1 Introduction

1.1 About Fon

Fon is the Wi-Fi roaming operator, based on the idea of sharing your broadband conveniently as a public Wi-Fi hotspot. The reward for sharing is the right to use Wi-Fi wherever Fon hotspots exist.

1.2 What Fon requires from a CPE?

For end users, Fon's service appears as Wi-Fi signals where they can be associated to. After an user is associated, authentication must be performed in order to gain access to the Internet. Only after authentication the users can connect to the Internet without restriction. In the implementation of the CPE, Fon requires features that make all this possible in a secure way. Fon requires the addition of actors into the CPE, for management of the signals and service status, configuration, authentication and access control. Moreover some modifications in the existing software are required. Incorporation of Fon Service/software should not require any hardware changes.

1.3 Incorporating Fon to CPE

In order to provide access to the Fon network through any device, it must provide all the functionality and interfaces defined by the doc CPE Technical Specifications. This can be done integrating the FonAP into the CPE.

2 FonAP functional overview

FonAP is the software **reference code** to implement a solution to incorporate Fon's service into the CPE. This solution is optimized and based on OpenWRT (OWRT) platform (https://openwrt.org/).

- A hotspot manager (*hotspotd module*). It controls user and client device access. It is the responsible to guarantee that only authenticated users are able to access the Fon network. Technically, it is a piece of logic that is able to configure CPE's firewall, routing tables, bridging tables and NAT procedures to treat Fon's users accordingly to their authentication and authorization status.
- A configure actor (*freecwmp2 module*). It is a conceptual actor that retrieves, keeps and provides the configuration parameters for other actors.

The hotspot manager functionalities are controlled by *hotspotd* daemon. The configure actor functionalities are controlled by *freecwmp2* daemon.

2.1 Hotspotd module

Hotspotd is designed as a daemon based on the Linux operating system, to provide a reliable light-weight solution to add Fon hotspot functionality to embedded devices as well as access concentrators. As such user traffic is never altered by or passed directly through hotspotd but rather remains in kernel space to ensure a high performance especially on embedded hardware. The program therefore configures the kernel directly through its user-space APIs.

Hotspotd uses a single-process, single-threaded event-multiplexer to schedule its tasks. This ensures a low memory overhead while retaining reasonable response times. Additionally this approach has proven to be easy to test and maintain as error-prone methods such as IPC, process synchronization or mutexes are avoided. This architecture is suitable for applications performing many different smaller tasks while being idle waiting for events most of the time.

Hotspotd uses a cooperative event-based scheduler to choose active tasks. This means that tasks are triggered by events and get full control over the program flow while active and will yield control back to the scheduler on their own when finished.

The hotspotd is componed by different modules. Modules are logical units of code serving a specific purpose and usually consist of a single C source code file and sometimes additional headers and source code files providing auxillary functions. There are core modules that implements the basic Fon's service and several optional modules called extensions.

2.1.1 Core modules

- Captive module. Captive Portal and HTTP-API. The captive module is responsible for intercepting HTTP traffic to redirect the user to the splash page as well as for communicating with the uamserver via HTTP-redirects through the user's browser.
- Client module. Client Management and AAA. The client module is responsible for managing client sessions. It maintains a list of all active sessions and provides an API for other modules to request session changes.
- Firewall module. Firewall Management. The firewall module is reponsible for managing the firewall rules for the hotspot interfaces and provides an API for other modules to change its state, e.g. add / remove client sessions, open services, whitelisted domains, etc. The module calls the iptables binary of the operating system.
- Routing module. Routing Control. The routing module provides the primary communication link to the kernel network subsystem.
- Traffic module. Traffic Control and QoS. The traffic module manages the traffic control classes and rules for the assigned ifb / imq interfaces based on the HFSC scheduler. It is also responsible for measuring user traffic.
- Whitelist module. Dynamic Walled Garden. The whitelist modules is responsible for intercepting DNS traffic and maintaining the walled garden.
- Loader module. Hotspot Loader. The loader module is responsible for monitoring the status of the system's network interfaces. It subscribes to the kernel netlink group for network link events. Whenever a network interface changes its state the module creates an internal signal and sends the information to subscribed modules.
- RPC module. RPC interface. he RPC Interface provides means to interact with a running hotspotd instance from third-party tools.
- Trigger module. Trigger Control. The trigger module provides an API for other modules to launch and watch external executables. The module uses the path for a trigger from the configuration and applies custom parameters and adjusts the environment.

