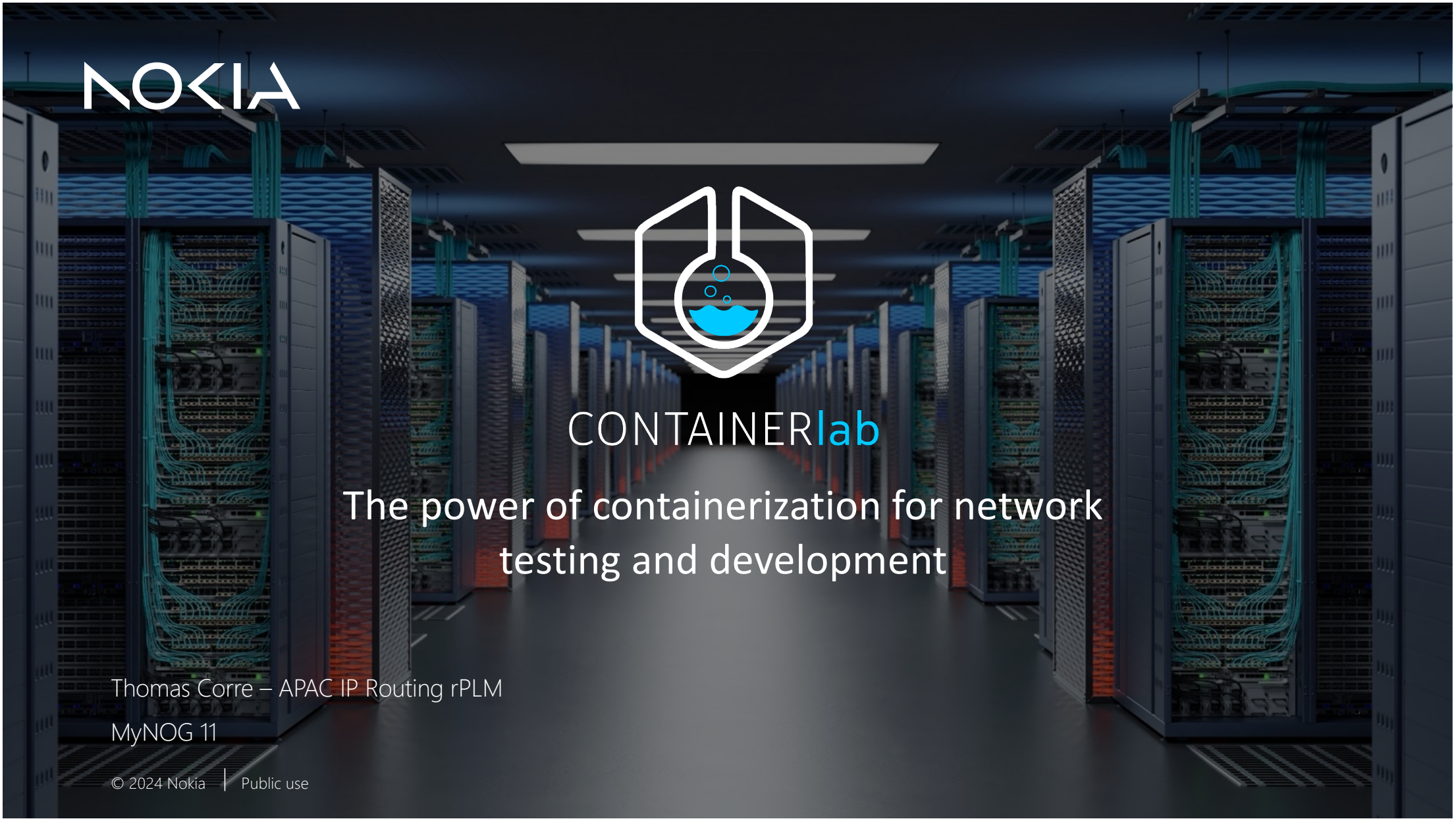
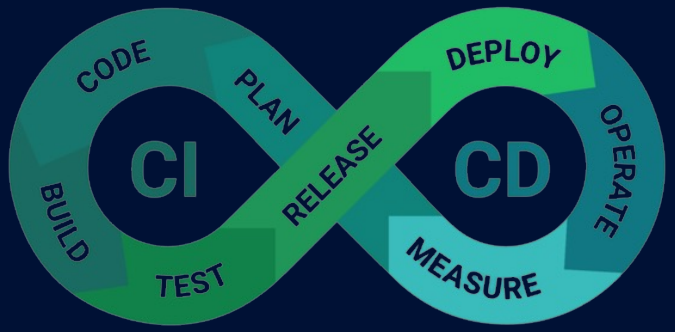

NOKIA
CONTAINERlab
The power of containerization for network testing and development
Thomas Corre – APAC IP Routing rPLM
MyNOG 11
© 2024 Nokia | Public use

Network labs

A right, not a privilege

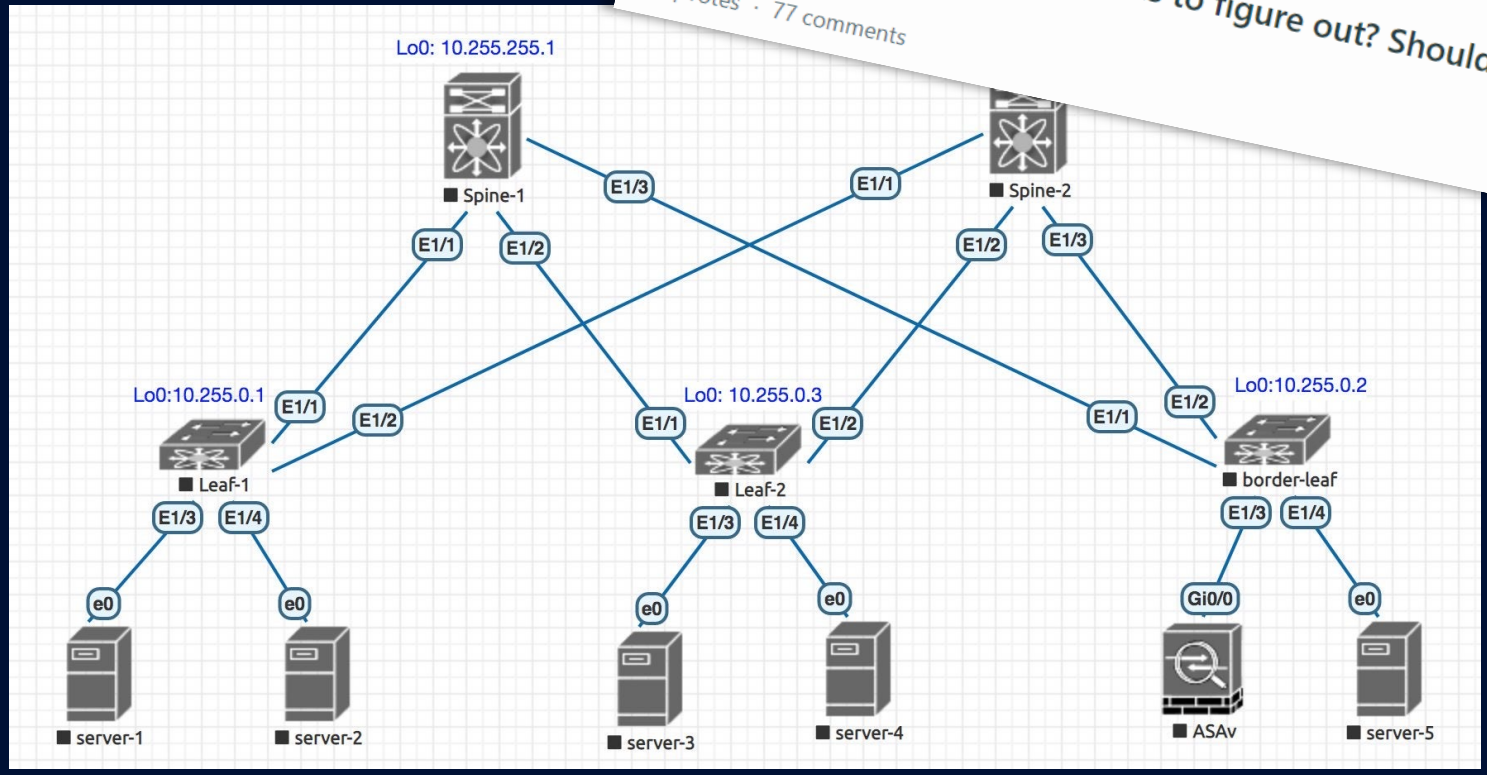


Education and Learning

Change management and validation

Network labs

How do we typically run labs today?

