

24-04-2020

White Box CPE

Grant Agreement No.:	856726
Work Package	WP6
Task Item:	Task 1
Dissemination Level:	PU (Public)
Document ID:	GN4-3-20-246f392
Authors:	Edin Salguero (RENATER), Bojan Jakovljevic (AMRES), Xavier Jeannin (RENATER), Ivana Golub (PSNC), Tim Chown (JISC), Alain Bidaud (CRIANN), Sébastien Vigneron (CRIANN)

© GÉANT Association on behalf of the GN4-3 project.

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 856726 (GN4-3).

Abstract

This document shows how a white box x86 server can be used as a CPE router, based on an implementation for a high school's network in Normandy, France. The document also provides a design based on user requirements, the results of the first implementation in production, and the concept of NFV for potential future implementations.

Table of Contents

Executive Summary	1
1 Introduction	2
2 Use Case Description	3
2.1 Performance Requirements	3
2.2 Reliability Requirements	4
2.3 Cost Requirements	4
3 Solutions and Methodology	5
3.1 Router CPE Validation	5
3.2 White Box Virtual CPE Router	6
3.3 Implementing Automation	7
4 Production Network Results	8
5 Conclusions	9
Appendix A FRR CPE router configuration	10
Appendix B Ansible implementation	12
References	16
Glossary	17

Table of Figures

Figure 3.1: Dell VEP series 4600	5
Figure 3.2: CPE Logical modular design	6
Figure 3.3: VNF stacking in virtualisation	7
Figure 4.1: White box router CPE bandwidth consumption	8

Executive Summary

This document describes a white box technology pilot run for a high-school network in Normandy, France. It shows how a white box x86 server can be used as a CPE router, including the requirements, design and results obtained from the devices installed in the production network. It also explores the concept of NFV for potential future implementations.

1 Introduction

A white box is a switch/router manufactured from commodity components that allows different Network Operating Systems (NOSs) to be run on the same piece of commodity hardware, decoupling the NOS software from the hardware.

White boxes, first deployed widely in data centres, offer an impressive forwarding capacity for a very low price, and are now potentially available for research and education (R&E) use cases. Although current NOS platforms do not provide all the features required by National Research and Education Networks (NRENs), the white box technology approach has the advantage of improving an NREN's level of independence from router vendors and could change the way NRENs manage their network deployments.

The white box chipset forwarding characteristics (forwarding capacity, internal memory, size of buffers) determine the scenarios in which it can be used in NREN networks (e.g. exchange point (IX) switch, data centre switch, CPE device, MPLS P/LSR router, etc.). For more details, see the review conducted by the *Network Technology Evaluation* task (WP6 T1) in the GN4-3 project on the applicability of new types of white box devices in to research and education community use cases [[D6.3](#)].

2 Use Case Description

The Normandy CPE use case is related to the refresh of the CPE router that interconnects high schools to the Normandy [RN] regional network, SYVIK [SYVIK]. Currently, there are around 120 high schools equipped with Cisco routers that are connected to the SYVIK network. One of the limitations of the existing network is that the current router platform does not offer a 1Gbps connection.

The SYVIK network is directly connected to the backbone network of RENATER (the French NREN). The main objective of the project is to improve the capacity and the flexibility of the networking devices that interconnects the high schools in the Normandy region. SYVIK would like to be more independent from its hardware provider and possibly bring additional services to the high schools. The actual equipment in use is not capable of sustaining the increasing network bandwidth demand, but also it is not the cheapest solution.

