1. Proxysmart manual.

1. Brief details

I have developed a software that allows you to run your own 4g proxy farm. It runs on a Linux box (PC\laptop) with USB hub and the modems.

Functions:

- IP resets on modems
- WebApp for checking status of each modem
- WEB\CLI API for actions like querying status, IP rotation, getting used bandwidth for the day\month, running speedtests
- setting bandwidth quota per modem per month
- bandwidth throttling per modem
- exposing proxy ports, so they are available from world wide
- reading\sending SMS and USSD
- OS spoofing, to simulate TCP fingerprints of: MacOS \ iOS \ Windows \ Android
- custom MTU\TTL per modem
- proxy ACLs (what to allow/deny to proxy users)

Basic configuration.

Variables are set `/etc/proxysmart/conf.txt`.

Each variable has brief description in place.

2. Adding a new modem,

- remove PIN from the modem's SIM card and plug in the modem into USB port or USB hub.
- Check whether your modem Web App (e.g. Huawei's Hilink) requires authentication, and if it does, set its admin password to `admin123`. Basically to the value of `$DEFAULT_HILINK_ADMIN_PASSWORD` variable in `/etc/proxysmart/conf.txt`. Otherwise many functions will not work, and its IMEI will be detected similarly to `2-1.1.2`

if modern WebApp is used with Mongodb backend (default)

- Plug in the modem
- wait ~5 minutes or run `proxysmart.sh reset_gently`
- the modem will appear in the WebApp, click EDIT on it, then click APPLY
- refresh the WebApp
- done!

if old style map.txt is used:

- **if you know its IMEI.**

In that case you will assign some specific proxy credentials to it.

- detect IMEI for the modem (check its back side).
- edit `/etc/proxysmart/map.txt` , add new line with new modem, IMEI and proxy ports, login & pass. See 1st line in the file with fields names.
- run `proxysmart.sh reset_gently`. It will detect it & apply settings.
- run `proxysmart.sh show_status` to confirm new creds applied.

- **if you don't know its IMEI.**

In that case some random proxy credentials will be assigned.

- make sure `GENERATE_RANDOM_PROXY_CREDS=1` in `/etc/proxysmart/conf.txt` . It is the default setting.
- run `proxysmart.sh reset_gently`. It will detect new modem & apply random proxy credentials to it. Autogeneration allocates ports in a fully random manner. Autogeneration is only useful to demonstrate the modems are working. It is not supposed to be used on a regular basis.
- run `proxysmart.sh show_status` to confirm new modem is detected. It will have random creds.

- if you want to make them static, edit `/etc/proxysmart/map.txt`, add the modems there, and then run `proxysmart.sh apply_settings_for_a_modem_by_imei_raw Imei` where Imei is Imei of the modem. Then proxy credentials from the map.txt will be applied
- run `proxysmart.sh show_status` to confirm new creds applied.

3. Proxy credentials for new modems

When adding new modems, please use

- unique HTTP ports from 8001 to 8999,
- unique SOCKS ports from 5001 to 5999.

If you want different ports ranges, update `firewall.conf` accordingly.

- please use unique nicknames like `dongleXXX` or whatever else. Don't use nicknames like `randomXXX`, that are assigned automatically.

4. Where is WebApp

One of

- http://localhost:8080/
- http://LAN_IP:8080/
- http://VPS_IP:8080/

By default login/password are `proxy` / `proxy`.

5. How to use proxies

- If proxy ports are forwarded via remote cloud VPS: then the proxies can be used from all over the Internet, by that VPS IP and proxy port numbers.
- From the same LAN where multimodem server is located: by the server's LAN IP and proxy port numbers.

6. Get list of all modems & their external IPs

Run: `proxysmart.sh show_status` for table-alike output.

7. Reconfigure all modems & proxies.

Run: `proxysmart.sh reset_complete`

It is done after reboot automatically by a Cron job.

8. How to change proxy credentials for a modem. How to rename a modem.

WebApp method

- click EDIT on a modem, set new port or password or nickname for a modem
- click APPLY

Old map.txt method

- Edit the map file (`/etc/proxysmart/map.txt`), set new port or password or nickname for a modem.
- Then either run `proxysmart.sh apply_settings_for_a_modem_by_imei_raw 9999999999999` or click APPLY in the WebApp

9. Reset (change) IP on a modem.

The options are below.

- **From Web App**

Click `Reset Ip` button.

- **From command line.**

Run: `proxysmart.sh reset_quick_nick dongle1`

Where `dongle1` is a Dongle "nickname" that is seen from output of `proxysmart.sh show_status`

- **From Web API.**

check WEB API section of this manual.

How to rotate a modem periodically?

- METHOD1 (new)

Update modem's settings in the WebApp and click APPLY.

- METHOD2 (old)

For global setting, edit `/etc/proxysmart/conf.txt` and set `AUTO_IP_ROTATION=5` in order to rotate each modem every 5th minute. If set to 0, automatic IP rotation is not done. You can also set hourly rotation, set 120 for every 2h rotation.

Also individual rotation intervals can be set in the Web App, or in `/etc/proxysmart/per_modem_conf.yaml` or in Mongodb , the key is `AUTO_IP_ROTATION`.

`/etc/proxysmart/per_modem_conf.yaml` or Mongodb takes takes precedense over global setting in `/etc/proxysmart/conf.txt`.

- METHOD3 (old)

Install a Cron job. Edit a file `/etc/cron.d/proxysmart`, add a line (or uncomment a commented line..)

```
*/10 * * * * root run-one /usr/local/bin/proxysmart.sh reset_quick_nick dongle3
```

so that a modem with the Nickname dongle3 is rotated every 10 min.

Repeat for each modem you want to rotate periodically.

10. How many modems can I run on a single computer?

Hi , technically it depends on how powerful this PC is, and how intensively proxies are used.

- Raspberry PI - 4 proxies (roughly)
- a miniPC (Intel NUC or similar) - up to 10
- a Laptop like Core i5 - up to 30.

Also it depends on what Plan you buy.

Also it depends on USB configuration, for maximum number of modems:

- disable USB3.0 in BIOS
- use USB2.0 hubs

11. How to set TTL and why?

In some cases custom TTL must be set in order to have Cell Operator think we are not using the modem in hotsport \ tethering mode. I.e. we don't share its data. By default Linux OS has ttl = 64. To change Cell Operator perception of the situation, we want to set it +1 i.e. 65.

Edit `/etc/proxysmart/conf.txt` and set `CUSTOM_TTL_SET=1` and `CUSTOM_TTL_VALUE=65` and regenerate settings.

12. How to set MTU and why?

In some cases different MTU values connect with different types of ISP's. You may want to change it.

Mtu can be only lowered. E.g. if you have MTU 1390, you can set 1340. Not opposite.

- Option 1. One value for all modems.

Edit `/etc/proxysmart/conf.txt` and set `CUSTOM_MTU_SET=1` , `CUSTOM_MTU=1410`.

- Option 2. Individual values for modems.

The same as above , but also edit `/etc/proxysmart/per_modem_conf.yaml` and add `mtu` value for some modems that need custom value.

13. How to set extra settings for a modem.

Those are optional and are set in YAML file `/etc/proxysmart/per_modem_conf.yaml` or in MongoDB.

