**Project ‘JonDoBrowser-Build’**

|  |  |  |
| --- | --- | --- |
| Document Created By | Browser Version | Kai Tian |
| Version 1.0.0 | 7.0a3 | 2017/05/17 |
| Version 1.0.1 | 7.5a1 | 2017/06/28 |
|  |  |  |
|  |  |  |

1. **Project Overview**

This project heavily relies on ‘tor-browser-build’ project from Tor Project and applies several patches to make TorBrowser into JonDoBrowser.

( <https://git.torproject.org/builders/tor-browser-build.git> )

1. **Project ‘tor-browser-build’**

‘tor-browser-build’ project uses RBM (Reproducible Build Manager) to cross-compile (for linux, windows, osx) tor-browser inside respective dockers by automatically downloading the latest source code.

The main project name is contained in ‘rbm.conf’ file, which is ‘tor-browser’.

Build components are contained in ‘projects’ directory.

Keyrings ‘keyring’ directory will be used to validate the identity of remote source code for each project.

1. **‘Rbm.conf’ file**

This file contains which project (including version number) to build in rbm environment.

‘torbrowser\_version’ shows the current version of ‘tor-browser’.

We will be using ‘torbrowser\_version’ when downloading ‘tor-browser’ source code.

1. **‘projects/tor-browser’ directory**

It contains ‘tor-browser’ project and other build components such as ‘torbutton’, ‘tor-launcher’, etc.

The main project is ‘tor-browser’.

‘projects/tor-browser/config’ contains necessary information.

* filename

This indicates which commit of ‘tor-browser’ repository ( <https://git.torproject.org/tor-browser.git> ) will be used to compile tor-browser.

e.g.

'tor-browser-[% c("version") %]-[% c("var/osname") %]-[% c("var/build\_id") %]'

‘tor-browser’ version number is contained in ‘rbm.conf’

* input\_files

This contains all the build components in ‘tor-browser’.

Each ‘project’ in this section has relevant directory in ‘projects’ folder. These projects will be prepared one by one and its output will be copied to the docker image.

Each ‘filename’ in this section indicates that files or directories of the same name in ‘projects/tor-browser’ directory will be copied to the root directory of docker image.

We will remove tor, torbutton, tor-launcher projects and add jondoaddon, jondo-launcher, JonDo directory to input\_files section.

1. **‘projects/firefox’ directory**

Firefox is the main part of tor-browser.

tor-browser uses Firefox ESR source code and added several patches necessary for tor-browser.

‘tor-browser’ repository contains several branches corresponding to each release of Firefox ESR. e.g. tor-browser-52.1.1esr-7.0-1

‘tor-browser’ repository also contains several tags corresponding to each release of tor-browser. e.g. tor-browser-52.1.1esr-7.0-1-build2

1. **‘projects/torbutton’ directory**

torbutton addon is a Firefox extension that modifies Firefox network settings so that it can be used with Tor.

This addon was previously replaced with JonDoFox addon in version 7.0.

This addon is now included in JonDoBrowser to enable tor network in JonDoBrowser as of version 7.5.

JonDoFox addon is derived from torbutton in which proxy settings are added to enable JonDo network.

1. **‘projects/tor-launcher’ directory**

tor-launcher is a Firefox extension that is responsible for directly communicating with Tor. It launches Tor at startup, and blocks Firefox until Tor network is successfully established.

This addon was previously replaced with jondo-launcher addon in version 7.0.

This addon is now included in JonDoBrowser to enable tor network swithching in JonDoBrowser as of version 7.5.

jondo-launcher which will launch JonDo (java local proxy server) at startup.

1. **‘projects/tbb-windows-installer’ directory**

After building ‘tor-browser’ for windows this project is used to create Windows installers. It will download NSIS installer script and use it to create Windows installers.

We will need to modify NSIS installer script to include JRE installer and to replace installer icon.

1. **‘keyring’ directory**

It contains several gpg keyrings.

RBM uses gpg to check if remote source code for each project is modified by any third party by validating gpg fingerprint of git commit.

Because we will need to apply our patches to ‘tor-browser’, ‘jondoaddon’ and ‘tbb-windows-installer’ projects, we have to tell RBM to uses our gpg keyring to validate our patches.

e.g. I used ‘tinkerbel.gpg’ to sign commits, hence I copied ‘tinkerbel.gpg’ to ‘keyring’ directory and modified ‘config’ files in projects/tor-browser, projects/jondoaddon, projects/tbb-windows-installer so that they will use ‘tinkerbel.gpg’ when validating commits.

1. **Patches to ‘tor-browser’**

We will patch ‘tor-browser-build’, ‘firefox’, ‘jondoaddon’, ‘tbb-windows-installer’, ‘torupdates’ and ‘torbutton’.

These patches will be automatically carried out by ‘JonDoBrowser-Build’ projects written in pure Bash script.

1. **JonDoBrowser-Build Preparation Process**
2. **build.sh (main script)**

This is the backbone of ‘JonDoBrowser-Build’ hierarchy.

