pem file as is shown in the following example, where you would replace mykey.pem with the name of your own .pem file.

```
chmod og-rwx mykey.pem
```

The second possibility is that you are not using the keypair you specified when you created the cluster. This is easy to do if you have created multiple key pairs. Check the cluster details in the Amazon EMR console (or use the `--describe` option in the CLI) for the name of the keypair that was specified when the cluster was created.

After you have verified that you are using the correct key pair and that permissions are set correctly on the .pem file, you can use the following command to use SSH to connect to the master node, where you would replace mykey.pem with the name of your .pem file and `hadoop@ec2-01-001-001-1.compute-1.amazonaws.com` with the public DNS name of the master node (available through the `--describe` option in the CLI or through the Amazon EMR console.)

> **Important**
>
> You must use the login name `hadoop` when you connect to an Amazon EMR cluster node, otherwise an error similar to `Server refused our key` error may occur.

```
ssh -i mykey.pem hadoop@ec2-01-001-001-1.compute-1.amazonaws.com
```

For more information, see Connect to the primary node using SSH.

If you are using IAM, do you have the proper Amazon EC2 policies set?

Because Amazon EMR uses EC2 instances as nodes, a users of Amazon EMR also need to have certain Amazon EC2 policies set in order for Amazon EMR to be able to manage those instances on the a user's behalf. If you do not have the required permissions set, Amazon EMR returns the error: **"account is not authorized to call EC2."**

For more information about the Amazon EC2 policies your IAM account needs to set to run Amazon EMR, see How Amazon EMR works with IAM.

Hive cluster errors

You can usually find the cause of a Hive error in the `syslog` file, which you link to from the **Steps** pane. If you can't determine the problem there, check in the Hadoop task attempt error message. Link to it on the **Task Attempts** pane.

The following errors are common to Hive clusters.

Topics

Are you using the latest version of Hive?

The latest version of Hive has all the current patches and bug fixes and may resolve your issue.

Did you encounter a syntax error in the Hive script?

If a step fails, look at the `stdout` file of the logs for the step that ran the Hive script. If the error is not there, look at the `syslog` file of the task attempt logs for the task attempt that failed. For more information, see View log files.

Did a job fail when running interactively?

If you are running Hive interactively on the master node and the cluster failed, look at the `syslog` entries in the task attempt log for the failed task attempt. For more information, see View log files.

Are you having trouble loading data to or from Amazon S3 into Hive?

If you are having trouble accessing data in Amazon S3, first check the possible causes listed in Are you experiencing trouble loading data to or from Amazon S3?. If none of those issues is the cause, consider the following options specific to Hive.

- Make sure you are using the latest version of Hive, which has all the current patches and bug fixes that may resolve your issue. For more information, see Apache Hive.

- Using `INSERT OVERWRITE` requires listing the contents of the Amazon S3 bucket or folder. This is an expensive operation. If possible, manually prune the path instead of having Hive list and delete the existing objects.
- If you use Amazon EMR release versions earlier than 5.0, you can use the following command in HiveQL to pre-cache the results of an Amazon S3 list operation locally on the cluster:

```
set hive.optimize.s3.query=true;
```

- Use static partitions where possible.
- In some versions of Hive and Amazon EMR, it is possible that using ALTER TABLES will fail because the table is stored in a different location than expected by Hive. The solution is to add or update following in `/home/hadoop/conf/core-site.xml`:

```
<property>
    <name>fs.s3n.endpoint</name>
    <value>s3.amazonaws.com</value>
</property>
```

VPC errors

The following errors are common to VPC configuration in Amazon EMR.

Topics

Invalid subnet configuration

On the **Cluster Details** page, in the **Status** field, you see an error similar to the following:

`The subnet configuration was invalid: Cannot find route to InternetGateway in main RouteTable` *`rtb-id`* `for vpc` *`vpc-id`*.

To solve this problem, you must create an Internet Gateway and attach it to your VPC. For more information, see Adding an internet gateway to your VPC.

Alternatively, verify that you have configured your VPC with **Enable DNS resolution** and **Enable DNS hostname support** enabled. For more information, see Using DNS with your VPC.

Missing DHCP Options Set

You see a step failure in the cluster system log (syslog) with an error similar to the following:

```
ERROR org.apache.hadoop.security.UserGroupInformation
(main): PriviledgedActionException as:hadoop (auth:SIMPLE)
cause:java.io.IOException:
org.apache.hadoop.yarn.exceptions.ApplicationNotFoundException: Application
with id 'application_id' doesn't exist in RM.
```

or

```
ERROR org.apache.hadoop.streaming.StreamJob (main): Error Launching job :
org.apache.hadoop.yarn.exceptions.ApplicationNotFoundException: Application
with id 'application_id' doesn't exist in RM.
```

To solve this problem, you must configure a VPC that includes a DHCP Options Set whose parameters are set to the following values:

> **Note**
>
> If you use the AWS GovCloud (US-West) region, set domain-name to **`us-gov-west-1.compute.internal`** instead of the value used in the following example.

