To follow this tutorial, you will need the following:

```
64-bit Ubuntu 16.04 with Root Access
```

RECOMMENDED HARDWARE

```
Xeon Based Machine with Minimum 64GB RAM
```

(Recommended Dual Xeon Processor with 128GB or 256GB RAM to make the compile process faster)

INSTALLING BUILD DEPENDENCIES

Update the package database:

```
$  sudo apt-get update
```

Add the GPG key for the official Docker repository to the system

```
$  sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-
   keys 58118E89F3A912897C070ADBF76221572C52609D
```

Install the software-properties-common Package

```
$  sudo apt-get install software-properties-common python-software-
   properties
```

Add the Docker repository to APT sources:

```
$  sudo apt-add-repository 'deb https://apt.dockerproject.org/repo ubuntu-
   xenial main'
```

Update the package database with the Docker packages from the newly added repo:

```
$  sudo apt-get update
```

Make sure you are about to install from the Docker repo instead of the default Ubuntu 16.04 repo:

```
$  apt-cache policy docker-engine
```

You should see output similar to the follow:

Output of apt-cache policy docker-engine

```
docker-engine:
  Installed: (none)
  Candidate: 1.11.1-0~xenial
  Version table:
     1.11.1-0~xenial 500
        500 https://apt.dockerproject.org/repo ubuntu-xenial/main amd64 Packages
     1.11.0-0~xenial 500
        500 https://apt.dockerproject.org/repo ubuntu-xenial/main amd64 Packages
```

Finally, install Docker:

```
$  sudo apt-get install -y docker-engine
```

Add your username to the docker group:

```
$  sudo usermod -aG docker $(whoami)
```

Add a user to the docker group that you're not logged in as,

```
$  sudo usermod -aG docker root
```

You also need a few perl modules installed:
Use the below command to install Perl and other Required Modules

```
$  apt-get install libyaml-libyaml-perl libtemplate-perl libio-handle-util-
   perl libio-all-perl libio-captureoutput-perl libfile-slurp-perl
   libstring-shellquote-perl libsort-versions-perl libdigest-sha-perl
   libdata-uuid-perl libdata-dump-perl git
```

Clone the Repository

```
$  git clone https://git.torproject.org/builders/tor-browser-build.git
$  cd tor-browser-build
```

STARTING A BUILD

To start a build, run one of the following commands, depending on the channel you want to build:

```
$ make release
$ make alpha
$ make nightly
$ make alpha_nightly
```

You can find the build result in the directory release/unsigned/$version or alpha/unsigned/$version for release or alpha builds. The result of nightly or alpha_nightly can be found in the nightly/$date or alpha_nightly/$date directory.

AUTOMATED BUILDS

If the build fails, a shell will automatically open in the build container to help you debug the problem. You probably want to disable this if you want to do automated builds. To disable this, set the RBM_NO_DEBUG environment variable to 1:

```
$ export RBM_NO_DEBUG=1
```

Or set the debug option to 0 in the rbm.local.conf file.

If you want to select the output directory, you can use rbm's --output-dir option. You can look at the Makefile to find the rbm command for what you want to build, and add the --output-dir option. For example if you want to build Tor Browser nightly for linux-x86_64:

```
$ ./rbm/rbm build release --output-dir=/var/builds/nightly/2017-01-23 --
  target nightly --target torbrowser-linux-x86_64
```

The files will be put in the directory selected by --output-dir in a subdirectory named as the version number (or current date for nightly).

To remove this version subdirectory, add the noversiondir target:

```
$ ./rbm/rbm build release --output-dir=/var/builds/nightly/2017-01-23 --
  target nightly --target torbrowser-linux-x86_64 --target noversiondir
```

AUTOMATED BUILDS USING TBB-TESTSUITE

The Tor Browser test suite scripts can also be used to do nightly builds and publish the build logs.

If you want to do that, start by cloning the git repository:

```
$ git clone https://git.torproject.org/boklm/tor-browser-bundle-
  testsuite.git
```

Install some dependencies:

```
$ apt-get install -y libdata-dump-perl libfile-slurp-perl libio-
  captureoutput-perl perlmagick libjson-perl libwww-perl liblwp-protocol-
  https-perl libtemplate-perl libyaml-syck-perl libdatetime-perl libemail-
  sender-perl libemail-simple-perl libfile-type-perl libipc-run-perl
  libxml-libxml-perl
```

Copy the config/tor-browser_build-boklm file and edit it:

```
$  cd tor-browser-bundle-testsuite
$  cp config/tor-browser_build-boklm config/tor-browser_build-$user
$  vim config/tor-browser_build-$user
```

Change the publish_dir and publish_url options. The publish_dir option is the local directory where the builds will be stored. The publish_url option is the public URL where the builds will be available.

Copy the tools/tor-browser-builds-boklm file and edit it to change the --config= option :

```
$  cp tools/tor-browser-builds-boklm tools/tor-browser-builds-$user
$  vim tools/tor-browser-builds-$user
```

You can now run ./tools/tor-browser-builds-$user to start the build, and add it to you crontab.

The html build reports will be available in the reports/ directory, and the build files in the tor-browser-builds/ directory (unless you changed the publish_dir option).

SIGNING BUILDS

If the environment variable RBM_SIGN_BUILD is set to 1, the sha256sums-unsigned-build.txt file will be signed with gpg. You can use the RBM_GPG_OPTS environment variable to add some options to the gpg command used to sign the file. You can also set the var/sign_build and var/sign_build_gpg_opts options in the rbm.local.conf file.

CLEANING OBSOLETE FILES AND CONTAINERS IMAGES

There will be a script to clean old build files and containers that are no longer used, but it has not been added yet.

MULTIPLE BUILD DIRECTORIES ON THE SAME HOST

You can do multiple builds of Tor Browser in different directories on the same host. However the docker images namespace is global, so you may have some conflicts with the same image names used by the different builds. By default, the docker images are prefixed with tor-browser_$USER. You can change this prefix by defining the docker_image_prefix option in rbm.local.conf, using a different prefix for each of your build directories.