Its source code and necessary explanation is provided below.

|  |
| --- |
| #!/bin/bash  # ‘tor-browser-build’ is constantly updated and some commits are bound to produce errors.  # If we want a stable build, we will need to pinpoint which commit will be used.  # If no commit hash is provided by user, build.sh will use b9fc5fc4a562ea…  tor\_browser\_build\_commit\_hash=${1:-8d3c245def1e9047715266953fe83df560da0328}  echo "Building with $tor\_browser\_build\_commit\_hash bundle."  #project\_dir is the absolute path the ‘JonDoBrowser-Build’ directory.  project\_dir="$(cd "$(dirname "${BASH\_SOURCE[0]}")" && pwd)"  # Check if keyring exists in OS.  # If keyring exists, we save its path and fingerprint.  # These will be used when signing commits after applying patches.  # If no keyring exists, we abort the build process and manually create keyring.  source "$project\_dir/check-gpg.sh"  # First, we will remove clones that are made in ‘tor-browser-build’  # These are just clones of local repositories, so we can safely remove them.  cd ..  if [ -d "tor-browser-build/git\_clones" ]; then  rm -r tor-browser-build/git\_clones/firefox  rm -r tor-browser-build/git\_clones/jondoaddon  rm -r tor-browser-build/git\_clones/tbb-windows-installer  fi  # Now we clone ‘tor-browser-build’ project and checkout specific commit  # clone tor-browser-build : need to be done before any other cloning  source "$project\_dir/local-clone-separate.sh" tor-browser-build https://git.torproject.org/builders/tor-browser-build.git  cd tor-browser-build  git checkout $tor\_browser\_build\_commit\_hash -f  # Some projects will be cloned in local directories.  # We will apply our patches to these local repositories.  # Later on we will use these local directories as remote origin.  # clone repositories to work with : jondoaddon, firefox, tbb-windows-installer, torupdates  source "$project\_dir/local-clone.sh"  # When we checkout specific commit, git repository will become detached-head state  # so we won’t be able to save our patches  # unless we create a new branch based on this commit.  # We will create a new branch each time when we checkout specific git commit.  tmp\_branch\_name=`date +%Y%m%d%H%M%S`  # Now we apply our patches to local repositories and commit the changes if necessary  source "$project\_dir/modify-jondoaddon.sh"  source "$project\_dir/modify-firefox.sh"  source "$project\_dir/modify-tor-browser-build.sh"  source "$project\_dir/modify-torupdates.sh"  source "$project\_dir/modify-tbb-windows-installer.sh"  source "$project\_dir/modify-torbutton.sh"  # Time to build jondobrowser  cd $project\_dir  cd ../tor-browser-build  # First make a new directory where final builds will be stored  # Or remove it if it already exists  mkdir -p "alpha/unsigned/$torbrowser\_version"  rm -r "alpha/unsigned/$torbrowser\_version"  # Bulid alpha version for all OS  make alpha  # Rename final builds to jondobrowser  cd "alpha/unsigned/$torbrowser\_version"  rename 's/torbrowser/jondobrowser/g' \*  rename 's/TorBrowser/JonDoBrowser/g' \*  rename 's/tor-browser/jondobrowser/g' \*  sed -i -- 's#tor-browser#jondobrowser#g' ./sha256sums-unsigned-build.txt  sed -i -- 's#torbrowser#jondobrowser#g' ./sha256sums-unsigned-build.txt  sed -i -- 's#TorBrowser#JonDoBrowser#g' ./sha256sums-unsigned-build.txt  # Copy final builds to server directory.  cd /var/www/torupdates/htdocs  if ! [ -d jondobrowser ] ; then  mkdir jondobrowser  fi  cd $project\_dir  cd ..  cp -r "tor-browser-build/alpha/unsigned/$torbrowser\_version" "/var/www/torupdates/htdocs/jondobrowser"  cd /var/www/torupdates  # Update server response with new builds  ./update\_responses |

1. **check-gpg.sh**

Check if gpg is installed and keyring exists.

If keyring exists, its absolute path and fingerprint will be saved for later used.

If it does not exist, we will exit build process and let the user manually install gpg and create keyring.

|  |
| --- |
| #!/bin/bash  # check if keyring exists  gpg\_done=0  while read output  do  if [[ $output == "pub"\*"/"\* ]]; then  gpg\_done=1  tmp\_index=`expr index "$output" /`  keyring\_fingerprint=${output:$tmp\_index:8}  fi  if [[ $output == \*"/.gnupg/"\*".gpg" ]]; then  keyring\_path=$output  fi  done <<< "$(gpg --list-keys)"  # if no keyring exist or gpg is not installed, exit build process  if [ $gpg\_done == 1 ]; then  echo "- Using keyring file $keyring\_path and fingerprint $keyring\_fingerprint"  else  echo "gpg not done"  echo "No keyring exist. Please generate a keyring using gpg --gen-key and try again."  exit  fi |

1. **local-clone.sh**

Clone and update ‘tor-browser’, ‘jondoaddon’, ‘tbb-windows-install’ and ‘torupdates’ repositories to ‘local-repo’ directory.

We will call local-clone-separate.sh to clone and update one by one.

|  |
| --- |
| #!/bin/bash  # create ‘local-repo’ directory if it does not exist  cd $project\_dir  cd ..  if ! [ -d 'local-repo' ]; then  echo 'Creating local-repo directory.'  mkdir 'local-repo'  fi  echo '- Using local-repo directory to clone remote repositories'  cd local-repo  git\_dir=`pwd`  # clone or update tor-browser  source "$project\_dir/local-clone-separate.sh" tor-browser https://git.torproject.org/tor-browser.git firefox-local  # clone or update jondo addon  source "$project\_dir/local-clone-separate.sh" jondobrowser https://github.com/jondos/jondobrowser jondoaddon-local  #clone torbutton  source "$project\_dir/local-clone-separate.sh" torbutton https://git.torproject.org/torbutton.git torbutton-local  # clone or update tbb-windows-installer  source "$project\_dir/local-clone-separate.sh" tbb-windows-installer https://github.com/moba/tbb-windows-installer.git tbb-windows-installer-local  # clone or update torupdates  cd /var  if ! [ -d 'www' ]; then  mkdir 'www'  fi  cd www  source "$project\_dir/local-clone-separate.sh" torupdates https://github.com/lancerajee/torupdates |

1. **local-clone-separate.sh**

This script is called with parameters such as repository directory, remote origin url and final directory name.

|  |
| --- |
| #!/bin/bash  #Usage:  #./local-clone [repo\_dir] [git repo url] [final\_dir]  repo\_dir=$1  repo\_url=$2  final\_dir="${3:-$1}"  # If clone directory exist, stash all local changes and update with the remote repository.  if [ -e "$final\_dir/.git" ]; then  cd $final\_dir  echo "Updating with the remote repository $repo\_url"  git fetch --all  git reset --hard FETCH\_HEAD  git clean -df  cd ..  # If it is not cloned yet, clone remote repository and rename the directory to final\_dir  else  git clone $repo\_url  if [ $repo\_dir != $final\_dir ]; then  mv $repo\_dir $final\_dir  fi  fi |

1. **local-commit.sh**

This script is called after patch is applied to local repository.

