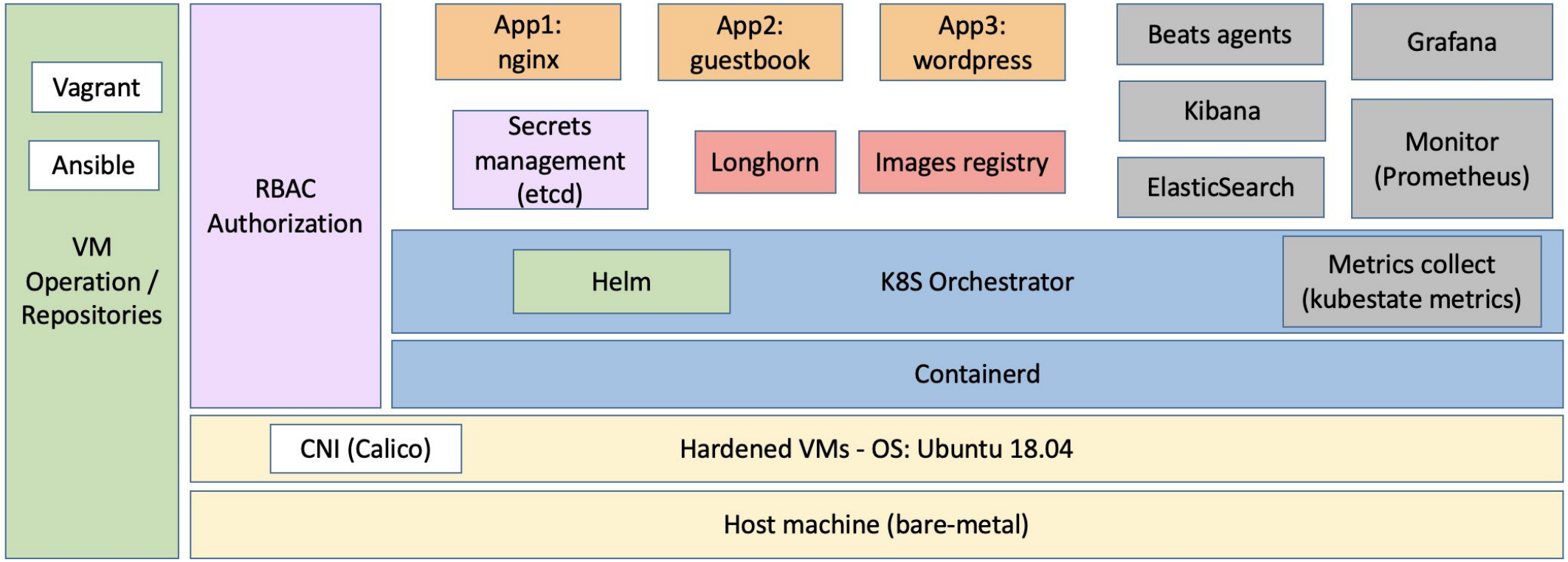
Cybersécurité des infrastructures pour le cloud

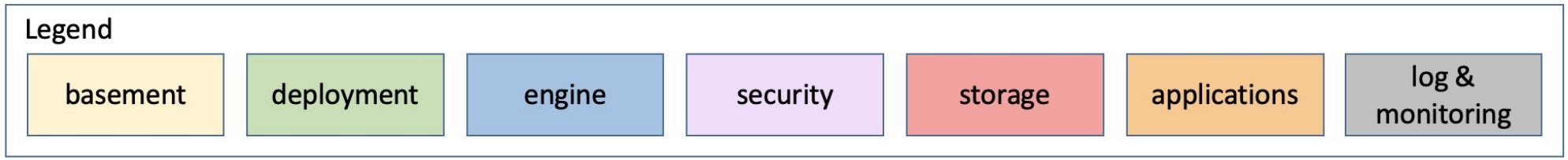
Agathe Blaise (Thales), Jacopo Bufalino (CNAM)

Practical testbed

{OPEN}

Practical testbed





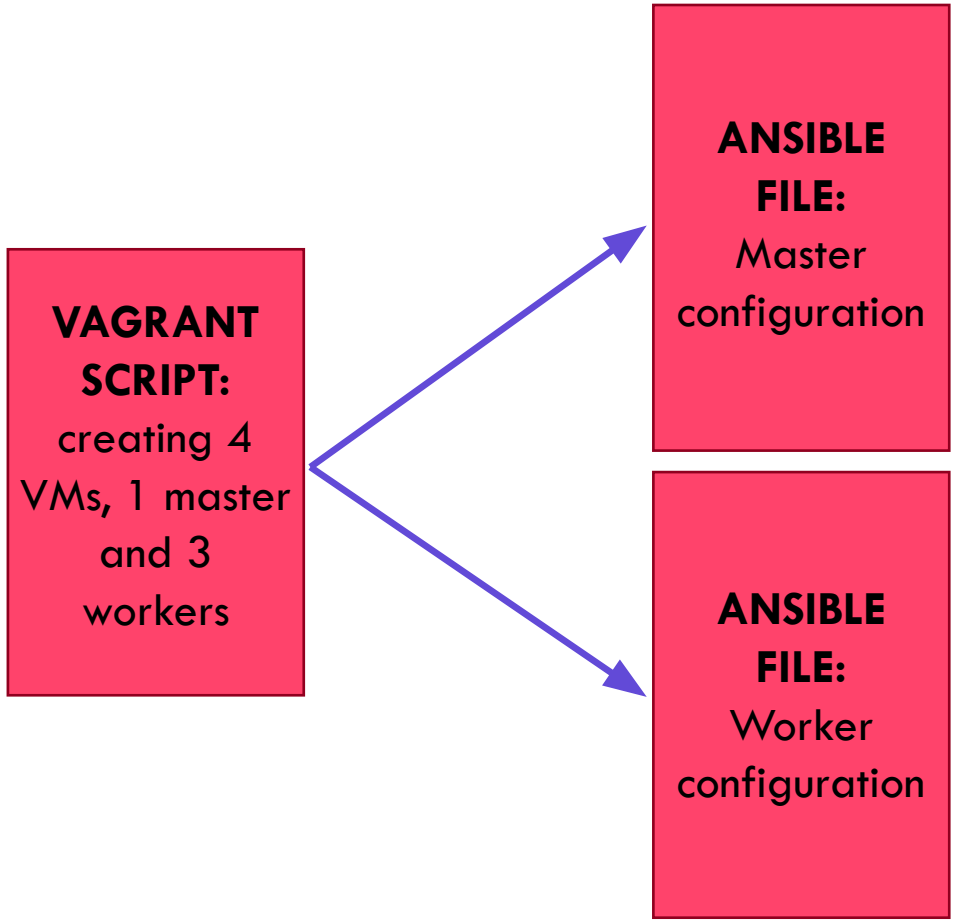
Summary

basement	- Set up the Kubernetes cluster - CNI: Calico
deployment	- Vagrant and Ansible scripts to initiate the kubernetes cluster
engine	- Containerd and Kubernetes orchestrator
security	- Network policies - RBAC authorization - Secrets management
storage	- Longhorn for distributed storage - NFS server for persistent storage
applications	- 3 applications: nginx, guestbook and wordpress - + monitoring applications: ElasticSearch, Kibana, vulnerable images, …
log & monitoring	- Monitoring: Metrics and alerts - Logging: Aggregation, Searching, & Filtering

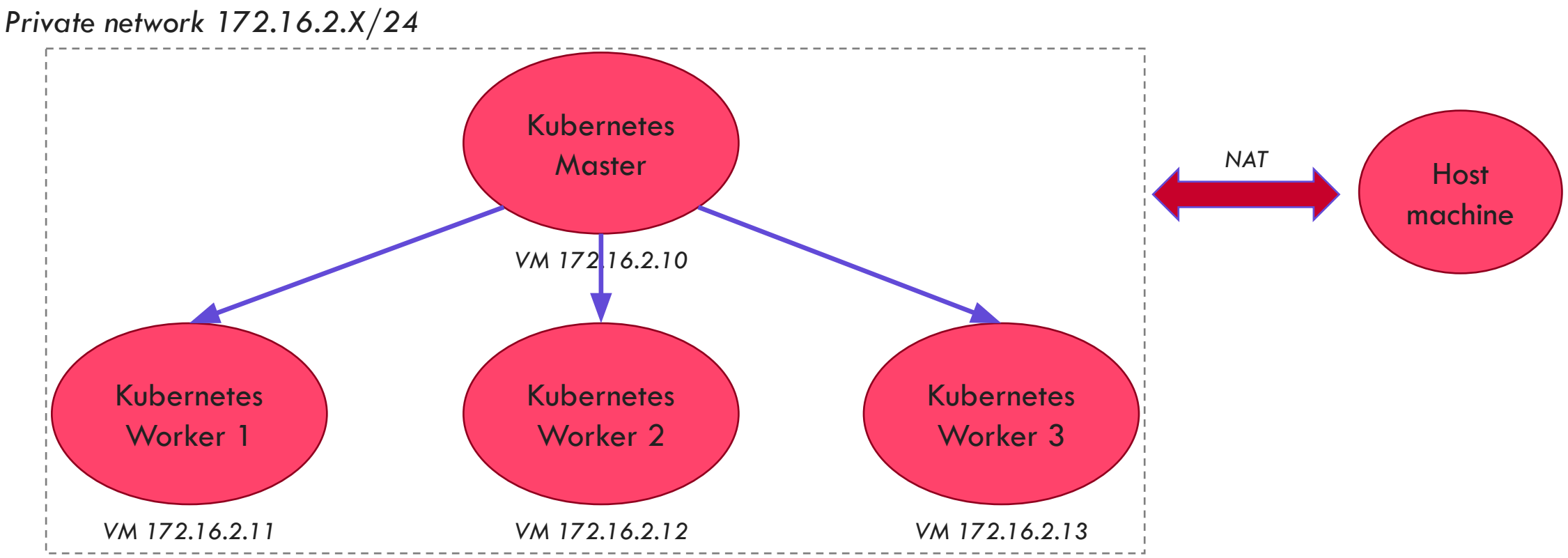
Deployment on the host machine

Objective: provide the environment to be replicated with very few dependencies:

- Vagrant: automation
- Ansible: provisioning
- A virtualization module: virtualbox, libvirt, …



Kubernetes cluster setup

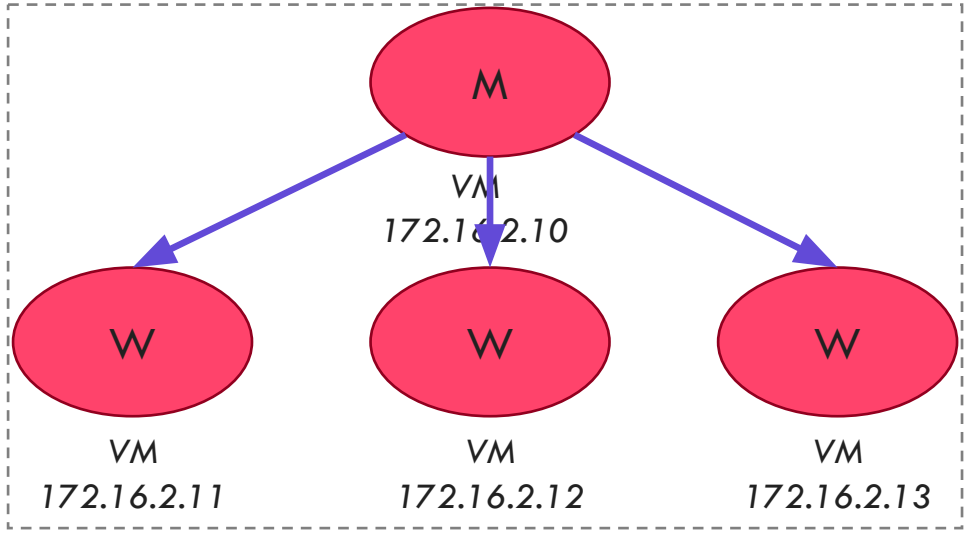


→ One single cluster

Dev and *Prod* environments isolated via namespaces

Isolated namespaces

Physical cluster:



Several isolated **namespaces**, i.e., ''virtual clusters"

kube-system	By default: DNS, scheduler, proxy, calico, etcd, …
default	By default: for objects created without providing the namespace
longhorn	Custom resources (deployments, services, network policies, …)
nfs-server	Custom resources (deployments, services, network policies, …)
prod	Custom resources (deployments, services, network policies, …)
dev	Custom resources (deployments, services, network policies, …)

Each namespace has its own:

