4G vs 5G Final Project Documentation

Setup

1. Copy files up to /opt/srsRAN_Project/docker:
    a. Dockerfile.4g
    b. Dockerfile.4g-bypass-s1
    c. Dockerfile.relay
    d. Dockerfile.fake-mme
    e. fake_mme_stub.py
    f. docker-compose-4g-5g.yaml
    g. 5g_ue.conf
    h. ngenb.yaml
    i. s1ap.cc.patched
    j. s1ap.h.patched
    k. rcc.cc.patched
    l. enb_s1ap_interfaces.h.patched
    m. test_helpers.h.patched
    n. Dockerfile.5g-gnb
    o. Dockerfile.5g-gnb-fail-auth
    p. ngap_impl.cpp.patched
2. Modify the docker/open5gs/Dockerfile to add at the bottom above ENTRYPOINT:

```
     RUN DEBIAN_FRONTEND=noninteractive apt update && apt-get
install -y --no-install-recommends netcat-openbsd \
        Iproute2 net-tools tcpdump iputils-ping gettext socat less
&& \
        apt-get autoremove && apt-get clean && rm -rf
/var/lib/apt/lists/*
```

3. Run the following build command to prepare all the containers:

```
sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-
5g.yaml --profile 4g-all build

sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-
5g.yaml --profile 5g-all build
```

4. Open three separate terminals

Commands

4G

Normal

Stage	Command
Start normal profile from terminal 1	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-normal up --build`
Tcpdump inside enb terminal 2	`sudo docker exec -it srsran_4g_enb-normal tcpdump -i eth0 -U -w /tmp/4g-normal-sctp-capture.pcap '(port 36412 or port 36422 or port 2152)'`
Start ue in terminal 3 detached	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-normal up 4g-ue-normal --build -d` Wait for the successful network attach and for the notification that it was disconnected. (it'll be in terminal 1 window)
Send Ping to EPC	`sudo docker exec -it srsran_4g_ue-normal ping 8.8.8.8`
Shut down UE	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-normal down 4g-ue-normal`
Stop Capture in terminal 2	`CTRL+C to stop the enb capture`
Copy capture	`sudo docker cp srsran_4g_enb-normal:/tmp/4g-normal-sctp-capture.pcap ./4g-normal-sctp-capture.pcap`
Bring down all baseline components from terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-normal down`

Download pcap file locally and open with wireshark to evaluate:

```
noglob scp -i ~/.ssh/SSH_PRIV USER@PC###.emulab.net:*.pcap .
```

Analysis

baseline

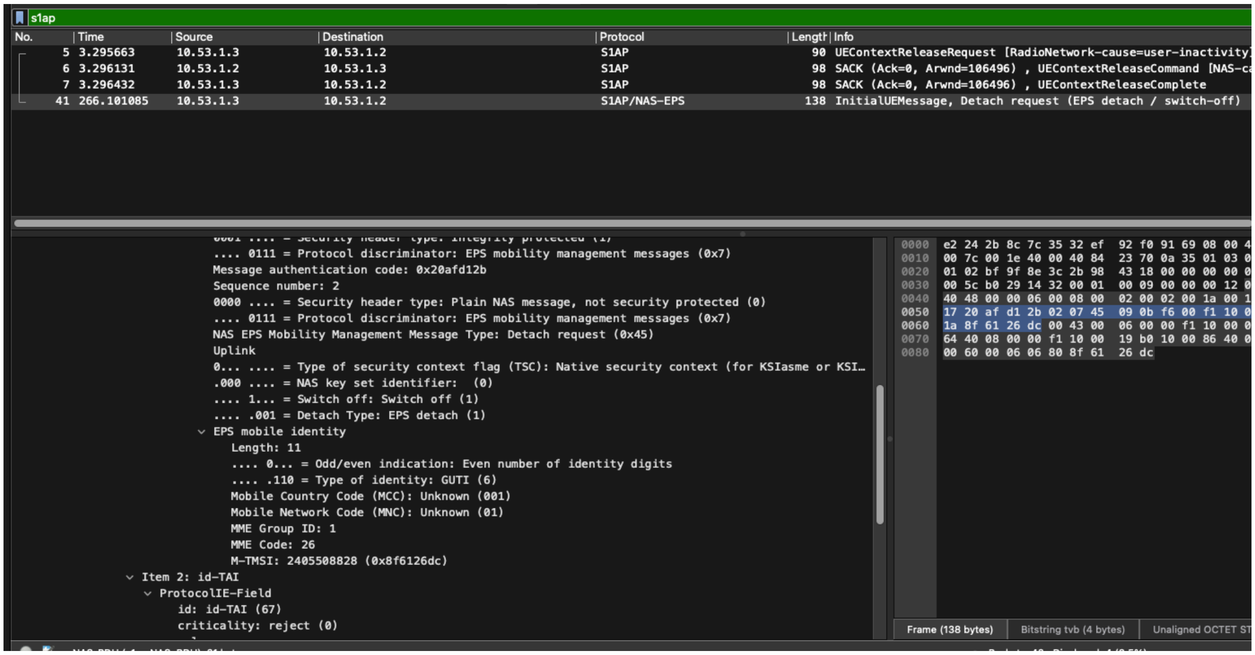
```
wireshark normal-sctp-capture.pcap
```

