



netlab

Bringing the joy back to virtual networking labs

**Ivan Pepelnjak (ip@ipSpace.net)
Network Architect**

ipSpace.net AG

Who is Ivan Pepelnjak (@ioshints)

Past

- Kernel programmer, network OS, and web developer
- Sysadmin, database admin, network engineer, CCIE
- Trainer, course developer, curriculum architect
- Team lead, CTO, business owner

Present

- Network architect, consultant, blogger, webinar, and book author

Focus

- SDN and network automation
- Large-scale data centers, clouds, and network virtualization
- Scalable application design
- Core IP routing/MPLS, IPv6, VPN



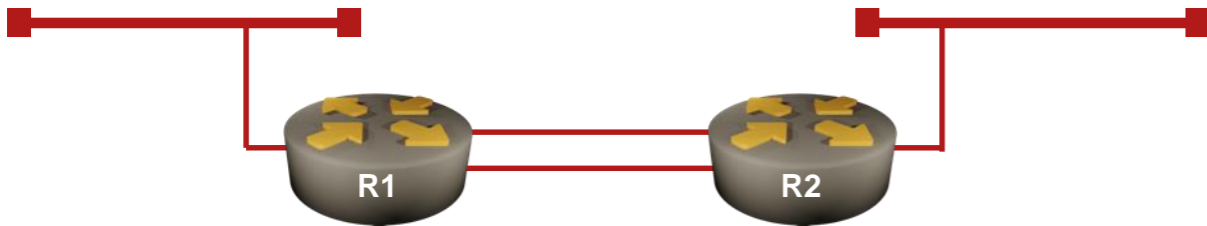
Also: I built too many labs for one lifetime, and hated that with passion

Based on a True Story

Pär Stolpe 01 September 2023 12:06

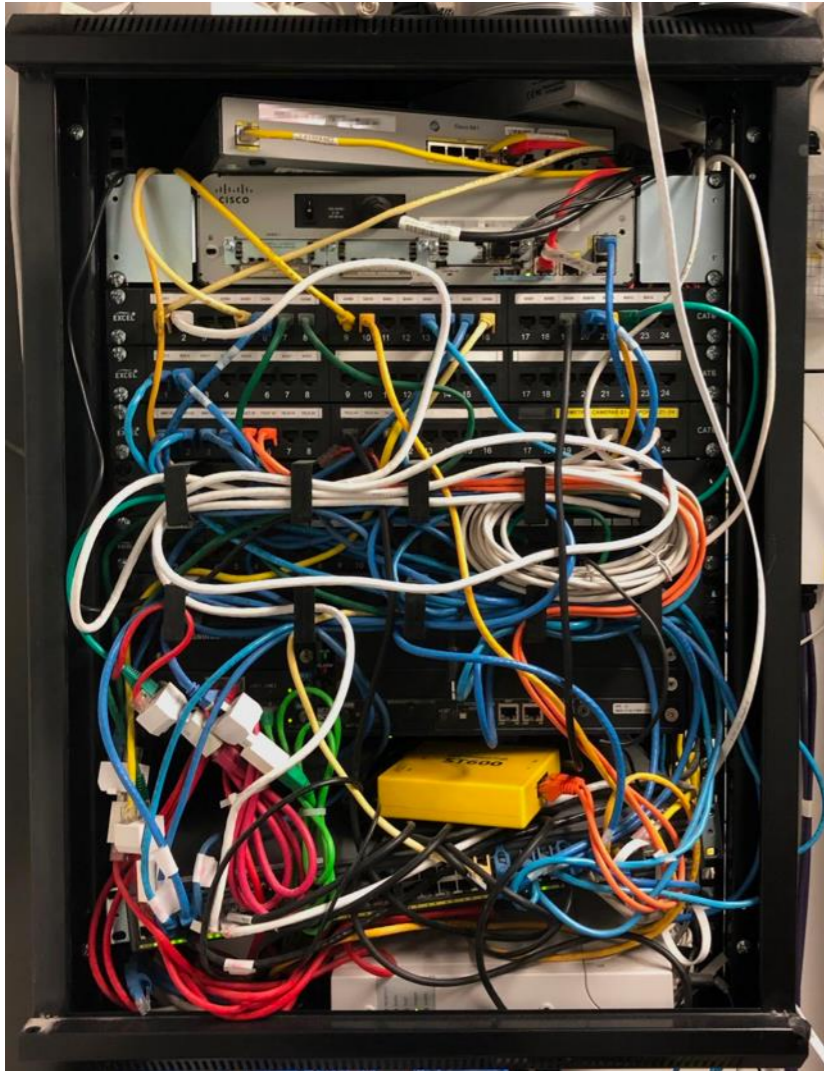
Beware of the fact that having more than one ospf link in between the same two nodes, together with unnumbered interfaces for multipathing purposes would most likely cause troubles. I don't know if any vendor have solved it or if they just recommend using link aggregation instead.

 **reply**



That should be trivial to test in a lab... However, someone has to build that lab...

The Reality Intervenes...



VM Maestro

100%

Design Simulation

Projects

My Topologies

- routeagen.virl
- routeserver.virl
- SpringBoard.virl

Thbbft!

virl:topology

LXC Jumphost

```

graph TD
    Seattle[Seattle 192.168.0.1] --- Boston[Boston 192.168.0.4]
    Seattle --- London[London 192.168.0.9]
    Seattle --- Berlin[Berlin 192.168.0.12]
    Boston --- London
    Boston --- Berlin
    Boston --- Denver[Denver 192.168.0.5]
    Boston --- Atlanta[Atlanta 192.168.0.2]
    London --- Berlin
    London --- Paris[Paris 192.168.0.11]
    London --- Vienna[Vienna 192.168.0.10]
    Berlin --- Paris
    Berlin --- Vienna
    Berlin --- Milan[Milan 192.168.0.13]
    Denver --- Atlanta
    Denver --- Dallas[Dallas 192.168.0.3]
    Atlanta --- Paris
    Atlanta --- Vienna
    Atlanta --- Milan
    Dallas --- Paris
    Dallas --- Vienna
    Dallas --- Milan
  
```

Simulations

Last updated: Sun Jul 10 10:40:35 PDT 2016

guest

Thbbft!

- Atlanta [ACTIVE]
- Berlin [ACTIVE]
- Boston [ACTIVE]
- Dallas [ABSENT]
- Denver [ACTIVE]
- London [ACTIVE]
- Milan [ACTIVE]
- Paris [ACTIVE]
- Seattle [ACTIVE]
- Vienna [ACTIVE]

~lxc-flat

- External Address [172.16.1.51]
- Forwarding Port on Server [10000]
- ~mgmt-lxc interface [eth0]
- ~mgmt-lxc [ACTIVE]