2.1 Performance Requirements

To fulfil the project requirements, the following set of features are recognised as necessary capabilities for deployment:

- RIB capacity of 4k/4k routes (IPv4/IPv6)
- FIB capacity of 1k/1k routes (IPv4/IPv6)
- Ethernet MTU up to 9022 bytes and IP MTU up to 9000 bytes
- Switching/Forwarding characteristics
 - Traffic forward capacity of 10Gb/s
 - The forwarding capacity will depend on the number of virtual network functions implemented (e.g. Router, Firewall, DPI etc.)
- Bandwidth connection
 - Being able to connect the high school at 1G up to 10G
- Support for orchestration and automation is required to streamline the deployment, the configuration and the daily management.
- NOS control plane - Reproduce the current feature at the same level of performance
 - BGP peering
 - IGP
 - VLAN
 - Logical interface
 - VRF light

2.2 Reliability Requirements

- Non-redundant control plane
- Hot-swappable power supplies desired
- Hot-swappable fan units desired
- The service maintenance on site (maximum outage time of 4 hours)

2.3 Cost Requirements

- The solution must not exceed the cost of the budget required for the previous network infrastructure (see the white box TCO calculator [[TCO](#)]).

3 Solutions and Methodology

In 2019, there was no white box as a CPE router solution, as the white box market was largely driven by data centre and cloud scenarios. Current solutions like bare-metal switches are equipped with high-forwarding capacity interfaces. These features increase the price drastically and they are not generally necessary for CPE purposes. To implement a solution based on a white box, a x86 server white box running a virtualised environment with a virtual router on top of the hypervisor is a suitable solution. To be specific, the virtual NOS can be implemented as a VM running on top of a Linux VM. This architecture also provides a way to support the massive deployment via an automation tool provided by Linux tools.

For these reasons, a specific x86 server white box designed to play the role of a small router was chosen from the Dell VEP (Virtual Edge Platform) family [[VEP](#)]. This type of device can run a virtualised environment, and provide 2*10 Gbps and several 1Gbps ports with a competitive price.

Server Dell VEP 4600:

- x86 architecture (No specific ASIC) (CPU 8 cores, 16GB RAM, 1TB SSD)
- Network connectivity 2x10Gbps + 4x1Gbps
- Extension cards slots (1 Gbps, 10 Gbps, Wifi)
- No graphic card, only console port
- Double power supply + hot swap fan
- Made to run without cooling system



Figure 3.1: Dell VEP series 4600

3.1 Router CPE Validation

To achieve the defined requirements, two different NOS (Cumulus Networks and Free Range Routing (FRR)) were tested and validated. Both of them run on a Ubuntu server Linux VM and have been proven to be suitable for a router CPE implementation. From the evaluation, the SYVIK administrators decided to use FRR [[FRR](#)]. FRR is an open source solution which provides the necessary set of features

for a router CPE device, and is free of charge, easy to understand and very familiar for those who have worked with most of the popular vendors' networking operating systems in the market.

The validation was based on three main categories:

- System management: User accounts, telnet, TACACS+, SNMP etc.
- Basic security: basic and advanced ACLs
- Routing: this category represents the core of router CPE features. During these tests, a BGP peering was established, sharing static and default routes, VRF-Lite configuration, OSPF deployment etc.

FRR supports only basic L3 ACLs. However, the control plane protection is done in the Linux OS with iptables, so advanced ACLs could be implemented at the Linux level. Even though Cumulus networks [CNT] performs better (100 % of tests completed in each category), the client chose FRR because the basic set of features specially for IP routing were all successfully completed, and in addition the platform is free of charge.

3.2 White Box Virtual CPE Router

Figure 3.2 shows the logical design for the white box server that was used for the implemented solution:

