ΠΑΝΕΠΙΣΤΗΜΙΟ
ΙΩΑΝΝΙΝΩΝ

ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΜΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΔΙΚΤΥΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΚΑΛΥΨΗ ΤΟΠΟΛΟΓΙΑΣ ΣΕ ΔΙΚΤΥΑ ΚΑΘΟΡΙΖΟΜΕΝΑ ΑΠΟ ΛΟΓΙΣΜΙΚΟ

Αριστοτέλης Ζαχάρης

Επιβλέπων: Ελευθέριος Στεργίου,
ΔΕΠ, Αναπληρωτής Καθηγητής

Άρτα, Απρίλιος, 2021

DISCOVERY TOPOLOGY IN SOFTWARE DEFINED NETWORKING

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Άρτα, Ημερομηνία

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Ελευθέριος Στεργίου,

   ΔΕΠ, Αναπληρωτής Καθηγητής

2. Μέλος επιτροπής

   Κωνσταντίνος Αγγέλης,

   ΔΕΠ, Καθηγητής Α

3. Μέλος επιτροπής

   Σπυριδούλα Μαργαρίτη,

   ΕΔΙΠ, Α

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα μεταπτυχιακή εργασία είναι εκ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Ζαχάρης, Αριστοτέλης

Υπογραφή

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εκπονήθηκε στο τμήμα Πληροφορικής και Τηλεπικοινωνιών του Πανεπιστημίου Ιωαννίνων για το μεταπτυχιακό πρόγραμμα Πληροφορικής και Δικτύων. Για την εκπόνησή αυτής θα ήθελα να ευχαριστήσω θερμά τον επιβλέπων καθηγητή κ. Ελευθέριο Στεργίου καθώς επίσης την καθηγήτρια κ. Σπυριδούλα Μαργαρίτη και τον καθηγητή κ. Κωνσταντίνο Αγγέλη για την πολύτιμη βοήθεια τους κατά την διάρκεια εκπόνησης της διπλωματικής μου εργασίας. Τέλος, ευχαριστώ την οικογένειά μου και τους φίλους μου για την στήριξή τους σε όλα τα έτη των σπουδών μου.

ΠΕΡΙΛΗΨΗ

Η δικτύωση καθοριζόμενη από το λογισμικό (Software Defined Networking -SDN) είναι ένα νέο πρότυπο δικτύωσης, με πολλές δυνατότητες καθώς συμβάλλει ώστε να αυξηθεί η αποδοτικότητα, να μειωθεί η πολυπλοκότητα του ελέγχου και της διαχείρισης και να επιταχυνθεί ο ρυθμός της τεχνολογικής ανάπτυξης των δικτύων. Μία από τις καινοτομίες του SDN είναι ο διαχωρισμός του δικτύου σε τρία λειτουργικά επίπεδα: επίπεδο δεδομένων (data plane), επιπέδου ελέγχου (control plane) και επίπεδο διαχείρισης (management plane). Η ευφυΐα και ο έλεγχος λειτουργίας και διαχείρισης του δικτύου, όπως η δρομολόγηση, αφαιρείται από τις δικτυακές συσκευές (δρομολογητές, μεταγωγείς) και συγκεντρώνεται σε ένα λογικά κεντρικοποιημένο στοιχείο, δηλαδή τον ελεγκτή SDN. Ο ελεγκτής, έχοντας μια συνοπτική εικόνα της τοπολογίας, μπορεί να διαμορφώσει και να διαχειριστεί το δίκτυο, όμως θα πρέπει να έχει ενημερωμένες πληροφορίες σχετικά με την κατάσταση του δικτύου και ιδίως να γνωρίζει την τοπολογία του. Κατά συνέπεια, η ανακάλυψη τοπολογίας είναι ένα κρίσιμο στοιχείο κάθε αρχιτεκτονικής δικτύου που καθορίζεται από λογισμικό.

Στη συγκεκριμένη εργασία θα μελετήσουμε τη δικτύωση SDN, και θα εξετάσουμε και θα συγκρίνουμε τα πρωτόκολλα ανακάλυψης τοπολογίας που χρησιμοποιούνται από αυτή την αρχιτεκτονική. Με τη βοήθεια του Mininet θα εφαρμόσουμε το OpenFlow Discovery Protocol (OFDP) πρωτόκολλο σε διάφορες τοπολογίες. Το OFDP είναι ένα defacto πρωτόκολλο που χρησιμοποιείται από τους ελεγκτές για να ανακαλύψει την υποκείμενη τοπολογία. Ωστόσο, η ανακάλυψη μιας τοπολογίας δικτύου είναι δύσκολη λόγω 1) της συχνής μετεγκατάστασης των εικονικών μηχανών στα κέντρα δεδομένων, 2) έλλειψη μηχανισμών ελέγχου ταυτότητας, 3) έλλειψη προτύπων SDN και 4) ενσωμάτωση μηχανισμών ασφαλείας για την ανακάλυψη τοπολογίας. Σκοπός της παρούσας εργασίας είναι: α) να διερευνήσει τις τεχνικές που εφαρμόζονται για την ανακάλυψη της τοπολογίας καθώς πρόκειται για μια σημαντική διαδικασία που παρέχεται από τον ελεγκτή και σχετίζεται με την ορθή λειτουργία και την απόδοση του δικτύου, β) να εξετάσει αν υπάρχουν παράμετροι που εμπλέκονται στη διαδικασία εντοπισμού της τοπολογίας, ποιες είναι (π.χ. overhead), πως επηρεάζουν την διαδικασία εντοπισμού και πως μπορούν να βελτιώσουν τη λειτουργικότητα του δικτύου.

Λέξεις κλειδιά: Δίκτυα καθοριζόμενα από λογισμικό-SDN, ανακάλυψη τοπολογίας, Mininet, OpenFlow, OFDP, LLDP

ABSTRACT

Software Defined Networking (SDN) is a new, multi-feature networking standard that can help increase efficiency, reduce the complexity of network control and management, and accelerate the pace of technological innovation. One of the innovations of SDN is the separation of the network into functional levels: data plane, control plane and management plane., such as routing, is removed from network devices (routers, switches) and assembled into a logically centralized component, namely the SDN controller. information about the state of the network, in particular its topology, therefore, topology discovery is a critical element of any software-defined network architecture.

In this paper we will study SDN networking and examine and compare the topology discovery protocols used by this architecture. With the help of Mininet we will apply the OpenFlow Discovery Protocol (OFDP) protocol in various topologies. OFDP is the de facto protocol used by OpenFlow controllers to discover the underlying topology. However, discovering a network topology is difficult due to 1) the frequent relocation of virtual machines to data centers, 2) the lack of authentication mechanisms, 3) the lack of SDN standards, and 4) the integration of security mechanisms for topology discovery.

The purpose of this work is: a) to investigate the techniques used to reveal the topology as it is an important process provided by the controller and related to the proper operation and performance of the network, b) to examine whether there are parameters that are involved in the process of locating the topology, what they are (eg overhead), how they affect the locating process and how they can improve the functionality of the network.

Keywords: Software defined networks- SDN, topology discovery, Mininet, OpenFlow, OFDP, LLDP

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΧΑΡΙΣΤΙΕΣ 1
ΠΕΡΙΛΗΨΗ 2
ABSTRACT 3
1.Εισαγωγή 6
1.1 Γενικά 6
1.2 Δομή της τοπολογία δικτύου 7
1.2.1 Τοπολογίες δικτύων 9
1.3 Το πρόβλημα της ανακάλυψης τοπολογίας 10
1.4 Το αντικείμενο της διπλωματικής εργασίας και η συνεισφορά της 11
1.5 Δομή της εργασίας 12
2. Software Defined Networking 14
2.1 Ανάγκη για μετάβαση από τα συμβατικά δίκτυα στα δίκτυα SDN 14
2.2 Υφιστάμενη Κατάσταση (State of the Art) των δικτύων 15
2.3 Ορισμός του SDN και βασικές αρχές του 16
2.4 Αρχιτεκτονική SDN δικτύωσης 18
2.5 SDN Controllers 19
2.6 Πλεονεκτήματα των SDN 22
2.7 Εφαρμογές της τεχνολογίας SDN 23
3. Το πρωτόκολλο OpenFlow 26
3.1 Εισαγωγή 26
3.2 Τα δομικά μέρη του OpenFlow 26
3.2.1 OpenFlow controller 27
3.2.2 OpenFlow Switch 27
3.3 Εύρεση τοπολογίας δικτύου 28
3.3.1 Το πρωτόκολλο LLDP 29
3.3.2 Το πρωτόκολλο OFDP 32
3.4 Αξιολόγηση πρωτοκόλλου 33
3.4.1 Overhead σε μηνύματα 34
3.4.2 Χρόνος ανακάλυψης τοπολογίας δικτύου (Network Topology Discovery Time) . 34
4. Πρωτόκολλα ανακάλυψης της τοπολογίας και σύγκριση με το OFDP 36
4.1 Το πρωτόκολλο OFDPv2 37
4.2 Το πρωτόκολλο sOFTDP 40
4.3 Το πρωτόκολλο Self-Healing (SHTD) 42

4.4 Το πρωτόκολλο TEDP 44
4.5 Το πρωτόκολλο eTDP 46
4.6 Το πρωτόκολλο OFDPp 47
4.7 Το πρωτόκολλο Lightweight (LADP) 49
4.8 Το πρωτόκολλο Im-OFDP 52
4.9 Το πρωτόκολλο HDDP 54
4.10 Σύγκριση και συζήτηση των πρωτοκόλλων εύρεσης τοπολογίας 55
4.11 Δίκτυα SDN και πολυβάθμια δικτυακά συστήματα 59
5.Υλοποίηση δικτύων με τη βοήθεια του Mininet 60
5.1 Εισαγωγή στο Mininet 60
5.1.1 Πλεονεκτήματα και περιορισμοί του εξομοιωτή Mininet 61
5.1.2 Εγκατάσταση του Mininet σε εικονική μηχανή 62
5.1.3 Εντολές έναρξης και γραμμή εντολών του Mininet 63
5.1.4 Δημιουργία τοπολογιών στο Mininet 65
5.2 Βοηθητικά προγράμματα Xming και Putty 67
5.3 Αναλυτής πρωτοκόλλων δικτύου Wireshark 68
6.Αξιολόγηση πρωτοκόλλων OFDP, OFDPv2a, OFDPv2b με την βοήθεια του Mininet 70
6.1 Η μεθοδολογία της αξιολόγησης των πρωτοκόλλων 70
6.1.1 Μετρικές απόδοσης 70
6.1.2 Τοπολογίες δικτύου 71
6.1.3 Εκτέλεση προσομοιώσεων 71
6.2 Ανάλυση απόδοσης για γραμμική τοπολογία 72
6.3 Ανάλυση απόδοσης για ισοζυγισμένο δέντρο 74
6.4 Ανάλυση απόδοσης τυχαίας τοπολογίας erdos-renyi 77
6.5 Σύγκριση του χρόνου ανακάλυψης της τοπολογίας για τις τοπολογίες γραμμική, δέντρου και τυχαία 81
7. Συμπεράσματα της διπλωματικής 82
ΒΙΒΛΙΟΓΡΑΦΙΑ 84

1.Εισαγωγή

1.1 Γενικά

Τα σημερινά δίκτυα βασίζονται σε ένα μοντέλο διασύνδεσης και σε μια κατανεμημένη διαδικασία μεταφοράς της πληροφορίας. Παραδοσιακά, τα πρωτόκολλα δικτύου εκτελούνται στους δρομολογητές ή/και στους μεταγωγείς. Οι μεταγωγείς διασυνδέουν συσκευές ίδιου τύπου ενώ οι δρομολογητές μπορεί να διασυνδέουν συσκευές ή και δίκτυα διαφορετικού τύπου. Αυτή η δόμηση είναι περίπλοκη και ιδιαίτερα δύσκολη ως προς τη διαχείρισή της καθώς υπάρχουν δύο βασικά προβλήματα που πρέπει να αντιμετωπιστούν: η ετερογένεια, δηλαδή η διαφορετικότητα των συσκευών, δικτύων και τεχνολογιών και η κλίμακα, ο τεράστιος αριθμός των συσκευών που απαρτίζουν το δίκτυο, και που αυξάνεται συνεχώς. Σύμφωνα με εκτιμήσεις το μέγεθος του διαδικτύου διπλασιάζεται κάθε 30 χρόνια.

Αυτή η πραγματικότητα απαιτεί την αντιμετώπιση μιας σειράς προκλήσεων. Οι διαχειριστές δικτύου πρέπει να διαμορφώσουν μεμονωμένα, μία προς μία, κάθε συσκευή δικτύου για να λειτουργήσει σύμφωνα με τις επιθυμητές πολιτικές και ανάγκες σε υψηλό επιπέδου δικτύου. Ένα περιβάλλον δικτύου πρέπει να έχει τη δυνατότητα προσαρμογής στις αλλαγές φορτίου και της αντοχής στη δυναμική των σφαλμάτων. Και, το σημαντικότερο από όλα, είναι η δημιουργία ενός συστήματος δρομολόγησης που θα μπορεί να διαχειρίζεται τη μετάδοση των μηνυμάτων από τα εκατοντάδες χιλιάδες δίκτυα και δισεκατομμύρια τελικών κόμβων. Δύο από τις πιο βασικές λειτουργίες που πρέπει να εκτελούνται σε μία δικτύωση είναι η προώθηση των πακέτων (forwarding) και η δρομολόγηση (routing) τους.

Η προώθηση πακέτων είναι η βασική μέθοδος για την ανταλλαγή πληροφοριών μεταξύ συστημάτων σε ένα δίκτυο. Τα πακέτα που φθάνουν σε μία δικτυακή συσκευή προωθούνται από μία διεπαφή εισόδου σε μία ή περισσότερες διεπαφές εξόδου. Σύμφωνα με το μοντέλο OSI, το επίπεδο ζεύξης δεδομένων ($2^{ο}$ επίπεδο) είναι υπεύθυνο για την προώθηση πακέτων. Η διαδικασία της προώθησης βασίζεται στους πίνακες προώθησης της συσκευής και στην πληροφορία που φέρει το πακέτο.

Η δρομολόγηση είναι η διαδικασία με την οποία συσκευές του 3ου επιπέδου (π.χ. δρομολογητής) αποφασίζουν ποια διαδρομή θα ακολουθήσει ένα πακέτο και στη συνέχεια το προωθούν στην κατάλληλη θύρα. Ουσιαστικά, πρώτα δημιουργείται ο

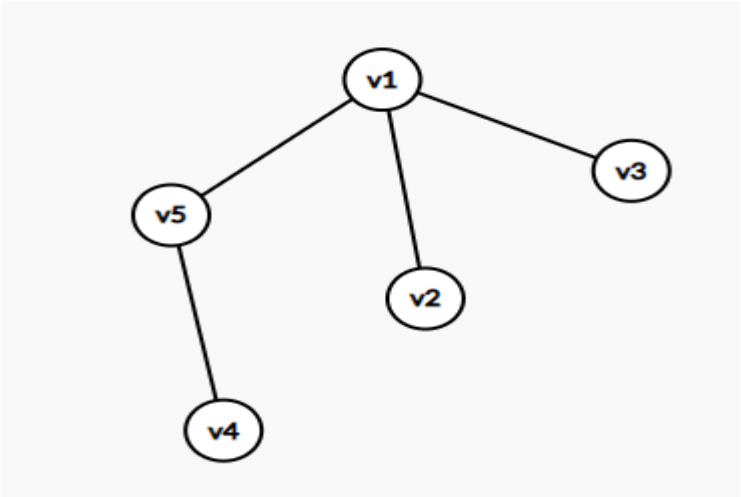
πίνακας προώθησης με την συνδρομή των πρωτοκόλλων δρομολόγησης και μετά προωθείται στην κατάλληλη θύρα το πακέτο. Όταν το σύστημα προέλευσης και το σύστημα προορισμού βρίσκονται στο ίδιο τοπικό δίκτυο, η διαδρομή που συνδέει τα δύο συστήματα μεταξύ τους ονομάζεται *απευθείας* διαδρομή . Εάν ένα πακέτο πρέπει να ταξιδεύει τουλάχιστον ένα βήμα (hop) πέρα από το σύστημα προέλευσής του, η διαδρομή μεταξύ του συστήματος προέλευσης και του συστήματος προορισμού ονομάζεται έμμεση διαδρομή . Τα πρωτόκολλα δρομολόγησης μαθαίνουν τη διαδρομή προς μια διεπαφή προορισμού και διατηρούν δεδομένα σχετικά με γνωστές διαδρομές στον πίνακα δρομολόγησης του συστήματος .Το επίπεδο δικτύου 3 στο μοντέλο OSI είναι υπεύθυνο για την δρομολόγηση.

1.2 Δομή της τοπολογία δικτύου

Ένας συνηθισμένος τρόπος αναπαράστασης των δικτύων είναι με γράφο. Γράφος (ή γράφημα) [30], ονομάζεται ένα διατεταγμένο ζεύγος G = (V, E), όπου V είναι μη κενό σύνολο κορυφών και E ένα σύνολο μη ακμών του V, δηλαδή

$$E \subseteq \binom{V}{2}$$

Ακολουθεί το παράδειγμα της εικόνας 1.1 για να κατανοήσουμε την έννοια του γράφου.



Εικόνα 1.1:Παράδειγμα γραφήματος με 5 κόμβους

Αν V = {v1, v2, v3, v4, v5} είναι ένα μη κενό σύνολο στοιχείων και E = {{v1, v2}, {v1, v3}, {v4, v5}} τότε το διατεταγμένο ζεύγος G = (V, E) είναι ένας γράφος. Τα

στοιχεία του μη κενού συνόλου V λέγονται κορυφές ή κόμβοι (vertices, nodes) του γράφου. Τα στοιχεία του συνόλου E λέγονται ακμές (edges) και μπορούν να συμβολιστούν και με ένα γράμμα, π.χ. e, όπου e = {x, y}, x, y ∈ V , x≠ y. Καμιά φορά, καταχρηστικώς, θεωρούμε και ακμές-βρόχους, δηλαδή e = {x, x}.

Επομένως οι κόμβοι (vertices) αναπαριστούν τις διάφορες συσκευές που μετέχουν ή αλληλοεπιδρούν με το δίκτυο (π.χ. switches, access point) και ως ακμές (edges) θεωρούνται οι τηλεπικοινωνιακές συνδέσεις (links) μεταξύ των κόμβων.

Η αναπαράσταση ενός δικτύου δηλαδή ο τρόπος που οι κόμβοι του διασυνδέονται μεταξύ τους και επικοινωνούν ορίζει την τοπολογία του δικτύου και αντιστοιχεί είτε στη φυσική είτε στη λογική θεώρηση του δικτύου. Στη *φυσική τοπολογία* απεικονίζεται σε ποια θέση του δικτύου βρίσκονται τα στοιχεία στο χώρο. Στη *λογική τοπολογία* παρουσιάζεται η ροή των δεδομένων μέσα στο δίκτυο. Δύο δίκτυα τα οποία έχουν την ίδια φυσική τοπολογία δεν είναι απαραίτητο να έχουν και ίδια λογική τοπολογία. Υπάρχει το ενδεχόμενο να έχουν διαφορετική λογική γιατί μπορεί να διαφέρουν τόσο ως προς το τεχνολογικό μέρος δηλαδή στις συσκευές όσο και ως προς τα μέσα μετάδοσης.

Στην τοπολογία ενός δικτύου τα κύρια συστατικά μέρη του είναι:

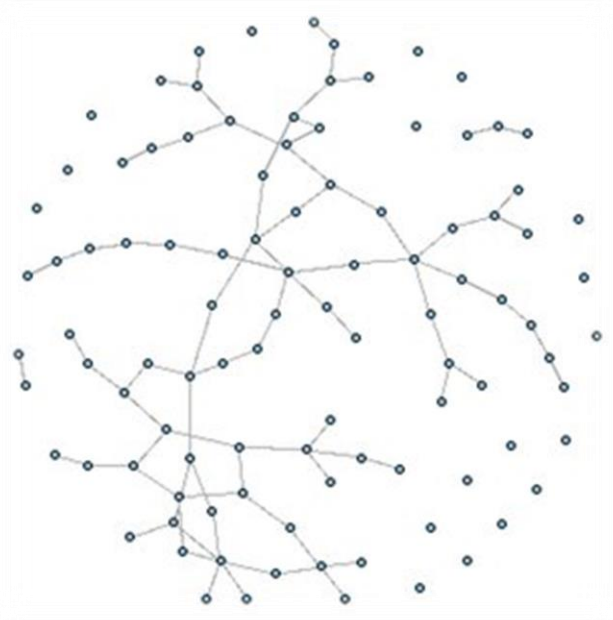
- *Οι ζεύξεις*: ονομάζονται τα φυσικά μέσα μεταφοράς της πληροφορίας που συνδέουν τις συσκευές του εκάστοτε δικτύου και μπορεί να είναι π.χ. ηλεκτρικά καλώδια, οπτικές ίνες κ.α.
- *Οι κόμβοι:* είναι οι συσκευές που χρησιμοποιούν τις ζεύξεις για να επικοινωνούν μεταξύ τους και για αυτό το λόγο έχουν μία ή παραπάνω διεπαφές δικτύου (network interfaces). Οι πιο συνήθης είναι:
  - Οι *μεταγωγείς* (*switches*) που κάνουν την δουλειά που κάνουν τα hubs και οι bridges μαζί. Με άλλα λόγια λαμβάνουν την πληροφορία από μια ζεύξη (όπως π.χ. τα hubs) και την μεταβιβάζουν μόνο στη ζεύξη προορισμού του (όπως π.χ. bridges).
  - Οι *δρομολογητές* (*routers*) η δουλειά τους είναι να επεξεργάζονται την πληροφορία που λαμβάνουν και να την προωθούν σε άλλα δίκτυα.
  - Ο ελεγκτής διασύνδεσης δικτύου (NIC) είναι η διεπαφή όπου χορηγεί άδεια σε μια συσκευή του δικτύου να συνδεθεί μέσω μιας ζεύξης.

Για να γίνει η μετάδοση των δεδομένων σε ένα δίκτυο, θα πρέπει αυτά να ακολουθήσουν μια απολύτως συγκεκριμένη διαδρομή από όλες τις δυνητικά διαθέσιμες που υπάρχουν στο δίκτυο. Η επιλογή της διαδρομής εξαρτάται από την τοπολογία αλλά και από τη γνώση που υπάρχει για την τοπολογία του δικτύου.

1.2.1 Τοπολογίες δικτύων

Τα πραγματικά δίκτυα σχηματίζουν διάφορες τοπολογίες. Ενδεικτικά αναφέρουμε την *γραμμική* τοπολογία, την *δενδρική* τοπολογία και την *τυχαία* τοπολογία.

Στη γραμμική τοπολογία ένας αριθμός από δικτυακούς κόμβους(π.χ. switches) συνδέονται γραμμικά μεταξύ τους, ενώ στον κάθε κόμβο μπορεί να συνδέονται ένας ή περισσότεροι hosts. Στην τοπολογία δέντρου οι δικτυακοί κόμβοι τοποθετούνται σε επίπεδα, σχηματίζοντας μια δεντρική δομή. Στο τελευταίο επίπεδο τοποθετούνται οι hosts. Στην τυχαία τοπολογία ενός δικτύου με N κόμβους, δυο κόμβοι συνδέονται μεταξύ τους με πιθανότητα p. Ένας τέτοιος γράφος αναφέρεται ως Erdős–Rényi [44][46]. Με άλλα λόγια από το σύνολο των N(N-1)/2 δυνητικά διαθέσιμων ακμών, η κάθε μία από αυτές, επιλέγεται μια από αυτές με πιθανότητα p και τοποθετείται στο γράφο.



Εικόνα 1.1: Μοντέλο Erdős–Rényi

1.3 Το πρόβλημα της ανακάλυψης τοπολογίας

Η ανακάλυψη της τοπολογίας του δικτύου είναι ζωτικής σημασίας για πολλές διαδικασίες στα δίκτυα όπως, δρομολόγηση, διαχείριση δικτύου, κατανομή πόρων και διαμόρφωση, ποιότητας υπηρεσιών (QoS), διάγνωση και αποκατάσταση σφαλμάτων. Σε αυτή τη κατεύθυνση, η ελαχιστοποίηση του χρόνου εντοπισμού της τοπολογίας του δικτύου είναι θεμελιώδης για την έγκαιρη και αποφασιστική απόκριση του δικτύου, η οποία και θα πρέπει να γίνει σε πραγματικό χρόνο.

Στα παραδοσιακά δίκτυα, η ανακάλυψη τοπολογίας ήταν χρήσιμη μόνο σε λειτουργίες δικτύου σχετικές με τον έλεγχο, τη διαχείριση του και την δρομολόγηση που γινόταν με τη βοήθεια πρωτοκόλλων δρομολόγησης, όπως το Routing Information Protocol (RIP) ή το Open Shortest Path First (OSPF) [14].

Σε επίπεδο δικτύου, η μεταφορά δεδομένων περιγράφεται από δυο όρους: *προώθηση* και *δρομολόγηση*. Η διαδικασία της *προώθησης* λαμβάνει χώρα στον δρομολογητή και έχει ως αποτέλεσμα τη μεταφορά του πακέτου από μια διεπαφή εισόδου στην κατάλληλη διεπαφή εξόδου. Η *δρομολόγηση* καθορίζει τη διαδρομή που θα ακολουθήσουν τα πακέτα από την πηγή στον προορισμό με τη χρήση κατάλληλων αλγορίθμων δρομολόγησης και τη βοήθεια του πίνακα προώθησης του εκάστοτε δρομολογητή.

Για την επίτευξη της δικτύωσης, ο *κάθε δικτυακός κόμβος* πρέπει να ρυθμιστεί ξεχωριστά, έτσι ώστε να:

α) να δρομολογεί την κίνηση, αφού λάβει τις κατάλληλες αποφάσεις

β) να προωθεί τα πακέτα στον προορισμό τους, και να

γ) διαχειρίζεται τη δικτυακή κίνηση.

Σε κάθε συσκευή τοποθετείται ένα *επίπεδο δεδομένων* και ένα *επίπεδο ελέγχου* και κάθε συσκευή επιτελεί το έργο της κάνοντας χρήση πολλών και διαφορετικών πρωτοκόλλων. Η αρχική διαμόρφωση ή η επαναδιαμόρφωση αυτών των συσκευών, γίνεται «κατά μονάς», δηλαδή ανά συσκευή από τον διαχειριστή του συστήματος. Αυτές οι συσκευές μέχρι και σήμερα είναι κλειστά συστήματα, που σημαίνει ότι δεν μπορεί να γίνουν αλλαγές, τροποποιήσεις ή ενσωμάτωση νέου κώδικα, στον κώδικα του.

Η λύση στα παραπάνω προβλήματα που θα συμβάλει στο μετριασμό της πολυπλοκότητας αλλά και στη μείωση του κόστος εγκατάστασης και συντήρησης του δικτύου είναι η *αποσύνδεση* του *επιπέδου ελέγχου* από το *επίπεδο δεδομένων*.

Ουσιαστικά, η διαδικασία της προώθησης των δεδομένων παραμένει σε επίπεδο συσκευής, ενώ ο έλεγχος γίνεται απομακρυσμένα. Με άλλα λόγια, γίνεται ένας φυσικός διαχωρισμός σε *επίπεδο δεδομένων* και σε επίπεδο ελέγχου.

Η λογική της τεχνολογίας αυτής αφορά τον διαχωρισμό του *επιπέδου ελέγχου* (control plane) από το *επίπεδο δεδομένων* (data plane). Στο επίπεδο δεδομένων γίνεται η *προώθηση* της κίνησης ενώ στο επίπεδο *ελέγχου* λαμβάνονται οι αποφάσεις για όλες τις λειτουργίες ελέγχου και την προώθηση. Το *επίπεδο ελέγχου* αναλαμβάνει τον υπολογισμό και την διανομή των πινάκων προώθησης της κάθε συσκευής (δρομολογητή). Το αποτέλεσμα αυτής της ιδέας είναι η δικτύωση καθοριζόμενη από το λογισμικό, και η οποία οδηγεί σε ένα ανοιχτό πλέον σύστημα. Τα συστήματα αυτά είναι πιο ευέλικτα και λειτουργικά καθώς μπορούμε να προγραμματίσουμε συνολικά τους δικτυακούς κόμβους και όχι μια μόνο μεμονωμένη συσκευή. Επομένως, οι εφαρμογές αφαιρούνται από τους κόμβους και τοποθετούνται για να τρέχουν πάνω από το ενιαίο λειτουργικό σύστημα το οποίο διαχειρίζεται όλο το δικτύωμα.

Στη δικτύωση SDN μία συσκευή στο *επίπεδο ελέγχου* μπορεί να διαχειριστεί πολλά στοιχεία στο *επίπεδο δεδομένων*. Κάθε δικτυακό στοιχείο έχει *πίνακες* με *κανόνες*. Κάθε *κανόνας* ταυτίζεται με κάποιο τμήμα κίνησης και δρα πάνω σε αυτό. Ανάλογα με τους *κανόνες* που έχουν εγκατασταθεί στον SDN Controller (ή στο δικτυακό switch), το switch αυτό μπορεί να λειτουργήσει είτε ως δρομολογητής, είτε ως μεταγωγέας, είτε ως τείχος προστασίας, είτε ως οποιαδήποτε άλλη δικτυακή συσκευή.

Οι αποφάσεις για τη δρομολόγηση και προώθηση των δεδομένων λαμβάνουν υπόψη την τοπολογία. Ο ελεγκτής "ανακαλύπτει" την τοπολογία και έχοντας αυτή τη γνώση αλληλοεπιδρά με τις εφαρμογές. Για παράδειγμα, μια εφαρμογή δρομολόγησης χρησιμοποιεί την τοπολογία του δικτύου για να πετύχει την δρομολόγηση της κυκλοφορίας του δικτύου. Η "ανακάλυψη" μιας τοπολογίας αφορά την εύρεση:

α) των κόμβων (hosts),

β) των μεταγωγέων (switches)

γ) των διασυνδέσεων (links) μεταξύ των μεταγωγέων.

1.4 Το αντικείμενο της διπλωματικής εργασίας και η συνεισφορά της

Το αντικείμενο της παρούσας διπλωματικής εργασίας είναι η διερεύνηση των μηχανισμών *εύρεσης της τοπολογίας* σε δίκτυα SDN. Γίνεται μελέτη και παρουσίαση των διάφορων *μεθόδων εύρεσης τοπολογίας* που έχουν προταθεί μέχρι σήμερα.

Ένα σημαντικό ζήτημα που απαιτείται για την επιτυχία της νέας αυτής τεχνολογίας είναι η "ανακάλυψη" της τοπολογίας του υφιστάμενου δικτύου, καθώς αυτή θα χρησιμοποιηθεί για την διαχείριση της κίνησης στο δίκτυο. Η ανακάλυψη της τοπολογίας, όπως αναφέρθηκε και παραπάνω, αφορά τον εντοπισμό των κόμβων και τον εντοπισμό των συνδέσμων με τους γειτονικούς τους κόμβους

Στα πλαίσια αυτής της εργασίας η συνεισφορά μας συνοψίζεται στα ακόλουθα:

- Εισάγουμε τις βασικές έννοιες της *καθοριζόμενης από το λογισμικό δικτύωσης* και διερευνούμε το πρόβλημα της "ανακάλυψης" της τοπολογίας του δικτύου.
- Παρουσιάζουμε και συγκρίνουμε τις υπάρχουσες προσεγγίσεις για την "ανακάλυψη" της τοπολογίας σε θεωρητικό επίπεδο
- Εξετάζουμε την απόδοση των *πρωτοκόλλων εντοπισμού* της τοπολογίας χρησιμοποιώντας διάφορες μετρικές, όπως ο *χρόνος εντοπισμού της τοπολογίας*, *το πλήθος των μηνυμάτων*, κ.ά. για διάφορες τοπολογίες και για διαφορετικό αριθμό κόμβων.
- Συγκρίνουμε πειραματικά, ως προς την απόδοσή τους τα βασικότερα πρωτόκολλα ανακάλυψης τοπολογίας και εξετάζουμε τη συμπεριφορά τους σε διάφορες δικτυακές τοπολογίες, όπως:
    - Γραμμική
    - Ισοζυγισμένου δέντρου και
    - Τυχαίου Γράφου

1.5 Δομή της εργασίας

Η εργασία αποτελείται από εφτά επιμέρους κεφάλαια. Στο πρώτο κεφάλαιο γίνεται αναφορά σε βασικές έννοιες των δικτύων, και ορίζεται το πρόβλημα *εύρεσης της τοπολογίας*. Το δεύτερο κεφάλαιο εισάγεται η τεχνολογία των δικτύων SDN και παρουσιάζονται τα βασικά τους στοιχεία. Στο τρίτο κεφάλαιο παρουσιάζεται το πρωτόκολλο OpenFlow και ο τρόπος λειτουργίας του. Στο τέταρτο κεφάλαιο γίνεται ανασκόπηση της βιβλιογραφίας και αναλυτική αναφορά στις μεθόδους εύρεσης τοπολογίας δικτύου σε δίκτυα SDN. Η βασικότερη μέθοδος εύρεσης τοπολογίας ενός δικτύου SDN είναι το πρωτόκολλο OFDP, το οποίο εμφανίζεται σε διάφορες εκδοχές. Στο πέμπτο κεφάλαιο παρουσιάζεται η χρήση των επιμέρους λειτουργιών του λογισμικού Mininet**,** το οποίο είναι και το βασικό λογισμικό στο οποίο στηρίχτηκε η

διεξαγωγή πειραμάτων για την αξιολόγηση των OFDP βασικών εκδόσεων του OFDP. Στο έκτο κεφάλαιο παρουσιάζονται οι πειραματικές μετρήσεις οι οποίες αφορούν το πρωτόκολλο OFDP με τις παραλλαγές του εφαρμοζόμενο σε διάφορες τοπολογίες, και εξετάζονται διάφορες μετρικές απόδοσης, όπως *ο χρόνος ανακάλυψης της τοπολογίας*, ή η *χρήση φυσικών πόρων του συστήματος* (παράδειγμα η μνήμη RAM ή η χρήση της CPU). Στο έβδομο και τελευταίο κεφάλαιο ολοκληρώνει τη μελέτη και παρουσιάζονται τα βασικά συμπεράσματα που έχουν προκύψει.