Run the following filters to demonstrate:

Goal	Wireshark Filter	Purpose
See all S1AP signaling	`s1ap`	✅ Show Attach Request, Initial Context Setup
Focus only on NAS messages	`nas-eps`	✅ Show Attach Request, Security Mode Command/Complete

Goal	Wireshark Filter	Purpose
Highlight security activation	`nas-eps.procedureTransactionIdentity == 1 && nas-eps.securityheader == 0x0`	✅ See initial Attach (no security yet)
Show Attach Complete with encryption active	`nas-eps.message_type == 0x41`	✅ Attach Complete, after encryption is turned on
Show GTP bearer setup	`gtp`	✅ GTP-U tunnel establishment (data plane)

Unencrypted show of IMEI



```
No.   Time        Source      Destination  Protocol      Length Info
   5 3.295663    10.53.1.3   10.53.1.2    S1AP              90 UEContextReleaseRequest [RadioNetwork-cause=user-inactivity]
   6 3.296131    10.53.1.2   10.53.1.3    S1AP              98 SACK (Ack=0, Arwnd=106496) , UEContextReleaseCommand [NAS-ca
   7 3.296432    10.53.1.3   10.53.1.2    S1AP              98 SACK (Ack=0, Arwnd=106496) , UEContextReleaseComplete
  41 266.101085  10.53.1.3   10.53.1.2    S1AP/NAS-EPS     138 InitialUEMessage, Detach request (EPS detach / switch-off)
```

```
            0001 .... = Security header type: Integrity protected (1)
            .... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
            Message authentication code: 0x20afd12b
            Sequence number: 2
            0000 .... = Security header type: Plain NAS message, not security protected (0)
            .... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
            NAS EPS Mobility Management Message Type: Detach request (0x45)
            Uplink
            0... .... = Type of security context flag (TSC): Native security context (for KSIasme or KSI…
            .000 .... = NAS key set identifier:  (0)
            .... 1... = Switch off: Switch off (1)
            .... .001 = Detach Type: EPS detach (1)
          ⌄ EPS mobile identity
              Length: 11
              .... 0... = Odd/even indication: Even number of identity digits
              .... .110 = Type of identity: GUTI (6)
              Mobile Country Code (MCC): Unknown (001)
              Mobile Network Code (MNC): Unknown (01)
              MME Group ID: 1
              MME Code: 26
              M-TMSI: 2405508828 (0x8f6126dc)
  ⌄ Item 2: id-TAI
    ⌄ ProtocolIE-Field
        id: id-TAI (67)
        criticality: reject (0)
```

```
0000  e2 24 2b 8c 7c 35 32 ef  92 f0 91 69 08 00 4
0010  00 7c 00 1e 40 00 40 84  23 70 0a 35 01 03 0
0020  01 02 bf 9f 8e 3c 2b 98  43 18 00 00 00 00 0
0030  00 5c b0 29 14 32 00 01  00 09 00 00 00 12 0
0040  40 48 00 00 06 00 08 00  02 00 02 00 1a 00 1
0050  17 20 af d1 2b 02 07 45  09 0b f6 00 f1 10 0
0060  1a 8f 61 26 dc 00 43 00  06 00 00 f1 10 00 0
0070  64 40 08 00 00 f1 10 00  19 b0 10 00 86 40 0
0080  00 60 00 06 06 80 8f 61  26 dc
```

