AREDN Phonebook Installation

Principle of operation

The “Original” Swiss AREDN phonebook is on Google: https://docs.google.com/spreadsheets/d/1g33BHSXMC8T4Cmfz_Zq-XxtPP17dtEBexF2i4KKe_Mc/edit?usp=sharing. You can create a comment to add or change something. The goal is to transfer this phonebook to all AREDN phones in Switzerland.

For the moment, we support Yealink telephones, and Cisco phones are in the test.

The telephones used for AREDN offer phonebooks that can be automatically loaded from a remote location. The file format used for that process is XML.

AREDN is a mesh network, and we do not want to create a single point of failure. This is why the telephones have to get their phonebook files from the hap router they are connected to. So there is no single point of failure, and a phone gets its phonebook as long as its router is working.

To avoid a single point of failure for communication, to reduce the latency time, and to reduce the overload of single mesh segments, we want to use direct calling instead of a PBX. The address used for this case is a FQDN like 178230@178230.local.mesh. If you want or need to operate a PBX, the address is just a phone number like 178230. In Switzerland, we use the “Postleitzahl” of the city of the HAM plus a two-digit number in the range of 30-70. Lower numbers are reserved for official use.

To support direct calling and PBBX, we need two different phone books in our phones (direct and via PBX).

How is the information transferred from the Google Sheets to your hap router? The first step is to transfer the .csv version of the sheet to a web server in the AREDN mesh. If Google is down, we could still edit this .csv File manually. This transfer is done every hour. An example job is in the repository.

On our router, we have to install three bash scripts. “phonebook_installer.sh” downloads the newest version of the files, and “AREDN_Phonebook.csv” from the web server mentioned in the previous paragraph. It also copies a settings.txt file into the directory “/arednstack/phonebook”. Then, it starts “phonebook_creator_direct.sh” to create the different XML files with the direct calling info (example: “phonebook_yealink_direct.xml”) and saves it to the /www directory of our router. From there, our phones can get through a simple HTTP download.

The ”phonebook_creator_pbx.sh” does the same but includes the phone numbers for PBX operation. It also creates a crontab entry to run it automatically every day once (phonebooks do not change frequently).

Here, the webserver is a single point of failure. However, it is not time critical, and the network will work without it. Only new telephone books cannot be distributed during this outage. Each router still has the newest version to distribute to all telephones attached to it.

To avoid unnecessary files on our routers, the settings file can be used to restrict their generation. You find the “settings.txt” file in /arednstack/phonebook.

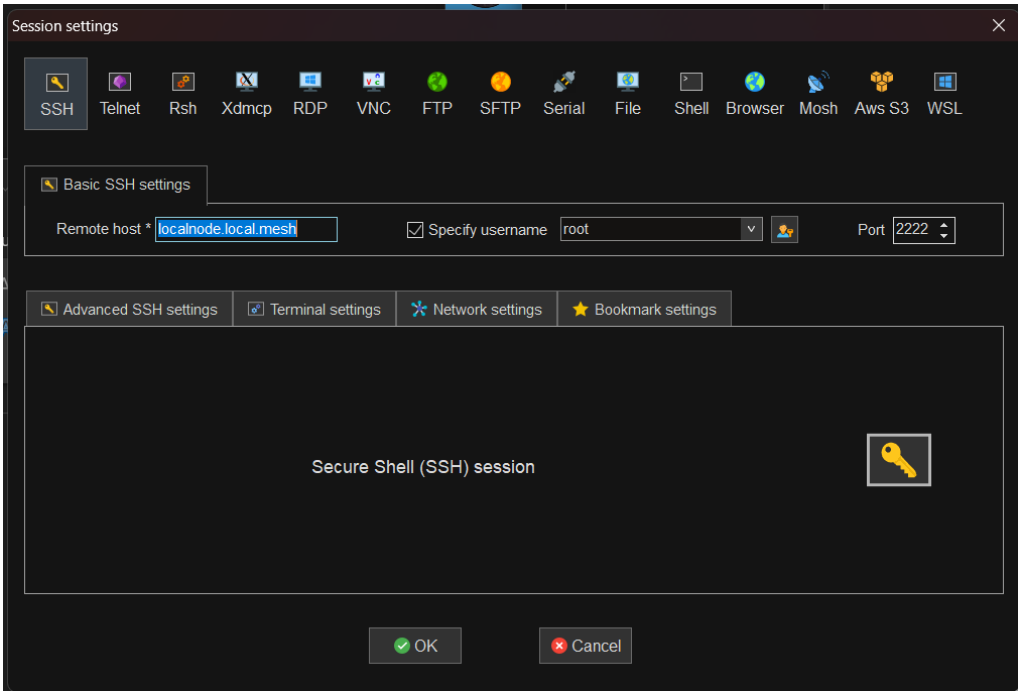
```
#Direct calling or PBX operation
download_directory_direct=YES
download_directory_pbx=YES

#Which Brands of phones are used on your router
create_yealink=YES
create_cisco=NO
create_noname=NO
```

