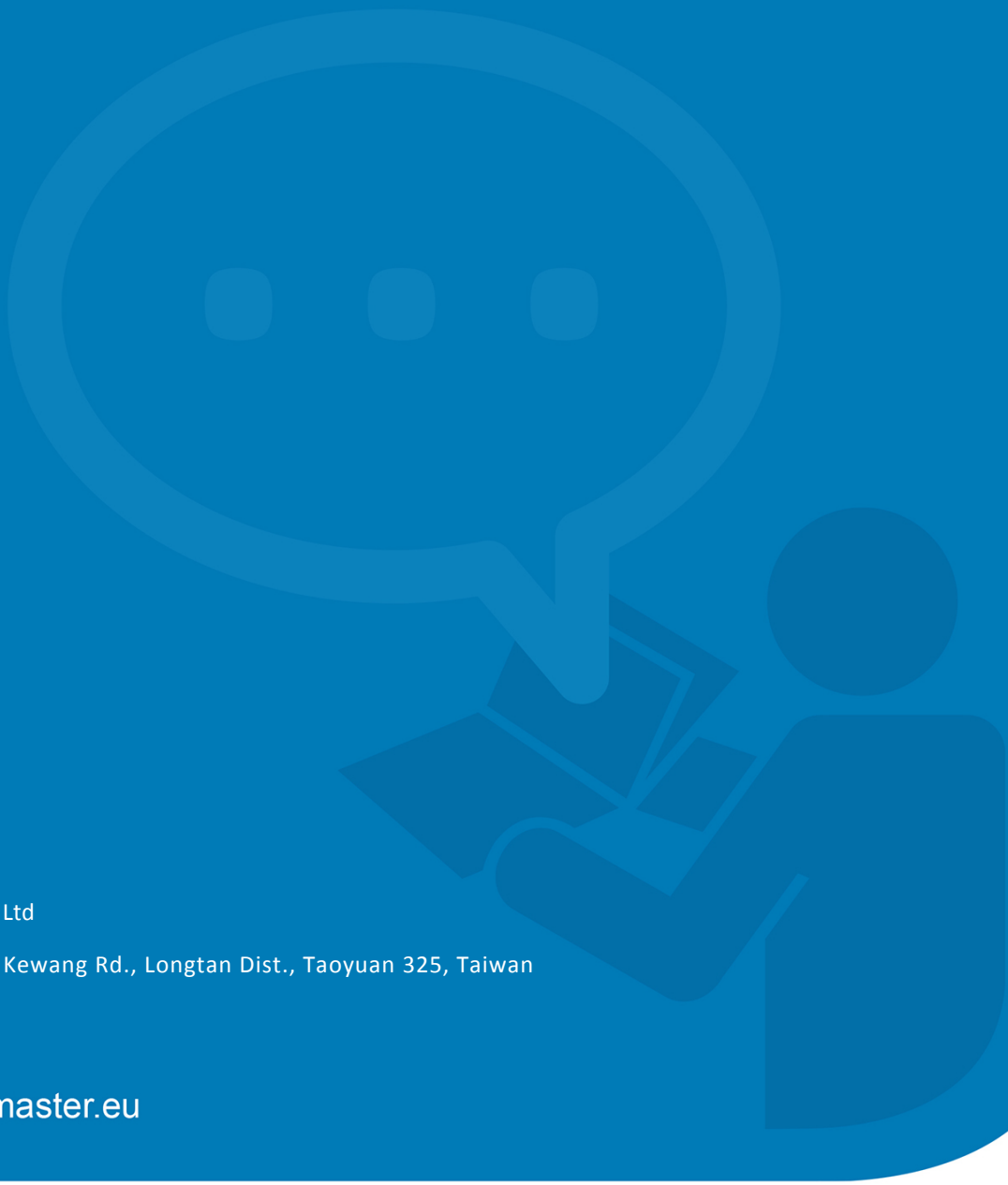
WoMaster

User Manual

IIoT Edge Computer Series

Feb.25.2019 V.1.1



WOM ASIA Co., Ltd

1F., No.185-3, Kewang Rd., Longtan Dist., Taoyuan 325, Taiwan

www.womaster.eu

WoMaster

IIoT Edge Computer Series

User Manual

About This Manual

This user manual is intended to guide a professional installer to install and to configure the IIoT Edge Computer Series. It includes procedures to assist you in avoiding unforeseen problems.

NOTE:

Only qualified and trained personnel should be involved with installation, inspection, and repairs of this router.

Disclaimer

WoMaster reserves the right to make changes to this Manual or to the product hardware at any time without notice. Information provided here is intended to be accurate and reliable. However, it might not cover all details and variations in the equipment and does not claim to provide for every possible contingency met in the process of installation, operation, or maintenance. Should further information be required or should particular problem arise which are not covered sufficiently for the user's purposes, the matter should be referred to WoMaster. Users must be aware that updates and amendments will be made from time to time to add new information and/or correct possible unintentional technical or typographical mistakes. It is the user's responsibility to determine whether there have been any such updates or amendments of the Manual. WoMaster assumes no responsibility for its use by the third parties.

WoMaster Online Technical Services

At WoMaster, you can use the online service forms to request the support. The submitted forms are stored in server for WoMaster team member to assign tasks and monitor the status of your service. Please feel free to write to help@womaster.eu if you encounter any problems.

TABLE OF CONTENTS

1. OVERVIEW

1.1 INTRODUCTION

The WR302G-EC/WR312G-EC/WR322GR-EC edge computing platform is designed for embedded data acquisition applications. The computer comes with one or two software selectable RS-232/422/485 full-signal serial ports and 2 10/100/1000 Mbps Ethernet ports, as well as optional one or two Mini PCIe socket for Wi-Fi/Cellular modules. These versatile communication capabilities let users efficiently adapt the WR302G-EC/WR312G-EC/WR322GR-EC to a variety of complex communications solutions. The WR302G-EC/WR312G-EC/WR322GR-EC is built around a QCA9558 MIPS-based processor that is widely applicable to a variety of industrial solutions. With built in Node-RED flow-based programming, this tiny embedded computer is a reliable and secure gateway for data acquisition and processing at field sites as well as a user friendly communication platform for many other large-scale deployments.

1.2 MODEL NAME

Model Name	Description
WR302G-EC	Industrial Edge Computing Secure Serial Server, 2GbE+2COM, USB, SD
WR312G-WLAN-EC	Industrial Secure Wireless Edge Computer, 2GbE+2COM, USB, SD, 802.11ac/n WLAN
WR312G-LTE-E-EC	Industrial Secure Cellular Edge Computer, 2GbE+2COM, USB, SD, LTE-E, 1SIM, FDD B1/3/5/7/8/20, TDD B38/40/41
WR322GR-WLAN+LTE-E-EC	Industrial Secure Cellular Edge Computer, 2GbE+2COM, USB, SD, 802.11ac/n WLAN, LTE-E, GPS, 2SIM, FDD B1/3/5/7/8/20, TDD B38/40/41
	*Embedded SIM by request *LTE-AU/LTE-U Cat.4 by request *LTE-AP/LTE-U Cat.6 by request *Dual LTE concurrent by request *GPS support for WR312G-LTE-E-EC series by request

2. GETTING START

In this chapter, we describe how to configure the basic settings WoMaster's Edge computers.

The following topics are covered in this chapter:

- **Connecting to the Edge Computer**
  - Connecting Through the SSH Console
- **Network Settings**
  - Configuring Network Setting over the Network
- **Determining Available Drive Space**
- **Shutting Down the Device**

2.1 CONNECTING TO EDGE COMPUTER

You will need another computer to connect to the Arm-based computer and log on to the command line interface.

There are two ways to connect: through serial console cable or through Ethernet cable.

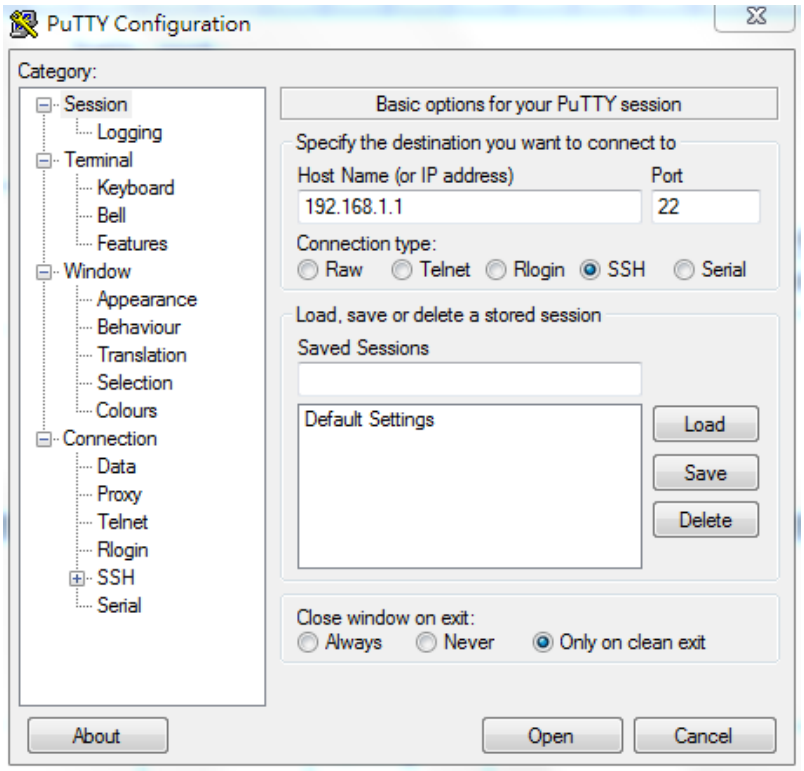
The default login username and password are:

login : root

Password: (no password)

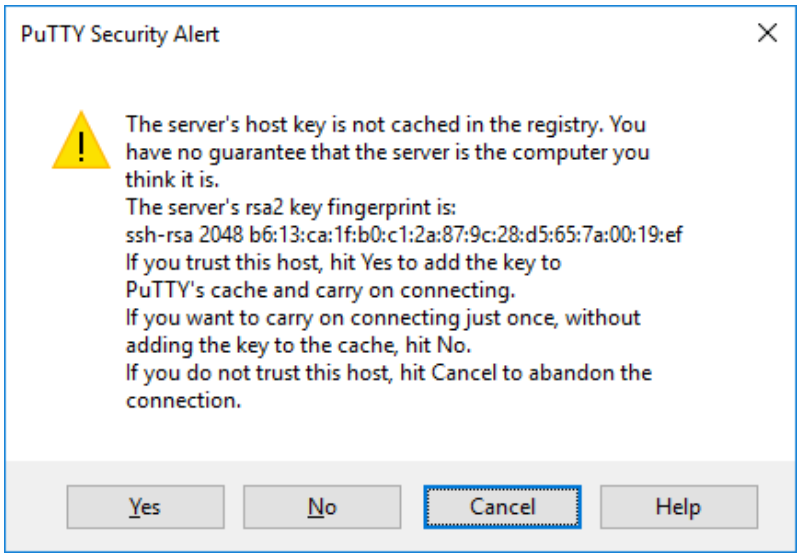
2.1.1 SSH CONSOLE

The Edge Computer supports an SSH Console by default to offer users with better security options. On PC, click on the link putty to download PuTTy(freeware) and set up an SSH console for EC Series in a Windows environment. The following figure shows an example of the configuration that is required.

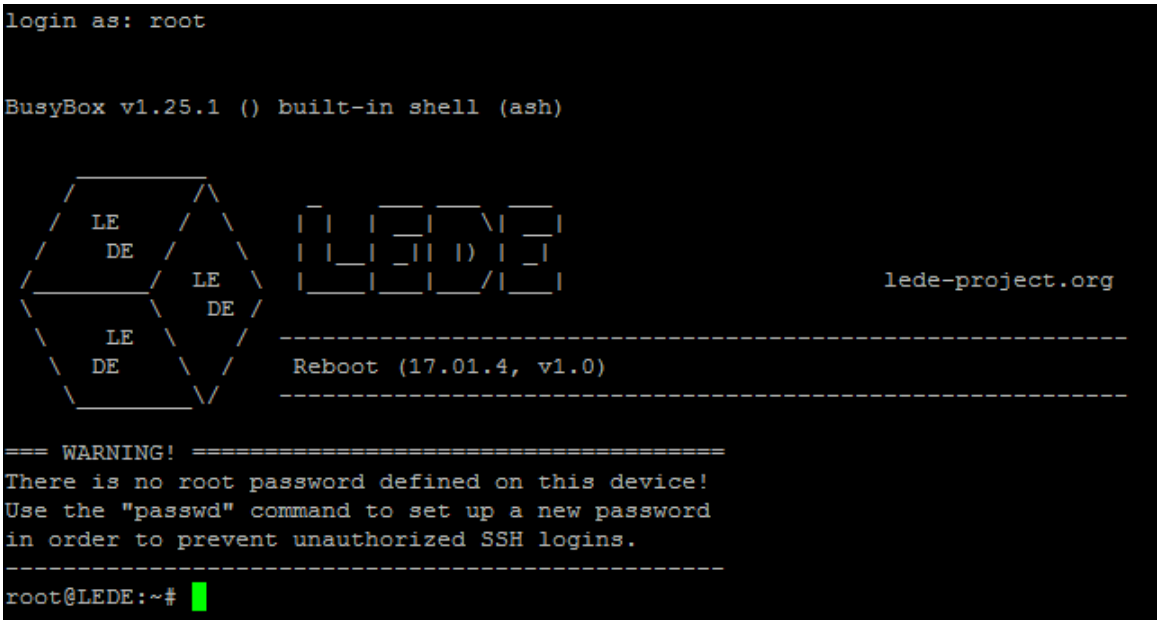


Windows PuTTy setting

Click Yes to add the key to PuTTY's cache and carry on connecting.



