

Design of LTE Solution

VERSION: 1.0.0

Author:	Vladimirs Zinovjevs	DATE:	28/12/16	SIGNATURE:	
Team Leader:		DATE:		SIGNATURE:	
RND Manager:		DATE:		SIGNATURE:	



DISTRIBUTION LIST:

NOTICE

This document contains proprietary and confidential material of MRV communication. Any unauthorized, reproduction, use or disclosure of this material, or any part thereof, is strictly prohibited. This document is solely for the use of MRV communication and any authorized customers. MRV communication reserves the right to make changes in the specifications at any time and without notice.




	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

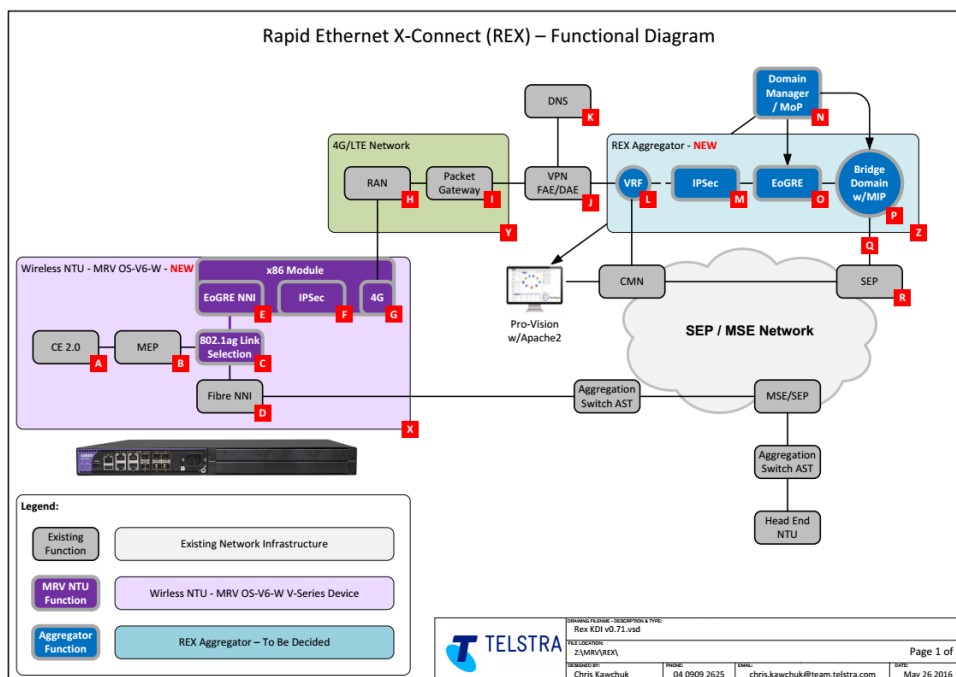
Table of Contents

1.	Introduction.....	4
1.	Application Purpose	4
2.	Overall Functional Description	4
3.	x86 Pluggable Module	6
4.	LTE LED State Description.....	6
2.	Components	7
1.	Components Scheme	7
2.	IPSec Service.....	7
3.	Firewall Service.....	9
4.	Sysctl Service	10
5.	Syslog Service	10
6.	User Management.....	10
7.	UDEV Subsystem	10
8.	Gobi Drivers.....	10
9.	Networking	11
10.	LTE Manager.....	11
11.	LTE Service	13
12.	MRV Service.....	13
3.	Process Flow	14
1.	Main Process Flow.....	14
4.	Other Operations	15
1.	Installation.....	15
2.	Image Creation	15
3.	Image Upgrade	17
4.	Zero Touch Procedure	17
5.	Modem Firmware Upgrade.....	18
5.	Applications	18
1.	Application A	18
2.	Application B	18
5.2.1.	Combination of Gobi drivers from Sierra and AT commands.	18
5.2.2.	Combination of Gobi drivers from Sierra and QMI API.....	19
5.2.3.	Linux in-tree implementation	19