Yes means that a file for the purpose is generated (example: download_directory_direct=YES means that you will find files for direct calling for all phones selected In the phones section)

Installation on the hap Router

First, you have to ssh into your router (address: localnode.local.mesh) using mobaXterm or PUTTY, or any other terminal program. Keep in mind to use port 2222.



Example: MobaXterm (https://mobaxterm.mobatek.net/download.html)

For the next steps, your hap Router needs to be connected to AREDN (SwissDigitalNet), usually via a tunnel.

Run phonebook_installer.sh:

```
curl http://hb9edi-apu-1.local.mesh:8080/filerepo/Phonebook/phonebook_installer.sh
| sh -s http://hb9edi-apu-1.local.mesh:8080/filerepo/Phonebook/
```

(Please replace “http://hb9edi-apu-1.local.mesh:8080/filerepo/Phonebook“ with the address of your web server)

Now you should have settings.txt on your router:

```
root@HB9BLA-HAP3-1:/arednstack/phonebook# ls -l
-rw-r--r--    1 root     root           186 Aug  8 17:45 settings.txt
root@HB9BLA-HAP3-1:/arednstack/phonebook#
```

In the /www directory, you should find all the requested xml files:

```
root@HB9BLA-HAP3-1:/www# ls *.xml -l
-rw-r--r--    1 root     root         11578 Jul 27 07:55 phonebook_yealink_direct.xml
-rw-r--r--    1 root     root         10123 Jul 27 07:55 phonebook_yealink_pbx.xml
```

(I only have Yealink phones, but I want to have direct and PBX calling phone books. You may have more or fewer files according to the definition of your settings.txt)

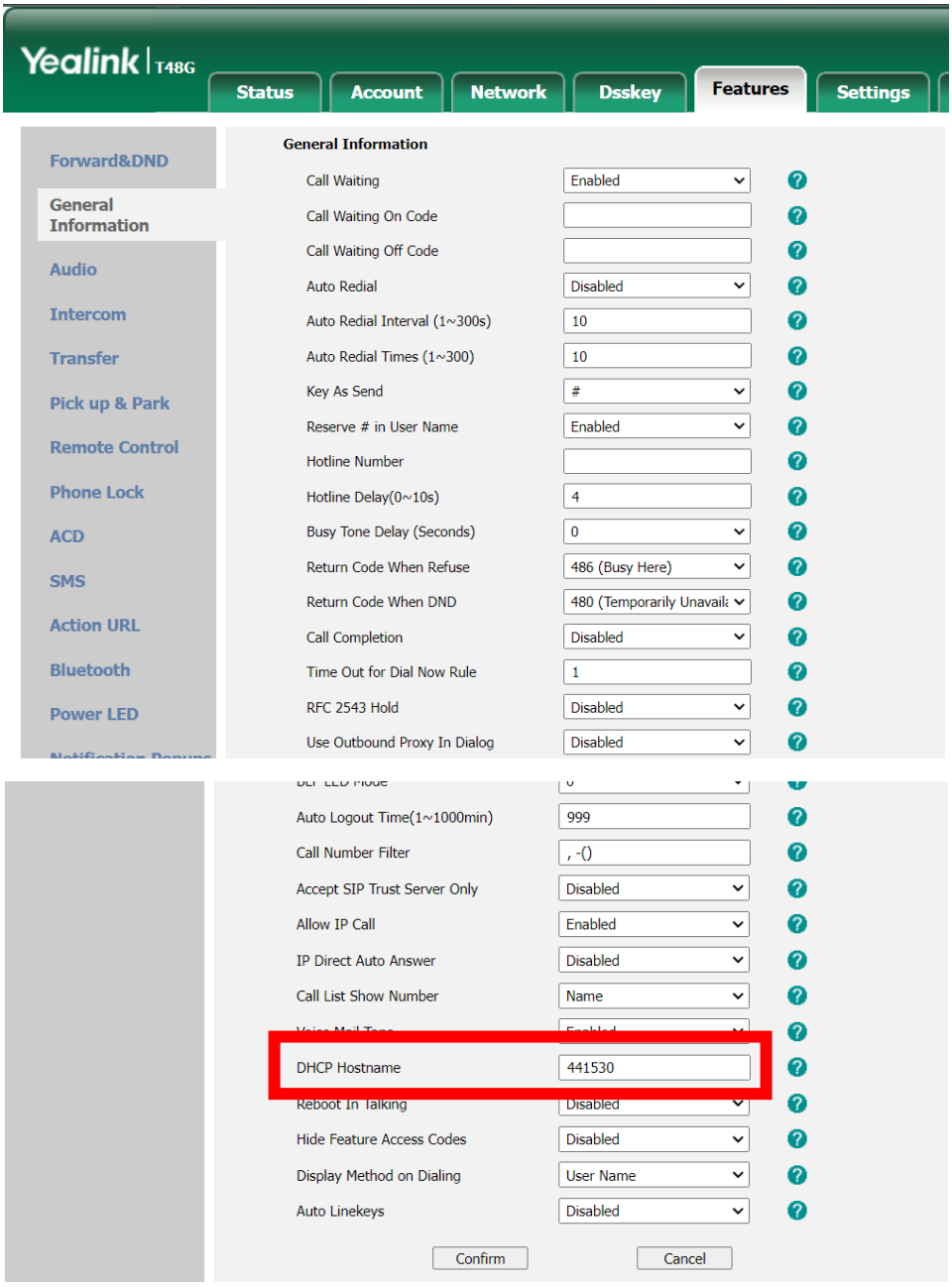
And in /etc/cron.daily/, you should see the download_phonebook file:

```
root@HB9BLA-HAP3-1:/etc/cron.daily# ls -l
-rwxr-xr-x    1 root     root            91 Aug  9 09:53 download_phonebook
-rwxr-xr-x    1 root     root          2129 Apr 27 22:28 update-clock
root@HB9BLA-HAP3-1:/etc/cron.daily#
```

