# VET5G: A Virtual End-to-End Testbed for 5G Network Security Experimentation

Zhixin Wen, Harsh Sanjay Pacherkar, Guanhua Yan

*Department of Computer Science*

Binghamton University

This work is supported by NSF Award CNS-1943079

# Background & Motivation

- 5G deployment has been ongoing around the world

Home › 5G › China ends Q1 with over 400 million 5G users

5G | Carriers | Featured | Network Infrastructure

## China ends Q1 with over 400 million 5G users

**Juan Pedro Tomás** - April 20, 2022 0

WIRELESS

## T-Mobile: 2.5 GHz 5G now covers 200M people

By **Monica Alleven** • Nov 15, 2021 12:13pm

2.5 GHz | 5G | midband spectrum | AT&T

- Attacks and defenses on 5G are being discovered and proposed, however either no experiments were conducted or was conducted in 5G NSA mode

5G SECURITY - 4 MIN READ

## 5G Network Slicing Vulnerability: Location Tracking Attacks

**Máirín OSullivan**

27 APR 2021

## 5GReasoner: A Property-Directed Security and Privacy Analysis Framework for 5G Cellular Network Protocol

Syed Rafiul Hussain, Purdue University, hussain1@purdue.edu
Mitziu Echeverria, University of Iowa, mitziu-echeverria@uiowa.edu
Imtiaz Karim, Purdue University, karim7@purdue.edu
Omar Chowdhury, University of Iowa, omar-chowdhury@uiowa.edu
Elisa Bertino, Purdue University, bertino@purdue.edu

# Existing 5G implementations/testbeds

So far open source/academic 5G solutions are mostly focused on either core network or air interface

Open5GS

Open source project of 5GC and EPC (Release-16)

free5GC

**Core network**

OPEN AIR INTERFACE

SRSRAN

**RAN**

open5Gcore

open5Gcore is closest to our work, however it is not designed for security research

# Testbeds

- DETER Project
- SCADA Testbed
- Cyber-Power Testbed
- Security Testbed for Internet-of-Things Devices
- Emulab
- And more…

# Goal & Solution

To address the problem of lacking of testbeds, we introduce **VET5G** to meet the following requirements:

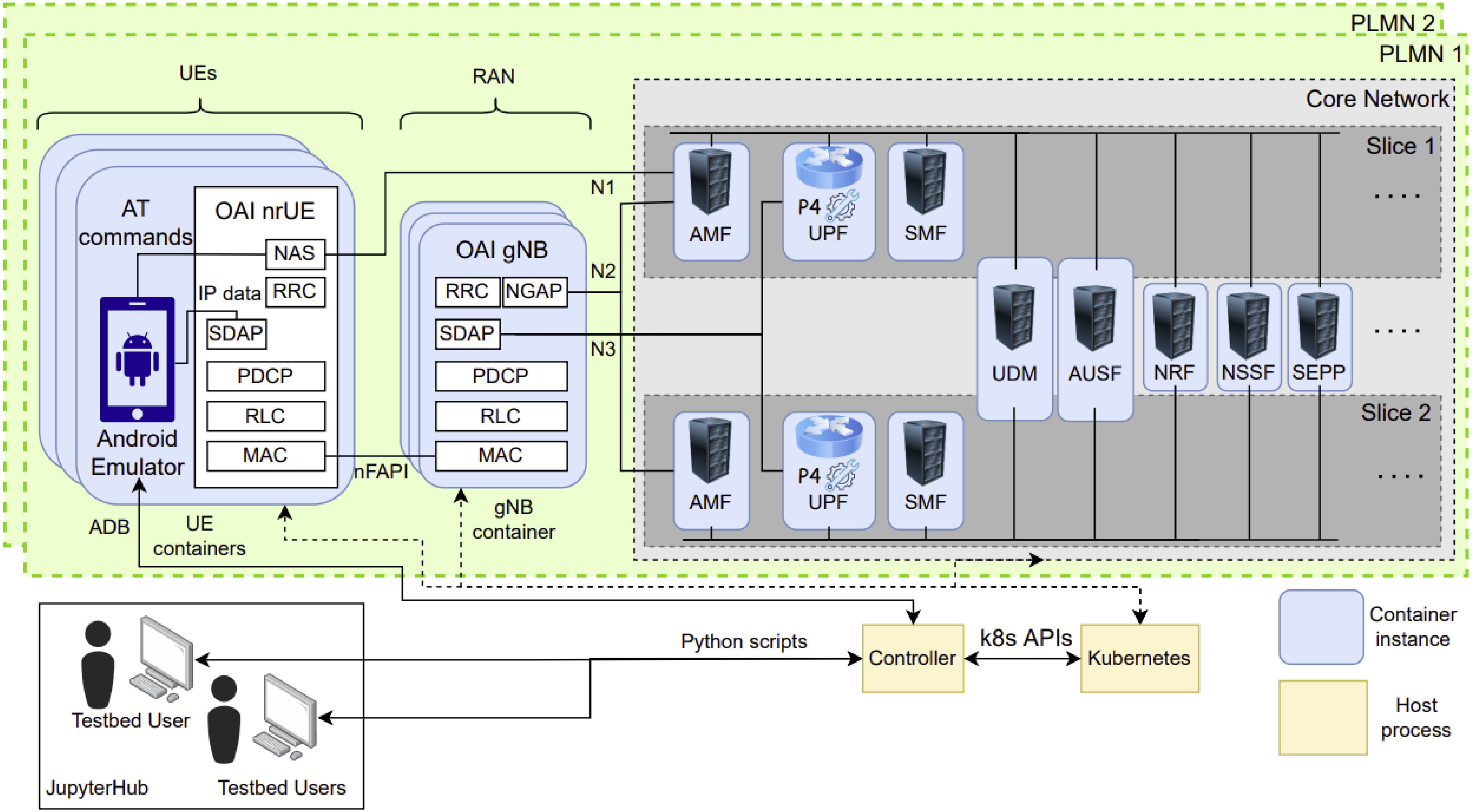
- 1. **End-to-end** emulation from UE to core network
- 2. **Easy** to use and **ready** to use testbed with JupyterLab

