**Index**

[Introduction. 2](#_Toc456291114)

[1. What is changed by me in custom FW vs official FW. 3](#_Toc456291115)

[2. Flashing modified firmware. 6](#_Toc456291116)

[3. Setup SSH access to router. 6](#_Toc456291117)

[4. Setup of Entware-ng. 7](#_Toc456291118)

[5. Open your own firewall ports. 8](#_Toc456291119)

[6. Enable dnscrypt-proxy. 9](#_Toc456291120)

[7. Using your own CA/CERT/KEY/DH files in OpenVPN server(s). 9](#_Toc456291121)

[8. Transmission. 10](#_Toc456291122)

[9. Debian (for advanced users). 11](#_Toc456291123)

Introduction.

This custom firmwares is based on official stock firmware versions for NETGEAR Nighthawk X4 R7500 v1 router and Nighthawk X4S R7800 router. The goal of modification is to extend the functionality of these routers and to use full power of CPU and FPU of IPQ806x processor, limited in official firmware.

**Warning:**

I am not responsible for any damage of your router if you decide to try this custom firmware. You should do all under your own risk and responsibility. Your router is your router and you should understand the risk to brick it.

**What improvements you can get with use of this firmware plus Entware-ng:**

* Improvements of OpenVPN (speed).
* Improvement of SAMBA server (speed of file transfer).
* Improvements of FTP server speed.
* Possibility to setup your own web server (Entware-ng).
* Possibility to setup your own anonymizer proxy with TOR and Privoxy (Entware-ng).
* Possibility to exclude the leaks of your DNS requests by DNSCRYPT (your privacy).
* Etc. etc. etc.

**Note**: Entware-ng installation archives is prepared by me in two variants:

1. For use of official Entware-ng repository maintained by Entware-NG team from their site. It is compiled by them for generic ARMV7-A CPU with soft float point operations. Advantage of this variant is frequent renew of Entware-ng packages versions.
2. Variant compiled by me especially for R7500/R7800, optimized for use with Cortex A-15 CPU (IPQ806x is Krait and thus supports all Cortex A-15 extensions of CPU instructions) and Neon VFPV4 FPU i.e. hardware float. It is significantly faster then soft float version on some tests where float point calculations are needed. Disadvantage of this version is that I do not intend to renew this version very often and you need either to setup your own webserver to have possibility of packages installation, or to install manually necessary packages directly from IPK files from your disk storage. Also, OpenSSL in this version of Entware-ng is optimized with assembler acceleration.

Comparison of speed my version vs official version (cpubench test from Entware, R7500 router):  
**My version of Entware-ng (hard float + optimization):**

*This is CPU and memory benchmark for OpenWRT v0.6. This will then take some time... (typically 30-60 seconds on a 200MHz computer)*

*Overhead for getting time: 0us*

*Time to run memory bench: 0.67[secs]*

*Time to run computation of pi (2400 digits, 10 times): 1.73[secs]*

*Time to run computation of e (9009 digits): 1.80[secs]*

*Time to run float bench: 0.01[secs]*

*Total time:* ***4.2s***

**Official version of Entware-ng (soft float):**

*This is CPU and memory benchmark for OpenWRT v0.6. This will then take some time... (typically 30-60 seconds on a 200MHz computer)*

*Overhead for getting time: 0us*

*Time to run memory bench: 0.82[secs]*

*Time to run computation of pi (2400 digits, 10 times): 3.50[secs]*

*Time to run computation of e (9009 digits): 2.85[secs]*

*Time to run float bench: 0.03[secs]*

*Total time: 7.2s*

So decide yourself what is better for you.

1. What is changed by me in custom FW vs official FW.
2. Most important for use with Entware-ng is that now native Linux filesystems (ext2/3/4) could be used and no “777” mask is applied to files and directories. In official FW when you mount external USB/ESATA disk with native Linux filesystem, you had 777 permissions for all files and directories (read/write/execute access for all, no any permissions restrictions). Use of filesystem without restrictions is nonsense under Linux. No any security, spoiled functionality, not workable daemons. NETGEAR staff modified original codes of Linux kernel (?!) to make this “777”, I returned original kernel code back.
3. Added dropbear SSH server. Started automatically after power on. No “telnetenable” is needed to access router console.
4. I used fresh version of toolchain for firmware compilation (compiler 2015 vs 2012 in stock FW). So more stable and fast codes (common general optimization).
5. “-O2” compilation flag and especial optimization for Cortex A-15 is used for firmware compilation, “-O3” for some key packages (performance).
6. Updated a lot of old OpenWRT packages used in FW to more fresh version, e.g.

openssl-0.9.8p-🡪openssl-1.0.2h

lzo 2.06-🡪lzo 2.09

zlib 1.2.7-🡪zlib 1.2.8

openvpn 2.3.2🡪openvpn 2.3.10

etc. etc. etc.