Console

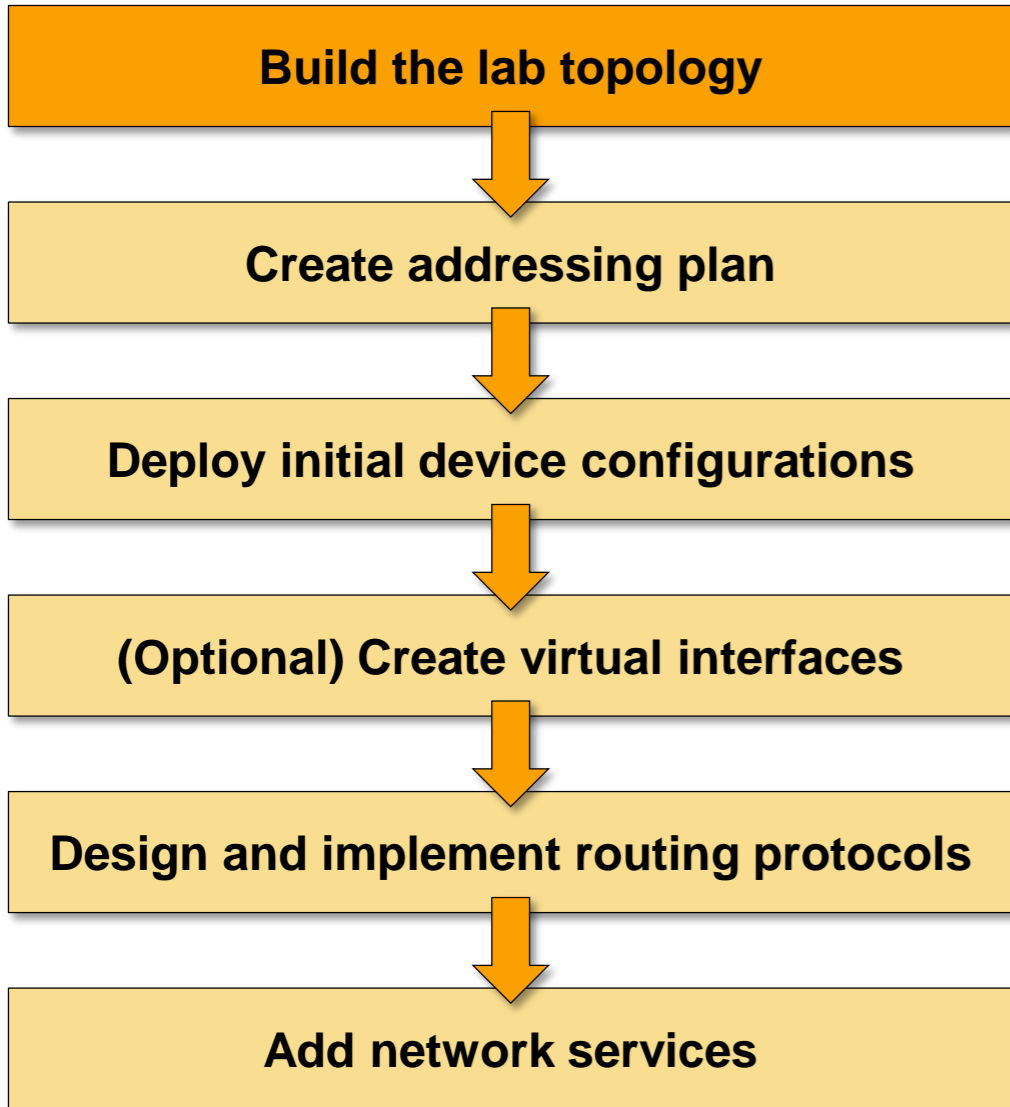
Unknown simulation Thbbft!

```

(INFO) [Jul/10/2016 16:46:30] Starting node "~mgmt-lxc"
(INFO) [Jul/10/2016 16:46:45] Node "Berlin" state changed from BUILDING to ACTIVE
(INFO) [Jul/10/2016 16:46:45] Node "Boston" state changed from BUILDING to ACTIVE
(INFO) [Jul/10/2016 17:39:47] Stopping node "Dallas"
(INFO) [Jul/10/2016 17:40:03] Node "Dallas" state changed from ACTIVE to ABSENT
  
```

guest

A Networking Lab Is Much More than Topology





And now for something completely different

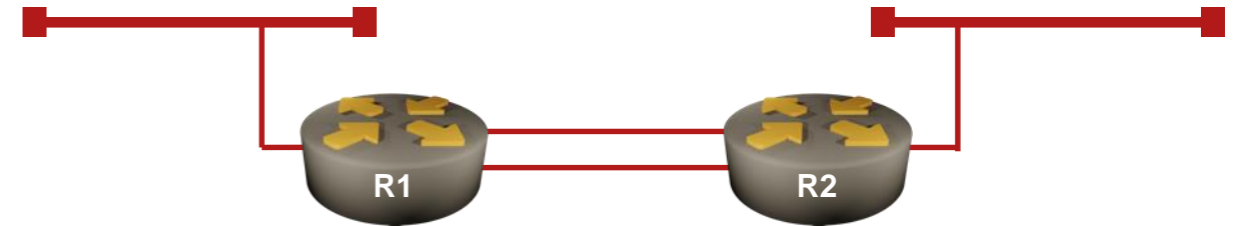
What Do We Need (in Unicorn Land of Infrastructure-as-Code)

Create a high-level description of the network

- Two devices: R1 and R2
- Let's make them Arista EOS containers
- They are running OSPF
- We need four links (two of them stub LANs)
- Oh, we're running unnumbered links...

Next

- Save the file
- Execute **netlab up** and you'll get a running network (including IP addressing and OSPF)



topology.yml

```
nodes: [ r1, r2 ]

defaults.device: eos
provider: clab

module: [ ospf ]

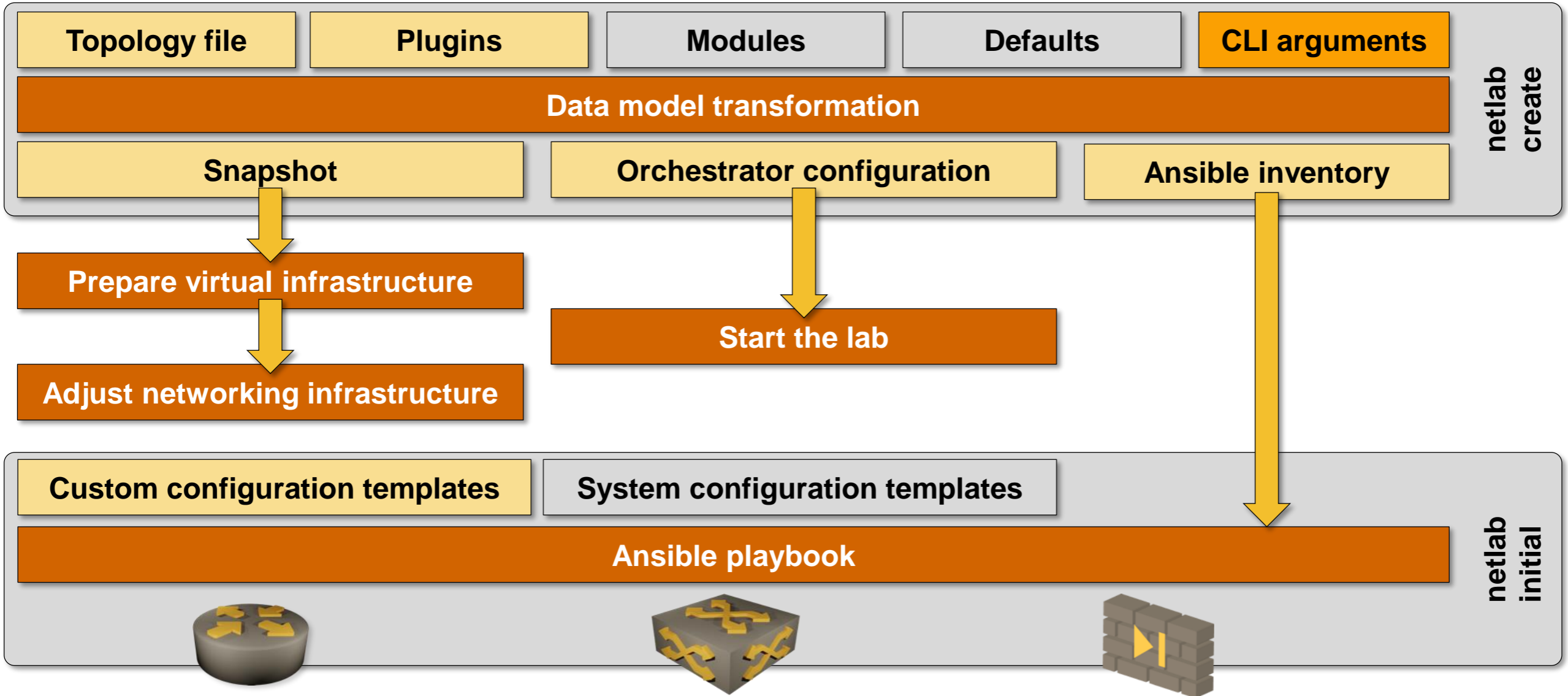
links: [ r1, r2, r1-r2, r1-r2 ]

addressing.p2p.ipv4: True
```