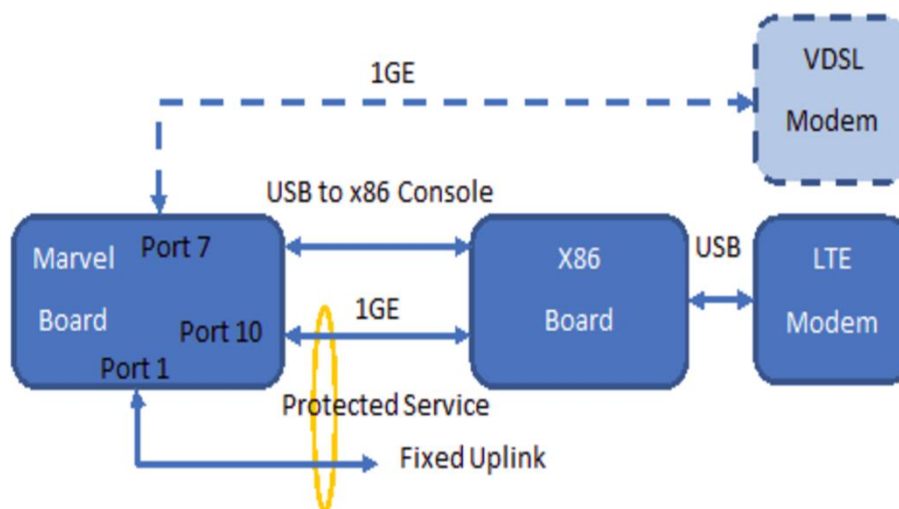
	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

5.2.4.	MBIM.....	19
6.	Links.....	19

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			



x86 board is connected to Marvel board over 1GE port. This 1GE port is bridged together with landline uplink port.




IPSec tunnel terminates on x86 board. IPSec tunnel local/remote addresses are assigned according to first phase of zero touch procedure as described later.

GRE tunnel is terminated on Marvel board. GRE tunnel is established as a part of second phase of zero touch procedure.

The x86 Linux IPSec interface address is routed over x86 LAN port, which is connected over backplane to Marvel board internal port. Static routes are used to provide the internal routing connectivity between Marvel and x86 boards over internal port so GRE packets from Marvel board can reach IPSec tunnel on x86 board.

At Marvel board side internal port is used to create GRE tunnel, which then used as a protection leg to uplink fixed port.

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

3. x86 Pluggable Module

Picture. LTE pluggable module with LTE modem



x86 board is constructed with carrier and plugin module cards. FPGA and EEPROM are located on carrier card. x86 board is supervised by moduled, which is running on OS. Moduled controls power and functional status of x86 board, LTE modem and IPSec tunnel.

x86 board console is accessible from OptiSwitch (OS) Linux shell using minicom over internal USB port. Another way is ssh connection from linux CLI and OS CLI.

4. LTE LED State Description


Front panel LEDs are managed via FPGA by moduled. LED management reflects x86 board/LTE modem and uplink statuses with three states:

- LED for module power
- LED for wireless link status
- LED for IPSec status

LED states are

- OFF - no link
- YELLOW – there is some error on link
- GREEN – normal operation (sync)
- IPSec LED states are
- YELLOW – IPSec is inactive or down
- BLINKING GREEN – IPSec is connecting
- GREEN – IPSec tunnel is up
- OFF – IPSec is unconfigured or failed

Module status changes are logged into Marvel board syslog. LTE module status and overall reports are available from CLI and NetConf.

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

2. Components

The functionality of LTE solution depends on several components. They include components running on Marvel board, components running on x86 pluggable module and a separate LTE modem board. Marvel board is a host for moduled Daemon, Module API library, Module UID Requests API, PSS daemon, PSS UID Requests API, OSMD CLI commands and VTYSH CLI commands, GRE component. All these components are briefly explained in Application Design of module daemon document. x86 pluggable module is a host for IPSec, firewall, sysctl, syslog services, user management, UDEV subsystem, QMI and USB drivers from Gobi, networking subsystem, LTE manager, LTE and MRV services.

1. Components Scheme

Next scheme represents relationship between the components related to handling of modules in OS devices:

PICTURE!!!!

2. IPSec Service

IPSec service is running on x86 module. Ubuntu system uses strongswan as IPSec implementation.

Service health can be checked by

```
sudo systemctl status strongswan
```

The output from correctly configured and fully functional service looks like

- strongswan.service - strongSwan IPsec services

Loaded: loaded (/lib/systemd/system/strongswan.service; disabled; vendor preset: enabled)

Active: active (running) since Fri 2016-12-16 18:12:16 IST; 2 days ago

Process: 18876 ExecStopPost=/bin/rm -f /var/run/charon.pid /var/run/starter.charon.pid (code=exited, st

Process: 18859 ExecStop=/usr/sbin/ipsec stop (code=exited, status=0/SUCCESS)

Process: 18885 ExecStart=/usr/sbin/ipsec start (code=exited, status=0/SUCCESS)

Process: 18882 ExecStartPre=/bin/mkdir -p /var/lock/subsys (code=exited, status=0/SUCCESS)

Main PID: 18903 (starter)

Tasks: 18

Memory: 1.8M

CPU: 37.737s

CGroup: /system.slice/strongswan.service

└─18903 /usr/lib/ipsec/starter --daemon charon

└─18905 /usr/lib/ipsec/charon --use-syslog

Dec 19 14:45:05 os-v8-m charon[18905]: 14[NET] received packet: from 84.237.228.43[4500] to 95.35.141.179


