Release Procedure

This page describes the steps that a release manager needs to take to perform a release of Apache CloudStack.

(Thanks to the couchdb project for a great template to use for this procedural document!)

Checklist

1. Ensure that you are working on the correct branch. The CloudStack community has adopted a branch-before-release strategy for release branches.
2. Confirm that the information in README.md is current
3. Confirm that the information in INSTALL.md is current
4. Update the CHANGES file with the critical changes introduced in the release (Be sure to highlight any changes the break backward compatibility)
5. Remove "version has not been released" warnings from the CHANGES file
6. Ensure that the CHANGES file in your release branch is synced with the version in the main branch (which may have a future release already in it)
7. Confirm that there is a Jenkins build process for the release branch and that all associated jobs are succeeding.
8. You'll need your GPG public key to be added to the CloudStack KEYS file (see *Appendix A* at the end of the article on how to do that).

Getting Community Consensus

Prior to an official vote, start a thread on the cloudstack-dev mailing list, specifically asking for comments on the project's readiness to cut a release.

Once it appears that any outstanding blockers have been addressed, you can proceed to the next step.

Preparing a Release

Update your local git repo from the ASF repository:

```
git fetch origin
```

Check out the release branch:

```
git checkout X.Y
```

(or if you haven't already checked it out locally with remote tracking enabled: git checkout -b X.Y origin/X.Y)

Make sure your local copy exactly matches the remote repo:

```
git reset --hard HEAD
git clean -qfxd
git rebase origin/X.Y
```

Then run the source build script (Replacing the parameters: X.Y.Z.0=your official version number for the release; X.Y=the branch (can be main) that the release is coming from; CCCC=the GPG Key to sign both the artifacts and the git tag with):

```
tools/build/build_asf.sh -v X.Y.Z.0 -b X.Y -t -u CCCC -s /path/to/the/source/dir -c
```

(optionally specifying your local directory layout - see build_asf.sh -h for details)

This will automatically commit (the -c) to the cloudstack dist dev folder (on SVN). This is the staging area for Apache CloudStack distributions.

Get the commit-sh to vote against for your VOTE email. This comes from the output of the command above. Example:

```
completed.  use commit-sh of b25d27d80b62de3408041821aa99e68712ae2728 when starting the VOTE thread
```

An RC branch will have been created. When a vote is done you can either delete it or merge it back depending on the outcome.

Test the Build

Follow the instructions documented in your release branch's test procedures wiki page.

If your personal tests pass, you are ready to propose the release to the community.

Managing the Vote(s)

dev@cloudstack.apache.org / users@cloudstack.apache.org VOTE

Email the dev@cloudstack.apache.org and users@cloudstack.apache.org mailing list, using the following template:

SUBJECT: [VOTE] Apache Cloudstack X.Y.Z.0

MESSAGE:

```
Hi All,

I've created a X.Y.Z.0 release, with the following artifacts up for a vote:

Git Branch and Commit SH:
https://git-wip-us.apache.org/repos/asf?p=cloudstack.git;a=shortlog;h=refs/heads/X.Y
Commit: XXXXXXXXXXXXXXXXX

Source release (checksums and signatures are available at the same
location):
https://dist.apache.org/repos/dist/dev/cloudstack/X.Y/

PGP release keys (signed using XXXXXXXX):
https://dist.apache.org/repos/dist/release/cloudstack/KEYS

Vote will be open for 72 hours.

For sanity in tallying the vote, can PMC members please be sure to indicate "(binding)" with their vote?

[ ] +1  approve
[ ] +0  no opinion
[ ] -1  disapprove (and reason why)
```

dev@cloudstack.apache.org / user@cloudstack.apache.org Voting Results

After 72 hours, the vote can be closed.

If (after tallying the vote) the binding +1 votes are not in the majority, the issues noted need to be addressed and process starts again. You need to send out a results email (or a less formal abort email) for the VOTE thread.