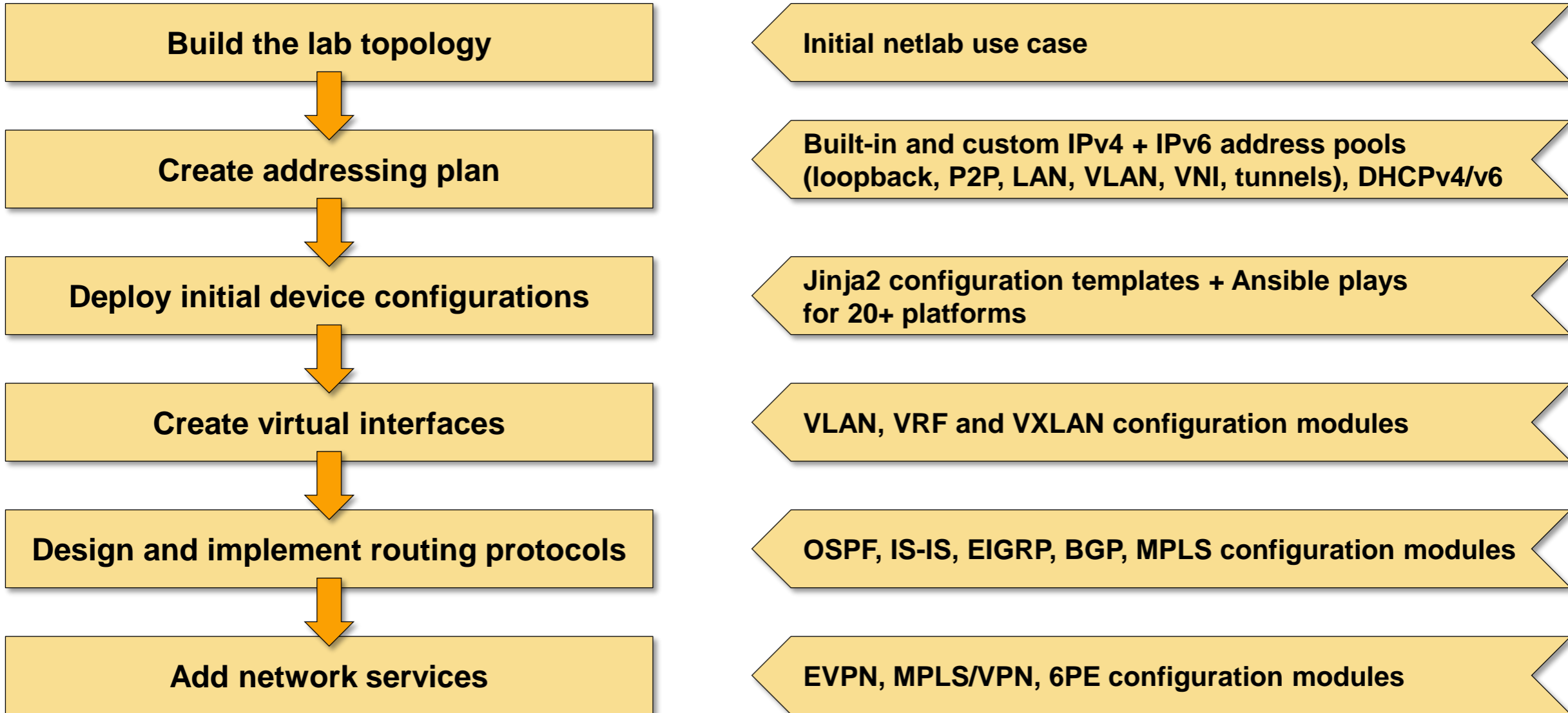
```
$ netlab up -p clab -d eos parallel.yml █
```

- Create configuration files
- Start the containers
- Start an Ansible playbook
- Initial device configuration
- Configuring OSPF
- Connect to the device
- ... and we have the answer!

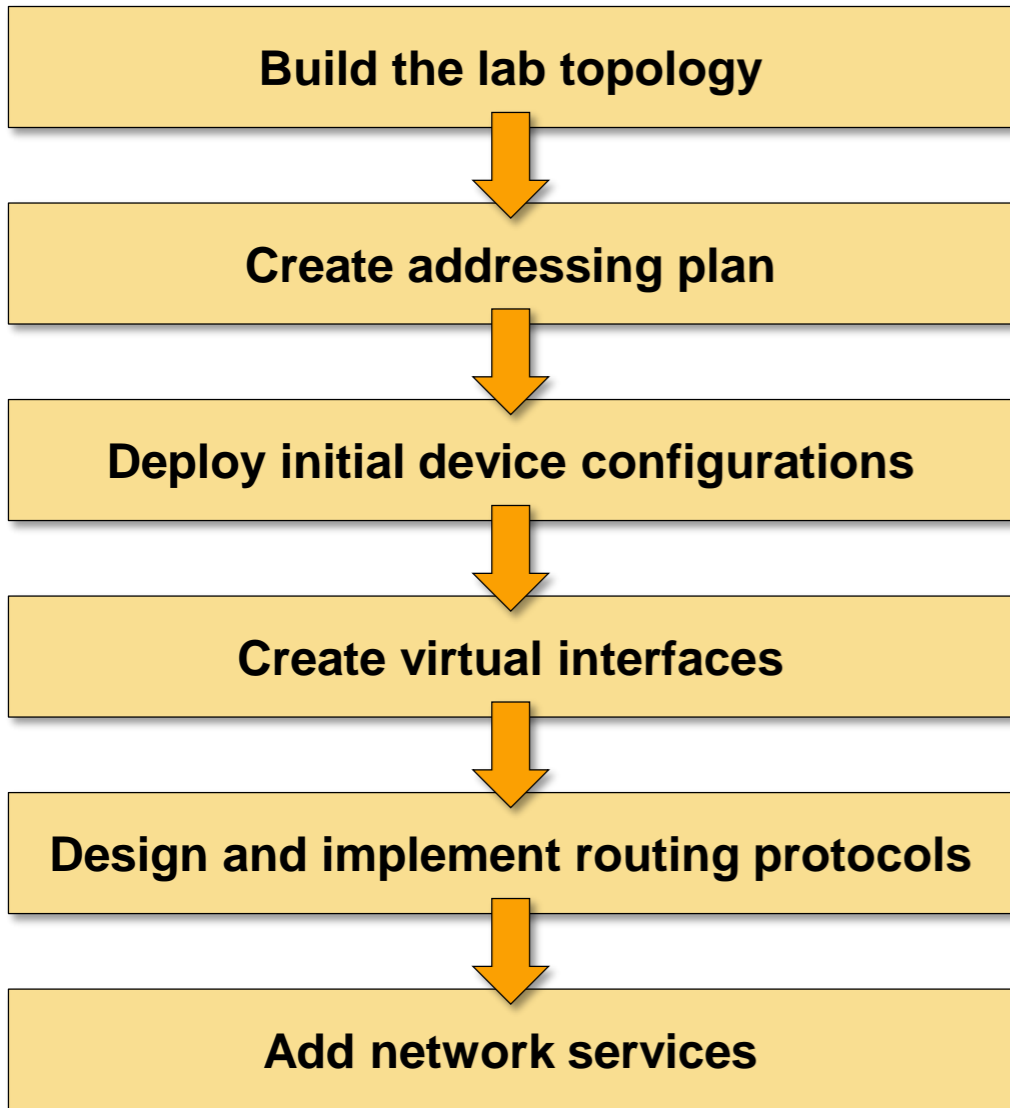
Wait, What Just Happened? netlab up Behind the Scenes



Building a Networking Lab with netlab



Every Lab Is Special



Add custom configuration

- Static configuration templates
- Configuration templates modifying built-in configurations
- Multi-vendor and multi-platform support

Modify data transformation with plugins

- Modify the built-in transformation rules
- Add new functionality (example: IP anycast)
- Add new attributes or functionality to existing configuration modules (example: BGP allowas-in)

Current State of netlab (April 2024)

Network devices

- Arista vEOS/cEOS
- Aruba CX
- Cisco ASAv, IOSv, IOS XE (CSR), Nexus OS (9300v), IOS XR/XRd
- Cumulus Linux 4.x and 5.x (NVUE)
- Dell OS10
- Fortinet
- FRR
- Juniper vSRX 3.0, vMX, vPTX (vEVO)
- Mikrotik RouterOS 6 and 7
- Nokia SR Linux and SR OS
- VyOS 1.4 and 1.5

Hosts and daemons

- Generic Linux host or container
- BIRD
- dnsmasq

Virtualization providers

- KVM with libvirt (Vagrant)
- Docker (containerlab)
- Hardware labs (requires extra interface information)
- VirtualBox (Vagrant) (deprecated)

Multi-provider topologies

- Combine containers and virtual machines in the same lab
- Connect external devices with the virtual lab