2.1.2 Extension modules

- Radius extension. RADIUS client backend. The RADIUS Client Backend implements a RADIUS client to communicate with RADIUS AAA servers for authentication and accounting.
- Tunnel extension. Tunnel management. The tunnel module is responsible for managing PPP based tunnel connections.
- EAP radius extension. EAP RADIUS Server. The EAP RADIUS server provides a minimalistic RADIUS server implementation to interact with WPA authentication daemons such as hostapd to add clients authenticated via WPA-Enterprise into the regular authentication flow.
- Private extension. Private signal accounting. The private module is responsible for measuring user traffic in ther private signal.
- Service extension. The service module mantains a list of avaliable services and grantees and keep a track of grantees' services.
- UBUS extension. (Only for OWRT platform). The ubus module provides an interface for UBUS communication.

2.2 Freecwmp2 module

Freecwmp2 is a daemon which is responsible of retrive Fon's service configuration. It provides a fully functional CWMP (TR-069) client implementation. The daemon provides a SOAP/HTTP client to connect with the ACS. Secure transport and authentication of the ACS is provided by usage of HTTPS. A XML parser and a backend is provided within the freecwmp2 module to setting and retrieving the device values. (Backend documentation is provided in the freecwmp2 **reference code**, *BACKEND.html* file).

3 Administration

3.1 Users

All the daemons must run as root user.

3.2 Daemon control

- Hotspotd
  Hotspotd binary is placed in */sbin* folder and it can be invoke calling *sbin/hotspod* with the following options.

```
Usage: hotspotd [options]
Options:
        -c <name>                       Use configuration file /etc/config/<name>
        -o section.key=val              Override config value from config file
        -e                              Output log messages on stderr
        -v                              Be more verbose (might be used multiple times)
        -d                              Daemonize
        -p <pidfile>                    Write a pidfile
        -h                              Show this help
```

- Freecwmp2
  Freecwmp2 binary is placed in */usr/sbin/* folder and it can be invoke calling */usr/sbin/freecwmp2* with the following options.

```
Usage: freecwmp2 [options]
        Backend:
                -b <backend>    CWMP-Backend
                -i <interface>  Listening interface
        Certificate Information:
                -s <cert>       Path to server certificate
        Invocation options:
                -p <pidfile>    Set pidfile (/var/run/freecwmp2.pid)
                -d              Daemonize
                -v              Increase logging verbosity
                -h              Show this help
```

3.3 Logging

The daemon's logs can be viewed in the syslog facility. If an OpenWRT based platform is used the command *logread* shows these logs. Each daemon is identified with its name/version.

```
fonera daemon.warn hotspotd/1.3.3: starting up...
fonera daemon.debug freecwmp2/1.0.1[1645]: Initializing freecwmp2/1.0.1 daemon
```

3.4 Configuration

Due to FonAP is OWRT based, hotspotd and freecwmp2 uses OpenWrt's UCI as a configuration backend. The default config file is */etc/config/hotspotd* for hotspotd and */etc/config/freecwmp2* for freecwmp2. The */etc/config* path must exists.

3.5 Signaling

- hotspotd listens to the following signals at runtime.

Signal	Effect
SIGHUP	Restart hotspotd and reapply configuration
SIGTERM SIGINT	Controlled shutdown
SIGUSR1	Reload hotspotd an apply configuration (without interrupting service)

- freecwmp2 listens to the following signals at runtime.

Signal	Effect
SIGTERM SIGINT SIGHUP	Controlled shutdown
SIGALARM	Sends an inform to the ACS

3.6 Other executables

These daemons use a collection of shell scripts/binaries that interact with the system to complete their actions. All these scripts are placed in */sbin* path.

