Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Τεχνών

Μεταπτυχιακό Πρόγραμμα Σπουδών
Ασφάλεια Υπολογιστών και Δικτυών

Μεταπτυχιακή Διατριβή

IMPROVING NETWORK SECURITY THROUGH SDN: A COMPARATIVE STUDY OF SOFTWARE DEFINED NETWORKS VS TRADITIONAL ETHERNET NETWORKS

Κωνσταντίνος Αδαμίδης

Επιβλέπων Καθηγήτρια
Αδαμαντίνη Περατικού

Μάιος 2021

Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Τεχνών

Μεταπτυχιακό Πρόγραμμα Σπουδών
Ασφάλεια Υπολογιστών και Δικτυών

Μεταπτυχιακή Διατριβή

IMPROVING NETWORK SECURITY THROUGH SDN: A COMPARATIVE STUDY OF SOFTWARE DEFINED NETWORKS VS TRADITIONAL ETHERNET NETWORKS

Κωνσταντίνος Αδαμίδης

Επιβλέπων Καθηγήτρια
Αδαμαντίνη Περατικού

Η παρούσα μεταπτυχιακή διατριβή υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων για απόκτηση μεταπτυχιακού τίτλου σπουδών στην Ασφάλεια Υπολογιστών και Δικτυών από τη Σχολή Θετικών και Εφαρμοσμένων Τεχνών του Ανοικτού Πανεπιστημίου Κύπρου.

Μάιος 2021

ΛΕΥΚΗ ΣΕΛΙΔΑ

Περίληψη

Με την όλο και αυξανομένη χρήση των δικτυών καθορισμένα από λογισμικό (Software Defined Networks) υπάρχει η ανάγκη για περεταίρω μελέτη για την ασφάλεια που προσφέρουν σε σύγκριση με τα παραδοσιακά δίκτυα (traditional networks).

Σκοπός της μεταπτυχιακής διατριβής είναι να αξιολογήσει την αξιοπιστία των δικτυών SDN σε διάφορους τύπους επιθέσεων, όπου ξεκινώντας αναλύουμε την τεχνολογία των SDN: Software Defined Networking και στη συνέχεια με την χρήση ενός προσομοιωτή δικτύου (Mininet) για να προσδιορίσουμε αν μπορεί να προσφέρει την ζητούμενη ασφάλεια ή οποιεσδήποτε άλλες βελτιώσεις στην ασφάλεια του δικτύου.

Εν συνέχεια, μελετήθηκαν μέσω βιβλιογραφικής ανασκόπησης οι υπάρχουσες εφαρμογές SDN που σχεδιάστηκαν για παροχή ενισχυμένης ασφάλειας, και τέλος, υλοποιήθηκε μια επίθεση στο SDN αλλά και στο παραδοσιακό δίκτυο.

Τα αποτελέσματα της μεταπτυχιακής διατριβής καταγράφονται μέσα σε αυτήν.

Ευχαριστίες

Για την εκπόνηση της μεταπτυχιακής διατριβής αυτής θα ήθελα να ευχαριστήσω την επιβλέπων καθηγήτρια που με την υπομονή και επιμονή της βοήθησε αρκετά για να εκπονηθεί, όπως κι επίσης την σταθερότητα/στωικότητα της να με καθοδήγηση με το σωστό τρόπο.

Αφιέρωση

φίλους και γονείς

ιδιαίτερα στα παιδιά μου

Μαρία και Νεφέλη

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΕΧΟΜΕΝΑ ΕΙΚΟΝΩΝ

Κεφάλαιο 1
Εισαγωγή

“Βελτίωση της ασφάλειας του δικτύου μέσω δικτύου που καθορίζεται από λογισμικό: μια συγκριτική μελέτη του δικτύου που καθορίζεται από λογισμικό έναντι των παραδοσιακών δικτυών ethernet.” (improving network security through SDN: a comparative study of SDN vs traditional ethernet networks) είναι το θέμα της μεταπτυχιακής διατριβής μου και όπως φυσικά γνωρίζουμε ζούμε σε μια εποχή όπου θα χαρακτηριζόταν ως ο τεχνολογικός αιώνας μιας κι εφόσον η τεχνολογική ανάπτυξη είναι ραγδαία και γι’ αυτό το λόγο θεωρώ ότι υπάρχει μεγάλο επιστημονικό ενδιαφέρον για ανάλυση.

Ζούμε σε μια εποχή, όπου η τεχνολογία παίζει έναν πρωταγωνιστικό ρόλο όπου πολλά πράγματα γίνονται, κινούνται και διαχειρίζονται μέσω αυτής. Στην σημερινή κοινωνία του τεχνολογικού αιώνα που ζούμε, πολλοί από την παιδική ακόμη ηλικία, ασχολούνται με την τεχνολογία και έρχονται σε επαφή σε διάφορες περιστάσεις κάνοντάς την όλο και πιο απαραίτητη για την καθημερινότητα τους χαρακτηρίζοντας ολόκληρη την γενιά τους σαν πιο Digital Native από την προηγούμενη.

Μέσα από αυτήν εκτελούν διάφορες εργασίες είτε επαγγελματικές είτε προσωπικές όπως ανταλλαγή μηνυμάτων SMS, μηνυμάτων ηλεκτρονικού ταχυδρομείου, επικοινωνία με διάφορες υπηρεσίες αλλά συναλλαγές με τράπεζες κλπ. είναι μερικές από τις καθημερινές ανθρώπινες δραστηριότητες χρησιμοποιώντας διάφορα τεχνολογικά μέσα.

Ειδικά στις μέρες μας, που όλη η κοινωνία έχει πληγεί από την πανδημία του κορονοϊού, οι άνθρωποι αναγκαστικά εκτελούν εργασίες ή εκπαιδεύονται μέσω διαδικτύου όπου μέσα από αυτές τις δραστηριότητες και συναλλαγές μεταφέρονται και ανταλλάσσονται ευαίσθητα προσωπικά δεδομένα, τα οποία γίνονται εύκολη λεία για κάποιον που θέλει

να κάνει οποιασδήποτε μορφής απάτης (π.χ. hackers), οι οποίοι υποκλέπτουν αυτά τα στοιχεία με απρόβλεπτες συνέπειες.

Δεν είναι λίγες οι περιπτώσεις που ακούμε σε καθημερινή βάση για ηλεκτρονικές απάτες μέσω τραπεζών ή ηλεκτρονικού ταχυδρομείου. Αλλά ακόμη κι στα μέσα μαζικής ενημέρωσης ακούμε διάφορες περιπτώσεις ανθρώπων που έπεσαν θύματα εκβιαστών, λόγω του ότι διέρρευσαν προσωπικά τους στοιχεία. Επιπλέον αρκετός κόσμος, κυρίως παιδικής και εφηβικής ηλικίας έπεσαν θύματα ηλεκτρονικού εκφοβισμού (cyber-bulling) λόγω διαρροής φωτογραφιών από προσωπικές τους στιγμές. Ενήλικες αλλά ακόμη και διάφορα άτομα σε ευαίσθητη ηλικία (παιδιά, έφηβοι, νέοι) όλο και περισσότερο βυθίζονται στον κόσμο της τεχνολογίας όπου κοινωνικά δίκτυα, εκπαιδευτικές πλατφόρμες και παιχνίδο-κοινωνίες (gaming communities) καθημερινά διακινούν τα προσωπικά μας στοιχεία που δεν είναι δύσκολο να πέσουν σε λάθος χέρια.

Σκεπτόμενος λοιπόν όλα τα παραπάνω θέματα ασφάλειας δικτυών, επέλεξα το θέμα της μεταπτυχιακής διατριβής μου να είναι η υλοποίηση ενός SDN (Software Defined Networking) περιβάλλοντος για να μελετήσουμε κατά πόσο μπορεί να προσφέρει στα δίκτυα την απαιτούμενη ασφάλεια ή οποιαδήποτε άλλη βελτίωση, αφού πρώτα όμως μελετήσουμε τις υφιστάμενες επιλογές.

1.1 Αναγκαιότητα και Σπουδαιότητα Έρευνας

Όπως θα αναφερθεί στα επόμενα κεφάλαια, το σενάριο που θα αναλυθεί στην παρούσα μεταπτυχιακή διατριβή είναι η υλοποίηση ενός δικτύου καθορισμένου από λογισμικό (SDN - software defined networking) και θα μελετηθεί κατά πόσο είναι αξιόπιστο σε σύγκριση με τα παραδοσιακά δίκτυα (traditional ethernet networks) σε διάφορους τύπους επιθέσεων.

Αυτό αποδεικνύει την αναγκαιότητα πραγματοποίησης και ολοκλήρωσης αυτής της έρευνας, καθώς οι κίνδυνοι από ξένους «εισβολείς» σε διάφορα δίκτυα είτε καθορισμένο από λογισμικό είτε όχι (δηλαδή παραδοσιακά δίκτυα) είναι ένας κίνδυνος που υπάρχει συνέχεια. Άγνωστοι «εισβολείς» είναι έτοιμοι να επιτεθούν σε έναν οργανισμό, υπηρεσία κλπ. και να υποκλέψουν στοιχεία πελατών, στοιχεία του προσωπικού κλπ. με απρόβλεπτες συνέπειες (νομικές, ψυχολογικές, οικονομικές).

Αυτό που θα αναλυθεί στη συνέχεια της μεταπτυχιακής διατριβής, είναι πώς ένα δίκτυο καθορισμένο από λογισμικό (SDN - software defined networking) μπορεί να υλοποιηθεί κι να χρησιμοποιηθεί, σε μια προσπάθεια να ελαχιστοποιηθούν οι απάτες. Αυτό σαν μια πρώτη σκέψη, θα μπορούσε να επιτευχθεί ανακατευθύνοντας ή απομονώνοντας τα πακέτα και παράλληλα την απομόνωση των κακόβουλων συσκευών. Επιπλέον, θα γίνει προσπάθεια να επιτευχθεί το παραπάνω με τη χρήση την του προσομοιωτή δικτύου Mininet. Μέσω αυτή της τεχνολογίας θα γίνει μια προσομοίωση κυκλοφορίας του δικτύου, διακοπτών (switches), ελεγκτών (controllers) και συνδέσμων (links).

Επιπρόσθετα μέσω του προσομοιωτή δικτύου, θα χρησιμοποιηθούν οι δυνατότητες του SDN, ώστε να γίνεται ένα φιλτράρισμα της κίνησης των συμμετεχόντων που επιχειρούν να αλληλοεπιδράσουν με τους κόμβους.

Το παρόν εγχείρημα είναι πάρα πολύ σημαντικό και καινοτόμο κι αυτό αναδεικνύει την αναγκαιότητα της παρούσας έρευνας σε μια προσπάθεια εξέλιξης και βελτιστοποίησης του υφιστάμενου τρόπου λειτουργίας της ασφάλειας των δικτύων και συστημάτων.

1.2 Διάταξη Μεταπτυχιακής Διατριβής

Όπως θα δούμε στην συνέχεια της μεταπτυχιακής διατριβής η οποία θα χωριστεί σε έξι κεφάλαια, όπου το πρώτο κεφαλαίο ανήκει στην εισαγωγή που είναι μαζί με την αναγκαιότητα και σπουδαιότητα της έρευνας.

Επιπρόσθετα, το δεύτερο κεφαλαίο ανήκει στην βιβλιογραφική ανασκόπηση όπου αρχικά γίνεται μια ιστορική αναδρομή μαζί με διάφορες γενικές πληροφορίες και εν συνέχεια βλέπουμε κάποιες σχετικές εργασίες.

Επιπλέον, το τρίτο κεφαλαίο ασχολείται με την μεθοδολογία όπου θα δούμε τον σκοπό και το είδος της έρευνας αλλά και τα βασικά ερευνητικά ερωτήματα μαζί με την μεθοδολογία υλοποίησης, ακολούθως, το τέταρτο κεφαλαίο θα ασχοληθεί με την υλοποίηση της μεταπτυχιακής διατριβής και εν συνέχεια το πέμπτο κεφαλαίο θα έχει τα αποτελέσματα αυτής.

Το τελευταίο κεφαλαίο είναι ο επίλογος.

Κεφάλαιο 2
Βιβλιογραφική Ανασκόπηση

Στο κεφάλαιο που ακολουθεί θα ασχοληθούμε με την βιβλιογραφική ανασκόπηση η οποία είναι μια συλλογή από επιλεγμένες δημοσιευμένες πηγές σχετικές με το θέμα της μεταπτυχιακής διατριβής όπου το αντικείμενο έρευνας είναι η "Βελτίωση της ασφάλειας του δικτύου μέσω δικτύου που καθορίζεται από λογισμικό: μια συγκριτική μελέτη του δικτύου που καθορίζεται από λογισμικό έναντι των παραδοσιακών δικτυών ethernet." (improving network security through SDN: a comparative study of SDN vs traditional ethernet networks) όπου θα συνοδεύσουμε το σχολιασμό αλλά και την κριτική ανάλυση των περιεχομένων και θα παραθέσουμε σε ορισμένες περιπτώσεις τα βασικά συμπεράσματα μας.

Ακολούθως θα κάνουμε αρχικά μια ιστορική αναδρομή σε μια σημαντική τεχνολογική καινοτομία η οποία ονομάζεται Software Defined Networks (SDN) δηλαδή τα δίκτυα που καθορίζονται από λογισμικό κι εν συνέχεια θα κάνουμε μια ιστορική αναδρομή στα παραδοσιακά δίκτυα (traditional networks).

Εν συνέχεια, θα δούμε για τα SDN όπου το επίπεδο ελέγχου (control plane) μας δίνει μια οπτική αναπαράσταση *της υποδομής του δικτύου* ενώ το επίπεδο δεδομένων (data plane) δίνει μια οπτική αναπαράσταση *της κίνησης των χρηστών.* Επιπλέον, θα δούμε τα δυο αυτά επίπεδα (επίπεδο ελέγχου/δεδομένων) με τη χρήση του πρωτοκόλλου Εποπτικού Ελέγχου και Απόκτησης Δεδομένων (SCADA - Supervisory Control and Data Acquisition Protocol).

Επίσης θα δούμε, τα αυτόνομα συστήματα (ΑΣ) και την σχέση τους με ειδικά ολοκληρωμένα κυκλώματα (ASICs – Application Specific Integrated Circuit)

Επιπρόσθετα θα κάνουμε μια αναδρομή, στο OpenFlow όπου συνήθως οι όροι “SDN” και “OpenFlow” χρησιμοποιούνται συχνά εναλλάξ μέσα στην επιστημονική κοινότητα αλλά και γενικότερα μέσα σε μεγάλες τεχνολογικές εταιρείες όπως η Cisco και η IBM που χρησιμοποιούν αυτή την ιδεολογία.

Ακολούθως και εν συντομία θα δούμε για την ασφάλεια του SDN όπου συνήθως ο administrator του SDN αποτελεί εύκολο στόχο για διάφορες επιθέσεις από τον κυβερνοχώρο.

Εν συνέχεια θα δούμε ότι όσο περισσότερο γίνεται η πραγματοποίηση υλοποιήσεων SDN (SDN implementations) τόσο ακόμη θα επιτρέπεται στους προμηθευτές δικτυών (Cisco, IBM κλπ.) να προσφέρουν αποτελεσματικότερα το NaaS μέσω του cloud.

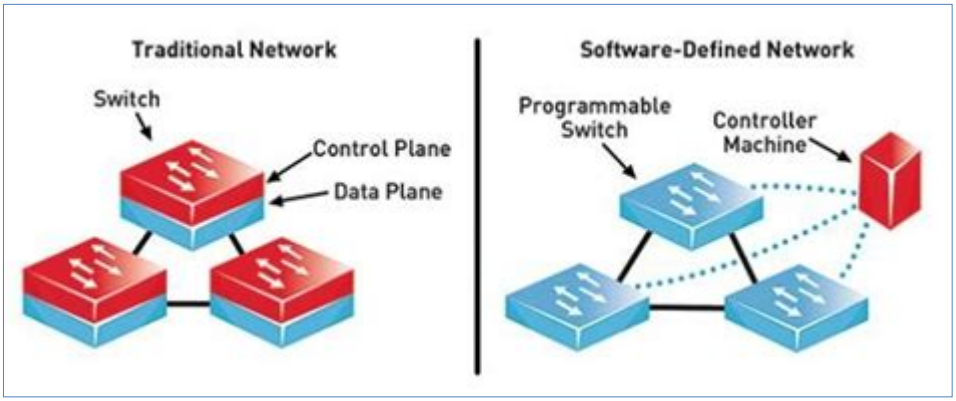
Επίσης θα δούμε, κάποιους γνωστούς κολοσσιαίους προμηθευτές δικτυών (Cisco, IBM, Microsoft, Brocade, Juniper, Citrix, NEC, Nuage Networks, or Red Hat) να συνεργάζονται με την κοινοπραξία του Linux Foundation όπου το ονόμασαν το ερευνητικό τους έργο “Open Daylight” που ουσιαστικά είναι μέχρι κι σήμερα ερευνητικό έργο ανοιχτού κώδικα.

Επιπλέον, θα δούμε ότι το Mininet σαν εξομοιωτής δικτύου δημιουργήθηκε για να επιτρέψει την έρευνα στο SDN και στο OpenFlow αλλά και να επιτρέπει την αλληλεπίδραση του μη τροποποιημένου κώδικα σε εικονικό υλικό σε έναν απλό υπολογιστή αλλά ταυτόχρονος να παρέχει ευκολία και ρεαλισμό με πολύ χαμηλό κόστος.

Και τελικά να καταλήξουμε να δούμε, τους οπτικούς πίνακες ελέγχου (visual dashboard) [1] - όπου οι πωλητές και οι παραγωγοί λογισμικού προσπαθούν να κάνουν τα πάντα όσο πιο οπτικά γίνεται, για να πετύχουν τους στόχους να προσφέρουν μια καλύτερη “αισθητική διεπαφή”.

2.1 Ιστορική Αναδρομή και Γενικές Πληροφορίες

Ας ξεκινήσουμε λοιπόν με την ιστορική αναδρομή για την τεχνολογική καινοτομία που προαναφέραμε προηγουμένως, δηλαδή τα δίκτυα που καθορίζονται από λογισμικό (SDN - Software Defined Networks)



Εικ. 2.1 Traditional Network vs Software Defined Network [2]

2.1.1 Software Defined Network

Το SDN (Software Defined Network) [3] κάνει ένα δίκτυο πιο ευέλικτο και πιο εύκολο στη διαχείριση του κι αυτό άλλωστε είναι ο σχεδιασμός της αρχιτεκτονικής αυτής.

2.1.2 Η Ιστορία του SDN

Ένας καλύτερος τρόπος για να κατανοήσουμε πλήρως τι ακριβώς είναι το **SDN (Software Defined Network)** είναι να επιστρέψουμε στην αρχή (τέλη του 20ου αιώνα) και να ρίξουμε μια ματιά από πού προήλθε και ξεκίνησε [4].

Τα πρώτα παραδείγματα αυτού του τύπου ιδεολογίας είναι τα πρώτα συστήματα δικτύωσης (networking systems) στα οποία μεμονωμένοι οικοδεσπότες (individual hosts) μπορούσαν να μιλήσουν σε ένα κεντρικό πλαίσιο (central mainframe computer όπου είναι ένας τύπος υπολογιστή γενικότερα γνωστός για το μεγάλο μέγεθος του, τον αποθηκευτικό του χώρο, την ισχύ επεξεργασίας του και το υψηλό επίπεδο αξιοπιστίας του) [5].

Αυτές οι ρυθμίσεις, ήταν πρωτοποριακές στα τέλη του 20ου αιώνα αλλά με βάση τα σημερινά πρότυπα θεωρούνται αρκετά πρωτόγονες έτσι ώστε θα μπορούσαμε να πούμε και να τις περιγράψουμε ως “μονόδρομη τροφοδοσία” (one way feedforward) δηλαδή

στο ότι οι επικοινωνίες πήγαιναν με μόνο μια διαδρομή. Η διαδικασία αποσύνδεσης του **επιπέδου ελέγχου (control plane)** και του **επιπέδου δεδομένων (data plane)** δεν είχαν ακόμη ολοκληρωθεί και ήταν σε πολύ πρόωρα στάδια.

> Το επίπεδο ελέγχου δηλαδή το control plane είναι το τμήμα ενός δικτύου που μεταφέρει τις απαραίτητες πληροφορίες για την δημιουργία και τον έλεγχο του δικτύου αλλά και επίσης είναι μέρος του θεωρητικού πλαισίου που χρησιμοποιείται για την κατανόηση της ροής πακέτων πληροφοριών μεταξύ διεπαφών δικτύου (network interfaces).
>
> Το επίπεδο δεδομένων δηλαδή το data plane είναι το μέρος ενός δικτύου μέσα από το οποίο μεταδίδονται πακέτα χρηστών (user packets) δηλαδή χρησιμοποιείται για την έννοια της ροής των πακέτων δεδομένων μέσω μιας υποδομής δικτύου (network infrastructure).
>
> Με αλλά λόγια, τα δυο πιο πάνω συχνά περιλαμβάνονται σε διαγράμματα και εικόνες για να δώσουν μια οπτική αναπαράσταση όπου το **επίπεδο ελέγχου (control plane)** δίνει μια οπτική αναπαράσταση *της υποδομής του δικτύου* ενώ το **επίπεδο δεδομένων (data plane)** δίνει μια οπτική αναπαράσταση *της κίνησης των χρηστών.* [6]

Ορισμένοι ερευνητές προσφέρουν το παράδειγμα του **δημόσιου τηλεφωνικού δικτύου (public switched telephone network - PSTN)** ως έναν από τους πρώτους τρόπους με τους οποίους οι μηχανικοί άρχισαν να διαχωρίζουν τα επίπεδα ελέγχου και δεδομένων για να μπορέσουν να τα διαμορφώσουν όπου και τελικά εφαρμόστηκε και έγινε η SDN στρατηγική. Κάποιοι άλλοι ερευνητές παρέχουν τη χρήση δικτύων peer-to-peer της δεκαετίας του 1980 και την άνοδο του μικρο-ϋπολογισμού (micro-computing) ως μέρος της βάσης για τις σύγχρονες αρχές δικτύωσης του SDN [7].

Ουσιαστικά, οι επιστήμονες άρχισαν να εργάζονται με την ιδέα ότι δεν είναι απαραίτητο το κάθε σήμα δρομολόγησης (routing signal) να διαμένει στον κόμβο του (node) αλλά να ενσωματώσει αυτήν τη φιλοσοφία σε διαφορετικά συστήματα (diverse systems). Ένας απλούστερος τρόπος για να το εξηγήσουμε αυτό είναι ότι με την πάροδο του χρόνου, οι επιστημονικές κοινότητες και οι κοινότητες πληροφορικής μπόρεσαν να καινοτομήσουν

σιγά-σιγά το τρόπο λειτουργείας των δίκτυών και το πού ακριβώς βρίσκονται οι διαδικασίες ελέγχου (control processes).

Όταν άρχισαν να συμμετέχουν μεγάλες εταιρίες στο τομέα των δικτυών όπως η Cisco, η όλη διαδικασία βελτίωσης των SDN επικροτήθηκε. Επιπρόσθετα, απογειώθηκαν ο τομέας του cloud computing και η εικονικοποίηση (virtualization) έγινε σταθερή αξία στην τεχνολογική κοινότητα κάνοντας τα SDN μια πιο ζωντανή αρχή (πειθαρχεία) μέσα στη κατηγορία των "λογισμικών που καθορίζεται από τον IT τομέα".

Καθώς γεννήθηκε το cloud computing, αναπτύχθηκε γρήγορα ως επαναστατικός τρόπος για να προσφέρει σχεδόν οποιοδήποτε είδος υπηρεσίας αλλά οι διαχειριστές δικτύου (network administrators) είδαν το cloud να ανοίγει το δρόμο για τα μοντέλα δικτύου που καθορίζονταν από το λογισμικό (software defined network models) νωρίτερα από το cloud computing.

Όλα αυτά τα κομμάτια έπρεπε να προέρχονται από στην ίδια τοποθεσία υλικού, οπότε δεν υπήρχε η ίδια εστίαση στην καινοτομία ορισμένων από αυτούς τους τύπους πλαισίων (frameworks).

Το cloud computing, το λογισμικό ως υπηρεσία (software as a service - SaaS) και η εικονικοποίηση (virtualization) προκάλεσαν γενικά πολλές εργασίες σχεδιασμού που οδήγησαν πολλούς επιστήμονες δεδομένων (data scientists) να εξετάσουν το ενδεχόμενο διαχωρισμού του **επιπέδου ελέγχου** από το **επίπεδο δεδομένων** με πιο ακριβείς τρόπους για τους οποίους μιλάμε με το SDN.

2.1.3 Η Ιστορία του Παραδοσιακού Δικτύου

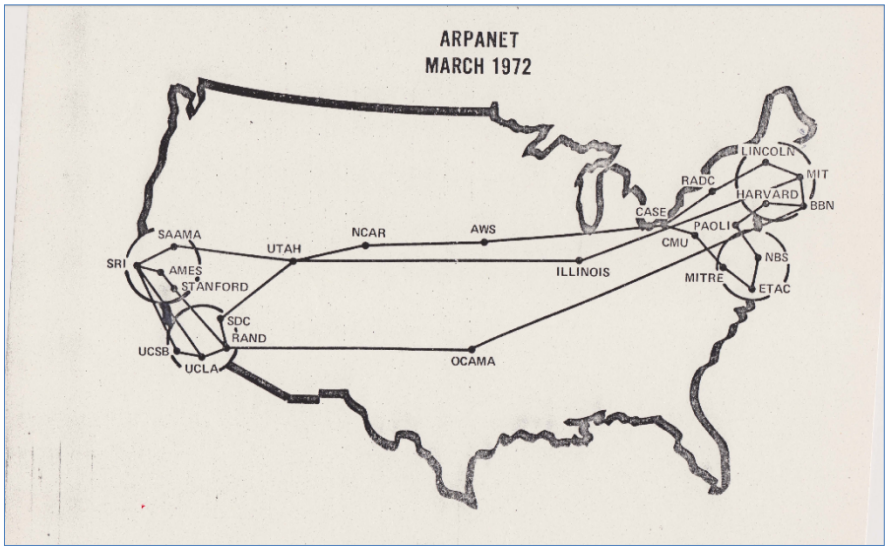
Η ιστορία του παραδοσιακού δικτύου **(traditional networks)** σαν γενικότερη ιδέα ξεκινάει μερικές δεκαετίες πολύ πιο πριν από τα τέλη του 20ου αιώνα που ξεκίνησαν τα πρώτα δίκτυα καθορισμένα από λογισμικό (software defined network). [8]

Αρχικά ήταν το **Telex** (γνωστό σαν teleprinters) το οποίο ξεκίνησε στη Γερμανία ως ερευνητικό πρόγραμμα κι ανάπτυξης το **1926**, το οποίο έγινε μια λειτουργική υπηρεσία τηλεμετατροπής το **1933**. Η υπηρεσία αυτή, που αρχικά διαχειριζόταν το Reichspost (ταχυδρομική υπηρεσία Reich – με αλλά λόγια η κρατική υπηρεσία ταχυδρομείου της

Γερμανίας) είχε ταχύτητα 50 baud (ταχύτητα μετάδοσης όπως το bits per second) - περίπου 66 λέξεις ανά λεπτό. Η υπηρεσία Telex εξαπλώθηκε αρχικά στην Ευρώπη (κυρίως μετά το 1945), ουσιαστικά μετα το τέλος του **Δεύτερου Παγκοσμίου Πολέμου**, και μετα από ένα με δυο χρόνια σε όλο τον κόσμο. Μέχρι το 1978, η Δυτική Γερμανία συμπεριλαμβανομένου του Δυτικού Βερολίνου, είχε εκατοντάδες χιλιάδες συνδέσεις τέλεξ και πολύ πριν από τη διάθεση της αυτόματης τηλεφωνίας οι περισσότερες χώρες όπως στην Κεντρική Αφρική και Ασία, είχαν τουλάχιστον μερικές χιλιάδες συνδέσεις τέλεξ υψηλής συχνότητας. [9]

Εικ. 2.2 Telex (1926) – SIEMENS [10]

Επιπρόσθετα, το **1961** ο **Leonard Kleinrock** στην πρόταση του για διδακτορικό με τίτλο “Ροή πληροφοριών με μεγάλα δίκτυα επικοινωνίας” (“Information Flow in large communication networks.”) [11] στο MIT των ΗΠΑ έδωσε την ιδεολογία του στο **ARPANET** (Advance Research Projects Agency Network) όπου πραγματοποίησε τα πρώτα Δίκτυα Ευρείας Περιοχής (Wide Area Networks – WAN) το 1972.



Εικ. 2.3 ARPANET (1972) – MAP [12]

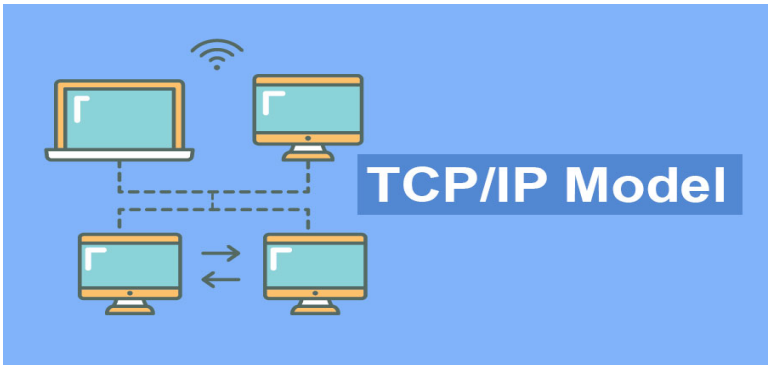
Παράλληλα με τα πιο πάνω το **1971**, ο **Ray Tomlinson** (προγραμματιστής) στέλνει το πρώτο ηλεκτρονικό μήνυμα αλληλογραφίας μεταξύ χρηστών σε διαφορετικούς κεντρικούς υπολογιστές που είναι συνδεδεμένοι στο **ARPAnet** όπου χρησιμοποίησε το σύμβολο **@ (παπάκι)** για να διαχωρίσει τον χρήστη από το μηχάνημά του, το οποίο χρησιμοποιείται από τότε σε διευθύνσεις ηλεκτρονικού ταχυδρομείου.

