Software-Defined Networking

– *introductory*

Carolina Fernández

Bio

- Carolina Fernández
- R&D Engineer
- Working on networks, virtualisation, automation
  - SDN, NFV applied to MEC, 5G, security, ...
- More interests: privacy *et al*

CarolinaFernandez
carolinafernandez.github.io

cfermart

Agenda

1. Networking basics
2. Virtual networking
3. Software-Defined Networking
4. Recap

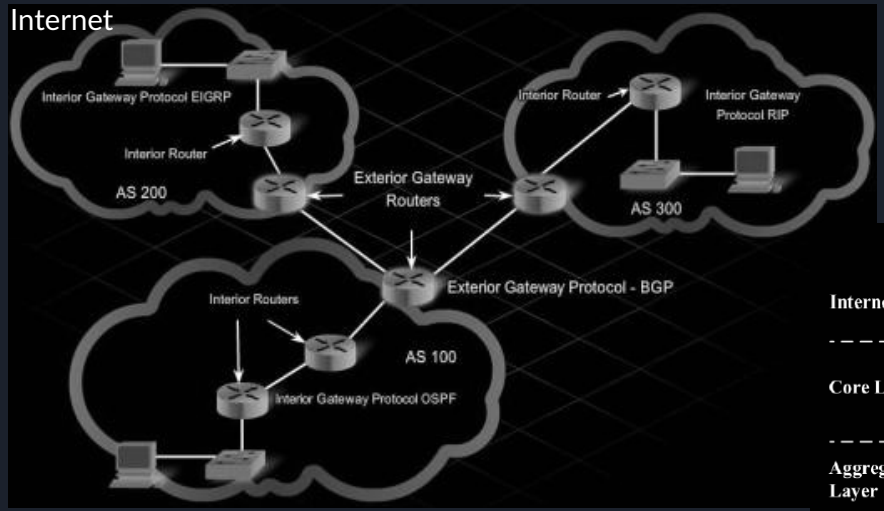
Networking basics

Networking (1): overview

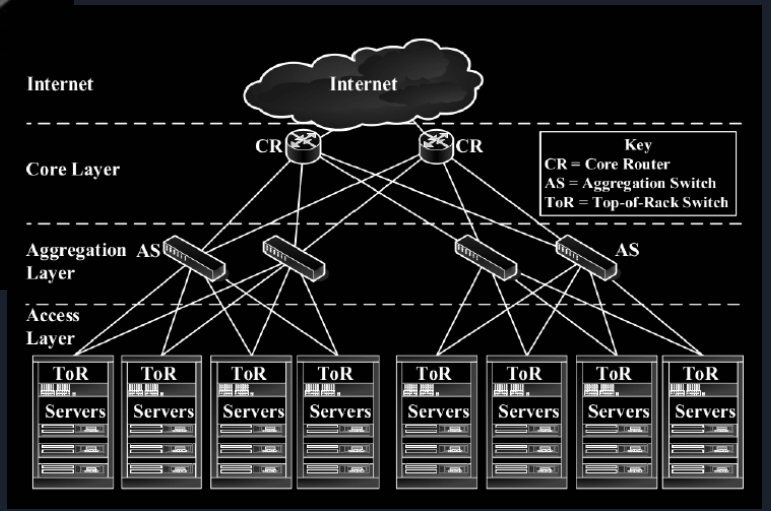
Source: https://9gag.com/gag/4527267

Networking (1): overview (Internet & DCs)

Autonomous Systems watching over the Internet



Internet/Core/Aggregation layers in DC

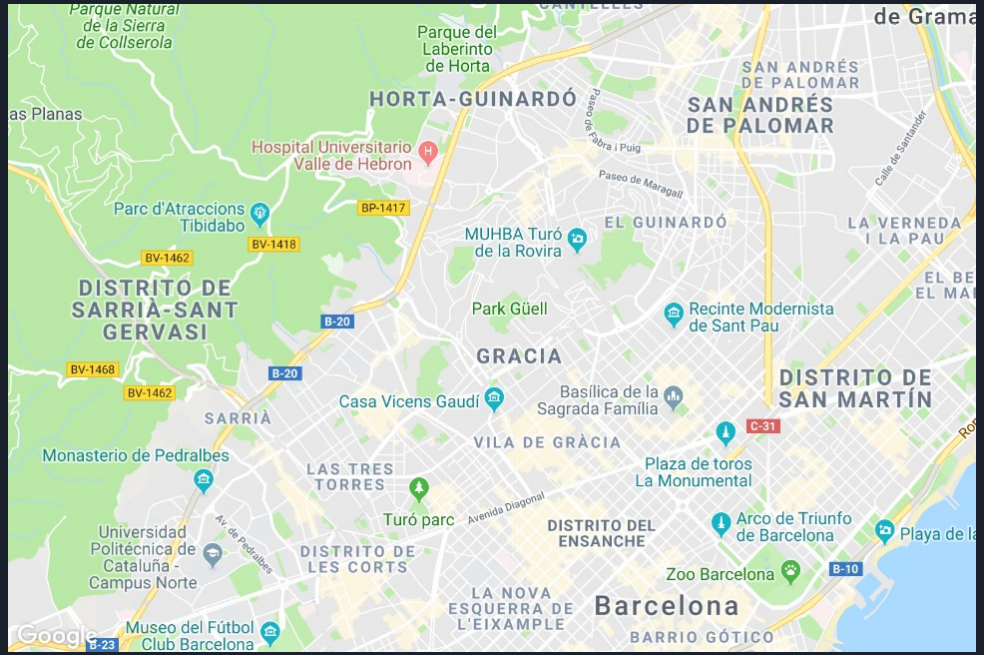


Source: https://krystalchisholm.wordpress.com/2010/11/24/chapter-16/, https://www.grotto-networking.com/BBNetVirtualizationDataCenter.html

Networking (2): layers and stacks

From lower to upper layers:

- **Physical layer**: fixed set of circuits that can be used

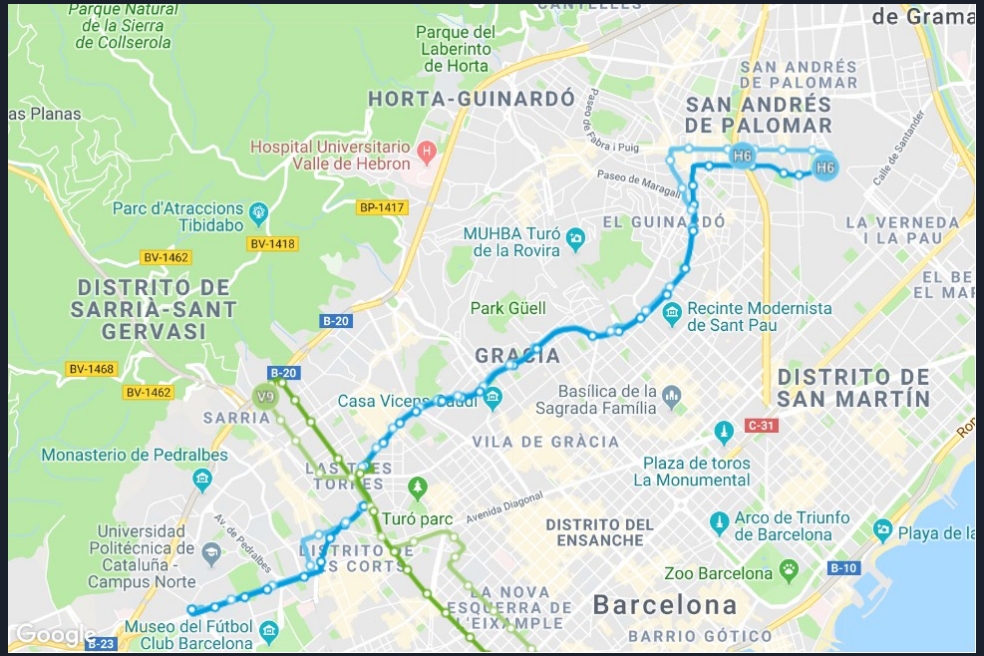


Source: https://www.tmb.cat/es/transporte-barcelona/mapa/bus

Networking (2): layers and stacks

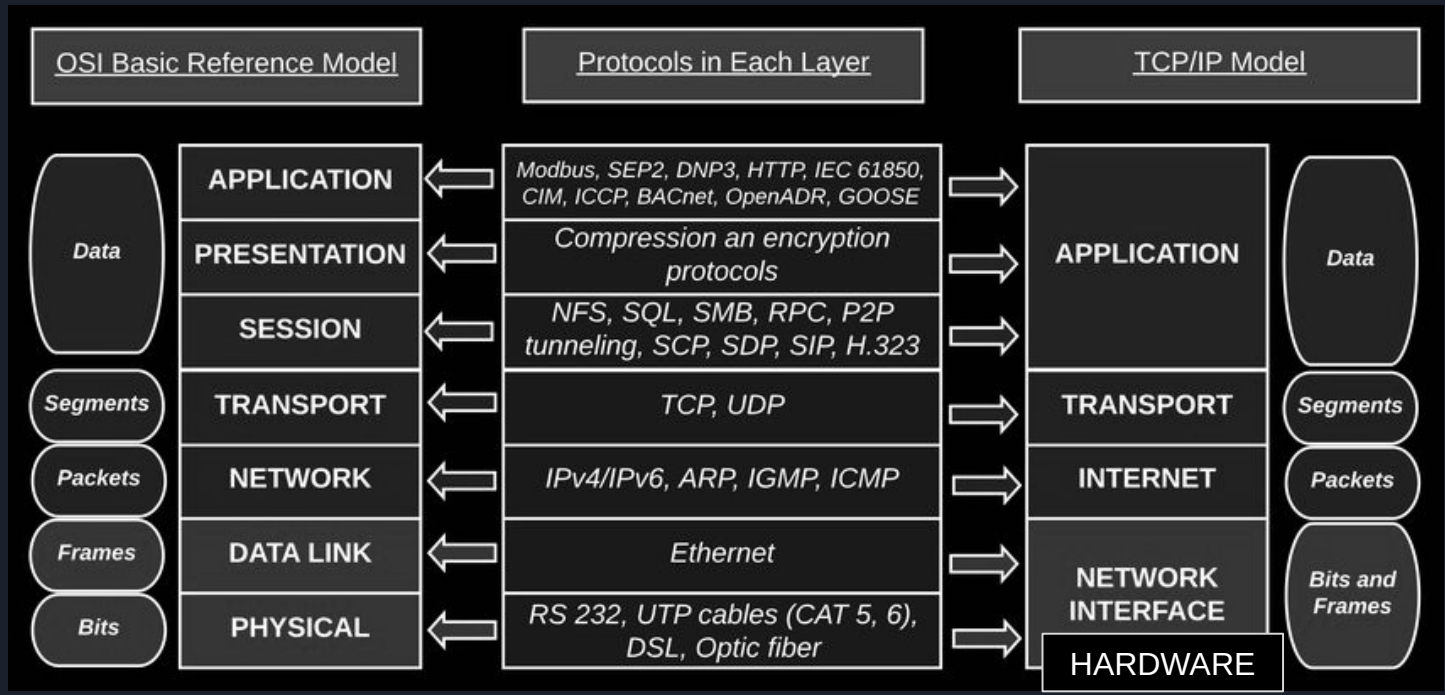
From lower to upper layers:

- **Physical layer**: fixed set of circuits that can be used
- **Upper layers**: define how to transmit information (specific per *layer*) and in which ways (*protocol*)
  - Stack
    - OSI
    - TCP/IP



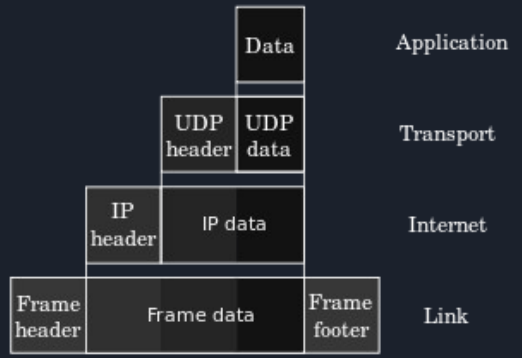
Source: https://www.tmb.cat/es/transporte-barcelona/mapa/bus

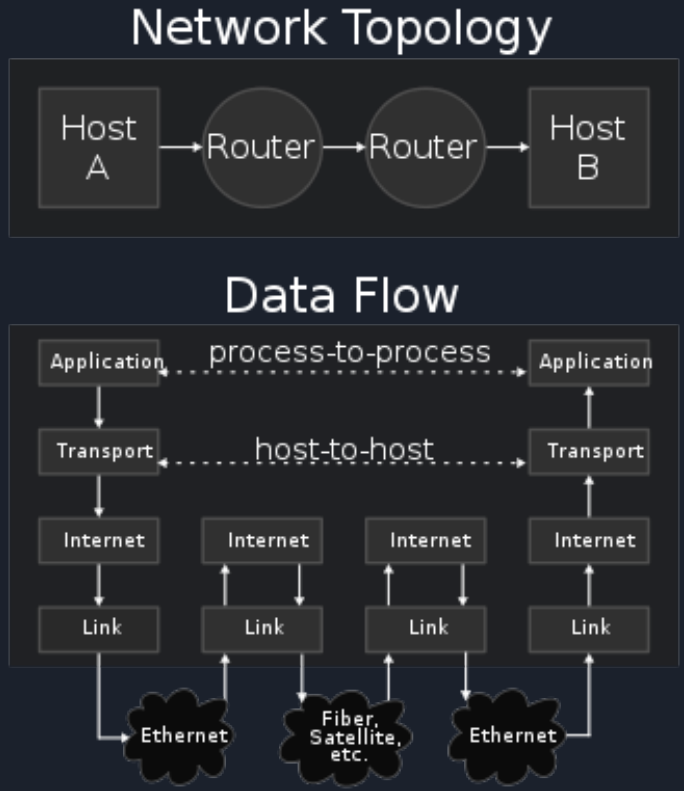
Networking (2): layers and stacks



Source: https://www.researchgate.net/figure/The-logical-mapping-between-OSI-basic-reference-model-and-the-TCP-IP-stack_fig2_327483011