3.6.1 Hotspotd

- hotspot-disconnect: Disassociates the client from a Fon signal. (*/sbin/hotspot-disconnect*).
- hotspot-down: Downs all Fon Wi-Fi signals (public signal and EAP signal if configured). (*/sbin/hotspot-down*).
- hotspot-up: Ups all Fon Wi-Fi signals (public signal and EAP signal if configured). (*/sbin/hotspot-up*).

3.6.2 Freecwmp2

- A reboot script/binary must exists.
- cwmp-commit Performs needed actions when the ACS (autoconfiguration service) session ends successfully.
- cwmp-abort Performs needed actions when the ACS session ends unsuccessfully.
- cwmp-apply Reloads different module to apply configuration changes.
  - *cwmp-apply <module>*. Where module can be hotspotd, network or wireless.

4 Building FonAP

FonAP **reference code** can be build to test some functionalities or to check how to compile a solution based in this **reference code**. Building FonAP compiles the hotspotd and freecwmp2 modules separately.

4.1 Hotspotd module

You can invoke make in the root directory of hotspotd with any of the following targets. You can check Makefile file in hotspotd folder.

Target	Result	Requirements
(default)	Builds binaries of hotspotd, fonctl and hotspotd loadable modules. This is the standard target that may be invoked locally when no target is given or by any cross-compiling toolchain like OpenWrt or the Fonera firmware toolchain.	C99 compiler (e.g. recent GCC or Clang / LLVM) eglibc, uClibc or compatible C standard library
deb	Also builds debian packages from hotspotd binaries.	debhelper (>= 8.0.0)
debimage	Also build a debian flash image hotspotd.loop to be copied to a hard drive or USB stick or to be used as a kvm / qemu disk image.	debootstrap, parted
debvm	Also builds a virtual applicance hotspotd.ova to be imported and used with a virtualmachine like VirtualBox (or possibly VMWare or VirtualPC)	virtualbox (>= 4.0)

4.1.1 Tuning hotspotd extensions

Hotspotd contains several optional modules called extensions (e.g. radius, tunnel, eapradserv, ...) that you might want to built into the hotspotd executable. Therefore you may adjust the MODULES parameter when invoking make. This parameter is a separeted list of extensions to build in.

```
MODULES_BUILTIN=tunnel radius fonctl fonapi eapradserv localusers rpc ubus private service
MODULES:=$(MODULES_BUILTIN)
```

- NOTE: The ubus module is implemented specifically to build in OWRT platform that supports ubus. Don't compile it if your OWRT platform doesn't have support for ubus.

4.1.2 Tuning external libraries

You may choose to change linking against several libraries to reduce code size incase you already have a working copy of that library in your firmware. However if you are unsure just use the default settings which require no external libraries (except the standard C library and its components) to be provided by you.

Parameter	Values
librt	(builtin) Do not link against *librt* (requires clock_gettime to be part of libc like in uClibc) or (shared) link against librt
libuci	(builtin) Build and bundle own copy of the library or (shared) link against external copy of the library
libunl	(builtin) Build and bundle own copy of the library or (shared) link against external copy of the library

- NOTE: If the private module is built. If iptables version is minor than 1.4.3 the *libiptc* is needed, otherwise *lip4tc* is needed.
- NOTE: If the ubus module is built (see Note above) the following libraries are needed *libubus*, *libubox*, *libssl* and *libcryto*.

4.2 Freecwmp2 module

Uses cmake to create the binary. You can can check CMakeList.txt file in freecwmp2 folder.

4.2.1 How to compile it

This is an example of how to use cmake.

```
mkdir build
cd build
cmake ..
make
```

4.2.2 Tuning compilation

Different compilation flags must be set (HTTPS_ENGINE, BACKEND, LIBJSON, CRLSYNC).

- HTTPS_ENGINE selects the ssl library that link to.

