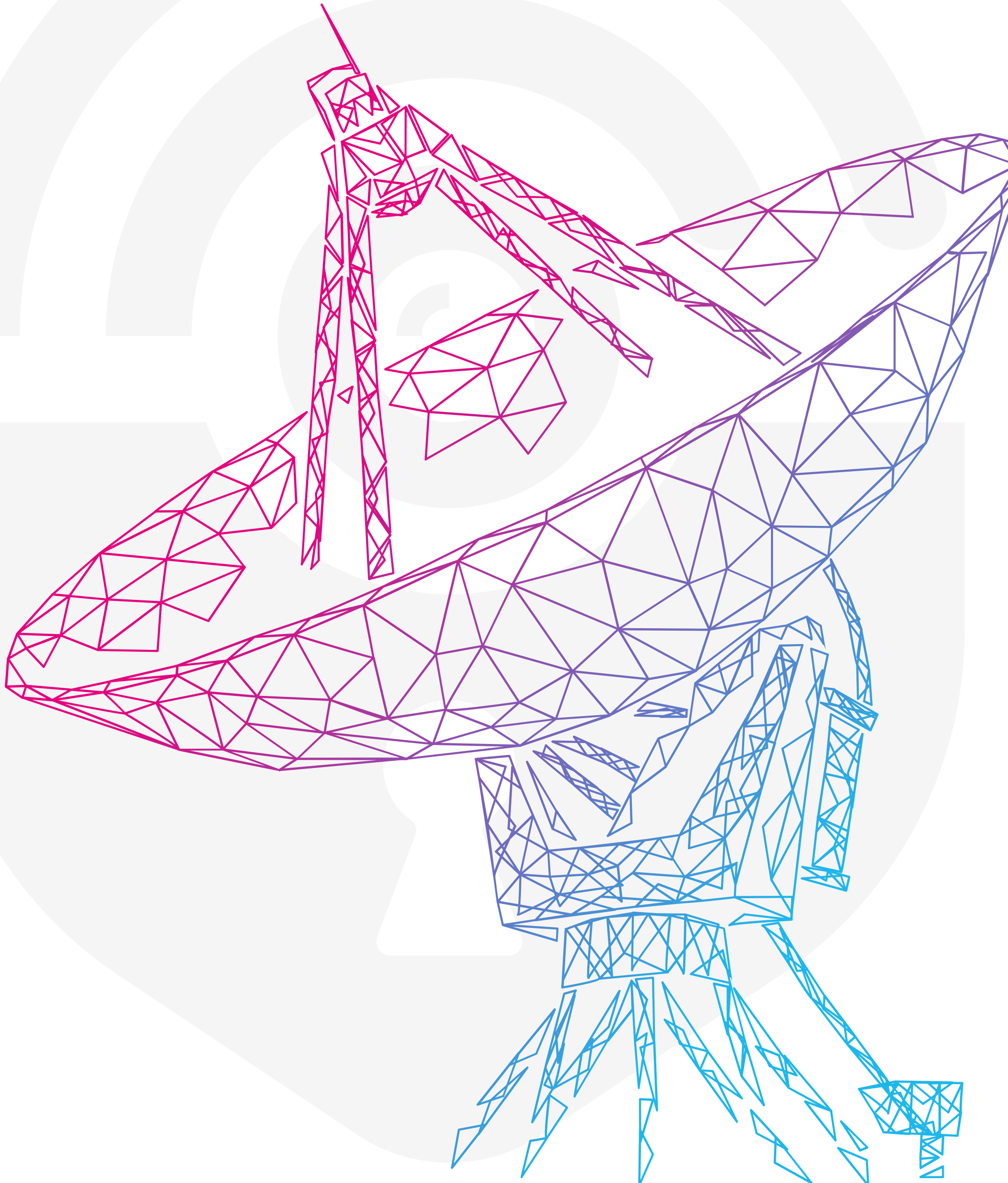
penthertz

Attacking IoT Devices through 5G interface

By Sébastien Dudek



PRASEC

About myself

Founder of Penthertz

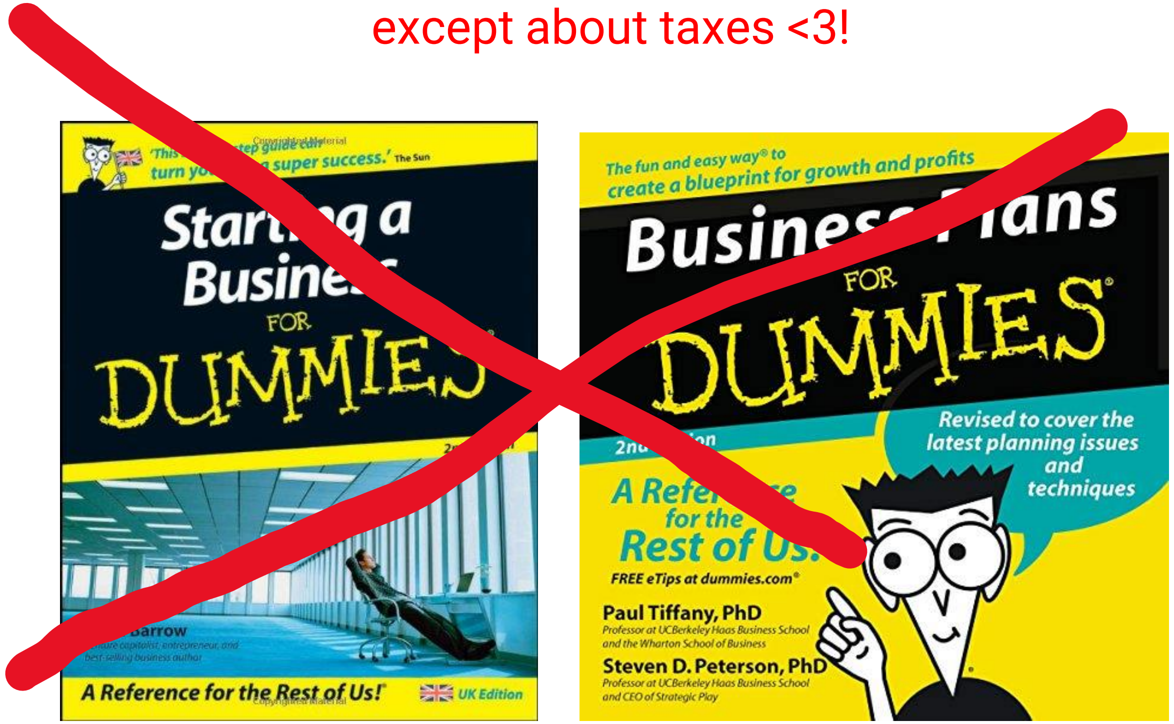
- Sébastien Dudek (@FlUxIuS)
- CTO of Penthertz (as Chief Taxes Officer...)
  - Specialized in Wireless communications security
- > 10 years of experience in Software & Hardware security
  - Security researcher
  - Pentester & Red Team
  - Vulnerability researcher

Perfect mix to make Penthertz!

- Started the company during COVID → thinking about writing a book

My next book (or not)

Forget everything you learned before,
except about taxes <3!



*but people have seen worse in restaurants… ☹

Main activities

Security assessments

- Wireless communications (RFID, Wi-Fi, Mobile communications, Bluetooth, etc.)
- Embedded devices
- Backend servers
- Red Team

Trainings

- Software-Defined Radio Hacking
- Wi-Fi Red teaming
- RFID Hacking
- Mobile attacks (2G/3G/4G/5G), and more…

Hardware security

- Firmware extraction
- Chip off
- Secrets extraction
- Library's analysis
- Vulnerability hunting

Setup to PWN the radio

Part of the SDR material

- Need to manage any type of transmission (2G-5G, Wi-Fi, Remotes, Bluetooth, ZigBee, RFID, **exotic communications**, etc.).
- Today’s challenges: handling from DC to 6 or even 8 GHz with a decent stability
- Next challenges → 30 GHz at least with mmWave bands
- Able to get large bandwidth in some situation (sometimes > 100 Msps even >= 300 Msps)

2021 Picture → the tables have never been so clean!

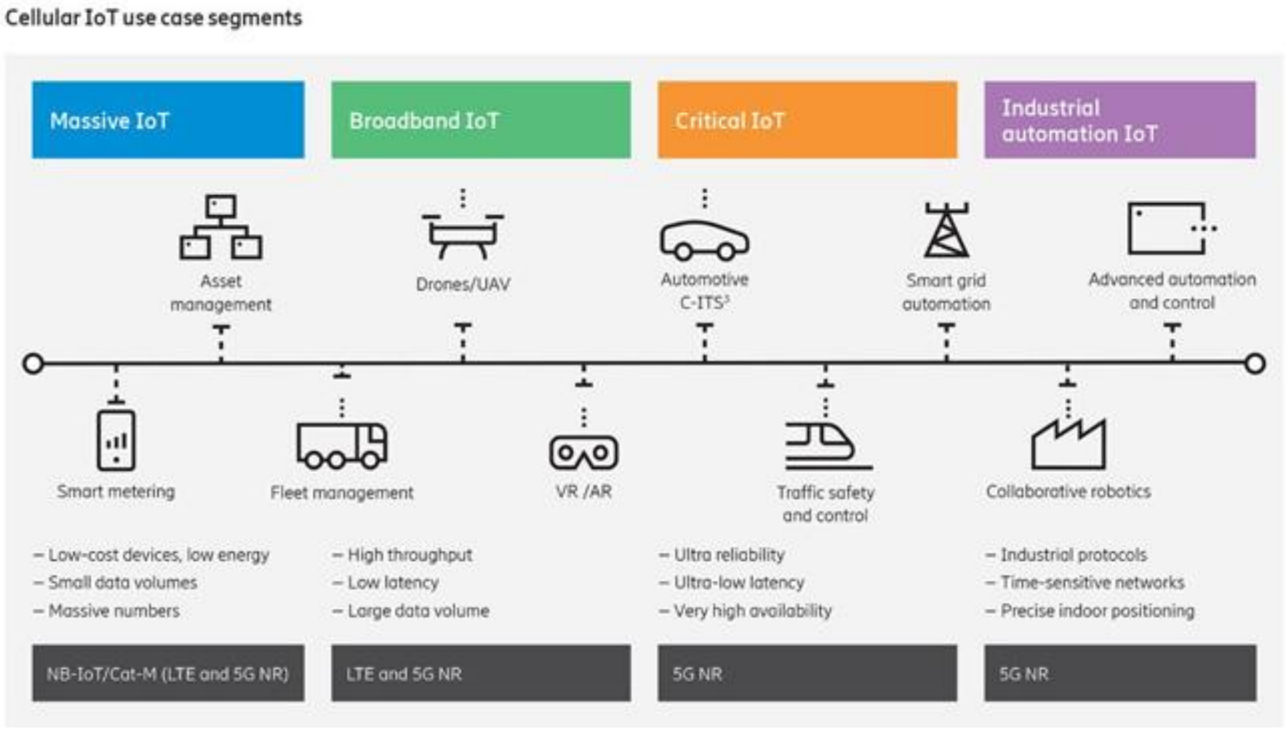
SDR has also performance limits to overcome, but let’s talk about 5G use case in IoT!

Let’s talk about 5G devices!

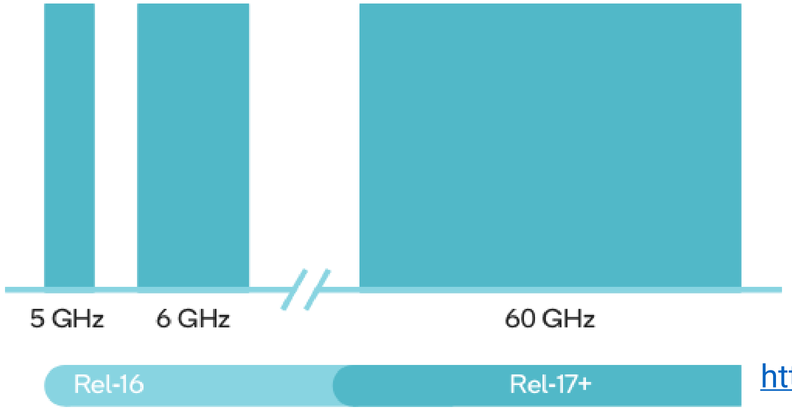
Context

5G case in IoT

- A lot of new technologies are expected in 2023/2024
- Why:
    - eMBB (enhanced Mobile Broadband → very low latency vs 4G
    - massive Machine Type Communications (mMTC)
    - Use of new bands
        - Even private/unlicensed bands



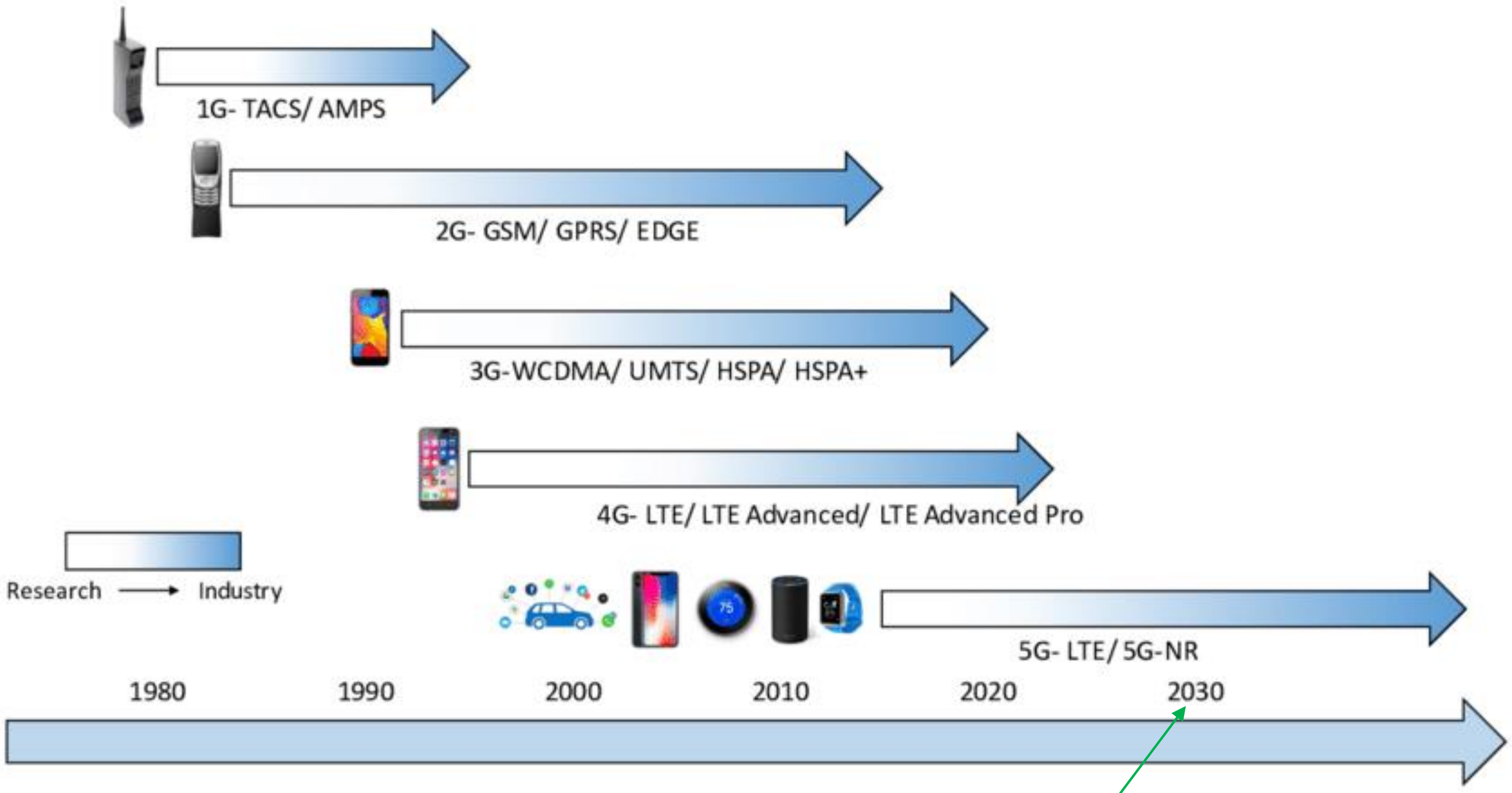
Source: https://www.helpnetsecurity.com/2019/06/14/5g-subscriptions-forecast/