- **domain-name** = **`ec2.internal`**

  Use **`ec2.internal`** if your region is US East (N. Virginia). For other regions, use *`region-name`*`.compute.internal`. For example in us-west-2, use **domain-name**=**`us-west-2.compute.internal`**.
- **domain-name-servers** = **`AmazonProvidedDNS`**

For more information, see DHCP Options Sets.

Permissions errors

A failure in the `stderr` log for a step indicates that an Amazon S3 resource does not have the appropriate permissions. This is a 403 error and the error looks like:

```
Exception in thread "main" com.amazonaws.services.s3.model.AmazonS3Exception: Access
 Denied (Service: Amazon S3; Status Code: 403; Error Code: AccessDenied; Request
 ID: REQUEST_ID
```

If the ActionOnFailure is set to `TERMINATE_JOB_FLOW`, then this would result in the cluster terminating with the state, `SHUTDOWN_COMPLETED_WITH_ERRORS`.

A few ways to troubleshoot this problem include:

- If you are using an Amazon S3 bucket policy within a VPC, make sure to give access to all buckets by creating a VPC endpoint and selecting **Allow all** under the Policy option when creating the endpoint.
- Make sure that any policies associated with S3 resources include the VPC in which you launch the cluster.
- Try running the following command from your cluster to verify you can access the bucket

  ```
  hadoop fs -copyToLocal s3://path-to-bucket /tmp/
  ```

- You can get more specific debugging information by setting the `log4j.logger.org.apache.http.wire` parameter to `DEBUG` in `/home/hadoop/conf/log4j.properties` file on the cluster. You can check the `stderr` log file after trying to access the bucket from the cluster. The log file will provide more detailed information:

  ```
  Access denied for getting the prefix for bucket - us-west-2.elasticmapreduce with
   path samples/wordcount/input/
  15/03/25 23:46:20 DEBUG http.wire: >> "GET /?prefix=samples%2Fwordcount%2Finput
  %2F&delimiter=%2F&max-keys=1 HTTP/1.1[\r][\n]"
  15/03/25 23:46:20 DEBUG http.wire: >> "Host: us-
  west-2.elasticmapreduce.s3.amazonaws.com[\r][\n]"
  ```

Errors that result in `START_FAILED`

Before AMI 3.7.0, for VPCs where a hostname is specified, Amazon EMR maps the internal hostnames of the subnet with custom domain addresses as follows:

ip-*X.X.X.X.customdomain.com*.tld. For example, if the hostname was ip-10.0.0.10 and the VPC has the domain name option set to customdomain.com, the resulting hostname mapped by Amazon EMR would be ip-10.0.1.0.customdomain.com. An entry is added in /etc/hosts to resolve the hostname to 10.0.0.10. This behavior is changed with AMI 3.7.0 and now Amazon EMR honors the DHCP configuration of the VPC entirely. Previously, customers could also use a bootstrap action to specify a hostname mapping.

If you would like to preserve this behavior, you must provide the DNS and forward resolution setup you require for the custom domain.

Cluster `Terminated with errors` and NameNode fails to start

When launching an EMR cluster in a VPC which makes use of a custom DNS domain name, your cluster may fail with the following error message in the console:

```
Terminated with errors  On the master instance(instance-id), bootstrap action 1
 returned a  non-zero return code
```

The failure is a result of the NameNode not being able to start up. This will result in the following error found in the NameNode logs, whose Amazon S3 URI is of the form: s3://*mybucket*/*logs*/*cluster-id*/daemons/*master instance-id*/hadoop-hadoop-namenode-*master node hostname*.log.gz:

```
2015-07-23 20:17:06,266 WARN
      org.apache.hadoop.hdfs.server.namenode.FSNamesystem (main): Encountered
 exception
      loading fsimage  java.io.IOException: NameNode is not formatted.
      at

 org.apache.hadoop.hdfs.server.namenode.FSImage.recoverTransitionRead(FSImage.java:212)
           at

 org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFSImage(FSNamesystem.java:1020)
           at

 org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFromDisk(FSNamesystem.java:739)
           at
      org.apache.hadoop.hdfs.server.namenode.NameNode.loadNamesystem(NameNode.java:537)
           at
      org.apache.hadoop.hdfs.server.namenode.NameNode.initialize(NameNode.java:596)
```

```
      at  org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:765)
           at
      org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:749)
      at

org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1441)
           at
      org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1507)
```

This is due to a potential issue where an EC2 instance can have multiple sets of fully qualified domain names when launching EMR clusters in a VPC, which makes use of both an AWS-provided DNS server and a custom user-provided DNS server. If the user-provided DNS server does not provide any pointer (PTR) records for any A records used to designate nodes in an EMR cluster, clusters will fail starting up when configured in this way. The solution is to add 1 PTR record for every A record that is created when an EC2 instance is launched in any of the subnets in the VPC.

