name value description yarn.ipc.client.factory.class Factory to create client IPC classes. yarn.ipc.server.factory.class Factory to create server IPC classes. yarn.ipc.record.factory.class Factory to create serializeable records. yarn.ipc.rpc.class org.apache.hadoop.yarn.ipc.HadoopYarnProtoRPC RPC class implementation yarn.resourcemanager.hostname 0.0.0.0 The hostname of the RM. yarn.resourcemanager.address ${yarn.resourcemanager.hostname}:8032 The address of the applications manager interface in the RM. yarn.resourcemanager.bind-host The actual address the server will bind to. If this optional address is set, the RPC and webapp servers will bind to this address and the port specified in yarn.resourcemanager.address and yarn.resourcemanager.webapp.address, respectively. This is most useful for making RM listen to all interfaces by setting to 0.0.0.0. yarn.resourcemanager.auto-update.containers false If set to true, then ALL container updates will be automatically sent to the NM in the next heartbeat yarn.resourcemanager.client.thread-count 50 The number of threads used to handle applications manager requests. yarn.resourcemanager.amlauncher.thread-count 50 Number of threads used to launch/cleanup AM. yarn.resourcemanager.nodemanager-connect-retries 10 Retry times to connect with NM. yarn.dispatcher.drain-events.timeout 300000 Timeout in milliseconds when YARN dispatcher tries to drain the events. Typically, this happens when service is stopping. e.g. RM drains the ATS events dispatcher when stopping. yarn.am.liveness-monitor.expiry-interval-ms 600000 The expiry interval for application master reporting. yarn.resourcemanager.principal The Kerberos principal for the resource manager. yarn.resourcemanager.scheduler.address ${yarn.resourcemanager.hostname}:8030 The address of the scheduler interface. yarn.resourcemanager.scheduler.client.thread-count 50 Number of threads to handle scheduler interface. yarn.resourcemanager.application-master-service.processors Comma separated class names of ApplicationMasterServiceProcessor implementations. The processors will be applied in the order they are specified. yarn.http.policy HTTP_ONLY This configures the HTTP endpoint for YARN Daemons.The following values are supported: - HTTP_ONLY : Service is provided only on http - HTTPS_ONLY : Service is provided only on https yarn.resourcemanager.webapp.address ${yarn.resourcemanager.hostname}:8088 The http address of the RM web application. If only a host is provided as the value, the webapp will be served on a random port. yarn.resourcemanager.webapp.https.address ${yarn.resourcemanager.hostname}:8090 The https address of the RM web application. If only a host is provided as the value, the webapp will be served on a random port. yarn.resourcemanager.webapp.spnego-keytab-file The Kerberos keytab file to be used for spnego filter for the RM web interface. yarn.resourcemanager.webapp.spnego-principal The Kerberos principal to be used for spnego filter for the RM web interface. yarn.resourcemanager.webapp.ui-actions.enabled true Add button to kill application in the RM Application view. yarn.webapp.ui2.enable false To enable RM web ui2 application. yarn.webapp.ui2.war-file-path Explicitly provide WAR file path for ui2 if needed. yarn.resourcemanager.resource-tracker.address ${yarn.resourcemanager.hostname}:8031 yarn.acl.enable false Are acls enabled. yarn.acl.reservation-enable false Are reservation acls enabled. yarn.admin.acl * ACL of who can be admin of the YARN cluster. yarn.resourcemanager.admin.address ${yarn.resourcemanager.hostname}:8033 The address of the RM admin interface. yarn.resourcemanager.admin.client.thread-count 1 Number of threads used to handle RM admin interface. yarn.resourcemanager.connect.max-wait.ms 900000 Maximum time to wait to establish connection to ResourceManager. yarn.resourcemanager.connect.retry-interval.ms 30000 How often to try connecting to the ResourceManager. yarn.resourcemanager.am.max-attempts 2 The maximum number of application attempts. It's a global setting for all application masters. Each application master can specify its individual maximum number of application attempts via the API, but the individual number cannot be more than the global upper bound. If it is, the resourcemanager will override it. The default number is set to 2, to allow at least one retry for AM. yarn.resourcemanager.container.liveness-monitor.interval-ms 600000 How often to check that containers are still alive. yarn.resourcemanager.keytab /etc/krb5.keytab The keytab for the resource manager. yarn.resourcemanager.webapp.delegation-token-auth-filter.enabled true Flag to enable override of the default kerberos authentication filter with the RM authentication filter to allow authentication using delegation tokens(fallback to kerberos if the tokens are missing). Only applicable when the http authentication type is kerberos. yarn.resourcemanager.webapp.cross-origin.enabled false Flag to enable cross-origin (CORS) support in the RM. This flag requires the CORS filter initializer to be added to the filter initializers list in core-site.xml. yarn.nm.liveness-monitor.expiry-interval-ms 600000 How long to wait until a node manager is considered dead. yarn.resourcemanager.nodes.include-path Path to file with nodes to include. yarn.resourcemanager.nodes.exclude-path Path to file with nodes to exclude. yarn.resourcemanager.node-ip-cache.expiry-interval-secs -1 The expiry interval for node IP caching. -1 disables the caching yarn.resourcemanager.resource-tracker.client.thread-count 50 Number of threads to handle resource tracker calls. yarn.resourcemanager.scheduler.class org.apache.hadoop.yarn.server.resourcemanager.scheduler.capacity.CapacityScheduler The class to use as the resource scheduler. yarn.scheduler.minimum-allocation-mb 1024 The minimum allocation for every container request at the RM in MBs. Memory requests lower than this will be set to the value of this property. Additionally, a node manager that is configured to have less memory than this value will be shut down by the resource manager. yarn.scheduler.maximum-allocation-mb 8192 The maximum allocation for every container request at the RM in MBs. Memory requests higher than this will throw an InvalidResourceRequestException. yarn.scheduler.minimum-allocation-vcores 1 The minimum allocation for every container request at the RM in terms of virtual CPU cores. Requests lower than this will be set to the value of this property. Additionally, a node manager that is configured to have fewer virtual cores than this value will be shut down by the resource manager. yarn.scheduler.maximum-allocation-vcores 4 The maximum allocation for every container request at the RM in terms of virtual CPU cores. Requests higher than this will throw an InvalidResourceRequestException. yarn.scheduler.include-port-in-node-name false Used by node labels. If set to true, the port should be included in the node name. Only usable if your scheduler supports node labels. yarn.resourcemanager.recovery.enabled false Enable RM to recover state after starting. If true, then yarn.resourcemanager.store.class must be specified. yarn.resourcemanager.fail-fast ${yarn.fail-fast} Should RM fail fast if it encounters any errors. By defalt, it points to ${yarn.fail-fast}. Errors include: 1) exceptions when state-store write/read operations fails. yarn.fail-fast false Should YARN fail fast if it encounters any errors. This is a global config for all other components including RM,NM etc. If no value is set for component-specific config (e.g yarn.resourcemanager.fail-fast), this value will be the default. yarn.resourcemanager.work-preserving-recovery.enabled true Enable RM work preserving recovery. This configuration is private to YARN for experimenting the feature. yarn.resourcemanager.work-preserving-recovery.scheduling-wait-ms 10000 Set the amount of time RM waits before allocating new containers on work-preserving-recovery. Such wait period gives RM a chance to settle down resyncing with NMs in the cluster on recovery, before assigning new containers to applications. yarn.resourcemanager.store.class org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore The class to use as the persistent store. If org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore is used, the store is implicitly fenced; meaning a single ResourceManager is able to use the store at any point in time. More details on this implicit fencing, along with setting up appropriate ACLs is discussed under yarn.resourcemanager.zk-state-store.root-node.acl. yarn.resourcemanager.ha.failover-controller.active-standby-elector.zk.retries When automatic failover is enabled, number of zookeeper operation retry times in ActiveStandbyElector yarn.resourcemanager.state-store.max-completed-applications ${yarn.resourcemanager.max-completed-applications} The maximum number of completed applications RM state store keeps, less than or equals to ${yarn.resourcemanager.max-completed-applications}. By default, it equals to ${yarn.resourcemanager.max-completed-applications}. This ensures that the applications kept in the state store are consistent with the applications remembered in RM memory. Any values larger than ${yarn.resourcemanager.max-completed-applications} will be reset to ${yarn.resourcemanager.max-completed-applications}. Note that this value impacts the RM recovery performance. Typically, a smaller value indicates better performance on RM recovery. yarn.resourcemanager.zk-state-store.parent-path /rmstore Full path of the ZooKeeper znode where RM state will be stored. This must be supplied when using org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore as the value for yarn.resourcemanager.store.class yarn.resourcemanager.zk-state-store.root-node.acl ACLs to be used for the root znode when using ZKRMStateStore in an HA scenario for fencing. ZKRMStateStore supports implicit fencing to allow a single ResourceManager write-access to the store. For fencing, the ResourceManagers in the cluster share read-write-admin privileges on the root node, but the Active ResourceManager claims exclusive create-delete permissions. By default, when this property is not set, we use the ACLs from yarn.resourcemanager.zk-acl for shared admin access and rm-address:random-number for username-based exclusive create-delete access. This property allows users to set ACLs of their choice instead of using the default mechanism. For fencing to work, the ACLs should be carefully set differently on each ResourceManger such that all the ResourceManagers have shared admin access and the Active ResourceManger takes over (exclusively) the create-delete access. yarn.resourcemanager.fs.state-store.uri ${hadoop.tmp.dir}/yarn/system/rmstore URI pointing to the location of the FileSystem path where RM state will be stored. This must be supplied when using org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore as the value for yarn.resourcemanager.store.class yarn.resourcemanager.fs.state-store.num-retries 0 the number of retries to recover from IOException in FileSystemRMStateStore. yarn.resourcemanager.fs.state-store.retry-interval-ms 1000 Retry interval in milliseconds in FileSystemRMStateStore. yarn.resourcemanager.leveldb-state-store.path ${hadoop.tmp.dir}/yarn/system/rmstore Local path where the RM state will be stored when using org.apache.hadoop.yarn.server.resourcemanager.recovery.LeveldbRMStateStore as the value for yarn.resourcemanager.store.class yarn.resourcemanager.leveldb-state-store.compaction-interval-secs 3600 The time in seconds between full compactions of the leveldb database. Setting the interval to zero disables the full compaction cycles. yarn.resourcemanager.ha.enabled false Enable RM high-availability. When enabled, (1) The RM starts in the Standby mode by default, and transitions to the Active mode when prompted to. (2) The nodes in the RM ensemble are listed in yarn.resourcemanager.ha.rm-ids (3) The id of each RM either comes from yarn.resourcemanager.ha.id if yarn.resourcemanager.ha.id is explicitly specified or can be figured out by matching yarn.resourcemanager.address.