Current State of netlab: Behind the Scenes

Network devices

- Arista vEOS/cEOS
- Aruba CX
- Cisco ASAv, IOSv, IOS XE (CSR), Nexus OS (9300v), IOS XR/XRd
- Cumulus Linux 4.x and 5.x (NVUE)
- Dell OS10
- Fortinet
- FRR
- Juniper vSRX 3.0, vMX, vPTX (vEVO)
- Mikrotik RouterOS 6 and 7
- Nokia SR Linux and SR OS
- VyOS 1.4 and 1.5

The Heroes



Stefano Sasso

Aruba, Junos, Dell OS10,
Mikrotik, VyOS



Jeroen van Bommel

FRR, Nokia SR Linux,
Nokia SR OS

Current State of netlab (April 2024)

Addressing

- IPv4 + IPv6
- Address pools + static prefixes
- VLAN-wide subnets
- Static interface addresses
- Unnumbered IPv4 and IPv6 (LLA) interfaces
- Layer-2-only interfaces
- DHCP (clients, servers, relays)

Data Plane

- VLANs and VRFs
- VXLAN (static ingress replication or EVPN)
- MPLS including SR-MPLS
- SRv6
- Tunnel interfaces

Routing Protocols

- OSPFv2 and OSPFv3
- IS-IS
- EIGRP
- BGP
- BFD
- VRRP and anycast gateways

MPLS Control Plane

- LDP, BGP-LU, SR-MPLS (OSPF or IS-IS)

Network Virtualization

- MPLS L3VPN and 6PE
- EVPN (bridging, VLAN bundles, asymmetric and symmetric IRB, most combinations of IGP and BGP)

Sample Platform Support

Operating system	Hostname	IPv4 hosts	LLDP	Loopback IPv4 address	Loopback IPv6 address
Arista EOS	✓	✓	✓	✓	✓
Aruba AOS-CX	✓	✗	✓	✓	✓
Cisco ASA v	✓	✓	✗	✗	✗
Cisco IOS/IOS XE	✓	✓	✓	✓	✓
Cisco IOS XRv	✓	✓	✓	✓	✓
Cisco Nexus OS	✓	✓	✓	✓	✓
Cumulus Linux	✓	✓	✓	✓	✓
Cumulus Linux 5.0 (NVUE)	✓	✓	✓	✓	✓
Dell OS10	✓	✓	✓	✓	✓
Fortinet FortiOS	✓	✗	✓	✓	✓
FRR	✓	✓	✗	✓	✓
Generic Linux	✓	✓	✓!	✓	✓
Juniper vMX	✓	✗	✓	✓	✓
Juniper vPTX	✓	✗	✓	✓	✓
Juniper vSRX 3.0	✓	✗	✓	✓	✓
Mikrotik RouterOS 6	✓	✓	✓!	✓	✓
Mikrotik RouterOS 7	✓	✓	✓!	✓	✓
Nokia SR Linux	✓	✓	✓	✓	✓
Nokia SR OS	✓	✓	✓	✓	✓
VyOS	✓	✓	✓	✓	✓

Operating system	OSPF	IS-IS	EIGRP	BGP	BFD	EVPN	FHRP
Arista EOS	✓	✓	✗	✓	✓	✓	✓
Aruba AOS-CX	✓	✗	✗	✓	✓	✓	✓
Cisco ASA v	✗	✓	✗	✓	✗	✗	✗
Cisco IOSv	✓	✓	✓	✓	✓	✗	✓
Cisco IOS XE	✓	✓	✓	✓	✓	✗	✓
Cisco IOS XRv	✓	✓	✗	✓	✗	✗	✗
Cisco Nexus OS	✓	✓	✓	✓	✓	✓	✓
Cumulus Linux	✓	✗	✗	✓	✗	✓	✓
Cumulus Linux 5.0 (NVUE)	✓	✗	✗	✓	✗	✗	✗
Dell OS10	!	✗	✗	✓	✓	✓	✗
Fortinet FortiOS	!	✗	✗	✗	✗	✗	✗
FRR	✓	✓	✗	✓	✗	✓	✗
Juniper vMX	✓	✓	✗	✓	✓	✗	✗
Juniper vPTX	✓	✓	✗	✓	✓	✗	✗
Juniper vSRX 3.0	✓	✓	✗	✓	✓	✗	✗
Mikrotik RouterOS 6	✓	✗	✗	✓	✓	✗	✗
Mikrotik RouterOS 7	✓	✗	✗	✓	✓	✗	✗
Nokia SR Linux	✓	✓	✗	✓	✓	✓	✓
Nokia SR OS	✓	✓	✗	✓	✓	✓	✓
VyOS	✓	✓	✗	✓	✓	✓	✗

But Wait, There's More


External Connectivity

- Macvtap libvirt interfaces
- Macvlan container interfaces
- Port forwarding for VMs and containers

External Tools

- SuzieQ
- Graphite

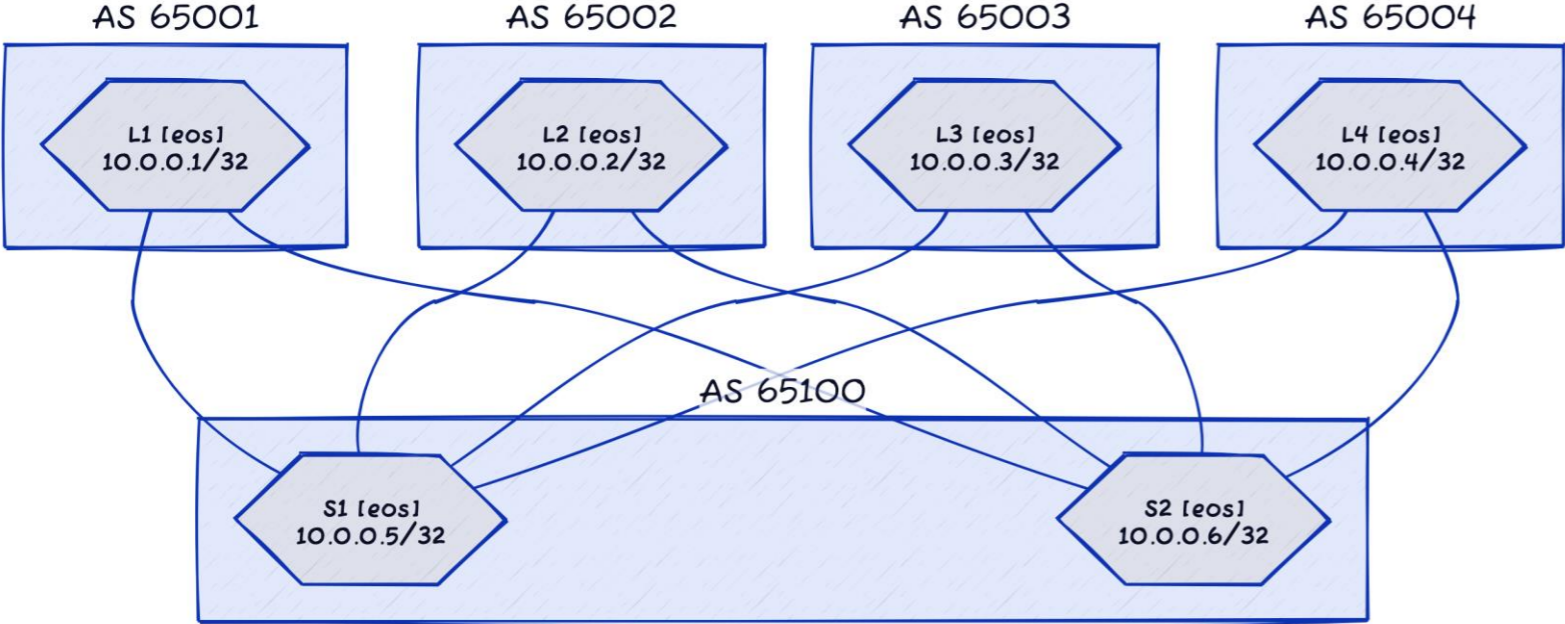
Ease-of-use

- Wiring, addressing, OSPF, and BGP reports (text, Markdown, HTML)
- Graphs (Graphviz or D2)
- Graphite GUI 
- Automated validation
- Multiple lab instances on a single server
- Restore previous configurations

Large topologies

- Topology templates
- Staggered device start
- Link groups

Example: Graph (D2)



Example: Addressing Report (BGP)

