ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
NETMODE – NETWORK MANAGEMENT & OPTIMAL DESIGN LABORATORY

Εξομοίωση κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας μέσω πολλαπλών υποδομών cloud

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΕΙΟΣ ΜΗΛΙΑΣ

Επιβλέπων : Βασίλειος Μάγκλαρης
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούνιος 2016

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

NETMODE – NETWORK MANAGEMENT & OPTIMAL DESIGN LABORATORY

Εξομοίωση κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας μέσω πολλαπλών υποδομών cloud

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΕΙΟΣ ΜΗΛΙΑΣ

Επιβλέπων : Βασίλειος Μάγκλαρης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 15η Ιουλίου 2016.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Βασίλειος Μάγκλαρης Καθηγητής Ε.Μ.Π.	Νεκτάριος Κοζύρης Καθηγητής Ε.Μ.Π.	Ευστάθιος Συκάς Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούνιος 2016

(Υπογραφή)

.....
ΒΑΣΙΛΕΙΟΣ ΜΗΛΙΑΣ
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Ευχαριστίες

Με την παρούσα διπλωματική εργασία ολοκληρώνονται οι σπουδές μου ως προπτυχιακό φοιτητή του τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου.

Καταρχάς, θα ήθελα να ευχαριστώ τον καθηγητή μου κ. Βασίλειο Μάγκλαρη για την ανάθεση της εργασίας αυτής. Ακόμη, ένα μεγάλο ευχαριστώ σε όλα τα μέλη του εργαστηρίου NETMODE τα οποία είναι υπεύθυνα για το εξαιρετικά φιλικό περιβάλλον που επικρατεί στο εργαστήριο. Ειδικότερα, θέλω να ευχαριστήσω τον Κωνσταντίνο Γιώτη και τον Αδάμ Παυλίδη που μου έδιναν την απαραίτητη καθοδήγηση καθ' όλη την διάρκεια της εργασίας.

Τέλος, ευχαριστώ την οικογένεια μου και τους φίλους μου, ο καθένας από τους οποίους βοήθησε με τον δικό του, ιδιαίτερο, τρόπο.

Περιεχόμενα

Κατάλογος σχημάτων

Περίληψη

Η παρούσα διπλωματική εργασία αφορά την εξομοίωση κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας (Distributed Denial of Service attacks – DDoS) μέσω πολλαπλών υποδομών cloud. Θεωρητικά η εργασία αυτή μπορεί διαχωριστεί σε 2 βασικά μέρη:

- Κατασκευή, απεικόνιση και διαίρεση (partitioning) γράφων επιθέσεων DDoS.

- Δημιουργία εργαλείου για την εξομοίωση τέτοιων γράφων το οποίο πετυχαίνει την διανομή των υπογράφων που έχουν προκύψει από το partitioning σε πολλαπλά εικονικά μηχανήματα (ένας υπογράφος ανά μηχάνημα) που βρίσκονται σε διάφορετικές μεταξύ τους υποδομές cloud (π.χ. Πολυτεχνείο, okeanos) καταλήγοντας στο ολικό «χτίσιμο» του αρχικού γράφου.

Για το πρώτο μέρος έγινε συλλογή πραγματικών δεδομένων τόσο για την εύρεση των σχέσεων μεταξύ των αυτόνομων συστημάτων όσο και για την μελέτη μιας πραγματικής DDoS attack που πραγματοποιήθηκε τον Αύγουστο του 2007. Με τα δεδομένα αυτά κατασκευάστηκε ένας αλγόριθμος που δημιουργεί δέντρα και τα απεικονίζει.

Το δεύτερο μέρος της εργασίας βασίστηκε σε μεγάλο βαθμό στον εξομοιωτή mininet, ο οποίος χρησιμοποιήθηκε για την εξομοίωση των επιμέρους υπογράφων. Έγινε σημαντική προσπάθεια στο να υπάρχει η ευκολία προσαρμογής του εργαλείου, αναλόγως με τις ανάγκες του ερευνητή που το χρησιμοποιεί και το να υπάρχει όσο το δυνατόν μεγαλύτερη αυτοματοποίηση για να είναι απλή η χρήση του.

Το εργαλείο αυτό μπορεί να φανεί ιδιαίτερα χρήσιμο σε κάθε περίπτωση που χρειάζεται να γίνει μια εξομοίωση γράφου μεγάλου μεγέθους, όπου δηλαδή, είναι απαραίτητο να διαμοιραστεί ο γράφος σε πολλαπλά εικονικά μηχανήματα σε πολλαπλές υποδομές cloud.

Λέξεις κλειδιά «Κατανεμημένες επιθέσεις άρνησης παροχής υπηρεσίας, εργαλείο εξομοίωσης, υποδομές cloud, mapping, διαμοιρασμός γράφου, απεικόνιση γράφου, Mininet, Ευφυή Προγραμματιζόμεννα Δίκτυα**»**

Abstract

The title of this thesis is «DDoS emulation via dataset-extracted attack graphs mapped on multiple cloud infrastructures». The scope of this project was the development of a tool for the emulation of Distributed Denial of Service attacks (DDoS). Theoretically, this project can be divided into two basic sectors:

- The building, visualization and partitioning of large DDoS attack graphs
- The creation of a tool for the emulation of large DDoS attacks. For this purpose, this tool has the ability to connect with multiple VMs from different cloud infrastructures (NTUA, okeanos, GENI etc), share subgraphs with them, command them to emulate the subgraphs via mininet emulator (one subgraph per VM) and connect the differnent VMs so that finally the initial large topology has been emulated.

The first component of this thesis includes the collection and use of real datasets for the inferring of the existing relationships among the Autonomous Systems (AS) and for the study of a real DDoS attack which took place in August 2007. An algorithm based on those datasets was built for the creation and visualization of trees of attack.

The second component is based in mininet emulator which is used for the emulation of each subgraph. A great effort was made for the tool to be adaptive in the needs of each researcher who may use it and for the process to be as more automated as it could be.

This tool can be very useful not only for the emulation of DDoS attacks but for the testing of any case which has to do with large graphs. For example it is useful in a situation where the researcher desires to test and investigate how a new networking utility is working.

Keywords «DDoS attacks, emulation tool, cloud infrastructures, mapping, graph partitioning, graph visualization, Software Defined Networks, Mininet **»**

1 Εισαγωγή

1.1 Ερευνητικό Πρόβλημα

Διανύουμε μια εποχή όπου οι κατανεμημένες επιθέσεις άρνησης παροχής υπηρεσίας (DDoS attacks) έχουν πληθύνει σημαντικά. Με τα σύγχρονα εργαλεία που κυκλοφορούν είναι δυνατόν να πραγματοποιηθούν αρκετά πολύλποκες και αποτελεσματικές επιθέσεις ακόμα και από χρήστες με μέτρια εμπειρία. Η βασική αρχή αυτών των επιθέσεων έγκειται στο ότι πολλαπλά συστήματα στέλνουν κίνηση προς το στοχευμένο σύστημα, συνήθως κάποιον server, με στόχο να κατακλύσουν τους πόρους του και να τον καταστήσουν μη διαθέσιμο για τους υπόλοιπους χρήστες του.

Η πρώτη τέτοιου είδους επίθεση μεγάλης κλίμακας που εντοπίστηκε έγινε το 1999 με στόχο τον Internet Relay Chat (IRC) server του πανεπιστημίου της Μινεσότα. Επηρέασε 227 συστήματα και κατέστησε ανενεργό τον server του πανεπιστημίου για αρκετές μέρες. Η επίθεση αυτή έδωσε το εναρκτήριο λάκτισμα για μια σειρά από άλλες παρόμοιες επιθέσεις, που είχαν ως αποτέλεσμα να αναδειχτεί η μέθοδος DDoS ως ένα δυνατό εργαλείο για τους χάκερς και τους διάφορους εγκληματίες του κυβερνοχώρου (cybercriminals).

Τον Φεβρουάριο του 2000 μερικές από τις πιο δημοφιλείς ιστοσελίδες έπεσαν εξαιτίας τέτοιων επιθέσων και οι απώλειες που υπέστησαν ήταν τεράστιες. Μεγάλες εμπορικές ιστοσελίδες εταιρειών όπως οι Yahoo, eBay, CNN και Amazon κατέληξαν μη διαθέσιμες για τους χρήστες τους επί ώρες. Ενδιαφέρον παρουσιάζει το γεγονός ότι ενώ κάποιος θα περίμενε οι επιθέσεις αυτές να είναι οργανωμένες από εξειδικευμένες συνεργαζόμενες ομάδες κυβερνο-εγληματιών, στην πραγματικότητα πραγματοποιήθηκαν από έναν 15χρονο Καναδό που είχε ως σκοπό να αναδείξει τις ικανότητές του. Αφού πραγματοποίησε μια δικτυακή σάρωση για να βρει ευάλωτους hosts, στην συνέχεια τους μόλυνε με κακόβουλο λογισμικό, ώστε άθελά τους να συμμετέχουν στην διάδοση της επίθεσης και στην μόλυνση άλλων hosts (τους μετέτρεψε δηλαδή σε αυτό που ονομάζουμε zombies). Ο αριθμός των zombies με αυτόν τον τρόπο αυξήθηκε εκθετικά και έτσι η κακόβουλη κίνηση

έφτασε σε τέτοιο σημείο ώστε η επίθεση αυτή να ρίξει κάποιες από τις μεγαλύτερες ιστοσελίδες του κόσμου.

Από το 2005 ο αριθμός των επιθέσεων DDoS έγινε αρκετά μεγάλος και απασχόλησε σημαντικά την επιστημονική κοινότητα. Η πολυπλοκότητα των επίθεσεων αυτών αλλά και των τρόπων άμυνας απέναντι σε τέτοιες επιθέσεις αυξάνεται μέρα με την μέρα.

Μεγάλη σημασία για την άμυνα απέναντι σε τέτοιες επιθέσεις έχει η Software-defined networking αρχιτεκτονική (SDN), η προέλευση της οποίας χρονολογείται περίπου στο 1995. Η χρήση των SDN έδωσε την δυνατότητα στους διαχειριστές των δικτύων να επιβάλλουν τις επιθυμητές πολιτικές τους άμεσα και με αυτόν τον τρόπο να χειρίζονται τις ροές των πακέτων που δέχονται. Έτσι, αν καταφέρουν δυναμικά να τις κατατάσσουν σε κακόβουλες και μη κακόβουλες, έχουν την δυνατότητα να απορρίψουν όλη την κακόβουλη κίνηση να δεχτούν όλη την μη κακόβουλη. Αυτό αποδεικνύεται εξαιρετικά σημαντικό καθώς η επιθυμητή αμυντική λειτουργία απέναντι στις επιθέσεις DDoS απορρίπτει μονάχα την κακόβουλη κίνηση ώστε να επιτρέπει τους μη κακόβουλους χρήστες να συνεχίσουν να χρησιμοποιούν την εκάστοτε υπηρεσία.

Ένα απαραίτητο κομμάτι για την μελέτη, κατανόηση και αντιμετώπιση των διαδικτυακών επιθέσεων έγκειται στην εξομοίωσή τους. Μια ρεαλιστική εξομοίωση οδηγεί σε μια σημαντικά καλύτερη αντίληψη της εκάστοτε επίθεσης αλλά και στην δυνατότητα αξιολόγησης και αποσφαλμάτωσης (debugging) των διαφόρων τεχνικών άμυνας.

1.2 Συνεισφορά εργασίας

Η παρούσα εργασία αφορά την μελέτη και την εξομοίωση κατανεμημένων επιθέσεων άρνησης υπηρεσίας. Η ουσιαστική συνεισφορά της μπορεί να χωριστεί σε 2 βασικούς κλάδους.

Ο πρώτος κλάδος συμβάλλει στην θεωρητική μελέτη τέτοιων επιθέσεων. Συγκεκριμένα αφορά την κατασκευή και απεικόνιση πραγματικών γράφων επίθεσης όπου ο κάθε κόμβος αντιπροσωπεύει ένα αυτόνομο σύστημα (ΑΣ). Ως γράφος επίθεσης στην εργασία αυτή, θεωρείται ένα δέντρο το οποίο έχει ως ρίζα το θύμα-ΑΣ και απαρτίζεται από τα έγκυρα μονοπάτια που σχηματίζονται από τα κακόβουλα-ΑΣ προς το θύμα-ΑΣ. Η ορολογία «έγκυρα μονοπάτια» θα εξηγηθεί σε επόμενο κεφάλαιο της εργασίας διεξοδικά. Επίσης γίνεται ένας διαμοιρασμός (partitioning) του γράφου σε k (παράμετρος) συνδεδεμένους υπογράφους, όπου η επιλογή των k υπογράφων γίνεται με στόχο την ελαχιστοποίηση του αριθμού των ακμών που χρειάζεται να αφαιρεθούν από τον αρχικό γράφο.

Ο δεύτερος κλάδος αφορά την δημιουργία του εργαλείου εξομοίωσης κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας σε πολλαπλές cloud υποδομές. Το εργαλείο αυτό έχει την δυνατότητα να κατασκευάζει τους

υπογράφους, που περιγράφηκαν στην προηγούμενη παράγραφο, σε εικονικά μηχανήματα με την βοήθεια του εξομοιωτή mininet και να τους συνδέει σχηματίζοντας έτσι τον συνολικό γράφο. Η σύνδεση με τα εικονικά μηχανήματα και η κατασκευή των υπογράφων μπορεί να γίνει σειριακά αλλά και παράλληλα. Ενδιαφέρον εδώ αποτελεί η περίπτωση στην οποία ο γράφος εξομοιώνεται μέσω εικονικών μηχανημάτων που βρίσκονται σε διαφορετικές cloud υποδομές (π.χ. μερικά στο Πολυτεχνείο και μερικά στο okeanos). Τέλος το εργαλείο πυροδοτεί την επίθεση βάζοντας όλους τους κακόβουλους hosts να στείλουν κίνηση στο θύμα.

1.3 Δομή εργασίας

Το υπόλοιπο της παρούσας εργασίας δομείται ως εξής:

• ***2ο κεφάλαιο****:* Δίνεται ένα βασικό θεωρητικό υπόβαθρο απαραίτητο για την κατανόηση των επόμενων κεφαλαίων. Ειδικά, αναφέρονται τα βασικά χαρακτηριστικά των Ευφυών Προγραμματιζόμενων Δικτύων (SDN) του πρωτοκόλλου OpenFlow και του εξομοιωτή Mininet όπως και κάποιες γενικές αρχές του Cloud Computing.

• ***3ο κεφάλαιο****:* Παρουσιάζονται τα χαρακτηριστικά των Αυτόνομων Συστημάτων, οι κύριες σχέσεις που επικρατούν μεταξύ τους, οι γράφοι που περιγράφουν τέτοιες σχέσεις και τα έγκυρα μονοπάτια σε τέτοιους γράφους.

• ***4ο κεφάλαιο***: Μελετάται η κατασκευή και απεικόνιση των γράφων επίθεσης, παρουσιάζονται οι αλγόριθμοι που χρησιμοποιήθηκαν προς τον σκοπό αυτό καθώς και διάφορα σχήματα τέτοιων γράφων. Εδώ παρουσιάζετια και ο σκοπός και η μέθοδος διαμοιρασμού του γράφου.

• ***5ο κεφάλαιο****:* Εξηγείται αναλυτικά η κατασκευή και ο τρόπος λειτουργίας του εργαλείου εξομοίωσης από την αρχή μέχρι το τέλος, παρουσιάζεται η αρχιτεκτονική του εργαλείου και δίνεται ένα διάγραμμα ροής του.

• ***6ο κεφάλαιο****:* Παρουσιάζονται τα αποτελέσματα του εργαλείου εξομοίωσης τα οποία αφορούν τόσο τον σχεδιασμό των δικτυακών γράφων που κατασκευάστηκαν όσο και τις μετρήσεις που πραγματοποιήθηκαν για την μελέτη της απόδοσης των δικτύων αυτών και αναφέρονται οι μελλοντικές επεκτάσεις της εργασίας αυτής.

2 Θεωρητικό Υπόβαθρο

Στο κεφάλαιο αυτό αναλύονται κάποιες βασικές έννοιες και τεχνολογίες απαραίτητες για την κατανόηση της υπόλοιπης εργασίας. Στην περίπτωση που ο αναγνώστης θεωρεί ότι είναι αρκετά εξοικειωμένος με τις έννοιες που παρουσιάζονται εδώ, μπορεί φυσικά να προσπεράσει το εν λόγω κεφάλαιο.

2.1 Βασικές αρχές

Η ανάπτυξη της τεχνολογίας έχει εδώ και καιρό καταστήσει σαφές ότι η αρχιτεκτονική των δικτύων οφείλει να εξελίσσεται ώστε να καλύπτει τις ανάγκες της εκάστοτε εποχής. Η ραγδαία αύξηση του αριθμού των φορητών συσκευών που συνδέονται στο Internet, του μεγέθους των πληροφοριών (Big data) που ανταλάσσονται μέσω αυτού και η συνεχώς μεγαλύτερη τάση για χρησιμοποίηση υπηρεσιών cloud οδηγούν την βιομηχανία στην επανεξέταση των παραδοσιακών δικτυακών αρχιτεκτονικών. Οι βασικοί λόγοι που συντέλεσαν στην ανάγκη δημιουργίας καινούργιων αρχιτεκτονικών μοντέλων μπορούν να χωριστούν στις εξής κατηγορίες:

- **Εναλλαγή μοτίβων κίνησης:** Σε αντίθεση με την αρχιτεκτονική client-server όπου ο κύριος όγκος δεδομένων ανταλάσσεται μεταξύ ενός πελάτη και ενός server, πλέον τα δεδομένα ανταλλάσσονται μεταξύ διαφόρων βάσεων δεδομένων και servers μέχρι η πληροφορία να φτάσει στον πελάτη.

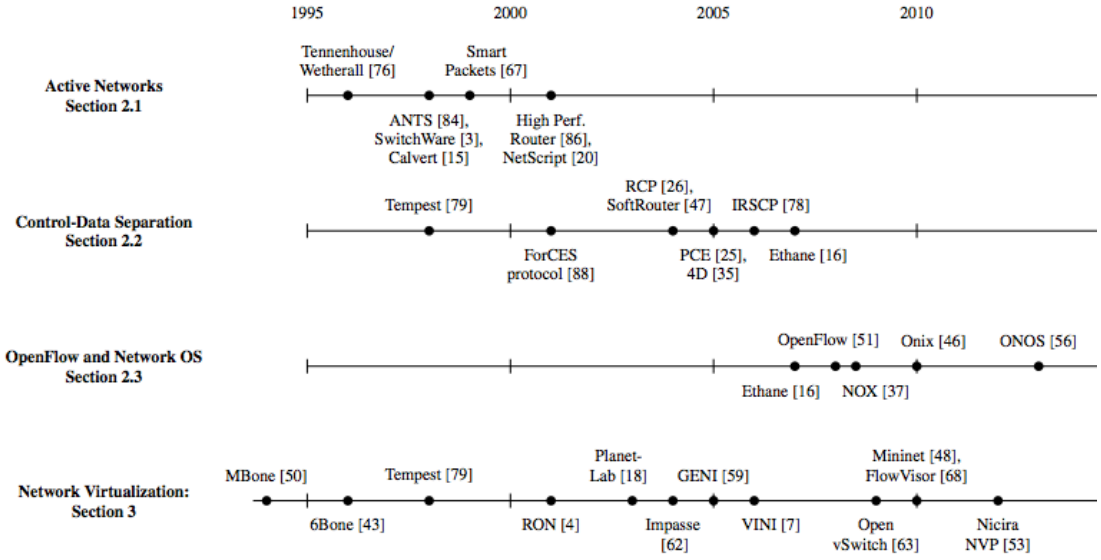
- **Αύξηση της χρήσης των υπηρεσιών cloud :** Ολοένα και παρισσότερες επιχειρήσεις και οργανισμοί χρησιμοποιούν σε μεγάλο βαθμό τόσο δημόσιες όσο και ιδιωτικές cloud υπηρεσίες και εφαρμογές και επιθυμούν να έχουν πρόσβαση σε διάφορες τέτοιες υποδομές άμεσα και κατά παραγγελία. Η στατικότητα που προσέφερε η παλαιότερη αρχιτεκτονική των δικτύων δεν ευνοεί φυσικά τέτοιες επιλογές.

-Αύξηση της χρήσης κινητών συσκευών: Ο αριθμός των διαφόρων κινητών συσκευών (smartphones, notebooks, tablets) με σύνδεση στο Internet έχει αυξηθεί και συνεχίζεται να αυξάνεται δραματικά τα τελευταία χρόνια με αποτέλεσμα να

πιέζει την υποδομή του διαδικτύου να βρει τρόπο να «φιλοξενήσει» τις συσκευές αυτές και να ανταπεξέλθει στις ανάγκες του κάθε χρήστη.

- Big Data ➔ Μεγαλύτερο bandwidth: Για την καλύτερη χρησιμοποίηση του τεράστιου όγκου δεδομένων που είναι πλέον προσβάσιμος από το διαδίκτυο απαιτείται η παράλληλη επεξεργασία των δεδομένων σε χιλιάδες servers οι οποίοι χρειάζονται απευθείας συνδέσεις μεταξύ τους.