Man-In-The-Middle/Weak Encryption Test

Stage	Command

Bring up real AMF, MITM relay	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-mitm up 4g-epc-normal`
	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-mitm up 4g-mitm-relay`

Tcpdump inside relay from terminal 2	`sudo docker exec -it srsran_4g_mitm-relay tcpdump -i any -U -vv -w /tmp/4g-mitm-sctp-capture.pcap port 36412 or port 2152`
Startup eNB from terminal 3	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm up 4g-enb-mitm`

Stage	**Command**
Startup MITM ue from terminal 4	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm up 4g-ue-mitm`
Run ping	Wait for the notification in of RCC release `sudo docker exec srsran_4g_ue-mitm sh -c 'gw=$(ip route | awk "/default/ {print \$3}"); echo "Pinging gateway: $gw"; ping -c 4 $gw; echo ""; echo "Pinging 8.8.8.8"; ping -c 4 8.8.8.8'`
Stop ue from terminal 3	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm down 4g-ue-mitm`
Copy rogue capture	CTRL+C to stop the capture `sudo docker cp srsran_4g_mitm-relay:/tmp/4g-mitm-sctp-capture.pcap ./4g-mitm-sctp-capture.pcap`
Bring down all baseline components from terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-mitm down`

Download pcap file locally and open with wireshark to evaluate:

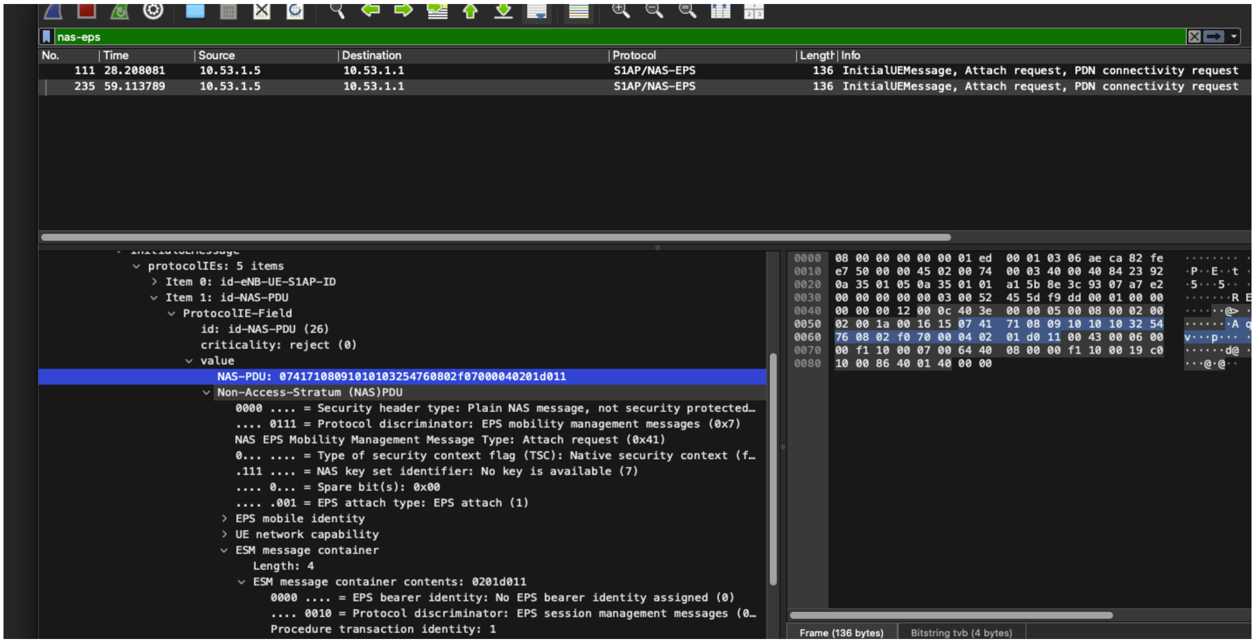
```
noglob scp -i ~/.ssh/SSH_PRIV USER@PC###.emulab.net:*.pcap
```

