"VIRTUALIZED NETWORK DESIGN & IMPLEMENTATION: LEVERAGING SDN, NFV, AUTOMATION, AND REDUNDANCY"

Overview

- Introduction
- Network Design and Architecture
- Testing and Validation
- Conclusion

Introduction

The execution of the project focuses on a virtualized network employing Software-Defined Networking for centralized control, Network Function Virtualization for effectiveness of the service, automation providing efficient network management, and embedded redundancy for enhanced dependability and reliability. Resulting a resilient network architecture, a scalable and flexible network design and architecture.

Scope

- **SDN:** The recommended SDN architecture establishes advanced controllers for centralized handling of network traffic via an SDN controller, such as OpenDaylight or ONOS.
- **NFV:** It enables the filtering and hosting of applications such as firewalls, load balancing, VPN to allow docker and virtual machines.
- **Automation:** Modifications and control of setups using scripts and Traffic flows, Python, and Ansible consisting of scripts.
- **Redundancy:** Safeguards measures and provision of standby connectivity to test the network at times of failure situations.

Network Design and Architecture

Network design uses SDN, NFV, automation, and redundancy to create a robust, elastic virtualized architecture. Services like virtual firewalls and load balancers perform traditional hardware functions, while automated configuration and monitoring ensure high availability.

TECHNOLOGIES USED:

Network Simulators:

It addressed for the project's network environment design and simulation utilizing GNS3 or EVE-NG.

SDN Controller:

OpenDaylight, is the recommended network operating system that will be in charge of the project's network's central administration.

NFV Tools:

Installing Docker for a Virtual Network Function hosting, such as firewalls, VPNs, and other load balancers in Ubuntu.

TECHNOLOGIES USED:

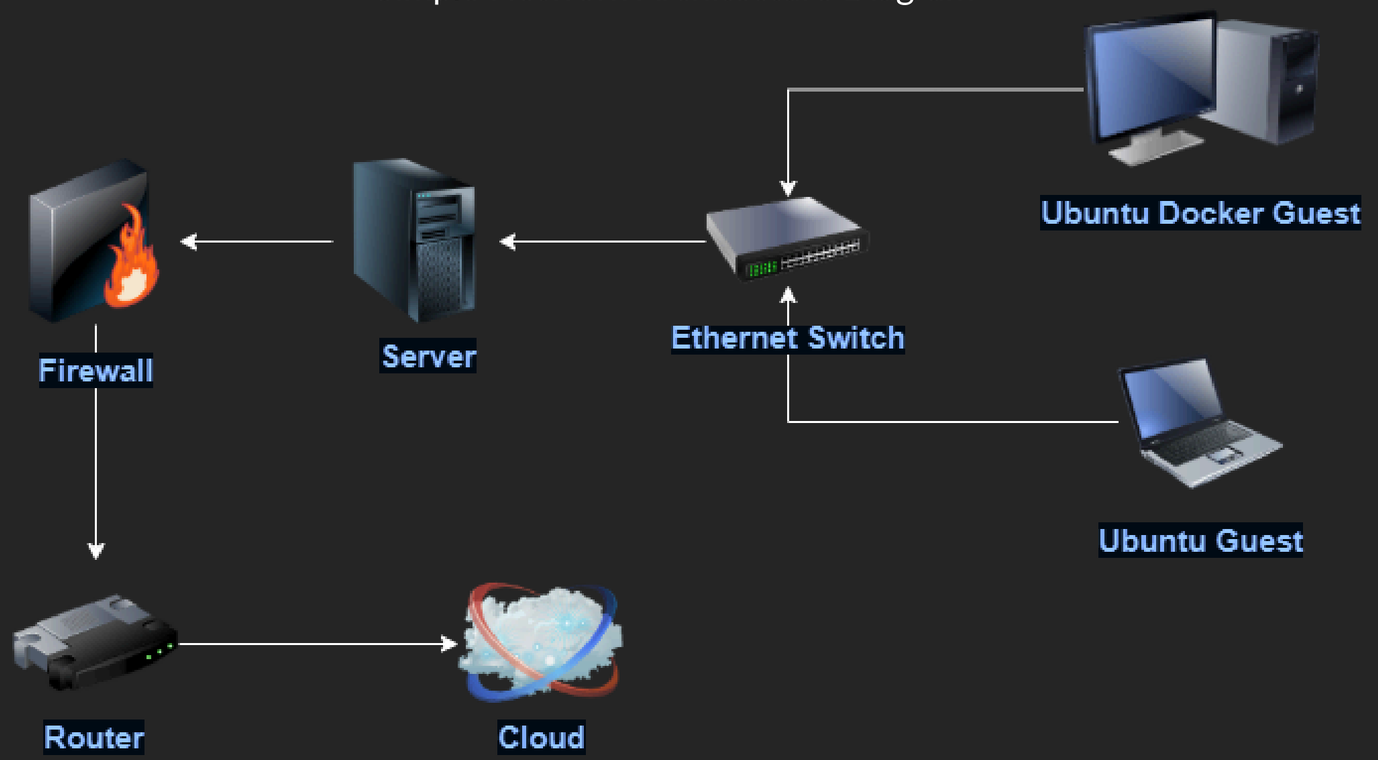
Automation Tools:

In automation, we have utilized Python and Ansible specifically for controlling automated tasks required for this project and for scripting network setups.

Visualization Tools:

Network visualizations created using the Draw.io.

Simple Network Architecture Diagram



The network design uses SDN for centralized management and control, enabling dynamic traffic control. It uses network function virtualization (NFV) for scalability and reduces hardware reliance. High-level redundancy techniques reduce outage time. This robust, flexible, and affordable network is suitable for complex software applications.

Testing and Validation

Ping Tests: These were carried out using the command interfaces including CMD and Ubuntu terminal to initialize hardware connection checks for confirmatory network connectivity of the nodes in the network.

Traffic Flow Validation: While not fully stated this presumably entails the application of various means to check correct routing and delivery of the packets through the network's canopy.

Log Collection: The logs were obtained with Windows PowerShell for diagnostic purposes. This involves tracking for errors, monitoring the networks and satisfying ourselves that certain configurations were done properly.

Docker Troubleshooting: Then for the Virtualized Network Functions (VNFs), the troubleshooting was done by managing Docker containers. For instance, rebooting the virtual firewall with specific Docker commands and checking all the containers' functionality.

Network Simulator Testing: Tools like GNS3 or EVE-NG can only have been used for emulating the network virtualization and the configurations tested before their implementation here.

Automation Verification: All the scripts that were written in Python and Ansible were run to check for any syntax errors, and overall test to see whether automation was achieved as planned.

Conclusion

This project developed a reliable, scalable, and flexible network using Software-Defined Networking (SDN), Network Function Virtualization (NFV), automation, and redundancy. Technologies used included SDN Controllers, NFV Tools, Network Simulators, and Visualization Tools. Traffic was centralized, virtual functions reduced hardware reliance, automated configuration, network monitoring, and backup mechanisms. Testing activities assessed connectivity, network functionalities, and automated script validation. Potential enhancements include finer network design, multiple controllers for better traffic handling, machine learning-based statistical analysis, advanced encryption, and intrusion detection systems.

Thank You