Streaming cluster errors

You can usually find the cause of a streaming error in a `syslog` file. Link to it on the **Steps** pane.

The following errors are common to streaming clusters.

Topics

- Is data being sent to the mapper in the wrong format?
- Is your script timing out?
- Are you passing in invalid streaming arguments?
- Did your script exit with an error?

Is data being sent to the mapper in the wrong format?

To check if this is the case, look for an error message in the `syslog` file of a failed task attempt in the task attempt logs. For more information, see View log files.

Is your script timing out?

The default timeout for a mapper or reducer script is 600 seconds. If your script takes longer than this, the task attempt will fail. You can verify this is the case by checking the `syslog` file of a failed task attempt in the task attempt logs. For more information, see View log files.

You can change the time limit by setting a new value for the `mapred.task.timeout` configuration setting. This setting specifies the number of milliseconds after which Amazon EMR will terminate a task that has not read input, written output, or updated its status string. You can update this value by passing an additional streaming argument `-jobconf mapred.task.timeout=800000`.

Are you passing in invalid streaming arguments?

Hadoop streaming supports only the following arguments. If you pass in arguments other than those listed below, the cluster will fail.

```
-blockAutoGenerateCacheFiles
-cacheArchive
-cacheFile
-cmdenv
-combiner
-debug
-input
-inputformat
-inputreader
-jobconf
-mapper
-numReduceTasks
-output
-outputformat
-partitioner
-reducer
-verbose
```

In addition, Hadoop streaming only recognizes arguments passed in using Java syntax; that is, preceded by a single hyphen. If you pass in arguments preceded by a double hyphen, the cluster will fail.

Did your script exit with an error?

If your mapper or reducer script exits with an error, you can locate the error in the `stderr` file of task attempt logs of the failed task attempt. For more information, see View log files.

Custom JAR cluster errors

The following errors are common to custom JAR clusters.

Topics

Is your JAR throwing an exception before creating a job?

If the main program of your custom JAR throws an exception while creating the Hadoop job, the best place to look is the `syslog` file of the step logs. For more information, see View log files.

Is your JAR throwing an error inside a map task?

If your custom JAR and mapper throw an exception while processing input data, the best place to look is the `syslog` file of the task attempt logs. For more information, see View log files.

AWS GovCloud (US-West) errors

The AWS GovCloud (US-West) region differs from other regions in its security, configuration, and default settings. As a result, use the following checklist to troubleshoot Amazon EMR errors that are specific to the AWS GovCloud (US-West) region before using more general troubleshooting recommendations.

- Verify that your IAM roles are correctly configured. For more information, see Configure IAM service roles for Amazon EMR permissions to AWS services and resources.
- Ensure that your VPC configuration has correctly configured DNS resolution/hostname support, Internet Gateway, and DHCP Option Set parameters. For more information, see VPC errors.

If these steps do not solve the problem, continue with the steps for troubleshooting common Amazon EMR errors. For more information, see Common errors in Amazon EMR.

Find a missing cluster

If your cluster is missing from the console list or `ListClusters` API, check the following:

- Confirm that the cluster age from time of completion is less than two months. Amazon EMR preserves metadata information for completed clusters for two months at no charge. You can't

delete completed clusters from the console — instead, Amazon EMR purges completed clusters automatically after two months.

- Confirm that you have role permissions to view the cluster.
- Confirm that you are viewing the same AWS Region where the cluster resides.

Troubleshoot a failed cluster

This section walks you through the process of troubleshooting a cluster that has failed. This means that the cluster terminated with an error code.

> **Note**
>
> When an EMR cluster terminates with an error, the `DescribeCluster` and `ListClusters` APIs return an error code and an error message. For some cluster errors, the `ErrorDetail` data array can also help you troubleshoot the failure. For more information, see Error codes with ErrorDetail information.

If your cluster runs but takes a long time to return results, see Troubleshoot a slow cluster.

Topics

Step 1: Gather data about the issue

The first step in troubleshooting a cluster is to gather information about what went wrong and the current status and configuration of the cluster. This information will be used in the following steps to confirm or rule out possible causes of the issue.

Define the problem

A clear definition of the problem is the first place to begin. Some questions to ask yourself:

- What did I expect to happen? What happened instead?
- When did this problem first occur? How often has it happened since?
- Has anything changed in how I configure or run my cluster?

Cluster details

The following cluster details are useful in helping track down issues. For more information on how to gather this information, see View cluster status and details.

- Identifier of the cluster. (Also called a job flow identifier.)
- AWS Region and Availability Zone the cluster was launched into.
- State of the cluster, including details of the last state change.
- Type and number of EC2 instances specified for the master, core, and task nodes.

Step 2: Check the environment

Amazon EMR operates as part of an ecosystem of web services and open-source software. Things that affect those dependencies can impact the performance of Amazon EMR.

Topics

Check for service outages

Amazon EMR uses several Amazon Web Services internally. It runs virtual servers on Amazon EC2, stores data and scripts on Amazon S3, and reports metrics to CloudWatch. Events that disrupt these services are rare — but when they occur — can cause issues in Amazon EMR.