- allowed customers IP's who are not required to type in proxy password (IP-based auth). Those are set in `white_list` array.
- bandwidth (speed) limit. Values are in bits per second. Set them in `bandlimin` and `bandlimout`. E.g. for 2/2 mbps it will be 2000000/2000000.
- DENIED_SITES_ENABLE (1 or 0) and DENIED_SITES_LIST (list of blocked sites patterns), see examples .
- bw_quota , value is in Megabytes
- mtu , ttl

After changing the file or MongoDB record, apply setting for the modem you changed settings for.

14. How can I access the web interface admin panel of each modem?

Just visit what you are seeing as GW (it is a modem IP) via corresponding proxy.

14.1. How can I prevent access to modems web interface via proxy?

Edit `/etc/proxysmart/conf.txt` and set

```
PROXY_ADMIN_ENABLE=1
PROXY_ADMIN_LOGIN=admin
PROXY_ADMIN_PASS=papapa
```

And regenerate configs. So only **admin** user is allowed to use modems web interfaces, and normal proxy users are not.

15. How to set monthly traffic quota per modem?

In the WebApp, set monthly traffic quota. Click EDIT & APPLY.

Old method (when YAML files are used):

Edit `/etc/proxysmart/per_modem_conf.yaml` and add `bw_quota` value for some modems that need custom value.

E.g. a line is below, where the modem that IMEI has **2000 Megabytes** monthly quota, from begin to the end of the month. It is applied to both Upload and Download.

```
-
    imei:   777777777777777
    bw_quota:      2000
```

16. How to make my proxes Open (i.e. not requiring authentication)

Set `OPEN_PROXIES=1` in `/etc/proxysmart/conf.txt` and regenerate all configs.

Note, when proxy ports are forrwarded via a VPS, the proxies are available to any internet user. Use it with caution.

17. Get monthly/daily proxy usage.

Click `bandwitdh stats` in the WebApp, or run `proxysmart.sh bandwidth_report_json dongleXXX`, you will see these columns:

- "bandwidth_bytes_day_in"
- "bandwidth_bytes_day_out"
- "bandwidth_bytes_month_in"
- "bandwidth_bytes_month_out"
- "bandwidth_bytes_yesterday_in"
- "bandwidth_bytes_yesterday_out"

Also reports are stored in `/var/lib/3proxy/reports/`. Files are named like `report.$IMEI.YYYY.MM.DD`

18. How to get current number of connections for a modem?

Run a command

```
ss -o state established | grep -c :8038
```

But change 8038 with HTTP port of a desired proxy

19. How to read SMS from a modem.

You have these options.

1. Browse to the modem IP (it is shown as GW in `proxysmart.sh show_status`) through the proxy. Click SMS button.
2. run `proxysmart.sh list_sms_for_a_modem_by_imei_json 999999999999999` i.e. IMEI of required modem.
3. Click SMS in the WebApp

20. How to change WebApp password for http://localhost:8080/

By default it is set to `proxy` / `proxy`. The password sits on the server's folder `/etc/nginx/`. It Can be updated from the Terminal , with the command as follows:

```
sudo htpasswd -b /etc/nginx/htpasswd proxy NewAweSomePassword999999
```

Then it will ask for password for current Ubuntu user.

21. OS Spoofing

Os Spoofing is used to simulate other OS TCP fingerprints, MacOS \ iOS \ Windows \ Android

How to enable OS Spoofing?

It applies to all modems at once.

- set `OS_SPOOF=1` in `/etc/proxysmart/conf.txt`.
- set `OSGENRE` and `DETAILS_P0F` to one of these (run `osfooler-ng -p` for full list):

```
android   1
android   3
ios   1
ios   2
macosx   1
macosx   2
macosx   3
macosx   4
windows   1
windows   2
windows   3
```

- run: `proxysmart.sh reset_complete`

How to test OS Spoofing ?

Visit one of these websites (IP checkers) through a proxy. Find something like "OS TCP fingerprints".

- http://witch.valdikss.org.ru/
- https://thesafety.us/
- https://Whoer.net , extended results
- https://browserleaks.com/ip

What OS can I spoof?

MacOS \ iOS \ Windows \ Android

Can I dump OS TCP fingerprint from a real device and use it?

Yes, contact me.

I enabled OS TCP spoofing, but it is not working!

The reason may be that the operator passes all traffic through its internal proxy, or in other way modifies TCP signatures. Then local OS TCP modifications are overwritten. Is it bad? No! Because still traffic looks natural as it was coming from this operator network.

Try other operator.

22. Performance tuning

When >10 modems are added, and when modem list is generated slowly, play with `MAX_PARALLEL_WORKERS_STATUS` variable, e.g. set it to 2 or 4. On faster CPU's it can be set to 8.

Also try to disable OS TCP reporting, i.e. set `ENABLE_VALDIK=0` in `/etc/proxysmart/conf.txt`. It will also make modem list generation faster.

Also you can disable detailed status, set `QUICK_STATUS=1` in `/etc/proxysmart/conf.txt` & refresh the WebApp.

23. How to get more cellular IP's?

Sometimes 3G has another IP pool, but speeds are lower. You can set random Auto(4G),3G rotation method. Check `/etc/proxysmart/conf.txt` for examples.

24. What if a modem connected via 3G or 2G, and I want 4G?

Rotate its IP.

25. I want to add extra users to a proxy

WebApp method (new)

Click EDIT on a modem, add some extra users, click APPLY.

map.txt method (old)

Add them to `per_modem_conf.yaml`, check the template. Basically each modem may have an array of extra users with user:password definition.

Then apply setting for the modem.

26. Is IPV6 supported?

Yes but it's off by default.

On modems , edit APN and set APN type for both IPv4 and IPv6 , e.g. Ip4Ip6 or Ip4+ip6, there is a dropdown list for that.

On Proxysmart box: Update `/etc/proxysmart/conf.txt` with

- `ALTNETWORKING_VERSION=2`
- `IPV6_SUPPORT=1`

and reset configuration `proxysmart.sh reset_complete` ; or even better do a reboot.

27. Nagios\Naemon integration.

There is a plugin embedded, run it as root,

```
/usr/lib/nagios/plugins/proxysmart-nagios-helper.sh IMEI
```

or

```
/usr/lib/nagios/plugins/proxysmart-nagios-helper.sh NICKNAME
```

so it will return OK/WARN/CRIT/UNKNOWN and corresponding exit code.

28. Secure (anonymous) IP rotation links.

These links

- Can be safely passed to your customers. They don't reveal real dongle parameters like IMEI or Nickname.
- They don't require HTTP basic authentication
- They have limited lifetime , it is set in `/etc/proxysmart/conf.txt` as `RESET_LINK_VALIDITY` variable, e.g.: `RESET_LINK_VALIDITY=2hour` or `RESET_LINK_VALIDITY=1week` .

A link can be retrieved this way: Open dongle status (click on its IMEI!) in the WebApp, take `RESET_SECURE_LINK->URL` value.

If you realized you gave a link to a customer, and want to revoke it, just set new password for the proxy.

If you want to invalidate all links of all modems, set a new secret: set `RESET_LINK_SECRET` in `/etc/proxysmart/conf.txt` .

29. QUIC support on Socks5 proxies, for HTTP/3.0

It is needed for proper work of HTTP/3.0 which uses UDP.

- Proxysmart box :

set in `/etc/proxysmart/conf.txt` : `QUIC_SUPPORT=1` and run `proxysmart.sh reset_complete`.