{id} with local address (4) The actual physical addresses come from the configs of the pattern - {rpc-config}.{id} yarn.resourcemanager.ha.automatic-failover.enabled true Enable automatic failover. By default, it is enabled only when HA is enabled yarn.resourcemanager.ha.automatic-failover.embedded true Enable embedded automatic failover. By default, it is enabled only when HA is enabled. The embedded elector relies on the RM state store to handle fencing, and is primarily intended to be used in conjunction with ZKRMStateStore. yarn.resourcemanager.ha.automatic-failover.zk-base-path /yarn-leader-election The base znode path to use for storing leader information, when using ZooKeeper based leader election. yarn.resourcemanager.zk-appid-node.split-index 0 Index at which last section of application id (with each section separated by _ in application id) will be split so that application znode stored in zookeeper RM state store will be stored as two different znodes (parent-child). Split is done from the end. For instance, with no split, appid znode will be of the form application_1352994193343_0001. If the value of this config is 1, the appid znode will be broken into two parts application_1352994193343_000 and 1 respectively with former being the parent node. application_1352994193343_0002 will then be stored as 2 under the parent node application_1352994193343_000. This config can take values from 0 to 4. 0 means there will be no split. If configuration value is outside this range, it will be treated as config value of 0(i.e. no split). A value larger than 0 (up to 4) should be configured if you are storing a large number of apps in ZK based RM state store and state store operations are failing due to LenError in Zookeeper. yarn.resourcemanager.zk-delegation-token-node.split-index 0 Index at which the RM Delegation Token ids will be split so that the delegation token znodes stored

in the zookeeper RM state store will be stored as two different znodes (parent-child). The split is done from the end. For instance, with no split, a delegation token znode will be of the form RMDelegationToken_123456789. If the value of this config is 1, the delegation token znode will be broken into two parts: RMDelegationToken_12345678 and 9 respectively with former being the parent node. This config can take values from 0 to 4. 0 means there will be no split. If the value is outside this range, it will be treated as 0 (i.e. no split). A value larger than 0 (up to 4) should be configured if you are running a large number of applications, with long-lived delegation tokens and state store operations (e.g. failover) are failing due to LenError in Zookeeper. yarn.resourcemanager.zk-max-znode-size.bytes 1048576 Specifies the maximum size of the data that can be stored in a znode. Value should be same or less than jute.maxbuffer configured in zookeeper. Default value configured is 1MB. yarn.resourcemanager.cluster-id Name of the cluster. In a HA setting, this is used to ensure the RM participates in leader election for this cluster and ensures it does not affect other clusters yarn.resourcemanager.ha.rm-ids The list of RM nodes in the cluster when HA is enabled. See description of yarn.resourcemanager.ha .enabled for full details on how this is used. yarn.resourcemanager.ha.id The id (string) of the current RM. When HA is enabled, this is an optional config. The id of current RM can be set by explicitly specifying yarn.resourcemanager.ha.id or figured out by matching yarn.resourcemanager.address.{id} with local address See description of yarn.resourcemanager.ha.enabled for full details on how this is used. yarn.client.failover-proxy-provider org.apache.hadoop.yarn.client.ConfiguredRMFailoverProxyProvider When HA is enabled, the class to be used by Clients, AMs and NMs to failover to the Active RM. It should extend org.apache.hadoop.yarn.client.RMFailoverProxyProvider yarn.client.failover-max-attempts When HA is enabled, the max number of times FailoverProxyProvider should attempt failover. When set, this overrides the yarn.resourcemanager.connect.max-wait.ms. When not set, this is inferred from yarn.resourcemanager.connect.max-wait.ms. yarn.client.failover-sleep-base-ms When HA is enabled, the sleep base (in milliseconds) to be used for calculating the exponential delay between failovers. When set, this overrides the yarn.resourcemanager.connect.* settings. When not set, yarn.resourcemanager.connect.retry-interval.ms is used instead. yarn.client.failover-sleep-max-ms When HA is enabled, the maximum sleep time (in milliseconds) between failovers. When set, this overrides the yarn.resourcemanager.connect.* settings. When not set, yarn.resourcemanager.connect.retry-interval.ms is used instead. yarn.client.failover-retries 0 When HA is enabled, the number of retries per attempt to connect to a ResourceManager. In other words, it is the ipc.client.connect.max.retries to be used during failover attempts yarn.client.failover-retries-on-socket-timeouts 0 When HA is enabled, the number of retries per attempt to connect to a ResourceManager on socket timeouts. In other words, it is the ipc.client.connect.max.retries.on.timeouts to be used during failover attempts yarn.resourcemanager.max-completed-applications 1000 The maximum number of completed applications RM keeps. yarn.resourcemanager.delayed.delegation-token.removal-interval-ms 30000 Interval at which the delayed token removal thread runs yarn.resourcemanager.delegation-token.max-conf-size-bytes 12800 Maximum size in bytes for configurations that can be provided by application to RM for delegation token renewal. By experiment, it's roughly 128 bytes per key-value pair. The default value 12800 allows roughly 100 configs, may be less. yarn.resourcemanager.proxy-user-privileges.enabled false If true, ResourceManager will have proxy-user privileges. Use case: In a secure cluster, YARN requires the user hdfs delegation-tokens to do localization and log-aggregation on behalf of the user. If this is set to true, ResourceManager is able to request new hdfs delegation tokens on behalf of the user. This is needed by long-running-service, because the hdfs tokens will eventually expire and YARN requires new valid tokens to do localization and log-aggregation. Note that to enable this use case, the corresponding HDFS NameNode has to configure ResourceManager as the proxy-user so that ResourceManager can itself ask for new tokens on behalf of the user when tokens are past their max-life-time. yarn.resourcemanager.am-rm-tokens.master-key-rolling-interval-secs 86400 Interval for the roll over for the master key used to generate application tokens yarn.resourcemanager.container-tokens.master-key-rolling-interval-secs 86400 Interval for the roll over for the master key used to generate container tokens. It is expected to be much greater than yarn.nm.liveness-monitor.expiry-interval-ms and yarn.resourcemanager.rm.container-allocation.expiry-interval-ms. Otherwise the behavior is undefined. yarn.resourcemanager.nodemanagers.heartbeat-interval-ms 1000 The heart-beat interval in milliseconds for every NodeManager in the cluster. yarn.resourcemanager.nodemanager.minimum.version NONE The minimum allowed version of a connecting nodemanager. The valid values are NONE (no version checking), EqualToRM (the nodemanager's version is equal to or greater than the RM version), or a Version String. yarn.resourcemanager.scheduler.monitor.enable false Enable a set of periodic monitors (specified in yarn.resourcemanager.scheduler.monitor.policies) that affect the scheduler. yarn.resourcemanager.scheduler.monitor.policies org.apache.hadoop.yarn.server.resourcemanager.monitor.capacity.ProportionalCapacityPreemptionPolicy The list of SchedulingEditPolicy classes that interact with the scheduler. A particular module may be incompatible with the scheduler, other policies, or a configuration of either. yarn.resourcemanager.configuration.provider-class org.apache.hadoop.yarn.LocalConfigurationProvider The class to use as the configuration provider. If org.apache.hadoop.yarn.LocalConfigurationProvider is used, the local configuration will be loaded. If org.apache.hadoop.yarn.FileSystemBasedConfigurationProvider is used, the configuration which will be loaded should be uploaded to remote File system first. yarn.resourcemanager.configuration.file-system-based-store /yarn/conf The value specifies the file system (e.g. HDFS) path where ResourceManager loads configuration if yarn.resourcemanager.configuration.provider-class is set to org.apache.hadoop.yarn.FileSystemBasedConfigurationProvider. yarn.resourcemanager.system-metrics-publisher.enabled false The setting that controls whether yarn system metrics is published to the Timeline server (version one) or not, by RM. This configuration is now deprecated in favor of yarn.system-metrics-publisher.enabled. yarn.system-metrics-publisher.enabled false The setting that controls whether yarn system metrics is published on the Timeline service or not by RM And NM. yarn.rm.system-metrics-publisher.emit-container-events false The setting that controls whether yarn container events are published to the timeline service or not by RM. This configuration setting