ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Συγκριτική μελέτη ελεγκτών Software Defined Networking (SDN)»

ΟΝΟΜΑΤΕΠΩΝΥΜΟ:ΚΟΥΛΟΥΡΑΣ ΙΩΑΝΝΗΣ

ΑΜ: PINT00088

ΕΠΙΒΛΕΠΟΥΣΑ ΚΑΘΗΓΗΤΡΙΑ: ΜΑΡΓΑΡΙΤΗ ΣΠΥΡΙΔΟΥΛΑ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΑΡΤΑ 2022

Ευχαριστίες

Με την ολοκλήρωση της μεταπτυχιακής διπλωματικής μου εργασίας, θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες στην επιβλέπουσα καθηγήτρια μου, κυρία Σπυριδούλα Μαργαρίτη, για την μεγάλη της στήριξη, τη συμπαράσταση της και το μεγάλο ενδιαφέρον της που έδειξε από την αρχή μέχρι το τέλος, καθώς χωρίς αυτά δεν θα μπορούσα να ολοκληρώσω την διπλωματική μου εργασία.

Επίσης ένα μεγάλο ευχαριστώ στην οικογένεια μου, για την μεγάλη τους στήριξη και κατανόηση όλα αυτά τα χρόνια, καθώς και σε όλους όσους με βοήθησαν κατά την διάρκεια των σπουδών μου.

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα διπλωματική εργασία είναι εξολοκλήρου 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.

ΚΟΥΛΟΥΡΑΣ ΙΩΑΝΝΗΣ

ΠΕΡΙΛΗΨΗ

Το Software Defined Networking (SDN) είναι ένα νέο παράδειγμα δικτύωσης που η αρχιτεκτονική του μετακινείται από το παραδοσιακό πλήρως κατανεμημένο μοντέλο σε μια κεντρικοποιημένη προσέγγιση. Αυτή η προσέγγιση χαρακτηρίζεται κυρίως από τον διαχωρισμό των επιπέδων δεδομένων και ελέγχου. Το επίπεδο δεδομένων περιλαμβάνει το στοιχείο προώθησης και το επίπεδο ελέγχου περιλαμβάνει τον ελεγκτή. Στο επίπεδο ελέγχου βασικό στοιχείο είναι ο ελεγκτής, και είναι το στοιχείο που χρησιμοποιείται για τη διαχείριση όλων των λειτουργιών του επιπέδου δεδομένων. Ο ελεγκτής παρέχει υψηλό επίπεδο αφαίρεσης της διαχείρισης στοιχείων προώθησης που απουσιάζει στα σημερινά δίκτυα. Ως εκ τούτου, ο ελεγκτής είναι ένα θεμελιώδες στοιχείο της αρχιτεκτονικής SDN που θα συμβάλει στην επιτυχία ή την αποτυχία του δικτύου SDN.

Οι ελεγκτές, είναι εργαλεία, συνήθως ανοικτού κώδικα, που χρησιμοποιούνται για την ανάπτυξη και προτυποποίηση δικτυακών εφαρμογών καθώς και τη διεξαγωγή πειραμάτων. Με την υποστήριξη κατάλληλων πρωτοκόλλων (π.χ. OpenFlow), επιτρέπουν την εξομοίωση δικτυακών συσκευών (hub, switches, load balancer, firewall) και να εξομοιώσει την λειτουργία τους. Ο χρήστης μπορεί να ορίσει τις παραμέτρους λειτουργίας του δικτυακού περιβάλλοντος, τις τοπολογίες και τους περιορισμούς και να εκτελέσει μια σειρά πειραμάτων για να διαπιστώσει τη συμπεριφορά μιας συγκεκριμένης δικτυακής υλοποίησης. Σήμερα υπάρχουν αρκετοί διαθέσιμοι SDN ελεγκτές που μπορούν να χρησιμοποιηθούν για συγκριτική αξιολόγηση (benchmarking).

Στόχος αυτής της εργασίας είναι η παρουσίαση και συγκριτική μελέτη των ευρέως χρησιμοποιούμενων ελεγκτών SDN. Η αξιολόγηση θα βασιστεί σε κριτήρια που θα οριστούν, όπως για παράδειγμα η απόδοση, η κλιμάκωση και η ποιότητα υπηρεσιών.

1 Περιεχόμενα

1.ΕΙΣΑΓΩΓΗ

Το διαδίκτυο είναι απαραίτητο πλέον στην καθημερινή μας ζωή λόγω της διευκόλυνσης που παρέχει στην ανθρωπότητα. Σε αυτό το δίκτυο των δικτύων, σχεδόν τα πάντα είναι συνδεδεμένα με τα πάντα και είναι προσβάσιμα από οπουδήποτε. Ως εκ τούτο, η διαχείριση του δικτύου είναι μια πραγματική πρόκληση, κυρίως στα μεγάλα δίκτυα. Η ανάπτυξη του δικτύου εξυπηρετεί ανάγκες όσο αφορά την ποιότητα, καλύτερες υπηρεσίες, την διαθεσιμότητα, την αξιοπιστία και την ασφάλεια. Όλες αυτές οι απαιτήσεις αύξησαν την πολυπλοκότητα των δικτύων με αποτέλεσμα το σημερινό δίκτυο να πρέπει να υποστηρίζει ετερογενείς εφαρμογές, τεχνολογίες και εξοπλισμό πολλών προμηθευτών.

Το Software-Defined Networking (SDN) χρησιμοποιείται για την ενίσχυση της προγραμματισιμότητας του δικτύου και την απλούστευση της διαχείρισης του δικτύου. Το SDN είναι ένα νέο παράδειγμα δικτύου που επιτρέπει το διαχωρισμό των λειτουργιών του επιπέδου ελέγχου και του επιπέδου δεδομένων των παραδοσιακών δικτύων, γεγονός που οδηγεί σε μια πιο δυναμική, ευέλικτη, αυτοματοποιημένη και διαχείριση αρχιτεκτονική. Τα δίκτυα SDN έχουν τρία επίπεδα όπως θα δούμε και παρακάτω: το επίπεδο υποδομής (Data Plane), το επίπεδο ελέγχου (Control Plane) και το επίπεδο εφαρμογής (Management Plane). Το επίπεδο ελέγχου είναι ένας ελεγκτής που λειτουργεί ως κεντρική οντότητα διαχείρισης. Ο ελεγκτής λαμβάνει αποφάσεις σχετικά με τη διαχείριση του δικτύου, την ποιότητα υπηρεσίας (QoS) και την ασφάλεια.

Κατά καιρούς έχουν εφαρμοστεί διάφορες προσεγγίσεις, γλώσσες, αρχιτεκτονικές, interface προγράμματος εφαρμογών (API) και πρωτόκολλα. Σε αυτή την εργασία, θα προσπαθήσουμε να συγκρίνουμε τους πιο δημοφιλείς ελεγκτές και να δοκιμάσουμε την απόδοσή και την ποιότητα υπηρεσιών χρησιμοποιώντας το Mininet Kaur et al. (Kaur, 2014) το οποίο είναι ένας εξομοιωτής για την ανάπτυξη μεγάλων δικτύων με τους περιορισμένους πόρους ενός απλού υπολογιστή ή εικονικής μηχανής. Το Mininet δημιουργήθηκε για την έρευνα στον τομέα της δικτύωσης που καθορίζεται από λογισμικό (SDN) και OpenFlow.

1.1 Αντικείμενο και συνεισφορά

Υπάρχουν δεκάδες ελεγκτές ανοικτού κώδικα που χρησιμοποιηθούν γενικά, για εξειδικευμένες εργασίες ή/και συγκεκριμένες εφαρμογές. Αντικείμενο αυτής της εργασίας είναι η συγκριτική μελέτη των ελεγκτών SDN: ONOS, Ryu, Floodlight και OpenDayLight. Η επιλογή του ελεγκτή είναι ένα σημαντικό ζήτημα καθώς ο ελεγκτής έχει την αρμοδιότητα και επιτρέπει την κεντρική διαχείριση και τον έλεγχο, την αυτοματοποίηση και την επιβολή πολιτικών σε φυσικά και εικονικά περιβάλλοντα δικτύου.

Η συνεισφορά μας στο ζήτημα αυτό συνοψίζεται στα ακόλουθα:

- Παρουσιάζουμε και συγκρίνουμε τις πρόσφατες εκδόσεις των παραπάνω ελεγκτών SDN.
- Διερευνούμε την συμπεριφορά τους για το μοντέλο client-server και τα πρωτόκολλα TCP και UDP. Για τη μελέτη αυτή θεωρούμε τις τοπολογίες:
  - Απλή τοπολογία
  - Τοπολογία FatTree
- Συγκρίνουμε πειραματικά, τους ελεγκτές SDN σε περιβάλλον Mininet χρησιμοποιώντας διάφορες μετρικές, όπως η απόδοση (throughput), ο μέσος όρος καθυστέρησης (average), η τυπική απόκλιση (standar deviation) και διακύμανση (jitter) για τις παραπάνω τοπολογίες

1.2 Δομή εργασίας

Η εργασία αποτελείται από 9 επιμέρους κεφάλαια. Στο πρώτο Κεφάλαιο γίνεται αναφορά στην δικτύωση που είναι καθορισμένη από λογισμικό SDN καθώς και στον τρόπο τον οποίο εργαστήκαμε. Στο 2ο Κεφάλαιο παρουσιάζονται οι σχετικές εργασίες από τη διεθνή βιβλιογραφία. Στο 3ο Κεφάλαιο αναφερόμαστε πιο συγκεκριμένα για τους ελεγκτές και την δικτύωση SDN παρέχοντας τους ορισμούς και άλλες βασικές έννοιες. Στο 4ο Κεφάλαιο εστιάζουμε στους ελεγκτές SDN ανοικτού κώδικα που θα χρησιμοποιήσουμε στη μελέτη μας. Συνεχίζοντας στο 5ο Κεφάλαιο παραθέτουμε τα διάφορα εργαλεία αξιολόγησης που χρησιμοποιούνται από την επιστημονική κοινότητα, όσον αφορά τους ελεγκτές SDN. Στα Κεφάλαια 6 και 7 παρουσιάζονται η πειραματική μελέτη και (τοπολογίες, η μεθοδολογία όπου εργαστήκαμε καθώς και τα πειράματα μας), και τα αποτελέσματα. Στο Κεφάλαιο 8 γίνεται η αξιολόγηση των πειραματικών αποτελεσμάτων και το Κεφάλαιο 9 παρουσιάζει τα τελικά συμπεράσματα της μελέτη μας.

2 Σχετικές εργασίες

Μέχρι σήμερα, έχουν δημοσιευθεί αρκετές εργασίες για τη συγκριτική αξιολόγηση διαφόρων ελεγκτών. Ο Mamushiane et al. (Mamushiane) αξιολόγησε και σύγκρινε την απόδοση των ελεγκτών ανοικτού κώδικα με βάση την απόδοση και την καθυστέρηση, χρησιμοποιώντας ένα εργαλείο συγκριτικής αξιολόγησης ανοικτού κώδικα. Οι ελεγκτές που επιλέχθηκαν για αυτόν τον έλεγχο επιδόσεων είναι οι εξής: Ryu, Floodlight, ONOS και OpenDayLight. Αυτοί οι ελεγκτές επιλέχθηκαν με βάση τη δημοτικότητά τους. Οι μετρικές απόδοσης που εξετάστηκαν στην συγκεκριμένη εργασία ήταν η απόδοση, ο ρυθμός μετάδοσης, η καθυστέρηση και η επεκτασιμότητα. Ως εργαλείο αξιολόγησης χρησιμοποιήθηκε το CBench καθώς διαθέτει δύο τρόπους λειτουργίας: λειτουργία καθυστέρησης και λειτουργία απόδοσης. Όσον αφορά τα αποτελέσματα της απόδοσης ο Ryu είχε την χαμηλότερη, ενώ ο ONOS την υψηλότερη, σε αντίθεση με τις μετρήσεις που έγιναν στην λειτουργία καθυστέρησης όπου είχε την χειρότερη επίδοση.

Ο Zhu et al. (Zhu, 2019), παρουσίασε τη γενική αρχιτεκτονική και την εξέλιξη των σύγχρονων ελεγκτών SDN . Έκανε μια ποιοτική συγκριτική ανάλυση 34 διαφορετικών ελεγκτών για τις ιδιότητες και τις δυνατότητές τους. Επίσης συζήτησε τις διαφορετικές περιπτώσεις χρήσης αυτών των ελεγκτών, και τις λύσεις που προτάθηκαν για τη βελτίωση της απόδοσής τους από άλλους ερευνητές. Παρουσίασε μια ολοκληρωμένη μελέτη των τεχνικών συγκριτικής αξιολόγησης και εργαλείων για ελεγκτές SDN. Η συγκεκριμένη έρευνα περιλαμβάνει συζητήσεις και προσεγγίσεις που χρησιμοποιούνται για την αξιολόγηση, τις δυνατότητες των εργαλείων συγκριτικής αξιολόγησης, και κυρίως τις λεπτομέρειες των μετρικών που πρέπει να χρησιμοποιούνται για την ποσοτική αξιολόγηση. Πραγματοποίησε ποσοτική ανάλυση 9 διαφορετικών ελεγκτών χρησιμοποιώντας 3 διαφορετικά εργαλεία συγκριτικής αξιολόγησης. Αυτά τα εργαλεία είναι το CBench, PktBlaster και OFNet. Όπως τονίζεται από τους συγγραφείς, κανένα από τα συγκεκριμένα εργαλεία δεν μπορεί να μετρήσει όλα τα στατιστικά στοιχεία επιδόσεων. Με βάση την ποιοτική ανάλυση των ελεγκτών αλλά και των ιδιοτήτων και δυνατοτήτων των εργαλείων συγκριτικής αξιολόγησης, και την αξιολόγηση των ελεγκτών με τη χρήση τους, κατέληξαν πως όσον αφορά την καθυστέρηση και την απόδοση οι ελεγκτές Floodlight, OpenMUL, Beacon, Maestro, OpenDaylight και ONOS έχουν σημαντικά καλύτερες επιδόσεις από τους NOX, POX και Ryu. Ωστόσο, απαιτούν επίσης περισσότερους φυσικούς πόρους προκειμένου να αποδώσουν πιο αποτελεσματικά. Επίσης οφείλουμε να αναφέρουμε πως η πλειονότητα των εργασιών που προτείνονται στη βιβλιογραφία δεν παρέχουν επαρκείς λεπτομέρειες για τον πειραματικό μέρος ώστε εύκολα να μπορεί να το αναπαράγει κάποιος τρίτος. Ως εκ τούτου, εκτός από θεωρητική σύγκριση, δεν είναι δυνατή η αξιολόγησή τους. Τέλος οι συγγραφείς έβγαλαν κάποια πορίσματα και για τα εργαλεία αξιολόγησης που χρησιμοποίησαν καθώς παρατηρήσαν ότι ορισμένα από τα διαθέσιμα χαρακτηριστικά του εργαλείων, όπως το μήκος πακέτων, το μέγεθος του buffer του vSwitch κ.λπ. επηρεάζουν την απόδοση του ελεγκτή.

Σύμφωνα με τον Zhu et al. (ZHU, 2020), τα δίκτυα SDN έχουν μελετηθεί από διαφορετικές πλευρές. Οι σχετικές μελέτες μπορούν να ταξινομηθούν σε δύο κύριους τύπους: οι γενικές έρευνες, οι οποίες καλύπτουν ευρέως τον τομέα και μελετούν τα διάφορα στοιχεία των δικτύων SDN, και οι μελέτες για ελεγκτές SDN, οι οποίες είναι περισσότερο στοχευμένες στη λειτουργία και τα χαρακτηριστικά των ελεγκτών SDN. Στη βιβλιογραφία υπάρχει μεγάλος αριθμός εργασιών για τους ελεγκτές SDN. Ορισμένες από τις μελέτες, εξετάζουν θεωρητικά διαφορετικούς ελεγκτές, τις ιδιότητές τους και τις δυνατότητές τους από διαφορετικές πλευρές. Ωστόσο, στην πλειονότητα αυτών των μελετών περιγράφεται μόνο η αρχιτεκτονική και τα χαρακτηριστικά ορισμένων ελεγκτών, καθώς υπάρχουν αρκετοί τέτοιοι ελεγκτές (π.χ., Floodlight, OpenMUL, Beacon, Maestro, ODL,ONOS, POX και Ryu).Η μελέτη του Zhu et al.,καταλήγει στα ακόλουθα συμπεράσματα: α) υπάρχουν ελεγκτές στους οποίους προτείνονται εξειδικεύσεις για χρήση σε συγκεκριμένες κατηγορίες δικτύων (π.χ. δίκτυα blockchain, δίκτυα οχημάτων, διαδίκτυο των πραγμάτων, και ασύρματα δίκτυα αισθητήρων) και ελεγκτές που επιχειρείται η βελτίωση συγκεκριμένων λειτουργιών τους, β) για την αξιολόγηση των επιδόσεων των ελεγκτών χρησιμοποιούνται διάφορες πλατφόρμες (π.χ. Mininet) και εργαλεία όπως το CBench, PktBlaster, and OFNet, γ) οι αξιολογήσεις μπορεί να είναι ποιοτικές ή ποσοτικές ή πειραματικές και βασίζονται σε κριτήρια ή μετρικές (π.χ. απόδοση, καθυστέρηση, απώλεια πακέτων, κ.ά.) δ) δεν υπάρχει ελεγκτής που να υπερτερεί έναντι των άλλων, αλλά υπάρχουν περιπτώσεις που κάποιοι ελεγκτές αναδεικνύονται καλύτεροι, Για παράδειγμα τα ODL, Floodlight, και ONOS έχουν καλύτερη διεκπαιρεωτική ικανότητα, αλλά απαιτούν περισσότερο χρόνο για την εγκατάσταση των ροών -υψηλότερη καθυστέρηση. Λαμβάνοντας υπόψη την απόδοση, οι ελεγκτές όπως ο Floodlight, OpenMUL, Beacon, Maestro αλλά και οι ODL και ONOS, αποδίδουν σημαντικά καλύτερα από τους κεντρικοποιημένους ελεγκτές όπως POX και Ryu. Ωστόσο, απαιτούν επίσης περισσότερους φυσικούς πόρους για αποτελεσματική απόδοση.

Η εργασία των Jawaharan et al. (2018) βασίζεται στην εμπειρική αξιολόγηση τριών ελεγκτών: ONOS, OpenMUL και POX. Οι ελεγκτές έχουν υλοποιηθεί με διαφορετική γλώσσα προγραμματισμού ο καθένας και συγκεκριμένα σε Java το ONOS, σε C το OpenMUL και σε Python το POX. Για τη σύγκρισή τους χρησιμοποιήθηκαν τα εργαλεία CBench σε περιβάλλον Mininet και Wireshark για την ανάλυση των πακέτων. Πρώτα συγκρίθηκαν χρησιμοποιώντας το CBench όσον αφορά θέματα καθυστέρησης του ελεγκτή και την απόδοση. Ωστόσο, το CBench παράγει μόνο ’’ψεύτικα μηνύματα εισόδου πακέτων’’ δημιουργώντας περιπτώσεις switches. Επίσης απαιτεί μια πρόσθετη cbenchAPI() σε κάθε ελεγκτή για να ανταποκριθεί στα ψεύτικα μηνύματα εισόδου πακέτων και να απαντήσει με ένα packet-out παρακάμπτοντας τα προεπιλεγμένα βήματα επεξεργασίας πακέτων.

Ο khondoker et al. (Khondoker, 2014) συζήτησε για τη σύγκριση και επιλογή SDN με βάση τα χαρακτηριστικά των ελεγκτών. Η εργασία συγκρίνει τους POX, Floodlight, Ryu, Trema και OpenDaylight με βάση στην αναλυτική διαδικασία ιεραρχίας (AHP), αλλά δεν παρουσίασε καμία απόδειξη πειραματισμού.

Σύγκριση πολλών δημοφιλών ελεγκτών έκανε ο Salman et al. (Salman, 2016) προσπαθώντας να δοκιμάσει την απόδοση τους με τη χρήση του CBench, το οποίο είναι ένα εργαλείο δοκιμών. Αφού αναφέρθηκε θεωρητικά σε κάποιους ελεγκτές, τους σύγκρινε με βάση την αποτελεσματικότητα και την καθυστέρηση τους, όπου δεν είναι τα μοναδικά κριτήρια για την επιλογή ενός ελεγκτή όπως ανέφερε αλλά είναι από τις βασικές απαιτήσεις ενός ελεγκτή. Η εκτέλεση των δοκιμών έγινε με δύο τρόπους, μεταβάλλοντας τον αριθμό των switches, και τη λειτουργία απόδοσης μεταβάλλοντας τον αριθμό των threads που συνδέονται με την περίπτωση του ελεγκτή. Τα αποτελέσματα που λάβανε δείχνουν ότι οι ελεγκτές που κωδικοποιήθηκαν με τη γλώσσα C έδωσαν την υψηλότερη απόδοση όπως ο (Mul, Libfluid_msg) και έπονται οι ελεγκτές που κωδικοποιήθηκαν με java, όπως οι Beacon, Iris και Maestro. Ωστόσο, στη λειτουργία καθυστέρησης, ο Maestro, έδωσε τα καλύτερα αποτελέσματα καθυστέρησης. Επιπλέον, παρατήρησαν ότι οι ελεγκτές που είναι κωδικοποιημένοι σε C και Java προσφέρουν υψηλότερες επιδόσεις σχετικά με τον αριθμό των threads. Τέλος, επισήμαναν πως ο POX που είναι κωδικοποιημένος σε Python δεν παρουσίασε σημαντική διαφορά επειδή η Python δεν υποστηρίζει πολύ αποδοτικά τη χρήση πολλαπλών threads,.

Για τους ελεγκτές Pox, Ryu, ONOS, ODL μίλησε στην έρευνα του ο Stancu et al. (Stancu1, 2015) όπου μετά από μια σύντομη επισκόπηση των συγκεκριμένων ελεγκτών που χρησιμοποιήθηκαν σε προσομοιώσεις, περιέγραψε την μεθοδολογία που χρησιμοποιήθηκε για την δοκιμή των ελεγκτών, συμπεριλαμβανομένης και της τοπολογίας και παρουσίασε τα αποτελέσματά του σε προσομοιώσεις. Πιο συγκεκριμένα οι μετρήσεις επιδόσεων των παραπάνω ελεγκτών έγιναν σε περιβάλλον του Mininet, όπου δημιουργήθηκε μια τοπολογία δέντρου. Τα αποτελέσματα έδειξαν πως το ONOS έχει τον μικρότερο χρόνο σύγκλισης (converge time), όπως ήταν αναμενόμενο, επειδή είναι ένας ελεγκτής έτοιμος για παραγωγή, σε αντίθεση με το POX, ο οποίος χρησιμοποιείται για ερευνητικούς σκοπούς. Και πάλι, οι διαφορές είναι μικρές επειδή η τοπολογία είναι απλή. Επίσης δοκιμές έγιναν και με την χρήση του εργαλείου *iperf*, όπου σε αυτήν την περίπτωση το ONOS φαίνεται να έχει την καλύτερη απόδοση, με εύρος ζώνης που ξεπερνά τα 9 Gbps.

Σύγκριση ελεγκτών έκανε και ο Bispo et al. (2017) όπου στόχευσε όχι να συγκρίνει ελεγκτές διαφορετικής φύσης, αλλά και ελεγκτές με παρόμοια χαρακτηριστικά για να εκτιμηθούν οι διαφορές στις επιδόσεις τους. Στο πλαίσιο αυτό, από έναν αυξανόμενο κατάλογο δημοφιλών ελεγκτών SDN, επιλέχθηκαν δύο ελεγκτές βασισμένοι στην Python (δηλαδή ο POX και Ryu), καθώς επίσης και δύο ελεγκτές που βασίζονται σε Java (δηλαδή, ο OpenDayLight και ONOS). Εκτελώντας το CBench και στις δύο λειτουργίες καθυστέρησης και απόδοσης, πραγματοποιήθηκε μια αξιολόγηση των τεσσάρων ελεγκτών SDN ανοικτού κώδικα. Οι εργασίες που πραγματοποιήθηκαν, τόσο ποιοτικά όσο και ποσοτικά οδήγησε σε σημαντικά αποτελέσματα και συμπεράσματα. Η ποιοτική σύγκριση έδειξε ότι τόσο το ONOS όσο και το ODL είναι οι πιο χαρακτηρισμένοι ελεγκτές. Όσον αφορά τα αποτελέσματα των επιδόσεων, όπου ο ODL ξεχώριζε σαφώς, αναφέρουν ότι ο ODL πρέπει να εκλεγεί ως ο καλύτερος ελεγκτής μεταξύ όλων των υπό δοκιμή ελεγκτών.

Ο Shalimov et al. (Shalimov, 2013) επιχείρησε την αμερόληπτη συγκριτική ανάλυση της αποτελεσματικότητας των πιο δημοφιλών και ευρέως χρησιμοποιούμενων ελεγκτών έως το 2013. Όρισαν ως "αποτελεσματικότητα του ελεγκτή" ένα σύνολο δεικτών, όπως απόδοση, επεκτασιμότητα, αξιοπιστία και ασφάλεια. Για την αξιολόγηση των ελεγκτών χρησιμοποιήσαν το γνωστό εργαλείο CBench. Ωστόσο, δεν έμειναν ικανοποιημένοι από τις δυνατότητες του CBench και έτσι ανέπτυξαν το δικό τους πλαίσιο για τη δοκιμή ελεγκτών SDN/OpenFlow, το οποίο ονόμασαν HCprobe. Οι ελεγκτές που αξιολογήθηκαν ήταν οι: NOX, POX, Beacon, Floodlight, MUL, Maestro και Ryu.

Δύο ελεγκτές, ο Floodlight και OpenDaylight, συγκρίνονται μεταξύ τους από τον Rowshanrad et al. (ROWSHANRAD, 2016) χρησιμοποιώντας το Mininet, έναν εξομοιωτή SDN. Η αξιολόγηση βασίστηκε στις παραμέτρους QoS, όπως την καθυστέρηση και τις απώλειες. Η σύγκριση έγινε για διαφορετικές τοπολογίες και φορτία δικτύου. Τα αποτελέσματα μπορούν να βοηθήσουν στην επιλογή του καλύτερου ελεγκτή για δίκτυα όταν πρέπει να ικανοποιούνται οι απαιτήσεις QoS, όπως δίκτυα που μεταφέρουν κίνηση πολυμέσων (multimedia traffic), ή για άλλες ειδικές περιπτώσεις χρήσης, όπως κέντρα δεδομένων και clouds. Υπάρχει επίσης μια σύντομη εισαγωγή στο Mininet και τις υποστηριζόμενες τοπολογίες του. Μετά από διάφορες μετρήσεις που έγιναν, παρατηρήσαν ότι η καθυστέρηση στο Floodlight είναι μεγαλύτερη από το OpenDayLight. Αυτό σημαίνει ότι το Floodlight χρειάζεται περισσότερο χρόνο για να βρει τη διαδρομή και να στείλει μια απόφαση για τις νεοεισερχόμενες ροές, όπως επίσης πως και τα δίκτυα που χρησιμοποιούν το Floodlight ως ελεγκτή τους, έχουν μεγαλύτερη καθυστέρηση ενώ η κυκλοφορία είναι χαμηλή.

Ο Raju et al. (RAJU, 2018) πραγματοποιήσαμε μια σύγκριση των διαφόρων ελεγκτών με βάση πλήθος κριτηρίων, όπως η απόδοση, διάφορα θέματα καθυστέρησης και αρχιτεκτονικής. Έτσι, λόγω της ποικιλομορφίας των εφαρμογών SDN και των διαφορετικών ελεγκτών, η επιλογή του καταλληλότερου ελεγκτή θα εξαρτάται κατά κάποιο τρόπο από την εφαρμογή. Οι συγγραφείς διαπιστώσαν ότι το OpenDaylight είναι μια καλή επιλογή ως ένας ελεγκτής με πλήρη χαρακτηριστικά.

Στην έρευνα που έγινε από τον Fancy et al. (C.Fancy, 2017), τα χαρακτηριστικά των POX και Floodlight αναλύονται με τη δημιουργία ενός δικτύου SDN. Η εγκατάσταση έγινε σε VirtualBox. Ο κόμβος SDN περιέχει διάφορους ελεγκτές ως πακέτο. Σύμφωνα με τις ανάγκες του χρήστη, ο εν λόγω ελεγκτής μπορεί να κατασκευαστεί και να χρησιμοποιηθεί για τον έλεγχο του δικτύου που αναπτύχθηκε. Για την ανάλυση της ικανότητας χειρισμού του δικτύου των ελεγκτών, οι παράμετροι καθυστέρηση και ρυθμός μετάδοσης λαμβάνονται υπόψη. Επίσης, μεταβλήθηκε το μέγεθος της τοπολογίας προσθέτοντας 10, 20, 30, 40 και 50 κεντρικούς υπολογιστές χωριστά και σημειώνονται οι μετρήσεις. Αρχικά το δίκτυο ελέγχεται από τον ελεγκτή POX. Η κυκλοφορία αναλύεται με το εργαλεία Wireshark, και γραμμής εντολών ping και iperf. Στη συνέχεια δημιουργούνται παρόμοια δίκτυα για τη μετάδοση πακέτων υπό τον έλεγχο του Floodlight Controller. Στο ανάλυση που έγινε πάνω σε βασικές

τοπολογίες, σε θέματα καθυστέρησης ο POX εμφάνισε μεγαλύτερες καθυστερήσεις ενώ σε θέματα απόδοσης στις περισσότερες τοπολογίες πάλι ο POX εμφανίστηκε κατώτερος από τον Floodlight.

Σε άλλη επιστημονική έρευνα, πραγματοποιήθηκε μια σύγκριση στην απόδοση δύο τυπικών ελεγκτών του Floodlight και Ryu, Li et al. (Li, 2020). Αυτοί οι δύο ελεγκτές χρησιμοποιούνται ευρέως λόγω της απλής αρχιτεκτονικής τους, της ευέλικτης ανάπτυξης και εύκολης επέκτασης. Υπάρχουν δύο κύριοι δείκτες απόδοσης του δικτύου: το εύρος ζώνης και η καθυστέρηση. Η καθυστέρηση καθορίζει τη μέγιστη ερώτηση ανά δευτερόλεπτο (QPS), ενώ το εύρος ζώνης καθορίζει το μέγιστο φορτίο που μπορεί να υποστηριχθεί. Οι συγγραφείς συγκρίνανε κυρίως τις δύο πιο σημαντικές παραμέτρους και στη συνέχεια τις επαληθεύσανε μέσω πειραμάτων προσομοίωσης. Εγκαταστήσαν και αναπτύξαν αυτούς τους δύο ελεγκτές σε περιβάλλον Linux, και αξιολόγησαν την απόδοση των δύο ελεγκτών σε διαφορετικές τοπολογίες. Χρησιμοποιώντας τα εργαλεία Mininet και Iperf, πήραν χαρακτηριστικές μετρήσεις απόδοσης, όπως καθυστέρηση και εύρος ζώνης, για διαφορετικές τοπολογίες. Ο ελεγκτής που υπερτερούσε σε όλες τις περιπτώσεις ήταν ο Floodlight..

Οι Paliwal et al. (MANISH PALIWAL, 2018), παρουσίασαν μια ανασκόπηση σχετικά με τους ελεγκτές στο SDN. Οι συγγραφείς συζήτησαν την κεντρική ικανότητα λήψης αποφάσεων των ελεγκτών και τους διέκριναν σε δύο κατηγορίες ανάλογα με τον τρόπο λειτουργίας τους: οι συγκεντρωτικοί ελεγκτές (Beacon, Maestro, Rosemary, NOX, ODL) και οι κατανεμημένοι ελεγκτές (ONOS, Fleet, Onix, PANE, HyperFlow). Οι συγγραφείς αξιολόγησαν τις επιδόσεις των ελεγκτών όσον αφορά τον χρόνο απόκρισης και την απόδοση και διαπίστωσαν ότι οι συγκεντρωτικοί ελεγκτές παρέχουν απλοποιημένη αρχιτεκτονική και αποτελεσματικό χειρισμό αιτημάτων, αλλά αποτυγχάνουν στην επεκτασιμότητα. Οι κατανεμημένοι ελεγκτές πλεονεκτούν ως προς την επεκτασιμότητα, αλλά απαιτούν κατάλληλη διαδικασία κατά την ανταλλαγή μηνυμάτων στη συστάδα.

Στην έρευνα των Arahunashi et al. (Arahunashi, 2019), αξιολογούνται οι επιδόσεις τεσσάρων διαφορετικών ελεγκτών, των Ryu, OpenDayLight, ONOS και Floodlight με τη χρήση του εξομοιωτή Mininet. Προκειμένου να αναλυθούν οι επιδόσεων, εξετάστηκαν δύο παράμετροι QoS, οι οποίες είναι η καθυστέρηση και η απόδοση. Για τον υπολογισμό των παραμέτρων QoS χρησιμοποιήθηκαν οι εντολές iperf και ping. Αυτές οι επιδόσεις QoS αναλύθηκαν για τρεις διαφορετικές τοπολογίες δικτύου, όπως η γραμμική, δέντρου και πλέγματος. Από τα αποτελέσματα που προέκυψαν, ο ελεγκτής Floodlight δίνει καλύτερη απόδοση από τους άλλους τρεις ελεγκτές όσον αφορά την καθυστέρηση της απόδοσης.

Η εργασία των Jehad Ali et al (Jehad Ali, 2018), ασχολείται με τη σύγκριση των επιδόσεων δύο ελεγκτών SDN βασισμένων στην Python, δηλαδή των POX και Ryu υπό διαφορετικές τοπολογίες δικτύου, όπως single, linear, tree, dumbbell, data center networks DCN και Software Defined naval networks που χρησιμοποιούν δορυφορική επικοινωνία, (SATCOM), δηλαδή SDN-SAT. Πειραματικά αποτελέσματα, που επικυρώθηκαν μέσω του Mininet έδειξαν σαφώς ότι το Ryu έχει ανώτερες επιδόσεις,

δηλαδή αύξηση της απόδοσης TCP και μείωση της καθυστέρησης κατά σε ενιαίο, γραμμικό, δέντρο, dumbbell και DCN τοπολογίες αντίστοιχα. Ομοίως, στην τοπολογία SDN-SAT ο Ryu έχει αύξηση της απόδοσης TCP και μείωση στην καθυστέρηση σε σύγκριση με τον ελεγκτή POX.

Οι Badotra et al. (Sumit Badotra S. N., 2019), πραγματοποίησαν πειραματική σύγκριση μεταξύ των δύο πιο γνωστών ελεγκτών, δηλαδή του OpenDayLight (ODL) και του λειτουργικού συστήματος ανοικτής δικτύωσης (ONOS). Οι πειραματικές μετρήσεις πραγματοποιούνται στο εργαλείο εξομοίωσης Mininet. Τα πακέτα σε πραγματικό χρόνο καταγράφονται και αναλύονται με τη χρήση του εργαλείου Wireshark. Τα αποτελέσματα της σύγκρισης κατέληξαν στο συμπέρασμα ότι η απόδοση του ελεγκτή ODL είναι καλύτερη από το ONOS, την απόδοση, τον χρόνο διαδρομής (Round Trip Time - RTT) και το εύρος ζώνης.

Ανάλυση της απόδοσης μεταξύ των ελεγκτών POX και Floodlight έγινε από τους Bholebawa et al. (Idris Z. Bholebawa, 2018). Για τους δύο ελεγκτές SDN έγινε σύγκριση των επιδόσεων τους με κριτήρια την απόδοση και το χρόνο round-trip μεταξύ των κόμβων και του τελικού χρήστη. Με τη βοήθεια της διεπαφής γραμμής εντολών (CLI) του Mininet, προτείνονται διάφορες τοπολογίες και οι επιδόσεις των προαναφερθέντων ελεγκτών OpenFlow συγκρίνονται και αναλύονται. Από τη συζήτηση των αποτελεσμάτων συνάγεται το συμπέρασμα ότι ο ελεγκτής Floodlight παρέχει καλύτερη απόδοση σε σύγκριση με τον ελεγκτή POX για διάφορες τοπολογίες δικτύου. Χρησιμοποιώντας έναν ελεγκτή Floodlight που βασίζεται σε Java ως επίπεδο ελέγχου εφαρμογής, η συνολική απόδοση του δικτύου βελτιώνεται από την άποψη του round-trip, του χρόνου και της απόδοσης.

Σε μια άλλη έρευνα Priya et al. (A. Vishnu Priya, 2019), οι συγγραφείς συγκρίναν τις επιδόσεις των πιο γνωστών ελεγκτών OpenFlow όπως οι NOX, POX, Ryu, Floodlight με βάση το χειρισμό των πακέτων τους μεταβάλλοντας το μέγεθος των πακέτων, τον αριθμό των πακέτων και το μοτίβο άφιξης σε στις ροές κυκλοφορίας IP. Το εργαλείο Distributed internet traffic flow generator (D-ITG) χρησιμοποιήθηκε για τη μέτρηση των επιδόσεων όσον αφορά την καθυστέρηση, την απόδοση και την απώλεια πακέτων. Τα αποτελέσματα του πειράματός μας δείχνουν ότι το Floodlight έχει την καλύτερη απόδοση και την μικρότερη καθυστέρηση σε σύγκριση με άλλους ελεγκτές. Αυτή η εργασία βοηθάει τον εκάστοτε ερευνητή στην επιλογή του κατάλληλου ελεγκτή για τις απαιτήσεις του.

Ο Yamei et al. (Fan Yamei, 2016) επέλεξε δύο δημοφιλείς, ευρύτερα χρησιμοποιούμενους ελεγκτές ανοικτού κώδικα, τους OpenDaylight και ONOS για σύγκριση. Ανέλυσε την αρχιτεκτονική υλοποίησης και πλαίσιο μοντέλου των δύο ελεγκτών. Στη συνέχεια, κατασκεύασε την πλατφόρμα ελεγκτή, προσομοιώσει την υποκείμενη τοπολογία χρησιμοποιώντας το CBench και το Mininet για να λάβουν τις παραμέτρους απόδοσης και να αναλύσουν τα δεδομένα. Τα αποτελέσματα έδειξαν ότι το ONOS έχει καλές επιδόσεις σε GUI, clusters, link-up, εναλλαγή και ρυθμό μετάδοσης. Το OpenDaylight έχει καλύτερες επιδόσεις όσον αφορά την ανακάλυψη τοπολογίας και τη σταθερότητα.

Συμπεραίνουμε ότι όλη η ευφυΐα του SDN προέρχεται από τον ελεγκτή που λειτουργεί σαν εγκέφαλος του συστήματος. Η ικανότητα του ελεγκτή μπορεί να καθοριστεί από τον αριθμό των προερχόμενων από τα switches αιτήσεων που μπορεί να χειριστεί. Διάφορες μονάδες μέσα στον ελεγκτή φροντίζουν για την ανακάλυψη δικτύου, την ανακάλυψη διαδρομής, τη λειτουργία προώθησης της ροής δεδομένων, κ.λπ. Ο κεντρικός ελεγκτής παρέχει απλοποιημένη αρχιτεκτονική και αποτελεσματικό χειρισμό των μηνυμάτων αιτήσεων. Από την άλλη πλευρά, οι κατανεμημένοι ελεγκτές αποδίδουν καλά στο θέμα της επεκτασιμότητας και παρέχουν μέγιστη απόδοση με υψηλή διαθεσιμότητα, αλλά απαιτούν κατάλληλη διαδικασία ανταλλαγής μηνυμάτων. Από τις προαναφερόμενες έρευνες παρατηρούμε ότι οι ελεγκτές που διακρίθηκαν αλλά και χρησιμοποιηθήκαν περισσότερο σε δοκιμές ήταν οι Ryu, OpenDayLight, ONOS, Floodlight και POX. Οι απόψεις διίστανται καθώς οι μελέτες δείχνουν ότι οι ελεγκτές έχουν ανταγωνιστική συμπεριφορά, καθώς υπάρχουν καταστάσεις όπου και οι δύο ελεγκτές έχουν την ίδια απόδοση. Έτσι, λόγω της ποικιλομορφίας των εφαρμογών SDN και των διαθέσιμων ελεγκτών, η επιλογή του καταλληλότερου ελεγκτή εξαρτάται κατά κάποιο τρόπο από την εκάστοτε εφαρμογή. Οι βασικότεροι παράμετροι που συγκρίνονται είναι η καθυστέρηση, απόδοση σε συνάρτηση με την τοπολογία του δικτύου . Οι περισσότερες έρευνες έγιναν με χρήση του Mininet και τα ποιο ευρέως εργαλεία συγκριτικής αξιολόγησης είναι το Cbench και το Hcprobe τα οποία θα τα αναπτύξουμε παρακάτω. Βασικό μειονέκτημα των περισσότερων εργασιών είναι πως ναι μεν παρουσιάζουν τα αποτελέσματα των συγκρίσεων μεταξύ των εκάστοτε ελεγκτών, αλλά σχεδόν κανείς δεν εξηγεί λεπτομερώς πως πάρθηκαν αυτά τα αποτελέσματα βήμα-βήμα.