This script will automatically commit changes to local repository and put a tag to this commit.

And it will sign the tag with gpg fingerprint.

|  |
| --- |
| #!/bin/bash  # tagname will be given as input parameter  tagname=$1  # After patch is applied, commit changes to local repository  git add \*  git commit -m "automatic commit for jondobrowser-local"  # Get the last commit hash  while read output  do  if [[ $output == "commit "\* ]]; then  commit\_hash=${output:7}  fi  done <<< "$(git log -1)"  # Put a tag to this commit  git tag -a $tagname $commit\_hash -m "automatic commit for jondobrowser-local" -f  # Config local repository to use fingerprint  git config --local user.signingkey $keyring\_fingerprint  # Sign the tag  echo $tagname, $keyring\_fingerprint  git tag -s $tagname -m "automatic signing" -f |

1. **makexpi.sh**

This script is from ‘torbutton’ project and will be copied into ‘jondoaddon’ local repository.

It will make .xpi file based on firefox extension source code.

1. **JonDoBrowser-Build Patch Process**
2. **modify-firefox.sh (Pacth to ‘tor-browser’ project)**

We will checkout specific tag into a new branch and modify several files in order to change brand, update url and update behavior.

|  |
| --- |
| #!/bin/bash  #modify firefox local repo  # Get firefox tagname from ‘projects/firefox/config’ to work with  cd $project\_dir  cd ../tor-browser-build  while read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"git\_hash: "\* ]]; then  line=${line##\*]-}  line=${line%%\'}  tmp\_build=${line##\*-}  tmp\_subversion=${line%%-build\*}  fi  if [[ $line == \*"firefox\_version: "\* ]]; then  firefox\_version=${line##\*firefox\_version: }  fi  if [[ $line == \*"torbrowser\_branch: "\* ]]; then  tmp\_branch=${line##\*torbrowser\_branch: }  break  fi  done < "./projects/firefox/config"  torbrowser\_branch="tor-browser-$firefox\_version-$tmp\_branch-$tmp\_subversion"  torbrowser\_tag="$torbrowser\_branch-$tmp\_build"  # make a new temporary branch from the above tag  cd ..  cd local-repo/firefox-local  git checkout -b "jondo$tmp\_branch\_name" $torbrowser\_tag  # modify JonDoBrowser update url (automatic url, manual url)  sed -i -- 's#aus1.torproject.org/torbrowser/update\_3#jondobrowser.jondos.de#g' ./browser/app/profile/firefox.js  sed -i -- 's#www.torproject.org/download/download-easy.html#jondobrowser.jondos.de/jondobrowser/#g' ./browser/branding/official/pref/firefox-branding.js  sed -i -- 's#www.torproject.org/projects/torbrowser.html#jondobrowser.jondos.de/jondobrowser/#g' ./browser/branding/official/pref/firefox-branding.js  # Modify tor-related proxy settings  sed -i -- 's#pref("network.proxy.socks", "127.0.0.1");#pref("network.proxy.socks", "");#g' ./browser/app/profile/000-tor-browser.js  sed -i -- 's#pref("network.proxy.socks\_port", 9150);#pref("network.proxy.socks\_port", 0);#g' ./browser/app/profile/000-tor-browser.js  sed -i -- 's#pref("network.proxy.socks\_remote\_dns", true);#pref("network.proxy.socks\_remote\_dns", false);#g' ./browser/app/profile/000-tor-browser.js  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"app.update.url"\* ]] && [[ $line == \*"torproject.org"\* ]]; then  echo "pref(\"app.update.url\", \"https://jondobrowser.jondos.de/%CHANNEL%/%BUILD\_TARGET%/%VERSION%/%LOCALE%\");"  else  echo "$line"  fi  done < "./browser/app/profile/firefox.js" > "./firefox.js\_tmp"  mv ./firefox.js\_tmp ./browser/app/profile/firefox.js  # When update is available JonDoBrowser will download .mar file  # JonDoBrowser will disable signature check for .mar file  VerifySignatureFound=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"ArchiveReader::VerifySignature()"\* ]]; then  VerifySignatureFound=1  fi  echo "$line"  if [ $VerifySignatureFound == 1 ] && [[ $line == \*"}"\* ]]; then  VerifySignatureFound=0  echo "return OK;"  fi  done < "./toolkit/mozapps/update/updater/archivereader.cpp" > "./archivereader\_tmp.cpp"  mv ./archivereader\_tmp.cpp ./toolkit/mozapps/update/updater/archivereader.cpp  # During update, .mar file contents will be overwritten to existing directories and files.  # We’d better kill all process before update.  # JonDoBrowser will be closed automatically, but JAP.jar will still be running.  # We will kill JAP.jar process right after JonDoBrowser is turned off and right before update is applied.  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"const int callbackIndex ="\* ]]; then  echo "#if defined(XP\_WIN)"  echo " system(\"taskkill /F /T /IM JonDo.exe\");"  echo " system(\"wmic process where \\\"name like \'%java%\'\\\" delete\");"  echo "#elif defined(XP\_MACOSX)"  echo " system(\"pkill -f \'JAP.app\'\")"  echo "#else"  echo " system(\"pkill -f \'java.\*JAP.jar\*\'\")"  echo "#endif"  fi  echo "$line"  done < "./toolkit/mozapps/update/updater/updater.cpp" > "./updater.cpp.tmp"  mv ./updater.cpp.tmp ./toolkit/mozapps/update/updater/updater.cpp  # tor-browser’s own extensions are not signed by Mozilla  # So they already disabled signature check for torbutton, tor-launcher.  # We will add ‘jondofox’, ‘jondo-launcher’ and ‘jondoswitcher’ where necessary.  # That way our extensions will be installed and enabled in JonDoBrowser without signature.  git grep -l 'addon.id == "torbutton@torproject.org" ||' | xargs sed -i 's/addon.id == "torbutton@torproject.org" ||/addon.id== "torbutton@torproject.org" || addon.id == "info@jondos.de" || addon.id == "jondo-launcher@jondos.de" || addon.id == "jondoswitcher@jondos.de" ||/g'  git grep -l 'addon.id != "torbutton@torproject.org" &&' | xargs sed -i 's/addon.id != "torbutton@torproject.org" &&/addon.id!= "torbutton@torproject.org" \&\& addon.id != "info@jondos.de" \&\& addon.id != "jondo-launcher@jondos.de" \&\& addon.id != "jondoswitcher@jondos.de" \&\&/g'  git grep -l 'aAddon.id == "torbutton@torproject.org" ||' | xargs sed -i 's/aAddon.id == "torbutton@torproject.org" ||/aAddon.id== "torbutton@torproject.org" || aAddon.id == "info@jondos.de" || aAddon.id == "jondo-launcher@jondos.de" || aAddon.id == "jondoswitcher@jondos.de" ||/g'  checkTorNetworkFound=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"bool isTor = false;"\* ]]; then  checkTorNetworkFound=1  fi  if [ $checkTorNetworkFound == 1 ]; then  echo "$line"  else  if [[ $line == \*"profile.default/extensions/torbutton@torproject.org.xpi"\* ]]; then  echo "bool isTor = false;"  echo "#ifdef XP\_WIN"  echo " WCHAR envValue[10];"  echo " int envValueLength = GetEnvironmentVariableW(L\"JONDO\_NETWORK\", envValue, 10);"  echo " if (envValueLength > 0 && lstrcmpW(envValue, L\"tor\") == 0) isTor = true;"  echo "#else"  echo " const char \*envValue = PR\_GetEnv(\"JONDO\_NETWORK\");"  echo " if(envValue != NULL && strcmp(envValue, \"tor\") == 0) isTor = true;"  echo "#endif"  echo "if(isTor) {"  echo " uriString.Append(\"profile.default/extensions/torbutton@torproject.org.xpi\");"  echo "} else {"  echo " uriString.Append(\"profile.default/extensions/info@jondos.de.xpi\");"  echo "}"  elif [[ $line == \*"extensions/torbutton@torproject.org.xpi"\* ]]; then  echo "bool isTor = false;"  echo "#ifdef XP\_WIN"  echo " WCHAR envValue[10];"  echo " int envValueLength = GetEnvironmentVariableW(L\"JONDO\_NETWORK\", envValue, 10);"  echo " if (envValueLength > 0 && lstrcmpW(envValue, L\"tor\") == 0) isTor = true;"  echo "#else"  echo " const char \*envValue = PR\_GetEnv(\"JONDO\_NETWORK\");"  echo " if(envValue != NULL && strcmp(envValue, \"tor\") == 0) isTor = true;"  echo "#endif"  echo "if(isTor) {"  echo " uriString.Append(\"extensions/torbutton@torproject.org.xpi\");"  echo "} else {"  echo " uriString.Append(\"extensions/info@jondos.de.xpi\");"  echo "}"  else  echo "$line"  fi  fi  done < "./toolkit/xre/nsAppRunner.cpp" > "./nsAppRunner.cpp.tmp"  mv ./nsAppRunner.cpp.tmp ./toolkit/xre/nsAppRunner.cpp  git grep -l 'torbutton%40torproject.org' | xargs sed -i 's/torbutton%40torproject.org/info%40jondos.de/g'  git grep -l 'tor-launcher%40torproject.org' | xargs sed -i 's/tor-launcher%40torproject.org/jondoswitcher%40jondos.de:0.1.1,jondo-launcher%40jondos.de/g'  # Modify brandname, icon, copyright notice etc.  git grep -l 'TorBrowser' | xargs sed -i 's/TorBrowser/JonDoBrowser/g'  git grep -l 'Tor Browser' | xargs sed -i 's/Tor Browser/JonDoBrowser/g'  git grep -l 'Tor Project' | xargs sed -i 's/Tor Project/JonDos GmbH/g'  git grep -l 'Firefox and the Firefox logos are trademarks of the Mozilla Foundation.' | xargs sed -i 's/Firefox and the Firefox logos are trademarks of the Mozilla Foundation./JonDoBrowser and the JonDoBrowser logos are trademarks of JonDos GmbH, Germany./g'  cp $project\_dir/img/\* ./browser/branding/official  # Patching ‘tor-browser’ is done.  # Commit changes to local repository.  source "$project\_dir/local-commit.sh" $torbrowser\_tag |

1. **modify-jondoaddon.sh (small patch to JonDoFox addon)**

Because ‘torbutton’ extension will be replaced by ‘jondoaddon’, we will patch ‘jondoaddon’ local repository to have similar structure as ‘torbutton’ does.

