Atop Technologies, Inc.

SE59XX Family
Node-RED user guide

User Manual

V1.2

July 20th, 2018

This PDF Document contains internal hyperlinks for ease of navigation.
For example, click on any item listed in the **Table of Contents** to go to that page.

Published by:

Atop Technologies, Inc.

2F, No. 146, Sec. 1, Tung-Hsing Rd,
30261 Chupei City,
Hsinchu County
Taiwan, R.O.C.

Tel: +886-3-550-8137
Fax: +886-3-550-8131
sales@atop.com.tw
www.atoponline.com
www.atop.com.tw

Important Announcement

Disclaimer

We have checked the contents of this manual for agreement with the hardware and the software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual is reviewed regularly and any necessary corrections will be included in subsequent editions.
Suggestions for improvement are welcome.
All other product's names referenced herein are registered trademarks of their respective companies.

Documentation Control

Author:	Carlos Hsu, Matteo Tabarelli
Revision:	1.2
Revision History:	Draft
Creation Date:	20 July 2018
Last Revision Date:	7 August 2018
Product Reference:	SE5901 (Node-RED), SE5901B (Node-RED), SE5904D (Node-RED), SE5908 (Node-RED), SE5916 (Node-RED), SE5900A (Node-RED), SE5908A (Node-RED), SE5916A (Node-RED)
Document Status:	Released

Table of Contents

Table of Figures

List of Tables

1 Preface

1.1 Purpose of the Manual

This manual supports you in understanding how to use Node-RED add-on on ATOP's SE59XX Series and should be a reference guide for application development on this platform.

1.2 Notice

(a) Node-Red is an open-source freeware for IoT developments, designed in cooperation with IBM. Node-RED is a is a browser-based logical flow building-block editor, and it embeds a web-based dashboard

(b) Node-Red requires a large amount of storage memory that is not available on SE59XX embedded computer. In order to run it, it should be plugged in to the device as a pre-loaded SD card or USB storage pen drive.

(c) If you purchased the pre-loaded SD card or USB pen drive from ATOP, there is no configuration necessary: it's enough for you to plug in the SD card or the USB pen drive in SE59XX hardware before powering on. Otherwise, if you are downloading the library from ATOP's website and loading it in the USB pen drive or SD card yourself, you need create data & swap partitions in SD card or USB storage first.

(d) All the details that require users to input or modify are highlighted in this document.

1.3 Who Should Use This User Manual

This manual is intended to be used by qualified programmers, network personnel, support technicians or from hands-on people that are familiar with Javascript. Familiarity with network operations and Javascript Language programming may be necessary. For any related problems, please contact your local distributor. If they are unable to assist you, please redirect your inquiries to www.atop.com.tw or www.atoponline.com.

1.4 Supported Platform

This manual is designed for the SE5901, SE5901B, SE5904D, SE5908, SE5916, SE5900A, SE5908A, and SE5916A Industrial Edge computers and for these models only.

1.5 Warranty Period

ATOP provides a **5-year limited warranty** for SE59XX Series.

2 Introduction to Atop SDK with Node-RED

2.1 ATOP Node-RED

Thank you for Purchasing ATOP's Embedded Edge Computer with Node-RED. The device, if purchased as-is, is an industrial grade, ruggedized hardware designed to be working "out of the box".
This makes us different from commercially available solutions that are often based on Raspberry PI, where no Industrial Hardware is provided and the whole configuration has to be carried out from the Linux command shell. We understand that who is interested in Node-RED, doesn't necessarily have a Linux or Programming background, so we decided to enable the user to carry out basic configuration through a simple interface.

2.1.1 Working "out of the box"

The device can be accessed from Web User Interface from the factory default IP address (10.0.50.100). Once inputting the access credentials, you will access ATOP Device's configuration dashboard (see below Figure 2.1) that will guide you through the whole basic set-up. More details about basic configuration are provided in **Section 5 below**

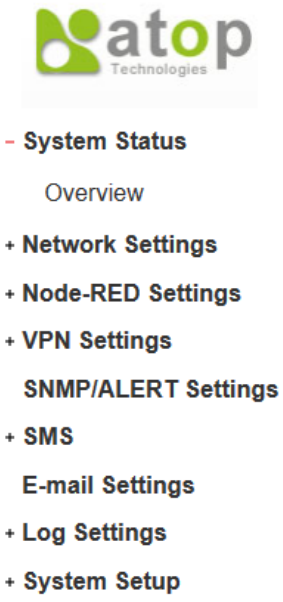


Figure 2.1 ATOP Dashboard on Web-UI

Node-RED application, thoroughly tested, is going to start up on background after power-on. The application files are stored on an internal SD or MicroSD-card (pre-installed) in devices with internal SD-card slot or on an USB stick that is provided inside of the box.

2.1.2 Running Node-RED after performing basic configuration

After performing basic configuration through the web UI, simply key-in the device's IP-address & ":1880" (e.g. http://10.0.50.100:1880) and you will be able to directly access Node-RED flow editor and Node-RED Dashboard.

2.2 Overview of SE59XX – Node-RED product line architecture

ATOP's SE59XX Embedded computers are industrial grade, wide temperature embedded computers running Linux. All devices are powered by a powerful 800MHz or 1000MHz ARM Cortex A8 Texas Instruments Sitara AM3354 or AM3352 CPU.The Embedded Linux operating System, properly customized to better fit inside ATOP's rugged hardware is already running on the device and is the backbone on which the Node-RED application is running.

Figure 2.2 shows the whole architecture of SE59XX SDK. The device can also be used as a C-programmable embedded computer allowing you to pre-compile binary applications, uploading them to the device and running them from Node-RED environment, but this is not the scope of this user manual.

For more information on how to create applications, compile and upload them to the device itself, please refer to SE59XX SDK user manual. Three types of Applications are provided in user's layer:

1) ATOP applications: providing multiple sample SDK programs to hardware devices
2) ATOP utility: providing firmware upgrade, network settings and storage mounting tools
3) Third-party : providing 3rd parties software required such as Node-RED /SNMP / Apache / SQLite

In Kernel Layer, Linux 4.5 is customized to provide complete networking protocols.
In Driver Layer, device drivers for all Industrial communication interfaces are provided.
In Hardware Layer, Customized ARM Cortex-A8 platform and Atop FPGA management core are provided.

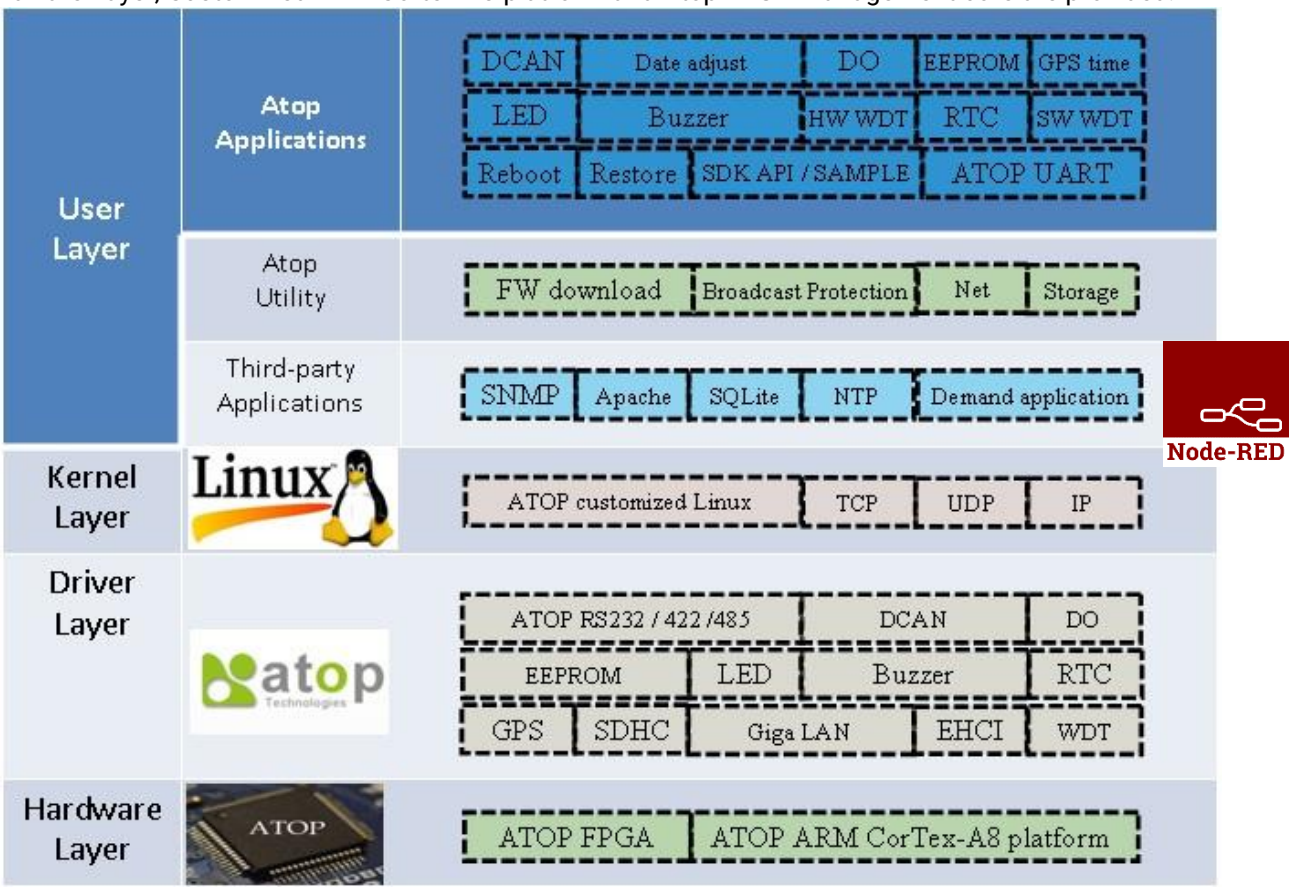
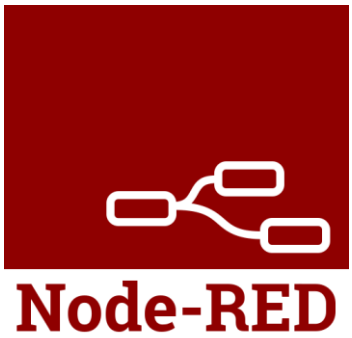


Figure 2.2 Architecture of SE59XX SDK

2.3 Node-RED



Node-RED is a freeware, open source building block programming tool, developed by IBM. It requires a large amount of storage memory that is not available on SE59XX embedded computer. In order to run it, it should be plugged in to the device as a pre-loaded SD card or USB storage pen drive. Once the device has been turned on, please move directly to Section 5 below.

If you purchased the standard product from ATOP or pre-loaded SD card or USB pen drive from, there is no Node-RED configuration necessary:

- If the device is purchased as Node-RED part number, just power it on
- If you purchased the pre-loaded Node-RED SD/microSD card or USB stick from ATOP, just install it and turn the power on.

Otherwise, if you are willing to use Node-RED on a standard SDK product, please download the library from ATOP's website and loading it in the USB pen drive or SD card yourself, you need to create data & swap partitions in SD card or USB storage first. Please see chapter 4.4 below for setting up partitions and installing Node-RED properly.

3 Hardware Specifications

3.1 Packing List

Inside the purchased package, you will find the following items:

Table 3.1 Packing List

Item	Quantity	Description
SE59XX	1	Industrial Embedded Computer
Node-RED	1	On SD-card capable devices: pre-installed SD-card with Node-RED On Micro-SD-card capable devices: pre-installed mSD card with Node-RED On other devices: 8GB USB stick, with Node-RED available in the box
Mounting Kit	1	On SE5908 / SE5916 / SE5908A / SE5916A Rack Mounting Type-L angles)x 2(Screws)x 6(On SE5901 / SE5904D / SE5901B - DIN Rail Kit
Terminal Block	See description	Power Supply/ Relay output: TB3 x 1: 3-pin 5.08mm lockable Terminal Block (SE5901, SE5901B) TB3 x 2: 3-pin 5.08mm lockable Terminal Block (SE5908-DC,SE5916-DC) TB7 x1: 7-pin 5.08mm lockable Terminal Block (SE5904D only) Serial ports: Terminal block is included only on TB model TB5 x 1: 5-pin 5.08mm lockable Terminal Block (SE5901) TB5 x 4: 5-pin 5.08mm lockable Terminal Block (SE5904D) TB5 x 8: 5-pin 5.08mm lockable Terminal Block (SE5908A) TB5 x 16: 5-pin 5.08mm lockable Terminal Block (SE5916A)
Documentation	1	Hardware Installation Guide)Warranty card is included(
Mounting Kit	1	DIN-Rail Kit (Already mounted on the device)

Note: Please notify your sales representative if any of the above items is missing or damaged in any form upon delivery. If your sales representative is unable to satisfy your enquiries, please contact us directly.

3.2 Optional Accessories

The following table lists optional accessories for SE59XX SDK series.

Table 3.2 Optional Accessories

Item	Description
UN315-1212(US-LDC)	Y-Type (5.08mm) power adapter, 100-240VAC input, 1.25A @ 12VDC output, US plug
UNE315-1212(EU-LDC)	Y-Type (5.08mm) power adapter, 100-240VAC input, 1.25A @ 12VDC output, EU plug
ADP-DB9(F)-TB5	Female DB9 to Female 3.81 TB5 Converter
CBL-RJ45(8P)-DB9(F)	8-pin RJ45-DB9 debug cable, 90cm
GDC-120	120mm copper woven grounding cable
LM28-C3S-TI-N	SFP Transceiver, 1250Mbps, 850nmVCSEL, Multi-mode, 550m, 3.3V, -20~85°C
LM38-C3S-TI-N	SFP Transceiver, 1250Mbps, 1310nmFP, Multi-mode, 2km, 3.3V, -40~85°C
LS38-C3S-TI-N	SFP Transceiver, 1250Mbps, 1310nmFP, Single-mode, 10km, 3.3V, -40~85°C
LS38-C3L-TI-N	SFP Transceiver, 1250Mbps, 1310nmDFB, Single-mode, 30km, 3.3V, -40~85°C
WMK-450-Black	Black Aluminum Wall Mount Kit (DIN-rail items only)

3.3 Hardware specifications

Table 3.3 Hardware features

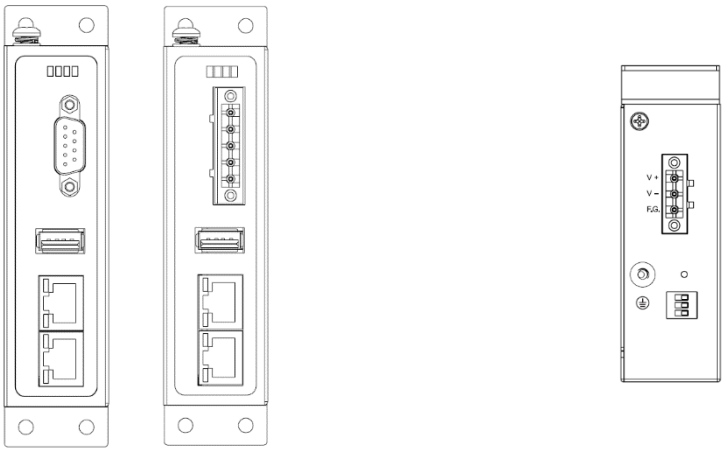
System	
CPU	32-bit ARM Based TI CPU AM3354 800MHz (except SE5908A/SE5916A use AM3352 1GHz)
Flash Memory	64MB
RAM	SE5901 DDR2 256MB (NodeRED version only) SE5901B DDR2 256MB SE5904D DDR3 256MB SE5900A/08A/16A/MB5908/16 DDR3 256MB
EEPROM	8 KB
Reset	Built-in Recessed Key (Restore to Factory Defaults)
Watchdog	Hardware built-in
Network	
Ethernet Interface	IEEE 802.3 10BaseT IEEE 802.3u 100BaseT(X) IEEE 802.3ac 1000BaseT(X) – SFP version of SE5904D only IEEE 802.3af (PoE PD) –selected SE5901 and SE5904D versions can be powered through PoE Connection: SFP or RJ45
Serial	
Serial Interface	RS-232/RS-422/RS-485 Software Selectable (Default: RS-232) • The first port available on SE5901B is RS-232/RS-485 • The second port available on SE5901B-IO-X is only RS-232 • The isolation version (-SiS) on SE5908/SE5916/SE5908A/ SE5916A supports only RS-422/ RS-485
Serial Connector	Connector Type • SE5916 -16 Serial Ports (RJ45) • SE5908 - 8 Serial Ports (RJ45) • SE5916A – 16 Serial Ports (TB-5 or DB-9) • SE5908A – 8 Serial Ports (TB-5 or DB-9) • SE5904 – 4 Serial Ports (TB-5 or DB-9) • SE5901 – 1 Serial Port (TB-5 or DB-9) • SE5901B – 1 Serial Port (TB-14 or DB-9) – includes I/O
Protection	SE5901/SE5901B no isolation SE5904D/ SE5908A/16A (optional 3V) SE5908/16 (optional 2.5kV)
Serial Port Communication	Baud-rate: 1200 bps ~ 921600 bps Parity: None, Even, Odd, Mark, or Space Data Bits: 5, 6, 7, 8 Stop Bits: 1, 2 Software Selectable Flow Control: RTS/CTS (RS-232 only), XON/XOFF, None
LED Indicator	
LED indication	Power x 2 (SE5901- SE5901B – SE5908 – SE5916 x 1) RUN x 1 ALARM x 1 LAN: • x 2 (all versions except SE5908A and SE5916A) • x 6 (SE5908A and SE5916A only)

	COM port: • x 16 (SE5916 and SE5916A); • x 8 (SE5908 and SE5908A); • x 4 (SE5904D); x 1 (SE5901 and SE5901B)	
Power Requirement & EMC		
Input	SE5908/ SE5916 : • Single 100~240 VAC (EU/US versions) • Single 24~48 VDC (DC version) SE5908A/ SE5916A • Redundant 100~240 VAC or 100~370 VDC (TB)– HV vers. • Redundant 24~48 VDC- DC version SE5901/SE5901B : Single 9~48 VDC SE5904D : Redundant 9~48 VDC	
Consumption	Max.17.5 W (SE5908 /SE5916) Max. 6W (SE5901) Max. 7.8W(SE5904D) Max. 17.5W(SE5908A/SE5916A) Max. 7.2W(SE5901B)	
EMI/EMC	FCC Part 15, Subpart B, Class A EN 55032, Class B, EN 61000-6-2, Class B EN 61000-3-2, EN 61000-3-3 EN 55024, EN 61000-6-4 IEC 61850-3 / IEEE 1613 (SE5908A and SE5916A only)	
Mechanical		
Dimensions (W x H x D, mm)	SE5901: 32 mm x 110 mm x 90 mm (1.26 x 4.33 x 3.54 in) SE5901B: 32 mm x 122mm x 92 mm (1.26 x 4.8 x 3.62 in) SE5904D: 55 mm x 145 mm x 113mm (2.17 x 5.17 x 4.45 in) SE5908: 436 mm x 43.5 mm x 200 mm (17.17 x 1.71 x 7.87 in) SE5916: 436 mm x 43.5 mm x 200 mm (17.17 x 1.71 x 7.87 in) SE5908A: 440.6mm x 44 mm x 309 mm (17.35 x 1.73 x 12.17 in) SE5916A: 440.6mm x 44 mm x 309 mm (17.35 x 1.73 x 12.17 in)	
Enclosure	IP30 protection, metal housing	
Environmental		
Temperature	Operations	-40°C ~ 85°C (-40°F ~ 185°F) (except SE5901B -40°C ~ 70°C and SE5908/SE5916 -20°C ~ 70°C)
	Storage	-40°C ~ 85°C (-40°F ~ 185°F)
Relative Humidity	5% ~ 95%, 55°C Non-condensing	

3.4 External Device's Overview

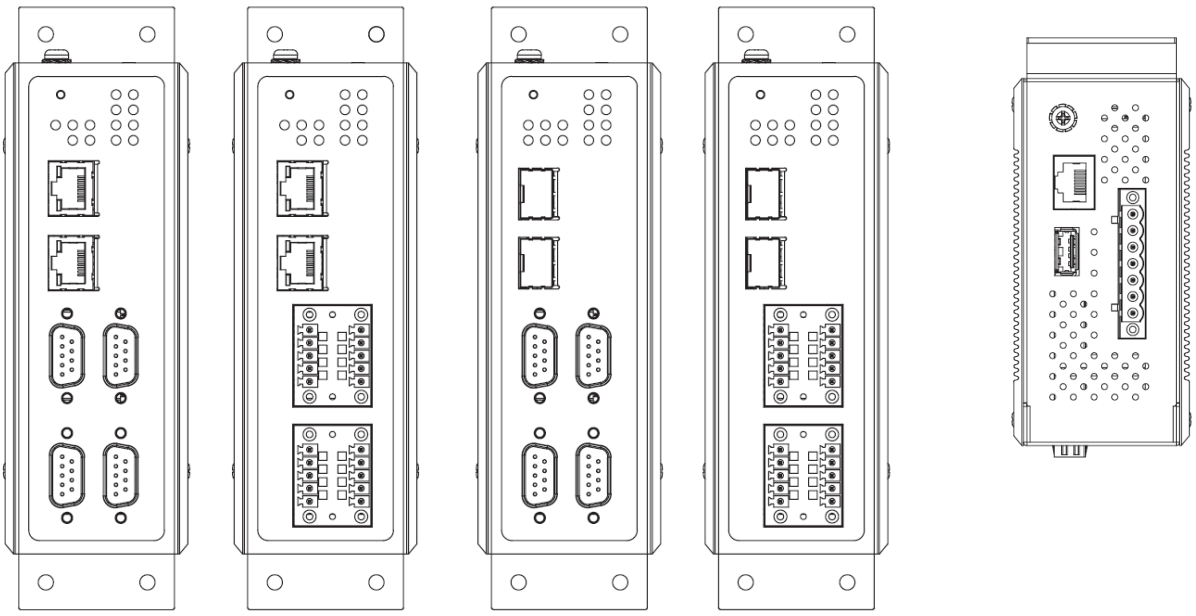
The following figures show particular SE59XX series device's front and rear panels.

SE5901

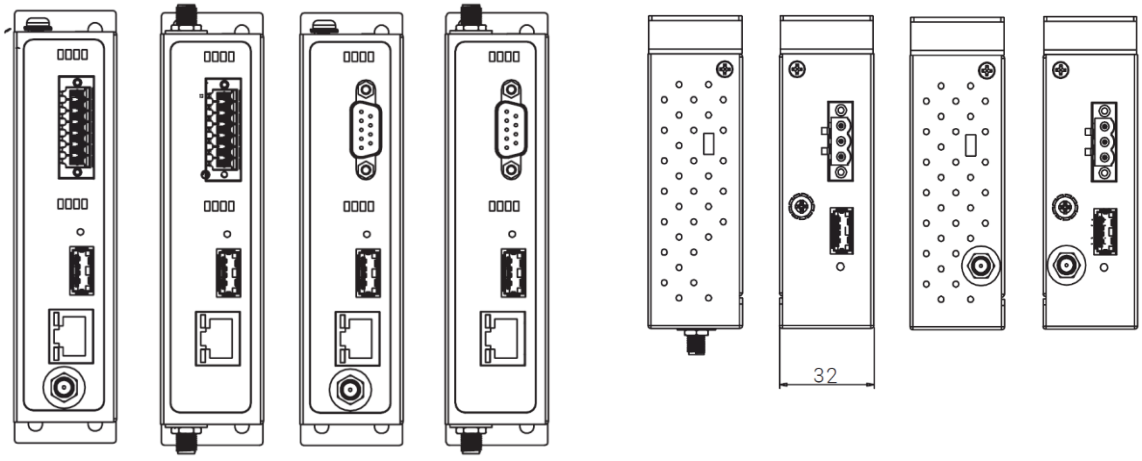


SE5904D

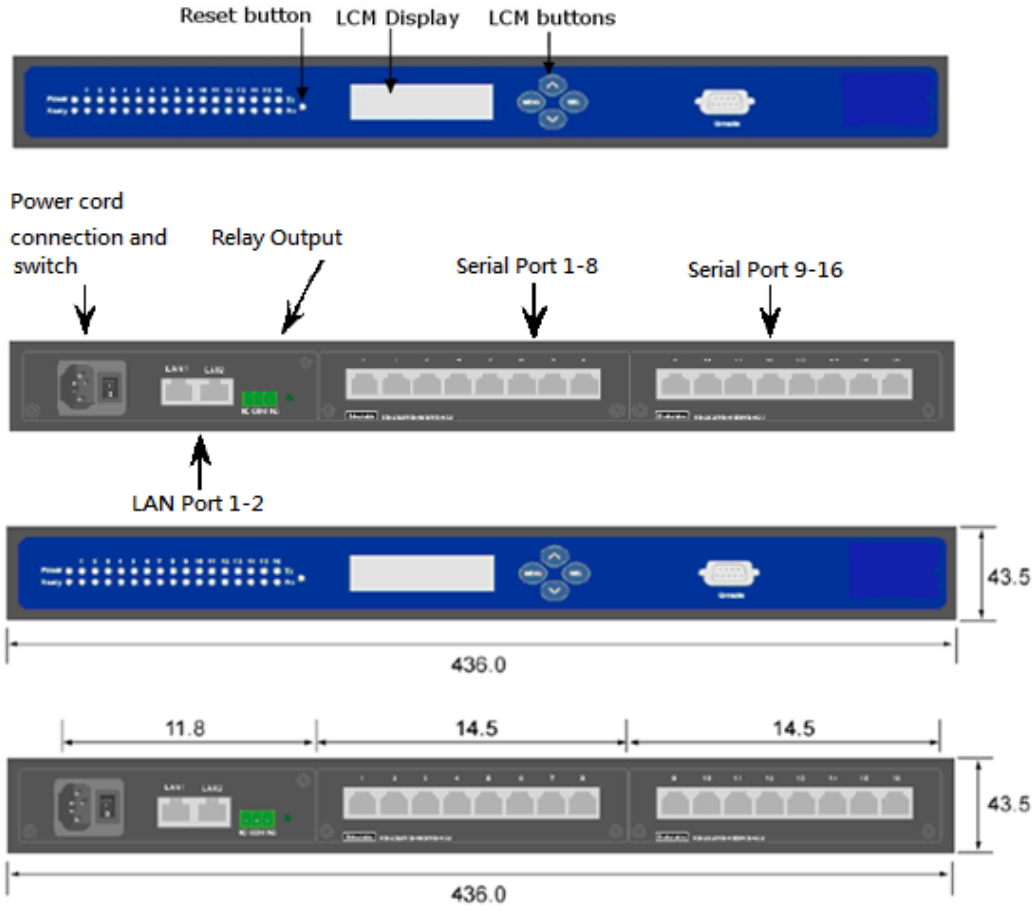
RJ45 DB9 Type RJ45 TB5 Type SFP DB9 Type SFP TB5 Type



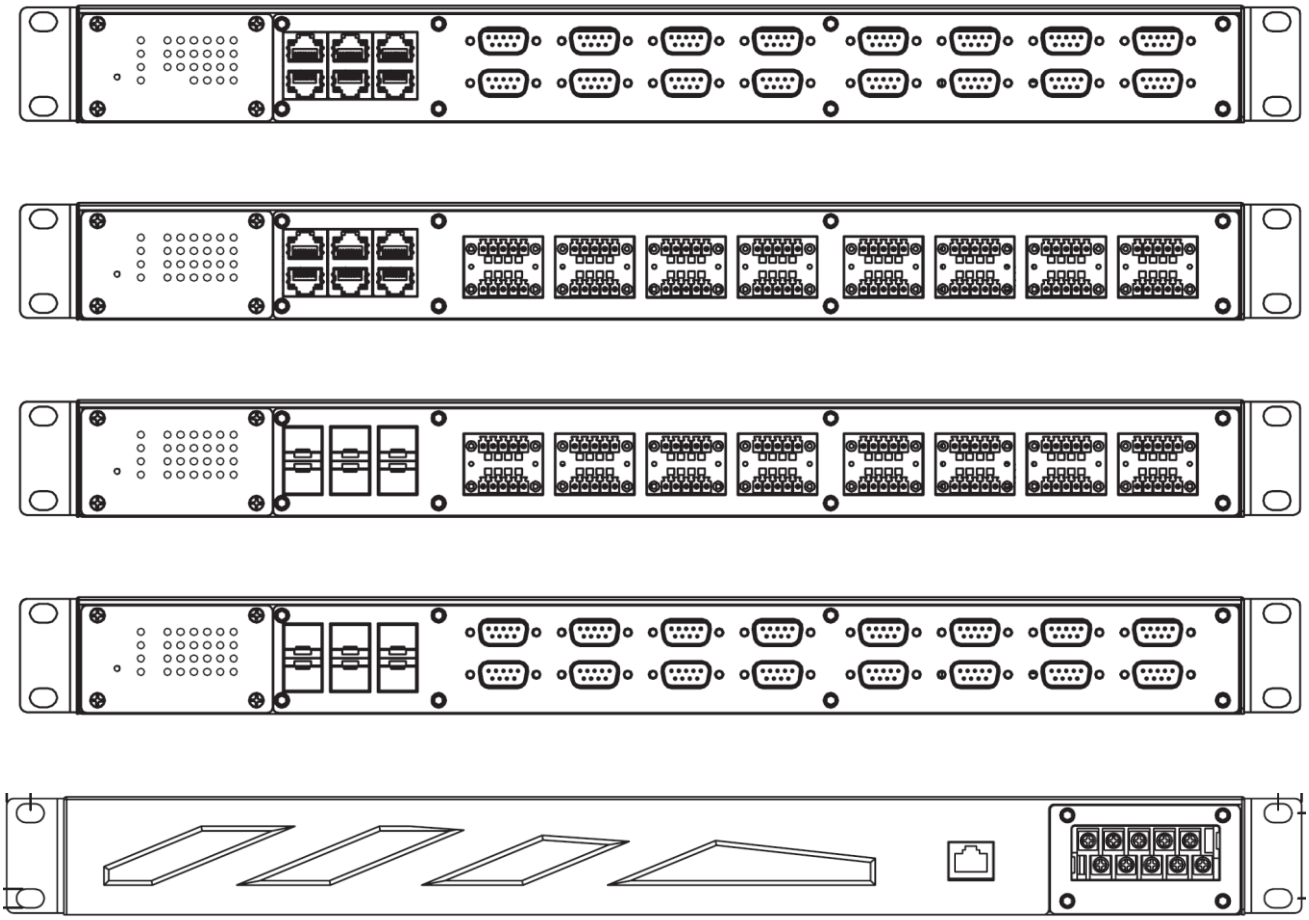
SE5901B



SE5908/16



SE5908A/16A



3.5 Serial Pin Assignments

3.5.1 SE5901 Pin Assignments for Serial Interfaces

DB9 to RS-232/RS-422/RS-485 connectors

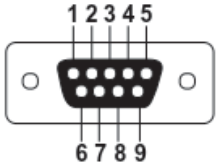


Figure 3.1 DB9 Pin Number

Table 3.4 SE5901 Pin Assignment for DB9 to RS-232/RS-422/RS-485 Connector

Pin#	RS-232 Full Duplex	RS-422/4-Wire RS-485 Full Duplex	2-Wire RS-485 Half Duplex
1	DCD	N/A	N/A
2	RxD	TXD+	N/A
3	TxD	RXD+	Data+
4	DTR	N/A	N/A
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	DSR	N/A	N/A
7	RTS	RXD-	Data-
8	CTS	TXD-	N/A
9	RI	N/A	N/A

1 x 5-pin (Male Terminal Block) for RS-232/RS-422/RS485 Connector

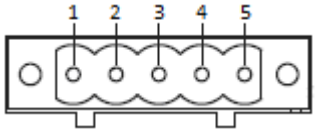


Figure 3.2 TB5 Pin Number

Table 3.5 SE5901 Pin Assignment for TB5 to RS-232/RS-422/RS-485 Connector

Pin#	RS-232 Full Duplex	RS-422/4-Wire RS-485 Full Duplex	2-Wire RS-485 Half Duplex
1	RxD	T+	NC
2	CTS	T-	NC
3	TxD	R+	Data+
4	RTS	R-	Data-
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)

3.5.2 SE5904D Pin Assignments

DB9 to RS-232/RS-485/RS-422 connectors

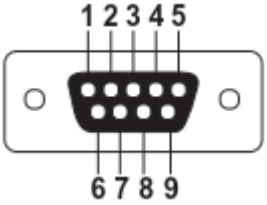


Figure 3.3 DB9 Pin Number

Table 3.6 SE5904D Pin Assignment for DB9 to RS-232/RS422/RS-485 Connectors

Pin#	RS-232 Full Duplex	RS-422 4-Wire RS-485	2-W RS-485
1	DCD	N/A	N/A
2	RxD	TxD+	Data+
3	TxD	RxD+	N/A
4	DTR	N/A	N/A
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	DSR	N/A	N/A
7	RTS	RxD-	N/A
8	CTS	TxD-	Data-
9	RI	N/A	N/A

5-Pin Terminal Block to RS-232/RS-485/RS-422 connectors

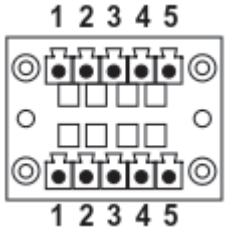


Figure 3.4 Terminal Block (TB-5) Pin Number

Table 3.7 SE5904D Pin Assignment for 5-Pin Terminal Block to RS-232/RS-422/RS-485 Connectors

Pin#	RS-232	RS-422 4-Wire RS-485	2-W RS-485
1	RxD	TxD+	Data+
2	CTS	TxD-	Data-
3	TxD	RxD+	N/A
4	RTS	RxD-	N/A
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)

3.5.3 SE5901B Pin Assignments

DB9 to RS-232/RS-485/RS-422 connectors

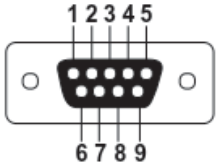


Figure 3.5 DB9 Pin Number

Table 3.8 SE5901B Pin Assignment for DB9 to RS-232/RS-485 Connector

Pin#	RS-232 Full Duplex	RS-485 Half Duplex
1	DCD	N/A
2	RxD	N/A
3	TxD	Data+
4	DTR	N/A
5	SG (Signal Ground)	SG (Signal Ground)
6	DSR	N/A
7	RTS	Data-
8	CTS	N/A
9	RI	N/A

2 x 7-pin Male Terminal Block for RS-232/485(COM 1),RS-232(COM 2) Relay and DI

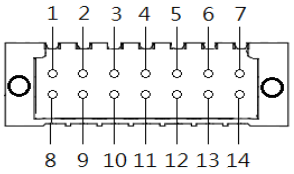


Figure 3.6 2 x 7-pin Male Terminal Block

Table 3.9 SE5901B 2 x 7-pin Male TB for RS-232/485(COM 1),RS-232(COM 2) Relay and DI pin-assignment

Pin#	DI and Relay	COM1 (RS-232)	COM1 (RS-485)	COM2 (RS-232)
1	DI1	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
2	DI2	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
3	Relay 1 -	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
4	Relay 1+	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
5	Relay 2 -	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
6	Relay 2+	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
7	*Dedicated for COM*	SG (Signal Ground)	SG (Signal Ground)	-
8	*Dedicated for COM*	Rx	-	-
9	*Dedicated for COM*	CTS	-	-
10	*Dedicated for COM*	Tx	Data +	-
11	*Dedicated for COM*	RTS	Data -	-
12	*Dedicated for COM*	-	-	SG (Signal Ground)
13	*Dedicated for COM*	-	-	Rx
14	*Dedicated for COM*	-	-	Tx

3.5.4 SE5908A/ SE5916A Pin Assignments

DB9 to RS-232/RS-485/RS-422 connectors

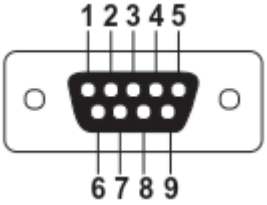


Figure 3.7 DB9 Pin Number

Table 3.10 SE5908A/16A Pin Assignment for DB9 to RS-232/RS422/RS-485 Connectors

Pin#	RS-232	RS-422	RS-485
1	-	-	-
2	RxD	TxD+	Data+
3	TxD	RxD+	-
4	-	-	-
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	-	-	-
7	RTS	RxD-	-
8	CTS	TxD-	Data-
9	-	-	-

5-Pin Terminal Block to RS-232/RS-485/RS-422 connectors

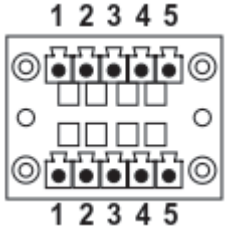


Figure 3.8 Terminal Block (TB-5) Pin Number

Table 3.11 SE5908A/16A Pin Assignment for 5-Pin Terminal Block to RS-232/RS-422/RS-485 Connectors

Pin#	RS-232	RS-422 4-Wire RS-485	2-W RS-485
1	RxD	TxD+	Data +
2	CTS	TxD-	Data -
3	TxD	RxD+	-
4	RTS	RxD-	-
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)

3.5.5 SE5908/ SE5916 Pin Assignments

RJ45 to RS-232/RS-485/RS-422 connectors

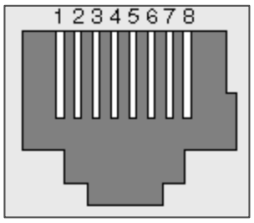


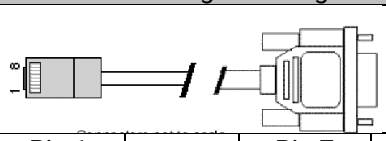
Figure 3.9 SE5908/SE5916 Serial port on RJ45 Pin Numbering

Table 3.12 MB5908/16 Pin Assignment for RJ45 to RS-232/RS422/RS-485 Connectors

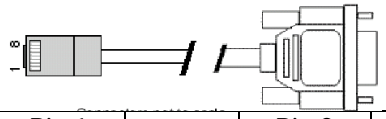
Pin#	RS-232	RS-422	RS-485
1	RTS	-	-
2	DTR	Tx -	-
3	TxD	Tx +	-
4	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	RxD	Rx +	Data +
7	DSR	Rx -	Data -
8	CTS	-	-

RJ45 to RS-232/RS-485/RS-422 accessories provided by ATOP

- **50891791G - RJ45 TO DB9 CABLE-FEMALE:**

RJ45			Straight Through Female DB9	
				
RTS	Pin 1	⇔	Pin 7	RTS
DTR	Pin 2	⇔	Pin 4	DTR
TXD	Pin 3	⇔	Pin 3	TXD
SG	Pin 4	⇔	Pin 5	SG
SG	Pin 5	⇔		
RXD	Pin 6	⇔	Pin 2	RXD
DSR	Pin 7	⇔	Pin 6	DSR
CTS	Pin 8	⇔	Pin 8	CTS

- **50891971G - RJ45 TO DB9 CROSS OVER CABLE-FEMALE:**

RJ45			Cross Over Female DB9	
				
RTS	Pin 1	⇔	Pin 8	CTS
DTR	Pin 2	⇔	Pin 6	DSR
TXD	Pin 3	⇔	Pin 2	RXD
SG	Pin 4	⇔	Pin 5	GND
SG	Pin 5	⇔		
RXD	Pin 6	⇔	Pin 3	TXD
DSR	Pin 7	⇔	Pin 4	DTR
CTS	Pin 8	⇔	Pin 7	RTS

50891781G - RJ45 TO DB9 CABLE-MALE:

RJ45			Straight Through Male DB9	
RTS	Pin 1	⇔	Pin 7	RTS
DTR	Pin 2	⇔	Pin 4	DTR
TXD	Pin 3	⇔	Pin 3	TXD
SG	Pin 4	⇔	Pin 5	SG
SG	Pin 5	⇔		
RXD	Pin 6	⇔	Pin 2	RXD
DSR	Pin 7	⇔	Pin 6	DSR
CTS	Pin 8	⇔	Pin 8	CTS

3.5.6 SE59XX Pin Assignments for LAN Interface

RJ45 connectors for 10/100/1000Base-T(X) Ethernet

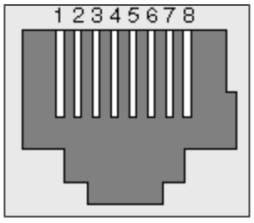


Figure 3.10 SE59XX Ethernet Port on RJ45 with Pin Numbering

Table 3.13 SE59XX Pin Assignment for RJ-45 Connector

10/100/1000Base-T(x)								
Pin#	1	2	3	4	5	6	7	8
Signal	Tx+	Tx-	Rx+	-	-	Rx-	-	-
1000Base-T								
Pin#	1	2	3	4	5	6	7	8
Signal	BI_DA+	BI_DA-	BI_DB+	BI_DC+	BI_DC-	BI_DB-	BI_DD+	BI_DD-

4 Installing Node-RED

Warning: this step is applicable only if you have not purchased ATOP standard Node-RED product or ATOP SD/microSD/USB stick preloaded with the Node-RED application. If you did, you can jump to Chapter 5 below.

4.1 Firmware upgrade

It may be necessary to upgrade firmware from time to time. There are two ways to upgrade the firmware on the SE59XX platform:

4.1.1 Use Device Manager or Device Management Utility

Please use a CAT5E cable to connect SE59XX to a PC running Windows where ATOP Device Management utility is already installed. To install Device Management Utility, please download the latest release from ATOP Website and follow its dedicated user manual for the installation.
The device doesn't have necessarily to be directly connected to the PC, as long as it is inside the same LAN. Atop Management Utility will scan the whole network automatically.

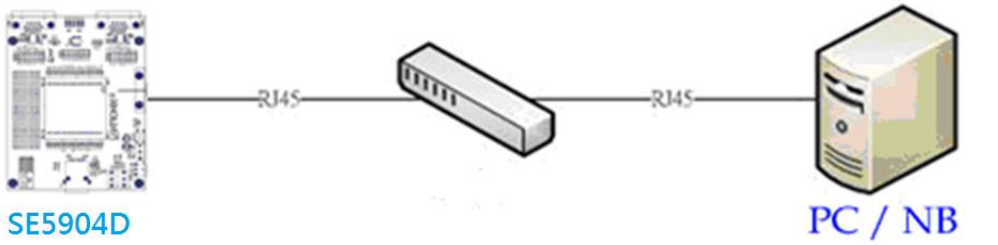


Figure 4.1 SE59XX connection scheme (example on SE5904D)

Now, please power on the device and run ATOP's Device Management Utility from your Host PC. Once the device is running, the utility will list all devices found. If the device doesn't show up, push the leftmost button (Rescan function). Once identified, select the device by mouse left button and select "Firmware" >> "Download Firmware" as per Figure 4.2.

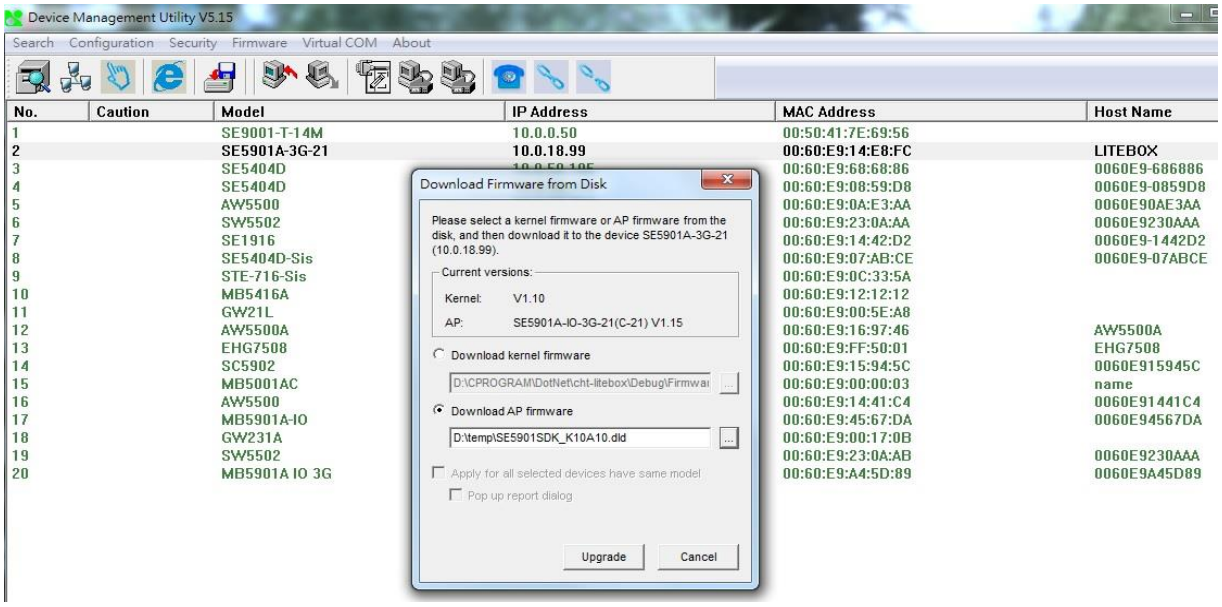


Figure 4.2 Firmware update prompt

Select the firmware (Kernel or AP) from this dialog and select the upgraded file as Figure 4.3. Then, click on the "**Upgrade**" button to upgrade the firmware selected.

Note: This example is made with SE5901A. All other models of SE59XX family share the same method.

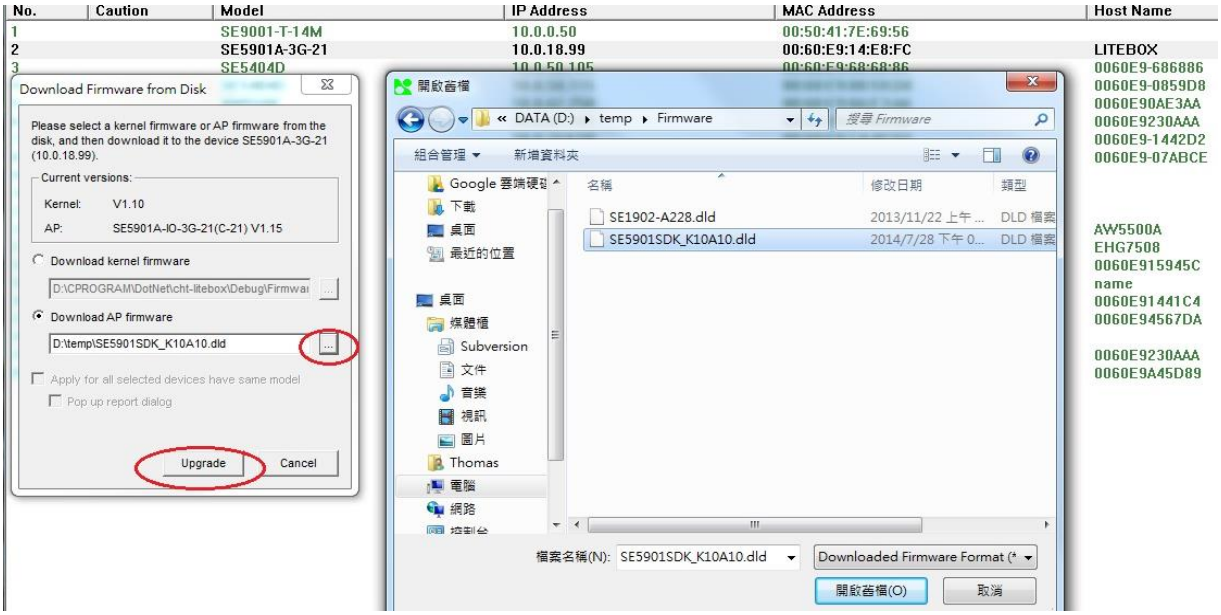


Figure 4.3 Firmware selection

Note that the extension file name of upgraded firmware should be .dld

4.1.2 Use boot-loader update via console port

Prepare a Debug Cable (RJ45 to Serial) and a CAT5E Ethernet cable. Then, follow below figure to connect the Debug port to PC's COM and CAT5E cable to connect to the Device's LAN1 Ethernet port to any Host PC's Ethernet port.