Then QUIC is supported for LAN clients.

- VPS :

if Haproxy is not installed, do nothing.

if Haproxy installed: free up SOCKS ports (5xxx) from Haproxy.

Run on VPS:

```
echo 'fwd  ALL=NOPASSWD:  ALL' >  /etc/sudoers.d/proxysmart
chmod 400 /etc/sudoers.d/proxysmart
usermod -s /bin/bash fwd
```

Then QUIC is supported for world-wide clients.

29. "Dirty" IP reset.

It may be needed when you need even faster IP reset. In this case, post-checks are not made, so it is not sure if the modem really went online after IP reset. It can be activated by `DIRTY_IP_ROTATION=1` in `/etc/proxysmart/conf.txt`.

30. Exclude some modems

In `/etc/proxysmart/conf.txt`

- by Device name, populate this array `IGNORED_DEV=( modem132 modem0000000002)` -- array of Network Interfaces that are not processed
- by IMEI, populate this array `IGNORED_IMEI=( 9999999999999999 8888888888888888 )` -- array of IMEI that are not processed

31. Use custom Speedtest server.

It is useful when for some reason you want to run speed tests towards a custom server, instead of Ookla servers. So set up a Apache web server with a large file (500MB) and get 2 URL's, one will test download and 2nd will test upload. The latter must accept large POST data.

The commands to setup a server part

```
apt install apache2
dd if=/dev/urandom  of=/var/www/html/file.bin bs=1M count=500
```

Update `/etc/proxysmart/conf.txt` with IP of the WEB server:

```
SPEEDTEST_CUSTOM=1
DL_URL=http://5.2.7.8/file.bin
UL_URL=http://5.2.7.8/i.php
```

---

2. Project description

1. project architecture (clients, servers, websites),

- onsite: box with Ubuntu, USB hub and modems
- remote: VPS with proxy ports (optional)

2. what is hosted where and from which repository,

Online services are used:

- http://ip.tanatos.org/ip.php which is simple PHP script that returns visitor's IP. It is used to detect whether a modem is really online. Can be replaced with one of https://ifconfig.co or similar, but I was not happy with their reliabiality, they are down sometimes. The URL is defined in `/etc/proxysmart/conf.txt`.
- http://witch.valdikss.org.ru/ : used for detecting p0f and MTU

Software used to build the box:

- 3proxy 0.8.13 https://github.com/z3APA3A/3proxy/ ; forwards requests from clients to modems; does authentication; provides HTTP/SOCKS services
- hlcli https://github.com/kenshaw/hilink ; gathers info from Huawei modems
- HiPi https://metacpan.org/pod/HiPi ; perl module, API for gathering info from Huawei modems and resetting IP's on them
- J2cli https://pypi.org/project/j2cli/ ; Python module for building 3proxy conf files
- Imgur image hosting for uploading screenshots
- everything else is in official Ubuntu repo (APT).

3. how all these elements communicate with each other,

- Ubuntu box initializes modems, creates Linux networking routing for each.
- 3proxy instance is started for each.
- Proxy ports are exposed to VPS.

4. what are common points of failure and how to deal with them,

- a modem is not recognized => insert it into Win/Mac and check there. Probably there is an issue with SIM.
- IP can't be rotated => check whether you completed setup of the modem in Huawei Web Gui (192.168.8.1) on Windows/Mac and set its admin password to `admin`

5. how to update each and every project element, what is the workflow, how to build and deploy it, what tools to use,

- 3proxy - I am not seeing a reason to update it, because newer releases may bring some incompatibilty, but anyways: https://3proxy.ru/howtoe.asp#GCCUNIX or check README in the project Github.

- Ubuntu version - it is better not update it, i.e. stick with same version. Security updates are good to apply.

---

3. CLI API

1. show status

Show full status of all modems, table (slower).

```
# proxysmart.sh  show_status

+--------+----+---------+-------+----------------+-----+-------------+-----------+-------------+------+------------+----+
| NICK   | N  | DEV     | MODEL | IMEI           | HTTP| LOCAL_IP    |GW         |EXT_IP       |ONLINE| CELL:MODE  | MSG|
+--------+----+---------+-------+----------------+-----+-------------+-----------+-------------+------+------------+----+
| dongle1| 0  | modem0  | E3372h| 862329099999999| 8001| 192.168.8.100|192.168.8.1|46.216.113.63|yes   | MTS BY:LTE |    |
| dongle2| 114| modem114| E3131 | 352221099999999| 8002| 192.168.8.100|192.168.8.1|             |no    | :NO_SERVICE|    |
+--------+----+---------+-------+----------------+-----+-------------+-----------+-------------+------+------------+----+
items TOTAL 2
```

Show brief status of all modems, table, (faster)

```
| NICK    | N | DEV    | IMEI           | HTTP| LOCAL_IP     | GW         | EXT_IP        | ONLINE| MSG|
| Cdongle2| 77| modem77| 862329099999999| 8002| 192.168.8.100| 192.168.8.1| 46.216.152.241| yes   |    |
| Client5 | 93| modem93| 352221099999999| 8004| 192.168.0.100| 192.168.0.1| 46.56.186.34  | yes   |    |
```

Show full status of all modems , json

```
# proxysmart.sh  show_status_json

[
   {
      "MSG" : "",
      "N" : "0",
      "modem_details" : {
         "HUB_ID" : "1-1",
         "HUB_PORT" : "1-1",
         "IMEI" : "899999999999999",
         "MODEL" : "E3372h-320",
         "NICK" : "dongle1"
      },
      "net_details" : {
         "CELLOP" : "MTS BY",
         "ConnectionStatus" : "(901) DATA:connected",
         "CurrentNetworkType" : "(101) LTE",
         "DEV" : "modem0",
         "EXT_IP" : "46.216.113.63",
         "GW" : "192.168.8.1",
         "IS_ONLINE" : "yes",
         "LOCAL_IP" : "192.168.8.100",
         "SimStatus" : "(1) valid SIM card",
         "VALDIK" : "Detected OS = Linux 2.2.x-3.x [generic];MTU = 1420;Network link =
                generic tunnel or VPN;PTR test = Probably home user;
                Fingerprint and OS match. No proxy detected ;No OpenVPN detected.",
         "workmode" : "LTE"
      },
      "proxy_creds" : {
         "HTTP_PORT" : "8001",
         "LOGIN" : "alice",
         "PASS" : "cool",
         "SOCKS_PORT" : "5001"
      },
      "redirector_status" : {
         "ActiveState" : "active",
         "NRestarts" : "0",
         "SubState" : "running",
         "UPTIME" : "2min 6s"
      }
   },
   {
      "MSG" : "",
      "N" : "1142",
      "modem_details" : {
         "HUB_ID" : "1-3",
         "HUB_PORT" : "3-1",
         "IMEI" : "352228888888888",
         "MODEL" : "E3131",
         "NICK" : "dongle2"
      },
      "net_details" : {
         "CELLOP" : null,
         "ConnectionStatus" : "(902) DATA:disconnected",
         "CurrentNetworkType" : "(0) NO_SERVICE",
         "DEV" : "modem1142",
         "EXT_IP" : null,
         "GW" : "192.168.8.1",
         "IS_ONLINE" : "no",
         "LOCAL_IP" : "192.168.8.100",
         "SimStatus" : "(255) SIM card is missing",
         "VALDIK" : null,
         "workmode" : "unknown"
      },
      "proxy_creds" : {
         "HTTP_PORT" : "8002",
         "LOGIN" : "alice",
         "PASS" : "cool",
         "SOCKS_PORT" : "5002"
      },
      "redirector_status" : {
         "ActiveState" : "active",
         "NRestarts" : "13",
         "SubState" : "running",
         "UPTIME" : "1s"
      }
   }
]
```

