Remote Management Guide for Smart Radio

Advanced Mesh Router for Private Wireless Networks

Introduction

The Smart Radio runs the Mesh Rider OS. It is a customized version of Openwrt with enhancements useful for applications requiring low-latency command-and-control transmission and high-throughput video - e.g. UAV and robotics.

The purpose of this guide is to aide a user in remotely configuring Smart Radio settings. There are four ways to configure the Smart Radio.

1. Using the Web GUI
2. Using SSH
3. Using the JSON-RPC API
4. Using SNMP

A full technical discussion of all of the different configuration parameters of the Openwrt system is beyond the scope of this guide, and the user is encouraged to explore the OpenWrt website.

Web GUI

The Web GUI can be accessed in any web browser at https://<IP ADDRESS> (port 443). Note that the web browser uses a self-signed certificate. This means that connection to the web browser is encrypted, but not authenticated. The first time you access the Smart Radio from a new browser, you will get a SSL certificate warning. It is okay to ignore the warning and proceed.

SSH

It is possible to remotely retrieve network information using the GUI by logging into the Smart Radio with a web browser at the ETH0 IP address, or over the command line using a combination of SSH and Linux command-line utilities. Please refer to Appendix A for some examples.

JSON-RPC API

If you plan on integrating remote network retrieval into an application, then the JSON-RPC API may be useful. For a typical application, there are two types of interactions you would normally make with the Smart Radio. Firstly you may want to reconfigure certain basic operating parameters such as the operating channel or operating bandwidth. Secondly, you may want to retrieve current network status such as RSSI. In the first case, we recommend you use the UCI system, and for the second case, you will need to execute specific programs in the radio such as iwinfo, iw, or batctl.

For detailed descriptions on how to use the JSON-RPC API, please refer to the Appendix B.

SNMP

The Smart Radio also supports the SNMP protocol. Please refer to Appendix C for more details on using the SNMP.

Appendix A – Using SSH

The followings are some examples of remotely executing a command via SSH to obtain network information from the node.

```
ssh root@10.223.134.70 'iw wlan0 info'
```

Lists the current configuration of wlan0

```
ssh root@10.223.134.70 'iw wlan0 station dump'
```

shows connection information to all currently connected stations.

```
ssh root@10.223.134.70 'iw wlan0 list'
```

shows the capabilities of the wlan0 interface. An example output is shown below.

```
root@LEDE:~# iw dev wlan0 info
Interface wlan0
        ifindex 13
        wdev 0x7
        addr 00:30:1a:4e:86:46
        type mesh point
        wiphy 0
        channel 12 (915 MHz), width: 20 MHz, center1: 915 MHz
        txpower 32.00 dBm
```

Appendix B – Using the JSON-RPC API

For detailed descriptions on how to use the JSON-RPC API, please refer to the following links:

https://openwrt.org/docs/techref/ubus

https://github.com/openwrt/luci/wiki/JsonRpcHowTo

The listings below, along with additional scripts and packages can be found here:

https://doodlelabs.sharepoint.com/:f:/s/Technical/Elhb0AcP3ClCltsuPOu9gqEBY0v34XZYp1GDrMCTaRS8oA?e=Y8g5kn

In order to access the radio, you first need to set the permissions for your user profile (to be created next). Create a new profile /usr/share/rpcd/acl.d/superuser.json like the one below.

```
{
	"superuser": {
		"description": "Super user access role",
		"read": {
			"ubus": {
				"*": [ "*" ]
			},
			"uci": [ "*" ],
			"file": {
				"*": ["*"]
			}
		},
		"write": {
			"ubus": {
				"*": [ "*" ]
			},
			"uci": [ "*" ],
			"file": {
				"*": ["*"]
			},
			"cgi-io": ["*"]
		}
	}
}
```

Next, create a user profile in /etc/config/rpcd, and then restart the process with /etc/init.d/rpcd restart. An example profile is shown below.

```
config login
        option username 'root'
        option password '$p$root'
        list read 'superuser'
        list write 'superuser'
```