Πίνακας 1 Πίνακας σχετικών εργασιών

Εργασία	Ελεγκτές									Κριτήρια Αξιολόγησης
	POX	RYU	ONOS	ODL	Floodlight	Trema	Faucet	MAESTRO	BEACON	
Mamushiane,2018		X	X	X	X					Throughput, Scalability, latency, Aggregate Performance
Salman 2016		X	X	X	X			X	X	Performance, Reliability, Scalability, and Security
Zhu,2019	X	X	X	X	X	X		X	X	Throughput, latency, Flow measurements
Fancy,2017	X				X					Throughput, traffic, Delay

Paliwal,2018			X	X				X	X	Throughput, Response time
Bispo, 2017	X	X	X	X						Throughput, Latency
Bholebawa, 2017	X				X					Roundtrip delay, throughput
Arahunashi, 2019		X	X	X	X					Average delay, throughput
Yamei, 2016			X	X						Delay, Flow-mod response speed under throughput mode
Khondoker, 2014	X	X		X	X	X				MultiCriteria Decision Making problem (MCDM) (platform, Openflow, age,documentation,modularity, etc.
Y.Li, 2020		X			X					Bandwidth, latency,
Madhukrishna Priyadarsini, 2017	X				X			X	X	Throughput, Response time, Jitter
Priya, 2019	X	X			X					Delay, Jitter, Packet loss Throughput,
Badotra, 2020			X	X						Bandwidth, jitter, response time
Jehad Ali, 2018	X	X								Throughput, Latency,
Raju, 2018	X	X		X	X	X		X	X	Review of Performance, Reliability, Scalability, and Security
ROWSHANRAD, 2016				X	X					Latency, Jitter, packet loss, Throughput
Stancu, 2015	X	X	X	X						Bandwidth, packet loss
Shalimov, 2013	X	X			X			X	X	Throughput, latency, reliability, Security,

3 Επισκόπηση του SDN

3.1 Εισαγωγή και ορισμοί

Ο όρος Software-Defined Networking (SDN) έχει επινοηθεί τα τελευταία χρόνια. Ωστόσο, η ιδέα πίσω από το SDN εξελίσσεται από το 1996 με γνώμονα την επιθυμία να παρέχεται ελεγχόμενη από τον χρήστη διαχείριση της προώθησης σε κόμβους δικτύου. Για λόγους σαφήνειας, το SDN περιγράφεται σε αυτό το άρθρο με τον ορισμό του Open Networking Foundation (opennetworking, χ.χ.): "*Στην αρχιτεκτονική SDN, τα επίπεδα ελέγχου και δεδομένων αποσυνδέονται, η ευφυΐα και η κατάσταση του δικτύου είναι λογικά συγκεντρωμένες και η υποκείμενη υποδομή δικτύου αφαιρείται από τις εφαρμογές*". Το SDN επικεντρώνεται σε τέσσερα βασικά χαρακτηριστικά:

- Διαχωρισμός του επιπέδου ελέγχου από το επίπεδο δεδομένων
- Κεντρικός ελεγκτής και προβολή του δικτύου
- Ανοιχτές διεπαφές μεταξύ των συσκευών στο επίπεδο ελέγχου και στο επίπεδο δεδομένων
- Δυνατότητα προγραμματισμού του δικτύου από εξωτερικές εφαρμογές

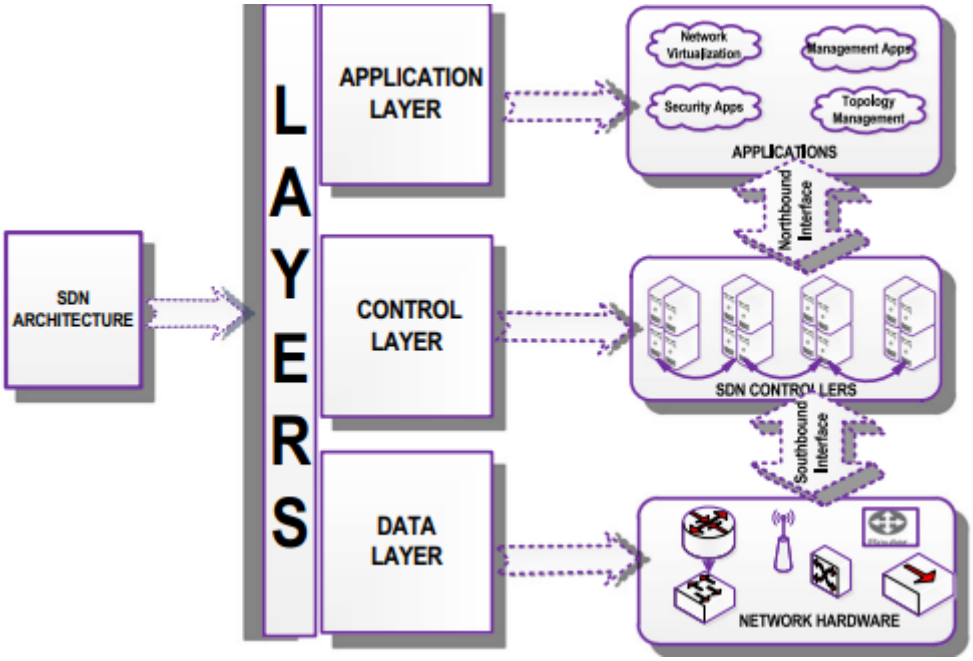
Η δικτύωση που καθορίζεται από το λογισμικό (SDN) διευκολύνει τους οργανισμούς να αναπτύσσουν εφαρμογές και να επιτρέπουν την ευέλικτη παράδοση, προσφέροντας τη δυνατότητα κλιμάκωσης των πόρων δικτύου σύμφωνα με τις ανάγκες των εφαρμογών και των δεδομένων. Το SDN είναι μια καινοτόμος προσέγγιση για το σχεδιασμό, την υλοποίηση και τη διαχείριση δικτύων που διαχωρίζει το επίπεδο ελέγχου του δικτύου και το επίπεδο δεδομένων για καλύτερη εμπειρία χρήστη. Αυτή η τμηματοποίηση του δικτύου προσφέρει πολυάριθμα οφέλη από την άποψη της ευελιξίας και της δυνατότητας ελέγχου του δικτύου. Στο παραδοσιακό σύστημα δικτύωσης οι συσκευές υλικού όπως το Switch, router, τείχος προστασίας και οποιεσδήποτε άλλες συσκευές ρυθμίζονται χειροκίνητα από τους managers/administrators του συστήματος IT και είναι πλήρως υπεύθυνοι για τη διασφάλιση ότι κάθε συσκευή ενημερώθηκε με την τελευταία έκδοση του λογισμικού. Οι παραδοσιακές λύσεις δικτύωσης βρίσκονται σε συνεχή παρακμή λόγω της νέας τεχνολογίας, όπως το cloud computing και virtualization στην αγορά, με τις οποίες μπορούμε να έχουμε διάφορες λειτουργίες να λαμβάνουν χώρα για να δώσουμε υψηλή διαθεσιμότητα και λειτουργίες στους τελικούς χρήστες.

Το SDN είναι μια νέα προοπτική για τον τρόπο που διαχειρίζεται το δίκτυο, καθώς προσφέρει γρήγορη λύση σε προβλήματα περιορισμών σε σχέση με την παραδοσιακή δικτύωση. Δίνει περισσότερη βαρύτητα στο λογισμικό από ότι στο υλικό καθώς διαχωρίζει του επίπεδο ελέγχου (το οποίο καθορίζει πού θα σταλεί η κυκλοφορία) από το επίπεδο δεδομένων. Επίσης είναι ανεξάρτητο, προγραμματιζόμενο και συγκεντρωτικό. Σε έναν παραδοσιακό switch, το επίπεδο δεδομένων και το επίπεδο ελέγχου υλοποιούνται στην ίδια συσκευή. Η κύρια λειτουργία του επιπέδου δεδομένων είναι η προώθηση πακέτων, ενώ το επίπεδο ελέγχου είναι υπεύθυνο για τον έλεγχο της κίνησης ροής, την εφαρμογή των

πολιτικών και των κανόνων για ολόκληρο το δικτύου. Στο SDN, το επίπεδο δεδομένων τοποθετείται στο Switch, ενώ ο Controller είναι λογισμικό που χρησιμοποιεί το πρωτόκολλο "Openflow" για την επικοινωνία με το επίπεδο δεδομένων, και εγκαθίσταται σε έναν ή περισσότερους διακομιστές.

3.2 Η αρχιτεκτονική του SDN

Το δίκτυο SDN χωρίζεται κάθετα σε τρία επίπεδα, δηλαδή ένα επίπεδο εφαρμογής, ένα επίπεδο ελέγχου και ένα επίπεδο δεδομένων. Αυτά τα επίπεδα διαχωρίζονται μεταξύ τους με τη χρήση Northbound και Southbound interfaces (API), όπως απεικονίζεται στην Εικόνα Εικόνα 3-1Εικόνα 3-1. Το Northbound διευκολύνει την επικοινωνία μεταξύ του επιπέδου εφαρμογής και του επιπέδου ελέγχου. Από την άλλη πλευρά, το Southbound διευκολύνει την επικοινωνία μεταξύ του επιπέδου ελέγχου και του επιπέδου δεδομένων μιας αρχιτεκτονικής SDN. Σύμφωνα με το Open Networking Foundation (ONF), τα επίπεδα ελέγχου και δεδομένων σε μια αρχιτεκτονική SDN είναι αποσυνδεδεμένα.



Εικόνα 3-1: Αρχιτεκτονική SDN. (Πηγή: Tao Hun et al. (Tao Han,2018)

- **Επίπεδο εφαρμογής:** Το επίπεδο εφαρμογής, παρέχει ένα σύνολο υπηρεσιών για διάφορες εφαρμογές, όπως παροχή ασφάλειας, QoS, δρομολόγηση και επιθεώρηση πακέτων (DPI). Κάθε εφαρμογή αποτελείται από ένα ή περισσότερα Northbound Interface (NBI). Κάθε εφαρμογή που υποστηρίζεται σε αυτό το επίπεδο δηλώνει προγραμματιστικά τις απαιτήσεις της και το

επιθυμητή συμπεριφορά δικτύου στον ελεγκτή μέσω του Northbound Interface.

- **Northbound Interface:** Το επίπεδο εφαρμογής συνομιλεί με το επίπεδο ελέγχου με χρήση Northbound API. Το πιο γνωστό Interface για επικοινωνία μεταξύ εφαρμογής και ελέγχου είναι το Rest API. Χρησιμοποιώντας αυτά τα API οι διαχειριστές μπορούν να καθοδηγήσουν δυναμικά τον ελεγκτή σχετικά με το πώς να κάνει προσαρμογές στα στοιχεία. Το Southbound όπως θα δούμε και παρακάτω χχρησιμοποιεί ένα πρωτόκολλο ανοικτού κώδικα, το OpenFlow, και αλληλοεπιδρά με το επίπεδο δεδομένων σύμφωνα με τον τρόπο που ορίζει ο ελεγκτής SDN. Το Northbound interface δεν διαθέτει τέτοιου είδους προτύπων πρωτοκόλλων. Ωστόσο, με την πρόοδο της τεχνολογίας τώρα έχουμε ένα ευρύ φάσμα API Northbound όπως ad-hoc API, RESTful APIs κ.λπ. Η επιλογή του Northbound Interface εξαρτάται από τη γλώσσα προγραμματισμού που χρησιμοποιείται στην ανάπτυξη της εφαρμογής. Η εμφάνιση του Northbound API είναι ένα κρίσιμο έργο, καθώς η απαίτηση κάθε εφαρμογής δικτύωσης μπορεί να διαφέρει. Για παράδειγμα, μια εφαρμογή ασφάλειας μπορεί να έχει απαιτήσεις διαφορετικές από μια εφαρμογή δρομολόγησης. Η ομάδα εργασίας για το Northbound της κοινότητας ONF εργάζεται ήδη πάνω στην κοινή τυποποίηση του northbound API.
- **Southbound Interface:** Το επίπεδο ελέγχου επικοινωνεί με το επίπεδο δεδομένων χρησιμοποιώντας το Southbound, έτσι το επίπεδο ελέγχου μπορεί να δώσει δυναμικά οδηγίες σε στοιχεία του επιπέδου δεδομένων σχετικά με τον τρόπο λειτουργίας. Το Southbound παρέχει ένα μέσο επικοινωνίας μεταξύ του ελεγκτή και των συσκευών switches. Εγκαθιστά τους κατάλληλους κανόνες ροής στον πίνακα προώθησης του Switch που αποφασίζεται από τον ελεγκτή. Το OpenFlow είναι το πιο ευρέως διαδεδομένο Southbound από την κοινότητα ανοικτού κώδικα. Παρέχει διάφορες πληροφορίες για τον ελεγκτή. Δημιουργεί το μηνύματα βάσει συμβάντων σε περίπτωση αλλαγής θύρας ή σύνδεσης. Το πρωτόκολλο παράγει επίσης στατιστικά στοιχεία με βάση τη ροή για την προώθηση της συσκευής και το διαβιβάζει στον ελεγκτή. Επίσης κάνει αποστολή μηνυμάτων Packet-in στην περίπτωση που ένας δρομολογητής δεν γνωρίζει πώς να χειριστεί το νέο εισερχόμενο πακέτο.
- **Επίπεδο ελέγχου:** Το στρώμα ελέγχου, γνωστό και ως επίπεδο ελέγχου, είναι υπεύθυνο για τη διαχείριση και τον έλεγχο του συνολικού δικτύου. Αυτό το επίπεδο περιέχει ένα σημαντικό συστατικό του δικτύου, γνωστό ως ελεγκτής SDN. Είναι υπεύθυνο για τη δημιουργία και τον τερματισμό ροών δεδομένων σε διάφορα στοιχεία του δικτύου στο επίπεδο δεδομένων, με βάση τις πολιτικές χειρισμού δεδομένων. Η πρωταρχική ευθύνη είναι η λεπτομερής ρύθμιση των πινάκων προώθησης, οι οποίοι βρίσκονται στο επίπεδο προώθησης. Αυτός ο συντονισμός βασίζεται στην τοπολογία του δικτύου ή σε εξωτερικά αιτήματα υπηρεσιών. Αυτό το επίπεδο αφηρημένα αφαιρεί την πολυπλοκότητα του δικτύου διατηρώντας μια ενημερωμένη ολιστική εικόνα του δικτύου.

- **Επίπεδο δεδομένων:** Το χαμηλότερο επίπεδο στην αρχιτεκτονική SDN είναι γνωστό ως επίπεδο δεδομένων. Αυτό το επίπεδο αποτελείται από στοιχεία του δικτύου προώθησης, όπως δρομολογητές, φυσικά και εικονικά Switches, σημεία πρόσβασης και τείχη προστασίας. Ως αποτέλεσμα, αυτό το επίπεδο είναι επίσης γνωστό ως επίπεδο υποδομής. Το επίπεδο δεδομένων είναι υπεύθυνο για την υλοποίηση λειτουργιών διαχείρισης, όπως η προώθηση δεδομένων, ο κατακερματισμός και η επανασυναρμολόγηση, σύμφωνα με τις οδηγίες του ελεγκτή προς τους Switches που διαθέτουν δυνατότητα SDN. Επιπλέον, οι πληροφορίες που συλλέγονται από αυτούς τους switches. OpenFlow προωθούνται στον ελεγκτή, χρησιμοποιώντας Southbound Interface.

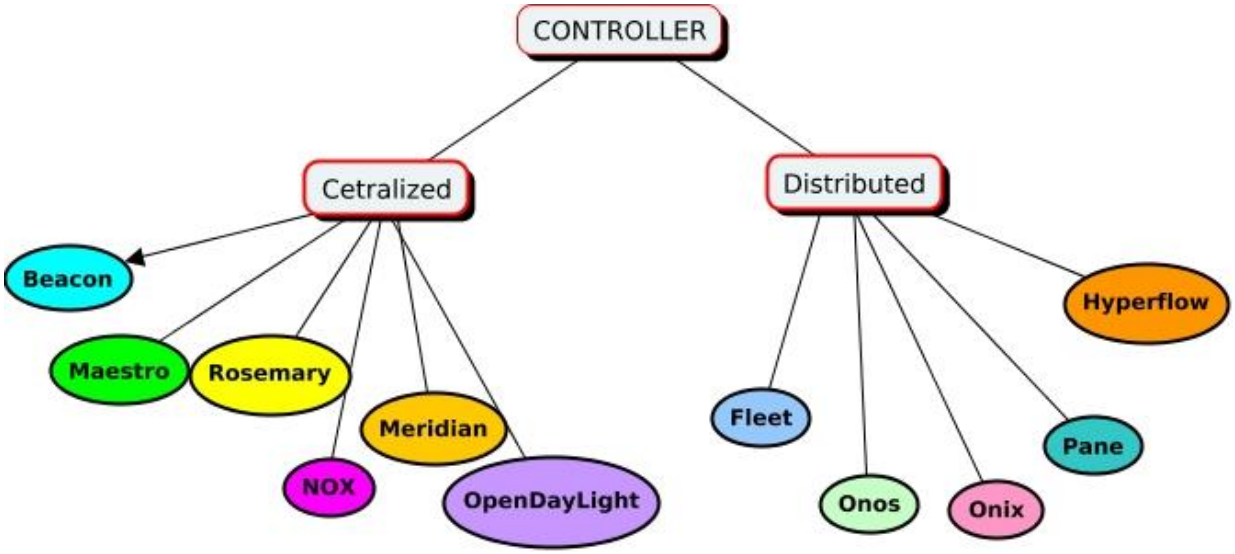
  Σύμφωνα με το ONF, η αρχιτεκτονική SDN πλεονεκτεί, καθώς είναι:

  - Ευέλικτη
  - Κεντρικά διαχειρίσιμη
  - Ρυθμισμένη μέσω προγραμματισμού
  - Βασισμένη σε ανοικτά πρότυπα και ουδέτερη ως προς τον προμηθευτή

Η αρχιτεκτονική SDN εξαρτάται σε μεγάλο βαθμό από την αποτελεσματικότητα των επιπέδου ελέγχου, όπου βρίσκεται ο ελεγκτής SDN. Για να γίνει το επίπεδο ελέγχου πιο αποδοτικό, είναι πολύ σημαντικό να έχουμε ένα ελεγκτή που όχι μόνο χρειάζεται λιγότερο χρόνο για να κάνει και να προωθήσει αποφάσεις στο επίπεδο προώθησης, αλλά και να χειρίζεται αποτελεσματικά τις σύγχρονες προκλήσεις της κυκλοφορίας του δικτύου. Η εξέλιξη της αρχιτεκτονικής SDN έχει γεννήσει μια σειρά από SDN ελεγκτές, και με τόσους πολλούς διαθέσιμους σήμερα ελεγκτές, συχνά γίνεται δύσκολο έργο η επιλογή του ελεγκτή που θα εργαστούμε.

3.2.1 Control Apps

Οι ελεγκτές SDN είναι εφαρμογές λογισμικού που τυπικά εκτελούνται σε έναν διακομιστή (server) και χρησιμοποιούν πρωτόκολλα για να ελέγχουν τη συμπεριφορά των στοιχείων στο επίπεδο υποδομής. Αυτές οι εφαρμογές είναι γραμμένες σε πολλές διαφορετικές γλώσσες προγραμματισμού υπολογιστών, όπως η Java, Python, C, C++ και Ruby. Μπορεί να είναι ανοικτού κώδικα ή να βασίζονται σε άδεια χρήσης. Οι πιο συνηθισμένοι ελεγκτές SDN που υπάρχουν σήμερα είναι ελεγκτές βασισμένοι στο OpenFlow, δηλαδή βασίζονται στο πρωτόκολλο OpenFlow για να μπορούν να επικοινωνούν με τα κατώτερα στρώματα και να καθορίζουν πως θα διαβιβαστούν τα δεδομένα στον προορισμό τους. Η δικτύωση που καθορίζεται από λογισμικό χρησιμοποιεί δύο τύπους ελεγκτών, οι οποίοι είναι ο κεντρικοποιημένος (Centralized) και ο κατανεμημένος (Distributed). Στην Εικόνα 3-2 βλέπουμε ένα παράδειγμα αυτών των κατηγοριών.



Εικόνα 3-3 Ενδεικτική ταξινόμηση ελεγκτών

- **CENTRALIZED CONTROLLER**

Οι κεντρικοί ελεγκτές υλοποιούν όλη τη λογική του επιπέδου ελέγχου σε ένα μία μόνο θέση. Σε έναν τέτοιο ελεγκτή, ο μοναδικός server αναλαμβάνει όλες τις δραστηριότητες του επιπέδου ελέγχου. Το κύριο πλεονέκτημα ενός τέτοιου ελεγκτή είναι η απλότητα και η διαχείριση, καθώς παρέχουν ένα ενιαίο σημείο ελέγχου. Ωστόσο, πάσχουν από πρόβλημα επεκτασιμότητας επειδή κάθε server έχει περιορισμένη χωρητικότητα για την αντιμετώπιση των συσκευών του επιπέδου δεδομένων.

- **DISTRIBUTED CONTROLLER**

Σε σύγκριση με τους κεντρικούς ελεγκτές, οι κατανεμημένοι ελεγκτές έχουν πλεονεκτήματα όσον αφορά την επεκτασιμότητα και τις υψηλές επιδόσεις κατά την αύξηση της ζήτησης αιτήσεων.

3.2.2 Network OS

Το λειτουργικό σύστημα δικτύου(Network Operating System) είναι ο εγκέφαλος του SDN, που υλοποιείται στον ελεγκτή (Manish Paliwal, 2018). Το NOS παρέχει βασικές λειτουργίες παρόμοιες με ένα κοινό λειτουργικό σύστημα. Αυτές οι λειτουργίες περιλαμβάνουν την εκτέλεση του προγράμματος, τη διαχείριση της λειτουργίας εισόδου/εξόδου, την παροχή ασφάλειας και προστασίας μηχανισμού προστασίας κ.λπ. Εκτός από αυτό, ένα λειτουργικό σύστημα δικτύου παρέχει λειτουργίες δικτύωσης, όπως λειτουργίες που σχετίζονται με την τοπολογία,

προώθηση δεδομένων μέσω συντομότερης διαδρομής κ.λπ. Σε αντίθεση με το παραδοσιακό δίκτυο IP, το NOS στο SDN υλοποιείται με έναν λογικά συγκεντρωτικό τρόπο.

Η έννοια του λειτουργικού συστήματος δικτύου γεννήθηκε με την εισαγωγή του λειτουργικού δικτύου που βασίζεται στο Openflow. Τα λειτουργικά συστήματα δικτύων υπάρχουν εδώ και δεκαετίες ένα από τα πιο γνωστά και διαδεδομένα είναι το Cisco IOS (V. Bollapragada, 2000), το οποίο σχεδιάστηκε αρχικά στις αρχές του 90s. Άλλα λειτουργικά συστήματα δικτύου που αξίζει να αναφερθούν είναι JUNOS, ExtremeXOS και SR OS. Είναι τα πιο εξειδικευμένα λειτουργικά συστήματα δικτύου, που στοχεύουν συσκευές δικτύου, όπως δρομολογητές και πυρήνες υψηλής απόδοσης. Αυτά τα NOS αφαιρούν το υποκείμενο υλικό από τον διαχειριστή του δικτύου, καθιστώντας ευκολότερο τον έλεγχο της υποδομής του δικτύου καθώς και την απλούστευση της ανάπτυξης και εγκατάστασης νέων πρωτοκόλλων και εφαρμογών διαχείρισης.

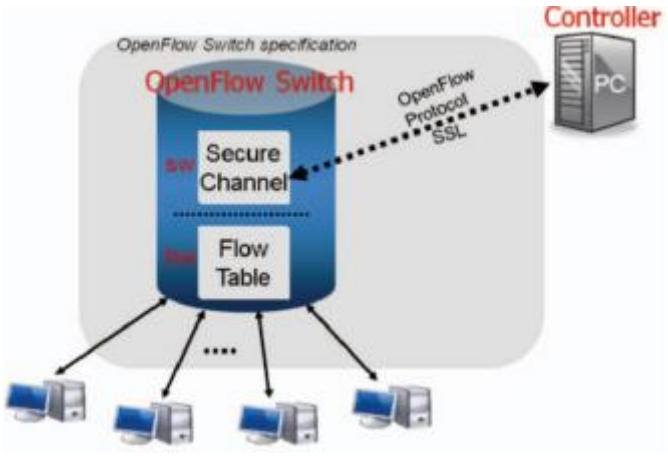
3.2.3 Switch

Παραδοσιακά το switch είναι μια δικτυακή συσκευή που χρησιμοποιείται για την προώθηση των δεδομένων προς τον προορισμός τους. Στο SDN, οι switches είναι είτε εικονικές είτε φυσικές συσκευές που παρέχουν την υποκείμενη βάση για τη μεταγωγή των δεδομένων στο περιβάλλον SDN. Οι switches SDN διαθέτουν θύρες (Ports), πίνακες (Tables) και μονάδα επεξεργασίας (Processing Unit) για τη λήψη την επεξεργασία και τη μεταγωγή των πακέτων δεδομένων. Σε κάθε μεταγωγέα εκτελείται το λειτουργικό σύστημα (Switch OS) που μπορεί να θεωρηθεί ως ανάλογο με ένα λειτουργικό σύστημα διακομιστή. Αυτό είναι υπεύθυνο για το χειρισμό κλήσεων API που απευθύνονται στο μεταγωγέα, και την εκτέλεση των κατάλληλων ενεργειών εκ μέρους του μεταγωγέα.

Βασική προϋπόθεση για τους switches SDN είναι η υποστήριξη πρωτοκόλλων SDN ώστε να είναι δυνατή η επικοινωνία στο δίκτυο και ο καθορισμός των κανόνων επικοινωνίας. Ένα τέτοιο πρωτόκολλο είναι και το OpenFlow, το οποίο είναι το αρχαιότερο και το πιο ευρέως χρησιμοποιούμενο πρωτόκολλο SDN. Οι switches που υποστηρίζουν το πρωτόκολλο OpenFlow, συνήθως καλούνται και switches OpenFlow. Η μεταγωγή καθορίζεται από τους πίνακες, που περιέχουν ταξινομητές και ένα σύνολο ενεργειών. Αν SDN switch λάβει ένα πακέτο το οποίο δεν υπάρχει αντιστοίχηση/ταιριάσμα στον πίνακα, επικοινωνεί με τον ελεγκτή SDN και ρωτάει τι πρέπει να κάνει με αυτό το πακέτο. Ο ελεγκτής μπορεί να ορίσει μια ροή, και να την κατεβάσει στο μεταγωγέα για να προχωρήσει στην προώθηση του πακέτου. Εκτός από το OpenFlow, ένας switch μπορεί να υποστηρίζει άλλα παρόμοια πρωτόκολλα SDN, όπως τα OpFlex, NETCONF, BGP (Border Gateway Protocol), XMPP (Extensible Messaging and Presence Protocol) κ.λπ.

3.3 OpenFlow

Το OpenFlow (OpenFlow, n.d.) ξεκίνησε από το πανεπιστήμιο του Στάνφορντ το 2008. Είχε ως στόχο να βοηθήσει διάφορους ερευνητές όσο αφορά την εφαρμογή διάφορων πειραματικών πρωτοκόλλων σε δίκτυα. Η αρχιτεκτονική OpenFlow αποτελείται από διάφορα στοιχεία: Τον ελεγκτή OpenFlow, τη συσκευή OpenFlow (switch) και το πρωτόκολλο OpenFlow. Στην εικόνα 6 παρουσιάζονται τα στοιχεία της αρχιτεκτονικής OpenFlow. Η προσέγγιση OpenFlow θεωρεί έναν κεντρικό ελεγκτή που διαμορφώνει όλες τις συσκευές. Συσκευές θα πρέπει να διατηρούνται απλές προκειμένου να επιτυγχάνεται καλύτερη απόδοση προώθησης και ο έλεγχος του δικτύου γίνεται από τον ελεγκτή.



Εικόνα 3-4 Αρχιτεκτονική OpenFlow

3.3.1 OpenFlow Controller

Ο ελεγκτής OpenFlow είναι ο κεντρικός ελεγκτής ενός δικτύου OpenFlow. Ρυθμίζει όλες τις συσκευές OpenFlow, διατηρεί πληροφορίες τοπολογίας και παρακολουθεί τη συνολική κατάσταση ολόκληρου του δικτύου. Η συσκευή OpenFlow είναι οποιαδήποτε συσκευή με δυνατότητα OpenFlow σε ένα δίκτυο, όπως ένα switch, ένας δρομολογητής ή ένα σημείο πρόσβασης. Κάθε συσκευή διατηρεί έναν πίνακα ροής που υποδεικνύει την επεξεργασία που εφαρμόζεται σε κάθε πακέτο μιας συγκεκριμένης ροής. Το πρωτόκολλο OpenFlow λειτουργεί ως interface μεταξύ του ελεγκτή και των switches που δημιουργούν τον πίνακα ροής και χρησιμοποιεί ένα ασφαλές κανάλι (Fernandez, 2013) που βασίζεται σε Transport Layer Security (TLS). Ο ελεγκτής Openflow παρουσιάζει δύο συμπεριφορές: την αντιδραστική και την προληπτική. Στην αντιδραστική προσέγγιση, το πρώτο πακέτο της ροής που λαμβάνει ο διακόπτης ενεργοποιεί τον ελεγκτή για την εισαγωγή ροής σε κάθε μεταγωγέα OpenFlow του δικτύου. Αυτή η προσέγγιση παρουσιάζει την πιο αποδοτική χρήση της υπάρχουσας μνήμης του πίνακα ροής, αλλά κάθε νέα ροή συνεπάγεται με έναν μικρό πρόσθετο χρόνο εγκατάστασης. Τέλος, εάν ένα switch χάσει τη σύνδεση, έχει

περιορισμένη χρησιμότητα. Στην προληπτική προσέγγιση, ο ελεγκτής προ συμπληρώνει εκ των προτέρων τον πίνακα ροής σε κάθε μεταγωγέα. Αυτή η προσέγγιση έχει μηδενικό πρόσθετο χρόνο ρύθμισης ροής, επειδή ο κανόνας προώθησης έχει οριστεί. Αν το switch χάσει τη σύνδεση με τον ελεγκτή δεν διακόπτεται η κυκλοφορία.

Όταν ένα switch λαμβάνει ένα πακέτο στο οποίο δεν υπάρχει καταχώρηση στον πίνακα ροής του, προωθεί το μήνυμα στον ελεγκτή ζητώντας την ενέργεια που πρέπει να γίνει σε αυτή τη νέα ροή. Ο ελεγκτής μπορεί να ορίσει τη θύρα που πρέπει να προωθηθεί η ροή ή να προβεί σε άλλες ενέργειες, όπως η απόρριψη του πακέτου. Ο ελεγκτής πρέπει να ορίσει ολόκληρη τη διαδρομή στέλνοντας μηνύματα διαμόρφωσης σε όλους τους switches από τον πηγή έως τον προορισμό.

3.3.2 OPENFLOW SWITCH

Ένας διακόπτης OpenFlow (cables-solutions, n.d.) είναι ένας διακόπτης switch με δυνατότητα OpenFlow που επικοινωνεί μέσω του καναλιού OpenFlow σε έναν εξωτερικό ελεγκτή. Εκτελεί αναζήτηση και προώθηση πακέτων σύμφωνα με έναν ή περισσότερους πίνακες ροής. Ο διακόπτης OpenFlow επικοινωνεί με τον ελεγκτή και ο ελεγκτής διαχειρίζεται το διακόπτη μέσω του πρωτοκόλλου διακόπτη OpenFlow. Είτε βασίζονται στο πρωτόκολλο OpenFlow είτε είναι συμβατά με αυτό.

Ένας διακόπτης OpenFlow μπορεί να λειτουργήσει μόνο με τη συνεργασία τριών βασικών στοιχείων: πίνακες ροής που είναι εγκατεστημένοι σε διακόπτες, ένας ελεγκτής και ένα αποκλειστικό πρωτόκολλο OpenFlow για τον ελεγκτή να μιλάει με ασφάλεια με switches. Οι πίνακες ροής τοποθετούνται στους switches. Οι ελεγκτές συνομιλούν με τους switches μέσω του πρωτοκόλλου OpenFlow και επιβάλλουν πολιτικές στις ροές. Ο ελεγκτής θα μπορούσε να δημιουργήσει διαδρομές μέσω του δικτύου βελτιστοποιημένες για συγκεκριμένα χαρακτηριστικά, όπως ταχύτητα, μικρότερο αριθμό αναπηδήσεων ή μειωμένη καθυστέρηση.

3.3.3 OPENFLOW PROTOCOL

Το πρωτόκολλο OpenFlow Rawal et al. (Rawal, n.d.) είναι ένα πρότυπο στην αρχιτεκτονική δικτύωσης που ορίζεται από λογισμικό (SDN). Αυτό το πρωτόκολλο ορίζει την επικοινωνία μεταξύ ενός ελεγκτή SDN και της συσκευής. Το πρωτόκολλο OpenFlow θέτει τα θεμέλια για την επικοινωνία μεταξύ ενός ελεγκτή SDN και μιας συσκευής δικτύου. Αναπτύχθηκε για πρώτη φορά από ερευνητές στο Πανεπιστήμιο του Στάνφορντ το 2008 και υιοθετήθηκε για πρώτη φορά από την Google στο δίκτυο κορμού τους το 2011-2012. Η διαχείριση του γίνεται πλέον από το Open Networking Foundation (ONF) . Η τελευταία έκδοση που χρησιμοποιείται στη βιομηχανία είναι η V1.5. Το OpenFlow είναι το Southbound πρωτόκολλο που χρησιμοποιείται μεταξύ του

ελεγκτή SDN και του switch. Ο ελεγκτής SDN παίρνει τις πληροφορίες από τις εφαρμογές και τις μετατρέπει σε καταχωρήσεις ροής, οι οποίες τροφοδοτούνται στο switch μέσω OF. Μπορεί επίσης να χρησιμοποιηθεί για την παρακολούθηση στατιστικών switches και θυρών στη διαχείριση δικτύου.

4 Ελεγκτές ανοικτού κώδικα SDN

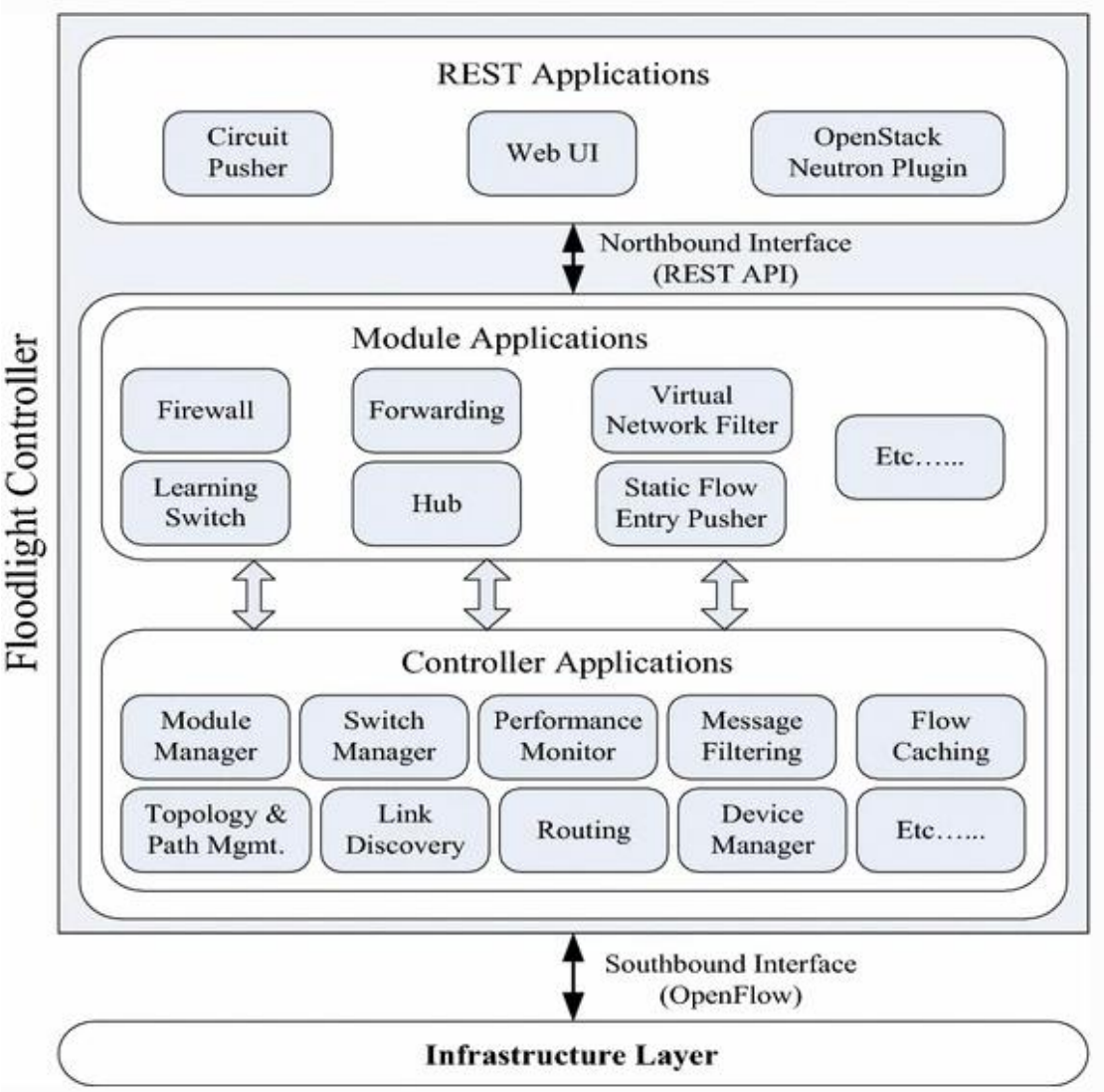
Οι SDN Controllers χωρίζονται σε δύο κατηγορίες, τους ελεγκτές ανοιχτού κώδικα που μπορεί να τους επεξεργαστεί οποιασδήποτε χρήστης και αναπτύσσονται συνεχώς, και τους εμπορικούς ελεγκτές οι οποίοι διαχειρίζονται από μεγάλες εταιρείες όπως η Cisco,IMB, Ericsson, κ.α. Σε αυτό το κεφάλαιο θα σταθούμε στους ελεγκτές ανοικτού κώδικα.

4.1 Floodlight

- **Floodlight**

Ο Floodlight (Project Floodlight, n.d.)παρουσιάζει ενδιαφέρον καθώς είναι ένας από τα διαθέσιμους ελεγκτές ανοικτού κώδικα. Υποστηρίζει το OpenFlow, το de facto πρότυπο για το Southbound API, διαθέτει πολύ καλή τεκμηρίωση και αναπτύσσεται συνεχώς. Επιπλέον, το Floodlight υποστηρίζει εικονικούς switches, διευκολύνοντας την ανάπτυξη και τη δοκιμή των πειραμάτων όσες φορές χρειάζεται σε ένα εικονικό περιβάλλον (Laura Victoria Morales, 2015). Δεδομένου ότι είναι γραμμένο σε Java, είναι αρθρωτό, υποστηρίζει multiple threads, διαχειρίζεται την μνήμη και τρέχει σε διάφορες πλατφόρμες.

Ο Floodlight δεν είναι απλώς ένας ελεγκτής OpenFlow αλλά επίσης μια συλλογή εφαρμογών που έχουν δημιουργηθεί πάνω στον ελεγκτή Floodlight. Υλοποιεί ένα σύνολο κοινών λειτουργιών για τον έλεγχο και την αναζήτηση ενός δικτύου OpenFlow, ενώ οι εφαρμογές που βρίσκονται πάνω σε αυτόν υλοποιούν διαφορετικά χαρακτηριστικά για την επίλυση διαφορετικών αναγκών των χρηστών στο δίκτυο. Η παρακάτω εικόνα δείχνει τη σχέση μεταξύ του Floodlight Controller, των εφαρμογών που έχουν κατασκευαστεί ως μονάδες Java και έχουν μεταγλωττιστεί με το Floodlight, και των εφαρμογών που έχουν κατασκευαστεί μέσω του Floodlight REST API. Το Floodlight δεν είναι μόνο ένας ελεγκτής OpenFlow, αλλά και μια συλλογή εφαρμογών οι οποίες είναι διαθέσιμες πάνω από τον ελεγκτή. Η εσωτερική αρχιτεκτονική του Floodlight ελεγκτή και των εφαρμογών που είναι διαθέσιμες, παρουσιάζεται στην εικόνα 4-1.

