OpenWRT and the Linksys WRT 1900 ACS Router

Linksys WRT 1900 ACS OpenWRT Firmware

OpenWrt
Wireless Freedom

Linksys WRT 1900 ACS OpenWRT Firmware

Default OpenWRT Admin Interface

http://192.168.1.1

Additional Package Installation

Install:

1. Huawei E3372 Hi-Link LTE Dongle: kmod-usb-net-cdc-ether usb-modeswitch
2. Mosquitto: mosquitto-nossl

3. Material Theme: luci-theme-material
4. Additional Packages: nano, git, htop

```
opkg update && opkg install kmod-usb-net-cdc-ether usb-modeswitch luci-theme-material mosquitto-no
```

Update Existing Packages

```
opkg list-upgradable | cut -f 1 -d ' ' | xargs opkg upgrade
```

Configure Huawei E3372 Hi-Link LTE Dongle

Using the Huawei E3372 Hi-Link LTE Dongle with OpenWRT

Plug in Huawei E3372 Hi-Link LTE Dongle to USB1

From Network -> Interfaces -> Add New Interface

- Name of new interface: Dongle
- Protocol: DHCP Client
- Cover the following interface: eth2

From Network -> Interfaces -> Edit Dongle -> Firewall Settings

- Add Dongle to **wan** group

Setting WAN Priority

Generally want to use Broadband first then fall back to LTE Dongle. This is done by setting a gateway metric - low numbers = highest priority

Network -> Interfaces -> WAN -> Advanced Settings

- Use gateway metric: 10

Network -> Interfaces -> Dongle -> Advanced Settings

- Use gateway metric: 20

Activate Huawei E3372 Hi-Link LTE Dongle

1. Ensure dongle is activated with the SIM Carrier
2. On Dongle Teal light will glow solidly (no blink)
3. You may need to disconnect Internet/Broadband connection and reboot
4. Try **Stopping** and **Connecting** the Dongle Interface
5. Eventually Dongle will provide a DHCP address to the router
   Similar to the following:

```
Protocol: DHCP client
Uptime: 0h 15m 11s
MAC: 0C:5B:8F:27:9A:64
RX: 28.90 MB (21750 Pkts.)
TX: 1.22 MB (11751 Pkts.)
IPv4: 192.168.8.100/24
```

Huawei E3372 Hi-Link LTE Data Usage (on Optus Australia)

1. http://192.168.8.1
2. From Dongle Admin interface, select SMS
3. Text 1 to 9999
4. Review information in the reply

Extending OpenWRT with USB Flash Storage

Extroot configuration

To maximize performance use a USB3 flash. SanDisk and PNY sell small devices Flash devices that sit near flush on the back of the router.

USB Flash storage used as follows:

1. Extend the base system storage

2. FTP mount point

Easiest from Ubuntu Disk app. Create two partitions on the USB Flash and then plug Flash drive into the router.

Routers with ≥ 8 MiB flash

These devices should have enough space to install the packages we need.
Remove all packages you have installed to add functionality, as they are only wasting space now. After you make the extroot you will have all space you need.

From the command line interface write (on a single line):

```
opkg update && opkg install block-mount kmod-fs-ext4 kmod-usb-storage e2fsprogs kmod-usb-ohci kmod
```

This installs packages needed for a partition with ext4 filesystem.

Risk-adverse users may wish to create a custom image (as described in the pervious section) containing these tools and especially the kernel modules that are consistent with the firmware kernel so that they are available in failsafe mode.

2. Configuring rootfs_data

Connect with ssh to the device.

Configure `/etc/config/fstab` to mount the `rootfs_data` in another directory in case you need to access the original root overlay to change your extroot settings. Run the following commands:

```
DEVICE="$(awk -e '/\s\/overlay\s/{print $1}' /etc/mtab)"
uci -q delete fstab.rwm
uci set fstab.rwm="mount"
uci set fstab.rwm.device="${DEVICE}"
uci set fstab.rwm.target="/rwm"
uci commit fstab
```

Or you can identify the `rootfs_data` partition manually:

```
grep -e rootfs_data /proc/mtd
```

The directory `/rwm` will contain the original root overlay, which is used as the main root overlay until the extroot is up and running.
Later you can edit `/rwm/upper/etc/config/fstab` to change your extroot configuration (or temporarily disable it) should you ever need to.