is for ATS V2. yarn.resourcemanager.system-metrics-publisher.dispatcher.pool-size 10 Number of worker threads that send the yarn system metrics data. yarn.resourcemanager.max-log-aggregation-diagnostics-in-memory 10 Number of diagnostics/failure messages can be saved in RM for log aggregation. It also defines the number of diagnostics/failure messages can be shown in log aggregation web ui. yarn.resourcemanager.delegation-token-renewer.thread-count 50 RM DelegationTokenRenewer thread count yarn.resourcemanager.delegation.key.update-interval 86400000 RM secret key update interval in ms yarn.resourcemanager.delegation.token.max-lifetime 604800000 RM delegation token maximum lifetime in ms yarn.resourcemanager.delegation.token.renew-interval 86400000 RM delegation token update interval in ms yarn.resourcemanager.history-writer.multi-threaded-dispatcher.pool-size 10 Thread pool size for RMApplicationHistoryWriter. yarn.resourcemanager.metrics.runtime.buckets 60,300,1440 Comma-separated list of values (in minutes) for schedule queue related metrics. yarn.resourcemanager.nm-tokens.master-key-rolling-interval-secs 86400 Interval for the roll over for the master key used to generate NodeManager tokens. It is expected to be set to a value much larger than yarn.nm.liveness-monitor.expiry-interval-ms. yarn.resourcemanager.reservation-system.enable false Flag to enable the ResourceManager reservation system. yarn.resourcemanager.reservation-system.class The Java class to use as the ResourceManager reservation system. By default, is set to org.apache.hadoop.yarn.server.resourcemanager.reservation.CapacityReservationSystem when using CapacityScheduler and is set to org.apache.hadoop.yarn.server.resourcemanager.reservation.FairReservationSystem when using FairScheduler. yarn.resourcemanager.reservation-system.plan.follower The plan follower policy class name to use for the ResourceManager reservation system. By default, is set to org.apache.hadoop.yarn.server.resourcemanager.reservation.CapacitySchedulerPlanFollower is used when using CapacityScheduler, and is set to org.apache.hadoop.yarn.server.resourcemanager.reservation.FairSchedulerPlanFollower when using FairScheduler. yarn.resourcemanager.reservation-system.planfollower.time-step 1000 Step size of the reservation system in ms yarn.resourcemanager.rm.container-allocation.expiry-interval-ms 600000 The expiry interval for a container yarn.nodemanager.hostname 0.0.0.0 The hostname of the NM. yarn.nodemanager.address ${yarn.nodemanager.hostname}:0 The address of the container manager in the NM. yarn.nodemanager.bind-host The actual address the server will bind to. If this optional address is set, the RPC and webapp servers will bind to this address and the port specified in yarn.nodemanager.address and yarn.nodemanager.webapp.address, respectively. This is most useful for making NM listen to all interfaces by setting to 0.0.0.0. yarn.nodemanager.admin-env MALLOC_ARENA_MAX=$MALLOC_ARENA_MAX Environment variables that should be forwarded from the NodeManager's environment to the container's. yarn.nodemanager.env-whitelist JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HAD
Environment variables that containers may override rather than use NodeManager's default. yarn.nodemanager.container-executor.class org.apache.hadoop.yarn.server.nodemanager.DefaultContainerExecutor who will execute(launch) the containers. yarn.nodemanager.container-state-transition-listener.classes Comma separated List of container state transition listeners. yarn.nodemanager.container-manager.thread-count 20 Number of threads container manager uses. yarn.nodemanager.collector-service.thread-count 5 Number of threads collector service uses. yarn.nodemanager.delete.thread-count 4 Number of threads used in cleanup. yarn.nodemanager.opportunistic-containers-max-queue-length 0 Max number of OPPORTUNISTIC containers to queue at the nodemanager. yarn.nodemanager.delete.debug-delay-sec 0 Number of seconds after an application finishes before the nodemanager's DeletionService will delete the application's localized file directory and log directory. To diagnose YARN application problems, set this property's value large enough (for example, to 600 = 10 minutes) to permit examination of these directories. After changing the property's value, you must restart the nodemanager in order for it to have an effect. The roots of YARN applications' work directories is configurable with the yarn.nodemanager.local-dirs property (see below), and the roots of the YARN applications' log directories is configurable with the yarn.nodemanager.log-dirs property (see also below). yarn.nodemanager.keytab /etc/krb5.keytab Keytab for NM. yarn.nodemanager.local-dirs ${hadoop.tmp.dir}/nm-local-dir List of directories to store localized files in. An application's localized file directory will be found in: ${yarn.nodemanager.local-dirs}/usercache/${user}/appcache/application_${appid}. Individual containers' work directories, called container_${contid}, will be subdirectories of this. yarn.nodemanager.local-cache.max-files-per-directory 8192 It limits the maximum number of files which will be localized in a single local directory. If the limit is reached then sub-directories will be created and new files will be localized in them. If it is set to a value less than or equal to 36 [which are sub-directories (0-9 and then a-z)] then NodeManager will fail to start. For example; [for public cache] if this is configured with a value of 40 (4 files + 36 sub-directories) and the local-dir is "/tmp/local-dir1" then it will allow 4 files to be created directly inside "/tmp/local-dir1/filecache". For files that are localized further it will create a sub-directory "0" inside "/tmp/local-dir1/filecache" and will localize files inside it until it becomes full. If a file is removed from a sub-directory that is marked full, then that sub-directory will be used back again to localize files. yarn.nodemanager.localizer.address ${yarn.nodemanager.hostname}:8040 Address where the localizer IPC is. yarn.nodemanager.collector-service.address ${yarn.nodemanager.hostname}:8048 Address where the collector service IPC is. yarn.nodemanager.localizer.cache.cleanup.interval-ms 600000 Interval in between cache cleanups. yarn.nodemanager.localizer.cache.target-size-mb 10240 Target size of localizer cache in MB, per nodemanager. It is a target retention size that only includes resources with PUBLIC and PRIVATE visibility and excludes resources with

APPLICATION visibility yarn.nodemanager.localizer.client.thread-count 5 Number of threads to handle localization requests. yarn.nodemanager.localizer.fetch.thread-count
4 Number of threads to use for localization fetching. yarn.nodemanager.container-localizer.java.opts -Xmx256m yarn.nodemanager.log-dirs ${yarn.log.dir}/userlogs Where
to store container logs. An application's localized log directory will be found in ${yarn.nodemanager.log-dirs}/application_${appid}. Individual containers' log directories
will be below this, in directories named container_{$contid}. Each container directory will contain the files stderr, stdin, and syslog generated by that container.
yarn.nodemanager.default-container-executor.log-dirs.permissions 710 The permissions settings used for the creation of container directories when using
DefaultContainerExecutor. This follows standard user/group/all permissions format. yarn.log-aggregation-enable false Whether to enable log aggregation. Log aggregation
collects each container's logs and moves these logs onto a file-system, for e.g. HDFS, after the application completes. Users can configure the "yarn.nodemanager.remote-
app-log-dir" and "yarn.nodemanager.remote-app-log-dir-suffix" properties to determine where these logs are moved to. Users can access the logs via the Application Timeline
Server. yarn.log-aggregation.retain-seconds -1 How long to keep aggregation logs before deleting them. -1 disables. Be careful set this too small and you will spam the name
node. yarn.log-aggregation.retain-check-interval-seconds -1 How long to wait between aggregated log retention checks. If set to 0 or a negative value then the value is
computed as one-tenth of the aggregated log retention time. Be careful set this too small and you will spam the name node. yarn.log-aggregation.file-formats TFile Specify
which log file controllers we will support. The first file controller we add will be used to write the aggregated logs. This comma separated configuration will work with the
configuration: yarn.log-aggregation.file-controller.%s.class which defines the supported file controller's class. By default, the TFile controller would be used. The user could
override this configuration by adding more file controllers. To support back-ward compatibility, make sure that we always add TFile file controller. yarn.log-aggregation.file-
controller.TFile.class org.apache.hadoop.yarn.logaggregation.filecontroller.tfile.LogAggregationTFileController Class that supports TFile read and write operations. yarn.log-
aggregation-status.time-out.ms 600000 How long for ResourceManager to wait for NodeManager to report its log aggregation status. If waiting time of which the log
aggregation status is reported from NodeManager exceeds the configured value, RM will report log aggregation status for this NodeManager as TIME_OUT
yarn.nodemanager.log.retain-seconds 10800 Time in seconds to retain user logs. Only applicable if log aggregation is disabled yarn.nodemanager.remote-app-log-dir
/tmp/logs Where to aggregate logs to. yarn.nodemanager.remote-app-log-dir-suffix logs The remote log dir will be created at {yarn.nodemanager.remote-app-log-
dir}/${user}/{thisParam} yarn.nodemanager.log-container-debug-info.enabled true Generate additional logs about container launches. Currently, this creates a copy of the
launch script and lists the directory contents of the container work dir. When listing directory contents, we follow symlinks to a max-depth of 5(including symlinks which
point to outside the container work dir) which may lead to a slowness in launching containers. yarn.nodemanager.resource.memory-mb -1 Amount of physical memory, in
MB, that can be allocated for containers. If set to -1 and yarn.nodemanager.resource.detect-hardware-capabilities is true, it is automatically calculated(in case of Windows
and Linux). In other cases, the default is 8192MB. yarn.nodemanager.resource.system-reserved-memory-mb -1 Amount of physical memory, in MB, that is reserved for non-
YARN processes. This configuration is only used if yarn.nodemanager.resource.detect-hardware-capabilities is set to true and yarn.nodemanager.resource.memory-mb is -1.