Figure 4.4 Console firmware update- connections

On your PC, run Windows' "Super Terminal" setup COM port parameters as follows:

- Port: the connected COM port
- Baud Rate: 115200 bps
- Parity: none
- Data: 8 data bits
- Stop: 1 stop bit
- Flow control: none

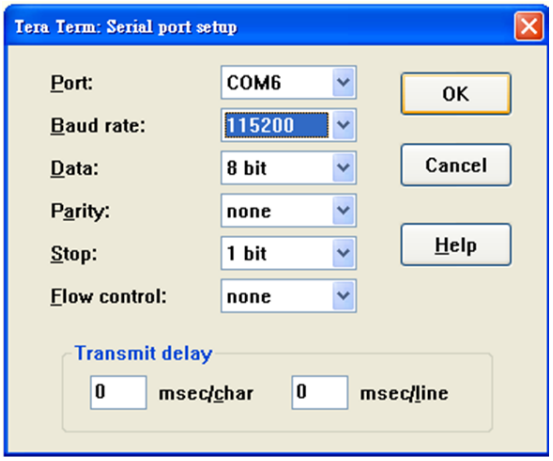


Figure 4.5 COM port Parameters for Console Firmware update

With this method, TFTP protocol is used. The TFTP client is already set-up and running inside the SE59XX platform. Thus, the user needs to execute TFTP server in Windows. An open source version is available for download and can be found as "tftpd32". Screenshot below shows "tftpd32.exe" after running the application.

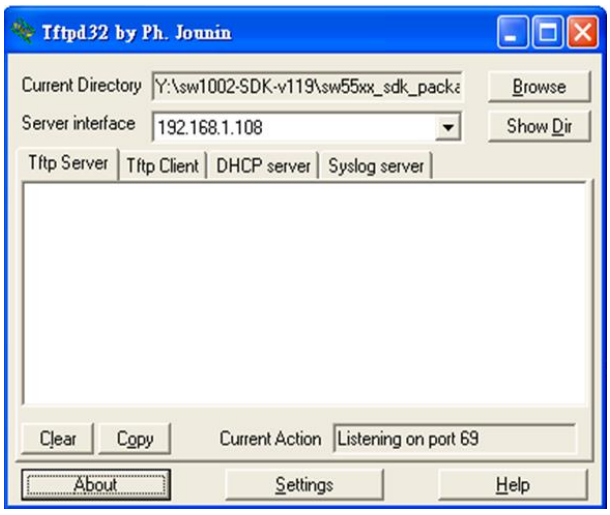


Figure 4.6 TFPD32 appearance after execution

Now, setup the IP address of the TFTP server. The current folder is the one where "tftpd32.exe" is located.
After executing TFTP server, reboot the target SE59XX platform and press the Escape ("Esc") key immediately. A boot-loader menu will be shown as Figure 4.7.

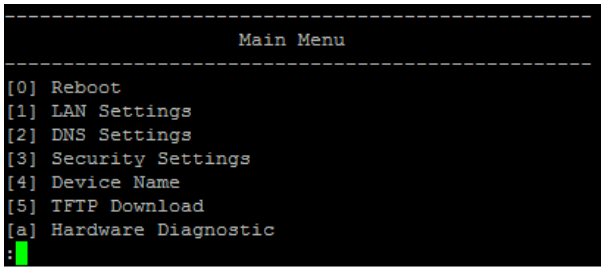


Figure 4.7 SE5904D Boot loader menu

Select item 1 to enter "LAN Setting" menu as Figure 4.8, and setup IP/Netmask/Gateway of LAN1 as Figure 4.9

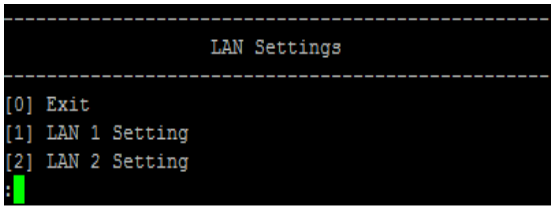


Figure 4.8 LAN Settings

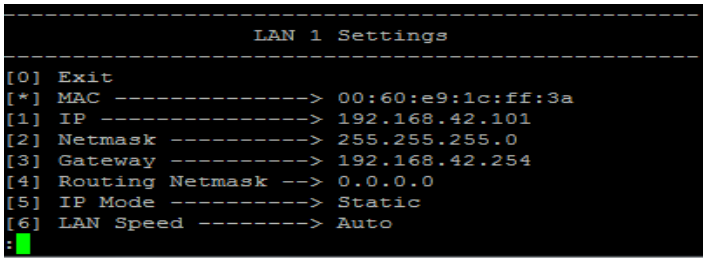


Figure 4.9 LAN1 settings

Enter 0 to exit to upper layer menu and select 5 to enter the "TFTP Download" menu, then select 1 to setup TFTP **server IP as Figure 4.10**

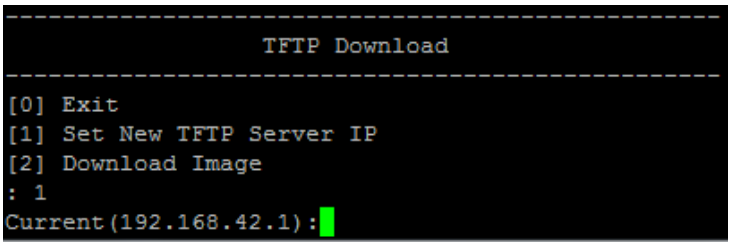


Figure 4.10 TFTP download menu

After the setup of the server IP is completed, select 2 to download the firmware image.

Note: the extension of the firmware should be .dld

4.2 Verify current firmware version

There are two methods to verify the firmware version:

1) Use Device Manager or Device Management Utility (Serial Manager) to check version number as per Figure 4.11. (Device Manager is currently supported to Simplified Chinese release)

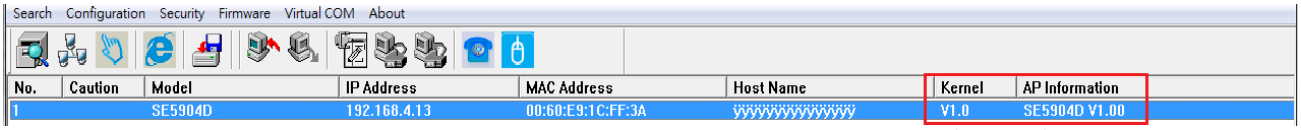


Figure 4.11 Firmware version in Device Management Utility (English)

2) Use a debug line as per Paragraph 4.1.1 above to connect console port of the device. After boot up, type "atop_show_ver" in the console command line to check current version as Figure 1-13 shown. The red rectangle shows information of boot-loader (V1.00), Kernel(V1.00) and AP (V1.00) version number.

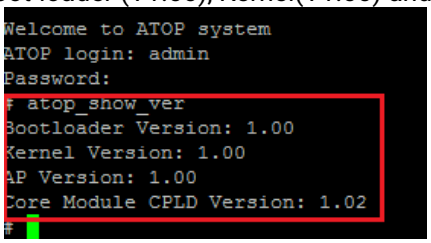


Figure 4.12 Firmware version - Console

4.3 Login or Remote Login to the device

4.3.1 Factory default settings

- IP address: 10.0.50.100
- Username: admin
- Password: default

4.3.2 Remote Login

1) Setup or read FTP account and password from Atop boot-loader menu as Figure 1-16
2) Use any tools supporting the telnet protocol such as "SSH" inside of Windows.
3) Enter SE59XX_TARGET_IP via SSH using putty utility.
4) Login account as first step shown

4.3.3 Use a debug command line to Login

If you're not pressing "Esc" button within 3 seconds from boot-up, the device will enter Linux login mode as per screenshot below.

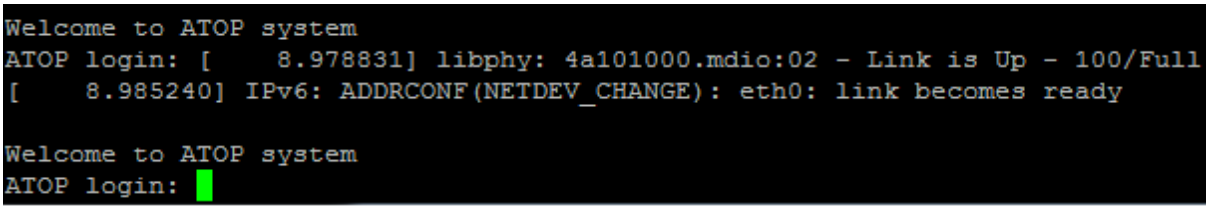


Figure 4.13 Command line Login

4.3.4 Check and Modify the network settings

If you don't need to change the default network settings, you can skip this step. In order to change IP address settings, DNS, Default Gateway and Subnet mask please follow the steps below.

1) Login to the device with SSH/Telnet
2) Use ifconfig command to check current settings and to modify IP address and subnet mask

```
# ifconfig                    <-- show ethernet infomations
```

```
eth0      Link encap:Ethernet  HWaddr 00:60:E9:1E:6C:48
          inet addr:192.168.4.33  Bcast:192.168.4.255  Mask:255.255.255.0 <- Example
          inet6 addr: fe80::260:e9ff:fe1e:6c48/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:391236 errors:0 dropped:3314 overruns:0 frame:0
          TX packets:85095 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:84832013 (80.9 MiB)  TX bytes:5144093 (4.9 MiB)
          Interrupt:56
……
```

```
# ifconfig eth0 10.0.50.1 netmask 255.255.0.0 <- change IP & subnet of eth0 to 10.0.50.1 and 255.255.0.0
```

```
eth0      Link encap:Ethernet  HWaddr 00:60:E9:1E:6C:48
          inet addr:10.0.50.100  Bcast:192.168.4.255  Mask:255.255.0.0 <- Changed
          inet6 addr: fe80::260:e9ff:fe1e:6c48/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:391236 errors:0 dropped:3314 overruns:0 frame:0
          TX packets:85095 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:84832013 (80.9 MiB)  TX bytes:5144093 (4.9 MiB)
          Interrupt:56
…….
```

3) Use vi editor to modify DNS settings by changing /etc/resolv.conf

```
# vi /etc/resolv.conf      <-- setup DNS
```

```
nameserver    8.8.8.8                    <-- modify your DNS IP (Google)
nameserver    168.95.1.1                 <-- modify your DNS IP (CHT)
nameserver    0.0.0.0
nameserver    0.0.0.0
nameserver    0.0.0.0
nameserver    0.0.0.0
nameserver    0.0.0.0
nameserver    0.0.0.0
nameserver    ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
nameserver    ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
```

4.4 Install Node-RED on SD card or USB pen-drive

Please follow the steps described below if you didn't purchase the pre-loaded SD-card or USB pen-drive with Node-RED installed on if and if you downloaded the Node-RED library from ATOP's website.

Note: if you purchased the pre-loaded SD-card or USB pen-drive from ATOP you can skip this step

4.4.1 Login to SE59XX

- Plug in the empty USB pen-drive or SD card in the SD card slot or in the USB socket. It is recommended to have more than 1GB storage available
- Turn on the device and wait until the RUN LED is blinking
- Connect the LAN cable to your computer. Please make sure the IP address of your computer is on the same subnet of the device.
- If working on Windows Operating System environment, use any tool supporting telnet protocol such as "ssh" inside of Windows (for example, putty).
- Enter SE59XX_IP_address via ssh.
- Login account as first step shown

Factory default settings

- IP address: 10.0.50.100
- Username: admin
- Password: default

4.4.2 Show SD card/ USB pen-drive partitions to discover the device name

Once the device is turned on, logged in, with the SD-card or USB plugged in properly, please input the following command and press "Enter". The example below shows what the result can be, but the actual result depends on the capacity of the drive used.

```
# fdisk -l    <-- lowercase L

Disk /dev/mtdblock0: 0 MB, 524288 bytes
255 heads, 63 sectors/track, 0 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/mtdblock0 doesn't contain a valid partition table

Disk /dev/mtdblock1: 0 MB, 131072 bytes
255 heads, 63 sectors/track, 0 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/mtdblock1 doesn't contain a valid partition table

Disk /dev/mtdblock2: 0 MB, 393216 bytes
255 heads, 63 sectors/track, 0 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/mtdblock2 doesn't contain a valid partition table

Disk /dev/mtdblock3: 5 MB, 5242880 bytes
255 heads, 63 sectors/track, 0 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
Disk /dev/mtdblock3 doesn't contain a valid partition table

Disk /dev/mtdblock4: 18 MB, 18874368 bytes
255 heads, 63 sectors/track, 2 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/mtdblock4 doesn't contain a valid partition table

Disk /dev/mtdblock5: 41 MB, 41943040 bytes
255 heads, 63 sectors/track, 5 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/mtdblock5 doesn't contain a valid partition table
Disk /dev/mmcblk0: 7932 MB, 7932477440 bytes
4 heads, 16 sectors/track, 242080 cylinders
Units = cylinders of 64 * 512 = 32768 bytes

   Device Boot      Start         End      BlocksId  System
/dev/mmcblk0p1               1      183106     5859384   83 Linux
/dev/mmcblk0p2          183107      242080     1887168   82 Linux swap
```

4.4.3 Delete existing partitions on the SD card/ USB pen-drive

If the external storage drive has a partition, this should be deleted before proceeding.

Note: Please make sure that you have backed up all your personal data saved on the USB pen-drive or the SD-card. The following activities may result in a permanent loss on all information on the external device.

```
# fdisk /dev/ mmcblk0  <-- please select your storage device name

The number of cylinders for this disk is set to 242080. There is nothing wrong with that, but
this is larger than 1024, and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): d
Partition number (1-4):1    <-- delete a partition 1

Command (m for help): d
Selected partition 2        <-- delete a partition 2 (auto select)

Command (m for help): p  <-- print partition information

Disk /dev/mmcblk0: 7932 MB, 7932477440 bytes
4 heads, 16 sectors/track, 242080 cylinders
Units = cylinders of 64 * 512 = 32768 bytes

      Device Boot      Start         End      Blocks  Id System

Command (m for help): w   <-- write table to disk and exit
```

4.4.4 Create a new partition

Please follow the steps below in order to prepare the partition for installing Node-RED. It is recommended to create one partition for the primary files and one partition for Swap.

```
# fdisk /dev/ mmcblk0  <-- please select your storage device name

The number of cylinders for this disk is set to 242080. There is nothing wrong with that, but
this is larger than 1024, and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): n   <------ add a partition 1 (for example, 6G)
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-242080, default 1): Using default value 1 (press enter)
Last cylinder or +size or +sizeM or +sizeK (1-242080, default 242080): +6000M

Command (m for help): n   <------ add a partition 2 (Extra space)
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (183107-242080, default 183107):Using default value 183107 (press enter)
Last cylinder or +size or +sizeM or +sizeK (3635-3814, default 3814):Using default value 3814
(press enter)

Command (m for help): p  <-- print partition information
   Disk /dev/mmcblk0: 7932 MB, 7932477440 bytes
      4 heads, 16 sectors/track, 242080 cylinders
      Units = cylinders of 64 * 512 = 32768 bytes

   Device Boot      Start         End      Blocks   Id  System
  /dev/mmcblk0p1           1      183106     5859384    83  Linux
  /dev/mmcblk0p2      183107      242080     1887168    83  Linux

Command (m for help):t      <-- change partiton 2 to swap type
Partition number (1-4): 2
Hex code (type L to list codes): 82

Command (m for help): p  <-- print partition information
   Disk /dev/mmcblk0: 7932 MB, 7932477440 bytes
      4 heads, 16 sectors/track, 242080 cylinders
      Units = cylinders of 64 * 512 = 32768 bytes

   Device Boot      Start         End      Blocks   Id  System
  /dev/mmcblk0p1           1      183106     5859384   83  Linux
  /dev/mmcblk0p2      183107      242080     1887168   82  Linux swap

Command (m for help): w     <-- write table to disk and exit
```

4.4.5 Prepare to upload Node-RED

There are several ways to load Node-RED libraries into the SD Card or into the USB Stick. In this manual, we will use FTP with the open-Source tool called “ftpd64.exe”. “ftpd64.exe” is a third party application that is recommended. More information about “tftpd64.exe”, its availability URL is available in Chapter 9 below

On ATOP’s website there will be four different archives available (YYYY, MM, DD represent the release year, month and day respectively):

1) node-red_sdk_release_YYYYMMDD.tar.bz2 - >> SE59XX SDK
2) Image_SE59XX-NR_vXXX.dld >> SE59XX Node-RED image version X.XX
3) nodered_preinstall.tar.gz >> Node-RED libraries
4) python.tar >> Python Libraries

Then, run Ttftpd64.exe and select the folder in which Node-RED files have been extracted and the Network interface connected to SE59XX.

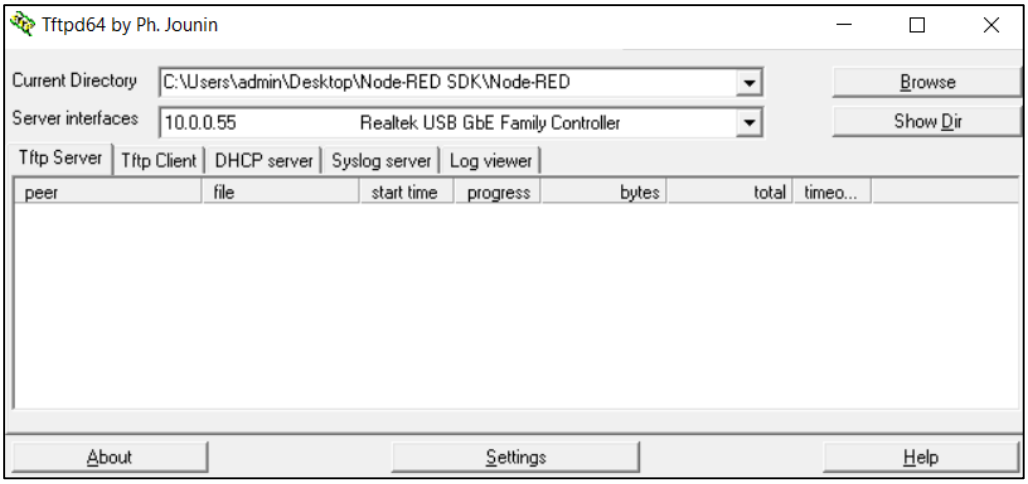


Figure 4.14 Example of Ttftpd64 configuration for uploading files

4.4.6 Prepare the Node-RED installation script

Once the device is turned on, logged in, with the SD-card or USB plugged in properly, please connect to the device using SSH/telnet. In order to install Node-RED automatically, it will be necessary to have SE59XX to run an installation script that will:

- prepare the SD card/USB pen drive partition by formatting it
- copy the contents of nodered_preinstall.tar.gz and/or python.tar archive that has been put in a folder of your choice in step 0 to the SD card/USB pen drive
- decompress those files into the storage
- install the software
- make some basic settings

The installation script is already pre-loaded on the device and it’s called Node-RED.sh.

If you’d like to customize your own installation script, you can copy the file from /usr/bin folder and edit it using SSH/telnet command line directly. When creating or modifying the file please make sure to provide the correct partition names used, as well as the correct .tar.bz file names as they are stored in the folder and the IP address of your computer.

If you're using your own script, it is suggested to save or upload it in /jffs2 folder, so that the file won't be deleted with a firmware upgrade. Please make sure the execution settings (rwx rights, read-write-execute) are set to 777 using the following command on the command line.

```
# chmod 777 Node-RED.sh
```

Then, execute the script using the following command on the command line to install Node-RED.
If the installation script has been renamed, please use the new file name

```
# ./ Node-RED.sh  install<-- Execute script
```

Note: this script installs Node-RED, npm and Node.JS only. In order to install Python, use Node-RED.sh pyinstall

During the script execution, you will be asked to input:

- on which physical support the installation should take place (USB stick or SD card)
- what is the IP address of the computer on which TFTPD is running
- whether or not you would like to format the physical support

Note: Linux filesystem is the only one that allows Node-RED to run successfully. If you had used your USB stick/ SD card on Windows Operating system before, please format it.

The script launching sequence will look as per below Figure 4.15 and Figure 4.16.

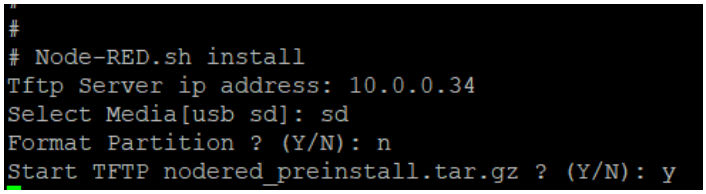


Figure 4.15 Node-RED Auto_Node_Red.sh installation script start

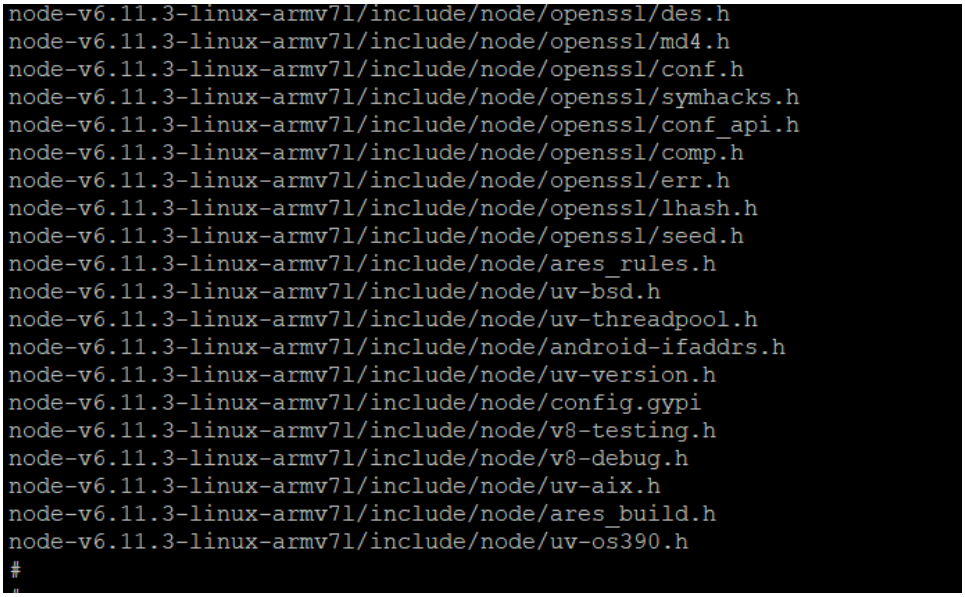


Figure 4.16 Node-RED Auto_Node_Red.sh installation script end

5 Basic Configuration

5.1 Working “out of the box”

The device can be accessed from Web User Interface from the factory default IP address (10.0.50.100). Once inputting the access credentials, you will access ATOP Device’s configuration dashboard (see below Figure 2.1) that will guide you through the whole basic set-up.

Every device is equipped with a built-in web server in the firmware. Therefore, the device can be accessed by using a web browser for configuring by entering the device’s IP address (default IP address is 10.0.50.100) in the URL field of your web browser. An authentication will be required and you will have to enter the username (Default value is “admin”) and password (Default value is “default”) for accessing the web interface as shown in Figure 5.1. Figure 5.2 shows the default landing page, once login is successful, and Figure 5.3 shows the extended menu.

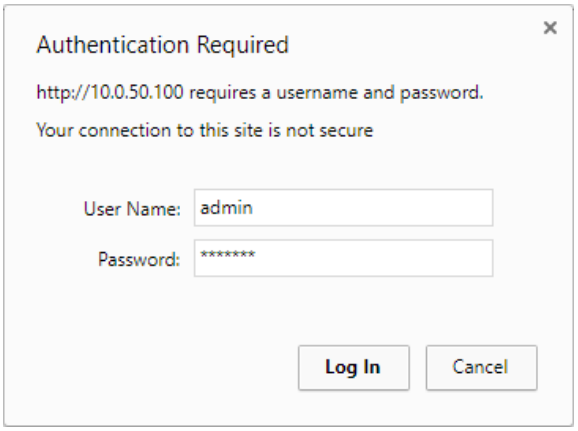


Figure 5.1 Authentication Required for Accessing Web Interface

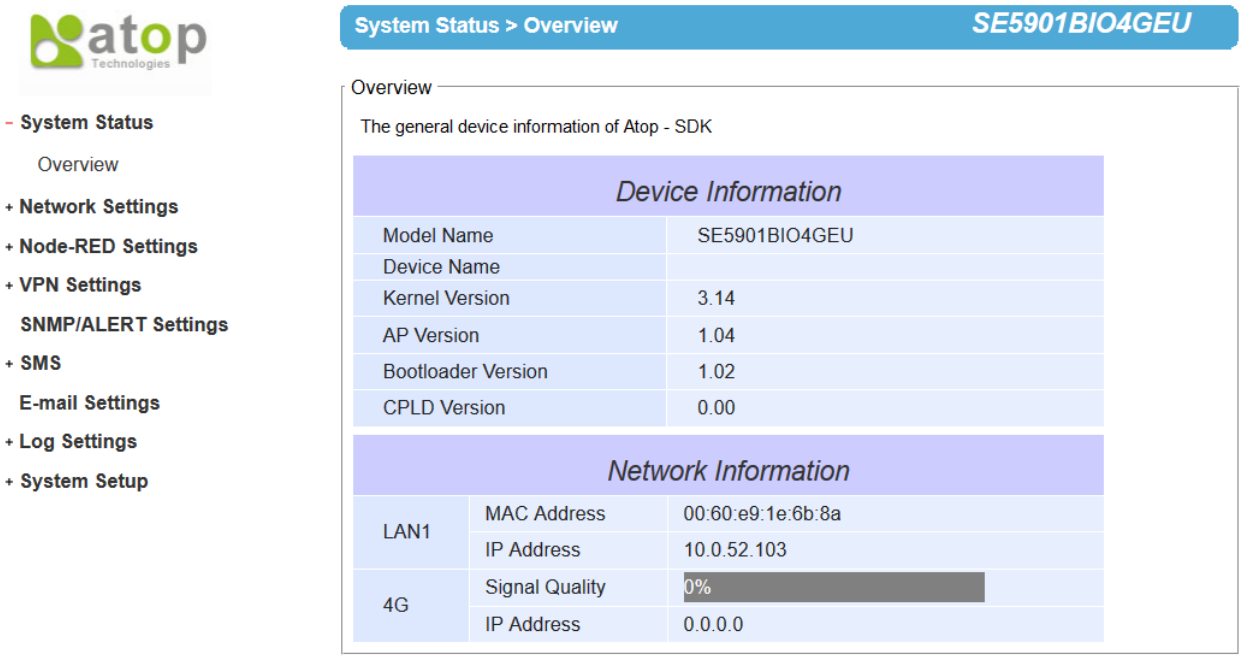


Figure 5.2 ATOP Default landing page on http://10.0.50.100

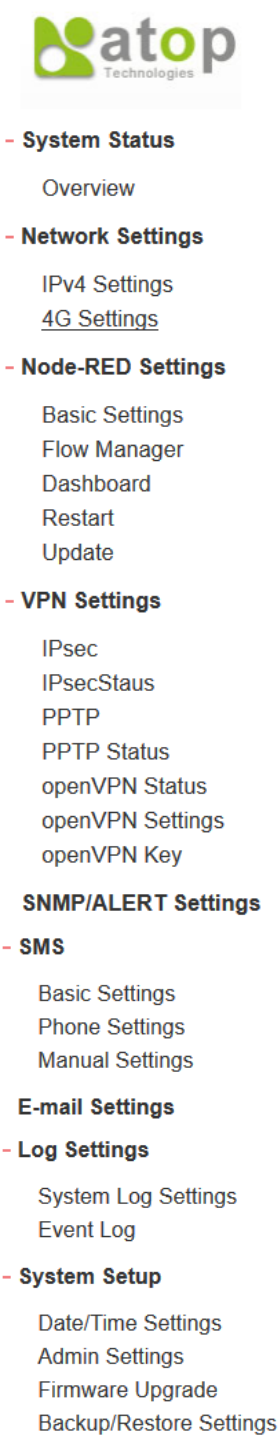


Figure 5.3 Configuration extended menu (example taken on SE5901B-Node-RED)

5.2 Configuring Automatic IP Assignment with DHCP

A DHCP server can automatically assign IP addresses, Subnet Mask and Network Gateway to LAN interface. You can simply check the **"DHCP (Obtain an IP Automatically)"** checkbox in the Network Setting dialog using Atop's **Device Management Utility©** and then restart the device. Once restarted, the IP address will be configured automatically.

5.3 Web Overview

In this section, current information on the device's status and settings will be displayed. An example of SE59XX's overview page is shown in Figure 5.4.

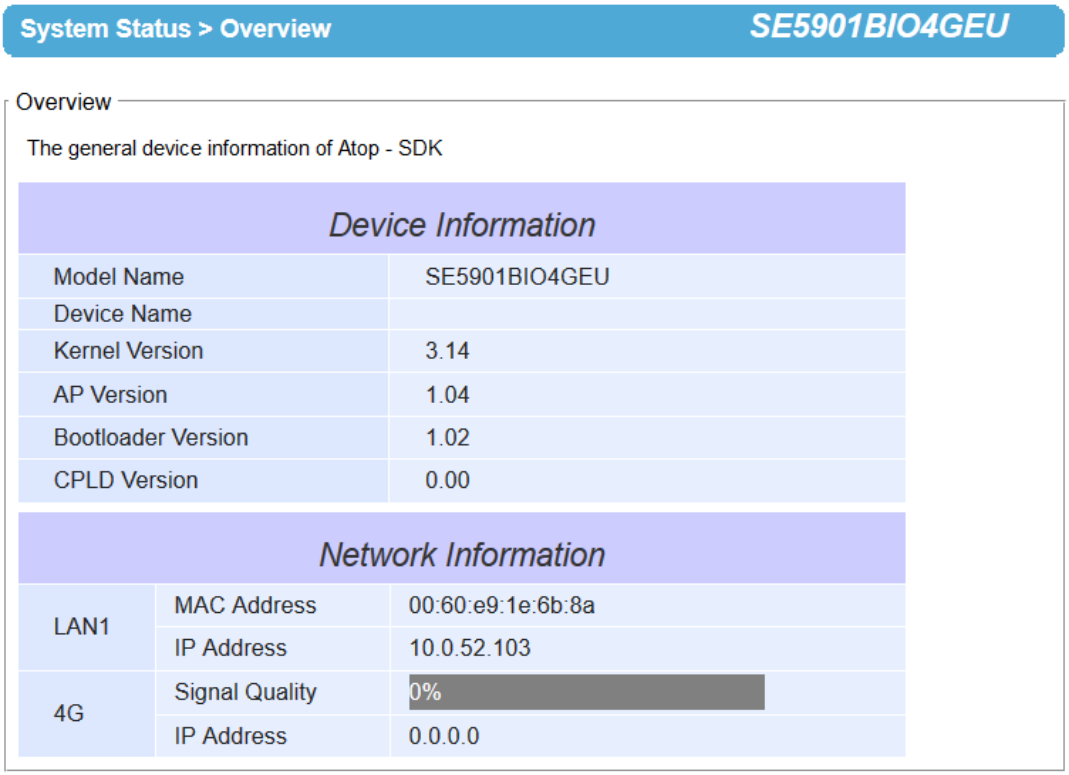


Figure 5.4 Overview Web Page (example on SE5901B)

In detail, the following information is given and divided into 2 parts (Device Information and Network Information):

- Device Information
  - **Model Name**, as its name implies, shows the device's model
  - **Device Name** shows a given name of the device in which the default value is the MAC address of the LAN interface.
  - **Kernel Version** is the value of the version of the kernel firmware of the device.
  - **AP Version** is the value of the version of the application firmware of the device.
  - **Bootloader Version** is the version of the program that loads the operating system of the device.
  - **CPLD Version** is the version of the Complex Programmable Logic Device (logic device) of the device.
- **Network Information** shows information about the wired network interface on the device.
  - **LAN:** This will display the current **MAC Address**, and **IP Address** of the Ethernet interface.
  - **4G (SE5901B only):** This will display signal quality and Public IP address on 4G network

5.4 IPv4 Settings (Network settings)

In this section, both network interfaces and related network settings of the SE59XX device can be configured. There are four sets of parameters which are **LAN1 Settings**, **LAN2 Settings**, **Default Gateway**, and **DNS Server** that can be entered as shown in Figure 5.5. First, **LAN1 Settings** part will allow you to configure the **IP Address**, **Subnet Mask**, and **Default Gateway** for your wired LAN1 network. You can check the box behind **DHCP** option to obtain an IP address automatically. If you checked the box, the rest of the options for **LAN1 Settings** will be greyed out or disabled. Second, **LAN2 Settings** is the same as LAN1 Settings but for the second Ethernet interface. Third, **Default Gateway** part is where you can select the default gateway network for your serial device server. You can either select **LAN1** or **LAN2** by clicking on the corresponding radio button. Fourth, **DNS Server** part is where you can specify the IP Address of your **Preferred DNS** (Domain Name Server) and **Alternate DNS**. If the SE59XX device is connected to the Internet and should connect to other servers over the Internet to get some services such as Network Time Protocol (NTP) server, you will need to configure the DNS server in order to be able to resolve the host name of the NTP server. Please consult your network administrator or internet service provider (ISP) to obtain local DNS's IP addresses.

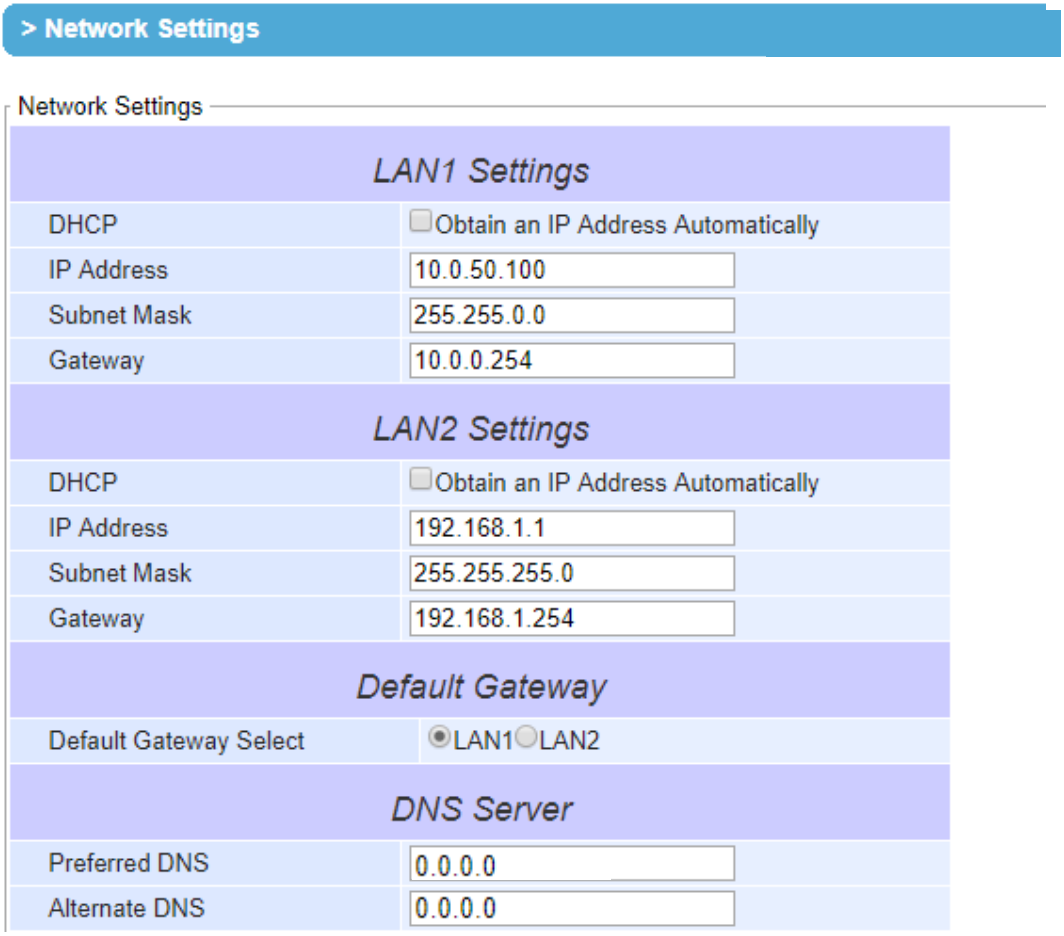


Figure 5.5 Network Settings Web Page – example on SE5901

For SE5901B only, there is another set of parameters for **NAT Settings** that can be configured at the end of the **Network Settings** Webpage as shown in Figure 5.6 and Figure 5.7. NAT is referred to Network Address Translation which is a technique that allows SE5901B to create a local IP network or subnetwork with private IP addresses that can connect to the Internet through a public IP address via its Wide Area Network (WAN) port. The SE5901B will map the private IP address and port of a local device connected to its local interface to a public port on its public interface (WAN port). Note that the local IP network is called **Virtual IP** in SE5901B. To enable **NAT** function on SE5901B, check on the **Enable** box behind **NAT** option under **NAT Settions** part as shown in Figure 5.6.

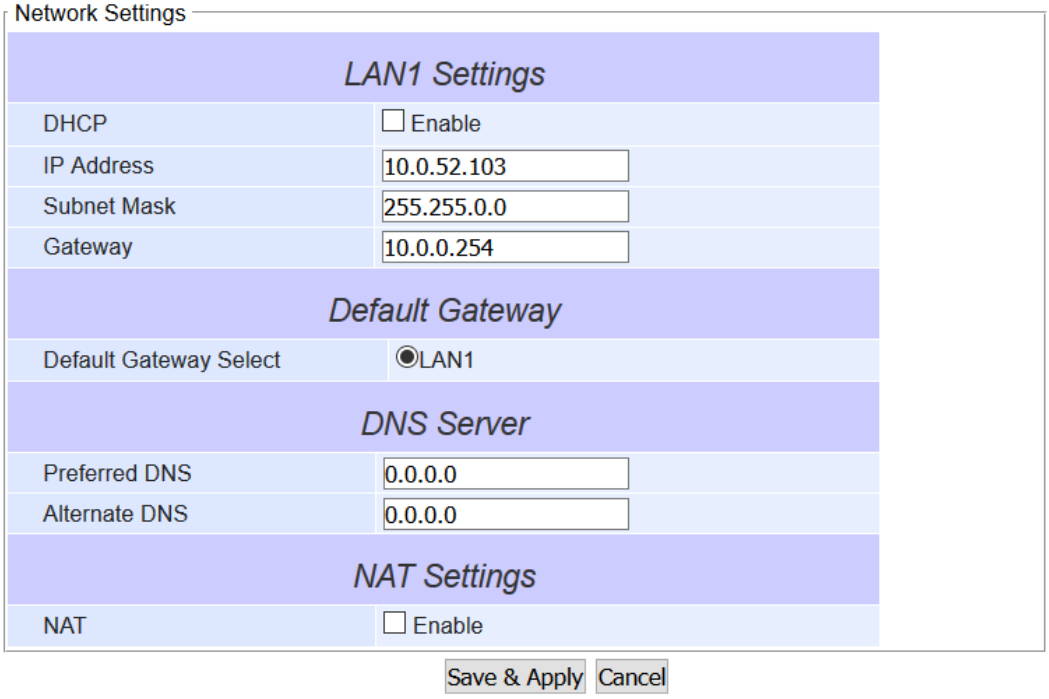


Figure 5.6 NAT Settings under Network Settings Web Page for SE5901B

When **NAT** function is enabled on SE5901B, additional set of parameters which are **WAN Port** and **DHCP Server** fields will appear as shown in Figure 5.7. The **WAN Port** option will allow the user to select a port on SE5901B that can access or connect to the Internet from the drop-down list such as LAN1. Note that SE5901B only supports one LAN (Local Area Network) port that can access the Internet through the **WAN Port** or is designated as the port that can connect to the Internet in this option. The next option is the **DHCP Server** or Dynamic Host Configuration Protocol Server which is another function on SE5901B under the **NAT Settings**. This will allow SE5901B to automatically assign IP address for its local network. If the **DHCP Server** option is enabled (by checking the **Enable** box behind **DHCP Server** option), **IP Pool Start Address** and **IP Pool End Address** fields will appear under it. The IP Pool Addresses are the range of addresses that **DHCP Server** will be used to configure local IP addresses. The user can enter the starting and ending addresses inside these two fields**. The DHCP** Server function inside SE5901B can only support one LAN port and provide that port with IP address in the given range (from **IP Pool Start Address** to **IP Pool End Address**). Note that the range must be in NAT LAN port's network segment.

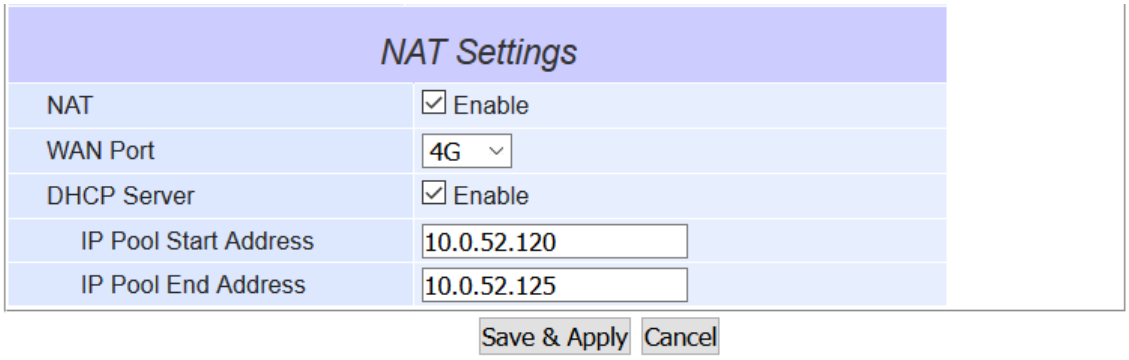


Figure 5.7 Enabling of NAT Settings with Additional Parameters for SE5901B

After finishing the network settings configuration, please click the **Save & Apply** button to save all changes that have been made. Finally, the web browser will be redirected to the **Overview** page as shown in Figure 5.4. If you would like to discard any setting, please click the **Cancel** button.

5.5 3G Settings or 4G Settings

SE5901B has a built-in 3G or 4G cellular network interface depending on your purchased model. On this web page, you can check the status of your cellular connection, set parameters for your cellular (3G or 4G) network configuration, and set three phone numbers that can reboot the SE5901B. Figure 5.8 shows an example of **3G Settings** web page which is divided into three parts: **3G Information**, **3G Configuration**, and **Phone Number Settings**.

Note: The user is required to insert a valid SIM card of your local cellular network operator (3G or 4G) into the SIM card socket inside the chassis of SE5901B.

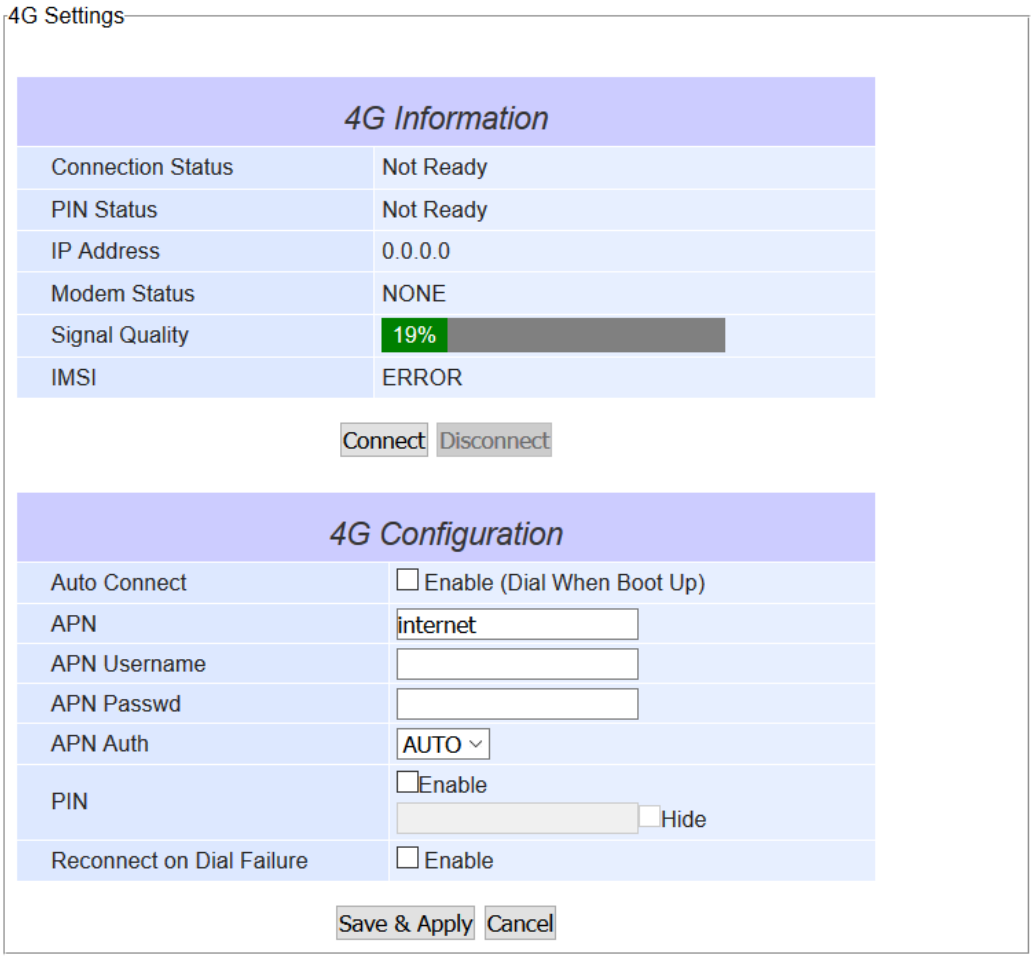


Figure 5.8 3G Settings Web Page – screenshot taken on SE5901B

Under the **3G Information** part, you can inspect the following information of your cellular network interface: **Connection Status**, **PIN Status**, **IP Address**, **Modem Status**, and **Signal Quality**. Table 5.1 describes each field under the 3G Information part. Under the 3G Information part, there are **Connect** button and **Disconnect** button that allow you to control the cellular connection.

