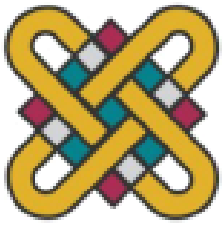
Πανεπιστήμιο Δυτικής Μακεδονίας

Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών



ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δρομολόγηση Κίνησης Κρίσιμων Υποδομών στις Παρυφές του Δικτύου

Οικονόμου Σπυρίδων – Αριθμός Μητρώου 1199

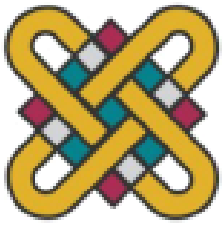
Ιούλιος 2022

Επιβλέπων Καθηγητής: Παναγιώτης Σαρηγιαννίδης

University of Western Macedonia

Polytechnic School

Department of Electrical & Computer Engineering



DIPLOMA THESIS

Routing of Critical Infrastructure Traffic at the Network Edge

Oikonomou Spyridon – Student ID 1199

July 2022

Supervisor: Panagiotis Sarigiannidis

Δήλωση Πνευματικών Δικαιωμάτων

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο:

"Δρομολόγηση Κίνησης Κρίσιμων Υποδομών στις Παρυφές του Δικτύου / Routing of Critical Infrastructure Traffic at the Network Edge"

καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν,
και η οποία έχει εκπονηθεί στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Παναγιώτη Σαρηγιαννίδη
αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.
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 μόνο.

Υπογραφή Φοιτητή:

Ευχαριστίες

Θα ήθελα να ευχαριστώ τους καθηγητές και τους συμφοιτητές μου για τα τελευταία πέντε χρόνια ακόμα και αν δεν τελείωσαν με τις ιδανικότερες συνθήκες.
Επιπλέον, θα ήθελα να ευχαριστήσω την οικογένεια και τους φίλους μου για την κατανόηση και την υποστήριξή τους.
Τέλος, θα ήθελα να ευχαριστήσω τον επιβλέποντα της διπλωματικής μου εργασίας, κύριο Παναγιώτη Σαρηγιαννίδη, για την καθοδήγηση και τις υποδείξεις που μου παρείχε, όπως και τους υποψήφιους διδάκτορες Θανάση Λιατίφη και Δημήτρη Πλιάτσιο για την πολύτιμη βοήθεια τους.

Περίληψη

Η συνεχής ανάπτυξη νέων εφαρμογών και υπηρεσιών τα τελευταία χρόνια, οδηγούν σε αύξηση των δικτυακών απαιτήσεων σε ταχύτητα, χωρητικότητα, και εύρος ζώνης. Ως αποτέλεσμα, η παροχή υψηλού επιπέδου ποιότητας υπηρεσιών (Quality of Service – QoS) καθώς και η αξιόπιστη παροχή συνδεσιμότητας εισάγουν σημαντικές προκλήσεις, ιδιαίτερα σε δίκτυα κρίσιμων υποδομών, όπως τα δίκτυα ηλεκτρικής ενέργειας. Προς το σκοπό αυτό, προτείνεται η αξιοποίηση των Δικτύων Καθορισμένα από Λογισμικό (Software Defined Networking - SDN), η οποία στοχεύει στο διαχωρισμό του επιπέδου ελέγχου από το επίπεδο δεδομένων. Στην παρούσα διπλωματική εργασία αξιολογήθηκαν υπάρχουσες λύσεις και αναπτύχθηκε μια υλοποίηση η οποία στοχεύει να μειώσει την καθυστέρηση σε δίκτυα που εξυπηρετούν κρίσιμες υποδομές. Συγκεκριμένα, η μέθοδος εστιάζει, αρχικά, στην αναγνώριση της τοπολογίας του δικτύου και την εύρεση όλων των διαθέσιμων μονοπατιών, με τον γρηγορότερο τρόπο και, στη συνέχεια, στην επιλογή του καλύτερου με στόχο τη βελτίωση της ποιότητας των υπηρεσιών του δικτύου. Η αναγνώριση της τοπολογίας του δικτύου και η εύρεση των διαθέσιμων διαδρομών πραγματοποιείται με την χρήση του αλγορίθμου αναζήτησης κατά βάθος (Depth First Search – DFS) και, με κριτήριο το εύρος ζώνης (bandwidth), επιλέγονται οι διαδρομές από τις οποίες θα μεταφερθεί η κρίσιμη κίνηση του δικτύου, ελαχιστοποιώντας τις απώλειες και καθυστερήσεις. Τέλος, η κρίσιμη κίνηση του δικτύου δρομολογείται ιεραρχικά με την χρήση των Διαφοροποιημένων Υπηρεσιών (Differentiated Services Code Point - DSCP), επιτυγχάνοντας καλύτερη ποιότητα υπηρεσιών.

Abstract

The continuous development of new applications and services over the last years leads to increased network requirements in terms of data rates, capacity and bandwidth. As a result, in order to provide a high level of Quality of Service (QoS) and reliable connectivity several challenges have to be addressed, especially in critical infrastructure networks, such as the electrical power grid. To this end, the use of Software Defined Networking (SDN) technology is suggested as a promising asset for addressing the challenge of differentiating the control and data layers. In this thesis, existing SDN-based solutions are evaluated, and a new method is developed, which aims to reduce the backlog of critical infrastructure networks. In more detail, the method is focused on identifying the network topology, determining all available routes and then, selecting the optimal route, aiming to improve the quality of network services. The topology identification is implemented using the Depth First Search (DFS) algorithm, while the optimal route is selected based on the available bandwidth, thereby mitigating the losses and delays. Finally, the critical network traffic is routed hierarchically using the Differentiated Services Code Point (DSCP), thus achieving a higher better QoS.

Περιεχόμενα

Πίνακας συντομογραφιών

AF	Assured Forwarding
API	Application Programming Interface
ARP	Address Resolution Protocol
BFS	Breadth-first search
CBWFQ	Class-Based Weighted Fair Queueing
CDPI	Control-Data Plane Interface
CU	Control Unit
DDoS	Distributed Denial-of-Service
DF	Default Forwarding
DFS	Depth First Search
DiffServ	Differentiated services
DS	Differentiated Services
DSCP	Differentiated Services Code Point
ECN	Explicit Congestion Notification
EF	Expedited Forwarding
ETSI	European Telecommunications Standards Institute
IETF	Internet Engineering Task Force
IntServ	Integrated Services
IP	Internet Protocol
LLQ	Low Latency Queueing
MAC	Medium Access Control
NBI	Northbound Interface
NFV	Network Function Virtualization
NFVI	Network functions virtualization Infrastructure
ONF	Open Networking Foundation
OSI model	Open Systems Interconnection model
OSPF	Open Shortest Path First
OVS	Open vSwitch
PHB	Per-Hop Behavior
QoS	Quality of Service
RFC	Request for Comments
RTT	Round Trip Time
RSPEC	Ruby Specification
RSVP	Resource Reservation Protocol
SBI	Southbound Interface
SDN	Software-Defined Networking
TCP	Transmission Control Protocol
TSPEC	Traffic Specification
VM	Virtual Machine
VNF	Virtual Network Function
WRED	Weighted Random Early Detection

Λίστα εικόνων

Λίστα πινάκων

1. Εισαγωγή

Η παρούσα διπλωματική εργασία προτείνει τρόπους βελτίωσης της ποιότητας των υπηρεσιών και της δρομολόγησης της κίνησης σε κρίσιμες υποδομές με την χρήση Δικτύων Καθορισμένα από Λογισμικό (Software Defined Networking - SDN). Αρχικά, θα γίνει ανάλυση υπαρχουσών λύσεων των προκλήσεων που τις χαρακτηρίζουν. Έπειτα, μετά την παρουσίαση των στόχων της διπλωματικής εργασίας, θα γίνει ανάλυση του θεωρητικού υποβάθρου που απαιτείται με σκοπό την εκτέλεση της υλοποίησης. Στην συνέχεια, παρουσιάζεται η προτεινόμενη υλοποίηση που στοχεύει στη μείωση της καθυστέρησης σε δίκτυα που εξυπηρετούν κρίσιμες υποδομές και στη βελτίωση της ποιότητας των υπηρεσιών αυτών. Τέλος, παρουσιάζονται τα συμπεράσματα της διπλωματικής όπως και οι δυνατότητες μελλοντικής επέκτασης.

1.1 Σχετικές Υλοποιήσεις

Κατά την διαδικασία έρευνας την παρούσα διπλωματική εργασία έγινε μελέτη μεγάλου αριθμού εργασιών προκειμένου να χτιστεί το βαθύτερο θεωρητικό υπόβαθρο σχετικά με τις διάφορες έννοιες που περιγράφονται. Κατά την αναζήτηση αυτή ξεχώρισαν οι παρακάτω 3 εργασίες ως οι ιδανικότερες ως εναρκτήριες βάσεις για την υλοποίηση της διπλωματικής, κυρίως στον άξονα των αλγορίθμων, όπως και τι προϋποθέσεις θα πληροί το τελικό δίκτυο. Παρακάτω, ακολουθεί η σύνοψη αυτών:

Η πειραματική εργασία με τίτλο “SDN-Routing” [1], έχει σαν στόχο την προσομοίωση της δρομολόγησης κίνησης σε δίκτυο με την χρήση του SDN και του Openflow. Πιο συγκεκριμένα, το Openflow είναι πρωτόκολλο δικτύου σχεδιασμένο να διαχειρίζεται και να κατευθύνει την κίνηση μεταξύ δρομολογητών και μεταγωγέων [2]. Επίσης, χρησιμοποιείται το TestNet για την δημιουργία και την δοκιμή μικρών δικτύων με καθορισμένα βάρη μεταξύ κάθε μεταγωγέα. Στην συνέχεια, με την χρήση του αλγορίθμου Dijkstra, υπολογίζονται τα μονοπάτια με το χαμηλότερο κόστος και χρησιμοποιούνται τα αποτελέσματα για την εισαγωγή εγγραφών στον πίνακα ροών. Από την συγκεκριμένη υλοποίηση, ως προς το κομμάτι των αλγορίθμων, θα γίνει διερεύνηση για την χρήση του αλγορίθμου Dijkstra όπως και τι εναλλακτικοί αλγόριθμοι υπάρχουν για την εύρεση των διαθέσιμων μονοπατιών με τον γρηγορότερο και κατ’ επέκταση και τον βέλτιστο τρόπο. Τα αρνητικά που φαίνονται από την πρώτη ανάγνωση είναι ότι στην παρούσα υλοποίηση, η τοπολογία του δικτύου δεν ανανεώνεται αυτόματα σε τυχόν αλλαγές του δικτύου και οι τιμές των βαρών είναι προκαθορισμένες, συνεπώς το δίκτυο δεν είναι δυναμικό, κάτι που το καθιστά ευάλωτο στις αλλαγές. Τα αποτελέσματα αυτής της έρευνας ως προς την απόδοση του αλγορίθμου Dijkstra θα χρησιμοποιηθούν για σύγκριση αυτού με άλλους αλγορίθμους στην συνέχεια της εργασίας, όπου και θα εξαχθούν συμπεράσματα για αυτούς.

Ανάλογη προσέγγιση με την προηγούμενη είναι και η [3] με τίτλο “Dijkstra-SDN-Ryu”, στην όποια εφαρμόζεται ο αλγόριθμος Dijkstra για την εύρεση των συντομότερων διαδρομών. Στην συνέχεια, χρησιμοποιούνται σαν παραδείγματα 3 διαφορετικές τοπολογίες και για τον υπολογισμό βαρών της κάθε διαδρομής χρησιμοποιείται ο αλγόριθμος Open Shortest Path First (OSPF), ο οποίος χρησιμοποιεί αποκλειστικά το εύρος ζώνης. Τέλος, μεταφέρονται τα αποτελέσματα σε σταθερό πίνακα ο οποίος έχει οριστεί εξαρχής. Παρόλου που η συγκεκριμένη υλοποίηση [3] προσθέτει τον υπολογισμό βαρών δυναμικά, τα αρνητικά της προηγούμενης υλοποίησης όπως η μη ανανέωση του δικτύου σε τυχόν αλλαγές και η χρήση του αλγορίθμου Dijkstra για την εύρεση των μονοπατιών, παραμένουν.

Στην παρακάτω προσέγγιση με τίτλο “Controllable Network Architecture Based on SDN” [4], η προσέγγιση είναι πιο θεωρητική, και στοχεύει στην παρακολούθηση, την κατανομή και την διαχείριση της κίνησης σε αρχιτεκτονική βασισμένη σε SDN. Πιο αναλυτικά, ορίζει μια αρχιτεκτονική “Q”, η οποία είναι παρόμοια με του SDN, βρίσκει την κατάσταση στην οποία βρίσκονται οι κόμβοι του δικτύου και την κατάσταση των συνδέσεων μεταξύ τους και στέλνει μέρος της κίνησης από τα διαθέσιμα μονοπάτια. Με αυτό τον τρόπο, εξάγει το συμπέρασμα ότι η κάθε ροή προωθείται σύμφωνα με τα διαθέσιμα μονοπάτια χωρίς να υπάρχει υπερκάλυψη από κάποιο άλλο. Όπως είναι φανερό, στην συγκεκριμένη έρευνα, τα αρνητικά είναι ότι δεν μελετάται η εύρεση της τοπολογίας με τον βέλτιστο τρόπο ούτε ο υπολογισμός βαρών για τους κόμβους αλλά χρησιμοποιείται μια διαφορετική προσέγγιση βασισμένη στον διαχωρισμό της κίνησης για την πλήρη κάλυψη όλων των διαθέσιμων συνδέσεων.

Όλες οι παραπάνω λύσεις παρουσιάζουν κάποια θετικά στοιχεία τα οποία θα γίνει προσπάθεια χρήσης και κατάλληλης αξιοποίησης στην προτεινόμενη λύση. Τα αποτελέσματα της σύγκρισης παρουσιάζονται και αναλύονται στο τέταρτο κεφάλαιο. Στην επόμενη υποενότητα ακολουθούν οι στόχοι που τέθηκαν στην παρούσα διπλωματική εργασία.

1.2 Στόχοι Διπλωματικής Εργασίας

Ο στόχος της συγκεκριμένης διπλωματικής εργασίας είναι η ανάπτυξη μιας νέας μεθόδου η οποία θα στοχεύει στην μείωση της καθυστέρησης σε δίκτυα που εξυπηρετούν κρίσιμες υποδομές. Πιο συγκεκριμένα, αξιοποιώντας τα υπάρχοντα δεδομένα σύγκρισης μεταξύ των πιο διαδεδομένων αλγορίθμων με σκοπό την εύρεση του βέλτιστου για την εύρεση όλων των διαθέσιμων μονοπατιών με τον γρηγορότερο δυνατό τρόπο. Στην συνέχεια, θα χρησιμοποιηθεί μια μέθοδος για την εύρεση των βέλτιστων διαδρομών εντός του δικτύου. Προς το σκοπό αυτό, η οποιαδήποτε μεταβολή στο δίκτυο θα πρέπει να γίνεται άμεσα αντιληπτή, με στόχο τον δυναμικό χαρακτήρα της υπάρχουσας τοπολογίας. Με τους παραπάνω τρόπους, θα επιτευχθεί η εξισορρόπηση του δικτυακού φορτίου με αποδοτικό τρόπο καθώς, μέσω πολλαπλών διαδρομών, θα διαμοιράζεται η δικτυακή κίνηση χωρίς απώλειες. Τέλος,

θα υλοποιηθεί η αποτελεσματική εξασφάλιση της ποιότητας των υπηρεσιών, με την χρήση μηχανισμών ή τεχνολογιών για τον έλεγχο της κυκλοφορίας με στόχο τη διασφάλιση της απόδοσης των κρίσιμων εφαρμογών.

1.3 Δομή Διπλωματικής Εργασίας

Στο πρώτο κεφάλαιο παρουσιάστηκαν οι σχετικές μελέτες που έχουν πραγματοποιηθεί, τα προβλήματα που χαρακτηρίζουν την κάθε μια, αλλά και τους στόχους της παρούσας εργασίας. Το περιεχόμενο των επόμενων κεφαλαίων διαμορφώνεται ως ακολούθως:

Στο δεύτερο κεφάλαιο περιγράφονται αναλυτικά οι τεχνολογίες που αφορούν το SDN και παρουσιάζεται η σχετική βιβλιογραφία. Πέραν της βιβλιογραφικής επισκόπησης σχετικά με την εφαρμογή τεχνολογιών SDN σε δίκτυα επικοινωνίας, παρουσιάζονται η αρχιτεκτονική και η δομή του SDN. Επίσης, παρατίθενται τα πλεονεκτήματα του SDN έναντι των παραδοσιακών τεχνολογιών δικτύωσης και οι σχετικές προκλήσεις. Επιπροσθέτως, παρουσιάζεται η εικονικοποίηση λειτουργιών δικτύου (Network Function Virtualization - NFV), η οποία αποτελεί μέθοδο μετάβασης των παραδοσιακών λειτουργιών του δικτύου από υλικό σε εικονικές μηχανές. Αναλύονται τα οφέλη της καθώς και η μέθοδος με την οποία θα μπορούσε να αξιοποιηθεί. Ακόμα, παρουσιάζονται οι αντίστοιχες προκλήσεις σχετικά με την χρήση τους. Στην συνέχεια, παρατίθεται το θεωρητικό υπόβαθρο σχετικά με την ποιότητα των υπηρεσιών και την δρομολόγηση της κίνησης εντός του δικτύου. Συγκεκριμένα, γίνεται διεξοδική ανάλυση των Ενσωματωμένων Υπηρεσιών (Integrated Services - IntServ) και των Διαφοροποιημένων Υπηρεσιών (Differentiated Services - DSCP). Το κεφάλαιο ολοκληρώνεται με την ανάλυση της κεντρικής ιδέας η οποία θα απασχολήσει και το τρίτο κεφάλαιο.

Το τρίτο κεφάλαιο αφορά στο πρακτικό-πειραματικό μέρος της διπλωματικής εργασίας. Αρχικά, εξετάζεται η χρήση του Ryu Ελεγκτή, του Mininet και του Oracle VM Virtualbox. Στην συνέχεια, παρουσιάζεται ένας οδηγός σχετικός με την μέθοδο που ακολουθήθηκε για τη διπλωματική, η μοντελοποίηση της και τέλος, η υλοποίηση των απαραίτητων αλγορίθμων και εργαλείων με τα ακόλουθα αποτελέσματα.

Τέλος, στο τέταρτο κεφάλαιο παρουσιάζονται τα συμπεράσματα που μπορούν να εξαχθούν από την πρακτική εφαρμογή καθώς και την σύγκριση με άλλες προσεγγίσεις. Έπειτα, παρουσιάζεται μια σύνοψη της εργασίας και το κεφάλαιο ολοκληρώνεται με πιθανές ερευνητικές επεκτάσεις που θα μπορούσαν να πραγματοποιηθούν μελλοντικά.

2. Θεωρητικό Υπόβαθρο

Σε αυτή την ενότητα αναλύονται όλες οι βασικές έννοιες και τεχνολογίες που χρησιμοποιούνται στη διπλωματική εργασία. Αρχικά, θα αναφερθεί εκτενώς η Δικτύωση Καθοριζόμενη από Λογισμικό, όπως και τα πλεονεκτήματα και οι προκλήσεις αυτής. Στην συνέχεια, θα γίνει αναφορά σε ένα δεύτερο κομμάτι το οποίο συνηθίζεται να χρησιμοποιείται σε συνδυασμό με το SDN. Αυτό ονομάζεται Εικονικοποίηση Λειτουργιών Δικτύου και θα αναλυθούν τα πλεονεκτήματα και οι προκλήσεις της αντίστοιχα. Έπειτα, θα ασχοληθούμε με την διασφάλιση της Ποιότητας Υπηρεσιών. Σύμφωνα με το Internet Engineering Task Force (IETF) υπάρχουν αρκετές αρχιτεκτονικές τεχνικές διασφάλισης QoS. Οι δύο πιο διαδεδομένες, οι οποίες θα αναλυθούν παρακάτω, είναι αυτές των ενσωματωμένων υπηρεσιών (IntServ) και των διαφοροποιημένων υπηρεσιών (DiffServ) [5]. Συνεπώς, οι βασικοί άξονες στους οποίους θα κινηθούν οι υποενότητες θα είναι η περιγραφή του SDN, του NFV και του QoS. Τέλος, στην τελευταία υποενότητα, παρουσιάζεται και αναλύεται η βασική ιδέα της διπλωματικής εργασίας.

2.1 Αρχιτεκτονική SDN

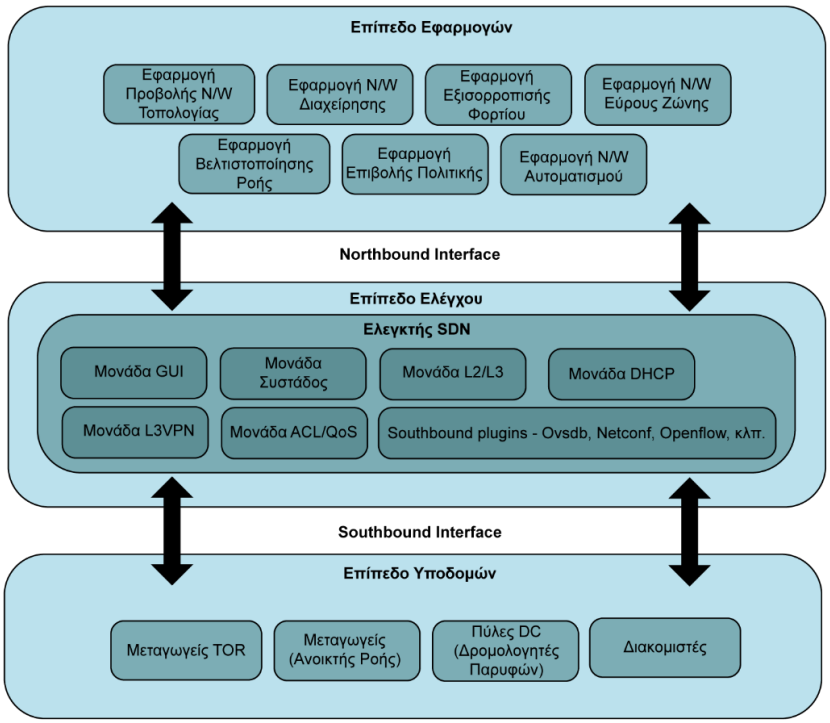
Υπάρχουν πολλοί ορισμοί σχετικά με την Δικτύωση Καθοριζόμενη από Λογισμικό [6] [7]. Μια προσέγγιση ορισμού των Δικτύων Καθορισμένα από Λογισμικό, σύμφωνα με το Open Networking Foundation (ONF), είναι «μια αρχιτεκτονική δικτύου, με σκοπό τον διαχωρισμό του επιπέδου ελέγχου από το επίπεδο δεδομένων». Σε αυτή την αρχιτεκτονική, οι ελεγκτές SDN του δικτύου μπορούν να διαχειρίζονται το δίκτυο ευκολότερα, έχοντας την πλήρη εικόνα του, το οποίο είναι αποδεσμευμένο από τις εφαρμογές λογισμικού και τις υπηρεσίες δικτύου. Για να επιτευχθεί αυτό, το SDN συνδυάζεται με το NFV, η οποία διαχωρίζει τις λειτουργίες δικτύου από το υλικό με τη μορφή εικονικοποιημένων λειτουργιών δικτύου (Virtual Network Function – VNF). Οι ελεγκτές SDN αναλαμβάνουν το ρόλο των ελεγκτών στο δίκτυο, έχοντας την δυνατότητα διαμόρφωσης και ελέγχου της κίνησης δεδομένων. Η ικανότητα της ανακατεύθυνσης και της αυτοματοποίησης της κυκλοφορίας των δεδομένων διευκολύνει την εφαρμογή της ποιότητας των υπηρεσιών (QoS) για εκπομπές μέσω Πρωτοκόλλου Διαδικτύου (Internet Protocol -IP).