If the vote passes, then send a [RESULTS] vote to the dev list. Template below:

SUBJECT: [RESULT][VOTE] Apache CloudStack X.Y.Z.0

MESSAGE:

```
Hi all,

After 72 hours, the vote for CloudStack X.Y.Z.0 *passes* with
Z PMC + Z non-PMC votes.

+1 (PMC / binding)
* person

+1 (non binding)
* person

0
none

-1
none

Thanks to everyone participating.

I will now prepare the release announcement to go out after 24 hours to give the mirrors time to catch up.
```

Publish the Release

Merge back the RC branch and set the HEAD version numbers to the next release number

Below are given instructions for both **new** (LTS/non-LTS) release and for the **maintenance** release, as the procedures are somewhat different - see the later examples for 4.14.0.0 and 4.13.1.0 releases

Merge and push the release candidate branch into the release branch, set the next POMs version, delete the RC branches

```
$ git checkout X.Y-RC$TIMESTAMP # checkout the RC branch
$ git tag X.Y.Z.0 # the tag has already been created by build_asf.sh
$ git push origin X.Y.Z.0 # push the tag to origin

# if Z=0 / i.e. NEW release (X.Y.0.0)
  # Rename the RC branch and set the next POMs version in this new branch
  $ git branch -m X.Y-RC$TIMESTAMP X.Y
  $ git push -u origin X.Y
  $ bash tools/build/setnextversion.sh -b X.Y -v X.Y.1.0-SNAPSHOT -s /root/cloudstack # replace with local
working directory
  $ grep -r "X.Y.0.0" . # doublecheck by grepping for the older POM version - should return a few git refs and
comments - not the POMs/code - and if all good push
  $ git push origin X.Y
  # Merge the new branch into the main AND set the nextversions in the main branch
  $ git checkout main
  $ git pull
  $ git merge X.Y
  $ bash tools/build/setnextversion.sh -b main -v X.Y+1.0.0-SNAPSHOT -s /root/cloudstack # replace with local
working directory
  $ grep -r "X.Y.1.0" . # doublecheck by grepping for the older POM version - should return a few git refs and
comments - not the POMs/code - and if all good push
  $ git push origin main

# else Z!=0 (LTS maintenance release, X.Y.Z.0)
  # Merge the RC branch into the release branch AND set the nextversions in the current X.Y branch
  $ git checkout X.Y
  $ git pull
  $ git merge X.Y-RC$TIMESTAMP
  $ bash tools/build/setnextversion.sh -b X.Y -v X.Y.Z+1.0-SNAPSHOT -s /root/cloudstack # replace with local
working directory
  $ grep -r "X.Y.Z.0" . # doublecheck by grepping for the older POM version - should return a few git refs and
comments - not the POMs/code - and if all good push
  $ git push origin X.Y

# end of IF/ELSE block, delete older RC branches (repeat for all RC branches)
  $ git branch -d X.Y-RC$TIMESTAMP # delete the RC branch locally
  $ git push origin :X.Y-RC$TIMESTAMP # deletes the RC branch on origin
```

Below are given specific example for 4.14.0.0 (new release) and 4.13.1.0 (maintenance release)

```
Example for 4.14.0.0 **new** release

$ git checkout 4.14.0.0-RC20200511T1503 # checkout the RC branch
$ git tag 4.14.0.0 # the tag has already been created by build_asf.sh
$ git push origin 4.14.0.0 # push the tag to origin
# Rename the RC branch and set the next version in this new branch
$ git branch -m 4.14.0.0-RC20200511T1503 4.14
$ git push -u origin 4.14
$ bash tools/build/setnextversion.sh -b 4.14 -v 4.14.1.0-SNAPSHOT -s /root/cloudstack # replace with local working directory
$ grep -r "4.14.0.0" . # doublecheck by grepping for the older POMs version - should return a few git refs and comments - not the POMs/code - and if all good push
$ git push origin 4.14
# Merge the new branch into the main AND set the nextversions in the main branch
$ git checkout main
$ git pull
$ git merge 4.14
$ bash tools/build/setnextversion.sh -b main -v 4.15.0.0-SNAPSHOT -s /root/cloudstack #replace with local working directory
$ grep -r "4.14.1.0" . # doublecheck by grepping for the older POM version - should return a few git refs and comments - not the POMs/code - and if all good push
$ git push origin main
# delete older RC branches (repeat for all RC branches)
$ git branch -d 4.14.0.0-RC20200511T1503 # delete the RC branch locally
$ git push origin :4.14.0.0-RC20200511T1503 # deletes the RC branch on origin

Example for 4.13.1.0 **maintenance** release

$ git checkout 4.13.1.0-RC20200423T1917 # checkout the RC branch
$ git tag 4.13.1.0 # the tag has already been created by build_asf.sh
$ git push origin 4.13.1.0 # push the tag to origin
# Merge 4.13.1.0-RC20200423T1917 into the 4.13 AND set the nextversions in the current/4.13 branch
$ git checkout 4.13
$ git pull
$ git merge 4.13.1.0-RC20200423T1917
$ bash tools/build/setnextversion.sh -b 4.13 -v 4.13.2.0-SNAPSHOT -s /root/cloudstack #replace with local working directory
$ grep -r "4.13.1.0" . # doublecheck by grepping for the older POM version - should return a few git refs and comments - not the POMs/code - and if all good push
$ git push origin 4.13
# delete older RC branches (repeat for all RC branches) - BUT better NOT delete the voted RC branch before generating the API doc - see below note on updating http://cloudstack.apache.org/
$ git branch -d 4.13.1.0-RC20200423T1917 # delete the RC branch locally
$ git push origin :4.13.1.0-RC20200423T1917 # deletes the RC branch on origin
```