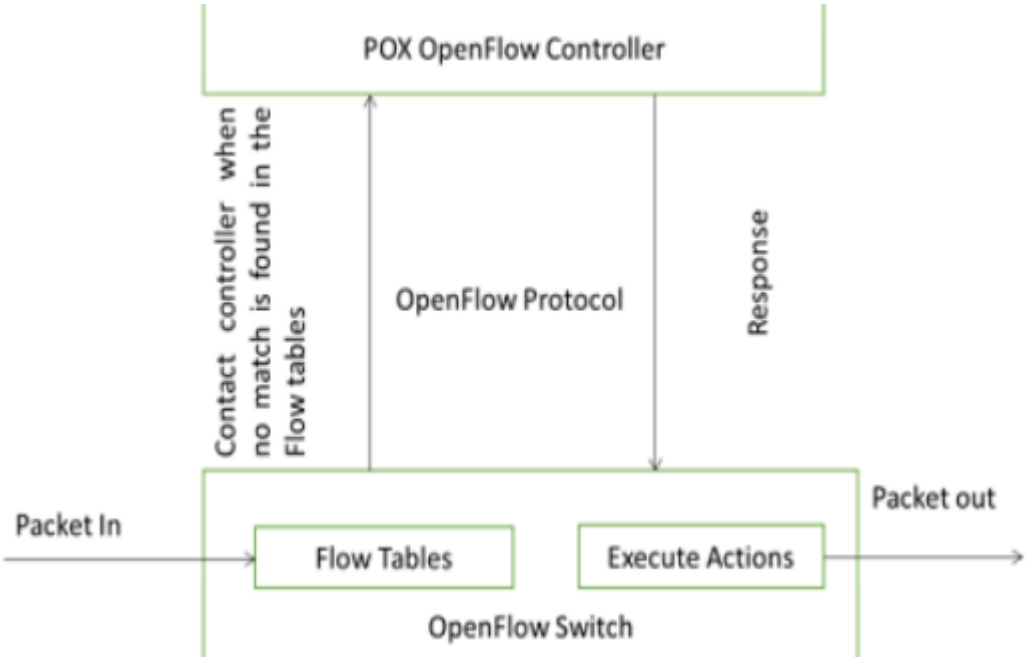


Εικόνα 4-1 Αρχιτεκτονική Floodlight (Πηγή: (Bholebawa, 2018)

4.2 POX

- **Pox**

Ο POX (Jehad Ali, 2018) είναι ένας ελεγκτής OpenFlow βασισμένος στην Python και είναι η αναβαθμισμένη έκδοση του NOX. Χρησιμοποιείται ευρέως στην ερευνητική κοινότητα λόγω της ευκολίας προγραμματισμού του. Παρέχει μια πλατφόρμα για την κατασκευή πρωτοτύπων και την ταχεία ανάπτυξη εφαρμογών για τον έλεγχο συσκευών δικτύου σε ένα OpenFlow δίκτυο. Το POX μπορεί να συνδεθεί με το Mininet και σε διάφορες εφαρμογές, όπως τείχος προστασίας, ανίχνευση εισβολής και σύστημα πρόληψης, εξισορροπητής φορτίου, μεταγωγή και δρομολόγηση μπορούν να διαμορφωθούν στους υποκείμενους OpenFlow Switches μέσω αυτού. Ο POX είναι ενσωματωμένος στο Mininet και μπορεί επίσης να εγκατασταθεί από το GitHub.



Εικόνα 4-2 Αρχιτεκτονική POX (Πηγή: (Rani, 2019))

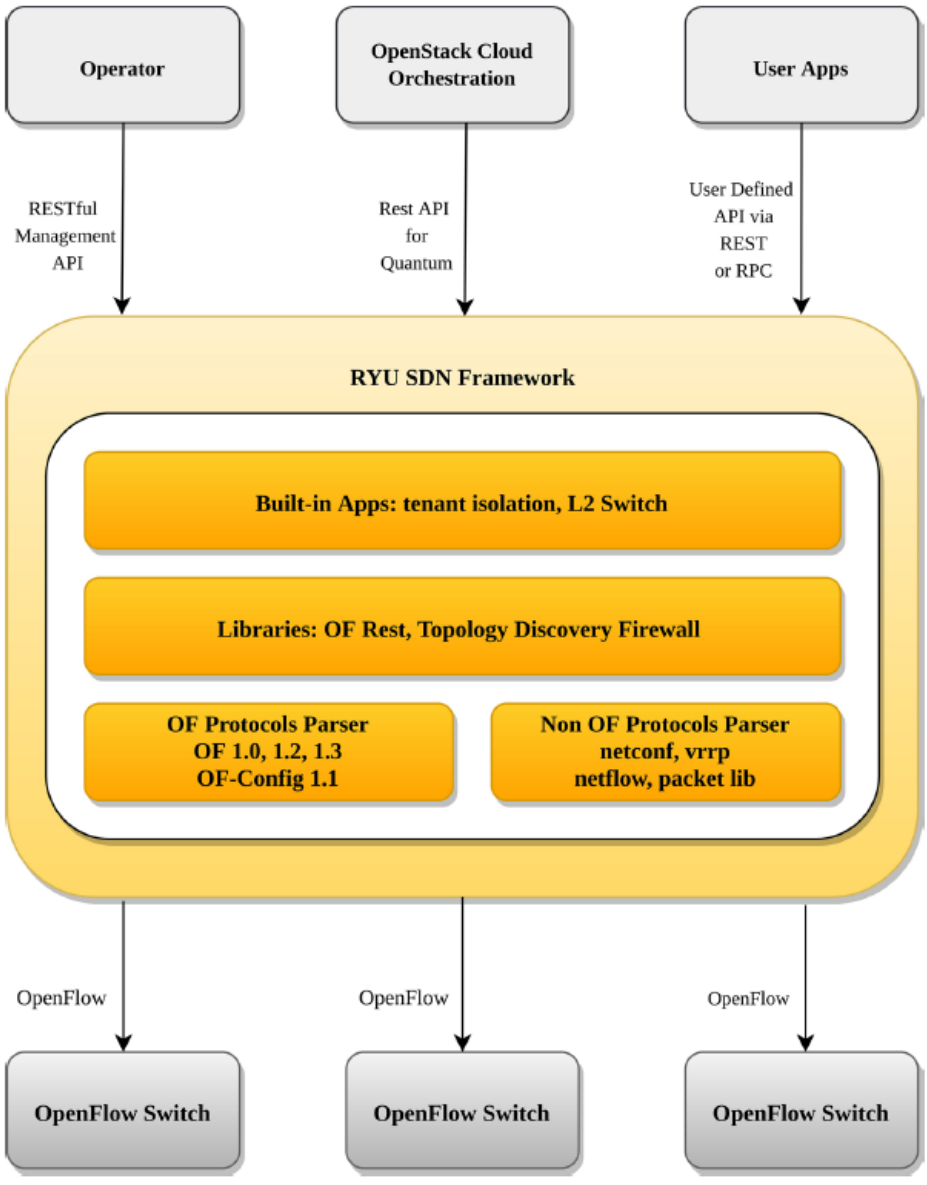
Η αρχιτεκτονική του είναι απλή σε σύγκριση με άλλους ελεγκτές. Η επικοινωνία μεταξύ του ελεγκτή και ενός Switch γίνεται με τη χρήση του πρωτοκόλλου OpenFlow. Οι OpenFlow switches συμπεριφέρονται ακριβώς όπως οι συσκευές προώθησης. Εκτελούν μόνο την εντολή από τον ελεγκτή. Όταν ένα Switch είναι ενεργοποιημένο, την επόμενη στιγμή θα συνδεθεί αμέσως με τον ελεγκτή. Κάθε switch έχει τον δικό του πίνακα ροής, ο οποίος αρχικά είναι άδειος. Κατά την άφιξη ενός πακέτου, το Switch στέλνει ένα Packet-in μήνυμα στον ελεγκτή. Στη συνέχεια, ο ελεγκτής εισάγει ολόκληρη τη ροή στον πίνακα ροής του Switch σχετικά με τον τρόπο που θα χειρίζεται το πακέτο. Καθώς κάθε πακέτο περνάει μια καταχώρηση ροής, εγκαθίσταται στο Switch έτσι ώστε να είναι σε θέση να χειριστεί το πακέτο χωρίς την παρέμβαση του ελεγκτή. Εάν η ροή καταχώρησης δεν ταιριάζει με αυτές στον ελεγκτή, στέλνει μια απάντηση για την απόρριψη του πακέτου. Τα πλεονεκτήματα της χρήσης του POX είναι ότι χρειάζεται λιγότερη μνήμη για να λειτουργήσει σε αντίθεση με άλλους ελεγκτές, αλλά έχει χαμηλή απόδοση ρυθμού μετάδοσης σε σύγκριση με άλλους ελεγκτές.

4.3 RYU

- **Ryu**

Ο RYU (Md. Tariqul Islam, Node to Node Performance Evaluation through RYU SDN Controller, 2020) είναι ένας ελεγκτής SDN, ανοικτού κώδικα που βασίζεται σε στοιχεία λογισμικού. Αναπτύχθηκε από την Nippon Telegraph and Telephone (NTT) και είναι πλήρως υλοποιημένος σε Python. Διαχειρίζεται τον έλεγχο ροής για να επιτρέψει την έξυπνη δικτύωση. Ο ελεγκτής RYU παρέχει στοιχεία

λογισμικού με καλά καθορισμένες διεπαφές προγραμματισμού εφαρμογών (Application Program Interfaces - APIs) που διευκολύνουν τους προγραμματιστές να δημιουργήσουν νέες εφαρμογές για τη διαχείριση και τον έλεγχο του δικτύου. Το RYU υποστηρίζει διάφορα πρωτόκολλα για τη διαχείριση συσκευών δικτύου, όπως OpenFlow, Netconf, OF-config κ.λπ. Σχετικά με το OpenFlow, το RYU υποστηρίζει πλήρως τις εκδόσεις 1.0, 1.2, 1.3, 1.4, 1.5. Η αρχιτεκτονική του Ryu απεικονίζεται στην Εικόνα 4-3.



Εικόνα 4-3 Αρχιτεκτονική Ryu (Πηγή: (Md. Tariqul Islam, SpringerLink, 2020)

Ο ελεγκτής RYU έχει τρία επίπεδα (Εικόνα 4-3). Το ανώτερο επίπεδο, το ονομαζόμενο επίπεδο εφαρμογής, που περιλαμβάνει τις εφαρμογές δικτυακής λογικής. Το μεσαίο επίπεδο που αποτελείται από υπηρεσίες δικτύου, και είναι

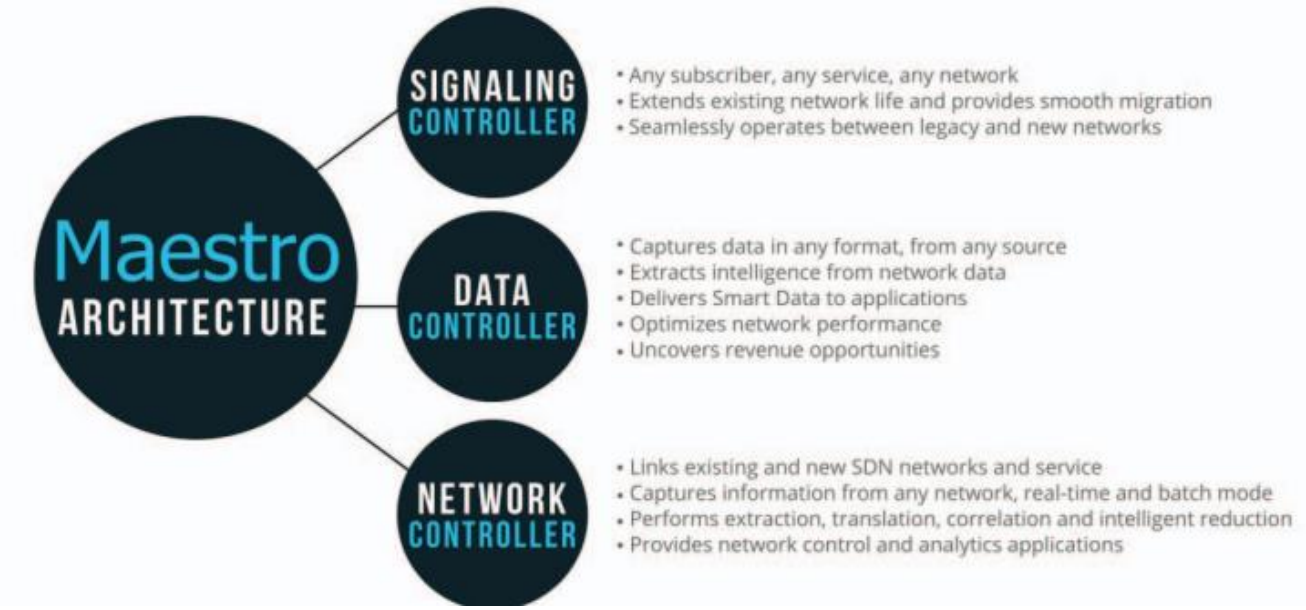
γνωστό ως επίπεδο ελέγχου ή πλαίσιο SDN, και το κατώτερο επίπεδο που αποτελείται από φυσικές και εικονικές συσκευές, και είναι γνωστό ως επίπεδο υποδομής. Το μεσαίο επίπεδο φιλοξενεί Northbound και Southbound APIs. Το RYU χρησιμοποιεί το OpenFlow για να αλληλοεπιδράσει με το επίπεδο προώθησης (Switches και δρομολογητές) για να τροποποιήσει τον τρόπο με τον οποίο το δίκτυο θα χειριστεί τις ροές κυκλοφορίας. Έχει δοκιμαστεί και πιστοποιηθεί για να λειτουργεί με διάφορους Switches OpenFlow, συμπεριλαμβανομένου του OpenvSwitch αλλά και των προτάσεων διάφορων εταιρειών όπως Centec, Hewlett Packard, IBM και NEC.

4.4 Maestro

- **Maestro**

Το Maestro είναι ένα «λειτουργικό σύστημα» για την ενορχήστρωση εφαρμογών ελέγχου δικτύου (Google Code, n.d.). Παρέχει interfaces για την υλοποίηση εφαρμογών ελέγχου δικτύου για πρόσβαση και τροποποίηση της κατάστασης του δικτύου και συντονισμό των αλληλεπιδράσεων τους. Επιπλέον, το Maestro προσπαθεί να εκμεταλλευτεί τον παραλληλισμό μέσα σε ένα μόνο μηχάνημα για να βελτιώσει την απόδοση του συστήματος. Το Maestro είναι φορητό και επεκτάσιμο, αναπτύχθηκε σε Java (τόσο η πλατφόρμα όσο και τα εξαρτήματα), εξαιρετικά φορητό σε διάφορα λειτουργικά συστήματα και αρχιτεκτονικές. Τέλος θα μπορούσαμε να πούμε ότι είναι ‘’ Multi-threaded’’ καθώς εκμεταλλεύεται πλήρως τους επεξεργαστές πολλαπλών πυρήνων.

Το πλαίσιο προγραμματισμού του Maestro παρέχει Interfaces για: την εισαγωγή νέων προσαρμοσμένων λειτουργιών ελέγχου με την προσθήκη στοιχείων ελέγχου, τη διατήρηση της κατάστασης του δικτύου για λογαριασμό των στοιχείων ελέγχου, τη σύνθεση στοιχείων ελέγχου προσδιορίζοντας την αλληλουχία εκτέλεσης και τις κοινή κατάσταση δικτύου των στοιχείων, και την παροχή ελέγχου για την υλοποίηση εκμάθησης είτε ενός switch δικτύου, είτε ενός δικτύου δρομολόγησης με χρήση Switches OpenFlow, όπως φαίνεται στην εικόνα 4-4.



Εικόνα 4-4 Αρχιτεκτονική Maestro (Πηγή: (Madhukrishna Priyadarsini, 2017)

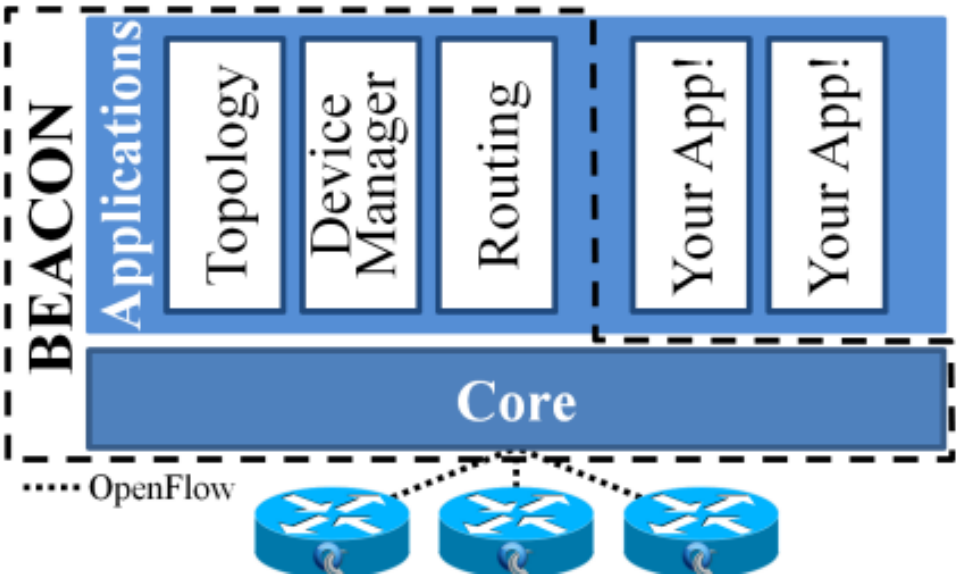
4.5 Beacon

➢ **Beacon**

Το Beacon αξιοποιεί πολλαπλές έτοιμες βιβλιοθήκες σε μια προσπάθεια να μεγιστοποιήσει την επαναχρησιμοποίηση του κώδικα και να διευκολύνει το βάρος της ανάπτυξης τόσο των του ίδιου του ελεγκτή, όσο και των εφαρμογών που τον χρησιμοποιούν (Erickson, The Beacon OpenFlow Controller, 2013). Η πιο σημαντική βιβλιοθήκη είναι η Spring. Το (API) για το Beacon έχει σχεδιαστεί ώστε να είναι απλό και να μην επιβάλλει ουσιαστικά κανέναν περιορισμό, έτσι ώστε οι προγραμματιστές είναι ελεύθεροι να χρησιμοποιήσουν οποιεσδήποτε διαθέσιμες δομές Java, όπως threads, χρονοδιακόπτες, υποδοχές, κ.λπ. Δίνει έμφαση στο να είναι φιλικός προς τους προγραμματιστές και να έχει υψηλές επιδόσεις, και στη δυνατότητα εκκίνησης και διακοπής υφιστάμενων και νέων εφαρμογών κατά την εκτέλεση.

Το Beacon περιλαμβάνει επίσης εφαρμογές αναφοράς που βασίζονται στην πυρήνα, προσθέτοντας επιπλέον API:

- **Διαχείριση συσκευών(Device Manager):** Παρακολουθεί τις συσκευές που εμφανίζονται στο δίκτυο, συμπεριλαμβανομένων τις διευθύνσεις τους (Ethernet και IP), την ημερομηνία τελευταίας εμφάνισης, καθώς και το switch και την θύρα στην οποία εθεάθη τελευταία φορά. Η Διαχείριση συσκευών παρέχει ένα Interface (IDeviceManager) για την αναζήτηση γνωστών συσκευών και τη δυνατότητα εγγραφής ώστε να λαμβάνει συμβάντα όταν προστίθενται, ενημερώνονται ή αφαιρούνται νέες συσκευές.



Εικόνα 4-5 Αρχιτεκτονική Beacon (Πηγή: (Erickson, The Beacon OpenFlow Controller, 2013)

- **Τοπολογία(Topology):** Ανακαλύπτει συνδέσεις μεταξύ συνδεδεμένων Switches OpenFlow. Το Interface του (ITopology) επιτρέπει την ανάκτηση ενός καταλόγου τέτοιων συνδέσεων, και την καταχώριση συμβάντων για την ειδοποίηση όταν προστίθενται ή αφαιρούνται συνδέσεις.
- **Δρομολόγηση(Routing):** Παρέχει δρομολόγηση συντομότερης διαδρομής επιπέδου δύο μεταξύ των συσκευών του δικτύου. Η εφαρμογή αυτή εξάγει την IRoutingEngine interface, επιτρέποντας εναλλάξιμες υλοποιήσεις μηχανών δρομολόγησης. Η περιλαμβανόμενη υλοποίηση χρησιμοποιεί το συντομότερο μονοπάτι όλων των ζευγών μεθόδου υπολογισμού. Η δρομολόγηση εξαρτάται τόσο από την Τοπολογία όσο και από τη Διαχείριση Συσκευών.
- **Web:** Παρέχει ένα Web UI για το Beacon. Η εφαρμογή Web παρέχει το Interface IWebManageable, επιτρέποντας στους υλοποιητές της διασύνδεσης να προσθέσουν τα δικά τους στοιχεία UI.

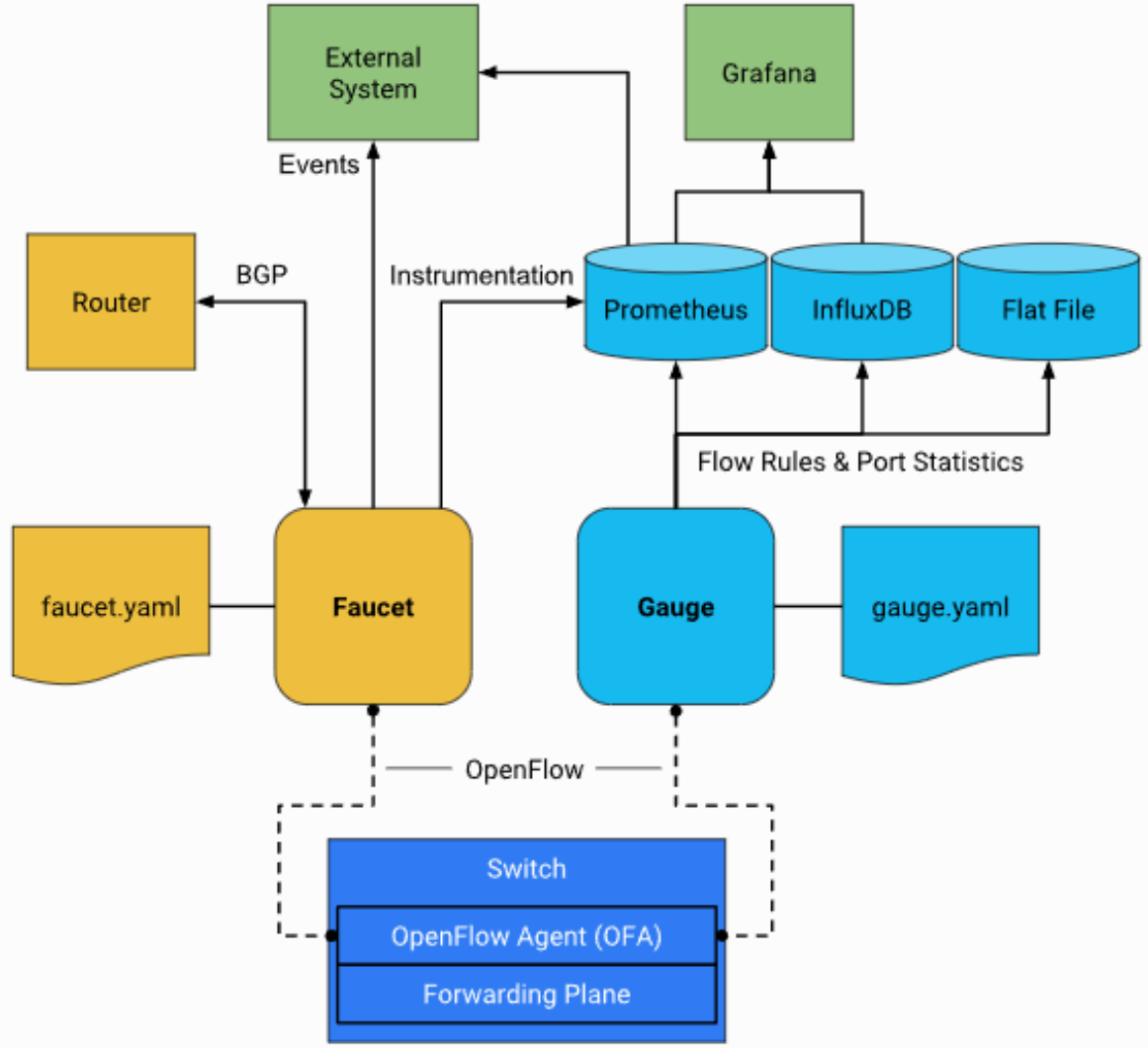
4.6 Faucet

➢ **Faucet**

Έχει αναπτυχθεί σε διάφορα περιβάλλοντα, συμπεριλαμβανομένου του Open Networking Foundation, το οποίο εκτελεί μια παρουσία του Faucet ως δίκτυο γραφείου. Το Faucet BAILEY et al. (JOSH BAILEY, 2016) παρέχει υψηλές επιδόσεις προώθησης χρησιμοποιώντας υλικό μεταγωγής, ενώ επιτρέπει στους φορείς να προσθέτουν χαρακτηριστικά τα δίκτυά τους και να τα αναπτύσσουν γρήγορα, σε πολλές περιπτώσεις χωρίς να χρειάζεται η αλλαγή (ή ακόμη και επανεκκίνηση)

υλικού. Επιπλέον, συνεργάζεται με γειτονικές συσκευές δικτύου μη SDN. Το Faucet βασίστηκε στο OpenFlow 1.3.

Όπως φαίνεται στην εικόνα 4-6, αρχιτεκτονικά, κάθε στιγμιότυπο Faucet έχει δύο συνδέσεις με τους υποκείμενους Switches. Το ένα για ενημερώσεις ελέγχου και διαμόρφωσης, το άλλο (Gauge) είναι μια σύνδεση μόνο για ανάγνωση, ειδικά για συλλογή, ταξινόμηση και μετάδοση πληροφοριών κατάστασης για επεξεργασία αλλού χρησιμοποιώντας Influxdb ή Prometheus.



Εικόνα 4-6 Αρχιτεκτονική Faucet (Πηγή: (faucet.nz, n.d.))

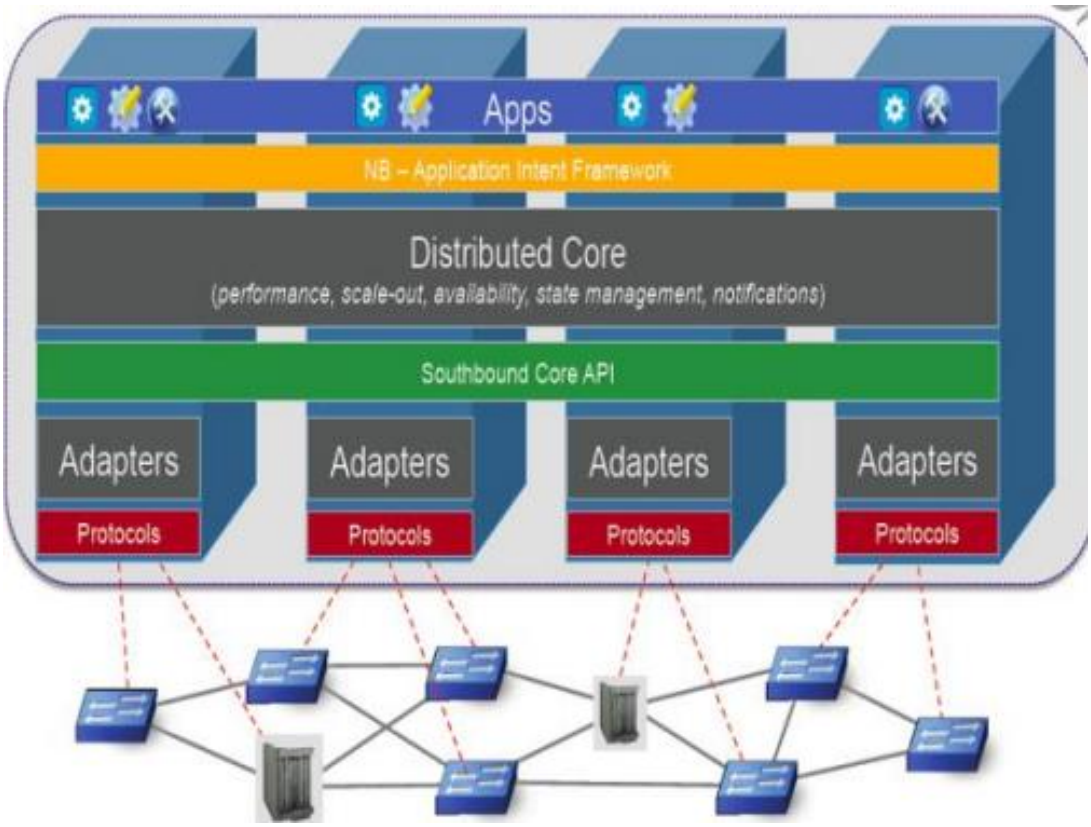
4.7 ONOS

- **Onos**

Το Open Network Operating System (ONOS) (opennetworking, n.d.) είναι ελεγκτής SDN ανοιχτού κώδικα για την δημιουργία λύσεων SDN/NFV επόμενης γενιάς. Το ONOS σχεδιάστηκε για να καλύψει τις ανάγκες των φορέων εκμετάλλευσης που επιθυμούν να δημιουργήσουν λύσεις ποιότητας, οι οποίες αξιοποιούν τα οικονομικά πλεονεκτήματα του υλικού, προσφέροντας παράλληλα την ευελιξία για τη δημιουργία και την εξάπλωση νέων δυναμικών υπηρεσιών δικτύου με απλοποιημένες προγραμματιστικές διεπαφές. Το ONOS υποστηρίζει τόσο τη ρύθμιση παραμέτρων όσο και τον έλεγχο του δικτύου σε πραγματικό χρόνο, απαλείφοντας την ανάγκη εκτέλεσης πρωτοκόλλων ελέγχου δρομολόγησης και εναλλαγής εντός του ιστού δικτύου. Διαβιβάζοντας τη νοημοσύνη στον ελεγκτή cloud ONOS, η καινοτομία ενεργοποιείται και οι τελικοί χρήστες μπορούν εύκολα να δημιουργήσουν νέες εφαρμογές δικτύου χωρίς να πρέπει να αλλάξουν τα συστήματα του επιπέδου δεδομένων. . Μια σημαντική εγκυκλοπαιδική πληροφορία για τον Onos, είναι πως είναι η μόνη πλατφόρμα ελεγκτών SDN που υποστηρίζει τη μετάβαση από τα παλαιά δίκτυα σε δίκτυα SDN. Αυτό επιτρέπει νέες συναρπαστικές δυνατότητες και ανατρεπτικά σημεία ανάπτυξης και λειτουργικού κόστους για τους φορείς εκμετάλλευσης δικτύων.

Η πλατφόρμα ONOS περιλαμβάνει:

- Μια πλατφόρμα και ένα σύνολο εφαρμογών που λειτουργούν ως επεκτάσιμος, αρθρωτός, κατανεμημένος ελεγκτής SDN.
- Απλοποιημένη διαχείριση, διαμόρφωση και ανάπτυξη νέου λογισμικού, υλικού και υπηρεσιών.
- Μια αρχιτεκτονική κλιμάκωσης για την παροχή της ανθεκτικότητας και της επεκτασιμότητας που απαιτούνται για να ανταποκριθεί στις δυσκολίες των περιβαλλόντων παραγωγής.



Εικόνα 4-7 Αρχιτεκτονική ONOS (Πηγή: (Mohammed Sameer, 2018))

Όπως βλέπουμε στην εικόνα 4-7, οι εφαρμογές επικοινωνούν για τον προγραμματισμό του υποκείμενου υλικού. Το Northbound Interface ασχολείται με τις εφαρμογές και το framework για την εφαρμογή πολιτικών και την επίλυση συγκρούσεων. Ο κατανεμημένος πυρήνας φροντίζει για την απόδοση και την διαχείριση. Το Southbound εκτελεί τη διαμόρφωση, προγραμματίζει τα προγραμματιζόμενα hardwares περαιτέρω και παρατηρεί για ενημερώσεις. Το Southbound ασχολείται επίσης με πρωτόκολλα όπως το OpenFlow και τα μελλοντικά API.

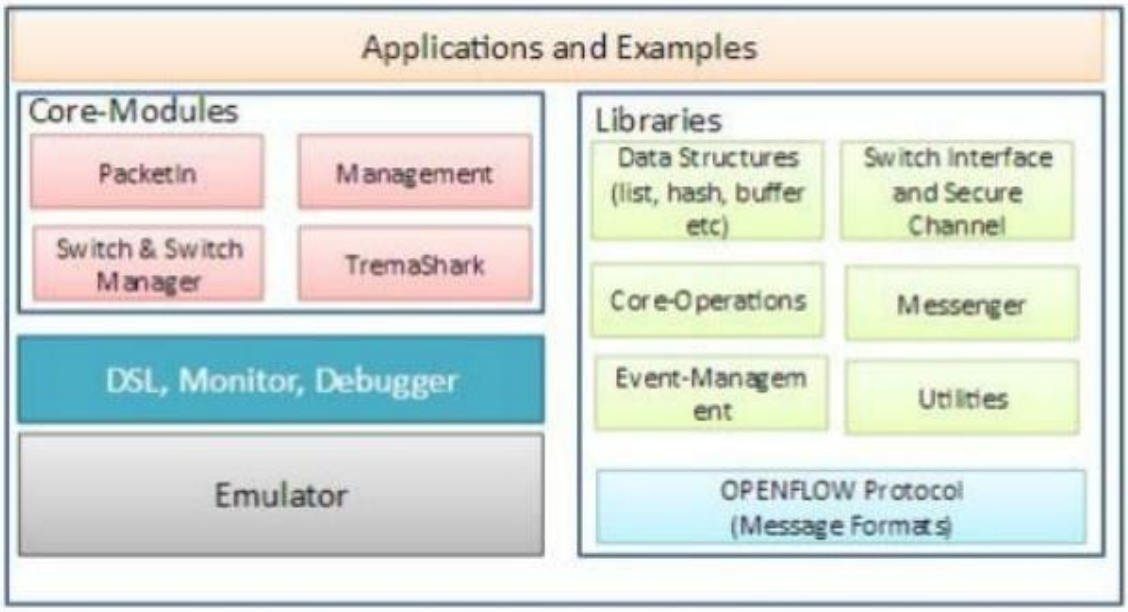
4.8 Trema

- **Trema**

Το Trema Rao et al. (Rao, 2014) είναι ένα πλαίσιο προγραμματισμού ελεγκτών OpenFlow που παρέχει όλα όσα απαιτούνται για τη δημιουργία ελεγκτών OpenFlow στη Ruby. Παρέχει μια υψηλού επιπέδου βιβλιοθήκη OpenFlow και επίσης έναν εξομοιωτή δικτύου που μπορεί να δημιουργήσει δίκτυα βασισμένα σε OpenFlow. Αυτό το αυτόνομο περιβάλλον βοηθά στη διαδικασία ανάπτυξης

και δοκιμής. Με το Trema, ο όρος ‘’framework’’ χρησιμοποιείται για να τονίσει το γεγονός ότι ο χρήστης έχει την ελευθερία να διαμορφώσει και να δημιουργήσει έναν ελεγκτή OpenFlow. Το ‘’framework’’, είναι σχεδιασμένο για να παρέχει επεκτασιμότητα, περιλαμβάνει όλες τις απαραίτητες βιβλιοθήκες και λειτουργίες που είναι απαραίτητες για την αλληλεπίδραση με τους OpenFlow switches . Με υποστήριξη και για τις γλώσσες C και Ruby, ο Trema δίνει στον χρήστη την ελευθερία να επιλέξει με βάση το επίπεδο άνεσης και τις απαιτήσεις απόδοσης. Το Trema είναι περισσότερο μια πλατφόρμα λογισμικού για προγραμματιστές/ερευνητές/λάτρεις του OpenFlow παρά ένας ελεγκτής παραγωγής. Για παράδειγμα, εάν ο χρήστης θέλει έναν ελεγκτή OpenFlow για τη διαχείριση της τοπολογίας του δικτύου, πρέπει απλώς να δημιουργήσει και να εκτελέσει το Trema με μια υπάρχουσα εφαρμογή διαχείρισης τοπολογίας.

Στην εικόνα 4-8 βλέπουμε την αρχιτεκτονική του ελεγκτή Trema:



Εικόνα 4-8 Αρχιτεκτονική Trema (Πηγή: (Rao, 2014))

Η αρχιτεκτονική Trema περιλαμβάνει βασικές ενότητες όπως φίλτρο εισόδου πακέτων (packet-in filter), switch και switch manager, εφαρμογή OpenFlow Interface, απαραίτητες βιβλιοθήκες, DSL δικτύου για διαμορφώσεις, εξομοιωτή για ολοκληρωμένη ανάπτυξη και Tremashark για την υποστήριξη της αποσφαλμάτωσης. Το Trema χρησιμοποιεί ένα μοντέλο πολλαπλών διεργασιών, στο οποίο πολλές λειτουργικές ενότητες συνδέονται.

- **Core Modules**

Οι βασικές ενότητες του πλαισίου Trema περιλαμβάνουν ως επί το πλείστων θεμελιώδεις μονάδες OpenFlow και συνήθως αυτές που είναι χρήσιμες για πολλαπλές εφαρμογές.

- **Switch and Switch-Manager**

Αυτές οι δύο βασικές μονάδες υλοποιούν τις απαραίτητες λειτουργίες για αλληλεπιδράσεις με τους διακόπτες OpenFlow. Διατηρούν επίσης όλες τις απαραίτητες πληροφορίες σχετικά με τους διακόπτες.

- **Packet-In Filter**

Το μήνυμα Packet-In είναι ένας τρόπος για τον διακόπτη OpenFlow να στείλει ένα καταγεγραμμένο πακέτο στον ελεγκτή. Το Switch στέλνει τα πακέτα στον ελεγκτή μόνο όταν του ζητηθεί από τον ελεγκτή ή όταν δεν υπάρχει κατάλληλη καταχώρηση στον πίνακα ροής του μεταγωγέα.

- **TremaShark**

Το Tremashark είναι το πρόσθετο Wireshark για την παρακολούθηση τυχόν συμβάντων επικοινωνίας μεταξύ των λειτουργικών μονάδων. Τα συμβάντα μπορεί να διαφέρουν από μηνύματα σε κατάσταση ασφαλούς καναλιού έως καταστάσεις ουράς.

- **Βιβλιοθήκες**
  1. Πρωτόκολλο: OpenFlow
  2. Διεπαφές: Εφαρμογή Openflow, Switch, Διαχείριση
  3. Δομές δεδομένων: hash table, timers, Linked list, doubly-linked list
  4. Βοηθητικές υπηρεσίες : Αρχείο καταγραφής, στατιστικά στοιχεία, timers
  5. Πρωτόκολλα δικτύου: TCP, IP, UDP, ICMP και IGMP.