https://www.qualcomm.com/

Evolution of mobile networks

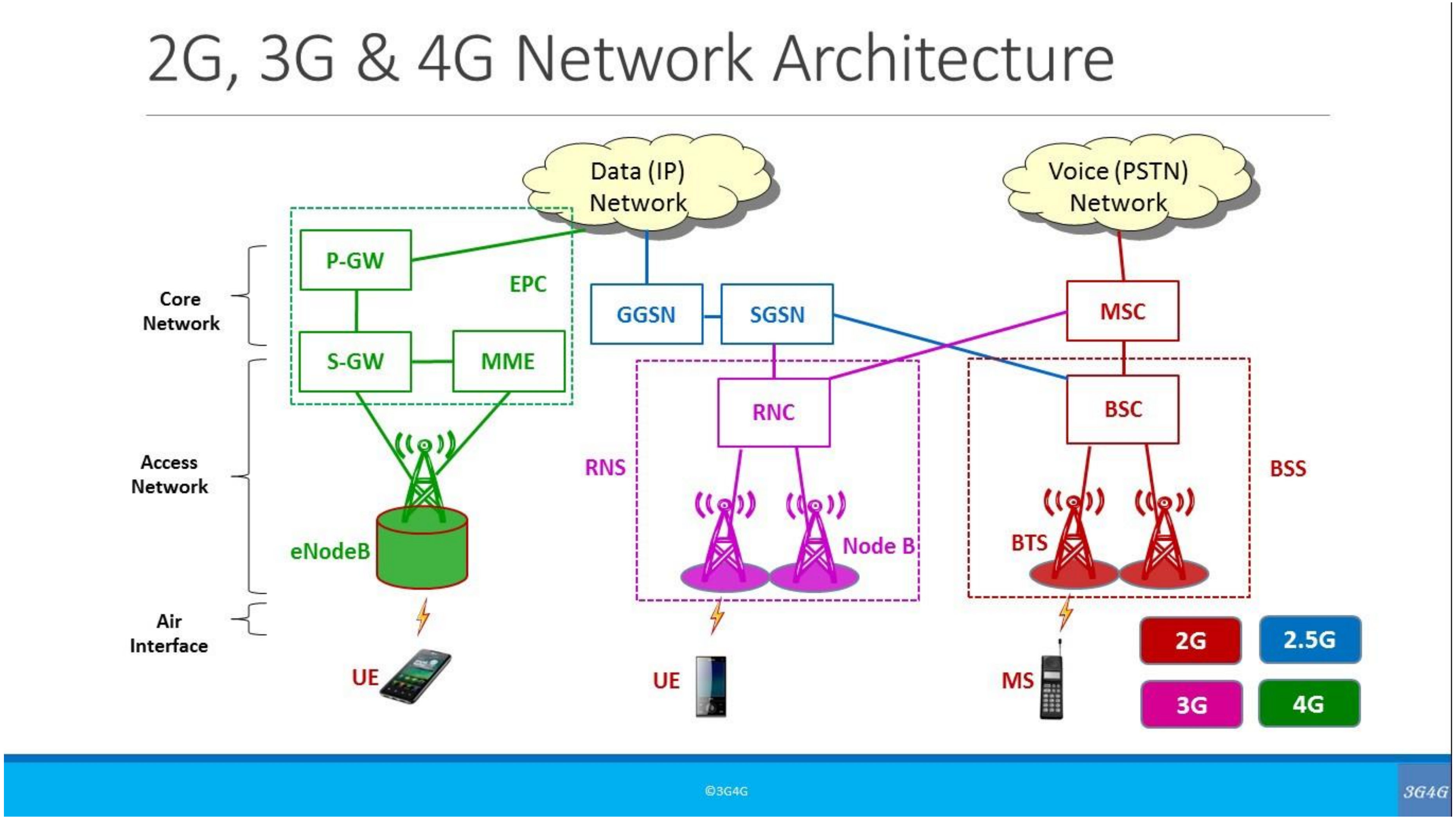
Mobile network* → more than 30 years



* Other technologies exist for US and Asia

And we are already talking about 6G!

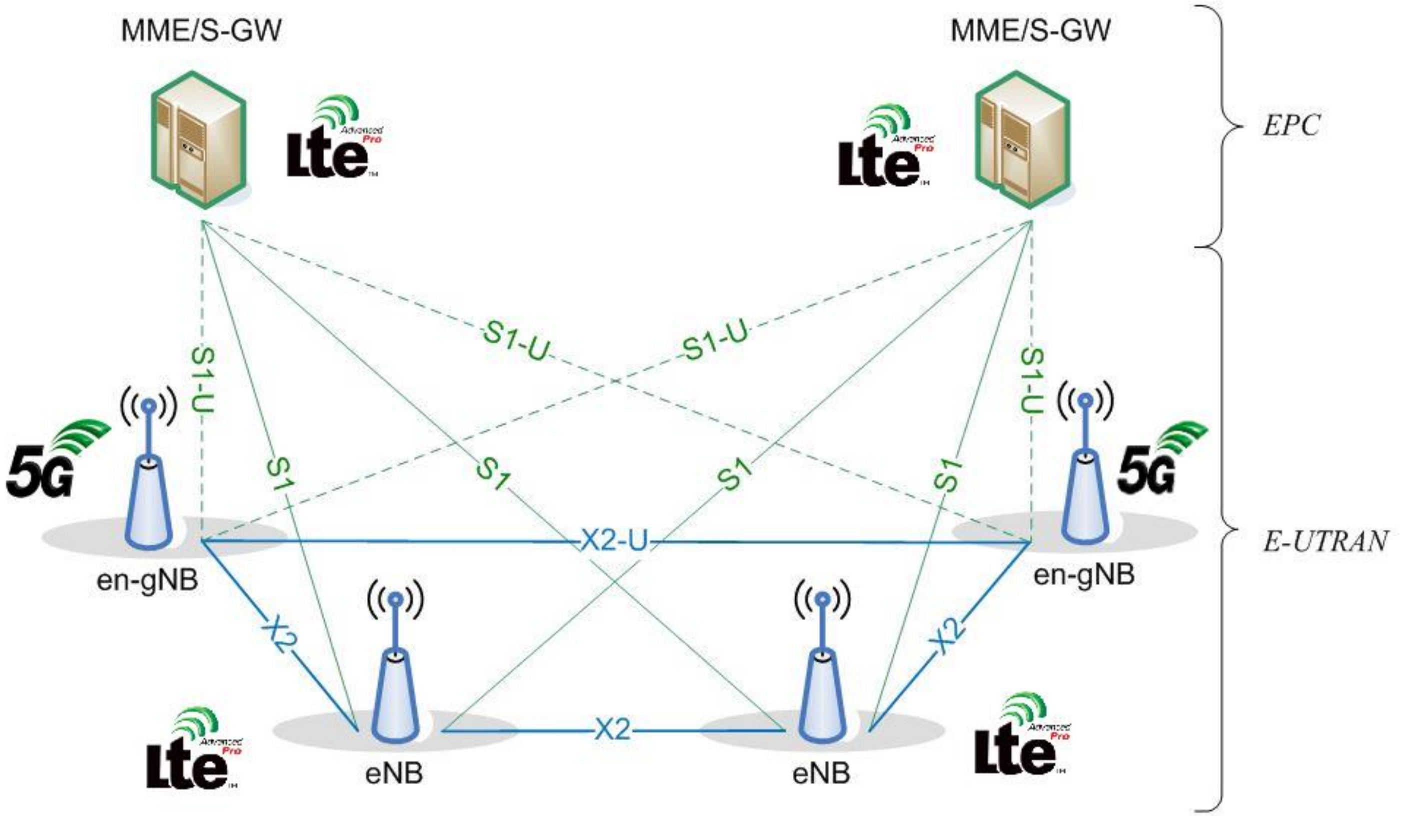
2G-4G networks



Our feeling about 5G in some EU countries…

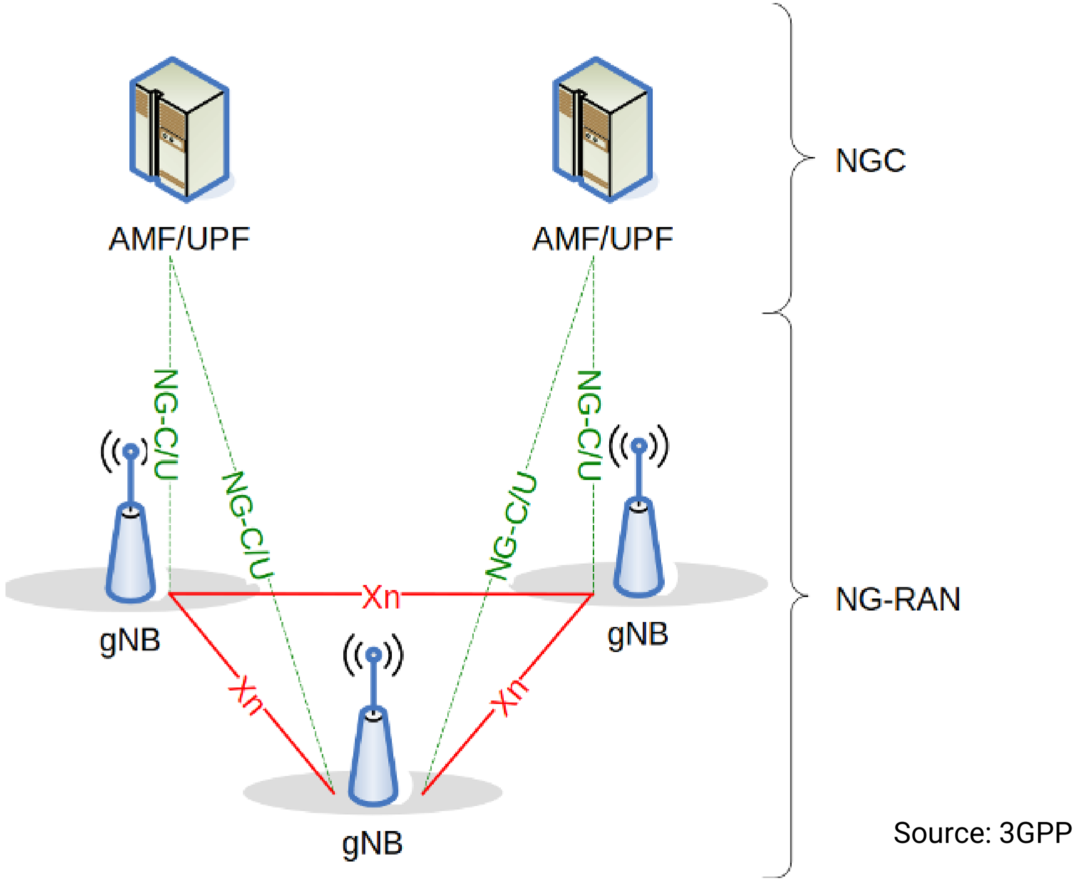
Source: 3GPP

5G NSA: what we have currently



Source: 3GPP

5G SA: what should we expect in mid-2021 2223 in FR?



Use of SDN (Software-Defined Network) → CUPS, reduce operation costs, faster services, plug/unplug instances in the network, etc.

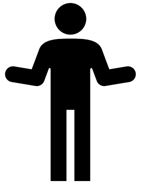
Security comparison in brief

	2G	3G	4G	5G
Client authentication	YES	YES	YES	YES
Network authentication	NO	Only in USIM mode	YES	YES
Signaling integrity	NO	YES	YES	YES
Encryption	A5/1 in use	KASUMI | SNOW-3G	SNOW-3G | AES-128 CTR | ZUC	SNOW-3G | AES-128 CTR | ZUC

State of the 5G

- People are disappointed with slow data → 5G NSA is still in use
  - But also, people paid the price for a “5G connectivity”…
- 5G SA implementation is still lagging in some EU countries
- Based on MWC 2023 comms → it’s coming soon…*

Old Meme, good soup!



*But what about your phone???

Targets

Goal

- Extract secrets exchanged between the device and the backend
- Attack the device and its exposed services
- Get knowledge on how to interact with the backend → attack!

Targets

- Everything requiring a mobile connectivity:
  - Intercoms
  - Alarms
  - IVI (In-Vehicle Infotainment) systems
  - Routers
  - Probes
  - Etc.

