

NOKIA



CONTAINERlab

The power of containerization for network
testing and development

Thomas Corre – APAC IP Routing rPLM

MyNOG 11

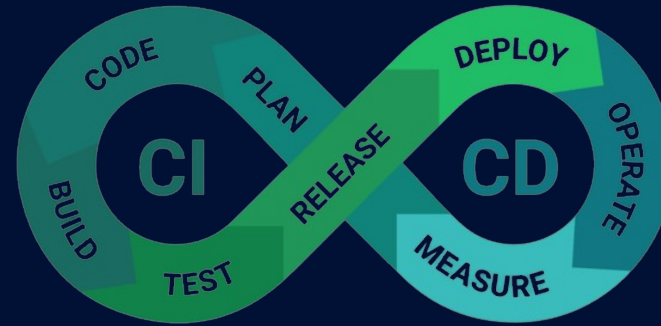
© 2024 Nokia | Public use

Network labs

A right, not a privilege



Education and Learning

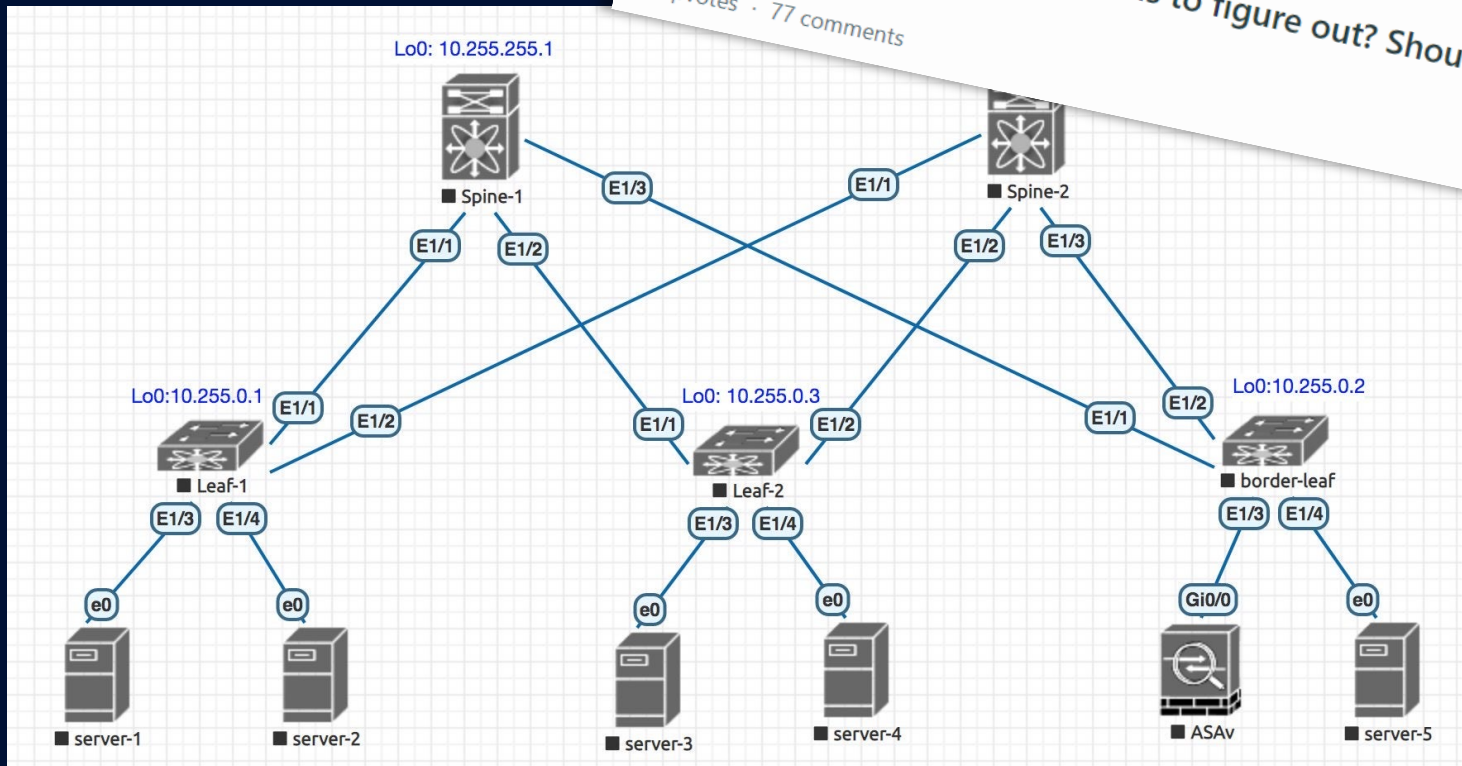


Change management and validation

Network labs

How do we typically run labs today?

r/networking · 7 months ago
Do all lab simulators take months to figure out? Should I just buy physical equipment??
84 upvotes · 77 comments