Networking (2): layers and stacks





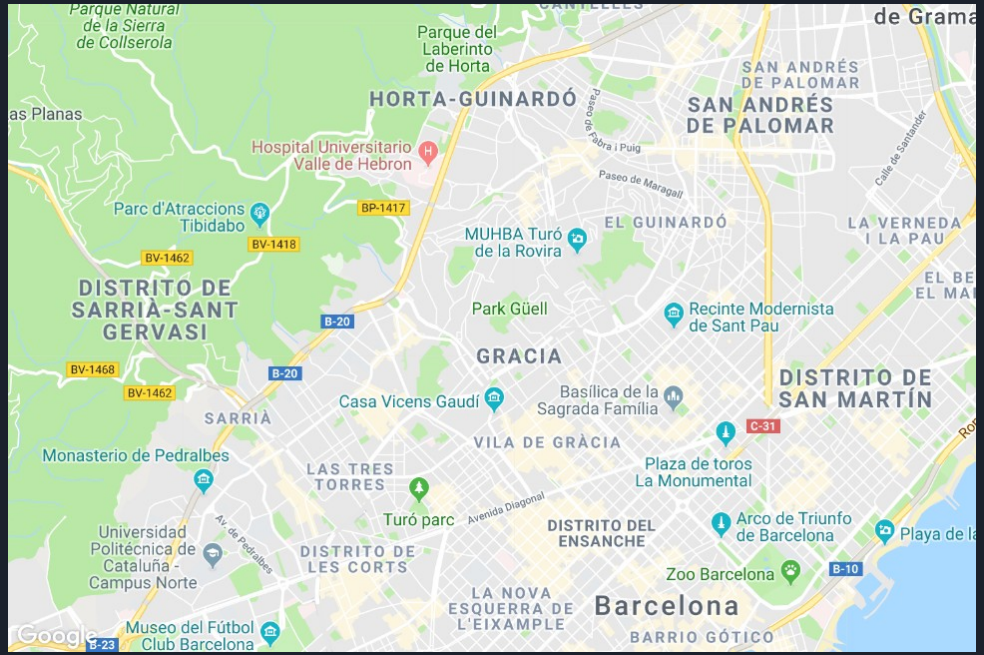
Source: https://en.wikipedia.org/wiki/Internet_protocol_suite

Networking (3): logical planes

Some additional logic definitions (from lower to upper):

- **Data plane**: carries data

Plane: carries a different type of traffic and is conceptually an overlay/independent network



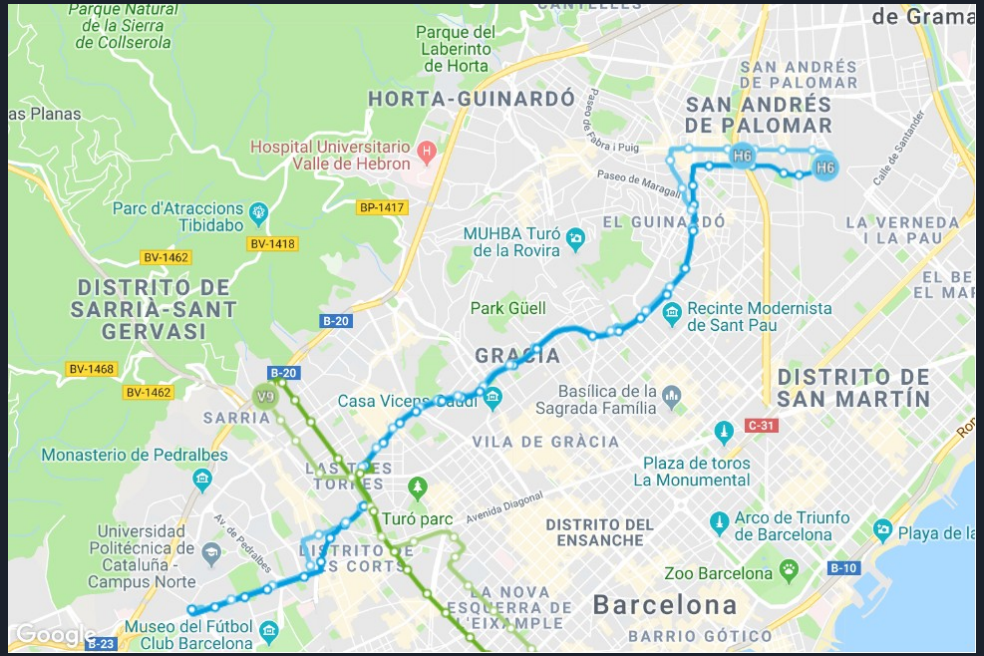
Source: https://www.tmb.cat/es/transporte-barcelona/mapa/bus, https://www.pinterest.com/pin/497366352594710913/

Networking (3): logical planes

Some additional logic definitions (from lower to upper):

- **Data plane**: carries data
- **Control plane**: instructs data plane on how to carry data

Plane: carries a different type of traffic and is conceptually an overlay/independent network



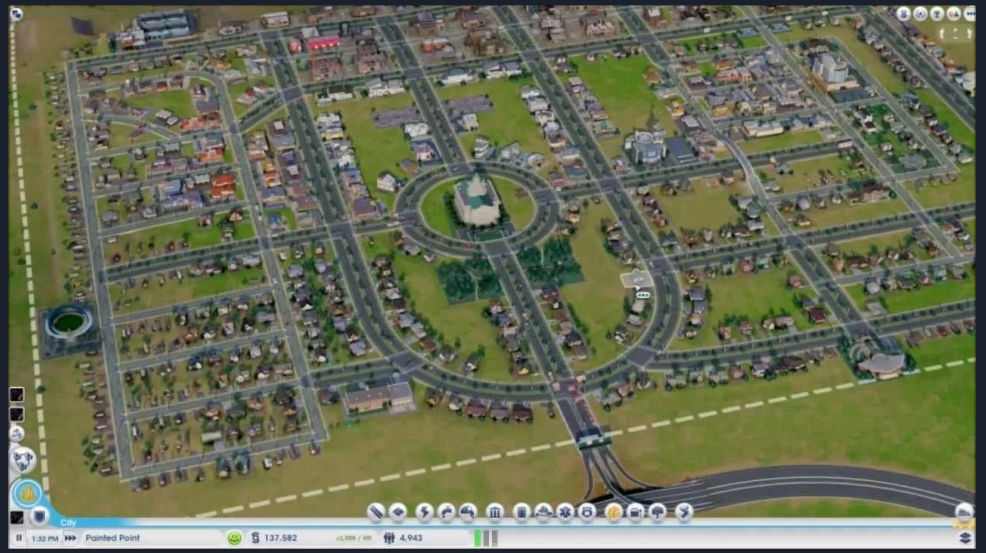
Source: https://www.tmb.cat/es/transporte-barcelona/mapa/bus, https://www.pinterest.com/pin/497366352594710913/

Networking (3): logical planes

Some additional logic definitions (from lower to upper):