Then Login with the Username and Password. (Username: root, Password: admin)



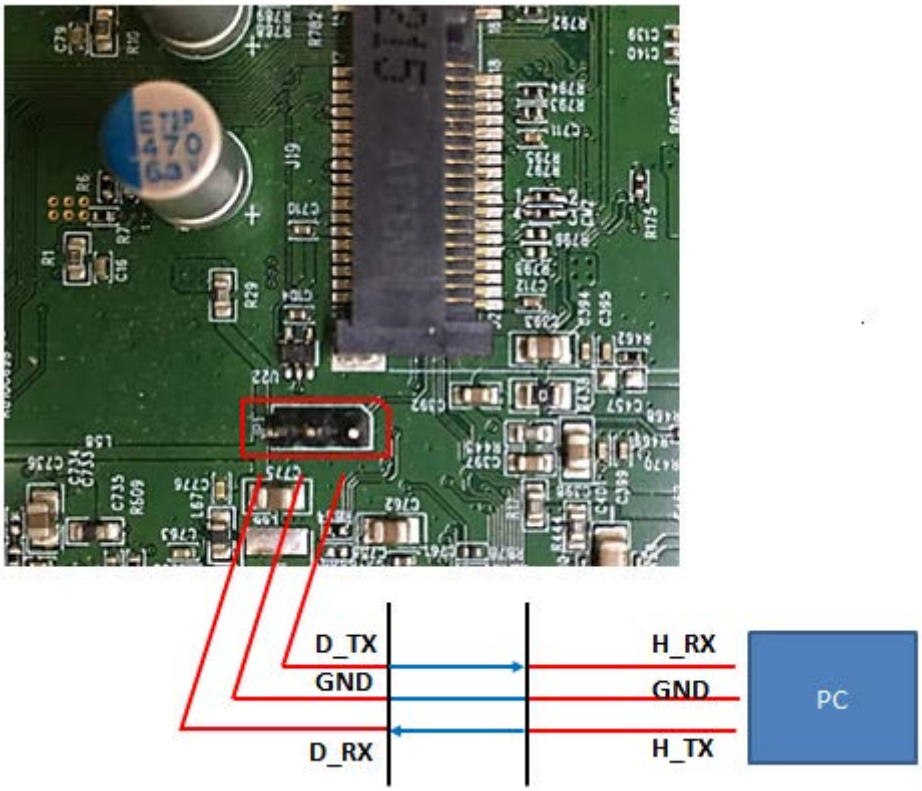
The MIPS-based computer supports SSH connections over an Ethernet network. Use the following default IP addresses to connect to the MIPS-based computer.

Connecting through the SSH Console

Port	Linux	Bridge	Description	Default IP
Port 1	eth0	br-wan	WAN	DHCP
Port 2	eth1	br-lan	LAN	192.168.1.1

2.1.2 TTL UART CONSOLE

For this connection user needs to open the box and using a suitable cable, such as the USB TTL Serial cable, user can connect it to the PC and using some simple terminal software set to 115200-8-N-1 use the command line interface to the WR322GR-EC in the same way as if user uses a keyboard and screen connected to it.



Note: The peripheral works with 3.3V logic levels, wiring to wrong voltage or wrong pin will damage the main board.

User can use any kind of terminal emulator software, to get into the console interface.

```
Connecting to COM6...
Connected.



BusyBox v1.25.1 () built-in shell (ash)

     _________
    /         /\      _    ___ ___  ___
   /  LE     /  \    | |  | __|   \| __|
  /     DE  /    \   | |__| _|| |) | _|
 /_________/  LE  \  |____|___|___/|___|                    lede-project.org
 \         \   DE /
  \    LE   \    /  -----------------------------------------------------
   \   DE    \  /    Reboot (17.01.4, v1.0)
    \_________\/    -----------------------------------------------------

root@LEDE:/#
```

2.2 CONFIGURE NETWORK SETTING

The central network configuration is located in the file /etc/config/network. This configuration file is responsible for interface configurations and network routes. After editing and saving /etc/config/network you need to execute

```
/etc/init.d/network reload
```

To stop and restart the network before any changes take effect. Rebooting the router is not necessary

.

1. Show current network setting with "**uci show network**"

```
root@LEDE:~# uci show network
network.loopback=interface
network.loopback.ifname='lo'
network.loopback.proto='static'
network.loopback.ipaddr='127.0.0.1'
network.loopback.netmask='255.0.0.0'
network.globals=globals
network.globals.ula_prefix='fd99:4997:04c3::/48'
network.lan=interface
network.lan.type='bridge'
network.lan.ifname='eth1'
network.lan.proto='static'
network.lan.netmask='255.255.255.0'
network.lan.ip6assign='60'
network.lan.ipaddr='192.168.10.3'
network.lan.gateway='192.168.10.254'
network.wan=interface
network.wan.ifname='eth0'
network.wan.proto='dhcp'
network.wan.type='bridge'
network.wan6=interface
network.wan6.ifname='eth0'
```

If you only want to show lan's setting, you can type "**uci show network.lan**"

```
root@LEDE:~# uci show network.lan
network.lan=interface
network.lan.type='bridge'
network.lan.ifname='eth1'
network.lan.proto='static'
network.lan.netmask='255.255.255.0'
```

```
network.lan.ip6assign='60'
network.lan.ipaddr='192.168.10.3'
network.lan.gateway='192.168.10.254'
```

2. To change lan's ip address to 192.168.10.4

```
root@LEDE:~# uci set network.lan.ipaddr=192.168.10.4
```

You can check the config again with "uci get"

```
root@LEDE:~# uci get network.lan.ipaddr
192.168.10.4
```

To apply the network settings

```
/etc/init.d/network reload
```

Reference

https://openwrt.org/docs/guide-user/base-system/basic-networking

https://oldwiki.archive.openwrt.org/doc/uci/network

2.2.1 MODIFYING NETWORK SETTINGS OVER THE NETWORK

Same the previous section, IP settings can be modified over the network, too. There is another way to change the IP address without modifying the file /etc/config/network, but the new settings will **not** be saved to the flash disk.

For example, type the command **#ifconfig eth1 192.168.10.4** to change the IP address of LAN interface to 192.168.10.4.

```
root@LEDE:~# ifconfig eth1 192.168.10.4
root@LEDE:~# ifconfig
br-lan    Link encap:Ethernet  HWaddr 94:66:E7:00:0D:A7
          inet addr:192.168.10.4  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::9666:e7ff:fe00:da7/64 Scope:Link
          inet6 addr: fdf0:7b1a:a679::1/60 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2178 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1803 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:459387 (448.6 KiB)  TX bytes:110566 (107.9 KiB)

eth0      Link encap:Ethernet  HWaddr 94:66:E7:00:0D:A6
          inet addr:192.168.10.5  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:643 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

```
          RX bytes:147750 (144.2 KiB)  TX bytes:20270 (19.7 KiB)
          Interrupt:4

eth1      Link encap:Ethernet  HWaddr 94:66:E7:00:0D:A7
          inet addr:192.168.10.4  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7385 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6956 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1580144 (1.5 MiB)  TX bytes:559221 (546.1 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2905 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2905 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:255279 (249.2 KiB)  TX bytes:255279 (249.2 KiB)

wlan0     Link encap:Ethernet  HWaddr 04:F0:21:3B:8A:04
          inet addr:192.168.1.249  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::6f0:21ff:fe3b:8a04/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2888 errors:0 dropped:0 overruns:0 frame:0
          TX packets:39 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:455800 (445.1 KiB)  TX bytes:4188 (4.0 KiB)
```

Network Setting over the Network

2.3 DETERMINING AVAILABLE DRIVE SPACE

To determine the amount of available drive space, use the **df** command with the **–h** parameter. The system will return the amount of drive space broken down by file system. Here is an example:

```
root@LEDE:~# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 9.8M      9.8M         0 100% /rom
tmpfs                   124.6M    536.0K    124.0M   0% /tmp
/dev/sda1                 7.2G    197.2M      6.6G   3% /overlay
overlayfs:/overlay        7.2G    197.2M      6.6G   3% /
tmpfs                   512.0K         0    512.0K   0% /dev
```

2.4 SHUTTING DOWN THE DEVICE

To shut down the device, disconnect the power source to the computer. When the computer is powered off, main components such as the CPU, RAM, and storage devices are powered off. You can use the command **poweroff** to close all software running on the device and halt the system. However, main components such as the CPU, RAM, and storage devices will continue to be powered after you run this command.

```
root@LEDE:~# poweroff now
```

3. SYSTEM FEATURE

This chapter includes information about version control, deployment, updates, and peripherals. The information in this chapter will be particularly useful when you need to run the same application.

3.1 SYSTEM VERSION

Querying the Firmware Version

To check the edge computer's firmware version, type:

```
root@LEDE:~# cat /etc/openwrt_version
v1.0
```

3.2 CURRENT RUNNING PROCESS

Type the command "**ps**" to list all processes currently running.

```
root@LEDE:~# ps
  PID USER       VSZ STAT COMMAND
    1 root      1608 S    /sbin/procd
    2 root         0 SW   [kthreadd]
    3 root         0 SW   [ksoftirqd/0]
    5 root         0 SW<  [kworker/0:0H]
    6 root         0 SW   [kworker/u2:0]
   77 root         0 SW<  [writeback]
   78 root         0 SW<  [crypto]
   80 root         0 SW<  [bioset]
   81 root         0 SW<  [kblockd]
  111 root         0 SW   [kworker/0:1]
  118 root         0 SW   [kswapd0]
  170 root         0 SW   [fsnotify_mark]
  183 root         0 SW   [spi0]
  200 root         0 SW<  [bioset]
  205 root         0 SW<  [bioset]
  210 root         0 SW<  [bioset]
  215 root         0 SW<  [bioset]
  220 root         0 SW<  [bioset]
  225 root         0 SW<  [bioset]
  230 root         0 SW<  [bioset]
  272 root         0 SW<  [ipv6_addrconf]
```

```
  279 root      0 SW<   [deferwq]
  281 root      0 SW<   [kworker/0:1H]
  334 root      0 SW    [kworker/0:2]
  368 root      0 SW    [scsi_eh_0]
  369 root      0 SW<   [scsi_tmf_0]
  370 root      0 SW    [usb-storage]
  380 root      0 SW<   [bioset]
  405 root      0 SW    [jbd2/sda1-8]
  406 root      0 SW<   [ext4-rsv-conver]
  492 root   1176 S     /sbin/ubusd
  506 root    892 S     /sbin/askfirst /usr/libexec/login.sh
  616 root      0 SW<   [rpciod]
  642 root      0 SW<   [nfsiod]
  665 root      0 SW<   [cfg80211]
  674 root      0 SW<   [ath10k_wq]
  675 root      0 SW<   [ath10k_aux_wq]
 1015 root   1224 S     /sbin/logd -S 64
 1024 root   1440 S     /sbin/rpcd
 1117 root   1484 S     /usr/sbin/odhcpd
 1169 root   1056 S     /usr/sbin/dropbear -F -P /var/run/dropbear.1.pid -p 22 -K 300
 1204 root   3584 S     /usr/sbin/uhttpd -f -h /www -r LEDE -x /cgi-bin -u /ubus -t 60 -T 30 -k 20 -A 1 -n 3 -N
100 -R -p 0.0.0.0:80 -p [::]:80 -C /etc/uhttpd.crt -K /etc/uhttpd.key -s 0.
 1351 root   1104 S     /sbin/mountd -f
 1463 root   1748 S     /usr/lib/ipsec/starter --daemon charon
 1465 root   6208 S     /usr/lib/ipsec/charon --use-syslog
 1497 root   1412 S     /usr/bin/rsync --daemon --no-detach
 1516 root   153m S     node-red
 1574 root   1184 S <   /usr/sbin/ntpd -n -N -S /usr/sbin/ntpd-hotplug -p 0.lede.pool.ntp.org -p
1.lede.pool.ntp.org -p 2.lede.pool.ntp.org -p 3.lede.pool.ntp.org
 1922 dnsmasq  1052 S     /usr/sbin/dnsmasq -C /var/etc/dnsmasq.conf.cfg02411c -k -x
/var/run/dnsmasq/dnsmasq.cfg02411c.pid
 1968 root      0 SW    [kworker/u2:2]
 2173 root   1704 S     /sbin/netifd
 2469 root   3088 S     /usr/sbin/snmpd -Lf /dev/null -f
 2492 root   2408 S     /usr/sbin/smbd -F
 2493 root   2468 S     /usr/sbin/nmbd -F
 2500 root   1624 S     /usr/sbin/wpa_supplicant -B -P /var/run/wpa_supplicant-wlan0.pid -D nl80211 -i
wlan0 -c /var/run/wpa_supplicant-wlan0.conf -C /var/run/wpa_supplicant
```

```
 2648 root      1184 S     udhcpc -p /var/run/udhcpc-wlan0.pid -s /lib/netifd/dhcp.script -f -t 0 -i wlan0 -C -O
121
 2744 root       904 S     /usr/sbin/ntpclient -i 600 -s -l -D -p 123 -h 1.openwrt.pool.ntp.org
 2813 root      1124 S     /usr/sbin/dropbear -F -P /var/run/dropbear.1.pid -p 22 -K 300
 2814 root      1188 S     -ash
 5424 root      1192 S     ash
 9159 root      1184 R     ps
```

3.3 SETTING SYSTEM TIME

The edge computer has two time settings. One is the system time, and the other is the RTC (Real Time Clock) time kept by the hardware.

Use the **#date** command to query the current system time or set a new system time.