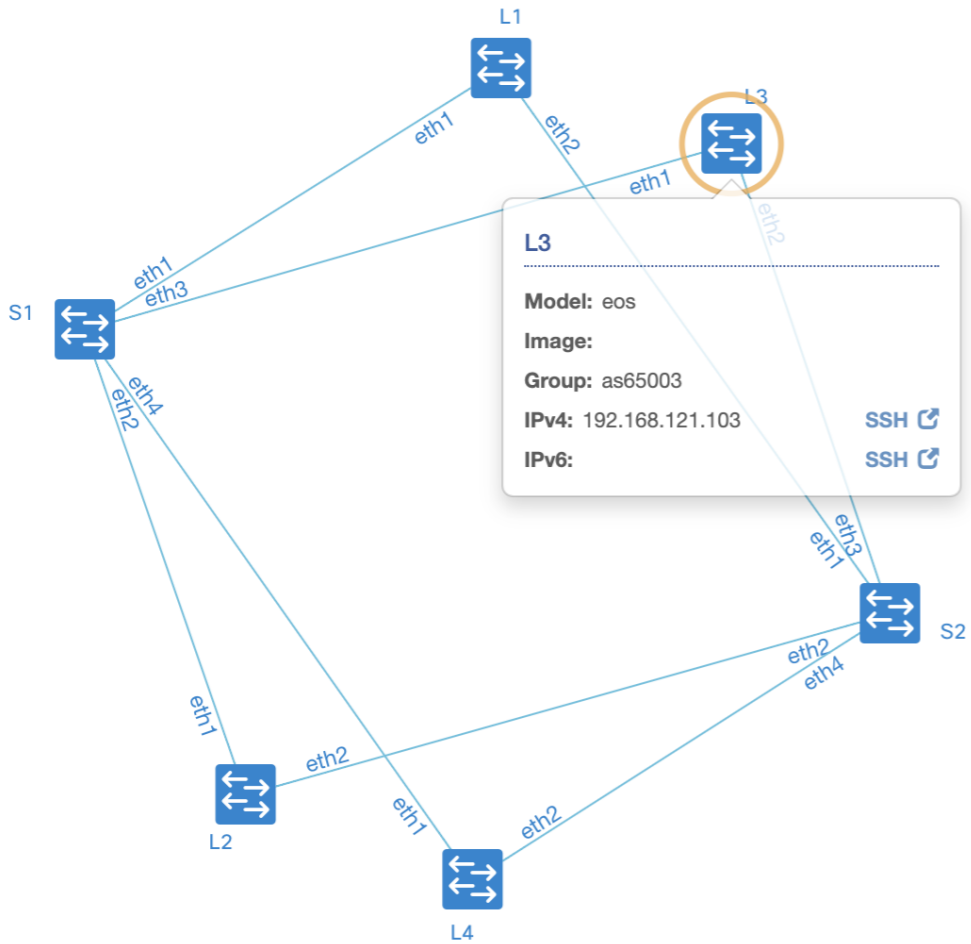
BGP AS Numbers

NODE/ASN	ROUTER ID	ADVERTISED PREFIXES
AS65001		
L1	10.0.0.1	10.0.0.1/32
AS65002		
L2	10.0.0.2	10.0.0.2/32
AS65003		
L3	10.0.0.3	10.0.0.3/32
AS65004		
L4	10.0.0.4	10.0.0.4/32
AS65100		
S1	10.0.0.5	10.0.0.5/32
S2	10.0.0.6	10.0.0.6/32

BGP Neighbors

NODE	NEIGHBOR	NEIGHBOR AS	NEIGHBOR IPV4
L1 (10.0.0.1 / AS 65001)			
	S1	65100	10.1.0.2
	S2	65100	10.1.0.6
L2 (10.0.0.2 / AS 65002)			
	S1	65100	10.1.0.10
	S2	65100	10.1.0.14
L3 (10.0.0.3 / AS 65003)			
	S1	65100	10.1.0.18
	S2	65100	10.1.0.22
L4 (10.0.0.4 / AS 65004)			
	S1	65100	10.1.0.26
	S2	65100	10.1.0.30

Example: Graphite GUI



Example: Network Validation

```
$ netlab validate
[session] Check IPv6 EBGP session with RTR on ISP routers [ node(s): x1,x2 ]
[PASS] Validation succeeded on x1
[PASS] Validation succeeded on x2
[PASS] The IPv6 EBGP session with RTR is in Established state

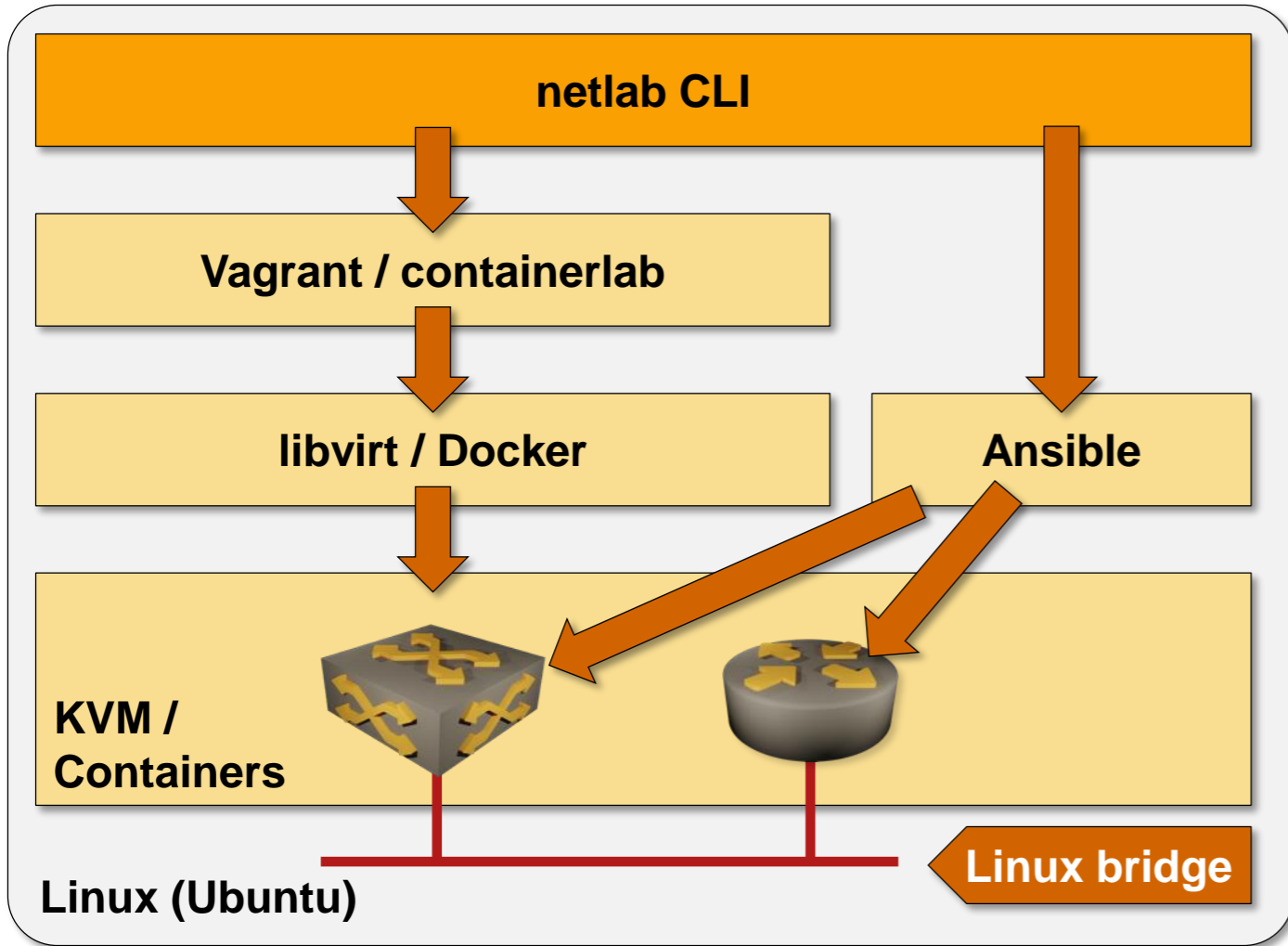
[pfxcnt] Check whether RTR receives and sends IPv6 prefixes [ node(s): x1,x2 ]
[PASS] Validation succeeded on x1
[PASS] Validation succeeded on x2
[PASS] RTR is advertising IPv6 prefixes to ISP routers

[advroute] Check whether RTR advertises 2001:db8:1::/48 [ node(s): x1,x2 ]
[PASS] Validation succeeded on x1
[PASS] Validation succeeded on x2
[PASS] RTR is advertising 2001:db8:1::/48 to ISP routers

[SUCCESS] Tests passed: 6
$ █
```