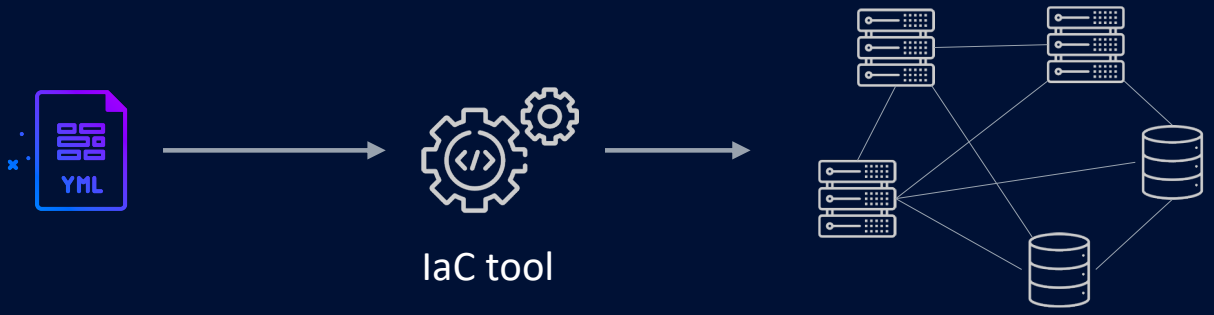


Pic from https://www.reddit.com/r/networking/comments/g5fb23/eveng_lab_strage_packet_loss/

Containerlab

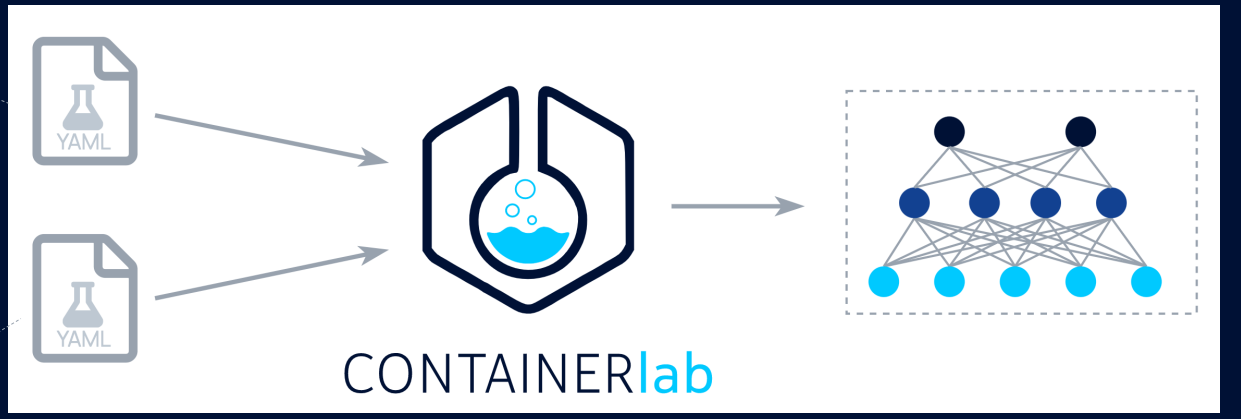
Bringing declarativeness to network labs

IT



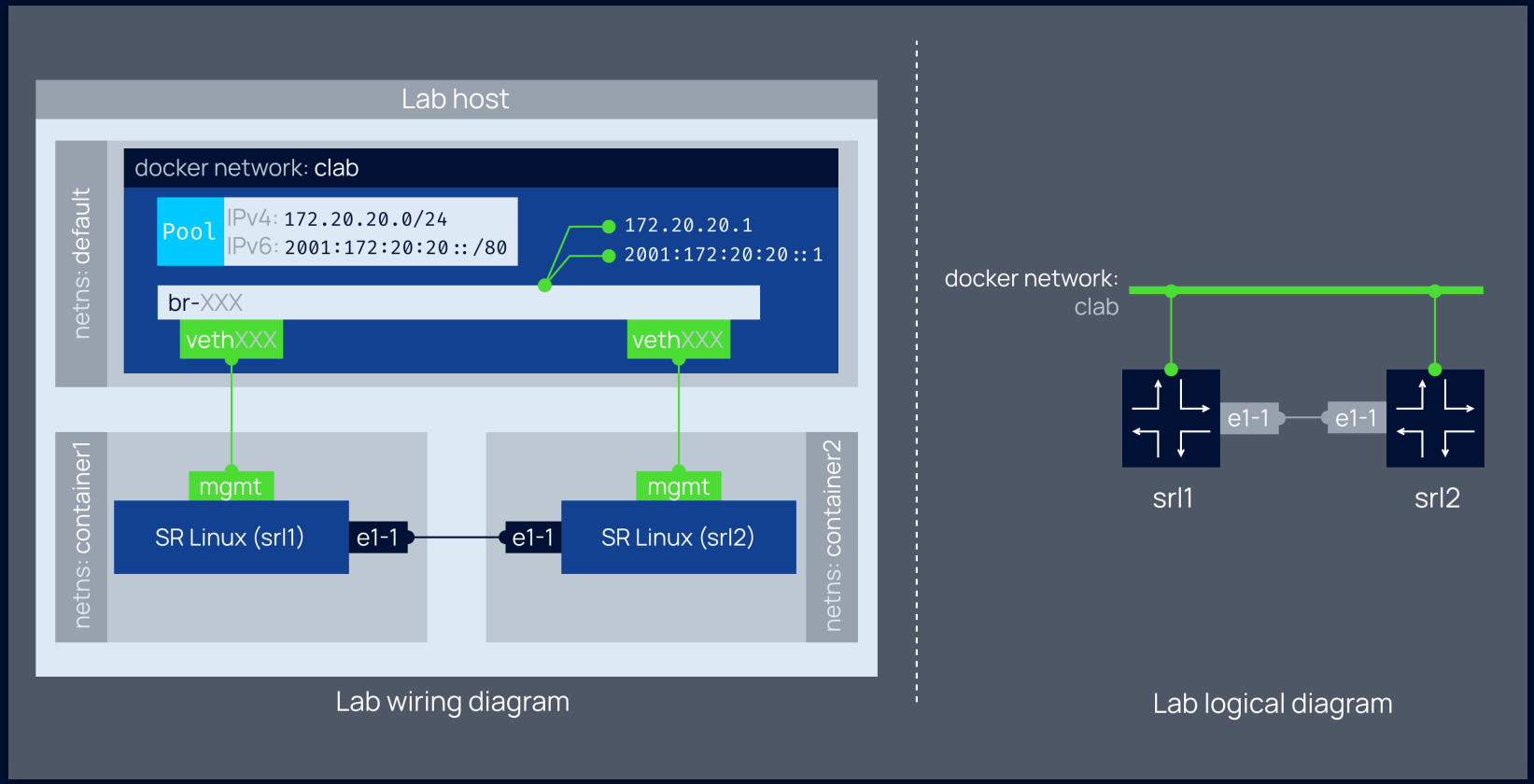
Network Labs

```
name: mylab

topology:
  nodes:
    [ ] : [ ]
    [ ] : [ ]

  links:
    - [ ] : [ ]
```



Containerlab networking

...is based on container networking



Why Containerlab

If we have lab emulation tools already?