Table 5.1 Description of 3G/4G Information

Field Name	Description	Possible Values
Connection Status	Reports the status of cellular data connection	No Sim Card Inserted, Disabled, Disconnect, Connect, Dialling
PIN Status	Reports the status of the PIN	READY or some wrong!
IP Address	IP address assigned by the cellular operator	-
Modem Status	Reports the status of cellular modem	3G-UTRAN, E-UTRAN,…, Unknown Status
Signal Quality	Indicates the cellular network signal strength in percentage and bar graph.	0% up to 100%
IMSI	The International Mobile Subscriber Identity or IMSI is used to identify the user of a cellular network and is a unique identification associated with all cellular networks. It is stored as a 64 bit field and is sent by the phone to the network	64-bit number

Under the **4G Configuration** part, you can configure how the cellular connection is established. A detailed explanation of the fields is available in Table 5.2.

First option is the **Auto Connect**. You can check the box in front of **Enable (Dial When Boot Up)** to let the SE5901B automatically dials 4G Modem when the device finished booting up. Next, the **APN** option which is the Access Point Name used for establishing the cellular connection. This name depends on your local cellular network operator's specifications. The default value is "internet". Some Network operators, require Authentication through the **APN**. SE5901B supports this feature, allowing the user to key in (in dedicated fields) username and password for APN Authentication. It is recommended to set the **Authentication method** as "AUTO".
Next, the **PIN** or Personal Identification Number option is the 4-digit code used to unlock the SIM of the 3G Modem on the SE5901B. You can enable this PIN security by checking the **Enable** box. After enabling the **PIN** option, you will be able to enter the **PIN Code** in the textbox. Note that the default display of the textbox is to hide the code. You have an option to uncheck the box in front of **Hide** to see the PIN Code. Finally, the last option is to enable the **Reconnect on Dial Failure** option by checking the **Enable** box. The default for this option is disable. After finishing the network settings configuration, please click the **Save & Apply** button to save all changes that have been made. A pop-up window will show up with **"Please wait for a while…"** message. Then, the web browser will return to the **3G Settings**/**4G Settings** web page again.

Table 5.2 Description of 4G Configuration fields

Field Name	Description	Factory Default
Auto-connect	Allows connection to connect to the internet via cellular as soon as the device is booted up	Unchecked
APN	Defines the APN that the cellular network needs in order to be online	internet
APN Username	If the network operator requires APN authentication, input here Username	*null*
APN Password	If the network operator requires APN authentication, input Password here	*null*
APN Authentication	Defines whether the Network operator requires APN authentication or not. Possible values: NONE/ PAP/ CHAP/ AUTO. Automatic setting is recommended	AUTO
PIN Enable	Enables or disables PIN checking	Unchecked
PIN	This field is disabled if "PIN Enable" field is unchecked. Input here the SIM card PIN	*null*
PIN Hide	This field is disabled if "PIN Enable" field is unchecked. When checked, it hides the characters of the PIN field	Unchecked
Reconnect on failure	Allows the device to re-attempt connection to 3G/4G if previously failed	Unchecked

5.6 Serial

SE59XX supports serial communication with COM port(s). Note that SE59XX series can have up to 16 COM ports: **COM1**, **COM2**, **COM3**, and **COM4**, while typical SE5901 model will have only one COM port (**COM1**). Figure 5.9 shows the **Serial** menu on the left frame of the web interface of SE59XX. The following subsections will describe how to configure these COM ports.

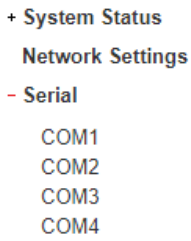


Figure 5.9 Serial Menu (example on SE5904D)

5.6.1 COM Configuration

Figure 5.10 excerpts the **Serial Settings** part of **COM** port settings of SE59XX. Note that these settings need to match the parameters on the serial port of the serial device. Each option is described as follows.

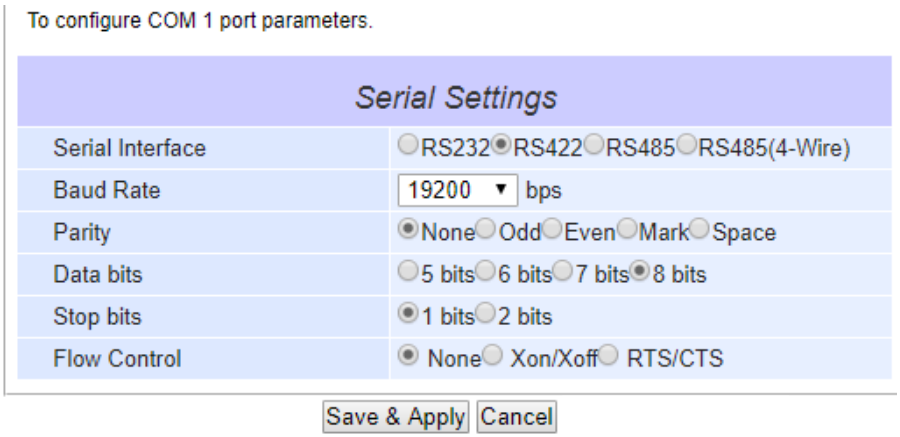


Figure 5.10 Serial Settings Part of COM 1 Port

- **Serial Interface**: This option allows selection between **RS-232**, **RS-422**, **RS-485**, and **RS-485 (4-Wire)** standards. **Note:**
  - RS-485 refers to 2-Wire RS-485 and RS-422 is compatible with 4-Wire RS-485.
  - SE5901B and MB5901B models do not support RS-422 and RS-485 (4-Wire). **Error! Reference source not found.** illustrates an example of Serial Settings of COM 1 port for SE5901B model that only support RS-232 and RS-485.
- **Baud Rate**: The user can select one of the baud rates (from 1200 to 921600 bps) from the drop-down list.
- **Parity**: The available Parity options are **None**, **Odd**, **Even**, **Mark**, or **Space**.
- **Data Bits**: The setting for Data Bits can be **5 bits**, **6 bits**, **7 bits**, or **8 bits**.
- **Stop Bits**: The number of Stop Bits can be either **1 bit** or **2 bits**.
- **Flow Control**: The user can choose among **None** (No Flow Control), **RTS/CTS** (Hardware Flow Control), or **Xon/Xoff** (Software Flow Control). If Xon/Xoff is selected, the Xon and Xoff characters are changeable. Defaults

are 0x11 for Xon and 0x13 for Xoff. Note that these are hexadecimal number of ASCII characters (i.e., 0x11 = '1' and 0x13 = '3').

After finish configuring the COM Port **Serial Settings**, please click on **Save & Apply** button to keep the change that you have made. Note that after click **Save & Apply**, the web browser will be refreshed and remain on the **Serial Settings** page. If you want to cancel the change and reset all changes back to their original values, just click the **Cancel** button. The **Advanced Settings** button will be described in the next subsection.

5.7 Node-RED settings

It is possible to define basic Node-RED settings from ATOP SE59XX-NR Web-UI. The Node-RED settings Menu contains 5 different elements, and is shown in below Figure 5.11. Node-RED settings allow to set enable/disable password checking for Node-RED editor and Node-RED dashboard, change the Node-RED access port, change the access credentials, Restart Node-RED application and check the currently version installed for Node.JS, Node-RED and npm.

Note: all changes carried out in Node-RED environment require a Node-RED restart. Once changes are made and saved, restart Node-RED application by using the "Restart" option on the menu. Being a resource-intensive software, the whole restart operation will take from 60 to 90 seconds to complete.

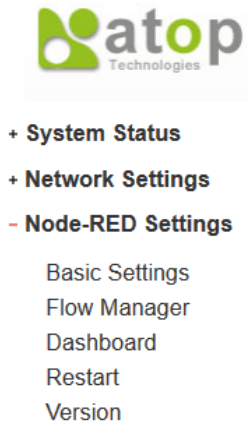


Figure 5.11 Node-RED Settings Menu

5.7.1 Node-RED Settings: Basic settings

Figure 5.12 below shows the Node-RED Basic settings user interface. It will give the user the possibility to:

- Have an insight whether Node-RED is currently up and running on the device
- Modify the Flow-editor and Dashboard access port
- Enable or disable Node-RED auto-start after device boot-up
- Enable or disable Node-RED flow-editor credential check
- Enable or disable Node-RED dashboard credential check

Credential management is useful if inside the organization there are several layers of users: some user, even though in the need to access the dashboard for monitoring purposes should not be allowed to modify the application lying underneath, and sometimes developers should not be allowed to visualize the data.

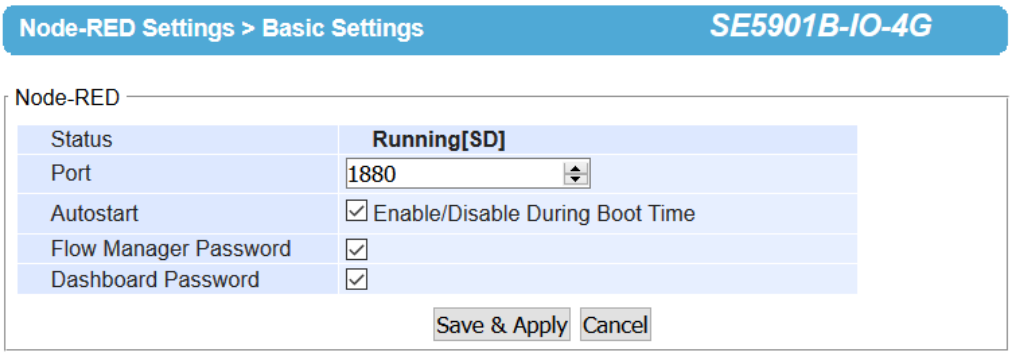


Figure 5.12 Node-RED Basic Settings

The meaning of the different fields is explained in below Table 5.3. When changes are complete, click on “Save & Apply” button.

Note: the changes are effective only after Node-RED restart.

Table 5.3 Description of Node-RED basic settings fields

Field Name	Description	Factory Default
Status	Shows the current Node-RED application status. This can be “Not Running”, “Running” and “Restarting”. The value in the square brackets [ ] shows the media support on which the Node-RED application is stored	-
Port	Shows the port where Node-RED application is available for users and developers. This can be freely modified, and allows the user to define freely the path, such as: http://10.0.0.100:1880	1880
Autostart	Allows the device to automatically run Node-RED after boot up	*Checked*
Flow Manager PW	Enables username/password checking for Flow-editor	*Checked*
Dashboard PW	Enables username/password checking for Dashboard	*Checked*

5.7.2 Node-RED Settings: Flow manager

This section enables the user to change the access credentials for Node-RED flow manager, that by default is reachable at the address http://10.0.50.100:1880/
The default access credentials are:

- Username: admin
- Password: default

When changes are complete, click on “**Change Password**” button.

Note: the changes are effective only after Node-RED restart.

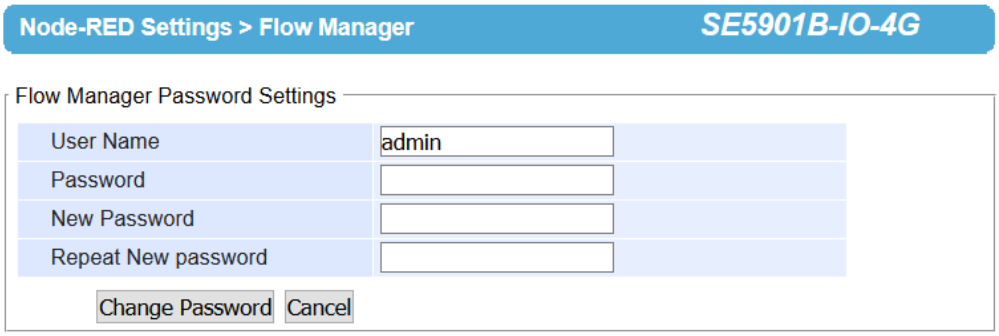


Figure 5.13 Node-RED Flow Manager Settings

5.7.3 Node-RED Settings: Dashboard

This section enables the user to change the access credentials for Node-RED dashboard, that by default is reachable at the address http://10.0.50.100:1880/ui
The default access credentials are:

- Username: admin
- Password: default

When changes are complete, click on “**Change Password**” button.

Note: the changes are effective only after Node-RED restart.

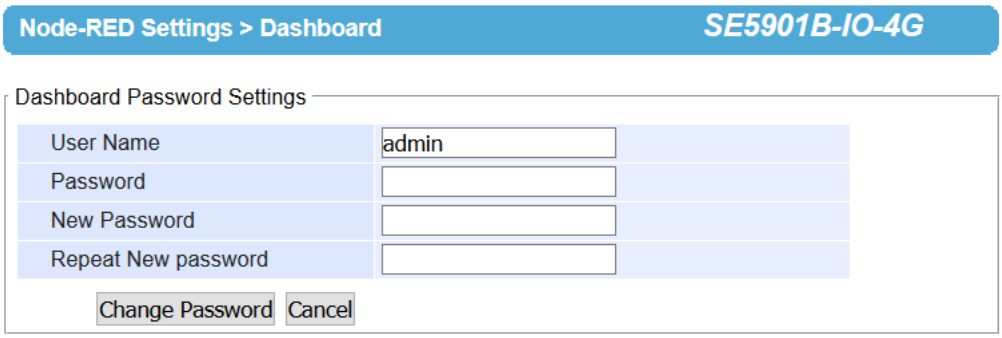


Figure 5.14 Node-RED Dashboard Settings

5.7.4 Node-RED Settings: Restart

All changes carried out in Node-RED environment require a Node-RED restart. Once changes are made and saved, move to the Restart tab on the menu on the left hand side and restart Node-RED application by clicking on the "Restart" button. Being a resource-intensive software, the whole restart operation will take from 60 to 90 seconds to complete.

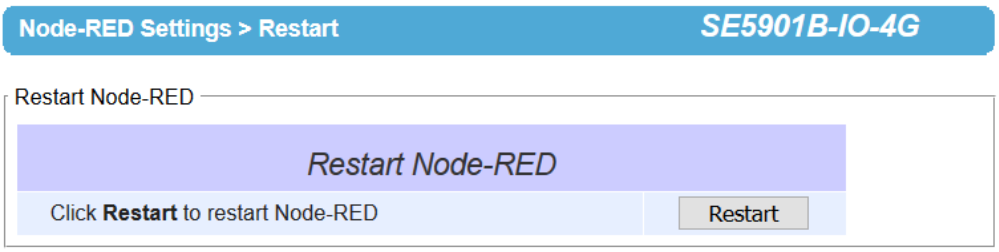


Figure 5.15 Node-RED Restart

5.7.5 Node-RED Settings: Version

This option allows the user to double-check which version of Node.JS, Node-RED and npm are installed on the device.

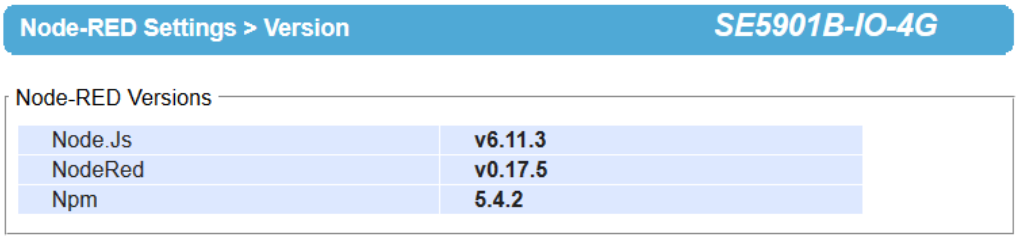


Figure 5.16 Node-RED Version

5.8 VPN

A virtual private network(VPN) extends a private network across a public network, such as the Internet. It enables a computer to send and receive data across shared or public networks as if it were directly connected to the private networks, while benefitting from the functionality, security and management policies of the private network. This is done by establishing a virtual point-to-point connection through the use of dedicated connections, encryption, or a combination of the two.

See below VPN scenario of SE/PG/MB59XX for your reference.

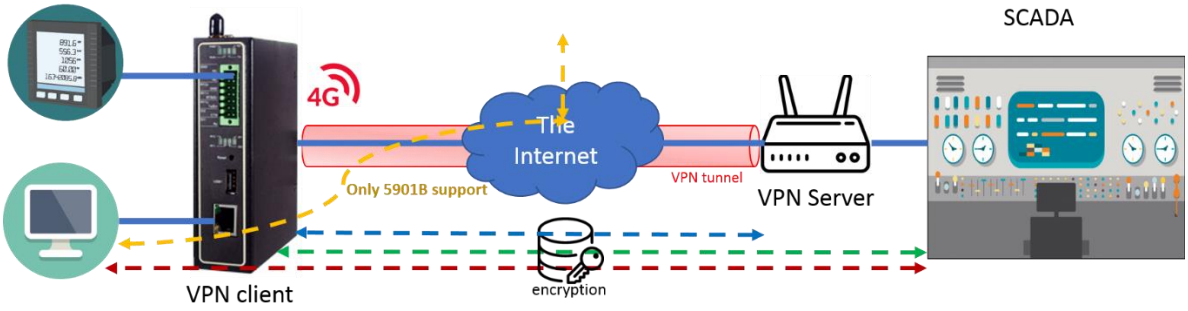


Figure 5.17 VPN Scenario of SE/PG/MB59XX

SE59XX supports several VPN protocols: PPTP (Point-to-Point-Tunneling-Protocol), IPsec (Internet Protocol Security), and OpenVPN. In order to configure VPN, please click on the related item in the dedicated VPN sub-menu on the left-hand side of the screen, as shown in Figure 5.18 below.

A better description of PPTP is available in Chapter 0 below.
A better description of OpenVPN is available in Chapter 5.10 below.
A better description of IPsec related settings is available in Chapter 0 below.

- **VPN**
  - PPTP
  - PPTP Status
  - IPSec Settings
  - IPSec Status
  - OpenVPN Settings
  - OpenVPN Keys
  - OpenVPN Status

Figure 5.18 VPN menu structure

5.9 PPTP Settings

PPTP (Point-to-Point Tunneling Protocol) is a method for implementing virtual private networks. PPTP uses a TCP control channel and a GRE tunnel to encapsulate PPP packets. Select the PPTP item in the menu to configure a PPTP tunnel. Figure 5.19 shows the PPTP configuration page under PPTP web setting. Currently SE59xx series only supports PPTP client. After settings are completed, click "**Save**" to save the configuration.

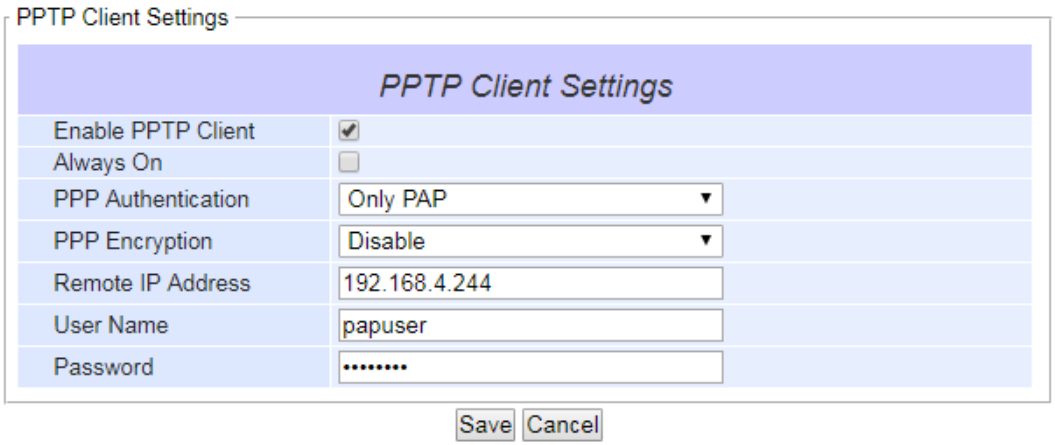


Figure 5.19 PPTP configuration page.

- Enable PPTP client: Check this to enable the PPTP client on SE59XX series.
- Always on: Check this to have SE59xx to automatically reconnect in event of disconnection.
- PPP Authentication: Specify the authentication algorithm – should be same as server
- PPP Encryption: Specify the encryption – should be same as server
- Remote IP address: Specify the IP address of PPTP server.
- User Name: Specify the User name for authentication.
- Password: Specify Password for authentication.

Figure 5.20 below shows the PPTP Link status.

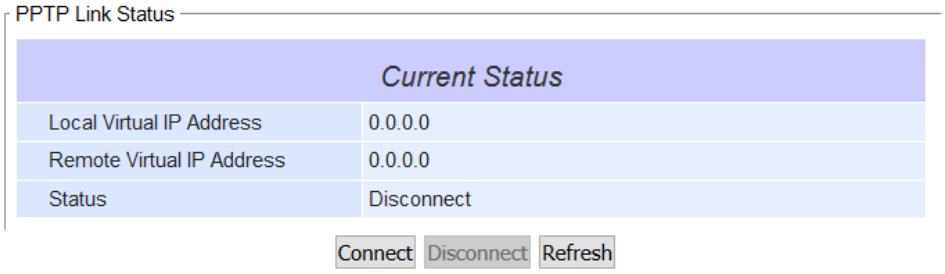


Figure 5.20 PPTP Link Status

- Local Virtual IP Address: The virtual IP address assigned by PPTP server.
- Remote Virtual IP Address: The virtual IP address of PPTP server.
- Status: It shows the PPTP tunnel connection status. It will show Disconnect, Connect and Connecting.
- Disconnect: No tunnel is established.
- Connect: PPTP Tunnel is established.
- Connecting: PPTP Tunnel is establishing.
- Connect: Click this button to connect to PPTP server.
- Disconnect: Click this button to disconnect PPTP tunnel.
- Refresh: Clieck this button to refresh the PPTP tunnel status.

5.10 OpenVPN Settings

OpenVPN is an application that implements VPN for creating secure point-to-point or site-to-site connections in routed or burdged configurations and remote access facilities. It uses a custom security protocol that utilizes SSL/TLS for key exchange.
OpenVPN allows peers to authenticate each other using a Static Key (pre-shared key) or certificates. When used in a multi-client-server configuration, it allows the server to release an authentication certificate for every client, using signature and certificate authority.
There are two OpenVPN connection scenarios. They are the TAP and TUN scenario. The product can create ether a layer-3 based IP tunnel(TUN), or a layer-2 based Ethernet TAP that can carry any type of Ethernet traffic. In addition to configuring the device as a Server or Client, you have to specify which type of OpenPVN connection scenario is to be adopted. Currently SE59xx series only support TUN mode.

5.10.1 OpenVPN Setting

In order to configure OpenVPN, click on the VPN tab in the left hand side of the menu and then **OpenVPN Settings**. The user interface is shown in below Figure 5.21.

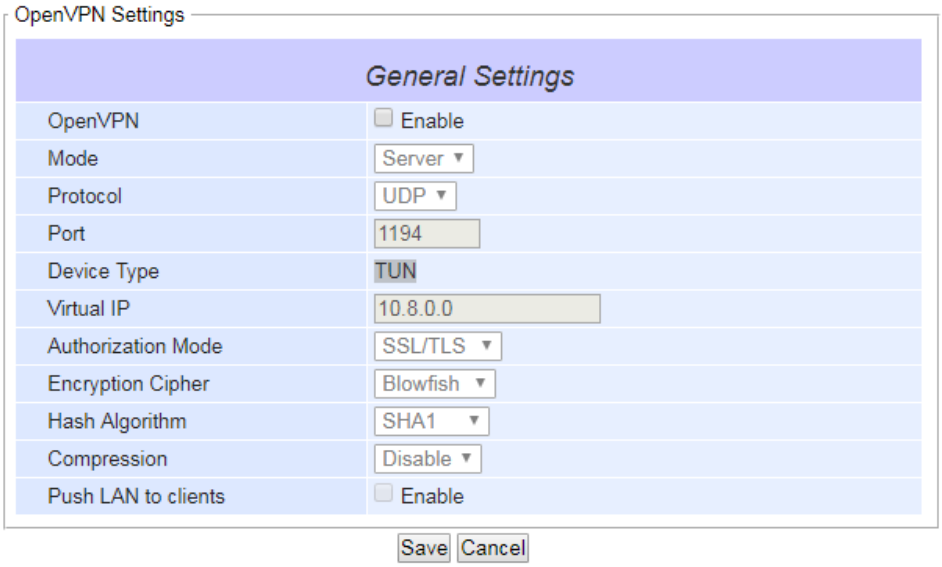


Figure 5.21 OpenVPN Setting

The OpenVPN parameters are described as below:

- **OpenVPN**: Check this to enable OpenVPN.
- **Mode**: Specifies what the scenario of this device, server or client. When choosing server mode, the device will play as server role and will standby for client connection.
- **Protocol**: Selects the transport layer protocol to be used for VPN (TCP or UDP).
- **Port**: Defines the port number for TCP/UDP connection.
- **Device Type**: OpenVPN tunnel connection by TUN (Tunnel) mode or TAP mode. Currently SE59xx series only supports TUN (Tunnel) mode.
- **Virtual IP** (only when “OpenVPN Server” mode is selected): Specify the server’s virtual IP. Virtual IP will only be available when SSL/TLS is chosen as the Authentication Mode. The Server’s virtual IP address will be 10.8.0.1/24 and client virtual IP address will be 10.8.0.x/24.
- **Local/Remote endpoint IP** (only when “OpenVPN Client” mode is selected): Specifies the local and remote endpoint virtual IP address of this OpenPVN gateway. Local/Remote endpoint IP only be available when static key is chosen in Authentication Mode.

- **Authentication Mode**: Specify the authorization mode the OpenVPN server. There are 2 options available:
  - SSL/TLS: OpenVPN will use TLS authorization mode, and the following items CA cert, Server Cert and DH PEM will be used. See section 5.10.2 below for mode details.
  - Static Key: OpenVPN will use static key authorization, and the static key will be used. See section 5.10.2 below for mode details.
- **Encryption Cipher**: Specify the Encryption cipher. There are 5 options available: blowfish, AES 256, AES 192, AES 128 and Disable. When Disable is selected, no encryption will be used.
- **Hash Algorithm**: Specify the Hash algorithm. There are 5 options available: SHA1, MD5, SHA 256, SHA 512 and Disable.When Disable is selected, no Hash algorithm will be used.
- **Compression**: Specify whether or not the tunnel packets will be compressed. There are three options available: LZ4, LZO and Disable. When Disable is chosen, the packet won't be compressed.
- **Push Lan to clients** (only when "OpenVPN Server" mode is selected): When enabled, SE59xx will push the LAN port subnet to the OpenVPN remote clients, so that the remote client will add a route to the SE59XX local network. Only SE5901B supports this function.

5.10.2 OpenVPN Keys

OpenVPN requires encryption keys (unless Encryption Cipher is disabled). In order to key-in, import or generate encryption keys, please select "OpenVPN Keys" from the VPN menu on the left-hand side of the user interface.

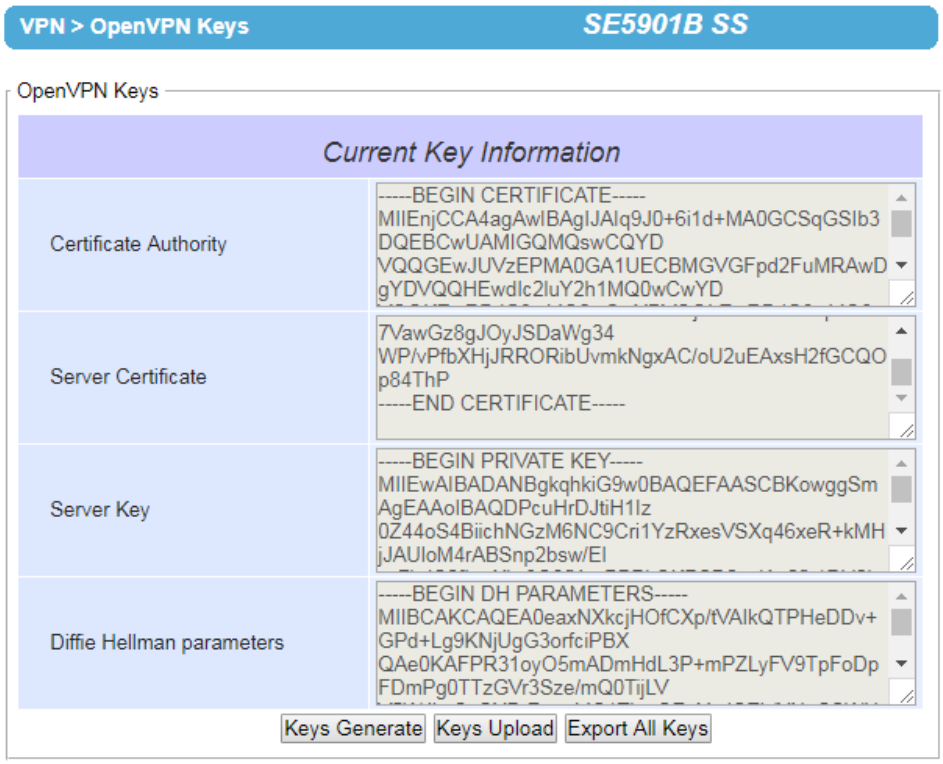


Figure 5.22 OpenVPN Keys

- **Certificate Authority**: A certificate authority(CA) is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate. A CA acts as a trusted third party, trusted both by the owner and by the party relying upon the certificate.
- **Server Certificate**: It shows the information of server certificate. You can check the information if you use upload server certificate file.
- **Server Key**: It shows the information of server key. You can check the information if you use upload server key file.
- **Diffie Hellman parameters**: It shows the information of Diffie Hellman paramaters.

When SE59XX acts as OpenVPN server, the user could define his own certification information by clicking on the **Key generate** button. Otherwise, the certificate can be imported. When generating a new key, a Pop-up window will open. Fill in the parameters and click on "**Generation Keys & Apply**" button.

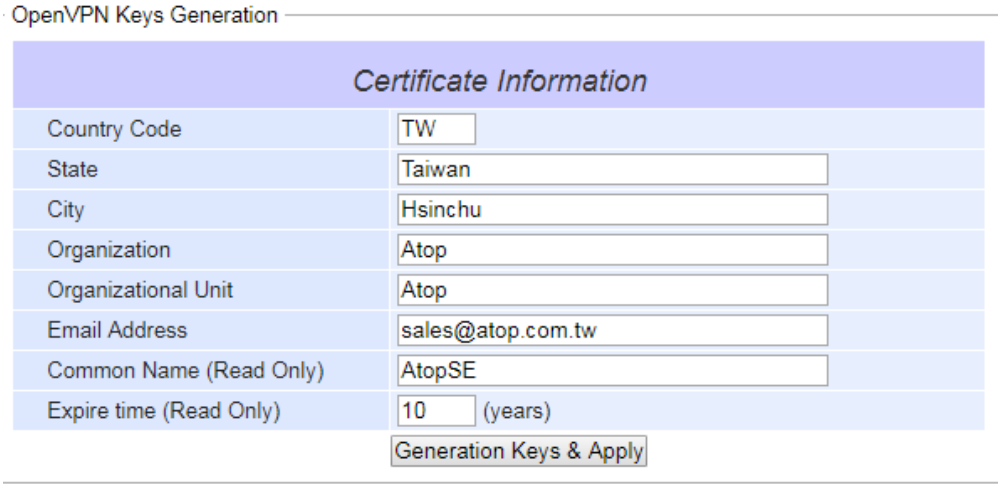


Figure 5.23 Certification information

- **Country Code**: Enter the country ISO code.
- **State**: Enter the state (if applicable)
- **City**: Enter the city
- **Organization**: Enter the name of organization.
- **Organization Unit**: Enter the unit or section in the organization.
- **Email Address**: Enter an email address.
- **Common Name**: The server name. (Read only)
- **Expire time**: The number of years the certificate is valid for. (Read only)

When clicking on the **Keys Upload** button instead, a pop-up window shown in Figure 5.24 will show up and will allow you to import the related server or client certificates.

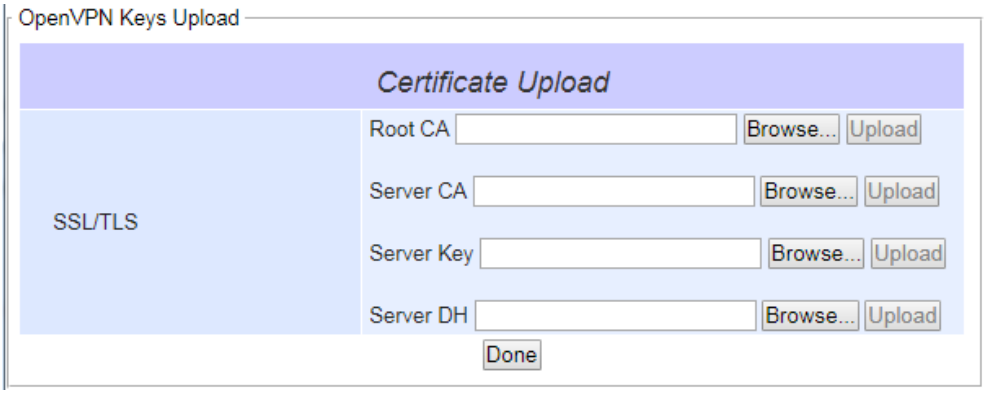


Figure 5.24 Certificate Upload

Click the **Browse** button to select your own server or client certificate and click on the **Upload** button. When SE59xx acts as an OpenVPN server, use **Export All Keys** button to download all the necessary certificates include CA.crt, CA.key and the certificate and key for client side.

5.10.3 OpenVPN Status

In order to check the current OpenVPN connection status, click “OpenVPN status” in the VPN menu on the left-hand side of the screen. A page like below Figure 5.25 or Figure 5.26 will show up depending whether OpenVPN is set as Client or Server.

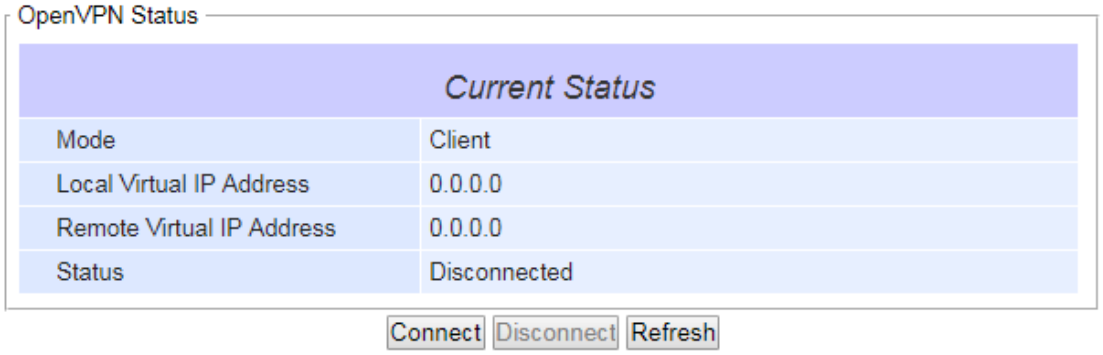


Figure 5.25 OpenVPN client status

- **Mode**: Displays the OpenVPN mode SE59xx is currently running as.
- **Local Virtual IP address**: Displays the Local virtual IP address.
- **Remote Virtual Status**: Displays the Remote virtual IP address.
- **Status**: Displays the current status of OpvnVPN connection. It will include Disconnected, Connecting and Connected.

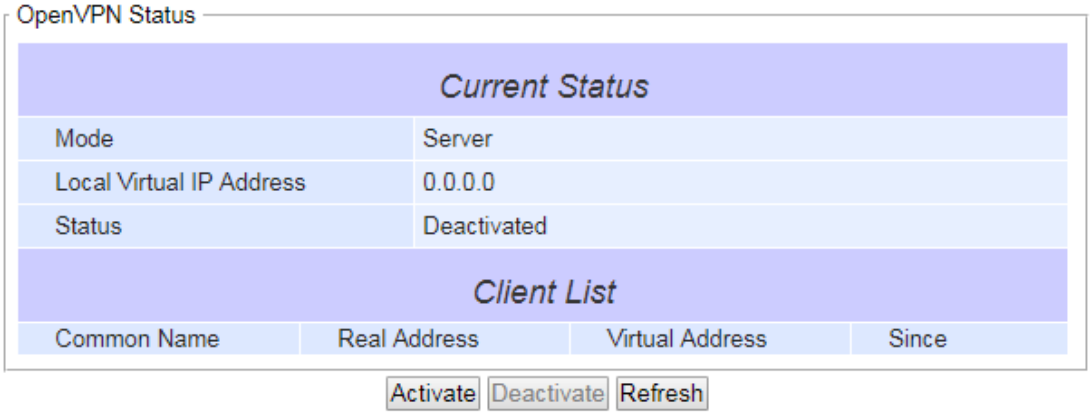


Figure 5.26 OpenVPN server status

- **Mode**: Displays the OpenVPN mode SE59xx is currently running as.
- **Local Virtual IP address**: Displays the Local virtual IP address.
- **Status**: Displays the current status of OpvnVPN connection. It will be either be Deactivated, Activating, Disconnected, Connecting and Connected.

5.11 IPsec Settings

IPsec (or Internet Protocol Security) which is a network protocol suit that can establish secure and reliable communications for different application scenarios. IPsec enables data confidentiality, data integrity, data origin authentication, and antireplay. For example, a corporate headquarter and its branch offices in the fields do not need to apply for dedicated communication lines for sharing their network resources securely. To securely communicate and shared company's resources over the Internet, IPsec connections can be employed to secure all applications at the IP layer. In another case, when employees are on a business trip, they can establish IPsec connections with their company over their mobile devices or the public network to access the internal network resources in their company.

SE59XX has an IPsec connection function to establish a secure communication link between **host-to-host**, **host-to-subnet** (or host-to-network), and **subnet-to-subnet** (or network-to-network). Note that at the other endpoint of the Internet, a router or gateway with full IPsec capability is required to successfully establish the secure communication. There are two types of IPsec connection modes or types supported by SE59XX which are **Tunnel mode** and **Transport mode.**

- In **Tunnel mode**, the entire IP packet is encrypted and authenticated. The IP packet is then encapsulated into a new IP packet with a new IP header. The **Tunnel mode** which is used to create Virtual Private Network (VPN) can be applied to the **host-to-host**, the **host-to-subnet**, and the **subnet-to-subnet** communications. The packet (datagram) format for **Tunnel mode** is as follow:

New IP Header	IPsec Header	Original IP Packet	Optional IPsec Trailer

- In **Transport mode**, only the payload of the IP packet is usually encrypted or authenticated. The routing is intact because the IP header is not modified and not encrypted. However, when the authentication header is used, the IP addresses cannot be modified by Network Address Translation (NAT). The **Transport mode** can only be applied in the **host-to-host** communication. The packet (datagram) format for **Transport mode** is as follow:

Original IP Header	IPsec Header	Original IP Packet	Optional IPsec Trailer

A **host-to-host** connection is typically used in a simple point-to-point communication. It is useful for a direct communication with a server or between the device (SE59XX) and a peer device (such as another SE59XX). Note that this type of connection cannot be use for accessing entire sub-network resources. Figure 5.27 illustrates an example of host-to-host connection. This configuration can be set in both **Tunnel mode** and **Transport mode.**

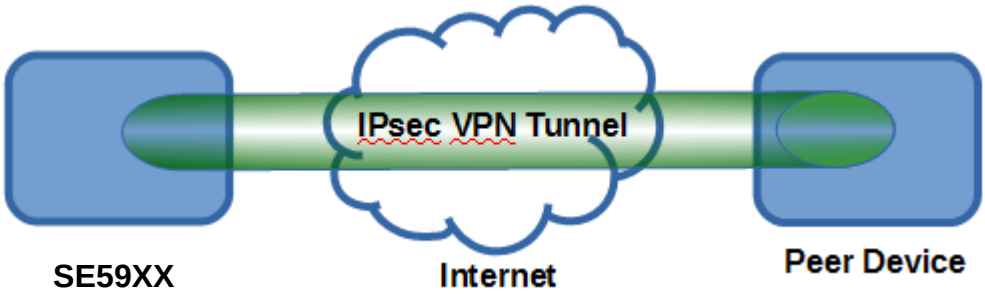


Figure 5.27 An example of Host-to-Host Connection

A **host-to-subnet** (or host-to-network) connection is mainly applied when one endpoint needs to access the other side's sub-networks. Typical applications are employees who are travelling on business and would like to connect back to their corporate headquarters via mobile devices. They can establish IPsec connections to access the internal corporate network resources. Figure 5.28 illustrates a road-warrior application in which SE59XX can access

a remote sub-network resource via a peer gateway. Figure 5.29 illustrates a gateway application in which SE59XX can passively accept connection requests from remote sides and provide access to the SE59XX sub-network resources. Note that both of these configurations must set the connection type to **Tunnel mode** only.

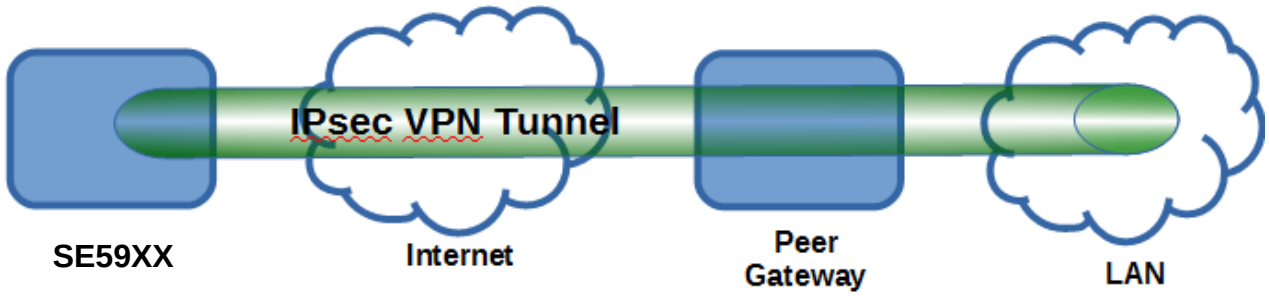


Figure 5.28 Roadwarrior Application using Host-to-Subnet Connection

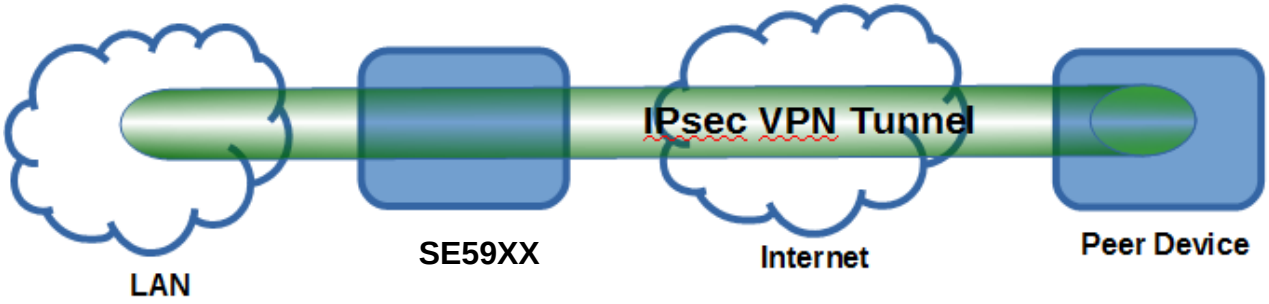


Figure 5.29 Gateway Application using Host-to-Subnet Connection

A **subnet-to-subnet** connection is mainly used to connect two subnets from different sides together. Typically, applications are corporate headquarters and branch offices that share internal network resources. A specific application can also be set up as one side's subnet to establish IPsec VPN tunnels for accessing a device in the other side's subnet or as a device in one's side subnet to establish IPsec VPN tunnels for accessing another device in the other side's subnet.

Figure 5.30 illustrates an example of the subnet-to-subnet connection with a network application. A host inside the remote subnet can also connect to a local subnet (host-network application) based on this subnet-to-subnet connection as shown in Figure 5.31. On the other hand, two different devices on two different subnets (host-host application) can be connected via a IPsec VPN tunnel based on this subnet-to-subnet connection as shown in Figure 5.32. Note that all subnet-to-subnet configurations must set the connection type to **Tunnel mode** only.

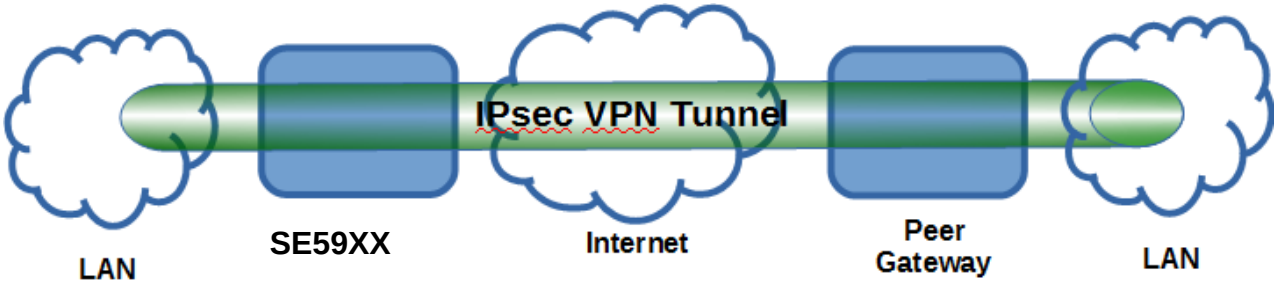


Figure 5.30 An example of network application using a subnet-to-subnet connection via the SE59XX and a peer device

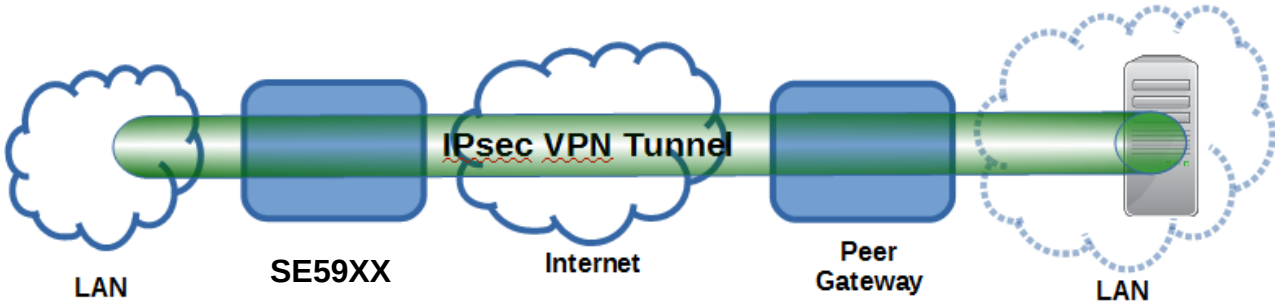


Figure 5.31 An example of host-network application via the subnet-to-subnet connection

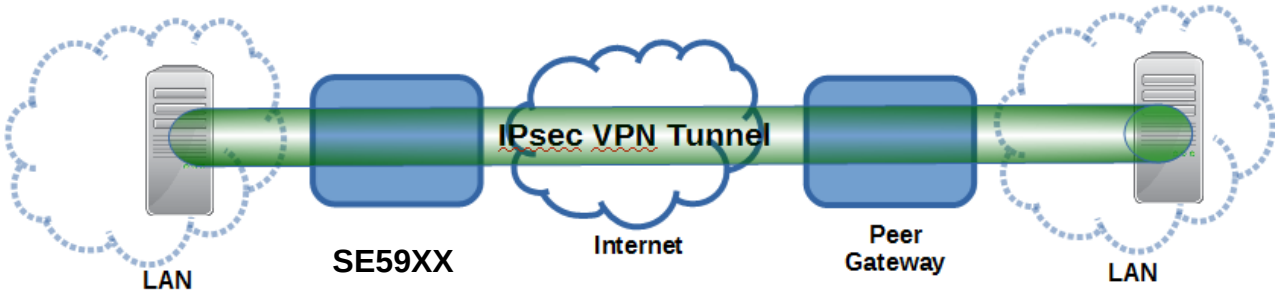


Figure 5.32 An example of host-host application via the subnet-to-subnet connection

In some network configuration, there is an implementation of network address translation (NAT) on its gateway/routers. NAT is typically used to allow private IP addresses on private networks behind gateways/routers with a single public IP address connecting to the public Internet. The internal network devices can communication with hosts on the external network by changing the source address of outgoing requests to that of the NAT device (gateway/router) and relaying replies back to the originating device. IPsec virtual private network (VPN) clients use network address translation (NAT) traversal in order to have Encapsulating Security Payload (ESP) packets traverse NAT. IPsec uses several protocols in its operation, which must be enabled to traverse firewalls and network address translators (NATs), such as

- Internet Key Exchange (IKE) protocol uses User Datagram Protocol (UDP) port number 500.
- Encapsulating Security Payload (ESP) uses IP protocol number 50.
- Authentication Header (AH) uses IP protocol number 51.
- IPsec NAT traversal uses UDP port number 4500 when NAT traversal is in use.

