



**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



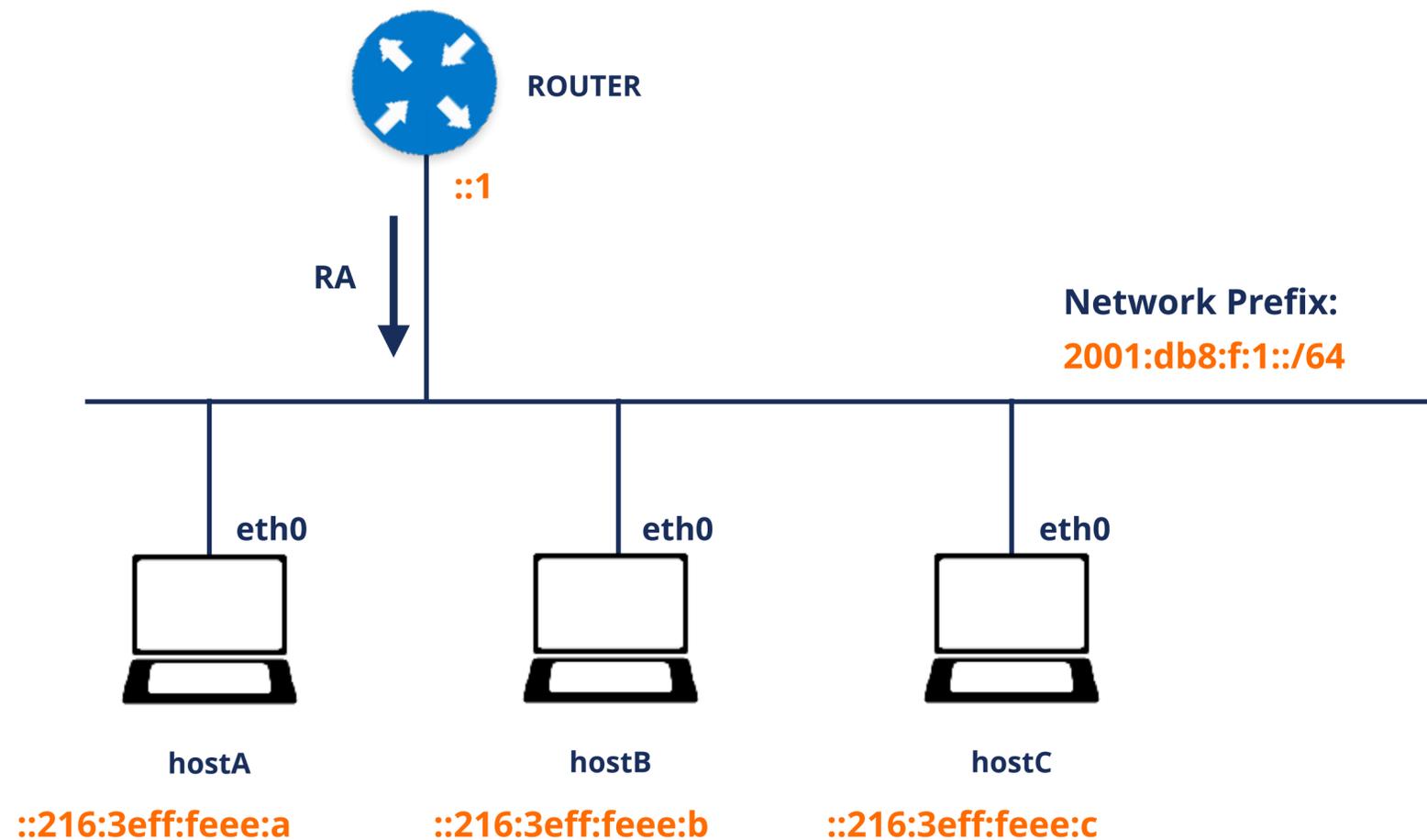
- Former Training Services of the RIPE NCC
- **Face-to-face trainings** for LIRs
  - Recently restarted after COVID-19
- **Webinars** for LIRs
  - Live interactive sessions lasting 1-2 hours
- **RIPE NCC Academy**
  - E-learning platform accessible to **everyone**
- **RIPE NCC Certified Professionals**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