Dec 19 14:45:05 os-v8-m charon[18905]: 14[ENC] parsed INFORMATIONAL request 8160 []

Dec 19 14:45:05 os-v8-m charon[18905]: 14[ENC] generating INFORMATIONAL response 8160 []

Dec 19 14:45:05 os-v8-m charon[18905]: 14[NET] sending packet: from 95.35.141.179[4500] to 84.237.228.43[

Dec 19 14:45:05 os-v8-m charon[18905]: 10[NET] received packet: from 84.237.228.43[4500] to 95.35.141.179

Dec 19 14:45:05 os-v8-m charon[18905]: 10[ENC] parsed INFORMATIONAL response 9267 []

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

Dec 19 14:45:25 os-v8-m charon[18905]: 13[NET] received packet: from 84.237.228.43[4500] to 95.35.141.179

Dec 19 14:45:25 os-v8-m charon[18905]: 13[ENC] parsed INFORMATIONAL request 8161 []

Dec 19 14:45:25 os-v8-m charon[18905]: 13[ENC] generating INFORMATIONAL response 8161 []

Dec 19 14:45:25 os-v8-m charon[18905]: 13[NET] sending packet: from 95.35.141.179[4500] to 84.237.228.43[

Status of IPSec tunnel can be checked by

`sudo ipsec status`

The output from fully functional IPSec tunnel looks like

Security Associations (1 up, 0 connecting):

mrsv[1]: ESTABLISHED 2 days ago, 95.35.141.179[a4:60:32:00:d4:b6]...84.237.228.43[server.mrv]

mrsv{1}: INSTALLED, TUNNEL, reqid 1, ESP SPIs: cb9e9c0b_i c184f32a_o

mrsv{1}: 5.6.7.8/32[gre] === 1.2.3.4/32[gre]

Life time (stability) of tunnel can be seen by the same command

mrsv[1]: ESTABLISHED 2 days ago

IPSec has 2 configuration files. Both files are uploaded by OS software from configuration server and are missing after clean install.

- /etc/ipsec.secrets contains PSK key. Current solution uses PSK as authentication algorithm. IDs must be the same as in ipsec.conf file.

@server.mrv @a4:60:32:00:d4:b6 : PSK "mrsv-key123456"

- /etc/ipsec.conf contains tunnel configuration

config setup

strictcrpolicys=yes

uniqueids = no

conn %default

ikelifetime=1d

keylife=20m

rekeymargin=3m

keyingtries=1

keyexchange=ikev2

authby=secret

type=tunnel

reauth=no


rekey=no

dpdaction=clear

dpddelay=20s

conn mrsv

left=%defaultroute

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

```

leftsubnet=5.6.7.8/32[gre]
leftid=@a4:60:32:00:d4:b6
leftfirewall=yes
right=84.237.228.43
rightsubnet=1.2.3.4/32[gre]
rightid=@server.mrv
auto=add

```

Some notes about IPSec configuration:

- Left means NTU.
- left=%defaultroute, ip address of IPSec on NTU is taken from default ip route (show ip route).
- leftid=@a4:60:32:00:d4:b6, id is configured based on OptiSwitch base MAC address, which is unique.
- leftsubnet=5.6.7.8/32[gre], GRE endpoint on optiswitch. Only GRE traffic comes into IPSec tunnel even if destination address points to provider's GRE endpoint. If it is required to send other traffic to IPSec tunnel then leftsubnet=5.6.7.8/32 options can be used.
- leftfirewall=yes, automatically modifies iptables when tunnel is up and down.
- Right means provider's side.
- rightsubnet=1.2.3.4/32[gre], GRE endpoint address on provider's side
- rightid=@server.mrv, id of server. Must be the same on both sides.
- DPD mechanism provided by strongswan is used for tunnel health monitoring.
- Do not use keyexchange=ikev1 as IPSec tunnel is destroyed each time during ike renegotiation.
- Try to keep tunnel lifetime as big as possible to avoid unnecessary tunnel renegotiations. It is possible that tunnel is destroyed during renegotiations of 2nd phase. (Must be checked by Clay to understand Juniper behaviour).
- Re-keying and re-auth must be properly configured on both sides to avoid unnecessarily out of service periods. Out of service periods are clearly visible during long stability test as periodical traffic drops.

3. Firewall Service

Iptables are used as firewall. There is a set of static rules, which is defined in /usr/local/bin/firewall_ipsec.sh. Rules are added at each system start from /etc/rc.local.