2. Software Defined Networking

2.1 Ανάγκη για μετάβαση από τα συμβατικά δίκτυα στα δίκτυα SDN

Η ραγδαία διάδοση των φορητών συσκευών και η αλματώδη αύξηση των δεδομένων που παράγουν και μεταδίδουν, το virtualization των servers και η εμφάνιση των υπηρεσιών cloud συμπεριλαμβάνεται στις τάσεις της εποχής, η οποία οδηγεί τη βιομηχανία της δικτύωσης να εξετάσει εκ νέου τις υπάρχουσες παραδοσιακές αρχιτεκτονικές δικτύων [3] [6].

Τα περισσότερα από τα παραδοσιακά δίκτυα είναι ιεραρχικά, δηλαδή είναι βασισμένα σε τρία επίπεδα: το επίπεδο *κορμού* (core), το επίπεδο *διανομής* (distribution) και το επίπεδο *πρόσβασης* (access). Η συγκεκριμένη αρχιτεκτονική δεν μπορεί πλέον να ανταποκριθεί στις δυναμικές ανάγκες computing και storage των σημερινών data centers. Τα δίκτυα SDN [7][11], έρχονται να αντιμετωπίσουν τις σύγχρονες προκλήσεις όπως είναι για παράδειγμα η διαμόρφωση της κάθε συσκευής από τον διαχειριστή, το κόστος, ο χρόνος που απαιτούνται για την διαμόρφωση κ.λπ.. Η ικανοποίηση αυτών των αναγκών θα προάγει τις παρεχόμενες υπηρεσίες.

Όταν πριν μερικά χρόνια, όλη η επιστημονική κοινότητα άρχισε να μιλάει για την SDN δικτύωση [12], θεωρούσε ότι πρόκειται για μια πρακτική διαχωρισμού του ελέγχου του δικτύου από το χειρισμό των πακέτων των δεδομένων που διακινούνται εντός του. Τα παραδοσιακά δίκτυα είχαν ήδη ξεκινήσει να υιοθετούν τη φιλοσοφία αυτή, με την προσθήκη ελεγκτών (controllers) μέσα σε μεταγωγείς (switches) και τη χρήση διαφόρων τεχνολογιών βασικής υποδομής data center. Τα SDN δίκτυα αξιοποίησαν αυτή την ιδέα και την εξέλιξαν καταργώντας την ανάγκη ο controller και ο διαχειριστής των πακέτων να βρίσκονται μέσα στην ίδια τη συσκευή ή ακόμα να προέρχονται από τον ίδιο κατασκευαστή.

Επομένως η τεχνολογία SDN, τοποθετεί το επίπεδο ελέγχου του traffic ένα επίπεδο πάνω από το hardware – χωρίς να εξαρτάται όμως από αυτό, δηλαδή από τον εκάστοτε κατασκευαστή – προσφέροντας παράλληλα προηγμένες δυνατότητες διαχείρισης και παρακολούθησης μέσα από έναν κεντρικό ελεγκτή. Με τη δικτύωση SDN, δίνονται στον διαχειριστή του δικτύου, απεριόριστες δυνατότητες διαχείρισης και επέμβασης όπου αυτό απαιτείται, χωρίς – ο διαχειριστής - να απασχολείται με άλλα ζητήματα, όπως π.χ. το πώς συνδέονται τα switches και οι διάφορες άλλες δικτυακές συσκευές

μέσα στο δίκτυο. Με τη χρήση της τεχνολογίας SDN, οι ρυθμίσεις συσκευών, η εφαρμογή διαφόρων πολιτικών κ.λπ., γίνεται ταχύτατα χωρίς την ανάγκη φυσικής παρουσίας του διαχειριστή. Αυτό αφενός σημαίνει ευκολία στη διαχείριση του δικτύου και αφετέρου μείωση κόστους από πλευράς επιχείρησης.

Το αποτέλεσμα είναι μια σύγχρονη υποδομή η οποία μπορεί να ενεργοποιεί νέες εφαρμογές και υπηρεσίες μέσα σε λίγα λεπτά, αντί για ημέρες ή και εβδομάδες που απαιτούνταν στο παρελθόν τα συμβατικά δίκτυα. Τα βασικά ζητήματα που αντιμετωπίζουν τα SDNs δίκτυα [14] είναι η ασφάλεια, η επεκτασιμότητα και η ευελιξία.

2.2 Υφιστάμενη Κατάσταση (State of the Art) των δικτύων

Οι παραδοσιακές τεχνολογίες δικτύωσης είναι «στατικές, άκαμπτες και μη ευέλικτες», και βασίζονται σε τυποποιημένα πρωτόκολλα επικοινωνίας που δεν αφήνουν αρκετά περιθώρια για βελτιώσεις [12][14]. Η δόμηση τους είναι ιδιαίτερα σύνθετη και η διαχείρισή τους δύσκολη. Η υλική υποδομή που χρησιμοποιείται είναι κλειστού τύπου και δεν επιτρέπει παρεμβάσεις στο λογισμικό πέρα από τους ίδιους τους κατασκευαστές. Επιπλέον απαιτεί τη ξεχωριστή διαμόρφωση της κάθε δικτυακής συσκευής (δρομολογητή, switch) που μετέχει στο δίκτυο. Σήμερα υπάρχουν πολλά διαφορετικά πρωτόκολλα δικτύωσης, διάφορες τοπολογίες, αλλά και πλήθος περιορισμών: στην επεκτασιμότητα, στην ευελιξία και στην απόδοση του δικτύου καθώς το επίπεδο δεδομένων, το επίπεδο ελέγχου και η διαχείριση όχι μόνο συνδέονται μεταξύ τους, αλλά βρίσκονται όλα μαζί σε κάθε συσκευή δικτύωσης.

Όλα τα παραπάνω συνάγουν στην ανάγκη ανάπτυξης και αντικατάστασης των συμβατικών δικτυών με μία νέα καινοτόμα αρχιτεκτονική όπως είναι η SDN έχοντας πολλά πλεονεκτήματα αλλά και προκλήσεις που πρέπει να αντιμετωπίσουν οι προγραμματιστές της SDN δικτύωσης. Η συγκεκριμένη τεχνολογία είναι καινούργια και όπως ισχυρίζονται πολλοί ερευνητές θα είναι το μέλλον στα δίκτυα αλλά βρίσκεται ακόμη σε ένα πρώιμο στάδιο ανάπτυξης.

Ένα σημαντικό ζήτημα που απαιτείται για την επιτυχία της νέας τεχνολογίας είναι η ανακάλυψη της τοπολογίας του υφιστάμενου δικτύου, καθώς αυτή θα χρησιμοποιηθεί για την διαχείριση της κίνησης στο δίκτυο. Η ανακάλυψη της τοπολογίας, όπως

αναφέρθηκε και παραπάνω, αφορά τον εντοπισμό του κόμβου και τον εντοπισμό των συνδέσμων με τους γειτονικούς του κόμβους. Η διαδικασία "ανακάλυψης" της τοπολογίας του δικτύου βασίζεται σε πρωτόκολλα, όπως το του Link Layer Discovery Protocol (LLDP) [8] και το OpenFlow Discovery Protocol (OFDP) [19]. Τα πρωτόκολλα αυτά βασίζονται στην ανταλλαγή μηνυμάτων μεταξύ των κόμβων που μετέχουν στην τοπολογία. Το πρωτόκολλο OFDP αξιοποιεί το LLDP με μικρές τροποποιήσεις για την εκτέλεση εντοπισμού τοπολογίας σε ένα δίκτυο SDN. Αυτές οι τροποποιήσεις επηρεάζουν τον τρόπο λειτουργίας του OFDP. Το πρωτόκολλο LLDP λειτουργεί με ενημερώσεις που προορίζονται για πολλαπλή διανομή με χρήση διευθύνσεων MAC (όπως ορίζεται στο πρότυπο 802.1d). Αυτό σημαίνει ότι ένας μεταγωγέας θα λαμβάνει μόνο ενημερώσεις LLDP από τον άμεσο συνδεδεμένο γειτονικό μεταγωγέα (δηλαδή, επίπεδο-1) και δεν θα προωθήσει ποτέ τις ενημερώσεις LLDP που έχουν λάβει. Καθώς πρόκειται για σχετικά νέα τεχνολογία, οι ερευνητές εξετάζουν τα θετικά αλλά και τα αρνητικά στοιχεία των παραπάνω πρωτοκόλλων και προτείνουν νέες λύσεις, τις οποίες εξετάζουμε σε αυτή την εργασία.

2.3 Ορισμός του SDN και βασικές αρχές του

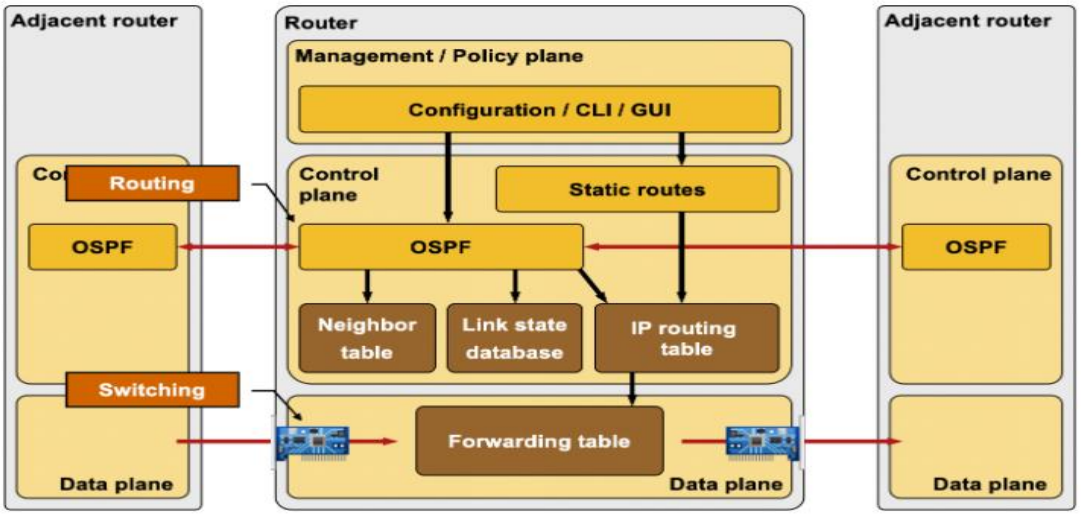
Υπάρχει μία πληθώρα από ορισμούς οι οποίοι σχετίζονται με τη δικτύωση SDN καθώς η τεχνολογία αυτή βρίσκεται ακόμη σε πολύ πρώιμο στάδιο ανάπτυξης. Το SDN αποτελεί μία δικτυακή αρχιτεκτονική μέσω της οποίας η διαχείριση των αποφάσεων για την δρομολόγηση λαμβάνεται από ένα κεντρικό σημείο διαχείρισης, τον ελεγκτή SDN [14].

Στα υφιστάμενα δίκτυα που βασίζονται στις βασικές αρχές των κλασικών δικτύων υπάρχει ένας διαχωρισμός των λειτουργιών τους σε τρεις βασικές κατηγορίες [32]:

1. *Επίπεδο διαχείρισης (Management Plane):* αναφέρεται σε εκείνες τις υπηρεσίες οι οποίες προσφέρουν την διαχείριση και την επίβλεψη στην εύρυθμη λειτουργία του εξοπλισμού από την πιο απλή πρόσβαση με τη βοήθεια της γραμμής εντολών (Command Line Interface(CLI)) μέχρι την χρήση πιο εξειδικευμένων εφαρμογών όπως π.χ. Netflow.
2. *Επίπεδο δεδομένων (Data Plane):* αναφέρεται σε εκείνες τις διεπαφές ή τα ports (πόρτες) που χρησιμοποιούνται τόσο για την μεταφορά όσο και για την λήψη πακέτων στις οποίες για να γίνει η επεξεργασία τους χρησιμοποιούνται οι

αντίστοιχοι πίνακες (routing πίνακες, TCAM πίνακες, Forwarding Information Base (FIB)).

3. *Επίπεδο ελέγχου (Control Plane):* αναφέρεται στην λήψη αποφάσεων για την επεξεργασία των πακέτων καθώς επίσης και στην δημιουργία των απαραίτητων πινάκων στους οποίους βασίζεται το επίπεδο δεδομένων για να επιτευχθεί η δρομολόγησή τους.



Εικόνα 2.1: Βασικός διαχωρισμός των λειτουργιών του δικτύου

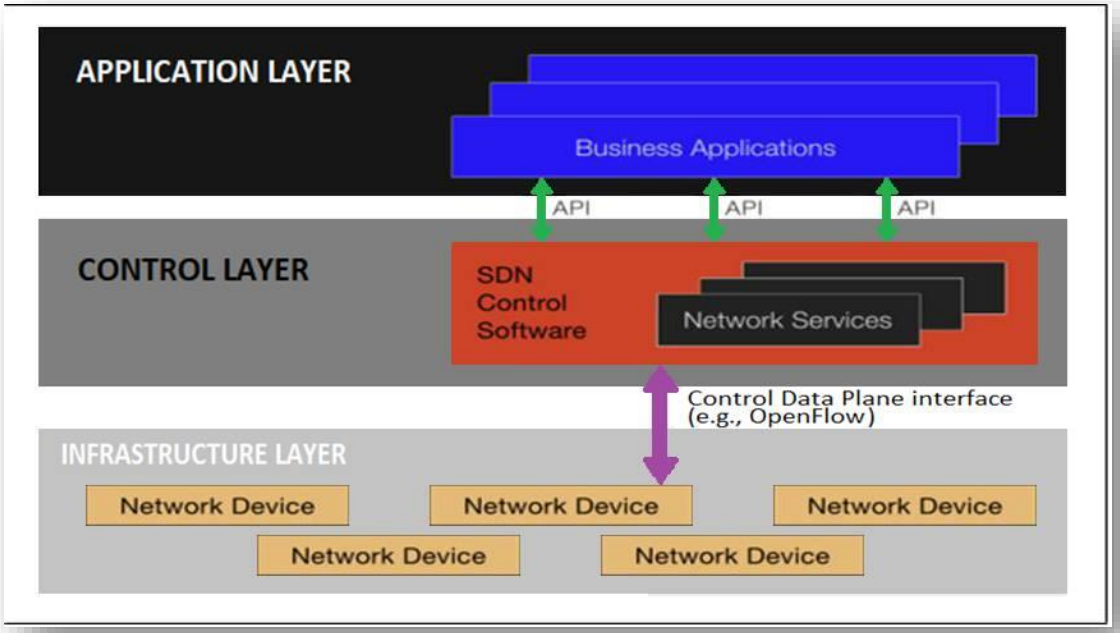
Το *επίπεδο ελέγχου* θεωρείται το «μυαλό» του εξοπλισμού καθώς σε αυτό λαμβάνονται οι αποφάσεις για την καλύτερη δρομολόγηση, την αποφυγή των βρόχων-επαναλήψεων, την ανακατεύθυνση (failover) σε περίπτωση βλάβης καθώς επίσης και την ανάκτηση της λειτουργίας σε περίπτωση σφάλματος. Παρόλα αυτά σε μια τεχνολογία SDN υπάρχει ο διαχωρισμός μεταξύ των επιπέδων ελέγχου και φυσικού. Το επίπεδο ελέγχου αποσπάται από τις αρμοδιότητες του εξοπλισμού και μετακινείται στον κεντρικό ελεγκτή. Με αυτόν τον τρόπο ο εξοπλισμός δεν είναι αναγκαίο πλέον να λαμβάνει αποφάσεις πάρα μόνο χρησιμοποιείται για να προωθεί τα πακέτα λαμβάνοντας υπόψιν τις πληροφορίες και τους πίνακες που δημιουργεί και ενημερώνει ο ελεγκτής.

2.4 Αρχιτεκτονική SDN δικτύωσης

Η αρχιτεκτονική της τεχνολογίας SDN [14] αποτελείται από τρία δομικά στοιχεία:

- *Εφαρμογές*: οι οποίες επικοινωνούν άμεσα με τους SDN ελεγκτές μέσω ενός API έτσι ώστε να λαμβάνουν τις κατάλληλες πληροφορίες για την συμπεριφορά και την δομή ενός δικτύου. Βασιζόμενοι σε αυτές τις πληροφορίες μπορούν να έχουν επακριβή εικόνα τόσο για την τοπολογία του δικτύου όσο και των πιθανών αλλαγών που ίσως προκύψουν και να δράσουν ανάλογα.
- *Ελεγκτής:* είναι μία λογική οντότητα μεταξύ των εφαρμογών και των δικτυακών συσκευών. Παίρνει τις εντολές από τις εφαρμογές και τις προωθεί στις συσκευές ως κομμάτι του επιπέδου ελέγχου. Παράλληλα, παίρνει όλες τις απαραίτητες πληροφορίες για την κατάσταση του δικτύου από τις συσκευές και τις προωθεί στις εφαρμογές.
- *Δικτυακή Υποδομή:* αναλαμβάνουν τόσο τον ρόλο της προώθησης των πακέτων στο δίκτυο ανάλογα με τις πληροφορίες όσο και τις εντολές που λαμβάνουν από τον ελεγκτή.

Η επικοινωνία μεταξύ των δομικών στοιχείων του δικτύου SDN βασίζεται στις διεπαφές προγραμματισμού εφαρμογών (APIs) της αρχιτεκτονικής SDN. Υπάρχουν δύο τέτοιες διεπαφές που συχνά αναφέρονται και ως *βόρεια διεπαφή* (*Northbound interface*) και *νότια διεπαφή* (*Southbound interface*). Η λεγόμενη *βόρεια διεπαφή* ορίζει την επικοινωνία μεταξύ των εφαρμογών και του ελεγκτή, ενώ η *νότια διεπαφή* αναλαμβάνει την επικοινωνία του ελεγκτή με το επίπεδο δεδομένων και την υποδομή. Μιλώντας για ένα πλήρως εικονοποιημένο δίκτυο δεν απαιτείται τα δομικά αυτά στοιχεία να βρίσκονται στον ίδιο φυσικό χώρο.



Εικόνα 2.2: Βασική αρχιτεκτονική SDN

Για να επιτευχθεί η επικοινωνία μεταξύ του ελεγκτή και του εξοπλισμού είναι αναγκαία η χρήση ενός πρωτοκόλλου, το οποίο θα πρέπει να μεταφέρει τα μηνύματα χωρίς να νοιάζεται για το λογισμικό που έχει στην διάθεση του και τον κατασκευαστή από τον οποίο προέρχεται, αλλά μόνο για την υποστήριξή του στο συγκεκριμένο πρωτόκολλο. Για αυτήν την συγκεκριμένη διαδικασία, το πλέον επικρατέστερο πρωτόκολλο είναι το OpenFlow, στο οποίο θα αναφερθούμε σε αμέσως επόμενη ενότητα με περισσότερες λεπτομέρειες.

2.5 SDN Controllers

Υπάρχει ένας μεγάλος αριθμός από SDN ελεγκτές [28] που αναπτυχθήκαν κατά καιρούς για διαφορετικούς σκοπούς. Θα αναφερθούμε στις πιο χαρακτηριστικές περιπτώσεις ελεγκτών SDN και θα γίνει μια σύντομη περιγραφή του καθενός.

- *NOX*: αποτελεί έναν από τους πρώτους αν όχι ο πρώτος OpenFlow ελεγκτής. Ο NOX controller έχει αναπτυχθεί αποκλειστικά στην C++. Είναι βασισμένο στην ιδέα του δικτυακού λειτουργικού συστήματος (NOS). Επομένως, αποτελεί

την βάση πάνω στην οποία προσφέρουν οι διάφορες λειτουργίες του δικτύου. Έτσι, λειτουργίες όπως η δρομολόγηση, ο έλεγχος κίνησης κλπ. παρουσιάζονται σαν εφαρμογές οι οποίες είναι ήδη εγκατεστημένες στο δικτυακό λειτουργικό σύστημα. Το NOS προσφέρει αυτοματοποιημένα μία σφαιρική όψη του δικτύου η οποία περιέχει και την δικτυακή τοπολογία. Στηριζόμενοι σε αυτή την τοπολογία εκτελούνται οι αλγόριθμοι διαχείρισης και ελέγχου.

Πίνακας 2.1: Συγκεντρωτικός πίνακας σύγκρισης των ελεγκτών OpenFlow

	NOX	**POX**	**Ryu**	**FloodLight**	**ODL**
Γλώσσα Προγραμματισμού	C++	Python	Python	Java	Java
Επίδοση	Υψηλή	Χαμηλή	Χαμηλή	Υψηλή	Υψηλή
Κατανεμημένη	Όχι	Όχι	Ναι	Όχι	Ναι
Έκδοση OpenFlow	1.0	1.0	1.0, 1.2-1.4	1.0, 1.3	1.0, 1.3
Υποστήριξη Cloud	Όχι	Όχι	Ναι	Ναι	Ναι
Δυσκολία εκμάθησης	Μέτρια	Εύκολη	Μέτρια	Υπερβολική	Υπερβολική
Πλατφόρμα ανάπτυξης	Linux	Linux	Linux	Win/Mac/Linux	Win/Mac/Linux
Κατηγορία	Single controller	Single controller	Single controller	Logically centralized	Logically centralized

- *POX*: αποτελεί την φυσική εξέλιξη μιας OpenFlow διεπαφής η οποία αναπτύχθηκε σε γλώσσα Python. Εκτός από την υλοποίηση του OpenFlow σε Python, προσφέρει και άλλες εφαρμογές όπως ο NOX, μπορεί να εκτελεστεί σε όλα τα λειτουργικά συστήματα καθώς επίσης έχει ενσωματωμένα μία σειρά από γραφικά εργαλεία. Έχει κερδίσει πολύ μεγάλο έδαφος τόσο στην εκπαίδευση όσο και στην έρευνα, αναπτύσσοντας εφαρμογές και τεχνικές πάνω στην σχεδίαση SDN τεχνολογιών.
- *Ryu*: βασίζεται σε γλώσσα Python και παρέχει ένα μεγάλο σύνολο υπηρεσιών δικτύου μέσω μιας καθορισμένης διεπαφής προγραμματισμού εφαρμογών, κάνοντας πιο εύκολη τόσο την ανάπτυξη νέων εφαρμογών διαχείρισης όσο και ελέγχου δικτύων για πολλαπλές συσκευές δικτύου. Ένα σημαντικό

χαρακτηριστικό του Ryu είναι η συγχώνευση με το σύστημα διαχείρισης OpenStack cloud, επιτρέποντας έτσι μεγάλες αναπτύξεις σε κέντρα δεδομένων cloud. Παρότι ο ελεγκτής αναπτύχθηκε σε γλώσσα Python, η εκμάθηση του είναι μέτριας ως υψηλής δυσκολίας, καθώς παρέχει ένα μεγάλο σύνολο στοιχείων και διεπαφών υπηρεσιών.

- *FloodLight*: είναι μια υλοποίηση OpenFlow σε Java. Ο συγκεκριμένος controller προκύπτει από τον ελεγκτή Beacon, όπου αποτελείται από μία συλλογή λειτουργικών μονάδων με έτοιμα υποπρογράμματα διαφορετικών λειτουργιών δικτύου, το οποίο το καθιστά πιο εύκολο στην κατανόηση για έναν νέο προγραμματιστή.
- *OpenDayLight project (ODL):* αποτελεί ένα μεγάλο σύνολο από ερευνητικά έργα που σχετίζονται με τα SDN δίκτυα τα οποία αναπτύσσονται κάτω από την επίβλεψη του Linux. Στόχος του ODL δεν είναι η δημιουργία προτύπων ,αλλά εφαρμόσιμες λειτουργίες. Στο κέντρο των έργων αυτών βρίσκεται ο SDN ελεγκτής, ο οποίος λειτουργεί σε οποιοδήποτε hardware ή software που υποστηρίζει γλώσσα Java. Επίσης, περιέχει ένα σύνολο από πακέτα λογισμικού που εκτελούν διάφορες εφαρμογές τόσο στην «προς νότο», όσο και στην «προς βορρά» διεπαφή.
- *Beacon*: είναι ένας ελεγκτής που αναπτύχθηκε σε γλώσσα Java κάτι το οποίο του επιτρέπει να τρέχει σε διαφορετικές πλατφόρμες, ακόμη και σε android συσκευές. Έχει την δυνατότητα να εκτελεί, να σταματά και να εγκαθιστά άλλες εφαρμογές ακόμη και σε κάθε στιγμή όπου ο ελεγκτής λειτουργεί σε αντίθεση με τους προαναφερθείς ελεγκτές.
- *ONOS:* είναι ένα εγχείρημα ελεύθερου λογισμικού υπό την αιγίδα του συνδέσμου Linux Foundation με στόχο τη δημιουργία ενός λειτουργικού συστήματος για την δικτύωση SDN, όπου θα προσφέρει δυνατότητα δημιουργίας κλιμακούμενων εφαρμογών, υψηλές αποδόσεις και μεγάλη διαθεσιμότητα. Η πλατφόρμα του ONOS είναι επεκτάσιμη και κατανεμημένη στοχεύοντας στη δημιουργία φιλικού περιβάλλοντος για τον χρήστη. Το λογισμικό είναι γραμμένο εξ ολοκλήρου σε Java. Παρουσιάζει καλές επιδόσεις όπως και το προαναφερθέν • OpenDayLight αλλά και παρόμοιες δυσκολίες εκμάθησης.

2.6 Πλεονεκτήματα των SDN

Τα επιχειρήματα για την χρήση της αρχιτεκτονικής SDN σε σχέση με τα κοινά δίκτυα είναι ισχυρά. Τα πιο σημαντικά από αυτά είναι:

- *Κεντρικοποιημένη διαχείριση (Centralized Operational Control)*: αυτό είναι το επίκεντρο ολόκληρης της προσέγγισης. Με άλλα λόγια η διαχείριση όλου του δικτύου στηρίζεται σε ένα κεντρικό σημείο, τον ελεγκτή ο οποίος διατηρεί μία ολοκληρωμένη άποψη της τοπολογίας και γίνεται αντιληπτό στις εφαρμογές που επικοινωνεί ως ένας ενιαίος εξοπλισμός.
- *Έμμεσα προγραμματιζόμενα δίκτυα*: εύκολα πλέον μπορούν να προστεθούν νέες εφαρμογές και λειτουργίες στον ήδη υπάρχον εξοπλισμό, χρησιμοποιώντας ανοιχτού κώδικα εργαλεία και ανεξάρτητα τον κατασκευαστή του.
- *Μειωμένες δαπάνες κεφαλαίου (CAPEX)*: με το όρο CapEx (Capital Expenditures) εννοούμε τις κεφαλαιουχικές δαπάνες δηλαδή τα κεφάλαια που χρησιμοποιούνται από μια εταιρεία για την απόκτηση, αναβάθμιση και συντήρηση των φυσικών περιουσιακών στοιχείων (π.χ. εξοπλισμό κλπ.). Στην προκειμένη περίπτωση το κόστος μιας δικτυακής συσκευής υψηλών προδιαγραφών σχετίζεται άμεσα με το επίπεδο ελέγχου και το λογισμικό το οποίο διαθέτει. Σε τέτοιου είδους λειτουργίες επενδύουν οι κατασκευαστές, και βάση αυτών επενδύουν οι εταιρίες στον εξοπλισμό. Έχοντας τον διαχωρισμό τους, οι δαπάνες αυτές μεταφέρεται στον ελεγκτή έχοντας ως αποτέλεσμα να μειώνεται σε ικανοποιητικό βαθμό το κόστος του δικτυακού εξοπλισμού.
- *Αυξημένη αξιοπιστία*: η ελαχιστοποίηση της ανθρώπινης παρέμβασης στον εξοπλισμό καθώς επίσης και η δυνατότητα πολλών αυτοματοποιημένων διαδικασιών προσφέρει την αύξηση της αξιοπιστίας των δικτύων αυτών.
- *Μείωση λειτουργικού κόστους (OPEX)*: με τον όρο OPEX (Operating expenses) εννοούμε τα λειτουργικά έξοδα δηλαδή τα καθημερινά έξοδα που πραγματοποιεί μια εταιρεία για να διατηρήσει τη λειτουργία της επιχείρησής της. Εν προκειμένω, κατά αντιστοιχία αφού έχουμε μείωση του CAPEX θα έχουμε και μείωση του κόστους του ανθρώπινου δυναμικού. Οι λειτουργίες από την κάθε συσκευή χωριστά μεταφέρονται σε ένα μοναδικό σημείο δηλαδή αυτό του ελεγκτή.
- *Υψηλή διαθεσιμότητα*: η δυνατότητα τόσο της επίβλεψης όσο και της ρύθμισης της δικτυακής τοπολογίας από τον ελεγκτή αυξάνει την πρόβλεψη σφαλμάτων

και της έγκαιρης διόρθωσής τους.

- *Καινοτομία*: προσφέρεται η δυνατότητα δοκιμής και ανάπτυξης νέων υπηρεσιών χωρίς την παρέμβαση και παρεμπόδιση των ήδη ενεργών υπηρεσιών.
- *Εικονοποίηση δικτύων*: Με την μείωση της πολυπλοκότητας του εξοπλισμού και κατ' επέκταση του εξοπλισμού, γίνεται πιο εύκολη η μετάβαση σε εικονοποιημένο εξοπλισμό χωρίς μεγάλες δυνατότητες και απαιτήσεις.

2.7 Εφαρμογές της τεχνολογίας SDN

Η τεχνολογία SDN μπορεί να εφαρμοστεί σε ένα τεράστιο φάσμα δικτυακών περιβαλλόντων. Με τον διαχωρισμό των επιπέδων ελέγχου και δεδομένων, τα δίκτυα καθοριζόμενα από λογισμικό επιτρέπουν έναν ολοκληρωτικό έλεγχο όλου του δικτύου, και ο έλεγχος αυτός μπορεί να προσαρμοστεί ανάλογα με την εκάστοτε δικτυακή κατάσταση. Παρακάτω, αναφέρουμε περιληπτικά διάφορα δικτυακά περιβάλλοντα στα τα οποία έχουν προταθεί ή έχουν εφαρμοστεί λύσεις με τεχνολογία SDN [18].

- *Δίκτυα Επιχειρήσεων***:** Οι μεγάλες επιχειρήσεις χρησιμοποιούν κατά κόρον μεγάλα δίκτυα, ενώ ταυτόχρονα διαθέτουν αυστηρές απαιτήσεις τόσο για την ασφάλεια όσο και για την απόδοση τους. Επιπροσθέτως, τα διάφορα περιβάλλοντα μπορούν να έχουν διαφορετικές απαιτήσεις, ποικίλα χαρακτηριστικά και σχετικά μεγάλο πλήθος χρηστών, π.χ. τα δίκτυα των Πανεπιστημίων μπορούν να λογίζονται ως ειδικές περιπτώσεις επιχειρηματικών δικτύων. Σε αυτό το περιβάλλον, οι περισσότερες από τίς συνδέσεις είναι προσωρινές και δεν ελέγχονται αυστηρά από το Πανεπιστήμιο για την δέσμευση πόρων. Επιπλέον, τα Πανεπιστήμια παρέχουν συχνή υποστήριξη για ερευνητικά πειράματα [18]. Η επαρκής και σωστή διαχείριση έχει τεράστια ζωτική σημασία σε περιβάλλοντα π.χ. σε περίπτωση επιχειρήσεων (Enterprises), η τεχνολογία SDN δίνει την δυνατότητα να προγραμματιστεί - εφαρμοστεί ολόκληρη η στρατηγική της επιχείρησης και να υλοποιηθεί η εκάστοτε κατάλληλη-επιθυμητή πολιτική. Ακόμη, η τεχνολογία SDN μπορεί να χρησιμοποιηθεί τόσο για την παρακολούθηση της δραστηριότητας όσο και για το έλεγχο της ρύθμοαπόδοσης του δικτύου.
- *Κέντρα δεδομένων (Data Centers):* Τα κέντρα δεδομένων έχουν εξελιχθεί με ένα ξέφρενο ρυθμό ιδίως τα τελευταία χρόνια. Η πολύ προσεκτική διαχείριση της κυκλοφορίας καθώς και η πολιτική που πρέπει να ακολουθηθεί είναι κρίσιμες

ενέργειες σε τέτοιες μεγάλες εγκαταστάσεις. Ειδικότερα αν υπάρξει μια διακοπή υπηρεσίας ή κάποια πρόσθετη καθυστέρηση πακέτων είναι πιθανό αυτό να οδηγήσει σε μία όχι και τόσο καλή παραγωγικότητα. Τα κέντρα δεδομένων είναι κατάλληλα εφοδιασμένα ώστε να αντιμετωπίζουν συνεχείς και υψηλές απαιτήσεις. Συνήθως όμως ρυθμίζονται να εξυπηρετούν, με χαμηλότερους ρυθμούς δεδομένων, αλλά πάντα είναι έτοιμα να εξυπηρετήσουν ταχύτατα ένα υψηλότερο φόρτο εργασίας το οποίο αιφνιδίως μπορεί να προκύψει [18].