Doing so researchers can focus on research not testbed setup, saving time.

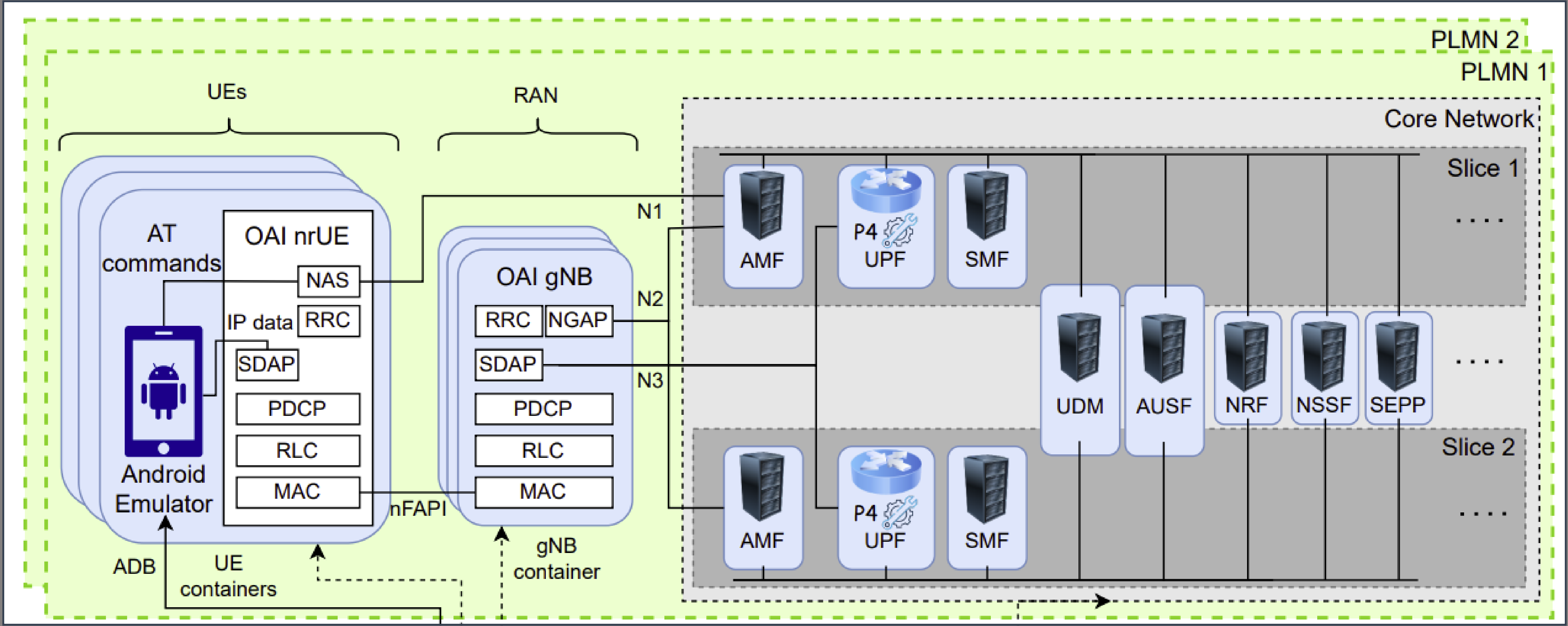
In addition we provide

- 3. **Secure** 5G core implementation in rust