Στο [1] παρουσιάζεται η ιστορία των προγραμματιζόμενων δικτύων (programmable networks) με το πέρασμα από τα ενεργά δίκτυα (Active Networks - μέσα 1900) [2] στα προγραμματιζόμενα δίκτυα [3] και καταλήγωντας στα «Δίκτυα καθοριζόμενα από Λογισμικό» ή όπως είναι ευρεώς γνωστά «Software Defined Networks» (SDN) [4]. Μια ενδιαφέρουσα εικόνα που παρουσιάζει την εξέλιξη αυτή είναι η παρακάτω:



Σχήμα 2.1: Εξέλιξη των προγραμματιζόμενων δικτύων τα τελευαία 20 χρόνια, Πηγή: [1]

2.2 SDN Αρχιτεκτονικές

Η SDN είναι μια αρχιτεκτονική που έχει στόχο να είναι δυναμική, εύχρηστη, προσαρμόσιμη, αποδοτική ως προς το κόστος και να υποστηρίζει το υψηλό bandwidth και τις σύγχρονες δυναμικές εφαρμογές. Οι αρχιτεκτονικές SDN περιέχουν συναρτήσεις για τον έλεγχο των δικτύων και την προώθηση της κίνησης. Καθιστούν τον έλεγχο ευκίνητο και διαχωρίζουν την βασική υποδομή από τις εφαρμογές και τις δικτυακές υπηρεσίες. Χαρακτηριστικά της αρχιτεκτονικής SDN είναι τα εξής:

- **Προγραμματίζεται άμεσα:** Ο έλεγχος του δικτύου γίνεται άμεσα, μέσω προγραμματισμού, διότι καθορίζεται από συναρτήσεις προώθησης.

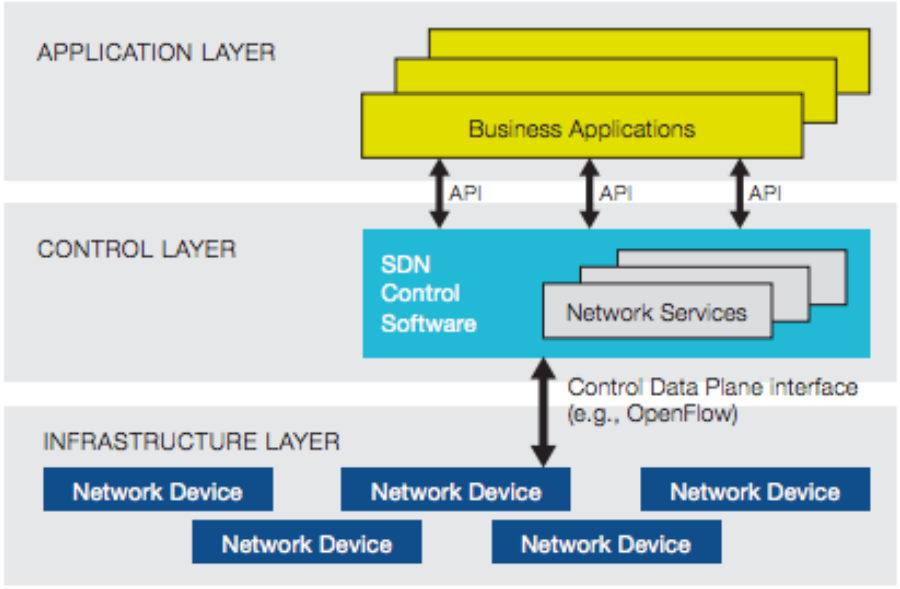
- **Ευκίνητη:** Με τον διαχωρισμό του ελέγχου με την προώθηση επιτρέπει στους διαχειριστές να προσαρμόζουν δυναμικά την κίνηση σε όλο το δίκτυο ώστε να καλύπτουν τις μεταβαλλόμενες ανάγκες.

- **Κεντρική διαχείριση:** Η ευφυία των δικτύων αυτών είναι συγκεντρωμένη στον βασισμένο σε λογισμικό ελεγκτή SDN (θα αναφερθεί σε επόμενη ενότητα) ο οποίος έχει μια ολοκληρωμένη εικόνα του δικτύου και εμφανίζεται στις εφαρμογές και στις διάφορες μηχανές πολιτικής (policy engines) σαν ένας απλός λογικός μεταγωγέας.

- **Διαμορφώνεται μέσω προγραμματισμού:** Οι αρχιτεκτονικές SDN επιτρέπουν στον διαχειριστή να διαμορφώσει, ασφαλίσει και βελτιστοποιήσει την χρήση των πόρων του δικτύου εξαιρετικά γρήγορα μέσω δυναμικών προγραμμάτων SDN τα οποία μπορεί να γράψει ο ίδιος ο διαχειριστής καθώς τα προγράμματα αυτά δεν βασίζονται σε ιδιόκτητο λογισμικό.

Στο σχήμα 2.2 παρουσιάζεται η βασική ιδέα πίσω από την αρχιτεκτονική SDN . Υπενθυμίζουμε ότι με την αρχιτεκτονική αυτή γίνεται ένας διαχωρισμός μεταξύ του ελέγχου και της διαδικασίας προώθησης.

Ξεκινώντας την περιγραφή bottom-up, στο χαμηλότερο επίπεδο βρίσκεται το στρώμα δεδομένων (infrastructure layer). Το επίπεδο αυτό είναι υπεύθυνο για την παρακολούθηση της κίνησης, την συλλογή δεδομένων, την εξαγωγή στατιστικών πληροφοριών αλλά για την προώθηση πακέτων.



Σχήμα 2.2: Αρχιτεκτονική SDN 3 επιπέδων, Πηγή: [4]

Το επόμενο επίπεδο αποτελείται από το στρώμα ελέγχου, που θεωρείται και ο πυρήνας των αρχιτεκτονικών SDN. Πρακτικά το στρώμα ελέγχου, όπως δηλώνει και το όνομά του, ελέγχει ή αλλιώς διαχειρίζεται το στρώμα δεδομένων. Το στρώμα δεδομένων παρέχει πληροφορίες στο στρώμα ελέγχου και έτσι το δεύτερο αποφασίζει για την δρομολόγηση και γενικότερα την συνολική λειτουργία του δικτύου.

Το υψηλότερο επίπεδο αποτελείται από το στρώμα εφαρμογής. Το στρώμα εφαρμογής έχει την δυνατότητα να παρέχει δικτυακές υπηρεσίες που μπορεί για παράδειγμα να σχετίζονται με την ασφάλεια ή την προώθηση κίνησης και έτσι καθορίζει σε μεγάλο βαθμό την συμπεριφορά του δικτύου.

Τελικά οι εφαρμογές αντιμετωπίζουν το δίκτυο μέσω του API του ελεγκτή ως μεταγωγέα (switch). Το γεγονός αυτό οδηγεί φυσικά σε σημαντική απλοποίηση του σχεδιασμού του δικτύου. Έτσι, με την αρχιτεκτονική SDN δεν είναι απαραίτητο οι δικτυακές συσκευές να κατανοούν πραγματικά τα δικτυακά πρωτόκολλα, αλλά μόνο να μπορούν να δέχονται οδηγίες για τον τρόπο επεξεργασίας της κίνησης από το λογισμικό του SDN που είναι υπεύθυνο για τον έλεγχο.

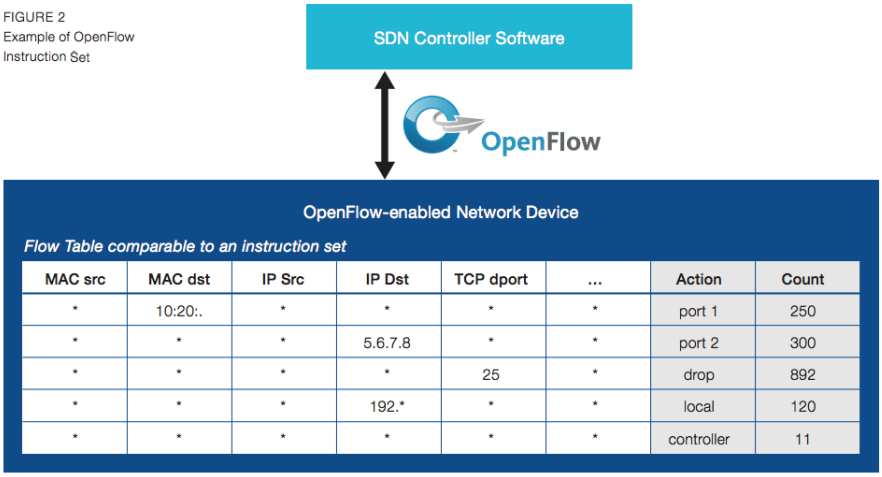
Η αρχιτεκτονική SDN είναι έντονα συνυφασμένη με το πρωτόκολλο OpenFlow [5]. Είναι σημαντικό να σημειωθεί ότι ενώ το πρωτόκολλο αυτό είναι χρησιμοποιείται στις περισσότερες περιπτώσεις υλοποίησης Software Defined Networks δεν είναι και το μόνο που υπάρχει. Καθώς η εν λόγω εργασία βασίστηκε βέβαια στο συγκεκριμένο πρωτόκολλο θα επεξηγηθούν τα βασικά του χαρακτηριστικά στην ενότητα που ακολουθεί.

2.3 Openflow

Το πρωτόκολλο OpenFlow ήταν το πρώτο που κατάφερε να παρέχει επικοινωνία μεταξύ του στρώματος ελέγχου και του στρώματος δεδομένων, δηλαδή μεταξύ ελέγχου και προώθησης. Επιτρέπει την άμεση πρόσβαση και επεξεργασία του επιπέδου προώθησης είτε των φυσικών είτε των εικονικών δικτυακών συσκευών, όπως είναι οι μεταγωγείς και οι δρομολογητές, από τον ελεγκτή. Πρακτικά, παρέχεται η δυνατότητα στον ελεγκτή να ορίσει το μονοπάτι που θα ακολουθήσουν τα πακέτα σε ένα δίκτυο από μεταγωγείς. Καθώς οι ελεγκτές είναι πάντα διαχωρισμένοι από τους μεταγωγείς υπάρχει μια μεγαλύτερη ευελιξία στην διαχείριση της κίνησης από ό,τι υπήρχε με την χρησιμοποίηση των access control lists (ACLs) και των πρωτοκόλλων δρομολόγησης.

Το OpenFlow χρησιμοποιεί την έννοια των ροών πακέτων (flows) για να επεξεργαστεί την δικτυακή κίνηση, η δρομολόγηση της οποίας εξαρτάται από προκαθορισμένους κανόνες ταιριάσματος (matching rules) που δημιουργούνται είτε στατικά είτε δυναμικά μέσω του ελεγκτή SDN. Με αυτόν τον τρόπο ολόκληρο το δίκτυο, που είναι βασισμένο σε μια αρχιτεκτονική OpenFlow, αποκτά την δυνατότητα να ανταπεξέλθει σε αλλαγές που συμβαίνουν στο δίκτυο σε πραγματικό χρόνο. Παρουσιάζεται παρακάτω μια εικόνα που περιέχει το instruction set του OpenFlow.

FIGURE 2
Example of OpenFlow
Instruction Set



MAC src	MAC dst	IP Src	IP Dst	TCP dport	...	Action	Count
*	10:20:.	*	*	*	*	port 1	250
*	*	*	5.6.7.8	*	*	port 2	300
*	*	*	*	25	*	drop	892
*	*	*	192.*	*	*	local	120
*	*	*	*	*	*	controller	11

Σχήμα 2.3: OpenFlow – Instruction Set, Πηγή [4]

Η επικοινωνία των μεταγωγέων με το στρώμα ελέγχου γίνεται μέσω ενός καναλιού το οποίο οφείλει να παρέχει την απαραίτητη ασφάλεια. Σε μια αντιστοιχία με την αρχιτεκτονική των SDN που παρουσιάστηκε παραπάνω ισχύει ότι:

- Στρώμα Δεδομένων ➔ Μεταγωγείς OpenFlow (Switches)

- Στρώμα Ελέγχου ➔ Ελεγκτές OpenFlow (Controllers)

Τα βασικότερα πλεονεκτήματα της αρχιτεκτονικής OpenFlow SDN είναι τα εξής:

- Κεντρικοποιημένος έλεγχος: Το λογισμικό ελέγχου μπορεί να ελέγχει κάθε δικτυακή συσκευή OpenFlow-enabled και έτσι μπορεί να ρυθμίσει, αναβαθμίσει και επεξεργαστεί κάθε μία από αυτές, άρα και όλο το δίκτυο, εύκολα και γρήγορα.

- Μειωμένη πολυπλοκότητα: Παρέχεται μια ιδιαίτερη ευελιξία για την διαχείριση του δικτύου καθώς και η δυνατότητα ανάπτυξης εργαλείων που μπορούν να αυτοματοποιούν διαχειριστικές εργασίες που σε άλλη περίπτωση θα έπρεπε να γίνουν χειροκίνητα.

- Δυνατότητα καινοτομίας: Οι διαχειριστές μπορούν μέσω του OpenFlow να κάνουν επανειλημμένα προγραμματιστικές αλλαγές στο δίκτυο που διαχειρίζονται έτσι ώστε να ικανοποιούν τις ανάγκες των χρηστών του δικτύου οι οποίες αλλάζουν συνεχώς. Μπορούν δηλαδή να παρέχουν καινούργιες υπηρεσίες μέσα σε λίγες μόνο ώρες.

- Μεγαλύτερη ασφάλεια και αξιοπιστία: Είναι δυνατή η διαμόρφωση πολιτικών και η ρύθμιση του δικτύου σε υψηλό επίπεδο τα οποία σε συνδυασμό με το ότι

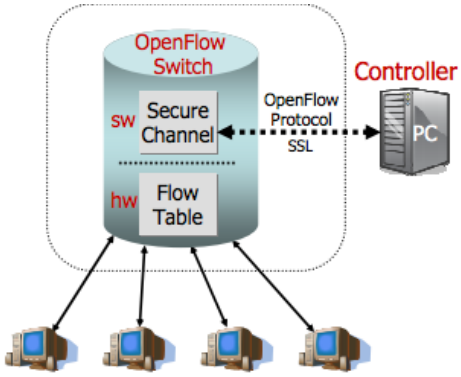
υπάρχει μια καθολική εποπτεία του δικτύου από τον ελεγκτή καταλήγουν σε ένα δίκτυο πιο ασφαλές και καλύτερης ποιότητας.

- Καλύτερος έλεγχος: Με το OpenFlow δίνεται η δυνατότητα εφαρμογής ευέλικτων κανόνων οι οποίοι μπορούν να εξυπηρετήσουν ακόμα και τις ανάγκες ενός απαιτητικού διαχειριστή.

- Βέλτιστη εμπειρία σε επίπεδο χρήστη: Το δίκτυο μπορεί να διαμορφώνεται σύμφωνα με τις δυναμικές ανάγκες του εκάστοτε χρήστη.

2.3.1 Μεταγωγέας

Ο μεταγωγέας OpenFlow είναι υπεύθυνος για την προώθηση των πακέτων. Η προώθηση αυτή, όπως έχει προαναφερθεί, βασίζεται στο flow table του εκάστοτε μεταγωγέα το οποίο ρυθμίζεται μέσω ενός ασφαλούς καναλιού από τον ελεγκτή όπως φαίνεται στο παρακάτω σχήμα:



Σχήμα 2.4: OpenFlow μεταγωγέας, Πηγή [6]

Το flow table περιέχει καταχωρήσεις της μορφής :

Header Fields	Counters	Actions

Σχήμα 2.5: Καταχώρηση σε flow table, Πηγή: [7]

Και το κάθε κελί από τα παραπάνω εξυπηρετεί έναν διαφορετικό σκοπό:

- **Πεδία επικεφαλίδας (Header fields):** Με βάση τα πεδία αυτά γίνεται το ταίριασμα του κανόνα με το εισερχόμενο πακέτο. Γίνεται δηλαδή μια σύγκριση ανάμεσα στα πεδία επικεφαλίδας του εκάστοτε πακέτου και στα πεδία επικεφαλίδας των διάφορων κανόνων και επιλέγεται το ποιον κανόνα θα ακολουθήσει το πακέτο αυτό. Τα πεδία αυτά για παράδειγμα μπορούν να

ταιριάζουν πακέτα με κανόνες ανάλογα με την IP διεύθυνση του απόστολέα ή του παραλήπτη του πακέτου. Τέτοια πεδία είναι τα παρακάτω:

Ingress Port	Ether source	Ether dst	Ether type	VLAN id	VLAN pri-or-ity	IP src	IP dst	IP proto	IP ToS bits	TCP/ UDP src port	TCP/ UDP dst port

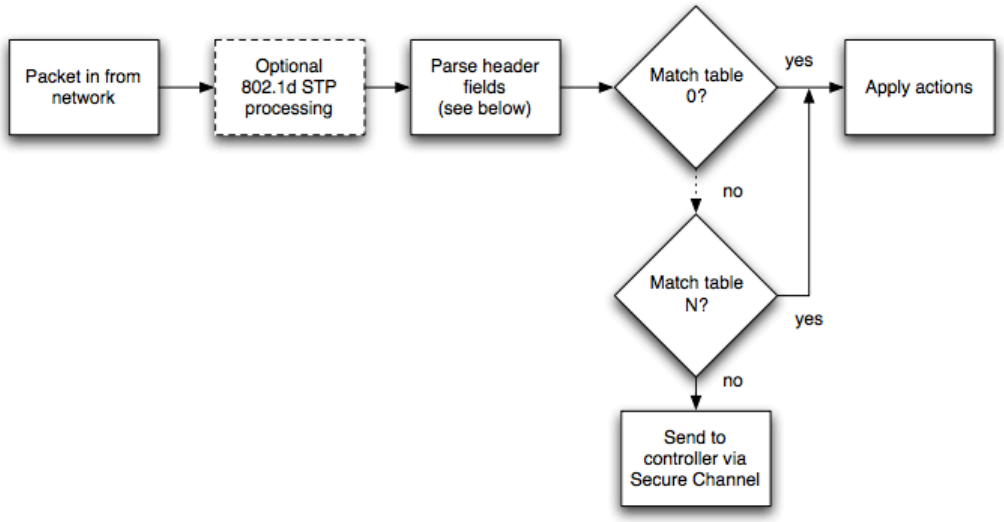
Σχήμα 2.6: Πεδία πακέτων που χρησιμοποιούνται για το ταίριασμα πακέτων-κανόνων, Πηγή: [7]

- **Μετρητές (Counters):** Οι μετρητές είναι υπεύθυνοι για την συλλογή πληροφοριών ανά table, ανά flow και ανά θύρα. Οι πληροφορίες που συλλέγουν αφορούν τον αριθμό και το μέγεθος των εισερχόμενων πακέτων αλλά και την διάρκεια για την οποία ο κανόνας συνεχίζει να ισχύει.

- **Ενέργειες (Actions):** Οι ενέργειες, αφορούν την «απόφαση» του μεταγωγέα για το πως θα χειριστεί το εισερχόμενο πακέτο. Αφού δηλαδή έχει γίνει το ταίριασμα του πακέτου με κάποιον κανόνα, ο κανόνας περιέχει την πληροφορία του τι θα γίνει το πακέτο στην συνέχεια (προώθηση, απόρριψη κλπ). Ένα απλό παράδειγμα τέτοιας ενέργειας είναι η απόρριψη όσων πακέτων έχουν ταιριάξει με μία διεύθυνση IP για την οποία είναι γνωστό ότι στέλνει κακόβουλη κίνηση.

2.3.2 Ταίριασμα (Matching)

Το παρακάτω σχήμα δείχνει την ροή των πακέτων σε έναν μεταγωγέα OpenFlow .

