ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη vCPE και vRouter για υπηρεσία VPN

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Καρογιάννης Ε. Ιωάννης

Επιβλέπων : Ευστάθιος Συκάς
Καθηγητής Ε.Μ.Π.

Αθήνα, Σεπτέμβριος 2017

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη vCPE και vRouter για υπηρεσία VPN

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Καρογιάννης Ε. Ιωάννης

Επιβλέπων : Ευστάθιος Συκάς
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 5η Σεπτεμβρίου 2017

.....	.....	.....
Ευστάθιος Συκάς	Συμεών Παπαβασιλείου	Γεώργιος Στασινόπουλος
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Σεπτέμβριος 2017

.....

Καρογιάννης Ε. Ιωάννης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Καρογιάννης Ε. Ιωάννης, 2017
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη και χρήση δύο Raspberry Pi's ως επιχειρησιακών vCPEs κι ενός εικονικού δρομολογητή (vRouter) για τη δημιουργία ενός εικονικού ιδιωτικού δικτύου (VPN – Virtual Private Network). Ταυτόχρονα, γίνεται χρήση τεχνολογιών SDN (Openflow) για την επικοινωνία των vCPEs με τον κεντρικό εικονικό δρομολογητή (WAN – SDN).

Ο εικονικός συνδρομητικός εξοπλισμός vCPE (virtual Customer Premise Equipment) είναι ένας εναλλακτικός τρόπος παροχής ευρυζωνικών υπηρεσιών στους συνδρομητές μέσω λογισμικού παρά με τη χρήση ειδικού εξοπλισμού. Για την ασφαλή επικοινωνία μεταξύ πολλών σημείων χρησιμοποιώντας το κοινόχρηστο δίκτυο ενός παρόχου απαιτείται η δημιουργία ενός εικονικού δικτύου (VPN – Virtual Private Network). Με στόχο την αποδοτικότερη διαχείριση κι εύκολη προσθήκη σημείων αλλά και τη χρησιμοποίηση τεχνολογιών Network Function Virtualization (NFV), οι παραδοσιακοί φυσικοί δρομολογητές μετασχηματίζονται σε εικονικές μηχανές, οι οποίες εκτελούνται εντός τυπικών εξυπηρετητών χωρίς τη χρήση εξειδικευμένου υλικού. Τα λειτουργικά συστήματα που επιλέχθηκαν για τη δημιουργία των vRouters είναι το VyOS και το CSR 1000v.

Το SDN (Software Defined Networking) είναι μια νέα αρχιτεκτονική που διαχωρίζει το στρώμα προώθησης δεδομένων από το στρώμα διαχείρισης. Το στρώμα διαχείρισης μπορεί απευθείας να προγραμματιστεί, ενώ το φυσικό δίκτυο που είναι κοινό μπορεί να αντιληφθεί τις διάφορες υπηρεσίες. Βασικό συστατικό της αρχιτεκτονικής είναι ο ελεγκτής (controller) που συντηρεί τον πίνακα δρομολόγησης των διαφορετικών ροών όπως ενημερώνονται από το επίπεδο ελέγχου. Ο ελεγκτής είναι ένα κεντρικοποιημένο σύστημα διαχείρισης της πληροφορίας δρομολόγησης. Ο ελεγκτής που επιλέχθηκε για την εποπτεία του δικτύου είναι ο OpenDaylight Controller, ενώ η παραμετροποίηση του δικτύου γίνεται μέσω του REST API του ελεγκτή και του πρωτοκόλλου RESTCONF. Η διαχείριση των ροών δρομολόγησης του δικτύου γίνεται με χρήση του εργαλείου Postman.

Λέξεις Κλειδιά: VPN, NFV, vCPE, vRouter, SDN, OpenDaylight Controller, REST API, RESTCONF, ροή δρομολόγησης

Abstract

The purpose of this diploma thesis is to develop and use two Raspberry Pi's as corporate vCPEs and a vRouter to create a Virtual Private Network (VPN). At the same time, SDN technologies are being used (Openflow) for the communication between the vCPEs and the central vRouter (WAN – SDN).

Virtual Customer Premise Equipment (vCPE) is an alternative way of providing telecommunication services to customers through the use of software rather than custom hardware. The installation of a VPN guarantees the secure communication between multiple nodes within the public network. In order to efficiently manage and add nodes using Network Functions Virtualization (NFV) techniques, traditional physical routers are replaced by Virtual Machines (VMs) that run on typical servers without the need for custom hardware. The operational systems that were selected for the vRouters are VyOS and CSR 1000v.

Software Defined Networking (SDN) is a new architectural design that separates the forwarding plane from the control plane. The control plane can be directly programmed, while the physical network which is common remains aware of the various services. The fundamental component of this architecture is the controller that handles the flow routing table, as they are accordingly updated through the control plane. The controller is a centralized management system that handles routing information. The controller that was selected for the network management is the OpenDaylight Controller, while the network configuration is handled through the controller's REST API and the RESTCONF protocol. The routing flow management is accomplished through the Postman tool.

Keywords: VPN, NFV, vCPE, vRouter, SDN, OpenDaylight Controller, REST API, RESTCONF, routing flow

Ευχαριστίες

Η επίδειξη της παρούσας διπλωματικής εργασίας αντιπροσωπεύει το τελικό στάδιο των σπουδών μου στο Εθνικό Μετσόβιο Πολυτεχνείο. Σε αυτό το σημείο θα ήθελα να εκφράσω τις ειλικρινείς ευχαριστίες μου στους ανθρώπους που συνέβαλαν στην ολοκλήρωση της προπτυχιακής ακαδημαϊκής μου πορείας.

Αρχικά ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου, κ. Ευστάθιο Συκά, όχι μόνο για την καθοδήγησή του στην επιλογή θέματος, αλλά και για τον τρόπο διδασκαλίας του, ο οποίος με ενέπνευσε να εμβαθύνω στα Δίκτυα Υπολογιστών. Επίσης θα ήθελα να ευχαριστήσω θερμά τον υποψήφιο διδάκτορα κ. Πάρι Χαραλάμπου για την προθυμία που εκδήλωσε στη διευκρίνιση πολυάριθμων αποριών κι επιπλέον για τις γνώσεις που απλόχερα μου μετέδωσε κατά τη διάρκεια της συνεργασίας μας.

Έπειτα θέλω να ευχαριστήσω τους συμφοιτητές και φίλους που συνάντησα στα χρόνια των σπουδών μου, με τους οποίους συνεργαστήκαμε, μελετήσαμε, κοπιάσαμε και εν τέλει επιτύχαμε τους στόχους μας.

Επιπροσθέτως, θα ήθελα να εκφράσω την ευγνωμοσύνη μου στη μητέρα μου Φωτεινή, τη Μαρία και τη Σίλα. Πρόκειται για τρία ξεχωριστά πρόσωπα, που στάθηκαν πλάι μου σε δύσκολες στιγμές και βοήθησαν να τις ξεπεράσω με επιτυχία.

Τέλος, θέλω να αφιερώσω τη διπλωματική μου εργασία στη μνήμη του πατέρα μου, Στάθη.

Ιωάννης Ε. Καρογιάννης,
Αθήνα, Σεπτέμβριος 2017

Πίνακας Περιεχομένων

Πίνακας Εικόνων

1. Εικονικοποίηση Δικτυακών Λειτουργιών και Προγραμματιζόμενη Δικτύωση

Στα πλαίσια της διπλωματικής αυτής εργασίας, γίνεται μια προσέγγιση στην επικοινωνία ανεξαρτήτων δικτύων κάνοντας χρήση λογισμικού (software) παρά εξειδικευμένου εξοπλισμού (hardware). Συγκεκριμένα, επιδιώκεται η *μεταφορά της ευφυΐας* από το εν λόγω hardware, σε μια απλοποιημένη φυσική **συσκευή παροχής συνδρομητικών υπηρεσιών (Customer Premise Equipment – CPE)** κι η *μεταφορά της λογικής της παρεχόμενης υπηρεσίας* σε αντίστοιχο software. Η υλοποίηση αυτού του εγχειρήματος γίνεται εφικτή με τη χρήση **εικονικών CPEs (virtual CPEs)** στην επιχειρησιακή πλευρά του πελάτη και τη χρήση **εικονικού δρομολογητή (virtual router – vRouter)** από την πλευρά του παρόχου.

Επιπλέον, η διαχείριση της διακινούμενης αυτής πληροφορίας θα επιτευχθεί μέσω κεντρικοποιημένου δικτυακού ελέγχου. Αυτό καθιστά αναγκαία την αξιοποίηση της τεχνολογίας της **Προγραμματιζόμενης Δικτύωσης (Software-Defined Networking)**. Προτού λοιπόν αναλυθούν οι τεχνοτροπίες αυτές, απαιτείται η επεξήγηση των υποκείμενων τεχνολογιών που επιτρέπουν την υλοποίησή τους.

1.1 Εικονικοποίηση (Virtualization)

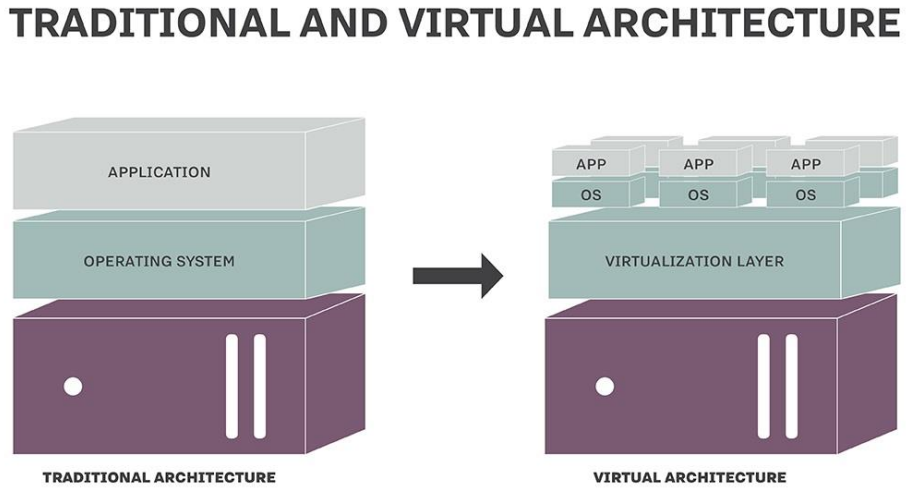
Στην επιστήμη των υπολογιστών, εικονικοποίηση είναι η δημιουργία μιας εικονικής εκδοχής μιας συσκευής ή ενός υπολογιστικού πόρου, όπως διακομιστές (servers), συσκευές αποθήκευσης, δίκτυα υπολογιστών ή ακόμη και λειτουργικά συστήματα. Η εικονικοποίηση ενός ολόκληρου υπολογιστικού συστήματος αποκαλείται **εικονική μηχανή (Virtual Machine – VM)**. Ως VM λογίζεται ένα πλήρες εικονικό σύστημα, αποτελούμενο από εικονικό hardware, λειτουργικό σύστημα κι εφαρμογές και λειτουργεί κάτω από το λειτουργικό σύστημα ενός αληθινού, φυσικού υπολογιστή.

Ένα VM συμπεριφέρεται ως αληθινό υπολογιστικό σύστημα: έχει τη δυνατότητα να φέρει σκληρούς δίσκους, κάρτα ήχου, επεξεργαστή με έναν ή περισσότερους πυρήνες, μνήμη RAM, κάρτες δικτύου, θύρες USB, κύκλωμα γραφικών, οπτικές μονάδες αποθήκευσης και BIOS. Δεδομένου ότι αποτελούνται από εικονικό μεν, τυπικό δε hardware, μπορούν να υποστηρίξουν οποιοδήποτε λειτουργικό σύστημα και υπερκείμενες εφαρμογές επιθυμεί ο χρήστης, τα οποία και εγκαθίστανται με τον ίδιο ακριβώς τρόπο όπως και στα φυσικά μηχανήματα.

Ένα επίπεδο οργάνωσης παραπάνω, αναλόγως των ικανοτήτων του φυσικού υπολογιστή που φιλοξενεί τις εικονικές μηχανές, ενδέχεται να συνυπάρχουν περισσότερα του ενός VMs στο ίδιο φυσικό μηχάνημα, τα οποία λειτουργούν ταυτόχρονα κι επικοινωνούν μεταξύ τους, συμμετέχοντας σε ένα εικονικό, τοπικό δίκτυο. Βασικός περιοριστικός παράγοντας για την ταυτόχρονη λειτουργία δύο ή περισσοτέρων VMs στον ίδιο φυσικό υπολογιστή, είναι η επάρκεια συνολικής μνήμης RAM και πυρήνων του κεντρικού επεξεργαστή στο φυσικό υπολογιστή.

Εναλλακτικά, τα VMs μπορούν να συμμετέχουν στο ίδιο το τοπικό δίκτυο που συμμετέχει κι ο αληθινός υπολογιστής. Σε αυτήν την περίπτωση, βρίσκονται στην ίδια τοπολογία με τον υπολογιστή που τα φιλοξενεί, πίσω από το CPE, με αποτέλεσμα να εμφανίζονται ως υπολογιστές του οικιακού, τοπικού δικτύου. Σε ένα τέτοιο σενάριο μπορούν να μοιράζονται την ίδια (ενσύρματη ή ασύρματη) κάρτα δικτύου του αληθινού υπολογιστή ή μερικές μηχανές να μοιράζονται μία κάρτα κι άλλες κάποια δεύτερη.

Το φυσικό μηχάνημα στο οποίο η εικονικοποίηση λαμβάνει χώρα αποκαλείται **μηχάνημα οικοδεσπότης (host machine)**, ενώ το εικονικό μηχάνημα αποκαλείται **μηχάνημα επισκέπτης (guest machine)**. Οι λέξεις host και guest χρησιμοποιούνται για να διαχωρίσουν το λογισμικό που εκτελείται στο φυσικό μηχάνημα από το λογισμικό που εκτελείται στο εικονικό μηχάνημα. Το εξειδικευμένο λογισμικό (software/firmware) που δημιουργεί την εικονική μηχανή στον host καλείται **hypervisor** ή **Διαχειριστής Εικονικών Μηχανών (Virtual Machine Manager – VMM)**. [1]



1.1 Αρχιτεκτονική Εικονικοποίησης [2]

1.1.1 Σκοπός και Στόχοι της Εικονικοποίησης

Η εικονικοποίηση θεωρείται μέρος μιας συνολικότερης τάσης στον κλάδο της πληροφορικής, η οποία στρέφεται προς την **αυτόνομη και κατά παραγγελία χρήση υπολογιστικών πόρων (autonomic and utility computing)**.

Η αυτόνομη χρήση αναφέρεται στην αυτοδιαχείριση κατανεμημένων υπολογιστικών πόρων η οποία προσαρμόζεται στις μεταβαλλόμενες συνθήκες, ενώ αποκρύπτει την εσωτερική πολυπλοκότητα από τους χειριστές και τους χρήστες των πόρων. Το σύστημα λαμβάνει μόνο του αποφάσεις, χρησιμοποιώντας υψηλού επιπέδου πολιτικές, ενώ συνεχώς ελέγχει και βελτιώνει την τρέχουσα κατάστασή του. [3]

Η κατά παραγγελία χρήση είναι ένα μοντέλο παροχής υπηρεσιών στο οποίο ο πάροχος διαθέτει υπολογιστικούς πόρους και διαχείριση υποδομών στον πελάτη και τον χρεώνει ανάλογα με τη χρήση κι όχι με ένα σταθερό πάγιο. Το μοντέλο αυτό αποσκοπεί στη μεγιστοποίηση της αποτελεσματικής χρήσης των πόρων και στην ελαχιστοποίηση του σχετικού κόστους. Επιπλέον, προσφέρει το πλεονέκτημα χαμηλού έως και μηδενικού αρχικού κόστους απόκτησης υπολογιστικού εξοπλισμού (hardware). [4]

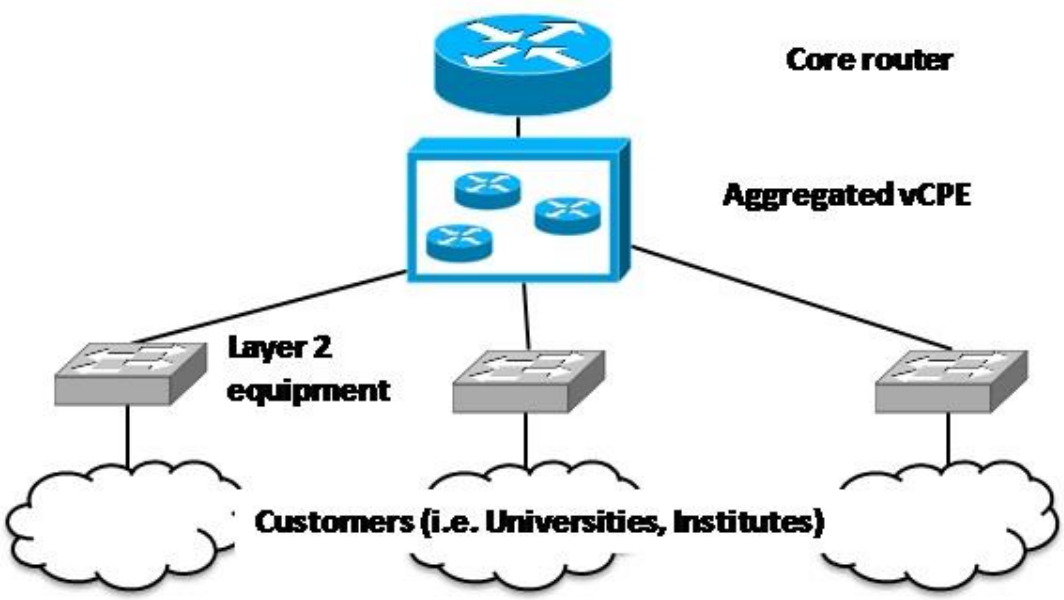
Στα πλαίσια της διπλωματικής εργασίας, ο **συνδρομητικός εξοπλισμός (Customer-premises equipment – CPE)** από την πλευρά του δικτύου της επιχείρησης αντικαθίσταται από την εικονικοποιημένη εκδοχή του (vCPE). Επιπροσθέτως, το ρόλο του μεσολαβητή μεταξύ των δυο επιχειρησιακών δικτύων στο VPN θα αναλάβει ένας **εικονικός δρομολογητής (vRouter)**, ο οποίος φιλοξενείται σε ένα εικονικό μηχάνημα. Η αρχιτεκτονική που θα ακολουθηθεί για τη σχεδίαση του δικτύου είναι γνωστή ως **Εικονικοποίηση Δικτυακών Λειτουργιών (Network Functions Virtualization)**.

1.1.2 Εικονικός Συνδρομητικός Εξοπλισμός (virtual Customer-Premises Equipment – vCPE)

Ο εικονικός συνδρομητικός εξοπλισμός είναι ένας τρόπος παροχής δικτυακών υπηρεσιών όπως δρομολόγηση, ασφάλεια τείχους προστασίας (firewall) και συνδεσιμότητα εικονικού ιδιωτικού δικτύου (VPN) σε επιχειρήσεις χρησιμοποιώντας λογισμικό (software) αντί για εξειδικευμένο εξοπλισμό (hardware). Χάρη στα vCPE, η παροχή υπηρεσιών απλοποιείται και επιταχύνεται δραματικά. Οι πάροχοι έχουν τη δυνατότητα να διαχειρίζονται και να παραμετροποιούν τις συσκευές απομακρυσμένα, επιτρέποντας στου πελάτες να παραγγέλνουν νέες υπηρεσίες ή να τροποποιούν τις υπάρχουσες κατά απαίτηση.

Ο κλασσικός **συνδρομητικός εξοπλισμός (Customer-Premises Equipment – CPE)** αποτελείται από εξειδικευμένες συσκευές που ανήκουν στους παρόχους υπηρεσιών, οι οποίες τοποθετούνται στα γραφεία μιας εταιρείας. Έπειτα, οι πάροχοι είναι υπεύθυνοι για την αποστολή τεχνικών δικτύου στα γραφεία για την παράδοση και παραμετροποίηση του εξοπλισμού, που σημαίνει ότι η εγκατάσταση νέων υπηρεσιών ενδέχεται να είναι χρονοβόρα και ακριβή.

Σε αντίθεση, ο εικονικός συνδρομητικός εξοπλισμός ή συνδρομητικός εξοπλισμός υπολογιστικού νέφους (cloud CPE), μεταφέρει την ευφυΐα από τις συσκευές αυτές σε νέο επίπεδο αφαίρεσης βασισμένο σε λογισμικό, το οποίο εκτελείται σε ένα απομακρυσμένο κέντρο δεδομένων. Το λειτουργικό εκτελείται σε μια απλή, οικονομική συσκευή που βρίσκεται στα γραφεία της επιχείρησης. Αυτό το μοντέλο επιτρέπει επίσης το συνδυασμό εξειδικευμένων, ξεχωριστών συσκευών σε μια συσκευή γενικών καθηκόντων, τόσο για διευκόλυνση όσο και για ελάττωση κόστους. [5]

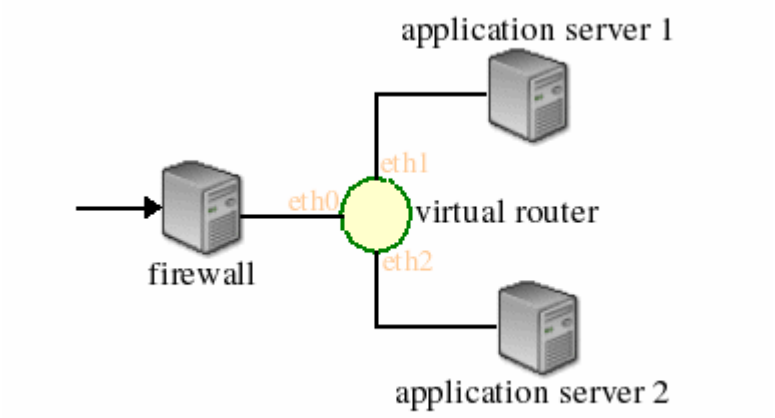


1.2 Εικονικός Συνδρομητικός Εξοπλισμός (vCPE)

1.1.3 Εικονικός δρομολογητής (Virtual Router – vRouter)

Ο εικονικός δρομολογητής προσομοιώνει πλήρως τη λειτουργία ενός κανονικού δρομολογητή, με τη διαφορά ότι πρόκειται για λογισμικό το οποίο εκτελείται σε εικονικό μηχάνημα κάποιου υπολογιστικού συστήματος (πχ διακομιστή) του παρόχου υπηρεσιών.

Η εικονική δρομολόγηση είναι μια μορφή Εικονικοποίησης Δικτυακών Λειτουργιών (Network Functions Virtualization – NFV), στην οποία οι λειτουργίες πολλών δικτυακών συσκευών μετατρέπονται σε λογισμικά τα οποία εκτελούνται σε ένα υπολογιστικό σύστημα. Το μοντέλο αυτό έχει πλεονεκτήματα ελάττωσης κόστους απόκτησης φυσικού εξοπλισμού (hardware) αλλά και χρήσης των ίδιων μηχανημάτων για το χειρισμό ποικίλων δικτυακών λειτουργιών αντί να απαιτείται ξεχωριστός εξοπλισμός για καθεμία από αυτές.



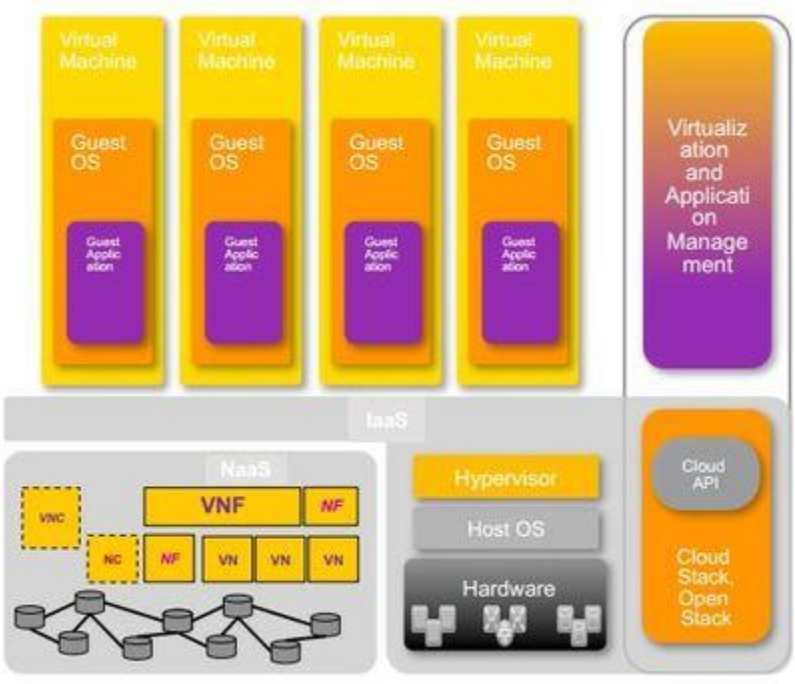
1.3 Εικονικός Δρομολογητής (vRouter)

1.2 Εικονικοποίηση Δικτυακών Λειτουργιών (Network Functions Virtualization – NFV)

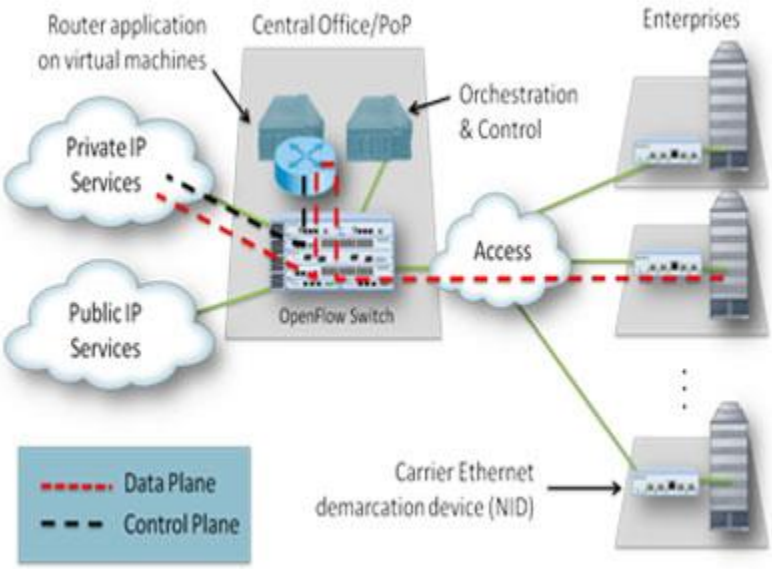
Κάθε δικτυακή λειτουργία απαιτεί εν γένει για τη διεκπεραίωσή της ειδικό εξοπλισμό. Η εικονικοποίηση δικτυακών λειτουργιών είναι μια αρχιτεκτονική δικτύου που εικονικοποιεί λειτουργίες κόμβων δικτύου σε δομικά στοιχεία, τα οποία συνδέονται μεταξύ τους για να δημιουργήσουν υπηρεσίες τηλεπικοινωνιών.

Στην NFV, μια **εικονικοποιημένη δικτυακή λειτουργία (Virtualized Network Function – VNF)** μπορεί να αποτελείται από ένα ή περισσότερα εικονικά μηχανήματα, τα οποία εκτελούν διαφορετικά λογισμικά και διεργασίες εντός ενός διακομιστή, μεταγωγέα, συσκευής αποθήκευσης ή υπολογιστικού νέφους.