- 4. **Programmable** UPF with P4

# Architecture overall

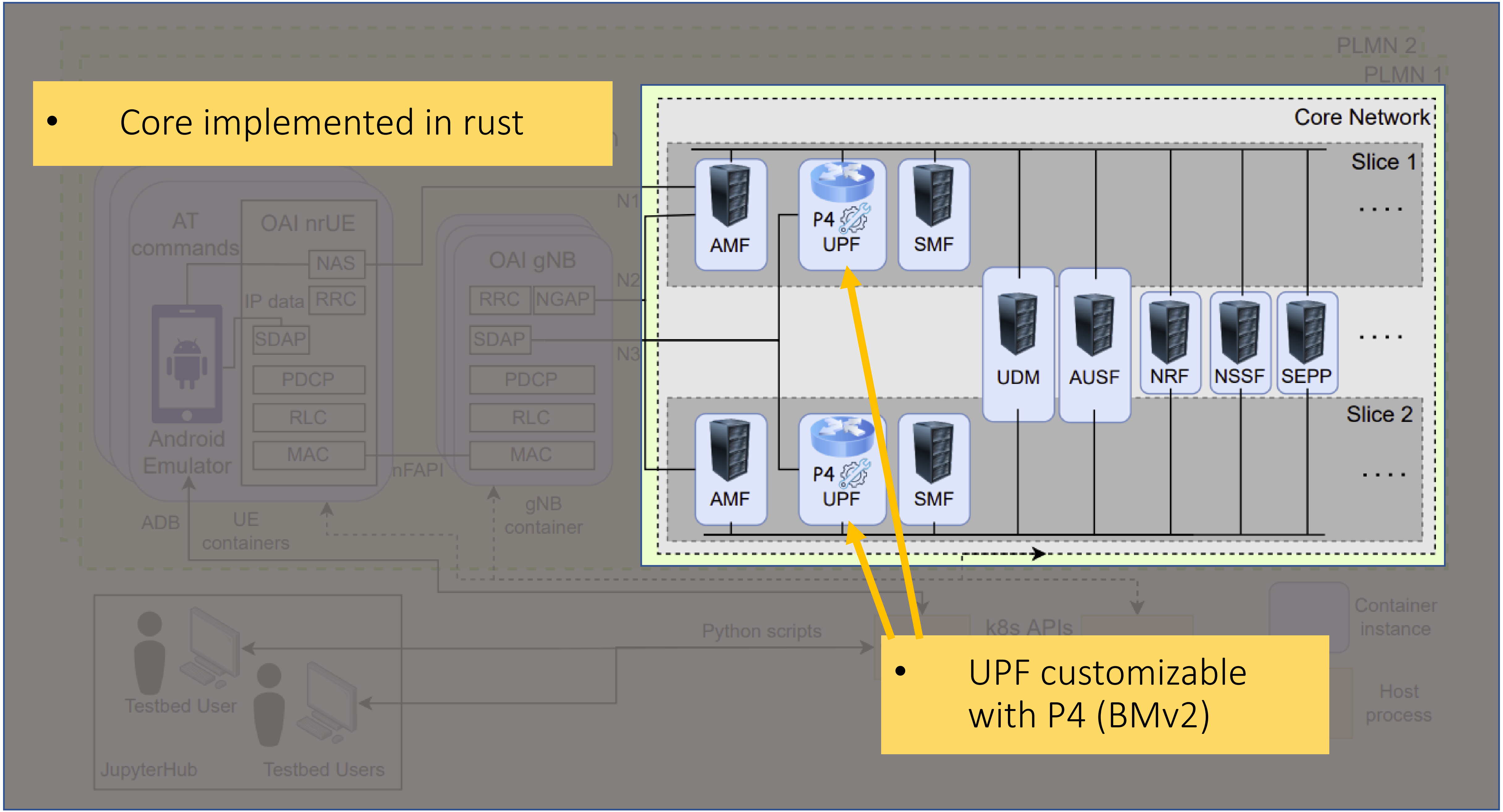


# Architecture overall




- Multiple PLMNs, each contains:
- Core network