Network emulation SW

- ⊕ Purpose built & proven
- ⊕ Free versions available
- ⊕ U
  I
- ⊖ VM-centric
  weak containers support
- ⊖ Heavy and semi-open
- ⊖ U
  I

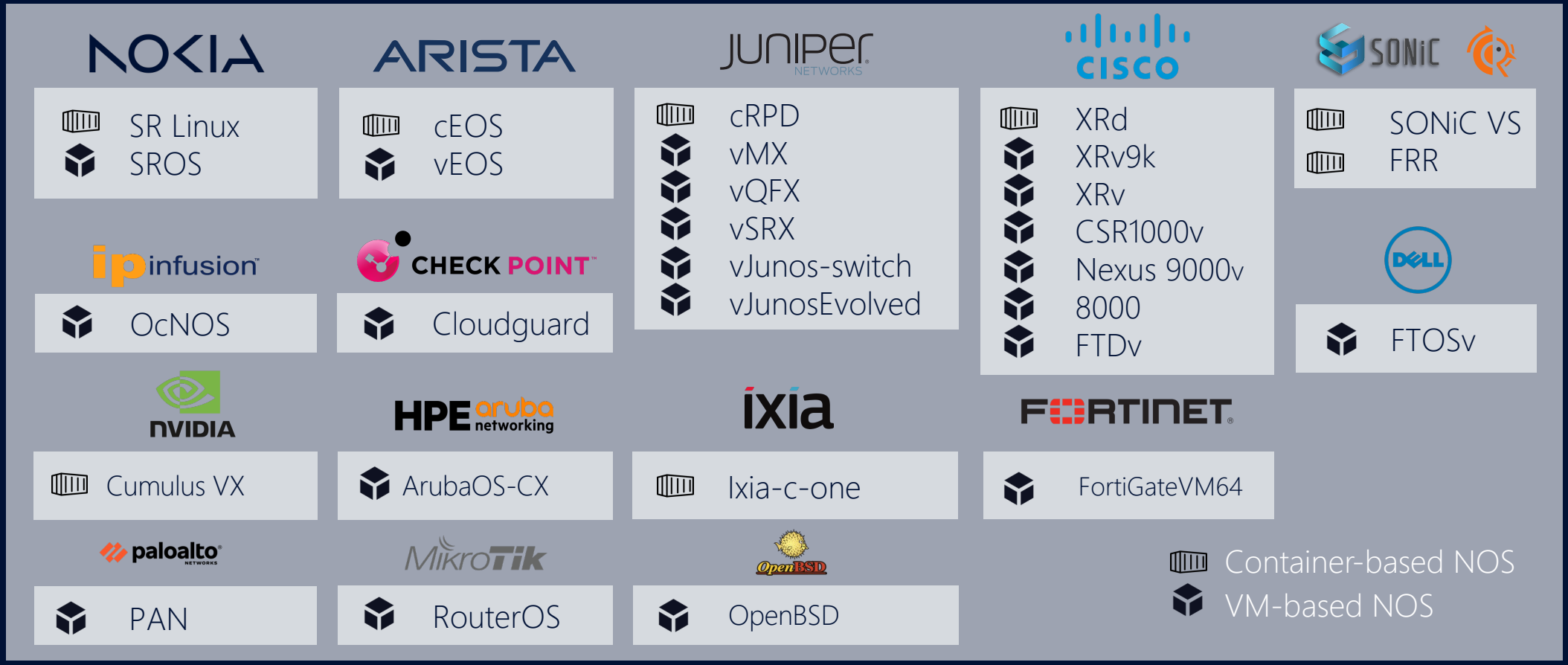
Containerlab

CONTAINERlab

- ⊕ First class support for containerized NOSes
- ⊕ Transparent datapath
- ⊕ Git friendly & better image sharing and handling
- ⊕ Repeatable lab builds and CI friendly
- ⊕ Small footprint, open, free and fast
- ⊖ No UI
- ⊖ Fewer Network OSes supported

NOKIA

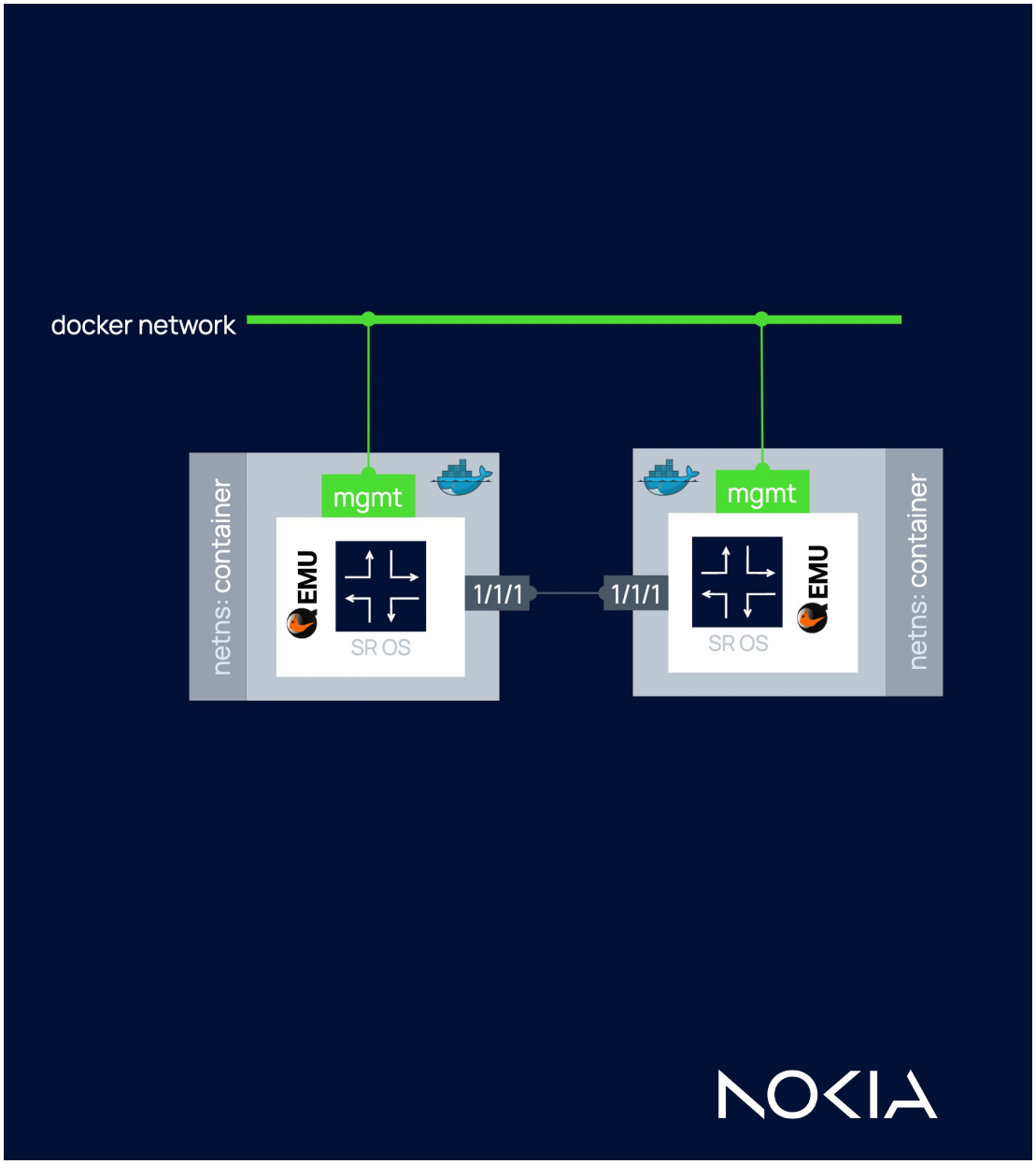
Supported NOSes



Containerlab node types

Virtual machines in a container package

- Traditional Network OS packaged as a VM
- Integrated with containerlab through vrnetlab open-source project
- Onboard existing VM-based NOSes



Container images

That should be easy to get … right ?