# IPv6 Security E-learning Course



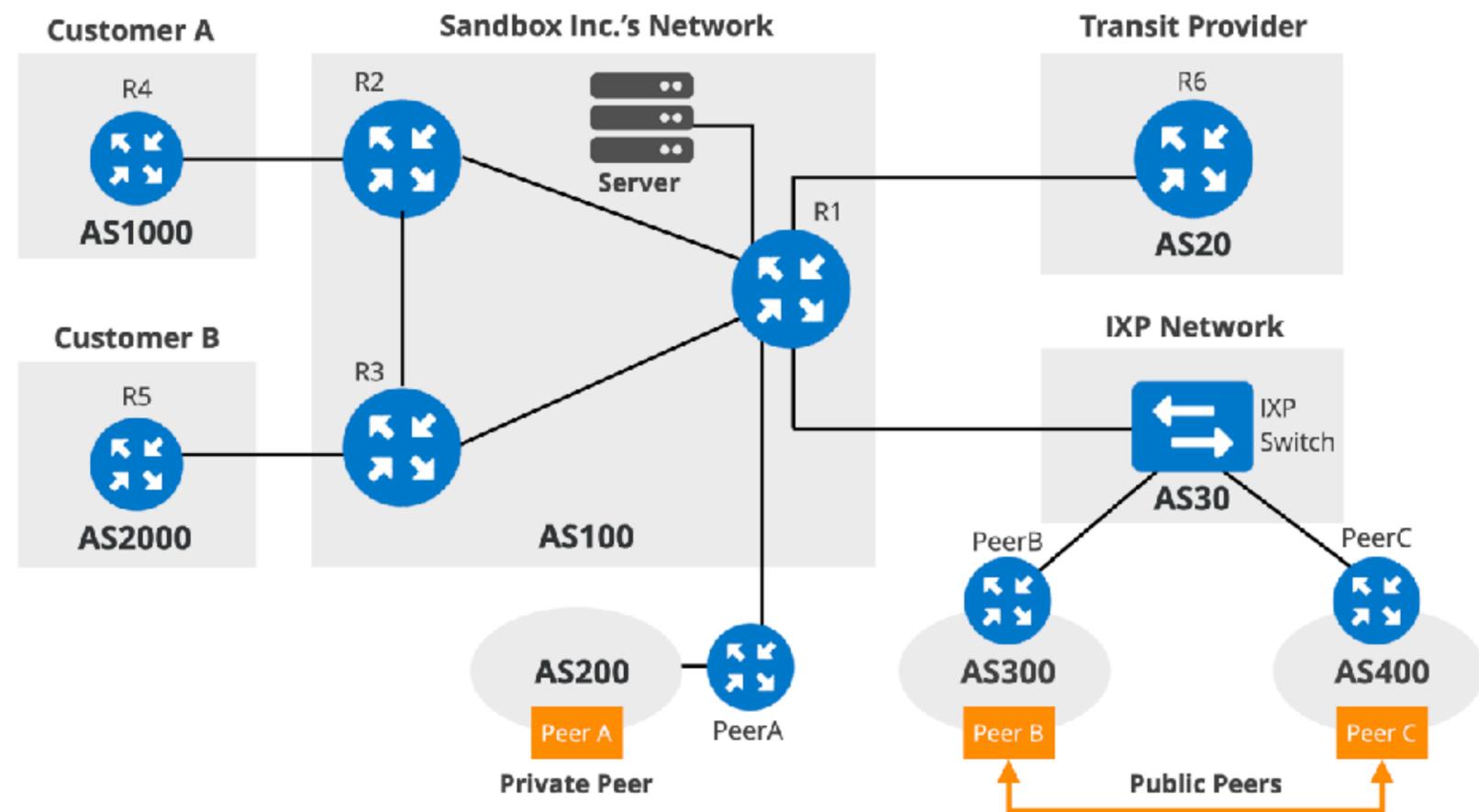
- A recent addition to IPv6 Security trainings and webinars
- Preparation for IPv6 Security Certified Professional exam
- First time with **hands-on labs**



# 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/>

The screenshot shows the RIPE NCC Academy dashboard at localhost:8080. The dashboard is divided into several sections:

- Host A:** A terminal window showing the installation of Scapy. The user runs `pip install scapy` and the output shows the installation of Scapy version 2.4.5. The user then runs `scapy` and enters `TPv6()`, `TPv6(dst='ff02::1')`, and `send(IPv6dst='ff02::1')`.
- Host B:** A terminal window showing the output of the `top` command, displaying system statistics and a list of running processes.
- Host C:** A terminal window showing the output of the `tcpdump` command, displaying a list of captured packets. The first packet is highlighted, showing its source and destination addresses and protocol.
- Available tools:** A list of tools available for the lab, including Scapy, THC-IPV6, SIB IPv6 Toolkit, and Tormshark.
- Hints:** A list of hints for running the lab, including instructions on how to resize terminal windows and scroll inside the tmux.
- Scratchpad:** A text area for taking notes.