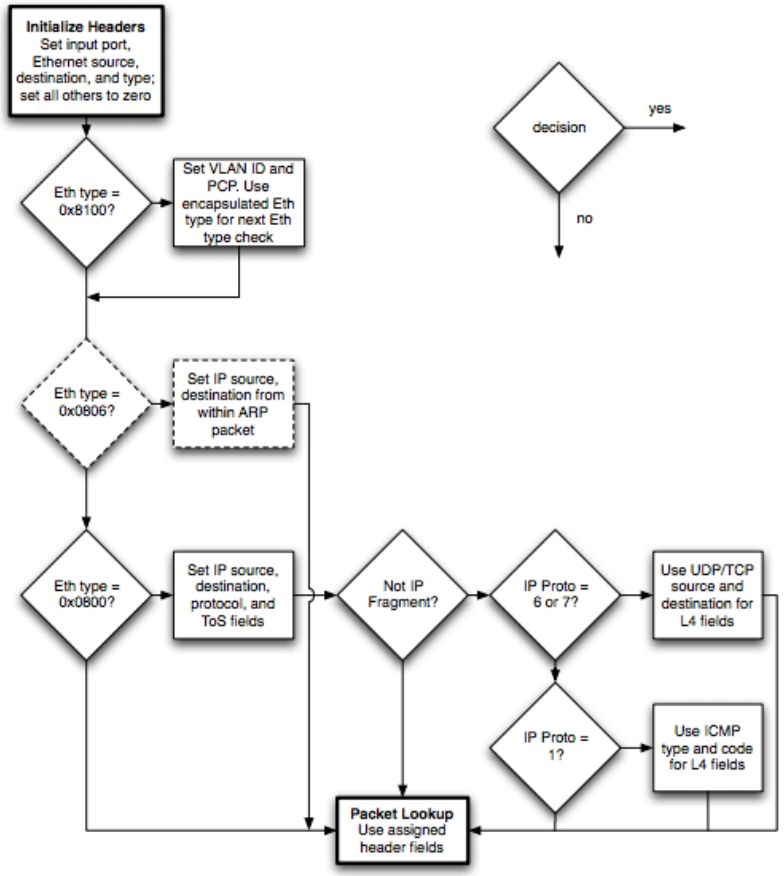


Σχήμα 2.7: Διαδικασία επεξεργασίας πακέτου – OpenFlow, Πηγή: [7]

Εισερχόμενο πακέτο από το δίκτυο ➔ Προαιρετική επεξεργασία (802.1d Spanning Tree Protocol) ➔ Σάρωση πεδίων επικεφαλίδας ➔ Ταίριασμα με κάποιον κανόνα του flow table ➔ Αν ναι τότε εφαρμόζονται οι εν λόγω ενέργειες, αν όχι τότε το πακέτο αποστέλλεται στον ελεγκτή μέσω του ασφαλούς καναλιού.

Εδώ έχει ενδιαφέρον να παρουσιαστεί λίγο αναλυτικότερα η διαδικασία σάρωσης των πεδίων επικεφαλίδας. Το σχήμα που ακολουθεί βοηθά στην κατανόηση των ενεργειών που ακολουθούνται κατά την διαδικασία αυτή. Η διαδικασία αφορά:

- Κανόνες που ορίζουν μία Ingress θύρα αντιστοιχίζονται με την φυσική θύρα από την οποία ήρθε το πακέτο.
- Κεφαλίδες Ethernet οι οποίες χρησιμοποιούνται για όλα τα πακέτα.
- Αν το πακέτο είναι VLAN τα πεδία VLAN ID και PCP χρησιμοποιούνται.
- Προαιρετικά για τα πακέτα ARP μπορούν να χρησιμοποιηθούν τα πεδία που περιέχουν τις διευθύνσεις IP (αποστολέα και παραλήπτη).
- Για τα πακέτα IP χρησιμοποιούνται τα πεδία της επικεφαλίδας IP ενώ αν αυτά είναι πακέτα TCP | UDP χρησιμοποιούνται και οι θύρες του επιπέδου μεταφοράς (transport ports).
- Για τα πακέτα ICMP χρησιμοποιούνται τα πεδία Type και Code.
- Για τα θρυμματισμένα πακέτα δεν χρησιμοποιούνται οι transport ports.



Σχήμα 2.8: Διαδικασία αντιστοίχησηςεπικεφαλίδων – OpenFlow, Πηγή: [7]

Ένα πακέτο κάνει ταίριασμα με μια καταχώρηση του flow table αν οι τιμές των πεδίων επικεφαλίδας του ταιριάζουν με αυτές που ορίζονται στο flow table. Αν μια καταχώρηση στο flow table έχει τιμή ANY, σημαίνει ότι ο κανόνας αυτός ταιριάζει με κάθε πακέτο.

2.3.3 Ελεγκτής

Ο ελεγκτής είναι υπεύθυνος για τον χειρισμό των πακέτων για τα οποία δεν υπάρχει καμμία καταχώρηση στο flow table του εκάστοτε μεταγωγέα τέτοια ώστε να ταιριάξει με τα πακέτα αυτά, καθώς και για την διαχείριση, μέσω προσθαφαίρεσης και επεξεργασίας, των κανόνων των flow tables. Ο κάθε μεταγωγέας λοιπόν, επικοινωνεί με τον ελεγκτή με ένα από τα 3 είδη μηνυμάτων:

- **Ελεγκτή - προς - Μεταγωγέα:** Τα μηνύματα αυτά αρχικοποιούνται από τον ελεγκτή και αναλόγως την περίπτωση απαιτούν ή όχι απάντηση από τον μεταγωγέα. Τα μηνύματα αυτά αφορούν την παραμετροποίηση του μεταγωγέα από τον ελεγκτή.

- **Ασύγχρονα:** Τα μηνύματα αυτά αποστέλλονται από τον μεταγωγέα προς τον ελεγκτή χωρίς να προηγηθεί κάποιου είδους πρόσκληση από τον ελεγκτή και χρησιμοποιούνται για να αναφέρουν την άφιξη ενός πακέτου, την κατάσταση του μεταγωγέα ή κάποιο σφάλμα. Οι κύριες κατηγορίες τέτοιων μηνυμάτων είναι: **Packet-in**, **Flow-Removed**, **Port-status** και **Error.**

- Συμμετρικά: Τα μηνύματα που εμπίπτουν στην κατηγορία αυτή αποστέλλονται είτε από τον ελεγκτή προς τον μεταγωγέα είτε αντίστροφα, και δεν χρειάζονται κανενός είδους πρόσκληση από την άλλη πλευρά. Τέτοια μηνύματα μπορεί να αφορούν την αρχική σύνδεση μεταξύ ελεγκτή και μεταγωγέα ή ακόμα και για να γίνει έλεγχος του bandwidth και του latency μεταξύ των δύο.

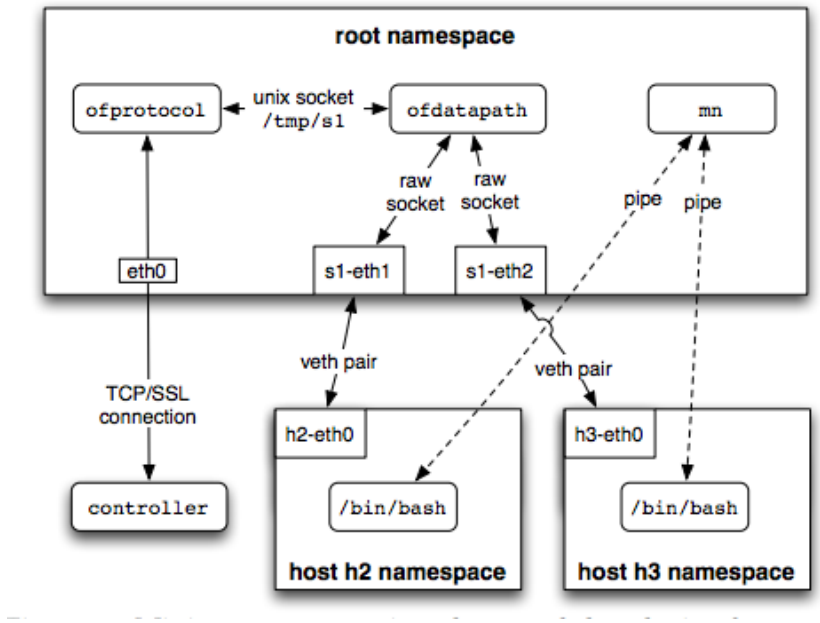
Περισσότερα για όλα τα παραπάνω που είναι σχετικά με το OpenFlow μπορούν να βρεθούν αναλυτικότερα και εξαιρετικά καλογραμμένα στο [7].

2.4 Εξομοιωτής Mininet

Το mininet [8] είναι ένα εργαλείο εξομοίωσης το οποίο έχει την δυνατότητα να κατασκευάζει δίκτυα αποτελούμενα από εικονικούς hosts, μεταγωγείς, ελεγκτές και συνδέσμους. Οι hosts «τρέχουν» λογισμικό Linux και οι μεταγωγείς υποστηρίζουν το πρωτόκολλο OpenFLow προκειμένου να μπορούν να εξομοιώνουν Δίκτυα Προγραμματιζόμενα από Λογισμικό (SDN). Το βασικό πλεονέκτημα το mininet έναντι άλλων εργαλείων εξομοίωσης SDN είναι ότι ο χρήστης μπορεί να δοκιμάσει μια καινούργια αρχιτεκτονική ή μια καινούργια υπηρεσία σε μια μεγάλη εξομοιωμένη τοπολογία και στην συνέχεια να χρησιμοποιήσει τον ίδιο κώδικα σε ένα πραγματικό εμπορικό δίκτυο.

Το Mininet ουσιαστικά χρησιμοποιεί εικονικοποίηση βασισμένη στην διεργασία (process-based virtualization) ώστε να «σηκώσει» πολλούς (έχουν καταφέρει μέχρι 4096) hosts και μεταγωγείς σε στον πυρήνα ενός λειτουργικού συστήματος. Είναι γραμμένο σχεδόν εξ ολοκλήρου σε Python.

Στο σχήμα *2.11* παρουσιάζονται τα συστατικά και οι σύνδεσμοι ενός δικτύου που αποτελείται από 2 hosts και είναι κατασκευασμένο μέσω Mininet.



Σχήμα 2.9: Εικονικό δίκτυο αποτελούμενο από 2-hosts – Mininet , Πηγή: [9]

Οι 2 hosts έχουν τοποθετηθεί σε 2 namespaces και συνδέονται μέσω εικονικών Ethernet (veth) pairs. Επίσης είναι συνδεδεμένα με έναν OpenFLow μεταγωγέα που βρίσκεται στο userspace. Πρακτικά τα εξής στοιχεία εμφανίζονται στο παραπάνω σχήμα:

- **Σύνδεσμοι (Links):** Ένα εικονικό Ethernet pair λειτουργεί ως ένα καλώδιο το οποίο συνδέει δύο εικονικές διεπαφές. Τα πακέτα τα οποία αποστέλλονται από την μία διεπαφή παραδίδονται στην άλλη και κάθε διεπαφή εμφανίζεται ως μια πλήρως λειτουργική θύρα Ethernet σε όλες τις υπόλοιπες εφαρμογές.

- **Hosts:** Στο Mininet κάθε host είναι πρακτικά μια διεργασία φλοιού (shell process) που έχει μεταφερθεί στο δικό της namespace. Κάθε host έχει την δικιά του εικονική διεπαφή Ethernet και ένα pipe που τον συνδέει με την parent-διεργασία.
- **Μεταγωγείς:** Οι μεταγωγείς είναι υλοποιημένοι με λογισμικό και φυσικά υποστηρίζουν το πρωτόκολλο OpenFlow. Λειτουργούν όπως λειτουργούν και οι φυσικοί μεταγωγείς (hardware switches).
- **Ελεγκτές:** Οι ελεγκτές μπορούν να βρίσκονται οπουδήποτε στο πραγματικό είτε στο εξομοιωμένο δίκτυο αρκεί το μηχάνημα στο οποίο «τρέχουν» οι μεταγωγείς να παρέχει συνδεσιμότητα επιπέδου IP. Για παράδειγμα σε περίπτωση που το Mininet «τρέχει» σε ένα εικονικό μηχάνημα, ο ελεγκτής θα μπορούσε να βρίσκεται είτε μέσα σε αυτό είτε στο host μηχάνημα.

Όπως θα φανεί και παρακάτω το Mininet χρησιμοποιήθηκε εξαιρετικά στην εργασία αυτή. Περισσότερα για το mininet μπορούν να βρεθούν στο [8] αλλά και στο [9].

2.5 Υπολογιστικό Νέφος (Cloud Computing)

Το υπολογιστικό νέφος αφορά την κατ' αίτηση διαδικτυακή κεντρική διάθεση υπολογιστικών πόρων για την αποθήκευση, επεξεργασία και χρήση δεδομένων, διαδικτυακά, μέσω απομακρυσμένων υπολογιστών που βρίσκονται σε κεντρικά Datacenter. Με τον τρόπο αυτό, μια εργασία που θα ήταν αδύνατο να πραγματοποιηθεί σε ένα μόνο μηχάνημα μπορεί να μοιραστεί σε ένα σύνολο απομακρυσμένων υπολογιστών οι οποίοι συντονισμένα να την φέρουν εις πέρας.

Στην παρούσα εργασία θα παρουσιαστεί ένα εργαλείο που έχει την δυνατότητα να διαμοιράζει και να συνδέει κόμβους μεταξύ mininet που βρίσκονται σε διαφορετικά εικονικά μηχανήματα τα οποία με την σειρά τους βρίσκονται σε διαφορετικές υποδομές cloud . Αυτό έχει ως αποτέλεσμα να έχει ο χρήστης την δυνατότητα να συνδυάσει την ισχύ των διάφορων εικονικών μηχανημάτων που έχει ακόμα και αν αυτά δεν βρίσκονται στο ίδιο cloud.

3 Αυτόνομα Συστήματα

3.1 Χαρακτηριστικά & τύποι Αυτόνομων Συστημάτων

Το διαδίκτυο είναι οργανωμένο σε Αυτόνομα Συστήματα (ΑΣ). ΑΣ ονομάζεται ένα δίκτυο ή μια συλλογή από δίκτυα που αντιπροσωπεύονται από μία κοινή διαχειριστική αρχή. Η δρομολόγηση των πακέτων εσωτερικά του ΑΣ ρυθμίζεται μέσω του Interior Gateway Protocol (IGP) ενώ η δρομολόγηση μεταξύ διαφορετικών ΑΣ ρυθμίζεται μέσω του Exterior Gateway Protocol (EGP).

Μέχρι το 2007 οι αριθμοί των ΑΣ αποτελούνταν από 16 bit, γεγονός το οποίο επέτρεπε 65.536 διαφορετικές τιμές ενώ στην συνέχεια επεκτάθηκαν σε 32 bit τιμές. Ο πρώτος και ο τελευταίος αριθμός των ακεραίων 16 bit, δηλαδή το 0 και το 65,535 και ο τελευταίος αριθμός των ακεραίων 32 bit, δηλαδή το 4,294,967,295 είναι κρατημένοι και δεν χρησιμοποιούνται. Επιπρόσθετα οι αριθμοί που βρίσκονται στο εύρος 64,512-65,534 (16 bit ΑΣ) και 4,200,000,000-4,294,967,294 (32 bit ΑΣ) είναι καταχωρημένοι για προσωπική χρήση (private use), επομένως, ενώ μπορούν να χρησιμοποιηθούν εσωτερικά, δεν ανακοινώνονται στο υπόλοιπο διαδίκτυο. Όλοι οι υπόλοιποι αριθμοί των ΑΣ εκχωρούνται από το Internet Assigned Numbers Authority (IANA).

Τα ΑΣ μπορούν να κατηγοριοποιηθούν σε 4 βασικές κατηγορίες:

• ***Υποστηρικτής πολλών δικτύων*** *(multihomed):* ΑΣ το οποίο διατηρεί συνδέσεις με περισσότερα από ένα ΑΣ έτσι ώστε να παραμείνει συνδεδεμένο στο διαδίκτυο ακόμα και αν κάποιο από αυτά «πέσει». Το συγκεκριμένο είδος ΑΣ δεν λειτουργεί ποτέ ως ενδιάμεσο για την διεκόλυνση μεταφοράς κίνησης μεταξύ δύο άλλων ΑΣ.

• ***Στέλεχος*** *(stub):* ΑΣ το οποίο διατηρεί σύνδεση με μονάχα άλλο ένα ΑΣ. Κάποια ΑΣ-stub μπορεί να έχουν και σχέσεις «peering» (θα εξηγηθούν παρακάτω) με άλλα ΑΣ οι οποίες κάποιες φορές δεν φαίνονται στους δημόσιους servers δρομολόγησης.

• ***Διαμετακομιστής*** *(transit):*ΑΣ το οποίο παρέχει συνδέσεις σε άλλα ΑΣ. Για παράδειγμα το δίκτυο Α μπορεί μέσω του δικτύου Β (transit AS) να επικοινωνήσει με το δίκτυο Γ. Όταν ένα ΑΣ είναι ο πάροχος υπηρεσιών Internet (ISP) ενός άλλου είναι transit-AS.

• ***Internet Exchange Point*** *(IXP):* ΑΣ το οποίο είναι μια φυσική υποδομή μέσω της οποίας οι πάροχοι υπηρεσιών Internet ανταλάσσουν κίνηση μεταξύ των ΑΣ τους. Συνήθως τα IXP-ΑΣ είναι διαφανή.

3.2 Σχέσεις Αυτόνομων Συστημάτων

Οι καθοριστικοί παράγοντες για τις σχέσεις που επικρατούν μεταξύ των ΑΣ σχετίζονται τόσο με τις τεχνικές όσο και με τις οικονομικές πτυχές της δομής του διαδικτύου. Γενικά οι επιχειρηματικές σχέσεις μεταξύ των πάροχων υπηρεσιών Internet μπορούν να είναι εξαιρετικά σύνθετες. Δεν υπάρχουν δημόσια διαθέσιμες πληροφορίες για τις σχέσεις μεταξύ των ΑΣ. Υπάρχουν καταχωρήσεις σχετικές με το ποιος διαχειρίζεται ποιο ΑΣ (π.χ. ARIN [10]) αλλά τα συμβόλαια μεταξύ των ISPs είναι ιδιόκτητα και οι εταιρείες αρνούνται να δώσουν τις σχετικές πολιτικές τους [11]. Οι περισσότερο πολυεμφανιζόμενες-βασικές σχέσεις είναι οι εξής:

• ***πελάτης-σε-πάροχο & πάροχος-σε-πελάτη*** *(customer-to-provider c2p | provider-to-customer p2c):* Ο πελάτης πληρώνει τον πάροχό του για να έχει την δυνατότητα σύνδεσης με το υπόλοιπο διαδίκτυο. Έτσι, ο πάροχος μεταφέρει κίνηση (transit) για όλους τους πελάτες του. Από την άλλη ο πελάτης δεν μεταφέρει κίνηση μεταξύ των παρόχων του.

• ***ομότιμος-σε-ομότιμο*** *(peer-to-peer p2p):* Στις σχέσεις αυτές τα ομότιμα ΑΣ δέχονται να ανταλάσσουν κίνηση μεταξύ των πελατών τους χωρίς χρέωση. Δεν μεταφέρουν όμως το ένα για το άλλο κίνηση προς τους παρόχους τους ή προς άλλα ΑΣ με τα οποία έχουν ομότιμο-σε ομότιμο σχέση.

• ***αδερφός-σε-αδερφό*** *(sibling-to-sibling s2s):* Στις σχέσεις αυτές τα ΑΣ μεταφέρουν κίνηση το ένα για το άλλο προς οπουδήποτε μπορούν.

Πρακτικά η μεταφορά της κίνησης από ένα ΑΣ για ένα άλλο αφορά την ανακοίνωση των γνωστών διαδρομών από το πρώτο στο δεύτερο. Έτσι, αν το ΑΣ Α θέλει να επικοινωνήσει με το Γ και το Β του ανακοινώσει μια διαδρομή που φτάνει στο Γ (η οποία προφανώς περνάει από το Β) , ουσιαστικά λέμε ότι το Β μεταφέρει την κίνηση για το Α.

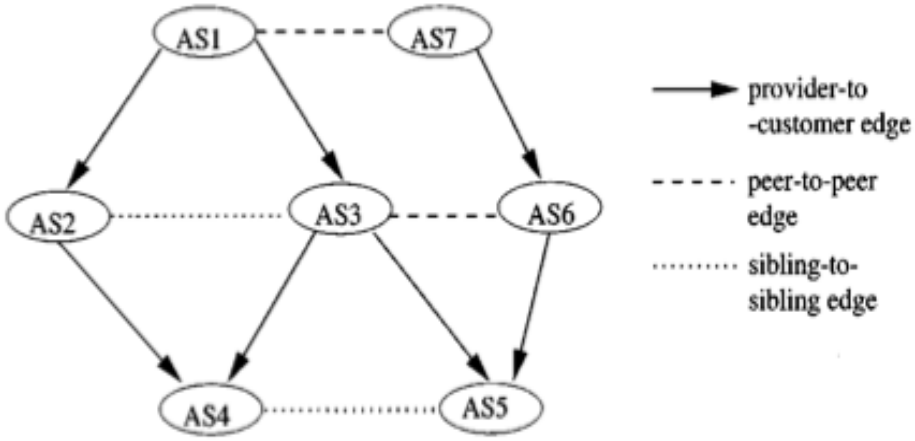
Κάθε ΑΣ διαλέγει την πολιτική που θα ακολουθήσει και μέσω του Border Gateway Protocol (BGP) επιλέγει ποιες διαδρομές θα ανακοινώνει στα υπόλοιπα ΑΣ, αναλόγως πάντοτε με τις σχέσεις που έχει με το καθένα.

Όπως αναφέρθηκε, οι πληροφορίες σχετικά με τις σχέσεις που επικρατούν μεταξύ των ΑΣ δεν δημοσιοποιούνται. Οι διάφοροι πάροχοι υπηρεσιών Internet αρνούνται

να μοιραστούν τις σχέσεις που έχουν με τους υπόλοιπους παρόχους και έτσι δεν δημοσιοποιούνται πληροφορίες που θα μπορούσαν να είναι εξαιρετικά χρήσιμες για την βελτίωση ολόκληρης της δομής του Internet. Η L.Gao για το σκοπό αυτό παρουσιάζει στο [12] έναν αλγόριθμο του οποίου ο σκοπός είναι να εντοπίζει τέτοιες σχέσεις μεταξύ των ΑΣ δεχόμενος ως είσοδο πίνακες δρομολόγησης BGP και με βάση τον οποίο εξάχθηκαν και τα δεδομένα που χρησιμοποιούνται στο 4^{ο} κεφάλαιο.