1. OpenSSL is optimized by using assembler acceleration. OpenSSL test w/o assembler optimization (R7500):

The 'numbers' are in 1000s of bytes per second processed.

type 16 bytes 64 bytes 256 bytes 1024 bytes 8192 bytes

md5 6484.94k 22414.88k 62388.40k 112649.90k 147072.43k

sha1 5612.70k 18752.30k 46857.16k 75064.01k 90426.03k

des cbc 21560.90k 22420.63k 22804.21k 22913.63k 22899.24k

des ede3 8423.97k 8586.75k 8673.35k 8615.25k 8665.99k

aes-128 cbc 43707.08k 48836.26k 50496.73k 50789.38k 50872.59k

aes-192 cbc 36966.49k 41938.32k 43277.10k 43413.85k 43606.65k

aes-256 cbc 34206.93k 36612.95k 37643.73k 37784.92k 37839.37k

sha256 6972.91k 17469.50k 32550.96k 41760.71k 45428.61k

sha512 1270.71k 5078.05k 7560.45k 10519.12k 11826.52k

sign verify sign/s verify/s

rsa 2048 bits 0.026332s 0.000678s 38.0 1474.8

sign verify sign/s verify/s

dsa 2048 bits 0.006750s 0.008101s 148.1 123.4

the same test with assembler acceleration (R7500):

The 'numbers' are in 1000s of bytes per second processed.

type 16 bytes 64 bytes 256 bytes 1024 bytes 8192 bytes

md5 8145.73k 31676.98k 78349.40k 124663.61k 150228.68k

sha1 10928.72k 34788.88k 94729.02k 164864.34k **211435.52k**

des cbc 21092.66k 22393.58k 22870.71k 22939.70k 23053.02k

des ede3 8586.80k 8702.14k 8796.61k 8856.37k 8870.44k

aes-128 cbc 63497.29k 70637.13k 73812.25k 73948.47k 74870.34k

aes-192 cbc 51884.68k 57350.49k 59327.96k 60090.21k 58730.55k

aes-256 cbc 44144.35k 47553.35k 49272.43k 49035.67k 48799.72k

sha256 18213.02k 45359.63k 83741.10k 107474.45k **116959.29k**

sha512 6495.08k 25933.38k 33175.52k 53079.87k 61937.07k

sign verify sign/s verify/s

rsa 2048 bits 0.008295s 0.000178s **120.5 5613.6**

sign verify sign/s verify/s

dsa 2048 bits 0.001678s 0.001969s **596.0 507.8**

i.e. at least your OpenVPN should work faster.

1. Changed automatic mount script: now *a)* disks with ext2/3/4 filesystems are mounted w/o “noexecute” option (i.e. you can run program/script from external drive); *b)* if partition has a label then it is mounted to /tmp/mnt/(labelname)/ directory instead of /tmp/mnt/sda1/ or /tmp/mnt/sdb1/ etc.; *c)* if external storage has the script autorun/scripts/post-mount.sh then it is executed automatically after you insert your USB stick/disk to router or after power on of your router with attached external stick/disk.
2. Script to run Entware-ng programs/daemons is added to /etc/init.d (etc/init.d/optware).
3. Symlink /opt is added and linked to /tmp/opt (needed to link Entware-ng).
4. /etc/profile is changed to use Entware-ng programs (from /opt directory) plus some improvements.
5. fsck.hfsplus is added to have possibility to check HFS/HFS+ filesystems (R7800).
6. Default root’s home is changed from /tmp to /root directory (important for SSH access).
7. Added transmission downloader.
8. It is possible to use your own CA/CRT/KEY/DH files for OpenVPN servers.
9. dnscrypt-proxy is included into firmware (privacy).
10. Some other changes/improvements/bug corrections.
11. Flashing modified firmware.

Nothing special. Just recommendation to restore factory settings in router WebGUI, after you flash my modified FW. Then setup your Wi-Fi, WAN LAN etc settings manually from the scratch.

1. Setup SSH access to router.

After flashing and your settings you need to have SSH access to router. SSH daemon dropbear in R7500 uses port 22 and accepts only authorization by SSH key (no password login due to security). So you need to copy your own authorized\_keys file into /root/.ssh directory. This process is automated, so steps to do that:

1. Prepare **authorized\_keys** file with your public key (what you need in /root/.ssh directory)
2. Optionaly: prepare your own server keys:

dropbear\_ecdsa\_host\_key

dropbear\_rsa\_host\_key

ssh\_host\_ecdsa\_key.pub

ssh\_host\_rsa\_key.pub

1. Prepare USB stick with ext2 filesystem and untar **setssh.tar** in the root of stick (keeping +x filemask (!) for autorun/scrips/post-mount.sh script, computer with Linux is recommended).
2. Place your own authorized\_keys file (obligatory) and your own server keys (optionally) above generic files you got after untar in the root of stick.
3. Insert this USB stick to router. Wait 1-2 minute and try to SSH to router with the key corresponding to your authorized\_keys file.