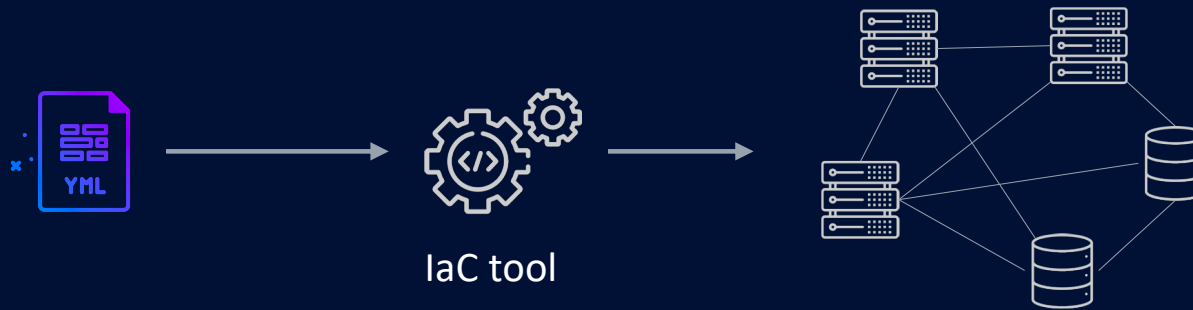


Pic from https://www.reddit.com/r/networking/comments/g5fb23/eveng_lab_strage_packet_loss/

Containerlab

Bringing declarativeness to network labs

IT



Network Labs

```
name: mylab
```

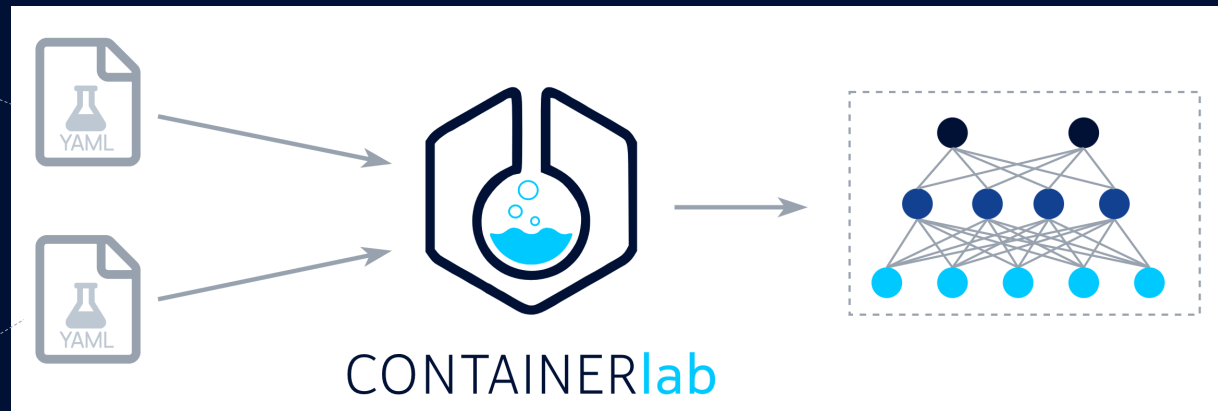
```
topology:
```

```
  nodes:
```

```
    [ ] : [ ]
    [ ] : [ ]
```

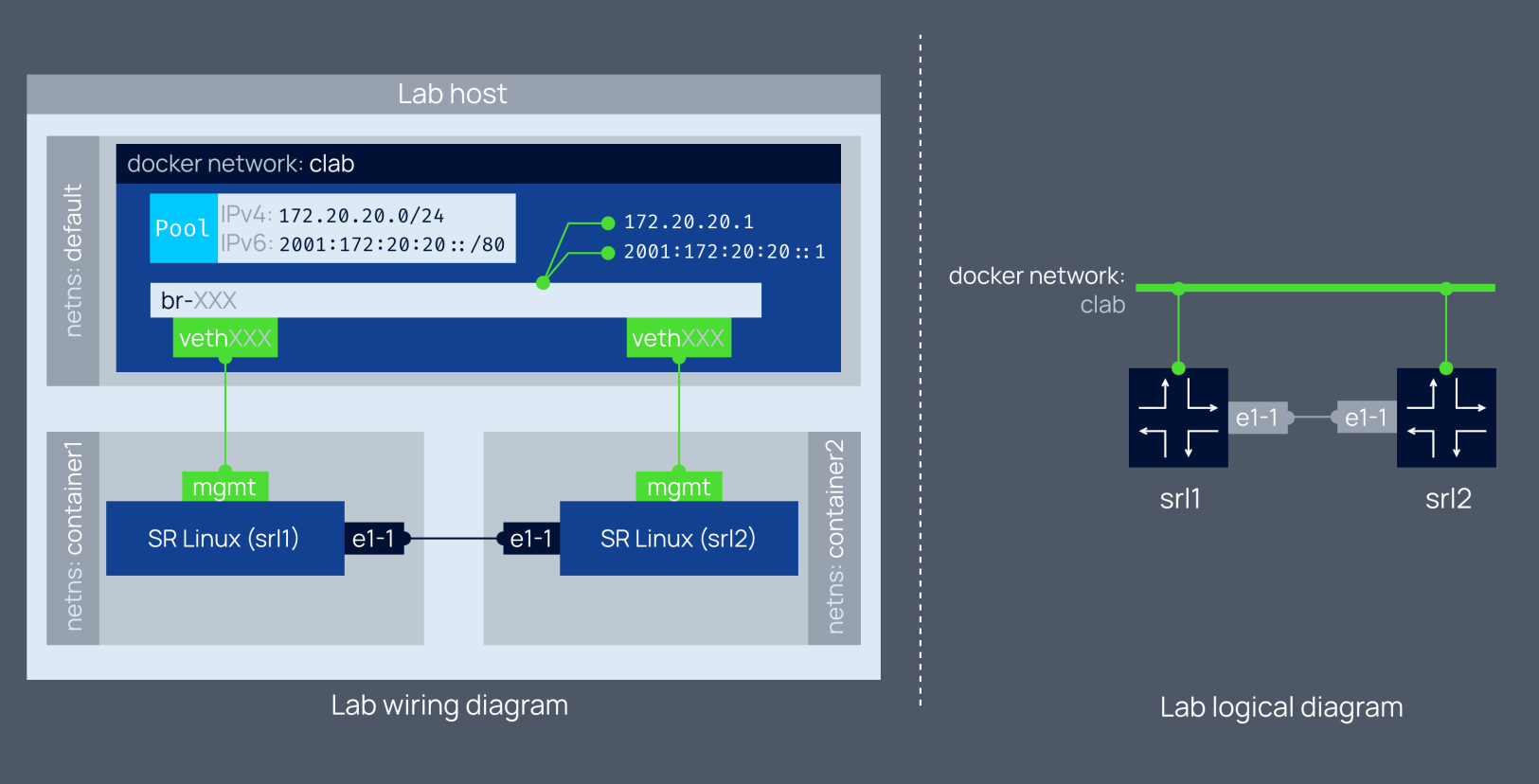
```
  links:
```

```
    - [ ] : [ ]
```