Analysis

```
wireshark mitm-sctp-capture.pcap
```

Run the following filters to demonstrate:

Goal	**Wireshark Filter**	**Purpose**
See all S1AP signaling	`s1ap`	✅ Attach attempts captured at mitm
Focus only on NAS messages	`nas-eps`	✅ Look at plain NAS Attach Request
Catch IMSI exposure	`nas-eps.esm.imsi` or manually inspect NAS Attach Request (no encryption)	✅ See IMSI in cleartext (payload visible)
Check absence of security activation	`nas-eps.securityheader == 0x0` (no secure NAS header)	✅ No encryption applied yet
Failed bearer setup (missing GTP)	`gtp`	❌ No successful bearer setups (almost no GTP packets)



```
No.   Time       Source      Destination   Protocol       Length Info
  111 28.208081  10.53.1.5   10.53.1.1     S1AP/NAS-EPS   136 InitialUEMessage, Attach request, PDN connectivity request
  235 59.113789  10.53.1.5   10.53.1.1     S1AP/NAS-EPS   136 InitialUEMessage, Attach request, PDN connectivity request
```

```
      protocolIEs: 5 items
        > Item 0: id-eNB-UE-S1AP-ID
        v Item 1: id-NAS-PDU
          v ProtocolIE-Field
              id: id-NAS-PDU (26)
              criticality: reject (0)
            v value
                NAS-PDU: 07417108091010103254760802f07000040201d011
              v Non-Access-Stratum (NAS)PDU
                  0000 .... = Security header type: Plain NAS message, not security protected…
                  .... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
                  NAS EPS Mobility Management Message Type: Attach request (0x41)
                  0... .... = Type of security context flag (TSC): Native security context (f…
                  .111 .... = NAS key set identifier: No key is available (7)
                  .... 0... = Spare bit(s): 0x00
                  .... .001 = EPS attach type: EPS attach (1)
                > EPS mobile identity
                > UE network capability
                v ESM message container
                    Length: 4
                  v ESM message container contents: 0201d011
                      0000 .... = EPS bearer identity: No EPS bearer identity assigned (0)
                      .... 0010 = Protocol discriminator: EPS session management messages (0…
                      Procedure transaction identity: 1
```

```
0000  08 00 00 00 00 00 01 ed  00 01 03 06 ae ca 82 fe   ········ ·
0010  e7 50 00 00 45 02 00 74  00 03 40 00 40 84 23 92   ·P··E··t ·
0020  0a 35 01 05 0a 35 01 01  a1 5b 8e 3c 93 07 a7 e2   ·5···5·· ·
0030  00 00 00 00 00 03 00 52  45 5d f9 dd 00 01 00 00   ·······R E
0040  00 00 00 12 00 0c 40 3e  00 00 05 00 08 00 02 00   ······@> ·
0050  02 00 1a 00 16 15 07 41  71 08 09 10 10 10 32 54   ·······A q
0060  76 08 02 f0 70 00 04 02  01 d0 11 00 43 00 06 00   v···p··· ·
0070  00 f1 10 00 07 00 64 40  08 00 00 f1 10 00 19 c0   ······d@ ·
0080  10 00 86 40 01 40 00 00                            ···@·@··
```