- Containerlab doesn't provide images
- Reach out to your favorite NOS vendor
- This *can* be the longest step in the process of deploying a lab …
- … but it doesn't have to be !
- Some vendors provide public and license-free images – e.g. SR Linux from Nokia 😉

github.com/nokia/srlinux-container-image/pkgs/container/srlinux

```
$ docker pull ghcr.io/nokia/srlinux:24.3.1
```

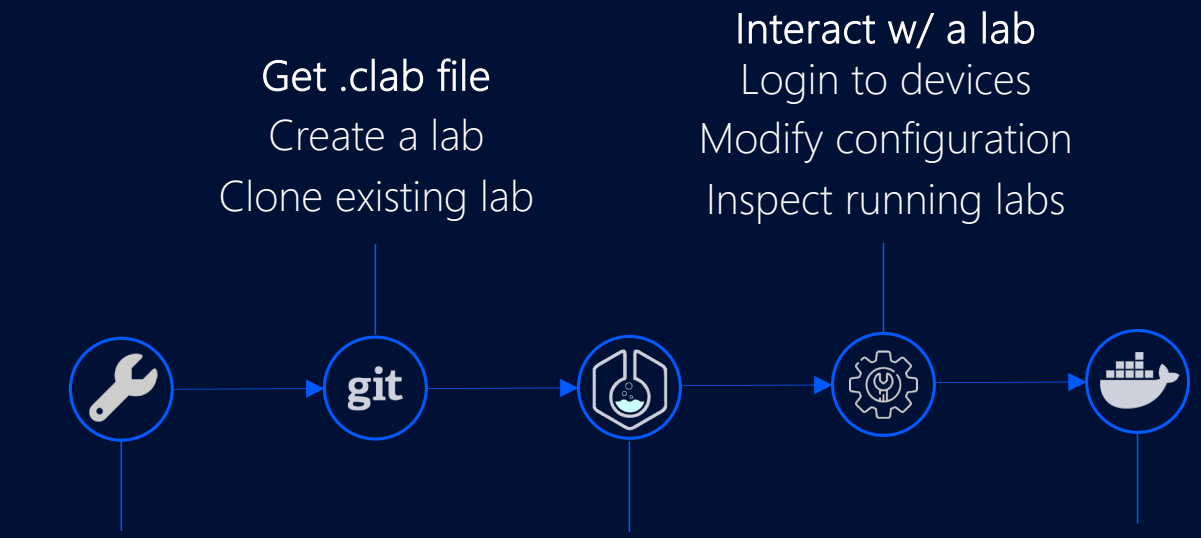
Containerlab node types

Regular container images

- All available container images
- Emulating clients
- Hundreds of network-focused software
  - Telemetry, logging stacks
  - Peering software
  - Flow collectors
  - etc

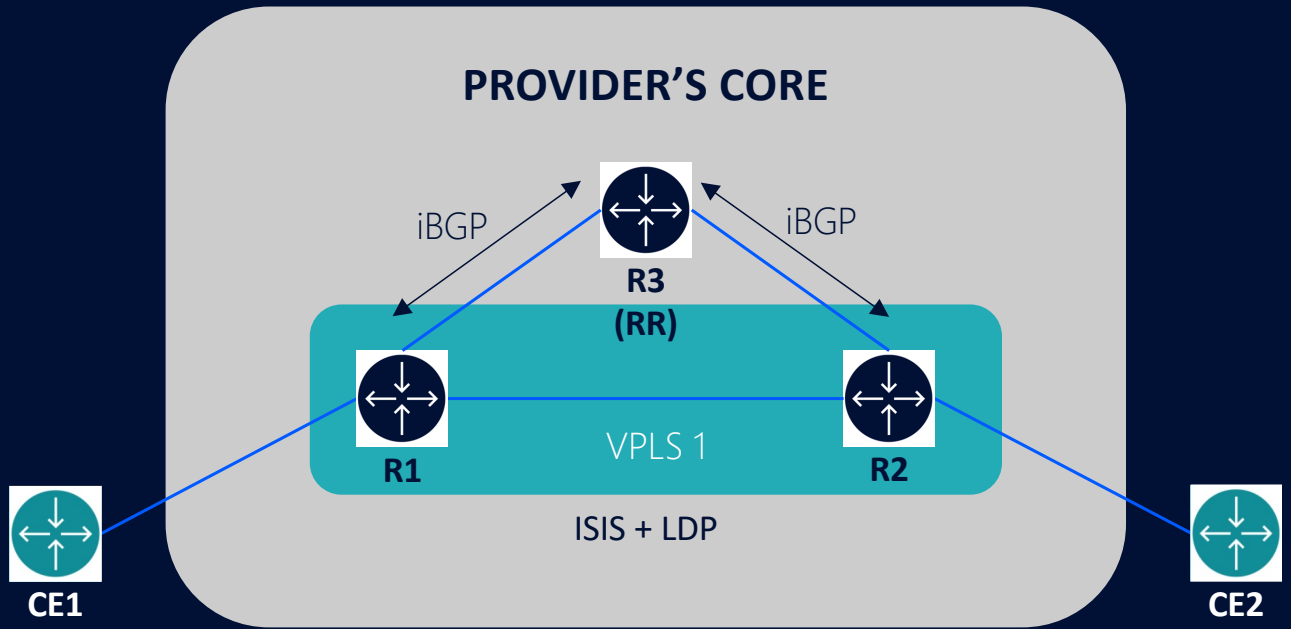
Up for an adventure ?

Basic workflow on Containerlab



Learn by doing

Basic service provider topology



Installation

Just a single command

```
~ bash -c "$(curl -sL https://get.containerlab.dev)"
Downloading https://github.com/srl-labs/containerlab/releases/dow
nload/v0.44.3/containerlab_0.44.3_linux_amd64.rpm
Preparing to install containerlab 0.44.3 from package


                     _                         _       _
                    (_)                       | |     | |
 ____ ___  ____ ____ _ ____   ____  ____ ____ | | ____| | _
/ ___) _ \|  _ \|  _)/ _  | |  _ \ / _  )/ ___) |/ _  | || \
( (__| |_|| | | | |_( ( | | | | | ( (/ /| |   | ( ( | | |_) )
\____)___/|_| |_|\___)_||_|_|_| |_|\____)_|   |_|\_||_|____/

    version: 0.44.3
     commit: cbfa6cbc
       date: 2023-08-22T12:42:06Z
     source: https://github.com/srl-labs/containerlab
 rel. notes: https://containerlab.dev/rn/0.44/#0443
```

Refer to docs for other installation options:
https://containerlab.dev/install/

Building a service provider lab

Adding a Nokia SR OS node

topology definition

vpls.clab.yml

```
name: vpls

topology:
  nodes:
    r1:
      kind: vr-nokia_sros
      image: vrnetlab/vr-sros:23.3.R2
      license: license.key
      startup-config: r1.base.cfg
```

logical view

Building a service provider lab

Adding a Juniper VMX node

topology definition

vpls.clab.yml

```
name: vpls

topology:
  nodes:
    r1: {…}

    r2:
      kind: vr-juniper_vmx
      image: vrnetlab/vr-vmx:20.4R1.12
```

logical view

r1

(Nokia SR OS)

r2

(Juniper vMX)

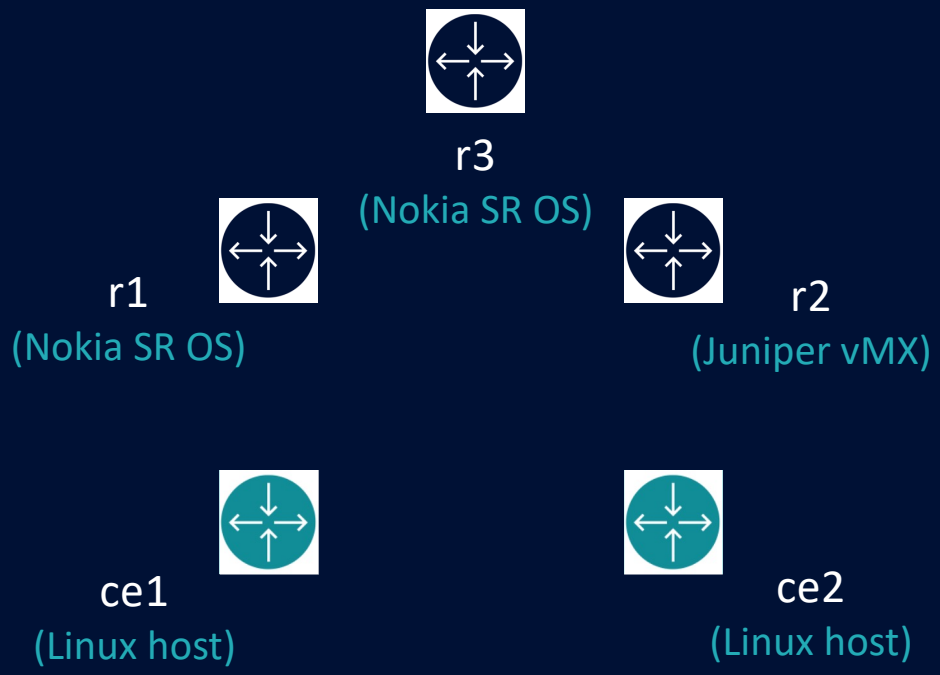
Building a service provider lab
Adding CE nodes