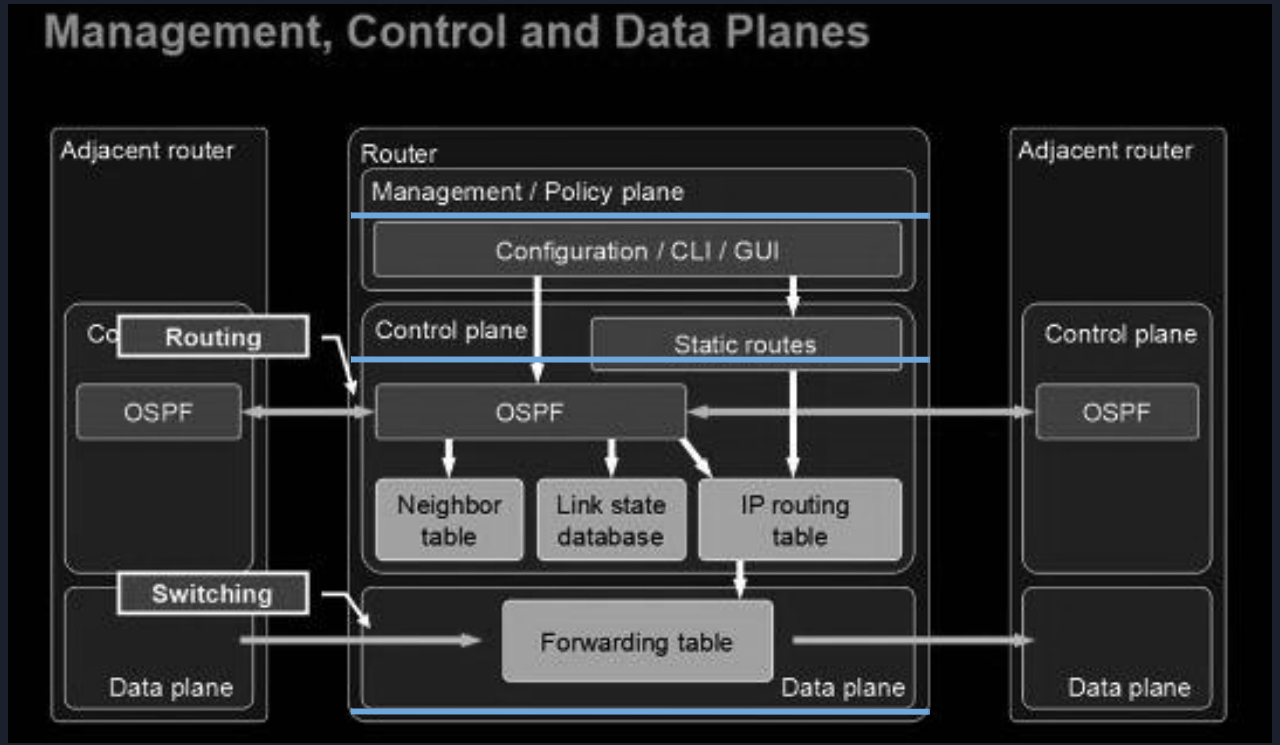
- **Data plane**: carries data
- **Control plane**: instructs data plane on how to carry data
- **Management plane**: tools to interact with control plane

Plane: carries a different type of traffic and is conceptually an overlay/independent network



Source: https://www.tmb.cat/es/transporte-barcelona/mapa/bus, https://www.pinterest.com/pin/497366352594710913/

Networking (3): logical planes



Source: https://blog.ipspace.net/2013/08/management-control-and-data-planes-in.html

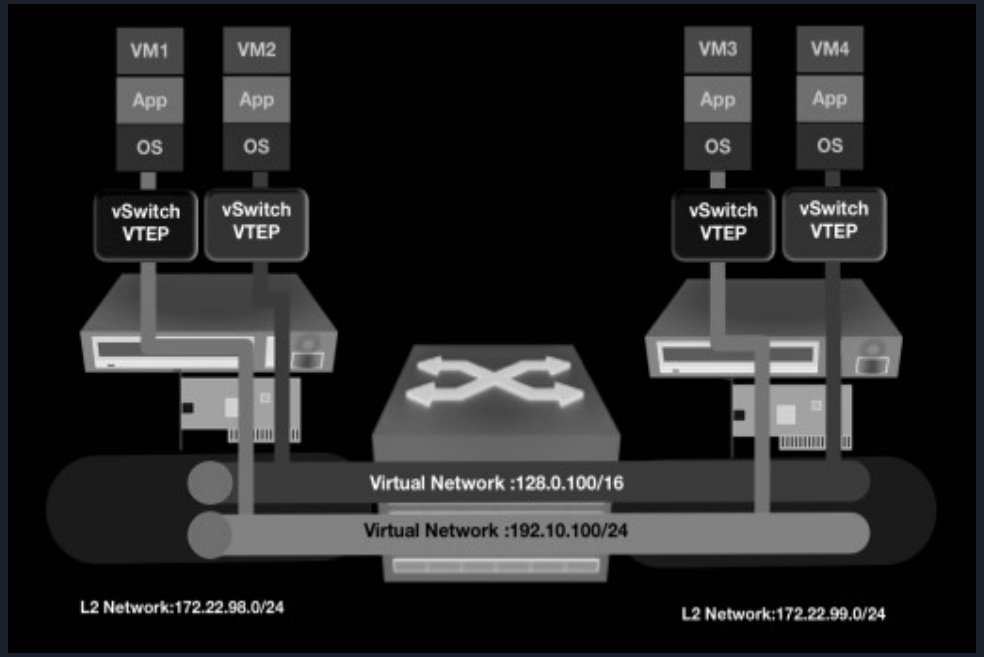
Virtual networking

Virtual networking (1): the basics

HW and SW resources combined into a single, SW-based entity (virtual network).

Virtual network is **decoupled** from the physical network.

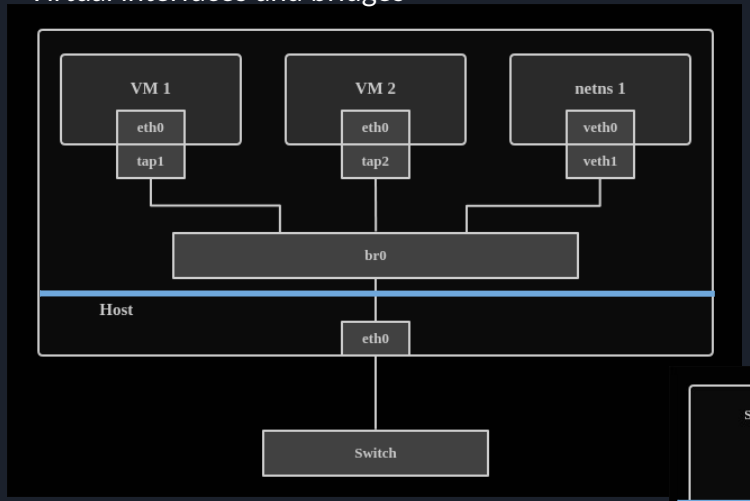
Multiple benefits: better logical separation, possible to control via SW, separate traffic, flexibility to adopt custom protocols, ...



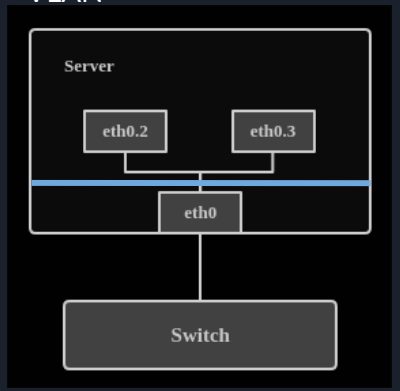
Source: https://www.arista.com/assets/data/pdf/Whitepapers/VXLAN_Scaling_Data_Center_Designs.pdf