Show status for a single modem, JSON

```
# proxysmart.sh  show_single_status_json dongle111
[
   {
      "IS_LOCKED" : "false",
      "MSG" : "",
      "N" : "115",
      "modem_details" : {
         "HUB_ID" : "1-1",
         "HUB_PORT" : "3",
         "IMEI" : "899999999999999",
         "MODEL" : "E3372h-320",
         "NICK" : "dongle111",
         "UDEV_UPTIME" : "1212172",
         "UPTIME" : "14 days + 43.883333 minutes"
      },
      "net_details" : {
         "CELLOP" : "MTS BY",
         "ConnectionStatus" : "901, DATA:connected OK",
         "CurrentNetworkType" : "(101) LTE",
         "DEV" : "modem115",
         "EXT_IP" : "46.216.224.164",
         "GW" : "192.168.8.1",
         "IS_ONLINE" : "yes",
         "LOCAL_IP" : "192.168.8.100",
         "SIGNAL_STRENGTH" : "4",
         "SimStatus" : "(1) valid SIM card",
         "VALDIK" : "Detected OS = Linux 2.2.x-3.x [generic];MTU = 1420;Network link = generic tunnel or VPN;
                PTR test = Probably home user;Fingerprint and OS match. No proxy detected ;No OpenVPN detected.",
         "workmode" : "LTE"
      },
      "proxy_creds" : {
         "HTTP_PORT" : "8004",
         "LOGIN" : "alice",
         "PASS" : "cool",
         "PROXYSTDLINE_LAN" : "192.168.100.2:8004:alice:cool",
         "PROXYSTDLINE_WWW" : "forwarding_disabled",
         "SOCKS_PORT" : "5004"
      },
      "redirector_status" : {
         "MSG" : "redirectors disabled globally"
      }
   }
]
```

2. full reconfiguration

```
# proxysmart.sh reset_complete

= old fake default route deleted
= adding faked default route
= restoring default linux TTL
net.ipv4.ip_default_ttl = 64
=start_connections

= lock acquired on DEV modem0
== [add_individual_dev] generating config for DEV=modem0 N=0 HUB_ID=1-1 HUB_PORT=1-1
= found IP 192.168.8.100 on modem0
= got GW=192.168.8.1
= got IMEI=899999999999999
= got model E3372h-320
= got NICK=dongle1
= start 3proxy config generation for N=0
= applying new settings:  DEV modem0, N 0, IMEI 899999999999999, nick dongle1
== starting proxy 0 on modem0
== starting redirector@0 on modem0
= lock released on DEV modem0

= lock acquired on DEV modem1142
== [add_individual_dev] generating config for DEV=modem1142 N=1142 HUB_ID=1-3 HUB_PORT=3-1
= found IP 192.168.8.100 on modem1142
= got GW=192.168.8.1
= got IMEI=352228888888888
= got model E3131
= got NICK=dongle2
= start 3proxy config generation for N=1142
= applying new settings:  DEV modem1142, N 1142, IMEI 352228888888888, nick dongle2
== starting proxy 1142 on modem1142
== starting redirector@1142 on modem1142
= lock released on DEV modem1142

= finding a live modem for setting as fallback default gateway
= testing modem0
= got EXT_IP 46.216.113.63
== marking that modem as LIVE, as tested
= setting fallback default gateway via modem dongle1 // modem0 // 192.168.8.1

it took 10 seconds to prepare the system

all found modems initialized. after 5 sec delay the status will be shown
+--------+----+---------+-------+---------------+-----+-------------+------------+-------------+-------+------------+----+
| NICK   | N  | DEV     | MODEL |IMEI           | HTTP| LOCAL_IP    | GW         |EXT_IP       | ONLINE| CELL:MODE  | MSG|
```

```
+--------+----+---------+-------+---------------+-----+-------------+-----------+------------+-------+-----------+----+
| dongle1| 0  | modem0  | E3372h|899999999999999| 8001| 192.168.8.100| 192.168.8.1|46.216.113.63| yes   | MTS BY:LTE |    |
| dongle2| 114| modem114| E3131 |352228888888888| 8002| 192.168.8.100| 192.168.8.1|             | no    | :NO_SERVICE|    |
+--------+----+---------+-------+---------------+-----+-------------+-----------+------------+-------+-----------+----+
items TOTAL 2
```

3. apply setting for a modem by IMEI

JSON output

```
# proxysmart.sh   apply_settings_for_a_modem_by_imei  868723023562406
{
   "debug" : "= lock acquired on DEV modem0,...",
   "message" : "",
   "result" : "success"
}
```

Plain text output.

```
 proxysmart.sh  apply_settings_for_a_modem_by_imei_raw    359999999999999
= lock acquired on DEV modem93
= start 3proxy config generation for N=93
= applying new settings:  DEV modem93, N 93, IMEI 359999999999999,
    nick Client5, http_port 8004, socks_port 5004, auth: alice / cool
= found ALLOWED_CLIENT_IPS=22.22.22.22,22.22.22.11
= got BANDLIMIN 12222
= got BANDLIMOUT 1444444
= got BW_QUOTA 20
= extra users detected: myuser1 : mypassword1,myuser2 : mypassword2
= purging old MTU rules from Iptables for modem N=93
deleted rule 8 from mangle/OUTPUT
= purging old MTU rules from Iptables for modem N=93
= adding MTU rules to Iptables for modem N=93 MTU=1400 MSS=1360
= starting redirector@93 on modem93
= lock released on DEV modem93
```

4. reset IP on a modem

Args: IMEI or NICKNAME.

JSON output:

```
# proxysmart.sh   reset_modem_by_imei    899999999999999
{
   "message" : "external ip changed from 46.216.188.74 to 46.216.113.63",
   "ext_ip" : "46.216.113.63",
   "result" : "success",
   "debug" : "= lock acquired on DEV modem0,= resetting DEV modem0 ..."
}
```

Plain text output:

```
# proxysmart.sh  reset_quick_nick  Client5

= lock acquired on DEV modem93
= resetting NICK Client5 DEV modem93 local IP 192.168.0.100 N 93 GW 192.168.0.1 IMEI 359999999999999
= external IP is 46.56.178.172
=stopping redirector N 93
...
=DNS test attempt 2/7 to DNS server 1.1.1.1
Checking/setting forced routing config (skip with /etc/proxysmart/altnetworking.sh -s ...)
Applying net_cls class identifier 0x0010093 to cgroup cgproxy93
Unset reverse path filtering for interface "all"
Unset reverse path filtering for interface "modem93"
DNS OK - 0.092 seconds response time
= passed
= restarting proxy@93 to definitely drop old connections..
= starting redirector N 93
=now detect EXT_IP
= external IP is 46.56.181.222
= purging old MTU rules from Iptables for modem N=93
deleted rule 9 from mangle/OUTPUT
= purging old MTU rules from Iptables for modem N=93
= adding MTU rules to Iptables for modem N=93 MTU=1400 MSS=1360
==save report:
start_time=2022-05-29@21:14:43 end_time=2022-05-29@21:15:13
    total_time=27 old_ip=46.56.178.172        new_ip=46.56.181.222 target_mode=auto
= lock released on DEV modem93
```

5. reboot a modem

Args: Nickname or IMEI.

```
#  proxysmart.sh   reboot_modem dongle61_us
and
#  proxysmart.sh   reboot_modem 899999999999999
```

6. Run speedtest on all modems at once

```
# proxysmart.sh  speedtest all
[
   {
      "IMEI" : "352228888888888",
      "N" : "1142",
      "NICK" : "dongle2",
      "test" : {
         "msg" : "some_error"
      }
   },
   {
      "IMEI" : "899999999999999",
      "N" : "0",
      "NICK" : "dongle1",
      "test" : {
         "download" : "5.9mbps",
         "share" : "http://www.speedtest.net/result/11130520118.png",
         "upload" : "12.3mbps"
      }
   }
]
```

7. report bandwitdh

On a single modem.