topology definition

vpls.clab.yml

```
name: vpls

topology:
  nodes:
    r1: {…}
    r2: {…}
    r3: {…}
    ce1: {…}

    ce2:
      kind: linux
      image: ghcr.io/hellt/network-multitool
```

logical view



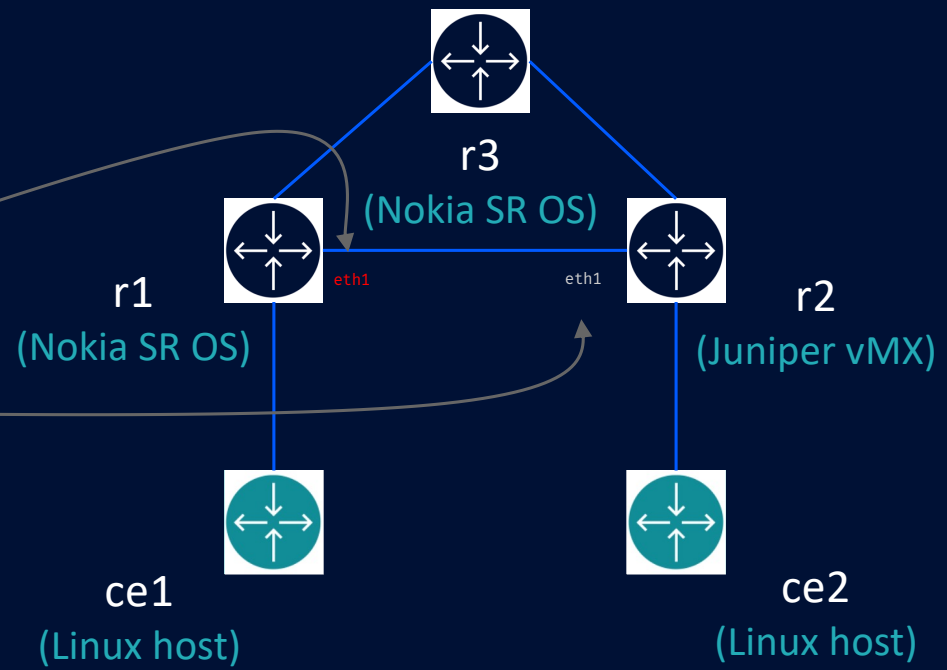
Building a service provider lab
Adding links

topology definition

vpls.clab.yml

```
name: vpls

topology:
  nodes:
    r1: {…}
    r2: {…}
    r3: {…}
    ce1: {…}
    ce2: {…}

links:
    - endpoints: ["r1:eth1", "r2:eth1"]
    - endpoints: ["r1:eth2", "r3:eth1"]
    - endpoints: ["r2:eth2", "r3:eth2"]
    - endpoints: ["ce1:eth1","r1:eth3"]
    - endpoints: ["ce2:eth1","r2:eth3"]
```

logical view



* Management links are not shown

Building a service provider lab

Deployment

```
..:~/hellt/sros-frr-ixp-lab

root@devbox:~/hellt/sros-frr-ixp-lab git:(main) (2.525s)
containerlab deploy -t ixp.clab.yml
INFO[0000] Containerlab v0.39.0 started
INFO[0000] Parsing & checking topology file: ixp.clab.yml
INFO[0000] Creating lab directory: /root/hellt/sros-frr-ixp-lab/clab-ixp
INFO[0000] Creating docker network: Name="clab", IPv4Subnet="172.20.20.0/24", IPv6Subnet="2001:172:20:20::/64", MTU="1450"
INFO[0000] Creating container: "rs2"
INFO[0000] Creating container: "rs1"
INFO[0000] Creating container: "peer2"
INFO[0000] Creating container: "peer1"
INFO[0001] Creating virtual wire: peer2:eth1 <--> ixp-net:port2
INFO[0001] Creating virtual wire: peer1:eth1 <--> ixp-net:port1
INFO[0001] Creating virtual wire: rs2:eth1 <--> ixp-net:port4
INFO[0001] Creating virtual wire: rs1:eth1 <--> ixp-net:port3
INFO[0002] Adding containerlab host entries to /etc/hosts file
+---+----------------+--------------+-------------------------------+--------------+---------+----------------+----------------------+
| # |      Name      | Container ID |             Image             |     Kind     |  State  |  IPv4 Address  |     IPv6 Address     |
+---+----------------+--------------+-------------------------------+--------------+---------+----------------+----------------------+
| 1 | clab-ixp-peer1 | 94f22546922e | sros:23.3.R1                  | vr-nokia_sros | running | 172.20.20.3/24 | 2001:172:20:20::3/64 |
| 2 | clab-ixp-peer2 | 8ba9c9bdbfce | quay.io/frrouting/frr:8.4.1   | linux         | running | 172.20.20.2/24 | 2001:172:20:20::2/64 |
| 3 | clab-ixp-rs1   | 0ac2e6518043 | quay.io/openbgpd/openbgpd:7.9 | linux         | running | 172.20.20.5/24 | 2001:172:20:20::5/64 |
| 4 | clab-ixp-rs2   | 37d7f3507b8b | ghcr.io/srl-labs/bird:2.0.11  | linux         | running | 172.20.20.4/24 | 2001:172:20:20::4/64 |
+---+----------------+--------------+-------------------------------+--------------+---------+----------------+----------------------+

root@devbox:~/hellt/sros-frr-ixp-lab git:(main)±1
```

Building a service provider lab
Connecting to the nodes

SSH

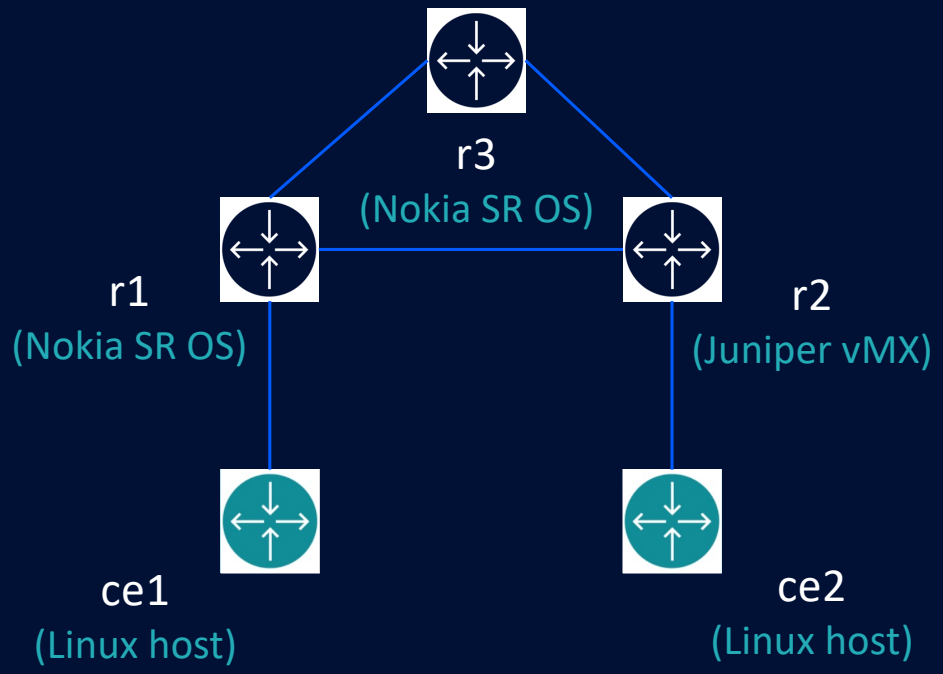
```
ssh admin@clab-vpls-r1

admin@clab-vpls-r1's password:

[/]
A:admin@r1#
```

