ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
& ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Μελέτη Δικτύωσης Οριζόμενης από Λογισμικό (SDN) με Χρήση του Εξομοιωτή Mininet

Τσίμπρης Ηλίας-Χαράλαμπος ΑΜ: 877

e-mail: ilias_tsi@hotmail.com

Επιβλέπων καθηγητής: Λιαροκάπης Δημήτριος

Πανεπιστήμιο Ιωαννίνων

Σεπτέμβριος 2025, Άρτα

ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Study of Software-Defined Networking (SDN) Using the Mininet Emulator

Τσίμπρης Ηλίας-Χαράλαμπος AM: 877

e-mail: ilias_tsi@hotmail.com

Επιβλέπων καθηγητής: Λιαροκάπης Δημήτριος

Πανεπιστήμιο Ιωαννίνων

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή 14/10/2025

Σπυριδούλα Μαργαρίτη	Ελευθέριος Στεργίου	Δημήτριος Λιαροκάπης

Σεπτέμβριος 2025, Άρτα

ΕΥΧΑΡΙΣΤΙΕΣ

Με την περάτωση της εργασίας αυτής θα ήθελα να ευχαριστήσω ιδιαίτερα τον επιβλέποντα καθηγητή κύριο Λιαροκάπη Δημήτριο για την ευκαιρία που μου έδωσε να ασχοληθώ με τον ιδιαίτερα ενδιαφέρον χώρο της σύγχρονης τεχνολογίας δικτύων της τρέχουσας δεκαετίας και με τον οποίο συνεργάστηκα καθ' όλη τη διάρκεια της εκπόνησης της εργασίας και του οποίου οι συμβουλές και η βοήθεια υπήρξαν πραγματικά ανεκτίμητες. Τέλος, νιώθω την ανάγκη να ευχαριστήσω την οικογένειά μου, και τους φίλους μου για την υποστήριξή τους σε όλη τη διάρκεια της φοιτητικής μου διαδρομής.

ΠΕΡΙΛΗΨΗ

Η Δικτύωση Οριζόμενη από Λογισμικό (Software Defined Networking - SDN) αποτελεί μία από τις πιο σύγχρονες τεχνολογίες δικτύων, προσφέροντας υψηλό επίπεδο προγραμματισμού και ευελιξίας μέσω του διαχωρισμού του επιπέδου ελέγχου από το επίπεδο δεδομένων. Η αυξανόμενη χρήση τεχνολογιών όπως τα μεγάλα δεδομένα (Big Data) και τα κέντρα δεδομένων δημιουργεί νέες προκλήσεις για τις παραδοσιακές αρχιτεκτονικές δικτύων, ενισχύοντας την ανάγκη για πιο έξυπνες και ασφαλείς λύσεις.

Η παρούσα εργασία παρουσιάζει συνοπτικά την αρχιτεκτονική του SDN, τα κύρια πρωτόκολλα που το υποστηρίζουν, καθώς και τα ζητήματα ασφάλειας που προκύπτουν. Ιδιαίτερη έμφαση δίνεται στη χρήση του εξομοιωτή Mininet για τη δημιουργία και αξιολόγηση τοπολογιών SDN σε εικονικό περιβάλλον. Πραγματοποιήθηκε εγκατάσταση και ρύθμιση του Mininet σε εικονική μηχανή (VM), δημιουργήθηκαν τοπολογίες μέσω του εργαλείου Miniedit και υλοποιήθηκαν δοκιμές εύρους ζώνης, καθυστέρησης και Jitter με χρήση του iperf.

Επιπλέον, προσομοιώθηκε επίθεση ICMP Flooding (DDoS) με στόχο την αξιολόγηση της συμπεριφοράς του δικτύου υπό κακόβουλη επίθεση στο δίκτυο. Τα αποτελέσματα έδειξαν αυξημένη καθυστέρηση και υποβάθμιση της απόδοσης, αναδεικνύοντας την ανάγκη για ενσωμάτωση μηχανισμών ανίχνευσης και μετριασμού επιθέσεων.

Η εργασία καταλήγει στο ότι ο Mininet μπορεί να αποτελέσει ένα ιδιαίτερα χρήσιμο εργαλείο για εκπαιδευτικούς και ερευνητικούς σκοπούς, επιτρέποντας την κατανόηση της λειτουργίας του SDN, την ανάλυση απειλών και την ανάπτυξη μελλοντικών σεναρίων ασφάλειας και καινοτομίας στα δίκτυα.

Λέξεις-κλειδιά: SDN, Mininet, OpenFlow, DDoS, αρχιτεκτονική δικτύου

Abstract

Software Defined Networking (SDN) has emerged as a modern paradigm that enables network programmability and flexibility by separating the control plane from the data plane. The increasing adoption of technologies such as Big Data and data centers poses new challenges for legacy network architectures, highlighting the need for more intelligent and secure solutions.

This thesis provides an overview of SDN architecture, its main protocols, and related security issues. A practical focus is given to the use of the Mininet emulator for building and testing SDN topologies in a virtualized environment. Mininet was installed on a VM, and topologies were created using the Miniedit tool, followed by performance tests measuring bandwidth, latency, and jitter through iperf.

Furthermore, an ICMP Flooding (DDoS) attack was simulated to evaluate the network's resilience under malicious traffic. Results indicated increased delays and performance degradation, emphasizing the importance of integrating detection and mitigation mechanisms.

Overall, this thesis demonstrates that Mininet is a valuable tool for both educational and research purposes, providing insights into SDN operation, security threats, and future experimental scenarios.

Keywords: SDN, Mininet, OpenFlow, DDoS, network architecture

Περιεχόμενα

Κατάλογος Σχημάτων & Εικόνων

Κατάλογος Πινάκων

ΚΕΦΑΛΑΙΟ 1ο

ΕΙΣΑΓΩΓΗ

Στα συμβατικά δίκτυα οι συσκευές (δρομολογητές, μεταγωγείς κ.λπ.) είναι στενά δεμένες με το υλικό (ASICs) και διαμορφώνονται με συγκεκριμένες εντολές ανά κατασκευαστή. Η διαχείριση γίνεται κατά κανόνα συσκευή-συσκευή (π.χ. μέσω CLI/console ή SSH), κάτι που αυξάνει την πολυπλοκότητα, το κόστος και τον κίνδυνο λαθών. Η σύζευξη ελέγχου και προώθησης πακέτων περιορίζει την ευελιξία, τη δυνατότητα αυτοματοποίησης και την κλιμάκωση.

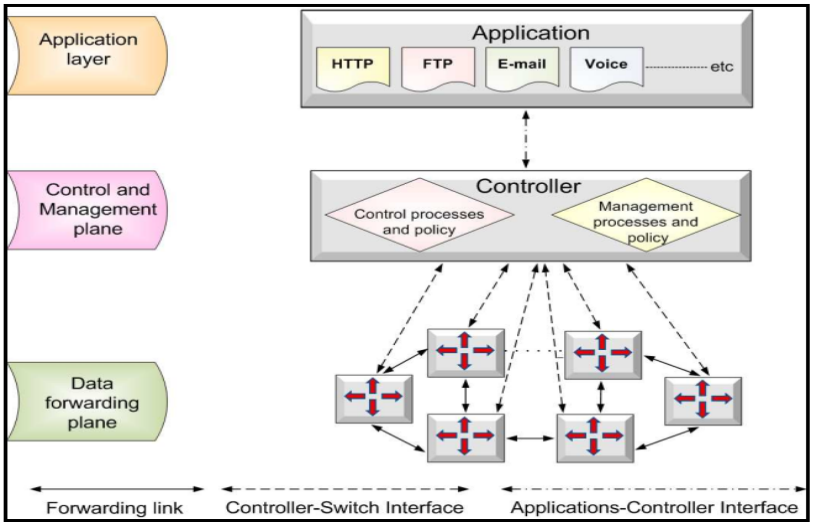
Η Δικτύωση Οριζόμενενη από Λογισμικό (Software Defined Networking - SDN) διαχωρίζει ρητά το επίπεδο ελέγχου από το επίπεδο δεδομένων. Ο «εγκέφαλος» του δικτύου (SDN controller) ενοποιεί τον έλεγχο και εκθέτει προγραμματιζόμενες διεπαφές προς τις εφαρμογές, ενώ οι συσκευές στο data plane επικεντρώνονται στην απλή προώθηση σύμφωνα με κανόνες ροής. Έτσι, επιτυγχάνονται κεντρική ορατότητα, αυτοματοποίηση και ταχεία προσαρμογή σε μεταβαλλόμενες απαιτήσεις [1].

Στο πλαίσιο εκπαιδευτικών/ερευνητικών εργαστηρίων, ο διαχωρισμός αυτός μειώνει δραστικά τον απαιτούμενο χρόνο δοκιμών και επιτρέπει πειράματα σε εικονικά περιβάλλοντα (π.χ. Mininet), χωρίς ακριβό φυσικό εξοπλισμό.

1.1 Αρχιτεκτονική και στοιχεία SDN

Η αρχιτεκτονική SDN οργανώνεται σε τρία επίπεδα: **Εφαρμογών**, **Ελέγχου** και **Δεδομένων** [2], [3]. Το μοντέλο στοχεύει σε δίκτυα πιο ευέλικτα, ασφαλή και επεκτάσιμα, ικανά να εξυπηρετήσουν απαιτήσεις υψηλού εύρους ζώνης (cloud, data centers, IoT).

Οι βασικές διεπαφές είναι οι Northbound APIs (εφαρμογές ↔ controller) και οι Southbound APIs (controller ↔ συσκευές). Συνήθεις υλοποιήσεις περιλαμβάνουν RESTful NBIs και OpenFlow, NETCONF/YANG, OF-Config στο SBIs.



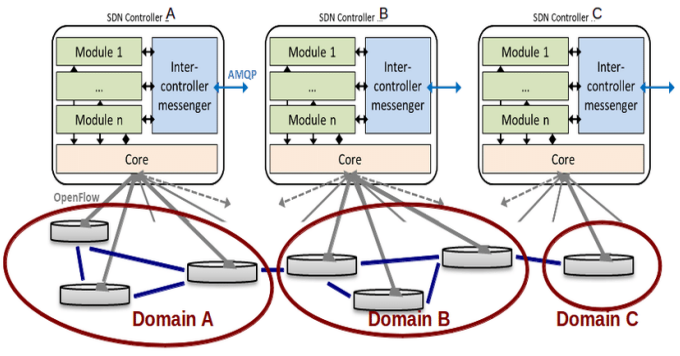
Σχήμα 1: Αρχιτεκτονική των δικτύων καθορισμένων από λογισμικό (SDN)

1.1.2 Επίπεδο εφαρμογής

Περιλαμβάνει εφαρμογές δικτύου για ορατότητα/τηλεμετρία, ασφάλεια, διαχείριση πολιτικών, εικονικοποίηση κ.ά. Μέσω Northbound APIs λαμβάνουν αφηρημένη εικόνα του δικτύου και εκφράζουν επιθυμητές πολιτικές υψηλού επιπέδου. Ο controller μεταφράζει αυτές τις πολιτικές σε κανόνες ροής και ενέργειες στο data plane [5], [6].

1.1.3 Επίπεδο ελέγχου

Ο **SDN controller** συγκεντρώνει την κατάσταση του δικτύου και εφαρμόζει αποφάσεις δρομολόγησης/ασφάλειας. Υπάρχουν κεντρικές και κατανεμημένες αρχιτεκτονικές (π.χ. ONOS, ONIX, DISCO, HyperFlow) που βελτιώνουν διαθεσιμότητα και κλιμάκωση [7]–[11]. Μειονεκτήματα μιας μονολιθικής υλοποίησης είναι το πιθανό single point of failure και τα όρια απόδοσης.



Σχήμα 2: Αρχιτεκτονική SDN με κατανεμημένους ελεγκτές

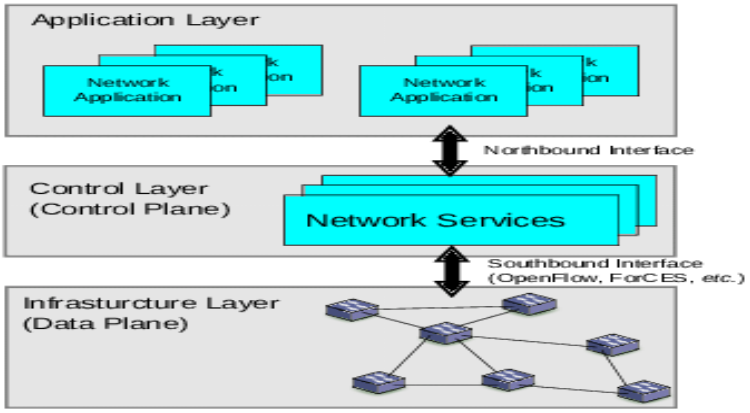
1.1.4 Επίπεδο δεδομένων

Το data plane αποτελείται από **forwarding devices** (π.χ. Open vSwitch, φυσικοί/εικονικοί μεταγωγείς και δρομολογητές). Οι συσκευές διατηρούν **πίνακες ροής**: κανόνες αντιστοίχισης (MAC/IP/θύρες/τύπος EtherType/VLAN), ενέργειες (προώθηση/απόρριψη/αποστολή στον

controller/τροποποίηση πεδίων) και μετρητές στατιστικών [12]. Η επικοινωνία με τον controller γίνεται τυπικά μέσω OpenFlow.

1.2 Northbound Application Program Interface (API)

Τα **Northbound APIs** επιτρέπουν σε εφαρμογές να εκφράζουν στόχους/πολιτικές χωρίς λεπτομέρειες υλοποίησης ενώ· διαφορετικοί controllers (Floodlight, ONOS, OpenDaylight, Ryu κ.ά.) προσφέρουν δικά τους σχήματα και γλώσσες προγραμματισμού, καθώς και RESTful endpoints [4]. Η ετερογένεια ευνοεί την καινοτομία αλλά δυσχεραίνει τη φορητότητα εφαρμογών.



Σχήμα 3: Northbound and Southbound API

1.3 Southbound Application Program Interface (API)

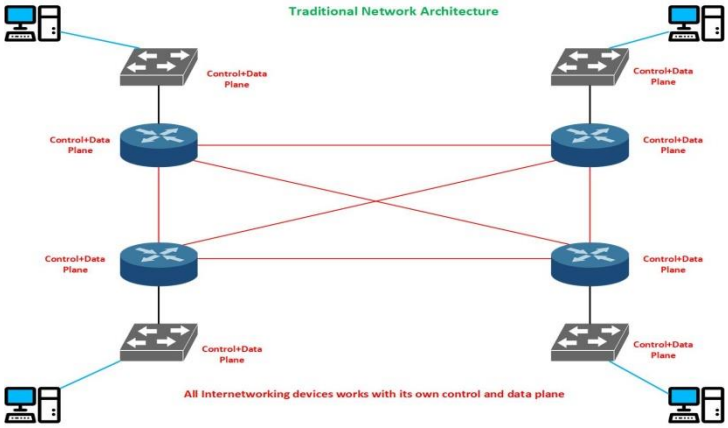
Τα **Southbound APIs** υλοποιούν τη σύζευξη controller–συσκευών. Το **OpenFlow** είναι το επικρατέστερο ενώ συμπληρωματικά χρησιμοποιούνται **NETCONF/YANG**, **OF-Config**, **OVSDB**, **SNMP**, ανάλογα με την ανάγκη (ρύθμιση, τηλεμετρία, διαχείριση). Ο controller λαμβάνει γεγονότα (π.χ. *packet-in*, αλλαγές θύρας/συνδέσμου), ζητά στατιστικά ροών και εγκαθιστά/αφαιρεί κανόνες [11].

Κριτική επισήμανση: Η εξάρτηση από ένα μόνο SB πρωτόκολλο μπορεί να δημιουργήσει «δέσμευση» σε πλατφόρμες. Συνδυαστικές προσεγγίσεις (OpenFlow για ροές, NETCONF/YANG για ρυθμίσεις) μειώνουν τον κίνδυνο.

1.4 Σύγκριση συμβατικού δικτύου και δικτύου SDN

Η συμβατική αρχιτεκτονική δικτύου βασίζεται στο μοντέλο TCP/IP και στηρίζεται αποκλειστικά σε εξειδικευμένο υλικό (ASICs) και ιδιόκτητο λογισμικό, κάτι που περιορίζει σημαντικά τον βαθμό προγραμματισμού και αυτοματοποίησης. Οι διαχειριστές δικτύου καλούνται να ρυθμίσουν κάθε συσκευή ξεχωριστά, μέσω κονσόλας ή απομακρυσμένης πρόσβασης, γεγονός που αυξάνει την πολυπλοκότητα και τον κίνδυνο λαθών. Αντίθετα, το SDN διαχωρίζει το επίπεδο ελέγχου από το επίπεδο δεδομένων, προσφέροντας κεντρική διαχείριση και δυναμική προσαρμοστικότητα.

Στο παρακάτω σχήμα παρουσιάζεται ενδεικτικά η παραδοσιακή αρχιτεκτονική δικτύου:



Σχήμα 4: Παραδοσιακή αρχιτεκτονική δικτύου

Η βασική διαφοροποίηση συνοψίζεται στον παρακάτω συγκριτικό πίνακα, όπου φαίνονται οι δυνατότητες κεντρικής διαχείρισης, ασφάλειας, κόστους και επεκτασιμότητας:

Πίνακας 1: Σύγκριση SDN και παραδοσιακών δικτύων βάσει χαρακτηριστικών

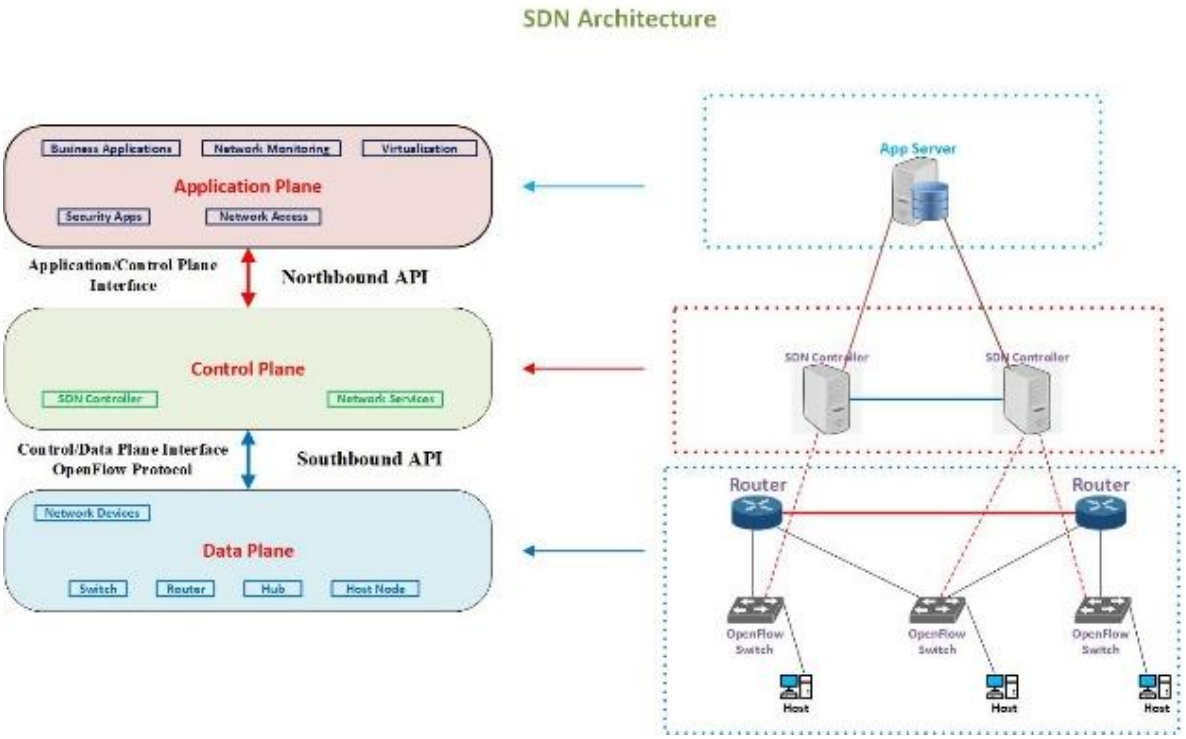
SDN (Software Defined Networking) Infrastructure	**Δίκτυο παλαιού τύπου**
Κεντρική διαχείριση και έλεγχος	Ξεχωριστή διαχείριση για κάθε συσκευή δικτύου
Μεγαλύτερη ασφάλεια	Λιγότερα χαρακτηριστικά ασφαλείας
Χαμηλότερο κόστος λειτουργίας και διαχείρισης	Υψηλότερο κόστος λειτουργίας & διαχείρισης
Λιγότερο επένδυση σε υλικό	Περισσότερες δαπάνες για hardware
Εύκολη επεκτασιμότητα	Σύνθετη επεκτασιμότητα
Ρύθμιση μέσο ανοικτού λογισμικού	Συσκευές που βασίζονται αποκλειστικά στο hardware του κατασκευαστή
Ασφάλεια μέσο κεντρικού ελεγκτή	Κατανεμημένη ασφάλεια σε επίπεδο συσκευής

Ολική προβολή του δικτύου	Δεν παρέχεται ορατότητα ολόκληρου του δικτύου
ΥποστήριξηContent Delivery Network(CDN)	Μη ΥποστήριξηCDN
Υποστήριξη Cloud	Μη υποστήριξη cloud
Καλύτερη υποστήριξη στην ροή δεδομένων (dataflow)	Λιγότερες επιλογές βελτιστοποίησης ροής δεδομένων
Ακριβής, σταθερή και προσαρμοστική ρύθμιση δικτύου	Χρονοβόρα ρύθμιση παραμέτρων επιρρεπής σε σφάλματα
Λιγότερες εξειδικευμένες γνώσεις για την κάθε συσκευή δικτύου που ορίζει ο εκάστοτε κατασκευαστής λόγο ύπαρξης του κεντρικού ελεγκτή	Τα περιβάλλοντα πολλαπλών κατασκευαστών απαιτούν υψηλό επίπεδο τεχνογνωσίας
Παροχή τμηματοποιημένης αρχιτεκτονικής	Περίπλοκη τμηματοποίηση δικτύου
Καινοτομία υπηρεσιών	Δεν έχουν περιθώρια εξέλιξης δικτύου
Διαχείριση από όλο το δίκτυο	Διαχείριση από συγκεκριμένα τμήματα του δικτύου
Κεντρικός ελέγχου δικτύου	Κατανεμημένος έλεγχος
Χρήση του API	Λειτουργία με χρήση πρωτοκόλλων

Όπως παρατηρείται, το SDN προσφέρει σημαντικά πλεονεκτήματα ως προς την προγραμματισιμότητα, την ορατότητα της τοπολογίας και τη μείωση του κόστους. Τα πλεονεκτήματα αυτά αποτυπώνονται και σε επίπεδο αρχιτεκτονικής, όπου ο έλεγχος

συγκεντρώνεται στον SDN Controller και οι συσκευές δεδομένων λειτουργούν ως «χαζοί μεταγωγείς» εκτελώντας μόνο κανόνες προώθησης.

Η φιλοσοφία αυτή οπτικοποιείται στο παρακάτω σχήμα:



Σχήμα 5: Γραφική απεικόνιση της πολυεπίπεδης αρχιτεκτονικής SDN

1.5 Πλεονεκτήματα και χαρακτηριστικά του SDN

Η αρχιτεκτονική SDN καθιστά το δίκτυο πιο ευέλικτο, πιο οικονομικό και πιο ανθεκτικό σε σφάλματα. Κεντρικά χαρακτηριστικά είναι:

- **Ενισχυμένη δυνατότητα προγραμματισμού**: Οι συσκευές μπορούν να ρυθμιστούν μέσω ανοικτών APIs και OpenFlow, προσαρμοζόμενες στις ανάγκες κάθε εφαρμογής.
- **Δυναμική διαμόρφωση**: Οι αλλαγές στην τοπολογία ή στις πολιτικές γίνονται άμεσα μέσω του κεντρικού ελεγκτή.
- **Βελτιωμένη απόδοση/εμπειρία χρήστη**: Χάρη στον κεντρικό έλεγχο και τη συνεχή παρακολούθηση, το δίκτυο προσαρμόζεται δυναμικά στις απαιτήσεις (π.χ. βίντεο streaming με σταθερή ποιότητα).
- **Υψηλή ευελιξία**: Νέες συσκευές μπορούν να προστεθούν και να ρυθμιστούν χωρίς φυσική παρουσία διαχειριστή.
- **Εύκολη ανάπτυξη**: Οι εικονικές συσκευές του επιπέδου δεδομένων απλοποιούν την εγκατάσταση και τη συντήρηση.
- **Βελτιωμένη επεκτασιμότητα**: Η απομόνωση control/data plane επιτρέπει ανεξάρτητη εξέλιξη και ευκολότερη κλιμάκωση.
- **Κεντρική διαχείριση**: Ο ελεγκτής SDN παρέχει συνολική εικόνα του δικτύου, μειώνοντας λάθη και χρόνο διαχείρισης.

- **Μείωση κόστους**: Δεν απαιτείται ακριβό ιδιόκτητο hardware, ενώ περιορίζονται οι λειτουργικές δαπάνες.

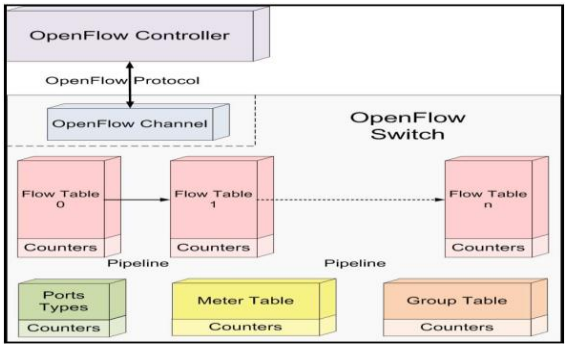
1.6 Πρωτόκολλα sdn

Η επικοινωνία στο SDN βασίζεται σε πρωτόκολλα που διασφαλίζουν τον συντονισμό μεταξύ ελεγκτή και επιπέδου δεδομένων. Το πιο διαδεδομένο είναι το **OpenFlow**, που εισήχθη το 2008 από το Stanford.

Ο ελεγκτής και οι μεταγωγείς επικοινωνούν μέσω καναλιού OpenFlow, ανταλλάσσοντας μηνύματα για:

- νέες ροές (Packet-in),
- στατιστικά πακέτων,
- αλλαγές κατάστασης θυρών.

Στο παρακάτω σχήμα απεικονίζεται η βασική λειτουργία του OpenFlow:



Εικόνα 6: Αρχιτεκτονική μπλοκ διακοπτών OpenFlow

Κάθε καταχώρηση ροής στον πίνακα OpenFlow περιλαμβάνει πεδία αντιστοίχισης, οδηγίες και μετρητές. Η επιλογή δράσης (forward, drop, send to controller) βασίζεται στην προτεραιότητα.

Σχήμα 7: Πεδία καταχώρησης πίνακα OpenFlow

Η εξέλιξη του OpenFlow περιγράφεται συνοπτικά στον επόμενο πίνακα, όπου φαίνονται οι εκδόσεις και οι ιδιότητες που εισήχθησαν σταδιακά:

Πίνακας 2: OpenFlow και ιδιότητες σύμφωνα με τις εκδόσεις OpenFlow

OpenFlow Version /Properties	**1.0**	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**
Publication date	Dec, 2009	Feb, 2011	Dec, 2011	Jun, 2012	Oct, 2013	Dec,2014
Widely deployed	Yes	No	No	No	No	No
Flow table	Single	Multiple	Multiple	Multiple	Multiple	Multiple

Group table	No	Yes	Yes	Yes	Yes	Yes
Meter table	No	No	Yes	Yes	Yes	
VLAN and MPLS Tag	No	Yes	Yes	Yes	Yes	Yes
Controller connection frailer	Emergency	Standalone	Standalone	Standalone	Standalone	Standalone
IPv6 support	No	No	Yes	Yes	Yes	Yes
Multiple controller	No	No	Yes	Yes	Yes	Yes
Eviction /Vacancy/Synchronization	No	No	No	No	Yes	Yes
Optical ports	No	No	No	No	Yes	Yes

1.7 Mininet (εξομοιωτής SDN) Υποστηρικτικά βοηθητικά προγράμματα

Η προσομοίωση της αρχιτεκτονικής SDN και η εκτέλεση τοπολογίας δικτύου για πειράματα δικτύου απαιτούσε πρόσθετα βοηθητικά εργαλεία τα οποία άλλοτε είναι ενσωματωμένα και διαθέσιμα στο περιβάλλον του λειτουργικού συστήματος Linux και άλλοτε απαιτείται η λήψη και η εγκατάστασή τους στο περιβάλλον εξομοίωσης SDN για υποστήριξη. Παρακάτω παρουσιάζονται ορισμένα χρήσιμα βοηθητικά προγράμματα υποστήριξης που βοηθούν στην προσομοίωση και την ανάλυση δικτύου στον εξομοιωτή Mininet SDN.

Iperf

Το Iperf είναι ένα εργαλείο δημιουργίας κίνησης που χρησιμοποιείται για τη μέτρηση του μέγιστου εφικτού εύρους ζώνης σε δίκτυα που βασίζονται σε IP και βοηθά επίσης στη δημιουργία προσαρμοσμένων πακέτων δεδομένων χρησιμοποιώντας διάφορες παραμέτρους, όπως ο χρονισμός, η προσωρινή μνήμη και τα πρωτόκολλα μεταφοράς (TCP, UDP, ICMP, IPv4, IPv6, ICMP). Το Iperf χρησιμοποιείται επίσης για τη δημιουργία μαζικής κυκλοφορίας δεδομένων προς τους κόμβους προορισμού από τους κόμβους προέλευσης και είναι επίσης ένα αποτελεσματικό εργαλείο για την προσομοίωση διαφόρων τύπων επιθέσεων DoS και DDoS πλημμύρας, όπως οι επιθέσεις πλημμύρας ICMP και UDP.

Τα εργαλεία Iperf χρησιμοποιούνται επίσης για τη μέτρηση της απόδοσης του δικτύου και κάθε δοκιμή που εκτελείται με τη χρήση του Iperf μπορεί να παρέχει πληροφορίες σχετικά με το εύρος ζώνης, την απώλεια πακέτων, την διακύμανση (jitter), την καθυστέρηση πακέτων και τον χρόνο διαδρομής (RTT). Για τα πρωτόκολλα TCP και SCTP, το Iperf μπορεί να βοηθήσει στην παροχή μέγιστης χρήσης εύρους ζώνης, μέγιστου μεγέθους μονάδας μετάδοσης (MTU) και μεγέθους παραθύρων TCP, ενώ για το UDP υποστηρίζει τη μέτρηση των Jitters πακέτων για πακέτα φωνής, εύρους ζώνης, απώλειας πακέτων και Delay Jitter [39].

Cbench

Το Cbench, επίσης γνωστό ως benchmark ελεγκτής, είναι ένα βοηθητικό πρόγραμμα λογισμικού που χρησιμοποιείται για τη δοκιμή ελεγκτών SDN με δυνατότητα OpenFlow, δημιουργώντας συμβάντα εισερχόμενων πακέτων (Packet-in) για νέες ροές κυκλοφορίας. Το Cbench χρησιμοποιείται για την προσομοίωση διαφόρων εικονικών μεταγωγέων με δυνατότητα Open Flow (μια δέσμη Open Virtual Switches) που συνδέονται με έναν κεντρικό ελεγκτή SDN και για την αποστολή μηνυμάτων packet-in και την παρατήρηση των μοντέλων ροής που θα προωθηθούν προς τα κάτω.

Οποιαδήποτε αλλαγή πραγματοποιείται στο δίκτυο και ο αντίκτυπός της στην απόδοση του δικτύου μπορεί να αναλυθεί και να μετρηθεί χρησιμοποιώντας το βοηθητικό πρόγραμμα λογισμικού Cbench. Η βασική διαφορά μεταξύ του Iperf και του Cbench είναι ότι το Iperf είναι λογισμικό παραγωγής κίνησης πακέτων που βοηθά στην ανάλυση της απόδοσης και τη μέτρηση της κίνησης του δικτύου και το Cbench είναι ένα βοηθητικό πρόγραμμα λογισμικού που βοηθά στην παρακολούθηση της απόδοσης της απόδοσης του ελεγκτή SDN που βασίζεται στο Open Flow [40].

Hping3

Το Hping3 είναι ένας συναρμολογητής και αναλυτής πακέτων που λειτουργεί ως βοηθητικό πρόγραμμα TCP/IP με προσανατολισμό στη γραμμή εντολών. Η διεπαφή γραμμής εντολών του Hping3 σχεδιάστηκε με βάση την έμπνευση του ping(8) Unix command line utility και το Hping3 είναι επίσης ικανό να στέλνει πακέτα ICMP echo request έχει επίσης την ικανότητα να στέλνει πακέτα με βάση τα πρωτόκολλα TCP, UDP, ICMP και RAW IP [41]. Το Hping3 υποστηρίζει επίσης τη λειτουργία trace route, την αποστολή αρχείων μεταξύ συνδεδεμένων κόμβων και έναν κατάλογο άλλων χαρακτηριστικών.

Το Hping3 μπορεί επίσης να χρησιμοποιηθεί στην αρχιτεκτονική SDN μέσω εξομοιωτή Mininet για την προσομοίωση επιθέσεων DoS και DDoS, επιθέσεων πλημμύρας ICMP και άλλων προσομοιώσεων επιθέσεων TCP και UDP [42]. Το Hping3 χρησιμοποιείται επίσης στην εξομοίωση δικτύου Mininet SDN για τον υπολογισμό του εύρους ζώνης της σύνδεσης δικτύου με χρήση της κυκλοφορίας TCP και την ανάλυση των διαταραχών με χρήση της κυκλοφορίας UDP για τη φωνητική κυκλοφορία. Το Hping3 είναι πλήρως σεναριοποιήσιμο με τη βοήθεια της γλώσσας εντολών εργαλείων (TCL) και τα πακέτα μπορούν να μεταφερθούν με τη μορφή δυαδικής αναπαράστασης και αναπαράστασης συμβολοσειράς.

Wireshark

Το Wireshark είναι ένας από τους πιο δημοφιλείς αναλυτές πρωτοκόλλων δικτύου στον κόσμο των δικτύων και των επικοινωνιών και της ασφάλειας δικτύων. Το Wireshark ως αναλυτής πακέτων δικτύου και εργαλείο καταγραφής της δικτυακής κίνησης και των πακέτων είναι εξοπλισμένο με πλούσιο και ισχυρό σύνολο χαρακτηριστικών και έχει συμβατότητα με τις περισσότερες πλατφόρμες υπολογιστών, όπως Windows, Linux OS, OS X, Mac OS και UNIX. Οι επαγγελματίες του δικτύου και της ασφάλειας, οι προγραμματιστές λογισμικού, οι ερευνητές επικοινωνίας και πρωτοκόλλων δικτύου, ακόμη και οι εκπαιδευτικοί χρησιμοποιούν συχνά αυτό το πολύτιμο εργαλείο καταγραφής και φιλτραρίσματος πακέτων για την ανάλυση της δικτυακής κίνησης σε δίκτυα υπολογιστών.

Το Wireshark είναι ένα ελεύθερα διαθέσιμο εργαλείο ανοιχτού κώδικα που είναι ενσωματωμένο σε ορισμένες διανομές Linux και συγκεκριμένα στο Kali Linux που

χρησιμοποιείται κυρίως για δοκιμές διείσδυσης και από τους χάκερς για την ανάλυση πακέτων και κίνησης δικτύου. Το Wireshark κυκλοφορεί ως λογισμικό ανοικτού κώδικα υπό την άδεια GNU License και αναπτύχθηκε από ειδικούς σε θέματα δικτύων και επικοινωνιακών πρωτοκόλλων από όλο τον κόσμο. Το Wireshark είναι μια δωρεάν εφαρμογή υπολογιστή packet sniffer που χρησιμοποιείται συχνά ως Ethereal και διευκολύνει επίσης για την ανάλυση της κίνησης δικτύου, την αντιμετώπιση προβλημάτων και την ανάπτυξη εφαρμογών δικτύου και πρωτοκόλλων επικοινωνίας, καθώς και στα ακαδημαϊκά ιδρύματα για την εκτέλεση πειραμάτων δικτύου.

Τον Ιούνιο του 2006, το Ethereal μετονομάστηκε σε Wireshark και πλέον χρησιμοποιείται κυρίως για την παρακολούθηση, τη σύλληψη και την ανάλυση της δικτυακής κίνησης και των πακέτων δεδομένων σε δίκτυα επικοινωνίας δεδομένων. Το Wireshark έχει επίσης μια εξελιγμένη υποστήριξη για διάφορα πρωτόκολλα ασύρματης επικοινωνίας που διευκολύνει τους διαχειριστές δικτύων για την αντιμετώπιση προβλημάτων στα δίκτυα ασύρματης επικοινωνίας [43].

Τα προηγμένα χαρακτηριστικά του εργαλείου καταγραφής και φιλτραρίσματος πακέτων Wireshark περιλαμβάνουν φιλτράρισμα και καταγραφή πακέτων με βάση διευθύνσεις, διάφορα πεδία πρωτοκόλλων, χαρακτηριστικά κίνησης δικτύου, δημιουργία προσαρμοσμένων στηλών για αποτελεσματική ανάλυση της κίνησης δικτύου, εύρεση πηγών καθυστέρησης με χρωματισμό και διάφορα μαλακά φίλτρα πακέτων, καταγραφή πακέτων χωρίς επίβλεψη με αυτόματη διακοπή, φιλτράρισμα πακέτων με βάση λέξεις-κλειδιά χρησιμοποιώντας κανονικές εκφράσεις και μάσκες wildcard, σύγκριση χρηστών, υποδικτύων, καταγραφή αρχείων από την καταγεγραμμένη κίνηση, αναγνώριση πακέτων και σφαλμάτων DNS και HTTP [44]. Το Wireshark υποστηρίζει επίσης την αποθήκευση των αποτελεσμάτων της ανάλυσης της κυκλοφορίας, του φιλτραρίσματος πακέτων και της σύλληψης σε μορφή .csv (MS Excel). Όλα αυτά τα προηγμένα χαρακτηριστικά του Wireshark το κατέστησαν ένα από τα πιο εύχρηστα και διαδεδομένα εργαλεία ανάλυσης κίνησης δικτύου.