Όπως παρουσιάζεται στην Εικόνα 1, η αρχιτεκτονική ενός δικτύου SDN απαρτίζεται από 3 επίπεδα και συγκεκριμένα το επίπεδο εφαρμογών, το επίπεδο ελέγχου και το επίπεδο υποδομών.

1. Το Επίπεδο Εφαρμογών (Application Plane – Application Layer) αποτελείται από εφαρμογές του δικτύου όπως είναι το τείχος προστασίας και η εξισορρόπηση φορτίου (Load Balancing), με την χρήση των οποίων αξιοποιούνται οι διαθέσιμες πληροφορίες και τα στοιχεία του [8]. Με αυτόν τον

τρόπο, επιτυγχάνεται ο έλεγχος, η αυτοματοποίηση και η αντιμετώπιση προβλημάτων του δικτύου. Η αμφίδρομη επικοινωνία με το υπόλοιπο δίκτυο επιτυγχάνεται μέσω του Northbound Interface με το επίπεδο ελέγχου.

2. Στο Επίπεδο Ελέγχου (Control Plane) περιλαμβάνονται οι ελεγκτές SDN, όπως ο ONOS, ο OpenDayLight, ο Ryu, ο Nox και ο Pox [8]-[10]. Το συγκεκριμένα επίπεδο, είναι υπεύθυνο για την δρομολόγηση της κίνησης εντός του δικτύου, λαμβάνοντας εντολές από το επίπεδο εφαρμογών και μεταφέροντας αυτές στις συσκευές του δικτύου. Ο ρόλος του ελεγκτή SDN είναι διπλός: να εξάγει την οποιαδήποτε χρήσιμη πληροφορία από τις φυσικές δικτυακές συσκευές και στην συνέχεια να παρακολουθεί, να διατηρεί και να επιστρέφει την ακριβή κατάσταση του δικτύου στο επίπεδο εφαρμογών μέσω του Northbound Interface [11]. Το Northbound Interface είναι μια διεπαφή αμφίδρομης επικοινωνίας μεταξύ του επιπέδου εφαρμογών και του επιπέδου ελέγχου. Το Southbound Interface είναι μια αντίστοιχη διεπαφή η οποία ενώνει το επίπεδο ελέγχου με το επίπεδο υποδομών.
3. Το επίπεδο υποδομών (Data Plane – Infrastructure Layer) περιλαμβάνει τον φυσικό εξοπλισμό του δικτύου, ο οποίος αποτελείται από συσκευές μεταγωγέων, υπεύθυνες για την διεκπεραίωση της δρομολόγησης και μετάδοσης των δεδομένων σύμφωνα με τις εντολές που λαμβάνει από τους ελεγκτές SDN. Η επικοινωνία με τον ελεγκτή SDN γίνεται μέσω του Southbound Interface, χρησιμοποιώντας πρωτόκολλα ελέγχου και επικοινωνίας όπως OpenFlow [2].

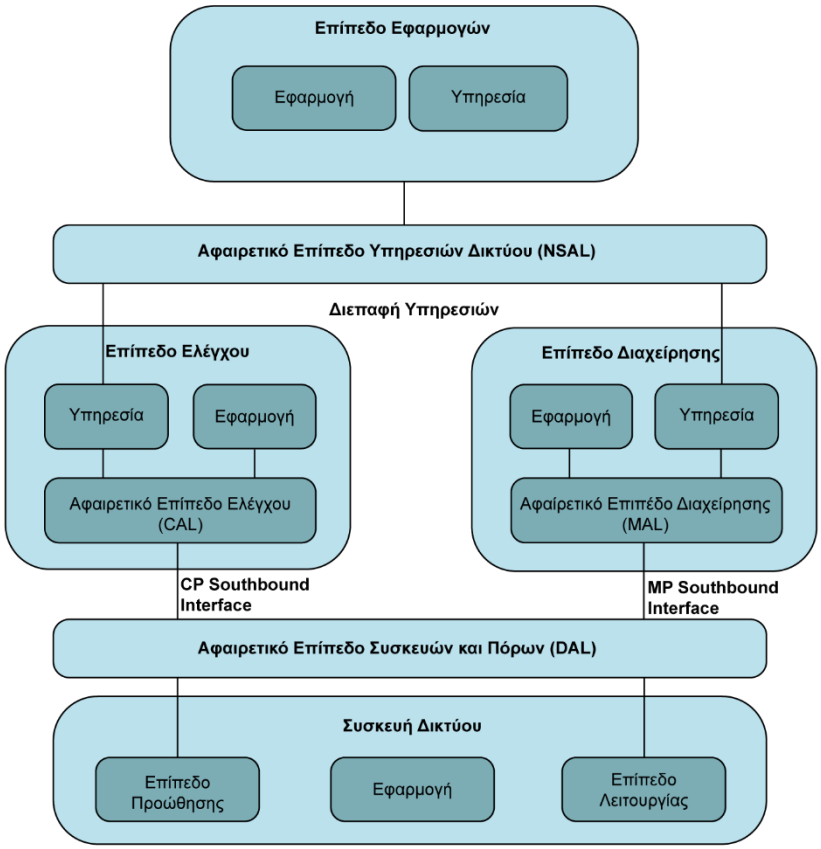


Εικόνα 1. Αρχιτεκτονική SDN τριών επιπέδων

Από τα παραπάνω, μπορεί να διαπιστωθεί ότι η αρχιτεκτονική του SDN είναι [2]:

- Ευέλικτη, καθώς με την χρήση πρωτοκόλλων όπως του Openflow, οι ελεγκτές μπορούν να μεταβάλουν της ροή της κίνησης ανάλογα με τις ανάγκες που προκύπτουν.
- Κεντρικά διαχειρίσιμη εφόσον οι ελεγκτές SDN έχουν πλήρη εικόνα και συνεχόμενο έλεγχο του δικτύου.
- Μεγαλύτερη ασφάλεια δικτύου καθώς όλοι οι κανόνες και οι πολιτικές του δικτύου επιβάλλονται από ένα κεντρικό σημείο ελέγχου με διαρκή διανομή ανανεωμένων πληροφοριών ασφάλειας σε ολόκληρο το δίκτυο.
- Ανεξάρτητη από τους προμηθευτές των συσκευών λόγω των ανοικτών πρωτοκόλλων και προτύπων που υποστηρίζονται όπως το Openflow.

Αξίζει να σημειωθεί ότι υπήρξε προσπάθεια εκσυγχρονισμού της αρχιτεκτονικής του δικτύου SDN σύμφωνα με το RFC7426 [12], το οποίο όμως δεν επιφέρει μεγάλες αλλαγές πέραν από την κατανομή του δικτύου σε 5 επίπεδα. Αναφορικά, τα επίπεδα αυτά είναι: α) Επίπεδο Προώθησης (Forwarding Plane), β) Επίπεδο Λειτουργίας (Operational Plane), γ) Επίπεδο Ελέγχου (Control Plane), δ) Επίπεδο Διαχείρισης (Management Plane) και ε) Επίπεδο Εφαρμογών (Application Plane).
Η Εικόνα 2 παρουσιάζει την αρχιτεκτονική RFC 7426.



Εικόνα 2. Αρχιτεκτονική SDN / RFC 7426 [13]

Το SDN βασίζεται τόσο στο λογισμικό όσο και σε μερικά απαραίτητα στοιχεία του. Συγκεκριμένα, τα στοιχεία αυτά είναι οι Εφαρμογές SDN (SDN Applications), ο Ελεγκτής SDN, οι Διεπαφές (Interfaces) και οι Μεταγωγείς (switches). Αναλυτικότερα:

Οι Εφαρμογές SDN είναι λογισμικά τα οποία βρίσκονται σε μόνιμη επαφή, μέσω της Northbound Interface, με τον ελεγκτή και έχουν σαν στόχο να επικοινωνούν με το δίκτυο για την διατήρηση της επιθυμητής κατάστασης του. Επιπλέον, μπορούν να διαθέτουν μια αφηρημένη εικόνα του δικτύου βασισμένη στις πληροφορίες τις οποίες λαμβάνουν δυναμικά από τον ελεγκτή και τις χρησιμοποιούν για την λήψη κατάλληλων αποφάσεων.

Ο Ελεγκτής SDN είναι ένα κεντρικοποιημένο σύστημα λογισμικού υπεύθυνο, αρχικά, για την διαχείριση και την μετάδοση της πληροφορίας σχετικά με την κατάσταση του δικτύου στις εφαρμογές και στη συνέχεια για την παροχή απαιτούμενων στατιστικών, όπου αυτά χρειάζονται. Συχνά, χρησιμοποιείται ένα RESTful Application Programming Interface (API) για την παροχή των υπηρεσιών του ελεγκτή SDN στις εφαρμογές. Ένας ελεγκτής SDN αποτελείται απαραιτήτως από μια λογική μονάδα ελέγχου, έναν Northbound Interface Agent και έναν Control-Data Plane Interface Agent (CDPI). Πιο συγκεκριμένα, η λογική μονάδα ελέγχου εκτελεί την επεξεργασία των δεδομένων που συλλέγονται από τα άλλα δύο επίπεδα, ο Northbound Interface Agent εξασφαλίζει την επικοινωνία μεταξύ του επιπέδου ελέγχου με το επίπεδο εφαρμογών, ο CDPI εξασφαλίζει την επικοινωνία μεταξύ του επιπέδου ελέγχου με το επίπεδο υποδομών, όπως φαίνεται και στην Εικόνα 1.
Μερικοί από τους πιο διαδεδομένους ελεγκτές SDN είναι ο Ryu, ο Open Network Operating System (ONOS), και ο OpenDaylight Ελεγκτής, οι σημαντικότερες διαφορές των οποίων παρατίθενται στον Πίνακα 1 [14-19].

Πίνακας 1. Συγκριτικός πίνακα διαθέσιμων Ελεγκτών SDN

	Ryu	**ONOS**	**OpenDaylight**	**Floodlight**	**Pox**	**Nox**	**Maestro**	**MUL**
Γλώσσα Προγραμματισμού	Python	Java	Java	Java	Python	C++	Java	C
Επεκτασιμότητα/Δημιουργία Συστάδων	Όχι	Ναι	Ναι	Ναι	Όχι	Ναι	Όχι	Όχι
Γραφικό Περιβάλλον Χρήστη	No	Web-based	Web-based	Web/Java-based	Python + QT4	Python + QT4	No	Web-based
REST API	Ναι	Ναι	Ναι	Ναι	Όχι	Ναι	Ναι	Ναι

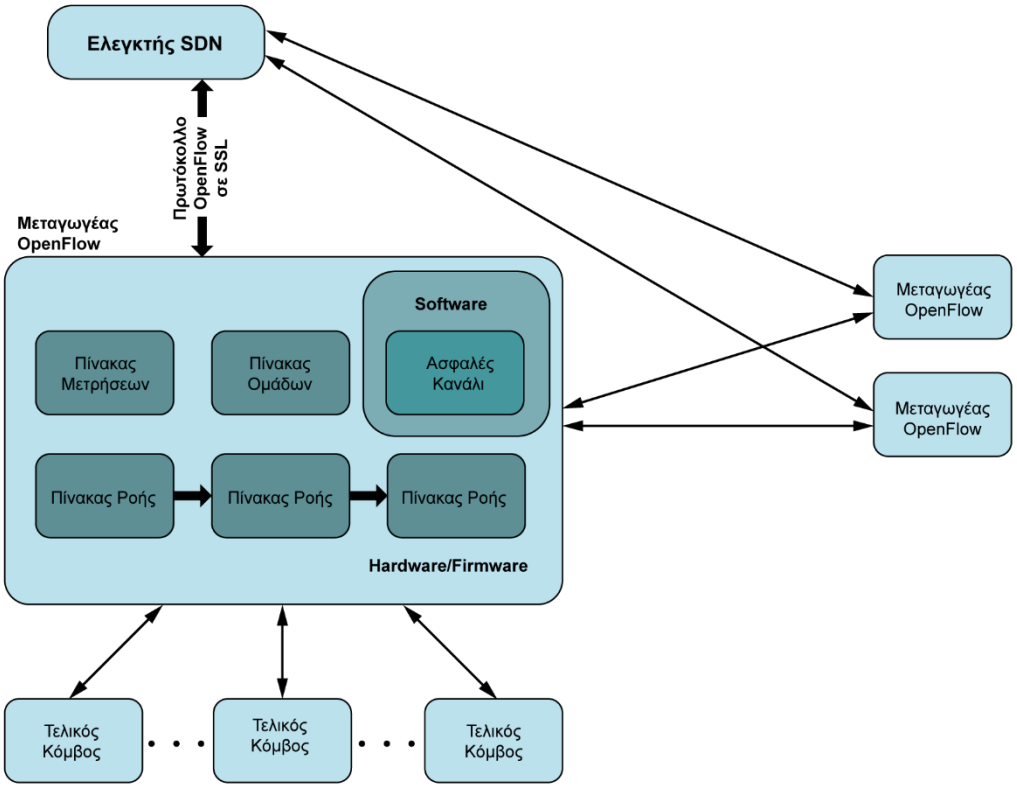
Λογισμικό	Mostly Linux	Mostly Linux but supports Windows and Mac	Mostly Linux but supports Windows and Mac	Mostly Linux but supports Windows and Mac	Mostly Linux but supports Windows and Mac	Mostly Linux	Mostly Linux but supports Windows and Mac	Mostly Linux
Εγχειρίδιο Χρήσης	Αρκετά Καλό	Αρκετά Καλό	Αρκετά Καλό	Ικανοποιητικό	Ελλιπές	Ελλιπές	Ελλιπές	Ικανοποιητικό
Υποστήριξη Multithread	Ναι	Ναι	Ναι	Ναι	Όχι	NOX_MT	Ναι	Ναι
Υποστήριξη Openstack	Ναι	Ναι	Όχι	Όχι	Όχι	Όχι	Όχι	Ναι

Διεπαφή (Interface) αποκαλείται γενικά η οντότητα ή μέρος αυτής που παρεμβάλλεται μεταξύ ενός αντικειμένου και του περιβάλλοντός του και δια μέσου της οποίας πραγματοποιείται κάθε επικοινωνία του αντικειμένου με το περιβάλλον του. Στο SDN, οι συγκεκριμένες διεπαφές είναι το Northbound Interface (NBI) και το Southbound Interface (SBI), τα οποία παρέχουν επικοινωνία μεταξύ του Επιπέδου Εφαρμογών-Επιπέδου Ελέγχου και του Επιπέδου Ελέγχου-Επιπέδου Υποδομών αντίστοιχα. Συγκεκριμένα, το NBI ορίζει την επικοινωνία μεταξύ των εφαρμογών και του ελεγκτή, με στόχο να προβάλλει μια εικόνα του δικτύου και παράλληλα, να γίνονται άμεσα οι κατάλληλες αλλαγές για την διατήρηση του στην ιδανικότερη κατάσταση. Το συγκεκριμένο επίπεδο είναι πλήρως βασισμένο σε λογισμικό ανοιχτού κώδικα με την δυνατότητα χρήσης και προτύπων κλειστού κώδικα.

Στην συνέχεια, το SBI είναι η διεπαφή που φέρνει σε επικοινωνία τον ελεγκτή με τις συσκευές του δικτύου όπως είναι οι μεταγωγείς. Μέσω αυτού, ο ελεγκτής ελέγχει όλες τις λειτουργίες του δικτύου, στέλνοντας τις εντολές για δυναμικές αλλαγές και στην συνέχεια λαμβάνει στατιστικά και δεδομένα που αφορούν την κίνηση των ροών, την τοπολογία του δικτύου και την κάλυψη των αιτημάτων του ελεγκτή. Οι πιο ευρέως διαδεδομένες διεπαφές είναι το Openflow και το Open vSwitch (OVS) [20]. Το OpenFlow είναι πρωτόκολλο επικοινωνίας για τον έλεγχο του ελεγκτή (π.χ. πρόσθεση, αφαίρεση αλλαγή των ροών). Για να επιτευχθεί αυτή η επικοινωνία χρησιμοποιείται ένας OpenFlow ελεγκτής που έχει σχεδιαστεί με στόχο να παρέχει μια σταθερότητα στην διαχείριση της κίνησης κάνοντας τον έλεγχο ανεξάρτητο από τις συσκευές και τον προμηθευτή. Το OVS στην ουσία είναι ένας “εικονικός” ελεγκτής, ο οποίος υποστηρίζει πολλά πρωτόκολλα διαχείρισης ένα εκ των οποίων είναι το OpenFlow, το οποίο και θα χρησιμοποιηθεί στην προτεινόμενη υλοποίηση [20].

Οι μεταγωγείς, όπως γίνεται αντιληπτό από την Εικόνα 3, είναι συσκευές που προωθούν τα πακέτα στο περιβάλλον του SDN. Τα σημαντικότερα μέρη τους είναι οι πύλες, από τις οποίες εισέρχεται και εξέρχεται η κίνηση, οι πίνακες ροών (flow tables),

οι οποίοι αποθηκεύουν τους κανόνες υπεύθυνους για την προώθηση της κίνησης εντός δικτύου και τα group tables τα οποία περιέχουν μια σειρά εντολών που εφαρμόζονται στα κατάλληλα πακέτα.



Εικόνα 3. Τυπικός Μεταγωγέας SDN

Τέλος, στην περίπτωση που εισέρχεται ένα πακέτο για το οποίο δεν υπάρχει ορισμένη ροή, ο μεταγωγέας επικοινωνεί με τον ελεγκτή SDN, μέσω του πρωτοκόλλου Openflow, για να ζητήσει πληροφορίες σχετικές με το πως να χειριστεί το συγκεκριμένο πακέτο (flow missmatch). Στην συνέχεια, κατεβάζει τις κατάλληλες πληροφορίες με τις οποίες θα διαχειρίζεται τα επόμενα πακέτα παρόμοιου τύπου.

2.2 Πλεονεκτήματα SDN

Συνεπώς, μερικά από τα πλεονεκτήματα της χρήσης SDN συγκριτικά με τα συμβατικά δίκτυα είναι [10]:

- Μεγαλύτερη ευελιξία λόγω των ελεγκτών που υπάρχουν εντός δικτύου, οι οποίοι σε συνδυασμό με το διαχωρισμό μεταξύ των επιπέδων προώθησης και ελέγχου, δίνουν την δυνατότητα ελέγχου της κίνησης σε πραγματικό χρόνο για να συμβαδίζει με τις εκάστοτε ανάγκες του δικτύου.

- Παροχή κεντρικής διαχείρισης καθώς ο ελεγκτής SDN έχει καθολική εικόνα του δικτύου, είναι υπεύθυνος για την διαχείριση και την μετάδοση της πληροφορίας και την παροχή απαιτούμενων στατιστικών.

- Χαμηλότερα κόστη προμήθειας, εγκατάστασης και συντήρησης, διότι απαιτούνται λιγότερα υλικά μέρη.

- Η διαχείριση και ο προγραμματισμός του δικτύου είναι στιγμιαία και ακριβής με δυνατότητα βελτιστοποίησης οποιαδήποτε στιγμή αφού το δίκτυο λειτουργεί με την χρήση λογισμικών SDN, τα οποία τροποποιούνται εύκολα χωρίς την ανάγκη αντικατάστασης υλικών μερών.

- Βασίζονται πλήρως σε λογισμικό ανοιχτού κώδικα με μεγάλες κοινότητες χρηστών για την βοήθεια και επίλυση προβλημάτων που τυχόν μπορεί να προκύπτουν.

2.3 Προκλήσεις SDN

Όπως αναφέρθηκε στην προηγούμενη υποενότητα, το SDN προσφέρει πλήθος πλεονεκτημάτων. Παρά το πλήθος των πλεονεκτημάτων, δεν παύουν να υπάρχουν προκλήσεις και πιθανά προβλήματα που πρέπει να αντιμετωπιστούν για την ομαλότερη μετάβαση από τα παραδοσιακά δίκτυα.

1) Από τις σημαντικότερες προκλήσεις του SDN είναι αυτές που αφορούν την την επεκτασιμότητα του δικτύου (scalability) και την συγκέντρωση του ελέγχου του δικτύου στον ελεγκτή SDN. Όσο επεκτείνεται το δίκτυο, οι απαιτήσεις του ελεγκτή SDN σε επεξεργαστική ισχύ, μνήμη και εύρος ζώνης αυξάνονται και επομένως προκύπτει ο κίνδυνος υπερφόρτωσης. Επιπλέον, όταν επεκτείνεται το δίκτυο, εισάγεται επιπλέον καθυστέρηση στην επικοινωνία μεταξύ του ελεγκτή SDN και των μεταγωγών, η οποία οδηγεί σε γενικότερη αύξηση καθυστέρησης αλλά και σε συμφόρηση στο επίπεδο ελέγχου και υποδομών [21].

2) Άλλη μια πρόκληση που μπορεί να προκύψει στα δίκτυα SDN είναι η παροχή αξιοπιστίας (reliability). Για οποιαδήποτε αποτυχία προκύψει στο σύστημα, οι διαχειριστές του δικτύου, όπως και οι χρήστες, θα πρέπει σε πραγματικό χρόνο να ενημερώνονται και η όποια λύση υπάρξει να υλοποιηθεί αυτόματα. Στην ουσία, ως αξιοπιστία θα μπορούσε να χαρακτηριστεί η πιθανότητα που αντιπροσωπεύει το ενδεχόμενο σωστής λειτουργίας του συστήματος, μετά από κάποια βλάβη, σε καθορισμένο περιβάλλον και χρόνο. Γι’ αυτό, η διαχείριση του δικτύου πρέπει να είναι ορθά καθορισμένη, έτσι ώστε να αναγνωρίζεται στον καλύτερο δυνατό χρόνο σε ποιο σημείο του δικτύου βρίσκεται η βλάβη και

να δρομολογείται η κίνηση από εναλλακτικές διαδρομές. Ένα χαρακτηριστικό παράδειγμα για την παραπάνω λειτουργία, είναι ο ελεγκτής ONOS, ο οποίος προσφέρει σταθερά αναφορές σφαλμάτων και ανίχνευση βλαβών σε πραγματικό χρόνο [22].