1. Named resources
2. Delegated management authority to trusted users
3. Ability to limit community resource consumption

{OPEN}

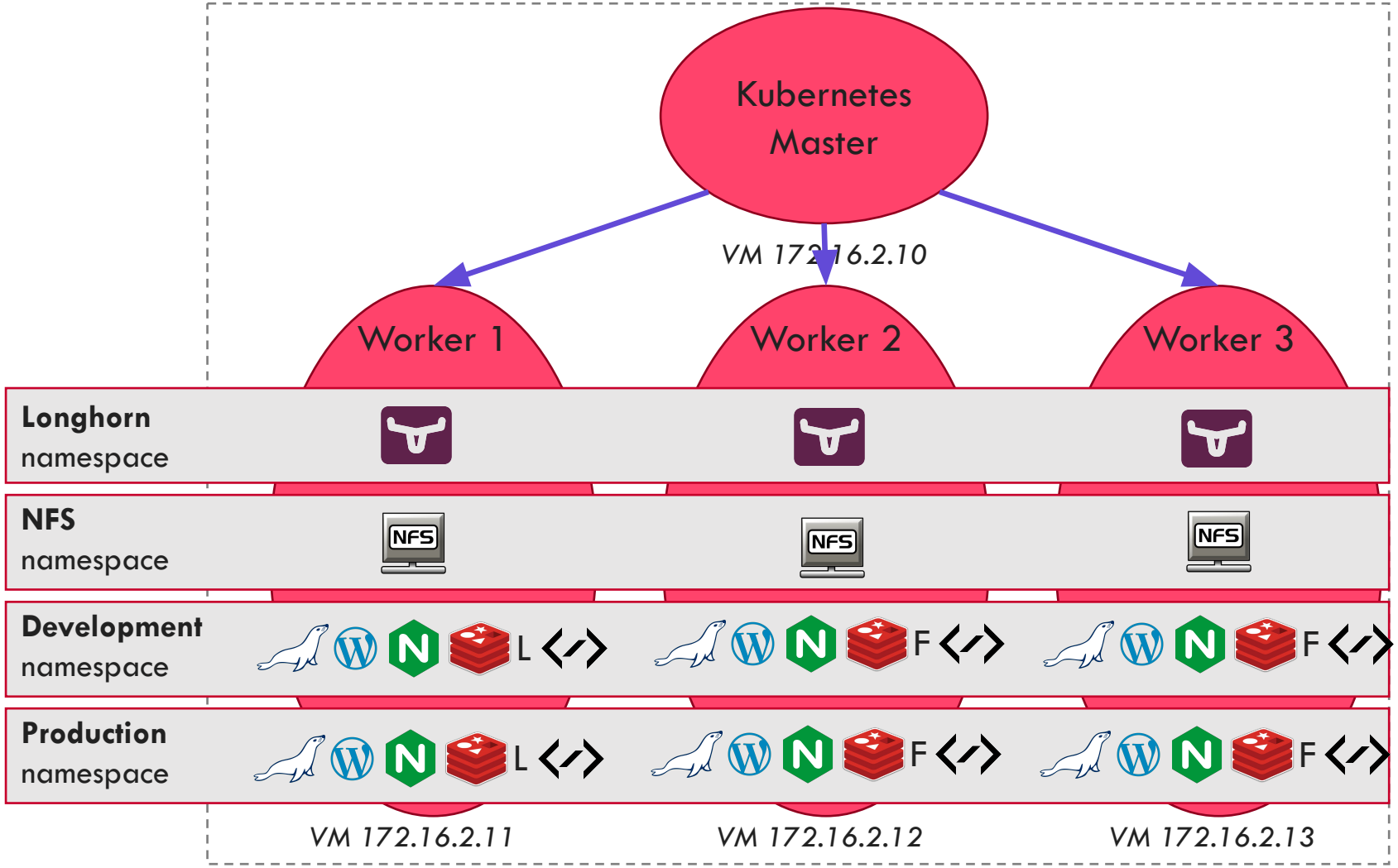
Set of resources within each namespace

- **Pods**: smallest deployable units of computing that you can create and manage in Kubernetes.
- **Deployments**: provides declarative updates for Pods.
- **StatefulSets**: workload API object used to manage stateful applications.
- **Services**: abstract way to expose an application running on a set of Pods.
- **Network Policies**: specify how a Pod is allowed to communicate with various network entities
  - **Kubernetes** network policies: apply only to pods.
  - **Calico** network policies: apply to pods, VMs, host interfaces. Richer set than policy capabilities of K8S including: policy ordering / priority, deny rules, more flexible match rules.
- **Pod Security Policies**: cluster-level resource that controls security sensitive aspects of the pod specification
- **Volumes and Persistent Volumes**: pieces of storage in the cluster.

Kubernetes testbed

4 custom namespaces:

- **Longhorn**: providing distributed storage if no cloud provider
- **NFS server**: providing persistent storage
- **Development**: 3 applications (nginx, guestbook and wordpress)
- **Production**: same applications than development namespace but stricter network policies



{OPEN}

Kubernetes testbed

Wordpress application

2 **deployment:**
- MariaDB + Wordpress

2 **services:**
- NodePort for Wordpress; else ClusterIP

Nginx application

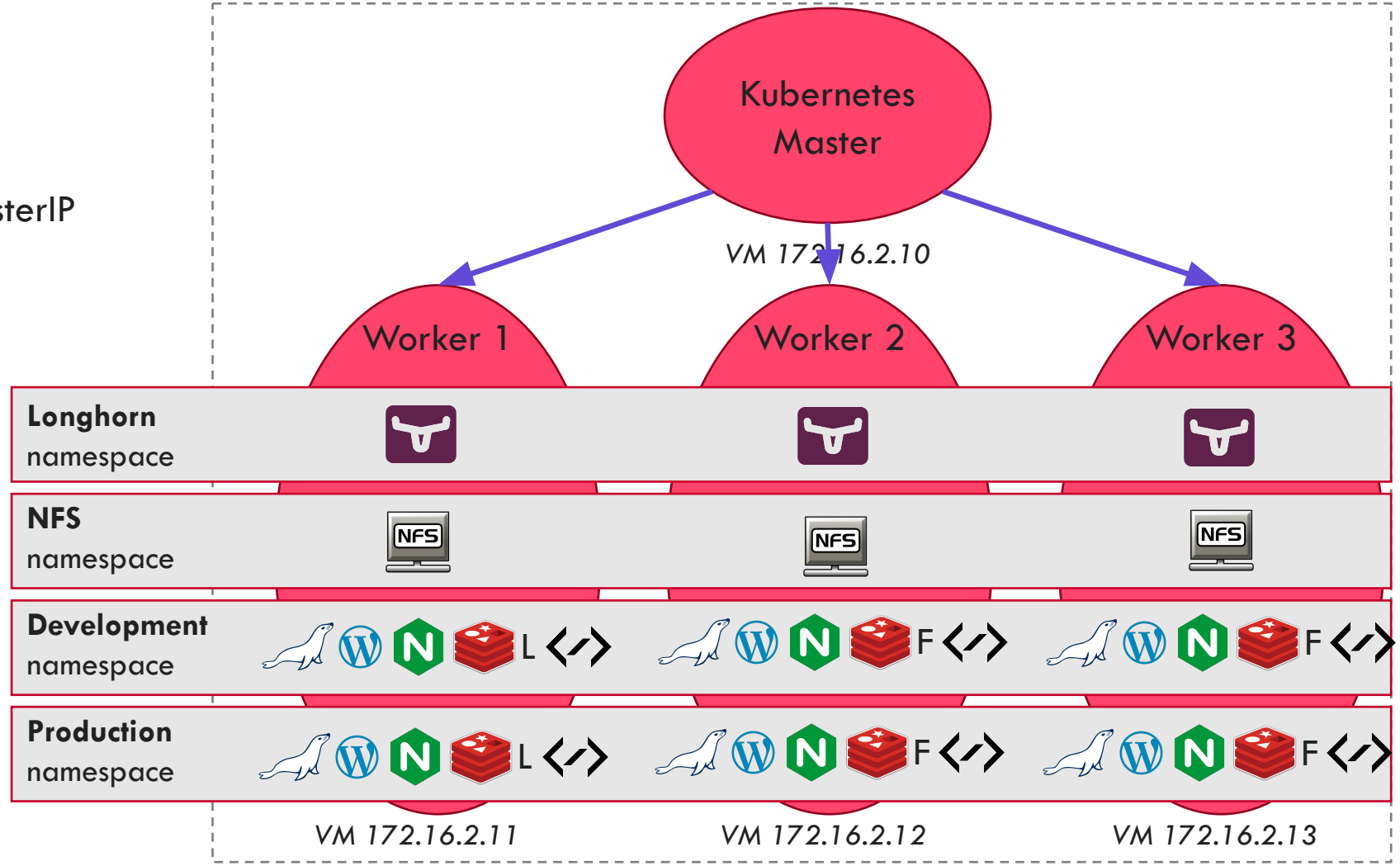
1 **deployment:**
- Nginx server (3 replicas)

1 **service:**
- NodePort for nginx

Guestbook application

3 **deployments:**
- Redis leader (1 replica) L
- Redis follower (2 replicas) F
- Frontend(3 replicas)

3 **services:**
- ClusterIPs for Redis nodes
- NodePort for frontend



{OPEN}

Deployments: guestbook application

kubernetes-engine-samples / guestbook / redis-follower-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-follower
  labels:
    app: redis
    role: follower
    tier: backend
spec:
  replicas: 2
  selector:
    matchLabels:
      app: redis
  template:
    metadata:
      labels:
        app: redis
        role: follower
        tier: backend
    spec:
      containers:
      - name: follower
        image: gcr.io/google_samples/gb-redis-follower:v2
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        ports:
        - containerPort: 6379
```

2 replicas for redis-follower

Reference of the Docker image

{OPEN}

Services: guestbook application

Service creation: exposing the deployment **redis-follower**

kubernetes-engine-samples / guestbook / **redis-follower-service.yaml**

```yaml
apiVersion: v1
kind: Service
metadata:
  name: redis-follower
  labels:
    app: redis
    role: follower
    tier: backend
spec:
  ports:
    # the port that this service should serve on
  - port: 6379
  selector:
    app: redis
    role: follower
    tier: backend
```

Service accessible via port 6379

Adding a label ("selector") to the service definition.
To help routing the traffic by adding iptables policies, via the *kube-proxy:* kubernetes component that powers the services concept and runs in iptables mode by default.

{OPEN}

Network policies: guestbook application

Redis service is well routed and accessible via port 6379, but **no restriction about the pods that can access it.** → Definition of k8s/calico network policies.

service/networking/nginx-policy.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: access-nginx
spec:
  podSelector:
    matchLabels:
      app: nginx
  ingress:
  - from:
    - podSelector:
        matchLabels:
          access: "true"
```

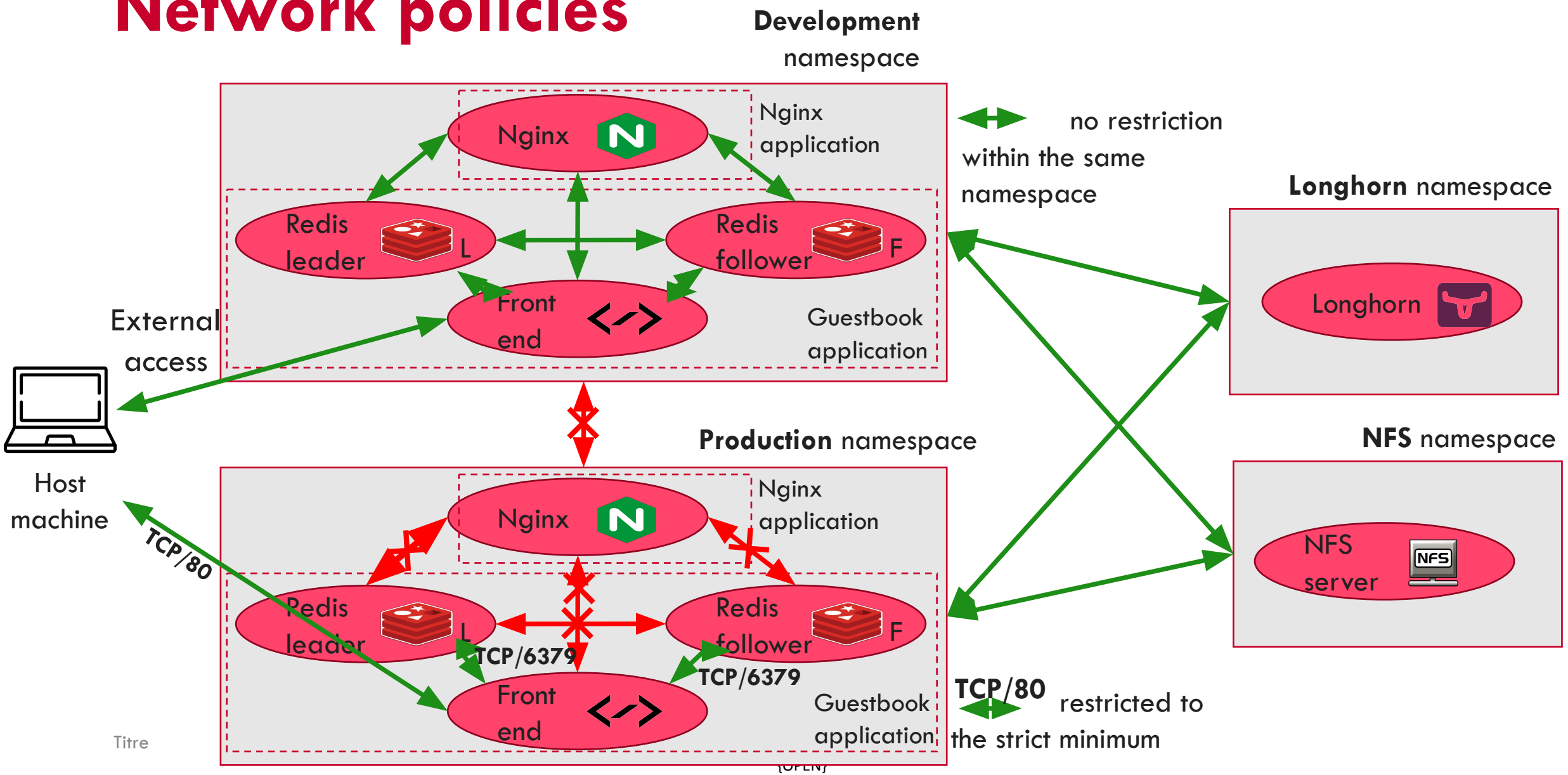
Limit the access to the nginx service so that only Pods with the label “access: true” can query it.