1.8 Διακύμανση (Jitter)

Το Jitter είναι μια διακύμανση ή καθυστέρηση στην παράδοση πακέτων δεδομένων μέσω ενός δικτύου, δηλαδή μια καθυστέρηση μεταξύ του χρόνου μετάδοσης και λήψης ενός σήματος. Η καθυστέρηση/διακύμανση/αλλαγή στο χρόνο είναι μια διακοπή στη συνήθη ακολουθία αποστολής πακέτων δεδομένων και μετράται σε χιλιοστά του δευτερολέπτου (ms).

Το Jitter επηρεάζει τις διαδικτυακές δραστηριότητες που βασίζονται σε αμφίδρομη επικοινωνία, όπως κάμερες ασφαλείας IP και κλήσεις συνδιάσκεψης. Όταν αντιμετωπίζετε jitter, η σύνδεσή έχει ασταθείς καθυστερήσεις μεταξύ των πακέτων δεδομένων. Τα καθαρά αποτελέσματα του υψηλού jitter είναι:

- Καθυστέρηση κλήσης
- Ηχώ κλήσης
- Παραμόρφωση ήχου και βίντεο

Ένας πιο αρνητικός αντίκτυπος γίνεται αισθητός όταν τα πακέτα δεδομένων χρειάζονται περισσότερο χρόνο για να φτάσουν στον προορισμό τους. Οι σχετικές διακυμάνσεις προκαλούν την απόρριψη πακέτων φωνής/δεδομένων.

Για ψυχαγωγικούς σκοπούς, αυτό είναι δυσάρεστο. Ωστόσο, είναι αφόρητο σε επαγγελματικό περιβάλλον, όπως οι κλήσεις συνδιάσκεψης. Τα προβλήματα απόκλισης είναι πιο κοινά μεταξύ των τελικών χρηστών WI-FI από ό,τι σε άλλες συνδέσεις δικτύου. Παραδείγματα jitter περιλαμβάνουν σταθερό jitter, παροδικό jitter και βραχυπρόθεσμη διακύμανση καθυστέρησης.

Είναι αποδεκτό το Jitter;

Υπάρχει ένα αποδεκτό επίπεδο jitter, αλλά ποικίλλει σημαντικά. Το αποδεκτό επίπεδο jitter θα πρέπει να είναι κάτω από 30 ms από τεχνική άποψη. Τα προβλήματα εμφανίζονται με το βίντεο και τον ήχο όταν γίνεται εγγραφή με τιμές μεγαλύτερες από 30 ms. Επίσης, η απώλεια πακέτων πρέπει να είναι μικρότερη από 1% και η καθυστέρηση όχι μεγαλύτερη από 150 ms.

ΚΕΦΑΛΑΙΟ 2^{Ο}

ΑΣΦΑΛΕΙΑ ΔΙΚΤΥΟΥ SDN ΚΑΙ ΖΗΤΗΜΑΤΑ

Η υιοθέτηση της αρχιτεκτονικής Software Defined Networking (SDN) προσφέρει νέες δυνατότητες, αλλά ταυτόχρονα εισάγει και σημαντικές προκλήσεις ασφάλειας. Ο διαχωρισμός του επιπέδου ελέγχου από το επίπεδο δεδομένων και η κεντρικοποίηση της διαχείρισης μέσω του SDN Controller δημιουργούν νέα σημεία πιθανής ευπάθειας. Επειδή ο ελεγκτής συγκεντρώνει όλες τις κρίσιμες λειτουργίες, η παραβίασή του μπορεί να έχει σοβαρότερες συνέπειες από ό,τι σε ένα παραδοσιακό δίκτυο [23], [24].

2.1. Κίνδυνοι και απειλές στο SDN

Σε ένα συμβατικό δίκτυο, οι επιθέσεις στοχεύουν συνήθως μεμονωμένες συσκευές (π.χ. routers, switches). Στο SDN, ωστόσο, οι απειλές συγκεντρώνονται κυρίως:

- **Στο επίπεδο ελέγχου**: Αν ο ελεγκτής παραβιαστεί, ο επιτιθέμενος αποκτά έλεγχο σε ολόκληρο το δίκτυο.
- **Στο επίπεδο δεδομένων**: Οι συσκευές (OpenFlow switches) είναι ευάλωτες σε flooding επιθέσεις ή κακόβουλους κανόνες ροής.
- **Στα APIs** (Northbound/Southbound): Λόγω έλλειψης τυποποίησης και χρήσης ανοιχτών διεπαφών, τα APIs μπορεί να αποτελέσουν πύλη εισόδου επιθέσεων [24].

Οι επιθέσεις αυτές περιλαμβάνουν **Denial of Service (DoS/DDoS)**, **παραποίηση ροών**, **man-in-the-middle** και **packet spoofing** [23].

2.2 Αδυναμίες του SDN Controller

Ο SDN Controller είναι ο «εγκέφαλος» του δικτύου. Τα βασικά προβλήματα που έχουν εντοπιστεί είναι:

- **Single Point of Failure**: Η κατάρρευση του ελεγκτή μπορεί να οδηγήσει σε πλήρη διακοπή λειτουργίας [23].
- **Επιθέσεις υπερφόρτωσης (Flooding)**: Η αποστολή τεράστιου όγκου packet-in μπορεί να εξαντλήσει τους πόρους του [25], [26].
- **Κακόβουλες εφαρμογές**: Επειδή πολλές εφαρμογές τρίτων εγκαθίστανται στο επίπεδο εφαρμογής, μπορούν να εισάγουν τρωτά σημεία ή backdoors [24].

Στο παρακάτω σχήμα παρουσιάζεται συνοπτικά το σημείο εστίασης επιθέσεων στην αρχιτεκτονική SDN:

Η κεντρικοποίηση του ελέγχου, αν και λειτουργική, αυξάνει την ανάγκη για αυστηρούς μηχανισμούς ελέγχου πρόσβασης, παρακολούθησης και ανθεκτικότητας. [23]

2.3 Επιθέσεις σε επίπεδο δεδομένων

Οι συσκευές του επιπέδου δεδομένων εκτελούν μόνο εντολές που λαμβάνουν από τον ελεγκτή. Ωστόσο, οι πίνακες ροής μπορούν να «γεμίσουν» με κακόβουλες καταχωρήσεις, οδηγώντας σε:

- υπερφόρτωση μνήμης,
- καθυστερήσεις στη δρομολόγηση,
- αδυναμία εξυπηρέτησης νόμιμης κίνησης.

Ενδεικτική περίπτωση αποτελεί το **Flow Table Overflow Attack**, όπου ένας επιτιθέμενος δημιουργεί χιλιάδες ψεύτικες ροές με σκοπό να καταλάβει όλους τους πόρους [27].

2.4 Απειλές στα APIs και το επίπεδο εφαρμογών

Το **Northbound API** επιτρέπει την ανάπτυξη εφαρμογών (π.χ. load balancing, monitoring). Αν όμως μια εφαρμογή είναι κακόβουλη ή κακώς υλοποιημένη, μπορεί να εκμεταλλευτεί την απουσία τυποποίησης και να αλλοιώσει κρίσιμα δεδομένα [24].

Αντίστοιχα, στο **Southbound API** (OpenFlow, NETCONF, OF-Config), μη ασφαλής επικοινωνία μπορεί να οδηγήσει σε υποκλοπή πακέτων ή παραποίηση κανόνων [23].

Η ασφάλεια των APIs είναι επομένως εξίσου κρίσιμη με την ασφάλεια του ίδιου του ελεγκτή.

2.5 Επιθέσεις DoS και DDoS σε SDN

Μία από τις πιο σοβαρές απειλές είναι οι επιθέσεις άρνησης υπηρεσίας (DoS/DDoS). Σε τέτοιες περιπτώσεις, μεγάλος όγκος ψευδούς κίνησης αναγκάζει το δίκτυο να εξαντλήσει τους πόρους του.

- Στα παραδοσιακά δίκτυα, οι επιθέσεις περιορίζονται σε συγκεκριμένους κόμβους.
- Στο SDN, μια τέτοια επίθεση μπορεί να παραλύσει **ολόκληρο το δίκτυο** επειδή ο ελεγκτής εμπλέκεται σε όλες τις αποφάσεις προώθησης [28], [29], [30].

Η ανάγκη για έγκαιρη ανίχνευση και αντιμετώπιση DDoS είναι λοιπόν θεμελιώδης.

2.6 Μηχανισμοί άμυνας

Η διεθνής βιβλιογραφία προτείνει ποικιλία λύσεων για την ενίσχυση της ασφάλειας στα SDN:

- **Κατανεμημένοι ελεγκτές** (ONOS, HyperFlow): Μειώνουν τον κίνδυνο single point of failure [23].
- **Ενσωματωμένα IDS/IPS**: Ανίχνευση ανωμαλιών σε πραγματικό χρόνο [27].

- **Machine Learning τεχνικές**: Εκπαίδευση μοντέλων για εντοπισμό DDoS patterns [30].
- **Κρυπτογράφηση επικοινωνίας**: TLS μεταξύ ελεγκτή και switches [24].
- **Role-based access control**: Περιορισμός των δικαιωμάτων κάθε εφαρμογής ή χρήστη [23].

Συνοπτικά, η προστασία του SDN δεν επιτυγχάνεται μόνο με την εφαρμογή ενός μέτρου, αλλά απαιτεί **πολυεπίπεδη στρατηγική** που να καλύπτει όλα τα επίπεδα της αρχιτεκτονικής.

2.7 Ορισμός των επιθέσεων DDoS

Οι επιθέσεις κατανεμημένης άρνησης υπηρεσίας (Distributed Denial of Service – DDoS) στοχεύουν την αδιάλειπτη λειτουργία υπηρεσιών και διαδικτυακών τόπων, επιχειρώντας να εξαντλήσουν τους διαθέσιμους πόρους ενός συστήματος ή εφαρμογής. Ο μηχανισμός τους βασίζεται στη μαζική αποστολή ψευδούς ή μη απαραίτητης κίνησης, η οποία οδηγεί σε υποβάθμιση της ποιότητας υπηρεσίας ή ακόμη και σε πλήρη διακοπή λειτουργίας [24]. Οι DDoS επιθέσεις αποτελούν μία από τις συνηθέστερες απειλές στον κυβερνοχώρο, επηρεάζοντας κλάδους όπως τα ηλεκτρονικά παιχνίδια, το ηλεκτρονικό εμπόριο και τις τηλεπικοινωνίες.

2.8 Επιθέσεις DDoS και μηχανισμός δράσης

Κατά τη διάρκεια μιας επίθεσης DDoS, ένας μεγάλος αριθμός μολυσμένων συσκευών (botnet) αποστέλλει μαζικά αιτήσεις ή πακέτα δεδομένων προς έναν στόχο, προκαλώντας συμφόρηση ή υπερφόρτωση των υποδομών του [25]. Εκτός από τη διακοπή παρεχόμενων υπηρεσιών, οι επιθέσεις μπορούν να αξιοποιηθούν και ως μέσο διείσδυσης σε βάσεις δεδομένων ή εφαρμογές, αυξάνοντας τον κίνδυνο διαρροής ευαίσθητων πληροφοριών [26]. Η διάρκεια μιας DDoS επίθεσης μπορεί να κυμαίνεται από λίγα λεπτά έως αρκετές ημέρες, επιφέροντας σοβαρές λειτουργικές και οικονομικές επιπτώσεις.

2.9 Τύποι επιθέσεων DDoS

Οι επιθέσεις DDoS διακρίνονται σε τρεις βασικές κατηγορίες [27]:

1. **Ογκομετρικές επιθέσεις**: Επικεντρώνονται στον κορεσμό του εύρους ζώνης με υψηλό όγκο κυκλοφορίας (π.χ. DNS amplification).
2. **Επιθέσεις πρωτοκόλλου**: Εκμεταλλεύονται αδυναμίες της στοίβας TCP/IP, όπως οι SYN floods, καταναλώνοντας κρίσιμους πόρους του διακομιστή.
3. **Επιθέσεις επιπέδου εφαρμογής**: Στοχεύουν υπηρεσίες εφαρμογών (π.χ. HTTP floods, SQL injection), καθιστώντας τις web εφαρμογές μη διαθέσιμες.

Συχνά, οι επιτιθέμενοι συνδυάζουν διαφορετικές τεχνικές ώστε να αυξήσουν την αποτελεσματικότητα της επίθεσης και τη δυσκολία άμυνας [28].

2.10 Εντοπισμός και αντιμετώπιση επιθέσεων DDoS

Η έγκαιρη αναγνώριση αποτελεί κρίσιμο παράγοντα για τον περιορισμό των επιπτώσεων μιας DDoS επίθεσης. Ενδείξεις περιλαμβάνουν ξαφνική αύξηση κυκλοφορίας από συγκεκριμένες διευθύνσεις IP, σημαντικές καθυστερήσεις απόκρισης ή πλήρη μη διαθεσιμότητα υπηρεσιών [29]. Η αξιοποίηση λογισμικών παρακολούθησης και ανάλυσης κίνησης, σε συνδυασμό με σχέδιο απόκρισης που προβλέπει καθορισμένους ρόλους και διαδικασίες, ενισχύει την ικανότητα γρήγορης και συντονισμένης αντίδρασης.

2.11 Μέτρα αποτροπής επιθέσεων DDoS

Η πρόληψη βασίζεται στην ύπαρξη στρατηγικού σχεδίου άμυνας, το οποίο περιλαμβάνει [24][30]:

- συστηματική αξιολόγηση κενών ασφαλείας και πιθανών απειλών,
- τακτική ενημέρωση και συντήρηση λογισμικού,
- προετοιμασία και εκπαίδευση της ομάδας απόκρισης,
- χρήση τεχνικών φιλτραρίσματος και ισοκατανομής φόρτου.

Η ετοιμότητα ενός οργανισμού μειώνει τον χρόνο αντίδρασης και τις επιπτώσεις μιας επίθεσης.

2.12 Στρατηγικές προστασίας από DDoS

Η αποτελεσματική προστασία περιλαμβάνει τόσο τεχνολογικά μέτρα όσο και οργανωτικές πρακτικές [25]:

- περιοδική ανάλυση κινδύνου,
- δημιουργία ειδικής ομάδας αντιμετώπισης περιστατικών,
- ενσωμάτωση εργαλείων ανίχνευσης/αποτροπής σε κρίσιμες υπηρεσίες,
- ασκήσεις εξοικείωσης και αξιολόγηση στρατηγικών άμυνας.

Στην πράξη, η προστασία μπορεί να επιτευχθεί μέσω εξειδικευμένων υπηρεσιών cloud, λογισμικών ανίχνευσης απειλών και υβριδικών λύσεων ασφαλείας.

ΚΕΦΑΛΑΙΟ 3ο

ΟΔΗΓΟΣ MININET ΚΑΙ ΣΧΕΔΙΑΣΜΟΣ ΔΙΚΤΥΟΥ SDN

3.1 ΕΙΣΑΓΩΓΗ

Το **Mininet** αποτελεί ένα από τα πρώτα ανοιχτά και ελεύθερα λογισμικά εξομοιωτών που αναπτύχθηκαν ειδικά για την υποστήριξη της τεχνολογίας SDN. Παρέχει τη δυνατότητα υλοποίησης και αξιολόγησης δικτύων μικρής κλίμακας με τεχνητή κυκλοφορία σε συστήματα που δεν απαιτούν υψηλή υπολογιστική ισχύ [13], [14]. Η ανάπτυξη μεγάλων εργαστηριακών δικτύων με φυσικό εξοπλισμό είναι δαπανηρή και δύσκολη. Το Mininet, μέσω της στρατηγικής της εικονικοποίησης, επιτρέπει τη δημιουργία πρωτοτύπων και την προσομοίωση τοπολογιών SDN με χαμηλό κόστος.

3.2 Λογισμικό Mininet

Το Mininet υλοποιεί μια συλλογή από hosts, switches, routers και links πάνω σε έναν ενιαίο πυρήνα Linux. Χρησιμοποιεί τεχνικές ελαφριάς εικονικοποίησης, ώστε ένα σύστημα να συμπεριφέρεται σαν πλήρες δίκτυο, επιτρέποντας την εκτέλεση πραγματικών εφαρμογών. Τα πακέτα διακινούνται μέσω εικονικών διεπαφών Ethernet, με δυνατότητα καθορισμού εύρους ζώνης, καθυστέρησης και ουρών αναμονής. Έτσι, η απόδοση που μετράται στο Mininet αντιστοιχεί σε αυτή που θα παρατηρούνταν σε πραγματικές συνθήκες, με τη μόνη διαφορά ότι προσομοιώνεται σε μικρότερη κλίμακα [21].

Τα κύρια χαρακτηριστικά που οδήγησαν στη δημιουργία του Mininet είναι:

- **Ευελιξία**: Δυνατότητα διαμόρφωσης νέων τοπολογιών με χρήση κοινών γλωσσών προγραμματισμού.
- **Εφαρμοσιμότητα**: Οι εφαρμογές που αναπτύσσονται μπορούν να χρησιμοποιηθούν σε πραγματικά δίκτυα χωρίς αλλαγή κώδικα.
- **Διαπεραστικότητα**: Παρουσιάζει συμπεριφορά όμοια με πραγματικά δίκτυα.
- **Επεκτασιμότητα**: Υποστηρίζει την κλιμάκωση σε δίκτυα εκατοντάδων ή χιλιάδων switches.
- **Ρεαλισμός**: Παρέχει αξιόπιστη αναπαράσταση δικτύων σε πραγματικό χρόνο.
- **Κοινοχρησία**: Οι τοπολογίες μπορούν εύκολα να αναπαραχθούν και να μοιραστούν.

3.3 Πλεονεκτήματα του Mininet

Το Mininet έχει καθιερωθεί ως βασικό εργαλείο προσομοίωσης για SDN, παρουσιάζοντας σημαντικά πλεονεκτήματα [13]:

- Γρήγορη εκκίνηση και ευκολία δοκιμών.
- Δυνατότητα δημιουργίας προσαρμοσμένων τοπολογιών (από απλές έως σύνθετες, όπως data centers).
- Υποστήριξη οποιουδήποτε λογισμικού είναι διαθέσιμο στο Linux.
- Προγραμματιζόμενη προώθηση πακέτων μέσω OpenFlow.
- Συμβατότητα με φυσικούς υπολογιστές, εικονικές μηχανές και cloud.
- Διαμοιρασμός/αντιγραφή τοπολογιών σε άλλα περιβάλλοντα.
- Εύχρηστο για χρήστες που αναπτύσσουν πειράματα με Python scripts.
- Ανοιχτού κώδικα, με ενεργή κοινότητα ανάπτυξης.

3.3.1 Περιορισμοί του Mininet

Παρά τα πλεονεκτήματα, το Mininet παρουσιάζει και ορισμένους περιορισμούς [14], [21]:

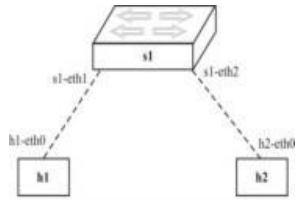
- Η εκτέλεση σε έναν υπολογιστή περιορίζεται από τους διαθέσιμους πόρους.
- Όλοι οι hosts μοιράζονται τον ίδιο πυρήνα Linux, γεγονός που αποκλείει λογισμικό εξαρτημένο από άλλα λειτουργικά συστήματα.
- Δεν προσφέρει ενσωματωμένο controller – απαιτείται ξεχωριστή εγκατάσταση.
- Από προεπιλογή, το δίκτυο είναι απομονωμένο από LAN/Internet (χρειάζεται NAT).
- Όλοι οι hosts μοιράζονται το ίδιο file system, κάτι που μπορεί να προκαλέσει συγκρούσεις.
- Δεν υπάρχει ακριβής αντίληψη «εικονικού χρόνου», άρα η εξομοίωση υπερ-υψηλών ταχυτήτων (π.χ. >100 Gbps) είναι περιορισμένη.

3.4 Δημιουργία Τοπολογιών

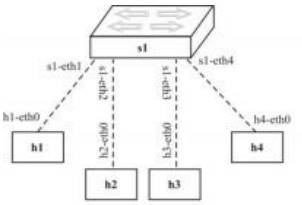
Το Mininet προσφέρει προκαθορισμένες αλλά και προσαρμοσμένες τοπολογίες. Ενδεικτικά [21]:

- **Minimal**: Ένας switch και δύο hosts.
- **Single**: Ένας switch και k hosts.
- **Reversed**: Όμοια με την single, αλλά με αντεστραμμένες συνδέσεις.
- **Linear**: k switches και k hosts σε γραμμική διάταξη.
- **Tree**: Δενδρική τοπολογία με k επίπεδα.

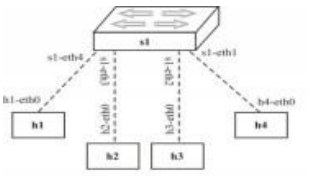
Για πιο εξειδικευμένες ανάγκες, οι χρήστες μπορούν να δημιουργήσουν δικές τους τοπολογίες με Python κώδικα, χρησιμοποιώντας τις κλάσεις και τις μεθόδους του Mininet (Topo, addSwitch(), addHost(), addLink(), start(), stop()).



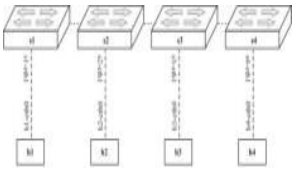
Minimal Topology



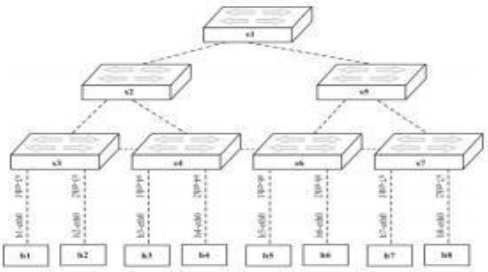
Single Topology



Reversed Topology



Linear Topology



Tree Topology

Εικόνα 8:ΤοπολογίεςMininet

Για τις παραμετροποιημένες τοπολογίες, το μόνο που χρειάζεται είναι λίγες γραμμές κώδικα Python, έτσι ώστε να δημιουργηθεί μία ευέλικτη τοπολογία η οποία μπορεί να ρυθμιστεί με βάση τις παραμέτρους που εισάγονται σε αυτήν και να επαναχρησιμοποιηθεί για διάφορα πειράματα.

```
#!/usr/bin/python

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class SingleSwitchTopo(Topo):
    "Single switch connected to n hosts."
    def build(self, n=2):
        switch = self.addSwitch('s1')
        # Python's range(N) generates 0..N-1
        for h in range(n):
            host = self.addHost('h%s' % (h + 1))
            self.addLink(host, switch)

def simpleTest():
    "Create and test a simple network"
    topo = SingleSwitchTopo(n=4)
    net = Mininet(topo)
    net.start()
    print "Dumping host connections"
    dumpNodeConnections(net.hosts)
    print "Testing network connectivity"
    net.pingAll()
    net.stop()

if __name__ == '__main__':
    # Tell mininet to print useful information
    setLogLevel('info')
    simpleTest()
```

Εικόνα 9 :Κώδικας τοπολογίας Mininet

Στο παραπάνω παράδειγμα της Εικόνας 4.2 παρουσιάζεται ο κώδικας μίας απλής τοπολογίας δικτύου βασισμένη στο Mininet, η οποία απαρτίζεται από ένα καθορισμένο σύνολο χρηστών και συνδέονται με έναν μεταγωγέα. Ο κώδικας της περιλαμβάνει σημαντικές κλάσεις, μεθόδους, συναρτήσεις και μεταβλητές.

- **Topo**: Η βασική κλάση για τοπολογίες Mininet.

- **build()**:Η μέθοδος παράκαμψης στην κλάση της τοπολογίας. Οι παράμετροι (n)θα εισαχθούν αυτόματα.

- **add Switch()**: Εισάγει έναν μεταγωγέα σε μια τοπολογία και επιστρέφει την ονομασία του.

- **Add Host()**: Εισάγει ένα τερματικό σε μια τοπολογία και επιστρέφει την ονομασία του.

- **Add Link()**: Προσθέτει έναν αμφίδρομο σύνδεσμο σε μια τοπολογία (και επιστρέφει ένα κλειδί σύνδεσης, αν και δεν είναι σημαντικό). Οι συνδέσεις στο Mininet είναι διπλής κατεύθυνσης (αμφίδρομες) εκτός και αν αναφέρεται διαφορετικά.

- **Mininet**: Κύρια κλάση για τη δημιουργία και τη διαχείριση ενός δικτύου.

- **start()**:Ενεργοποιείται η λειτουργία του δικτύου.

- **Ping All( )**: Δοκιμάζει την συνδεσιμότητα προσπαθώντας να έχουν όλοι οι κόμβοι ping ο ένας με τον άλλον. Η εντολή ping χρησιμοποιεί το ICMP (Internet Control Message Protocol) για να στείλει πακέτα τύπου request/reply.

- **stop()**:Παύει η λειτουργία του δικτύου.

- **net.hosts**:Εμφανίζει όλα τα ονόματα των κόμβων του δικτύου.

- **dumpNodeConnections( )**: Καταργεί τις συνδέσεις σε / από ένα σύνολο κόμβων**.**

- **setLogLevel ('info' | 'debug' | 'output')**: Ορίζει το προεπιλεγμένο επίπεδο εξόδου του Mininet. Συνιστάται η εντολή "info" καθώς παρέχει χρήσιμες πληροφορίες.

3.5 Εκτέλεση Προγραμμάτων σε Hosts και Μέθοδοι Διαμόρφωσης

Ένα από τα πιο σημαντικά πράγματα που θα πρέπει να εφαρμόζονται, κατά την διάρκεια των πειραμάτων, είναι η εκτέλεση προγραμμάτων στα τερματικά (hosts), ώστε να μπορούν να χρησιμοποιηθούν περισσότερες από τις απλές εντολές, όπως ping All() και iperf(), που παρέχονται από το ίδιο το Mininet.

Κάθε host στο Mininet είναι ουσιαστικά ένα κέλυφος (shell) τύπου bash που συνδέεται με μία ή περισσότερες διεπαφές δικτύου, με αποτέλεσμα ο ευκολότερος τρόπος αλληλεπίδρασης με οποιονδήποτε host είναι η αποστολή δεδομένων στο κέλυφος χρησιμοποιώντας τη μέθοδο cmd().

Για την εκτέλεση μιας εντολής σε έναν host με σκοπό να ληφθεί το αποτέλεσμα, χρησιμοποιείται ο παρακάτω κώδικας:

```
h1=net.get('h1')
result=h1.cmd('ifconfig') print
```

Htb: Αλγόριθμος ο οποίος χρησιμοποιείται για να ελέγχει αν οι μεταδόσεις δεδομένων συμφωνούν με τα καθορισμένα όρια του εύρους ζώνη

Τα τερματικά του Mininet παρέχουν μια σειρά από εύκολες μεθόδους για τη διαμόρφωση του δικτύου και είναι οι εξής:

- **IP()**:Επιστρέφει την IP διεύθυνση ενός host ή μίας συγκεκριμένης διεπαφής.
- **MAC()**:Επιστρέφει την MAC διεύθυνση ενός host ή μίας συγκεκριμένης διεπαφής.
- **setARP( )**: Προσθέτει μια στατική καταχώρηση ARP στην προσωρινή μνήμη ARP του host.
- **setIP()**:Ορίζει τη διεύθυνση IP ενός host ή μίας συγκεκριμένης διεπαφής.
- **setMAC()**:Ορίζει τη διεύθυνση MAC ενός host ή μίας συγκεκριμένης διεπαφής.

3.6 Κοινό Σύστημα Αρχείων

Ένα αξιοσημείωτο γεγονός που συμβαίνει με τα τερματικά του Mininet, και μάλιστα από προεπιλογή, είναι ότι μοιράζονται τα ριζικά αρχεία συστήματος του υποκείμενου εξυπηρετητή. Συνήθως αυτό επηρεάζει θετικά την όλη διαδικασία, δεδομένου ότι η δημιουργία ενός ξεχωριστού συστήματος αρχείων για κάθε τερματικό του Mininet απαιτεί πολύ χρόνο.

Επίσης, ένα πλεονέκτημα που προκύπτει από το κοινό σύστημα αρχείων είναι ότι, δεν χρειάζεται η λειτουργία της αντιγραφής των δεδομένων μεταξύ των τερματικών του Mininet, διότι υπάρχουν είδη εκεί.

Όπως κάθε σύστημα εκτός από πλεονεκτήματα έχει και μειονεκτήματα. Ένα από αυτά είναι η αναγκαστική δημιουργία νέων αρχείων διαμόρφωσης παραμέτρων για κάθε host, σε περίπτωση που χρειαστεί συγκεκριμένη ρύθμιση παραμέτρων για κάποιο πρόγραμμα (π.χ. httpd). Επιπλέον, μειονέκτημα αποτελεί η δημιουργία ίδιων αρχείων στον ίδιο κατάλογο σε διάφορους hosts, καθώς προκύπτουν συγκρούσεις μεταξύ των αρχείων αυτών.

3.7 Mininet CLI

Το Mininet περιλαμβάνει μια διασύνδεση γραμμής εντολών (CLI), η οποία μπορεί να εφαρμοστεί σε δίκτυο και παρέχει μια ποικιλία από χρήσιμες εντολές, εκ των οποίων μία από αυτές είναι η δυνατότητα προβολής xterm παραθύρων για την εκτέλεση εντολών σε μεμονωμένους κόμβους του δικτύου. [21]

Η εκκίνηση του CLI μπορεί να φανεί χρήσιμη για τον εντοπισμό σφαλμάτων στο δίκτυό, καθώς επιτρέπει την προβολή της τοπολογίας του δικτύου (με την εντολή net), τον έλεγχο της συνδεσιμότητας (με την εντολή pingall) και την αποστολή εντολών σε μεμονωμένα τερματικά.

3.8 Mininet API

Σε προηγούμενη παράγραφο αναφερθήκαμε στις κλάσεις της Python και πιο συγκεκριμένα στις Topo, Mininet, Host, Switch, Link. Όλες αυτές οι κλάσεις μαζί με διάφορες άλλες περιλαμβάνονται στο API του Mininet και μπορούν να διαιρεθούν σε επίπεδα, έτσι ώστε να κάνουν πιο εύκολο τον προγραμματισμό. [21]

Το API του Mininet είναι χτισμένο σε τρία βασικά επίπεδα:

- **API χαμηλού επιπέδου**: Αποτελείται από τους βασικές κλάσεις των κόμβων και των συνδέσμων (όπως οι Host, Switch, Link και οι υποκλάσεις τους), οι οποίες μπορούν να αρχικοποιηθούν μεμονωμένα και να χρησιμοποιηθούν για τη δημιουργία ενός δικτύου, διαδικασία που θεωρείται αρκετά δύσχρηστη.

  Παράδειγμα API χαμηλού επιπέδου:

```
h1=Host('h1')
h2=Host('h2')
s1 = OVSSwitch( 's1', inNamespace=False )
c0=Controller('c0',inNamespace=False)
Link( h1, s1 )
Link( h2, s1 )
h1.setIP('10.1/8')
h2.setIP('10.2/8')
c0.start()
```

- **API μεσαίου επιπέδου**: Προσθέτει αντικείμενα τύπου Mininet τα οποία ενεργούν ως επιπρόσθετα στοιχεία στους κόμβους και συνδέσμους. Παρέχει ένα σύνολο μεθόδων (όπως add Host( ), add Switch( ) και add Link( )) για την προσθήκη κόμβων και συνδέσεων σε ένα δίκτυο, καθώς και για την διαμόρφωση, εκκίνηση και τερματισμό δικτύου (κυρίως start ( ) και stop ( ).).

  Παράδειγμα API μεσαίου επιπέδου:

```
net=Mininet()
h1 = net.addHost( 'h1') h2
= net.addHost( 'h2')
s1=net.addSwitch('s1')
c0=net.addController('c0')
net.addLink( h1, s1 )
net.addLink( h2, s1 )
net.start()
```

- **API υψηλού επιπέδου**: Προσθέτει ένα πρότυπο διαμόρφωσης της τοπολογίας, την κλάση Topo, η οποία παρέχει τη δυνατότητα δημιουργίας επαναχρησιμοποιούμενων, παραμετροποιημένων προτύπων τοπολογίας. Αυτά τα πρότυπα μπορούν να οριστούν από την εντολή mn (μέσω της επιλογής --custom) και να εκτελεστούν από τη γραμμή εντολών.

  Παράδειγμα API υψηλού επιπέδου:

```
class SingleSwitchTopo( Topo ):
     "Single Switch
     Topology"defbuild(self,coun
     t=1):

         hosts=[self.addHost('h%d'%i)

                   foriinrange(1,count+1)] s1 =
         self.addSwitch( 's1')

         for h in hosts:
             self.addLink(h,s1)
```

3.9 Open Flow και Προσαρμοσμένη Δρομολόγηση

Ένα από τα πιο ισχυρά και χαρακτηριστικά του Mininet είναι ότι χρησιμοποιεί την τεχνολογία SDN. Με τη χρήση του OpenFlow πρωτοκόλλου και των σχετικών εργαλείων, είναι εφικτός ο προγραμματισμός των μεταγωγέων έτσι ώστε να εκτελούν οποιαδήποτε εντολή σχετικά με τα πακέτα που εισέρχονται στους μεταγωγείς. Το OpenFlow καθιστά τους εξομοιωτές όπως το Mininet πολύ πιο χρήσιμους, αφού τα σχέδια του συστήματος δικτύου, συμπεριλαμβανομένης και της προώθησης πακέτων με OpenFlow, μπορούν εύκολα να μεταφερθούν σε OpenFlow μεταγωγείς για διάφορες λειτουργίες. [21]

Εάν η εντολή mn (εκκίνηση του Mininet) εκτελεστεί χωρίς να προσδιοριστεί ένας ελεγκτής, το αποτέλεσμα θα είναι η επιλογή ενός προεπιλεγμένου ελεγκτή (π.χ. Controller ή OVS Controller), ανάλογα με το ποιος είναι διαθέσιμος. Η εντολή αυτή ισοδυναμεί στην:

```
$ sudomn controller default
```

Αυτός ο ελεγκτής εφαρμόζει έναν απλό μεταγωγέα εκμάθησης Ethernet υποστηρίζοντας έως και 16 μεμονωμένους μεταγωγείς.

Σε περίπτωση που κλιθεί ο κατασκευαστής Mininet( ) σε ένα πείραμα χωρίς να καθοριστεί μία κλάση ελεγκτή, τότε από προεπιλογή θα χρησιμοποιηθεί η κλάση Controller( ) για την δημιουργία ενός ελεγκτή τύπου Stanford / OpenFlow. Ωστόσο, υφίσταται η δυνατότητα χρησιμοποίησης ενός διαμορφωμένου ελεγκτή, όπου ο χρήστης εύκολα μπορεί να δημιουργήσει μια υποκατηγορία κλάσης του Controller( ) και να την εκτελέσει στο Mininet.

3.10 Ορισμός POX

Το POX είναι μια πλατφόρμα ανάπτυξης ανοιχτού κώδικα για εφαρμογές ελέγχου δικτύωσης που ορίζονται από λογισμικό (SDN) που βασίζονται σε Python, όπως ελεγκτές OpenFlow SDN.

Το POX παρέχει ένα πλαίσιο για την επικοινωνία με διακόπτες SDN χρησιμοποιώντας είτε το πρωτόκολλο OpenFlow είτε OVSDB. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν το POX για να δημιουργήσουν έναν ελεγκτή SDN χρησιμοποιώντας τη γλώσσα προγραμματισμού Python. Είναι ένα δημοφιλές εργαλείο για τη διδασκαλία και την έρευνα προγραμματισμού δικτύων και εφαρμογών δικτύου που ορίζονται από λογισμικό.

Χαρακτηριστικά POX

• Διεπαφή Pythonic Open Flow.
• Επαναχρησιμοποιήσιμα στοιχεία δειγμάτων για επιλογή διαδρομής, ανακάλυψη τοπολογίας κ.λπ.
• Εκτελούνται οπουδήποτε μπορούν να συνδυαστούν με χρόνο εκτέλεσης PyPy χωρίς εγκατάσταση για εύκολη ανάπτυξη.
• Στοχεύει συγκεκριμένα Linux, Mac OS και Windows.
• Ανακάλυψη τοπολογίας.
• Υποστηρίζει το ίδιο GUI και εργαλεία οπτικοποίησης με το NOX.
• Έχει καλή απόδοση σε σύγκριση με εφαρμογές NOX γραμμένες σε Python.

Στοιχεία POX

Τα στοιχεία POX είναι πρόσθετα προγράμματα Python που μπορούν να κληθούν όταν το POX ξεκινά από τη γραμμή εντολών. Αυτά τα στοιχεία υλοποιούν τη λειτουργικότητα του δικτύου στο δίκτυο που ορίζεται από το λογισμικό. Η λειτουργικότητα POX παρέχεται από τα εξαρτήματά του, μπορεί να χρησιμοποιηθεί αμέσως ως βασικός ελεγκτής SDN χρησιμοποιώντας τα στοκ εξαρτήματα που συνοδεύουν .Οι προγραμματιστές μπορούν να δημιουργήσουν έναν πιο περίπλοκο ελεγκτή SDN δημιουργώντας νέα στοιχεία POX. Οι προγραμματιστές μπορούν να γράψουν εφαρμογές δικτύου που απευθύνονται στο API του POX.
$

3.11 Προγραμματισμός για POX

Ο γενικός σκοπός όλων των ελεγκτών SDN, συμπεριλαμβανομένου του POX, είναι να επιτρέπουν στους χρήστες να γράφουν τις δικές τους εφαρμογές που χρησιμοποιούν τον ελεγκτή ως ενδιάμεσο — ή επίπεδο αφαίρεσης — μεταξύ των εφαρμογών δικτύου και του εξοπλισμού δικτύου.

