Preliminary Design Review

Containerized, Intelligent Network Functions on Hosts

Project Instructor: Dr. Kevin Gifford
Project Advisor: Dr. Levi Perigo

Team 06:

Afure Martha Oyibo
Kiran Yeshwanth
Manesh Yadav
Prarthana Shedge
Soumya Velamala

Agenda

Topic	Presenter	Slide No.
Introduction	Prarthana	3 - 5
Why using SDN-NFV in Containers is beneficial?	Soumya	6
Problem Statement	Soumya	7
Project Elements	Soumya	8
FBD and CONOPS	Manesh	9 - 10
Evaluation Parameters	Manesh	11
Baseline Design from CDD	Manesh & Kiran	12 - 24
Baseline Design Summary	Kiran	25 - 27
Technical, Resource & Cost Feasibility	Kiran	28 - 37
Identifying Risk Factors & Strategy for Mitigation	Martha	38 - 39
Project Targets & Key Deadlines	Martha	40

Introduction

SDN Overview

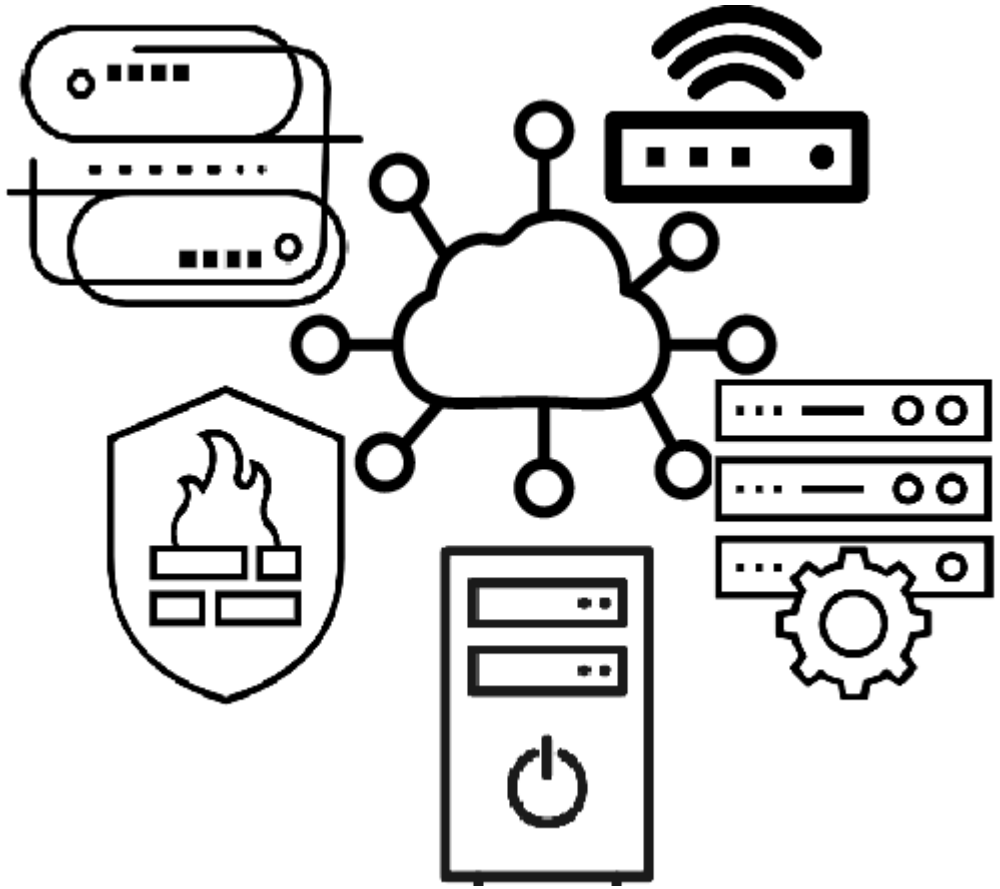
- Software Defined Networking
- Automation, orchestration and abstraction
- Efficient network management
- Flexible solutions [1]
- Scalability
- Faster service delivery
- Easier implementation of DevOps practices
- Redundant and robust architecture
- Global view
- Policy-based network supervision [2]
- Enhanced Quality of Service (QoS)

[1] https://en.wikipedia.org/wiki/Software-defined_networking
[2] https://www.fidelus.com/software-defined-networking-advantages/

Introduction (continued)

NFV Overview

- Network Function Virtualization
- Virtualizing network services [3]
- Virtual Network Function (VNFs) form individual network components
- Routers, firewalls packaged as software
- Hosted on Containers/Virtual Machines
- Bare metal hardware (x86 server) implementation
- CAPEX and OPEX savings
- Enhanced network resource utilization
- Easier implementation of DevOps practices
- Agility
- Facilitates the logical centralization

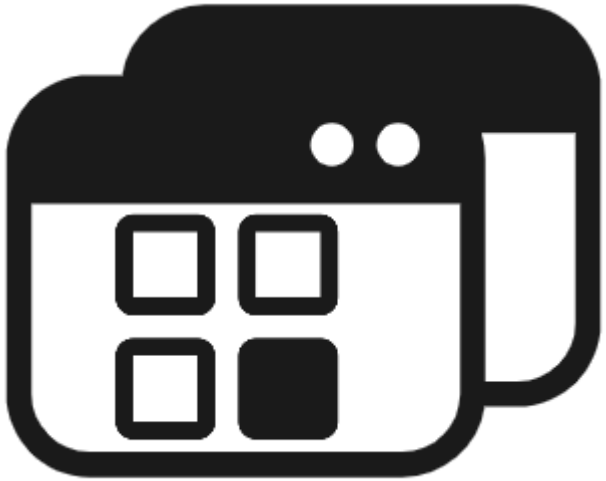


[3] https://searchnetworking.techtarget.com/definition/network-functions-virtualization-NFV

Introduction (continued)

Containers Overview

- Standalone, executable software packages [4]
- Contains code, libraries, user information
- Lightweight
- Seamless functionality
- CAPEX and OPEX savings
- Quick deployment
- Reduced management overhead
- Streamlines network functionalities and operations
- Consistent functionality from development to operational environments



[4] https://www.docker.com/resources/what-container

Why using SDN-NFV in Containers is beneficial?

Advantages of SDN-NFV

- Quick deployment
- Reduced kernel overhead
- Enhanced system performance
- Granular QoS based deployment
- Cost saving
- Reduces network complexities
- Proactive detection of network failures
- Resiliency
- Redundancy
- Reduces risk of packet losses
- Scalability
- Enhanced flexibility

Problem Statement

Objective

- Create a low-cost solution for the deployment of SDN based network services such as routers, firewalls, VoIP, etc. on the host devices through NFV:
  - NFV based services will be deployed through container based VNFs
  - Enhances performance by bringing SDN based intelligence closer to the end hosts
  - Rapid deployment of network services

Stretch Goal

- Detection of network failovers and implementation of redundant solutions:
  - Reduces the risk of packet losses
  - Seamless end user experience

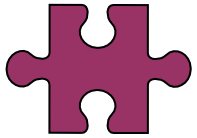
Project Elements

Network Service Deployment using Containers

Deploying multiple VNFs corresponding to OVS, VoIP and other network services using containers with push of a button

Deployment of SDN Infrastructure

Deploy and configure SDN controller to install flows in the VNFs and manage the entire network

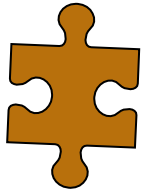


Creating Service Chain

Create multiple service chains with VNFs to simulate and test different scenarios

Creating Test Environment

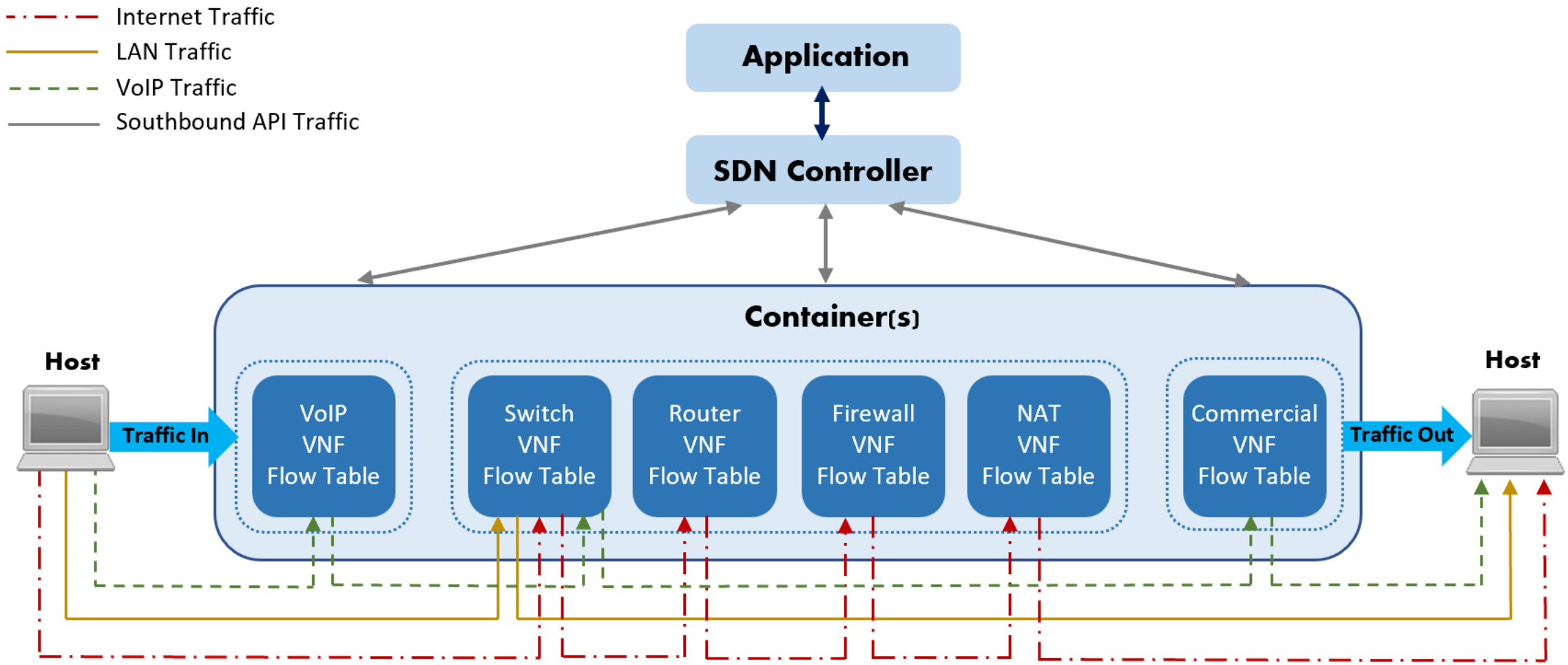
Emulate a test environment in test VM to perform operational, performance, and functional tests



