HowToRelease

This page is prepared for Hadoop Core committers. You need committer rights to create a new Hadoop Core release.

These instructions have been updated to use dev-support/bin/create-release. Earlier versions of this document are at HowToReleaseWithSvnAndAnt and HowToReleasePostMavenization and HowToReleasePreDSBCR. For releasing from the 2.6.x or the 2.7.x line, you'll need to consult HowToReleasePreDSBCR to find applicable steps.

Applicable Hadoop versions: 2.8.0 and above; 3.0.0 and above.

READ ALL OF THESE INSTRUCTIONS THOROUGHLY BEFORE PROCEEDING!

Preparation

1. If you have not already done so, append your code signing key to the KEYS file. Once you commit your changes, they will automatically be propagated to the website. Also upload your key to a public key server if you haven't. End users use the KEYS file (along with the web of trust) to validate that releases were done by an Apache committer. For more details on signing releases, see Signing Releases and Step-By-Step Guide to Mirroring Releases. Make sure you have PGP 2.2 installed. PGP 1 and 2.0/2.1 doesn't work properly with the PGP 2.2 installed in the docker image.
2. Make sure to update the various LICENSE and NOTICE files per Apache policy.
3. Bulk update JIRA to unassign from this release all issues that are open non-blockers. This is involved since you can only bulk change issues within the same project, so minimally requires four bulk changes for each of HADOOP, HDFS, MAPREDUCE, and YARN. Editing the "Target Version/s" field is also a blind write, so you need to be careful not to lose any other fix versions that are set. For updating 3.0.0-beta1 to 3.0.0, the process looked like this:
    a. Start with this query:

    ```
    project in (HADOOP, HDFS, YARN, MAPREDUCE) AND "Target Version/s" = 3.0.0-beta1 and statusCategory
    != Done
    ```

    b. Filter this list down until it's only issues with a Target Version of just "3.0.0-beta1". My query ended up looking like:

    ```
    project in (HADOOP, HDFS, YARN, MAPREDUCE) AND "Target Version/s" = 3.0.0-beta1 and "Target
    Versions/" not in (2.9.0, 2.8.3, 2.8.2) AND statusCategory != Done
    ```

    c. Do the bulk update for each project individually to set the target version to 3.0.0.
    d. Check the query for the next most common set of target versions and again filter it down:

    ```
    project in (HADOOP, HDFS, YARN, MAPREDUCE) AND "Target Version/s" = 3.0.0-beta1 and "Target Version
    /s" = 2.9.0 and statusCategory != Done
    project in (HADOOP, HDFS, YARN, MAPREDUCE) AND "Target Version/s" = 3.0.0-beta1 and "Target Version
    /s" = 2.9.0 and "Target Version/s" not in (2.8.2, 2.8.3) and statusCategory != Done
    ```

    e. Do the bulk update for each project individually to set the target version field to (3.0.0, 2.9.0).
    f. Return to the original query. If there aren't too many, update the remaining straggler issues by hand (faster than doing the bulk edits):

    ```
    project in (HADOOP, HDFS, YARN, MAPREDUCE) AND "Target Version/s" = 3.0.0-beta1 and statusCategory
    != Done
    ```

4. Send follow-up notification to the developer list that this was done.
5. To deploy artifacts to the Apache Maven repository create `~/.m2/settings.xml`:

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
                      http://maven.apache.org/xsd/settings-1.0.0.xsd">
  <servers>
    <server>
     <id>apache.staging.https</id>
     <username>Apache username</username>
     <password>Apache password</password>
    </server>
  </servers>
</settings>
```

Branching

When releasing Hadoop X.Y.Z, the following branching changes are required. Note that a release can match more than one of the following if-conditions. For a major release, one needs to make the changes for minor and point releases as well. Similarly, a new minor release is also a new point release.

1. If this is a new major release (i.e., Y = 0 and Z = 0)
    a. Create a new branch (branch-X) for all releases in this major release.
    b. Update the version on trunk to (X+1).0.0-SNAPSHOT

    ```
    mvn versions:set -DnewVersion=(X+1).0.0-SNAPSHOT
    ```

    c. Set `hadoop.version` in the root `pom.xml` file to the same value; validate with a clean build.
    d. Commit the version change to trunk.

    ```
    git commit -a -m "Preparing for (X+1).0.0 development"
    ```

2. If this is a new minor release (i.e., Z = 0)
    a. Create a new branch (branch-X.Y) for all releases in this minor release.
    b. Update the version on branch-X to X.(Y+1).0-SNAPSHOT

    ```
    mvn versions:set -DnewVersion=X.(Y+1).0-SNAPSHOT
    ```

    c. Set `hadoop.version` in the root `pom.xml` file to the same value; validate with a clean build.
    d. Commit the version change to branch-X.

    ```
    git commit -a -m "Preparing for X.(Y+1).0 development"
    ```

3. If this is a new point release (i.e., always)
    a. Create a new branch (branch-X.Y.Z) for this release.
    b. Update the version on branch-X.Y to X.Y.(Z+1)-SNAPSHOT

    ```
    mvn versions:set -DnewVersion=X.Y.(Z+1)-SNAPSHOT
    ```

    c. Set `hadoop.version` in the root `pom.xml` file to the same value; validate with a clean build.
    d. Commit the version change to branch-X.Y.

    ```
    git commit -a -m "Preparing for X.Y.(Z+1) development"
    ```

4. Release branch (branch-X.Y.Z) updates:
    a. Update `hadoop-project/src/site/markdown/index.md.vm` to reflect the right versions, new features and big improvements.
    b. Update the version on branch-X.Y.Z TO X.Y.Z

    ```
    mvn versions:set -DnewVersion=X.Y.Z
    ```

Note: Please also also update the hadoop.version property in the root pom.xml (see HADOOP-15369) and for releases off branches earlier than 3.3.1, the hadoop.assemblies.version in hadoop-project/pom.xml

```
mvn versions:set-property -Dproperty=hadoop.version -DnewVersion=X.Y.Z

mvn versions:set-property -Dproperty=hadoop.assemblies.version -DnewVersion=X.Y.Z
```

(The hadoop.assemblies.version update isn't needed on recent releases; since HADOOP-17663 the command to set it will not touch any files, and so completely harmless)

Now, for any branches in {trunk, branch-X, branch-X.Y, branch-X.Y.Z} that have changed, push them to the remote repo taking care of any conflicts.

```
git push <remote> <branch>
```

Creating the release candidate (X.Y.Z-RC<N>)

These steps need to be performed to create the _N_th RC for X.Y.Z, where *N* starts from 0.

1. Check if the release year for Web UI footer is updated (the property `<release-year>` in `hadoop-project/pom.xml`). If not, create a JIRA to update the property value to the right year, and propagate the fix from trunk to all necessary branches. Consider the voting time needed before publishing, it's better to use the year of (current time + voting time) here, to be consistent with the publishing time.
2. In JIRA, ensure that only issues in the "Fixed" state have a "Fix Version" set to release X.Y.Z.
3. Verify that $HOME/.gpg defaults to the key listed in the KEYS file.
4. For the Apache release, a machine capable of running Docker- and Internet- capable, build the release candidate with `create-release`. Unless the `--logdir` is given, logs will be in the `patchprocess/` directory. Artifacts will be in the target/artifacts NOTE: This will take quite a while, since it will download and build the entire source tree, including documentation and native components, from scratch to avoid maven repository caching issues hiding issues with the source release.

   For x86 architecture CPU, we can use the following command:

   ```
   dev-support/bin/create-release --asfrelease --docker --dockercache
   ```

   Note:

   if we encounter a 401 permission issue, it may be due to the local Maven configuration not being read. We can try the following command.

   ```
   dev-support/bin/create-release --asfrelease --docker --dockercache --mvnargs="-Duser.home=<mvn setting.
   xml path>"
   ```

   For Arm architecture CPU, we can use the following command:

   ```
   dev-support/bin/create-release --docker --dockercache --mvnargs="-Dhttp.keepAlive=false -Dmaven.wagon.
   http.pool=false" --deploy --native --sign
   ```

   Note:

   For compiling on ARM architecture CPU, we can refer to the https://github.com/apache/hadoop-release-support project.
5. While it should fail `create-release` if there are issues, doublecheck the rat log to find and fix any potential licensing issues.

   ```
   grep 'Rat check' patchprocess/mvn_apache_rat.log
   ```

6. Check that release files look ok - e.g. install it somewhere fresh and run examples from tutorial, do a fresh build, read the release notes looking for WARNINGs, etc.
7. Set environment variable version for later steps. `export version=X.Y.Z-RCN`
8. Tag the release candidate:

   ```
   git tag -s release-$version -m "Release candidate - $version"
   ```

9. Push branch-X.Y.Z and the newly created tag to the remote repo.
10. (This is for branch-2.10 only. `mvn deploy` is invoked by the dev-support/bin/create-release script of 3.0.0 and above. see HADOOP-15058.) Deploy the maven artifacts, on your personal computer. Please be sure you have completed the prerequisite step of preparing the `settings.xml` file before the deployment. You might want to do this in private and clear your history file as your gpg-passphrase is in clear text.