Για να μάθουν πώς να γράφουν εφαρμογές για POX, οι προγραμματιστές μπορούν να μελετήσουν τα στοκ στοιχεία POX ως παραδείγματα που δείχνουν πώς να γράφουν τα δικά τους στοιχεία ή μπορούν να αναθεωρήσουν την τεκμηρίωση του POX API για να μάθουν πώς να γράφουν εφαρμογές δικτύωσης που χρησιμοποιούν το POX Python API.

Εγκατάσταση POX

Προκειμένου να γίνει εγκατάσταση του Mininet και του POX σε ένα σύστημα Linux — είτε υλικό είτε εικονική μηχανή — μπορεί να γίνει χρήση του σεναρίου , το οποίο εγκαθιστά επίσης τον ελεγκτή POX όταν εγκαθιστά το Mininet. Εάν πρόκειται να πραγματοποιηθεί η εγκατάσταση του POX από μόνο του, τότε θα πρέπει να ακολουθουθούν οι οδηγίες εγκατάστασης του POX από την τεκμηρίωση του POX.

Εκτέλεση POX

Το POX ξεκινάει εκτελώντας το πρόγραμμα *pox.py* και προσδιορίζοντας τα στοιχεία POX που θα χρησιμοποιηθούν. Για παράδειγμα, για να εκτελεστεί το POX ώστε οι διακόπτες που ελέγχει να μιμούνται τη συμπεριφορά των διακοπτών εκμάθησης Ethernet, πρέπει να εκτελεστεί η εντολή:

mininet@mininet-vm:~$ sudo ~/pox/pox.py forwarding.l2_learning

Η κονσόλα POX

Η *κονσόλα POX* είναι η συνεδρία τερματικού από την οποία γίνεται η εκτέλεση του ελεγκτή POX. Μετά την εκκίνηση του POX, θα εμφανίσει πληροφορίες καταγραφής και μπορεί προαιρετικά να εμφανίσει μια διαδραστική γραμμή εντολών Python, εάν το POX ξεκινήσει με το στοιχείο *py* .

Έξοδος από το POX

Για έξοδο από το POX, πραγματοποιείτε ο συνδυασμός πλήκτρων *control-C* στην κονσόλα POX.

ΚΕΦΑΛΑΙΟ 4ο

ΕΦΑΡΜΟΓΗ ΕΞΟΜΕΙΩΤΗ MININET

4.1 Εισαγωγή

Για την εγκατάσταση του προσομοιωτή Mininet για δίκτυα SDN απαιτείται λειτουργικό σύστημα Linux που μπορεί να εγκατασταθεί σε οποιοδήποτε μηχάνημα ως μεμονωμένο λειτουργικό σύστημα ή σε οποιαδήποτε εφαρμογή εικονικής μηχανής (VM) όπως το Oracle Virtual Box ή το VM Ware Workstation. Μετά την επιτυχή εγκατάσταση του λειτουργικού συστήματος Linux σε περιβάλλον εικονικής μηχανής, απαιτείται η εγκατάσταση του Mininet στο λειτουργικό σύστημα Linux.

Συνιστάται να εγκαταστήσουμε την τελευταία έκδοση του Ubuntu 24.04 . Μετά την επιτυχή εγκατάσταση του Linux (Ubuntu) σε Virtual Machine (VM Ware work station), το Mininet μπορεί να εγκατασταθεί από το github χρησιμοποιώντας την εντολή git από την ιστοσελίδα του github. Εάν το βοηθητικό πρόγραμμα git δεν είναι εγκατεστημένο στο Ubuntu, μπορεί να εγκατασταθεί «sudoapt-getinstallgit»

Πριν από την εγκατάσταση του Mininet, συνιστάται επίσης να ενημερώσετε την εγκατάσταση του Linux. Για την ενημέρωση του λειτουργικού συστήματος Linux χρησιμοποιείται η εντολή «sudoapt-getupdate» για την ενημέρωση της εγκατάστασης Linux και την εγκατάσταση όλων των ελλείψεων και των απαιτούμενων εξαρτήσεων.

Μετά την ενημέρωση και την εγκατάσταση του βοηθητικού προγράμματος git όπως αναφέρθηκε παραπάνω, το Mininet μπορεί να μεταφορτωθεί από τον παρακάτω ιστότοπο github.

«https://github.com/mininet/mininnet»

Συνιστάται η λήψη του «Mininet» κάτω από τον κατάλογο «/» (root). Για τη λήψη του πακέτου Mininet από το github, χρησιμοποιείται η ακόλουθη εντολή.

«git clone https://github.com/mininet/mininet»

Το Mininet θα μεταφορτωθεί στο φάκελο «/». Αλλάζουμε τον φάκελο χρησιμοποιώντας την εντολή «cd» και κάτω από το «/root/Mininet/util» θα υπάρχει ένα αρχείο με όνομα «install.sh». Εκτελούμε το «./install.sh» για την εγκατάσταση του Mininet. Μετά την εκτέλεση αυτής της εντολής το Mininet θα αρχίσει να κατεβάζει το λογισμικό και θα ξεκινήσει την εγκατάσταση του Mininet. Μετά την επιτυχή εγκατάσταση θα λάβουμε το μήνυμα «Enjoy Mininet» ως επιβεβαίωση ότι το Mininet στο μηχάνημα Linux Ubuntu έχει εγκατασταθεί με επιτυχία και μπορούμε να προχωρήσουμε με την προσομοίωση του δικτύου μας.

4.2 Δημιουργία τοπολογίας

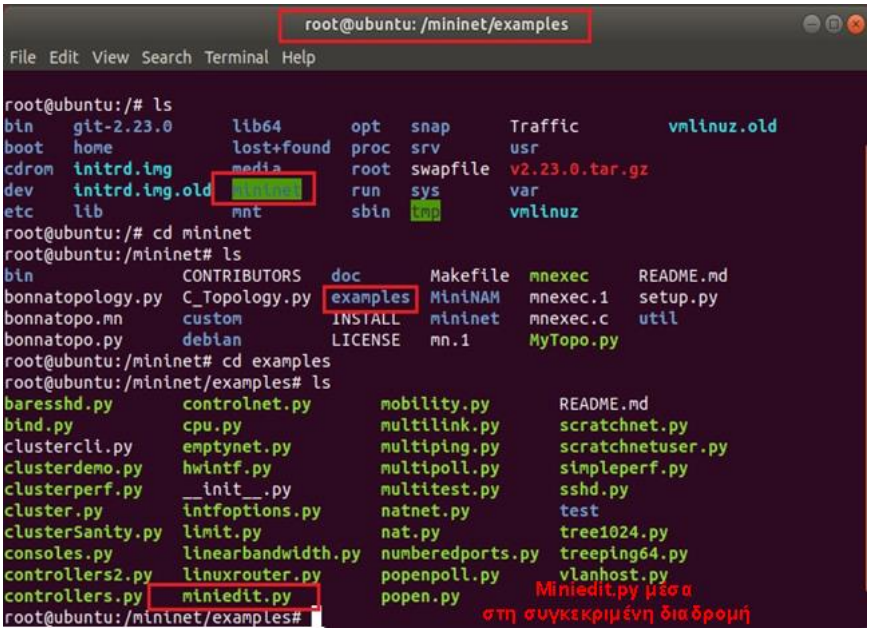
Υπάρχουν δύο τρόποι για τη δημιουργία τοπολογιών δικτύου στο Mininet, ο ένας είναι χρησιμοποιώντας το Python API μέσω προγραμματισμού με τη συγγραφή γραμμών κώδικα και ο άλλος τρόπος για τη δημιουργία τοπολογίας δικτύου είναι η χρήση του Miniedit (Mininet built-in Utility) Graphical User Interface (GUI). Η δημιουργία τοπολογίας δικτύου χρησιμοποιώντας το βοηθητικό πρόγραμμα Miniedit GUI κάνει τη δημιουργία τοπολογίας πιο εύκολη και βολική χρησιμοποιώντας επιλογές drag and drop devices. Με τη χρήση του βοηθητικού προγράμματος Miniedit GUI οι τοπολογίες δικτύου μπορούν να αποθηκευτούν με επέκταση .mn (Mininet Extension) καθώς και ως σενάριο python με επέκταση .py.

Δημιουργία τοπολογίας με το βοηθητικό πρόγραμμα Miniedit:

Για τη δημιουργία τοπολογίας δικτύου πρέπει πρώτα να αποκτήσουμε πρόσβαση στο βοηθητικό πρόγραμμα Miniedit. Το αρχείο Miniedit.py υπάρχει στο φάκελο εγκατάστασης του Mininet στο φάκελο examples.

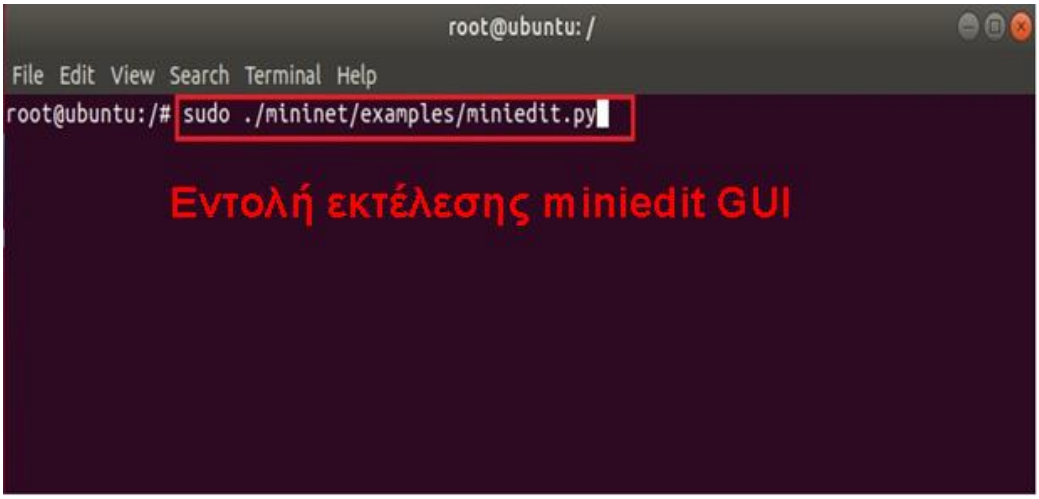
/Mininet/examples/Miniedit.py

Η θέση του αρχείου Miniedit.py αναφέρεται στην παρακάτω εικόνα.



Εικόνα 10: Φάκελος εγκατάστασης Mininet Κάτω από τον κατάλογο Root / και τη διαδρομή του φακέλου Examples.

Για να εκτελέσουμε το βοηθητικό πρόγραμμα Miniedit GUI για τη δημιουργία τοπολογίας δικτύου χρησιμοποιούμε την ακόλουθη εντολή.

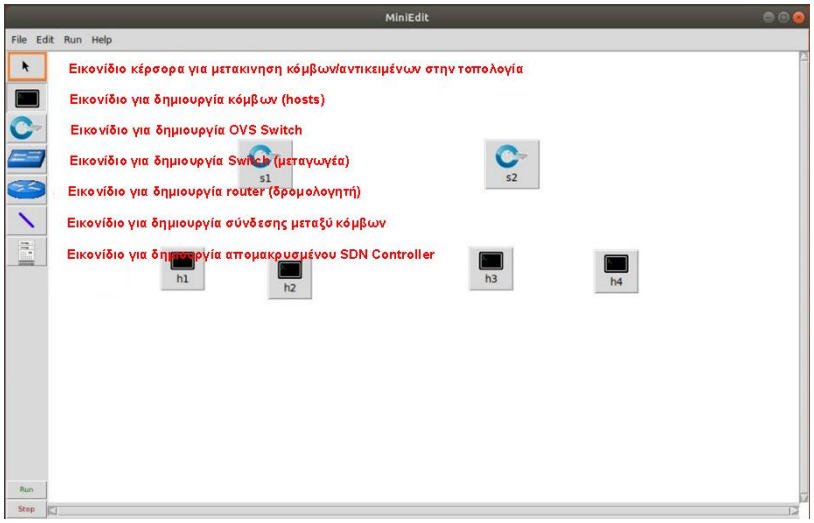


Εικόνα 11: Εντολή για την εκτέλεση του βοηθητικού προγράμματος Mininet GUI (Miniedit)

Η παραπάνω εντολή πρέπει να εκτελεστεί στο τερματικό εντολών του Linux για τη δημιουργία τοπολογίας δικτύου. Επίσης η εντολή για την εκτέλεση του miniedit.py απαιτεί δικαιώματα Root. Μετά την εκτέλεση της εντολής θα εμφανιστεί η οθόνη του Miniedit GUI που θα μας βοηθήσει να δημιουργήσουμε τοπολογία δικτύου χρησιμοποιώντας επιλογές drag and drop. Η οθόνη του Miniedit GUI δίνεται παρακάτω

Εικόνα 12: Χώρος εργασίας σχεδιασμού τοπολογίας δικτύου Mini Edit

Το παραπάνω σχήμα δείχνει την οθόνη Miniedit που χρησιμοποιείται για τη δημιουργία τοπολογίας δικτύου. Διάφορα σύμβολα κόμβων εμφανίζονται στη αριστερή πλευρά της οθόνης και μπορούν να χρησιμοποιηθούν για τη δημιουργία τοπολογίας δικτύου. Το εικονίδιο με το βέλος χρησιμοποιείται για σύρσιμο και τοποθέτηση κόμβων στην περιοχή της τοπολογίας του δικτύου.

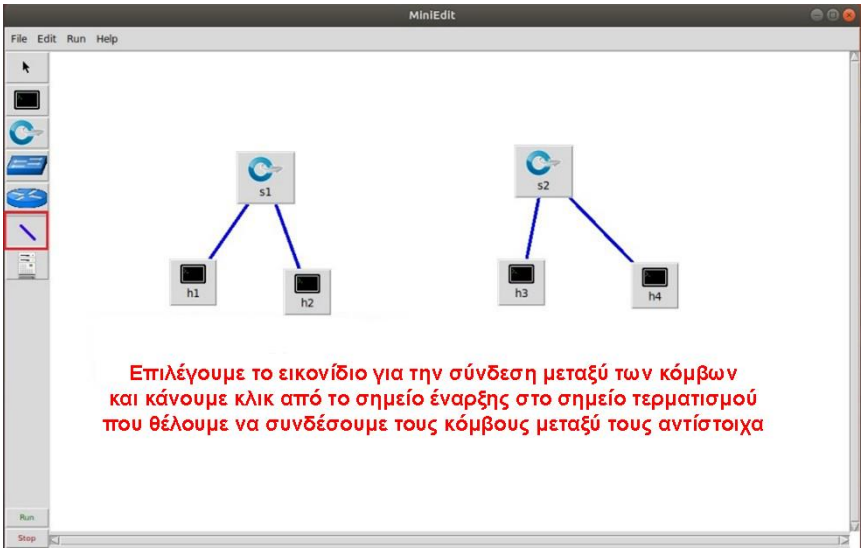


Εικόνα 13: Επεξήγηση στοιχείων της γραμμής εργαλείων Miniedit

Δεύτερο εικονίδιο, στη στήλη Εικονιδίων που χρησιμοποιείται για τη δημιουργία κόμβων. Επιλέγουμε το εικονίδιο κόμβου και κάνουμε κλικ στο χώρο εργασίας τοπολογίας δικτύου για τη δημιουργία κόμβου. Στο παραπάνω παράδειγμα υπάρχουν 4 κόμβοι με ονομασίες h1, h2, h3 και h4.

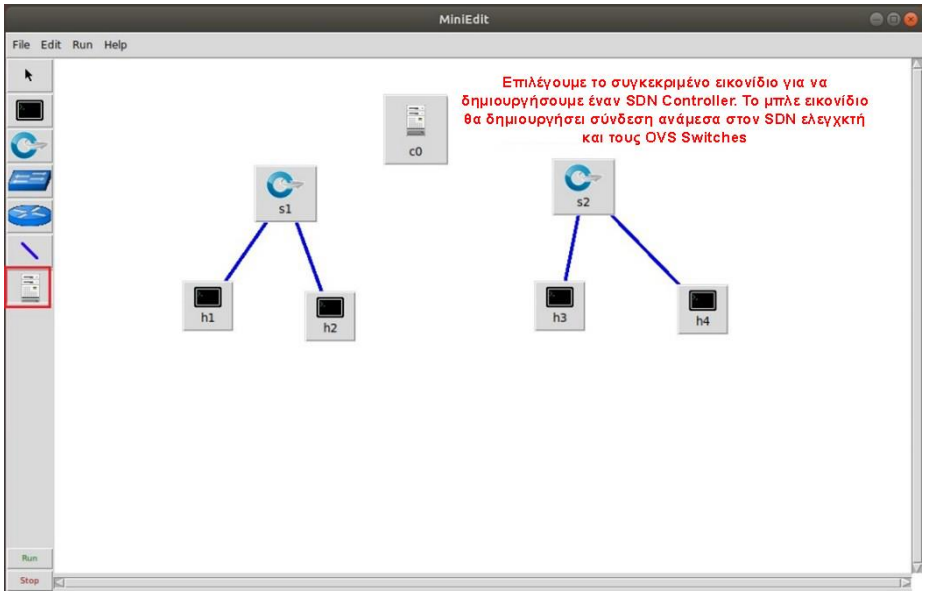
Το τρίτο εικονίδιο στη στήλη χρησιμοποιείται για την προσθήκη OVS (Open Virtual Switches) στην τοπολογία δικτύου. Κάνουμε κλικ στο εικονίδιο OVS και επιλέγουμε στο χώρο εργασίας της τοπολογίας δικτύου για την προσθήκη μεταγωγέων OVS. Το τέταρτο εικονίδιο χρησιμοποιείται για την προσθήκη παλαιών μεταγωγέων δικτύου και πέμπτο για την προσθήκη παλαιών δρομολογητών δικτύου στην τοπολογία.

Το έκτο εικονίδιο στη στήλη χρησιμοποιείται για την προσθήκη συνδέσεων μεταξύ των δημιουργημένων κόμβων. Για τη σύνδεση συσκευών δικτύου στην τοπολογία δικτύου, κάνουμε κλικ στο εικονίδιο συνδέσμου και κάνουμε κλικ στους κόμβους από τους οποίους απαιτείται η δημιουργία συνδέσμου ενώ παράλληλα, χωρίς να αφήσουμε το κουμπί του ποντικιού, κάνουμε κλικ στον κόμβο προς τον οποίο απαιτείται η δημιουργία συνδέσμου. Ένα παράδειγμα σύνδεσης αναφέρεται και στην παρακάτω εικόνα



Σχήμα 14: Τοποθέτηση κόμβων και δημιουργία συνδέσμων στο σχεδιασμό τοπολογίας δικτύου.

Στο παραπάνω σχήμα παρουσιάζεται η σύνδεση μεταξύ των κόμβων, εάν απαιτείται η διαμόρφωση οποιουδήποτε κόμβου ή συνδέσμου και των ιδιοτήτων του, κάνουμε δεξί κλικ στον κόμβο ή τον σύνδεσμο και επιλέγουμε ιδιότητες συνδέσμου ή κόμβου για να τους διαμορφώσουμε αντίστοιχα.

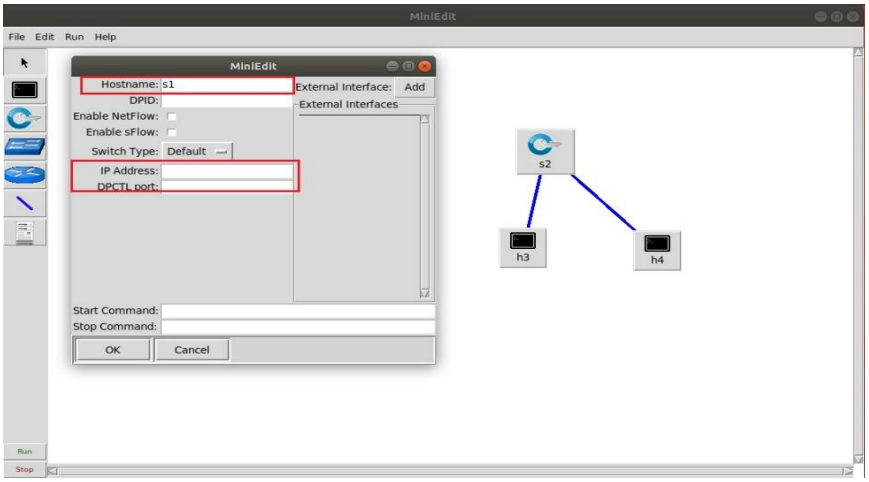


Εικόνα 15: Δημιουργία ελεγκτή SDN στην τοπολογία δικτύου

Εάν απαιτείται η προσθήκη οποιουδήποτε ελεγκτή SDN (POX, RYU, ONOS) στην τοπολογία, το τελευταίο εικονίδιο στη στήλη μπορεί να χρησιμοποιηθεί για την τοποθέτηση του ελεγκτή SDN. Οι ιδιότητες του ελεγκτή μπορούν επίσης να διαμορφωθούν κάνοντας δεξί κλικ στον ελεγκτή SDN και στη συνέχεια επιλέγοντας την επιλογή properties.

4.3 Διαμόρφωση ιδιοτήτων OVS Switch, Host Nodes, Links και SDN Controllers Properties:

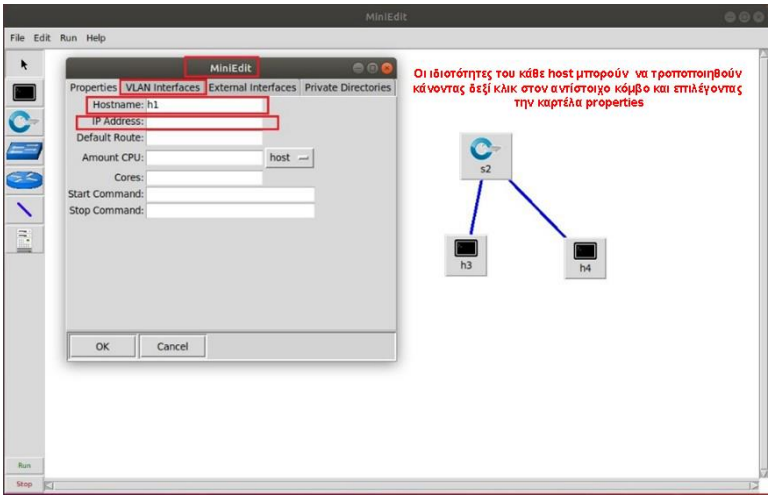
Η εκ των προτέρων διαμόρφωση και οι ιδιότητες μπορούν να τροποποιηθούν για τα OVS Switch, HostNodes, Links και SDN Controllers χρησιμοποιώντας τις ιδιότητες των OVS. Παραδείγματα για τη διαμόρφωση των ιδιοτήτων των OVS Switch, HostNode, Link και SDN Controller παρουσιάζονται παρακάτω.



Εικόνα 16: Διαμόρφωση ιδιοτήτων OVS, όνομα κεντρικού υπολογιστή, διεύθυνση IP, ρύθμιση θύρας DPTCL.

Εάν απαιτείται διαμόρφωση του OVS Switch, οι ιδιότητες μπορούν να αλλάξουν κάνοντας δεξί κλικ στο OVS Switch και κάνοντας κλικ στην επιλογή properties. Αφού κάνουμε κλικ στην επιλογή properties (ιδιότητες), το παραπάνω παράθυρο θα μας ζητήσει να διαμορφώσουμε τις ιδιότητες του. Μπορούμε να αλλάξουμε το όνομα κεντρικού υπολογιστή για το OVS Switch, τη διεύθυνση IP του καθώς και τα πρωτόκολλα που απαιτούνται για την εκτέλεση όπως το NetFlow ή το SFlow.

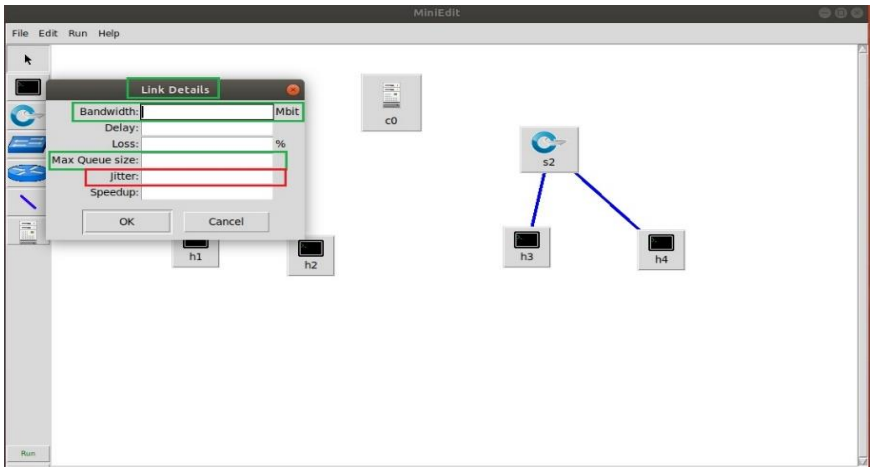
Όπως και οι μεταγωγείς OVS, οι κόμβοι κεντρικού υπολογιστή μπορούν επίσης να διαμορφωθούν χρησιμοποιώντας την επιλογή ιδιοτήτων. Για τη διαμόρφωση των ιδιοτήτων του κόμβου κεντρικού υπολογιστή, κάνουμε δεξί κλικ στον κόμβο κεντρικού υπολογιστή και επιλέγουμε το properties . Θα εμφανιστεί μια οθόνη διαμόρφωσης του κόμβου υποδοχής για την τροποποίηση των ιδιοτήτων του κόμβου υποδοχής.



Εικόνα 17: Διαμόρφωση ιδιοτήτων κεντρικού υπολογιστή, όνομα κεντρικού υπολογιστή, διεύθυνση IP, ποσότητα CPU, ρύθμιση πυρήνων.

Με τη διαμόρφωση των ιδιοτήτων του κόμβου κεντρικού υπολογιστή, μπορούμε να αλλάξουμε το όνομα κεντρικού υπολογιστή όπως (h1, h2, τοπικός κόμβος κ.λπ.), τη διεύθυνση IP των κόμβων κεντρικού υπολογιστή, την προεπιλεγμένη διαδρομή και το ποσοστό CPU και τους πυρήνες που χρησιμοποιούνται για τον κόμβο υπολογιστή. Επιπλέον, για τον κόμβο κεντρικού υπολογιστή μπορούν επίσης να οριστούν οι διεπαφές VLAN, οι εξωτερικές διεπαφές και οι ιδιωτικοί κατάλογοι.

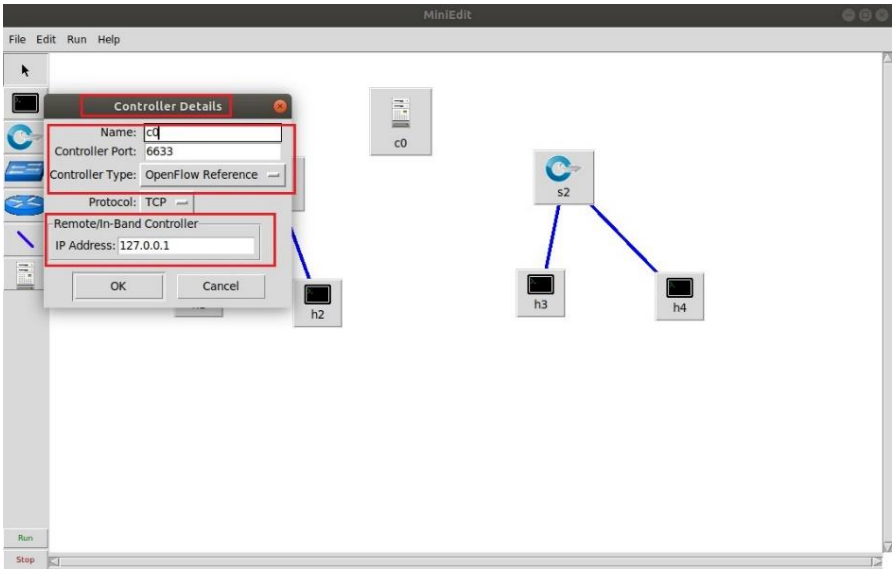
Στο ακόλουθο σχήμα απεικονίζεται η διαμόρφωση των ιδιοτήτων του συνδέσμου, η οποία μπορεί να πραγματοποιηθεί κάνοντας δεξί κλικ στο σύνδεσμο και κάνοντας κλικ στην επιλογή ιδιοτήτων.



Εικόνα 18: Διαμόρφωση ιδιοτήτων σύνδεσης, εύρος ζώνης, καθυστέρηση, απώλεια, μέγιστο μέγεθος ουράς και ρύθμιση Jitter.

Χρησιμοποιώντας την επιλογή διαμόρφωσης ιδιοτήτων συνδέσμου, το εύρος ζώνης, η καθυστέρηση, η απώλεια, το μέγιστο μέγεθος ουράς, το τρεμόπαιγμα (jitter) για συνδέσμους και η επιτάχυνση συνδέσμων μπορούν να διαμορφωθούν σύμφωνα με τις απαιτήσεις. Χρησιμοποιώντας τις ίδιες μεθόδους, η διαμόρφωση του ελεγκτή SDN μπορεί να πραγματοποιηθεί κάνοντας δεξί κλικ στον ελεγκτή SDN και

κάνοντας κλικ στην επιλογή ιδιοτήτων. Η επιλογή Διαμόρφωση ιδιοτήτων SDN θα διευκολύνει την τροποποίηση διαφόρων ιδιοτήτων του ελεγκτή SDN.

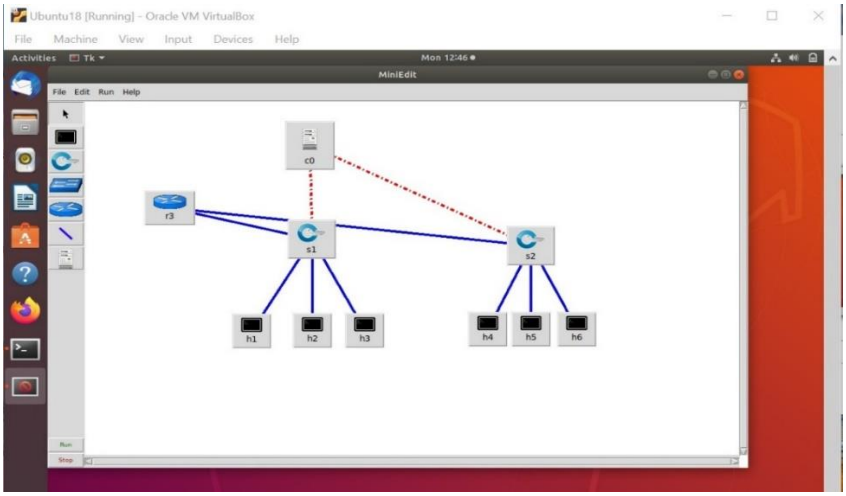


Εικόνα 19: Ρύθμιση παραμέτρων ελεγκτή SDN.

Χρησιμοποιώντας τα παράθυρα διαμόρφωσης SDN, μπορούν να ρυθμιστούν το όνομα ελεγκτή SDN, η θύρα ελεγκτή, ο τύπος ελεγκτή SDN, το πρωτόκολλο που χρησιμοποιείται από τον ελεγκτή SDN και η διεύθυνση IP για τον απομακρυσμένο ελεγκτή SDN.

4.4 Παράδειγμα πλήρους τοπολογίας δικτύου:

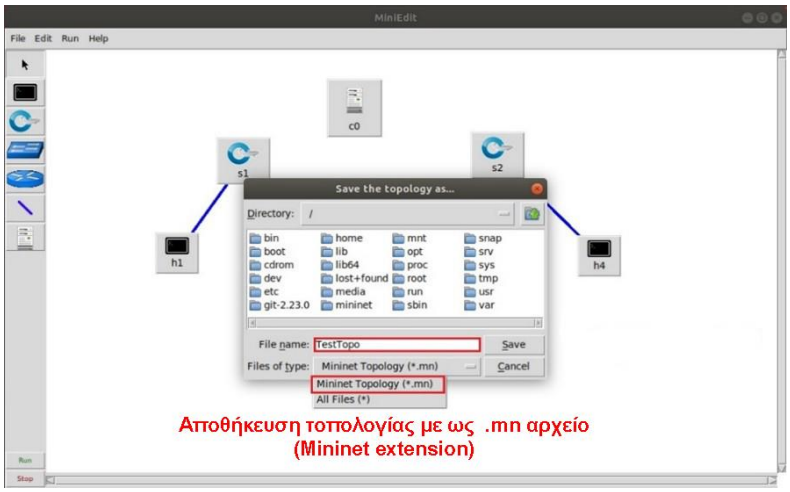
Στο ακόλουθο σχήμα απεικονίζεται μια πλήρης τοπολογία δικτύου που δημιουργήθηκε με τη χρήση του Miniedit και περιέχει 6 κόμβους υπολογιστών, δύο μεταγωγείς OVS που συνδέονται με έναν δρομολογητή δικτύου Legacy και επίσης συνδέονται απευθείας με τον απομακρυσμένο ελεγκτή SDN.



Σχήμα 20: Παραδείγματα τοπολογίας δικτύου.

Μετά την ολοκλήρωση του σχεδιασμού της τοπολογίας δικτύου, η τοπολογία μπορεί να αποθηκευτεί με την επέκταση .mn (Mininet Extension) ως Mininet

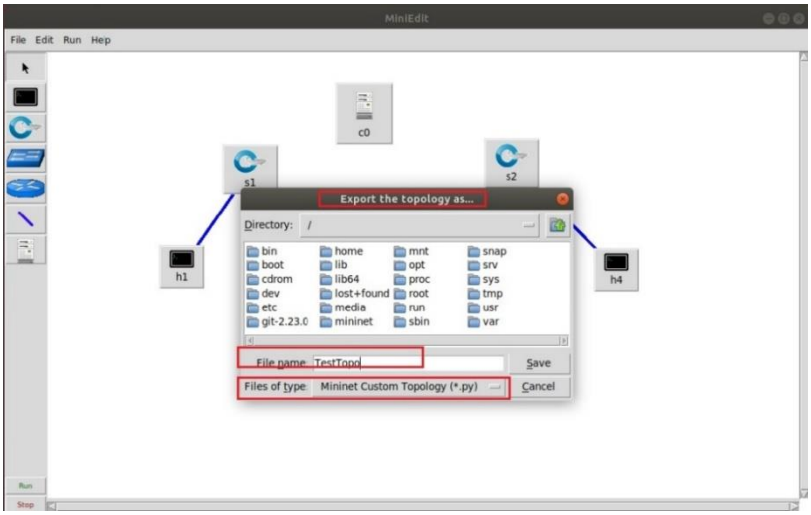
Network Topology ή με την επέκταση .py ως Python Script Code. Και οι δύο μορφοποιήσεις .mn και .py για τη σχεδιασμένη τοπολογία δικτύου παρουσιάζονται στην ακόλουθη εικόνα. Για την αποθήκευση της τοπολογίας δικτύου με επέκταση .mn, κάνουμε κλικ στην επιλογή αρχείο, και μετά κλικ στην επιλογή αποθήκευση και δίνουμε το όνομα της τοπολογίας και αποθηκεύουμε το αρχείο.



Εικόνα 21: Αποθήκευση τοπολογίας δικτύου με χρήση της επέκτασης .mn

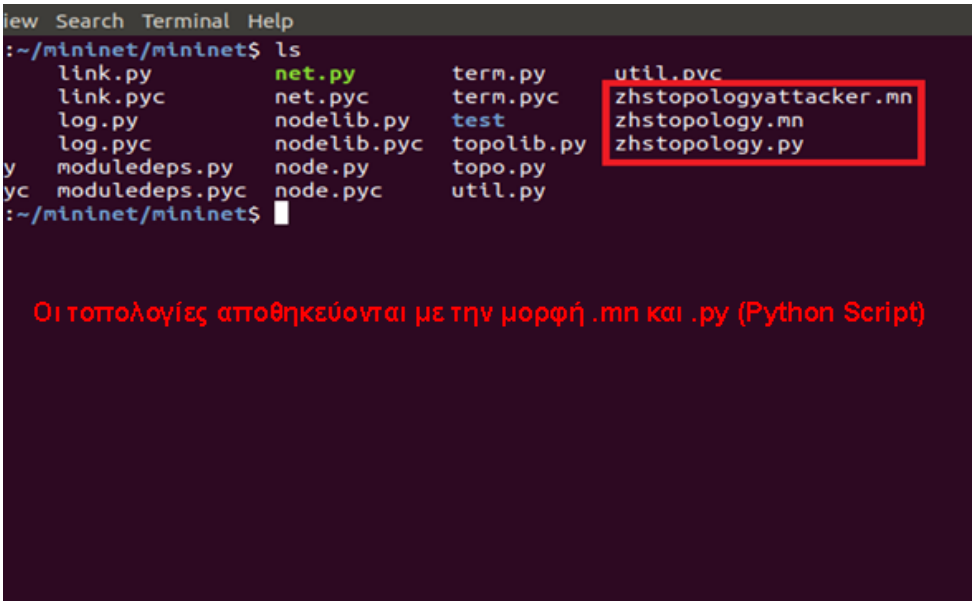
Το παραπάνω σχήμα δείχνει πώς μπορούμε να αποθηκεύσουμε τη σχεδιασμένη τοπολογία δικτύου με επέκταση .mn (Mininet) και να την εκτελέσουμε χρησιμοποιώντας την εντολή mn στο Mininet.

Η τοπολογία δικτύου μπορεί επίσης να αποθηκευτεί ως .py χρησιμοποιώντας το python API, για την αποθήκευση σε μορφή κώδικα python, κάνοντας κλικ στην επιλογή αρχείου και επιλέγοντας το «exportlevel 2 script» και αφού δώσουμε το όνομα της τοπολογίας αποθηκεύεται με την επέκταση .py.



Εικόνα 22: Αποθήκευση τοπολογίας δικτύου με την επέκταση .py

Μετά την αποθήκευση της τοπολογίας δικτύου με επέκταση .mn και .py, τα αρχεία αποθηκεύονται και εμφανίζονται στην παρακάτω εικόνα με τα ονόματα αρχείων zhstopology.mn και zhstopology.py.