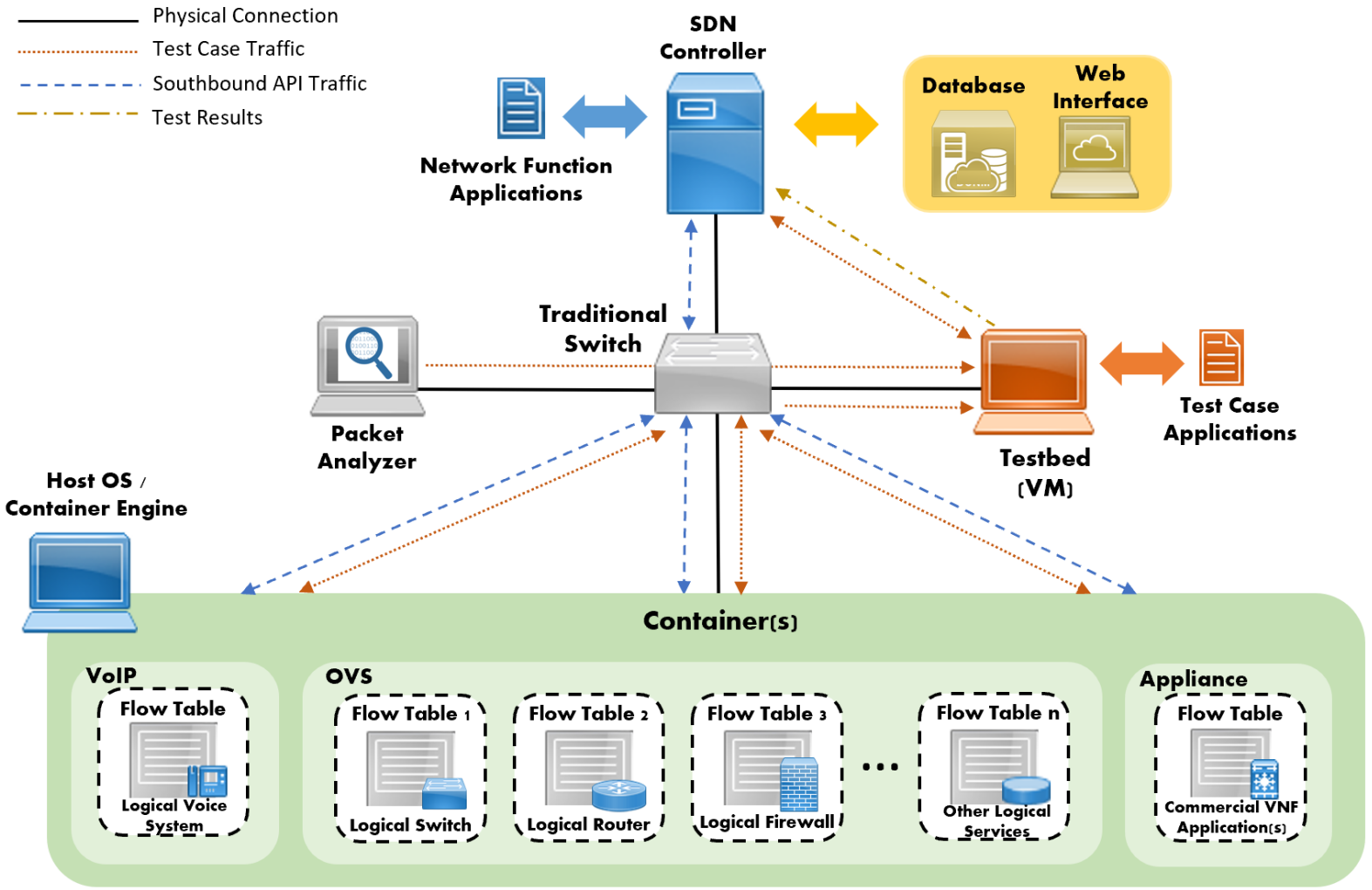
Storage and Display of Test Results

Create a database to store the test results, perform data analysis using data visualization tool and display the results on a web interface

Functional Block Diagram (FBD)

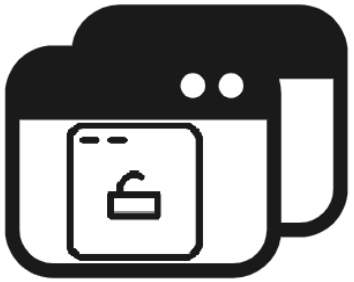


Concept of Operations (CONOPS)



