RIPE NCC
RIPE NETWORK COORDINATION CENTRE

Run Your Own Networking Lab

With Vagrant and Ansible

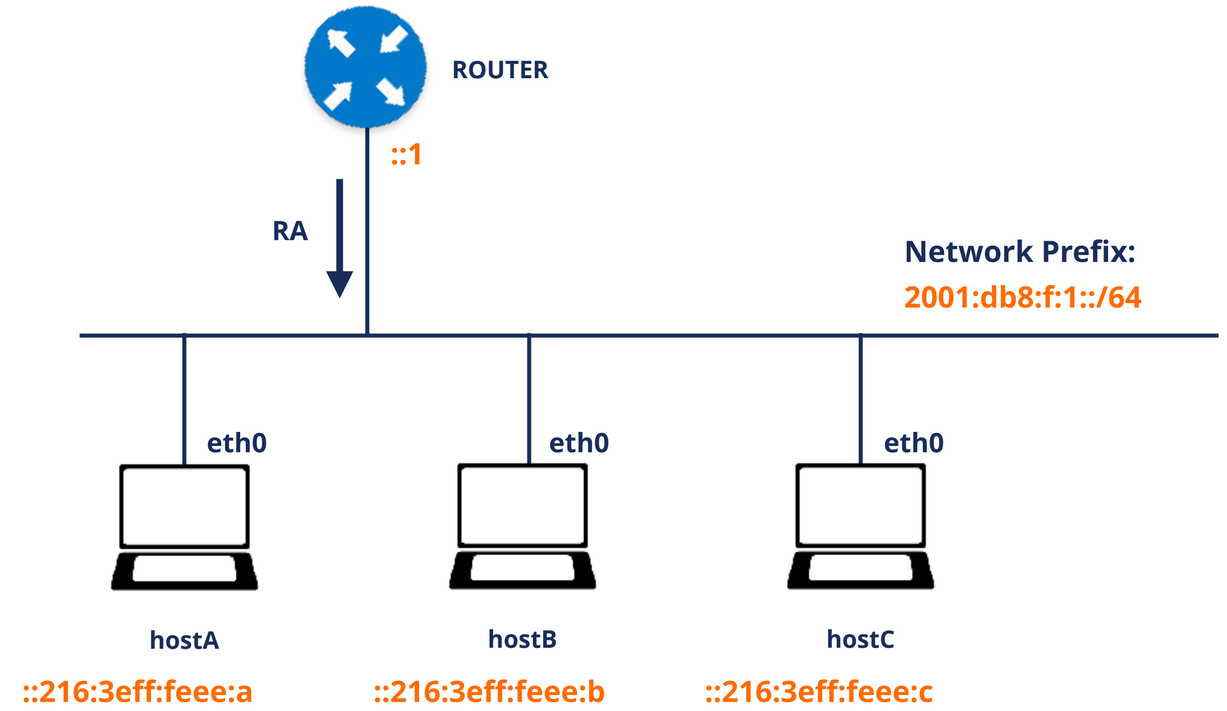
Ondřej Caletka | 29 November 2022 | NLUUG 2022

RIPE NCC Learning & Development

- Former Training Services of the RIPE NCC
- **Face-to-face trainings** for LIRs
  - Recently restarted after COVID-19
- **Webinars** for LIRs
  - Live interactive sessions lasting 1-2 hours
- RIPE NCC **Academy**
  - E-learning platform accessible to **everyone**
- RIPE NCC Certified Professionals
  - Prove your skills and receive a digital badge