Args: NICKNAME or IMEI.

```
# proxysmart.sh  bandwidth_report_json  869076043182393
[
   {
      "IMEI" : "869076043182393",
      "NICK" : "dongle2",
      "bandwidth_bytes_day_in" : "3482408",
      "bandwidth_bytes_day_out" : "460261",
      "bandwidth_bytes_lifetime_in" : "16MB",
      "bandwidth_bytes_lifetime_out" : "4.9MB",
      "bandwidth_bytes_month_in" : "18163459",
      "bandwidth_bytes_month_out" : "2929636",
      "bandwidth_bytes_yesterday_in" : "3924623",
      "bandwidth_bytes_yesterday_out" : "625495"
   }
]
```

With arbitrary time interval.

```
# proxysmart.sh get_counters_imei  899999999999999 '2023-01-28 18:10' '2023-01-28 19:20:01'
{ "in": "1101534", "out": "2158378" }
```

On all modems:

```
# proxysmart.sh  bandwidth_report_json_all

[
   {
      "IMEI" : "352228888888888",
      "NICK" : "dongle2",
      "bandwidth_bytes_day_in" : "1202",
      "bandwidth_bytes_day_out" : "322",
      "bandwidth_bytes_lifetime_in" : "16MB",
      "bandwidth_bytes_lifetime_out" : "4.9MB",
      "bandwidth_bytes_month_in" : "10729051",
      "bandwidth_bytes_month_out" : "689922",
      "bandwidth_bytes_yesterday_in" : null,
      "bandwidth_bytes_yesterday_out" : null
   },
   {
      "IMEI" : "899999999999999",
      "NICK" : "dongle1",
      "bandwidth_bytes_day_in" : "5254",
      "bandwidth_bytes_day_out" : "3866",
      "bandwidth_bytes_lifetime_in" : "16MB",
      "bandwidth_bytes_lifetime_out" : "4.9MB",
      "bandwidth_bytes_month_in" : "19502452",
      "bandwidth_bytes_month_out" : "1376472",
      "bandwidth_bytes_yesterday_in" : null,
      "bandwidth_bytes_yesterday_out" : null
   }
]
```

8. reset bandwidth counter on a modem

```
# proxysmart.sh   bandwidth_reset_counter  dongle4

{"result":"success","debug":null}
```

9. list sms on a modem

```
# proxysmart.sh  list_sms_json  869086046197801
[
   {
      "Date" : "2021-07-08 14:05:23",
      "Content" : "Your free month has started. https://smarty.co.uk/dashboard",
      "Index" : "40001",
      "Phone" : "SMARTY"
   },
   {
      "Date" : "2021-07-12 10:23:47",
      "Content" : "621036 is your SMARTY login verification code.
      "Index" : "40002",
      "Phone" : "SMARTY"
   }
]
```

10. send sms

Plain output:

```
# proxysmart.sh  send_sms_raw 899999999999999 +11111111111 "ypa ypa 333"
= Logging in with admin:admin123
= preparing token
= Logged in
= Sending the following message to {+11111111111}: {ypa ypa 333}
= preparing token
= SENT OK
= Logging OUT
= preparing token
= RESPONSE=OK
```

JSON output:

```
# proxysmart.sh  send_sms_json  899999999999999 +11111111111 "ypa ypa 333"
{
   "debug" : "= Logging in with admin:admin123,= prepari..",
   "result" : "success"
}
```

11. purge SMS

Purges SMS from all folders.

Call by IMEI or nickname, json output:

```
# proxysmart.sh purge_sms_json 899999999999999
...
```

```
# proxysmart.sh purge_sms_json dongle1
...
```

12. send ussd

Plain output:

```
# proxysmart.sh  send_ussd_raw 899999999999999 '*100#'
= Logging in with admin:admin123
= preparing token
= Logged in
= sending USSD \*100#
= preparing token
= SENT OK
= getting response. attempt 1
= preparing token
= not yet response received
= getting response. attempt 2
= preparing token
= not yet response received
= getting response. attempt 3
= preparing token
= OK response received
Zapros nedostupen na vashem TP, naberite \*120#vyzov.
= Logging OUT
```

```
= preparing token
= RESPONSE=OK
```

JSON output:

```
# proxysmart.sh  send_ussd_json  899999999999999 '*100#'
{
   "RESPONSE" : "Zapros nedostupen na vashem TP, naberite \*120#vyzov.",
   "debug" : "= Logging in with admin:admin123,= preparing token,= ..."
   "result" : "success"
}
```

13. get bandwidth counters from a modem

..use bandwidth stats..

14. Get IP rotations log for a modem

By Nickname or IMEI

```
proxysmart.sh  get_rotation_log dongle2
proxysmart.sh  get_rotation_log 899999999999999
```

```
[
  {
    "start_time": "2022-08-10@19:29:38",
    "end_time": "2022-08-10@19:29:49",
    "total_time": "10",
    "old_ip": "4.26.28.14",
    "new_ip": "4.26.28.13",
    "target_mode": "auto"
  },
  {
    "start_time": "2022-08-10@19:29:54",
    "end_time": "2022-08-10@19:30:04",
    "total_time": "9",
    "old_ip": "4.26.248.13",
    "new_ip": "4.26.152.10",
    "target_mode": "auto"
  }
]
```

---

4. WEB API

1. Web API description.

`localhost:8080` is the URL that Proxysmart Web-App sits on. Basically it is the URL that you can browse Proxysmart WebApp with.

It can be also `$LanIP:8080` or when you forward HTTP port 8080 to you static Home IP, or to the cloud VPS, it will became as well `YourStaticIP:8080` or `VPS_IP:8080` . Use any endpoint that suits best!

Also attach proper username:password (the -u parameter).