- *Οπτικά δίκτυα*: Η κίνηση των δεδομένων και η διαχείριση της στα δίκτυα SDN πραγματοποιείται κατά ροές (flows), έτσι επιτρέπουν την υποστήριξη και την ενσωμάτωση πολλών και διαφορετικών από τεχνολογικής πλευρά δικτύων. Έχουν γίνει αρκετές προσπάθειες και έχουν γίνει προτάσεις για τον έλεγχο των δικτύων *μεταγωγής κυκλωμάτων* και δικτύων *μεταγωγής πακέτων* με την χρήση του πρωτοκόλλου OpenFlow. Μία τέτοια αρχιτεκτονική μεταγωγής πακέτων βασίζεται σε Wavelegth Selective Switching (WSS) κάνοντας χρήση του πρωτοκόλλου OpenFlow. Ένα οπτικό SDN (SDON) αναπτύσσει ένα ενιαίο πρωτόκολλο ποιότητας υπηρεσιών (Quality of Services-QOS) για την διακοπή λειτουργίας [18].
- *Δίκτυα Backbone*: Στην περίπτωση των δικτύων backbone μεγάλης κλίμακας μπορεί να γίνει χρήση της αρχιτεκτονικής SDN για να επιτευχθεί προγραμματισιμότητα και υψηλή διαθεσιμότητα στο δίκτυο. Σε γενικές γραμμές ακολουθώντας κεντρικοποιήμενο έλεγχο τα αποτελέσματα είναι πέραν του επιθυμητού και βλέπουμε να αποδίδουν καρπούς, όπως π.χ. η καλύτερη αξιοποίηση του δικτύου, η προγραμματιζόμενη κατανομή πόρων κ.α. Επιπρόσθετα, αυτή η προσέγγιση διευκολύνει την δοκιμή του δικτύου, αφού ο κεντρικοποιημένος έλεγχος μπορεί να χρησιμοποιεί την παραγωγικότητα του δικτύου για έρευνα και δοκιμή νέων ιδεών και εφαρμογών [18].
- *Εφαρμογή σε σπίτια και μικρές επιχειρήσεις*: Η δικτύωση των σπιτιών και των μικρών επιχειρήσεων αυξάνεται ολοένα και περισσότερο, έτσι αυξάνεται η γενικότερη δικτυακή πολυπλοκότητα, και αυτό μας εξωθεί στην ανάγκη να γίνει μια πιο προσεγμένη διαχείριση του δικτύου και να αυξηθεί η ασφάλεια. Τα ευάλωτα δίκτυα ως προς την ασφάλεια ενδέχεται να αποτελούν στόχο κακόβουλων λογισμικών, ενώ κάποιες διακοπές που οφείλονται σε προβλήματα ρύθμισης του δικτύου εγκυμονούν κίνδυνο για τις επιχειρήσεις. Η λύση σε αυτό το πρόβλημα έρχεται να δοθεί από την SDN τεχνολογία μιας και δίνεται η δυνατότητα μιας

εξωτερικής ανάθεσης διαχείρισης αυτών των δικτύων σε «τρίτους» μέσω του τηλεχειρισμού των εμπλεκομένων switches καθώς και την εφαρμογή διαφόρων κατανεμημένων αλγορίθμων παρακολούθησης του δικτύου μέσω των οποίων εξάγονται χρήσιμα συμπεράσματα για την ανίχνευση πιθανών προβλημάτων ασφαλείας [18].

3. Το πρωτόκολλο OpenFlow

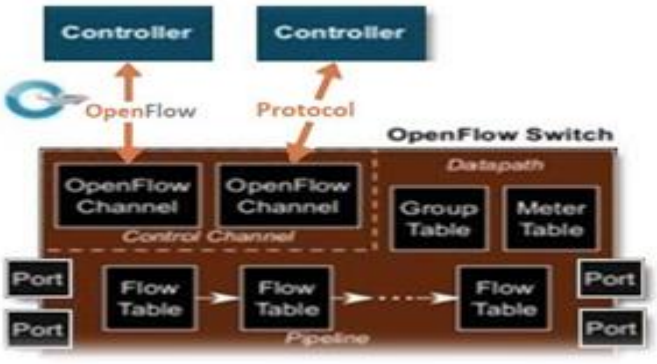
3.1 Εισαγωγή

Πολλοί από τους ερευνητές που εργάζονται στη συγκεκριμένη περιοχή θεωρούν το πρωτόκολλο OpenFlow [34][35] ως τον βασικό παράγοντα ενεργοποίησης της δικτύωσης SDN. Το πρωτόκολλο OpenFlow υλοποιείται στην "*νότια διεπαφή"* ενός ελεγκτή SDN η οποία επιτρέπει την επικοινωνία μεταξύ του ελεγκτή και των μεταγωγεών. Το πρωτόκολλο OpenFlow επιτρέπει στους ελεγκτές δικτύου να καθορίσουν τη διαδρομή των πακέτων μέσα σε ένα πλέγμα από μεταγωγείς. Επίσης, υλοποιεί την απομακρυσμένη διαχείριση των πινάκων προώθησης πακέτων του μεταγωγέα, προσθέτοντας, τροποποιώντας και αφαιρώντας κανόνες και ενέργειες αντιστοίχισης πακέτων. Ωστόσο δεν αποτελεί το μοναδικό πρωτόκολλο επικοινωνίας για τη δικτύωση SDN.

3.2 Τα δομικά μέρη του OpenFlow

Βασικά δομικά μέρη του προτύπου OpenFlow είναι:

- Ο ελεγκτής OpenFlow (OpenFlow controller)
- Ο μεταγωγέας OpenFlow (OpenFlow Switch) όπως φαίνεται και στην Εικόνα 3.1. Το OpenFlow Switch περιλαμβάνει:
  a) τις πόρτες (port)
  b) το ασφαλές κανάλι OpenFlow
  c) τους πίνακες ροών (Flow Tables)



Εικόνα 3.1:Δομή OpenFlow

3.2.1 OpenFlow controller

Ένας ελεγκτής SDN είναι το στρατηγικό σημείο της δικτύωσης που καθορίζεται από λογισμικό. Ένας OpenFlow controller χρησιμοποιεί το πρωτόκολλο OpenFlow για τη σύνδεση και τη διαμόρφωση των διαφόρων συσκευών δικτύου (δρομολογητές, μεταγωγείς κ.λπ.) για να προσδιορίσει την καλύτερη διαδρομή για τις ροές δεδομένων των διαφόρων εφαρμογών και των μηνυμάτων. Στο προηγούμενο κεφάλαιο παρουσιάστηκαν οι πιο δημοφιλείς ελεγκτές στην καθοριζόμενη από λογισμικό δικτύωση.

Πίνακας 3.1: Τα πεδία μιας εγγραφής του Πίνακα Ροής ενός OpenFlow switch

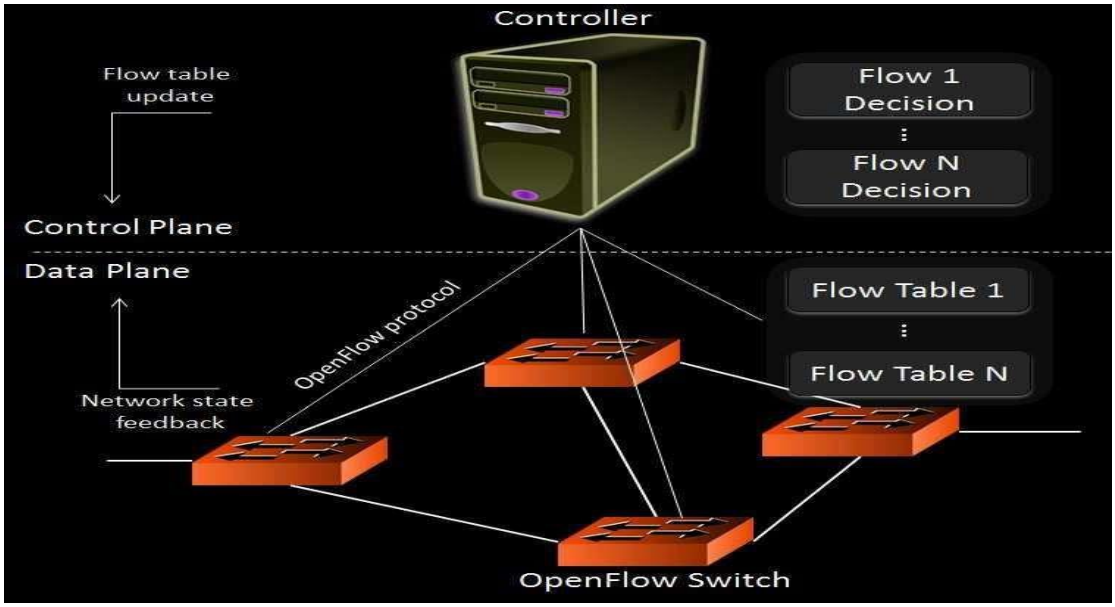
Match Fields	Priority	Counters	Actions	Timeouts	Cookie

3.2.2 OpenFlow Switch

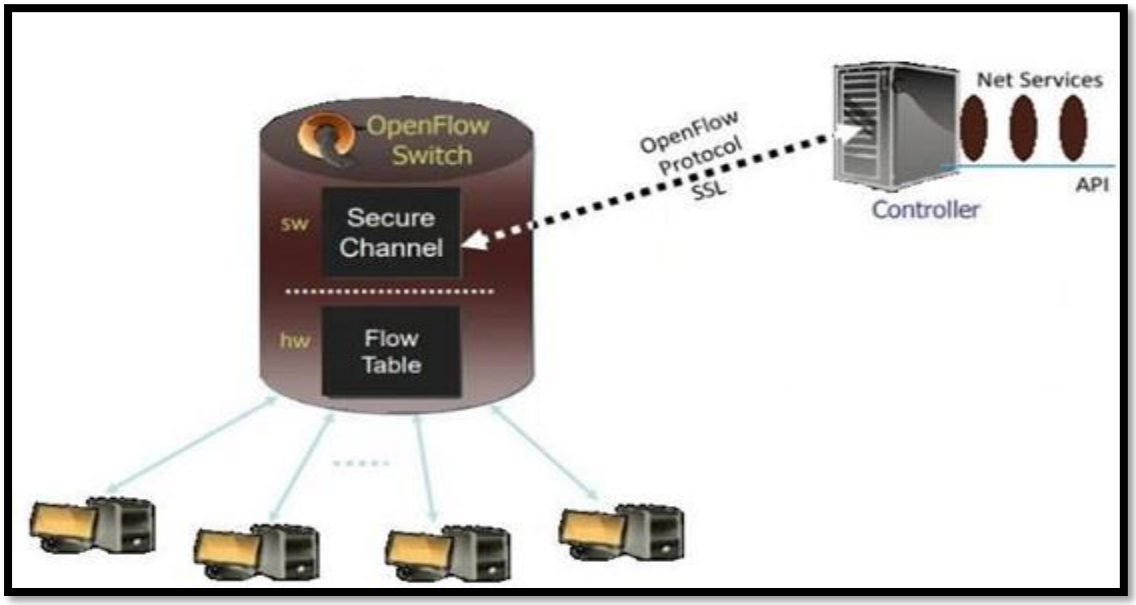
Είναι ένας μεταγωγέας δεδομένων που επικοινωνεί μέσω του ασφαλούς καναλιού OpenFlow με έναν εξωτερικό ελεγκτή. Εκτελεί αναζήτηση πακέτων και προώθηση σύμφωνα με έναν ή περισσότερους πίνακες ροής και έναν πίνακα δρομολόγησης. Το OpenFlow Switch επικοινωνεί με τον ελεγκτή και ο ελεγκτής διαχειρίζεται τον μεταγωγέα μέσω του πρωτοκόλλου OpenFlow. Τα OpenFlow Switches βασίζονται είτε αποκλειστικά στο πρωτόκολλο OpenFlow είτε σε άλλα πρωτόκολλα τα οποία είναι συμβατά με αυτό. Επομένως αυτοί οι μεταγωγείς υποστηρίζουν το OpenFlow πρωτόκολλο και μπορούν να λειτουργήσουν μόνο με τη συνεργασία των τριών βασικών στοιχείων τα οποία είναι:

- *οι πίνακες ροής*, οι οποίοι βρίσκονται στους μεταγωγείς και κάθε εγγραφή τους περιλαμβάνει μια σειρά από πεδία, όπως φαίνεται και στον Πίνακα 3.1,
- *ο ελεγκτής*, ο οποίος επικοινωνεί με τους μεταγωγείς μέσω του πρωτοκόλλου Openflow και μπορεί να επιβάλλει διάφορες πολιτικές στις ροές δεδομένων
- *Flow Rule*, είναι ένα ζεύγος κανόνων αντιστοίχισης/δράσης (Match-Action), που σημαίνει ότι κάθε εισερχόμενο πακέτο αντιστοιχίζεται σε μια εγγραφή του πίνακα ροής, και εκτελεί το σύνολο των ενεργειών, οι οποίες σχετίζονται με αυτόν τον κανόνα.

Η Εικόνα 3.2 δείχνει ένα *OpenFlow controller* ο οποίος ελέγχει διάφορα *Switches* ενός δικτύου. Πιο αναλυτική κατάσταση φαίνεται στη Εικόνα 3.3, όπου φαίνεται και το ασφαλές κανάλι επικοινωνίας μεταξύ ενός switch και ενός controller.



Εικόνα 3.2: OpenFlow controller and OpenFlow Switch



Εικόνα 3.3: OpenFlow controller and OpenFlow Switch

3.3 Εύρεση τοπολογίας δικτύου

Εξαιτίας της δυναμικής αρχιτεκτονικής και της προγραμματιζόμενης λειτουργικότητας, η δικτύωση SDN για να παρέχει υπηρεσίες δικτύωσης σε εφαρμογές διαφορετικών περιοχών χρειάζεται να γνωρίζει την τοπολογία ολόκληρου του δικτύου. Για το λόγο αυτό ο ελεγκτής OpenFlow συλλέγει πληροφορίες σχετικά με την τοπολογία ολόκληρου του δικτύου εφαρμόζοντας *πρωτόκολλα ανακάλυψης της*

τοπολογίας. Η διαδικασία εύρεσης τοπολογίας στα δίκτυα ελεγχόμενα από το πρωτόκολλο OpenFlow είναι πολύ σημαντική, καθώς η γνώση της επιτρέπει την πιο αποτελεσματική δρομολόγηση των πακέτων από τον ελεγκτή.

Οι περισσότεροι ελεγκτές (π.χ. NOX, POX, Beacon, OpenDaylight, Floodlight, Ryu, Trema, ONOS, Juniper Contrail κλπ.) χρησιμοποιούν το OpenFlow Topology Discovery Protocol (OFDP) για την ανακάλυψη της τοπολογίας του δικτύου [1][36].

Η ανακάλυψη της τοπολογίας βασίζεται:

- στον εντοπισμό του κόμβου
- στον εντοπισμό των συνδέσεων

Ο εντοπισμός του κόμβου γίνεται με τη βοήθεια του πρωτοκόλλου OpenFlow ως ακολούθως. Κατά την εκκίνηση της διαδικασίας, ένα OpenFlow Switch δημιουργεί μια συνεδρία με τον ελεγκτή SDN. Κατά την έναρξη της περιόδου λειτουργίας, ο μεταγωγέας μεταδίδει στον ελεγκτή πληροφορίες σχετικά με την *ταυτότητά* του (Datapath ID), τα *χαρακτηριστικά* του (π.χ. χωρητικότητα των πινάκων ροής), τις *ενεργές θύρες* του και τις *λειτουργίες* που υποστηρίζονται μέσω της ανταλλαγής μηνυμάτων "Features Request/Reply" (Αίτηση / Απάντηση). Η διαδικασία αυτή βοηθά στον εντοπισμό του κόμβου.

Για τον εντοπισμό των συνδέσεων χρησιμοποιείται το πρωτόκολλο OFDP. Το πρωτόκολλο OFDP βασίζεται στο προ υπάρχον πρωτόκολλο (Link Layer Discovery Protocol) LLDP και αποτελεί μια ελαφρώς τροποποιημένη εκδοχή του.

3.3.1 Το πρωτόκολλο LLDP

Το LLDP [8] μπορεί να χρησιμοποιηθεί από μία δικτυακή συσκευή συνδεδεμένη σε ένα LAN για να "κοινοποιήσει" την ταυτότητα της συσκευής καθώς και τις δυνατότητές της, αλλά και για να λάβει αντίστοιχες πληροφορίες από τις γειτονικές δικτυακές συσκευές. Συνήθως ο μηχανισμός λήψης αυτών των πληροφοριών είναι το πρωτόκολλο SNMP. Το πρωτόκολλο LLDP είναι ένα ανεξάρτητο πρωτόκολλο 2ου επιπέδου, το οποίο λειτουργεί ανεξαρτήτως του "μέσου" που χρησιμοποιείται και σκοπό έχει να:

- Να "κοινοποιήσει" τις πληροφορίες και τις δυνατότητες της συσκευής σε άλλες συσκευές που βρίσκονται στον ίδιο τομέα (domain) διαχείρισης
- να "ανακαλύψει" αντίστοιχες πληροφορίες για τις συσκευές που συνδέονται

απ' ευθείας.

Κάθε συσκευή, στην οποία έχει ενεργοποιηθεί το πρωτόκολλο LLDP, στέλνει περιοδικά μηνύματα στη multicast MAC address (01:23:00:00:00:01). Το LLDP είναι ένα πρωτόκολλο λειτουργίας "one hop". Ο κάθε μεταγωγέας, όταν λάβει LLDP μηνύματα, χτίζει έναν πίνακα με πληροφορίες για όλες τις απευθείας συνδεδεμένες γειτονικές συσκευές.

Από το LLDP μπορούμε να συλλέξουμε τις ακόλουθες πληροφορίες [8]:

- Όνομα συστήματος
- Όνομα θύρας
- Όνομα εικονικού δικτύου
- Διεύθυνση υλικού

Οι πληροφορίες LLDP [19] μπορούν να σταλούν και να ληφθούν μόνο από συσκευές που συνδέονται απευθείας μεταξύ τους μέσω του ίδιου συνδέσμου. Έτσι, οι διάφορες συσκευές δικτύου ανακαλύπτουν την κατάσταση των γειτόνων τους καθώς συνεχώς μεταδίδουν και "ακούνε" μηνύματα LLDP σε κάθε σύνδεση, ανακαλύπτοντας πότε προστίθεται ή αφαιρείται μια νέα συσκευή. Με αυτόν τον τρόπο, κάθε συσκευή δικτύου μπορεί να διατηρεί ενημερωμένες τις πληροφορίες τοπολογίας του τοπικού δικτύου. Οι δικτυακές συσκευές αναγνωρίζουν τους γείτονες τους σύμφωνα με τις πληροφορίες που λαμβάνουν με μηνύματα LLDP (όπως π.χ. είναι η διεύθυνση MAC). Αυτές οι πληροφορίες αποστέλλονται σε πακέτα που ονομάζονται LLDP Data Units (LLDPDUs). Τα δεδομένα που αποστέλλονται και λαμβάνονται μέσω LLDPDUs είναι χρήσιμα καθώς οι συσκευές δικτύου ανακαλύπτουν:

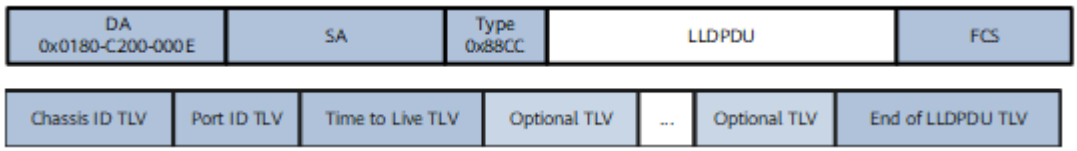
i) τους *γειτονικούς κόμβους* κάθε συσκευής και

ii) μέσω των *θυρών* (*ports*) που συνδέονται μεταξύ τους. Το LLDP ορίζει τα ακόλουθα:

   - Ένα σύνολο κοινών μηνυμάτων (τιμές μήκους, τύπου),
   - Ένα πρωτόκολλο για τη μετάδοση και τη λήψη μηνυμάτων,
   - Μια μέθοδο για την αποθήκευση των πληροφοριών που περιέχονται στις ενημερώσεις που έλαβε. Το LLDP είναι μονοκατευθυντικό πρωτόκολλο, δηλαδή λειτουργεί μόνο με λειτουργία 'κοινοποίησης'. Επομένως, το LLDP δεν ζητά πληροφορίες ούτε παρακολουθεί τις αλλαγές κατάστασης μεταξύ συσκευών δικτύου LLDP.

Οι πληροφορίες LLDP αποστέλλονται από τους ελεγκτές σε όλες τις ενεργές διεπαφές του μεταγωγέα σε σταθερό χρονικό διάστημα, με τη μορφή πλαισίου Ethernet. Οι Πίνακες 3.1 και 3.2 δείχνουν τη μορφή πλαισίου Ethernet που χρησιμοποιείται από το LLDP πρωτόκολλο. Τα πεδία DA και SA αντιπροσωπεύουν τη διεύθυνση *MAC προορισμού (*συνήθως μια διεύθυνση πολλαπλής διανομής) και τη διεύθυνση *MAC προέλευσης* αντίστοιχα. Το πεδίο τύπου ethernet έχει οριστεί σε 0x88CC. Κάθε πλαίσιο περιέχει μία μονάδα δεδομένων πρωτοκόλλου LLDP (LLDPDU).

Κάθε LLDPDU φέρει επικεφαλίδα και ένα μεταβλητό αριθμό από πεδία πληροφορίας γνωστά ως TLVs. Το ωφέλιμο φορτίο ενός πλαισίου LLDP είναι μια ακολουθία πεδίων για τον τύπο (T), το μήκους (L) και την τιμή (V) της πραγματικής πληροφορίας (TLV).και περιγράφεται στον Πίνακα 3.2. Κάθε πλαίσιο LLDP περιλαμβάνει πάντα τα ακόλουθα αναπόσπαστα υποχρεωτικά TLVs: chassis ID, port ID και time-to-live. Τα υποχρεωτικά TLV ακολουθούνται από οποιονδήποτε αριθμό προαιρετικών TLV. Το πλαίσιο τελειώνει με ένα ειδικό TLV, που ονομάζεται άκρο του LLDPDU στο οποίο τόσο τα πεδία τύπου όσο και τα μήκη είναι 0. Ο Πίνακας 3.2 δείχνει τη δομή ενός LLDPTLV, στην οποία η πραγματική τιμή μπορεί να είναι 0 έως 511 οκτάδες.



Εικόνα 3.4: Δομή LLDP και δομή LLDPDU

Πίνακας 3.1: Περιγραφή δομής ενός LLDP πακέτου

Field	Value	Length (Byte)
DA	LLDP multicast address	6
SA	MAC address	6
Ether type	0x88cc	2
Data pad	LLDPDU	1500
FCS	Check digit	4

Πίνακας 3.2: Περιγραφή δομής ενός LLDPDU πακέτου

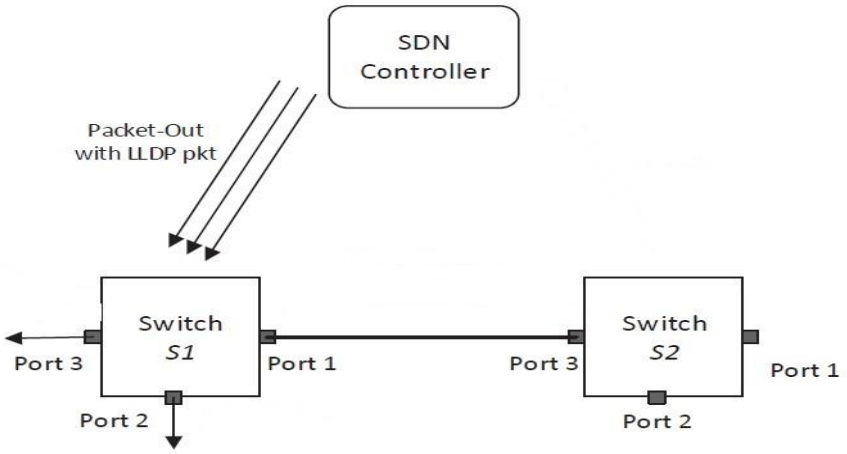
Item	Length (Byte)
TLV type	7
TLV information string length	9
TLV actual value of information	0-511

3.3.2 Το πρωτόκολλο OFDP

Η μεθοδολογία που πρέπει να ακολουθηθεί κατά την εκτέλεση του αλγορίθμου OFDP [9] για την εύρεση νέου κόμβου θα εξηγηθεί παρακάτω αφού προηγηθεί μία εισαγωγή της διαδικασίας. Αρχικά ο ελεγκτής SDN στέλνει ένα μήνυμα *Packet-Out* σε κάθε μεταγωγέα. Ο κάθε μεταγωγέας, αν λάβει το μήνυμα Packet-Out, τότε, προωθεί τα πακέτα LLDP προς όλους τους μεταγωγείς με τους οποίους είναι απευθείας συνδεδεμένος (τους γειτονικούς του μεταγωγείς) από την προκαθορισμένη θύρα τους. Ο κάθε μεταγωγέας που θα λάβει το πακέτο LLDP, στέλνει ένα *Packet-In* στον ελεγκτή για να τον ενημερώσει με πληροφορίες σχετικά με την σύνδεση με τον γείτονα του. Οι πληροφορίες περιλαμβάνουν τις *MAC διευθύνσεις* στα δύο άκρα της σύνδεσης (link) και τα αντίστοιχα *ports*.

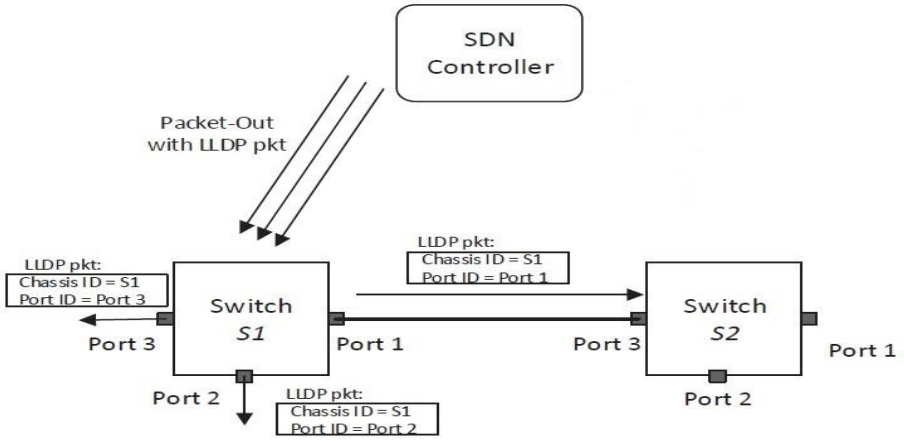
Από προεπιλογή, μετά τη λήψη ενός πακέτου LLDP η οποία θα γίνει από διάφορες θύρες -και όχι από τη θύρα του ελεγκτή-, ο κάθε ενεργός μεταγωγέας πρέπει να στείλει ένα μήνυμα πακέτου που περιέχει το ληφθέν LLDP στον ελεγκτή. Τόσο τα Packet-In μηνύματα όσο και τα μηνύματα Packet-Out είναι απαραίτητα στους μηχανισμούς ανακάλυψης τοπολογίας. Όλη η παραπάνω διαδικασία που περιεγράφηκε, γίνεται εύκολα κατανοητή στις Εικόνες 3.4 - 3.6 όπου παρουσιάζονται τα βήματα του αλγορίθμου OFDP για την εύρεση νέου κόμβου. Συγκεκριμένα, η διαδικασία ανακάλυψης νέου κόμβου έχει ως ακολούθως:

1. Το στοιχείο ελέγχου στέλνει τα μηνύματα Packet-Out με ενθυλακωμένα τα LLDP πακέτα στον μεταγωγέα (Εικόνα 3.4).



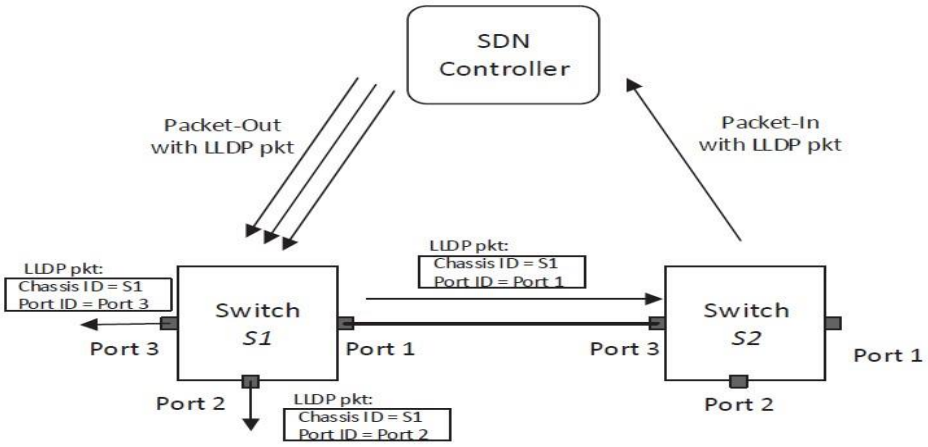
Εικόνα 3.4: Αποστολή Packet-Out από τον ελεγκτή στο μεταγωγέα

2. Με τη λήψη των μηνυμάτων Packet-Out οι μεταγωγείς προωθούν τα πακέτα LLDP σε όλες τις ενεργές θύρες (Εικόνα 3.5).



Εικόνα 3.5: Αποστολή LLDP πακέτων μεταξύ των μεταγωγέων

3. Μόλις ένας γειτονικός κόμβος OpenFlow, λάβει το πακέτο LLDP, στέλνει ένα Packet-In στον ελεγκτή για ενημέρωση σχετικά με τον γείτονα (Εικόνα 3.6).



Εικόνα 3.6: Αποστολή Packet-In από το μεταγωγέα στον ελεγκτή

Η διαδικασία εφαρμόζεται σε κάθε switch του δικτύου.

3.4 Αξιολόγηση πρωτοκόλλου

Η διαδικασία ανακάλυψης της τοπολογίας είναι μια υπηρεσία που εκτελείται συνεχώς στο παρασκήνιο σε όλους του ελεγκτές των δικτύων SDN. Αυτή η συνεχής εκτέλεση μπορεί να επιβαρύνει τον ελεγκτή λόγω των μηνυμάτων που παράγει η διαδικασία

ανίχνευσης της τοπολογίας. Επίσης, είναι σημαντικό να γνωρίζουμε πόσος χρόνος απαιτείται για τον εντοπισμό της τοπολογίας. Και οι δύο αυτές παράμετροι είναι κρίσιμοι για την απόδοση του πρωτοκόλλου.

3.4.1 Overhead σε μηνύματα

Από την εφαρμογή του πρωτοκόλλου OFDP προκύπτει ότι ο αριθμός των Packet-Out πακέτων που πρέπει να στείλει ένας ελεγκτής OpenFlow είναι ίσος με τον συνολικό αριθμό θυρών στο δίκτυο. Τα συνολικά πακέτα τύπου Packet-In που λαμβάνει τελικά ο ελεγκτής SDN είναι διπλάσιος από τον αριθμό των ενεργών συνδέσμων (links) στο δίκτυο, καθώς υπάρχει ένα πακέτο για κάθε κατεύθυνση διάσχισης του συνδέσμου.

$$PIN_{OFDP} = 2L$$

$$POUT_{OFDP} = \sum_{n=1}^{N} pi$$

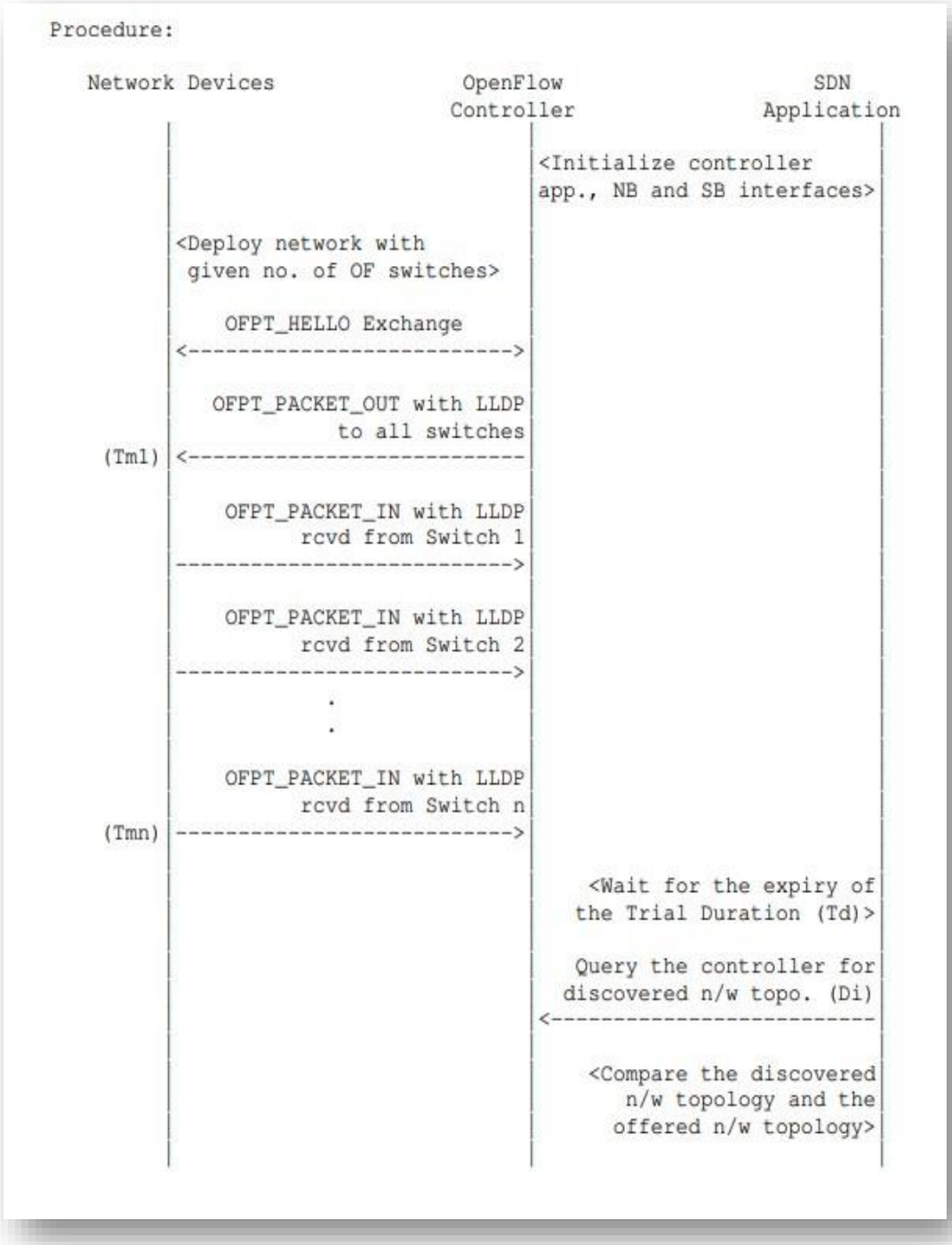
3.4.2 Χρόνος ανακάλυψης τοπολογίας δικτύου (Network Topology Discovery Time)

Ο *χρόνος ανακάλυψης τοπολογίας δικτύου* [29] ορίζεται ως ο χρόνος που μεσολαβεί από την χρονική στιγμή που θα σταλεί το πρώτο Packet-Out μήνυμα έως ότου ληφθεί και το τελευταίο Packet-In από τον ελεγκτή.