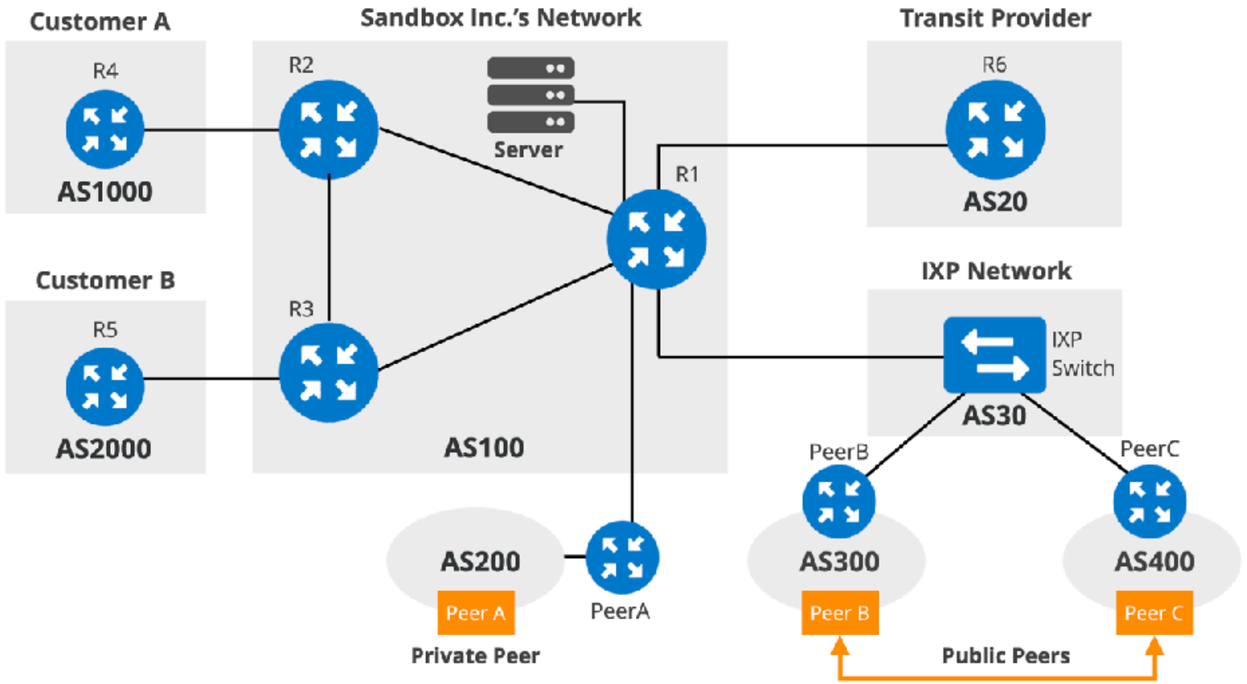
IPv6 Security E-learning Course

- A recent addition to IPv6 Security trainings and webinars
- Preparation for IPv6 Security Certified Professional exam
- First time with **hands-on labs**



BGP Operations and Security

- Brand new e-learning course on the RIPE NCC Academy
- A sort-of realistic ISP network with many routers



Delivering Lab Environment

- Should be **universally scalable**
- Should not cost us *too much* money
- Should allow **enough time** to play with it
- Should be **easy to use**
- We decided to deliver a **Virtual Machine image**

Image: Markus Meier, FSFE, CC-BY-SA 4.0

Virtual Machine Challenges

- Different **virtualisation technology** on each platform
  - The only *common* solution is **Oracle VM VirtualBox**, available on Windows, macOS or Linux
  - Still *suboptimal* compared to native solutions like Hyper-V or KVM
- No **common keyboard layout** or screen resolution
  - Therefore, we deliver the VM **headless** with everything accessible over a web interface
- Deploying a VM image is hard
  - We try to make it easier by using **Vagrant**