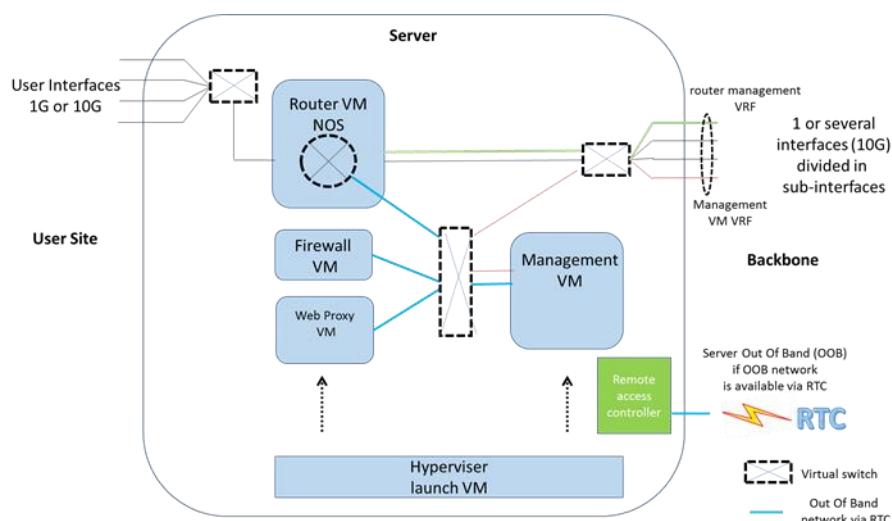


Figure 3.2: CPE Logical modular design

This diagram shows how multiple network services can be delivered using a single interface (usually fibreoptic) by differentiating them with VLANs or sub-interfaces. The CPE white box implementation uses this approach in the same way, and the services are delivered using the VEP x86 server interfaces (2*10G ports). Internally, since the server is running over a virtualised environment, different services are provisioned as Virtual Network Functions (VNFs). They all communicate via the internal virtual switch. The Router VM NOS part references the NOS running on the top of Linux and represents the router CPE. This design provides a modular architecture for a CPE router implementation.

A significant advantage of a white box running in a virtualised environment is that it can support VNFs. Depending on the hardware components (RAM, CPU, etc.), it can support one or several VNFs on the same hardware, delivering a single box solution for many purposes at customer premises (example: router + firewall + web proxy). By doing so, the customer can have a lower total cost of ownership with a single white box device. The VNFs have the advantage of reducing the space and power consumption, have a faster time to deploy and are more flexible. All of them represent cost savings for the network operators.

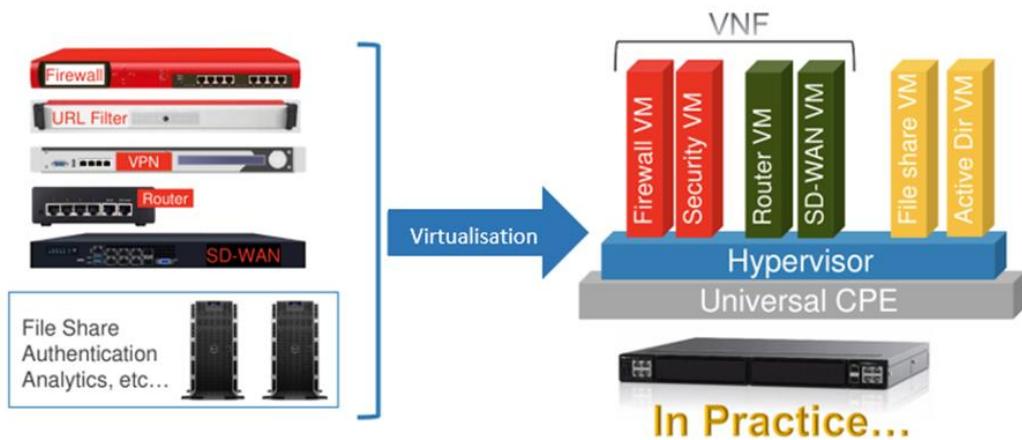


Figure 3.3: VNF stacking in virtualisation

An important consideration during the design phase of white box CPE solution is the choice of the hypervisor and the orchestrator as this determines the structure of the entire management solution. In this particular case, the white box CPE solution was tested over free Proxmox/KVM [[PROX](#)] and commercial VMWare ESXi [[VMW](#)] hypervisors. Both of them are fully compatible with white box x86 server, and Proxmox/KVM was chosen for economic reasons.