View Exposed IMSI Test

Stage	Command
Bring up fake MME, MITM relay, and base station	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-mitm up --build`
View Exposed IMSI	`sudo docker logs srsran_4g_ue-mitm | grep -i imsi`

Simulate Signaling Storm Test/DOS Attack Test

Stage	Command

Bring up fake MME, MITM relay, and base station and scale the UEs to 10	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-storm up 4g-enb-rogue`
	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-storm up --build --scale 4g-ue-storm=10`

Network Load Demonstration	`sudo docker logs srsran_4g_enb-rogue | grep -i rrc`
Check out repeated Connections	`Look at log of UE to see repeated connection attempts`

Rogue Base Station

Stage	Command
Bring up patched ENB	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-rogue up 4g-enb-rogue` Wait for the notification that it has failed a connection
Bring up mitm relay and fake mme in terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-rogue up -d` Wait for the notification that MME has received an S1SetupResponse
Tcpdump rogue enb from terminal 2	`sudo docker exec -it srsran_4g_enb-rogue tcpdump -i eth0 -s 0 -vv -U -w /tmp/4g-rogue-sctp-capture.pcap '(port 36412 or port 36422 or port 2152)'`
Startup ue from terminal 3	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-rogue up 4g-ue-rogue` Let it fail to connect a few times
Stop ue from terminal 3	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-rogue down 4g-ue-rogue`
Copy rogue capture	CTRL+C to stop the capture `sudo docker cp srsran_4g_enb-rogue:/tmp/4g-rogue-sctp-capture.pcap ./4g-rogue-sctp-capture.pcap`
Bring down all rogue components from terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 4g-rogue down`

Download pcap file locally and open with wireshark to evaluate:

```
noglob scp -i ~/.ssh/SSH_PRIV USER@PC###.emulab.net:*.pcap
```

Analysis

```
wireshark rogue-sctp-capture.pcap
```

Run the following filters to demonstrate:

Goal	Wireshark Filter	Purpose
See all S1AP signaling	`s1ap`	✅ Attach attempts captured at mitm
Focus only on NAS messages	`nas-eps`	✅ Look at plain NAS Attach Request
Catch IMSI exposure	`nas-eps.esm.imsi` or manually inspect NAS Attach Request (no encryption)	✅ See IMSI in cleartext (payload visible)
Check absence of security activation	`nas-eps.securityheader == 0x0` (no secure NAS header)	✅ No encryption applied yet

Goal	Wireshark Filter	Purpose
Failed bearer setup (missing GTP)	`gtp`	❌ No successful bearer setups (almost no GTP packets)

🧠 Key Difference: 4G eNB vs. 5G gNB

Layer	4G (eNB)	5G (gNB + 5GC split)
Control	eNB talks to **MME (S1AP)** directly	gNB talks to **AMF over NGAP/SCTP (38412)**
Data	eNB handles **GTP-U directly**	UPF (not gNB) handles **GTP-U**
GTP Port	Terminated at eNB	Terminated at **UPF (inside 5g-core)**

5G

Normal

Stage	Command

Start normal profile from terminal 1	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-normal up`
	if srsran_5g_gnb-normal fails to connect run
	`sudo docker restart srsran_5g_gnb-normal`
	and verify it connected

Start ue in terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-normal up 5g-ue-normal --`Wait for the successful network attach and for the notification that registration is complete (In terminal 1)
Send Ping to Core in terminal 3	`sudo docker exec srsran_5g_ue-normal sh -c 'gw=$(ip route | awk "/default/ {print \$3}"); echo "Pinging gateway: $gw"; ping -c 4 $gw; echo ""; echo "Pinging 8.8.8.8"; ping -c 4 8.8.8.8'`
Shut down UE	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-normal down 5g-ue-normal`
Bring down all baseline components from terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-normal down` The pcap files will be under /tmp/pcap