By default: pods are **non-isolated** and accept traffic from any source.

Become isolated by having a NetworkPolicy selecting them.

In our setting, we do not provide network policy for the Longhorn, NFS-server, Wordpress and MariaDB applications, as they are set in the StatefulSet and Deployment provided by Helm.

Network policies



Secrets: wordpress applications

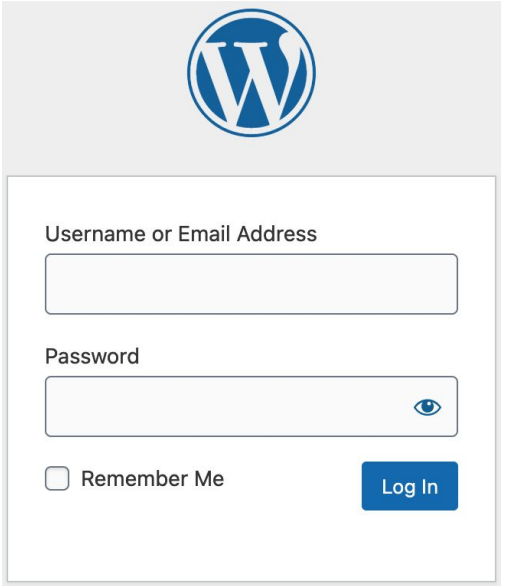
Secret: small amount of sensitive information data such as a password, an OAuth token, or a SSH key, stored and managed by Kubernetes. Users can create Secrets and the system also creates some Secrets.

Example when setting up the Wordpress application with Helm:

- **wordpressUsername=admin and wordpressPassword=adminpassword**

Run `kubectl get secret --all-namespaces` from the k8s-master to see all the stored secrets.

Authentication to the admin website at http://172.16.2.10:32000/admin with these credentials



{OPEN}

5G core network

{OPEN}

Some history on mobile networks

{OPEN}

Cellular networks

- **Cellular networks** enable millions of wireless phones to operate simultaneously, whether stationary or in motion — even at high speeds or across long distances

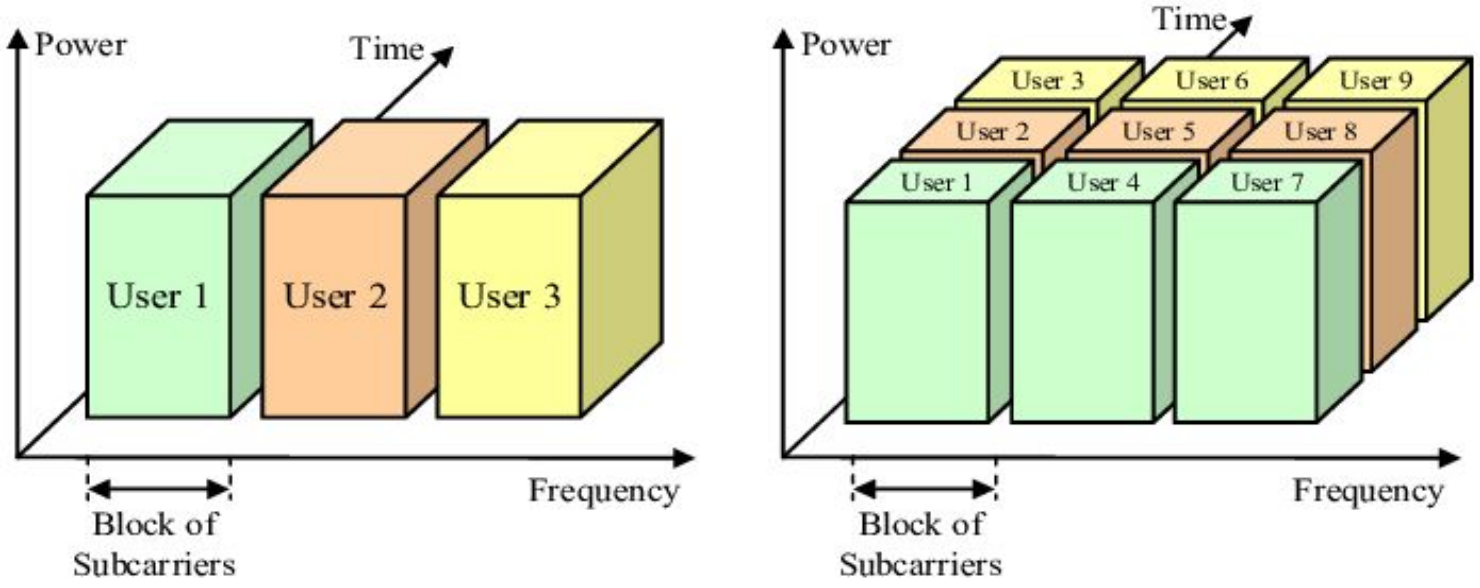
- **Objective** : efficiently distribute a single radio frequency band among a large number of users

- Achieved through "**cellular**" structure that allows the same frequencies to be reused multiple times

- **Handover** management: ensures continuous communication as users move between cells, without any interruption.

Cellular networks

- Frequency, time, and code **multiplexing**.
- Most often : a **combinaison** of these techniques.



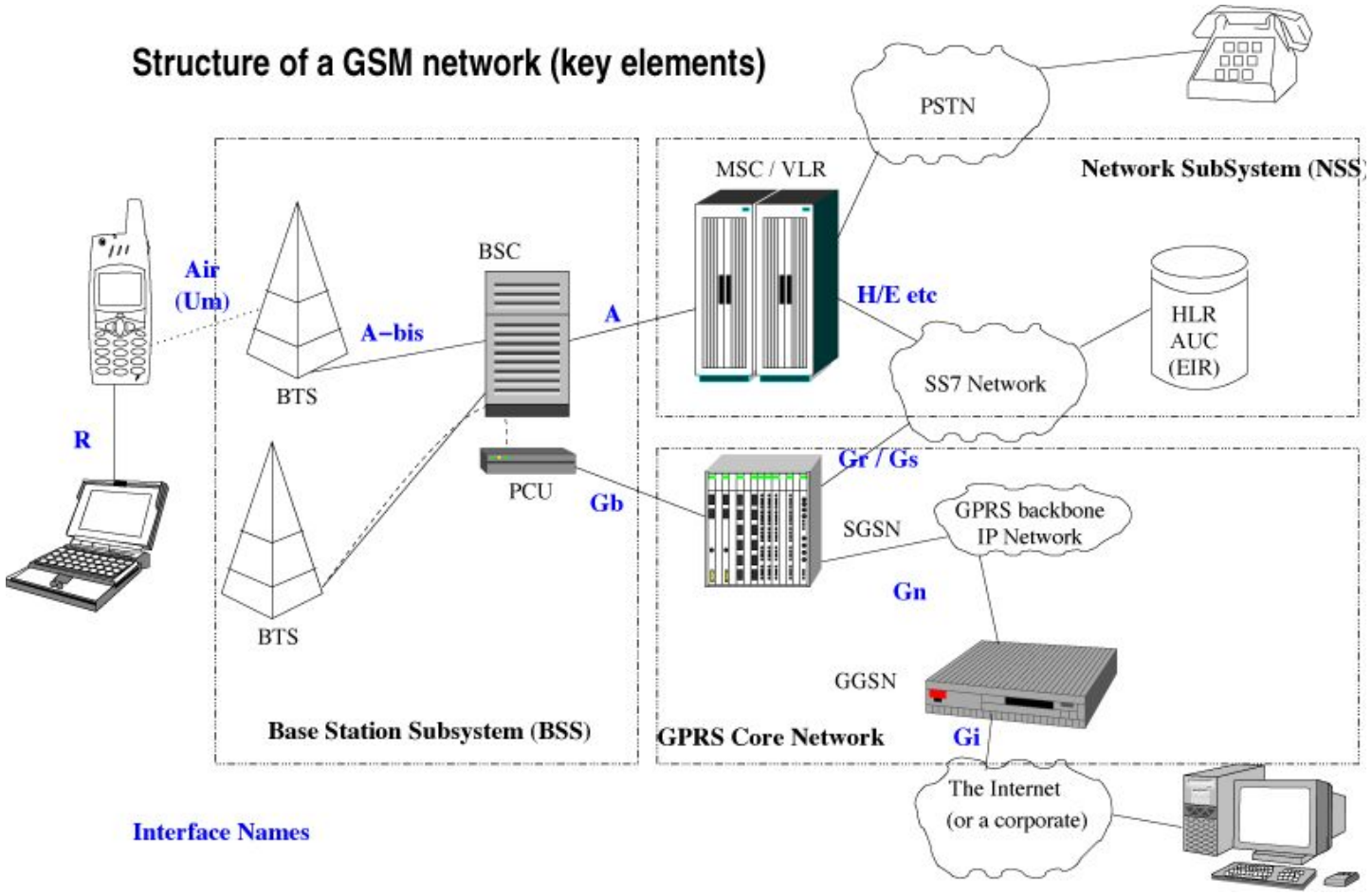
{OPEN}

Cellular networks

Mobile networks (2G, 3G et 4G) are based on a generally similar structure:

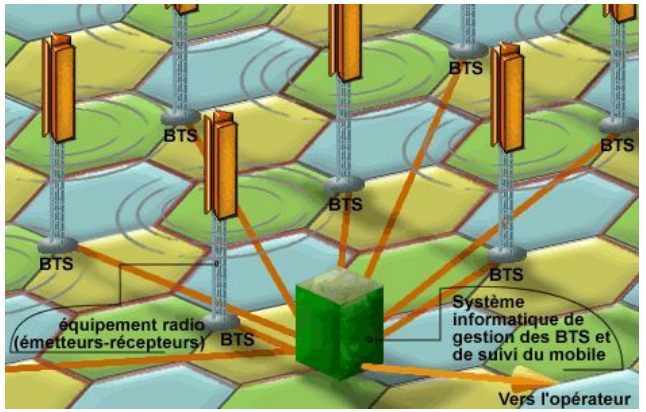
- A **Radio Access Network (RAN)**: This includes radio technology (notably antennas) and is responsible for transmitting information from the user (the person using their phone) to the core network.
- A **Core Network,** which handles:
  - The routing of user traffic to its destination
  - Cross-functional services (e.g., user identification, security, gateways to other networks, roaming, etc.)