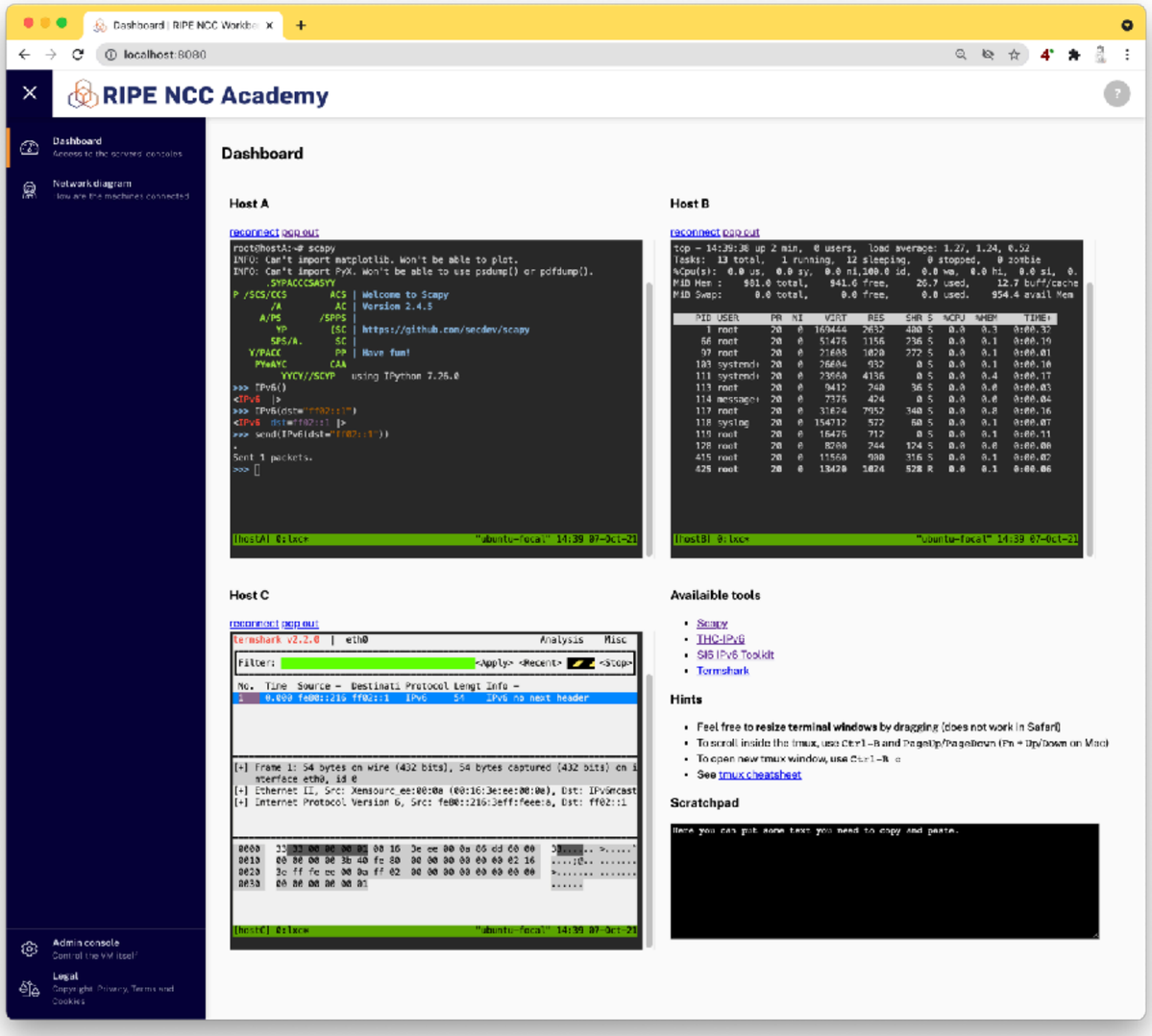
Running The Labs

- Install VirtualBox
- Install Vagrant
- Open terminal and type:

```
vagrant init ripencc/ipv6seclab
vagrant up
```