Close the current GitHub milestone

Issues are tracked via GitHub. After release X.Y.Z.0 has been voted, close the corresponding milestone in GitHub and create a milestone for new versions as needed.

Move the release from "dev" to "release" on SVN repo

Move the voted release artefacts from "dev" to "release" (replace X.Y.Z.0 with the voted release number i.e. 4.14.0.0):

```
# svn mv -m "Publishing X.Y.Z.0 release" https://dist.apache.org/repos/dist/dev/cloudstack/X.Y.Z.0/ https://dist.apache.org/repos/dist/release/cloudstack/releases/
```

Wait 24 hours for the mirrors to catch up.

Update http://cloudstack.apache.org/downloads.html to point to the new release.

Remove the prior release from the release dist area (it's still present).

```
$ oldrelease=X.Y.Z.O # If new release is i.e. 4.13.1.0, then old one would be 4.13.0.0. Be carefull with
multiple/supported LTS releases
$ svn delete -m "Removing old $oldrelease release" https://dist.apache.org/repos/dist/release/cloudstack
/releases/$oldrelease
```

Contact Security Team

For some releases, the CloudStack security team may be waiting for release announcement so they can simultaneously announce any security advisories related to the release.

Version references documentation

New CloudStack versions should be properly referenced in many places in the documentation. This is done by editing the variables in 'source/conf.py' and 'source/_global.rst'. Most documentation content should have already been transformed to using version variables instead of the hardcoded versions.

Release Notes / Upgrade / Install procedures update

Version references documentation

New CloudStack versions should be properly referenced in many places in the documentation. This is done by editing the variables in 'source/conf.py' and 'source/_global.rst'. Most documentation content should have already been transformed to using version variables instead of the hardcoded versions.

Documentation on the Read The Docs must be updated to include the changes in the release and in the upgrade and install procedures for the new release

Release notes sources to be updated:

- https://github.com/apache/cloudstack-documentation/blob/main/source/releasenotes/about.rst
- https://github.com/apache/cloudstack-documentation/blob/main/source/releasenotes/api-changes.rst (generated using scripts from here)
- https://github.com/apache/cloudstack-documentation/blob/main/source/releasenotes/changes.rst (formerly known as known_issues.rst, generated using scripts from here)
- https://github.com/apache/cloudstack-documentation/blob/main/source/releasenotes/compat.rst

Upgrade and install documentation sources (i.e. for the new 4.14.0.0 release) to be updated:

- https://github.com/apache/cloudstack-documentation/blob/main/source/upgrading/index.rst (i.e. to include the new upgrade-4.13.rst file)
- https://github.com/apache/cloudstack-documentation/blob/main/source/upgrading/upgrade/upgrade-4.XX.rst (from upgrade-4.2.rst to i.e. upgrade-4.13.rst)
- https://github.com/apache/cloudstack-documentation/blob/main/source/quickinstallationguide/qig.rst (Quick installation guide might need updates)
- https://github.com/apache/cloudstack-documentation/tree/main/source/installguide (Full install guide might need updates)
- Specific files, such as 'source/conf.py' and 'source/_global.rst' needs to be properly updated (version variables used for new release / branch / systemVM templates, etc.) to reflect the new release