HTTPS_ENGINE values	HTTP client	Libraries
openssl	http-builtin	*libssl*, *libcrypto*
cyassl	http-builtin	*libcyassl*
polarssl	http-builtin	*libpolarssl*
default	http-curl	*libcurl*

- The CWMP-client requires a backend to execute CWMP commands in the local environment. Fon provides a backend placed in *openwrt* folder into freecwmp2 module. BACKEND flag is set to "openwrt" (BACKEND=*openwrt*) to point to this *openwrt* folder. If CPE implements a new backend in the freecwmp2 module, it must point the BACKEND flag to this new folder implementation. Provided *openwrt* backend links against *libjson* and *libuci*, so the CPE must provide these library.
- CRLSYNC checks certificate against a CRL. A binary downloads a certificate revocation list (CRL) to check the CPE certificate against it. If CRLSYNC flag is set to 1 a *crlsync* binary is compiled and created to . This binary download a certificate revocation list (CRL) to check the CPE certificate against it. It must be placed in */usr/sbin/* path.

4.2.3 Tuning external libraries

You may choose to change linking against *libroxml* library to reduce code size incase you already have a working copy of that library in your firmware. The backend need *libuci* to link against.

5 Hotspotd runtime dependencies

To integrate FonAP there are some runtime dependencies that might be satisfied. The hotspotd binary use them to work as expected.

- User-space
  - *iptables* >= 1.4.0. To control the Linux firewall.
  - *ip6tables*. If IPv6 is required.
  - *pppd* >= 2.4.4. For tunnel support.
    - plugin *passwordfd.so*
    - plugin *pppol2tp.so* (for L2TP tunnel backend).
- Kernel
  - *Linux* >= 2.6.23 (l2tp kernel module was introduced in 2.6.23) (>= 2.6.37 for IPv6 support)
  - eventpoll support (CONFIG_EPOLL)
  - Intermediate Functional Block support network device driver (or alternatively IMQ support)
  - QoS and/or fair queueing
    - Hierarchical Fair Service Curve (HFSC)
    - Ingress Qdisc (not necessary if IMQ is used)
    - Elementary classification (BASIC)
    - Netfilter mark (FW)
    - Redirecting and Mirroring (not necessary if IMQ is used)
    - Connection Tracking Marking (not necessary if IMQ is used, not upstream, shipped in "kmod-act_connmark" directory)

- Core Netfilter Configuration
  - Netfilter netlink interface (CONFIG_NETFILTER_NETLINK)
  - Netfilter LOG over NFNETLINK interface (CONFIG_NETFILTER_NETLINK_LOG)
  - Netfilter NFQUEUE over NFNETLINK interface
  - Connection tracking netlink interface (CONFIG_NF_CT_NETLINK)
  - Connection tracking flow accounting (CONFIG_NF_CT_ACCT)
  - "CONNMARK" target support
  - "MARK" target support
  - "NFLOG" target support
  - "NFQUEUE" target support
  - "TPROXY" target support (alternatively "REDIRECT" for IPv4, see below)
  - "connbytes" per-connection counter match support
  - "length" match support
  - "mac" address match support
  - "mark" match support
  - "state" match support
- IP Netfilter Configuration
  - "recent" match support
  - "REJECT" target support (recommended)
  - "REDIRECT" target support (use "TPROXY" instead if possible, see above)
- l2tp_ppp networking module or pppol2tp network device driver (for l2tp tunnel backend only)

6 Configuration files

The configuration file needed by the startup script contains some basic information needed to start the Fon service. Without the correct configuration the service will never starts.

For hotspotd module there is a configuration file example into *conf* folder named *hotspotd*. Freecwmp2 config file example is placed into the *openwrt* folder that is placed into the *src* directory. This file is named *freecwmp2.config*.

Note: Use the configuration parameters given by FON (cpeteam@fon.com)

6.1 Hotspotd config file

You can see a briefly explanation of the parameters into the configuration file. Some examples are:

- option *iface wopen*. It is the name of the public interface to identified it into the network config.
- option *iface_eap weap*. It is the name of the EAP interface to identified it into the network config.
- option *nasid c4-71-30-3e-84-b4*. it is the way a CPE will identify itself to the Fon servers.