Εικ. 2.4 Ray Tomlinson @ (1971) [13]

Εν συνέχεια, το **1981** το **Transmission Control Protocol/Internet Protocol version 4 (TCP/IP v4)** το πρωτόκολλο δικτύου που ορίζεται στο **RFC791** που είναι η κύρια μετάδοση δικτύου Ethernet και γενικότερα του Διαδικτύου έκανε την δική του εμφάνιση για να μείνει για τα καλά στη ζωή μας μέχρι και σήμερα.

Όμως όλα στην τεχνολογία δεν γεννήθηκαν με την τέταρτη έκδοση τους αλλά ουσιαστικά η ιστορία του TCP (Transmission Control Protocol) ξεκινάει το **1973** με την πρώτη έκδοση να σχεδιάζεται και να τεκμηριώνετε με το **RFC675** για να δώσει την σκυτάλη της το **Μάρτιο του 1977** στην δεύτερη έκδοση με εξέλιξη μέχρι τον **Αύγουστο** της τρίτης έκδοσης όπου σε κάποιο στάδιο ο **Jon Postel** να παντρέψει τα δυο πρωτοκολλά TCP και IP για να μας δώσει την τέταρτη έκδοση.

Με την πέμπτη έκδοση, οι επιστημονική κοινότητα προσπάθησε να βρει έναν τρόπο μετάδοσης φωνής μέσω δικτύων εναλλαγής πακέτων, πράγμα που αρχικά απέτυχε, αλλά σε κοινοπραξία μεγάλων τεχνολογικών κολοσσών που τεκμηριώνετε στο **RFC1190** η ιδέα της ροής βίντεο και άλλων νέων μέσων γίνεται πραγματικότητα. [14]



Εικ. 2.5 TCP/IP MODEL (1981-today) [15]

2.2 BGP και OSPF

"Θα είναι ενδιαφέρον να παρακολουθήσουμε και να δούμε αν οι εταιρείες δικτύωσης πρόκειται να γίνουν εταιρείες λογισμικού", και επίσης, "δεν βρίσκουν όλοι το BGP και το OSPF τόσο συναρπαστικό όσο εγώ." είναι τα δύο χαρακτηριστικά αποσπάσματα που αναφέρονται στο [16], [17] όπου ειπώθηκαν από τον David Ward της Cisco στο Open Networking Summit to 2011.

Όπου το BGP (Border Gateway Protocol) [18], [19] είναι ένα τυποποιημένο εξωτερικό πρωτόκολλο πύλης που έχει σχεδιαστεί για την ανταλλαγή πληροφοριών δρομολόγησης και προσβασιμότητας μεταξύ αυτόνομων συστημάτων (AS) στο διαδίκτυο, και εν συνέχεια, το Open Shortest Path First (OSPF) [20] είναι ένα πρωτόκολλο δρομολόγησης για δίκτυα πρωτοκόλλου Internet (IP).

2.2.1 Βασικές Διάφορες Ανάμεσα στο BGP και το OSPF

Αρχικά να πούμε ότι είναι και τα δύο πρωτοκολλά δρομολόγησης (routing protocols), το Open Shortest Path First (OSPF) είναι εσωτερικής πύλης (interior gateway) σε αντίθεση με το Border Gateway Protocol (BGP) που είναι εξωτερικής πύλης (exterior gateway). [21]

Από την μια, το OSPF βασίζεται στη δρομολόγηση κατάστασης σύνδεσης (Link State Routing) όπου κάθε δρομολογητής στέλνει την κατάσταση του γειτονικού δρομολογητή σε κάθε δρομολογητή που υπάρχει στην περιοχή.

Από την άλλη, το BGP βασίζεται σε δρομολόγηση διανύσματος διαδρομής (Path Vector Routing) όπου ένας δρομολογητής έχει μια λίστα δικτύων που μπορούν να προσεγγιστούν με τη διαδρομή για να φτάσουν σε καθένα από αυτά.

Παραθέτω πιο κάτω συγκριτικό πίνακα μεταξύ OSPF και BGP με τις διάφορες μεταξύ των δυο αυτών πρωτοκόλλων δρομολόγησης (routing protocols).

#	Βάση Σύγκρισης	OSPF	BGP
1.	Ορισμός	Open Shortest Path First	Border Gateway Protocol
2.	Εφαρμογή	Εύκολη	Δύσκολη
3.	Σύγκλιση (Convergence)	Γρήγορη	Αργή
4.	Τοπολογία (σχεδιασμού)	Σχεδιασμός ιεραρχικού δικτύου	Πλέγματος (mesh).
5.	Πρωτόκολλο	Εσωτερικής πύλης	Εξωτερικής πύλης
6.	Πρωτόκολλο Διαδικτύου	IP	TCP
7.	Port Number (αριθμό θύρας)	89	179
8.	Τύπος Κατάστασης	Σύνδεσης (Link State)	Διανυσματικής (Path Vector)
9.	Αλγόριθμος	Dijkstra	Best Path
10.	Λειτουργία (Function)	Ταχύτερη διαδρομή παρά τη μικρότερη διαδρομή	Προτιμά το καλύτερο μονοπάτι

Πίνακας 2.6 ΣΥΓΚΡΙΤΙΚΟΣ ΠΙΝΑΚΑΣ

2.3 Επίπεδο Ελέγχου/Δεδομένων

Το SDN δεν είναι κάτι απόλυτο ή δύσκαμπτο όσον αφορά την κατασκευή του - είναι μια κεντρική ιδέα που την χρησιμοποιούν σε παγκόσμιο επίπεδο οι εταιρείες για να φέρουν επανάσταση στα διαφορά προϊόντα και υπηρεσίες της πληροφορικής.

Τα SDN είναι μια νέα αναπτυσσόμενη τεχνολογία η οποία αλλάζει πλήρως την αρχιτεκτονική και τη λειτουργία των παραδοσιακών δικτύων. Διαχωρίζοντας το επίπεδο δεδομένων (**data plane**) από το επίπεδο ελέγχου (**control plane**), η διαχείριση γίνεται κεντρική οδηγώντας στη δημιουργία δυναμικών και ευέλικτων δικτύων. [4] [5]

2.3.1 Επίπεδο Δεδομένων (Data Plane)

Το επίπεδο δεδομένων (data plane) του δικτύου διαχειρίζεται την προώθηση πακέτων μέσω του δικτύου (user packets) όπου πιο πριν στα παραδοσιακά συστήματα, οι έλεγχοι

είχαν ενσωματωθεί στις ίδιες οδούς με τα στοιχεία προώθησης δεδομένων. Πλέον αυτό που κάνουν όλοι οι κόμβοι επιπέδου δεδομένων (data plane nodes) είναι να προωθούν τα πακέτα (forward packets). [6]

2.3.2 Επίπεδο Ελέγχου (Control Plane)

Το επίπεδο ελέγχου (control plane) του δικτύου είναι το τμήμα "δρομολόγησης" όπου θα δρομολογήσει τα δεδομένα πακέτων και θα διαμορφώσει τον τρόπο με τον οποίο το δίκτυο μετακινεί δεδομένα άρα ουσιαστικά χρησιμοποιείται για την κατανόηση της ροής πακέτων πληροφοριών μεταξύ διεπαφών δικτύου (network interfaces). [6] [7]

2.3.3 Επίπεδο Ελέγχου/Δεδομένων με την χρήση SCADA protocol

Η επιστημονική κοινότητα αναφέρεται στο επίπεδο ελέγχου (control plane) σαν "τον εγκέφαλο" κι αντίστοιχα για το επίπεδο δεδομένων (data plane) σαν "το μυώνα" του δικτύου, όπου σίγουρα το καθήκον της απλής προώθησης πακέτων είναι πολύ περισσότερο από μια διαδικασία συνεννόησης από τη δρομολόγηση, έτσι ο χωρισμός αυτών των δύο να οδηγεί σε τέτοιου είδους συγκρίσεις. [22] [23]

Ένας εναλλακτικός δεύτερος τρόπος είναι να δούμε και να διερευνήσουμε πώς χρησιμοποιούνται τα δυο αυτά επίπεδα (ελέγχου/δεδομένων) στη βιομηχανική διαδικασία, όπου σε πολλές περιπτώσεις εφαρμόζονται σαν προγραμματιζόμενοι ελεγκτές λογικής (logic controllers) για την διαχείριση του τι συμβαίνει σε όλα αυτά τα φυσικά κατευθυνόμενα δίκτυα.

Στο επίπεδο δεδομένων (data plane) οι λειτουργίες του σχετίζονται με τη χρήση του πρωτοκόλλου SCADA (Supervisory Control and Data Acquisition Protocol – Πρωτόκολλο Εποπτικού Ελέγχου και Απόκτησης Δεδομένων) όπου κατευθύνουν την προώθηση πακέτων για τη ροή πληροφοριών, επιπρόσθετα, το επίπεδο ελέγχου (control plane) είναι όπου θα χρησιμοποιηθούν δεδομένα, παραδείγματος χάρη, για να ειδοποιήσει έναν ανθρώπινο χειριστή αν οι αισθητήρες και τα προγράμματα στην αρχιτεκτονική κατανεμημένης γραμμής βλέπουν υψηλή συχνότητα ελαττωμάτων.

Με άλλα λόγια αυτό είναι το SDN, όπου με την απομόνωση του επιπέδου ελέγχου αλλάζει στην κυριολεξία ο τρόπος με τον οποίο βλέπουμε, λειτουργούμε και χρησιμοποιούμε πόρους δικτύου. Όλα αυτά όμως δεν λένε την ιστορία για το πώς έχει ρυθμιστεί κάθε

μεμονωμένη παρουσία (individual instance) του SDN αλλά το πώς ο διαχωρισμός του επιπέδου ελέγχου από το επίπεδο δεδομένων έχει μεγάλες δυνατότητες στην αλλαγή της υποδομής δικτύου.

2.4 Σχετικές Εργασίες (Related Works)

Εν συνέχεια, θα δούμε για τα SDN όπου το επίπεδο ελέγχου (control plane) μας δίνει μια οπτική αναπαράσταση *της υποδομής του δικτύου* ενώ το επίπεδο δεδομένων (data plane) δίνει μια οπτική αναπαράσταση *της κίνησης των χρηστών.* Επιπλέον, θα δούμε τα δυο αυτά επίπεδα (επίπεδο ελέγχου/δεδομένων) με τη χρήση του πρωτοκόλλου Εποπτικού Ελέγχου και Απόκτησης Δεδομένων (SCADA - Supervisory Control and Data Acquisition Protocol).

2.4.1 Αυτόνομα Συστήματα (ΑΣ)

Τα αυτόνομα συστήματα (ΑΣ) χρειάζονται ένα σύνολο εφαρμογών για να μπορούν να διαχειρίζονται τα δίκτυά τους όπου η πολυπλοκότητα αυτού του έργου αυξάνεται, ιδίως με παραδοσιακά στοιχεία δικτύων λόγω του γεγονότος ότι τα στοιχεία αυτά βρίσκονται σε ειδικά ολοκληρωμένα κυκλώματα (ASICs – Application Specific Integrated Circuit) [24]. Το SDN αναμένεται να επιλύσει αυτά τα ζητήματα, αλλά οι δυνατότητες του SDN παρακίνησαν τις εργασίες που παρουσιάζονται στο άρθρο [25], δηλαδή να προτείνουν μια νέα αρχιτεκτονική βασισμένη στο SDN η οποία θα μπορούσε να παρέχει ευέλικτη διαχείριση πόρων και να ενισχύει την αξιοποίηση των πόρων σε αυτόνομα δίκτυα οχημάτων (AV).

Ο R. Masoudi στο άρθρο του [26] μαζί με τον A. Ghaffari εξέτασε σχολαστικά τα επίπεδα δεδομένων, ελέγχου και εφαρμογής του SDN με διάφορα εργαλεία προσομοίωσης, αποσφαλματώσεις (debuggers) και με δοκιμές για ανάπτυξη.

Στο επόμενο άρθρο [27], διερεύνησαν και περιέγραψαν διαφορετικές προσπάθειες σε αρχιτεκτονικές, στοιχεία και εφαρμογές SDN, ενώ συνέκριναν διαφορετικές εφαρμογές σε παραδοσιακά δίκτυα και σε περιβάλλοντα SDN. Η τρέχουσα κατάσταση καθώς κι τα οφέλη του SDN αναλύονται επίσης στο άρθρο [28] του IEEE.org.

Έχουν επίσης διεξαχθεί μελέτες στα ακόλουθα άρθρα [29] των Karakus και Durresi, αλλά και στο άρθρο [30] των Hu, Guo, Yi, Lan και Baker, σχετικά με τη δυνατότητα κλιμάκωσης των ζητημάτων επιπέδου ελέγχου, συνέπειας και αξιοπιστίας των SDN αλλά ταυτόχρονα

κατηγοριοποιούνται τα θέματα αυτά σε διάφορες προσεγγίσεις, όπως υβριδικά, ιεραρχικά και επίπεδα σχέδια SDN. Επιπρόσθετα, ο ερευνητικός τομέας για τα SDN εξετάζει τεχνολογίες όπως IoT, WSN (wireless sensor network) και εικονική διαμόρφωση (virtualization).

Στα ακόλουθα δυο άρθρα [31], [32] οι συγγραφείς διερεύνησαν την έκχυση/ένεση του SDN σε WSN, αυτές οι δύο τεχνολογίες μαζί μπορούν να διαδραματίσουν ζωτικό ρόλο στην άνοδο της IoT. Η υλοποίηση IoT στην τρέχουσα κατάσταση περιγράφεται στο άρθρο του Vasilakou, Misra και Bera [33] όπου απεικονίζει τις διακριτές κατηγορίες τεχνολογιών SDN από τις πτυχές του IoT. Στο άρθρο των Li και Chen [34] παρουσιάζεται μια εκτενή έρευνα για τα hypervisor, στην οποία ταξινομούνται διαφορετικά hypervisor με βάση τις αρχιτεκτονικές τους αλλά και η σχέση μεταξύ SDN και NFV.

2.4.1.1 Open SDN and OpenFlow

Αρχικά υπήρξε ένα κεντρικό πρότυπο με την ονομασία OpenFlow, όπως διατυπώνετε στο άρθρο [35], όπου αυτό αρχικά καθόριζε το πρωτόκολλο επικοινωνίας σε αρχιτεκτονικές SDN οι οποία επέτρεπε στον ελεγκτή SDN να αλληλοεπιδρά άμεσα με το επίπεδο προώθησης συσκευών δικτύου, όπως διακόπτες (switches) και δρομολογητές (routers), τόσο φυσικοί αλλά κι εικονικοί (πάντα βασισμένοι σε υπεύθυνους εποπτείας – hypervisor based) έτσι ώστε να μπορεί να προσαρμόζει καλύτερα τις μεταβαλλόμενες επιχειρηματικές απαιτήσεις. [36]

Άρα ένα πρωτόκολλο επικοινωνίας ανοιχτού κώδικα που διαχωρίζει τις εργασίες των δυο επιπέδων δηλαδή ελέγχου και δεδομένων μπορεί να χαρακτηρίσει ως το πρωτόκολλο OpenFlow. [37] Οι όροι “SDN” και “OpenFlow” χρησιμοποιούνται συχνά εναλλάξ μέσα στην επιστημονική κοινότητα.

Το σημαντικότερο πράγμα που πρέπει να γνωρίζουμε για το OpenFlow είναι ότι δεν είναι ο μοναδικός πρακτικά τρόπος για να γίνει μια δικτύωση που καθορίζεται από λογισμικό (SDN) αλλά πολλοί ειδικοί του κλάδου επισημαίνουν ότι υπάρχουν πολλά πρωτόκολλα από/για συγκεκριμένους προμηθευτές (vendor specific protocols). [35] [38] [39]

Μεγάλες εταιρείες τεχνολογίας ήθελαν να είναι σε θέση να προσφέρουν οι ίδιοι δίκτυα που καθορίζονται από λογισμικό (SDN) και για να το προσφέρουν αυτό δημιούργησαν τους δικούς τους τρόπους ούτως ώστε πολλές εταιρείες να κινηθούν προς τη λύση

ανοιχτού κώδικα (open source solutions) και μακριά από τα παραδοσιακά προϊόντα τους.

Τα οφέλη του OpenFlow μαζί με την μεγαλύτερης δυνατότητας γρήγορης εισαγωγής νέων δυνατοτήτων (new features) και υπηρεσιών δικτύου (network services), απλής παροχής (simple provisioning – configuration, deployment and management multiple IT resources) και βελτιστοποιημένης απόδοσης (optimized performance).

Διάφορα παραδείγματα από μεγάλες τεχνολογικές εταιρείες που εφαρμόζουν την προσέγγιση της φιλοσοφίας του OpenFlow, είναι η Cisco με το "Cisco OpenFlow" όπου χρησιμοποιούν διακόπτες με δυνατότητα OpenFlow αλλά και κάτι παρόμοιο είναι αυτό τις IBM με το "IBM vSwitch". [40] [41]

2.4.1.2 Ασφάλεια του SDN

Πολλοί οργανισμοί ενδιαφέρονται να αποκομίσουν τα οφέλη των SDN δικτύων υλοποιώντας την αρχιτεκτονική ως επικάλυψη στα δίκτυά τους όπου συνήθως ο administrator ενός SDN [42] αποτελεί εύκολο στόχο για επιθέσεις Denial of Service (DoS) και Man in the Middle Attack (MiMT) [43], [44], άρα συνεπώς θα πρέπει να δίνεται έμφαση στην καινοτομία για την εφαρμογή νέων υπηρεσιών και την αύξηση της αποτελεσματικότητας, και ο τομέας της ασφάλειας να αναπτύσσεται ως ξεχωριστός τομέας ανάπτυξης όπου ο στόχος του θα είναι να ξεπεράσει όλες τις αδυναμίες των παραδοσιακών δικτύων και να βελτιώσει την αξιοπιστία και την ακεραιότητά τους.

Τα δίκτυα SDN διερευνώνται ως μέσο για την εξασφάλιση της τελικής επικοινωνίας σε διάφορους τομείς των αυτόνομων συστημάτων (ΑΣ) [45]. Άρα ουσιαστικά τα SDN μαζί με ένα σύνολο διάφορων εφαρμογών ασφάλειας (Intrusion Detection Software, Intrusion Prevention Software κλπ.) που στηρίζονται σε SDN απαρτίζουν το οπλοστάσιο που θα κερδίσει τον αγώνα κατά του εγκλήματος στον κυβερνοχώρο.

Επιπλέον, τα δίκτυα SDN μπορούν να διευκολύνουν τη συλλογή πληροφοριών χρήσης δικτύου (ποιος χρησιμοποιά και τι) με την χρήση εργαλείων παρακολούθησης δικτύου (network monitoring tools), οι οποίες θα μπορούσαν να υποστηρίξουν βελτιωμένο σχεδιασμό αλγορίθμων που χρησιμοποιείται για τον εντοπισμό επιθέσεων.

Η νέα αυτή γενιά εφαρμογών θα εκμεταλλευτεί πιο ενημερωμένους πράκτορες SDN και έτσι θα μπορέσει να βελτιώσει την επιβολή πολιτικής και τον εντοπισμό και τον περιορισμό της ανωμαλίας της κυκλοφορίας. Αυτές οι εφαρμογές ενδέχεται να είναι σε θέση να αποκλείσουν κακόβουλους εισβολείς πριν εισέλθουν στις κρίσιμες περιοχές του δικτύου.

2.5 Software-Defined Networking as a Service (NaaS)

Όπως προ-αναφέραμε προηγουμένως αρκετές φορές, η δικτύωση που καθορίζεται από λογισμικό (SDN) [3] υπάρχει στο πλαίσιο νέων παραδειγμάτων λογισμικού ως υπηρεσία (software-as-a-service) [46] και cloud computing [47] όπου αυτό οδηγεί σε συζήτηση μέσα στην επιστημονική κοινότητα για το μέλλον της δικτύωσης που καθορίζεται από λογισμικό (SDN) και την εμφάνιση νέων εφαρμογών δικτύου ως υπηρεσίας (NaaS). [48]

Όπως αναφέρει ο J. Barabas στο άρθρο του [49] για την υποδομή ως υπηρεσία (IaaS), την πλατφόρμα ως υπηρεσία (PaaS) και το λογισμικό ως υπηρεσία (SaaS) “Στην παραδοσιακή μέθοδο κατανάλωσης υπηρεσιών ή πόρων, ο ιδιοκτήτης της υποδομής είναι υπεύθυνος για τη διαχείριση κάθε υλικού και λογισμικού που χρησιμοποιεί. Κανονικά, χρειάζεται κάποιος χρόνος για να αποκτήσει πρόσβαση ένας χρήστης σε έναν νέο πόρο, αλλά μπορεί να ρυθμιστεί ακριβώς όπως απαιτείται.”

Με αυτό υπόψη μας, στο άρθρο τους οι P. Costa, M. Migliavacca, P. Pietzuch και A.L. Wolf [50] αναφέρουν συγκεκριμένα ότι “Το cloud computing πραγματοποιεί το όραμα του βοηθητικού υπολογιστή (utility computing). Οι ενοικιαστές μπορούν να επωφεληθούν από την παροχή κατ 'απαίτηση υπολογιστικών πόρων σύμφωνα με ένα μοντέλο πληρωμής ανά χρήση (pay per use model) και μπορούν να αναθέσουν σε εξωτερικούς συνεργάτες αγορές και συντήρηση υλικού. Ωστόσο, οι ενοικιαστές έχουν περιορισμένη ορατότητα και έλεγχο των πόρων του δικτύου. Ακόμη και για απλές εργασίες, οι ενοικιαστές πρέπει να καταφεύγουν σε αναποτελεσματικά δίκτυα επικάλυψης. Για την αντιμετώπιση αυτών των ελλείψεων, προτείνουμε το Network-as-a-Service (NaaS), ένα πλαίσιο που ενσωματώνει τις τρέχουσες προσφορές cloud computing με άμεση, αλλά ασφαλή, πρόσβαση ενοικιαστών στην υποδομή δικτύου.

Χρησιμοποιώντας το NaaS, οι ενοικιαστές μπορούν εύκολα να αναπτύξουν προσαρμοσμένα πρωτόκολλα δρομολόγησης και πολλαπλής διανομής. Περαιτέρω,

τροποποιώντας το περιεχόμενο των πακέτων onpath, μπορούν να εφαρμόσουν αποτελεσματικά προηγμένες υπηρεσίες δικτύου, όπως συγκέντρωση δεδομένων εντός δικτύου (in-network data aggregation), εξάλειψη πλεονασμού (redundancy elimination) και έξυπνη προσωρινή αποθήκευση (smart caching).

Συζητάμε εφαρμογές που μπορούν να επωφεληθούν από το NaaS, υποκινούν τη λειτουργικότητα που απαιτείται από το NaaS και σχεδιάζουμε ένα πιθανό μοντέλο υλοποίησης και προγραμματισμού που μπορεί να υποστηριχθεί από την τρέχουσα τεχνολογία. Η αρχική μας μελέτη προσομοίωσης δείχνει ότι, ακόμη και με περιορισμένη ικανότητα επεξεργασίας σε διακόπτες δικτύου, το NaaS μπορεί να αυξήσει σημαντικά την απόδοση της εφαρμογής και να μειώσει την κίνηση του δικτύου."

Άρα ουσιαστικά καθώς βελτιώνουμε την πραγματοποίηση υλοποιήσεων SDN (SDN implementations), θα επιτρέπεται όλο και περισσότερο στους προμηθευτές να προσφέρουν αποτελεσματικότερα το NaaS μέσω του cloud.

2.6 Προμηθευτές Δικτύων κι SDN Ανοιχτού Κώδικα

Καθώς διάφοροι προμηθευτές δικτυών και διάφορες άλλες επιχειρήσεις (Cisco, IBM, AT&T, Vodafone κλπ.) κινούνται προς ιδιόκτητες λύσεις NaaS [48] και άλλες εφαρμογές SDN ως εμπορικές υπηρεσίες, η κοινότητα ανοιχτού κώδικα (open source community) μέσα από το Linux Foundation εξετάζει τη δικτύωση που καθορίζεται από λογισμικό (SDN) μέσω ενός νέου ερευνητικού έργου που ονομάζεται OpenDaylight [51] όπου απαρτίζεται από γνωστούς προμηθευτές δικτύων στο πλαίσιο της κοινοπραξίας του "The Linux Foundation" και έχει σαν κύριο στόχο του την ανάπτυξη ανοικτών και τυποποιημένων ελεγκτών (standardized controllers) για δικτύωση που καθορίζεται από λογισμικό (SDN).

Έτσι το 2013 το πρωτοπόρο ερευνητικό έργο OpenDaylight φιλοξενείται από το Linux Foundation υποστηρίζοντας τη συνεργατική εργασία ανοιχτού κώδικα (open-source collaboration) σχετικά με την πρόοδο εικονικοποίησης δικτύων (NFV - Network Functions Virtualization) και λειτουργιών δικτύου που καθορίζεται από λογισμικό (SDN).

Υποστηρίζοντας το OpenFlow, το OpenDayLight κληρονομεί και διάφορες πτυχές από τις βάσεις κώδικα αυτών των μεγάλων εταιρειών τεχνολογίας όπως για παράδειγμα της

Cisco που παραδοσιακά προσέφεραν διακόπτες δικτύου (network switches) και άλλα εργαλεία δικτύου.

Ο Tom Nolle στο άρθρο [52], αναφέρει ότι: “Στον κόσμο του SDN, έχουμε δει μια πόλωση (polarization) λύσεων με βάση ποια από αυτά τα σημεία είναι πιο σημαντικά. Μπορείτε να δημιουργήσετε το "SDN" εάν μπορείτε να δημιουργήσετε υπηρεσίες - "Networking as-a-Service" ή NaaS - ανεξάρτητα από το πώς το κάνετε; Μπορείτε να κάνετε SDN χωρίς κεντρικό έλεγχο, χωρίς OpenFlow; Όλες οι έγκυρες ερωτήσεις, καθώς οι αγοραστές της δικτύωσης επόμενης γενιάς θα αγοράσουν / κάνουν ό, τι είναι χρήσιμο σε αυτούς, κάτι που μπορεί να είναι οποιοδήποτε ή όλα αυτά τα πράγματα.”

Οπότε το σημαντικότερο είναι ότι το OpenDayLight απεικονίζετε σαν το νέο πρότυπο μοντέλο ελεγκτή (controller model) που δημιουργεί καινούργια πρωτόκολλα δικτύου και διαφοροποιεί τους τύπους των υποστηριζόμενων συσκευών παλαιού τύπου.

Έτσι έχουμε μια πρόοδο όπου τα SDN ξεκινούν να “αναλαμβάνουν” το κόσμο της δικτυώσεις αφήνοντας στο παρελθόν τα παραδοσιακά, στατικά δίκτυα σε μεμονωμένες αίθουσες υλικού εξοπλισμού.

2.6.1 Εξομοιωτής Δικτύου Ανοιχτού Κώδικα: Mininet

Όπως αναφέρθηκε σε προηγουμένη υπό-ενότητα το Mininet [53] είναι ένας εξομοιωτής δικτύου ανοιχτού κώδικα (open source network emulator) που δημιουργεί ένα δίκτυο εικονικών κεντρικών υπολογιστών (virtual hosts), διακοπτών (switches), ελεγκτών (controllers) και συνδέσμων (links).

Όπως αναφέρουν στο άρθρο [54] οι Kaur, Singh και Ghumman, το Mininet είναι ένας εξομοιωτής δικτύου που είναι υπεύθυνο για την ανάπτυξη μεγάλων δικτύων στους περιορισμένους πόρους πλέον ενός απλού υπολογιστή ή/και μιας εικονικής μηχανής.

Το Mininet ουσιαστικά δημιουργήθηκε για να επιτρέψει την έρευνα στο SDN και στο OpenFlow όπου ο εξομοιωτής επιτρέπει την αλληλεπίδραση του μη τροποποιημένου κώδικα σε εικονικό υλικό σε έναν απλό υπολογιστή αλλά ταυτόχρονος παρέχει ευκολία και ρεαλισμό με πολύ χαμηλό κόστος.

Απλουστέρα το SDN διαχωρίζει το επίπεδο ελέγχου (control plane) από το επίπεδο δεδομένων (data plane), καθιστώντας τις συσκευές δικτύου όπως διακόπτες (switches) και δρομολογητές (routers) πλήρως προγραμματιζόμενες και έτσι, το δίκτυο συμπεριφέρεται σύμφωνα με τις απαιτήσεις των χρηστών.

Σαν προεπιλογή, το Mininet παρέχει διακόπτες (switches) Open vSwitch (OVS) και ελεγκτές (controllers) OVS, όπου το Open vSwitch [55] ή OVS, είναι μια εφαρμογή ανοιχτού κώδικα ενός διανεμημένου εικονικού διακόπτη πολλαπλών επιπέδων (virtual multiplayer switch).

Ωστόσο το Mininet, έχει την υποστήριξη να εγκαταστήσει άλλους/προτιμώμενους ελεγκτές SDN (controllers) και διακόπτες (switches) αντί αυτών που έχει σαν προεπιλογές. Το κύριο χαρακτηριστικό που διακρίνει τις συσκευές SDN από τις παραδοσιακές συσκευές δικτύου είναι το πεδίο για την προσαρμογή πρωτοκόλλων και λειτουργιών.

2.7 Οπτικός Πίνακας Ελέγχου (Visual Dashboard)

Ζούμε στην εποχή του τεχνολογικού αιώνα, και πιο συγκεκριμένα στην εποχή του οπτικού πίνακα ελέγχου (visual dashboard) [1] - όπου οι πωλητές και οι παραγωγοί λογισμικού προσπαθούν να κάνουν τα πάντα όσο πιο οπτικά γίνεται, για να πετύχουν τους στόχους να προσφέρουν μια καλύτερη “αισθητική διεπαφή”.

Οι χρήστες του οπτικού πίνακα ελέγχου θέλουν να βλέπουν εικόνες για το τι συμβαίνει σε ένα πολύ περίπλοκο ψηφιακό περιβάλλον, και όχι να διαβάζουν για το πώς λειτουργούν οι ελεγκτές (controllers) ή για το πού πηγαίνουν τα δεδομένα (data), θέλουν με απλά λόγια αυτές τις πληροφορίες στα χέρια τους με τη μορφή οπτικών εικόνων. [56]

Έτσι οι χρήστες λαμβάνουν συχνά μια καλή εικόνα των εννοιών δικτύωσης που καθορίζονται από το λογισμικό μέσω ενός οπτικού ταμπλό που έχουν στα χέρια τους, π.χ. πού βρίσκεται ο "εγκέφαλος" του επιπέδου ελέγχου (control plane) και πώς σχετίζεται με τους κατανεμημένους κόμβους ελέγχου (control nodes) ή διακόπτες (switches) που έχουν δεδομένα (data) ενσωματωμένα σε αυτές.