If set to -1, this amount is calculated as 20% of (system memory - 2*HADOOP_HEAPSIZE) yarn.nodemanager.pmem-check-enabled true Whether physical memory limits
will be enforced for containers. yarn.nodemanager.vmem-check-enabled true Whether virtual memory limits will be enforced for containers. yarn.nodemanager.vmem-
pmem-ratio 2.1 Ratio between virtual memory to physical memory when setting memory limits for containers. Container allocations are expressed in terms of physical
memory, and virtual memory usage is allowed to exceed this allocation by this ratio. yarn.nodemanager.resource.cpu-vcores -1 Number of vcores that can be allocated for
containers. This is used by the RM scheduler when allocating resources for containers. This is not used to limit the number of CPUs used by YARN containers. If it is set to
-1 and yarn.nodemanager.resource.detect-hardware-capabilities is true, it is automatically determined from the hardware in case of Windows and Linux. In other cases,
number of vcores is 8 by default. yarn.nodemanager.resource.count-logical-processors-as-cores false Flag to determine if logical processors(such as hyperthreads) should be
counted as cores. Only applicable on Linux when yarn.nodemanager.resource.cpu-vcores is set to -1 and yarn.nodemanager.resource.detect-hardware-capabilities is true.
yarn.nodemanager.resource.pcores-vcores-multiplier 1.0 Multiplier to determine how to convert phyiscal cores to vcores. This value is used if
yarn.nodemanager.resource.cpu-vcores is set to -1(which implies auto-calculate vcores) and yarn.nodemanager.resource.detect-hardware-capabilities is set to true. The
number of vcores will be calculated as number of CPUs * multiplier. yarn.nodemanager.logaggregation.threadpool-size-max 100 Thread pool size for
LogAggregationService in Node Manager. yarn.nodemanager.resource.percentage-physical-cpu-limit 100 Percentage of CPU that can be allocated for containers. This setting
allows users to limit the amount of CPU that YARN containers use. Currently functional only on Linux using cgroups. The default is to use 100% of CPU.
yarn.nodemanager.resource.detect-hardware-capabilities false Enable auto-detection of node capabilities such as memory and CPU. yarn.nodemanager.webapp.address
${yarn.nodemanager.hostname}:8042 NM Webapp address. yarn.nodemanager.webapp.https.address 0.0.0.0:8044 The https adddress of the NM web application.
yarn.nodemanager.webapp.spnego-keytab-file The Kerberos keytab file to be used for spnego filter for the NM web interface. yarn.nodemanager.webapp.spnego-principal
The Kerberos principal to be used for spnego filter for the NM web interface. yarn.nodemanager.resource-monitor.interval-ms 3000 How often to monitor the node and the
containers. If 0 or negative, monitoring is disabled. yarn.nodemanager.resource-calculator.class Class that calculates current resource utilization. yarn.nodemanager.container-
monitor.enabled true Enable container monitor yarn.nodemanager.container-monitor.interval-ms How often to monitor containers. If not set, the value for
yarn.nodemanager.resource-monitor.interval-ms will be used. If 0 or negative, container monitoring is disabled. yarn.nodemanager.container-monitor.resource-calculator.class
Class that calculates containers current resource utilization. If not set, the value for yarn.nodemanager.resource-calculator.class will be used. yarn.nodemanager.health-
checker.interval-ms 600000 Frequency of running node health script. yarn.nodemanager.health-checker.script.timeout-ms 1200000 Script time out period.
yarn.nodemanager.health-checker.script.path The health check script to run. yarn.nodemanager.health-checker.script.opts The arguments to pass to the health check script.
yarn.nodemanager.disk-health-checker.interval-ms 120000 Frequency of running disk health checker code. yarn.nodemanager.disk-health-checker.min-healthy-disks 0.25 The
minimum fraction of number of disks to be healthy for the nodemanager to launch new containers. This correspond to both yarn.nodemanager.local-dirs and
yarn.nodemanager.log-dirs. i.e. If there are less number of healthy local-dirs (or log-dirs) available, then new containers will not be launched on this node.
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 equal to 100, the nodemanager will check for full disk. This applies to yarn.nodemanager.local-dirs
and yarn.nodemanager.log-dirs. yarn.nodemanager.disk-health-checker.disk-utilization-watermark-low-per-disk-percentage The low threshold percentage of disk space used
when a bad disk is marked as good. Values can range from 0.0 to 100.0. This applies to yarn.nodemanager.local-dirs and yarn.nodemanager.log-dirs. Note that if its value is
more than yarn.nodemanager.disk-health-checker. max-disk-utilization-per-disk-percentage or not set, it will be set to the same value as yarn.nodemanager.disk-health-
checker.max-disk-utilization-per-disk-percentage. 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. yarn.nodemanager.linux-container-executor.path The path to the Linux container
executor. yarn.nodemanager.linux-container-executor.resources-handler.class org.apache.hadoop.yarn.server.nodemanager.util.DefaultLCEResourcesHandler The class which
should help the LCE handle resources. yarn.nodemanager.linux-container-executor.cgroups.hierarchy /hadoop-yarn The cgroups hierarchy under which to place YARN
proccesses (cannot contain commas). If yarn.nodemanager.linux-container-executor.cgroups.mount is false (that is, if cgroups have been pre-configured) and the YARN user
has write access to the parent directory, then the directory will be created. If the directory already exists, the administrator has to give YARN write permissions to it
recursively. This property only applies when the LCE resources handler is set to CgroupsLCEResourcesHandler. yarn.nodemanager.linux-container-executor.cgroups.mount
false Whether the LCE should attempt to mount cgroups if not found. This property only applies when the LCE resources handler is set to CgroupsLCEResourcesHandler.
yarn.nodemanager.linux-container-executor.cgroups.mount-path This property sets the path from which YARN will read the CGroups configuration. YARN has built-in
functionality to discover the system CGroup mount paths, so use this property only if YARN's automatic mount path discovery does not work. The path specified by this
property must exist before the NodeManager is launched. If yarn.nodemanager.linux-container-executor.cgroups.mount is set to true, YARN will first try to mount the
CGroups at the specified path before reading them. If yarn.nodemanager.linux-container-executor.cgroups.mount is set to false, YARN will read the CGroups at the specified
path. If this property is empty, YARN tries to detect the CGroups location. Please refer to NodeManagerCgroups.html in the documentation for further details. This property
only applies when the LCE resources handler is set to CgroupsLCEResourcesHandler. yarn.nodemanager.linux-container-executor.cgroups.delete-delay-ms 20 Delay in ms
between attempts to remove linux cgroup yarn.nodemanager.linux-container-executor.nonsecure-mode.limit-users true This determines which of the two modes that LCE
should use on a non-secure cluster. If this value is set to true, then all containers will be launched as the user specified in yarn.nodemanager.linux-container-
executor.nonsecure-mode.local-user. If this value is set to false, then containers will run as the user who submitted the application. yarn.nodemanager.linux-container-
executor.nonsecure-mode.local-user nobody The UNIX user that containers will run as when Linux-container-executor is used in nonsecure mode (a use case for this is using
cgroups) if the yarn.nodemanager.linux-container-executor.nonsecure-mode.limit-users is set to true. yarn.nodemanager.linux-container-executor.nonsecure-mode.user-
pattern ^[_.A-Za-z0-9][-@_.A-Za-z0-9]{0,255}?[$]?$ The allowed pattern for UNIX user names enforced by Linux-container-executor when used in nonsecure mode (use
case for this is using cgroups). The default value is taken from /usr/sbin/adduser yarn.nodemanager.linux-container-executor.cgroups.strict-resource-usage false This flag
determines whether apps should run with strict resource limits or be allowed to consume spare resources if they need them. For example, turning the flag on will restrict apps
to use only their share of CPU, even if the node has spare CPU cycles. The default value is false i.e. use available resources. Please note that turning this flag on may reduce
job throughput on the cluster. yarn.nodemanager.runtime.linux.allowed-runtimes default Comma separated list of runtimes that are allowed when using
LinuxContainerExecutor. The allowed values are default, docker, and javasandbox. yarn.nodemanager.runtime.linux.docker.capabilities
CHOWN,DAC_OVERRIDE,FSETID,FOWNER,MKNOD,NET_RAW,SETGID,SETUID,SETFCAP,SETPCAP,NET_BIND_SERVICE,SYS_CHROOT,KILL,AUDIT_WRI
This configuration setting determines the capabilities assigned to docker containers when they are launched. While these may not be case-sensitive from a docker perspective,
it is best to keep these uppercase. To run without any capabilites, set this value to "none" or "NONE" yarn.nodemanager.runtime.linux.docker.privileged-containers.allowed
false This configuration setting determines if privileged docker containers are allowed on this cluster. Use with extreme care.