Ουσιαστικά, η NFV αποσκοπεί στην αποδέσμευση των δικτυακών λειτουργιών, όπως δρομολογητής, τείχος προστασίας, εξισορροπητής φορτίου, μετάφραση διευθύνσεων δικτύου (Network Address Translation – NAT), σύστημα ονομάτων περιοχών (Domain Name System – DNS) κι άλλους εξειδικευμένους διακομιστές δικτύου, από τον ειδικά κατασκευασμένο για αυτές εξοπλισμό. Στη συνέχεια, τις υλοποιεί ως τμήματα λογισμικού σε πλήρως εικονικοποιημένες δικτυακές υποδομές, χρησιμοποιώντας γνωστές τεχνολογίες και τεχνικές εικονικοποίησης, ώστε να βελτιώσει και να επιτρέψει την καινοτομία στην παροχή και τη διαχείριση υπηρεσιών δικτύου. [6]



1.4 Εικονικοποιημένη Δικτυακή Λειτουργία (Virtual Network Fuction – VNF) [7]



1.5 Υπηρεσία Δρομολόγησης υλοποιημένη με NFV [8]

1.2.1 Συστατικά στοιχεία του NFV

Το πλαίσιο λειτουργίας της NFV αποτελείται από 3 βασικά μέρη:

- Τις **εικονικοποιημένες δικτυακές λειτουργίες (Virtualized Network Function – VNFs)**, δηλαδή υλοποιήσεις διαφόρων δικτυακών λειτουργιών σε λογισμικό, οι οποίες μπορούν να αναπτυχθούν πάνω σε υποδομή εικονικοποίησης δικτυακών λειτουργιών (NFVI)

- Την **υποδομή εικονικοποίησης δικτυακών λειτουργιών (Network Functions Virtualization Infrastructure – NFVI)**, που είναι το σύνολο του υλικού και λογισμικού που συνθέτει το περιβάλλον όπου αναπτύσσονται οι VNFs. Η NFVI μπορεί να εκτείνεται σε περισσότερες από μια τοποθεσίες ενώ το δίκτυο που παρέχει συνδεσιμότητα ανάμεσα σε αυτές θεωρείται μέρος της υποδομής.

- Το **πλαίσιο διαχείρισης και ενορχήστρωσης της εικονικοποίησης δικτυακών λειτουργιών (NFV – Management and Orchestration/MANO)**, που είναι το σύνολο όλων των λειτουργικών μονάδων, των δεδομένων που αυτές χρησιμοποιούν, των σημείων αναφοράς και των διεπαφών μέσω των οποίων αυτές ανταλλάσσουν πληροφορίες, με σκοπό τη διαχείριση και την ενορχήστρωση των VNFs και της NFVI.

Η βασική μονάδα τόσο για την NFVI όσο και για το NFV-MANO είναι η **πλατφόρμα NFV**. Από την πλευρά της NFVI, αυτή αποτελείται από τους εικονικούς και φυσικούς πόρους επεξεργασίας και αποθήκευσης και από το λογισμικό εικονικοποίησης. Από την πλευρά του NFV-MANO, η πλατφόρμα NFV αποτελείται από τους **διαχειριστές (managers)** των VNFs και της NFVI και το λογισμικό εικονικοποίησης, το οποίο εκτελείται εντός του ελεγκτή. Η πλατφόρμα NFV υλοποιεί λειτουργίες που την καθιστούν κατάλληλη για χρήση σε τηλεπικοινωνιακό περιβάλλον, όπως διαχείριση και επίβλεψη των διάφορων δομικών στοιχείων της πλατφόρμας, ανάνηψη από σφάλματα κι αποτελεσματική ασφάλεια, στοιχεία απαραίτητα για το δίκτυο δημόσιας χρήσης. [6]

1.2.2 Συνδυασμός Υπηρεσιών (Service Chaining) και Ενορχήστρωση

Είναι ευρέως γνωστό στις εταιρίες τηλεπικοινωνιών ότι η παροχή νέων υπηρεσιών στα δίκτυά τους είναι μια χρονοβόρα διαδικασία που απαιτεί πολλούς πόρους, τόσο δικτυακούς όσο και οικονομικούς. Μέχρι τώρα, η εισαγωγή νέων υπηρεσιών διαρκούσε μήνες ως κι ολόκληρα έτη κι η υλοποίησή τους γινόταν με φυσικές συσκευές που αποκτιόνταν κατόπιν παραγγελίας. Αυτό συμβαίνει γιατί η διαδικασία αυτή χειρίζεται την ανάγκη της παροχής όλων των διαφορετικών συστατικών της παρεχόμενης υπηρεσίας, τη ρύθμισή τους ανάλογα με την τοπολογία του δικτύου και τέλος το συνδυασμό τους.

Ένας πάροχος υπηρεσιών που υιοθετεί τη σχεδίαση NFV υλοποιεί μια ή περισσότερες εικονικοποιημένες δικτυακές λειτουργίες (VNFs). Μια VNF από μόνη της δεν παρέχει αυτομάτως ένα χρησιμοποιήσιμο προϊόν ή υπηρεσία στον πελάτη. Για την κατασκευή πιο σύνθετων υπηρεσιών εφαρμόζεται η έννοια του **συνδυασμού επιμέρους υπηρεσιών (service chaining)**, όπου πολλαπλές VNFs χρησιμοποιούνται σε ακολουθία για την παράδοση μιας πολυπλοκότερης υπηρεσίας.

Με την εμφάνιση του υπολογιστικού νέφους (cloud), το οποίο είναι δυναμικής φύσεως και προσφέρει νέες, ευέλικτες δυνατότητες, είναι φυσικό για τις εταιρίες τηλεπικοινωνιών να στραφούν προς αυτήν την κατεύθυνση, την οποία κι εισήγαγε πρώτη στη βιομηχανία η σχεδίαση NFV. Αρχικά, διαχωρίζονται οι εφαρμογές οι οποίες στο παρελθόν εκτελούνταν εντός ενός “μαύρου κουτιού”, σε διαφορετικά VMs ή και κέντρα δεδομένων ακόμη. Στη συνέχεια, δημιουργούνται εικονικές αντιστοιχίσεις από ό,τι προηγουμένως ήταν φυσικές συσκευές. Καθιστώντας πλέον δυνατή την εκτέλεση όλων των ανωτέρω σε απλά hardware εξαρτήματα του εμπορίου, εμφανίστηκε ένας εντελώς νέος κόσμος εξοικονόμησης κόστους. Το επόμενο βήμα προς αυτήν την κατεύθυνση ήταν η εικονικοποίηση του συνδυασμού των επιμέρους υπηρεσιών που στο παρελθόν αποτελούνταν αποκλειστικά από φυσικά μηχανήματα.

Η παραπάνω διαδικασία καλείται πλέον δυναμικός συνδυασμός υπηρεσιών (dynamic service chaining). Πρόκειται ουσιαστικά για τη δυνατότητα του συνδυασμού των επιμέρους VNFs για την παροχή μιας ολοκληρωμένης υπηρεσίας, διασφαλίζοντας τελικά πως στην κίνηση που κυκλοφορεί στο δίκτυο μέσω της νέας υπηρεσίας επιβάλλεται η ανάλογη πολιτική, συνήθως στο χρόνο εκτέλεσης. Επιπλέον, πρέπει να διασφαλίζεται ότι η ροή μέσα από τις επιμέρους υπηρεσίες γίνεται με τη σωστή σειρά. Αυτό είναι ιδιαίτερα περίπλοκο, καθώς οι ρυθμίσεις των VMs εξαρτώνται τη λειτουργία του κάθε VM εντός της ολοκληρωμένης υπηρεσίας. Για αυτό το λόγο, αν αλλαχθεί η σειρά με την οποία συνδυάζονται οι επιμέρους υπηρεσίες, απαιτείται κι αλλαγή της ρύθμισης κάθε συστατικού στοιχείου.

Επειδή όμως πάντα υπάρχουν εξαρτήσεις, είναι απαραίτητη η ύπαρξη ενός **ενορχηστρωτή (orchestrator)**, ο οποίος επιβλέπει την εξέλιξη της διαδικασίας του chaining, είναι ενήμερος για όλες τις επιμέρους υπηρεσίες που συμμετέχουν και είναι σε θέση να τις συνδυάσει με ευφυή τρόπο αλλά και με τη σειρά με την οποία πρέπει να ρυθμιστούν. Αυτό είναι το σημείο όπου η ενορχήστρωση της σχεδίασης NFV έχει απλουστεύσει εξαιρετικά τη διαδικασία του chaining, δίνοντας τη δυνατότητα στις εταιρείες τηλεπικοινωνιών να συνδυάζουν τις υπηρεσίες με δυναμικό τρόπο. Ως αποτέλεσμα, έχει επιφέρει στη βιομηχανία μειώσεις κόστους κι απαιτήσεις υπολογιστικών πόρων άνευ προηγουμένου.

Η **διαδικασία της ενορχήστρωσης** είναι μια εξαιρετικά σημαντική πτυχή της NFV υλοποίησης. Για την κατασκευή επεκτάσιμων υπηρεσιών υψηλής αξιοπιστίας, η αρχιτεκτονική NFV απαιτεί από το δίκτυο να είναι σε θέση να δημιουργήσει VNF στιγμιότυπα, να τα παρακολουθεί, να τα επισκευάζει και κυρίως να χρεώνει τον πελάτη ανάλογα με τις υπηρεσίες που χρησιμοποιήθηκαν. Τα χαρακτηριστικά αυτά μεταφέρονται στο επίπεδο ενορχήστρωσης έτσι ώστε να εξασφαλίζεται υψηλό επίπεδο διαθεσιμότητας κι ασφάλειας αλλά και χαμηλό κόστος λειτουργίας και συντήρησης. Σημαντικό είναι το επίπεδο ενορχήστρωσης να είναι ικανό να διαχειρίζεται τις VNFs ανεξάρτητα από την υποκείμενη τεχνολογία εντός κάθε μιας VNF.

Η ενορχήστρωση της NFV έχει καταστήσει δυνατό το δυναμικό chaining σε οποιαδήποτε υποδομή, είτε πρόκειται για κέντρα δεδομένων, υπολογιστικά νέφη ή περιβάλλοντα που αποτελούνται από ετερογενείς τεχνολογίες. Έτσι, η παροχή, ρύθμιση και εισαγωγή υπηρεσιών σε ένα δίκτυο μπορεί πλέον να ολοκληρωθεί άψογα οπουδήποτε κι οποιαδήποτε στιγμή. [9]

1.2.3 Κατανεμημένη NFV

Η αρχική αντίληψη όσον αφορά την NFV στόχευε στη συγκέντρωση των δυνατοτήτων εικονικοποίησης στα κέντρα δεδομένων (data centers). Η προσέγγιση αυτή είναι αποτελεσματική σε αρκετές αλλά όχι όλες τις περιπτώσεις. Για το λόγο αυτό, η αρχιτεκτονική NFV έχει σχεδιαστεί με τέτοιο τρόπο, ώστε να μπορεί να χειρίζεται με τη μεγαλύτερη δυνατή ευελιξία τις VNFs ως προς την τοποθεσία των φυσικών μηχανημάτων που τις φιλοξενούν.

Ιδανικά λοιπόν, οι VNFs πρέπει να τοποθετούνται σε σημεία που εξασφαλίζουν τη μεγαλύτερη αποτελεσματικότητα και το χαμηλότερο κόστος για το δίκτυο. Αυτό σημαίνει ότι ο πάροχος υπηρεσιών θα πρέπει να είναι σε θέση να εφαρμόσει την αρχιτεκτονική NFV σε όλες τις δυνατές τοποθεσίες, δηλαδή το κέντρο δεδομένων, τους κόμβους δικτύου και το συνδρομητικό εξοπλισμό του πελάτη. Αυτή η προσέγγιση ονομάζεται **κατανεμημένη NFV (Distributed NFV)** και είναι κυρίαρχη λόγω της ευελιξίας που παρέχει στο διαμοιρασμό των VNFs. [6]

1.2.4 Πλεονεκτήματα κατάτμησης της NFV

Κατά το σχεδιασμό και την υλοποίηση του λογισμικού που παρέχει τις VNFs, οι σχεδιαστές μπορεί να διαμερίσουν το λογισμικό αυτό σε **τμήματα VNF (VNF Components – VNFCs)** και να τα συνδυάσουν έπειτα σε μια ή περισσότερες εικονικές μηχανές. Οι VNFs αποτελούνται από ένα ή περισσότερα VNFCs και θεωρείται, χωρίς βλάβη της γενικότητας, ότι σε κάθε VNFC αντιστοιχεί και ένα εικονικό μηχάνημα.

Τα VNFCs πρέπει εν γένει να έχουν τη δυνατότητα να επεκταθούν οριζόντια και κατακόρυφα. Αναδιανέμοντας εικονικούς επεξεργαστές (CPUs) στο εικονικό μηχάνημα που φιλοξενεί ένα VNFC, το επίπεδο διαχείρισης δικτύου μπορεί να **επεκτείνει** το VNFC αυτό **κατακόρυφα (scale up/down)** ώστε να ρυθμίσει την απόδοση και τις προσδοκίες επεκτασιμότητας για ένα σύστημα ή πλατφόρμα στο επιθυμητό επίπεδο.

Ομοίως, το επίπεδο διαχείρισης δικτύου μπορεί να **επεκτείνει οριζόντια** ένα VNFC **(scale out/in)**. Τροποποιώντας τον αριθμό των αντιγράφων (εικονικών μηχανών) του VNFC στις διάφορες πλατφόρμες, το VNFC μπορεί να ανταποκριθεί στην απόδοση και τις προδιαγραφές της αρχιτεκτονικής, χωρίς να θέτει σε κίνδυνο τη σταθερότητα της λειτουργίας των υπόλοιπων VNFCs. [6]

1.2.5 Επιπτώσεις της NFV στη Βιομηχανία της Παροχής Δικτυακών Υπηρεσιών

Η NFV έχει αποδειχθεί πως είναι δημοφιλείς αρχιτεκτονική από πολύ νωρίς. Οι άμεσες εφαρμογές της NFV είναι πολυάριθμες, όπως εικονικοποίηση σταθμών βάσης ασύρματης επικοινωνίας, υπηρεσιών πλατφόρμας (Platform as a Service – PaaS) δικτύων παροχής υπηρεσιών και σταθερών υπηρεσιών πρόσβασης οικιακών χώρων. Τα δυνητικά προνόμια της NFV αναμένονται να είναι σημαντικά. Η εφαρμογή εικονικοποίησης δικτυακών λειτουργιών σε συσκευές γενικών καθηκόντων αναμένεται να μειώσει τα έξοδα απόκτησης και συντήρησης εξειδικευμένου εξοπλισμού, καθώς και έναρξη χρήσης νέων προϊόντων κι υπηρεσιών από τους πελάτες.

Για την επίτευξη των αναμενόμενων προνομίων της εικονικοποίησης, οι πάροχοι δικτυακού εξοπλισμού βελτιώνουν την τεχνολογία εικονικοποίησης. Αυτό θα τους επιτρέψει να ενσωματώσουν τα χαρακτηριστικά της NFV που προαναφέρθηκαν, τα οποία είναι απαραίτητα για την επίτευξη υψηλής διαθεσιμότητας, επεκτασιμότητας, απόδοσης και δυνατοτήτων αποτελεσματικής διαχείρισης ενός δικτύου.

Για την ελαχιστοποίηση του συνολικού κόστους ιδιοκτησίας, τα χαρακτηριστικά αυτά πρέπει να ενσωματωθούν όσο πιο αποτελεσματικά γίνεται. Αυτό προϋποθέτει ότι η υλοποίηση NFV κάνει αποτελεσματική χρήση των πλεονάζουσων πόρων του συστήματος για την επίτευξη της υψηλότερης δυνατής διαθεσιμότητας χωρίς να διακινδυνεύεται η προβλεψιμότητα της απόδοσης.

Η πλατφόρμα NFV αποτελεί το θεμέλιο για την υλοποίηση αποτελεσματικών δικτυακών λύσεων για τους πελάτες. Πρόκειται για μια πλατφόρμα λογισμικού που εκτελείται σε πολυπύρηνο υπολογιστικό εξοπλισμό και κατασκευάζεται με τη χρήση λογισμικού ανοιχτού κώδικα που ενσωματώνει τα χαρακτηριστικά της NFV που προαναφέρθηκαν. Το λογισμικό της πλατφόρμας NFV είναι υπεύθυνο για τη δυναμική ανάθεση των VNFs εξαιτίας των βλαβών κι αλλαγών στο φόρτο της διακινούμενης πληροφορίας και για αυτό παίζει σημαντικό ρόλο στην επίτευξη υψηλής διαθεσιμότητας.

Η εικονικοποίηση αλλάζει επίσης τον τρόπο με τον οποίο διευκρινίζεται η διαθεσιμότητα, η οποία μετράται κι επιτυγχάνεται στις υλοποιήσεις NFv για πελάτες. Καθώς οι VNFs αντικαθιστούν τον παραδοσιακό εξειδικευμένο δικτυακό εξοπλισμό, παρατηρείται μια στροφή από τη διαθεσιμότητα με βάση τον εξοπλισμό σε μια προσέγγιση που βασίζεται στην στρωματική παροχή υπηρεσιών από άκρη-σε-άκρη (end-to-end). Η εικονικοποίηση δικτυακών λειτουργιών αφαιρεί την εξάρτηση της παροχής υπηρεσιών από εξειδικευμένο εξοπλισμό και για αυτό η διαθεσιμότητα ορίζεται πλέον από τη διαθεσιμότητα των VNF υπηρεσιών.

Επειδή η τεχνολογία NFV μπορεί να εικονικοποιήσει ένα ευρύ φάσμα από τύπους δικτυακών λειτουργιών, καθεμία με τις δικές τις προσδοκίες διαθεσιμότητας υπηρεσιών, οι πλατφόρμες NFV πρέπει να υποστηρίζουν ένα υψηλό επίπεδο ανοχής σε σφάλματα. Αυτή η ευελιξία επιτρέπει στους παρόχους υπηρεσιών να βελτιώνουν τις NFV υλοποιήσεις τους ώστε να ανταποκρίνονται σε οποιαδήποτε απαίτηση διαθεσιμότητας VNF. [6]

1.2.6 Διαχείριση και Ενορχήστρωση της NFV (NFV – Management and Orchestration/MANO)

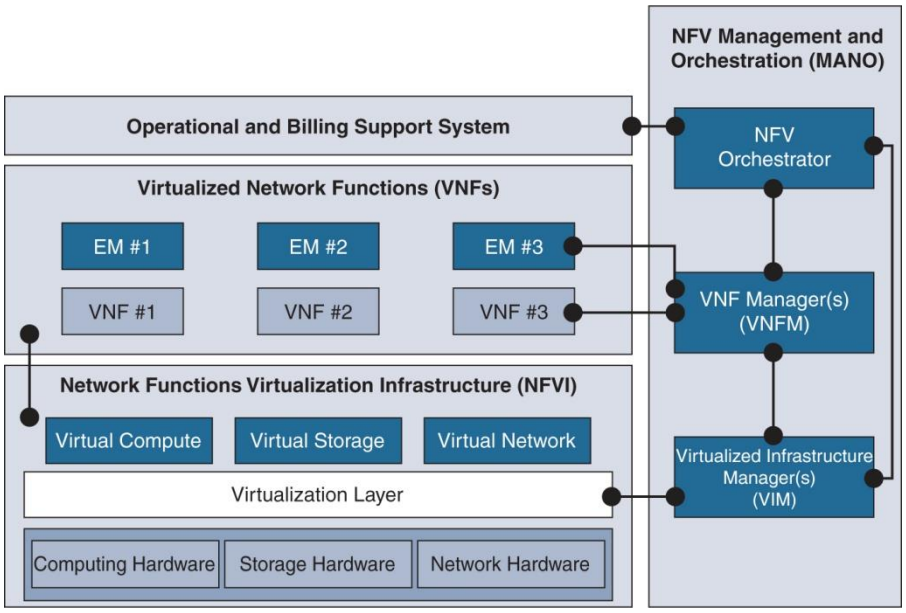
Η NFV-MANO, όπως περιγράφηκε νωρίτερα, αναλαμβάνει την ενορχήστρωση και διαχείριση του κύκλου ζωής των πόρων του συστήματος (εξοπλισμού και λογισμικού). Αυτοί περιλαμβάνουν τους υπολογιστικούς, δικτυακούς και αποθηκευτικούς πόρους καθώς και τις εικονικές μηχανές του συστήματος, συμπεριλαμβανομένων των VNFs. Η NFV-MANO διακρίνεται σε τρία λειτουργικά τμήματα:

- **Ενορχηστρωτής Εικονικοποίησης Δικτυακών Λειτουργιών (NFV Orchestrator)**: Είναι υπεύθυνος για την εκκίνηση νέων υπηρεσιών δικτύου και το συνδυασμό VNFs, τη διαχείριση του κύκλου ζωής των δικτυακών υπηρεσιών, την καθολική διαχείριση των πόρων του συστήματος και την επικύρωση και έγκριση των αιτημάτων για πόρους από την NFVI.

- **Διαχειριστής Εικονικοποιημένων Δικτυακών Λειτουργιών (VNF Manager)**: Επιβλέπει τη διαχείριση του κύκλου ζωής των εικονικών μηχανημάτων που εκτελούν την κάθε VNF, συντονίζει και προσαρμόζει τις ρυθμίσεις και την αναφορά γεγονότων μεταξύ της NFVI και του λογισμικού διαχείρισης δικτύου.

- **Διαχειριστής Εικονοποιημένης Υποδομής (Virtualized Infrastructure Manager – VIM)**: Ελέγχει και διαχειρίζεται τους υπολογιστικούς, αποθηκευτικούς και δικτυακούς πόρους της NFVI.

Για τη διασφάλιση της σωστής και αποτελεσματικής λειτουργίας της NFV-MANO αρχιτεκτονικής, πρέπει στα υπάρχοντα συστήματα να ενσωματωθούν **διεπαφές προγραμματισμού εφαρμογών (Application Programming Interfaces – APIs)**. Το στρώμα MANO λειτουργεί με έτοιμες APIs για τις περισσότερες VNFs και δίνει στους χρήστες τη δύναμη να επιλέξουν από υπάρχοντες πόρους της NFVI για να τους αναπτύξουν στην πλατφόρμα ή το σύστημά τους. [10]



1.6 Αρχιτεκτονικό Πλαίσιο Λειτουργίας της NFV [11]

1.3 Προγραμματιζόμενη Δικτύωση (Software-Defined Networking)

Η **προγραμματιζόμενη δικτύωση (Software-Defined Networking/SDN)** είναι μια προσέγγιση στη δικτύωση υπολογιστών που επιτρέπει στους διαχειριστές δικτύου να εκκινούν, ελέγχουν, αλλάζουν και να διαχειρίζονται τη συμπεριφορά του δικτύου δυναμικά μέσω διεπαφών ανοιχτού λογισμικού, προσφέροντας ένα επίπεδο αφαίρεσης στη λειτουργικότητα των χαμηλότερων επιπέδων.

Η SDN έχει σχεδιαστεί με σκοπό να αντιμετωπίσει το γεγονός ότι η στατική αρχιτεκτονική των παραδοσιακών δικτύων, δεν υποστηρίζει τις δυναμικές, επεκτάσιμες υπολογιστικές και αποθηκευτικές ανάγκες των σύγχρονων υπολογιστικών περιβαλλόντων όπως τα κέντρα δικτύου (data centers). Αυτό επιτυγχάνεται **διαχωρίζοντας** το σύστημα που λαμβάνει τις αποφάσεις σχετικά με τον παραλήπτη της διακινούμενης πληροφορίας (**επίπεδο ελέγχου – control plane**) από τα υποκείμενα συστήματα που προωθούν την κίνηση στον επιλεγμένο προορισμό (**επίπεδο δεδομένων – data plane**).

Στα πλαίσια αυτής της διπλωματικής θα χρησιμοποιηθεί το **πρωτόκολλο OpenFlow**,το οποίο συσχετίζεται άμεσα με την SDN. Χρησιμοποιείται για απομακρυσμένη επικοινωνία με το επίπεδο δικτύου, για το σκοπό της εξακρίβωσης του μονοπατιού των πακέτων της διακινούμενης πληροφορίας μεταξύ των μεταγωγέων του δικτύου. [12]

1.3.1 Γενική Ιδέα της Προγραμματιζόμενης Δικτύωσης

Η SDN είναι μια δυναμική, διαχειρίσιμη, οικονομική και προσαρμόσιμη αρχιτεκτονική που αρμόζει για τις σημερινές εφαρμογές, οι οποίες χαρακτηρίζονται από δυναμικότητα κι υψηλού επιπέδου εύρος ζώνης. Οι SDN αρχιτεκτονικές αποσυνδέουν τις λειτουργίες ελέγχου και προώθησης, επιτρέποντας στον έλεγχο του δικτύου να είναι άμεσα προγραμματιζόμενος. Επιπλέον, η υποκείμενη υποδομή βρίσκεται σε διαφορετικό αφαιρετικό επίπεδο από τις εφαρμογές και τις υπηρεσίες δικτύου.

Χαρακτηριστικά της SDN αρχιτεκτονικής:

- **Άμεσα Προγραμματιζόμενη**: Ο έλεγχος του δικτύου προγραμματίζεται άμεσα επειδή είναι αποσυνδεδεμένος από τις λειτουργίες προώθησης.

- **Ευέλικτη**: Η μετακίνηση του ελέγχου σε διαφορετικό αφαιρετικό επίπεδο από την προώθηση επιτρέπει στους διαχειριστές να προσαρμόζουν δυναμικά τη ροή κίνησης του δικτύου ώστε να ανταποκρίνεται στις ανάγκες που διαρκώς αλλάζουν.

- **Κεντρικοποιημένη διαχείριση**: Η ευφυΐα του δικτύου συγκεντρώνεται σε **ελεγκτές SDN (SDN controllers)** βασισμένους σε λογισμικό, οι οποίοι διατηρούν μια καθολική οπτική του δικτύου, το οποίο εμφανίζεται στις εφαρμογές και στο σύστημα επιβολής πολιτικών ως ένας απλός, λογικός μεταγωγέας.

- **Διαμορφώνεται προγραμματιστικά**: Η SDN επιτρέπει στους διαχειριστές δικτύου να επεξεργάζονται, διαχειρίζονται, διασφαλίζουν και να βελτιώνουν τους πόρους του δικτύου ταχύτατα μέσω δυναμικών, αυτόματων SDN προγραμμάτων, τα οποία συντάσσουν οι ίδιοι επειδή τα προγράμματα δεν εξαρτώνται από ιδιόκτητο λογισμικό.

- **Βασίζεται σε λογισμικό ανοιχτού κώδικα και δεν εξαρτάται από συγκεκριμένους διανομείς δικτυακού εξοπλισμού**: Η SDN απλοποιεί το σχεδιασμό και τη λειτουργία του δικτύου, επειδή οι εντολές παρέχονται από τους ελεγκτές SDN, αντί από πολλαπλές συσκευές και πολλαπλά πρωτόκολλα που διαφέρουν μεταξύ κατασκευαστών. [12]

1.3.2 Πρωτόκολλο OpenFlow

Πρόκειται για το πρωτόκολλο επικοινωνίας που παρέχει πρόσβαση στο επίπεδο προώθησης (forwarding plane) ενός μεταγωγέα (switch) ή δρομολογητή (router) στο δίκτυο.