Docker exec

```
docker exec -it clab-vpls-ce1 bash

bash-5.0#>
```

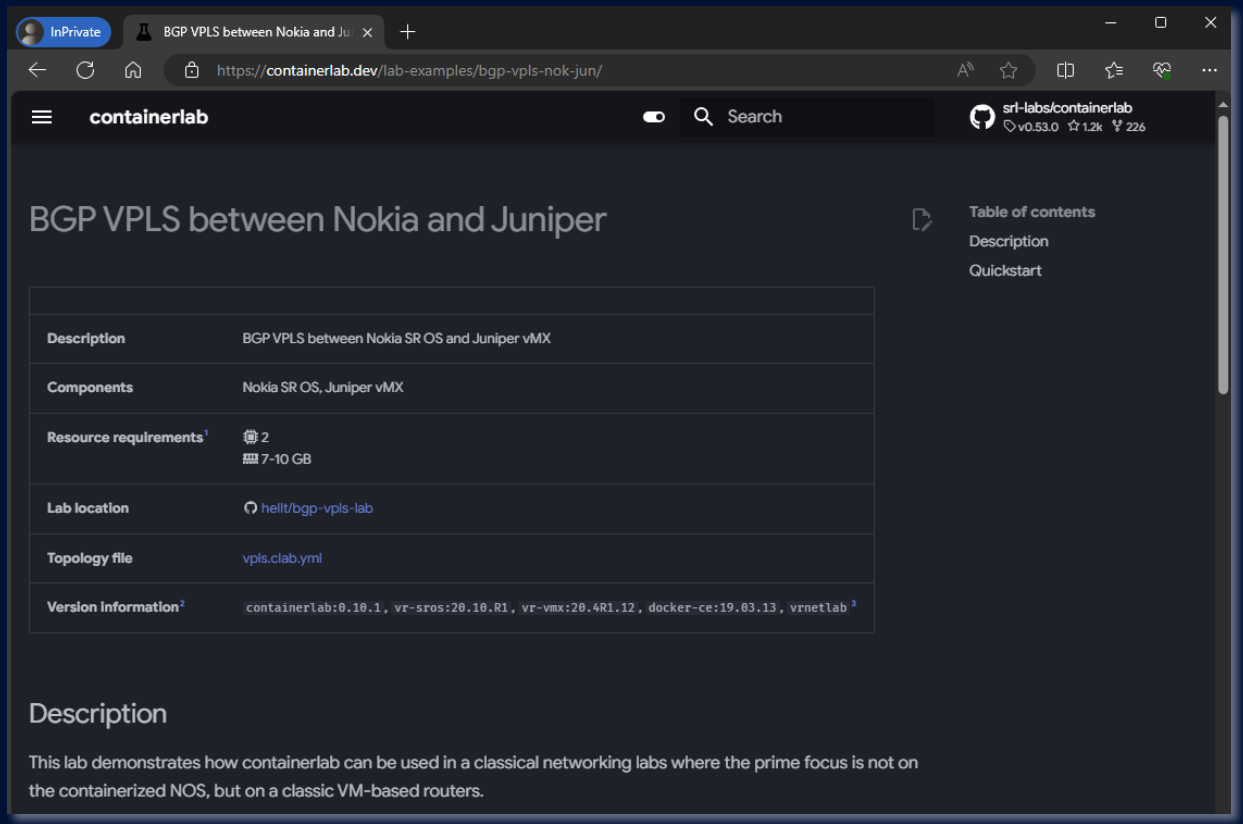
What’s next ?

- Basic VPLS configuration
  - Interface
  - BGP
  - VPLS
  - Control & data plane walkthrough