6.2 Freecwmp2 config file

A briefly explanation of the freecwmp2 config file. There are several groups in the freecwmp2 config:

- DeviceInfo. It is info to set in all inform request sent to the ACS.
- ManagementServer. It is the configuration of the proper service.
  - URL: It is the complete URL of the ACS.

  - CWMPRetryMinimumWaitInterval: Value in seconds to retry a failed communication with the ACS.
  - CWMPRetryIntervalMultiplier: Value to modify the previous value.
  - PeriodicInformEnable: If true a periodic inform is sent to the ACS.
  - PeriodicInformInterval: Interval in seconds to send to the ACS the periodic inform.
  - ParameterKey: This value is used in the ACS to check if the CPE configuration is updated.
- Trigger. Sets the path of the different executables that freecwmp2 needs. (See Administration chapter).
- The rest of params indicates which configuration value is set when a new confugation is retrieved.

7 OpenWRT platform specifics

7.1 Other OpenWRT configuration

In OWRT platform the network is configured through configure files placed in */etc/config* folder. These are examples of network configuration to obtain a full Fon's service.

7.1.1 /etc/config/network

```
config interface 'loopback'
        option ifname 'lo'
        option proto 'static'
        option ipaddr '127.0.0.1'
        option netmask '255.0.0.0'

config interface 'lan'
        option ifname 'eth1'
        option type 'bridge'
        option proto 'static'
        option ipaddr '192.168.10.1'
        option netmask '255.255.255.0'
        option ip6assign '60'

config interface 'wan'
        option ifname 'eth0'
        option proto 'dhcp'

config interface 'wan6'
        option ifname '@wan'
        option proto 'dhcpv6'

config interface 'wwan'
        option proto 'dhcp'

config interface 'fonopen'
        option ifname 'wopen'
        option proto 'static'
        option ipaddr '192.168.182.1'
        option netmask '255.255.255.0'
        option ip6assign '64'
        option ip6class 'wan6'
        option auto '1'

config interface 'foneap'
        option ifname 'weap'
        option proto 'static'
        option ipaddr '192.168.183.1'
```

```
        option netmask '255.255.255.0'
        option ip6assign '64'
        option ip6class 'wan6'
        option auto '0'

config interface 'webui'
        option ifname 'br-lan'
        option proto 'static'
        option ipaddr '169.254.10.1'
        option netmask '255.255.0.0'
```

7.1.2 /etc/config/wireless

```
config wifi-device 'radio'
        option type 'mac80211'
        option channel 'auto'
        option hwmode '11ng'
        option path 'platform/ar934x_wmac'
        option htmode 'HT20'
        list ht_capab 'LDPC'
        list ht_capab 'SHORT-GI-20'
        list ht_capab 'SHORT-GI-40'
        list ht_capab 'TX-STBC'
        list ht_capab 'RX-STBC1'
        list ht_capab 'DSSS_CCK-40'
        option disabled '0'

config wifi-iface 'hotspot'
        option ifname 'wopen'
        option device 'radio'
        option network 'fonopen'
        option mode 'ap'
        option isolate '1'
        option encryption 'none'
        option macaddr 'ff:ff:ff:ff:ff:ff'
        option ssid 'ssid pub'

config wifi-iface 'private'
        option ifname 'wpriv'
        option device 'radio'
        option network 'lan'
        option mode 'ap'
        option ssid 'ssid priv'
        option encryption 'psk2'
        option macaddr 'ff:ff:ff:ff:ff:ff'
        option wps_pushbutton '1'
        option wps_device_name 'device name'
        option wps_manufacturer 'Manufacturer'
        option key 'key'

config wifi-iface 'hotspoteap'
        option ifname 'weap'
        option device 'radio'
        option network 'foneap'
        option mode 'ap'
        option isolate '1'
        option encryption 'wpa2'
        option auth_server '127.0.0.1'
        option auth_port '1818'
        option auth_secret 'eureka'
```

```
        option acct_server '127.0.0.1'
        option acct_port '1818'
        option acct_secret 'eureka'
        option macaddr 'ff:ff:ff:ff:ff:ff'
        option ssid 'ssid eap'
        option hidden '0'

config wifi-iface 'wcli'
        option ifname 'wcli'
        option device 'radio'
        option network 'wan'
        option mode 'sta'
        option ssid 'default'
        option disabled '1'
        option macaddr 'ff:ff:ff:ff:ff:ff'
```