SE59XX also has a feature called NAT traversal (NAT-T) that allows the IPsec tunnel to pass through the NAT in its network. SE59XX will activate this option automatically and encapsulate the IPsec packets inside UDP port 4500 to be able to pass through a NAT router.

To provide security service for all types of tunnel connections and applications described above, SE59XX utilizes the Internet Key Exchange (IKE) protocol to set up a security association (SA) in the IPsec protocol suite. Note that IKE builds upon the Oakley protocol and ISAKMP (Internet Security Association and Key Management Protocol). IKE uses X.509 certificates for authentication either pre-shared or distributed using DNS (preferably with DNSSEC). IKE also uses a Diffie-Hellman key (DH) key exchange to set up a shared session secret from which cryptographic keys are derived. The IPsec security associations (SA) is divided into two phases. In phase one, IKE creates an authenticated secure channel between SE59XX and its peer device, which is called the IKE Security Association. The Diffie-Hellman (DH) key agreement is always performed in this phase to create a shared secret key or DH key. In phase two, IKE negotiates the IPsec security associations and generates the required key material for IPsec. This IPsec key which is a symmetrical key will be used for bulk data transfer inside the IPsec tunnel. A new Diffie-Hellman agreement can be done in phase two, or the keys can be derived from the phase one shared secret.

5.11.1 IPsec Settings

Figure 5.33 shows the **IPsec Settings** web page under the **IPsec Settings** menu. There are four sections on this page: **General Settings**, **Authentication Settings**, **IKE Settings**, and **Dead Peer Detection Settings**.

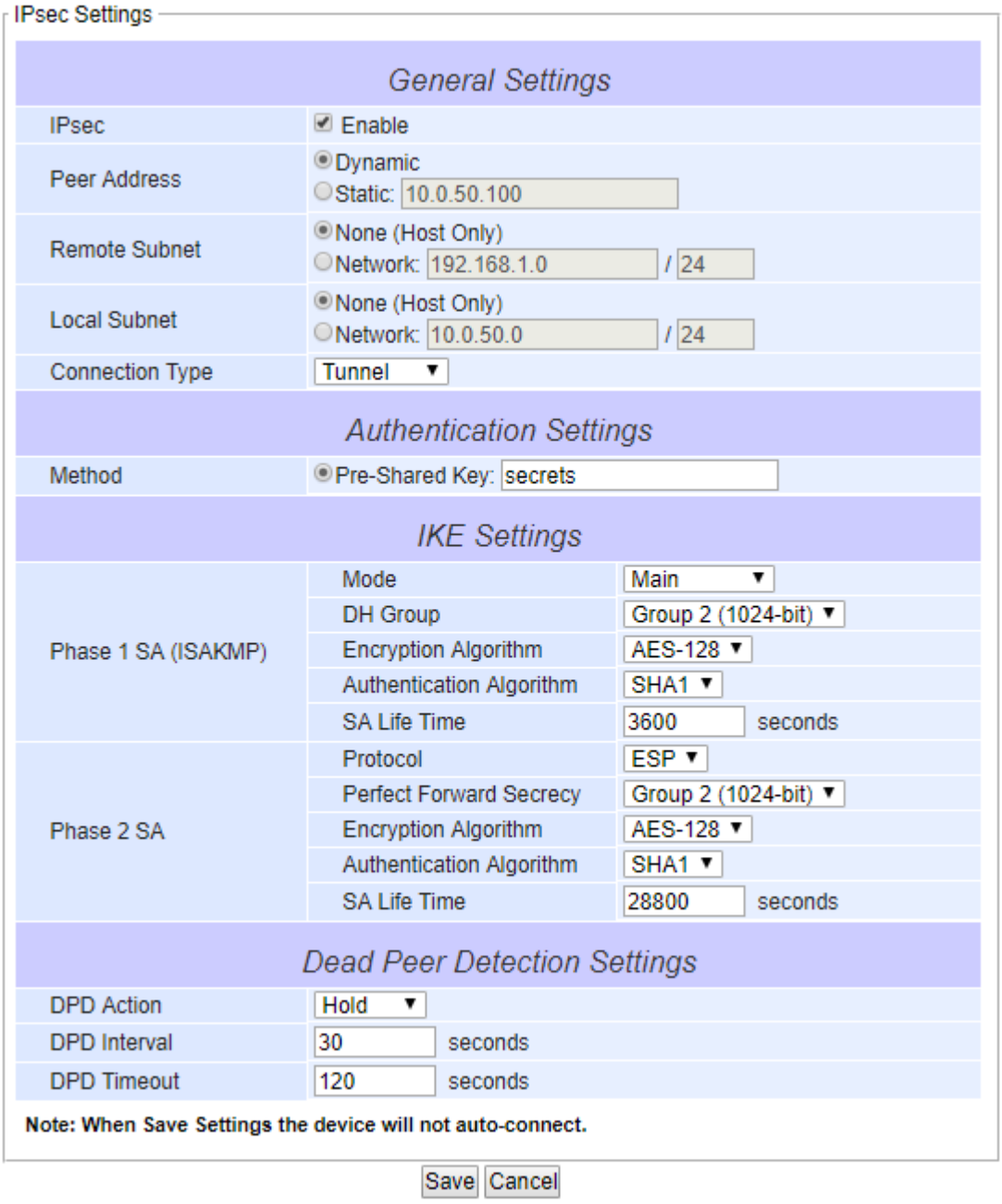


Figure 5.33 IPsec Tunnels Web Page under IPsec Setting Menu

To configure **IPsec Settings**, first you need to configure the **General Settings** section under the **IPsec Settings** menu. Under the **General Settings**, there are five parameters that need to be set as follows:

- **IPsec**: By checking the box for this option, you enable the IPsec feature for SE59XX.
- **Peer Address:** This option is to specify the IP address of a remote host or peer host or remote gateway. There are two choices for the **Peer Address** which are **Dynamic** and **Statics**.

- **Dynamic**: When you selected the **Dynamic** by choosing the **Dynamic** radio button, the **Peer Address** or the remote device IP address is not fixed or unknown. Note that when **Peer Address** is set to dynamic mode, the SE59XX can accept remote connection request or will be the responder.
- **Static**: On the other hand, if you know the IP address of the remote device, you can choose the ratio button for **Static** option and enter the IP address in the text box behind it. The SE59XX will be the initiator/responder.

- **Remote Subnet**: This option is to indicate whether you want to create an IPsec connection to the remote subnetwork. There are also two choices for **Remote Subnet** access:
  - **None**)**Host Only**(: This option is to specify that the remote subnet is not supported or no remote subnet and only host access is supported. That is the remote end of the IPsec tunnel is a host or peer device only.
  - **Network**: This option is to specify the **Remote Subnet** by entering the **Subnet IP Address** and the number of **Subnet Masking Bits** or associated routing prefix. This option supports the Classless Inter-Domain Routing (CIDR) notation. For example, Subnet IP Address is 192.168.11.0 and Subnet mask are 24 bits (from 255.255.255.0).
- **Local Subnet:** This option is to enable an IPsec connection to the local subnetwork. There are two choices for **Local Subnet** access:
  - **None (Host Only):** This option is to specify that the local subnet is not supported or no local subnet and only local host access is supported. That is the local end of the IPsec tunnel is a host or peer device only.
  - **Network:** This option is to specify the **Local Subnet** by entering the **Subnet IP Address** and the number of **Subnet Masking Bits** or associated routing prefix. This option supports the Classless Inter-Domain Routing (CIDR) notation. For example, Subnet IP Address is 192.168.11.0 and Subnet mask are 24 bits (from 255.255.255.0).
- **Connection Type**: This option is to specify the IPsec connection type which can be either **Tunnel** mode or **Transport** mode. Please select the corresponding connection type from the drop-down list. Note that the **Tunnel mode** can be applied to the **host-to-host**, the **host-to-subnet**, and the **subnet-to-subnet** communications. The **Transport mode** can only be applied in the **host-to-host** communication.

The second part of **IPsec Settings** is the **Authentication Settings.** Here you have an authentication's **Method** which already selected as the **Pre-Shared Key.** Then, you must enter in a secret key or a pass-phrase in the textbox behind it**.** Both ends of the the VPN tunnel must use the same secret key or password. The pre-shared key can be 1 to 60 case-sensitive ASCII characters and special symbols.

The third part of **IPsec Settings** is the **IKE** (Internet Key Exchange) **Settings.** Internet Key Exchange (IKE) that SE59XX supports is the IKE version 1 or **IKEv1.** Within the **Phase 1 SA (ISAKMP)**, there are five security options to be configured**.** In phase 1, the two VPN gateway exchange information about the encryption algorithms that they support and then establish a temporary secure connection to exchange authentication information.

- First option is the **Mode** of IKE session which defines how many steps or packets will be used or exchanged during the IKE SA negotiation**.** You can choose either **Main Mode** or **Aggressive Mode.** The **Main Mode** will send SA proposals, Diffie-Hellman public key, and ISAKMP session authentication in three exchange packets, while the **Aggressive Mode** will put all SA proposals, DH public key, and ISAKMP session authentication in to one exchange packet**. Aggressive Mode** makes the IKE negotiation quicker than **Main Mode.** The difference between **Main Mode** and **Aggressive Mode** is that the "identity protection" is used in the **Main Mode**. The identity is transferred encrypted in the **Main Mode** but it is not encrypted in **Aggressive Mode**. Typically, the **Main Mode** is recommended.
- Second option is the selection of Diffie-Hellman's group (**DH Group**) of standardized global unique prime numbers and generators that will be used to provide secure asymmetric key exchange**.** The **DH Group** is

used to encrypt this IKE communication. SE59XX supports two **DH groups** which are **DH Group 2**, which is a 1024-bit modular exponentiation group (MODP), and **DH Group 5**, which is a 1536-bit MODP group**.**

- Third option is the selection of **Encryption Algorithm** which can be either **AES-128** or **3DES.** This option will select the key size and encryption algorithm to be used in the IKEv1 Phase 1. The default value is **AES-128**.
- Fourth option is the selection of **Authentication Algorithm** which can be either **SHA1** or **MD5.** This option will select which hash algorithm will be used to authenticate packet data in the IKEv1 Phase 1. The default value is **SHA1**.
- Fifth option is the **SA Life Time** which must be set in unit of seconds**.** This value represents the lifetime of the IKE key which is dedicated at Phase 1 between both end host or network. The default **SA Life Time** is 10800 seconds**.** The configurable range for **SA Life Time** is between 300 to 86400 seconds.

Within the **Phase 2 SA**, there are five security options to be configured**.** Similar to **Phase 1 SA**, SE59XX and its peer device will negotiate or exchange proposals to determine which security parameters will be used in this Phase 2 SA. A Phase 2 proposal also includes a security **Protocol** (first option), which you can choose either Encapsulating Security Payload (**ESP**) or Authentication Header (**AH**). The second option is the **Perfect Forward Secrecy** which is a property of key-agreement protocol to ensure that a session key derived from a set of long-term keys cannot be compromised if one of the long-term keys is compromised in the future. In Phase 2 SA, SE59XX also supports two **DH groups** which are **DH Group 2** (1024-bit) and **DH Group 5** (1536-bit).

Then you can proceed to select encryption and authentication algorithms. Third option is the selection of **Encryption Algorithm** which can be either **AES-128** or **3DES.** This encryption algorithm will be used in the IPsec tunnel. The default setting is the **AES128**. Fourth option is the selection of **Authentication Algorithm** which can be either **SHA1** or **MD5.** This is the hash algorithm that will be used to authenticate packet data in the IPsec tunnel. The default selection is the **SHA1**. Finally, the last option is the **SA Life Time** for phase 2 which must be set in unit of seconds**.** The range of this setting can be from 180 to 86400 seconds. The default **SA Life Time** is 3,600 seconds**.**

The final part of the **IPsec Settings** is the **Dead Peer Detection Settings.** Dead peer detection (DPD) is a mechanism that SE59XX use to verify the existence of a remote Internet Key Exchange (IKE) gateway or the peer device of SE59XX**.** To detect the peer device, SE59XX will sent encrypted IKE Phase 1 notification payloads (or hello message) to its peer device and wait for DPD acknowledgement from the peer device**.** If SE59XX does not receive an acknowledge message during a specific time interval (**DPD timeout**), it will consider that the peer device is dead**.** Then, SE59XX will remove the Phase 1 Security Association and all Phase 2 Security Association of that dead peer device**.** Under the **Dead Peer Detection Settings**, you will have to choose the **DPD Action** that the SE59XX will perform if it found that the peer device is dead**.** You can choose either **Hold** to still hold the security association for the peer device and wait for the peer device to return or **Restart** to restart the security association process again**.** The **DPD Interval** is the period of time for sending the hello message to the peer device or the interval that SE59XX will repeatly check the endpoint with keep-alive message**.** The **DPD interval** can be ranged from 1 to 65535 seconds. The default value for **DPD Interval** is 30 seconds**.** The **DPD Timeout** will be the time that SE59XX declares the peer device dead if it did not receive any reply or traffic from the peer device**.** If the keep-alive check fails before this time period expires, the SE59XX will take the PDP action. The **DPD Timeout** value range from 1 to 65535 seconds. The default value of **DPD Timeout** is 120 seconds**.** Description of each parameters in the IPsec Tunnels web page is summarized in
Table 5.4**.**

Table 5.4 Description of Parameters in IPsec Tunnels Web Page

Field Name		Description	Default Value
General Settings			
IPsec		Enable the IPsec Tunnel	Disable
NAT Traversal		Enable the NAT Traversal mechanism	Enable
Peer Address		IP address of the remote device which can be dynamic (any address) or static (fixed address)	Dynamic

Field Name		Description	Default Value
Remote Subnet		Remote subnet can be either None (Host only) or Network (IP and Netmask)	None (Host Only)
Local Subnet		Local subnet can be either None (Host Only) or Network (IP and Netmask)	None (Host Only)
Connection type		Tunnel mode or Transport mode	Tunnel
Authentication Settings			
Method		Pre-Shared Key	secrets
IKE Settings			
Phase 1 SA	**Mode**	Choose how IKE negotiation is performed between Main Mode and Aggressive Mode	Main Mode
	DH Group	Diffie-Hellman groups, determine the strength of the key used in the key exchange process**:** DH Group 2 (1024-bit) or DH Group 5 (1536-bit)	Group 2 (1024-bit)
	Encryption Algorithm	Encryption algorithm used in the key exchange process**:** Either 3DES or AES	AES128
	Authentication Algorithm	Hash algorithm used to authenticate packet data in the key exchange process of IKEv1 phase 1**:** Either MD5 or SHA1	SHA1
	SA Life Time	How long a particular instance of a connection (a set of encryption/authentication keys for user packets) should last, from successful negotiation to expiry. The value can be from 300 to 86,400 seconds.	3600
Phase 2 SA	**Protocol**	Choose how IP packet will be encrypted and verify: either Encapsulate Security Payload (ESP) or IP Authentication Header (AH)	ESP
	Perfect Forward Secrecy	Diffie-Hellman groups for Perfect Forward Secrecy of keys, determine the strength of the key used in the key exchange process**:** DH Group 2 (1024-bit) or DH Group 5 (1536-bit)	Group 2 (1024-bit)
	Encryption Algorithm	Select which key size and encryption algorithm will be used in IPsec tunnel: either 3DES or AES128	AES128
	Authentication Algorithm	Section of hash algorithm to be used to authenticate packet data in the IPsec tunnel: either MD5 or SHA1	SHA1
	SA Life Time	Value that represents the lifetime of the IKE key which is dedicated in Phase 2 between both end host or network. The available setting ranges is from 180 to 86,400 seconds.	28800
Dead Peer Detection Settings			
DPD Action		Select either Hold or Restart the tunnel's security association for the peer**.** Note that Hold is suitable for a statistically defined tunnel**.**	Hold
DPD Interval		Duration of time for sending hello message to the peer device: value from 1 to 65535 seconds.	30 seconds
DPD Timeout		Duration of time to declare that the peer is dead: value from 1 to 65535 seconds.	120 seconds

After finishing the **IPsec settings** configuration, please click the **Save** button to save all changes that have been made. If you would like to discard any setting, please click the **Cancel** button.

5.11.2 IPsec Status

On this web page, you can check the status of your IPsec connection between SE59XX and its peer device in different connection types and modes. The first information is the **Peer Address** which is the IP address of the other device that is connected to SE59XX. The second information is the **VPN Tunnel**'s status. The third information is the **Status** of the IPsec connection which can be **Disabled**, **Listening**, or **Connected**. shows the **IPsec Status** web page under the **IPsec Settings** menu. There are three buttons at the end of the web page which are **Connect**, **Disconnect**, and **Refresh**. The **Connect** and **Disconnect** buttons allow you to establish or tear down the IPsec connection. The **Refresh** button enable you to check the latest status of the connection.

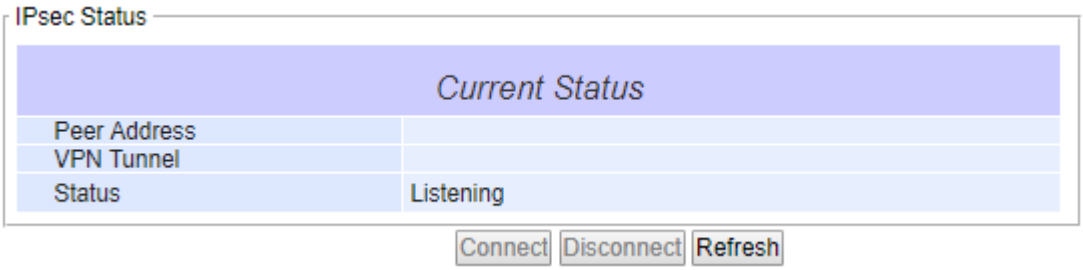


Figure 5.34 IPsec Status Web Page

5.11.3 Examples of IPsec Settings

The following subsections provide examples of **IPsec settings**. However, each example will be focused only on the **General Settings** part. The other parts of the **IPsec Settings** can be configured according to the user's preference. Please consult previous section on the details of **Authentication Settings**, **IKE Settings**, and **Dead Peer Detection Settings**. **Note** that the network-to-network (or subnet-to-subnet) connections are now supported in new firmware of SE59XX.

5.11.3.1 Host-to-Host Connections

Two scenarios can be configured for host-to-host connections: with static peer and with dynamic peer. A host-to-host topology for both scenarios is illustrated in Figure 5.35. Please follow the steps provided next for each scenario to set the **General Settings**.

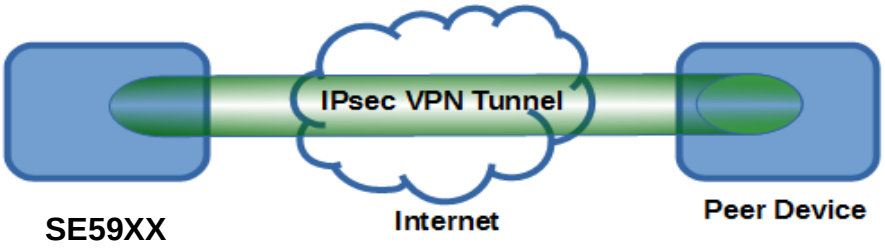


Figure 5.35 IPsec VPN Tunnel with Host-to-Host Topology

Scenario: host-to-host with static peer as shown in Figure 5.36

- Check the **Enable** box for **IPsec**.
- In the **Peer Address** field, select the **Static** option and enter the peer IPv4 address.
  Note: When peer address is entered as the static address, the SE59XX acts as an **initiator** which takes the initiative and establishes a connection. The SE59XX also acts as a **responder** and passively accepts the connection initiated by the remote gateway.
- Select the raio button for **None (Host Only)** in the **Remote Subnet** field.
- Since this VPN connection is established on two hosts, the **Connection Type** option can be either **Transport** or **Tunnel**.

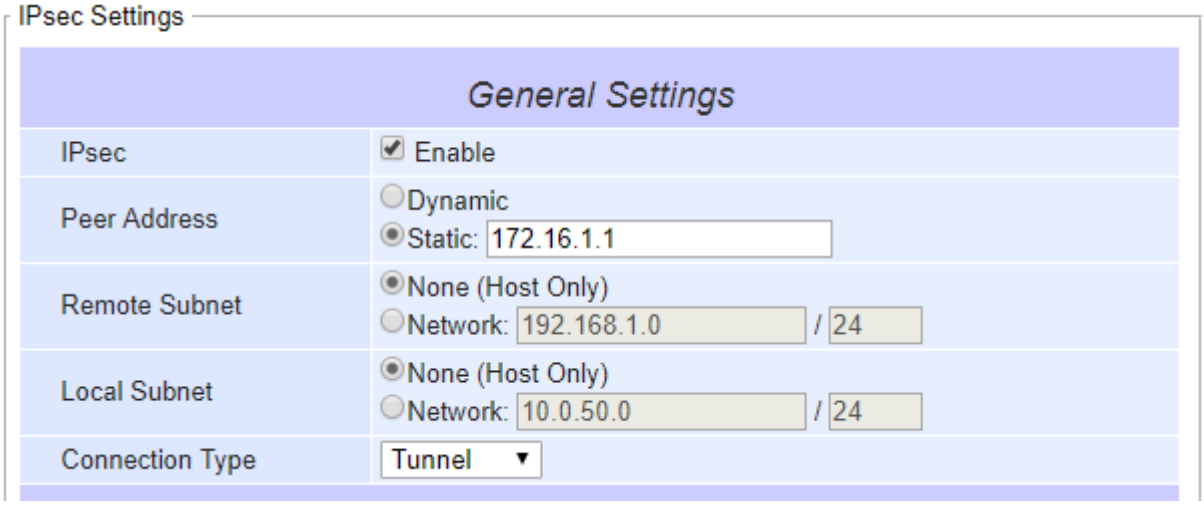


Figure 5.36 General Settings for Host-to-Host with Static Peer

Scenario: host-to-host with dynamic peer as shown in Figure 5.37

- Check the **Enable** box for **IPsec**.
- In the **Peer Address** field, select the **Dynamic** option.
  Note: When VPN connects to a peer with dynamic IP address, the SE59XX acts as a **responder** and passively accepts the connection initiated by the remote gateway.
- The remaining settings are the same as the host-to-host with static peer scenario described above.

IPsec Settings

General Settings	
IPsec	☑ Enable
Peer Address	◉ Dynamic ○ Static:
Remote Subnet	◉ None (Host Only) ○ Network: 192.168.1.0 / 24
Local Subnet	◉ None (Host Only) ○ Network: 10.0.50.0 / 24
Connection Type	Tunnel ▾

Figure 5.37 General Settings for Host-to-Host with Dynamic Peer

5.11.3.2 Host-to-Network Connections

Two scenarios can also be configured for host-to-network (or host-to-subnet or host-to-site) connections: with static peer and with dynamic peer. Note that the SE59XX is the host in these scenarios. A host-to-network topology for both scenarios is illustrated in Figure 5.38. Please follow the steps provided next for each scenario to set the **General Settings**.

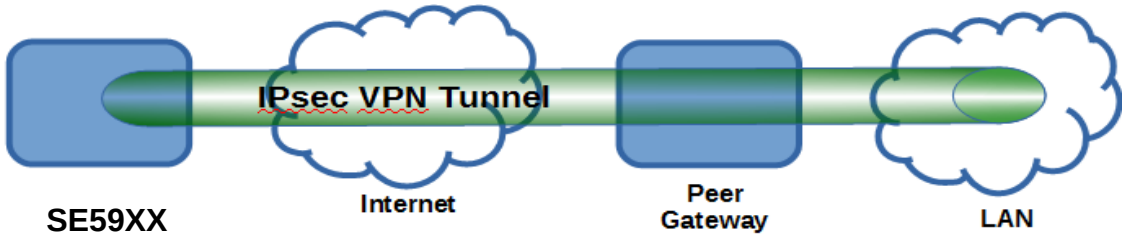


Figure 5.38 IPsec VPN Tunnel with Host-to-Network Topology

Scenario: host-to-network with static peer as shown in Figure 5.39

- Check the **Enable** box for **IPsec**.
- In the **Peer Address** field, select the **Static** option and enter the peer IPv4 address.
  Note: When peer address is entered as a static address, the SE59XX is an **initiator** which takes the initiative and establish a connection, or can be a **responder** waiting for connection. The SE59XX also acts as a **responder** and passively accepts the connection initiated by the remote gateway.
- Set the network IPv4 address in the **Remote Subnet** with the number of bits for subnet mask in "address prefix length" or behind the "**/**" symbol.
- Because this IPsec VPN connection has a subnet at one end, the **Connection Type** option must be set to **Tunnel** only.

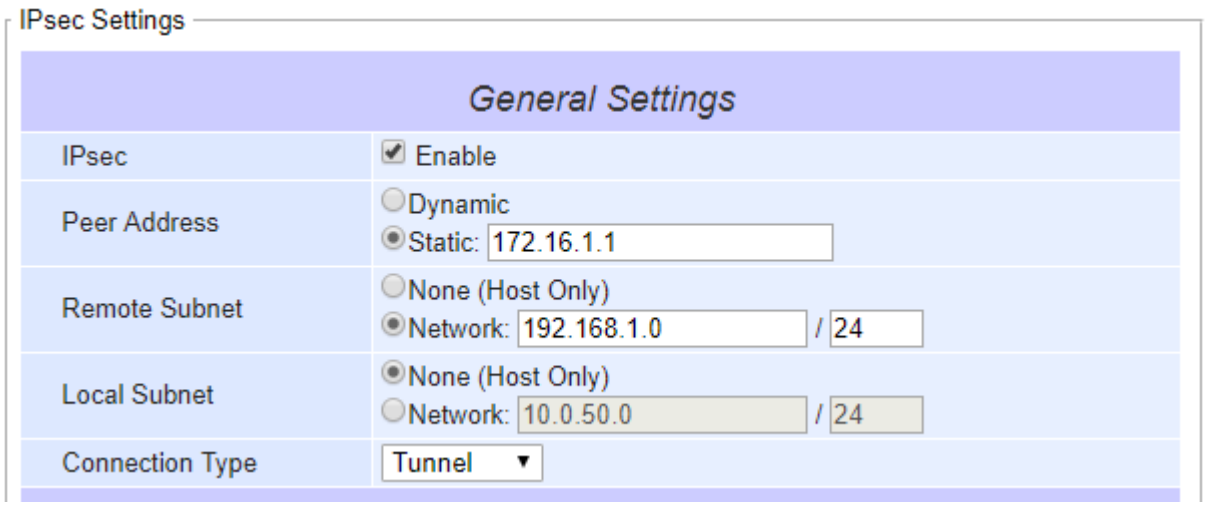


Figure 5.39 General Settings for Host-to-Network with Static Peer

Scenario: host-to-network with dynamic peer as shown in Figure 5.40

- Check the **Enable** box for **IPsec**.
- In the **Peer Address** field, select the **Dynamic** option.
  Note: When VPN connection is set to a peer with dynamic IP address, the SE59XX will act as a **responder** and will passively accept the connection initiated by the remote gateway.
- Set the network IPv4 address in the **Remote Subnet** with the number of bits for subnet mask in "address prefix length" or behind the "/" symbol.
- Because this IPsec VPN connection has a subnet at one end, the **Connection Type** option must be set to **Tunnel** only.

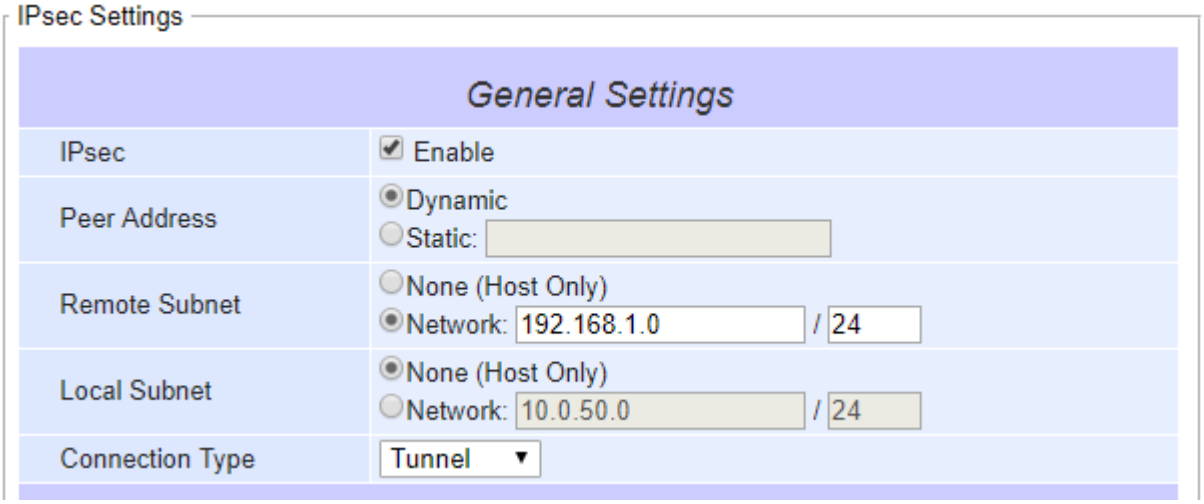


Figure 5.40 General Settings for Host-to-Network with Dynamic Peer

5.11.3.3 Network-to-Network (Subnet-to-Subnet) Connections

Two scenarios can also be configured for network-to-network (or subnet-to-subnet) connections: with static peer or with dynamic peer. A VPN tunnel will be created between two separate private sub-networks. Note that the SE59XX is the gateway to a local network in these scenarios. A network-to-network topology for both scenarios is illustrated in Figure 5.41. Please follow the steps provided next for each scenario to set the **General Settings**.

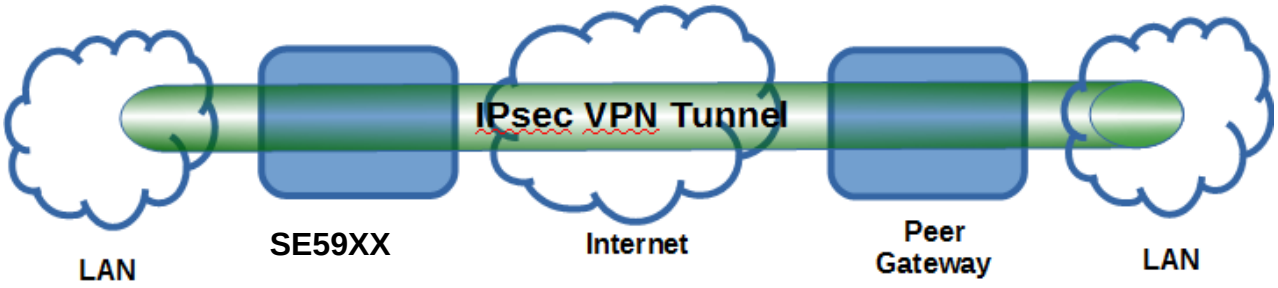


Figure 5.41 IPsec VPN Tunnel with Network-to-Network Topology

Scenario: network-to-network with static peer as shown in Figure 5.42

- Check the **Enable** box for **IPsec**.
- In the **Peer Address** field, select the **Static** option and enter the peer IPv4 address.
  Note: When peer address is entered as a static address, the SE59XX is an **initiator** which takes the initiative and establish a connection, or can be a **responder** waiting for connection. The SE59XX also acts as a **responder** and passively accepts the connection initiated by the remote gateway.
- Set the network IPv4 address in the **Remote Subnet** with the number of bits for subnet mask in "address prefix length" or behind the "**/**" symbol.
- Set the network IPv4 address in the **Local Subnet** with the number of bits for subnetmask in "address prefix length" or behind the "**/**" symbol.
- Because this IPsec VPN connection has subnets at both ends, the **Connection Type** option must be set to **Tunnel** only.

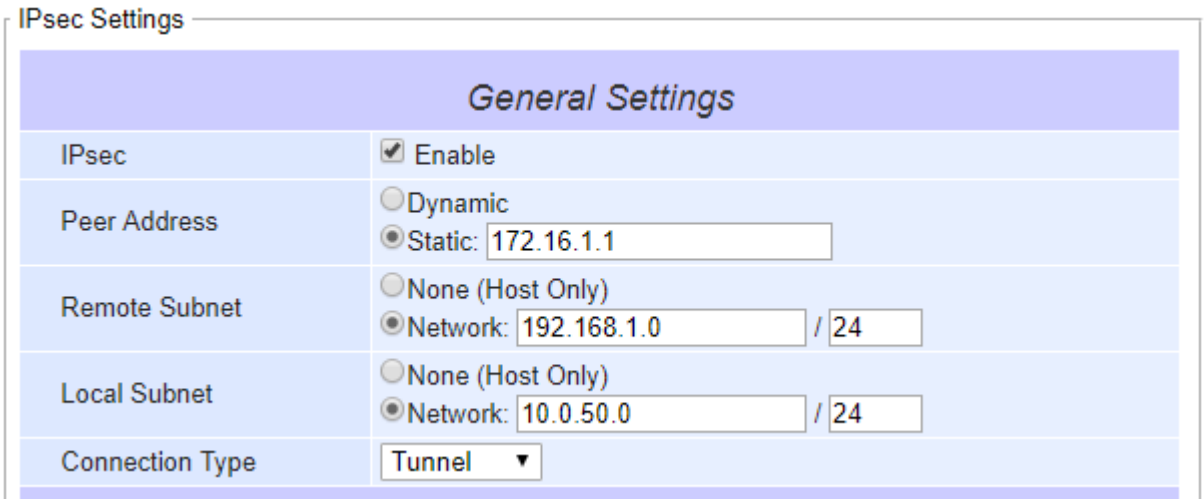


Figure 5.42 General Settings for Network-to-Network with Static Peer

Scenario: network-to-network with dynamic peer as shown in Figure 5.43

- Check the **Enable** box for **IPsec**.
- In the **Peer Address** field, select the **Dynamic** option.
  Note: When VPN connection is set to a peer with dynamic IP address, the SE59XX will act as a **responder** and will passively accept the connection initiated by the remote gateway.
- Set the network IPv4 address in the **Remote Subnet** with the number of bits for subnet mask in "address prefix length" or behind the "**/**" symbol.
- Set the network IPv4 address in the **Local Subnet** with the number of bits for subnetmask in "address prefix length" or behind the "**/**" symbol.
- Because this IPsec VPN connection has subnets at both ends, the **Connection Type** option must be set to **Tunnel** only.

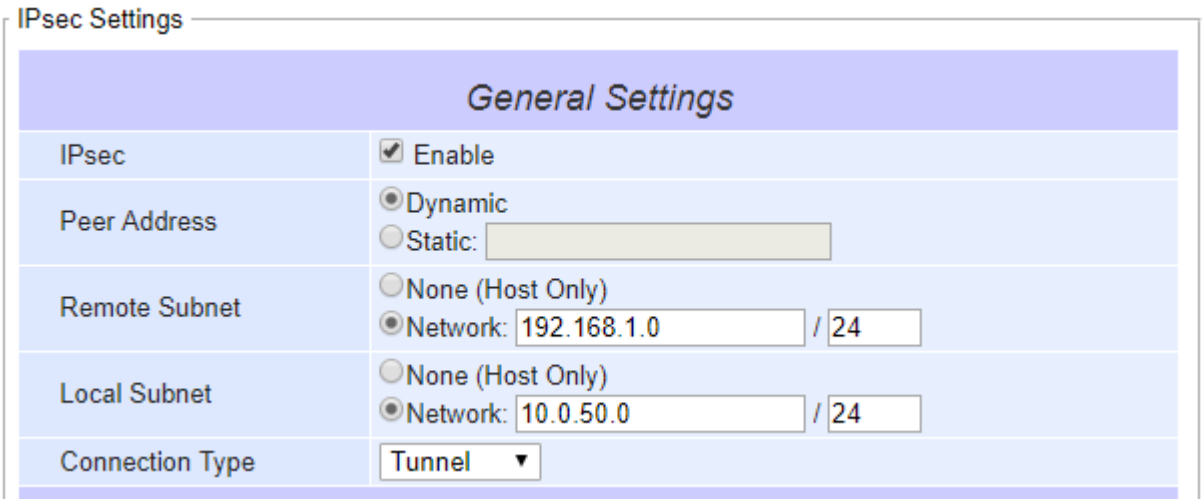


Figure 5.43 General Settings for Network-to-Network with Dynamic Peer

5.12 SNMP/ALERT Settings

The Simple Network Management Protocol (SNMP) is used by network management software to monitor devices in a network, to retrieve network status information of the devices, and to configure network parameters of the devices. The **SNMP/ALERT Settings** page showed in Figure 5.44 allows user to configure SE59XX device so that it can be viewed by third-party SNMP software, and allows SE59XX to send alert events to administrator and SNMP trap server.

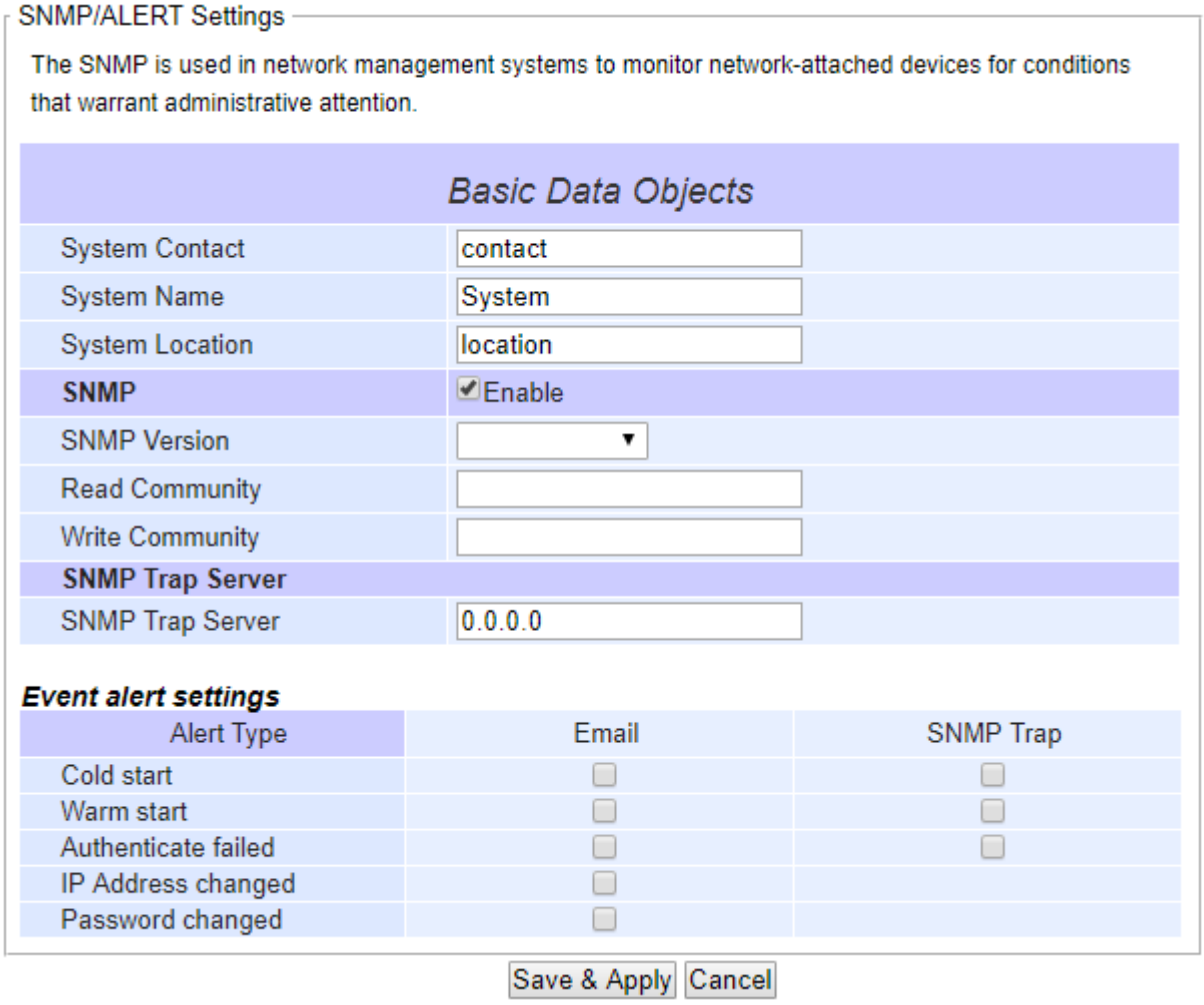


Figure 5.44 SNMP/Alert Settings Web Page

SE59XX provides three basic SNMP fields under the **Basic Data Objects** part which are: "**System Contact"** usually used to specify the device's contact information in case of emergency (default value is "contact"), "**System Name"** usually used to identify this device (default value is "System"), and "**System Location"** usually used to specify the device location (default value is "location").

To make the device's information available for public viewing/editing, you can enable the **SNMP** function by checking the **Enable** box and fill in the two passphrases (or SNMP Community Strings) below it. Note that when the SNMP is unchecked, three setting option lines will show up as depicted in Figure 5.44. By filling in the passphrase for the "**Read Community**", SE59XX device allows other network management software to read its information. By filling in the passphrase for the "**Write Community**", SE59XX device allows other network management software

to read/modify its information. The default SE59XX's SNMP Community Strings (or passphrases) for **Read Community** and **Write Community** as shown in Figure 5.44 are "public" and "private", respectively.

Additionally, you can setup a **SNMP Trap Server** in the network to receive and collect all alert messages from SE59XX. To configure SE59XX to dispatch alert messages originated from any unexpected incidents, you can fill in the IP Address of the **SNMP Trap Server** in the field shown in Figure 5.44. Note that any changes in these settings will take effect after the SE59XX device is restarted.

Under the **SNMP Trap Server** part, there is a list of **Alert Type** under **Event alert settings** box in Figure 5.44. There can be up to two possible actions for each alert event: **Email** and **SNMP Trap**. You can enable the associated action(s) of each alert event by checking the box under the column of **Email** and/or **SNMP Trap**. When the **Email** box is checked and the corresponding event occurs, it will trigger an action for SE59XX to send an e-mail alert to designated addresses configured in the E-Mail Settings (described in the next section). When the **SNMP Trap** box is checked and the corresponding event occurs, it will trigger an action for SE59XX to send a trap alert to the designated SNMP Trap server (specified in the above paragraph). There are five events that will trigger the alarm from SE59XX as listed in Figure 5.44. However, some event can only be reported by e-mail. These alerts are useful for security control or security monitoring of the SE59XX device:

- **Cold Start**: This event occurs when there is a power interruption.
- **Warm Start**: This event occurs when the device resets.
- **Authentication Failure**: This event occurs when an incorrect username and/or password are entered which could indicate an unauthorized access to the SE59XX.
- **IP Address Changed**: This event occurs when the SE59XX device's IP address is changed.
- **Password Changed**: This event occurs when the administrator password is changed.

After finish configuring the **SNMP**/**Alert Settings**, please click on **Save & Apply** button to keep the change that you have made and to apply your setting. When the saving and applying are finished, the web browser will remain on the **SNMP**/**Alert Settings** page. If you want to cancel the change and reset all changes back to their original values, just click the **Cancel** button.

5.13 SMS Settings (SE5901B only)

5.14 E-Mail Settings

When SE59XX device raises an alert and/or a warning message, it can send an e-mail to an administrator's mailbox. This **E-mail Settings** page allows you to set up the SE59XX to be able to send an e-mail. Figure 5.45 shows the **E-mail Settings** page in which there are two configurable parts: **E-mail Address Settings** and **E-mail Server**. First for the **E-mail Address Settings** part, a **Sender**'s e-mail address is required to be filled in the **Sender**'s text box which will be used in the **From** field of the e-mail. Note that the maximum length of sender email address is 48 characters. Then, for the **Receiver**'s text box you can enter multiple recipients which will be used in the **To** field of the e-mail. Note that to fill in multiple receiver e-mail addresses in the **Receiver**'s text box, please separate each e-mail address with semicolon (;).