Virtual networking (2): examples

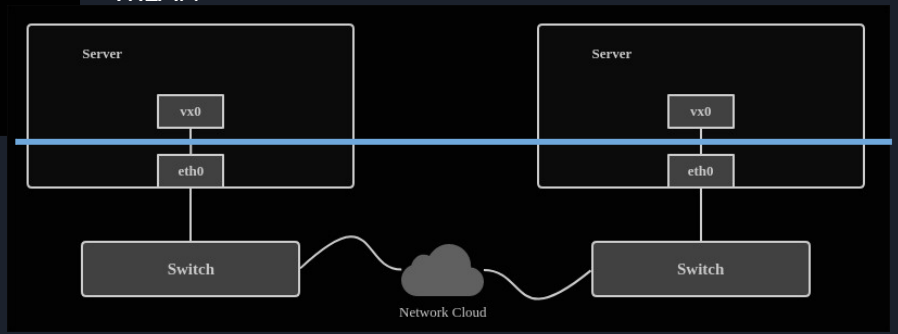
Virtual interfaces and bridges



VLAN



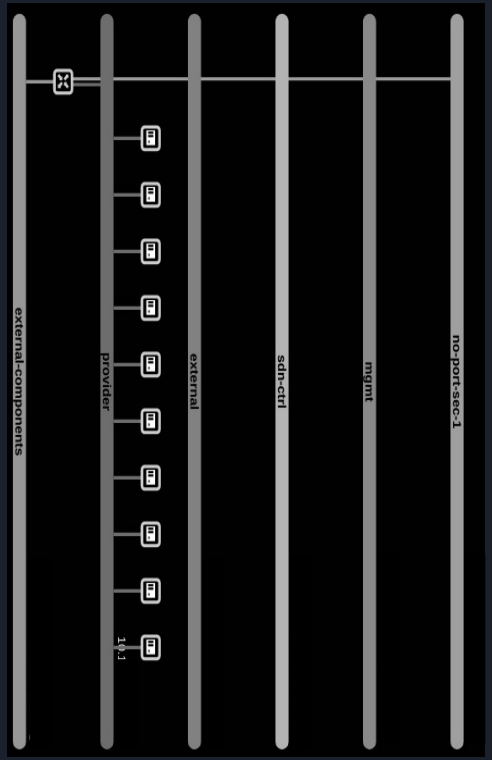
VxLAN

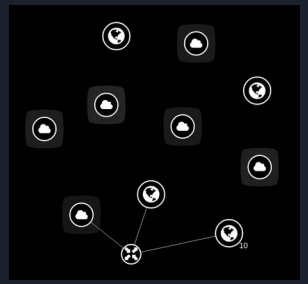


Virtual Machines make use of **virtual interfaces**, **bridges** and **tags** (VLAN, VxLAN, etc)

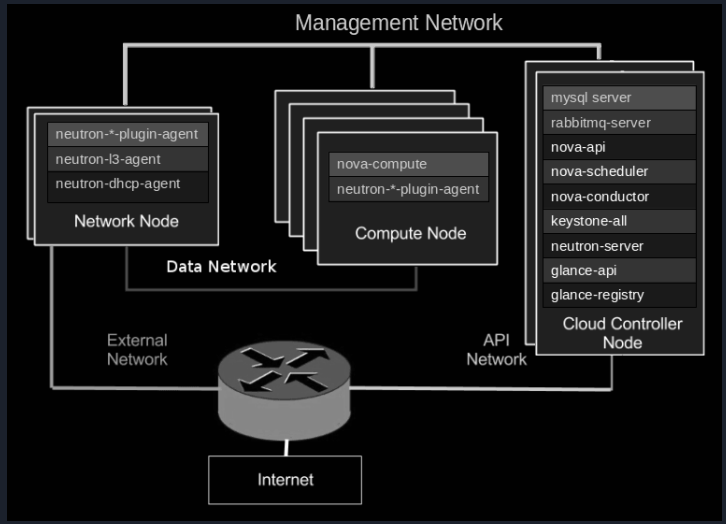
Source: https://developers.redhat.com/blog/2018/10/22/introduction-to-linux-interfaces-for-virtual-networking/

Virtual networking (2): examples





Multiple virtual networks: to control, to send data, etc



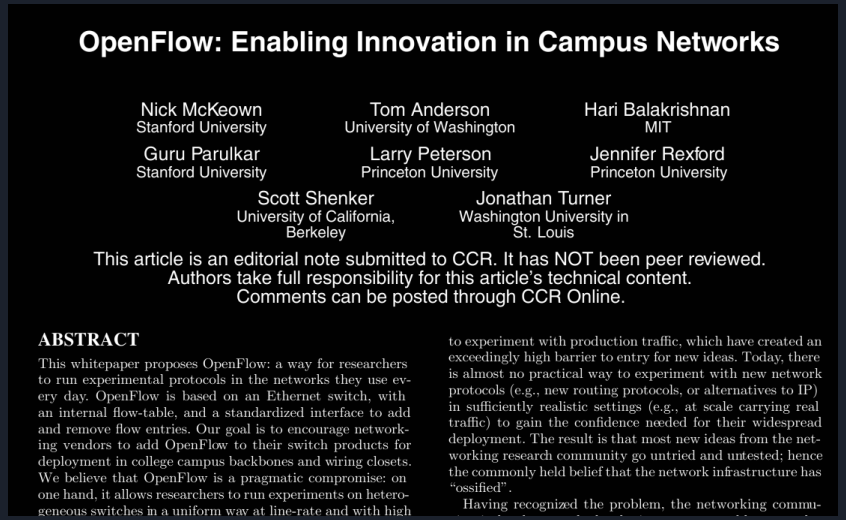
Source: internal OpenStack deployment (Ocata), https://www.dbigcloud.com/cloud-computing/176-openstack-desde-cero-neutron-parte-1.html

Software-Defined Networking

SDN: where it all started (1)

OpenFlow (2008)...

Or maybe in operator's experiments: e.g., AT&T (1989)



OpenFlow: Enabling Innovation in Campus Networks

Nick McKeown, Stanford University
Tom Anderson, University of Washington
Hari Balakrishnan, MIT
Guru Parulkar, Stanford University
Larry Peterson, Princeton University
Jennifer Rexford, Princeton University
Scott Shenker, University of California, Berkeley
Jonathan Turner, Washington University in St. Louis

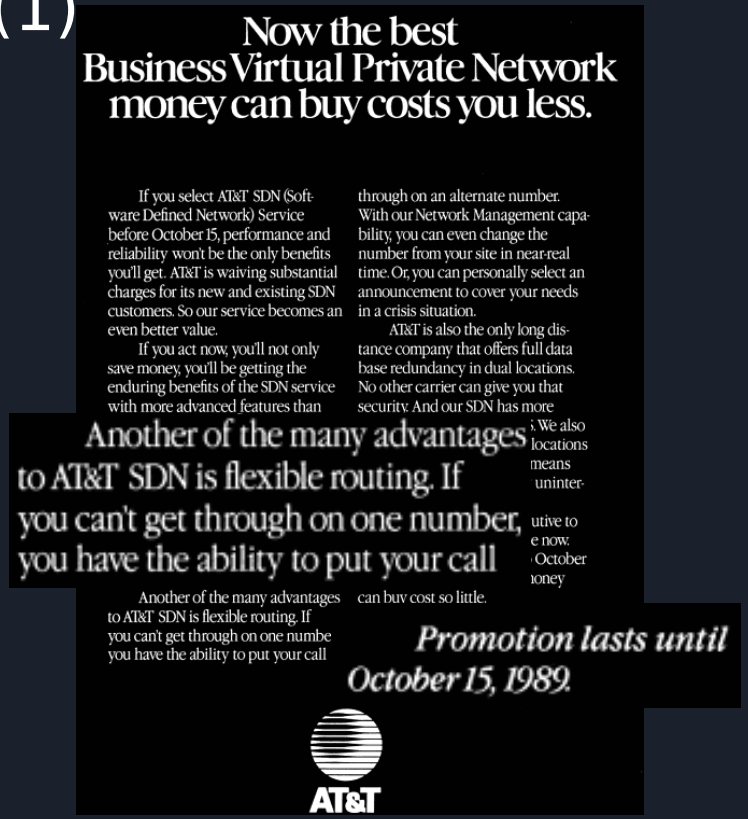
This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
Authors take full responsibility for this article's technical content.
Comments can be posted through CCR Online.