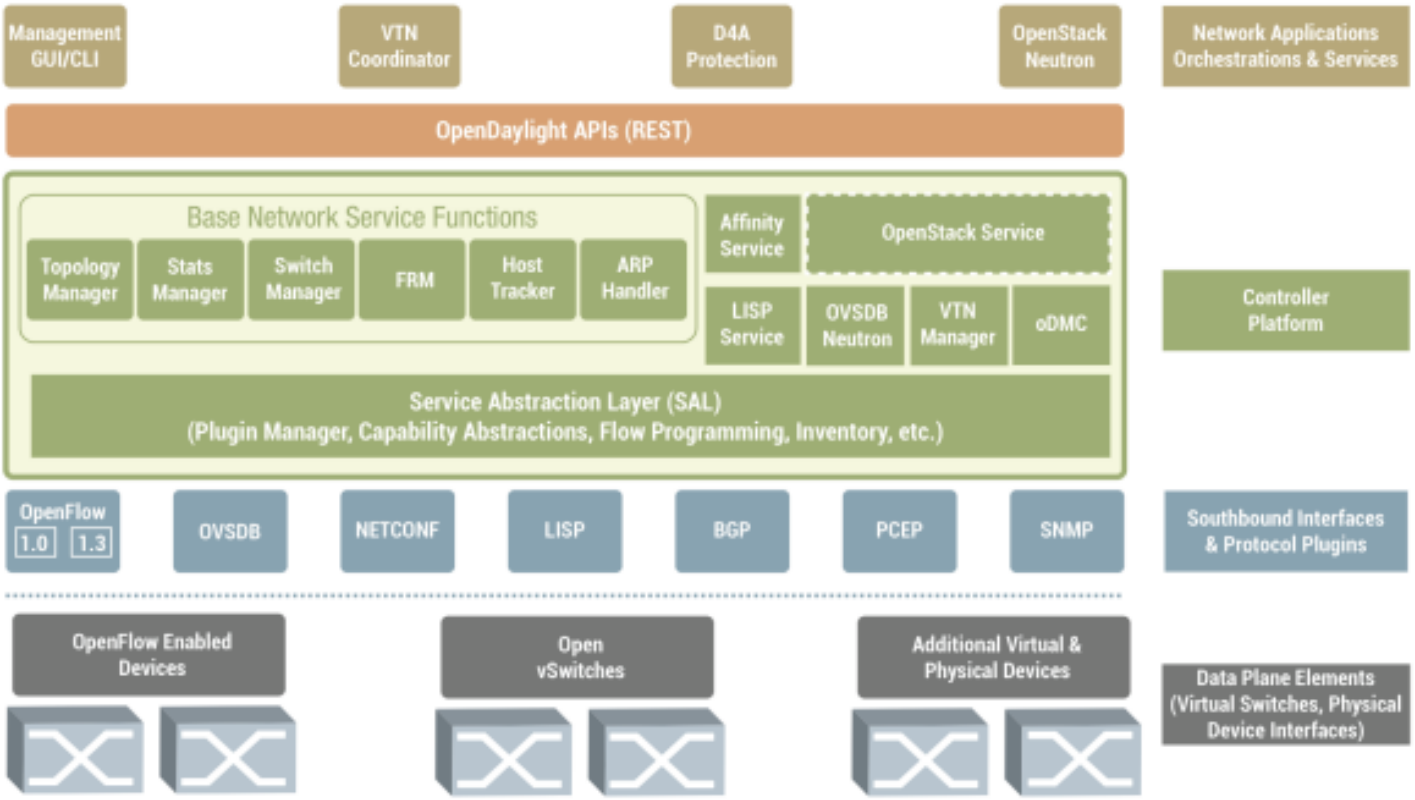
4.9 OpenDayLight

- **OpenDayLight**

Ο OpenDayLight Badotra et al. (Sumit Badotra J. S., 2017) είναι ένας ελεγκτής ανοιχτού κώδικα, για την προσαρμογή και την αυτοματοποίηση δικτύων οποιουδήποτε μεγέθους και κλίμακας. Το έργο OpenDaylight προέκυψε από το δίκτυο SDN, με σαφή εστίαση στον προγραμματισμό του δικτύου. Σχεδιάστηκε εξαρχής ως θεμέλιο για εμπορικές λύσεις που καλύπτουν ποικίλες περιπτώσεις χρήσης σε υφιστάμενα περιβάλλοντα δικτύων. Υποστηρίζεται από την IBM, Cisco, Juniper, VMWare και πολλές άλλες μεγάλες εταιρείες δικτύωσης. Το OpenDaylight είναι μια πλατφόρμα ελεγκτή SDN που υλοποιείται σε Java. Ως εκ τούτου, μπορεί να αναπτυχθεί σε οποιοδήποτε υλικό και πλατφόρμα λειτουργικού συστήματος που υποστηρίζει Java. Παρέχει υπηρεσίες με βάση το μοντέλο (service abstraction layer)

που επιτρέπει στους χρήστες να αναπτύσσουν εφαρμογές που λειτουργούν εύκολα σε ένα ευρύ φάσμα υλικού και Southbound πρωτοκόλλων. Βασίζεται στις ακόλουθες τεχνολογίες GARCIA et al. (GARCIA, 2015):

- Maven: Εργαλείο διαχείρισης έργων που απλοποιεί και αυτοματοποιεί τις εξαρτήσεις μεταξύ ενός έργου ή διαφορετικών έργων. Αυτό το εργαλείο θα βοηθήσει τους προγραμματιστές να διαχειριστούν όλες τα απαιτούμενα plugins, καθώς και να παρέχει την εκκίνηση ενός έργου χρησιμοποιώντας τα καθορισμένα του αρχικά πρότυπα.
- Java: Είναι η γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάπτυξη εφαρμογών και λειτουργιών στον ελεγκτή OpenDaylight. Η ανάπτυξη σε Java παρέχει πολύτιμη ασφάλεια κατά τη μεταγλώττιση, καθώς και έναν εύκολο τρόπο υλοποίησης καθορισμένων υπηρεσιών.
- Open Service Gateway Interface (OSGi): Είναι το back-end του OpenDaylight καθώς επιτρέπει να φορτώνει δυναμικά δέσμες και πακέτα JAR (συνθέτουν τις εφαρμογές) και να δεσμεύει ενότητες μεταξύ τους για την ανταλλαγή πληροφοριών.
- Karaf: Πρόκειται για ένα ‘’δοχείο’’ εφαρμογών που είναι χτισμένο πάνω στο OSGi, το οποίο απλοποιεί και βοηθάει την εγκατάσταση εφαρμογών.
- YANG: Είναι το σημείο-κλειδί της συμπεριφοράς του ελεγκτή. Οι προγραμματιστές χρησιμοποιούν το YANG για να μοντελοποιήσουν μια λειτουργικότητα εφαρμογής και να δημιουργήσουν APIs από τα καθορισμένα μοντέλα, τα οποία θα χρησιμοποιηθούν αργότερα για την παροχή των υλοποιήσεών της. Το YANG υποστηρίζει τη μοντελοποίηση λειτουργικών δεδομένων και δεδομένων διαμόρφωσης, καθώς και RPC και notifications.



Εικόνα 4-9 Αρχιτεκτονική OpenDayLight (Πηγή: (opendaylight.org, χ.χ.)

Ο ελεγκτής OpenDaylight έχει πολλά επίπεδα (Zuhran Khan Khattak, 2014). Το ανώτερο επίπεδο αποτελείται από εφαρμογές επιχειρησιακής και δικτυακής λογικής. Το μεσαίο είναι το επίπεδο πλαισίου και το κάτω επίπεδο αποτελείται από φυσικές και εικονικές συσκευές. Στο μεσαίο επίπεδο φιλοξενούνται Northbound και Southbound APIs. Ο ελεγκτής εκθέτει ανοικτά Northbound APIs τα οποία χρησιμοποιούνται από τις εφαρμογές. Το OpenDaylight υποστηρίζει το OSGi όπως προαναφέραμε για το Northbound API. Η επιχειρησιακή λογική βρίσκεται στις εφαρμογές πάνω από τον μεσαίο επίπεδο. Αυτές οι εφαρμογές χρησιμοποιούν τον ελεγκτή για να συγκεντρώνουν πληροφορίες δικτύου, την εκτέλεση αλγορίθμων για την εκτέλεση αναλύσεων και στη συνέχεια χρησιμοποιούν τον ελεγκτή για την ενορχήστρωση των νέων κανόνων, εάν υπάρχουν, σε όλο το δίκτυο.

Με βάση του τέσσερις ελεγκτές που θα χρησιμοποιήσουμε για να κάνουμε τις μετρήσεις στην συνέχεια, δηλαδή του Ryu, Floodlight, Onos και OpenDayLight προσδιορίσαμε δύο πίνακες κριτηρίων, σύμφωνα με τις προαναφερόμενες σχετικές εργασίες που είδαμε.

Πίνακας 2 Γενικά Χαρακτηριστικά

	Ryu	Floodlight	Onos	OpenDayLight
Διαθέσιμο Υλικό	Καλό	Μέτριο	Καλό	Καλό
Δημοτικότητα	Υψηλή	Υψηλή	Υψηλή	Υψηλή
Ανοικτού Κώδικα	Ναι	Ναι	Ναι	Ναι
Επιστημονική κοινότητα	Μικρή	Μεγάλη (Μεγάλες εταιρείες δικτύων)	Μεγάλη,Cisco, Ericsson, Fujitsu, Huawei, Intel, Nec)	Μεγάλη (Cisco, IBM, NEC)
Εικονικοποίηση	Mininet & Open vSwitch	Mininet & Open vSwitch	Mininet	Mininet
Υλικό για την εγκατάσταση	Καλό	Ανεπαρκές	Καλό	Καλό

Πίνακας 3 Τεχνικά χαρακτηριστικά

	Ryu	Floodlight	Onos	OpenDayLight
Αρχιτεκτονική	Κεντρικοποιημένη	Κεντρικοποιημένη	Κατανεμημένη	Κατανεμημένη
Γλώσσα προγραμματισμού	python	java	java	java
Γραφικό περιβάλλον	Ναι	Ναι	Ναι	Ναι
Ευκολία εκτέλεσης	Ναι	Όχι	Ναι	Όχι
Αρθρωτότητα και επεκτασιμότητα	Μέτρια	Υψηλή	Υψηλή	Υψηλή
Απόδοση	Υψηλή	Υψηλή	Υψηλή	Υψηλή
Καθυστέρηση	Υψηλή	Μικρή	Μικρή	Πολύ μικρή

5 Εργαλεία συγκριτικής αξιολόγησης

Οι θεωρητικές συγκρίσεις που γίνονται κατά καιρούς από διάφορους ερευνητές όσον αφορά τους ελεγκτές SDN, δεν αντικατοπτρίζουν τις πραγματικές τους επιδόσεις. Για την όσο πιο λεπτομερή μελέτη των ελεγκτών πρέπει να γίνονται συγκριτικές αξιολογήσεις. Η αξιολόγηση ή η συγκριτική αξιολόγηση της απόδοσης ενός ελεγκτή μπορεί να γίνει είτε μέσω προσομοίωση/εξομοίωση ή με τη χρήση ενός εικονικού περιβάλλοντος βασισμένου σε λογισμικό (ZHU, 2020). Παρόλο που τα περιβάλλοντα δοκιμών λογισμικού παρέχουν μετρήσεις που είναι πιο κοντά στις πραγματικές τιμές σε περιβάλλον παραγωγής, το κόστος τους είναι σημαντικό για την ερευνητική κοινότητα. Ως εκ τούτου, οι αξιολογήσεις με βάση την εξομοίωση αποτελούν κοινή πρακτική. Παρ' όλα αυτά, για τη συγκριτική αξιολόγηση ελεγκτών SDN, το εργαλείο λογισμικού που χρησιμοποιείται πρέπει να είναι εξαιρετικά αποτελεσματικό και ακριβές. Σε αυτή την ενότητα, παρουσιάζουμε ορισμένα γνωστά εργαλεία που θα χρησιμοποιήσουμε για τη συγκριτική μας αξιολόγηση, καθώς επίσης και την ανάλυση των ιδιοτήτων τους και των δυνατοτήτων συγκριτικής αξιολόγησης.

Οι μετρήσεις που μπορούμε να λάβουμε από τα εργαλεία που θα δούμε παρακάτω είναι οι εξής:

- **Μετρήσεις απόδοσης:** Η απόδοση μετριέται συνήθως ως ρυθμός επεξεργασίας αιτημάτων ροής από τον ελεγκτή. Από την πλευρά των εργαλείων η απόδοση στην ουσία είναι ο αριθμός των μηνυμάτων ‘’packet-in’’ που αποστέλλονται, και ο αντίστοιχος αριθμός μηνυμάτων ‘’packet-out’’ που λαμβάνονται ανά μονάδα χρόνου.

- **Μετρήσεις καθυστέρησης**: Αυτή η ομάδα μετρήσεων μετράτε σε μονάδες χρόνου. Παρόμοια με την απόδοση, ασχολείται μόνο με την αποστολή πακέτων στον ελεγκτή και της λήψης απάντησης στο vSwitch. Ορισμένοι παράγοντες μπορούν να επηρεάσουν την καθυστέρηση ενός ελεγκτή, συμπεριλαμβανομένου του χρόνου υπολογισμού που απαιτεί ο ελεγκτής και η καθυστέρηση σύνδεσης.

- **Μετρήσεις ροής**: Οι μετρήσεις αυτές αφορούν την πλήρη παροχή διαδρομής και την εγκατάσταση ροής. Η πρωταρχική διαφορά μεταξύ αυτών και της απόδοσης είναι η πλήρης διαδρομή. Η απόδοση μετρά μόνο το ρυθμό από το vSwitch στον ελεγκτή και πίσω στο vSwitch. Ωστόσο, η πλήρης εγκατάσταση ροής απαιτεί την εγκατάσταση καταχωρήσεων ροής σε άλλους vSwitches κατά μήκος της διαδρομής.

- **Μετρήσεις με βάση την τοπολογία**: Η ικανότητα ανίχνευσης ή προσδιορισμού μιας τοπολογίας, συμπεριλαμβανομένου του τύπου της (απλή, γραμμική, δέντρο), το μέγεθος και τον αριθμό των ολοκληρωμένων κόμβων που αντιπροσωπεύουν συνολικά μια σημαντική πτυχή για την αξιολόγηση της αποτελεσματικότητας ενός ελεγκτή. Η αλληλεπίδραση με το Southbound παίζει επίσης σημαντικό ρόλο σε αυτές τις μετρήσεις.

- **Μετρήσεις threads**: Αυτό το σύνολο μετρήσεων προσδιορίζει την ικανότητα του ελεγκτή σε σχέση με τη χρήση του συστήματος αρχιτεκτονικής, των δυνατοτήτων υλικού και των μονάδων εισόδου/εξόδου. Βελτιστοποίηση των δυνατοτήτων που βασίζονται σε threads, όπως το multi-threading, προσφέρει αρκετές πλεονεκτήματα της ομαδοποίησης εργασιών, του χρονοπρογραμματισμού συμβάντων, των ροών διεργασιών ως ομάδες και κυρίως αυξάνει τη ροή του ελεγκτή.

- **Διάφορες μετρήσεις**: Εδώ ομαδοποιούμε άλλες παραμέτρους οι οποίες μπορούν επίσης να χρησιμοποιηθούν για την αξιολόγηση των ελεγκτών. Μερικές από αυτές μπορεί να είναι κρίσιμες σε εξειδικευμένα σενάρια. Για παράδειγμα η κατανάλωση ενέργειας σε κινητά περιβάλλοντα όπου οι ελεγκτές αναπτύσσονται σε συσκευές με περιορισμένη ενέργεια. Ομοίως, σε καταστάσεις όπου η αποτυχία του hardware αποτελεί ανησυχία, πρέπει να μειωθεί ο χρόνος εναλλαγής αποτυχίας, ώστε οι εφεδρικοί ελεγκτές να μπορούν να αναλάβουν το συντομότερο δυνατό.

5.1 CBENCH

Σύμφωνα με το Open Networking Foundation (htt1), το CBench μπορεί να οριστεί ως ένα ’’(Controller Benchmarker)’’ δηλαδή ένα πρόγραμμα για τη δοκιμή ελεγκτών OpenFlow με τη δημιουργία εισόδου πακέτων για νέες ροές. Με την τρέχουσα έκδοση του λογισμικού CBench, δύο κύριες δοκιμές αναφοράς για την απόδοση και την καθυστέρηση είναι διαθέσιμες. Είναι ένα εργαλείο ανοιχτού κώδικα, αλλά λόγω περιορισμένης συμβατότητας οι ελεγκτές με την έκδοση OpenFlow 1.3 μπορεί να παρουσιάσουν θέματα επιδόσεων, καθώς το CBench υποστηρίζει την έκδοση OpenFlow 1.0 και έχει χρόνια να ενημερωθεί. Εκτελείται ουσιαστικά με δύο τρόπους: απόδοσης και καθυστέρησης. Στη λειτουργία απόδοσης, στέλνει όσο το δυνατόν περισσότερα πακέτα για τον υπολογισμό του μέγιστου αριθμού πακέτων που χειρίζεται ο ελεγκτής. Στην λειτουργία καθυστέρησης, στέλνει ένα πακέτο και περιμένει την απάντηση για τον υπολογισμό του χρόνου που απαιτείται για την επεξεργασία ενός πακέτου από τον ελεγκτή. Το CBench και ο

ελεγχόμενος ελεγκτής μπορούν να λειτουργούν στο ίδιο μηχάνημα ή σε διαφορετικά μηχανήματα.

Κάποιες βασικές εντολές για την εκτέλεση του CBench μπορούμε να τις βρούμε με την βοήθεια του Mininet πληκτρολογώντας την εντολή ‘’cbench -h’’. Κάποιες από αυτές τις βλέπουμε στον παρακάτω πίνακα:

Εντολή	**Περιγραφή**	**Default τιμές**
-c/--controller	Το hostname του controller που θα συνδεθεί.	("localhost")
-l/--loops	Επαναλήψεις ανά δοκιμή.	(16)
-m/--ms-per test	Χρονική διάρκεια σε ms ανά δοκιμή.	(10000 ms)
p/--port	Το port του ελεγκτή.	(6633)
-s/--switches	Εικονικοί switches.	(16)
-t/--throughput	Λειτουργία απόδοσης αντί για καθυστέρησης.	
-D/--delay received (in ms)	Η καθυστέρηση που θα δεχθούμε σε ms.	
-i/--connect delay to the controller	Η καθυστέρηση σύνδεσης στον controller.	
-M/--mac addresses	Μοναδική διεύθυνση Mac ανά switch.	(100000)

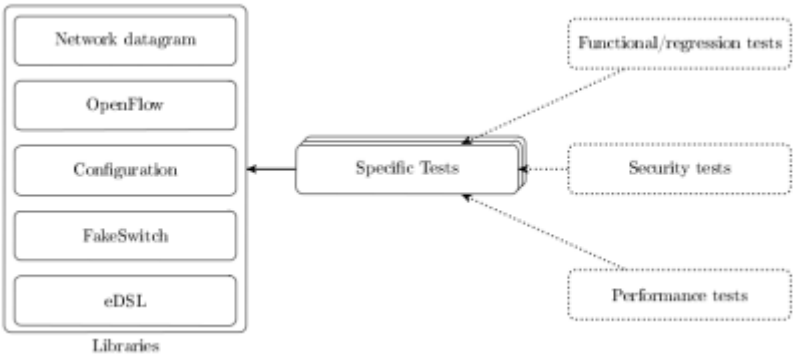
5.2 HCPROBE

Παρόμοιό με τα συμβατικά benchmarks, το Hcprobe μιμείται κάθε αριθμό switches και hosts OpenFlow που είναι συνδεδεμένοι σε έναν ελεγκτή. Είναι γραμμένο σε γλώσσα Haskell και επιτρέπει στους χρήστες να δημιουργήσουν εύκολα τα δικά τους σενάρια για δοκιμές ελεγκτών (Shalimov, 2013). Επίσης, περιλαμβάνει μια υλοποίηση αναφοράς ενός OpenFlow switch και μια γλώσσα ειδικού τομέα (EDSL) για την κατασκευή νέων προσαρμοσμένων μεταγωγέων και τη συγγραφή προγραμμάτων για αυτούς.

Τα βασικά χαρακτηριστικά του Hcprobe είναι:

- Σωστή παραγωγή πακέτων OpenFlow
- Υλοποίηση σε γλώσσα υψηλού επιπέδου, η οποία καθιστά ευκολότερη την επέκτασή του
- API για τον σχεδιασμό προσαρμοσμένων δοκιμών
- Εισαγωγή μιας ενσωματωμένης γλώσσας ειδικού τομέα (eDSL) για τη δημιουργία δοκιμών

Στην παρακάτω εικόνα βλέπουμε την αρχιτεκτονική του Hcprobe.



Εικόνα 5-1 Αρχιτεκτονική Hcprobe

- **Network datagram:** Είναι μια βιβλιοθήκη, η οποία παρέχει παρέχει βοήθεια στην δημιουργία πακέτων για πρωτόκολλα δικτύου όπως πακέτα IP, πακέτα UDP, πακέτα TCP. Επιτρέπει να ορίσετε μια προσαρμοσμένη τιμή ‘έτσι ώστε δημιουργήσει ένα προσαρμοσμένο ωφέλιμο φορτίο.
- **OpenFlow:** Η βιβλιοθήκη για την ανάλυση και τη δημιουργία μηνυμάτων OpenFlow.
- **Configuration:** Η βιβλιοθήκη που δίνει την δυνατότητα για τη διαμόρφωση των παραμέτρων των δοκιμών με τη χρήση επιλογών γραμμής εντολών ή ενός αρχείου διαμόρφωσης.
- **Fake-Switch:** Βασικός "εικονικός διακόπτης", ο οποίος διαχειρίζεται την αλληλεπίδραση με τον ελεγκτή.
- **eDSL:** Κάνει ευκολότερη την δημιουργία εικονικών δοκιμών.

Χρησιμοποιώντας τις προαναφερόμενες βιβλιοθήκες και μονάδες της αρχιτεκτονικής του Hcprobe, είναι δυνατή η δημιουργία συγκεκριμένων δοκιμών για ελεγκτές SDN/OpenFlow, συμπεριλαμβανομένων λειτουργικών δοκιμών, δοκιμών ασφάλειας και επιδόσεων. Ο πιο βολικός τρόπος για τη δημιουργία δοκιμών είναι η χρήση της ενσωματωμένης γλώσσας ειδικού τομέα. Η ειδική για τον τομέα γλώσσα Hcprobe παρέχει τα εξής ικανότητες:

- Δημιουργία OpenFlow Switches.
- Δυνατότητα δημιουργίας προγραμμάτων που περιγράφουν την λογική ενός switch αλλά και εκτέλεσης προγραμμάτων, για διαφορετικούς switches ταυτόχρονα.
- Δημιουργία διάφορων τύπων μηνυμάτων OpenFlow με προσαρμοσμένες τιμές πεδίων και επίσης να δημιουργία προσαρμοσμένου ωφέλιμου φορτίου για Packet-in μηνύματα.

- Συγκεντρωτικά στατιστικά στοιχεία για την ανταλλαγή μηνυμάτων μεταξύ OpenFlow και Switches.

Έτσι, το Hcprobe παρέχει το πλαίσιο για τη δημιουργία διαφόρων περιπτώσεων δοκιμής για τη μελέτη της συμπεριφοράς των ελεγκτών OpenFlow που επεξεργάζονται διαφορετικούς τύπους μηνυμάτων. Κάποιος μπορεί να δημιουργήσει όλα τους τύπους μηνυμάτων OpenFlow από τον switch στον ελεγκτή και να αναπαράγει διαφορετικά σενάρια επικοινωνίας, ακόμη και τα πιο περίπλοκα, τα οποία δεν μπορούν εύκολα να αναπαραχθούν με τη χρήση OpenFlow switches υλικού και λογισμικού. Το εργαλείο θα μπορούσε να είναι χρήσιμο για τους προγραμματιστές για την εκτέλεση λειτουργικότητας, παλινδρόμησης και δοκιμές ασφαλείας των ελεγκτών.

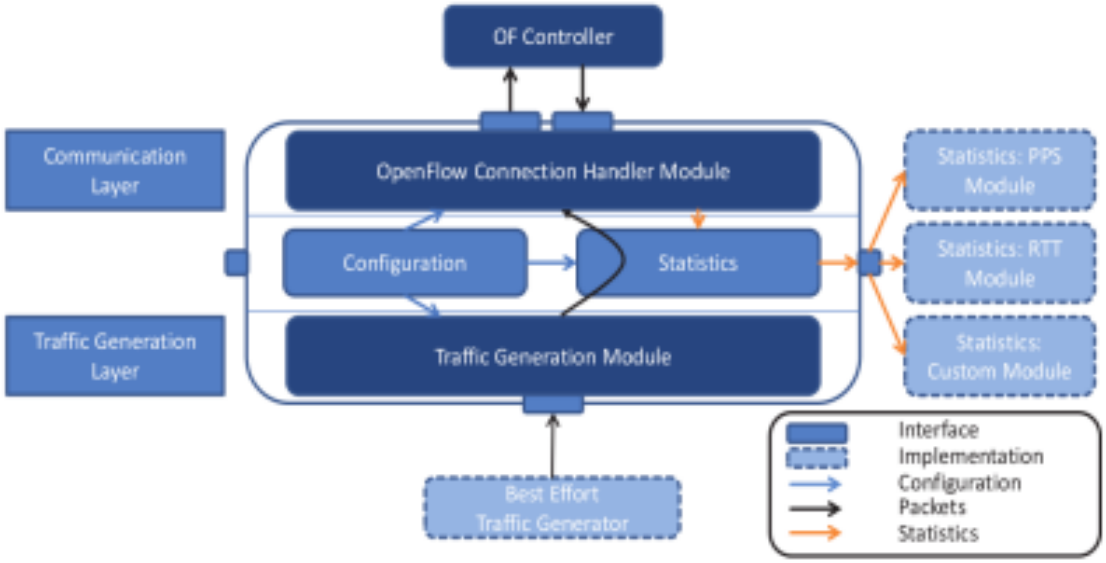
5.3 OFCProbe

Το OFCProbe είναι ένα ανεξάρτητο εργαλείο πλατφόρμας για τη συγκριτική αξιολόγηση ελεγκτών SDN. Φτάνει πέντε σημαντικά πράγματα που είναι πραγματικά απαραίτητα για μια καλή αξιολόγηση των ελεγκτών (Michael larschel, 2014):

1. Ανεξαρτησία πλατφόρμας: Αυτό το λογισμικό μπορεί να εκτελεστεί στο πιο κοινό σύστημα αρχιτεκτονικής, επειδή έχει αναπτυχθεί σε Java.
2. Επεκτασιμότητα: Το OFCProbe έχει δυνατότητα multi-threading και επίσης η επιβάρυνση threading του θα πρέπει να είναι μειωμένη σε σύγκριση με άλλα εργαλεία συγκριτικής αξιολόγησης.
3. Modularity: Αυτό είναι πραγματικά σημαντικό προκειμένου να προσαρμοστεί το εργαλείο σε μια πιθανή νέα έκδοση πρωτοκόλλων επικοινωνίας, και το OFCProbe το επιτυγχάνει αυτό λόγω του διαχωρισμού της λογικής του προγράμματος και της επικοινωνίας του ελεγκτή.
4. Ανάλυση επιδόσεων: Το εργαλείο παράγει διαφορετικές ποσότητες κίνησης προς τον ελεγκτή και μπορεί να καταγράφει τις απαντήσεις του ελεγκτή επιτρέποντας την ανάλυση της συμπεριφοράς του ελεγκτή σε αυτά τα διαφορετικά σενάρια.
5. Λεπτομερή στατιστικά στοιχεία: Αυτό το χαρακτηριστικό επιτρέπει τη διερεύνηση της συμπεριφοράς του ελεγκτή, όπως η καθυστέρηση ή η χρήση της CPU του εν λόγω ελεγκτή. Έτσι μπορούμε να μελετήσουμε αυτή τη συμπεριφορά σε διαφορετικά περιβάλλοντα, όπως η τοπολογία ή ο αριθμός των συσκευών που είναι συνδεδεμένες στον ελεγκτή.

Εκτός από όλα αυτά τα χαρακτηριστικά, το OFCProbe μιμείται πολύ καλά τη συμπεριφορά του δικτύου, δίνοντας στον διαχειριστή την ευκαιρία να εξομοιώσει την τοπολογία, τα πρότυπα κίνησης και τα πακέτα ARP, προκειμένου να αξιολογήσει

τους ελεγκτές σε ένα πιο ρεαλιστικό περιβάλλον. Παρακάτω βλέπουμε (Εικόνα 4) βλέπουμε την αρχιτεκτονική του OFCProbe.



Εικόνα 5-2 Αρχιτεκτονική OFCProbe

Όπως μπορούμε να δούμε στην Εικόνα 4, το εργαλείο χωρίζεται σε τέσσερα σημαντικά τμήματα-ενότητες: ενότητα χειρισμού συνδέσεων(Connection handler module, ενότητα διαμόρφωσης (configuration module), ενότητα στατιστικών στοιχείων(statistics module) και ενότητα κίνησης παραγωγής (traffic generation module).

- **Η μονάδα χειρισμού συνδέσεων OpenFlow:**

Η συγκεκριμένη μονάδα εκτελεί 3 βασικές λειτουργίες. Η πρώτη είναι η δημιουργία και ο χειρισμός της σύνδεση μεταξύ του εργαλείου και του ελεγκτή. Για τον σκοπό αυτό, το OFCProbe χρησιμοποιεί το NIO Selector της Java. Η δεύτερη κύρια εργασία αυτής της μονάδας είναι η διαχείριση των πινάκων ροής των εικονικών Switches. Καθώς το εργαλείο πρέπει να στέλνει όσα πακέτα θέλουμε, οι πίνακες ροής των εικονικών Switches δεν είναι τόσο σημαντικοί. Κάθε φορά που ένας εικονικός Switch θα θέλει να στείλει ένα πακέτο σε έναν άλλο εικονικό Switch, θα στέλνει ένα packet-In στον ελεγκτή χωρίς να ελέγξει το πακέτο στους πίνακες ροής. Η τρίτη και τελευταία εργασία, αλλά όχι η λιγότερο σημαντική, είναι η αποδοχή των μηνυμάτων από τη "μονάδα παραγωγής κίνησης"(Traffic Generation Module), η ενθυλάκωση τους σε OpenFlow μηνύματα και η αποστολή τους στον ελεγκτή. Πρέπει να διαχειρίζεται τις απαντήσεις για τον ελεγκτή, και να τις στείλει στη μονάδα στατιστικών στοιχείων προκειμένου να τις επεξεργαστεί και να είναι έτοιμες για περαιτέρω ανάλυση. Είναι σημαντικό να εξηγήσουμε πώς αυτή η ενότητα στέλνει τα πακέτα στον ελεγκτή, προκειμένου να κατανοήσουμε τη συμπεριφορά του εργαλείου. Εάν ορίσουμε την επιλογή "αποστολή δέσμης" (”batch sending’’) στο

αρχείο ρυθμίσεων, το εργαλείο θα προσπαθήσει να στείλει όλα τα πακέτα στην ουρά πριν διαβάσει τα πακέτα που έλαβε από τον ελεγκτή. Αυτό σημαίνει ότι το εργαλείο στέλνει όλα τα πακέτα μαζί και στη συνέχεια περιμένει.

- **Η Μονάδα Δημιουργίας Κυκλοφορίας:**

Αυτή η μονάδα έχει το δικό της thread και είναι ένας επεξεργαστής ουρών αναμονής χρονοπρογραμματισμού με βάση τα γεγονότα. Η μονάδα διαθέτει μία ουρά με όλα τα συμβάντα της προσομοίωσης. Ένα συμβάν αποτελείται από χρόνο συμβάντος, τύπο συμβάντος και έναν κατάλογο εικονικών Switches στους οποίους πρέπει να μεταφερθεί αυτό το συμβάν και να παραχθεί. Όταν ένα συμβάν επεξεργάζεται για αυτή τη μονάδα, καλεί διάφορες ενέργειες στους εικονικούς Switches του συμβάντος ανάλογα με τον τύπο του συμβάντος (ARP EVENT, PACKET IN EVENT, DISCONNECT EVENT, κ.λπ.).

- **Η Ενότητα Στατιστικής:**

Κάθε Packet-In που αποστέλλεται από κάθε Switch και το Packet-Out ή Flow Mod που λαμβάνεται, αποστέλλονται σε αυτή τη μονάδα προκειμένου να τα επεξεργαστεί και να λάβει κάποιες πληροφορίες, όπως: χρόνοι άφιξης, τύπος πακέτου κ.λπ. Όταν η εξομοίωση τελειώσει και η μονάδα έχει όλα τα πακέτα που αποστέλλονται και λαμβάνονται από το εργαλείο, μπορεί να λάβει πολύ περισσότερες πληροφορίες για κάθε εικονικό Switch και φυσικά για τον ελεγκτή. Μπορεί να δείξει τον αριθμό των πακέτων ανά δευτερόλεπτο που αποστέλλονται και λαμβάνονται από κάθε Switch, το RTT για κάθε πακέτο και Switch, καθώς και την CPU και τη RAM του ελεγκτή. Για αυτά τα τελευταία στατιστικά στοιχεία, εμείς πρέπει να εφαρμόσουμε το SNMP στον ελεγκτή.

- **Η ενότητα διαμόρφωσης:**

Όπως ο χρόνος προσομοίωσης, ο αριθμός των Switches, το αρχείο τοπολογίας, η IP του ελεγκτή κ.λπ. Μπορούμε να τροποποιήσουμε αυτές τις παραμέτρους επεξεργαζόμενοι το αρχείο διαμόρφωσης που πρέπει να βρίσκεται στον ίδιο κατάλογο του εργαλείου μας. Είναι πραγματικά διαισθητικό να αλλάξουμε τις τιμές αυτού του αρχείου επειδή όλες εξηγούνται σε αυτό το αρχείο.

Μιλώντας για την εσωτερική συμπεριφορά του εργαλείου, θα πρέπει να γράψουμε για τα threads που τρέχουν σε αυτό. Όπως προαναφέραμε το OFCProbe είναι multi-thread. Μπορούμε να ρυθμίσουμε τον αριθμό των threads του εργαλείου με τη διαμόρφωση αρχείου, αλλά θα έχουμε πάντα ένα thread για τη μονάδα παραγωγής κίνησης. Τα εικονικά switches θα μπορούν να εκτελούνται

με το δικό τους thread ή να μοιράζονται ένα thread μεταξύ περισσότερων από έναν εικονικό Switch.

5.4 Iperf tool

Το ‘’iperf’’ (Madhukrishna Priyadarsini, 2017) είναι ένα εργαλείο που χρησιμοποιείται για χαρακτηριστικές μετρήσεις των πρωτοκόλλων TCP και UDP σε δίκτυα IP. Με τον καθορισμό διαφόρων παραμέτρων και των χαρακτηριστικών του πρωτοκόλλου TCP/UDP, ο χρήστης είναι σε θέση να εκτελέσει έναν αριθμό δοκιμών που παρέχουν μια διορατικότητα σχετικά με τη διαθεσιμότητα του εύρους ζώνης του δικτύου, την καθυστέρηση, το Jitter και τα δεδομένα απώλεια δεδομένων.

6 Πλαίσιο σύγκρισης ελεγκτών SDN

6.1 Μεθοδολογία

Στην παρούσα εργασία αξιολογήθηκαν οι ελεγκτές Ryu, ONOS, OpenDayLight και Floodlight. Πιο συγκεκριμένα χρησιμοποιήσαμε όσον αφορά τον ONOS την έκδοση 2.2 (ONOS, χ.χ.) και στον Ryu έχουμε την έκδοση 4.34 (RYU, χ.χ.). Όσον αφορά τον OpenDayLight κατεβάσαμε για τα πειράματα μας μια παλιότερη έκδοση, και πιο συγκεκριμένα την Carbon 0.6.4 (OpenDayLight, n.d.), λόγω του ότι οι πιο καινούργιες εκδόσεις δεν υποστηρίζουν κάποιες παραμέτρους που χρειαζόμασταν για να τρέξουμε τις τοπολογίες. Τέλος στον floodlight τρέξαμε την έκδοση v1.2 (Floodlight, n.d.). Για την επιλογή των συγκεκριμένων ελεγκτών βασιστήκαμε σε δύο κριτήρια: α)τη δημοτικότητά τους σύμφωνα με τις σχετικές αναφορές στο Google Scholar και β) τη σειρά ταξινόμησης του προϊόντος (Product Rating) όπως αποτυπώνεται σε διάφορους ιστότοπους.

Σε κάθε πείραμα στην εκάστοτε τοπολογία, οι μετρήσεις έγιναν μεταξύ δύο hosts, όπου ο ένας host λειτουργούσε ως server και ο άλλος ως client.Οι μετρήσεις που πάρθηκαν αφορούν την απόδοση, την καθυστέρηση την διακύμανση και την τυπική απόκλιση. Η διάρκεια του κάθε πειράματος ήταν συνολικής διάρκειας 1 λεπτού, όπου ανά 5 δευτερόλεπτα παίρναμε τα αποτελέσματα, με την ορθή χρήση της εντολής ‘’iperf’’ στο TCP αλλά και στο UDP.

6.2 Τοπολογίες

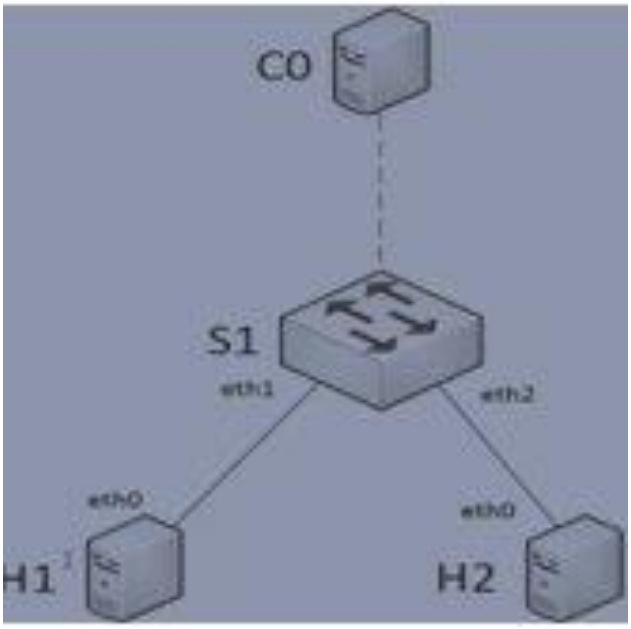
Μια τοπολογία αντιπροσωπεύει τη φυσική διάταξη του δικτύου. Η τοπολογία ορίζει τον τρόπο που οι δικτυακές συσκευές συνδέονται μεταξύ τους σε ένα δίκτυο και επηρεάζει βασικές λειτουργίες του όπως η δρομολόγηση, την ποιότητα υπηρεσιών όπως throughput, καθυστέρηση και τον τρόπο -ευκολία κατασκευής.

Στα πλαίσια αυτής της εργασίας επιλέγονται δυο τοπολογίες:

- Απλή τοπολογία
- Τοπολογία FatTree

6.2.1 Απλή τοπολογία

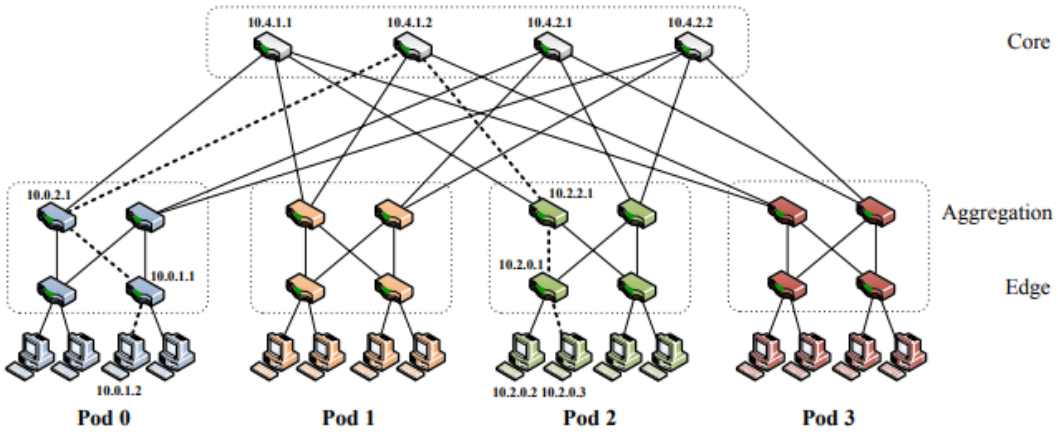
Μια απλή τοπολογία αποτελείται από ένα switch και *N* hosts που συνδέονται με το switch.Το switch επικοινωνεί με τον ελεγκτή και σύμφωνα με τις οδηγίες του μεταφέρει/μεταδίδει τα δεδομένα από host σε host. Μια τυπική τοπολογία με 2 hosts φαίνεται στην Εικόνα 6-1.



…

6.2.2 Τοπολογία FatTree

Η τοπολογία FatTree είναι θεωρείται μια αποτελεσματική επικοινωνία. Εφευρέθηκε από τον Charles E. Leiserson του Ινστιτούτου Τεχνολογίας της Μασαχουσέτης το 1985. Είναι μια περίπλοκη τοπολογία που χρησιμοποιείται σε μεγάλα κέντρα δικτύων.



Εικόνα 6-1 Fattree topology (Πηγή: (Al-Fares, 2008))

Στην παρούσα εργασία δημιουργήθηκε μια προσαρμοσμένη απλή και μια FAT-TREE τοπολογία με πολλούς switches και hosts. Οι λεπτομέρειες για κάθε μία από τις κάθε τύπου τοπολογίας, δηλαδή ο αριθμός των hosts και των switches δίνονται στον παρακάτω πίνακα. Στην τοπολογία δέντρου, έχουμε τις ίδιες ορολογίες όπως ρίζα,

γονέας, παιδί κ.λπ. Χρησιμοποιείται κυρίως για τη σύνδεση μεγάλου αριθμού φυσικών διακομιστών/υπολογιστών σε ένα μεγάλο κέντρο δεδομένων.

Οργανώνουμε ένα k-ary FatTree (Al-Fares et al., 2008) που όπως φαίνεται στην εικόνα 6-1. Υπάρχουν k pods, καθένα από τα οποία περιέχει δύο στρώματα από k/2 switches. Κάθε switch k θυρών στο κατώτερο στρώμα συνδέεται απευθείας με k/2 κεντρικούς υπολογιστές. Κάθε μία από τις υπόλοιπες k/2 θύρες συνδέεται με k/2 από τις k θύρες στη συνάθροιση στρώμα της ιεραρχίας. Υπάρχουν (k/2)2 switches πυρήνα k θυρών. Κάθε μεταγωγέας πυρήνα έχει μια θύρα συνδεδεμένη σε κάθε ένα από τα k pods. Η *i* θύρα οποιουδήποτε κεντρικού switch συνδέεται στο pod *i* έτσι ώστε οι διαδοχικές θύρες στη συνάθροιση επιπέδου κάθε switch pod συνδέονται με switches πυρήνα σε (k/2) βήματα. Γενικά, ένα FatTree κατασκευάζεται με k-port switches υποστηρίζει $k^3/4$ υποδοχείς. Στην παρούσα εργασία, η παράμετρος k κυμαίνεται από 4 έως 10. Ένα πλεονέκτημα της τοπολογίας FatTree είναι ότι όλα τα στοιχεία μεταγωγής είναι πανομοιότυπα, επιτρέποντάς να αξιοποιηθούν φτηνά εξαρτήματα πρώτης ανάγκης για όλους τους switches στην αρχιτεκτονική επικοινωνίας. Επιπλέον, τα FatTrees είναι αναδιατάξιμα μη-μπλοκαρισμένα, που σημαίνει ότι για αυθαίρετα μοτίβα επικοινωνίας, υπάρχει κάποιο σύνολο διαδρομών που θα κορεστεί όλο το εύρος ζώνης που είναι διαθέσιμο στους τελικούς υπολογιστές της τοπολογίας.