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

Supported NOSes

NOKIA

 SR Linux
 SROS

ipinfusion™

 OcNOS


NVIDIA

 Cumulus VX



 PAN


ARISTA

 cEOS
 vEOS



 Cloudguard



 ArubaOS-CX



 RouterOS


JUNIPER
NETWORKS

 cRPD
 vMX
 vQFX
 vSRX
 vJunos-switch
 vJunosEvolved

ixia

 Ixia-c-one



 OpenBSD



 XRd
 XRv9k
 XRv
 CSR1000v
 Nexus 9000v
 8000
 FTDv

FORTINET®

 FortiGateVM64

 Container-based NOS
 VM-based NOS



 SONiC VS
 FRR

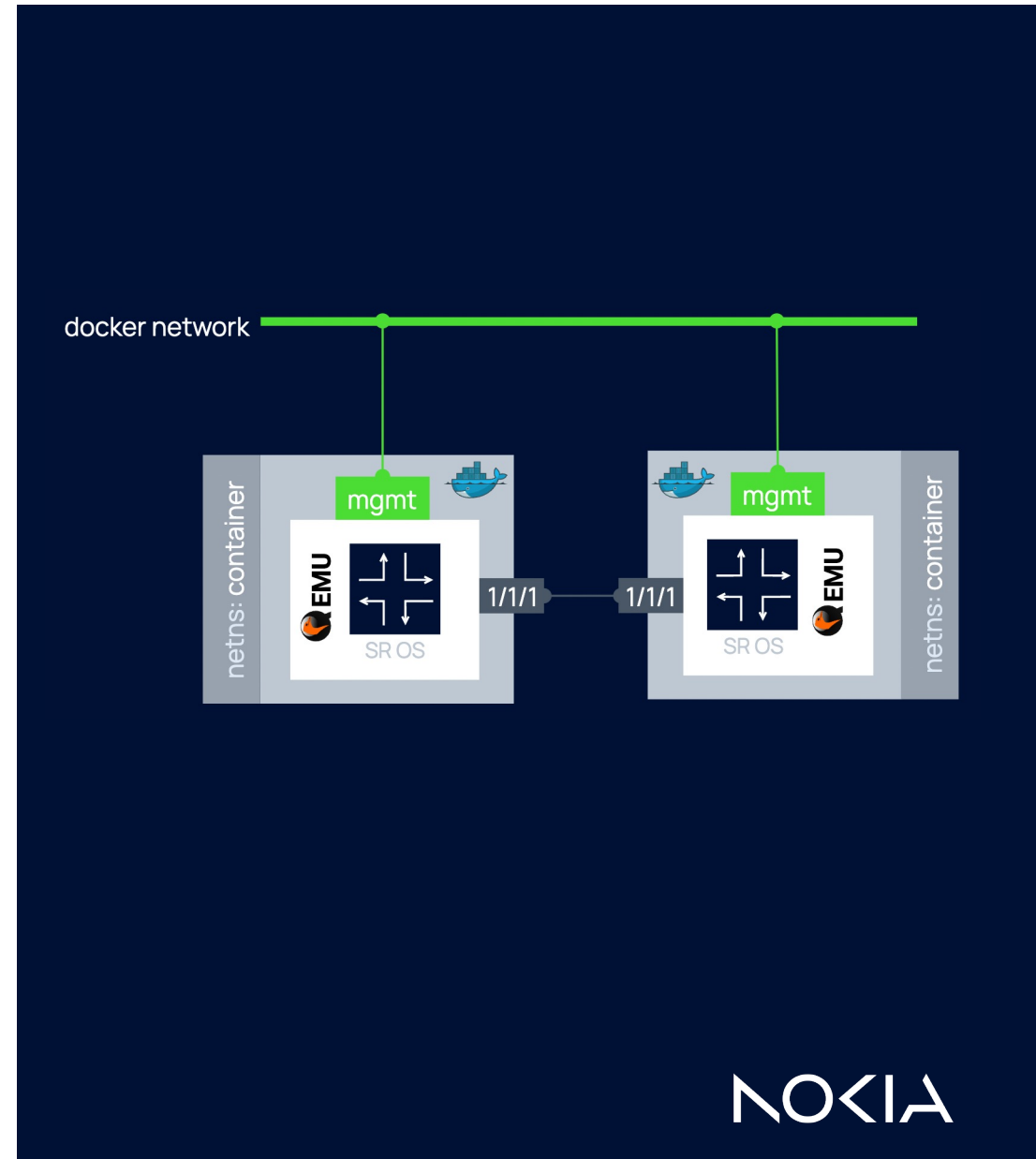


 FTOSv

Containerlab node types

Virtual machines in a container package

- Traditional Network OS packaged as a VM
- Integrated with containerlab through [vnetlab](#) 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