- RAN
- UEs

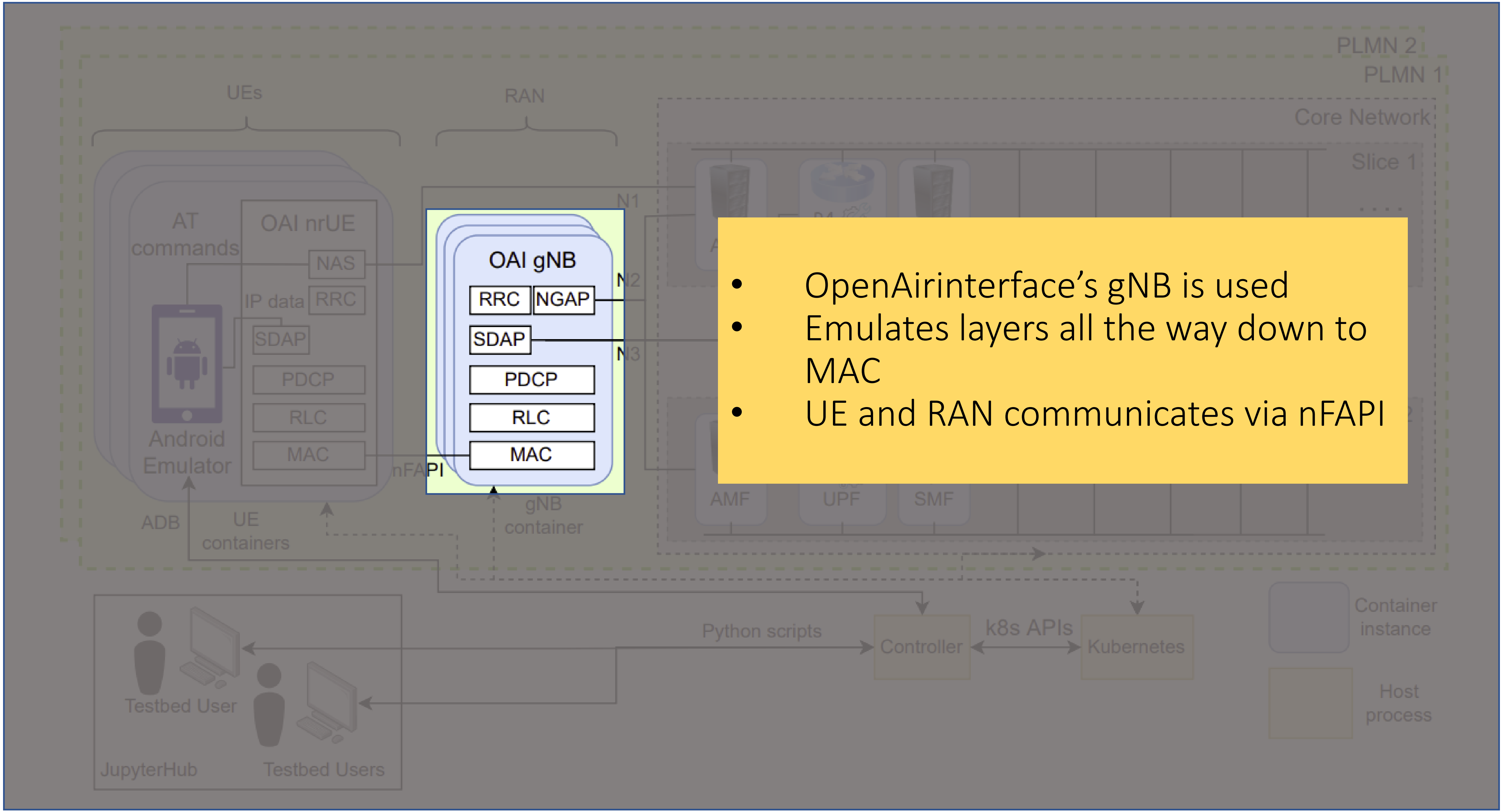
# Core network emulation




- Core implemented in rust

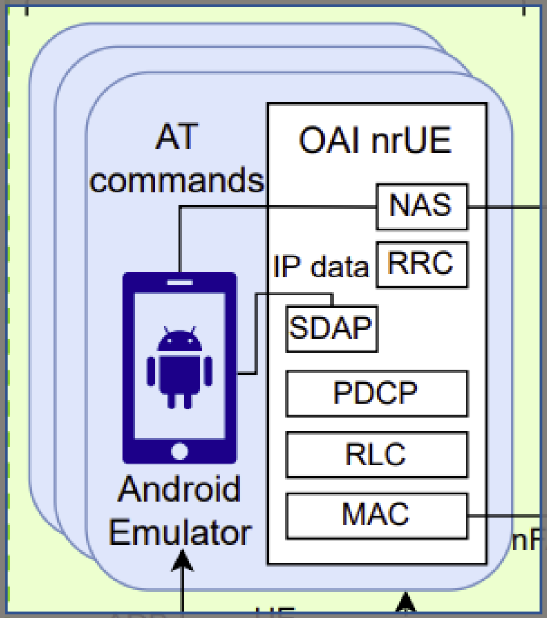
- UPF customizable with P4 (BMv2)