7.1.3 /etc/config/dhcp

```
config dhcp lan
        option interface        lan
        option start            100
        option limit            63
        option leasetime        600s

config dhcp hotspotopen
        option interface        fonopen
        option start            100
        option limit            63
        option leasetime        600s

config dhcp hotspoteap
        option  interface       foneap
        option  start           100
        option  limit           63
        option  leasetime       600s

config dhcp wan
        option interface        wan
        option ignore           1
```

7.1.4 ifconfig output

This is the output of an *ifconfig* call with the previous configuration

```
br-lan    Link encap:Ethernet  HWaddr C4:71:30:3E:84:B8
          inet addr:192.168.10.1  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:6151 (6.0 KiB)

eth0      Link encap:Ethernet  HWaddr C4:71:30:3E:84:B9
          inet addr:192.168.0.78  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:709 errors:0 dropped:1 overruns:0 frame:0
          TX packets:723 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:222884 (217.6 KiB)  TX bytes:78635 (76.7 KiB)
```

```
          Interrupt:4

eth1      Link encap:Ethernet  HWaddr C4:71:30:3E:84:B8
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:5

ifb0      Link encap:Ethernet  HWaddr 62:8B:6F:67:D0:24
          UP BROADCAST RUNNING NOARP  MTU:1500  Metric:1
          RX packets:787 errors:0 dropped:0 overruns:0 frame:0
          TX packets:787 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:32
          RX bytes:92596 (90.4 KiB)  TX bytes:92596 (90.4 KiB)

ifb1      Link encap:Ethernet  HWaddr 7E:E1:5C:A2:E9:39
          UP BROADCAST RUNNING NOARP  MTU:1500  Metric:1
          RX packets:635 errors:0 dropped:0 overruns:0 frame:0
          TX packets:635 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:32
          RX bytes:214557 (209.5 KiB)  TX bytes:214557 (209.5 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:849 errors:0 dropped:0 overruns:0 frame:0
          TX packets:849 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:57238 (55.8 KiB)  TX bytes:57238 (55.8 KiB)

wopen     Link encap:Ethernet  HWaddr C4:71:30:3E:84:B4
          inet addr:192.168.182.1  Bcast:192.168.182.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:794 errors:0 dropped:0 overruns:0 frame:0
          TX packets:640 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:89098 (87.0 KiB)  TX bytes:227217 (221.8 KiB)

weap      Link encap:Ethernet  HWaddr C4:71:30:3E:84:B6
          inet addr:192.168.183.1  Bcast:192.168.183.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:4706 (4.5 KiB)

wpriv     Link encap:Ethernet  HWaddr C4:71:30:3E:84:B5
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:6001 (5.8 KiB)
```

7.2 Compiling in a SDK based on OWRT

These makefiles allows to select the package throught the *make menuconfig* of a sdk. The modules source code are placed in git repositories.