The following rules prepare system to IPSec:

- allow UDP ports 500 and 4500
- allow ESP and AH protocols
- mark IPSec traffic with DSCP class cs7.

The following rules protects the system:


- deny SSH traffic coming to LTE modem interface
- allow ICMP from x86 board
- allow SSH from x86 board
- allow SSH traffic coming to backplane interfaces
- deny all other incoming traffic.

There are also several rules added by IPSec on the fly, when tunnel is up. E.g.:

```

-A FORWARD -s 1.2.3.4/32 -d 5.6.7.8/32 -i enp0s29u1u1i8 -p gre -m policy --dir in --pol ipsec --reqid 1 --proto esp -j ACCEPT

```

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

-A FORWARD -s 5.6.7.8/32 -d 1.2.3.4/32 -o enp0s29u1u1i8 -p gre -m policy --dir out --pol ipsec --reqid 1 --proto esp -j ACCEPT

4. Sysctl Service

Sysctl rules are added in /etc/sysctl.d/98-mrv-lte.conf and do the following:

- allow ip forwarding
- disable rp filtering on LTE modem interface.

5. Syslog Service

MRV defines the remote syslog client in /etc/rsyslog.d/98-mrv-lte.conf. Currently, remote client is located on OS.

6. User Management

Permanent /etc/sudoers.d/98-mrv-lte file allows mrv user to be a sudoer and access ttyUSB port.

7. UDEV Subsystem

Udev subsystem reacts on modem attach/detach according to /usr/local/bin/mrv-service.sh file.

KERNEL=="qcqmi0", ACTION=="add", RUN+="/bin/sh -c /usr/local/bin/lte-start.sh"

KERNEL=="qcqmi0", ACTION=="remove", RUN+="/bin/sh -c /usr/local/bin/lte-stop.sh"

It launch either lte-start.sh or lte-stop.sh scripts.

/usr/local/bin/lte-start.sh starts LTE service.

```
#!/bin/sh
```

```
systemctl start lte.service
```

```
exit $?
```

/usr/local/bin/lte-stop.sh kills both LTE and MRV services.

```
cat /usr/local/bin/lte-stop.sh
```

```
#!/bin/sh
```

```
systemctl stop lte.service
```

```
systemctl stop mrv.service
```


```
exit 0
```

8. Gobi Drivers

A combination of Sierra drivers with QMI support and AT commands is used to manage modem.

GobiSerial is used to control USB part of device and GobiNet for managing network side. Drivers are available after registration from <http://source.sierrawireless.com/resources/airprime/software/usb-drivers-linux-qmi-software-s2,-d-,26n2,-d-,39/> page. New drivers must be placed in router/module/x86-lte directory (e.g. S2.27N2.40). The current content of earlier mentioned directory:

Makefile S2.25N2.36 S2.26N2.38 S2.26N2.39 S2.27N2.40 send-at ubuntu. Update GOBI_VERSION (e.g. GOBI_VERSION = S2.27N2.40) in router/module/x86-lte/Makefile. Drivers must be patched. There is an issue with rawip mode in GobiNet driver. It seems that ubuntu attaches different mac address twice to the same network interface controlled by GobiNet. GobiNet driver contains a piece of code, which preserves mac address after init phase. Once mac has been stored it is used for all operations by GobiNet. Kernel, from its side, accepts only the latest mac address, so all packets coming with preserved mac address are dropped quietly. MRV provides a patch to solve the issue. It can be found in git history:

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

cd S2.26N2.39

git log .

commit 390338da9c725b33e27e9f27b15c5317079fefef

Author: Vladimir Zinovjev <vzinovjevs@mrsv.com>

Date: Mon Nov 28 14:57:32 2016 +0200

1. apply MRV path on top of the latest Sierra driver
2. take 2.26.39 into use

This patch must be applied each time a fresh code has been downloaded from Sierra site. Please note, that according to release notes of S2.27N2.40 there is "Dynamic detect between Ethernet mode & RAWIP mode", which can be a solution of rawip issue but must be carefully tested.

Drivers must be recompiled in case of ubuntu kernel update (change KERNEL_VERSION = 4.4.0-51-generic in router/module/x86-lte/Makefile).

Drivers are located in

/lib/modules/XYZ-generic/kernel/drivers/net/usb/GobiNet.ko

/lib/modules/XYZ-generic/kernel/drivers/usb/serial/GobiSerial.ko.

To prevent qcserial and qmi_wwan drivers from control overtaking both are disabled in /etc/modprobe.d/blacklist-modem.conf.

GobiSerial exposes 3 serial devices ttyUSB0, ttyUSB1 and ttyUSB2. ttyUSB2 is a control device and AT commands are sent to it.