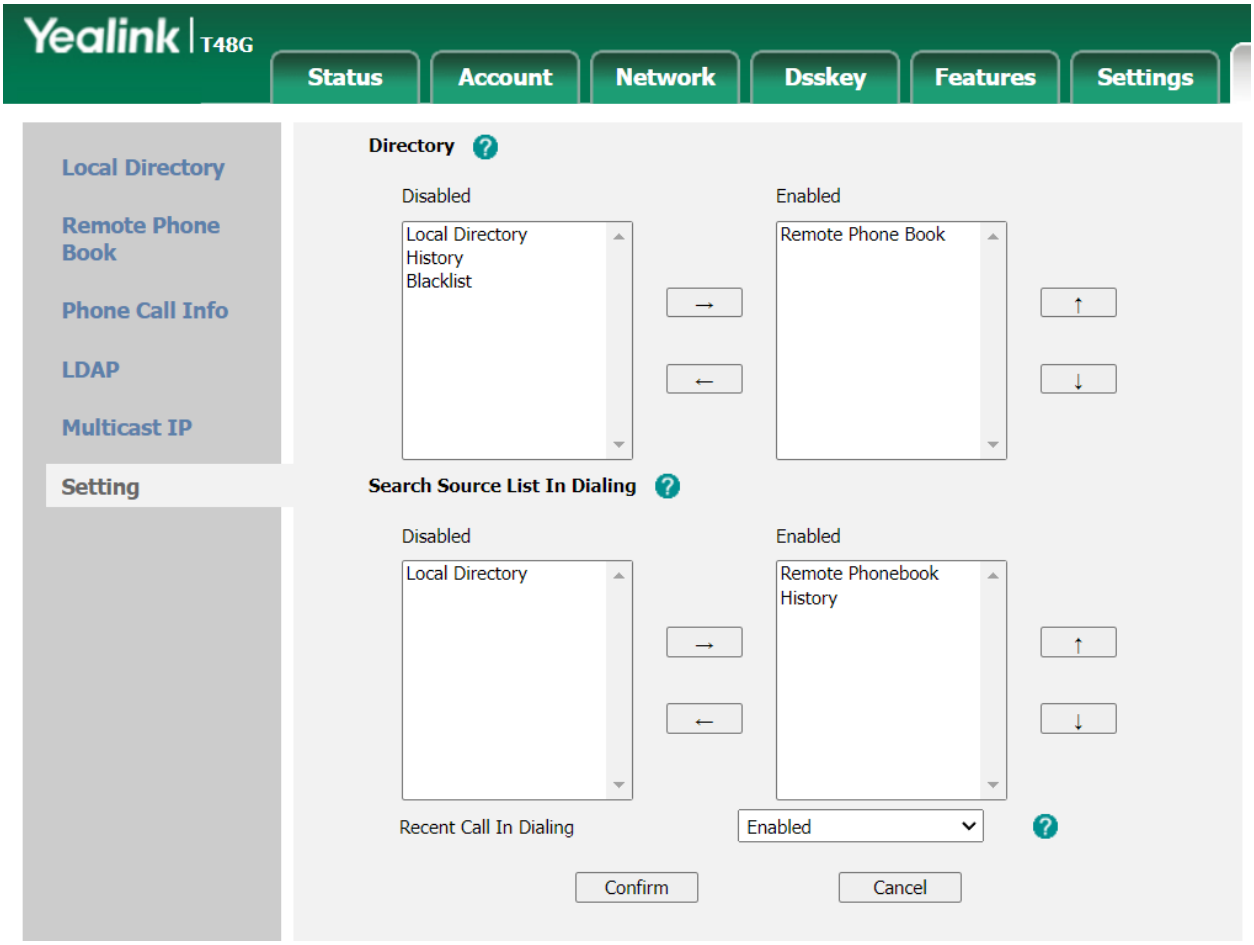
This job downloads the phonebook every day once. Because our routers do not support crontab, we had to use this trick.