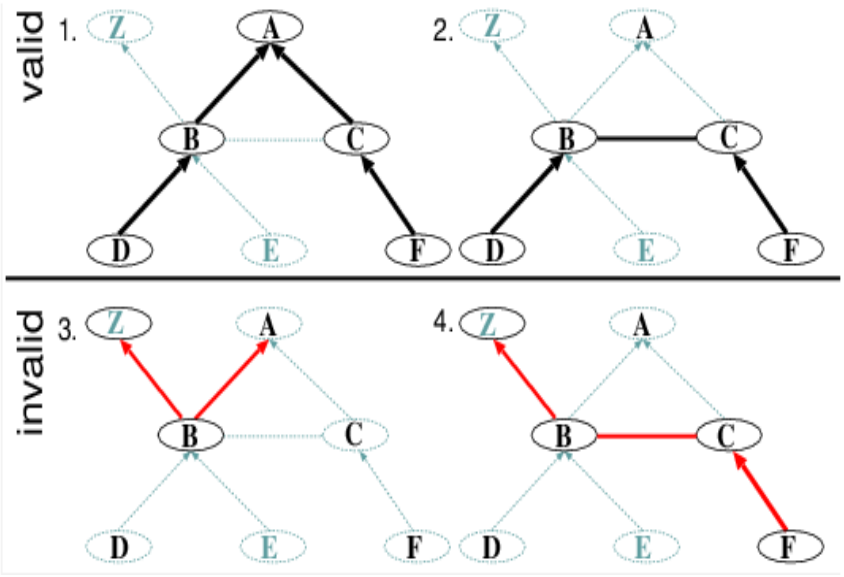
3.3 Γράφοι ΑΣ & Έγκυρα Μονοπάτια

Μελετώντας τις σχέσεις μεταξύ των ΑΣ γίνεται δυνατός ο σχεδιασμός ενός μερικώς κατευθυνόμενου γράφου, του οποίου οι κόμβοι αντιπροσωπεύουν τα ΑΣ, ενώ οι ακμές τις μεταξύ τους σχέσεις. Υπάρχουν δηλαδή οι εξής ακμές: πάροχος-σε-πελάτη (p2c), πελάτης-σε-πάροχο (c2p), ομότιμος-σε-ομότιμο (p2p) και αδερφός σε αδερφό (s2s). Οι κατευθυνόμενες γραμμές φυσικά είναι οι p2c και c2p ακμές. Το παρακάτω διάγραμμα αποτελεί ένα παράδειγμα των παραπάνω.



Σχήμα 3.1: Γράφος με τις σχέσεις μεταξύ των ΑΣ, Πηγή: [13]

Γνωρίζοντας την σημασία της κάθε ακμής, γίνεται εύκολα αντιληπτό το γεγονός ότι υπάρχουν συγκεκριμένα μονοπάτια από τα οποία μπορεί να περνάει κίνηση, τα οποία ονομάζονται έγκυρα μονοπάτια. Πρακτικά τα έγκυρα μονοπάτια είναι εκείνα για τα οποία για κάθε ΑΣ που μεταφέρει κίνηση για κάποιο άλλο, υπάρχει πληρωμή από το δεύτερο στο πρώτο. Στο παρακάτω σχήμα δίνονται 4 παραδείγματα:



Σχήμα 3.2: Τα 2 πάνω μονοπάτια (1,2) είναι έγκυρα ενώ τα δύο κάτω (3,4) είναι άκυρα, Πηγή: [13]

Στην **1η περίπτωση** τα A, B, και C είναι εκείνα που μεταφέρουν την κίνηση. Τα B και C πληρώνουν το A, ενώ τα D και F πληρώνουν τα B και C, αντίστοιχα.

Στην **2η περίπτωση** τα D και F πληρώνουν τα B και C αντίστοιχα.

Στην **3η περίπτωση** ενώ το B είναι το ΑΣ που μεταφέρει την κίνηση, όχι μόνο δεν πληρώνεται από κανέναν αλλά πληρώνει κιόλας τα Z και A.

Τέλος, στην **4η περίπτωση** κανείς δεν πληρώνει για την μεταφορά κίνησης τον B.

Η L.Gao πρότεινε έναν αλγόριθμο για την ανακάλυψη των σχέσεων μεταξύ των ΑΣ που αναλύεται στο [12] και έδωσε μια ακολουθία μοτίβων από τα οποία πρέπει να αποτελείται ένα μονοπάτι ώστε να είναι έγκυρο. Συγκεκριμένα παρέθεσε ότι ένα έγκυρο μονοπάτι ακολουθεί ένα από τα παρακάτω μοτίβα στα οποία το ανηφορικό μονοπάτι αποτελείται από μια ακολουθία ακμών οι οποίες είναι είτε c2p είτε s2s ακμές, ενώ το κατηφορικό μονοπάτι αποτελείται από είτε p2c είτε s2s ακμές.

- **Ένα ανηφορικό μονοπάτι (uphill path)**
- **Ένα κατηφορικό μονοπάτι (downhill path)**
- **Ένα ανηφορικό μονοπάτι που ακολουθείται από ένα κατηφορικό**
- **Ένα ανηφορικό μονοπάτι που ακολουθείται από μια p2p ακμή**
- **Μία p2p ακμή που ακολουθείται από ένα κατηφορικό μονοπάτι**
- **Ένα ανηφορικό μονοπάτι που ακολουθείται από μια p2p ακμή η οποία ακολουθείται από ένα κατηφορικό μονοπάτι**

Η γνώση όλων των παραπάνω είναι απαραίτητη για την κατανόηση του κεφαλαίου που ακολουθεί.

4 Κατασκευή και Απεικόνιση Κατανεμημένων Γράφων Επίθεσης από πραγματικά δεδομένα

Το παρόν κεφάλαιο ασχολείται με την κατασκευή γράφων επιθέσεων DDoS χρησιμοποιώντας πραγματικά δεδομένα. Η λογική έγκειται στο ότι στην περίπτωση που χρειάζεται να γίνει μια μελέτη μιας πραγματικής επίθεσης από την οποία είναι γνωστά το θύμα-ΑΣ, τα επιτιθέμενα-ΑΣ και οι σχέσεις μεταξύ των ΑΣ που περιγράφηκαν στο προηγούμενο κεφάλαιο είναι δυνατό να δημιουργηθεί και να εικονικοποιηθεί το δέντρο της επίθεσης αυτής έτσι ώστε να μελετηθούν κάποια περαιτέρω χαρακτηριστικά της επίθεσης. Στην συνέχεια χρησιμοποιώντας το δέντρο της επίθεσης κατασκευάζεται ο κατανεμημένος γράφος επίθεσης. Τι είναι όμως ένα δέντρο επίθεσης και τι ένας κατανεμημένος γράφος επίθεσης;

4.1 Σημασία Δεδομένων

Η δημιουργία του World Wide Web, το οποίο έχει κατασκευαστεί εδώ και περίπου 20 χρόνια, υπήρξε σαφέστατα ένα τεράστιο εξελικτικό βήμα για την τεχνολογία. Το διαδίκτυο προσπάθησε και κατάφερε να καταστήσει μια τεράστια ποσότητα πληροφορίας προσβάσιμη από σχεδόν όλους τους ανθρώπους. Πλέον ζούμε σε μια εποχή όπου η πληροφορία υπάρχει online και το ενδιαφέρον έγκειται στην επεξεργασία της. Όπως ο Tim-Berners-Lee ‘’We have to reframe the way we use the data’’.

Ο αριθμός των διαθέσιμων δεδομένων αυξάνεται εκθετικά και η χρήσιμη πληροφορία που περιέχεται μέσα στα δεδομένα αυτά, αν και αποτελεί πολύ μικρό ποσοστό της συνολικής πληροφορίας, μπορεί να έχει τεράστια σημασία. Φυσικά όσο περισσότερα τα δεδομένα, τόσο πιο κοντά στην αλήθεια θα είναι το συμπέρασμα που θα προκύψει.

Ένα όμορφο, αληθινό παράδειγμα, που δείχνει ξεκάθαρα την διαφορά της εξαγωγής συμπερασμάτων μέσω της επεξεργασίας μικρού όγκου δεδομένων έναντι της επεξεργασίας μεγάλου όγκου δεδομένων, δόθηκε από τον **Kenneth Cukier** στην ομιλία του με τίτλο « **Big data is better data ».** Το παράδειγμα αφορά μια μελέτη που είχε γίνει στην Αμερική με στόχο την εύρεση της πιο δημοφιλούς πίτας. Πραγματοποιήθηκε για τον σκοπό αυτό μια συλλογή δεδομένων από τις πωλήσεις των super market και προέκυψε ότι σε ένα ποσοστό εντυπωσιακά μεγάλο οι πρώτες σε πωλήσεις πίτες ήταν οι μηλόπιτες. Κανείς, λοιπόν, δεν μπορούσε να αμφισβητήσει το γεγονός ότι η μηλόπιτα είναι η αγαπημένη πίτα των Αμερικανών. Στην συνέχεια όμως, τα super μαρκετ άρχισαν να πουλάνε πίτες μικρότερου μεγέθους. Η έρευνα ξαναέγινε από την αρχή και έδειξε ότι πλέον η μηλόπιτα βρισκόταν στην $3^{η}$ – $4^{η}$ θέση. Ο λόγος που έγινε αυτό είναι ο εξής: Αρχικά που τα super market πουλούσαν πίτες διαμέτρου 30 cm (small data) μοιράζανε την πίτα ανάμεσα σε διάφορα άτομα λόγω του μεγέθους της και έτσι η επιλογή της πίτας γινόταν με βάση την λογική του κοινού παρονομαστή: ποια πίτα αρέσει σε όλους; Στην συνέχεια όμως που άρχισαν να πουλιούνται πίτες 11 cm (big data) ο καθένας μπορούσε να επιλέξει την προσωπική του αγαπημένη γεύση και έτσι τα αποτελέσματα προέκυψαν διαφορετικά.

Από αυτό το απλό παράδειγμα φαίνεται πώς ο όγκος των δεδομένων και η κατάλληλη επεξεργασία τους επηρεάζουν σημαντικά τα τελικά συμπεράσματα. Μία εξαιρετικά αποτελεσματική μέθοδος για την ανάλυση δεδομένων είναι μέσω της απεικόνισής τους (visualization).

4.2 Απεικόνιση Δεδομένων

Η απεικόνιση δεδομένων (data visualization) αφορά την εικονική ή γραφική αναπαράσταση ενός dataset. Είναι εξαιρετικά σημαντική διότι επιτρέπει στον εκάστοτε αναλυτή να μελετήσει τα δεδομένα με τέτοιο τρόπο ώστε να μπορέσει να κατανοήσει δυσνόητες εννοιες αλλά και να αναγνωρίσει μοτίβα που δεν θα μπορούσε να εντοπίσει αλλιώς.

Η γενική ιδέα της χρησιμοποίησης εικόνων για την καλύτερη αντίληψη των διαθέσιμων δεδομένων υπάρχει εδώ και αρκετούς αιώνες. Για παράδειγμα τον $17^{ο}$ χρησιμοποιούνταν χάρτες και διαφόρων ειδών γραφήματα ενώ στις αρχές του $18^{ου}$ αιώνα εφευρέθηκε το pie-chart. Μερικές δεκάδες χρόνια αργότερα, ο Charles Minard απεικόνισε την εισβολή του Ναπολέοντα στην Ρωσία. Έφτιαξε έναν χάρτη στον οποίο φαινόταν το μέγεθος του στρατού καθώς και η πορεία που χρησιμοποίησε ο Ναπολέων για την υποχώρησή του από την Μόσχα και συνδύασε στον χάρτη πληροφορίες για την θερμοκρασία και την χρονική αλληλουχία των γεγονότων για να αποκτήσει βαθύτερη κατανόηση του τι συνέβη.

Στην εποχή μας, η τεχνολογία, η οποία είναι και ο βασικός λόγος που η έννοια της απεικόνισης δεδομένων έχει γίνει τόσο γνωστή, καθιστά δυνατή την ταχύτατη απεικόνιση εξαρετικά μεγάλου όγκου δεδομένων. Ο κλάδος αυτός εξελίσσεται

συνεχώς και τα αποτελέσματα της εξέλιξης αυτής είναι πολλές φορές συναρπαστικά.

4.3 Δέντρα Επίθεσης & Διαθέσιμα Δεδομένα

Στην παρούσα εργασία Δέντρο Επίθεσης θεωρείται ένας συνεκτικός ακυκλικός γράφος (δέντρο στην θεωρία γράφων) ο οποίος έχει ως ρίζα το θύμα-ΑΣ και απαρτίζεται από έγκυρα μονοπάτια, σύμφωνα με τις σχέσεις μεταξύ των ΑΣ, από τα κακόβουλα-ΑΣ προς το θύμα.

Θεωρητικά η εύρεση ενός τέτοιου γράφου θα πραγματοποιούνταν στην εντέλεια αν είχαμε δεδομένα τέτοια ώστε να γνωρίζουμε από ποια routers πέρασαν τα επιτιθέμενα-πακέτα στην διαδρομή από τα κακόβουλα-ΑΣ μεχρι το θύμα-ΑΣ. Αν το γνωρίζαμε αυτό θα μπορούσαμε πολύ εύκολα να κατασκευάσουμε ένα δέντρο επίθεσης το οποίο ανταποκρίνεται πλήρως στην αληθινή επίθεση. Δεν θα χρειαζόταν να ασχοληθούμε καθόλου με τις σχέσεις μεταξύ των ΑΣ καθώς θα γνωρίζαμε την πραγματική διαδρομή των πακέτων, η οποία προφανώς θα έρχεται σε συμφωνία με τα έγκυρα μονοπάτια και απλώς θα μετατρέπαμε τις αληθινές δεδομένες διαδρομές σε ένα γράφο. Δυστυχώς, τα δεδομένα αυτά δεν υπάρχουν.

Τα δεδομένα που κατάφεραν να συγκεντρωθούν και βοηθάνε στον σχηματισμό ενός τέτοιου δέντρου αποτελούνται από ένα dataset που περιέχει τις σχέσεις μεταξύ των ΑΣ [13] και ένα dataset μιας πραγματικής επίθεσης DDoS που έγινε στις 4 Αυγούστου του 2007 [14]. Τα δεδομένα αυτά συλλέχθηκαν και δημοσιοποιήθηκαν από το Κέντρο Εφαρμοσμένης Ανάλυσης Δικτυακών Δεδομένων (Center of Applied Internet Data Analysis - CAIDA) το οποίο είναι μια συλλογική επιχείρηση που απαρτίζεται από οργανώσεις που δραστηριοποιούνται στον εμπορικό, κυβερνητικό και ερευνητικό τομέα με στόχο την διατήρηση μιας ισχυρής κλιμακούμενης παγκόσμιας υποδομής του διαδικτύου.

4.3.1 Δεδομένα σχέσεων Α.Σ.

Το dataset της CAIDA που περιέχει τις σχέσεις μεταξύ των ΑΣ είναι διαθέσιμο από το 2004 μέχρι σήμερα. Μέχρι το 2006 ανανεωνόταν με την προσθήκη ενός αρχείου ανά εβδομάδα. Τα τελευταία χρόνια ανανεώνεται με ένα αρχείο το μήνα. Από αυτά χρησιμοποιήθηκε το αρχείο που αναφερόταν σε μετρήσεις που έγιναν τον Αύγουστο του 2007 ώστε να βρίσκεται σε συμφωνία με το dataset της επίθεσης DDoS .

Κάθε αρχείο περιέχει έναν πλήρες γράφο ΑΣ ο οποίος προέρχεται από στιγμιότυπα πινάκων δρομολόγησης BGP (μέσω του Route Views project [15]) τα οποία λαμβάνονται ανά διαστήματα 8 ωρών σε περίοδο 5 ημερών. Τα βασικά βήματα για την δημιουργία ενός τέτοιου αρχείου είναι τα εξής:

1) **Απόσπαση των ΑΣ-ακμών από τα στιγμιότυπα RouteViews.**

2) **Εξαγωγή συμπερασμάτων για σχέσεις πελάτη-σε-πάροχο και καταγραφή αντίστοιχων ακμών.**

3) **Εξαγωγή συμπερασμάτων για σχέσεις ομότιμος-σε-ομότιμο, καταγραφή αντίστοιχων ακμών και πιθανή αλλαγή σε κάποιες από τις σχέσεις που εξαχθήκαν στο $2^{ο}$ βήμα.**

4) **Χρησιμοποίηση Heuristic μεθόδων για τον καθορισμό και την διόρθωση ύποπτων σχέσεων (π.χ. ένα χαμηλού βαθμού ΑΣ ενεργεί ως πάροχος σε ένα υψηλού βαθμού ΑΣ).**

Ένα $5^{ο}$ βήμα θα έπρεπε να αφορούσε την εξαγωγή συμπερασμάτων για σχέσεις αδελφών ΑΣ. Το συγκεκριμένο dataset παρ' ολ' αυτά περιέχει σχέσεις πελάτη-σε-πάροχο και ομότιμος-σε-ομότιμο ενώ δεν μας δίνει κάποια πληροφορία για τις σχέσεις αδελφός-σε-αδελφό. Περισσότερες λεπτομέρειες για τους αλγορίθμους που χρησιμοποιήθηκαν για την εξαγωγή των σχέσεων αυτών μπορούν να βρεθούν στα [13], [17].

Υποθέσεις για τα δεδομένα των σχέσεων Α.Σ.

Για την δημιουργία γράφων επιθέσεων DDoS που να ανταποκρίνονται σε αληθινά δεδομένα, το συγκεκριμένο dataset υπήρξε εξαιρετικά χρήσιμο. Σε γενικότερο πλαίσιο αν θεωρηθεί ότι έχουν βρεθεί οι σωστές σχέσεις μεταξύ όλων των ΑΣ υπάρχει η δυνατότητα να κατασκευαστεί ένας γράφος που να αναπαριστά απόλυτα ρεαλιστικά όλο το διαδίκτυο, σε επίπεδο ΑΣ.

Πρακτικά τα δεδομένα αυτά χρησιμοποιήθηκαν ως εξής: έχοντας μια λίστα με τα κακόβουλα-ΑΣ και γνωρίζοντας το θύμα-ΑΣ πραγματοποιήθηκε, χρησιμοποιώντας έναν αλγόριθμο που θα περιγραφεί στο επόμενο υποκεφάλαιο, η εύρεση του συντομότερου έγκυρου μονοπατιού από το κάθε κακόβουλο-ΑΣ προς το θύμα.

Το μελανό σημείο της διαδικασίας αυτής είναι ότι φυσικά δεν είναι εξασφαλισμένο ούτε ότι υπάρχει μοναδικό συντομότερο έγκυρο μονοπάτι μεταξύ ενός κακόβουλου-ΑΣ και του θύματος-ΑΣ ούτε ότι η κακόβουλη κίνηση θα ακολουθήσει απαραίτητα το συντομότερο μονοπάτι. Αυτό σημαίνει ότι οι διαδρομές οι οποίες έχουμε υποθέσει ότι ακολουθήσαν οι κακόβουλες ροές δεν ανταποκρίνονται απαραίτητα στις πραγματικές. Είναι διαδρομές οι οποίες θα μπορούσαν, σύμφωνα με τις σχέσεις μεταξύ των ΑΣ να υπάρξουν. Για την σίγουρη εύρεση των αληθινών διαδρομών που πραγματοποιήθηκαν θα έπρεπε να υπήρχε πρόσβαση σε έναν

τεράστιο αριθμό πινάκων δρομολόγησης, μετά την μελέτη των οποίων θα μπορούσαν με βεβαιότητα να εξαχθούν τα ζητούμενα συμπεράσματα αφού θα είχαμε τις αληθινές καταγραφές των σημείων από τα οποία περάσε η κακόβουλη ροή. Ωστόσο με τα παρόντα δεδομένα και στα πλαίσια της παρούσας διπλωματικής εργασίας έγινε μια όσο το δυνατόν ρεαλιστικότερη εκτίμηση των διαδρομών αυτών.

4.3.2 Δεδομένα κατανεμημένης επίθεσης άρνησης παροχής υπηρεσίας

Το dataset που χρησιμοποίηθηκε αφορά μία επίθεση DDoS που πραγματοποιήθηκε το 2007 και είναι συνολικού μεγέθους 21 GB. Η κίνηση αφορά μονάχα κακόβουλη κίνηση προς το θύμα και τις απαντήσεις του θύματος στους κακόβουλους χρήστες ενώ η μη κακόβουλη κίνηση ως επί το πλείστον έχει αφαιρεθεί.