Το OpenFlow επιτρέπει στον ελεγκτή ενός δικτύου να προσδιορίσει το μονοπάτι των πακέτων που ταξιδεύουν εντός ενός δικτύου μεταγωγέων. Ο ελεγκτής ξεχωρίζει από τους μεταγωγείς. Ο διαχωρισμός αυτός του επιπέδου ελέγχου από το επίπεδο προώθησης επιτρέπει πιο εκλεπτυσμένη διαχείριση της δικτυακής κίνησης απ' ό,τι δύναται με τη χρήση λιστών ελέγχου πρόσβασης (Access Control Lists – ACL) και πρωτοκόλλων δρομολόγησης.

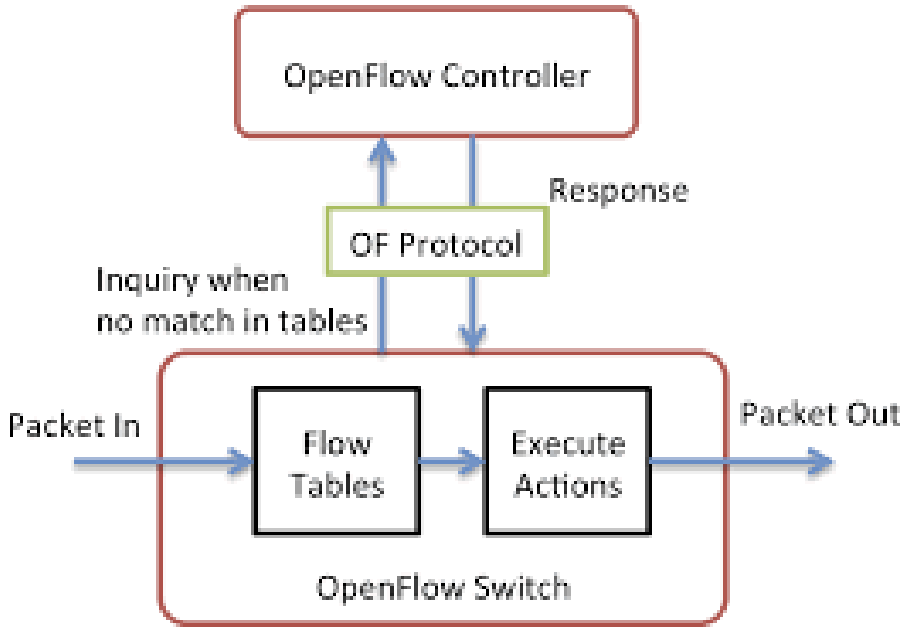
Επιπλέον, το OpenFlow επιτρέπει τη χρήση μεταγωγέων από διαφορετικούς κατασκευαστές – συνήθως ο καθένας με τις δικές του διεπαφές και γλώσσες προγραμματισμού – για απομακρυσμένη διαχείριση χρησιμοποιώντας ένα μοναδικό, ανοιχτό πρωτόκολλο. Οι εφευρέτες του πρωτοκόλλου θεωρούν το OpenFlow ως αναπόσπαστο κομμάτι για την ομαλή λειτουργία των προγραμματιζόμενων δικτύων (Software-Defined Networks – SDNs).

Το Openflow επιτρέπει την απομακρυσμένη διαχείριση των πινάκων δρομολόγησης πακέτων των μεταγωγέων του Επιπέδου 3 (Layer 3) με την προσθήκη, τροποποίηση και αφαίρεση κανόνων ταιριάσματος (matching rules) και δράσεων (actions). Με αυτόν τον τρόπο, οι αποφάσεις δρομολόγησης μπορούν να γίνουν περιοδικά ή στο παρασκήνιο από τον ελεγκτή και να μεταφραστούν σε κανόνες και δράσεις με επεξεργάσιμη διάρκεια ζωής.

Έπειτα, εφαρμόζονται στον **πίνακα ροών (flow table)** ενός μεταγωγέα, αφήνοντας την δρομολόγηση των ταιριασμένων πακέτων να τρέχει για το χρονικό διάστημα που είναι ενεργοί οι κανόνες. Αναλυτική περιγραφή για το πώς επεξεργαζόμαστε τις ροές των μεταγωγέων μέσω του ελεγκτή θα ακολουθήσει σε επόμενο κεφάλαιο.

Τα πακέτα που δεν έχουν ταιριάξει με κάποιο κανόνα δρομολόγησης από το μεταγωγέα προωθούνται στον ελεγκτή. Ο ελεγκτής μπορεί να αποφασίσει να τροποποιήσει τους κανόνες στον υπάρχοντα πίνακα ροών ενός ή περισσότερων μεταγωγέων ή να δημιουργήσει νέους κανόνες για να εμποδίσει τη ροή κίνησης μεταξύ μεταγωγέα και ελεγκτή. Μπορεί ακόμη και να αποφασίσει να προωθήσει κίνηση ο ίδιος, δεδομένου ότι έχει ενημερώσει το μεταγωγέα να προωθεί ολόκληρα πακέτα αντί μονάχα την επικεφαλίδα τους.

Το πρωτόκολλο OpenFlow είναι δομημένο πάνω από το πρωτόκολλο ελέγχου μεταφοράς (Transmission Control Protocol – TCP) και προτρέπει τη χρήση του πρωτοκόλλου κρυπτογράφησης Transport Layer Security (TLS). [13]



1.7 Αρχιτεκτονική πρωτοκόλλου OpenFlow [14]

1.3.3 Αρχιτεκτονική Διάρθρωση της SDN

1.3.3.1 Εφαρμογή SDN (SDN Application)

Οι εφαρμογές SDN είναι προγράμματα, τα οποία μεταφέρουν με άμεσο και σαφή τρόπο τις δικτυακές τους απαιτήσεις και την επιθυμητή τους συμπεριφορά στο δίκτυο, στον ελεγκτή SDN μέσω μιας **διεπαφής Northbound (Northbound Interface – NBI)**. Επιπροσθέτως, μπορούν να κατασκευάσουν ένα νέο επίπεδο αφαίρεσης, μέσα από το οποίο αντιλαμβάνονται το δίκτυο για δική τους εσωτερική λήψη αποφάσεων.

Μια εφαρμογή SDN αποτελείται από την αρχιτεκτονική της λογική (Application Logic) και ένα η περισσότερους οδηγούς NBI (NBI drivers). Οι εφαρμογές SDN ενδέχεται να φανερώσουν ένα νέο επίπεδο αφαίρεσης του ελέγχου δικτύου, προσφέροντας με αυτόν τον τρόπο μια ή περισσότερες υψηλότερου επιπέδου NBIs μέσω των αντίστοιχών τους βοηθών NBI (NBI agents).

1.3.3.2 Ελεγκτής SDN (SDN Controller)

Ο ελεγκτής SDN είναι μια **λογική κεντρικοποιημένη οντότητα**, επικεφαλής της μετάφρασης και μεταφοράς των απαιτήσεων από το επίπεδο εφαρμογής SDN μέχρι τα μονοπάτια δεδομένων SDN. Επιπλέον, παρέχει στις εφαρμογές SDN ένα νέο επίπεδο αφαίρεσης του δικτύου, το οποίο μπορεί να περιλαμβάνει στατιστικά και γεγονότα. Ένας ελεγκτής SDN αποτελείται από έναν ή περισσότερους βοηθούς NBI, την κεντρική λογική SDN (SDN Control Logic) και τον οδηγό της διεπαφής επιπέδου ελέγχου – επιπέδου δεδομένων (Control to Data-Plane Interface driver – CDPI driver).

1.3.3.3 Μονοπάτι Δεδομένων SDN (SDN Datapath)

Το μονοπάτι δεδομένων SDN είναι μια λογική συσκευή δικτύου, η οποία εξασφαλίζει την ορατότητα και τον αδιαμφισβήτητο έλεγχο των ανακοινωμένων δυνατοτήτων προώθησης και επεξεργασίας δεδομένων. Η λογική απεικόνιση μπορεί να περιλαμβάνει όλους τους φυσικούς πόρους του υποστρώματος ή ένα υποσύνολό τους.

Ένα μονοπάτι δεδομένων SDN περιλαμβάνει έναν βοηθό CDPI, ένα σύνολο από μια ή περισσότερες μηχανές προώθησης κίνησης (traffic forwarding engines) και καμία ή περισσότερες λειτουργίες επεξεργασίας. Αυτές οι μηχανές κι οι λειτουργίες μπορεί να περιλαμβάνουν απλή προώθηση μεταξύ των εξωτερικών διεπαφών του μονοπατιού δεδομένων, των λειτουργιών εσωτερικής επεξεργασίας της κίνησης ή των λειτουργιών τερματισμού.

Ένα ή περισσότερα μονοπάτια δεδομένων SDN μπορεί να περιέχονται σε ένα **φυσικό στοιχείο δικτύου (physical network element)**, δηλαδή έναν ενσωματωμένο φυσικό συνδυασμό επικοινωνιακών πόρων, οι οποίοι διαχειρίζονται σα μονάδα. Ένα μονοπάτι δεδομένων SDN μπορεί επίσης να ορισθεί ταυτόχρονα σε πολλαπλά φυσικά δικτυακά στοιχεία.

1.3.3.4 SDN Διεπαφή Επιπέδου Ελέγχου – Επιπέδου Δεδομένων (SDN Control to Data-Plane Interface/CDPI)

Η SDN CDPI είναι η διεπαφή που ορίζεται μεταξύ ενός ελεγκτή SDN κι ενός μονοπατιού δεδομένων SDN, η οποία παρέχει προγραμματιστικό έλεγχο όλων των λειτουργιών προώθησης, ανακοίνωση δυνατοτήτων, αναφορά στατιστικών και ενημέρωση γεγονότων. Η σημαντικότερη αξία της αρχιτεκτονικής SDN έγκειται στη δυνατότητα υλοποίησης της CDPI με τέτοιο τρόπο, ώστε να μην εξαρτάται από συγκεκριμένο κατασκευαστή και να είναι σε θέση να προσφέρει διαλειτουργικότητα.

1.3.3.5 Northbound Διεπαφές SDN (SDN Northbound Interfaces – NBI)

Οι SDN NBIs είναι διεπαφές μεταξύ των εφαρμογών και ελεγκτών SDN. Τυπικά παρέχουν ένα αφαιρετικό επίπεδο ορατότητας του δικτύου, εκφράζοντας άμεσα τη συμπεριφορά και τις απαιτήσεις του. Αυτό μπορεί να συμβεί σε οποιοδήποτε επίπεδο της αφαίρεσης και σε διαφορετικά σύνολα λειτουργικότητας. Όμοια με τη CDPI, οι NBIs πρέπει να κατασκευάζονται με τέτοιο τρόπο, ώστε να εξασφαλίζουν διαλειτουργικότητα και να μην εξαρτώνται από τους κατασκευαστές δικτυακού εξοπλισμού. [12]

1.3.4 Επίπεδο ελέγχου SDN (SDN Control Plane)

Η υλοποίηση του επιπέδου ελέγχου SDN μπορεί να ακολουθήσει μια κεντρικοποιημένη, αποκεντρικοποιημένη ή ιεραρχική σχεδίαση. Οι αρχικές προτάσεις για το επίπεδο ελέγχου SDN επικεντρώνονταν σε μια κεντρικοποιημένη υλοποίηση, όπου μια μοναδική οντότητα ελέγχου έχει καθολική εποπτεία του δικτύου. Παρόλο που αυτή η σχεδίαση απλοποιεί τη λογική του ελέγχου, περιορίζει την επεκτασιμότητα, καθώς το μέγεθος κι οι ανάγκες του δικτύου αυξάνονται. Για να ξεπερασθούν αυτοί οι περιορισμοί, οι προσεγγίσεις που έχουν προταθεί διαχωρίζονται σε δυο κατηγορίες, **ιεραρχικές και πλήρως κατανεμημένες προσεγγίσεις**.

- Στην **ιεραρχική προσέγγιση**, οι κατανεμημένοι ελεγκτές λειτουργούν με μια καταμερισμένη οπτική του δικτύου, όπου οι αποφάσεις που απαιτούν γνώση καθ’ όλη την έκταση του δικτύου λαμβάνονται από ένα λογικά κεντρικοποιημένο ελεγκτή.

- Στην **κατανεμημένη προσέγγιση**, οι ελεγκτές λειτουργούν ο καθένας με την τοπική του οπτική του δικτύου ή μπορούν να ανταλλάσσουν μηνύματα συγχρονισμού για να ενισχύσουν τη γνώση τους. Οι κατανεμημένες υλοποιήσεις είναι καταλληλότερες για την υποστήριξη προσαρμοστικών εφαρμογών SDN.

Η επιλογή του αριθμού των οντοτήτων ελέγχου και της τοποθέτησής τους στο δίκτυο είναι από τα σημαντικότερα ζητήματα κατά τη σχεδίαση ενός κατανεμημένου επιπέδου ελέγχου SDN. Μια σημαντική παράμετρος που πρέπει να ληφθεί υπόψιν κατά τη λήψη της ανωτέρω απόφασης είναι η καθυστέρηση διάδοσης μεταξύ των ελεγκτών και των συσκευών δικτύου, ειδικά στο πλαίσιο των μεγάλων δικτύων. Άλλοι στόχοι που έχουν εξετασθεί περιλαμβάνουν την αξιοπιστία του μονοπατιού ελέγχου, την ανοχή στα σφάλματα και τις απαιτήσεις των εφαρμογών. [12]

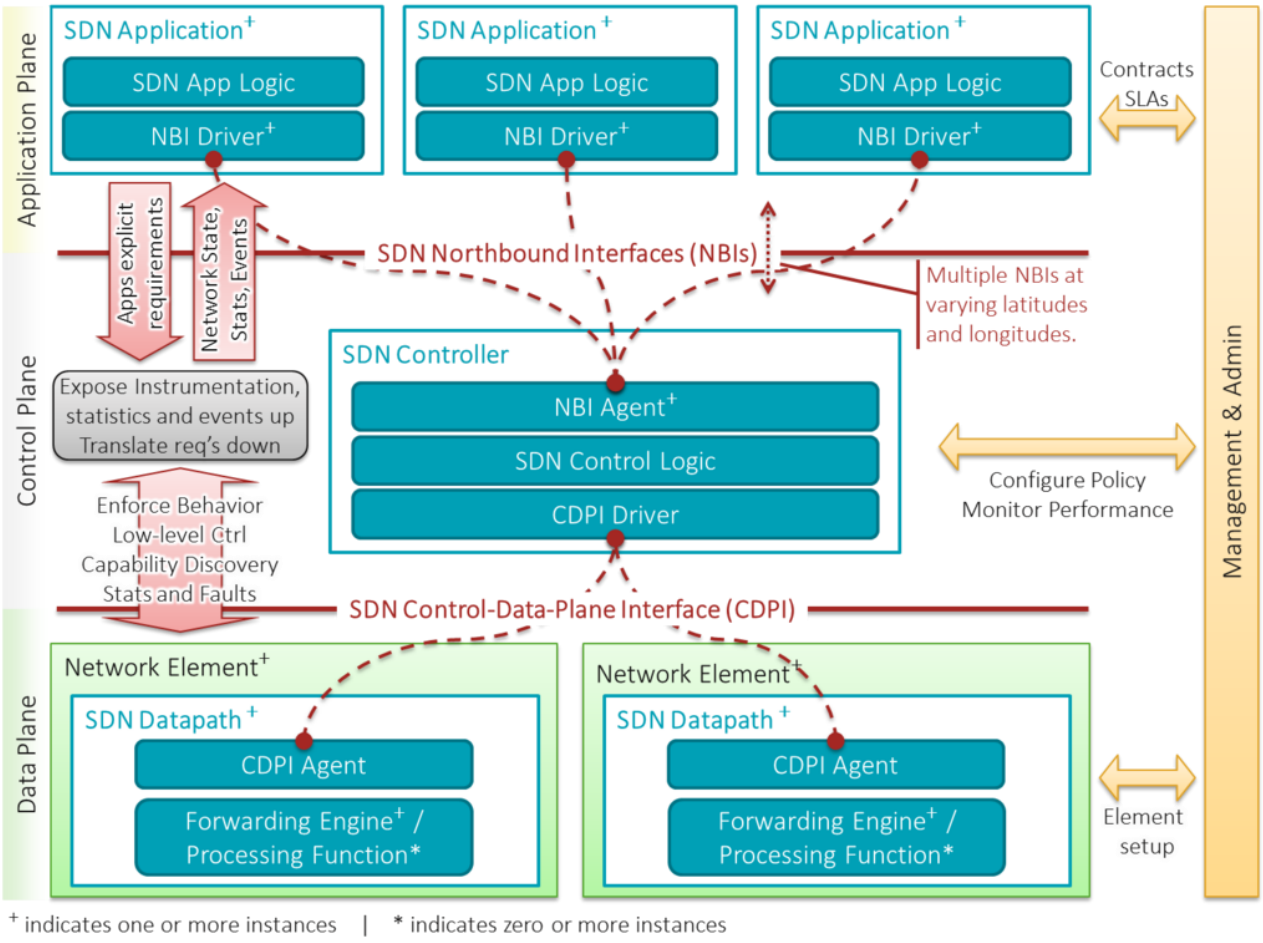
1.3.5 Επίπεδο Δεδομένων SDN και Προώθηση Ροών (SDN Data Plane – Flow Forwarding)

Το πρωτόκολλο OpenFlow χρησιμοποιεί πίνακες για τη δρομολόγηση πακέτων ακολουθίας, ή αλλιώς **ροές (flows)**. Αν μια ροή φτάσει σε ένα μεταγωγέα, τότε εκτελείται αναζήτηση στον πίνακα ροών. Στην περίπτωση που δεν πραγματοποιηθεί **ταίριασμα (matching)** με τη ροή που μόλις κατέφτασε, αποστέλλεται προς τον ελεγκτή αίτημα για περαιτέρω οδηγίες. Η διαχείριση των αιτημάτων αυτών πραγματοποιείται με τρεις διαφορετικούς τρόπους:

- Στην **αντιδραστική λειτουργία (reactive mode)**, ο ελεγκτής λαμβάνει δράση μετά την άφιξη των αιτημάτων αυτών, όπου αν χρειάζεται, δημιουργεί κι εγκαθιστά έναν **κανόνα προώθησης (forwarding rule)** στον πίνακα ροών για το αντίστοιχο πακέτο.

- Στην **προληπτική λειτουργία (proactive mode)**, ο ελεγκτής δημιουργεί εξ αρχής εγγραφές στον πίνακα ροών για όλα τα πιθανά ταιριάσματα του συγκεκριμένου μεταγωγέα. Αυτή η λειτουργία μπορεί να συγκριθεί με τις εγγραφές ενός συνηθισμένου πίνακα δρομολόγησης, όπου όλες οι στατικές εγγραφές εγκαθίστανται εκ των προτέρων. Στη συνέχεια, δεν αποστέλλεται προς τον ελεγκτή αίτημα για περαιτέρω οδηγίες, αφού όλες οι εισερχόμενες ροές θα βρουν εγγραφή με την οποία να ταιριάζουν. Ένα μείζον πλεονέκτημα της προληπτικής λειτουργίας είναι ότι όλα τα πακέτα προωθούνται χωρίς προστιθέμενη καθυστέρηση.

- Στην **υβριδική λειτουργία (hybrid mode)**, αξιοποιείται η ευελιξία της αντιδραστικής λειτουργίας για ένα σύνολο της κίνησης και η χαμηλή καθυστέρηση προώθησης της προληπτικής λειτουργίας για την υπόλοιπη κίνηση. [12]



1.8 Αρχιτεκτονική της Προγραμματιζόμενης Δικτύωσης (SDN Architecture) [12]

1.4 Σχέση μεταξύ NFV και SDN

Η SDN είναι μια έννοια που σχετίζεται άμεσα με την NFV αλλά διαφέρουν σε συγκεκριμένα πεδία.

Στην ουσία, η SDN είναι μια προσέγγιση για την κατασκευή εξοπλισμού και λογισμικού διαχείρισης δικτυακών δεδομένων, η οποία διαχωρίζει στοιχεία από αυτά τα συστήματα, δημιουργώντας ένα νέο αφαιρετικό επίπεδο δικτύου. Αυτό επιτυγχάνεται διαχωρίζοντας το επίπεδο ελέγχου από το επίπεδο δεδομένων, με τέτοιο τρόπο ώστε το επίπεδο ελέγχου να βρίσκεται συγκεντρωμένο σε ένα κεντρικό σημείο και τα στοιχεία προώθησης να παραμένουν κατανεμημένα.

Το επίπεδο ελέγχου αλληλεπιδρά τόσο με τα Northbound όσο και με τα Southbound APIs. Στη Northbound κατεύθυνση, το επίπεδο ελέγχου παρέχει μια κοινή οπτική αφαίρεσης του δικτύου στις εφαρμογές και τα προγράμματα υψηλότερου επιπέδου χρησιμοποιώντας τις APIs. Στη Southbound κατεύθυνση, το επίπεδο ελέγχου προγραμματίζει τη συμπεριφορά προώθησης του επιπέδου δεδομένων, χρησιμοποιώντας τα APIs που βρίσκονται στις διάφορες συσκευές του εξοπλισμού του φυσικού δικτύου, που είναι κατανεμημένες σε όλο το μήκος του δικτύου.

Συνεπώς, η NFV δεν εξαρτάται από την SDN ή έννοιές της. Είναι πλήρως δυνατό να υλοποιηθεί μια VNF ως μια ανεξάρτητη οντότητα, χρησιμοποιώντας τις υπάρχουσες τεχνολογίες και τεχνικές δικτύωσης και ενορχήστρωσης. Ωστόσο, υπάρχουν αδιαμφισβήτητα προνόμια στην αξιοποίηση εννοιών της SDN για την υλοποίηση και διαχείριση μιας NFVI, συγκεκριμένα παρατηρώντας τη MANO των VNFs. Για αυτό το λόγο, πλατφόρμες που προέρχονται από πολλαπλούς κατασκευαστές, ορίζονται με τέτοιο τρόπο ώστε να ενσωματώνουν την SDN και την NFV σε ενοποιημένα οικοσυστήματα.

Μια NFVI χρειάζεται ένα κεντρικό σύστημα MANO, το οποίο λαμβάνει αιτήματα χειριστών που σχετίζονται με μια VNF και τα μεταφράζει στην κατάλληλη ρύθμιση επεξεργασίας, αποθήκευσης και δικτύου που απαιτείται για να λειτουργήσει η VNF. Μόλις είναι σε θέση να λειτουργήσει η VNF, πρέπει ενδεχομένως να παρακολουθούνται τα επίπεδα χωρητικότητας και χρηστικότητας και να προσαρμόζονται όπου κρίνεται απαραίτητο.

Όλες αυτές οι λειτουργίες μπορούν να επιτευχθούν με τη χρήση εννοιών SDN. Η NFV μπορεί να θεωρηθεί ως μια από τις πρωταρχικές περιπτώσεις χρήσης της SDN στα περιβάλλοντα παροχής υπηρεσιών. Είναι επίσης εμφανές, πως πολλές περιπτώσεις χρήσης SDN μπορούν να ενσωματώσουν έννοιες που παρουσιάστηκαν αρχικά με την εμφάνιση της NFV. Το χαρακτηριστικότερο παράδειγμα είναι ο έλεγχος μιας κατανεμημένης λειτουργίας προώθησης από έναν κεντρικοποιημένο ελεγκτή, ο οποίος θα μπορούσε στην πραγματικότητα να εικονικοποιηθεί πάνω στον υπάρχοντα εξοπλισμό επεξεργασίας ή δρομολόγησης. Η **τεχνολογία NFV είναι το παρόν, ενώ η τεχνολογία SDN αποτελεί το μέλλον.** [6]

2. Εικονικό Ιδιωτικό Δίκτυο (Virtual Private Network – VPN)

2.1 Επισκόπηση – Ορισμός

Η εξάπλωση της δικτυωμένης οικονομίας έχει επιφέρει ουσιαστικές αλλαγές στον τρόπο λειτουργίας των επιχειρήσεων. Ο ανταγωνισμός σε πολλές βιομηχανίες έχει οδηγήσει τόσο σε συμμαχίες αλλά και σε συνεταιρισμούς μεταξύ τους. Αυτές οι εξελίξεις έχουν μεν αυξήσει την παραγωγικότητα και την κερδοφορία πολλών επιχειρήσεων, έχουν όμως ταυτόχρονα δημιουργήσει νέες απαιτήσεις για τις επιχειρήσεις αυτές.

Ένα δίκτυο που επικεντρώνεται στο να συνδέει απλά σταθερά σημεία των συνεργαζόμενων επιχειρήσεων δεν είναι πλέον αρκετό για πολλές επιχειρήσεις. Οι απομακρυσμένοι χρήστες του δικτύου των επιχειρήσεων, όπως για παράδειγμα οι εξωτερικοί συνεργάτες, απαιτούν πλέον πρόσβαση στους πόρους του δικτύου της επιχείρησης. Για παράδειγμα, θα πρέπει ένας εξωτερικός συνεργάτης μιας επιχείρησης να μπορεί να συνδεθεί στο τοπικό της δίκτυο από οπουδήποτε, μέσω του φορητού του υπολογιστή. Το κλασικό Δίκτυο Ευρείας Ζώνης (WAN) πρέπει λοιπόν να επεκταθεί ώστε να συμπεριλάβει και αυτού του τύπου τους εργαζόμενους.

Ταυτόχρονα, οι επιχειρήσεις με περισσότερα από ένα παραρτήματα (καταστήματα, γραφεία) πολύ συχνά αντιμετωπίζουν προβλήματα επικοινωνίας ή λειτουργίας που απορρέουν από τη γεωγραφική απόσταση που τα χωρίζει. Συνεπώς, πολλές επιχειρήσεις στρέφονται προς τα **Εικονικά Ιδιωτικά Δίκτυα (Virtual Private Networks – VPNs)** για να συμπληρώσουν την υπάρχουσα WAN υποδομή τους και να επιλύσουν προβλήματα επικοινωνίας, οργάνωσης, διαχείρισης και κατανομής πληροφοριών σε όλα τα τμήματα ή τα υποκαταστήματα τους, όπου κι αν βρίσκονται.

Το VPN είναι ένα **δίκτυο εικονικών ζεύξεων** ανεπτυγμένο σε μία υπάρχουσα δικτυακή υποδομή (κατά κύριο λόγο το δημόσιο δίκτυο – Internet), με τη ιδιότητα ότι έχει την ίδια **ασφάλεια, διαχείριση και πολιτική** σε όλο το μήκος του σαν να επρόκειτο για ιδιωτικό δίκτυο. Στην πραγματικότητα, δίνει τη δυνατότητα στους χρήστες να ανταλλάσσουν δεδομένα μέσω κοινόχρηστων ή δημόσιων δικτύων, με τον ίδιο τρόπο ως αν τα υπολογιστικά τους συστήματα ήταν άμεσα συνδεδεμένα στο ιδιωτικό δίκτυο.