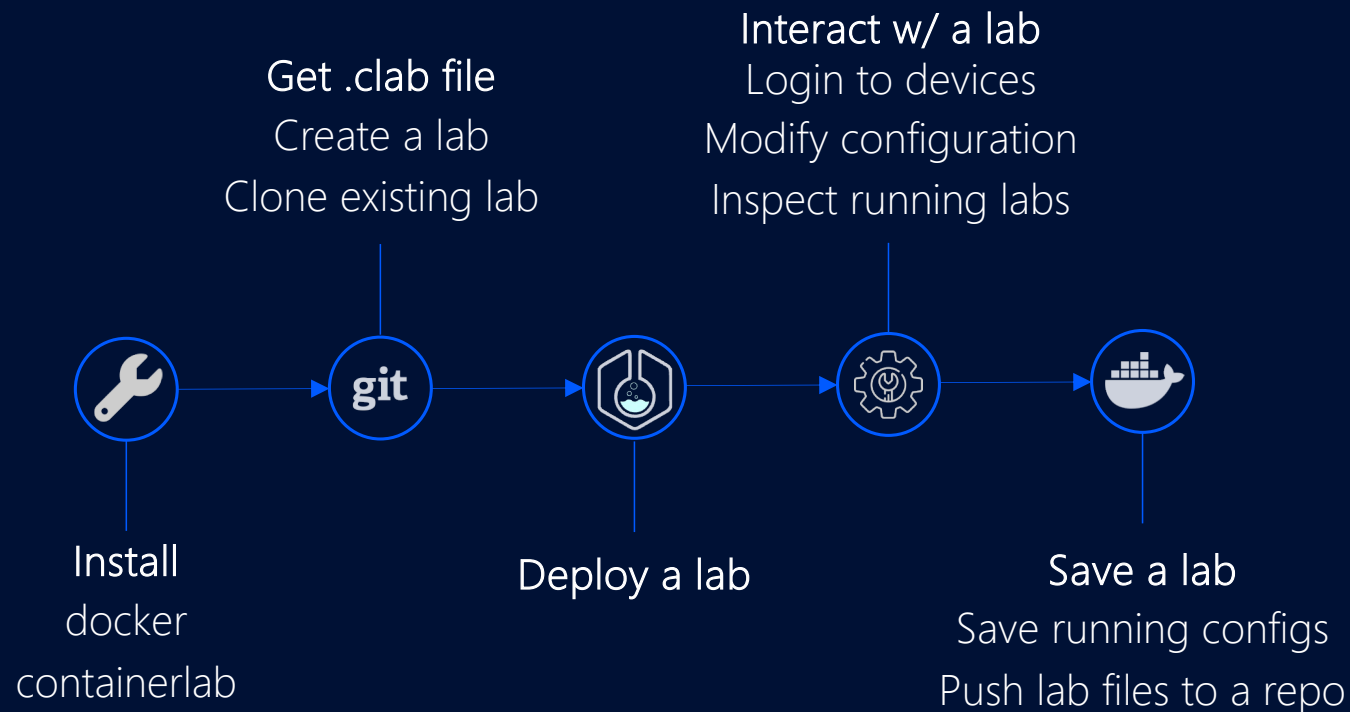
Regular container images

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



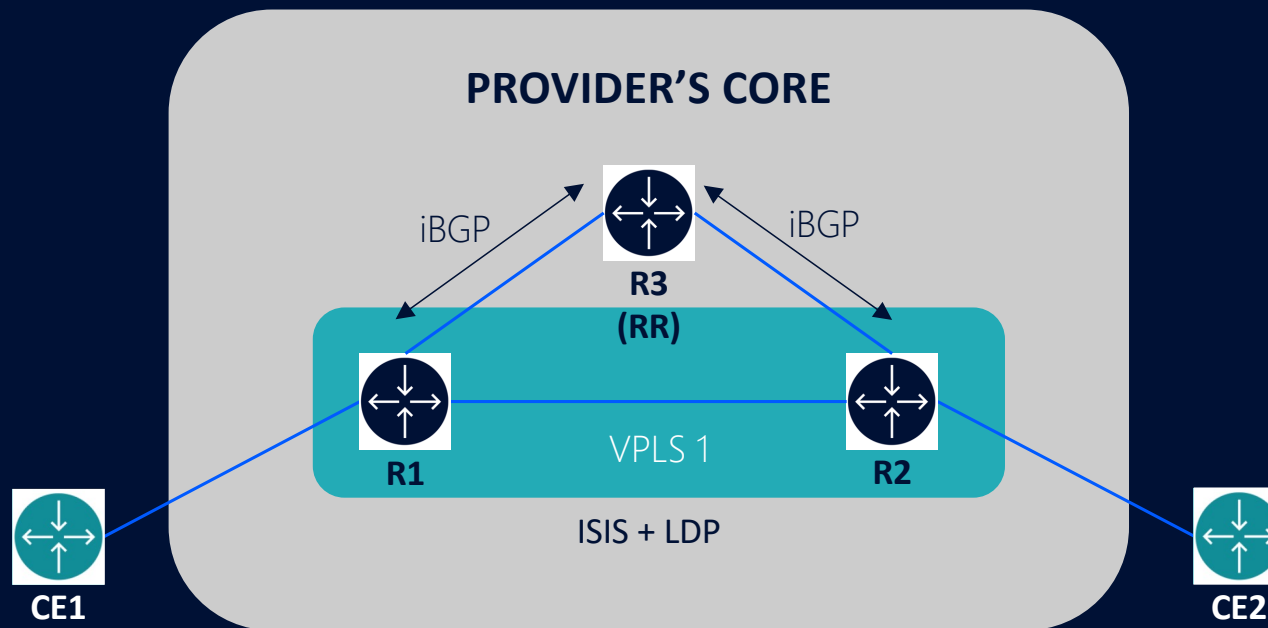
Up for an adventure ?

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/download/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

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

logical view

r1
(Nokia SR OS)



Building a service provider lab

Adding a Juniper VMX node

topology definition

```
name: vpls vpls.clab.yml
topology:
  nodes:
    r1: {...}

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

logical view



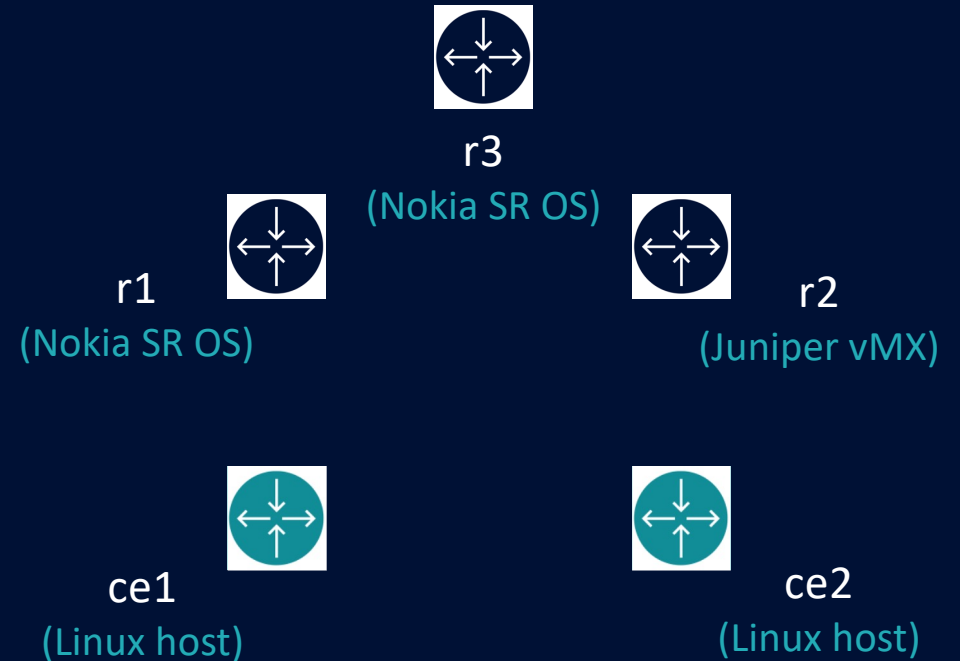
Building a service provider lab

Adding CE nodes

topology definition

```
name: vpls vpls.clab.yml
topology:
  nodes:
    r1: {...}
    r2: {...}
    r3: {...}
    ce1: {...}
    ce2:
      kind: linux
      image: ghcr.io/hellt/network-multitool
```

logical view



Building a service provider lab

Adding links

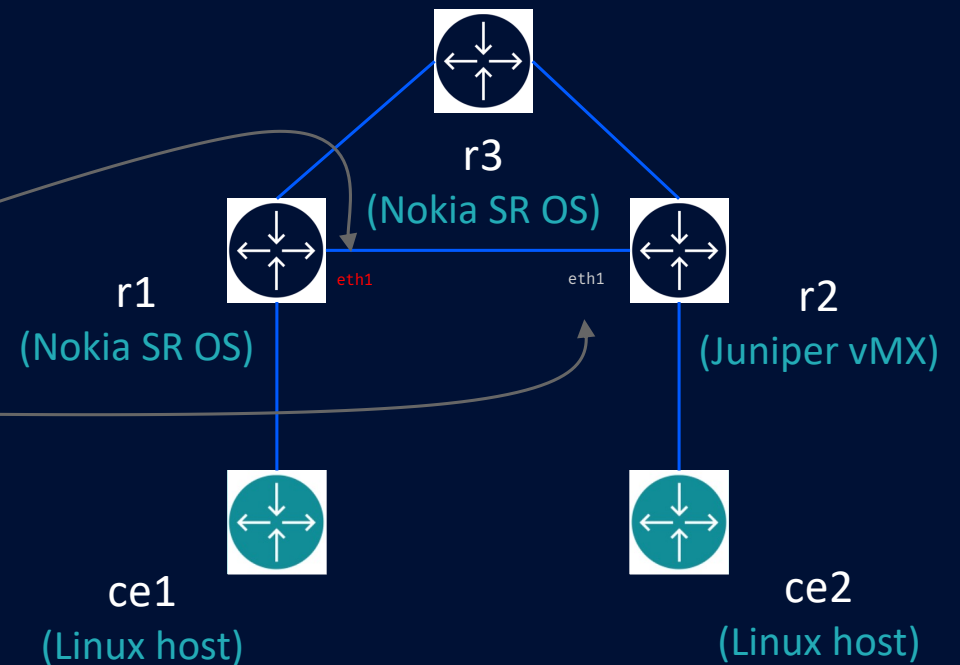
topology definition