Deployment Scenarios

Recommended: Ubuntu, KVM, libvirt, Docker



Prerequisite software

- Python3
- Ansible (to configure the devices)

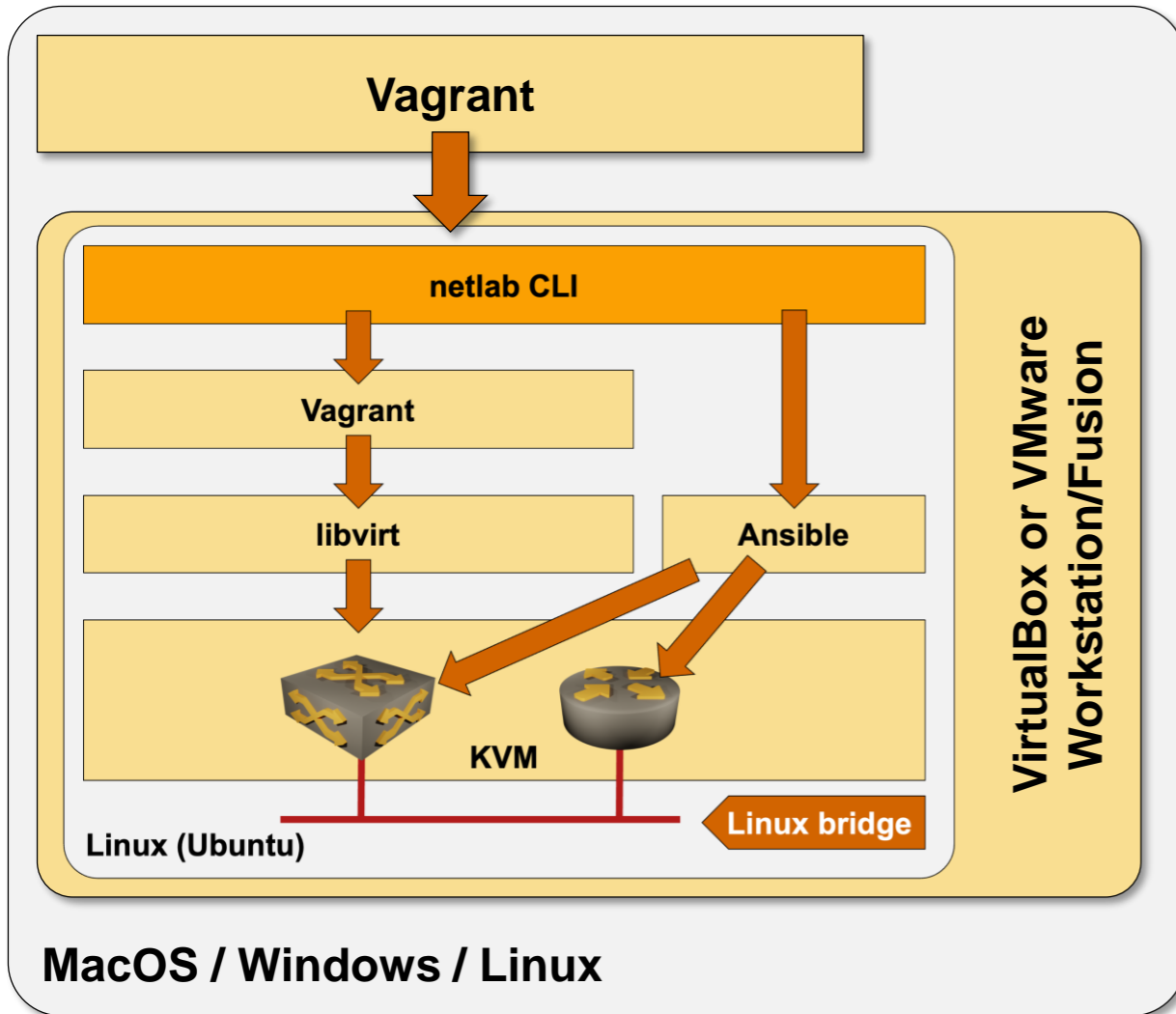
Devices as virtual machines

- KVM
- libvirt
- Vagrant with vagrant-libvirt plugin

Containers

- Docker
- Containerlab

Use Existing x86 Device: Ubuntu VM



Requirements

- You can run containers with any VM virtualization product
- Nested virtualization is required to run network device VMs

Virtualization solution with nested virtualization

- Hyper-V (WSL)
- KVM
- VirtualBox
- VMware Workstation/Fusion

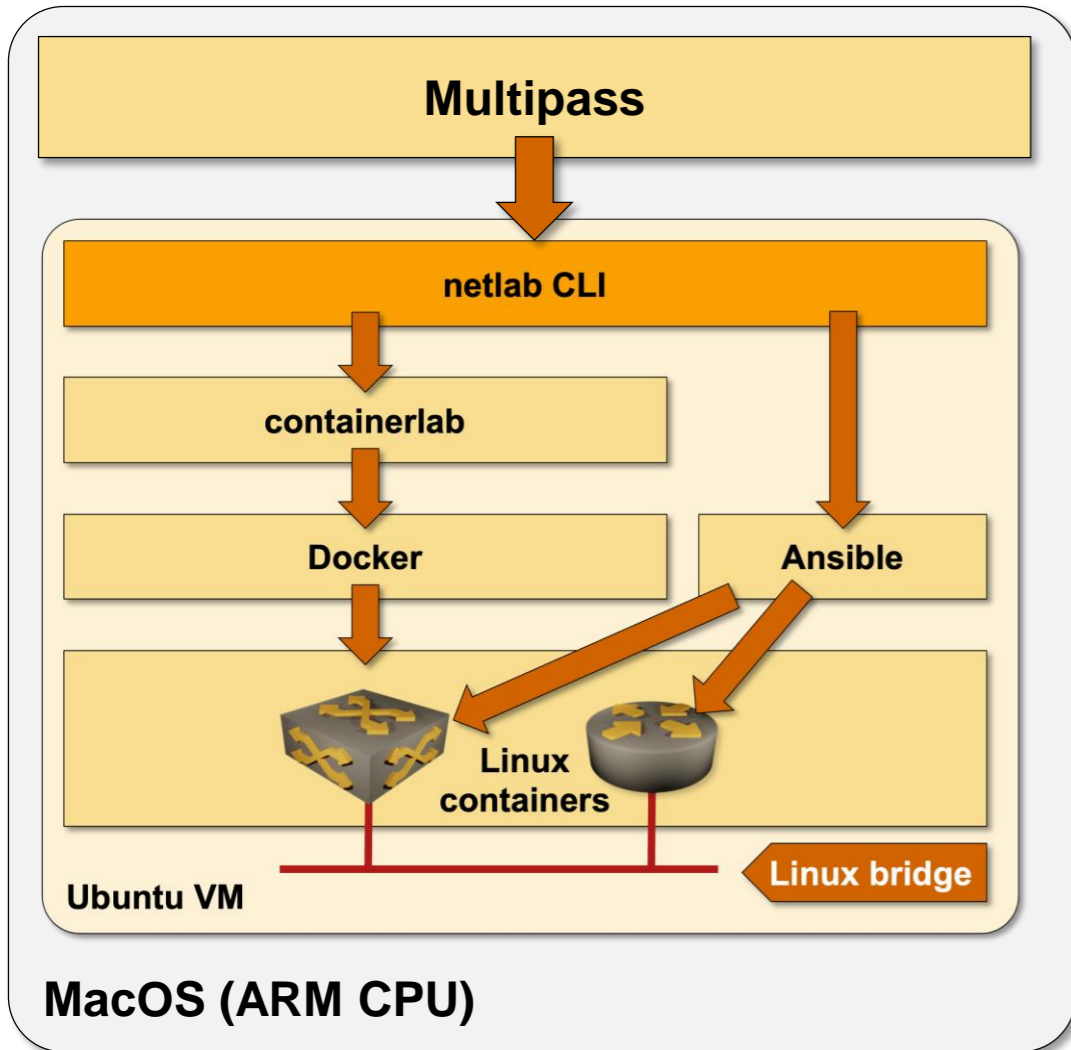
Optional

- Start the VM with Vagrant (simplifies the operations)