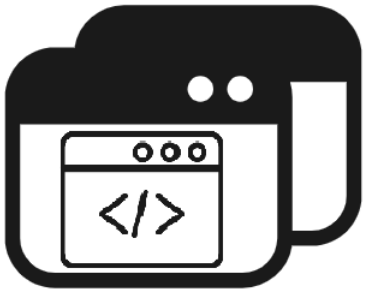
Evaluation Parameters

Cost

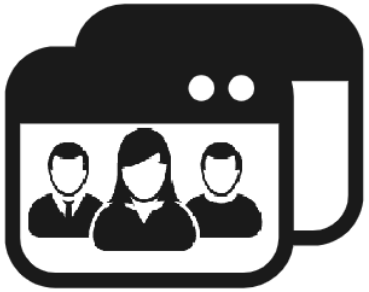


Open-source

Features

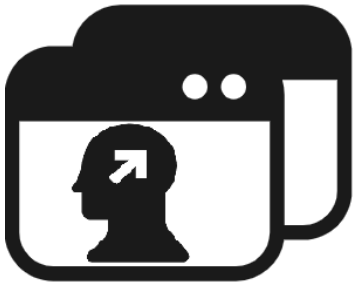


Programmability

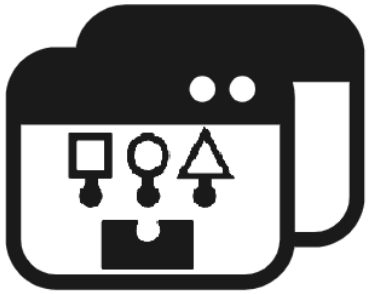


Community Support

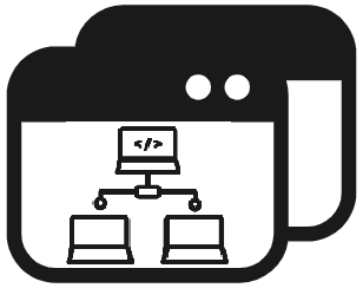
Documentation



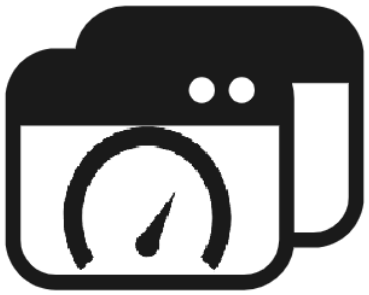
Learning Curve



Modularity



Ease of Deployment

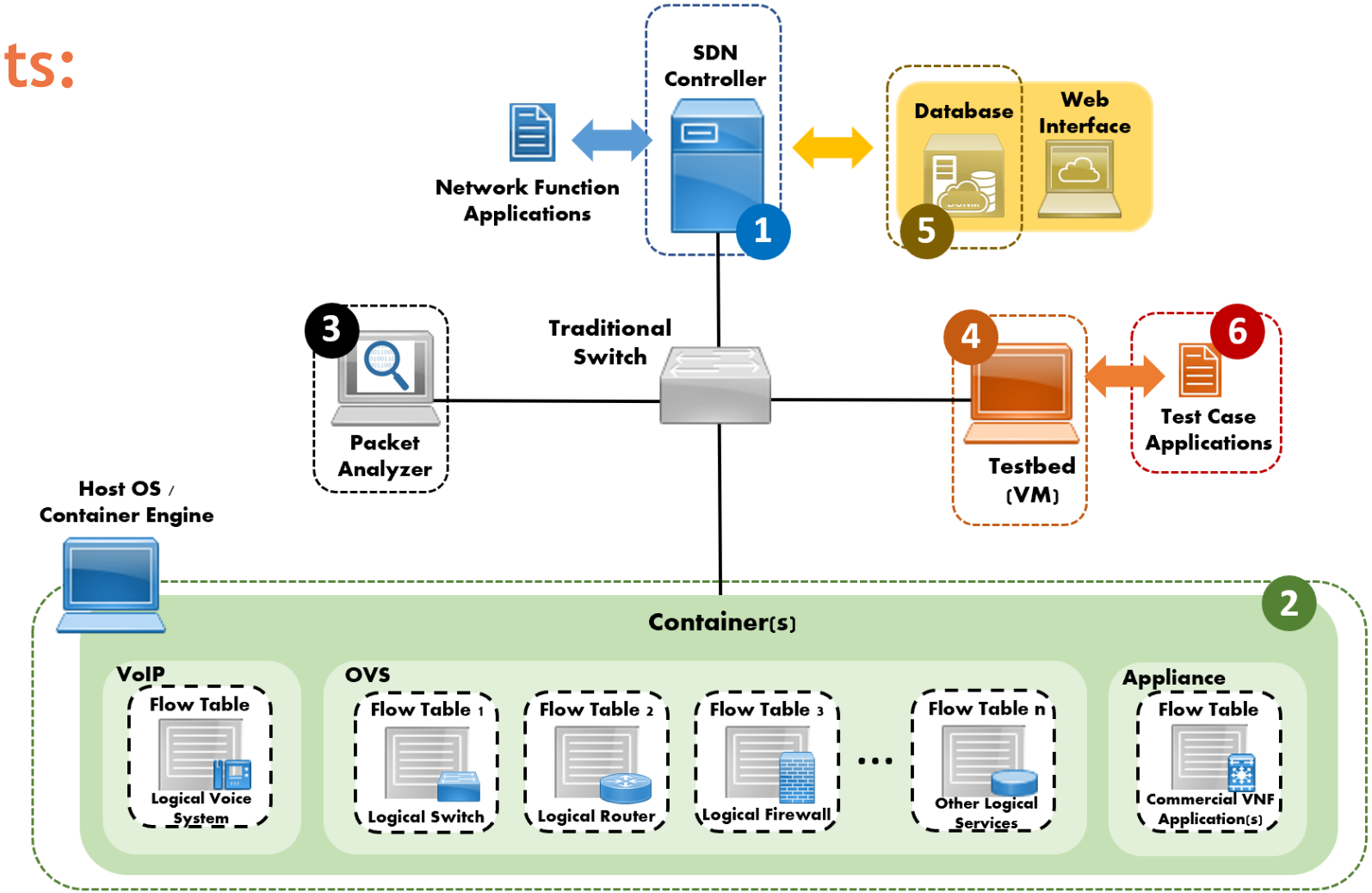


Bare Metal Speed

Baseline Design from CDD

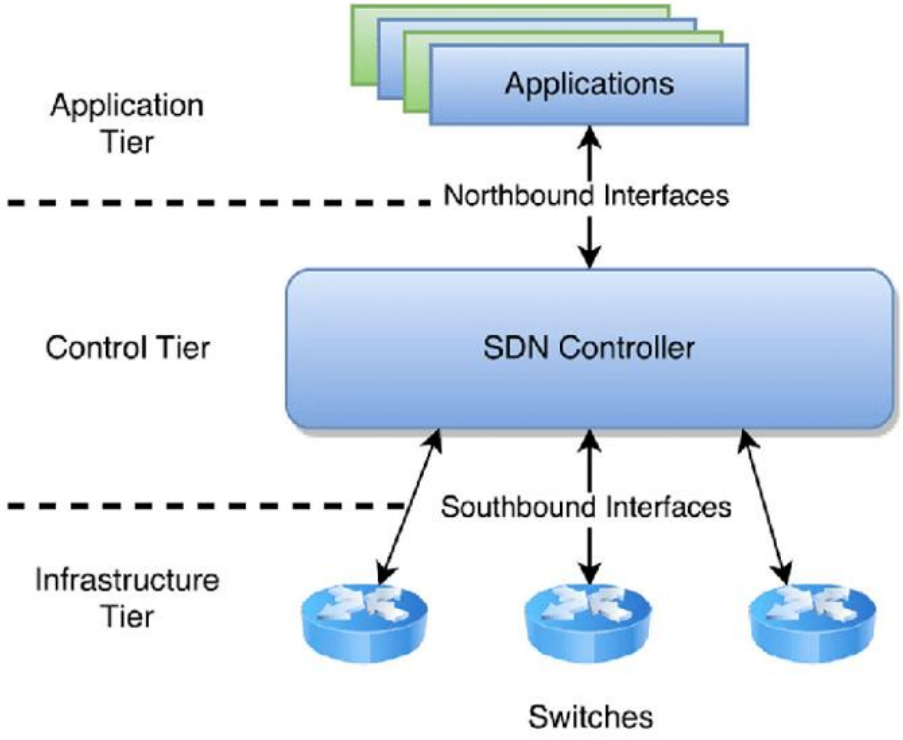
Key Design Elements:

1 SDN Controller
2 Containers
3 Packet Analyzer
4 Hypervisor
5 Database manager
6 Traffic generator



SDN Controller

- Application layer in SDN: Responsible for provisioning, orchestration and abstraction of flows in the network devices
- Brain of the network
- Global view
- Open-source vs Commercial



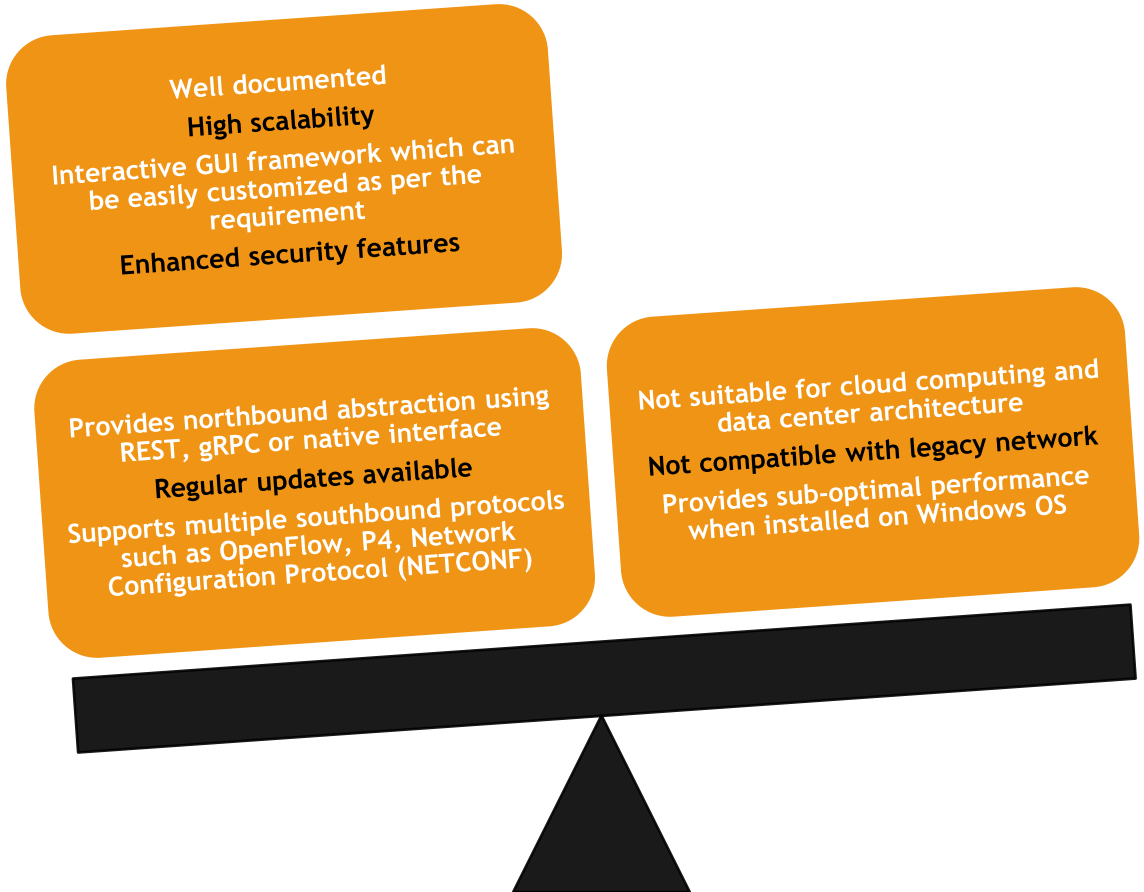
Baseline Design from CDD – SDN Controller

Comparison of SDN Controllers

Parameters	Weights	ONOS	ODL	Ryu	Floodlight	Cisco APIC
Cost	0.20	**5**	5	5	5	1
Open-source	0.15	**4**	3	4	4	1
Feature Support	0.10	**4**	2	3	4	2
Programmability	0.15	**5**	3	4	2	3
Community Support & Documentation	0.15	**4**	2	3	3	3
Learning Curve	0.10	**4**	2	3	3	3
Modularity	0.15	**4**	4	3	3	2
Total	1.00	**4.35**	3.20	3.70	3.50	2.05

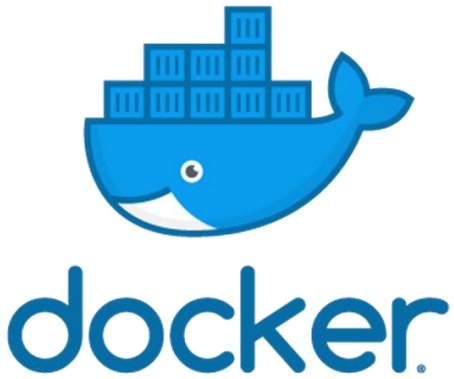
5 – Best choice for the Project
1 – Worst choice for the Project

