**Index**

[Introduction. 2](#_Toc30496849)

[1. What is changed by me in custom FW vs official FW. 3](#_Toc30496850)

[2. Flashing modified firmware. 5](#_Toc30496851)

[3. Setup SSH access to router. 5](#_Toc30496852)

[4. Setup of Entware. 6](#_Toc30496853)

[5. Open your own firewall ports. 7](#_Toc30496854)

[6. Enable dnscrypt-proxy-2 and stubby. 8](#_Toc30496855)

[7. Using your own CA/CERT/KEY/DH files in OpenVPN server(s). 8](#_Toc30496856)

[8. OpenVPN client (R7800 and R9000 only). 9](#_Toc30496857)

[9. Transmission. 10](#_Toc30496858)

[10. Disable ReadyCLOUD and/or Kwilt (R7800/R9000). 11](#_Toc30496859)

[11. Debian (for advanced users). 12](#_Toc30496860)

[Appendix A. Get SSH access to router (alternative method). 12](#_Toc30496861)

[Appendix B. OpenVPN Client setup example. 14](#_Toc30496862)

Introduction.

This custom firmwares is based on official stock firmware versions for NETGEAR Nighthawk X4 R7500 v1 router, Nighthawk X4S R7800 router and Nighthawk X10 router. The goal of modification is to extend the functionality of these routers and to use full power of CPU and FPU of IPQ806x and AL-514 processors, 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:**

* 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).
* Possibility to setup your own anonymizer proxy with TOR and Privoxy (Entware).
* Possibility to exclude the leaks of your DNS requests by DNSCRYPT (your privacy).
* Etc. etc. etc.

**Note**: Entware installation archive is prepared by me especially for R7500/R7800/R9000, it is 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 than soft float version on some tests where float point calculations are needed.

1. What is changed by me in custom FW vs official FW.
2. Most important for use with Entware 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 2016 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-1.0.2\*

lzo 2.06-🡪lzo 2.10

zlib 1.2.7-🡪zlib 1.2.11

openvpn 2.3.2🡪openvpn 2.4.x

etc. etc. etc.

1. OpenSSL is optimized by using assembler acceleration. OpenSSL test w/o assembler optimization (R9000):

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

type 16 bytes 64 bytes 256 bytes 1024 bytes 8192 bytes

sha1 19729.61k 54213.54k 111554.18k 150575.10k 168700.40k

des cbc 33284.58k 34141.59k 34585.00k 34665.81k 34553.86k

des ede3 12548.81k 12727.87k 12788.65k 12801.71k 12782.25k

aes-128 cbc 57205.07k 60562.69k 62545.32k 63109.12k 63310.51k

aes-192 cbc 50571.55k 52632.14k 53764.35k 54159.02k 54274.73k

aes-256 cbc 44746.83k 45857.66k 47048.96k 47419.08k 47363.41k

sha256 13311.57k 29732.76k 50673.44k 61281.28k 65227.43k

sha512 3768.93k 14927.25k 21400.58k 29089.11k 32216.41k

 sign verify sign/s verify/s

rsa 2048 bits 0.036533s 0.001101s 27.4 908.0

 sign verify sign/s verify/s

 dsa 2048 bits 0.012148s 0.013405s 82.3 74.6

the same test with assembler acceleration (R9000):

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

type 16 bytes 64 bytes 256 bytes 1024 bytes 8192 bytes

sha1 21691.86k 67717.40k 163728.90k 251297.48k 296394.75k

des cbc 33224.61k 34769.92k 35351.13k 35573.21k 35370.33k

des ede3 13231.06k 13375.81k 13498.79k 13595.49k 13485.29k

aes-128 cbc 76702.52k 80093.80k 83207.17k 84156.70k 83875.16k

aes-192 cbc 61568.46k 66469.16k 70230.95k 71435.13k 71363.24k

aes-256 cbc 55345.12k 57141.60k 58567.85k 58935.30k 59026.09k

sha256 24173.65k 56915.65k 102226.09k 128476.16k 139047.56k

sha512 11151.64k 44457.20k 66356.57k 93356.71k 105865.22k

 sign verify sign/s verify/s

rsa 2048 bits 0.008718s 0.000212s 114.7 4709.8

 sign verify sign/s verify/s

dsa 2048 bits 0.002358s 0.002485s 424.1 402.4

 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. /etc/profile is changed to use Entware programs (from /opt directory) plus some improvements.
3. fsck.hfsplus is added to have possibility to check HFS/HFS+ filesystems (R7800).
4. Default root’s home is changed from /tmp to /root directory (important for SSH access).
5. Added transmission downloader.
6. It is possible to use your own CA/CRT/KEY/DH files for OpenVPN servers.
7. dnscrypt-proxy-2 is included into firmware (privacy).
8. Some other changes/improvements/bug corrections.
9. 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 may need to have SSH access to router (e.g. if you wish to use Entware). SSH daemon dropbear in R7500/R7800/R9000 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 are (see Appendix A for step-by-step instruction, alternative method):

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)”. Commands dropbearkey and dropbearconvert are available from console.