Alternative

- Cloud deployment

Ubuntu VM on Apple Silicon



- **multipass** starts an Ubuntu VM on an ARM CPU
- Nested virtualization is not supported → containers only
- Container images must be built for the ARM CPU → FRRouting only (at the moment)

Interesting use cases

- Run BGP, VXLAN, or EVPN labs on your Apple laptop
- FRRouting control-plane configuration syntax is pretty close to the *industry standard CLI*



Some Assembly Required (Thank You, Vendors)

Automatically downloadable images and containers

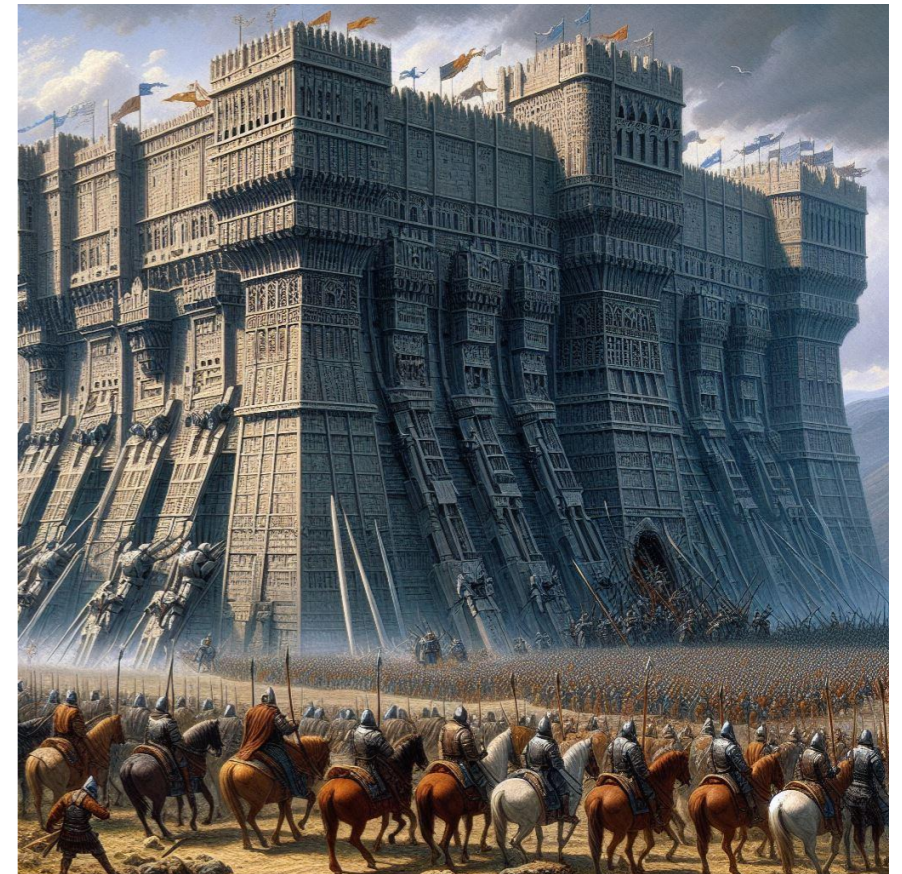
- FRR
- Cumulus Linux
- Nokia SR Linux
- VyOS

Easy to download (no registration required)

- Juniper vPTX, Dell OS10, Mikrotik RouterOS7

Most everything else (from bad to worse)

- Registration (Arista EOS, Aruba CX, Cisco Nexus OS)
- Download tied to a valid support contract (Cisco CSR)
- Begging your SE
- Available only if you know the right dev person



But Wait, That's Not All

After you download a container image (Arista cEOS)

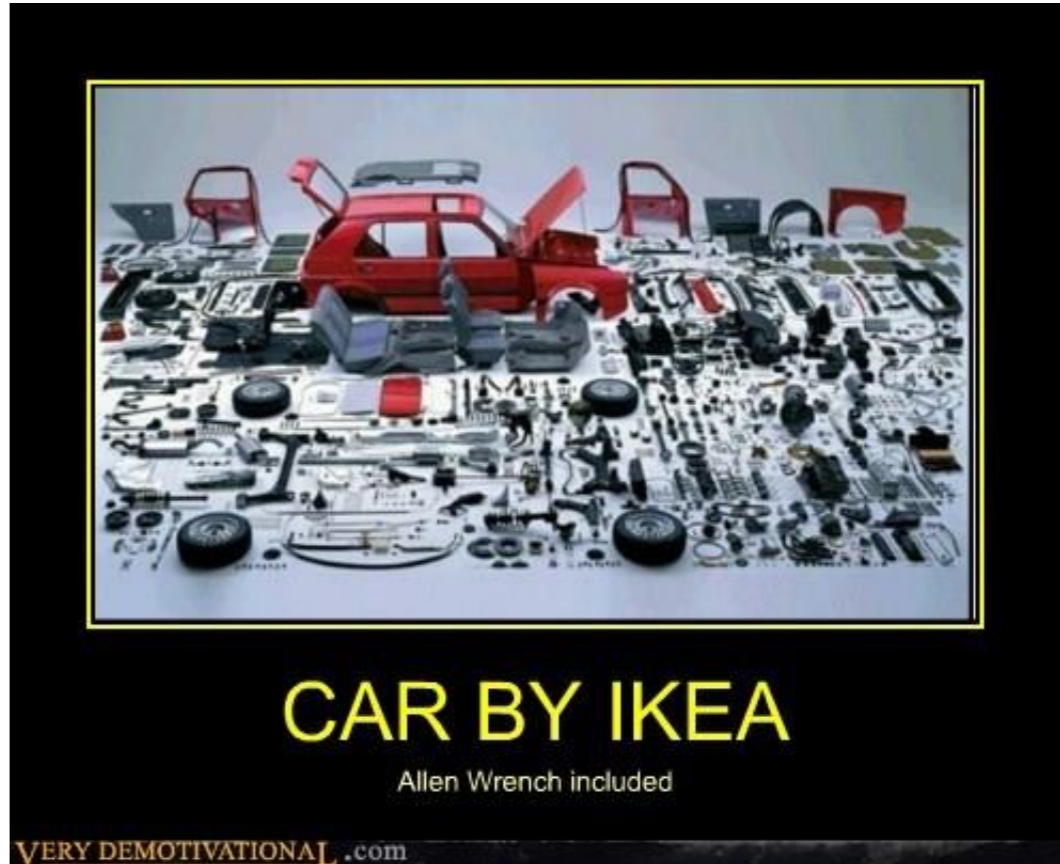
- Unpack and install it (easy)

Virtual machines are a nightmare

- Download a virtual disk
- It boots without a configuration and expects stuff on serial port
- Exception: Junos and ASA can take configuration from a mounted CD-ROM

Building a Vagrant box

- What we need to automate lab startup
- Start the VM, answer a dozen questions
- Copy-paste initial configuration
- Save the configuration, shut down the VM
- Package as a Vagrant box, hope it works



Getting Started

Getting Started: Hardware

Get decent hardware

- Network devices consume between 256MB (Mikrotik) and 12GB (Cisco Nexus 9300v release 10.3) per node
- Containers consume between 200MB (FRR) and 1GB (Arista cEOS, Nokia SR Linux)

Decide which deployment architecture you want to use

- Existing x86 hardware using Windows/Mac: Linux VM (might need nested virtualization support)
- ARM CPU: Linux VM running FRR containers
- Standalone Linux server on dedicated hardware: use a fresh copy of Ubuntu

Standalone server recommendations

- Intel NUC works great (but is no longer available)
- Run containers (Arista, Cumulus, FRRouting) on a VM in the cloud (example: Oracle Cloud Free Tier)
- Some cloud providers offer nested virtualization (Google Cloud, Packet, Digital Ocean)