3) Η επίτευξη υψηλής ασφάλειας είναι άλλη μια πρόκληση για το SDN. Οι νέες διεπαφές μπορούν να επιφέρουν νέα είδη κυβερνοεπιθέσεων που μπορούν να μειώσουν την απόδοση του δικτύου ή να το κάνουν μη λειτουργικό. Παραδείγματα επιθέσεων είναι οι επιθέσεις άρνησης υπηρεσιών (Denial of Service – DoS) και εκμετάλλευσης ευπαθειών στους μηχανισμούς ταυτοποίησης και εξουσιοδότησης μεταξύ των ελεγκτών SDN και μεταγωγέων [5].

2.4 Εικονικοποίηση Λειτουργιών Δικτύου

Η Εικονικοποίηση Λειτουργιών Δικτύου είναι μια μέθοδος μετάβασης των παραδοσιακών λειτουργιών του δικτύου, όπως είναι η δρομολόγηση, το τείχος προστασίας και η εξισορρόπηση του φόρτου, από φυσικές υποδομές που προορίζονταν για αυτές τις λειτουργίες σε εικονικές μηχανές οι οποίες φιλοξενούνται σε ένα κέντρο δεδομένων (datacenter). Συνεπώς, παύει να είναι απαραίτητη η φυσική υποδομή ειδικού σκοπού και όλα βασίζονται στον ψηφιακό κόσμο. Η αρχιτεκτονική στην οποία βασίζεται φαίνεται στην Εικόνα 4. Αυτό ακολουθείται από πλεονεκτήματα και μειονεκτήματα τα οποία θα αναλυθούν.

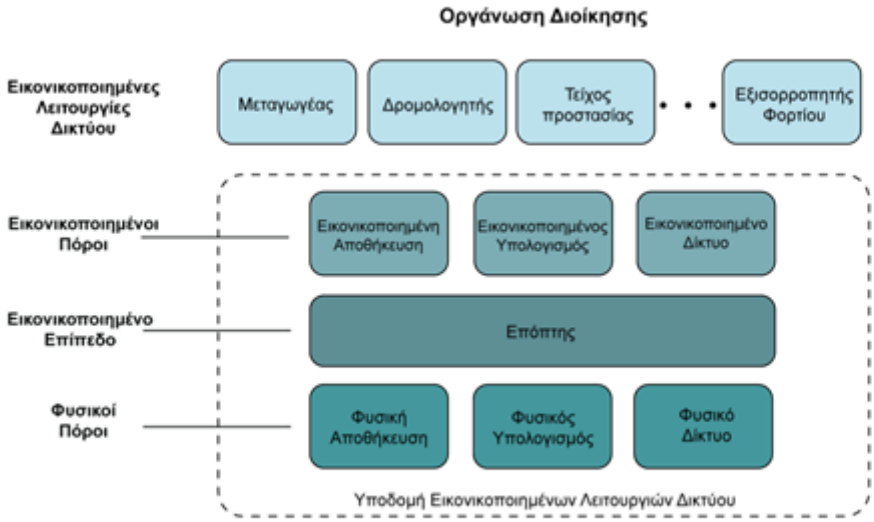
2.5 Πλεονεκτήματα NFV

Μερικά από τα πλεονεκτήματα που προσφέρει η συγκεκριμένη τεχνολογία είναι η υψηλή επεκτασιμότητα και κλιμάκωση του δικτύου χωρίς την ανάγκη αγοράς νέου εξοπλισμού, είτε κάθε φορά που το δίκτυο φτάνει τις οριακές συνθήκες των δυνατοτήτων του, είτε στην περίπτωση όπου οι πάροχοι επιθυμούν να παρέχουν νέες υπηρεσίες και εφαρμογές, κατόπιν αιτήματος [23]. Το Ευρωπαϊκό Ινστιτούτο Προτύπων Τηλεπικοινωνιών (European Telecommunications Standards Institute – ETSI), συνέβαλε στον καθορισμό της αρχιτεκτονικής NFV, με τον ορισμό μερικών προτύπων για την επίτευξη σταθερότητας και διαλειτουργικότητας [24].

Πιο συγκεκριμένα, τα μέρη της NFV αρχιτεκτονικής είναι:

- VNFs, τα οποία αποτελούνται από εφαρμογές λογισμικού για κοινή χρήση δεδομένων, υπηρεσιών και την διαμόρφωση του πρωτοκόλλου διαδικτύου (IP protocol).

- Υποδομές Εικονικοποίησης Λειτουργιών Δικτύου (Network Functions Virtualization Infrastructure – NFVI) οι οποίες περιλαμβάνουν τα απαιτούμενα υλικοτεχνικά στοιχεία όπως μονάδες αποθήκευσης, επεξεργασίας και δικτύωσης.
- Διαχείριση, αυτοματοποίηση και οργάνωση δικτύου (management, automation and network orchestration), τα οποία υποστηρίζουν την παροχή και τον έλεγχο νέων ή υπαρχόντων NFV εγκαταστάσεων.



Εικόνα 4. Αρχιτεκτονική NFV [25]

2.6 Προκλήσεις NFV

Αν και η εικονικοποίηση λειτουργιών δικτύου παρέχει πολλαπλά οφέλη, αντιμετωπίζει επίσης και ορισμένες προκλήσεις οι οποίες θα μπορούσαν να υποβαθμίσουν την απόδοση του δικτύου. Στη συνέχεια, παρουσιάζονται ορισμένες προκλήσεις καθώς και πιθανοί τρόποι αντιμετώπισης τους [23]:

- Καθώς οι περισσότερες λειτουργίες μεταφέρονται σε εικονικοποιημένο περιβάλλον, αυξάνεται ο κίνδυνος κυβερνοεπιθέσεων, οι οποίες μπορούν να οδηγήσουν σε μη εξουσιοδοτημένη πρόσβαση ή υποκλοπή δεδομένων. Για υψηλότερη ασφάλεια, μπορεί να γίνει απομόνωση της εικονικής μνήμης με δυνατότητα πρόσβασης μόνο μετά από έλεγχο ταυτότητας. Επίσης, τα δεδομένα που αποθηκεύονται θα πρέπει να κρυπτογραφούνται με τέτοιο τρόπο που θα έχουν πρόσβαση μόνο τα VNF.

- Οι λειτουργίες εικονικού δικτύου εξαρτώνται άμεσα με τα χαρακτηριστικά των διακομιστών όπως η αρχιτεκτονική που χρησιμοποιούν, η επεξεργαστική ισχύς, η μνήμη cache και το εύρος ζώνης. Συνεπώς, για την παροχή συγκρίσιμων ή καλύτερων επιδόσεων από τον συμβατικό εξοπλισμό απαιτείται η λήψη σωστών αποφάσεων, όπως η χρήση ανεξαρτήτων δομών μνήμης για την αποφυγή συμφόρησης, η χρήση πολλαπλών κατανεμημένων τμημάτων για την εκτέλεση σε πολλαπλούς πυρήνες ή διαφορετικές συσκευές.
- Τα VNFs θα πρέπει να είναι ανεξάρτητα από οποιοδήποτε συγκεκριμένο υλικό ή λογισμικό για να υπάρχει η δυνατότητα να αναπαραχθούν σε διαφορετικά εικονικά μηχανήματα και να εκμεταλλεύονται τα πλεονεκτήματα της εικονικοποίησης. Η υλοποίηση των VNFs, θα πρέπει να βασίζεται σε εικονικό διαχειριστή πόρων πολλαπλών πλατφορμών (Cross Platform Virtual Resource Manager) για την διασφάλιση της φορητότητας.
- Τέλος, η διαχείριση όπως και η αναβάθμιση από παλαιού τύπου συστήματα σε νέα θα πρέπει να γίνεται εύκολα, χωρίς τον φόβο απώλειας δεδομένων. Για την εξασφάλιση αυτού του ζητουμένου, κρίνεται απαραίτητη η χρήση καλά ορισμένων προτύπων συσκευών NFV που θα έχουν όσο το δυνατόν ελάχιστες επιπτώσεις στο υπάρχον δίκτυο.

2.7 Ενσωματωμένες Υπηρεσίες

Στην συγκεκριμένη ενότητα θα αναφερθούμε στις Ενσωματωμένες Υπηρεσίες (Integrated Services – IntServ). Ως ποιότητα εξυπηρέτησης (QoS) μπορεί να ορισθεί η χρήση μηχανισμών ή τεχνολογιών που λειτουργούν εντός του δικτύου για τον έλεγχο προτεραιότητας της κυκλοφορίας και τη διασφάλιση της απόδοσης των κρίσιμων εφαρμογών. Μέσω του QoS, επιτρέπεται στο σύστημα η παροχή διαφορετικών προτεραιοτήτων σε διαφορετικές εφαρμογές, χρήστες ή ροές δεδομένων δίνοντας προτεραιότητα σε αυτές που απαιτούν υψηλή απόδοση. Για να επιτευχθεί αυτό, είναι απαραίτητο να γίνει ιεράρχηση της κίνησης προσδιορίζοντας τον τύπο των πακέτων και χωρίζοντας τη σε εικονικές ροές σύμφωνα με την προτεραιότητα τους. Ως αποτέλεσμα, δεσμεύεται ένα ικανοποιητικό εύρος ζώνης για τις κρίσιμες εφαρμογές ή συγκεκριμένους χρήστες με σκοπό τη μείωση της δικτυακής συμφόρηση. Η επιλογή της ποιότητας υπηρεσιών διαφέρει σύμφωνα με το χαρακτηριστικό που θεωρεί ότι έχει την κυριότερη σημασία σε ένα πακέτο αλλά και το κριτήριο σχετικά με την επιλογή της βέλτιστης διαδρομής [26].
Η βέλτιστη διαδρομή μπορεί να υπολογιστεί με βάση τα παρακάτω κριτήρια:

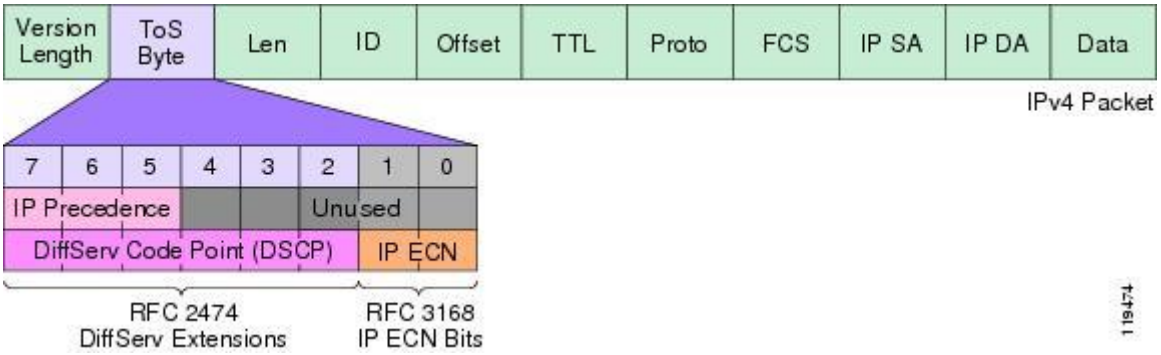
- Εύρος ζώνης (Bandwidth) : Το QoS ορίζει στον δρομολογητή ως βασικό κριτήριο επιλογής των βέλτιστων διαδρομών, την υψηλότερη ταχύτητα μεταξύ των συνδέσμων. Παραδείγματος χάρη, μπορεί να οριστεί συγκεκριμένο εύρος ζώνης σε συγκεκριμένες ροές για να ακολουθούν μια μοναδική διαδρομή μέσα στο δίκτυο, η οποία έχει την μεγαλύτερη ταχύτητα.
- Καθυστέρηση (Delay) : Το QoS θα ορίσει τη μεταφορά της κίνησης μέσα από διαδρομές ώστε να ελαχιστοποιείται ο χρόνος μετάβασης ενός πακέτου από την πηγή στον τελικό προορισμό. Σε πολλές περιπτώσεις, ενώ χρησιμοποιούνται οι διαδρομές με το μέγιστο εύρος ζώνης, ο χρόνος για να μεταδοθεί ένα πακέτο μπορεί να είναι πολύ μεγάλος λόγω δικτυακής συμφόρησης. Για την αποφυγή αυτού του φαινομένου, δημιουργούνται ουρές προτεραιότητας για ορισμένους τύπους κίνησης.
- Απώλεια (Loss) : Αρκετές φορές, όταν το δίκτυο υπερφορτώνεται, αναγκάζεται να απορρίψει (drop) πακέτα, γεγονός το οποίο μπορεί να οδηγήσει σε απώλεια σημαντικών πληροφοριών. Για την αποφυγή αυτής της κατάστασης, το QoS μπορεί να ορίσει ποια πακέτα είναι υψίστης σημασίας και να απαγορεύεται η απώλεια αυτών.
- Jitter : Τέλος, σε μερικές περιπτώσεις, κάποια πακέτα μπορεί να φτάσουν με καθυστέρηση και εκτός της ορθής ακολουθίας λόγω των διάφορων διαδρομών που ακολουθούν. Στην συγκεκριμένη περίπτωση, μεταφέρεται η πληροφορία μερικώς κατεστραμμένη (παραμόρφωση ήχου και εικόνας). Για την μείωση αυτού του φαινομένου, μπορούν να χρησιμοποιηθούν ουρές, με σκοπό να δοθεί προτεραιότητα στα ευαίσθητα σε καθυστέρηση πακέτα και όχι σε άλλου είδους κίνηση αν το πρόβλημα δημιουργείται από την συμφόρηση του δικτύου.

Η ενσωμάτωση υπηρεσιών επιτυγχάνει υψηλή ποιότητα υπηρεσίας μέσω δέσμευσης πόρων, ορίζοντας κοινή πολιτική για κάθε δρομολογητή. Πιο συγκεκριμένα, όταν μια εφαρμογή χρειάζεται υψηλό QoS, κάνει δέσμευση σε κάθε δρομολογητή μεταξύ της εφαρμογής και του προορισμού για να δεσμεύσει τους απαραίτητους πόρους. Στην συνέχεια, ο κάθε δρομολογητής λαμβάνει το αίτημα και απαντάει αν βρίσκεται σε θέση να διαθέσει τους πόρους. Η ενσωμάτωση υπηρεσιών πραγματοποιείται σε δύο στάδια. Το πρώτο στάδιο ονομάζεται Flow Spec και χρησιμοποιείται από την εκάστοτε εφαρμογή για την περιγραφή των απαραίτητων κρατήσεων από τον δρομολογητή. Χωρίζεται στον Traffic Specification (TSPEC) και το Ruby Specification (RSPEC) τα οποία αφορούν τα χαρακτηριστικά της κίνησης και των αιτημάτων αντίστοιχα. Το δεύτερο στάδιο είναι το πρωτόκολλο κράτησης πόρων (Resource Reservation Protocol – RSVP) το οποίο σηματοδοτεί την δέσμευση του εύρους ζώνης και της καθυστέρησης που απαιτείται για την κάθε περίοδο λειτουργίας. Το κύριο πρόβλημα του IntServ είναι ότι απαιτεί μεγάλο αριθμό ροών κάτι το οποίο κάνει περίπλοκο και δαπανηρό το δίκτυο. Για αυτόν τον λόγο δημιουργήθηκε μια νεότερη τεχνολογία, διαφοροποιημένων υπηρεσιών (DSCP) [27].

2.8 Διαφοροποίηση Υπηρεσιών

Η διαφοροποίηση υπηρεσιών (Differentiated Services - DiffServ) χρησιμοποιεί ένα μοντέλο πολλαπλών υπηρεσιών, με κοινή πολιτική για όλους τους δρομολογητές που έχει την δυνατότητα να ικανοποιήσει μια ποικιλία QoS. Σε αντίθεση με την ενσωμάτωση υπηρεσιών, δεν απαιτείται δέσμευση πόρων ούτε γνώση του εύρους ζώνης και της καθυστέρησης για την λήψη κάποιας απόφασης, πριν την αποστολή του πακέτου. Η αρχιτεκτονική καθορίζει έναν απλό και επεκτάσιμο μηχανισμό για την ταξινόμηση και τη διαχείριση της κυκλοφορίας του δικτύου για την παροχή ποιότητας υπηρεσιών. Το σκεπτικό είναι ότι στην αρχή της κεφαλίδας IP υπάρχει συγκεκριμένος αριθμός που τοποθετεί το πακέτο σε διαφορετική κλάση για την κατηγοριοποίηση και περαιτέρω την ιεράρχηση. Στην συνέχεια, ο δρομολογητής, διαβάζει την τιμή του DSCP και διαφοροποιεί αναλόγως την κίνηση [27].

Το πεδίο του IPv4 Packet το οποίο μας ενδιαφέρει είναι το Type Of Service (TOS), το οποίο αποτελείται από το DiffServ Code Point (DSCP) (6 bits) και το ECN (2 bits) όπως φαίνεται στην Εικόνα 5. Για να επιτευχθούν διαφορετικά επίπεδα προτεραιότητας χρησιμοποιούνται οι τιμές του DSCP. Χρησιμοποιούνται για τον συμβολισμό τους 6 bits. Τα υπολειπόμενα 2 bits τα οποία ονομάζονται Explicit Congestion Notification (ECN) είναι υπεύθυνα για τις αλλαγές στον τρόπο συμφόρησης εντός του δικτύου.



Εικόνα 5. Παρουσίαση DSCP-ECN [28]

Τα περισσότερα δίκτυα απορρίπτουν τα πακέτα όταν βρίσκονται σε συμφόρηση γεγονός το οποίο αποτελεί πρόβλημα. Με την χρήση των ECN bits, δεδομένου ότι ο δέκτης και ο πομπός έχουν ενεργοποιημένη την λήψη και επεξεργασία τους, ο ελεγκτής SDN μπορεί να αντιληφθεί πότε βρίσκεται σε συμφόρηση και να λάβει τις απαραίτητες ενέργειες.

Σύμφωνα με παλαιότερα Request for Comments (RFC), τα πρώτα 3 bits αφορούν τις τιμές προτεραιότητας της IP (IP Precedence Values). Τα συγκεκριμένα bits ορίζουν την προτεραιότητα του πακέτου. Το εύρος τιμών είναι από 0 έως 7 με την ελάχιστη προτεραιότητα να είναι στην τιμή μηδέν και την μέγιστη στην τιμή εφτά. Ο Πίνακας 2 συνοψίζει τις βαθμίδες προτεραιότητας.

Πίνακας 2. Τιμές Προτεραιότητας IP [29]

Περιγραφή	**Τιμές**
Routine	000 (0)
Priority	001 (1)
Immediate	010 (2)
Flash	011 (3)
Flash Override	100 (4)
Critic/Critical	101 (5)
Internetwork Control	110 (6)
Network Control	111 (7)

Σύμφωνα με νεότερα RFC, όπως το 2475, το πεδίο του TOS πλέον αποτελείται από τις τιμές του DSCP και το IP ECN ή CU, όπως φαίνονται στην Εικόνα 6.

DS FIELD							
0	1	2	3	4	5	6	7
DSCP						CU	

Εικόνα 6. Πεδίο DS

Οι τιμές DSCP καταλαμβάνουν τα πρώτα 6 bits και είναι υπεύθυνα για την προτεραιότητα του πακέτου και τον καθορισμό του Per Hop Behavior (PHB) για τον κάθε κόμβο. Θεωρητικά, ένα δίκτυο θα μπορούσε να έχει έως 64 διαφορετικές κλάσεις χρησιμοποιώντας τις διαθέσιμες τιμές του DSCP, οι οποίες παρουσιάζονται στον Πίνακα 3 [30].

Πίνακας 3. Τιμές DSCP [30]

Δυαδική Τιμή	**Δεκαδική Τιμή**	**Πιθανότητα Απωλειών**	**Περιγραφή**	**Τιμές Προτεραιότητας IP**
101 110	46	Μη διαθέσιμη	High Priority Expedited Forwarding (EF)	101 - Critical
000 000	0	Μη διαθέσιμη	Best Effort	000 - Routine
001 010	10	Χαμηλή	AF11	001 - Priority
001 100	12	Μεσαία	AF12	001 - Priority
001 110	14	Υψηλή	AF13	001 - Priority
010 010	18	Χαμηλή	AF21	010 – Immediate
010 100	20	Μεσαία	AF22	010 – Immediate

010 110	22	Υψηλή	AF23	010 – Immediate
011 010	26	Χαμηλή	AF31	011 – Flash
011 100	28	Μεσαία	AF32	011 – Flash
011 110	30	Υψηλή	AF33	011 – Flash
100 010	34	Χαμηλή	AF41	100 - Flash Override
100 100	36	Μεσαία	AF42	100 - Flash Override
100 110	38	Υψηλή	AF43	100 - Flash Override
001 000	8		CS1	1
010 000	16		CS2	2
011 000	24		CS3	3
100 000	32		CS4	4
101 000	40		CS5	5

110 000	48		CS6	6
111 000	56		CS7	7
000 000	0		Default	
101 110	46		EF	

Τα RFC των DiffServ συνιστούν, αλλά δεν απαιτούν, κάποια συγκεκριμένη κωδικοποίηση. Αυτό παρέχει ευελιξία στον καθορισμό κατηγοριών κίνησης. Στην πράξη, ωστόσο, τα περισσότερα δίκτυα χρησιμοποιούν τις ακόλουθες κοινώς καθορισμένες συμπεριφορές per-hop:

a) PHB Default Forwarding (DF), η οποία προωθεί την κίνηση με τον βέλτιστο δυνατό τρόπο δεδομένης μόνο της κατάστασης του δικτύου.
b) PHB Expedited Forwarding (EF), η οποία δίνει προτεραιότητα στην κίνηση με χαμηλές απώλειες και χαμηλή καθυστέρηση.
c) PHB Assured Forwarding (AF), η οποία παρέχει εγγύηση παράδοσης υπό καθορισμένες συνθήκες.
d) PHB Selector Class, η οποία διατηρεί συμβατότητα με το προϋπάρχον πεδίο προτεραιότητας IP [31].

