white\textit{nfv}, in our third formal major release.

Whitenfv has a release calendar compatible with OSM
Evolving towards containers

Baremetal

Virtual Machines

Containers

Classical Network Appliance Approach

- Message Router
- CDN
- Session Border Controller
- WAN Acceleration
- DPI
- Firewall
- Carrier Grade NAT
- Tester/QoE monitor
- SGSN/GGSN
- PE Router
- BRAS
- Radio Access Network Nodes

- Fragmented non-commodity hardware.
- Physical install per appliance per site.
- Hardware development large barrier to entry for new vendors, constraining innovation & competition.

© 2019 Whitestack, LLC - ALL RIGHTS RESERVED. Reproduction, republication or redistribution is prohibited.
Containers for **OSM** and for **CNFs**

Kubernetes is used to support OSM, but also for supporting CNFs

For OSM

- Open Source MANO
- Ansible
- Kubernetes
- Cluster whitenfv

For CNFs

- Helm
- Juju
- Kubernetes
- Cluster kubernetes
Deploying whiteNFV by using Ansible on top of Kubernetes

WhiteNFV comes in Docker Containers, we deploy them in Kubernetes, on top of Openstack virtual machines or by using a cloud provider.
Deploying white nfv by using Ansible on top of Kubernetes

The fastest growing platform for Containers deployment.

**Kubernetes**

The most popular configuration tool, in the market (OpenSource)
How to deploy `whitenfv`?

Deployment benefits:

- Sub-second High-Availability and Auto-healing (VCA in progress)
- Easily updatable (seconds) through deployer machine (Upgrading a container, is a matter of seconds) "Rolling upgrades"
- Layered Deployment
  Components spread across several layers, to minimize security risks (avoid an attacker to get access to all components)
Deploying white\textsuperscript{nfv}

For testing

Express-all-in-one model (for testing and experimenting)

- “Express All-in-one” deployment, with co-located deployer docker container.
- Single VM, single interface
- Minimum: 2 vCPU, 8GB RAM
- Keeps WhiteMist Kubernetes for elasticity
**VNFs vs CNFs**

*Implementation differences*

- **VNF**
  - VDU = [Diagram of VDU]

- **CNF**
  - CNF = [Diagram of CNF with Kubernetes]
  - POD ~ [Diagram of POD]

© 2019 Whitestack, LLC - ALL RIGHTS RESERVED. Reproduction, republication or redistribution is prohibited.
Deploying **CNFs**
With Juju or Helm on Kubernetes
Helm uses Charts
A templating system to describe all the dependencies

dependencies

charts.yml

```
wordpress/
Chart.yaml  # A YAML file containing information about the chart
LICENSE     # OPTIONAL: A plain text file containing the license for the chart
README.md   # OPTIONAL: A human-readable README file
values.yaml # The default configuration values for this chart
values.schema.json # OPTIONAL: A JSON Schema for imposing a structure on the values.yaml file
charts/     # A directory containing any charts upon which this chart depends.
crds/       # Custom Resource Definitions
templates/   # A directory of templates that, when combined with values, # will generate valid Kubernetes manifest files.
templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

$ helm install {application}
Helm Hub
Thousands of Charts available (and many other Repositories)
Release Calendar

Our Release calendar relies on the community releases

- We committed to release for testing (rc) on this calendar
  - **Alcobendas** (Q1/2018) - Based on Release FIVE
  - **Barcelona** (Q3/2019) - Based on Release SIX
  - **Castelldefels** (Q1/2020) - Based on Release SEVEN
  - **D Release** (Q3/2020)
  - **E Release** (Q1/2021)

![Open Source MANO](image)

- `{release}-{n}-rc1` → `{release}-{n}-rc2` → `{release}-{n}`
  - **Release candidates (testing)**
  - **General Availability (production)**
Roadmap

- More functionality around CNFs
- Deployment of OSM by using Help Charts!
  
  `helm install whitenfv`

- More integration with Openstack
  - Load Balancing with Octavia
  - Storage with Cinder, Swift or Manila

- More monitoring
  - Improved Network Services Dashboards
  - System monitoring

- More Security
  - Session Encryption
  - Auditing
Successful Implementation of an **OpenRAN** deployment in Latam

**POPULATION DENSITY DISTRIBUTION**
- Data Mining: Satellite imagery
- Estimation Model: Neural networks to identify and count households
- Clustering (DBSCAN) to group households into settlements
- Training Sample: Census data to train and iterate the model

**INTERNET COVERAGE BY TECHNOLOGY & OPERATOR**
- Data Mining: Geolocate mobile sessions
- Estimation Logic: Generate internet coverage polygons by technology as perceived by mobile internet users
- Training Sample: Telefonica coverage and infrastructure information + reported regulatory data

**INTERNET UNSERVED POPULATION DISTRIBUTION**
- Geolocate population distribution by internet coverage status, by technology (LTE, 3G, Wifi, 2G/Edge)

---

Parallel Wireless
Telefonica
Facebook

© 2019 Whitestack, LLC - ALL RIGHTS RESERVED. Reproduction, republication or redistribution is prohibited.
Orchestrating a **Hybrid Network** Service

**Facebook Magma**

**Containers (Helm Chart)**

**Operator Core**

**IP Edge**

**Baremetal => VNF**

© 2019 Whitestack, LLC - ALL RIGHTS RESERVED.
Reproduction, republication or redistribution is prohibited.
Getting Ready for 5G

Onboarding Open5GS

An containerized open-source implementation of a 5G Core
VNF Catalogs

Whitestack, the VNFOB Task Force, and TSC members are promoting the concept of a repositories of VNFs, that will facilitate the distribution or Virtual Network Functions.

VNF Consumers

Open Source MANO

VNF Repository Specification

VNF Repository

VNF Consumers

VNF Publishers

© 2019 Whitestack, LLC - ALL RIGHTS RESERVED. Reproduction, republication or redistribution is prohibited.
Thanks!