The API is now ready to be used. You should adjust the user profile and permissions to your liking. The permissions above could pose a security risk. In order to use API, you need to get a session ID and use it for subsequent requests. For ubus, for example, you can try the following call,

```
curl -k https://<IP-ADDRESS>/ubus -d '
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "call",
  "params": [ "00000000000000000000000000000000", "session", "login", { "username":
"root", "password": "secret"  } ]
}'
```

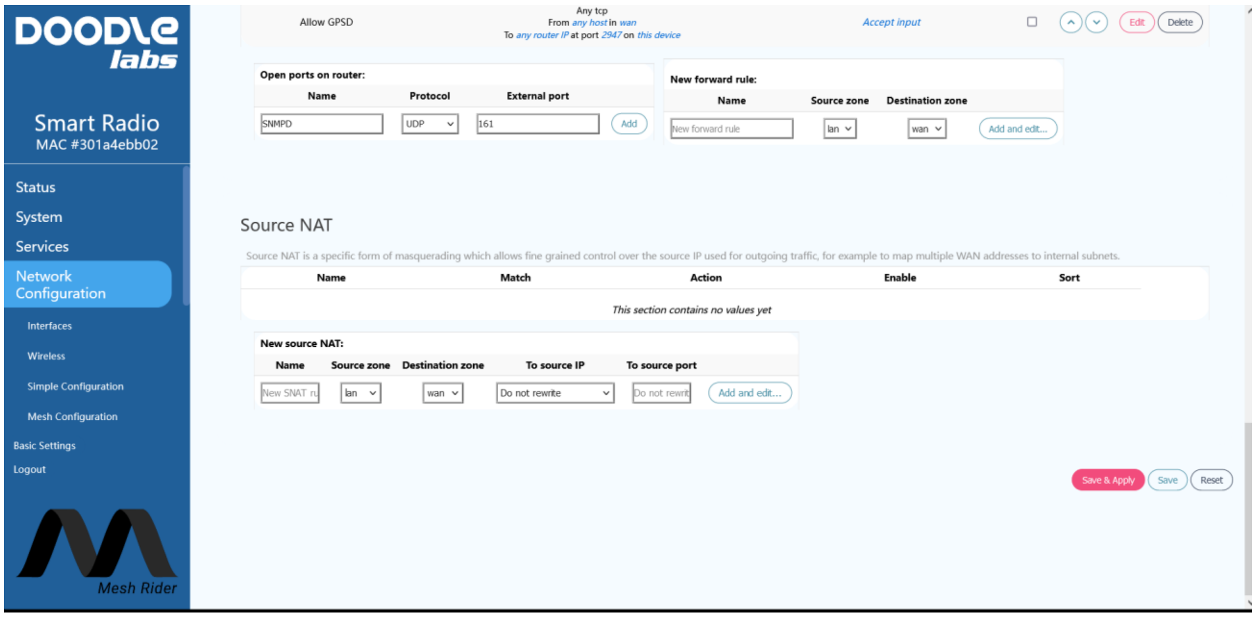
the -k option is required because the Smart Radio doesn’t use a third party certificate authority. An example of using JSON-RPC API for file access is shown below. Substitute <TOKEN> with the value returned above.

```
curl -k https://<IP-ADDRESS>/ubus -d '
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "call",
  "params": [ “<TOKEN>”, "file", "read", { "path": "/etc/board.json" } ]

}'
```

Appendix C – Using SNMP

Before you can use SNMP, you need to open up the firewall at port 161 for UDP traffic. In the GUI, navigate to `Network Configuration → Firewall`. Under the section `Open ports on router`, add a new rule like that shown in the figure below. Click `Add`, and after the page reloads, click `Save & Apply`.



Allow GPSD	Any tcp From *any host* in *wan* To *any router IP* at port *2947* on *this device*	*Accept input*	☐

Open ports on router:

Name	Protocol	External port
SNMPD	UDP	161

New forward rule:

Name	Source zone	Destination zone
New forward rule	lan	wan

Source NAT

Source NAT is a specific form of masquerading which allows fine grained control over the source IP used for outgoing traffic, for example to map multiple WAN addresses to internal subnets.