# 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

```
localhost:7681
1 [|||||] 32.9% 5 [|||||] 19.2%
2 [|||||] 4.0% 6 [|||||] 2.6%
3 [|||||] 24.0% 7 [|||||] 18.7%
4 [|||||] 2.7% 8 [|||||] 2.0%
Mem [|||||] 5.71G/8.00G Tasks: 434, 1757 thr, 0 kthr; 2 running
Swp [|||||] 15.10G/6.00G Load average: 1.87 2.88 3.08
Uptime: 10 days, 06:07:02

PID USER PRI NI VIRT RES S CPU%MEM% TIME+ Command
0 root 32 0 0 0 0 ? 0.0 0.0 0:00.00 kernel_task
1 root 8 0 0 0 0 ? 0.0 0.0 0:00.00 launchd
70 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 syslogd
71 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 UserEventAgent
75 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 uninstalld
76 root 50 0 0 0 0 ? 0.0 0.0 0:00.00 fseventsd
77 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 mediaremoted
80 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 systemstats
82 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 configd
85 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 powerd
89 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 remoted
91 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 logd
97 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 watchdogd
101 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 mds
105 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 kernelmanagerd
106 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 diskarbitrationd
113 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 thermalmonitord
114 root 17 0 0 0 0 ? 0.0 0.0 0:00.00 contextstored

+ vagrant-netlab-IPv6-security git:(main) ttyd http
[2021/10/14 15:28:25:8945] N: ttyd 1.6.3 (libwebsockets 4.3.0-v4.3.0)
[2021/10/14 15:28:25:8957] N: tty configuration: 411981: 0 Καθαρός Δ... Support
[2021/10/14 15:28:25:8957] N: start command: http
[2021/10/14 15:28:25:8957] N: close signal: SIGHUP (1)
[2021/10/14 15:28:25:8957] N: terminal type: xterm-256color
[2021/10/14 15:28:26:0342] N: /usr/local/Cellar/libwebsockets/4.3.0/lib/libwebsocke
ts-evlib_uv.dylib
[2021/10/14 15:28:26:0342] N: LWS: 4.3.0-v4.3.0, NET CLI SRV H1 H2 WS ConMon IPV6-off
[2021/10/14 15:28:26:0346] N: elops_init_pt_uv: Using foreign event loop...
[2021/10/14 15:28:26:0346] N: ++ [wsil0|pipe] (1)
[2021/10/14 15:28:26:0347] N: ++ [vh0|default|7681] (1)
[2021/10/14 15:28:26:0442] N: [null wsi]: lws_socket_bind: source ads 0.0.0.0
[2021/10/14 15:28:26:0443] N: ++ [wsil1|listen|default|7681] (2)
[2021/10/14 15:28:26:0443] N: Listening on port: 7681
[2021/10/14 15:28:44:4740] N: ++ [wsisrv0|adopted] (1)
[2021/10/14 15:28:44:4747] N: HTTP / - 127.0.0.1
[2021/10/14 15:28:44:6064] N: HTTP /token - 127.0.0.1
[2021/10/14 15:28:44:8348] N: ++ [wsisrv1|adopted] (2)
[2021/10/14 15:28:44:8357] N: WS /ws - 127.0.0.1, clients: 1
[2021/10/14 15:28:44:8516] N: started process, pid: 1743
[2021/10/14 15:28:49:6120] N: -- [wsisrv0|adopted] (1) 5.137s
```

# 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

```
ocaletka - _zsh_tmux_plugin_run a - tmux - tmux a - 86x27
1[|||||] 32.7% 5[|||||] 22.7%
2[||] 3.3% 6[||] 2.6%
3[|||||] 35.3% 7[|||||] 19.9%
4[||] 3.3% 8[||] 2.0%
Mem[|||||] 5.90G/8.00G Tasks: 460, 1813 thr, 0 kt
Swp[|||||] 5.25G/6.00G Load average: 2.66 3.49 3.
Uptime: 10 days, 06:19:09

PID USER PRI NI VIRT RES S CPU%MEM% TIME+
0 root 32 0 0 0 ? 0.0 0.0 0:00.00
1 root 17 0 0 0 ? 0.0 0.0 0:00.00
70 root 17 0 0 0 ? 0.0 0.0 0:00.00
71 root 17 0 0 0 ? 0.0 0.0 0:00.00
75 root 17 0 0 0 ? 0.0 0.0 0:00.00
76 root 50 0 0 0 ? 0.0 0.0 0:00.00
77 root 17 0 0 0 ? 0.0 0.0 0:00.00
80 root 17 0 0 0 ? 0.0 0.0 0:00.00
82 root 17 0 0 0 ? 0.0 0.0 0:00.00
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -

[0] 0:htop* "macicek" 15:42:04 2021-10-14
```

# 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>