|  |
| --- |
| #!/bin/bash  #modify jondo addon local repo  cd $project\_dir  cd ../local-repo/jondoaddon-local  # Copy ‘makexpi.sh’ from ‘JonDoBrowser-Build’ to ‘jondoaddon’ directory.  cp $project\_dir/makexpi.sh ./  # Get version of jondoaddon  while read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"<em:version>"\* ]]; then  line=${line##\*<em:version>}  jondoaddon\_version=${line%%</em:version>}  break  fi  done < "src/install.rdf"  # Rename every tor-related string to JonDo-related string  git grep -l 'About Torbutton' | xargs sed -i 's/About Torbutton/About JonDo addon/g'  git grep -l 'About Tor Browser' | xargs sed -i 's/About Tor Browser/About JonDoBrowser/g'  git grep -l 'About Tor' | xargs sed -i 's/About Tor/About JonDoBrowser/g'  git grep -l 'TorBrowser' | xargs sed -i 's/TorBrowser/JonDoBrowser/g'  git grep -l 'Tor Browser' | xargs sed -i 's/Tor Browser/JonDoBrowser/g'  git grep -l 'Tor-Browser' | xargs sed -i 's/Tor-Browser/JonDoBrowser/g'  git grep -l 'Tor Project' | xargs sed -i 's/Tor Project/JonDos GmbH/g'  git grep -l 'Firefox and the Firefox logos are trademarks of the Mozilla Foundation.' | xargs sed -i 's/Firefox and the Firefox logos are trademarks of the Mozilla Foundation./JonDoBrowser and the JonDoBrowser logos are trademarks of JonDos GmbH, Germany./g'  #modify german version  #tor specific string change  git grep -l 'Tor-Projekt' | xargs sed -i 's/Tor-Projekt/JonDos GmbH/g'  # Commit changes to local git repo  source "$project\_dir/local-commit.sh" $jondoaddon\_version |

1. **modify-tbb-windows-installer.sh (patch to windows installer script)**

We will include JAP.jar and JRE online installer to Windows installer script.

|  |
| --- |
| #!/bin/bash  # modify tbb-windows-installer addon local repo  cd $project\_dir  cd ../local-repo/tbb-windows-installer-local  # Checkout current version from repository  git checkout -b "jondo$tmp\_branch\_name" "v$tbb\_windows\_installer\_version"  # Replace tor-related strings with jondo-related strings  git grep -l 'torbrowser.ico' | xargs sed -i 's/torbrowser.ico/jondobrowser.ico/g'  git grep -l 'TorBrowser' | xargs sed -i 's/TorBrowser/JonDoBrowser/g'  git grep -l 'Tor Browser' | xargs sed -i 's/Tor Browser/JonDoBrowser/g'  rm torbrowser.ico  # Replace installer icon  cp "$project\_dir/img/firefox.ico" ./jondobrowser.ico  # Modify NSIS script to include JRE installer  # First, make installer require Administrator permission because of JRE install  # Add a section to installer JRE automatically during install  jre\_section\_found=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"RequestExecutionLevel"\* ]]; then  echo " RequestExecutionLevel admin"  elif [[ $line == \*"Install Java SE Runtime Environment"\* ]]; then  jre\_section\_found=1  echo "$line"  elif [[ $line == \*"Function CreateShortcuts"\* ]] && [ $jre\_section\_found == 0 ]; then  echo "Section \"Install Java SE Runtime Environment\""  echo " MessageBox MB\_OK \"Install Java SE Runtime Environment?\""  echo " File \"\${TBBSOURCE}\\Browser\\JonDo\\jre.exe\""  echo " ExecWait \"\$INSTDIR\\Browser\\JonDo\\jre.exe\""  echo "SectionEnd"  echo ""  echo "$line"  else  echo "$line"  fi  done < "torbrowser.nsi" > "torbrowser.nsi.tmp"  rm torbrowser.nsi  mv torbrowser.nsi.tmp torbrowser.nsi  # Patching Done  # Commit to local git repo  source "$project\_dir/local-commit.sh" "v$tbb\_windows\_installer\_version" |

1. **modify-tor-browser-build.sh**

This script is the main patching part.

We will patch ‘rbm.conf’, ‘keyring’, ‘projects/firefox’, ‘projects/jondoaddon’, ‘projects/tor-browser’ and ‘projects/tbb-windows-installer’.