1. Setup of Entware.

To setup Entware (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”. Ext4 is highly recommended for USB HDD. Example to create ext4 filesystem with label “optware”:

mkfs.ext4 -L optware -O ^64bit /dev/sda1

(IMPORTANT: additionally to mkfs.ext4 use the command

tune2fs -O ^metadata\_csum /dev/sda1

for routers with kernel < 3.6, such as R7500/R7800)

1. Untar **entware-cortexa15-3x-initial.tar** at the root of your stick/disk.
2. Reboot the router. Check that “ls –l /opt/\*” shows entware directories or symlinks (bin, usr, share, var etc.)
3. Create swap file (optional) in /mnt/sda1 or /mnt/sdb1 or /mnt/sdc1 etc.:

cd /mnt/sda1

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

*(for R7500)*

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

*(for R7800)*

dd if=/dev/zero of=swap bs=1024 count= 2097152

*(for R9000)*

mkswap swap

chmod 0600 swap

swapon swap

1. Reboot router again. After this use “/opt/bin/opkg update” and “/opt/bin/opkg upgrade” Entware repository. Install and use necessary for you packages.
2. Open your own firewall ports.

If you need to make several ports accessible from WAN then create the text file /etc/netwall.conf 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).

NOTE: this file should contain LF symbol at the end of last line (press ENTER key in your text editor).

Additionally you can use your own custom script to add your own iptables rules. This script should be named **firewall-start.sh** and be placed in the **/opt/scripts** directory, i.e. **/opt/scripts/firewall-start.sh**.

1. Enable dnscrypt-proxy-2 and stubby.

To enable DNSCrypt Proxy-2 run from telnet console the commands:

nvram set dnscrypt2=1

nvram commit

reboot

To enable stubby run from telnet console the commands:

nvram set stubby=1

nvram commit

reboot

If both DNSCrypt Proxy-2 and stubby are enabled, only stubby will be used.

To disable DNSCrypt Proxy-2 or/and stubby set them to "0" by nvram.

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 /etc/openvpn/config/ 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 /etc/openvpn/config directory, then OpenVPN will use them.

Example (files in /etc/openvpn/config/):

my-ca.crt

myserver.crt

myserver.key

dh2048.pem

1. OpenVPN client (R7800 and R9000 only).

Important: only TUN clients are supported and it is impossible to use both OpenVPN server and OpenVPN client at the same time. Disable OpenVPN server to use OpenVPN client.

To install OpenVPN client you can use two methods. First, semiautomatic:

1. Create the folder **/openvpn-client** at the root of USB stick (name of folder should be lowercase).
2. Put your \*.ovpn config file into this folder (.ovpn extension of the file must be lowercase).
3. Insert this USB stick into router. OpenVPN client will be started after 30 seconds. And it will be started automatically every time after next reboot already w/o USB stick.

It is suggested to use CA/CERT/KEY of client embedded into you \*.ovpn. But separate CA/CERT/KEY files also could be used. Every file from /openvpn-client folder on the USB stick will be copied to **/etc/openvpn/config/client** directory of your router.

To disable OpenVPN client just create the file “disable” in the folder /openvpn-client (**/openvpn-client/disable**) on your USB stick and insert it into router. Now OpenVPN client will not be used.

Second method of installation is manual: just create /etc/openvpn/config/client directory and put your \*.ovpn file (and CA/CERT/KEY if any) from console using telnet or SSH. Then run:

/etc/init.d/openvpn-client start

Or remove config files manually to disable client and stop client:

/etc/init.d/openvpn-client stop

Log file for OpenVPN client is **/var/log/openvpn-client.log**, check it if you have problems.

NOTE: you can add your own delay for starting OpenVPN client after reboot by the command from telnet:

**nvram set vpn\_client\_delay=120**

*(to set 120 sec. delay)*

See Appendix B for example of custom setup of OpenVPN client.

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 (R7500v1 only). 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/R9000 only*) If Netgear Downloader is enabled, transmission will be disabled. And vice versa. You should use either or.
5. (*R7800/R9000 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. Disable ReadyCLOUD and/or Kwilt (R7800/R9000).