```
root@LEDE:~# date "2018-12-24 10:28" – yyyy-mm-dd hh:mm
Mon Dec 24 10:28:00 UTC 2018
mm = Month
dd = Date
hh:mm = hour and minute
yyyy = Year
```

Use **#hwclock** to query the current RTC time

Use the following command to set system time from hardware clock:

```
root@LEDE:~# hwclock -w
```

The following figure illustrates how to update the system time and set the RTC time.

```
root@LEDE:~# date "2018-12-24 10:57"
Mon Dec 24 10:57:00 UTC 2018
root@LEDE:~# hwclock -w
root@LEDE:~# date
Mon Dec 24 10:57:09 UTC 2018
```

Setting the Time Manually

```
Use root@LEDE:~# hwclock –h to see the help function
Usage:
  hwclock [function] [option...]


Query or set the hardware clock.
```

Functions:

-h, --help show this help text and exit

-r, --show read hardware clock and print result

--get read hardware clock and print drift corrected result

--set set the RTC to the time given with --date

-s, --hctosys set the system time from the hardware clock

-w, --systohc set the hardware clock from the current system time

--systz set the system time based on the current timezone

--adjust adjust the RTC to account for systematic drift since the clock was last set or adjusted

-c, --compare periodically compare the system clock with the CMOS clock

--getepoch print out the kernel's hardware clock epoch value

--setepoch set the kernel's hardware clock epoch value to the value given with --epoch

--predict predict RTC reading at time given with --date

-V, --version display version information and exit

Options:

-u, --utc the hardware clock is kept in UTC

--localtime the hardware clock is kept in local time

-f, --rtc <file> special /dev/... file to use instead of default

--directisa access the ISA bus directly instead of /dev/rtc

--badyear ignore RTC's year because the BIOS is broken

--date <time> specifies the time to which to set the hardware clock

--epoch <year> specifies the year which is the beginning of the hardware clock's epoch value

--update-drift update drift factor in /etc/adjtime (requires --set or --systohc)

--noadjfile do not access /etc/adjtime; this requires the use of either --utc or --localtime

--adjfile <file> specifies the path to the adjust file; the default is /etc/adjtime

--test do not update anything, just show what would happen

-D, --debug debugging mode

3.4 ADJUST SYSTEM TIME

NTP provides time synchronization based on a network of reference clocks located around the world. OpenWrt supports both NTP client protocol (to synchronize local time with a distant clock) and NTP server protocol (to deliver time to your local network).

The ntp configuration is located in system uci subsystem, and found in file /etc/config/system.

3.4.1 TIMESERVER SECTION

The NTP configuration is found in timeserver section of system uci subsystem.

```
root@LEDE:/# uci show system.ntp
system.ntp=timeserver
system.ntp.enabled='1'
system.ntp.enable_server='0'
system.ntp.server='0.lede.pool.ntp.org' '1.lede.pool.ntp.org' '2.lede.pool.ntp.org' '3.lede.pool.ntp.org'
```

By default, NTP client is enabled

```
root@LEDE:/# cat /etc/config/system
…
config timeserver 'ntp'
        option enabled '1'                      # NTP client is enabled
        option enable_server '0'
        list server '0.lede.pool.ntp.org'
        list server '1.lede.pool.ntp.org'
        list server '2.lede.pool.ntp.org'
        list server '3.lede.pool.ntp.org'
```

3.4.2 NTP CLIENT

If you only wish to synchronize your clock when the device boots up, you can use **ntpclient**. This may be appropriate for some devices which are frequently rebooted and only require infrequent synchronization.

Using **ntpclient** at boot time is also a good idea for devices that run **ntpd**. The **ntpd** program changes the clock gradually, whereas **ntpclient** sets the clock, no matter how great the difference between a device's current clock setting and the correct time.

The Edge Computer has a built-in NTP (Network Time Protocol) client that is used to initialize a time request to a remote NTP server.

Use #**ntpclient** to update the system time.

```
root@LEDE:~# ntpclient -h time.stdtime.gov.tw -c 1
43460 09625.766      8344.0       67.1       3355.8   36758.4      278237
root@LEDE:~# hwclock –w
```

NTP client request

Visit http://www.ntp.org for more information about NTP and NTP server addresses.

NOTE

Before using the NTP client utility, check your IP and DNS settings to make sure that an Internet connection is available.

3.4.2 NTP SERVER

To start the server, execute the command:

```
root@LEDE:~# ntpd -l
```

3.5 SETTING THE TIMEZONE AND HOSTNAME

To configure the embedded computer's **Timezone,** user can use the **TZ** variable.

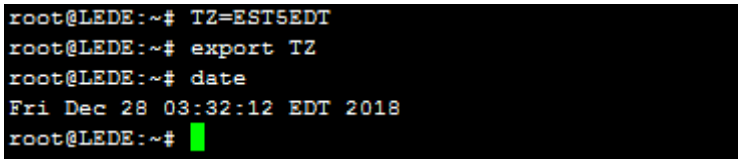
The format of the TZ environment variable looks like this:

TZ=<*Value*>*HH[:MM[:SS]][daylight[HH[:MM[:SS]]][,start date[/starttime], enddate[/endtime]]]*

Here are some possible settings for the North American Eastern time zone:

1. **TZ=EST5EDT**
2. **TZ=EST0EDT**
3. **TZ=EST0**

In the first case, the reference time is GMT and the stored time values are correct worldwide. A simple change of the TZ variable can print the local time correctly in any time zone. In the second case, the reference time is Eastern Standard Time and the only conversion performed is for Daylight Saving Time. Therefore, there is no need to adjust the hardware clock for Daylight Saving Time twice per year. In the third case, the reference time is always the time reported. You can use this option if the hardware clock on your machine automatically adjusts for Daylight Saving Time or you would like to manually adjust the hardware time twice a year.



The system UCI subsystem configuration file is located in ***/etc/config/system***.

The *system* section contains settings that apply to the most basic operation of the system, such as the hostname, the time zone, and how and where to write logging information to. The default settings are:

```
root@LEDE:/# uci show system
system.@system[0]=system
system.@system[0].hostname='LEDE'
system.@system[0].timezone='UTC'
system.@system[0].ttylogin='0'
system.@system[0].log_size='64'
system.@system[0].urandom_seed='0'
system.ntp=timeserver
system.ntp.enabled='1'
system.ntp.enable_server='0'
system.ntp.server='0.lede.pool.ntp.org' '1.lede.pool.ntp.org' '2.lede.pool.ntp.org'
'3.lede.pool.ntp.org'
```

Change the system's timezone

root@LEDE:/# uci set system.@system[0].timezone="EAT-3"

Change the system's hostname

root@LEDE:/# uci set system.@system[0].hostname=WR322

Apply the system settings

root@LEDE:/# /etc/init.d/system reload

Save the changes to startup configuration

root@LEDE:/# uci commit system

```
root@LEDE:~# uci set system.@system[0].hostname=WR322
root@LEDE:~# /etc/init.d/system reload
root@WR322:~# uci commit system
```

Link Reference: https://openwrt.org/docs/guide-user/base-system/system_configuration

The following table lists other possible values for the TZ environment variable: **Hours from Greenwich Mean Time (GMT)**

	Value	Description
0	GMT	Greenwich Mean Time
+1	ECT	European Central Time
+2	EET	European Eastern Time
+2	ART	
+3	EAT	Saudi Arabia
+3.5	MET	Iran
+4	NET	
+5	PLT	West Asia
+5.5	IST	India
+6	BST	Central Asia
+7	VST	Bangkok
+8	CTT	China
+9	JST	Japan
+9.5	ACT	Central Australia
+10	AET	Eastern Australia
+11	SST	Central Pacific
+12	NST	New Zealand
-11	MIT	Samoa
-10	HST	Hawaii
-9	AST	Alaska
-8	PST	Pacific Standard Time
-7	PNT	Arizona
-7	MST	Mountain Standard Time

-6	CST	Central Standard Time
-5	EST	Eastern Standard Time
-5	IET	Indiana East
-4	PRT	Atlantic Standard Time
-3.5	CNT	Newfoundland
-3	AGT	Eastern South America
-3	BET	Eastern South America
-1	CAT	Azores

3.6 CONNECT PERIPHERALS

This chapter is included more information on the MIPS-based computer's peripherals, such as the serial interface, storage, diagnostic LEDs, and the cellular module. The instructions in this chapter cover all functions supported in WoMaster's Edge computers. Before referring to the sections in this chapter, make sure that they are applicable to and are supported by the hardware specification of your MIPS-based computer.

While plug-in a USB mass storage or a SD card, use **dmesg** command can help showing USB-storage device status.

usb-storage device scan status

To check the external storage, just use ***mount*** command as following:

```
root@LEDE:~# mount
/dev/mtdblock2 on /rom type squashfs (ro,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,noatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,noatime)
tmpfs on /tmp type tmpfs (rw,nosuid,nodev,noatime)
/dev/sda1 on /overlay type ext4 (rw,relatime,data=ordered)
overlayfs:/overlay on / type overlay (rw,noatime,lowerdir=/,upperdir=/overlay/upper,workdir=/overlay/work)
tmpfs on /dev type tmpfs (rw,nosuid,relatime,size=512k,mode=755)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,mode=600)
debugfs on /sys/kernel/debug type debugfs (rw,noatime)
mountd(pid1351) on /tmp/run/mountd type autofs
(rw,relatime,fd=5,pgrp=1349,timeout=60,minproto=5,maxproto=5,indirect)
```

To manually mount a usb-storage, execute

```
root@LEDE:~# mount <device path> <mount path>
```

To manually un-mount the usb-storage, execute

```
root@LEDE:~# umount <mount path>
```

NOTE

To be able to unmount a device, you have to close all the open files in it.

Type **sync** can help commits all pending writes, which can then be removed in a safe way.

Check the mount command line:

```
mount -h

Usage:
 mount [-lhV]
 mount -a [options]
 mount [options] [--source] <source> | [--target] <directory>
 mount [options] <source> <directory>
 mount <operation> <mountpoint> [<target>]

Mount a filesystem.

Options:
 -a, --all                mount all filesystems mentioned in fstab
 -c, --no-canonicalize    don't canonicalize paths
 -f, --fake               dry run; skip the mount(2) syscall
 -F, --fork               fork off for each device (use with -a)
 -T, --fstab <path>       alternative file to /etc/fstab
 -i, --internal-only      don't call the mount.<type> helpers
 -l, --show-labels        show also filesystem labels
 -n, --no-mtab            don't write to /etc/mtab
 -o, --options <list>     comma-separated list of mount options
 -O, --test-opts <list>   limit the set of filesystems (use with -a)
 -r, --read-only          mount the filesystem read-only (same as -o ro)
 -t, --types <list>       limit the set of filesystem types
     --source <src>       explicitly specifies source (path, label, uuid)
     --target <target>    explicitly specifies mountpoint
 -v, --verbose            say what is being done
 -w, --rw, --read-write   mount the filesystem read-write (default)

 -h, --help       display this help and exit
 -V, --version    output version information and exit

Source:
 -L, --label <label>      synonym for LABEL=<label>
 -U, --uuid <uuid>        synonym for UUID=<uuid>
 LABEL=<label>            specifies device by filesystem label
 UUID=<uuid>              specifies device by filesystem UUID
```

```
 PARTLABEL=<label>          specifies device by partition label
 PARTUUID=<uuid>             specifies device by partition UUID
 <device>                   specifies device by path
 <directory>                mountpoint for bind mounts (see --bind/rbind)
 <file>                     regular file for loopdev setup