Η μέτρηση του γίνεται υπολογίζοντας τη διαφορά ώρας μεταξύ της αποστολής του πρώτου μηνύματος Packet-Out, που μεταδίδεται την χρονική στιγμή (Tm_1) από τον ελεγκτή και της ώρας που λαμβάνεται από τον ελεγκτή το Packet-In, τη χρονική στιγμή (Tm_n). Στην εικόνα 3.8 φαίνεται η διαδικασία που θα πρέπει να ακολουθηθεί για να υπολογιστεί ο χρόνος αυτός.

Όπως προκύπτει από την θεωρία και φαίνεται στην Εικόνα 3.8 το Topology Discovery Time (DTI) [29] προκύπτει από τον εξής τύπο:

$$Topology\ Discovery\ Time(DTI) = \mathrm{Tm_n} - \mathrm{Tm_1}$$



Εικόνα 3.8: Διαδικασία εύρεσης του Topology Discovery Time

4. Πρωτόκολλα ανακάλυψης της τοπολογίας και σύγκριση με το OFDP

Το βασικό πρωτόκολλο ανακάλυψης της τοπολογίας σε όλους τους ελεγκτές είναι το OFDP. Όμως, το OFDP έχει αρκετούς περιορισμούς. Οι περιορισμοί αυτοί είναι οι εξής:

α) το πρωτόκολλο OFDP *δεν είναι ασφαλές*, δηλαδή χρησιμοποιεί καθαρά, μη αυθεντικά πακέτα LLDP για την ανακάλυψη συνδέσμων μεταξύ των διαφόρων μεταγωγέων, πράγμα που το καθιστούν ευάλωτο σε διάφορες επιθέσεις (π.χ. Switch spoofing, Link Fabrication, Controller fingerprinting, LLDP Flood).

β) Το OFDP *δεν είναι αρκούντως αποδοτικό*, γιατί ο ελεγκτής στέλνει περιοδικά πολλά πακέτα σε κάθε μεταγωγέα στο δίκτυο, κάτι που θα μπορούσε να οδηγήσει σε μείωση της απόδοσης του επιπέδου δεδομένων. Τα πειράματα που έχουν γίνει σε διάφορους ελεγκτές [37] δείχνουν ότι όταν το μέγεθος του δικτύου (δηλαδή ο αριθμός των μεταγωγέων) υπερβαίνει κάποιο όριο, η εκτέλεση της διαδικασίας εντοπισμού από μόνη της οδηγεί σε σημαντική αύξηση της χρήσης CPU του ελεγκτή και σημαντική μείωση της απόδοσης του δικτύου.

γ) Άλλα ζητήματα όπως, το OFDP *ενδέχεται να μην λειτουργεί αξιόπιστα* για συνδέσμους που έχουν φορτωθεί πολύ επειδή τα πακέτα εντοπισμού ενδέχεται να πέσουν ή να καθυστερήσουν. Επιπλέον, όταν χρησιμοποιείτε OFDP σε δίκτυο SDN πολλαπλών ελεγκτών (π.χ. εκτέλεση πολλών ελεγκτών επισκέπτη μέσω FlowVisor), το κόστος ανακάλυψης αυξάνεται γραμμικά καθώς προστίθενται περισσότεροι ελεγκτές.

Από τα παραπάνω μειονεκτήματα του OFDP, μεγάλη σημασία έχει το (β), δηλαδή ότι ο Controller στέλνει μηνύματα Packet-Out όσες είναι και οι συνολικές θύρες των μεταγωγέων. Τα παραπάνω προβλήματα οδήγησαν τους ερευνητές στη σχεδίαση νέων πρωτοκόλλων που θα μετριάζουν αυτά τα προβλήματα. Στην ενότητα αυτή παρουσιάζονται τα πρωτόκολλα εντοπισμού της τοπολογίας ενός δικτύου SDN, που έχουν προταθεί μέχρι σήμερα.

4.1 Το πρωτόκολλο OFDPv2

Ο Pakzad κ.ά. [19] προτείνουν απλές αλλά πρακτικές τροποποιήσεις στο πρωτόκολλο OFDP, οι οποίες επιτυγχάνουν σημαντική βελτίωση στην απόδοση του. Οι συγγραφείς ονόμασαν το νέο πρωτόκολλο OFDPv2.

Ο στόχος του OFDPv2 είναι να μειώσει το φόρτο του μηχανισμού ανακάλυψης τοπολογίας μειώνοντας τον αριθμό των μηνυμάτων ελέγχου που πρέπει να σταλούν από τον ελεγκτή. Η βασική ιδέα είναι απλή. Αντί ο ελεγκτής να δημιουργεί ένα πακέτο LLDP για κάθε θύρα για κάθε μεταγωγέα και να το στέλνει στον αντίστοιχο μεταγωγέα μέσω ενός ξεχωριστού μηνύματος Packet-Out, όπως συμβαίνει στο πρωτόκολλο OFDP, δημιουργεί και στέλνει μόνο ένα πακέτο LLDP (Packet-Out) σε κάθε μεταγωγέα. Με αυτή την τροποποίηση μειώνεται ο αριθμός των Packet-Out σε ένα ανά μεταγωγέα. Ο αλγόριθμος παρέχει περαιτέρω οδηγίες στο μεταγωγέα για το πως θα γίνει η προώθηση του πακέτου LLDP σε καθεμιά από τις θύρες του, τις οποίες προσδιορίζει, από το ένα μοναδικό αναγνωριστικό (διεύθυνση MAC της θύρας) που επιτρέπει στον μεταγωγέα λήψης να αναγνωρίσει τη θύρα προέλευσης.

Υπάρχουν δύο απαιτήσεις που πρέπει να ικανοποιούνται για ένα τέτοιο αναγνωριστικό θύρας. Πρώτον, ο ελεγκτής πρέπει να γνωρίζει με σαφήνεια το αντίστοιχο αναγνωριστικό θύρας του μεταγωγέα. Αυτό είναι εφικτό, καθώς ο μεταγωγέας ενημερώνει τον ελεγκτή για τα διαθέσιμα ports. Δεύτερον, ο μεταγωγέας SDN πρέπει να είναι σε θέση να ξαναγράψει το πεδίο του αναγνωριστικού. Το OFDP υποστηρίζει αυτή τη λειτουργία η οποία χρησιμοποιείται για να επανεγγράψει τις επικεφαλίδες του του πακέτου LLDP για διάφορες ενημερώσεις. Η διεύθυνση MAC χρησμοποιείται ως το μοναδικό αναγνωριστικό θύρας, και επιπλέον η διεύθυνση MAC προέλευσης μπορεί να αντικατασταθεί με την διεύθυνση MAC προορισμού, πληρώντας έτσι αυτές τις δύο απαιτήσεις. Το OpenFlow υποστηρίζει την επανεγγραφή κεφαλίδων πακέτων, που χρησιμοποιούνται συνήθως για την ενημέρωση πεδίων TTL. Θα χρησιμοποιηθεί αυτός ο μηχανισμός για να ξαναγραφτεί η διεύθυνση MAC προέλευσης των εξερχόμενων πακέτων LLDP. Κατά τη στιγμή της σύνδεσης μεταξύ ενός μεταγωγέα και ελεγκτή OpenFlow, ο μεταγωγέας ενημερώνει τον ελεγκτή για τις διαθέσιμες θύρες του, τα αναγνωριστικά θύρας και τις σχετικές διευθύνσεις MAC. Ο ελεγκτής έχει επομένως μια αντιστοίχιση διευθύνσεων MAC και αναγνωριστικών θύρας για κάθε μεταγωγέα, γεγονός που καθιστά τη διεύθυνση MAC ένα έγκυρο μοναδικό αναγνωριστικό θύρας.

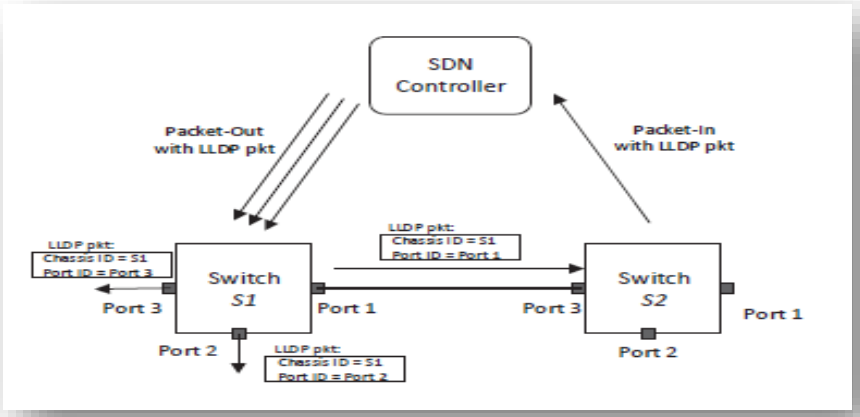
Χρησιμοποιώντας αυτούς τους βασικούς μηχανισμούς, προτείνετε μια νέα έκδοση του τρέχοντος μηχανισμού ανακάλυψης τοπολογίας SDN και το ονομάζονται OFDPv2. Η βασική ιδέα, υλοποιείται ως OFDPv2-A και OFDPv2-B.

```
for all received packets pkt do
   if pkt.etherType=LLDP and pkt.inPort=CONTROLLER then
        for all switch ports P do
           pkt.srcMAC <-- P.MACaddr
           sent copy of pkt out on port P
        end for
     end if
end for
```

Εικόνα 4.1: Αλγόριθμος 1 OFDPv2 LLDP επεξεργασίας πακέτων στους μεταγωγείς

Το OFDPv2-A περιλαμβάνει τις ακόλουθες αλλαγές στην τρέχουσα έκδοση του OpenFlowbased εντοπισμού τοπολογίας στο SDN (OFDP):

1) Εγκαθιστά ένα νέο σύνολο κανόνων σε κάθε μεταγωγέα, το οποίο καθορίζει ότι κάθε πακέτο LLDP που λαμβάνεται από τον ελεγκτή πρέπει να προωθείται σε όλες τις διαθέσιμες θύρες και ότι η διεύθυνση MAC προέλευσης του αντίστοιχου πλαισίου Ethernet πρέπει να ρυθμιστεί στη διεύθυνση μέσω του οποίου αποστέλλεται. (Εικόνα 4.1, Αλγόριθμος 1).
2) Τροποποιεί τη συμπεριφορά του ελεγκτή για να περιορίσει σε ένα τον αριθμό των μηνυμάτων Packet-Out που στέλνονται σε κάθε μεταγωγέα. Το πεδίο Port ID TLV στο ωφέλιμο φορτίο LLDP έχει οριστεί σε 0 και θα αγνοηθεί. Επίσης, ορίζεται, ότι κάθε τέτοιο μήνυμα Packet-Out, πρέπει να υποβληθεί σε επεξεργασία σύμφωνα με τους κανόνες που είναι εγκατεστημένοι στον πίνακα ροής του.
3) Τέλος, τροποποιείται το πρόγραμμα χειρισμού συμβάντων Packet-In στον ελεγκτή, ο οποίος επεξεργάζεται τα εισερχόμενα πακέτα LLDP. Αντί να αναλυθεί το Port ID TLV του ωφέλιμου φορτίου LLDP, εξετάζεται η διεύθυνση MAC του αποστολέα στην κεφαλίδα του Ethernet πλαισίου και αναζητά το αντίστοιχο αναγνωριστικό θύρας στη βάση δεδομένων του ελεγκτή.



Εικόνα 4.2: Λειτουργία του πρωτοκόλλου OFDP. Πηγή [19]

Όπως αναφέρεται παραπάνω, ο Αλγόριθμος 1 δείχνει την επεξεργασία των τροποποιημένων πακέτων LLDP σε κάθε μεταγωγέα, δηλαδή δείχνει τους εγκατεστημένους κανόνες αντιστοίχισης δράσης. Δεδομένου ότι ο ελεγκτής έχει καθορίσει τον τρόπο επεξεργασίας των πακέτων LLDP, τα πακέτα θα επεξεργαστούν σύμφωνα με αυτούς τους κανόνες, οι οποίοι περιγράφονται παρακάτω.

Όταν ένα πακέτο LLDP (EtherType = 0x88cc) φθάσει στον μεταγωγέα από τον ελεγκτή μέσω ενός μηνύματος Packet-Out (inPort = CONTROLLER), τότε ο μεταγωγές στέλνει ένα αντίγραφο του πακέτου από κάθε ενεργή θύρα του, αφού πρώτα θέσει τη διεύθυνση της θύρας ως διεύθυνση πηγής στην επικεφαλίδα του προς αποστολή μηνύματος.

Το OFDPv2-B ακολουθεί την ίδια βασική προσέγγιση του OFDPv2-A, με λίγες μόνο μικρές διαφορές. Η βασική διαφορά είναι ότι σε αυτήν την περίπτωση, δεν εγκαθιστά συγκεκριμένους κανόνες προώθησης στους μεταγωγείς. Αντ 'αυτού, ρυθμίζεται ο ελεγκτής ώστε να προωθεί μια λίστα ενεργειών για κάθε εξερχόμενο μήνυμα LLDP Packet-Out, το οποίο περιέχει οδηγίες σχετικά με τον τρόπο προώθησης του πακέτου. Η λίστα ενεργειών περιέχει ουσιαστικά τη λογική προώθησης που καθορίζεται στον Αλγόριθμο 1, χωρίς τον έλεγχο στη γραμμή 2. Το πλεονέκτημα αυτής της προσέγγισης είναι ότι δεν χρησιμοποιεί την ειδική μνήμη υψηλής ταχύτητας που συνήθως βρίσκεται στους μεταγωγείς SDN.καταναλώνει καμία από τις δαπανηρές και περιορισμένες μνήμες διευθυνσιοδότησης. Ένα άλλο όφελος, όπως ανακαλύψαμε, είναι ότι το OFDPv2-B μπορεί να χρησιμοποιηθεί σε περιπτώσεις όπου οι διακόπτες SDN δεν υποστηρίζουν την επιλογή OpenFlow OFPP TABLE, η οποία απαιτείται στο OFDPv2-A. Ωστόσο, τα οφέλη που έχει ο OFDPv2-B μετριάζονται από το κόστος του

αυξημένου αριθμού μηνυμάτων OpenFlow Packet-Out. Τα μηνύματα αυτά είναι περισσότερα σε σχέση με αυτά του OFDPv2-A

4.2 Το πρωτόκολλο sOFTDP

Το secure OpenFlow Topology Discovery Protocol (sOFTDP) [20], επιδιώκει να εξαλείψει τις σημαντικές ευπάθειες στη διαδικασία ανακάλυψης τοπολογίας και να βελτιώνει την απόδοσή του OFDP με ελάχιστες αλλαγές στη σχεδίαση του μεταγωγέα OpenFlow.

Σε ένα δυναμικό περιβάλλον δικτύωσης SDN, ο ελεγκτής πρέπει να ενημερώνεται κάθε φορά που συμβαίνει μία αλλαγή στην τοπολογία προκειμένου να ληφθούν οι κατάλληλες αποφάσεις δρομολόγησης για τη νέα τοπολογία. Οι αλλαγές τοπολογίας συμβαίνουν συνήθως ως συνέπεια δύο συμβάντων:

(i) ένας νέος σύνδεσμος προστίθεται στο δίκτυο ή

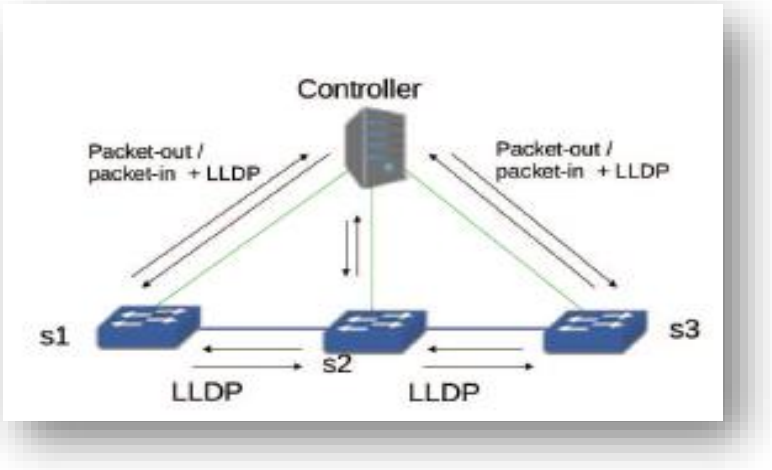
(ii) ένας υπάρχων σύνδεσμος αφαιρείται από το δίκτυο.

Και τα δύο συμβάντα είναι το αποτέλεσμα είτε της προσθήκης ενός νέου μεταγωγέα, ή της κατάργησης ενός υπάρχοντος μεταγωγέα ή της προσθήκης / αφαίρεσης ενός συνδέσμου μεταξύ δύο υπαρχόντων μεταγωγών (οι τελευταίοι περιλαμβάνουν συνδέσεις και αστοχίες μεταγωγέα). Ο σχεδιασμός του sOFTDP προϋποθέτει ότι ο ελεγκτής δεν έχει προηγούμενη γνώση της εμφάνισης τέτοιων συμβάντων και αναμένεται να αναβαθμίσει δυναμικά τον τοπολογικό του χάρτη και να προσαρμόσει ανάλογα τις αποφάσεις δρομολόγησης.

Η ‘ανακάλυψη’ τοπολογίας είναι μια κρίσιμη διαδικασία που απαιτείται να είναι:

- *Χωρίς σφάλμα*: ένα σφάλμα τοπολογίας οδηγεί σε λάθος δρομολόγηση ροών. Ο αντίκτυπος μπορεί να είναι πολύ επιβλαβής εάν το σφάλμα βρίσκεται στον πυρήνα δρομολόγησης (πυρήνες και συνδέσμους)
- *Ασφαλές*: ένα πρωτόκολλο ανακάλυψης πρέπει να είναι ασφαλές, αποτρέποντας την εισαγωγή ψεύτικων συνδέσμων και διαρροής πληροφοριών (συμπεριλαμβανομένων πληροφοριών τοπολογίας).
- *Αποτελεσματικό*: ένα πρωτόκολλο ανακάλυψης δεν πρέπει να πλημμυρίσει τον ελεγκτή με περιττές πληροφορίες και να μεταδίδει μόνο πληροφορίες για τα γεγονότα τοπολογίας όταν συμβαίνουν.

Το sOFTDP έχει σχεδιαστεί για να ικανοποιεί τις παραπάνω απαιτήσεις.
Η κύρια ιδέα είναι να μετακινηθεί ένα μέρος της διαδικασίας ‘ανακάλυψης’ από τον ελεγκτή στον μεταγωγέα. Με την εισαγωγή ελάχιστων αλλαγών στη σχεδίαση μεταγωγέα OpenFlow, το sOFTDP επιτρέπει στον μεταγωγέα να εντοπίζει αυτόνομα συμβάντα σύνδεσης και να ειδοποιεί τον ελεγκτή.



Εικόνα 4.3:Λειτουργία του πρωτοκόλλου sOFTDP πηγή [20]

Στο sOFTDP, ο ελεγκτής στέλνει πακέτα LLDP (Packet_Out) σε όλους τους συνδεδεμένους μεταγωγείς, όπως στο παραδοσιακό OFDP (εικόνα 4.3), αλλά αντί για διευθύνσεις MAC χρησιμοποιεί τιμές από μια συνάρτηση κατακερματισμού, ώστε να αποκρύψει τις πραγματικές διευθύνσεις. Μόλις συνδεθεί ένας νέος μεταγωγέας στο δίκτυο, ανταλλάσσει μηνύματα με τον ελεγκτή. Οι κόμβοι, με τους οποίους συνδέεται ο νεοεισερχόμενος κόμβος, ενημερώνουν επίσης τον ελεγκτή για τα αντίστοιχα ports που ενεργοποιήθηκαν. Ο ελεγκτής ενημερώνει αντίστοιχα την εικόνα που έχει για την τοπολογία, εκτιμώντας παράλληλα τις συντομότερες δυνατές διαδρομές. Αν ένας μεταγωγέας αποχωρήσει από το δίκτυο, οι γειτονικοί του κόμβοι ενημερώνουν τον ελεγκτή, ο οποίος και αφαιρεί τον συγκεκριμένο σύνδεσμο. Αν προστεθεί νέος σύνδεσμος μεταξύ των μεταγωγέων, οι μεταγωγείς ενημερώνουν τον ελεγκτή. Ο ελεγκτής στέλνει πακέτα LLDP στους συγκεκριμένους μεταγωγείς και μόνο για τις θύρες που ενεργοποιήθηκαν, ώστε με τις απαντήσεις που θα λάβει να ενημερώσει την τοπολογία του.

4.3 Το πρωτόκολλο Self-Healing (SHTD)

Στο άρθρο [21] παρουσιάζεται ο σχεδιασμός ενός πρωτοκόλλου "αυτοθεραπείας" (Self-Healing) για αυτόματη ανακάλυψη και συντήρηση της τοπολογίας του δικτύου σε δίκτυα.

Το πρωτόκολλο Self-Healing Topology Discovery (SHTD) ανακαλύπτει και διατηρεί μια ακριβή προβολή δικτύου που ενσωματώνει δύο χαρακτηριστικά. Έναν μηχανισμό *ανακάλυψης τοπολογίας 2ου επιπέδου* και έναν μηχανισμό *αυτόματης αποκατάστασης σφαλμάτων*. Αυτό το πρωτόκολλο μπορεί να εφαρμοστεί σε έναν τομέα με πολλούς ελεγκτές SDN μέσω ενός λογισμικού που εκτελείται σε κάθε συσκευή δικτύου. Το επίπεδο ελέγχου μπορεί να διασυνδεθεί με ένα πλήθος συσκευών δικτύου μέσω διαφορετικών μέσων μετάδοσης. Η καθολική προβολή του δικτύου παρέχεται από τους ελεγκτές SDN, μετά την εκτέλεση του πρωτοκόλλου SHTD. Σε αντίθεση με τους συμβατικούς μηχανισμούς το προτεινόμενο πρωτόκολλο χρησιμοποιεί μηνύματα 2ου επιπέδου για να δημιουργήσει ένα δέντρο ελέγχου ριζωμένο στους ελεγκτές SDNs και ανακαλύπτει την τοπολογία του δικτύου πριν από τη δημιουργία της αρχικής σύνδεσης ελέγχου. Μόλις δημιουργηθεί το δέντρο ελέγχου, κάθε συσκευή δικτύου μπορεί να δημιουργήσει ένα ασφαλές κανάλι ελέγχου στους ελεγκτές SDN.

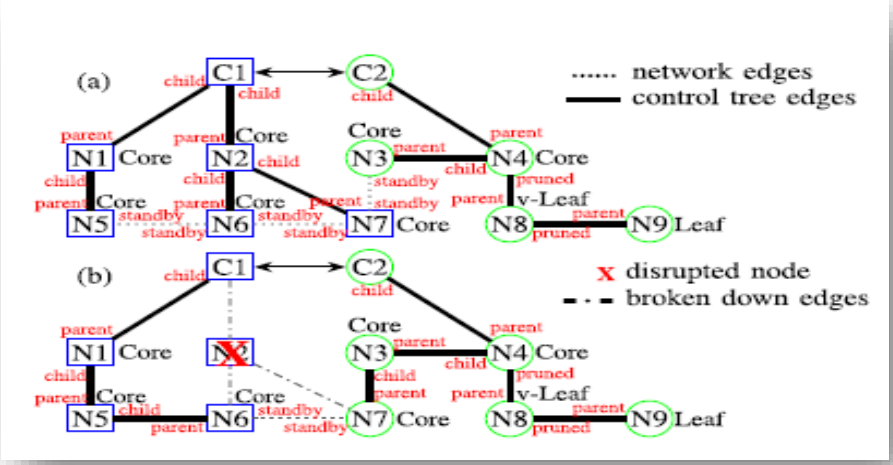
Ο αλγόριθμος στην Εικόνα 4.4 καθώς και το παράδειγμα στην Εικόνα 4.5 δείχνουν την λειτουργία του μηχανισμού, για έναν δεδομένο κόμβο v, μετά τη λήψη του μηνύματος topo_Request. Όταν ο κόμβος v λαμβάνει το topo_Request από έναν κόμβο u, στέλνει ένα μήνυμα echo_Reply "one-hop" στον κόμβο u. Αυτή η αυτόματη απάντηση επιτρέπει τη μέτρηση του χρόνου RTT σε κάθε σύνδεσμο δικτύου. Επιπλέον, περιέχει ένα bit συσχέτισης το οποίο χρησιμοποιείται από τον κόμβο v ως μήνυμα σύνδεσης για να ανακοινώσει στους γείτονές του εάν έχει ενώσει ένα από αυτά στο δέντρο ελέγχου. Εάν ο κόμβος v λαμβάνει το topoRequest από μια θύρα αναμονής, επιβεβαιώνει τη συσχέτιση στο echoReply, ορίζει τη μητρική κατάσταση στην εισερχόμενη θύρα p και προωθεί το topoRequest σε όλες τις θύρες εκτός από την εισερχόμενη θύρα. Αντίθετα, εάν το topoRequest φτάσει σε θύρα μη αναμονής, ο κόμβος v αρνείται τη συσχέτιση στο echoReply και απορρίπτει το μήνυμα. Με αυτόν τον τρόπο, ο κόμβος v εκτελεί έναν σιωπηρό μηχανισμό για την ανίχνευση και αποφυγή βρόχων. Καθώς το δέντρο δημιουργείται, κάθε κόμβος στέλνει περιοδικά τα δεδομένα γειτονιάς του μέσω της γονικής θύρας χρησιμοποιώντας ένα μήνυμα topoReply. Αυτή η κυκλική διαδικασία ξεκινά ασύγχρονα από τους κόμβους των φύλλων. Δεδομένης

της τοπολογικής τους φύσης, εάν οι μητρικές θύρες των κόμβων φύλλων και κόμβων ν-φύλλων αποτύχουν, δεν μπορούν να παρέχουν μια εναλλακτική διαδρομή ελέγχου στους ελεγκτές SDN. Ως εκ τούτου, ένα πεδίο bit χρησιμοποιείται στο μήνυμα topoReply για να κλαδεύσει η γειτονική θύρα τους. Με αυτόν τον τρόπο, τα μηνύματα topoReply συγκεντρώνονται στους ελεγκτές SDN, οι οποίοι λαμβάνουν το πολύ ένα συγκεντρωτικό μήνυμα από καθεμία από τις ενεργές διεπαφές τους.

Τα οφέλη από την υιοθέτηση του πρωτοκόλλου SHTD είναι πολλά. Πρώτον, ο μηχανισμός χρησιμοποιεί την ελάχιστη δυνατή ποσότητα μηνυμάτων (δηλ. ελάχιστη επιβάρυνση επικοινωνίας) και κάθε μήνυμα έχει μικρό μέγεθος. Επομένως, είναι εύκολο να εφαρμοστεί και να είναι αποτελεσματικό. Δεύτερον, τα μηνύματα echoReply επιτρέπουν ακριβείς μετρήσεις λανθάνουσας κατάστασης στην αφηρημένη προβολή δικτύου στους ελεγκτές SDN. Τρίτον, το προτεινόμενο σχήμα αποκατάστασης σφαλμάτων επιτρέπει στις συσκευές δικτύου να ανακτήσουν σταθερά το δίκτυο από «σπασμένες» καταστάσεις με πολύ χαμηλούς χρόνους ανάκτησης, λίγες απώλειες πακέτων και μειωμένες απαιτήσεις μνήμης σε συσκευές δικτύου. Τέλος, το προτεινόμενο πρωτόκολλο SHTD επιτρέπει μια στρατηγική επιβίωσης για τη διασφάλιση της αξιοπιστίας του επιπέδου ελέγχου.

```
Node v receives topoRequest from node u by port p
if p.state=Standby then
  Send echoReply to node u
  STATEMACHINE(p)
  Send topoRequest for all ports except p
else
  Send echoReply to node u
  Discard topoRequest
end if
```

Εικόνα 4.4 : Αλγόριθμος 1 προώθησης μηνύματος topoRequest



Εικόνα 4.5: Λειτουργία του πρωτοκόλλου Self-Healing πηγή [21]

4.4 Το πρωτόκολλο TEDP

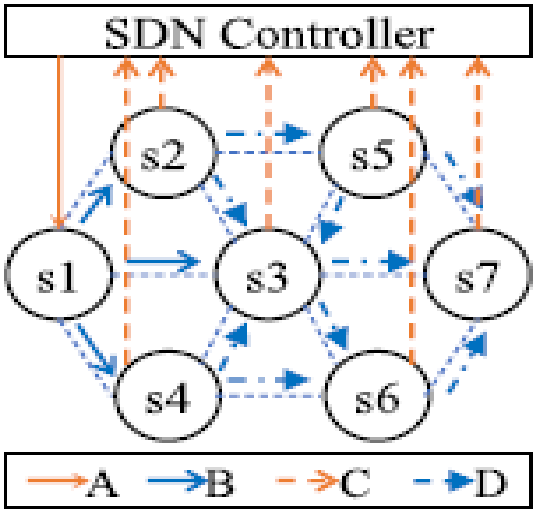
Στο άρθρο [22], παρουσιάζεται το πρωτόκολλο ανακάλυψης εξερεύνησης δέντρων (Tree Exploration Discovery Protocol-TEDP), αποδεικνύοντας ότι οι συντομότερες διαδρομές μπορούν να κατασκευαστούν την ίδια στιγμή που συγκεντρώθηκαν οι πληροφορίες τοπολογίας, χωρίς επιπλέον μηνύματα σε σύγκριση με το LLDP.

Σε αντίθεση με άλλα σχετικά πρωτόκολλα, δεν βασίζεται στην ανταλλαγή LLDP μηνυμάτων μεταξύ γειτονικών κόμβων για την ανακάλυψη. Τα μηνύματα ανταλλάσσονται από σημείο σε σημείο με μεταγωγείς, και αυτές οι πληροφορίες κοινοποιούνται στη συνέχεια στον κεντρικό ελεγκτή SDN. Το TEDP ακολουθεί μια εντελώς διαφορετική προσέγγιση: αντί να στέλνει και να λαμβάνει μηνύματα ανακάλυψης μεταξύ γειτόνων, στέλνει απλώς ένα πλαίσιο ανιχνευτή που πλημμυρίζει το δίκτυο και εξερευνά ολόκληρη την τοπολογία του ταυτόχρονα. Από αυτήν την ιδέα, προκύπτουν δύο προκλήσεις:

1) *Εκπομπή χωρίς βρόχο*: Ποια είναι η διαδρομή που πρέπει να ακολουθήσει αυτό το πλαίσιο (εάν υπάρχει) για να φτάσει σε όλους τους κόμβους τοπολογίας; Μπορούν να αποτραπούν οι βρόχοι;
2) *Μεταφορά πληροφοριών τοπολογίας*: Πρέπει αυτό το πλαίσιο να μεταφέρει όλες τις πληροφορίες τοπολογίας που έχουν ενσωματωθεί ή μόνο επαυξητικές ενημερώσεις;

Για να ανταποκριθεί στην πρώτη πρόκληση, το TEDP αξιοποιεί τον μηχανισμό κλειδώματος All-Path, ο οποίος επιτρέπει σε ένα πλαίσιο να εξερευνά το δίκτυο χωρίς

βρόχους. Η μόνη απαίτηση είναι οι συσκευές δικτύου να μπορούν να επεξεργάζονται το πλαίσιο TEDP, όπως και με το LLDP. Όσον αφορά τη δεύτερη πρόκληση, το TEDP συλλέγει τις πληροφορίες τοπολογίας σε κάθε hop και τις στέλνει στον ελεγκτή SDN, υπεύθυνος για τη συγκέντρωσή τους. Έτσι, η συλλογή πληροφοριών πραγματοποιείται ακριβώς όπως στο LLDP. Η διαφορά μεταξύ LLDP και TEDP, και ταυτόχρονα ένα πλεονέκτημα του πρωτοκόλλου TEDP, είναι ότι τα πλαίσια εξερεύνησης TEDP διασχίζουν το δίκτυο, προωθώντας έτσι την ευκαιρία να βρεθούν βέλτιστες διαδρομές μεταξύ ζευγών κόμβων χωρίς επιπλέον κόστος.