yarn.nodemanager.runtime.linux.docker.privileged-containers.acl This configuration setting determines who is allowed to run privileged docker containers on this cluster. Use

with extreme care. yarn.nodemanager.runtime.linux.docker.allowed-container-networks host,none,bridge The set of networks allowed when launching containers using the DockerContainerRuntime. yarn.nodemanager.runtime.linux.docker.default-container-network host The network used when launching containers using the DockerContainerRuntime when no network is specified in the request . This network must be one of the (configurable) set of allowed container networks. yarn.nodemanager.runtime.linux.docker.enable-userremapping.allowed false Property to enable docker user remapping yarn.nodemanager.runtime.linux.docker.userremapping-uid-threshold 1 lower limit for acceptable uids of user remapped user yarn.nodemanager.runtime.linux.docker.userremapping-gid-threshold 1 lower limit for acceptable gids of user remapped user yarn.nodemanager.runtime.linux.sandbox-mode disabled The mode in which the Java Container Sandbox should run detailed by the JavaSandboxLinuxContainerRuntime. yarn.nodemanager.runtime.linux.sandbox-mode.local-dirs.permissions read Permissions for application local directories. yarn.nodemanager.runtime.linux.sandbox-mode.policy Location for non-default java policy file. yarn.nodemanager.runtime.linux.sandbox-mode.whitelist-group The group which will run by default without the java security manager. yarn.nodemanager.windows-container.memory-limit.enabled false This flag determines whether memory limit will be set for the Windows Job Object of the containers launched by the default container executor. yarn.nodemanager.windows-container.cpu-limit.enabled false This flag determines whether CPU limit will be set for the Windows Job Object of the containers launched by the default container executor. yarn.nodemanager.linux-container-executor.cgroups.delete-timeout-ms 1000 Interval of time the linux container executor should try cleaning up cgroups entry when cleaning up a container. yarn.nodemanager.linux-container-executor.group The UNIX group that the linux-container-executor should run as. yarn.nodemanager.log-aggregation.compression-type none T-file compression types used to compress aggregated logs. yarn.nodemanager.principal The kerberos principal for the node manager. yarn.nodemanager.aux-services A comma separated list of services where service name should only contain a-zA-Z0-9_ and can not start with numbers yarn.nodemanager.sleep-delay-before-sigkill.ms 250 No. of ms to wait between sending a SIGTERM and SIGKILL to a container yarn.nodemanager.process-kill-wait.ms 5000 Max time to wait for a process to come up when trying to cleanup a container yarn.nodemanager.resourcemanager.minimum.version NONE The minimum allowed version of a resourcemanager that a nodemanager will connect to. The valid values are NONE (no version checking), EqualToNM (the resourcemanager's version is equal to or greater than the NM version), or a Version String. yarn.nodemanager.container-diagnostics-maximum-size 10000 Maximum size of contain's diagnostics to keep for relaunching container case. yarn.nodemanager.container-retry-minimum-interval-ms 1000 Minimum container restart interval in milliseconds. yarn.client.nodemanager-client-async.thread-pool-max-size 500 Max number of threads in NMClientAsync to process container management events yarn.client.nodemanager-connect.max-wait-ms 180000 Max time to wait to establish a connection to NM yarn.client.nodemanager-connect.retry-interval-ms 10000 Time interval between each attempt to connect to NM yarn.nodemanager.resourcemanager.connect.max-wait.ms Max time to wait for NM to connect to RM. When not set, proxy will fall back to use value of yarn.resourcemanager.connect.max-wait.ms. yarn.nodemanager.resourcemanager.connect.retry-interval.ms Time interval between each NM attempt to connect to RM. When not set, proxy will fall back to use value of yarn.resourcemanager.connect.retry-interval.ms. yarn.client.max-cached-nodemanagers-proxies 0 Maximum number of proxy connections to cache for node managers. If set to a value greater than zero then the cache is enabled and the NMClient and MRAppMaster will cache the specified number of node manager proxies. There will be at max one proxy per node manager. Ex. configuring it to a value of 5 will make sure that client will at max have 5 proxies cached with 5 different node managers. These connections for these proxies will be timed out if idle for more than the system wide idle timeout period. Note that this could cause issues on large clusters as many connections could linger simultaneously and lead to a large number of connection threads. The token used for authentication will be used only at connection creation time. If a new token is received then the earlier connection should be closed in order to use the new token. This and (yarn.client.nodemanager-client-async.thread-pool-max-size) are related and should be in sync (no need for them to be equal). If the value of this property is zero then the connection cache is disabled and connections will use a zero idle timeout to prevent too many connection threads on large clusters. yarn.nodemanager.recovery.enabled false Enable the node manager to recover after starting yarn.nodemanager.recovery.dir ${hadoop.tmp.dir}/yarn-nm-recovery The local filesystem directory in which the node manager will store state when recovery is enabled. yarn.nodemanager.recovery.compaction-interval-secs 3600 The time in seconds between full compactions of the NM state database. Setting the interval to zero disables the full compaction cycles. yarn.nodemanager.recovery.supervised false Whether the nodemanager is running under supervision. A nodemanager that supports recovery and is running under supervision will not try to cleanup containers as it exits with the assumption it will be immediately be restarted and recover containers. yarn.nodemanager.container-executor.os.sched.priority.adjustment 0 Adjustment to the container OS scheduling priority. In Linux, passed directly to the nice command. yarn.nodemanager.container-metrics.enable true Flag to enable container metrics yarn.nodemanager.container-metrics.period-ms -1 Container metrics flush period in ms. Set to -1 for flush on completion. yarn.nodemanager.container-metrics.unregister-delay-ms 10000 The delay time ms to unregister container metrics after completion. yarn.nodemanager.container-monitor.process-tree.class Class used to calculate current container resource utilization. yarn.nodemanager.disk-health-checker.enable true Flag to enable NodeManager disk health checker yarn.nodemanager.log.deletion-threads-count 4 Number of threads to use in NM log cleanup. Used when log aggregation is disabled. yarn.nodemanager.windows-secure-container-executor.group The Windows group that the windows-container-executor should run as. yarn.nodemanager.aux-services.mapreduce_shuffle.class org.apache.hadoop.mapred.ShuffleHandler yarn.web-proxy.principal The kerberos principal for the proxy, if the proxy is not running as part of the RM. yarn.web-proxy.keytab Keytab for WebAppProxy, if the proxy is not running as part of the RM. yarn.web-proxy.address The address for the web proxy as HOST:PORT, if this is not given then the proxy will run as part of the RM yarn.application.classpath CLASSPATH for YARN applications. A comma-separated list of CLASSPATH entries. When this value is empty, the following default CLASSPATH for YARN applications would be used. For Linux: $HADOOP_CONF_DIR, $HADOOP_COMMON_HOME/share/hadoop/common/*, $HADOOP_COMMON_HOME/share/hadoop/common/lib/*, $HADOOP_HDFS_HOME/share/hadoop/hdfs/*, $HADOOP_HDFS_HOME/share/hadoop/hdfs/lib/*, $HADOOP_YARN_HOME/share/hadoop/yarn/*, $HADOOP_YARN_HOME/share/hadoop/yarn/lib/* For Windows: %HADOOP_CONF_DIR%, %HADOOP_COMMON_HOME%/share/hadoop/common/*, %HADOOP_COMMON_HOME%/share/hadoop/common/lib/*, %HADOOP_HDFS_HOME%/share/hadoop/hdfs/*, %HADOOP_HDFS_HOME%/share/hadoop/hdfs/lib/*, %HADOOP_YARN_HOME%/share/hadoop/yarn/*, %HADOOP_YARN_HOME%/share/hadoop/yarn/lib/* yarn.timeline-service.version 1.0f Indicate what is the current version of the running timeline service. For example, if "yarn.timeline-service.version" is 1.5, and "yarn.timeline-service.enabled" is true, it means the cluster will and should bring up the timeline service v.1.5 (and nothing else). On the client side, if the client uses the same version of timeline service, it should succeed. If the client chooses to use a smaller version in spite of this, then depending on how robust the compatibility story is between versions, the results may vary. yarn.timeline-service.enabled false In the server side it indicates whether timeline service is enabled or not. And in the client side, users can enable it to indicate whether client wants to use timeline service. If it's enabled in the client side along with security, then yarn client tries to fetch the delegation tokens for the timeline server. yarn.timeline-service.hostname 0.0.0.0 The hostname of the timeline service web application. yarn.timeline-service.address ${yarn.timeline-service.hostname}:10200 This is default address for the timeline server to start the RPC server. yarn.timeline-service.webapp.address ${yarn.timeline-service.hostname}:8188 The http address of the timeline service web application. yarn.timeline-service.webapp.https.address ${yarn.timeline-service.hostname}:8190 The https address of the timeline service web application. yarn.timeline-service.bind-host The actual address the server will bind to. If this optional address is set, the RPC and webapp servers will bind to this address and the port specified in yarn.timeline-service.address and yarn.timeline-service.webapp.address, respectively. This is most useful for making the service listen to all interfaces by setting to 0.0.0.0. yarn.timeline-service.generic-application-history.max-applications 