Operations:
 -B, --bind                mount a subtree somewhere else (same as -o bind)
 -M, --move                 move a subtree to some other place
 -R, --rbind               mount a subtree and all submounts somewhere else
 --make-shared              mark a subtree as shared
 --make-slave               mark a subtree as slave
 --make-private            mark a subtree as private
 --make-unbindable          mark a subtree as unbindable
 --make-rshared             recursively mark a whole subtree as shared
 --make-rslave             recursively mark a whole subtree as slave
 --make-rprivate           recursively mark a whole subtree as private
 --make-runbindable         recursively mark a whole subtree as unbindable
```

4. CELLULAR, GPS, AND WIRELESS CONNECTIVITY

4.1 CELLULAR

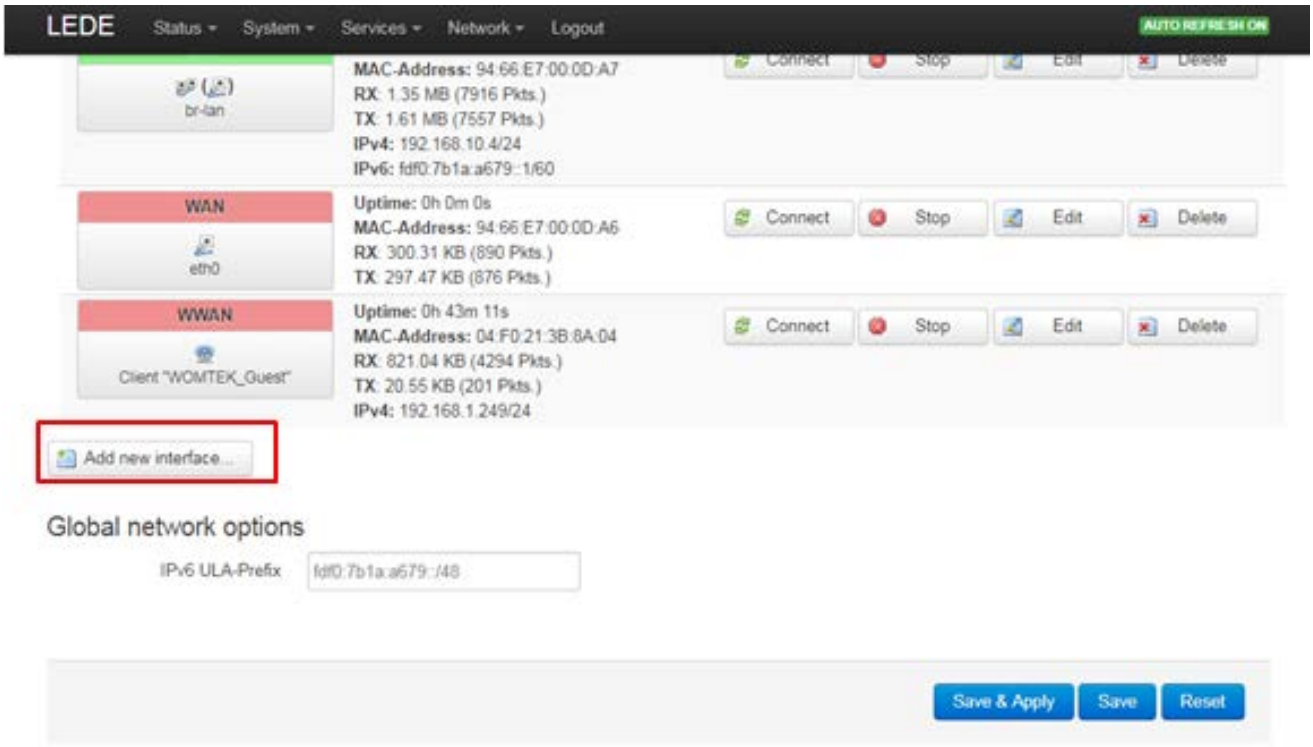
For the cellular feature, user needs to install and activate the function first. Follow the step below to activate the cellular. **Note: Make sure the SIM Card has been installed before power on the device.**

LTE module	SIM1	https://openwrt.org/docs/guide-user/network/wan/wwan/ltedongle Change to SIM1 (1.) cat /sys/class/gpio/gpio3/value (Default=0) (2.) echo 1 > /sys/class/gpio/gpio3/value (Change the SIM Card) (3.) microcom /dev/ttyUSB3 (Check the SIM Card) Type: at+cfun=1,1 Type: at+ccid Press CTRL+X (4.) ifconfig wwan0 up (5.) uqmi -d /dev/cdc-wdm0 --start-network internet --autoconnect (6.) Add DHCP client (add a wwan0 to the web page and select dhcp client), you can get the IP	
	SIM2	(Default SIM) (1.) ifconfig wwan0 up (2.) uqmi -d /dev/cdc-wdm0 --start-network internet --autoconnect (3.) Add DHCP client (add a wwan0 to the web page and select dhcp client), you can get the IP	GPIO3: 0 = Mylar SIM 2 GPIO3: 1 = Mylar SIM 1
	Connection	Establish the LTE connection uqmi -d /dev/cdc-wdm0 --start-network internet –autoconnect uqmi -d /dev/cdc-wdm0 --get-data-status "connected" uqmi -d /dev/cdc-wdm0 --get-data-status "disconnected"	
	Check sim	Microcosm /dev/ttyUSB3 Type: at+ccid If the sim is detected then it will show the sim card id.	

		root@Wr322:~# microcom /dev/ttyUSB3 at+ccid +CCID: 89886891000288047481 OK If the sim card cannot be detected, it will show the CME Error. root@Wr322:~# microcom /dev/ttyUSB3 at+ccid +CME ERROR: 13	

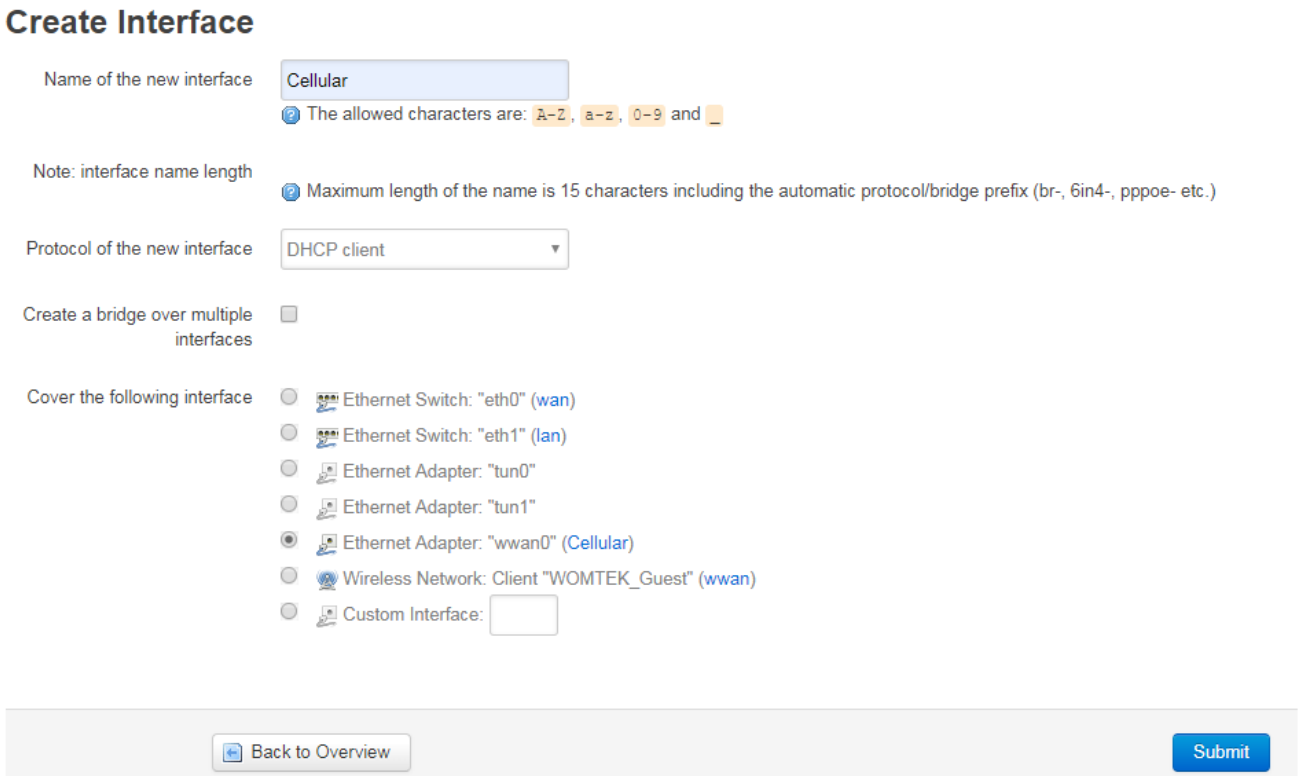
Add the DCHP Client or the Cellular Interface

Go to Network -> Interface

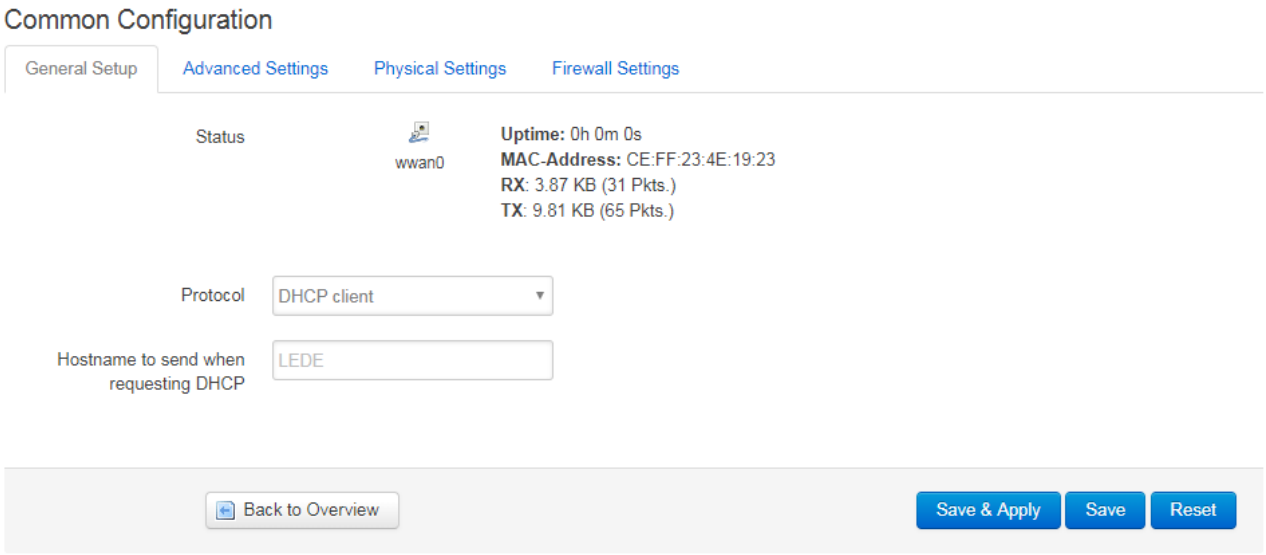


Configure the new interface

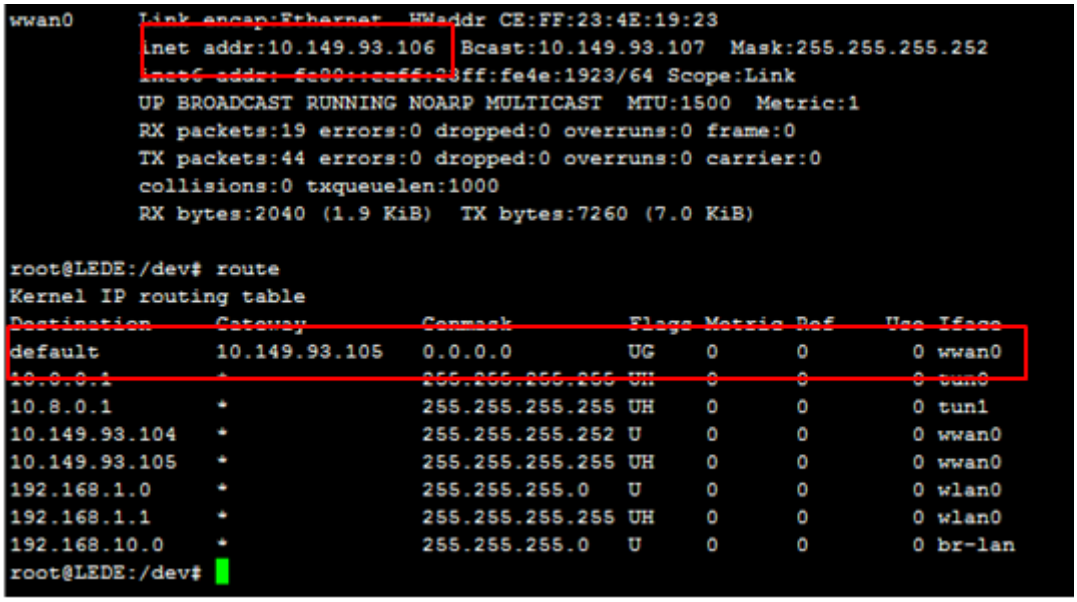
Name the interface and set the protocol to DHCP Client. Choose the “wwan0” interface. And click Submit.



Click Save & Apply to apply the configuration



Check the interface configuration, and user will see it gets the IP Address from the cellular interface.


wwan0 Link encap:Ethernet HWaddr CE:FF:23:4E:19:23
inet addr:10.149.93.106 Bcast:10.149.93.107 Mask:255.255.255.252
UP BROADCAST RUNNING NOARP MULTICAST MTU:1500 Metric:1
RX packets:19 errors:0 dropped:0 overruns:0 frame:0
TX packets:44 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:2040 (1.9 KiB) TX bytes:7260 (7.0 KiB)
root@LEDE:/dev# route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default 10.149.93.105 0.0.0.0 UG 0 0 0 wwan0
10.8.0.1 * 255.255.255.255 UH 0 0 0 tun1
10.149.93.104 * 255.255.255.252 U 0 0 0 wwan0
10.149.93.105 * 255.255.255.255 UH 0 0 0 wwan0
192.168.1.0 * 255.255.255.0 U 0 0 0 wlan0
192.168.1.1 * 255.255.255.255 UH 0 0 0 wlan0
192.168.10.0 * 255.255.255.0 U 0 0 0 br-lan
root@LEDE:/dev#

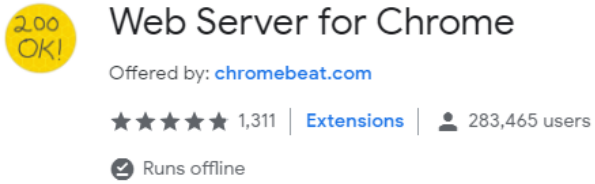
4.2 GPS

Install the GPS feature to the device.

1. **In your Chrome browser, open the link below**

   https://chrome.google.com/webstore/detail/web-server-for-chrome/ofhbbkphhbklhfoeikjpcbhemlocgigb

2. **Install and lunch the app (Web Server from Chrome)**



3. **Choose the web server's root dir. Copy the sdcard.tgz to the folder.**

4. **Login to device with ssh**
5. **Config the ip address with 'ifconfig br-lan <device's ip address>**
6. **Download gps.gcom to device with 'wget http://<user-pc-ip>:8887/gps.gcom .'**
7. **Type linux command 'sync' to make sure data has been write to the SD card.**
8. **Reboot and check**

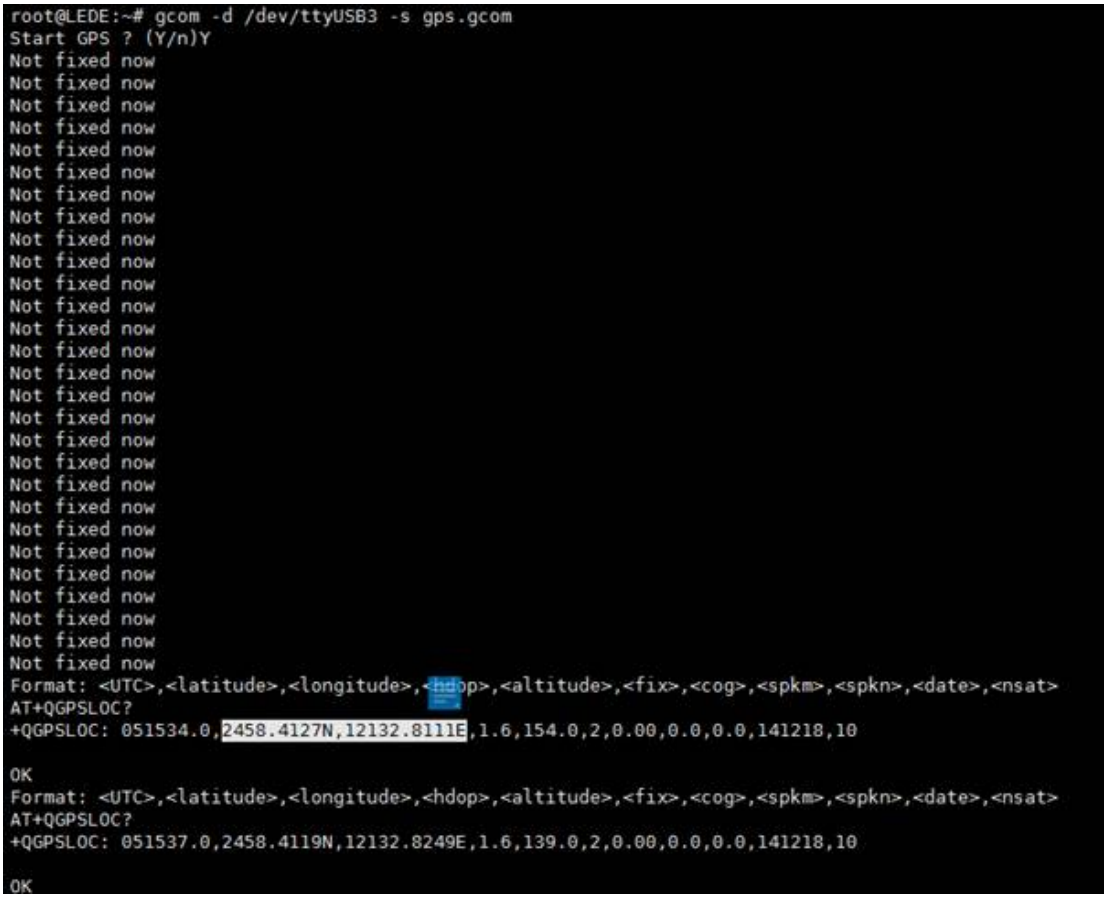
Below is the script to get the GPS information, user needs to save the script on a file (gps.gcom).