Εικόνα 4.6: Λειτουργία του πρωτοκόλλου TEDP πηγή [22]

Η λειτουργία του TEDP συνοψίζεται στην Εικόνα 4.6, και διακρίνεται σε τέσσερα στάδια. Στο βήμα Α, ο ελεγκτής SDN επιλέγει έναν μεταγωγέα για να ξεκινήσει η διαδικασία ανακάλυψης με πρωτόκολλο TEDP (μεταγωγέας s1). Το πλαίσιο TEDP αποστέλλεται ως Packet-Out από τον μεταγωγέα (s1) και μεταφέρεται αργότερα μέσω όλων των θυρών στο βήμα Β. Στο βήμα Γ, το εισερχόμενο πλαίσιο TEDP αποστέλλεται στον ελεγκτή SDN από όλες τις συσκευές δικτύου που το λαμβάνουν, δηλαδή όλοι οι γείτονές του (s2, s3 και s4 στο σχήμα). Μέχρι αυτό το σημείο, η λειτουργία στο TEDP είναι ίδια με το LLDP. Η διαφορά εμφανίζεται στο βήμα D (διακεκομμένα βέλη), το οποίο ορίζει ότι το πλαίσιο TEDP πρέπει να προωθηθεί στο υπόλοιπο δίκτυο, έως ότου φτάσει σε κάθε συσκευή σε αυτό (επαναλαμβάνοντας το βήμα C για κάθε μεταγωγέα που έχει επιτευχθεί). Κατά συνέπεια, ο αριθμός των μηνυμάτων που ανταλλάσσονται στο επίπεδο δεδομένων είναι ο ίδιος όπως στο LLDP, αλλά η διαδικασία ενεργοποιείται από έναν μόνο μεταγωγέα στο TEDP, απαιτώντας έτσι λιγότερα μηνύματα επιπέδου ελέγχου από το LLDP. Επιπλέον, το TEDP μπορεί να συγκεντρώσει διαδρομές που

βασίζονται σε λανθάνουσα κατάσταση στη διαδικασία. Για να το επιτύχει, αποθηκεύει τη θύρα άφιξης του πρώτου αντιγράφου.

4.5 Το πρωτόκολλο eTDP

Στο άρθρο [23] παρουσιάζεται ένα πρωτόκολλο το οποίο, σε αντίθεση με τις υπάρχουσες προσεγγίσεις, ενεργοποιεί ένα κατανεμημένο πρωτόκολλο 2^{ου} επιπέδου, χωρίς να χρειάζεται ο ελεγκτής να γνωρίζει το δίκτυο και κάποια πρότερη διαμόρφωση του δικτύου. Χρησιμοποιώντας αυτόν τον μηχανισμό, ο ελεγκτής SDN μπορεί να ανακαλύψει την προβολή δικτύου χωρίς να υπάρχουν προβλήματα κλιμάκωσης, ενώ ταυτόχρονα εκμεταλλεύεται τις συντομότερες διαδρομές ελέγχου προς κάθε μεταγωγέα.

Ο παρουσιαζόμενος μηχανισμός ανακάλυψης τοπολογίας αρχικοποιείται από κάθε ελεγκτή SDN που στέλνει ένα μήνυμα topoRequest. Αυτό το μήνυμα πολλαπλής διανομής στη συνέχεια διαδίδεται σε όλο το δίκτυο δημιουργώντας μια τοπολογία δέντρου ελέγχου ριζωμένη στους ελεγκτές SDN για τη συλλογή δεδομένων κατάστασης δικτύου. Επιπλέον, αυτό το δέντρο ελέγχου κατανέμει επίσης τη διαχείριση της φυσικής υποδομής σε διάφορους ελεγκτές SDN. Με εξαίρεση τον ελεγκτή SDN, οι κόμβοι έχουν έναν από τους τρεις ρόλους, δηλαδή κόμβοι φύλλα (v-leaf), φύλλα (leaf) ή πυρήνες (core), ανάλογα με τη θέση τους στην τοπολογία του δικτύου. Οι κόμβοι φύλλων είναι οι κόμβοι στο δίκτυο που έχουν μόνο έναν γείτονα. Ένας κόμβος είναι v-leaf όταν έχει περισσότερους από έναν γείτονες, αλλά μόνο ένας από αυτούς μπορεί να παρέχει μια διαδρομή στους ελεγκτές SDN. Οι υπόλοιποι μεταγωγείς χαρακτηρίζονται ως κόμβοι πυρήνα. Επιπλέον, κάθε ενεργή θύρα λαμβάνει μία από τις τέσσερις καταστάσεις που σχετίζονται με το δέντρο ελέγχου: κατάσταση αναμονής, γονέας, παιδί ή κλάδεμα.

Αρχικά, κάθε κόμβος δικτύου βρίσκεται σε κατάσταση μη ανακάλυψης, με όλες τις θύρες του σε κατάσταση αναμονής, περιμένοντας ένα μήνυμα topoRequest από έναν ελεγκτή SDN ή έναν άλλο κόμβο. Αφού λάβουν το πρώτο τους μήνυμα topoRequest, ανακαλύπτονται κόμβοι μέσω του προτεινόμενου eTDP. Ο αλγόριθμος 1 (Εικόνα 4.7) δείχνει τον μηχανισμό προώθησης για έναν δεδομένο κόμβο v, αφού λάβει το μήνυμα topoRequest. Όταν ο κόμβος v λαμβάνει ένα topoRequest από τον κόμβο u, ένα one hop echoReply μήνυμα αποστέλλεται πίσω στον κόμβο u. Αυτή η αυτόματη απάντηση επιτρέπει την ανταλλαγή αναγνωριστικών κόμβου και θύρας μεταξύ αυτών των

γειτόνων καθώς και τη μέτρηση του RTT σε αυτόν τον σύνδεσμο δικτύου. Επιπλέον, το bit συσχέτισης που περιέχεται σε αυτήν την απάντηση χρησιμοποιείται από τον κόμβο v ως επιβεβαίωση σύνδεσης, για να ανακοινώσει στον γειτονικό κόμβο u αν είναι συνδεδεμένα ή όχι στο δέντρο ελέγχου. Έτσι, ο κόμβος v έχει έναν σιωπηρό μηχανισμό που ανιχνεύει και αποτρέπει βρόχους.

```
Node v receives topoRequest from node u by port p
if p.state=Standby then
  Send echoReply to node u
  StateMachine(p)       |>p.state=Parent
  Send topoRequest for all ports except p
else
  Send echoReply to node u
  Discard topoRequest
end if
```

Εικόνα 4.7: Αλγόριθμος 1 προώθησης μηνύματος topoRequest

4.6 Το πρωτόκολλο OFDPp

Στο άρθρο [24], προτείνεται μια προσέγγιση ανακάλυψης τοπολογίας βασισμένη σε διακομιστή μεσολάβησης (proxy) που μπορεί να μειώσει σημαντικά πολλές παραμέτρους σε σύγκριση με την μέθοδο αναζήτησης OFDP. Με τη χρήση ενός διακομιστή μεσολάβησης, το κέρδος απόδοσης μπορεί να επιτευχθεί χωρίς να αλλάξει η διαμόρφωση του μεταγωγέα ή το πρωτόκολλο OpenFlow. Η προτεινόμενη μέθοδος πειραματικά υπερτερεί τόσο από τη μέθοδο OFDP όσο και από την βελτιστοποιημένη OFDPv2 εκδοχή. Το πρωτόκολλο OFDPp μειώνει το overhead του ελεγκτή, χωρίς όμως να υπολείπεται της ικανότητας "ανακάλυψης" της τοπολογίας του δικτύου.

Σε μια προσπάθεια να μειωθεί ακόμη περισσότερο το πλήθος των μηνυμάτων που επιβαρύνουν τον ελεγκτή, μα λύση είναι η αύξηση ενός χρονομέτρου του κύκλου του πακέτου LLDP. Η προφανής επίδραση από μια τέτοια τροποποίηση είναι η μείωση του χειρισμού της δυναμικής του δικτύου. Ο στόχος της λύσης αυτής είναι, να μειωθεί η κίνηση του επιπέδου ελέγχου, διατηρώντας παράλληλα την ίδια ικανότητα και να 'ανακαλύπτονται' σύνδεσμοι στο δίκτυο. Μια διαισθητική λύση θα μπορούσε να είναι η εκχώρηση της διαδικασίας κύκλου LLDP στο μεταγωγέα, καθώς αυτό θα μειώσει

άμεσα τον αριθμό των μηνυμάτων που ανταλλάσσονται στο επίπεδο ελέγχου. Ωστόσο, αυτή η λύση θα απαιτήσει τροποποιήσεις σε όλους τους μεταγωγείς του δίκτυο. Για να διατηρηθεί η συμβατότητα με υπάρχοντα δίκτυα και εξοπλισμό, δεν επιδιώχτηκε να αλλαχτεί το πρωτόκολλο OpenFlow, ο ελεγκτής SDN ή οι μεταγωγείς. Μια λύση που βασίζεται σε διακομιστή μεσολάβησης μπορεί να πληροί αυτές τις απαιτήσεις και να παρέχει μια γενική λύση που δεν εξαρτάται από ένα συγκεκριμένο κατασκευαστή μεταγωγέα ή από ένα συγκεκριμένο ελεγκτή SDN

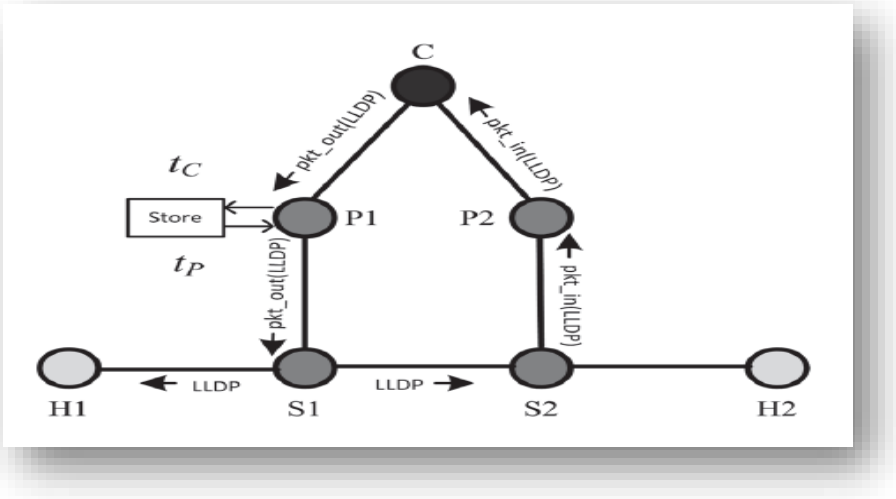
```
function RECEIVE(packet, source s, dest d)
     if s==controller then
        if typeof(p)==pkt_out then
            if typeofEncapsulatePkt(p)==LLDP then
                 Store(p,d)
                 ResetTimer(t p)
             end if
          end if
          SendToSwitch(p,d)
       else
          SendToController(p,d)
        end if
end function

function ONTIMEREXPIRED(t p)
       p,d<-- RetrieveStoredPacket()
       SendToSwitch(p,d)
       ResetTimer(t p)
end function
```

Εικόνα 4.8: Ψευδοκώδικας για το OFDPp

Στην Εικόνα 4.8 δείχνει τη λειτουργικότητα διακομιστή μεσολάβησης σε ψευδο κώδικα. Στην εικόνα εμφανίζεται μόνο ο ψευδοκώδικας που σχετίζεται με την αποθήκευση, προώθηση και επανάληψη του μηνύματος LLDP. Το μήνυμα LLDP είναι ενθυλακωμένο μέσα στο μήνυμα OpenFlow και αναγνωρίζεται από το πεδίο Ethertype στην κεφαλίδα του ενθυλακωμένου μηνύματος, το οποίο είναι 0x88cc. Όταν ο διακομιστής μεσολάβησης λαμβάνει και αναγνωρίζει ένα πακέτο LLDP, αντιγράφεται και αποθηκεύεται πριν προωθηθεί στον μεταγωγέα. Όλα τα άλλα πακέτα προωθούνται με διαφάνεια μεταξύ του ελεγκτή και του μεταγωγέα. Ο διακομιστής μεσολάβησης είναι επίσης διαφανής για τα μηνύματα στην αντίθετη κατεύθυνση, δηλαδή μεταξύ του μεταγωγέα και του ελεγκτή.

Σε αυτό το άρθρο προτείνουμε μια λύση με βάση το διακομιστή μεσολάβησης (OFDPp) που μπορεί να εφαρμοστεί σε υπάρχοντα δίκτυα SDN χωρίς να αλλάξει τους μεταγωγείς, τον ελεγκτή ή το OpenFlow. Μέσω πειραμάτων, δείξαμε ότι η μέθοδος μπορεί να μειώσει σημαντικά την επιβάρυνση της κυκλοφορίας ελέγχου σε σύγκριση

με το OFDP χωρίς να μειώσει την ικανότητα να ανακαλύψει την τοπολογία του δικτύου.



Εικόνα 4.9: Λειτουργία του πρωτοκόλλου OFDPp πηγή [24]

Η λειτουργία του είναι απλή και απεικονίζεται στην Εικόνα 4.9.

Η διαδικασία εκτέλεσης του LLDP μεταβιβάζεται σε έναν διακομιστή μεσολάβησης, ο οποίος φυσικά τοποθετείται κοντά στο μεταγωγέα (στην Εικόνα 4.9, τα P1 και P2 είναι διακομιστές μεσολάβησης). Ο διακομιστής μεσολάβησης είναι φανερός σε κάθε επικοινωνία μεταξύ του ελεγκτή και του μεταγωγέα. Ωστόσο, ο διακομιστής μεσολάβησης μπορεί να διαβάσει, να ερμηνεύσει και να μεταδώσει μηνύματα LLDP που είναι ενσωματωμένα ως μηνύματα Packet-Out. Επιπλέον, ο διακομιστής μεσολάβησης εκμεταλλεύεται το γεγονός ότι όλα τα μηνύματα LLDP που προορίζονται για τον ίδιο μεταγωγέα είναι ίδια. Ως εκ τούτου, ο διακομιστής μεσολάβησης μπορεί να στείλει ένα αντίγραφο του προηγούμενου LLDP (το οποίο είναι αποθηκευμένο) αντί να περιμένει ένα νέο μήνυμα. Με τον τρόπο αυτό επιτυγχάνεται

4.7 Το πρωτόκολλο Lightweight (LADP)

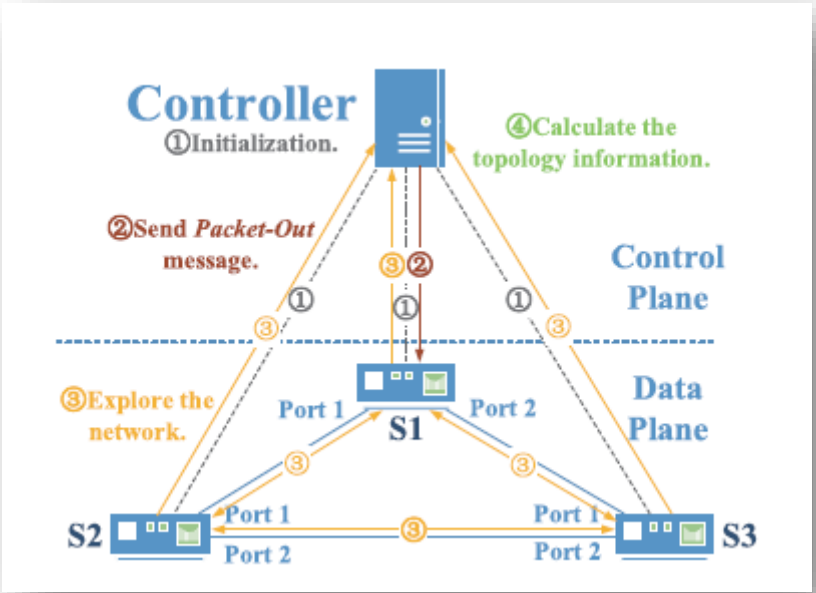
Στο άρθρο [25], προτείνεται ένα αυτόματο πρωτόκολλο ανακάλυψης τοπολογίας για τη συλλογή και ενημέρωση των πληροφοριών τοπολογίας με τρόπο ασφαλή και αποτελεσματικό. Αυτή η τεχνική υλοποιείται χωρίς περιττές τροποποιήσεις στο τυπικό πρωτόκολλο OpenFlow και χωρίς προσθήκη ‘λογικής’ στο υλικό των μεταγωγέων.

Σχεδιάστηκε μια νέα δομή πλαισίου ανιχνευτή για το Lightweight Automatic Discovery Protocol (LADP) με σκοπό τη μείωση της κυκλοφορίας των πακέτων και τη βελτίωση της απόδοσης. Η δομή του πλαισίου LADP φαίνεται στην Εικόνα 4.10. Το πλαίσιο LADP αποτελείται από δύο μέρη: την κεφαλίδα (*header*) του πλαισίου LADP και το ωφέλιμο φορτίο (*payload*) του πλαισίου LADP. Η κεφαλίδα πλαισίου LADP περιλαμβάνει τρία πεδία: διεύθυνση προορισμού Dst (*Dst address*), διεύθυνση πηγής Src (*Src address*) και το πεδίο *Ethernet-type*. Το πεδίο *διεύθυνσης Dst* και το πεδίο τύπου Ether χρησιμοποιούνται για την καταγραφή μιας διεύθυνσης προορισμού πολλαπλής διανομής και ενός συγκεκριμένου αριθμού πρωτοκόλλου, αντίστοιχα. Το πεδίο *διεύθυνσης Src* χρησιμοποιείται για την καταγραφή της διεύθυνσης MAC της θύρας εξόδου, η οποία αντιπροσωπεύει την πηγή του πλαισίου. Παρόμοια με το LLDP, το πλαίσιο LADP αξιοποιεί τη δομή τιμής μήκους τύπου (TLV) για τη δημιουργία του ωφέλιμου φορτίου πλαισίου LADP. Ωστόσο, το ωφέλιμο φορτίο πλαισίου LLDP περιλαμβάνει τέσσερα υποχρεωτικά TLVs (δηλαδή Chassis ID TLV, Port ID TLV, TTL TLV και End TLV), ενώ το LADP διατηρεί μόνο το End TLV για να υποδείξει το τέλος του πλαισίου.

|<----------Frame header---------->|<----------Frame payload---------->

Dst address	Src address	Ether-type	AUTH TLV	End TLV

Εικόνα 4.10:Μορφή πλαισίου LADP



Εικόνα 4.11: Λειτουργία του πρωτοκόλλου Lightweight πηγή [25]

Στην Εικόνα 4.11 δείχνει την προσέγγιση LADP σε ένα απλό σενάριο. Η προσέγγιση του LADP περιλαμβάνει τα ακόλουθα τέσσερα στάδια:

1) Αρχικοποίηση: Κάθε πλευρά της σύνδεσης OpenFlow ξεκινά μια διαδικασία χειραψίας για αυτόματη ανακάλυψη κόμβου. Μετά τη διαδικασία χειραψίας, ο ελεγκτής ανακαλύπτει όλους τους μεταγωγείς. Στη συνέχεια, ο ελεγκτής καταγράφει τις πληροφορίες του μεταγωγέα και τις πληροφορίες θύρας (π.χ. θύρα ID και διεύθυνση MAC), οι οποίες είναι απαραίτητες για τον ελεγκτή να παρέχει υπηρεσίες εντοπισμού τοπολογίας. Για να επιτρέψει στους μεταγωγείς να μεταδίδουν το πλαίσιο LADP και να συλλέγουν αυτόματα τις πληροφορίες σύνδεσης, ο ελεγκτής στέλνει μηνύματα Flow-Mod σε καθέναν από τους διακόπτες για την εγκατάσταση ενός συνόλου κανόνων.
2) Το επίπεδο ελέγχου δημιουργεί και στέλνει ένα πλαίσιο LADP στο επίπεδο δεδομένων: Ο ελεγκτής επιλέγει έναν μεταγωγέα ως τον κεντρικό τυχαία (π.χ. S1) και στέλνει ένα μήνυμα Packet-Out που περιλαμβάνει ένα πλαίσιο LADP και μια οδηγία προώθησης σε αυτό. Η οδηγία προώθησης τροποποιεί και εξάγει μόνο το πλαίσιο LADP. Ο αριθμός των μηνυμάτων Packet-Out σε κάθε περίοδο ανακάλυψης είναι Pout = 1.
3) Το πλαίσιο LADP εξερευνά το δίκτυο: Ο κεντρικός διακόπτης S1 αναλύει το ληφθέν μήνυμα Packet-Out και μεταδίδει το πλαίσιο LADP. Μετά τη λήψη των πλαισίων LADP, οι S2 και S3 τροποποιούν και μεταδίδουν. Στη συνέχεια, το S1 λαμβάνει τα πλαίσια LADP που αποστέλλονται από τα S2 και S3, το S2 λαμβάνει το πλαίσιο LADP που αποστέλλεται από το S3 και το S3 λαμβάνει το πλαίσιο LADP που αποστέλλεται από το S2. Ταυτόχρονα, κάθε μεταγωγέας στέλνει το ληφθέν πλαίσιο LADP στον ελεγκτή μέσω μηνύματος Packet-In. Επομένως, ο ελεγκτής λαμβάνει όλα τα LADP. Δηλώνουμε τον αριθμό των συνδέσμων ως L, ο αριθμός των μηνυμάτων Packet-In σε κάθε περίοδο ανακάλυψης είναι Pin = 2L.
4) Ο ελεγκτής υπολογίζει τις πληροφορίες τοπολογίας: για κάθε LADP, από τη μία πλευρά, ο ελεγκτής εξάγει το αναγνωριστικό αποστολέα (π.χ. S2) και τον αριθμό θύρας (π.χ. Θύρα 1) από το μήνυμα Packet-In, το οποίο αντιπροσωπεύει τον προορισμό ενός μονοκατευθυντικού συνδέσμου. Από την άλλη πλευρά, ο ελεγκτής αναλύει το πλαίσιο LADP και αναζητά τη διεύθυνση Src στη διεύθυνση MAC της θύρας που είναι αποθηκευμένη στον ελεγκτή (π.χ. διεύθυνση MAC της θύρας 1 στο

S1), η οποία αντιπροσωπεύει την πηγή του μονοκατευθυντικού συνδέσμου. Στη συνέχεια, ο ελεγκτής μπορεί να συμπεράνει την ύπαρξη του μονοκατευθυντικού συνδέσμου (δηλαδή από τη θύρα 1 στο S1 έως τη θύρα 1 στο S2). Τέλος, ο ελεγκτής μπορεί να υπολογίσει όλους τους διασυνδεδεμένους συνδέσμους αναλύοντας όλα τα ληφθέντα πλαίσια LADP.

4.8 Το πρωτόκολλο Im-OFDP

Οι σημερινοί ελεγκτές χρησιμοποιούν το OFDP για να εξερευνήσουν την υποκείμενη τοπολογία δικτύου. Για τη βελτίωση της απόδοσης και της ασφάλειας του OFDP, προτείνεται ένα βελτιωμένο πρωτόκολλο Im-OFDP [26], εκμεταλλευόμενο τον αλγόριθμο ελάχιστο κάλυψης κορυφής. Τα πειράματα δείχνουν ότι το Im-OFDP υπερτερεί των OFDP και OFDPv2 όσον αφορά τον αριθμό πακέτου LLDP, το φορτίο της CPU και την ασφάλεια του ελεγκτή.

Σε αυτό το πρωτόκολλο, η βασική υπόθεση είναι ότι όλοι οι σύνδεσμοι μεταξύ των μεταγωγέων είναι αμφίδρομοι, οι οποίοι είναι σύμφωνοι με την πραγματική κατάσταση. Το Im-OFDP μπορεί να μειώσει τον αριθμό των μηνυμάτων Packet-Out και Packet-In με πακέτα LLDP σημαντικά υπό αυτήν την κατάσταση.

Για την ‘ανακάλυψη’ της τοπολογίας, το Im-OFDP χρησιμοποιεί τρεις ειδικούς κανόνες ροής τους οποίους ο ελεγκτής προεγκαταστεί στους μεταγωγείς. Ο κανόνας I εγκαθίσταται μόνο στους μεταγωγείς υποστήριξης, ενώ οι κανόνες II και III είναι εγκατεστημένοι σε όλους τους μεταγωγείς. Στον κανόνα I, οι θύρες εξόδου εκχωρούνται από τον ελεγκτή. Ο κανόνας I είναι υψηλής προτεραιότητας, πράγμα που σημαίνει ότι εάν ο μεταγωγέας λαμβάνει μηνύματα Packet Out από τον ελεγκτή με πακέτα LLDP με dl src = 0, θα τον προωθεί σε θύρες που έχουν εκχωρηθεί από τον ελεγκτή. Ο κανόνας II έχει μεσαία προτεραιότητα, πράγμα που σημαίνει ότι εάν ο μεταγωγέας λαμβάνει πακέτα LLDP με dl src = 0, θα ξαναγράψει το πεδίο dl src με τη διεύθυνση mac της θύρας που λαμβάνει τα πακέτα LLDP και θα τα στείλει ξανά στην ίδια θύρα. Τέλος, ο κανόνας III είναι χαμηλής προτεραιότητας, και αυτό σημαίνει ότι εάν ο μεταγωγέας λαμβάνει πακέτα LLDP, θα τα προωθήσει στον ελεγκτή εντός μηνυμάτων Packet In.

Η εικόνα 4.12 απεικονίζει τον αλγόριθμο του Im-OFDP για την ανακάλυψη αμφίδρομης σύνδεσης (S1, Port1) - (S2, Port3) και η Εικόνα 4.14 την λειτουργία του

που περιγράφεται παρακάτω.

Η λειτουργία έχει ως εξής:

Βήμα 1: Ο ελεγκτής εγκαθιστά τους κανόνες I II III που περιγράφεται στο S1 και εγκαθιστά τους κανόνες II III στο S2.

Βήμα 2: Ο ελεγκτής στέλνει μήνυμα LLDP Packet-Out στο οποίο το ChassisID είναι S1, το PortID είναι 0 και το SrcMAC είναι 0 έως S1.

Βήμα 3: Το S1 λαμβάνει το μήνυμα LLDP Packet-Out από τον ελεγκτή και στη συνέχεια, διανέμεται το πακέτο LLDP σε ρυθμιζόμενες θύρες στον Κανόνα I.

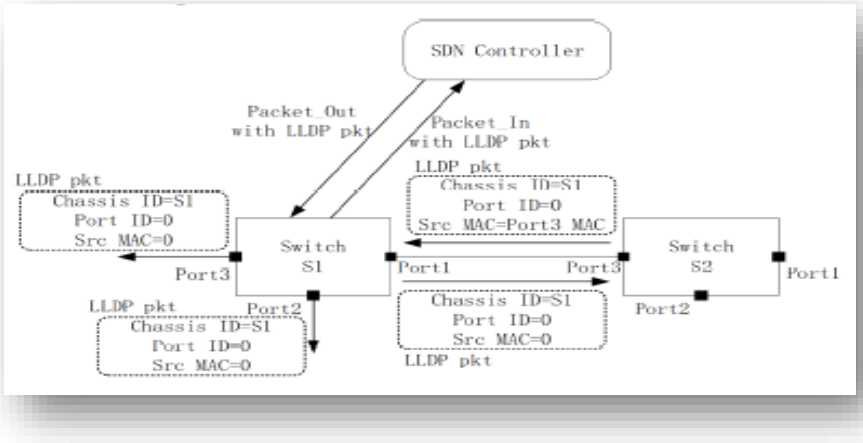
Βήμα 4: Το S2 λαμβάνει το πακέτο LLDP στο Port3, στη συνέχεια το επεξεργάζεται. Εάν SrcMAC = 0, τότε ξαναγράφει το πεδίο SrcMAC με διεύθυνση MAC του (S2, Port3) και στέλνει πίσω στο S1 σύμφωνα με τον Κανόνα II.

Βήμα 5: Το S1 λαμβάνει το πακέτο LLDP που αποστέλλεται από το S2 στη θύρα 1 και, στη συνέχεια, το προωθεί στον ελεγκτή μέσω μηνύματος Packet-In.

Βήμα 6: Ο ελεγκτής λαμβάνει το μήνυμα LLDP Packet In από το S1, μπορεί να συμπεράνει την ύπαρξη συνδέσμου (S1, Port1) - (S2, Port3). Μπορούμε να διαπιστώσουμε ότι το Im-OFDP χρειάζεται μόνο να αναλύσει ένα μήνυμα Packet In για να μάθει έναν αμφίδρομο σύνδεσμο, ενώ τόσο το OFDP όσο και το OFDPv2 απαιτούν δύο μηνύματα Packet-In.

```
for received packet pkt do
   if pkt.etherType=LLDP and pkt.inPort=CONT and pkt.srcMac=0 then
        for all switch ports P in Rule I do
           forward pkt copy on port P
        end for
    else if pkt.etherType=LLDP and pkt.srcMac=0 then
         pkt.srcMac=P.MACaddr
    else if pkt.etherType=LLDP then
         forward pkt to Controller
    end if
end for
```

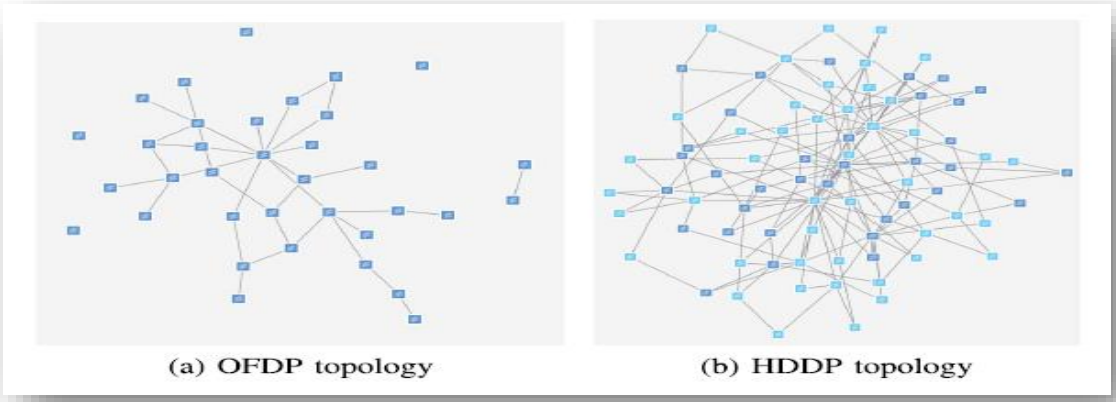
Εικόνα 4.12: LLDP frame processing algorithm



Εικόνα 4.13:Λειτουργία του πρωτοκόλλου Im-OFDP πηγή [26]

4.9 Το πρωτόκολλο HDDP

Στο άρθρο [27] παρουσιάζεται το HDDP, ένα καινοτόμο πρωτόκολλο ικανό να επιτύχει μια πλήρη ανακάλυψη τοπολογίας σε υβριδικούς τομείς SDN, όπου συνυπάρχουν συσκευές SDN και μη SDN συσκευές. Το πρωτόκολλο βασίζεται σε έναν μηχανισμό εξερεύνησης που ενεργοποιείται από το επίπεδο ελέγχου, ο οποίος φτάνει σε όλες τις συσκευές μέσω ενός ελεγχόμενου μηχανισμού πλημμύρας για να συλλέξει όλες τις απαραίτητες πληροφορίες από τα υβριδικά δίκτυα. Η πρότασή αυτή ξεπερνά το OFDP δεδομένου ότι είναι σε θέση να ανακαλύψει μια πλήρη υβριδική τοπολογία από ένα δίκτυο τομέα που αποτελείται από συσκευές SDN και εκτός SDN με λιγότερα πακέτα ανταλλαγής.



Εικόνα 4.14:Σύγκριση αποτελεσμάτων του OFDP με το HDDP πρωτόκολλο πηγή [27]

4.10 Σύγκριση και συζήτηση των πρωτοκόλλων εύρεσης τοπολογίας

Στην ενότητα αυτή έγινε μία ανασκόπηση όλων των μεθόδων που προτάθηκαν για την εύρεση της τοπολογίας δικτύων σε δίκτυα SDN. Στη συνέχεια παρουσιάζονται τα συγκριτικά αποτελέσματα αυτών των πρωτοκόλλων.

Πίνακας 4.1 : Γενικά χαρακτηριστικά των πρωτοκόλλων

Reference	Name	Number of references	Controller	Control Frame	Language	Details
Pakzad et al., 2015 [19]	OFDPv2	94	Pox	LLDP	Python	Mininet
Azzouni, et al., 2018 [20]	sOFTDP	26	Floodlight	LLDP	Java	Mininet
Ochoa et al., 2018 [21]	Self-Healing	12	Floodlight	Author defined	Java	OMNeT++
Rojas et al., 2018 [22]	TEDP	14	Ryu	TEDP	Python	Mininet
Ochoa et al., 2019 [23]	eTDP	11	All	Author defined	C++	OMNeT++
Flathagen et al., 2019 [24]	OFDPp	2	Onos	LLDP	Java	Mininet
Jia et al., 2020 [25]	Lightweight	5	Ryu	LADP	Python	Mininet
Gu et al., 2020 [26]	Im-OFDP	3	Pox	LLDP	Python	Mininet
Rojas et al., 2020 [27]	HDDP	3	Onos	HDDP	Java	Mininet

Στον Πίνακα 4.1 βλέπουμε τα *γενικά χαρακτηριστικά* των μεθόδων όπως: η ταυτότητα τους, ο αριθμός των αναφορών που συγκέντρωσαν σύμφωνα με το Google Scholar από την δημοσίευση τους έως και σήμερα (11/4/2021), τον controller για τον οποίο υλοποιήθηκαν, την τεχνική στην οποία βασίστηκε η υλοποίησή τους, και η γλώσσα υλοποίησης του πρωτοκόλλου (που συνήθως συμφωνεί με αυτή του ελεγκτή).

Όλα τα πρωτόκολλα υλοποιήθηκαν σχετικά πρόσφατα, εκτός από το OFDPv2 που παρουσιάστηκε το 2015 και ίσως για το λόγο αυτό να είναι και πιο ευρέως γνωστό. Οι υλοποιήσεις μοιράζονται σχεδόν εξίσου ανά ελεγκτή και οι περισσότερες βασίζονται στο πρωτόκολλο LLDP. Τα περισσότερα πρωτόκολλα έχουν γραφτεί σε γλώσσα Python, ενώ υπάρχουν υλοποιήσεις και σε Java. Για την αξιολόγησή τους

χρησιμοποιήθηκε ο εξομοιωτής Mininet εκτός από δύο περιπτώσεις που χρησιμοποιήθηκε το πλαίσιο OMNeT++.