ABSTRACT

This whitepaper proposes OpenFlow: a way for researchers to run experimental protocols in the networks they use every day. OpenFlow is based on an Ethernet switch, with an internal flow-table, and a standardized interface to add and remove flow entries. Our goal is to encourage networking vendors to add OpenFlow to their switch products for deployment in college campus backbones and wiring closets. We believe that OpenFlow is a pragmatic compromise: on one hand, it allows researchers to run experiments on heterogeneous switches in a uniform way at line-rate and with high

to experiment with production traffic, which have created an exceedingly high barrier to entry for new ideas. Today, there is almost no practical way to experiment with new network protocols (e.g., new routing protocols, or alternatives to IP) in sufficiently realistic settings (e.g., at scale carrying real traffic) to gain the confidence needed for their widespread deployment. The result is that most new ideas from the networking research community go untried and untested; hence the commonly held belief that the network infrastructure has "ossified".

Having recognized the problem, the networking commu-



Now the best Business Virtual Private Network money can buy costs you less.

If you select AT&T SDN (Software Defined Network) Service before October 15, performance and reliability won't be the only benefits you'll get. AT&T is waiving substantial charges for its new and existing SDN customers. So our service becomes an even better value.

If you act now, you'll not only save money, you'll be getting the enduring benefits of the SDN service with more advanced features than

through on an alternate number. With our Network Management capability, you can even change the number from your site in near-real time. Or, you can personally select an announcement to cover your needs in a crisis situation.

AT&T is also the only long distance company that offers full data base redundancy in dual locations. No other carrier can give you that security. And our SDN has more

Another of the many advantages to AT&T SDN is flexible routing. If you can't get through on one number, you have the ability to put your call

Another of the many advantages to AT&T SDN is flexible routing. If you can't get through on one numbe you have the ability to put your call

can buy cost so little.

Promotion lasts until October 15, 1989.

AT&T

Source: https://www.researchgate.net/publication/220195143_OpenFlow_Enabling_innovation_in_campus_networks, https://blog.ipspace.net/2015/04/when-did-sdn-really-start.html

SDN: the basics (2)

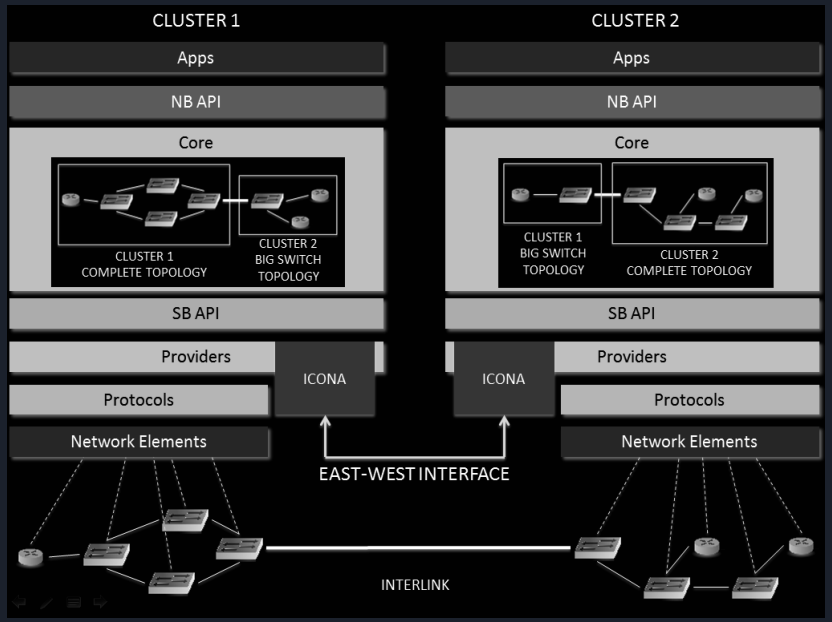
- The behaviour of the network devices is now defined by software
- Typical separation to be concerned about:
  - Data forwarding
    - Typically more strict and defined by the vendor
    - Vendor now adheres to a subset of best practices
  - Control
    - Rules for network devices are inserted from controllers
    - Each controller follows its particular syntax
- Benefits:
  - More independence from the HW, more flexibility (e.g., to support non-standardised protocols)

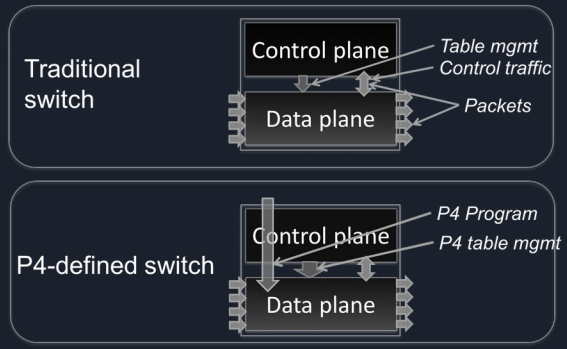
SDN: some approaches (3)

- Different ways to implement
  - Evolving from a static infrastructure
    - Usage of scripts to automate changes and management on network devices
      - Examples: Ansible/Puppet, paramiko/netmiko, ...
  - Leveraging on virtual stacks
    - On-premise DC-like services
      - Examples: OpenStack, Juniper Contrail, ...
  - OpenFlow (OF) way
    - One (or few) SDN controllers to rule them all
      - Examples: OF v1.5 via ONOS, ODL, Ryu with physical or virtual devices (OVS, ...)
  - P4 way
    - Each device programmed and compiled internally
      - Examples: P4_16 in simple-switch (virtual & ASICS), Barefoot tools (Tofino), ...

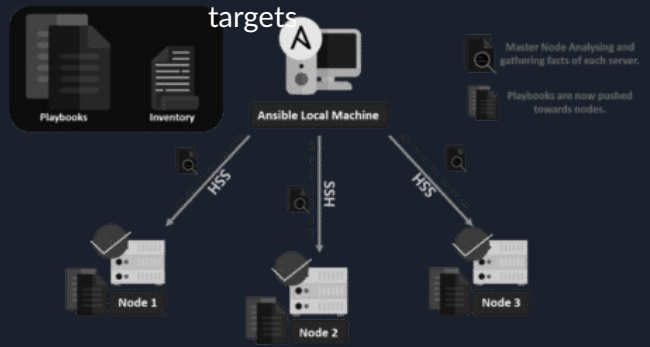
SDN: some approaches (3)

Distributed cluster of SDN controllers





Single deployment - multiple targets



Source: https://wiki.onosproject.org/display/ONOS/Design, https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html, https://intellipaat.com/blog/tutorial/devops-tutorial/ansible-tutorial/