2019 5G Cybersecurity hackathon

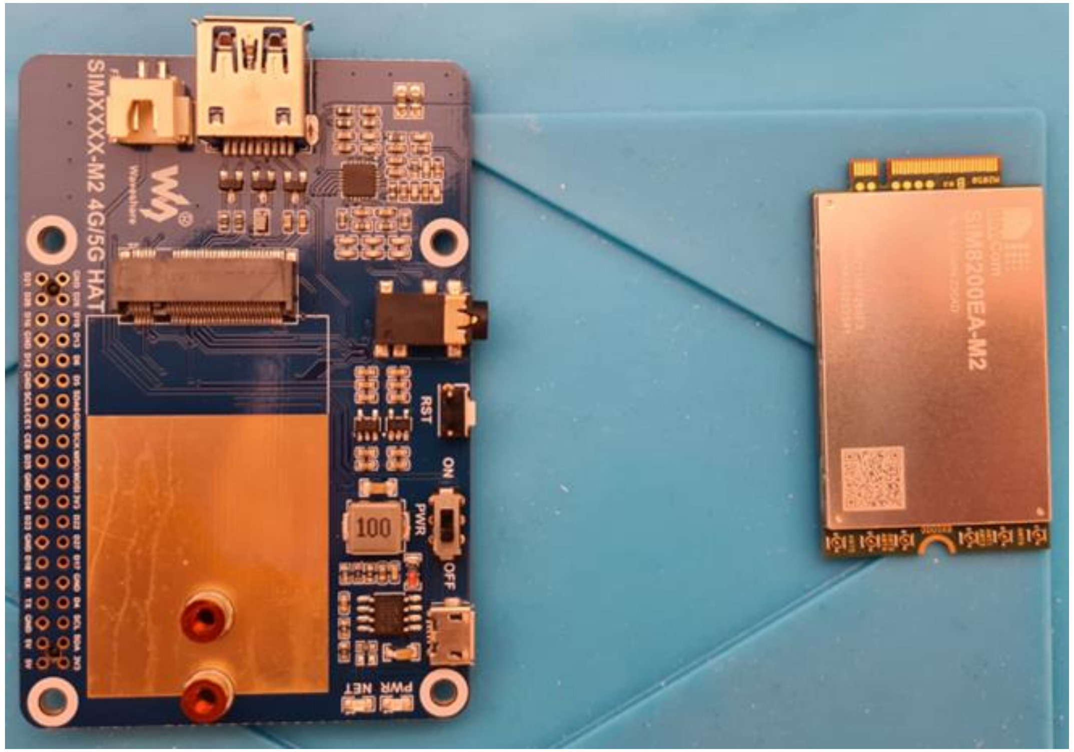
- Target to hack:
  - Read our story there: https://medium.com/mobile-stacks-and-networks-security

Our first targets since 2021

- Goal:
  - Extract secrets exchanged between the device and the backend
  - Attack the device and its exposed services
  - Get knowledge on how to interact with the backend → attack!
- Mostly devices support different mobile stacks 2G/3G/4G, etc.
- We can still use older stacks to perform assessment

Targeting 5G capable devices

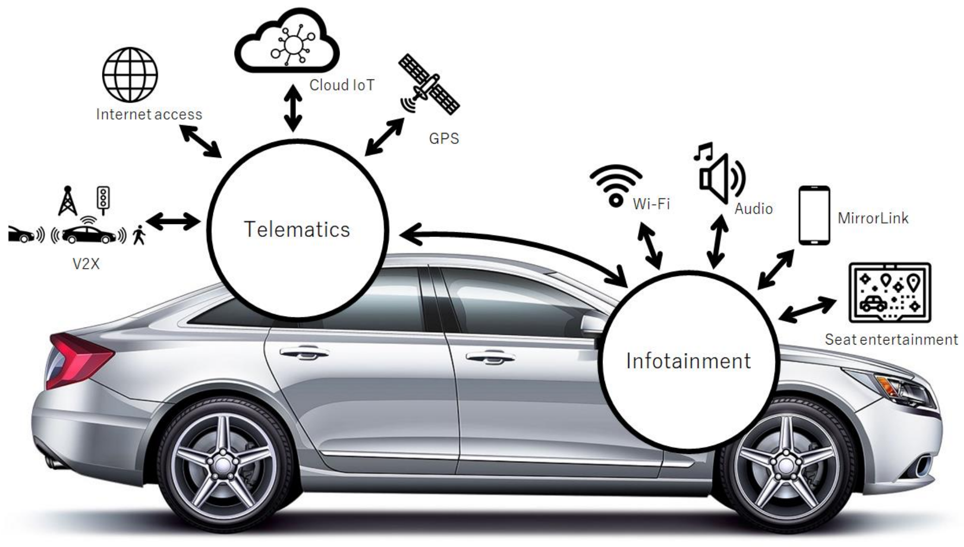
Using devkits



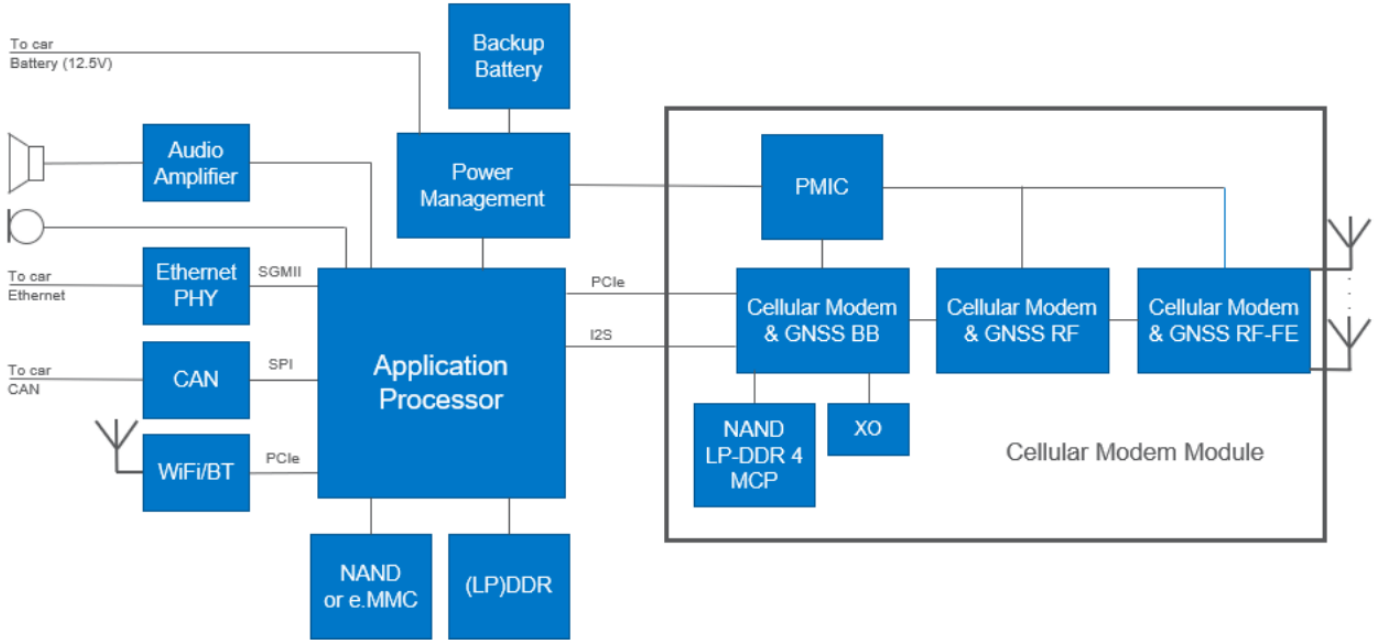
Targeting 5G capable devices

TCUs with 5G stacks used in cars

Not very common, but starting to be developed



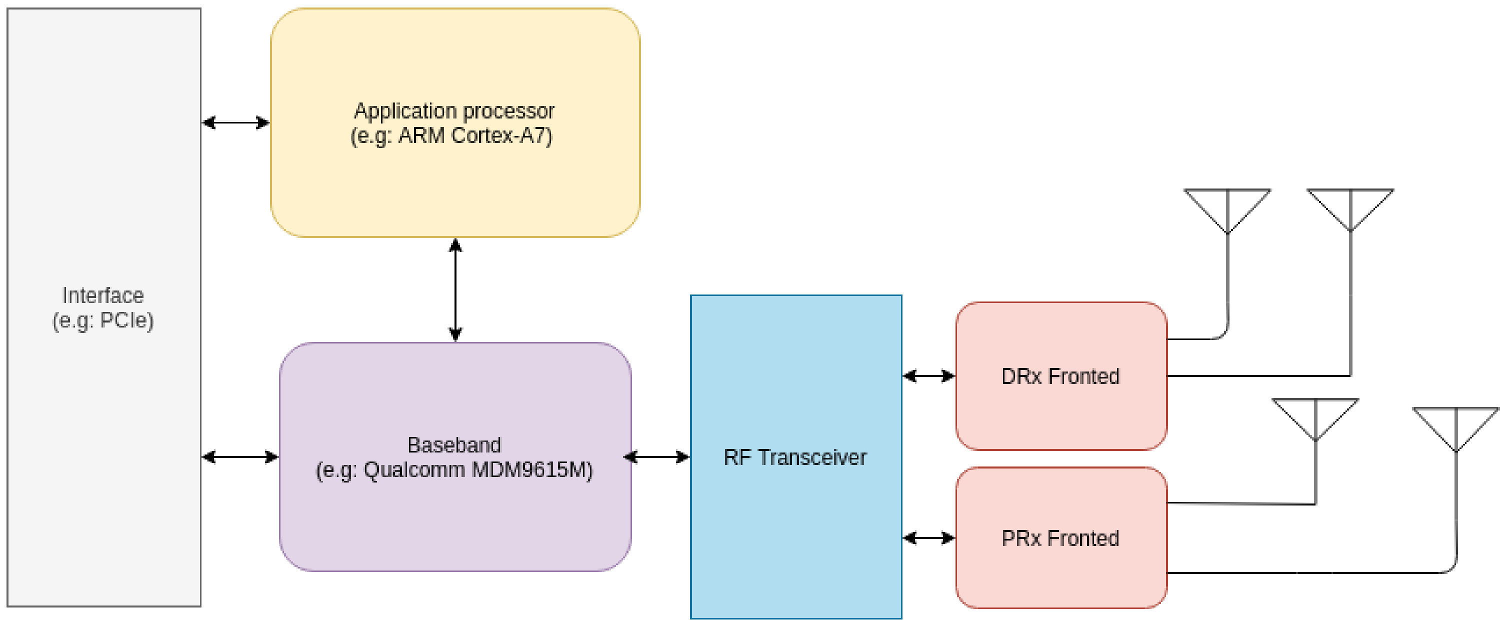
Source: https://www.i-pex.com/



Source: https://media-www.micron.com/

What do they have in common?

- Composed of:
  - Applicative processor
  - 2 frontends:
    - DRx & PRx → radio transmission
  - Baseband processor → implementing the mobile stacks
  - Memory:
    - NAND & DDR
  - And other interfaces…



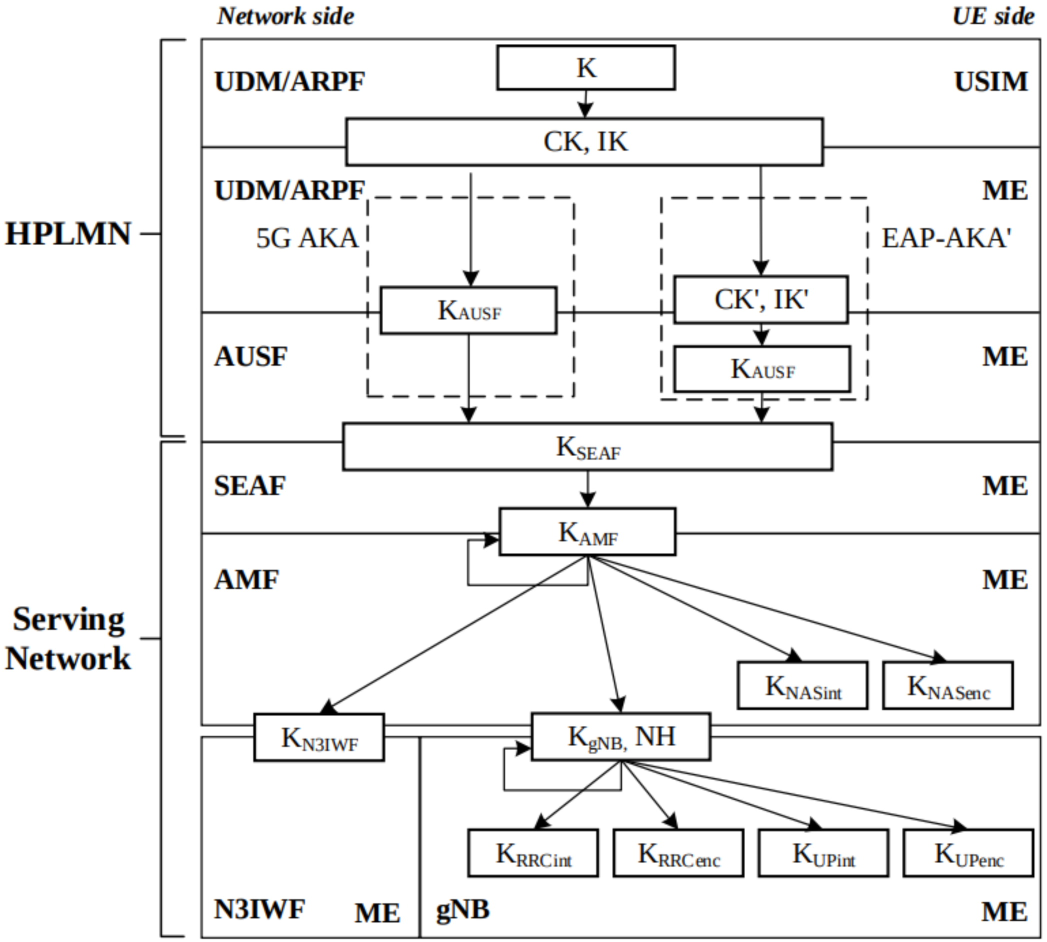
5G stack security in brief

Security mechanisms

- For integrity and confidentiality → inherit from 4G algorithms:
  - 128-NEA1/128-NIA1: SNOW-3G
  - 128-NEA2/128-NIA2: AES-128 CTR
  - and 128-NEA3/128-NIA3: 128-bit ZUC used mainly in China

Security mechanisms (2)

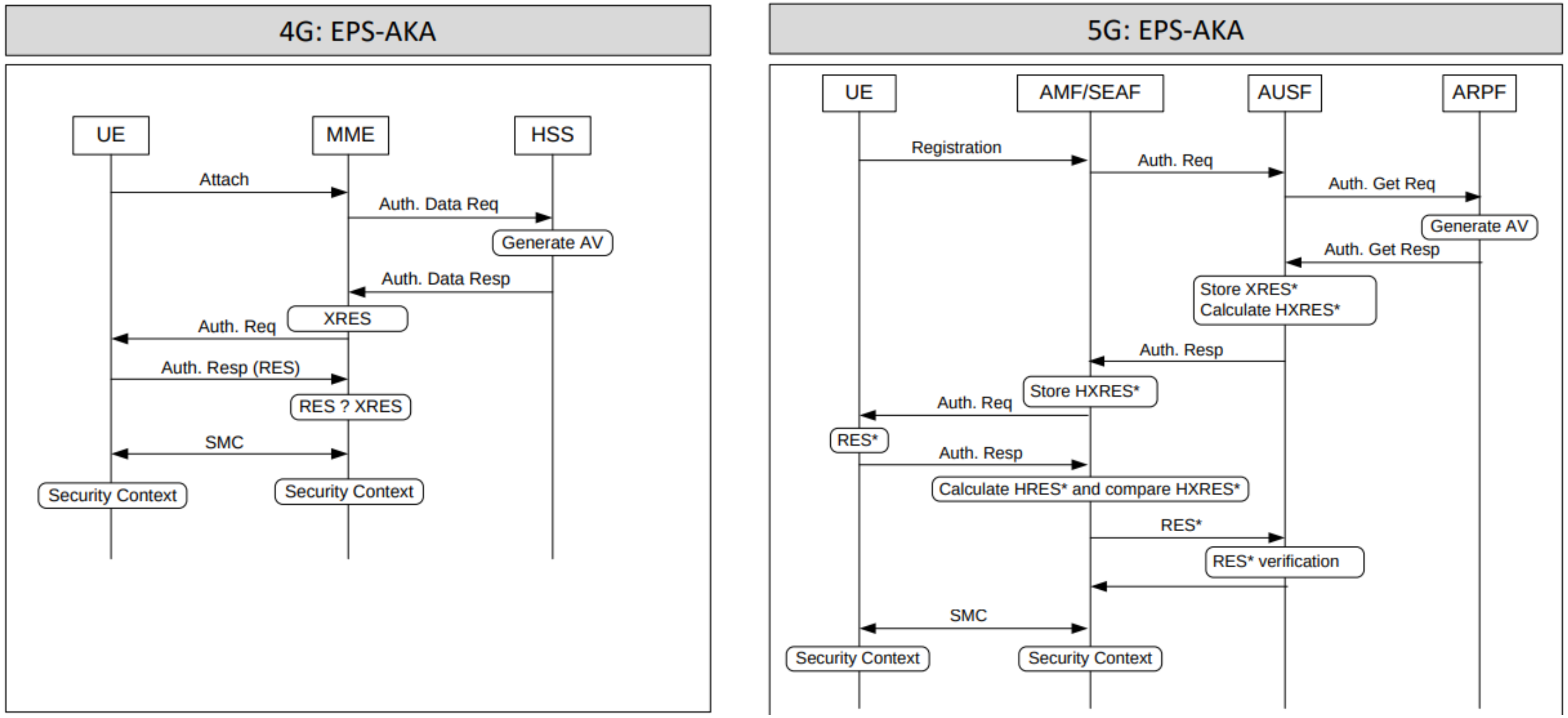
- In NSA: same architecture as 4G → same security
- In SA: improvements with authentication schemas 5G AKA, EAP-AKA’ et EAP-TLS