Before you go further, check the Service Health Dashboard. Check the Region where you launched your cluster to see whether there are disruption events in any of these services.

Check usage limits

If you are launching a large cluster, have launched many clusters simultaneously, or you are a user sharing an AWS account with other users, the cluster may have failed because you exceeded an AWS service limit.

Amazon EC2 limits the number of virtual server instances running on a single AWS Region to 20 on-demand or reserved instances. If you launch a cluster with more than 20 nodes, or launch a cluster that causes the total number of EC2 instances active on your AWS account to exceed 20, the cluster will not be able to launch all of the EC2 instances it requires and may fail. When this happens, Amazon EMR returns an `EC2 QUOTA EXCEEDED` error. You can request that AWS increase the number of EC2 instances that you can run on your account by submitting a Request to Increase Amazon EC2 Instance Limit application.

Another thing that may cause you to exceed your usage limits is the delay between when a cluster is terminated and when it releases all of its resources. Depending on its configuration, it may take up to 5-20 minutes for a cluster to fully terminate and release allocated resources. If you are getting an `EC2 QUOTA EXCEEDED` error when you attempt to launch a cluster, it may be because resources from a recently terminated cluster may not yet have been released. In this case, you can either request that your Amazon EC2 quota be increased, or you can wait twenty minutes and re-launch the cluster.

Amazon S3 limits the number of buckets created on an account to 100. If your cluster creates a new bucket that exceeds this limit, the bucket creation will fail and may cause the cluster to fail.

Check the release version

Compare the release label that you used to launch the cluster with the latest Amazon EMR release. Each release of Amazon EMR includes improvements such as new applications, features, patches, and bug fixes. The issue that is affecting your cluster may have already been fixed in the latest release version. If possible, re-run your cluster using the latest version.

Check the Amazon VPC subnet configuration

If your cluster was launched in a Amazon VPC subnet, the subnet needs to be configured as described in Configure networking. In addition, check that the subnet you launch the cluster into has enough free elastic IP addresses to assign one to each node in the cluster.

Step 3: Look at the last state change

The last state change provides information about what occurred the last time the cluster changed state. This often has information that can tell you what went wrong as a cluster changes state to `FAILED`. For example, if you launch a streaming cluster and specify an output location that already exists in Amazon S3, the cluster will fail with a last state change of "Streaming output directory already exists".

You can locate the last state change value from the console by viewing the details pane for the cluster, from the CLI using the `list-steps` or `describe-cluster` arguments, or from the API using the `DescribeCluster` and `ListSteps` actions. For more information, see View cluster status and details.

Step 4: Examine the log files

The next step is to examine the log files in order to locate an error code or other indication of the issue that your cluster experienced. For information on the log files available, where to find them, and how to view them, see View log files.

It may take some investigative work to determine what happened. Hadoop runs the work of the jobs in task attempts on various nodes in the cluster. Amazon EMR can initiate speculative task attempts, terminating the other task attempts that do not complete first. This generates significant activity that is logged to the controller, stderr and syslog log files as it happens. In addition, multiple tasks attempts are running simultaneously, but a log file can only display results linearly.

Start by checking the bootstrap action logs for errors or unexpected configuration changes during the launch of the cluster. From there, look in the step logs to identify Hadoop jobs launched as part of a step with errors. Examine the Hadoop job logs to identify the failed task attempts. The task attempt log will contain details about what caused a task attempt to fail.

The following sections describe how to use the various log files to identify error in your cluster.

Check the bootstrap action logs

Bootstrap actions run scripts on the cluster as it is launched. They are commonly used to install additional software on the cluster or to alter configuration settings from the default values. Checking these logs may provide insight into errors that occurred during set up of the cluster as well as configuration settings changes that could affect performance.

Check the step logs

There are four types of step logs.

- **controller—**Contains files generated by Amazon EMR (Amazon EMR) that arise from errors encountered while trying to run your step. If your step fails while loading, you can find the stack trace in this log. Errors loading or accessing your application are often described here, as are missing mapper file errors.
- **stderr—**Contains error messages that occurred while processing the step. Application loading errors are often described here. This log sometimes contains a stack trace.
- **stdout—**Contains status generated by your mapper and reducer executables. Application loading errors are often described here. This log sometimes contains application error messages.
- **syslog—**Contains logs from non-Amazon software, such as Apache and Hadoop. Streaming errors are often described here.

Check stderr for obvious errors. If stderr displays a short list of errors, the step came to a quick stop with an error thrown. This is most often caused by an error in the mapper and reducer applications being run in the cluster.

Examine the last lines of controller and syslog for notices of errors or failures. Follow any notices about failed tasks, particularly if it says "Job Failed".

Check the task attempt logs

If the previous analysis of the step logs turned up one or more failed tasks, investigate the logs of the corresponding task attempts for more detailed error information.