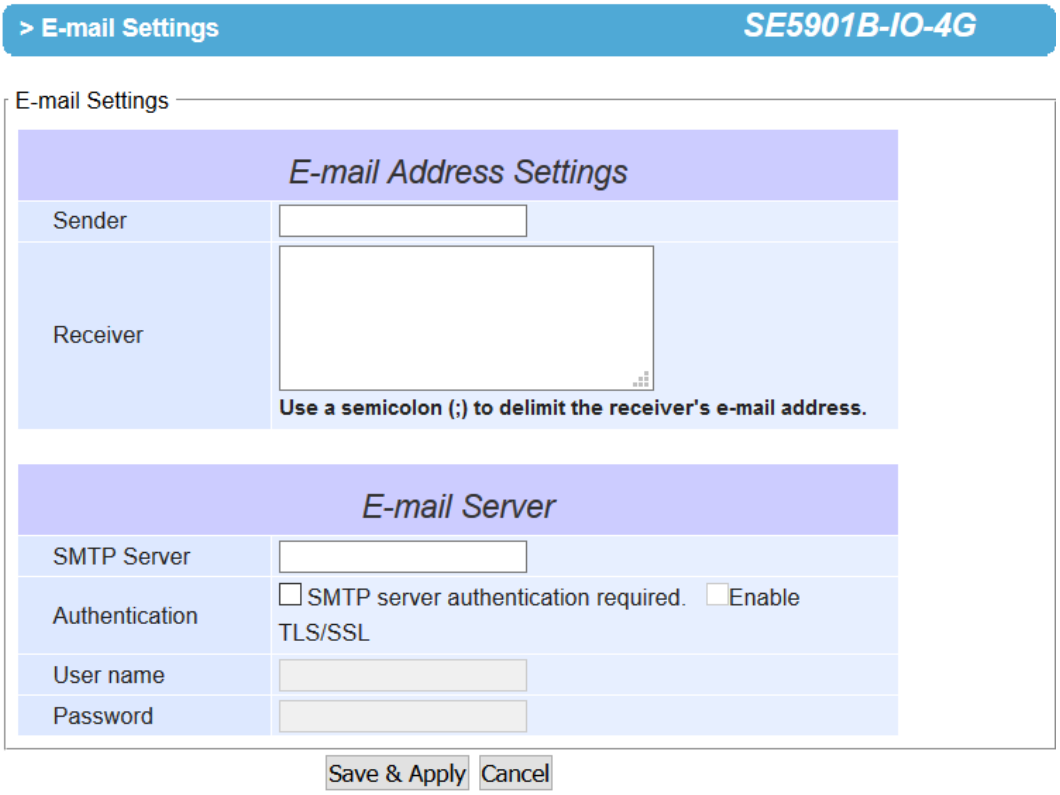


Figure 5.45 E-mail Setting Web Page

Second for the **E-mail Server** part, you must enter an **IP address** or **Host Name** of a **Mail Server** which is in your local network in the **SMTP Server**'s text box. Note that the maximum length of SMTP server address is 31 characters. If your Mail Server (or Simple Mail Transfer Protocol (SMTP) Server) requires a user authentication, you must check the "**SMTP server authentication required**" and/or "**Enable TLS/SSL**" checkbox in the **Authentication** option based on your ISP settings.

After enabling this option, you can fill in the **Username** and the **Password** below. Please consult your local network administrator for the **IP address** of your **Mail Server** and the required **Username** and **Password**.

Attention

It is also important to setup Default Gateway and DNS Servers in the Network Settings properly so that SE59XX can lookup domain names and route the e-mails to the proper default gateway. Please see the Default Gateway and DNS Sever Settings in Section o .

After finish configuring the **E-mail Settings**, please click on **Save & Apply** button to keep the change that you have made and to apply your setting. When the saving and applying are finished, the web browser will remain on the **E-mail Settings** page. If you want to cancel the change and reset all changes back to their original values, just click the **Cancel** button.

5.15 Log Settings

Under the **Log Settings** menu of web interface of SE59XX, you can configure various data logging for the device. Figure 5.46 lists the sub-menu under the **Log Settings**. It consists of **System Log Settings**, **and Event Log.** Each of this sub-menu will be described in the following subsections.

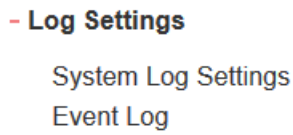


Figure 5.46 Log Settings Menu

5.15.1 System Log Settings

The Syslog function is turned on by default and cannot be turned off for SE59XX. It is used to keep log for system events and report to an external Syslog server if necessary. Figure 5.47 shows the **System Log Settings** page under the **Log Settings** menu. Description of each option is provided as follows.

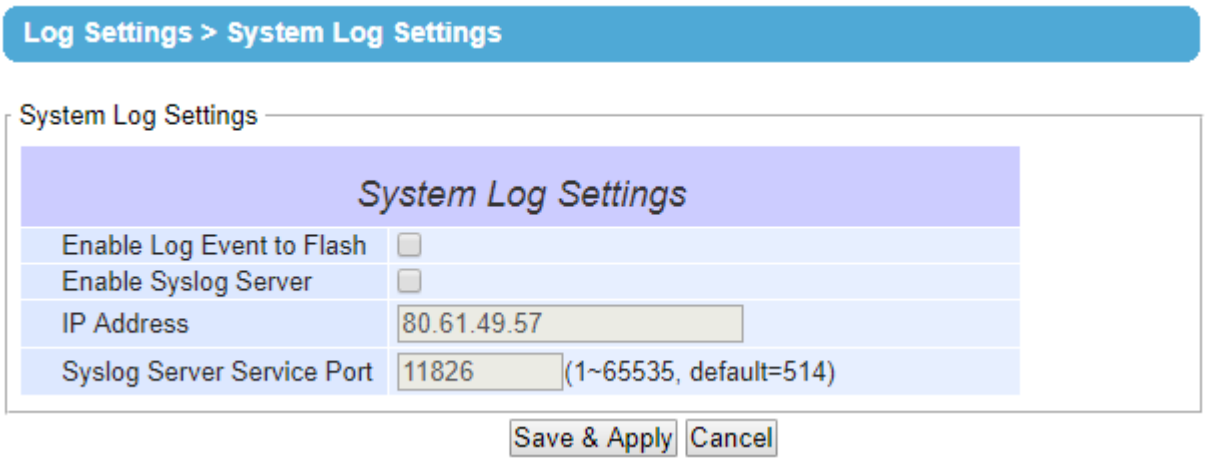


Figure 5.47 Log Settings Web Page under Log Settings

- **Enable Log Event to Flash**: When the check box is enabled, SE59XX will write log events to the local flash. Otherwise the log events would be cleared when the device restarts because they are stored in the RAM by default.
- **Enable Syslog Server**: When the check box is enabled, it will allow SE59XX to send Syslog events to the remote Syslog server with the specified IP address (next option). All the data sent/received from serial interface will be logged and sent to Syslog Server.
- **Syslog Server IP**: The user must specify the IP address of a remote Syslog Server in this field.
- **Syslog Server Service Port**: This option allows user to specify the remote Syslog Server Port number between 1 and 65535. Note that the default port number is 514.

After finish configuring the **Log Settings**, please click on **Save & Apply** button to keep the change that you have made and to apply your setting. When the saving and applying are finished, the web browser will remain on the **Log Settings** page. If you want to cancel the change and reset all changes back to their original values, just click the **Cancel** button.

5.15.2 Event Log

This page displays the current event log or system log stored in the SE59XX device. Figure 5.48 shows an example of logged event. Each record of the **System Log** consists of **Time**, **Severity**, and **Message** description.

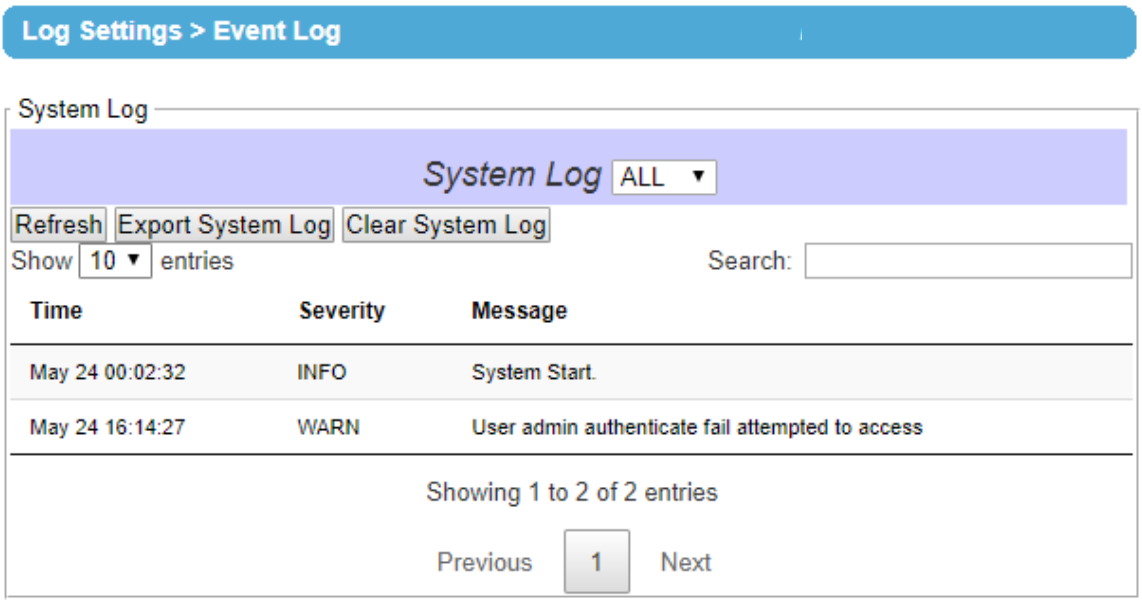


Figure 5.48 System Log Web Page under System Setup

At the end of the **System Log** page, there are three hyperlinks which can be used to navigate through all records. You can click on the "**Previous**" `link to go to the last page of the log and click on the "**Next**" button to go to the next page. At the top of the **System Log** table, there are three buttons: **Refresh**, **Export System Log**, and **Clear System Log**. To display the latest event, you can click on **"Refresh"** button. When you click on the Export System Log button, a log file will be save on to your PC. By clicking on "**Clear System Log**" button, you can clear all events stored in the device and the **System Log** will be empty. A message "No data available in table" will be displayed in the middle of the table. Moreover, you can choose from the drop-down list of 10 or 25 entries for the **Show entries**. Finally, you can search over the **System Log** by entering a keyword in the **Search** box.

5.16 System Setup

Under the **System Setup** menu of web interface of SE59XX series, you can perform a number of administration tasks for the device. Figure 5.49 lists the sub-menu under the **System Setup**. It consists of **Date/Time Settings**, **Admin Settings**, **Firmware Upgrade**, and **Backup/Restore Setting**. Each of this sub-menu will be described in the following subsections.

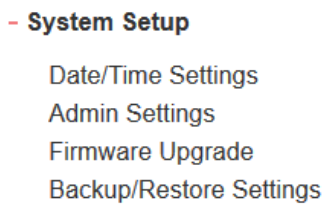


Figure 5.49 System Setup Menu

5.16.1 Date/Time Settings

Date and time can be set manually or using Network Time Protocol (NTP) to automatically synchronize date and time of SE59XX with a Time Server. Figure 5.50 shows the **Date/Time Settings** page. The first part of the page is the latest **Current Date/Time** which is in the format of **DD/Month/YYYY HH:MM:SS**. The second part of the page is the **Time Zone Settings**. You can select your local **Time Zone** from the drop-down list. The third part of the page is the **NTP Server Settings**. In this part, you can either enable the local NTP service inside SE59XX by checking the option **Local NTP Service** below **NTP Settings** part or automatically synchronize with a time server or NTP server. To enable automatic time synchronization, please check the box behind the **Sync with NTP Server** option. Then proceed to enter the **IP address** or **host name** for the **NTP Server**. Note that if a host name is entered, the DNS server must be configured properly (see detail in Section o). The fourth part is the **Daylight Saving Time Settings** that can be enabled when **Enable Daylight Saving Time** box is checked. When it is enabled, the user can select the detailed setting of the daylight saving period, such as **Start Date** and **End Date** with **Offset**. Finally, the last part of the page is the **Manual Time Settings** where you can set **Date** and **Time** using corresponding drop-down lists in Figure 5.50.

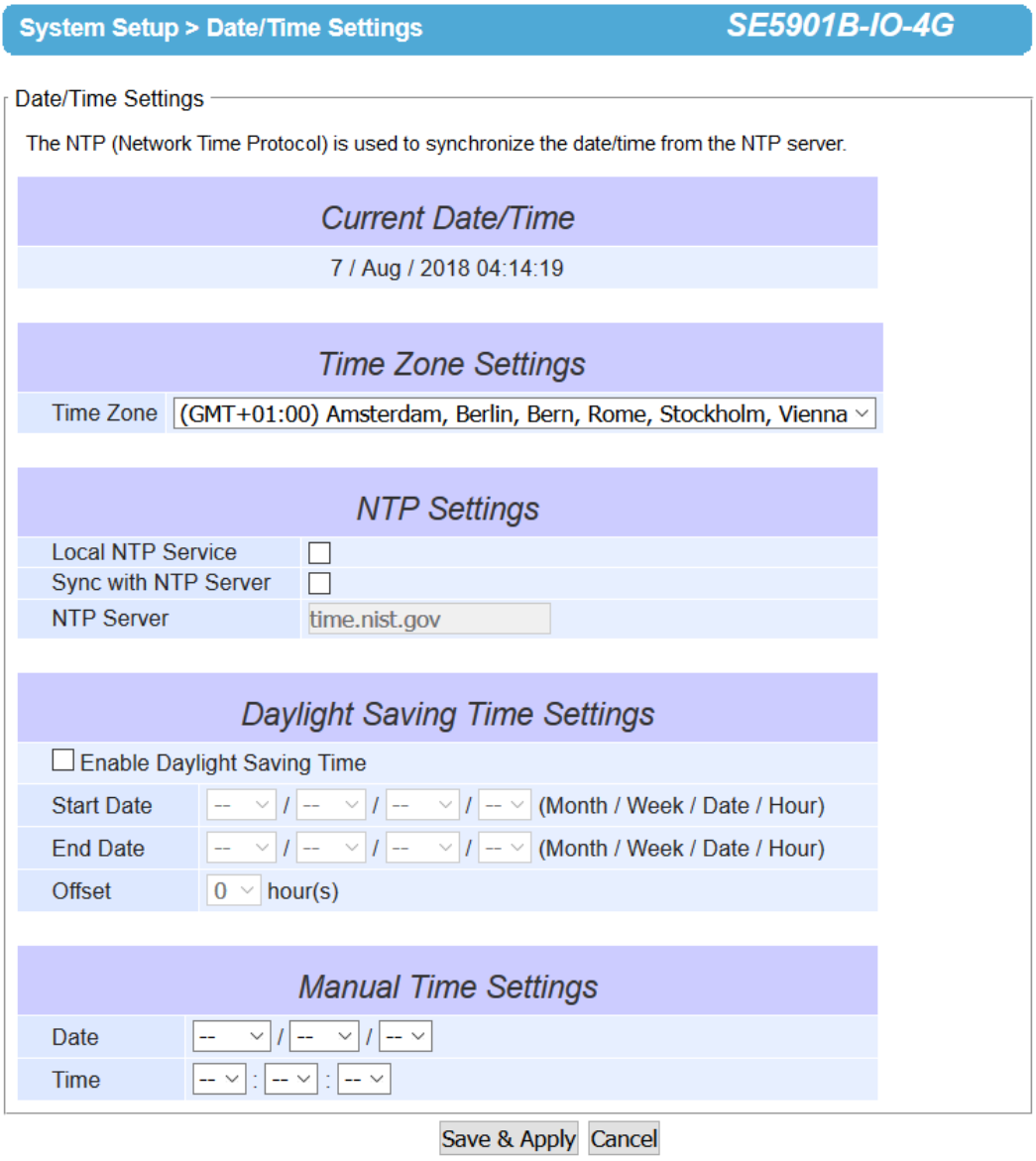


Figure 5.50 Date/Time Settings Web Page under System Setup

Attention

It is also important to setup Default Gateway and DNS Servers in the Network Settings properly (See Section o), so SE59XX can lookup DNS names and point to the proper NTP server.

After finish configuring the **Date/Time Settings**, please click on **Save & Apply** button to keep the change that you have made and to apply your setting. When the saving and applying are finished, the web browser will remain on the **Date/Time Settings** page. If you want to cancel the change and reset all changes back to their original values, just click the **Cancel** button.

5.16.2 Admin Settings

The SE59XX Series allows user and password management through this **Admin Settings** page under **System Setup** menu. By default, the user name is "**admin**" and the password is "**default**". To set or change their values, you can enter the information in the **User name**, the **Old password**, the **New password** and the **Repeat new password** fields under the **Account Settings** part as shown in Figure 5.51. At the end of the **Admin Settings** web page, there is the **Web mode** part which allow the user to select the radio button of normal **HTTP** or **HTTPS** for secure communication with the device's web user interface (Web UI).

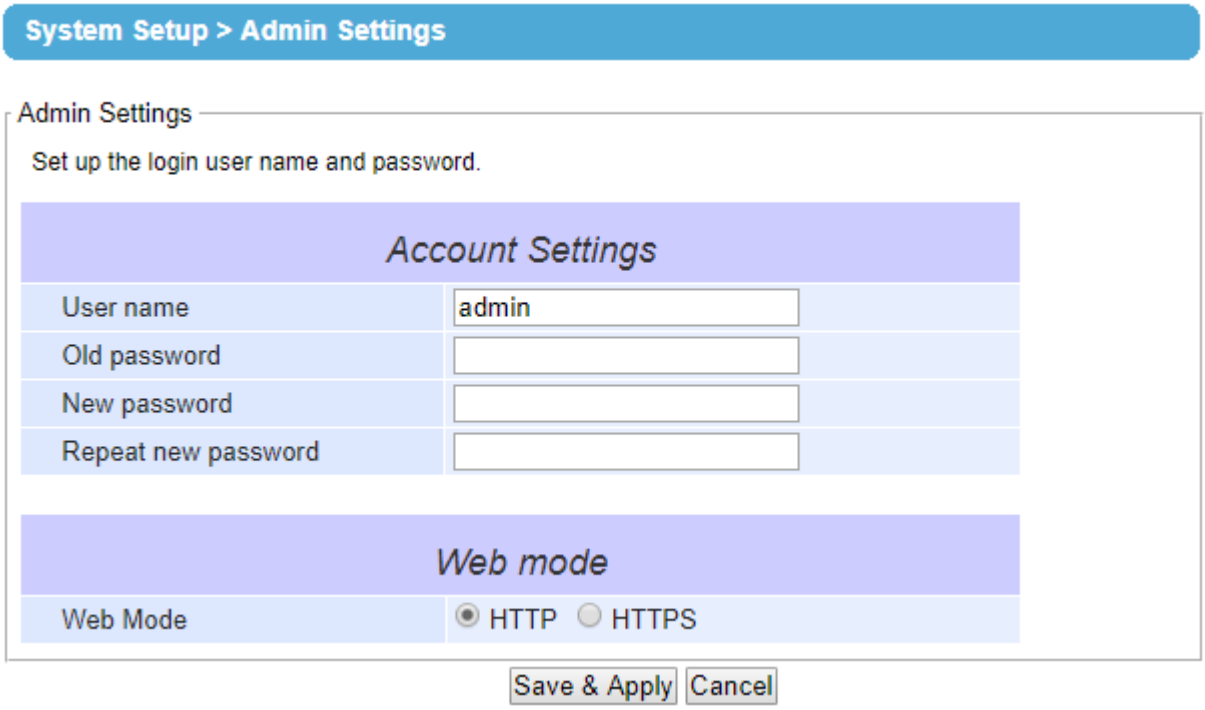


Figure 5.51 Admin Settings Web Page under System Setup

After finish configuring the **Admin Settings**, please click on **Save & Apply** button to keep the change that you have made and to apply your setting. Another pop-up window will be displayed to re-authenticate the user to access the Web UI of SE59XX as shown in Figure 5.1. You must re-enter the username and the password to login to the SE59XX. When the saving, applying, and re-authentication are finished, the web browser will remain on the **Admin Settings** page. If you want to cancel the change and reset all changes back to their original values, just click the **Cancel** button.

5.16.3 Firmware Upgrade

Updated firmware for SE59XX is provided by Atop from time to time (for more information please visit Atop News & Events webpage) to fix bugs and optimize performance. It is very important that the device must **NOT be turned off or powered off during the firmware upgrading, (please be patient as this whole process might take up to 5 minutes)**. Before upgrading the firmware, please make sure that the device has a reliable power source that will not be powered off or restarted during the firmware upgrading process.

To upgrade a new firmware to SE59XX, please downloaded the latest firmware for your SE59XX model from the download tab on the SE59XX product page or from the Download page under the Support link on Atop's main webpage. Then, copy the new firmware file to your local computer. Note that the firmware file is a binary file with ".dld" extension. Next, open the Web UI and select **Firmware Upgrade** page under the **System Setup** menu. Then, click "**Browse**…" button as shown in Figure 5.52 below to find and choose the new firmware file. Then, click "**Upload**" button to start the firmware upgrade process. The program will show the upload status. Please wait until the uploading process is finished (the amount of time varies depending on the equipment used). Finally, the SE59XX device will then proceed to restart itself. In some cases, you might require to re-configure your SE59XX device. To restore your backup configuration from a file, please see the procedure in the next subsection.

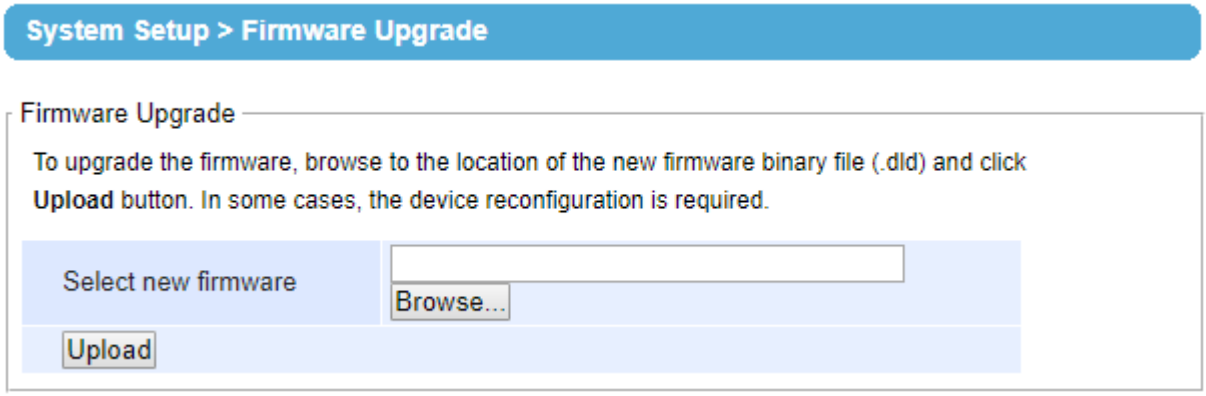


Figure 5.52 Firmware Upgrade Web Page under System Setup

Note: if the firmware upgrade process fails and the device becomes unreachable, please follow the TFTP recovery procedure in Chapter **Error! Reference source not found.** on Emergency System Recovery at the end of this manual.

5.16.4 Backup/Restore Settings

Once all the configurations are set and the device is working properly, the user should back up the current configuration of SE59XX. The backup configuration file can be used when the new firmware is uploaded and the device is reset to a factory default setting. This is done to prevent accidental loading of incompatible old settings. The backup configuration file could also be used to efficiently deploy multiple SE59XX Series devices of similar settings by uploading these settings to all devices.

To back up configuration, click "**Backup**" button under the **Backup Configuration** part as shown in Figure 5.53, and a the backup file (ModelName-MACAddress.dat) will be automatically saved on your computer. It is important **NOT to manually modify the saved configuration file by any editor. Any modification to the file may corrupt the file and it may not be used for later restoration.** Please contact Atop authorized distributors for more information on this subject.

To restore the backup configuration, click "**Browse**" button under the **Restore Configuration** part as shown in Figure 5.53 to locate the backup configuration file on user's computer. Then, click on "**Upload**" button to upload the backup configuration file to the device. Once the backup configuration file is successfully uploaded, the device will restart. Note that the time needed for this process may vary on the equipment used.

If you need to restore the SE59XX device to its factory default configuration, you can click on the **Restore** button under the **Restore Factory Default** section as shown in Figure 5.53.

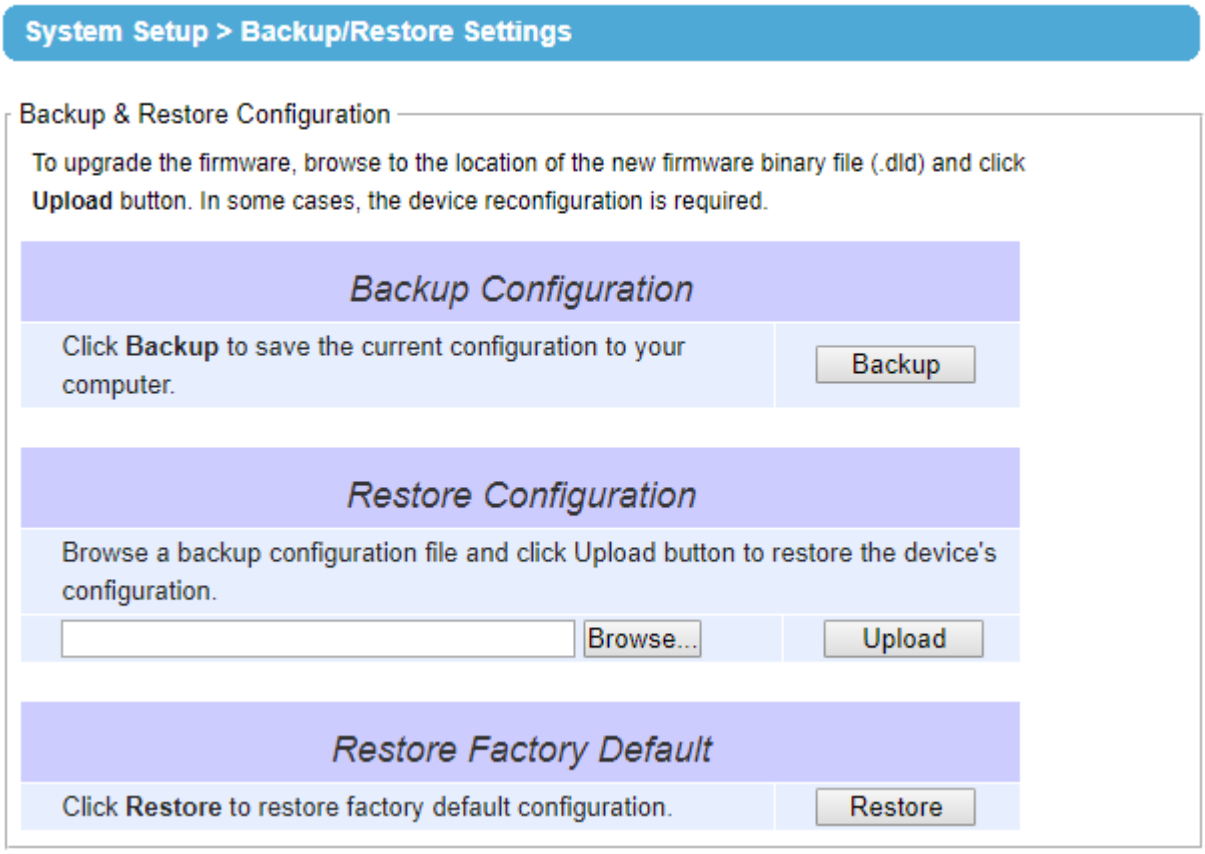


Figure 5.53 Backup/Restore Settings Web Page under System Setup

5.17 Accessing Node-RED flow-editor and dashboard

6 Using Node-RED

6.1 Accessing Node-RED flow-editor and dashboard

This chapter will explain you how to access Node-RED after this has been properly set-up and installed on the SD-Card or USB pen drive on SE59XX. If you have purchased ATOP's Node-RED version with preinstalled SD-card or USB Pen-drive, or if you have already followed all steps described in Section 4 above then Node-RED will start up automatically as soon as the device is powered on and Operating System is loaded.

Note: If you're installing the Node-RED library on the device by yourself, please read Section 4 above first.

Node-RED flow editor is accessible by default on the following path: http://DEVICE_IP_ADDRESS:1880/#
Node-RED dashboard is accessible by default on the following path: http://DEVICE_IP_ADDRESS:1880/ui/#/0

For example: http://10.0.50.100:1880/# and http://10.0.50.100:1880/ui/#/0

Node-RED is designed to work on all major browsers, such as Google Chrome, Mozilla Firefox, Safari (both mobile and desktop) and Internet Explorer

Note: Please make sure your computer/mobile 's IP address is in the same subnet of SE59XX.

Note: If port number has been modified as per Section 5.7.1 above, please replace "1880" with the port number set.

After inputting the address above in the address bar of the browser, you'll be redirected to the Node-RED login page, as Figure 6.1. Please note that in below example the IP address has been changed to avoid conflict.

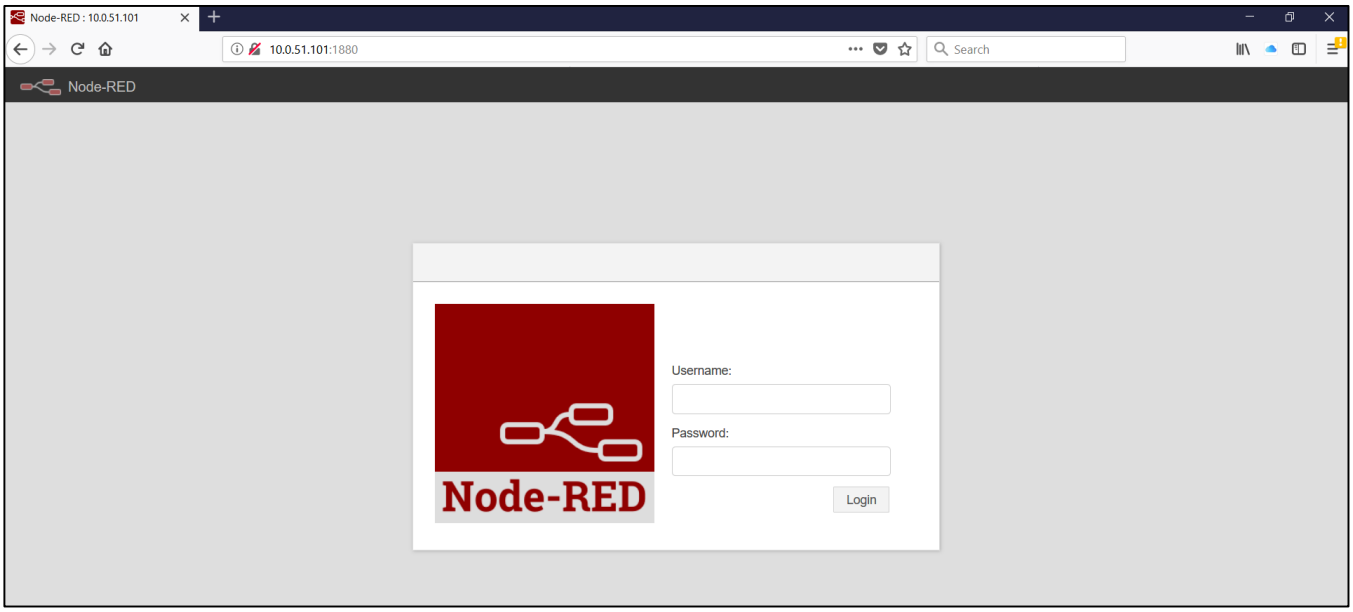


Figure 6.1 Node-RED login page

6.1.1 Login to Node-RED

Access to Node-RED flow editor and dashboard requires a username and password. The default username is “admin” and the default password is “password”. After logging in is successful, you will be presented a sample flow generated by ATOP or, if you’re logging in on the UI, with a sample UI based on ATOP’s flow. This can be deleted and replaced with your own flow.

In order to change your access credentials, please check Section 5.7.2 for Flow editor and Section 5.7.3 for dashboard.

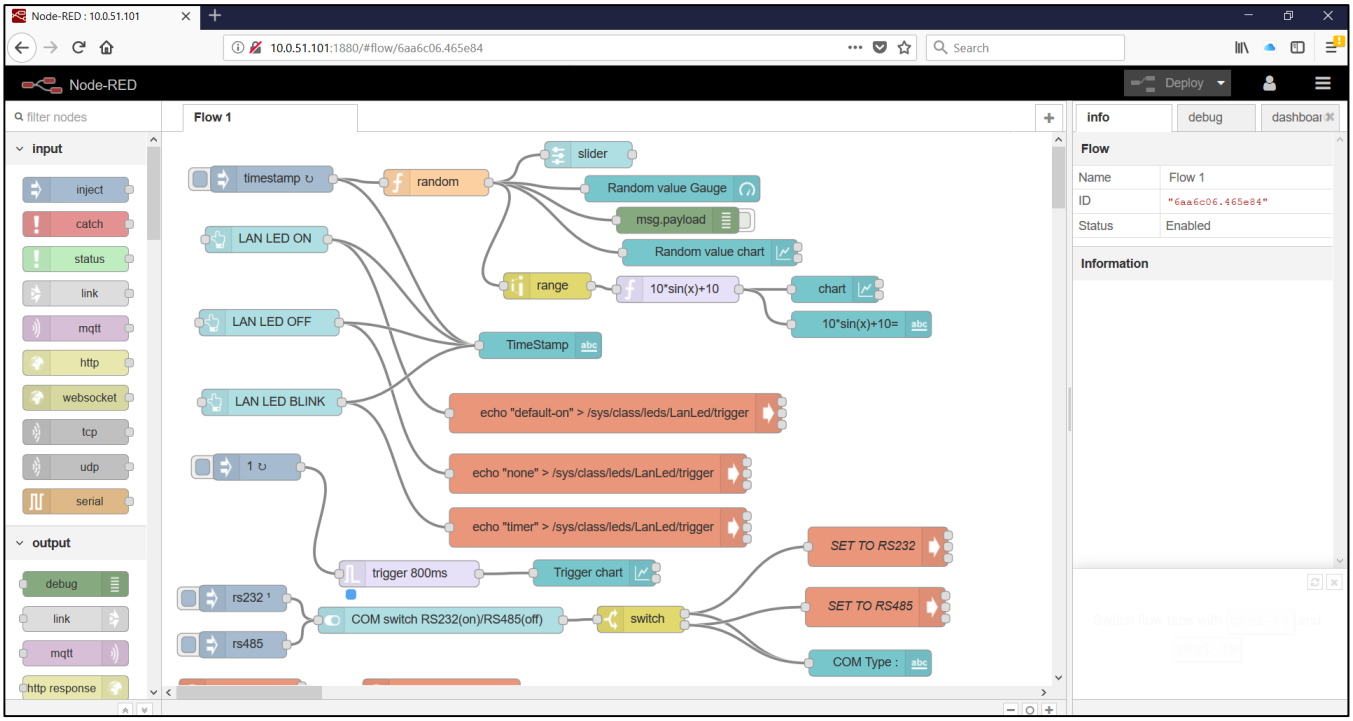


Figure 6.2 Node-RED flow example

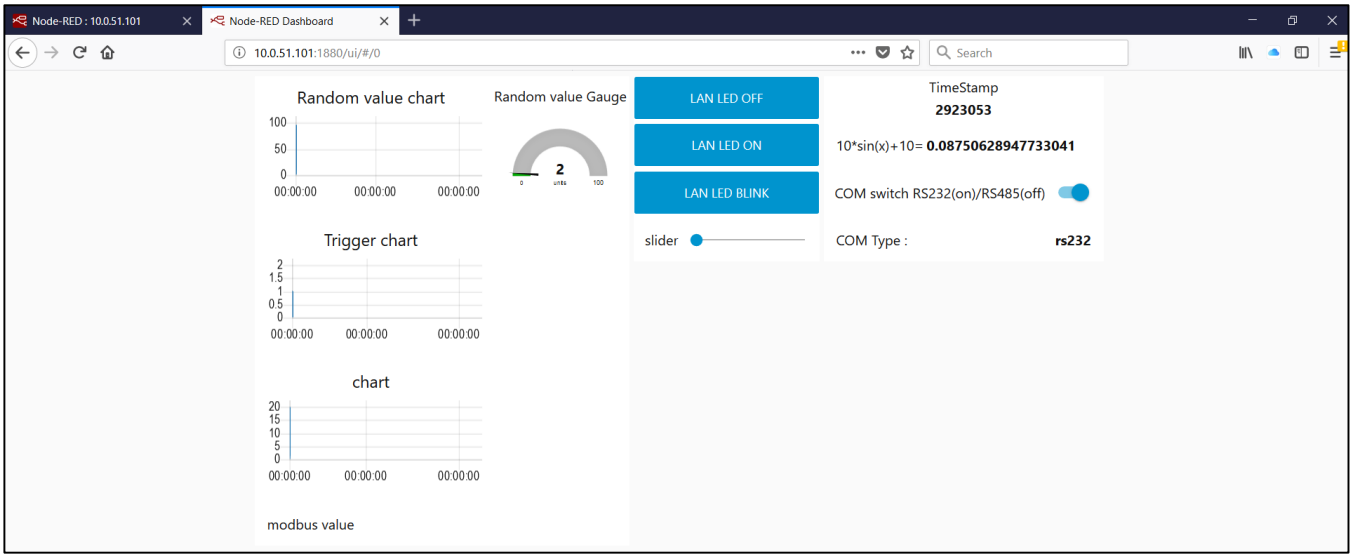


Figure 6.3 Node-RED dashboard example

6.2 Node-RED overview

6.2.1 Node-RED flow

Figure 6.4 shows the sample Node-RED flow that is pre-loaded in the device. This is for allowing the user to get a hands-on understanding on how it works. The Flow-editor page is divided into 4 sections:

- Node palette selector : displays the available building block “nodes” that can be used inside the flow
- Flow workspace : displays the application running on Node-RED
- Information panel : provides access to information related to the Node selected on the flow-area, to the debug information or to the dashboard-related settings (ordering, size, etc..)
- User Menu : provides access to user configuration (such as change of access credentials), import and export of Flows, generation of flows or sub-flows and application deployment

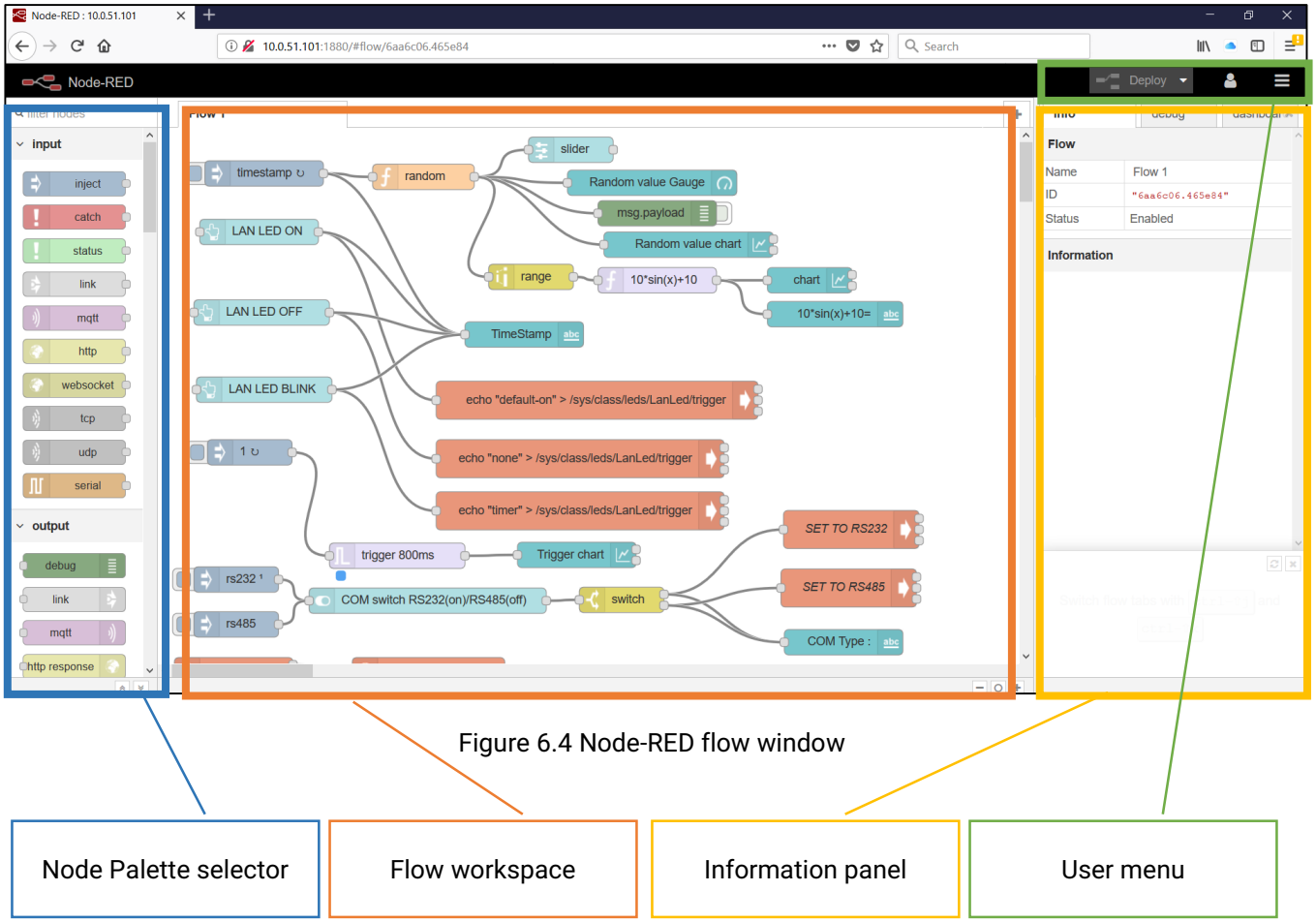


Figure 6.4 Node-RED flow window

Node Palette selector | Flow workspace | Information panel | User menu

6.2.2 Node palette selector

ATOP's SE59XX Node-RED has several Nodes that are pre-installed on the device. The user is able to add its own customized Nodes or function specific nodes working directly on the SD-card or on the USB pen-drive filesystem. How to add or install additional nodes is outside the scope of this user manual. For more information, please refer to Node-RED documentation on www.node-red.org . Figure 6.5 shows the different Node Categories available on Node-RED. They are split in 8 categories, based on the function.

1) INPUT: represent an input node (or a fixed variable) – this can also be an MQTT subscriber.
2) OUTPUT: represents an output node (or a fixed variable) – this can also be an MQTT publisher, an HTTP POST, etc.
3) Function: represents a function node that is used to process data or introduce a delay. Such as function, delay, trigger, switch, sort, split, join
4) Social: represents a node that can interact with social networks, e-mail, etc..
5) Storage: represents a node that can read-write on the filesystem. This is useful for datalogging
6) Analysis: mainly not used
7) Advanced: Exec function is useful for allowing SE59XX running binary applications on the device. Please note that ATOP provides simple I/O programs that allow the use of hardware such as Digital Inputs, Digital Outputs, Relays, Buzzer, LEDs etc.. via a binary application preinstalled on the device.
8) Modbus: function nodes that integrate and parse a Modbus Protocol Stack
9) Dashboards: graphical elements linked to the NodeRED dashboards. These can be input buttons, fields, sliders or output gauges, graphs, etc..

Figure 6.5 Node-RED nodes categories

In order to use a specific node, please select it from the Node Palette selector and drag it to the Flow workspace. Afterwards, please set the related node parameters and make the necessary connections in order to allow the node to receive the proper inputs and/or generate the correct outputs.

Table 6.1 SE5908A/16A Pin Assignment for 5-Pin Terminal Block to RS-232/RS-422/RS-485 Connectors

input	inject	catch	status	link	mqtt
	http	websocket	tcp	udp	serial
output	debug	link	mqtt	http response	websocket
	tcp	udp	serial		
function	function	template	delay	trigger	comment
	http request	tcp request	switch	change	range
	split	join	csv	html	json
	xml	yaml	f(x) curve	rbe	
social	e mail	twitter			
storage	tail	file			
analysis	sentiment				
advanced	watch	feedparse	exec		
modbus	modbus response	modbus read	modbus getter	modbus flex getter	modbus write
	modbus server	modbus queue info	modbus flex connector		
dashboard	button	dropdown	switch	slider	numeric
	text input	date picker	colour picker	form	text
	gauge	chart	audio out notification	ui control	template

6.2.3 Flow Workspace

The flow workspace is where the real embedded application is designed. The user-friendly interface allows to drag the nodes necessary to the application from the Node Palette selector and drop them into the Flow Workspace, set them up and wire them generating the application you need.

A section of ATOP's factory-default Flow is the following:

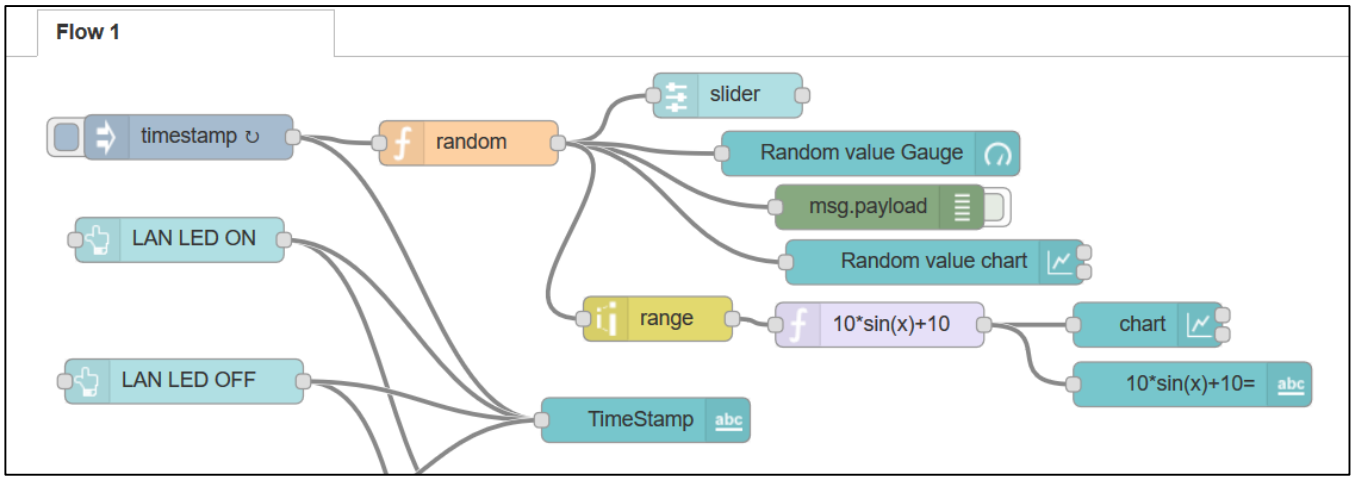


Figure 6.6 Node-RED Flow workspace

By clicking on any node, the Information panel on the right side of the browser will highlight, in the "Info" section, the information related to the function and the current settings as Figure 6.7.