ETSI TS 133 501 V16.3.0 (2020-08

Security mechanisms (3)

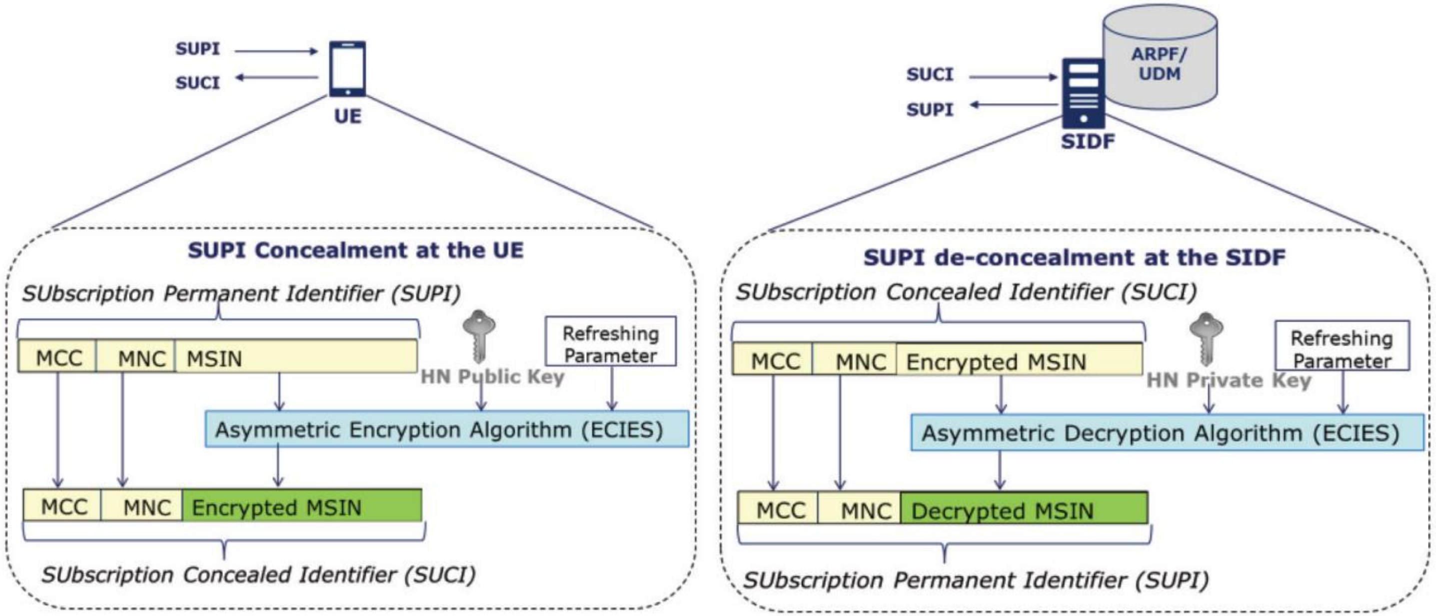
- Improved authentication method



Source: 5G Security: Standard and Technologies par Dr. Haiguang Wang, Senior Researcher, Huawei Internationa

Optional mechanisms

- User plane integrity (3GPP technical report TR 33.853) → not supported by all Ues
- Identity anonymization in SA (ETSI TS 133 501) → protect the SUPI (replacement of IMSI) → SUCI
  - Needs the ISIM to be ready with a PLMN public key
  - Can be compromised with a downgrade attack to 4G → forcing an *Attach-Request* to happen

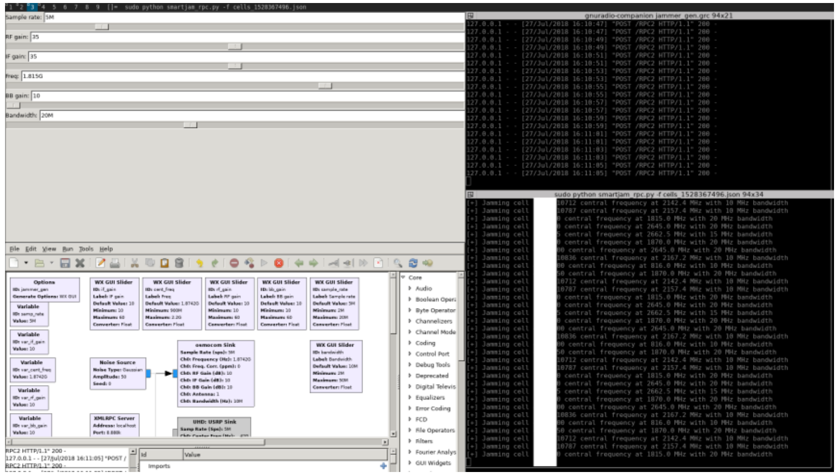


Source: 3GPP 5G Security par Anand R. Prasad, Sivabalan Arumugam, Sheeba B and Alf Zugenmaier

Downgrading security: dumb way

- Goal → downgrade to 2G → bypass mutual authentication
- Could be done naturally (parking stations, uncovered areas…)
- Complex way: using signaling vulnerability/tricks (e.g: https://www.researchgate.net/figure/Mobile-device-soft-downgrading-to-GSM-by-rogue-LTE-base-station_fig6_305401180)
- Or dumb way with Jamming:
  - Dedicated device → available on AliExpress
  - Smart-jamming:
    - ModMobMap (soon available for 5G) & Modmobjam

    → https://github.com/PentHertz/Modmobjam



Caution: > stacks to downgrade → more resources!

Caution 2: Jamming is not legal

Downgrading security: smart way

- Like for 4G, playing with Tracking Area Update procedure → reject causes → make the baseband switching to older stacks → need to modify srsRAN's stack
- New: 5G NSA NEA0 Bidding-Down Attack + 5G to 2G demonstration in "Never Let Me Down Again: Bidding-Down Attacks and Mitigations in 5G and 4G" by Bedran Karakoc, Nils Fürste, David Rupprecht, Katharina Kohls from Radix-security

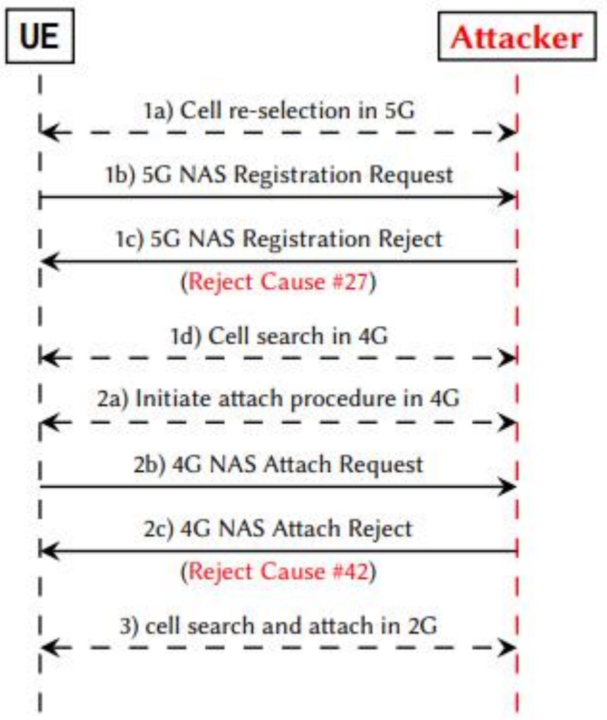


Figure 3: Protocol flow of downgrade dance from 5G to 2G.

Downgrading security: smart way (2)

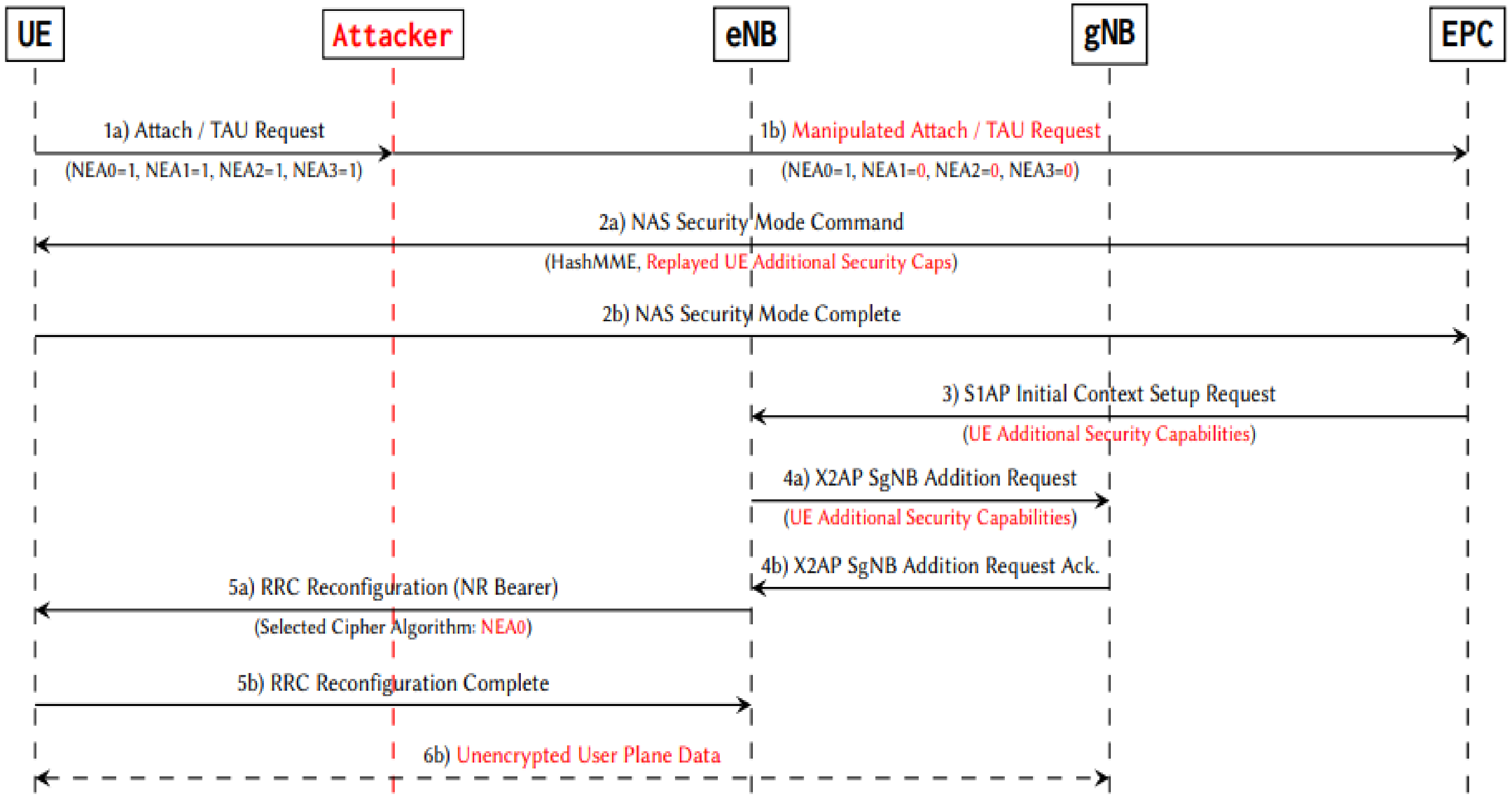


Figure 2: 5G NSA NEA0 Bidding-Down Attack.

https://radix-security.com/files/2021_downgrade.pdf

Attack setup

Isolation

- Use of Faraday cage/shield: ~5000€ (for custom one) → be warned about reflections
- Goal: avoid conflict with operators + isolate from other devices, noise, etc.
- Cheaper ways:
  - A home-made can be made + using attenuation in software + hardware → but not certified ¯_(ツ)_/¯
  - **Possible to use good well-made SMA extension cable → requiring an antenna rework sometimes**

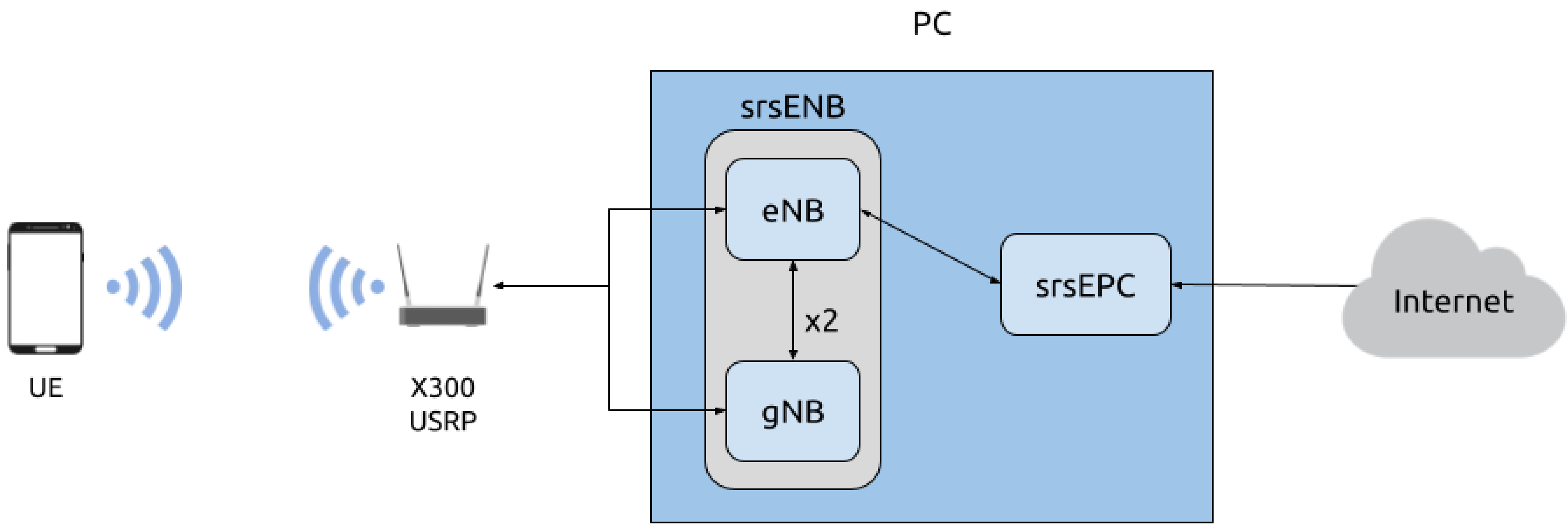
Our setup starting 2021

- Commercial:
  - Amarisoft → preferable for SA for the moment
    - Minimum budget: >20 000€ (hardware + license)
- Other options but more expensive:
  - Rohde&Schwarz
  - Keysight