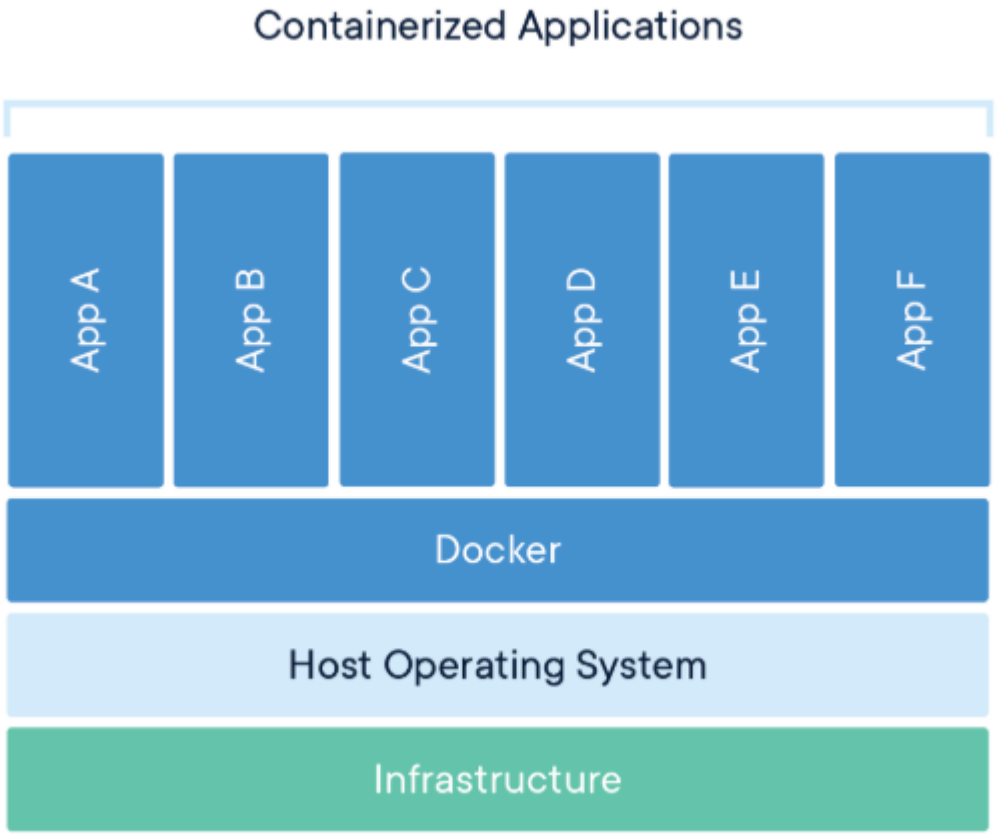
Pros & Cons of ONOS SDN Controller



Containers

- Standalone, executable software packages: Encases code, user libraries and other dependencies needed to run an application
- Easy to deploy
- Operates on Hosts Operating System





Baseline Design from CDD – Containers

Pros & Cons of Docker Containers

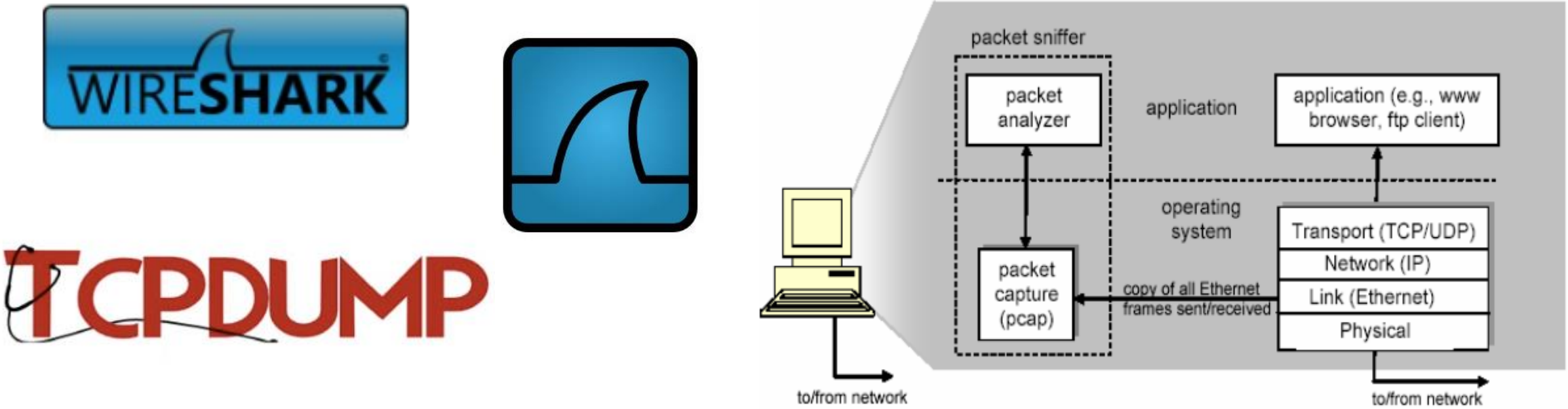
Comparison of Containers

Parameters	Weights	Docker	Rocket	Solaris Containers
Cost	0.20	**5**	5	1
Open-source	0.15	**4**	3	1
Compatibility with different OS	0.20	**4**	2	1
Community Support & Documentation	0.15	**4**	4	3
Learning Curve	0.10	**3**	4	4
Ease of Deployment	0.20	**4**	3	2
Total	1.00	**4.10**	3.45	1.80

5 – Best choice for the Project
1 – Worst choice for the Project

Packet Analyzer

- Software that is used to capture and perform packet analysis of the network traffic
- Visualization of network bandwidth and resource utilization [5]



[5] https://en.wikipedia.org/wiki/Packet_analyzer

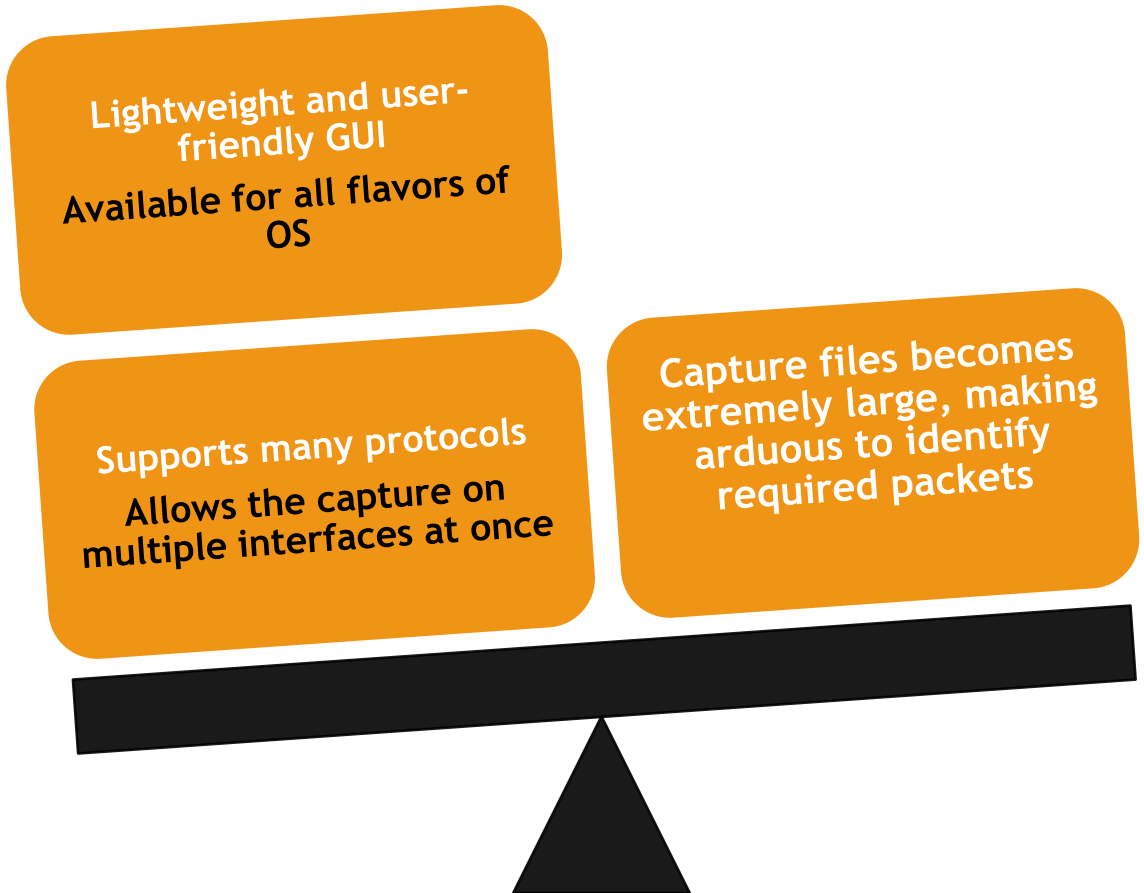
Baseline Design from CDD – Packet Analyzer

Comparison of Packet Analyzers

Parameters	Weights	Wireshark	Tshark	Tcpdump
Cost	0.20	**5**	5	5
Open-source	0.15	**5**	5	5
Feature Support	0.10	**4**	4	3
Programmability	0.15	**5**	5	4
Community Support & Documentation	0.15	**5**	4	4
Learning Curve	0.10	**4**	4	3
Modularity	0.15	**5**	5	4
Total	1.00	**4.80**	4.65	4.15

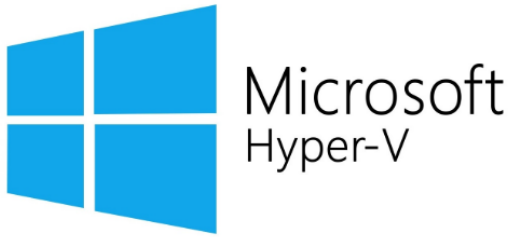
5 – Best choice for the Project
1 – Worst choice for the Project