```
opengt
        set com 115200n81
```

```
	set senddelay 0.02
	set comecho off
	waitquiet 0.2 0.2
	flash 0.1

	print "Start GPS ? (Y/n)"
	input $a

	let $x=$left($a,1)
	#print $x
	if $x <> "Y" goto stopgps

:start
	send "AT+QGPS=1^m"
	get 1 "" $s
	#print $s

:getl
	send "AT+QGPSLOC=2^m"
	get 3 "" $s
	let $r=$mid($s,14,15)
	#print "|"$r"|"
	if $r = "+CME ERROR: 516" print "Not fixed now\n" goto getl
	print "Format: <UTC>,<latitude>,<longitude>,<hdop>,<altitude>,<fix>,<cog>,<spkm>,<spkn>,<date>,<nsat>\n"
	print $s
	goto getl

:stopgps
	print "Stop GPS \n"
	send "AT+QGPSEND^m"
	get 1 "" $s
	#print $s
```

Upload the “gps.gcom” script to the device, and run the command bellow:

root@LEDE:~# gcom -d /dev/ttyUSB3 -s gps.gcom

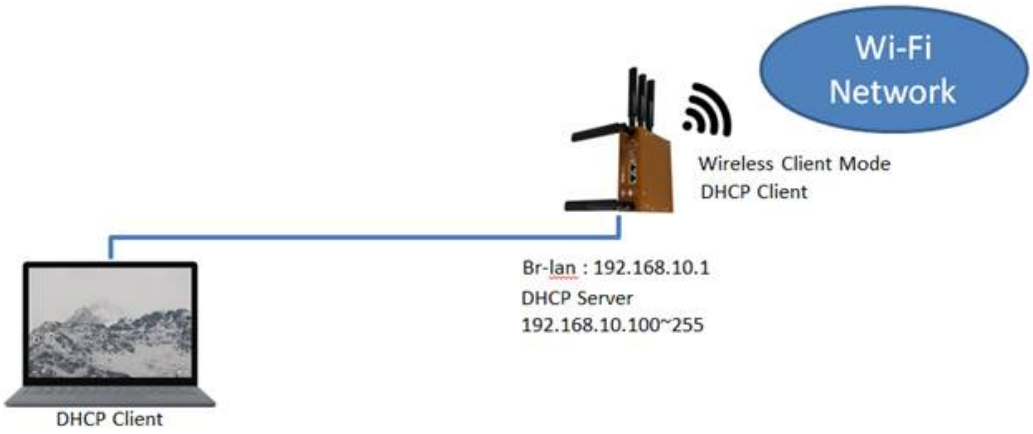


When the GPS is fixed, it will display in the following format

Format: <UTC>,<latitude>,<longitude>,<hdop>,<altitude>,<fix>,<cog>,<spkm>,<spkn>,<date>,<nsat>

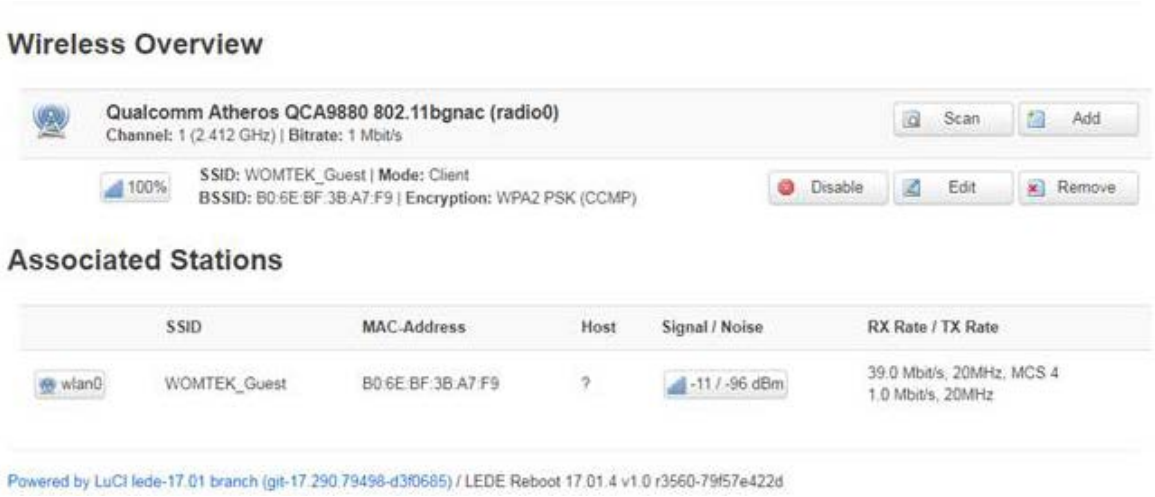
Press **ctrl + c** to stop the program.

4.3 WIRELESS



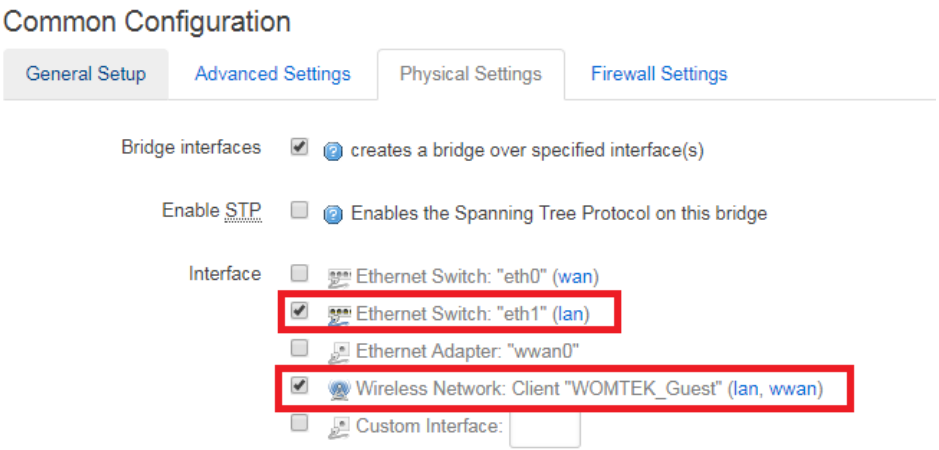
Below is the example for the configuration. (As Wireless Client)

Scan the wireless that user wants to connect.

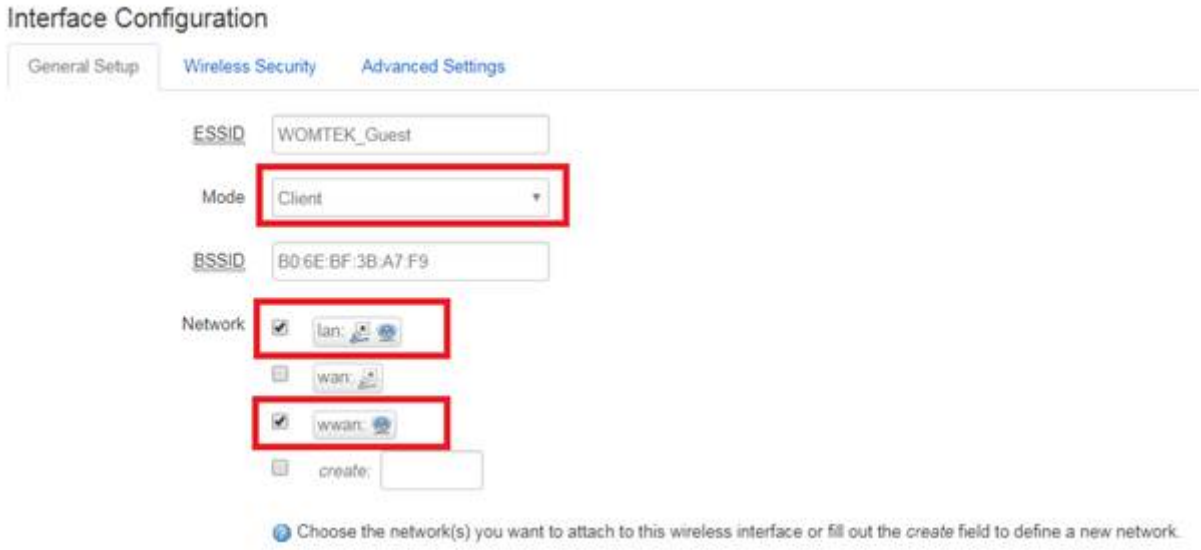


1. Go to Network -> Interface -> LAN -> Edit -> Physical Setting.

Check the bridge interface and choose the interface for bridge mode. (eth1 and the wireless network)

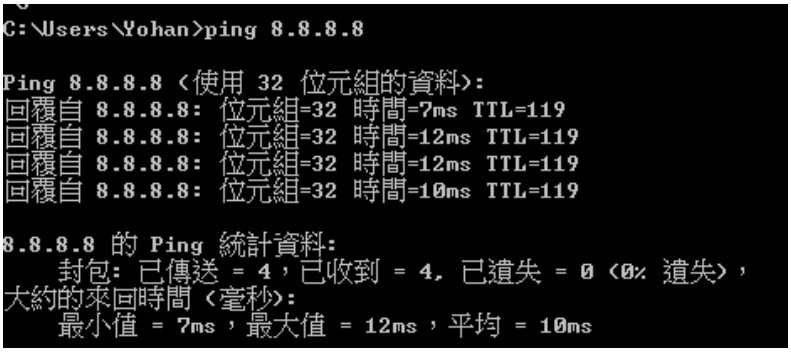


2. Go to Network -> Wireless -> Click edit on the connected wireless. Choose the network that want to attach the wireless connection.



3. Check the DHCP status on Network -> DHCP and DNS -> Active DHCP Lease to make sure if your notebook get the IP.

4. Try to ping to the internet from notebook. Ping 8.8.8.8



5. NETWORK FEATURES

In this chapter, we explain how to configure the Edge Computer various communication functions.

5.1 OpenVPN

OpenVPN is a full-featured SSL VPN which implements OSI layer 2 or 3 secure network extension using the industry standard SSL/TLS protocol, supports flexible client authentication methods based on certificates, smart cards, and/or username/password credentials, and allows user or group-specific access control policies using firewall rules applied to the VPN virtual interface.

This section covers several steps as below:

On each OpenVPN machine, you should generate a working directory, such as /**etc/openvpn**, where script files and key files reside. Once established, all operations will be performed in that directory.

5.1.1 Static-Key VPN

Run the following command in the /etc/openvpn directory to generate a static key:

```
root@LEDE:~# openvpn --genkey --secret static.key
```

Copy this static key to the clients /etc/openvpn directory using a secure channel like scp or sftp.

On the server, create a new /etc/openvpn/tun0.conf file and add the following:

```
dev tun0
ifconfig 10.9.8.1 10.9.8.2
secret /etc/openvpn/static.key
```

This is where 10.9.8.x is your VPN subnetwork; 10.9.8.1 is the IP of the server, and 10.9.8.2 the IP of the client.

On the client, copy /etc/openvpn/static.key from the server and create a new /etc/openvpn/tun0.conf file, and then add the following to the file:

```
remote myremote.mydomain
dev tun0
ifconfig 10.9.8.2 10.9.8.1
secret /etc/openvpn/static.key
```

Start OpenVPN using the following command:

```
root@LEDE:~# openvpn --config /etc/openvpn/tun0.conf --verb 6
```

NOTE:

When using an OpenVPN-related application, you need to create a firewall policy.

Original example: https://openvpn.net/community-resources/static-key-mini-howto/

openvpn Link Reference: https://openvpn.net/

5.2 ser2net: SERIAL TO NETWORK PROXY

ser2net provides a way for a user to connect from a network connection to a serial port. It could be like a bridge between the Ethernet cable and the serial cable.

The program comes up normally as a daemon, opens the TCP ports specified in the configuration file, and waits for connections. Once a connection occurs, the program attempts to set up the connection and open the serial port. If another user is already using the connection or serial port, the connection is refused with an error message.

Default config file	/etc/ser2net.conf

OPTIONS:

-c <config file> - use a config file besides /etc/ser2net.conf

-C <config line> - Handle a single configuration line. This may be specified multiple times for multiple lines. This is just like a line in the config file. This disables the default config file, you must specify a -c after the last -C to have it read a config file, too.

-p <controller port> - Start a controller session on the given TCP port

-P <file> - set location of pid file

-n - Don't detach from the controlling terminal

-d - Don't detach and send debug I/O to standard output

-l - Increate the debugging level

-u - Disable UUCP locking

-t <num threads> - Use the given number of threads, default 1

-b - unused (was Do CISCO IOS baud-rate negotiation, instead of RFC2217)

-v - print the program's version and exit

-s - specify a default signature for RFC2217 protocol

5.2.1 EXAMPLE

Setup a TCP server with following operation parameter:

Serial port : 1

TCP port : 3020

Baud rate : 9600

Data bits : 8

Parity : none

Stop bit : 1

Hardware flow control : none

State : raw state

timeout : never timeout

modem mode : none

Edit /etc/ser2net.conf add the following line:

```
3020:raw:0:/dev/ttyXRUSB1:9600 NONE 1STOPBIT 8DATABITS LOCAL -RTSCTS
```

Then run the ser2net program

```
root@LEDE:~# ser2net
```

5.3 IPSec

strongSwan is an OpenSource IPsec implementation, The focus of strongSwan is on

- simplicity of configuration
- strong encryption and authentication methods
- powerful IPsec policies supporting large and complex VPN networks
- modular design with great expandability

5.3.1 CONFIGURATION CONCEPT

If you have already worked with strongSwan you should know the different files you need to configure. It includes:

- /etc/strongswan.conf: Central configuration file
- /etc/ipsec.conf: Tunnel definitions
- /etc/ipsec.secrets: List of preshared keys
- /etc/ipsec.d: Folder for certificates

5.3.2 EXAMPLE SITE-TO-SITE

In this scenario two security gateways *moon* and *sun* will connect the two subnets moon-net and sun-net with each other through an IPSec VPN tunnel set up between the two gateways:

192.168.10.5/24 -- | 192.168.1.1 | === | 192.168.1.2 | -- 192.168.20.5/24

moon-net moon sun sun-net

Configuration on gateway sun**:**

/etc/ipsec.secrets:

```
# /etc/ipsec.secrets - strongSwan IPsec secrets file


%any : PSK "test"
```

/etc/ipsec.conf:

```
root@LEDE:/# cat /etc/ipsec.conf
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
        # strictcrlpolicy=yes
        # uniqueids = no

# Add connections here.

# Sample VPN connections

conn %default
        keylife=20m
        rekeymargin=3m
        keyingtries=1
        authby=secret
        keyexchange=ikev2
        mobike=no
        authby=secret
        ike=aes128-sha1-modp1024
        esp=aes128-sha1-modp1024

conn sample1
    leftsubnet=192.168.10.0/24
    left=192.168.1.1
    leftfirewall=yes
    right=192.168.1.2
    rightsubnet=192.168.20.0/24
    auto=start
```

Configuration on gateway moon:

/etc/ipsec.secrets:

```
root@LEDE:/# cat /etc/ipsec.secrets
```

```
# /etc/ipsec.secrets - strongSwan IPsec secrets file

%any : PSK "test"
```

/etc/ipsec.conf:

```
root@LEDE:/# cat /etc/ipsec.conf
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
        # strictcrlpolicy=yes
        # uniqueids = no

# Add connections here.

# Sample VPN connections

conn %default
        keylife=20m
        rekeymargin=3m
        keyingtries=1
        authby=secret
        keyexchange=ikev2
        mobike=no
        authby=secret
        ike=aes128-sha1-modp1024
        esp=aes128-sha1-modp1024

conn sample1
        leftsubnet=192.168.20.0/24
        left=192.168.1.2
        leftfirewall=yes
        right=192.168.1.1
        rightsubnet=192.168.10.0/24
        auto=start
```

Then execute

sun# ipsec start

Starting strongSwan 5.7.2 IPsec [starter]...

moon# ipsec start

Starting strongSwan 5.7.2 IPsec [starter]...

sun# ipsec status all

```
no files found matching '/etc/strongswan.d/*.conf'
Status of IKE charon daemon (strongSwan 5.5.3, Linux 4.4.92, mips):
   uptime: 33 minutes, since Oct 17 17:46:51 2017
   worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 0
   loaded plugins: charon aes des rc2 sha2 sha1 md5 random nonce x509 revocation constraints pubkey pkcs1 pgp
dnskey sshkey pem fips-prf gmp xcbc hmac attr kernel-netlink resolve socket-default connmark stroke updown
xauth-generic
Listening IP addresses:
   192.168.1.2
   192.168.20.1
Connections:
Security Associations (0 up, 0 connecting):
   none
```

moon# ipsec status all

```
no files found matching '/etc/strongswan.d/*.conf'
Status of IKE charon daemon (strongSwan 5.5.3, Linux 4.4.92, mips):
   uptime: 37 minutes, since Jan 09 12:43:05 2019
   worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 5
   loaded plugins: charon aes des rc2 sha2 sha1 md5 random nonce x509 revocation constraints pubkey pkcs1 pgp
dnskey sshkey pem fips-prf gmp xcbc hmac attr kernel-netlink resolve socket-default connmark stroke updown
xauth-generic
Listening IP addresses:
   192.168.1.1
   192.168.10.1
   fd99:4997:4c3::1
Connections:
       sample1:    192.168.1.1...192.168.1.1    IKEv2
       sample1:      local:    [192.168.1.1] uses pre-shared key authentication
```

```
        sample1:     remote: [192.168.1.1] uses pre-shared key authentication
        sample1:     child:    192.168.20.0/24 === 192.168.10.0/24 TUNNEL
Security Associations (2 up, 0 connecting):
        sample1[1]: ESTABLISHED 37 minutes ago, 192.168.1.1[192.168.1.1]...192.168.1.1[192.168.1.1]
        sample1[1]: IKEv2 SPIs: 43f1657886fc2de3_i* fec6474c2c978549_r, pre-shared key reauthentication in 2 hours
        sample1[1]: IKE proposal: AES_CBC_128/HMAC_SHA2_256_128/PRF_AES128_XCBC/MODP_3072
        sample1[2]: ESTABLISHED 37 minutes ago, 192.168.1.1[192.168.1.1]...192.168.1.1[192.168.1.1]
        sample1[2]: IKEv2 SPIs: 43f1657886fc2de3_i fec6474c2c978549_r*, pre-shared key reauthentication in 2 hours
        sample1[2]: IKE proposal: AES_CBC_128/HMAC_SHA2_256_128/PRF_AES128_XCBC/MODP_3072
```

Original example: https://www.strongswan.org/testing/testresults/ikev2/net2net-start/index.html

StrongSwan Link Reference: https://wiki.strongswan.org/projects/strongswan/wiki/UserDocumentation

6. ADVANCED FEATURE

6.1 wr-uart-ctl

PROGRAM NAME	WR-UART-CTL
Description	**Serial Ports** The serial ports support RS-232, RS-422, and RS-485 2-wire operation modes with flexible baudrate settings. The default operation mode is set to RS-232; use the **wr-uart-ctl** command to change the operation mode.

Usage:

Usage: wr-uart-ctl -p <#port_number> -m <#uart_mode>

Example:

Set serial port mode:

Port number: n = 0,1

port	Linux device name	Front panel name
0	/dev/ttyXRUSB0	Serial2
1	/dev/ttyXRUSB1	Serial1

uart mode: As in the following table

Uart mode	Operation Mode
None	Display current setting
0	RS-232
1	RS-485 2-wire
2	RS-422 / RS-485 4-wire

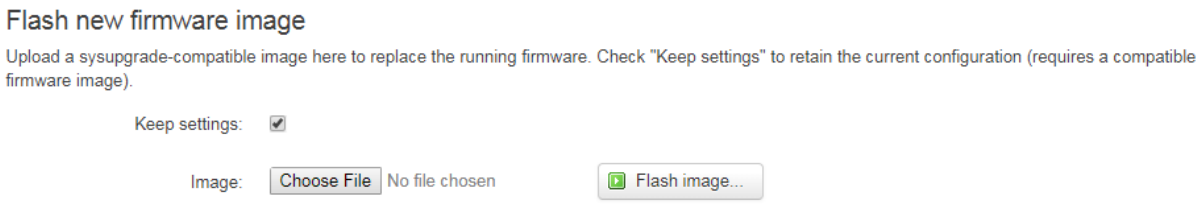
For example, to set Port 0 (com1) to RS-485 2-wire mode, use the following command:

root@LEDE:/# wr-uart-ctl -p 0 -m 1

6.2 FIRMWARE UPGRADE

From the web management, please follow the step below.

System -> Backup / Flash Firmware -> Flash New Firmware Image



Note! After upgrade the firmware, the SD card will need to be remounted manually again. Please follow the instructions below:

1. After the firmware upgrade.
2. Type 'df -h' check the SD card is not overlay on /
3. mount /dev/sda1 /mnt
4. rm -f /mnt/etc/.extroot-uuid
6. umount /dev/sda1
7. Run the extroot below command again

```
block detect > /etc/config/fstab; \
sed -i s/option$'\t'enabled$'\t'\'0\'/option$'\t'enabled$'\t'\'1\'/ /etc/config/fstab; \
sed -i s#/mnt/sda1#/overlay# /etc/config/fstab; \
cat /etc/config/fstab;
```

8. Reboot and check the SD card is mounted and overlay on /

7. PROGRAMMING GUIDE

7.1 TEST PROGRAM DEVELOPING – HELLO.C

In this section, we use the standard “Hello world” programming example to illustrate how to develop a program for the edge computer.

7.1.1 SYSTEM REQUIREMENT FOR YOUR BUILD SERVER

The Linux Operating System must be pre-installed in the PC before installing the OpenWRT Toolchain. First we need to make sure the dependencies are installed (for Debian 9/Ubuntu 16.04) on your build server:

```
sudo apt-get install build-essential

sudo apt-get install subversion g++ zlib1g-dev build-essential git python rsync man-db

sudo apt-get install libncurses5-dev gawk gettext unzip file libssl-dev wget zip time
```

7.1.2 COMPILE C SOURCE CODE

A. Download toolchain from the OpenWrt released supplementary files:

$

wget https://archive.openwrt.org/releases/17.01.4/targets/ar71xx/generic/lede-sdk-17.01.4-ar71xx-generic_gcc-5.4.0_musl-1.1.16.Linux-x86_64.tar.xz

B. Create toolchain dir

$ sudo mkdir /opt/toolchains/

C. Extract the toolchain to your working dir

$ sudo tar Jxvf lede-sdk-17.01.4-ar71xx-generic_gcc-5.4.0_musl-1.1.16.Linux-x86_64.tar.xz -C /opt/toolchains/

D. Add toolchain path to your $PATH environment

$ export

PATH=/opt/toolchains/lede-sdk-17.01.4-ar71xx-generic_gcc-5.4.0_musl-1.1.16.Linux-x86_64/staging_dir/toolchain-mips_24kc_gcc-5.4.0_musl-1.1.16/bin/:$PATH

E. Write a Hello World program

vim helloworld.c