Step 5: Test the cluster step by step

A useful technique when you are trying to track down the source of an error is to restart the cluster and submit the steps to it one by one. This lets you check the results of each step before processing the next one, and gives you the opportunity to correct and re-run a step that has failed. This also has the advantage that you only load your input data once.

To test a cluster step by step

1. Launch a new cluster, with both keep alive and termination protection enabled. Keep alive keeps the cluster running after it has processed all of its pending steps. Termination protection

prevents a cluster from shutting down in the event of an error. For more information, see Configuring a cluster to continue or terminate after step execution and Using termination protection.

2. Submit a step to the cluster. For more information, see Submit work to a cluster.
3. When the step completes processing, check for errors in the step log files. For more information, see Step 4: Examine the log files. The fastest way to locate these log files is by connecting to the master node and viewing the log files there. The step log files do not appear until the step runs for some time, finishes, or fails.
4. If the step succeeded without error, run the next step. If there were errors, investigate the error in the log files. If it was an error in your code, make the correction and re-run the step. Continue until all steps run without error.
5. When you are done debugging the cluster, and want to terminate it, you will have to manually terminate it. This is necessary because the cluster was launched with termination protection enabled. For more information, see Using termination protection.

Troubleshoot a slow cluster

This section walks you through the process of troubleshooting a cluster that is still running, but is taking a long time to return results. For more information about what to do if the cluster has terminated with an error code, see Troubleshoot a failed cluster

Amazon EMR enables you to specify the number and kind of instances in the cluster. These specifications are the primary means of affecting the speed with which your data processing completes. One thing you might consider is re-running the cluster, this time specifying EC2 instances with greater resources, or specifying a larger number of instances in the cluster. For more information, see Configure cluster hardware and networking.

The following topics walk you through the process of identifying alternative causes of a slow cluster.

Topics

Step 1: Gather data about the issue

The first step in troubleshooting a cluster is to gather information about what went wrong and the current status and configuration of the cluster. This information will be used in the following steps to confirm or rule out possible causes of the issue.

Define the problem

A clear definition of the problem is the first place to begin. Some questions to ask yourself:

- What did I expect to happen? What happened instead?
- When did this problem first occur? How often has it happened since?
- Has anything changed in how I configure or run my cluster?

Cluster details

The following cluster details are useful in helping track down issues. For more information on how to gather this information, see View cluster status and details.

- Identifier of the cluster. (Also called a job flow identifier.)
- AWS Region and Availability Zone the cluster was launched into.
- State of the cluster, including details of the last state change.
- Type and number of EC2 instances specified for the master, core, and task nodes.

Step 2: Check the environment

Topics

Check for service outages

Amazon EMR uses several Amazon Web Services internally. It runs virtual servers on Amazon EC2, stores data and scripts on Amazon S3, and reports metrics to CloudWatch. Events that disrupt these services are rare — but when they occur — can cause issues in Amazon EMR.

Before you go further, check the Service Health Dashboard. Check the Region where you launched your cluster to see whether there are disruption events in any of these services.

Check usage limits

If you are launching a large cluster, have launched many clusters simultaneously, or you are a user sharing an AWS account with other users, the cluster may have failed because you exceeded an AWS service limit.

Amazon EC2 limits the number of virtual server instances running on a single AWS Region to 20 on-demand or reserved instances. If you launch a cluster with more than 20 nodes, or launch a cluster that causes the total number of EC2 instances active on your AWS account to exceed 20, the cluster will not be able to launch all of the EC2 instances it requires and may fail. When this happens, Amazon EMR returns an `EC2 QUOTA EXCEEDED` error. You can request that AWS increase the number of EC2 instances that you can run on your account by submitting a Request to Increase Amazon EC2 Instance Limit application.

Another thing that may cause you to exceed your usage limits is the delay between when a cluster is terminated and when it releases all of its resources. Depending on its configuration, it may take up to 5-20 minutes for a cluster to fully terminate and release allocated resources. If you are getting an `EC2 QUOTA EXCEEDED` error when you attempt to launch a cluster, it may be because resources from a recently terminated cluster may not yet have been released. In this case, you can either request that your Amazon EC2 quota be increased, or you can wait twenty minutes and re-launch the cluster.

Amazon S3 limits the number of buckets created on an account to 100. If your cluster creates a new bucket that exceeds this limit, the bucket creation will fail and may cause the cluster to fail.

Check the Amazon VPC subnet configuration

If your cluster was launched in a Amazon VPC subnet, the subnet needs to be configured as described in Configure networking. In addition, check that the subnet you launch the cluster into has enough free elastic IP addresses to assign one to each node in the cluster.

Restart the cluster

The slow down in processing may be caused by a transient condition. Consider terminating and restarting the cluster to see if performance improves.

Step 3: Examine the log files

The next step is to examine the log files in order to locate an error code or other indication of the issue that your cluster experienced. For information on the log files available, where to find them, and how to view them, see View log files.