Lab

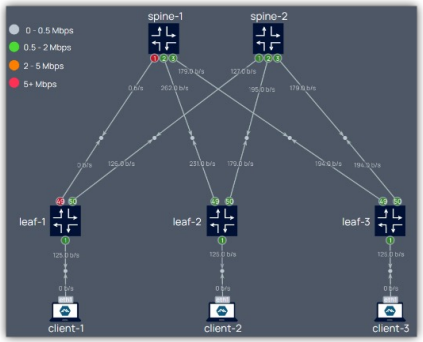
A to Z explanation



VPLS Lab

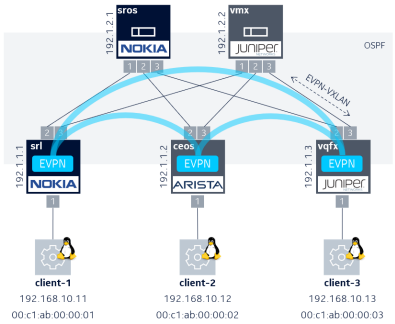
Other labs for the community

SR Linux Telemetry



srl-labs/srl-telemetry-lab

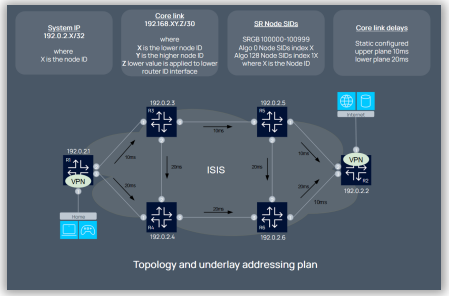
Multivendor EVPN



srl-labs/multivendor-evpn-lab

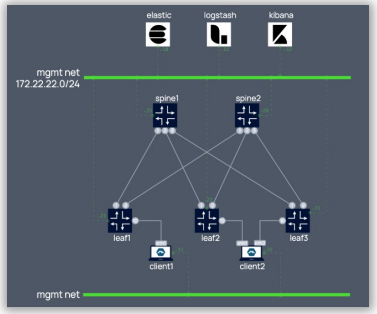
CONTAINERlab

Segment routing



srl-labs/nokia-segment-routing-lab

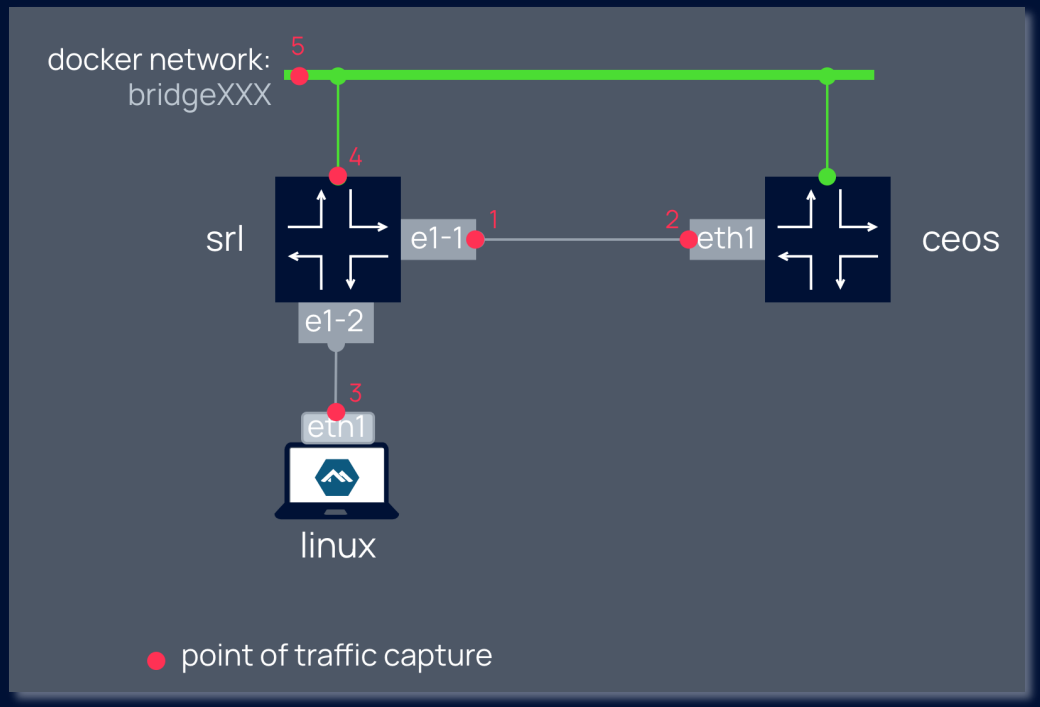
ELK Logging



srl-labs/srl-elk-lab

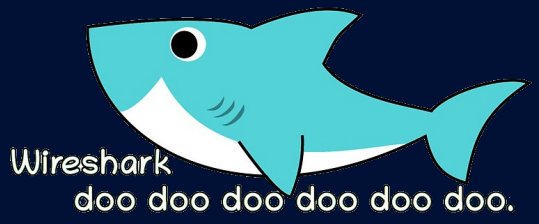
Traffic capture

Pcapng or it didn’t happen



Command to capture at point #1

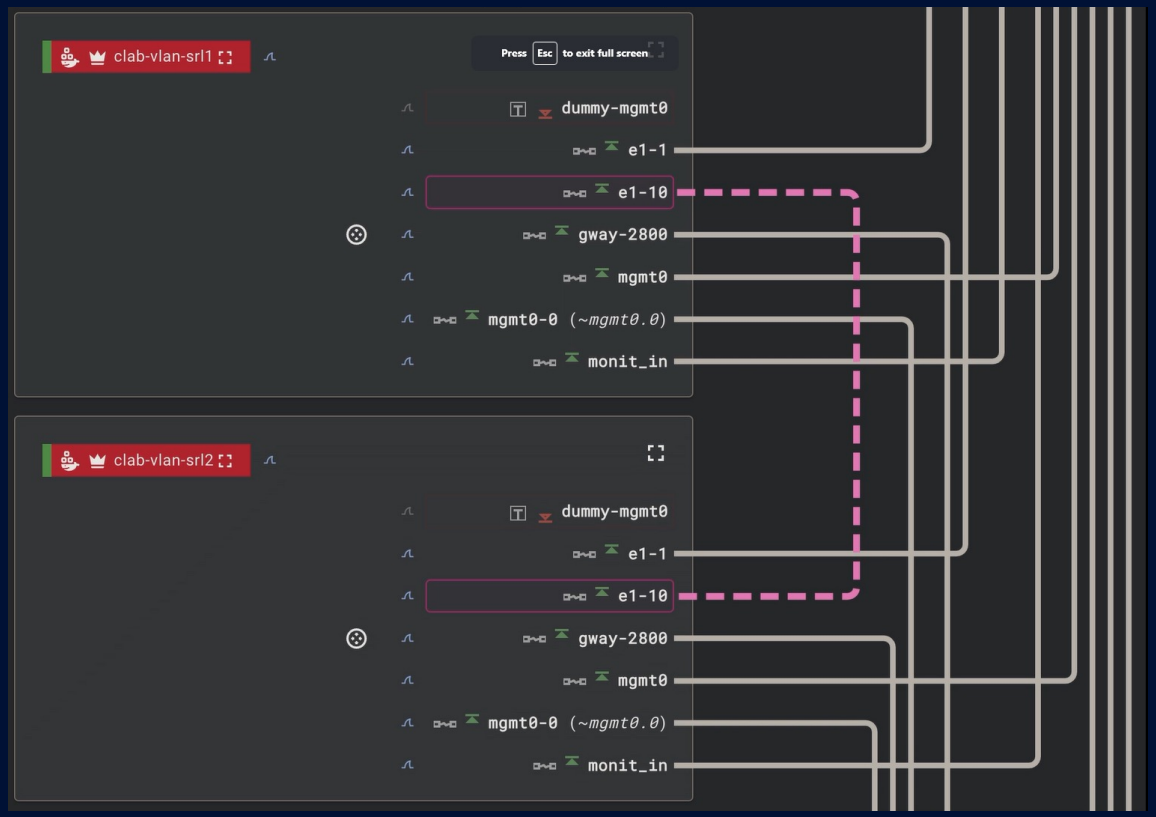
```
ssh $clab_host "ip netns exec $container tcpdump -
U -nni e1-1 -w -" | wireshark -k -i -
```



Traffic capture with EdgeShark

Capturing traffic from the comfort of a web UI

- **EdgeShark** provides visibility on the communication in-between containers as well as with the outside world
- Coupled with Containerlab, it's a powerful tool to easily inspect traffic
- Try it – it just works!



Link impairment

Ain’t no network reliable enough

```
❯ clab tools netem set -n leaf1 -i e1-49 --delay 10ms --jitter 1ms
+-----------+-------+--------+-------------+-------------+
| Interface | Delay | Jitter | Packet Loss | Rate (kbit) |
+-----------+-------+--------+-------------+-------------+
| e1-49     | 10ms  | 1ms    | 0.00%       |           0 |
+-----------+-------+--------+-------------+-------------+
```

```
❯ clab tools netem set -n leaf1 -i e1-50 --loss 50
+-----------+-------+--------+-------------+-------------+
| Interface | Delay | Jitter | Packet Loss | Rate (kbit) |
+-----------+-------+--------+-------------+-------------+
| e1-50     | 0s    | 0s     | 50.00%      |           0 |
+-----------+-------+--------+-------------+-------------+
```

```
❯ clab tools netem set -n leaf1 -i mgmt0 --rate 500
+-----------+-------+--------+-------------+-------------+
| Interface | Delay | Jitter | Packet Loss | Rate (kbit) |
+-----------+-------+--------+-------------+-------------+
| mgmt0     | 0s    | 0s     | 0.00%       |         500 |
+-----------+-------+--------+-------------+-------------+
```

- Delay & jitter
- Packet loss
- Rate limiting

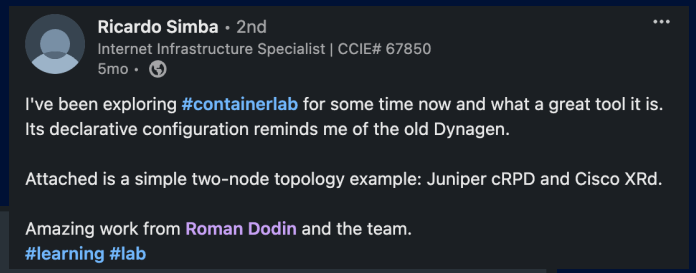
```
❯ clab tools netem show -n leaf1
+-----------+-------+--------+-------------+-------------+
| Interface | Delay | Jitter | Packet Loss | Rate (kbit) |
+-----------+-------+--------+-------------+-------------+
| lo        | N/A   | N/A    | N/A         | N/A         |
| mgmt0     | 0s    | 0s     | 0.00%       |         500 |
| e1-1      | N/A   | N/A    | N/A         | N/A         |
| e1-49     | 10ms  | 1ms    | 0.00%       |           0 |
| e1-50     | 0s    | 0s     | 50.00%      |           0 |
| gway-2800 | N/A   | N/A    | N/A         | N/A         |
| monit_in  | N/A   | N/A    | N/A         | N/A         |
| mgmt0-0   | N/A   | N/A    | N/A         | N/A         |
| gway-2801 | N/A   | N/A    | N/A         | N/A         |
| e1-49-0   | N/A   | N/A    | N/A         | N/A         |
| e1-50-0   | N/A   | N/A    | N/A         | N/A         |
+-----------+-------+--------+-------------+-------------+
```

