Virtual ICN-based IoT networks with vICN

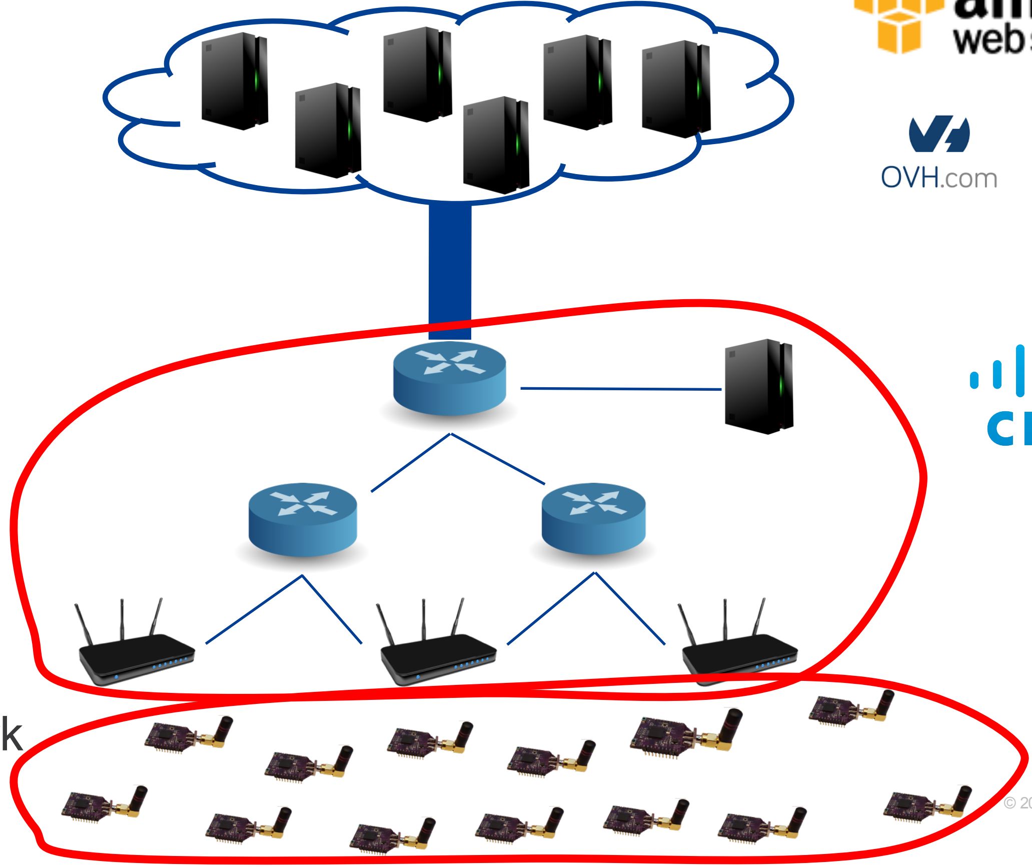
Marcel Enguehard
Software Engineer & PhD Candidate
RIOT Summit 2017, Berlin, Germany