2. List all modems (full status, slow)

Request:

```
curl 'http://localhost:8080/modems' -u proxy:proxy
```

Response:

```
{
  "message": null,
  "modems": [
    {
      "cellMode": "LTE",
      "cellOp": "A1 BY",
      "dev": "modem115",
      "extIp": "46.56.228.215 ",
      "imei": "899999999999999",
      "model": "E3372h-320",
      "modemIp": "192.168.8.1",
      "nick": "random7",
      "proxyCreds": [
        "http: 28007:def:def",
        "socks: 25007:def:def"
```

```
      ],
      "redirectorStatus": "redirectors disabled globally",
      "simStatus": "(1) valid SIM card",
      "statusMessage": "",
      "valdik": "Detected OS = Linux 2.2.x-3.x [generic];MTU = 1400;
            Network link = Probably IPsec or other VPN;
            PTR test = Probably home user;Fingerprint and OS match. No proxy detected ;
            No OpenVPN detected."
    }
  ],
  "success": true
}
```


3. List all modems (brief status, fast)

Request:

```
curl localhost:8080/apix/show_status_brief_json -u proxy:proxy
```

Response:


```
[
  {
    "MSG": "",
    "N": "172",
    "IS_LOCKED": "false",
    "modem_details": {
      "NICK": "dongle2",
      "IMEI": "352228888888888"
    },
    "net_details": {
      "DEV": "modem172",
      "GW": "192.168.8.1",
      "LOCAL_IP": "192.168.8.100",
      "EXT_IP": "46.216.112.104",
      "IS_ONLINE": "yes"
    },
    "proxy_creds": {
      "HTTP_PORT": "8003",
      "SOCKS_PORT": "5003",
      "LOGIN": "alice",
      "PASS": "cool",
      "PROXYSTDLINE_LAN": "192.168.100.6:8003:alice:cool",
      "PROXYSTDLINE_WWW": "forwarding_disabled"
    },
    "redirector_status": {
      "MSG": "redirectors disabled globally"
    }
  }
]
```


4. Single modem status

Request:

(either by IMEI or Nickname)

```
curl http://localhost:8080/apix/show_single_status_json?arg=dongle111    -u proxy:proxy
curl http://localhost:8080/apix/show_single_status_json?arg=899999999999999    -u proxy:proxy
```

Response:


```
[
  {
     "IS_LOCKED" : "false",
     "MSG" : "",
     "N" : "115",
     "modem_details" : {
        "HUB_ID" : "1-1",
        "HUB_PORT" : "3",
        "IMEI" : "899999999999999",
        "MODEL" : "E3372h-320",
        "NICK" : "dongle111",
        "UDEV_UPTIME" : "1212291",
        "UPTIME" : "14 days + 45.866667 minutes"
     },
     "net_details" : {
        "CELLOP" : "MTS BY",
        "ConnectionStatus" : "901, DATA:connected OK",
        "CurrentNetworkType" : "(19) LTE",
        "DEV" : "modem115",
        "EXT_IP" : "46.216.224.164",
        "GW" : "192.168.8.1",
        "IS_ONLINE" : "yes",
        "LOCAL_IP" : "192.168.8.100",
        "SIGNAL_STRENGTH" : "4",
```

```
         "SimStatus" : "(1) valid SIM card",
         "VALDIK" : "Detected OS = Linux 2.2.x-3.x [generic];MTU = 1420;
                Network link = generic tunnel or VPN;PTR test = Probably home user;
                Fingerprint and OS match. No proxy detected ;No OpenVPN detected.",
         "workmode" : "LTE"
      },
      "proxy_creds" : {
         "HTTP_PORT" : "8004",
         "LOGIN" : "alice",
         "PASS" : "cool",
         "PROXYSTDLINE_LAN" : "192.168.100.2:8004:alice:cool",
         "PROXYSTDLINE_WWW" : "forwarding_disabled",
         "SOCKS_PORT" : "5004"
      },
      "redirector_status" : {
         "MSG" : "redirectors disabled globally"
      }
   }
]
```


5. Reset (change) IP on a modem.

Request:

(either by IMEI or Nickname)

```
curl http://localhost:8080/apix/reset_modem_by_imei?IMEI=899999999999999 -u proxy:proxy
curl http://localhost:8080/apix/reset_modem_by_nick?NICK=dongle22 -u proxy:proxy
```

Response:


```
{
   "debug" : "...",
   "ext_ip" : "46.216.248.48",
   "message" : "external ip changed from 46.216.225.112 to 46.216.248.48",
   "result" : "success"
}
```


6. Reboot a modem

Request:

(either by IMEI or Nickname)

```
curl http://localhost:8080/apix/reboot_modem_by_imei -d IMEI=860493043888886 -u proxy:proxy
curl http://localhost:8080/apix/reboot_modem_by_nick -d NICK=dongle2 -u proxy:proxy
```

Response:


```
{
   "debug" : "...",
   "message" : "new external ip cannot be detected",
   "result" : "failure"
}
```


or


```
{
   "debug" : "...",
   "ext_ip" : "172.58.172.255",
   "message" : "external ip changed from 172.58.172.251 to 172.58.172.255",
   "result" : "success"
}
```


ETA: ~ 1.5 minute

7. Send SMS

Request:

```
curl 'http://localhost:8080/modem/send-sms' -u proxy:proxy \
    --data-urlencode 'imei=899999999999999' \
    --data-urlencode 'phone=+11111111111' \
    --data-urlencode "sms=txt txt fff"
```

Response:


```
{"message":"Result: success","success":true}
```

8. Send USSD and read response

Request:

```
curl 'http://localhost:8080/modem/send-ussd' -u proxy:proxy \
    --data-urlencode 'imei=899999999999999' --data-urlencode 'ussd=*100#'
```

Response:

```
{
"RESPONSE":"Zapros nedostupen na vashem TP, naberite \*120# vyzov.",
"debug":"...",
"result":"success",
"success":true
}
```

9. Read SMS from a modem

Request:

```
curl 'http://localhost:8080/modem/sms/8623298888888888?json=1' -u proxy:proxy
```

Response:


```
{
   "data" : [
      {
         "Content" : "Missed call : +333333333370 at 10:45 22/07.",
         "Date" : "2020-07-22 14:59:35",
         "Index" : "40001",
         "Phone" : "+333333333370"
      },
      {
         "Content" : "Welcome, your data limit 0-100МБ. .. Details: cell.org",
         "Date" : "2021-02-27 00:53:11",
         "Index" : "40002",
         "Phone" : "MYCELL"
      },
      {
         "Content" : "Hh",
         "Date" : "2021-07-16 20:32:11",
         "Index" : "40042",
         "Phone" : "+11111111111"
      }
   ],
   "success" : true
}
```


10. Read bandwidth stats from a modem

Request:

```
curl localhost:8080/apix/bandwidth_report_json?IMEI=899999999999999   -u proxy:proxy
```

Response:


```
[
   {
      "IMEI" : "899999999999999",
      "NICK" : "dongle111",
      "bandwidth_bytes_day_in" : "2945",
      "bandwidth_bytes_day_out" : "2314",
      "bandwidth_bytes_month_in" : "62859",
      "bandwidth_bytes_month_out" : "49559",
      "bandwidth_bytes_yesterday_in" : "5048",
      "bandwidth_bytes_yesterday_out" : "3984"
   }
]
```


With arbitrary time interval.