2G network (GSM)

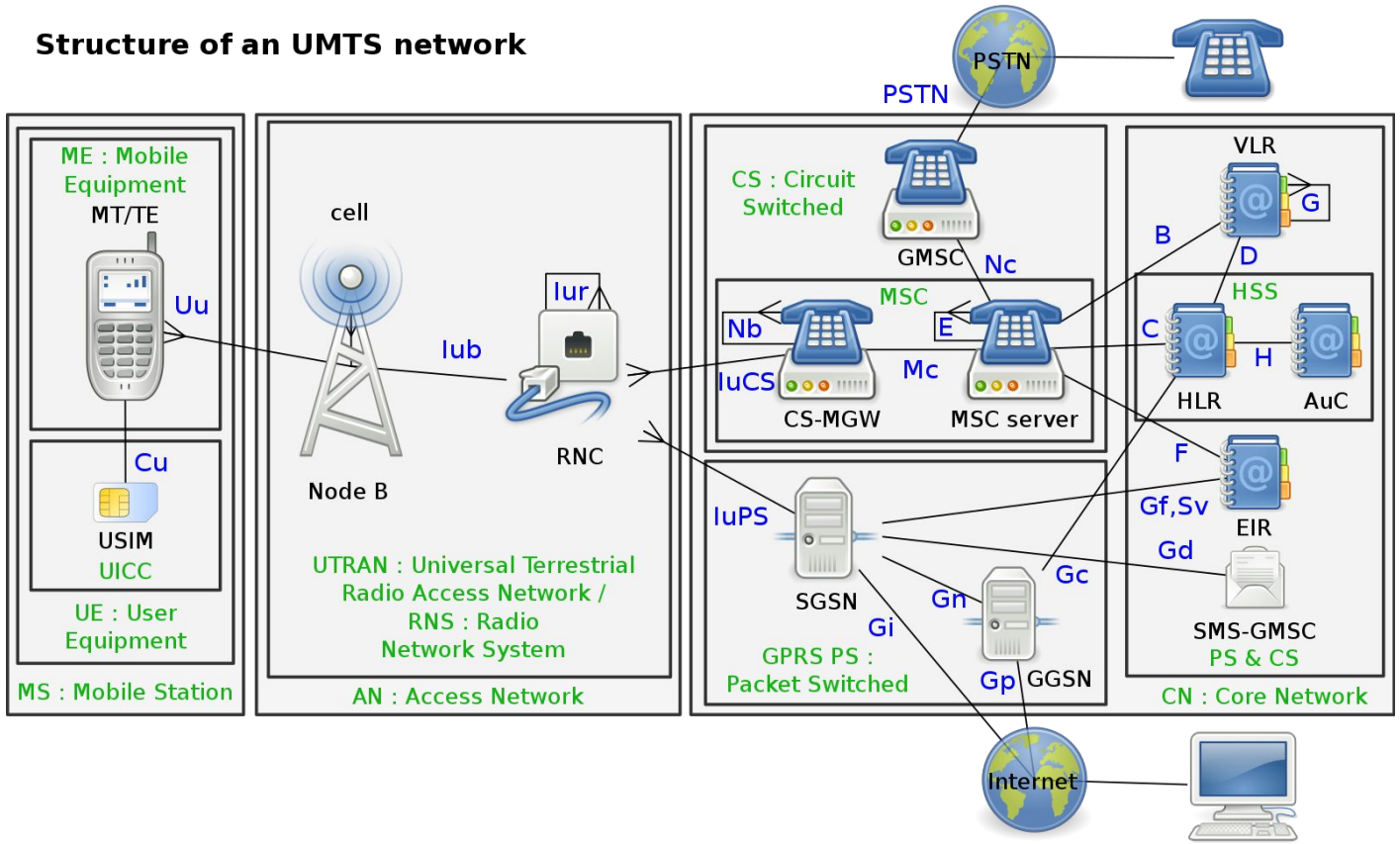


- Appeared in the 1990s, more reliable and efficient than 1G.
- It addresses only mobile telephony needs (SMS and calls, no data).

	Émission (en MHz)	Réception (en MHz)
Groupe 1	890-915	935-960
Groupe 2	1710-1785	1805-1880



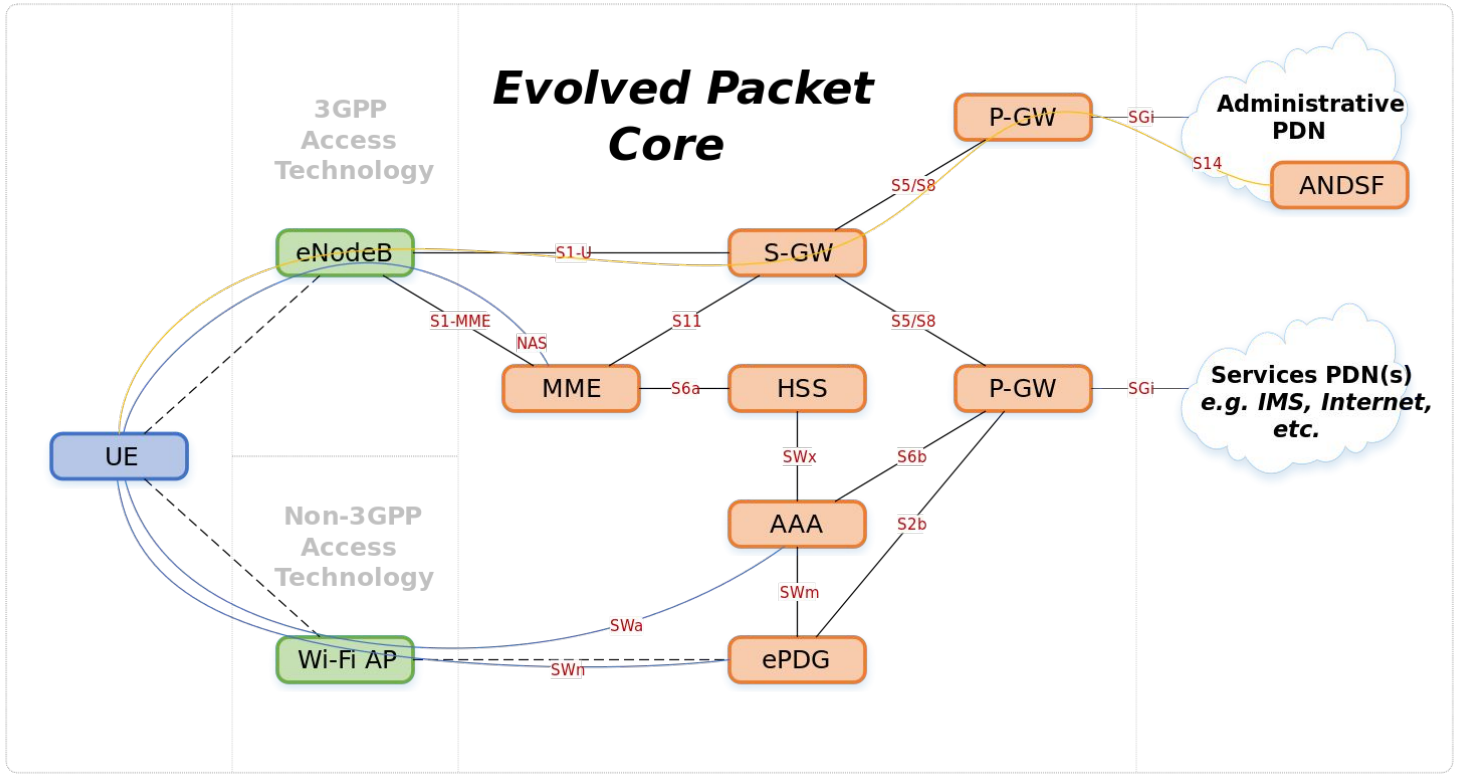
3G network (UMTS)



Major evolutions:

- Introduced the use of a **packet-switched network** (like the Internet) for data transmission, alongside the existing **circuit-switched network** used for voice calls.
- **Unlocked higher data speeds,** offering a mobile Internet experience comparable to that of fixed broadband.
- **Complete overhaul of the radio access network.**

4G network (LTE)



Major evolutions:

- The **radio access network**, known as e-UTRAN (Evolved UMTS Terrestrial Radio Access Network), is simplified to a single component called e-Node B (Evolved Node-B) — reducing latency and increasing resilience.
- **The core network**, called EPC (Evolved Packet Core), is fully IP-based (i.e., packet-switched). The circuit-switched network disappears, and voice communications are carried over IP, just like on fixed-line Internet.

Context for 5G

{OPEN}

5G Verticals

5G is not (only) a technology for the final user

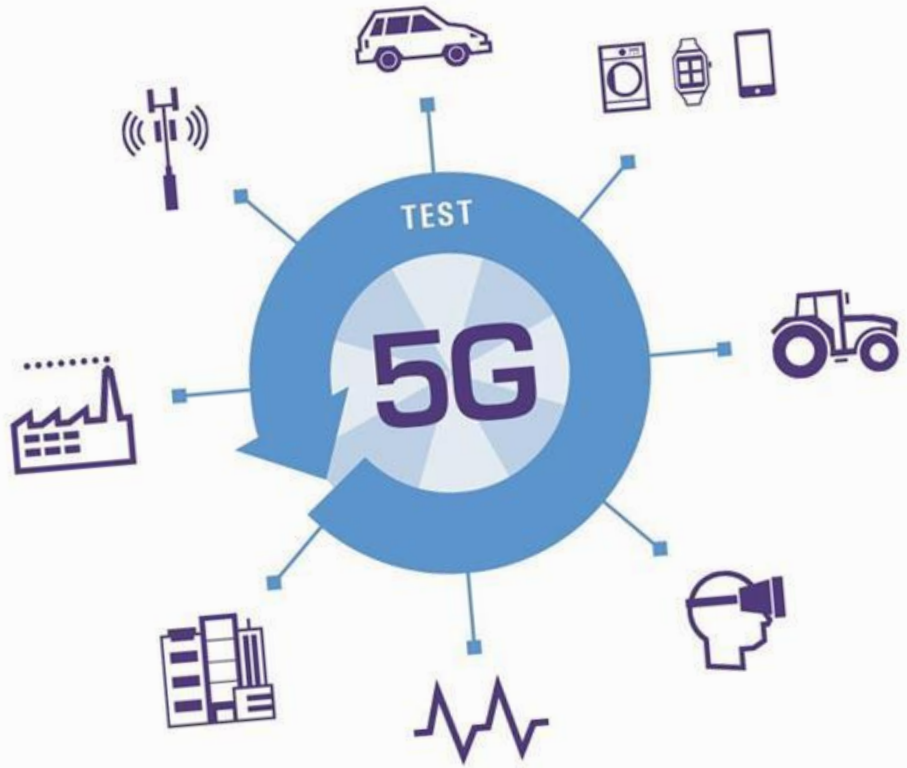
Platform for enabling diverse industries:

- Through **advanced virtualization**

Decoupling hardware from services

Run as software on general-purpose infrastruct
(NFV, SDN)

Enabling agility, flexibility, and multi-tenant environments



{OPEN}

5G Verticals

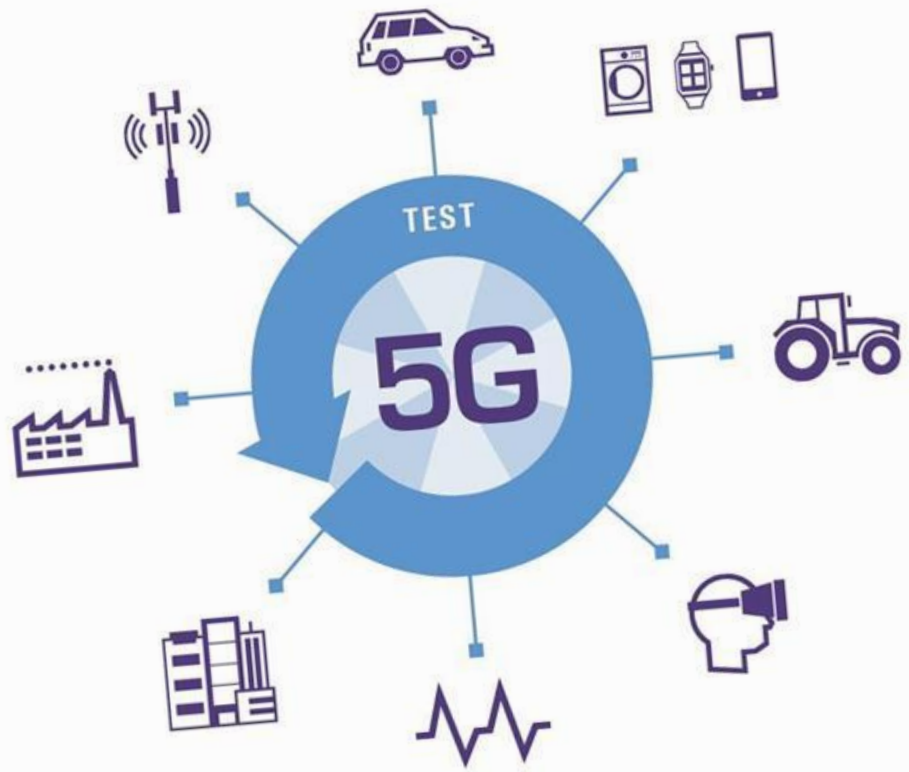
5G is not (only) a technology for the final user

Platform for enabling diverse industries:

- With "**network slicing**"