Mini version only supports 5G-NR SA

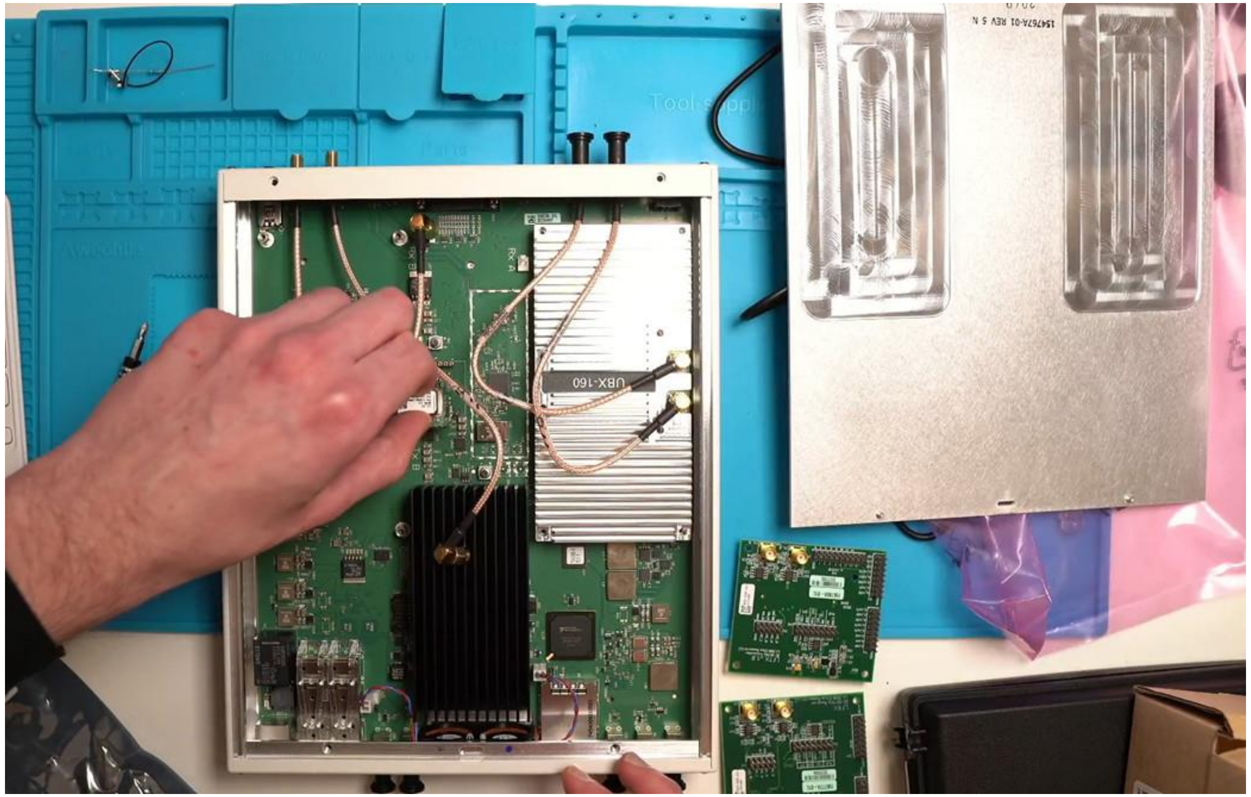
5G NSA with Opensource today

- Two main projects:
  - OpenAirInterface5G
  - srsRAN → our favorite stack



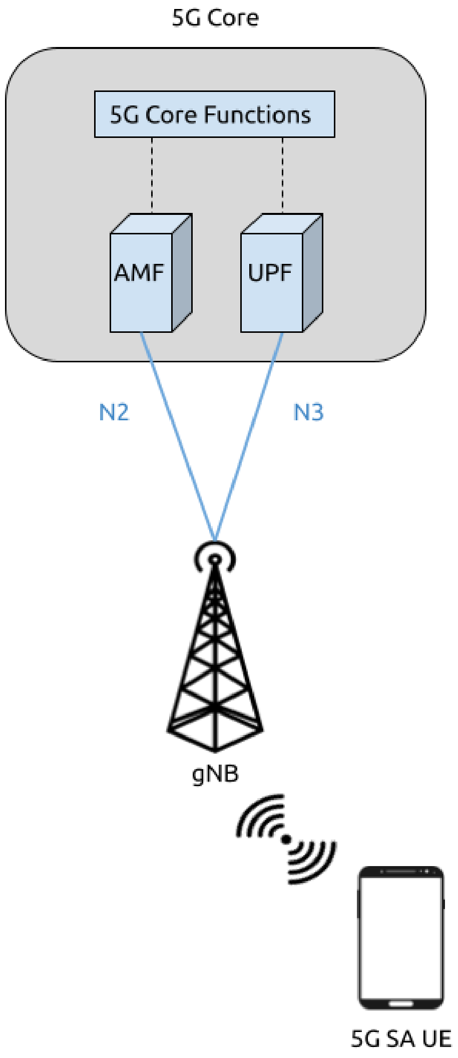
5G NSA with Opensource today (2)

- Needs at least a USRP X300 with 2 specific daughter boards



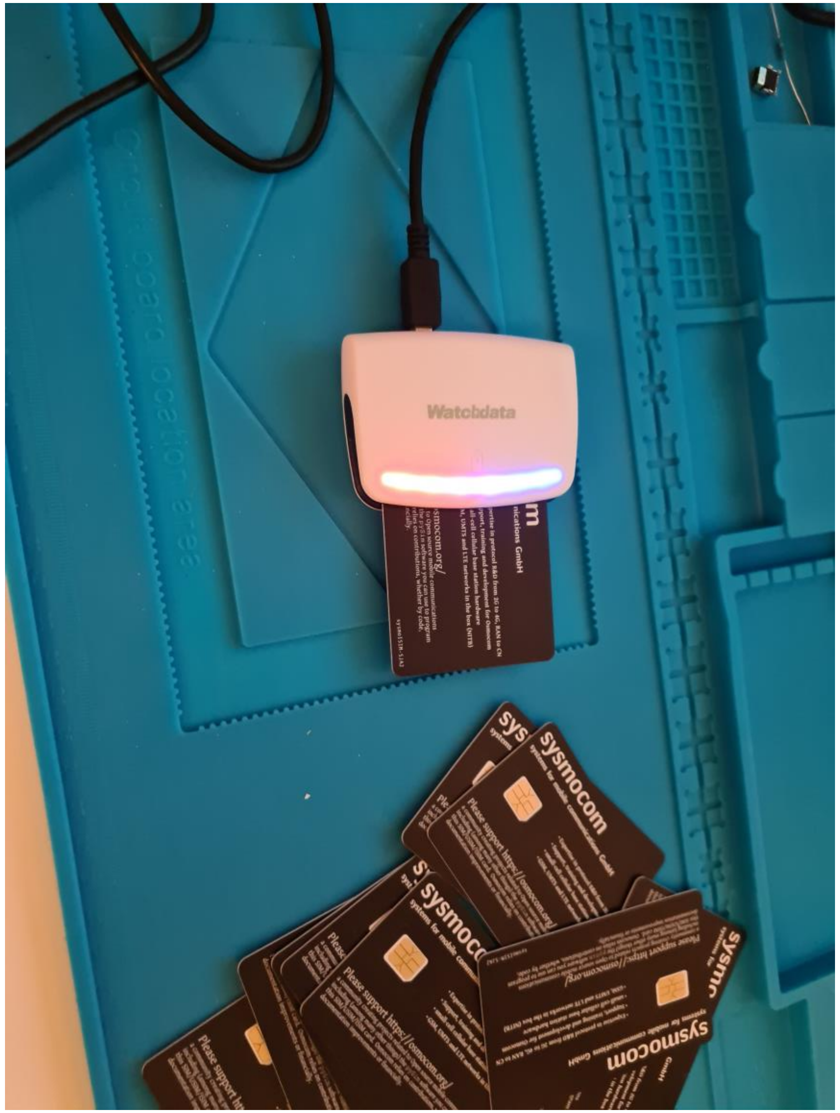
5G SA with Opensource today

- Same as for NSA:
  - OpenAirInterface5G
  - srsRAN → our favorite stack



Use of custom ISIM cards

- Goal: Complete mutual authentication
- At least a custom USIM, preferably ISIM:
  - USIM: Universal Subscriber Identity Module
  - ISIM: IP Multimedia Services Identity Module → SIP/IMS procedure
  - Generate Ki, OPC → provide on the gNB side
- Reference: http://shop.sysmocom.de/products/sysmoISIM-SJA2 (USIM version is interesting to disable USIM mode)
- Only a PC/SC reader is needed
- Cheaper version → AliExpress (with a *nice* software)



Use of custom ISIM cards (2)

```
┌─fluxius@trendmate ~/Projects/USIM/pysim ‹master›
└─$ sudo python3 pySim-prog.py -p0 -t sysmoISIM-SJA2 -i 901700000048419 -c 33 -x 001 -y 01 -s 8988211000000484199 -a 63682414
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
Using PC/SC reader interface
Ready for Programming: Insert card now (or CTRL-C to cancel)
Generated card parameters :
 > Name      : Magic
 > SMSP      : e1ffffffffffffffffffffffff058100335555ffffffffffff000000
 > ICCID     : 8988211000000484199
 > MCC/MNC   : 001/01
 > IMSI      : 901700000048419
 > Ki        : c6bd3a9172b188b8daf69643b65d686a
 > OPC       : 787f3270a48ef92b91146f5ef21681ea
 > ACC       : None
```

Note: Need for mutual authentication → need to know secrets → but an attacker can have access to SS7, DIAMETER, 5G infra (we will few aspects later…)

Use of custom ISIM cards (3)

- By default, the registration can fail → processing of SUCI
- Two ways to fix
  - Providing it to the SIM (ETSI TS 131 121)
  - Or deactivating it:

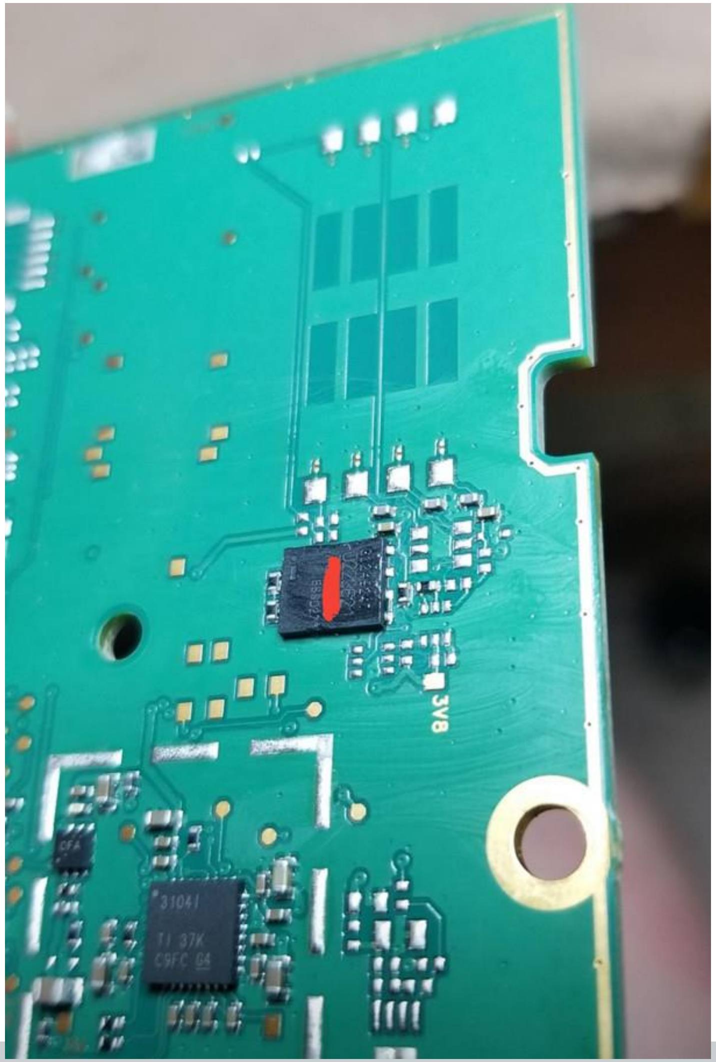
```
$ python3 pySim-shell.py -p0
[…]
pySIM-shell (MF/ADF.USIM/DF.5GS)> select EF.SUCI_Calc_Info
pySIM-shell (MF/ADF.USIM/DF.5GS/EF.SUCI_Calc_Info)> deactivate_file
```

Use of custom ISIM cards (4)

- Other points to look:
  - Use of same PLMN
  - Enabled 5G services
  - And some parameters relative to the network…

Other types of SIM

Soldered eUICC



https://f30.bimmerpost.com/forums/showthread.php?t=1642417

Soldered eUICC

- After desoldering, we can put our custom SIM card
- If IP is whitelisted, we can use the legitimate SIM card with a computer to forward accesses:

Soldered eUICC but extra SIM slot

- Embedded SIM needs to be chipped off before hooking them
- But 2nd slot exists in most cases + need to force the use with AT commands

Pin name	Pin no.	Electrical description	Description	Comment
(U)SIM1_PWR	36	PO	Power supply for (U)SIM1 card	1.8/3.0V voltage domain, all (U)SIM interfaces should be protected against ESD. If unused, please keep open
(U)SIM 1_DATA	34	DIO	(U)SIM1 card data, which has been pulled up to (U)SIM1_VDD via a 20KR resistor internally	
(U)SIM 1_CLK	32	DO	(U)SIM1 clock signal	
(U)SIM1_RESET	30	DO	(U)SIM1 Reset control	
(U)SIM 1_DET	66	DI	(U)SIM1 card detect, which has been pulled up to VDD_P3 via a 470KR resistor internally	
(U)SIM2_PWR	48	PO	Power supply for (U)SIM2 card	
(U)SIM2_DATA	42	DIO	(U)SIM2 card data, which has been pulled up to (U)SIM2_VDD via a 20KR resistor internally	
(U)SIM2_CLK	44	DO	(U)SIM2 clock signal	
(U)SIM2_RESET	46	DO	(U)SIM2 Reset control	
(U)SIM2_DET	40	DI	(U)SIM2 card detect, which has been pulled up to VDD_P3 via a 470KR resistor internally	

Surprises

5G SA support on phones

- Be careful of supported bands!
- A good reference: https://cacombos.com/
- But despite these notes → surprises!