10000 Defines the max number of applications could be fetched using REST API or application history protocol and shown in timeline server web ui. yarn.timeline-service.store-class org.apache.hadoop.yarn.server.timeline.LeveldbTimelineStore Store class name for timeline store. yarn.timeline-service.ttl-enable true Enable age off of timeline store data. yarn.timeline-service.ttl-ms 604800000 Time to live for timeline store data in milliseconds. yarn.timeline-service.leveldb-timeline-store.path ${hadoop.tmp.dir}/yarn/timeline Store file name for leveldb timeline store. yarn.timeline-service.leveldb-timeline-store.ttl-interval-ms 300000 Length of time to wait between deletion cycles of leveldb timeline store in milliseconds. yarn.timeline-service.leveldb-timeline-store.read-cache-size 104857600 Size of read cache for uncompressed blocks for leveldb timeline store in bytes. yarn.timeline-service.leveldb-timeline-store.start-time-read-cache-size 10000 Size of cache for recently read entity start times for leveldb timeline store in number of entities. yarn.timeline-service.leveldb-timeline-store.start-time-write-cache-size 10000 Size of cache for recently written entity start times for leveldb timeline store in number of entities. yarn.timeline-service.handler-thread-count 10 Handler thread count to serve the client RPC requests. yarn.timeline-service.http-authentication.type simple Defines authentication used for the timeline server HTTP endpoint. Supported values are: simple | kerberos | #AUTHENTICATION_HANDLER_CLASSNAME# yarn.timeline-service.http-authentication.simple.anonymous.allowed true Indicates if anonymous requests are allowed by the timeline server when using 'simple' authentication. yarn.timeline-service.principal The Kerberos principal for the timeline server. yarn.timeline-service.keytab /etc/krb5.keytab The Kerberos keytab for the timeline server. yarn.timeline-service.ui-names Comma separated list of UIs that will be hosted yarn.timeline-service.client.max-retries 30 Default maximum number of retries for timeline service client and value -1 means no limit. yarn.timeline-service.client.best-effort false Client policy for whether timeline operations are non-fatal. Should the failure to obtain a delegation token be considered an application failure (option = false), or should the client attempt to continue to publish information without it (option=true) yarn.timeline-service.client.retry-interval-ms 1000 Default retry time interval for timeline servive client. yarn.timeline-service.client.drain-entities.timeout.ms 2000 The time period for which timeline v2 client will wait for draining leftover entities after stop. yarn.timeline-service.recovery.enabled false Enable timeline server to recover state after starting. If true, then yarn.timeline-service.state-store-class must be specified. yarn.timeline-service.state-store-class org.apache.hadoop.yarn.server.timeline.recovery.LeveldbTimelineStateStore Store class name for timeline state store. yarn.timeline-service.leveldb-state-store.path ${hadoop.tmp.dir}/yarn/timeline Store file name for leveldb state store. yarn.timeline-service.entity-group-fs-store.cache-store-class org.apache.hadoop.yarn.server.timeline.MemoryTimelineStore Caching storage timeline server v1.5 is using. yarn.timeline-service.entity-group-fs-store.active-dir /tmp/entity-file-history/active HDFS path to store active application’s timeline data yarn.timeline-service.entity-group-fs-store.done-dir /tmp/entity-file-history/done/ HDFS path to store done application’s timeline data yarn.timeline-service.entity-group-fs-store.group-id-plugin-classes Plugins that can translate a timeline entity read request into a list of timeline entity group ids, separated by commas. yarn.timeline-service.entity-group-fs-store.group-id-plugin-classpath Classpath for all plugins defined in yarn.timeline-service.entity-group-fs-store.group-id-plugin-classes. yarn.timeline-service.entity-group-fs-store.summary-store org.apache.hadoop.yarn.server.timeline.LeveldbTimelineStore Summary storage for ATS v1.5 yarn.timeline-service.entity-group-fs-store.scan-interval-seconds 60 Scan interval for ATS v1.5 entity group file system storage reader.This value controls how frequent the reader will scan the HDFS active directory for application status. yarn.timeline-service.entity-group-fs-store.cleaner-interval-seconds 3600 Scan interval for ATS v1.5 entity group file system storage cleaner.This value controls how frequent the reader will scan the HDFS done directory for stale application data. yarn.timeline-service.entity-group-fs-store.retain-seconds 604800 How long the ATS v1.5 entity group file system storage will keep an application's data in the done directory. yarn.timeline-service.entity-group-fs-store.leveldb-cache-read-cache-size 10485760 Read cache size for the leveldb cache storage in ATS v1.5 plugin storage. yarn.timeline-service.entity-group-fs-store.app-cache-size 10 Size of the reader cache for ATS v1.5 reader. This value controls how many entity groups the ATS v1.5 server should cache. If the number of active read entity groups is greater than the number of caches items, some reads may return empty data. This value must be greater than 0. yarn.timeline-service.client.fd-flush-interval-secs 10 Flush interval for ATS v1.5 writer. This value controls how frequent the writer will flush the HDFS FSStream for the entity/domain. yarn.timeline-service.client.fd-clean-interval-secs 60 Scan interval for ATS v1.5 writer. This value controls how frequent the writer will scan the HDFS FSStream for the entity/domain. If the FSStream is stale for a long time, this FSStream will be close. yarn.timeline-service.client.fd-retain-secs 300 How long the ATS v1.5 writer will keep an FSStream open. If this fsstream does not write anything for this configured time, it will be close. yarn.timeline-service.writer.class org.apache.hadoop.yarn.server.timelineservice.storage.HBaseTimelineWriterImpl Storage implementation ATS v2 will use for the TimelineWriter service. yarn.timeline-service.reader.class org.apache.hadoop.yarn.server.timelineservice.storage.HBaseTimelineReaderImpl Storage implementation ATS v2 will use for the TimelineReader service. yarn.timeline-service.client.internal-timers-ttl-secs 420 How long the internal Timer Tasks can be alive in writer. If there is no write operation for this configured time, the internal timer tasks will be close. yarn.timeline-service.writer.flush-interval-seconds 60 The setting that controls how often the timeline collector flushes the timeline writer. yarn.timeline-service.app-collector.linger-period.ms 1000 Time period till which the application collector will be alive in NM, after the application master container finishes. yarn.timeline-service.timeline-client.number-of-async-entities-to-merge 10 Time line V2 client tries to merge these many number of async entities (if available) and then call the REST ATS V2 API to submit. yarn.timeline-service.hbase.coprocessor.app-final-value-retention-milliseconds 259200000 The setting that controls how long the final value of a metric of a completed app is retained before merging into the flow sum. Up to this time after an application is completed out-of-order values that arrive can be recognized and discarded at the cost of increased storage. yarn.timeline-service.hbase.coprocessor.jar.hdfs.location /hbase/coprocessor/hadoop-yarn-server-timelineservice.jar The default hdfs location for flowrun coprocessor jar. yarn.timeline-service.hbase-schema.prefix prod. The value of this parameter sets the prefix for all tables that are part of timeline service in the hbase storage schema. It can be set to "dev." or "staging." if it is to be used for development or staging instances. This way the data in production tables stays in a separate set of tables prefixed by "prod.". yarn.timeline-service.hbase.configuration.file Optional URL to an hbase-site.xml configuration file to be used to connect to the timeline-service hbase cluster. If empty or not specified, then the HBase configuration will be loaded from the classpath. When specified the values in the specified configuration file will override those from the ones that are present on the classpath. yarn.sharedcache.enabled false Whether the shared cache is enabled yarn.sharedcache.root-dir /sharedcache The root directory for the shared cache yarn.sharedcache.nested-level 3 The level of nested directories before getting to the checksum directories. It must be non-negative. yarn.sharedcache.store.class org.apache.hadoop.yarn.server.sharedcachemanager.store.InMemorySCMStore The implementation to be used for the SCM store yarn.sharedcache.app-checker.class org.apache.hadoop.yarn.server.sharedcachemanager.RemoteAppChecker The implementation to be used for the SCM app-checker yarn.sharedcache.store.in-memory.staleness-period-mins 10080 A resource in the in-memory store is considered stale if the time since the last reference exceeds the staleness period. This value is specified in minutes. yarn.sharedcache.store.in-memory.initial-delay-mins 10 Initial delay before the in-memory store runs its first check to remove dead initial applications. Specified in minutes. yarn.sharedcache.store.in-memory.check-period-mins 720 The frequency at which the in-memory store checks to remove dead initial applications. Specified in minutes. yarn.sharedcache.admin.address 0.0.0.0:8047 The address of the admin interface in the SCM (shared cache manager) yarn.sharedcache.admin.thread-count 1 The number of threads used to handle SCM admin interface (1 by default) yarn.sharedcache.webapp.address 0.0.0.0:8788 The address of the web application in the SCM (shared cache manager) yarn.sharedcache.cleaner.period-mins 1440 The frequency at which a cleaner task runs. Specified in minutes. yarn.sharedcache.cleaner.initial-delay-mins 10 Initial delay before the first cleaner task is scheduled. Specified in minutes. yarn.sharedcache.cleaner.resource-sleep-ms 0 The