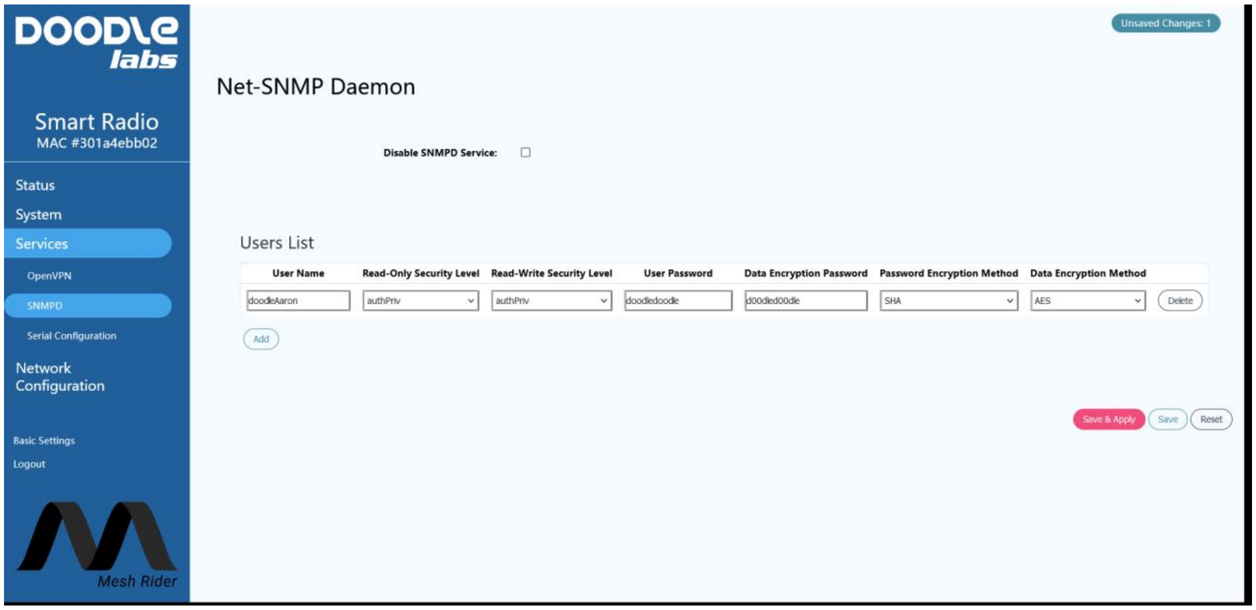
Name	Match	Action	Enable	Sort
This section contains no values yet				

New source NAT:

Name	Source zone	Destination zone	To source IP	To source port
New SNAT ru	lan	wan	Do not rewrite	Do not rewrit

Fig. 1 – Opening the Firewall for SNMP

In order to enable SNMP, click the `Advanced Settings` button in the bottom left corner of the GUI. Then navigate to `services → SNMPD`. Uncheck `Disable SNMPD Service`, and fill in the table. Finally click `Save & Apply`.



Net-SNMP Daemon

Disable SNMPD Service: ☐

Users List

User Name	Read-Only Security Level	Read-Write Security Level	User Password	Data Encryption Password	Password Encryption Method	Data Encryption Method
doodleAaron	authPriv	authPriv	doodledoodle	d00dled00dle	SHA	AES

Fig. 2 – Enabling and Configuring SNMP

With SNMP enabled, we can test it using the program `snmpwalk` in Linux. The listing below shows an example test script.

```
#!/bin/sh
IPADDR="10.223.187.2"
USRNAME="doodleAaron"
USRPASS="doodledoodle"
DATAPASS="d00dled00dle"

snmpwalk -m ALL -M ./ -v 3 -u $USRNAME -l authPriv -a SHA -A $USRPASS -x AES
-X $DATAPASS $IPADDR 1.3.6.1.3.12314
```

Modify the user login information in `snmpwalk.sh` using the information you used in the web GUI setup. The MIB is defined in /usr/lib/lua/smithsnmpd/mibs/wifi.lua and needs to be modified to be able to support the information defined in WIFI-MIB.txt which is available here:

https://doodlelabs.sharepoint.com/:f:/s/Technical/Elhb0AcP3ClCltsuPOu9gqEBY0v34XZYp1GDrMCTaRS8oA?e=Y8g5kn