SDN: some approaches (3)

```
curl -X POST --header 'Content-Type: application/json' --header
'Accept: application/json' --user
karaf:karaf -d
'{"priority":40006,"timeout":0,"isPermanent":true,"deviceId":"of:0010
000000000005","treatment":{"i
nstructions":[{"type":"OUTPUT","port":"2"}]},"selector":{"criteria":
[{"type":"ETH_TYPE","ethType":"0x
0800"},{"type":"IN_PORT","port":"11"}]}}'
http://x.x.x.x:8181/onos/v1/flows/of:0010000000000005?
appId=org.onosproject.openflow
```

```
   action clone_packet() {
       const bit<32> REPORT_MIRROR_SESSION_ID =
500;
       clone(CloneType.I2E,
REPORT_MIRROR_SESSION_ID);
   }
   apply {
       if (hdr.ipv4.isValid()) {
           ipv4_forward_table.apply();
           clone_packet();
       }
```

```
dpid =
self.__dpid_to_int(event.dpid)
out_port = 5 or event.port

msg = of.ofp_flow_mod()
msg.match.dl_vlan = 3021
msg.match.in_port = 2
msg.match.out_port = out_port

msg.idle_timeout = 10
msg.hard_timeout = 120
msg.priority = 200

msg.actions.append(of.ofp_action
_output(port = out_port))
event.connection.send(msg)
```

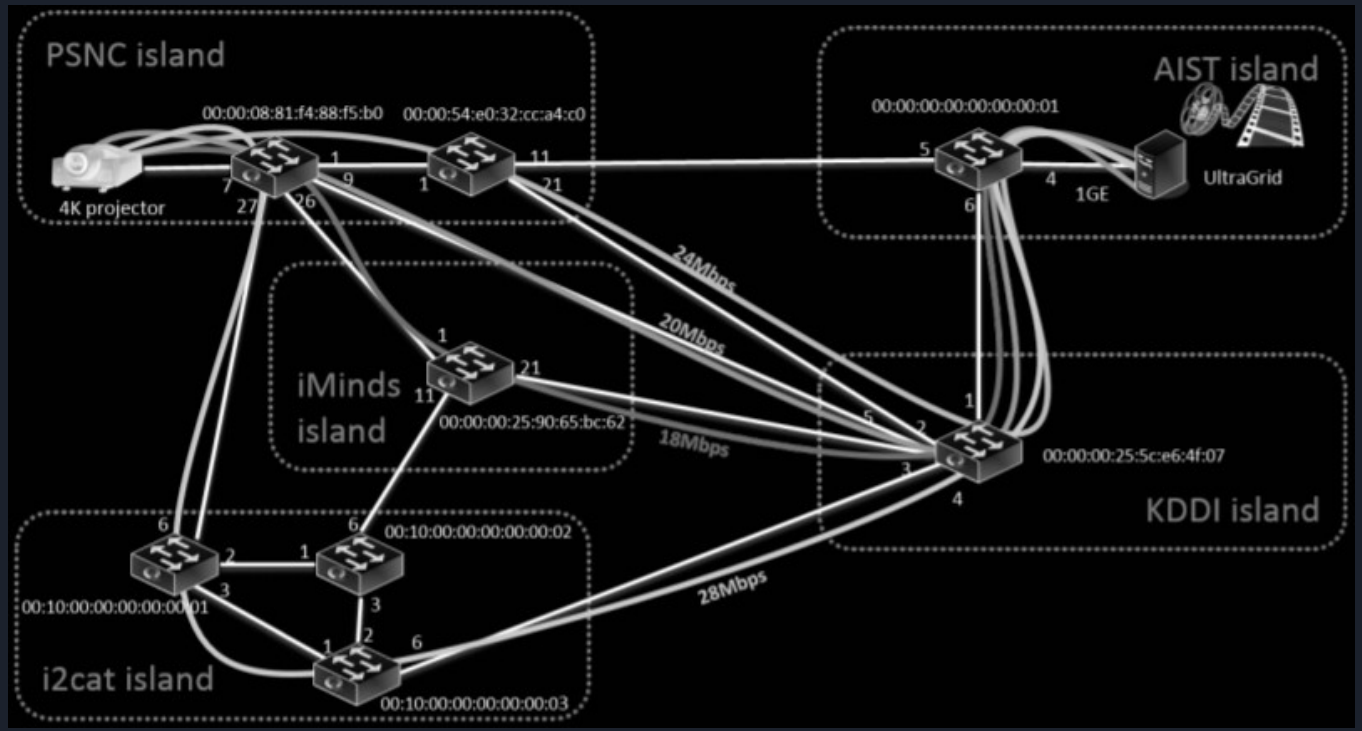
```
 shell:
   ip addr flush dev {{ iface_name }} \
   && ip addr add {{ iface_ip_net }} dev
{{ iface_name }} \
   && ip link set {{ iface_name }} up
 args:
   executable: /bin/bash
 ignore_errors: True
 when: (iface_name | search("eth")) and
(iface_in_target_node) and (iface_exists)
```

SDN: analysis (4)

- Pros
  - Visibility: providing tools to interact with the different planes and see each of them
  - All in one place: rules defined once (as part of app, sent by REST, ...) and pushed everywhere
  - Automation: behaviour of network devices can be re-configured on-the-fly
  - Extensibility: using virtual devices it is possible to define new protocols / otherwise it is still more flexible than a closed HW device
- Cons
  - Some feature Single* Point of Failure: everything connected to one or few* (cluster) controllers
    - From availability (disconnections) to security -- but there are options to mitigate risk!
  - Some still depending on vendors for implementation of basic subset of features

SDN: research examples (5)

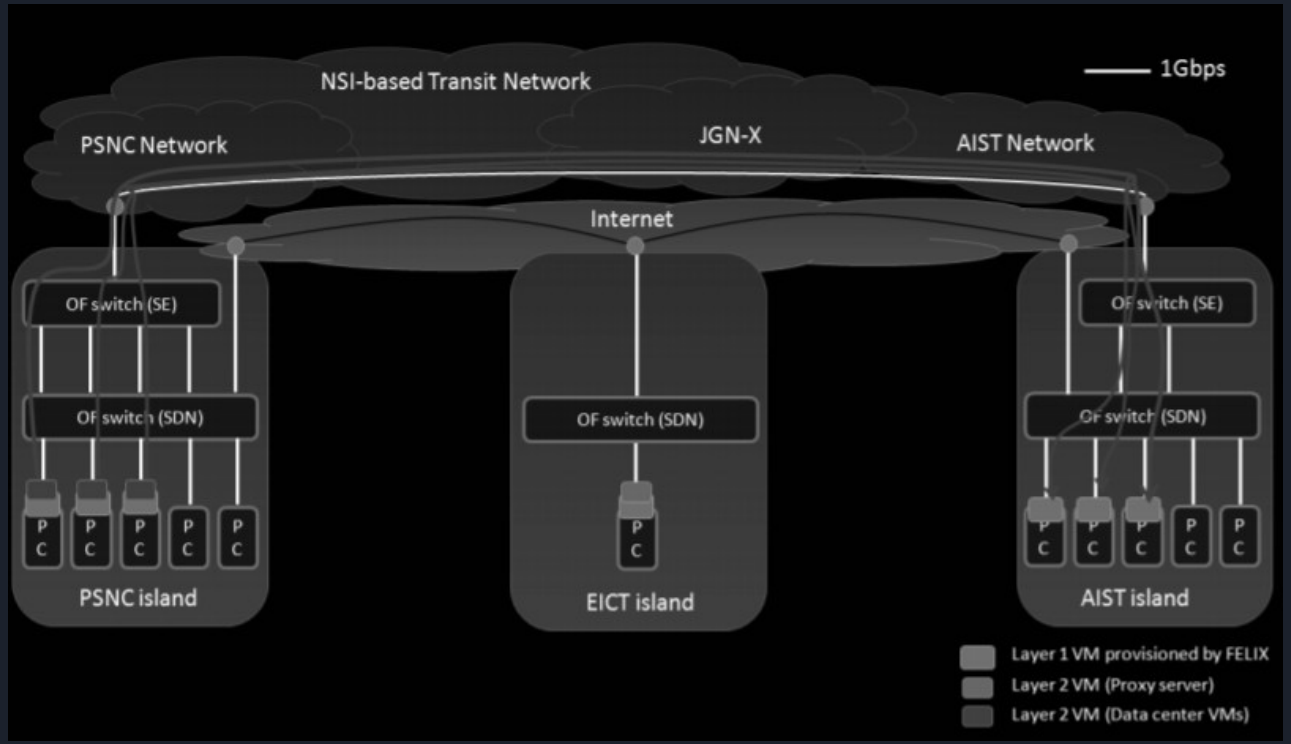
High Quality media transmission with dynamic selection of paths via SDN