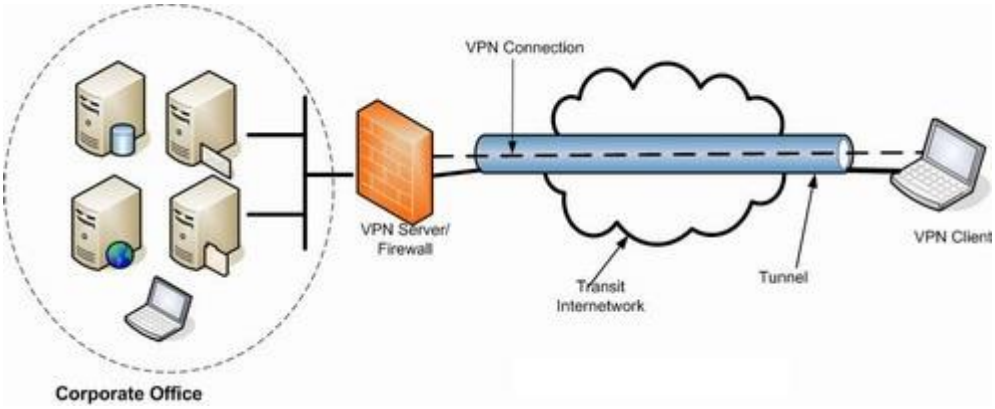
2.2 Είδη VPN - Απαιτήσεις

Οι απαιτήσεις των VPNs δεν είναι άλλες από αυτές των WANs: υποστήριξη πολλαπλών πρωτοκόλλων, υψηλή αξιοπιστία και εκτεταμένη διαβάθμιση. Ένα VPN μπορεί να αξιοποιήσει τις πιο γνωστές τεχνολογίες μεταφοράς που υπάρχουν σήμερα: το δημόσιο Internet (κατά κύριο λόγο), τα δίκτυα διαφόρων παρόχων υπηρεσιών όπως επίσης και τις μεταξύ τους ζεύξεις.

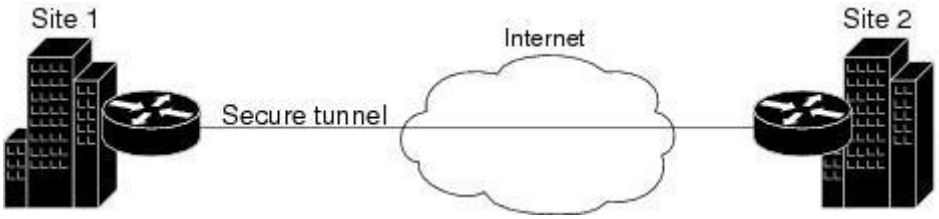
Συγκεκριμένα, εδώ κι αρκετά χρόνια, τα VPNs έχουν πάψει να χτίζονται πάνω σε τηλεφωνικά δίκτυα και επεκτείνονται πάνω από IP δίκτυα, εξαιτίας της σημαντικής μείωσης του κόστους αλλά και του αυξημένου εύρους ζώνης. Τις αλλαγές αυτές έφερε η ανακάλυψη τεχνολογιών όπως της **Ψηφιακής Συνδρομητικής Γραμμής (Digital Subscriber Line – DSL)** και των **δικτύων οπτικών ινών**.

Ένα VPN μπορεί να είναι **απομακρυσμένης πρόσβασης (remote-access)**, δηλαδή να συνδέει έναν υπολογιστή σε ένα δίκτυο ή **δικτύου-προς-δίκτυο (site-to-site)**, δηλαδή να συνδέει δυο δίκτυα μεταξύ τους. Στα πλαίσια μιας εταιρείας, ένα remote-access VPN επιτρέπει στους υπαλλήλους να έχουν πρόσβαση στο εσωτερικό δίκτυο της εταιρίας (intranet) από το σπίτι τους ή όταν βρίσκονται εκτός του χώρου εργασίας γενικότερα. Ένα site-to-site VPN επιτρέπει στους υπαλλήλους που εργάζονται σε απομακρυσμένα μεταξύ τους γραφεία μιας εταιρείας να μοιράζονται ένα συνεκτικό εικονικό δίκτυο.

Τέλος, ένα VPN μπορεί να χρησιμοποιηθεί για να συνδέσει δυο ίδιου τύπου δίκτυα μεταξύ τους πάνω από ένα διαφορετικού τύπου δίκτυο. Για παράδειγμα, δυο IPv6 δίκτυα μπορεί να συνδέονται μεταξύ τους μέσω VPN πάνω από ένα ενδιάμεσο IPv4 δίκτυο.



2.1 Mobile VPN [15]



2.2 Site-to-Site VPN [16]

2.3 Αρχιτεκτονική του VPN – Τοπολογία

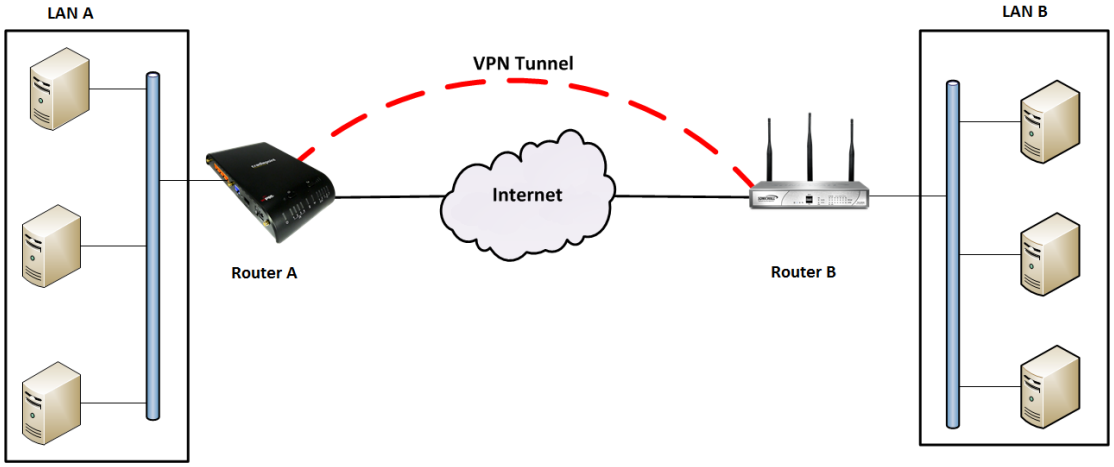
Στα πλαίσια αυτής της διπλωματικής θα γίνει υλοποίηση **υπηρεσίας VPN στο νέφος (Cloud VPN)**, η οποία θα συνδέει τα vCPEs (Raspberry Pi's) δυο ανεξάρτητων επιχειρησιακών δικτύων. Το ρόλο του διαμεσολαβητή θα αναλάβει ο εικονικός δρομολογητής που θα τοποθετηθεί ανάμεσά τους. Τέλος, η επικοινωνία μεταξύ vCPEs και vRouter επιτυγχάνεται με τη δημιουργία δυο **διόδων Γενικής Ενθυλάκωσης Δρομολόγησης (Generic Routing Encapsulation – GRE tunnels)**.

2.3.1 Τεχνική Εγκαθίδρυσης Διόδου (Tunneling)

Τα Εικονικά Ιδιωτικά Δίκτυα με πρωτόκολλα επιπέδου 2 αναπτύχθηκαν κυρίως ως Δίκτυα Απομακρυσμένης Πρόσβασης: με άλλα λόγια, επιτρέπουν σε έναν απομακρυσμένο χρήστη να συνδεθεί μέσω μίας γραμμής Internet στο εσωτερικό δίκτυο μίας εταιρίας. Οι δίοδοι (tunnels) μπορούν να δημιουργηθούν είτε ανάμεσα σε ένα ζεύγος δρομολογητών (router-to-router) είτε μεταξύ δύο τερματικών κόμβων (host-to-host).

Η εγκαθίδρυση διόδου μπορεί να υλοποιείται σε μία τοπολογία σημείου-προς-σημείο ή σημείου-προς-πολλά σημεία: η σημείου-προς-σημείου έχει λιγότερο διαχειριστικό φορτίο, από την άποψη της εγκαθίδρυσης και της συντήρησης.
Η εγκαθίδρυση «διόδου» (tunneling) είναι η τεχνική ενθυλάκωσης ενός ολόκληρου πακέτου/πλαισίου δεδομένων σε ένα πακέτο/πλαίσιο διαφορετικού πρωτοκόλλου. Η επικεφαλίδα του tunneling πρωτοκόλλου προσαρτάται στο αρχικό πακέτο ενώ η μεταφορά/μετάδοση πραγματοποιείται με χρήση του νέου πρωτοκόλλου.

Έτσι, όταν ένα τέτοιο πακέτο δρομολογείται προς τον κόμβο προορισμού, διατρέχει το δίκτυο μέσα από λογικό μονοπάτι, το οποίο αναφέρεται ως δίοδος (tunnel). Όταν ο κόμβος προορισμού λάβει το πακέτο, το μετατρέπει στην αρχική του μορφή. Σημειώνεται ότι η τεχνολογία tunneling μπορεί να αναπτυχθεί στο δεύτερο ή στο τρίτο επίπεδο του μοντέλου OSI.
Ένα από τα πλεονεκτήματα του tunneling είναι ότι τα διασυνδεδεμένα υποδίκτυα VPN δεν απαιτούν μοναδικές διευθύνσεις δικτύου. Αυτό είναι σημαντικό όταν η πλειοψηφία των οργανισμών σήμερα χρησιμοποιεί ιδιωτικές διευθύνσεις. Επίσης ένα VPN με τη χρήση του tunneling μπορεί να δημιουργηθεί με ή χωρίς τη γνώση του παρόχου δικτύου και θα μπορούσε να «περάσει» μέσα από διαδοχικούς παρόχους δικτύου.



2.3 Εγκαθίδρυση διόδου (tunneling) [17]

2.3.2 Πρωτόκολλο GRE

Ο μηχανισμός της Cisco GRE (Generic Routing Encapsulation) χρησιμοποιείται για tunneling ανάμεσα σε δρομολογητές πηγής και προορισμού (router-to-router). Το πρωτόκολλο GRE μπορεί να ενθυλακώσει ένα ευρύ φάσμα πρωτοκόλλων επιπέδου δικτύου εντός εικονικών ζεύξεων σημείο-προς-σημείο που εκτείνονται πάνω από ένα IP δίκτυο.

Τα GRE tunnels παρέχουν ένα ειδικό μονοπάτι κατά μήκος μίας διαμοιραζόμενης υποδομής WAN που δεν ανήκει μόνο σε έναν χρήστη-πελάτη (π.χ. Internet) και ενθυλακώνουν την κίνηση με νέες επικεφαλίδες πακέτου για να εξασφαλίσουν τη διανομή σε ένα συγκεκριμένο προορισμό.

Ένα GRE tunnel διαμορφώνεται ανάμεσα στο δρομολογητή πηγής και το δρομολογητή προορισμού. Τα πακέτα που πρόκειται να προωθηθούν κατά μήκος της διόδου ενθυλακώνονται με μία επικεφαλίδα GRE, μεταφέρονται κατά μήκος της διόδου και στο τέλος της αφαιρείται η επικεφαλίδα GRE.

2.4 Δίοδος GRE [18]

2.4 Παροχή υπηρεσιών VPN στο νέφος (VPN Service in Cloud)

Η ραγδαία εξέλιξη της τεχνολογίας τα τελευταία χρόνια έχει επιτρέψει σε πλήθος νέων τεχνοτροπιών να κάνουν την εμφάνισή τους. Όπως έχει περιγραφεί και σε προηγούμενο κεφάλαιο, μια τέτοια τεχνολογία είναι το **υπολογιστικό νέφος (cloud computing)**.

Το VPN που υλοποιείται στα πλαίσια αυτής της διπλωματικής εργασίας πρόκειται για μια υπηρεσία που παρέχεται εξ ολοκλήρου σε ένα τέτοιο νέφος. Συγκεκριμένα, ο **εικονικός δρομολογητής (vRouter)** ανάμεσα στις δυο διόδους GRE που θα κατασκευασθούν για την ροή της πληροφορίας μεταξύ των δικτύων, φιλοξενείται σε **εικονικό μηχάνημα (Virtual Machine)** που βρίσκεται εντός του υπολογιστικού νέφους του παρόχου με τον οποίο συνεργάζεται η εκάστοτε εταιρεία.

Η **παροχή υπηρεσίας VPN μέσω του νέφους (Cloud VPN – VPN as a Service/VPNaaS)** είναι μια λύση για επιχειρήσεις, η οποία δίνει τη δυνατότητα στους απομακρυσμένους υπαλλήλους να αποκτούν πρόσβαση με ασφάλεια στο ιδιωτικό εταιρικό δίκτυο μέσω του διαδικτύου.

Το Cloud VPN χρησιμοποιεί ένα δίκτυο που βασίζεται σε υποδομή νέφους για να προσφέρει υπηρεσίες VPN. Παρέχει στους τελικούς χρήστες πρόσβαση στο VPN σε παγκόσμιο επίπεδο μέσω μιας πλατφόρμας νέφους που εκτείνεται πάνω από το δημόσιο δίκτυο (Internet). Για οικονομικούς κυρίως λόγους, πλήθος επιχειρήσεων επιλέγουν τη μίσθωση της παροχής υπηρεσίας του VPN σε ένα πάροχο Cloud.

Ο στόχος πίσω από το Cloud VPN είναι η παροχή του ίδιου επιπέδου ασφάλειας και παγκόσμιας πρόσβασης σε υπηρεσίες VPN χωρίς την ανάγκη για ύπαρξη υποδομής VPN σε επίπεδο εξοπλισμού (hardware) από πλευράς πελατών – επιχειρήσεων. Ο τελικός χρήστης συνδέεται στο Cloud VPN μέσω της ιστοσελίδας του παρόχου ή μιας εφαρμογής (desktop/mobile application).

Παρομοίως, η κοστολόγηση της υπηρεσίας του παρεχόμενου Cloud VPN είναι διαφορετική από αυτή του κλασσικού VPN, καθώς χρεώνει τον πελάτη ανάλογα με τη χρήση που αυτός κάνει ή με ένα σταθερό πάγιο ανά προσυμφωνημένα χρονικά διαστήματα. Οι χρήστες χρεώνονται ανάλογα με το πλήθος εξοπλισμού, αποθηκευτικού χώρου, δικτύου, λογισμικού κι άλλων πόρων που καταλήγει να χρησιμοποιεί.

Προνόμια παρόχου:

Υπηρεσία VPN με χρήση τεχνολογίας εικονικοποίησης (virtualization), που προσφέρει εύκολη ενσωμάτωση για άλλες υπηρεσίες, όπως παροχή αποθηκευτικού χώρου
Ταχεία υλοποίηση σε ιδιωτικό cloud
Υψηλή επεκτασιμότητα
Κεντρικοποιημένη διαχείριση του VPN, με ευελιξία στις τροποποιήσεις
Υψηλό επίπεδο συμβατότητας
Ασφάλεια της υπηρεσίας

Προνόμια πελάτη:

Δεν απαιτείται επένδυση σε εξοπλισμό, λογισμικό και ειδικό προσωπικό συντήρησης
Γρήγορη εφαρμογή των απαιτούμενων υπηρεσιών απομακρυσμένης πρόσβασης
Χαμηλό μηνιαίο κόστος
Υψηλή χρηστικότητα
Ασφάλεια συσκευών μέσω ενσωματωμένου προσωπικού τείχους προστασίας (firewall) στην εφαρμογή VPN του πελάτη
Υποστήριξη όλων των γνωστών λειτουργικών συστημάτων [19]

3. Αρχιτεκτονική Συστήματος

Προτού γίνει ανάλυση των δομικών στοιχείων της τοπολογίας του δικτύου που θα αναπτυχθεί για την εργασία, παρουσιάζεται η συνολική αρχιτεκτονική του συστήματος, ώστε να υπάρχει γενική εποπτεία του σκοπού κάθε τέτοιου στοιχείου. Αναλυτική περιγραφή της υλοποίησης κάθε μιας τεχνολογίας ξεχωριστά ακολουθεί στις επόμενες ενότητες.

Σκοπός της εργασίας είναι η δημιουργία ενός **εικονικού ιδιωτικού δικτύου (Virtual Private Network – VPN)** μεταξύ δυο ανεξάρτητων τοπικών δικτύων (Local Area Networks – LANs). Η τεχνολογία που επιλέχθηκε για την υλοποίηση του VPN είναι **κανάλια πρωτοκόλλου Generic Routing Encapsulation (GRE Protocol)**, τα οποία επικοινωνούν μεταξύ τους μέσω ενός **εικονικού δρομολογητή (vRouter)**.

Ταυτόχρονα, πρέπει η επικοινωνία μεταξύ αυτών των LANs να μην είναι δυνατή μέσω του διαδικτύου (Internet). Αυτό επιτυγχάνεται με τη χρήση της τεχνολογίας **Προγραμματιζόμενης Δικτύωσης (Software Defined Networking – SDN)**. Κεντρικό ρόλο επιτελεί ο **SDN Ελεγκτής (SDN Controller)**, ο οποίος ελέγχει τη διακινούμενη πληροφορία επικοινωνώντας με τους προγραμματιζόμενους **μεταγωγείς (switches)** του SDN χρησιμοποιώντας το **πρωτόκολλο OpenFlow**. Ο εν λόγω έλεγχος θα υλοποιηθεί υπό τη μορφή **ροών δρομολόγησης (routing flows)** που αποστέλλονται στον SDN ελεγκτή μέσω του **REST API** που αυτός διαθέτει, χρησιμοποιώντας το **πρωτόκολλο RESTCONF**.

3.1 Κατασκευή VPN

Ως πύλες δρομολόγησης (gateways) των LANs χρησιμοποιούνται δυο **Raspberry Pis** τα οποία λειτουργούν ως **εικονικά CPEs (virtual Customer Premise Equipment – vCPEs)**. Ένας από τους στόχους της εργασίας είναι η μεταξύ τους εγκατάσταση **διόδου επικοινωνίας**, τέτοιου ώστε η απόσταση μεταξύ των υποδικτύων που εκπροσωπεί το καθένα να φαίνεται ως δυο μόνο hops:

- ένα hop από το vCPE – αποστολέα μέχρι τον vRouter, που λειτουργεί σα μεσολαβητής στην υλοποίηση του VPN
- ένα hop από τον vRouter μέχρι το vCPE – δέκτη.

Αυτό επιτυγχάνεται με τη δημιουργία δυο καναλιών GRE:

- ένα μεταξύ του πρώτου Raspberry Pi και του vRouter
- ένα μεταξύ του δεύτερου Raspberry Pi και του vRouter

Έτσι, η τελική επικοινωνία των δυο Raspberry Pis μπορεί πλέον να επιτευχθεί και μέσω του VPN που έχει υλοποιηθεί με τη χρήση του vRouter. Απαραίτητη είναι η διαμόρφωση κατάλληλου πίνακα δρομολόγησης (routing table) στο vRouter για την προώθηση της κίνησης από το ένα tunnel στο άλλο.

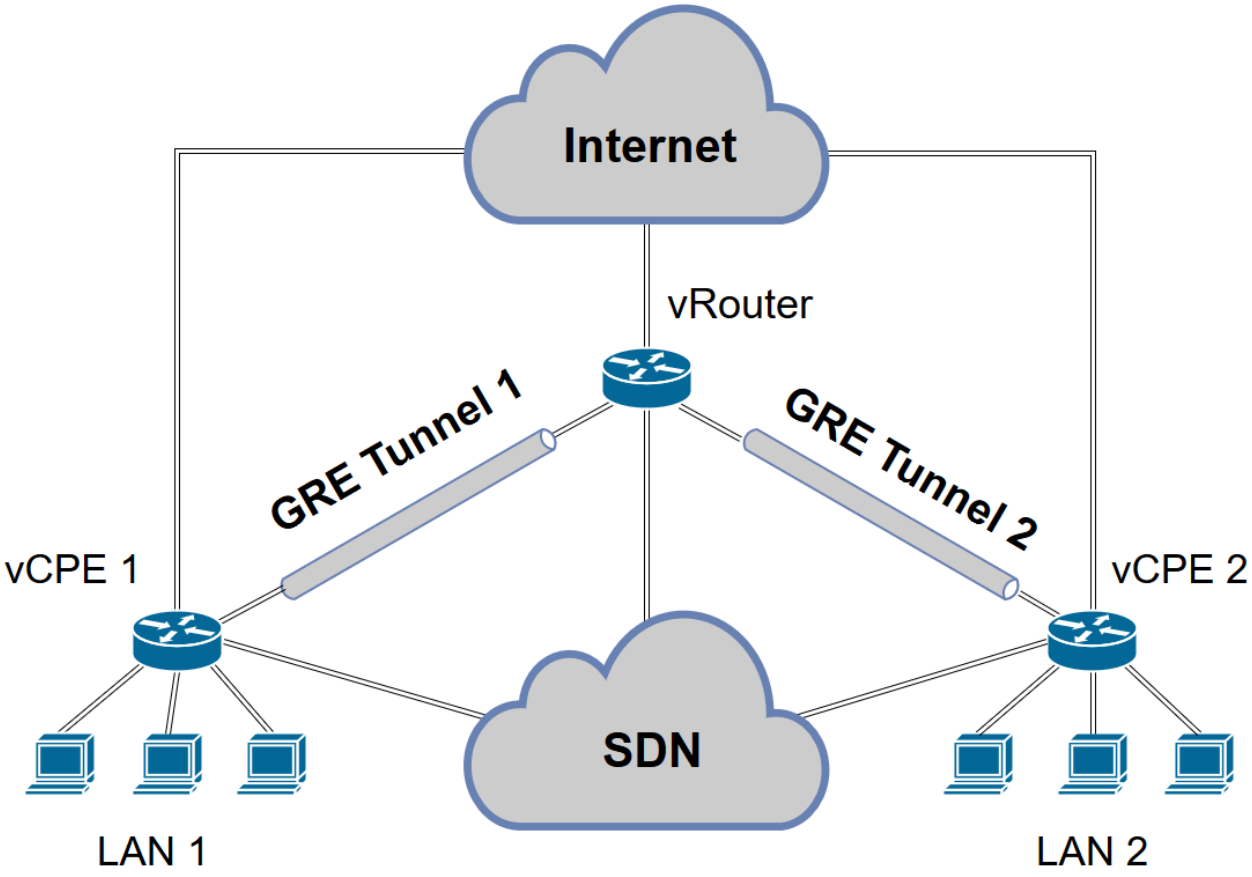
3.2 Διακοπή επικοινωνίας μέσω του δημόσιου WAN δικτύου

Το επόμενο στάδιο είναι η απομόνωση των δυο υποδικτύων, ώστε η μεταξύ τους επικοινωνία να είναι δυνατή μόνο μέσω του VPN κι όχι από το δημόσιο δίκτυο που βρίσκεται ανάμεσά τους. Από τους αποδοτικότερους τρόπους για να γίνει αυτό είναι, όπως περιγράφηκε κι ανωτέρω, με τον **κεντρικοποιημένο έλεγχο** της κίνησης χρησιμοποιώντας έναν SDN Controller.

Πρώτο βήμα λοιπόν η εγκατάσταση στην τοπολογία του OpenDaylight Controller. Στη συνέχεια, γίνεται ενημέρωση του **MD-SAL data store** του Controller μέσω REST API για τις απαραίτητες ροές που θα χρειαστούν για την αποκοπή της ανεπιθύμητης κίνησης στο δίκτυο. Ο Controller με τη σειρά του, θα ελέγξει τις προτεινόμενες ροές του χρήστη που βρίσκονται στο **configuration store** για συμβατότητα με το σύστημα. Έπειτα, θα ενημερώσει το **operational store** με τα ανωτέρω flows, τροποποιώντας τις εγγραφές ροών στους κατάλληλους μεταγωγείς του δικτύου (συγκεκριμένα στο HP Switch).

3.3 Συνολική Εικόνα

Μετά την απαραίτητη αναμονή για την ενημέρωση του συστήματος από τον Controller, το τελικό εγχείρημα έχει ολοκληρωθεί. Τα δυο υποδίκτυα επικοινωνούν μέσω του VPN, ενώ ταυτόχρονα εμποδίζεται η κίνηση πληροφορίας μέσω του δημόσιου δικτύου, χάρη στην εποπτεία των switches από τον ODL Controller. Ακολουθεί εικόνα με την αρχιτεκτονική της τοπολογίας.



3.1 Αρχιτεκτονική Συστήματος

4. Τα δομικά στοιχεία του συστήματος

Όπως έχουμε αναφέρει και ανωτέρω, το σύστημά μας μπορεί να χωριστεί σε δυο μέρη:

- **Επικοινωνία υποδικτύων μέσω VPN και**
- **Έλεγχος κίνησης μέσω SDN Controller**

4.1 VPN

Πρώτα θα περιγράψουμε τις τεχνολογίες που θα χρησιμοποιήσουμε για την υλοποίηση του VPN μεταξύ των δυο vCPE. Βασική προϋπόθεση για τη διαχείριση του VPN είναι η κατασκευή ενός δρομολογητή. Για τους σκοπούς της εργασίας μας, χρησιμοποιήσαμε ένα εικονικό μηχάνημα στο οποίο και θα φορτώσουμε τα κατάλληλα εργαλεία για να εξυπηρετήσει το σκοπό ενός **εικονικού δρομολογητή (vRouter)**. Όσον αφορά το λειτουργικό σύστημα έχουμε δυο επιλογές, μια ανοιχτού κώδικα και μια εμπορική.

4.1.1 VyOS vRouter

Η πρώτη μας επιλογή είναι ο εξοπλισμός του VM με το VyOS. Το VyOS είναι ένα λειτουργικό σύστημα ανοιχτού κώδικα βασισμένο στο GNU/Linux. Προσφέρει μια ελεύθερη κι ανοιχτού κώδικα πλατφόρμα, η οποία ανταγωνίζεται ισάξια τις εμπορικά διαθέσιμες αντίστοιχου σκοπού ευκαιρίες από τους γνωστούς παρόχους δικτυακών λύσεων. Επειδή το VyOS τρέχει σε αρχιτεκτονική x86 συστημάτων, ενδείκνυται για χρήση ως λειτουργικό σύστημα δρομολογητή αλλά και πλατφόρμα τείχους προστασίας (firewall) σε συστήματα υπολογιστικού νέφους (cloud).

Το VyOS ενσωματώνει πολλαπλά χαρακτηριστικά που είναι απαραίτητα για τη σωστή διαχείριση ενός δικτύου, όπως επιβολή πρωτοκόλλων δρομολόγησης, πλήθος διαθέσιμων διεπαφών δικτύου, υπηρεσίες firewall, μετάφρασης διεύθυνσης δικτύου (Network Address Translation - NAT), πρωτόκολλο δυναμικής εκχώρησης διευθύνσεων (Dynamic Host Configuration Protocol - DHCP), υποστήριξη Internet Protocol version 6 (IPv6) κλπ. Εμείς θα αναφερθούμε συγκεκριμένα σε αυτά που χρησιμοποιήθηκαν για την υλοποίηση του VPN. [20]

Quagga: Πρόκειται για το λογισμικό που χρησιμοποιεί το VyOS για τη διαχείριση της δρομολόγησης στο δίκτυο, γνωστό σε πλατφόρμες που τρέχουν σε Unix-like λειτουργικά συστήματα. Η χρήση του επικεντρώνεται κυρίως στη διαχείριση των διεπαφών και στην επιβολή πρωτοκόλλων δρομολόγησης, όπως Open Shortest Path First (OSPF), Border Gateway Protocol (BGP) κλπ. Σε αυτήν την εργασία χρησιμοποιήθηκε για την εγκατάσταση των απαραίτητων διεπαφών στον vRouter καθώς και για τους απαραίτητους κανόνες δρομολόγησης μεταξύ των επιμέρους κομματιών του VPN. Αναλυτικότερη περιγραφή της υλοποίησης ακολουθεί σε επόμενο κεφάλαιο. [21]

Τεχνικές VPN: Το VyOS δίνει τη δυνατότητα για δημιουργία ποικιλίας διεπαφών, καλύπτοντας πλήθος αναγκών στο δίκτυο. Συγκεκριμένα υποστηρίζει την κατασκευή διόδων Generic Routing Encapsulation (GRE) και Layer 2 Tunneling Protocol (L2TP), απαραίτητα χαρακτηριστικά όπως εξηγήσαμε και στο κεφάλαιο 2 για την υλοποίηση ενός VPN. Εμείς θα χρησιμοποιήσουμε **GRE Tunnels**.