Εικόνα 23: Τοπολογίες αποθηκευμένες με επεκτάσεις .mn και .py

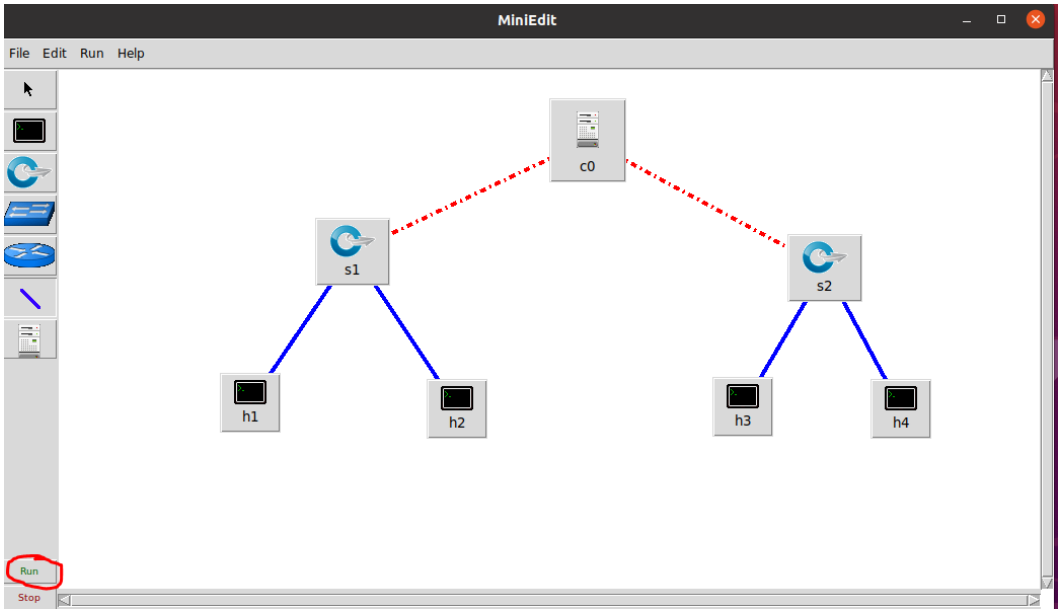
Μετά την επιτυχή ολοκλήρωση του σχεδιασμού της τοπολογίας, η τοπολογία του δικτύου αποθηκεύεται ως αρχείο Mininet (.mn) και σενάριο python (.py). Η αποθηκευμένη τοπολογία μπορεί να εκτελεστεί χρησιμοποιώντας το Miniedit (Mininet Topology Editor) ή μέσω της εντολής mn για την εκκίνηση της διεπαφής γραμμής εντολών Mininet (CLI).

4.5 Εκτέλεση τοπολογίας δικτύου με χρήση του Miniedit

Οι τοπολογίες δικτύου μπορούν να εκτελεστούν με τη χρήση του miniedit και της εντολής «sudo mn». Οι τοπολογίες δικτύου που έχουν αποθηκευτεί με την επέκταση .mn μπορούν να εκτελεστούν με τη χρήση του miniedit και αναφέρονται στο ακόλουθο σχήμα.

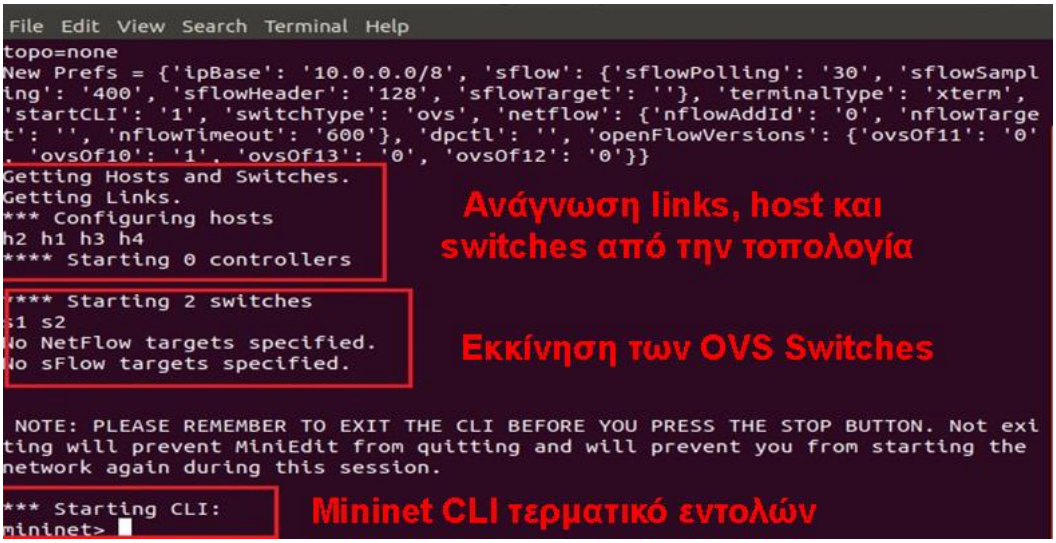
Παράδειγμα απλής τοπολογίας που εκτελείται με το miniedit

Για την εκτέλεση μιας τοπολογίας δικτύου με τη χρήση του Miniedit, απαιτείται κλικ στην επιλογή αρχείο, άνοιγμα και στη συνέχεια επιλογή του αρχείου τοπολογίας δικτύου που έχει αποθηκευτεί με επέκταση .mn, και τέλος κλικ στο ok. Η τοπολογία δικτύου θα ανοίξει στο Miniedit. Στο αριστερό κάτω μέρος της οθόνης του Miniedit υπάρχουν δύο κουμπιά run και stop. Επιλέγοντας κλικ στο κουμπί run θα ξεκινήσει η εκτέλεση της τοπολογίας δικτύου.



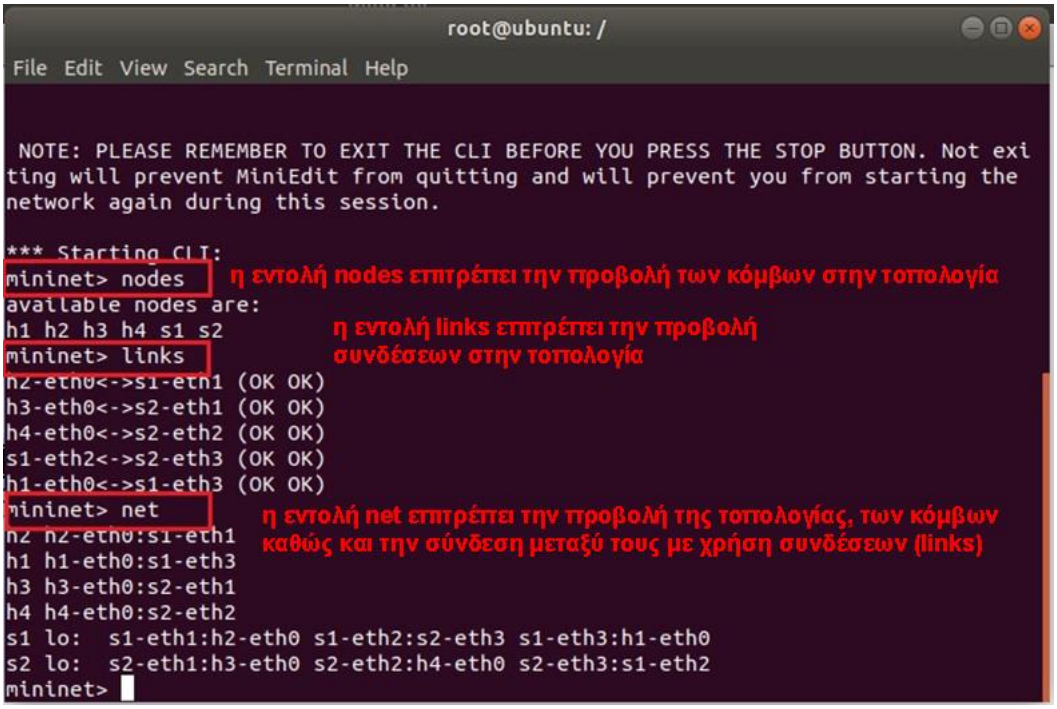
Εικόνα 24: Τοπολογία δικτύου σε λειτουργία

Αφού πραγματοποιηθεί κλικ στο κουμπί Εκτέλεση, η τοπολογία του δικτύου θα είναι σε θέση λειτουργίας και θα ξεκινήσει επίσης μια διεπαφή γραμμής εντολών στο τερματικό Mininet. Σε αυτό το τερματικό CLI μπορούν να εκτελεστούν εντολές που σχετίζονται με την τοπολογία δικτύου και τα αποτελέσματα θα εμφανιστούν επίσης εκεί. Το τερματικό Mininet CLI απεικονίζεται στην παρακάτω εικόνα.



Εικόνα 25: Εκτέλεση τοπολογίας δικτύου στη γραμμή εντολών του Mininet

Μερικά παραδείγματα εντολών απεικονίζονται παρακάτω.



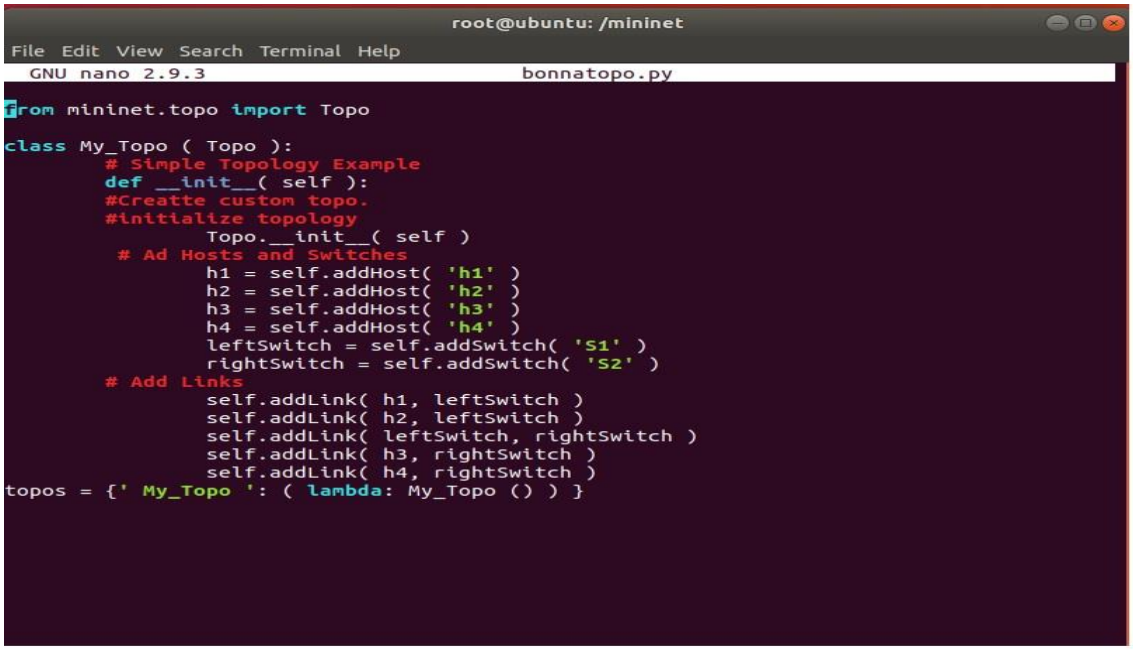
Εικόνα 26: Εμφάνιση εξόδου εντολών κόμβων, συνδέσμων και δικτύου στη γραμμή εντολών

4.6 Τοπολογία δικτύου με χρήση Python API (κωδικοποίηση Python)

Ο δεύτερος τρόπος για τη δημιουργία τοπολογίας δικτύου είναι ο προγραμματισμός με την γλώσσα Python . Χρησιμοποιώντας αυτόν τον τρόπο δημιουργούμε τοπολογίες δικτύων με τη χρήση της γλώσσας αυτής. Η προεπιλεγμένη τοπολογία που μπορεί να εκτελεστεί χωρίς τη δημιουργία τοπολογίας Mininet είναι η ελάχιστη τοπολογία δικτύου στο Mininet που περιέχει ένα Open Virtual Switch (OVS) και δύο κόμβους υποδοχής. Σε αυτή την ελάχιστη τοπολογία δικτύου δύο υπολογιστές-ξενιστές είναι συνδεδεμένοι με έναν μεταγωγέα OVS και ο μεταγωγέας OVs συνδέεται περαιτέρω με τον ελεγκτή SDN Open Flow Controller. Αυτή η τοπολογία μπορεί να εκτελεστεί και να τρέξει χρησιμοποιώντας την εντολή «sudomn» στο τερματικό του linux με την παράμετρο -topo=minimal.

Η γραμμή εντολών του Linux και το python API χρησιμοποιούνται κυρίως για τη δημιουργία προσαρμοσμένων τοπολογιών δικτύου. Αυτές οι προσαρμοσμένες τοπολογίες δικτύου μπορούν να δημιουργηθούν με τη χρήση προγραμματισμού python. Το Mininet βοηθά στη δημιουργία προσαρμοσμένων τοπολογιών σύμφωνα με τις απαιτήσεις του χρήστη με την υποστήριξη του Python API (Application Program Interface) όπου μπορούν να δημιουργηθούν με τη χρήση λίγων γραμμών κώδικα. Στο παρακάτω σχήμα δημιουργείται μια απλή προσαρμοσμένη τοπολογία με τη χρήση python API και κωδικοποίησης. Αυτή η τοπολογία αποτελείται από τέσσερις κόμβους υπολογιστών που συνδέονται με δύο μεταγωγείς OVS. Δύο κόμβοι συνδέονται με το S1 (OVS Switch) και δύο κόμβοι συνδέονται με το S2. Ο κώδικας

για τη δημιουργία προσαρμοσμένης τοπολογίας δικτύου παρουσιάζεται στο παρακάτω σχήμα.

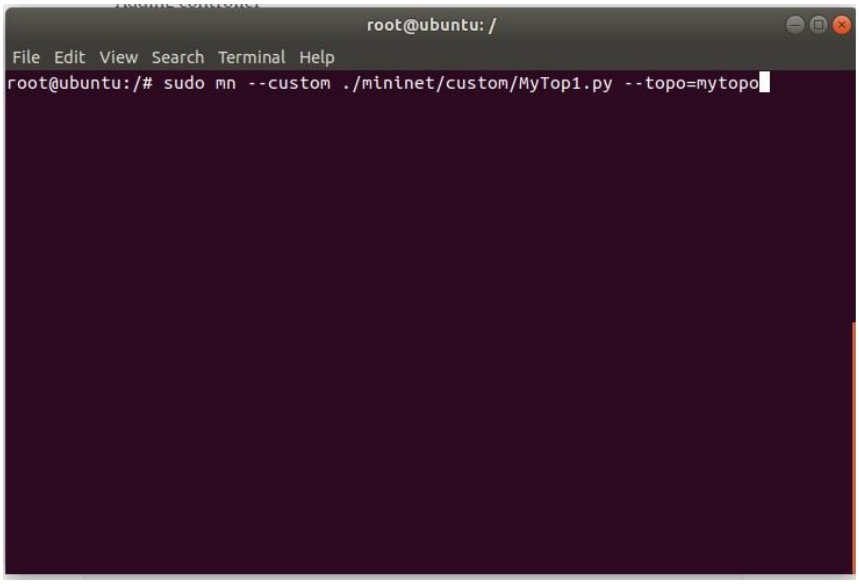


Εικόνα 27: Συγγραφή τοπολογίας με χρήση κώδικα Python

Στο παραπάνω σχήμα παρουσιάζεται ένας απλός κώδικας python που έχει γραφτεί για τη δημιουργία μιας απλής τοπολογίας δικτύου. Ο κώδικας δείχνει ότι τέσσερις κόμβοι υποδοχής h1, h2, h3 και h4 προστίθενται στην τοπολογία, δύο μεταγωγείς OVS S1 και S2 προστίθενται επίσης και για τη συνδεσιμότητα των κόμβων με τους μεταγωγείς OVS δημιουργούνται σύνδεσμοι. Οι κόμβοι H1 και H2 συνδέονται με τον S1 και οι δύο κόμβοι h3 και h4 συνδέονται με τον S2 (OVS Switch). Μετά τη δημιουργία του κώδικα για την τοπολογία δικτύου, η τοπολογία μπορεί να αποθηκευτεί ως κώδικας σεναρίου python με επέκταση .py.

Για τη συγγραφή κώδικα python για προσαρμοσμένη τοπολογία δικτύου μπορεί να χρησιμοποιηθεί οποιοσδήποτε επεξεργαστής κειμένου Linux. Αφού γραφτεί και αποθηκευτεί η προσαρμοσμένη τοπολογία δικτύου με κώδικα python, η τοπολογία μπορεί να εκτελεστεί με τη χρήση του «sudomn», όπως περιγράφεται επίσης παραπάνω. Η εντολή για την εκτέλεση του κώδικα που έχει αποθηκευτεί με επέκταση .py αναφέρεται παρακάτω.

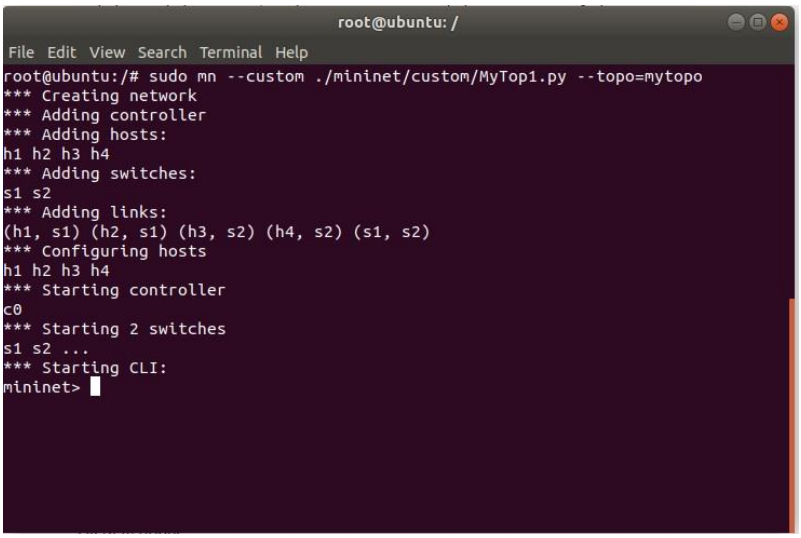
Εντολή: «Sudomn -custom ./mininet/custom/Mytop1.py -topo=mytopo»



Εικόνα 28: Εντολή για την εκτέλεση προσαρμοσμένης τοπολογίας δικτύου στο Mininet

Μετά την εκτέλεση της εντολής στο τερματικό CLI, η τοπολογία θα αρχίσει να εκτελείται στο τερματικό Linux και θα εμφανιστεί η προτροπή Mininet CLI που ενημερώνει ότι η τοπολογία δικτύου εκτελέστηκε με επιτυχία. Χρησιμοποιώντας τη διεπαφή γραμμής εντολών, μπορούν να περαστούν εντολές στο τερματικό της τοπολογίας δικτύου και τα αποτελέσματα θα εμφανιστούν επίσης στο τερματικό CLI.

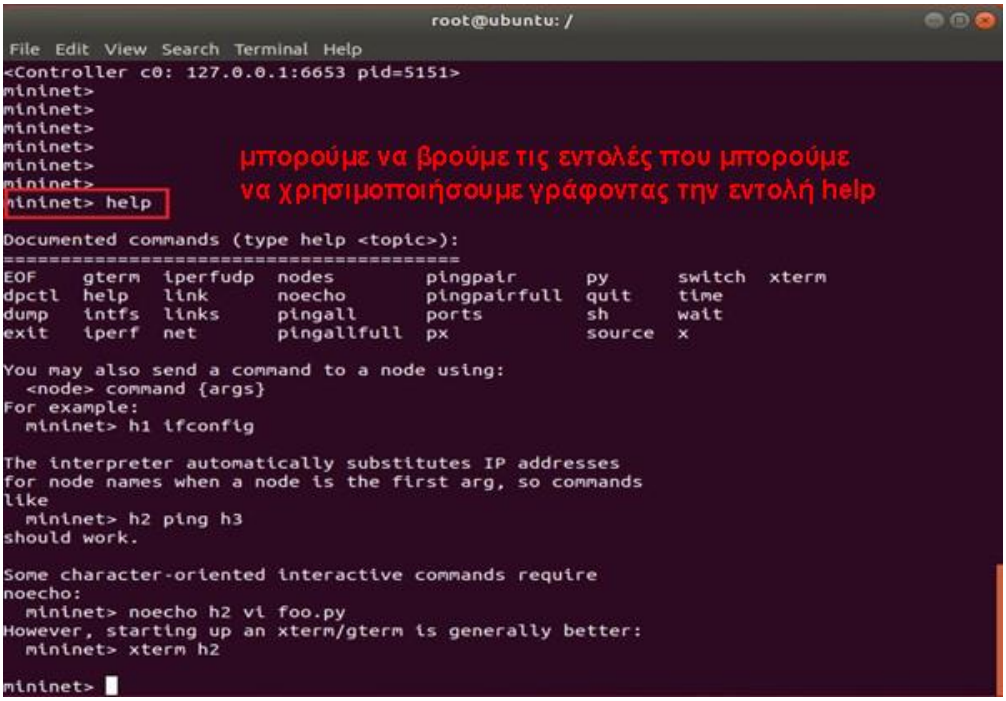
Το τερματικό εντολών του MininetCLi μετά την εκτέλεση της εντολής εμφανίζεται παρακάτω.



Εικόνα 29: Εκτέλεση προσαρμοσμένης τοπολογίας στη γραμμή εντολών του Mininet

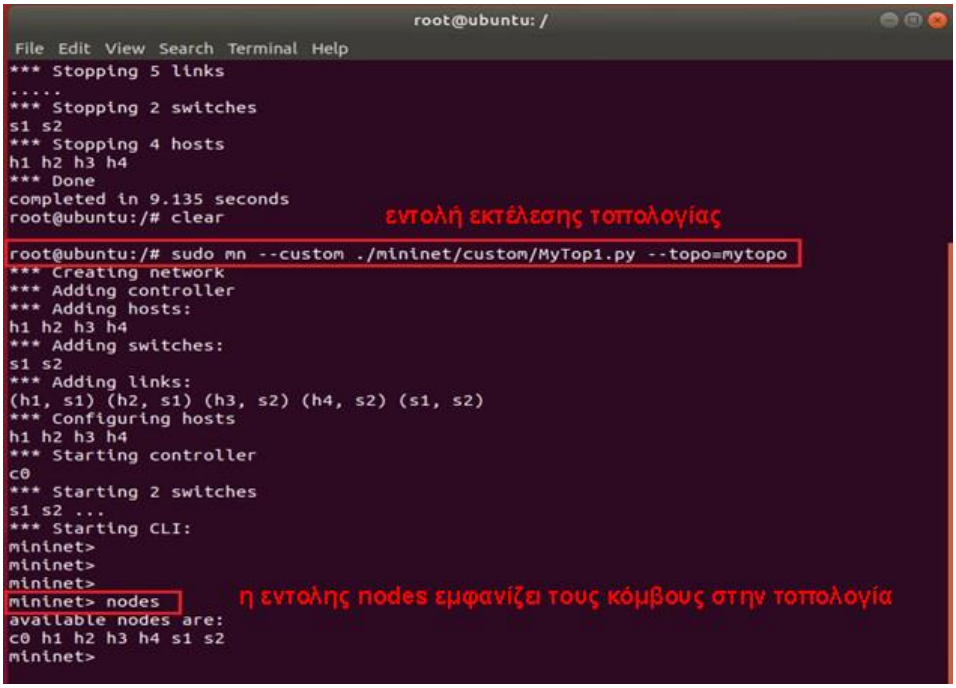
Σε αυτό το παράδειγμα προσαρμοσμένης τοπολογίας δικτύου, δημιουργούνται συνολικά έξι κεντρικοί υπολογιστές με δύο μεταγωγείς OVS και τρεις κόμβους, που συνδέονται με κάθε μεταγωγέα OVS χρησιμοποιώντας συνδέσεις δικτύου. Η εντολή help στο τερματικό Mininet CLI βοηθά για τις εντολές υποστήριξης του Mininet που μπορούν να εκτελεστούν στο τερματικό Mininet CLI. Ο κατάλογος των εντολών που

υποστηρίζονται από το Mininet και μπορούν να εκτελεστούν απεικονίζεται στο ακόλουθο σχήμα.



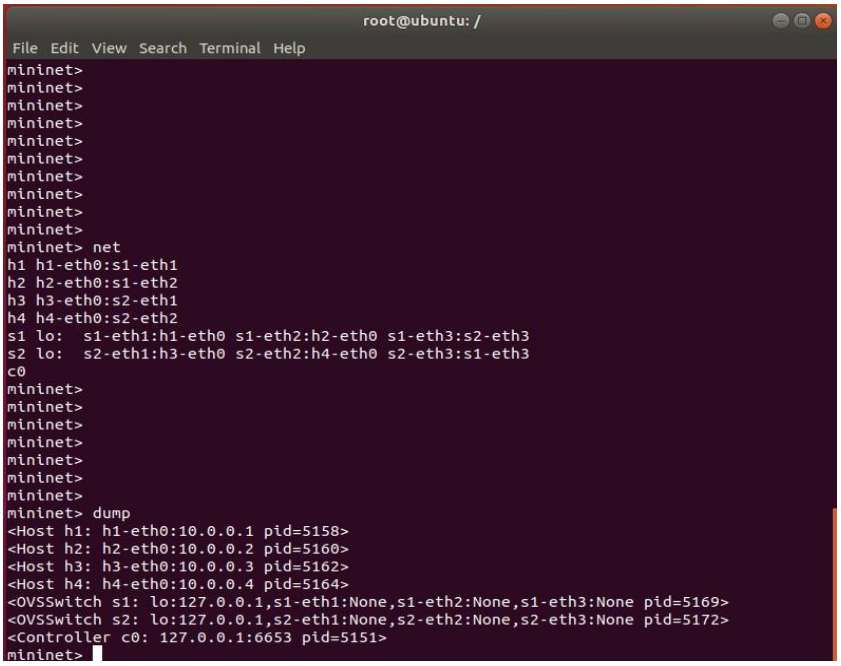
Εικόνα 30: Εντολή Help στο Mininet για την προβολή εντολών Mininet

Στο τερματικό CLI του Mininet μπορούν να εκτελεστούν διάφορες εντολές που σχετίζονται με την τοπολογία του δικτύου όπως περιγράφηκε προηγουμένως. Το παράδειγμα της εντολής «nodes» εκτελείται επίσης στο τερματικό CLI και παρουσιάζεται στο ακόλουθο σχήμα.



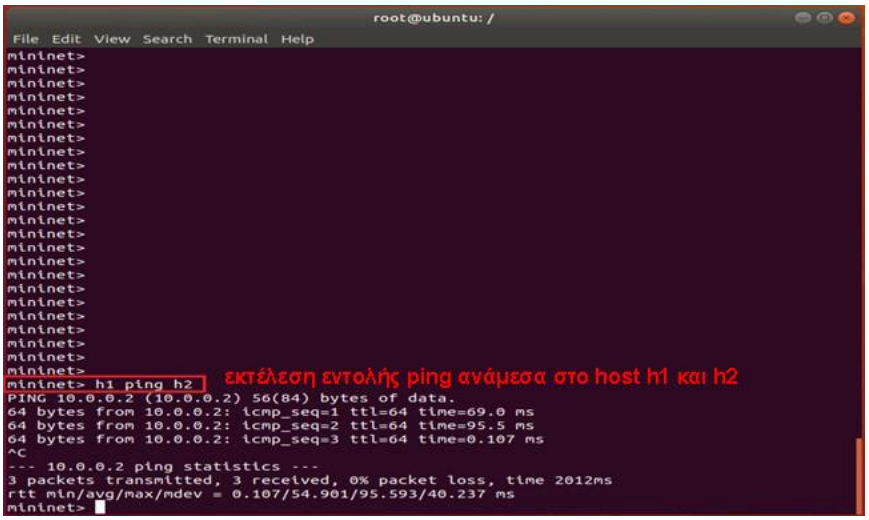
Εικόνα 31: Εντολή για την εκτέλεση της τοπολογίας δικτύου

Άλλες υποστηριζόμενες από το Mininet εντολές όπως, Net, Dumps, Ping, Nodes μπορούν επίσης να εκτελεστούν από το τερματικό μετά την έναρξη της προσαρμοσμένης τοπολογίας δικτύου.



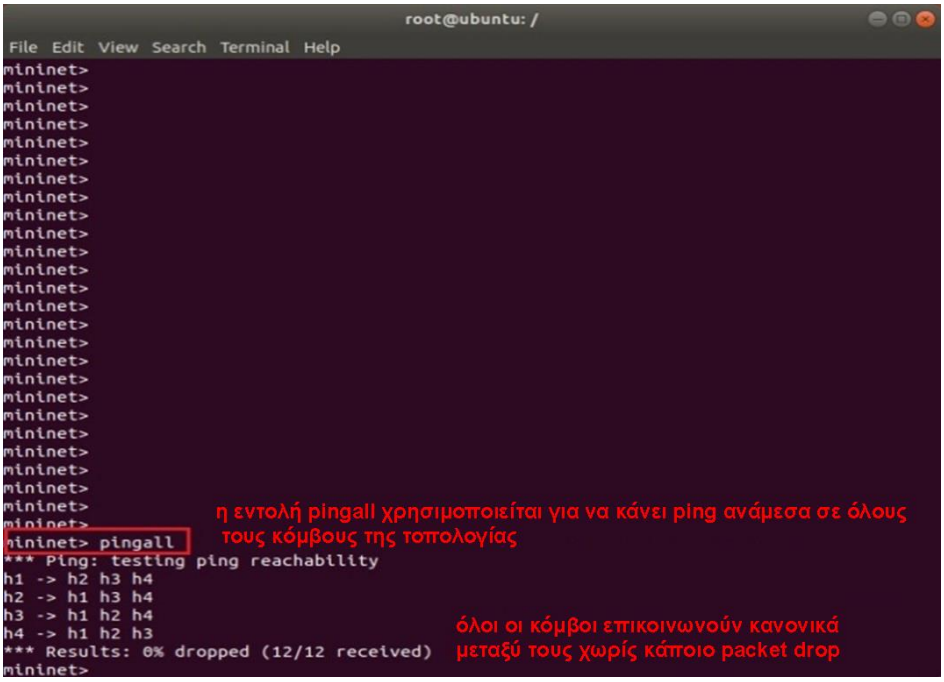
Εικόνα 32: Αποτελέσματα εξόδου των εντολών Net, Dump

Στο παραπάνω σχήμα απεικονίζονται οι εντολές net και dump που εκτελούνται στην εντολή Mininet CLI. Η εντολή net χρησιμοποιείται για την εμφάνιση της λεπτομερούς τοπολογίας του δικτύου και η εντολή dump χρησιμοποιείται για την εμφάνιση των λεπτομερειών σχετικά με τους κόμβους του δικτύου, συμπεριλαμβανομένων των διεπαφών δικτύου τους, της συνδεσιμότητας τους με άλλες θύρες και των διευθύνσεων IP τους.



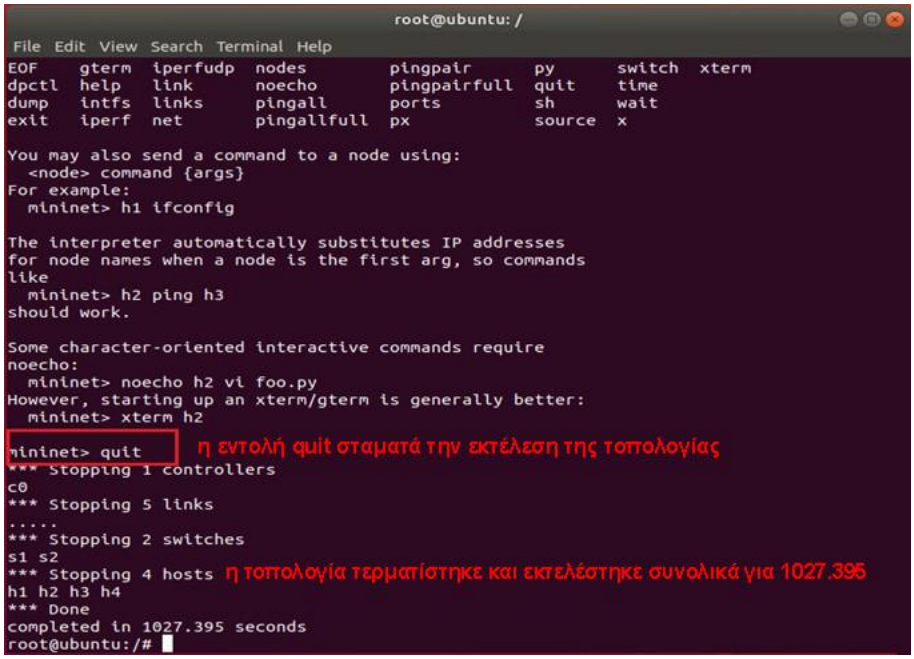
Εικόνα 33: Εντολή Ping για τους υπολογιστές h1 και h2 για τον έλεγχο της συνδεσιμότητας.

Η εντολή Ping που χρησιμοποιείται για τον έλεγχο της συνδεσιμότητας και της επικοινωνίας μεταξύ δύο συνδεδεμένων κόμβων μπορεί επίσης να εκτελεστεί από το τερματικό του Mininet CLI, όπως απεικονίζεται στο παραπάνω σχήμα. Η εντολή Ping στέλνει πακέτα ICM Pecho από τον κόμβο-πηγή στον κόμβο-προορισμό και λέει είτε οι δύο συνδεδεμένοι κόμβοι μπορούν να επικοινωνήσουν μεταξύ τους είτε όχι. Στο παραπάνω σχήμα η εντολή «h1 ping h2» δείχνει ότι ο κόμβος h1 μπορεί να κάνει ping στον κόμβο h2 και τα πακέτα αποστολής λαμβάνονται στους κόμβους προορισμού και επιβεβαιώνονται πίσω στον κόμβο πηγή. Τα στατιστικά αποτελέσματα ping δίνονται στο τέλος της εντολής ping και αφορούν τον RTT (Round TripTime) και δείχνουν το ελάχιστο, το μέσο, το μέγιστο και τη μέση απόκλιση του ρυθμού καθυστέρησης πακέτων.



Εικόνα 34: Αποτέλεσμα εξόδου εντολής Pingall

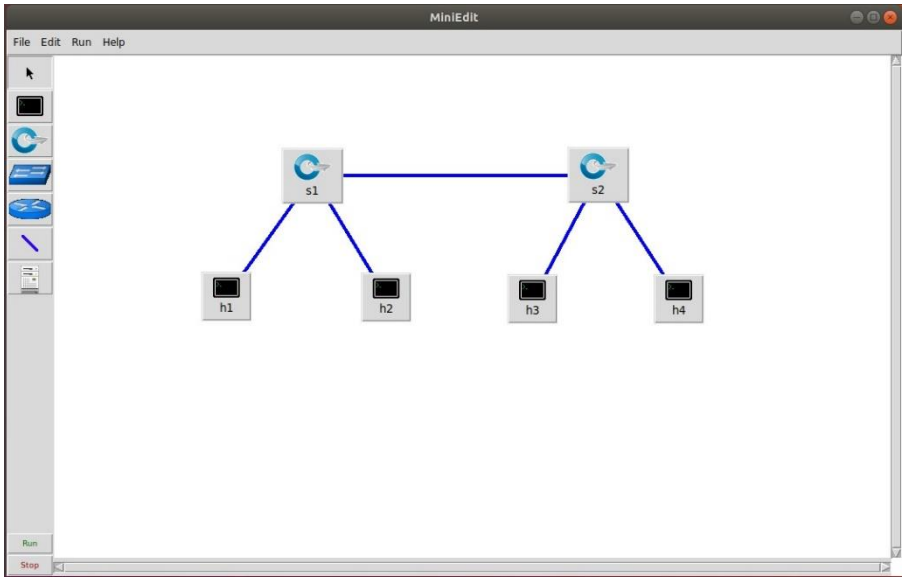
Η εντολή Ping All μπορεί να περάσει στο τερματικό Mininet CLI για τον έλεγχο της επικοινωνίας και της συνδεσιμότητας μεταξύ όλων των συνδεδεμένων κόμβων. Στο τέλος της εντολής ping, τα αποτελέσματα δείχνουν ότι συνολικά 12 πακέτα έχουν σταλεί και όλα τα πακέτα έχουν ληφθεί και κανένα πακέτο δεν έπεσε κατά τη διάρκεια της επικοινωνίας.



Εικόνα 35: Εντολή Quit για διακοπή και έξοδο από την εκτέλεση της τοπολογίας

Μετά την εκτέλεση της τοπολογίας δικτύου, ο τερματισμός γίνεται με την εντολή quit. Δίνοντας την εντολή αυτή, η εκτέλεση της τοπολογίας δικτύου θα σταματήσει και θα αναφερθεί πόσο χρόνο εκτελέστηκε συνολικά.

Η προαναφερθείσα προσαρμοσμένη τοπολογία δικτύου παρουσιάζεται σε γραφική μορφή στο ακόλουθο σχήμα.