RIOT in the IoT vertical

Cloud

Access

Sensor network



amazon web services

OVH.com

Google Cloud Platform

cisco

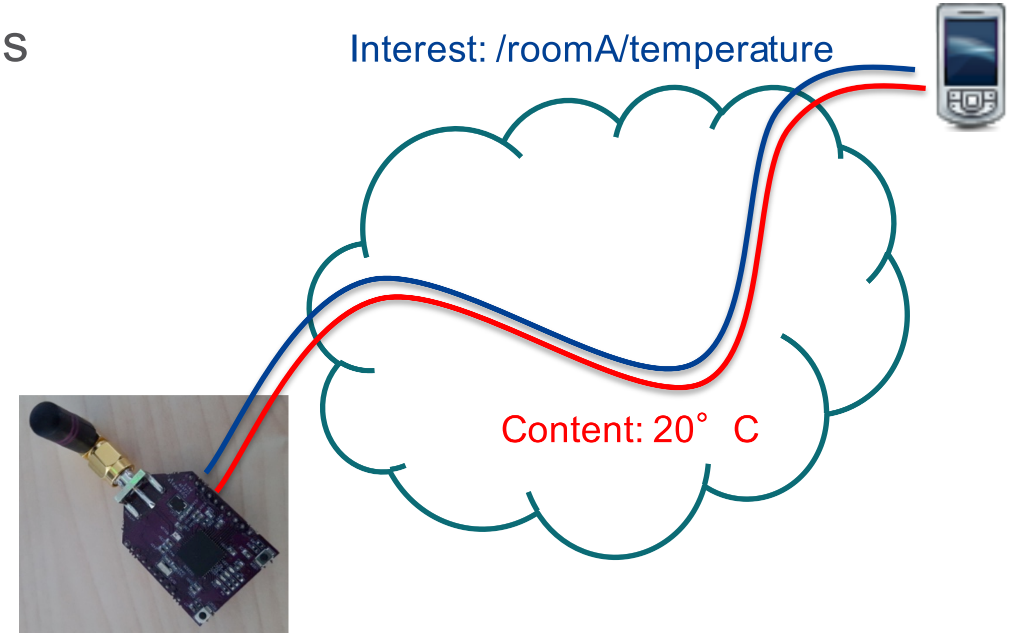
RIOT

How do we handle this huge amount of data in the access network?

Information Centric Networking

- Each retrievable content is named
- Name-based routing
- Pull based model w/ symmetric routing
- Every node is a cache