# Access Network emulation

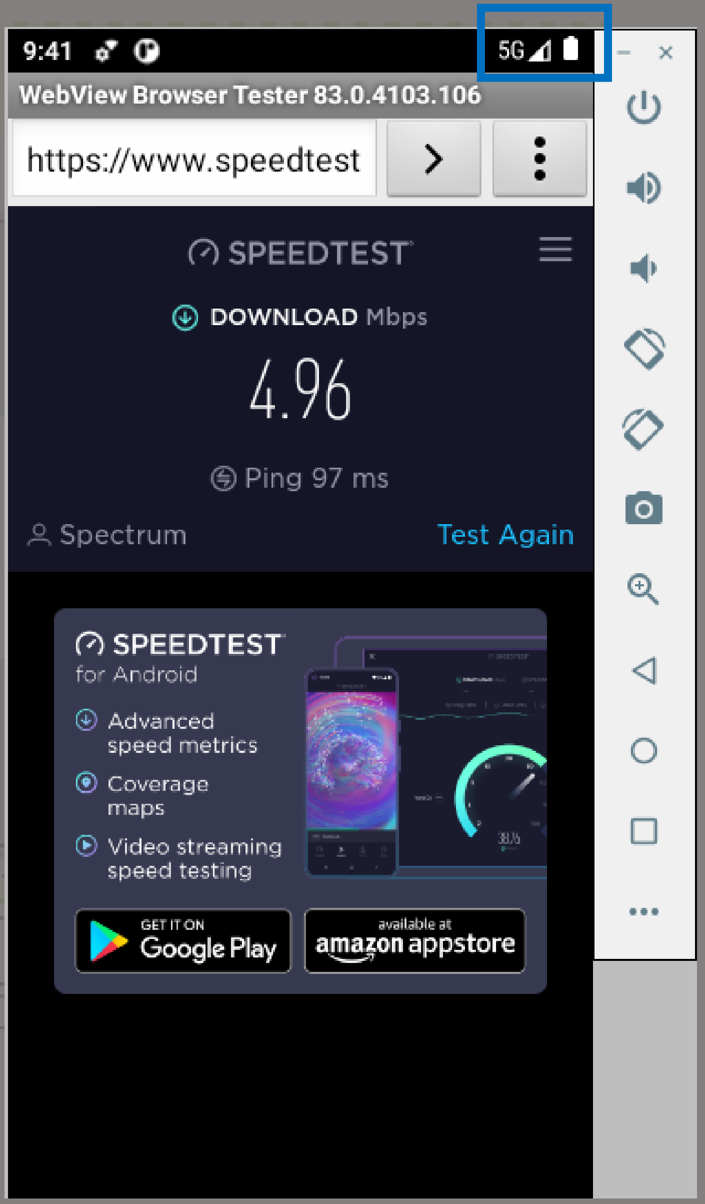



- OpenAirinterface's gNB is used
- Emulates layers all the way down to MAC
- UE and RAN communicates via nFAPI

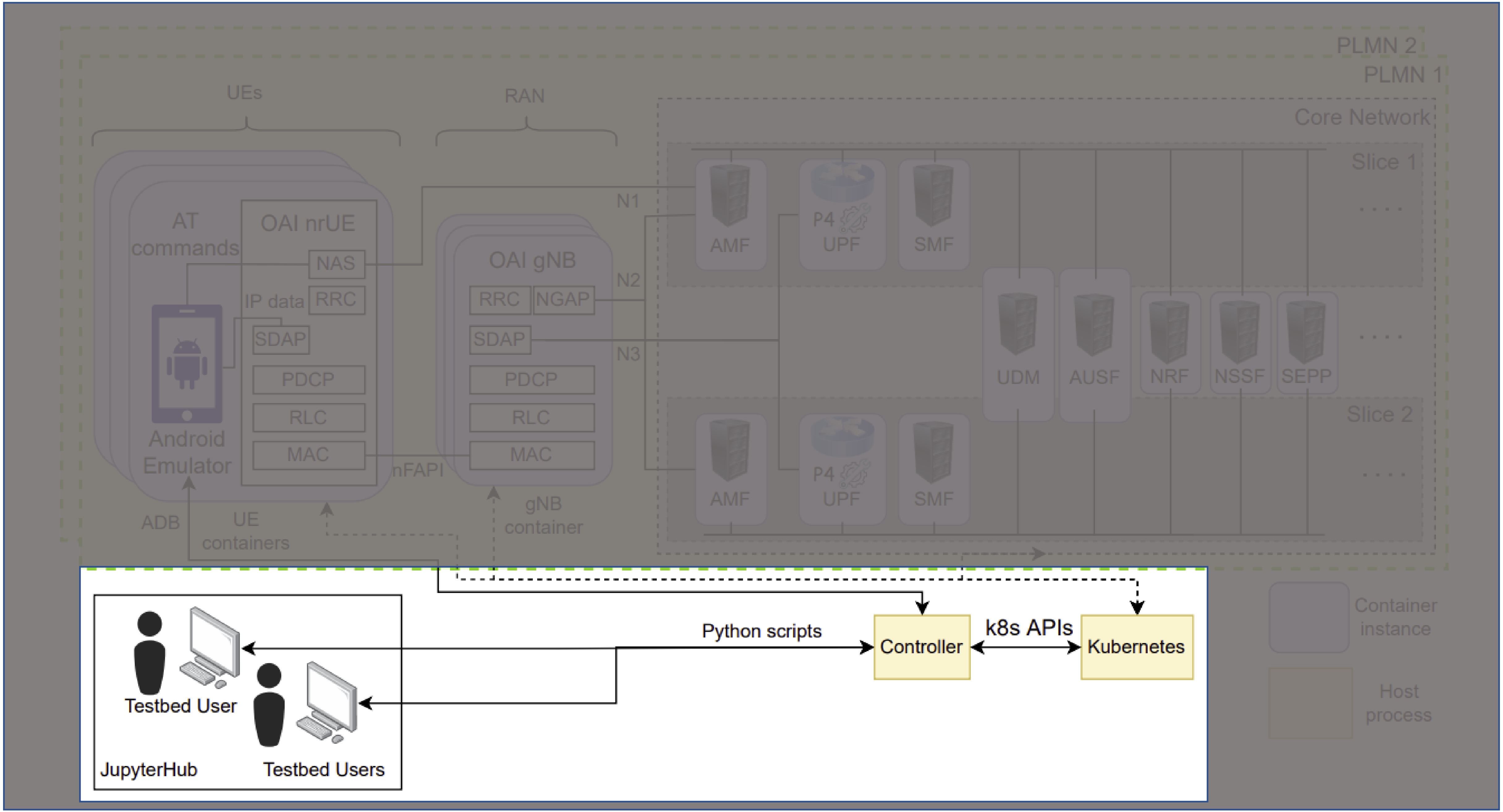
# User Equipment emulation




- Android emulator with Android 11 is used
- OAI’s nrUE is used to emulate baseband processor