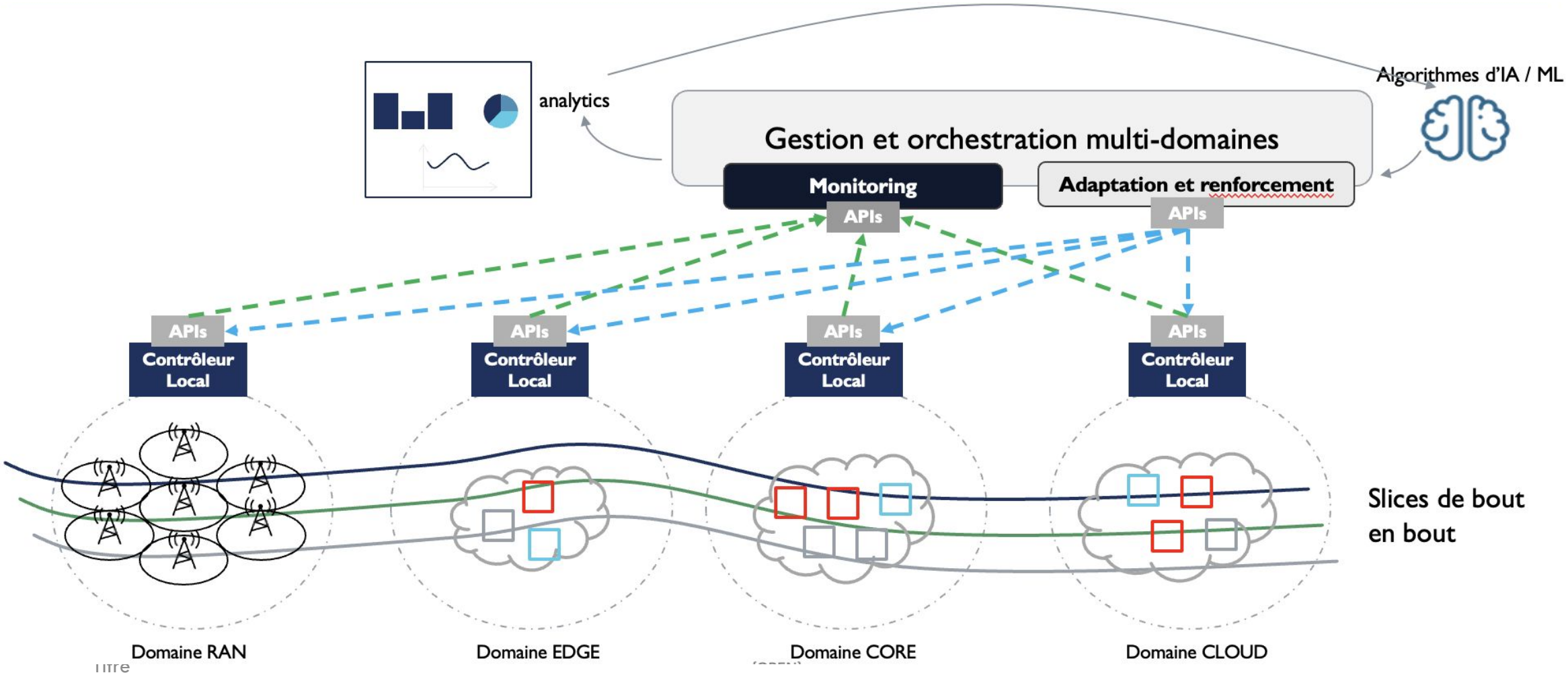
Creating custom virtual networks on the same physical infrastructure

Each "slice" is tailored for a specific use case: ultra-low latency, massive IoT, high throughput



{OPEN}

5G vision: virtualized and highly reconfigurable



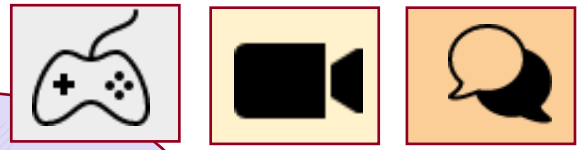
Beyond the diffusion of IP data

Controllers and orchestrators

Internet

Cybersecurity

Cyber and analytics



SDN and network virtualization

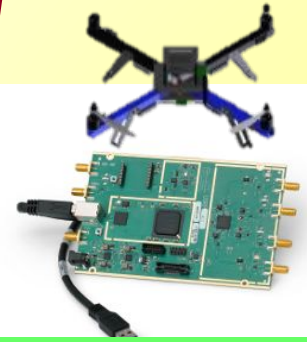
Edge server

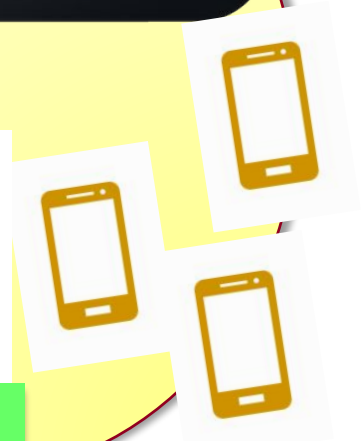
Standard HW

Edge server

Microservices and chaining

5G waveforms





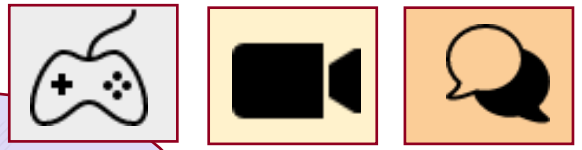
Virtualized edge PMR architectures

Beyond the diffusion of IP data

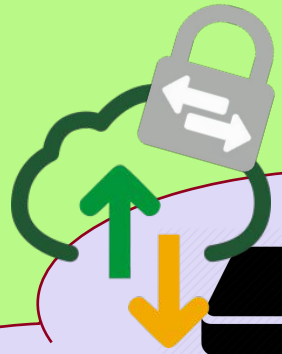
Controllers and orchestrators

Internet

Cyber and analytics



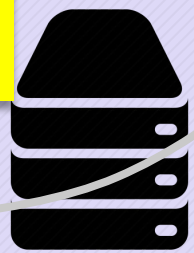
Cybersecurity

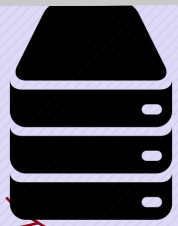


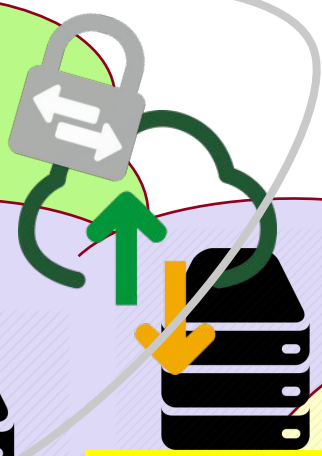
SDN and network virtualization

Edge server

Standard HW



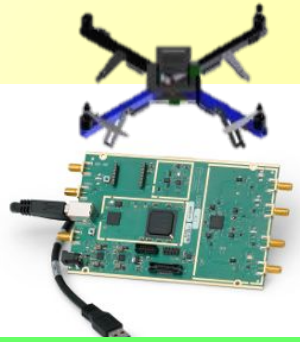




Microservices and chaining

Edge server

5G waveforms



Virtualized edge PMR architectures

Detection probes

Beyond the diffusion of IP data

Controllers and orchestrators

Internet

Cyber and analytics

Cybersecurity

SDN and network virtualization

Edge server

Standard HW

Edge server

Microservices and chaining

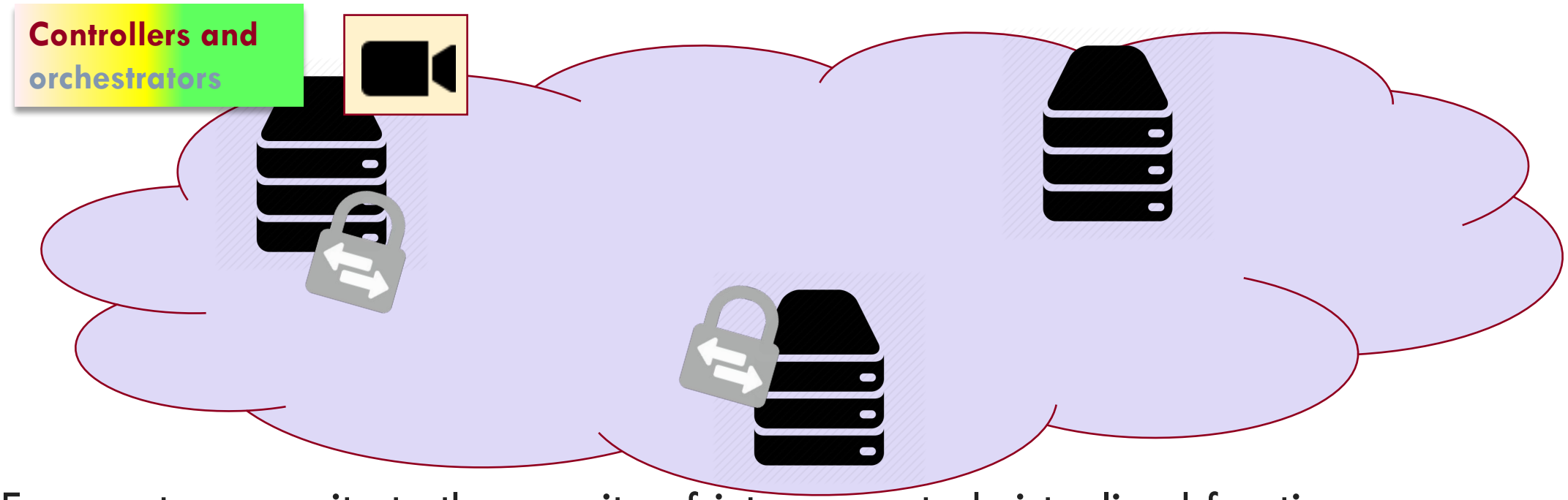
Provisioning and mobility

Configuration and scheduling

5G waveforms

Virtualized edge PMR architectures

Security in 5G networks



From system security to the security of interconnected virtualized functions:

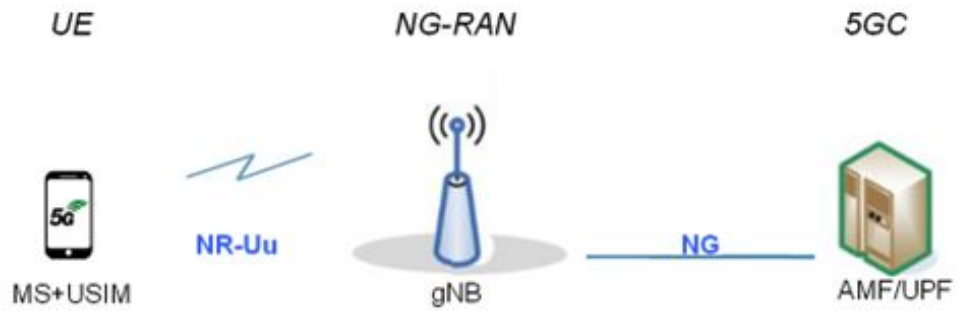
- Where should security functions be placed?
- How can access to functions be ensured for authorized users?
- How can end-to-end security be guaranteed when functions and services are separated?

5G network use case

{OPEN}

5G Network Architecture

- Reference is the **3GPP standard**: release 17/18
- Same elements as the previous generations:
  - A User Equipement (UE), itself composed of a Mobile Station and a USIM
  - The Radio Access Network (NG-RAN)
  - The Core Network (5GC)



Main entity is the **gNB** (base stations)

- **User Plane Function** (UPF): handling the user data
- **Access and Mobility management Function** (AMF): access the UE and the RAN in the signalling phase

{OPEN}

5G Network Architecture

- Reference is the **3GPP standard**: release 17/18

https://www.3gpp.org/technologies/5g-system-overview

- Same elements as the previous generations:
  - A User Equipement (UE), itself composed of a Mobile Station and a USIM
  - The Radio Access Network (NG-RAN)
  - The Core Network (5GC)

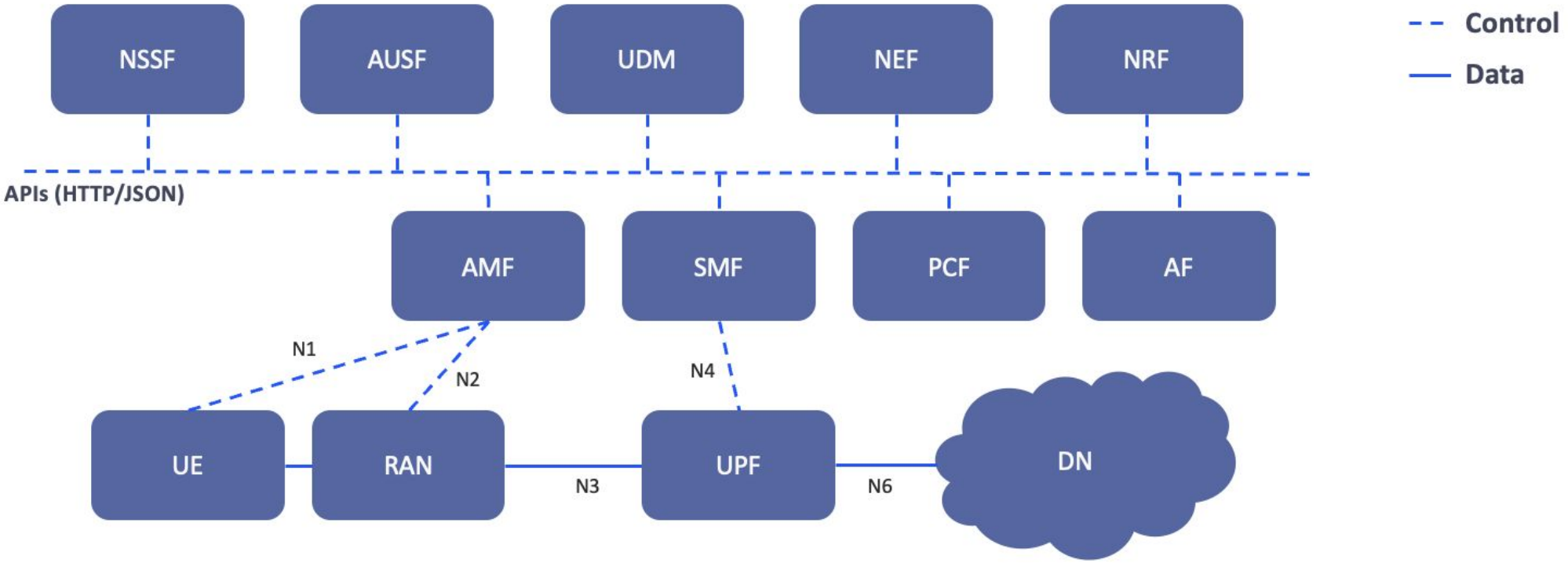
- 5G core **is Service-Based Architecture (SBA)**
- Virtualized Network Functions rather than "traditional" network entities
- Such architecture offers modularity and reusability

Network Functions

- The **UPF** and **AMF** previously described
- The **Session Management Function (SMF)**, that handles the calls and sessions, and contacts the UPF accordingly
- The **AUSF (Authentication Server Function)**, performing authentication in collaboration with the UDM.
- The **UDM (Unified Data Management)**, managing user subscription data and profiles
- The **PCF** (**Policy Control Function**), that controls that the user data traffic does not exceed the negotiated bearer(s) capacities
- The **Network Repository Function (NRF)**, which controls the other NFs and provides support for NF register, deregister and update service.
- The **Application Function (AF)**, controlling the application(s) (with possible involvement also in the user plane)
- The **Network Slice Selection Function (NSSF)**, assigning users to network slices based on their requirements.
- The **NEF (Network Exposure Function)**, providing secure access to network capabilities and events for external applications (APIs).