- Open web browser on

```
http://localhost:8080/
```



How It Works

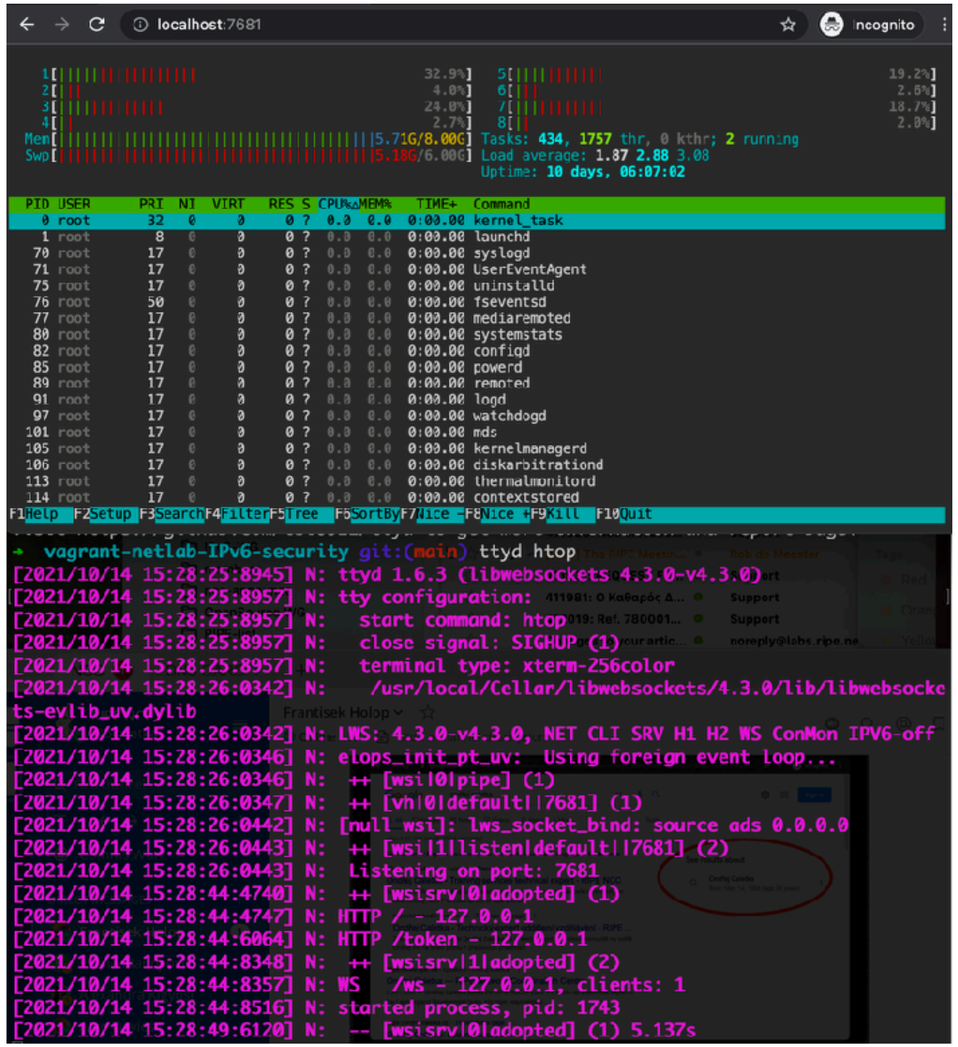
How It Works

- Based on Ubuntu 20.04 LTS VM
- Containers managed by **LXD**
  - Ubuntu for a Linux Host
  - Alpine Linux with FRR as a virtual router (very small footprint)
- Consoles accessible from web browser using **ttyd** and **tmux**
- Static website and WebSocket proxy by **NGINX**
- Everything deployed using **Ansible**
- **Public development** in RIPE NCC's GitHub repository

```
https://github.com/RIPE-NCC/ipv6-security-lab/
https://github.com/RIPE-NCC/bgp-security-lab/
```