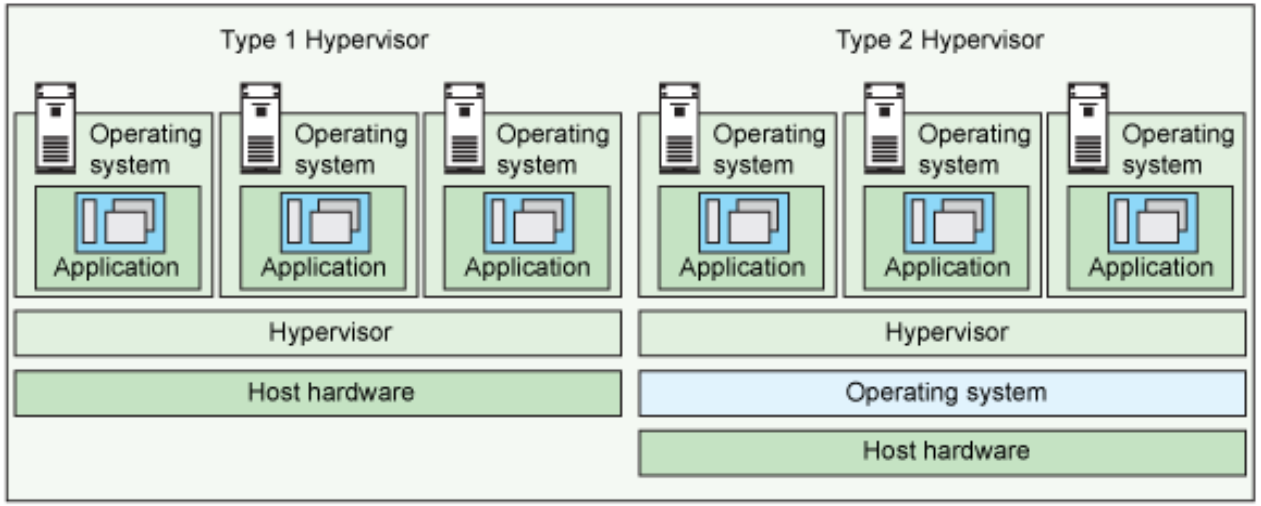
Pros & Cons of Wireshark Packet Analyzer



Hypervisors

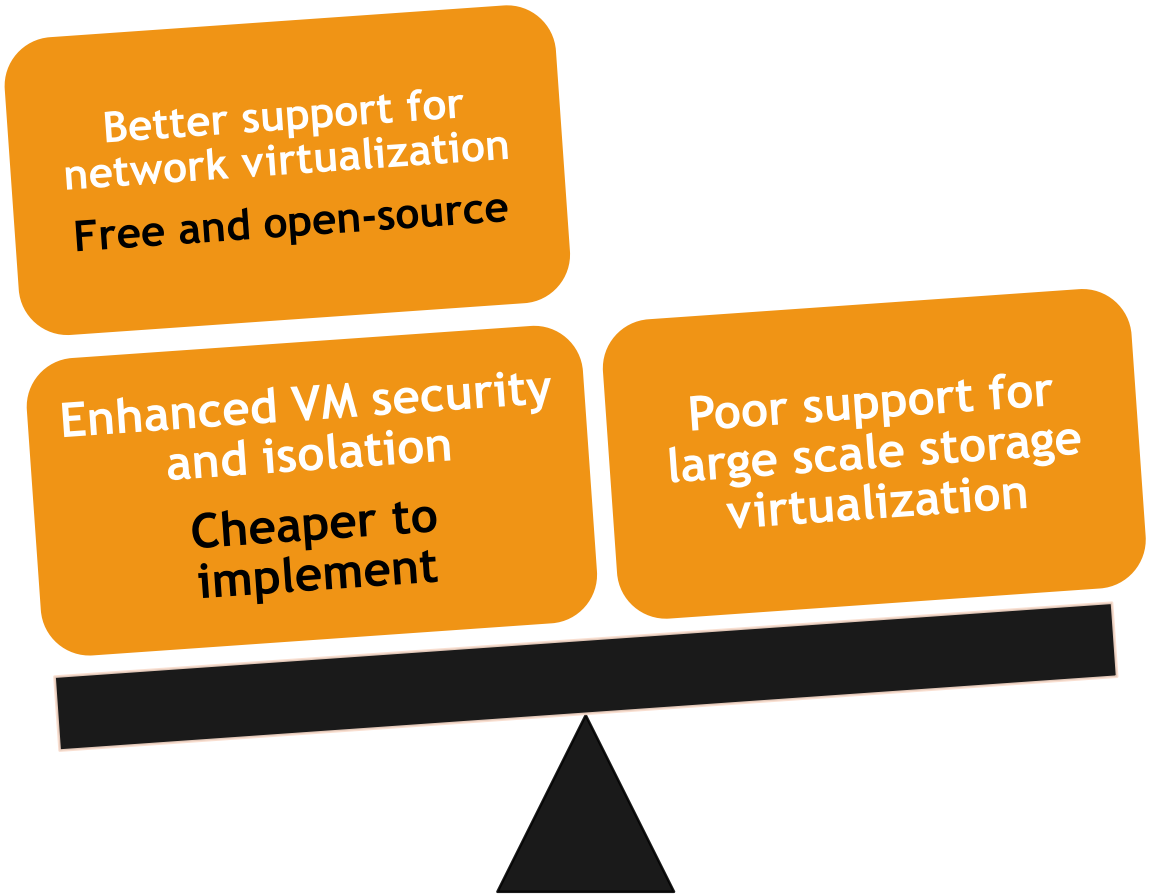
- Software that enables multiple Operating Systems to share the same hardware resources
- Controls the allocation of physical host hardware and kernel resources such as memory, disk space on each virtualized OS





Baseline Design from CDD – Hypervisors

Pros & Cons of KVM Hypervisor



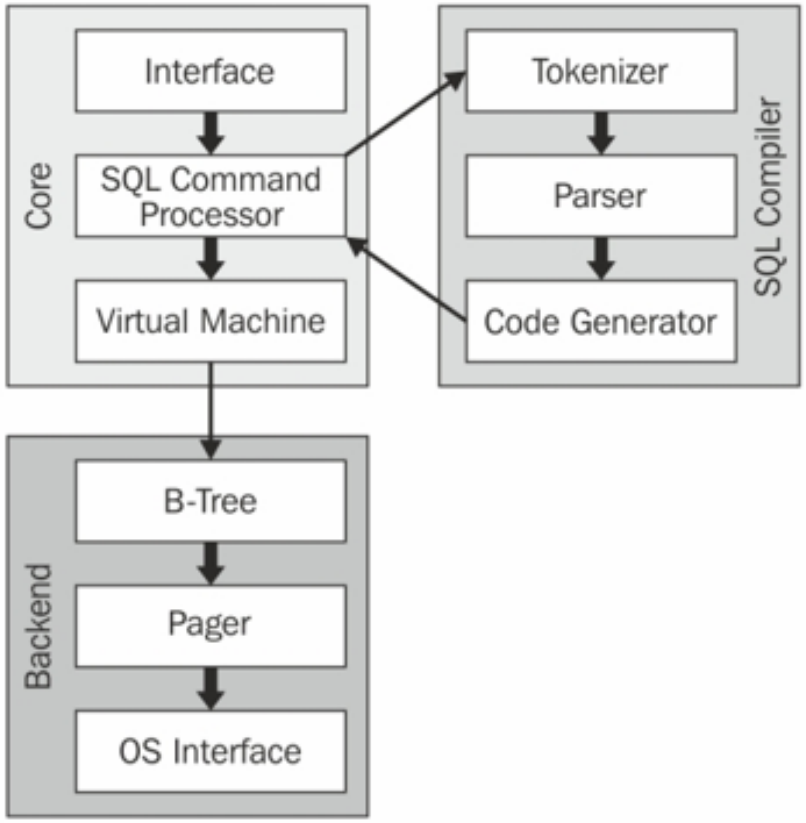
Comparison of Hypervisors

Parameters	Weights	ESXi	KVM	Hyper V	VMware Workstation	VirtualBox	QEMU
Cost	0.20	2	**5**	2	3	5	5
Open-source	0.15	4	**4**	2	4	4	4
Feature Support	0.10	3	**4**	3	4	4	3
Community Support & Documentation	0.15	3	**4**	3	4	3	3
Learning Curve	0.10	3	**4**	3	4	4	3
Modularity	0.15	2	**4**	2	4	4	3
Bare Metal Speed	0.15	4	**4**	4	2	2	2
Total	1.00	2.95	**4.20**	2.65	3.50	3.75	3.40