3.3 Implementing Automation

Another advantage of white box solutions is that they can run with a Linux OS instance base. For this reason, they are compatible with most of the available Linux utilities. To support a large-scale deployment in the production network, and to reduce the installation and support time required of the NOC, it was decided to provide a solution that includes automation. Ansible [[ANS](#)] is the tool that was chosen for its simplicity and its different modules for integration with other platforms. Another advantage is that in order to work, Ansible only requires an SSH connection to remote machines (the deployed white boxes) without installation of any external agent.

As expected, the Ansible automation tool was completely compatible with the white box and the NOS FRR. Using Ansible, the white box device can be properly set, starting with FRR software installation and patching, Telnet and SNMP installation, and then configured using a configuration YAML file (an easily understandable file containing the configuration values). These tasks are performed via SSH from the main provisioning server that is running Ansible software.

4 Production Network Results

Based on the results obtained in the tests, this solution was deployed in the production network of two high schools. The details of these implementations are as follows:

- Deployed in production in two high schools in 2019.
- The white box devices were configured and set by Ansible scripts.
- Reliability: no issues found after four months of use.
- Performance: 1Gb/s without any optimisation.
 - Max 5% CPU | 2 cores | 4G RAM used at virtual CPE level.
 - The following figure shows the bandwidth consumption for almost one day at one of the high schools:

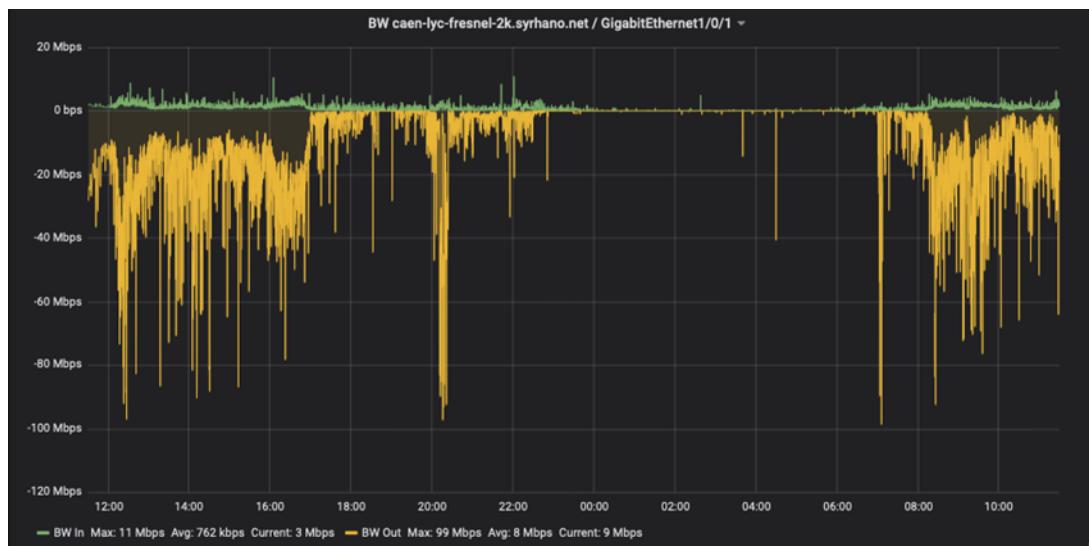


Figure 4.1: White box router CPE bandwidth consumption

Note: when the image was captured, the traffic was limited to 100 Mbps as requested by the client.

The next step for testing would be to add an NFV service to optimise the return on investment. The obvious choice would be a firewall VNF that might enhance network security at the remote site. A firewall could be integrated with the CPE router using the same white box, but logically separated by the virtualisation layer.