Πίνακας 4.2 : Ποιοτικά χαρακτηριστικά των πρωτοκόλλων ανακάλυψης τοπολογίας

Identify		**Throughput measurement**			**Topology features**		**Other**
Reference	**Name**	**Controller Overhead**	**Number of Packet-Out**	**Number of Packet-In**	**Topology**	**Number of nodes N**	**Scalability**
Pakzad, et al., 2015 [19]	OFDP	High	$Pout = \sum_{n=1}^{N} p_i$	$P_{IN} = 2L$	Tree, Linear Fat tree	20 85 100 127	yes
Pakzad, et al., 2015 [19]	OFDPv2a OFDPv2b	Low	$Pout = N$	$P_{IN} = 2L$	Tree, Linear Fat tree	20 85 100 127	yes
Azzouni et al., 2018 [20]	sOFTDP	High	$Pout = \sum_{n=1}^{N} p_i$	$P_{IN} = 2L$	BFD machine		yes
Ochoa et al., 2018 [21]	Self-Healing	Low	Pout=N	Pin=N	tree network random graphs	10 17 54	yes
Rojas et al., 2018 [22]	TEDP	High	Pout=1	$P_{IN} = 2L$	GÉANT pan-European network topology	44	no
Ochoa et al., 2019 [23]	eTDP	Low	$Pout = \sum_{k=1}^{m} \sum_{i=1}^{n} p_{(enabled)_{ki}}$	P_{IN}=2Pout	Based on SNDlib [43]	15 27 40	yes
Flathagen et al., 2019 [24]	OFDPp	Low	Pout=N	$P_{IN} = 2L$	Tree, Linear Linear, Fat Tree		no
Jia et al., 2020 [25]	Lightweigh t	Low	Pout=1	$P_{IN} = 2L$	Tree Linear Fat tree		yes
Gu et al., 2020 [26]	Im-OFDP	Low	Pout=Nx	$P_{IN} = L$	Tree, Tree Linear Fat Tree	85 511 500 45	no
Rojas et al., 2020 [27]	HDDP	Low	Pout=N	$P_{IN} = 2L$	GÉANT pan-European network topology, random Barabasi-Albert networks [45][46]	44	yes

Στον Πίνακα 4.2 βλέπουμε τα *ποιοτικά χαρακτηριστικά* των πρωτοκόλλων που μελετήθηκαν και τα οποία αφορούν το πλήθος των πακέτων Packet-In, Packet-Out τα οποία δημιουργεί και στέλνει αντίστοιχα το κάθε πρωτόκολλο, η τοπολογία στην οποία εφαρμόστηκε το πρωτόκολλο και συλλέχτηκαν μετρήσεις επίδοσης από τους ερευνητές καθώς επίσης και τον αριθμό των μεταγωγέων που χρησιμοποιούσαν αντίστοιχα για την κάθε τοπολογία.

Στον Πίνακα 4.2 το Pout συμβολίζει το σύνολο των Packet-Out που αποστέλλονται από τον ελεγκτή στους μεταγωγείς, το N συμβολίζει το σύνολο των μεταγωγέων που μετέχουν στην εκάστοτε τοπολογία. Το Pin συμβολίζει τον αριθμό των πακέτων Packet-In που στέλνουν στον ελεγκτή οι μεταγωγείς κάθε φορά που λαμβάνουν το πακέτο LLDP και με αυτό τον τρόπο ενημερώνουν τον ελεγκτή για την ύπαρξη των συνδέσεων μεταξύ των κόμβων. Το L συμβολίζει τον αριθμό των συνδέσεων του δικτύου, της εκάστοτε τοπολογίας στις περισσότερες των περιπτώσεων είναι 2L. Η επικοινωνία μεταξύ των μεταγωγέων είναι αμφίδρομη δηλαδή και προς τις δυο κατευθύνσεις.

Από τη σύγκριση διαπιστώνουμε ότι τα πρωτόκολλα OFDP, sOFTDP, TEDP, επιφέρουν υψηλή επιβάρυνση στον ελεγκτή ενώ τα υπόλοιπα προκαλούν χαμηλή επιβάρυνση του ελεγκτή. Τα πρωτόκολλα TEDP, Lightweight και το eTDP εισάγουν μόνο 1 πακέτο Packet-out ενώ ο μέγιστος αριθμός αυτών των πακέτων σε άλλα πρωτόκολλα όπως είναι το κλασσικό OFDP είναι το άθροισμα $\sum_{n=1}^{N} pi$. Τα πρωτόκολλα Im-OFDP, Self-Healing παρατηρούμε ότι έχουν το μικρότερο αριθμό πακέτων Packet-In, ο οποίος είναι όσος είναι και ο αριθμός των μεταγωγέων που μετέχουν στην εκάστοτε τοπολογία ενώ ο μέγιστος αριθμός που μπορεί να φτάσει στα περισσότερα από τα πρωτόκολλα είναι το 2L γιατί όπως προαναφέρθηκε και παραπάνω η επικοινωνία των μεταγωγέων είναι αμφίδρομη επομένως το μήνυμα Packet-In θα σταλεί και από τις δυο μεριές.

Με βάση την παραπάνω συζήτηση ξεχωρίζουν τα πρωτόκολλα Im-OFDP και Self-Healing επειδή επιτυγχάνουν και στέλνουν τον μικρότερο αριθμό πακέτων Packet-In και συνεπώς επιφέρουν χαμηλότερη επιβάρυνση στον ελεγκτή σε σχέση με τα υπόλοιπα πρωτόκολλα. Αντίθετα τα πρωτόκολλα OFDP, sOFDP, TEDP, OFDPp, Lightweight, HDDP, eTDP εισάγουν αρκετά μεγαλύτερο αριθμό μηνυμάτων Packet-In με αποτέλεσμα να επιβαρύνουν περισσότερο τον ελεγκτή. Συμπέρασμα όλων των παραπάνω πρωτοκόλλων είναι η προσπάθεια των μελετητών να μειώσουν την

επιβάρυνση που προκαλείται στον ελεγκτή όπως προκύπτει από το κλασσικό πρωτόκολλο OFDP, μειώνοντας σταδιακά είτε τα πακέτα Packet-Out όπως βλέπουμε στο TEDP είτε τα πακέτα Packet-In όπως παρατηρούμε στο Self-Healing ή και τον συνδυασμό αυτών των δύο που επιτυγχάνεται με την υλοποίηση του πρωτοκόλλου Im-OFDP.

Πίνακας 4.3:Χαρακτηριστικές μετρήσεις για την αξιολόγηση των πρωτοκόλλων

Reference	Name	Protocol Performance
Pakzad et al., 2015 [19]	OFDPv2	Controller Overhead, Impact On Controller CPU load
Azzouni, et al., 2018 [20]	sOFTDP	Controller Overhead
Ochoa et al., 2018 [21]	Self-Healing	Controller Overhead, Recovery time, Discovery Packets Per Switch
Rojas et al., 2018 [22]	TEDP	Controller Overhead
Ochoa et al., 2019 [23]	eTDP	Discovery Time, Controller Overhead, Discovery Packets Per Switch
Flathagen et al., 2019 [24]	OFDPp	Controller Overhead
Jia et al., 2020 [25]	Lightweight	Controller Overhead
Gu et al., 2020 [26]	Im-OFDP	Controller Overhead, Impact On Controller CPU load
Rojas et al., 2020 [27]	HDDP	Controller Overhead

Στον Πίνακα 4.3 περιγράφονται οι μετρικές επίδοσης που εξετάστηκαν από τους δημιουργούς των πρωτοκόλλων για να συμπεράνουν την απόδοση της προτεινόμενης μεθόδου. Οι ερευνητές εστιάζουν κυρίως στην επιβάρυνση του ελεγκτή, σε αριθμό μηνυμάτων που εισάγει η διαδικασία της ανίχνευσης της τοπολογίας. Επίσης, σε δύο έρευνες διερευνάται η επίδραση που έχει η διαδικασία ανακάλυψης της τοπολογίας στην χρήση CPU. Αν και, ο χρόνος που απαιτείται για την ανακάλυψη της τοπολογίας είναι μια κρίσιμη παράμετρος για την λειτουργία του δικτύου, εξετάζεται μόνο σε μια ερευνητική εργασία.

4.11 Δίκτυα SDN και πολυβάθμια δικτυακά συστήματα

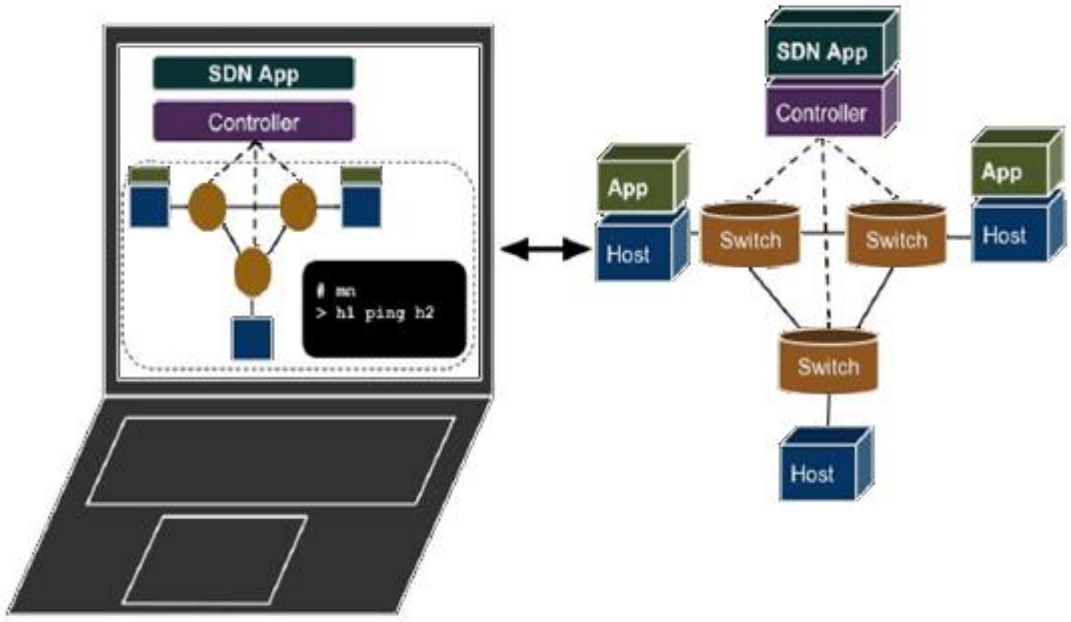
Σε περιπτώσεις που οι SDN controllers βρίσκονται σε μεγάλες αποστάσεις σε σχέση με τις ελεγχόμενες δικτυακές συσκευές (network devices), έτσι ώστε τα πακέτα ελέγχου (Packet-out, Packet-In κλπ.) να είναι υποχρεωμένα να διέλθουν μέσα από διαφόρων τύπων πολυ-βάθμια συστήματα (π.χ. Banyan delta type [38] ή Banyan Omega type [39]), οι ροές αυτές των πακέτων ελέγχου δεν επηρεάζονται και ολοκληρώνουν τον κύκλο τους κανονικά. Επίσης, η ροή των πακέτων δεν επηρεάζεται από τον μηχανισμό λειτουργίας των πολυβάθμιων συστημάτων π.χ. store and forward, wormhole [40] ή cut-through μηχανισμό προώθησης δεδομένων [39].

Επιπλέον οι ροές των πακέτων ελέγχου μένουν ανεπηρέαστες είτε τα ενδιάμεσα Multistage Interconnection Networks εξυπηρετούν Unicast ή Multicast [41] τύπου φορτίο, ακόμη και όταν αυτά είναι U-type fabrics [42].

5.Υλοποίηση δικτύων με τη βοήθεια του Mininet

5.1 Εισαγωγή στο Mininet

Το Mininet [15] είναι ένα από τα πρώτα λογισμικά εξομοιωτών ανοιχτού κώδικα που έχουν αναπτυχθεί αποκλειστικά για την υποστήριξη της δικτύωσης καθοριζόμενης από λογισμικό. Είναι ένα σύστημα που επιτρέπει την γρήγορη πρωτοτυποποίηση μεγάλων δικτύων σε έναν απλό υπολογιστή. Επιπλέον, δημιουργεί επεκτάσιμα δίκτυα SDN χωρίς να χρησιμοποιεί μεγάλες απαιτήσεις σε μηχανισμούς εικονοποίησης, όπως διαδικασίες και χώρους ονομάτων δικτύων. Αυτές οι δυνατότητες του επιτρέπουν να δημιουργεί, να αλληλοεπιδρά, να προσαρμόζει και να μοιράζεται τα πρωτότυπα με μεγάλη ταχύτητα. Με άλλα λόγια διαχειρίζεται μια μεγάλη συλλογή από hosts, routers, και switches σε ένα πυρήνα Linux. Τα προγράμματα τα οποία εκτελούνται στέλνουν πακέτα μέσω Ethernet διεπαφής που συμπεριφέρεται σαν πραγματική διεπαφή, με δεδομένη ταχύτητα σύνδεσης και δεδομένη καθυστέρηση.



Εικόνα 5.1: Εξομοίωση Mininet σε σχέση με πραγματικό δίκτυο

5.1.1 Πλεονεκτήματα και περιορισμοί του εξομοιωτή Mininet

Βάση των πλεονεκτημάτων του που παρουσιάζονται στην συνέχεια το Mininet έχει καταφέρει να αναπτυχθεί και να εδραιωθεί πολύ υψηλά στην κατηγορία των εξομοιωτών δικτύων SDN [15]. Αυτά τα πλεονεκτήματα είναι τα παρακάτω:

- Η ταχύτητα, όπου η εκκίνηση μιας εξομοίωσης ενός δικτύου έχει διάρκεια λίγα μονάχα δευτερόλεπτα, αυτό έχει ως φυσικό αποτέλεσμα ότι ο βρόγχος εκτέλεσης-επεξεργασίας-εντοπισμού σφαλμάτων είναι πολύ γρήγορος και μικρός.
- Η δημιουργία προσαρμοσμένων τοπολογιών, δηλαδή δίνεται η δυνατότητα στο χρήστη να επιλέξει το είδος και το μέγεθος της τοπολογίας του δικτύου που θέλει να προσομοιώσει.
- Έχει την ικανότητα εκτέλεσης οποιουδήποτε προγράμματος το οποίο είναι εγκατεστημένο στο λειτουργικό σύστημα Linux.
- Η προώθηση πακέτων, όπου επιτυγχάνεται με την χρήση του πρωτοκόλλου OpenFlow που οφείλεται στους προγραμματιζόμενους μεταγωγείς.
- Η ευελιξία του, καθώς η εγκατάσταση είναι συμβατή από διάφορες πλατφόρμες, υπολογιστές, virtual machines.
- Η ευχρηστία, μιας και ο χρήστης μπορεί να δημιουργεί και να εκτελεί πειράματα απλώς γράφοντας κώδικα σε γλώσσα προγραμματισμού Python.
- Είναι λογισμικό ανοιχτού κώδικα και υποστηρίζεται ενεργά από την κοινότητα του Mininet, όπου έμπειροι προγραμματιστές και χρήστες μπορούν να σου παρέχουν υποστήριξη ή κάποιες συμβουλές σε πιθανά προβλήματα που μπορούν να προκύψουν.

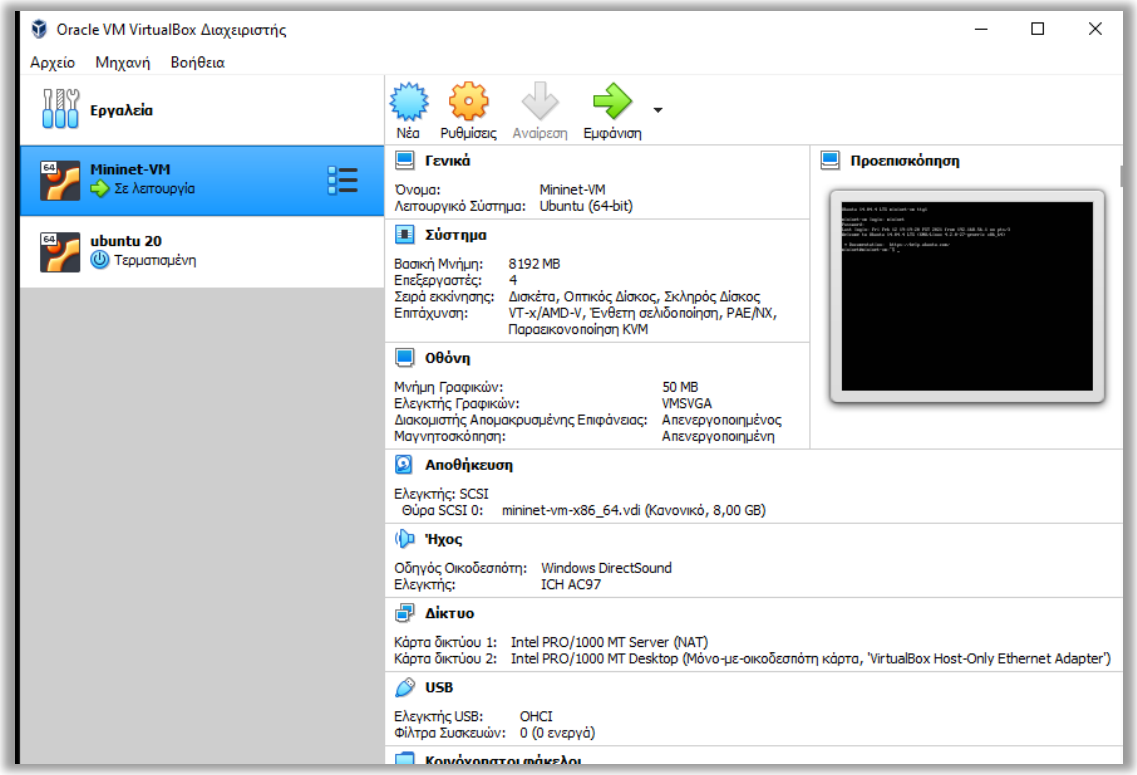
Παρόλα τα πλεονεκτήματα τα οποία έχει και παρουσιάστηκαν παραπάνω, το Mininet έχει και κάποιους περιορισμούς που πρέπει να τους λάβεις υπόψιν σου αφού αποφασίσεις να ασχοληθείς με τον συγκεκριμένο εξομοιωτή δικτύου. Η μεγαλύτερη δυσκολία είναι οι περιορισμένοι πόροι όταν γίνεται πείραμα σε ένα σύστημα. Επιπροσθέτως, όταν το λογισμικό εξαρτάται από άλλο λειτουργικό σύστημα τότε ο πυρήνας Linux θα πρέπει να εγκατασταθεί σε κάποια εικονική μηχανή στην περίπτωση μας στο Virtual Box. Επίσης, το δίκτυο του εξομοιωτή έχει περιορισμένη πρόσβαση στο διαδίκτυο και το τοπικό δίκτυο (LAN) και για αυτό το λόγο συνηθίζεται η χρήση του NAT για να επιτευχθεί σύνδεση με το τοπικό δίκτυο.

5.1.2 Εγκατάσταση του Mininet σε εικονική μηχανή

Για να γίνει η εγκατάσταση του σε εικονική μηχανή ακολουθήθηκαν τα παρακάτω βήματα.

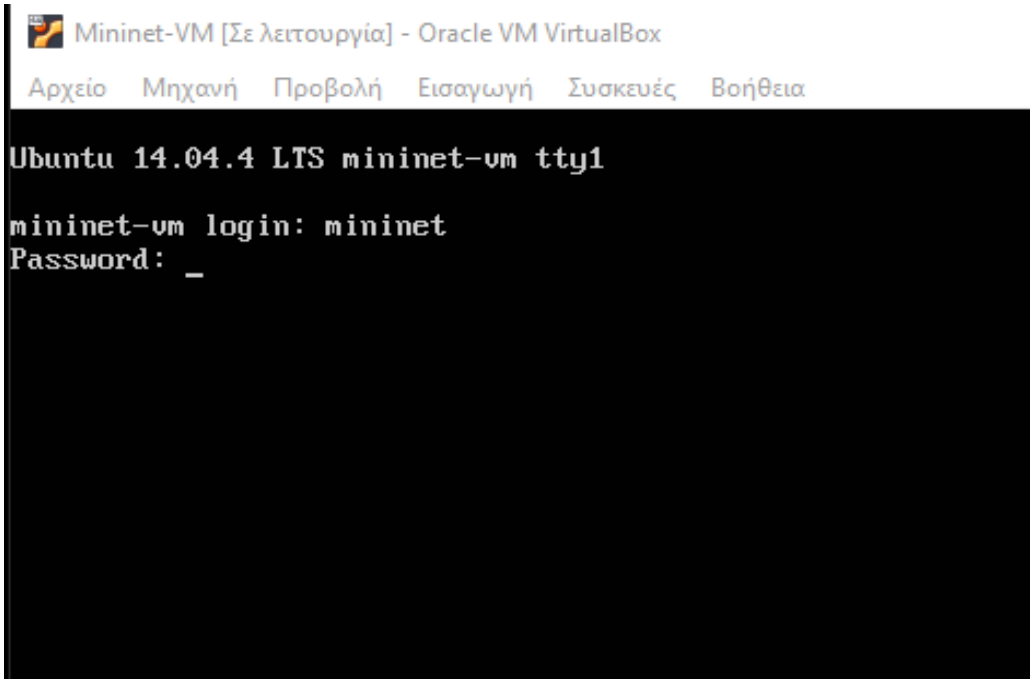
1ο βήμα: Κάνουμε download από την σελίδα https://www.virtualbox.org/ της εικονικής μηχανής VirtualBox 6.1 και κάνουμε εγκατάσταση στο λειτουργικό μας σύστημα είτε αυτό είναι Windows,είτε είναι MacOS

2ο βήμα: Κατεβάζουμε το αρχείο με την πιο πρόσφατη έκδοση του από την σελίδα https://github.com/Mininet/Mininet/wiki/Mininet-VM-Images . Το αρχείο είναι σε μορφή .zip και αφού γίνει αποσυμπίεση του το αρχείο θα έχει κατάληξη .ovf . Στην συνέχεια πάμε στην εικονική μηχανή VirtualBox και πατάμε το κουμπί εισαγωγή όπως φαίνεται και στην παρακάτω εικόνα και κάνουμε την εγκατάσταση του Mininet δίνοντας τιμές σε κάποιους παραμέτρους όπως είναι η μνήμη Ram η οποία θα δεσμεύεται καθ' όλη την εκτέλεση του, όπως επίσης και πόσους πυρήνες του εκάστοτε επεξεργαστή θα χρησιμοποιεί κ.α.



Εικόνα 5.2:Προεπισκόπηση της εικονικής μηχανής

3ο βήμα: Αφότου ολοκληρωθεί η εγκατάσταση στο βήμα 2ο και εφόσον έχουν πάει όλα καλά με την εγκατάσταση πατάμε το κουμπί εκκίνησης της μηχανής περιμένουμε να φορτώσει όλα τα περιφερειακά και στο τέλος θα βρεθούμε μπροστά σε μία μαύρη οθόνη όπως είναι στην παρακάτω εικόνα και εκεί που μας ζητάει login και password θα βάλουμε και στα δύο πεδία την λέξη Mininet



Εικόνα 5.3:Είσοδος στο Mininet

Τα παραπάνω 3 βήματα ακολουθήθηκαν για την εγκατάσταση του Mininet σε λειτουργικό σύστημα Windows με τη βοήθεια της εικονικής μηχανής VirtualBox. Για το δικό μας πειραματικό μέρος πέραν των δυο προαναφερθέντων προγραμμάτων χρησιμοποιήθηκαν και κάποια άλλα τα οποία θα παρουσιαστούν στην αμέσως επόμενή ενότητα.

5.1.3 Εντολές έναρξης και γραμμή εντολών του Mininet

Στο υποκεφάλαιο αυτό θα παρουσιαστούν οι βασικές εντολές για τη δημιουργία τοπολογιών στον εξομοιωτή Mininet [15]. Το MIninet παρέχει στον χρήστη τη δυνατότητα να δημιουργήσει μια σειρά από τοπολογίες που ενσωματώνονται στον

κώδικά του (π.χ. γραμμική, δέντρο, κ.ά.) ή να ορίσει κατά προτίμηση τις δικές του τοπολογίες.. Οι εντολές με τις οποίες δημιουργούνται οι συνηθισμένες τοπολογίες είναι οι εξής:

$\$\boldsymbol{sudo\ mn}$: Δημιουργείται η προεπιλεγμένη τοπολογία η οποία αποτελείται από έναν μεταγωγέα συνδεδεμένο σε δύο hosts.

$\$\boldsymbol{sudo\ mn - topo\ single, 5}$: Δημιουργείται η τοπολογία η οποία αποτελείται από έναν μεταγωγέα που είναι συνδεδεμένος με έναν host.

$\$\boldsymbol{sudo\ mn - topo\ linear, 6}$: Δημιουργείται η γραμμική τοπολογία η οποία αποτελείται από 6 μεταγωγείς συνδεδεμένους μεταξύ τους σε μια γραμμή και ο κάθε μεταγωγέας είναι συνδεδεμένος με έναν host.

$\$\boldsymbol{sudo\ mn - topo\ tree, 2, 3}$: Δημιουργείται η τοπολογία δέντρου όπου αποτελείται από τέσσερις μεταγωγείς και εννέα hosts. Με άλλα λόγια, ο μεταγωγέας S1 είναι συνδεδεμένος με τους άλλους 3 μεταγωγείς S2,S3,S4 και αυτοί από την μεριά τους έχει ο καθένας τους από 3 υπολογιστές

$\$\boldsymbol{sudo\ mn\ -c}$: Με την εκτέλεση αυτής της εντολής καθαρίζει το Mininet. Η εντολή αυτή χρησιμοποιείται κάθε φορά που χρειάζεται να τερματιστεί η λειτουργία κάποιας εξομοίωσης.

Εν συνεχεία θα αναφερθούν διάφορες εντολές που εκτελούνται στη γραμμή εντολών του Mininet (CLI). Οι εντολές αυτές εκτελούνται αφού πρώτα γίνει υλοποίηση του δικτύου και κάθε μια από αυτές μας δίνουν πληροφορίες για το εκάστοτε δίκτυο είτε πραγματοποιούν κάποια ενέργεια πάνω στο δίκτυο.

$mininet > help$: Εμφανίζονται οι εντολές που υποστηρίζει το CLI του Mininet

$mininet > nodes$: Εμφανίζονται οι κόμβοι του δικτύου

$mininet > net$: Εμφανίζονται οι συνδέσεις του δικτύου

$mininet > dump$: Εμφανίζονται οι πληροφορίες για όλους τους κόμβους

$mininet > xterm\ h1\ h2\ h3$: Ανοίγει τερματικό ή τερματικά κόμβων

$mininet > link\ s1\ h1\ down$: Απενεργοποιείται η σύνδεση μεταξύ του μεταγωγέα S1 και του υπολογιστή h1.

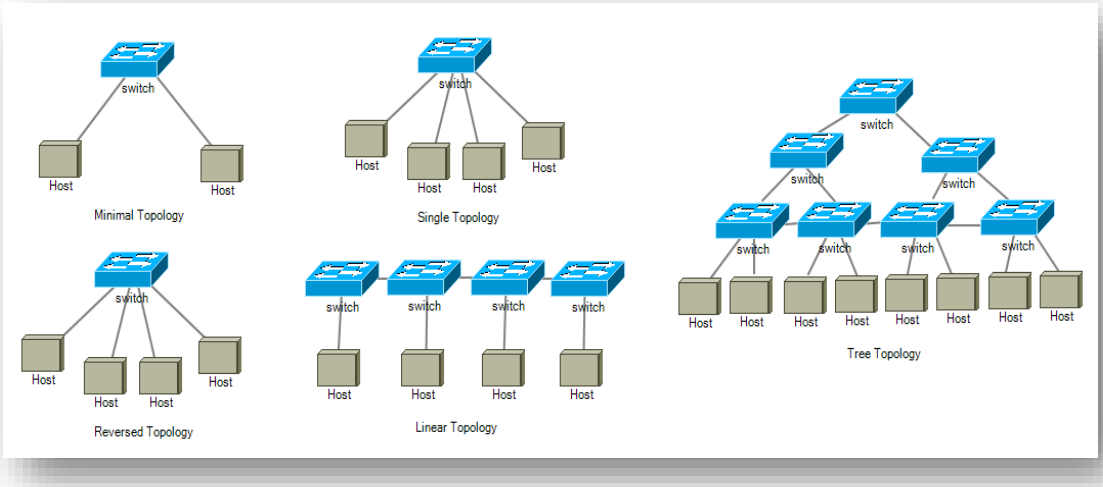
$mininet > link\ s1\ h1\ up$: Ενεργοποιείται η σύνδεση μεταξύ του μεταγωγέα S1 και του υπολογιστή h1.

5.1.4 Δημιουργία τοπολογιών στο Mininet

Στο υποκεφάλαιο αυτό θα γίνει μια μικρή αναφορά στη δημιουργία τοπολογιών στον εξομοιωτή Mininet [15] σε κάποιες προκαθορισμένες τοπολογίες όπως επίσης και σε τοπολογίες που μπορούν να παραμετροποιηθούν.

Οι πιο γνωστές και προκαθορισμένες τοπολογίες που γνωρίζουμε από τα δίκτυα και είναι εμφανής και στην παρακάτω εικόνα είναι:

- *Minimal*: είναι μια απλή τοπολογία η οποία αποτελείται από έναν OpenFlow μεταγωγέα και δύο υπολογιστές, καθώς επίσης δημιουργείται σύνδεση μεταξύ αυτών.
- *Single*: είναι τοπολογία με έναν OpenFlow μεταγωγέα και n υπολογιστές (hosts), καθώς επίσης δημιουργείται σύνδεση μεταξύ αυτών.
- *Reversed*: Έχει παρόμοια μορφή με την παραπάνω , αφού η μοναδική διαφορά είναι η σειρά σύνδεσης τους η οποία αντιστρέφεται.
- *Linear*: είναι μία τοπολογία η οποία περιέχει n μεταγωγείς και n υπολογιστές, η οποία δημιουργεί σύνδεση μεταξύ κάθε μεταγωγέα και κάθε υπολογιστή καθώς επίσης και μεταξύ των μεταγωγέων.
- *Tree*: Η δεντρική τοπολογία περιέχει n επίπεδα και k υπολογιστές που συνδέονται στο κάθε μεταγωγέα.
- *Random*: είναι το τυχαίο γράφημα από κόμβους N που συνδέονται με άκρα n που επιλέγονται τυχαία από τις πιθανές άκρες N (N-1) ^{2}.



Εικόνα 5.4: Τοπολογίες δικτύων

Για τις υπό παραμετροποίηση τοπολογίες, το μοναδικό πράγμα που είναι αναγκαίο είναι μόνο μερικές γραμμές κώδικα σε γλώσσα Python, έτσι ώστε να μπορεί να γίνει μία ευέλικτη τοπολογία και να μπορεί να ρυθμιστεί βασιζόμενη στις παραμέτρους που θα πρέπει να εισαχθούν σε αυτήν και να γίνεται επαναχρησιμοποίηση της σε διάφορα πειράματα. Η παρακάτω εικόνα μας δείχνει μια τέτοια τοπολογία η συγκεκριμένη απεικονίζει μία υπό παραμετροποίηση τοπολογία δέντρου.

```python
class GenericTree(Topo):
    """Simple topology example."""

    def build(self, depth=1, fanout=2):
        # Numbering:  h1..N, s1..M
        self.hostNum = 1
        self.switchNum = 1

    def build(self, depth=1, fanout=2):
        # Numbering:  h1..N, s1..M
        self.hostNum = 1
        self.switchNum = 1
        # Build topology
        self.addTree(depth, fanout)

    def addTree(self, depth, fanout):
        """Add a subtree starting with node n.
           returns: last node added"""
        isSwitch = depth > 0
        if isSwitch:
            node = self.addSwitch('s%s' % self.switchNum)
            self.switchNum += 1
            for _ in range(fanout):
                child = self.addTree(depth - 1, fanout)
                self.addLink(node, child)
        else:
            node = self.addHost('h%s' % self.hostNum)
            self.hostNum += 1
        return node
```

Εικόνα 5.5: Κώδικας σε Python δημιουργίας τοπολογίας δέντρου

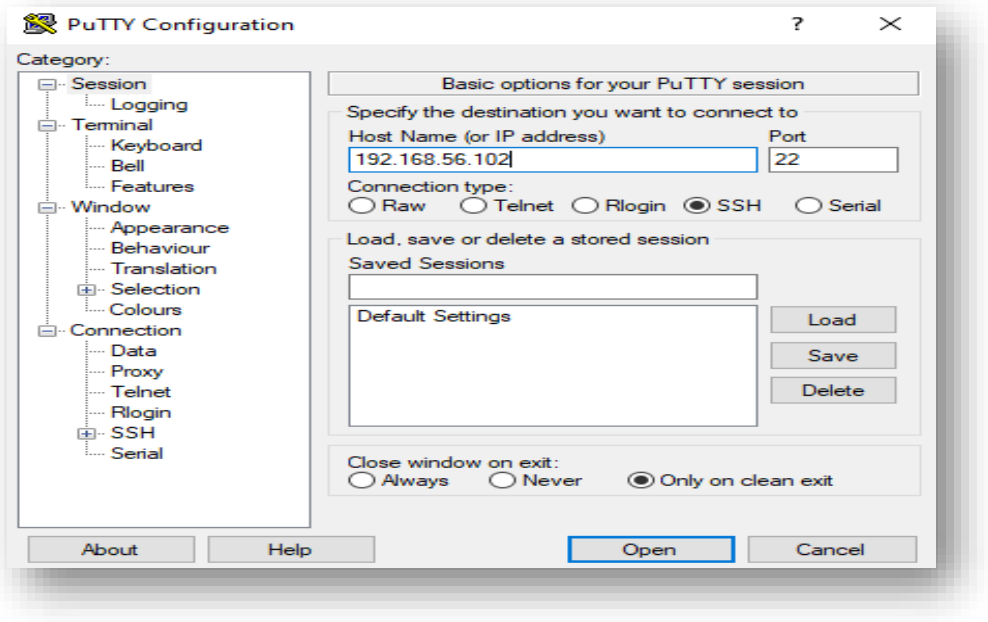
5.2 Βοηθητικά προγράμματα Xming και Putty

Για τις ανάγκες αυτής της διπλωματικής εργασίας χρειάστηκε η εγκατάσταση του διακομιστή οθόνης Xming. Το Xming μπορεί να χρησιμοποιηθεί με εφαρμογές του Secure Shell (SSH) για την ασφαλή προώθηση των συνεδριών X11 από άλλους υπολογιστές. Επίσης χρησιμοποιήθηκε το Putty το οποίο είναι ένα λογισμικό εξομοίωσης τερματικού και υποστηρίζει πολλά πρωτόκολλα δικτύου, όπως SCP, SSH, Telnet , rlogin και σύνδεση raw socket. Μπορεί επίσης να συνδεθεί σε σειριακή θύρα. Στην παρούσα διπλωματική χρειάστηκε να χρησιμοποιήσουμε το πρωτόκολλο δικτύου SSH. Στις παρακάτω εικόνες παρουσιάζεται η χρήση τους στην παρούσα εργασία.