```
name: vpls vpls.clab.yml

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

NOKIA

Building a service provider lab Deployment

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

Building a service provider lab

Connecting to the nodes

SSH

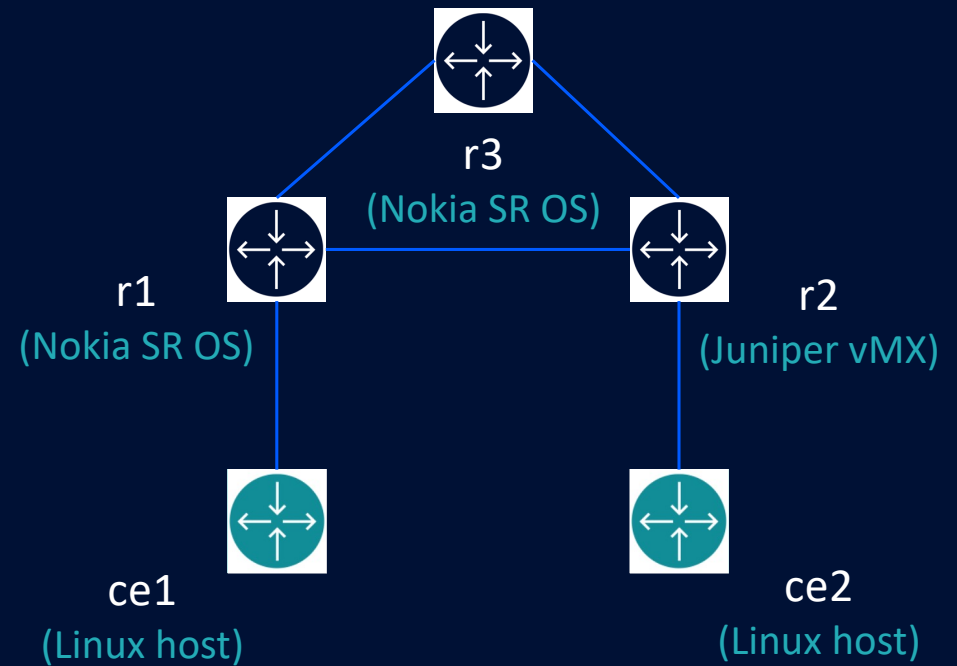
```
ssh admin@clab-vpls-r1  
admin@clab-vpls-r1's password:  
[/  
A:admin@r1#
```

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

The screenshot shows a web browser window with the URL `https://containerlab.dev/lab-examples/bgp-vpls-nok-jun/`. The page title is "BGP VPLS between Nokia and Juniper". The page content includes a table of contents with links for "Table of contents", "Description", and "Quickstart". Below this is a table with the following data:

Description	BGP VPLS between Nokia SR OS and Juniper vMX
Components	Nokia SR OS, Juniper vMX
Resource requirements ¹	2 7-10 GB
Lab location	hell/bgp-vpls-lab
Topology file	vpls.clab.yml
Version Information ²	containerlab:0.10.1, vr-sros:20.10.R1, vr-vmx:20.4R1.12, docker-ce:19.03.13, vrnetlab ³

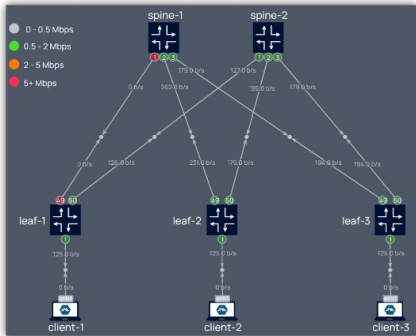
Below the table is a "Description" section with the text: "This lab demonstrates how containerlab can be used in a classical networking labs where the prime focus is not on the containerized NOS, but on a classic VM-based routers."