GobiNet exposes 2 interfaces with a names based on modem position (e.g. enp0s29u1u1i8 and enp0s29u1u1i10). USB interface 8 is described as enpXXxi8 and is utilized by OS LTE solution. USB interface 10 is described as enpXXxi10 and is unused.

The behaviour of LTE modem can be changed by the extended set of AT commands (like, switch to one network interface or switch from QMI to MBIM).

It is a configurable behaviour on boot and/or modem levels whether network interface's name is position based or looks as ethX. Currently, it is tried to avoid any interface renaming and follow basic system configuration.

9. Networking

/etc/network/interfaces provides static configuration for backplane interfaces. E.g.

```
auto enp2s0
```

```
iface enp2s0 inet static
```

```
address 169.254.1.1
```

```
netmask 255.255.255.0
```

```
mtu 9000
```

And hotplug for LTE modem interface. E.g.


```
allow-hotplug enp0s29u1u1i8
```

```
iface enp0s29u1u1i8 inet dhcp
```

10. LTE Manager

LTE manager is a bash script, which is located in /usr/local/bin/lte-mgr.sh file. It operates with 2 types of commands: sets of AT commands and non-AT commands. The list of supported options can be retrieved by 'lte-mgr.sh help'.

Non AT commands are defined inside lte-mgr.sh (e.g. 'ipsec status' and 'info statistics').

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

AT commands are specified in

<http://source.sierrawireless.com/resources/airprime/minicard/74xx/4117727-airprime-em74xx-mc74xx-at-command-reference/>.

Sets of AT commands are located in /usr/local/bin/commands directory. One can easily add his own set of commands like:

```
cat /usr/local/bin/commands/reset
```

```
atv1
```

```
ATE0
```

```
at+cme=2
```

```
AT!RESET
```

As soon as a set is created it is usable by lte-mgr and send-at.

There are several simple rules:

- AT commands are executed one by one
- if there is '#text' in the beginning of a line then text is printed together with a result of the next AT command. Like

```
#blah-blah:
```

```
at!scact?
```

output is:


```
blah-blah:!SCACT: 1,1
```

- if one AT command fails then other are still executed
- if there is '#!' in the beginning of a line then the result of next AT command is checked and script is stopped in case of error

List of predefined sets of AT commands:

- create profile: /usr/local/bin/commands/profile_create
currently is Telstra specific only and contains APN. It can contain other than APN options, e.g. user name and password. This AT set requires reconfiguration for each ISP.
- delete profile: /usr/local/bin/commands/profile_delete
- list profiles: /usr/local/bin/commands/profile
- modem reset: /usr/local/bin/commands/reset
- power on: /usr/local/bin/commands/cfun
- sim and pin status: /usr/local/bin/commands/pin
- connect PDP: /usr/local/bin/commands/pdp_connect
- disconnect PDP: /usr/local/bin/commands/pdp_disconnect
- PDP/control bearer status: /usr/local/bin/commands/pdp
- connect data PDP/bearer: /usr/local/bin/commands/connect
- disconnect data PDP/bearer: /usr/local/bin/commands/disconnect
- connection status: /usr/local/bin/commands/scact
- current modem state: /usr/local/bin/commands/info_state
- full modem state: /usr/local/bin/commands/info
- diagnostics: /usr/local/bin/commands/diag
- list of supported commands: /usr/local/bin/commands/support

Usually sets are launched either by 'lte-mgr.sh command' or 'sudo send-at command'.

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

If one wants to use minicom directly then he has to rename lte-mgr.sh to prevent moduled of messing up AT commands.

11. LTE Service

LTE service starts, configures modem and monitors the state of IPsec tunnel. Service status changes are logged into syslog.

LTE service itself is supervised by systemd and is restarted in case of failure. The systemd service description is located in /lib/systemd/system/lte.service.

[Unit]

Description=MRV LTE interface

After=remote-fs.target

After=syslog.target

[Service]

ExecStart=/usr/local/bin/lte-daemon.sh

Restart=on-failure

RestartSec=10

Service is a bash script located in /usr/local/bin/lte-daemon.sh file.

Service health can be checked by

sudo systemctl status lte

The usual output looks like

- lte.service - MRV LTE interface

Loaded: loaded (/lib/systemd/system/lte.service; static; vendor preset: enabled)

Active: active (running) since Fri 2016-12-16 18:12:16 IST; 2 days ago

Main PID: 18827 (lte-daemon.sh)

Tasks: 2

Memory: 212.0K

CPU: 9min 17.926s

CGroup: /system.slice/lte.service

└─18827 /bin/sh /usr/local/bin/lte-daemon.sh

└─23636 sleep 10

LTE service creates /tmp/.mrv_lte_modem when initial profile is pushed to modem (after every x86 module restart).