Επιπρόσθετα παραδείγματα που μπορούμε να δούμε είναι όταν ένας προγραμματιστής μπορεί να σχεδιάσει έναν πίνακα εργαλείων (control panel) με έναν ελεγκτή SDN όπου θα κατευθύνει σε διάφορους πράκτορες SDN που είναι ενσωματωμένοι σε διακόπτες Ethernet ή κόμβους δικτύου τις διάφορες αναγκαίες πληροφορίες.

Ο οπτικός πίνακας ελέγχου σε αυτή την περίπτωση μπορεί να δείχνει τη θέση των κεντρικών υπολογιστών ή servers σε αυτήν την αρχιτεκτονική και με ποιον τον τρόπο οι πληροφορίες αναπηδούν μέσω του δικτύου. Επιπλέον με τα πιο πάνω, μπορεί να περιλαμβάνει όλα τα είδη μετρήσεων και δεδομένων σχετικά με τη ροή πακέτων και πληροφορίων που κινούνται μέσω του συστήματος, αλλά και, να εμφανίζει όλα τα είδη πρωτοκόλλων που σχετίζονται με τη διαχείριση πληροφοριών ή τη ροή πακέτων.
Οπότε εξυπακούεται ότι ο οπτικός πίνακας ελέγχου (visual dashboard) [1] που καθορίζετε από το λογισμικό (SDN) θα μπορούσε πραγματικά να είναι το εγχειρίδιο διαχειριστή δικτύου (administrator's handbook) για τον λόγο του ότι επειδή είναι οπτικό και δεν περιορίζεται στην αρχαϊκή μορφή της εξήγησης κάτι μέσω παραγράφων κειμένου.

Οι χρήστες μπορούν να μάθουν με μια ματιά πώς διαμορφώνεται η αρχιτεκτονική, πώς να την αλλάξουν, εάν υπάρχουν ανεπάρκειες ή προβλήματα, και τι ακριβώς συμβαίνει μέσα στο σύστημα σε μια δεδομένη στιγμή. Σκεφτείτε να επιδιώξετε μια επιθυμητή κατάσταση, να βελτιστοποιήσετε τις ροές εργασίας και να εξαλείψετε τα σημεία συμφόρησης.

Όλα αυτά βοηθιούνται από τις διεπαφές (interfaces) που βοηθούν τους χρήστες να καταλάβουν τι κάνει το SDN, και επίσης, κατά πόσο μια υπηρεσία βρίσκεται στο cloud, είτε πρόκειται για καλώδιο οπτικών ινών, η ακόμη και άν επρόκειτο για εταιρεία που χρησιμοποιεί κοντέινερ ή συστήματα προμηθευτών όπως το AWS (AMAZON) ή το Azure (Microsoft).

Η ιδέα του προσανατολισμού των χρηστών SDN με έναν οπτικό πίνακα ελέγχου (visual dashboard) επιστρέφει πίσω στην ιδέα ότι τα συστήματα SDN είναι διαφορετικά και δεν υπάρχει μόνο ένα μοντέλο για όλα τα μεγέθη.

Συνεπώς, είναι λογικό ότι αυτά τα γραφικά κατασκευασμένα από προμηθευτές ή οποιονδήποτε εργάζεται στον τομέα του SDN, να μπορούν να προσανατολίσουν τον καθένα όχι μόνο στον τρόπο λειτουργίας του συστήματος αλλά και πώς να τα ελέγξει, να τα παρακολουθήσει ή να τα διαχειριστεί με άλλο τρόπο αυτά τα εξαιρετικά εμπλεκόμενα δίκτυα επόμενης γενιάς.

Κεφάλαιο 3
Μεθοδολογία

Στο κεφάλαιο που ακολουθεί θα ασχοληθούμε με την μεθοδολογία, τον σκοπό και το είδος της έρευνας, θα δούμε επιπλέον τα βασικά ερευνητικά ερωτήματα αλλά και με την μεθοδολογία υλοποίησης που θα ακολουθηθεί στην μεταπτυχιακή διατριβή.

3.1 Σκοπός και Είδος Έρευνας

Κυρίως σκοπός της μεταπτυχιακής διατριβής είναι να υλοποιηθεί ένα SDN [3] (software-defined network) για να μελετήσουμε κατά πόσο μπορεί να προσφέρει στην ασφάλεια των δικτύων την ζητούμενη ασφάλεια ή οποιεσδήποτε άλλες βελτιώσεις, εφόσον πρώτα μέσω της προτεινομένης μεθοδολογίας μελετήσουμε τις υπάρχουσες λύσεις και όπως είδαμε στο προηγούμενο κεφάλαιο μέσω βιβλιογραφικής ανασκόπησης της εφαρμογές SDN που σχεδιάστηκαν για παροχή ενισχυμένης ασφάλειας, για να καταλήξουμε να υλοποιήσουμε ένα SDN όπου θα εφαρμοστούν οι προτεινόμενες λύσεις SDN.

Επιπρόσθετα, θα δούμε την ποσοτική έρευνα όπου η ερευνητική στρατηγική εστιάζει στην ποσοτικοποίηση της συλλογής και ανάλυσης δεδομένων με έμφαση στη δοκιμή της θεωρίας. [57]

3.2 Βασικά Ερευνητικά Ερωτήματα

Στην παρούσα μεταπτυχιακή διατριβή θα αναλυθεί πώς το SDN δίκτυο, μέσω του Mininet, μπορεί να υλοποιηθεί και να χρησιμοποιηθεί έτσι ώστε να ελαχιστοποιηθούν οι απάτες, ανακατευθύνοντας τα πακέτα και απομονώνοντας κακόβουλες συσκευές και πακέτα με την προσομοίωση της κυκλοφορίας του δικτύου, χρησιμοποιώντας την τεχνολογία OpenFlow. Επιπλέον μέσω του προσομοιωτή δικτύου θα χρησιμοποιηθούν οι δυνατότητες του SDN για το φιλτράρισμα της κίνησης συμμετεχόντων που επιχειρούν να αλληλοεπιδράσουν με άλλους κόμβους.

Κάποια βασικά αρχικά ερευνητικά ερωτήματα που μπορούν να προκύψουν για ανάλυση είναι τα πιο κάτω.

3.2.1 Τι είναι το SDN και πως μπορεί να υλοποιηθεί;

Το SDN (Software Defined Network) [3] ουσιαστικά κάνει ένα δίκτυο πιο ευέλικτο και πιο εύκολο στη διαχείριση του κι αυτό άλλωστε είναι ο σχεδιασμός της αρχιτεκτονικής αυτής. Για την υλοποίηση του SDN μπορεί να χρησιμοποιηθεί ένας από τους ακόλουθους τρεις προσομοιωτές δικτύου το GNS3, Mininet ή NetSim όπου θα προτιμήσω το Mininet [53] που είναι ένας εξομοιωτής δικτύου ανοιχτού κώδικα (open source network emulator) που δημιουργεί ένα δίκτυο εικονικών κεντρικών υπολογιστών (virtual hosts), διακοπτών (switches), ελεγκτών (controllers) και συνδέσμων (links) όπου θα γίνει με το προ-δημιουργημένο εικονικό μηχάνημα (SDN_VM_64bit.ova) το οποίο βασίζεται στο περιβάλλον του λειτουργικού συστήματος των Ubuntu 14.04 64-bit.

3.2.2 Πώς διαβάζονται και διανέμονται τα πακέτα μέσα σε ένα SDN;

Εδώ θα χρησιμοποιηθεί το Mininet που αναφέραμε πιο πάνω, όπου είναι υπεύθυνο για την δρομολόγηση πακέτων (route packets) και πλαισίων μεταγωγών (switch frames) γιατί προσομοιώνει την κυκλοφορία δικτύου χρησιμοποιώντας την τεχνολογία OpenFlow.

3.2.3 Πώς απομονώνονται ανεπιθύμητα πακέτα/συσκευές μέσα σε ένα SDN;

Αφότου χρησιμοποιηθεί το Mininet που αναφέραμε πιο πάνω, τότε είναι υπεύθυνο για την απομονώσει των ανεπιθύμητων πακέτων/συσκευών μέσα στο SDN.

3.2.4 Μπορεί το SDN να προσφέρει ένα επιπλέον επίπεδο ασφάλειας;

Αφότου χρησιμοποιηθεί το Mininet, θα κάνουμε διάφορες δοκιμές (running tests) όπου θα μπορέσουμε να δούμε τα διάφορα επίπεδα ασφάλειας του SDN ουσιαστικά το Mininet θα χρησιμοποιηθεί σαν firewall για σκοπούς ασφάλειας.

3.3 Μεθοδολογία Υλοποίησης

Κυρίως σκοπός της μεταπτυχιακής διατριβής όπως προαναφέραμε είναι να υλοποιηθεί ένα SDN [3] (software-defined network) περιβάλλον για να μελετήσουμε κατά πόσο μπορεί να προσφέρει στην ασφάλεια των δικτύων την ζητούμενη ασφάλεια ή οποιαδήποτε άλλη βελτίωση, εφόσον πρώτα μέσω της προτεινομένης μεθοδολογίας μελετήσουμε τις υπάρχουσες λύσεις και όπως είδαμε στο προηγούμενο κεφάλαιο μέσω βιβλιογραφικής ανασκόπησης της εφαρμογές SDN που σχεδιάστηκαν για παροχή ενισχυμένης ασφάλειας, για να καταλήξουμε να υλοποιήσουμε ένα SDN περιβάλλον όπου θα εφαρμοστούν οι προτεινόμενες λύσεις SDN.

3.3.1 Κύκλοι Ζωής Ανάπτυξης Λογισμικού (Software Development Life Cycles)

Ανάμεσα σε διάφορους ερευνητές τις τεχνολογικής κοινότητας, αλλά, γενικότερα και μέσα σε διάφορες τεχνολογικές εταιρείες παρατηρείτε ότι χρησιμοποιούνται διάφοροι κύκλοι ζωής ανάπτυξης λογισμικού (Software Development Life Cycles) όπου η Maryna Demchenko στο άρθρο της [58], αναφέρει ότι: "Ο κύκλος ζωής ανάπτυξης λογισμικού (SDLC) είναι μια σειρά βημάτων που πρέπει να ακολουθήσει μια ομάδα ανάπτυξης για την ανάπτυξη και συντήρηση λογισμικού."

Άρα ουσιαστικά με άλλα λόγια, ένας κύκλος ζωής ανάπτυξης λογισμικού είναι ένα λεπτομερές περιγραφικό σχέδιο για τον τρόπο ανάπτυξης, συντήρησης, αντικατάστασης και αλλαγής ή βελτίωσης συγκεκριμένου λογισμικού.

Αυτό σημαίνει ότι περνάει από τα ακόλουθα έξι (6) διαφορετικά στάδια, όπου το πρώτο στάδιο είναι ο σχεδιασμός και η ανάλυση των απαιτήσεων, το δεύτερο, ορίζοντας τις απαιτήσεις για το προϊόν, το τρίτο, ο σχεδιασμός της αρχιτεκτονικής του προϊόντος έτσι ώστε να προχωρήσει στο επόμενο στάδιο που είναι η κατασκευή ή ανάπτυξη του προϊόντος καθώς και, η δοκιμασία του και μετά στο τελικό στάδιο που είναι ανάπτυξη στην αγορά και συντήρηση του.

3.3.1.1 Μοντέλο Καταρράκτη (Waterfall Model)

Ο καλύτερος τρόπος κατανόησης αυτού του μοντέλου [59] μιας και είναι το πρώτο μοντέλο της οικογένειας από τα μοντέλα που ασχολούνται με τους κύκλους ζωής

ανάπτυξης λογισμικού (SDLC) [58] είναι ότι χρησιμοποιείται ευρέως στη μηχανική λογισμικού (software engineering) για να εξασφαλίσει την επιτυχία μιας έρευνας.

Η όλη διαδικασία ανάπτυξης λογισμικού χωρίζεται σε ξεχωριστές φάσεις και αυτή είναι η προσέγγιση του μοντέλου, όπου συνήθως το αποτέλεσμα μιας φάσης λειτουργεί ως η είσοδος για την επόμενη φάση διαδοχικά.

3.3.1.2 Επαναληπτικό Μοντέλο (Iterative Model)

Σε αυτό το μοντέλο [60] γίνεται μια επαναληπτική διαδικασία με την απλή εφαρμογή ενός υποσυνόλου απαιτήσεων για το λογισμικού όπου βελτιώνεται επαναληπτικά στις εξελισσόμενες εκδόσεις μέχρι να εφαρμοστεί το πλήρες σύστημα.

Γίνονται τροποποιήσεις στον σχεδιασμό και προστίθενται νέες λειτουργικές δυνατότητες, με κάθε επανάληψη, οπότε η βασική ιδέα πίσω από αυτό το μοντέλο είναι η ανάπτυξη ενός συστήματος μέσω επαναλαμβανόμενων κύκλων όπου κάθε φορά σταδιακά γίνεται σε μικρότερα τμήματα.

3.3.1.3 Σπειροειδές Μοντέλο (Spiral Model)

Σε αυτό το μοντέλο [61] γίνεται μια επανειλημμένη διαδικασία που περνάει από τέσσερις φάσεις (αναγνώριση, σχεδιασμός, κατασκευή, αξιολόγηση και ανάλυση κινδύνου) σε σπειροειδείς επαναλήψεις (spirals) μέχρι να ολοκληρωθεί η έρευνα του λογισμικού.

3.3.1.4 Μοντέλο Επαλήθευσης/Επικύρωσης (V-Model)

Επιπρόσθετα σε αυτήν την υπό-ενότητα, θα δούμε το μοντέλο επαλήθευσης/επικύρωσης όπου σε αυτό το μοντέλο (V-Model – Verification and Validation Model) [62] η δυο φάσεις δοκιμής και ανάπτυξης γίνονται και σχεδιάζεται αντίστοιχα, οπότε, υπάρχουν φάσεις επαλήθευσης στη μία πλευρά του «V» και φάσεις επικύρωσης από την άλλη όπου ενδιάμεσα τους η φάση της κωδικοποίησης (coding) ενώνει τις δύο πλευρές του μοντέλου.

3.3.2 Σύγκριση Μοντέλων Ανάπτυξης Λογισμικού

Υπάρχουν αρκετά διαφορετικά μοντέλα για το κύκλο ζωής ανάπτυξης ενός λογισμικού (Software Development Life Cycles) και σε αυτήν την υπό-ενότητα θα επικυρωθούμε στα τέσσερα βασικά αυτά μοντέλα που προανάφερα προηγούμενος και θα τα συγκρίνουμε

με βάση τα πλεονεκτήματα και μειονεκτήματα τους ούτως ώστε να μπορέσουμε να επιλέξουμε το κατάλληλο μοντέλο σύμφωνα με τις δικές μας απαιτήσεις. [63]

Αρχικά είδαμε το Μοντέλο Καταρράκτη (Waterfall Model) όπου παρόλο που είναι το πρώτο μοντέλο που χρησιμοποιήθηκε, με τα πλεονεκτήματα του να είναι απλό και εύκολο στη χρήση όπως επίσης εύκολο στη διαχείριση λόγω της ακαμψίας του μοντέλου όπου η κάθε φάση έχει συγκεκριμένα παραδοτέα και μια διαδικασία αναθεώρησης, αλλά επίσης, οι φάσεις υποβάλλονται σε επεξεργασία και ολοκληρώνονται μία κάθε φορά, και ένα σαν τελευταίο πλεονέκτημα είναι ότι λειτουργεί καλά για μικρότερες απαιτήσεις που είναι πολύ καλά κατανοητές.

Μειονεκτήματα που θα μπορούσαμε να πούμε για το πιο πάνω μοντέλο είναι ότι είναι φτωχό μοντέλο για ένα ερευνητικό έργο που είναι πολύ-σύνθετο και αντικειμενοστραφές (complex and object oriented projects), αλλά και επίσης οι απαιτήσεις διατρέχουν μέτριο έως υψηλό κίνδυνο αλλαγής. Επίσης σαν μοντέλο, δεν παράγει κανένα λογισμικό εργασίας μέχρι αργά κατά τη διάρκεια του κύκλου ζωής.

Επιπρόσθετα, είδαμε το Επαναληπτικό Μοντέλο (Iterative Model) που είναι πιο ευέλικτο από το βασικό μοντέλο του καταρράκτη με επιπλέον πλεονεκτήματα του να είναι ότι αν υπάρξει συνέχεια του προσωπικού μεταξύ των φάσεων, η τεκμηρίωση μπορεί να μειωθεί σημαντικά αλλά και η εφαρμογή των εύκολων απαιτήσεων του λογισμικού δεν χρειάζεται να περιμένει τις δύσκολες όπου σαν μοντέλο λειτουργεί καλά για έργα μικρό-μεσαίου μεγέθους.

Τα ορόσημα σημεία σε αυτό το μοντέλο είναι πιο διφορούμενα από το προηγούμενο μοντέλο του καταρράκτη, αλλά και οι δραστηριότητες που εκτελούνται παράλληλα υπόκεινται σε λανθασμένη επικοινωνία και σε λανθασμένες υποθέσεις. Επίσης, οι απροσδόκητες αλληλεξαρτήσεις (interdependencies) δημιουργούν προβλήματα όπως και οι αλλαγές που είναι δυνατές καθώς είναι επαναληπτικό μοντέλο.

Το τρίτο μοντέλο που είδαμε, το Σπειροειδές Μοντέλο (Spiral Model) έχει σαν πλεονέκτημα του την υψηλή ανάλυση κινδύνου αλλά είναι καλό για μεγάλα και κρίσιμα έργα αποστολής όπου το λογισμικό παράγεται νωρίς στους κύκλους του λογισμικού και

λειτουργούν καλά για έργα όπου η ανάλυση κινδύνου περιέχει υψηλότερη προτεραιότητα.

Μειονεκτήματα που θα μπορούσαμε να πούμε για το πιο πάνω μοντέλο είναι ότι μπορεί να είναι ένα δαπανηρό μοντέλο στη χρήση, επίσης, η ανάλυση κινδύνου απαιτεί πολύ ειδική εμπειρογνωμοσύνη, και η επιτυχία του έργου εξαρτάται σε μεγάλο βαθμό από τη φάση ανάλυσης κινδύνου, και τέλος, δεν λειτουργεί καλά για μικρότερα έργα.

Το τελευταίο μοντέλο που είδαμε, το Μοντέλο Επαλήθευσης/Επικύρωσης (V-Model) που έχει σαν μοντέλο κι αυτό τα δικά του πλεονέκτημα εκ των οποίων ότι είναι απλό και εύκολο στη χρήση, αλλά και κάθε φάση έχει συγκεκριμένα παραδοτέα (deliverables), επίσης έχει υψηλότερες πιθανότητες επιτυχίας από το μοντέλο του καταρράκτη που πεθαίνει να αναπτύξει πολλά δοκιμαστικά σχέδια κατά τους κύκλους ζωής, και τέλος, λειτουργεί καλά για μικρά έργα όπου οι απαιτήσεις είναι κατανοητές.

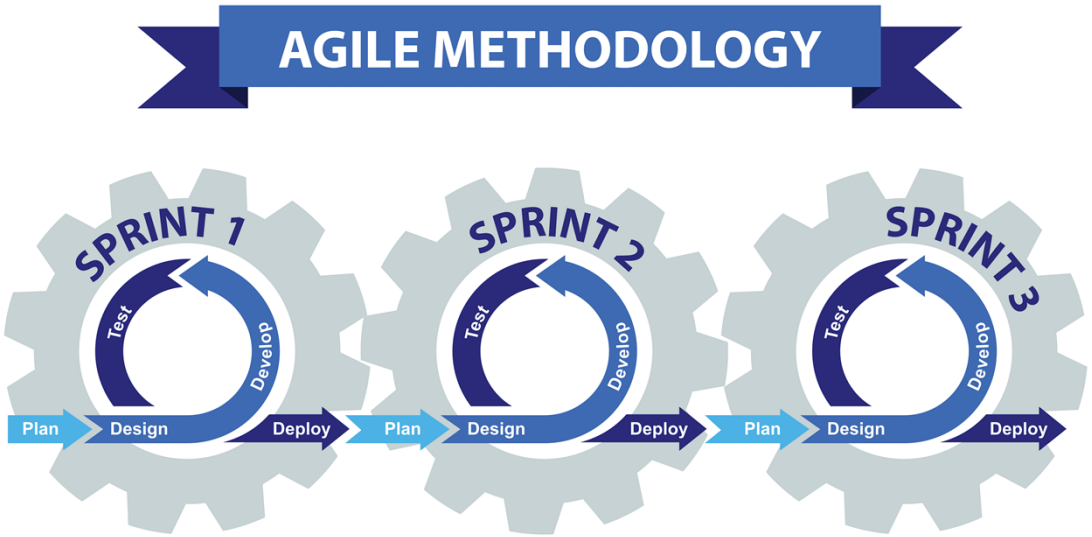
Μειονεκτήματα που θα μπορούσαμε να πούμε για το πιο πάνω μοντέλο είναι ότι είναι πολύ άκαμπτο, όπως άλλωστε και το μοντέλο του καταρράκτη. Έχει λίγη ευελιξία και προσαρμογή στις απαιτήσεις πράγμα που το κάνουν να είναι δύσκολο και δαπανηρό, επιπρόσθετο μειονέκτημα είναι ότι το λογισμικό αναπτύσσεται κατά τη φάση υλοποίησης οπότε δεν παράγονται πρόωρα πρωτότυπα του λογισμικού, και τέλος, σαν μοντέλο δεν παρέχει σαφή διαδρομή για προβλήματα που εντοπίστηκαν κατά τις φάσεις δοκιμών.

3.3.3 Επιλεγμένο Μοντέλο Ανάπτυξης Λογισμικού

Έχει επιλεχτεί το Ευκίνητο Μοντέλο (Agile Model) ανάπτυξης λογισμικού [58] αν και όπου έχω ψάξει δεν βρίσκω την ακριβείς ελληνική μετάφραση για το συγκεκριμένο μοντέλο, αλλά το συγκεκριμένο μοντέλο είναι ένας συνδυασμός του επαναληπτικού μοντέλου (iterative model) [60] και του σταδιακού μοντέλου (incremental model) [64] όπου οι διεργασίες γίνονται με έμφαση στην προσαρμοστικότητα της διαδικασίας και την ικανοποίηση των πελατών με την ταχεία παράδοση του προϊόντος λογισμικού που λειτουργεί.

Όπως αναφέρει η Maryna Demchenko στο άρθρο της [58], “Το Ευκίνητο Μοντέλο (Agile Model) σάς επιτρέπει να δημιουργείτε προϊόντα που πραγματικά θέλουν οι πελάτες,

χρησιμοποιώντας μικρούς κύκλους ("sprints") που τελειώνουν με ένα προϊόν που λειτουργεί, αν και με περιορισμένες δυνατότητες. Κάθε κύκλος περιλαμβάνει σχεδιασμό (design), ανάπτυξη (development), δοκιμή (testing) και παράθεση (deployment). Ο πελάτης ή οι ενδιαφερόμενοι μπορούν να δουν τα αποτελέσματα κάθε κύκλου και να παρέχουν τα σχόλιά τους. Στον επόμενο κύκλο, η ομάδα αναθεωρεί το προϊόν και το παρουσιάζει ξανά για τον επόμενο κύκλο σχολίων. Ως εκ τούτου, η ανάπτυξη του Agile είναι μια συνεχής διαδικασία."



Εικ. 3.3.3.1 AGILE SDLC METHODOLOGY MODEL [58]

Κεφάλαιο 4
Υλοποίηση και Αποτελέσματα

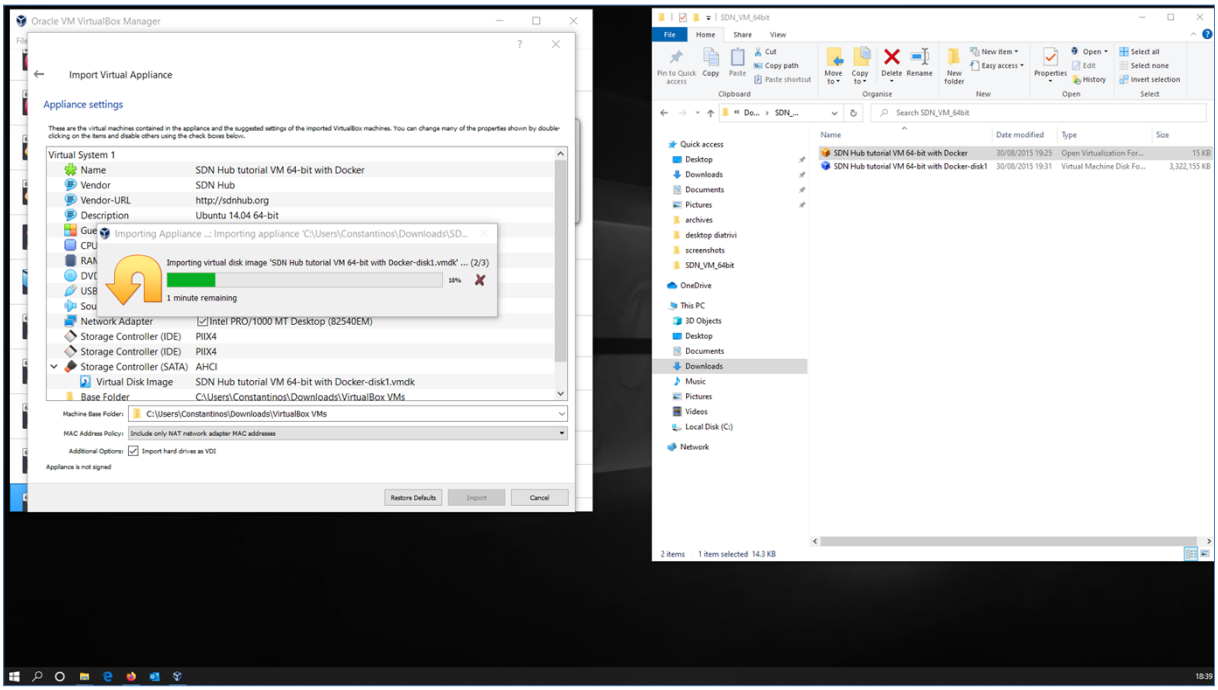
Σε αυτό το κεφάλαιο θα ασχοληθούμε με την υλοποίηση τις μεταπτυχιακής διατριβής, όπου ουσιαστικά είναι η διαδικασία κατά την οποία μια αρχική ιδέα ή ένα σχέδιο μετατρέπεται σε πράξη ή σε υλική πραγματικότητα, με αλλά λόγια, "παίρνει σάρκα και οστά".

4.1 Εισαγωγή

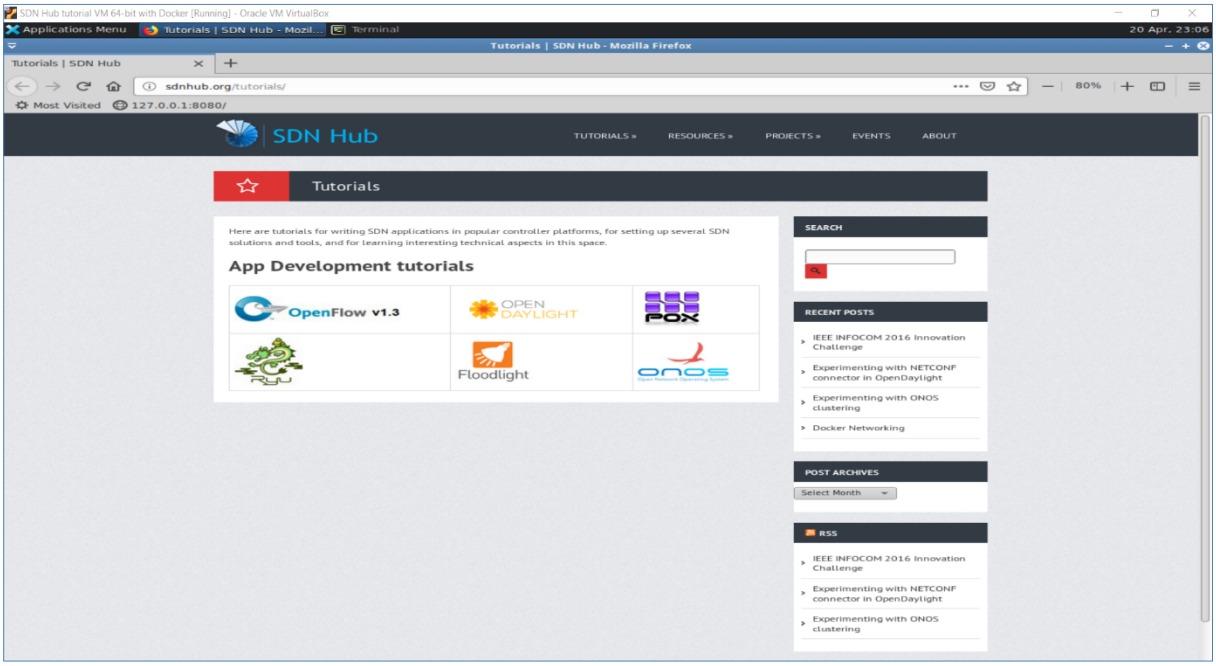
Να ξεκινήσουμε λοιπόν κι να σημειωθεί ότι έχει χρησιμοποιηθεί ένα προ-δημιουργημένο εικονικό μηχάνημα (SDN_VM_64bit.ova) [65] το οποίο ουσιαστικά είναι το λειτουργικό σύστημα των Ubuntu 14.04 64-bit το οποίο μας συνιστάτε ανεπιφύλακτα να έχει την κατανομή τουλάχιστον 2 vCPU processors και 2GB (2560MB) base memory (με αλλά λόγια τα RAM) για να μπορεί να τρέχει μέσα από το Oracle VM VirtualBox Manager 6.1.18 που είναι ο επόπτης φιλοξενίας ανοιχτού κώδικα για εικονικοποίηση που αναπτύχθηκε από την Oracle. (free and open source software)

Παραθέτω πιο κάτω τα αρχικά στιγμιότυπα οθόνης, όπου στην πρώτη, ακολουθώντας τις οδηγίες για χρήση τις εικονικής μηχανής από τον ιστότοπο τις εν λόγου προ-δημιουργημένης εικονικής μηχανής (SDN_VM_64bit.ova) με την βοήθεια του Oracle VM VirtualBox Manager 6.1.18 γίνεται η εγκατάσταση της.