5 – Best choice for the Project
1 – Worst choice for the Project

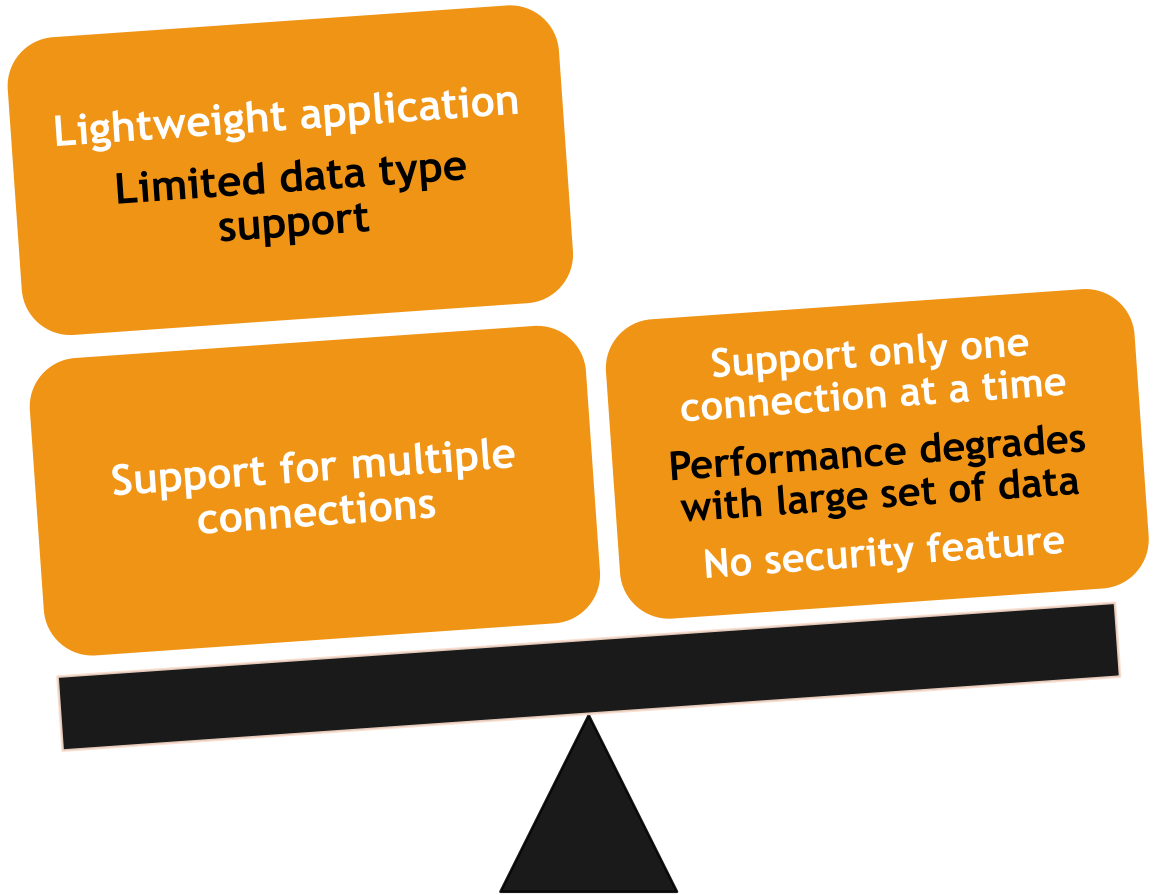
Database Storage

- Logical Structures: Used to store and index data
- Collection of data can be accessed, updated and analyzed using tools



Baseline Design from CDD – Database Storage

Pros & Cons of SQLite Database Storage



Comparison of Database Storages

Parameters	Weights	Oracle	MySQL	SQLite
Cost	0.20	1	5	**5**
Open-source	0.15	1	5	**5**
Feature Support	0.10	2	4	**3**
Programmability	0.15	3	4	**4**
Community Support & Documentation	0.15	3	4	**4**
Learning Curve	0.10	2	3	**4**
Modularity	0.15	2	4	**4**
Total	1.00	1.95	4.25	**4.25**

5 – Best choice for the Project
1 – Worst choice for the Project

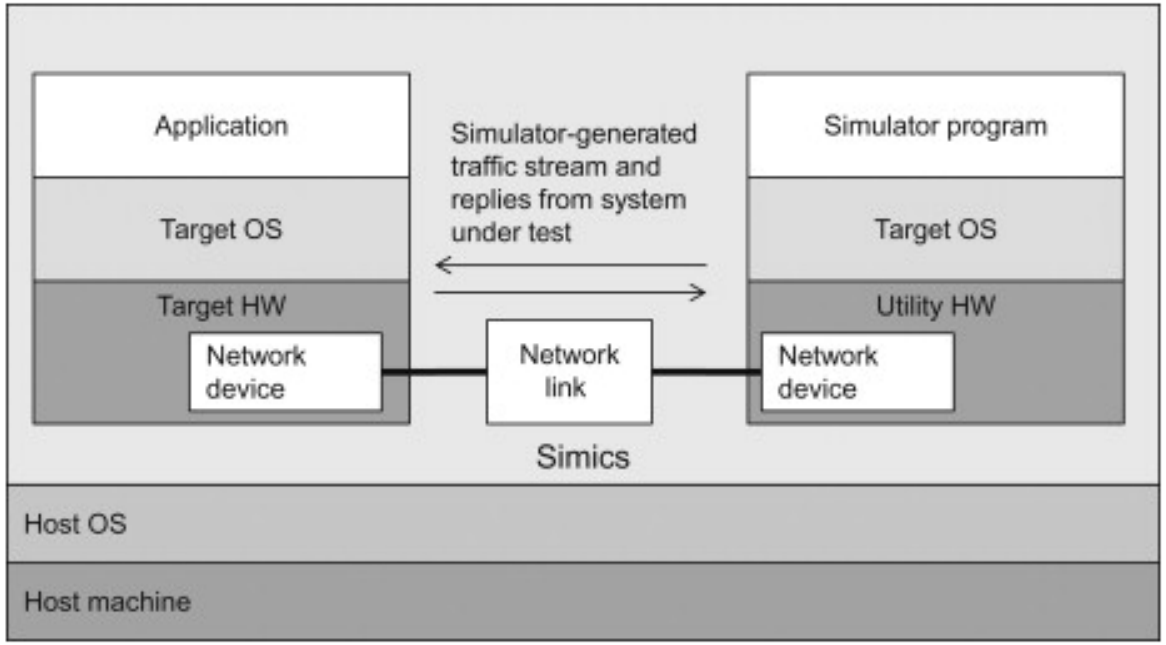
Traffic Generator

- Tool used to generate customized traffic and monitor network performance parameters
- Test the robustness of network under different network scenarios

OSTINATO
Network Traffic Generator and Analyzer

Iperf
Testing the limits of your network

ixia



Baseline Design from CDD – Traffic Generator

Comparison of Traffic Generators

Parameters	Weights	Ostinato	Iperf	Ixia	WAN Killer
Cost	0.20	5	**5**	2	2
Open-source	0.15	4	**4**	2	2
Feature Support	0.10	2	**4**	3	2
Programmability	0.15	4	**4**	3	2
Community Support & Documentation	0.15	2	**4**	3	4
Learning Curve	0.10	2	**4**	2	2
Modularity	0.15	3	**4**	2	2
Total	1.00	3.35	**4.20**	2.40	2.30

5 – Best choice for the Project
1 – Worst choice for the Project

Pros & Cons of Iperf Traffic Generator

Baseline Design Summary

Project Elements	Selected Element	Rationale
SDN Controller	ONOS	Provides northbound abstraction using REST Supports multiple southbound protocols Well documented Interactive GUI
Container	Docker	Lightweight Can be deployed on Linux and Windows OS Suitable for continuous deployment and integration
Hypervisor	KVM	Enhanced VM security and isolation Bare metal speed Better network virtualization support

Baseline Design Summary (continued)

Project Elements	Selected Element	Rationale
Packet Analyzer	Wireshark	Lightweight and user-friendly GUI Available for all flavors of OS Supports many protocols Allows the capture on multiple interfaces at once
Database Manager	SQLite	Support for multiple connections Lightweight
Traffic Generator	Iperf	Easy to install and implement across multi-vendor platforms Supports both IPv4 and IPv6 traffic The results are saved in easily readable log files Supports multithreading

Baseline Design Summary (continued)

Project Elements	Selected Element	Rationale
Southbound Protocol	OpenFlow	Standard SDN Protocol Supported by most existing hardware Supports TCP & TLS connection
Northbound Protocol	REST API	Supported by most devices Easy to implement using curl & requests.post method Popular in industry

Technical Feasibility for SDN Controller

Licensing

- ONOS is an open-source carrier grade SDN controller that is licensed under Apache 2.0
- The ONOS image can be downloaded from the internet without cost

Feature set availability

- ONOS supports OpenFlow version 1.0 and 1.3
- Supports configuration of legacy (brown field) and SDN (green field) network
- Provides view of network elements, connectivity and network errors through GUI
- Supports southbound interface abstraction [6]
- Can run as a distributed system across multiple servers

Software development requirement

- No modification to the source code
- Modular Java applications that spawn containers on hosts and install flows function-specific flows on hosts

Skills required

- Basic traditional networking knowledge
- Ability to write and interpret code in REST API

[6] https://onosproject.org/features/

Technical Feasibility for SDN Controller (continued)

1. Starting the ONOS controller on a Hypervisor

```
sdn@sdn-controllers:~$ sudo /opt/onos/bin/onos-service start
[sudo] password for sdn:
Welcome to Open Network Operating System (ONOS)!

Documentation: wiki.onosproject.org
Tutorials:     tutorials.onosproject.org
Mailing lists: lists.onosproject.org
```

2. Activate in-built and customized apps

```
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos> app activate org.onosproject.fwd
```

3. Intents can be defined from the GUI

Intents (25 total)

Application ID	Key	Type	Priority
29 : org.onosproject.cli-random	0x57	HostToHostIntent	100
Resources: 00:00:00:00:00:04/-1, 00:00:00:00:00:17/-1			
Details: Host 1: 00:00:00:00:00:04/-1, Host 2: 00:00:00:00:00:17/-1			

4. Intuitive REST calls for Network Management

ONOS Fault Management Application REST API

APIs for interacting with the Fault Management application.

alarms : Alarms on devices or ONOS — Show/Hide | List Operations | Expand Operations

PUT	/alarms/{alarm_id}	Update book-keeping fields on the alarm
GET	/alarms	Get all alarms
GET	/alarms/{id}	Get specified alarm

Technical Feasibility for Container

Licensing

- Docker is an open-source container software, licensed under the Apache 2.0 license
- Open vSwitch (OVS) is licensed under Apache 2.0 license

Software availability / Feature set availability

- Flows will be used to define multiple network services that are handled by OVS

Network automation

- Multiple VNF's will be used to create service chain, and flow entry addition would be automated

Skills required

- Ability to write and manipulate python code
- Installing docker's components and libraries

Technical Feasibility for Container (continued)