```
#include <stdio.h>


int main(void)
{
	printf("\nHello, world!\n\n");
	return 0;
}
```

F. Compile helloworld

$ mips-openwrt-linux-gcc helloworld.c -o helloworld

G. Upload helloworld to device

Upload helloworld to the /tmp on device by scp program

H. Execute helloworld on the device

root@LEDE:/tmp# chmod +x helloworld

root@LEDE:/tmp# ./helloworld

OpenWrt Link Reference: https://openwrt.org/docs/guide-developer/helloworld/start

7.2 SYSTEM APIS

This section shows how to use some standard APIs on WoMaster's Edge computers.

7.2.1 WDT (WATCH DOG TIMER)

The WDT works like a watchdog function. You can enable it or disable it. When the WDT is enabled, but the application does not acknowledge it, the system will reboot. You can set the ack time from a minimum of 1 sec to a maximum of 1 day. The default timer is 60 seconds and the NO WAY OUT is enabled by default; there is no way to disable the watchdog once it has been started. For this reason, if the watchdog daemon crashes, the system will reboot after the timeout has passed.

Show the watchdog setting

root@LEDE:~# ubus call system watchdog

```
root@LEDE:~# ubus call system watchdog
{
        "status": "running",
        "timeout": 30,
        "frequency": 5,
        "magicclose": false
}
```

To stop the watchdog function:

root@LEDE:~# ubus call system watchdog '{ "stop": true }'

```
root@LEDE:~# ubus call system watchdog '{ "stop": true }'
{
        "status": "stopped",
        "timeout": 30,
        "frequency": 5,
        "magicclose": false
}
```

To start the watchdog function:

root@LEDE:~# ubus call system watchdog '{ "stop": false }'

```
root@LEDE:~# ubus call system watchdog '{ "stop": false }'
{
        "status": "running",
        "timeout": 30,
        "frequency": 5,
        "magicclose": false
}
```

Watchdog link reference: https://openwrt.org/docs/guide-developer/ubus/system

7.2.2 RTC (REAL TIME CLOCK)

Real Time Clock is a computer clock that keeps track of the current time. RTC can be used to complete time critical tasks. Using RTC can benefit from its lower power consumption and higher accuracy.

Below are the commands:

hwclock -r ... to "read" contents of RTC

```
root@LEDE:~# hwclock -r
2018-12-28 06:18:44.858654+0000
```

hwclock -s ... to "set" time/date from contents of RTC

hwclock -w ... to "write" current date into RTC

7.2.3 GPIO

GPIOs are represented by entries in the *sys* filesystem. You can check which GPIOs are available in the */sys/class/gpio* directory.

Function	Sys filesystem path	Default value	
Daughter Board SIM Select	/sys/class/gpio/gpio3/value	0 (SIM2)	Example Change from SIM2 to SIM1 (1.) cat /sys/class/gpio/gpio3/value (2.) echo 1 > /sys/class/gpio/gpio3/value
Enable Daughter Board Cellular Module PWR	/sys/class/gpio/gpio4/value	1 (power on)	Read Daughter Board PCIe Power (1.) cat /sys/class/gpio/gpio4/value 1 Turn Off Power (2.) echo 0 > /sys/class/gpio/gpio4/value
Enable Main Board Cellular Module PWR	/sys/class/gpio/gpio11/value	1 (power on)	Read Main Board PCIe Power (1.) cat /sys/class/gpio/gpio11/value 1 Turn off Power (2.) echo 0 > /sys/class/gpio/gpio11/value
Reset Button	/sys/class/gpio/gpio23/value	1 (not pressed) 0 (pressed)	Check the reset push button if is pressed cat /sys/class/gpio/gpio23/value

Important! The main board and daughter board PCIe power is only for turning on/off Cellular module. The Wi-Fi modules DO NOT support to turn on/off at run time.

7.2.4 LED

All LEDs are represented by entries in the *sys* filesystem. You can check which LEDs are available in the */sys/class/leds* directory.

The name of an entry typically consists of the name of the hardware providing the LED (the router model), or it's designation (usually the label on the case).

LED	COMMAND	SYS FILESYSTEM PATH
Power LED	Direct Link	NA
Serial1_LED	Web > System > LED Configuration	/sys/class/leds/s1
Serial2_LED	Web > System > LED Configuration	/sys/class/leds/s2
DO_RELAY / LED	Web > System > LED Configuration	/sys/class/leds/relay
SYSTEM_LED	Web > System > LED Configuration	/sys/class/leds/sys
Ra	Web > System > LED Configuration	/sys/class/leds/Ra
Rb	Web > System > LED Configuration	/sys/class/leds/Rb
Rc	Web > System > LED Configuration	/sys/class/leds/Rc
Rd	Web > System > LED Configuration	/sys/class/leds/Rd
Re	Web > System > LED Configuration	/sys/class/leds/Re
Rf	Web > System > LED Configuration	/sys/class/leds/Rf
Port 1 LED	Direct Link	NA
Port 2 LED	Direct Link	NA

The LED can be controlled by various events in the system, which is selected by the *trigger* option. Depending on the trigger, additional options must be specified.

First of all, you need to know which triggers are available for a led, to do that simply look at the *trigger* file of that led. Example:

```
root@LEDE:/# cat /sys/class/leds/Ra/trigger
[none] timer default-on netdev gpio heartbeat oneshot phy0rx phy0tx phy0assoc phy0radio phy0tpt
```

If we wanted to (temporarily) assign a **default-on** trigger to the led, we would write

```
root@LEDE:/# echo "default-on" > /sys/class/leds/Ra/trigger
```

You can confirm that you changed this by using cat again, you will see it has changed and the selected trigger is highlighted.

```
root@LEDE:/# cat /sys/class/leds/Ra/trigger
none timer [default-on] netdev gpio heartbeat oneshot phy0rx phy0tx phy0assoc phy0radio phy0tpt
```

Now, this change will be lost on reboot, if you want to make a permanent change, you need to add the trigger in uci configuration. If this is the first time, you don't have any configuration for it, you can add it and the trigger by editing the following example text (that sets Ra led as “default-on” and then copy-pasting it whole in the terminal window.

```
rule_name=$(uci add system led_ra)
uci batch <<EOF
set system.$rule_name=led
set system.$rule_name.name='Ra'
set system.$rule_name.sysfs='Ra'
set system.$rule_name.trigger='default-on'
EOF

uci commit
```

Now, if you want change the trigger assigned to Ra into “heartbeat” and you already have a block of options for it like this when you write **uci show system | grep “system.Ra”** (you can have more or less, it may also not have a trigger already assigned).

```
root@LEDE:/# uci set system.@led[0].trigger='heartbeat'
root@LEDE:/# uci commit
```

Types of led trigger can be found in the following link.

Link References: https://openwrt.org/docs/guide-user/base-system/led_configuration

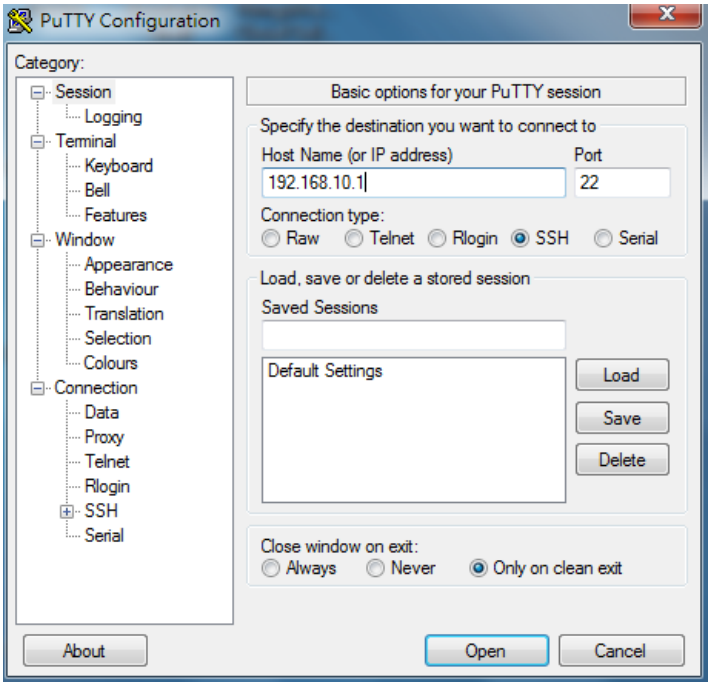
7.3 NODE-RED

WoMaster Edge Computer is equipped with Node-Red. Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.

7.3.1 OPEN THE NODE-RED

To execute the Node-Red please follow the step below:

The device supports an SSH Console. Open a putty or any software and set up an SSH console for the Edge Computer in a Windows environment.



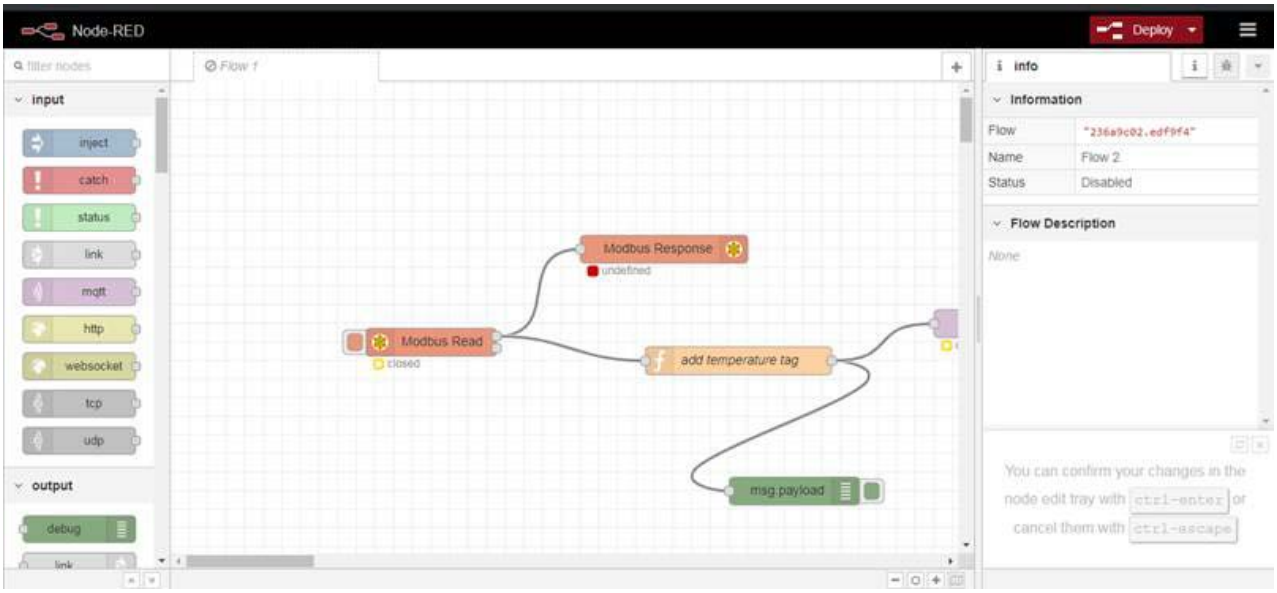
Then type **“node-red”** on the command line and press Enter. Wait till the establishing process is done.

```
root@LEDE:~# node-red
4 Jan 02:45:30 - [info]

Welcome to Node-RED
===================

4 Jan 02:45:30 - [info] Node-RED version: v0.19.4
4 Jan 02:45:30 - [info] Node.js  version: v8.12.0
4 Jan 02:45:30 - [info] Linux 4.4.92 mips BE
4 Jan 02:45:40 - [info] Loading palette nodes
4 Jan 02:45:45 - [warn] rpi-gpio : Raspberry Pi specific node set inactive
4 Jan 02:45:45 - [warn] rpi-gpio : Cannot find Pi RPi.GPIO python library
4 Jan 02:45:59 - [info] Settings file  : /root/.node-red/settings.js
4 Jan 02:45:59 - [info] Context store  : 'default' [module=memory]
4 Jan 02:45:59 - [info] User directory : /root/.node-red
4 Jan 02:45:59 - [warn] Projects disabled : editorTheme.projects.enabled=false
4 Jan 02:45:59 - [info] Flows file     : /root/.node-red/flows_LEDE.json
4 Jan 02:46:00 - [info] Server now running at http://127.0.0.1:1880/
4 Jan 02:46:00 - [warn]

---------------------------------------------------------------------
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
---------------------------------------------------------------------