It may take some investigative work to determine what happened. Hadoop runs the work of the jobs in task attempts on various nodes in the cluster. Amazon EMR can initiate speculative task attempts, terminating the other task attempts that do not complete first. This generates significant activity that is logged to the controller, stderr and syslog log files as it happens. In addition, multiple tasks attempts are running simultaneously, but a log file can only display results linearly.

Start by checking the bootstrap action logs for errors or unexpected configuration changes during the launch of the cluster. From there, look in the step logs to identify Hadoop jobs launched as part of a step with errors. Examine the Hadoop job logs to identify the failed task attempts. The task attempt log will contain details about what caused a task attempt to fail.

The following sections describe how to use the various log files to identify error in your cluster.

Check the bootstrap action logs

Bootstrap actions run scripts on the cluster as it is launched. They are commonly used to install additional software on the cluster or to alter configuration settings from the default values. Checking these logs may provide insight into errors that occurred during set up of the cluster as well as configuration settings changes that could affect performance.

Check the step logs

There are four types of step logs.

- **controller—**Contains files generated by Amazon EMR (Amazon EMR) that arise from errors encountered while trying to run your step. If your step fails while loading, you can find the stack trace in this log. Errors loading or accessing your application are often described here, as are missing mapper file errors.

- **stderr—**Contains error messages that occurred while processing the step. Application loading errors are often described here. This log sometimes contains a stack trace.
- **stdout—**Contains status generated by your mapper and reducer executables. Application loading errors are often described here. This log sometimes contains application error messages.
- **syslog—**Contains logs from non-Amazon software, such as Apache and Hadoop. Streaming errors are often described here.

Check stderr for obvious errors. If stderr displays a short list of errors, the step came to a quick stop with an error thrown. This is most often caused by an error in the mapper and reducer applications being run in the cluster.

Examine the last lines of controller and syslog for notices of errors or failures. Follow any notices about failed tasks, particularly if it says "Job Failed".

Check the task attempt logs

If the previous analysis of the step logs turned up one or more failed tasks, investigate the logs of the corresponding task attempts for more detailed error information.

Check the Hadoop daemon logs

In rare cases, Hadoop itself might fail. To see if that is the case, you must look at the Hadoop logs. They are located at `/var/log/hadoop/` on each node.

You can use the JobTracker logs to map a failed task attempt to the node it was run on. Once you know the node associated with the task attempt, you can check the health of the EC2 instance hosting that node to see if there were any issues such as running out of CPU or memory.

Step 4: Check cluster and instance health

An Amazon EMR cluster is made up of nodes running on Amazon EC2 instances. If those instances become resource-bound (such as running out of CPU or memory), experience network connectivity issues, or are terminated, the speed of cluster processing suffers.

There are up to three types of nodes in a cluster:

- **master node** — manages the cluster. If it experiences a performance issue, the entire cluster is affected.

- **core nodes** — process map-reduce tasks and maintain the Hadoop Distributed Filesystem (HDFS). If one of these nodes experiences a performance issue, it can slow down HDFS operations as well as map-reduce processing. You can add additional core nodes to a cluster to improve performance, but cannot remove core nodes. For more information, see Manually resizing a running cluster.
- **task nodes** — process map-reduce tasks. These are purely computational resources and do not store data. You can add task nodes to a cluster to speed up performance, or remove task nodes that are not needed. For more information, see Manually resizing a running cluster.

When you look at the health of a cluster, you should look at both the performance of the cluster overall, as well as the performance of individual instances. There are several tools you can use:

Check cluster health with CloudWatch

Every Amazon EMR cluster reports metrics to CloudWatch. These metrics provide summary performance information about the cluster, such as the total load, HDFS utilization, running tasks, remaining tasks, corrupt blocks, and more. Looking at the CloudWatch metrics gives you the big picture about what is going on with your cluster and can provide insight into what is causing the slow down in processing. In addition to using CloudWatch to analyze an existing performance issue, you can set alarms that cause CloudWatch to alert if a future performance issue occurs. For more information, see Monitoring Amazon EMR metrics with CloudWatch.

Check job status and HDFS health

Use the **Application user interfaces** tab on the cluster details page to view YARN application details. For certain applications, you can drill into further detail and access logs directly. This is particularly useful for Spark applications. For more information, see View application history.

Hadoop provides a series of web interfaces you can use to view information. For more information about how to access these web interfaces, see View web interfaces hosted on Amazon EMR clusters.

- JobTracker — provides information about the progress of job being processed by the cluster. You can use this interface to identify when a job has become stuck.
- HDFS NameNode — provides information about the percentage of HDFS utilization and available space on each node. You can use this interface to identify when HDFS is becoming resource bound and requires additional capacity.

- TaskTracker — provides information about the tasks of the job being processed by the cluster. You can use this interface to identify when a task has become stuck.

Check instance health with Amazon EC2

Another way to look for information about the status of the instances in your cluster is to use the Amazon EC2 console. Because each node in the cluster runs on an EC2 instance, you can use tools provided by Amazon EC2 to check their status. For more information, see View cluster instances in Amazon EC2.