Τα traces του dataset είναι ανωνυμοποιημένα με διατήρηση προθέματος σύμφωνα με το εργαλείο CryptoPAn (Cryptography-based Prefix-preserving Anonymization) [18]. Το εργαλείο αυτό βασίζεται σε τεχνικές που αναλύονται ενδελεχώς στα [19], [20] και [21] και έχει τις εξής ιδιότητες:

- **Μία-προς-μία** Υπάρχει μία προς μία αντιστοίχηση από τις αυθεντικές διευθύνσεις IP προς τις ανωνυμοποιημένες IP διευθύνσεις.

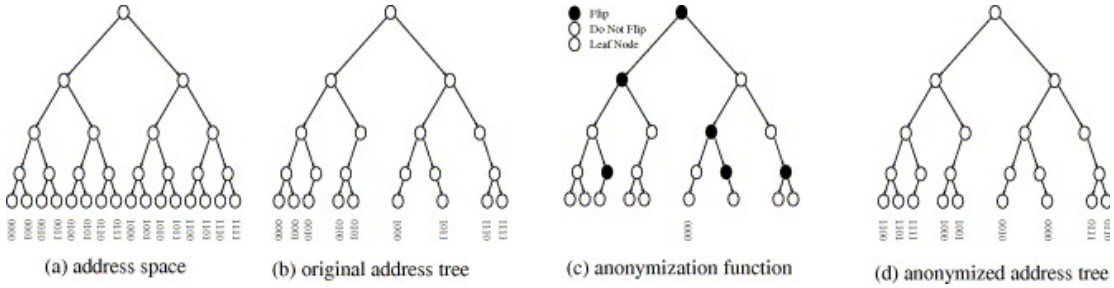
- **Διατήρηση προθέματος** Αν 2 αυθεντικές διευθύνσεις IP μοιράζονται ένα πρόθεμα k-bit, οι ανωνυμοποιημένες τους αντιστοιχήσεις θα μοιράζονται και αυτές ένα πρόθεμα k-bit.

- **Συνέπεια μεταξύ των traces** Η ίδια Ιδιεύθυνση IP σε διαφορετικά traces θα μετατραπεί στην ίδια ανωνυμοποιημένη διεύθυνση IP ακόμα και αν τα traces προέρχονται από διαφορετική τοποθεσία και ώρα.

- **Βασισμένο σε κρυπτογραφία** Οι ιδιοκτήτες των traces παρέχουν στο Crypto-PAn ένα μυστικό κλειδί και έτσι πρακτικά επιτυγχάνεται και η ανωνυμοποιημένη συνέπεια μεταξύ των διαφορετικών traces. Η κατασκευή του εργαλείου προστατεύει επίσης την μυστικότητα του κλειδιού.

Για να γίνει περισσότερο κατανοητή η έννοια της διατήρησης προθέματος, θα εξηγηθεί η μέθοδος που ακολουθείται για τον σκοπό αυτό, η οποία περιγράφεται και στο [11]. Καταρχάς να σημειωθεί ότι ολόκληρο το σύνολο των πιθανών διαφορετικών διευθύνσεων IPv4 μπορεί να αναπαρασταθεί ως ένα δυαδικό δέντρο ύψους 32. Ακολούθως, το σύνολο των διαφορετικών διευθύνσεων που περιέχεται σε ένα ανωνυμοποιημένο trace μπορεί να αναπαρασταθεί ως ένα υποδέντρο του συνολικού δέντρου όπου κάθε διεύθυνση αντιστοιχεί σε ένα φύλλο. Αυτό το ονομάζουμε αυθεντικό δέντρο διευθύνσεων. Κάθε κόμβος του δέντρου, εκτός της ρίζας, αντιπροσωπεύει ένα bit, η θέση του οποίου καθορίζεται από το ύψος του κόμβου, και η τιμή του οποίου καθορίζεται από την κατεύθυνση του κλάδου στον

οποίο βρίσκεται σε σχέση με τον κόμβο-πατέρα του. Στην εικόνα (a) του σχήματος 4.1 παρουσιάζεται ένα πλήρες δυαδικό δέντρο διευθύνσεων (με διευθύνσεις 4-bit για απλούστευση) ενώ στην εικόνα (b) παρουσιάζεται ένα αυθεντικό δέντρο διευθύνσεων.



Σχήμα 4.1: (a) πλήρες δυαδικό δέντρο διευθ., (b) αυθεντικό δέντρο διευθ., (c) συνάρτηση ανωνυμοποίησης, (d) ανωνυμοποιημένο δέντρο διευθ. Πηγή: [11]

Για την κατανόηση της μεθόδου διατήρησης προθέματος μπορεί κανείς να σκεφτεί ότι υπάρχει μια συνάρτηση ανωνυμοποίησης η οποία αντιστοιχεί μια δυαδική μεταβλητή σε κάθε κόμβο, εκτός των φύλλων. Η τιμή της κάθε τέτοιας μεταβλητής καθορίζει το αν η συνάρτηση ανωνυμοποίησης θα αναστρέψει το συγκεκριμένο bit (από 1 σε 0 ή από 0 σε 1) ή αν θα το αφήσει ως έχει. Με την εφαρμογή αυτής της συνάρτησης ανωνυμοποίησης στο αυθεντικό δέντρο διευθύνσεων δημιουργείται το ανωνυμοποιημένο δέντρο διευθύνσεων. Η εικόνα (c) δείχνει την συνάρτηση ανωνυμοποίησης ενώ η εικόνα (d) δείχνει το ανωνυμοποιημένο δέντρο διεθύνσεων που προκύπτει. Τέλος το [11] αποδεικνύει ότι αυτή είναι η μόνη μέθοδος για ανωνυμοποίηση με διατήρηση προθέματος.

Υποθέσεις για τα δεδομένα κατανεμημένης επίθεσης άρνησης παροχής υπηρεσίας

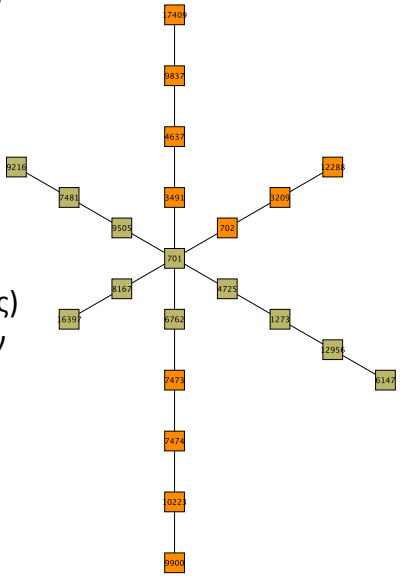
Φυσικά, η ανωνυμοποίηση των δεδομένων που μόλις περιγράφηκε δεν βοηθά καθόλου στην μελέτη που γίνεται στην παρούσα εργασία καθώς για την κατασκευή ενός γράφου επίθεσης χρειάζονται η γνώση της διεύθυνσης IP του πραγματικού θύματος και των IP διευθύνσεων των πραγματικών κακόβουλων-χρηστών.

Η συνέχεια της εργασίας βασίζεται στην υπόθεση ότι έχουμε τις αληθινές διευθύνσεις IP τόσο του θύματος όσο και των κακόβουλων χρηστών. Οι διευθύνσεις που χρησιμοποιήθηκαν είναι οι ανωνυμοποιημένες με διατήρηση προθέματος διευθύνσεις που βρέθηκαν στα δεδομένα της επίθεσης DDoS του 2007 και έτσι η διαδικασία που έγινε μπορεί να επαναληφθεί ακριβώς με τον ίδιο τρόπο για την κατασκευή γράφου επίθεσης μέσω ενός αρχείου pcap που περιέχει τα traces μιας επίθεσης με μη ανωνυμοποιημένες διευθύνσεις.

4.4 Διαδικασία Κατασκευής Κατανεμημένων Γράφων Επίθεσης

Στην ενότητα αυτή θα εξηγηθεί αναλυτικά η διαδικασία κατασκευής των κατανεμημένων γράφων επίθεσης. Κατανεμημένο γράφο επίθεσης ονομάζουμε το δέντρο επίθεσης του οποίου οι κόμβοι έχουν διαμοιραστεί σε ομάδες. Η λογική με την οποία πραγματοποιείται ο διαμοιρασμός βασίζεται στο ότι στόχος είναι να μοιραστούν οι κόμβοι του γράφου σε k-ομάδες (όπου το k ορίζεται από τον χρήστη) έτσι ώστε να ελαχιστοποιηθεί ο αριθμός των ακμών μεταξύ των κόμβων διαφορετικών ομάδων (minimum cut).

Ο σκοπός κατασκευής τέτοιων γράφων στην παρούσα εργασία έγκειται στο γεγονός ότι στην περίπτωση που ζητείται να εξομοιωθεί ένας γράφος, όσο μεγαλύτερος είναι ο αριθμός των κόμβων και των ακμών του, τόσο δυσκολότερο είναι να γίνει η εξομοίωση του σε ένα μοναδικό μηχάνημα. Ο κατανεμημένος γράφος επίθεσης παρέχει την πληροφορία του πώς μπορεί να «σπάσει» ο γράφος σε k-ομάδες όπου η κάθε μία από τις οποίες αντιπροσωπεύει το κομμάτι του γράφου που θα χτιστεί σε ένα από τα μηχανήματα. Για παράδειγμα, από τον κατανεμημένο γράφο επίθεσης του σχήματος 4.2 φαίνεται ότι στην περίπτωση που ήταν επιθυμητό να χτιστεί ο γράφος αυτός σε 2 μηχανήματα (που το καθένα θα αναλάβει σχεδόν ίδιο αριθμό από κόμβους) θα χωριζόταν στους πορτοκαλί κόμβους που θα χτιζόντουσαν στο ένα μηχάνημα και στους μπεζ που θα χτιζόντουσαν στο άλλο.



Σχήμα 4.2: Κατανεμημένος Γράφος Επίθεσης σε 2 μηχανήματα

4.4.1 Κατανομή IPs σε AS

Αρχικά, με την υπόθεση ότι έχουμε τις διευθύνσεις IP του θύματος και των κακόβουλων-χρηστών ενώ η επιθυμητή μελέτη πρέπει να είναι σε επίπεδο ΑΣ, έγινε η μετάφραση των διευθύνσεων IP στα ΑΣ στα οποία ανήκει η κάθε μία από αυτές .

Στο σημείο αυτό αξίζει να αναφερθεί ότι ένας συνηθισμένος τρόπος εύρεσης του ΑΣ μιας διεύθυνσης IP είναι μέσω του πρωτοκόλλου whois το οποίο ψάχνει στις

διάφορες βάσεις δεδομένων και επιστρέφει πληροφορίες όπως το domain name μιας IP ή το ΑΣ της. Στην συγκεκριμένη περίπτωση όμως που τα αρχεία επιθέσεων περιέχουν έναν τεράστιο αριθμό από διευθύνσεις IP, το να γίνει online το ψάξιμο των αριθμών των ΑΣ στα οποία ανήκουν θα καθυστερούσε υπερβολικά.

Για τον λόγο αυτό χρησιμοποιήθηκε ένα module extension της Python που ονομάζεται *pyasn* [22], [23]. Το εργαλείο αυτό κατασκευάστηκε και διατηρείται από μέλη της ερευνητικής ομάδας Economics of Cybersecurity του Delft University of Technology. Διαφέρει από τα υπόλοιπα εργαλεία εύρεσης αριθμών ΑΣ με την έννοια ότι παρέχει πληροφορίες offline από ιστορικά αρχεία [24] και έτσι είναι εξαιρετικά γρήγορο. Τα αρχεία αυτά βασίζονται σε πληροφορίες που παρέχονται από το Route-Views project [15]. Από αυτά στην προκειμένη περίπτωση επιλέχθηκε το αρχείο που αναφέρεται στον Αύγουστο του 2007, καθώς ταιριάζει με το αρχείο της επίθεσης DDoS που χρησιμοποιήθηκε.

Έτσι, κατασκευάστηκε τελικά ένα python script (*ip_to_as.py* [25])*,* το οποίο δεχόμενο ως είσοδο ένα αρχείο με διευθύνσεις IP -το ιστορικό αρχείο του 2007 που περιγράφηκε παραπάνω- και χρησιμοποιώντας το *pyasn,* δίνει ως έξοδο ένα αρχείο με τις αντιστοιχίσεις των IPs σε ΑΣ. Στην συνέχεια κατασκευάστηκε ένα δεύτερο python script (*mh.py* [26]) το οποίο δεχόμενο ως είσοδο το αρχείο με τις αντιστοιχίσεις των IPs σε ΑΣ δίνει ως έξοδο ένα αρχείο που περιέχει τα ΑΣ, που ουσιαστικά είναι τα κακόβουλα-ΑΣ της επίθεσης.

4.4.2 Αλγόριθμος Κατασκεύης Δέντρου Επίθεσης

Έχοντας το θύμα-ΑΣ, τα κακόβουλα-ΑΣ καθώς και τις σχέσεις που επικρατούν μεταξύ των ΑΣ είναι δυνατή η κατασκευή του δέντρου επίθεσης. Ο βασικός αλγόριθμος που χρησιμοποιήθηκε για τον σκοπό αυτό βρίσκεται στο script *tree_of_attack.py* [18] και υλοποιείται μέσω της συνάρτησης *bfs_as_relationships (G, victim, mh)*.

Το πρώτο πρόβλημα που έπρεπε να λυθεί για να είναι εφικτή αυτή η κατασκευή, ήταν να εκφραστούν με κάποιο τρόπο, σε μορφή γράφου, οι διαφορετικές σχέσεις που επικρατούν μεταξύ των ΑΣ. Το αρχείο που περιέχει τις σχέσεις των ΑΣ περιέχει γραμμές της μορφής ***Από-ΑΣ | Προς-ΑΣ | {0,-1}*** όπου στην περίπτωση που η τρίτη τιμή είναι **0** αυτό μεταφράζεται στο ότι η σχέση μεταξύ των 2 αυτών ΑΣ είναι ***ομότιμος-σε-ομότιμο*** ενώ στην περίπτωση που η τιμή αυτή είναι -1 αυτό μεταφράζεται στο ότι τα ***Από-ΑΣ, Προς-ΑΣ*** συνδέονται με την σχέση ***πάροχος-σε-πελάτη***.

Για την επίλυση του προβλήματος αυτού, το dataset με τις σχέσεις μεταξύ των ΑΣ μετατρέπεται σε έναν κατευθυνόμενο σταθμισμένο γράφο (directed weighted graph) μέσω της συνάρτησης *convert_to_weighted_edgelist(list)*. Αυτό που γίνεται πρακτικά μέσα στην συνάρτηση αυτή είναι ότι κάθε φορά που οι Α, Β συνδέονται με σχέση ***ομότιμος-σε-ομότιμο*** πραγματοποιείται η δημιουργία 2 ακμών με βάρος 0, μία από τον Α στον Β και μια από τον Β στον Α, ενώ κάθε φορά που οι Α, Β

συνδέονται με σχέση ***πάροχος-σε-πελάτη*** γίνεται η δημιουργία μιας ακμής με βάρος 1 από τον πάροχο προς τον πελάτη και μίας ακμής με βάρος 2 από τον πελάτη προς τον πάροχο. Με αυτόν τον τρόπο δίνεται η δυνατότητα επεξεργασίας του γράφου έχοντας τρόπο να ξεχωρίζουμε τις επικρατούσες σχέσεις μεταξύ των ΑΣ.

Η βασική συνάρτηση *bfs_as_relationships(G, victim, mh),* η οποία αναφέρθηκε και πρωτύτερα, αποτελεί μια επέκταση του BFS αλγόριθμου. Έχοντας τον συνολικό γράφο που εκφράζει τις σχέσεις μεταξύ των ΑΣ, ο στόχος πλέον είναι να βρεθεί ένα από τα συντομότερα έγκυρα μονοπάτια από το κάθε κακόβουλο-ΑΣ προς το θύμα-ΑΣ. Στο σημείο αυτό πρέπει να γίνει ξεκάθαρα αντιληπτό το γεγονός ότι ο κατευθυνόμενος σταθμισμένος γράφος που έχει δημιουργηθεί δεν μπορεί να αντιμετωπισθεί με τους γνωστούς αλγόριθμους συντομότερων μονοπατιών που γνωρίζουμε από την θεωρία γράφων. Ο λόγος που συμβαίνει αυτό είναι ότι αναλόγως με τις σχέσεις μεταξύ των ΑΣ, οι οποίες όπως αναφέρθηκε ξεχωρίζονται από τα βάρη των ακμών, υπάρχουν συγκεκριμένα μονοπάτια που μπορεί μια ροή πακέτων να ακολουθήσει (έγκυρα μονοπάτια) σύμφωνα με όσα έχουν αναφερθεί στο 3^{ο} κεφάλαιο και άλλα που δεν μπορεί. Έτσι, δεν αρκεί να υπάρχουν ακμές με κατεύθυνση βολική για την επικοινωνία μεταξύ 2 κόμβων, αλλά πρέπει και τα βάρη να παρέχουν μια ακολουθία αριθμών η οποία να εκφράζει έγκυρα μονοπάτια.

Συνοψίζοντας τους κανόνες που δίνονται στο 3^{ο} κεφάλαιο για τα έγκυρα μονοπάτια ισχύει ότι ένα έγκυρο μονοπάτι μπορεί να έχει οποιονδήποτε αριθμό από c2p ακμές (ακόμα και 0), εν συνεχεία να ακολουθείται από μία ή καμμία p2p ακμή και τέλος να έχει οποιονδήποτε αριθμό από p2c ακμές (ακόμα και 0). Στην περίπτωση που έχει σχηματιστεί ο κατευθυνόμενος σταθμισμένος γράφος η παραπάνω μετάφραση του c2p είναι ακμή με βάρος 2, του p2p είναι ακμή με βάρος 0 και του p2c είναι ακμή με βάρος 1. Για όσους νοιώθουν εξοικείωση με τις τυπικές γραμματικές, η τυπική γραμματική η οποία εκφράζει την ακολουθία βαρών των έγκυρων μονοπατιών είναι:

S → 2S | 0B | 1C | 0 | 1
B → 1C | 0 | 1
C → 0 | 1C

Δηλαδή οι έγκυρες ακολουθίες βαρών είναι: κανένα ή περισσότερα «2» ακολουθούμενα από κανένα ή ένα «0» ακολουθούμενο από κανένα ή περισσότερα «1».

Έτσι, ο γράφος διασχίζεται ξεκινώντας από κάθε κακόβουλο-ΑΣ με την λογική του αλγορίθμου BFS. Δηλαδή αρχικά γίνεται η εξερεύνηση των γειτόνων σε απόσταση «1», μετά η εξερεύνηση σε απόσταση «2» κ.ο.κ.. Ταυτόχρονα γίνεται έλεγχος με την εξερεύνηση κάθε κόμβου για το αν το μονοπάτι που έχει ακολουθηθεί μέχρι στιγμής είναι έγκυρο, δηλαδή για το αν τα βάρη των ακμών εκφράζουν μια έγκυρη ακολουθία. Επίσης, λόγω του ότι η απόσταση μεταξύ 2 οποιονδήποτε ΑΣ θα είναι μέχρι 6 βήματα (hops) [28], ο αλγόριθμος ψάχνει για το θύμα-ΑΣ σε απόσταση μικρότερη ή ίση από «6».

Με αυτόν τον τρόπο κατασκευάζεται έτσι το τελικό δέντρο επίθεσης το οποίο, όπως έχει προαναφερθεί, αποτελείται από το θύμα-ΑΣ που είναι και η ρίζα του δέντρου, και από τα έγκυρα μονοπάτια από κάθε κακόβουλο-ΑΣ προς το θύμα-ΑΣ.

Για την κατασκευή του δέντρου επίθεσης χρησιμοποιήθηκε σημαντικά το software package της Python ***NetworkX*** [29]. Η βιβλιοθήκη αυτή χρησιμοποιείται για την δημιουργία και την μελέτη σύνθετων δικτύων. Προσφέρει μια πληθώρα από γνωστές συναρτήσεις και επιτρέπει την επεξεργασία γράφων με εντυπωσιακά λειτουργικό τρόπο.

4.4.3 Μετατροπή Δέντρου Επίθεσης σε Κατανεμημένο Γράφο Επίθεσης

Στο σημείο αυτό έχοντας πλέον το δέντρο επίθεσης μπορεί να κατασκευαστεί ο κατανεμημένος γράφος επίθεσης. Όπως έχει προαναφερθεί, ο κατανεμημένος γράφος επίθεσης είναι πρακτικά το δέντρο επίθεσης το οποίο έχει τους κόμβους του χωρισμένους σε ομάδες (partitions), ή κάθε μία από τις οποίες εκφράζει το κομμάτι του γράφου που θα χτιστεί σε ένα εικονικό μηχάνημα. Ο αριθμός των partitions λοιπόν, αντιστοιχεί στον αριθμό των εικονικών μηχανημάτων στον οποίο επιθυμείται να μοιραστεί η συνολική τοπολογία.