# User Interface



# Orchestration & User Interface

- **Kubernetes** is used to orchestrate all containerized components
- Users interact with the testbed via **JupyterHub**
- Users define the testbed via yaml config file, this includes
  - Config of each container, be that NF, UE or gNB
  - Config of each subscriber

```yaml
smfs:
  - name: 'smf1'
    slices:
      - name: 'slice-1'
        sst: 1
        sd: '000001'
    ip_pool:
      - name: 'default'
        ipv4_cidr: '10.10.0.0/16'
    allowed_slices: null
    dump_pcap: true
    dump_logs: true
```

```yaml
subscribers:
  - imsi: '2089900007487'
    ul_ambr: 100 # unit: KBps
    dl_ambr: 100 # unit: KBps
    subscribed_slices:
      - name: 'slice-1'
        sst: 1
        sd: '000001'
```

| **Python API** | **Meaning** |
|---|---|
| *create_subscribers* | Create mobile subscribers within a PLMN |
| *start_core_network* | Start a core network instance within a PLMN |
| *start_gnb* | Start gNB(s) within a PLMN |
| *start_ue* | Start UE(s) within a PLMN |
| *execute_shell_command* | Execute a shell command in an Android Emulator |
| *install_apk* | Install an Android APK in an Android Emulator and return its package name |
| *start_app* | Start an Android APK with a given package name in an Android Emulator |
| *stop_all* | Stop an experiment and collect results |
| *get_nf_log* | Get logs of an NF instance (it must be called after *stop_all*) |

# Orchestration & User Interface

In [18]:
```python
from vet5g.client import VET5GClient
from vet5g.models import *
```

In [19]:
```python
plmn = PlmnId("301", "91")
vet5g = VET5GClient("username", [plmn], "/tmp/")
await vet5g.create_subscribers(plmn, cfg_file = 'demo1_subscribers.yaml')
cn_cfg = await vet5g.start_core_network(plmn, cfg_file = 'demo1_cn.yaml')
gnb_cfg = await vet5g.start_gnb(plmn, cfg_file = 'demo1_gnb.yaml')
ue_cfg = await vet5g.start_ue(plmn, cfg_file = 'demo1_ue.yaml')
await vet5g.stop_all(pcap)
```

| Python API | Meaning |
|---|---|
| *create_subscribers* | Create mobile subscribers within a PLMN |
| *start_core_network* | Start a core network instance within a PLMN |
| *start_gnb* | Start gNB(s) within a PLMN |
| *start_ue* | Start UE(s) within a PLMN |
| *execute_shell_command* | Execute a shell command in an Android Emulator |
| *install_apk* | Install an Android APK in an Android Emulator and return its package name |
| *start_app* | Start an Android APK with a given package name in an Android Emulator |
| *stop_all* | Stop an experiment and collect results |
| *get_nf_log* | Get logs of an NF instance (it must be called after *stop_all*) |

# VET5G use cases

- 1. Slicing attack
- 2. Cellular botnet and its defense
- 3. Course project

# Slicing attack

- A design flaw in 5G protocol gives attacker access to information in another slice it otherwise shouldn't
- We can simulate this attack by creating a **custom NF** using scaffolding code provided

5G SECURITY - 4 MIN READ

**5G Network Slicing Vulnerability: Location Tracking Attacks**

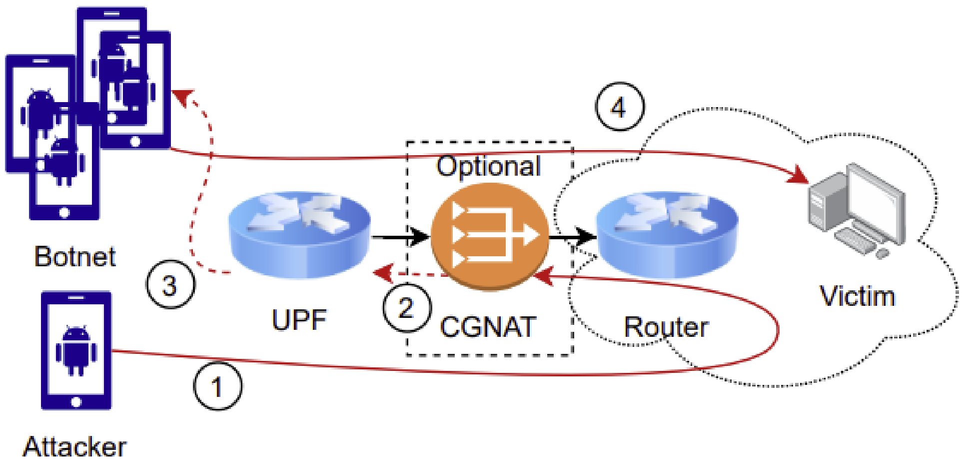
Máirín OSullivan

27 APR 2021

```yaml
custom_nfs:
  - name: slicing-attacker
    image_name: 'slicing-attacker-nf.tar'
    image_args: []
    slices:
      - name: 'slice-2'
        sst: 1
        sd: '000002'
    allowed_slices: null
    dump_pcap: true
    dump_logs: true
```