User application



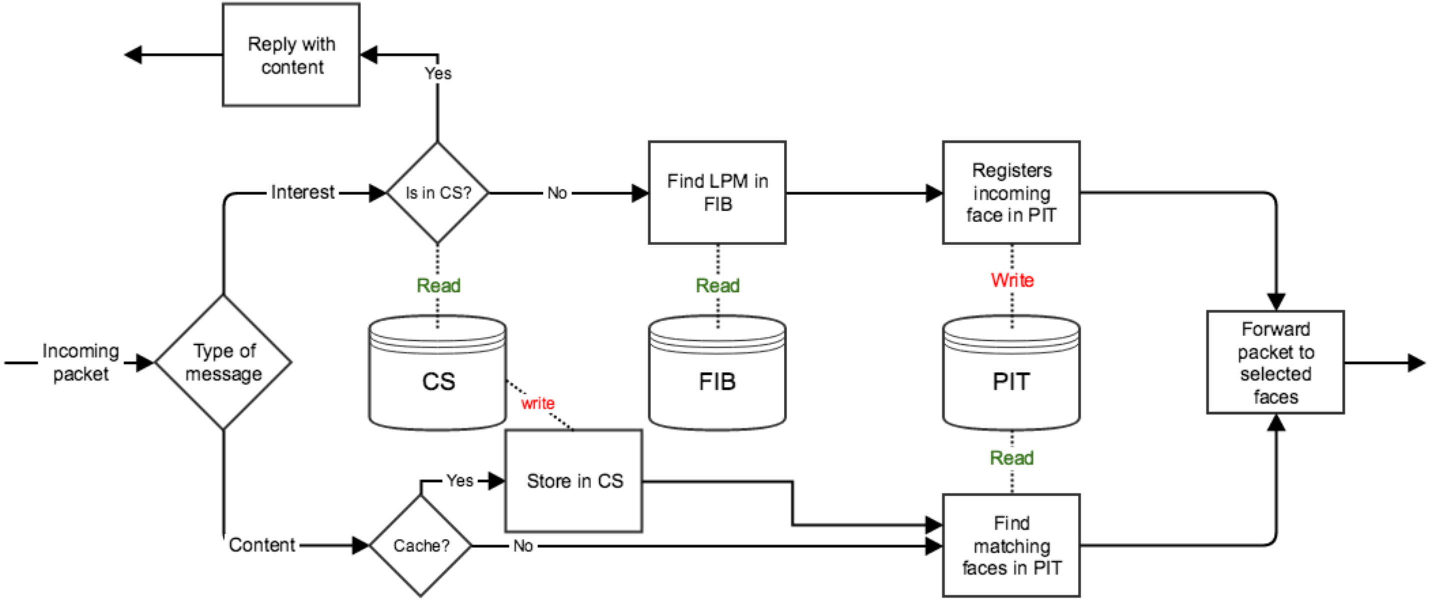
Interest: /roomA/temperature

Content: 20° C

Sensor board

Vanilla ICN Forwarding

- Forward to “faces”
- 2 modules: Forwarding Interest Base (FIB) and Pending Interest Table (PIT)
- Interest → LPM in FIB
- Content → Symmetric routing through PIT



ICN for the IoT

- For the devices
  - Reduced stack size (Bacelli et al., ICN14)
  - Accommodate low duty cycles (Hahm et al., ICN17)
- For the access
  - Consumer and producer mobility handling (Augé et al., ICN15)
  - Native multicast (Samain et al., IEEE TMM 2017)
- For data processing
  - Independence from compute location
  - Aggregation of request

We need to test our verticals before deploying them in production

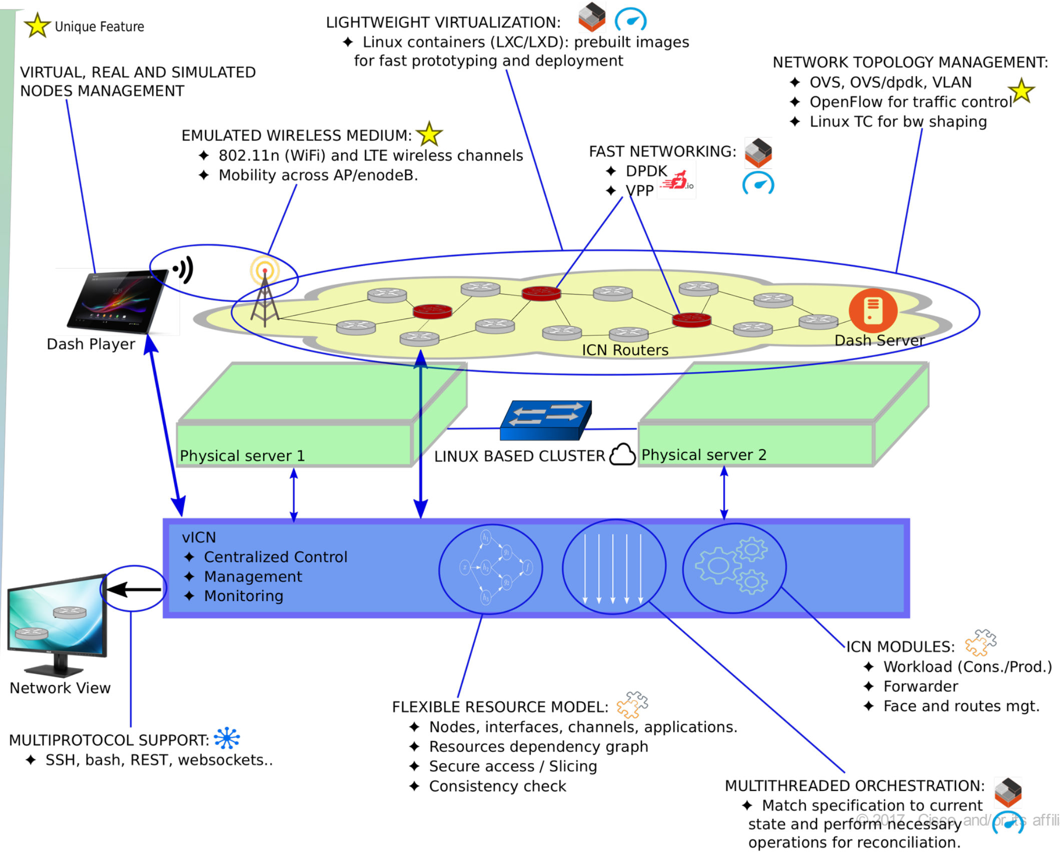
Challenges

- Large number of devices
- Mobility & traffic patterns
- Unified API for traffic generation, network management, etc

vICN

Design Principles

- Scalability
- Modularity
- Performance
- Distributed Computing
- Centralized Control