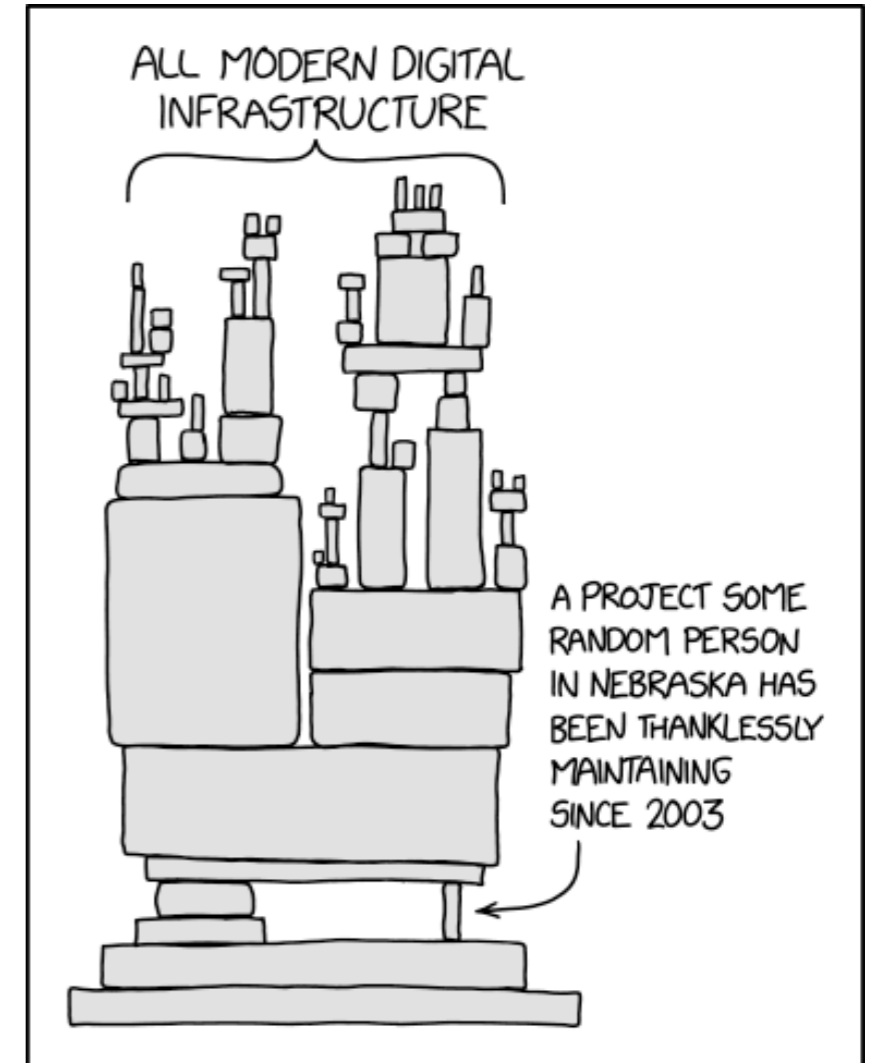
Getting Started: Installation

Ubuntu

- Start with a (fresh) Ubuntu VM or a bare-metal server
- Install Python3 if needed
- Install *networklab* package with **pip**
- Use **netlab** installation scripts to install libvirt/KVM, Vagrant, Ansible, Docker, containerlab...
- Download or build Vagrant boxes or containers (tons of recipes on netlab.tools/labs/libvirt and netlab.tools/labs/clab)

Other Linux distributions

- Everything *should* work (apart from installation scripts)
- Box-building tools have been tested on Ubuntu
- We won't be able to help you ☐ ♂ ☐



Is It Worth It? What Others Are Saying...



Lou D. • 2nd

4h (edited) ...

Network Security Engineer | Network Automation

I fully agree with Ivan's point on the show about being exhausted building labs . Netlab is really one the best tools to help me build fairly complex network designs quick and also cheap ! I don't need so much resource provisioning to get something started a tiny aws Ubuntu box will build an decent amount of nodes .

Like ·  2 | Reply · 1 Reply



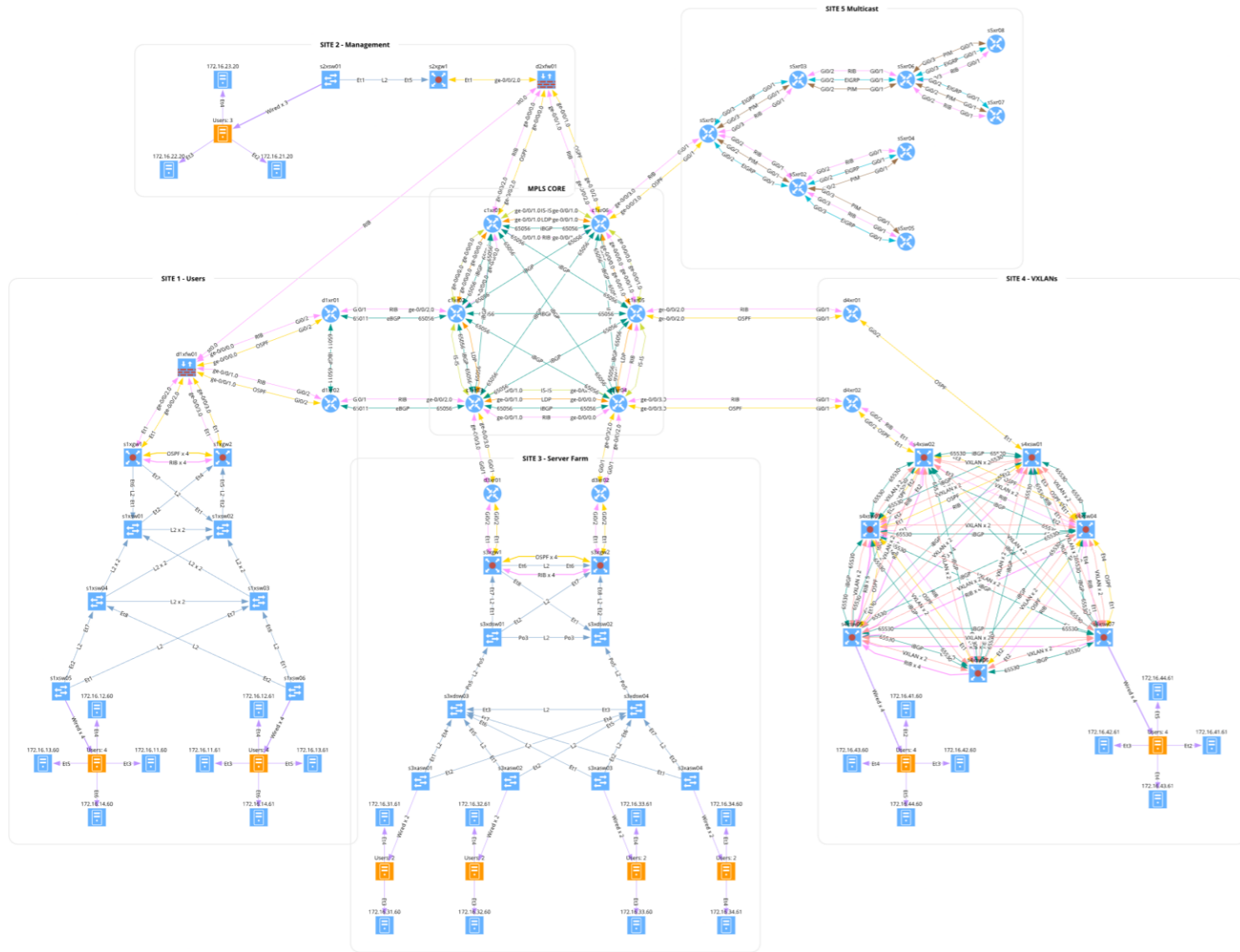
Ethan Banks • 1st

IT Podcast Host & Industry Analyst

1mo · 

If you work with networking labs but haven't worked with the netlab project, you need to. YAML-defined labbing, essentially. So much of the tedious work is just done for you. You don't have to drag icons around on a canvas and make links and setup IP addressing and configure routing and and and. Just tell netlab what you want it to build in the YAML file. Then "netlab up".

What Others Are Building with netlab



How Can You Help?

- Spread the word ;)
- Use the tool to build your labs
- Ask questions and report bugs

Want to contribute?

- Fix documentation
- Fix bugs
- Add new functionality to existing devices (example: VXLAN on IOS XR or vPTX)
- Add new devices

Still not enough?

- Develop new plugins (hint: OSPF interface parameters)
- Develop new modules (IP multicast, Babel, RIP...)



Questions?

Netlab resources

Documentation: netlab.tools

Blog posts: blog.ip-space.net/tag/netlab.html

Source code: github.com/ip-space/netlab

Examples: github.com/ip-space/netlab-examples

Sample project: bgplabs.net

To reach me

Web: [ipSpace.net](https://ip-space.net)

Email: [ip@ipSpace.net](mailto:ip@ip-space.net)