Πρακτικά, το ζητούμενο είναι να διαμοιραστεί με τέτοιο τρόπο η συνολική τοπολογία ώστε οι ταχύτητες της αποστολής και λήψης δεδομένων μεταξύ των εικονικών μηχανημάτων να είναι όσο το δυνατόν μεγαλύτερες. Η ανταλλαγή δεδομένων σε υψηλές ταχύτητες εξυπηρετεί ως προς το ότι παρέχει την δυνατότητα στον ερευνητή που χρησιμοποιεί τον εξομοιωτή να ρυθμίσει, ανάλογα με τις ανάγκες του, τον ρυθμό ανταλλαγής πακέτων. Όσο υψηλότερο είναι το όριο της μέγιστης αυτής ταχύτητας τόσο υψηλότερα είνα τα όρια του εξομοιωτή.

Επιπρόσθετα, είναι προφανές ότι οι κόμβοι ενός δικτύου το οποίο εξομοιώνεται μέσω του Mininet σε **ένα** εικονικό μηχάνημα μπορούν να ανταλλάσσουν κίνηση με εξαιρετικά μεγάλη ταχύτητα, μεγαλύτερη από την ταχύτητα με την οποία ανταλλάσσουν κίνηση οι κόμβοι ενός αληθινού δικτύου. Επομένως όσο η τοπολογία που επιθυμείται να εξομοιωθεί κατασκευάζεται μέσα σε **ένα** εικονικό μηχάνημα θα είναι δυνατό να «παιχτεί» οποιοδήποτε ρεαλιστικό σενάριο.

Έτσι, αυτό που πρέπει να επιτευχθεί με τον διαμοιρασμό του γράφου είναι πλέον ευκόλως εννοούμενο. Εχει αποφασιστεί ότι ο συνολικός γράφος είναι μεγέθους τέτοιου που δεν επιτρέπει την εξομοίωσή του σε ένα μοναδικό εικονικό μηχάνημα. Ταυτόχρονα όσο η κίνηση βρίσκεται εντός ενός εικονικού μηχανήματος τα περιθώρια αύξησης της ταχύτητας αποστολής-λήψης πακέτων είναι ευρύτερα από όσο χρειάζεται. Επομένως, ο διαμοιρασμός πρέπει να γίνει με τρόπο τέτοιο ώστε **ο αριθμός των ακμών που συνδέουν τα διαφορετικά εικονικά μηχανήματα να είναι όσο το δυνατόν μικρότερος**.

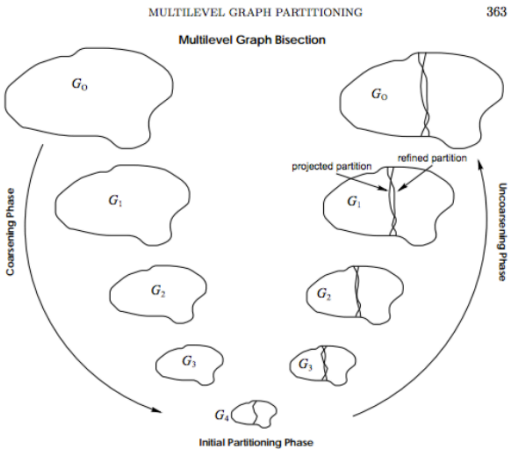
Το πρόβλημα της διαίρεσης των γράφων (graph partitioning) ανήκει στην κατηγορία των NP-complete προβλημάτων. Αυτό σημαίνει ότι δεν μπορεί να λυθεί σε πολυωνυμικό χρόνο. Παρόλαυτά έχουν αναπτυχθεί διάφοροι αλγόριθμοι που βρίσκουν πώς να διαιρέσουν γράφους αρκετά καλά και σε λογικό χρόνο. Σημαντική πηγή για τα όσα αναφέρονται παρακάτω είναι το [30].

Ένας τρόπος προσέγγισης του προβλήματος είναι μέσω **μεθόδων φασματικής διαίρεσης (spectral partitioning methods)** οι οποίοι χρησιμοποιούνται αρκετά. Αυτές οι μέθοδοι όμως βασίζονται στον υπολογισμό ιδιοδιανυσμάτων και έτσι έχουν αρκετά υψηλό χρόνο εκτέλεσης.

Μία άλλη προσέγγιση αφορά τεχνικές διαίρεσης που βασίζονται στην **γεωμετρία του γράφου** τον οποίο αναλύουν. Οι αλγόριθμοι αυτού του είδους συνηθίζουν να είναι γρηγορότεροι αλλά τα αποτελέσματα που δίνουν πολλές φορές είναι χειρότερα από τα αποτελέσματα των φασματικών μεθόδων. Για να έχουν καλύτερα αποτελέσματα μπορεί να χρειαστεί να «τρέξουν» αρκετές φορές, γεγονός το οποίο και αυξάνει σημαντικά τον συνολικό χρόνο επίλυσης.

Μια τρίτη προσέγγιση αφορά τους **αλγορίθμους διαμοιρασμού γράφων σε πολλά επίπεδα (multilevel graph partitioning schemes).** Η βασική λογική πίσω από αυτού του τύπου τους αλγορίθμους έγκειται στο ότι ο διαμοιρασμός του γράφου γίνεται σε στάδια. Αρχικά αφαιρούνται κόμβοι και ακμές από τον συνολικό γράφο ώστε να σχηματιστεί ένας μικρότερος γράφος. Γίνεται ο διαμοιρασμός του και στην συνέχεια προστίθεται σε αυτόν ένα ακόμα κομμάτι του αρχικού γράφου οπότε και ξαναδιαμοιράζεται, ο μεγαλύτερος τώρα, υπογράφος. Η διαδικασία αυτή συνεχίζει μέχρι να έχει διαμοιραστεί ολόκληρος ο συνολικός γράφος.

Πιο συγκεκριμένα στην περίπτωση που το πρόβλημα είναι να διαμοιραστεί ο γράφος σε ένα συγκεκριμένο αριθμό από partitions (k-way partition problem) είναι σύνηθες να χρησιμοποιούνται τεχνικές αναδρομικής διχοτόμησης. Οι τεχνικές αυτές ανήκουν στην κατηγορία των αλγορίθμων διαμοιρασμού γράφων σε πολλά επίπεδα. Η λογική της διχοτόμησης σε πολλά επίπεδα παρουσιάζεται ξεκάθαρα στο παρακάτω σχήμα:



Σχήμα 4.3 Διαίρεση γράφου σε πολλά επίπεδα

Πρακτικά ένας αλγόριθμος διχοτόμησης πολλών επιπέδων (A multilevel graph bisection algorithm) περιέχει 3 φάσεις. Υποθέτοντας τον σταθμισμένο γράφο G_0 = (V_0, E_0), του σχήματος *4.3.3.1*, με βάρη τόσο στις ακμές όσο και στους κόμβους του έχουμε τα εξής:

- **Coarsening phase:** Ο γράφος G_0 μετατρέπεται σε μια ακολουθία μικρότερων γράφων G_1, G_m, … , G_m ώστε $|V_0| > |V_1| > \ldots > |V_m|$

- **Partitioning phase:** Ο γράφος $G_m = (V_m, E_m)$διαιρείται σε 2 partition (P_m) με το καθένα να περιέχει τις μισές ακμές του G_m

- **Uncoarsening phase:** Ο διαμοιρασμός P_m του G_m προβάλλεται μέσω ενδιάμεσων διαμοιρασμών (έναν για κάθε στάδιο) τελικά στον αρχικό γράφο G_0. Σε κάθε τέτοιο στάδιο δηλαδή υπάρχει η προβολή του προηγούμενου partition (του μικρότερου υπογράφου) στον ένα στάδιο μεγαλύτερο υπογράφο, με βάση αυτή δημιουργείται το καινούργιο partition του μεγαλύτερου υπογράφου κ.ο.κ μέχρι να διχοτομηθεί ο G_0.

Για το σκοπό αυτό χρησιμοποιήθηκε η βιβλιοθήκη *METIS* [31] η οποία αποτελείται από μία σειρά προγραμμάτων ειδικά για *graph partitioning.* Χρησιμοποιήθηκαν αλγόριθμοι που βασίζονται σε τεχνικές αναδρομικής διχοτόμησης πολλών επιπέδων. Τα πλεονεκτήματα του εργαλείου αυτού είναι τα εξής:

- **Εξαιρετικά υψηλή ταχύτητα**

- **Υψηλής ποιότητας partitions:** Τα partitions που παράγονται από το METIS είναι 10% - 50% καλύτερα από ότι αυτά που παράγονται από τους φασματικούς αλγορίθμους διαμοιρασμού.

4.5 Απεικόνιση Κατανεμημένων Γράφων Επίθεσης

Στην ενότητα αυτή θα παρουσιαστούν μερικοί από τους κατανεμημένους γράφους επίθεσης που απεικονίστηκαν και θα επεξηγηθούν τα βήματα τα οποία πρέπει να ακολουθήσει οποιοσδήποτε θέλει να χρησιμοποιήσει τον κώδικα που γράφτηκε για την κατασκευή και απεικόνιση αντίστοιχων γράφων με βάση δικά του δεδομένα.

4.5.1 Διαδικασία εκτέλεσης κώδικα

Δέντρο της επίθεσης - tree_of_attack.py

Όπως έχει προαναφερθεί το βασικό script που χρησιμοποιήθηκε για την δημιουργία των δέντρων επίθεσης είναι το ***tree_of_attack.py*** [27]. Αυτό, δέχεται ως είσοδο το dataset που περιέχει της σχέσεις μεταξύ των ΑΣ το οποίο είναι ένα ***txt*** αρχείο που αποτελείται από γραμμές της μορφής *Από ΑΣ | Προς ΑΣ | {0,-1}* η ερμηνεία των οποίων εξηγήθηκε στην ενότητα **4.4.2** καθώς και ένα ακόμα **txt** αρχείο που περιέχει έναν αριθμό κακόβουλου ΑΣ ανά γραμμή.

Στην συνέχεια υπάρχει ένας βρόχος for (for-loop) όπου ο αριθμός επανάληψής του καθορίζει πόσα από τα κακόβουλα ΑΣ είναι επιθυμητό να συμπεριληφθούν στο δέντρο επίθεσης.

Έπειτα, γίνεται η μετατροπή των δεδομένων που περιέχουν τις σχέσεις μεταξύ των ΑΣ σε σταθμισμένο γράφο (συνάρτηση ***convert_to_weighted_edgelist****)* υπό μορφή edgelist και ακολουθεί η κατασκευή του γράφου αυτού με την βοήθεια του networkX που έχει επίσης αναφερθεί στην ενότητα **4.4.2**.

Τέλος, καλείται η συνάρτηση **bfs_as_relationships(AS_graph, victim-AS, mh-AS)** η οποία έχει εξηγηθεί στην ενότητα **4.4.2** μετά την εκτέλεση της οποίας έχουν βρεθεί τα μονοπάτια που σχηματίζουν το δέντρο επίθεσης τα οποία και τυπώνονται.

Η εκτέλεση των παραπάνω γίνεται με την εντολή:

$ python tree_of_attack.py AS.txt MH.txt

Διαμοιρασμός Δέντρου Επίθεσης

Ο διαμοιρασμός του γράφου, δηλαδή η μετατροπή του δέντρου επίθεσης σε κατανεμημένο γράφο επίθεσης όπως έχει εξηγηθεί παραπάνω, γίνεται μέσω του

from_paths_to_vms.py [32]. Η εντολή ώστε να προκύψουν τα όσα παρουσιάζονται παρακάτω είναι:

$ python from_paths_to_vms.py paths.txt #(number of vms) MH.txt

Το script αυτό δέχεται ως **είσοδο** το δέντρο της επίθεσης σε μορφή ενός **txt** αρχείου που αποτελείται από μια λίστα που περιέχει τα μονοπάτια του (όπως τυπώθηκαν από το ***tree_of_attack.py*** που περιγράφηκε πρωτύτερα), τον αριθμό των partitions στα οποία επιθυμεί ο χρήστης να διαμοιράσει τον γράφο και ένα ακόμα αρχείο **txt** που περιέχει τους αριθμούς των κακόβουλων ΑΣ.

Ως **έξοδο** δίνει ένα αρχείο που ονομάζεται ***new_edges.txt*** το οποίο απότελείται από μια λίστα από τούπλες 4 στοιχείων στην κάθε μία εκ των οποίων είναι ως εξής: ***(ΑΣ-1, ΑΣ-2, Group-1, Group-2)***

Το παραπάνω σημαίνει ότι υπάρχει μια ακμή μεταξύ των ΑΣ-1 και ΑΣ-2 και το ΑΣ-1 έχει μοιραστεί στην ομάδα Group-1 ενώ το ΑΣ-2 έχει μοιραστεί στην ομάδα Group-2. Προφανώς, στην περίπτωση που και τα 2 ΑΣ ανήκουν στο ίδιο partition μετά τον διαμοιρασμό το Group-1 θα είναι το ίδιο με το Group-2. Ένα παράδειγμα του περιεχομένου ενός τέτοιο αρχείου είναι : ***[(9505, 701, 0, 0), (3491, 701, 1, 0), (9900, 10223, 1, 1)]*** όπου το ΑΣ 9505 ανήκει στο partition 0 και συνδέεται με το ΑΣ 701 το οποίο ανήκει και αυτό στο partition 0. Με αντίστοιχη λογική, το ΑΣ 3491 ανήκει στο partition 1 και συνδέεται με το ΑΣ 701 (που όπως αναφέρθηκε προηγουμένως ανήκει στο partition 0) και τέλος το ΑΣ 9900 ανήκει στο partition 1 και συνδέεται με το ΑΣ 10223 το οποίο ανήκει επίσης στο partition 1.

Δίνει επίσης ως **έξοδο** τα αρχεία ***new._edges.dot, new_edges.graphml*** τα οποία αφορούν την απεικόνιση του κατανεμημένου γράφου επίθεσης και εξηγούνται στην συνέχεια.

Διαδικασία Απεικόνισης

Για την απεικόνιση του γράφου χρησιμοποιείται και πάλι το εργαλείο networkX [29]. Αφού έχει γίνει πλέον το partition και υπάρχει το ***new_edges.txt*** που περιγράφηκε πρωτύτερα και περιέχει την πληροφορία για τον τρόπο με τον οποίο θα μοιραστούν οι ΑΣ-κόμβοι στα εικονικά μηχανήματα, γίνεται ο χρωματισμός του γράφου ανάλογα με τον τρόπο που είναι διαμοιρασμένος. Έτσι κάθε κόμβος παίρνει ένα χρώμα σύμφωνα με το partition στο οποίο ανήκει. Επιπλέον υπάρχει η επιλογή οι ΑΣ-κόμβοι που ανήκουν στην λίστα με τα κακόβουλα-ΑΣ να χρωματισθούν κόκκινοι στην περίπτωση που αυτή είναι η πληροφορία που μας ενδιαφέρει.

Μετά από αυτήν την διαδικασία, ο κατανεμημένος γράφος επίθεσης γράφεται σε ένα αρχείο σε μορφή **dot** file. Τα .**dot** αρχεία είναι αρκετά δημοφιλή και χρησιμοποιούνται για την περιγραφή γράφων. Χαρακτηριστικό τους επίσης είναι ότι περιέχουν κείμενο το οποίο μπορεί να αναγνωστεί και από ανθρώπους και από προγράμματα.

Το ***graphml*** δημιουργείται με την μετατροπή του **dot** αρχείου μέσω του εργαλείου ***dottoxml.py*** [33]. Η μετατροπή αυτή πραγματοποιήθηκε για να είναι εφικτό να χρησιμοποιηθεί το πρόγραμμα επεξεργασίας γράφων (Graph Editor) **yEd.**

Το **yEd** επιτρέπει την εισαγωγή δεδομένων από αρχεία τύπου **.graphml** και παρέχει ένα μεγάλο εύρος επιλογών ως προς την επιλογή της μορφής με την οποία θα παρουσιαστούν τα δεδομένα (tree, circular κτλ). Εκτός από το γεγονός ότι το εργαλείο αυτό είναι ταχύτατο και παράγει υψηλής ποιότητας γράφους, επιλέχθηκε επίσης λόγω του ότι είναι διαθέσιμο δωρεάν. Αυτό είναι αρκετά σημαντικό ώστε ο καθένας να έχει την δυνατότητα να δουλέψει με τα εργαλεία που παρουσίαζονται στην παρούσα διπλωματική και να μπορεί να ακολουθήσει όλα τα βήματα, ένα προς ένα.

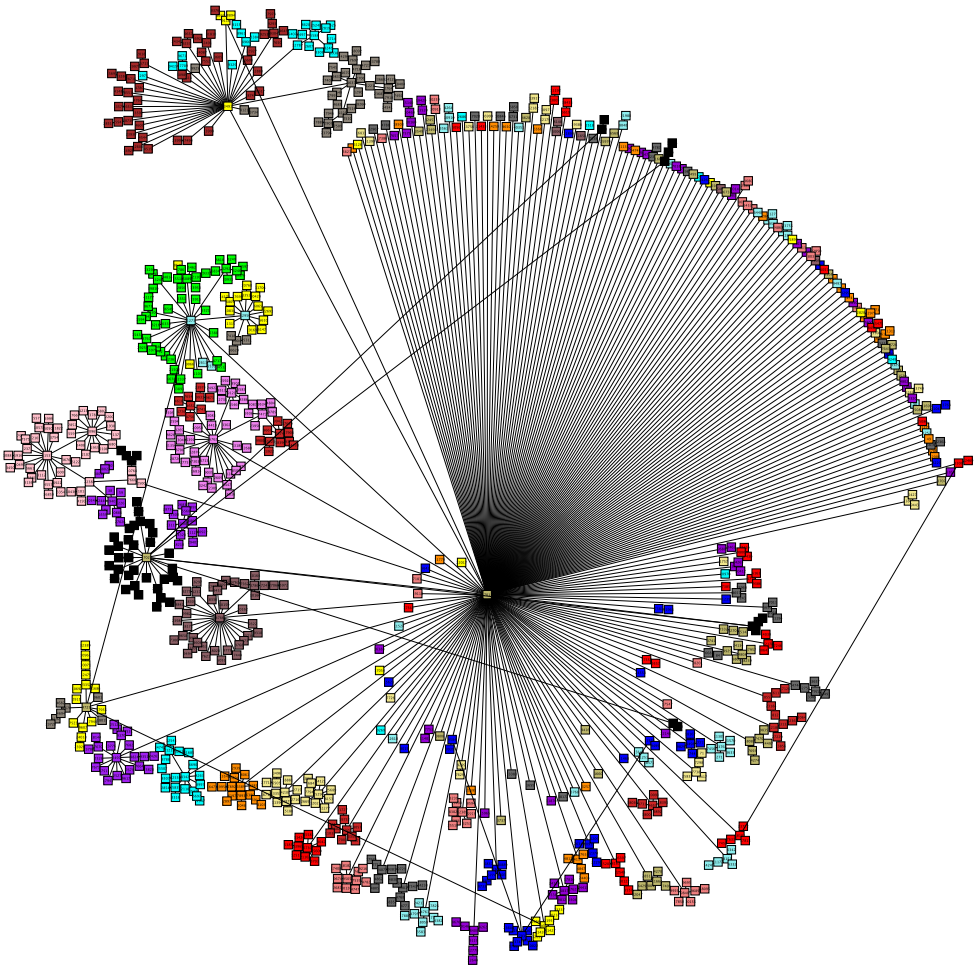
4.5.2 Απεικόνιση Κατανεμημένων Γράφων Επίθεσης

Παρακάτω παρουσιάζονται κάποιοι από τους κατανεμημένους γράφους επίθεσης που δημιοργήθηκαν με την μεθοδολογία που έχει περιγραφεί.Το σχήμα που ακολουθεί αφορά ένα κατανεμημένο γράφο επίθεσης ο οποίος αποτελείται από σχεδόν 1000 ΑΣ-κόμβους και είναι χωρισμένος σε 20 partitions.

Όπως πρέπει να έχει γίνει ήδη αντιληπτό, ο κάθε κόμβος αντιπροσωπεύει ένα ΑΣ και το κάθε χρώμα συμβολίζει το εκάστοτε VM στο οποίο θα χτιστεί ο αντίστοιχος ΑΣ-κόμβος. Έτσι, για παράδειγμα, όλοι οι ροζ κόμβοι θα χτιστούν στο ίδιο VM ενώ όλοι οι γαλάζιοι σε άλλο, κ.ο.κ.