```
mvn deploy -Psign,dist -DskipTests
```

11. Copy release files to a public place and ensure they are readable. Note that `home.apache.org` only supports SFTP, so this may be easier with a graphical SFTP client like Nautilus, Konqueror, etc.

```
sftp home.apache.org
> cd public_html
> mkdir hadoop-${version}
> put -r /home/hadoop/hadoop-${version}
....
> bye
```

12. Log into Nexus, select "`Staging` Repositories" from the left navigation pane, select the check-box against the specific hadoop repository, and `close` the release. At the time of deploying the maven artifacts, if there are different hadoop items of this release candidate at “Staging Repositories”, drop the stale RC first.
13. Call a release vote on common-dev at hadoop.apache.org. It's usually a good idea to start the release vote on Monday so that people will have a chance to verify the release candidate during the week. Example
14. If the release candidate contains a serious issue, withdraw the vote, make necessary changes, and repeat this process.
15. If non-trivial changes are committed to the release branch, ensure the commits are present in the upstream branches.

Publishing

In 5 days if the release vote passes, the release may be published.

1. In JIRA, "release" the version, setting the date to the end-of-vote date. Visit the "Administer Project" page, then the "Manage versions" page. You need to have the "Admin" role in HADOOP, HDFS, MAPREDUCE, and YARN.
2. Set environment variable version for later steps. `export version=X.Y.Z`
3. Tag the release. Do it from the release branch and push the created tag to the remote repository:

```
git tag -s rel/release-${version} -m "Hadoop ${version} release"
git push origin rel/release-${version}
```

4. Copy release files to the distribution directory
    a. Check out the corresponding svn repo if need be

    ```
    svn co https://dist.apache.org/repos/dist/release/hadoop/common/ hadoop-dist
    ```

    b. Copy the release files to hadoop-dist/hadoop-${version}
    c. Update the symlinks to current2 and stable2. The release directory usually contains just two releases, the most recent from two branches.
    d. Commit the changes (it requires a PMC privilege)

    ```
    svn ci -m "Publishing the bits for release ${version}"
    ```

    e. Usually binary tarball becomes larger than 300MB, so it cannot be directly uploaded to the distribution directory. Use the dev directory (https://dist.apache.org/repos/dist/dev/hadoop/) first and then move it to the distribution directory by `svn move`.
5. Update upstream branches to make them aware of this new release:
    a. Copy and commit the CHANGELOG.md and RELEASENOTES.md:

    ```
     cp target/artifacts/RELEASENOTES.md hadoop-common-project/hadoop-common/src/site/markdown/release
    /${version}/RELEASENOTES.${version}.md
     cp target/artifacts/CHANGELOG.md hadoop-common-project/hadoop-common/src/site/markdown/release
    /${version}/CHANGELOG.${version}.md
    ```

    b. Copy the jdiff xml files for this version to their appropriate directory.

```
  cp hadoop-common-project/hadoop-common/target/site/jdiff/xml/Apache_Hadoop_Common_${version}.xml
hadoop-common-project/hadoop-common/dev-support/jdiff
  cp hadoop-hdfs-project/hadoop-hdfs/target/site/jdiff/xml/Apache_Hadoop_HDFS_${version}.xml
hadoop-hdfs-project/hadoop-hdfs/dev-support/jdiff
  find hadoop-yarn-project -name "Apache_Hadoop_YARN_*_${version}.xml" | xargs -I{} cp {} hadoop-
yarn-project/hadoop-yarn/dev-support/jdiff
  find hadoop-mapreduce-project -name "Apache_Hadoop_MapReduce_*_${version}.xml" | xargs -I{} cp
{} hadoop-mapreduce-project/dev-support/jdiff
```

c. Update `hadoop-project-dist/pom.xml`