☆ Unique Feature

VIRTUAL, REAL AND SIMULATED NODES MANAGEMENT

LIGHTWEIGHT VIRTUALIZATION:
- ✦ Linux containers (LXC/LXD): prebuilt images for fast prototyping and deployment

EMULATED WIRELESS MEDIUM: ☆
- ✦ 802.11n (WiFi) and LTE wireless channels
- ✦ Mobility across AP/enodeB.

NETWORK TOPOLOGY MANAGEMENT:
- ✦ OVS, OVS/dpdk, VLAN
- ✦ OpenFlow for traffic control ☆
- ✦ Linux TC for bw shaping

FAST NETWORKING:
- ✦ DPDK
- ✦ VPP

vICN
- ✦ Centralized Control
- ✦ Management
- ✦ Monitoring

ICN MODULES:
- ✦ Workload (Cons./Prod.)
- ✦ Forwarder
- ✦ Face and routes mgt.

MULTIPROTOCOL SUPPORT:
- ✦ SSH, bash, REST, websockets..

FLEXIBLE RESOURCE MODEL:
- ✦ Nodes, interfaces, channels, applications.
- ✦ Resources dependency graph
- ✦ Secure access / Slicing
- ✦ Consistency check

MULTITHREADED ORCHESTRATION:
- ✦ Match specification to current state and perform necessary operations for reconciliation.

vICN objectives

Programmability

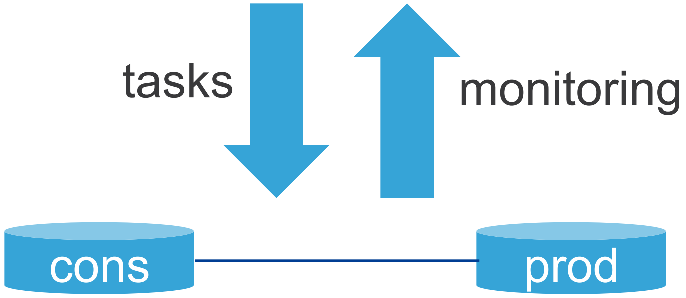
Reliability

Scalability

vICN resource model

- Intent based-framework
- Object-based model
- State reconciliation between model and deployment

```
cons = LxcContainer()
prod = LxcContainer()
link = Link(src=cons, dst=prod)
```