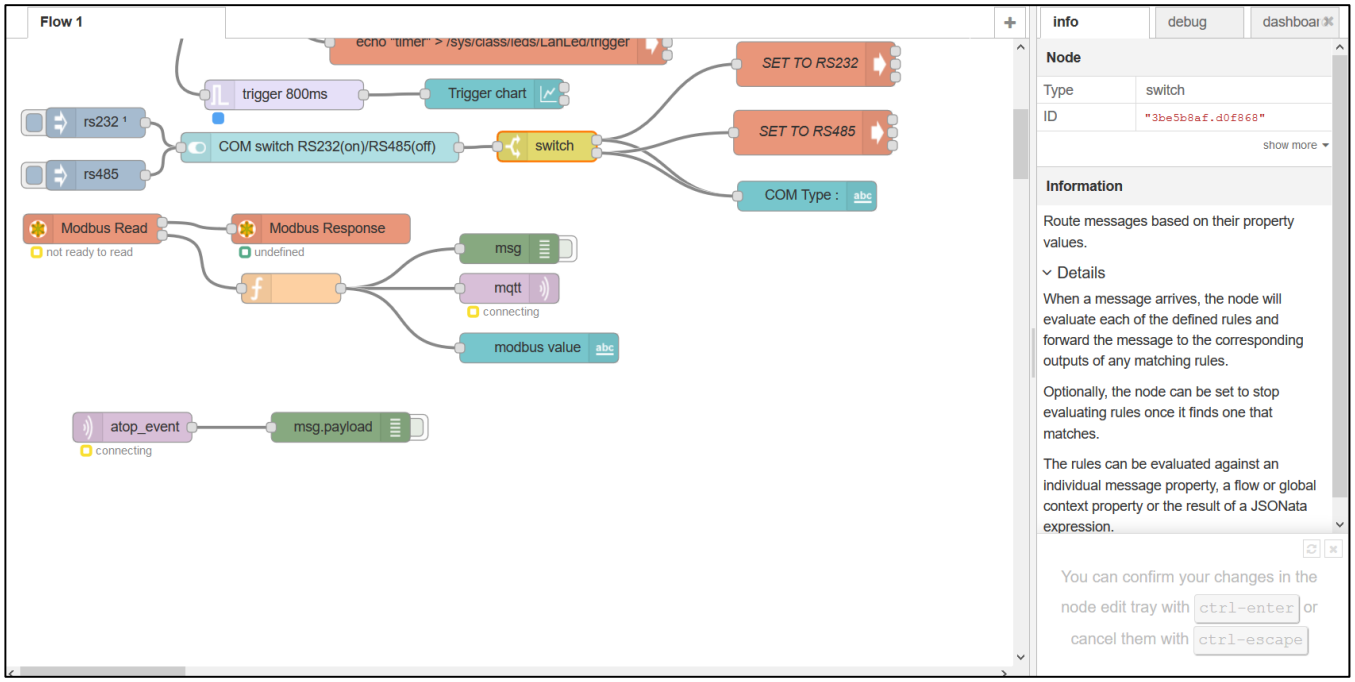


Figure 6.7 Node information panel (example on "switch" function)

By double-clicking the node, a configuration panel will open up allowing the user to input the Node-specific parameters for smooth and effective configuration. An example is provided in Figure 6.8. The parameters are:

- Node Name (useful for remembering what is the Node about). Examples in Section 7.2 below
- Node-specific settings (related to which node you're using). Examples in Section 7.2 below
- Dashboard- specific settings (related to the appearance of the element on the dashboard page) Examples in Section 7.3 below

Note: Dashboard settings will appear on dashboard-related nodes only

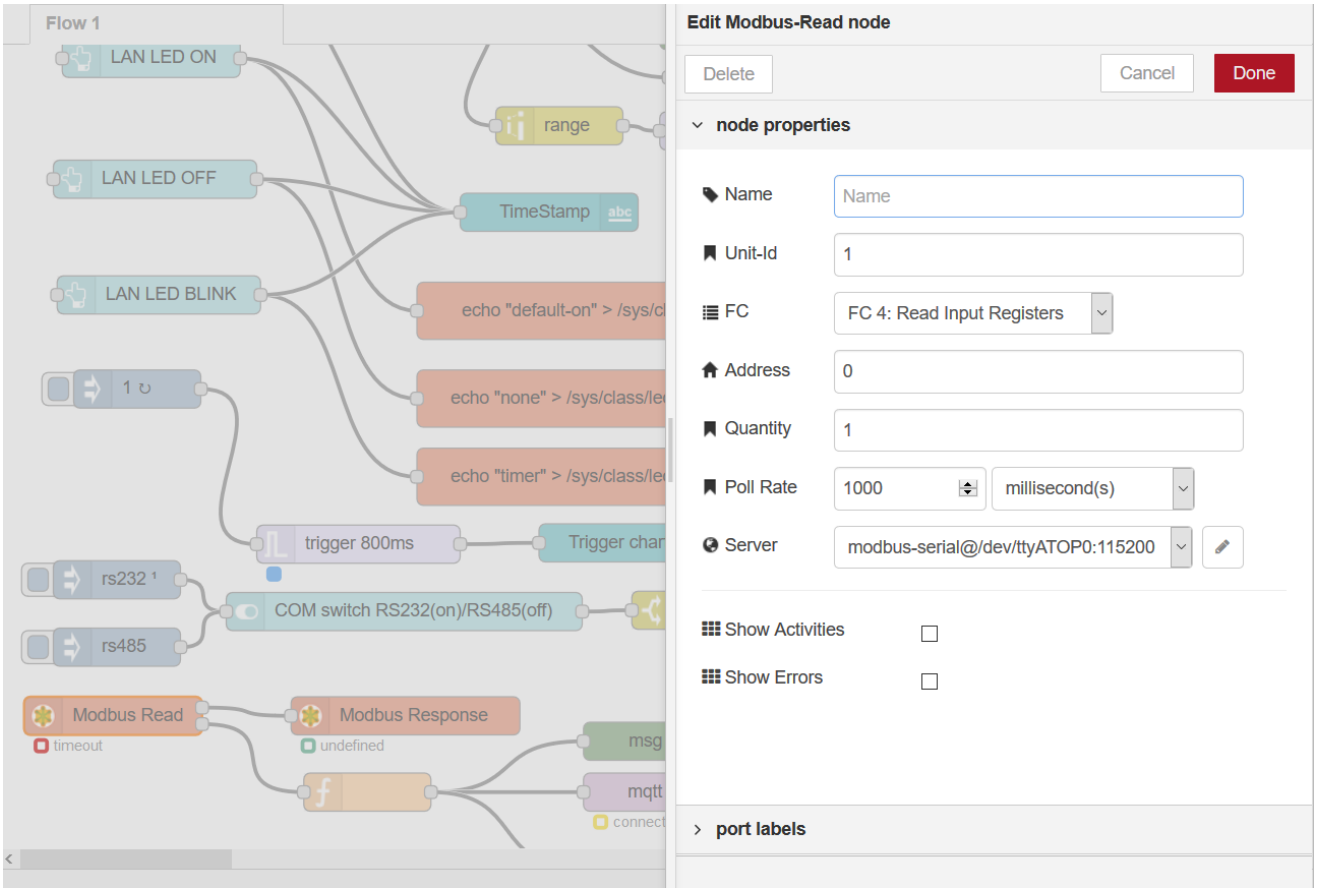


Figure 6.8 Node configuration panel (example on "Modbus-Read" Node)

6.2.4 Information Panel

The information panel is made of three different tabs: info, debug and dashboard.
The info tab is designed to show additional information that can be helpful for the developer to use the Node properly. It will display what is the fundamental parameters to have the function to run, the format of the result and the current configuration settings. Example available in Figure 6.7

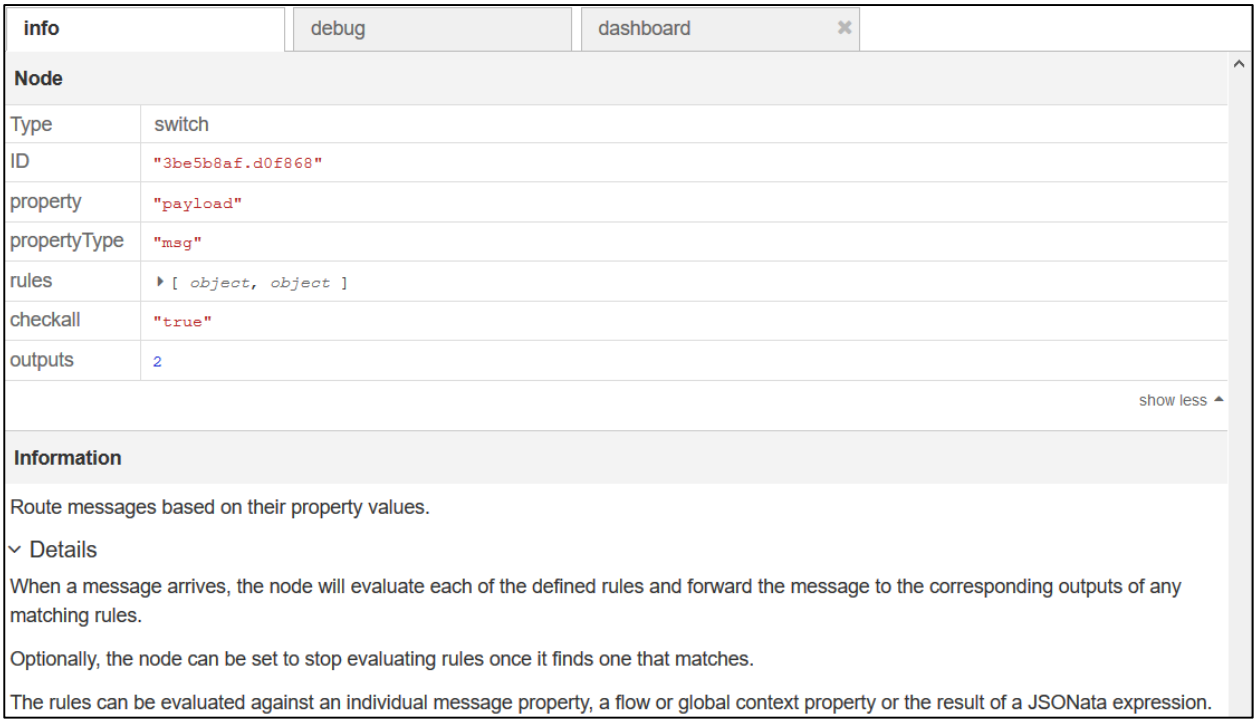


Figure 6.9 Information panel for "Switch" node

The debug tab allows the user to review real-time data passing through the Node, in order to make debug operations easier.
The Dashboard tab is split into three different sub-tabs. The purpose of Dashboard section is to allow the user to define the order and the appearance of the graphical elements. This tab is selection-independent and from here the user can select any element and configure it for Layout (position), Theme (colors), Site (title, formats, etc..). Click on the circled item in order to open a Dashboard tab. This is an alternative to key in the dashboard path manually (http://IP_ADDRESS:1881/ui/#/0). Examples and details are available in Section 7.3 below

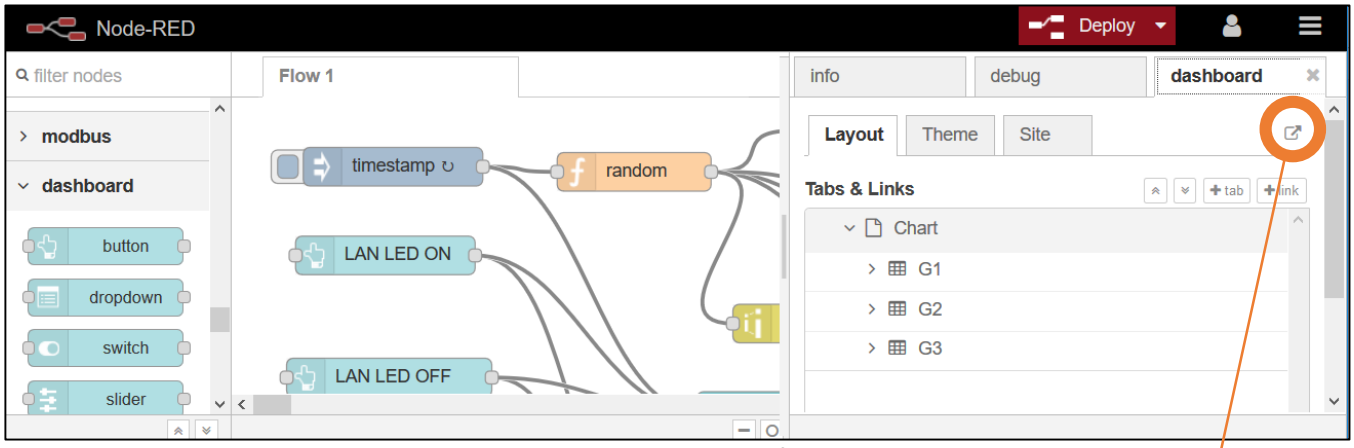


Figure 6.10 Dashboard configuration

6.2.5 User Menu

The user menu is located on the top-right corner of the screen and allows the user to define settings, save and export flows, import flows, etc.. The red-button “deploy” on the left side will become clickable as soon as some changes in the flow editor are made. When the user would like to make the changes operational, clicking this button will restart the application in the new way.

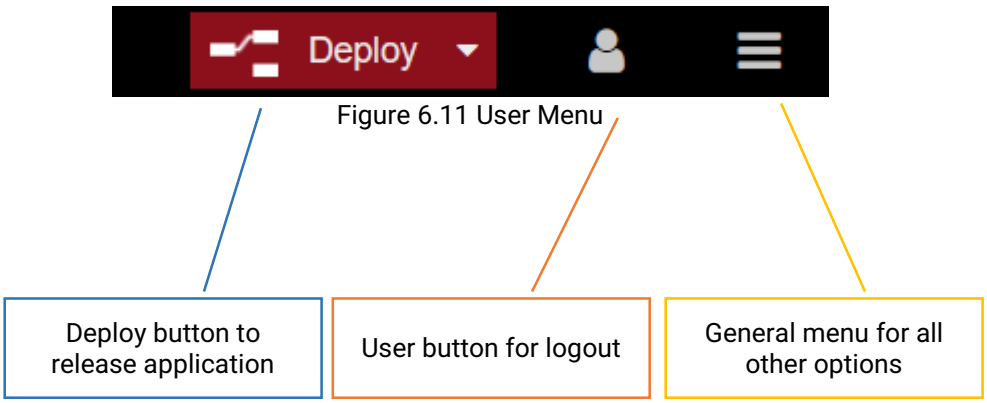
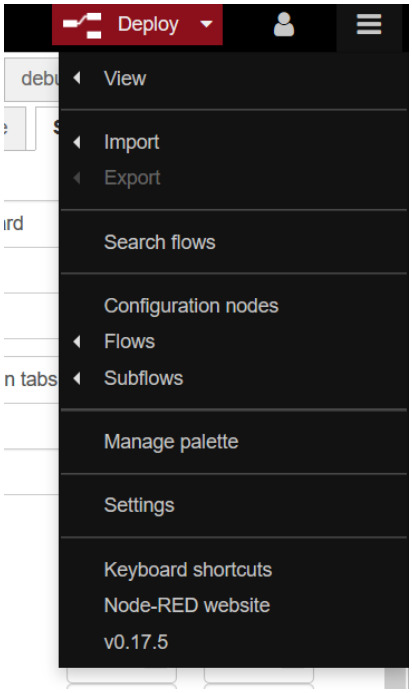


Figure 6.11 User Menu

Clicking on the general menu will open a sub-menu that allows the user to carry out other activities:



For example:

1. **VIEW**: open sidebar, open Dashboard, open debug window
2. **IMPORT**: import new nodes, by copy and paste the code.
3. **EXPORT**: export nodes or library to filesystem or clipboard
4. **CONFIGURATION NODES**: opens a configuration tab into the info area
5. **FLOWS**: allows the user to create, rename or delete flows
6. **SUBFLOWS**: allows the user to create, rename or delete subflows. Subflows can be used inside a normal flow as a Node. Once saved, the subflow will appear inside the Node Palette Menu
7. **MANAGE PALETTE**: allows the user to add Nodes, update Nodes etc.. This is very useful when willing to use third party Nodes
8. **SETTINGS**: allows the user to customize NodeRED

Figure 6.12 Drop-down menu

7 Using Node-RED

This chapter aims to help you to design a simple flow, and manage its related Dashboard.
Node-RED is Node.JS based, written in Javascript. Therefore, advanced implementations require a knowledge of Javascript programming language. Some nodes especially may require the user to input part of their own script inside of it.

7.1 Create a new flow

Creating a new flow with Node-RED is extremely simple and intuitive. If you'd like to remove the default flow set up by ATOP, select with the mouse all Nodes on the flow and then press the <delete> button on your keyboard. All the content of the Node-RED flow workspace will be emptied, and you'll be ready to create a new flow. If, instead, you'd like to create a new flow running in parallel, then left-click on the Menu on the User menu bar, select Flows, and then Add. Figure 7.1 shows the successful creation of a new flow, where the empty flow-workspace will be shown.

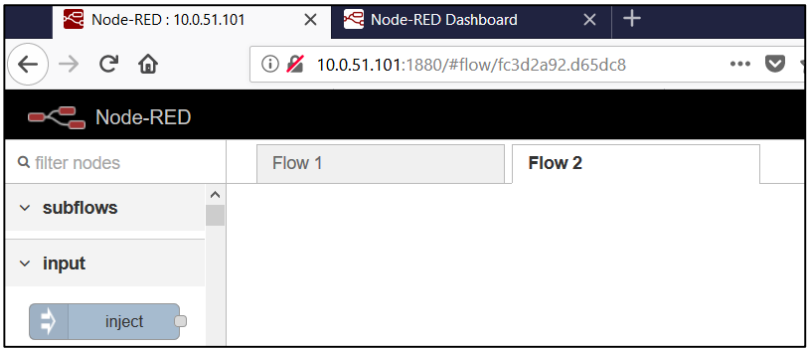


Figure 7.1 New Node-RED flow created

In order to change the flow name, add some descriptive text or temporarily disable the flow, please double-click on the "Flow 2" tab (or the tab of the flow you'd like to edit), as shown in Figure 7.2. Click "Done" to save and exit.

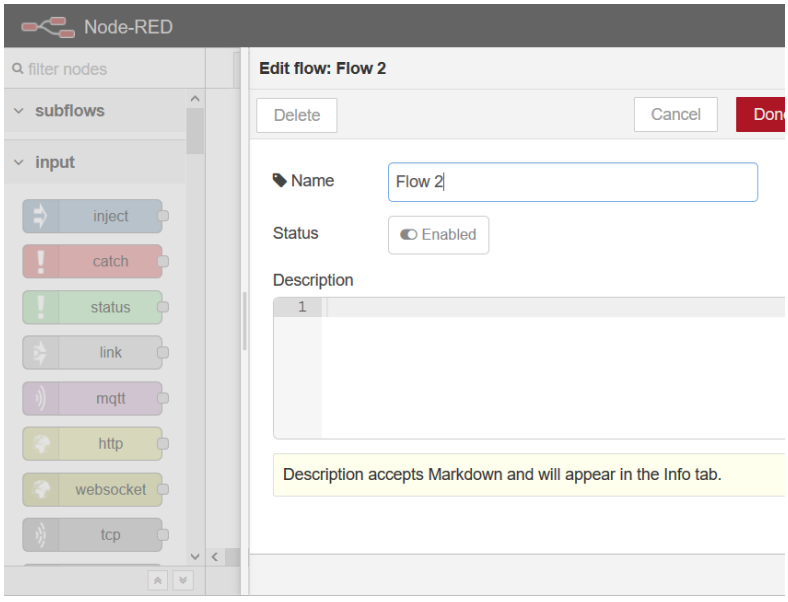


Figure 7.2 Flow options

7.2 Node-RED flow example

In general, Node-RED flows start on Inputs (no matter fixed constants or inputs from hardware, web and so on), have data processed by functions (no matter pre-configured or customized with a Javascript script), and outputs.

The example shown below will generate a simple flow where a random number is generated every second and displayed on a Gauge on the Node-RED dashboard. Some functions like “inject” allows the user to input some constants or, if constants are set-up to be empty, to carry out an action in a specific amount of time that can be set by the user.

In order to design this example, drag from the Node-Palette section to the Node-RED flow workspace the inject node (Input section), the function node (function section) and the Gauge node (dashboard section).

Then, wire the nodes together as per Figure 7.3 by connecting the grey dots on the right of the node (outputs) to the grey dots to the left of the node (inputs).

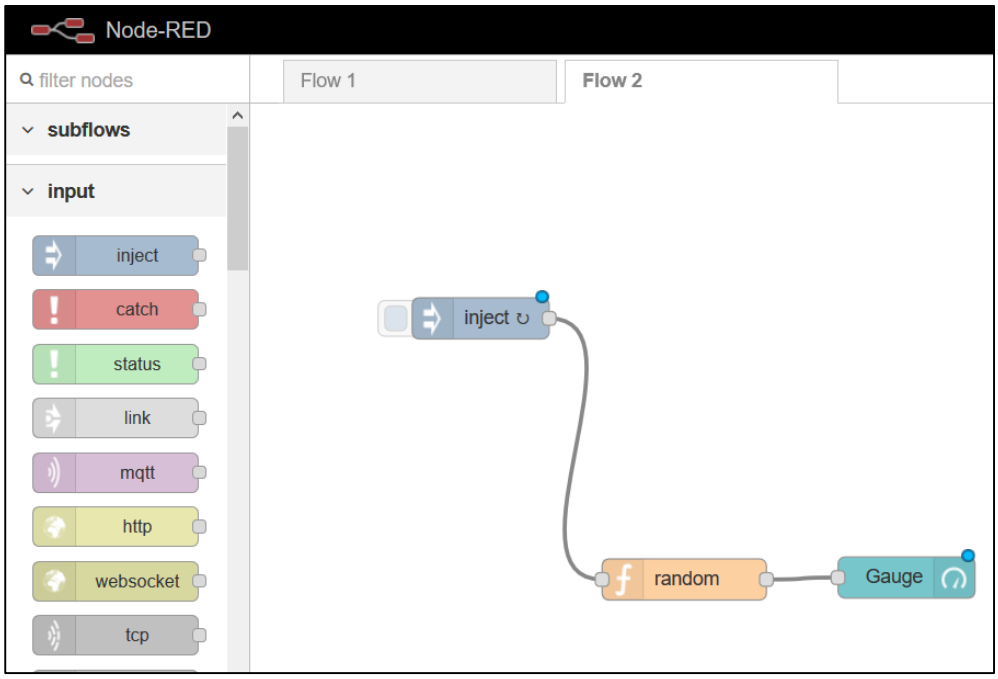


Figure 7.3 Node-RED flow example

Single-clicking a node will show on the information panel section the description of that specific node.
Double-clicking a node, will open up a node-configuration section that is specific to the selected node. Figures Figure 7.4, Figure 7.5 and Figure 7.6 show the different configurations possible for the inject, function and gauge nodes.

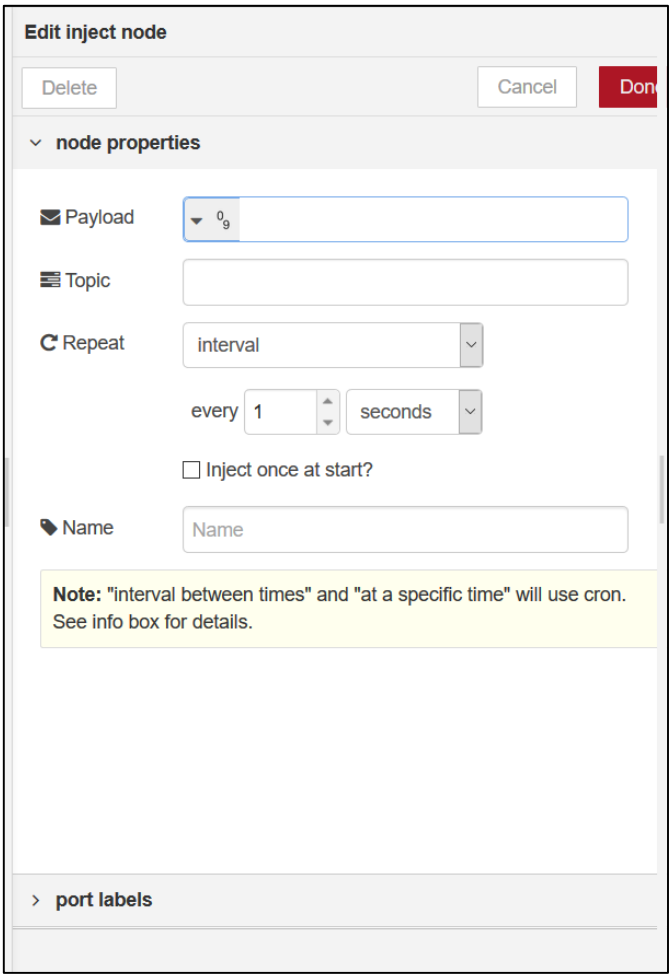


Figure 7.4 Inject node properties

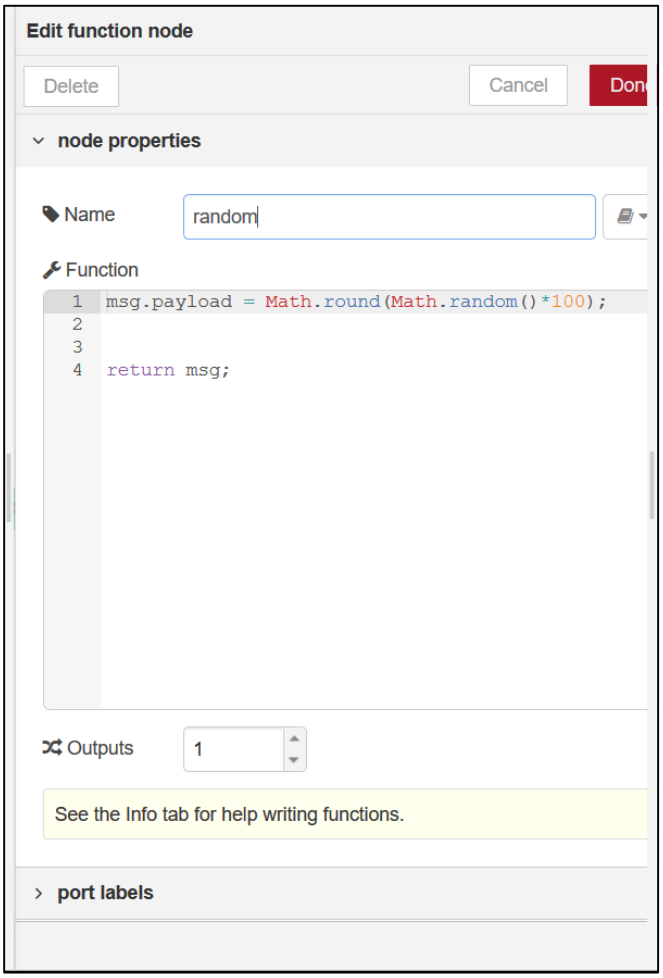


Figure 7.5 Function node properties

Figure 7.4 shows the properties of inject node. This function is usually used in injecting a fixed constant. Here, the example is generating a *null* string every 1 second, and sent to the subsequent node.

Figure 7.5 shows the properties of the function node. This function, one of the most used, doesn't include a specific function, but allows the user to write, in Javascript language, the code for the function itself. The "*name*" field allows the user to define a label of the function, making it easy to remember. The "*function*" field is the function itself, written in Javascript. In this specific example, the function generates a random number and multiplies it times 100. The User can also have a function to generate more than one output. When changing the "Outputs" number, automatically the node will have an additional dot, which another node can be wired to.

Javascript and the explanation on how to write functions for Node-RED is outside the scope of this document. Anyway, a very detailed documentation is accessible online on Node-RED website. Please refer to the following URL: https://nodered.org/docs/writing-functions.html

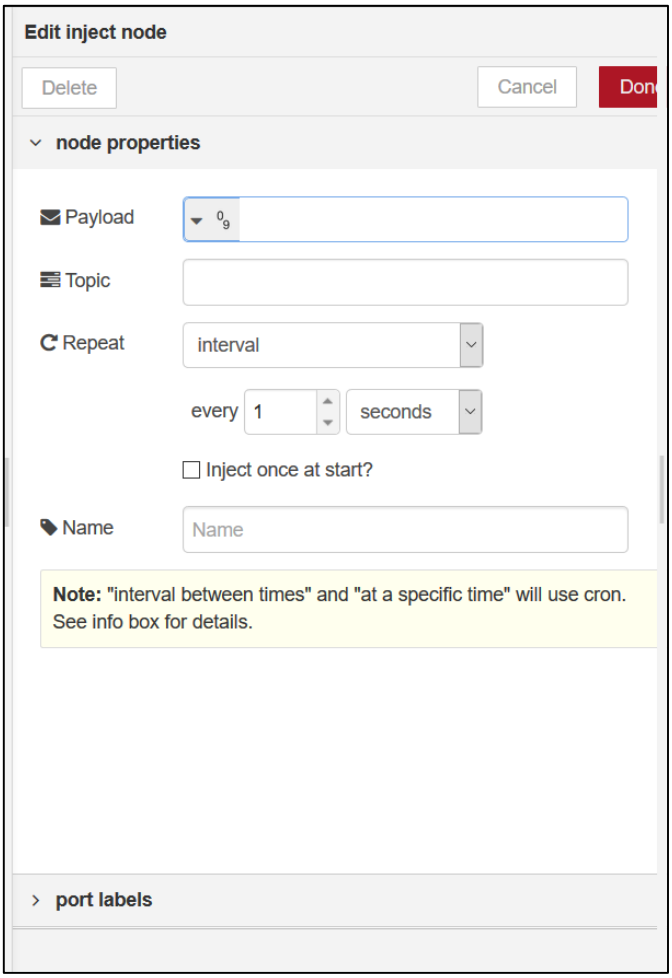


Figure 7.4 Inject node properties

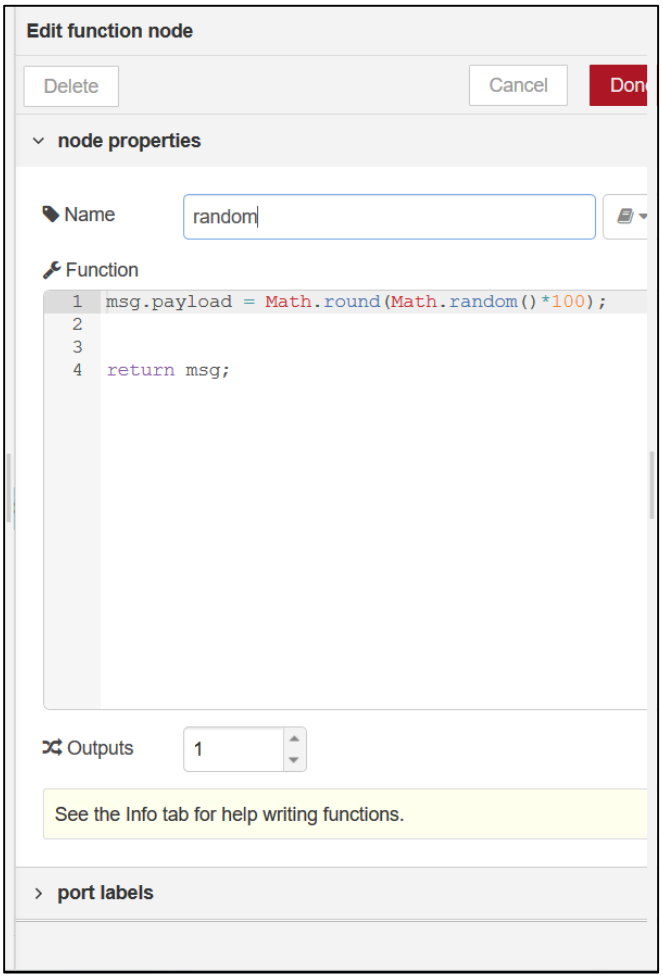


Figure 7.5 Function node properties

Figure 7.6 shows the properties of the Gauge node. This function is one of the many output functions provided by Node-RED in the Node-RED Dashboard. The options allow the user to configure the Group (area of the graphical element inside of the dashboard page), the size, the label, the format of the value shown, the unit of measure (if applicable), the range and the color gradient. The latter, is very useful if it's needed to show ok, alert and alarm values.

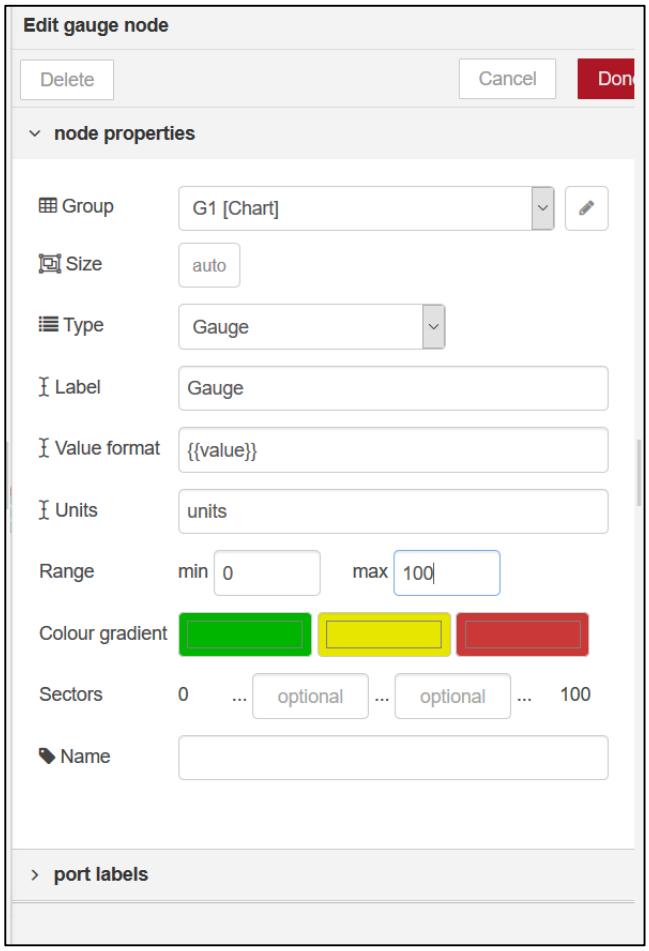


Figure 7.6 Flow options

Once all the parameters are set, click on the “deploy” button in the User Menu space (Figure 6.11). The System will stop existing flows, and run the new application completely. The Node-RED dashboard, if opened, will also refresh. The dashboard of the sample flow described in the previous pages will look approximately like Figure 7.7, with the value refreshing randomly every second.

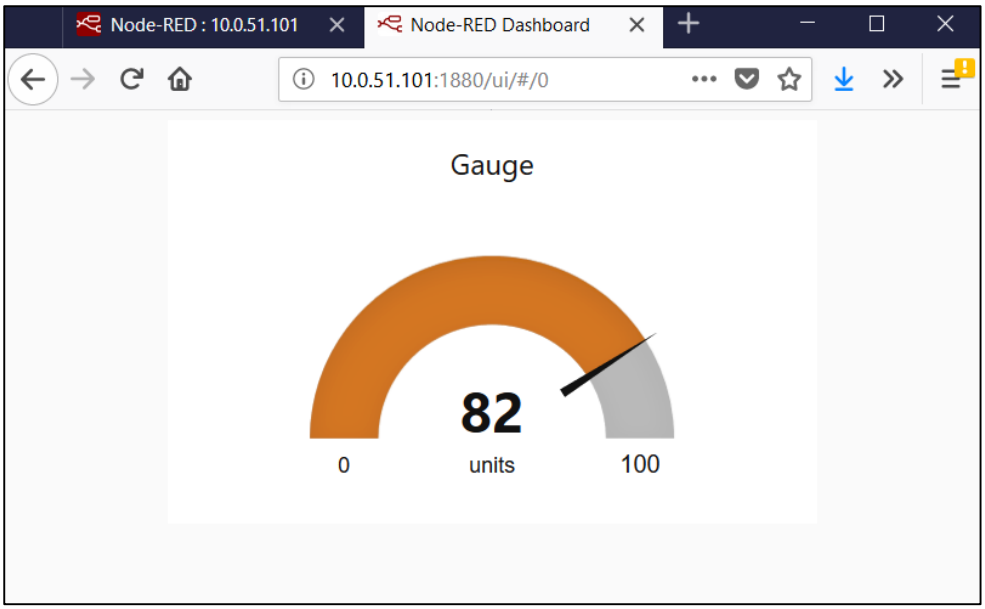


Figure 7.7 Sample Flow dashboard

7.3 Dashboard-specific settings

The dashboard is a powerful UI that can be used for monitoring the application residing on SE59XX-SDK Node-RED. It is not fundamental: the application runs on the device and doesn't have the need to have a Web-Client (such as Desktop or Laptop computer, Smartphone or Tablet) connected to it to work.

When accessing the Node-RED flow editor, on the right hand side of the screen there is a dashboard tab that can be selected. This is divided into three different sub-tabs, called "*Layout*", "*Theme*", "*Site*". Figure 7.8 shows the dashboard settings how they appear on the Node-RED editor once "*dashboard*" tab is selected. In Orange highlight the button that, when clicked, opens a separate tab on the browser displaying the dashboard itself.

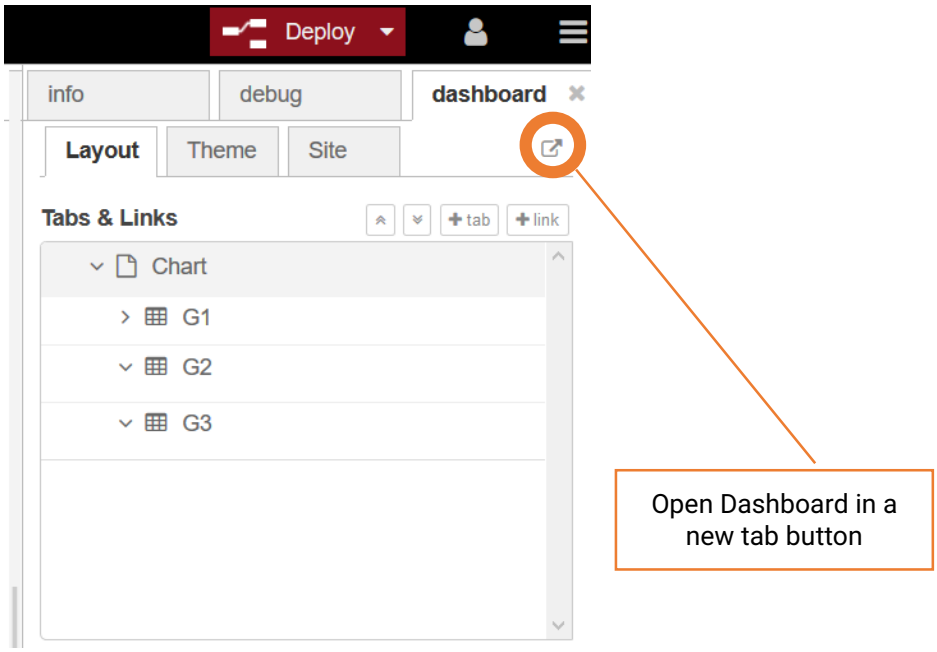


Figure 7.8 Dashboard settings in Node-RED flow editor

The three different sub-tabs have the following meaning:

- **LAYOUT** sub-tab: defines where the Dashboard objects will be located, in a which dashboard tab and in which group of objects. The meaning of tabs and groups is explained in Figure 7.12 below.
- **THEME** sub-tab: allows the user to define the color theme and/or the font of the Dashboard. The different results are shown in
- Figure 7.13 below
- **SITE** sub-tab: allows the user to customize other parameters, such as the webpage title, date formats, etc..

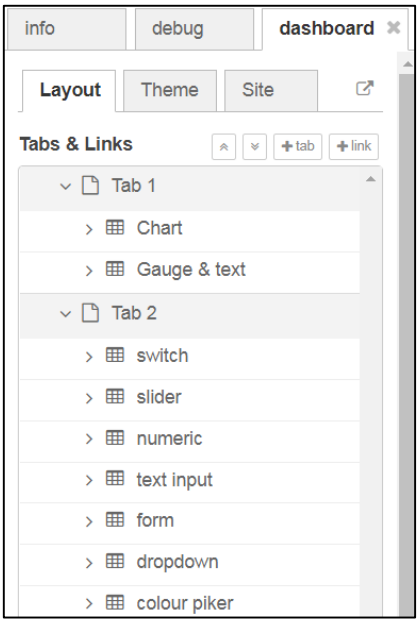


Figure 7.9 Dashboard Layout Settings

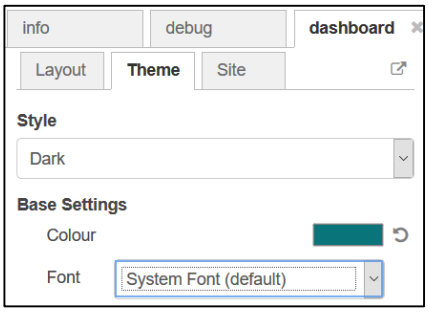


Figure 7.10 Dashboard Theme settings

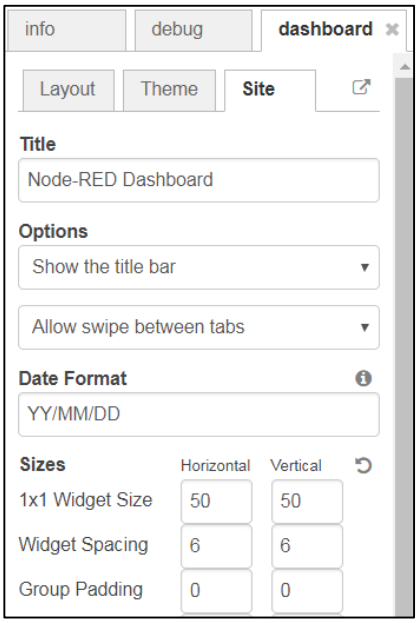


Figure 7.11 Dashboard site settings

7.3.1 Dashboard: layout settings

A **group** is a logical and graphical grouping of similar elements, or of elements that graphically make sense to be put nearby. For example, a dashboard may monitor temperatures from 3 sensors and pressure from three other sensors. It makes sense to have all temperatures grouped together or all pressures grouped together, on the same page. Or instead, pressure and temperature of location A grouped together and so on. This is at discretion of the user and its heavily application-oriented.

A **tab** is a logical and graphical grouping of elements, same as group. The difference with the group is that while the group shows the elements in different "groups" of the same page, when defining different tabs, the webpage displayed on the browser will show only the elements assigned to that specific tab only. Inside a Tab, groups can be defined.

Figure 7.12 shows the same flow designed as example (Figure 7.3), connecting to the "random number" generator function totally 4 different Dashboard output elements. 3 Elements are assigned to Tab 1 (2 items to Group1, 1 item to Group 2) and 1 Element is assigned to Tab 2, Group 1.

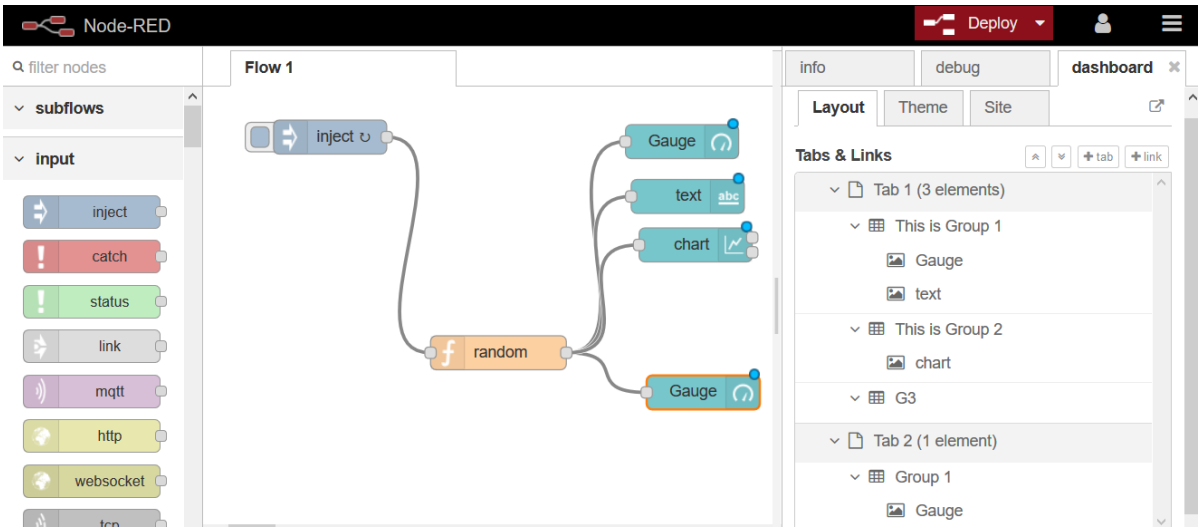


Figure 7.12 Dashboard Groups and tabs Flow example

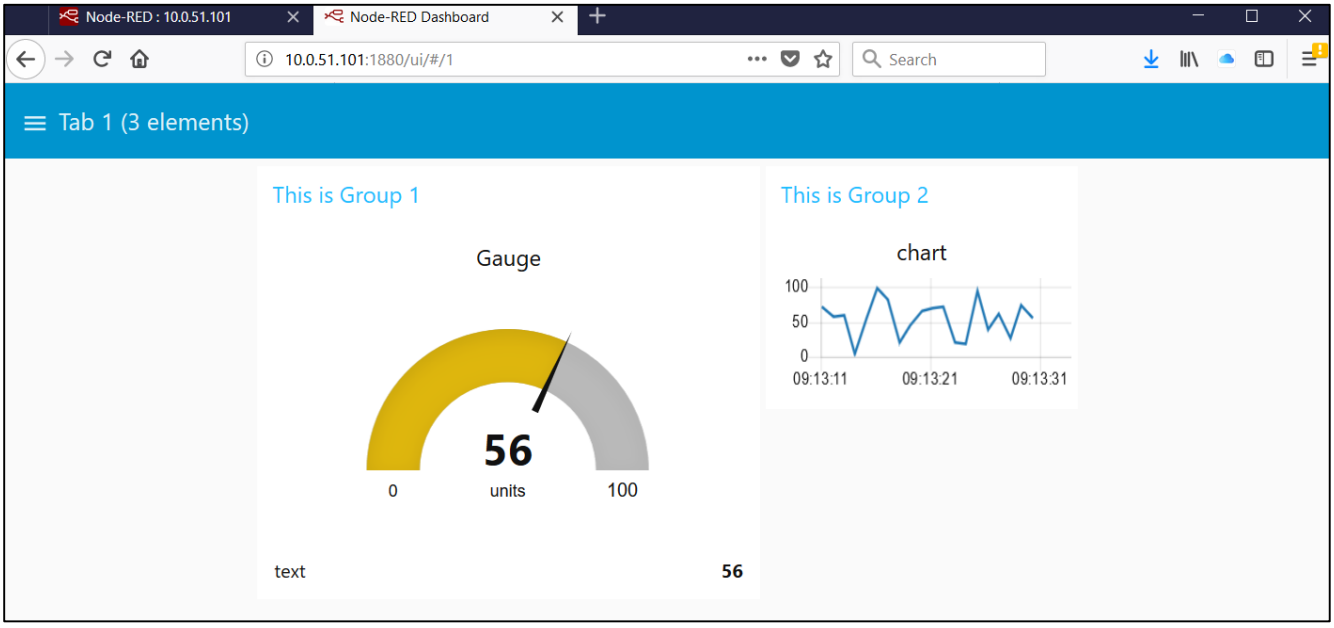


Figure 7.13 Dashboard Groups and tabs Dashboard result on Tab 1

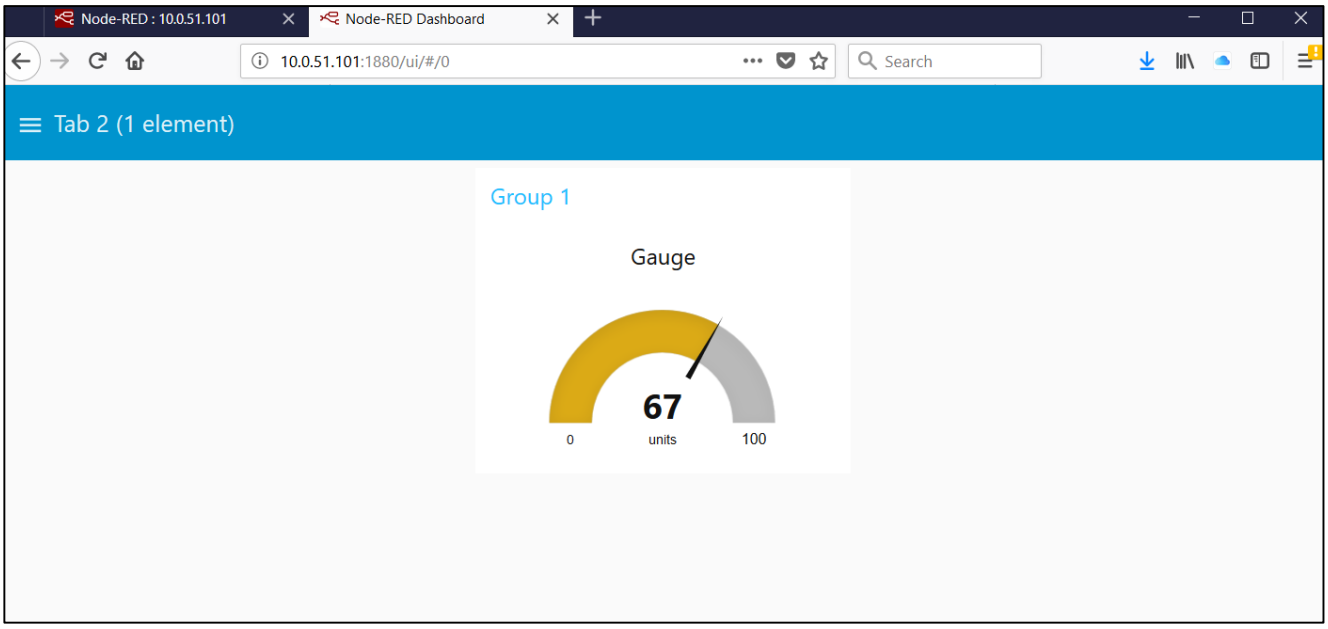


Figure 7.14 Dashboard Groups and tabs Dashboard result on Tab 2

Note: in order to show the tab switching bar on top, it is necessary to set “Show title bar” in Dashboard/Site options as explained in Section 7.3.3 below.