5 Conclusions

A white box x86 server has proven to be an effective solution for implementation of a CPE router, not only due to its performance and the IP features that it delivers, but also due to its flexibility which helps tailor the solution according to the specified requirements. Configuring a white box server to run a virtualisation environment allows integration of NFV solutions, and the extension of the network and security services available in a single box. In this way, the amount of delivered services can be increased without increasing the number of devices required at the customer premises. Finally, the integration with open-source automation tools facilitates a more efficient deployment and operation of white box CPEs in large scale networks.

Compared with some of the most popular devices from commercial manufacturers in the networking domain, a white box server usually is a cheaper solution, but at the same time they might not be as cheap as other CPE solutions in the market. However, since the equipment is able to run virtualised environments, the CPE solution can be readily improved by dynamically adding functions that can give a considerable benefit to the economics of the solution by improving the return on investment and provide a better overall total cost of ownership

Appendix A FRR CPE router configuration

FRR uses a configuration file stored in a specific path /etc/frr for the initial configuration of the device. The following extract shows a configuration example of a CPE running BGP with some basic sets of configuration:

```
frr version 7.0
frr defaults traditional
hostname wb1-server
no ip forwarding
no ipv6 forwarding
service integrated-vtysh-config
!
ip route 203.0.113.0/24 10.11.11.2 ens19
!
interface ens19
    description SUBINTERFACE
    ip address 10.7.7.1/24
    ipv6 address 2001:DB8:0:baba::1/64
!
interface ens19.10
    description VLAN10
    ip address 10.10.10.1/24
!
interface ens19.11
    description VLAN 11
    ip address 10.11.11.1/24
!
interface lo
    description LOOPBACK SIMULATING A LAN
    ip address 192.0.1.1/32
    ipv6 address 2001:DB8:0:cafe::cafe/128
!
router bgp 6002
    neighbor 10.10.10.2 remote-as 6001
    !
    address-family ipv4 unicast
        redistribute connected
        neighbor 10.10.10.2 soft-reconfiguration inbound
    exit-address-family
!
```

```
line vty
!
```

Appendix B Ansible implementation

In Ansible implementations, the playbook YAML file represents the desired set of instructions that is sent to the machine (CPE router) for provisioning. This playbook can be applied to a single or a list of high schools (list of white boxes) which is described in the *white_box_provisioning* file. Moreover, to improve flexibility, it is possible to use variables in the playbook stored in different files (vars_FRR.basic.yml, etc.). The following extract shows the script of the playbook that was applied for the white box configuration:

```
---
- name: FRR Basic Configuration Test
  hosts: white_box_provisioning
  vars_files:
    - vars_FRR.basic.yml

  tasks:
    - name: Get current configuration before changes
      fetch:
        src: /etc/frr/frr.conf
        dest: /etc/ansible/conf_backups/{{ansible_hostname}}_{{ansible_date_time.date}}.conf
        mode: 0774
        flat: yes
      become: true

    - name: Configuring WAN int
      shell: vtysh -c 'configure terminal' -c 'interface {{ifWAN.name}}' -c 'ip address {{ifWAN.ip}}' -c 'description {{ifWAN.description}}'
      become: true
      when: ifWAN.ip != none and ifWAN.description != none and ifWAN.name != none

    - name: Configuring LAN int
      shell: vtysh -c 'configure terminal' -c 'interface {{ifLAN.name}}' -c 'ip address {{ifLAN.ip}}' -c 'description {{ifLAN.description}}'
      become: true
      when: ifLAN.ip != none and ifLAN.description != none and ifLAN.name != none
```

```
- name: Configuring WAN ipv6
  shell: vtysh -c 'configure terminal' -c 'interface
{{ifWAN.name}}' -c 'ip address {{ifWAN.ipv6}}'
  become: true
  when: ifWAN.ipv6 != none

- name: Configuring LAN ipv6
  shell: vtysh -c 'configure terminal' -c 'interface
{{ifLAN.name}}' -c 'ip address {{ifLAN.ipv6}}'
  become: true
  when: ifLAN.ipv6 != none

#For the Following task, should change the 203.0.113.0/24
... for the default route 0.0.0.0
- name: Configuring Default Route
  shell: vtysh -c 'configure terminal' -c 'ip route
203.0.113.0/24 {{defaultRoute.gateway}} {{ifWAN.name}}'
  become: true
  when: defaultRoute.gateway != none

- name: Configuring telnet password
  shell: vtysh -c 'configure terminal' -c 'password
{{CLIPass.telnetpass}}'
  become: true
  when: CLIPass.telnetpass != none

- name: Configuring super user password
  shell: vtysh -c 'configure terminal' -c 'enable
password {{CLIPass.enablepass}}'
  become: true
  when: CLIPass.enablepass != none

- name: Installing Telnet and Traceroute
  apt:
    name: "{{ packages }}"
    update_cache: yes
    state: present
  vars:
    packages:
      - telnetd
      - traceroute
  become: true

- name: Saving all changes
  shell: vtysh -c 'write'
  become: true
```