Για την αποφυγή συμφόρησης απαιτούνται και μηχανισμοί για την διαχείριση των ουρών. Η πιο συνηθισμένη μέθοδος για την αποφυγή συμφόρησης είναι η Class-Based Weighted Fair Queueing (CBWFQ) [32], η οποία είναι ένας μηχανισμός προγραμματισμού που χρησιμοποιείται για την παροχή ελάχιστης εγγύησης εύρους ζώνης στις κλάσεις κίνησης σε περιόδους συμφόρησης δικτύου σε μια διεπαφή. Για καλύτερη αποτελεσματικότητα, συνδυάζεται με το Low Latency Queueing (LLQ) [33] ο οποίος είναι μηχανισμός προγραμματισμού που φέρνει αυστηρή σειρά προτεραιότητας στο CBWFQ. Η σειρά προτεραιότητας, στη συνέχεια, επιτρέπει στα δεδομένα που είναι ευαίσθητα στην καθυστέρηση, όπως η φωνή, να αφαιρεθούν και να σταλούν πριν αφαιρεθούν πακέτα από άλλες ουρές. Επίσης, μία μέθοδος αποφυγής της συμφόρησης είναι η Weighted Random Early Detection (WRED) [34] που χρησιμοποιείται συνδυαστικά με την CBWFG και παρέχει την δυνατότητα να χρησιμοποιούνται οι DSCP τιμές ή οι IP Precedence για την απόφαση του δικτύου σχετικά με το αν πρέπει να απορρίψει κάποιο πακέτο ή όχι.

2.9 Ιστορική Αναδρομή & Ανάλυση της κεντρικής ιδέας

Στην παρούσα ενότητα, παρουσιάζεται η ανάλυση της ιδέας της προτεινόμενης μεθόδου. Αρχικά, παρουσιάζεται η υπάρχουσα κατάσταση, στη συνέχεια τα προβλήματα της, καθώς και πώς αυτά μπορούν να επιλυθούν μέσω της προτεινόμενης μεθόδου.

Το πρόβλημα είναι πολυεπίπεδο και συγκεκριμένα χωρίζεται σε τρεις άξονες, οι οποίοι είναι η διαχείριση των δικτύων, η ανανέωση της τοπολογίας του δικτύου και η διαχείριση της κίνησης του δικτύου. Αρχικά, στο καθαρό κομμάτι της διαχείρισης των δικτύων, υπήρχαν διάφορες γενιές. Η πρώτη γενιά δικτύων ήταν για απλή μετάδοση πληροφορίας, χωρίς ιδιαίτερες απαιτήσεις, το οποίο όμως οδήγησε σε προβλήματα μοναδικότητας και αναγνώρισης των συσκευών μέσα στο δίκτυο. Έτσι, με το IEEE Std 802-2001 [35] ορίστηκε για πρώτη φορά το πρωτόκολλο Medium Access Control (MAC). Στη δεύτερη φάση, τα δίκτυα μεγάλωσαν σε κλίμακα αλλά με γραμμικό τρόπο, μέσω της χρήσης συσκευών ενίσχυσης και επανεκπομπής της πληροφορίας (repeaters) όπως και συσκευές για σύνδεση παραπάνω συσκευών στο ίδιο δίκτυο παράλληλα μέσω μεταγωγών, όπως και σύνδεση παραπάνω δικτύων μεταξύ τους μέσω δικτυακών γεφυρών (network bridge). Προβλήματα που προέκυπταν εξαιτίας αυτής της αλλαγής, στο κομμάτι των μεταγωγέων είναι η διαχείριση συσκευών που παρόλο που είχαν διαφορετικές MAC addresses, όλες κατέληγαν σε έναν μεταγωγέα που επικοινωνούσε με τον έξω κόσμο, δικτυακά, χωρίς να έχει πληροφορία για το πού θα προωθεί κάθε πακέτο, κάθε συσκευής, το οποίο οδήγησε στο Address Resolution Protocol (ARP) [36], το οποίο λύνει αυτό το πρόβλημα μέσω της δημιουργίας ενός πίνακα αντιστοιχίας πορτών στους μεταγωγείς με την MAC διεύθυνση της συσκευής. Στο κομμάτι των δικτυακών γεφυρών, το πρόβλημα ήταν η «ενοποίηση» που εφάρμοζε αυτός ο μηχανισμός στα δύο υποδίκτυα με αποτέλεσμα να μην υπάρχουν οι διαφορετικές τους ταυτότητες. Έτσι, έρχεται η εποχή των δρομολογητών (routers), κατά την οποία πλέον είμαστε στο τρίτο επίπεδο του μοντέλου OSI. Το μοντέλο OSI είναι μία περιγραφή επιπέδων όπου αναλύεται η δομή επικοινωνίας των τηλεπικοινωνιακών/ υπολογιστικών συστημάτων. Το ενδιαφέρον μας είναι στα επίπεδα 2,3 και 4, όπου αναλύεται η δικτυακή κλιμάκωση που γίνεται προκειμένου να επιτευχθεί η επικοινωνία αυτή [37]. Οι δρομολογητές λοιπόν, αρχίζουν να παρέχουν τη δυνατότητα στο χρήστη δυναμικά να ορίζει κανόνες στο δίκτυο ώστε να αποφεύγονται προβλήματα του παρελθόντος και να δίνεται η δυνατότητα χρήσης των δυνατοτήτων που υπάρχουν με το τρίτο επίπεδο και τη χρήση του IP πρωτοκόλλου και των πληροφοριών που μεταφέρει. Το πιο απλό και καθημερινό παράδειγμα είναι ο δρομολογητής που υπάρχει στα σπίτια μας και μας παρέχει καθημερινά σύνδεση στο διαδίκτυο. Το πρόβλημα που προκύπτει πλέον είναι ότι υπάρχει η ανάγκη διαχείρισης αυτού του πρώτου δείγματος ελεγκτών σε μεγαλύτερη κλίμακα, όπως παραδείγματος χάριν σε εταιρείες, όπου αποτελούνται από πολλά επιμέρους δίκτυα τα οποία το καθένα εξυπηρετεί διαφορετικό σκοπό με διαφορετικό τρόπο. Τα δίκτυα αυτά παρουσιάζουν προβλήματα όπως το γεγονός ότι αποτελούνται από πολλές

υποσυσκευές, οι οποίες χρειάζονται χωριστή διαχείριση και συντήρηση και παρακολούθηση. Για την λύση αυτού του προβλήματος ξεκίνησε η χρήση του SDN. Πιο συγκεκριμένα, το SDN είναι μια προσέγγιση δικτύωσης στην οποία συνεχίζουν να υπάρχουν δρομολογητές και μεταγωγείς, αλλά σε ψηφιακή μορφή, συνδυαστικά με την εισαγωγή ψηφιακών ελεγκτών. Η συγκεκριμένη προσέγγιση έχοντας την παραπάνω ψηφιακή μορφή και με την βοήθεια πρωτοκόλλων υπεύθυνων για τον έλεγχο της συμπεριφοράς των δικτυακών συσκευών, όπως το Openflow, αρχικά επιτυγχάνει την βέλτιστη διαχείριση του δικτύου καθώς μπορεί να λειτουργεί με γρήγορο ρυθμό. Στην συνέχεια, έχοντας ως συνδετικό κρίκο τον ελεγκτή στον οποίο εκτελούνται εφαρμογές υπεύθυνες για την ορθή λειτουργία του δικτύου, εκτελεί δύο βασικές λειτουργίες.
Αφενός, ελέγχει την προώθηση της κίνησης μέσα στο δίκτυο επιλέγοντας δυναμικά μέσα από ένα σύνολο κανόνων, οι οποίοι δίνονται να αλλάξουν ανάλογα με τις ανάγκες που προκύπτουν κατά την χρήση του δικτύου. Αφετέρου, πραγματοποιεί τον έλεγχο της κίνησης για τυχόν προβλήματα που μπορεί να προκύψουν κατά την μεταφορά, όπως κατεστραμμένα αρχεία, μη διαθέσιμοι κόμβοι τα οποία επιλύονται άμεσα με την χρήση εφαρμογών οι οποίες κάνουν απαραίτητους ελέγχους και εφαρμόζουν την βέλτιστη λύση. Πρέπει να σημειωθεί ότι και οι δυο αυτές λειτουργίες γίνονται σε πραγματικό χρόνο. Επιπλέον, ο ελεγκτής του SDN, έχοντας την πλήρη εικόνα του δικτύου, παρέχει κεντρικοποιημένη διαχείριση για την μετάδοση των πληροφοριών και των στατιστικών που απαιτούνται, γεγονός το οποίο συντελεί στην αποδοτικότερη λειτουργία του δικτύου καθώς και στην δημιουργία βάσης δεδομένων για μελλοντικά δίκτυα.
Από αυτήν τη μικρή περιγραφή της λειτουργίας του μπορεί να επιβεβαιωθεί ο λόγος για τον οποίο το SDN είναι η βέλτιστη λύση για την παρούσα διπλωματική εργασία. Το γεγονός ότι το SDN είναι στο μεγαλύτερο μέρος του ψηφιοποιημένο, συντελεί στην καλύτερη απόδοση αυτού με το μικρότερο δυνατό κόστος. Δηλαδή, η αντικατάσταση των φυσικών δρομολογητών με εφαρμογές, των αυστηρά ορισμένων κανόνων σε αυτούς με δυναμικά προγραμματιζόμενους πίνακες και ο κεντρικοποιημένος έλεγχος του δικτύου σε συνδυασμό με τα χαμηλά έξοδα που προκύπτουν από την έλλειψη της πλειοψηφίας των υλικών υποδομών που χρειάζονταν στο παρελθόν για την υποστήριξη αντιστοίχου μεγέθους δικτύων, όπως και το μηδενικό ή ελάχιστο κόστος των εφαρμογών, όπου αυτό είναι εφικτό.
Έτσι, παρατηρήθηκε ότι με το πέρασμα των χρόνων, τα δίκτυα αναπτύσσονταν αλλά ποτέ δεν ήταν αρκετά καθώς οι ανάγκες αυξάνονταν εκθετικά με αυτά. Πλέον, βρισκόμαστε σε μία εποχή που όλα είναι ψηφιοποιημένα, οπότε η χρήση του SDN ως βάση ελέγχου των σύγχρονων δικτύων αποτελεί την βέλτιστη λύση.

Ένα δεύτερο αρκετά σημαντικό πρόβλημα, ως προς τη δομή του δικτύου είναι ο ρυθμός ανανέωσης και επανυπολογισμού των διαδρομών. Στις υπάρχουσες μεθόδους, οι πιο δημοφιλείς αλγόριθμοι για την εύρεση των διαδρομών είναι οι DFS, BFS και Dijisktra. Λαμβάνοντας ως δεδομένες μελέτες που έχουν γίνει για τη σύγκριση αυτών, συμπεραίνεται ότι ο DFS είναι βέλτιστος, σε συνδυασμό με την υπόλοιπη λύση που προσφέρεται στην παρακάτω εργασία, καθώς βελτιώνει τους χρόνους εύρεσης

της τοπολογίας του δικτύου σε σημαντικό βαθμό, όπως θα αναλυθεί και διεξοδικά στο Κεφάλαιο 3.

Ένα τρίτο σημαντικό πρόβλημα είναι η διαχείριση της κίνησης. Τα πρώτα χρόνια η κίνηση δεν είχε διαφοροποιήσεις. Με την έλευση του IP πρωτοκόλλου, δίνεται η δυνατότητα απόκτησης και χρήσης παραπάνω πληροφοριών για τα πακέτα μας εκτός από τον αποστολέα και τον παραλήπτη μόνο, όπως στο Ethernet Frame. Στο IP πλαίσιο, συγκεκριμένα, υπάρχει η πληροφορία που μπορεί να δώσει σημαντικότητα στο πακέτο κάνοντας το να ξεχωρίζει από την υπόλοιπη απλή κίνηση. Σε συνδυασμό με τις λειτουργίες του διαδικτύου, οι οποίες αυξάνονται εκθετικά σε ποσότητα, όπως η προβολή βίντεο υψηλής ευκρίνειας, μετάδοση ζωντανών προβολών, μετάδοση τηλεφωνίας μέσω του διαδικτύου, τηλεδιασκέψεις [31], αλλά και σε ανάγκες ταχύτερης και περισσότερης μετάδοσης δεδομένων στον ίδιο χρόνο και στις ίδιες εγκαταστάσεις, καθίσταται απαραίτητο το σύστημα σήμανσης της σημαντικότητας των πακέτων[38]. Προκειμένου όμως να λυθεί το πρόβλημα, θα πρέπει και να χρησιμοποιηθεί και η αντίστοιχη σήμανση αυτή με κάποιο τρόπο. Για αυτό, χρησιμοποιήθηκε το QoS και συγκεκριμένα μέσω της DSCP σήμανσης, όπως έχουμε αναφέρει στην υποενότητα 2.8.

Στην παραπάνω υποενότητα έγινε ανάλυση και σύγκριση στον τρόπο λειτουργίας των δικτύων μέχρι σήμερα όπως και στην δομή αυτών και την διαχείριση της κίνησης τους. Οι λύσεις που προτάθηκαν υπερέχουν τόσο σε ταχύτητα όσο και σε σχεδιασμό και ποιότητα παροχής υπηρεσιών. Το SDN, ο DFS και το DSCP θα είναι οι βασικοί άξονες αυτής της ιδέας που στοχεύει στην δρομολόγηση κίνησης κρίσιμων υποδομών στις παρυφές του δικτύου. Το αποτέλεσμα αυτής της διπλωματικής εργασίας θα είναι ένα δίκτυο το οποίο θα ανταποκρίνεται στις αυξανόμενες ανάγκες για μεγάλα δίκτυα, τα οποία διατηρούν την ταχύτητα λειτουργίας των μικρών δικτύων, και καταφέρνουν να ικανοποιούν σωστά και γρήγορα τις λειτουργίες του διαδικτύου, οι οποίες ολοένα αυξάνονται σε πολυπλοκότητα και απαιτήσεις.

Στο κεφάλαιο που ακολουθεί, αφού πλέον έχει τεθεί τόσο το γενικό θεωρητικό πλαίσιο, όσο και της συγκεκριμένης ιδέας, θα περιγράφουν τα εργαλεία που χρησιμοποιήθηκαν και πρακτικά πλέον στην υλοποίηση της εργασίας, όπως και η εργασία αυτή καθαυτή.

3. Προτεινόμενη Υλοποίηση

Σε αυτό το κεφάλαιο περιγράφεται το λογισμικό που χρησιμοποιήθηκε, ενώ στη συνέχεια παρουσιάζεται ένας οδηγός που βοηθάει στην περιήγηση στην εργασία και η εκτέλεση του κατάλληλου λογισμικού για την επίτευξη καλύτερης ποιότητας υπηρεσιών και την εξισορρόπηση του δικτυακού φορτίου.

3.1 Ryu Controller

Η αυξανόμενη χρήση του SDN οδηγεί σε αύξηση των δυνατοτήτων των ελεγκτών SDN. Η επιλογή του εκάστοτε ελεγκτή SDN γίνεται βάσει των απαιτήσεων της εφαρμογής και του δικτύου. Συγκρίνοντας τους πιο διαδεδομένους ελεγκτές, σχετικά με την γλώσσα προγραμματισμού που χρησιμοποιούν, τις επιδόσεις τους, την αξιοπιστία τους και άλλα κριτήρια, εξάγεται το συμπέρασμα ότι οι ελεγκτές όπως οι Ryu, ONOS και OpenDaylight, που θεωρούνται από τους πιο διαδεδομένους, καλύπτουν πλήρως από πλευρά δυνατοτήτων τις ανάγκες ενός SDN [39][40]. Στην παρούσα διπλωματική εργασία, επιλέχθηκε ο ελεγκτής Ryu. Ο Ryu είναι ένας ελεγκτής SDN ανοιχτού κώδικα που παρέχει αρκετά εργαλεία μέσω διεπαφών, εύκολα τροποποιήσιμα, με στόχο την δημιουργία νέων εφαρμογών για τον έλεγχο και την διαχείριση ενός δικτύου. Για την επικοινωνία μεταξύ των διεπαφών, υποστηρίζονται τα πρωτόκολλα Openflow και OF-Config τα οποία επικοινωνούν με το επίπεδο προώθησης σχετικά με τον τρόπο διαχείρισης των πακέτων. Οι εφαρμογές που παρέχονται είναι περιορισμένες ωστόσο ο προγραμματιστής έχει την δυνατότητα ανάπτυξης εφαρμογών για την επιθυμητή χρήση το οποίο παρέχει πλήρη ευελιξία για την δημιουργία εξειδικευμένων εφαρμογών. Τέλος, η γλώσσα προγραμματισμού που χρησιμοποιείται στον Ryu είναι η Python.

3.2 Mininet

Το Mininet είναι ένα λογισμικό ανοιχτού κώδικα το οποίο παρέχει δυνατότητες προσομοίωσης ολοκληρωμένων δικτύων [41]. Χρησιμοποιείται ευρέως στα SDN και οι τοπολογίες που δημιουργούνται μπορούν πολύ εύκολα να μεταφερθούν με ελάχιστες αλλαγές σε πραγματικά δίκτυα. Το Mininet θα χρησιμοποιηθεί για την δημιουργία των δικτύων με διάφορες παραμέτρους, όπως πλήθος συσκευών, διαθέσιμο εύρος ζώνης, και την αξιολόγηση του QoS.

3.3 Oracle VM Virtualbox

Το λογισμικό διαχείρισης εικονικών μηχανών που θα χρησιμοποιηθεί είναι η Oracle VM Virtualbox. Η επιλογή του Oracle VM Virtualbox έγινε εξαιτίας της καλής διαλειτουργικότητας (interoperability) μεταξύ των υπολοίπων στοιχείων που

χρησιμοποιούνται για αυτήν την εργασία (Ryu ελεγκτής, Mininet) και προσφέρεται χωρίς επιπλέον κόστος.

3.4 Μέθοδος Υλοποίησης

Ο στόχος της διπλωματικής εργασίας είναι η εύρεση ενός τρόπου κατά τον οποίο η κρίσιμη κίνηση θα μπορούσε να φτάσει όσο το δυνατό γρηγορότερα στον τελικό προορισμό με όσο το δυνατόν μικρότερες απώλειες. Για την επίτευξη του, τα θεωρητικά μοντέλα τα οποία μελετήθηκαν κυρίως έχουν να κάνουν με την δρομολόγηση κίνησης μέσω πολλών διαδρομών (Multipath Routing), την εύρεση των διαθέσιμων μονοπατιών και την επιλογή των κατάλληλων με την χρήση υπολογισμού βαρών.

Αρχικά, σε μια τοπολογία δικτύου, η δρομολόγηση πολλαπλών διαδρομών είναι μια στρατηγική δρομολόγησης που προσδιορίζει πολλές διαδρομές προς έναν στόχο. Η κυκλοφορία του δικτύου μπορεί να κατανέμεται ομοιόμορφα μέσω πολλών διαδρομών στο δίκτυο, μιας μεθόδου γνωστής ως εξισορρόπησης φορτίου (load-balancing), προσφέροντας πολλαπλές διαδρομές προς έναν προορισμό, ενισχύοντας έτσι την αποτελεσματικότητα του δικτύου. Με αυτόν τον τρόπο επιτυγχάνεται, ένα ευρύτερο εύρος ζώνης και μιας μικρότερης καθυστέρησης (latency).

Για την επίτευξη δρομολόγησης πολλαπλών διαδρομών, χρειάζεται να είναι γνωστά όλα τα πιθανά μονοπάτια από και προς μια συγκεκριμένη διαδρομή. Για να επιτευχθεί αυτό, απαιτείται ένας αλγόριθμος εύρεσης μονοπατιών. Παραδείγματα αλγορίθμων είναι ο αλγόριθμος αναζήτησης κατά βάθος (Depth First Search - DFS), ο αλγόριθμος αναζήτησης κατά πλάτος (Breadth-first search - BFS) και ο αλγόριθμος Dijkstra, με την κατάλληλη επεξεργασία.

Συνοπτικά, ο DFS είναι ένας αλγόριθμος αναζήτησης κατά βάθος με σκοπό να επιτύχει αναζήτηση σε ένα δέντρο ή ένα γράφημα. Ξεκινάει από μία ρίζα και εξερευνά όσο το δυνατό περισσότερα μονοπάτια κατά μήκος κάθε κλαδιού του δέντρου μέχρι να φτάσει σε κάποιο αδιέξοδο ή στην δική μας περίπτωση στον τελικό προορισμό. Ο αλγόριθμος DFS πραγματοποιεί αναζήτηση όλων των διαθέσιμων επιλογών χωρίς κανένα κριτήριο (brute-force). Αρχικά, ξεκινάει με τον πρώτο διαθέσιμο κόμβο του δέντρου αναζήτησης που εμφανίζεται και συνεχίζει σε βάθος του δέντρου μέχρι να βρεθεί ο ζητούμενος κόμβος ή μέχρι να φτάσει σε κόμβο που δεν συνεχίζει. Έπειτα, συνέχεια έχει ο τελευταίος ανεξερεύνητος κόμβος. Σε μια μη-αναδρομική υλοποίηση, όλοι οι νέοι ανεξερεύνητοι κόμβοι τοποθετούνται στην κορυφή μιας στοίβας και η αναζήτηση συνεχίζεται με έναν από αυτούς [42][43]. Στην περίπτωση μας, λόγω του ότι πολλές διακλαδώσεις καταλήγουν στους ίδιους κόμβους και θα δημιουργούνταν πρόβλημα ατέρμονου βρόχου, αποτρέπουμε το λογισμικό να κάνει εξερεύνηση των μονοπατιών

που έχει ήδη επισκεφθεί. Η χρονική πολυπλοκότητα που απαιτείται καθορίζεται από το μέγεθος του δικτύου και είναι O(V) όπου V ο αριθμός των κόμβων.

Ο BFS είναι ένας αλγόριθμος αναζήτησης κατά πλάτος, που χρησιμοποιείται αντίστοιχα σε γραφήματα και σε δέντρα. Η αναζήτηση στην συγκεκριμένη περίπτωση ξεκινάει από έναν αρχικό κόμβο που ονομάζεται ρίζα και ξεκινάει την διερεύνηση του κόμβου από τους γειτονικούς και συνεχίζει με τους γειτονικούς των γειτονικών και συνεχίζει με την ίδια διαδικασία μέχρι το τέλος [43]. Η χρονική πολυπλοκότητα και στην συγκεκριμένη περίπτωση είναι O(V) όπου V ο αριθμός των κόμβων.
Τέλος, ο αλγόριθμος Dijkstra είναι ένας αλγόριθμος εύρεσης συντομότερων διαδρομών από κοινή αφετηρία αρκεί να μην υπάρχουν αρνητικά βάρη στις ακμές. Ο αλγόριθμος Dijkstra θεωρείται άπληστος αλγόριθμος επειδή σε κάθε βήμα βρίσκει την τοπική βέλτιστη λύση, μέχρι να ολοκληρωθεί και να βρει την συνολικά βέλτιστη λύση. Η χρονική πολυπλοκότητα του αλγορίθμου Dijkstra είναι O(n^2).