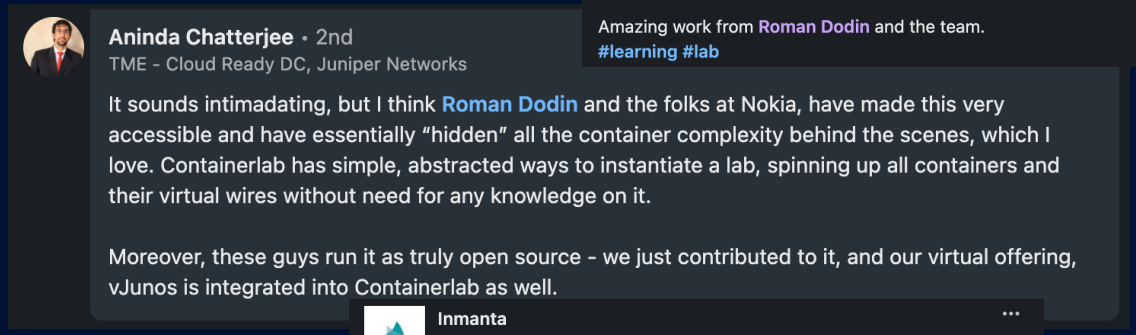
Recent Features

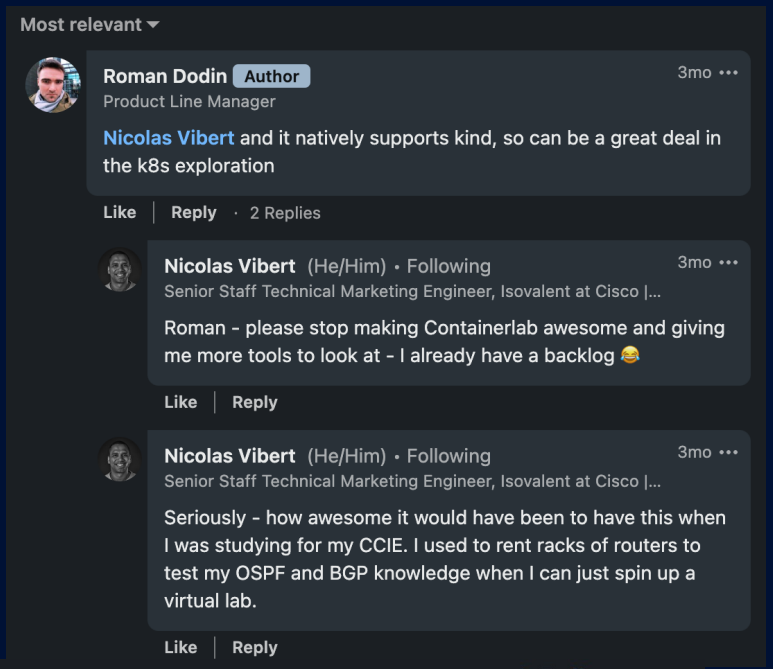
Last 12 months of development

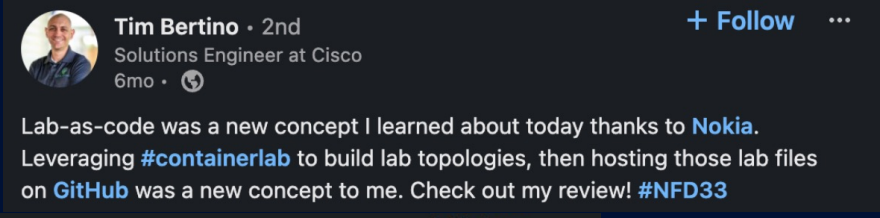
- Jan ’23 – Added DNS options for containers (servers, search paths etc.).
- Mar ‘23 – Remote start-up configs from HTTP/S server, Link MTU configurable.
- Apr ‘23 – Partial SR OS start-up configs, embedded SROS configurations.
- May ‘23 – Environnent variable support in topology file, mgmt IP range configurable.
- July ‘23 – MACVLAN Interfaces, Mermaid diagrams, override bind mounts.
- Aug ‘23 – Link Impairments, External CA.
- Sep ‘23 – Suppress startup config option.
- Oct ‘23 – Limited Apple M1/M2 support, Remote Labs, SSH Config file per-lab.
- Dec ‘23 – Healthchecks.
- Feb ‘24 – Stages, Edgeshark integration.
- Mar ‘24 – Enhancements to stages, Draw.io diagram generator.

Containerlab
Community feedbacks









I'm blown away. This is exactly what I wanted from a lab environment. The speed and ease of getting to this point does feel like a game-changer. I know this will have it's specific audience - there will be many out there who don't want to look at YAML or understand containers, and many who may actually need to test hardware/software combos on physical kit, but what this offers to the rest of us is an amazing amount of flexibility, ability to skip the tedious stuff, and be able to jump in once the party has already started, rather than having to turn up early and hang the bunting yourself.

Containerlab

Try it, join the community

To be continued…

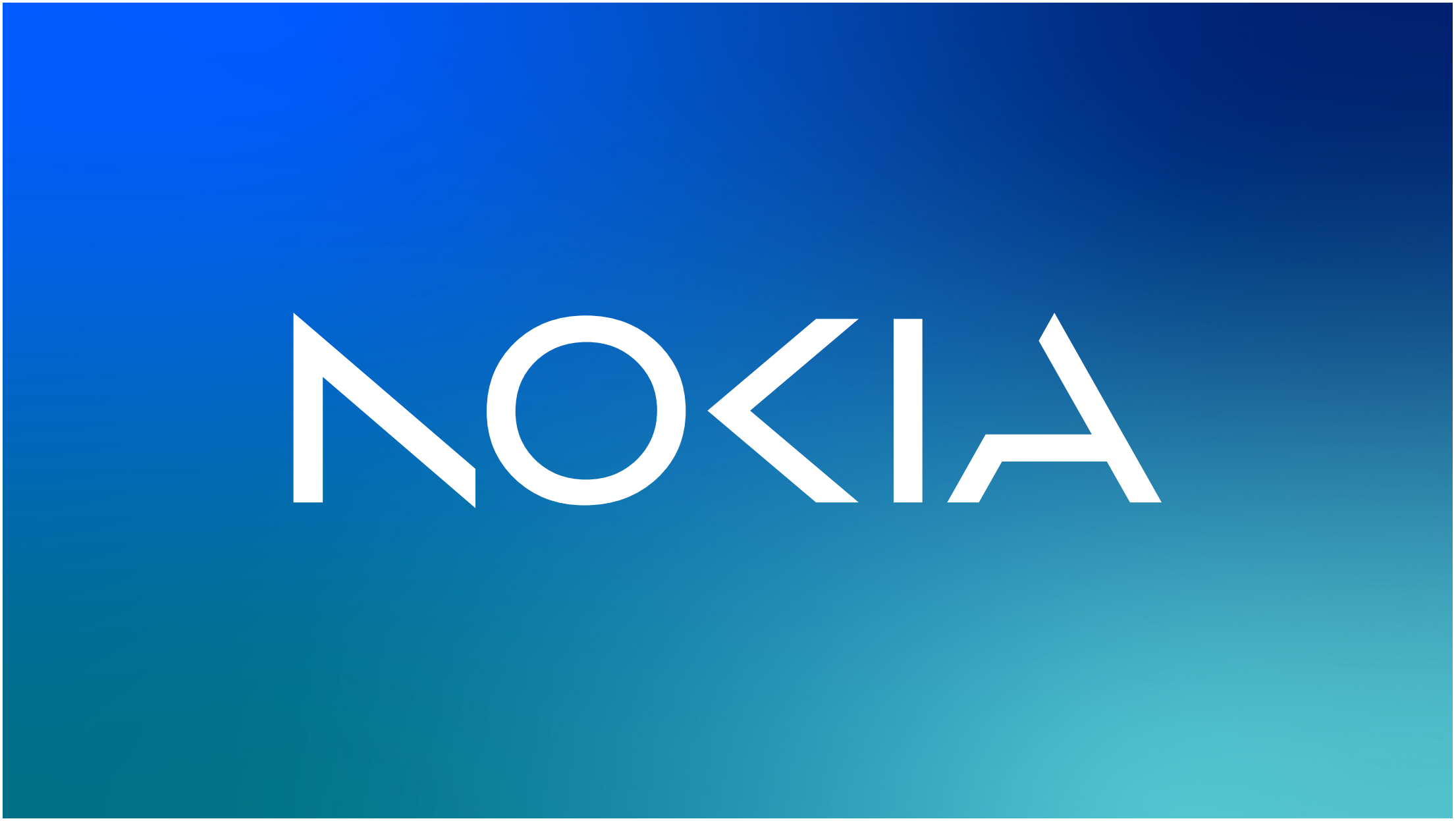
CONTAINERlab

containerlab.dev

Discord server

Containerlab repo
github.com/srl-labs/containerlab

Nokia lab topologies
github.com/srl-labs/


NOKIA