Now we can go on with the phones.

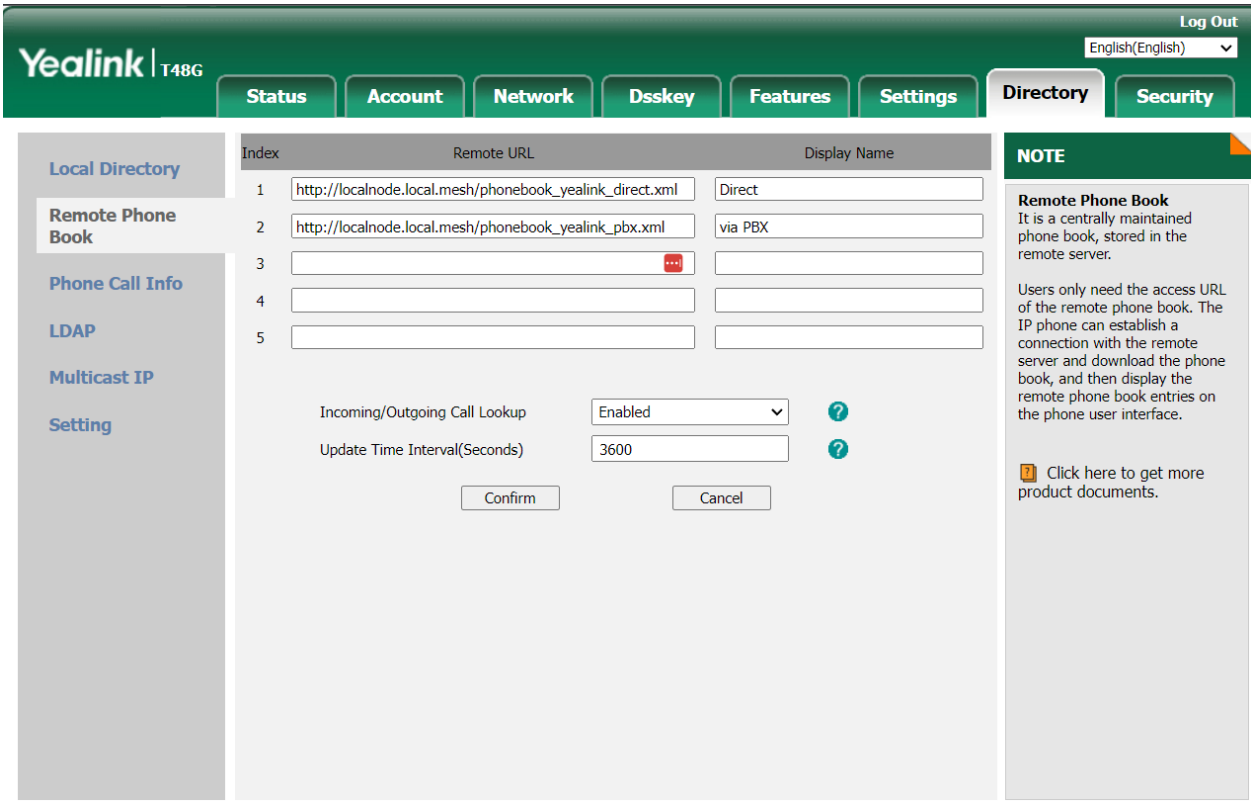
Parameters in Yealink Phones



Please make sure you have your phones ready. You find the respective document (AREDN_Setup_English.pdf) in the documentation folder of this project.