vICN resources

Class

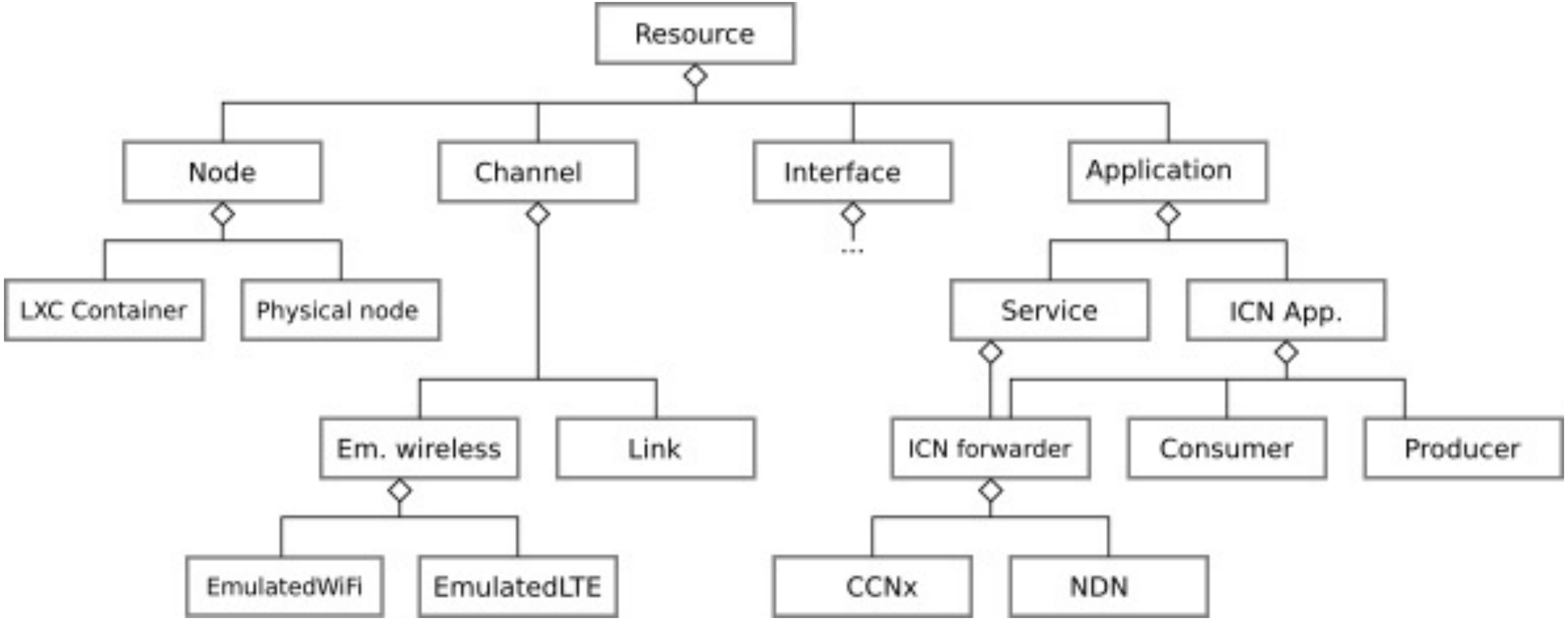
- Virtual representation of deployment element
- Node, forwarder, application, link, etc.
- Described by *attributes*

Class members

Example resource: Forwarder

- Represents an ICN forwarder
- Attributes:
  - node
  - cache_size
  - cache_policy (e.g., LRU)
  - log_file
  - etc.

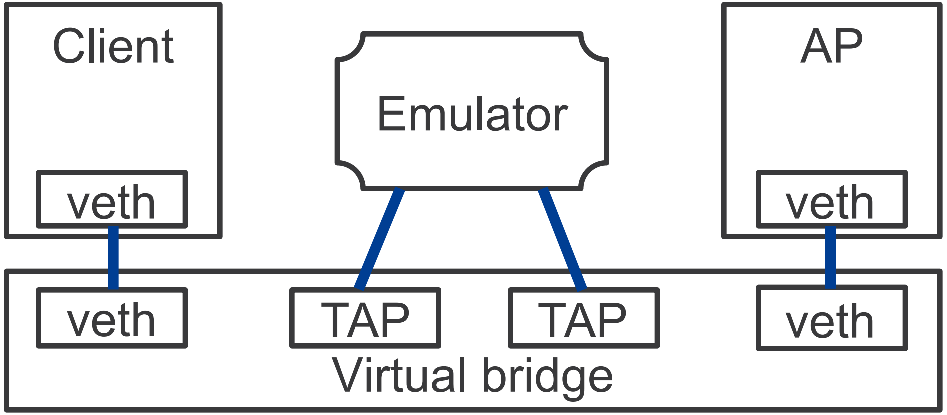
vICN resources dependencies



Wireless emulators

- Currently implemented: WiFi, LTE
- Soon 802.15.4?

```
Emulated802154Channel(
         ap=AP,
         stations=[client],
         node=server
)
```



Monitoring through resource model

- Python model is used for both probing and actions
- Can be use to monitor network status and performances, handle errors, etc

vICN and RIOT

- RIOT as a resource in vICN
- Link emulation
- Foster research on full IoT vertical

Conclusion

- vICN is a programmable and efficient framework for network virtualization
- Complementary with RIOT for IoT experimentation