Στο παρελθόν έχουν πραγματοποιηθεί σχετικές έρευνες σχετικά με την επιλογή του κατάλληλου αλγορίθμου για τον βέλτιστο χρόνο. Αρχικά, η σύγκριση έγινε μεταξύ του DFS και του BFS [44]. Σύμφωνα με τους αρθρογράφους, ο Πίνακας 4 περιλαμβάνει τις πηγές και τους προορισμούς (Sources - Destinations) για την δημιουργία του δικτύου με στόχο την εύρεση σε βέλτιστο χρόνο το συντομότερο μονοπάτι:

Πίνακας 4. Τοπολογία DFS & BFS σύμφωνα με [44]

Πηγές (H)	Διαθέσιμες Διαδρομές (S)	Συντομότερες Διαδρομές (S)	Ελάχιστη Απόσταση	Προορισμός(H)
1	[1,6,7,5], [1,8,5],[1,2,3,4,5]	[1,8,5]	2	2
1	[1,8],[1,6,7,5,8], [1,2,3,4,5,8]	[1,8]	1	3
1	[1,2], [1,8,5,4,3,2], [1,6,7,5,4,3,2]	[1,2]	1	4
1	[1,6,7,5,4],[1,2,3,4], [1,8,5,4]	[1,2,3,4]	3	5
2	[5,8], [5,4,3,2,1,8] [5,7,6,1,8]	[5,8]	1	3
2	[5,4,3,2], [5,8,1,2] ,[6,7,6,1,2]	[5,4,3,2]	3	4
2	[5,4], [5,8,1,2,3,4] [5,7,6,1,2,3,4],	[5,4]	1	5
3	[8,5,4,3,2], [8,1,2],	[8,1,2]	2	4

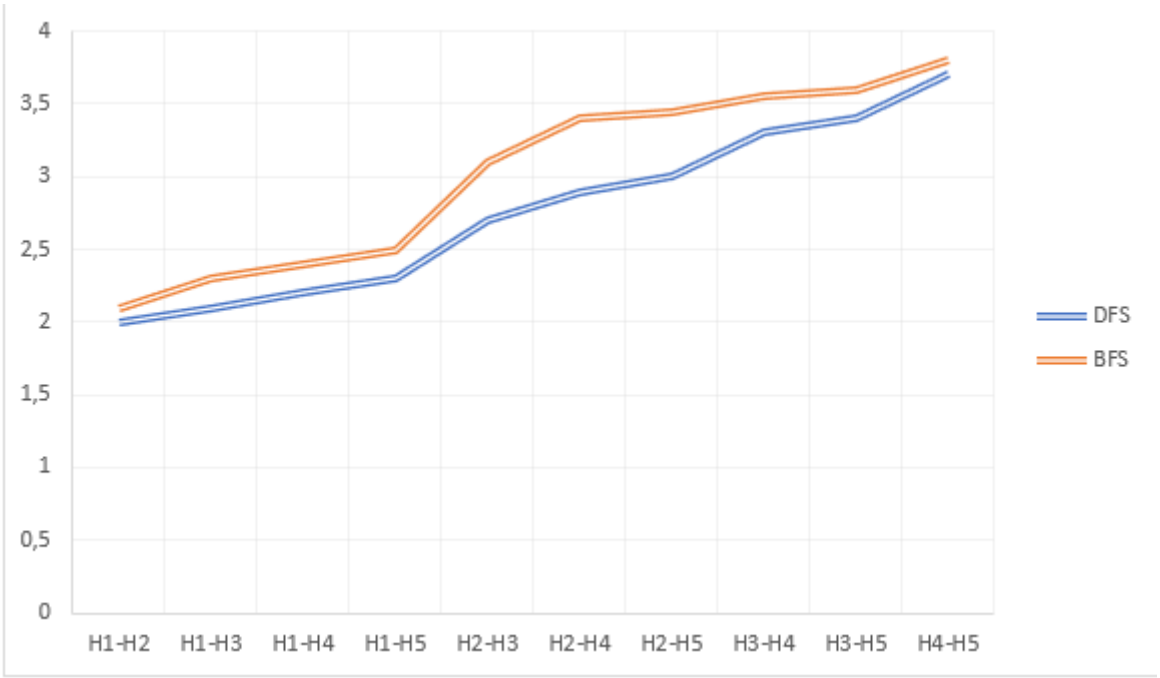
	[8,5,7,6,1,2], [8,1,6,7,5,4,3,2]			
3	[8,5,4], [8,1,2,3,4], [8,5,7,6,1,2,3,4], [8,1,6,7,5,4],	[8,5,4]	2	5
4	[2,3,4], [2,1,6,7,5,4], [2,1,8,5,4]	[2,3,4]	2	5

Αρχικά, οι αρθρογράφοι υπολογίζουν τα μονοπάτια με την χρήση και των δυο αλγορίθμων και παρατηρούμε ότι και στις δύο περιπτώσεις επιλέγονται τα ίδια μονοπάτια απλά με την ανάποδη σειρά. Στην συνέχεια, γίνεται υπολογισμός των βαρών με την χρήση του αλγορίθμου επιλογής συντομότερου μονοπατιού (Open Shortest Path First - OSPF) και πιο συγκεκριμένα με την εξίσωση :

$$WoB(p) = (1 - (WoP(p)/\Sigma WoP(p)[N])) * 10$$

WoB(p) τα βάρος για μια διαδρομή,
WoP(p) το βάρος της διαδρομής σύμφωνα με τον OSPF και
N ο πλήρης αριθμός των διαθέσιμων μονοπατιών.

Έπειτα, επιθυμώντας το μικρότερο Round Trip Time (RTT) και delay λαμβάνονται οι παρακάτω μετρήσεις, στην Εικόνα 7, όσον αφορά το ping ανά ζευγάρι (η μονάδα μέτρησης είναι ms):



Εικόνα 7. Round trip time (ms) ανά ζευγάρι

Συνεπώς, είναι φανερό ότι μεταξύ των δύο ο προτιμότερος αλγόριθμος είναι ο DFS.

Στην συνέχεια, σύγκρινα τον DFS με το Dijkstra αντλώντας δεδομένα από την παρακάτω έρευνα [45]. Στην συγκεκριμένη έρευνα, οι αρθρογράφοι σύγκριναν τον DFS με τον Dijkstra και μια τροποποιημένη μορφή του DFS (DFS Mod) , η οποία είχε σαν στόχο να ολοκληρώνεται η έρευνα του μονοπατιού μόλις βρεθεί ο τελικός προορισμός και να αφαιρείται ο σύνδεσμος και ο κόμβος για την αποφυγή επαναλήψεων.
Για το πρακτικό μέρος χρησιμοποιήθηκε η παρακάτω τοπολογία, όπως φαίνεται στον Πίνακα 5:

Πίνακας 5. DFS & Dijkstra Topology according to [45]

Πηγές(Η)	Προορισμοί(Η)	Βέλτιστα Μονοπάτια (Η)	Ελάχιστη Απόσταση
1	3,5,7,8,6,4,2	[13,6,14],[13,5,1,7,15],[13,5,1,7,16],[13,5,1,2,3,9,17],[13,5,1,2,3,4,10,18],[13,5,1,2,3,11,19],[13,5,1,2,3,4,12,20]	3
3	1,5,7,8,6,4,2	[13,6,14],[14,6,2,8,15],[14,6,2,8,16],[14,6,2,3,9,17],[14,6,2,3,4,10,18],[14,6,2,3,11,20]	3
5	3,1,7,8,6,4,2	[15,7,1,5,14],[15,7,1,5,13],[15,7,16],[15,7,1,2,3,9,17],[15,7,1,2,3,9,18],[15,7,1,2,3,11,19],[15,7,1,2,3,11,20]	3
7	3,5,1,8,6,4,2	[16,8,2,6,14],[16,7,15],[16,8,26,13],[16,8,2,3,9,17],[16,8,2,3,9,18],[16,8,2,3,11,19],[16,8,2,3,11,20]	3
8	3,5,7,1,6,4,2	[17,9,3,2,6,14],[17,9,3,2,8,15],[17,9,3,2,8,16],[17,9,3,2,6,13],[17,9,18],[17,9,	3

		3,11,19],[17,9,3,11,20]	
6	3,5,7,8,1,4,2	[18,9,3,2,6,14],[18,9,3,2,8,15],[18,9,3,2,8,16],[18,10,17],[18,9,3,2,6,13],[18,9,3,11,19],[18,10,4,12,20]	3
4	3,5,7,8,6,1,2	[19,11,3,2,6,14],[19,11,3,2,8,15],[19,11,3,2,8,16],[19,11,3,9,17],[19,11,3,9,18],[19,11,3,2,6,13],[19,12,20]	3
2	3,5,7,8,6,4,1	[20,11,3,2,6,14],[20,11,3,2,8,15],[20,11,3,2,8,16],[20,11,3,9,17],[20,11,3,9,18],[20,11,19],[20,11,3,2,6,13]	3

Για την λήψη αποτελεσμάτων σχετικά με τους χρόνους χρησιμοποιήθηκε ο Host 5 (h5) ως διακομιστής και ο Host 6 (h6) ως χρήστης, για να γίνει η προσομοίωση κίνησης UDP μέσω 10 παράλληλων συνδέσεων και ταχυτήτων μεταξύ 20-100 Mbit, σε διάστημα χρόνου 120 δευτερολέπτων. Στην συνέχεια, ο κάθε αλγόριθμος παρήγαγε το ιδανικό μονοπάτι το οποίο στην περίπτωση του τροποποιημένου DFS και του Dijkstra είναι ίδια και είναι τα [15, 7, 1, 2, 9, 18] ενώ στον απλό DFS είναι το [15, 8, 3, 4, 10, 18]. Αυτό δεν επιφέρει μεγάλη διαφορά εφόσον και στις δύο περιπτώσεις η ελάχιστη απόσταση είναι η σωστή και είναι ίση με 6. Στην συνέχεια, μετρήθηκαν οι χρόνοι εύρεσης μονοπατιού και για τους 3 αλγορίθμους και τα αποτελέσματα παρατίθενται στην Εικόνα 8:

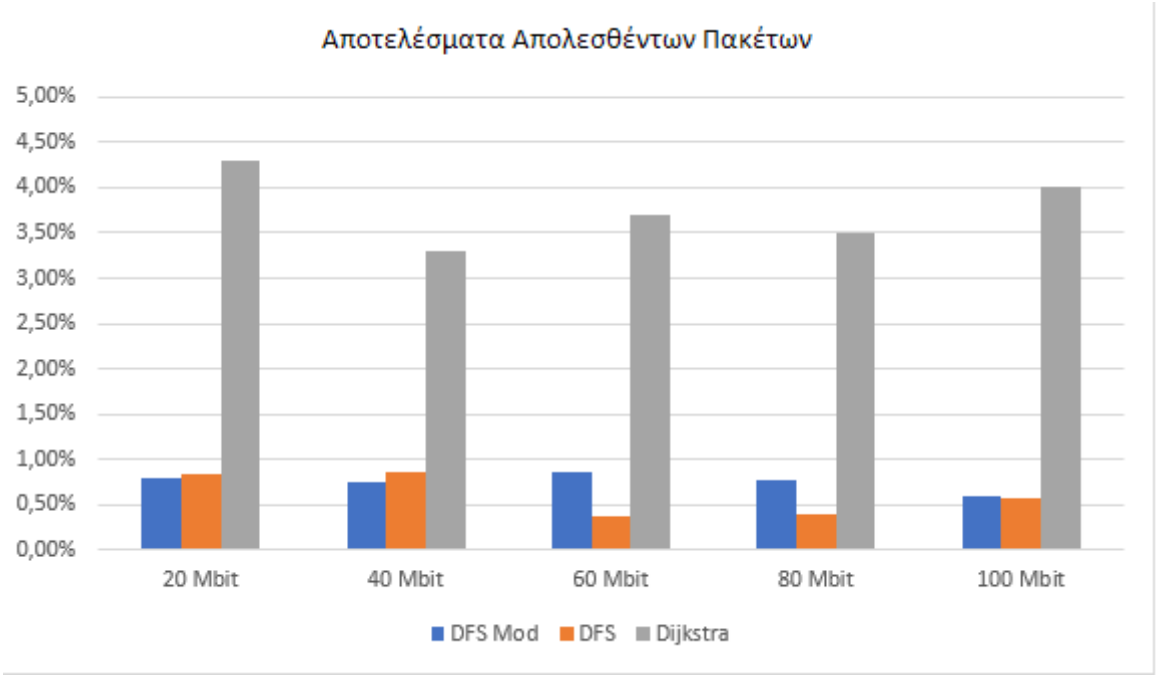
Εικόνα 8. Execution Time (ms).

Όπως είναι φανερό για όλες τις τιμές ο DFS Mod και ο απλός DFS πέτυχαν τους καλύτερους χρόνους με μικρές διαφορές μεταξύ τους, ενώ ο Dijkstra είχε αρκετά μεγάλη απόκλιση. Στην συνέχεια, έγιναν αντίστοιχες μετρήσεις για να μετρηθεί η ρυθμοαπόδοση (Throughput) αναλόγως την κίνηση του καναλιού (Traffic load) του δικτύου. Τα αποτελέσματα των μετρήσεων φαίνονται στον Πίνακα 6:

Πίνακας 6. Αποτελέσματα Ρυθμοαπόδοσης

Traffic Load (Mbit)	***20***	***40***	***60***	***80***	***100***	***Average (Kbps)***
DFS Mod (Kbps)	3722	6119	4846	5409	4008	**4820**
DFS(Kbps)	6124	5239	2816	4147	5151	**4695**
Dijkstra(Kbps)	3583	4328	4758	4883	4571	**4424**

Συνεπώς, όπως συμπεραίνουμε από τον παραπάνω πίνακα οι αλγόριθμοι DFS έχουν τα καλύτερα αποτελέσματα. Τέλος, έγινε σύγκριση όσον αφορά την απώλεια πακέτων μεταξύ των τριών αλγορίθμων αναλόγως το traffic load.
Τα αποτελέσματα παρουσιάζονται στην Εικόνα 9:



Εικόνα 9. Σύγκριση των αποτελεσμάτων των αλγορίθμων ως προς την απώλεια πακέτων

Όπως φαίνεται στο παραπάνω διάγραμμα ο απλός DFS έχει την μικρότερη απώλεια πακέτων στις περισσότερες περιπτώσεις, ακολουθούμενος από τον DFS Mod και στο τέλος ο Dijkstra με την αρκετά μεγαλύτερη απώλεια πακέτων (packet loss).

Μετά από την θεωρητική μελέτη που έγινε στους αλγόριθμους DFS, BFS και Dijkstra, παίρνοντας ως κριτήρια σύγκρισης η καθυστέρηση, η ρυθμοαπόδοση, ο χρόνος εκτέλεσης (execution time) και την απώλεια πακέτων μπορεί να εξαχθεί το συμπέρασμα ότι στην περίπτωση που μελετάμε η καλύτερη λύση δίνεται με την χρήση του DFS.

Ως προς το κομμάτι του QoS το οποίο θα υλοποιηθεί προκειμένου να δοθεί λύση στην επιλογή και δρομολόγηση της κρίσιμης κίνησης πριν από την υπόλοιπη κίνηση του δικτύου, όπως έχει αναλυθεί και νωρίτερα θα χρησιμοποιήσουμε τον μηχανισμό του DSCP, καθώς κάνει με τον βέλτιστο τρόπο την δημιουργία ουρών με συγκεκριμένες τιμές στις οποίες παρέχεται η ανάλογη προτεραιότητα.

3.5 Μοντελοποίηση

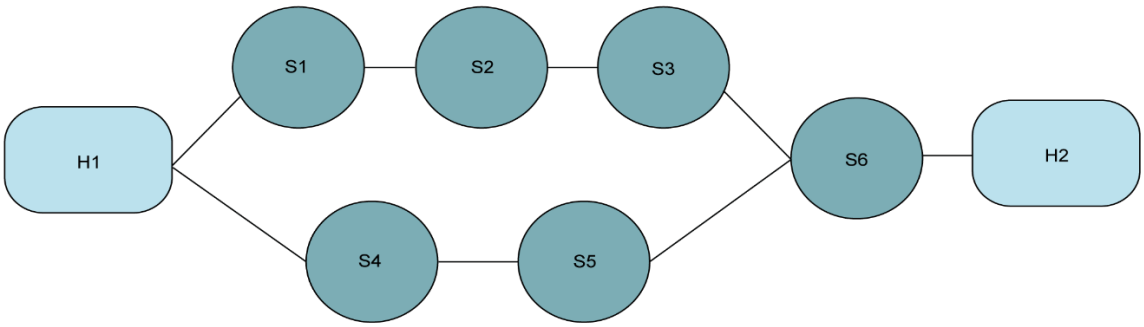
Με βάση την ανάλυση που έγινε στην παραπάνω ενότητα, η μοντελοποίηση αρχίζει με την υλοποίηση του αλγορίθμου DFS. Μαζί με τον DFS θα χρησιμοποιηθούν επίσης ο αλγόριθμος OSPF και οι βιβλιοθήκες του ελεγκτή Ryu για το QoS, προκειμένου να γίνει σωστή δρομολόγηση της κρίσιμης κίνησης.

Αρχικά, με τον DFS εξετάζονται οι πιθανές κορυφές ή πηγές του γραφήματος εντοπίζοντας πρώτα την βαθύτερη πηγή και στην συνέχεια με την μέθοδο

οπισθοδρόμησης (backtracking) εντοπίζονται πιθανές κορυφές χρησιμοποιώντας μια στοίβα. Με αυτόν τον τρόπο λειτουργίας του αλγορίθμου, μπορούμε και κάνουμε μικρές αλλαγές στον αλγόριθμο για να βρίσκει όλα τα πιθανά μονοπάτια μεταξύ δύο κορυφών και να λύσουμε το πρόβλημα της δρομολόγησης πολλών διαδρομών. Συνοπτικά δηλαδή, μπορούμε να βρούμε όλα τα μονοπάτια μεταξύ δύο κόμβων.

Παραδείγματος χάρη αν έχουμε την παρακάτω τοπολογία, όπως φαίνεται στην Εικόνα 10, μπορούμε να χρησιμοποιήσουμε τον DFS και θα βρει τα πιθανά μονοπάτια:

1. S1-S2-S3-S6
2. S4-S5-S6



Εικόνα 10. Τοπολογία Παραδείγματος

Ο αλγόριθμος DFS, όπως αυτός φαίνεται στην Εικόνα 11, αφού εκτελεστεί, μας επιστρέφει τα πιθανά μονοπάτια χωρίς όμως να λαμβάνει υπόψη κάποιο βάρος. Συνεπώς, για να υπολογιστεί το βέλτιστο μονοπάτι, πρέπει να εισαχθούν βάρη στον κάθε σύνδεσμο μεταξύ κόμβων ή σε κάθε ολοκληρωμένη διαδρομή. Μια λύση για την εύρεση των βαρών θα ήταν να συγκριθεί το εύρος ζώνης κάθε μονοπατιού.

```
def get_paths(self, src, dst):

    # DFS algorith to obtain all the available paths
    if src == dst:
        return [[src]]      #if src is the same as the destination
    paths = []
    stack = [(src, [src])]
    while stack:
        (node, path) = stack.pop()
        for next in set(self.adjacency[node].keys()) - set(path):
            if next is dst:
                paths.append(path + [next])
            else:
                stack.append((next, path + [next]))
    print ("All the accesible paths from ", src, " to ", dst, " : ", paths) #prints the available paths
    return paths
```

Εικόνα 11. Get Paths Function

Έχοντας χρησιμοποιήσει τις δυνατότητες του DFS, συνεχίζεται η υλοποίηση στην ανάλυση του OSPF. Για τον υπολογισμό του κόστους των μονοπατιών ο τρόπος που ακολούθησα είναι ο υπολογισμός του κόστους συνδέσμων με τον ίδιο τρόπο όπως στον αλγόριθμο επιλογής συντομότερου μονοπατιού και στην συνέχεια την προσθήκη των βαρών των συνδέσμων που αποτελούν ένα μονοπάτι. Πιο συγκεκριμένα, ο OSPF, βρίσκει τις συντομότερες διαδρομές σε όλους τους γνωστούς προορισμούς. Κατά την διαδικασία της έναρξης, ο δρομολογητής δημιουργεί μια αναπαράσταση του συνόλου των συσκευών και την κατάσταση τους. Όλοι οι δρομολογητές, στη συνέχεια, ανταλλάζουν την κατάσταση την οποία λαμβάνουν, την αποθηκεύουν και την μεταφέρουν για να ανανεώσουν τις πληροφορίες από τους υπόλοιπους δρομολογητές. Όσο δεν υπάρχουν αλλαγές στην τοπολογία του δικτύου και στην κατάσταση των συσκευών εντός δικτύου, δεν χρειάζεται αλλαγή βαρών, ωστόσο, αν υπάρξει κάποια αλλαγή υπολογίζονται τα νέα βάρη από την αρχή. Ο κάθε δρομολογητής εντός του δικτύου τοποθετείται σαν “ρίζα” του δικτύου και υπολογίζει την ελάχιστη διαδρομή που απαιτείται για να φτάσει στον τελικό προορισμό. Με αυτόν τον τρόπο, κάθε δρομολογητής θα έχει τη δική του αντίληψη σχετικά με την συντομότερη διαδρομή ωστόσο θα υπάρχει και συνολικά σαν δίκτυο οι συντομότερες διαδρομές από και προς έναν προορισμό. Το κόστος μιας διεπαφής στον OSPF ορίζεται ως η ένδειξη του αριθμού των πακέτων που απαιτούνται για την αποστολή προς έναν προορισμό και είναι αντιστρόφως ανάλογο του εύρους ζώνης αυτής της διεπαφής. Συνεπώς, όσο μεγαλύτερο είναι το εύρος ζώνης τόσο μικρότερο είναι το κόστος. Η μαθηματική έκφραση που χρησιμοποιείται παραδείγματος χάρη για μια γραμμή συγκεκριμένη χωρητικότητας είναι [46]:

Κόστος = χωρητικότητα γραμμής / bandwith in bps.