After the documentation has been updated on GitHub, the following needs to be done to build the documentation on the Read The Docs and make it visible publicly:

- for a **new release** (i.e. new one being 4.14.0.0, while the current is 4.13.x), make sure to create a new branch (4.14 in this example) out of the main branch (where all the documentation updates are previously done)
  - Set the new git tag (i.e. "4.14.0.0") to point to the last commit in the **new** branch (4.14 in this example) - tags are used by Read The Docs to display different documentation versions
  - From the Read The Docs admin section, configure the "Latest" version to point to the new tag ("4.14.0.0") and also add/activate/build the new version explicitly (from tag "4.14.0.0")
  - Additionally, merge back the previously created branch (4.14) to the main branch and consider editing the main variable files ('source/conf.py' and 'source/_global.rst) in the main branch - this is optional and will have to be done for the future release at some point
- for the **maintenance release** (i.e. new one being 4.13.1.0, while the current one is 4.13.0.0), set the new git tag ("4.13.1.0") to the last commit id in the **current** (4.13) documentation branch (where all the documentation updates are previously done)
  - From the Read The Docs admin section, configure the "Latest" version to point to the new git tag (4.13.1.0) and also add/activate/build the new version explicitly (from tag "4.13.1.0")
- Verify the new documentation is visible (i.e.https://docs.cloudstack.apache.org/en/latest, https://docs.cloudstack.apache.org/en/4.13.1.0/, https://docs.cloudstack.apache.org/en/4.14.0.0/)

Community Binaries

The community 'hosters' at download.cloudstack.org and packages.shapeblue.com should be notified and requested to create and upload the convenience binaries to their repositories.

Announcing the Release

After waiting 24 hours for the ASF mirrors to catch up, the release is ready to be announced. Send separate announcement emails to announce@apache.org, announce@cloudstack.apache.org, dev@cloudstack.apache.org, marketing@cloudstack.apache.org and users@cloudstack.apache.org. This is best done using your apache.org address so that the announcement gets moderated through to the lists.

The contents of the message should have been discussed on marketing@cloudstack.apache.org first.

Link to the boilerplate maintenance release announcement

Update http://cloudstack.apache.org/ Pages

The downloads page at http://cloudstack.apache.org/downloads.html must be updated - this involves updating the /data/cloudstack.yaml in http://github.com/apache/cloudstack-www and then building the website (build.sh).
Also, API docs need to be updated if this is a new release (i.e. not maintenance release) - this is done by adding the generated API docs to the https://github.com/apache/cloudstack-www/tree/main/source/api/ folder. (i.e. apidocs are generating as part of "mvn -Pdeveloper -Dnoredist clean install -pl :cloud-apidoc" when running this script - and will be located in the "tools/apidoc/" folder)

Add CloudStack announcement at Apache Blogs Page

Add a post at https://blogs.apache.org/

You need author permissions on Apache Roller, which need to be given by an Admin of the Apache CloudStack project blog.

Removing/Archiving Older Releases

Remove older releases from ASF dist which are mirrored, for example:
svn rm -m "Remove old versions" https://dist.apache.org/repos/dist/release/cloudstack/releases/4.11.3.0
svn rm -m "Remove old versions" https://dist.apache.org/repos/dist/release/cloudstack/releases/cloudmonkey-6.0.0

Appendix A: Adding your GPG key to CloudStack KEYS

Install subversion on your release environment (preferably Linux or OSX based).

```
cd /tmp
svn co https://dist.apache.org/repos/dist/release/cloudstack/ --depth empty
cd cloudstack
svn up KEYS # this will get the latest KEYS file
# append your GPG key to the KEYS file, see the KEYS file for instructions as well.
# gpg --list-sigs <key ID> >> KEYS
# gpg --armor --export<key ID> >> KEYS
svn diff
svn add KEYS
svn commit KEYS -m "KEYS: add key of apache_id FirstName LastName to CloudStack KEYS"
```