Step 5: Check for suspended groups

An instance group becomes suspended when it encounters too many errors while trying to launch nodes. For example, if new nodes repeatedly fail while performing bootstrap actions, the instance group will — after some time — go into the `SUSPENDED` state rather than continuously attempt to provision new nodes.

A node could fail to come up if:

- Hadoop or the cluster is somehow broken and does not accept a new node into the cluster
- A bootstrap action fails on the new node
- The node is not functioning correctly and fails to check in with Hadoop

If an instance group is in the `SUSPENDED` state, and the cluster is in a `WAITING` state, you can add a cluster step to reset the desired number of core and task nodes. Adding the step resumes processing of the cluster and put the instance group back into a `RUNNING` state.

For more information about how to reset a cluster in a suspended state, see Suspended state.

Step 6: Review configuration settings

Configuration settings specify details about how a cluster runs, such as how many times to retry a task and how much memory is available for sorting. When you launch a cluster using Amazon EMR, there are Amazon EMR-specific settings in addition to the standard Hadoop configuration settings. The configuration settings are stored on the master node of the cluster. You can check the configuration settings to ensure that your cluster has the resources it requires to run efficiently.

Amazon EMR defines default Hadoop configuration settings that it uses to launch a cluster. The values are based on the AMI and the instance type you specify for the cluster. You can modify

the configuration settings from the default values using a bootstrap action or by specifying new values in job execution parameters. For more information, see Create bootstrap actions to install additional software. To determine whether a bootstrap action changed the configuration settings, check the bootstrap action logs.

Amazon EMR logs the Hadoop settings used to execute each job. The log data is stored in a file named `job_`*`job-id`*`_conf.xml` under the `/mnt/var/log/hadoop/history/` directory of the master node, where *`job-id`* is replaced by the identifier of the job. If you've enabled log archiving, this data is copied to Amazon S3 in the `logs/`*`date`*`/`*`jobflow-id`*`/jobs` folder, where *`date`* is the date the job ran, and *`jobflow-id`* is the identifier of the cluster.

The following Hadoop job configuration settings are especially useful for investigating performance issues. For more information about the Hadoop configuration settings and how they affect the behavior of Hadoop, go to http://hadoop.apache.org/docs/.

Warning

1. Setting `dfs.replication` to 1 on clusters with fewer than four nodes can lead to HDFS data loss if a single node goes down. We recommend you use a cluster with at least four core nodes for production workloads.
2. Amazon EMR will not allow clusters to scale core nodes below `dfs.replication`. For example, if `dfs.replication = 2`, the minimum number of core nodes is 2.
3. When you use Managed Scaling, Auto-scaling, or choose to manually resize your cluster, we recommend that you to set `dfs.replication` to 2 or higher.

Configuration setting	Description
dfs.replication	The number of HDFS nodes to which a single block (like the hard drive block) is copied to in order to produce a RAID-like environment. Determines the number of HDFS nodes which contain a copy of the block.
io.sort.mb	Total memory available for sorting. This value should be 10x io.sort.factor. This setting can also be used to calculate total memory used by task node by figuring

Configuration setting	Description
	io.sort.mb multiplied by mapred.tasktracker.ap.tasks .maximum.
io.sort.spill.percent	Used during sort, at which point the disk will start to be used because the allotted sort memory is getting full.
mapred.child.java.opts	Deprecated. Use mapred.map.child.java.opts and mapred.reduce.child.java.opts instead. The Java options TaskTracker uses when launching a JVM for a task to execute within. A common parameter is "-Xmx" for setting max memory size.
mapred.map.child.java.opts	The Java options TaskTracker uses when launching a JVM for a map task to execute within. A common parameter is "-Xmx" for setting max memory heap size.
mapred.map.tasks.speculativ e.execution	Determines whether map task attempts of the same task may be launched in parallel.
mapred.reduce.tasks.specula tive.execution	Determines whether reduce task attempts of the same task may be launched in parallel.
mapred.map.max.attempts	The maximum number of times a map task can be attempted. If all fail, then the map task is marked as failed.
mapred.reduce.child.java.opts	The Java options TaskTracker uses when launching a JVM for a reduce task to execute within. A common parameter is "-Xmx" for setting max memory heap size.
mapred.reduce.max.attempts	The maximum number of times a reduce task can be attempted. If all fail, then the map task is marked as failed.
mapred.reduce.slowstart.com pleted.maps	The amount of maps tasks that should complete before reduce tasks are attempted. Not waiting long enough may cause "Too many fetch-failure" errors in attempts.

Configuration setting	Description
mapred.reuse.jvm.num.tasks	A task runs within a single JVM. Specifies how many tasks may reuse the same JVM.
mapred.tasktracker.map.tasks.maximum	The max amount of tasks that can execute in parallel per task node during mapping.
mapred.tasktracker.reduce.tasks.maximum	The max amount of tasks that can execute in parallel per task node during reducing.