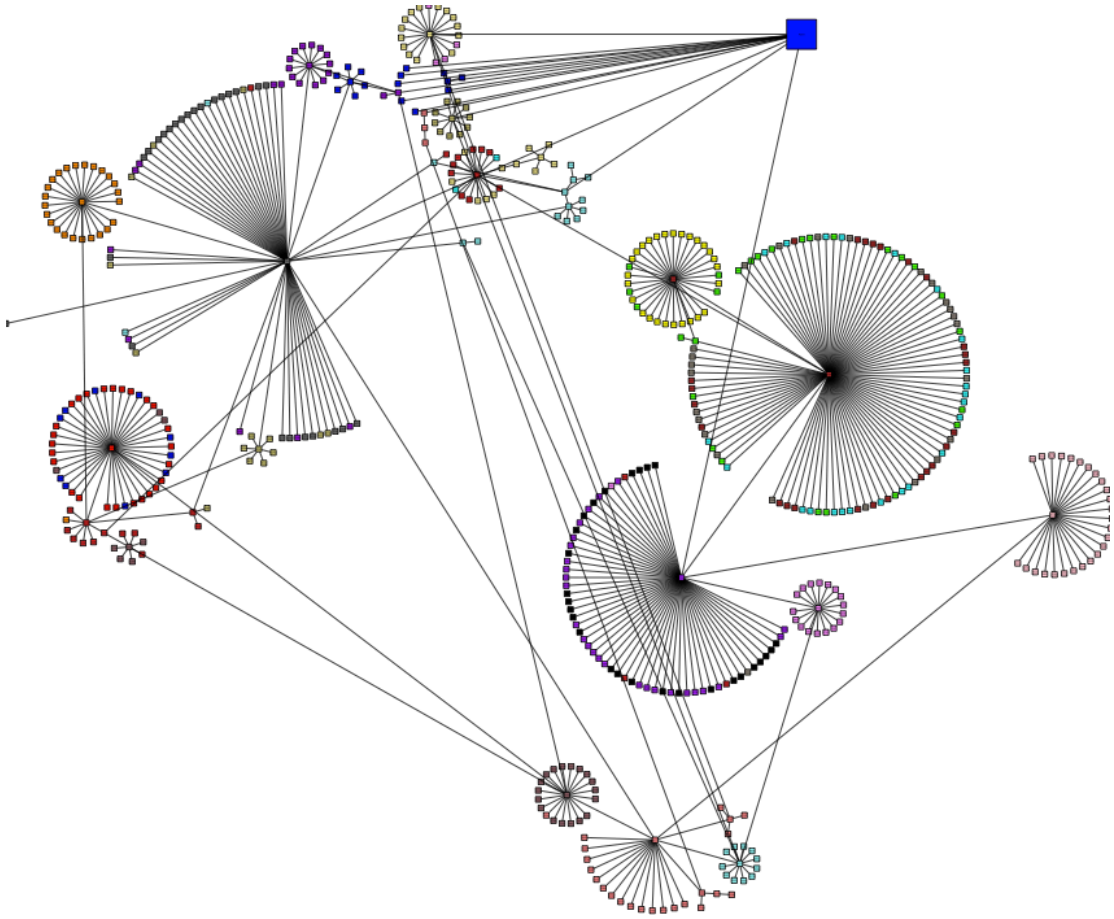
Για το κάθε σχήμα έχει επιλεχθεί διαφορετική λογική στην κατανομή των κόμβων για να τονισθεί τόσο η ευελιξία του εργαλείου που χρησιμοποιήθηκε, όσο και το πώς αλλάζει η πληροφορία που δέχεται ο εκάστοτε ερευνητής όταν αλλάζει ο τρόπος παρουσίασης του γράφου.

Το παρακάτω σχήμα απαρτίζεται από 1000 κόμβους, είναι διαμοιρασμένο σε 20 partitions και το ΑΣ-θύμα βρίσκεται κέντρο. Εδώ φαίνεται ότι το θύμα δεν είναι τόσο κεντρικό όσο το προηγούμενο, λιγότερα ΑΣ δηλαδή έχουν απευθείας σύνδεση με αυτό.



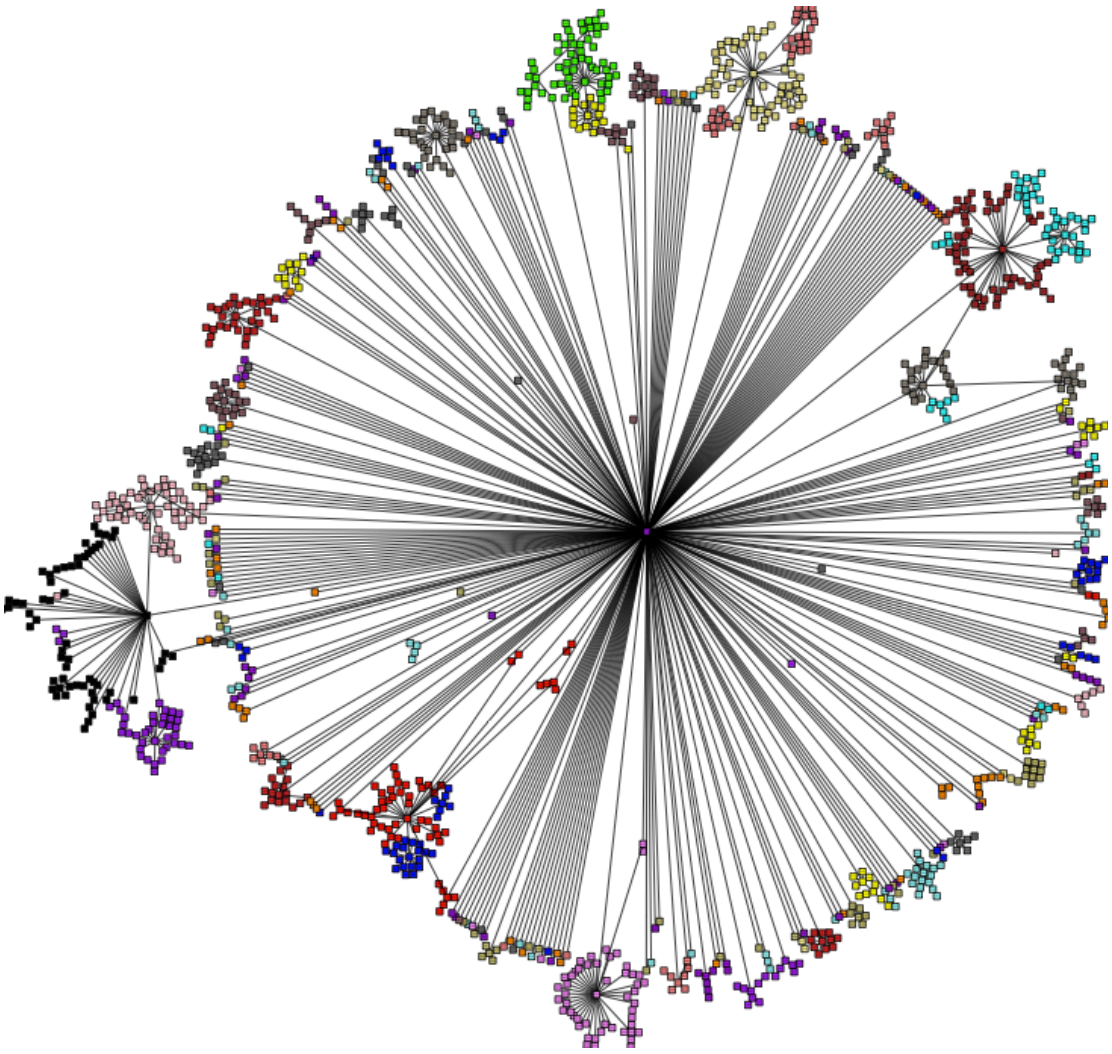
Σχήμα 4.4: Γράφος 1000 κόμβων χωρισμένος σε 20 partitions

Το παρακάτω σχήμα απαρτίζεται από 500 κόμβους και το ΑΣ-θύμα βρίσκεται πάνω δεξιά (μεγάλο μπλε τετράγωνο). Εδώ φαίνεται ότι το θύμα δεν είναι τόσο κεντρικό όσο το προηγούμενο, λιγότερα ΑΣ δηλαδή έχουν απευθείας σύνδεση με αυτό.



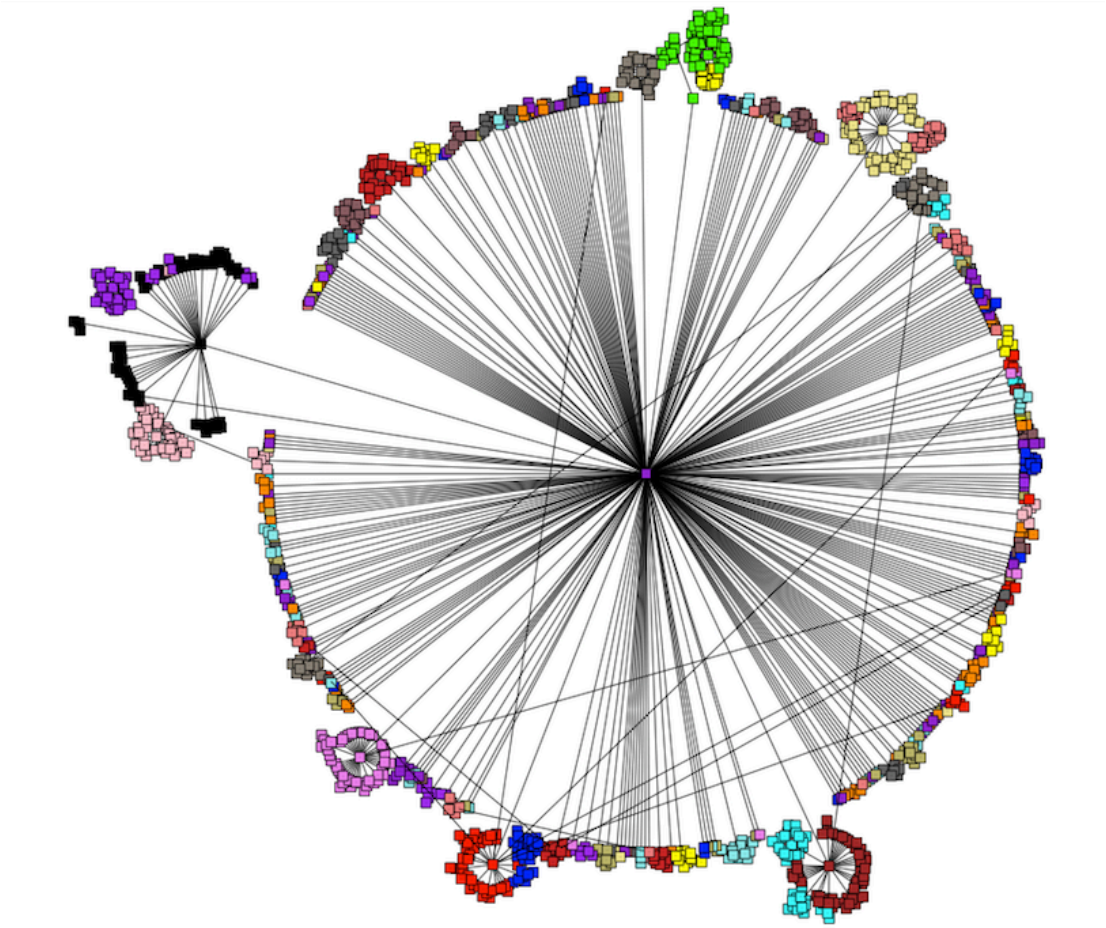
Σχήμα 4.5: Γράφος 500 κόμβων – Το ΑΣ-θύμα βρίσκεται πάνω δεξιά

Το επόμενο σχήμα αποτελείται από 1188 κόμβους οι οποίοι και πάλι είναι χωρισμένοι σε 20 partitions. Παρουσιάζει αρκετό ενδιαφέρον η μορφή αυτή, καθώς φαίνονται εύκολα τα βασικά ΑΣ μέσω των οποίων πολλά άλλα ΑΣ στελνουν την κίνηση στο ΑΣ-θύμα. Το ΑΣ-θύμα βρίσκεται και πάλι στο κέντρο του σχήματος.



Σχήμα 4.6: Γράφος 1188 κόμβων – Το ΑΣ θύμα βρίσκεται στο κέντρο

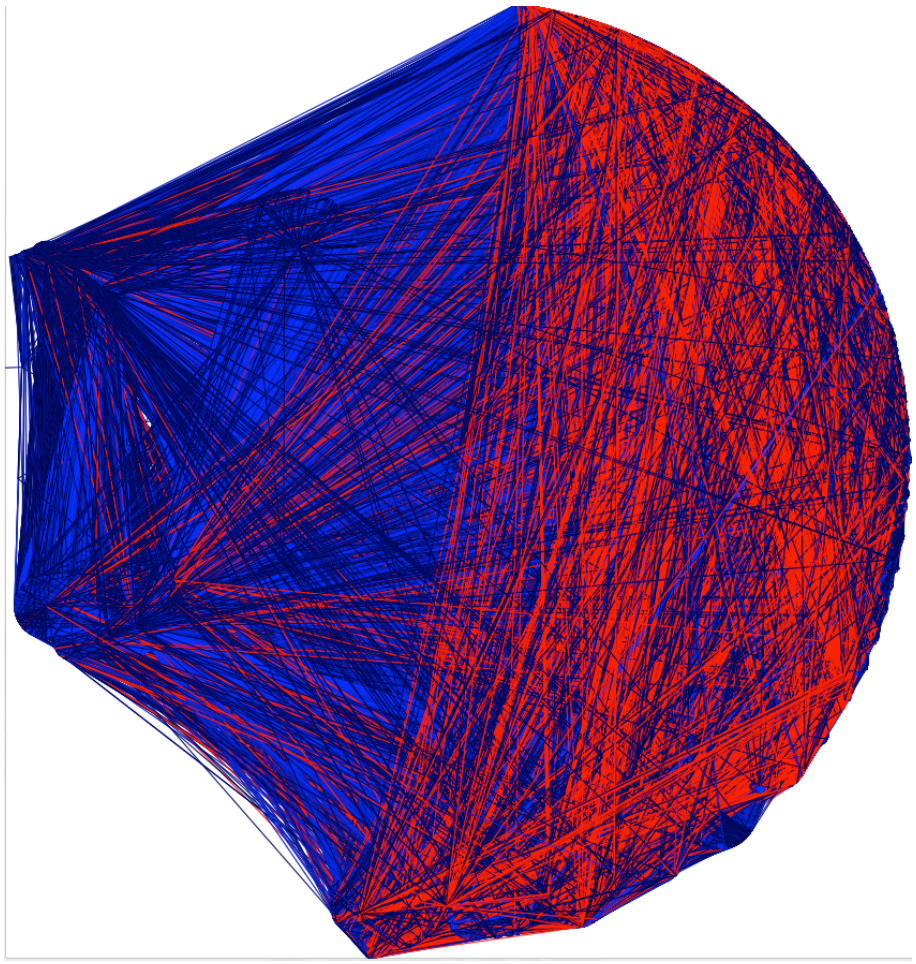
Αντίστοιχα με το προηγούμενο, αλλά με μια μεγαλύτερη έμφαση στα χρώματα, προκύπτει και το σχήμα αυτό:



Σχήμα 4.7: Γράφος 1188 κόμβων – Το ΑΣ θύμα βρίσκεται στο κέντρο

Από όλα τα παραπάνω σχήματα φαίνεται πόσο ενδιαφέρουσες πληροφορίες μπορούμε να εξάγουμε με την απεικόνιση τέτοιων γράφων επίθεσης. Το γεγονός ότι μπορούν εύκολα μέσω της απεικόνισης να εντοπιστούν τα ΑΣ, μέσω των οποίων διοχετεύεται κίνηση που αφορά την επίθεση από πολλά άλλα ΑΣ, ίσως δίνει την δυνατότητα της δημιουργίας τεχνικών άμυνας οι οποίες θα βασίζονται στην επικοινωνία μεταξύ των «σημαντικών» αυτών κόμβων για την ανίχνευση της επίθεσης. Οι τεχνικές άμυνας ξεφεύγουν από τα πλαίσια της διπλωματικής αυτής αλλά εκτιμάται ότι μέσω τέτοιων απεικονίσεων θα είναι εφικτό να γίνει σοβαρή εργασία πάνω στην άμυνα κατά των επιθέσεων DDoS.

Για το κλείσιμο του κεφαλάιου αυτού, και απλώς από ενδιαφέρον, παρουσιάζεται και το γράφημα που προέκυψε από το αρχείο με τις σχέσεις μεταξύ των ΑΣ. Κάθε κόκκινη ακμή σημαίνει σχέση ομότιμος-σε-ομότιμο, κάθε μπλε σημαίνει σχέση πάροχος-σε-πελάτη και κάθε μαύρη σημαίνει σχέση πελάτης-σε-πάροχο.



Σχήμα 4.8: Σχέσεις ΑΣ

5 Διαδικασία Κατασκευής Εργαλείου Εξομοίωσης Κατανεμημένων Επιθέσεων Άρνησης Παροχής Υπηρεσίας

Στο κεφάλαιο αυτό θα εξηγηθεί αναλυτικά ολόκληρη η διαδικασία της κατασκευής του εργαλείου εξομοίωσης κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας. Για την καλύτερη κατανόηση της διαδικασίας αυτής θα παρουσιαστεί αρχικά ο σκοπός του εργαλείου και θα δοθεί μια βασική εικόνα για τις λειτουργίες που εκτελεί. Με τον τρόπο αυτό ο αναγνώστης θα αποκτήσει μια καλύτερη αντίληψη για τον τελικό στόχο και έτσι θα είναι ικανός να καταλάβει τη λογική με την οποία γίνονται οι συγκεκριμένες εισηγήσεις, κατά την κατασκευή του εργαλείου.

5.1 Βασικά Χαρακτηριστικά Εργαλείου

Ο βασικός σκοπός του εργαλείου είναι η εξομοίωση κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας μεγάλων γράφων όπως αυτοί που παρουσιάστηκαν στο $4^{ο}$ κεφάλαιο αλλά και ακόμα μεγαλύτερων. Προφανέστατα, όταν ζητείται να γίνει μια εξομοίωση μεγάλου γράφου, είτε για την εξομοίωση κάποιας επίθεσης είτε και για οποιονδήποτε άλλο λόγο, οι υπολογιστικοί πόροι που χρειάζονται είναι πάρα πολλοί για να υπάρχουν σε ένα μοναδικό μηχάνημα.

Το εργαλείο αυτό λοιπόν λύνει το πρόβλημα διαμοιράζοντας τον μεγάλο αυτό γράφο σε όσα εικονικά μηχανήματα επιθυμεί και έχει πρόσβαση ο χρήστης. Έχει γίνει μία προσπάθεια επίσης να πραγματοποιούνται όλα όσο το δυνατόν πιο αυτοματοποιημένα. Με τον τρόπο αυτό είναι εξαιρετικά εύκολο να χτίσει-διαμοιράσει την τοπολογία που επιθυμεί καθώς για παράδειγμα τα αρχεία-scripts που χρειάζεται κάθε φορά να υπάρχουν στα διάφορα εικονικά μηχανήματα, ώστε

όλα να χτίζουν την ίδια τοπολογία με τον ίδιο τρόπο και να διαμοιράζονται σε αυτά κάθε φορά από το κύριο πρόγραμμα (manager).

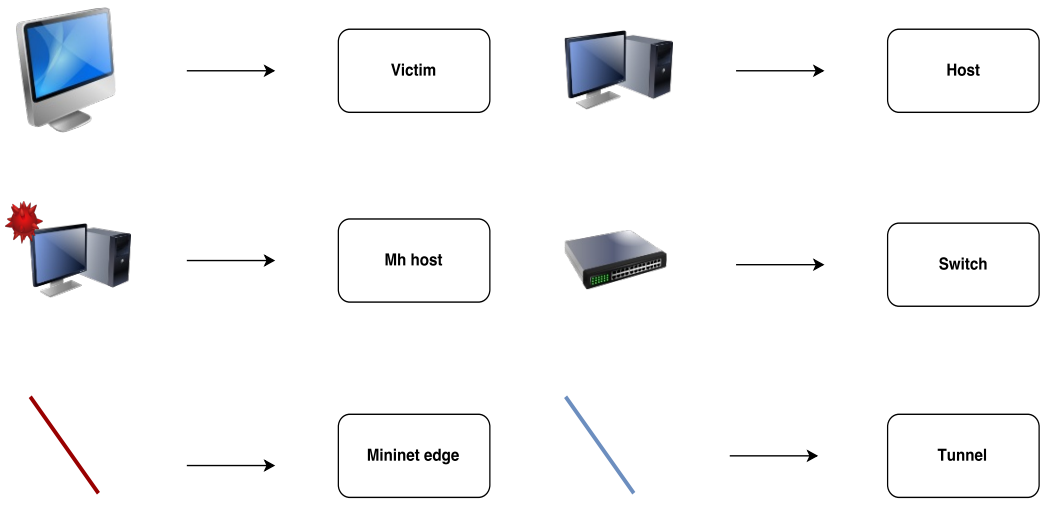
5.1.1 Λειτουργίες

Πρακτικά, οι λειτουργίες του εργαλείου είναι:

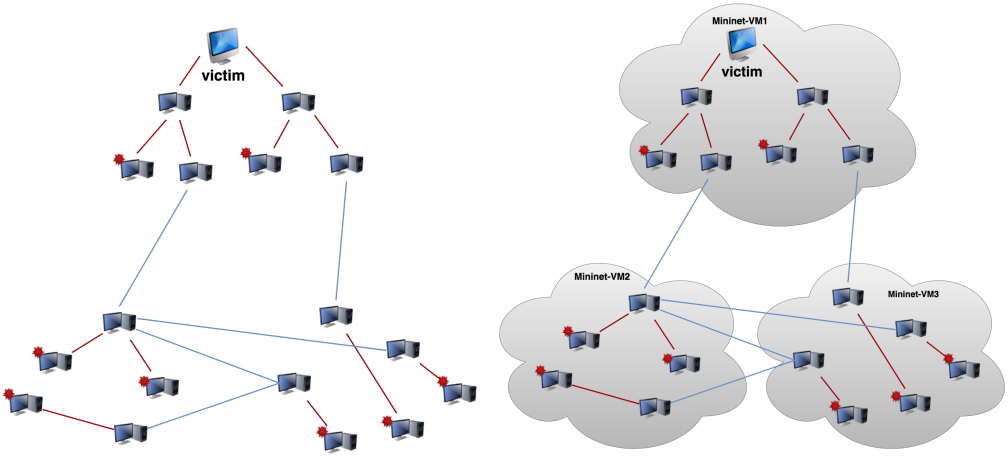
- Χωρισμός συνολικού γράφου της επίθεσης σε υπογράφους
- Απεικόνιση διαμοιρασμένου γράφου
- Αποστολή υπογράφων και αρχείων στα εικονικά μηχανήματα
- Χτίσιμο κάθε υπογράφου στο αντ/χο μηχάνημα
- Χτίσιμο ακμών μεταξύ διαφορετικών εικονικών μηχανημάτων μέσω tunnelling
- Εκτέλεση εντολών επίθεσης και συλλογής δεδομένων

Παρακάτω παρουσιάζεται γραφικά η γενική ιδέα της εξομοίωσης. Το κάθε σύννεφο αντιστοιχεί σε ένα εικονικό μηχάνημα (virtual machine – VM).