|  |
| --- |
| #!/bin/bash  #modify tor-browser-build local repository  cd $project\_dir  cd ../tor-browser-build  # Copy .gpg to ‘keyring’ directory  cp $keyring\_path ./keyring/tinkerbel.gpg  # Rename TorBrowser/Tor Browser to JonDoBrowser inside Bundle-Data directory  git grep -l 'Tor Browser' | xargs sed -i 's/Tor Browser/JonDoBrowser/g'  git grep -l 'TorBrowser' | xargs sed -i 's/TorBrowser/JonDoBrowser/g'  mv projects/tor-browser/Bundle-Data/mac/TorBrowser projects/tor-browser/Bundle-Data/mac/JonDoBrowser  mv projects/tor-browser/Bundle-Data/PTConfigs/mac/TorBrowser.app.meek-http-helper projects/tor-browser/Bundle-Data/PTConfigs/mac/JonDoBrowser.app.meek-http-helper  # Modify ‘projects/firefox/config’  # Change git\_url and gpg\_keyring to local repository path and our own keyring respectively.  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"git\_url: "\* ]]; then  echo "git\_url: $git\_dir/firefox-local/.git"  elif [[ $line == \*"gpg\_keyring: "\* ]]; then  echo "gpg\_keyring: tinkerbel.gpg"  else  echo "$line"  fi  done < "./projects/firefox/config" > "./config.tmp"  mv ./config.tmp ./projects/firefox/config  # Modify ‘projects/tor-browser/build’  # We will add ‘jondoaddon’, ‘jondo-launcher’, ‘jondoswitcher’.  # Add a section to copy JAP.jar and its launcher.  # For Windows, copy JAP.jar, JonDo.exe, dll’s.  # For Linux, copy JAP.jar, JonDoLauncher.  # For OSX, copy JAP.app.  JonDoToTorFound=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"mv"\*"input\_files\_by\_name/tor-launcher"\* ]]; then  echo "$line"  echo "mv \$rootdir/JonDo/jondo-launcher@jondos.de.xpi \$TBDIR/\$EXTSPATH/jondo-launcher@jondos.de.xpi"  echo "mv \$rootdir/JonDo/jondoswitcher@jondos.de.xpi \$TBDIR/\$EXTSPATH/jondoswitcher@jondos.de.xpi"  elif [[ $line == \*"Extract the MAR tools"\* ]]; then  #copy JonDo for windows  echo "[% IF c(\"var/windows\") %]"  echo " mkdir -p \$TBDIR/JonDo"  echo " mv \$rootdir/JonDo/JonDo\_Windows/\* \$TBDIR/JonDo/"  echo "[% END %]"  echo ""  echo "[% IF c(\"var/linux\") %]"  echo " mkdir -p \$TBDIR/JonDo"  echo " mv \$rootdir/JonDo/JonDo\_Linux/\* \$TBDIR/JonDo/"  echo "[% END %]"  echo ""  echo "[% IF c(\"var/osx\") %]"  echo " mkdir -p \$TBDIR/Contents/MacOS/JonDo"  echo " mv \$rootdir/JonDo/JonDo\_OSX/JonDoLauncher \$TBDIR/Contents/MacOS/JonDo/"  echo " tar -xvf \$rootdir/JonDo/JonDo\_OSX/JAP.app.tar.gz -C \$TBDIR/"  echo "[% END %]"  echo ""  echo "$line"  elif [[ $line == \*"mv"\*"input\_files\_by\_name/torbutton"\* ]]; then  echo "$line"  echo "mv [% c('input\_files\_by\_name/jondoaddon') %] "'$TBDIR/$EXTSPATH/info@jondos.de.xpi'  elif [ $JonDoToTorFound == 0 ] && [[ $line == \*"zip -Xm omni.ja update.locale"\* ]]; then  JonDoToTorFound=1  echo "$line"  elif [[ $line == \*"MYDIR1"\* ]]; then  JonDoToTorFound=2  echo "$line"  elif [ $JonDoToTorFound == 1 ] && [[ $line == \*"var/windows"\* ]]; then  JonDoToTorFound=3  echo "MYDIR1=\$TBDIR/JonDo"  echo "MYDIR2=\$TBDIR/\$EXTSPATH"  echo "[% IF c(\"var/osx\") %]"  echo " MYDIR1=\$TBDIR/Contents/MacOS/JonDo"  echo " ls \$TBDIR/\$EXTSPATH"  echo " cp -r \$TBDIR/Contents/Resources/JonDoBrowser \$TBDIR/Contents/Resources/TorBrowser"  echo "[% ELSE %]"  echo " cp -r \$TBDIR/JonDoBrowser \$TBDIR/TorBrowser"  echo "[% END %]"  echo "cp \$MYDIR2/jondo-launcher@jondos.de.xpi \$MYDIR1/jondo-launcher@jondos.de.xpi"  echo "cp \$MYDIR2/info@jondos.de.xpi \$MYDIR1/info@jondos.de.xpi"  echo "mv \$MYDIR2/torbutton@torproject.org.xpi \$MYDIR1/torbutton@torproject.org.xpi"  echo "mv \$MYDIR2/tor-launcher@torproject.org.xpi \$MYDIR1/tor-launcher@torproject.org.xpi"  echo "$line"  else  echo "$line"  fi  done < "./projects/tor-browser/build" > "./build.tmp"  mv ./build.tmp ./projects/tor-browser/build  sed -i -- 's#tor-browser\_#jondobrowser\_#g' ./projects/tor-browser/build  sed -i -- 's#PKG\_DIR="tor-browser"#PKG\_DIR="jondobrowser"#g' ./projects/tor-browser/build  sed -i -- 's#OUTDIR/tor-browser#OUTDIR/jondobrowser#g' ./projects/tor-browser/build  sed -i -- 's#OUTDIR/torbrowser-install#OUTDIR/jondobrowser-install#g' ./projects/tor-browser/build  sed -i -- 's#MAR\_FILE=tor-browser#MAR\_FILE=jondobrowser#g' ./projects/tor-browser/build  # Modify ‘projects/tor-browser/config’  # Leave ‘tor’, ‘torbutton’, ‘tor-launcher’.  # Add ‘jondoaddon’, ‘jondo-launcher’, ‘jondo-switcher’.  # Add ‘JonDo’ directory which contains JAP.jar and JonDoLauncher for all OS’s.  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"- project: tor-launcher"\* ]] || [[ $line == \*"name: tor-launcher"\* ]] ;  then  #nothing=""  echo "$line"  elif [[ $line == \*"- project: torbutton"\* ]]; then  echo " - project: jondoaddon"  echo " name: jondoaddon"  echo "$line"  elif [[ $line == \*"- project: tor" ]] || [[ $line == \*"name: tor" ]] ; then  #nothing=""  echo "$line"  elif [[ $line == \*"filename: Bundle-Data"\* ]]; then  echo "$line"  echo " - filename: JonDo"  else  echo "$line"  fi  done < "./projects/tor-browser/config" > "./config.tmp"  mv ./config.tmp ./projects/tor-browser/config  # JAP.app cannot be directly copied.  # First package it with tar and extract it when finalizing build.  cp -r "$project\_dir/JonDo" ./projects/tor-browser/JonDo  cd projects/tor-browser/JonDo/JonDo\_OSX  if [ -d JAP.app ]; then  tar -cvzf JAP.app.tar.gz ./JAP.app  rm -r JAP.app  fi  cd ../../../../  # Copy ‘torbutton’ directory to ‘jondoaddon’ and modify config.  # Change git\_url and gpg\_keyring to local repository path and our own keyring.  # Replace project version with jondoaddon’s version number.  cp -r ./projects/torbutton ./projects/jondoaddon  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"git\_url: "\* ]]; then  echo "git\_url: $git\_dir/jondoaddon-local/.git"  elif [[ $line == \*"gpg\_keyring: "\* ]]; then  echo "gpg\_keyring: tinkerbel.gpg"  elif [[ $line == "version: "\* ]]; then  echo "version: $jondoaddon\_version"  else  echo "$line"  fi  done < "./projects/jondoaddon/config" > "./config.tmp"  mv ./config.tmp ./projects/jondoaddon/config  #modify torbutton git url and gpg keyring in order to change jondo-related strings  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"git\_url: "\* ]]; then  echo "git\_url: $git\_dir/torbutton-local/.git"  elif [[ $line == \*"gpg\_keyring: "\* ]]; then  echo "gpg\_keyring: tinkerbel.gpg"  elif [[ $line == "version: "\* ]]; then  torbutton\_version=${line##\*version: }  echo "$line"  else  echo "$line"  fi  done < "./projects/torbutton/config" > "./config.tmp"  mv ./config.tmp ./projects/torbutton/config  # Get browser main version number  while read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"torbrowser\_version: "\* ]]; then  line=${line##\*: \'}  torbrowser\_version=${line%%\'}  break  fi  done < "rbm.conf"  # Remove other locales except for German and English  # Also comment out ‘mesg…’ line in docker image so that no error occurs when building with ‘runc’.  RootProfileEdited=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"- ar" ]] || [[ $line == \*"- es-ES" ]] || [[ $line == \*"- fa" ]] || [[ $line == \*"- fr" ]] || [[ $line == \*"- it" ]] || [[ $line == \*"- ko" ]] || [[ $line == \*"- nl" ]] || [[ $line == \*"- pl" ]] || [[ $line == \*"- pt-BR" ]] || [[ $line == \*"- ru" ]] || [[ $line == \*"- tr" ]] || [[ $line == \*"- vi" ]] || [[ $line == \*"- zh-CN" ]] || [[ $line == \*"var/locale\_ja"\* ]]; then  nothing = ""  elif [ $RootProfileEdited == 0 ] && [[ $line == \*"sudo sed -i -re"\*"/rootfs/root/.profile"\* ]]; then  RootProfileEdited=1  echo "$line"  elif [ $RootProfileEdited == 0 ] && [[ $line == \*"sudo runc start -b"\* ]]; then  echo " sudo sed -i -re 's/^(mesg n)(.\*)$/#\\1\\2/g' '[% c(\"var/container/dir\") %]'/rootfs/root/.profile"  RootProfileEdited=1  echo "$line"  else  echo "$line"  fi  done < "rbm.conf" > "rbm.conf.tmp"  mv rbm.conf.tmp rbm.conf  sed -i -- 's#sudo runc start -b#sudo runc run -b#g' rbm.conf  #fix selfrando path bug  MoveSelfrandoEdited=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [ $MoveSelfrandoEdited == 0 ] && [[ $line == "mv Tools/TorBrowser Tools/JonDoBrowser" ]]; then  MoveSelfrandoEdited=1  echo "$line"  elif [ $MoveSelfrandoEdited == 0 ] && [[ $line == \*"sed -i"\*"/tc-wrapper/ld"\* ]]; then  echo "mv Tools/TorBrowser Tools/JonDoBrowser"  MoveSelfrandoEdited=1  echo "$line"  else  echo "$line"  fi  done < "projects/selfrando/build" > "build.tmp"  mv build.tmp projects/selfrando/build |