Αρχικά έγινε η εγκατάσταση του εργαλείου Xming και συγκεκριμένα η έκδοση 6.09.031.

1ο Βήμα

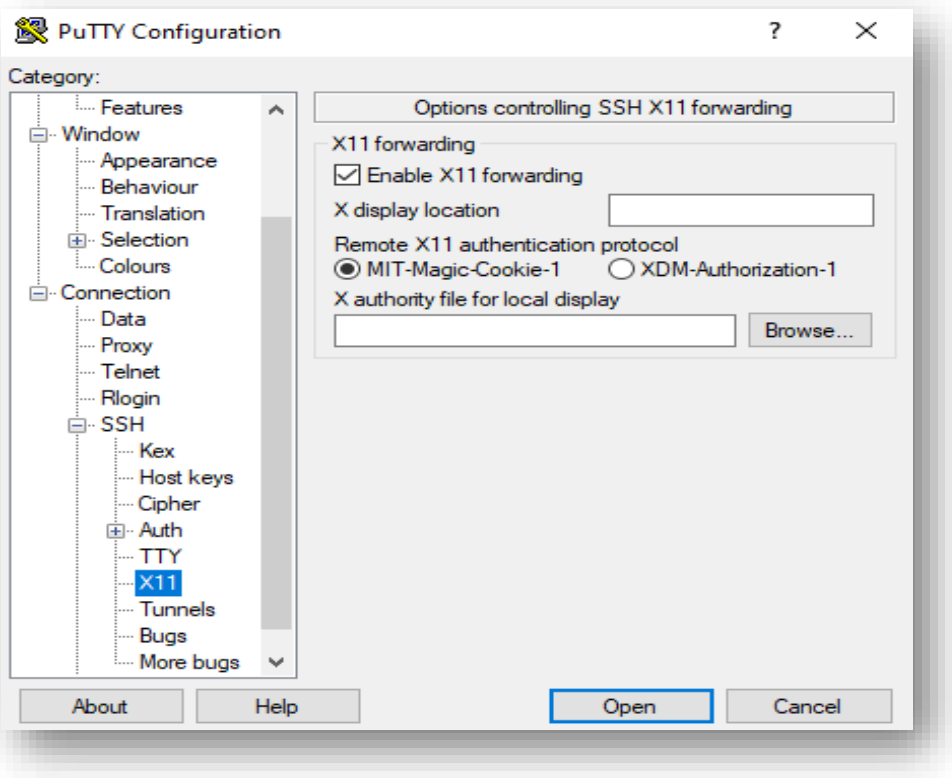
Αφού εγκατασταθεί το Putty τρέχουμε το εκτελέσιμο αρχείο Putty.exe και εμφανίζεται το παρακάτω παράθυρο όπου θα πρέπει να επιλέξουμε Connection type SSH και στην συνέχεια να βάλουμε τη διεύθυνση του local host του VirtualBox 192.168.56.102 και θύρα 22



Εικόνα 5.6:Η βασική οθόνη παραμετροποίησης του Putty

2ο βήμα

Πάμε στην αριστερή πλευρά στην καρτέλα Connection και πατάμε να αναπτυχθεί η καρτέλα SSH στην συνέχεια επιλέγουμε το X11 για να γίνει σύνδεση με τον εξομοιωτή οθόνης Xming και το κουτάκι Enable X11 Forwarding όπως φαίνεται και στην παρακάτω εικόνα. Με το κουμπί Open πλέον είναι έτοιμο προς χρήση ο εξομοιωτής τερματικού παραθύρου.



Εικόνα 5.7: Putty Configuration

5.3 Αναλυτής πρωτοκόλλων δικτύου Wireshark

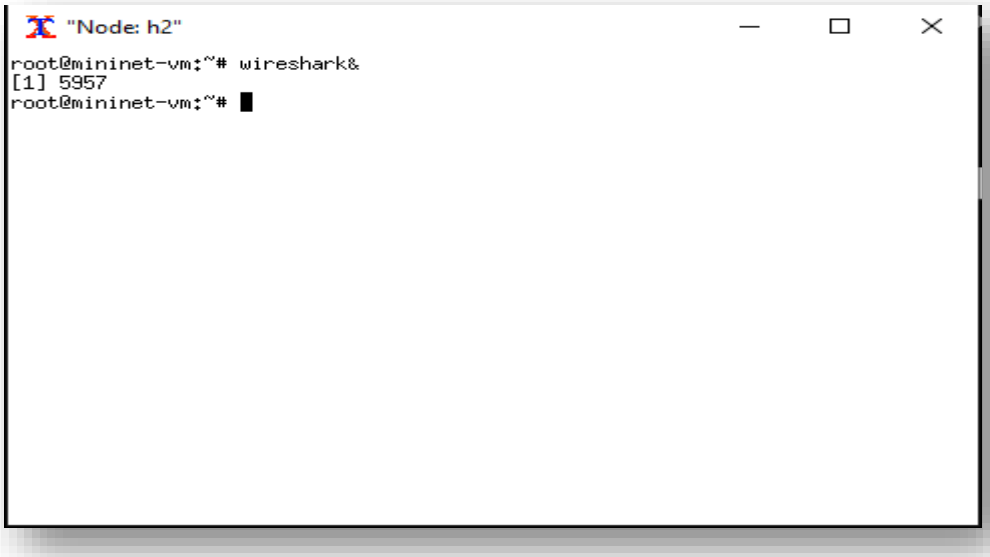
Το Wireshark [33] είναι ένα ανοιχτού κώδικα λογισμικό ανάλυσης πρωτοκόλλων δικτύου. Η κύρια χρήση του είναι η ανάλυση πακέτων του δικτύου, η παρακολούθηση, ο εντοπισμός και η αντιμετώπιση προβλημάτων στο δίκτυο. Δίνει την δυνατότητα στον χρήστη να παρακολουθεί όλη την κίνηση που δημιουργείται στο δίκτυο ανά πάσα στιγμή. Η εκτέλεση του Wireshark, στο περιβάλλον Mininet γίνεται δίνοντας στην κονσόλα που τρέχει η τοπολογία την εντολή:

$$mininet > xterm\ h1\ h2$$

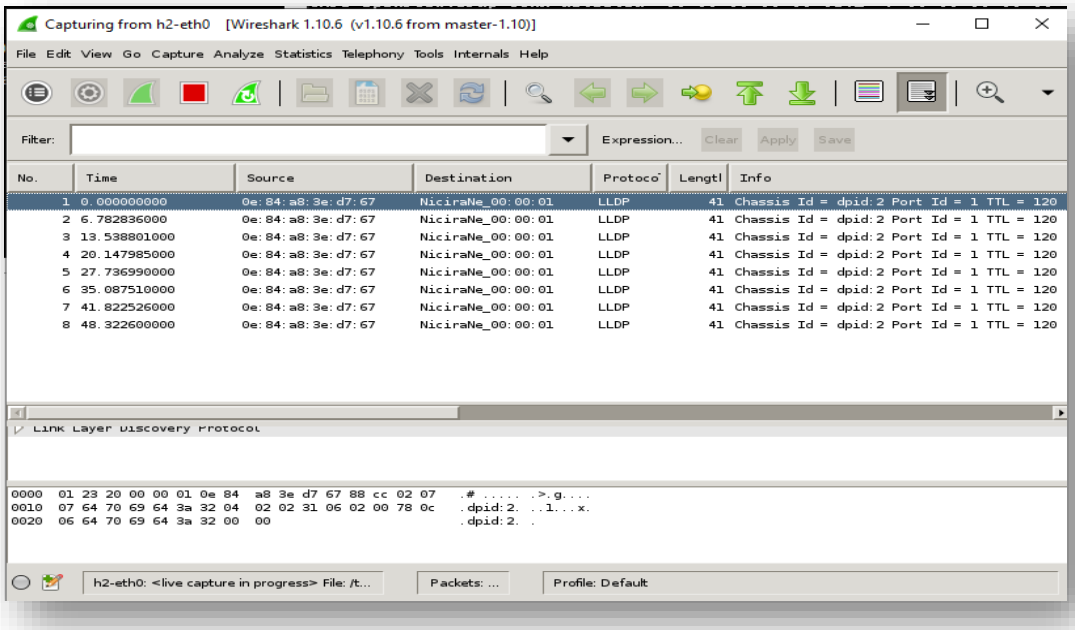
θα εμφανιστεί το παρακάτω παράθυρο της εικόνας 15 του host2 και δίνοντας την εντολή:

$$wireshark\&$$

Στην εικόνα 4.9 του αναλυτή Wireshark όπου μπορούμε να παρατηρούμε την κίνηση των πακέτων LLDP που στέλνονται ανά τακτά χρονικά διαστήματα



Εικόνα 4.8:Παράθυρο host2 με τη βοήθεια του Xming



Εικόνα 4.9: Ανάλυση των πακέτων LLDP στο host2

6.Αξιολόγηση πρωτοκόλλων OFDP, OFDPv2a, OFDPv2b με την βοήθεια του Mininet

Στην ενότητα αυτή παρουσιάζεται η πειραματική αξιολόγηση των πρωτοκόλλων OFDP,OFDPv2a και OFDPv2b που έγινε για να μελετηθεί η συμπεριφορά τους και η αποτελεσματικότητα τους.

6.1 Η μεθοδολογία της αξιολόγησης των πρωτοκόλλων

Για την αξιολόγηση χρησιμοποιήθηκε το Mininet και τα εργαλεία Xming και Putty. Το λογισμικό εγκαταστάθηκε σε επιτραπέζιο υπολογιστή με τα ακόλουθα χαρακτηριστικά 16GB μνήμη RAM, Intel(R) Core(TM) i5-2400 @ 3.10GHz CPU, storage 1TB και λειτουργικό σύστημα Windows 10 Enterprise. Η προσομοίωση έγινε σε εικονική μηχανή που εγκαταστάθηκε στον συγκεκριμένο υπολογιστή. Για την εικονική μηχανή όπου εγκαταστάθηκε το Mininet αποδόθηκαν 4 cores και μνήμη RAM 8GB. Το λογισμικό που χρησιμοποιήθηκε για την εικονική μηχανή περιλάμβανε:
Virtual Box έκδοση 6.1, λειτουργικό σύστημα Linux (Ubuntu) έκδοση 14.04.4 LTS , Miminet έκδοση GNU/Linux 4.2.0-27 generic-x86_64, POX έκδοση 0.2.0, putty έκδοση 0.74, Xming έκδοση 6.9.0.31 και Python έκδοση 2.7.
Η σύγκριση έγινε μεταξύ:

i) του τυπικού πρωτοκόλλου OFDP που χρησιμοποιείται για την ανακάλυψη της τοπολογίας στον ελεγκτή POX,
ii) του πρωτοκόλλου ανακάλυψης της τοπολογίας OFDPv2a [19],
iii) του πρωτοκόλλου ανακάλυψης της τοπολογίας OFDPv2b [19].

6.1.1 Μετρικές απόδοσης

Η αξιολόγηση των πρωτοκόλλων βασίστηκε στις παρακάτω μετρικές:

- Controller overhead: Πλήθος μηνυμάτων που ανταλλάσσει ο ελεγκτής με τους μεταγωγείς και συγκεκριμένα το πλήθος των:
    - Packet-Out: μηνύματα που στέλνει ο controller
    - Packet-In: μηνύματα που λαμβάνει ο controller από τους μεταγωγείς (data plane)

- Χρόνος ανακάλυψης της τοπολογίας: ο χρόνος που μεσολαβεί από την αποστολή του 1ου Packet-Out από τον controller μέχρι τη λήψη του τελευταίου Packet-In επίσης, από τον controller.
- Χρήση CPU: το ποσοστό του επεξεργαστή που απασχολεί ο ελεγκτής κατά τη λειτουργία του.
- Χρήση RAM: το ποσοστό της μνήμης που δεσμεύεται για τη λειτουργία του ελεγκτή.

Τροποποιήσαμε κατάλληλα τους αλγορίθμους για να μπορούμε να παίρνουμε διάφορα μέτρα σύγκρισης τους όπως για τον αριθμό πακέτων Packet-In και Packet-Out καθώς επίσης και ενός νέου μέτρου που υλοποιήθηκε του χρόνου εύρεσης της τοπολογίας του δικτύου.

Εκτός αυτών των μέτρων κρατούσαμε και τις τιμές στο κομμάτι των πόρων που κατανάλωναν οι εκάστοτε τοπολογίες, ο ελεγκτής και η χρήση της CPU και της RAM κατά την χρήση του controller κατά την διαδικασία ανακάλυψης της τοπολογίας.

6.1.2 Τοπολογίες δικτύου

Η πειραματική μελέτη έγινε για τις παρακάτω τοπολογίες:

i) γραμμική τοπολογία

ii) τοπολογία ισοζυγισμένου δέντρου και

iii) τυχαία τοπολογία

Για κάθε τοπολογία μεταβάλλεται το πλήθος των κόμβων και συγκεκριμένα εξετάστηκαν μεταξύ άλλων και τα παρακάτω δίκτυα.

Τύπος τοπολογίας	Switches	Ports
Γραμμική	5,31,121, 341	8,60,240,680
Ισοζυγισμένο Δέντρο	5,31,121, 341	8,60,240,680
Τυχαία	10,20,30,40	42,138,282,508

6.1.3 Εκτέλεση προσομοιώσεων

Η εκτέλεση των πειραμάτων έγινε σε περιβάλλον Mininet. Για την κάθε τοπολογία και για κάθε δίκτυο με συγκεκριμένο αριθμό κόμβων, έγιναν τουλάχιστον 100 εκτελέσεις

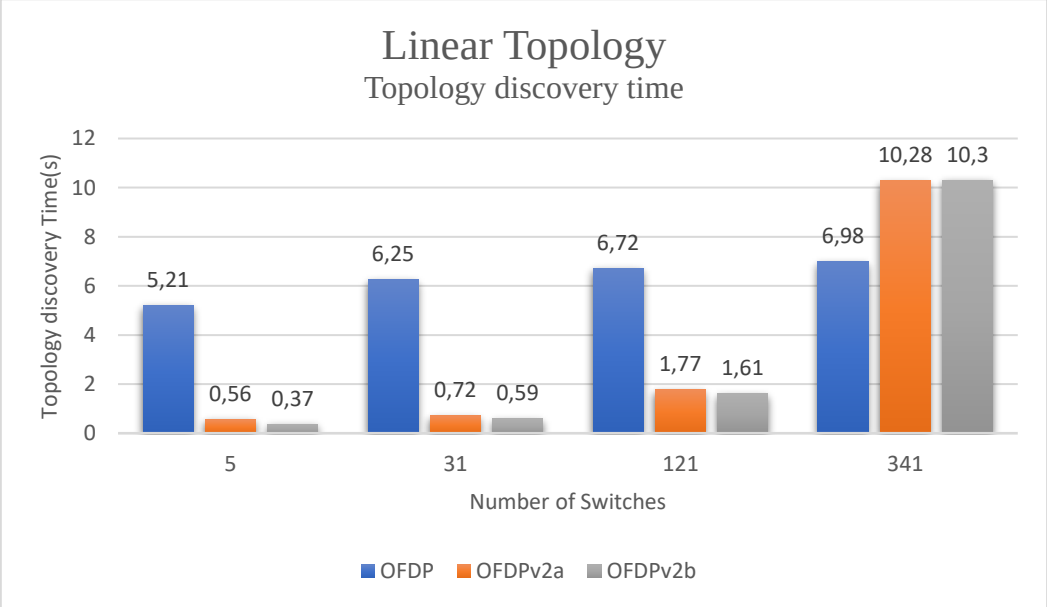
και λήφθηκε ο μέσος όρος των μετρήσεων, ώστε να ελαχιστοποιήσουμε τη πιθανότητα σφάλματος των αποτελεσμάτων. Η υλοποίηση των πρωτοκόλλων OFDP, OFDPv2a, OFDPv2b βασίστηκε στον κώδικα που παρέχεται στο [19] αφού έγιναν οι κατάλληλες τροποποιήσεις για τη λήψη των μετρήσεων.

6.2 Ανάλυση απόδοσης για γραμμική τοπολογία

Για την δημιουργία της τοπολογίας linear εκτελούμε την εντολή

$sudo mn –topo linear, N –controller=remote

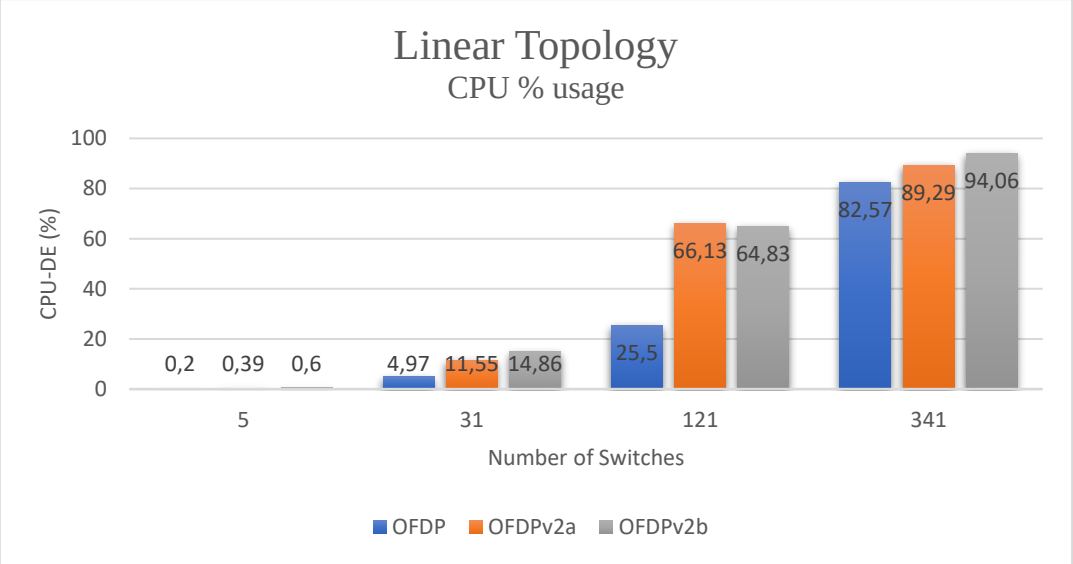
όπου N ο αριθμός των switches και λαμβάνουμε μετρήσεις για την παράμετρο Topology Discovery Time για κάθε πρωτόκολλο ανακάλυψης τοπολογίας. Στο Διάγραμμα 1 απεικονίζεται ο χρόνος ανακάλυψης της τοπολογίας για τα πρωτόκολλα OFDP, OFDPv2a και OFDPv2b, για διαφορετικό πλήθος κόμβων που κυμαίνεται από 5 έως 340. Όπως παρατηρούμε από τα αποτελέσματα ο χρόνος εντοπισμού με την εφαρμογή του πρωτοκόλλου OFDP παραμένει σχετικά σταθερός (κυμαίνεται μεταξύ 5,2 sec και 6,98 sec), ανεξάρτητα από το πλήθος των κόμβων που μετέχουν στην τοπολογία. Όμως, με την εφαρμογή του OFDPv2, και για τις δύο παραλλαγές του, ο χρόνος εντοπισμού αυξάνεται και μάλιστα εκθετικά όταν το πλήθος των κόμβων αυξάνεται.



Διάγραμμα 1: Απεικόνιση του topology discovery time στην γραμμική τοπολογία

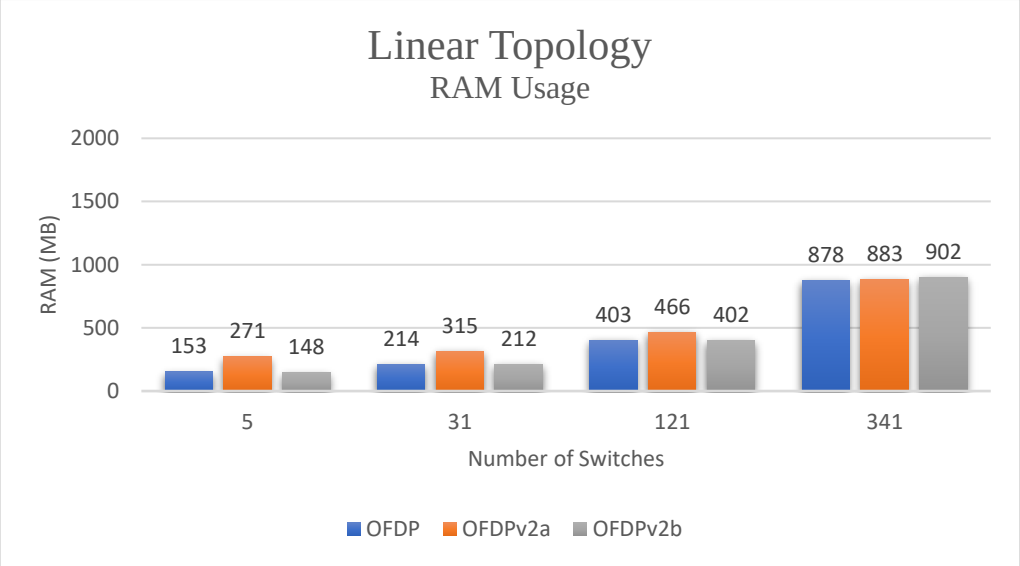
Στο Διάγραμμα 2 απεικονίζεται το ποσοστό χρήσης της CPU που χρησιμοποιείται από τον ελεγκτή κατά την διάρκεια ανακάλυψης της γραμμικής τοπολογίας για τους τρεις

αλγορίθμους που υλοποιήθηκαν. Παρατηρούμε ότι το πρωτόκολλο OFDP απασχολεί λιγότερο τον επεξεργαστή για μικρό αριθμό κόμβων σε αντίθεση με τα πρωτόκολλα OFDPv2-A, OFDPv2-B. Όμως, όσο αυξάνεται το μέγεθος του δικτύου, η συμπεριφορά των πρωτοκόλλων ως προς το ποσοστό χρήσης της CPU συγκλίνει.



Διάγραμμα 2: Απεικόνιση της χρήσης CPU στη γραμμική τοπολογία

Στο Διάγραμμα 3 απεικονίζεται το ποσοστό χρήσης της RAM που χρησιμοποιείται κατά τη διάρκεια ανακάλυψης της γραμμικής τοπολογίας για τους τρεις αλγορίθμους που υλοποιήθηκαν. Γίνεται εύκολα κατανοητό ότι για μικρό αριθμό μεταγωγέων ο αλγόριθμος OFDPv2a χρησιμοποιεί περισσότερη RAM σε σχέση με τους άλλους δύο αλγορίθμους ενώ όσο αυξάνεται το πλήθος των switches υπάρχει μια σταθεροποίηση των τιμών της χρήσης RAM και από για τους τρεις αλγορίθμους και κυμαίνονται στα ίδια επίπεδα.



Διάγραμμα 3: Απεικόνιση της χρήσης RAM στην γραμμική τοπολογία

Από τα παραπάνω αποτελέσματα διαπιστώνουμε ότι για τη γραμμική τοπολογία, η συμπεριφορά των πρωτοκόλλων είναι παρόμοια ως προς την χρήση της CPU και της μνήμης RAM, ακόμη και όταν ο αριθμός των κόμβων αυξάνεται. Όμως, αυτό δεν ισχύει στην περίπτωση που εξετάζεται ο χρόνος ανακάλυψης της τοπολογίας. Στην περίπτωση του κλασικού OFDP ο χρόνος ανακάλυψης παραμένει σχετικά σταθερός ενώ για τις δύο παραλλαγές του OFDP, ο χρόνος ανακάλυψης αυξάνει εκθετικά και κυμαίνεται από 0.5sec έως περίπου 10sec, όταν ο αριθμός των switches μεταβάλλεται από 5 έως 340 αντίστοιχα. Είναι ολοφάνερο πως ο σε αυτή την περίπτωση ο χρόνος εντοπισμού της τοπολογίας είναι αρκετά μεγαλύτερος σε σχέση με τον χρόνο που απαιτεί το πρωτόκολλο OFDP.

6.3 Ανάλυση απόδοσης για ισοζυγισμένο δέντρο

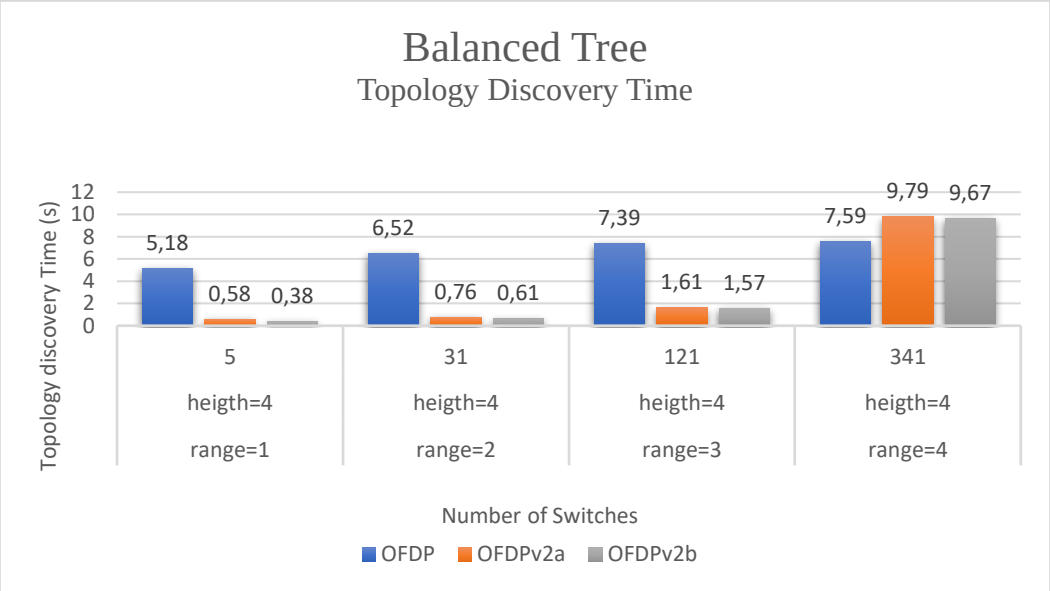
Για την δημιουργία της τοπολογίας ισοζυγισμένου δέντρου εκτελούμε την εντολή

$ sudo mn --custom mn_nx_topos.py --topo balanced_tree,r=2,h=2–

controller=remote

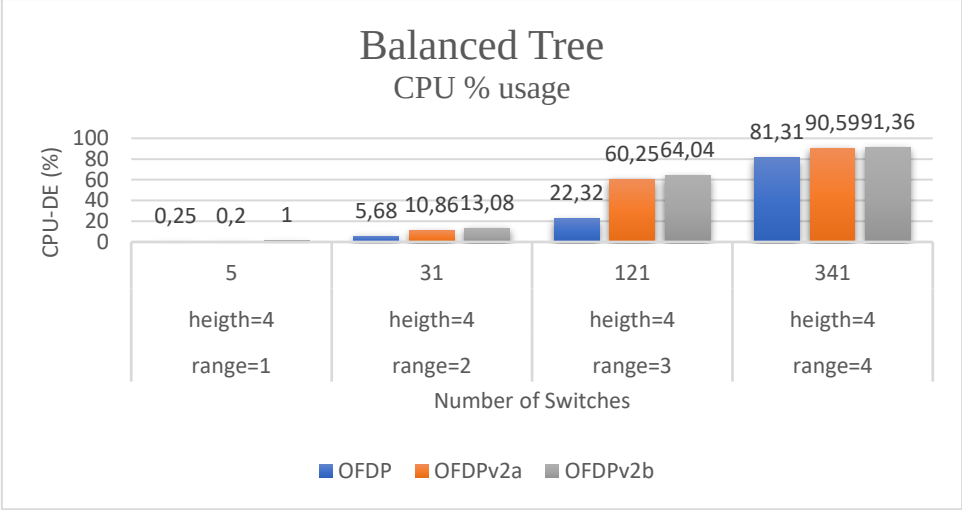
Το r αντιστοιχεί στο range (εύρος) και h το height (ύψος) του δέντρου αντίστοιχα. Οι τιμές του r κυμάνθηκαν από 1 έως 4 κρατώντας κάθε φορά σταθερό το ύψος στην τιμή 4 και λαμβάνουμε μετρήσεις για το Topology Discovery Time για κάθε πρωτόκολλο ανακάλυψης τοπολογίας. Στο Διάγραμμα 4 απεικονίζεται ο χρόνος ανακάλυψης της

τοπολογίας για τα πρωτόκολλα OFDP, OFDPv2a και OFDPv2b, για διαφορετικό πλήθος κόμβων που κυμαίνεται από 5 έως 340. Όπως παρατηρούμε από τα αποτελέσματα ο χρόνος εντοπισμού με την εφαρμογή του πρωτοκόλλου OFDP παραμένει σχετικά σταθερός (κυμαίνεται μεταξύ 5,18 sec και 7,59 sec), ανεξάρτητα από το πλήθος των κόμβων που μετέχουν στην τοπολογία. Όμως, με την εφαρμογή του OFDPv2, και για τις δύο παραλλαγές του, ο χρόνος εντοπισμού αυξάνεται και μάλιστα εκθετικά όταν το πλήθος των κόμβων αυξάνεται.



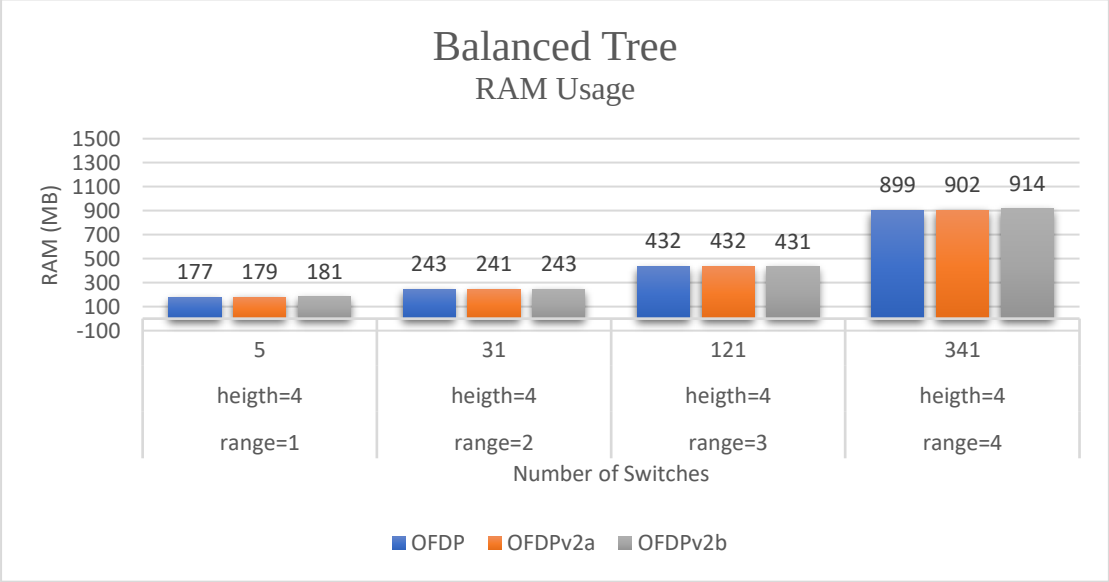
Διάγραμμα 4: Απεικόνιση του topology discovery time στην τοπολογία ισοζυγισμένου δέντρου

Στο Διάγραμμα 5 απεικονίζεται το ποσοστό χρήσης της CPU που χρησιμοποιείται από τον ελεγκτή κατά την διάρκεια ανακάλυψης της τοπολογίας του ισοζυγισμένου δέντρου για τους τρεις αλγορίθμους που υλοποιήθηκαν. Παρατηρούμε ότι ο OFDP δεν επιβαρύνει τόσο πολύ τον επεξεργαστή όσο γίνεται στην περίπτωση των OFDPv2a, OFDPv2b. Βέβαια, και εδώ παρατηρούμε μια ανάλογη συμπεριφορά με αυτή της γραμμικής τοπολογίας.



Διάγραμμα 5: Απεικόνιση της χρήσης CPU στη τοπολογία ισοζυγισμένου δέντρου

Στο Διάγραμμα 6 απεικονίζεται το ποσοστό χρήσης της RAM που χρησιμοποιείται κατά τη διάρκεια ανακάλυψης της τοπολογίας ισοζυγισμένου δέντρου για τους τρεις αλγορίθμους που υλοποιήθηκαν. Γίνεται εύκολα κατανοητό ότι όσο αυξάνεται το πλήθος των switches υπάρχει μια σταθεροποίηση των τιμών της χρήσης RAM και από τους τρεις αλγορίθμους και κυμαίνονται στα ίδια επίπεδα χρήσης.