7.3.2 Dashboard: theme settings

It is possible to change the theme of the dashboard. There are three options available:

- Light (default) : white background, and colors as in all dashboard previous examples
- Dark : dark grey background, and colors as in Figure 7.15
- Custom : gives the possibility to customize the appearance settings, including font, colors, etc..

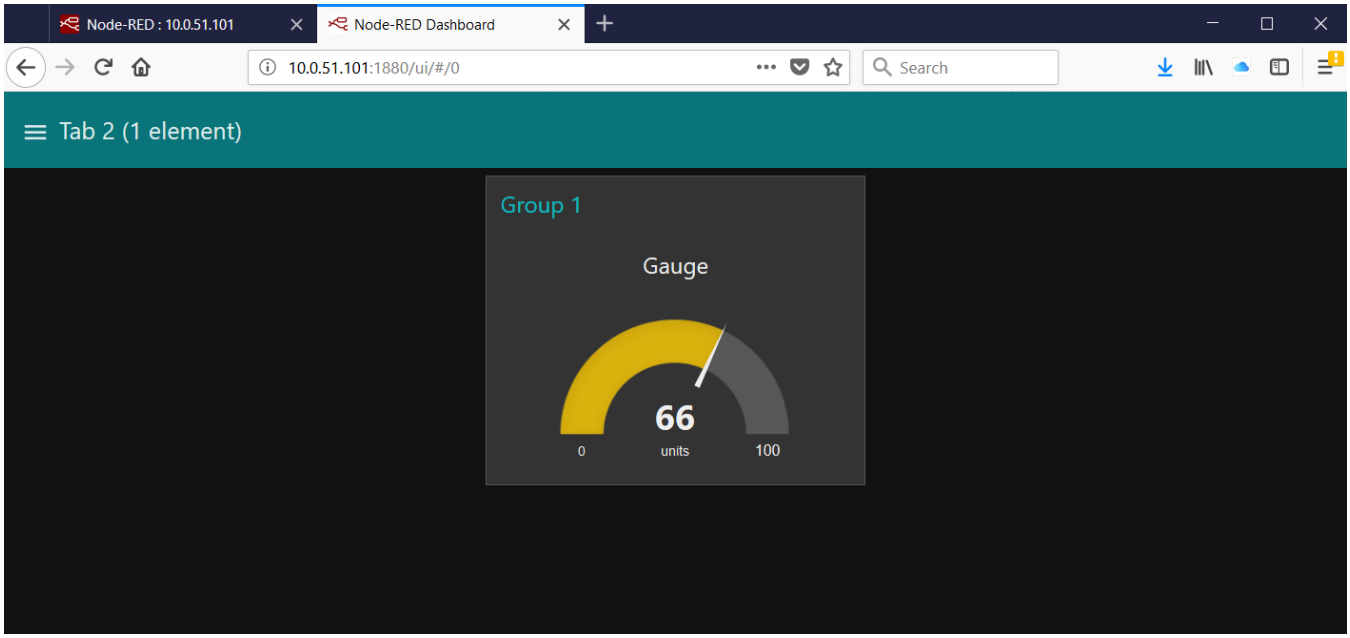


Figure 7.15 Dashboard Dark Theme settings

7.3.3 Dashboard: site settings

The site settings can be used to further customize the Node-RED dashboard

- Title: allows to modify the website title (Replaces “Node-RED Dashboard”, set by default)
- Show/ don’t show title bar: allows to display the title bar that allows the user to switch between tabs. It is recommended to use “show title bar” option for easier user management.
- Allow/ don’t allow swipe between bars: allows the user to switch between tabs from the top menu
- Date format : allows the user to set the regional date format standard.
- Spacing and sizing option: allows the user to specify size and spacing options.

Figure 7.16 below shows an example where the site settings have “ATOP TEST” as Title and hidden title bar.

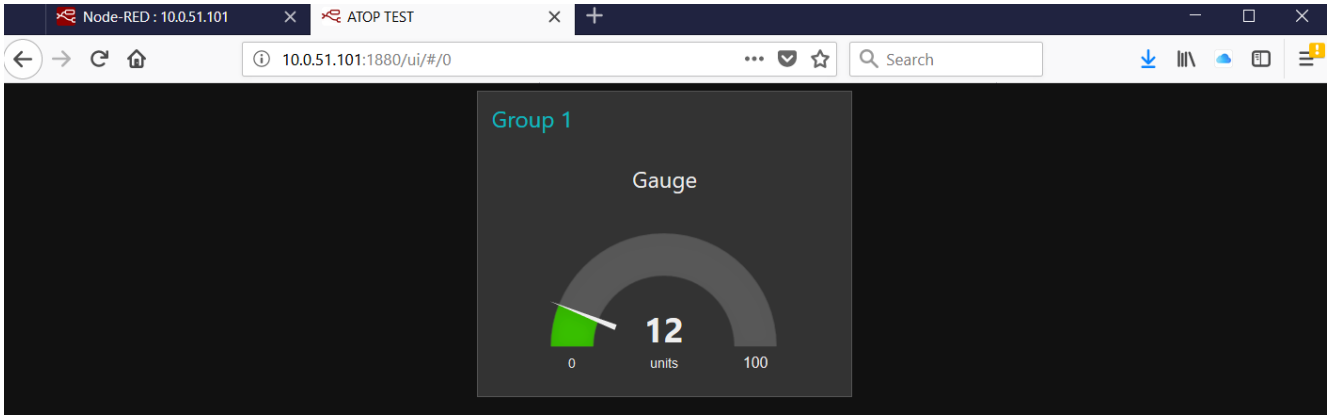


Figure 7.16 Dashboard showing Tab 2 with customized title and hidden title bar

7.4 Dashboard- user inputs

Node-RED allows, along with displaying data elements, also to have data-input elements on the dashboard. These can be wired as desired to the other building blocks inside the flow in order to make it interactive. For example, a switch can be used to enable or disable an automatic algorithm, a slider can be used to make changes in a variable, and so on.

Note: Node-RED has the capability to run embedded applications but ATOP does not recommend do use it for mission-critical process control, automation, utilities and so on. Being a Javascript-based application, the run-time performance is not as good as a binary application running on SE59XX-SDK. Node-RED is very useful to be used for monitoring, but it may be dangerous if deployed in applications where a wrong behavior can put people's live in danger.

Figure 7.17 shows, on the dashboard, the different user inputs supported in Node-RED. The configuration of each single node is outside the scope of this document. If there are questions, please consult documentation on www.nodered.org

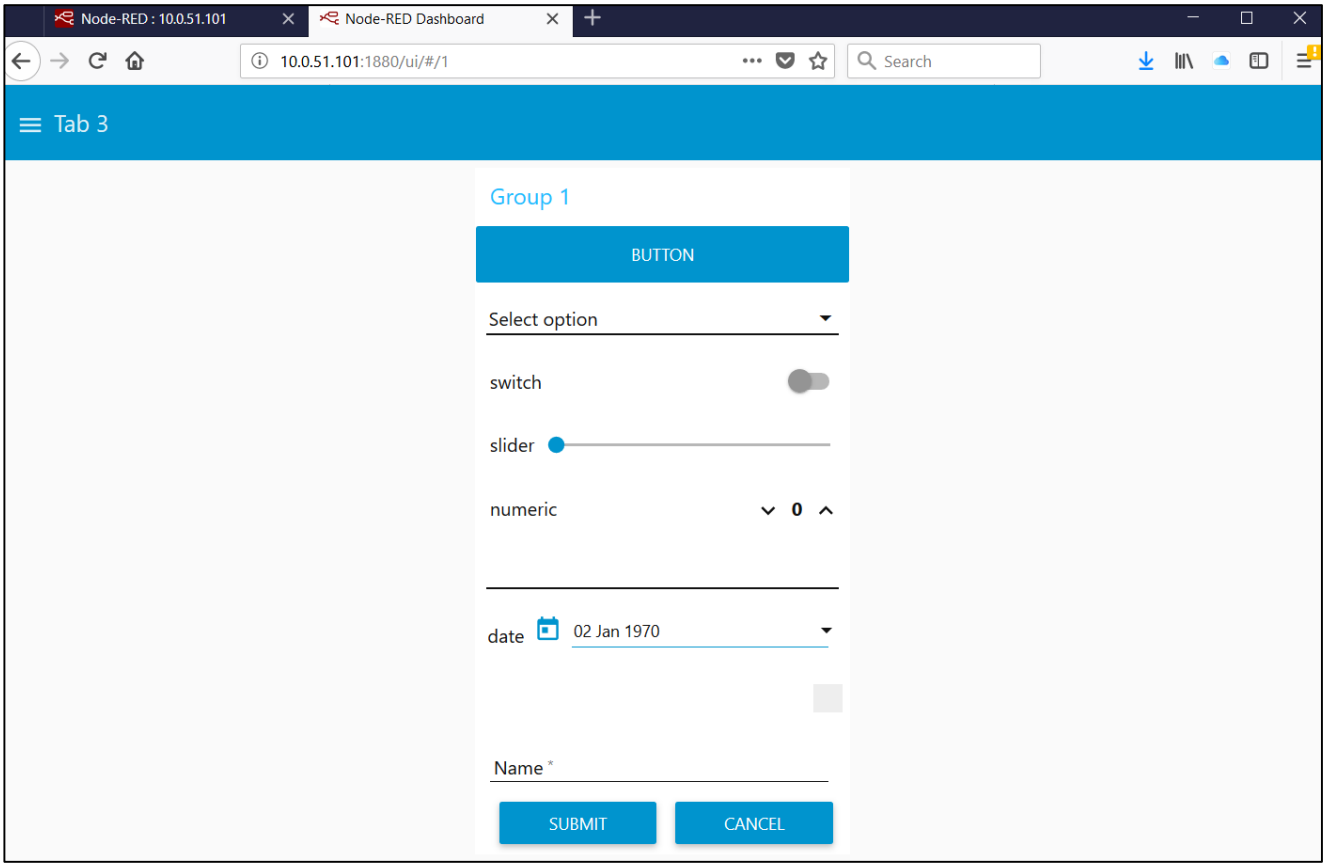


Figure 7.17 Dashboard showing all different available user inputs

7.5 Accessing and controlling ATOP SE59XX Hardware with Node-RED

Using Node-RED as embedded application on SE59XX-SDK Node-RED will allow you to get the best out of ATOP hardware and access all hardware interfaces available on it, no matter Serial, Ethernet, Digital Inputs, Digital Outputs, Buzzer, Relays and so on. In this chapter we will go through the methods to be used to access data.

7.5.1 Configure Serial Port mode

The configuration of theSerial Mode on SE59XX-SDK Node-RED is using the “exec” Node, shown in Figure 7.18. The “exec” Node, basically runs a Linux binary program that is stored in the Filesystem or a Linux command. The “exec” Node is located in the “advanced” section of the Node palette on the left hand side of the screen.

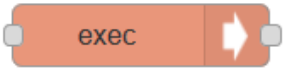


Figure 7.18 Exec Node

The configuration of the “exec” node requires some parameters, as shown in Figure 7.19 below and as explained in the following paragraph.

- **Command**: defines the linux command or the binary application to be run
- **Append “msg.payload”** checkbox: allows the execution of the application or the command to have appended the input wired to the “exec” node. If the checkbox is not checked, then the command will be executed without any input parameters, as it is written
- **Append extra input parameters:** allows the user to input additional constant parameters to the command or to the binary application
- **Output:** allows the user to define whether issue the output only when the command execution is finished or during run-time
- **Timeout:** allows the user to define the execution timeout time in ms before having the process killed.

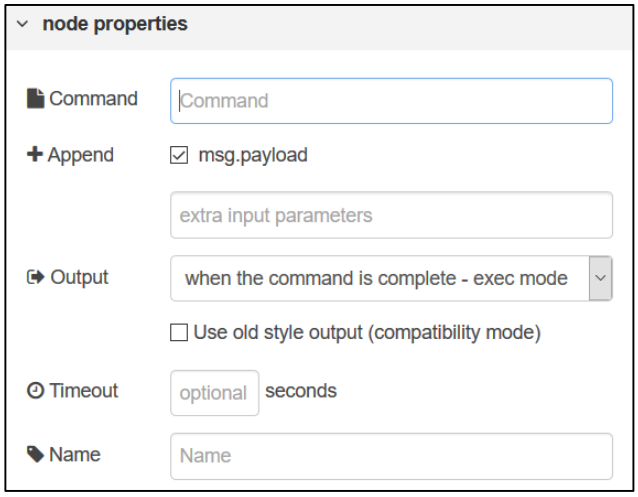


Figure 7.19 Exec Node Configuration Parameters

Use exec Node to configure the COM port Mode. The tables below list device node of COM port for each model.

Table 7.1 SE59XX Programming commands per device node

Device node	Ioctl command	Command Description
ttyATOP0 ~ttyATOPX	0x9000	Configure SE59XX COM port as one of RS232 / RS485 / RS422

Table 7.2 SE59XX ioctl command of COM Port

Ioctl command	**parameter type**	**Value**	**Description**
0x9000	integer	1	Configure to RS232 mode
		2	Configure to RS422 mode
		3	Configure to RS485 mode

7.5.2 Read and Write data to Serial Ports

The nodes shown below in Figure 7.20 are used to read and write from the serial port

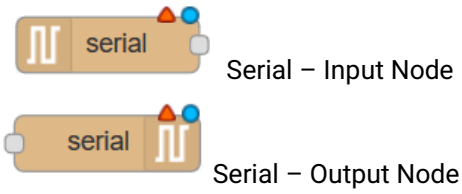


Figure 7.20 Serial Read/Write Nodes

Besides the configuration, shown in Figure 7.21 and Figure 7.23 below, the Serial Input Node will produce an output, and the Serial Input Node will require an input.
Inside the serial port configuration, the default Serial port will automatically be selected (COM1). This will be marked as /dev/ttyATOP0. To select a different serial port, please cross-reference to Table 7.3 below.

Table 7.3 SE59XX device node

Device node	**Major & Minor number**	**Device Type**	**Description**
ttyATOP0	266 0	Character	ATOP COM port 1
ttyATOP1	266 1	Character	ATOP COM port 2
ttyATOP2	266 2	Character	ATOP COM port 3
ttyATOP3	266 3	Character	ATOP COM port 4
…	…	…	…

Figure 7.21 below shows the Serial Read Node configuration options.

- **Serial port**: defines on which serial port Node-RED should listen to. Click on the "*pencil*" icon to open up the window shown in Figure 7.22.
- **Name**: defines the node name, easy way to remember which port/application the node is referring to.

Figure 7.21 Serial Read Node options

Figure 7.22 shows the Serial port configuration options. The meaning of the fields is explained below:

- **Serial port**: Defines which device serial port Node-RED should listen to. Refer to Table 7.3
- **Baud rate**: Defines the baud rate which the sending device is transmitting to SE59XX
- **Data bits**: Defines how many data bits are inside each frame. This should be the same of the one set on the device that is transmitting to SE59XX
- **Parity**: defines whether there's a parity bit to check data consistency
- **Stop Bits**: defines whether there's a stop bit to mark the end of the frame
- **Split input**: defines how the data received should be split in different output messages.
- **And deliver**: defines the data output format
- **Add split character to output messages**: defines whether the split character should be appended to the output message or shouldn't be considered

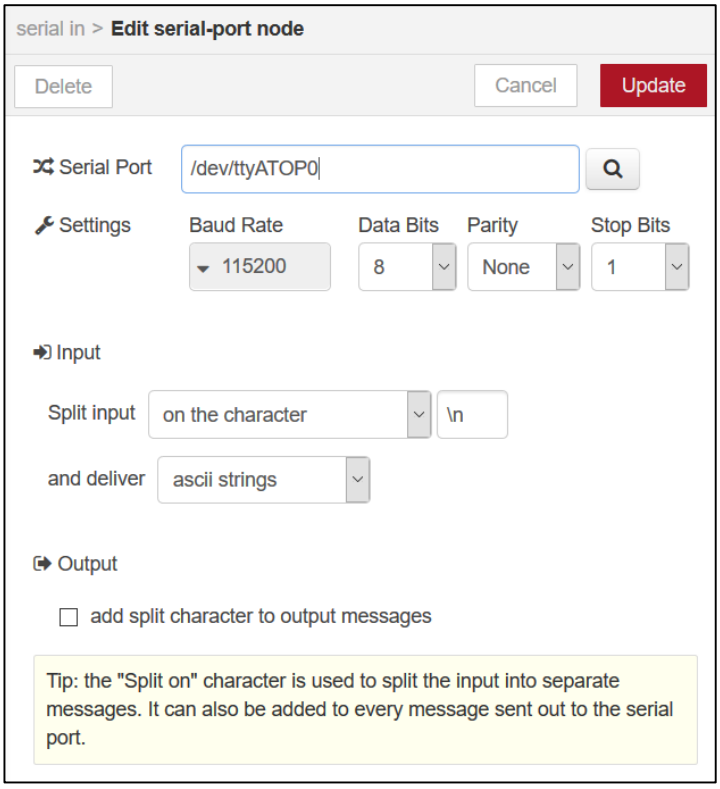


Figure 7.22 Serial Read Node port configuration options

Figure 7.23 below shows the Serial Read Node configuration options.

- **Serial port**: defines on which serial port Node-RED should write data to. Click on the "*pencil*" icon to open up the window shown in Figure 7.22.
- **Name**: defines the node name, easy way to remember which port/application the node is referring to.

Figure 7.23 Serial Write Node options

Figure 7.24 shows the Serial port configuration options. The meaning of the fields is explained below:

- **Serial port**: Defines which device serial port Node-RED should write data to. Refer to Table 7.3
- **Baud rate**: Defines the baud rate which the SE59XX should be transmitting
- **Data bits**: Defines how many data bits are inside each frame. This should be the same of the one set on the device that is receiving data from SE59XX
- **Parity**: defines whether there's a parity bit to check data consistency
- **Stop Bits**: defines whether there's a stop bit to mark the end of the frame
- **Split input**: defines how the data received by the input node should be split in different output messages.
- **And deliver**: defines the data output format
- **Add split character to output messages**: defines whether the split character should be appended to the serial write message or shouldn't be considered

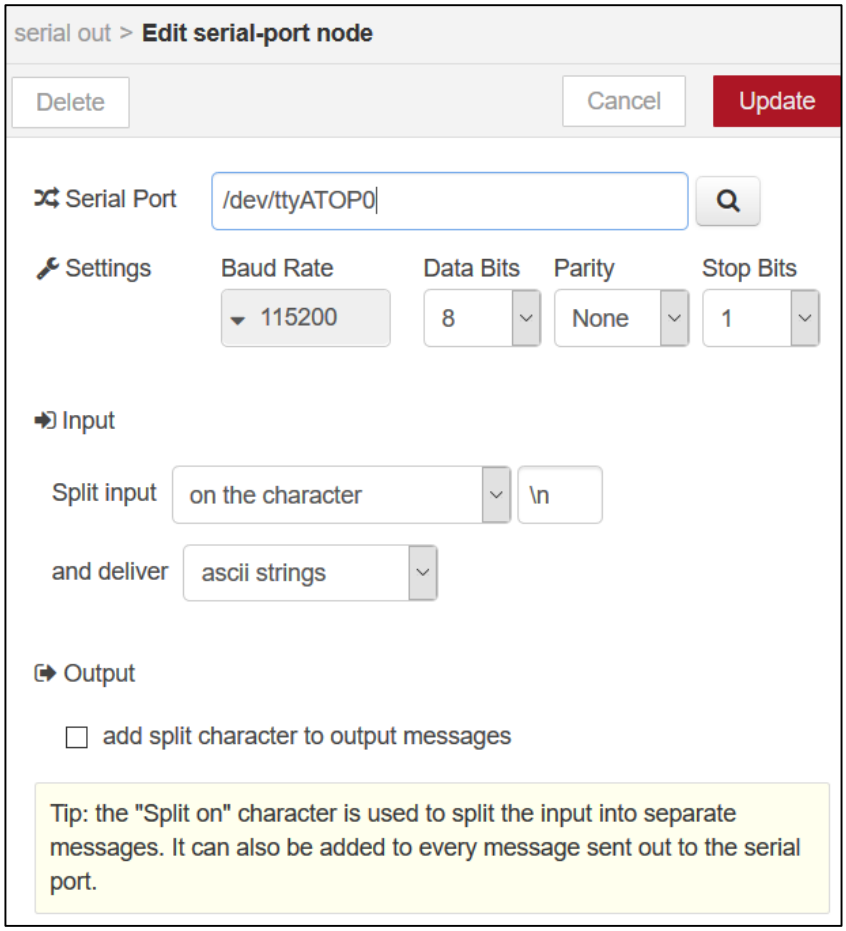


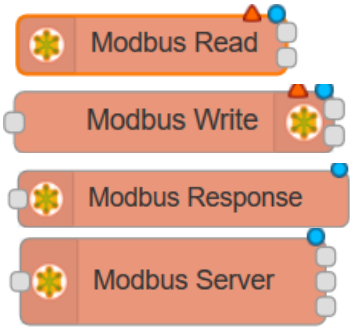
Figure 7.24 Serial Write Node port configuration options

7.5.3 Modbus TCP/RTU/ASCII

The nodes shown below in Figure 7.25 are used to read and write from the serial port using Modbus RTU/ASCII. There is one node (Modbus Server) specifically designed to open a server on SE59XX-SDK Node-RED, while "Modbus Read", "Modbus Write" and "Modbus response" nodes can be used either in connection with "Modbus Server" node or separately for data polling or command writing (in Modbus Master mode).

All Modbus Nodes support Modbus TCP, RTU, ASCII and can be used either to control serial ports or to use the device's LAN ports. In addition to the below mentioned Nodes, Node-RED provides additional Modbus functions, very useful when carrying out multiple pollings or multiple writings.

The examples below listed, and the configuration is not inside the scope of this document. If there are any questions or doubt, please review Node-RED official documentation on www.nodered.org



Modbus – Read Node

Modbus – Output Node

Modbus – Write Node

Modbus – Server Node

Figure 7.25 Main Modbus Nodes

7.5.4 Read data to Serial Ports using Modbus RTU/ASCII

In order to read data from Serial ports using Modbus RTU/ASCII, please set-up the flow as shown in below Figure 7.26. This flow uses 1 "Modbus read" node, 1 "Modbus Response Node" and 1 "Dashboard text output" node, that here is used to show the data on the screen.

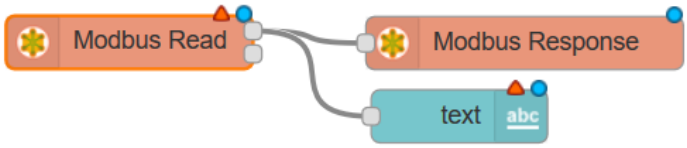


Figure 7.26 Modbus Serial Read example flow

The Modbus Read node has the following arguments, as per Figure 7.27:

- **Name**: arbitrary name
- **FC**: Modbus read function number to use (drop-down menu) – please see Modbus specifications
- **Address**: Modbus ID from which the information should be read
- **Quantity**: Quantity of coils or words that should be read starting from address mentioned
- **Poll-rate**: Frequency of data polling. This can be either in milliseconds, seconds, minutes, hours
- **Show activities flag**: shows the progress of the function
- **Show errors flag**: shows the polling errors, if any.
- **Server**: allows the user to choose which interface should be used for data polling. The default value is COM1 (/dev/ttyATOP0). By clicking on the arrow icon, an additional window will open up, as per Figure 7.28. The parameters of this window are the following
    - **Type:** Defines whether use Modbus over Serial (RTU or ASCII) or over Ethernet (TCP). The Serial Expert mode allows a more detailed parameter settings explanation.
    - **Serial port**: Defines which device serial port Node-RED should write data to. Refer to Table 7.3
    - **Serial type (only if Type is "Serial" or "Serial Expert")**: Defines whether use Modbus RTU or ASCII
    - **Baud rate**: Defines the baud rate which the SE59XX should be transmitting
    - **Data bits ("Serial Expert" only)**: Defines how many data bits are inside each frame. This should be the same of the one set on the device that is transmitting to SE59XX
    - **Stop Bits ("Serial Expert" only)**: defines whether there's a stop bit to mark the end of the frame
    - **Parity**: defines whether there's a parity bit to check data consistency
    - **Timeout**: Defines after how many millisecond the system should generate an error
    - **Reconnect timeout:** defines time to wait on reconnect before making next polling
    - **Serial Connection delay** (default 500 ms) - time to delay first command sending after reconnect

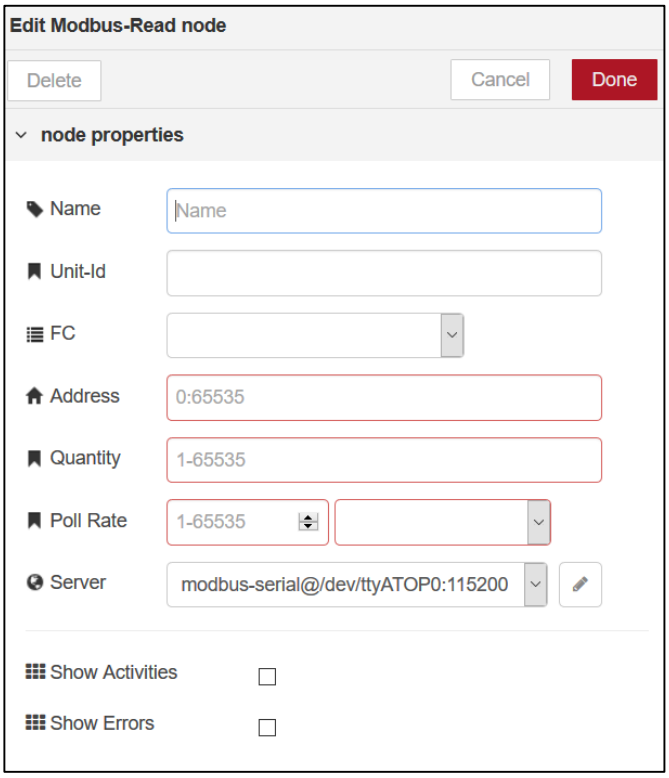


Figure 7.27 Modbus Read Node Settings

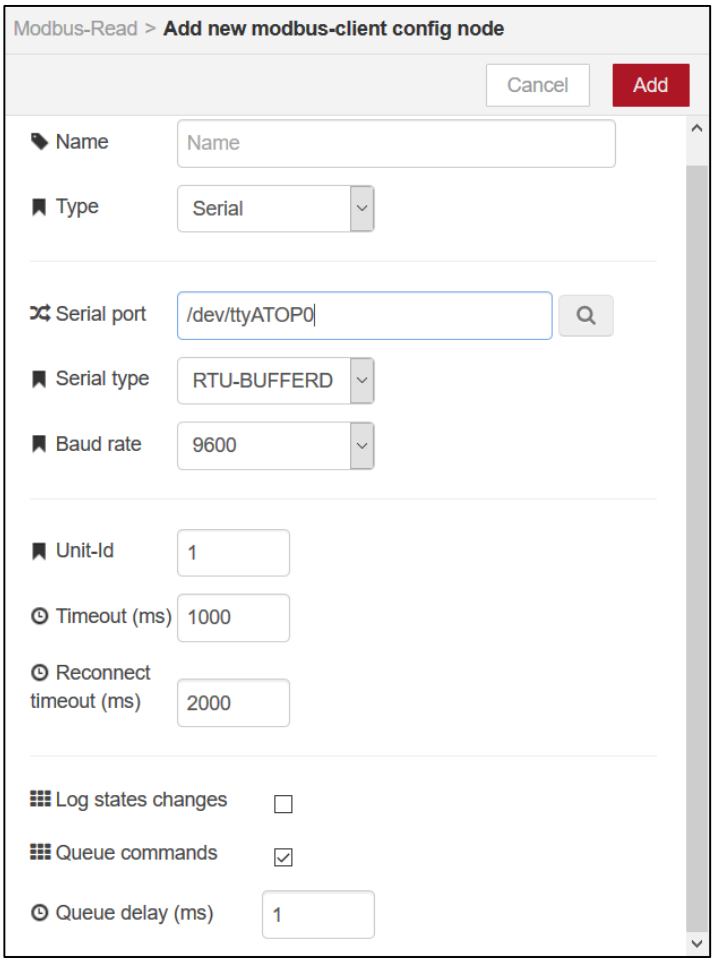


Figure 7.28 Modbus Settings – RTU/TCP/ASCII etc..

Please also set up the Modbus-response node as shown in Figure 7.23, defining only the name, for easy remembering what kind of polling it refers to.

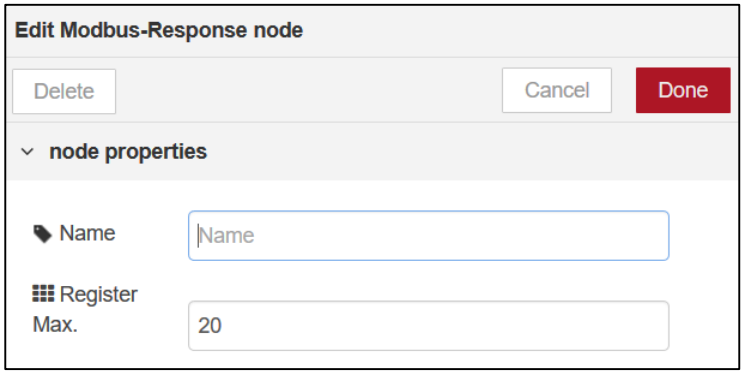


Figure 7.29 Modbus RTU/ASCII Read Node Settings

7.5.5 Read data from Ethernet ports using Modbus TCP

The way Modbus TCP should be configured is very similar to set up of Modbus RTU/ASCII. The only difference lies in the configuration of the "Server settings" located in Figure 7.28. Please select Type as "TCP". When this is done, the possible configuration parameters underneath change to what is shown in Figure 7.30. The parameters are explained as follows:

- **Type:** Defines whether use Modbus over Serial (RTU or ASCII) or over Ethernet (TCP). Select TCP
- **Host**: Defines the Modbus Server IP Address
- **Port (default 502)**: Defines the TCP port used for Modbus TCP communication (usually 502)
- **Timeout, Reconnect timeout**: same as Chapter 7.5.4 above

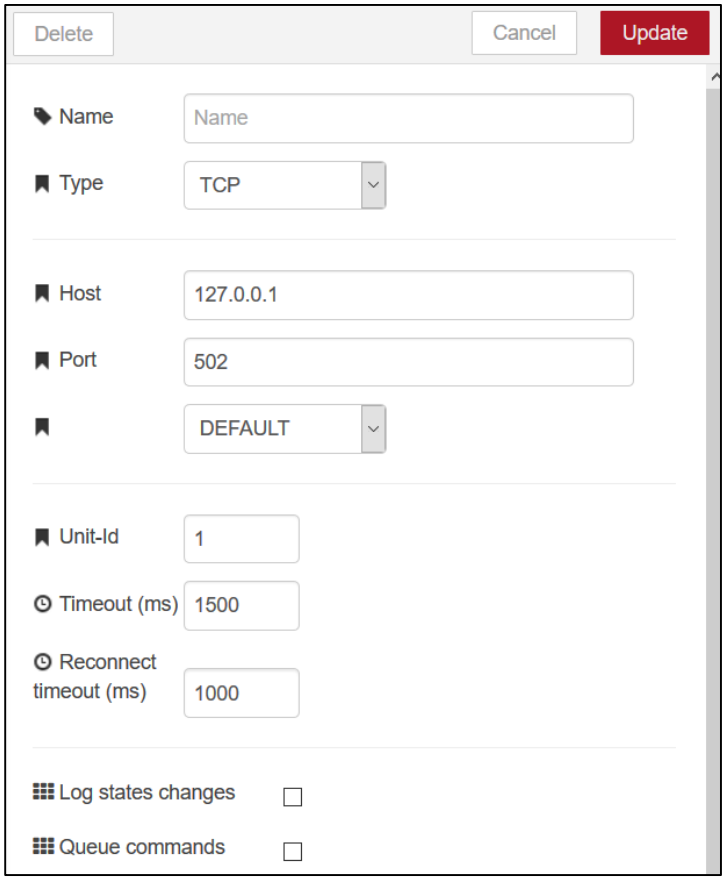


Figure 7.30 Modbus TCP Read Node Settings

7.5.6 Write data using Modbus TCP/RTU/ASCII

The functioning of Modbus Write function is very similar to the Modbus read function previously explained in Chapters 7.5.4 and 7.5.5. Please follow the exact same proceedings for setting it up. The only difference lies in the FC (Function Code) drop-down menu, that will show different Modbus write function numbers. Figure 7.31 below shows the configuration options of Modbus-Write node.

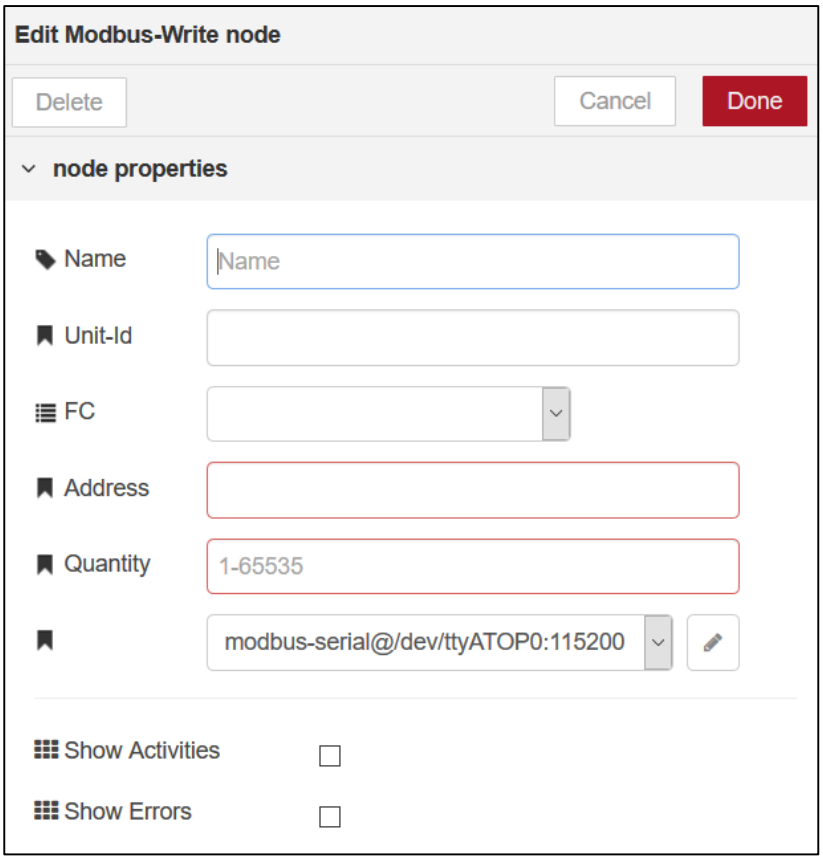


Figure 7.31 Modbus TCP/RTU/ASCII Write Node Settings

7.5.7 Acting as a passive Modbus TCP/RTU/ASCII Slave/Server

SE59XX-SDK Node-RED can also act as a Modbus TCP Server. Please fill in the Modbus Server as below, in order to activate it. This will work in combination with other Modbus nodes

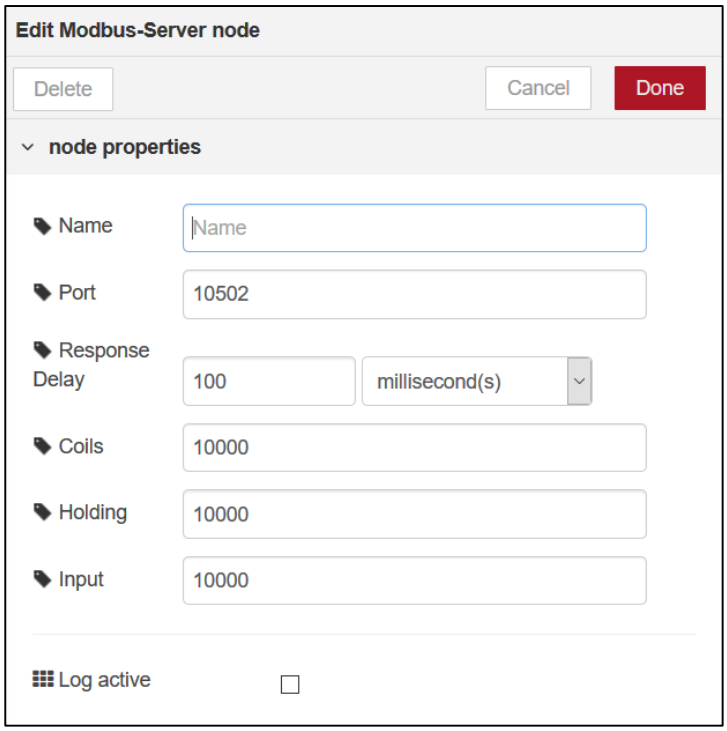


Figure 7.32 Modbus TCP Server Settings

7.5.8 Access other interfaces

The access of other interfaces on SE59XX-SDK Node-RED is using only the “exec” Node, shown in below **Error! Reference source not found.**. The “exec” Node, basically runs a Linux binary program that is stored in the Filesystem or a Linux command. The “exec” Node is located in the “advanced” section of the Node palette on the left hand side of the screen. The detailed explanation on the way the “exec” Node works, is described in Chapter 7.5.1 above.

7.5.8.1 Buzzer

There is one Buzzer in each SE59XX device. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 7.4 Sample program for Buzzer

Command	Description	Attributes
buzzer	A sample program to use the device’s Buzzer.	*on* or *off*

7.5.8.2 Turn the LEDs on or off

Table 7.5 Sample program for LEDs

Command	Description	Attributes
alarmLed	A sample program to use the device’s Alarm (RED) LED	*on* or *off*
runLed	A sample program to use the device’s Run (GREEN) LED	*on* or *off*

There is an LCM in SE5908 and SE5916. The sample program is available in the software/atop_application/utils/atop_sdk folder. Since the application is very strictly customer-dependent, there's no standard Binary application to manage the LCM. It is suggested to use the following sample program as a reference and compile it. For this specific issue, a deep programming knowledge is needed. Users can refer to SE59XX-SDK user manual.

Table 7.6 Sample program for LCM

File Name	Description
lcm_test.c	A sample program to use the device's LCM.

All SE59XX hardware platforms have a reset button. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 7.7 Sample program for Reset Button

File Name	Description
button.c	A sample program to use the device's reset button.

7.5.8.3 Digital Inputs

There are 2 Digital inputs on SE5901B-IO:

Table 7.8 Sample program for Digital Input

Command	Description	Attributes
di_test	A sample program to use the device's Digital Inputs.	0 or 1 (read DI0/ DI1)

7.5.8.4 Digital Outputs

There are 2 Digital Outputs on SE5901B-IO.