Source: http://www.ict-felix.eu/wp-content/uploads/2013/06/FELIX-D4.2-FELIX_Experiments_Report.pdf

SDN: research examples (5)

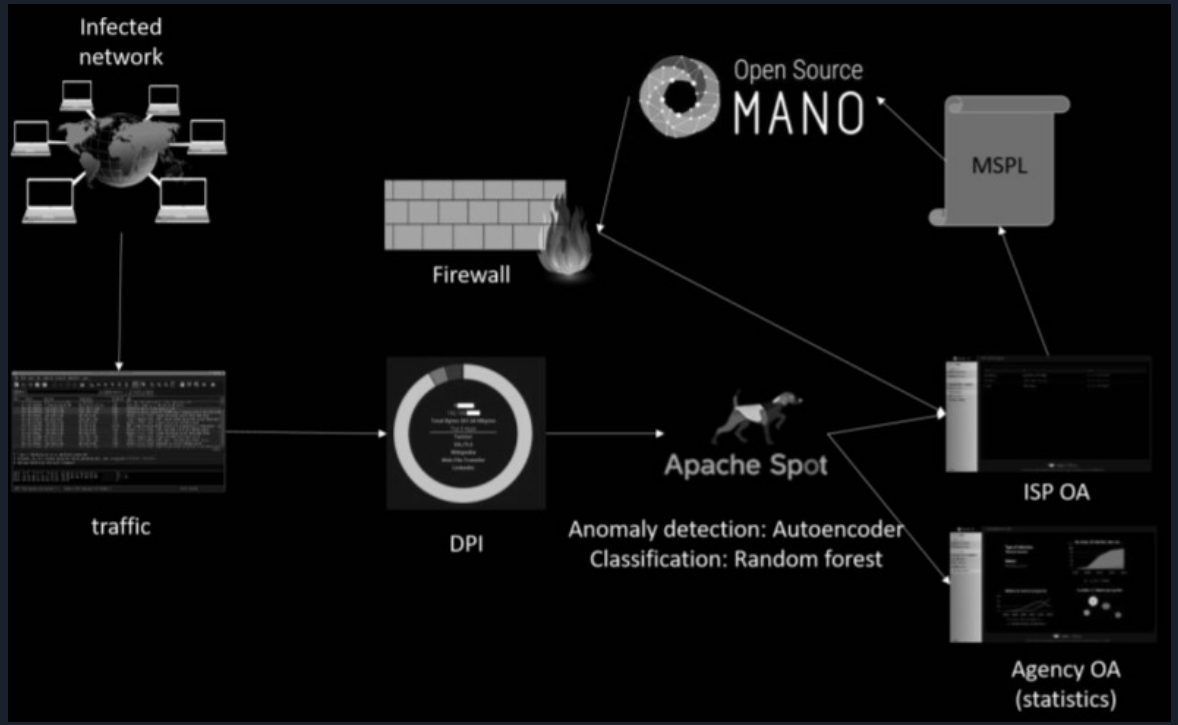
Dynamically move the processing of intensive work to one or another Data Centre



Source: http://www.ict-felix.eu/wp-content/uploads/2013/06/FELIX-D4.2-FELIX_Experiments_Report.pdf

SDN: research examples (5)

Wannacry (worm) attack detection via ML (unsupervised learning) and mitigation (via NFV/SDN)



Source: *https://www.shield-h2020.eu/documents/project-deliverables/SHIELD_D5.2_Final_demonstration,_roadmap_and_validation_results_v.1.0.pdf*

Recap: virtualisation in networking

- Process Automation, CI/CD, ...
  - Establish pipelines for processing (build, deploy, ...)
- Network Function Virtualisation
  - Virtualising logic that was once only in the HW
- 5G operation
  - Defining who uses the network and how (QoS, etc)
- Media streaming, data recovery plans, ...
  - Dynamic reaction & definition of network paths
- Cloud computing
  - Remote processing of data, dynamic configuration of resources (computing, networking)
- IoT nodes, edge computing
  - Aggregate data and process locally before going to the cloud
- Edge computing, Federated Learning, ...
  - Process data locally/interact before going to the cloud

Source: https://twitter.com/5gppp, https://federated.withgoogle.com/

Recap: where is virtualisation* used

**and potentially SDN*

- NFV MANO (Management and Orchestration) stacks
  - OSM, ONAP, OPNFV, CORD, OPEN-O, ...
- Virtual processing of traffic and configuration
  - OVS, ...
- Accelerating virtual packet processing
  - DPKD, FD.io, ...
- On-site cloud operating systems and stacks
  - OpenStack, ...
- Cloud services for remote processing
  - AWS, Google Cloud, ...
- Continuous Integration / Continuous Deployment
  - Jenkins, Bamboo, Travis, ...

Recap: where is virtualisation* used

**and potentially SDN*

- NFV MANO (Management and Orchestration) stacks
  - OSM, ONAP, OPNFV, CORD, OPEN-O, ...
- Virtual processing of traffic and configuration
  - OVS, ...
- Accelerating virtual packet processing
  - DPKD, FD.io, ...
- On-site cloud operating systems and stacks
  - OpenStack, ...
- Cloud services for remote processing
  - AWS, Google Cloud, ...
- Continuous Integration / Continuous Deployment
  - Jenkins, Bamboo, Travis, ...
- Containers, orchestration, automation
  - LXC, Docker; Kubernetes, Mesos; Ansible, Terraform, ...

Materials

Materials: docs, projects and courses

Documentation

- An introduction to SDN: https://qmonnet.github.io/whirl-offload/2016/07/08/introduction-to-sdn/
- What is OpenFlow: https://www.sdxcentral.com/networking/sdn/definitions/what-is-openflow/
- What is P4: https://rickmur.com/what-is-p4/

Projects

- ONOS — an open-source SDN controller: https://onosproject.org/
- STRATUM project (switch OS for SDN): https://stratumproject.org
- OSM — an NFV orchestrator: https://osm.etsi.org/
- OpenStack — the cloud computing OS: https://www.openstack.org/

Courses

- Software-Defined Networking: https://www.coursera.org/learn/sdn
- Cloud computing: https://www.coursera.org/learn/cloud-networking
- Network transformation 101 (*also check 102*): https://www.coursera.org/learn/network-transformation-101