Τέλος, για τον υπολογισμό του συνολικού βάρους ενός μονοπατιού αρκεί να προστεθούν τα συνολικά κόστη των συνδέσμων του κάθε μονοπατιού.

```
def get_link_cost(self, switch1, switch2):

   #Find the costs of the network for every combination of switches
    cost1 = self.adjacency[switch1][switch2]
    cost2 = self.adjacency[switch2][switch1]
    bl = min(self.bandwidths[switch1][cost1], self.bandwidths[switch2][cost2])
    ew = REFERENCE_BW/bl
    return ew #returns link cost
```

Εικόνα 12. Get Link Cost Function

```
def get_path_cost(self, path):

    #Calculates the cost of each path
    cost = 0
    for i in range(len(path) - 1):
        cost += self.get_link_cost(path[i], path[i+1])
    return cost
```

Εικόνα 13. Get Path Cost Function

```
def get_optimal_paths(self, src, dst):

    #Finds the best available path
    paths = self.get_paths(src, dst)
    paths_count = len(paths) if len(
        paths) < MAX_PATHS else MAX_PATHS
    return sorted(paths, key=lambda x: self.get_path_cost(x))[0:(paths_count)] #best path for each case
```

Εικόνα 14. Get Optimal Paths Function

Στην συνέχεια, πρέπει να προσαρμοστεί ο τρόπος λειτουργίας του δικτύου για την συνεργασία του πρωτοκόλλου Openflow με τον ελεγκτή Ryu. Για την υλοποίηση αυτή, επιλέξαμε ο ελεγκτής να είναι υπεύθυνος για την δρομολόγηση της κίνησης, σύμφωνα με τα μονοπάτια που βρέθηκαν στο προηγούμενο βήμα, αφού πρώτα τα "εγκαταστήσει" σε κάθε μεταγωγέα. Χρησιμοποιώντας τις βασικές αρχές των πρωτοκόλλων IP, Address Resolution Protocol (ARP), Ethernet συνεχίζουμε την δρομολόγηση της επιθυμητής κίνησης.

Μια ροή στην περίπτωση μας ορίζεται ως μια σειρά ενεργειών που θα πραγματοποιηθούν αν πληρούνται συγκεκριμένα κριτήρια ενός πακέτου του δικτύου. Παραδείγματος χάρη, αν ένα πακέτο έχει μια διεύθυνση αποστολέα με IP 10.0.0.1 και διεύθυνση παραλήπτη με IP 10.0.0.2 (τα κριτήρια), το πακέτο θα δρομολογηθεί απευθείας σε συγκεκριμένη ροή ή θα απορριφθεί ή θα εφαρμοστεί σε αυτό οποιαδήποτε άλλη ενέργεια όπως αλλαγή της κεφαλίδας του πακέτου ή δρομολόγηση σε συγκεκριμένη ουρά.

Στην δική μας περίπτωση, τα κριτήρια που χρησιμοποιήθηκαν είναι η αναγνώριση της IP του πακέτου όπως και της MAC αυτού. Παρακάτω, στην Εικόνα 15, απεικονίζεται ο αντίστοιχος κώδικας:

```
for in_port in ports:

    match_ip = ofp_parser.OFPMatch(
        eth_type=0x0800, # IPv4 value according to EtherType
        ipv4_src=ip_src,
        ipv4_dst=ip_dst
    )
    match_arp = ofp_parser.OFPMatch(
        eth_type=0x0806, # ARP value according to EtherType
        arp_spa=ip_src,
        arp_tpa=ip_dst
    )

    out_ports = ports[in_port]
```

Εικόνα 15. Κριτήρια αναγνώρισης πακέτων

Πιο συγκεκριμένα, αναγνωρίζουμε ένα πακέτο από τον τελικό του προορισμό (με την χρήση του ipv4_dst). Για την επιλογή των πρωτοκόλλων που επιθυμούμε χρησιμοποιούμε το EtherType. Το EtherType είναι πεδίο του πλαισίου Ethernet που χρησιμοποιείται για να μας δείξει ποιο πρωτόκολλο βρίσκεται στο πακέτο που επιθυμούμε για να καταλήξουμε στον τρόπο επεξεργασίας και ενεργειών που θα πραγματοποιηθούν στο κατάλληλο πακέτο. Υπάρχει πλήθος τιμών για πάρα πολλά πρωτόκολλα με μερικά ενδεικτικά παραδείγματα να είναι τα παρακάτω όπως φαίνονται στον Πίνακα 7 [47]:

Πίνακας 7. Τιμές EtherType για συγκεκριμένα πρωτόκολλα

Protocol	EtherType (hexadecimal)
Internet Protocol version 4 (IPv4)	0x0800
Address Resolution Protocol (ARP)	0x0806
Wake-on-LAN[9]	0x0842
Audio Video Transport Protocol (AVTP)	0x22F0
IETF TRILL Protocol	0x22F3

Στην δική μας περίπτωση, αναγνωρίζονται μόνο τα πακέτα Internet Protocol version 4 (IP) και (ARP) για αυτό τον λόγο χρησιμοποιούνται στο πεδίο eth_type και οι τιμές τους (0x0800 και 0x0806 αντίστοιχα όπως φαίνεται στον Πίνακα 7).

Το ARP θα χρησιμοποιηθεί επειδή δίνει την δυνατότητα της εύρεσης των διευθύνσεων MAC για μια συγκεκριμένη IP, γεγονός το οποίο θα βοηθήσει για την τοπολογία του δικτύου, καθώς βασίζεται σε Layer 2. Πρακτικά, είναι το πρώτο πακέτο που στέλνεται μεταξύ 2 hosts για την επικοινωνία εντός του δικτύου οπότε πρέπει να έχει ανάλογη αντιμετώπιση για την αποφυγή προβλημάτων (π.χ. ARP Flooding το οποίο θα καταστήσει το δίκτυο μη λειτουργικό).

Έπειτα, δημιουργούμε τα group tables. Τα group tables αποτελούνται από καταχωρήσεις ομάδων (group entries). Η δυνατότητα μιας καταχώρησης ροής να οδηγείται σε μια ομάδα δίνει την δυνατότητα στο Openflow για έξτρα μεθόδους προώθησης. Κάθε καταχώρηση ομάδας αποτελείται το group identifier, group type, μετρητή και action buckets. Ο group identifier είναι ένας ακέραιος 32 bit που προσδιορίζει το group στον μεταγωγέα Openflow. Ο μετρητής ανανεώνεται όταν τα πακέτα επεξεργάζονται σε κάποιο group. Τέλος, το κάθε action bucket περιέχει ένα σύνολο από ενέργειες προς εκτέλεση με σχετικές παραμέτρους. Συνοπτικά, τα group tables χρησιμοποιούνται για τον ορισμό ή την εφαρμογή πολλαπλών ενεργειών για μια συγκεκριμένη ροή αναλόγως την περίσταση και στην συνέχεια, βρίσκουμε τον χρόνο που χρειάζεται για την εγκατάσταση του μονοπατιού όπως φαίνεται παρακάτω στην Εικόνα 16:

```
            for port, weight in out_ports:
                bucket_weight = int(round((1 - weight/sum_of_pw) * 10))
                bucket_action = [ofp_parser.OFPActionOutput(port)]
                buckets.append(
                    ofp_parser.OFPBucket(
                        weight=bucket_weight,
                        watch_port=port,
                        watch_group=ofp.OFPG_ANY,
                        actions=bucket_action
                    )
                )

            if group_new:
                req = ofp_parser.OFPGroupMod(
                    dp, ofp.OFPGC_ADD, ofp.OFPGT_SELECT, group_id,
                    buckets
                )
                dp.send_msg(req)
            else:
                req = ofp_parser.OFPGroupMod(
                    dp, ofp.OFPGC_MODIFY, ofp.OFPGT_SELECT,
                    group_id, buckets)
                dp.send_msg(req)

            actions = [ofp_parser.OFPActionGroup(group_id)]

            self.add_flow(dp, 32768, match_ip, actions)
            self.add_flow(dp, 1, match_arp, actions)

        elif len(out_ports) == 1:
            actions = [ofp_parser.OFPActionOutput(out_ports[0][0])]

            self.add_flow(dp, 32768, match_ip, actions)
            self.add_flow(dp, 1, match_arp, actions)
print ("Path installation finished in ", time.time() - computation_start )  #the time needed for the path installation
return paths_with_ports[0][src][1]
```

Εικόνα 16. Group table & Path time calculation

Ορίζουμε τα buckets, τα οποία είναι οι μεμονωμένες λίστες ενεργειών του group table, στα οποία εισάγουμε διάφορες ενέργειες εξόδου για κάθε θύρα που έχει κάποια διαδρομή προς το group table. Παραδείγματος χάρη, για την τοπολογία του Σχήματος 6, ο Host 2 για να φτάσει τον Host 1 στέλνει πρώτα ένα πακέτο στον S6. Από εκεί το S6 έχει δύο διαθέσιμα μονοπάτια να επιλέξει και συνεπώς 2 διαθέσιμες θύρες (μία για το κάθε μονοπάτι), οπότε ορίζουμε port 1 και port 2 σαν θύρες εξόδους για τον S6 στο group table. Τα μονοπάτια αυτά στο OpenFlow χαρακτηρίζονται ως group actions.
Ένα από τα στοιχεία που απαιτούνται για την εισαγωγή στοιχείων στο group table είναι το bucket weight. Για τον υπολογισμό του χρησιμοποιήθηκε η μέθοδος όπως είχε αρχικά αναλυθεί στην παρακάτω έρευνα σχετικά με την εύρεση πιθανών και αποτελεσματικών μονοπατιών σε ένα αβέβαιο δίκτυο [48]. Στην συγκεκριμένη έρευνα, καταλήγουν ότι οι προτεινόμενες μέθοδοι ακριβούς υπολογισμού πιθανοτήτων απαιτούν μεγάλο αριθμό υπολογισμών. Συνεπώς προτείνουν την ανάπτυξη προσεγγιστικών αλγορίθμων. Στην παραπάνω περίπτωση σχετικά με τον πολυωνυμικό αλγορίθμου βασισμένο σε ευρετική συνάρτηση με ιεραρχική κατανομή σε δέντρα αξιολογούν ότι απαντάει στο πρόβλημα που θέτουν. Στην συνέχεια, η παραπάνω έρευνα επεκτείνεται για την δημιουργία της παρακάτω εξίσωσης, την οποία χρησιμοποιούμε στον κώδικα, για τον υπολογισμό των μεμονωμένων βαρών των μονοπατιών για όλα τα διαθέσιμα μονοπάτια [44]:

$$w_0 B\{p\} = \left(1 - \left(\frac{w_0 P}{\Sigma w_0 P(P)[N]}\right)\right) * 10$$

Εξίσωση 17. Εξίσωση Υπολογισμού Βαρών

Όπου:

WoB(p) → Bucket Weight το οποίο παίρνει τιμές από 0 έως 10.
WoP → Το βάρος του μονοπατιού όπως παρουσιάστηκε παραπάνω.
N → Το σύνολο των διαθέσιμων μονοπατιών.

Στην ουσία η παραπάνω εξίσωση βρίσκει την αναλογία του βάρους ενός μονοπατιού p προς το συνολικό βάρος των διαθέσιμων μονοπατιών.
Στην πραγματικότητα, αναζητείται το μονοπάτι με όσο το δυνατόν μικρότερες τιμές βάρους. Επειδή όμως το πρωτόκολλο Openflow δίνει προτεραιότητα στα buckets με τις μεγαλύτερες τιμές για αυτόν τον λόγο κάνουμε και την αφαίρεση και καταλήγουμε σε ένα αποτέλεσμα. Το πρωτόκολλο επιλέγει τα μεγαλύτερα αποτελέσματα, καθώς συνεπάγεται ότι έχουν το μεγαλύτερο περιθώριο να διαχειριστούν κίνηση.

Επιστρέφουμε στο παράδειγμα της εικόνας 10, όπου αποτυπώνεται η τοπολογία του παραδείγματος, υποθέτοντας ότι το βάρος κάθε συνδέσμου ισούται με 1. Για την πρώτη διαδρομή και αντίστοιχα port έχουμε:

P1 = (S6 - S5) + (S5 - S4) = 2 και

P2 = (S6 - S3) + (S3 - S2) + (S2 - S1) =3.

Έπειτα βρίσκουμε τα bucket weight και των δύο περιπτώσεων τα οποία είναι:

BW1 = (1 - ⅖) *10 = 6 και
BW2 = (1 - ⅗) *10 = 4.

Συνεπώς, όπως φαίνεται και στο σχήμα εύκολα καταλαβαίνουμε πως το μικρότερο μονοπάτι (στην περίπτωση μας το S4 - S5 - S6) έχει το μεγαλύτερο bucket weight.

Έχοντας λύσει το πρόβλημα της εύρεσης της τοπολογίας και συγκεκριμένα της βέλτιστης διαδρομής πλέον μένει να λύσουμε το πρόβλημα της επιλογής της κρίσιμης κίνησης με σκοπό την επίτευξη υψηλού QoS. Για το σκοπό αυτό, θα χρησιμοποιηθεί το DSCP το οποίο ως πληροφορία υπάρχει ήδη σε κάθε πακέτο συνεπώς δεν επιβαρύνει το δίκτυο με περαιτέρω πληροφορία. Υπάρχουν πολλοί τρόποι εύρεσης και χρήσης της τιμής DSCP από τα πακέτα, αλλά θα επιλεχθούν οι υπηρεσίες και το υλικό που προσφέρει ο Ryu Ελεγκτής σύμφωνα με τον οδηγό και την βοήθεια του Ryubook [49]. Αρχικά, θα χρησιμοποιήσουμε τα αρχεία που προσφέρονται από τον ryu τα οποία είναι: ryu/ryu/app/rest_qos.py, ryu/ryu/app/rest_conf_switch.py και ryu/ryu/app/qos_rest_router.py [49].

- RyuApp: rest_qos.

  Το συγκεκριμένο RyuApp βοηθάει για τον ορισμό και αφαίρεση κανόνων, που αφορούν την ποιότητα των υπηρεσιών και πληροφορίες σχετικά με την κατάσταση της ουράς του φορτίου.[49]

- RyuApp: rest_conf_switch

  Το συγκεκριμένο RyuApp βοηθάει στην υλοποίηση μιας υπηρεσίας REST υπεύθυνη για την διαμόρφωση των μεταγωγέων.[49]

- RyuApp: qos_rest_router

  Το συγκεκριμένο RyuApp βοηθάει με το να είναι υπεύθυνο για να λαμβάνει, να ορίζει και να διαγράφει δεδομένα διεύθυνσης και δεδομένα δρομολόγησης. Η μόνη αλλαγή που έγινε είναι η κατοχύρωση της εισόδου ροής (flow entry) στον πίνακα με ID 1.

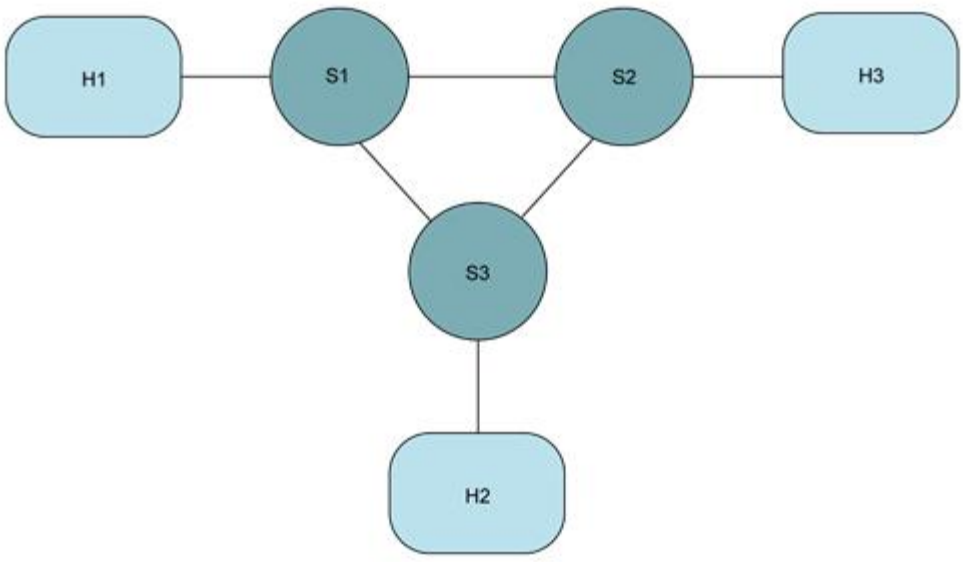
Τέλος, στην παραπάνω ενότητα δημιουργήθηκε και αναλύθηκε η αλγοριθμική δομή πάνω στην οποία θα βασιστεί η υλοποίηση της διπλωματικής εργασίας στην επόμενη

ενότητα. Πιο συγκεκριμένα, η αφετηρία ήταν ο αλγόριθμος DFS για την εύρεση των μονοπατιών, έπειτα, χρησιμοποιήθηκε η τεχνική του OSPF για τον υπολογισμό των βαρών με βάση το εύρος ζώνης και τέλος χρησιμοποιήθηκαν οι βιβλιοθήκες του Ryu για την υλοποίηση του QoS μέσω του DSCP.

3.6 Πρακτική Εφαρμογή

Στο κομμάτι της εφαρμογής θα εκτελεστεί το κομμάτι του κώδικα που είναι υπεύθυνο για την εξισορρόπηση του φόρτου του δικτύου και στην συνέχεια το κομμάτι για το QoS. Για το πρώτο κομμάτι, θα χρησιμοποιηθούν οι αλγόριθμοι DFS και OSPF οι οποίοι εκτελούνται τοπικά σε ένα VM και προγραμματίζουν τον ελεγκτή ώστε στο τέλος αυτών να έχει υπολογισμένη την βέλτιστη διαδρομή. Σε δεύτερο χρόνο, υλοποιείται και η εξισορρόπηση του φόρτου του δικτύου. Σχετικά με το QoS θα χρησιμοποιηθούν οι διαθέσιμες εφαρμογές του Ryu ελεγκτή, που αναφέρθηκαν στην προηγούμενη υποενότητα.

Αρχικά, εκτελείται ο κώδικας που περιεγράφηκε στην ενότητα 3.4 σχετικά με την εύρεση της βέλτιστης τοπολογίας.
Έστω μια τυχαία τοπολογία δικτύου, όπως φαίνεται στο σχήμα της Εικόνας 18:



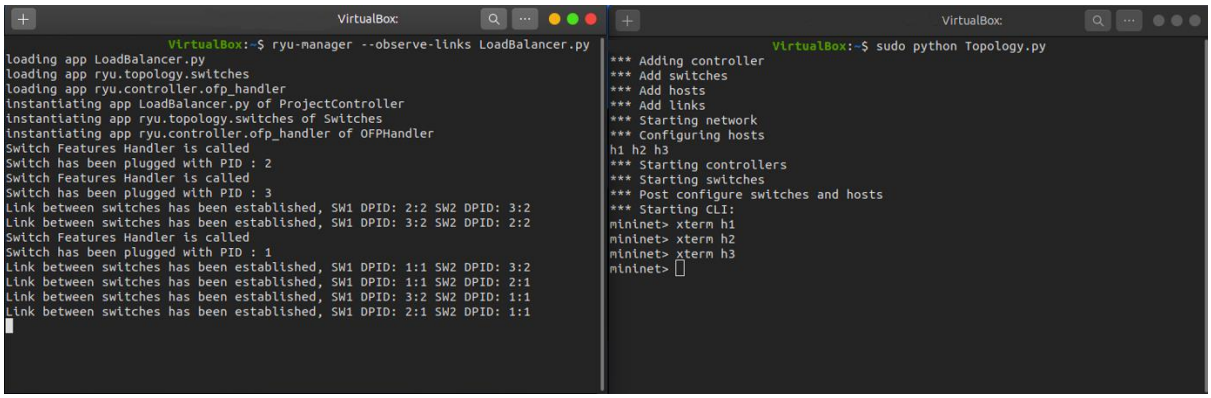
Εικόνα 18. Χρησιμοποιούμενη Τοπολογία

Τα διαθέσιμα μονοπάτια της τοπολογίας, όπως φαίνονται στον Πίνακα 8, είναι:

Πίνακας 8. Μονοπάτια της Τοπολογίας

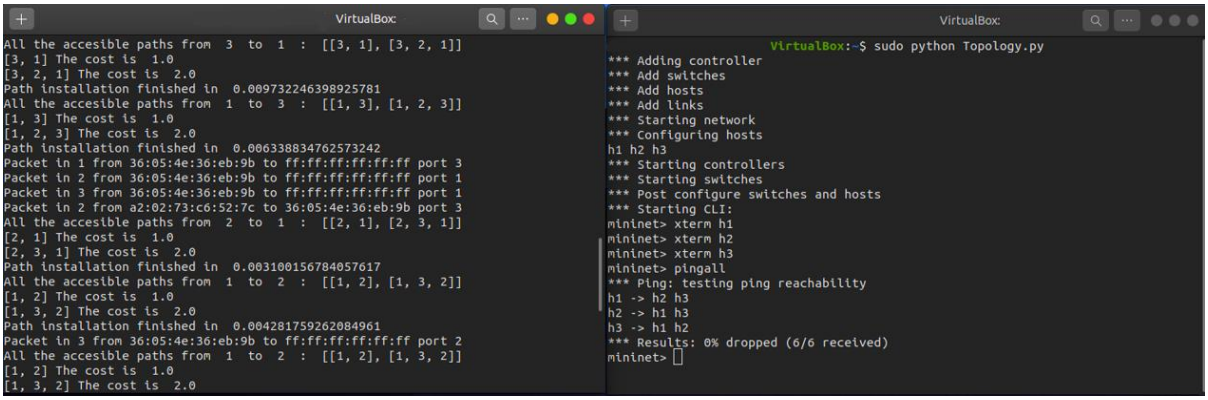
	H1 – H2	H2 – H3	H1 – H3
Βέλτιστη Διαδρομή	S1 – S3	S3 – S2	S1 – S2
Δευτερεύουσα Διαδρομή	S1 – S2 – S3	S3 – S1 – S2	S1 – S3 – S2

Το στιγμιότυπο του παραθύρου της γραμμής εντολών, μετά την εκτέλεση, είναι το παρακάτω.



Εικόνα 19. Σύνδεση Μεταγωγέων

Παρατηρούμε στην Εικόνα 19, ότι προστέθηκαν με επιτυχία οι μεταγωγείς και πραγματοποιούνται οι κατάλληλες συνδέσεις.
Έπειτα θα ελέγξουμε αν το δίκτυο βρίσκεται σε σωστή λειτουργία με την εντολή pingall. Από την εκτέλεση αυτή προκύπτουν δύο εκτυπώσεις. Η πρώτη (αριστερά) είναι η εκτύπωση του Load Balancer και η δεύτερη (δεξιά) είναι η εκτύπωση του mininet.