4 Jan 02:46:00 - [info] Starting flows
4 Jan 02:46:00 - [info] Started flows
```

Open the node-red on your browser: <EC Series IP Address>:1880

For example: 192.168.1.1:1880

And you will directly see the node-red interface.



7.3.2 CREATE SIMPLE FLOW

1. Add an Inject node

The Inject node allows you to inject messages into a flow, either by clicking the button on the node, or setting a time interval between injects.

Drag one onto the workspace from the palette.

Open the sidebar (Ctrl-Space, or via the dropdown menu) and select the Info tab.

Select the newly added Inject node to see information about its properties and a description of what it does.

2. Add a Debug node

The Debug node causes any message to be displayed in the Debug sidebar. By default, it just displays the payload of the message, but it is possible to display the entire message object.

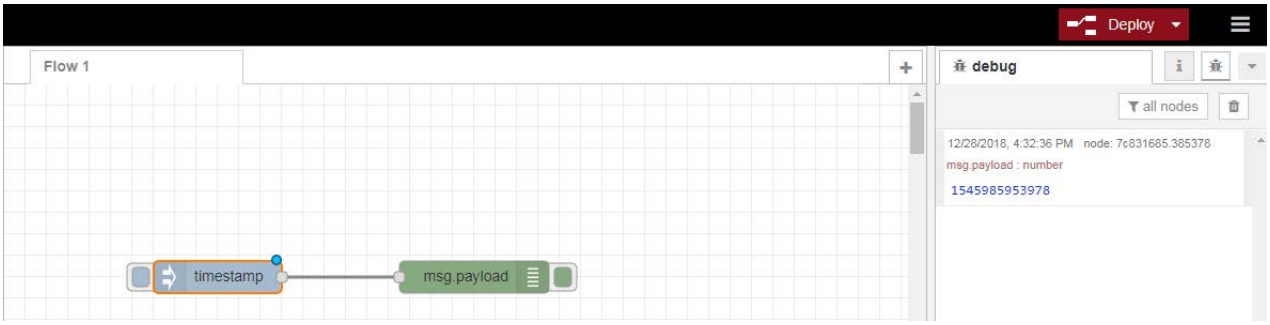
3. Wire the two together

Connect the Inject and Debug nodes together by dragging between the output ports of one to the input port of the other.

4. Deploy

At this point, the nodes only exist in the editor and must be deployed to the server.

Click the Deploy button. With the Debug sidebar tab selected, click the Inject button. You should see numbers appear in the sidebar. By default, the Inject node uses the number of milliseconds since January 1st, 1970 as its payload. Let's do something more useful with that.



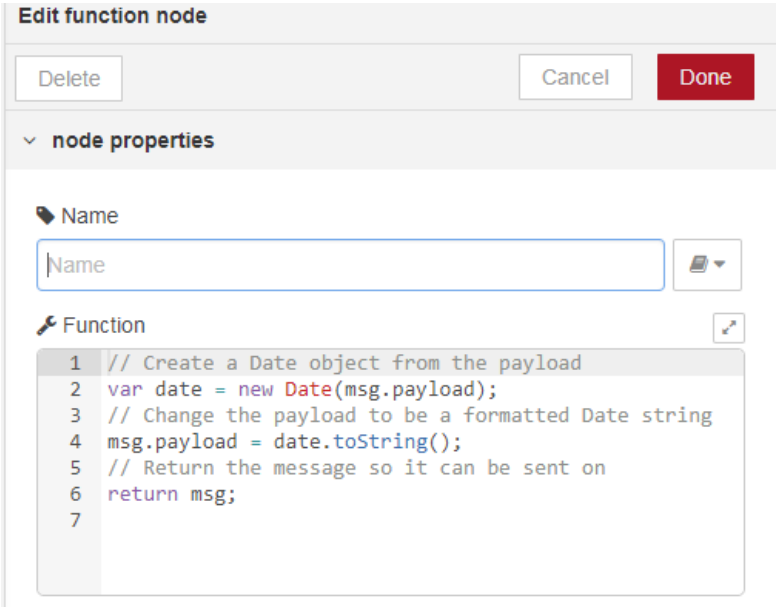
5. Add a Function node

The Function node allows you to pass each message though a JavaScript function.

Wire the Function node in between the Inject and Debug nodes. You may need to delete the existing wire (select it and hit delete on the keyboard).

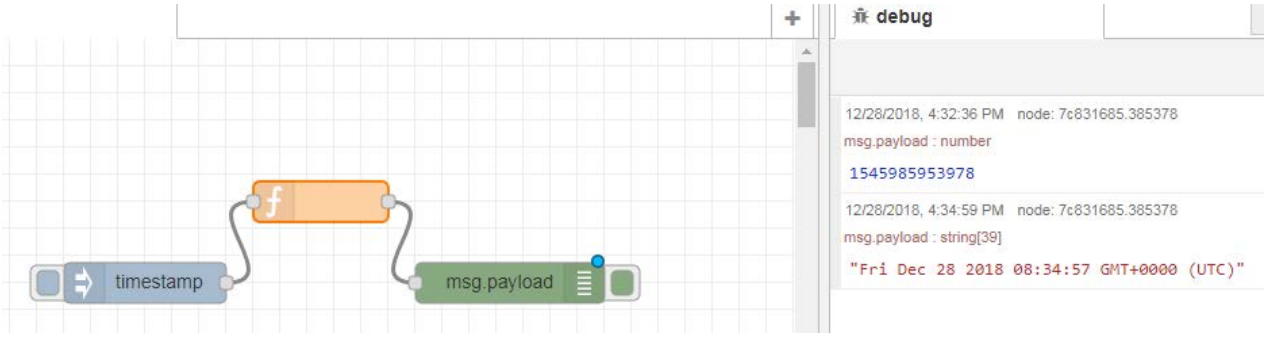
Double-click on the Function node to bring up the edit dialog. Copy the follow code into the function field:

```
// Create a Date object from the payload
var date = new Date(msg.payload);
// Change the payload to be a formatted Date string
msg.payload = date.toString();
// Return the message so it can be sent on
return msg;
```



Click Ok to close the edit dialog and then click the deploy button.

Now when you click the Inject button, the messages in the sidebar will be more readable time stamps.



SOURCE

The flow created in this example is represented by the following json. It can be imported straight into the editor by pasting the json into the Import dialog (Ctrl-I or via the dropdown menu).


```
[{"id":"c5ba81c5.8fc89","type":"tab","label":"Flow 1","disabled":false,"info":""},{"id":"99bdf2d8.0c5aa","type":"inject","z":"c5ba81c5.8fc89","name":"","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":200,"y":180,"wires":[["edd0d57.cfabb28"]]},{"id":"7c831685.385378","type":"debug","z":"c5ba81c5.8fc89","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","x":470,"y":180,"wires":[]},{"id":"edd0d57.cfabb28","type":"function","z":"c5ba81c5.8fc89","name":"","func":"// Create a Date object from the payload\nvar date = new Date(msg.payload);\n// Change the payload to be a formatted Date string\nmsg.payload = date.toString();\n// Return the message so it can be sent on\nreturn msg;\n","outputs":1,"noerr":0,"x":330,"y":120,"wires":[["7c831685.385378"]]}]
```


7.3.3 NODE-RED CONFIGURATION FILE

Use --userDir parameter to specify which folder to load the node red flows.

For example: The node red flows are in /root/.node-red/, and then it can start node-red with the following parameters:

node-red --userDir=/root/.node-red/

When it runs as a standalone application, these properties are read from the settings.js file.

The file can be downloaded from

https://raw.githubusercontent.com/node-red/node-red/master/settings.js

The configuration file can be used to change properties like uiHost, uiPort, ui authentication methods... For detail information please check the link below:

https://nodered.org/docs/configuration

Login to system and use wget to download the file to your node-red userDir.

wget --no-check-certificate https://raw.githubusercontent.com/node

-red/node-red/master/settings.js

Note: After change the setting in the file, it will need to restart node-red to take effects.

7.3.4 AUTO START NODE-REDS

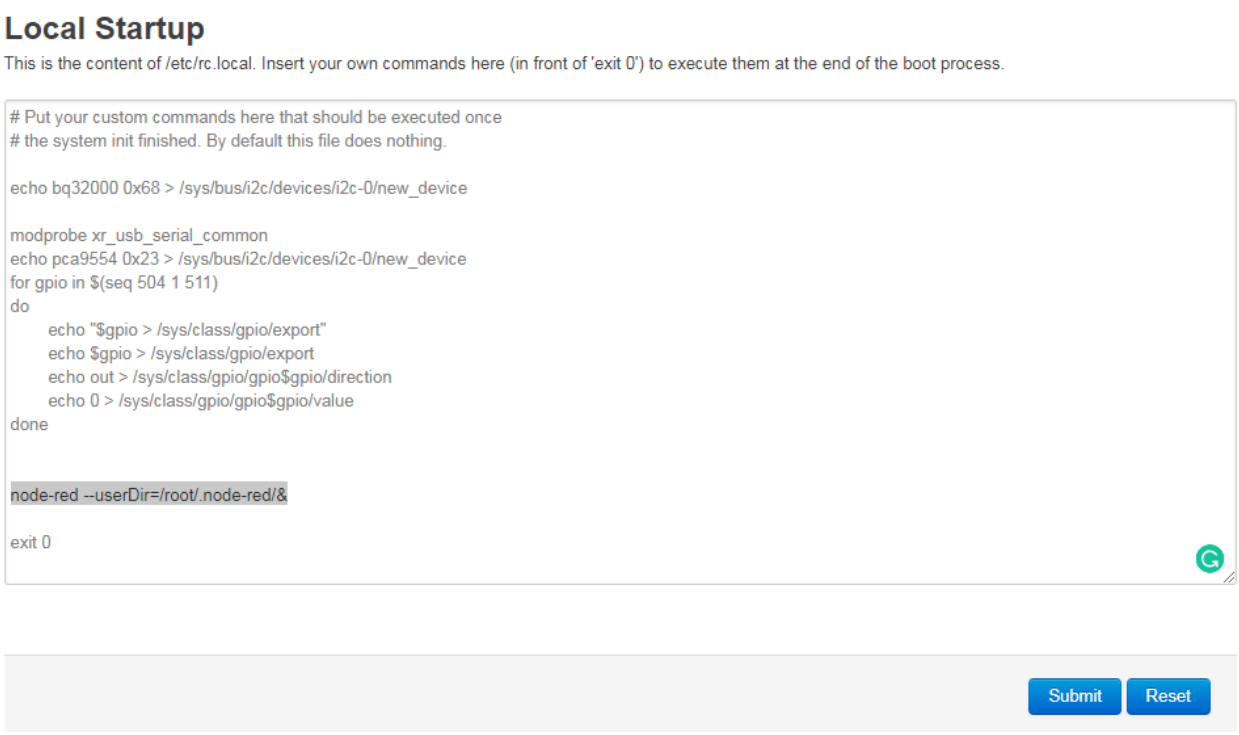
To auto start node-red after system startup, add one line in the /etc/rc.local

node-red --userDir=/root/.node-red/ &

```
root@LEDE:/# vi /etc/rc.local
root@LEDE:/# cat /etc/rc.local
# Put your custom commands here that should be executed once
# the system init finished. By default this file does nothing.

echo bq32000 0x68 > /sys/bus/i2c/devices/i2c-0/new_device

modprobe xr_usb_serial_common
echo pca9554 0x23 > /sys/bus/i2c/devices/i2c-0/new_device
for gpio in $(seq 504 1 511)
do
        echo "$gpio > /sys/class/gpio/export"
        echo $gpio > /sys/class/gpio/export
        echo out > /sys/class/gpio/gpio$gpio/direction
        echo 0 > /sys/class/gpio/gpio$gpio/value
done


node-red --userDir=/root/.node-red/&

exit 0
root@LEDE:/#
```

Auto Start the Node-Red from Web GUI

Web -> System->Startup



Add “node-red --userDir=/root/.node-red/ &”, it **MUST** have the "&" at the end of command in order to start the function in background. Click Submit to apply the command.

8. REFERENCE

1. ser2net(8) – Linux man page
2. iptalbes(8) – Linux man Page
3. Netfilter IPTables Mini Howto
4. Factory Reset

8.1 BUSYBOX COMMAND

```
root@LEDE:~# ash


BusyBox v1.25.1 () built-in shell (ash)
```

busybox(V1.25.1): Linux command collection

File Manager	
cp	copy file
ls	list file
ln	make symbolic link file
mount	mount and check file system
rm	delete file
chmod	change file owner & group & user
chown	change file owner
chgrp	change file group
sync	Sync file system, let system file buffer be saved to hardware
mv	move file
pwd	display now file directly
df	list now file system space
mkdir	make new directory
rmdir	delete directory

Editor	
vi	text editor
cat	dump file context
zcat	compress or expand files
grep	search string on file
cut	get string on file
find	find file where are there
less	View a file or list of files.

test	test file exist or not
sleep	sleep(seconds)
echo	Echo string
awk	Pattern scanning and processing language.
sed	perform text transformations on a file or input from a pipeline.
xargs	execute a specified command on every item from standard input.

Archival Utilities	
bzip2/	Compress/Uncompress bzip FILE
gzip/gunzip	Compress/Uncompress FILE with maximum compression.
tar	Create, extract, or list files from a tar file

System logging	
logger	Utility to send arbitrary text messages to the system log
Network	
ping	ping to test network
arping	Ping host by ARP packets
nslookup	Tool to query Internet name servers
traceroute	Utility to trace the route of IP packets
wget	Utility for non-interactive download of files from HTTP, HTTPS, and FTP servers.
udhcpc	DHCP client
route	routing table manager
netstat	display network status
Ifconfig	set ip address and configure network interfaces
Logread	Display all of the system log

Others	
dmesg	dump kernel log message
zcat	dump .gz file context
mknod	make device node
free	display system memory usage
date	print or set the system date and time
env	run a program in a modified environment
clear	clear the terminal screen
reboot	reboot / power off/on the server
halt	halt the server
du	estimate file space usage

kill/killall	Send specified signal to the specified process or process group

For complete command usage and explanation, please refer to following website: http://www.busybox.net/downloads/BusyBox.html

Version	Modification	By	Date
V1.0	Release	Yohan	17/01/2019
V1.1	- Update the Cellular section - Update the Node-red part (7.3.3 & 7.3.4)	Yohan	25/02/2019