7.2.1 Hotspotd

- NOTE: The extensions can be selected/deselected through the menuconfig

```
include $(TOPDIR)/rules.mk

PKG_NAME:=hotspotd
PKG_RELEASE:=
PKG_VERSION:=

PKG_SOURCE_PROTO:=git
PKG_SOURCE_URL:=<repository_url>
PKG_SOURCE_SUBDIR:=$(PKG_NAME)-$(PKG_VERSION)
PKG_SOURCE_VERSION:=
PKG_SOURCE:=$(PKG_NAME)-$(PKG_VERSION)-$(PKG_SOURCE_VERSION).tar.gz

include $(INCLUDE_DIR)/package.mk

HOTSPOTD_extensions:=
HOTSPOTD_src_remove:=

HOTSPOTD_extensions+=rpc

ifeq ($(CONFIG_PACKAGE_hotspotd_fonapi),y)
HOTSPOTD_extensions+=fonapi
else
HOTSPOTD_src_remove+=fonapi
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_fonctl),y)
HOTSPOTD_extensions+=fonctl
else
HOTSPOTD_src_remove+=fonctl
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_radius),y)
HOTSPOTD_extensions+=radius
else
HOTSPOTD_src_remove+=radius
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_tunnel),y)
HOTSPOTD_extensions+=tunnel
else
HOTSPOTD_src_remove+=tunnel
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_eapradserv),y)
HOTSPOTD_extensions+=eapradserv
else
HOTSPOTD_src_remove+=eapradserv
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_localusers),y)
HOTSPOTD_extensions+=localusers
else
HOTSPOTD_src_remove+=localusers
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_fonrpc),y)
HOTSPOTD_extensions+=fonrpc
```

```
else
HOTSPOTD_src_remove+=fonrpc
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_ubus),y)
HOTSPOTD_extensions+=ubus
else
HOTSPOTD_src_remove+=ubus
endif

ifeq ($(CONFIG_PACKAGE_hotspotd_service),y)
HOTSPOTD_extensions+=service
else
HOTSPOTD_src_remove+=service
endif

MAKE_FLAGS += LIBDL=no LIBUCI=shared LIBUBUS=shared LIBRT=builtin MODULES_BUILTIN="$( ↩
    HOTSPOTD_extensions)"


ifneq ($(CONFIG_PACKAGE_hotspotd_testuser),)
MAKE_FLAGS += DEBUG_TUNNEL=1
endif


define Build/Prepare
        $(call Build/Prepare/Default,)
        for dir in $(HOTSPOTD_src_remove); do rm -rf $(PKG_BUILD_DIR)/ext_$$$$dir; done
        cp -r ./doc $(PKG_BUILD_DIR)/
        mkdir -p $(TOPDIR)/src
        cd $(BUILD_DIR); tar cfz release_$(PKG_SOURCE) $(PKG_NAME)-$(PKG_VERSION); mv  ↩
            release_$(PKG_SOURCE) $(TOPDIR)/src/
endef

define Package/hotspotd/default
  SECTION:=fon
  CATEGORY:=Fon
  MAINTAINER:=
endef

define Package/hotspotd
  $(call Package/hotspotd/default)
  TITLE:=Hotspot Daemon
  DEPENDS:=+libuci +libubus +libubox +kmod-sched +kmod-nfnetlink-log +kmod-nf-conntrack- ↩
      netlink +kmod-sched-connmark +iptables +iptables-mod-nat-extra +iptables-mod-conntrack ↩
      -extra +iptables-mod-ipopt +kmod-ifb +kmod-nfnetlink-queue +iptables-mod-nfqueue + ↩
      libopenssl
endef

define Package/hotspotd/config
        config PACKAGE_hotspotd_fonapi
                bool "FonAPI client backend"
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_fonctl
                bool "FonCTL hotspotd CLI"
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_radius
                bool "RADIUS extension and client backend"
```

```
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_tunnel
                bool "PPP tunnel client backend"
                depends on PACKAGE_hotspotd && PACKAGE_ppp-mod-pppol2tp
                default n

        config PACKAGE_hotspotd_localusers
                bool "extension to allow users defined locally"
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_eapradserv
                bool "hotspotd support for eap authentication"
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_fonrpc
                bool "extension to support fonrpc protocol"
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_ubus
                bool "extension to support ubus communication"
                depends on PACKAGE_hotspotd
                default n

        config PACKAGE_hotspotd_service
                bool "extension to support user type services access"
                depends on PACKAGE_hotspotd
                default n

endef

define Package/hotspotd/install
        $(INSTALL_DIR) $(1)/sbin/
        $(INSTALL_BIN) $(PKG_BUILD_DIR)/hotspotd $(1)/sbin/
        ln -sf hotspotd $(1)/sbin/fonctl
        cp -r ./files/* $(1)
endef

$(eval $(call BuildPackage,hotspotd))
```