# Cellular botnet and its defense

- As more devices are connected, the harm a botnet can cause to mobile network increases
- Simple 5G botnet and its defense is demonstrated here
- Matryosh-like cellular botnet is studied here
- P4 **programmable UPF** allows defense to be quickly **prototyped**

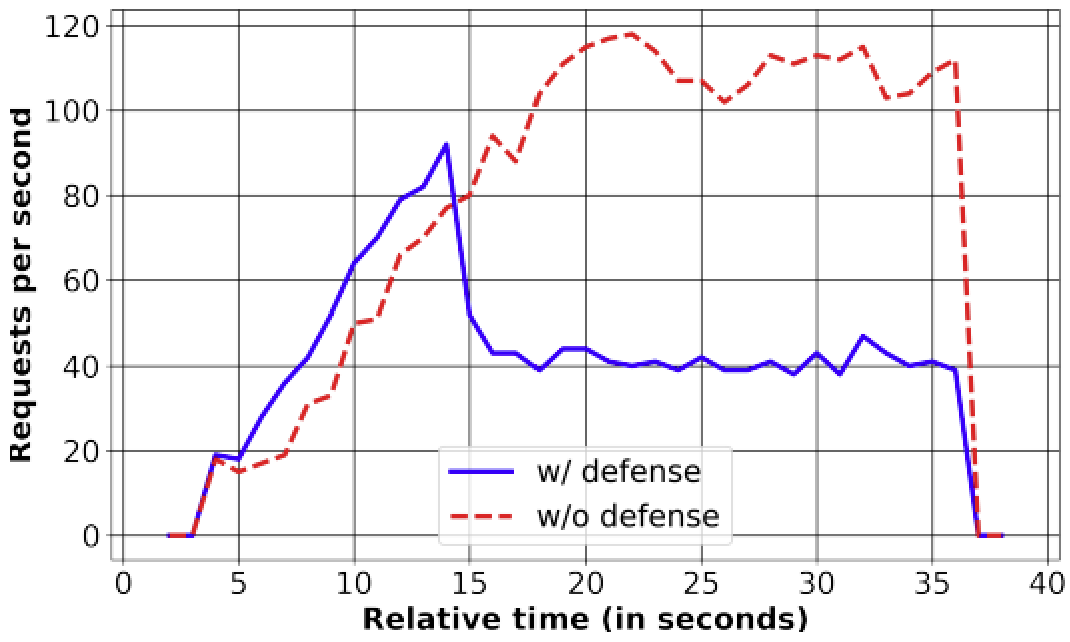



```
register<bit<32>>(2048) count_min_sketch_1;
register<bit<32>>(2048) count_min_sketch_2;
register<bit<32>>(2048) count_min_sketch_3;

bit<32> regpos_src;
bit<32> regpos1;
bit<32> regpos2;
bit<32> regpos3;
bit<32> regval1;
bit<32> regval2;
bit<32> regval3;
bit<32> count1;
bit<32> count2;
bit<32> count3;
bit<32> minvalue;
bool ddos_detected = false;
bool ddos_confirmed = false;

action detect_ddos(bit<32> srcAddr, bit<32> dstAddr) {
    hash(regpos_src, HashAlgorithm.crc16, (bit<32>)0, {srcAddr,
    hash(regpos1, HashAlgorithm.crc16, (bit<32>)0, {dstAddr, 32w
    hash(regpos2, HashAlgorithm.crc32, (bit<32>)0, {dstAddr}, (b
    hash(regpos3, HashAlgorithm.crc32_custom, (bit<32>)0, {dstAd
    count_min_sketch_1.read(regval1, regpos1);
    count_min_sketch_2.read(regval2, regpos2);
    count_min_sketch_3.read(regval3, regpos3);
    regval1 = regval1 | (32w1 << (bit<8>)(regpos_src));
    regval2 = regval2 | (32w1 << (bit<8>)(regpos_src));
    regval3 = regval3 | (32w1 << (bit<8>)(regpos_src));
    count_min_sketch_1.write(regpos1, regval1);
    count_min_sketch_2.write(regpos2, regval2);
    count_min_sketch_3.write(regpos3, regval3);
    count1 = numberOfSetBits(regval1);
    count2 = numberOfSetBits(regval2);
    count3 = numberOfSetBits(regval3);
```