Table 7.9 Sample program for Digital Output

Command	Description	Attributes
do_test	A sample program to use the device's Digital Outputs.	0 or 1 (write DI0/ DI1) 0 or 1 (on, off)

8 Global Nodes list

Besides web notes, ATOP SE59XX-SDK Node-RED has pre-installed the following Global nodes

```
+-- node-red@0.17.5
| +-- basic-auth@1.1.0
| +-- UNMET OPTIONAL DEPENDENCY bcrypt@~1.0.1
| +-- bcryptjs@2.4.3
| +-- body-parser@1.17.2
| | +-- bytes@2.4.0
| | +-- content-type@1.0.4
| | +-- debug@2.6.7
| | | `-- ms@2.0.0
| | +-- depd@1.1.1
| | +-- http-errors@1.6.2
| | | +-- depd@1.1.1 deduped
| | | +-- inherits@2.0.3 deduped
| | | +-- setprototypeof@1.0.3 deduped
| | | `-- statuses@1.3.1 deduped
| | +-- iconv-lite@0.4.15
| | +-- on-finished@2.3.0
| | | `-- ee-first@1.1.1
| | +-- qs@6.4.0
| | +-- raw-body@2.2.0 deduped
| | `-- type-is@1.6.15
| |   +-- media-typer@0.3.0 deduped
| |   `-- mime-types@2.1.17
| |     `-- mime-db@1.30.0
| +-- cheerio@0.22.0
| | +-- css-select@1.2.0
| | | +-- boolbase@1.0.0
| | | +-- css-what@2.1.0
| | | +-- domutils@1.5.1
| | | | +-- dom-serializer@0.1.0 deduped
| | | | `-- domelementtype@1.3.0 deduped
| | | `-- nth-check@1.0.1
| | |   `-- boolbase@1.0.0 deduped
| | +-- dom-serializer@0.1.0
| | | +-- domelementtype@1.1.3
| | | `-- entities@1.1.1 deduped
| | +-- entities@1.1.1
| | +-- htmlparser2@3.9.2
| | | +-- domelementtype@1.3.0
| | | +-- domhandler@2.4.1
| | | | `-- domelementtype@1.3.0 deduped
| | | +-- domutils@1.5.1 deduped
| | | +-- entities@1.1.1 deduped
| | | +-- inherits@2.0.3 deduped
| | | `-- readable-stream@2.3.3 deduped
| | +-- lodash.assignin@4.2.0
| | +-- lodash.bind@4.2.1
| | +-- lodash.defaults@4.2.0
| | +-- lodash.filter@4.6.0
| | +-- lodash.flatten@4.4.0
| | +-- lodash.foreach@4.5.0
| | +-- lodash.map@4.6.0
| | +-- lodash.merge@4.6.0
| | +-- lodash.pick@4.4.0
| | +-- lodash.reduce@4.6.0
| | +-- lodash.reject@4.6.0
| | `-- lodash.some@4.6.0
| +-- clone@2.1.1
| +-- cookie@0.3.1
```

```
| +-- cookie-parser@1.4.3
| | +-- cookie@0.3.1 deduped
| | `-- cookie-signature@1.0.6
| +-- cors@2.8.3
| | +-- object-assign@4.1.1
| | `-- vary@1.1.2
| +-- cron@1.2.1
| | `-- moment-timezone@0.5.13
| |   `-- moment@2.18.1
| +-- express@4.15.3
| | +-- accepts@1.3.4
| | | +-- mime-types@2.1.17 deduped
| | | `-- negotiator@0.6.1
| | +-- array-flatten@1.1.1
| | +-- content-disposition@0.5.2
| | +-- content-type@1.0.4 deduped
| | +-- cookie@0.3.1 deduped
| | +-- cookie-signature@1.0.6 deduped
| | +-- debug@2.6.7
| | | `-- ms@2.0.0 deduped
| | +-- depd@1.1.1 deduped
| | +-- encodeurl@1.0.1
| | +-- escape-html@1.0.3
| | +-- etag@1.8.1
| | +-- finalhandler@1.0.6
| | | +-- debug@2.6.9 deduped
| | | +-- encodeurl@1.0.1 deduped
| | | +-- escape-html@1.0.3 deduped
| | | +-- on-finished@2.3.0 deduped
| | | +-- parseurl@1.3.2 deduped
| | | +-- statuses@1.3.1 deduped
| | | `-- unpipe@1.0.0 deduped
| | +-- fresh@0.5.0
| | +-- merge-descriptors@1.0.1
| | +-- methods@1.1.2
| | +-- on-finished@2.3.0 deduped
| | +-- parseurl@1.3.2
| | +-- path-to-regexp@0.1.7
| | +-- proxy-addr@1.1.5
| | | +-- forwarded@0.1.2
| | | `-- ipaddr.js@1.4.0
| | +-- qs@6.4.0
| | +-- range-parser@1.2.0
| | +-- send@0.15.3
| | | +-- debug@2.6.7
| | | | `-- ms@2.0.0 deduped
| | | +-- depd@1.1.1 deduped
| | | +-- destroy@1.0.4
| | | +-- encodeurl@1.0.1 deduped
| | | +-- escape-html@1.0.3 deduped
| | | +-- etag@1.8.1 deduped
| | | +-- fresh@0.5.0 deduped
| | | +-- http-errors@1.6.2 deduped
| | | +-- mime@1.3.4
| | | +-- ms@2.0.0 deduped
| | | +-- on-finished@2.3.0 deduped
| | | +-- range-parser@1.2.0 deduped
| | | `-- statuses@1.3.1 deduped
| | +-- serve-static@1.12.3
| | | +-- encodeurl@1.0.1 deduped
| | | +-- escape-html@1.0.3 deduped
| | | +-- parseurl@1.3.2 deduped
| | | `-- send@0.15.3 deduped
| | +-- setprototypeof@1.0.3
| | +-- statuses@1.3.1
| | +-- type-is@1.6.15 deduped
```

```
| | +-- utils-merge@1.0.0
| | `-- vary@1.1.2 deduped
| +-- express-session@1.15.2
| | +-- cookie@0.3.1 deduped
| | +-- cookie-signature@1.0.6 deduped
| | +-- crc@3.4.4
| | +-- debug@2.6.3
| | | `-- ms@0.7.2
| | +-- depd@1.1.1 deduped
| | +-- on-headers@1.0.1 deduped
| | +-- parseurl@1.3.2 deduped
| | +-- uid-safe@2.1.5
| | | `-- random-bytes@1.0.0
| | `-- utils-merge@1.0.0 deduped
| +-- follow-redirects@1.2.4
| | `-- debug@2.6.9
| |   `-- ms@2.0.0 deduped
| +-- fs-extra@1.0.0
| | +-- graceful-fs@4.1.11
| | +-- jsonfile@2.4.0
| | | `-- graceful-fs@4.1.11 deduped
| | `-- klaw@1.3.1
| |   `-- graceful-fs@4.1.11 deduped
| +-- fs.notify@0.0.4
| | +-- async@0.1.22
| | `-- retry@0.6.1
| +-- hash-sum@1.0.2
| +-- i18next@1.10.6
| | +-- cookies@0.7.1
| | | +-- depd@1.1.1 deduped
| | | `-- keygrip@1.0.2
| | +-- i18next-client@1.10.3
| | `-- json5@0.2.0
| +-- is-utf8@0.2.1
| +-- js-yaml@3.8.4
| | +-- argparse@1.0.9
| | | `-- sprintf-js@1.0.3
| | `-- esprima@3.1.3
| +-- json-stringify-safe@5.0.1
| +-- jsonata@1.2.6
| +-- media-typer@0.3.0
| +-- mqtt@2.9.0
| | +-- commist@1.0.0
| | | +-- leven@1.0.2
| | | `-- minimist@1.2.0
| | +-- concat-stream@1.6.0
| | | +-- inherits@2.0.3 deduped
| | | +-- readable-stream@2.3.3 deduped
| | | `-- typedarray@0.0.6
| | +-- end-of-stream@1.4.0
| | | `-- once@1.4.0
| | |   `-- wrappy@1.0.2
| | +-- help-me@1.1.0
| | | +-- callback-stream@1.1.0
| | | | +-- inherits@2.0.3 deduped
| | | | `-- readable-stream@2.3.3 deduped
| | | +-- glob-stream@6.1.0
| | | | +-- extend@3.0.1 deduped
| | | | +-- glob@7.1.2
| | | | | +-- fs.realpath@1.0.0
| | | | | +-- inflight@1.0.6
| | | | | | +-- once@1.4.0 deduped
| | | | | | `-- wrappy@1.0.2 deduped
| | | | | +-- inherits@2.0.3 deduped
| | | | | +-- minimatch@3.0.4
| | | | | | `-- brace-expansion@1.1.8
```

```
| | | | | |   +-- balanced-match@1.0.0
| | | | | |   `-- concat-map@0.0.1
| | | | | +-- once@1.4.0 deduped
| | | | | `-- path-is-absolute@1.0.1
| | | | +-- glob-parent@3.1.0
| | | | | +-- is-glob@3.1.0
| | | | | | `-- is-extglob@2.1.1
| | | | | `-- path-dirname@1.0.2
| | | | +-- is-negated-glob@1.0.0
| | | | +-- ordered-read-streams@1.0.1
| | | | | `-- readable-stream@2.3.3 deduped
| | | | +-- pumpify@1.3.5
| | | | | +-- duplexify@3.5.1 deduped
| | | | | +-- inherits@2.0.3 deduped
| | | | | `-- pump@1.0.2 deduped
| | | | +-- readable-stream@2.3.3 deduped
| | | | +-- remove-trailing-separator@1.1.0
| | | | +-- to-absolute-glob@2.0.1
| | | | | +-- extend-shallow@2.0.1
| | | | | | `-- is-extendable@0.1.1
| | | | | +-- is-absolute@0.2.6
| | | | | | +-- is-relative@0.2.1
| | | | | | | `-- is-unc-path@0.1.2
| | | | | | | `-- unc-path-regex@0.1.2
| | | | | | `-- is-windows@0.2.0
| | | | | `-- is-negated-glob@1.0.0 deduped
| | | | `-- unique-stream@2.2.1
| | | |   +-- json-stable-stringify@1.0.1
| | | |   | `-- jsonify@0.0.0
| | | |   `-- through2-filter@2.0.0
| | | |     +-- through2@2.0.3 deduped
| | | |     `-- xtend@4.0.1 deduped
| | | +-- through2@2.0.3
| | | | +-- readable-stream@2.3.3 deduped
| | | | `-- xtend@4.0.1 deduped
| | | `-- xtend@4.0.1 deduped
| | +-- inherits@2.0.3
| | +-- minimist@1.2.0
| | +-- mqtt-packet@5.4.0
| | | +-- bl@1.2.1
| | | | `-- readable-stream@2.3.3 deduped
| | | +-- inherits@2.0.3 deduped
| | | +-- process-nextick-args@1.0.7
| | | `-- safe-buffer@5.1.1
| | +-- pump@1.0.2
| | | +-- end-of-stream@1.4.0 deduped
| | | `-- once@1.4.0 deduped
| | +-- readable-stream@2.3.3
| | | +-- core-util-is@1.0.2
| | | +-- inherits@2.0.3 deduped
| | | +-- isarray@1.0.0
| | | +-- process-nextick-args@1.0.7 deduped
| | | +-- safe-buffer@5.1.1 deduped
| | | +-- string_decoder@1.0.3
| | | | `-- safe-buffer@5.1.1 deduped
| | | `-- util-deprecate@1.0.2
| | +-- reinterval@1.1.0
| | +-- split2@2.2.0
| | | `-- through2@2.0.3 deduped
| | +-- websocket-stream@5.0.1
| | | +-- duplexify@3.5.1
| | | | +-- end-of-stream@1.4.0 deduped
| | | | +-- inherits@2.0.3 deduped
| | | | +-- readable-stream@2.3.3 deduped
| | | | `-- stream-shift@1.0.0
| | | +-- inherits@2.0.3 deduped
```

```
| | | +-- readable-stream@2.3.3 deduped
| | | +-- safe-buffer@5.1.1 deduped
| | | +-- ws@3.2.0
| | | | +-- async-limiter@1.0.0
| | | | +-- safe-buffer@5.1.1 deduped
| | | | `-- ultron@1.1.0
| | | `-- xtend@4.0.1 deduped
| | `-- xtend@4.0.1
| +-- multer@1.3.0
| | +-- append-field@0.1.0
| | +-- busboy@0.2.14
| | | +-- dicer@0.2.5
| | | | +-- readable-stream@1.1.14
| | | | | +-- core-util-is@1.0.2 deduped
| | | | | +-- inherits@2.0.3 deduped
| | | | | +-- isarray@0.0.1
| | | | | `-- string_decoder@0.10.31
| | | | `-- streamsearch@0.1.2
| | | `-- readable-stream@1.1.14
| | |   +-- core-util-is@1.0.2 deduped
| | |   +-- inherits@2.0.3 deduped
| | |   +-- isarray@0.0.1
| | |   `-- string_decoder@0.10.31
| | +-- concat-stream@1.6.0 deduped
| | +-- mkdirp@0.5.1
| | | `-- minimist@0.0.8
| | +-- object-assign@3.0.0
| | +-- on-finished@2.3.0 deduped
| | +-- type-is@1.6.15 deduped
| | `-- xtend@4.0.1 deduped
| +-- mustache@2.3.0
| +-- node-red-node-email@0.1.24
| | +-- imap@0.8.19
| | | +-- readable-stream@1.1.14
| | | | +-- core-util-is@1.0.2 deduped
| | | | +-- inherits@2.0.3 deduped
| | | | +-- isarray@0.0.1
| | | | `-- string_decoder@0.10.31
| | | `-- utf7@1.0.2
| | |   `-- semver@5.3.0 deduped
| | +-- mailparser@0.6.2
| | | +-- encoding@0.1.12
| | | | `-- iconv-lite@0.4.15 deduped
| | | +-- mime@1.3.4 deduped
| | | +-- mimelib@0.3.1
| | | | +-- addressparser@1.0.1
| | | | `-- encoding@0.1.12 deduped
| | | `-- uue@3.1.0
| | |   `-- extend@3.0.1 deduped
| | +-- nodemailer@1.11.0
| | | +-- libmime@1.2.0
| | | | +-- iconv-lite@0.4.15 deduped
| | | | +-- libbase64@0.1.0
| | | | `-- libqp@1.1.0
| | | +-- mailcomposer@2.1.0
| | | | +-- buildmail@2.0.0
| | | | | +-- addressparser@0.3.2
| | | | | +-- libbase64@0.1.0 deduped
| | | | | +-- libmime@1.2.0 deduped
| | | | | +-- libqp@1.1.0 deduped
| | | | | `-- needle@0.10.0
| | | | |   +-- debug@2.6.9 deduped
| | | | |   `-- iconv-lite@0.4.15 deduped
| | | | `-- libmime@1.2.0 deduped
| | | +-- needle@0.11.0
| | | | +-- debug@2.6.9 deduped
```

```
| | | | `-- iconv-lite@0.4.15 deduped
| | | +-- nodemailer-direct-transport@1.1.0
| | | | `-- smtp-connection@1.3.8
| | | `-- nodemailer-smtp-transport@1.1.0
| | |   +-- clone@1.0.2
| | |   +-- nodemailer-wellknown@0.1.10
| | |   `-- smtp-connection@1.3.8 deduped
| | `-- poplib@0.1.7
| |   `-- optimist@0.6.1
| |     +-- minimist@0.0.8 deduped
| |     `-- wordwrap@0.0.3
| +-- node-red-node-feedparser@0.1.8
| | +-- feedparser@1.1.3
| | | +-- addressparser@0.1.3
| | | +-- array-indexofobject@0.0.1
| | | +-- readable-stream@1.0.34
| | | | +-- core-util-is@1.0.2 deduped
| | | | +-- inherits@2.0.3 deduped
| | | | +-- isarray@0.0.1
| | | | `-- string_decoder@0.10.31
| | | `-- sax@0.6.1 deduped
| | `-- request@2.74.0
| |   +-- aws-sign2@0.6.0
| |   +-- aws4@1.6.0
| |   +-- bl@1.1.2
| |   | `-- readable-stream@2.0.6
| |   |   +-- core-util-is@1.0.2 deduped
| |   |   +-- inherits@2.0.3 deduped
| |   |   +-- isarray@1.0.0 deduped
| |   |   +-- process-nextick-args@1.0.7 deduped
| |   |   +-- string_decoder@0.10.31
| |   |   `-- util-deprecate@1.0.2 deduped
| |   +-- caseless@0.11.0
| |   +-- combined-stream@1.0.5
| |   | `-- delayed-stream@1.0.0
| |   +-- extend@3.0.1
| |   +-- forever-agent@0.6.1
| |   +-- form-data@1.0.1
| |   | +-- async@2.5.0
| |   | | `-- lodash@4.17.4 deduped
| |   | +-- combined-stream@1.0.5 deduped
| |   | `-- mime-types@2.1.17 deduped
| |   +-- har-validator@2.0.6
| |   | +-- chalk@1.1.3
| |   | | +-- ansi-styles@2.2.1
| |   | | +-- escape-string-regexp@1.0.5
| |   | | +-- has-ansi@2.0.0
| |   | | | `-- ansi-regex@2.1.1
| |   | | +-- strip-ansi@3.0.1
| |   | | | `-- ansi-regex@2.1.1 deduped
| |   | | `-- supports-color@2.0.0
| |   | +-- commander@2.11.0
| |   | +-- is-my-json-valid@2.16.1
| |   | | +-- generate-function@2.0.0
| |   | | +-- generate-object-property@1.2.0
| |   | | | `-- is-property@1.0.2
| |   | | +-- jsonpointer@4.0.1
| |   | | `-- xtend@4.0.1 deduped
| |   | `-- pinkie-promise@2.0.1
| |   |   `-- pinkie@2.0.4
| |   +-- hawk@3.1.3
| |   | +-- boom@2.10.1
| |   | | `-- hoek@2.16.3 deduped
| |   | +-- cryptiles@2.0.5
| |   | | `-- boom@2.10.1 deduped
| |   | +-- hoek@2.16.3
```

```
| |  | `-- sntp@1.0.9
| |  |  `-- hoek@2.16.3 deduped
| |  +-- http-signature@1.1.1
| |  | +-- assert-plus@0.2.0
| |  | +-- jsprim@1.4.1
| |  | | +-- assert-plus@1.0.0 deduped
| |  | | +-- extsprintf@1.3.0
| |  | | +-- json-schema@0.2.3
| |  | | `-- verror@1.10.0
| |  | |   +-- assert-plus@1.0.0 deduped
| |  | |   +-- core-util-is@1.0.2 deduped
| |  | |   `-- extsprintf@1.3.0 deduped
| |  | `-- sshpk@1.13.1
| |  |  +-- asn1@0.2.3
| |  |  +-- assert-plus@1.0.0 deduped
| |  |  +-- bcrypt-pbkdf@1.0.1
| |  |  | `-- tweetnacl@0.14.5 deduped
| |  |  +-- dashdash@1.14.1
| |  |  | `-- assert-plus@1.0.0 deduped
| |  |  +-- ecc-jsbn@0.1.1
| |  |  | `-- jsbn@0.1.1 deduped
| |  |  +-- getpass@0.1.7
| |  |  | `-- assert-plus@1.0.0 deduped
| |  |  +-- jsbn@0.1.1
| |  |  `-- tweetnacl@0.14.5
| |  +-- is-typedarray@1.0.0
| |  +-- isstream@0.1.2
| |  +-- json-stringify-safe@5.0.1 deduped
| |  +-- mime-types@2.1.17 deduped
| |  +-- node-uuid@1.4.8
| |  +-- oauth-sign@0.8.2
| |  +-- qs@6.2.3
| |  +-- stringstream@0.0.5
| |  +-- tough-cookie@2.3.3
| |  | `-- punycode@1.4.1
| |  `-- tunnel-agent@0.4.3
| +-- node-red-node-rbe@0.1.13
| +-- node-red-node-twitter@0.1.11
| | +-- oauth@0.9.14
| | +-- request@2.83.0
| | | +-- aws-sign2@0.7.0
| | | +-- aws4@1.6.0 deduped
| | | +-- caseless@0.12.0
| | | +-- combined-stream@1.0.5 deduped
| | | +-- extend@3.0.1 deduped
| | | +-- forever-agent@0.6.1 deduped
| | | +-- form-data@2.3.1
| | | | +-- asynckit@0.4.0
| | | | +-- combined-stream@1.0.5 deduped
| | | | `-- mime-types@2.1.17 deduped
| | | +-- har-validator@5.0.3
| | | | +-- ajv@5.2.3
| | | | | +-- co@4.6.0
| | | | | +-- fast-deep-equal@1.0.0
| | | | | +-- json-schema-traverse@0.3.1
| | | | | `-- json-stable-stringify@1.0.1 deduped
| | | | `-- har-schema@2.0.0
| | | +-- hawk@6.0.2
| | | | +-- boom@4.3.1
| | | | | `-- hoek@4.2.0 deduped
| | | | +-- cryptiles@3.1.2
| | | | | `-- boom@5.2.0
| | | | |   `-- hoek@4.2.0 deduped
| | | | +-- hoek@4.2.0
| | | | `-- sntp@2.0.2
| | | |   `-- hoek@4.2.0 deduped
```

```
| | | +-- http-signature@1.2.0
| | | | +-- assert-plus@1.0.0
| | | | +-- jsprim@1.4.1 deduped
| | | | `-- sshpk@1.13.1 deduped
| | | +-- is-typedarray@1.0.0 deduped
| | | +-- isstream@0.1.2 deduped
| | | +-- json-stringify-safe@5.0.1 deduped
| | | +-- mime-types@2.1.17 deduped
| | | +-- oauth-sign@0.8.2 deduped
| | | +-- performance-now@2.1.0
| | | +-- qs@6.5.1
| | | +-- safe-buffer@5.1.1 deduped
| | | +-- stringstream@0.0.5 deduped
| | | +-- tough-cookie@2.3.3 deduped
| | | +-- tunnel-agent@0.6.0
| | | | `-- safe-buffer@5.1.1 deduped
| | | `-- uuid@3.1.0
| | `-- twitter-ng@0.6.2
| |   `-- oauth@0.9.14 deduped
| +-- nopt@3.0.6
| | `-- abbrev@1.1.1
| +-- oauth2orize@1.8.0
| | +-- debug@2.6.9 deduped
| | +-- uid2@0.0.3
| | `-- utils-merge@1.0.0 deduped
| +-- on-headers@1.0.1
| +-- passport@0.3.2
| | +-- passport-strategy@1.0.0
| | `-- pause@0.0.1
| +-- passport-http-bearer@1.0.1
| | `-- passport-strategy@1.0.0 deduped
| +-- passport-oauth2-client-password@0.1.2
| | `-- passport-strategy@1.0.0 deduped
| +-- raw-body@2.2.0
| | +-- bytes@2.4.0 deduped
| | +-- iconv-lite@0.4.15 deduped
| | `-- unpipe@1.0.0
| +-- semver@5.3.0
| +-- sentiment@2.1.0
| +-- uglify-js@3.0.20
| | +-- commander@2.9.0
| | | `-- graceful-readlink@1.0.1
| | `-- source-map@0.5.7
| +-- when@3.7.8
| +-- ws@1.1.1
| | +-- options@0.0.6
| | `-- ultron@1.0.2
| `-- xml2js@0.4.17
|   +-- sax@0.6.1
|   `-- xmlbuilder@4.2.1
|     `-- lodash@4.17.4
`-- npm@5.4.2
  +-- abbrev@1.1.0
  +-- ansi-regex@3.0.0
  +-- ansicolors@0.3.2
  +-- ansistyles@0.1.3
  +-- aproba@1.1.2
  +-- archy@1.0.0
  +-- bluebird@3.5.0
  +-- cacache@9.2.9
  | +-- bluebird@3.5.0 deduped
  | +-- chownr@1.0.1 deduped
  | +-- glob@7.1.2 deduped
  | +-- graceful-fs@4.1.11 deduped
  | +-- lru-cache@4.1.1
  | | +-- pseudomap@1.0.2
```

```
| | `-- yallist@2.1.2
| +-- mississippi@1.3.0 deduped
| +-- mkdirp@0.5.1 deduped
| +-- move-concurrently@1.0.1 deduped
| +-- promise-inflight@1.0.1 deduped
| +-- rimraf@2.6.1 deduped
| +-- ssri@4.1.6 deduped
| +-- unique-filename@1.1.0 deduped
| `-- y18n@3.2.1
+-- call-limit@1.1.0
+-- chownr@1.0.1
+-- cmd-shim@2.0.2
| +-- graceful-fs@4.1.11 deduped
| `-- mkdirp@0.5.1 deduped
+-- columnify@1.5.4
| +-- strip-ansi@3.0.1
| | `-- ansi-regex@2.1.1
| `-- wcwidth@1.0.1
|   `-- defaults@1.0.3
|     `-- clone@1.0.2
+-- config-chain@1.1.11
| +-- ini@1.3.4 deduped
| `-- proto-list@1.2.4
+-- debuglog@1.0.1
+-- detect-indent@5.0.0
+-- dezalgo@1.0.3
| +-- asap@2.0.5
| `-- wrappy@1.0.2 deduped
+-- editor@1.0.0
+-- fs-vacuum@1.2.10
| +-- graceful-fs@4.1.11 deduped
| +-- path-is-inside@1.0.2 deduped
| `-- rimraf@2.6.1 deduped
+-- fs-write-stream-atomic@1.0.10
| +-- graceful-fs@4.1.11 deduped
| +-- iferr@0.1.5 deduped
| +-- imurmurhash@0.1.4 deduped
| `-- readable-stream@2.3.3 deduped
+-- glob@7.1.2
| +-- fs.realpath@1.0.0
| +-- inflight@1.0.6 deduped
| +-- inherits@2.0.3 deduped
| +-- minimatch@3.0.4
| | `-- brace-expansion@1.1.8
| |   +-- balanced-match@1.0.0
| |   `-- concat-map@0.0.1
| +-- once@1.4.0 deduped
| `-- path-is-absolute@1.0.1
+-- graceful-fs@4.1.11
+-- has-unicode@2.0.1
+-- hosted-git-info@2.5.0
+-- iferr@0.1.5
+-- imurmurhash@0.1.4
+-- inflight@1.0.6
| +-- once@1.4.0 deduped
| `-- wrappy@1.0.2 deduped
+-- inherits@2.0.3
+-- ini@1.3.4
+-- init-package-json@1.10.1
| +-- glob@7.1.2 deduped
| +-- npm-package-arg@5.1.2 deduped
| +-- promzard@0.3.0
| | `-- read@1.0.7 deduped
| +-- read@1.0.7 deduped
| +-- read-package-json@2.0.12 deduped
| +-- semver@5.4.1 deduped
```

```
| +-- validate-npm-package-license@3.0.1 deduped
| `-- validate-npm-package-name@3.0.0 deduped
+-- JSONStream@1.3.1
| +-- jsonparse@1.3.1
| `-- through@2.3.8
+-- lazy-property@1.0.0
+-- libnpx@9.6.0
| +-- dotenv@4.0.0
| +-- npm-package-arg@5.1.2 deduped
| +-- rimraf@2.6.1 deduped
| +-- safe-buffer@5.1.1 deduped
| +-- update-notifier@2.2.0 deduped
| +-- which@1.3.0 deduped
| +-- y18n@3.2.1
| `-- yargs@8.0.2
|   +-- camelcase@4.1.0
|   +-- cliui@3.2.0
|   | +-- string-width@1.0.2
|   | | +-- code-point-at@1.1.0
|   | | +-- is-fullwidth-code-point@1.0.0
|   | | | `-- number-is-nan@1.0.1
|   | | `-- strip-ansi@3.0.1 deduped
|   | +-- strip-ansi@3.0.1
|   | | `-- ansi-regex@2.1.1
|   | `-- wrap-ansi@2.1.0
|   |   +-- string-width@1.0.2 deduped
|   |   `-- strip-ansi@3.0.1 deduped
|   +-- decamelize@1.2.0
|   +-- get-caller-file@1.0.2
|   +-- os-locale@2.1.0
|   | +-- execa@0.7.0
|   | | +-- cross-spawn@5.1.0
|   | | | +-- lru-cache@4.1.1 deduped
|   | | | +-- shebang-command@1.2.0
|   | | | | `-- shebang-regex@1.0.0
|   | | | `-- which@1.3.0 deduped
|   | | +-- get-stream@3.0.0
|   | | +-- is-stream@1.1.0
|   | | +-- npm-run-path@2.0.2
|   | | | `-- path-key@2.0.1
|   | | +-- p-finally@1.0.0
|   | | +-- signal-exit@3.0.2
|   | | `-- strip-eof@1.0.0
|   | +-- lcid@1.0.0
|   | | `-- invert-kv@1.0.0
|   | `-- mem@1.1.0
|   |   `-- mimic-fn@1.1.0
|   +-- read-pkg-up@2.0.0
|   | +-- find-up@2.1.0
|   | | `-- locate-path@2.0.0
|   | |   +-- p-locate@2.0.0
|   | |   | `-- p-limit@1.1.0
|   | |   `-- path-exists@3.0.0
|   | `-- read-pkg@2.0.0
|   |   +-- load-json-file@2.0.0
|   |   | +-- graceful-fs@4.1.11 deduped
|   |   | +-- parse-json@2.2.0
|   |   | | `-- error-ex@1.3.1
|   |   | |   `-- is-arrayish@0.2.1
|   |   | +-- pify@2.3.0
|   |   | `-- strip-bom@3.0.0
|   |   +-- normalize-package-data@2.4.0 deduped
|   |   `-- path-type@2.0.0
|   |     `-- pify@2.3.0
|   +-- require-directory@2.1.1
|   +-- require-main-filename@1.0.1
```

```
|   +-- set-blocking@2.0.0
|   +-- string-width@2.1.1
|   | +-- is-fullwidth-code-point@2.0.0
|   | `-- strip-ansi@4.0.0 deduped
|   +-- which-module@2.0.0
|   +-- y18n@3.2.1 deduped
|   `-- yargs-parser@7.0.0
|     `-- camelcase@4.1.0 deduped
+-- lockfile@1.0.3
+-- lodash._baseindexof@3.1.0
+-- lodash._baseuniq@4.6.0
| +-- lodash._createset@4.0.3
| `-- lodash._root@3.0.1
+-- lodash._bindcallback@3.0.1
+-- lodash._cacheindexof@3.0.2
+-- lodash._createcache@3.1.2
| `-- lodash._getnative@3.9.1 deduped
+-- lodash._getnative@3.9.1
+-- lodash.clonedeep@4.5.0
+-- lodash.restparam@3.6.1
+-- lodash.union@4.6.0
+-- lodash.uniq@4.5.0
+-- lodash.without@4.4.0
+-- lru-cache@4.1.1
| +-- pseudomap@1.0.2
| `-- yallist@2.1.2
+-- meant@1.0.0
+-- mississippi@1.3.0
| +-- concat-stream@1.6.0
| | +-- inherits@2.0.3 deduped
| | +-- readable-stream@2.3.3 deduped
| | `-- typedarray@0.0.6
| +-- duplexify@3.5.0
| | +-- end-of-stream@1.0.0
| | | `-- once@1.3.3
| | |   `-- wrappy@1.0.2 deduped
| | +-- inherits@2.0.3 deduped
| | +-- readable-stream@2.3.3 deduped
| | `-- stream-shift@1.0.0
| +-- end-of-stream@1.4.0
| | `-- once@1.4.0 deduped
| +-- flush-write-stream@1.0.2
| | +-- inherits@2.0.3 deduped
| | `-- readable-stream@2.3.3 deduped
| +-- from2@2.3.0
| | +-- inherits@2.0.3 deduped
| | `-- readable-stream@2.3.3 deduped
| +-- parallel-transform@1.1.0
| | +-- cyclist@0.2.2
| | +-- inherits@2.0.3 deduped
| | `-- readable-stream@2.3.3 deduped
| +-- pump@1.0.2
| | +-- end-of-stream@1.4.0 deduped
| | `-- once@1.4.0 deduped
| +-- pumpify@1.3.5
| | +-- duplexify@3.5.0 deduped
| | +-- inherits@2.0.3 deduped
| | `-- pump@1.0.2 deduped
| +-- stream-each@1.2.0
| | +-- end-of-stream@1.4.0 deduped
| | `-- stream-shift@1.0.0
| `-- through2@2.0.3
|   +-- readable-stream@2.3.3 deduped
|   `-- xtend@4.0.1
+-- mkdirp@0.5.1
| `-- minimist@0.0.8
```

```
+-- move-concurrently@1.0.1
| +-- aproba@1.1.2 deduped
| +-- copy-concurrently@1.0.3
| | +-- aproba@1.1.2 deduped
| | +-- fs-write-stream-atomic@1.0.10 deduped
| | +-- iferr@0.1.5 deduped
| | +-- mkdirp@0.5.1 deduped
| | +-- rimraf@2.6.1 deduped
| | `-- run-queue@1.0.3 deduped
| +-- fs-write-stream-atomic@1.0.10 deduped
| +-- mkdirp@0.5.1 deduped
| +-- rimraf@2.6.1 deduped
| `-- run-queue@1.0.3
|   `-- aproba@1.1.2 deduped
+-- node-gyp@3.6.2
| +-- fstream@1.0.11
| | +-- graceful-fs@4.1.11 deduped
| | +-- inherits@2.0.3 deduped
| | +-- mkdirp@0.5.1 deduped
| | `-- rimraf@2.6.1 deduped
| +-- glob@7.1.2 deduped
| +-- graceful-fs@4.1.11 deduped
| +-- minimatch@3.0.4
| | `-- brace-expansion@1.1.8
| |   +-- balanced-match@1.0.0
| |   `-- concat-map@0.0.1
| +-- mkdirp@0.5.1 deduped
| +-- nopt@3.0.6
| | `-- abbrev@1.1.0 deduped
| +-- npmlog@4.1.2 deduped
| +-- osenv@0.1.4 deduped
| +-- request@2.81.0 deduped
| +-- rimraf@2.6.1 deduped
| +-- semver@5.3.0
| +-- tar@2.2.1
| | +-- block-stream@0.0.9
| | | `-- inherits@2.0.3 deduped
| | +-- fstream@1.0.11 deduped
| | `-- inherits@2.0.3 deduped
| `-- which@1.3.0 deduped
+-- nopt@4.0.1
| +-- abbrev@1.1.0 deduped
| `-- osenv@0.1.4 deduped
+-- normalize-package-data@2.4.0
| +-- hosted-git-info@2.5.0 deduped
| +-- is-builtin-module@1.0.0
| | `-- builtin-modules@1.1.1
| +-- semver@5.4.1 deduped
| `-- validate-npm-package-license@3.0.1 deduped
+-- npm-cache-filename@1.0.2
+-- npm-install-checks@3.0.0
| `-- semver@5.4.1 deduped
+-- npm-lifecycle@1.0.2
| +-- graceful-fs@4.1.11 deduped
| +-- slide@1.1.6 deduped
| +-- uid-number@0.0.6 deduped
| +-- umask@1.1.0 deduped
| `-- which@1.3.0 deduped
+-- npm-package-arg@5.1.2
| +-- hosted-git-info@2.5.0 deduped
| +-- osenv@0.1.4 deduped
| +-- semver@5.4.1 deduped
| `-- validate-npm-package-name@3.0.0 deduped
+-- npm-packlist@1.1.8
| +-- ignore-walk@3.0.0
| | `-- minimatch@3.0.4
```

```
| |   `-- brace-expansion@1.1.8
| |     +-- balanced-match@1.0.0
| |     `-- concat-map@0.0.1
| `-- npm-bundled@1.0.3
+-- npm-registry-client@8.4.0
| +-- concat-stream@1.6.0
| | +-- inherits@2.0.3 deduped
| | +-- readable-stream@2.3.3 deduped
| | `-- typedarray@0.0.6
| +-- graceful-fs@4.1.11 deduped
| +-- normalize-package-data@2.4.0 deduped
| +-- npm-package-arg@5.1.2 deduped
| +-- npmlog@4.1.2 deduped
| +-- once@1.4.0 deduped
| +-- request@2.81.0 deduped
| +-- retry@0.10.1 deduped
| +-- semver@5.4.1 deduped
| +-- slide@1.1.6 deduped
| `-- ssri@4.1.6 deduped
+-- npm-user-validate@1.0.0
+-- npmlog@4.1.2
| +-- are-we-there-yet@1.1.4
| | +-- delegates@1.0.0
| | `-- readable-stream@2.3.3 deduped
| +-- console-control-strings@1.1.0
| +-- gauge@2.7.4
| | +-- aproba@1.1.2 deduped
| | +-- console-control-strings@1.1.0 deduped
| | +-- has-unicode@2.0.1 deduped
| | +-- object-assign@4.1.1
| | +-- signal-exit@3.0.2
| | +-- string-width@1.0.2
| | | +-- code-point-at@1.1.0
| | | +-- is-fullwidth-code-point@1.0.0
| | | | `-- number-is-nan@1.0.1
| | | `-- strip-ansi@3.0.1 deduped
| | +-- strip-ansi@3.0.1
| | | `-- ansi-regex@2.1.1
| | `-- wide-align@1.1.2
| |   `-- string-width@1.0.2 deduped
| `-- set-blocking@2.0.0
+-- once@1.4.0
| `-- wrappy@1.0.2 deduped
+-- opener@1.4.3
+-- osenv@0.1.4
| +-- os-homedir@1.0.2
| `-- os-tmpdir@1.0.2
+-- pacote@6.0.2
| +-- bluebird@3.5.0 deduped
| +-- cacache@9.2.9 deduped
| +-- glob@7.1.2 deduped
| +-- lru-cache@4.1.1 deduped
| +-- make-fetch-happen@2.5.0
| | +-- agentkeepalive@3.3.0
| | | `-- humanize-ms@1.2.1
| | |   `-- ms@2.0.0
| | +-- cacache@9.2.9 deduped
| | +-- http-cache-semantics@3.7.3
| | +-- http-proxy-agent@2.0.0
| | | +-- agent-base@4.1.1
| | | | `-- es6-promisify@5.0.0
| | | |   `-- es6-promise@4.1.1
| | | `-- debug@2.6.8
| | |   `-- ms@2.0.0
| | +-- https-proxy-agent@2.1.0
| | | +-- agent-base@4.1.1
```

```
| | | | `-- es6-promisify@5.0.0
| | | |   `-- es6-promise@4.1.1
| | | `-- debug@2.6.8
| | |   `-- ms@2.0.0
| | +-- lru-cache@4.1.1 deduped
| | +-- mississippi@1.3.0 deduped
| | +-- node-fetch-npm@2.0.2
| | | +-- encoding@0.1.12
| | | | `-- iconv-lite@0.4.18
| | | +-- json-parse-better-errors@1.0.1
| | | `-- safe-buffer@5.1.1 deduped
| | +-- promise-retry@1.1.1 deduped
| | +-- socks-proxy-agent@3.0.0
| | | +-- agent-base@4.1.1
| | | | `-- es6-promisify@5.0.0
| | | |   `-- es6-promise@4.1.1
| | | `-- socks@1.1.10
| | |   +-- ip@1.1.5
| | |   `-- smart-buffer@1.1.15
| | `-- ssri@4.1.6 deduped
| +-- minimatch@3.0.4
| | `-- brace-expansion@1.1.8
| |   +-- balanced-match@1.0.0
| |   `-- concat-map@0.0.1
| +-- mississippi@1.3.0 deduped
| +-- normalize-package-data@2.4.0 deduped
| +-- npm-package-arg@5.1.2 deduped
| +-- npm-packlist@1.1.8 deduped
| +-- npm-pick-manifest@1.0.4
| | +-- npm-package-arg@5.1.2 deduped
| | `-- semver@5.4.1 deduped
| +-- osenv@0.1.4 deduped
| +-- promise-inflight@1.0.1 deduped
| +-- promise-retry@1.1.1
| | +-- err-code@1.1.2
| | `-- retry@0.10.1 deduped
| +-- protoduck@4.0.0
| | `-- genfun@4.0.1
| +-- safe-buffer@5.1.1 deduped
| +-- semver@5.4.1 deduped
| +-- ssri@4.1.6 deduped
| +-- tar@4.0.1 deduped
| +-- unique-filename@1.1.0 deduped
| `-- which@1.3.0 deduped
+-- path-is-inside@1.0.2
+-- promise-inflight@1.0.1
+-- read@1.0.7
| `-- mute-stream@0.0.7
+-- read-cmd-shim@1.0.1
| `-- graceful-fs@4.1.11 deduped
+-- read-installed@4.0.3
| +-- debuglog@1.0.1 deduped
| +-- graceful-fs@4.1.11 deduped
| +-- read-package-json@2.0.12 deduped
| +-- readdir-scoped-modules@1.0.2 deduped
| +-- semver@5.4.1 deduped
| +-- slide@1.1.6 deduped
| `-- util-extend@1.0.3
+-- read-package-json@2.0.12
| +-- glob@7.1.2 deduped
| +-- graceful-fs@4.1.11 deduped
| +-- json-parse-better-errors@1.0.1
| +-- normalize-package-data@2.4.0 deduped
| `-- slash@1.0.0
+-- read-package-tree@5.1.6
| +-- debuglog@1.0.1 deduped
```

```
| +-- dezalgo@1.0.3 deduped
| +-- once@1.4.0 deduped
| +-- read-package-json@2.0.12 deduped
| `-- readdir-scoped-modules@1.0.2 deduped
+-- readable-stream@2.3.3
| +-- core-util-is@1.0.2
| +-- inherits@2.0.3 deduped
| +-- isarray@1.0.0
| +-- process-nextick-args@1.0.7
| +-- safe-buffer@5.1.1 deduped
| +-- string_decoder@1.0.3
| | `-- safe-buffer@5.1.1 deduped
| `-- util-deprecate@1.0.2
+-- readdir-scoped-modules@1.0.2
| +-- debuglog@1.0.1 deduped
| +-- dezalgo@1.0.3 deduped
| +-- graceful-fs@4.1.11 deduped
| `-- once@1.4.0 deduped
+-- request@2.81.0
| +-- aws-sign2@0.6.0
| +-- aws4@1.6.0
| +-- caseless@0.12.0
| +-- combined-stream@1.0.5
| | `-- delayed-stream@1.0.0
| +-- extend@3.0.1
| +-- forever-agent@0.6.1
| +-- form-data@2.1.4
| | +-- asynckit@0.4.0
| | +-- combined-stream@1.0.5 deduped
| | `-- mime-types@2.1.15 deduped
| +-- har-validator@4.2.1
| | +-- ajv@4.11.8
| | | +-- co@4.6.0
| | | `-- json-stable-stringify@1.0.1
| | |   `-- jsonify@0.0.0
| | `-- har-schema@1.0.5
| +-- hawk@3.1.3
| | +-- boom@2.10.1
| | | `-- hoek@2.16.3 deduped
| | +-- cryptiles@2.0.5
| | | `-- boom@2.10.1 deduped
| | +-- hoek@2.16.3
| | `-- sntp@1.0.9
| |   `-- hoek@2.16.3 deduped
| +-- http-signature@1.1.1
| | +-- assert-plus@0.2.0
| | +-- jsprim@1.4.0
| | | +-- assert-plus@1.0.0
| | | +-- extsprintf@1.0.2
| | | +-- json-schema@0.2.3
| | | `-- verror@1.3.6
| | |   `-- extsprintf@1.0.2 deduped
| | `-- sshpk@1.13.1
| |   +-- asn1@0.2.3
| |   +-- assert-plus@1.0.0
| |   +-- bcrypt-pbkdf@1.0.1
| |   | `-- tweetnacl@0.14.5 deduped
| |   +-- dashdash@1.14.1
| |   | `-- assert-plus@1.0.0 deduped
| |   +-- ecc-jsbn@0.1.1
| |   | `-- jsbn@0.1.1 deduped
| |   +-- getpass@0.1.7
| |   | `-- assert-plus@1.0.0 deduped
| |   +-- jsbn@0.1.1
| |   `-- tweetnacl@0.14.5
| +-- is-typedarray@1.0.0
```

```
| +-- isstream@0.1.2
| +-- json-stringify-safe@5.0.1
| +-- mime-types@2.1.15
| | `-- mime-db@1.27.0
| +-- oauth-sign@0.8.2
| +-- performance-now@0.2.0
| +-- qs@6.4.0
| +-- safe-buffer@5.1.1 deduped
| +-- stringstream@0.0.5
| +-- tough-cookie@2.3.2
| | `-- punycode@1.4.1
| +-- tunnel-agent@0.6.0
| | `-- safe-buffer@5.1.1 deduped
| `-- uuid@3.1.0 deduped
+-- retry@0.10.1
+-- rimraf@2.6.1
| `-- glob@7.1.2 deduped
+-- safe-buffer@5.1.1
+-- semver@5.4.1
+-- sha@2.0.1
| +-- graceful-fs@4.1.11 deduped
| `-- readable-stream@2.3.3 deduped
+-- slide@1.1.6
+-- sorted-object@2.0.1
+-- sorted-union-stream@2.1.3
| +-- from2@1.3.0
| | +-- inherits@2.0.3 deduped
| | `-- readable-stream@1.1.14
| |   +-- core-util-is@1.0.2
| |   +-- inherits@2.0.3 deduped
| |   +-- isarray@0.0.1
| |   `-- string_decoder@0.10.31
| `-- stream-iterate@1.2.0
|   +-- readable-stream@2.3.3 deduped
|   `-- stream-shift@1.0.0
+-- ssri@4.1.6
| `-- safe-buffer@5.1.1 deduped
+-- strip-ansi@4.0.0
| `-- ansi-regex@3.0.0
+-- tar@4.0.1
| +-- chownr@1.0.1 deduped
| +-- minipass@2.2.1
| | `-- yallist@3.0.2 deduped
| +-- minizlib@1.0.3
| | `-- minipass@2.2.1 deduped
| +-- mkdirp@0.5.1 deduped
| `-- yallist@3.0.2
+-- text-table@0.2.0
+-- uid-number@0.0.6
+-- umask@1.1.0
+-- unique-filename@1.1.0
| `-- unique-slug@2.0.0
|   `-- imurmurhash@0.1.4 deduped
+-- unpipe@1.0.0
+-- update-notifier@2.2.0
| +-- boxen@1.1.0
| | +-- ansi-align@2.0.0
| | | `-- string-width@2.1.0 deduped
| | +-- camelcase@4.1.0
| | +-- chalk@1.1.3 deduped
| | +-- cli-boxes@1.0.0
| | +-- string-width@2.1.0
| | | +-- is-fullwidth-code-point@2.0.0
| | | `-- strip-ansi@4.0.0
| | |   `-- ansi-regex@3.0.0 deduped
| | +-- term-size@0.1.1
```

```
| | | `-- execa@0.4.0
| | |   +-- cross-spawn-async@2.2.5
| | |   | +-- lru-cache@4.1.1 deduped
| | |   | `-- which@1.3.0 deduped
| | |   +-- is-stream@1.1.0
| | |   +-- npm-run-path@1.0.0
| | |   | `-- path-key@1.0.0 deduped
| | |   +-- object-assign@4.1.1
| | |   +-- path-key@1.0.0
| | |   `-- strip-eof@1.0.0
| | `-- widest-line@1.0.0
| |   `-- string-width@1.0.2
| |     +-- code-point-at@1.1.0
| |     +-- is-fullwidth-code-point@1.0.0
| |     | `-- number-is-nan@1.0.1
| |     `-- strip-ansi@3.0.1
| |       `-- ansi-regex@2.1.1
| +-- chalk@1.1.3
| | +-- ansi-styles@2.2.1
| | +-- escape-string-regexp@1.0.5
| | +-- has-ansi@2.0.0
| | | `-- ansi-regex@2.1.1
| | +-- strip-ansi@3.0.1
| | | `-- ansi-regex@2.1.1
| | `-- supports-color@2.0.0
| +-- configstore@3.1.0
| | +-- dot-prop@4.1.1
| | | `-- is-obj@1.0.1
| | +-- graceful-fs@4.1.11 deduped
| | +-- make-dir@1.0.0
| | | `-- pify@2.3.0
| | +-- unique-string@1.0.0
| | | `-- crypto-random-string@1.0.0
| | +-- write-file-atomic@2.1.0 deduped
| | `-- xdg-basedir@3.0.0 deduped
| +-- import-lazy@2.1.0
| +-- is-npm@1.0.0
| +-- latest-version@3.1.0
| | `-- package-json@4.0.1
| |   +-- got@6.7.1
| |   | +-- create-error-class@3.0.2
| |   | | `-- capture-stack-trace@1.0.0
| |   | +-- duplexer3@0.1.4
| |   | +-- get-stream@3.0.0
| |   | +-- is-redirect@1.0.0
| |   | +-- is-retry-allowed@1.1.0
| |   | +-- is-stream@1.1.0
| |   | +-- lowercase-keys@1.0.0
| |   | +-- safe-buffer@5.1.1 deduped
| |   | +-- timed-out@4.0.1
| |   | +-- unzip-response@2.0.1
| |   | `-- url-parse-lax@1.0.0
| |   |   `-- prepend-http@1.0.4
| |   +-- registry-auth-token@3.3.1
| |   | +-- rc@1.2.1
| |   | | +-- deep-extend@0.4.2
| |   | | +-- ini@1.3.4 deduped
| |   | | +-- minimist@1.2.0
| |   | | `-- strip-json-comments@2.0.1
| |   | `-- safe-buffer@5.1.1 deduped
| |   +-- registry-url@3.1.0
| |   | `-- rc@1.2.1
| |   |   +-- deep-extend@0.4.2
| |   |   +-- ini@1.3.4 deduped
| |   |   +-- minimist@1.2.0
| |   |   `-- strip-json-comments@2.0.1
```

```
| |   `-- semver@5.4.1 deduped
| +-- semver-diff@2.1.0
| | `-- semver@5.4.1 deduped
| `-- xdg-basedir@3.0.0
+-- uuid@3.1.0
+-- validate-npm-package-license@3.0.1
| +-- spdx-correct@1.0.2
| | `-- spdx-license-ids@1.2.2
| `-- spdx-expression-parse@1.0.4
+-- validate-npm-package-name@3.0.0
| `-- builtins@1.0.3
+-- which@1.3.0
| `-- isexe@2.0.0
+-- worker-farm@1.5.0
| +-- errno@0.1.4
| | `-- p
```

9 Appendix

The software tools that are useful to work with SE59XX-SDK Node-RED are the following:

- **PuTTY**: to use command line interface - https://www.putty.org/
- **Tftpd64**: ftpd64 is a free, opensource IPv6 ready application which includes DHCP, TFTP, DNS, SNTP and Syslog servers as well as a TFTP client. - https://tftpd64.codeplex.com/releases/view/630491
- **Node.js**: an open-source, cross-platform JavaScript run-time environment for executing JavaScript code server-side. Historically, JavaScript was used primarily for client-side scripting, in which scripts written in JavaScript are embedded in a webpage's HTML, to be run client-side by a JavaScript engine in the user's web browser. Node.js enables JavaScript to be used for server-side scripting, and runs scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js has become one of the foundational elements of the "JavaScript everywhere" paradigm, allowing web application development to unify around a single programming language, rather than rely on a different language for writing server side scripts. https://nodejs.org/en/download/
- **Node-RED package (original):** Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.- https://nodered.org/

Atop Technologies, Inc.

www.atoponline.com
www.atop.com.tw

TAIWAN HEADQUARTER:

2F, No. 146, Sec. 1, Tung-Hsing Rd,
30261 Chupei City, Hsinchu County
Taiwan, R.O.C.
Tel: +886-3-550-8137
Fax: +886-3-550-8131

ATOP CHINA BRANCH:

3F, 75th, No. 1066 Building,
Qingzhou North Road,
Shanghai, China
Tel: +86-21-64956231

ATOP INDIA OFFICE:

Abhishek Srivastava
Head of India Sales
Atop Communication Solution(P) Ltd.
No. 22, Kensington Terrace,
Kensington Rd,
Bangalore, 560008, India
Tel: +91-80-4920-6363
E-mail: Abhishek.S@atop.in

ATOP INDONESIA BRANCH:

Jopson Li
Branch Director
Wisma Lampung Jl.
No. 40, Tomang Raya
Jakarta, Barat, 11430, Indonesia
Tel: +62-857-10595775
E-mail: jopsonli@atop.com.tw

ATOP EMEA OFFICE:

Bhaskar Kailas (BK)
Vice President (Business Development)
Atop Communication Solution(P) Ltd.
No. 22, Kensington Terrace,
Kensington Rd,
Bangalore, 560008, India
Tel: +91-988-0788-559
E-mail: Bhaskar.k@atop.in

ATOP AMERICAs OFFICE:

Venke Char
Sr. Vice President & Head of Business
11811 North Tatum Blvd, Suite 3031
Phoenix, AZ 85028,
United States
Tel: +1-602-953-7669
E-mail: venke@atop.in