time to sleep between processing each shared cache resource. Specified in milliseconds. yarn.sharedcache.uploader.server.address 0.0.0.0:8046 The address of the node manager interface in the SCM (shared cache manager) yarn.sharedcache.uploader.server.thread-count 50 The number of threads used to handle shared cache manager requests from the node manager (50 by default) yarn.sharedcache.client-server.address 0.0.0.0:8045 The address of the client interface in the SCM (shared cache manager) yarn.sharedcache.client-server.thread-count 50 The number of threads used to handle shared cache manager requests from clients (50 by default) yarn.sharedcache.checksum.algo.impl org.apache.hadoop.yarn.sharedcache.ChecksumSHA256Impl The algorithm used to compute checksums of files (SHA-256 by default) yarn.sharedcache.nm.uploader.replication.factor 10 The replication factor for the node manager uploader for the shared cache (10 by default) yarn.sharedcache.nm.uploader.thread-count 20 The number of threads used to upload files from a node manager instance (20 by default) security.applicationhistory.protocol.acl ACL protocol for use in the Timeline server. yarn.is.minicluster false Set to true for MiniYARNCluster unit tests yarn.minicluster.control-resource-monitoring false Set for MiniYARNCluster unit tests to control resource monitoring yarn.minicluster.fixed.ports false Set to false in order to allow MiniYARNCluster to run tests without port conflicts. yarn.minicluster.use-rpc false Set to false in order to allow the NodeManager in MiniYARNCluster to use RPC to talk to the RM. yarn.minicluster.yarn.nodemanager.resource.memory-mb 4096 As yarn.nodemanager.resource.memory-mb property but for the NodeManager in a MiniYARNCluster. yarn.node-labels.enabled false Enable node labels feature yarn.node-labels.fs-store.root-dir URI for NodeLabelManager. The default value is /tmp/hadoop-yarn-${user}/node-labels/ in the local filesystem. yarn.node-labels.configuration-type centralized Set configuration type for node labels. Administrators can specify "centralized", "delegated-centralized" or "distributed". yarn.nodemanager.node-labels.provider When "yarn.node-labels.configuration-type" is configured with "distributed" in RM, Administrators can configure in NM the provider for the node labels by configuring this parameter. Administrators can configure "config", "script" or the class name of the provider. Configured class needs to extend org.apache.hadoop.yarn.server.nodemanager.nodelabels.NodeLabelsProvider. If "config" is configured, then "ConfigurationNodeLabelsProvider" and if "script" is configured, then "ScriptNodeLabelsProvider" will be used. yarn.nodemanager.node-labels.provider.fetch-interval-ms 600000 When "yarn.nodemanager.node-labels.provider" is configured with "config", "Script" or the configured class extends AbstractNodeLabelsProvider, then periodically node labels are retrieved from the node labels provider. This configuration is to define the interval period. If -1 is configured then node labels are retrieved from provider only during initialization. Defaults to 10 mins. yarn.nodemanager.node-labels.resync-interval-ms 120000 Interval at which NM syncs its node labels with RM. NM will send its loaded labels every x intervals configured, along with heartbeat to RM. yarn.nodemanager.node-labels.provider.configured-node-partition When "yarn.nodemanager.node-labels.provider" is configured with "config" then ConfigurationNodeLabelsProvider fetches the partition label from this parameter. yarn.nodemanager.node-labels.provider.fetch-timeout-ms 1200000 When "yarn.nodemanager.node-labels.provider" is configured with "Script" then this configuration provides the timeout period after which it will interrupt the script which queries the Node labels. Defaults to 20 mins. yarn.resourcemanager.node-labels.provider When node labels "yarn.node-labels.configuration-type" is of type "delegated-centralized", administrators should configure the class for fetching node labels by ResourceManager. Configured class needs to extend org.apache.hadoop.yarn.server.resourcemanager.nodelabels. RMNodeLabelsMappingProvider. yarn.resourcemanager.node-labels.provider.fetch-interval-ms 1800000 When "yarn.node-labels.configuration-type" is configured with "delegated-centralized", then periodically node labels are retrieved from the node labels provider. This configuration is to define the interval. If -1 is configured then node labels are retrieved from provider only once for each node after it registers. Defaults to 30 mins. yarn.resourcemanager.nodemanager-graceful-decommission-timeout-secs 3600 Timeout in seconds for YARN node graceful decommission. This is the maximal time to wait for running containers and applications to complete before transition a DECOMMISSIONING node into DECOMMISSIONED. yarn.resourcemanager.decommissioning-nodes-watcher.poll-interval-secs 20 Timeout in seconds of DecommissioningNodesWatcher internal polling. yarn.nodemanager.node-labels.provider.script.path The Node Label script to run. Script output Line starting with "NODE_PARTITION:" will be considered as Node Label Partition. In case of multiple lines have this pattern, then last one will be considered yarn.nodemanager.node-labels.provider.script.opts The arguments to pass to the Node label script. yarn.federation.enabled false Flag to indicate whether the RM is participating in Federation or not. yarn.federation.machine-list Machine list file to be loaded by the FederationSubCluster Resolver yarn.federation.subcluster-resolver.class org.apache.hadoop.yarn.server.federation.resolver.DefaultSubClusterResolverImpl Class name for SubClusterResolver yarn.federation.state-store.class org.apache.hadoop.yarn.server.federation.store.impl.MemoryFederationStateStore Store class name for federation state store yarn.federation.cache-ttl.secs 300 The time in seconds after which the federation state store local cache will be refreshed periodically yarn.client.application-client-protocol.poll-interval-ms 200 The interval that the yarn client library uses to poll the completion status of the asynchronous API of application client protocol. yarn.client.application-client-protocol.poll-timeout-ms -1 The duration (in ms) the YARN client waits for an expected state change to occur. -1 means unlimited wait time. yarn.nodemanager.container-monitor.procfs-tree.smaps-based-rss.enabled false RSS usage of a process computed via /proc/pid/stat is not very accurate as it includes shared pages of a process. /proc/pid/smaps provides useful information like Private_Dirty, Private_Clean, Shared_Dirty, Shared_Clean which can be used for computing more accurate RSS. When this flag is enabled, RSS is computed as Min(Shared_Dirty, Pss) + Private_Clean + Private_Dirty. It excludes read-only shared mappings in RSS computation. yarn.log.server.url URL for log aggregation server yarn.log.server.web-service.url URL for log aggregation server web service yarn.tracking.url.generator RM Application Tracking URL yarn.authorization-provider Class to be used for YarnAuthorizationProvider yarn.nodemanager.log-aggregation.roll-monitoring-interval-seconds -1 Defines how often NMs wake up to upload log files. The default value is -1. By default, the logs will be uploaded when the application is finished. By setting this configure, logs can be uploaded periodically when the application is running. The minimum rolling-interval-seconds can be set is 3600. yarn.intermediate-data-encryption.enable false Enable/disable intermediate-data encryption at YARN level. For now, this only is used by the FileSystemRMStateStore to setup right file-system security attributes. yarn.nodemanager.webapp.cross-origin.enabled false Flag to enable cross-origin (CORS) support in the NM. This flag requires the CORS filter initializer to be added to the filter initializers list in core-site.xml. yarn.cluster.max-application-priority 0 Defines maximum application priority in a cluster. If an application is submitted with a priority higher than this value, it will be reset to this maximum value. yarn.nodemanager.log-aggregation.policy.class org.apache.hadoop.yarn.server.nodemanager.containermanager.logaggregation.AllContainerLogAggregationPolicy The default log aggregation policy class. Applications can override it via LogAggregationContext. This configuration can provide some cluster-side default behavior so that if the application doesn't specify any policy via LogAggregationContext administrators of the cluster can adjust the policy globally. yarn.nodemanager.log-aggregation.policy.parameters The default parameters for the log aggregation policy. Applications can override it via LogAggregationContext. This configuration can provide some cluster-side default behavior so that if the application doesn't specify any policy via LogAggregationContext administrators of the cluster can adjust the policy globally. yarn.nodemanager.amrmproxy.enabled false Enable/Disable AMRMProxyService in the node manager. This service is used to intercept calls from the application masters to the resource manager. yarn.nodemanager.amrmproxy.address 0.0.0.0:8049 The address of the AMRMProxyService listener. yarn.nodemanager.amrmproxy.client.thread-count 25 The number of threads used to handle requests by the AMRMProxyService. yarn.nodemanager.amrmproxy.interceptor-class.pipeline org.apache.hadoop.yarn.server.nodemanager.amrmproxy.DefaultRequestInterceptor The comma separated list of class names that implement the RequestInterceptor interface. This is used by the AMRMProxyService to create the request processing pipeline for applications. yarn.nodemanager.distributed-scheduling.enabled false Setting that controls whether distributed scheduling is enabled. yarn.resourcemanager.opportunistic-container-allocation.enabled false Setting that controls whether opportunistic container allocation is enabled. yarn.resourcemanager.opportunistic-container-allocation.nodes-used 10 Number of nodes to be used by