If your cluster tasks are memory-intensive, you can enhance performance by using fewer tasks per core node and reducing your job tracker heap size.

Step 7: Examine input data

Look at your input data. Is it distributed evenly among your key values? If your data is heavily skewed towards one or few key values, the processing load may be mapped to a small number of nodes, while other nodes idle. This imbalanced distribution of work can result in slower processing times.

An example of an imbalanced data set would be running a cluster to alphabetize words, but having a data set that contained only words beginning with the letter "a". When the work was mapped out, the node processing values beginning with "a" would be overwhelmed, while nodes processing words beginning with other letters would go idle.

Troubleshoot a Lake Formation cluster

This section walks you through the process of troubleshooting common issues when using Amazon EMR with AWS Lake Formation.

Data lake access not allowed

You must explicitly opt in to data filtering on Amazon EMR clusters before you can analyze and process data in your data lake. When data access fails, you will see a generic `Access is not allowed` message in the output of your notebook entries.

To opt in and allow data filtering on Amazon EMR, see Allow data filtering on Amazon EMR in the *AWS Lake Formation Developer Guide* for instructions.

Session expiration

The session timeout for EMR Notebooks and Zeppelin is controlled by the IAM Role for Lake Formation's `Maximum CLI/API session duration` setting. The default value for this setting is one hour. When a session timeout occurs, you will see the following message in the output of your notebook entries when trying to run Spark SQL commands.

```
Error 401    HTTP ERROR: 401 Problem accessing /sessions/2/statements.
Reason:  JWT token included in request failed validation.
Powered by Jetty:// 9.3.24.v20180605
 org.springframework.web.client.HttpClientErrorException: 401 JWT token included in
 request failed validation…
```

To validate your session, refresh the page. You will be prompted to re-authenticate using your IdP and be redirected back to the Notebook. You can continue to run queries after re-authentication.

No permissions for user on requested table

When attempting to access a table that you do not have access to, you will see the following exception in the output of your notebook entries when trying to run Spark SQL commands.

```
org.apache.spark.sql.AnalysisException:
 org.apache.hadoop.hive.ql.metadata.HiveException: Unable to fetch table table.
Resource does not exist or requester is not authorized to access requested
 permissions.
(Service: AWSGlue; Status Code: 400; Error Code: AccessDeniedException; Request ID: …
```

To access the table, you must grant access to the user by updating the permissions associated with this table in Lake Formation.

Querying cross-account data shared with Lake Formation

When you use Amazon EMR to access data shared with you from another account, some Spark libraries will attempt to call `Glue:GetUserDefinedFunctions` API operation. Since versions 1 and 2 of the AWS RAM managed permissions does not support this action, you receive the following error message:

```
"ERROR: User: arn:aws:sts::012345678901:assumed-role/my-
spark-role/i-06ab8c2b59299508a is not authorized to perform:
glue:GetUserDefinedFunctions on resource: arn:exampleCatalogResource
```

```
because no resource-based policy allows the glue:GetUserDefinedFunctions
action"
```

To resolve this error, the data lake administrator who created the resource share must update the AWS RAM managed permissions attached to the resource share. Version 3 of the AWS RAM managed permissions allows principals to perform the `glue:GetUserDefinedFunctions` action.

If you create a new resource share, Lake Formation applies the latest version of the AWS RAM managed permission by default, and no action is required by you. To enable cross-account data access for existing resource shares, you need to update the AWS RAM managed permissions to version 3.

You can view the AWS RAM permissions assigned to resources shared with you in AWS RAM. The following permissions are included in version 3:

```
Databases
  AWSRAMPermissionGlueDatabaseReadWriteForCatalog
  AWSRAMPermissionGlueDatabaseReadWrite

Tables
  AWSRAMPermissionGlueTableReadWriteForCatalog
  AWSRAMPermissionGlueTableReadWriteForDatabase

AllTables
  AWSRAMPermissionGlueAllTablesReadWriteForCatalog
  AWSRAMPermissionGlueAllTablesReadWriteForDatabase
```

To update AWS RAM managed permissions version of existing resource shares

You (data lake administrator) can either update AWS RAM managed permissions to a newer version by following instructions in the *AWS RAM User Guide* or you can revoke all existing permissions for the resource type and regrant them. If you revoke permissions, AWS RAM deletes the AWS RAM resource share associated with the resource type. When you regrant permissions, AWS RAM creates new resource shares attaching the latest version of AWS RAM managed permissions.

Inserting into, creating, and altering tables

Inserting into, creating, or altering tables in databases protected by Lake Formation policies is not supported. When performing these operations, you will see the following exception in the output of your notebook entries when trying to run Spark SQL commands:

```
java.io.IOException:
 com.amazon.ws.emr.hadoop.fs.shaded.com.amazonaws.services.s3.model.AmazonS3Exception:
           Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied; Request ID: …
```

For more information, see Limitations of Amazon EMR integration with AWS Lake Formation.