Στο δεύτερο στιγμιότυπο οθόνης, αφότου ξεκίνησε το προ-δημιουργημένο εικονικό μηχάνημα και συνδεθήκαμε σε αυτό τότε από μόνο του μας ανοίγει τον ιστότοπο με τα μαθήματα (tutorials) που είναι δημιουργημένα για την χρήση των εργαλείων εκ των οποίων πολλά από αυτά δεν είναι επικαιροποιημένα (updated) και κάποιοι σύνδεσμοι είναι απαρχαιωμένοι (outdated).



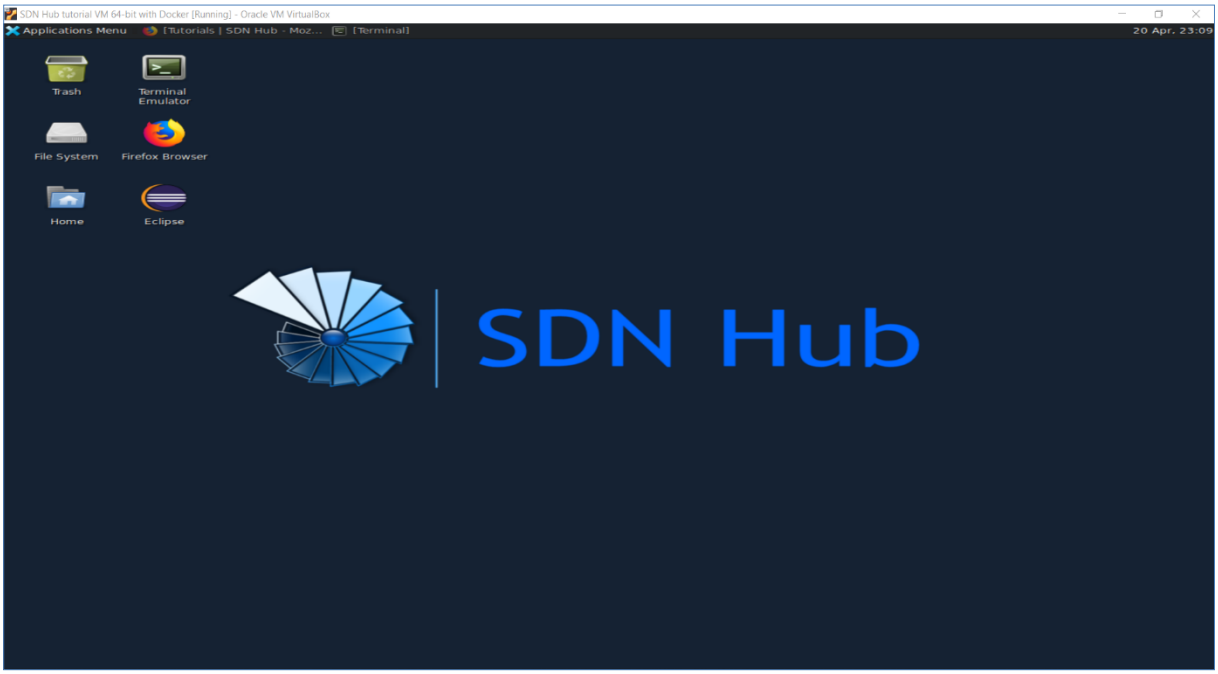
Εικ. 4.1.1 ΕΓΚΑΤΑΣΤΑΣΗ ΠΡΟ-ΔΗΜΙΟΥΡΓΗΜΕΝΗΣ ΜΗΧΑΝΗΣ



Εικ. 4.1.2 ΕΚΚΙΝΗΣΗ ΠΡΟ-ΔΗΜΙΟΥΡΓΗΜΕΝΗΣ ΜΗΧΑΝΗΣ

Φυσικότατα όσοι σύνδεσμοι ήταν απαρχαιωμένοι (outdated) κατάφερα και βρήκα μέσα από το διαδίκτυο τους σωστούς συνδέσμους.

Εν συνέχεια, παραθέτω στιγμιότυπο οθόνης από το αρχικό ολοκληρωμένο περιβάλλον ανάπτυξης (IDE – Integrated Development Environment) οπου το περιβάλλον επιφάνειας εργασίας (desktop environment) είναι το XFCE. [66]



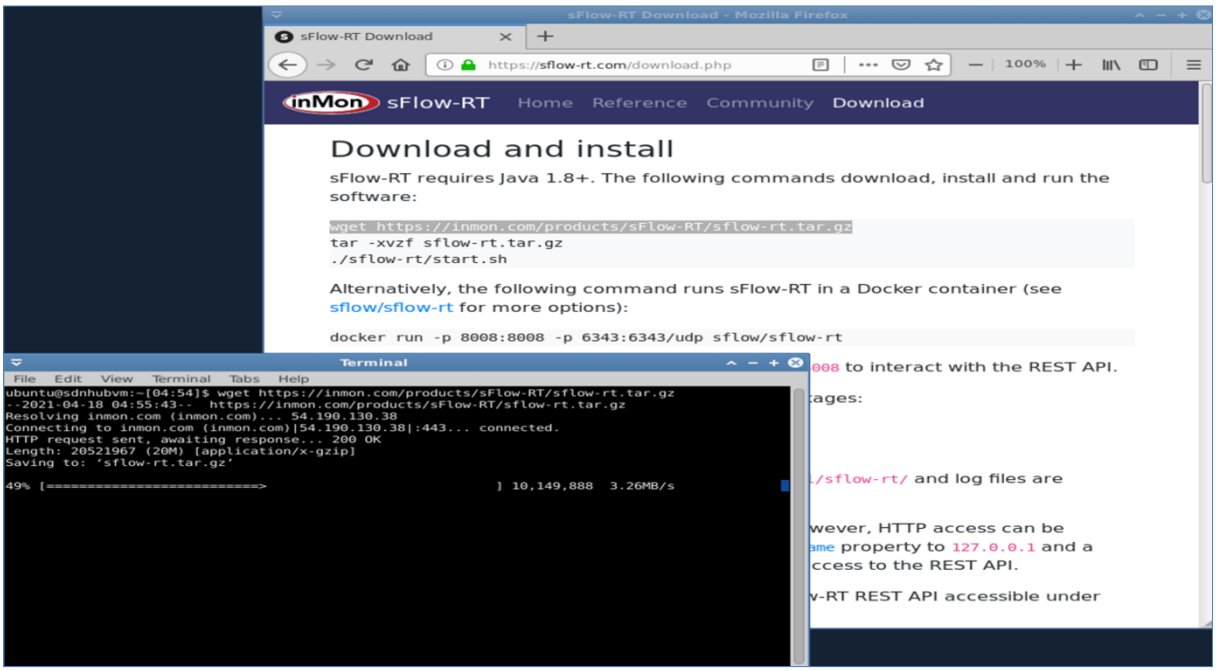
Εικ. 4.1.3 IDE ΠΡΟ-ΔΗΜΙΟΥΡΓΗΜΕΝΗΣ ΜΗΧΑΝΗΣ

Το προ-δημιουργημένο αυτό εικονικό μηχάνημα μας προμηθεύει με μια σειρά από εφαρμογές, προγράμματα και εργαλεία που οι πλειονότητα (ευτυχώς) από αυτές είναι δεν είναι απαρχαιωμένες όπου είναι τα εξής:

1. Ελεγκτές δικτύου που καθορίζονται από λογισμικό:
    1. OpenDaylight,
    2. ONOS,
    3. RYU,
    4. Floodlight,
    5. Floodlight-OF1.3,
    6. POX και
    7. Trema
2. Παρέχει το λογισμικό του εικονικού διακόπτη πολλαπλών επίπεδων και ποιότητας παραγωγής (multilayer virtual switch and production quality software) Open vSwitch 2.3.0 με υποστήριξη πρωτοκόλλου στο Openflow 1.2, 1.3, 1.4 καθώς και το LINC switch που είναι ένα OpenFlow λογισμικό γραμμένο στην Erlang.
3. Εξομοιωτής Δικτύου Ανοιχτού Κώδικα: Mininet
4. Pyretic που είναι μια γλώσσα προγραμματισμού οι οποία είναι ενσωματωμένη στην Python, η οποία προγραμματίζει τους διακόπτες δικτύου.

5. Το κουτί ανάπτυξης JAVA (JDK 1.8)
6. Eclipse Luna που υποστηρίζει διάφορα εργαλεία της Java 8 αλλά και τα εξής:
    1. Ενσωμάτωση με το Apache Maven,
    2. Ενσωμάτωση με το Xtext
    3. Ενσωμάτωση με το Xtend
    4. Ενσωμάτωση με το Web Tools Platform και
    5. Διάφορα αλλά Plug-in Development Tools
7. Ξεχωριστή εγκατάσταση του Apache Maven για την καλύτερη διαχείριση και κατανόηση διάφορων άλλων λογισμικών.
8. Wireshark που είναι αναλυτής πακέτων δικτύου με υποστήριξη στο πρωτόκολλο OpenFlow.

Το μοναδικό λογισμικό που χρειάστηκε να εγκαταστήσω είναι το sFlow-RT το οποίο είναι μια μηχανή ανάλυσης η οποία λαμβάνει μια συνεχή ροή τηλεμετρίας από τους πράκτορες sFlow που είναι ενσωματωμένοι σε συσκευές δικτύου, σε άλλους οικοδεσπότες (υπολογιστές) ή κι ακόμα σε εφαρμογές που τις μετατρέπει σε δραστήριες μετρήσεις οι οποίες είναι προσβάσιμες μέσω του RESTflow API, όπου αυτό με τη σειρά του διευκολύνει τη διαμόρφωση προσαρμοσμένων μετρήσεων, την ανάκτηση μετρήσεων, τον καθορισμό ορίων και τη λήψη ειδοποιήσεων.



Εικ. 4.1.4 ΕΓΚΑΤΑΣΤΑΣΗ sFlow-RT

4.2 Αποτελέσματα

Εν συνέχεια, θα ασχοληθούμε με τα αποτελέσματα που βγήκαν από την υλοποίηση του προηγουμένου κεφαλαίου, συνεπώς θα ασχοληθούμε με την προσομοίωση ενός SDN (Software Defined Network) αλλά και παράλληλα με το αντίθετο τους δηλαδή τα παραδοσιακά δίκτυα (Traditional Networks).

4.2.1 Pietro Manzoni's Mininet Methodology

Στην ακόλουθη ενότητα, θα χρησιμοποιηθεί η μεθοδολογία του Pietro Manzoni [67] μέσα από διάφορα στιγμιότυπα οθόνης και με την βοήθεια του εξομοιωτή δικτύου ανοιχτού κώδικα Mininet αλλά και παράλληλα του αναλυτή πακέτων δικτύου Wireshark.

Αρχικά όπως θα παρατηρήσουμε από το πρώτο στιγμιότυπο οθόνης που παραθέτουμε πιο κάτω, μέσα από το τερματικό (terminal) από το εικονικό μηχάνημα απενεργοποιούμε τον πύρινο τοίχο (firewall) του εικονικού μηχανήματος από το να ξεκίνα κάθε φορά που ξεκινάει το μηχάνημα (machine startup) με την ακόλουθη γραμμή εντολής:

sudo ufw disable

όπου:

sudo: super user do
ufw: uncomplicated firewall
status: enable/disable/status/allow/deny/reset

Εν συνέχεια, μέσα στο τερματικό (terminal) από το εικονικό μηχάνημα δημιουργούμαι ένα δίκτυο καθορισμένο από το λογισμικό (SDN – Software Defined Network) εκτελώντας την ακόλουθη γραμμή εντολής:

sudo mn –arp –topo single,3 –mac –controller remote –switch ovsk

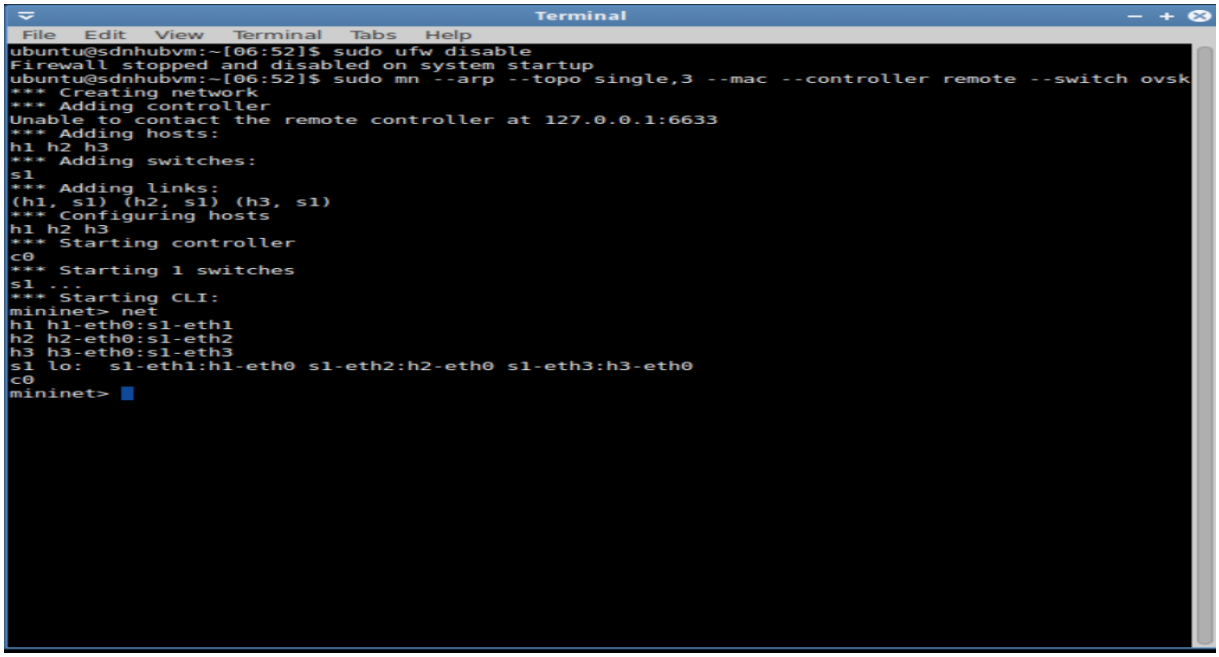
όπου:

sudo: super user do
mn: mininet command
arp: static ARP entries for each host in each other
topo: mininet topology
mac: auto set MAC addresses

controller: the remote controller

switch: kernel modes

Αυτό δημιουργεί μια απλή τοπολογία που αποτελείται από τρεις οικοδεσπότες (hosts), ένα διακόπτη (switch), και έναν ελεγκτή (controller). Επιπρόσθετα μέσα στο τερματικό (terminal) με την γραμμή εντολής, **net**, εμφανίζονται οι πληροφορίες όλων των κόμβων (nodes) που έχουν δημιουργηθεί.

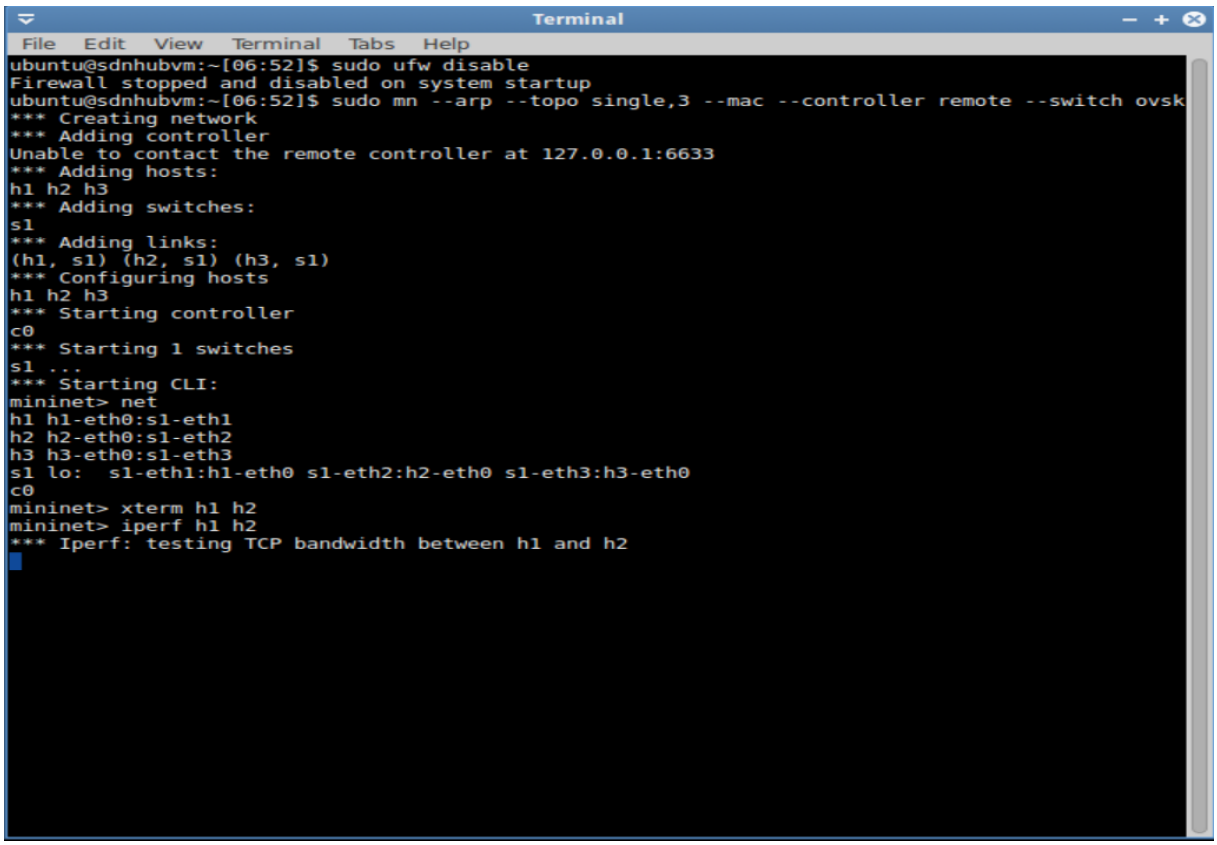


Εικ. 4.2.1.1 ΔΗΜΙΟΥΡΓΙΑ SDN

Ακολούθως μέσα στο τερματικό (terminal) με τις ακόλουθες γραμμές εντολών που εκτελέστηκαν, όπου με την πρώτη γραμμή εντολής, ανοίγουμε τα τερματικά για τους οικοδεσπότες (hosts) του δικτύου και με τη δεύτερη, παίρνουμε μέτρα για το εύρος ζώνης (TCP bandwidth) μεταξύ των δυο οικοδεσποτών (hosts) του δικτύου.

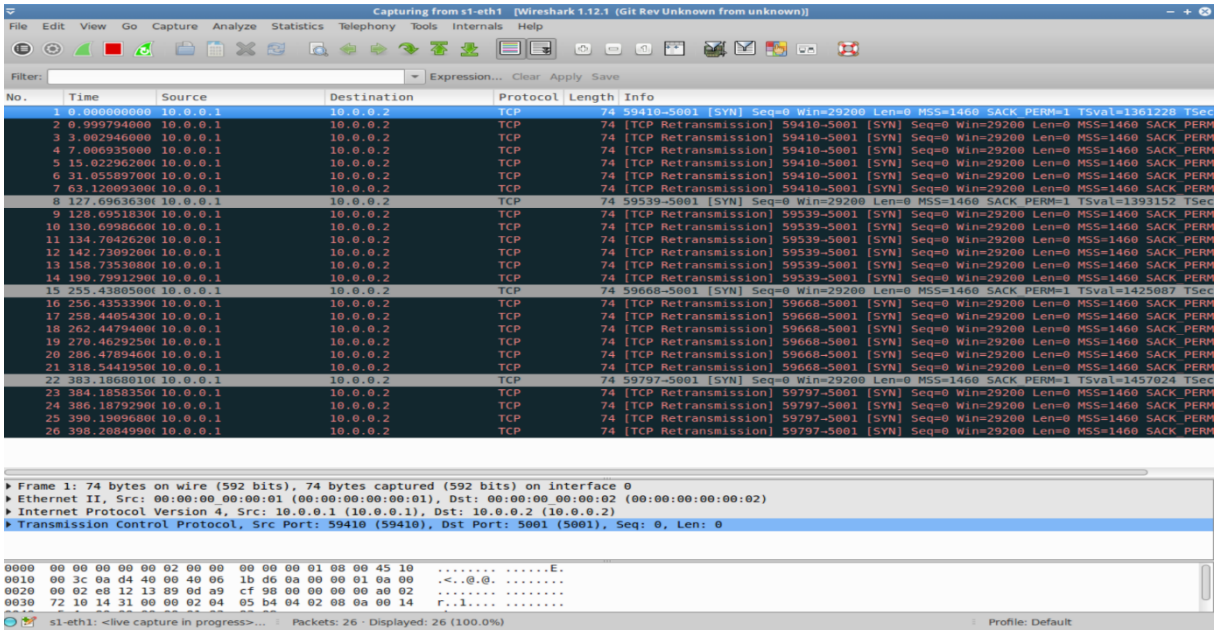
xterm h1 h2

iperf h1 h2



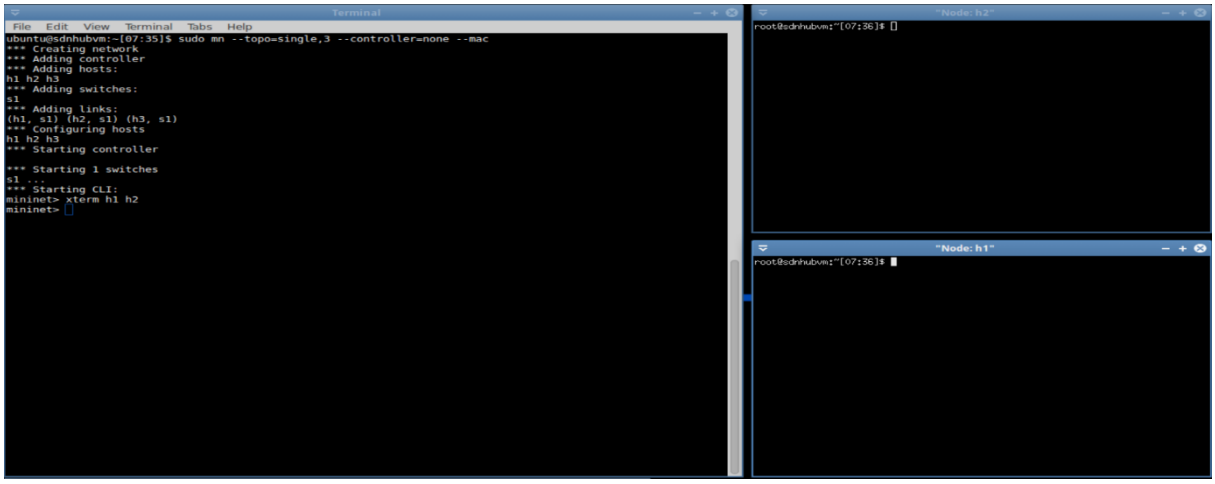
Εικ. 4.2.1.2 ΕΛΕΧΓΟΣ TCP BANDWIDTH

Επιπρόσθετα από το τερματικό (terminal) του πρώτου οικοδεσπότη (host) h1 εκτελώντας την εντολή, **wireshark &** ανοίγουμε τον αναλυτή πακέτων δικτύου wireshark για να δούμε ότι όντως διακινούνται τα πακέτα μέσα στο δίκτυο που καθορίζονται από λογισμικό (SDN).



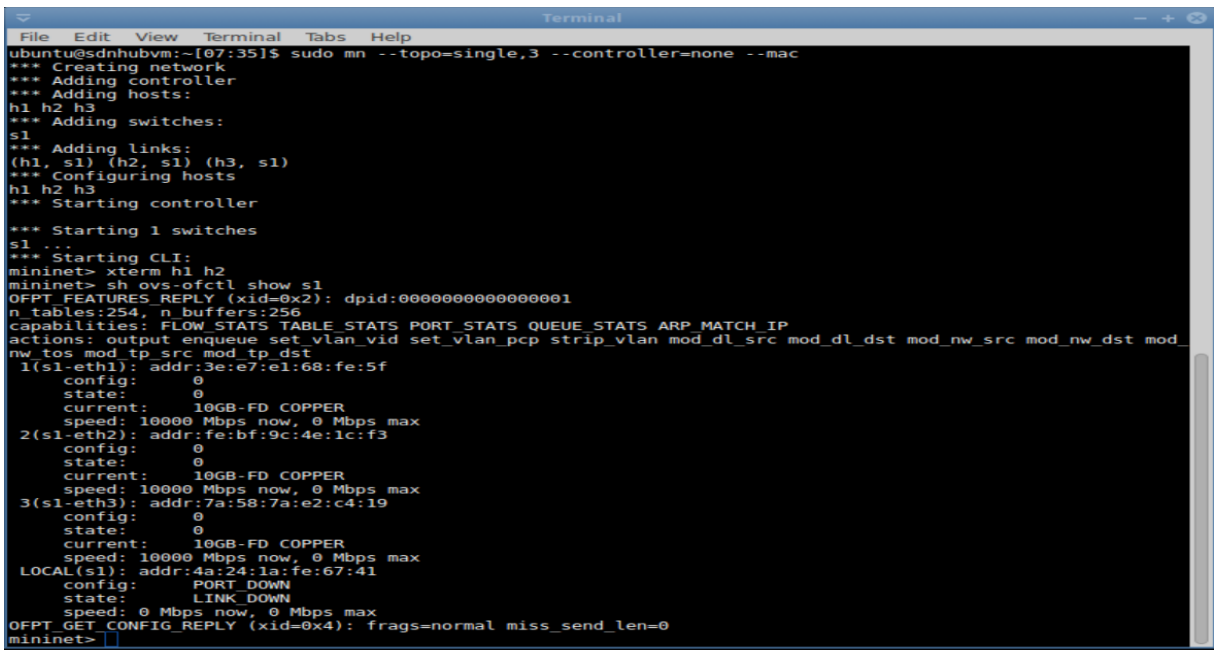
Εικ. 4.2.1.3 WIRESHARK

Εν συνέχεια με τις ακόλουθες γραμμές εντολών που εκτελέστηκαν, **sudo mn –topo=single,3 –controller=none –mac** όπου με αυτήν δημιουργούμε ένα παραδοσιακό δίκτυο με τρεις οικοδεσπότες (hosts) και ένα διακόπτη (switch), με αυτήν τη γραμμή εντολής, **xterm h1 h2** ανοίγουμε τα τερματικά για τους οικοδεσπότες (hosts) του δικτύου.



Εικ. 4.2.1.4 ΔΗΜΙΟΥΡΓΙΑ ΠΑΡΑΔΟΣΙΑΚΟΥ ΔΙΚΤΥΟΥ

Επίσης εκτελώντας αυτή τη γραμμή εντολής, **sh ovs-ofctl show s1** κάνουμε αντιστοίχιση κόμβων σε αριθμούς του πρωτοκόλλου Openflow, με αλλά λόγια είναι η γραμμή εντολής όπου αλλάζει ο χάρτης των θυρών του διακόπτη (s1).