Request:

```
curl -G http://localhost:8080/apix/get_counters_imei -X GET -d IMEI=868888888888888 --data-urlencode 'START=2023-01-28 18:10' --dat
```

Response:


```
{ "in": "1101534", "out": "2158378" }
```

11. Read bandwidth stats from all modems

Request:

```
curl localhost:8080/apix/bandwidth_report_json_all -u proxy:proxy
```

Response:

```
[
   {
      "IMEI" : "899999999999999",
      "NICK" : "dongle111",
      "bandwidth_bytes_day_in" : "2945",
      "bandwidth_bytes_day_out" : "2314",
      "bandwidth_bytes_month_in" : "62859",
      "bandwidth_bytes_month_out" : "49559",
      "bandwidth_bytes_yesterday_in" : "5048",
      "bandwidth_bytes_yesterday_out" : "3984"
   },
   {
      "IMEI" : "862329041089999",
      "NICK" : "dongle111",
      "bandwidth_bytes_day_in" : "1295",
      "bandwidth_bytes_day_out" : "1234",
      "bandwidth_bytes_month_in" : "16259",
      "bandwidth_bytes_month_out" : "49259",
      "bandwidth_bytes_yesterday_in" : "5018",
      "bandwidth_bytes_yesterday_out" : "3294"
   }
]
```

12. Reset bandwidth stats for a modem

Request (by IMEI or nickname):

```
curl localhost:8080/apix/bandwidth_reset_counter?arg=dongle111    -u proxy:proxy
curl localhost:8080/apix/bandwidth_reset_counter?arg=2727233671671676    -u proxy:proxy
```

Response:

```
{"result":"success","debug":null}
```

13. Reset a modem via USB

Request either

- by network interface e.g. modem77
- by Nickname
- by IMEI

```
curl localhost:8080/apix/usb_reset_modem_json?arg=modem77      -u proxy:proxy
curl localhost:8080/apix/usb_reset_modem_json?arg=dongle22      -u proxy:proxy
curl localhost:8080/apix/usb_reset_modem_json?arg=868888888888889      -u proxy:proxy
```

Response:

```
{
  "USB_RESET_METHOD": "uhubctl",
  "debug": "......",
  "result": "ok"
}
```

14. Get IP rotations log for a modem

Request

- by Nickname
- by IMEI

```
curl localhost:8080/apix/get_rotation_log?arg=899999999999999  -u proxy:proxy
curl localhost:8080/apix/get_rotation_log?arg=dongle2          -u proxy:proxy
```

Response:

```
[
  {
    "start_time": "2022-08-10@19:29:38",
    "end_time": "2022-08-10@19:29:49",
    "total_time": "10",
    "old_ip": "4.26.28.14",
```

```
    "new_ip": "4.26.28.13",
    "target_mode": "auto"
  },
  {
    "start_time": "2022-08-10@19:29:54",
    "end_time": "2022-08-10@19:30:04",
    "total_time": "9",
    "old_ip": "4.26.248.13",
    "new_ip": "4.26.152.10",
    "target_mode": "auto"
  }
]
```

15. Apply settings for a modem

Request:

```
curl http://localhost:8080/modem/settings -d imei=862329099999999 -u proxy:proxy
```

Response:

```
{
  "message": "Result: success, message: applied",
  "success": true
}
```

5. Mongodb integration

Instead of defining modems details in `map.txt` , you can use MongoDB.

It is installed by default.

Mongodb contains a collection `modems` with elements, 1 element = 1 modem.

Mandatory fields are

- IMEI
- name
- http_port
- socks_port
- proxy_login
- proxy_password

Other fields are optional.

Sample file **modems.json** with 2 modems. 1st modem: only mandatory fields. 2nd modem: also arbitrary fields.

```
{
  "IMEI": "868888888888888",
  "name": "dongle5",
  "http_port": "8005",
  "socks_port": "5005",
  "proxy_login": "kileq",
  "proxy_password": "Jdh27dh"
}
{
    "IMEI": "869777777777777",
    "name": "dongle4",
    "http_port": "8004",
    "socks_port": "5004",
    "proxy_login": "mokos",
    "proxy_password": "rQ1h6J",
    "white_list": [
        "78.140.162.201",
        "78.140.162.202"
        ],
    "bandlimin":1000000,
    "bandlimout":1000000,
    "DENIED_SITES_ENABLE": 1,
    "DENIED_SITES_LIST": [
        "bad.com",
        "porn.com"
        ],
    "bw_quota": 2000,
    "mtu":  1400,
    "extra_users": [
            {
              "BANDLIMIN": "100000",
              "BANDLIMOUT": "100000",
```

```
            "login": "aaaaa",
            "password": "aaaaa"
          }
          {
            "BANDLIMIN": "100000",
            "BANDLIMOUT": "100000",
            "login": "bbbbbbbbb",
            "password": "bbbbbbb"
          }
    ],
    "PROXY_VALID_BEFORE":"2028-02-22T12:54",
    "AUTO_IP_ROTATION": 0
}
```

Install Mongodb and database

```
apt install mongodb mongo-tools
mongo
> use proxysmart
> db.createUser( { user: "proxysmart", pwd: "JQdMJe7Rkw", roles: [ { role: "readWrite", db: "proxysmart" } ] })
> exit
```

Then import the collection to the DB

```
mongoimport  --uri=mongodb://proxysmart:JQdMJe7Rkw@localhost:27017/proxysmart -c modems  < modems.json   --drop
```

Update mongodb uri in `/etc/proxysmart/conf.txt` :

```
MONGODB_URI="mongodb://proxysmart:JQdMJe7Rkw@localhost:27017/proxysmart?readPreference=primary&ssl=false"
```

Set `DB_BACKEND=mongo` there

Regenerate all config files:

```
proxysmart.sh reset_complete
```

So it will detect modems and look up for values from MongoDB.

6. Installation

1. Install DEB package

Install a fresh OS.

Supported OS and architectures:

- Ubuntu 22.04, 20.04 on amd64, arm64.
- Debian 11 or Raspberry PI OS (ex-Raspbian) on amd64, arm64, armhf.

- Raspberry PI : https://ubuntu.com/download/raspberry-pi , choose Ubuntu Server 64bit
- Normal PC/laptop: Choose Server or Desktop, https://ubuntu.com/download

Armhf (arm 32 bit) doesn't have Mongodb support!

Those steps will take 5..10 minutes.

Unplug any 4g modems.

Get a DEB package from Developer. Or download from an APT repo.

```
wget -O-  https://pathos.tanatos.org/proxysmart.apt.repo/GPG.txt | \
    gpg --dearmor | sudo dd of=/etc/apt/trusted.gpg.d/proxysmart.gpg

source /etc/os-release
ARCH=$(dpkg --print-architecture)

echo "deb [arch=$ARCH] http://pathos.tanatos.org/proxysmart.apt.repo $VERSION_CODENAME main" \
    | sudo tee /etc/apt/sources.list.d/proxysmart.list

sudo apt update
sudo apt install proxysmart
```

Then follow instructions: It will tell what to do next (run 2 files).

```
sudo /usr/lib/proxysmart/install_pkgs.sh
sudo /usr/lib/proxysmart/install_webapp.sh
```

After that either enjoy the Demo version or check License section.

New WebApp activation

(only for those who installed the software in 2021 or earlier)

It allows editing modems details right in browser.

It is activated by default, so no need to do these setps.

In order to activate: run `sudo /usr/lib/proxysmart/install_webapp.sh` and it will print values for `DB_BACKEND` and `MONGODB_URI` , update `/etc/proxysmart/conf.txt` with them and restart prosymart , run `systemctl restart proxysmart` .

2. Building DEB package from source code

only makes sense when you have got the source code

In a folder of source code run

```
sudo ./build_deb.sh
```

after building, find a file `../proxysmart*deb` and `sudo dpkg -i ../proxysmart*deb`

Then follow instructions that were printed out on the console (run 2 files).