1. Command to check images listed under Dockers

```
root@shivababa-Lenovo-Y70-70-Touch:~# docker images
REPOSITORY                                TAG                 IMAGE ID            CREATED             SIZE
gcr.io/vaulted-zodiac-236605/worker       <none>              a05557ef73f1        5 days ago          882MB
rabbit                                    latest              5e099b52d0a9        5 days ago          150MB
gcr.io/vaulted-zodiac-236605/rabbitmq     latest              5e099b52d0a9        5 days ago          150MB
worker                                    latest              75f819eeaea3        6 days ago          515MB
gcr.io/vaulted-zodiac-236605/rest_server  latest              75f819eeaea3        6 days ago          515MB
gcr.io/vaulted-zodiac-236605/worker       latest              9d3fdbf4fc2c        6 days ago          882MB
gcr.io/vaulted-zodiac-236605/rabbitmq     <none>              b9ec6120de8e        6 days ago          1.31GB
<none>                                    <none>              1498723f5658        6 days ago          1.31GB
<none>                                    <none>              caeac3abefd6        6 days ago          252MB
<none>                                    <none>              52ec194d3513        6 days ago          252MB
<none>                                    <none>              1f0c3ed29546        6 days ago          252MB
<none>                                    <none>              749e93c96d73        6 days ago          141MB
<none>                                    <none>              378ecefe8f10        6 days ago          141MB
gcr.io/vaulted-zodiac-236605/rabbitmq     <none>              57002ea5f9d0        6 days ago          150MB
gcr.io/vaulted-zodiac-236605/rest_server  <none>              a0ceef69b0ca        6 days ago          515MB
gcr.io/vaulted-zodiac-236605/worker       <none>              3d881aacf2e8        6 days ago          882MB
rabbitmq                                  latest              843e6712e712        8 days ago          150MB
gcr.io/vaulted-zodiac-236605/rabbitmq     <none>              843e6712e712        8 days ago          150MB
ubuntu                                    19.04               51b0783967fc        12 days ago         70MB
redis                                     latest              de25a81a5a0b        3 weeks ago         98.2MB
gcr.io/vaulted-zodiac-236605/redis        latest              de25a81a5a0b        3 weeks ago         98.2MB
root@shivababa-Lenovo-Y70-70-Touch:~#
```

2. Command to run an image in Dockers

```
root@shivababa-Lenovo-Y70-70-Touch:~# docker run -td de25a81a5a0b
07e91ee7b3a4fe73d655dfa5e293895ccd5e9a600db16795a3a905ecb9e64746
root@shivababa-Lenovo-Y70-70-Touch:~# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               6379/tcp
07e91ee7b3a4        de25a81a5a0b        "docker-entrypoint.s…"   6 seconds ago       Up 2 seconds
```

3. Command to check functionality of the image

```
root@shivababa-Lenovo-Y70-70-Touch:~# docker exec -it 07e91ee7b3a4 /bin/sh
# ls
# redis-cli
127.0.0.1:6379> PING
PONG
127.0.0.1:6379>
```

Technical Feasibility for Packet Analyzer

Licensing

- Wireshark is a free software and licensed under GNU General Public License
- Runs on cross-platforms

Software availability / Feature set availability

- Wireshark supports all standard networking protocols and all OpenFlow versions
- Provides excellent graphical front-end with multiple filter options
- Wireshark captures application specific traffic too, e.g. VoIP, which will be used for VNF's performance analysis [7]
- Traffic capture can be done across various interfaces like Ethernet, loopback, and sub-interface

Software development requirement

- No software development will be required for packet analysis
- Raw/dumpcap packet capturing can be done with only superuser privileges, hence more secure

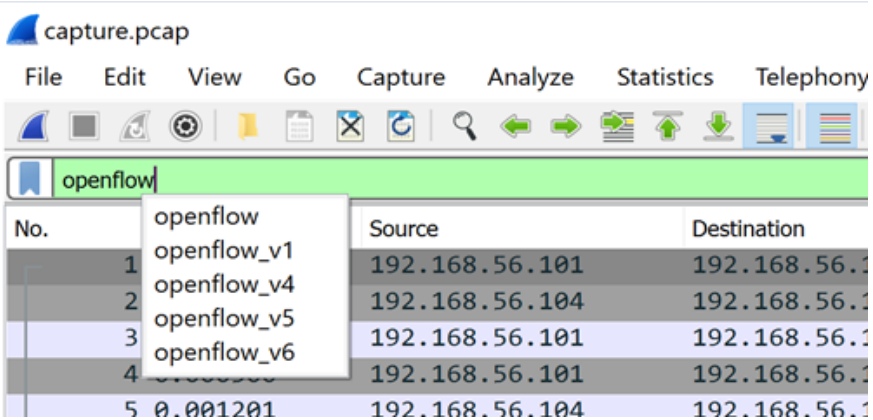
Skills required

- Ability to run and capture traffic files, essential for network automation in python
- Understanding of basic networking protocols
- Familiarizing with various filters

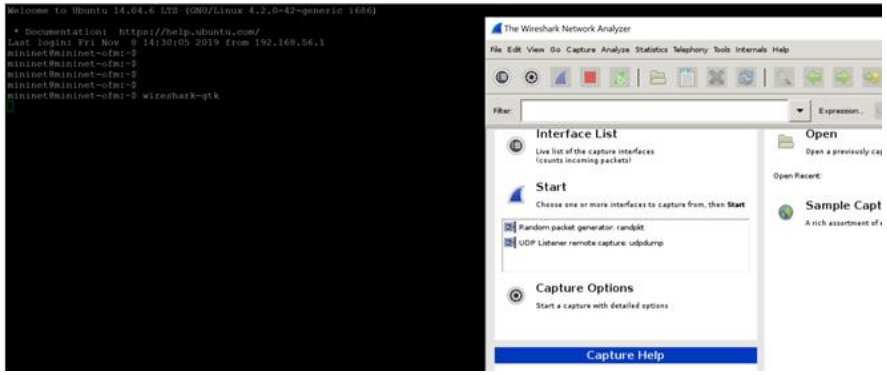
[7] https://www.wireshark.org/about

Technical Feasibility for Packet Analyzer (continued)

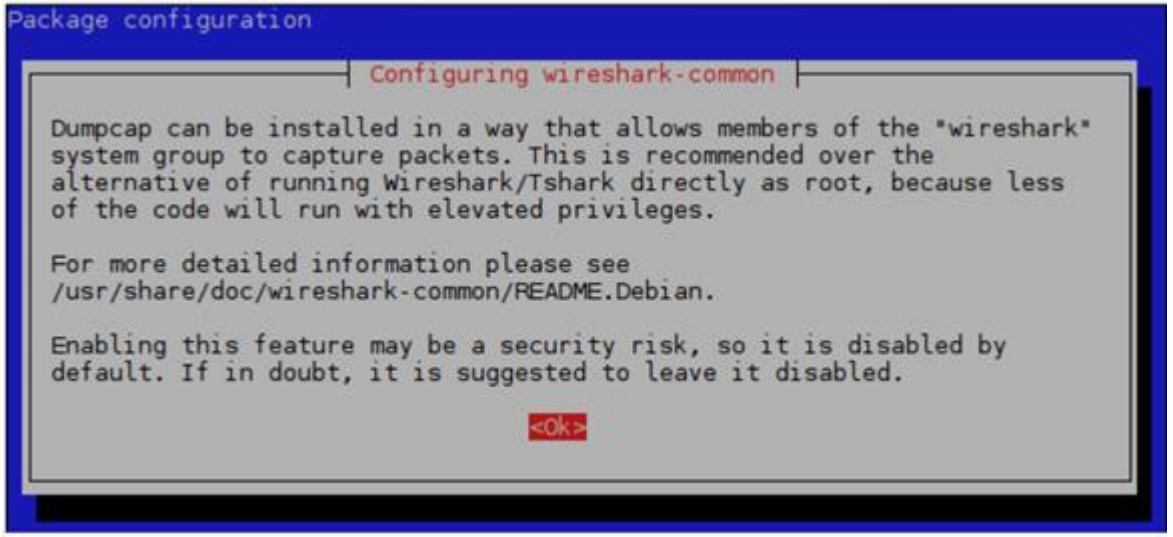
Supports various OpenFlow versions

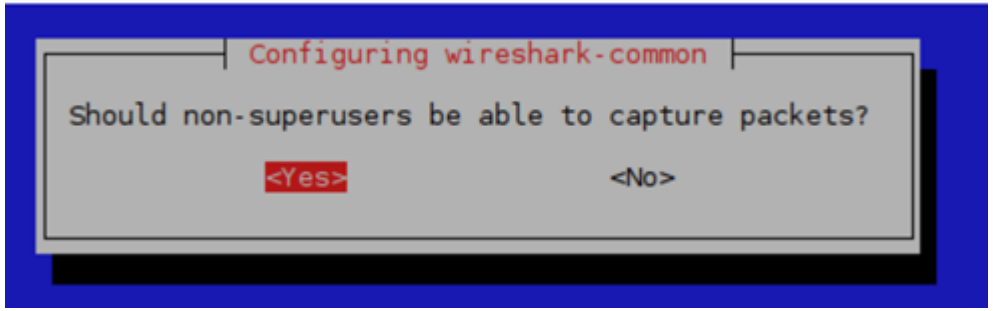


Ease to run application from the terminal, great for automation



Superuser privilege to capture raw packets, hence more secure





Technical Feasibility for Database

Licensing

- SQLite is licensed under GNU GPLv2 for open source projects
- Runs on cross platforms

Software availability / Feature set availability

- Single database file for easy accessibility
- Lightweight application
- Supports multiple application access to same database for read/write operations
- No intermediary server process: Access to read/write the database can be done directly on host machine disk

Software development requirement

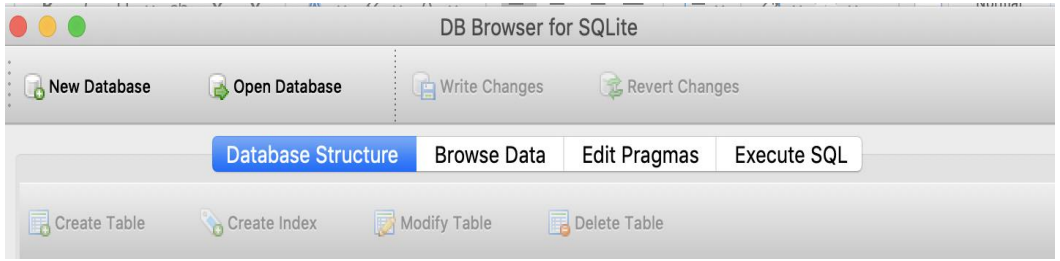
- No software development will be required