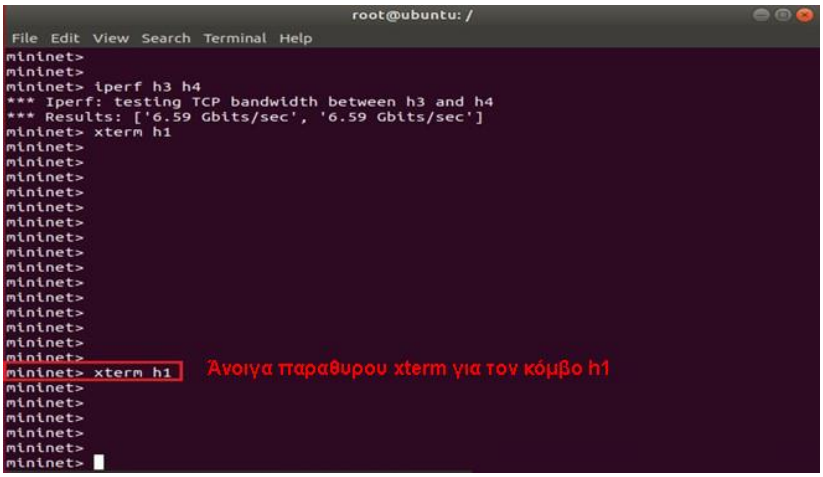
Εικόνα 36: Γραφική παρουσίαση της τοπολογίας του δικτύου στο Miniedit

.

4.7 Δοκιμές εύρους ζώνης, καθυστέρησης πακέτων, Jitter με τη χρήση του βοηθητικού προγράμματος Iperf.

Κόμβοι Xterm

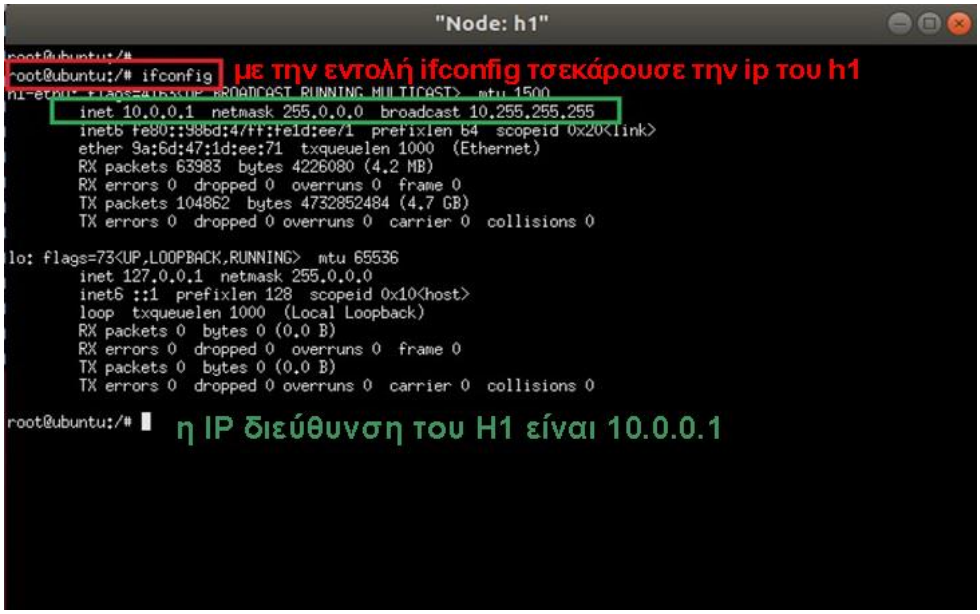
Το Xterm είναι ένα ενσωματωμένο βοηθητικό πρόγραμμα του Mininet που βοηθάει στη διαμόρφωση κάθε κόμβου σε ξεχωριστό τερματικό. Δίνοντας την εντολή xterm μπορείτε να ανοίξετε ξεχωριστό τερματικό για κάθε κόμβο και τον διακόπτη OVS για να τον ρυθμίσετε. Στο ακόλουθο σχήμα ανοίγει το παράθυρο xterm για τον κόμβο υποδοχής 1. Για το σκοπό αυτό χρησιμοποιείται η εντολή «xterm h1» που αναφέρεται στο παρακάτω σχήμα.



Εικόνα 37: Παραγωγή κίνησης με χρήση της εντολής IperfUtility/Xterm

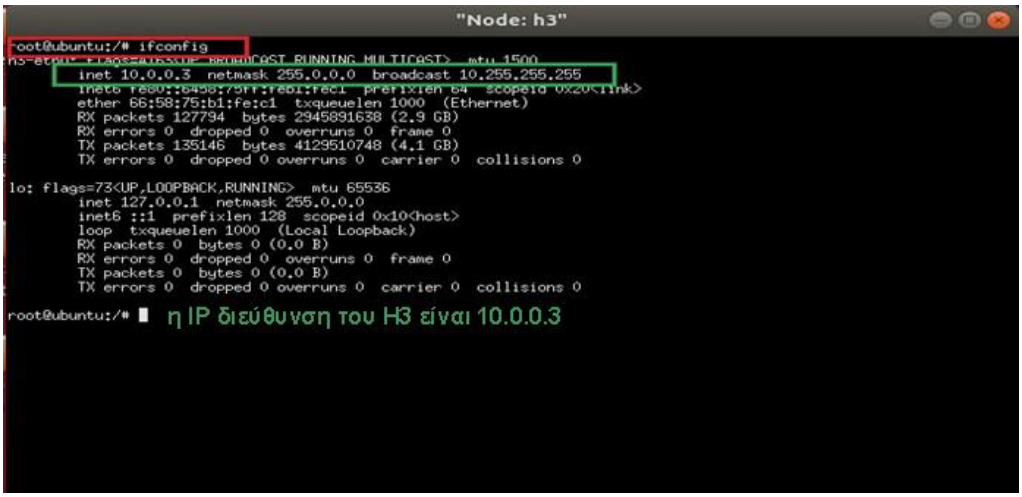
Εμφάνιση της διεύθυνσης IP των κεντρικών υπολογιστών με χρήση του Xterm

Στα παράθυρα xterm για τον καθορισμένο κόμβο, μπορούν να εκτελεστούν διάφορες εντολές για τη διαμόρφωση και τη διαχείριση των κόμβων. Για την εμφάνιση της διεύθυνσης IP του κόμβου, στα παράθυρα xterm εκτελείται η εντολή «ifconfig», η οποία θα παρέχει πληροφορίες σχετικά με τη διαμόρφωση της διασύνδεσης του κεντρικού υπολογιστή «node1» που περιλαμβάνει τη διεύθυνση IP, τη μάσκα υποδικτύου, τη διεύθυνση εκπομπής, τη διεύθυνση MAC των κόμβων, το εισερχόμενο πακέτο, τα αποσταλμένα πακέτα κ.λπ.



Εικόνα 38: Εντολή ifconfig για έλεγχο διαμόρφωσης διεύθυνσης IP κεντρικού υπολογιστή.

Ανοίγοντας το τερματικό Xterm, μπορεί να χρησιμοποιηθεί η εντολή ifconfig για να παρουσιαστεί η διεύθυνση IP του κόμβου h1. Η διεύθυνση IP του κόμβου 1 είναι 10.0.0.1, όπως αναφέρεται στην παραπάνω εικόνα.



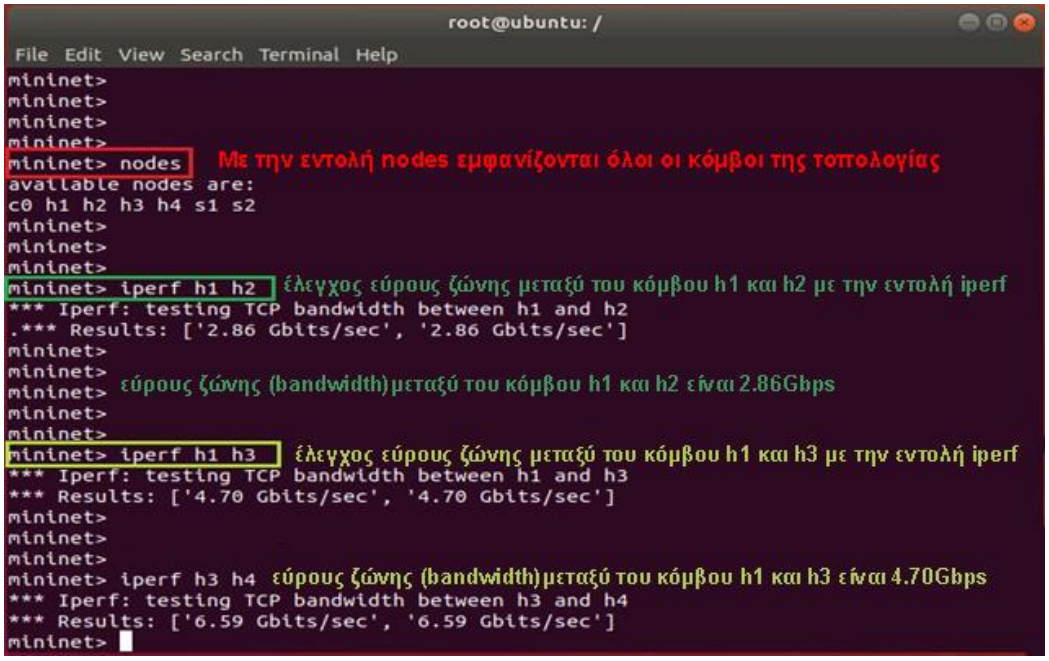
Σχήμα 39: αποτέλεσμα εξόδου της εντολής ifconfig.

Αν η εντολή Ifconfig εκτελεστεί στο Xterm για την προβολή της διεύθυνσης IP του h3, η διεύθυνση IP του h3 είναι 10.0.0.3.

4.8 Δοκιμή εύρους ζώνης: (χωρίς πελάτη/εξυπηρετητή με χρήση του βοηθητικού προγράμματος IPERF

Το Iperf είναι ένα ενσωματωμένο διαθέσιμο βοηθητικό πρόγραμμα στο Mininet που βοηθάει στην παραγωγή διαφόρων τύπων κίνησης. Το Iperf υποστηρίζει τη δημιουργία πακέτων κίνησης TCP, UDP και ICMP μεταξύ δύο συνδεδεμένων κόμβων. Άλλα διαθέσιμα βοηθητικά προγράμματα δημιουργίας κίνησης περιλαμβάνουν το hping3 και τη συμβατική εντολή ping. Η εντολή ping επιτρέπει μόνο τη δημιουργία πακέτων ICM echo, ενώ το Iperf επιτρέπει τη δημιουργία

κίνησης διαφόρων τύπων πακέτων, όπως TCP, UDP και ICMP, καθώς και τη δημιουργία άλλων προσαρμοσμένων πακέτων κίνησης. Το Iperf στο Mininet χρησιμοποιείται επίσης για τη δοκιμή του εύρους ζώνης της σύνδεσης που συνδέεται μεταξύ δύο κόμβων. Στο ακόλουθο σχήμα, το Iperf χρησιμοποιείται για τη δοκιμή του εύρους ζώνης μεταξύ των κόμβων h1 και h2 και μεταξύ των h1 και h3, όπου οι h1 και h2 συνδέονται με τους ίδιους μεταγωγείςOVS ενώ οι h1 και h3 συνδέονται με διαφορετικούς μεταγωγείςOVS.

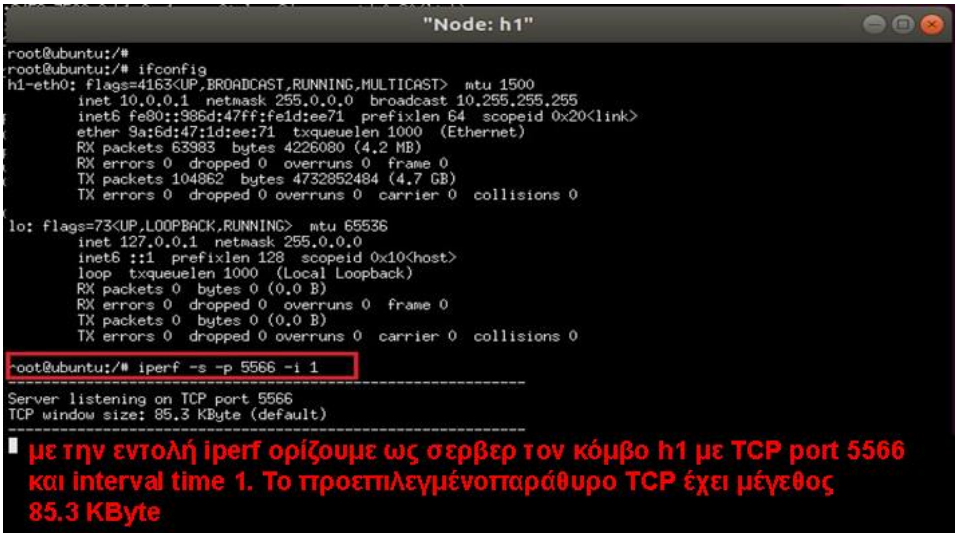


Εικόνα 40: Δοκιμή εύρους ζώνης με χρήση της εντολής Iperf μεταξύ δύο κεντρικών υπολογιστών.

Όπως στο προαναφερθέν στιγμιότυπο, η εντολή Iperf χρησιμοποιήθηκε για τη δοκιμή των κόμβων εύρους ζώνης μεταξύ h1 και h3, το εύρος ζώνης μεταξύ h1 και h2 παρατηρήθηκε 2,86 Gbps και το εύρος ζώνης μεταξύ h1 και h3 καταγράφεται ως 4,70 Gbps.

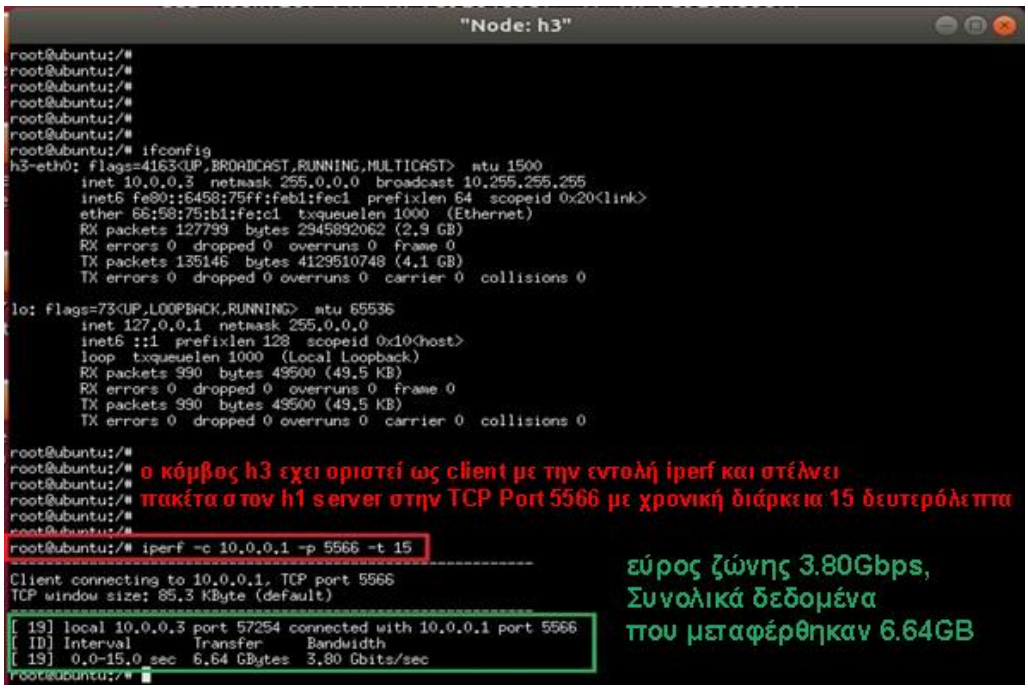
4.9 Δοκιμή εύρους ζώνης με διακομιστή-πελάτη (διακομιστής H1, πελάτης H3) με χρήση του βοηθητικού προγράμματος IPERF.

Για τη δοκιμή εύρους ζώνης μεταξύ των h1 και h3, ο h1 έχει ρυθμιστεί ως κόμβος διακομιστή και ο h3 ως κόμβος-πελάτης. Στο παρακάτω σχήμα απεικονίζεται πώς μπορεί να διαμορφωθεί ο κόμβος ως κόμβος διακομιστή.



Εικόνα 41: Εντολή iperf για τον ορισμό ενός κεντρικού υπολογιστή ως κόμβου διακομιστή.

Χρησιμοποιώντας την εντολή iperf «iperf -s -p 5566 -i 1», ο κόμβος h1 ορίζεται ως διακομιστής(server). Στις παραμέτρους της εντολής το «-s» χρησιμοποιείται για να δηλώσει ότι ο κόμβος θα λειτουργεί ως κόμβος διακομιστής. Η παράμετρος «-p» ορίζει τη θύρα TCP που θα χρησιμοποιείται για την ακρόαση των αιτήσεων του πελάτη από τον διακομιστή. Σε αυτή την περίπτωση χρησιμοποιείται η θύρα TCP 5566 για την ακρόαση των αιτημάτων του πελάτη. Η τελευταία παράμετρος «-i» περιγράφει ότι ο διακομιστής θα καταγράφει πακέτα με χρονικό διάστημα 1 δευτερόλεπτο.

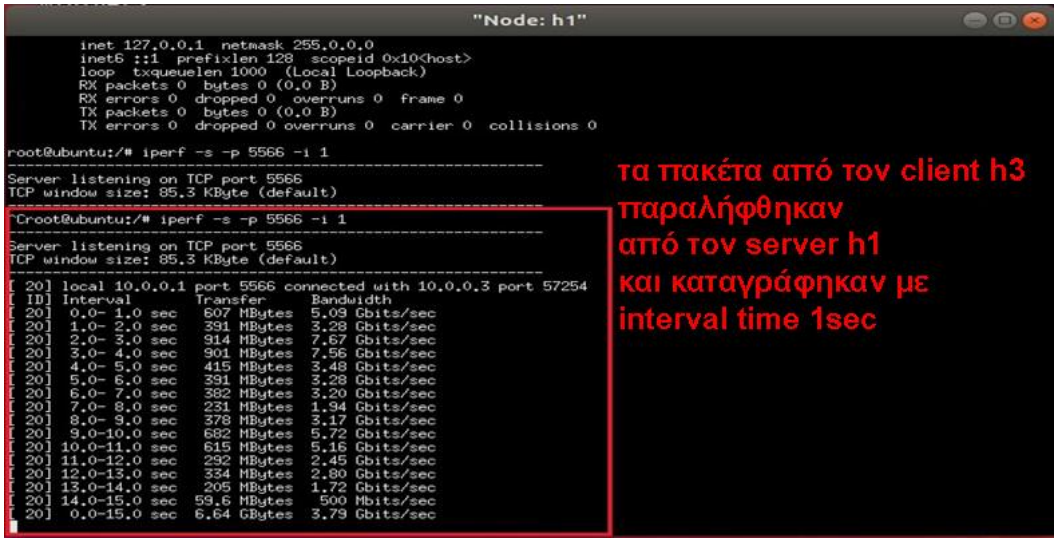


Εικόνα 42: Εντολή Iperf για τον ορισμό του κεντρικού υπολογιστή ως κόμβου-πελάτη.

Στο παραπάνω σχήμα, η εντολή Iperf χρησιμοποιείται στον κόμβο h3 για να τον ρυθμίσει ως κόμβο-πελάτη: iperf «iperf -c 10.0.0.1 -p 5566 -t 15». Η παράμετρος - «c» δείχνει ότι ο κόμβος έχει ρυθμιστεί ως κόμβος-πελάτης. Η παράμετρος «-c» δίπλα στην παράμετρο IP του κόμβου διακομιστή (h1, 10.0.0.1) στέλνει πακέτο TCP για να ελέγξει το εύρος ζώνης μεταξύ h1 και h3 ενώ η παράμετρος «-p» χρησιμοποιείται για να ορίσει τη θύρα TCP που έχει οριστεί για τον κόμβο-διακομιστή στην οποία ο κόμβος-διακομιστής (h1) θα ακούει τα αιτήματα του πελάτη.

Αφού δοθεί η εντολή στον κόμβο πελάτη, ο κόμβος πελάτη θα αρχίσει να στέλνει πακέτο στον κόμβο διακομιστή για χρόνο μετάδοσης 15 δευτερολέπτων και μετά από αυτό στον κόμβο πελάτη εμφανίζονται ο ρυθμός μεταφοράς και το εύρος ζώνης μεταξύ των κόμβων h1 (διακομιστής) και h3 (πελάτης). Σε αυτή την περίπτωση, ο ρυθμός μεταφοράς μεταξύ των h1 και h3 καταγράφεται ως 6,64 Gbps και το εύρος ζώνης παρατηρείται μέχρι 3,80 Gbps.

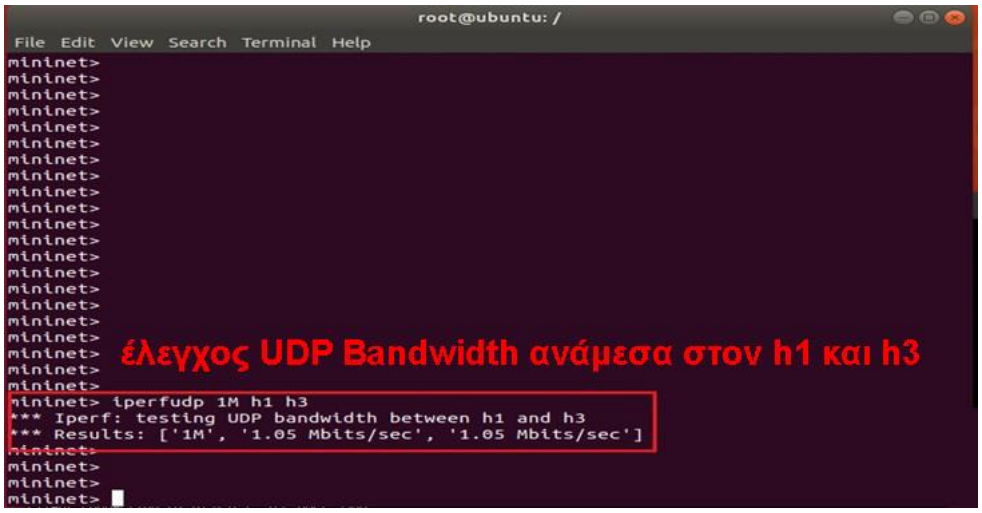
Στο παρακάτω σχήμα, τα πακέτα που καταγράφονται στον κόμβο h1 (Server Node) καταγράφονται με διάστημα 1 δευτερόλεπτο.



Σχήμα 43: Πακέτο που καταγράφηκε στο H1 (κόμβος διακομιστή), η θύρα έχει οριστεί σε 5566 με εσωτερικό 1

Ο h1 (κόμβος διακομιστή) λαμβάνει πακέτα που μεταδίδονται από τον κόμβο πελάτη h3 και καταγράφονται με χρονικό διάστημα 1 sec.

4.10 Δοκιμή εύρους ζώνης μεταξύ των κόμβων h1 και h3 με χρήση πακέτων UDP (δοκιμή Jitter).



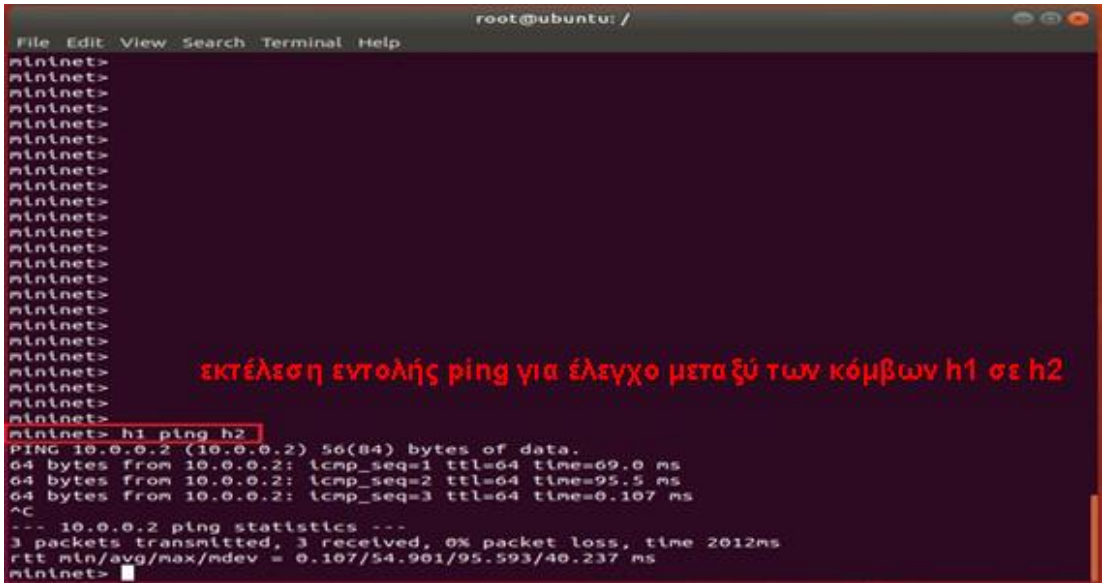
Σχήμα 44: Δοκιμή εύρους ζώνης με χρήση του πρωτοκόλλου UDP

Δοκιμή εύρους ζώνης UDP με χρήση της εντολής iperf μεταξύ h1 και h3 με σύνδεση 1Mbps, αυτή η εντολή Iperf UDP βοηθά στη δοκιμή του jitter για τις υπηρεσίες φωνής.

4.11 Εντολή PING για τη δοκιμή της καθυστέρησης μεταξύ κόμβων

Για τη δοκιμή του ρυθμού καθυστέρησης, των πακέτων που αποστέλλονται, των πακέτων που λαμβάνονται και του αριθμού των απορριφθέντων πακέτων μπορεί να χρησιμοποιηθεί η εντολή ping. Θα λέγαμε ότι η εντολή ping είναι ένα βοηθητικό πρόγραμμα παρακολούθησης της συνδεσιμότητας και της απόδοσης του δικτύου που βοηθά στην παρακολούθηση της υγείας του δικτύου.

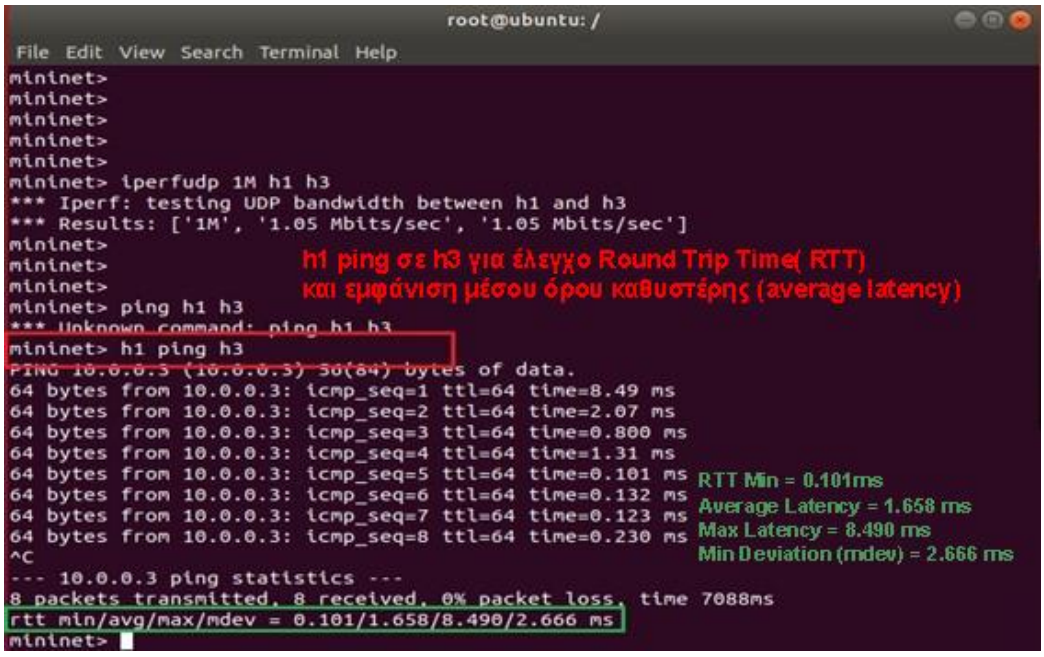
Στο ακόλουθο σχήμα, η εντολή ping εκτελείται μεταξύ των h1 και h2 για να ελέγξει τον RTT (Roung Trip Time) που καταναλώνει ένα πακέτο για να ταξιδέψει από τον κόμβο προέλευσης στον κόμβο προορισμού και να λάβει πίσω στον κόμβο προέλευσης και να ολοκληρώσει έναν γύρο.



Σχήμα 45: Εντολή Ping για τον έλεγχο της συνδεσιμότητας μεταξύ h1 και h2

Στο παραπάνω σχήμα, μετά την εκτέλεση της εντολής Ping για τον κόμβο h1 προς τον h2, το αποτέλεσμα δείχνει ότι τα πακέτα ICMP echo προέρχονται από τον h1 και αποστέλλονται στον h2 και η εντολή λαμβάνει ξανά ένα γύρο στον h1 για να ολοκληρωθεί το ταξίδι και τα τρία πακέτα αποστολής δείχνουν διαφορετικό RTT για κάθε πακέτο. Στο τέλος της εκτέλεσης της εντολής, εμφανίζονται τα στατιστικά στοιχεία ping που δείχνουν ότι τρία πακέτα ping αποστέλλονται από την πηγή στον κόμβο προορισμού, τρία πακέτα λαμβάνονται και δεν παρατηρείται απώλεια πακέτων.

4.12 Παρακολούθηση της καθυστέρησης και του RTT (Round TripTime) μεταξύ h1 και h3.



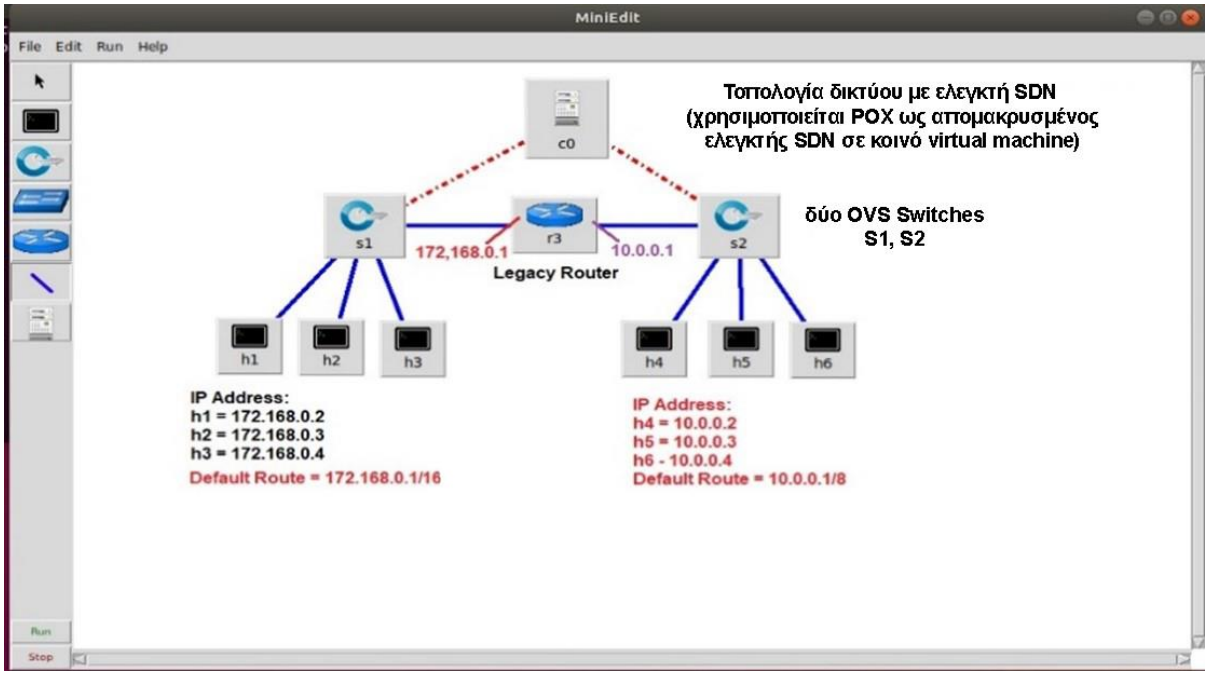
Εικόνα 46: Μέτρηση του χρόνου διαδρομής (RTT) μεταξύ h1 και h2

Στο παραπάνω σχήμα, η εντολή ping εκτελείται μεταξύ h1 και h3 και στο τέλος της εκτέλεσης της εντολής ping εμφανίζονται στατιστικά στοιχεία πακέτων. Στα στατιστικά στοιχεία πακέτων εμφανίζονται τα αποτελέσματα RTT (Round TripTime), Minimum, Average, Max και MinDeviation. Σύμφωνα με τα αποτελέσματα των στατιστικών πακέτων, το RTT Min (Minimum Latency Rate) καταγράφεται ως 0,101 δευτερόλεπτα, η μέση καθυστέρηση παρατηρείται ως 1,658 δευτερόλεπτα, η μέγιστη καθυστέρηση υπολογίζεται ως 8,490 δευτερόλεπτα και η μέση απόκλιση της καθυστέρησης καταγράφεται ως 2,66 ms.

Η καθυστέρηση πακέτων δείχνει πόσο γρήγορα ταξιδεύουν τα πακέτα δεδομένων στις συνδέσεις δικτύου και πόσο χρόνο χρειάζονται για να ταξιδέψουν από τον πελάτη στον διακομιστή και πίσω. Αυτό το ταξίδι από τον κόμβο προέλευσης στον προορισμό και πίσω στον κόμβο προέλευσης είναι επίσης γνωστό ως χρόνος επιστροφής. Αυτός ο χρόνος RTT (Round TripTime) ενός πακέτου και ο χρόνος που περνάει ένα πακέτο στη σύνδεση δικτύου ονομάζεται καθυστέρηση δικτύου. Η καθυστέρηση πακέτων εξαρτάται από τη φυσική απόσταση μεταξύ δύο κόμβων ή τη σύνδεση μεταξύ των κόμβων. Επιπλέον, συμπεραίνεται ότι η μέση καθυστέρηση μεταξύ των κόμβων h1 (διακομιστής) και h3 (πελάτης) παρατηρείται ως 1,658 δευτερόλεπτα.

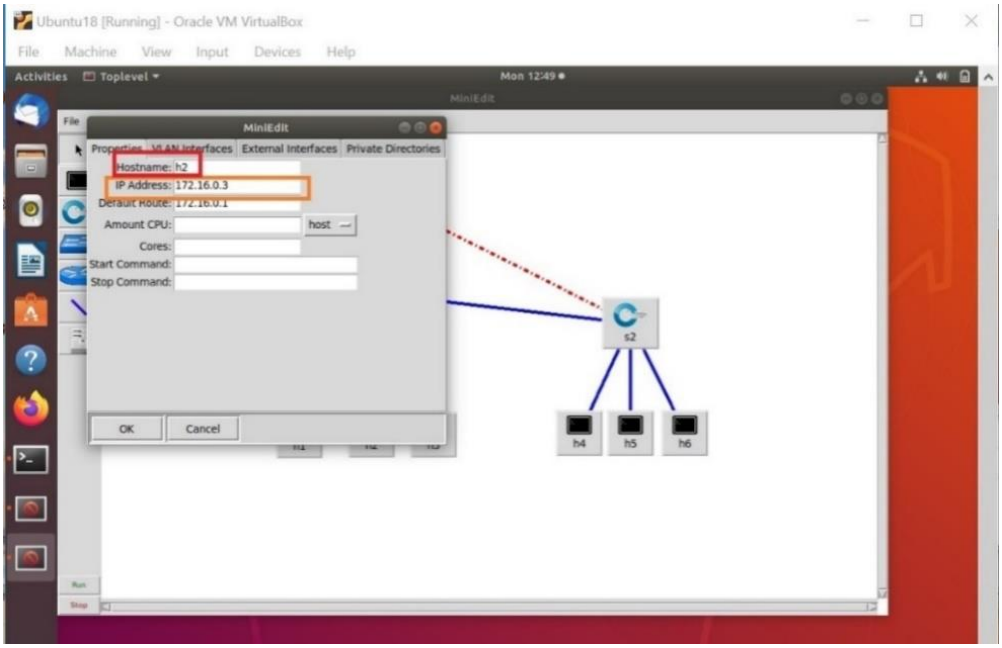
4.13 Παραγωγή και ανάλυση της κυκλοφορίας στο δίκτυο

Ο προσομοιωτής Mininet είναι εγκατεστημένος στα Windows 10, χρησιμοποιώντας περιβάλλον VM (VM Ware του Oracle Virtual Box). Ο σχεδιασμός της τοπολογίας δικτύου απεικονίζεται, όπως αναφέρθηκε στο επίπεδο 1, η τοπολογία δικτύου αποτελείται από έναν δρομολογητή (συσκευή επιπέδου 3), δυο μεταγωγείς ανοικτού V επιπέδου 2, έναν ελεγκτή SDN και τρεις κόμβους υποδοχής που συνδέονται με κάθε μεταγωγέα ανοικτού V. Επίσης θα γίνει χρήση ελεγκτή SDN (POX Controller). Η παρακάτω αναφερόμενη τοπολογία δικτύου με ελεγκτή SDN χρησιμοποιείται για την επίδειξη παραγωγής και ανάλυσης της κυκλοφορίας.