The YAML file defined in the hosts field of the playbook defines the list of white box to which the playbook is applied. The following extract shows the script information in this file (white_box_provisioning):

```
[white_box_Caen]
192.168.XX.XX ansible_user=root ansible_password="my
password"

[White_box_Rouen]
192.168.YY.YY ansible_user=root ansible_password="my
password"
```

The *vars_FRR.basic.yml* vars file contains all the variables that are unique for each white box at each High School. Variables such as IP address, hostname, etc. The following extract shows the script information in this file:

```
---
# Basic configuration variables
# WAN INTERFACE CONFIGURATION
ifWAN:
  name: ens19
  description: "CPE-B_to_PEs"
  ip:
    ipv6:

# LAN INTERFACE CONFIGURATION

ifLAN:
  name: ens20
  description: "HIGH SCHOOL CAEN CPE-B LAN"
  ip: 192.168.12.1/24
  ipv6:

CLIpss:
  telnetpass: "my password"
  enablepass:

# Advanced configuration variables

# SUB-INTERFACES AND VLANS CONFIGURATION

subIF:
  l2if: ens19
  vlan1: 10
  v1ip: 192.168.251.3/24
  v1desc: "CPE-B_to_PE-A Service"
  vlan2: 11
  v2ip: 192.168.252.2/24
```

```
v2desc: "CPE-B_to_PE-B Management"

routerBGP:
  asn: 6002
  neighbor1: 192.168.251.1
  neighbor2:
  remoteas1: 6001
  remoteas2:
  network:
    redis: connected
  ipv4: "yes"
  ipv6: ""
```

References

[ANS]	Ansible automation tool https://www.ansible.com/products/automation-platform
[CNT]	Cumulus Networks operating system https://cumulusnetworks.com/
[D6.3]	Deliverable D6.3 White Box Evaluation https://www.geant.org/Projects/GEANT_Project_GN4-3/GN43_deliverables/D6-3_White-Box-Evaluation.pdf
[FRR]	Free Range Routing operating system https://frrouting.org/
[RN]	Région Normandie https://www.normandie.fr/
[SYVIK]	SYVIK Regional Network https://www.criann.fr/reseau-syvik/
[PROX]	Proxmox virtual environment https://www.proxmox.com/en/
[TCO]	White box TCO calculator https://www.geant.org/Resources/Documents/GN4-3_White-Paper_White-Box-TCO.pdf
[VEP]	Dell VEP 4600 server https://www.dell.com/en-us/work/shop/povw/virtual-edge-platform-4600
[VMW]	VMWare ESXi hypervisor https://www.vmware.com/cloud-solutions.html

Glossary

ACL	Access Control List
CPE	Customer Premises Equipment
FRR	Free Range Routing
NFV	Network Virtualization Function
NOC	Network Operations Center
NOS	Network Operating System
TCO	Total Cost of Ownership
VNF	Virtual Network Function
VM	Virtual Machine
VPE	Virtual Edge Platform