the Opportunistic Container Allocator for dispatching containers during container allocation. yarn.resourcemanager.nm-container-queuing.sorting-nodes-interval-ms 1000 Frequency for computing least loaded NMs. yarn.resourcemanager.nm-container-queuing.load-comparator QUEUE_LENGTH Comparator for determining node load for Distributed Scheduling. yarn.resourcemanager.nm-container-queuing.queue-limit-stdev 1.0f Value of standard deviation used for calculation of queue limit thresholds. yarn.resourcemanager.nm-container-queuing.min-queue-length 5 Min length of container queue at NodeManager. yarn.resourcemanager.nm-container-queuing.max-queue-length 15 Max length of container queue at NodeManager. yarn.resourcemanager.nm-container-queuing.min-queue-wait-time-ms 10 Min queue wait time for a container at a NodeManager. yarn.resourcemanager.nm-container-queuing.max-queue-wait-time-ms 100 Max queue wait time for a container queue at a NodeManager. yarn.nodemanager.opportunistic-containers-use-pause-for-preemption false Use container pause as the preemption policy over kill in the container queue at a NodeManager. yarn.nodemanager.container.stderr.pattern {*stderr*,*STDERR*} Error filename pattern, to identify the file in the container's Log directory which contain the container's error log. As error file redirection is done by client/AM and yarn will not be aware of the error file name. YARN uses this pattern to identify the error file and tail the error log as diagnostics when the container execution returns non zero value. Filename patterns are case sensitive and should match the specifications of FileSystem.globStatus(Path) api. If multiple filenames matches the pattern, first file matching the pattern will be picked. yarn.nodemanager.container.stderr.tail.bytes 4096 Size of the container error file which needs to be tailed, in bytes. yarn.node-labels.fs-store.impl.class org.apache.hadoop.yarn.nodelabels.FileSystemNodeLabelsStore Choose different implementation of node label's storage yarn.resourcemanager.webapp.rest-csrf.enabled false Enable the CSRF filter for the RM web app yarn.resourcemanager.webapp.rest-csrf.custom-header X-XSRF-Header Optional parameter that indicates the custom header name to use for CSRF protection. yarn.resourcemanager.webapp.rest-csrf.methods-to-ignore GET,OPTIONS,HEAD Optional parameter that indicates the list of HTTP methods that do not require CSRF protection yarn.nodemanager.webapp.rest-csrf.enabled false Enable the CSRF filter for the NM web app yarn.nodemanager.webapp.rest-csrf.custom-header X-XSRF-Header Optional parameter that indicates the custom header name to use for CSRF protection. yarn.nodemanager.webapp.rest-csrf.methods-to-ignore GET,OPTIONS,HEAD Optional parameter that indicates the list of HTTP methods that do not require CSRF protection yarn.nodemanager.disk-validator basic The name of disk validator. yarn.timeline-service.webapp.rest-csrf.enabled false Enable the CSRF filter for the timeline service web app yarn.timeline-service.webapp.rest-csrf.custom-header X-XSRF-Header Optional parameter that indicates the custom header name to use for CSRF protection. yarn.timeline-service.webapp.rest-csrf.methods-to-ignore GET,OPTIONS,HEAD Optional parameter that indicates the list of HTTP methods that do not require CSRF protection yarn.webapp.xfs-filter.enabled true Enable the XFS filter for YARN yarn.resourcemanager.webapp.xfs-filter.xframe-options SAMEORIGIN Property specifying the xframe options value. yarn.nodemanager.webapp.xfs-filter.xframe-options SAMEORIGIN Property specifying the xframe options value. yarn.timeline-service.webapp.xfs-filter.xframe-options SAMEORIGIN Property specifying the xframe options value. yarn.resourcemanager.node-removal-untracked.timeout-ms 60000 The least amount of time(msec.) an inactive (decommissioned or shutdown) node can stay in the nodes list of the resourcemanager after being declared untracked. A node is marked untracked if and only if it is absent from both include and exclude nodemanager lists on the RM. All inactive nodes are checked twice per timeout interval or every 10 minutes, whichever is lesser, and marked appropriately. The same is done when refreshNodes command (graceful or otherwise) is invoked. yarn.resourcemanager.application-timeouts.monitor.interval-ms 3000 The RMAppLifetimeMonitor Service uses this value as monitor interval yarn.app.attempt.diagnostics.limit.kc 64 Defines the limit of the diagnostics message of an application attempt, in kilo characters (character count * 1024). When using ZooKeeper to store application state behavior, it's important to limit the size of the diagnostic messages to prevent YARN from overwhelming ZooKeeper. In cases where yarn.resourcemanager.state-store.max-completed-applications is set to a large number, it may be desirable to reduce the value of this property to limit the total data stored. yarn.timeline-service.http-cross-origin.enabled false Flag to enable cross-origin (CORS) support for timeline service v1.x or Timeline Reader in timeline service v2. For timeline service v2, also add org.apache.hadoop.security.HttpCrossOriginFilterInitializer to the configuration hadoop.http.filter.initializers in core-site.xml. yarn.timeline-service.http-cross-origin.enabled false Flag to enable cross-origin (CORS) support for timeline service v1.x or Timeline Reader in timeline service v2. For timeline service v2, also add org.apache.hadoop.security.HttpCrossOriginFilterInitializer to the configuration hadoop.http.filter.initializers in core-site.xml. yarn.router.clientrm.interceptor-class.pipeline org.apache.hadoop.yarn.server.router.clientrm.DefaultClientRequestInterceptor The comma separated list of class names that implement the RequestInterceptor interface. This is used by the RouterClientRMService to create the request processing pipeline for users. yarn.router.pipeline.cache-max-size 25 Size of LRU cache for Router ClientRM Service and RMAdmin Service. yarn.router.rmadmin.interceptor-class.pipeline org.apache.hadoop.yarn.server.router.rmadmin.DefaultRMAdminRequestInterceptor The comma separated list of class names that implement the RequestInterceptor interface. This is used by the RouterRMAdminService to create the request processing pipeline for users. yarn.router.bind-host The actual address the server will bind to. If this optional address is set, the RPC and webapp servers will bind to this address and the port specified in yarn.router.address and yarn.router.webapp.address, respectively. This is most useful for making Router listen to all interfaces by setting to 0.0.0.0. yarn.scheduler.queue-placement-rules user-group Comma-separated list of PlacementRules to determine how applications submitted by certain users get mapped to certain queues. Default is user-group, which corresponds to UserGroupMappingPlacementRule. yarn.router.webapp.interceptor-class.pipeline org.apache.hadoop.yarn.server.router.webapp.DefaultRequestInterceptorREST The comma separated list of class names that implement the RequestInterceptor interface. This is used by the RouterWebServices to create the request processing pipeline for users. yarn.router.webapp.address 0.0.0.0:8089 The http address of the Router web application. If only a host is provided as the value, the webapp will be served on a random port. yarn.router.webapp.https.address 0.0.0.0:8091 The https address of the Router web application. If only a host is provided as the value, the webapp will be served on a random port. yarn.timeline-service.entity-group-fs-store.with-user-dir false It is TimelineClient 1.5 configuration whether to store active application's timeline data with in user directory i.e ${yarn.timeline-service.entity-group-fs-store.active-dir}/${user.name} yarn.resourcemanager.display.per-user-apps false Flag to enable display of applications per user as an admin configuration. yarn.scheduler.configuration.store.class file The type of configuration store to use for scheduler configurations. Default is "file", which uses file based capacity-scheduler.xml to retrieve and change scheduler configuration. To enable API based scheduler configuration, use either "memory" (in memory storage, no persistence across restarts), "leveldb" (leveldb based storage), or "zk" (zookeeper based storage). API based configuration is only useful when using a scheduler which supports mutable configuration. Currently only capacity scheduler supports this. yarn.scheduler.configuration.mutation.acl-policy.class org.apache.hadoop.yarn.server.resourcemanager.scheduler.DefaultConfigurationMutationACLPolicy The class to use for configuration mutation ACL policy if using a mutable configuration provider. Controls whether a mutation request is allowed. The DefaultConfigurationMutationACLPolicy checks if the requestor is a YARN admin. yarn.scheduler.configuration.leveldb-store.path ${hadoop.tmp.dir}/yarn/system/confstore The storage path for LevelDB implementation of configuration store, when yarn.scheduler.configuration.store.class is configured to be "leveldb". yarn.scheduler.configuration.leveldb-store.compaction-interval-secs 86400 The compaction interval for LevelDB configuration store in secs, when yarn.scheduler.configuration.store.class is configured to be "leveldb". Default is one day. yarn.scheduler.configuration.store.max-logs 1000 The max number of configuration change log entries kept in config store, when yarn.scheduler.configuration.store.class is configured to be "leveldb" or "zk". Default is 1000 for either. yarn.scheduler.configuration.zk-store.parent-path /confstore ZK root node path for configuration store when using zookeeper-based configuration store. yarn.resource-types The resource types to be used for scheduling. Use resource-types.xml to specify details about the individual resource types. yarn.client.load.resource-types.from-server false Provides an option for client to load supported resource types from RM instead of depending on local resource-types.xml file.