Εικόνα 20. Εκτύπωση Διαθέσιμων Μονοπατιών, Βαρών και Χρόνου Απόκρισης

Αναλύοντας την Εικόνα 20 από τα αριστερά, εμφανίζονται αρχικά όλα τα διαθέσιμα μονοπάτια, όπως τα υπολογίζει ο αλγόριθμος μας, από τον h1 – h3 και αντίστροφα, το βέλτιστο διαθέσιμο μονοπάτι, το κόστος του κάθε μονοπατιού και ο χρόνος που απαιτείται για την εγκατάσταση των μονοπατιών. Έπειτα, με αντίστοιχο τρόπο εμφανίζονται οι ίδιες πληροφορίες για τον h2 – h3.

Συνεχίζοντας προς τα δεξιά, ο συνδυασμός του Mininet και της εντολής pingall μας προσφέρει την δημιουργία του εικονικού δικτύου και την επιβεβαίωση της επιτυχημένης σύνδεσης των hosts, καθώς η εντολή pingall δεν έχει καμία απώλεια πακέτων (*** Results: 0% dropped (6/6 received)).

Στην συνέχεια, ελέγχουμε σε ένα τυχαίο μεταγωγέα (s2) αν έχει γίνει σωστή εγκατάσταση των ροών με την παρακάτω εντολή:

sudo ovs-ofctl -O OpenFlow13 dump-flows s2

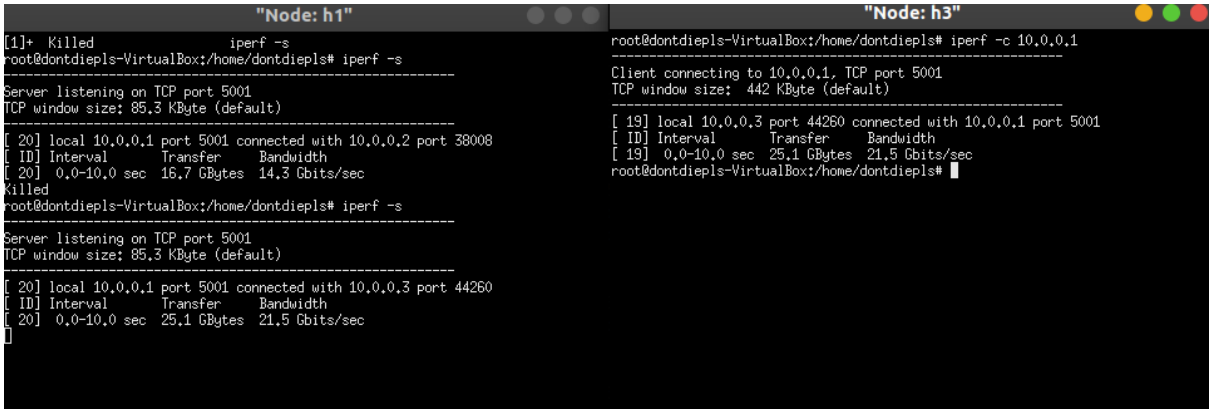
και έχουμε τα παρακάτω αποτελέσματα:

```
cookie=0x0, duration=2161.907s, table=0, n_packets=4715, n_bytes=282900, priority=65535,dl_dst=01:80:c2:00:00:0e,dl_type=0x88cc actions=CONTROLLER:65535
cookie=0x0, duration=903.094s, table=0, n_packets=0, n_bytes=0, ip,nw_src=10.0.0.2,nw_dst=10.0.0.1 actions=output:"s2-eth1"
cookie=0x0, duration=903.094s, table=0, n_packets=3, n_bytes=126, priority=1,arp,arp_spa=10.0.0.2,arp_tpa=10.0.0.1 actions=output:"s2-eth1"
cookie=0x0, duration=903.094s, table=0, n_packets=0, n_bytes=0, ip,nw_src=10.0.0.1,nw_dst=10.0.0.2 actions=output:"s2-eth2"
cookie=0x0, duration=903.094s, table=0, n_packets=1, n_bytes=42, priority=1,arp,arp_spa=10.0.0.1,arp_tpa=10.0.0.2 actions=output:"s2-eth2"
cookie=0x0, duration=903.049s, table=0, n_packets=2, n_bytes=196, ip,nw_src=10.0.0.1,nw_dst=10.0.0.3 actions=output:"s2-eth3"
cookie=0x0, duration=903.049s, table=0, n_packets=2, n_bytes=84, priority=1,arp,arp_spa=10.0.0.1,arp_tpa=10.0.0.3 actions=output:"s2-eth3"
cookie=0x0, duration=903.049s, table=0, n_packets=2, n_bytes=196, ip,nw_src=10.0.0.3,nw_dst=10.0.0.1 actions=group:2109611615
cookie=0x0, duration=903.049s, table=0, n_packets=2, n_bytes=84, priority=1,arp,arp_spa=10.0.0.3,arp_tpa=10.0.0.1 actions=group:2109611615
cookie=0x0, duration=903.042s, table=0, n_packets=2, n_bytes=196, ip,nw_src=10.0.0.2,nw_dst=10.0.0.3 actions=output:"s2-eth3"
cookie=0x0, duration=903.042s, table=0, n_packets=2, n_bytes=84, priority=1,arp,arp_spa=10.0.0.2,arp_tpa=10.0.0.3 actions=output:"s2-eth3"
cookie=0x0, duration=903.042s, table=0, n_packets=2, n_bytes=196, ip,nw_src=10.0.0.3,nw_dst=10.0.0.2 actions=group:3834021616
cookie=0x0, duration=903.042s, table=0, n_packets=2, n_bytes=84, priority=1,arp,arp_spa=10.0.0.3,arp_tpa=10.0.0.2 actions=group:3834021616
```

Εικόνα 21. Action Fields

Παρατηρούμε ότι ο host 3 με IP 10.0.0.3 στον οποίο είναι συνδεδεμένος στον μεταγωγέα 2 φτάνει στον host 1 (10.0.0.1) και στον host 2 (10.0.0.2), όπως φαίνεται στην Εικόνα 21. Επειδή υπάρχουν αρκετά ports και όπως αναφέρθηκε στην ενότητα 3.5, δημιουργήθηκαν τα αντίστοιχα group actions. Οι τιμές των action fields έχουν id 2109611615 και 3834021616 για την κίνηση από τον h3 προς το h1 και για την κίνηση από τον h3 προς τον h2 αντίστοιχα. Αυτό σημαίνει ότι η ενέργεια που θα εφαρμοστεί στις ροές αυτές είναι ένα group action με τα αντίστοιχα ids (2109611615 και 3834021616).

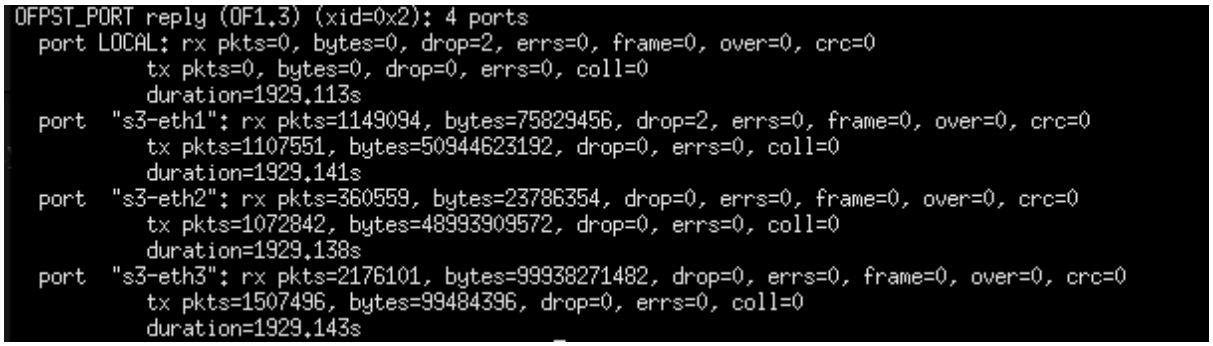
Στην συνέχεια, αφού ελέγξαμε ότι η κίνηση δεν έχει κάποιο πρόβλημα θα παρατηρηθεί αν τα πακέτα κινούνται σωστά κατανεμημένα μεταξύ των μονοπατιών. Για να το δούμε αυτό θα θέσουμε τον h1 σαν server με την εντολή
“iperf -s” και συγχρόνως θα θέσουμε τον h3 σαν client με την εντολή “iperf -c 10.0.0.1” και θα έχουμε τα αποτελέσματα στην Εικόνα 22:



Εικόνα 22. iperf h1 - h3

Τώρα θα ελεγχθεί ξανά η κίνηση από τα ports του μεταγωγέα 3 με την εντολή:

sudo ovs-ofctl -O OpenFlow13 dump-groups s3



Εικόνα 23. Επίτευξη εξισορρόπησης φόρτου

Όπως παρατηρούμε στην Εικόνα 23, χρησιμοποιούνται τα υπάρχοντα ports κανονικά και υπάρχει εξισορρόπηση του φορτίου διότι τα πακέτα είναι μοιρασμένα στα ports ως εξής:
1107551 pkts στο “s3-eth1” και 1072842 pkts στο “s3-eth2” → 2176101 pkts στο “2176101”

Αξίζει να σημειωθεί ότι το OVS, ειδικότερα στις παλαιότερες εκδόσεις, μπορεί να αντιμετωπίσει πρόβλημα και να μην χωρίσει την κίνηση κατάλληλα [50]. Για την επίλυση αυτού του προβλήματος, αρκεί, όταν εκτελούμε τον client, να δηλώσουμε το πλήθος των παράλληλων ports που υπάρχουν (στην περίπτωση μας θα ήταν 4). Συνεπώς, σε αυτή την περίπτωση, η προηγούμενη εντολή θα άλλαζε σε:

iperf -c 10.0.0.1 -P 4

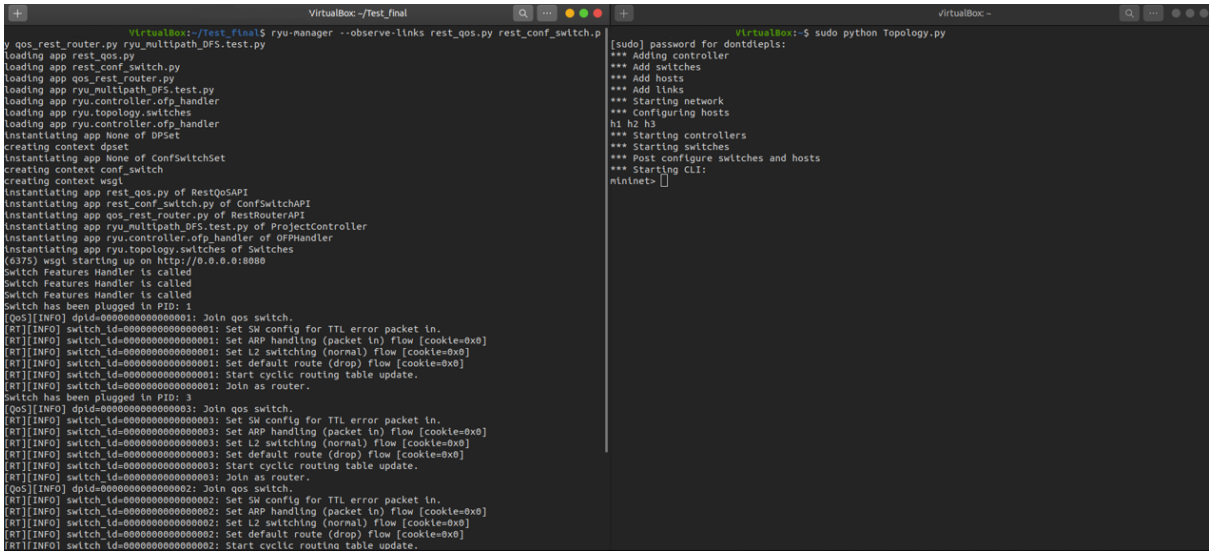
Με αυτόν τον τρόπο επετεύχθη εξισορρόπηση του φόρτου.

Στην συνέχεια, θα βελτιώσουμε και άλλο το δίκτυο μας χρησιμοποιώντας τις υπηρεσίες QoS, που προσφέρει ο Ryu Ελεγκτής με την βοήθεια του Ryubook, το οποίο είναι

ειδικά διαμορφωμένο βιβλίο για την ανάπτυξη του ελεγκτή μας στο περιβάλλον μας, σχετικά με την χρήση των τιμών του DSCP, όπως περιγράψαμε στην ενότητα 2.8 [49]. Οι εφαρμογές που θα χρησιμοποιήσουμε είναι:

- RyuApp: rest_qos.
- RyuApp: rest_conf_switch
- RyuApp: qos_rest_router

Ξεκινάμε την εκτέλεση των εφαρμογών και της τοπολογίας όπως φαίνεται παρακάτω:



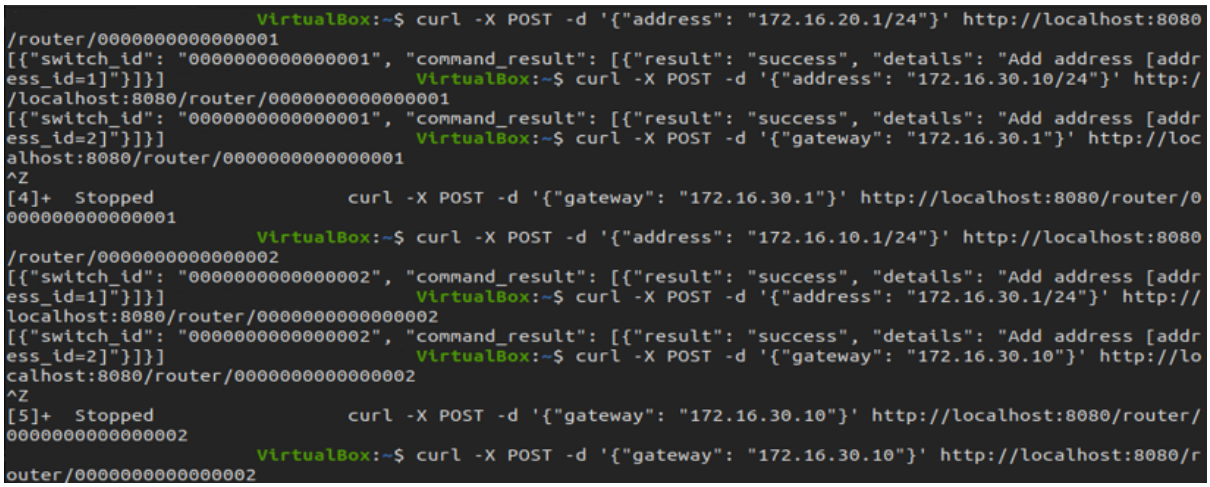
Εικόνα 24. Σύνδεση Μεταγωγέων για QoS

Εφόσον επιβεβαιώνουμε ότι συνδέονται κανονικά στην Εικόνα 24, αρχικά κάνουμε set ovsdb_addr για να έχουμε πρόσβαση στο OVSDB, το οποίο είναι η βάση δεδομένων του OVS για την διαχείριση του δικτύου, και στην συνέχεια ορίζουμε και την ουρά. Αυτό γίνεται με τις εντολές:

curl -X PUT -d '"tcp:127.0.0.1:6632"'
http://localhost:8080/v1.0/conf/switches/0000000000000001/ovsdb_addr και

curl -X POST -d '{"port_name": "s1-eth1", "type": "linux-htb", "max_rate": "1000000", "queues":[{"max_rate": "1000000"}, {"min_rate": "200000"}, {"min_rate": "500000"}]}'
http://localhost:8080/qos/queue/0000000000000001

Έπειτα, ορίζουμε τις διευθύνσεις IP και την προκαθορισμένη διαδρομή για τον κάθε δρομολογητή όπως φαίνεται παρακάτω:



Εικόνα 25. Ορισμός IP και της προκαθορισμένης διαδρομής

Στην συνέχεια, ορίζουμε την ροή, όπως φαίνεται στην Εικόνα 25, σύμφωνα με τις τιμές DSCP. Πιο συγκεκριμένα, θέτουμε την DSCP τιμή 26 (AF31) για την ουρά 1 και την τιμή 34 (AF41) για την ουρά 2 με τις εντολές:

curl -X POST -d '{"match": {"ip_dscp": "26"}, "actions":{"queue": "1"}}' http://localhost:8080/qos/rules/0000000000000001

curl -X POST -d '{"match": {"ip_dscp": "34"}, "actions":{"queue": "2"}}' http://localhost:8080/qos/rules/0000000000000001

Έπειτα, ορίζουμε τους κανόνες για τον δεύτερο δρομολογητή όπως φαίνεται στον Πίνακα 9:

Πίνακας 9. Ορισμός Κανόνων

Προτεραιότητα	Τιμή DSCP	QoS ID	Διεύθυνση Προορισμού	Θύρα Προορισμού
1	26	1	172.16.20.10	5002
1	34	2	172.16.20.10	5002

Ο κώδικας που υλοποιεί τους παραπάνω κανόνες είναι:

curl -X POST -d '{"match": {"nw_dst": "172.16.20.10", "nw_proto": "UDP", "tp_dst": "5002"}, "actions":{"mark": "26"}}' http://localhost:8080/qos/rules/0000000000000002
και

curl -X POST -d '{"match": {"nw_dst": "172.16.20.10", "nw_proto": "UDP", "tp_dst": "5003"}, "actions":{"mark": "34"}}' http://localhost:8080/qos/rules/0000000000000002

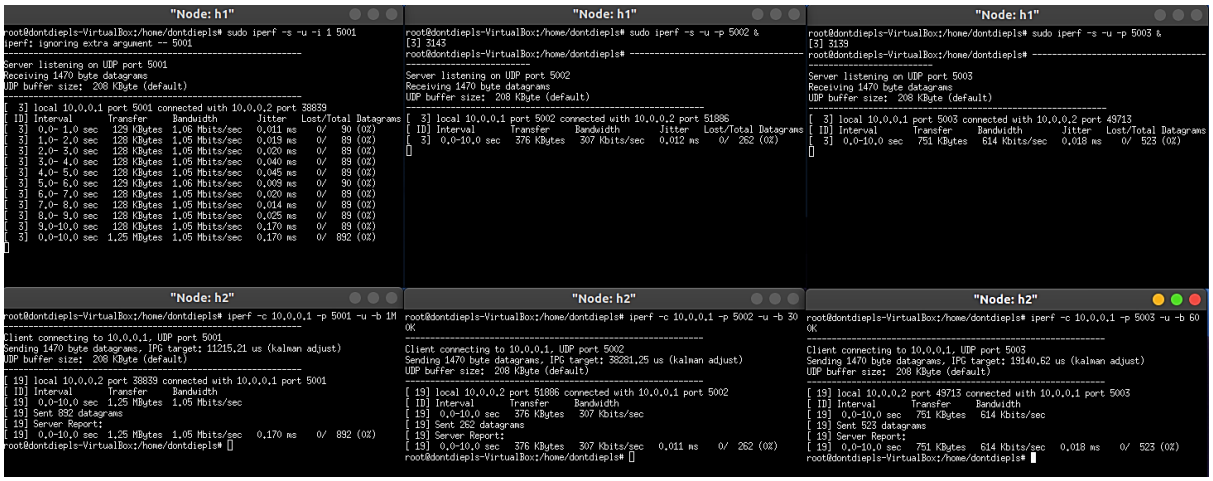
Έπειτα, για τον έλεγχο του QoS, χρησιμοποιούμε το iperf στέλνοντας 1Mbps κίνηση UDP στο port 5001 του host 1, 300Kbps κίνηση UDP στο port 5002 του host 1 και 600Kbps κίνηση UDP στο port 5003 του host 1 με τις εντολές :

iperf -s -u -p 5002 &
iperf -s -u -p 5003 &
iperf -s -u -i 1 5001

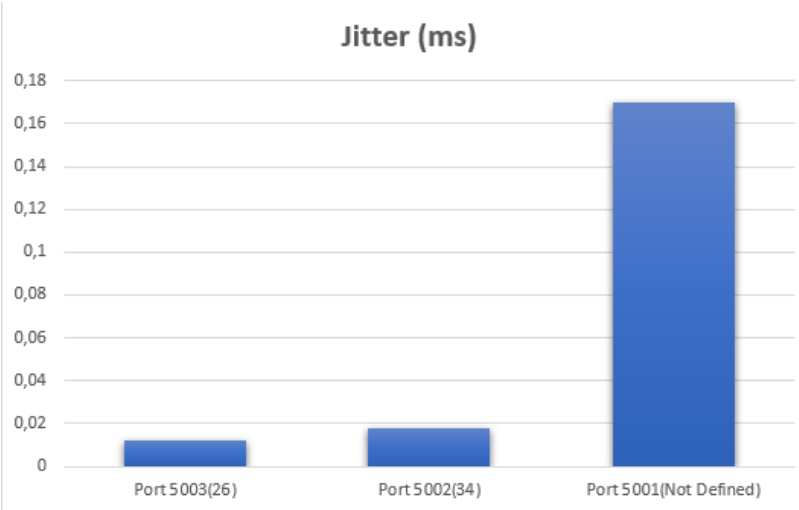
αντίστοιχα και από τον h2:

iperf -c 172.16.20.10 -p 5001 -u -b 1M
iperf -c 172.16.20.10 -p 5002 -u -b 300K
iperf -c 172.16.20.10 -p 5003 -u -b 600K

Τα αποτελέσματα που λαμβάνουμε είναι :



Εικόνα 26. Αποτελέσματα χρήσης της εντολής Iperf



Εικόνα 27. Γραφική Αναπαράσταση του Jitter

Όπως είναι φανερό, από την Εικόνα 26 που περιλαμβάνει την εκτέλεση και την Εικόνα 27 που περιλαμβάνει την γραφική αναπαράσταση των αποτελεσμάτων, στις θύρες 5003 & 5002 στις οποίες έχουν οριστεί οι DSCP τιμές 26 και 34 αντίστοιχα, έχουν σταθερό εύρος ζώνης χωρίς αυξομειώσεις στα 600K και 300K αντίστοιχα με αρκετά χαμηλό jitter ενώ στο port 5001 στο οποίο δεν έχει οριστεί καμία DSCP τιμή υπάρχει αρκετά μεγάλη αύξηση του jitter.
Συγκεκριμένα, από 0.018 ms και 0.012 ms στα ports 5003 και 5002, στο port 5001 έχουμε μέχρι και 0.170 ms. Δεδομένης της φύσης της κίνησης που είχαμε στην συγκεκριμένη περίπτωση δεν είχαμε αυξομειώσεις στο εύρος ζώνης, ούτε απώλειες πακέτων.