Στον πίνακα 4, βλέπουμε τις τοπολογίες που δημιουργήσαμε στην εργασίας μας, όπου έτρεξαν ι ελεγκτές και πήραμε τα αποτελέσματα.

Πίνακας 4 Τοπολογίες πειραμάτων

ΤΟΠΟΛΟΓΙΑ	ΑΡΙΘΜΟΣ HOSTS	ΑΡΙΘΜΟΣ SWITCHES
Single	4	1
Fat-tree	16	20
Fat-tree	54	45
Fat-tree	128	80
Fat-tree	250	125

Στην απλή τοπολογία όπου την δημιουργήσαμε με 4 hosts και 1 switch, θελήσαμε να δούμε λόγω της απλότητας της συνδεσμολογίας μεταξύ των κόμβων της, τι αποδόσεις θα μας έδινε ο εκάστοτε ελεγκτής σε θέματα απόδοσής αλλά και καθυστέρησης. Οι μετρήσεις έγιναν σε όλες τις διαδρομές μεταξύ των κόμβων στην συγκεκριμένη τοπολογία λόγω του ότι ήταν μικρή στο μέγεθος. Έπειτα δημιουργήσαμε και την τοπολογία Fat-tree σε διαφορετικά μεγέθη (π.χ. κ=4, κ=6, κ=8, κ=10), όπου είναι μια περίπλοκη τοπολογία που χρησιμοποιείται σε μεγάλα κέντρα δικτύων, για να δούμε την ανάλογη συμπεριφορά των ελεγκτών. Τα πειράματα στην συγκεκριμένη περίπτωση λόγω του μεγάλου μεγέθους αλλά και της πολυπλοκότητας της έγιναν μεταξύ τυχαίων hosts, αλλά σε κάθε μέγεθος διατηρήσαμε τις ίδιες μετρήσεις. Όλα τα πειράματα επαναλήφθηκαν τρείς φορές για σιγουρευτούμε πώς τα αποτελέσματα ήταν σε ίδια επίπεδα.

6.3 Κριτήρια-Μετρικές

Η αξιολόγηση των ελεγκτών βασίστηκε στις παρακάτω μετρικές:

- Throughput: Απόδοση
- Jitter: Διακύμανση
- Standar deviation: Τυπική απόκλιση
- Average: Μέσος όρος καθυστέρησης

6.4 Σενάρια

Για τη σύγκριση μεταξύ των ελεγκτών υλοποιήθηκαν τα ακόλουθα σενάρια:

i) Απλή τοπολογία: Η τοπολογία περιλάμβανε ένα switch και 4 hosts. Για την τοπολογία αυτή μελετήθηκε η συμπεριφορά των τεσσάρων ελεγκτών ως προς τις ακόλουθες περιπτώσεις και αντίστοιχες μετρικές:

(1) Μελέτη απόδοσης TCP: Ως μετρητική χρησιμοποιήθηκε η διεκπεραιωτική ικανότητα (throughput) μεταξύ των κόμβων ….

(2) Μελέτη απόδοσης UDP: οι μετρικές που χρησιμοποιήθηκαν ήταν οι …

ii) Τοπολογία FatTree: Δημιουργήθηκαν και μελετήθηκαν τοπολογίες FatTree για k=[4,6,8,10]. Χρησιμοποιήθηκαν οι ίδιες περιπτώσεις και μετρικές όπως και στο προηγούμενο σενάριο και τις ακόλουθες παραδοχές:

(1) Μετρήσεις για hosts που βρίσκονται στο ίδιο pod

(2) Μετρήσεις για hosts που βρίσκονται σε γειτονικά pods

(3) Μετρήσεις για hosts που βρίσκονται σε απομακρυσμένα pods

6.5 Περιβάλλον Εκτέλεσης πειραματικών μετρήσεων/προσομοιώσεων

Για την δοκιμή των ελεγκτών έγινε η εγκατάστασή τους σε επιτραπέζιο υπολογιστή με τα ακόλουθα χαρακτηριστικά: Επεξεργαστής AMD Ryzen 5 3600 6-Core Processor στα 3.60 GHz, 2 RAM των 8GB καθεμία, λειτουργικό σύστημα 64 bit, για μητρική χρησιμοποιήθηκε η MSI B450 Tomahawk Max Motherboard ATX με AMD AM4 Socket και κάρτα γραφικών η Asus GeForce GT 1030 στα 2GB GDDR5 OC. Κλείνοντας με τα χαρακτηριστικά ο σκληρός δίσκος ήταν ο Samsung 970 Evo SSD 500GB M.2 NVMe.

Οι controllers εγκαταστάθηκαν στην εικονική μηχανή Ubuntu 20.04.4 (ubuntu.com, n.d.). Για την εικονική μηχανή χρησιμοποιήθηκε το λογισμικό virtual Box.

Εκτός από τους controllers στην εικονική μηχανή έγινε και εγκατάσταση του ενός εργαλείου εξομοίωσης, του Mininet, προκειμένου να αξιολογηθεί η απόδοση των διαφόρων ελεγκτών. Το Mininet είναι ένα εργαλείο που δημιουργεί το εικονικές δικτυακές συσκευές (switch, hosts) και οι οποίες αποτελούν το επίπεδο δεδομένων και τον τρόπο που αυτές συνδέονται μεταξύ τους. Τα switches επικοινωνούν με τον εκάστοτε controller. Ο controller καθορίζει με ποιο τρόπο θα γίνει η προώθηση των πακέτων στα switches (δηλαδή ποια θύρα εξόδου θα επιλεγεί κάθε φορά από το κάθε switch. Είναι πολύ χρήσιμο στην εκτέλεση των διαφορετικών τοπολογιών δικτύου μέσα σε λίγα δευτερόλεπτα. Κάθε φορά για μια νέα τοπολογία δικτύου και για διαφορετικούς ελεγκτές SDN έχουμε εκτελέσει τη διαδικασία εκκαθάρισης για το πείραμα. Αυτό βοηθάει στην αποφυγή τυχόν προηγούμενων αρχείων καταγραφής και δεδομένων της προσωρινής μνήμης.

6.5.1 Το Mininet

Το Mininet (Karamjeet Kaur, 2015) είναι ένας εξομοιωτής για την ανάπτυξη μεγάλων δικτύων με τους περιορισμένους πόρους ενός απλού υπολογιστή ή εικονικής μηχανής. Το Mininet δημιουργήθηκε για την έρευνα στον τομέα της δικτύωσης που καθορίζεται από λογισμικό (SDN) και το OpenFlow. Ο εξομοιωτής Mininet επιτρέπει την εκτέλεση μη τροποποιημένου κώδικα δια δραστικά σε εικονικό υλικό σε ένα απλό υπολογιστή. Παρέχει ευκολία και ρεαλισμό σε πολύ χαμηλό κόστος. Η άλλη επιλογή είναι η χρήση προσομοιωτή που είναι πολύ φθηνός αλλά μερικές φορές αργός και απαιτεί τροποποίηση κώδικα. Το Mininet προσφέρει ευκολία χρήσης, ακρίβεια επιδόσεων και επεκτασιμότητα.

Τα τελευταία χρόνια, υπάρχει μεγάλη ανάγκη μοντελοποίησης των κεντρικών υπολογιστών, των switches, των συνδέσμων και των ελεγκτών SDN/Openflow.Το Mininet επιτρέπει την δημιουργία τοπολογιών πολύς μεγάλης κλίμακας μεγέθους έως και χιλιάδων κόμβων και εκτελεί δοκιμές σε αυτές πολύ εύκολα. Έχει απλά εργαλεία γραμμής εντολών API και επιτρέπει στον χρήστη να δημιουργεί, να προσαρμόζει, να μοιράζεται και να δοκιμάζει εύκολα SDN δίκτυα. Είναι ελεύθερο λογισμικό ανοικτού κώδικα που εξομοιώνει συσκευές Openflow ελεγκτές SDN. Για την εφαρμογή προηγμένων εννοιών ο POX (Al-Shabib, 2015) χρησιμοποιείται ως ελεγκτής.

Ορισμένα χαρακτηριστικά de Oliveira et al. (Rogério Leão Santos de Oliveira, 2014) που οδήγησαν στη δημιουργία του Mininet είναι τα εξής:

- Ευελιξία, δηλαδή μπορούν να οριστούν νέες τοπολογίες και νέα χαρακτηριστικά στο λογισμικό, χρησιμοποιώντας γλώσσες προγραμματισμού και κοινά λειτουργικά συστήματα.

- Εφαρμοσιμότητα, σωστές υλοποιήσεις που έγιναν σε πρωτότυπα θα πρέπει να είναι επίσης χρησιμοποιήσιμες σε πραγματικά δίκτυα που βασίζονται σε υλικό χωρίς αλλαγές στους πηγαίους κώδικες
- Δια δραστικότητα, διαχείριση και λειτουργία του προσομοιωμένου δικτύου πρέπει να γίνεται σε πραγματικό χρόνο, σαν να συμβαίνει σε πραγματικά δίκτυα.
- Επεκτασιμότητα, το περιβάλλον προτυποποίησης πρέπει να μπορεί να κλιμακώνεται σε μεγάλα δίκτυα με εκατοντάδες ή χιλιάδες switches.
- Ρεαλιστικότητα, η συμπεριφορά του πρωτοτύπου πρέπει να αντιπροσωπεύει τη συμπεριφορά σε πραγματικό χρόνο με υψηλό βαθμό αξιοπιστίας, οπότε οι εφαρμογές και οι στοίβες πρωτοκόλλων θα πρέπει να μπορούν να χρησιμοποιηθούν χωρίς οποιαδήποτε τροποποίηση κώδικα.
- Κοινόχρηστο, το δημιουργημένα πρωτότυπα θα πρέπει να μοιράζονται εύκολα με άλλους συνεργάτες, οι οποίοι μπορούν στη συνέχεια να εκτελέσουν και να τροποποιήσουν τα πειράματα.

Το Mininet μπορεί να δημιουργήσει στοιχεία SDN, να τα προσαρμόσει, να τα μοιραστεί με άλλα δίκτυα και να εκτελεί αλληλεπιδράσεις. Αυτά τα στοιχεία περιλαμβάνουν Hosts, Switches, Controllers και Links. Ένας κεντρικός υπολογιστής στο Mininet είναι μια απλή διαδικασία με το δικό του δίκτυο περιβάλλον που εκτελείται στο λειτουργικό σύστημα. Κάθε ένας παρέχει στις διεργασίες με αποκλειστική ιδιοκτησία εικονικό δίκτυο διεπαφής, θύρες, διευθύνσεις και πίνακες δρομολόγησης (όπως ARP και IP). Οι switches OpenFlow που δημιουργήθηκαν από το Mininet παρέχουν την ίδια σημασιολογία παράδοσης πακέτων που θα παρείχε ένας μεταγωγέα υλικού. Τόσο οι switches χώρου χρήστη όσο και οι switches χώρου πυρήνα είναι διαθέσιμοι. Στην προσομοίωση του Mininet, οι ελεγκτές μπορούν να εκτελούνται σε πραγματικό ή σε προσομοιωμένο δίκτυο, εφόσον το μηχάνημα που εκτελούνται οι switches έχει συνδεσιμότητα με το ελεγκτή. Εάν είναι επιθυμητό, το Mininet δημιουργεί έναν τυπικό ελεγκτή μέσα στο τοπικό περιβάλλον προσομοίωσης.

6.5.1.1 Πλεονεκτήματα του Mininet

Κάποια από τα πλεονεκτήματα του Mininet είναι τα παρακάτω.

- Είναι γρήγορο. Η εκκίνηση ενός απλού δικτύου διαρκεί μόλις λίγα δευτερόλεπτα, άρα ο βρόχος run-edit-debug μπορεί να είναι πολύ γρήγορος.

- Δυνατότητα δημιουργίας προσαρμοσμένης τοπολογίας.
- Δυνατότητα εκτέλεσης πραγματικών προγραμμάτων. Οτιδήποτε εκτελείται σε Linux είναι διαθέσιμο για εκτέλεση, όπως το Wireshark αλλά και εργαλεία παρακολούθησης παραθύρων TCP.

- Δυνατότητα στον χρήστη να προσαρμόζει την προώθηση των πακέτων. Χρησιμοποιώντας το OpenFlow οι διακόπτες του Mininet μπορούν να προγραμματιστούν.
- Δυνατότητα εκτέλεσης του Mininet σε φορητό υπολογιστή, σε διακομιστή, σε VM, σε εγγενές κιβώτιο Linux (Ubuntu) ή σε Cloud.

- Επιτρέπει τον μοιρασμό (Share) και την αναπαραγωγή των αποτελεσμάτων. Οποιοσδήποτε διαθέτει υπολογιστή, μπορεί να εκτελέσει κώδικα άλλου χρήστη.

- Εύκολο στην χρήση του. Δυνατότητα σύνταξης και εκτέλεσης προγραμμάτων, γράφοντάς σε γλώσσα προγραμματισμού Python.

- Είναι έργο ανοιχτού κώδικα. Άρα οποιασδήποτε μπορεί να το τροποποίησή, να διορθώσει σφάλματα και να υποβάλλει διάφορα αιτήματα.

- Το Mininet ως εργαλείο βρίσκεται ακόμα υπό ανάπτυξη,επομένως αν κάτι δεν λειτουργεί σωστά, οποιασδήποτε χρήστης μπορεί να ενημερώσει την κοινότητα χρηστών και προγραμματιστών Mininet στο **Mininet-discuss**.

6.5.1.2 ΜΕΙΟΝΕΚΤΗΜΑΤΑ ΤΟΥ MININET

Αν και το Mininet είναι το ποιο διαδεδομένο εργαλείο εξομοίωσης δικτύων SDN, έχει και κάποια μειονεκτήματα όπως για παράδειγμα:

- Η εκτέλεση σε ένα μόνο σύστημα είναι βολική, αλλά επιβάλλει όρια πόρων. Εάν για παράδειγμα ο διακομιστής ενός χρήστη έχει CPU 3 GHz και μπορεί να αλλάξει περίπου 10 Gbps προσομοιωμένης κίνησης, αυτοί οι πόροι θα πρέπει να εξισορροπηθούν και να μοιραστούν μεταξύ των εικονικών κεντρικών υπολογιστών και μεταγωγέων.

- Το Mininet χρησιμοποιεί έναν ενιαίο πυρήνα Linux για όλους τους εικονικούς κεντρικούς υπολογιστές. Αυτό σημαίνει ότι ο χρήστης δεν μπορεί να εκτελέσει λογισμικό που εξαρτάται από BSD, Windows ή άλλους πυρήνες λειτουργικού συστήματος. (Αν και μπορείτε να συνδέσετε VM στο Mininet.)

- Το Mininet δεν θα γράψει τον ελεγκτή OpenFlow για εσάς. Εάν χρειάζεστε προσαρμοσμένη συμπεριφορά δρομολόγησης ή εναλλαγής, θα χρειαστεί να βρείτε ή να αναπτύξετε έναν ελεγκτή με τις δυνατότητες που χρειάζεστε.

- Από προεπιλογή το δίκτυό Mininet είναι απομονωμένο από το LAN και από το Διαδίκτυο. Θα πρέπει ο εκάστοτε χρήστης να συνδέσει το Mininet στο LAN μέσω της διεύθυνσης δικτύου.

- Σε αντίθεση με έναν προσομοιωτή, το Mininet δεν έχει ισχυρή έννοια του εικονικού χρόνου. Αυτό σημαίνει ότι οι μετρήσεις θα βασίζονται σε πραγματικό χρόνο και ότι τα αποτελέσματα θα είναι ταχύτερα από τον πραγματικό χρόνο, (π.χ. δίκτυα 100 Gbps) δεν μπορούν εύκολα να εξομοιωθούν.

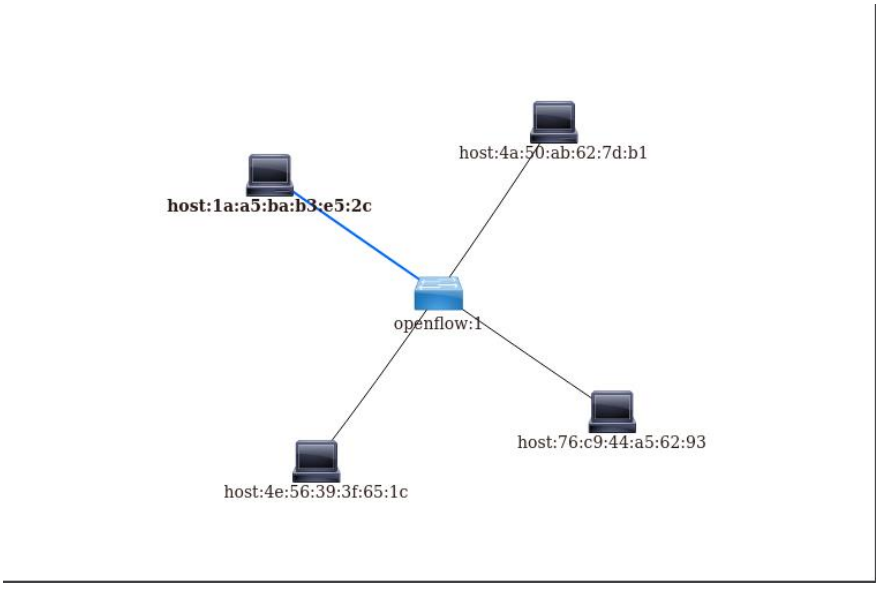
Περισσότερες πληροφορίες για τη λειτουργία του Mininet παρουσιάζονται στο ΠΑΡΑΡΤΗΜΑ Α.

7 Πειραματικά αποτελέσματα

7.1 Απλή τοπολογία

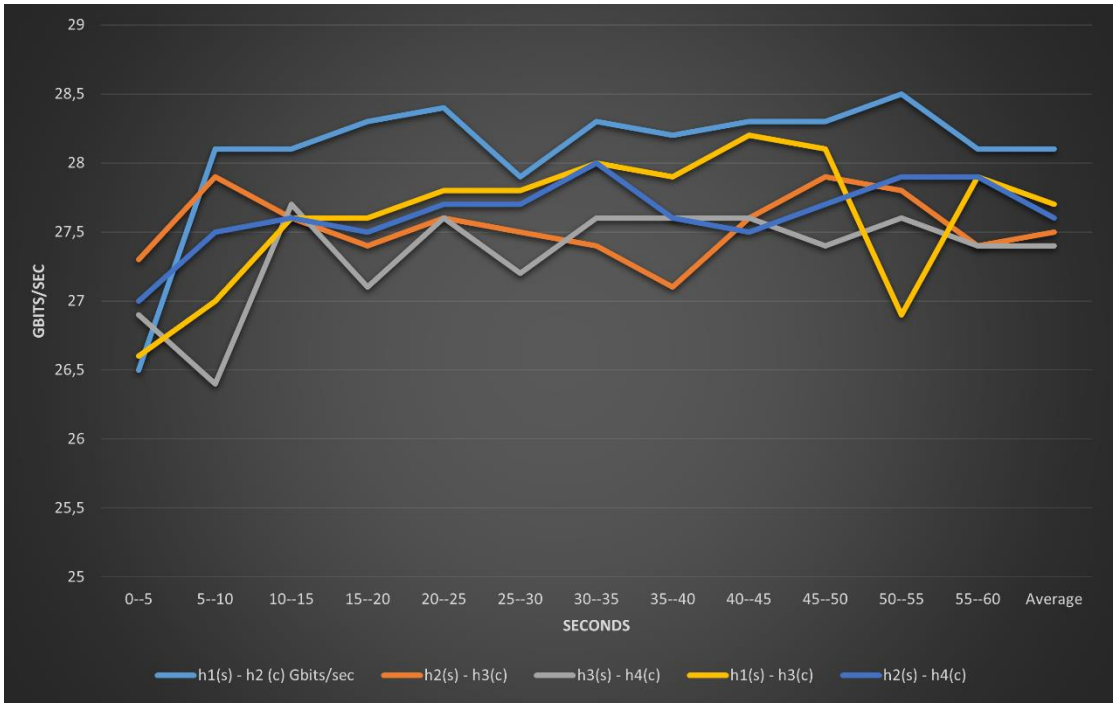
7.1.1 Μελέτη συμπεριφοράς ελεγκτών με χρήση του πρωτοκόλλου TCP

Σε αυτήν την ενότητα θα δούμε τα αποτελέσματα των ελεγκτών όσο αφορά την απόδοση και την καθυστέρηση που μετρήθηκαν στην απλή εικονική τοπολογία (Εικ.7-1) που δημιουργήσαμε μέσω του Mininet. Χρησιμοποιώντας το εργαλείο iperf για αποστολή και λήψη δεδομένων μεταξύ των hosts και πραγματοποιούμε μετρήσεις, για χρονικό διάστημα ενός λεπτού, καταγράφοντας τα αποτελέσματα ανά 5 δευτερόλεπτα. Για την επικοινωνία θεωρούμε ότι ο ένας host είναι ο server και ο άλλος ο client.



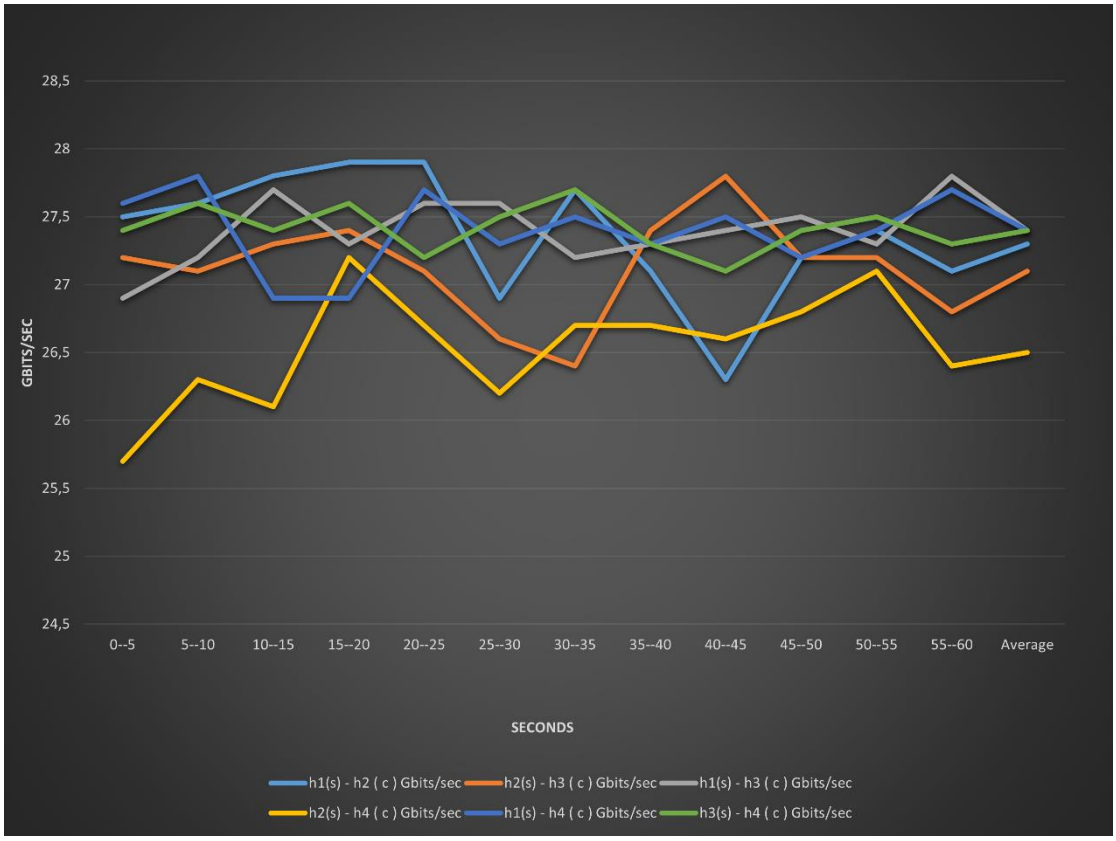
Εικόνα 7-1 SINGLE TOPOLOGY

Ξεκινώντας με τον Ryu, βλέπουμε στην εικόνα 7-2 την απόδοση του μεταξύ όλων των διαδρομών των Hosts που περιέχει η τοπολογία. Παρατηρούμε πώς απόδοση του έχει ελάχιστη τιμή περίπου τα 26,5 Gbits/sec μεταξύ των Hosts h3 και h4, και φτάνει τα 28,5 Gbits/sec στην διαδρομή h1-h2. Παρόλο που είναι από τους πιο παλιούς ελεγκτές έχει αρκετά ικανοποιητικές τιμές απόδοσης.



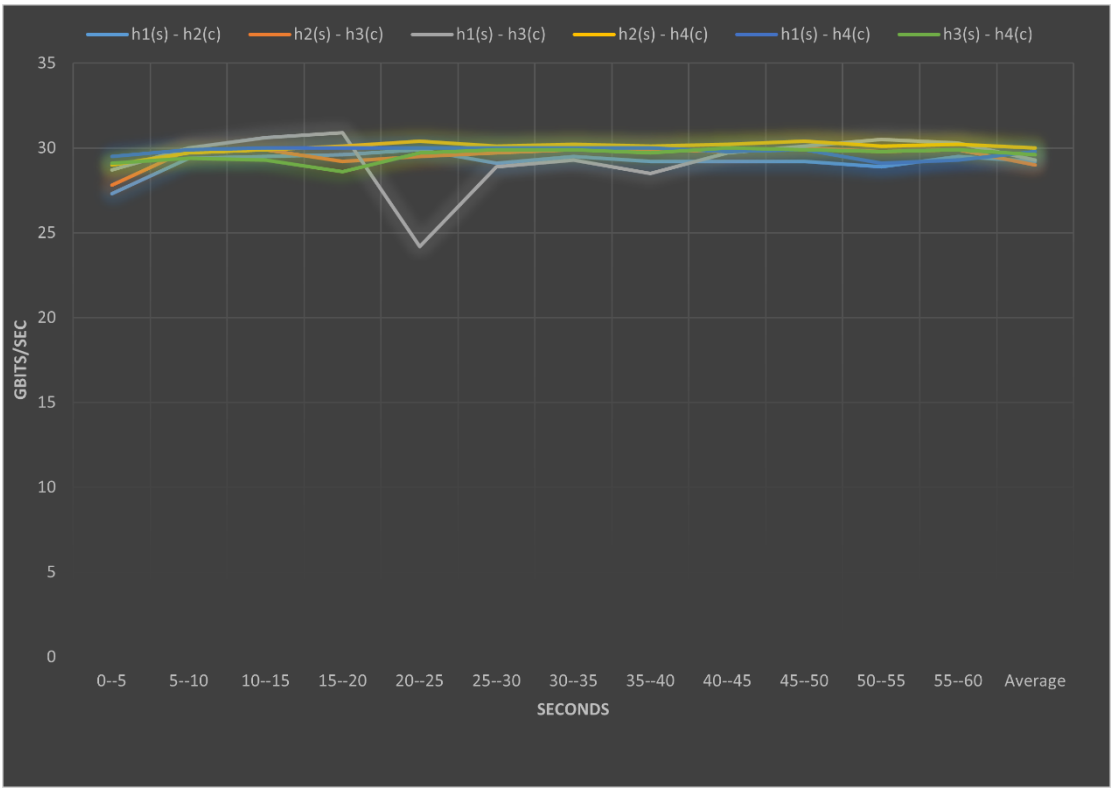
Εικόνα 7-2 RYU THROUGHPUT AT SINGLE TOPOLOGY

Στη συνέχεια (Εικ.7-3) βλέπουμε την συμπεριφορά του Floodlight, όπου η διακύμανση της απόδοσης του ξεκινάει από τα 25,7 Gbits/sec μεταξύ των hosts h2-h4, και φτάνει ως τα 27,7 Gbits/sec. Θα περιμέναμε καλύτερες τιμές στις αποδόσεις του Floodlight καθώς είναι πιο αναβαθμισμένος και ευρέως χρησιμοποιείται πολύ στην επιστημονική κοινότητα, σε σχέση με τον Ryu. Επίσης παρατηρούμε πως ο και οι δύο ελεγκτές ως τώρα έχουν μια αυξομείωση στην απόδοση τους περίπου στα 2 Gbits/sec, όπου φαίνεται να υπάρχει μια μικρή αστάθεια, παρόλο που τρέχουν σε μια απλή και μικρή τοπολογία.



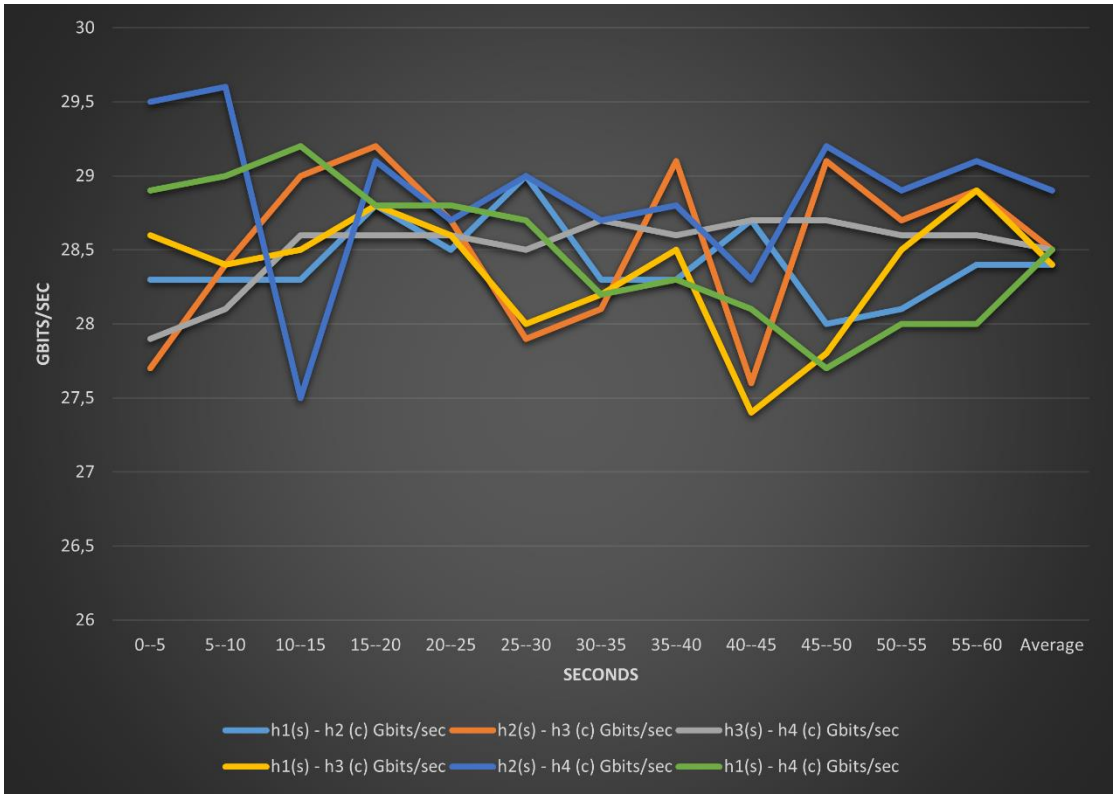
Εικόνα 7-3 FLOODLIGHT THROUGHPUT AT SINGLE TOPOLOGY

Ο ONOS βλέπουμε πως η απόδοση του είναι πολύ υψηλή καθώς ξεπερνάει τα 30 Gbits/sec. Επίσης έχει και μια καταπληκτική σταθερότητα μεταξύ όλων των hosts της τοπολογίας όπως βλέπουμε (Εικ.7-4). Είναι ο μοναδικός ελεγκτής από αυτούς που μελετήσαμε όπου έχει τόσο υψηλές και σταθερές τιμές στην απόδοση του και δικαίως θεωρείται ένας από τους πιο ισχυρούς και αναπτυσσόμενους ελεγκτές.



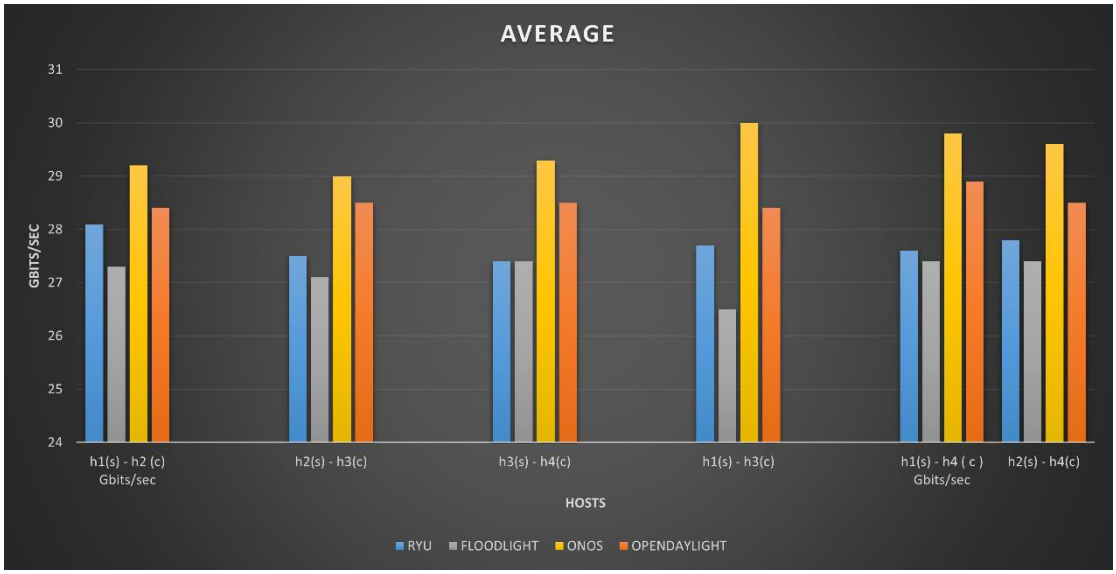
Εικόνα 7-4 ONOS THROUGHPUT AT SINGLE TOPOLOGY

Ο OpenDayLight, έχει την μεγαλύτερη διακύμανση στην απόδοση του (Εικ.7-5), καθώς η χαμηλότερη τιμή του είναι στα 27,3 Gbits/sec στους hosts h1-h3, και η υψηλότερη τιμή είναι 29,6 Gbits/sec μεταξύ των hosts h1-h2. Παρόλη την μικρή αστάθεια που παρατηρούμε και στον συγκεκριμένο ελεγκτή, είναι αξιοσημείωτες οι πολύ υψηλές τιμές στην απόδοση του καθώς αγγίζει σχεδόν τα επίπεδα του ONOS.



Εικόνα 7-5 OPENDAYLIGHT THROUGHPUT AT SINGLE TOPOLOGY

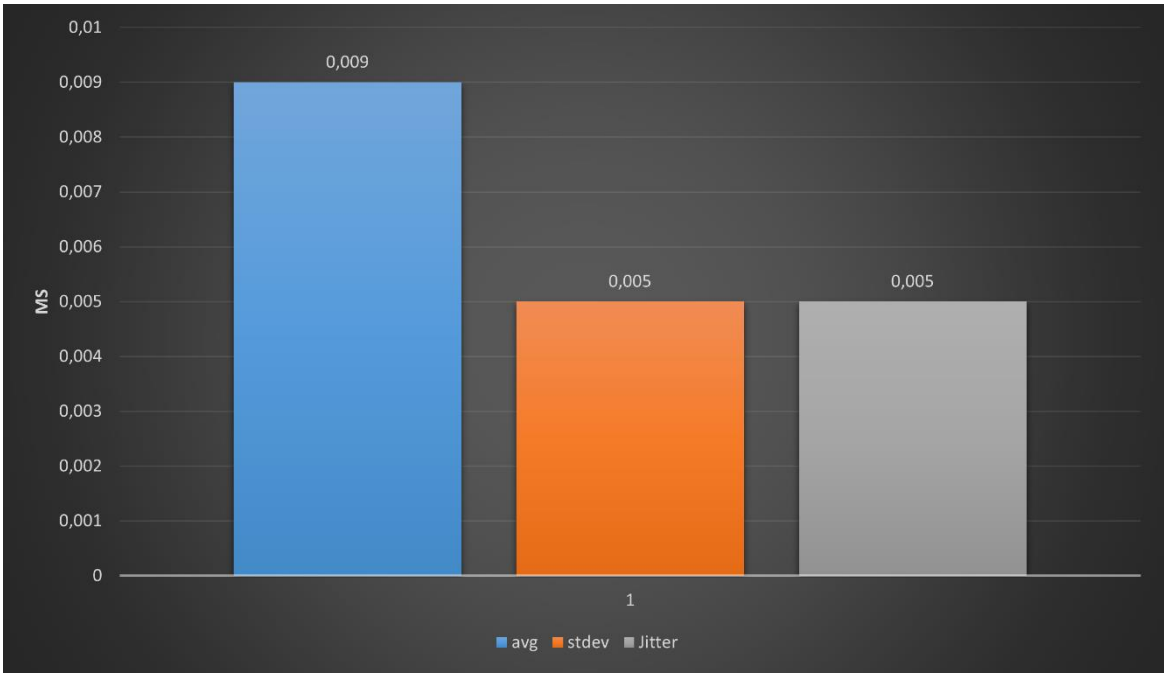
Στην εικόνα 7-6, βλέπουμε τους μέσους όρους απόδοσης του εκάστοτε ελεγκτή, σε χρονική διάρκεια 60 δευτερολέπτων, για όλες τις δυνατές διαδρομές της απλής τοπολογίας. Ξεχωρίζει όπως παρατηρούμε από όλους τους άλλους ο ONOS με σημαντική διαφορά στην απόδοση, και ακολουθεί με επίσης πολύ υψηλά ποσοστά απόδοσης ο OpenDayLight. Επίσης αξίζει να αναφέρουμε την απόδοση του Floodlight καθώς είναι η χαμηλότερη από όλους τους υπόλοιπους ακόμα και από τον Ryu, όπου σύμφωνα με έρευνες που έχουν δημοσιευθεί κατά καιρούς συνήθως εμφανίζεται ο πιο αδύναμος.



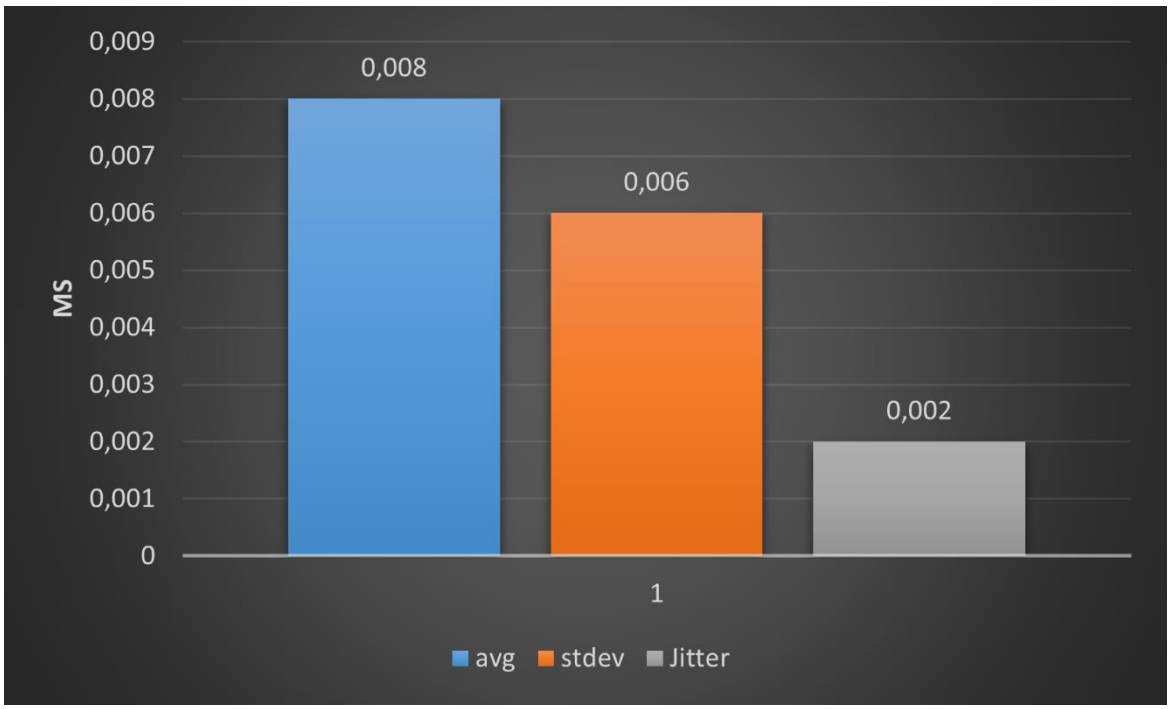
Εικόνα 7-6 AVERAGE OF CONTROLLERS

7.1.2 Μελέτη συμπεριφοράς ελεγκτών με χρήση του πρωτοκόλλου UDP.