LG Velvet 5G (T-Mobile USA) (LM-G900TM) 4G/5G Bands and Combos

Modem Specification	
Modem Model	Dimensity 1000C
Release Year	2020
LTE DL/UL Modulation	256QAM / 64QAM
LTE Bands	1, 2, 3, 4, 5, 8, 12, 13, 17, 20, 25, 26, 28, 39, 41, 66, 71
LTE 4x4 Bands	2, 4, 25, 41, 66
LTE Category (DL/UL)	18 / 13
LTE Max Speed (DL/UL)	1200 / 150 Mbps
NR NSA Bands	25, 41, 66, 71
NR SA Bands	71

Surprises

5G NR Bands

- To be respected in the configuration:

NR band n71 basic information

RAT	NR	**NR band**	n71
Name	600	**Duplex mode**	FDD
Frequency (UL)	663.00 MHz - 698.00 MHz	**Frequency (DL)**	617.00 MHz - 652.00 MHz
NR-ARFCN (UL)	132600 - 139600	**NR-ARFCN (DL)**	123400 - 130400
N_{ref} Step	20	**ΔF_{raster} (kHz)**	100
Band bandwidth (UL/DL)	35 MHz	**Duplex spacing**	46 MHz
Geography area	NAR		

Channel bandwidth

Channel Bandwidth(MHz)													
SCS(KHz)	5	10	15	20	25	30	40	50	60	70	80	90	100
15	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
30	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
60	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

NR band n71 spectrum/overlapped bands

https://itectec.com/band/nr-band-n71/

Surprises

But support is “fictive” sometimes…

- Not all phones support 5G SA
- Even if the commercial tells you so (e.g.: Exynos basebands, MKT Dimensity 1000C, HiSilicon, etc.)
- Some Huawei phones used to support it (e.g.: Mate X):
  - But now only in firmware for CN → need to downgrade them, maybe patch them
  - New upgrade in EU → restrict SA
  - Sometimes need to adjust PLMN to Chinese one + in ISIM (e.g.: MCC/MNC = “460 11”)

MCC	MNC	Brand	Operator
460	00	China Mobile	China Mobile
460	01	China Unicom	China Unicom
460	02	China Mobile	China Mobile
460	03	China Telecom	China Telecom
460	04	China Mobile	Global Star Satellite
460	05	China Telecom	China Telecom
460	06	China Unicom	China Unicom
460	07	China Mobile	China Mobile
460	08	China Mobile	China Mobile
460	09	China Unicom	China Unicom
460	11	China Telecom	China Telecom
460	20	China Tietong	China Tietong

Ref:
https://en.wikipedia.org/wiki/Mobile_network_codes_in_ITU_region_4xx_(Asia)

Case of failure (e.g.: LG Velvet 5G)

- Fail even respecting:
  - Duplex mode
  - NR-ARFCN
  - NR band
  - SCS
  - Etc.
- Possible to debug → LTE registration try

```
22:56:00.267 [NAS] UL 0003 EMM: Attach request
        Protocol discriminator = 0x7 (EPS Mobility Management)
        Security header = 0x1 (Integrity protected)
        Auth code = 0x9b6dbbb3
        Sequence number = 0x03
        Protocol discriminator = 0x7 (EPS Mobility Management)
        Security header = 0x0 (Plain NAS message, not security protected)
        Message type = 0x41 (Attach request)
        EPS attach type = 2 (combined EPS/IMSI attach)
        NAS key set identifier:
          TSC = 0
          NAS key set identifier = 0
        Old GUTI or IMSI:
          MCC = 001
          MNC = 01
          MME Group ID = 32769
          MME Code = 1
          M-TMSI = 0x84749e1a
        UE network capability:
          0xf0 (EEA0=1, 128-EEA1=1, 128-EEA2=1, 128-EEA3=1, EEA4=0, EEA5=0, EEA6=0, EEA7=0)
          0xf0 (EIA0=1, 128-EIA1=1, 128-EIA2=1, 128-EIA3=1, EIA4=0, EIA5=0, EIA6=0, EIA7=0)
          0xc0 (UEA0=1, UEA1=1, UEA2=0, UEA3=0, UEA4=0, UEA5=0, UEA6=0, UEA7=0)
          0xc0 (UCS2=1, UIA1=1, UIA2=0, UIA3=0, UIA4=0, UIA5=0, UIA6=0, UIA7=0)
          0x1d (ProSe-dd=0, ProSe=0, H.245-ASH=0, ACC-CSFB=1, LPP=1, LCS=1, 1xSRVCC=0, NF=1)
          0x00 (ePCO=0, HC-CP CIoT=0, ERw/oPDN=0, S1-U data=0, UP CIoT=0, CP CIoT=0, ProSe-relay=0, ProSe-dc=0)
          0x10 (15 bearers=0, SGC=0, N1mode=0, DCNR=1, CP backoff=0, RestrictEC=0, V2X PC5=0, multipleDRB=0)
```

Missing N1 mode

Case of success

- Expected even in LTE registration:

```
00:01:03.640 [NAS] UL 003d EMM: Attach request
        0000:  17 91 af 06 b7 11 07 41  12 0b f6 00 f1 10 80 01  .......A........
        0010:  01 e2 f8 e3 5f 07 f0 70  c0 40 18 80 b0 00 38 02  ...._..p.@....8.
        0020:  06 d0 11 d1 27 31 80 80  21 10 01 00 00 10 81 06  ....'1..!.......
        0030:  00 00 00 00 83 06 00 00  00 00 00 0d 00 00 0a 00  ................
        0040:  00 05 00 00 10 00 00 11  00 00 17 01 02 00 1a 01  ................
        0050:  01 00 23 00 00 24 00 5c  0a 00 31 04 65 e0 34 01  ..#..$.\..1.e.4.
        0060:  90 11 03 57 58 a6 5d 01  00 e0 c1 6f 04 f0 00 70  ...WX.]....o...p
        0070:  00 6d 01 00                                       .m..
        Protocol discriminator = 0x7 (EPS Mobility Management)
        Security header = 0x1 (Integrity protected)
        Auth code = 0x91af06b7
        Sequence number = 0x11
        Protocol discriminator = 0x7 (EPS Mobility Management)
        Security header = 0x0 (Plain NAS message, not security protected)
        Message type = 0x41 (Attach request)
        EPS attach type = 2 (combined EPS/IMSI attach)
        NAS key set identifier:
          TSC = 0
          NAS key set identifier = 1
        Old GUTI or IMSI:
          MCC = 001
          MNC = 01
          MME Group ID = 32769
          MME Code = 1
          M-TMSI = 0xe2f8e35f
        UE network capability:
          0xf0 (EEA0=1, 128-EEA1=1, 128-EEA2=1, 128-EEA3=1, EEA4=0, EEA5=0, EEA6=0, EEA7=0)
          0x70 (EIA0=0, 128-EIA1=1, 128-EIA2=1, 128-EIA3=1, EIA4=0, EIA5=0, EIA6=0, EIA7=0)
          0xc0 (UEA0=1, UEA1=1, UEA2=0, UEA3=0, UEA4=0, UEA5=0, UEA6=0, UEA7=0)
          0x40 (UCS2=0, UIA1=1, UIA2=0, UIA3=0, UIA4=0, UIA5=0, UIA6=0, UIA7=0)
          0x18 (ProSe-dd=0, ProSe=0, H.245-ASH=0, ACC-CSFB=1, LPP=1, LCS=0, 1xSRVCC=0, NF=0)
          0x80 (ePCO=1, HC-CP CIoT=0, ERw/oPDN=0, S1-U data=0, UP CIoT=0, CP CIoT=0, ProSe-relay=0, ProSe-dc=0)
          0xb0 (15 bearers=1, SGC=0, N1mode=1, DCNR=1, CP backoff=0, RestrictEC=0, V2X PC5=0, multipleDRB=0)
```

5G-NR SA registration: Amarisoft’s Web interface

Logs: 6217 | MME | Stats | IMS | ENB

UL/DL | Layer | UE ID | IMSI | Cell ID | Info | Level

Time origin: 00:00:00.000 Group UE ID: Clear

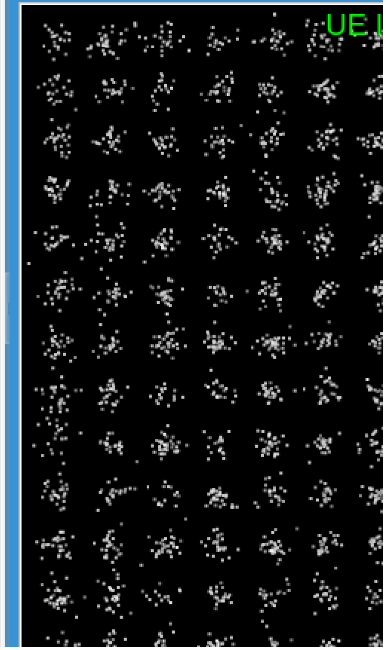
Search Analytics RB UE Caps

Time	Diff	RAN	CN	IMS	UE ID	IMSI	Cell	SFN	RNTI	Info	Message
-		PHY			3		1	4.399.19	0x4603	PUCCH	format=1 prb=50 prb2=0 symb=0:14 cs=0 occ=0 ack=1 snr=21.
-		RLC			3					DRB1	D/C=0 CPT=0 ACK_SN=39
-		MAC			3		1				LCID:4 len=3 PAD: len=219
-		PHY			3		1	4.400.6	0x4603	PDCCH	ss_id=2 cce_index=4 al=2 dci=1_1
-		PHY			3		1	4.400.6	0x4603	PDSCH	harq=0 prb=50 symb=1:13 k1=12 nl=2 CW0: tb_len=225 mod=8
23:53:01.691	+0.003	GTPU									127.0.1.100:2152 G-PDU TEID=0x32149638 QFI=1 SDU_le
23:53:01.692	+0.001	PDCP			3					DRB1	D/C=1 SN=41
-		RLC			3					DRB1	D/C=1 P=1 SI=00 SN=41
-		MAC			3		1				LCID:4 len=59 PAD: len=163
-		PHY			3		1	4.400.14	0x4603	PDCCH	ss_id=2 cce_index=4 al=2 dci=1_1
-		PHY			3			0.14	0x4603	PDSCH	harq=1 prb=50 symb=1:13 k1=5 nl=2 CW0: tb_len=225 mod=8
-		GTPU									127.0.1.100:2152 G-PDU TEID=0x32149638 QFI=1 SDU_le
-		PHY			3		1	4.400.8	0x4603	PUCCH	format=1 prb=50 prb2=0 symb=0:14 cs=8 occ=2 sr=1 snr=21.5
-		PDCP			3					DRB1	D/C=1 SN=42
-		RLC			3					DRB1	D/C=1 P=1 SI=00 SN=42
-		MAC			3		1				LCID:4 len=59 PAD: len=163
-		PHY			3		1	4.400.15	0x4603	PDCCH	ss_id=2 cce_index=2 al=2 dci=0_1 k2=4
-		PHY			3		1	4.400.15	0x4603	PDCCH	ss_id=2 cce_index=6 al=2 dci=1_1
-		PHY			3		1	4.400.15	0x4603	PDSCH	harq=2 prb=50 symb=1:13 k1=4 nl=2 CW0: tb_len=225 mod=8
23:53:01.693	+0.001	PHY			3		1	4.400.9	0x4603	PUCCH	format=2 prb=49 prb2=1 symb=2:2 csi=1011111 epre=-104.4
23:53:01.697	+0.004	PHY			3		1	4.400.18	0x4603	PUCCH	format=1 prb=50 prb2=0 symb=0:14 cs=0 occ=0 ack=1 snr=21.

Frame: 3.936.9
Cell: #1

From: ENB
Info: 192.168.1.116:9001, v2021-09-18
Time: 23:52:49.213
Index: 5031
Message: format=2 prb=49 prb2=1 symb=
epre=-106.5

ENB constellations



And profit!

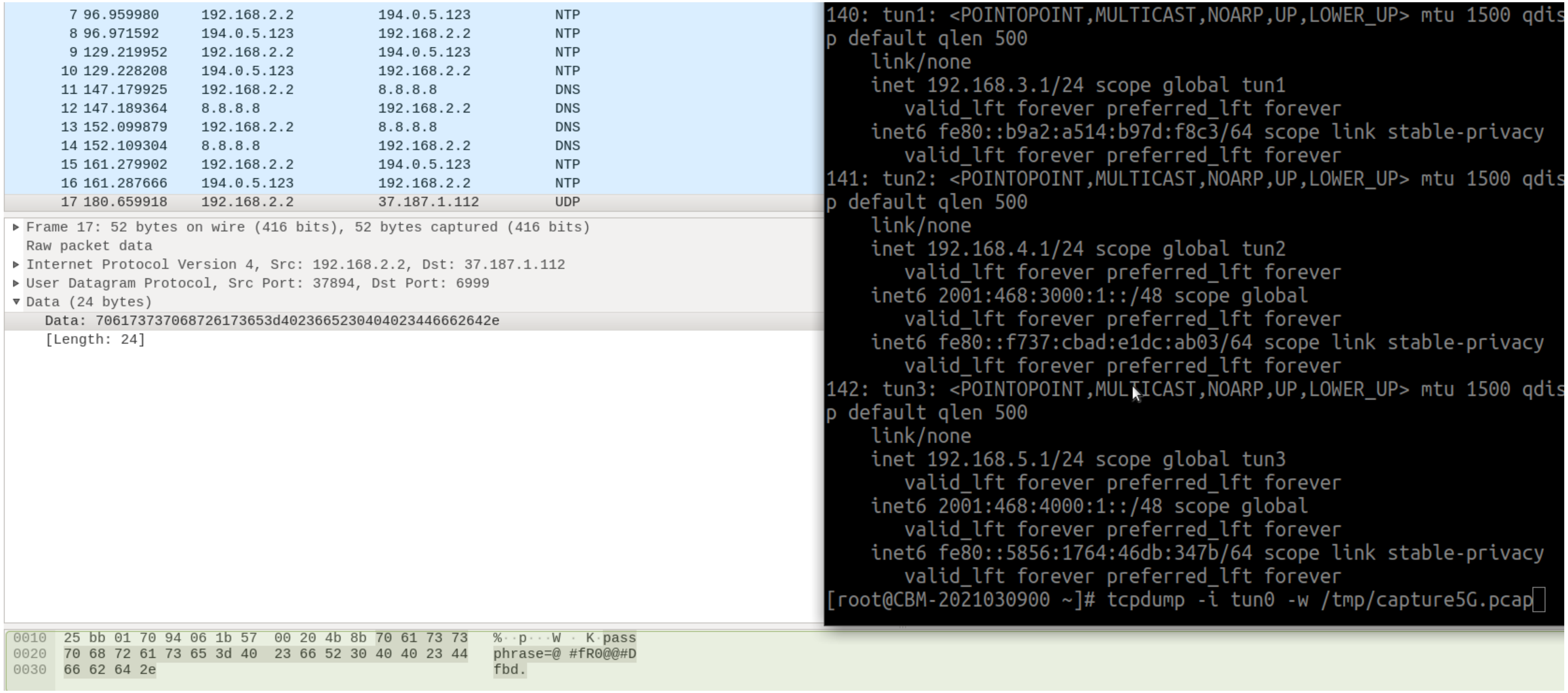
Assets to look

- Clear-text communications
- Checked certificates
- Data confidentiality
- Use leaked endpoints to pivot
- Pass the NAT via authenticated STUN
- Look at other secrets to pivot (e.g.: 802.1x, etc.)

And profit!

Extracting secret: monitoring net interfaces

- Looking at right interface, we can then smoothly



And profit!