Συνοψίζοντας, στην παραπάνω ενότητα δόθηκε πρακτικά η λύση στο πρόβλημα της διαχείρισης της κρίσιμης κίνησης σε δίκτυα SDN. Στην αρχή παρουσιάσαμε και αναλύσαμε την υλοποίηση του κώδικα που επιλύει το πρόβλημα της εξισορρόπησης του φόρτου του δικτύου μας και στην συνέχεια παρουσιάσαμε και αναλύσαμε την υλοποίηση του QoS με την χρήση του DSCP.

4. Συμπεράσματα

4.1 Αξιολόγηση των αποτελεσμάτων

Η βέλτιστη δρομολόγηση κίνησης σε κρίσιμες υποδομές είναι μια πρόκληση η οποία υπήρχε από τα παραδοσιακού τύπου δίκτυα και συνεχίζει να υπάρχει ακόμα και στα σύγχρονα. Οι περισσότερες προσπάθειες σχετικά με την βελτιστοποίηση του φόρτου του δικτύου και την παροχή καλύτερης ποιότητας υπηρεσιών αφορούσαν πιο συμβατικές αρχιτεκτονικές δικτύων. Ωστόσο, όπως έγινε αναφορά και στην ενότητα 1.1, κατά την διάρκεια της έρευνας για την παρούσα διπλωματική εργασία βρέθηκαν πειραματικές υλοποιήσεις και άρθρα που παρείχαν παρεμφερείς λύσεις. Μετά το πέρας της παρούσας διπλωματικής εργασίας, μπορούν να εξαχθούν συμπεράσματα όσον αναφορά τις ελλείψεις και τα μειονεκτήματα τους. Πολλά μειονεκτήματα όπως η μη αυτόματη ανανέωση σε τυχόν αλλαγές του δικτύου και οι προκαθορισμένες τιμές βαρών έχουν αναφερθεί ήδη στην ενότητα 1.1.

Η σημαντική επέκταση που έγινε σε αυτά τα προβλήματα είναι στο κομμάτι της σύγκρισης των αλγορίθμων εύρεσης μονοπατιών στην ενότητα 3.4. Πιο συγκεκριμένα, έγινε σύγκριση μεταξύ τριών αλγορίθμων: DFS, BFS και Dijkstra. Η πρώτη σύγκριση έγινε μεταξύ των αλγορίθμων DFS και BFS εκ των οποίων αποδείχθηκε καλύτερος ο DFS. Στην συνέχεια, ο DFS συγκρίθηκε με τον Dijkstra και εξάχθηκε το συμπέρασμα ότι ο DFS ξανά ήταν βέλτιστος. Τα κριτήρια σύγκρισης των αλγορίθμων ήταν ο χρόνος μετάδοσης πακέτου από ένα σημείο του δικτύου σε ένα άλλο, η ρυθμοαπόδοση, ο χρόνος εκτέλεσης και το ποσοστό απώλειας πακέτων.

Τα αποτελέσματα της πρακτικής εφαρμογής έρχονται να υποστηρίξουν τις επιλογές που έγιναν, όσον αφορά τους αλγορίθμους και την υλοποίηση, στην συγκεκριμένη εργασία με τους εξής τρόπους:

Ο αλγόριθμος DFS έκανε εύρεση όλων των διαθέσιμων μονοπατιών με τον γρηγορότερο δυνατό τρόπο. Πιο συγκεκριμένα, βρήκε τα μονοπάτια με RTT : 0,01607 (h1-h3) και 0,00738 (h1-h2), όπως φαίνεται στην εικόνα 20.

Σε συνδυασμό με τον OSPF, υπολογίστηκαν και στην συνέχεια ορίστηκαν τα βάρη με στόχο την εύρεση των βέλτιστων διαδρομών του δικτύου, όπως φαίνεται στην εικόνα 20. Είναι αξιοσημείωτο ότι οποιαδήποτε μεταβολή υπάρξει στο δίκτυο γίνεται άμεσα αντιληπτή, καθώς ο ρυθμός ανανέωσης είναι 1 sec, και έτσι υπάρχει δυναμικός χαρακτήρας στην τοπολογία του δικτύου.

Η ταχύτητα του DFS σε συνδυασμό με τον δυναμικό υπολογισμό της τοπολογίας οδηγούν στην εξισορρόπησης φορτίου με αποδοτικό τρόπο καθώς μέσα από

πολλαπλές διαδρομές διαμοιράζεται η υπάρχουσα κίνηση χωρίς απώλειες, όπως φαίνεται στην εικόνα 23.

Τέλος, η αποτελεσματική εξασφάλιση της ποιότητας των υπηρεσιών για την κρίσιμη κίνηση, που είναι το τελικό ζητούμενο της συγκεκριμένης εργασίας, επιτυγχάνεται με την χρήση του DSCP, το οποίο επιβεβαιώνεται και από τις τιμές του jitter, όπως φαίνεται στην Εικόνα 26.

Η λύση η οποία προτάθηκε στην συγκεκριμένη διπλωματική εργασία περιλαμβάνει την ιδανικότερη λύση. Χρησιμοποιώντας το θεωρητικό υπόβαθρο για την κατανόηση και διαμόρφωση άποψης πάνω στο θέμα καταλήξαμε στον συνδυασμό του DFS με τον OSPF, υλοποιημένοι πάνω στον ελεγκτή Ryu. Επίσης, αξίζει να σημειωθεί ότι με την σημαντική προσθήκη του QoS προσφέρεται μια λύση η οποία κάνει τόσο τον υπολογισμό της βέλτιστης διαδρομής σε ένα δίκτυο όσο και την κατανομή και ιεράρχηση της κίνησης αυτού με τη χρήση των ελάχιστων δυνατών πόρων και της αποδοτικότερης μεθόδου με βάση τα κριτήρια της αξιοπιστίας και της ταχύτητας και της ακρίβειας.

4.2 Σύνοψη

Τα τελευταία χρόνια είναι αισθητή η ραγδαία αύξηση των τεχνολογικών επιτευγμάτων όπως και των απαιτήσεων τους. Αυτό αναγκάζει με αντίστοιχο ρυθμό να αναβαθμίζονται συνεχώς και τα δίκτυα που χρησιμοποιούμε για να ξεπεράσουμε προβλήματα και περιορισμούς που υπάρχουν στα παραδοσιακά δίκτυα. Μια αναβάθμιση της αρχιτεκτονικής των δικτύων είναι και το SDN το οποίο μας προσφέρει τη δυνατότητα για επίλυση σημαντικών προβλημάτων και αρκετά οφέλη με ένα από τα σημαντικότερα να είναι η προσαρμογή με την χρήση κατάλληλων εφαρμογών στις εκάστοτε συνθήκες. Κατά την διάρκεια της παραπάνω διπλωματικής εργασίας, αναπτύχθηκε μια μέθοδος εξισορρόπησης του φόρτου που υπάρχει σε ένα δίκτυο αλλά και της ανάπτυξης μεθόδων που αφορούν την ποιότητα υπηρεσιών για την διασφάλιση της καλύτερης ροής κρίσιμης κίνησης. Πιο συγκεκριμένα, συγκρίθηκε και βρέθηκε ο βέλτιστος τρόπος για την εύρεση των διαθέσιμων μονοπατιών, μεταξύ του DFS, BFS και Dijkstra, από μεριάς χρόνου αλλά και τα βέλτιστα μονοπάτια σύμφωνα με τον κατάλληλο υπολογισμό βαρών. Επίσης η προτεινόμενη υλοποίηση παρέχει στα δίκτυα μια δυναμική μορφή καθώς αναγνωρίζει τυχόν αλλαγές που πραγματοποιούνται και συγχρόνως εντοπίζει ή διαγράφει τα νέα μονοπάτια που δημιουργούνται. Στην συνέχεια, μπορεί να πραγματοποιήσει διαμοιρασμό της κίνησης για την μείωση φόρτου όπου είναι δυνατό αλλά και να δώσει προτεραιότητα στην κρισιμότερη κίνηση. Επιπλέον, προσφέρει QoS, με την χρήση του DSCP ως κριτήριο ιεράρχησης. Ωστόσο, θα ήταν σημαντική παράλειψη να μην γίνει αναφορά στα προβλήματα που μπορεί να αντιμετωπίσει το δίκτυο που αποτελούνται από ένα ευρύ φάσμα που ξεκινάει από τις εγκαταστάσεις στις οποίες βρίσκεται μέχρι και σε τυχόν επιθέσεις που μπορεί να δεχθεί. Εν κατακλείδι, το SDN συνεχίζει να είναι μια άριστη

λύση, στην οποία όμως υπάρχουν αρκετές προκλήσεις οι οποίες επιλύνονται και θα συνεχίσουν να επιλύνονται για τα επόμενα χρόνια.

4.3 Μελλοντική Εργασία

Η συγκεκριμένη διπλωματική εργασία μπορεί να χρησιμοποιηθεί μελλοντικά ως ένα εναρκτήριο σημείο για περεταίρω έρευνα με στόχο την ανάπτυξη ενός καλύτερου συστήματος για εξισορρόπηση φόρτου δικτύου αλλά και την παροχή καλύτερης ποιότητας υπηρεσιών. Συγκεκριμένα, συνοψίζονται πιθανές μελλοντικές επεκτάσεις στις παρακάτω:

Η ανάπτυξη ενός συστήματος για την μεταφορά και ταξινόμηση της κίνησης του δικτύου βασισμένο σε μηχανική μάθηση όπως ακόμα και την αναγνώριση της κρίσιμης κίνησης του δικτύου δυναμικά με μεγαλύτερη αποτελεσματικότητα. Η μηχανική μάθηση είναι μια λύση η οποία μπορεί να ξεπεράσει τα ζητήματα δυναμικότητας και προσαρμοστικότητας του δικτύου σε διαφορετικά είδη κίνησης και μορφές του δικτύου καθώς η διαχείριση του δικτύου ο τρόπος διαχείρισης του δικτύου εξαρτάται σε μεγάλο βαθμό από την εκπαίδευση του.

Την ένταξη λειτουργιών προστασίας του δικτύου με την μορφή λογισμικού (NFV) από πάσης φύσεως επιθέσεις όπως είναι οι επιθέσεις παρακολούθησης δικτύου (Network Monitoring Attacks) στις οποίες ο εισβολέας αποκτάει πρόσβαση στον ελεγκτή και μπορεί και τροποποιεί τα σήματα ελέγχου. Άλλη μορφή επιθέσεων από τις οποίες χρειάζεται προστασία ο ελεγκτής είναι οι επιθέσεις κατανεμημένης άρνησης παροχής υπηρεσιών (Distributed Denial of Service - DDoS) στις οποίες ο εισβολέας στέλνει πολλά αιτήματα με σκοπό την υπερφόρτωση του ελεγκτή και την άρνηση υπηρεσιών. Τέλος, επιθέσεις της μορφής IP spoofing στις οποίες ο εισβολέας δημιουργεί ψευδείς διευθύνσεις IP με σκοπό να λάβει την “εμπιστοσύνη” του δικτύου και να εκτελέσει οποιαδήποτε λειτουργία επιθυμεί. Το κομμάτι της ασφάλειας σε όλα τα υπολογιστικά συστήματα είναι έτσι και αλλιώς ένας ταχέα αναπτυσσόμενος κλάδος.

Η δημιουργία ενός γραφικού περιβάλλοντος για τον χρήστη το οποίο θα παρέχει μέσω εικόνων, ενδείξεων και εργαλείων μια πιο κατανοητή εικόνα και παραμετροποίηση του δικτύου. Με αυτόν τον τρόπο, θα επιτευχθεί η αλληλεπίδραση εφαρμογής-χρήστη και με απλές ενέργειες, για τον χρήστη, θα μπορεί να εκτελέσει τις ίδιες ή και πιο περίπλοκες εργασίες χωρίς την ανάγκη της γραμμής εντολών. Το προσδοκώμενο αποτέλεσμα από την παραπάνω αλλαγή θα είναι τελικά, η δημιουργία ενός όμορφου εύχρηστου και λειτουργικού περιβάλλοντος για τον οποιοδήποτε χρήστη.

Τέλος, με την εξέλιξη των τεχνολογιών που θα επέλθει, θα μπορούσε να δοθεί μια διαφορετική οπτική γωνία στην αρχιτεκτονική του συστήματος προκειμένου να αυξηθεί η ταχύτητα σε συνδυασμό με την μείωση των απαιτούμενων πόρων τόσο από τους

κόμβους όσο και από την ενέργεια που απαιτείται για την πλήρη λειτουργία του δικτύου.

Βιβλιογραφικές Αναφορές

[1] Punchkov Maxim, SDN-Routing, (2020), GitHub Repository, https://github.com/maxim-puchkov/SDN-Routing

[2] N. McKeown, T.Anderson, H.Balakrishnan, G.Parulkar, L.Peterson, J.Rexford, S.Shenker, J.Turner, “OpenFlow: enabling innovation in campus networks,” SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pp. 69–74, Mar. 2008.

[3] Aida K., Dijkstra-SDN-Ryu, (2020), GitHub Repository, https://github.com/aidakrr/dijkstra-SDN-Ryu

[4] L. Mousheng, T. Yong, Z. Qiang, and W. Wenyong, “Controllable Network Architecture Based on SDN,” 2015 2nd International Conference on Information Science and Control Engineering, pp. 174–180, 2015.

[5] E.-B. Fgee, J. D. Kenney, W. J. Phillips, W. Robertson, and S. Sivakumar, “Comparison of QoS Performance between IPv6 QoS Management Model and IntServ and DiffServ QoS Models,” 3rd Annual Communication Networks and Services Research Conference (CNSR’05), pp. 287–292, 2005.

[6] M. Alsaeedi, M. M. Mohamad, and A. A. Al-Roubaiey, “Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey,” in IEEE Access, vol. 7, pp. 107346–107379, 2019.

[7] K. Cabaj, J. Wytrębowicz, S. Kukliński, P. Radziszewski and K. Truong Dinh, “SDN Architecture Impact on Network Security,” in FedCSIS, vol 3, pp. 143–148, 2014.

[8] J. Bhatia, R. Govani, and M. Bhavsar, “Software Defined Networking: From Theory to Practice,” 2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC), pp. 789–794, Dec. 2018.

[9] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, and G. Parulkar, "ONOS: towards an open, distributed SDN OS," in 3rd Workshop on Hot Topics in Software Defined Networking (HotSDN '14). Association for Computing Machinery, pp. 1–6, 2014.

[10] Sandhya, Y. Sinha, and K. Haribabu, “A survey: Hybrid SDN,” Journal of Network and Computer Applications, vol. 100, pp. 35–55, Dec. 2017.

[11] D. S. Rana, S. A. Dhondiyal, and S. K. Chamoli, “Software Defined Networking (SDN) Challenges, issues, and Solution,” IJCSE, vol. 7, no. 1, pp. 884–889, Jan. 2019.

[12] [RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, January 2015, <https://www.rfc-editor.org/info/rfc7426>.

[13] [RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, January 2015, <https://www.rfc-editor.org/info/rfc7426>.

[14] T. D. Nadeau and K. Gray, “SDN: Software Defined Networks,” in Beijing: O'Reilly, 2013, [Online]. Available: https://www.oreilly.com/library/view/sdn-software-defined/9781449342425/ch04.html [Accessed: 08-Jan-2022].

[15] S. A. Shah, J. Faiz, M. Farooq, A. Shafi, and S. A. Mehdi, “An architectural evaluation of SDN controllers,” 2013 IEEE International Conference on Communications (ICC), pp. 3504–3508, Jun. 2013.

[16] L. Mamushiane, A. Lysko, and S. Dlamini, “A comparative evaluation of the performance of popular SDN controllers,” 2018 Wireless Days (WD), pp. 54–59, Apr. 2018.

[17] R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou, “Feature-based comparison and selection of Software Defined Networking (SDN) controllers,” 2014 World Congress on Computer Applications and Information Systems (WCCAIS), pp. 1–7, Jan. 2014.

[18] O. Salman, I. H. Elhajj, A. Kayssi, and A. Chehab, “SDN controllers: A comparative study,” 2016 18th Mediterranean Electrotechnical Conference (MELECON), pp. 1–6, Apr. 2016.

[19] V. Thirupathi, Ch. Sandeep, S. Naresh Kumar, P. Pramod Kumar “A comprehensive review on SDN architecture, applications and major benefits of SDN”, IJAST, vol. 28, no. 20, pp. 607 - 614, Dec. 2019.

[20] Open Networking Foundation, “Openflow Switch Specification”, Version 1.3.0, June 25, 2012, [Online]. Available: https://opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf [Accessed: 15-May-2022].

[21] P. Vizarreta, K. Trivedi, B. Helvik, P. Heegaard, W. Kellerer, and C. M. Machuca, “An empirical study of software reliability in SDN controllers,” 2017 13th International Conference on Network and Service Management (CNSM), pp. 1–9, Nov. 2017.

[22] D. S. Rana, S. A. Dhondiyal, and S. K. Chamoli, “Software Defined Networking (SDN) Challenges, issues, and Solution,” IJCSE, vol. 7, no. 1, pp. 884–889, Jan. 2019.

[23] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, “NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC),” in IEEE Network, vol. 28, no. 6, pp. 18–26, Nov. 2014.

[24] J. Gil Herrera and J. F. Botero, “Resource Allocation in NFV: A Comprehensive Survey,” in IEEE Transactions on Network and Service Management, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[25] D. Pliatsios, P. Sarigiannidis, S. Goudos, and G. K. Karagiannidis, “Realizing 5G vision through Cloud RAN: technologies, challenges, and trends,” EURASIP J Wireless Com Network, vol. 2018, no. 1, pp. 136, Dec. 2018.

[26] J. Harju and P. Kivimaki, “Co-operation and comparison of DiffServ and IntServ: performance measurements,” Proceedings 25th Annual IEEE Conference on Local Computer Networks. LCN 2000, pp. 177–186, 2000.

[27] A. O. Adedayo and B. Twala, “QoS functionality in software defined network,” 2017 International Conference on Information and Communication Technology Convergence (ICTC), pp. 693–699, Oct. 2017.

[28] D. Matthews, “QoS – Classification and Marking,” 19 April 2010, [Online]. Available: http://darenmatthews.com/blog/?p=758 [Accessed: 28-June-2022].

[29] R. Molenaar, “IP precedence and DSCP values,” NetworkLessons.com, 17 Feb. 2021, [Online]. Available: https://networklessons.com/quality-of-service/ip precedence-dscp-values [Accessed: 02-Mar-2022].

[30] Cisco, Cisco Nexus 1000V Quality of Service Configuration Guide. Cisco, release 4.2(1) sv1(4) edition,17 March 2016, [Online].Available: https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus1000/sw/4_2_1_s_v_1_4/qos/configuration/guide/n1000v_qos.html [Accessed: 09-Jan-2022].

[31] [RFC4594] Babiarz, J., Chan, K., and F. Baker, "Configuration Guidelines for DiffServ Service Classes", RFC 4594, August 2006, [Online]. <https://www.rfc-editor.org/info/rfc4594>.

[32] Cisco, “Class-Based Weighted Fair Queueing”, 2012, [Online]. Available: https://www.cisco.com/en/US/docs/ios/12_0t/12_0t5/feature/guide/cbwfq.html [Accessed: 17-Apr-2022].

[33] Network Lessons, “QoS LLQ (Low Latency Queueing) on Cisco IOS”, [Online]. Available: https://networklessons.com/quality-of-service/qos-llq-low-latency-queueing-cisco-ios [Accessed: 08-Jan-2022].

[34] K. Wallace, “Cisco IP Telephony Flash Cards: Weighted Random Early Detection (WRED)”, Nov 24, 2004. [Online]. Available: https://www.ciscopress.com/articles/article.asp?p=352991&seqNum=8 [Accessed: 08-Jan-2022].

[35] "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture," in IEEE Std 802-2001 (Revision of IEEE Std 802-1990), vol., no., pp.1-48, 7 Feb. 2002.

[36] [RFC0826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, November 1982, <https://www.rfc-editor.org/info/rfc826>.

[37] S. Kumar, S. Dalal, and V. Dixit, “The OSI Model: Overview on the seven layers of computer networks,” International Journal of Computer Science and Information Technology Research, vol. 2, no. 3, pp. 461-466, 2014.

[38] [RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", RFC 3260, April 2002, <https://www.rfc-editor.org/info/rfc3260>.

[39] L. Zhu, M. M. Karim, K. Sharif, F. Li, X. Du, and M. Guizani, “SDN Controllers: Benchmarking & Performance Evaluation,” arXiv:1902.04491 [cs.NI], Feb. 2019.

[40] V.R.S. Raju, “SDN Controllers Comparison,” In Proceedings of the Science Globe International Conference, vol. 6, no. 7, pp. 34-38, June 2018.

[41] D. Pliatsios, P. Sarigiannidis, T. Liatifis, K. Rompolos, and I. Siniosoglou, “A Novel and Interactive Industrial Control System Honeypot for Critical Smart Grid Infrastructure,” in 2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp. 1–6, Sep. 2019.

[42] A. M. J. Sadik, M. A. Dhali, H. M. A. B. Farid, T. U. Rashid, and A. Syeed, “A Comprehensive and Comparative Study of Maze-Solving Techniques by Implementing Graph Theory,” 2010 International Conference on Artificial Intelligence and Computational Intelligence, pp. 52–56, Oct. 2010.

[43] T. Everitt and M. Hutter, “Analytical Results on the BFS vs. DFS Algorithm Selection Problem: Part II: Graph Search,” in AI 2015: Advances in Artificial Intelligence, vol. 9457, pp. 166–178, 2015.

[44] M. S. Hossen, M. H. Rahman, M. Al-Mustanjid, M. A. Shakil Nobin, and M. A. Habib, “Enhancing Quality of Service in SDN based on Multi-path Routing Optimization with DFS,” 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), pp. 1–5, Dec. 2019.

[45] S. Syaifuddin, M. F. Azis, and F. D. S. Sumadi, “Comparison Analysis of Multipath Routing Implementation in Software Defined Network,” KINETIK, vol. 6, no. 2, pp. 141-148, May 2021.

[46] Cisco, “OSPF Design Guide”, August 2005, [Online]. Available: https://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/7039-1.html [Accessed: 24-Apr-2022].

[47] [RFC7042] Eastlake 3rd, D., and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, October 2013, <https://www.rfc-editor.org/info/rfc7042>.

[48] S. Gummadi, S. Tatineni, and V. G. Kukkapalli, “Probabilistic Path Queries in Road Networks: Traffic Uncertainty Aware Path Selection,” CSCE 6350, Spring 2012.

[49] sphinx-quickstart, “Ryubook 1.0 documentation”,16 October 2013, [Online]. Available: https://osrg.github.io/ryu-book/en/html/ [Accessed: 8-Mar-2022].

[50] Open vSwitch, “OVS FAQ”, 12 March.2022, [Online]. Available: https://docs.openvswitch.org/en/latest/faq/ [Accessed: 12-Mar-2022].