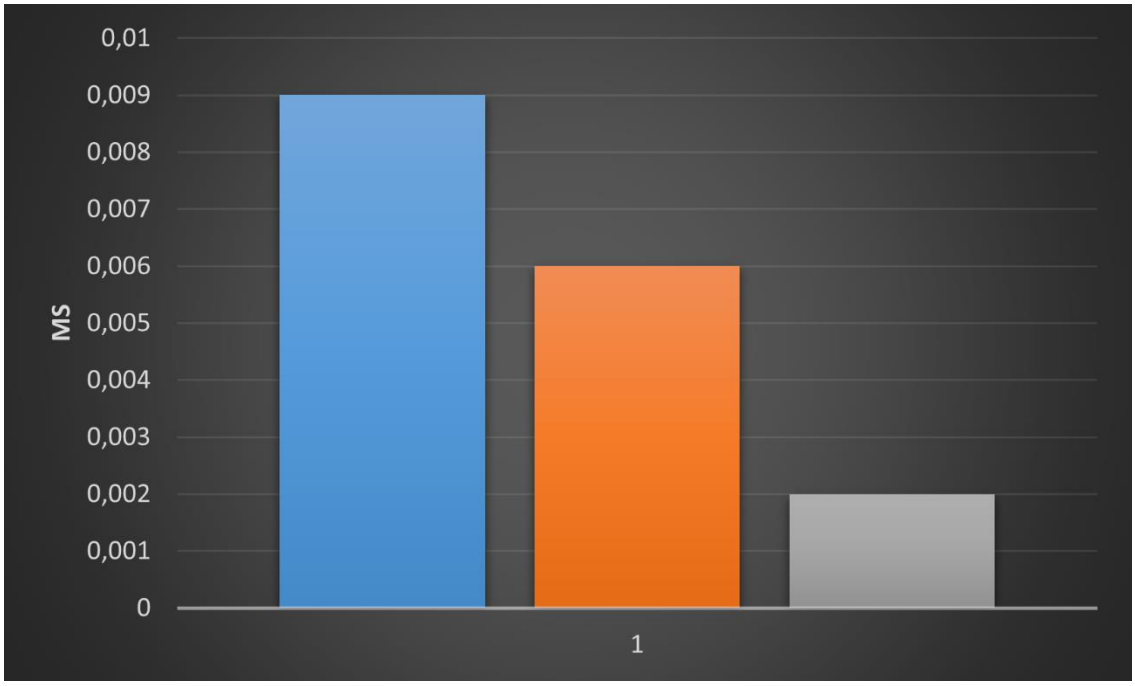
Με τη χρήση της εντολής ‘iperf’ για το πρωτόκολλο UDP, πραγματοποιήσαμε μετρήσεις για την τυπική απόκλιση (standard deviation) και την τιμή της διακύμανσης του χρόνου καθυστέρησης/απόκρισης (Jitter) κατά τη μεταφορά πακέτων μεταξύ δύο hosts (average),. Το Jitter επίσης μπορεί να προσεγγιστεί ως η διαφορά μεταξύ της μέγιστης και της ελάχιστης καθυστέρησης από πακέτο σε πακέτο για μια δεδομένη χρονική περίοδο. Στις εικόνες 7-7, 7-8, 7-9 και 7-10 απεικονίζονται οι μέσες τιμές του εκάστοτε ελεγκτή στα προαναφερόμενα μετρικά.



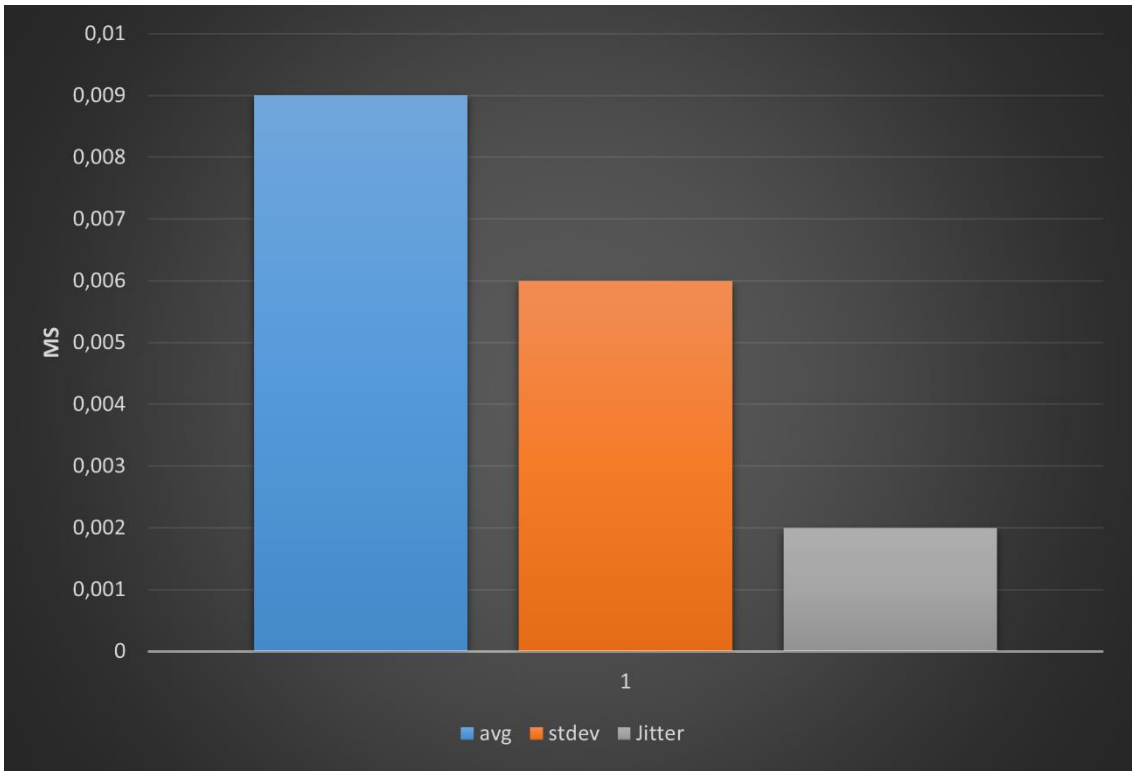
Εικόνα 7-7 Average-Standard deviation-Jitter of Ryu



Εικόνα 7-8 Average-Standard deviation-Jitter of Floodlight



Εικόνα 7-9 Average-Standard deviation-Jitter of Onos



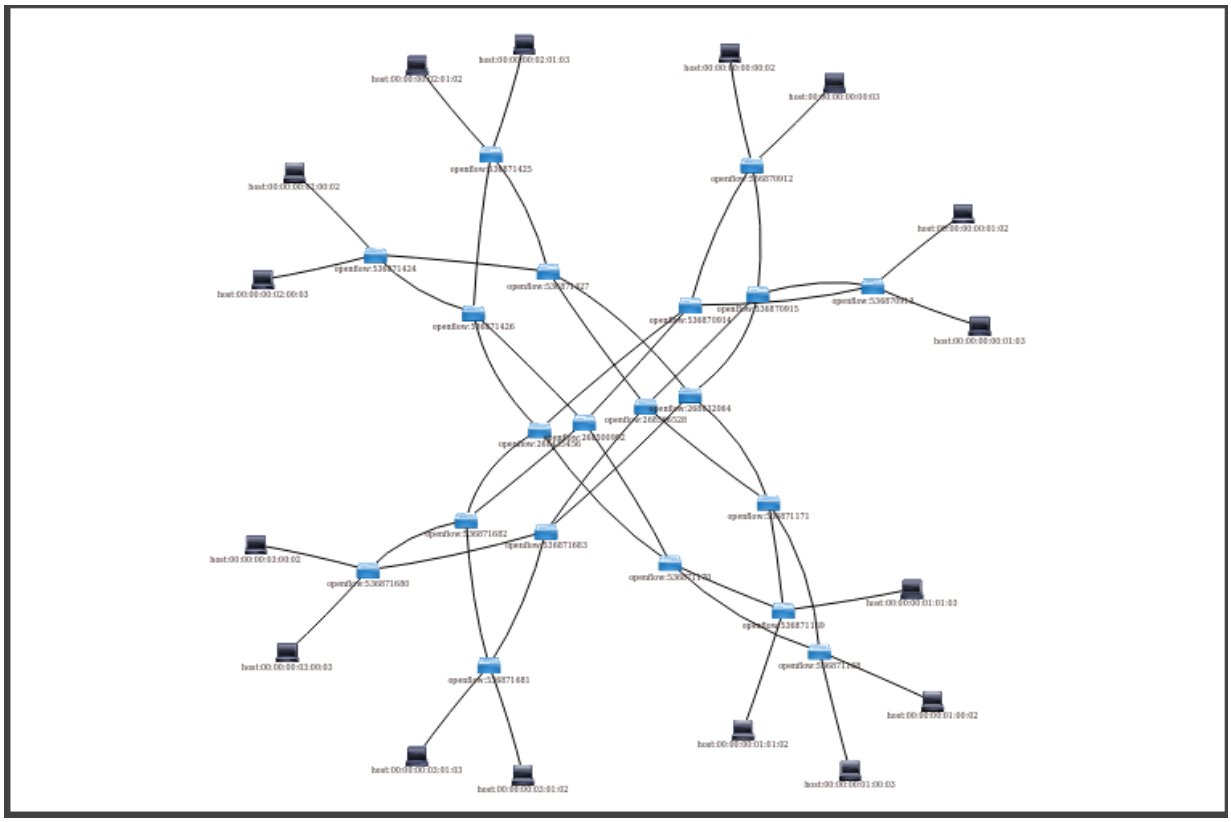
Εικόνα 7-10 Average-Standard deviation-Jitter of OpenDayLight

Είναι φανερό πως το jitter έχει μέσο όρο 0,2 ms σε όλους τους ελεγκτές, αλλά στην περίπτωση του Ryu βλέπουμε ο μέσος όρος φτάνει στα 0,5 ms, που σημαίνει ότι υπάρχουν μεγαλύτερες διακυμάνσεις στην καθυστέρηση σε σχέση με τους άλλους. Όσο αφορά την τυπική απόκλιση, ο μέσος όρος όλων το ελεγκτών είναι 0,006 ms, και 0,005 ms στον ONOS. Ο μέσος όρος της καθυστέρησης μπορούμε να θεωρήσουμε ότι είναι στα ίδια επίπεδα όπως η τυπική απόκλιση. Παρατηρούμε πως και οι 4 ελεγκτές σε θέματα καθυστέρησης, έχουν τους ίδιους μέσους όρους όσο αφορά μια απλή τοπολογία.

7.2 Τοπολογία FatTree

7.2.1 Μελέτη απόδοσης TCP

Η τοπολογία δικτύου FatTree μοιάζει με τοπολογία δέντρου όπως στη παρακάτω εικόνα.

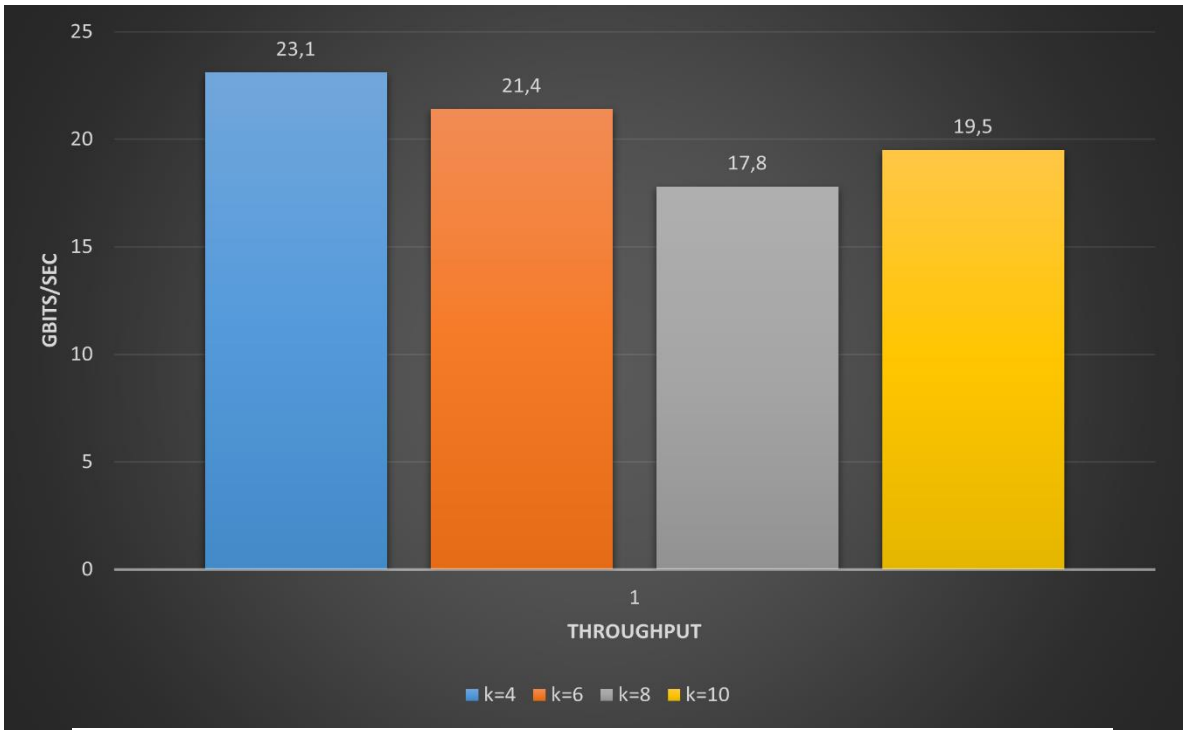


Εικόνα 7-11 Fattree Topology k=4

Οι μετρήσεις έγιναν με την ίδια μεθοδολογία όπως στην απλή τοπολογία. Ένας host, δηλώθηκε ως server και ένας client. Έγιναν μετρήσεις μεταξύ οκτώ τυχαία επιλεγμένων ζευγών από hosts για διάρκεια 60 δευτερολέπτων. Στη διάρκεια των 60 δευτερολέπτων, οι μετρήσεις λαμβάνονται κάθε 5 δευτερόλεπτα και από αυτές υπολογίζουμε τη μέση τιμή. Οι μετρήσεις που λαμβάνουμε είναι για την απόδοση, την καθυστέρηση , τη διακύμανση και την τυπική απόκλιση.

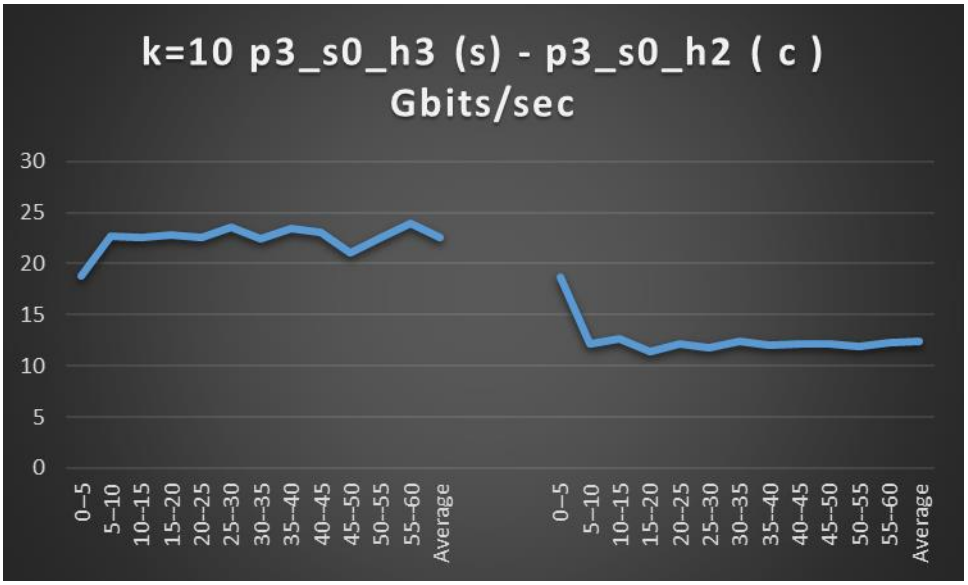
Οι μετρήσεις γίνονται μεταξύ hosts που βρίσκονται στο ίδιο pod, σε γειτονικά ή σε απομακρυσμένα, όπως αναφέρθηκε και στην Ενότητα 6.2.2.Όσο αυξάνεται το μέγεθος της τοπολογίας (παράμετρος κ), τόσο αυξάναμε αντίστοιχα και το πλήθος των τυχαίων μετρήσεων μεταξύ των hosts.

Για τον ελεγκτή Ryu βλέπουμε στην εικόνα 7-12 τα αποτελέσματα των μετρήσεων σχετικά με την απόδοση του.



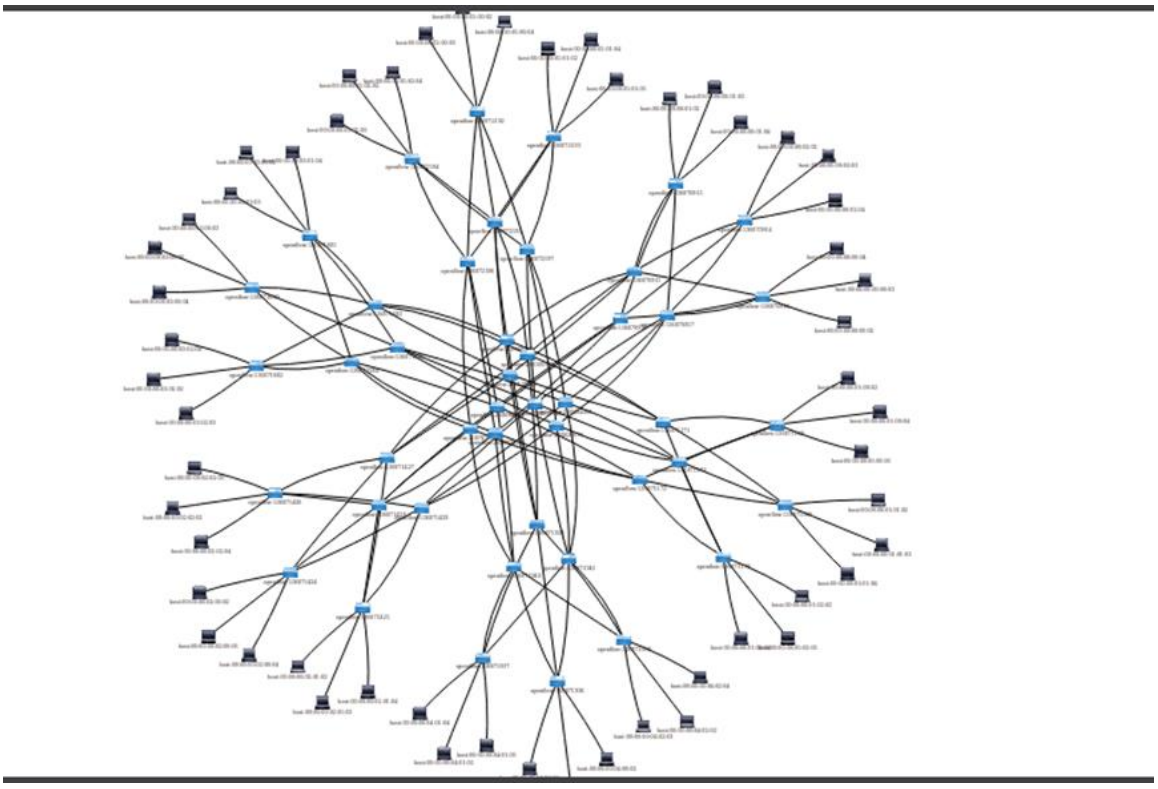
Εικόνα 7-12 Throughput of Ryu at Fattree topology

Παρατηρούμε ότι όσο μικρότερη είναι η τοπολογία FatTree τόσο καλύτερες είναι οι μέσες τιμές απόδοσης (Εικ.7-12). Συγκεκριμένα, για κ=4 η απόδοση φθάνει κατά μέσο όρο την τιμή 23,1 Gbits/sec. Επίσης για κ=6, η μέση τιμή της απόδοσης παραμένει σε υψηλά επίπεδα καθώς είναι 21,4 Gbits/sec. Αξιοσημείωτο είναι το γεγονός, πως για κ=10 που είναι μια τοπολογία με 250 hosts και 125 switches η μέση τιμή της απόδοσης από τις τυχαίες μετρήσεις είναι 19,5 Gbits/sec. Στην αμέσως επόμενη μικρότερη με κ=8, προκαλεί εντύπωση το αποτέλεσμα 17,8 Gbits/sec,καθώς περιμέναμε αρκετά μεγαλύτερη απόδοση καθώς πρόκειται για μια μικρότερη τοπολογία. Το συμπέρασμα είναι πως δεν υπάρχει σταθερή απόδοση από τον Ryu σε καμιά μορφή της τοπολογίας Fattree, αλλά όταν είναι με λίγους hosts και switches, η μέση τιμή της απόδοσης είναι σχετικά υψηλή. Αξίζει σε αυτό το σημείο να δούμε το παρακάτω γράφημα, για να καταλάβουμε καλύτερα πως συμπεριφέρεται σε θέματα απόδοσης ο Ryu σε μεγάλες τοπολογίες ανάμεσα σε διαδρομές γειτονικών hosts και απομακρυσμένων.



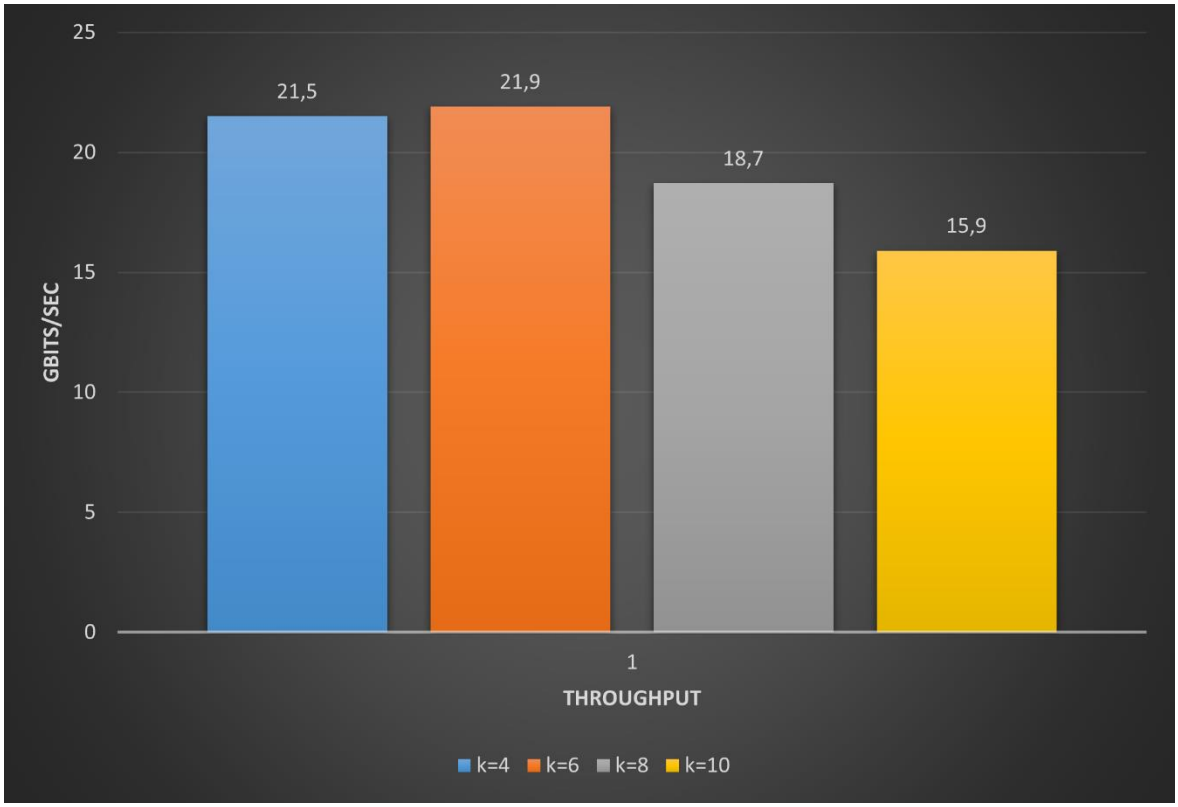
Εικόνα 7-13 Ryu Throughput

Βλέπουμε στην εικόνα 7-13, τις δοκιμές για γειτονικούς κόμβους (π.χ. μεταξύ των hosts p3_s0_h3 – p3_s0_h2 που βρίσκονται στο ίδιο pod), όπου η απόδοση του Ryu φτάνει ακόμα και τα 24 Gbits/sec (αριστερή γραμμή του γραφήματος της εικόνας 7-13), ενώ σε απομακρυσμένους hosts απόδοση του πέφτει κατακόρυφα καθώς έχει για μέγιστη μέση τιμή απόδοσης περίπου τα 18 Gbits/sec και μέση τιμή περίπου 12 Gbits/sec. Συμπεραίνουμε πως όσο μεγάλη και αν είναι η τοπολογία μεταξύ δύο κοντινών hosts ο Ryu εξακολουθεί να έχει υψηλά ποσοστά απόδοσης σε αντίθεση με την απόδοση μεταξύ απομακρυσμένων κόμβων.

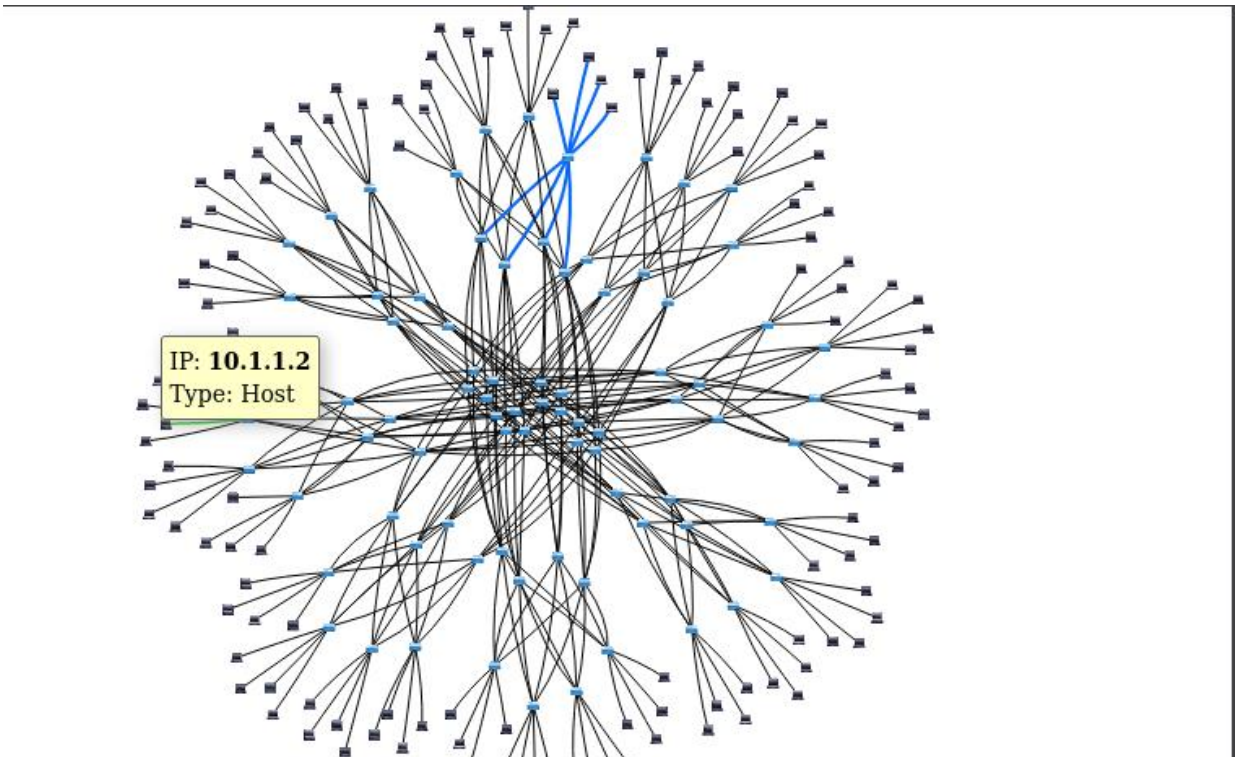


Εικόνα 7-14 Fattree Topology k=6

Ο Floodlight, όπως βλέπουμε παρακάτω (Εικ.7-15), όσο μικρότερη είναι η τοπολογία Fattree, τόσο καλύτερη η μέση τιμή απόδοσης. Για κ=4 και κ=6 παρατηρούμε ικανοποιητική απόδοση όπως στον Ryu, με μέγιστη μέση τιμή 21,9 Gbits/sec. Αλλά, όσο μεγαλώνει η τοπολογία, η απόδοση μειώνεται αρκετά καθώς παρατηρούμε ότι φτάνει στα 15,9 Gbits/sec. Όπως και στην απλή τοπολογία, έτσι και εδώ περιμέναμε από τον Floodlight καλύτερα επίπεδα στην απόδοση. Οι μέσες τιμές είναι γενικά μέτριες και υστερεί σε σχέση με τον Ryu σε αρκετές μορφές της τοπολογίας. Επίσης και εδώ δεν παρατηρούμε κάποια σταθερότητα στην απόδοση, ανεξάρτητα από το μέγεθος της τοπολογίας.

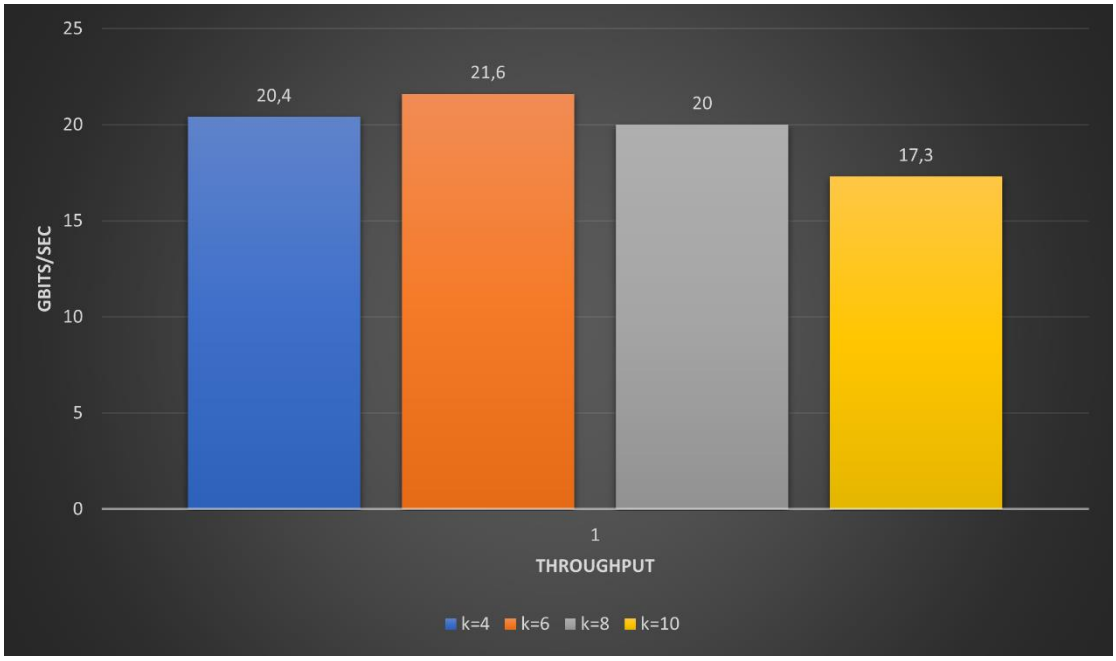


Εικόνα 7-15 Throughput of Floodlight at Fattree topology

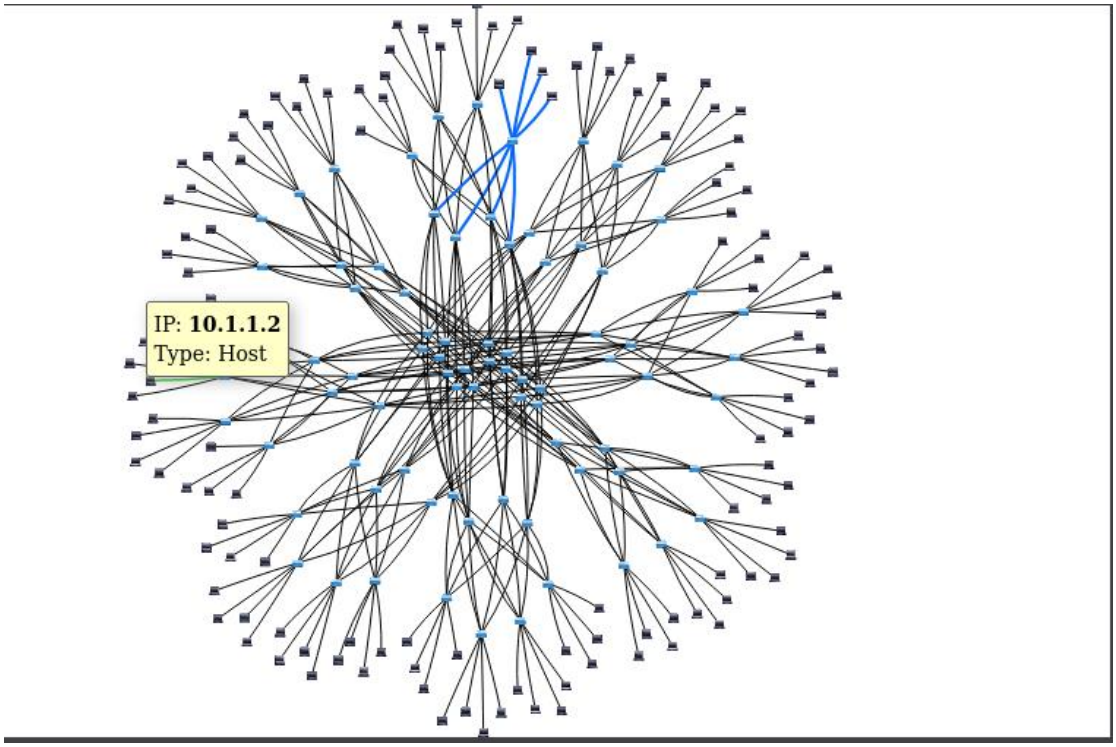


Εικόνα 7-16 Fattree Topology k=8

Συνεχίζοντας με την απόδοση του Onos (Εικ.7-17), βλέπουμε ότι και σε πιο περίπλοκες τοπολογίες, όπως είναι η Fattree, εξακολουθεί να έχει μια σταθερότητα στην απόδοση του για τα διαφορετικά μεγέθη της τοπολογίας. Αξιοσημείωτο είναι επίσης και οι υψηλές αποδόσεις που έχει σε σχέση με όλους τους υπόλοιπους ελεγκτές που μελετήσαμε. Υπάρχει μια διακύμανση από τα 20 Gbits/sec έως τα 21,6 Gbits/sec αν εξαιρέσουμε την μεγαλύτερη τοπολογία (κ=10), όπου η μέση τιμή της απόδοσης πέφτει στα 17,3 Gbits/sec.

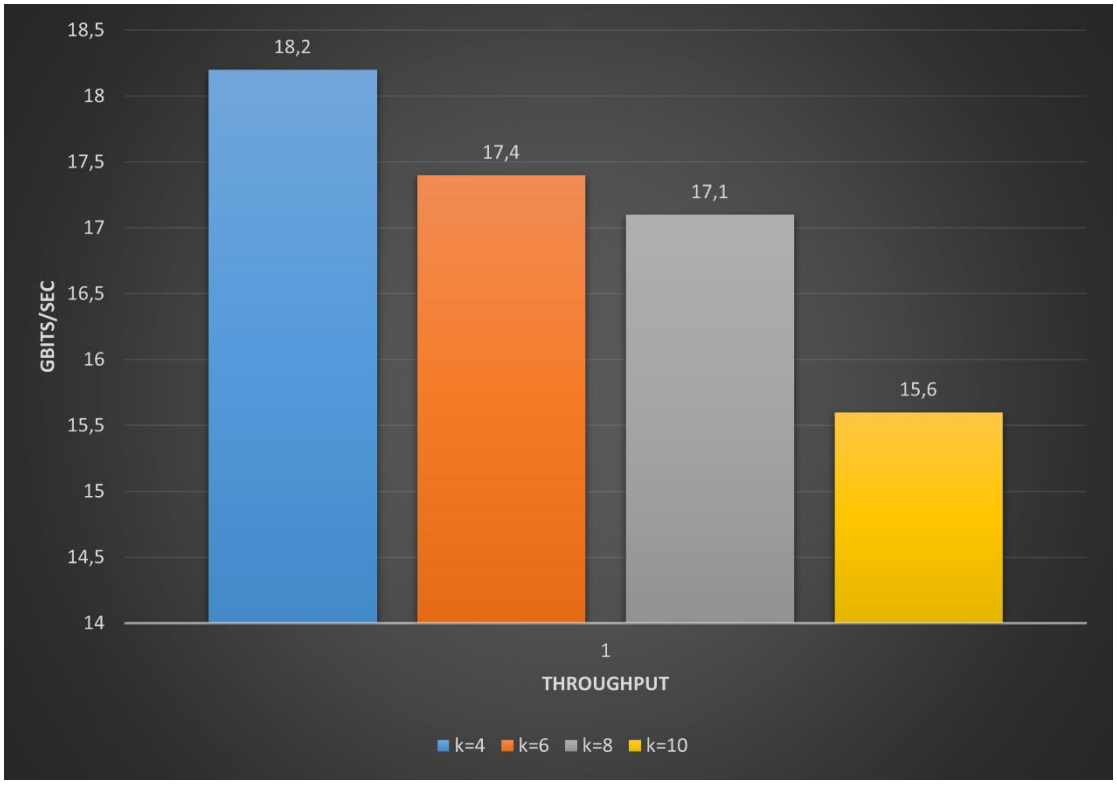


Εικόνα 7-17 Throughput of Onos at Fattree topology



Εικόνα 7-18 Fattree Topology k=10

Στην εικόνα 7-19 βλέπουμε την μέση τιμή απόδοσης του OpenDayLight. Καταλαβαίνουμε πως η απόδοσή του είναι πολύ μέτρια σε σύγκριση με όλους τους άλλους ελεγκτές. Σε άλλες έρευνες και εργασίες που έχουν γίνει κατά καιρούς AL-Fares et al. (Al-Fares, 2008), σε πολλά πειράματα η απόδοση του OpenDayLight ήταν σχετικά χαμηλή, χωρίς όμως να μπορεί εξηγηθεί επαρκώς αυτή η συμπεριφορά. Πιστεύετε ότι οφείλεται σε κάποιο σφάλμα (bug) του πρωτοκόλλου OpenFlow. Βέβαια είδαμε προηγουμένως στην απλή τοπολογία πως τα επίπεδα της απόδοσης του, ήταν πολύ ικανοποιητικά. Είναι ένα θέμα που χρίζει περισσότερης μελέτης και έρευνας. Επίσης παρόλη τη χαμηλή απόδοση δεν υπάρχει εξίσου και σε αυτόν τον ελεγκτή κάποια σταθερότητα στην απόδοση του αναλογικά με το μέγεθος των τοπολογιών. Η διακύμανση των τιμών ξεκινάει από τα 15,6 Gbits/sec στην μεγαλύτερη μορφή της τοπολογίας και φτάνει ως τα 18,2 Gbits/sec.