You can disable any future installations of ReadyCLOUD and/or Kwilt for R7800 and R9000 routers. For this you should run the following commands from console

nvram set nocloud=1

nvram set nokwilt=1

nvram commit

then you can manually remove ReadyCLOUD and Kwilt installations (several reboots might be needed). Optionally you can disable Transmission:

 nvram set transmission\_disable=1

 nvram commit

1. 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.

Appendix A. Get SSH access to router (alternative method).

**Generate dropbear key:**

1. Enable telnet login to your router (select corresponding checkbox in <http://routerlogin.net/debug.htm> page).
2. Enter to your router console by telnet.
3. Make /tmp/ssh directory and enter to it:

mkdir /tmp/ssh

cd /tmp/ssh

1. Generate your RSA private key using dropbear:

dropbearkey -t rsa -s 2048 -f id\_dropbear

1. Output your public key:

dropbearkey -y -f id\_dropbear | grep "^ssh-rsa " > id\_rsa.pub

1. Create your /root/.ssh/authorized\_keys file:

mkdir /root/.ssh

cat id\_rsa.pub > /root/.ssh/authorized\_keys

chmod 0600 /root/.ssh/authorized\_keys

1. Convert your private dropbear key to OpenSSH format:

dropbearconvert dropbear openssh id\_dropbear id\_rsa

**Backup your private and public keys on USB drive (example):**

1. Insert your USB drive into router.
2. Check its mount point (should be /mnt/sda1 or /mnt/sdb1 or /mnt/sdc1 etc.)
3. Backup your private and public RSA keys to USB drive, to folder /ssh:

mkdir /mnt/sda1/ssh

cp /tmp/ssh/\* /mnt/sda1/ssh

**Convert your OpenSSH key to putty format PPK, for use from MS Windows (optional):**

1. Run puttygen.exe (PC with Windows).
2. Select in puttygen’s menu “Conversion->Import key”.
3. Browse and choose you private key in OpenSSH format (id\_rsa but not id\_rsa.pub).
4. [Optional] Correct key comment to what-you-wish.
5. Press “Save private key” button and save your PPK file for use it with putty to enter by SSH to router.

**Generate your own dropbear host keys (optional):**

1. Enter to your router console by telnet or by SSH.
2. Generate RSA and ECDSA host keys:

dropbearkey -t rsa -s 2048 -f /tmp/ssh/dropbear\_rsa\_host\_key

dropbearkey -t ecdsa -s 521 -f /tmp/ssh/dropbear\_ecdsa\_host\_key

chmod 0600 /tmp/dropbear\_\*\_host\_key

1. Copy your generated keys to /etc/dropbear directory:

cp -p -f /tmp/ssh/dropbear\_\*\_host\_key /etc/dropbear

1. Reboot your router.

Appendix B. OpenVPN Client setup example.

It is copied from this link:

https://www.myopenrouter.com/forum/openvpn-client-setup-guide-using-voxels-firmware-nighthawk-x4s-r7800

I would like to share my configuration and setup with people who want a secure, private and stable connection to the internet using an OpenVPN client connection to the internet on this forum.

I would like to thank the following people:

Voxel - for his excelent firmware and pointing me in the right direction when I had no clue where to begin.

kinakuta - for his insight and scripts for maintaining the OpenVPN tunnel always-on and the bypass VPN tunnel functionality.

Sven Taylor - for sharing honest and vital VPN information, views and reviews on https://restoreprivacy.com

I received my Netgear R7800 X4S in early December and didn't waste more than an hour on the stock firmware.

I flashed the latest Voxel's Custom Firmware for this router - https://www.voxel-firmware.com/Downloads/Voxel/R7800-Voxel-firmware

The mandatory and consice README is provided (https://www.voxel-firmware.com/Downloads/Voxel/readme.docx).

After flashing Voxel's firmware, don't forget to restore factory settings in the router WebGUI.

Start by setting up the following:

* 1. **SSH** access to router (Well documented in Voxel's README)
	2. Setup of **Entware** on a USB stick (Documented in Voxel's README)

PS: The **crontab** provided by Entware is essential.

PS2: I chose to use Voxel's optimized repository (https://www.voxel-firmware.com/Downloads/Voxel/Entware/entware-cortex-a1...)

**Configure DNS and DNSCRYPT**

DNS queries are THE primary source of your ISP's tracking strategy. I highly recommend NOT using their DNS servers.

* 1. Configure your DNS servers in the WEBGUI

I used 208.67.222.222, 208.67.220.220 and 8.8.8.8 as the DNS servers.

* 1. Enable dnscrypt-proxy (Documented in Voxel's README)

Simply edit /etc/dnscrypt.conf with one entry "adguard-dns" to wipe out any and all publicity.

Don't forget to test DNS leaks (https://www.dnsleaktest.com/) and make sure you do NOT use any of your ISP's DNS servers.