7.2.2 Freecwmp2

```
include $(TOPDIR)/rules.mk

PKG_NAME:=freecwmp2
PKG_VERSION:=
PKG_RELEASE=$(PKG_SOURCE_VERSION)

PKG_SOURCE_SUBDIR:=$(PKG_NAME)-$(PKG_VERSION)
PKG_SOURCE_URL:=
PKG_SOURCE_PROTO:=git
PKG_SOURCE_VERSION:=
PKG_SOURCE:=$(PKG_NAME)-$(PKG_VERSION)-$(PKG_SOURCE_VERSION).tar.bz2


PKG_MAINTAINER:=
```

```
include $(INCLUDE_DIR)/package.mk
include $(INCLUDE_DIR)/cmake.mk

CMAKE_OPTIONS = -DHTTPS_ENGINE=openssl -DBACKEND=openwrt -DLIBJSON=json-c -DCRLSYNC=1

define Package/freecwmp2
  SECTION:=fon
  CATEGORY:=Fon
  TITLE:=freecwmp2 client
  DEPENDS:=+libjson +libubus +ubusd +libblobmsg-json +libuci +libopenssl
endef

define Build/Prepare/actions
        mkdir -p  $(PKG_BUILD_DIR)/actions
        $(CP) ./src/* $(PKG_BUILD_DIR)/actions
endef

define Build/Compile/actions
        $(TARGET_CONFIGURE_OPTS) \
                CFLAGS="$(TARGET_CFLAGS) -std=gnu99" \
                REAL_CFLAGS="$(TARGET_CFLAGS)" \
                LDFLAGS="$(TARGET_LDFLAGS)" \
                CONFIGURE_ARGS="$(CONFIGURE_ARGS)" \
                PKG_BUILD_DIR="$(PKG_BUILD_DIR)" \
                LIBTOOL="$(LIBTOOL)" \
                STAGING_DIR_HOST="$(STAGING_DIR_HOST)" \
                $(MAKE) -C $(PKG_BUILD_DIR)/actions
endef

define Package/freecwmp2/conffiles
/etc/config/freecwmp2
endef

define Build/Prepare
        $(call Build/Prepare/Default,)
        $(call Build/Prepare/actions)
endef

define Build/Compile
        $(call Build/Compile/Default,)
        $(call Build/Compile/actions)
endef

define Package/freecwmp2/install
        $(INSTALL_DIR) $(1)/usr/sbin/
        $(INSTALL_BIN) $(PKG_BUILD_DIR)/freecwmp2 $(1)/usr/sbin/
        $(INSTALL_BIN) $(PKG_BUILD_DIR)/freecwmp2-backend $(1)/usr/sbin/
        $(INSTALL_BIN) $(PKG_BUILD_DIR)/crlsync $(1)/usr/sbin/
        $(INSTALL_DIR) $(1)/etc/config
        $(INSTALL_DATA) ./files/freecwmp2.config $(1)/etc/config/freecwmp2
        $(INSTALL_DIR) $(1)/etc/init.d
        $(INSTALL_BIN) ./files/freecwmp2.init $(1)/etc/init.d/freecwmp2
        $(INSTALL_DIR) $(1)/sbin/
        $(INSTALL_BIN) ./files/cwmp-factory-reset $(1)/sbin/cwmp-factory-reset
        $(INSTALL_BIN) ./files/cwmp-abort $(1)/sbin/cwmp-abort
        $(INSTALL_BIN) ./files/cwmp-commit $(1)/sbin/cwmp-commit
        $(INSTALL_BIN) ./files/cwmp-download $(1)/sbin/cwmp-download
        $(INSTALL_BIN) ./files/cwmp-apply $(1)/sbin/cwmp-apply
        $(INSTALL_BIN) ./files/cwmp-begin $(1)/sbin/cwmp-begin
endef
```

```
$(eval $(call BuildPackage,freecwmp2))
```