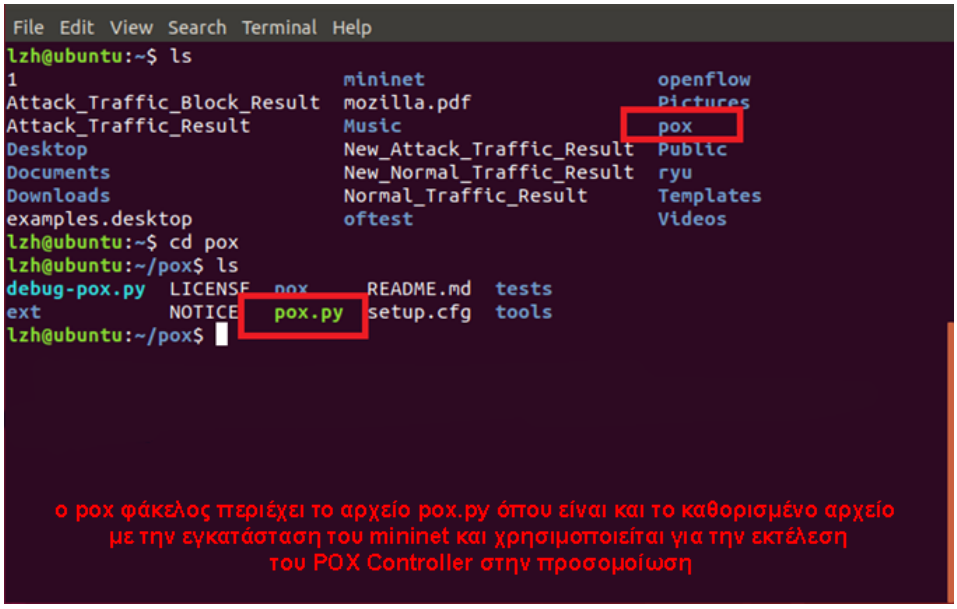
Σχήμα 47: Τοπολογία δικτύου για δοκιμές ασφαλείας με διευθύνσεις IP

Κόμβοι κεντρικών υπολογιστών συνδεδεμένοι με Open V Switches κατανεμημένοι σε δύο υποδίκτυα που χρησιμοποιούν δύο διαφορετικές κλάσεις διευθυνσιοδότησης IP 172.168.0.1/16 για το ένα υποδίκτυο και 10.0.0.0/8 για ένα άλλο υποδίκτυο. Και οι δύο μεταγωγείς Open V συνδέονται με έναν δρομολογητή με διεύθυνση IP που έχει εκχωρηθεί και στις δύο διεπαφές του R3 (δρομολογητής). Η μέθοδος διαμόρφωσης IP των κόμβων Hose απεικονίζεται στην παρακάτω εικόνα.



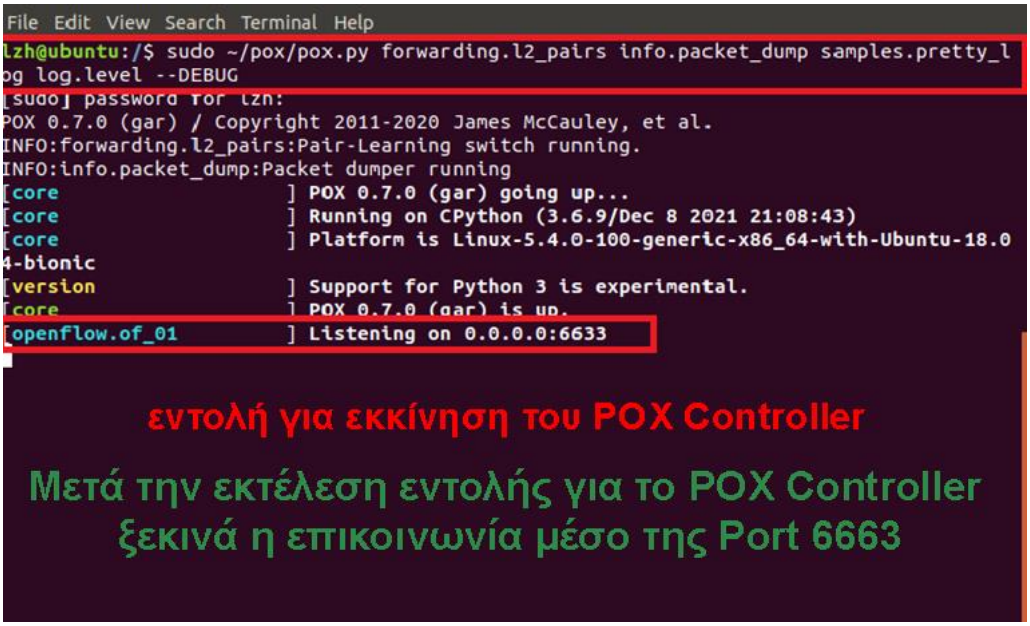
Εικόνα 48: Διαμόρφωση διεύθυνσης IP κεντρικού υπολογιστή (h2)

Η παραπάνω εικόνα δείχνει τη διαδρομή του ελεγκτή POX που είναι διαθέσιμος στο φάκελο εγκατάστασης Mininet Default.



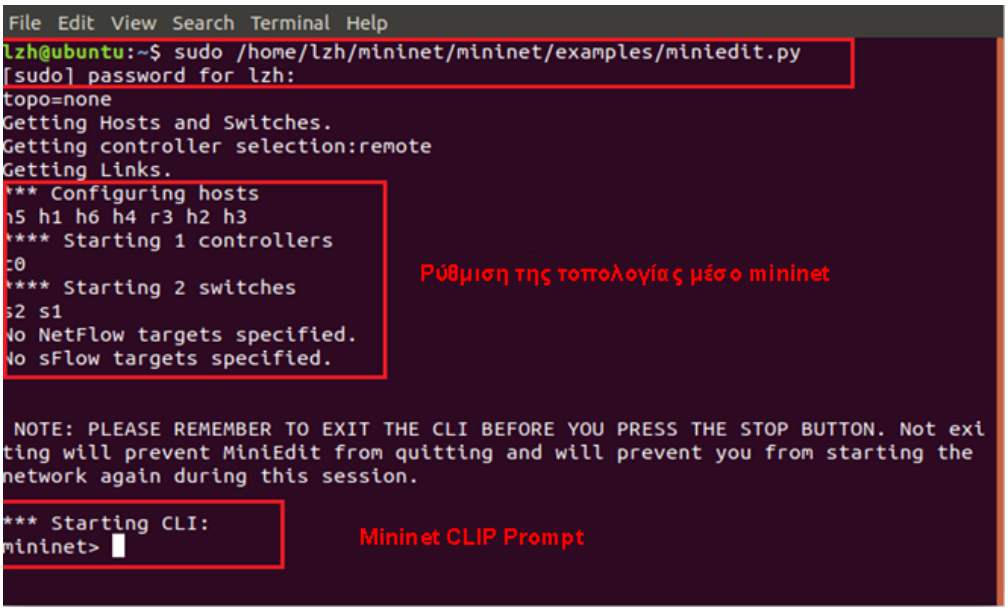
Εικόνα 49: Διαδρομή ελεγκτή POX SDN (ενσωματωμένο διαθέσιμο στο Mininet)

Μόλις γίνει σύνδεση με το Ubuntu χρησιμοποιώντας σύνοδο SSH, γίνεται εκκίνηση του POX SDN Controller και το σενάριο δικτύου Mininet που σχεδιάστηκε. Στα ακόλουθα σχήματα φαίνεται η εκτέλεση του POX SDN Controller και η εκτέλεση της τοπολογίας δικτύου στον επεξεργαστή τοπολογίας δικτύου Mininet (Miniedit).



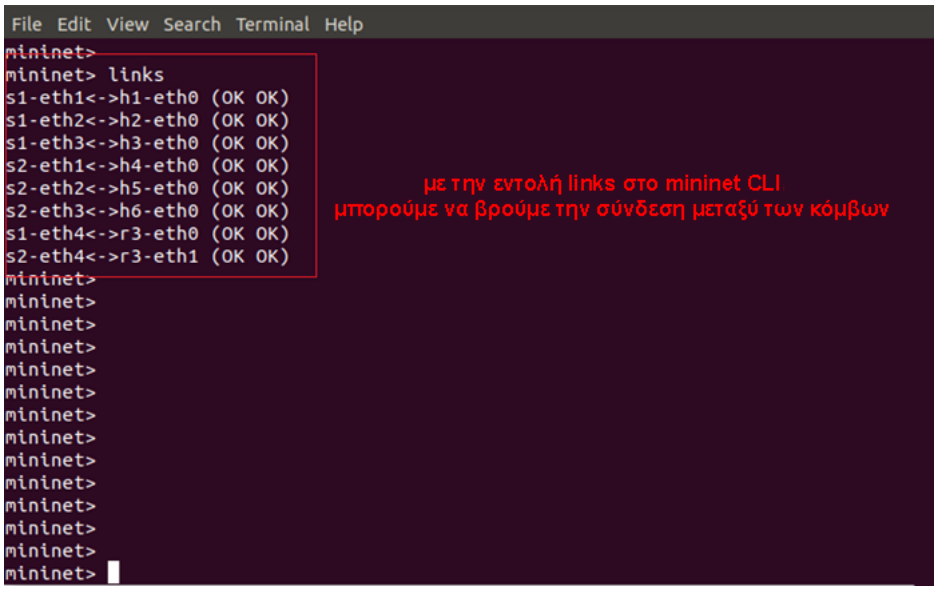
Εικόνα 50: Εντολή για την εκτέλεση του ελεγκτή POX

Μετά την εκτέλεση του POX SDN Controller, απαιτείται εκκίνηση της τοπολογίας δικτύου χρησιμοποιώντας το Miniedit. Με την εκκίνηση της τοπολογίας στο miniedit σε μια άλλη συνεδρία SSH, θα εμφανιστεί η προτροπή Mininet CLI που θα δείχνει ότι η τοπολογία δικτύου έχει ξεκινήσει με επιτυχία. Στην παρακάτω εικόνα, απεικονίζεται η προτροπή CLI της τοπολογίας δικτύου Mininet.



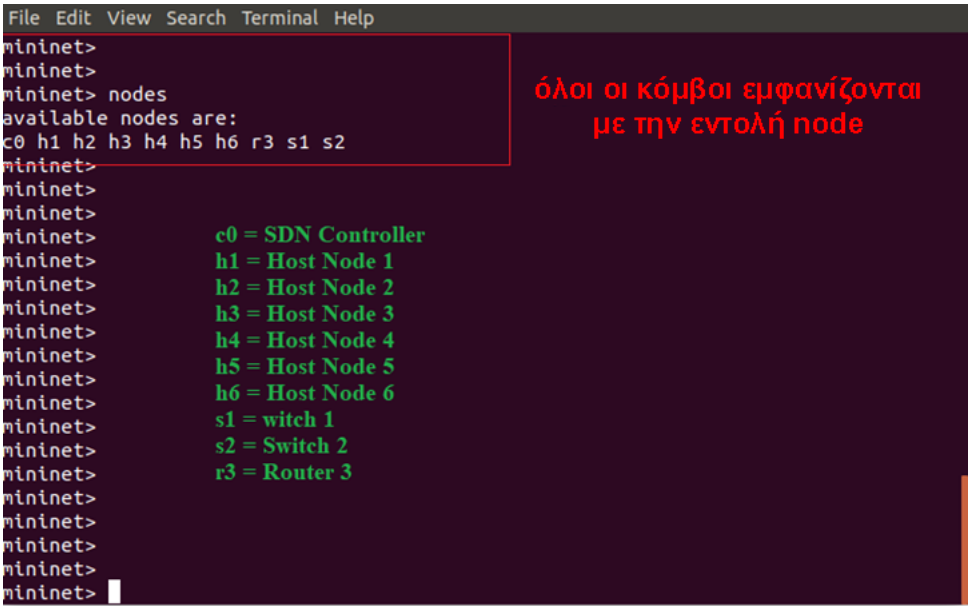
Εικόνα 51: Εκτέλεση τοπολογίας δικτύου και γραμμή εντολών Mininet.

Στο παρακάτω στιγμιότυπο οθόνης, φαίνεται ότι η τοπολογία δικτύου αποτελείται από κόμβους που έχουν σχεδιαστεί στο επίπεδο 1, υπάρχουν 6 κόμβοι υποδοχής, h1 έως h6, δύο μεταγωγείς ανοικτού V S1, S2, ένας δρομολογητής παλαιού τύπου r3 και ένας ελεγκτής SDN c0. Για την προβολή της συνδεσιμότητας των κόμβων στην τοπολογία δικτύου και των συνδέσμων στην τοπολογία δικτύου, μπορεί να χρησιμοποιηθεί η εντολή (Links and Nodes).



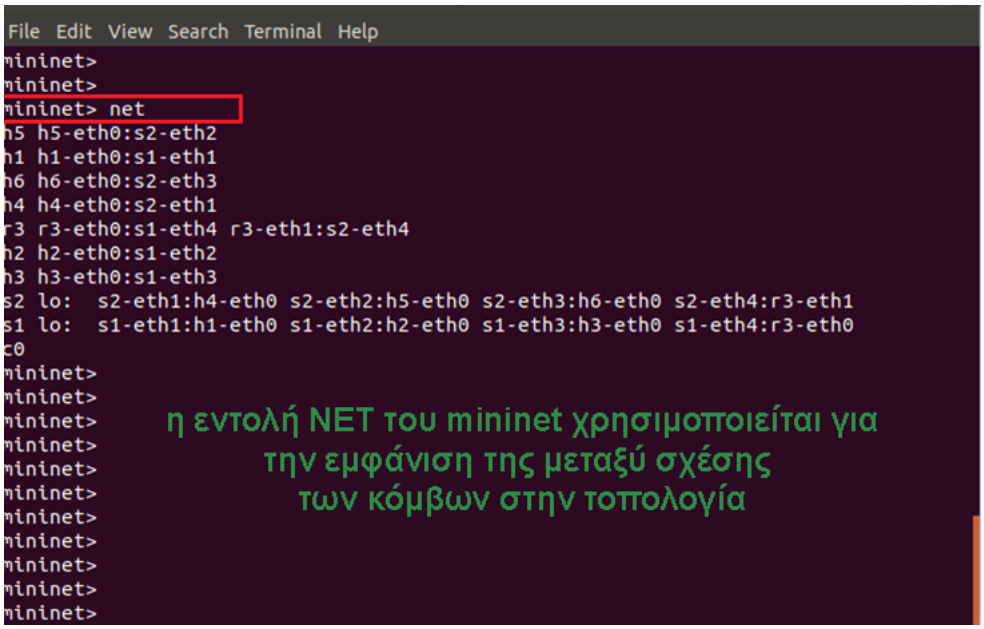
Εικόνα 52: Εντολή Σύνδεσμοι για τον έλεγχο της συνδεσιμότητας κόμβων.

Για την προβολή των κόμβων του δικτύου (nodes) χρησιμοποιείται η εντολή Mininet CLI prompt, όπως στο ακόλουθο σχήμα οι κόμβοι στην τοπολογία του δικτύου εμφανίζονται με την εντολή node.



Εικόνα 53: Εντολή Nodes για τον έλεγχο των κόμβων στην τοπολογία δικτύου

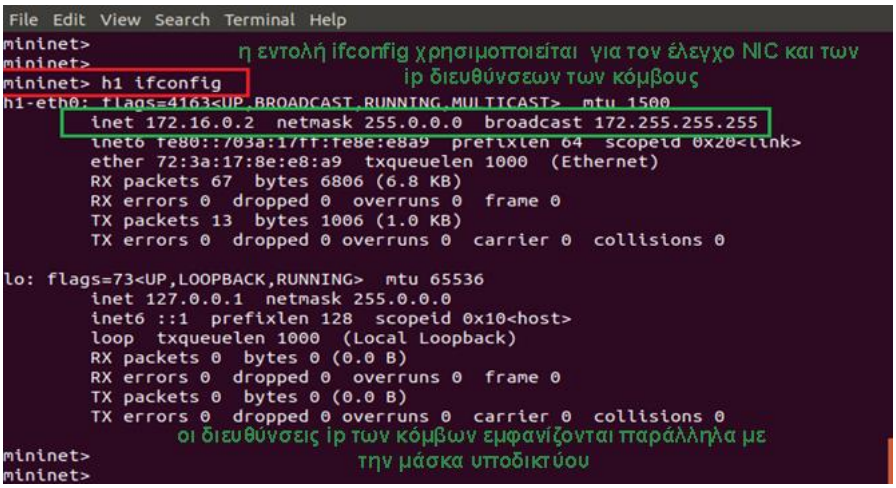
Για την προβολή του σεναρίου δικτύου στο Mininet CLI prompt, χρησιμοποιείται η εντολή net που απεικονίζεται στο ακόλουθο σχήμα.



Εικόνα 54: Εντολή Net για την προβολή της πλήρους τοπολογίας δικτύου με τη συνδεσιμότητα κόμβων.

Η εντολή Net εμφανίζει τους κόμβους στην τοπολογία δικτύου και τη συνδεσιμότητα τους μεταξύ τους με όλες τις χρησιμοποιούμενες διασυνδέσεις κάθε κόμβου.

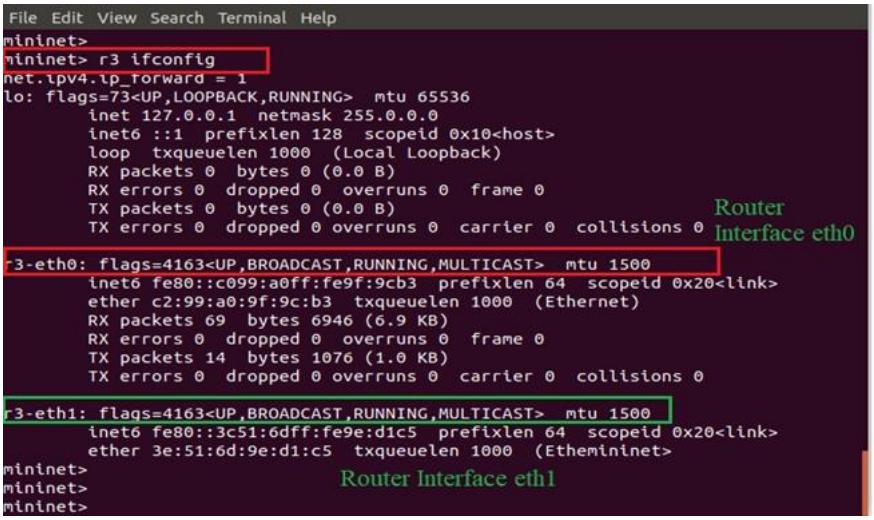
Η εντολή «ifconfig» (Διαμόρφωση διεπαφής) με το όνομα του κεντρικού υπολογιστή μπορεί να χρησιμοποιηθεί για την προβολή της διαμόρφωσης οποιουδήποτε συγκεκριμένου κόμβου όπως αναφέρεται στην παρακάτω εικόνα (Διαμόρφωση του κεντρικού υπολογιστή 1). Για την προβολή της διαμόρφωσης συγκεκριμένου κόμβου, μπορεί να χρησιμοποιηθεί η εντολή με τη μορφή (host name if config).



Εικόνα 55: Εντολή για τον έλεγχο των ρυθμίσεων της διασύνδεσης δικτύου H1 Έλεγχος.

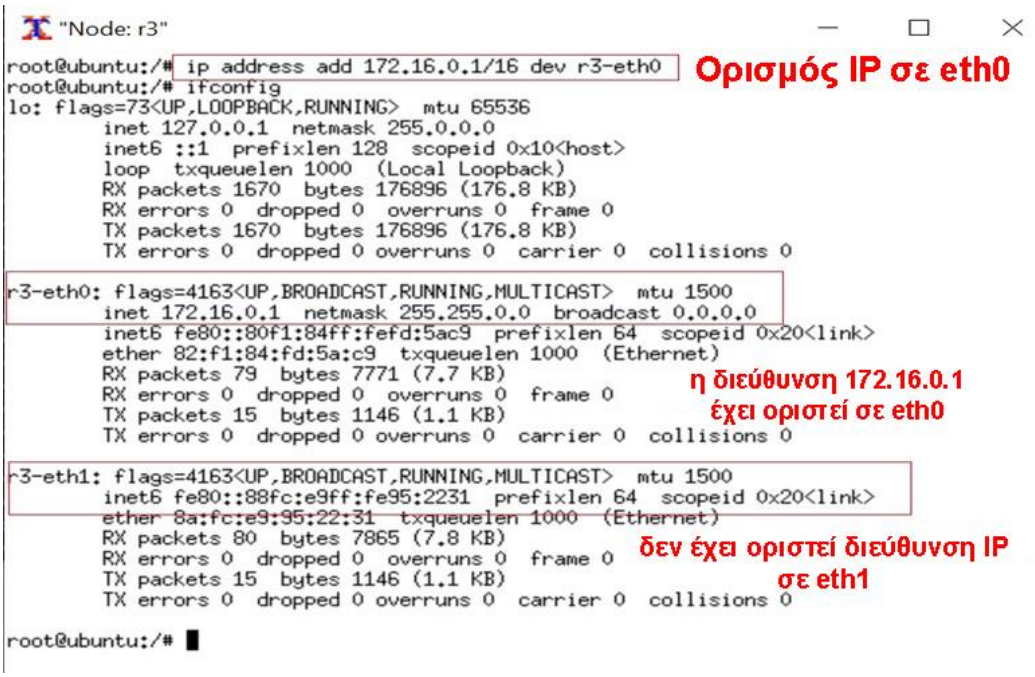
Στο παραπάνω σχήμα, για την προβολή της διαμόρφωσης του κεντρικού υπολογιστή 1, χρησιμοποιείται η εντολή h1 ifconfig που εμφανίζει όλες τις πληροφορίες διαμόρφωσης του κεντρικού υπολογιστή 1.

Ο δρομολογητής r3 έχει διαμορφωθεί με δύο διεπαφέςr3-eth0 και r3-eth1 όπου η μία διεπαφή είναι συνδεδεμένη με το S1 (Switch 1) και η άλλη διεπαφή είναι συνδεδεμένη με το S2 (Switch 2). Στην ακόλουθη εικόνα παρουσιάζονται οι λεπτομέρειες των διεπαφών του δρομολογητή.



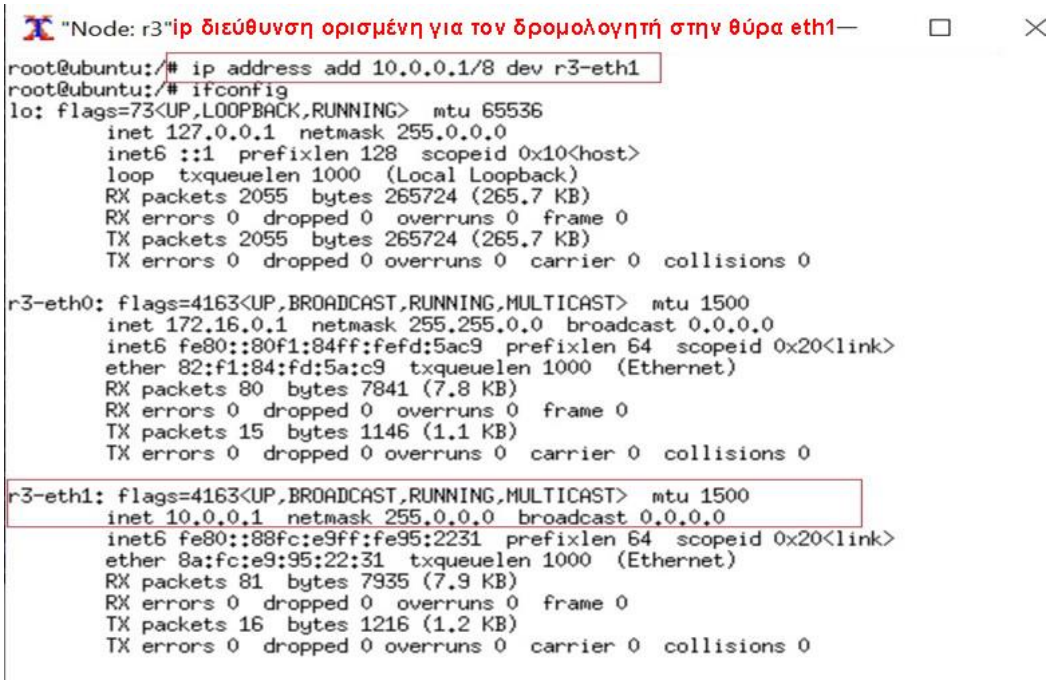
Εικόνα 56: Εντολή για τον έλεγχο της διαμόρφωσης και των διεπαφών του δρομολογητή (r3).

Όπως αναφέρθηκε στην τοπολογία δικτύου, έχουν διαμορφωθεί δύο διαφορετικά υποδίκτυα, οπότε για την επικοινωνία μεταξύ των δύο υποδικτύων είναι απαραίτητο να εκχωρηθεί διεύθυνση IP στις διεπαφές του δρομολογητή. Η διεπαφή δρομολογητή r3-eth0 που συνδέεται με το S1 (Switch 1) εκχωρεί τη διεύθυνση IP 172.16.0.1/16 και η διεπαφή δρομολογητή r3-eth1 που συνδέεται με το S2 (Switch 2) εκχωρεί τη διεύθυνση IP10.0.0.1/8. Στην ακόλουθη εικόνα απεικονίζεται η ανάθεση IP στις διεπαφές του δρομολογητή.



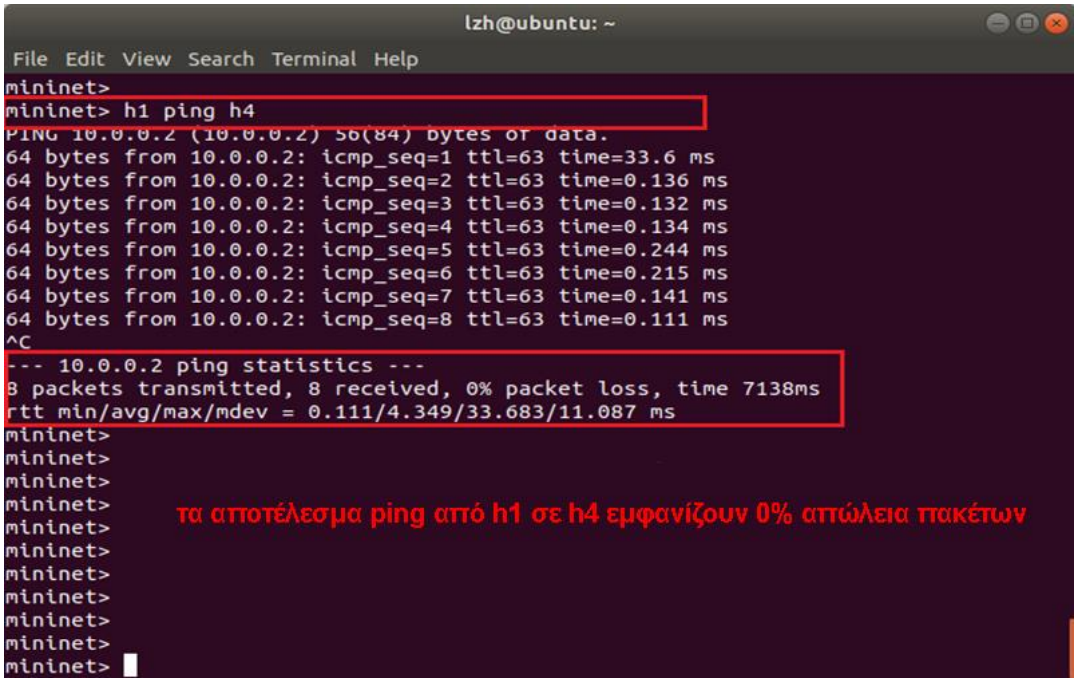
Εικόνα 57: IP Address Assignment to Router (r3) Interfaces eth0.

Στο παραπάνω σχήμα απεικονίζεται η διαμόρφωση του r3-eth0 και η διεύθυνση IP 172.16.0.1/16 που έχει εκχωρηθεί για επικοινωνία με το υποδίκτυο 172.16.0.0/16 που είναι συνδεδεμένο με το Switch 1 με μάσκα υποδικτύου 255.255.0.0 και η μέγιστη μονάδα μετάδοσης (mtu) για το r3-eth0 έχει οριστεί σε 1500 bytes που είναι το mtu του Ethernet Medium. Στο ακόλουθο σχήμα απεικονίζεται η διαμόρφωση της διασύνδεσης και η κατανομή της διεύθυνσης IP για το r3-eth1 που συνδέεται με το S2 (Switch 2) που χρησιμοποιείται για τη συνδεσιμότητα του υποδικτύου 10.0.0.0/8 και το mtu είναι το ίδιο 1500 bytes.



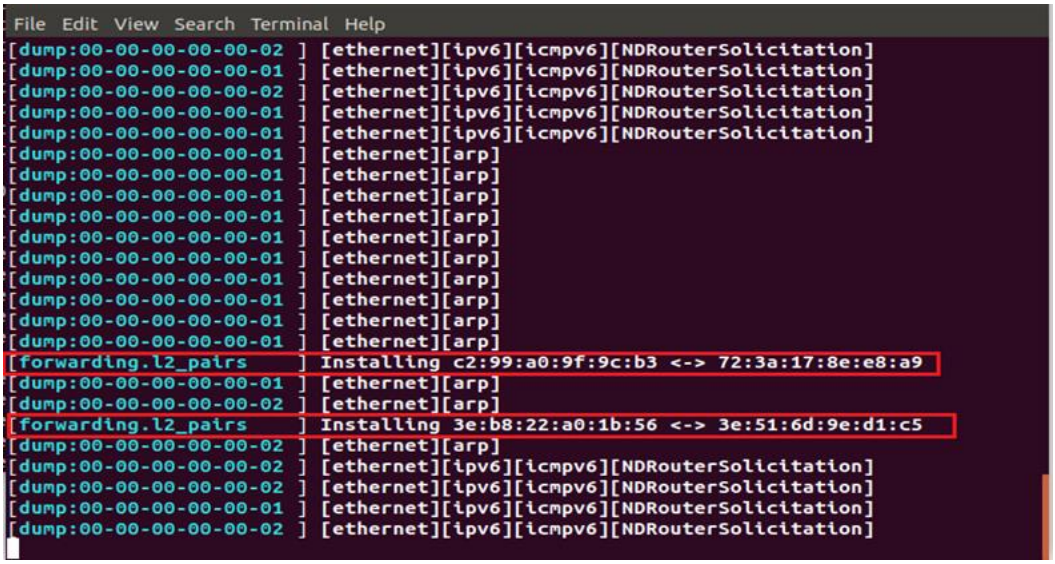
Εικόνα 58: Διασύνδεση eth1 του δρομολογητή (r3) με διεύθυνση IP.

Μετά την επιτυχή κατανομή των διευθύνσεων IP στις διασυνδέσεις του δρομολογητή και με τα δύο υποδίκτυα, αποστέλλεται ping από τους κόμβους H1 έως H4, οι οποίοι είναι συνδεδεμένοι με δύο διαφορετικούς μεταγωγείς S1 και S2 και ανήκουν σε διαφορετικά υποδίκτυα ενώ παράλληλα ελέγχεται αν οι δύο κόμβοι στέλνουν/δέχονται pings μεταξύ τους ή όχι. Η δοκιμή ping πραγματοποιήθηκε με επιτυχία και η επικοινωνία μεταξύ δύο διαφορετικών υποδικτύων πραγματοποιήθηκε με τη χρήση του απομακρυσμένου ελεγκτή SDN. Τα αποτελέσματα ping για τη δοκιμή ping από το H1 έως το H4 απεικονίζονται στα ακόλουθα σχήματα:



Εικόνα 59: Ping μεταξύ h1 και h4.

Στο παραπάνω σχήμα, η δοκιμή Ping εκτελέστηκε με επιτυχία μέσω του Mininet Command Line Interface (CLI). Το H1 Ping H4 στέλνει 15 πακέτα, και όλα τα πακέτα αυτά λαμβάνονται επιτυχώς στο H4 χωρίς καμία απώλεια πακέτων, όπως απεικονίζεται στο παραπάνω σχήμα. Για την επιβεβαίωση των πακέτων echo που μεταδίδονται μεταξύ h1 και h4, το τερματικό του ελεγκτή POXSDN χρησιμοποιείται για την επαλήθευση των πακέτων pingecho και στην εικόνα παρακάτω απεικονίζονται τα αποτελέσματα για τη δοκιμή ping μεταξύ h1 και h4.

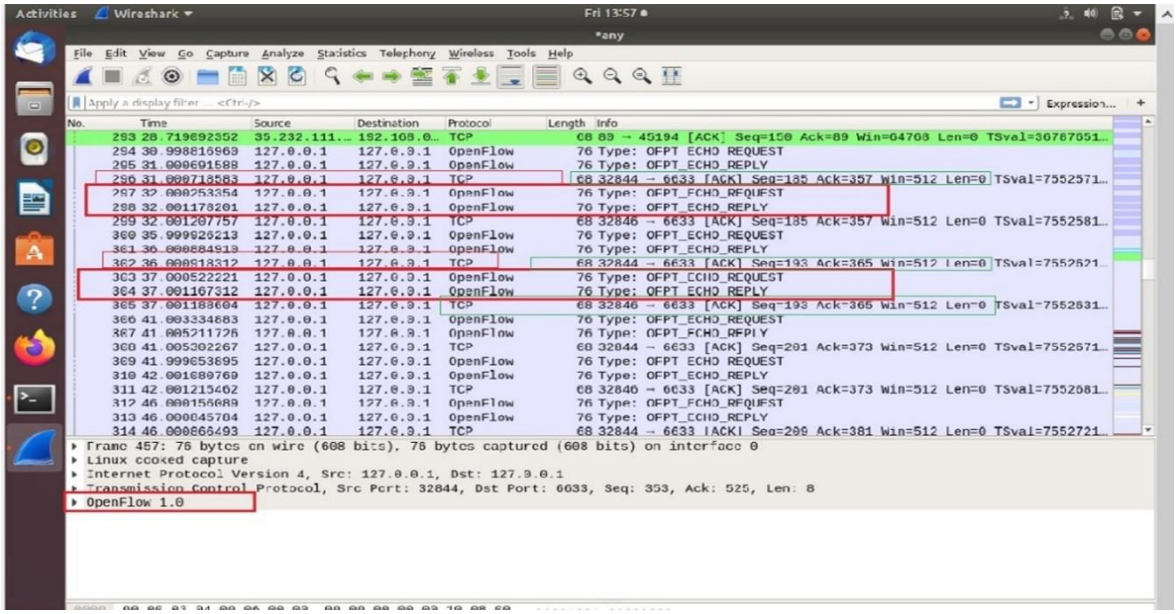


Εικόνα 60: Απόκριση πακέτου Ping στον ελεγκτή POXSDN.

Στην παραπάνω εικόνα, ανοίγει το τερματικό του ελεγκτή POX και φαίνεται ότι όταν εκτελείται η εντολή «h1 pingh4» στο τερματικό Mininet CLI, το τερματικό του ελεγκτή SDNδείχνει την εκτέλεση της εντολής ping με μήνυμα Installing και ταυτόχρονα εμφανίζει τη διεύθυνση MAC και των δύο κόμβων H1 και H4. Η ειδοποίηση που επισημαίνεται με κόκκινα πλαίσια δείχνει ότι ο κανόνας προώθησης πακέτων στον ελεγκτή SDN για την ανταλλαγή πακέτων pingecho μεταξύ των H1 και H4 έχει εγκατασταθεί στον ελεγκτή SDN.

4.14 Φιλτράρισμα και ανάλυση πακέτων με Wireshark

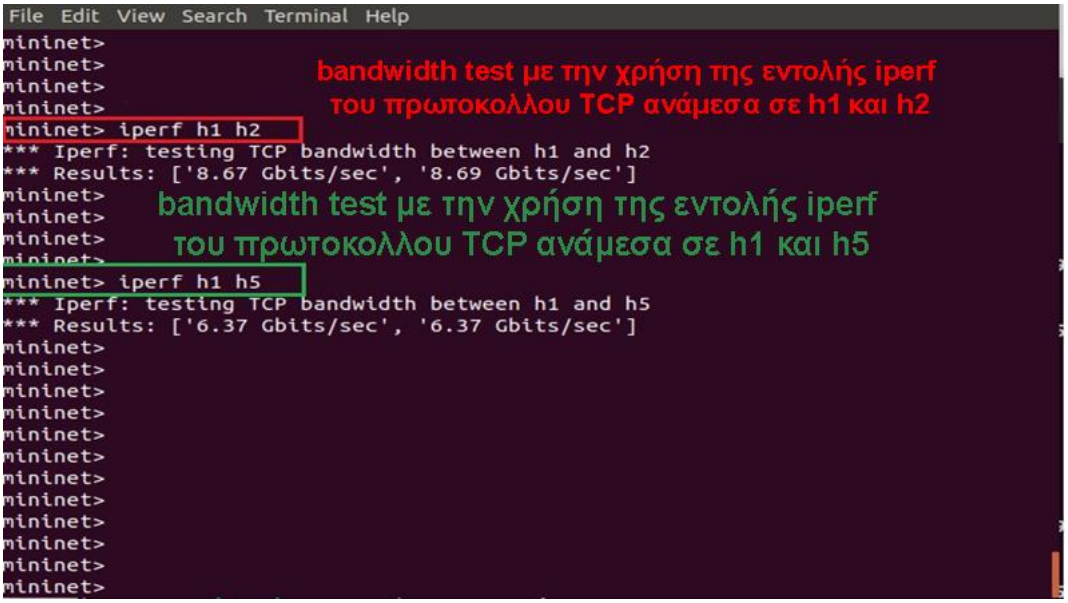
Το Wireshark είναι επίσης ένα ενσωματωμένο βοηθητικό πρόγραμμα του Linux που βοηθά στη σύλληψη πακέτων και στην ανάλυση της κίνησης δικτύου. Το Wireshark είναι διαθέσιμο στο Ubuntu και αν δεν είναι διαθέσιμο μπορούμε να το κατεβάσουμε χρησιμοποιώντας την εντολή «sudoapt-get install Wireshark». Το βοηθητικό πρόγραμμα Wireshark στο Ubuntu μπορεί να εκτελεστεί ανοίγοντας ένα τερματικό γραμμής εντολών και περνώντας την εντολή «Sudo Wireshark &».



Εικόνα 61: Καταγραφή πακέτων στο Wireshark για Ping.

Το παραπάνω στιγμιότυπο οθόνης παρουσιάζει το βοηθητικό πρόγραμμα Wireshark που χρησιμοποιήθηκε για τη λήψη πακέτων. Οι ειδοποιήσεις που επισημαίνονται με κόκκινο πλαίσιο δείχνουν τα μηνύματα pingecho Request και Reply που λαμβάνονται όταν το H1 κάνει ping στον κόμβο υποδοχής H4. Η ειδοποίηση δείχνει επίσης ότι τα μηνύματα μεταφέρονται με τη χρήση του πρωτοκόλλου ελεγκτή SDN Open Flow. Η ειδοποίηση τύπου πακέτου (OFPT) Open Flow Packet Type Option που αναφέρεται ως Echo Request και Echo Reply πακέτο. Αυτό σημαίνει ότι τα πακέτα που αποστέλλονται και λαμβάνονται είναι το πακέτο Ping Echo Request Reply που ανταλλάσσονται με τη χρήση του πρωτοκόλλου Open Flow SDN.