Add one or two files into the “remote phonebook” of your telephone. Make sure you do this only when the telephone is connected to a router where you installed the appropriate phonebooks



Here are the entries for copy-paste (you can use these addresses to test if your router installation is ok)

```
http://localnode.local.mesh/phonebook_yealink_direct.xml
http://localnode.local.mesh/phonebook_yealink_pbx.xml
```

The names change according to the brand of your phone. Currently, Cisco is supported as a test

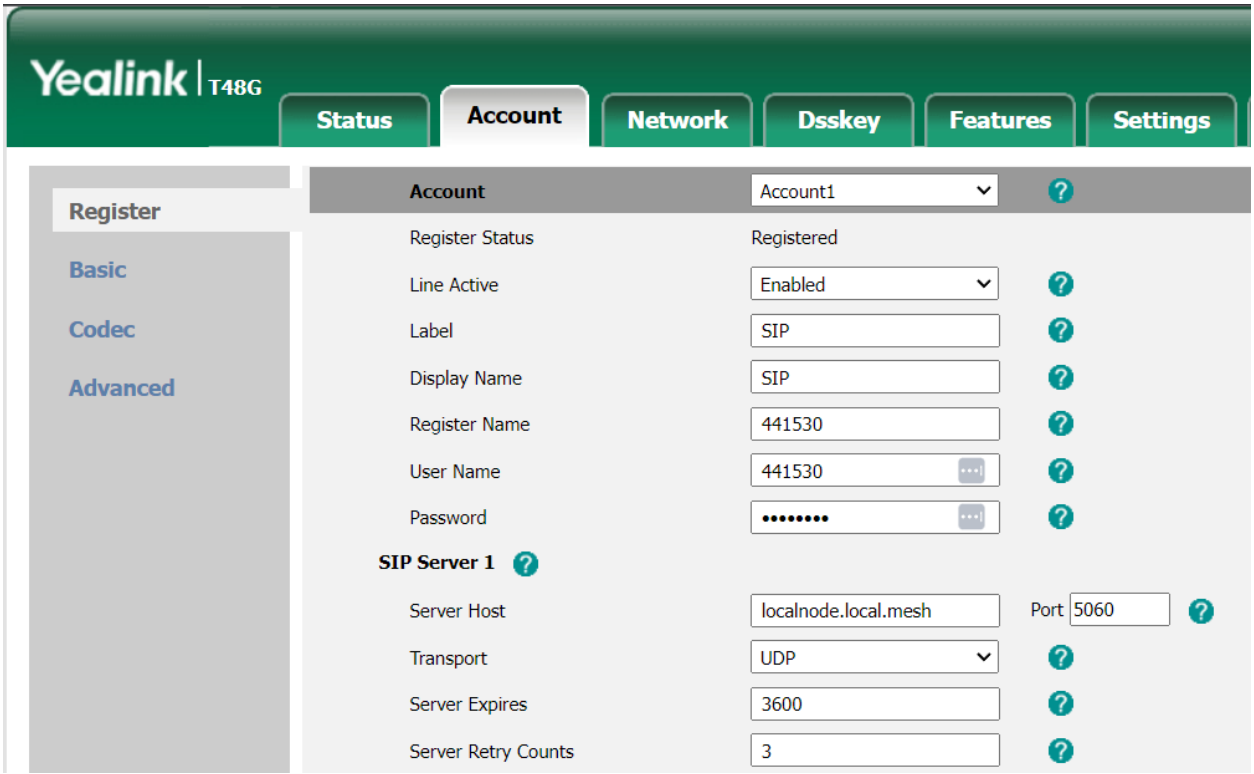
Install SIPproxd on your hap Router (only ac2 for the moment)