**OpenVPN client**

The reasons why I chose Voxel's firmware was because it maintains NETGEAR's propriety (and speedy) drivers, all stock functionality (ReadyShare, QoS, DNLA, etc.) and adds the OpenVPN client functionality. Centralizing the VPN client connection on the router guarantees encrypted internet access on all connected devices in your home.

Don't forget to follow Voxel's README.

* 1. Download your VPN providers OVPN file and place them in the /etc/openvpn/config/client directory

PS: Use full path directory filenames on any referenced files in the OVPN file. Example: change "auth-user-pass credentials.txt" to "auth-user-pass /etc/openvpn/config/client/credentials.txt"

* 1. Test "/etc/init.d/openvpn-client start/stop" thoroughly and read the log file /var/log/openvpn-client.log before you advance.

**Bypassing OpenVPN client tunnel** (Thank you, kinakuta)

You can bypass the OpenVPN client tunnel of the outgoing traffic for specific IP's in two simple steps:

* 1. Reserve DHCP addresses in the WEBGUI (Advanced -> Configuration -> LAN Configuration)
	2. Change the /etc/openvpn/ovpnclient-up.sh file to:

#!/bin/sh

# Don't forget to reserve the list of IPs for exclusion devices on the DHCP server

# Edit the following IP list to bypass the VPN. Seperate individual IP's using a single space between them.

NO\_VPN\_LST="192.168.1.7 192.168.1.3"

WAN\_GWAY=`nvram get wan\_gateway`

for excludeip in $NO\_VPN\_LST; do

 /usr/sbin/ip rule add from $excludeip table 200

done

/usr/sbin/ip route add table 200 default via $WAN\_GWAY dev brwan

/usr/sbin/ip route flush cache

exit 0

**Create a OpenVPN client tunnel monitoring script** (Thank you, kinakuta)

The OpenVPN client connection can sometimes disconnect or even cease to respond.

/usr/bin/vpncmon.sh:

#!/bin/sh

IP\_FOR\_TEST="8.8.8.8"

PING\_COUNT=1

INTERFACE="tun0"

FFLAG="/tmp/vpn\_stuck.fflg"

LOGFILE="/var/log/vpncmon.log"

NOW=$(date +"%H:%M, %d-%m-%Y")

restartvpnc()

{

 /etc/init.d/openvpn-client restart

 /bin/sleep 5

 /etc/init.d/dnscrypt-proxy restart

}

# check logfile

if [ ! -f $LOGFILE ]; then

 /bin/touch $LOGFILE

 /bin/echo "$NOW - VPN client LOGFILE $LOGFILE created.\n" >> $LOGFILE

fi

#Check if date is at least 2016 to validade VPN certificates

YEAR=`date "+%Y"`

while [ $YEAR -le 2016 ]; do

 /bin/echo "We do not have a valid date.\n" >> $LOGFILE

 /etc/init.d/ntpclient stop

 /usr/sbin/ntpclient -s -h pool.ntp.org

 /bin/sleep 2

 /etc/init.d/ntpclient start

 NOW=$(date +"%H:%M, %d-%m-%Y")

 YEAR=`date "+%Y"`

done

# check if interface is up

FOUND=`grep "$INTERFACE" /proc/net/dev`

if [ ! "$FOUND" ]; then

 /bin/echo "$NOW - $INTERFACE not up, restarting OpenVPN client.\n" >> $LOGFILE

 restartvpnc

fi

# check if successful with ping test

/bin/ping -c $PING\_COUNT $IP\_FOR\_TEST 2>/dev/null 1>/dev/null

if [ $? -ne 0 ]; then

 if [ -f $FFLAG ]; then

 /bin/echo "$NOW - Network and OpenVPN client down. Rebooting router!\n" >> $LOGFILE

 /bin/rm -f $FFLAG 2>/dev/null

 /opt/sbin/reboot

 else

 /bin/touch $FFLAG

 /bin/echo "$NOW - IP $IP\_FOR\_TEST can't be pinged, restarting OpenVPN client.\n" >> $LOGFILE

 restartvpnc

 fi

else

 if [ -f $FFLAG ]; then

 /bin/rm -f $FFLAG # 2>/dev/null

 fi

fi

exit 0

All that is left to do is **automate the script execution**

1. Change /etc/rc.local to run /usr/bin/vpncmon.sh on every boot

/etc/rc.local:

# Put your custom commands here that should be executed once

# the system init finished. By default this file does nothing.

/usr/bin/vpncmon.sh

exit 0

1. Add a crontab entry to run it every 5 minutes:

\*/5 \* \* \* \* /usr/bin/vpncmon.sh

PS: The Entware crontab is mandatory

I hope this guide helps somebody.

Voxel.