Για τη δοκιμή του εύρους ζώνης μεταξύ των H1, H2 και H5 χρησιμοποιήθηκε το βοηθητικό πρόγραμμα «IPerf» με εντολή στο τερματικό Mininet CLI, όπως απεικονίζεται στο παρακάτω στιγμιότυπο οθόνης. Στο Mininet CLI εκτελείται η εντολή «iperfh1 h2» για τη δοκιμή εύρους ζώνης μεταξύ των κόμβων H1 και H2 και η εντολή «iperfh1 h5» για τη δοκιμή εύρους ζώνης μεταξύ των host 1 και host5. Μετά την εκτέλεση των εντολών για τη δοκιμή εύρους ζώνης μεταξύ των δύο h1 έως h2 (κεντρικοί υπολογιστές συνδεδεμένοι στα ίδια υποδίκτυα) και h1 έως h5 (κεντρικός υπολογιστής συνδεδεμένος με διαφορετικά υποδίκτυα) εμφανίζονται τα αποτελέσματα μετά την άμεση εκτέλεση της εντολής. Τα αποτελέσματα της δοκιμής εύρους ζώνης με χρήση του Mininet CLI παρουσιάζονται στο ακόλουθο σχήμα:

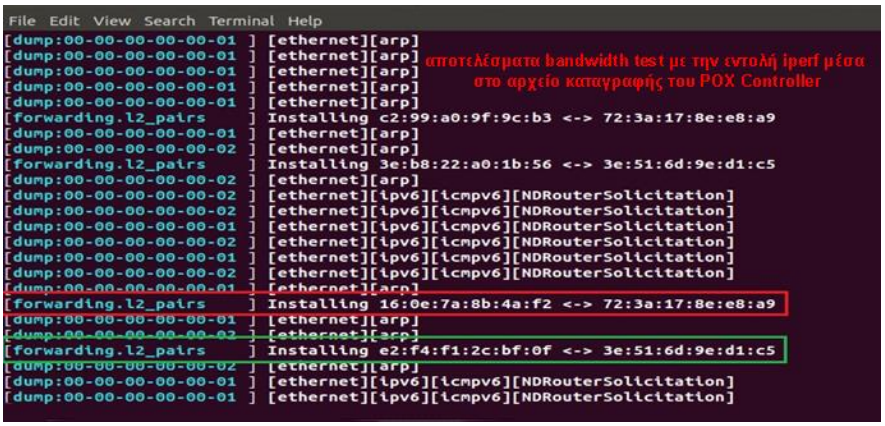


Εικόνα 62: Παραγωγή κυκλοφορίας με Iperf μεταξύ h1 και h2, h1 και h5.

Στο παραπάνω σχήμα, η εντολή iperf εκτελείται στο Mininet CLI για τη δοκιμή του εύρους ζώνης μεταξύ h1 και h2 και στη συνέχεια h1 και h5. Οι κόμβοι υποδοχής 1 και 2 είναι συνδεδεμένοι με τον ίδιο μεταγωγέαS1 και βρίσκονται στο ίδιο υποδίκτυο, ενώ οι κόμβοι υποδοχής h1 και h5 είναι συνδεδεμένοι με διαφορετικούς μεταγωγείς (h1 με S1) και h5 με S2 και επικοινωνούν μεταξύ τους χρησιμοποιώντας τις διεπαφές δρομολογητή. Τα αποτελέσματα εκτέλεσης εντολών εμφανίζονται μετά την άμεση εκτέλεση των εντολών.

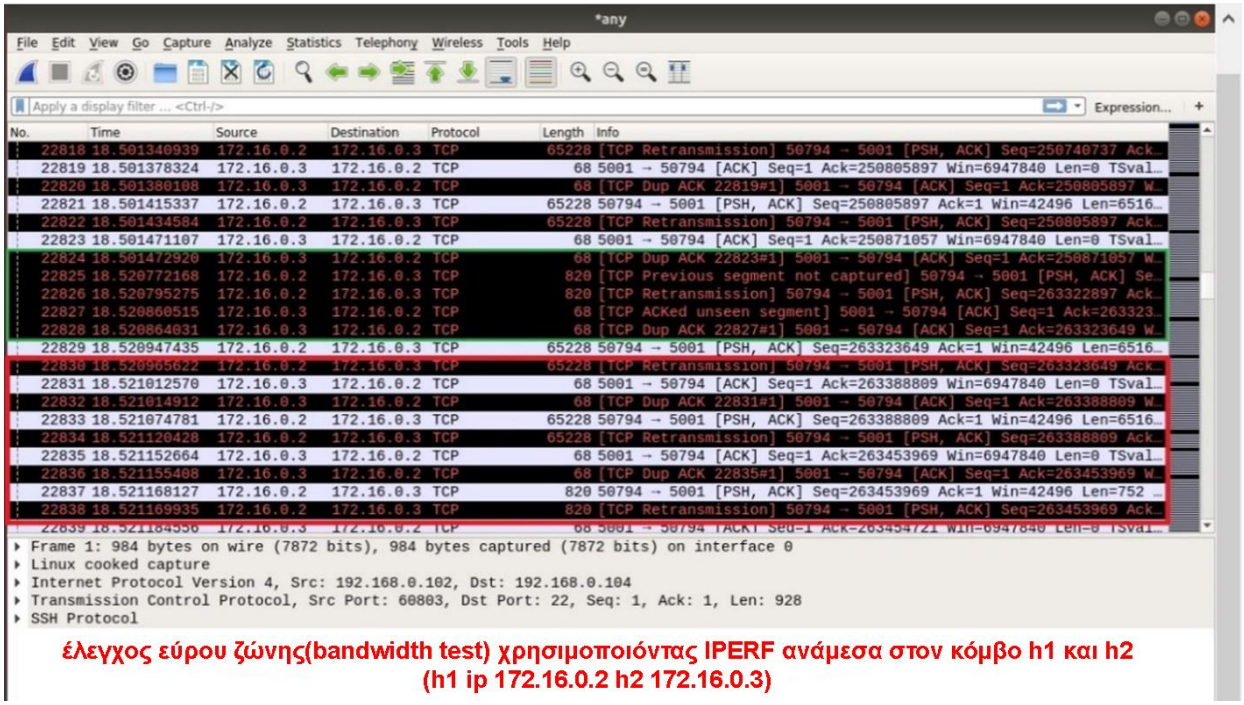
Σε απάντηση της εκτέλεσης της εντολής «iperfh1 h2» τα αποτελέσματα δείχνουν ότι μια δοκιμή εύρους ζώνης μεταξύ h1 και h2 εκτελείται με τη χρήση του πρωτοκόλλου TCP και το εύρος ζώνης μεταξύ h1 και h2 παρατηρείται μέχρι 8,67Gbits/Sec. Το ίδιο ισχύει και για τη δοκιμή εύρους ζώνης μεταξύ των h1 και h5, η εντολή «iperfh1 h5» δείχνει τα αποτελέσματα του υπολογισμού του εύρους ζώνης TCP και απεικονίζει το εύρος ζώνης 6,837Gbits/sec που παρατηρήθηκε μεταξύ του κεντρικού υπολογιστή 1 και του κεντρικού υπολογιστή 5.

Στην παρακάτω εικόνα το τερματικό του ελεγκτή POXSDN δείχνει τα αποτελέσματα για τις εντολές «iperfh1 h2» και «iperfh1 h5».



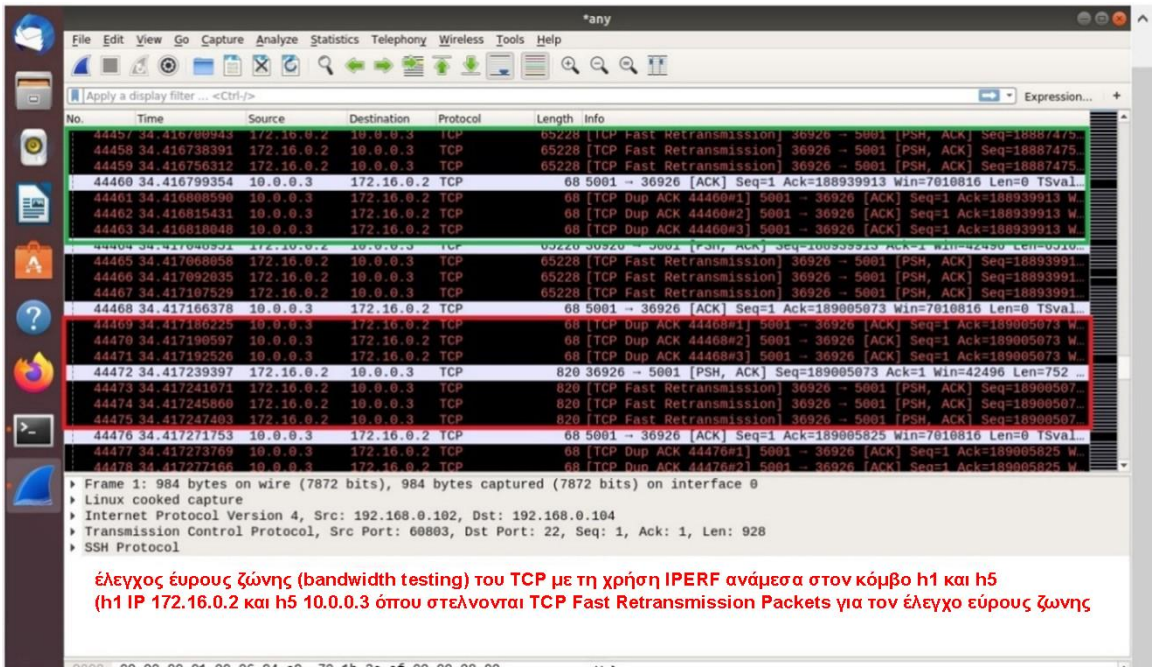
Εικόνα 63: Απόκριση iperf στον ελεγκτή SDN POX.

Στο παραπάνω στιγμιότυπο οθόνης, τα κόκκινα πλαίσια δείχνουν ότι εκτελέστηκαν οι εντολές δοκιμής εύρους ζώνης iperfμαζί με τον κανόνα προώθησης πακέτων που έχει εγκατασταθεί στον ελεγκτή SDN και εμφανίζεται με τις διευθύνσεις MAC των h1, h2 και h5. Η δοκιμή εύρους ζώνης Iperfφαίνεται μέσο του Wireshark Packet Capturing και απεικονίζεται παρακάτω



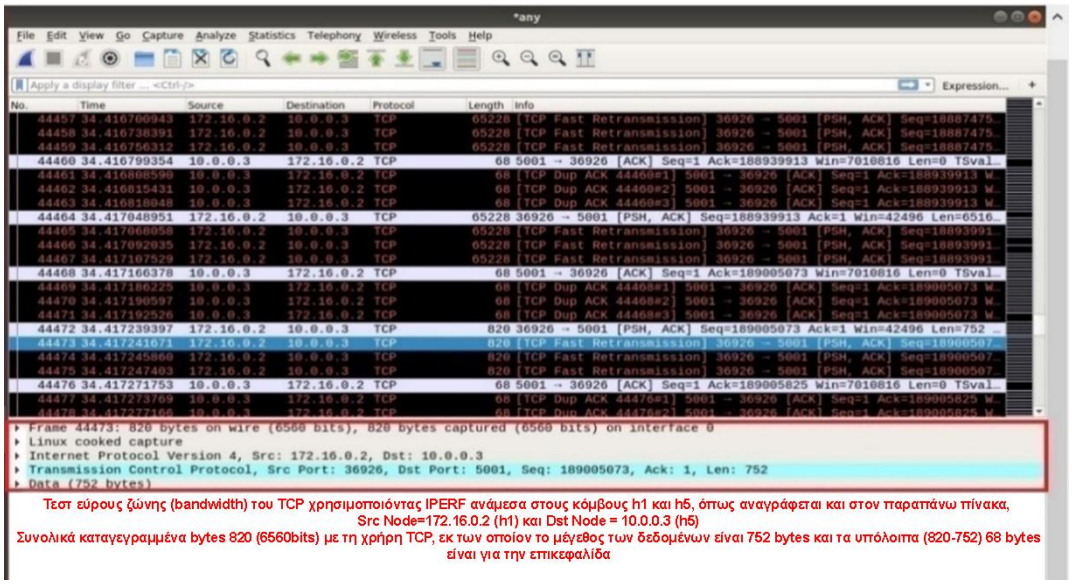
Σχήμα 64: Καταγραφή πακέτων μεταξύ h1 και h2 στο Wireshark.

Η εκτέλεση της εντολής δοκιμής εύρους ζώνης iperf επαληθεύτηκε από το βοηθητικό πρόγραμμα καταγραφής πακέτων Wireshark. Τα επισημασμένα πλαίσια κειμένου στο βοηθητικό πρόγραμμα Wireshark packet capturing δείχνει ότι τα πακέτα TCP αποστέλλονται από τους h1 και h2 που βρίσκονται στα ίδια υποδίκτυα και συνδέονται με τον ίδιο μεταγωγέα (S1). Το Σχήμα 69 δείχνει τα αποτελέσματα της δοκιμής εύρους ζώνης iperf μεταξύ των h1 και h5 (και οι δύο κόμβοι συνδέονται με διαφορετικά υποδίκτυα.



Εικόνα 65: Καταγραφή πακέτων TCP μεταξύ h1 και h5.

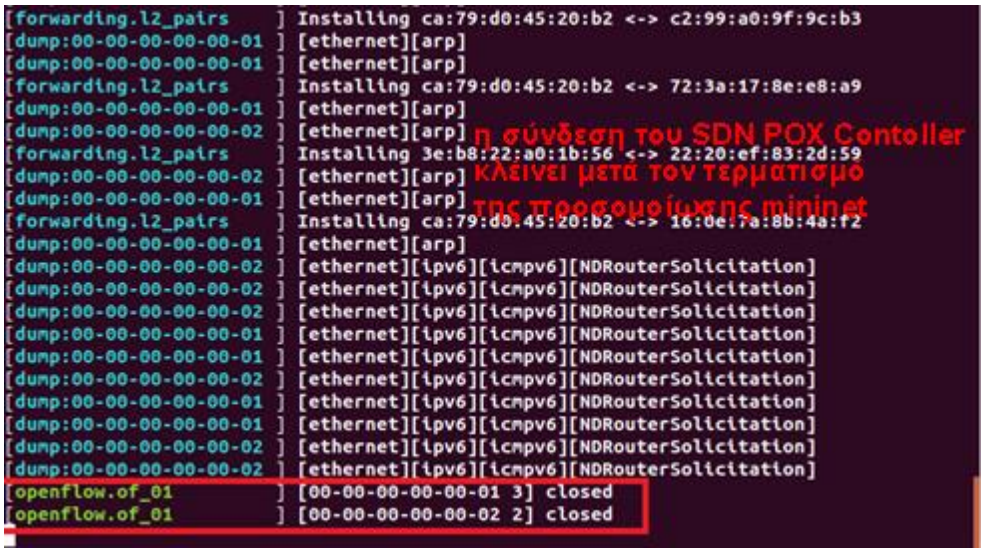
Στο παραπάνω σχήμα, η καταγραφή πακέτων του Wireshark δείχνει το αποτέλεσμα της δοκιμής εύρους ζώνης TCP των h1 και h5 και επίσης ότι και οι δύο κόμβοι (h1 και h5) είναι σε θέση να επικοινωνούν μεταξύ τους. Η διεύθυνση IP του κόμβου υποδοχής 1 είναι 172.16.0.2 (συνδεδεμένου το S1) και η διεύθυνση IP 10.0.0.3 του κόμβου υποδοχής 5 απεικονίζεται επίσης ότι είναι συνδεδεμένος με το S2 σε διαφορετικό υποδίκτυο και επικοινωνούν μεταξύ τους.



Εικόνα 66: Καταγραφή πακέτων μεταξύ h1 και h5 στο Wireshark.

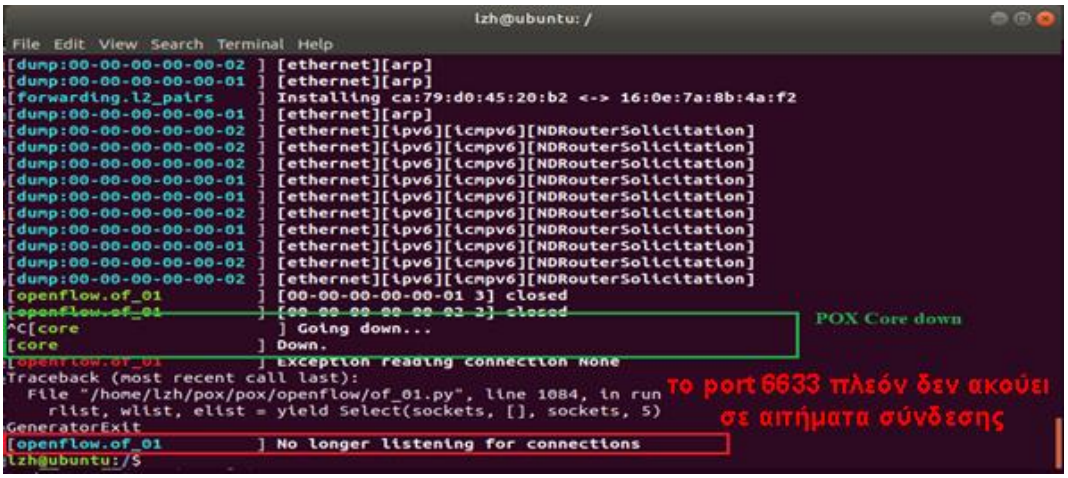
Το παραπάνω σχήμα δείχνει τον έλεγχο εύρους ζώνης με την εντολή iperf μεταξύ των κόμβων υποδοχής 1, 2 και 1 και 5, όπου η διεύθυνση πηγής είναι 172.16.0.2 (h1) και η διεύθυνση προορισμού 10.0.0.3 (h5).

Μετά τον επιτυχή σχεδιασμό της τοπολογίας δικτύου, τη συνδεσιμότητα της προσομοίωσης με τον ελεγκτή SDN, την εκτέλεση της δοκιμής ping και της δοκιμής εύρους ζώνης TCP με τη χρήση του iperf μεταξύ των κόμβων h1, h2 και h5, το βοηθητικό πρόγραμμα iperf κλείνει όπως επίσης και ο ελεγκτής POXSDN και ο προσομοιωτής Mininet κλείνουν.



Εικόνα 67: Διακοπή τοπολογίας στον ελεγκτή SDN Open Flow Protocol Closed.

Μετά τη διακοπή και το κλείσιμο της συνδεσιμότητας μεταξύ της προσομοίωσης δικτύου Mininet και του POX SDN Controller, ο SDN Controller σταματά και τερματίζει το τερματικό CLI. Η διακοπή και το κλείσιμο του ελεγκτή SDN δείχνει ότι ο πυρήνας του ελεγκτή SDN έχει κλείσει και έχει σταματήσει να ακούει πακέτα OpenFlow από συνδέσεις.



Εικόνα 68: Ο ελεγκτής SDN κλείνει και σταματά την ακρόαση της τοπολογίας δικτύου.

Στο παραπάνω επίπεδο (Επίπεδο 2) φτιάχτηκε ο σχεδιασμός της τοπολογίας δικτύου, εκτελέστηκε η συνδεσιμότητα του ελεγκτή SDN με το σενάριο δικτύου Mininet, πραγματοποιήθηκαν δοκιμές PING μεταξύ κόμβων του ίδιου υποδικτύου αλλά και διαφορετικών υποδικτύων και διαπιστώθηκε ότι η επικοινωνία μεταξύ των h1, h2 και h1 με h5 είναι εντάξει.

ΚΕΦΑΛΑΙΟ 5ο

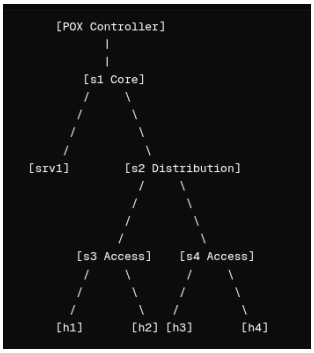
ΠΡΟΣΟΜΟΙΩΣΗ ΔΙΚΤΥΟΥ ΑΠΟ ΠΙΘΑΝΕΣ ΕΠΙΘΕΣΕΙΣ

5.1 Προσομοίωση δικτύου SDN με επίθεση DDoS χρησιμοποιώντας Mininet και ελεγκτή POX

Το mininet εκτός από δημιουργία τοπολογιών μέσω του γραφικού περιβάλλοντος όπως είδαμε και παραπάνω με προηγούμενα παραδείγματα, υποστηρίζει επίσης και την δυνατότητα δημιουργίας τοπολογιών μέσω κώδικα python. Παρακάτω είναι γραμμένος ένας κώδικας python όπου δημιουργείται για την προσομοίωση και περιλαμβάνει: έναν ελεγκτή (sdn controller), τέσσερις μεταγωγείς (switches), τέσσερις κύριοι υπολογιστές (hosts) και έναν εξυπηρετητή (server) με συγκεκριμένους παραμέτρους όπου αναγράφονται με μπλε σχόλια δίπλα από κάθε γραμμή του κώδικα python.

Σχεδιασμός τοπολογίας

Αυτή η εικόνα είναι απλώς για να κατανοήσουμε πώς είναι η συνδεσμολογία στην τοπολογία μας. Το Mininet δεν παρέχει γραφική διεπαφή χρήστη (GUI) για την οπτικοποίηση της τοπολογίας. Είναι ένας εξομοιωτής δικτύου που βασίζεται σε διεπαφή γραμμής εντολών (CLI) και δημιουργεί και διαχειρίζεται τοπολογίες εικονικού δικτύου μέσα σε ένα εικονικοποιημένο περιβάλλον. Ενώ επιτρέπει στους χρήστες να σχεδιάζουν σύνθετες τοπολογίες και να προσομοιώνουν πραγματική συμπεριφορά δικτύου, το δίκτυο κατασκευάζεται και διαχειρίζεται μέσω εντολών CLI και όχι μέσω οπτικής διεπαφής.



Εικόνα 69: Οπτική απεικόνιση του δικτύου

Δημιουργία της Τοπολογίας Δικτύου

Αποθηκεύουμε τον παρακάτω κώδικα με το όνομα "fixed_topology.py".

```
# Python code for topology
#!/usr/bin/python
from mininet.net import Mininet  # για τη δημιουργία εικονικού δικτύου
from mininet.node import Controller, RemoteController, OVSKernelSwitch
                                  # προσθήκη ελεγκτών SDN, τύπος εικονικού switch
from mininet.cli import CLI   # ανοίγει τη γραμμή εντολών
from mininet.log import setLogLevel, info # εμφάνιση μηνυμάτων στο τερματικό
from mininet.link import TCLink      # προσθήκη συνδέσμων με όρια εύρους ζώνης
import time                          # για την καθυστέρηση των ενεργειών
def createSimpleTopology():  #(Όλα όσα υπάρχουν μέσα σε αυτήν τη συνάρτηση δημιουργούν το δίκτυο.)
  net = Mininet(controller=RemoteController, link=TCLink,
switch=OVSKernelSwitch)
                                   #Δημιουργεί ένα εικονικό δίκτυο όπου:
```

- Χρησιμοποιεί απομακρυσμένο ελεγχκτή (POX σε δίκτυο).
- Χρησιμοποιεί το TCLink για τον έλεγχο του εύρους ζώνης σύνδεσης.
- Χρησιμοποιεί Open vSwitch Kernel Switches.

```
  # Add remote controller
    info('*** Adding controller\n')
    c0 = net.addController('c0', controller=RemoteController, ip='127.0.0.1',
port=6633)

    # Add switches
    info('*** Adding switches\n')
    s1 = net.addSwitch('s1')  # Core switch
```

```
s2 = net.addSwitch('s2')  # Distribution switch
s3 = net.addSwitch('s3')  # Access switch 1
s4 = net.addSwitch('s4')  # Access switch 2

# Add hosts
info('*** Adding hosts\n')
# Client hosts
h1 = net.addHost('h1', ip='10.0.0.1/24')
h2 = net.addHost('h2', ip='10.0.0.2/24')
h3 = net.addHost('h3', ip='10.0.0.3/24')
h4 = net.addHost('h4', ip='10.0.0.4/24')

# Server host (target for DDoS)
srv1 = net.addHost('srv1', ip='10.0.0.100/24')

# Add links
info('*** Creating network links\n')
# Core to distribution
net.addLink(s1, s2, bw=100) # ένωση μεταξύ s1 and s2 με 100mb εύρος ζώνης
# Distribution to access
net.addLink(s2, s3, bw=50)
net.addLink(s2, s4, bw=50)

# Connect hosts to access switches
info('*** Connecting hosts to switches\n')
net.addLink(h1, s3, bw=10)
net.addLink(h2, s3, bw=10)
net.addLink(h3, s4, bw=10)
net.addLink(h4, s4, bw=10)
```

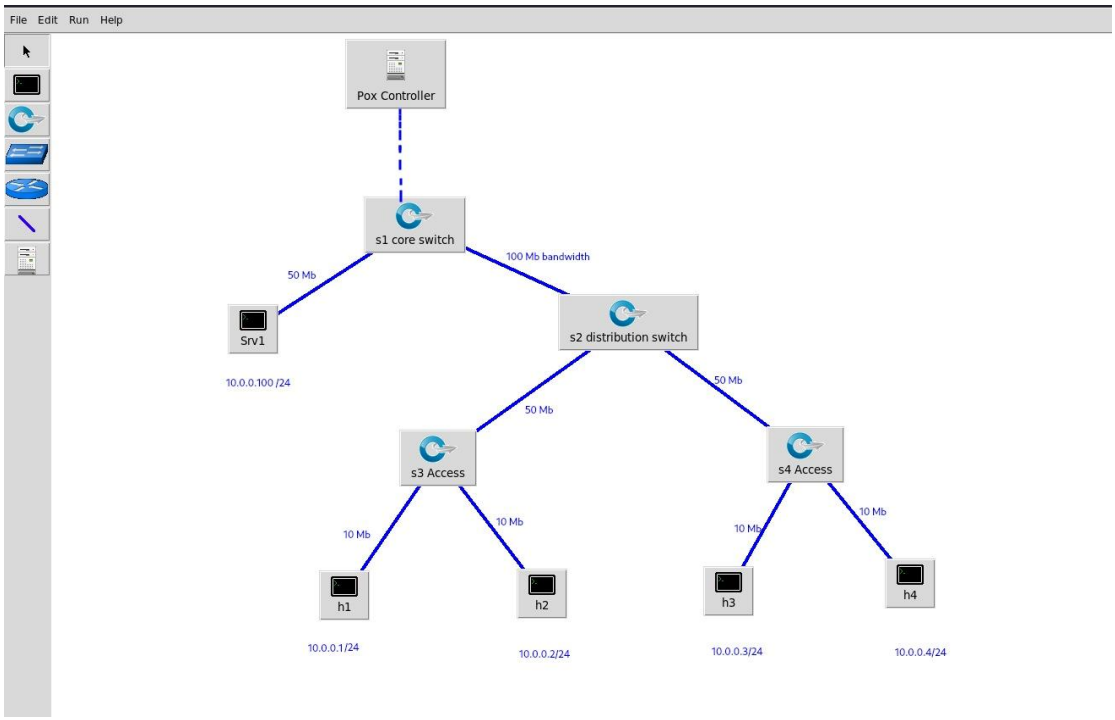
```
    # Connect server to core switch
    net.addLink(srv1, s1, bw=50)

    # Start network
    info('*** Starting network\n')
    net.build()
    c0.start()
    s1.start([c0])
    s2.start([c0])
    s3.start([c0])
    s4.start([c0])

    # Wait for controller to establish connections
    info('*** Waiting for controller to establish connections\n')
    time.sleep(5) # Αυτή η γραμμή κάνει το πρόγραμμα να διακόψει προσωρινά για 5 δευτερόλεπτα πριν προχωρήσει στο επόμενο βήμα και δίνει στον ελεγκτή και τους διακόπτες αρκετό χρόνο για να συνδεθούν πλήρως και να είναι έτοιμοι.

    info('*** Running CLI\n')
    CLI(net)

    info('*** Stopping network\n')
    net.stop()
if __name__ == '__main__':
    setLogLevel('info')
    createSimpleTopology()
```



Εικόνα 70: Οπτική απεικόνηση του δικτύου σε mininet

Εκτέλεση ελεγκτή**:**

Ανοίγουμε το τερματικό στο ubuntu και εκτελούμε τις ακόλουθες εντολές:

cd ~/pox

./pox.py log.level –DEBUG openflow.discovery forwarding.l2_learning

```
└─$ ./pox.py log.level --DEBUG openflow.discovery forwarding.l2_learning
POX 0.7.0 (gar) / Copyright 2011-2020 James McCauley, et al.
DEBUG:core:POX 0.7.0 (gar) going up...
DEBUG:core:Running on CPython (3.13.2/Mar 13 2025 14:29:07)
DEBUG:core:Platform is Linux-6.11.2-amd64-x86_64-with-glibc2.41
WARNING:version:POX requires one of the following versions of Python: 3.6 3.7
 3.8 3.9
WARNING:version:You're running Python 3.13.
WARNING:version:If you run into problems, try using a supported version.
INFO:core:POX 0.7.0 (gar) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[00-00-00-00-00-01 2] connected
DEBUG:openflow.discovery:Installing flow for 00-00-00-00-00-01
DEBUG:forwarding.l2_learning:Connection [00-00-00-00-00-01 2]
INFO:openflow.of_01:[00-00-00-00-00-02 3] connected
DEBUG:openflow.discovery:Installing flow for 00-00-00-00-00-02
DEBUG:forwarding.l2_learning:Connection [00-00-00-00-00-02 3]
INFO:openflow.discovery:link detected: 00-00-00-00-00-02.1 → 00-00-00-00-00-
01.1
```

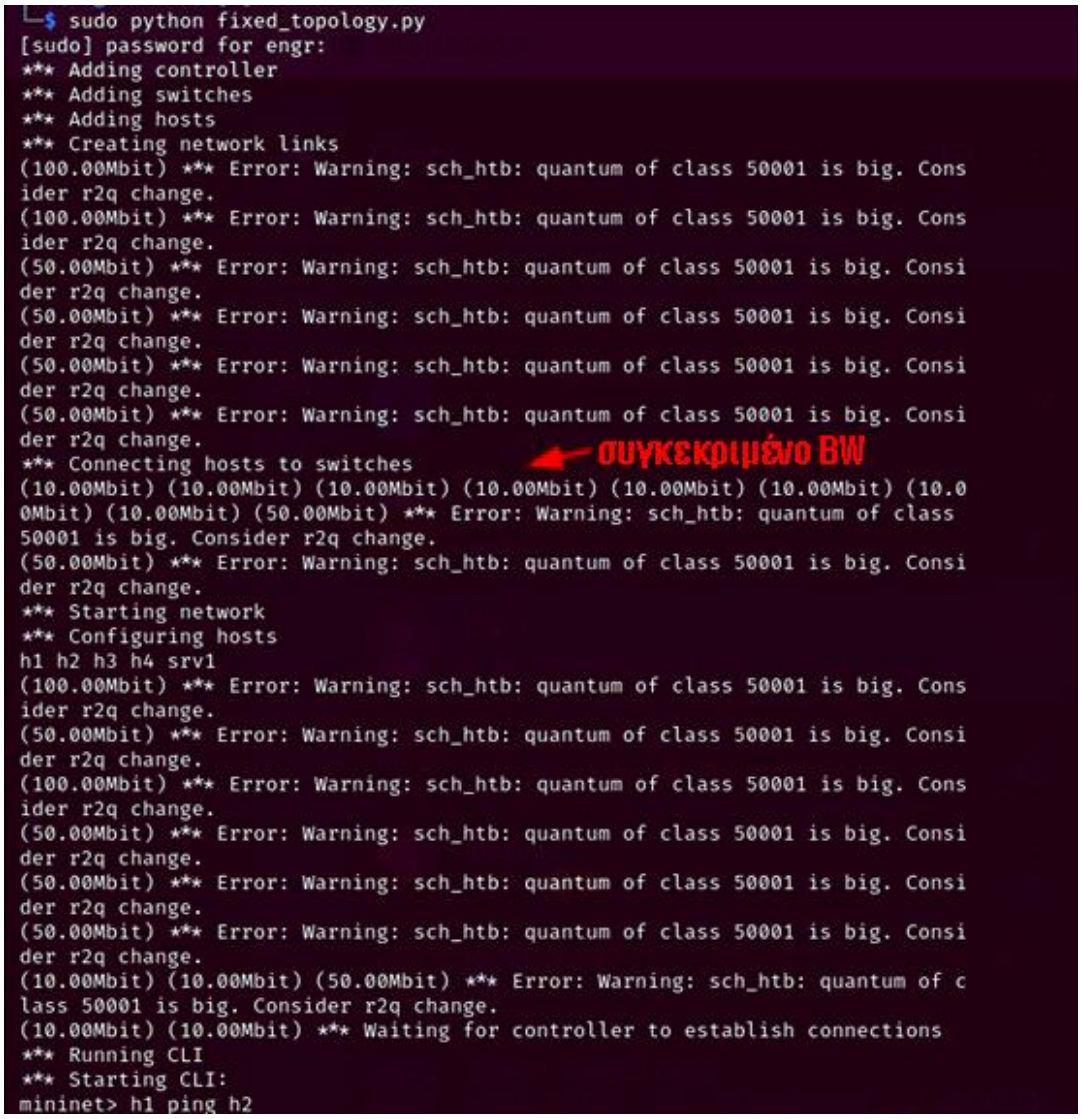
Εικόνα 71: Εκτέλεση του POX Controller

Δείχνει ότι ο ελεγκτής POX SDN εκκινείται με το **openflow.discovery** και **forward.l2_learning**. Το αρχείο καταγραφής επιβεβαιώνει ότι το POX εκτελεί ακρόαση στη θύρα 6633 και ότι οι μεταγωγείς έχουν συνδεθεί με επιτυχία. Επιπλέον, δείχνει εγκαταστάσεις ροής και ανίχνευση συνδέσεων σε δράση όπως φαίνεται στην παραπάνω εικόνα.

Εκτέλεση τοπολογίας

Ανοίγουμε ένα ακόμη τερματικό και γράφουμε την ακόλουθη εντολή για να εκτελέσουμε την κύρια τοπολογία. Αυτό θα δημιουργήσει την τοπολογία που θα περιλαμβάνει κεντρικούς υπολογιστές, switches και τον βασικό ελεγκτή, θα ορίσει επίσης το εύρος ζώνης για κάθε σύνδεσμο και στη συνέχεια θα μας μεταφέρει στη διεπαφή mininet όπου θα δοκιμάσουμε την τοπολογία σας.

Sudo python fixed_topology.py // αλλάζουμε το fixed_topology με το όνομα του αρχείου μας.



Εικόνα 72: Εκτέλεση τοπολογίας με συγκεκριμένο bandwidth

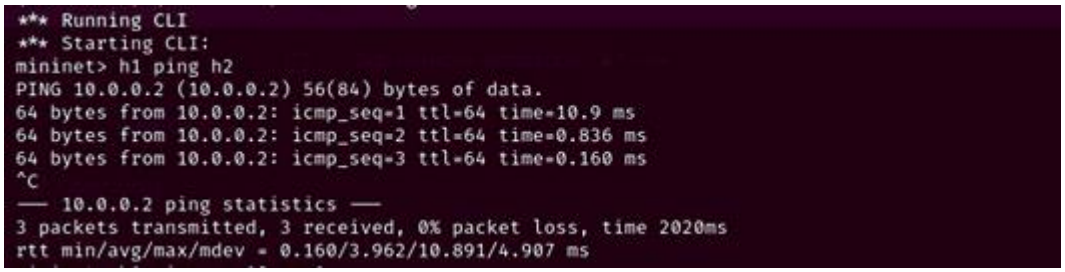
Η παραπάνω εικόνα μας δείχνει τη διαδικασία δημιουργίας της τοπολογίας δικτύου, και τη σύνδεσή τους με καθορισμένους περιορισμούς εύρους ζώνης. Το εύρος ζώνης για κάθε σύνδεσμο κατανέμεται ρητά (π.χ.,**10,00Mbit,50,00Mbit**), το οποίο είναι κρίσιμο για την προσομοίωση της πραγματικής συμπεριφοράς του δικτύου. Οι προειδοποιήσεις σχετικά με (**sch_htb**)υποδεικνύουν ότι το σύστημα προτείνει τη χρήση μικρότερων τιμών για ακριβέστερη διαμόρφωση εύρους ζώνης, αλλά αυτά δεν επηρεάζουν τη λειτουργικότητα της προσομοίωσης.

Προσομοίωση & Αποτελέσματα

Δοκιμές απόδοσης πριν από την επίθεση DDOS

Συνδεσιμότητα μεταξύ κεντρικών υπολογιστών:

h1 ping h2 // για έλεγχο της συνδεσιμότητας μεταξύ των κεντρικών υπολογιστών

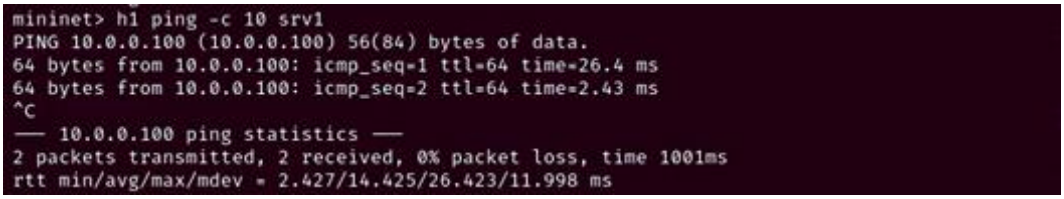


Εικόνα 73: Εκτέλεση εντολής ping από h1 σε h2

Αυτή η εικόνα δείχνει το αποτέλεσμα μιας βασικής δοκιμής συνδεσιμότητας και καθυστέρησης χρησιμοποιώντας την εντολή ping στο Mininet. Ο Host 1 ως κεντρικός υπολογιστής κάνει ping στον h2(IP: 10.0.0.2) και λαμβάνει επιτυχημένες απαντήσεις με ελάχιστη καθυστέρηση, υποδεικνύοντας ένα υγιές και ευέλικτο δίκτυο. Τα στατιστικά στοιχεία επιβεβαιώνουν μηδενική απώλεια πακέτων και ένα εύρος χρόνων μετ' επιστροφής (RTT), με μέση καθυστέρηση περίπου 3,96 ms. Αυτή η δοκιμή βοηθά στην επαλήθευση ότι το δίκτυο λειτουργεί σωστά πριν από την εισαγωγή οποιωνδήποτε διακοπών, όπως μια επίθεση DDoS.