LTE service expects that /tmp/.mrv_lte is created by OS software as a last step of x86 configuration. Missing /tmp/.mrv_lte file is treated as inconsistent IPsec configuration.


12. MRV Service

MRV service is used to bring up IPsec tunnel. It is started when modem has been configured, external interface is up and IPsec service has been started. MRV service executes /usr/local/libexec/ipsec/client_up file (if such exists) just after there has been an attempt to bring up the tunnel. OS can upload that file if necessary. File can specify some client specific actions but currently is not used.

Systemd does not monitor the health of MRV service. The systemd service description is located in /lib/systemd/system/mrv.service.

[Unit]

Description=MRV ipsec service

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

After=strongswan.service lte.service

Requires=strongswan.service lte.service

[Service]

ExecStart=/usr/local/bin/mrv-service.sh

TimeoutStartSec=2s

[Install]

WantedBy=multi-user.target

Service is a bash script located in /lib/systemd/system/mrv.service file.

Service health can be checked by

sudo systemctl status mrv

The usual output looks like

- mrv.service - MRV ipsec service

Loaded: loaded (/lib/systemd/system/mrv.service; disabled; vendor preset: enabled)


Active: inactive (dead)

3. Process Flow

1. Main Process Flow

The main flow is:

- Modem is attached as 3 usb serial devices. AT control port is at /dev/ttyUSB2.
- Gobi drivers create /dev/qcqmIX device.
- Udev starts LTE service on qcqmIX device attach (and stops on detach).
- LTE service pushes provider's profile to modem if necessary (/tmp/.mrv_lte_modem is missing).
- Checks SIM card and PIN. If SIM is blocked or missing then LTE is stopped.
- LTE service tries to connect to PDP/bearer.
 - FYI, there is a difference between 3G and LTE attach procedures:
 - Device has no active PDP after start in 3G network. It requires 'connect' command to get ip and be able to send/receive data.
 - Device has always active control bearer (ip address exists) after start in LTE network. But data bearer (network traffic) requires separate 'connect' command.
- Modem is now up and external network is available.
- LTE service checks whether zero touch procedure has already been completed. If it has not then service is stopped. Connection to internet remains alive in any case.
- After LTE connection has been established zero touch procedure gets an access to a remote server, downloads and executes the configuration script. The config script must restart LTE service after all necessary files have been copied.
- LTE service launches IPSec and MRV services.
- LTE service constantly (once per 10 seconds) monitors the state of IPSec tunnel. If it does not exist then modem is restarted and LTE is stopped.

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

4. Other Operations

1. Installation

Prerequisites for debian installation are: usbutils, tcpdump, strongswan, minicom, awk and sed.

During the installation:

- mrv user is created
- ssh keys are pushed on x86
- Gobi drivers are installed
- new sysctl options are applied
- new rsyslog options are applied
- backplane interface is located and /etc/network/interfaces is updated. Currently only one backplane interface is supported.
- LTE modem interface is located and all files containing its configuration are updated. Eth0 is LTE modem interface by default in all configuration files.

2. Image Creation

x86 pluggable module uses ubuntu server 16.04.1 LTS as an operational system. Image must be deployed to a system with a small SSD drive (< 3.5 GB).

BIOS boot menu must be launched for installation (F10 for the current devices). Non-UEFI option should be selected.

The following options are used during system clean installation:

- english language and keyboard layout
- Israel time zone
- no home encryption
- add user mrv/123456
- manual disk partitioning
 - Primary partition on the beginning of drive with size 3.2GB is created as rootmount point with bootable flag on
 - Primary partition on the beginning of drive with size 300MB is created as swap.
- no automatic updates
- manual package selection and open ssh server
- install grub to MBR

The following must be done after system installation:


- update the system


```
sudo apt-get update
sudo apt-get upgrade
```
- disable sleep, suspend and hibernate modes


```
sudo systemctl mask sleep.target suspend.target hibernate.target hybrid-sleep.target
```

 to prevent suspending when the lid is closed set the following options in /etc/systemd/logind.conf:


```
[Login]
```

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

HandleLidSwitch=ignore

HandleLidSwitchDocked=ignore

c) disable wan drivers

```
sudo vi /etc/modprobe.d/blacklist-modem.conf
```

```
blacklist qcserial
```

```
blacklist qmi_wwan
```

d) reboot the system

e) install the necessary kernel. There are two different options:

- Install the well known kernel. Currently kernel 4.4.0.51 is used. The configured kernel version can be found in router/module/x86-lte/Makefile (KERNEL_VERSION = 4.4.0-51-generic).

```
sudo apt-get update
```

```
apt-cache search linux-image
```

```
sudo apt-get install linux-image-4.4.0.51-generic linux-image-extra-4.4.0.51-generic
```

```
sudo su
```

```
echo linux-image-4.4.0-51-generic hold | dpkg --set-selections
```

```
reboot
```