Other labs for the community



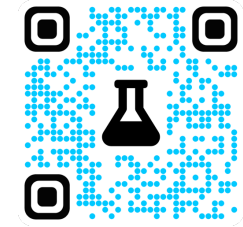
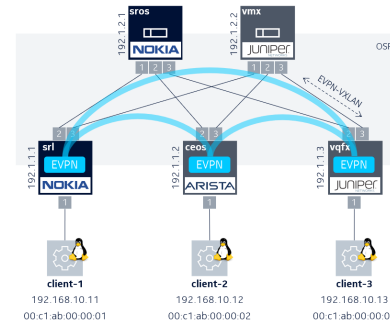
SR Linux Telemetry



[srl-labs/srl-telemetry-lab](https://github.com/srl-labs/srl-telemetry-lab)



Multivendor EVPN



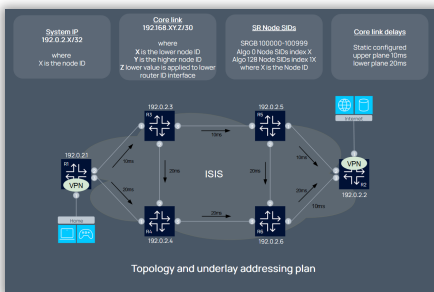
[srl-labs/multivendor-evpn-lab](https://github.com/srl-labs/multivendor-evpn-lab)



CONTAINERlab



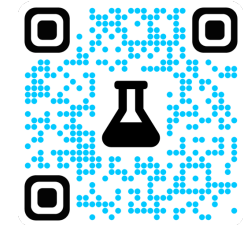
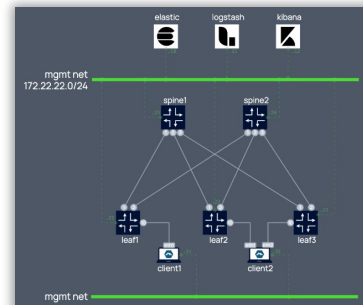
Segment routing



[srl-labs/nokia-segment-routing-lab](https://github.com/srl-labs/nokia-segment-routing-lab)



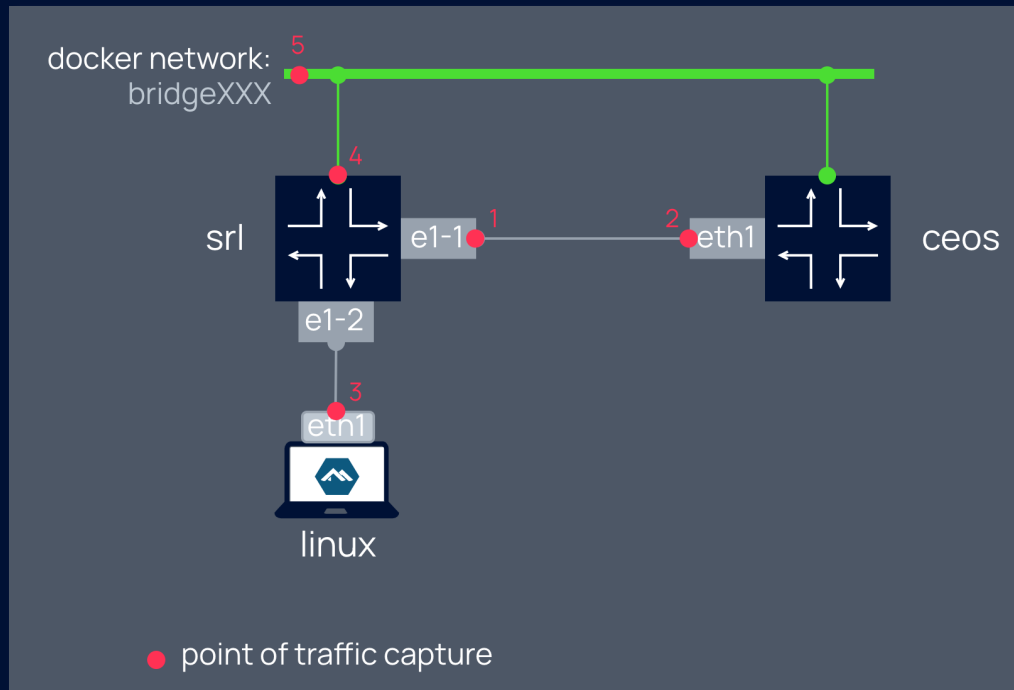
ELK Logging



[srl-labs/srl-elk-lab](https://github.com/srl-labs/srl-elk-lab)