Leaked APN and PPP conf

```
6796 10.672777362   127.0.0.1            127.0.0.1            ICMP        180 Destination unre
▶ Packet Data Protocol Address - Requested PDP address
▼ Access Point Name
    Element ID: 0x28
    Length: 12
    APN: i[redacted]r
▼ Protocol Configuration Options
    Element ID: 0x27
    Length: 59
    [Link direction: MS to network (0)]
    1... .... = Extension: True
    .... .000 = Configuration Protocol: PPP for use with IP PDP type or IP PDN type (0)
  ▼ Protocol or Container ID: Password Authentication Protocol (0xc023)
      Length: 0x24 (36)
    ▼ PPP Password Authentication Protocol
        Code: Authenticate-Request (1)
        Identifier: 1
        Length: 36
      ▼ Data
          Peer-ID-Length: 20
          Peer-ID: sd[redacted].fg
          Password-Length: 10
          Password: v[redacted]r
  ▼ Protocol or Container ID: Internet Protocol Control Protocol (0x8021)
      Length: 0x10 (16)
```

Limits

- Sometimes, we can be limited → data is encrypted, etc.
- We can attack the microcontroller, or host using the mobile modules
- But sometimes, manufacturers prefer using the mobile module SDK, rather than using limited microcontroller libraries…
  - So, there is also something to look on this mobile module ;)

Attacking the mobile module

The mobile module

- Can be composed of an application processor running Linux and the baseband processor
- To access to it we can try:
  - Serial port interfaces
  - Using I2C, SPI, JTAG
  - Or to interface with AT and DIAG
- Usually AT and DIAG access are accessible

Attacking via AT commands

- Inspired by Harald Welte at 33c3 in 2016:

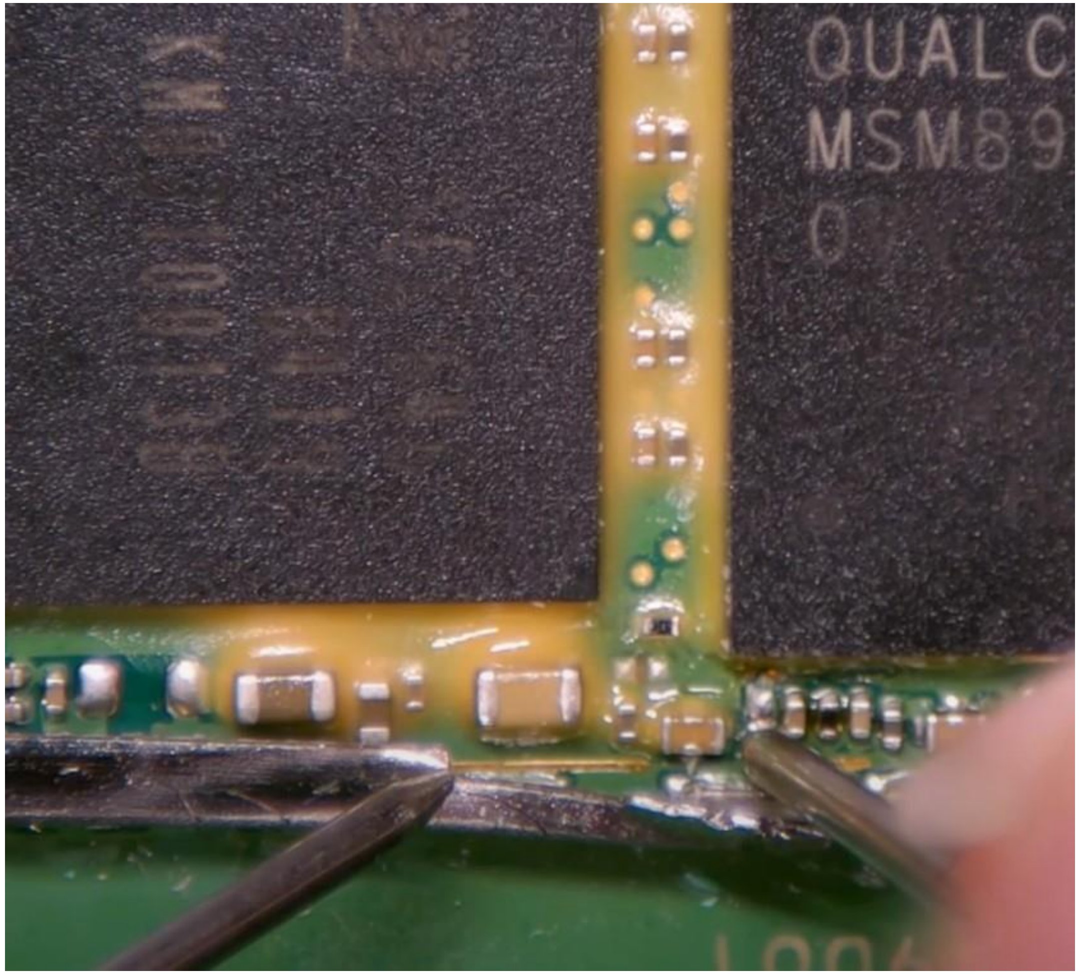
```
# echo -e 'AT+QLINUXCMD="/sbin/getty -L ttyGS0 115200 console"\r\n' >
/dev/ttyUSB2
# microcom /dev/ttyUSB1

OpenEmbedded Linux 9615-cdp ttyGS0

msm 20160923 9615-cdp ttyGS0

9615-cdp login: root
Password: oelinux123
root@9615-cdp:~#
```

- Some kind of accesses we found on SIM8200A (presented at NoHat 2021):

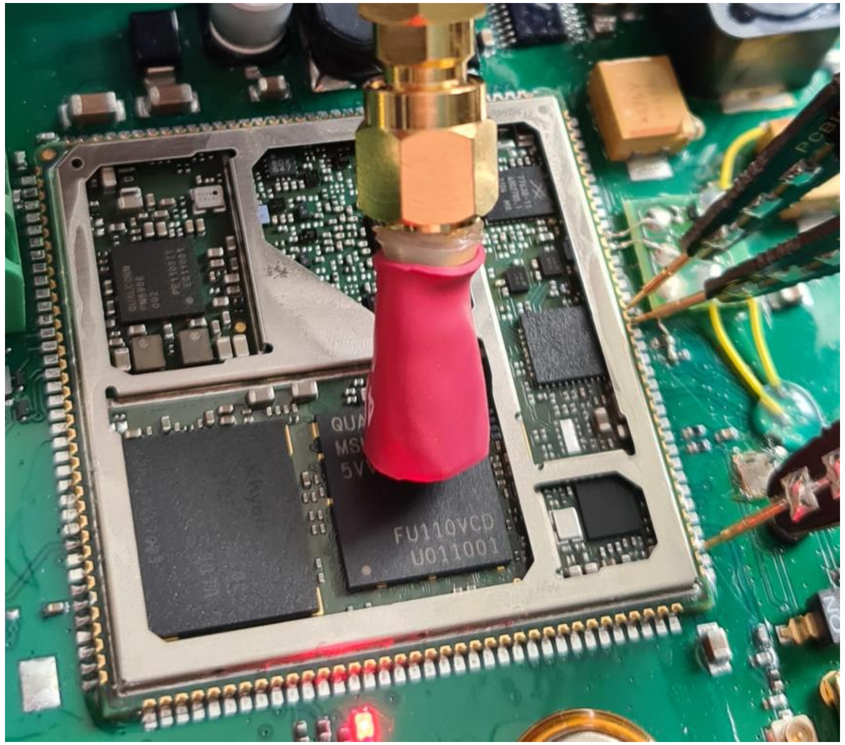
```
AT+CUSBCFG=usbadb,1
```

Hardware: shortcutting EDL PIN

- Exposed PIN allows us to pass in **adb** or **fastboot** → Yes! Android is everywhere!
- If this PIN is not directly exposed → search or 1 millivolt PIN and short cutting it with a 1.8-volt tension



Or with EMFI attacks



- Triggering **fastboot** mode

```
Android Bootloader - UART_DM Initialized!!!
[0] welcome to lk
[...]
[640] ERROR: Cannot read boot image
[640] ERROR: Could not do normal boot. Reverting to fastboot mode.
[650] battery is not present
[650] fastboot_init()
```

- Or **EDL** mode:

```
[…]
[310] undefined abort, halting
[310] r0  0x00000018 r1  0x83362e02 r2  0x00000000 r3  0x00000003
[...]
0x8f6c15e8: 8f6a6734 8f6a6734 00000000 8f6a6734 |.gj..gj......gj.|
[310] HALT: reboot into dload mode.�
```

To finally get the FS → and extract secrets!

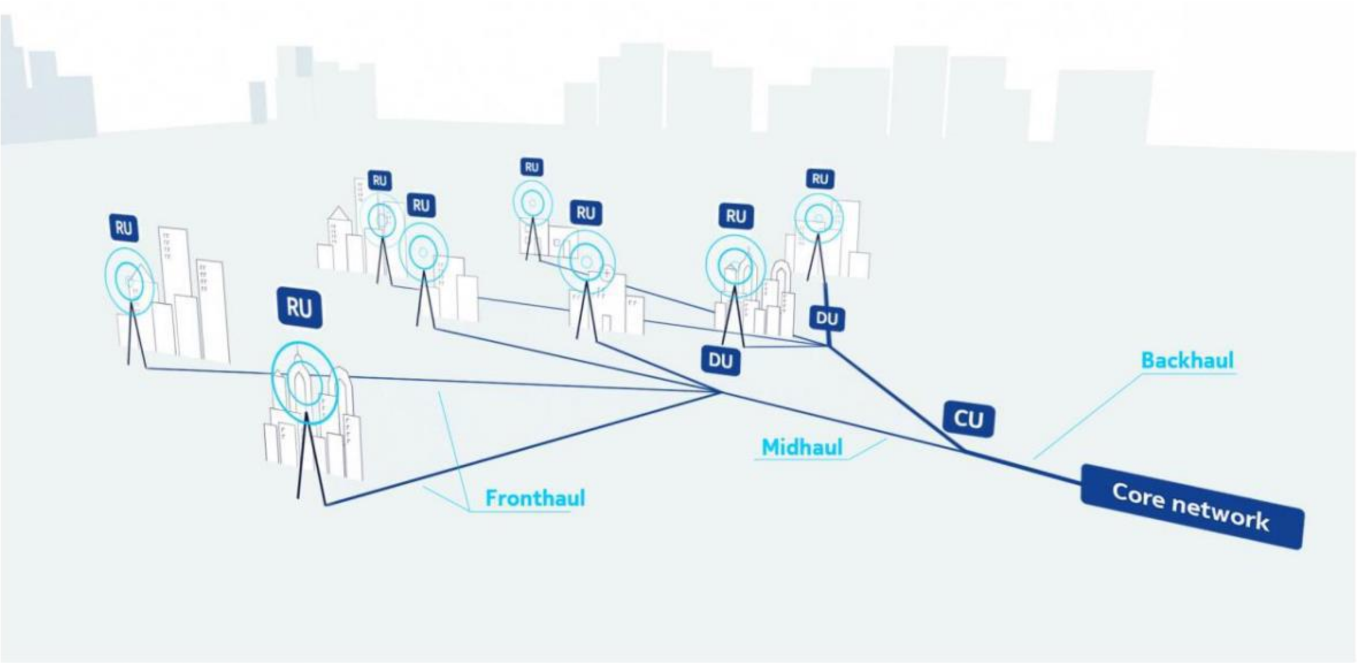
```
boot:                    Offset 0x0000000083****, Length 0x000000000*0000, Flags
0x1000000000000000, UUID b512967*********010e94, Type 0x20117f86, Active False
system:                  Offset 0x000000000*****, Length 0x000000003****0000, Flags
0x1000000000000000, UUID d51623ef-*********7f1e, Type 0x97d7b011, Active False
persist:                 Offset 0x00000000***c000, Length 0x0000000002000000, Flags
0x1000000000000000, UUID debadb79********65f3b, Type 0x6c95e238, Active False
cache:                   Offset 0x0000000**c000, Length 0x00000000006e00000, Flags
0x1000000000000000, UUID 98e95bc2*********cd6c, Type 0x5594c694, Active False
recovery:                Offset 0x00000000***c000, Length 0x0000000002000000, Flags
0x1000000000000000, UUID 2798********b03a9e7, Type 0x9d72d4e4, Active False
[…]
```

Thanks to https://github.com/bkerler/edl ;) → avoid all Sahara / Firehose setup

Future targets!

What’s a RAN

- Radio Access Network (RAN)
  - Link between mobile core network and the user equipment
- Exists since 2G:
  - GSM → GRAN
  - 4G → E-UTRAN (Evolved Universal Terrestrial Radio Access Network)
  - 5G → NG-RAN (Next Generation Radio Access Network)



Simplified representation of an Open RAN architecture (source: Nokia)

Opening the RAN

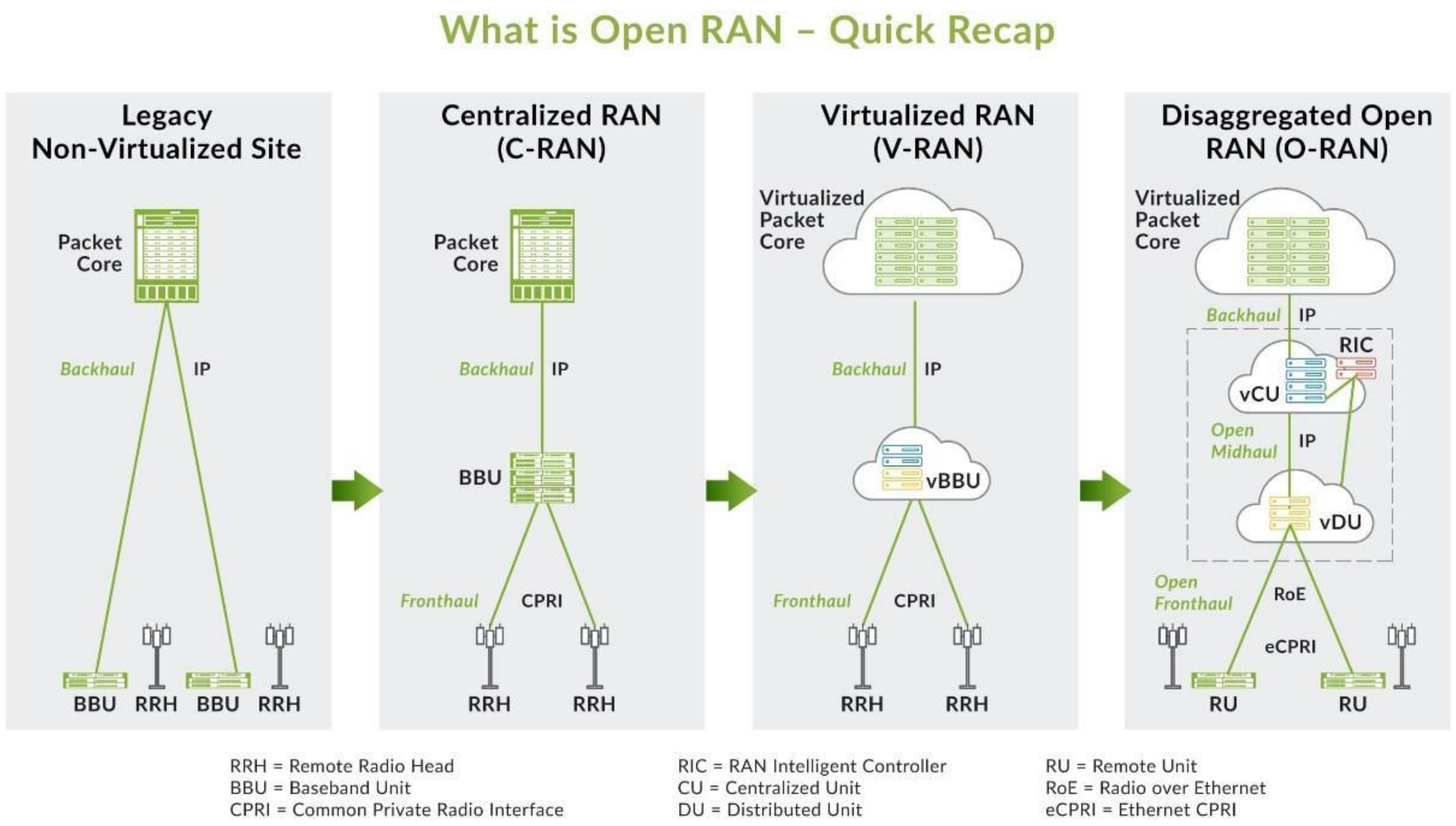
- Classic issues with current RANs:
  - Inflexibility
  - Compatibility issues
  - High costs due to small number of competitors
- Main vendors:
  - Ericsson
  - Nokia
  - Huawei (banned in some countries)