- install the latest kernel

```
sudo apt-get dist-upgrade
```

```
reboot
```

remove old kernel to free disk space

```
dpkg --get-selections|grep linux
```

```
sudo apt-get purge linux-image-4.4.0-31-generic linux-image-extra-4.4.0-31-generic linux-headers-4.4.0-31-generic linux-headers-4.4.0.31
```

```
sudo update-grub
```

```
reboot
```

modify router/module/x86-lte/Makefile (KERNEL_VERSION = 4.4.0-51-generic), recompile modem modules and recreate mrv debian package with a new version of installed kernel

f) Update grub to redirect system output to console

```
sudo vi /etc/default/grub
```

```
add
```

```
GRUB_CMDLINE_LINUX_DEFAULT="console=ttyS0,115200"
```

```
uncomment
```

```
GRUB_TERMINAL=console
```

```
sudo update-grub
```

g) install necessary packages

```
sudo apt-get install usbutils tcpdump strongswan minicom sed
```

h) disable system auto upgrade

```
sudo apt-get purge unattended-upgrades snapd
```

```
vi /etc/apt/apt.conf.d/10periodic
```


```
APT::Periodic::Update-Package-Lists "0";
```

i) Create version file and put there version number in format YYYY-MM-DD-N like "2017-01-18-1"

```
sudo vim /etc/mrv-release
```

j) create a basic snapshot of the system

- reboot

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

- insert boot (or live cd) disk and start new installation
- choose language and keyboard, continue installation until "Configure the network" menu
- activate console (e.g. press ALT-F2)
- remove installation disk
- insert an empty disk and mount it to a directory
- create image
dd if=/dev/mmcblk0 of=xyz bs=1G count=4
- reboot the system
- Copy snapshot to different system and truncate it properly taking into account disk pseudo geometry

fdisk -l ubuntu+deb-3.img

Disk ubuntu+deb-3.img: 4294 MB, 4294967296 bytes, 8388608 sectors

I/O size (minimum/optimal): 512 bytes / 512 bytes

```

Device Boot  Start    End  Blocks  Id System
ubuntu+deb-3.img1  *      2048  6250495   3124224  83 Linux
ubuntu+deb-3.img2      6250496  6836223    292864  82 Linux swap / Solaris
truncate --size=$((6836223+1)*512) ubuntu-3.img

```

k) install mrv debian package

- sudo su
- LTE_PROFILE_NAME=profile_create.telstra dpkg -i mrv-lte_20170118-1_amd64.deb
(or use another appropriate profile)
answer Y on all questions
- delete deb package
- Poweroff!!! As profile is pushed to modem only once that allows to update modems on newly burned systems.

l) create a hardware specific snapshot of system (follow point h). Snapshot is a hardware dependant as interface names can be different and is customer dependant as LTE settings (e.g. APN) are different. Different x86 boards/customers require different images.


3. Image Upgrade

TODO

4. Zero Touch Procedure

OS-V8-M base MAC address is used as device identifier. As it is currently implemented in OS production phase, its base MAC address is recorded for farther reference in PV provisioning process so auto-discovered OS device could be mapped to its physical location by the recorded MAC address. OS-V8-M base MAC address is also used as IPsec endpoint identification.

Once LTE uplink connection is established, OS software reaches the predefined configuration server by DNS name using scp and fetches an initial configuration file. This configuration file is applied to x86 module so IPsec tunnel could be auto-created. GRE tunnel configuration is applied to Marvel board. Configuration files include GRE and IPsec local and remote addresses, SSH secret key and other necessary for GRE/IPsec tunneling parameters. IPsec configuration is stored as a separate ipsec.conf file in a way specified by strongswan implementation. Configuration server stores configuration files per device named by the device

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

base MAC address. x86 Linux script fetches the base MAC address from Marvel board using ssh and uses it as a file name to get the configuration file from the server.

Once IPsec tunnel is established, the Linux script bridges it to the x86 board LAN port. After this Marvel board creates GRE tunnel and establishes L2 connectivity to service provider network and OS zero touch provisioning is executed.

TODO: fix ipsec config file: remove null config. Ilja to complete the section. What is linux script?

5. Modem Firmware Upgrade

MRV does not provide a mechanism of LTE modem firmware update at time being. There has already appeared a fresh firmware, which has been approved by Telstra. List of fresh images can be found:

http://source.sierrawireless.com/resources/airprime/minicard/74xx/airprime-em_mc74xx-approved-fw-packages/

5. Applications

1. Application A

The following has been left out of development scope.