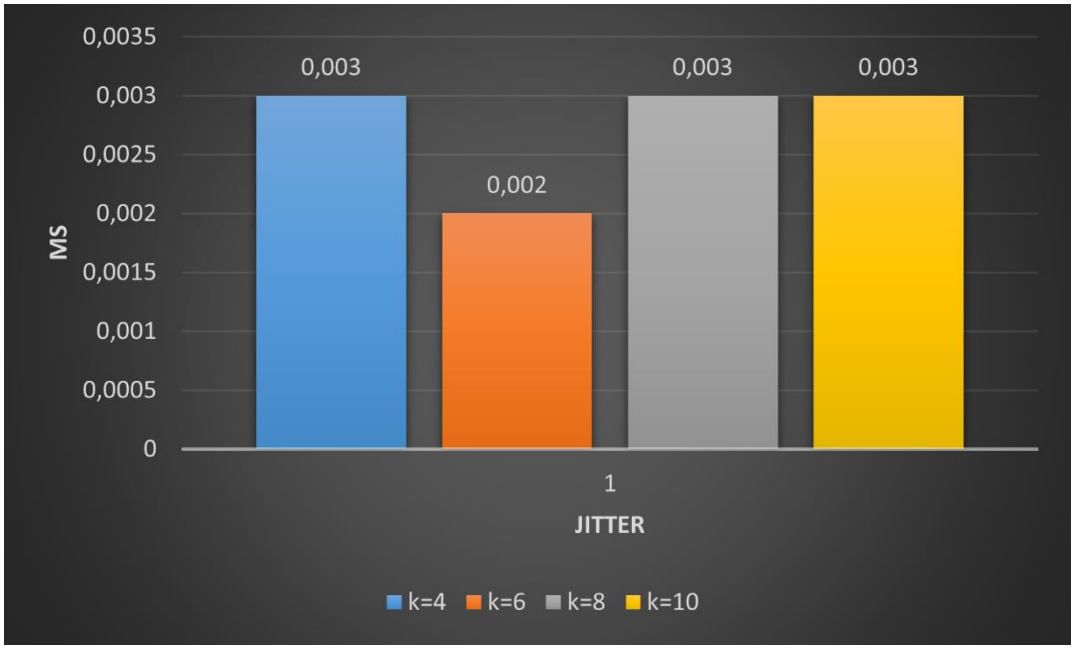


Εικόνα 7-19 Throughput of OpenDayLight at Fattree topology

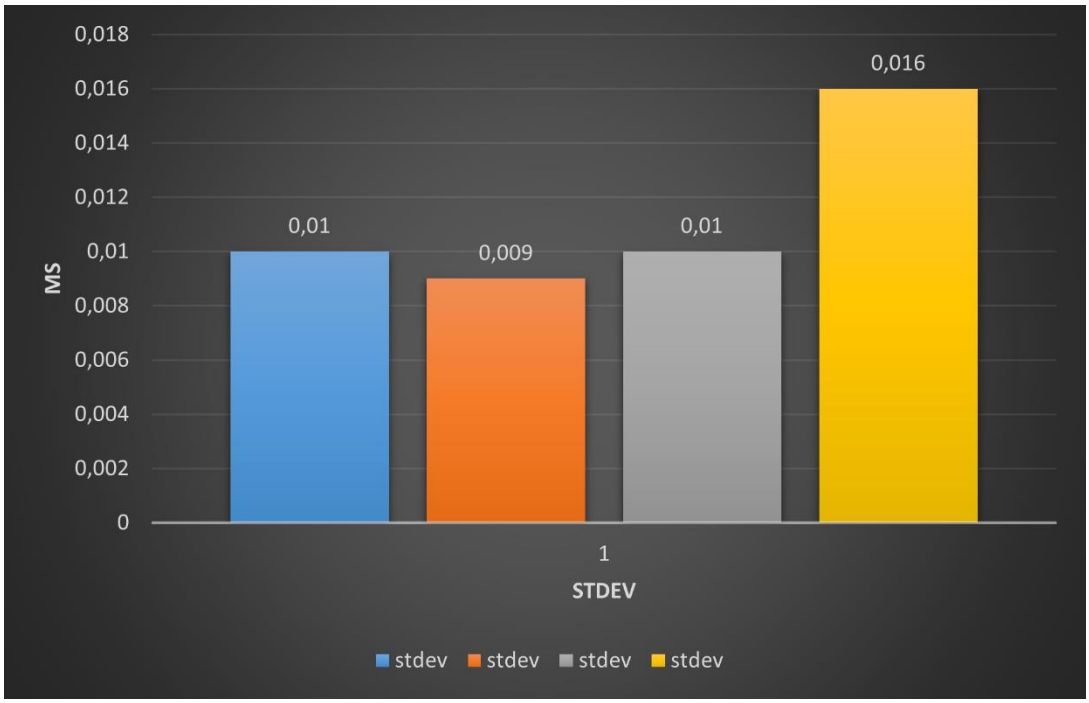
7.2.2 Μελέτη απόδοσης UDP

Με τον ίδιο τρόπο εργαστήκαμε και στην περίπτωση μελέτης απόδοσης με χρήση του πρωτοκόλλου UDP, για να πάρουμε μετρήσεις από τον εκάστοτε ελεγκτή σε θέματα καθυστέρησης στην τοπολογία Fattree. Δημιουργήσαμε την τοπολογία σε τέσσερα διαφορετικά μεγέθη, δηλαδή για κ=4, 6, 8, 10 όπως και στις προαναφερόμενες μετρήσεις. Όσο αυξάνεται το μέγεθος της τοπολογίας (παράμετρος κ), τόσο αυξάναμε αντίστοιχα και το πλήθος των τυχαίων μετρήσεων μεταξύ των hosts.

Στις εικόνες 7-20, 7-21 και 7-22 βλέπουμε τα αποτελέσματα των μέσων τιμών των παραμέτρων της καθυστέρησης που μελετάμε του ελεγκτή Ryu.

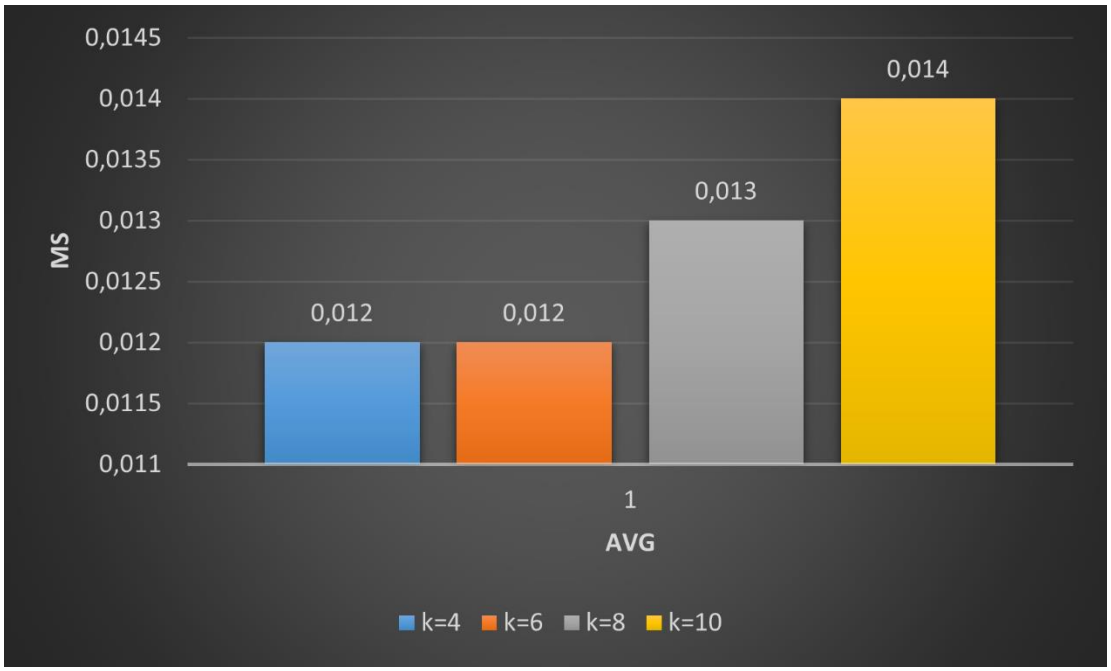


Παρατηρούμε πως το Jitter του ελεγκτή, παρόλη την αλλαγή μεγέθους της τοπολογίας παραμένει σταθερό, καθώς έχει μια πολύ μικρή διακύμανση της τάξεως του 0,001 ms. Αναμέναμε οι τιμές να είναι λίγο μεγαλύτερες απ' ότι στην απλή τοπολογία που είδαμε προηγουμένως αλλά βλέπουμε ότι οι τιμές είναι παρόμοιες.



Εικόνα 7-20 Stdev of Ryu controller

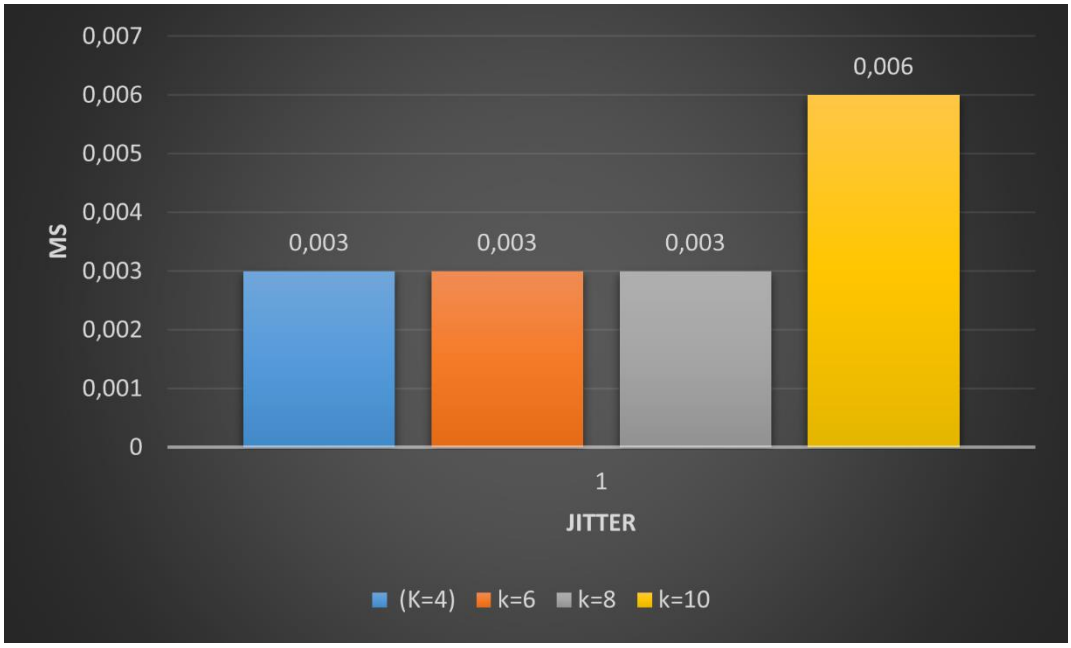
Όσο αφορά την τυπική απόκλιση, κυμαίνεται στα ίδια επίπεδα σχεδόν σε όλα τα μεγέθη της τοπολογίας. Η μέση τιμή της είναι στα 0,010 ms και φτάνει στα 0,016 ms όταν μεγαλώνουμε το μέγεθος της τοπολογίας. Αρκετά μεγαλύτερες οι τιμές της μέσης τυπικής απόκλισης απ' ότι στην απλή τοπολογία της προηγούμενης ενότητας, κάτι που είναι φυσιολογικό, καθώς πρόκειται για μια μεγάλη και περίπλοκη τοπολογία.



Εικόνα 7-21 average delay of Ryu controller

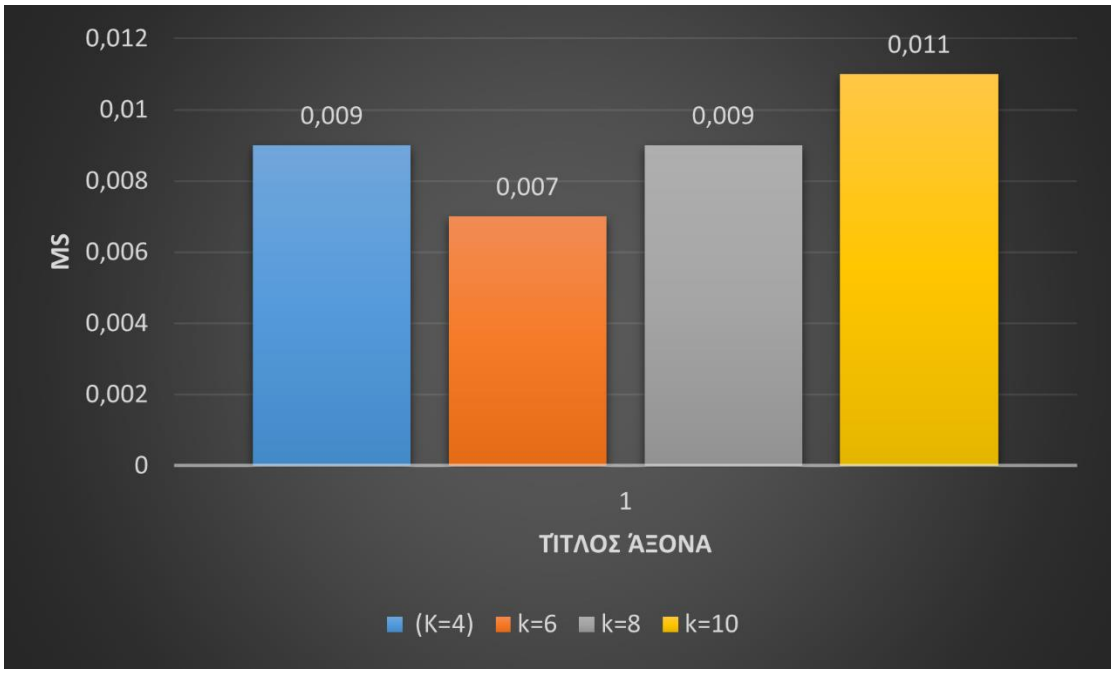
Ο μέσος όρος της καθυστέρησης του Ryu είναι αναλογικός με το μέγεθος της τοπολογίας. Δηλαδή όσο μεγαλύτερη κάναμε την τοπολογία τόσο αυξάνεται ο μέσος όρος της καθυστέρησης. Οι τιμές είναι στα 0,012 ms στα μικρά μεγέθη και φτάνει στο 0,014 ms. Είναι τρείς φορές πάνω οι μέσες τιμές της καθυστέρησης από ότι στην απλή τοπολογία.

Συνεχίζοντας με τον Floodlight, βλέπουμε τα αποτελέσματα τις καθυστέρησης στα παρακάτω γραφήματα.



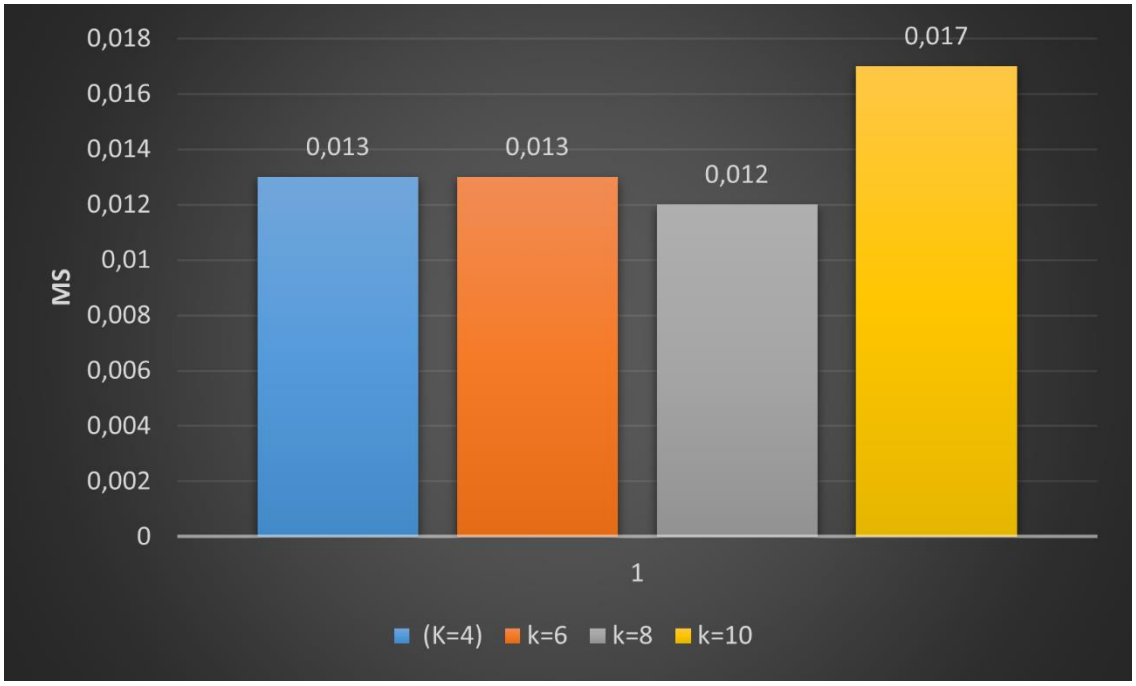
Εικόνα 7-22 Jitter of Floodlight controller

Παρατηρούμε στην τοπολογία με κ=10, το Jitter διπλασιάζεται σε σχέση με τίς άλλες μικρότερες σε μέγεθος τοπολογίες Για τα πρώτα τρία μεγέθη της τοπολογίας FatTree παρατηρείται παρόμοια συμπεριφορά με αυτή του ελεγκτή Ryu, δηλαδή το Jitter λαμβάνει την τιμή 0,003 ms., όπως επίσης και με αυτή όσο αφορά τον ίδιο ελεγκτή στην απλή τοπολογία.



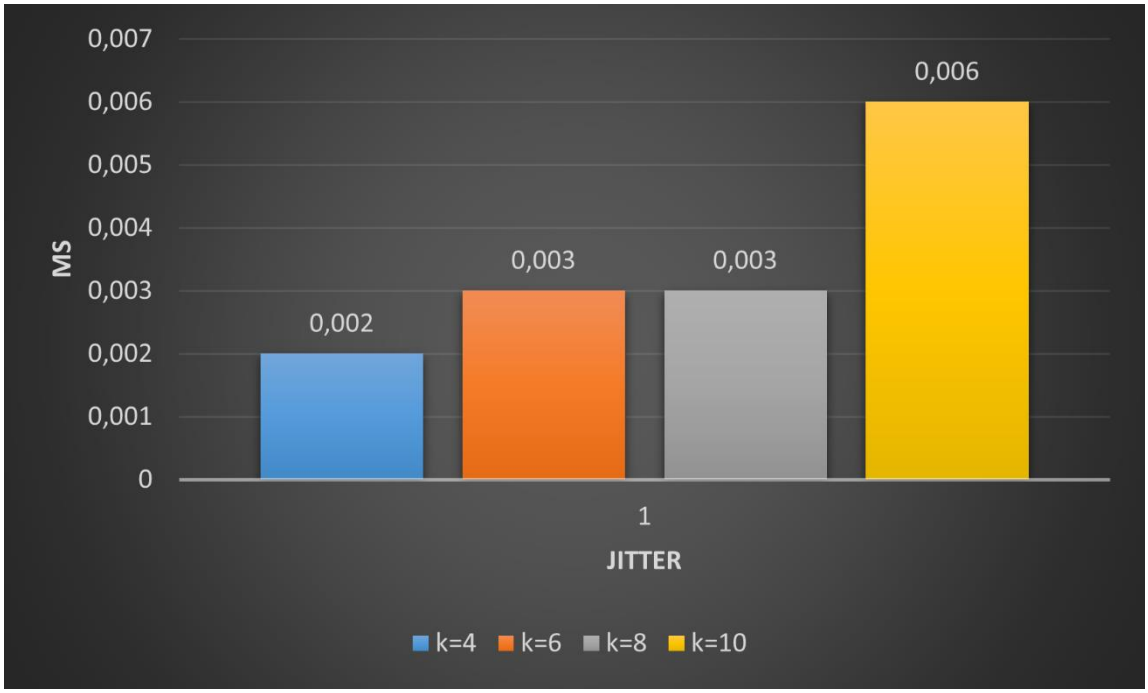
Εικόνα 7-23 stdev of Floodlight controller

Η τυπική απόκλιση είναι σε χαμηλότερα επίπεδα σε σχέση με τον Ryu σε όλα τα πειράματα. Έχουμε ως μικρότερη τιμή τα 0,007 ms και βλέπουμε πως και εδώ έχουμε μια σχετική σταθερότητα και διακύμανση μεταξύ των πειραμάτων.

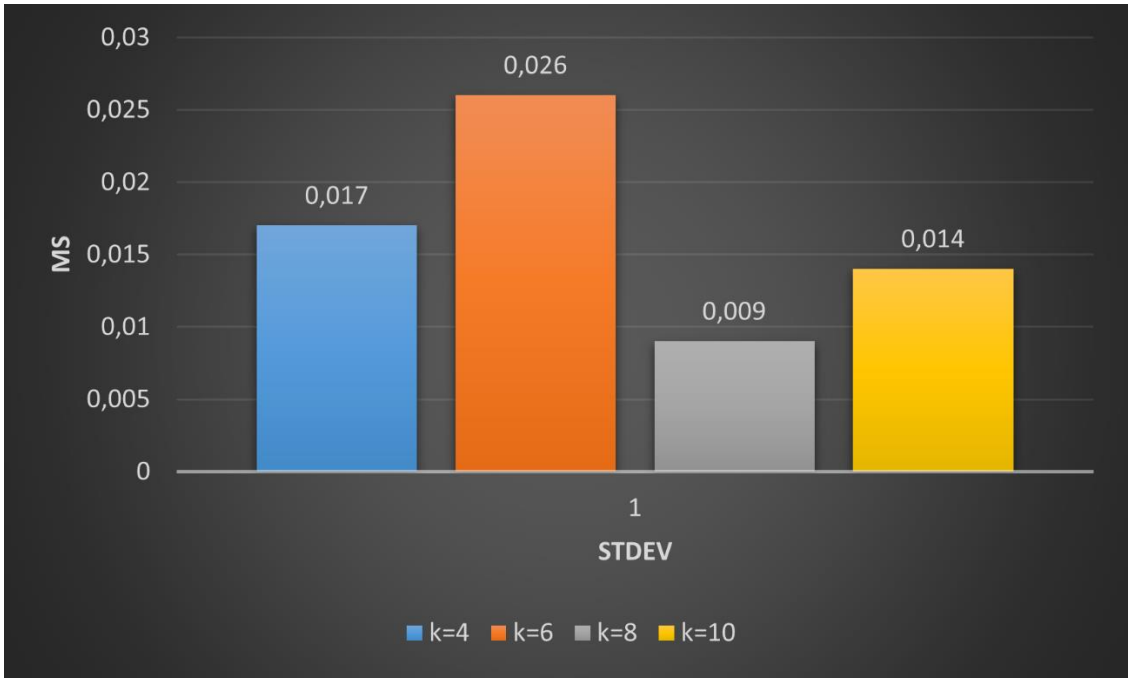


Εικόνα 7-24 average delay of Floodlight controller

Ο μέσος χρόνος της καθυστέρησης κυμαίνεται από 0,013 ms έως 0,017 ms, ανάλογα με την παράμετρο κ της τοπολογίας Fattree. Θεωρούμε ότι τα αποτελέσματα είναι αναμενόμενα καθώς τώρα αυξάνει το πλήθος των συνδέσεων μέσω των οποίων διέρχονται τα δεδομένα σε σχέση με την απλή τοπολογία. Βέβαια, για κ=8 παρατηρείται μια ελάχιστη μείωση που μπορεί όμως να οφείλεται και σε στατιστικό λάθος.

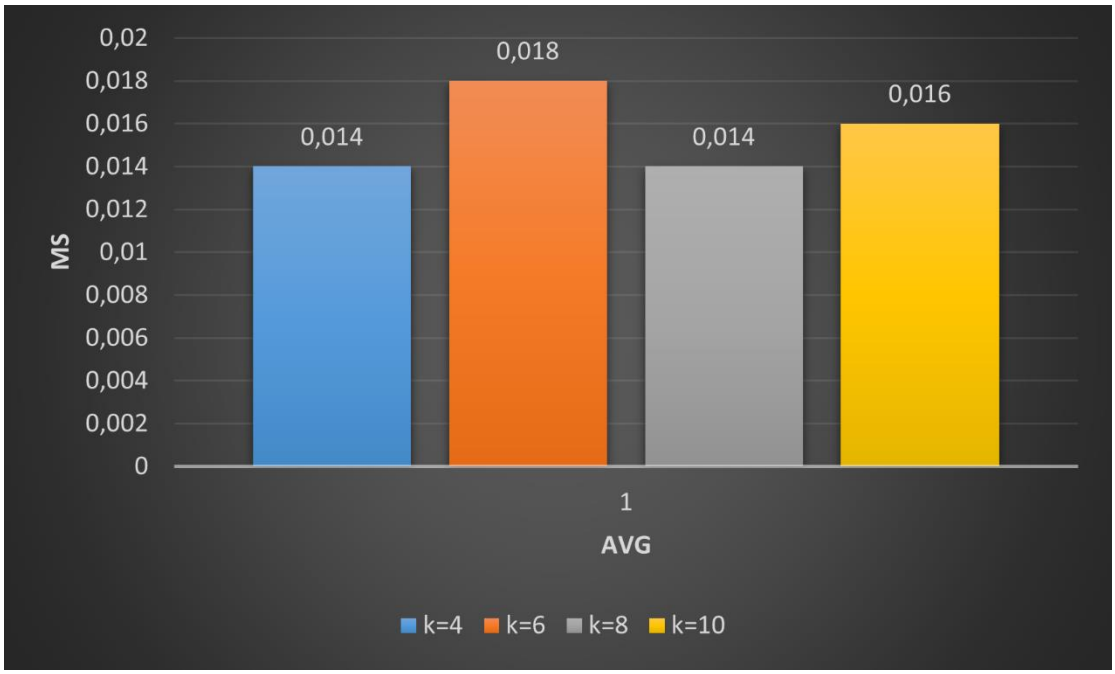


Το Jitter παραμένει και εδώ ίδιο όπως με τους υπόλοιπους ελεγκτές όσο αφορά το UDP στην τοπολογία Fattree. Ο ONOS έχει μια σταθερή τιμή στο jitter που είναι 0,003 ms όπου και διπλασιάζεται στα 0,006 ms όταν μεγαλώσουμε την τοπολογία.



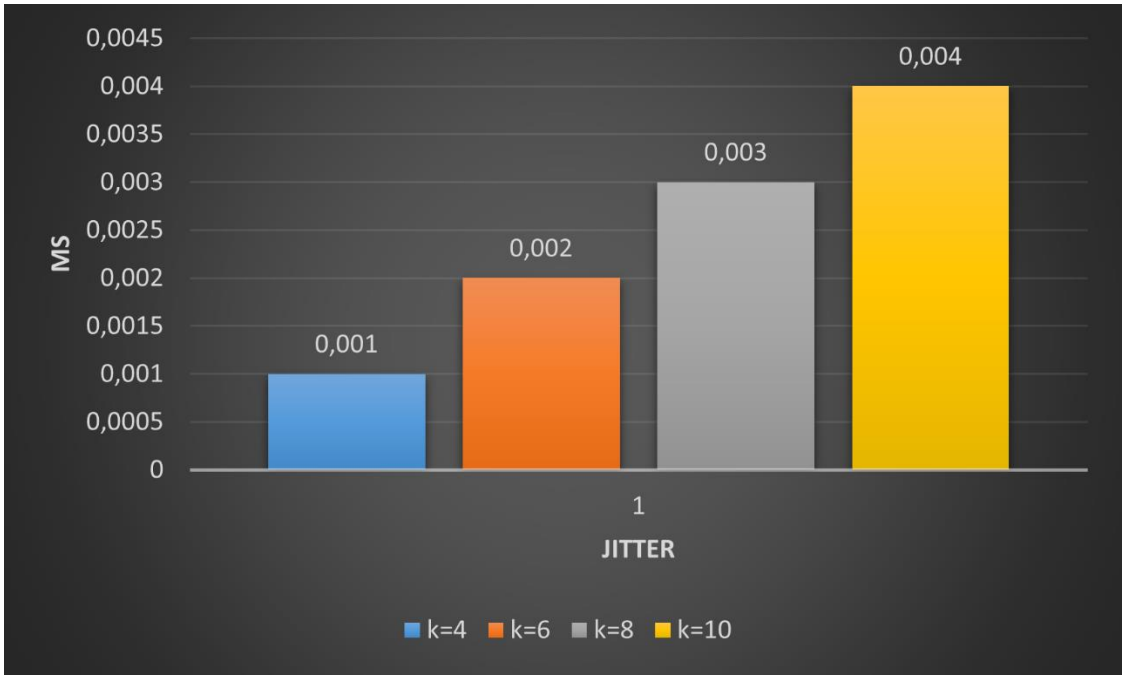
Εικόνα 7-25 stdev of Onos

Μεγάλη τυπική απόκλιση παρουσιάζει ο Onos στην τοπολογία Fattree, καθώς βλέπουμε τιμές που φτάνουν στα 0,026 ms. Επίσης παρατηρείται και μεγάλη διαφοροποίηση μεταξύ διαφορετικών μεγεθών του δικτύου, κάτι το οποίο προκαλεί εντύπωση καθώς περιμέναμε μια σταθερότητα όπως είδαμε στην απόδοση του. Δεν μπορούμε να γνωρίζουμε που οφείλονται οι τόσο μεγάλες τιμές στην τυπική απόκλιση. Επισημαίνουμε ότι χρησιμοποιούμε την τελευταία έκδοση του ελεγκτή.



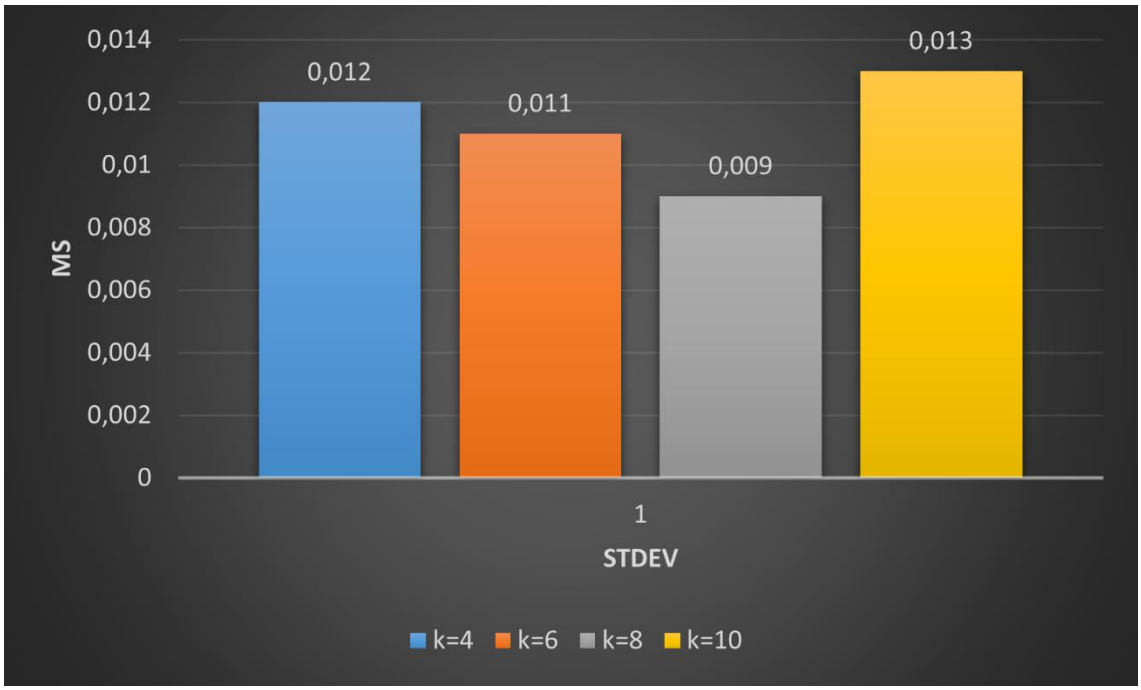
Εικόνα 7-26 avg of Onos

Η μέση καθυστέρηση του ONOS όπως παρατηρούμε είναι πιο μεγάλη από όλους τους προηγούμενες ελεγκτές. Οι μικρότερες τιμές είναι 0,014 ms και φτάνει ως 0,018 ms. Αρκετά προβληματική η συμπεριφορά του σε θέματα καθυστέρησης, τουλάχιστον σε περίπλοκες τοπολογίες όπως αυτή του Fattree.



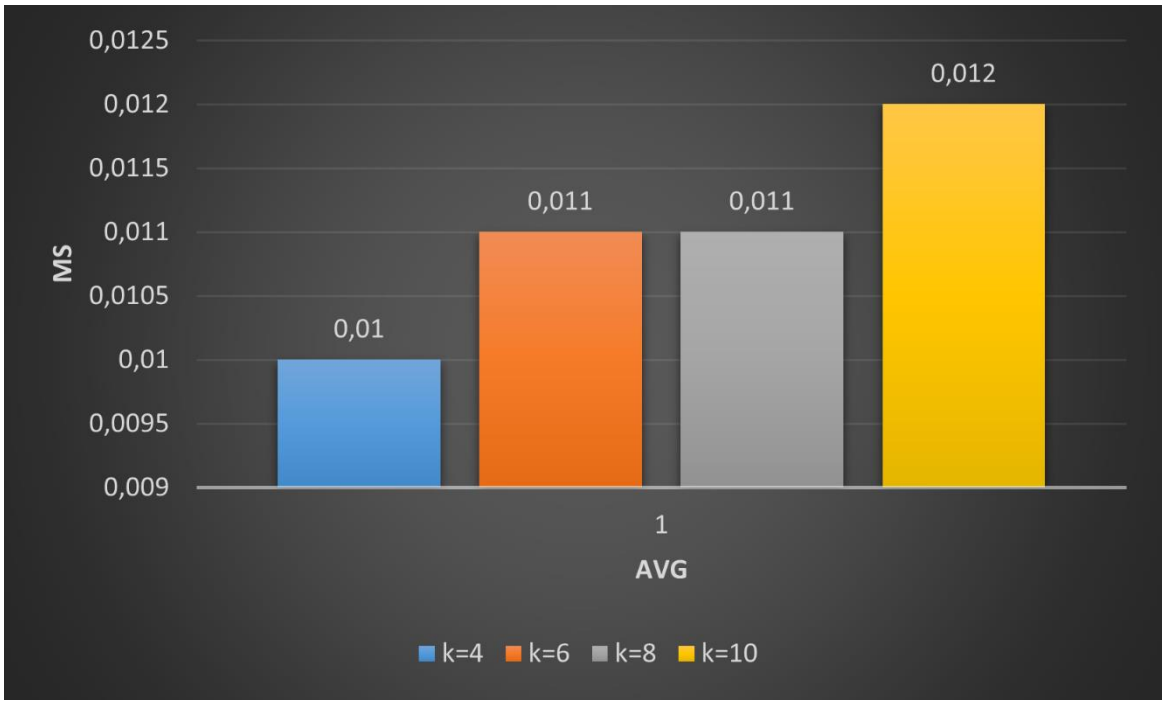
Εικόνα 7-27 Jitter of OpenDayLight

Όσο αφορά το Jitter του OpenDayLight, διακρίνουμε μια τυπική αναλογία όσο μεγαλώνουμε την τοπολογία όπου σε όλες τις περιπτώσεις διατηρείτε σε χαμηλά επίπεδα. Επίσης αν και με μικρή διαφορά έχει τις χαμηλότερες τιμές απ' όλους τους προηγούμενες ελεγκτές.



Εικόνα 7-28 stdev of OpenDayLight

Οι μέσες τιμές της τυπικής απόκλισης του OpenDayLight είναι υψηλές, όπως και στον ONOS. Στην τοπολογία με κ=8, είναι 0,008 ms, με την διαφοροποίηση στα υπόλοιπα μεγέθη να είναι μεταξύ 0,011 ms και 0,013 ms.



Εικόνα 7-29 Average delay of OpenDayLight

Σε σύγκριση με τους υπόλοιπους ελεγκτές, θα μπορούσαμε να θεωρήσουμε φυσιολογικό τον μέσο όρο καθυστέρησης, καθώς στην μικρότερη τοπολογία είναι 0,010 ms και στις μεγαλύτερες κυμαίνεται στα 0,012 ms.

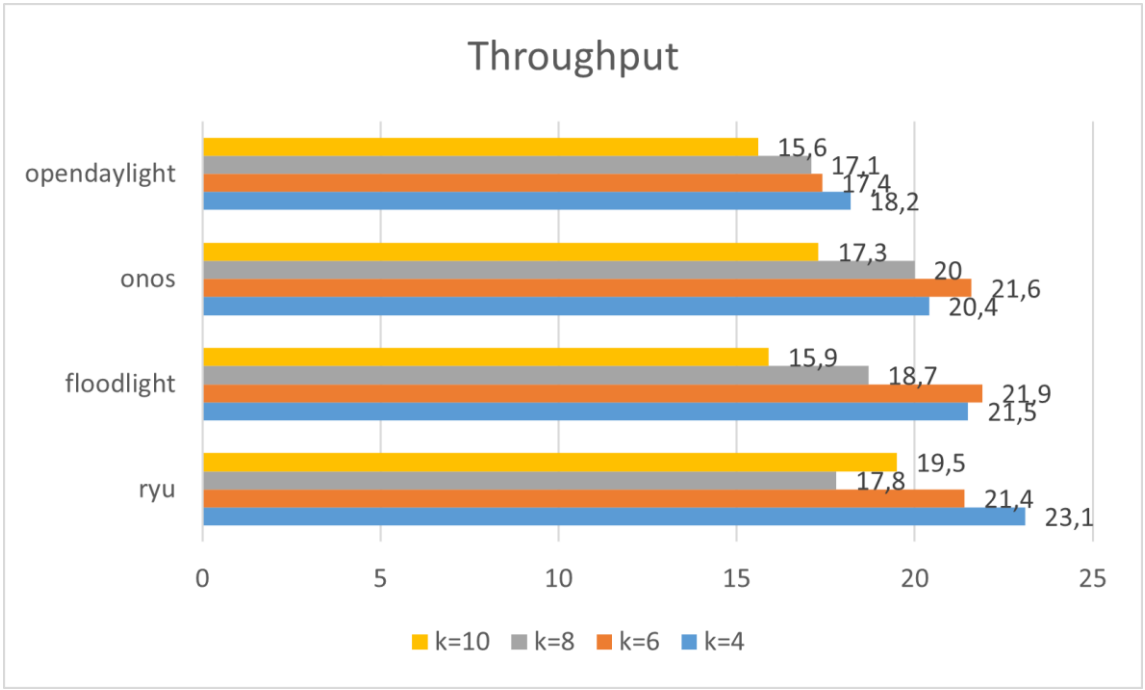
8 Αξιολόγηση

8.1 Ανάλυση Αποτελεσμάτων

Στην ενότητα αυτή θα δούμε τα τελικά συμπεράσματα της μελέτης. Ξεκινώντας με την απόδοση στην απλή τοπολογία, παρατηρούμε πως οι τιμές απόδοσης όλων των ελεγκτών κυμαίνονται από 27,6 Gbits/sec και φτάνει ως τα 30,5 Gbits/sec. Αισθητή γίνετε η διαφορά απόδοσης του ONOS που εμφανίζεται υψηλότερη αλλά και πιο σταθερή. Υψηλή απόδοση έχουμε και στον Ryu που πιάνει τις τιμές του OpenDayLight, κάτι το οποίο δεν περιμέναμε καθώς σε άλλες επιστημονικές εργασίες (Salman, 2016) (Jehad Ali, 2018) που έχει χρησιμοποιηθεί σε παρόμοιες τοπολογίες η απόδοση του είναι χαμηλότερη. Ο OpenDayLight ακολουθεί στις υψηλές αποδόσεις τον Onos και τελευταίος με τις χαμηλότερες τιμές είναι ο Floodlight.

Όσον αφορά την καθυστέρηση στο UDP, παρατηρούμε πως όλοι οι παράμετροι που εξετάστηκαν (average,standar deviation, jitter) είναι σχεδόν στα ίδια επίπεδα, κάτι το οποίο δεν προκαλεί εντύπωση, καθώς είναι και οι τέσσερις ελεγκτές πολύ ισχυροί και τους χρησιμοποιήσαμε σε απλή τοπολογία για να δούμε την γενική συμπεριφορά τους.

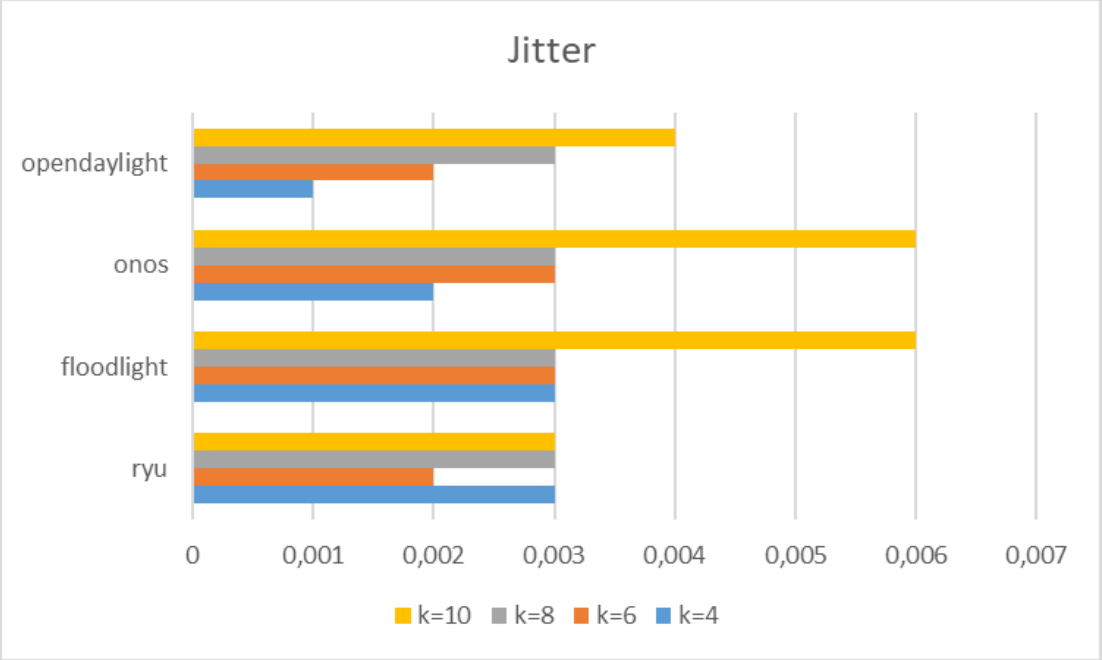
Συνεχίζοντας, δημιουργήσαμε την τοπολογία FatTree και πήραμε μετρήσεις με τους ίδιους ελεγκτές σε διαφορετικά μεγέθη της (κ=4, 6, 8, 10). Στο παρακάτω γράφημα (εικόνα 8-1) βλέπουμε τις συνολικές αποδόσεις όλων των ελεγκτών συγκριτικά.