# Cellular botnet and its defense

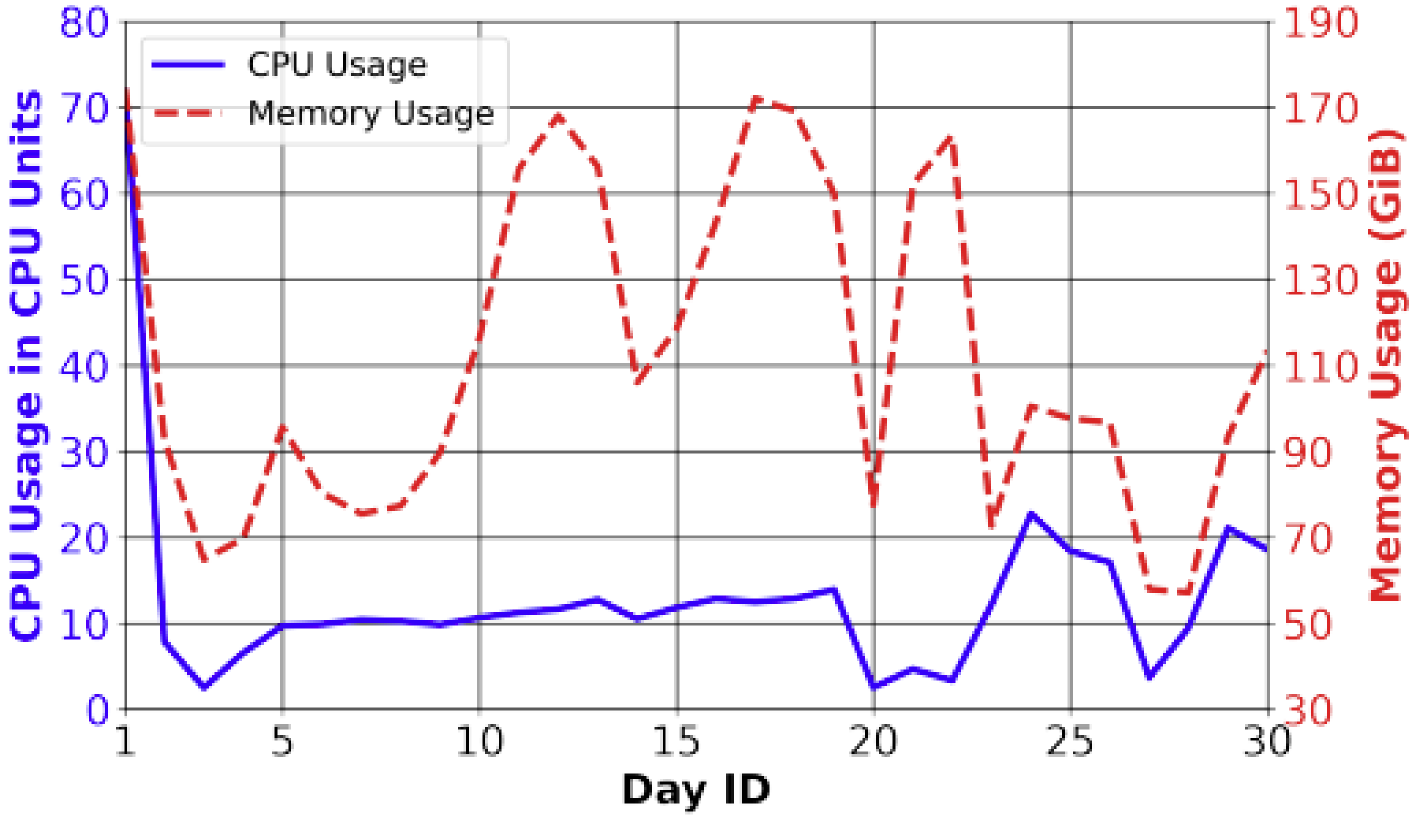


# Course project

- VET5G is here used for educational activity
- 40 students in 15 groups try to attack 5G system from an vantage point within the core network

| Attack Type | Description | Number of groups |
|---|---|---|
| Fuzzing | Use tools such as wfuzz and sfuzz to fuzz test 5G core NFs | 14 |
| Reconnaissance | Use tools such as nmap to find attack targets in 5G networks | 9 |
| GTP attack | Guess TEIDs (Tunnel Endpoint Identifiers) used by legitimate GTP sessions | 3 |
| DDoS attack | Use tools such as slowloris to perform DDoS attacks | 3 |
| Password attack | Use tools such as Hydra to guess passwords | 3 |
| MITM attack | Use tools such as Bettercap to perform MITM (Man-In-The-Middle) attacks | 2 |
| SSH attack | Use tools such as metasploit to attack vulnerable SSH services | 1 |

# Course project



# Conclusion & Future work

- Integrate real SDR devices in VET5G we can use real 5G devices
- Incorporating more diverse end devices to study their security in a holistic way
- Integrate a Tofino based programable UPF
- Intentionally introduce vulnerabilities for future CTF events
- Study the security of rust implementation