{OPEN}

Network Functions



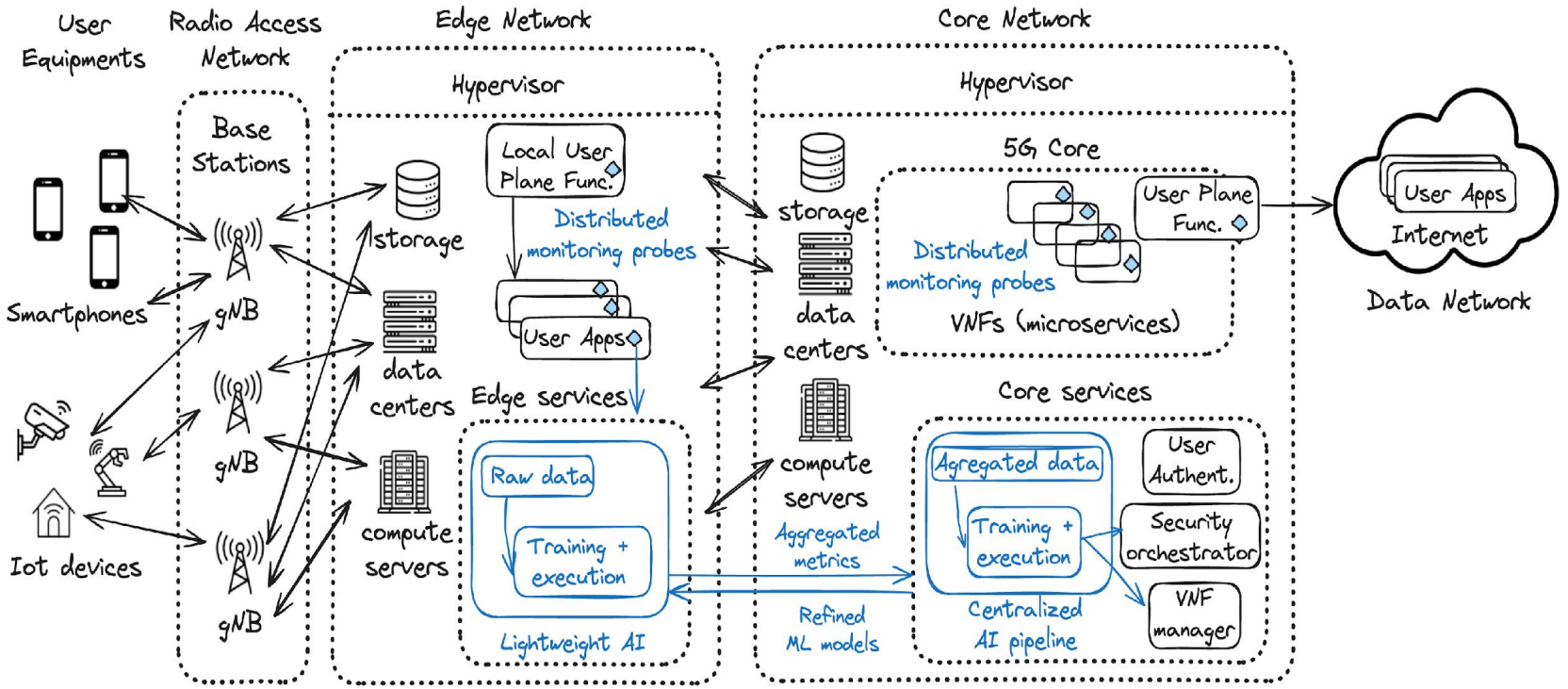
NSSF: Network Slice Selection Function
AUSF: Authentication Server Function
UDM: Unified Data Management
NEF: Network Exposure Function
NRF: Network Repository Function

AMF: Access & Mobility Management Function
SMF: Session Management Function
PCF: Policy Control Function
AF: Application Function

RAN: Radio Access Network
UE: User Equipment

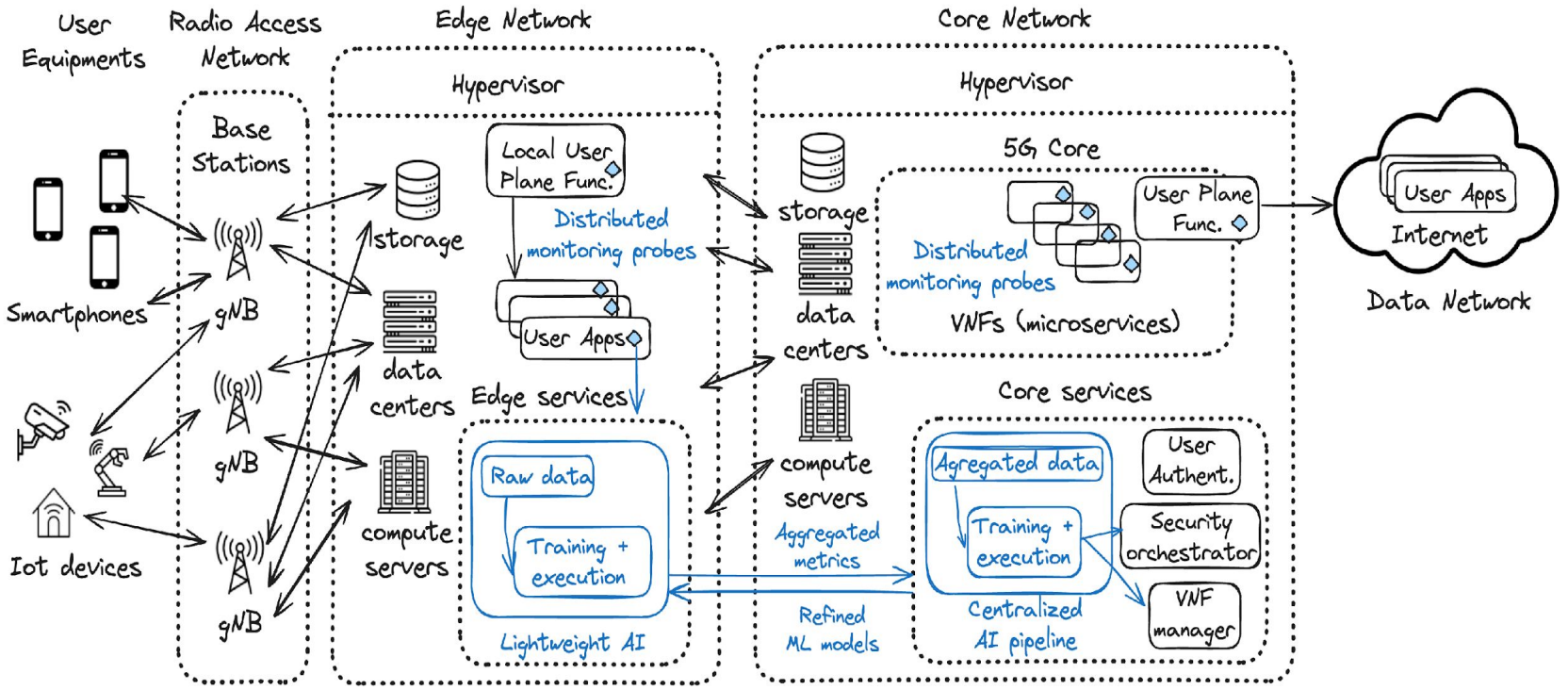
UPF: User Plane Function
DN: Data Network

5G Network Architecture



- **Edge**: edge servers to position services close to the users
- **Core** (control plane): standardized virtual network functions useful e.g., for authentication, managing user sessions, handling user mobility

5G Network Architecture

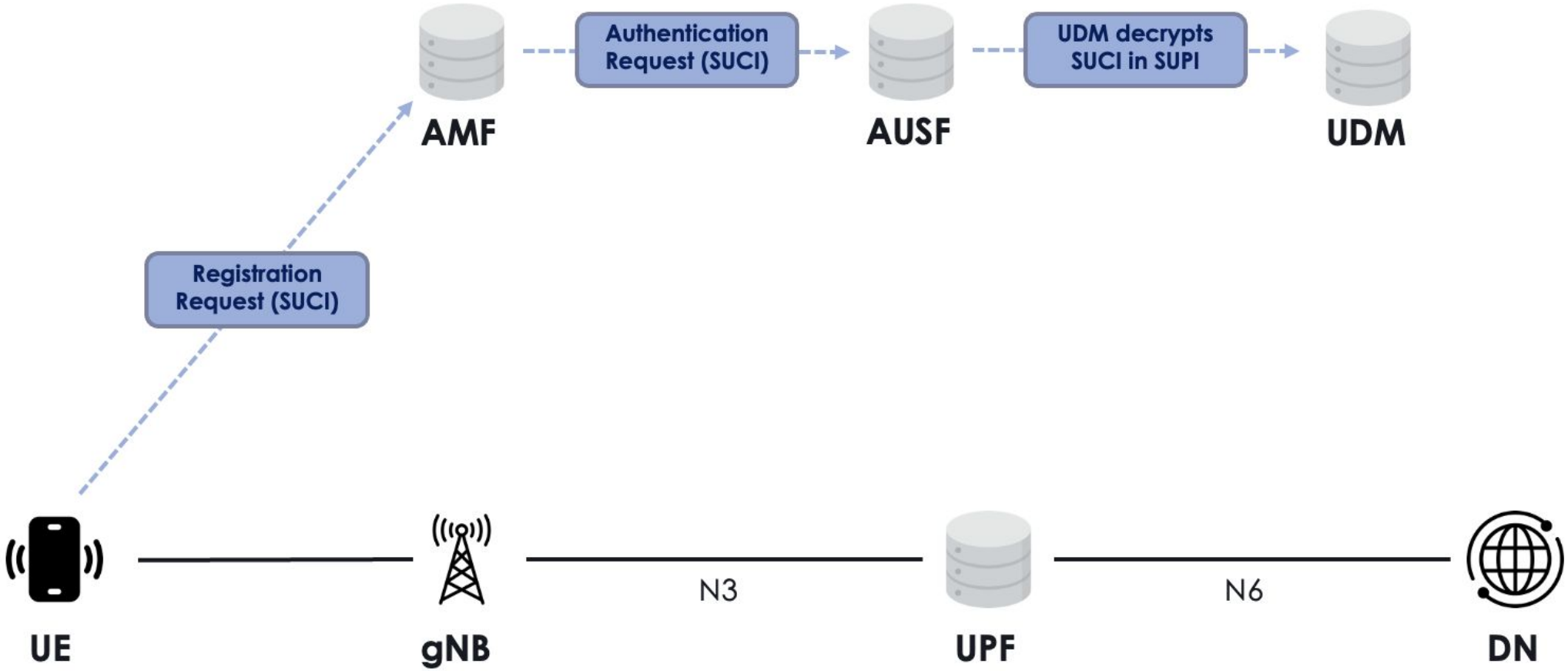


- 5G networks (edge & core) deployed by public telecommunication operators
- 5G tactical bubbles can also be deployed on-the-fly in disaster areas