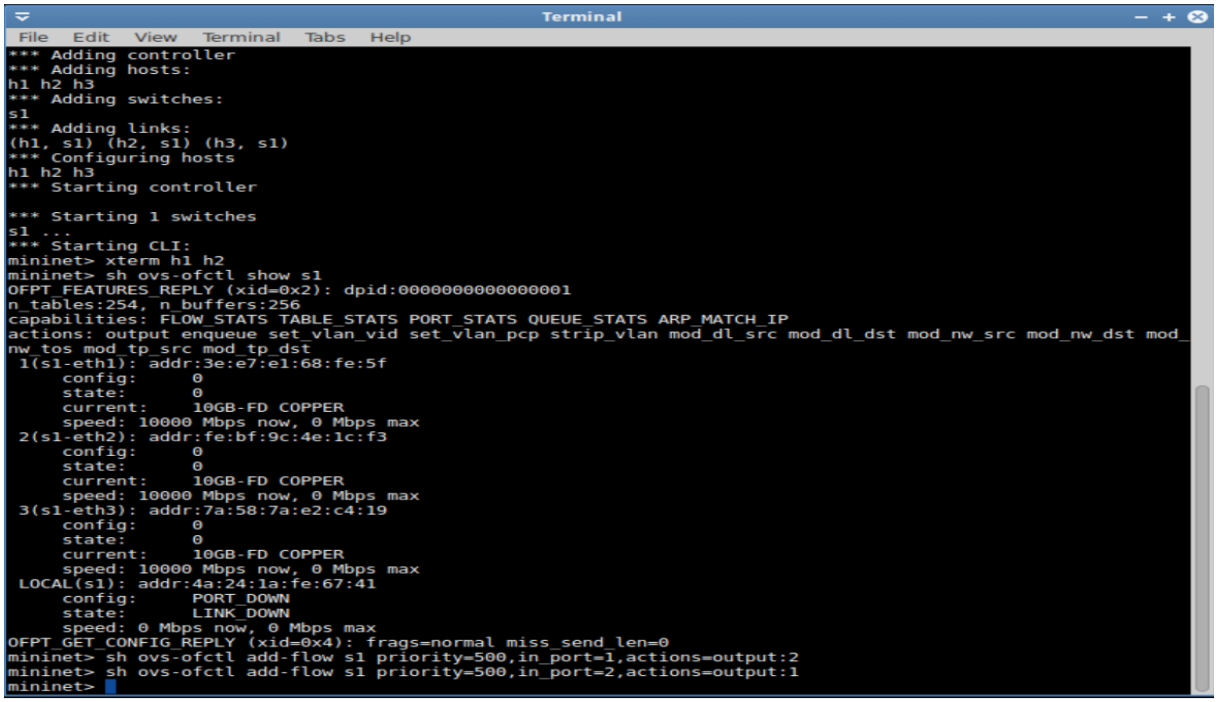


Εικ. 4.2.1.5 ΑΛΛΑΓΗ ΘΥΡΩΝ ΧΑΡΤΗ

Επιπρόσθετα εκτελώντας τις ακόλουθες γραμμές εντολών, δημιουργείτε μια ροή κυκλοφορίας μεταξύ των δυο οικοδεσποτών (hosts – h1 και h2)

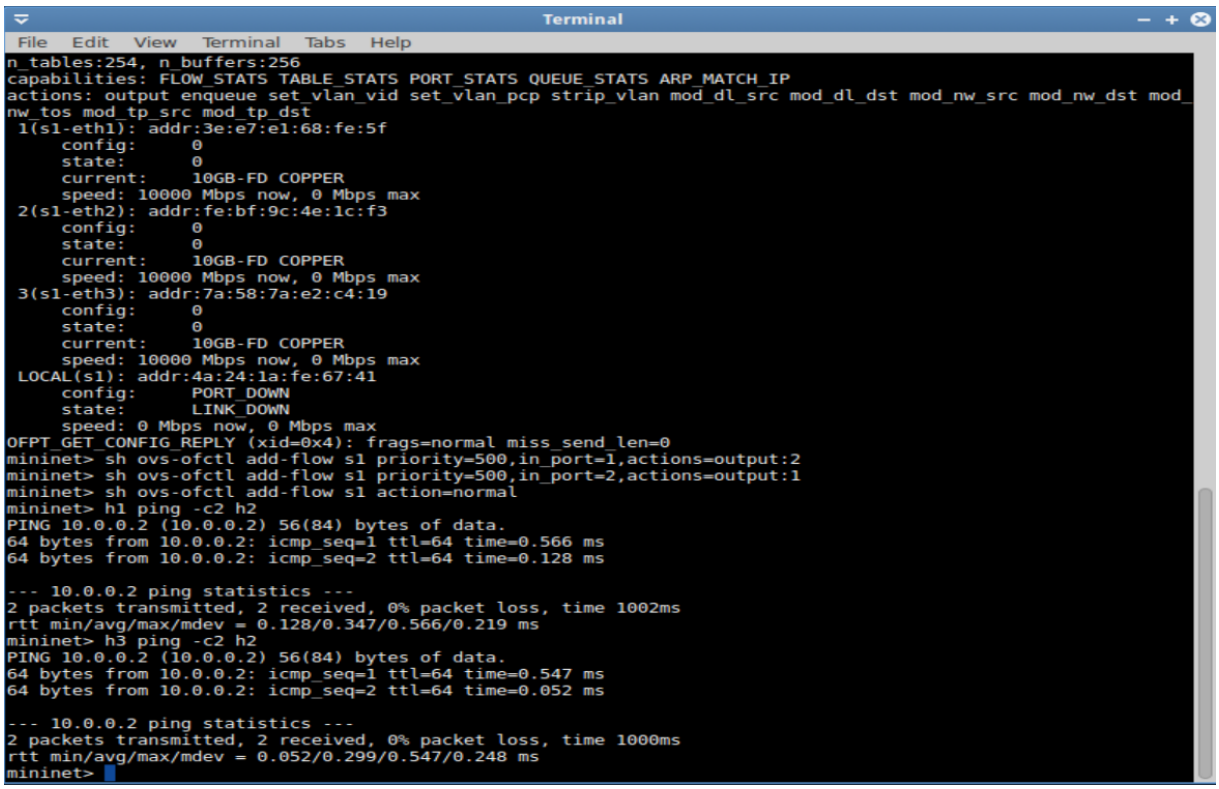
sh ovs-ofctl add-flow s1 priority=500,in_port=1,actions=output:2

sh ovs-ofctl add-flow s1 priority=500,in_port=2,actions=output:1



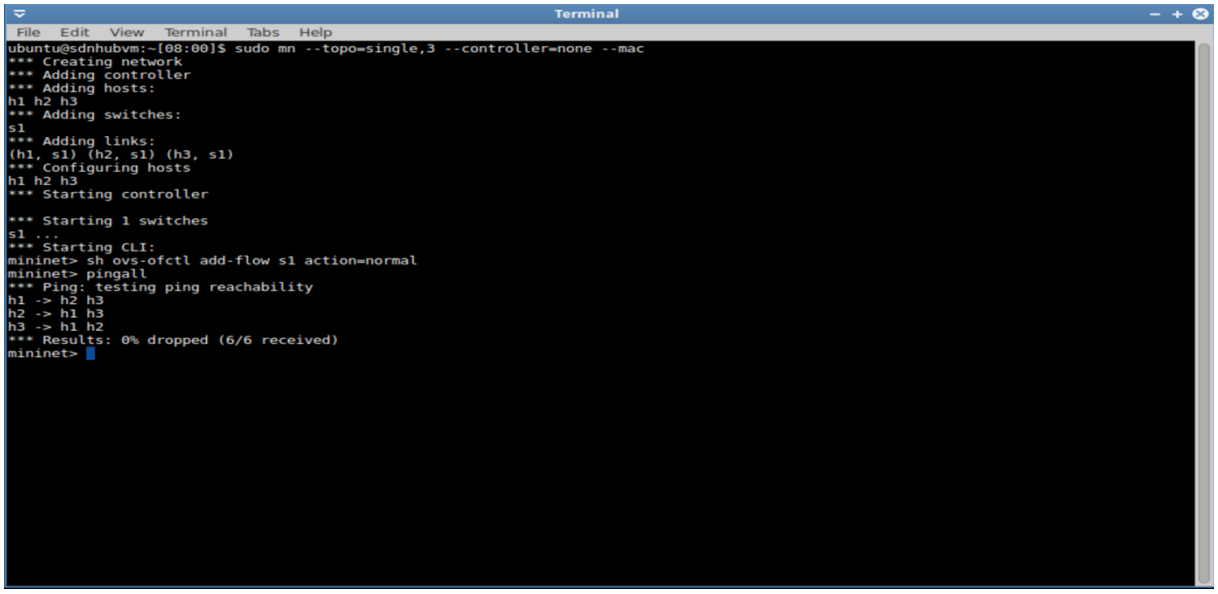
Εικ. 4.2.1.6 ΡΟΗ ΚΥΚΛΟΦΟΡΙΑΣ ΔΥΟ ΟΙΚΟΔΕΣΠΟΤΩΝ

Ακολούθως εκτελώντας αυτήν την εντολής, **h3 ping -c2 h2,** ο οικοδεσπότης 3 (host 3 - h3) κάνει ping δυο φορές τον οικοδεσπότη 2 (host 2 – h2)



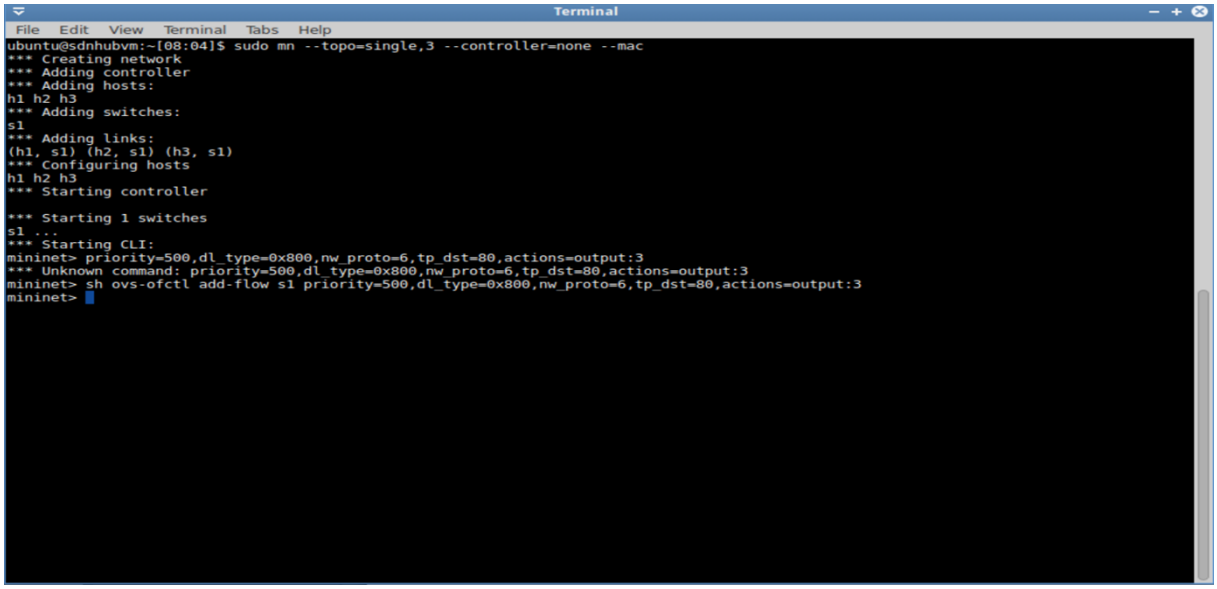
Εικ. 4.2.1.7 PING ΔΥΟ ΟΙΚΟΔΕΣΠΟΤΩΝ

Επίσης εκτελώντας τις πιο κάτω γραμμές εντολών ελέγχουμε τη λειτουργιά των δημιουργημένων ροών κυκλοφορίας.



Εικ. 4.2.1.8 ΕΛΕΓΧΟΣ ΔΗΜΙΟΥΡΓΗΜΕΝΩΝ ΡΟΩΝ

Εν συνέχεια με αυτήν την εντολής, **sh ovs-ofctl add-flow s1 priority=500,dl_type=0x800,nw_proto=6,tp_dst=80,actions=output:3** δημιουργούμαι ένα κανόνα για την προώθηση ροών στον οικοδεσπότη 3 (host 3 - h3)



Εικ. 4.2.1.9 ΔΗΜΙΟΥΡΓΕΙΑ ΠΡΩΟΘΗΣΗΣ ΡΟΩΝ

Επιπρόσθετα εκτελώντας τις ακόλουθες γραμμές εντολών,

h3 python2 -m SimpleHTTPServer 80 &

sh ovs-ofctl add-flow s1 arp,actions=normal

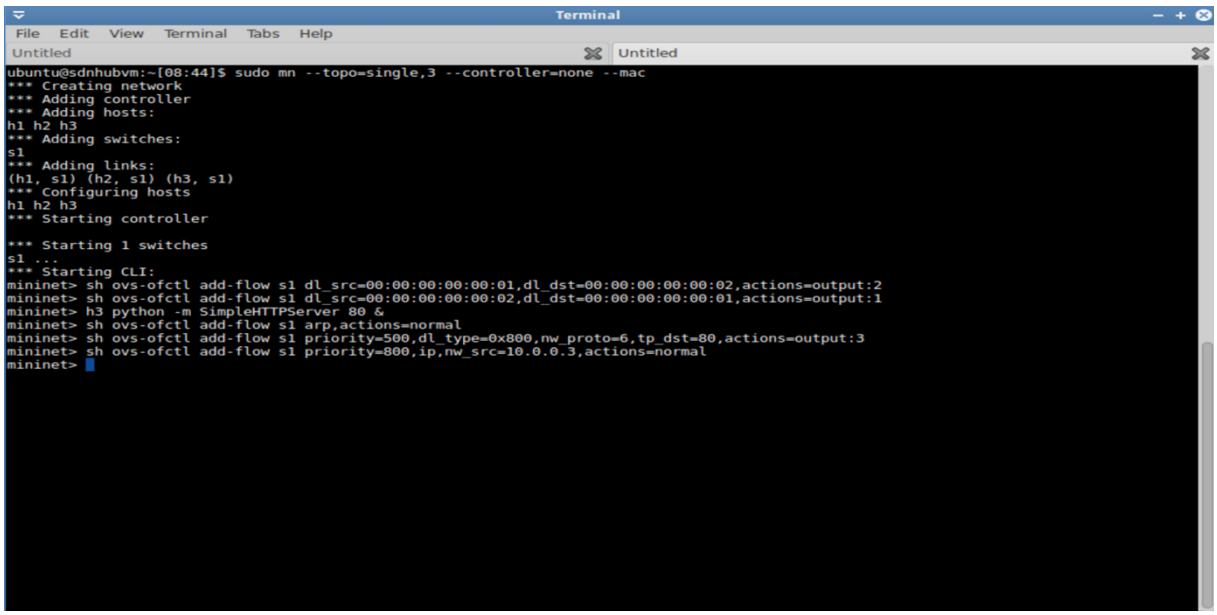
sh ovs-ofctl add-flow s1

priority=500,dl_type=0x800,nw_proto=6,tp_dst=80,actions=output:3

sh ovs-ofctl add-flow s1 priority=800,ip,nw_src=10.0.0.3,actions=normal

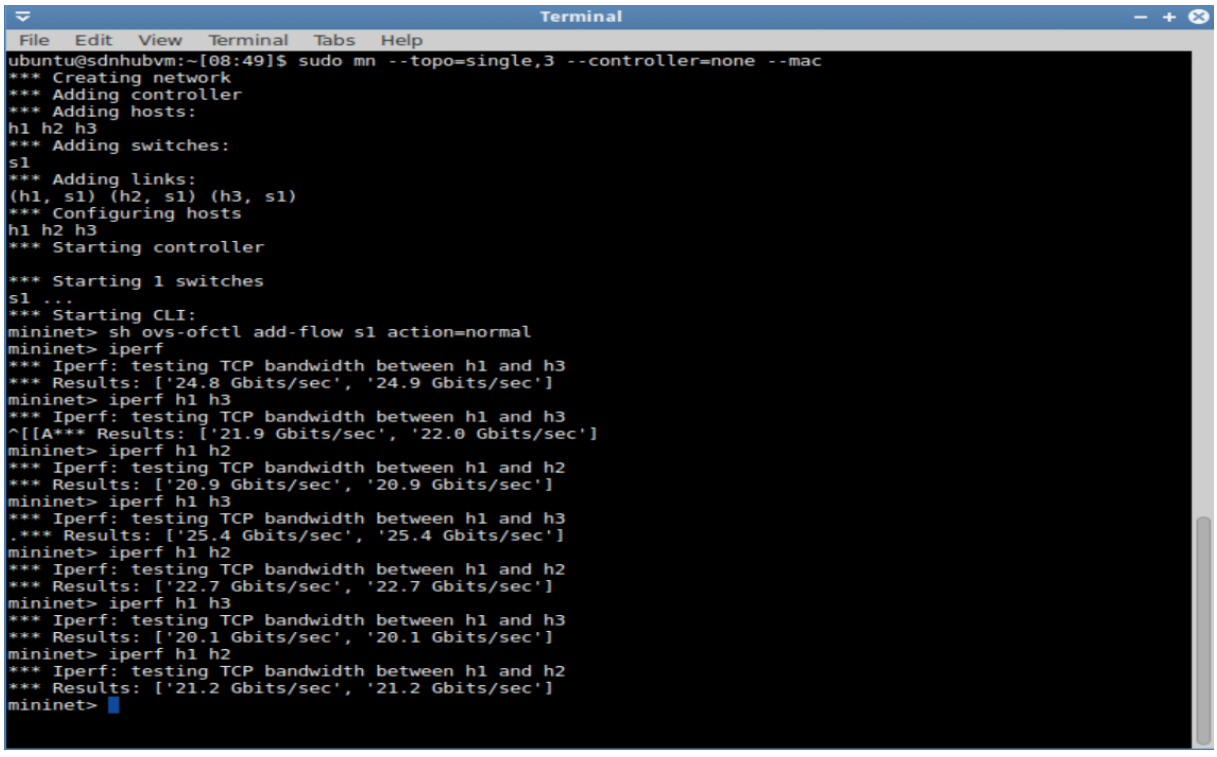
όπου με την πρώτη, ο οικοδεσπότης 3 (host 3 – h3) γίνεται ένας web server, ενώ με την δεύτερη, ενεργοποιούνται η στατικές καταχωρήσεις ARP για όλους τους οικοδεσπότες, επίσης με την τρίτη, δημιουργείτε ένας κανόνας που προωθεί όλη την κυκλοφορία TCP (nw_proto = 6) με τη θύρα προορισμού 80 στη θύρα εναλλαγής 3, και τέλος με την τέταρτη γραμμή εντολής, η οποία μειώνει την όλη συνδεσιμότητα.

Να σημειωθεί ότι η τρίτη γραμμή εντολής, θα μπορούσε να χρησιμοποιηθεί για την ανακατεύθυνση όλης της κυκλοφορίας δεδομένων σε ένα τείχος προστασίας (firewall) που είναι συνδεδεμένο σε μια συγκεκριμένη θύρα.



Εικ. 4.2.1.10 ΕΛΕΓΧΟΣ ΡΟΩΝ

Επιπλέον στο πιο κάτω στιγμιότυπο οθόνης, παίρνουμε μέτρα για το εύρος ζώνης (TCP bandwidth) μεταξύ όλων των οικοδεσποτών (hosts) του δικτύου.



Εικ. 4.2.1.11 TCP BANDWIDTH

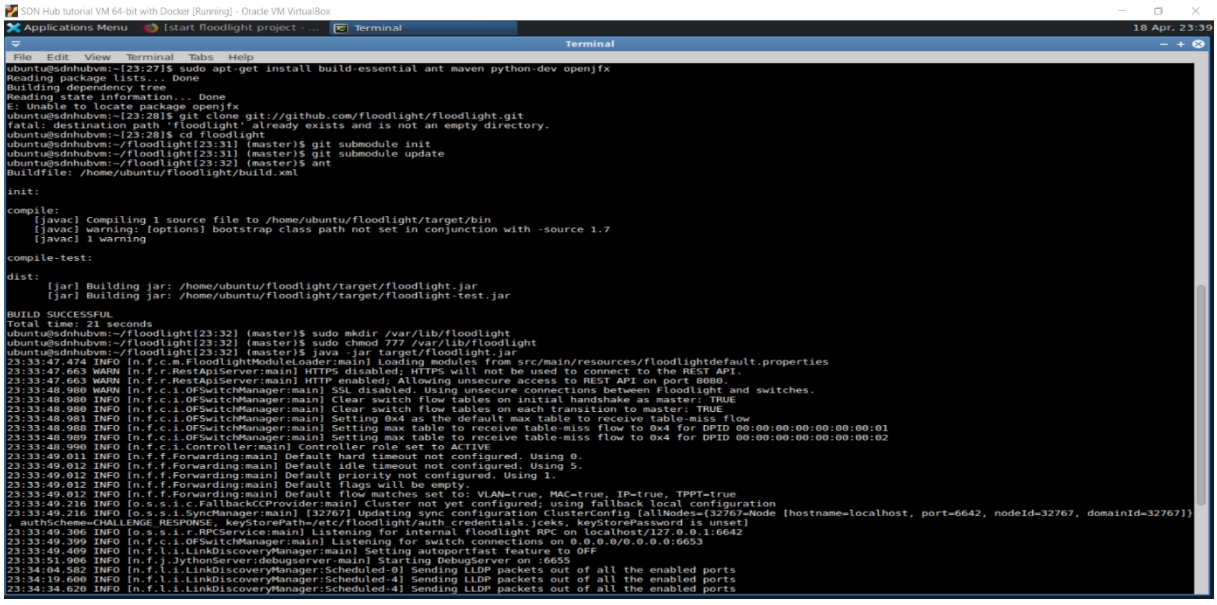
4.3 Floodlight Project

Το Floodlight Project αρχικά είχε τον δικό του ιστότοπο projectfloodlight.org όπου με την πάροδο του χρόνου μετα-εξελίχθηκε σε Floodlight OpenFlow Controller κι έτσι αναγκαστικά μετακόμισε σε ένα καινούργιο wiki ιστότοπο όπου ήταν το σπιτικό της Atlassian.

Γενικότερα το floodlight είναι ελεγκτής ανοιχτού κώδικα που υποστηρίζει πρωτόκολλα OpenFlow 1.0 έως 1.5 αλλά και Java OpenFlow.

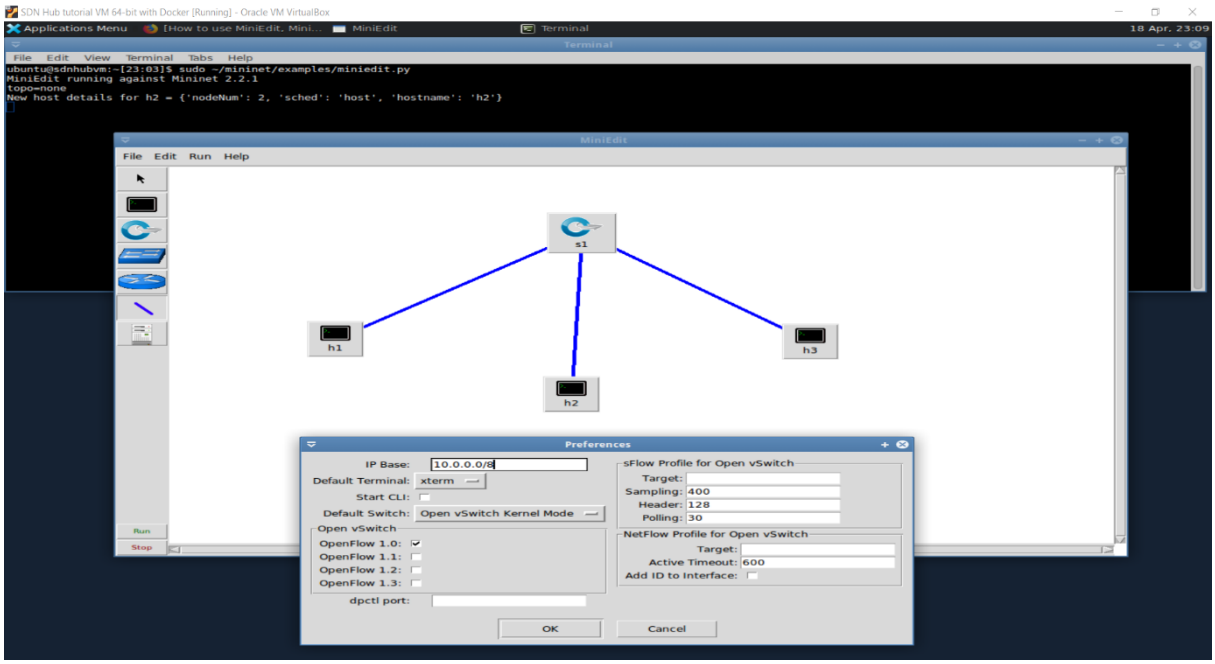
4.3.1 Floodlight Controller

Αρχικά χρειάστηκε με την ακόλουθη γραμμή εντολής **sudo apt-get install build-essential openjdk-7-jdk ant maven python-dev eclipse** να ξανά-εγκατασταθεί το floodlight controller και να ξανά-κλωνοποιηθεί από την github αποθήκη (repository) με την γραμμή εντολής **git clone git://github.com/floodlight/floodlight.git**



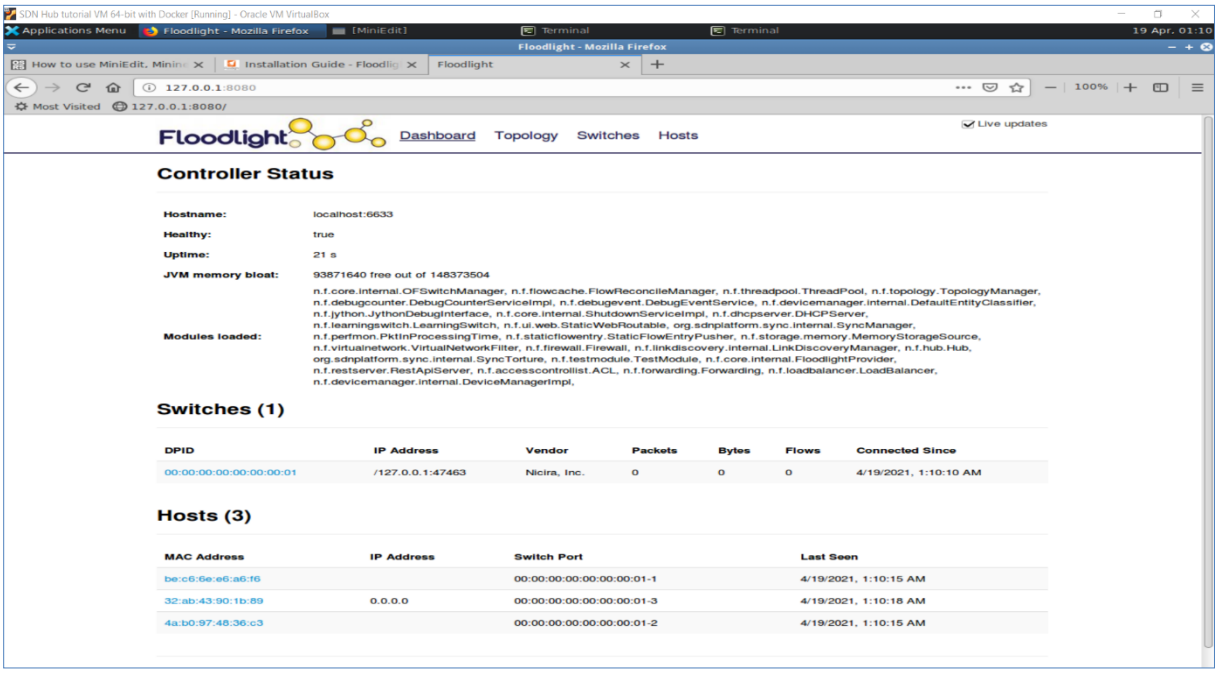
Εικ. 4.3.1 ΕΓΚΑΤΑΣΤΑΣΗ FLOODLIGHT

Όπου στην συνέχεια με την εντολή **java -jar target/floodlight.jar** ξεκινούσε μέσα στο τερματικό (terminal). Εν συνέχεια στο πιο κάτω στιγμιότυπο οθόνης, με την εντολη **sudo ~/mininet/examples/miniedit.py** ξεκίνησα το mininet με γραφική διεπαφή (gui – graphical user interface) όπου και δημιούργησα τρεις οικοδεσπότες (hosts), ένα διακόπτη (switch), και κανέναν ελεγκτή (controller).

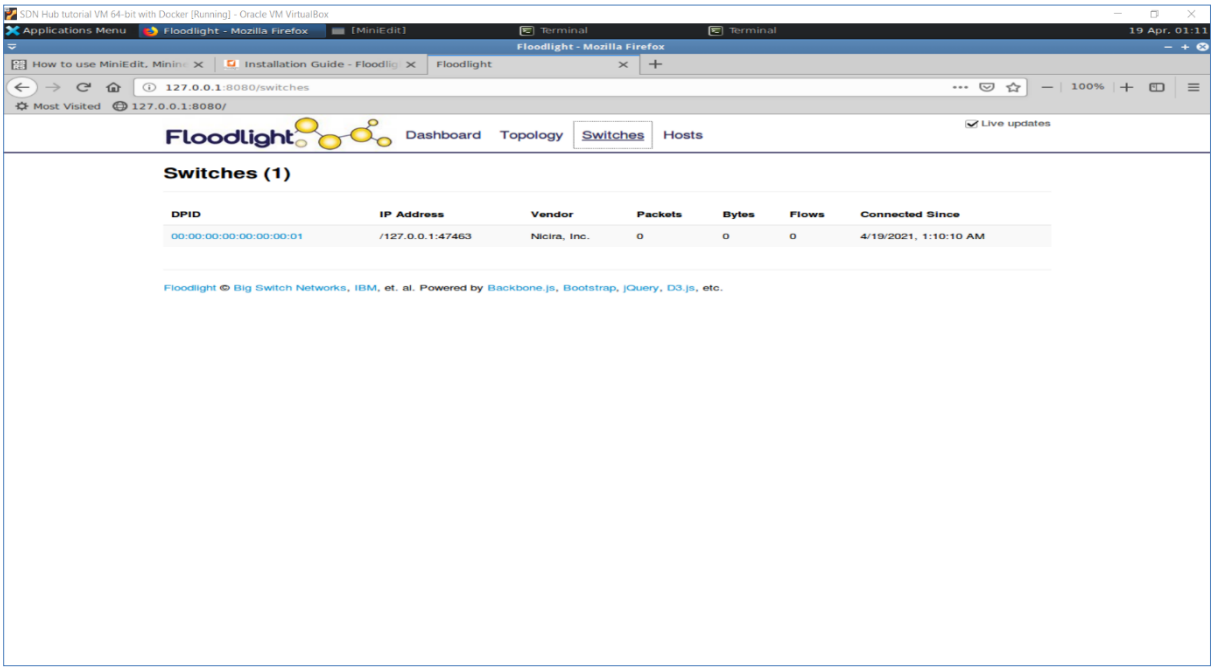


Εικ. 4.3.2 MINIEDIT

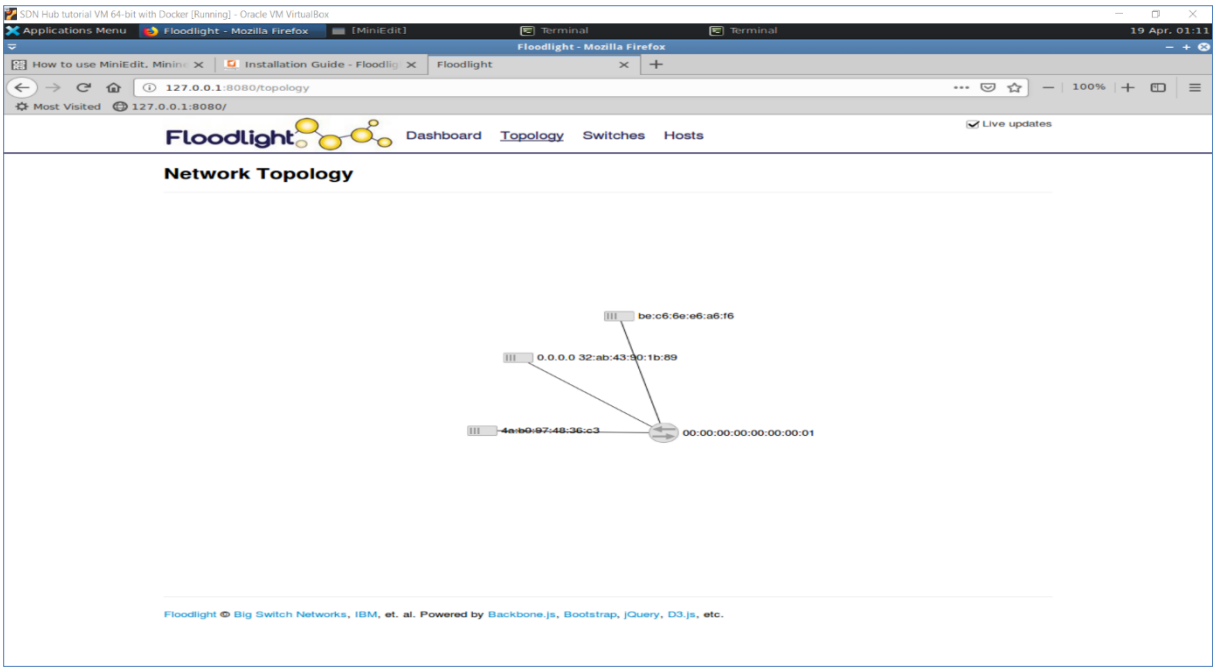
Αφότου εγκαταστάθηκε το floodlight controller, το εκτέλεσα κι έκανα άνοιγμα του πιο πάνω δικτύου πράγμα που παρατηρείτε στα επόμενα τέσσερα στιγμιότυπα οθόνης.