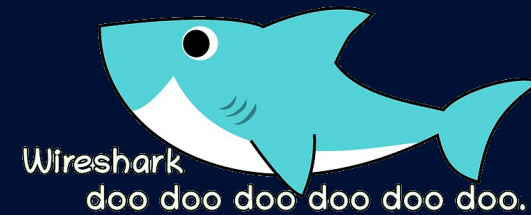
Traffic capture

Pcapng or it didn't happen



Command to capture at point #1

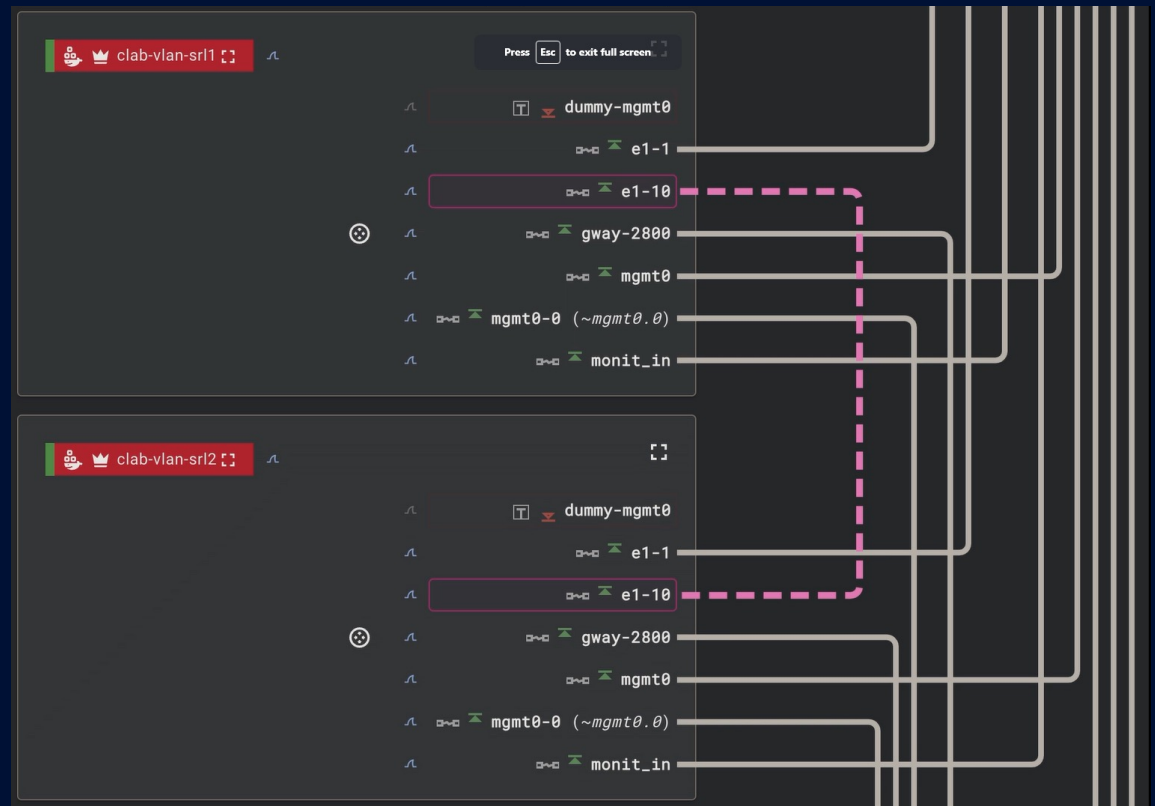
```
ssh $clab_host "ip netns exec $container tcpdump -U -nni e1-1 -w -" | wireshark -k -i -
```



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

- Delay & jitter
- Packet loss
- Rate limiting

```
> 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

```
> 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 – Environment 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



Aninda Chatterjee · 2nd
TME - Cloud Ready DC, Juniper Networks

It sounds intimidating, but I think **Roman Dodin** and the folks at Nokia, have made this very accessible and have essentially "hidden" all the container complexity behind the scenes, which I love. Containerlab has simple, abstracted ways to instantiate a lab, spinning up all containers and their virtual wires without need for any knowledge on it.

Moreover, these guys run it as truly open source - we just contributed to it, and our virtual offering, vJunos is integrated into Containerlab as well.



Inmanta
1,167 followers
8mo · 🌐

It's **#ThrowbackThursday**, and we're reminiscing about one of our favourite videos featuring a tool that remains an essential part of our daily use: <https://lnkd.in/eSYzQPcy>

Roman Dodin **#containerlab** **#tbt** **#networkautomation**

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.



Ricardo Simba · 2nd
Internet Infrastructure Specialist | CCIE# 67850
5mo · 🌐

I've been exploring **#containerlab** for some time now and what a great tool it is. Its declarative configuration reminds me of the old Dynagen.

Attached is a simple two-node topology example: Juniper cRPD and Cisco XRd.

Amazing work from **Roman Dodin** and the team.
#learning **#lab**

Most relevant ▾



Roman Dodin **Author**
Product Line Manager

3mo ...

Nicolas Vibert and it natively supports kind, so can be a great deal in the k8s exploration

Like | Reply · 2 Replies



Nicolas Vibert (He/Him) · Following
Senior Staff Technical Marketing Engineer, Isovalent at Cisco |...

3mo ...

Roman - please stop making Containerlab awesome and giving me more tools to look at - I already have a backlog 😊

Like | Reply



Nicolas Vibert (He/Him) · Following
Senior Staff Technical Marketing Engineer, Isovalent at Cisco |...

3mo ...

Seriously - how awesome it would have been to have this when I was studying for my CCIE. I used to rent racks of routers to test my OSPF and BGP knowledge when I can just spin up a virtual lab.

Like | Reply



Tim Bertino · 2nd
Solutions Engineer at Cisco
6mo · 🌐

+ Follow ...

Lab-as-code was a new concept I learned about today thanks to **Nokia**. Leveraging **#containerlab** to build lab topologies, then hosting those lab files on **GitHub** was a new concept to me. Check out my review! **#NFD33**



Suresh Vina · 2nd
Cloud Network Engineer at Motability Operations Ltd
2mo · 🌐

+ Follow ...

As promised, check out my new blog post on 'Containerlab'! I explain what it is, how it simplifies network lab creation, and share a hands-on example. Plus, I cover using VM-based nodes like Palo Alto. Take a look and let me know what you think!

<https://lnkd.in/e6XK5cQ2>

#ContainerLab **#networkengineer** **#networkautomation**

NOKIA

Containerlab

Try it, join the community



[Discord server](#)



CONTAINERlab

containerlab.dev



Containerlab repo

github.com/srl-labs/containerlab

Nokia lab topologies

github.com/srl-labs/



To be continued...

NOKIA