Download pcap file locally and open with wireshark to evaluate:

```
noglob scp -i ~/.ssh/SSH_PRIV USER@PC###.emulab.net:*.pcap .
```

Analysis

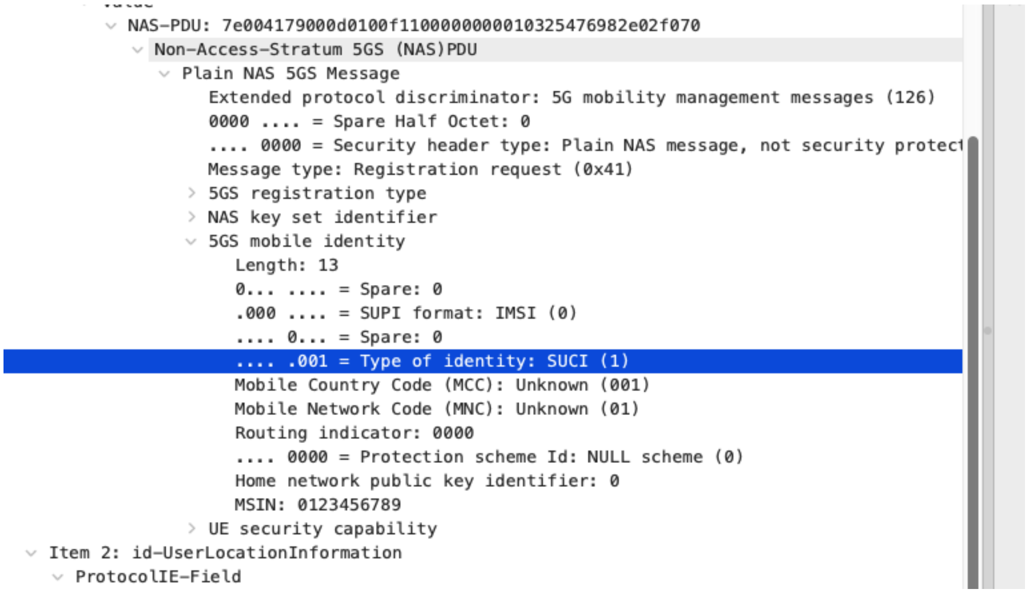
baseline

📦 `gnb_ngap.pcap` – Control Plane Messages (NGAP + embedded NAS)

🔐 Identity & Authentication

Message Type	Purpose	Notes
`Registration Request`	UE initiates connection to the network	Contains SUCI (identity-protected)
`Authentication Request`	Core challenges UE with RAND	AKA authentication begins
`Authentication Response`	UE proves knowledge of shared secret	
`Security Mode Command`	AMF selects encryption/integrity algorithms	Enables NAS encryption
`Security Mode Complete`	UE confirms and activates encryption	

📁 **Found in**: `gnb_ngap.pcap` (inside NGAP → NAS-PDU)



```
    Value
  ⌄ NAS-PDU: 7e004179000d0100f1100000000010325476982e02f070
    ⌄ Non-Access-Stratum 5GS (NAS)PDU
      ⌄ Plain NAS 5GS Message
          Extended protocol discriminator: 5G mobility management messages (126)
          0000 .... = Spare Half Octet: 0
          .... 0000 = Security header type: Plain NAS message, not security protect
          Message type: Registration request (0x41)
        › 5GS registration type
        › NAS key set identifier
        ⌄ 5GS mobile identity
            Length: 13
            0... .... = Spare: 0
            .000 .... = SUPI format: IMSI (0)
            .... 0... = Spare: 0
            .... .001 = Type of identity: SUCI (1)
            Mobile Country Code (MCC): Unknown (001)
            Mobile Network Code (MNC): Unknown (01)
            Routing indicator: 0000
            .... 0000 = Protection scheme Id: NULL scheme (0)
            Home network public key identifier: 0
            MSIN: 0123456789
        › UE security capability
⌄ Item 2: id-UserLocationInformation
  ⌄ ProtocolIE-Field
```