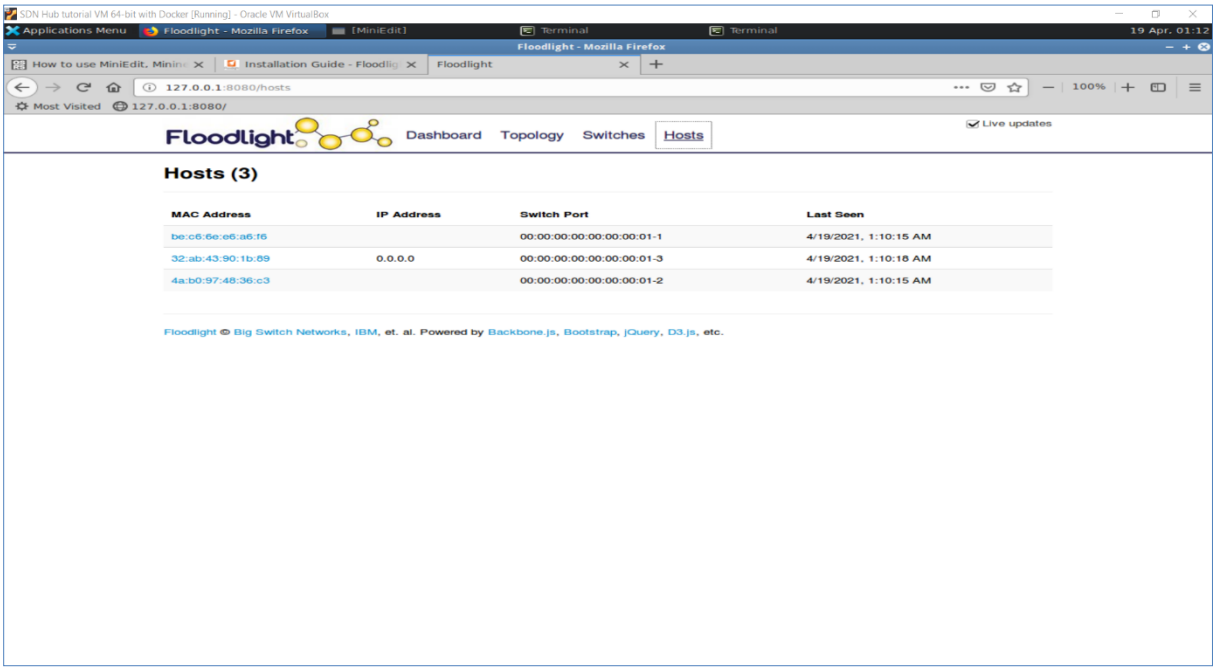
Εικ. 4.3.3 FLOODLIGHT TRADITIONAL NETWORK DASHBOARD



Εικ. 4.3.4 FLOODLIGHT TRADITIONAL NETWORK SWITCHES

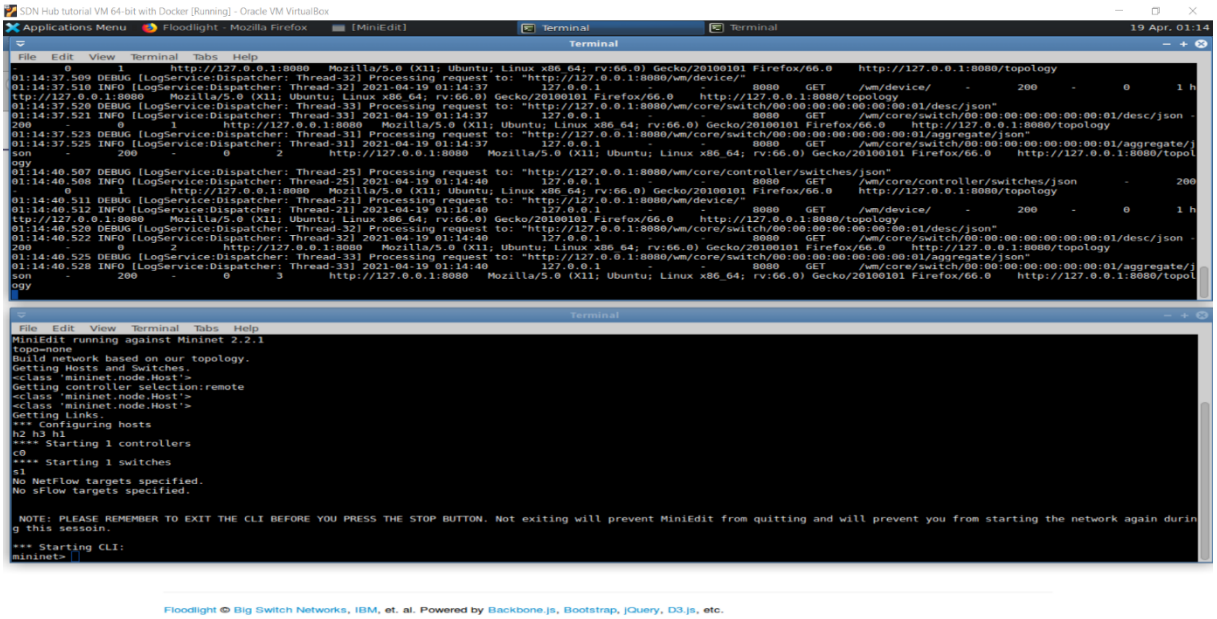


Εικ. 4.3.5 FLOODLIGHT TRADITIONAL NETWORK TOPOLOGY



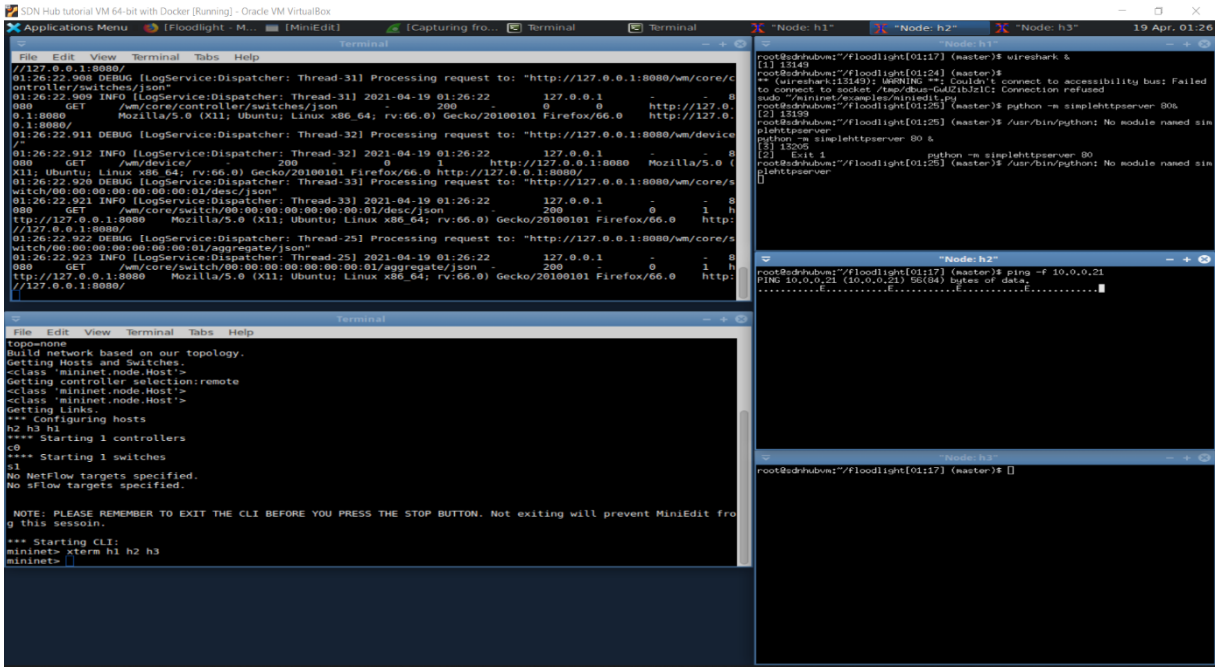
Εικ. 4.3.6 FLOODLIGHT TRADITIONAL NETWORK HOSTS

Επιπρόσθετα το στιγμιότυπο οθόνης ανήκει στο πιο πάνω δίκτυο.



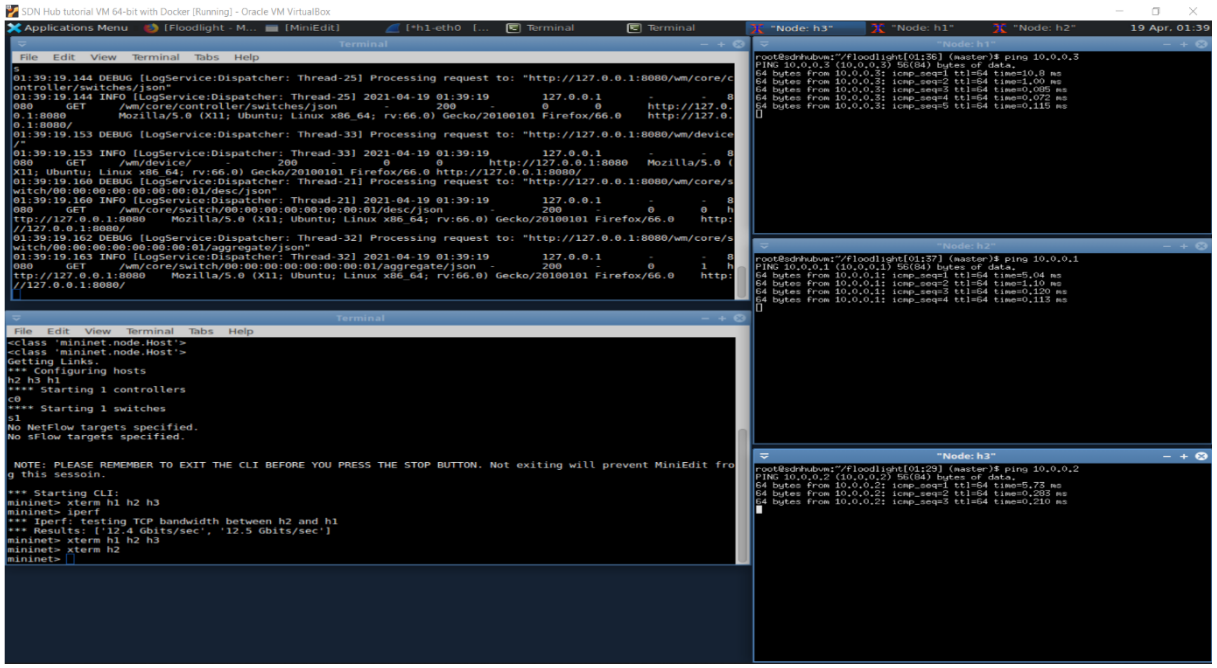
Εικ. 4.3.7 FLOODLIGHT TRADITIONAL NETWORK TERMINAL

Επιπλέον, στο πιο κάτω το στιγμιότυπο οθόνης το DDoS Attack από έναν οικοδεσπότη 2 (host 2 – h2) σε άλλον οικοδεσπότη 1 (host 1 – h1).



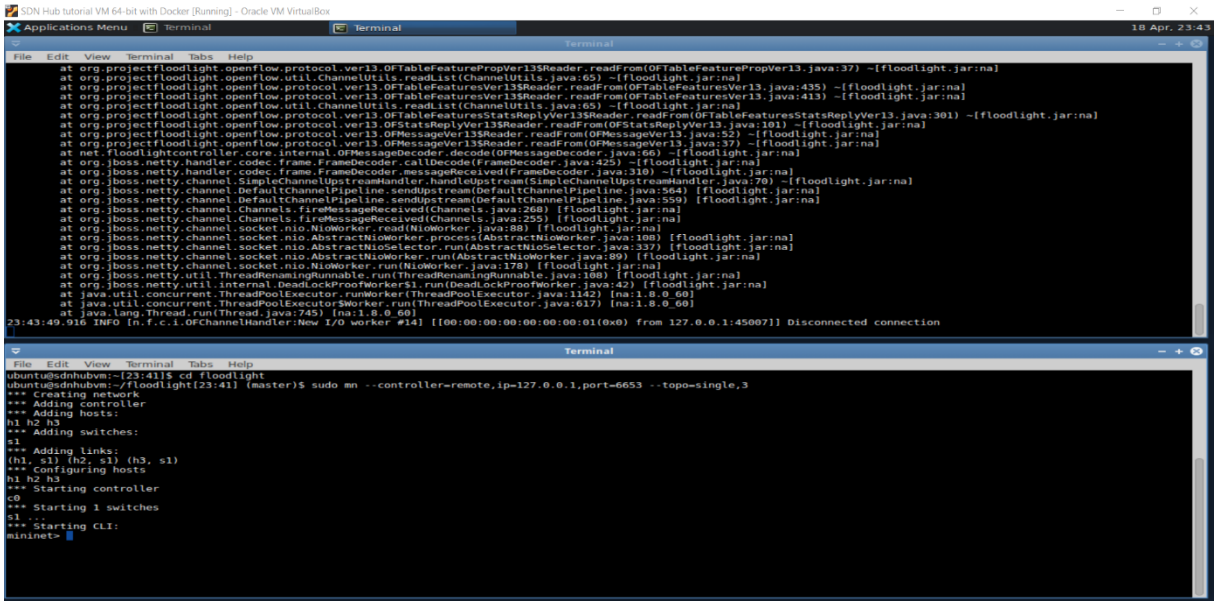
Εικ. 4.3.8 DDoS Attack on FLOODLIGHT TRADITIONAL NETWORK

Αλλά και επίσης, στο πιο κάτω το στιγμιότυπο οθόνης τα ping που κάνει ο ένας οικοδεσπότης στον άλλο. (h1>h3, h2>h1, h3>h2)



Εικ. 4.3.9 PING with 2 HOSTS on FLOODLIGHT TRADITIONAL NETWORK

Επιπρόσθετα, μέσα στο τερματικό με την εντολή **sudo mn –controller=remote,ip=127.0.0.1,port=6653 –topo=single,3** (εντολή mininet) δημιούργησα τρεις οικοδεσπότες (hosts), ένα διακόπτη (switch), και έναν ελεγκτή (controller) πράγμα που φαίνεται στο επόμενο στιγμιότυπο οθόνης.

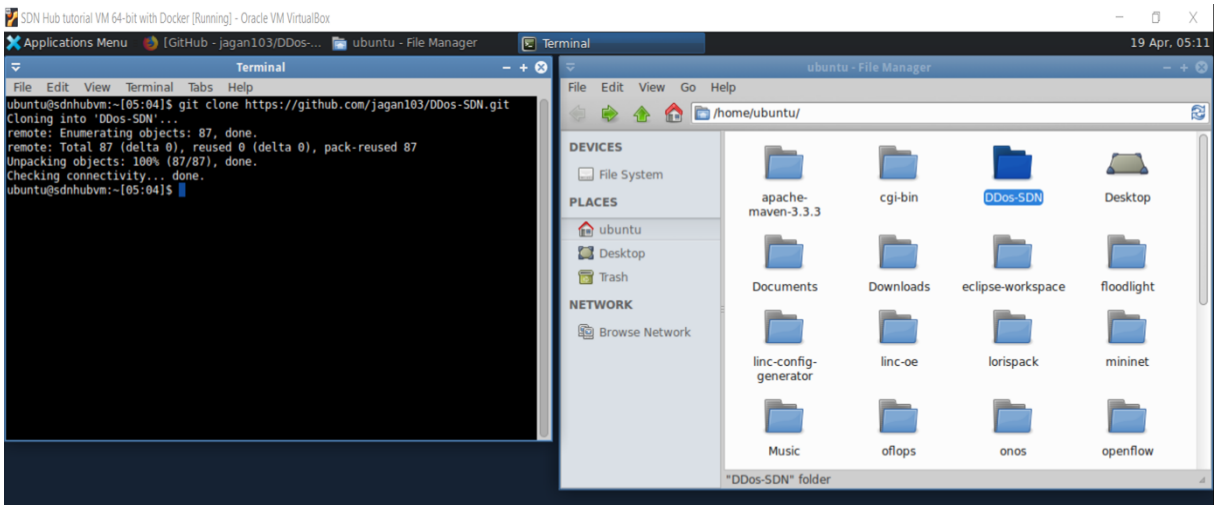


Εικ. 4.3.10 SDN with 3 HOSTS, 1 SWITCH and 1 CONTROLLER

4.4 Διανεμημένη Επίθεση Άρνησης Υπηρεσίας (DDoS Attack)

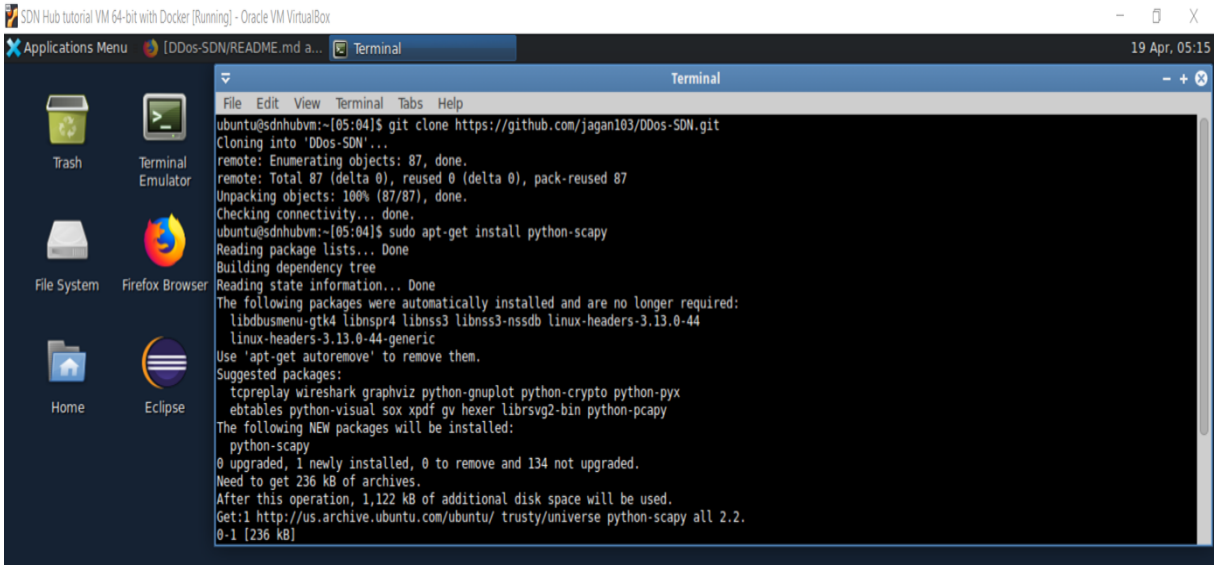
Για να καταλάβουμε πλήρως τι ακριβώς είναι η μια επίθεση διανεμημένης άρνησης υπηρεσίας (DDoS) είναι μια κακόβουλη προσπάθεια να διαταραχθεί η κανονική κυκλοφορία ενός στοχευμένου διακομιστή (server), υπηρεσίας ή δικτύου κατακλύζοντας τον στόχο ή τη γύρω υποδομή του με μια πλημμύρα από κίνηση στο δίκτυο.

Εν συνέχεια, έχω κλωνοποίηση από την github αποθήκη (repository) με την γραμμή εντολής **git clone https://github.com/jagan103/DDos-SDN.git** έναν Openflow ελεγκτή (controller) και τον χρησιμοποίησα μαζί με τον POX controller (που λειτουργά στην Python) για να παρατηρήσουμε τις συνθήκες επίθεσης, παραθέτω το πιο κάτω το στιγμιότυπο οθόνης.



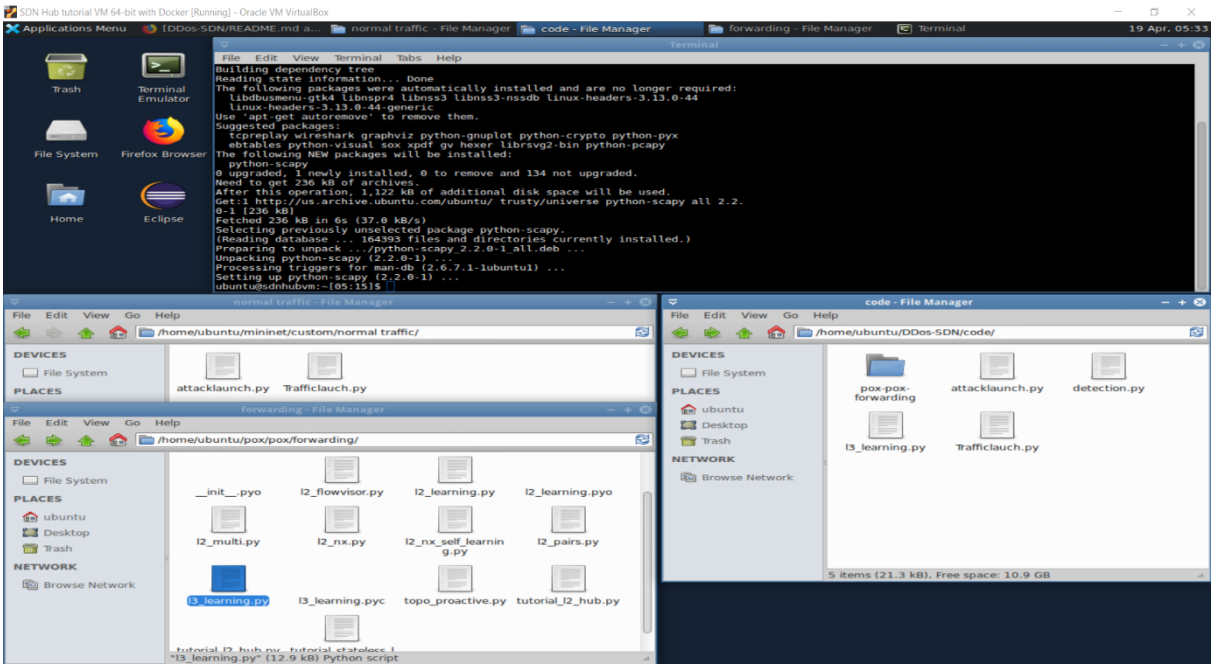
Εικ. 4.4.1 ΚΛΩΝΟΠΟΙΗΣΗ DDoS Git ΑΠΟΘΗΚΗΣ

Ακολούθως, με την εντολή **sudo apt-get install python-scapy** εγκατέστησα το scapy που είναι ένα πρόγραμμα χειρισμού πακέτων και βιβλιοθήκης με βάση την Python.



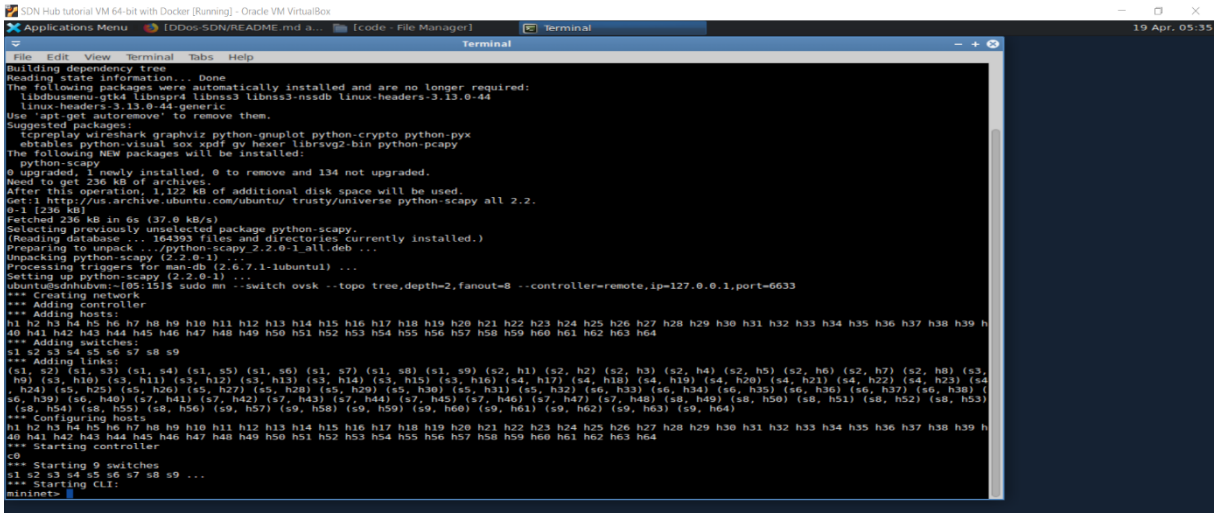
Εικ. 4.4.2 ΕΓΚΑΤΑΣΤΑΣΗ PYTHON SCAPY

Επιπρόσθετα έκανα κάποιες διαμορφώσεις (configurations) σε κάποια αρχεία που χρειαζόντουσαν για να δουλέψει όπως πρέπει με τα python scripts και έτσι παραθέτω το πιο κάτω το στιγμιότυπο οθόνης.

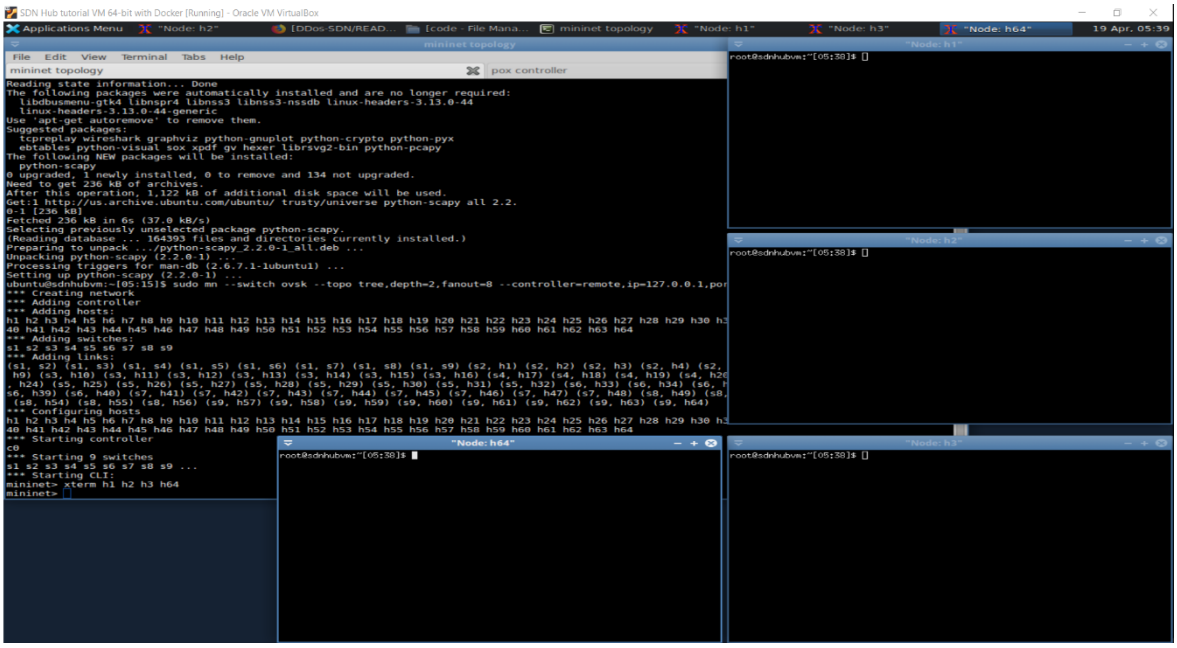


Εικ. 4.4.3 PYTHON SCAPY CONFIGURATIONS

Επιπλέον έπρεπε να δημιουργήσω μια mininet τοπολογία **sudo mn --switch ovsk --topo tree,depth=2,fanout=8 --controller=remote,ip=127.0.0.1,port=6633** και με την βοήθεια του POX controller από τις ακόλουθες εντολές, **cd pox** και **python ./pox.py forwarding.l3_edited** να τρέξω τέσσερα τερματικά με την ακόλουθη γραμμή εντολής **xterm h1 h2 h3 h64** όπου αυτό φαίνεται στα επόμενα δυο στιγμιότυπα οθόνης.