3. Configuring extroot

See what partitions you have:

```
# block info
/dev/mtdblock2: UUID="9fd43c61-c3f2c38f-13440ce7-53f0d42d" VERSION="4.0" MOUNT="/rom" TYPE="squash
/dev/mtdblock3: MOUNT="/overlay" TYPE="jffs2"
/dev/sda1: UUID="fdacc9f1-0e0e-45ab-acee-9cb9cc8d7d49" VERSION="1.4" TYPE="ext4"
```

Here we see `mtdblock` devices (partitions in internal flash memory), and a partition on `/dev/sda1` that is on a usb flash drive.

Format the partition `/dev/sda1` as ext4 if required:

```
mkfs.ext4 /dev/sda1
```

Now we configure `/dev/sda1` as new overlay via fstab uci subsystem:

```
DEVICE="/dev/sda1"
eval $(block info "${DEVICE}" | grep -o -e "UUID=\S*")
uci -q delete fstab.overlay
uci set fstab.overlay="mount"
uci set fstab.overlay.uuid="${UUID}"
uci set fstab.overlay.target="/overlay"
uci commit fstab
```

If you have a swap partition it will also get recognized and added automatically.

4. Transferring the data

We now transfer the content of the current overlay inside the external drive:

```
mount /dev/sda1 /mnt
cp -a -f /overlay/. /mnt
umount /mnt
```

Reboot the device:

```
reboot
```

Testing

Web interface

1. **LuCI** → **System** → **Mount Points** should show USB partition mounted as `overlay`.
2. **LuCI** → **System** → **Software** should show free space of overlay partition.

Command-line interface

The USB partition should be mounted to `/overlay`.
Free space for `/` should be the same as `/overlay`.

```
# grep -e /overlay /etc/mtab
/dev/sda1 /overlay ext4 rw,relatime,data=ordered
overlayfs:/overlay / overlay rw,noatime,lowerdir=/,upperdir=/overlay/upper,workdir=/overlay/work

# df /overlay /
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1              7759872    477328   7221104   6% /overlay
overlayfs:/overlay     7759872    477328   7221104   6% /
```

Install Very Secure FTP Server (vsftpd) with Anonymous Access

Very Secure FTP Server

Install:

```
opkg update && opkg install vsftpd
mkdir /ftp
```

nano ***/etc/vsftpd.conf***

Config assumes ftp directory is located at **/ftp**

replace existing contents with thg following

```
background=YES
listen=YES
anonymous_enable=YES
ftp_username=nobody
anon_root=/ftp
no_anon_password=YES
local_enable=NO
write_enable=NO
local_umask=022
check_shell=NO
dirmessage_enable=YES
ftpd_banner=Welcome to Glovebox FTP service.
session_support=NO
#syslog_enable=YES
#userlist_enable=YES
#userlist_deny=NO
#userlist_file=/etc/vsftpd/vsftpd.users
#xferlog_enable=YES
#xferlog_file=/var/log/vsftpd.log
#xferlog_std_format=YES
###
### TLS/SSL options
### example key generation: openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/vsftpd/vsftpd_privkey.p
#ssl_enable=YES
#allow_anon_ssl=NO
#force_local_data_ssl=NO
#force_local_logins_ssl=NO
#ssl_tlsv1=YES
#ssl_sslv2=NO
#ssl_sslv3=NO
#rsa_cert_file=/etc/vsftpd/vsftpd_cert.pem
#rsa_private_key_file=/etc/vsftpd/vsftpd_privkey.pem
```

Reload and Start Very Secure FTP Services

```
/etc/init.d/vsftpd reload
/etc/init.d/vsftpd start
```

Install wget

```
opkg update && opkg install wget libustream-openssl ca-bundle ca-certificates
```

GitHub

Note: use **git:** rather than **https:** transport

```
git clone --depth 1 git://github.com/gloveboxes/PyLab-0-Raspberry-Pi-Set-Up
```

Python3 and Pip Usage

```
python3 -m pip install ...
```

Example:

```
python3 -m pip install flask paho-mqtt
```

CPU Temperature

```
cat /sys/class/thermal/thermal_zone0/temp
```

Connect to WPA2-Enterprise (WPA2 - 802.1X)

Connect to a WPA2-Enterprise

```
opkg update && opkg remove wpad-mini && opkg install wpad
```