If you cannot get an access, try to reboot router with this stick attached. Check that autorun/scripts/post-mount.sh has has +x attribute (executable). Check that your authorized\_keys file is valid.

It is recommended to replace generic server keys in /etc/dropbear keys by your own keys after you have an access by SSH if you did not do “2)”. Conmmand dropbearkey and dropbearconvert are available from console.

1. Setup of Entware-ng.

To setup Entware-ng (original or compiled for cortex-a15 with hard float):

1. Prepare new USB stick or disk with ext2 or ext3 or ext4 filesystem from console. Label it “optware”. Ext2 is recommended for USB flash stick, ext4 is recommended for USB HDD. Example to create ext2 filesystem with label “optware”:

mkfs.ext2 –L optware /dev/sda1

1. Untar **entware-initial-official.tar** or **entware-initial-cortexa15.tar** at the root of your stick/disk.
2. Reboot the router. Check that “ls –l /opt/\*” shows entware directories (bin, usr, share, var etc.)
3. Create swap file in /opt/:

cd /opt

dd if=/dev/zero of=swap bs=1024 count=524288

*(for R7500)*

dd if=/dev/zero of=swap bs=1024 count= 1048576

*(for R7800)*

mkswap swap

chmod 0600 swap

swapon swap

1. Reboot router again. After this use “opkg update” and “opkg upgrade” for original Entware repository. Install and use necessary for you packages. Or if you use my version of repository (hard float, Cortex-A15 optimization), then download archive with repository, prepared by me, place them to your webserver and correct **/opt/etc/opkg.conf** file pointing your webserver with packages. Or you can install packages just from local files, unpacking archive in your HDD/stick.
2. Open your own firewall ports.

If you need to make several ports accessible from WAN then create the text file /root/netwall-rules with ports you need to open. Example of this file:

ACCEPT net fw tcp 22,8443

ACCEPT net fw udp 1194

(to open TCP ports 22 and 8443 and UDP port 1194).

1. Enable dnscrypt-proxy.

If you want to use dnscrypt-proxy then create the text file /root/dnscrypt-list with your list of DNS Crypto servers you want to use. Current list is available from this link:

<https://github.com/jedisct1/dnscrypt-proxy/blob/master/dnscrypt-resolvers.csv>

example of the /root/dnscrypt-list file:

4armed

dnscrypt.eu-dk

dnscrypt.org-fr

dnscrypt.eu-nl

It is recommended to use 3-4 servers. W/o this file the router will work as before (w/o use of dnscrypt-proxy). You can test that it works:

<https://www.perfect-privacy.com/dns-leaktest/>

1. Using your own CA/CERT/KEY/DH files in OpenVPN server(s).

If you want to use your own CA/CERT/KEY/DH files and push\_routing\_rule script, put them into /root/openvpn directory. Filenames should be with the following mask:

**\*ca.crt** CA file

**\*.crt** CERT file

**\*.key** KEY file

**dh\*.pem** DH file

If they are in the /root/openvpn directory, then OpenVPN will use them.

Example (files in /root/openvpn/):

my-ca.crt

myserver.crt

myserver.key

dh2048.pem

1. Transmission.

Transmission program (torrents) is included into firmware. It could be run from WebGUI of router.

Important for use of transmission:

1. You need external USB drive attached to router.
2. You need to have swap enabled. See above how to create and enable swap file. If swap is in in /opt directory it will be enabled automatically after reboot of your router.
3. Transmission is not enabled in WebGUI of router if your router is in AP/extender mode, but you still can use transmission, use IP:9091 in your browser (e.g. <http://192.168.1.3:9091>).
4. (*R7800 only*) If Netgear Downloader is enabled, transmission will be disabled. And vice versa. You should use either or.
5. (*R7800 only*) Use section [Netgear Downloader] to run transmission and set the place for downloads by [Configure Settings]->Save Path in WebGUI of your router.
6. (*R7500 only*) Default save path for transmission is /mnt/sda1/downloads. If you want to change it (or other settings for transmission), then stop transmission daemon (/etc/init.d/transmission stop), edit its config file (/etc/transmission/settings.json) and start the daemon again (/etc/init.d/transmission start).
7. Debian (for advanced users).

Also, I prepared the version of chroot-ed Debian Jessie for ARMHF (i.e. with hard float, which will use all power of your FPU). It is in archive **debian-jessie-armhf.tar.qz**. Unpack it to /tmp/mnt/optware and use set-debian.sh script to start it manually. Also it is possible to run start of Debian daemons (e.g. nginx, proftp, tor or what-you-need) together with Entware services. See an example of startup script in /opt/etc/init.d in:

<https://www.hqt.ro/how-to-install-debian-jessie-arm/>

NOTE: I use Debian ARMHF (hard float). It is faster then ARMEL (soft float) in the link above, and incompatible. So use only startup script example from the link above to create your version.

Voxel.