Skills required

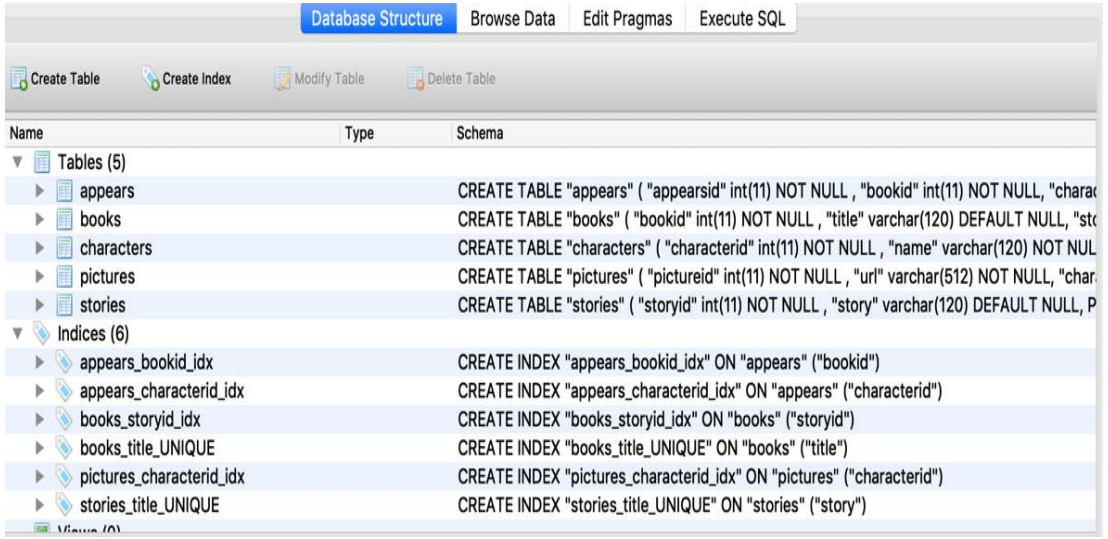
- Ability to write queries and integrate with python code
- Ability to define the right schema for monitoring metrics

Technical Feasibility for Database (continued)

User friendly GUI for easy management



Easy application setup and schema definitions



Resource Feasibility

SDN Controller

- Utilizing the controller supporting hardware available in the Telecommunications Lab
- The SDN controller selected would be hosted on this hardware

Testbed

- Physical server available in the Telecommunications lab will be utilized to emulate Linux based testbed using KVM as the hypervisor

Containers

- OVS switches would be used to host multiple network services on host OS
- Containers will be used to encase OVS switches
- Laptops that support Windows/Linux/macOS would be used as host devices

Intermediate Hardware Devices

- Devices such as traditional L2 switch would be utilized to interconnect SDN Controller and testbed to the end hosts

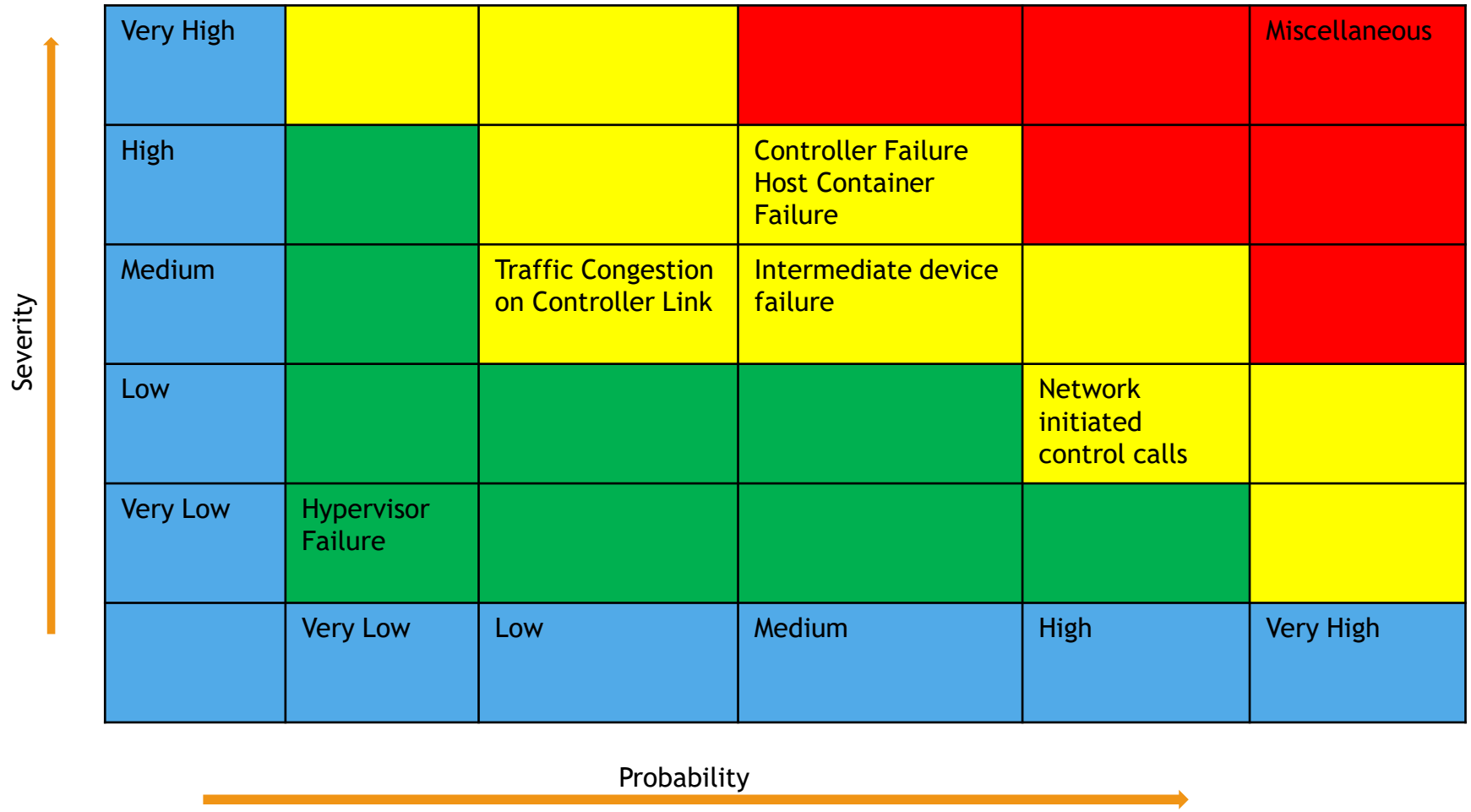
Cost Feasibility

VNFs will be hosted and tested on Windows, Linux, and macOS, hence procurement of laptops that supports Windows/Linux and macOS are required

Besides laptops, Telecommunication lab resources, and open-source software's, no additional resource requirements are necessary

Laptops are used to build, test, and simulate containers, code, and network service chaining

Risk Factors and their Impact

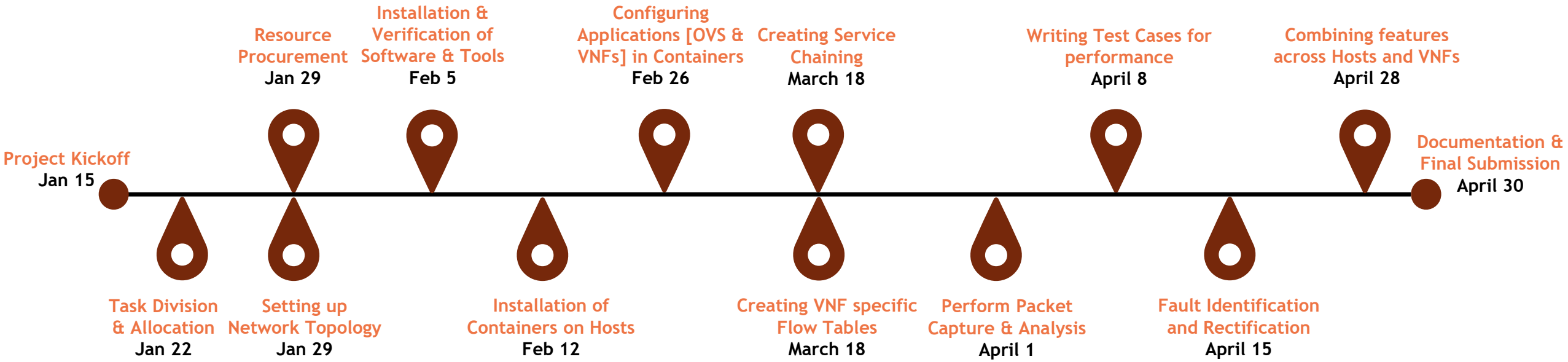


Very High					Miscellaneous
High			Controller Failure Host Container Failure		
Medium		Traffic Congestion on Controller Link	Intermediate device failure		
Low				Network initiated control calls	
Very Low	Hypervisor Failure				
	Very Low	Low	Medium	High	Very High

Risks and Mitigation

	Risk	Mitigation Action
1.	Controller Failure	Introduce a secondary controller
2.	Host Container Failure	Spin up a backup container on test VM
3.	Intermediate device failure	Use a wireless router for redundant connection between devices
4.	Traffic Congestion on Controller Link	Backup secondary link for test traffic
5.	Hypervisor Failure	Spin up a backup test bed on SDN Controller hardware
6.	Network Control Calls	Implement proactive flows.

Project Targets and Key Deadlines



Q&A

Thank You