→ **Mission critical systems**

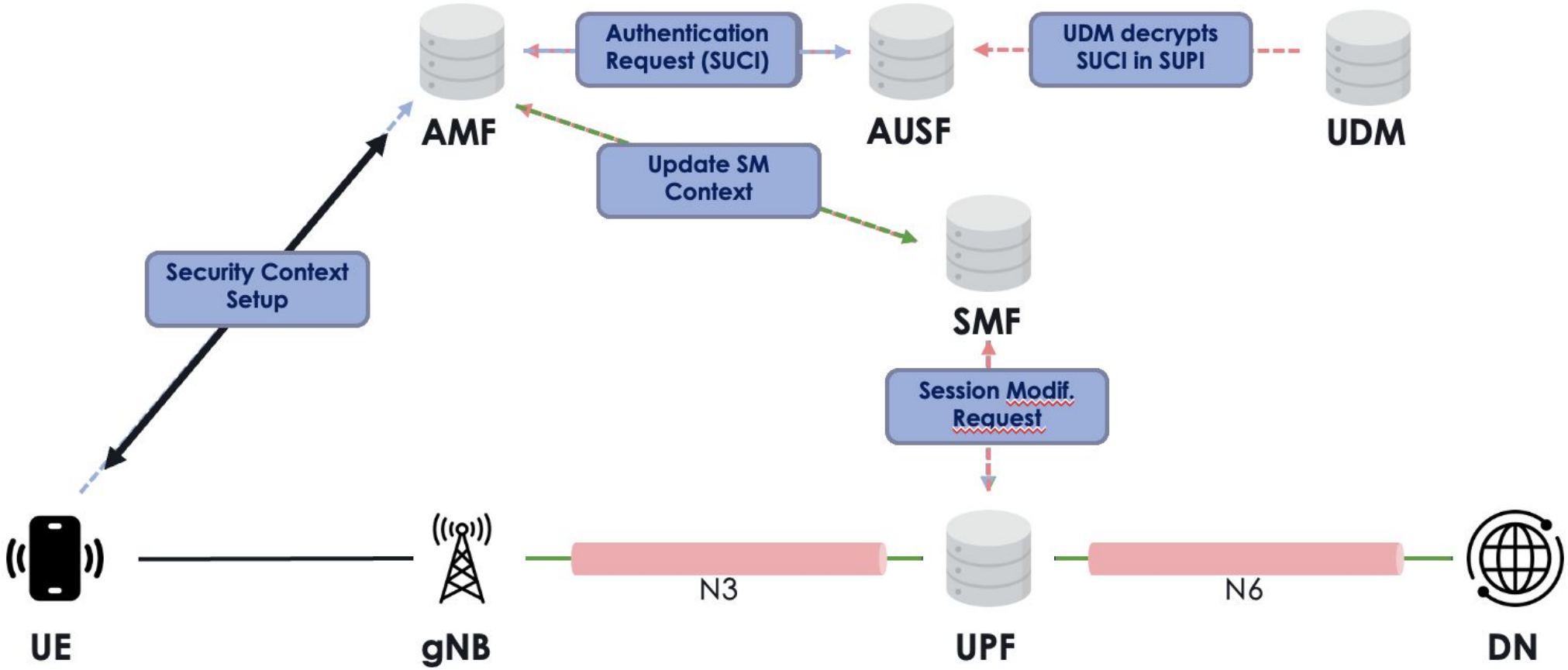
- Dynamic network reconfiguration
  - Resource orchestration
  - Security policies

5G Network Architecture – UE Registration



{OPEN}

5G Network Architecture – UE Registration



{OPEN}

Cybersecurity threats

{OPEN}

5G Network Threats

Service-Based Architecture: Virtual Network Functions deployed as microservices

→ Risk link to virtualised infrastructures with software isolation

→ Example: container escape, access to the host's machine resources, exhausting other tenants' resources

Network management & orchestration: attacks directed to orchestration functions leveraging on AI

→ Example: progressively injecting fake traffic to make the orchestrator scale up the infrastructure

5G core functions: attacks directed to 5G VNFs in the core network (5G APIs with HTTP protocol)

→ Example: HTTP injections (e.g., SQL injections, deploying malware), DDoS attacks, IMSI/SUCI catcher

{OPEN}

New 5G requirements: network slicing

- URLLC: Ultra-Reliable, Low-Latency Communication

/!\ challenges regarding **latency/availability of services and resources**

- eMBB: enhanced Mobile Broadband → need for extreme data rates

/!\ challenges regarding **traffic processing** for security (attack detection, AI/ML pipelines)

Need for distributed and scalable security solutions, deployed from the edge to the cloud

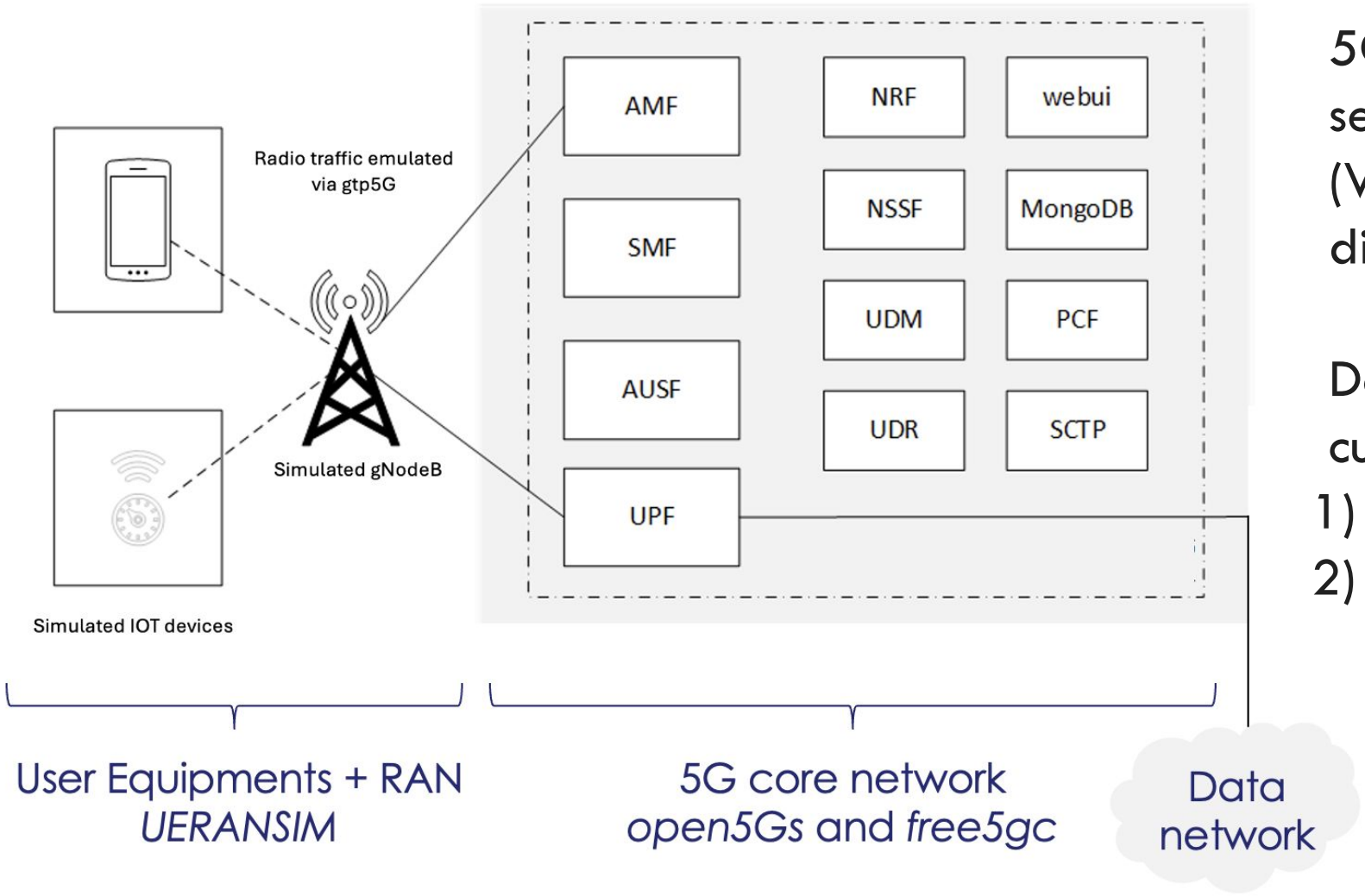
- umMTC: Ultra Massive Machine Type Communications

/!\ Internet of Things with very diverse capabilities (resource-constrained devices) with distributed AI/ML and privacy concerns

Configuration

{OPEN}

Testbed

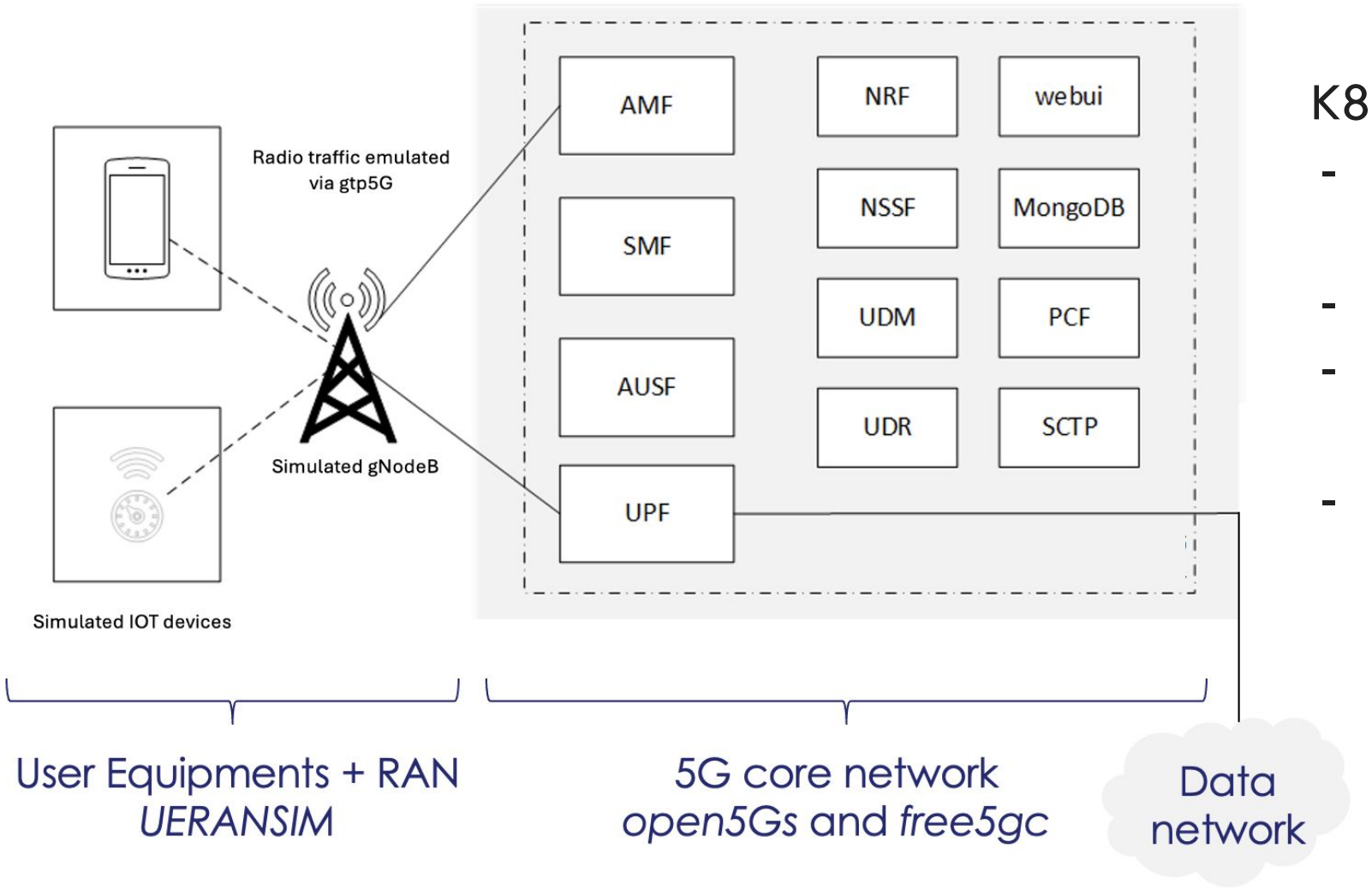


5G core network implementing several Virtual Network Functions (VNFs) running as Pods on different hosts

Development of 2 instances currently running on K8s:

1) Free5gc – 3GPP release 15+
2) Open5gs – 3GPP release 17

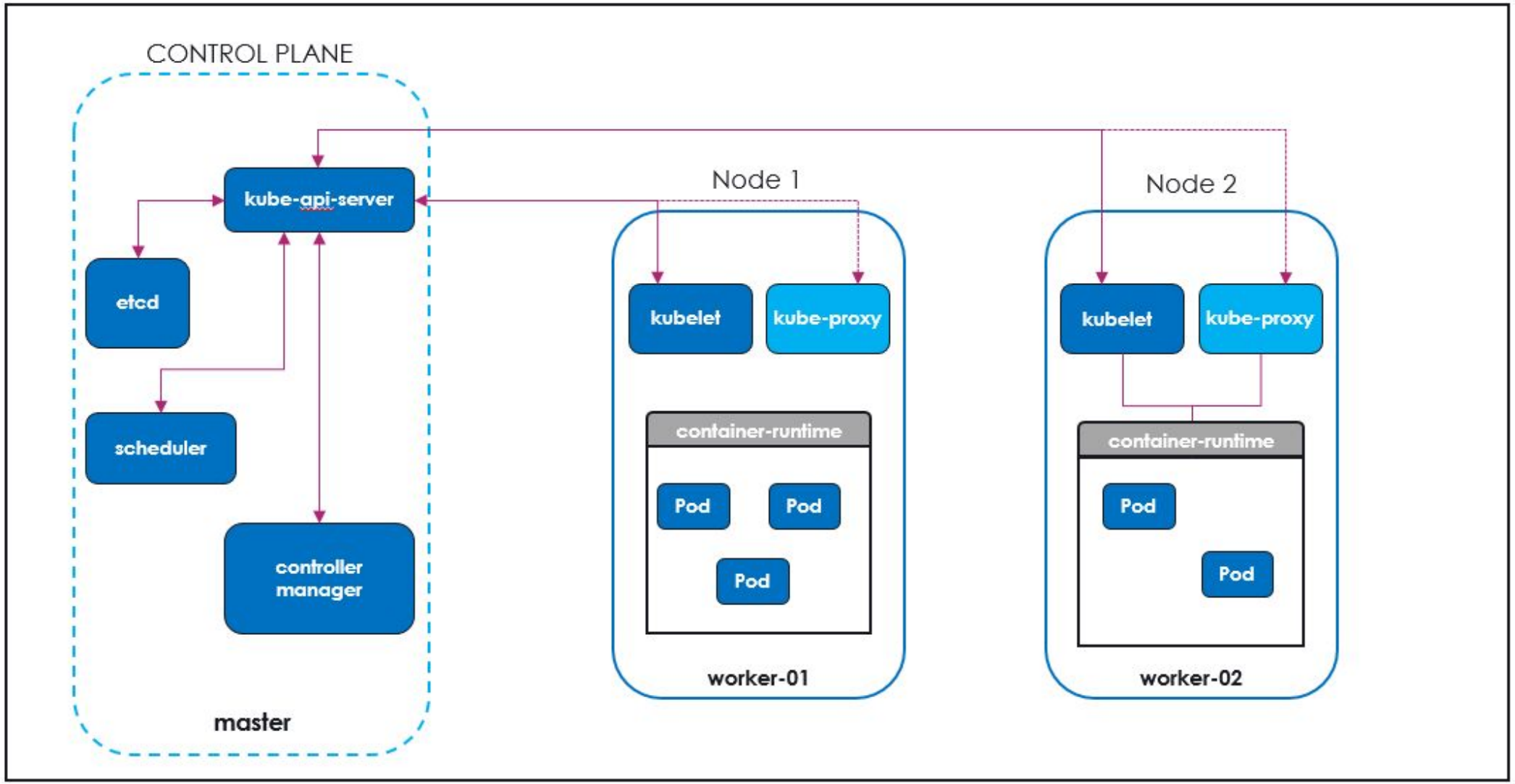
Testbed



K8s cluster contains:

- UERANSIM : gNodeB and UE devices
- 5G Core network
- Calico CNI and Multus framework
- Monitoring resources

Kubernetes cluster architecture



Attack scenarios

{OPEN}

Definition of different attack scenarios

1. **Virtualization** related scenarios

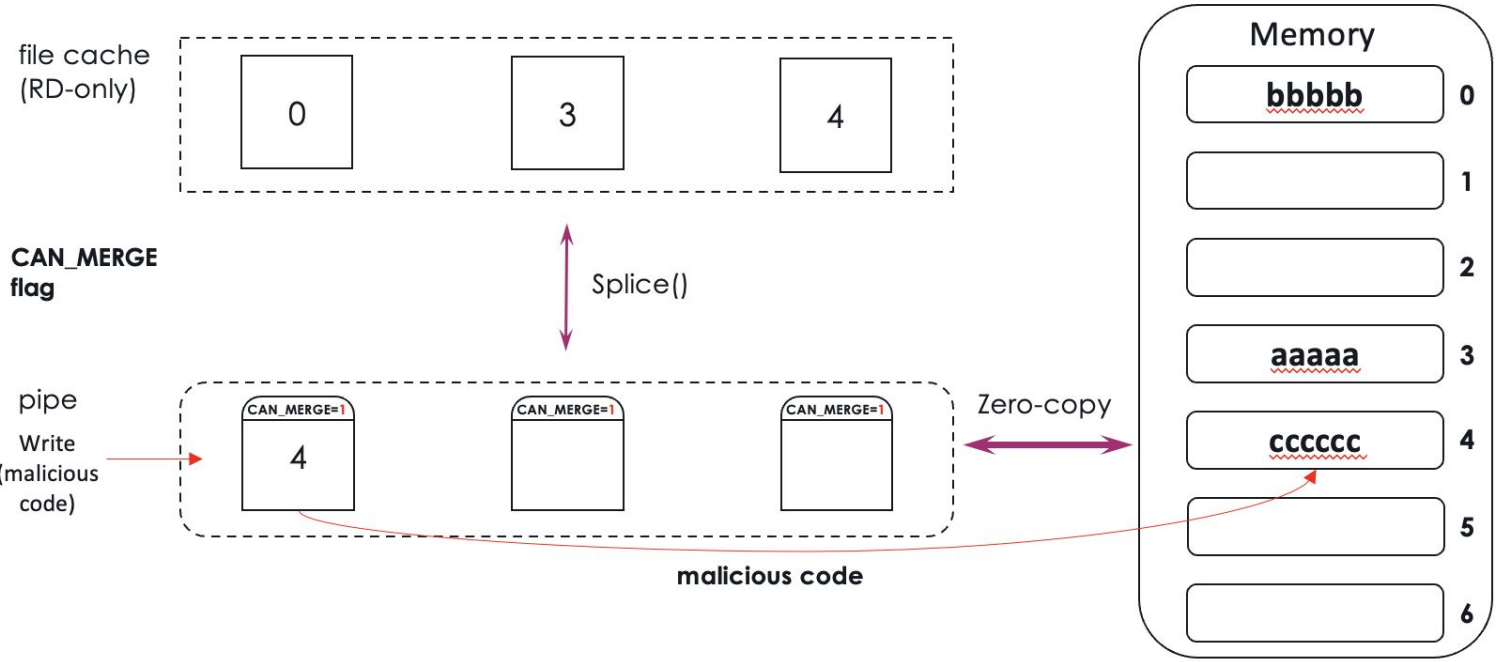
- Privilege Escalation & Container Escape
  - Dirty Pipe (CVE-2022-0847)
  - runC Overwriting (CVE-2019-5736)

- 2. **5G architecture**-related scenarios
  - Packet Replay/modification & API Injection/Patching
    - PFCP related attacks & NRF API

- 3. **Others**: reverse shell, sscans, denial of service, …

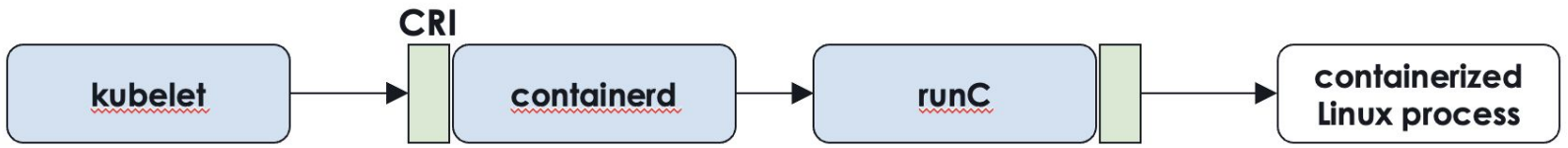
Virtualization related scenario

1) Privilege Escalation
DirtyPipe CVE
Can write into read-only files and elevate its privileges



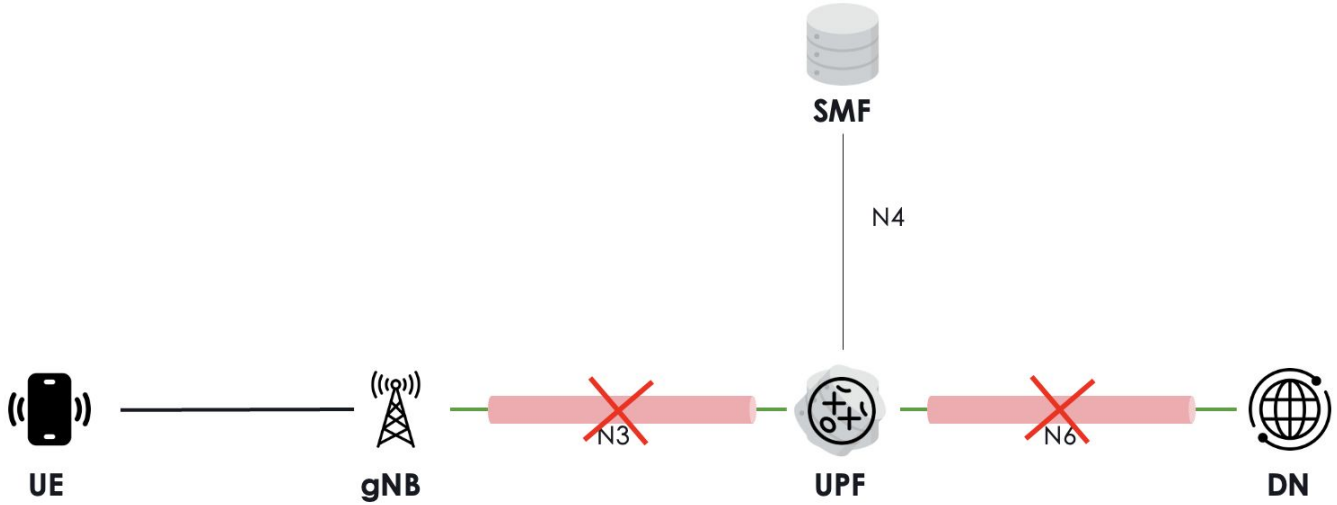
2) runC overwriting

Overwrite the runC binary, escaping the container by running a malicious process outside



PFCP attacks

- Denial-of-Service via PFCP flooding (see figure) – numerous packets sent to the UPF until the function crashed and network disruption for all users
- Denial-of Service via PFCP Session Deletion – SMF sending modification or deletion session requests to the UPF – causing network disruption for all users



{OPEN}

API injections

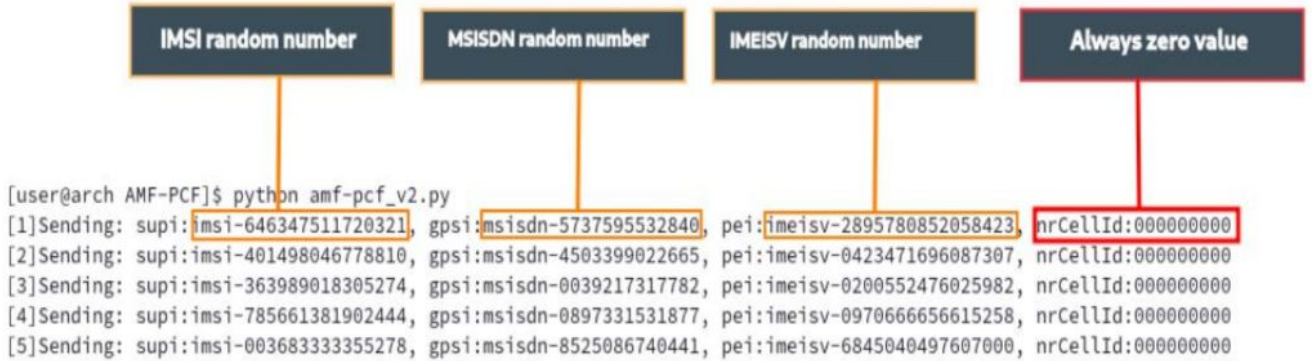
- API request from the Access and Mobility Function (AMF) to the Policy Control Function (PCF)

Contains information about the subscriber and the session so that PCF can decides the policy to apply

{"**notificationUri**":"http://10.85.245.37:8080/npcf-am-policy-control/v1.0.4/policies/imsi-460450301108147","**supi**": "imsi-460450301108147", "**gpsi**": "msisdn-3587003608347", "accessType": "3GPP_ACCESS", "**pei**": "imeisv-6045030110814701", "userLoc": { "nrLocation": { "tai": { "plmnId": { "mcc": "460", "mnc": "45" }, "tac": "001965" }, "ncgi": { "plmnId": { "mcc": "460", "mnc": "45" }, "**nrCellId**": "00008CCE0" } } }, "timeZone": "-05:00+1", "servingPlmn": { "mcc": "460", "mnc": "45" }, "ratType": "NR", "servAreaRes": { "restrictionType": "NOT_ALLOWED_AREAS", "areas": [{ "tacs": ["0003E8", "0003E9", "0003EA"] }, { "tacs": ["0003E8", "0003E9", "0003EA"] }] }, "guami": { "plmnId": { "mcc": "460", "mnc": "45" }, "amfId": "010041" }, "suppFeat": "0000000000000000" }

- Can send falsified parameters to the PCF → e.g., wrong location (nrCellId) to disable services that rely on location data
- Or request for session deletion

curl -X POST "http://10.98.211.58/nsmf-pdusession/v1/sm-contexts/urn:uuid:bec1d7cd-4d3e-40bc-bee4-3f585a063d60/release"



{OPEN}