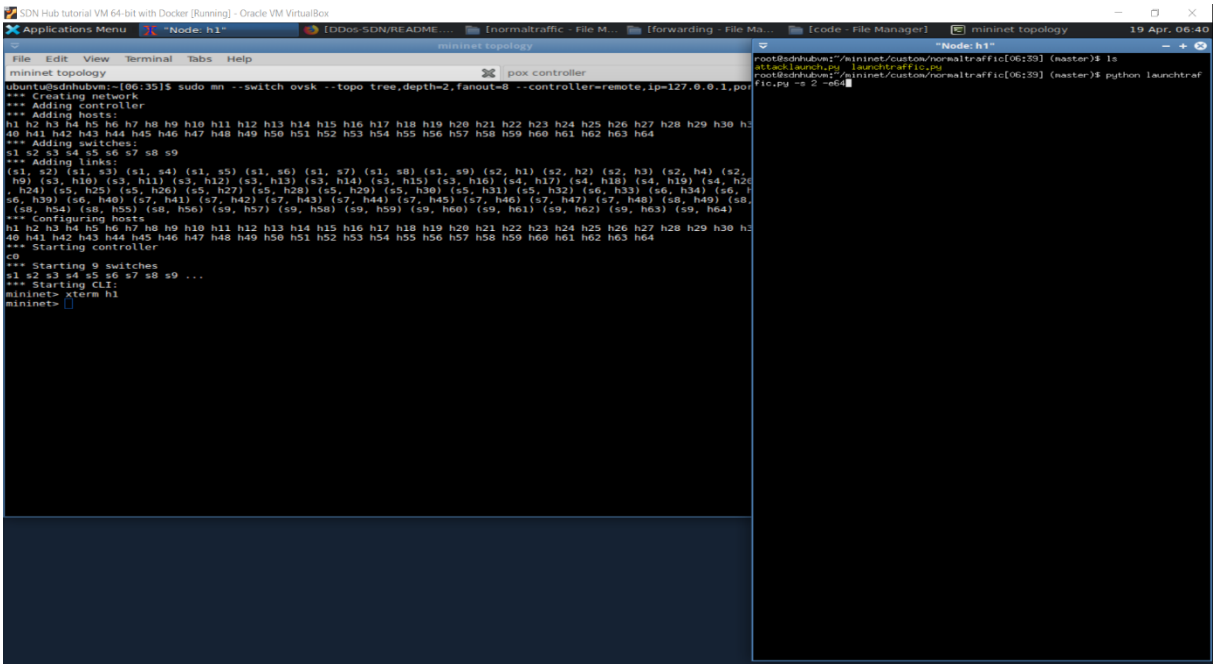


Εικ. 4.4.4 MININET TOPOLOGY with 9 switches, 64 hosts and 1 controller

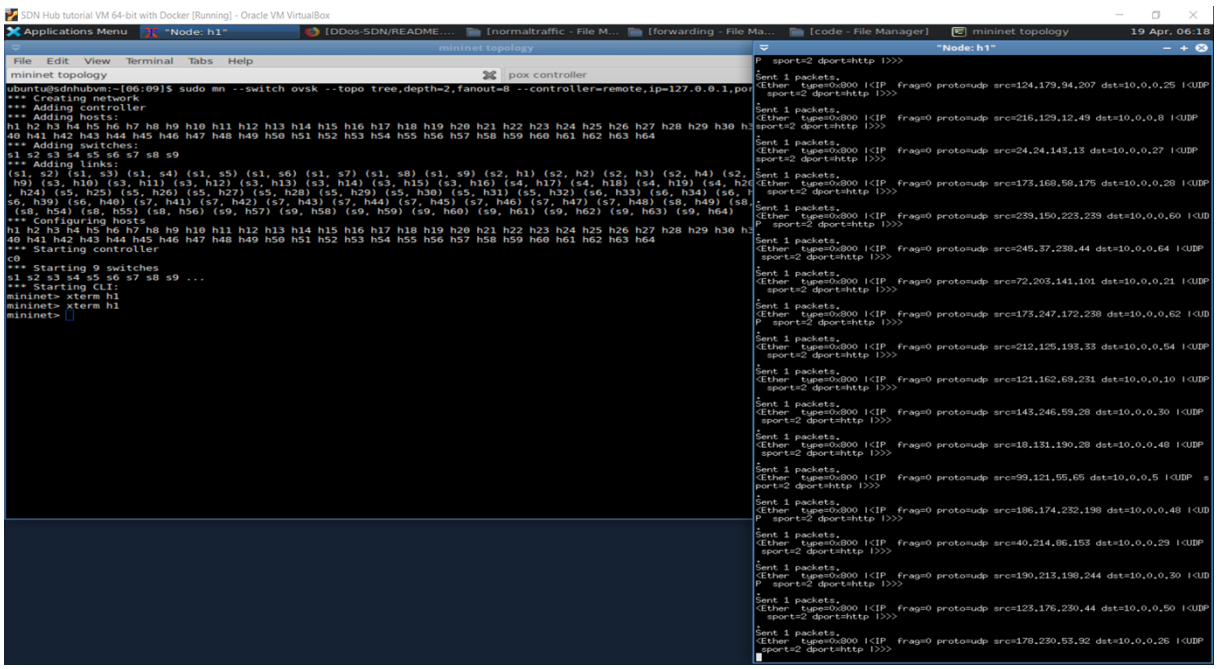


Εικ. 4.4.5 xterm h1 h2 h3 h64

Επιπλέον, με την γραμμή εντολής **python launchtraffic.py -s 2 -e 64** (s = first value e = last value) που είναι το νούμερο του οικοδεσπότη – host που θέλουμε να τους στείλουμε πακέτα και αυτό γίνεται από το τερματικό του οικοδεσπότη (host) όπου αυτό φαίνεται στα επόμενα δυο στιγμιότυπα οθόνης.

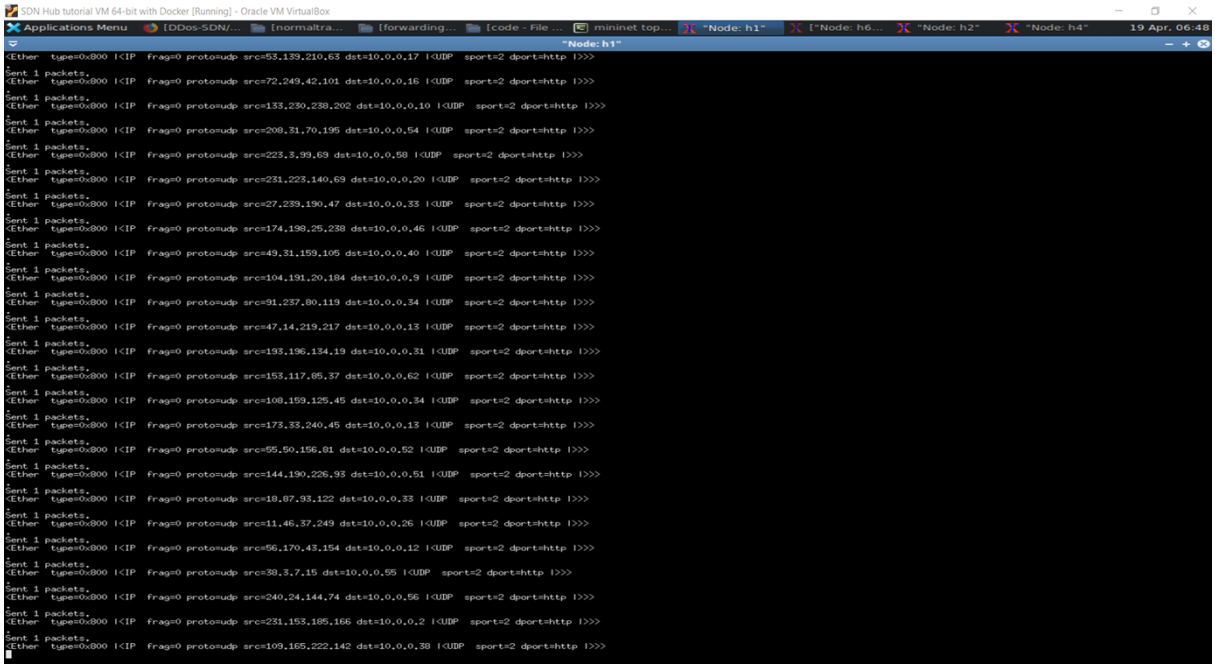


Εικ. 4.4.6 launch attack python script

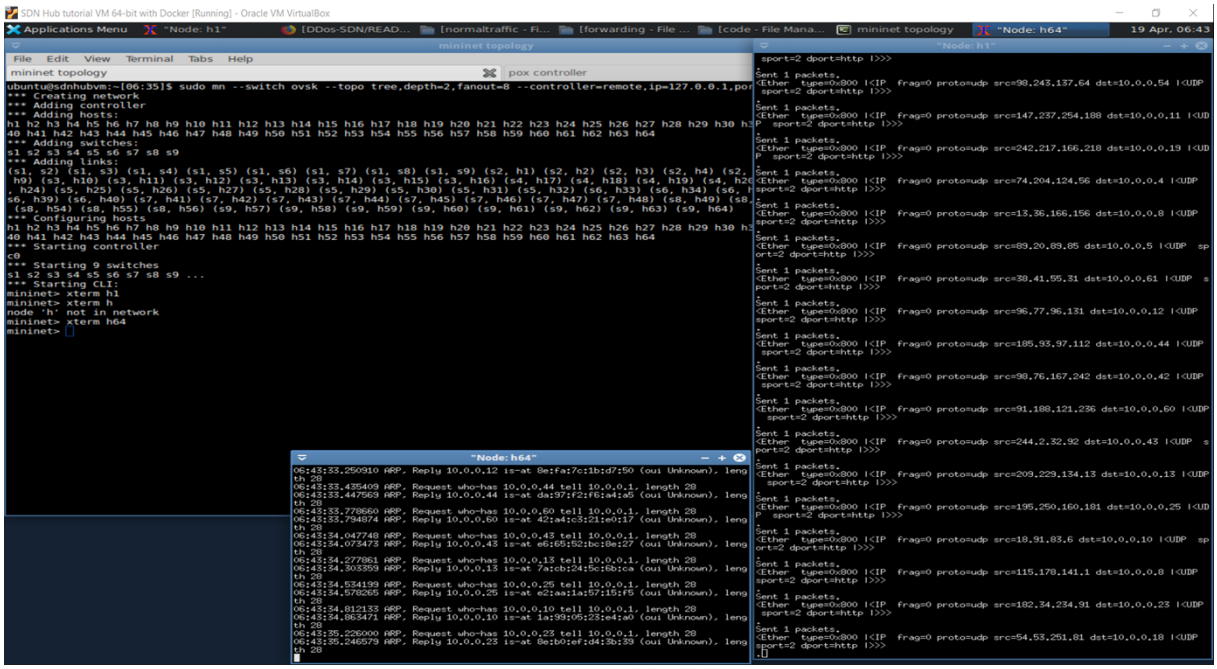


Εικ. 4.4.7 launch attack python script (cont.)

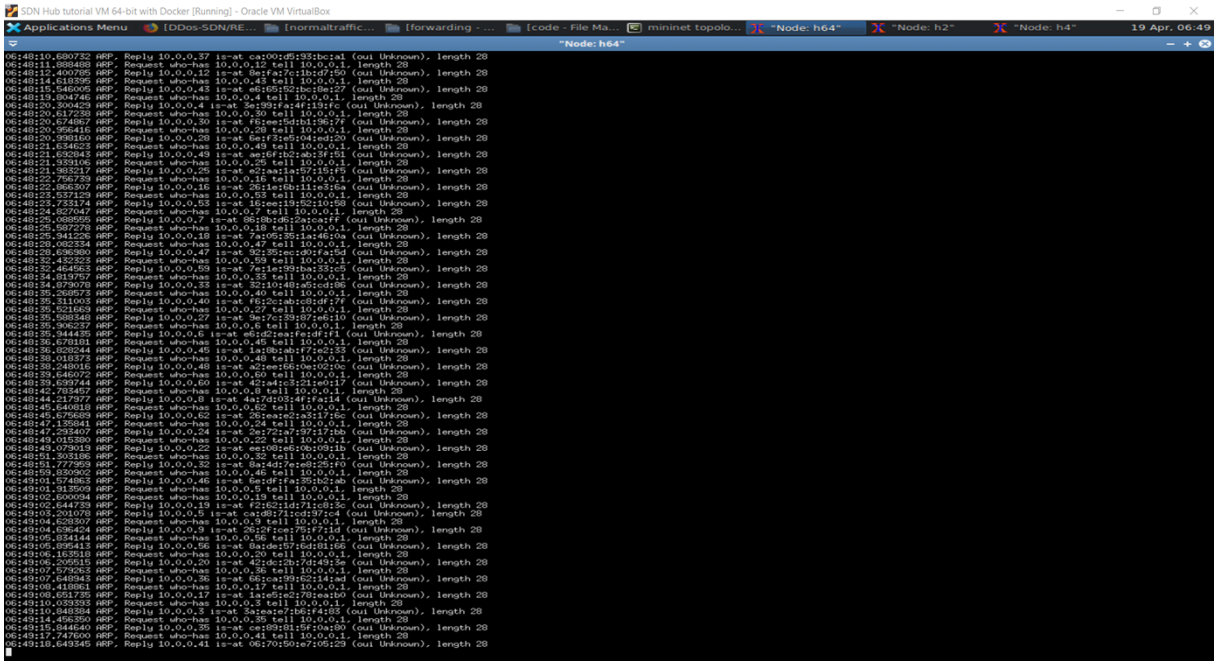
Ακολούθως, χρησιμοποιήθηκε η εντολή **tcpdump** που είναι ένα βοηθητικό πρόγραμμα γραμμής εντολών που μπορείτε να χρησιμοποιήσετε για να καταγράψετε και να ελέγξετε την κυκλοφορία του δικτύου από και προς το σύστημά σας, όπου αυτό φαίνεται στα επόμενα τρία στιγμιότυπα οθόνης.



Εικ. 4.4.8 tcpdump καταγραφή και έλεγχος κυκλοφορίας

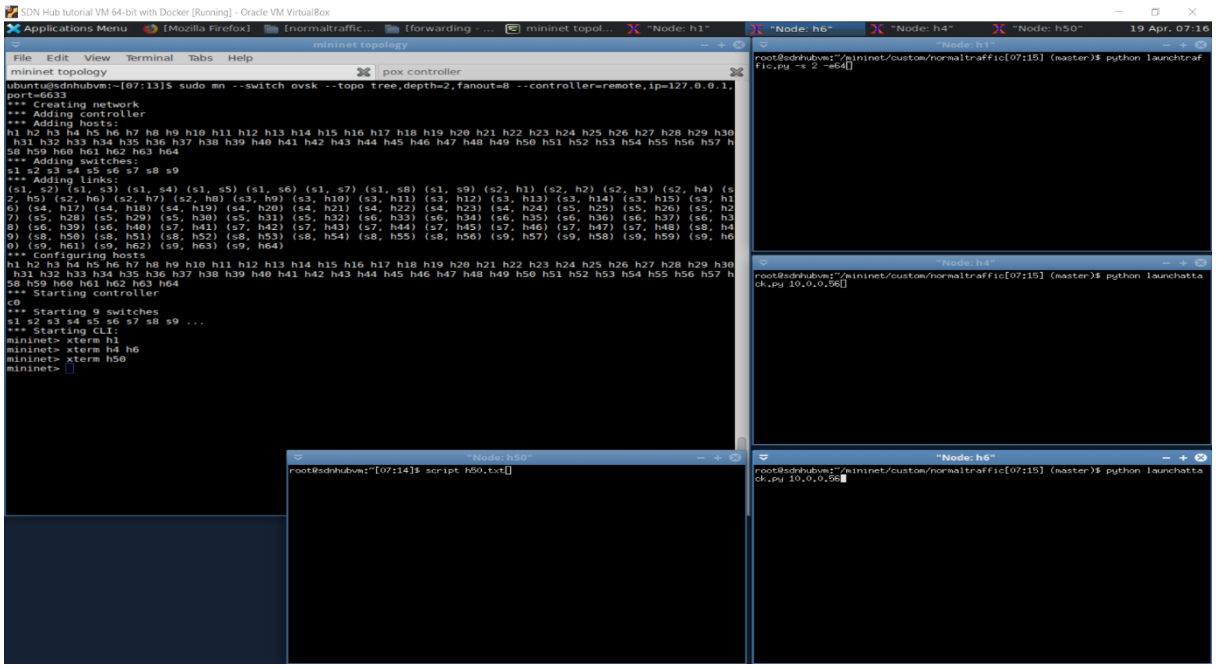


Εικ. 4.4.9 tcpdump καταγραφή και έλεγχος κυκλοφορίας (cont.)

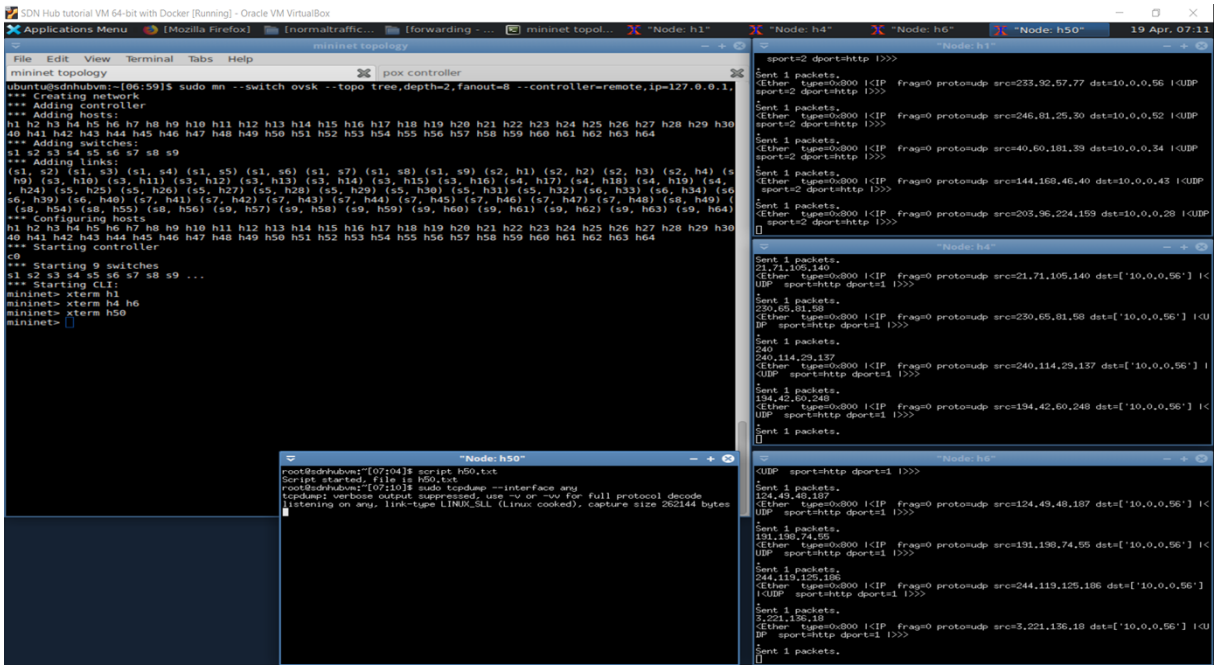


Εικ. 4.4.10 tcpdump καταγραφή και έλεγχος κυκλοφορίας (cont.)

Επιπλέον, με την γραμμή εντολής **python launchattack.py 10.0.0.56** όπου 10.0.0.56 είναι το νούμερο του οικοδεσπότη – host του οποίου θέλουμε να κάνουμε την επίθεση και αυτό γίνεται από το τερματικό του οικοδεσπότη (host) όπου αυτό φαίνεται στα επόμενα δυο στιγμιότυπα οθόνης.



Εικ. 4.4.11 python launchattack.py 10.0.0.56



Εικ. 4.4.12 python launchattack.py 10.0.0.56 (cont.)

4.5. Συμπεράσματα

Σε αυτή την υπό-ενότητα, θα δούμε τα συμπεράσματα όλων των πιο πάνω πειραματισμών, αποτελεσμάτων αλλά και την συμπεριφορά που αφορά τα παραδοσιακά αλλά και τα καθορισμένα από λογισμικό δίκτυα.

Όταν το παραδοσιακό δίκτυο δέχθηκε επίθεση διανεμημένης άρνησης υπηρεσίας (DDoS Attack) δεν στάθηκε δυνατή η ρύθμιση το επιπλέον ροών πακέτων με αντίκτυπο να μειωθεί το εύρος ζώνης (bandwidth) με αποτέλεσμα να χάνονται τα πακέτα, και αυτό οφείλεται στην απουσία του λογισμικού, όπου σε αντίθεση με τα καθορισμένα από λογισμικό δίκτυα ο ελεγκτής (controller) παρόλο που υπήρξε η ίδια επίθεση (DDoS Attack) δεν υπήρξε καθόλου απώλεια πακέτων αλλά ούτε και εύρος ζώνης (bandwidth).

Άρα ουσιαστικά η ύπαρξη του ελεγκτή (controller) μέσα στα καθορισμένα από λογισμικό δίκτυα (SDN) δίνει μια επιπλέον ανθεκτικότητα στο δίκτυο, η οποία δεν υπάρχει στα παραδοσιακά δίκτυα, πράγμα που φαίνεται από παρόμοιες μελέτες και άρθρα σε επιστημονικές μελέτες όπως [68], [69], [70] όπου δείχνουν την ευάλωτη πλευρά του κεντρικού ελεγκτή σε ορισμένα είδη επιθέσεων αλλά οι περισσότεροι εστιάζουν στο ότι κερδίζει το πόσο βελτιωμένη είναι η ασφάλεια των SDN σε αντίθεση με αυτήν των παραδοσιακών δικτυών.

4.6 SDN vs Traditional Networks

Σε αυτή την υπό-ενότητα, θα δούμε πως τα δίκτυα που καθορίζονται από λογισμικό (SDN - Software Defined Networks) είναι διαφορετικά από τα παραδοσιακά δίκτυα (traditional networks). [7]

Τα παραδοσιακά δίκτυα είναι ένα συμβατικό περιβάλλον δικτύωσης, όπου οι συσκευές δικτύου διαθέτουν λειτουργίες επιπέδου ελέγχου που χρησιμοποιούνται για "λήψη αποφάσεων" που είναι βασισμένες σε προγραμματιζόμενους κανόνες για το πού στέλνονται πακέτα που εισέρχονται στη συσκευή δικτύου όπως π.χ. ένας διακόπτης. Στο περιβάλλον αυτό του παραδοσιακού δικτύου, κάθε φορά που επεκτείνετε με νέες εφαρμογές ή αλλάζουν οι κανόνες για τον τρόπο ροής της κίνησης στο δίκτυο τότε πρέπει να ενημερώνονται και να γίνονται αλλαγές απευθείας στις φυσικές συσκευές.

Άρα αντιλαμβάνεστε το πόσο δύσκολο είναι αυτό το σενάριο για ομάδες πληροφορικής με δεκάδες ή εκατοντάδες συσκευές δικτύου που έχουν εξαπλωθεί σε καταστήματα ή κέντρα δεδομένων, που αυτό ίσως συμβαίνει σε εταιρίες που παρέμειναν στα παραδοσιακά δίκτυα. Από αυτό σίγουρα οι οργανισμοί περιορίζονται από τους περιορισμούς των παραδοσιακών περιβαλλοντικών δικτύων επιχειρήσεων και έτσι γίνεται μια προσπάθεια να ικανοποιήσουν τις απαιτήσεις των σημερινών ψηφιακών επιχειρήσεων.

Κάποιοι περιορισμοί του παραδοσιακού δικτύου:

1. Στατικό Περιβάλλον (static environment) όπου η εφαρμογή πολιτικών σε όλο το δίκτυο είναι χρονοβόρα και πολύπλοκη.
2. Αδυναμία Κλιμάκωσης (inability to scale) όπου η δυνατότητα κλιμάκωσης μειώνεται επειδή τα δίκτυα είναι υπερβολικά φορτωμένα και δεν μπορούν γρήγορα να προγραμματιστούν για να μπορέσουν εξυπηρετήσουν.
3. Κλείδωμα Προμηθευτή (vendor lockin) όπου οι εταιρείες είναι συχνά κλειδωμένες με έναν μόνο προμηθευτή, χωρίς τυπικά πρωτόκολλα για τη διαμόρφωση εξοπλισμού μεταξύ προμηθευτών δικτύου.

Καθώς ο κόσμος επικεντρώνεται περισσότερο στην χρήση δίκτυών, οι οργανισμοί πρέπει να δημιουργήσουν νέους τρόπους για να ενσωματώσουν μεγαλύτερη ευελιξία στην αρχιτεκτονική του δικτύου τους.

Όπως αναφέρει το Open Network Foundation [71] ένα δίκτυο καθορισμένο από λογισμικό (SDN) επιτυγχάνει κάτι τέτοιο μέσω της τυποποίησης και αφαίρεσης των λειτουργιών δικτύου, όπου το SDN αποσυνδέει τις λειτουργίες προώθησης δικτύου και ελέγχου ούτως ώστε να επιτρέπεται καλύτερος έλεγχος του δικτύου που να ‘ναι προγραμματιζόμενος.

Αφαιρώντας τις λειτουργίες δικτύου, συμπεριλαμβανομένων των λειτουργιών του επιπέδου ελέγχου και τοποθετώντας τις σε έναν ελεγκτή SDN που εκτελεί λογισμικό SDN, αποκτείτε καλύτερος κεντρικός έλεγχος.

Χρησιμοποιώντας τον ελεγκτή (controller) και το λογισμικό του δικτύου, οι διάφορες ομάδες ενός οργανισμού μπορούν να επικοινωνούν με τις φυσικές ή εικονικές συσκευές

δικτύου χρησιμοποιώντας το πρωτόκολλο δικτύωσης OpenFlow, μέσα από την διεπαφή προγράμματος εφαρμογών (API) όπου οι ομάδες αυτές μπορούν να χρησιμοποιήσουν έναν ελεγκτή (controller) για να πραγματοποιήσουν αλλαγές στο δίκτυο που ελέγχουν πολλές συσκευές χωρίς να χρειάζεται να μάθουν αρκετές εντολές.

Επειδή τα δίκτυα που καθορίζονται από λογισμικό (SDN - Software Defined Networks) διαχειρίζονται κεντρικά και διαμορφώνονται χρησιμοποιώντας ανοιχτό λογισμικό αυτό τα κάνει να είναι προγραμματιζόμενα και να μπορούν να αλλάξουν γρήγορα με βάση τις νέες επιχειρηματικές απαιτήσεις.

Το λογισμικό που καθορίζει αυτά τα δίκτυα (SDN) καθιστά τα σημερινά περιβάλλοντα δικτύου πολύ πιο δυναμικά (dynamic), διαχειρίσιμα (manageable), οικονομικά αποδοτικά (cost effective) και προσαρμόσιμα (adaptable).

4.7 SDN Ευλογία ή Κατάρα

Όπως αναφέρουν στο άρθρο τους οι Baier, Abt και Schehlmann [72] "Από άποψη ασφαλείας, το SDN έχει δύο πλευρές. Πρώτον, επιτρέπει τις λειτουργίες ασφάλειας δικτύου από το σχεδιασμό του, επειδή οι ροές κυκλοφορίας μπορούν να ανακατευθυνθούν ή να φιλτραριστούν βάσει περιεχομένου πακέτου ή κατάστασης επιπέδου εφαρμογής - λειτουργικότητα, η οποία μέχρι σήμερα απαιτεί πρόσθετες συσκευές ασφάλειας δικτύου, όπως τείχη προστασίας, συστήματα ανίχνευσης εισβολών ή φίλτρα ανεπιθύμητου περιεχομένου σε συμβατικά δίκτυα. Από την άλλη πλευρά, λόγω του φυσικού διαχωρισμού των επιπέδων, το SDN πιθανώς προσφέρει επιπλέον διανύσματα επίθεσης σε σύγκριση με τις παραδοσιακές αρχιτεκτονικές δικτύου, τα οποία ενδέχεται να επηρεάσουν σοβαρά τη συνολική διαθεσιμότητα του δικτύου, καθώς και την εμπιστευτικότητα, την αυθεντικότητα, την ακεραιότητα και τη συνέπεια της κίνησης του δικτύου και των δεδομένων ελέγχου."

Άρα ουσιαστικά τα δίκτυα που καθορίζονται από λογισμικό (SDN - Software Defined Networks) είναι μια ευλογιά σε γενικότερο επίπεδο με βάση τα όλα όσα λέγονται στην επιστημονική κοινότητα, και αυτό φαίνεται και στην επισκόπηση [73] των Sezer, Natarajan και Scott-Hayward.

4.8 DDoS Attack in SDN

Όπως προ-αναφέραμε σε προηγουμένη ενότητα (4.4) για την επίθεση διανεμημένης άρνησης υπηρεσίας (DDoS – Distributed Denial of Service) που είναι μια συμβατική επίθεση σε ένα περιβάλλον που το δίκτυο που καθορίζεται από λογισμικό (SDN) όπου οι κακόβουλοι χρήστες ή εισβολείς, χρησιμοποιούν πάρα πολλές καταχωρήσεις ροής (data flows) οι οποίες στοχεύουν σε κάποιον ελεγκτή κι πολλές φορές στον κεντρικό ελεγκτή ενός δικτύου ως γνωστό κίνδυνο να τον βγάλουν εκτός λειτουργίας.

Εν συνέχεια, οι ελεγκτές (controllers) υπολογίζουν τους διαθέσιμους πόρους για την ικανοποίηση των αιτημάτων αυτών, όπου σε κάποιες περιπτώσεις, ο ελεγκτής μπορεί να καταστεί αδύνατον να αντιμετωπίσει τυχόν νόμιμα αιτήματα που λαμβάνει.

Στόχος του εισβολέα είναι να κάνει το σύνολο του δικτύου ή όσο μεγαλύτερο μέρος του δικτύου σε μεγάλο βαθμό αχρησιμοποίητο, ενώ οι διαδρομές δεδομένων (data paths) που βρίσκονται είδη μέσα στο δίκτυο είναι πιθανόν να είναι σε θέση να λειτουργούν προσωρινά με έναν ελεγμένο χειριστή, όταν έχει περάσει ένα μεγάλο χρονικό όριο των κανόνων (που αποθηκεύεται στον πίνακα τους), θα πρέπει να αιτηθούν ξανά στον ελεγκτή, ο οποίος δεν θα είναι σε θέση να αντιμετωπίσει τα αιτήματα.

Στην τελική, αν η επίθεση πνίξει με πολλαπλά αιτήματα για αρκετό χρονικό διάστημα το δίκτυο τότε το δίκτυο είναι πιθανόν να τεθεί εκτός λειτουργίας. [74]

4.9 Υπάρχουσες λύσεις για την ασφάλεια των SDN

Σε αυτή την υπό-ενότητα, θα δούμε τις υπάρχουσες λύσεις για την ασφάλεια των δίκτυών που καθορίζονται από λογισμικό (SDN - Software Defined Networks) μιας και κατά την πάροδο του χρόνου από την ημέρα που ξεκίνησαν τα SDN έχουν προταθεί αρκετές γλώσσες προγραμματισμού οι οποίες μπορούν να ενισχύσουν την δυνατότητα απεριόριστης δημιουργικότητας λειτουργιών μιας και το συγκεκριμένο είδος το επιτρέπει.

Αρχικά θα δούμε, το BigTap [75] που είναι “μια προηγμένη εφαρμογή παρακολούθησης δικτύου που αξιοποιεί ένα υψηλής απόδοσης υφασμάτινο διακόπτη Ethernet με δυνατότητα OpenFlow για να παρέχει το πιο επεκτάσιμο και ευέλικτο δίκτυο παρακολούθησης για να αξιοποιεί την κυκλοφορία παντού στο δίκτυό σας και να το

παρέχει με πολιτική σε οποιαδήποτε από τις επιδόσεις ή την παρακολούθηση της απόδοσής σας εργαλεία."