1. **modify-torupdates.sh**

We will modify version number and necessary url’s.

|  |
| --- |
| #!/bin/bash  #modify torupdates local repository  cd /var/www/torupdates  git pull  # Modify ‘config.yml’  # Change url’s and keyring  # Change version number  VersionLineFound=0  while IFS='' read -r line || [[ -n "$line" ]]; do  if [[ $line == \*"archive\_url: "\* ]]; then  echo " archive\_url: https://jondobrowser.jondos.de/jondobrowser/"  elif [[ $line == \*"gpg\_keyring: "\* ]]; then  echo " gpg\_keyring: $project\_dir/../tor-browser-build/keyring/tinkerbel.gpg"  elif [[ $line == \*"bundles\_url: "\* ]]; then  echo " bundles\_url: https://jondobrowser.jondos.de/jondobrowser/"  elif [[ $line == \*"mars\_url: "\* ]]; then  echo " mars\_url: https://jondobrowser.jondos.de/jondobrowser/"  elif [[ $line == \*"alpha: "\* ]]; then  echo " alpha: $torbrowser\_version"  elif [[ $line == \*"release: "\* ]]; then  echo " release: $torbrowser\_version"  elif [[ $line == \*"nightly: "\* ]]; then  echo " #nightly: $torbrowser\_version"  elif [ $VersionLineFound == 0 ] && [[ $line == "versions:" ]]; then  VersionLineFound=1  echo "$line"  elif [ $VersionLineFound == 1 ]; then  VersionLineFound=2  echo " $torbrowser\_version:"  elif [[ $line == \*"platformVersion: "\* ]]; then  echo " platformVersion: $firefox\_version"  else  echo "$line"  fi  done < "config.yml" > "config.yml.tmp"  mv config.yml.tmp config.yml  # Change tor-related strings with jondo-related strings.  git grep -l 'TorBrowser' | xargs sed -i 's/TorBrowser/JonDoBrowser/g'  git grep -l 'torbrowser' | xargs sed -i 's/torbrowser/jondobrowser/g'  sed -i -- 's#tor-browser#jondobrowser#g' ./update\_responses |