Opening the RAN (2)

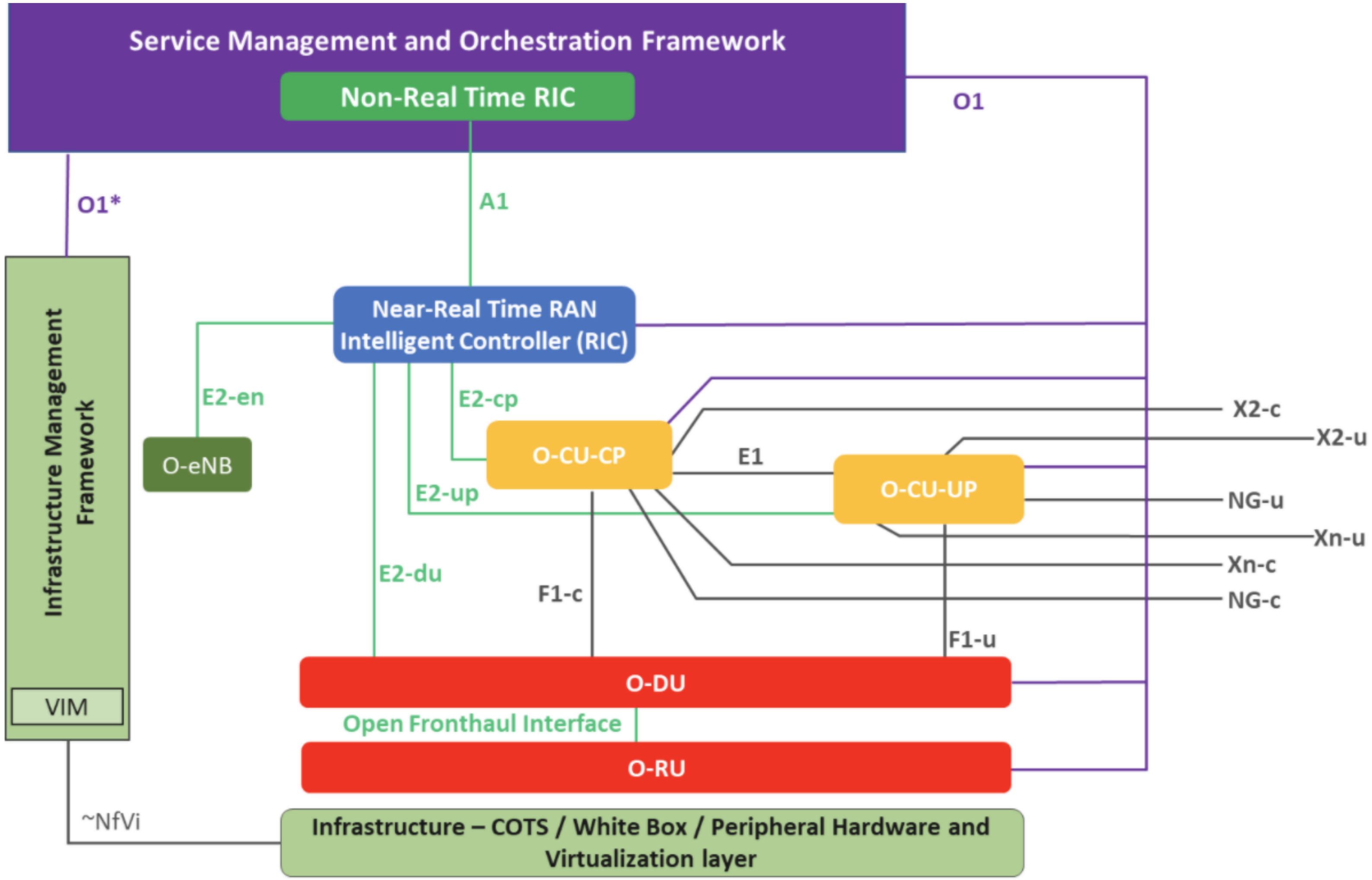
- Advantages of Opening RAN:
  - Reduce costs
  - Interoperability
  - And also a good opportunity for the US to enter the market finally! ;)
- O-RAN architecture → enhancement to existing 3GPP standards

Evolution

- Complexity of protocol stacks increases → sub-6GHz bands, mmWave, over 100 GHz + IoT systems, etc.
- Multiple solutions were studied before O-RAN:

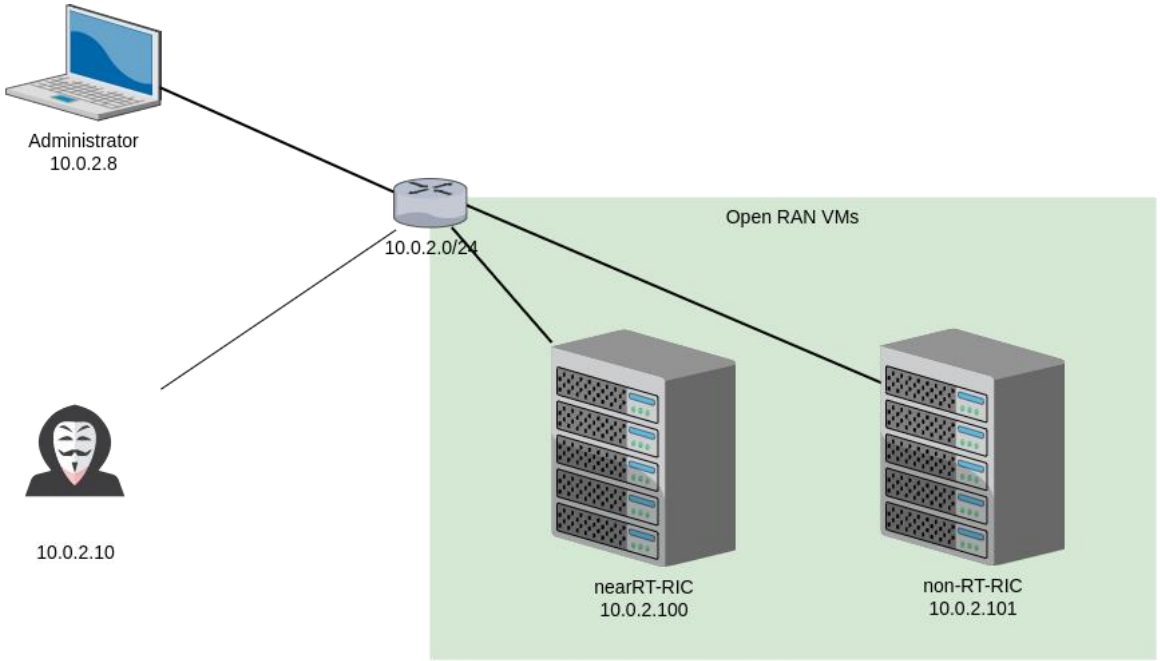


Architecture



Exposure

- Each RT-RIC and non-RT-RIC → run in containers insides Kubernetes pods
- Without proper isolation → exposition to attacks
- This was discussed by Karsten Nohl at MCH2022 → but not very clear :/
- So let's create a scenario:

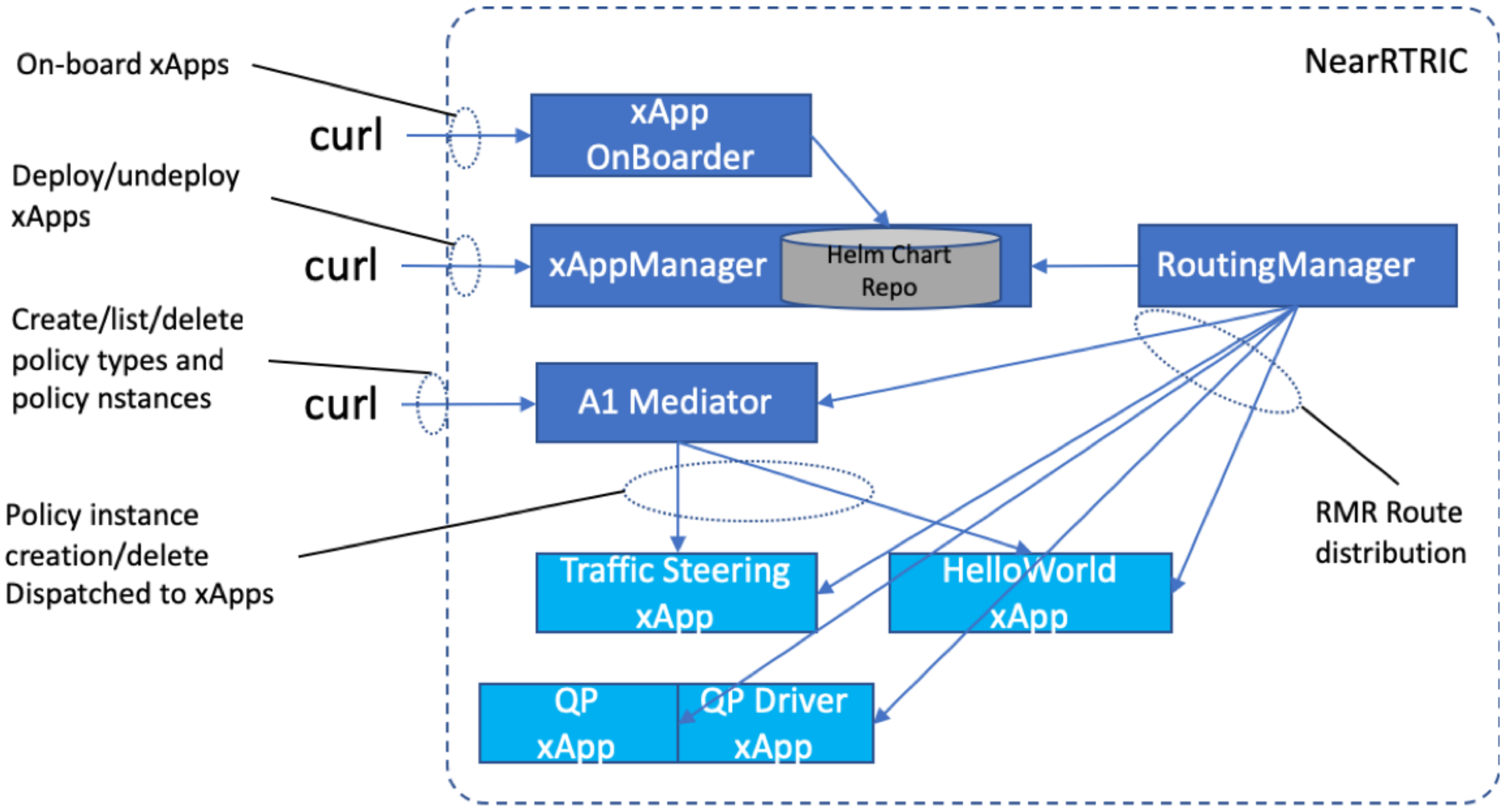


Kubernetes Interfaces

- Kubernetes interfaces can be scanned with **Kube-hunter**:

```
$ python3 kube-hunter.py
Choose one of the options below:
1. Remote scanning      (scans one or more specific IPs or DNS names)
2. Interface scanning   (scans subnets on all local network interfaces)
3. IP range scanning    (scans a given IP range)
Your choice: 3
CIDR separated by a ',' (example -
192.168.0.0/16,!192.168.0.8/32,!192.168.1.0/24): 10.0.2.10/24
2022-10-11 13:33:34,153 INFO kube_hunter.modules.report.collector Started
hunting
2022-10-11 13:33:34,154 INFO kube_hunter.modules.report.collector Discovering
Open Kubernetes Services
2022-10-11 13:33:34,259 INFO kube_hunter.modules.report.collector Found open
service "Etcd" at 10.0.2.100:2379
2022-10-11 13:33:34,266 INFO kube_hunter.modules.report.collector Found open
service "Kubelet API" at 10.0.2.100:10250
2022-10-11 13:33:34,291 INFO kube_hunter.modules.report.collector Found open
service "API Server" at 10.0.2.100:6443
2022-10-11 13:33:34,304 INFO kube_hunter.modules.report.collector Found
vulnerability "K8s Version Disclosure" in 10.0.2.100:6443
```

A good candidate



xApp Flows

Backdooring the chart

- Looking at exposed **ChartMuseum** of the **Appmanager** interface:

```
# nmap -A 10.0.2.100
[…]
PORT     STATE SERVICE       VERSION
[…]
8090/tcp open  opsmessaging?
| fingerprint-strings:
|   GenericLines, Help, RTSPRequest, SSLSessionReq, TerminalServerCookie:
|     HTTP/1.1 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     Connection: close
|     Request
|   GetRequest:
|     HTTP/1.0 200 OK
|     Content-Type: text/html
|     X-Request-Id: cf122339-6f7f-4196-aa83-a1ab09da314a
|     Date: Wed, 12 Oct 2022 09:20:43 GMT
|     Content-Length: 547
|     <!DOCTYPE html>
|     <html>
|     <head>
|     <title>Welcome to ChartMuseum!</title>
```

Backdooring the chart (2)

- We can use the O-RAN tools **dms_cli** to onboard new xApps:

```
export CHART_REPO_URL=http://10.0.2.100:8090
$ dms_cli health
True
```

- We can implement an xApp in different languages:
  - Python: https://github.com/o-ran-sc/ric-app-hw-python
  - Rust: https://github.com/o-ran-sc/ric-app-hw-rust
  - Go: https://github.com/o-ran-sc/ric-app-hw-go
  - C++: https://github.com/o-ran-sc/ric-app-hw

Conclusion

To conclude

- Only eMBB (enhanced Mobile Broadcast) have been tested yet
- But mMTC (massive Machine Type Communication) and URLLC (Ultra Reliable Low Latency Communication) may be strictly used by some IoT devices, like with NB-IoT and Cat-M1 in 4G → more challenges with Open-source setup
- 5G modules are not used a lot yet, only development prototypes have been tested with our clients
  - But may change in the future
- We are also C-V2X but still postponed → available devices in production → contact us if you've any to test! ;)
- Baseband attacks would also be still a thing to go further:
  - VoLTE as for VoNR and other complex protocols implemented in BB would be a good candidate → https://googleprojectzero.blogspot.com/2023/03/multiple-internet-to-baseband-remote-rce.html
  - Uncyphered ARM basebands are generally good candidates

penthertz

Thank You

Please contact us:

contact@penthertz.com

+33 1 73 13 82 77

penthertz.com

Watch us on YouTube