🛰 Session & Attach Flow

Message Type	Purpose	Notes
`Initial UE Message`	gNB forwards UE info to AMF	Contains NAS Registration Req

Message Type	Purpose	Notes
`NG Setup Request/Response`	gNB and AMF setup S1-like connection	Initial NGAP link
`UE Context Setup Request`	AMF asks gNB to configure a PDU session	Includes QoS and IP setup
`UE Context Release`	Triggered on detach or failure	Ends UE session gracefully

---

📦 `gnb_n3.pcap` – User Plane Messages (GTP-U)

Message Type	Purpose	Notes
GTP-U Echo Request	Health check between gNB and UPF	Not always present
GTP-U Data	User IP traffic (e.g., ICMP/Ping)	Indicates working data path

✅ Use this to demonstrate **user-plane encryption need** or **GTP visibility**.

📦 (Optional) `gnb_e1ap.pcap`, `gnb_f1ap.pcap`, `gnb_rlc.pcap`

These are **lower-layer gNB messages**:

- **E1AP**: Between CU-CP and CU-UP
- **F1AP**: Between CU and DU
- **RLC/MAC**: Radio stack (not always useful unless testing RAN behavior)

⚠️ For most **demo/report purposes**, stick to:

- `gnb_ngap.pcap` for **control plane**
- `gnb_n3.pcap` for **data/user plane**

Man-In-The-Middle/String Encryption Test

Stage	Command

Bring up real AMF, MITM relay	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm up 5g-core-normal`
	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm up 5g-mitm-relay`

Tcpdump inside relay from terminal 2	`sudo docker exec -it srsran_5g_mitm-relay tcpdump -i any -U -vv -w /tmp/5g-mitm-sctp-capture.pcap port 38412 or port 2152`
Startup gNB from terminal 3	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm up 5g-gnb-mitm`

Stage	**Command**
	Should see srsran_5g_core-normal | 05/07 05:47:06.212: [amf] INFO: gNB-N2[10.56.1.99] connection refused!!! (../src/amf/amf-sm.c:793)
Startup MITM ue from terminal 4	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm up 5g-ue-mitm` Wait for the notification in of RCC release
Run ping	`sudo docker exec srsran_5g_ue-mitm sh -c 'gw=$(ip route | awk "/default/ {print \$3}"); echo "Pinging gateway: $gw"; ping -c 4 $gw; echo ""; echo "Pinging 8.8.8.8"; ping -c 4 8.8.8.8'`
Stop ue from terminal 3	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm down 5g-ue-mitm`
Copy rogue capture	CTRL+C to stop the capture `sudo docker cp srsran_5g_mitm-relay:/tmp/5g-mitm-sctp-capture.pcap ./5g-mitm-sctp-capture.pcap`
Bring down all baseline components from terminal 2	`sudo docker compose -f /opt/srsRAN_Project/docker/docker-compose-4g-5g.yaml --profile 5g-mitm down`

Download pcap file locally and open with wireshark to evaluate:

```
noglob scp -i ~/.ssh/SSH_PRIV USER@PC###.emulab.net:*.pcap
```

Analysis

```
wireshark mitm-sctp-capture.pcap
```

Run the following filters to demonstrate:

Goal	**Wireshark Filter**	**Purpose**
See all S1AP signaling	`s1ap`	✅ Attach attempts captured at mitm
Focus only on NAS messages	`nas-eps`	✅ Look at plain NAS Attach Request
Catch IMSI exposure	`nas-eps.esm.imsi` or manually inspect NAS Attach Request (no encryption)	✅ See IMSI in cleartext (payload visible)
Check absence of security activation	`nas-eps.securityheader == 0x0` (no secure NAS header)	✅ No encryption applied yet
Failed bearer setup (missing GTP)	`gtp`	❌ No successful bearer setups (almost no GTP packets)