- QoS for LTE modem. As mobile broadband network provides only best effort throughput, there is no possibility to guarantee that management/oam(VLAN 4095) service always gets the required bandwidth under heavy load or in case of narrow data channel.
- This particular EM7430 modem provides two network interfaces. Each can be used as a separate network. It is not clear how the second interface can be utilized by Telstra. Currently there is no any implementation for it in MRV code. But several mobile providers are known that require different APNs for different traffic types: e.g. one for VoIP, second for data. The implementation must adapt the second interface if required.

2. Application B

Currently, several ways for modem communication are known.

5.2.1. Combination of Gobi drivers from Sierra and AT commands.

Drivers are available after registration on


<http://source.sierrawireless.com/resources/airprime/software/usb-drivers-linux-qmi-software-s2,-d-,26n2,-d-,39/>

Pros:

- Interface is simple and flexible. Everything can be done in script style.
- Vendor support.

Cons.

- There could be issues when applying the latest Ubuntu kernel. Sometimes there are pretty long delay between kernel release and its adaptation by Sierra.
- MRV patch must be applied each time on a fresh code or Sierra should provide a support to MRV and solve the rawip issue.
- AT commands can be different for another model of LTE modem. Adaption of new modem models require extra effort.
- Gobi drivers must be recompiled each time ubuntu kernel is updated.

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

- If there is another issue then MRV could be on its own solving it.
- It sounds pretty strange to use old fashioned at commands instead of more modern QMI interface, which provides callbacks, responses and information as TLV. TLVs are more human friendly.

5.2.2. Combination of Gobi drivers from Sierra and QMI API

There is SDK from Sierra with QMI API. Initially it was planned to use this interface but some instabilities were found (that may be were caused by a mistake in PoC code). It could be Sierra's bugs as Sierra has already introduced several new versions of API.

Pros:

- More human friendly.
- Vendor support.

Cons.

- There could be issues when applying the latest Ubuntu kernel. Sometimes there are pretty long delay between kernel release and and its adaptation by Sierra.
- MRV patch must be applied each time on a fresh code or Sierra should provide a support to MRV and solve the rawip issue.
- Gobi drivers must be recompiled each time ubuntu kernel is updated.
- Requires in-house manager to properly open/close sessions in kernel. Otherwise 'out of resource' error happens and modem must be restarted.
- If a fresh version of API appears then it requires some extra effort for adaptation.
- As it has already been mentioned there were some possible instabilities.
- If there is another issue then MRV could be on its own solving it.

5.2.3. Linux in-tree implementation

There is linux in-tree QMI implementation. The following document contains pretty good description of it <http://www.lanedo.com/documents/Qualcomm%20Gobi%20devices%20on%20Linux.pdf>

Pros:

- Supported by opensource community.
- Existing and incoming issues will be solved more or less fast (most probably).
- ModemManager can be used.
- Interface is simple and flexible. Everything can be done in script style.

Cons.

- Implementation has not been tested by MRV.

5.2.4. MBIM


MBIM has not been tried it as it is not clear whether EM7430 has MBIM interface. May be Sierra support can clarify.

6. Links

[1] AT commands for EM7430

<http://source.sierrawireless.com/resources/airprime/minicard/74xx/4117727-airprime-em74xx-mc74xx-at-command-reference/>

[2] Gobi drivers

	Subject	Design of LTE Solution	Creation Date	27/12/16	
	Version	1.0.0	Update Date	28/01/17	
	Name	Vladimirs Zinovjevs			

<http://source.sierrawireless.com/resources/airprime/software/usb-drivers-linux-qmi-software-s2,-d-,26n2,-d-,39/>

[3] Approved modem firmware list

http://source.sierrawireless.com/resources/airprime/minicard/74xx/airprime-em_mc74xx-approved-fw-packages/

[4] IPSec Solution

<https://www.strongswan.org/>

[5] Qualcomm Gobi devices in Linux based systems

<http://www.lanedo.com/documents/Qualcomm%20Gobi%20devices%20on%20Linux.pdf>

[6] OS-V8-M Modules support

<https://mrvil.sharepoint.com/MRVPortal/Engineering/Shared%20Documents/Carrier%20Ethernet/OptiSwitch/OS-V8-M/OS-V8-M%20Modules%20Support%20System%20Architecture.docx?d=w4a2e62fa66d84b009482f2e6fcee5029>

[7] Application Design of module daemon

<https://mrvil.sharepoint.com/MRVPortal/Engineering/Shared%20Documents/Carrier%20Ethernet/OptiSwitch/OS-V8-M/Moduled-Application-Design.docx?d=w47722fa0ec3948daba279729666848fc>