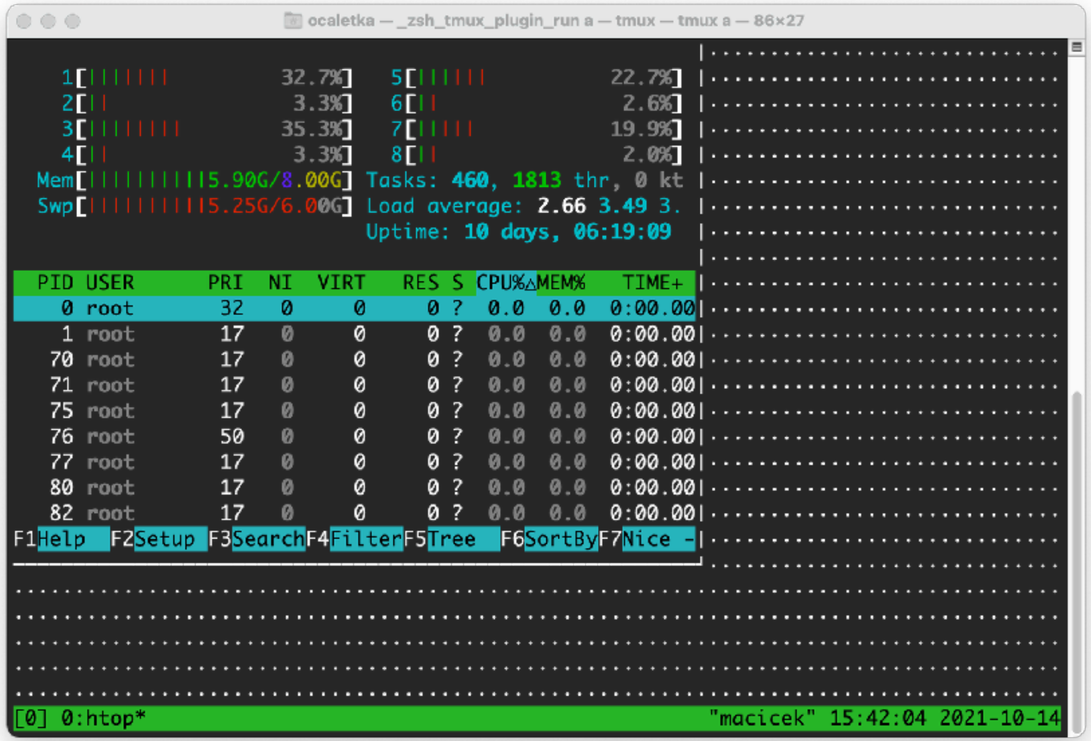
How to Get Terminal into a Web Browser

- At least three options:
  - Gotty (Go)
  - WeTTy (node.js)
  - ttyd (C)
- They all act as a webserver with WebSocket support
- **Xterm.js** runs in the browser
- For *each* WebSocket connection a command is run
  - the session is gone when you refresh/close



How to Not Lose the Session

- GNU **Screen** or **tmux**
- Allows **multiple connections** to single terminal session
- Actually more complex than you think
- Getting **scrollback** to work is hard
- Tmux newer than 2.3 ~~shortens~~ *optimises* long listings
- *Workaround*: rollback to tmux 2.3



Putting Everything Together

- NGINX works as a **reverse proxy** in front of ttyd
- Port 80 is **redirected** from the host computer
- User opens their browser pointing to the redirected port

Vagrant and Ansible

Automating the most of it

Vagrant Automates the Boring Parts

- A Virtual Machine image is downloaded and set up
  - Network is set up
  - SSH key authentication is configured
  - A host folder is shared with the VM
- Ansible is installed into VM
- Ansible is run to install everything into the VM
- **Everything is reproducible**
  - development is easy

Ansible Playbooks Can Break

- Third party sources tend to break randomly
- Reachability of various websites **varies** around the world
- It is not safe to assume the playbook will keep working
- Therefore we create **ready-made VM image**
  - Only one big file to download
  - Faster deployment
  - Everything frozen in time
  - This is **automated using a GitHub Action**

Vagrant Limitations

- Boxes (VM images) work only with particular hypervisor
  - Supporting multiple hypervisors means duplicate work
- Only very basic networking support *(no IPv6 in the VM)*
  - Manual modifications break Vagrant's automation
- No **VirtualBox support for Apple Silicon**
  - This significantly impacts usability of Vagrant on new Apple devices
  - Hopefully there will be some easy-to-use solution to automate running VMs

Demo Time

Everything will break now

Other Ready-made Solutions

- Containerlab
  - Run network devices in Docker containers
  - Automate creation/teardown of lab topologies
- ipSpace.net Virtual Networking Labs Tool
  - A tool to prepare Vagrant and Ansible scripts to setup a networking lab
  - Support for libvirt VMs, VirtualBox VMs and Containerlab

Probably not a good idea to run on a consumer-grade laptop though

Questions

Ondrej.Caletka@ripe.net
@ripencc

https://academy.ripe.net