3. Post Installation

Plug in all 4g modems you have, wait ~20 sec to let them initialize.

Now test if `ip li` shows you any `modem*` interfaces, otherwise reboot to apply UDEV rules.

If it does, continue next below. (Otherwise reboot to apply UDEV rules.)

Now you can start all the modems:

You have to run `proxysmart.sh reset_complete` or reboot the multi-modem server.

Command `proxysmart.sh show_status` will return a table with proxy port, external IP's.

Navigate to the WebApp http://localhost:8080 proxy/proxy and assign login/password/nicknames/ports to the modems.

Test reboot, reboot the box, wait 1 minute, make sure the WebApp shows the modems.

WebApp

Visit http://your_box_lan_IP_address:8080/ or http://localhost:8080/

Default user:password pair is proxy:proxy

4. Cloud VPS

The VPS is needed to forward proxy ports from a cloud VPS IP back to the multi modem server, so proxy ports are available for all users around the world.

Do I need a VPS?

A VPS is NOT needed when all the conditions are met:

- you have static IP at 4g proxy farm location, i.e. ISP provides it, and
- ISP allows incoming connections to that static IP
- Upload and Download of "ground" Internet is at least 20 Mbps.

Without a VPS, you can forward proxy ports on your Home/Office router to multi-modem server in the LAN. In that case users from around the world will connect to your static IP, so these connections are forwarded to the 4g farm server situated in the LAN.

The VPS server can be a cheap 1GB DigitalOcean / Linode / Vultr VPS or similar.

It has to be located as close as possible to the 4g farm server (for lowest ping).

VPS setup steps.

On multi modem server

Copy content from the file `/root/.ssh/fwd.pub` [1]

On VPS

Check if your VPS has no firewall. Disable it if it has -- Both inside Linux OS and in hoster panel.

Create a user `fwd` :

```
useradd -s /bin/true -m fwd
mkdir -p /home/fwd/.ssh/
touch /home/fwd/.ssh/authorized_keys
chown -R fwd: /home/fwd/
chmod 700 /home/fwd/.ssh/
chmod 600 /home/fwd/.ssh/authorized_keys
```

edit the file and paste the content [1] you copied in the step above.

```
nano /home/fwd/.ssh/authorized_keys
```

Make sure there are these lines in `/etc/ssh/sshd_config` or in an extra conf file:

```
mkdir -p /etc/ssh/sshd_config.d
echo '
GatewayPorts clientspecified
ClientAliveInterval 3
ClientAliveCountMax 3
MaxStartups 100:30:1000
LoginGraceTime 10
' > /etc/ssh/sshd_config.d/proxysmart.conf

service ssh restart
```

On multi modem server

in `/etc/proxysmart/conf.txt` :

- set `VPS` variable to VPS IP
- set `PROXY_PORTS_FORWARDER_ENABLE=1`
- run `proxysmart.sh reset_complete`
- edit `/etc/systemd/system/fwdssh-vps.service` , change `CONNECT_HOST` to VPS IP
- Pick a free port for `SSH_REMOTE_PORT`, in most cases 6902 is fine.

Run:

```
systemctl daemon-reload
systemctl start fwdssh-vps
systemctl enable fwdssh-vps
systemctl status fwdssh-vps
```

Make sure it is green.

On VPS

issue the command `ss -tnlp` and you will see proxy ports are bound with `sshd` daemon. That means the ports are forwarded.

On your private desktop

- visit `http://vps_ip:8080` for the WebApp , default login:password is proxy:proxy
- you can ssh to VPS IP and port 6902, and that goes to the multi-modem-server:22.

Cloud VPS IP change

If CLoud VPS IP is changed, update it on multi-modem-server side by defining new `VPS` variable in the `/etc/proxysmart/conf.txt` file, and rerun `proxysmart.sh reset_complete` there.

Also change VPS IP in `/etc/systemd/system/fwdssh-vps.service` on multi-modem-server and run these:

```
systemctl daemon-reload
systemctl restart fwdssh-vps
systemctl status fwdssh-vps
```

Make sure it is green.

7. License

1. Demo license

Installation is shipped with default **demo** license.

It allows you to run proxy on 1 modem.

In order to run more modems, ask the developer for an extra license, send him the MachineData field from `proxysmart.sh license_status` output and he will issue new license and you will install it.

2. New license installation

You will be given the **license** and **license signature**. Both are sequences of numbers and characters. Then submit both either via WebApp or CLI:

submitting via CLI

run commands

```
proxysmart.sh submit_license LICENSE
proxysmart.sh submit_license_signature LICENSE_SIGNATURE
```

submitting via WebApp

Open the WebApp , http://localhost:8080 , expand License section and type in the keys & submit.

3. Restoring Demo license.

If your paid license expired or broken, restore DEMO license, run:

```
sudo cp -v /usr/share/doc/proxysmart/examples/license.txt* /etc/proxysmart/
```

8. Residential VPN [BETA]

Together with building proxies, it is possible to build Residential VPN.

Assumption is, your proxies are already available via Cloud VPS. So pick a free TCP port on Cloud VPS e.g. 1594

On multi modem server, edit `/etc/proxysmart/conf.txt` and set `OPENVPN_SERVER_HOST=3.3.3.3` i.e. to the VPS IP ; and `OPENVPN_SERVER_PORT=1594` , to the free TCP port on Cloud VPS.

These 2 above means that VPN client certificates will be generated with this value, so VPN clients will connect there.

Set `OPENVPN_INTEGRATION=1` so that Proxysmart will understand Openvpn is in use.

Edit `/etc/systemd/system/fwdssh-vps.service` , set `CONNECT_HOST` to VPS IP; Uncomment & set `OPENVPN_LOCAL_PORT` to 1194 , `OPENVPN_REMOTE_PORT` to the same port as `OPENVPN_SERVER_PORT` in `/etc/proxysmart/conf.txt` above. Run

```
systemctl daemon-reload
systemct restart fwdssh-vps
systemct enable fwdssh-vps
```

This just enabled port forwarding VPS:OPENVPN_REMOTE_PORT to localhost:OPENVPN_LOCAL_PORT.

Then run `/usr/lib/proxysmart/install_openvpn.sh` , it will do the installation of Openvpn server. If it says "Openvpn integration already ready.." then you can remove the file `/etc/openvpn/.proxysmart.conf.completed` and rerun it.

Then finally reconfigure the system by running `proxysmart.sh reset_complete` .

Mongodb backend (default) : For each modem it will generate a VPN profile.

Map backend : Assign mapping between VPN clients and dongles, by editing `/etc/openvpn/map.txt` , the format is `VPN_USER:IMEI:VPN_LOGIN:VPN_PASSWORD` , And generate each vpn profile with a command `openvpn_create_user MyVpnuser111`

You can download them later as from http://localhost:8080/vpn_profiles/$NICK.ovpn or grab from `/home/vpn/` folder.

So download VPN profiles and connect using any VPN client software.

Windows: https://openvpn.net/community-downloads/ or https://openvpn.net/client-connect-vpn-for-windows/

MacOS: https://tunnelblick.net/

Android: https://play.google.com/store/apps/details?id=de.blinkt.openvpn or https://f-droid.org/en/packages/de.blinkt.openvpn/

IOS: https://apps.apple.com/us/app/openvpn-connect/id590379981