Συνδεσιμότητα μεταξύ κεντρικού υπολογιστή και κόμβου υπηρεσίας:

h1 ping -c 10 srv1# αυτό σημαίνει ότι ο κεντρικός υπολογιστής h1 θα στείλει 10 μηνύματα ping στον διακομιστή srv1 για να ελέγξει αν είναι προσβάσιμος. Βοηθά στον έλεγχο της συνδεσιμότητας δικτύου μεταξύ h1 και srv1 στο εικονικό περιβάλλον. Εάν θέλουμε να αυξήσουμε ή να μειώσουμε τον αριθμό των πακέτων, θα αντικαταστήσουμε τον αριθμό 10.



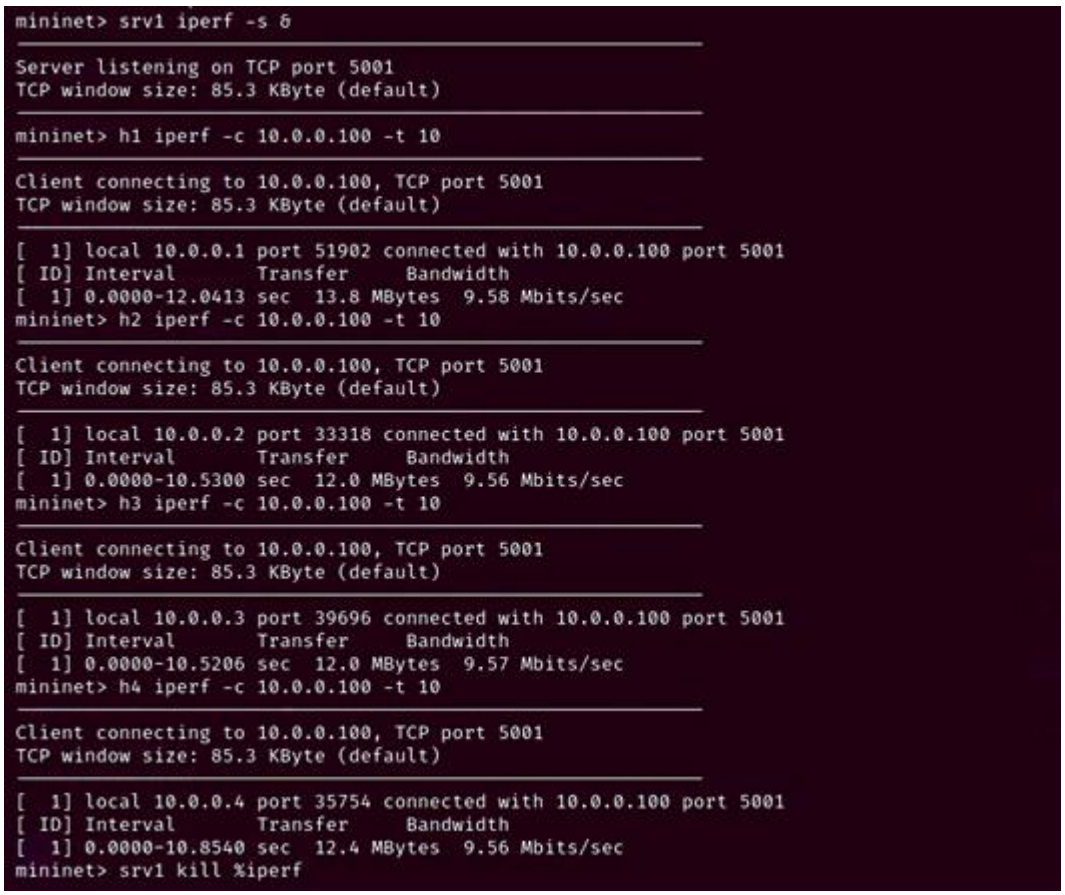
Εικόνα 74: Εκτέλεση εντολής ping από h1 σε srv1 με παραμέτρους

Παρακολούθηση εύρους ζώνης:

srv1 iperf -s & ξεκινά ένα εργαλείο που ονομάζεται iperf σε έναν υπολογιστή με το όνομα srv1, κάνοντάς το να λειτουργεί σαν διακομιστής που περιμένει μια σύνδεση για να μετρήσει την απόδοση του δικτύου, και συγκεκριμένα το εύρος ζώνης και στο τέλος κάνει την εντολή να εκτελείται στο παρασκήνιο, ώστε να μπορείτε να χρησιμοποιήσετε το τερματικό για άλλες εργασίες.

h1 iperf -c 10.0.0.100 -t 10 αυτό τρέχει iperf σε έναν άλλο υπολογιστή με το όνομα h1 και συνδέεται με τον διακομιστή στο 10.0.0.100 για 10 δευτερόλεπτα. Κατά τη διάρκεια αυτών των 10 δευτερολέπτων, μετρά την ταχύτητα με την οποία μπορούν να ταξιδέψουν τα δεδομένα μεταξύ των δύο υπολογιστών.

Επαναλαμβάνουμε αυτό το βήμα και για τους άλλους κεντρικούς υπολογιστές h2, h3 και h4 για να ελέγξουμε την απόδοση και το εύρος ζώνης μεταξύ του κεντρικού υπολογιστή και του κόμβου διακομιστή.



Εικόνα 75: Εκτέλεση εντολής iperf για μέτρηση εύρους ζώνης

Χρησιμοποιήσαμε το iperf εργαλείο μέσα σε ένα περιβάλλον δικτύου Mininet για τη μέτρηση του εύρους ζώνης TCP μεταξύ πολλαπλών κεντρικών υπολογιστών και ενός κεντρικού διακομιστή. Το iperf χρησιμοποιείται ευρέως για τον έλεγχο της απόδοσης του δικτύου και επιτρέπει την αξιολόγηση παραμέτρων όπως η απόδοση, η καθυστέρηση και το jitter.

Ένας διακομιστής αρχικοποιείται χρησιμοποιώντας το iperf -s εντολή σε έναν από τους κεντρικούς υπολογιστές Mininet (σε αυτήν την περίπτωση,srv1), το οποίο ακούει για εισερχόμενες συνδέσεις TCP. Κάθε κεντρικός υπολογιστής-πελάτης (h1,h2,h3, και h4) στη συνέχεια ξεκινά μια δοκιμή εύρους ζώνης στον διακομιστή χρησιμοποιώντας την iperf -c εντολή με καθορισμένη διάρκεια 10 δευτερολέπτων. Αυτό προσομοιώνει την επικοινωνία πολλαπλών πελατών με έναν κεντρικό κόμβο σε ένα Δίκτυο που Ορίζεται από Λογισμικό (SDN) ή σε μια παραδοσιακή αρχιτεκτονική δικτύου.

Ο σκοπός αυτής της ρύθμισης είναι η παρατήρηση και η ανάλυση των ρυθμών μεταφοράς δεδομένων μεταξύ διαφορετικών κόμβων, βοηθώντας στην κατανόηση της απόδοσης του δικτύου υπό ταυτόχρονες συνδέσεις. Αυτή η προσέγγιση είναι χρήσιμη για την αξιολόγηση της αποτελεσματικότητας της δρομολόγησης, της χωρητικότητας σύνδεσης και της συνολικής συμπεριφοράς του δικτύου σε ένα ελεγχόμενο εικονικό περιβάλλον.

Προσομοίωση επίθεσης DDoS

Τώρα ανοίγουμε ένα ξεχωριστό παράθυρο για κάθε κεντρικό υπολογιστή χρησιμοποιώντας την εντολή xterm ως εξής για να επιτεθείτε ταυτόχρονα από πολλαπλά σημεία του δικτύου

xterm h1

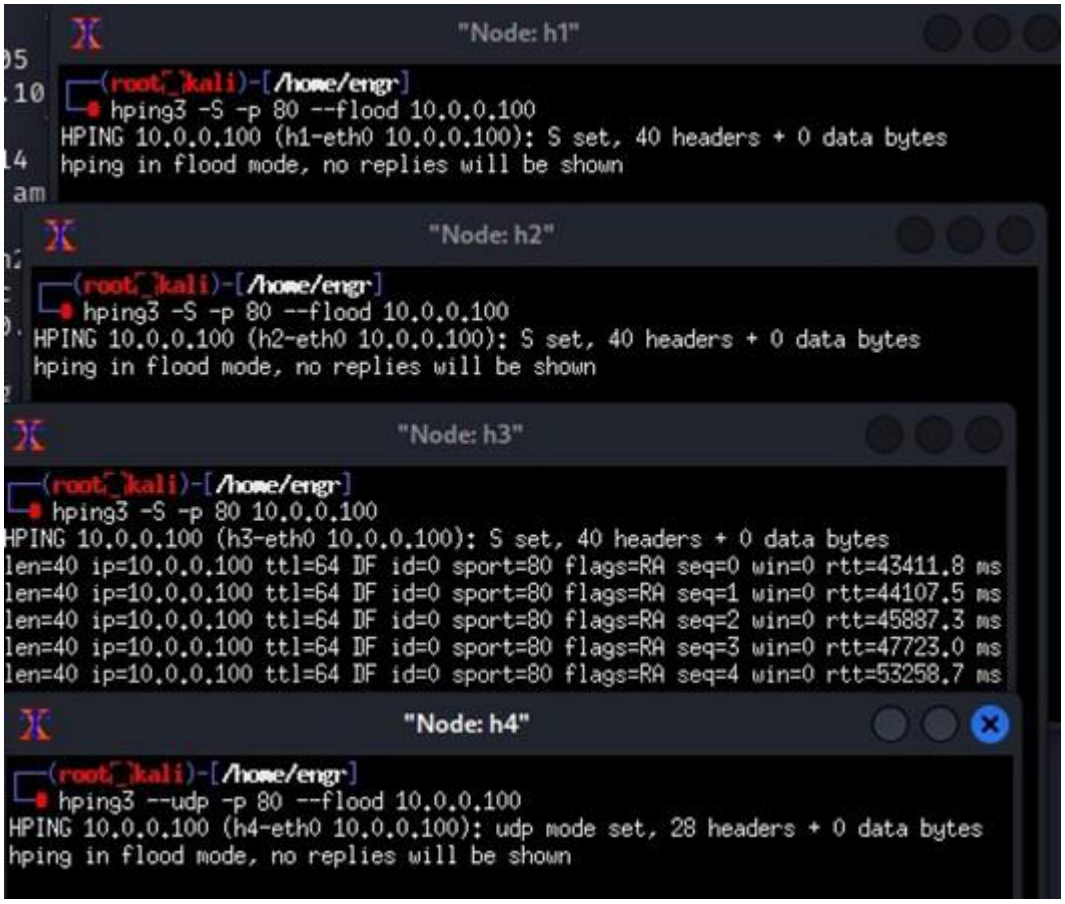
xterm h2

xterm h3

xterm h3

Επιθέσεις από τον οικοδεσπότη host:

Γίνεται εκτέλεση της εντολής hping3 -S -p 80 –flood 10.0.0.100 από κάθε κεντρικό υπολογιστή και μπορούμε επίσης να αλλάξουμε τον τύπο πακέτου και να αντικαταστήσουμε το -S με έναν από τους παρακάτω τύπους όπως: -udp, -icmp κ.λπ.



Εικόνα 76: Εκτέλεση εντολής hping3 από όλους τους host ξεχωριστά h1...h4

Το εργαλείο hping3 χρησιμοποιείται για την προσομοίωση διαφόρων τύπων επιθέσεων πλημμύρας δικτύου σε ένα ελεγχόμενο περιβάλλον Mininet.Γενικά, το hping3 είναι ένα ευέλικτο βοηθητικό πρόγραμμα δημιουργίας πακέτων που υποστηρίζει τη δημιουργία προσαρμοσμένων πακέτων TCP, UDP, ICMP και RAW-IP. Χρησιμοποιείται ευρέως για έλεγχο ασφαλείας, δοκιμές τείχους προστασίας και αξιολόγηση της απόδοσης δικτύου. Σε αυτό το πείραμα, χρησιμεύει ως μέσο για την εξομοίωση επιθέσεων άρνησης υπηρεσίας (DoS) και κατανεμημένων επιθέσεων άρνησης υπηρεσίας (DDoS).

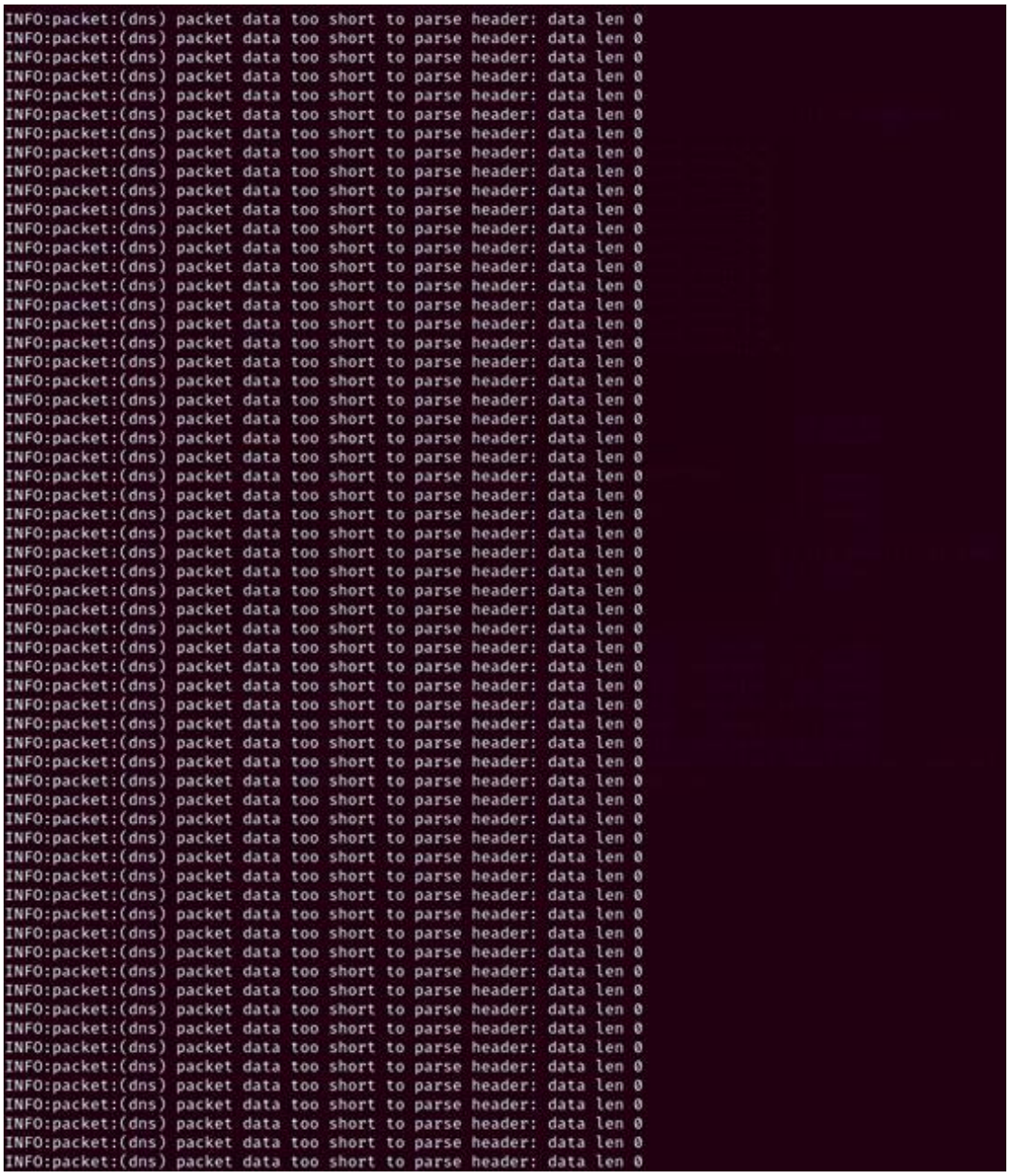
Η ακόλουθη εντολή εκτελέστηκε σε πολλαπλούς κεντρικούς υπολογιστές για την προσομοίωση μιας πλημμύρας flood SYN:

hping3 -S -p 80 --flood 10.0.0.100

- -S: Ορίζει τη σημαία TCP SYN, η οποία υποδεικνύει την έναρξη μιας σύνδεσης TCP.
- -p 80: Στοχεύει στη θύρα 80, η οποία χρησιμοποιείται συνήθως για κύρια για το πρωτόκολλο HTTP.
- --flood: Ενεργοποιεί τη λειτουργία flood, στέλνοντας πακέτα το συντομότερο δυνατό χωρίς να αναμένονται απαντήσεις.

Αυτή η εντολή προκαλεί τον κεντρικό υπολογιστή να μεταδίδει συνεχώς πακέτα TCP SYN στον καθορισμένο προορισμό (10.0.0.100), επιχειρώντας να ξεκινήσει ένας μεγάλος αριθμός μισάνοιχτων συνδέσεων TCP. Σκοπός είναι να κορεστούν οι πόροι χειρισμού συνδέσεων του στόχου, οδηγώντας σε πιθανή διακοπή της υπηρεσίας.

Επιθέσεις που καταγράφηκαν:

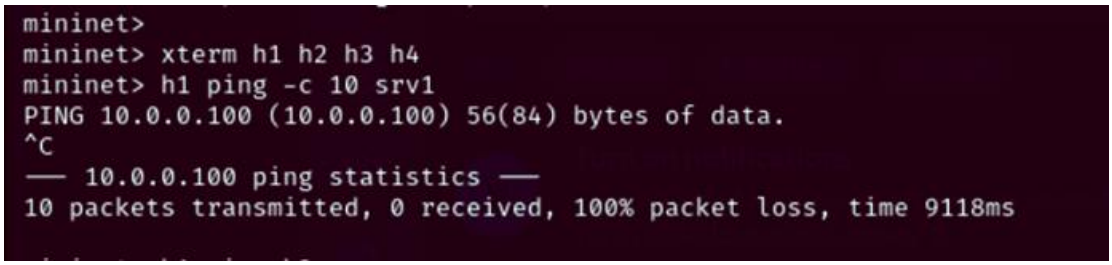


Εικόνα 77: Εκτέλεση εντολής SYN-flood με χρήση hping3

Δοκιμές απόδοσης κατά τη διάρκεια της επίθεσης

Επαναλήφθηκαν οι δοκιμές με iperf και ping κατά τη διάρκεια της επίθεσης. Παρακάτω θα δούμε ότι όσο υπήρξε η επίθεση, καμία από τις παραπάνω εντολές δεν επέστρεφε τα ίδια αποτελέσματα με πριν.

Συνδεσιμότητα μεταξύ κεντρικού υπολογιστή και κόμβου υπηρεσίας:

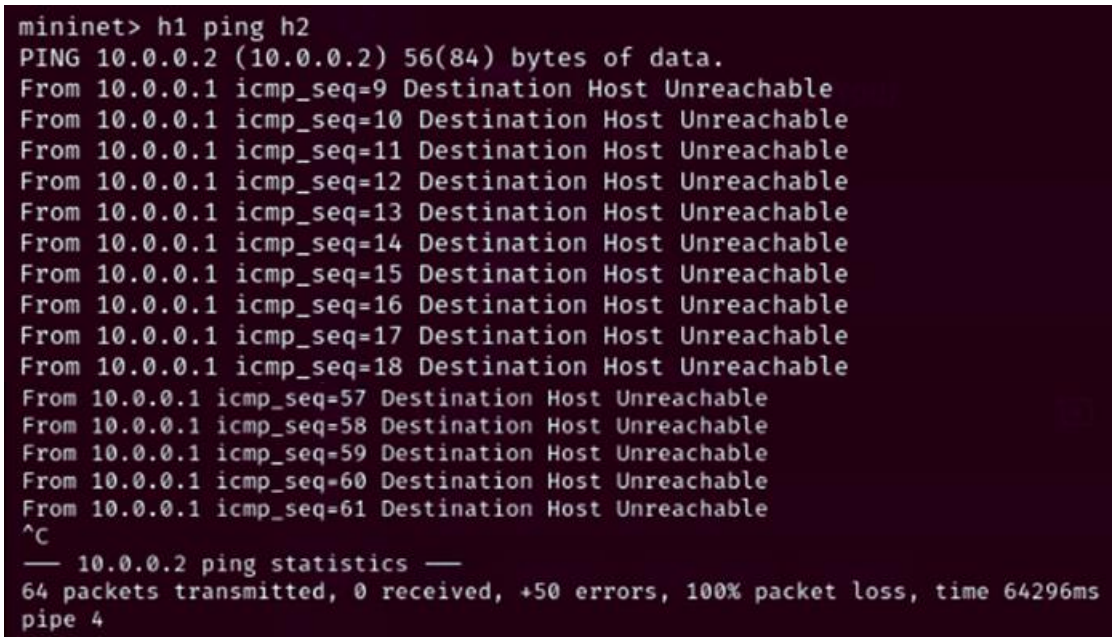


Εικόνα 78: Αποτυχία επικοινωνίας μέσω ping (100% απώλεια πακέτων) κατά την επίθεση

Η παραπάνω εικόνα μας δείχνει ότι κατά τη διάρκεια της επίθεσης, ο h1 δεν είναι σε θέση να κάνει ping στον κόμβο srv1. Χρησιμοποιήθηκε ένα βασικό αίτημα echo ICMP (ping) όπου εκτελέστηκε από τον κεντρικό υπολογιστή h1 στον διακομιστή προορισμού (srv1) με διεύθυνση IP10.0.0.100 για να αξιολογήσει τη διαθεσιμότητα του δικτύου κατά τη διάρκεια των συνεχιζόμενων προσομοιωμένων επιθέσεων flood.

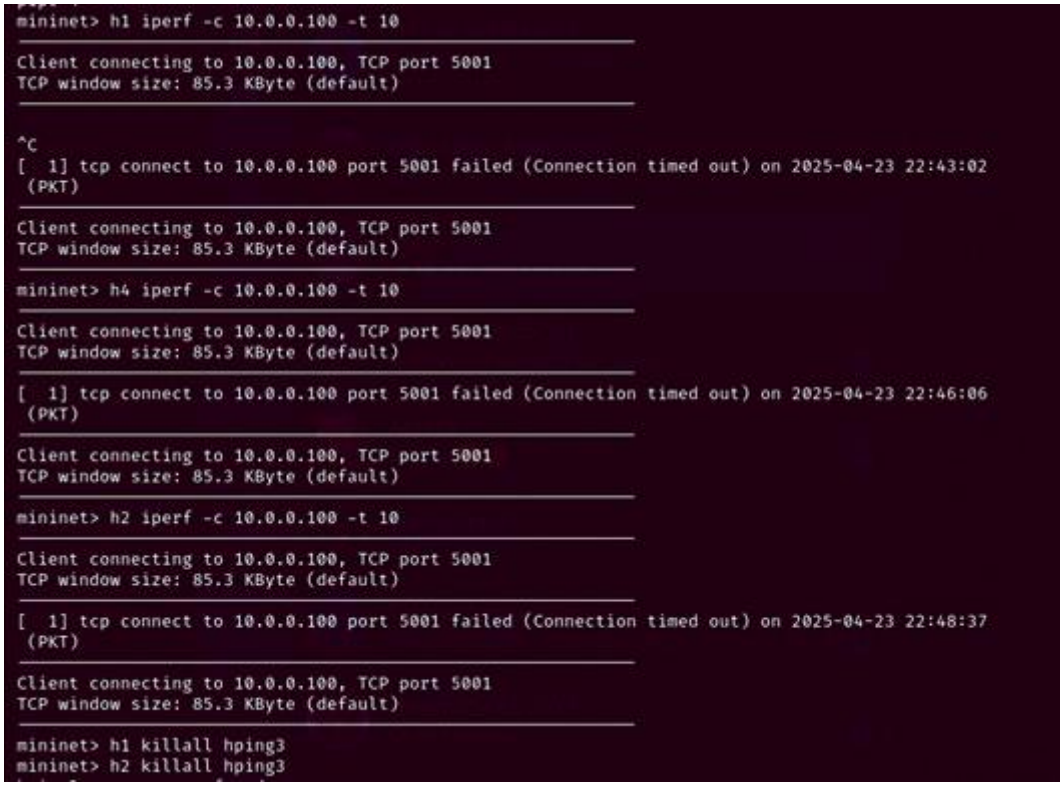
Το αποτέλεσμα δείχνει ότι και τα 10 αιτήματα ICMP echo μεταδόθηκαν, αλλά δεν ελήφθη κανένα ως απάντηση. Αυτό είχε ως αποτέλεσμα την απώλεια πακέτων 100%. Η αδυναμία του στόχου να ανταποκριθεί σε οποιοδήποτε από τα αιτήματα ping υποδεικνύει σαφώς ότι ο διακομιστής δεν ανταποκρινόταν κατά τη στιγμή της δοκιμής.

Συνδεσιμότητα μεταξύ κεντρικών υπολογιστών:



Εικόνα 79: Αποτυχία επικοινωνίας h1 σε h2

Παρακολούθηση εύρους ζώνης:

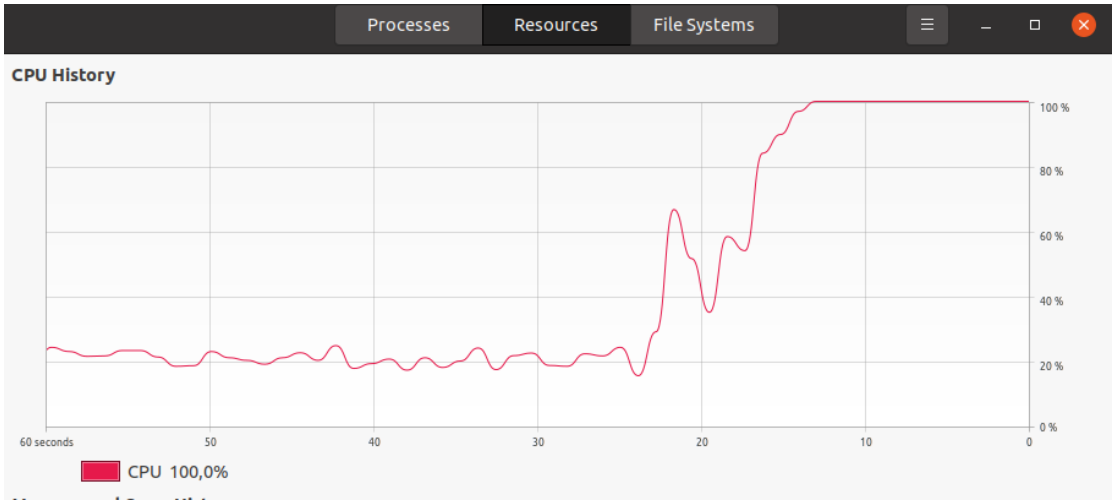


Εικόνα 80: Παρακολούθηση εύρους ζώνης – λήξη χρόνου σύνδεσης λόγω επίθεσης flood

Εκτός από την εκτέλεση εντολής ping εκτελούμε και την εντολή ipetf για να μετρήσουμε την ταχύτητα όμως η σύνδεση τερματίζει με time out μιας και είναι αδύνατη η επικοινωνία μεταξύ του h1 και srv1 .

Ανάλυση της Συμπεριφοράς Δικτύου

Πριν από την επίθεση DDoS, το δίκτυο SDN λειτουργούσε αποτελεσματικά με ελάχιστη καθυστέρηση και σταθερή απόδοση εύρους ζώνης. Μέσω του iperf οι δοκιμές επιβεβαίωσαν σταθερή επικοινωνία μεταξύ των κεντρικών υπολογιστών και μέσω της εντολής ping οι δοκιμές έδειξαν χαμηλούς χρόνους απόκρισης, επικυρώνοντας την σωστή ρύθμιση δικτύου και την προσβασιμότητα του κεντρικού υπολογιστή.



Εικόνα 81: Απότομη αύξηση χρήσης cpu (spike) με την εκτέλεση της εντολής hping3

Όταν η επίθεση DDoS ξεκίνησε χρησιμοποιώντας hping3, υπήρξε αισθητή αύξηση στη χρήση της CPU(όταν ξεκινούν επιθέσεις, η CPU υπερφορτώνεται και το δίκτυο θα αντιμετωπίσει αστάθειες, το οποίο παρατηρείται απλώς κατά την προσομοίωση σε πραγματικό χρόνο)καθώς και απώλεια πακέτων και καθυστέρηση σε όλο το δίκτυο. Οι μετρήσεις εύρους ζώνης μειώθηκαν σημαντικά στον στοχευμένο κεντρικό υπολογιστή, δείχνοντας πώς η κακόβουλη επίθεση προκάλεσε συμφόρηση στη σύνδεση. Αυτό κατέδειξε σαφώς την ευπάθεια του δικτύου σε επιθέσεις που βασίζονται σε όγκο δεδομένων.

Παρά την πίεση, ο ελεγκτής POX συνέχισε να διαχειρίζεται τους κανόνες ροής, επιδεικνύοντας βασική ανοχή σφαλμάτων. Ωστόσο, χωρίς μια μονάδα μετριασμού (π.χ., ένα τείχος προστασίας ή ανιχνευτή ανωμαλιών), το δίκτυο SDN δεν μπορούσε να ανταποκριθεί αυτόνομα στην επίθεση.

6. ΣΥΜΠΕΡΑΣΜΑΤΑ

Μέσα από την εκπόνηση της παρούσας εργασίας απέκτησα ουσιαστική κατανόηση της αρχιτεκτονικής και των δυνατοτήτων του Software Defined Networking (SDN) ή αλλιώς δικτύων καθορισμένων από λογισμικό, καθώς και των προκλήσεων που το συνοδεύουν. Η μελέτη σε θεωρητικό επίπεδο ανέδειξε τη σημασία του διαχωρισμού του επιπέδου ελέγχου από το επίπεδο δεδομένων, στοιχείο που επιτρέπει την ευκολότερη διαχείριση, τον κεντρικό έλεγχο και τη μεγαλύτερη ευελιξία στη λειτουργία των δικτύων.

Σε πρακτικό επίπεδο, μέσω της χρήσης του εξομοιωτή Mininet και του ελεγκτή POX, κατανόησα καλύτερα πώς μπορούν να σχεδιαστούν και να αξιολογηθούν διαφορετικές τοπολογίες δικτύου. Επιπλέον, η προσομοίωση επίθεσης DDoS έδειξε ξεκάθαρα ότι, παρόλο που το SDN παρέχει ισχυρά εργαλεία διαχείρισης και προγραμματισμού, εξακολουθεί να παραμένει ευάλωτο χωρίς την ύπαρξη μηχανισμών ανίχνευσης και άμυνας.

Συνολικά, κατέληξα στο συμπέρασμα ότι το SDN αποτελεί μια πολλά υποσχόμενη τεχνολογία για την ανάπτυξη δικτύων νέας γενιάς, με δυνατότητες που μπορούν να αλλάξουν ριζικά τον τρόπο σχεδίασης και διαχείρισης των υποδομών. Ωστόσο, η ασφάλεια πρέπει να αποτελεί πρωταρχική προτεραιότητα, καθώς οι επιθέσεις και τα τρωτά σημεία μπορούν να υπονομεύσουν τα οφέλη του, ειδικά στην περίπτωση που ο κεντρικός έλεγχος απευθύνεται αποκλειστικά μόνο σε ένα σημείο του δικτύου .

Για μελλοντική έρευνα, θα ήταν ιδιαίτερα ενδιαφέρον να εξεταστεί η χρήση εναλλακτικών SDN controllers, η αξιολόγηση μεγαλύτερων και πιο πολύπλοκων τοπολογιών, καθώς και η ενσωμάτωση σύγχρονων τεχνικών ανίχνευσης και μετριασμού επιθέσεων, προκειμένου να ενισχυθεί η ανθεκτικότητα και η αξιοπιστία των SDN δικτύων σε πραγματικές συνθήκες.

ΑΝΑΦΟΡΕΣ

[1] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2014. [Online]. Available: https://doi.org/10.1109/JPROC.2014.2371999

[2] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1617–1634, 2014. [Online]. Available: https://doi.org/10.1109/SURV.2014.012214.00180

[3] A. Lara, A. Kolasani, and B. Ramamurthy, "Network innovation using OpenFlow: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 493–512, 2014. [Online]. Available: https://doi.org/10.1109/SURV.2013.081313.00105

[4] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," White Paper, 2012. [Online]. Available: https://opennetworking.org/wp-content/uploads/2013/05/wp-sdn-newnorm.pdf

[5] S. Sezer, et al., "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, 2013. [Online]. Available: https://doi.org/10.1109/MCOM.2013.6553676

[6] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 333–354, 2018. [Online]. Available: https://doi.org/10.1109/COMST.2017.2782482

[7] K. Phemius, M. Bouet, and J. Leguay, "DISCO: Distributed multi-domain SDN controllers," in *Proc. IEEE/IFIP NOMS*, 2014, pp. 1–4. [Online]. Available: https://doi.org/10.1109/NOMS.2014.6838229

[8] A. Tootoonchian and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," in *Proc. INM/WREN*, 2010, pp. 1–6. [Online]. Available: https://doi.org/10.1145/1863133.1863136

[9] T. Koponen, et al., "Onix: A distributed control platform for large-scale production networks," in *Proc. OSDI*, 2010, pp. 351–364. [Online]. Available: https://www.usenix.org/legacy/event/osdi10/tech/full_papers/Koponen.pdf

[10] ON.Lab, "ONOS: Open Network Operating System," 2015. [Online]. Available: https://onosproject.org

[11] N. McKeown, et al., "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008. [Online]. Available: https://doi.org/10.1145/1355734.1355746

[12] N. Gude, et al., “NOX: Towards an operating system for networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 3, pp. 105–110, 2008. [Online]. Available: https://doi.org/10.1145/1384609.1384625

[13] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, “Reproducible network experiments using container-based emulation,” in *Proc. ACM CoNEXT*, 2012, pp. 253–264. [Online]. Available: https://conferences.sigcomm.org/co-next/2012/eproceedings/conext/p253.pdf

[14] Mininet Team, “Mininet: An instant virtual network on your laptop,” 2023. [Online]. Available: http://mininet.org

[15] POX Controller, “POX: A Python-based SDN controller,” 2023. [Online]. Available: https://github.com/noxrepo/pox

[16] Ryu SDN Framework, “Ryu: Component-based software-defined networking framework,” 2023. [Online]. Available: https://osrg.github.io/ryu/

[17] Daylight Project, “OpenDaylight: A Linux Foundation Collaborative Project,” 2023. [Online]. Available: https://www.opendaylight.org

[18] Floodlight Controller, “Floodlight Open SDN Controller,” 2023. [Online]. Available: https://github.com/floodlight/floodlight

[19] Open Networking Foundation, “OpenFlow Switch Specification Version 1.5.1,” 2015. [Online]. Available: https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf

[20] Open Networking Foundation, “OpenFlow Management and Configuration Protocol (OF-Config) 1.2,” 2013. [Online]. Available: https://opennetworking.org/wp-content/uploads/2013/02/of-config-1.2.pdf

[21] R. Enns, M. Bjorklund, J. Schoenwaelder, and J. Bierman, “Network Configuration Protocol (NETCONF),” IETF RFC 6241, 2011. [Online]. Available: https://doi.org/10.17487/RFC6241

[22] M. Bjorklund, “YANG – A Data Modeling Language for the Network Configuration Protocol (NETCONF),” IETF RFC 6020, 2010. [Online]. Available: https://doi.org/10.17487/RFC6020

[23] S. Scott-Hayward, G. O’Callaghan, and S. Sezer, “SDN security: A survey,” in *Proc. IEEE SDN4FNS*, 2013, pp. 1–7. [Online]. Available: https://doi.org/10.1109/SDN4FNS.2013.6702553

[24] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, “Security in software defined networks: A survey,” *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2317–2346, 2015. [Online]. Available: https://doi.org/10.1109/COMST.2015.2474118

[25] R. Braga, E. Mota, and A. Passito, “Lightweight DDoS flooding attack detection using NOX/OpenFlow,” in *Proc. IEEE LCN*, 2010, pp. 408–415. [Online]. Available: https://doi.org/10.1109/LCN.2010.5735752

[26] S. A. Mehdi, J. Khalid, and S. A. Khayam, “Revisiting traffic anomaly detection using software-defined networking,” in *Proc. RAID*, 2011, pp. 161–180. [Online]. Available: https://doi.org/10.1007/978-3-642-23644-0_9

[27] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, “Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments,” *Comput. Netw.*, vol. 62, pp. 122–136, 2014. [Online]. Available: https://doi.org/10.1016/j.comnet.2013.10.021

[28] S. M. Mousavi and M. St-Hilaire, “Early detection of DDoS attacks in software defined networks,” in *Proc. IEEE ICC*, 2015, pp. 6489–6494. [Online]. Available: https://doi.org/10.1109/ICC.2015.7249344

[29] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, “AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks,” in *Proc. ACM CCS*, 2013, pp. 413–424. [Online]. Available: https://doi.org/10.1145/2508859.2516684

[30] C. Fan, N. M. Kaliyamurthy, S. Chen, H. Jiang, Y. Zhou, and C. Campbell, “Detection of DDoS Attacks in Software Defined Networking Using Entropy,” *Applied Sciences*, vol. 12, no. 1, Art. 370, 2022. [Online]. Available: https://doi.org/10.3390/app12010370