1. **modify-torbutton.sh**

This will change tor-related strings to jondo-related strings.

|  |
| --- |
| #!/bin/bash  #modify torbutton addon local repo  cd $project\_dir  cd ../local-repo/torbutton-local  #make a new temporary branch from the tag  git checkout -b "jondo$tmp\_branch\_name" $torbutton\_version  git grep -l 'About Torbutton' | xargs sed -i 's/About Torbutton/About JonDo addon/g'  git grep -l 'About Tor Browser' | xargs sed -i 's/About Tor Browser/About JonDoBrowser/g'  git grep -l 'About Tor' | xargs sed -i 's/About Tor/About JonDoBrowser/g'  git grep -l 'TorBrowser' | xargs sed -i 's/TorBrowser/JonDoBrowser/g'  git grep -l 'Tor Browser' | xargs sed -i 's/Tor Browser/JonDoBrowser/g'  git grep -l 'Tor-Browser' | xargs sed -i 's/Tor-Browser/JonDoBrowser/g'  git grep -l 'Tor Project' | xargs sed -i 's/Tor Project/JonDos GmbH/g'  git grep -l 'Firefox and the Firefox logos are trademarks of the Mozilla Foundation.' | xargs sed -i 's/Firefox and the Firefox logos are trademarks of the Mozilla Foundation./JonDoBrowser and the JonDoBrowser logos are trademarks of JonDos GmbH, Germany./g'  #modify german version  git grep -l 'Tor-Projekt' | xargs sed -i 's/Tor-Projekt/JonDos GmbH/g'  #copy jondo-strings from jondofox addon  cp ../jondoaddon-local/src/chrome/locale/en/about\* ./src/chrome/locale/en/  cp ../jondoaddon-local/src/chrome/locale/de/about\* ./src/chrome/locale/de/  cp ../jondoaddon-local/src/chrome/locale/en/brand\* ./src/chrome/locale/en/  cp ../jondoaddon-local/src/chrome/locale/de/brand\* ./src/chrome/locale/de/  cp ../jondoaddon-local/src/chrome/skin/\* ./src/chrome/skin/  #add necessary strings to aboutTor.dtd  echo "<!ENTITY aboutTor.torbrowser\_user\_manual.accesskey \"M\">" >> ./src/chrome/locale/en/aboutTor.dtd  echo "<!ENTITY aboutTor.torbrowser\_user\_manual.label \"JonDoBrowser User Manual\">" >> ./src/chrome/locale/en/aboutTor.dtd  echo "<!ENTITY aboutTor.helpInfo3.label \"Run a Tor Relay Node »\">" >> ./src/chrome/locale/en/aboutTor.dtd  echo "<!ENTITY aboutTor.helpInfo3.link \"https://www.torproject.org/docs/tor-doc-relay.html.en\">" >> ./src/chrome/locale/en/aboutTor.dtd  #german version  echo "<!ENTITY aboutTor.torbrowser\_user\_manual.accesskey \"M\">" >> ./src/chrome/locale/de/aboutTor.dtd  echo "<!ENTITY aboutTor.torbrowser\_user\_manual.label \"JonDoBrowser-Benutzerhandbuch\">" >> ./src/chrome/locale/de/aboutTor.dtd  echo "<!ENTITY aboutTor.helpInfo3.label \"Einen Tor-Relaisknoten betreiben »\">" >> ./src/chrome/locale/de/aboutTor.dtd  echo "<!ENTITY aboutTor.helpInfo3.link \"https://www.torproject.org/docs/tor-doc-relay.html.en\">" >> ./src/chrome/locale/de/aboutTor.dtd  #commit to local git repo  source "$project\_dir/local-commit.sh" $torbutton\_version |

1. **Project Summary**
2. **Update process**

**How should we do a new stable release of JonDoBrowser with the JonDoFox xpi?**

When JonDoFox needs to be updated, first commit the change to JonDoFox git repository. (JonDoBrowser-Build will use the last commit to make .xpi and copy it to JonDoBrowser automatically, so there is nothing to worry about. You may change the version number in install.rdf and JonDoBrowser-Build will use this version number automatically.)

Next, change version number of JonDoBrowser in ‘tor-browser-build/rbm.conf’.

Even if you change it manually, JonDoBrowser-Build will update it with git repository everytime you run ./build.sh, so you will need to tell JonDoBrowser-Build to explicitly change the version number. (This can be done in modify-tor-browser-build.sh)

**What should we do when a new version of Tor browser or Torbutton is released? Do you have suggestions for the process?**

tor-browser-build contains which version of tor-browser and torbutton to use.

Current JonDoBrowser-Build takes an optional parameter, which indicates which commit of tor-browser-build to use to build JonDoBrowser. If no parameter is given, b9fc5fc4a562ea8c5d7fdb797fd3c1ea6f6b413e will be used, which uses tor-browser-7.0-1-build 2.

All you have to do to make a new release of JonDoBrowser is to provide a commit hash to build.sh.

JonDoBrowser doesn’t directly rely on torbutton, what you will need to do is to update JonDoFox and follow 1).

1. **Patches**

For patching process, please refer to Section 3.

1. **Patching Update URL**

To exchange the update url, you need to modify the appropriate lines in ‘modify-firefox.sh’ and ‘modify-torupdates.sh’.

1. **Notes for faster build**

in build.sh  
you will see  
make alpha  
instead of alpha, just type  
make alpha-win-x86\_64  
make alpha-linux-i686  
etc  
and for languages  
check that in rbm.conf  
there are whole bunch of languages  
remove all except the one you want to build  
  
currently, autoscript removes all but english and german