4.1.2 Cisco Cloud Services Router (CSR) 1000v

Η αντίστοιχη επιλογή σε εμπορική λύση είναι το CSR 1000v. Πρόκειται για μια έτοιμη διανομή vRouter που περιλαμβάνει επιλεγμένα χαρακτηριστικά από το λειτουργικό σύστημα Cisco IOS-XE ® και μπορεί να τρέξει σε Cisco UCS ® εξυπηρετητές (servers) ή servers γνωστών παρόχων υπηρεσιών δικτύου που υποστηρίζουν την πλατφόρμα εικονικοποίησης **VMware ESXi 6.0**. Αποσκοπεί κυρίως στην ανάπτυξη σε κέντρα δεδομένων υπολογιστικού νέφους. Στα πλαίσια της εργασίας μας θα το χρησιμοποιήσουμε με τον ίδιο τρόπο που θα χρησιμοποιήσουμε και το VyOS vRouter, δηλαδή για τη διαχείριση του VPN και της διακινούμενης πληροφορίας εντός αυτού.

Τα χαρακτηριστικά του CSR 1000v δε διαφέρουν από το VyOS vRouter σε λειτουργικότητα. Πρόκειται για ένα ολοκληρωμένο πακέτο που παρέχει λύσεις για δρομολόγηση, VPN, firewall, NAT, βελτιστοποίηση δικτύων ευρείας ζώνης (Wide Area Network – WAN) κλπ. Η κύρια διαφορά με την άλλη μας επιλογή για vRouter έγκειται στην απουσία του λογισμικού Quagga, μιας και το CSR 1000v τρέχει με προεγκατεστημένο λογισμικό της Cisco. Αναλυτική περιγραφή της υλοποίησης του VPN με χρήση του CSR 1000v ακολουθεί σε επόμενο κεφάλαιο. [22]

4.1 Παράδειγμα χρήσης των vRouters [23]

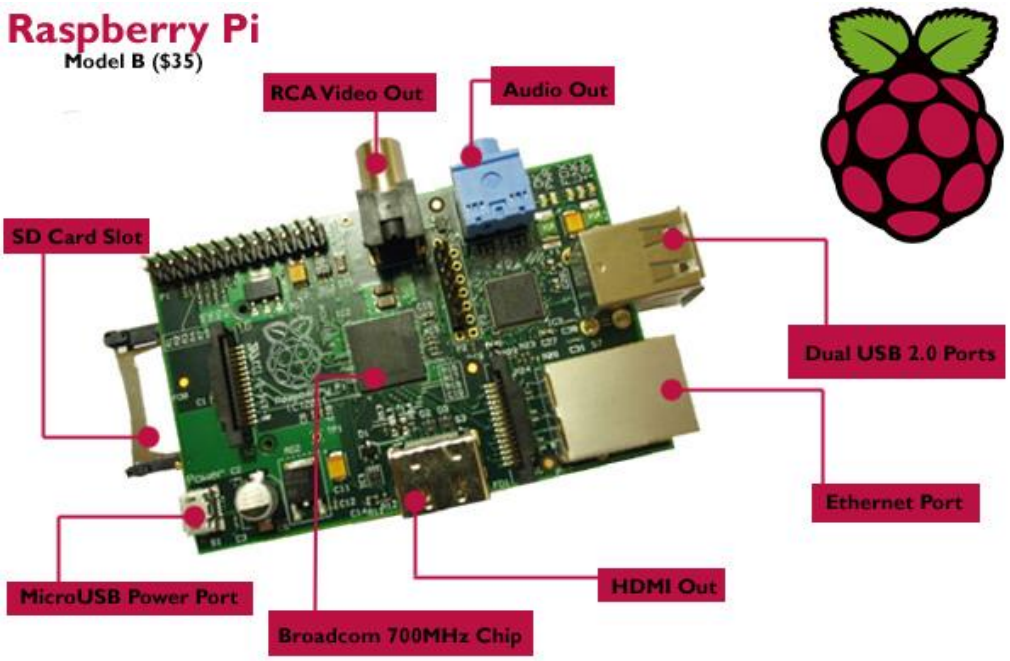
4.2 Raspberry Pi (RPi)

Πρόκειται για έναν ολοκληρωμένο υπολογιστή σχεδιασμένο στο Ηνωμένο Βασίλειο με σκοπό τη διδασκαλία. Αποτελείται από μια μονάχα πλακέτα, η οποία έχει το μέγεθος πιστωτικής κάρτας. Το κύκλωμα περιλαμβάνει μικροεπεξεργαστή, μνήμη, κάρτα γραφικών, θύρες εισόδου/εξόδου κι άλλα χαρακτηριστικά που απαιτούνται για τη σωστή λειτουργία ενός υπολογιστικού συστήματος. Στην αρχή η παραγωγή τους αποσκοπούσε για χρήση σε ακαδημαϊκό πλαίσιο για την εκμάθηση την επιστήμης των υπολογιστών, αλλά αργότερα χρησιμοποιήθηκε και για άλλες εφαρμογές, όπως multimedia center, υπηρεσίες νέφους, οικιακούς αυτοματισμούς, ρομποτική κλπ.

Στα πλαίσια της εργασίας μας θα χρησιμοποιήσουμε δυο RPis ως CPEs που ανήκουν σε διαφορετικά υποδίκτυα και θέλουν να επικοινωνήσουν μέσω του VPN. Συγκεκριμένα πρόκειται για το μοντέλο Raspberry Pi Model B, το οποίο τρέχει πάνω στο λειτουργικό σύστημα **Raspbian**.

Το Raspbian πρόκειται για διανομή βασισμένη στο λειτουργικό σύστημα Debian και έχει αναπτυχθεί ειδικά για το RPi. Μάλιστα, είναι το επίσημο λειτουργικό σύστημα που χρησιμοποιείται σήμερα στα RPis, καθώς διανέμεται επισήμως και από την το ίδρυμα Raspberry Pi Foundation που το σχεδίασε. Η επιτυχία του έγκειται στο γεγονός ότι συλλειτουργεί άριστα με τον ARM version 6 μικροεπεξεργαστή που διαθέτει το RPi.

Όπως θα δείξουμε αναλυτικότερα σε επόμενο κεφάλαιο, το κάθε RPi θα συνδέεται μέσω ενός διόδου GRE με το vRouter για να υλοποιήσουμε το συνολικό VPN. Για το σκοπό αυτό, θα κατασκευάσουμε από μια διεπαφή tunnel στα RPis έτσι ώστε να μπορέσουν τελικά να επικοινωνήσουν μεταξύ τους μέσω του εικονικού ιδιωτικού δικτύου που θα κατασκευασθεί. [24] [25]



4.2 Raspberry Pi Model B [26]

4.3 OpenDaylight Controller

Το επόμενο μέλημά μας μετά την κατασκευή του VPN είναι ο κεντρικός έλεγχος της διακινούμενης πληροφορίας στο δημόσιο δίκτυο εκτός του VPN. Το **OpenDaylight Project** πρόκειται για ένα συνεργατικό έργο ανοιχτού κώδικα που θα μας επιτρέψει να κάνουμε ακριβώς αυτό. Στόχος του είναι να προωθήσει τη χρήση τεχνικών SDN και NFV για τη διαχείριση των όλο και συχνότερα εμφανιζόμενων σύγχρονων δυναμικών δικτύων σε σύγκριση με τα παλαιότερα στατικά δίκτυα. Το λογισμικό είναι γραμμένο σε γλώσσα προγραμματισμού Java.

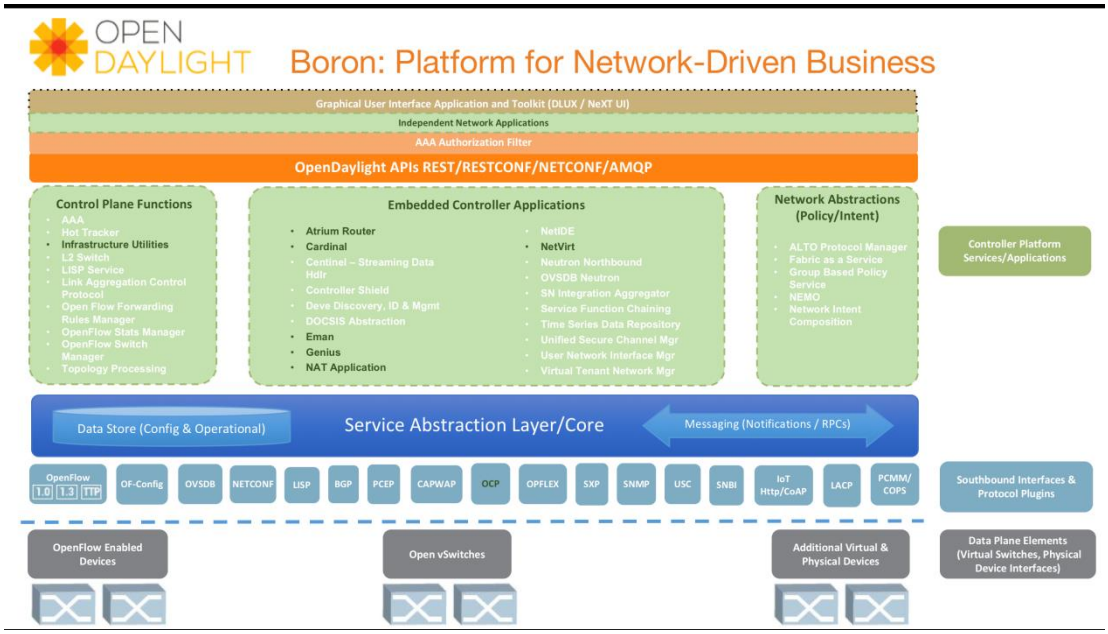
Το OpenDaylight Project αποτελείται από το συνασπισμό μεγάλων ονομάτων στο χώρο της παροχής δικτυακών υπηρεσιών. Ανάμεσά τους οι Brocade, Cisco, Ericsson, HP, IBM, Juniper Networks, Microsoft, NEC, Red Hat και VMware. Υποστηρίζει τη χρήση του **πρωτοκόλλου OpenFlow** και παρέχει SDN Ελεγκτή (Controller), λειτουργικότητα για εικονικά δίκτυα που επαφίενται (overlay) πάνω σε άλλα, όπως πχ peer-to-peer και client-server, επέκταση λειτουργιών μέσω plug-in πρωτοκόλλων αλλά και βελτίωση συσκευών μεταγωγής.

Έχοντας αναλύσει σε προηγούμενο κεφάλαιο τη λειτουργία του SDN Controller, εδώ θα επικεντρωθούμε στην περιγραφή των χαρακτηριστικών της υλοποίησης που προσφέρει το OpenDaylight Project (ODL).

Ο ODL Controller είναι ικανός να αναπτυχθεί σε ποικιλία δικτύων. Προσφέρει μοντελοποιημένο πλαίσιο λειτουργίας αλλά αν χρειαστεί παρέχει υποστήριξη ενσωμάτωσης ανερχόμενων αλλά και ήδη υπαρχόντων SDN πρωτοκόλλων.

Ο ελεγκτής ODL αναπτύσσει ανοιχτές **Northbound APIs**, οι οποίες χρησιμοποιούνται από εφαρμογές. Αυτές οι εφαρμογές αξιοποιούν τον ελεγκτή για να συλλέξουν πληροφορίες για το δίκτυο, να τρέξουν αλγορίθμους ανάλυσης της επίδοσής του και στη συνέχεια να δημιουργήσουν μέσω του ελεγκτή νέους κανόνες εντός του δικτύου.

Ο ODL ελεγκτής αναπτύσσεται αποκλειστικά σε λογισμικό και διατηρείται εντός ξεχωριστής εικονικής μηχανής Java (Java Virtual Machine – JVM). Αυτό σημαίνει ότι μπορεί να αναπτυχθεί σε φυσικά μηχανήματα και πλατφόρμες λειτουργικών συστημάτων που υποστηρίζουν τη γλώσσα προγραμματισμού Java. Για βέλτιστα αποτελέσματα, συστήνεται η χρήση από τον ODL ελεγκτή πρόσφατης διανομής λειτουργικού συστήματος Linux και ενός JVM έκδοσης τουλάχιστον 1.7. Για τη δική μας εργασία, θα χρησιμοποιήσουμε την έκδοση **Boron-SR2** του ODL Controller. [27]



4.3 Αρχιτεκτονική ODL Boron Controller (Δεκέμβριος 2016) [28]

Ακολουθεί περιγραφή των πυλώνων λειτουργίας του ODL ελεγκτή: πλατφόρμα ελεγκτή, Northbound API, Southbound API, και βασικών λειτουργιών που εγκαταστάθηκαν προκειμένου να εξασφαλίσουμε την ορθή εκτέλεση των λειτουργιών που απαιτούνται για την περάτωση της τρέχουσας εργασίας.

4.3.1 Πλατφόρμα ελεγκτή

Όπως αναλύσαμε και σε προηγούμενο κεφάλαιο, τα ανεξάρτητα στοιχεία του ελεγκτή περιγράφονται στο σύστημα από τη northbound διεπαφή του, ενώ οι υψηλότερου επιπέδου αφαιρετικές ενέργειες περιγράφονται από τη southbound διεπαφή του. Η northbound διεπαφή μεταφέρει δηλαδή χαμηλού επιπέδου πληροφορία προς τα υψηλότερα στρώματα. Έτσι εξασφαλίζεται η σωστή λειτουργία αλλά και ενδοεπικοινωνία των ξεχωριστών λειτουργιών που προσφέρει ο ODL ελεγκτής. Όλα αυτά διαχειρίζονται στα πλαίσια της πλατφόρμας του ελεγκτή όπως φαίνεται και στο ανωτέρω σχήμα.

4.3.2 Διεπαφή Northbound

Οι υπηρεσίες κι οι ReST API εφαρμογές που χρειάζεται ο ελεγκτής για να διαχειριστεί και να παραμετροποιήσει αποτελεσματικά το δίκτυο για το οποίο είναι υπεύθυνος παρέχονται σε αυτόν από τη **Northbound** διεπαφή του. Πρόσβαση σε αυτή εξασφαλίζεται μέσω επιβεβαίωσης ταυτότητας του χρήστη που θέλει να αποκτήσει πρόσβαση σε αυτή (user authentication).

4.3.3 Διεπαφή Southbound

Τα πρωτόκολλα και οι επεκτάσεις (plug-ins) που απαιτούνται για τη διαχείριση και τον έλεγχο του δικτύου αλλά και την πιθανή ανάγκη επικοινωνίας με το φυσικό μηχάνημα (hardware) περιγράφονται από τη **Southbound** διεπαφή του ελεγκτή. Παράδειγμα τέτοιων πρωτοκόλλων είναι το OpenFlow και το πρωτόκολλο διαχείρισης των εικονικών μεταγωγέων (Open vSwitch Database – OVSDB).

4.3.4 Βασικές Λειτουργίες Δικτυακής Εξυπηρέτησης (Base Network Service Functions)

Ο ODL ελεγκτής δεν έρχεται προεγκατεστημένος με όλα τα απαραίτητα στοιχεία που χρειάζονται για να καλύψουν τις ανάγκες μας. Για αυτό θα περιγράψουμε τα βασικά χαρακτηριστικά που θα χρειαστεί να εγκαταστήσουμε χειροκίνητα πριν τον εκκινήσουμε. Η πλήρης λίστα με τα διαθέσιμα APIs βρίσκεται στον ακόλουθο σύνδεσμο:

http://<odl _controller_ip>:8181/apidoc/explorer/index.html

όπου *<odl _controller_ip>* η IP διεύθυνση του εικονικού μηχανήματος στο οποίο βρίσκεται ο ελεγκτής.
Ο αναλυτικός τρόπος εγκατάστασής τους ακολουθεί σε επόμενο κεφάλαιο.

- **Διαχειριστής Τοπολογίας (Topology Manager)**

Αποθηκεύει και διαχειρίζεται πληροφορίες για τις συσκευές του δικτύου, συμπεριλαμβανομένων των ικανοτήτων και της εμβέλειάς τους. Αρχικά, η μόνη γνώση περιορίζεται στον αρχικό κόμβο της τοπολογίας (ένας κεντρικός μεταγωγέας). Καθώς ο διαχειριστής στήνει το δίκτυό του, ο Topology Manager ενημερώνεται κατάλληλα και προσθέτει τους αντίστοιχους κόμβους στην τοπολογία του δικτύου.

- **Διαχειριστής Μεταγωγέων (Switch Manager)**

Το API του Switch Manager διατηρεί τις λεπτομέρειες κάθε στοιχείου του δικτύου. Όταν ένα νέο στοιχείο ανακαλύπτεται στο δίκτυο, τα χαρακτηριστικά του (πχ τι μεταγωγέας/δρομολογητής είναι, έκδοση, ικανότητες, θύρες στις οποίες είναι συνδεδεμένο κλπ) αποθηκεύονται στη βάση δεδομένων από τον Switch Manager. Εναλλακτική πρόσβαση σε αυτές τις πληροφορίες παρέχεται από το Northbound API.
Κατά την άφιξη της πληροφορίας, αν η διεύθυνση MAC (Layer 2) του αποστολέα είναι άγνωστη, τότε εκπέμπεται μήνυμα προς όλους (broadcast) για την ανακάλυψή της. Αλλιώς, το πακέτο παραδίδεται στον κόμβο προορισμού.

- **Διαχειριστής τερματικών (Host Tracker)**

Εντοπίζει την τοποθεσία των τερματικών χρησιμοποιώντας τις MAC διευθύνσεις τους ως κύριο αναγνωριστικό. Στη συνέχεια, δημιουργεί ένα κόμβο στην τοπολογία για την αναπαράσταση του νέου τερματικού και το συνδέει με τους κατάλληλους μεταγωγείς. Τέλος, αποθηκεύει τις απαραίτητες πληροφορίες (διευθύνσεις MAC, IP, τύπος συνδεόμενων μεταγωγέων, αριθμός θυρών κλπ) και παρέχει τα απαραίτητα APIs για την πρόσβαση σε αυτά.

- **Διαχειριστής Κανόνων Προώθησης (Forwarding Rules Manager)**

Επιβάλλει/τροποποιεί/διαγράφει κανόνες προώθησης μεταξύ των διάφορων κόμβων του δικτύου και τους ενημερώνει κατάλληλα. Για να το κάνει αυτό, μιλάει άμεσα με τις Southbound επεκτάσεις του ελεγκτή.

- **Διαχειριστής Πρωτοκόλλου επίλυσης διευθύνσεων – Address Resolution Protocol (ARP Handler)**

Διαχειρίζεται τις αιτήσεις/απαντήσεις ARP εντός του δικτύου. Επιπλέον, εξασφαλίζει τον περιορισμό της περιττής broadcast κίνησης, καθιστώντας τον καίριας σημασίας για την αλληλεπίδραση με τους τελικούς κόμβους του δικτύου. Αυτό επιτυγχάνεται καθώς απαντά αυτός άμεσα σε ARP αιτήματα για τερματικά που γνωρίζει.

4.3.5 Επίπεδο Αφαίρεσης Μοντελοποιημένων Υπηρεσιών (Model-Driven Service Abstraction Layer – MD-SAL)

Διαχωρίζει τις Southbound (SB) επεκτάσεις πρωτοκόλλων από τις Northbound (NB) επεκτάσεις υπηρεσιών/εφαρμογών. Ορίζει τη δυνατότητα ανταλλαγής μηνυμάτων και αποθήκευσης πληροφορίας για τα δεδομένα, τις ειδοποιήσεις και τις απομακρυσμένες κλήσεις διεργασιών (Remote Procedure Calls– RPCs) που αναπτύσσει ο προγραμματιστής εφαρμογών για την πλατφόρμα του ελεγκτή. Πριν περάσουμε στην ανάλυση της αρχιτεκτονικής, θα περιγράψουμε κάποιες βασικές έννοιες πάνω στις οποίες βασίζεται το MD-SAL.

Το MD-SAL χρησιμοποιεί τη **γλώσσα YANG** για τη μοντελοποίηση των ορισμών υπηρεσιών και δεδομένων.

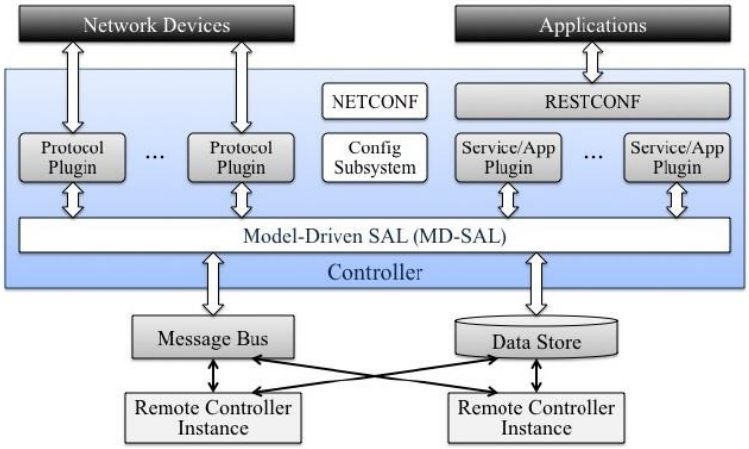
- **YANG**: Πρόκειται για τη γλώσσα που χρησιμοποιεί το MD-SAL για τη μοντελοποίηση των ορισμών υπηρεσιών και δεδομένων. Αποτελεί σημείο – κλειδί της μοντελοποιημένης συμπεριφοράς εντός του ελεγκτή. Χρησιμοποιείται από τους προγραμματιστές για τη μοντελοποίηση της λειτουργικότητας των εφαρμογών και για τη δημιουργία APIs από τα ορισμένα μοντέλα, τα οποία αργότερα θα χρησιμοποιηθούν για να παρέχουν τις υλοποιήσεις των εφαρμογών αυτών. Η YANG υποστηρίζει τη μοντελοποίηση **λειτουργικών και ρυθμιστικών χώρων αποθήκευσης δεδομένων (operational & configuration data stores)**.

- **Configuration data store**: Περιλαμβάνει όλες τις πληροφορίες παραμετροποίησης με τις οποίες ο χρήστης επιθυμεί να **ενημερώσει (push)** τον ελεγκτή. Η επικοινωνία χρήστη – ελεγκτή γίνεται με διάφορους τρόπους. Στα πλαίσια της εργασίας μας θα χρησιμοποιήσουμε το **REST API** του ελεγκτή αλλά και το **πρωτόκολλο RESTCONF** για να το κάνουμε αυτό. Περισσότερα για την αρχιτεκτονική ReST θα ακολουθήσουν σε επόμενη ενότητα.

- **Operational data store**: Περιλαμβάνει όλες τις πληροφορίες που προέρχονται από το σύστημα για την τρέχουσα λειτουργική διαρρύθμιση του δικτύου. Δεν επιδέχονται τροποποίηση (read-only) και εμφανίζονται μόνο στην περίπτωση που η επικοινωνία χρήστη – configuration store ήταν επιτυχημένη, με αποτέλεσμα οι επιθυμητές ρυθμίσεις να περαστούν σωστά (pushed) στο operational store του ελεγκτή.

Αναλυτική περιγραφή της επικοινωνίας του χρήστη με τα stores και της ενημέρωσης των συσκευών του δικτύου από τα stores του ελεγκτή (push) ακολουθεί σε επόμενη ενότητα. [29]

Έχοντας λοιπόν ορίσει τις παραπάνω έννοιες, μπορούμε να χάρη στα data stores να διαχωρίσουμε τις επεκτάσεις (plug-ins) του ελεγκτή σε δυο κατηγορίες στο MD-SAL: μπορούν να είναι δεδομένα/πάροχοι υπηρεσιών (data/service providers) ή δεδομένα/καταναλωτές υπηρεσιών (data/service consumers). Ένας πάροχος παρέχει δεδομένα/υπηρεσίες στα data stores μέσω των APIs του. Ένας καταναλωτής καταναλώνει υπηρεσίες/δεδομένα που παρέχονται από έναν ή περισσότερους παρόχους, τα οποία βρίσκονται στα data stores.



4.4 Αρχιτεκτονική MD-SAL [30]

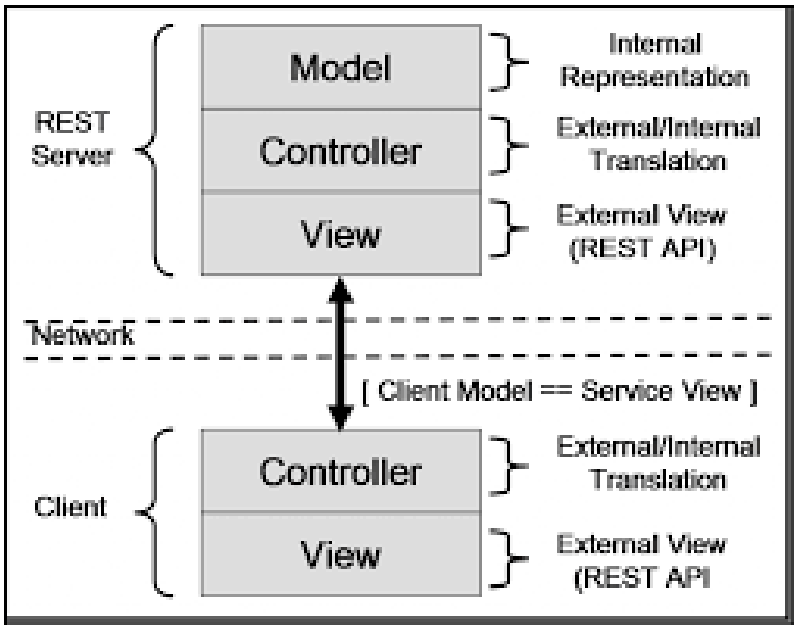
4.3.6 Αρχιτεκτονική REST και πρωτόκολλο RESTCONF

Η **αναπαραστατική κατάσταση μεταφοράς (Representational State Transfer / RESTful)** πρόκειται για ένα σύνολο λειτουργιών που δίνουν τη δυνατότητα επικοινωνίας μεταξύ υπολογιστικών συστημάτων και του διαδικτύου. Οι δικτυακές υπηρεσίες που είναι σύμφωνες με την αρχιτεκτονική REST, επιτρέπουν σε συστήματα να έχουν πρόσβαση και να επεξεργάζονται δικτυακούς πόρους σε μορφή κειμένου, χρησιμοποιώντας ένα ενιαίο και προκαθορισμένο σύνολο λειτουργιών “χωρίς κατάσταση” (stateless).