Εν συνέχεια, τα δυο παραδείγματα τα οποία αποτελούν αξιόπιστα παραδείγματα του απλοποιημένου επιπέδου δεδομένων και της τροποποίησης του, εκ τον οποίον το πρώτο είναι το NetFPGA [76] "είναι μια προσπάθεια ανάπτυξης υλικού και λογισμικού ανοιχτού κώδικα για ταχεία δημιουργία πρωτοτύπων συσκευών δικτύου υπολογιστών." Και το δεύτερο, το DevoFlow [77] "χρησιμοποιεί 10-53 φορές λιγότερες καταχωρήσεις πίνακα ροής σε ένα μέσο διακόπτη και χρησιμοποιεί 10-42 φορές λιγότερα μηνύματα ελέγχου."

Επιπρόσθετα τα δίκτυα SDN μπορούν να διευκολύνουν τη συλλογή πληροφοριών χρήσης δικτύου (ποιος χρησιμοποιά και τι) με την χρήση εργαλείων παρακολούθησης δικτύου (network monitoring tools), οι οποίες θα μπορούσαν να υποστηρίξουν βελτιωμένο σχεδιασμό αλγορίθμων που χρησιμοποιείται για τον εντοπισμό επιθέσεων.

Όπου το FlowSense [78] είναι "μια μέθοδος για τον προσδιορισμό της κατάστασης ενός ολόκληρου δικτύου, συμπεριλαμβανομένης της κατάστασης των εφαρμογών και της υποδομής περιλαμβάνει τη λήψη μηνυμάτων ελέγχου δικτύου σε ένα δίκτυο OpenFlow."

Επιπλέον το FlowChecker [79] το οποίο προσφέρει μια διάφορα εργαλεία επαλήθευσης βάση ιδιοτήτων που εντοπίζουν διαφορετικές σφαλλόμενες ρυθμίσεις μέσα στο δίκτυο.

Το NICE [80] που είναι "ένα εργαλείο για τη δοκιμή της εφαρμογής ελεγκτή OpenFlow για την πλατφόρμα ελεγκτή NOX μέσω ενός συνδυασμού ελέγχου μοντέλου και συμβολικής εκτέλεσης." άλλο ένα εργαλείο εύρεσης σφαλμάτων.

Επιπρόσθετα με τα πιο πάνω, που χρησιμοποιούνται πριν την έναρξη του δικτύου, το VeriFlow [81] ελέγχει σε πραγματικό χρόνο την ακρίβεια του δικτύου καθώς αυτό προχωρά προοδευτικά.

Και τέλος, χρειάζεται μια δυναμική ανταπόκριση σε απειλές όπου μέσα στο δίκτυο μπορούμε να ανακατευθύνουμε την κίνηση των κακόβουλων επιθέσεων με το να

επανα-προγραμματίσουμε τους διακόπτες μέσω του ελεγκτή (controller) κάτι που γίνεται στο FRESCO [82].

4.10 Υπάρχουσες λύσεις για την ασφάλεια των Παραδοσιακών Δικτυών

Σε αυτή την υπό-ενότητα, θα δούμε τις υπάρχουσες λύσεις για την ασφάλεια των Παραδοσιακών Δίκτυών (Traditional Networks) όσο αφορά τον τρόπο με τον οποίο θα μπορούσαμε να εντοπίσουμε μια απειλή ή κάποιο ιό που εγκαταστάθηκε σε κάποιο οικοδεσπότη (host) ή και ακόμα σε servers.

Θα πρέπει να κάνουμε μια καλύτερη έρευνα για το όλο θέμα ούτος ώστε να δημιουργήσουμε μια γκάμα ή ένα οπλοστάσιο ή ακόμη να βάλουμε ξεχωριστά μηχανήματα με διαφορά λογισμικά ούτος ώστε να είμαστε σε θέση και γενικότερα να είναι το δίκτυο πιο ικανό να ανταπεξέλθει στην αντιμετώπιση μιας τέτοιας απειλής ή ιού.

Δηλαδή, από μεριάς του τοίχου προστασίας (firewalls) θα μπορούσαμε να χρησιμοποιήσουμε το **smoothwall** [83] το οποίο είναι ένα linux distribution σχεδιασμένο να χρησιμοποιείτε σαν firewall, αλλά και επίσης μπορούμε να βρούμε και cloud servers οι οποίοι μπορούν να βοηθήσουν σε αυτή την δουλειά δηλαδή στον τομέα του portscanning και bruteforce attacks [84] το οποίο περιορίζει τα ICMP και δέχεται ο server μας μόνο από συγκεκριμένα IPs.

Επιπλέον μέσα στο οπλοστάσιο μας, θα ήταν καλό να έχουμε κάποια λογισμικά ανίχνευσης/πρόληψης εισβολής **(IDS/IPS - intrusion detection software/intrusion prevention software)** όπου συγκρίνει τα πακέτα δικτύου σε μια βάση δεδομένων cyberthreat που περιέχει γνωστές υπογραφές cyberattacks - και επισημαίνει τα πακέτα που ταιριάζουν.

Μερικά **IDS/IPS** που γνωρίζω για περιβάλλον Windows/Linux εκτός από το **Splunk** είναι και το **Sagan**, αλλά και **Open WIPS-NG**, **FAIL2BAN** και **Zeek** για περιβάλλον **Linux** και επιπλέον, το **WINPATROL** για περιβάλλον **Windows.**

Επιπρόσθετα λογισμικά που μπορούμε να λάβουμε υπόψη μας είναι, **COMODO Firewall Pro** [85], το **AlienVault** (AT&T) [86] και ακόμα το **SNORT** [87].

Κεφάλαιο 5
Επίλογος

5.1 Περίληψη

Αρχικός σκοπός της μεταπτυχιακής διατριβής ήταν να υλοποιηθεί ένα δίκτυο καθορισμένο από λογισμικό (SDN - software defined networking) και να μελετηθεί κατά πόσο είναι αξιόπιστο σε σύγκριση με τα παραδοσιακά δίκτυα (traditional ethernet networks) σε διανεμημένη επίθεση άρνησης υπηρεσίας (DDoS Attack) όπου είναι η επίθεση η οποία επέλεξαν μιας και είναι η πιο γνωστή σε χρήση από όλες.

Εν συνέχεια, για το κομμάτι τις υλοποίησης χρησιμοποιήθηκε ένα προ-δημιουργημένο εικονικό μηχάνημα [65] με συγκεκριμένες διαμορφώσεις (configurations) που προτείνονταν έτσι ώστε να τρέχει σε κανονικούς ρυθμούς όπου με την βοήθεια ενός επόπτη φιλοξενίας για εικονικοποίηση (Oracle VM VirtualBox Manager) έγινε εφικτή η εγκατάσταση του.

Επιπρόσθετα, η προτεινομένη μεθοδολογία που έχει χρησιμοποιηθεί είναι η μεθοδολογία του Pietro Manzoni [67] όπου φαίνεται μέσα από διάφορα στιγμιότυπα οθόνης και επιπλέον με την βοήθεια του εξομοιωτή/προσομοιωτή δικτύου Mininet (Miniedit) και σε κάποιες φάσεις με τον αναλυτή πακέτων δικτύου Wireshark αποδείχτηκε κατά πόσο μπορεί να προσδιοριστεί αν μπορεί να προσφέρει την ζητούμενη ασφάλεια ή οποιεσδήποτε άλλες βελτιώσεις στην ασφάλεια του δικτύου.

Επιπλέον, έχει χρησιμοποιηθεί το Ευκίνητο Μοντέλο Ανάπτυξης Λογισμικού (Agile SDLC Model) και η μεθοδολογία του ούτος ώστε να παραχθούν λειτουργικά αποτελέσματα.

Εν συνέχεια, μέσα από την βιβλιογραφική ανασκόπηση μελετήθηκαν οι υπάρχουσες εφαρμογές δικτύων καθορισμένα από λογισμικό (SDN - software defined networking)

που σχεδιάστηκαν για παροχή ενισχυμένης ασφάλειας, και στην τελική, υλοποιήθηκε μια επίθεση στο SDN αλλά και στο παραδοσιακό δίκτυο.

5.2 Περιθώρια Επιπλέον Έρευνας

Σκεπτόμενος πάντα όσο πιο θετικά γίνεται στην ζωή κι με τις σύγχρονες κατάστασης που βιώνουμε όλοι μαζί τον τελευταίο καιρό λόγο τις πανδημίας, ευελπιστούμε για ότι καλύτερο για όλους μας, έτσι και εδώ περιθώρια επιπλέον ερευνάς πάντα υπάρχουν στην επιστημονική κοινότητα οπότε ρίχνοντας μια ματιά στις επιλογές που έχουν οι εταιρείες κι γενικότερα οι προμηθευτές δικτυών αυτή τη στιγμή για να προμηθευτούν "προγραμματιζόμενες" τεχνολογίες δικτύου, θα μπορούσαμε να πούμε ότι το μέλλον της δικτύωσης που καθορίζεται από λογισμικό (software defined networking) είναι ήδη εδώ.

Το πιο συναρπαστικό πράγμα ενός δικτύου καθορισμένο από λογισμικό (SDN - software defined networking) είναι η ικανότητά του να απλοποιεί το σχεδιασμό του δικτύου, συνεπώς, εάν μέσα σε ένα τέτοιο δίκτυο παραγγείλετε κάποια πράγματα μέσω μιας συνδρομής (ως χρήστης βέβαια) ενδέχεται να μπορείτε να τα δείτε ως πιο διαφανείς ενότητες - με καλύτερο οπτικό τρόπο – από τον πίνακα ελέγχου όπως είδαμε μέσα από τη βιβλιογραφική ανασκόπηση. (visual dashboards)

Από την μια, το λογισμικό ως υπηρεσία (software as a service) προφυλάσσει τον χρήστη από το να προσπαθεί να καταλάβει τους πραγματικούς μηχανισμούς κι τι υπάρχει πίσω από αυτούς για να μπορούν να χρησιμοποιήσουν τις τεχνολογίες χωρίς να κολλήσουν κάπου. Αλλά από την άλλη, γίνεται μια προσπάθεια να εμποδιστεί η πρόσβαση του χρήστη να μπορεί να αλλάξει ή να ελέγξει την τεχνολογία που χρησιμοποιεί, γι' αυτό το λόγο υπάρχει κι η ορολογία όπως το "κλείδωμα προμηθευτή" (vendor lock-in) όσο αφορά μια συμφωνία σε επίπεδο υπηρεσίας SDN.

Οδεύοντας στον 21ο αιώνα, αυτό εισακούει το πως κατανοούμαι διάφορες τεχνολογίες όπως της δικτύωσης που καθορίζεται από λογισμικό (software defined networking) αλλά επίσης το πως τις εφαρμόζουμε κι διαχειριζόμαστε αφού είναι ήδη εδώ και είναι δίκτυα επομένης γενιάς.

Παράρτημα Α
Αποτελέσματα

Α.1 Κώδικας Python

Οποιοσδήποτε κώδικας έχει χρησιμοποιηθεί στην μεταπτυχιακή διατριβή αυτή έχει κλωνοποιηθεί από την github αποθήκη (repository) με την γραμμή εντολής **git clone https://github.com/jagan103/DDos-SDN.git**

Βιβλιογραφία

[1] klipfolio.com, «What is a data dashboard? Definition, examples and strategic tips!,» klipfolio, 2021. [Ηλεκτρονικό]. Available: https://www.klipfolio.com/resources/articles/what-is-data-dashboard.

[2] E. Rondeau, «Traditional Network versus SDN,» ResearchGate, 2 May 2021. [Ηλεκτρονικό]. Available: https://www.researchgate.net/figure/Traditional-Network-versus-SDN_fig1_319876305.

[3] Cisco, «Software-defined networking (SDN) definition,» Cisco, 2000. [Ηλεκτρονικό]. Available: https://www.cisco.com/c/en/us/solutions/software-defined-networking/overview.html.

[4] SDxCentral, «Understanding the SDN Architecture - SDN Control Plane & SDN Data Plane,» SDxCentral, 13 March 2015. [Ηλεκτρονικό]. Available: https://www.sdxcentral.com/networking/sdn/definitions/inside-sdn-architecture/.

[5] V. Beal, «SDN Meaning - What is Software Defined Networking?,» webopedia, 24 May 2012. [Ηλεκτρονικό]. Available: https://www.webopedia.com/definitions/sdn/.

[6] H. Arora, «Software Defined Networking (SDN) explained for beginners,» HowtoForge, [Ηλεκτρονικό]. Available: https://www.howtoforge.com/tutorial/software-defined-networking-sdn-explained-for-beginners/.

[7] IBM, «SDN Versus Traditional Networking Explained,» IBM, 2020. [Ηλεκτρονικό]. Available: https://www.ibm.com/services/network/sdn-versus-traditional-networking.

[8] Computer History Museum, «Networking & The Web - Timeline of Computer History,» Computer History Museum, 20 March 2000. [Ηλεκτρονικό]. Available: https://www.computerhistory.org/timeline/networking-the-web/.

[9] Britannica - The Editors of Encyclopaedia, «Telex,» Britannica - The Editors of Encyclopaedia, [Ηλεκτρονικό]. Available: https://www.britannica.com/technology/telex.

[10] Engineering Channel, «Engineering Channel: Telex History,» Engineering Channel, 19 September 1980. [Ηλεκτρονικό]. Available: https://engineers-channel.blogspot.com/p/telex-history.html.

[11] L. Kleinrock, «Information flow in large communication nets,» RLE Quarterly Progress Report, 1961. [Ηλεκτρονικό]. Available: https://scholar.google.com/scholar?hl=en&as_sdt=0,5&cluster=16417442163223334 93.

[12] elevarbetensys, «History of the internet – Benji's Site,» elevarbetensys, [Ηλεκτρονικό]. Available: https://elevarbetensys.se/IN17/LB17/history.php.

[13] J. McCallion, «Inventor of email, Ray Tomlinson, dies aged 74Inventor of email, Ray Tomlinson, dies aged 74,» IT PRO, 7 March 2016. [Ηλεκτρονικό]. Available: https://www.itpro.co.uk/email-clients/26170/inventor-of-email-ray-tomlinson-dies-aged-74.

[14] S. Coty, «Where is IPv1, 2, 3,and 5?,» Alert Logic, 7 August 2015. [Ηλεκτρονικό]. Available: https://www.alertlogic.com/blog/where-is-ipv1-2-3-and-5/.

[15] IoT Security News, «Treck TCP/IP Stack (Update A),» IoT Security News, 30 June 2020. [Ηλεκτρονικό]. Available: https://iotsecuritynews.com/treck-tcp-ip-stack-update-a/.

[16] B. Salisbury, «https://networkstatic.net/cisco-onepk-about-time/,» Cisco onePK Announcement, big Surprise or About Time?, 15 June 2012. [Ηλεκτρονικό]. Available: https://networkstatic.net/cisco-onepk-about-time/.

[17] Open Networking Summit, «Open Networking Summit - David Ward,» Open Networking Summit, 28 October 2011. [Ηλεκτρονικό]. Available: https://www.youtube.com/watch?v=6qMgurC0juo.

[18] CLOUDFLARE, «What Is BGP? - BGP Routing Explained,» CLOUDFLARE, 2021. [Ηλεκτρονικό]. Available: https://www.cloudflare.com/learning/security/glossary/what-is-bgp/.

[19] CISCO, «Border Gateway Protocol (BGP),» CISCO, 28 April 2020. [Ηλεκτρονικό]. Available: https://www.cisco.com/c/en/us/products/ios-nx-os-software/border-gateway-protocol-bgp/index.html.

[20] P. Pedamkar, «What is OSPF? - Implementation And Application of OSPF,» EDUCBA, 18 June 2019. [Ηλεκτρονικό]. Available: https://www.educba.com/what-is-ospf/.

[21] M. K. Singh, «Difference between OSPF and BGP,» geeksforgeeks.org, 27 May 2019. [Ηλεκτρονικό]. Available: https://www.geeksforgeeks.org/difference-between-ospf-and-bgp/.

[22] A. S. d. S. J. A. W. P. S. L. Z. G. a. A. S.-F. Eduardo Germano da Silva, «A One-Class NIDS for SDN-Based SCADA Systems,» IEEE Xplore, 10 June 2016. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/abstract/document/7552026.

[23] A. Devasia, «An Introduction to Supervisory Control and Data Acquisition (SCADA),» Control, 29 May 2020. [Ηλεκτρονικό]. Available: https://control.com/technical-articles/an-introduction-to-supervisory-control-and-data-acquisition-scada/.

[24] Z. Latif, «A Comprehensive Survey of Interface Protocols for Software Defined Networks,» *Journal of Network and Computer Applications,* τόμ. 156, 2019.

[25] H. Peng, Y. Qiang και S. S. Xuemin, «SDN-Based Resource Management for Autonomous Vehicular Networks: A Multi-Access Edge Computing Approach,» *IEEE Wireless Communications,* τόμ. 26, αρ. 4, 2018.

[26] A. G. Rahim Masoudi, «Software defined networks: A survey,» *Journal of Network and Computer Applications,* τόμ. 9, αρ. 67, pp. 1-25, 2016.

[27] W. H. W. W. a. Y. L. Yili Gong, «A survey on software defined networking and its applications,» *Frontiers of Computer Science,* τόμ. 9, pp. 827-845, 2015.

[28] J. C. S. D. J. I. R. J. C. G. R. a. H. L. O. Jacob H. Cox, «Advancing Software-Defined Networks: A Survey,» *IEEE,* τόμ. 5, 2017.

[29] M. K. a. A. Durresi, «A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN),» *Computer Networks,* τόμ. 112, pp. 279-293, 2017.

[30] Z. G. P. Y. T. B. a. J. L. Tao Hu, «Multi-controller Based Software-Defined Networking: A Survey,» *IEEE,* 2018.

[31] A. M. A.-M. α. G. P. H. Hlabishi I. Kobo, «A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements,» IEEE Access, 2017. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7847327.

[32] A. M. S. K. N. a. I. A. A. Nur Faiza Ali, «A survey on software defined network approaches for achieving energy efficiency in wireless sensor network,» IEEE Conference on Wireless Sensors (ICWiSe), 2017. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/abstract/document/8267157.

[33] S. M. a. S. B. Athanasios Vasilakos, «Software-Defined Networking for Internet of Things: A Survey,» IEEE Internet of Things Journal, 29 August 2017. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/abstract/document/8017556.

[34] Y. L. a. M. Chen, «Software-Defined Network Function Virtualization: A Survey,» *IEEE Access,* τόμ. 3, 2015.

[35] SDxCentral, «What Is OpenFlow? - Definition and How it Relates to SDN,» SDxCentral, 26 August 2013. [Ηλεκτρονικό]. Available: https://www.sdxcentral.com/networking/sdn/definitions/what-is-openflow/.

[36] SDxCentral, «SDN articles, whitepapers, videos & more,» SDxCentral, 2021. [Ηλεκτρονικό]. Available: https://www.sdxcentral.com/networking/sdn/.

[37] V. Beal, «OpenFlow,» webopedia, 24 May 2012. [Ηλεκτρονικό]. Available: https://www.webopedia.com/definitions/openflow/.

[38] SDxCentral, «What Is Open SDN? A Definition,» SDxCentral, 19 November 2014. [Ηλεκτρονικό]. Available: https://www.sdxcentral.com/networking/sdn/definitions/open-sdn/.

[39] AlternativeTo, «NetSim Alternatives and Similar Software,» AlternativeTo, 24 November 2020. [Ηλεκτρονικό]. Available: https://alternativeto.net/software/netsim/.

[40] T. P. Morgan, «IBM and NEC tag team on OpenFlow networking,» The Register, 24 January 2012. [Ηλεκτρονικό]. Available: https://www.theregister.com/2012/01/24/ibm_nec_openflow_switches/.

[41] C. Lee, «Use IBM Security Network Protection in an OpenFlow-based Software-Defined Network,» IBM Developer Works archives, 17 May 2019. [Ηλεκτρονικό]. Available: https://www.ibm.com/developerworks/library/se-xgsopenflow/index.html.

[42] M. Y. S. N. a. I. A. Andrei Gurtov, «Security in Software Defined Networks: A Survey,» *IEEE Communications Surveys & Tutorials,* τόμ. 17, αρ. 4, 2015.

[43] S. Weisman, «What are Denial of Service (DoS) attacks? DoS attacks explained,» NortonLifeLock, 5 February 2020. [Ηλεκτρονικό]. Available: https://us.norton.com/internetsecurity-emerging-threats-dos-attacks-explained.html.

[44] K. Chivers, «What is a man-in-the-middle attack?,» NortonLifeLock, 26 March 2020. [Ηλεκτρονικό]. Available: https://us.norton.com/internetsecurity-wifi-what-is-a-man-in-the-middle-attack.html.

[45] U. T. a. V. Varadharajan, «Securing communication in multiple Autonomous System domains with Software Defined Networking,» 15th IFIP/IEEE International Symposium on Integrated Network and Service Management, 2017. [Ηλεκτρονικό]. Available: https://researchers.mq.edu.au/en/publications/securing-communication-in-multiple-autonomous-system-domains-with.

[46] M. Grant, «Understanding Software-as-a-Service (SaaS),» investopedia, 2 February 2020. [Ηλεκτρονικό]. Available: https://www.investopedia.com/terms/s/software-as-a-service-saas.asp.

[47] S. Vennam, «Cloud Computing,» IBM, 18 August 2020. [Ηλεκτρονικό]. Available: https://www.ibm.com/cloud/learn/cloud-computing.

[48] SDxCentral, «What is Networking as a Service or NaaS?,» SDxCentral, 7 August 2015. [Ηλεκτρονικό]. Available: https://www.sdxcentral.com/networking/virtualization/definitions/what-is-naas/.

[49] IBM Cloud Education, «IaaS vs. PaaS vs. SaaS,» IBM Cloud Education, 10 October 2018. [Ηλεκτρονικό]. Available: https://www.ibm.com/cloud/learn/iaas-paas-saas.

[50] M. M. P. P. a. A. L. W. Paolo Costa, «NaaS: Network-as-a-Service in the Cloud,» Microsoft Research, April 2012. [Ηλεκτρονικό]. Available: https://www.microsoft.com/en-us/research/publication/naas-network-as-a-service-in-the-cloud/.

[51] K. Sharief, «What is OpenDaylight? – Definition, Requirements, and More,» ComputerTechReviews, 2019. [Ηλεκτρονικό]. Available: https://www.computertechreviews.com/definition/opendaylight/.

[52] CIMI Corp., «Should SDN be About OpenDaylight and not OpenFlow?,» CIMI Corp., 9 September 2014. [Ηλεκτρονικό]. Available: https://blog.cimicorp.com/?p=1892.

[53] Mininet, «Mininet Overview,» Mininet, 2021. [Ηλεκτρονικό]. Available: http://mininet.org/overview/.

[54] J. S. a. N. G. K. Kaur, «Mininet as Software Defined Networking Testing Platform,» Semantic Scholar, 2014. [Ηλεκτρονικό]. Available: https://www.semanticscholar.org/paper/Mininet-as-Software-Defined-Networking-Testing-Kaur-Singh/b1c7f8ac477a5553303802bb7785dd3b53372057.

[55] Linux Foundation, «Open vSwitch,» Linux Foundation, 2009. [Ηλεκτρονικό]. Available: https://www.openvswitch.org/.

[56] R. B. a. M. Peters, «Dashboard design: Why design is important,» DMReview, 15 October 2004. [Ηλεκτρονικό]. Available: cs.furman.edu/~pbatchelor/csc105/articles/TUN_DM_ONLINE.pdf.

[57] A. Bryman, Social Research Methods, Oxford: Oxford University Press, 2016.

[58] M. Demchenko, «Software Development Life Cycle: A Guide to Phases and Models,» NCube, 19 August 2019. [Ηλεκτρονικό]. Available: https://ncube.com/blog/software-development-life-cycle-guide.

[59] M. Kramer, «Best Practices in Systems Development Lifecycle: An Analyses Based on the Waterfall Model,» SSRN.com, 8 March 2018. [Ηλεκτρονικό]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3131958.

[60] A. A. A. A. A. O. O. A. O. A. a. I. O.-A. Olatunji J. Okesola, «Software Requirement in Iterative SDLC Model,» SpringerLink, 09 August 2020. [Ηλεκτρονικό]. Available: https://link.springer.com/chapter/10.1007/978-3-030-51965-0_2.

[61] D. Gurendo, «Software Development Life Cycle (SDLC). Spiral Model,» XBsoftware.com, 26 October 2015. [Ηλεκτρονικό]. Available: https://xbsoftware.com/blog/software-development-life-cycle-spiral-model/.

[62] Rajikumar, «V Model in Software Development Life Cycle,» softwaretestingmaterial.com, 18 April 2016. [Ηλεκτρονικό]. Available: https://www.softwaretestingmaterial.com/v-model-in-sdlc/.

[63] V. Rastogi, «Software Development Life Cycle Models-,» *(IJCSIT) International Journal of Computer Science and Information Technologies,* τόμ. 6, αρ. 1, pp. 168-172, 2015.

[64] K. Rungta, «Incremental Model in SDLC: Use, Advantage & Disadvantage,» guru99.com, 13 April 2021. [Ηλεκτρονικό]. Available: https://www.guru99.com/what-is-incremental-model-in-sdlc-advantages-disadvantages.html.

[65] A. R. a. S. N. Srini Seetharaman, «All-in-one SDN App Development Starter VM,» SDN Hub, 30 August 2015. [Ηλεκτρονικό]. Available: http://sdnhub.org/tutorials/sdn-tutorial-vm/.

[66] XFCE Development Team, «XFCE Desktop Environment,» XFCE Development Team, 22 December 2020. [Ηλεκτρονικό]. Available: https://xfce.org/.

[67] P. Manzoni, «Mininet Lab 2: SDN with Mininet,» Pietro Manzoni, November 2020. [Ηλεκτρονικό]. Available: https://hackmd.io/@pmanzoni/mininetlab2.

[68] N. Neykov, «Real-time detection and mitigation of flood attacks in SDN networks,» *Zeszyty Naukowe Uczelni Jana Wyżykowskiego. Studia z Nauk Technicznych,* τόμ. 6, pp. 171-182, 2017.

[69] A. P. a. D. S. A. Yaar, «SIFF: a stateless Internet flow filter to mitigate DDoS flooding attacks,» *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004,* May 2004.

[70] A. A. a. S. S. R. A. Khan, «Prevention mechanism for infrastructure based Denial-of-Service attack over software Defined Network,» *International Conference on Computing, Communication & Automation,* May 2015.

[71] Open Networking Foundation, «Software-Defined Networking (SDN) Definition,» Open Networking Foundation, 2011. [Ηλεκτρονικό]. Available: https://opennetworking.org/sdn-definition/.

[72] S. A. a. L. S. Harald Baier, «Blessing or curse? Revisiting security aspects of Software-Defined Networking,» ieeexplore, 17 November 2014. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/abstract/document/7014199/.

[73] S. S. a. S. N. Sandra Scott-Hayward, «A Survey of Security in Software Defined Networks,» ieeexplore, 6 July 2015. [Ηλεκτρονικό]. Available: https://ieeexplore.ieee.org/document/7150550.

[74] J. P. M. P. a. S. C. Nhu-Ngoc Dao, «A feasible method to combat against DDoS attack in SDN network,» *2015 International Conference on Information Networking (ICOIN),* 2015.

[75] Open Networking Foundation, «Big Switch Networks – Big Tap,» Open Networking Foundation, 29 March 2013. [Ηλεκτρονικό]. Available: https://opennetworking.org/sdn-resources/sdn-products/big-switch-networks-big-tap/.

[76] NetFPGA, «NetFPGA,» NetFPGA.org, 2007. [Ηλεκτρονικό]. Available: https://netfpga.org/site/#/.

[77] J. C. M. J. T. P. Y. P. S. a. S. B. Andrew R. Curtis, «DevoFlow: scaling flow management for high-performance networks,» ACM Digital Library, August 2011. [Ηλεκτρονικό]. Available: https://dl.acm.org/doi/10.1145/2043164.2018466.

[78] V. S. Y. W. a. G. J. Yueping Zhang, «FlowSense: light-weight networking sensing with openflow,» ReasearchGate.net, 2014. [Ηλεκτρονικό]. Available: https://www.researchgate.net/publication/302811336_FlowSense_light-weight_networking_sensing_with_openflow.

[79] S. S. a. K. V. Deon Herbert, «Flow checker,» Microsoft Dynamics 365, 21 March 2019. [Ηλεκτρονικό]. Available: https://docs.microsoft.com/en-us/business-applications-release-notes/october18/microsoft-flow/flow-checker.

[80] M. Canini, «A NICE way to test OpenFlow controller applications,» github, 7 August 2015. [Ηλεκτρονικό]. Available: https://github.com/mcanini/nice.

[81] J. Chao, «VeriFlow Aims to Verify Application-Defined Networks in Real Time,» Enterprise Networking Planet, 8 August 2013. [Ηλεκτρονικό]. Available: https://www.enterprisenetworkingplanet.com/data-center/veriflow-aims-to-verify-application-defined-networks-in-real-time/.

[82] L. Xu, «FRESCO (Floodlight),» Texas A&M University, 2017. [Ηλεκτρονικό]. Available: https://github.com/xuraylei/fresco_floodlight.

[83] Smoothwall, «Smoothwall open source community,» Smoothwall, 21 October 2014. [Ηλεκτρονικό]. Available: http://www.smoothwall.org/.

[84] GIP Technologies, «How to Install and Configure CSF (Config Server Firewall),» GIP Technologies, [Ηλεκτρονικό]. Available: https://www.globalitproviders.com/tlds/index.php?rp=/knowledgebase/16/How-to-Install-and-Configure-CSF-Config-Server-Firewall.html.

[85] COMODO, «Comodo Award Winning Free Firewall,» COMODO, [Ηλεκτρονικό]. Available: https://www.comodo.com/home/internet-security/firewall.php.

[86] AT&T Cybersecurity, «AlienVault is Now AT&T Cybersecurity,» AT&T Cybersecurity, [Ηλεκτρονικό]. Available: https://cybersecurity.att.com/.

[87] SNORT, «Network Intrusion Detection & Prevention System,» SNORT, [Ηλεκτρονικό]. Available: https://www.snort.org/.

[88] «Treck TCP/IP Stack (Update A),» (I)IoT Security News, [Ηλεκτρονικό]. Available: https://iotsecuritynews.com/treck-tcp-ip-stack-update-a/.

ΛΕΥΚΗ ΣΕΛΙΔΑ