```
 <jdiff.stable.api>X.Y.Z</jdiff.stable.api>
```

6. In Nexus
   a. effect the release of artifacts by selecting the staged repository and then clicking `Release`
   b. If there were multiple RCs, simply drop the staging repositories corresponding to failed RCs.
7. Wait 24 hours for release to propagate to mirrors.
8. Edit the website (Generic docs about the new website generation can be found [here|How+to+generate+and+push+ASF+web+site)
   a. Checkout the website if you haven't already

```
git clone https://gitbox.apache.org/repos/asf/hadoop-site.git -b asf-site
```

   b. Install hugo if you haven't already ((tldr; apt-get install/pacman -S/brew install hugo))
   c. Create the new release announcement (Set environment variable version first. `export VERSION=X.Y.Z).`

```
cat << EOF > src/release/${VERSION}.md
---
title: Release ${VERSION} available
date: 202X-XX-XX
linked: true
---
<!---
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

   http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

This is the first stable release of Apache Hadoop TODO line.

It contains TODO bug fixes, improvements and enhancements since TODO.

Users are encouraged to read the [overview of major changes][1] since TODO.
For details of TODO bug fixes, improvements, and other enhancements since the previous TODO
release,
please check [release notes][2] and [changelog][3].

[1]: /docs/r${VERSION}/index.html
[2]: http://hadoop.apache.org/docs/r${VERSION}/hadoop-project-dist/hadoop-common/release/${VERSION}
/RELEASENOTES.${VERSION}.html
[3]: http://hadoop.apache.org/docs/r${VERSION}/hadoop-project-dist/hadoop-common/release/${VERSION}
/CHANGELOG.${VERSION}.html

EOF
```

   d. Note: update all the TODO + the date. **Don't use date from the future**, it won't be rendered.

e. Remove the `linked: true` line from the previous release file, eg. from src/release/3.0.0.md. Docs/downloads of the releases with `linked:true` will be linked from the menu.
f. add the docs and update the `content/docs/current` link, by doing the following:

```
cd content/docs
tar xvf /path/to/hadoop-${version}-site.tar.gz
# Update current2, current, stable and stable2 as needed.
# For example
rm current2 current
ln -s r${version} current2
ln -s current2 current
```

g. Similarly update the symlinks for stable if need be.
h. Check the rendering of the new site: `hugo serve && firefox http://localhost:1313`
i. Regenerate the site, review it, then commit it per the instructions in HowToCommit. (The generated HTML files also should be committed. Both src and the rendered site are in the same repo.)

```
hugo
git add .
git commit
git push
```

9. Send announcements to announce@apache.org and to the user and developer lists once the site changes are visible. Optionally on Hadoop-PMC owned twitter account @hadoop. (For password reach out to private@)
10. ~~In JIRA, close issues resolved in the release. Disable mail notifications for this bulk change.~~ Recommend **not** closing, since it prevents JIRAs from being edited and makes it more difficult to track backports.
11. Add the release in "Apache Committee Report Helper" for the next board report to pick that up automatically at https://reporter.apache.org/addrelease.html?hadoop (Requires PMC member rights)
12. Update the DOAP file with the release details, DOAP File. The changes will be reflected here after almost 1 hour, if done correctly.

Docker images

Docker images containing release binaries are accessible at DockerHub. A docker image is automatically built by pushing a commit to one of the two special branches: docker-hadoop-2 and docker-hadoop-3. An example is HADOOP-18681. Please contact Ayush Saxena and Wei-Chiu Chuang who have admin privilege to the Apache Hadoop DockerHub repo and can add additional tags to the docker image produced.

The docker images can be downloaded using docker command:

```
docker pull apache/hadoop:3.3.5
```

Hadoop-thirdparty

The hadoop-thirdparty repository shades a number of thirdparty dependencies that are used by Hadoop. They are shaded to avoid classpath conflict with downstream applications. The Hadoop project may occasionally need to update and release hadoop-thirdparty artifacts. The steps are largely similar to the above. See the How To Release Hadoop-Thirdparty wiki for details.

See Also

- Apache Releases FAQ