Στο διαδίκτυο, σε μια υπηρεσία RESTful, τα αιτήματα που γίνονται στο URI ενός πόρου, επιστρέφουν απάντηση η οποία μπορεί να είναι σε **μορφή XML, JSON** ή κάποια άλλη ορισμένη μορφή. Απομακρυσμένοι εξυπηρετητές αποκτούν πρόσβαση στους πόρους αυτούς με τη χρήση των προκαθορισμένων μεθόδων αιτήματος (request methods) που παρέχει το **Πρωτόκολλο Μεταφοράς Υπερκειμένου (HyperText Transfer Protocol - HTTP)**, όπως είναι οι **GET, POST, PUT, DELETE, PATCH** κλπ. Αναλυτική περιγραφή της χρήσης των μεθόδων αυτών για το πέρασμα παραμέτρων στα data stores του ελεγκτή ακολουθούν σε επόμενο κεφάλαιο.

Κάνοντας χρήση των stateless πρωτοκόλλων και προκαθορισμένων λειτουργιών, τα συστήματα REST στοχεύουν σε:

- υψηλή απόδοση
- αξιοπιστία
- ικανότητα κλιμάκωσης
- επαναχρησιμοποίηση στοιχείων τα οποία μπορούν να διαχειριστούν και να ενημερωθούν χωρίς να επηρεάζουν το σύστημα, ακόμη και κατά τη διάρκεια της εκτέλεσης
- αξιοποίηση της μνήμης cache
- απλουστευμένη χρήση χάρη στην αξιοποίηση ενιαίας διεπαφής
- ορατότητα της επικοινωνίας μεταξύ των στοιχείων και των παρόχων/καταναλωτών υπηρεσιών
- φορητότητα των στοιχείων μετακινώντας τον κώδικα του προγράμματος εντός των δεδομένων [31]



4.5 Αρχιτεκτονική REST [32]

Όσον αφορά το **RESTCONF**, πρόκειται για ένα πρωτόκολλο που βασίζεται στην αρχιτεκτονική REST και παρέχει μια προγραμματιστική διεπαφή για πρόσβαση σε δεδομένα που έχουν ορισθεί στη γλώσσα YANG χρησιμοποιώντας τα data stores του ελεγκτή.

Πρόσβαση στα δεδομένα που βρίσκονται στα configuration και operational data stores αποκτάται με χρήση της HTTP μεθόδου GET, ενώ τα δεδομένα του configuration data store μπορούν να επεξεργασθούν με χρήση των HTTP μεθόδων POST, PUT, PATCH και DELETE. Όπως αναφέρθηκε και πιο πάνω, τα δεδομένα είναι κωδικοποιημένα σε μορφή XML ή JSON.

Ο ODL ελεγκτής λοιπόν δίνει τη δυνατότητα στο χρήστη, όπως εξηγήσαμε, να διαβάσει τα δεδομένα και των δυο data stores του ελεγκτή, αλλά να επεξεργαστεί μόνο αυτά που βρίσκονται εντός του configurational data store. Έπειτα αυτός, μετά την έγκριση των αλλαγών, μεταφέρει τις κατάλληλες ρυθμίσεις στο operational data store και κάνει push τις ενημερώσεις στους αντίστοιχους κόμβους του δικτύου.

Κάθε αίτηση που του γίνεται μέσω κάποιας εκ των HTTP μεθόδων, πρέπει να απευθύνεται στο αντίστοιχο URI που θα έχει τη μορφή:

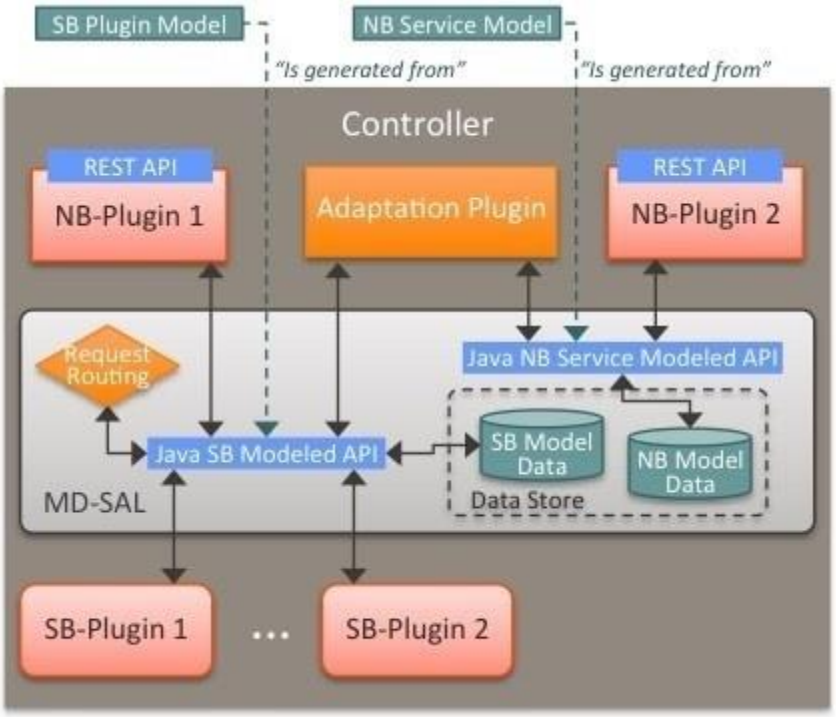
http://<odl _controller_ip>:8181/restconf/config/opendaylight-inventory:nodes/node/<path>

http://<odl _controller_ip>:8181/restconf/operational/opendaylight-inventory:nodes/node/<path>

όπου:

- *<odl _controller_ip>* η IP διεύθυνση του εικονικού μηχανήματος στο οποίο βρίσκεται ο ελεγκτής
- *<path>* το υπόλοιπο μονοπάτι εντός του αντίστοιχου δέντρου του MD-SAL για τον εντοπισμό του κατάλληλου κόμβου

Αναλυτικά παραδείγματα για την προσθήκη/τροποποίηση/διαγραφή ρυθμίσεων στον ελεγκτή ακολουθούν στο επόμενο κεφάλαιο. [29]



4.6 Συνολική εποπτεία της λειτουργίας του ελεγκτή [29]

4.4 Μεταγωγέας Hewlett Packard – Σειρά 2920 (HP 2920 Switch Series)

Ο μεταγωγέας HP 2920 πρόκειται για μια οικονομικά αποτελεσματική, ασφαλή και επεκτάσιμη λύση για πελάτες που χτίζουν δίκτυα υψηλής απόδοσης. Μπορεί να αναπτυχθεί σε μεμονωμένα επιχειρησιακά δίκτυα, σε απομακρυσμένα επιχειρησιακά παραρτήματα και σε ολοκληρωμένα δίκτυα παροχής τηλεφωνικών και διαδικτυακών υπηρεσιών. Υποστηρίζει δυνατότητες δρομολόγησης τόσο σε επίπεδο υλικού (στατική δρομολόγηση και δρομολόγηση με χρήση πρωτοκόλλου Routing Information – RIP) όσο και σε επίπεδο λογισμικού (υλοποίηση SDN με χρήση πρωτοκόλλου OpenFlow).

Ο μεταγωγέας HP 2920 χρησιμοποιείται στην υλοποίηση του SDN της εργασίας. Μέσω του πρωτοκόλλου Openflow, επικοινωνεί με τον OpenDaylight ελεγκτή για να αποκτήσει τις απαραίτητες ροές για την αποκοπή της ανεπιθύμητης πληροφορίας μεταξύ των δυο vCPEs. [33]

4.7 HP 2920 Switch [33]

4.5 Επιπρόσθετα εργαλεία και εφαρμογές

Προτού προβούμε στην ανάλυση των βημάτων που ακολουθήσαμε για την υλοποίηση της απαιτούμενης για την εργασία τοπολογίας, θα περιγράψουμε τα επιπρόσθετα εργαλεία που χρειάζονται για την επίτευξη του σκοπού αυτού.

4.5.1 OpenDaylight User Experience (DLUX)

Πρόκειται για την OpenFlow εφαρμογή διαχείρισης δικτύου που έχει αναπτυχθεί για τον OpenDaylight ελεγκτή. Ο ελεγκτής λαμβάνει πληροφορίες από τα ποικίλα ανεξάρτητα μοντέλα μέσω των υπηρεσιών που παρέχει το MD-SAL. Τις υπηρεσίες αυτές τις χρησιμοποιεί το DLUX για την απόκτηση πληροφοριών σχετικές με το δίκτυο και τη χρήση τους για την παροχή δυνατοτήτων διαχείρισης του δικτύου.

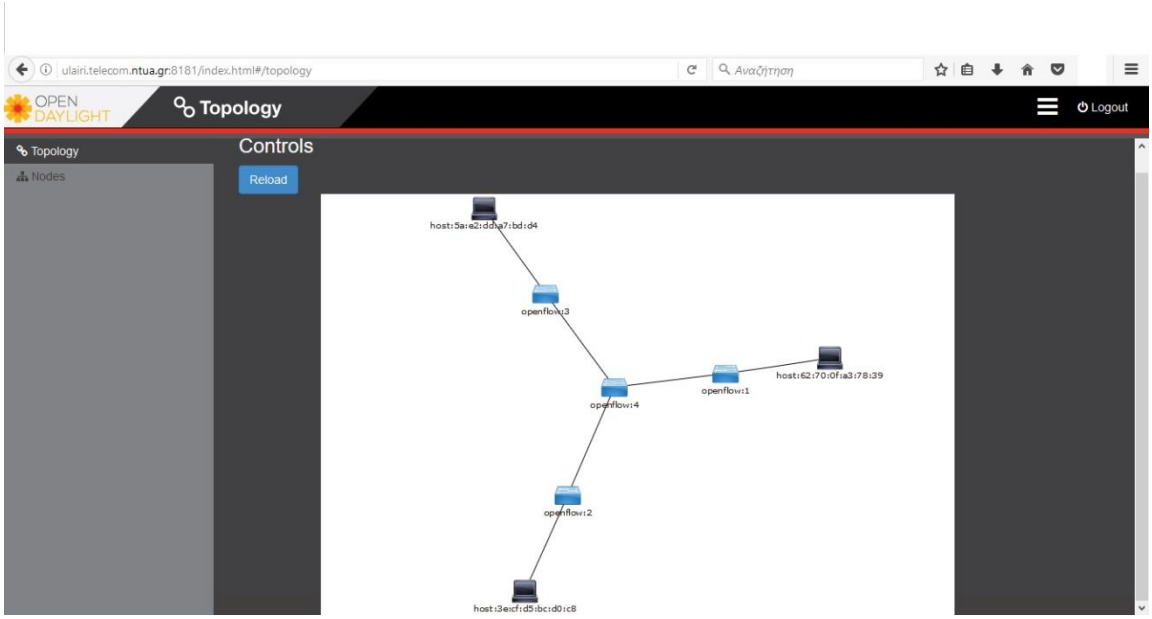
Η γραφική διεπαφή χρήστη (Graphical User Interface – GUI) αναπτύσσεται ως μια εφαρμογή και χρησιμοποιεί τη Northbound REST API για να επικοινωνήσει με τις άλλες λειτουργίες του ελεγκτή. Με αυτόν τον τρόπο, η αρχιτεκτονική του ελεγκτή διασφαλίζει πως οτιδήποτε είναι δυνατό με το GUI είναι επίσης διαθέσιμο μέσω του REST API και έτσι ο ελεγκτής μπορεί να ενσωματωθεί εύκολα σε άλλα συστήματα MANO.

Μετά την εκκίνηση του ελεγκτή, αναλυτική περιγραφή της οποίας ακολουθεί σε επόμενη ενότητα του κεφαλαίου, η πρόσβαση στο DLUX αποκτάται μετά την πιστοποίηση των στοιχείων του χρήστη προσπελάζοντας το ακόλουθο URL σε έναν οποιοδήποτε περιηγητή διαδικτύου (Internet browser):

http://<odl_controller_ip>:8181/index.html#/topology

όπου <odl_controller_ip> η IP διεύθυνση του εικονικού μηχανήματος στο οποίο είναι εγκατεστημένος ο OpenDaylight ελεγκτής.

Παράθεση των απαραίτητων χαρακτηριστικών που πρέπει να προεγκατασταθούν για τη ορθή λειτουργία του DLUX και αντικειμενική αναπαράσταση του δικτύου από αυτό ακολουθούν σε επόμενη ενότητα. [29]



4.8 Γραφική απεικόνιση ενός ευφυούς δικτύου μέσω DLUX

4.5.2 Ροές REST API (REST API flows)

Ο OpenDaylight Ελεγκτής χάρη στο MD-SAL υποστηρίζει διάφορα πρωτόκολλα ως Southbound επεκτάσεις. Συγκεκριμένα για την υλοποίηση της εργασίας μάς απασχολεί το **πρωτόκολλο RESTCONF**. Όπως εξηγήθηκε πρωτύτερα, χρησιμοποιείται για την εξωτερική ενημέρωση των παραμέτρων του ελεγκτή μέσω του REST API που διαθέτει. Οι παράμετροι αυτοί έπειτα εξετάζονται για εγκυρότητα από τον ελεγκτή, ο οποίος στη συνέχεια ενημερώνει τους κατάλληλους κόμβους των δικτύων για τις αλλαγές που προκύπτουν στο δίκτυο.

Υπάρχουν δυο είδη κόμβων σε ένα SDN δίκτυο ελεγχόμενο από OpenDaylight ελεγκτή:

- Τερματικά (Hosts): Πρόκειται για κόμβους – χρήστες. Βρίσκονται στα άκρα του δικτύου και ανάλογα με το σχεδιασμό αυτού επικοινωνούν ή όχι μεταξύ τους.

- Μεταγωγείς/Δρομολογητές (Switches/Routers): Πρόκειται για τους ενδιάμεσους κόμβους του δικτύου. Εφόσον στα SDN δίκτυα η δρομολόγηση γίνεται στο επίπεδο ελέγχου από τον ελεγκτή, οι μεταγωγείς/δρομολογητές είναι απλά υπεύθυνοι για τη μεταγωγή τις πληροφορίας εντός του δικτύου.

Επιβολή ροών δρομολόγησης στα τερματικά δεν είναι δυνατή, παρά μόνο η αποθήκευση πληροφοριών προσπέλασης (διευθύνσεις MAC & IP, θύρες στις οποίες ακούν κλπ). Συνεπώς, η αποθήκευση κανόνων δρομολόγησης γίνεται στους μεταγωγείς. Οι ενημερώσεις αυτές εκφράζονται με τη μορφή των **ροών REST API.**

Όπως αναλύσαμε και σε προηγούμενο κεφάλαιο, η αρχιτεκτονική του OpenDaylight ελεγκτή του παρέχει μέσω των υπηρεσιών MD-SAL, data stores για την αποθήκευση πληροφοριών σχετικών με την παραμετροποίηση του δικτύου και των κόμβων που ανήκουν σε αυτό. Αυτά τα data stores έχουν δενδροειδή μορφή, ακριβώς όπως λειτουργεί και το σύστημα αρχειοθέτησης (file system) ενός Linux-like λειτουργικού συστήματος.

Πληροφορίες λοιπόν για κάθε **μεταγωγέα/δρομολογητή** του δικτύου είναι αποθηκευμένες σε διαφορετικά "κλαδιά" του εν λόγω δένδρου. Η πρόσβαση σε ένα τέτοιο κλαδί επιτυγχάνεται με δυο τρόπους:

- Μέσω του γραφικού περιβάλλοντος DLUX κάνοντας κλικ πάνω αριστερά στην καρτέλα **"Nodes"**

- Προσπελάζοντας το κατάλληλο URL σε έναν περιηγητή δικτύου:

  http://<odl_controller_ip>:8181/index.html#/node/openflow:<node_identifier>

  όπου *<node_identifier>* ο αριθμός του κόμβου που επιθυμούμε να προσπελάσουμε.

Ο χρήστης δεν μπορεί άμεσα να τροποποιήσει τις λειτουργικές ρυθμίσεις του δικτύου. Υποβάλλει πρώτα τις ροές που επιθυμεί να επιβάλλει, τροποποιήσει ή διαγράψει στον ελεγκτή, οι οποίες και αποθηκεύονται στο configuration data store του ελεγκτή. Έπειτα ο ελεγκτής εξετάζει τις ροές αυτές για συμβατότητα με το δίκτυο το οποίο επιβλέπει.

Μόλις οι ροές αυτές εγκριθούν, ο ελεγκτής τις μεταφέρει στο operational data store, όπου και τις αποθηκεύει (push) στον **πίνακα ροών (flow table)** του μεταγωγέα/δρομολογητή, ο οποίος είναι υπεύθυνος για την εκτέλεση των κανόνων που αυτές περιγράφουν. Ο πίνακας αυτός καταγράφει σε μορφή εγγραφών ροών (flow entries) τις τρέχουσες ροές που είναι εγκατεστημένες σε ένα μεταγωγέα/δρομολογητή. Κάθε μεταγωγέας/δρομολογητής διαθέτει το δικό του πίνακα ροών.

Όπως έχουμε περιγράψει και σε προηγούμενο κεφάλαιο, το πρωτόκολλο RESTCONF εκφράζει τις αλλαγές με τις οποίες ενημερώνει τον ελεγκτή σε μορφή XML ή JSON. Η αναπαράσταση λοιπόν των data stores είναι δυνατή και σε αυτή τη μορφή. Συγκεκριμένα μπορούμε να δούμε μια απεικόνισή τους εισάγοντας τα ακόλουθα URLs σε ένα περιηγητή διαδικτύου:

Configuration data store
http://<odl_controller_ip>:8181/restconf/config/opendaylight-inventory:nodes/node/openflow:<node_identifier>

Operational data store
http://<odl_controller_ip>:8181/restconf/operational/opendaylight-inventory:nodes/node/openflow: <node_identifier>

όπου:

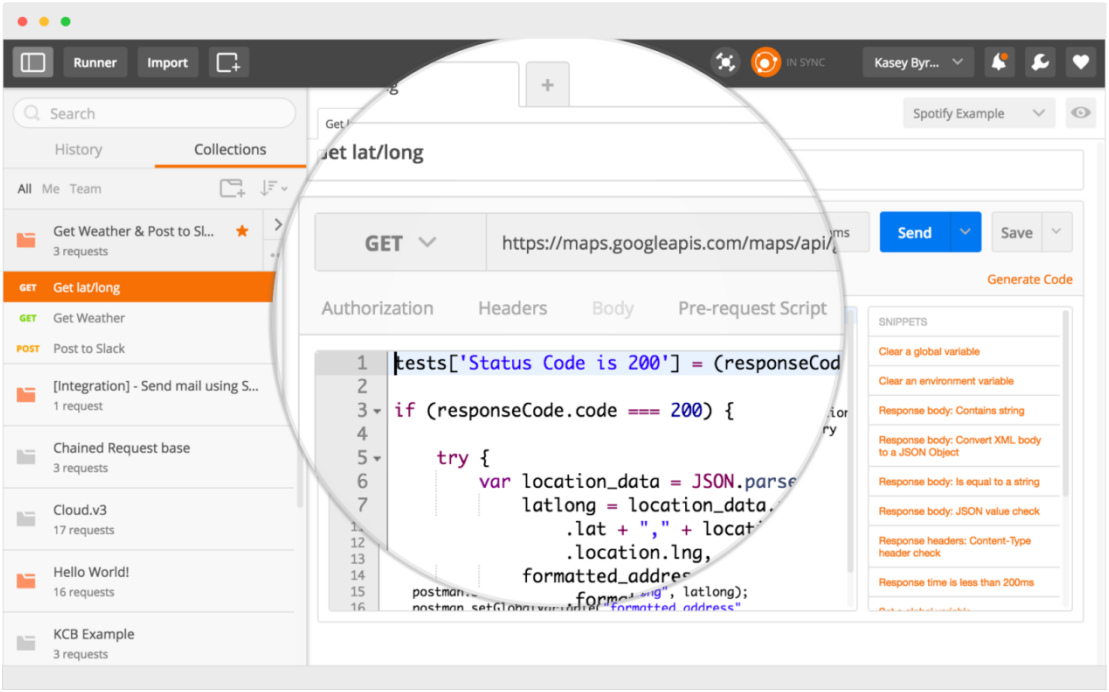
- *<odl_controller_ip>* η διεύθυνση IP του εικονικού μηχανήματος στο οποίο είναι εγκατεστημένος ο OpenDaylight ελεγκτής

- *<node_identifier>* ο αριθμός που χαρακτηρίζει μοναδικά ένα μεταγωγέα/δρομολογητή στο SDN δίκτυο.

Για τη δημιουργία τέτοιων ροών και αποθήκευσής τους στον OpenDaylight ελεγκτή θα χρησιμοποιήσουμε το εργαλείο Postman. Επεξήγηση των παραμέτρων που μπορεί να θέσει ο χρήστης κατά τη δημιουργία μιας ροής ακολουθούν σε επόμενη ενότητα. [29]

4.5.3 Postman

Ο Postman είναι ένας πελάτης REST ο οποίος εκτελείται ως εφαρμογή εντός του περιηγητή διαδικτύου Chrome (Chrome browser). Είναι πολύ χρήσιμος για την αλληλεπίδραση με REST APIs και ενδείκνυται για την εύκολη και γρήγορη ανάπτυξή τους. Παρέχει φιλικό γραφικό περιβάλλον προς το χρήστη για τη δημιουργία HTTP αιτημάτων και την ανάγνωση των HTTP απαντήσεων. Η αξιοποίησή τους στη συνέχεια πραγματοποιείται με κλήσεις REST πληκτρολογώντας διάφορες μεθόδους HTTP, όπως GET, PUT, DELETE κλπ. Η εγκατάστασή του γίνεται μέσα από το Chrome App Store. Αναλυτική περιγραφή της δημιουργίας ροών δρομολόγησης μέσω Postman ακολουθεί σε επόμενη ενότητα. [34]



4.9 Γραφικό περιβάλλον Postman [35]

5. Ενσωμάτωση Τεχνολογιών

5.1 Κατασκευή Εικονικού Ιδιωτικού Δικτύου (VPN)

Πρώτο βήμα για την υλοποίηση του VPN είναι η **κατασκευή GRE καναλιών** ανάμεσα στα vCPEs και στον vRouter. Για το σκοπό αυτό, θα δημιουργήσουμε τις κατάλληλες **διεπαφές διόδου (tunnel interfaces)**, τόσο στα Raspberry Pis που δρουν ως vCPEs, όσο και στο vRouter που δρα ως διαμεσολαβητής ανάμεσά τους.

Για την εργασία αυτή υλοποιήσαμε 2 VPNS δοκιμάζοντας διαφορετικές τεχνολογίες όσον αφορά τον εικονικό δρομολογητή (vRouter): ένα με χρήση του ανοιχτού λειτουργικού συστήματος **VyOS** κι ένα με χρήση του εμπορικού λειτουργικού συστήματος **CSR 1000v** της Cisco.

5.1.1 Διεπαφές διόδου στον εικονικό δρομολογητή (vRouter tunnel interfaces)

Χρειαζόμαστε μια δίοδο για να επικοινωνούν το RPi – αποστολέας με τον εικονικό δρομολογητή κι άλλον ένα για να επικοινωνούν ο εικονικός δρομολογητής με το RPi – παραλήπτη. Για αυτόν τον σκοπό, απαιτείται η δημιουργία δυο tunnel διεπαφών στον εικονικό δρομολογητή, μια για κάθε δίοδο.

Ακολουθεί αναλυτική περιγραφή των βημάτων παραμετροποίησης του εικονικού δρομολογητή. Οι τεχνικές διαφορές μεταξύ των δυο λειτουργικών συστημάτων φαίνονται στο παράρτημα κώδικα που βρίσκεται στο τέλος της εργασίας. Σημειώνεται ότι ο vRouter **VyOS** είναι εγκατεστημένος στο εικονικό μηχάνημα με **διεύθυνση IP 147.102.7.79**, ενώ ο vRouter **CSR 1000v** είναι εγκατεστημένος στο εικονικό μηχάνημα με **διεύθυνση IP 147.102.7.82**.

Ο εικονικός δρομολογητής (ανεξαρτήτως λειτουργικού συστήματος) έχει δυο κύριες κονσόλες (terminals) λειτουργίας: την γενική κονσόλα **(global terminal)** και την κονσόλα παραμετροποίησης **(configure terminal)**. Οι ρυθμίσεις λαμβάνουν χώρα από το configure terminal. Για αυτό το λόγο λοιπόν εκτελούμε όλες τις εντολές παραμετροποίησης από αυτό.

Πρώτα πρέπει να δημιουργήσουμε τον απαιτούμενο χώρο στην TCP επικεφαλίδα για να περιλαμβάνει και την επικεφαλίδα του πρωτοκόλλου ενθυλάκωσης GRE (GRE header). Για αυτό, δημιουργούμε μια νέα **πολιτική δρομολόγησης (routing policy)**. Την ονομάζουμε ***change-mss*** και της ορίζουμε τις εξής παραμέτρους:

- Θέτουμε ως μέγιστο μέγεθος δεδομένων που μεταδίδονται σε ένα κομμάτι (Maximum Segment Size – MSS) τα 1360 bytes, ορίζοντας έμμεσα τα υπόλοιπα 40 bytes ως χώρο για τη GRE header

- Ορίζουμε ως πρωτόκολλο της πολιτικής το Πρωτόκολλο Ελέγχου Μεταφοράς (Transmission Control Protocol – TCP)

- Επιλέγουμε η ρύθμιση του MSS να γίνει στο αρχικό πακέτο εγκατάστασης της σύνδεσης (SYN)

Μετά την παραμετροποίηση των πακέτων, προχωράμε στην δημιουργία των tunnel διεπαφών στον εικονικό δρομολογητή. Η δίοδος GRE λογίζεται ως ένα ανεξάρτητο δίκτυο. Ως μάσκα υποδικτύου επιλέγουμε 30 γιατί μας αρκούν 4 διευθύνσεις για τον ορισμό του:

- Διεύθυνση δικτύου 10.0.0.0

- Διευθύνσεις αποστολέα – παραλήπτη 10.0.0.1 και 10.0.0.2 αντίστοιχα

- Διεύθυνση εκπομπής προς όλους (broadcast) 10.0.0.3

Ονομάζουμε τη **διεπαφή** του εικονικού δρομολογητή ως ***tun0*** και ορίζουμε τις ακόλουθες ρυθμίσεις:

- τύπος ενθυλάκωσης πακέτων GRE

- διεύθυνση IP της διεπαφής ως 10.0.0.1/30

- μέγιστη μεταδιδόμενη μονάδα πληροφορίας

- (Maximum Transmission Unit – MTU) ως 1400 bytes

- επιλογή της πολιτικής δρομολόγησης *change-mss* που ορίσαμε προηγουμένως

- τοπική (local) διεύθυνση δημόσιου δικτύου τη διεύθυνση του εικονικού μηχανήματος στο οποίο είναι εγκατεστημένος ο εικονικός δρομολογητής: 147.102.7.79

- απομακρυσμένη (remote) διεύθυνση δημοσίου δικτύου τη διεύθυνση του RPi – αποστολέα: 147.102.40.69

Η παραμετροποίηση της διεπαφής διόδου του εικονικού δρομολογητή σε αυτό το σημείο έχει ολοκληρωθεί. Η ουσιαστική διαφορά των δυο λειτουργικών συστημάτων που χρησιμοποιούμε, έγκειται στον τρόπο με τον οποίο αποθηκεύονται οι αλλαγές ώστε να μη χαθούν μετά από έξοδο από την κονσόλα παραμετροποίησης (configure terminal).

Στο CSR 1000v αρκεί απλά να εξέλθουμε από αυτό. Στο VyOS όμως δεν αρκεί η έξοδος: προηγούνται τα στάδια της **αποθήκευσης (save)** των ρυθμίσεων και της δέσμευσης **(commit)** τους. Αν εξέλθουμε από το configure terminal χωρίς να προβούμε στην εκτέλεση των δυο αυτών σταδίων, οι ρυθμίσεις θα χαθούν.