Yealink phones need to be registered to a SIP server to work properly with the phone books. If you have no PBX available, you can install SIPproxy on your hap router and connect your Yealinks to this server. If you want to use a PBX, configure the account as usual. Then, you have the single point of failure (if the connection to the PBX is lost).

```
curl http://hb9edi-apu-
1.local.mesh:8080/filerepo/Siproxd/SIProxd_installer_3.23.4_Mikrotik_HAP_lite.sh |
sh
```

(Please replace “http://hb9edi-apu-1.local.mesh:8080/filerepo/Phonebook “ with the address of your web server)

Add “localnode.local.mesh” as a server host in your phones:



Check if your phone is registered. The Username will be transferred to the other phone (not the alias)

Now your telephones attached to tis particular router should see the requested phone books and names should be shown when you get calls.

Appendix

Install Nano editor on hap ac3

https://downloads.openwrt.org/releases/19.07.0/packages/arm_cortex-a7_neon-vfpv4/base/terminfo_6.1-5_arm_cortex-a7_neon-vfpv4.ipk

https://downloads.openwrt.org/releases/19.07.0/packages/arm_cortex-a7_neon-vfpv4/base/libncurses6_6.1-5_arm_cortex-a7_neon-vfpv4.ipk

https://downloads.openwrt.org/releases/19.07.0/packages/arm_cortex-a7_neon-vfpv4/packages/nano_6.2-1_arm_cortex-a7_neon-vfpv4.ipk

Install SIPproxy auf ac3

curl http://hb9edi-apu-1.local.mesh:8080/filerepo/Siproxd/SIProxd_installer_3.23.4_Mikrotik_HAP3.sh | sh

#!/bin/ash

SIProxd_installer_3.23.4_Mikrotik_HAP3.sh

```
# $Author: Serge HB9EDI $
# $Date: 2023/07/22 $
# $Revision: 1.0 $


opkg install http://hb9edi-apu-1.local.mesh:8080/filerepo/Siproxd/libltdl7_2.4.6-2_arm_cortex-a7_neon-vfpv4.ipk
opkg install http://hb9edi-apu-1.local.mesh:8080/filerepo/Siproxd/libosip2_5.0.0-3_arm_cortex-a7_neon-vfpv4.ipk
opkg install http://hb9edi-apu-1.local.mesh:8080/filerepo/Siproxd/siproxd_0.8.3-1_arm_cortex-a7_neon-vfpv4.ipk


ln -s /etc/init.d/siproxd /etc/rc.d/S99siproxd
ln -s /etc/init.d/siproxd /etc/rc.d/K99siproxd


curl http://hb9edi-apu-1.local.mesh:8080/filerepo/Siproxd/siproxd.conf --output /etc/config/siproxd


/etc/init.d/siproxd start


ln -s /etc/init.d/siproxd /etc/rc.d/S99siproxd
ln -s /etc/init.d/siproxd /etc/rc.d/K99siproxd
```

siproxd.conf

```
config siproxd general
#       option interface_inbound lan
#       option interface_outbound wan

# Define low-level network devices, overriding interface_in/outbound:

option if_inbound br-lan
option if_outbound lo

option hosts_allow_reg 10.0.0.0/8
option hosts_allow_sip 10.0.0.0/8
```

```
nano /etc/config/siproxd
```

Change these lines:

```
option if_inbound br-lan
option if_outbound lo
```

```
/etc/init.d/siproxd status
```

Replication of Github Files

New phonebook from Google

```
http://hb9edi-apu-1.local.mesh:8080/cgi-bin/update_phonebook
```

Update all installation files

```
http://hb9edi-apu-1.local.mesh:8080/cgi-bin/update_phonebook_installer
```

Show result

```
http://hb9edi-apu-1.local.mesh:8080/filerepo/index.php?dir=Phonebook
```

Copy Google Sheet

```
curl -L "https://docs.google.com/spreadsheets/d/e/2PACX-1vTZw1cwlV6pdFETvC-JnI0gPwKRwR0rBUc2XqX9V3LV1NfrB0zvhhWKmrYVS1eippbs91lMLfkeXj6-/pub?output=csv" -o /www/filerepo/Phonebook/AREDN_Phonebook.csv
```

Show software release

```
cat /etc/openwrt_release | grep DISTRIB_RELEASE | sed -n "s/^.*'\(.*\)'.*$/\1/ p"
```

Show CPU architecture

```
cat /etc/openwrt_release | grep DISTRIB_ARCH | sed -n "s/^.*'\(.*\)'.*$/\1/ p"
```