Διάγραμμα 6: Απεικόνιση της χρήσης RAM στην τοπολογία ισοζυγισμένου δέντρου

Από τα παραπάνω αποτελέσματα διαπιστώνουμε ότι για την τοπολογία ισοζυγισμένου δέντρου, η συμπεριφορά των πρωτοκόλλων είναι παρόμοια ως προς την χρήση της μνήμης RAM, ακόμη και όταν ο αριθμός των κόμβων αυξάνεται. Όμως, αυτό δεν

ισχύει στην περίπτωση που εξετάζεται ο χρόνος ανακάλυψης της τοπολογίας. Στην περίπτωση του κλασικού OFDP ο χρόνος ανακάλυψης παραμένει σχετικά σταθερός ενώ στις δύο παραλλαγές του ο χρόνος ανακάλυψης, αν και αρχικά κυμαίνεται σε πολύ χαμηλές τιμές για μικρό πλήθος κόμβων, αυξάνεται όσο αυξάνεται ο αριθμός των κόμβων. Μάλιστα παρατηρούμε ότι ξεπερνάει κατά πολύ τον αντίστοιχο χρόνο του κλασικού OFDP. Όσον αφορά την χρήση CPU, στον OFDP δεν υπερφορτώνεται τόσο πολύ ο επεξεργαστής όσο στις περιπτώσεις των OFDPv2a, OFDPv2b.

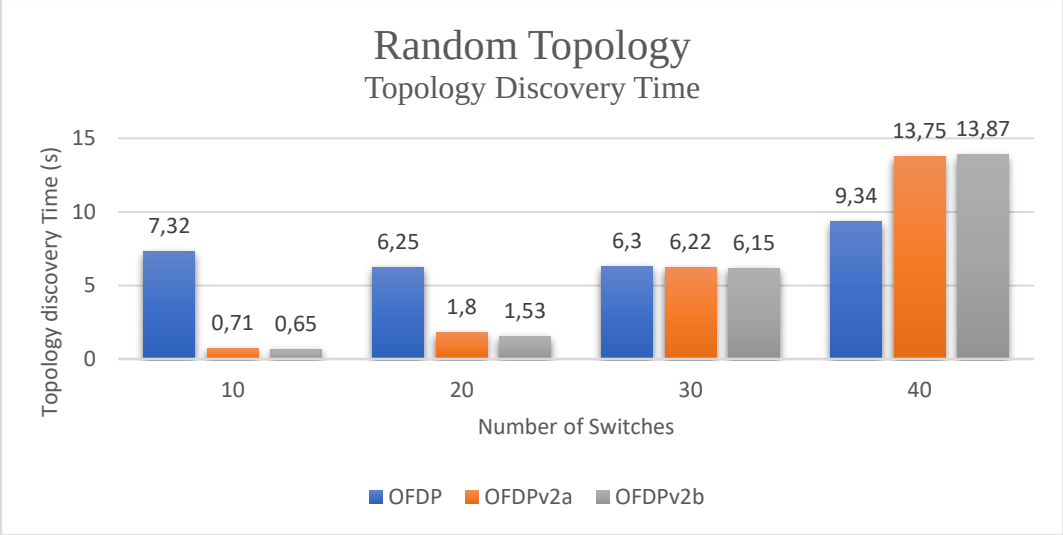
6.4 Ανάλυση απόδοσης τυχαίας τοπολογίας erdos-renyi

Για την δημιουργία της τυχαίας τοπολογίας εκτελούμε την εντολή

$sudo mn --custom mn_nx_topos.py --topo erdos_renyi,n,p=0.6 –

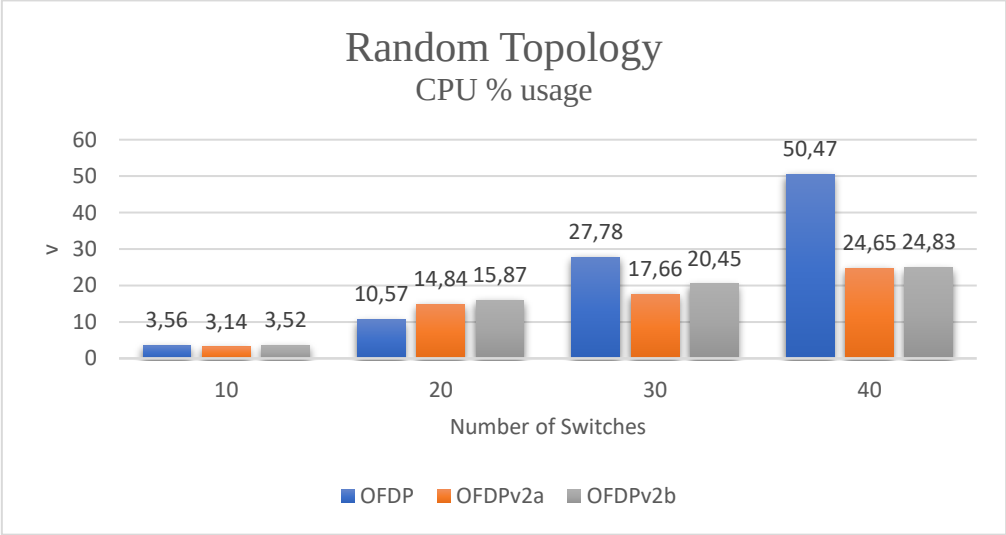
controller=remote

όπου n ο αριθμός των κόμβων και το p μια παράμετρος που παίρνει τιμές από 0 έως 1 και εκφράζει την πιθανότητα ύπαρξης μια ακμής του γράφου από το σύνολο των διαθέσιμων ακμών (των πιθανών n(n-1)/2 ακμών) . Στο Διάγραμμα 7 απεικονίζεται ο χρόνος ανακάλυψης της τοπολογίας για τα πρωτόκολλα OFDP, OFDPv2a και OFDPv2b, για διαφορετικό πλήθος κόμβων που κυμαίνεται από 10 έως 40. Όπως παρατηρούμε από τα αποτελέσματα ο χρόνος εντοπισμού με την εφαρμογή του πρωτοκόλλου OFDP παραμένει σχετικά σταθερός στις τρείς πρώτες περιπτώσεις για 10,20 και 30 κόμβους (κυμαίνεται μεταξύ 6,25 sec και 7,32 sec), ενώ για 40 κόμβους βλέπουμε μια αύξηση της τάξεως των 3 sec. Εφαρμόζοντας τον OFDPv2, και για τις δύο παραλλαγές του, ο χρόνος εντοπισμού αυξάνεται και μάλιστα εκθετικά καθώς το πλήθος των κόμβων αυξάνεται.



Διάγραμμα 7: Απεικόνιση του topology discovery time στην τυχαία τοπολογία γράφου

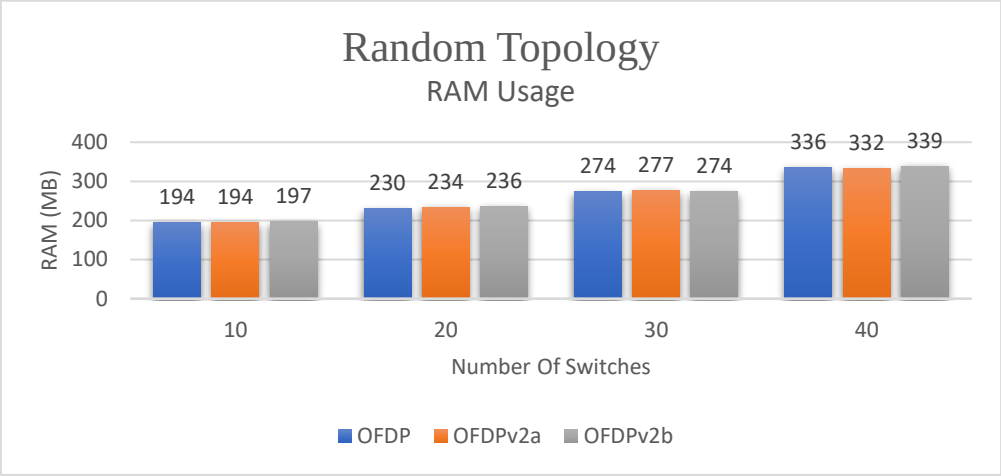
Στο Διάγραμμα 8 απεικονίζεται το ποσοστό χρήσης της CPU που χρησιμοποιεί ο ελεγκτής κατά την διάρκεια ανακάλυψης της τυχαίας τοπολογίας για τους τρεις αλγορίθμους που υλοποιήθηκαν. Παρατηρούμε ότι ο OFDP σε αυτή την περίπτωση είναι ο αλγόριθμος που επιβαρύνει πολύ τον επεξεργαστή μας ενώ γίνεται καλύτερη χρήση της CPU στις περιπτώσεις των OFDPv2a, OFDPv2b



Διάγραμμα 8: Απεικόνιση της χρήσης CPU στην τυχαία τοπολογία γράφου

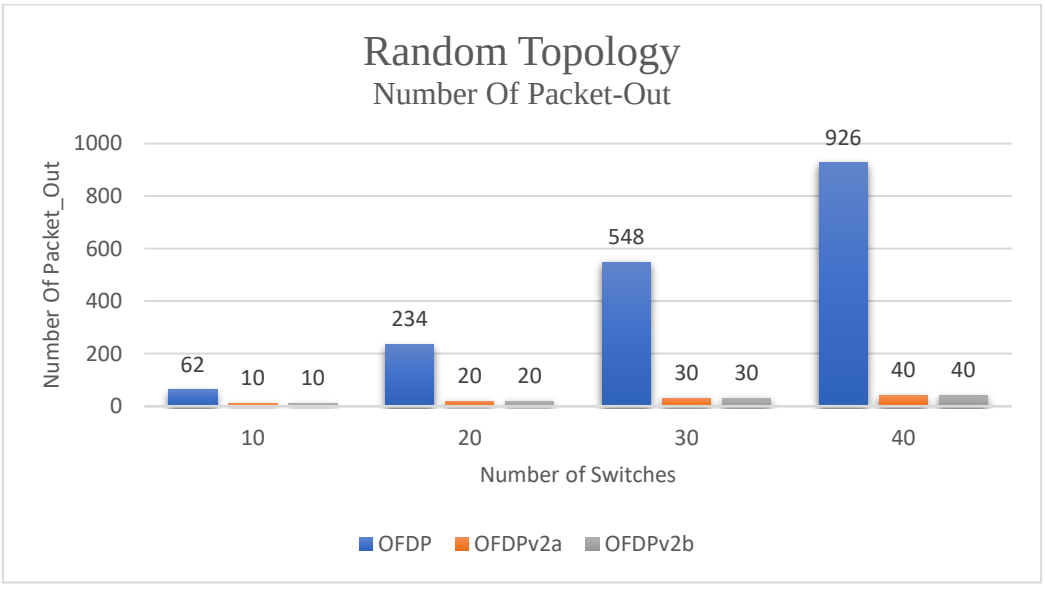
Στο Διάγραμμα 9 απεικονίζεται το ποσοστό χρήσης της RAM που χρησιμοποιείται κατά τη διάρκεια ανακάλυψης της τυχαίας τοπολογίας για τους τρεις αλγορίθμους που υλοποιήθηκαν. Γίνεται εύκολα κατανοητό ότι όσο μεγαλώνει η τιμή των κόμβων

υπάρχει μια σταθεροποίηση των τιμών της χρήσης RAM και από τους 3 αλγορίθμους και κυμαίνονται στα ίδια επίπεδα χρήσης.



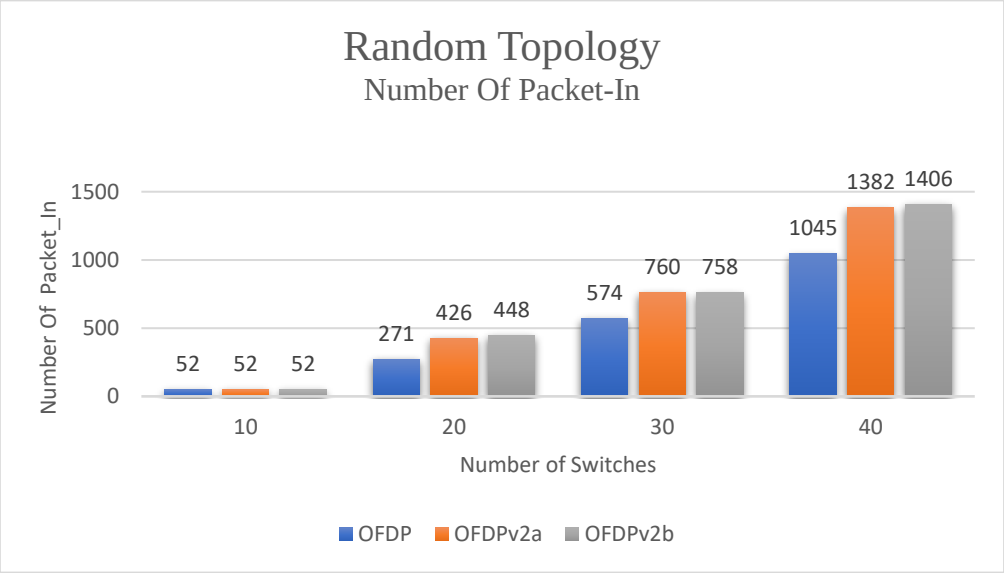
Διάγραμμα 9: Απεικόνιση της χρήσης RAM στην τυχαία τοπολογία γράφου

Στο Διάγραμμα 10 απεικονίζεται ο αριθμός των Packet-Out που χρησιμοποιείται κατά την διάρκεια ανακάλυψης της τυχαίας τοπολογίας για τους τρεις αλγορίθμους που υλοποιήθηκαν. Παρατηρούμε την βελτίωση που έχουν επιτύχει οι ερευνητές των OFDPv2a και OFDPv2b σε σχέση με τον αρχικό OFDP. Βλέπουμε ότι ο αριθμός των πακέτων αυξάνεται με εκθετική πορεία ενώ στα OFDPv2a και OFDPv2b πρωτόκολλα ο αριθμός αυτός είναι ίδιος με τον αριθμό των switches κάθε φορά.



Διάγραμμα 10: Απεικόνιση των αριθμών των Packet-Out στην τυχαία τοπολογία

Στο Διάγραμμα 11 απεικονίζεται ο αριθμός των Packet-In που χρησιμοποιείται κατά την διάρκεια ανακάλυψης της τυχαίας τοπολογίας για τους τρεις αλγορίθμους που υλοποιήθηκαν. Παρατηρούμε σε αυτή την περίπτωση ότι η αλλαγή που έχει γίνει σε σχέση με τον αρχικό OFDP στους OFDPv2a και OFDPv2b, έχει επηρεάσει τον αριθμό των πακέτων packet_in στον οποίο βλέπουμε ότι είναι κατά πολύ αυξημένος.

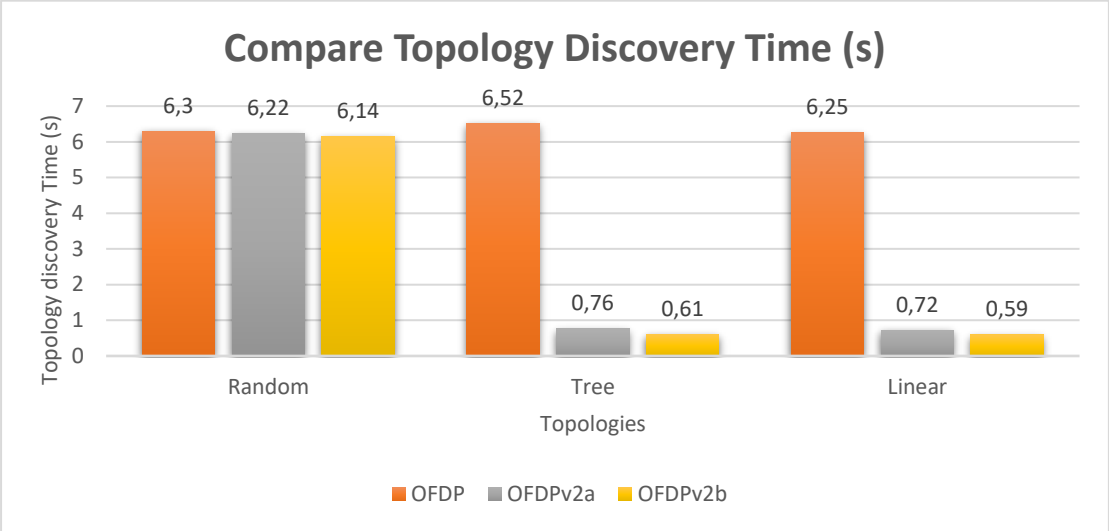


Διάγραμμα 11: Απεικόνιση των αριθμών των Packet_In στην τυχαία τοπολογία

Από τα παραπάνω αποτελέσματα διαπιστώνουμε ότι για την τυχαία τοπολογία, η συμπεριφορά των πρωτοκόλλων είναι παρόμοια ως προς την χρήση της μνήμης RAM, ακόμη και όταν ο αριθμός των κόμβων αυξάνεται. Όμως, αυτό δεν ισχύει στην περίπτωση που εξετάζεται ο χρόνος ανακάλυψης της τοπολογίας καθώς και ο αριθμός των Packet-Out, Packet-In και της χρήσης CPU. Στην περίπτωση του κλασικού OFDP ο χρόνος ανακάλυψης παραμένει σχετικά σταθερός ενώ στις 2 παραλλαγές του ο χρόνος ξεκινάει από πολύ χαμηλά για μικρές τιμές κόμβων, αλλά όσο αυξάνεται ο αριθμός των κόμβων και φτάνει στις μέγιστες τιμές του (30 και 40 κόμβοι) ανεβαίνει και ξεπερνάει κατά πολύ αυτόν του κλασικού OFDP. Επίσης, ο αριθμός των Packet-Out στον OFDP είναι πολύ μεγάλος κάτι που επιβεβαιώνει και την θεωρία μας, αφού τα OFDPv2a, OFDPv2b πετυχαίνουν να μειώσουν τον αριθμό αυτό και να γίνει ίδιος με τον αριθμό των κόμβων που μετέχουν στην εκάστοτε τοπολογία. Όσον αφορά τα Packet-In βλέπουμε μια αντιστροφή στους αλγορίθμους δηλαδή ο OFDP έχει τον μικρότερο αριθμό Packet-In σε σχέση με τους OFDPv2a, OFDPv2b οι οποίες ξεπερνάνε κατά πολύ τον αριθμό των πακέτων Packet-In σε συνάρτηση με τον κλασικό OFDP. Κλείνοντας, παρατηρούμε ότι στην τυχαία τοπολογία επιτυγχάνεται χαμηλή

χρήση της CPU για τους αλγορίθμους OFDPv2a, OFDPv2b όσο αυξάνεται ο αριθμός των κόμβων ενώ στον OFDP γίνεται μεγαλύτερη η επιβάρυνση της.

6.5 Σύγκριση του χρόνου ανακάλυψης της τοπολογίας για τις τοπολογίες γραμμική, δέντρου και τυχαία



Διάγραμμα 12: Σύγκριση του χρόνου ανακάλυψης της τοπολογίας για τις τρεις τοπολογίες και για αριθμό κόμβων ίσο με 30

Στο Διάγραμμα 12 γίνεται μια σύγκριση των χρόνων ανακάλυψης της τοπολογίας. Εξετάζουμε τις τρεις τοπολογίες (Random, Balanced Tree, Linear), για τα τρία διαφορετικά πρωτόκολλα και για αριθμό κόμβων ίσο με 30. Μετά από αυτή την σύγκριση και όπως είναι εύκολα κατανοητό στη τυχαία τοπολογία το πλεονέκτημα των αλγορίθμων OFDPv2a και OFDPv2b δεν βοηθάει στο να μειωθεί ο χρόνος ανακάλυψης της τοπολογίας. Όπως παρατηρούμε ο χρόνος και για τα τρία πρωτόκολλα είναι κατά μέσο όρο ο ίδιος αφού κυμαίνονται από 6,14 έως 6,3. Από την άλλη μεριά, όπως έχουμε αναφέρει και παραπάνω για μικρό αριθμό κόμβων οι αλγόριθμοι OFDPv2a και OFDPv2b για τις τοπολογίες του ισοζυγισμένου δέντρου και της γραμμικής αποδίδουν άριστα και μειώνουν τον χρόνο της ανακάλυψης της τοπολογίας.

7. Συμπεράσματα της διπλωματικής

Η αρχιτεκτονική δικτύου που καθορίζεται από λογισμικό εξακολουθεί να είναι μια νέα μέθοδος για τη δημιουργία και διαχείριση δικτύων υπολογιστών, τα οποία εξακολουθούν να αναπτύσσονται και να δοκιμάζονται. Στο νέο μοντέλο δικτύωσης, διαχωρίζεται το επίπεδο ελέγχου από το επίπεδο δεδομένων. Στο επίπεδο ελέγχου τοποθετείται ένας ή περισσότεροι ελεγκτές που διαχειρίζονται τις λειτουργίες του δικτύου. Ο ελεγκτής (π.χ. POX, Beacon ή Floodlight) έχει καθολική εικόνα για τη διαμόρφωση (τοπολογία) του δικτύου που την αποκτά εφαρμόζοντας πρωτόκολλα ανακάλυψης τοπολογίας. Η ανακάλυψη της τοπολογίας είναι καθοριστική και κρίσιμη για τη λειτουργία του δικτύου και τις υπηρεσίες που παρέχει και αφορούν:

- τη δρομολόγηση,
- τη διαχείριση δικτύου,
- την κατανομή πόρων και διαμόρφωση,
- την ποιότητα υπηρεσίας (QoS),
- τη διάγνωση και αποκατάσταση σφαλμάτων.

Σε αυτή τη διαδικασία, ο χρόνος ανακάλυψης της τοπολογίας και το overhead που εισάγει το πρωτόκολλο ανακάλυψης, είναι παράγοντες θεμελιώδους σημασίας καθώς α)επηρεάζει την ταχύτητα λήψης αποφάσεων ή αποστολής απαντήσεων που αφορούν γεγονότα σε πραγματικό χρόνο , και β) πρόκειται για διαδικασία που τρέχει συνεχώς στο παρασκήνιο. Και οι δυο αυτές παράμετροι επηρεάζουν την αξιοπιστία και την αποτελεσματικότητα του δικτύου.

Στα πλαίσια αυτής της διατριβής, εξετάσαμε το βασικό πρωτόκολλο OFDP που χρησιμοποιείται για το σκοπό αυτό από όλους τους ελεγκτές. Ο ελεγκτής ανακαλύπτει την τοπολογία του δικτύου συλλέγοντας πληροφορίες μέσω των πακέτων που φθάνουν σε αυτόν σε απάντηση των Packet-Out που έχει ήδη στείλει. Επειδή η τεχνική αυτή έχει αρκετά μειονεκτήματα, όπως ο μεγάλος αριθμός, ή έλλειψη αποδοτικότητας, αξιοπιστίας και ασφάλειας, οι ερευνητές προτείνουν νέα πρωτόκολλα που θα ξεπεράσουν τα ζητήματα αυτά. Εδώ παρουσιάζονται συγκριτικά τα πρωτόκολλα OFDPv2, sOFTDP, Self-Healing, TEDP, eTDP, OFDPp, Lightweight, Im-OFDP, HDDP και διαπιστώνουμε ότι λύνουν επί μέρους προβλήματα και όχι συνολικά.

Τέλος, βασιζόμενοι στο εργαλείο Mininet, εξετάζουμε πειραματικά την απόδοση τριών πρωτοκόλλων, των OFDP, OFDPv2a και OFDPv2b για τοπολογίες όπως η γραμμική,

ισοζυγισμένου δέντρου και τυχαίου γράφου. Τα αποτελέσματα δείχνουν καλύτερη συμπεριφορά του OFDP ως προς το χρόνο ανακάλυψης της τοπολογίας, όχι όμως κι ως προς τον αριθμό των παραγόμενων Packet-Out μηνυμάτων.

Αυτές οι διαπιστώσεις αλλά και η συνολική έρευνα αποτελούν κίνητρο για περαιτέρω έρευνα και μελλοντική εργασία. Συγκεκριμένα τα ερωτήματα που προκύπτουν και στα οποία διερευνούμε την λύση τους αφορούν στον τρόπο α)μείωσης τόσο των Packet-Out όσο και των Packet-In μηνυμάτων, και β) στην ελαχιστοποίηση του χρόνου ανακάλυψης της τοπολογίας ή των πιθανών αλλαγών της.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] A. K. Saha, K. Sambyo, and C. Bhunia, "Topology discovery, loop finding and alternative path solution in POX controller," *in Proceedings of the International MultiConference of Engineers and Computer Scientists*, 2016.

[2] T. Alharbi, M. Portmann, and F. Pakzad, "The (in) security of topology discovery in software defined networks," *in Local Computer Networks Conference on (LCN), 2015 IEEE 40th*, pp. 502-505.

[3] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, pp. 27-51, 2015.

[4] D. F. Macedo, D. Guedes, L. F. M. Vieira, M. A. M. Vieira, and M. Nogueira, "Programmable networks--from software-defined radio to software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, pp. 1102-1125, 2015.

[5] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in software defined networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 17, pp. 2317-2346, 2015.

[6] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," *IEEE Communications Surveys & Tutorials*, vol. 18, pp. 623-654, 2016.

[7] S. Hong, et al. "Poisoning network visibility in software-defined networks: New Attacks and Countermeasures, "NDSS. 2015.

[8] P. Congdon, (2002). Link layer discovery protocol and MIB.[online] Available from: https://www.ieee802.org/1/files/public/docs2002/lldp-protocol-00.pdf (Accessed 25 October 2020)

[9] F. Pakzad, et al., "Efficient topology discovery in software defined networks," *Signal Processing and Communication Systems (ICSPCS)*, 2014 *8th International Conference on. IEEE*, 2014.

[10] D. Kreutz, et al., "Software-defined networking: A comprehensive survey," *proceedings of the IEEE 103.1*, 2015: 14-76.

[11] M. Jammal, et al., "Software defined networking: State of the art and research challenges," *Computer Networks 72*, 2014: 74-98.

[12] O. Wendell, (2020) Introduction to Controller-Based Networking [online]. Available from: https://www.ciscopress.com/articles/article.asp?p=2995354&seqNum=2 (Accessed 21 January 2021)

[13] S. Khan, A. Gani,, Wahab, A.W.A., M. Guizani, and M.K. Khan, 2016. "Topology discovery in software defined networks: Threats, taxonomy, and state-of-the-art," *IEEE Communications Surveys & Tutorials*, 19(1), pp.303-324.

[14] V. Jain, V. Yatri, Kanchan, and C. Kapoor, (2019). " Software defined networking: State-of-the-art, " *Journal of High Speed Networks,* 25(1), 1–40. doi:10.3233/jhs-190601

[15] Mininet Wiki. (2021, September) "*Introduction to Mininet*", [Online] Available: https://github.com/Mininet/Mininet/wiki/Introduction-to-Mininet

[16] G. Tarnaras, F. Athanasiou, and S. Denazis, "Efficient topology discovery algorithm for software-defined networks," *IET Netw.*, vol. 6, no. 6, pp. 157-161, Nov. 2017.

[17] L. Ochoa-Aday, C. Cervello-Pastor, and A.Fernandez-Fernandez, ”Current trends of topology discovery in OpenFlow-based software defined networks.” 2015.

[18] B. Astuto, A. Nunes et al., "A survey of software-defned networking: past, present, and future of programmable networks, " *in IEEE Communications Surveys & Tutorials*, Jan. 2014.

[19] F. Pakzad,, M. Portmanna, W. L. Tanb, and J. Indulskaa, "Efficient topology discovery in OpenFlow-based software defined networks," *Computer Communications* doi: 10.1016/j.comcom.2015.09.013

[20] A. Azzouni, R. Boutaba, N. Thi Mai Trang, and G. Pujolle, " sOFTDP: secure and efficient OpenFlow topology discovery protocol," *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*. doi:10.1109/noms.2018.8406229

[21] L. Ochoa-Aday, C. Cervelló-Pastor , and A. Fernández-Fernández, "Self-healing topology discovery protocol for software-defined networks, " *Ieee Communication Letters*, Vol. 22, No. 5, May 2018

[22] E. Rojas, J. Alvarez-Horcajo, I. Martinez-Yelmo, J. A. Carral, and J. M.

Arco, "TEDP: An enhanced topology discovery service for software-defined networking," *Ieee Communications Letters*, Vol. 22, No. 8, August 2018

[23] L. Ochoa-Aday, C. Cervelló-Pastor, and A. Fernández-Fernández, "eTDP: Enhanced topology discovery protocol for software-defined networks," *Digital Object Identifier* 10.1109/ACCESS.2019.2899653

[24] J. Flathagen, O. I. Bentstuen, "Proxy-based optimization of topology discovery in software defined networks," 2019 *International Conference on Military Communications and Information Systems (ICMCIS)*. doi:10.1109/icmcis.2019.8842763

[25] Y. Jia, L. Xu, Y. Yang, and X. Zhang, "Lightweight automatic discovery protocol for OpenFlow-based software defined networking," *Ieee Communications Letters*, Vol. 24, No. 2, February 2020

[26] Y. Gu, D. Li, and J. Yu, "Im-OFDP: An improved OpenFlow-based topology discovery protocol for software defined network," 2020 *Ifip Networking Conference (Networking)*

[27] J. Alvarez-Horcajo, E. Rojas, I. Martinez-Yelmo, M. Savi, and D. Lopez-Pajares, "HDDP: Hybrid domain discovery protocol for heterogeneous devices in SDN," *IEEE Communications Letters*, Vol. 24, No. 8, August 2020

[28] C. El Khalfi, A. El Qadi, and H. Bennis, (2017). "A comparative study of software defined networks controllers," *Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems - ICCWCS*'17. doi:10.1145/3167486.3167509

[29] Bhuvaneswaran, et al., "Benchmarking Methodology for Software-Defined Networking (SDN) Controller Performance," Oct 2018, [Online] Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc8456.txt.pdf

[30] J. A. Bondy and U. S. R. Murty, *Graph theory with applications*. Canadas: Department of Combina tories and Optimization University of Waterloo, 1976. [Online] Available: http://www.iro.umontreal.ca/~hahn/IFT3545/GTWA.pdf

[31] A. Taha, (2014) "Software-defined networking and its security," *School of Electrical Engineering, Aalto University*

[32] F. Hu, Q. Hao, and K. Bao, (2014). "A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation," *IEEE Communications Surveys & Tutorials*, 16(4), 2181–2206. doi:10.1109/comst.2014.2326417

[33] R. Sharpe, "Wireshark User's Guide 20350 for Wireshark 0.99.5", *NS Computer Software and Services P/L Ed*, Ulf Lamping [Online] Available: http://penta3.ufrgs.br/videoconferencia/cdEspecRedesVidconfVOIP/software/wireshark-user-guide-us.pdf

[34] S. Xiang, H. Zhu, X. Wu, L. Xiao, M. Bonsangue, W. Xie, and L. Zhang, (2019). "Modeling and verifying the topology discovery mechanism of OpenFlow controllers in software-defined networks using process algebra," *Science of Computer Programming*, 102343.

[35] A. Pranata, T. S. Jun, and D. S. Kim, (2018). "Overhead reduction scheme for SDN-based data center networks," *Computer Standards & Interfaces*.

[36] A. Azzouni, N. T. Mai Trang, R. Boutaba, and G. Pujolle, (2017). "Limitations of OpenFlow topology discovery protocol, " *16th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*. doi:10.1109/medhocnet.2017.8001642

[37] IXIA and NEC. "White paper: SDN Controller Testing, Part1," [Online] Available: https://www.necam.com/docs/?id=2709888a-ecfd-4157-8849-1d18144a6dda

[38] E. Stergiou, E. Glavas and S. Margariti, "Performance Evaluation of Delta Networks Operating via Cut-Through Switching under Hotspot Traffic", Proceedings of the 2020 *IEEE International Conference on Information Technologies* (*InfoTech-2020*), pp. 120-125, 17-18 September 2020, St. St. Constantine and Elena, Bulgaria

[39] E. Stergiou, J. Garofalakis, D. Liarokapis and S. Margariti, "A Study of Multilayer Omega Networks Operating with Cut-Through Mechanism under Uniform Traffic", *Proceedings of the 2020 IEEE 5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference*, Corfu, September 25-27, 2020

[40] E. Stergiou, "A study of multistage interconnection networks operating with wormhole routing and equipped with multi-lane storage", *Taylor*

Francis : International Journal Of Parallel, Emergent and Distributed Systems, pp. 1-23, Published 18 Jul 2020 (on line).

[41] J. Garofalakis, and E. Stergiou, "Analytical Model for Performance Evaluation of Multilayer Multistage Interconnection Networks servicing Unicast and Multicast Traffic by Partial multicast operation", *Elsevier Performance Evaluation*, Volume 67, Issue 10, October 2010, pp.959-976, 2010.

[42] E. Stergiou, D. Liarokapis, E. Glavas, C. Angelis, and S. Margariti, "Reliability study of U-type multistage interconnection networks", *Proceeding of 3rd International Conference on Reliability Engineering (ICRE 2018),* Barcelona, November 24 - 26, 2018, IOP Publishing, IOP Conf. Series : Materials Science and Engineering 575 (2019) 012001.

[43] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "SNDlib 1.0_Survivable network design library," Netw., Int. J., vol. 55, no. 3, pp. 276_286, May 2010.

[44] N. Lefevr. " Complex networks: Study and analysis of biological networks-Application in the spread of diseases," Bc. D. dissertation, Dept. Inform. Engineers , Technological Institution Of Epirus, Arta, Greece, 2015, [Online] Available: https://apothetirio.lib.uoi.gr/xmlui/bitstream/handle/123456789/4903/1406.pdf?sequence=1

[45] (Aug. 2017). GÉANT Topology Map. [Online]. Available: https://www.geant.org/Resources/PublishingImages/GEANT_topology_map_august2017.pdf

[46] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks, " *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999.