Με τον ίδιο τρόπο κατασκευάζουμε και τη διεπαφή διόδου μεταξύ εικονικού δρομολογητή – RPi παραλήπτη. Φυσικά, η δεύτερη αυτή δίοδος λογίζεται ως ανεξάρτητο δίκτυο από τη δίοδο που χρησιμοποιήθηκε μεταξύ
RPi – εικονικού δρομολογητή. Παρά το γεγονός αυτό, το μέγεθός του δεύτερου καναλιού GRE θα είναι το ίδιο με το πρώτο, απλά θα έχει άλλες διευθύνσεις. Οι διαφορές στις ρυθμίσεις είναι οι εξής:

- Διεύθυνση IP της διεπαφής ως 10.0.0.5/30

- απομακρυσμένη (remote) διεύθυνση δημοσίου δικτύου τη διεύθυνση του RPi – παραλήπτη: 147.102.39.3

Για να επιβεβαιώσουμε την ορθή δημιουργία των διεπαφών διόδου στου εικονικούς δρομολογητές, εκτελούμε τις ακόλουθες εντολές:

Στον VyOS vRouter πληκτρολογούμε *show configuration* και βλέπουμε το εξής αποτέλεσμα:

```
interfaces {
  tunnel tun0 {
        address 10.0.0.1/30
        encapsulation gre
        local-ip 147.102.7.79
        mtu 1400
        parameters {
            ip {
                ttl 64
            }
        }
        policy {
            route CHANGE-MSS
        }
        remote-ip 147.102.40.69
  }
}
policy {
    access-list 100 {
            ip {
                ttl 64
            }
        }
        policy {
            route CHANGE-MSS
        }
        remote-ip 147.102.40.69
    }
    route CHANGE-MSS {
        rule 1 {
            protocol tcp
            set {
                tcp-mss 1360
            }
            tcp {
                flags SYN
            }
        }
    }
}
```

```
giannis@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

S>* 0.0.0.0/0 [1/0] via 147.102.7.200, eth0
C>* 10.0.0.0/30 is directly connected, tun0
C>* 127.0.0.0/8 is directly connected, lo
C>* 147.102.7.0/24 is directly connected, eth0
giannis@vyos:~$
```

5.1 Πίνακας δρομολόγησης VyOS vRouter

Ενώ στον CSR 1000v vRouter παρατηρούμε το ακόλουθο αποτέλεσμα:

```
R_vpn#show interfaces tunnel0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 10.0.0.5/30
  MTU 9976 bytes, BW 100 Kbit/sec, DLY 50000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel linestate evaluation up
  Tunnel source 147.102.7.82, destination 147.102.40.69
  Tunnel protocol/transport GRE/IP
```

5.2 Κατάσταση διεπαφής διόδου του εικονικού δρομολογητή CSR 1000v

```
R_vpn#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override

Gateway of last resort is 147.102.7.200 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 147.102.7.200, GigabitEthernet2
      10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        10.0.0.4/30 is directly connected, Tunnel0
L        10.0.0.5/32 is directly connected, Tunnel0
      147.102.0.0/16 is variably subnetted, 4 subnets, 2 masks
C        147.102.7.0/24 is directly connected, GigabitEthernet2
L        147.102.7.82/32 is directly connected, GigabitEthernet2
C        147.102.40.0/24 is directly connected, GigabitEthernet3
L        147.102.40.119/32 is directly connected, GigabitEthernet3
R_vpn#
```

5.3 Πίνακας δρομολόγησης εικονικού δρομολογητή CSR 1000v

5.1.2 Διεπαφή διόδου στο Raspberry Pi (Raspberry Pi tunnel interface)

Το επόμενο βήμα είναι φυσικά η δημιουργία μιας διεπαφής διόδου από πλευράς Raspberry Pi, ώστε να κλείσει η δίοδος RPi – εικονικού δρομολογητή και να είναι δυνατή η ιδιωτική επικοινωνία μεταξύ τους. Ακολουθεί αναλυτική περιγραφή των βημάτων δημιουργίας της διεπαφής. Ο αντίστοιχος κώδικας για την επίτευξη του σκοπού αυτού παρατίθεται στο παράρτημα που βρίσκεται στο τέλος της εργασίας.

Πριν την έναρξη των εργασιών, προσέχουμε να βρισκόμαστε σε **περιβάλλον διαχειριστή** εντός του Raspberry Pi. Αν αυτό δεν είναι δυνατό, τότε πριν κάθε εντολή γράφουμε ***sudo*** για την απόκτηση των απαραίτητων δικαιωμάτων.

Στο λειτουργικό σύστημα Raspbian το οποίο είναι φορτωμένο στο Raspberry Pi (και στην πλειοψηφία των Linux – like λειτουργικών συστημάτων), το **module** που καθιστά δυνατή τη δημιουργία διεπαφών διόδου είναι το **ip_gre**. Πριν ξεκινήσουμε επιβεβαιώνουμε την ύπαρξή του πληκτρολογώντας στο τερματικό του Raspberry Pi την εντολή *lsmod | grep ip_gre*.

Ονομάζουμε τη **διεπαφή tun0** και θέτουμε τις εξής παραμέτρους:

- Τύπος διεπαφής: δίοδος (tunnel)
- Πρωτόκολλο ενθυλάκωσης: GRE
- Τοπική (local) διεύθυνση δημοσίου δικτύου του Raspberry Pi (αποστολέας): 147.102.40.69
- Απομακρυσμένη (remote) διεύθυνση δημοσίου δικτύου του εικονικού μηχανήματος στο οποίο είναι εγκατεστημένος ο εικονικός δρομολογητής (VyOS): 147.102.7.79
- Κατάσταση ζεύξης από πλευράς διεπαφής Raspberry Pi: up
- Διεύθυνση IP της διεπαφής διόδου του Raspberry Pi: 10.0.0.2
- Διεύθυνση IP της διεπαφής διόδου του γείτονα (εικονικός δρομολογητής VyOS): 10.0.0.1/30

Η δημιουργία της διεπαφής διόδου του Raspberry Pi και κατά συνέπεια της ζεύξης RPi αποστολέα – εικονικού δρομολογητή σε αυτό το σημείο έχει ολοκληρωθεί. Με τον ίδιο τρόπο δημιουργούμε και τη ζεύξη με τον εικονικό δρομολογητή CSR 1000v, προσέχοντας να χρησιμοποιήσουμε τη σωστή διεύθυνση IP δημοσίου δικτύου (147.102.7.82), διαφορετικό ιδιωτικό υποδίκτυο (πχ 10.0.0.4) και νέο όνομα για τη διεπαφή διόδου (πχ tun1).
Στη συνέχεια, με την εντολή *netstat -rn* εξετάζουμε στον πίνακα δρομολόγησης του Raspberry Pi την κατάσταση των ζεύξεων:

```
pi@raspberrypi ~ $ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         147.102.40.200  0.0.0.0         UG        0 0          0 eth0
10.0.0.0        0.0.0.0         255.255.255.252 U         0 0          0 tun0
10.0.0.4        0.0.0.0         255.255.255.252 U         0 0          0 tun1
147.102.40.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
pi@raspberrypi ~ $
```

5.4 Πίνακας δρομολόγησης RPi μετά τη δημιουργία καναλιών GRE

Όπως φαίνεται και στην εικόνα, οι διεπαφές tun0 (VyOS) και tun1 (CSR 1000v) έχουν ως σημαία (Flag) την U (up), που σημαίνει ότι λειτουργούν ορθά.

Επιβεβαιώνουμε την ορθή επικοινωνία κάνοντας ping στις ιδιωτικές διευθύνσεις των δυο εικονικών δρομολογητών, 10.0.0.1 (VyOS) και 10.0.0.5 (CSR 1000v):

```
pi@raspberrypi ~ $ ping -c4 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_req=1 ttl=64 time=0.958 ms
64 bytes from 10.0.0.1: icmp_req=2 ttl=64 time=1.02 ms
64 bytes from 10.0.0.1: icmp_req=3 ttl=64 time=0.978 ms
64 bytes from 10.0.0.1: icmp_req=4 ttl=64 time=0.894 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 0.894/0.964/1.029/0.061 ms
pi@raspberrypi ~ $
pi@raspberrypi ~ $ ping -c4 10.0.0.5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_req=2 ttl=255 time=2.93 ms
64 bytes from 10.0.0.5: icmp_req=3 ttl=255 time=2.97 ms
64 bytes from 10.0.0.5: icmp_req=4 ttl=255 time=2.05 ms

--- 10.0.0.5 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3009ms
rtt min/avg/max/mdev = 2.056/2.656/2.977/0.426 ms
pi@raspberrypi ~ $
```

5.5 Αποτελέσματα ping προς τους δυο εικονικούς δρομολογητές

Τον ίδιο έλεγχο πραγματοποιούμε κι από πλευράς των εικονικών δρομολογητών:

```
giannis@vyos:~$ show interfaces tunnel
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
tun0             10.0.0.1/30                       u/u
giannis@vyos:~$
```

5.6 Κατάσταση καναλιού GRE (up/up) στον εικονικό δρομολογητή VyOS

```
giannis@vyos:~$ ping  10.0.0.2 count 4
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=0.803 ms
64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=0.734 ms
64 bytes from 10.0.0.2: icmp_req=3 ttl=64 time=0.749 ms
64 bytes from 10.0.0.2: icmp_req=4 ttl=64 time=0.727 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.727/0.753/0.803/0.035 ms
giannis@vyos:~$
```

5.7 Ping από εικονικό δρομολογητή VyOS προς RPi – αποστολέα

```
R_vpn#ping 10.0.0.6
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/5/20 ms
R_vpn#
```

5.8 Ping από εικονικό δρομολογητή CSR 1000v προς RPi – αποστολέα

Με τον ίδιο τρόπο κατασκευάζεται και το κανάλι GRE από τους εικονικούς δρομολογητές VyOS και CSR 1000v προς το RPi – δέκτη, προσέχοντας να χρησιμοποιήσουμε τη σωστή διεύθυνση IP δημοσίου δικτύου του RPi – δέκτη (147.102.39.3) και ξεχωριστά ιδιωτικά δίκτυα για τις ζεύξεις GRE μεταξύ των εικονικών δρομολογητών και του RPi (πχ 10.0.08/30 και 10.0.0.12/30).

5.2 Διακοπή επικοινωνίας μέσω του δημόσιου WAN δικτύου

Έχοντας ολοκληρώσει το εικονικό ιδιωτικό δίκτυο μεταξύ των vCPEs, προχωράμε στην εγκατάσταση κεντρικοποιημένου ελέγχου για τη διαχείριση της κίνησης στο δημόσιο δίκτυο.

5.2.1 Ελεγκτής OpenDaylight (OpenDaylight Controller)

Ακολουθώντας τις οδηγίες του επίσημου site, εγκαθιστούμε τα απαραίτητα αρχεία που απαιτούνται για την εκτέλεση του ελεγκτή OpenDaylight.

5.9 Εκκίνηση του ελεγκτή OpenDaylight

Στη συνέχεια πρέπει να πραγματοποιηθεί εγκατάσταση συγκεκριμένων χαρακτηριστικών (features) για να έχει ο ελεγκτής την απαιτούμενη συμπεριφορά κατά την εκτέλεσή του. Συγκεκριμένα, εκτελούμε την εντολή *feature:install <feature>*, όπου <feature> το επιθυμητό χαρακτηριστικό προς εγκατάσταση. Με αυτόν τον τρόπο εγκαθιστούμε τα ακόλουθα χαρακτηριστικά:

odl-dlux-all, odl-l2switch-all, odl-mdsal-all, odl-openflowplugin-all, odl-restconf

Έχοντας ολοκληρώσει τα διαδικαστικά, περνάμε στο ουσιαστικότερο κομμάτι της εργασίας: Για να λειτουργεί το δίκτυο που φτιάξαμε ανωτέρω μεταξύ των RPis και του εικονικού δρομολογητή ως VPN, θα πρέπει η επικοινωνία των RPIs να καθίσταται αδύνατη μέσω του δημοσίου δικτύου. Εδώ είναι που αποκτά τεράστια χρηστικότητα η έννοια του SDN και συγκεκριμένα ο OpenDaylight ελεγκτής.

5.2.2 Δημιουργία ροών δρομολόγησης

Η λειτουργία που καλούμαστε να επιτελέσουμε με τον ελεγκτή είναι η εξής:

Τον τοποθετούμε εντός του δημοσίου δικτύου που βρίσκεται ανάμεσα στα δυο vCPEs. Αυτό το δίκτυο ευρείας ζώνης (WAN) αποτελείται από ενδιάμεσους μεταγωγείς (ανάμεσά τους και ο HP Switch) και διάφορους άλλους κόμβους που δε μας απασχολούν. Ο επιθυμητός μας στόχος είναι οποιαδήποτε κίνηση εντοπίζεται εντός του δικτύου αυτού μεταξύ των δυο RPis να αποκόπτεται.

Συγκεκριμένα, για να μην πλημμυρίζει (flood) το δίκτυο με περιττή πληροφορία, είναι επιθυμητό (σε μικρά όπως το δικό μας δίκτυα, γιατί σε μεγαλύτερου μεγέθους είναι πλέον απαραίτητο) η κίνηση αυτή να αποκόπτεται όσο το δυνατόν γρηγορότερα, ώστε να μην ταξιδεύει άσκοπα εντός του δικτύου. Για την επίτευξη του σκοπού αυτού, απαιτείται η τοποθέτηση κανόνων δρομολόγησης στους μεταγωγείς που βρίσκονται όσο το δυνατόν πλησιέστερα στα vCPEs. Αυτήν τη λειτουργία θα αναλάβει ο OpenDaylight ελεγκτής με τη χρήση των ροών δρομολόγησης (flows) που θα επιβάλλει στους μεταγωγείς του SDN δικτύου, μέσω του REST API που διαθέτει.

Εντοπίζουμε λοιπόν τους πλησιέστερους στα RPi μεταγωγείς, έστω **openflow:1** και **openflow:2**. Στη συνέχεια, δημιουργούμε τις απαραίτητες ροές τις οποίες και θα στείλουμε στον ελεγκτή με χρήση των γνωστών HTTP μεθόδων. Οι μέθοδοι που θα χρειαστούμε είναι οι ακόλουθες τρεις:

GET: με τη χρήση αυτής της μεθόδου θα αιτηθούμε στον ελεγκτή να μας επιστρέψει το τρέχον σύνολο ροών που είναι ενεργές στους δυο μεταγωγείς.

PUT: με τη χρήση αυτής της μεθόδου στέλνουμε στον ελεγκτή τη ροή που δημιουργούμε για κάποιο μεταγωγέα.

DELETE: με τη χρήση αυτής της μεθόδου ζητάμε από τον ελεγκτή να διαγράψει μια ροή από κάποιο μεταγωγέα. Στην εργασία αυτή δε χρησιμοποιείται, αλλά παρατίθεται σε περίπτωση που ο χρήστης επιθυμεί να αφαιρέσει μια ροή από ένα μεταγωγέα.

Σημειώνεται πως, οι μέθοδοι GET και DELETE μπορούν να γίνουν και στα δυο data stores του ελεγκτή, δηλαδή configuration και operational. Η μέθοδος PUT μπορεί να γίνει μόνο προς το configuration data store. Όπως έχει περιγραφεί σε προηγούμενο κεφάλαιο, αφού ο ελεγκτής λάβει μια ροή από το χρήστη στο configuration store του, τότε θα την ελέγξει πρώτα για συμβατότητα. Μετά το στάδιο αυτό, εκείνος είναι υπεύθυνος για την ενημέρωση του operational store με τη νέα ροή, την οποία και θα προωθήσει στον κατάλληλο μεταγωγέα.

Δημιουργούμε λοιπόν τις δυο ροές που θα χρειαστεί να στείλουμε στον OpenDaylight ελεγκτή για την αποκοπή της ανεπιθύμητης κίνησης. Όπως αναφέρθηκε και πρωτύτερα, το περιεχόμενο της πληροφορίας που ανταλλάσσεται με τον ελεγκτή (συγκεκριμένα στην εργασία αυτή, οι ροές δρομολόγησης) μέσω του REST API εκφράζεται σε μορφή XML ή JSON. Θα περιγραφούν οι απαραίτητες παράμετροι που θέτουμε κατά τη δημιουργία των ροών. Ο πλήρες κώδικας που περιγράφει τις ροές βρίσκεται σε μορφή XML στο παράρτημα που βρίσκεται στο τέλος της εργασίας.

Σύμφωνα λοιπόν με τη σελίδα wiki του OpenDaylight Project, κατασκευάζουμε τη ροή για το μεταγωγέα openflow:1, με **όνομα flow11**. Θέτουμε τις εξής παραμέτρους:

- Αριθμός πίνακα ροών (flow table) του μεταγωγέα: 0

- Προτεραιότητα (priority) ροής ως 311. Επιλέγουμε ένα μεγάλο νούμερο στην εμβέλεια 0 έως 32.767. Όσο μεγαλύτερη η αριθμητική τιμή, τόσο υψηλότερη η προτεραιότητα της ροής. Αυτό γίνεται γιατί κατά την επίλυση (resolution) των διαφορών μεταξύ των ροών εντός του μεταγωγέα, επιλύεται πρώτα εκείνη με την υψηλότερη προτεραιότητα.

- Διάρκεια ζωής της ροής (hard timeout) ως 0. Μετά την πάροδο του χρόνου που θέτουμε, αν η τιμή είναι *μη μηδενική*, η ροή διαγράφεται από το μεταγωγέα.

- Διάρκεια αδράνειας της ροής (idle timeout) ως 0. Αν η τιμή του χρόνου που θέτουμε είναι *μη μηδενική*, αν η τρέχουσα ροή δεν έχει ταιριάξει (match) κανένα πακέτο μετά την πάροδο του χρόνου, τότε διαγράφεται από το μεταγωγέα.

- Θύρα εισόδου (in-port) του μεταγωγέα ως 1. Πρόκειται για τη θύρα του μεταγωγέα από την οποία εισέρχεται η κίνηση την οποία θέλουμε να αποκόψουμε.

- Τύπος ethernet ως 2048 (0x800). Η τιμή αυτή υποδεικνύει ότι θα ακολουθήσουν φίλτρα στη ροή που θα περιέχουν διευθύνσεις IP.

- Διεύθυνση αποστολής (ipv4-source) ως 147.102.40.69. Πρόκειται για τη διεύθυνση του δημόσιου δικτύου του RPi – αποστολέα, του οποίου και την εξερχόμενη προς το RPi – προορισμού θέλουμε να αποκόψουμε.

- Διεύθυνση προορισμού (ipv4-destination) ως 147.102.39.3. Πρόκειται για τη διεύθυνση του δημοσίου δικτύου του RPi – προορισμού.

- Αριθμός θύρας του μεταγωγέα στην οποία θα προωθηθεί η κίνηση που θα κάνει match τα ανωτέρω φίλτρα της ροής ως 5. Εδώ θέτουμε μια αριθμητική τιμή θύρας η οποία δεν υπάρχει στην κατασκευή του μεταγωγέα. Εμμέσως δηλαδή, η κίνηση που θα ταιριάξει τα ανωτέρω φίλτρα θα απορριφθεί (drop).

Η παραμετροποίηση της ροής σε αυτό το σημείο έχει ολοκληρωθεί. Πλέον είμαστε έτοιμοι να την προωθήσουμε στον OpenDaylight ελεγκτή μέσω του πρωτοκόλλου RESTCONF χρησιμοποιώντας την HTTP μέθοδο PUT.

5.2.3 Αποστολή των ροών στον ελεγκτή και προβολή αποτελεσμάτων

Θέτουμε πρώτα στο XML αρχείο τις ακόλουθες επικεφαλίδες:

- Τύπος εξουσιοδότησης (authorization): Basic Auth, με το κατάλληλο όνομα χρήστη (username) και κωδικό (password) για την απόκτηση πρόσβασης στον ελεγκτή

- Τύπος περιεχομένου (content-type): application/xml

- Αποδοχή (accept): application/xml

Στο σημείο αυτό, στέλνουμε με τη μέθοδο **PUT** στο configuration store του OpenDaylight ελεγκτή, τη ροή που δημιουργήθηκε πληκτρολογώντας το ακόλουθο URI:

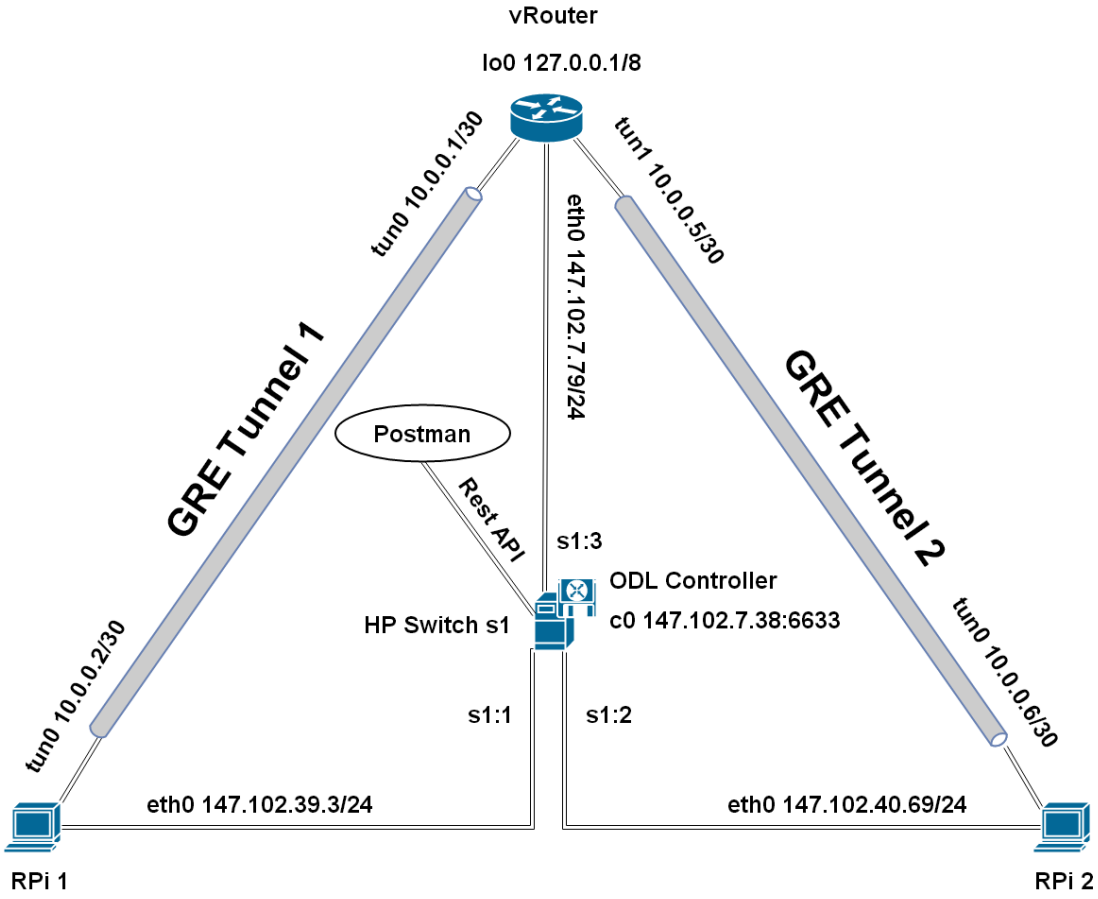
http://ulairi.telecom.ntua.gr:8181/restconf/config/opendaylight-inventory:nodes/node/openflow:1/table/0/flow/11?Accept=application/xml&Content-Type=application/xml

Μετά την απαραίτητη αναμονή για την ενημέρωση του δικτύου από τον ελεγκτή, χρησιμοποιούμε τη μέθοδο **GET** για να επιβεβαιώσουμε την ενημέρωση του operational store, ζητώντας από τον ελεγκτή να μας επιστρέψει τον πίνακα ροών του μεταγωγέα openflow:1. Αυτό επιτυγχάνεται πληκτρολογώντας το ακόλουθο URI:

http://ulairi.telecom.ntua.gr:8181/restconf/operational/opendaylight-inventory:nodes/node/openflow:1/table/0?Accept=application/xml&Content-Type=application/xml

Με τον ίδιο τρόπο δημιουργούμε και τη ροή με **όνομα flow22** για τον μεταγωγέα openflow;2 που βρίσκεται στην πλευρά του RPi – παραλήπτη. Η μόνη διαφορά μεταξύ των ροών είναι ότι σε αυτήν τη ροή, θέτουμε ως αποστολέα το RPi – παραλήπτη (147.102.39.3) κι ως παραλήπτη το RPi – αποστολέα (147.102.40.69).

Τα αρχεία XML με τις ροές που δημιουργήθηκαν καθώς και τα αποτελέσματα των HTTP μεθόδων GET παρατίθενται στο παράρτημα που βρίσκεται στο τέλος της εργασίας.



5.10 Τελική τοπολογία του VPN με SDN έλεγχο κίνησης

6. Μελλοντικές Επεκτάσεις

6.1 Προτάσεις

Η υλοποίηση της παρούσας διπλωματικής εργασίας προσφέρει μια νέα οπτική γωνία στην επικοινωνία απομακρυσμένων χρηστών με ασφαλή κι οικονομικό τρόπο αλλά και στην κεντρικοποιημένη διαχείριση της διακινούμενης πληροφορίας. Στη βιομηχανία των τηλεπικοινωνιών όμως, βρίσκεται ακόμη σε πειραματικό στάδιο κι επιδέχεται βελτιώσεις σε διάφορα σημεία της.

Μια σημαντική προσθήκη θα είναι να προστεθεί μια λειτουργία, με τη χρήση της οποίας θα μπορεί η ρύθμιση των συστατικών του VPN να γίνει με δυναμικό τρόπο. Αυτή τη στιγμή, η παραμετροποίηση τόσο των vCPEs όσο και του vRouter είναι στατική και γίνεται χειροκίνητα. Μια δυναμική προσαρμογή η οποία επεξεργάζεται αυτόματα κάθε κόμβο και τον προετοιμάζει για εισαγωγή στο δίκτυο θα επιταχύνει κατακόρυφα τη διαδικασία δημιουργίας και τροποποίησης του δικτύου.

Το ίδιο ισχύει και για το SDN δίκτυο. Αυτή τη στιγμή, ο έλεγχος των ροών από τον ελεγκτή γίνεται με στατικό τρόπο, καθώς οι ροές ελέγχονται χειροκίνητα από το διαχειριστή δικτύου μέσω του εργαλείου Postman και στη συνέχεια γίνεται το push προς τον ελεγκτή. Μια μελλοντική βελτιστοποίηση θα ήταν η δημιουργία ενός script, το οποίο θα αυτοματοποιεί τη διαδικασία δημιουργίας και επεξεργασίας των ροών δρομολόγησης καθώς και τη διάθεσή τους (push) στο επίπεδο ελέγχου του ελεγκτή του δικτύου και μετέπειτα ενημέρωση του επιπέδου δρομολόγησης.

Τέλος, θα ήταν σαφέστατα χρήσιμη η δημιουργία ενός GUI ώστε η έναρξη μιας συνεδρίας VPN να είναι ακόμη πιο εύχρηστη και να μπορεί να χρησιμοποιηθεί κι από λιγότερο έμπειρους χρήστες.