Εικόνα 8-1 Throughput of controllers

Ξεκινώντας με τον OpenDayLight, όπως είδαμε οι αποδόσεις του είναι σε πολύ χαμηλά επίπεδα σε σχέση με τους άλλους ελεγκτές. Ο Ryu έφτασε τα 23,1 Gbits/sec για κ=4, όπου είναι και η υψηλότερη τιμή, αλλά όσο μεγαλώναμε την τοπολογία έπεφτε η απόδοση του κάτι το οποίο είναι φυσιολογικό, αλλά δεν κράτησε σταθερές τις αποδόσεις του, κάτι το οποίο θέλαμε να δούμε σε κάποιον ελεγκτή. Βέβαια το είδαμε στον ONOS, και στην FatTree τοπολογία που είναι περίπλοκη και μεγάλη κατάφερε να κρατήσει σταθερή απόδοση αλλά και σε υψηλές τιμές παρόλες τις αλλαγές μεγέθους της τοπολογίας που κάναμε. Τέλος, ο Floodlight είχε μια αναλογία στις μέσες τιμές καθώς όσο μεγαλώναμε την τοπολογία τόσο υστερούσε, ξεκινώντας από τα 21,5 Gbits/sec ‘όπου είναι μια πολύ καλή τιμή για μια πολύπλοκη τοπολογία σαν αυτή, και έφτασε σε πολύ χαμηλά επίπεδα της τάξεως των 15,9 Gbits/sec.

Οι παράμετροι της καθυστέρησης που προαναφέραμε, όπου εξετάστηκαν και σε αυτήν την τοπολογία παρατηρήσαμε ότι ήταν σχετικά στα ίδια επίπεδα μεταξύ όλων των ελεγκτών, όπως και στην απλή τοπολογία. Στις εικόνες 8-2, 8-3, 8-4 βλέπουμε τις συγκεντρωτικές τιμές.

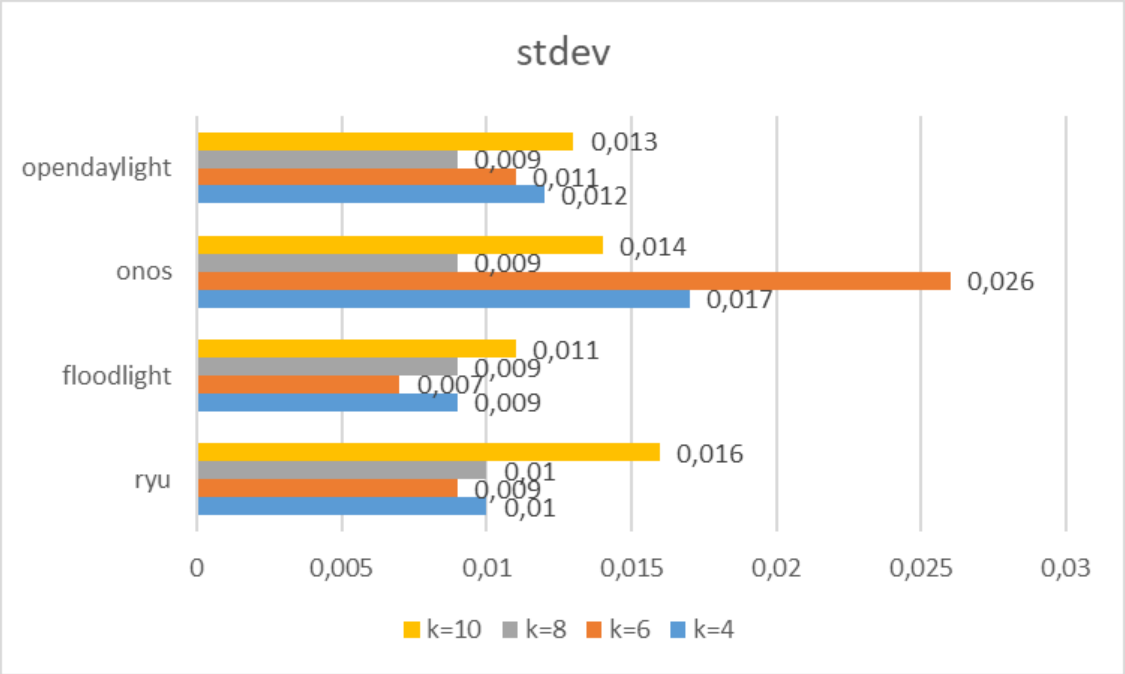


Εικόνα 8-2 Jitter of controllers

Το Jitter αν εξαιρέσουμε τον Onos και τον Floodlight όπου στην τοπολογία Fattree κ=10, διπλασιάζεται από τον μέσο όρο των άλλων είναι ακριβώς στα ίδια επίπεδα. Να τονίσουμε εδώ πως η τιμή του jitter στα 0,006 ms δεν επηρεάζει πουθενά τον ελεγκτή σε θέματα απόδοσης σε σχέση με την τιμή 0,003 ms.

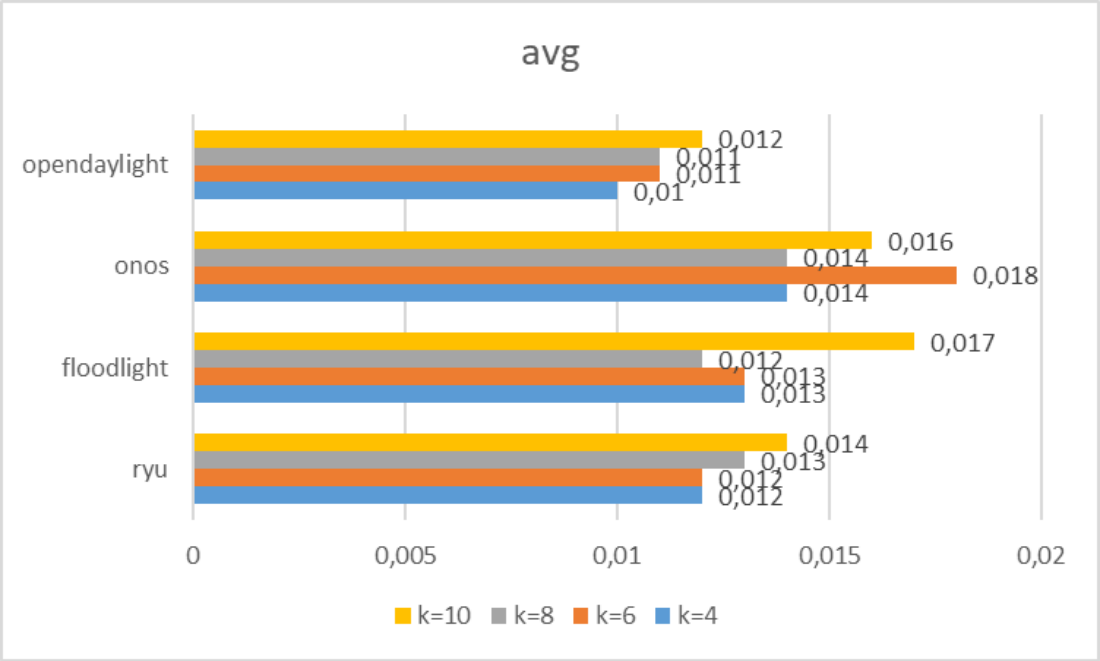
Υψηλή τυπική απόκλιση συναντάμε στον Onos, όπως βλέπουμε στην εικόνα 8-3, το οποίο σημαίνει πως όσο καλή και σταθερή απόδοση έχει σαν ελεγκτής, υστερεί αρκετά σε θέματα καθυστέρησης. Έχει τις πιο μεγάλες τυπικές αποκλίσεις σε

μέσες τιμές που φτάνουν τα 0,026 ms κάτι το οποίο τον επηρεάζει αρνητικά. Οι υπόλοιποι τρείς ελεγκτές έχουν σχετικά τις ίδιες μέσες τιμές τυπικών αποκλίσεων, με τον Ryu σε μεγάλα μεγέθη της τοπολογίας να έχει ως μέσο όρο τα 0,016 ms.



Εικόνα 8-3 stdev of controllers

Τέλος, ο μέσος όρος της καθυστέρησης των ελεγκτών, έχει μια τυπική διακύμανση από 0,011 ms έως 0,018, όπου την μεγαλύτερη τιμή την έχει πάλι ο Onos και ακολουθεί ο Floodlight. Οι διαφορές αυτές όπως βλέπουμε (Εικ.8-3) είναι πολύ μικρές και σε γενικά πλαίσια όπως και στην απλή τοπολογία οι καθυστερήσεις των ελεγκτών στο UDP της τοπολογίας Fattree είναι παρόμοιες. Καταλήγουμε στο συμπέρασμα πως σε όλες τις μορφές της τοπολογίας που χρησιμοποιήσαμε ο Onos έχει τις καλύτερες αποδόσεις αλλά με αρκετά μεγάλες καθυστερήσεις ειδικά στην τοπολογία Fattree. Ακολουθεί σε θέματα απόδοσης με μικρή διαφορά από τους άλλους ο Ryu με πολύ καλά ποσοστά. Μεγάλο ερωτηματικό παραμένει η πολύ μέτρια απόδοση του OpenDayLight, καθώς δεν καταφέραμε να επιλύσουμε αυτό το ζήτημα ύστερα από πολλή έρευνα και προσπάθειες.



Εικόνα 8-4 avg of controllers

8.2 Σύγκριση με εργασίες άλλων

Τα αποτελέσματα μας, συμφωνούμε αρχικά με την έρευνα του Mamushiane et al. (2018), όσο αφορά την απλή τοπολογία καθώς ο ONOS εμφανίζει παρόμοια συμπεριφορά και καλύτερη απόδοση.

Ο Priya et al. (2019) σε θέματα καθυστέρησης σε διάφορες απλές τοπολογίες, εμφανίζει τον Floodlight να έχει τις μικρότερες καθυστερήσεις. Στην δικιά μας απλή τοπολογία παρατηρήσαμε πως και οι τέσσερις ελεγκτές κυμαίνονται στα ίδια επίπεδα.

Τον OpenDayLight προτείνει ως καλύτερο ελεγκτή σε θέματα καθυστέρησης και απόδοσης ο RAJU et al. (RAJU, 2018) συγκρίνοντας τον και με τον Ryu αλλά και με τον Floodlight χωρίς όμως να περιλαμβάνει τον ONOS στην έρευνα του. Παρατηρήσαμε πως και στην δική μας εργασία ο OpenDayLight έρχεται δεύτερος μετά τον ONOS στην απλή τοπολογία.

Σε μια απλή τοπολογία δέντρου, ο Stancu et al. (Stancu1, 2015), έτρεξε τους ελεγκτές Ryu, ONOS και OpenDayLight με την χρήση της εντολής iperf, για θέματα απόδοσης και συμπέρανε πως ONOS και OpenDayLight είχαν τα καλύτερα αποτελέσματα, πράγμα που συμφωνεί και με τα δικά μας αποτελέσματα σε αυτή την περίπτωση.

Σε τοπολογία FatTree εργάστηκε και ο Cabarkara et al. (CABARKAPA, 2021) με 54 hosts, και σύγκρινε τους ελέγκτές Ryu και POX. Η μέση τιμή απόδοσης που του έδωσε ο Ryu κυμαίνεται στα 17.02 Gbits/sec, τιμή η οποία είναι περίπου στα 4 Gbits/Sec λιγότερη από τις δικές μας στην ανάλογη τοπολογία. Παρόμοιες τιμές είχαμε βέβαια, καθώς διάφοροι παραμέτροι παίζουν ρόλο στα αποτελέσματα όπως η έκδοση του ελεγκτή, το εικονικό περιβάλλον, εργαλεία σύγκρισεις κ.α. τα οποία δεν ήταν όλα κοινά.

9 Συμπεράσματα

Στα πλαίσια αυτής της διπλωματικής εργασίας μελετήθηκαν οι ελεγκτές SDN, και πιο συγκεκριμένα αξιολογήθηκαν πειραματικά οι ελεγκτές ONOS, Ryu OpenDayLight και Floodlight. Για το σκοπό αυτό διαμορφώθηκε το κατάλληλο περιβάλλον δοκιμών σε μια εικονική μηχανή με λειτουργικό σύστημα Ubuntu 20.0.4 όπου εγκαταστάθηκε το λογισμικό των παραπάνω ελεγκτών. Χρησιμοποιήθηκαν δύο είδη τοπολογιών, η απλή τοπολογία και η τοπολογία FatTree, οι οποίες υλοποιήθηκαν με τη βοήθεια του Mininet. Από την έρευνα αυτή προέκυψαν οι παρακάτω διαπιστώσεις και συμπεράσματα:

- Για την συγκριτική αξιολόγηση των ελεγκτών SDN απαιτείται ένα σταθερό και μεθοδολογικό πλαίσιο.

- Η συμπεριφορά των ελεγκτών εξαρτάται από το εκάστοτε σενάριο αλλά και τις συγκεκριμένες παραμέτρους που χρησιμοποιούνται (π.χ. είδος τοπολογίας, πλήθος κόμβων, κλπ.)

- Στην απλή τοπολογία, ο ONOS εμφανίζει υψηλότερη απόδοση και καλύτερες επιδόσεις. Επίσης, διαπιστώνεται σταθερότητα συμπεριφοράς, για όλες τις πιθανές διαδρομές επικοινωνίας. Ακολουθεί ο OpenDayLight δεύτερος με πολύ ικανοποιητικά αποτελέσματα απόδοσης.

- Σε θέματα καθυστέρησης στην απλή τοπολογία, όπως είδαμε όλοι οι ελεγκτές κυμαίνονται στα ίδια επίπεδα χωρίς να ξεχωρίζει κάποιος.

- Στην τοπολογία Fattree ο ONOS και RYU έχουν τις καλύτερες μέσες τιμές σε θέματα απόδοσης και στο μεγαλύτερο μέγεθος της τοπολογίας που χρησιμοποιήσαμε για κ=10, παρατηρήσαμε ότι η απόδοση του Ryu είναι η καλύτερη.

- Ως προς την καθυστέρηση, αυτοί οι δύο ελεγκτές (ONOS και RYU) και ιδιαίτερα ο ONOS έχουν τις μεγαλύτερες κατά μέσο όρο καθυστερήσεις. Το ίδιο ισχύει και για την τυπική απόκλιση και την διακύμανση.

- Μεγάλο προβληματισμό δημιουργεί η απόδοση του OpenDayLight στην τοπολογία fattree όπου είναι σε πολύ χαμηλά επίπεδα. Αυτό ίσως να οφείλεται όπως αναφέρει και στην έρευνα του Mamadou et al. (Mamadou-Tahirou.Bah, 2019) σε κάποιο σφάλμα μεταξύ του ελεγκτή και του πρωτόκολλου Openflow.

Από τα παραπάνω συμπεραίνουμε ότι ο ONOS και ο RYU δίνουν τα καλύτερα πειραματικά αποτελέσματα, ενώ ο προσδιορισμός των παραγόντων που αναστέλει τις επιδόσεις των άλλων δύο ελεγκτών και ιδιαίτερα του OpenDayLight αποτελούν μελλοντικό πεδίο έρευνας.

Αναφορές

SPARQL 1.1 Query Language. W3.org. (2013). Ανακτήθηκε 26 Ιουλίου 2022, από https://www.w3.org/TR/sparql11-query/)

Open Networking Foundation (2022). Ανακτήθηκε 26 ιουλίου 2022, από https://opennetworking.org/

A. Vishnu Priya, N. R. (2019). *Performance comparison of SDN OpenFlow.* India: Inderscience Enterprises Ltd.

Abdelghny Orogat, A. E.-R. (2021). *CBench: Demonstrating Comprehensive Evaluation of Question.* Ottawa/Canada.

Al-Fares, M. (2008). A Scalable, Commodity Data Center Network Architecture

Al-Shabib, A. (2015, March 5). *POX WIKI .* Ανάκτηση από http://intronetworks.cs.luc.edu/auxiliary_files/mininet/poxwiki.pdf

Arahunashi, A. K., Neethu, S., & Aradhya, H. R. (2019, May). Performance analysis of various sdn controllers in mininet emulator. In 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT) (pp. 752-756). IEEE. Bholebawa, I. Z. (2018). *SpringerLink*. Ανάκτηση από Performance Analysis of SDN/OpenFlow Controllers: POX Versus Floodlight: https://link.springer.com/article/10.1007/s11277-017-4939-z

Bispo, P., Corujo, D., & Aguiar, R. L. (2017, May). A qualitative and quantitative assessment of sdn controllers. In 2017 International Young Engineers Forum (YEF-ECE) (pp. 6-11). IEEE. C.Fancy. (2017). *Performance Evaluation of SDN controllers POX and Floodlight in Mininet Emulation Environment.* Tamil Nadu, India: International Conference on Intelligent Sustainable Systems (ICISS 2017).

Cabarkapa, D., & Rancic, D. (2021). Performance Analysis of Ryu-POX Controller in Different Tree-Based SDN Topologies. Advances in Electrical and Computer Engineering, 21(3), 31-38.*cables-solutions*. (χ.χ.). Ανάκτηση από https://www.cables-solutions.com/whats-openflow-switch-how-it-works.html

Erickson, D. (2013, August). The beacon openflow controller. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (pp. 13-18).

Yamei, F., Qing, L., & Qi, H. (2016, October). Research and comparative analysis of performance test on SDN controller. In 2016 first IEEE international conference on computer communication and the internet (ICCCI) (pp. 207-210). IEEE. *faucet.nz*. (χ.χ.). Ανάκτηση από Faucet Design and Architecture: https://docs.faucet.nz/en/latest/architecture.html#faucet-design-and-architecture

Fernandez, M. P. (2013, March). Comparing openflow controller paradigms scalability: Reactive and proactive. In 2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA) (pp. 1009-1016). IEEE.

Installation Guide (2018). Ανακτήθηκε 26 Ιουλίου 2022, από https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343544/Installation+Guide

Ribes Garcia, B. (2015). OpenDaylight SDN controller platform (Bachelor's thesis, Universitat Politècnica de Catalunya). *Google Code*.

maestro-platform. (2022) Ανακτήθηκε 26 Ιουλίου 2022, από https://code.google.com/archive/p/maestro-platform/

Bholebawa, I. Z., & Dalal, U. D. (2018). Performance analysis of SDN/OpenFlow controllers: POX versus floodlight. Wireless Personal Communications, 98(2), 1679-1699.

Ali, J., Lee, S., & Roh, B. H. (2018, April). Performance analysis of POX and Ryu with different SDN topologies. In Proceedings of the 2018 international conference on information science and system (pp. 244-249).

Bailey, J., & Stuart, S. (2016). Faucet: Deploying SDN in the enterprise. Communications of the ACM, 60(1), 45-49.

Kaur, K., Singh, J., & Ghumman, N. S. (2014, February). Mininet as software defined networking testing platform. In International conference on communication, computing & systems (ICCCS) (pp. 139-42).

Khondoker, R., Zaalouk, A., Marx, R., & Bayarou, K. (2014, January). Feature-based comparison and selection of Software Defined Networking (SDN) controllers. In 2014 world congress on computer applications and information systems (WCCAIS) (pp. 1-7). IEEE.

Sharma, K. K., & Sood, M. (2014). Mininet as a container based emulator for software defined networks. International Journal of Advanced Research in Computer Science and Software Engineering, 4(12).

Morales, L. V., Murillo, A. F., & Rueda, S. J. (2015, September). Extending the floodlight controller. In 2015 IEEE 14th International Symposium on Network Computing and Applications (pp. 126-133). IEEE.

Li, Y., Guo, X., Pang, X., Peng, B., Li, X., & Zhang, P. (2020, August). Performance analysis of floodlight and ryu SDN controllers under mininet simulator. In 2020 IEEE/CIC International Conference on Communications in China (ICCC Workshops) (pp. 85-90). IEEE.

Madhukrishna Priyadarsini, 1. B. (2017). *Performance Analysis of Software Defined Network.* Bhubaneswar, India.

Mamadou-Tahirou.Bah, T.-M.-T. (2019). Ανάκτηση από https://ieeexplore.ieee.org/document/8685915

Mamushiane, L., Lysko, A., & Dlamini, S. (2018, April). A comparative evaluation of the performance of popular SDN controllers. In 2018 Wireless Days (WD) (pp. 54-59). IEEE.

Paliwal, M., Shrimankar, D., & Tembhurne, O. (2018). Controllers in SDN: A review report. IEEE access, 6, 36256-36270. Md. Tariqul Islam, N. I. (2020). *Node to Node*

Performance Evaluation through RYU SDN Controller. Tangail, Bangladesh: © Springer.

Islam, M., Islam, N., & Refat, M. (2020). Node to node performance evaluation through RYU SDN controller. Wireless Personal Communications, 112(1), 555-570.

Jarschel, M., Metter, C., Zinner, T., Gebert, S., & Tran-Gia, P. (2014, July). OFCProbe: A platform-independent tool for OpenFlow controller analysis. In 2014 IEEE Fifth International Conference on Communications and Electronics (ICCE) (pp. 182-187). IEEE.

Mininet (2022). An Instant Virtual Network on your Laptop (or other PC). Ανακτήθηκε 26 Ιουλίου 2022, από https://mininet.org/

Al-Fares, M., Loukissas, A., & Vahdat, A. (2008). A scalable, commodity data center network architecture. ACM SIGCOMM computer communication review, 38(4), 63-74.

Sameer, M., & Goswami, B. (2018). Experimenting with ONOS scalability on software defined network. Journal of Advanced Research in Dynamical and Control Systems, 10(14-Special Issue), 1820-1830.

OpenDaylight (2022). Ανάκτηση από opendaylight: https://www.opendaylight.org/#more

OpenFlow (2022) Wikipedia. Ανάκτηση από Wikipedia: https://en.wikipedia.org/wiki/OpenFlow

Opennetworking.(2022) ONOS. Ανάκτηση από opennetworking.org: https://opennetworking.org/onos/

Project Floodlight. (2022). Ανάκτηση από Floodlight Controller: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview

python. Ανάκτηση από python: https://www.python.org/

Jawaharan, R., Mohan, P. M., Das, T., & Gurusamy, M. (2018, July). Empirical evaluation of sdn controllers using mininet/wireshark and comparison with cbench. In 2018 27th international conference on computer communication and networks (icccn) (pp. 1-2). IEEE.

Raju, V. S. (2018). SDN controllers comparison. In Proceedings of science globe international conference.

Saritakumar, N., Srinivasan, A. V., Albert, E., & Rani, S. S. (2019). Performance evaluation of pox controller for software defined networks. Int J Innov Technol Explor Eng (IJITEE), 8.

Rao, S. K. (2014). SDN and its use-cases-NV and NFV. Network, 2, H6.

Rawal, A. (2022). *Section*. Ανάκτηση από https://www.section.io/engineering-education/openflow-sdn/#what-is-the-openflow-protocol

Rogério Leão Santos de Oliveira, A. A. (2014). *Using Mininet for Emulation and Prototyping.* Jales, Brazil : IEEE.

ROWSHANRAD, S. (2016). *PERFORMANCE EVALUATION OF SDN CONTROLLERS: FLOODLIGHT AND OPENDAYLIGHT.* Shiraz, Iran.: IIUM Engineering Journal.

RYU. (2019). *ryu.readthedocs.io*. Ανάκτηση από https://ryu.readthedocs.io/en/latest/getting_started.html

Salman, O., Elhajj, I. H., Kayssi, A., & Chehab, A. (2016, April). SDN controllers: A comparative study. In 2016 18th mediterranean electrotechnical conference (MELECON) (pp. 1-6). IEEE.

Shalimov, A., Zuikov, D., Zimarina, D., Pashkov, V., & Smeliansky, R. (2013, October). Advanced study of SDN/OpenFlow controllers. In Proceedings of the 9th central & eastern european software engineering conference in russia (pp. 1-6).

Stancu, A. L., Halunga, S., Vulpe, A., Suciu, G., Fratu, O., & Popovici, E. C. (2015, October). A comparison between several Software Defined Networking controllers. In 2015 12th international conference on telecommunication in modern satellite, cable and broadcasting services (TELSIKS) (pp. 223-226). IEEE.

Badotra, S., & Singh, J. (2017). Open Daylight as a Controller for Software Defined Networking. International Journal of Advanced Research in Computer Science, 8(5).

Badotra, S., & Panda, S. N. (2020). Evaluation and comparison of OpenDayLight and open networking operating system in software-defined networking. Cluster Computing, 23(2), 1281-1291.

Tao Han, S. R. (2018). *A comprehensive survey of security threats and their.* Hawthorn, Australia.

Ubuntu 20.04.4 LTS (Focal Fossa) (2022). Ανάκτηση από ubuntu releases: https://releases.ubuntu.com/20.04/

V. Bollapragada, C. M. (2000). *Inside Cisco IOS software.* Cisco Press.

VirtualBox. (χ.χ.). *www.virtualbox.org*. Ανάκτηση από https://www.virtualbox.org/wiki/Downloads

Zhu, L., Karim, M. M., Sharif, K., Li, F., Du, X., & Guizani, M. (2019). SDN controllers: Benchmarking & performance evaluation. arXiv preprint arXiv:1902.04491.

Zhu, L., Karim, M. M., Sharif, K., Xu, C., Li, F., Du, X., & Guizani, M. (2020). SDN controllers: A comprehensive analysis and performance evaluation study. ACM Computing Surveys (CSUR), 53(6), 1-40.

Khattak, Z. K., Awais, M., & Iqbal, A. (2014, December). Performance evaluation of OpenDaylight SDN controller. In 2014 20th IEEE international conference on parallel and distributed systems (ICPADS) (pp. 671-676). IEEE.

ONOS. (2020) *wiki.onosproject.org*. Ανάκτηση από https://wiki.onosproject.org/display/ONOS/Developer+Quick+Start

How to install OpenDaylight as a Service on Ubuntu 18.04 LTS (2018). *john.soban.ski*. Ανάκτηση από https://john.soban.ski/how-to-install-opendaylight-as-a-service-on-ubuntu.html

ΠΑΡΑΡΤΗΜΑ Α.

Mininet -βασικές λειτουργίες

Το Mininet δημιουργήθηκε από μια ομάδα καθηγητών στο Πανεπιστήμιο του Στάνφορντ για να χρησιμοποιηθεί ως εργαλείο έρευνας και διδασκαλίας στις τεχνολογίες δικτύων. Σήμερα, το Mininet έχει σχεδιαστεί (Kuldeep K. Sharma, 2014) για την εύκολη δημιουργία εικονικών δικτύων καθορισμένων από λογισμικό που αποτελούνται από ένα ελεγκτή OpenFlow όπου θα αναλύσουμε παρακάτω τι είναι, ένα επίπεδο δίκτυο Ethernet από πολλαπλά OpenFlow enabled Ethernet, switches και πολλαπλούς κεντρικούς υπολογιστές. που συνδέονται σε αυτούς τους switches. Διαθέτει ενσωματωμένες λειτουργίες που υποστηρίζουν τη χρήση διαφορετικών τύπων ελεγκτών και μεταγωγέων. Μπορούμε να δημιουργήσουμε επίσης πολύπλοκα προσαρμοσμένα σενάρια χρησιμοποιώντας το Mininet Python API [5]. Οι SDN switches, οι Hosts, οι ελεγκτές και οι συνδέσεις μπορούν να δημιουργηθούν πληκτρολογώντας εντολές μέσω της διεπαφής γραμμής εντολών του Mininet. Η εντολή Sudo mn είναι μια εντολή που χρησιμοποιείται για τη δημιουργία δύο Hosts με 1 switch και έναν ελεγκτή. Η SDN αρχιτεκτονική αποσυνδέει τις λειτουργίες ελέγχου και προώθησης του δικτύου επιτρέποντας στον έλεγχο του δικτύου να γίνει άμεσα προγραμματιζόμενη και η υποκείμενη υποδομή να αφαιρεθεί για εφαρμογές και υπηρεσίες δικτύου. Το OpenFlow που όπως προαναφέραμε θα το δούμε στην συνέχεια αποτελεί θεμελιώδες στοιχείο για την οικοδόμηση λύσεων SDN.

Το Mininet μπορεί να προσομοιώσει δίκτυα SDN και να εκτελέσει έναν ελεγκτή για πειράματα. Το Mininet από προεπιλογή περιλαμβάνει τον ελεγκτή POX και το Open vSwitch. Μπορούμε να εγκαταστήσουμε τον ελεγκτή της επιλογής μας χρησιμοποιώντας την εντολή που δίνεται παρακάτω.

- sudo apt-get install POX/FloodLight

Το Mininet παρέχει έναν μεγάλο αριθμό εντολών (Mininet.org, n.d.) . Ορισμένες χρήσιμες εντολές του Mininet παρατίθενται παρακάτω:

- sudo mn

Για να τρέξουμε το Mininet ως root πρέπει να χρησιμοποιήσουμε την εντολή sudo για να τρέξουμε το mininet. Αυτή η εντολή εκκινεί την ελάχιστη τοπολογία και εισέρχεται στη διεπαφή γραμμής εντολών. Πρόκειται για την προεπιλεγμένη τοπολογία και περιλαμβάνει τον μεταγωγέα OpenFlow ο οποίος είναι συνδεδεμένος με δύο hosts και τον ελεγκτή OpenFlow.

- sudo mn -h

Αυτή η εντολή χρησιμοποιείται για να δείτε το μενού βοήθειας που είναι διαθέσιμο στο Mininet.

- mininet> nodes

Αυτή η εντολή εμφανίζει τους κόμβους που είναι διαθέσιμοι στο τρέχον δίκτυο και οι οποίοι είναι διαθέσιμοι για την προεπιλεγμένη ελάχιστη τοπολογία mininet.

- mininet> dump

Αυτή η εντολή εμφανίζει τις πληροφορίες απόρριψης για όλους τους κόμβους που είναι διαθέσιμοι στο τρέχον δίκτυο mininet.

- mininet> h1 ping h2

Αυτή η εντολή ελέγχει τη συνδεσιμότητα μεταξύ των κεντρικών υπολογιστών h1 και h2. Αυτή η εντολή θα συνεχίσει να ελέγχει τη συνδεσιμότητα μεταξύ των hosts μέχρι να σταματήσουμε την εντολή.

- mininet> h1 ping -c1 h2

Αυτή η εντολή ελέγχει τη σύνδεση μεταξύ των κεντρικών υπολογιστών h1 και h2 για ένα πακέτο.

- sudo mn -c

Αυτή η εντολή χρησιμοποιείται για την εκκαθάριση του mininet ή της εντολής που χρησιμοποιήθηκε προηγουμένως.

Το Mininet περιέχει πέντε προεπιλεγμένες τοπολογίες [19], όπως minimal, single, reversed, linear και tree. Ο ελεγκτής δικτύου συμβολίζεται με C0, οι switches συμβολίζονται με S1 έως Sn και οι κεντρικοί υπολογιστές συμβολίζονται με H1 έως Hn στα ακόλουθα σχήματα που χρησιμοποιούνται για την αναπαράσταση αυτών των τοπολογιών. Για να παρακολουθήσουμε τις αλληλεπιδράσεις μεταξύ του ελεγκτή και οποιουδήποτε μεταγωγέα στο δίκτυο, ξεκινάμε το Wireshark [20] και καταγράφουμε την κυκλοφορία στο Mininet. Το Wireshark πρέπει να εκκινηθεί με δικαιώματα superuser έτσι στο τερματικό του Mininet VM. Ρυθμίζουμε το Wireshark να καταγράφει πακέτο στη διεπαφή loopback του Mininet VM.

Τοπολογίες στο Mininet

Οι τοπολογίες του mininet χωρίζονται σε δύο κατηγορίες:

Α)Προεπιλεγμένες τοπολογίες (default)

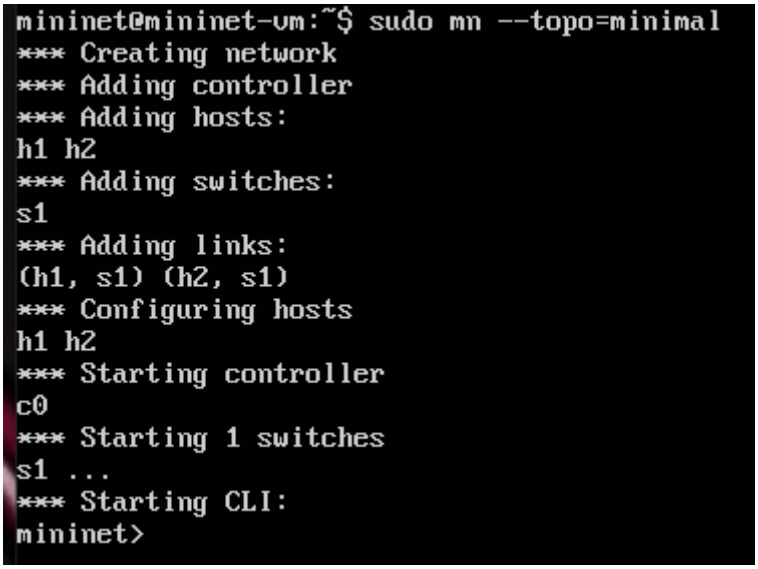
Β) Προσαρμοσμένες τοπολογίες (custom)

Το mininet περιέχει πέντε προεπιλεγμένες τοπολογίες, αυτές είναι σε αγγλικούς όρους οι:

- minimal
- single
- reversed
- linear
- tree

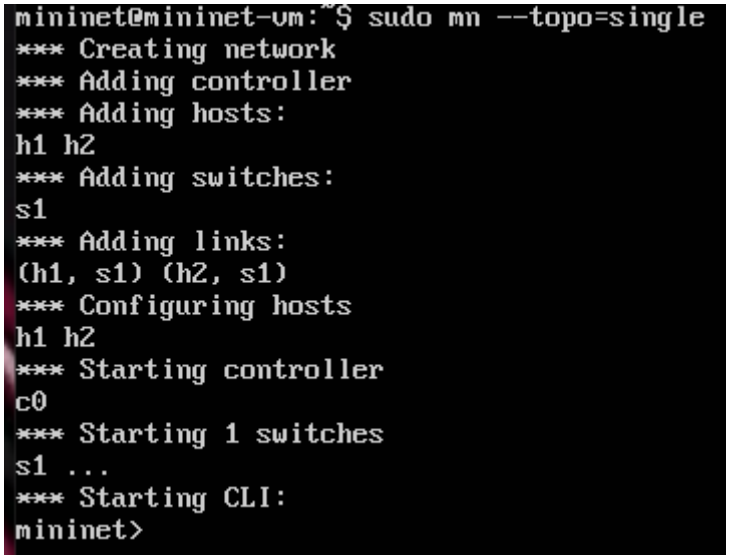
Ο προεπιλεγμένος ελεγκτής δικτύου συμβολίζεται ως ‘’C0’’, οι switches συμβολίζονται με S1 έως Sn και οι κεντρικοί υπολογιστές συμβολίζονται με H1 έως Hn.Παρακάτω μπορούμε να δούμε στο περιβάλλον του mininet πως δημιουργούμε τις προεπιλεγμένες τοπολογίες.

1. Minimal: Περιέχει 1 ελεγκτή, 1 μεταγωγέα Open-Flow και δύο κεντρικούς υπολογιστές με τη δημιουργία σύνδεσης μεταξύ του μεταγωγέα και των κεντρικών υπολογιστών.



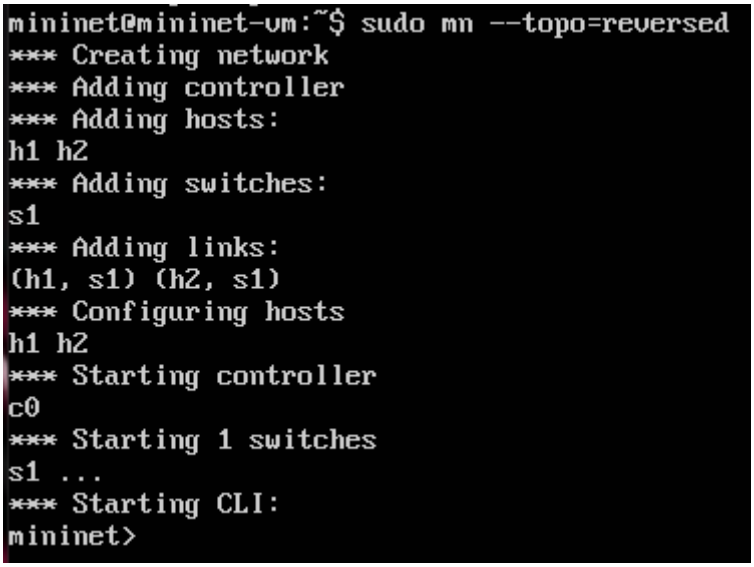
Εικόνα 0-1 Δημιουργία τοπολογίας minimal μέσω της γραμμής εντολών του mininet

2. Single: Περιέχει 1 διακόπτη και έχει n αριθμό κεντρικών υπολογιστών.



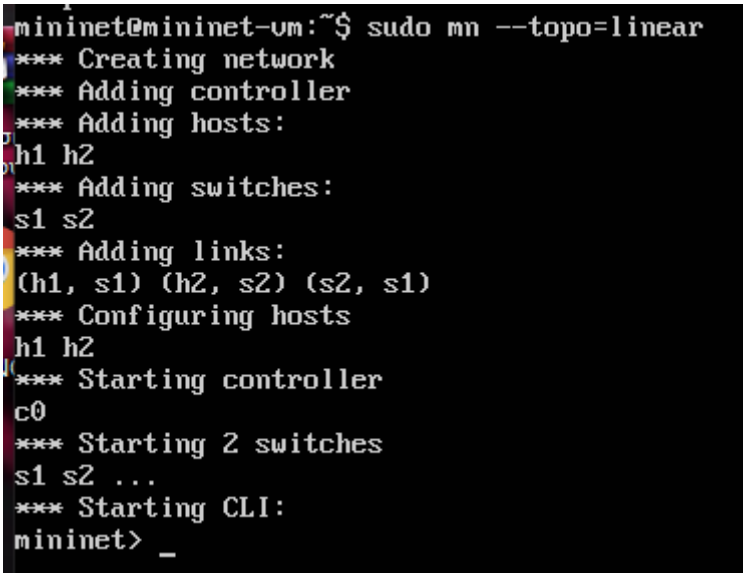
Εικόνα 0-2Δημιουργία τοπολογίας single μέσω της γραμμής εντολών του mininet.

3. Reversed: Είναι παρόμοια με την single τοπολογία αλλά έχει σύνδεση με αντίστροφη σειρά.



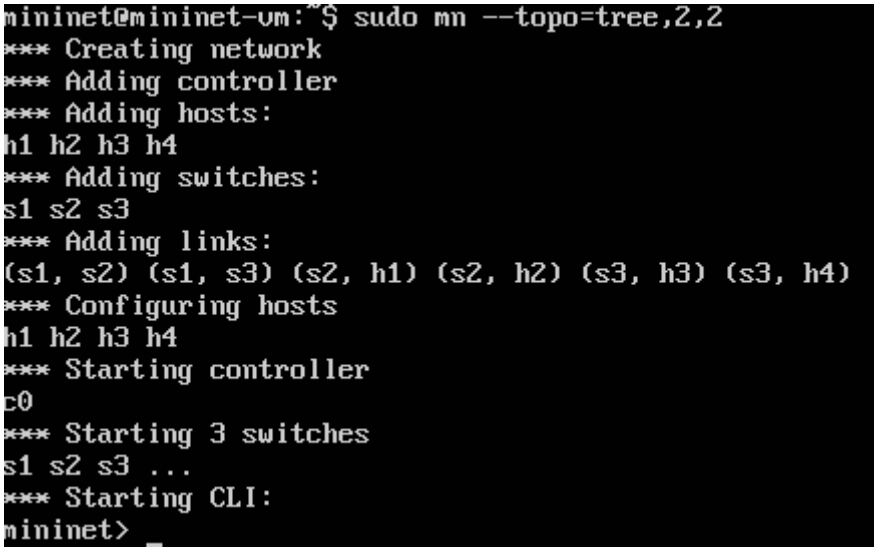
Εικόνα 0-3 Δημιουργία τοπολογίας reversed μέσω της γραμμής εντολών του mininet

4. Linear: Περιέχει n-αριθμούς switches με n-αριθμούς κεντρικούς υπολογιστές συνδεδεμένους με τους switches με γραμμική σειρά.



Εικόνα 0-4 Δημιουργία τοπολογίας linear μέσω της γραμμής εντολών του mininet.

5. Tree: Περιέχει διακόπτες n-επιπέδου και οι κεντρικοί υπολογιστές συνδέονται με διακόπτες χαμηλότερου επιπέδου.



Εικόνα 0-5 Δημιουργία τοπολογίας tree μέσω της γραμμής εντολών του mininet.

Οι προσαρμοσμένες τοπολογίες (Custom) στο Mininet μπορούν να δημιουργηθούν με τη χρήση κώδικα Python. Αυτές οι τοπολογίες εκτελούνται στο Mininet χρησιμοποιώντας εντολές. Το API της Python (python, n.d.) χρησιμοποιεί τις δικές του κλάσεις, μεθόδους, συναρτήσεις και μεταβλητές για τη δημιουργία αυτών των τοπολογιών.