Σε γενικότερα πλαίσια η παρούσα εργασία ασχολείται με διάφορους τομείς εξειδίκευσης με αποτέλεσμα να μπορεί να προσεγγιστεί από διάφορες σκοπιές και να βελτιωθεί αντιστοίχως.

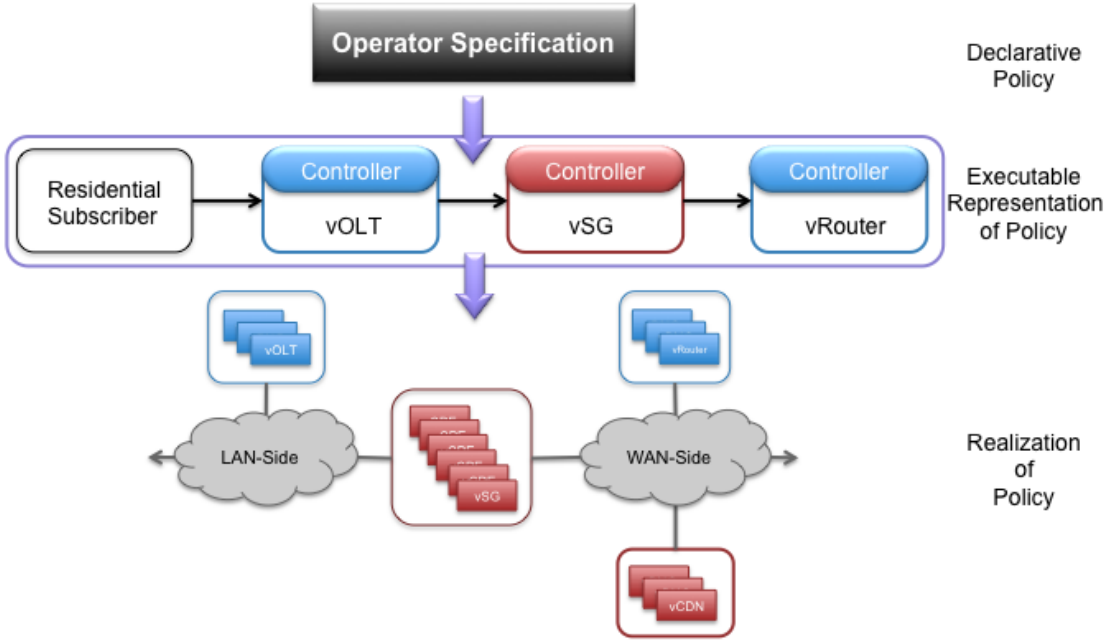
6.2 Business Solutions

Τέλος, αξίζει να γίνει αναφορά σε δυο προσεγγίσεις που έχουν γίνει πάνω στην εφαρμογή της παρούσας διπλωματικής εργασίας.

6.2.1 Central Office Re-architected as a Datacenter – CORD

Η **"άκρη" ενός επιχειρησιακού δικτύου**, για παράδειγμα το κεντρικό γραφείο για εταιρείες τηλεπικοινωνιών, είναι το σημείο στο οποίο η επιχείρηση συνδέεται με τους πελάτες. Το CORD είναι ένα project που στοχεύει στη μεταμόρφωση της "άκρης" του δικτύου σε μια ευκίνητη πλατφόρμα παροχής υπηρεσιών, η οποία δίνει τη δυνατότητα στο χειριστή του δικτύου να παρέχει την καλύτερη δυνατή εμπειρία στον τελικό χρήστη, εμπλουτισμένη με καινοτόμες υπηρεσίες επόμενης γενιάς.

Το CORD είναι μια πλατφόρμα που συνδυάζει τις τεχνολογίες SDN, NFV και Cloud για τη δημιουργία ευέλικτων κέντρων δεδομένων (datacenters) για την "άκρη" του δικτύου. Το CORD αναπτύσσεται υπό την αιγίδα του Open Networking Foundation, μιας πρωτοβουλίας πολλών μεγάλων εταιριών τηλεπικοινωνίας και πληροφορικής, όπως οι AT&T, Cisco, Ericsson, Google, Huawei, Intel, NEC, Nokia, Samsung, Verizon κ.α. Ενσωματώνοντας πολλαπλά projects ανοιχτού κώδικα, το CORD παρέχει μια ανοιχτή, προγραμματιζόμενη, ευέλικτη πλατφόρμα στο υπολογιστικό νέφος που δίνει τη δυνατότητα στους χειριστές δικτύου να δημιουργήσουν καινοτόμες υπηρεσίες. [36]



6.1 Αρχιτεκτονική CORD [37]

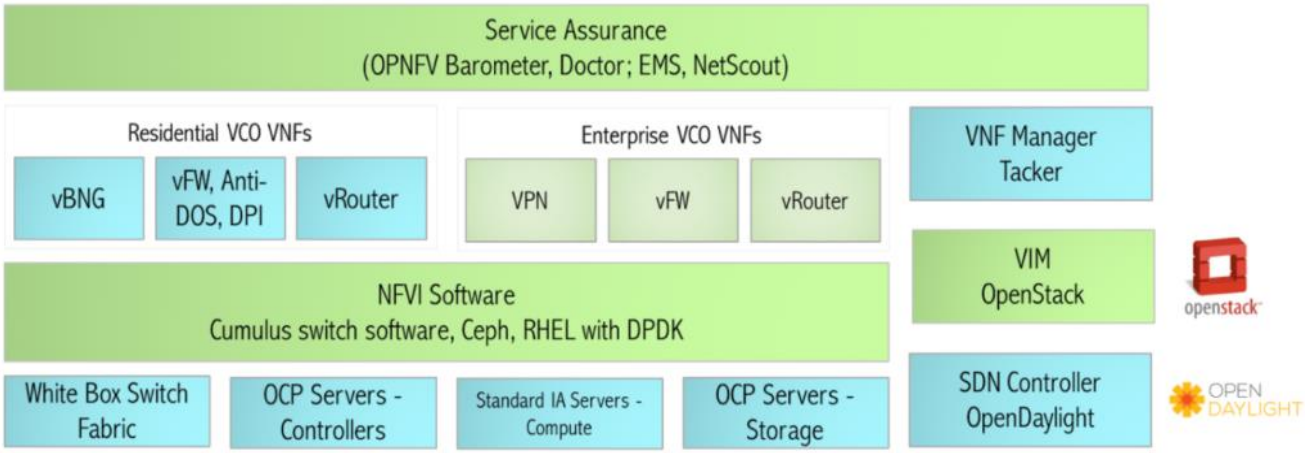
6.2.2 Virtual Central Office – VCO

Όπως αναφέραμε και πριν, τα κεντρικά γραφεία μιας επιχείρησης αποτελούν το σημείο που παρέχει στους πελάτες μαζική πρόσβαση στις προσφερόμενες δικτυακές υπηρεσίες. Μέχρι τώρα, οι παρεχόμενες αυτές δικτυακές υπηρεσίες στα κεντρικά γραφεία κατασκευάζονταν χρησιμοποιώντας πολύπλοκο, ειδικής κατασκευής εξοπλισμό που υλοποιεί μονάχα μια λειτουργία. Η προσθήκη νέων υπηρεσιών ισοδυναμούσε με την αγορά, ρύθμιση και διαχείριση νέου εξοπλισμού.

Το VCO είναι μια προσπάθεια που έγινε στα πλαίσια πρόσφατης συνόδου OPNFV (Open Platform for NFV), με σκοπό να επισημάνει τα αναφερθέντα αυτά προβλήματα και τον τρόπο με τον οποίο εξουδετερώνονται μέσω της εικονικοποίησης των κεντρικών γραφείων. Πρόκειται δηλαδή για μια ζωντανή επίδειξη εξακρίβωσης της δυνατότητας υλοποίησης (Proof of Concept) της εικονικοποίησης του κεντρικού γραφείου, βασιζόμενη στην υποδομή της NFV αρχιτεκτονικής, κατά τη διάρκεια της οποίας ένα παράρτημα της εταιρείας συνδέεται σε ένα σύνολο από επιχειρησιακές VNFs εντός vCPE.

Το OPNFV είναι ένα συνεργατικό project υπό την αιγίδα του Linux Foundation το οποίο μεταμορφώνει παγκόσμια δίκτυα μέσω τεχνολογιών NFV. Ορισμένες από τις εταιρείες που συμμετέχουν είναι οι Dell, Hewlett Packard, IBM, Juniper, Lenovo, Redhat, ZTE κ.α.

Ενέργειες σαν κι αυτές τονίζουν το αυξανόμενο κόστος της σύνδεσης απομακρυσμένων γραφείων, καθώς απαιτεί προκατασκευασμένο εξοπλισμό μιας χρήσης που έχει ανάγκη από χειροκίνητη ρύθμιση. Η εικονικοποίηση και επιλεκτική μετατροπή των λειτουργιών αυτών σε VNFs εντός του κεντρικού γραφείου βελτιώνει την ευελιξία και μειώνει τα χειριστικά έξοδα του δικτύου. [38]



6.2 Αρχιτεκτονική VCO [39]

Βιβλιογραφία

[1] “Virtualization”, https://en.wikipedia.org/wiki/Virtualization

[2] “How virtualization works”, http://searchservervirtualization.techtarget.com/definition/virtualization

[3] “Autonomic Computing”, https://en.wikipedia.org/wiki/Autonomic_computing

[4] “Utility Computing”, https://en.wikipedia.org/wiki/Utility_computing

[5] “Virtual Customer Premises Equipment”, http://searchsdn.techtarget.com/definition/vCPE-virtual-customer-premise-equipment

[6] “Network Function Virtualization”, https://en.wikipedia.org/wiki/Network_function_virtualization

[7] “Virtual Network Function”, http://network-functions-virtualization.com/mano.html

[8] “How a Managed Router Service Can be Deployed with NFV”, https://www.sdxcentral.com/nfv/definitions/whats-network-functions-virtualization-nfv/

[9] “NFV Service Chaining Orchestration for Any Network Topology & Architecture”, http://cloudify.co/2015/10/13/nfv-vnf-network-topology-architecture-automation-tosca-service-chaining-orchestration.html

[10] “What is NFV MANO?”, https://www.sdxcentral.com/nfv/definitions/nfv-mano/

[11] “The Journey to Network Functions Virtualization (NFV) Era”, http://www.informit.com/articles/article.aspx?p=2755705&seqNum=2

[12] “Software – Defined Networking”, https://en.wikipedia.org/wiki/Software-defined_networking

[13] “OpenFlow”, https://en.wikipedia.org/wiki/OpenFlow

[14] “OpenFlow Overview”, https://s3f.iti.illinois.edu/usrman/openflow.html

[15] “Mobile VPN”, http://leadz.tk/voxuk/mobile-vpn-204.php

[16] “Managing Site-to-Site VPNs”, http://www.cisco.com/c/en/us/td/docs/security/security_management/cisco_security_manager/security_manager/3-3/user/guide/CSMUserGuide_wrapper/vpchap.html

[17] “VPN Tunnel”, http://usatcorp.com/faqs/series-3-setup-ipsec-vpn-tunnel-capable-cradlepoint-router-sonicwall-tz-series-firewall/

[18] “Configuring a Site-to-Site GRE Tunnel”, http://tcpflag.blogspot.gr/2011/01/configuring-site-to-site-gre-tunnel.html

[19] “Cloud VPN Solution – Provider and Users on Cloud”, https://www.ncp-e.com/en/solutions/vpn/vpn-scenarios/cloud-vpn/

[20] “VyOS”, https://vyos.io/

[21] “Quagga Routing Suite”, http://www.nongnu.org/quagga/docs/quagga.html#SEC_Contents

[22] “Cisco Cloud Services Router 1000v Data Sheet”, http://www.cisco.com/c/en/us/products/collateral/routers/cloud-services-router-1000v-series/datasheet-c78-733443.html

[23] “Cisco CSR 1000v Use Cases”, http://www.cisco.com/c/en/us/products/collateral/routers/cloud-services-router-1000v-series/white-paper-c11-736574.html

[24] “Raspberry Pi”, https://en.wikipedia.org/wiki/Raspberry_Pi

[25] “Raspbian”, https://www.raspbian.org/

[26] “Raspberry Pi Model B”, https://www.raspberrypi.org/forums/viewtopic.php?t=129525&p=866050

[27] “What is an OpenDaylight Controller?”, https://www.sdxcentral.com/sdn/definitions/sdn-controllers/opendaylight-controller/

[28] “OpenDaylight Boron”, https://www.opendaylight.org/odlboron

[29] “OpenDaylight Controller: MD-SAL Developers’ Guide”, http://docs.inocybe.com/dev-guide/content/_controller.html

[30] Jan Medved, Robert Varga, Anton Tkacik, Ken Gray, “OpenDaylight: Towards a Model-Driven SDN Controller Architecture”, 2014

[31] “Representational State Transfer”, https://en.wikipedia.org/wiki/Representational_state_transfer

[32] Dennis Heimbigner, “A REST Design Methodology”

[33] “HP 2920 Switch Series”, http://h17007.www1.hpe.com/us/en/products/switches/HP_2920_Switch_Series/index.aspx?jumpid=reg_r1002_usen_c-001_title_r0001#.WJSyoH-o3IU

[34] “API Testing with Postman”, https://seesparkbox.com/foundry/api_testing_with_postman

[35] “Postman”, https://www.getpostman.com/

[36] “CORD: Reinventing Central Offices for Efficiency & Agility”, http://opencord.org/

[37] “Service Assembly and Composition in CORD”, https://wiki.opencord.org/pages/viewpage.action?pageId=1278081

[38] “Virtual Central Office: A demonstration of pure community effort”, http://verticalindustriesblog.redhat.com/virtual-central-office-a-demonstration-of-pure-community-effort/

[39] “OPNFV Virtual Central Office”, https://www.opnfv.org/resources/virtual-central-office

Παράρτημα

Εγκατάσταση διόδου μεταξύ Raspberry Pi και VyOS vRouter

// πριν από κάθε εντολή τρέχουμε σε superuser για την απόκτηση των απαραίτητων δικαιωμάτων, γράφοντας sudo

// δημιουργία GRE tunnel προς VyOS vRouter
// ζεύξη GRE: 10.0.0.0/30
pi@raspberrypi ~ $ ip tunnel add tun0 mode gre local 147.102.40.69 remote 147.102.7.79
pi@raspberrypi ~ $ ip link set dev tun0 up
pi@raspberrypi ~ $ ip addr add 10.0.0.2 dev tun0 peer 10.0.0.1/30

// έλεγχος status tunnel
pi@raspberrypi ~ $ netstat -rn

Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
0.0.0.0	147.102.40.200	0.0.0.0	UG	0	0	0	eth0
10.0.0.0	0.0.0.0	255.255.255.252	U	0	0	0	tun0
147.102.40.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0

//επικοινωνία με VyOS vRouter μέσω tunnel
pi@raspberrypi ~ $ ping 10.0.0.1 -c4

PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_req=1 ttl=64 time=1.11 ms
64 bytes from 10.0.0.1: icmp_req=2 ttl=64 time=1.02 ms
64 bytes from 10.0.0.1: icmp_req=3 ttl=64 time=0.942 ms
64 bytes from 10.0.0.1: icmp_req=4 ttl=64 time=1.02 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 0.942/1.025/1.115/0.072 ms

Παραμετροποίηση εικονικού δρομολογητή VyOS

//λίστα εντολών

giannis@vyos:~$ configure

//δημιουργία χώρου για το GRE header
giannis@vyos:~$ set policy route CHANGE-MSS rule 1 set tcp-mss 1360
giannis@vyos:~$ set policy route change-mss rule 1 protocol tcp
giannis@vyos:~$ set policy route change-mss rule 1 tcp flags SYN

//δημιουργία tunnel
giannis@vyos:~$ set interfaces tunnel tun0 encapsulation gre
giannis@vyos:~$ set interfaces tunnel tun0 address 10.0.0.1/30
giannis@vyos:~$ set interfaces tunnel tun0 mtu 1400
giannis@vyos:~$ set interfaces tunnel tun0 policy route CHANGE-MSS
giannis@vyos:~$ set interfaces tunnel tun0 local-ip 147.102.7.79
giannis@vyos:~$ set interfaces tunnel tun0 remote-ip 147.102.40.69

giannis@vyos:~$ commit
giannis@vyos:~$ save

giannis@vyos:~$ show configuration //για το interface του tunnel

```
interfaces {
 tunnel tun0 {
      address 10.0.0.1/30
      encapsulation gre
      local-ip 147.102.7.79
      mtu 1400
      parameters {
         ip {
            ttl 64
         }
      }
      policy {
         route CHANGE-MSS
      }
      remote-ip 147.102.40.69
 }
}
policy {
   access-list 100 {
         ip {
            ttl 64
         }
      }
      policy {
         route CHANGE-MSS
      }
      remote-ip 147.102.40.69
   }
   route CHANGE-MSS {
      rule 1 {
         protocol tcp
         set {
            tcp-mss 1360
         }
         tcp {
            flags SYN
         }
      }
   }
}
```

//έλεγχος
giannis@vyos:~$ show interfaces tunnel

Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface IP Address S/L Description
--- --- --- ---
tun0 10.0.0.1/30 u/u

//Μετά το setup και στο RPi, μπορούμε να κάνουμε και ping προς αυτό.
giannis@vyos:~$ ping 10.0.0.2 count 4
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=0.910 ms
64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=0.956 ms
64 bytes from 10.0.0.2: icmp_req=3 ttl=64 time=0.876 ms
64 bytes from 10.0.0.2: icmp_req=4 ttl=64 time=0.857 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3001ms
rtt min/avg/max/mdev = 0.857/0.899/0.956/0.052 ms

Εγκατάσταση διόδου μεταξύ Raspberry Pi και CSR 1000v vRouter

// πριν από κάθε εντολή τρέχουμε σε superuser για την απόκτηση των απαραίτητων δικαιωμάτων, γράφοντας sudo

// δημιουργία GRE tunnel προς Cisco vRouter
// ζεύξη GRE: 10.0.0.4/30
pi@raspberrypi ~ $ ip tunnel add tun1 mode gre local 147.102.40.69 remote 147.102.7.82
pi@raspberrypi ~ $ ip link set dev tun1 up
pi@raspberrypi ~ $ ip addr add 10.0.0.6 dev tun1 peer 10.0.0.5/30

// έλεγχος status tunnel
pi@raspberrypi ~ $ netstat -rn

Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
0.0.0.0	147.102.40.200	0.0.0.0	UG	0	0	0	eth0
10.0.0.0	0.0.0.0	255.255.255.252	U	0	0	0	tun0
10.0.0.4	0.0.0.0	255.255.255.252	U	0	0	0	tun1
147.102.40.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0

//επικοινωνία με Cisco vRouter μέσω tunnel
pi@raspberrypi ~ $ ping 10.0.0.5 -c4

PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_req=1 ttl=255 time=3.55 ms
64 bytes from 10.0.0.5: icmp_req=2 ttl=255 time=3.02 ms
64 bytes from 10.0.0.5: icmp_req=3 ttl=255 time=1.92 ms
64 bytes from 10.0.0.5: icmp_req=4 ttl=255 time=3.09 ms

--- 10.0.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 1.929/2.899/3.555/0.596 ms

Παραμετροποίηση εικονικού δρομολογητή CSR 1000v

// λίστα εντολών
R_vpn# configure terminal

// δημιουργία GRE tunnel με RPi
R_vpn(config)# interface tunnel 0
R_vpn(config-if)# ip address 10.0.0.5 255.255.255.252

// όπως και στο VyOS vRouter, δίνουμε μικρότερο MTU για να χωρέσει και το overhead του GRE.
R_vpn(config-if)# ip mtu 1400
R_vpn(config-if)# ip tcp adjust-mss 1360
R_vpn(config-if)# tunnel source 147.102.7.82
R_vpn(config-if)# tunnel destination 147.102.40.69

R_vpn(config-if)# end

// έλεγχος διεπαφής
R_vpn# show interfaces tunnel0

```
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 10.0.0.5/30
  MTU 9976 bytes, BW 100 Kbit/sec, DLY 50000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel linestate evaluation up
  Tunnel source 147.102.7.82, destination 147.102.40.69
  Tunnel protocol/transport GRE/IP
    Key disabled, sequencing disabled
    Checksumming of packets disabled
  Tunnel TTL 255, Fast tunneling enabled
  Tunnel transport MTU 1476 bytes
  Tunnel transmit bandwidth 8000 (kbps)
  Tunnel receive bandwidth 8000 (kbps)
  Last input never, output never, output hang never
  Last clearing of "show interface" counters 2d02h
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     30 packets input, 2756 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     27 packets output, 3236 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
```

// ping προς το RPi μέσω του GRE Tunnel σε 1 hop
R_vpn# ping 10.0.0.6

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/10/20 ms
```

Ροή δρομολόγησης ‘flow11’ που αποκόπτει την εξερχόμενη κίνηση από το switch ‘openflow:1’ (Αρχείο XML)

/*
Τι προσέχουμε:

- βάζουμε υψηλό priority ώστε να ελέγχεται το flow μας πάντα πρώτο
- βάζουμε μηδενική τιμή στα πεδία "hard timeout" και "idle timeout" για να μη διαγραφεί ποτέ το flow entry
- επιλέγουμε την κατάλληλη θύρα εισόδου για το matching
- επιλέγουμε τον κατάλληλο τύπο πακέτων για το ethernet matching (0x800)
- για να γίνει drop η κίνηση, επιλέγουμε ως θύρα εξόδου μια μη υπαρκτή

πως προσθέτουμε ένα flow entry:

- χρησιμοποιούμε κατά προτίμηση τον ReST client "Postman"
- θέτουμε την τιμή "application/xml" στις παραμέτρους "Accept" και "Content-Type"
- θέτουμε "Basic Auth" για την ταυτοποίηση των στοιχείων κατά το αίτημα εισαγωγής νέου flow entry
- κάνουμε PUT το flow μας σε μορφή XML αρχείου στο κατάλληλο path εντός του MD-SAL config store του controller:

http://ulairi.telecom.ntua.gr:8181/restconf/config/opendaylight-inventory:nodes/node/openflow:1/table/0/flow/11?Accept=application/xml&Content-Type=application/xml

Μετά τον έλεγχο για συμβατότητα από τον controller, αυτός παίρνει το flow από το config store και το κάνει push στο MD-SAL operational store του αντίστοιχου switch.
Κάνω υπομονή και περιμένω μέχρι η αλλαγή που ζήτησα να γίνει reflected στο δίκτυο (μπορεί το push να πάρει λίγη ώρα μέχρι να περάσει).

Για να επιβεβαιώσουμε την αλλαγή στο δίκτυο, κάνουμε GET στο operational store του κατάλληλου switch:

http://ulairi.telecom.ntua.gr:8181/restconf/operational/opendaylight-inventory:nodes/node/openflow:1/table/0/flow/11?Accept=application/xml&Content-Type=application/xml

Η έξοδος αυτού του request είναι οι λεπτομέρειες του flow entry που μόλις προστέθηκε.

Ακολουθεί το περιεχόμενο του αρχείου XML για το flow11.
*/

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<flow xmlns="urn:opendaylight:flow:inventory">
   <strict>false</strict>
   <flow-name>flow11</flow-name>
   <table_id>0</table_id>
   <id>11</id>
   <cookie_mask>255</cookie_mask>
   <cookie>101</cookie>
   <priority>311</priority>
   <hard-timeout>0</hard-timeout>
   <idle-timeout>0</idle-timeout>
   <installHw>false</installHw>
   <match>
         <in-port>1</in-port>
      <ethernet-match>
         <ethernet-type>
            <type>2048</type>
         </ethernet-type>
      </ethernet-match>
      <ipv4-source>10.10.1.1/32</ipv4-source>
      <ipv4-destination>10.10.1.2/32</ipv4-destination>
   </match>
   <instructions>
      <instruction>
         <order>0</order>
         <apply-actions>
            <action>
               <order>0</order>
               <output-action>
                  <output-node-connector>5</output-node-connector>
                  <max-length>60</max-length>
               </output-action>
            </action>
         </apply-actions>
      </instruction>
   </instructions>
</flow>
```

Ροή δρομολόγησης ‘flow12’ που αποκόπτει την εξερχόμενη κίνηση από το switch ‘openflow:2’ (Αρχείο XML)

/*
Τι προσέχουμε:

- βάζουμε υψηλό priority ώστε να ελέγχεται το flow μας πάντα πρώτο
- βάζουμε μηδενική τιμή στα πεδία "hard timeout" και "idle timeout" για να μη διαγραφεί ποτέ το flow entry
- επιλέγουμε την κατάλληλη θύρα εισόδου για το matching
- επιλέγουμε τον κατάλληλο τύπο πακέτων για το ethernet matching (0x800)
- για να γίνει drop η κίνηση, επιλέγουμε ως θύρα εξόδου μια μη υπαρκτή

πως προσθέτουμε ένα flow entry:

- χρησιμοποιούμε κατά προτίμηση τον ReST client "Postman"
- θέτουμε την τιμή "application/xml" στις παραμέτρους "Accept" και "Content-Type"
- θέτουμε "Basic Auth" για την ταυτοποίηση των στοιχείων κατά το αίτημα εισαγωγής νέου flow entry
- κάνουμε PUT το flow μας σε μορφή XML αρχείου στο κατάλληλο path εντός του MD-SAL config store του controller:

  http://ulairi.telecom.ntua.gr:8181/restconf/config/opendaylight-inventory:nodes/node/openflow:2/table/0/flow/12?Accept=application/xml&Content-Type=application/xml

Μετά τον έλεγχο για συμβατότητα από τον controller, αυτός παίρνει το flow από το config store και το κάνει push στο MD-SAL operational store του αντίστοιχου switch. Κάνω υπομονή και περιμένω μέχρι η αλλαγή που ζήτησα να γίνει reflected στο δίκτυο (μπορεί το push να πάρει λίγη ώρα μέχρι να περάσει).

Για να επιβεβαιώσουμε την αλλαγή στο δίκτυο, κάνουμε GET στο operational store του κατάλληλου switch:

http://ulairi.telecom.ntua.gr:8181/restconf/operational/opendaylight-inventory:nodes/node/openflow:2/table/0/flow/12?Accept=application/xml&Content-Type=application/xml

Η έξοδος αυτού του request είναι οι λεπτομέρειες του flow entry που μόλις προστέθηκε.

Ακολουθεί το περιεχόμενο του αρχείου XML για το flow12.
*/

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<flow xmlns="urn:opendaylight:flow:inventory">
   <strict>false</strict>
   <flow-name>flow12</flow-name>
   <table_id>0</table_id>
   <id>12</id>
   <cookie_mask>255</cookie_mask>
   <cookie>101</cookie>
   <priority>312</priority>
   <hard-timeout>0</hard-timeout>
   <idle-timeout>0</idle-timeout>
   <installHw>false</installHw>
   <match>
         <in-port>1</in-port>
      <ethernet-match>
         <ethernet-type>
            <type>2048</type>
         </ethernet-type>
      </ethernet-match>
      <ipv4-source>10.10.1.2/32</ipv4-source>
      <ipv4-destination>10.10.1.1/32</ipv4-destination>
   </match>
   <instructions>
      <instruction>
         <order>0</order>
         <apply-actions>
            <action>
               <order>0</order>
               <output-action>
                  <output-node-connector>5</output-node-connector>
                  <max-length>60</max-length>
               </output-action>
            </action>
         </apply-actions>
      </instruction>
   </instructions>
</flow>
```