Το τι αντιπροσωπεύει το κάθε σχήμα στις επόμενες εικόνες εμφανίζεται εδώ:

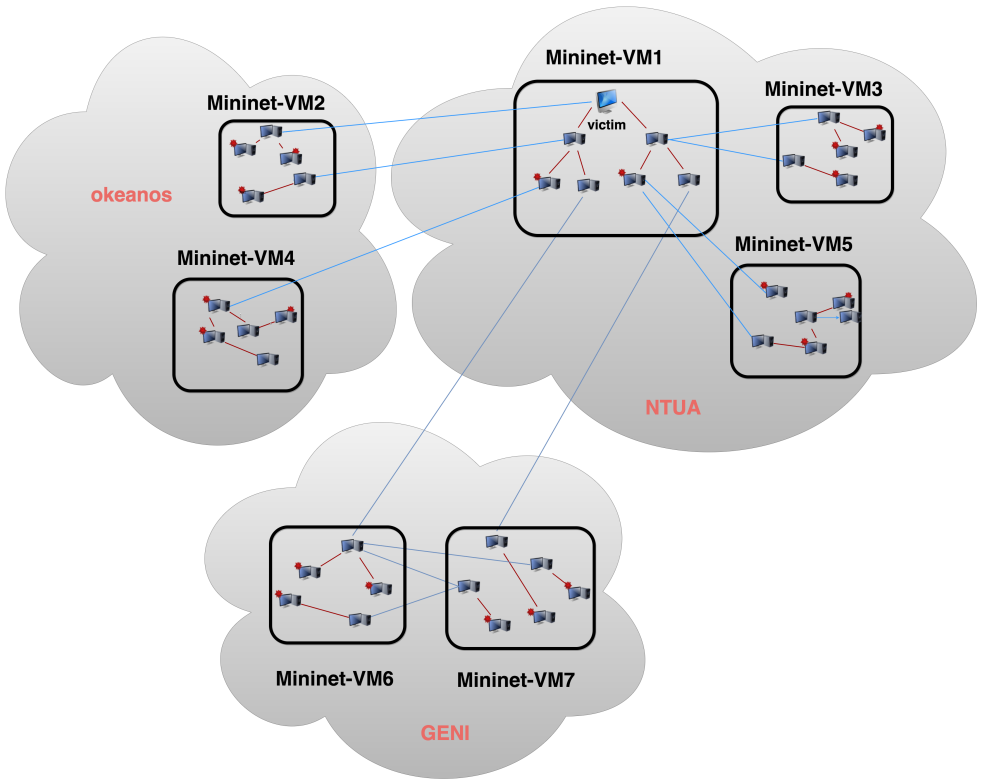


Σχήμα 5.1: Σημασία Συμβόλων



Σχήμα 5.2: Γενική ιδέα – Χωρισμός γράφου

Για λόγους που θα εξηγήσουμε σε επόμενη ενότητα τους ενδιάμεσους hosts τους υλοποιούμε με switches. Ένα πιο «αληθινό» σχήμα του τι ακριβώς υλοποιεί το εργαλείο αυτό φαίνεται από το παρακάτω σχήμα:



Σχήμα 5.3: Συνολική εικόνα

Από τα παραπάνω σχήματα φαίνεται αρκετά ξεκάθαρα το αποτέλεσμα της λειτουργίας του εργαλείου που κατασκευάστηκε. Ξεκινώντας από έναν γράφο με κόμβους (ΑΣ) και ακμές (σχέσεις μεταξύ των ΑΣ) γίνεται ο διαμοιρασμός, χωρίζονται δηλαδή κομμάτια του υπογράφου σε συννεφάκια, το ΑΣ-θύμα όπως και κάθε κακόβουλο-ΑΣ αντικαθίσταται από έναν host ενώ κάθε ενδιάμεσο ΑΣ από ένα switch. Εξαίρεση, η οποία δεν φαίνεται στα σχήματα αυτά, αποτελούν τα κακόβουλα-ΑΣ που είναι ταυτόχρονα και ενδιάμεσα-ΑΣ τα οποία υλοποιούνται και αυτά με host (θα εξηγηθεί αναλυτικότερα παρακάτω η λογική υλοποίησης). Στην συνέχεια, χτίζεται σε κάθε εικονικό μηχάνημα το αντίστοιχο ‘’συννεφάκι’’ μέσω mininet και τέλος χτίζονται με tunnelling οι ακμές που βρίσκονται μεταξύ κόμβων οι οποίοι ανήκουν σε διαφορετικό εικονικό μηχάνημα.

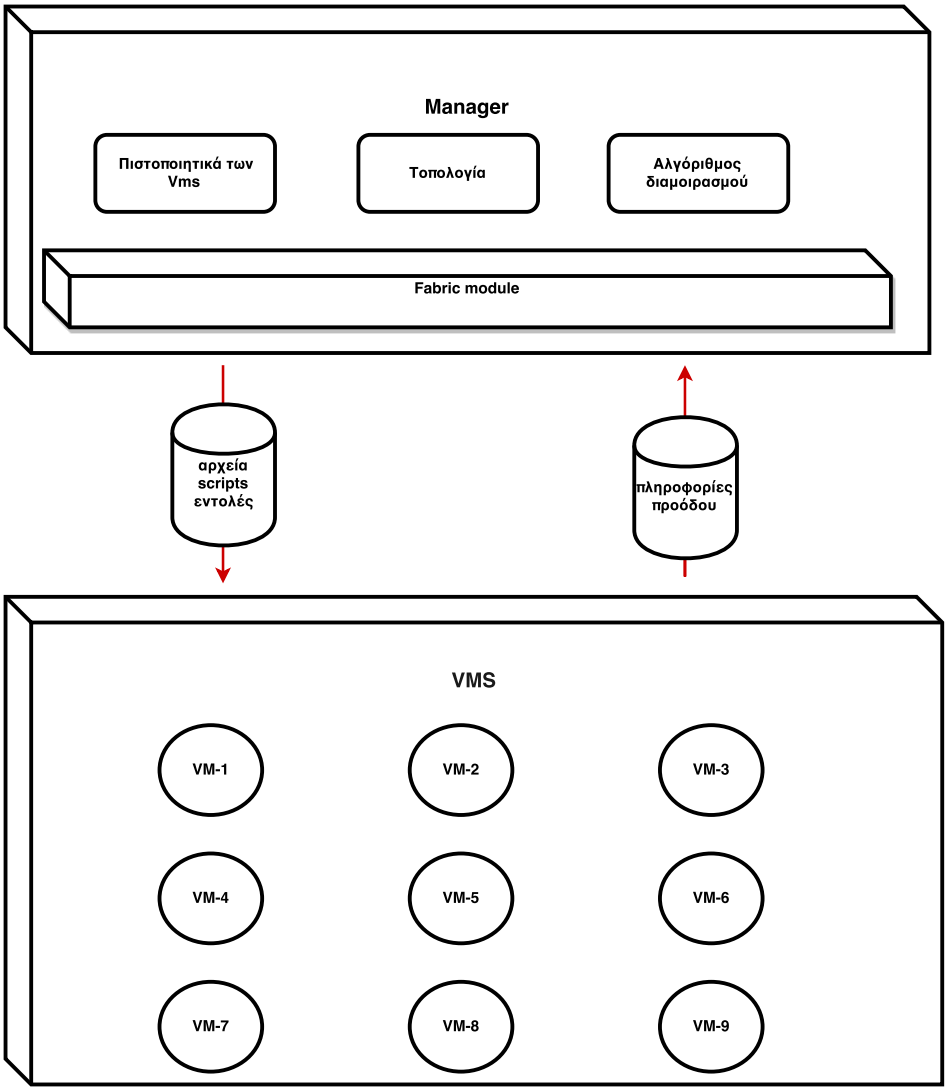
5.1.2 Αρχιτεκτονική

Το εργαλείο εξομοίωσης έχει ως βάση τον Manager. Ο Manager έχει πρόσβαση στα πιστοποιητικά (credentials) που είναι απαραίτητα για την σύνδεσή του με τα εικονικά μηχανήματα, στο δέντρο επίθεσης (σε μορφή λίστας μονοπατιών) το οποίο ο εκάστοτε χρήστης επιθυμεί να εξομοιώσει και στον αλγόριθμο διαμοιρασμού του δέντρου επίθεσης.

Έχει επίσης την δυνατότητα να μεταφέρει αρχεία και scripts καθώς και να εκτελεί προκαθορισμένες εντολές στα εικονικά μηχανήματα σειριακά ή και παράλληλα. Η πορεία της διαδικασίας εμφανίζεται στον Manager σε πραγματικό χρόνο .

Σχηματικά η αρχιτεκτονική του εργαλείου παρουσιάζεται στην επόμενη εικόνα. Ο Manager είναι εκείνος ο οποίος πυροδοτεί την όλη διαδικασία. Σημαντικότατο ρόλο στην διαδικασία αυτή έχει η βιβλιοθήκη της Python Fabric [34]. Αυτή, προσφέρει μια συλλογή λειτουργιών μέσω της χρήσης του πρωτοκόλλου SSH για την εκτέλεση local ή και remote εντολών φλοιού (shell commands) αλλά και για την μεταφορά αρχείων.

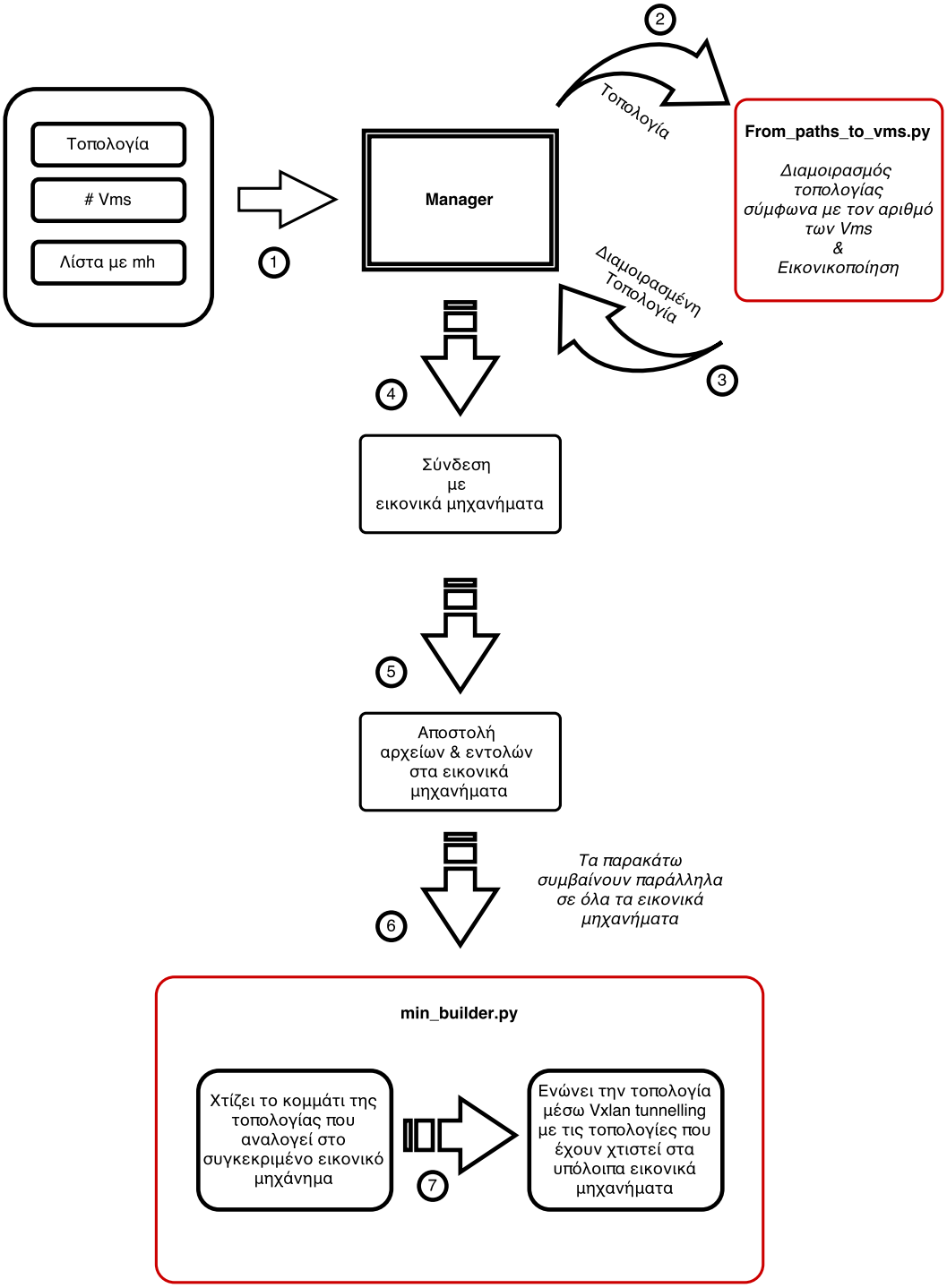
Έτσι, με την χρήση της βιβλιοθήκης αυτής γίνεται η σύνδεση του manager με τα εικονικά μηχανήματα όπως και η αποστολή των απαραίτητων αρχείων, εντολών και scripts που θα περιγραφούν παρακάτω στην ενότητα *5.3*. Τέλος, τα εικονικά μηχανήματα «τρέχουν» όσα έχουν ζητηθεί από τον Manager, και τον ενημερώνουν σε πραγματικό χρόνο με τα αποτελέσματα της εκτέλεσής τους.



Σχήμα 5.4: Βασική αρχιτεκτονική εργαλείου

5.1.3 Διάγραμμα Ροής Εξομοίωσης

Παρακάτω παρουσιάζονται χρονικά τα βασικά γεγονότα που λαμβάνουν μέρος για την επίτευξη της εξομοίωσης. Στην συνέχεια ακολουθεί μια σύντομη περιγραφή των βημάτων.



Σχήμα 5.5: Διάγραμμα ροής εξομοίωσης

(1) Δίνονται ως ορίσματα στον Manager η τοπολογία των ΑΣ, ο αριθμός των VMs στον οποίο θα μοιραστεί η τοπολογία (στον κώδικα του Manager υπάρχουν τα credentials τους) και η λίστα με τα κακόβουλα-ΑΣ.

(2) Η τοπολογία και το πλήθος των VMs δίνεται ως όρισμα στο script *from_paths_to_vms.py* (ενότητα *4.5.1)* το οποίο διαμοιράζει και εικονικοποιεί τον διαμοιρασμένο γράφο.

(3) Το *from_paths_to_vms.py* αποθηκεύει μία εικόνα του διαμοιρασμένου γράφου και επιστρέφει ένα αρχείο που περιέχει τον γράφο αλλά και την πληροφορία του πώς αυτός διαμοιράστηκε.

(4) Γίνεται η σύνδεση με τα εικονικά μηχανήματα μέσω της βιβλιοθήκης fabric.

(5) Το αρχείο που επιστράφηκε από το βήμα (3) μαζί με το script που είναι υπεύθυνο για το χτίσιμο του κάθε υπογράφου στο κάθε VM και μαζί με την λίστα με τα κακόβουλα-ΑΣ στέλνονται στα VM. Στέλνονται επίσης όσες εντολές είναι επιθυμητό να εκτελεστούν σε αυτά όπως το «τρέξιμο» του script που χτίζει τον υπογράφο και η εκκίνηση της επίθεσης από το κάθε κακόβουλο-ΑΣ.

(6) Από το βήμα αυτό και έπειτα φαίνεται το τι συμβαίνει πλέον μέσα στο κάθε VM. Μέσω του script *min_builder.py* [35], το οποίο θα αναλυθεί περεταίρω σε επόμενη ενότητα, χτίζεται στο κάθε VM το κομμάτι του γράφου που του αναλογεί.

(7) Δημιουργούνται και οι τελικές ακμές μέσω tunnelling μεταξύ των ΑΣ διαφορετικών VM οπότε και σχηματίζεται ολόκληρη η τοπολογία.

5.2 Διαδικασία Εκτέλεσης Κώδικα Εργαλείου

Στην ενότητα αυτή θα παρουσιαστεί αναλυτικά πως χρησιμοποιείται το εργαλείο αυτό καθώς και τι ακριβώς κάνει βήμα-βήμα. Μέχρι το τέλος της ενότητας ο αναγνώστης θα έχει καταλάβει την λογική του και θα είναι ικανός να το χρησιμοποιήσει.

Η όλη διαδικασία πυροδοτείται με την παρακάτω εντολή:

$ python fabfile.py topology.txt #number_of_VMs MH.txt

Το script *fabfile.py* [36] αντιστοιχεί πρακτικά στον Manager που έχει αναφερθεί παραπάνω και αναλύεται στην συνέχεια.

5.2.1 Manager - Fabfile.py

Το script αυτό περιέχει τις διευθύνσεις των VMs καθώς και τους κωδικούς πρόσβασής τους, ώστε να είναι δυνατόν να πραγματοποιηθεί η σύνδεση με αυτά. Επίσης περιέχει έναν πίνακα με τις διευθύνσεις IP αυτών. Αρχικά μέσα στο script αυτό εκτελείται η εντολή φλοιού:

$ python from_paths_to_vms.py topology.txt #number_of_VMs MH.txt

της οποίας η λειτουργία έχει αναλυθεί στην ενότητα *4.5.1*. Μετά το πέρας της, όπως έχει προαναφερθεί σε διάφορα σημεία, έχει αποθηκευτεί μια εικόνα της διαμοιρασμένης τοπολογίας που θα χτιστεί και έχει καταγραφεί σε ένα αρχείο η τοπολογία με ενσωματωμένη την πληροφορία του πώς έχει γίνει ο διαμοιρασμός αυτός (έχει επίσης αναλυθεί στην ενότητα *4.5.1*).

Στην συνέχεια το αρχείο ***new_edges.txt*** *(4.5.1)* με την διαμοιρασμένη τοπολογία μετατρέπεται στο αρχείο ***edges_mininet.txt*** με τον εξής τρόπο: γίνεται αντικατάσταση των ομάδων (0,1,2..) που περιέχονται μέσα στις τούπλες (tuples) του πρώτου αρχείου και δείχνουν σε ποια ομάδα ανήκει το κάθε ΑΣ σύμφωνα με τον διαμορασμό που έγινε, με τις διευθύνσεις IP των VMs. Έτσι για παράδειγμα την ομάδα 0 αναλαμβάνει να την χτίσει το VM με διεύθυνση IP XXX.XXX.XXX.XXX και αντιστοίχως για κάθε άλλη ομάδα και διεύθυνση IP. Έτσι όταν το *edges_mininet.txt* αποσταλλεί στο κάθε VM, αναλόγως με την διεύθυνση IP του το VM θα γνωρίζει ποιο μέρος της τοπολογίας αντιστοιχεί σε αυτό, έτσι ώστε να το χτίσει.

Έχοντας πλέον το ***edges_mininet.txt*** έχει έρθει η στιγμή να γίνει η σύνδεση. Καλείται λοιπόν η συνάρτηση *connection()* στην οποία γίνεται **παράλληλα** η σύνδεση με κάθε VM και πραγματοποιούνται επίσης **παράλληλα** σε κάθε ένα από αυτά οι εξής εντολές:

- ***$ sudo mn –c*** :καθαρισμός οποιουδήποτε υπολείμματος από προηγούμενη τοπολογία
- ***$ mkdir –p testing****: Δημιουργία φακέλου για καλύτερη οργάνωση των όσων θα εκτελεστούν*
- ***$*** **put('edges_mininet.txt','~/testing/')**: Αποστολή αρχείου *edges_mininet.txt* μέσα στον φάκελο ~/*testing/*
- ***$*** **put('min_builder.py','~/testing/'))**: Αποστολή script *min_builder.py* μέσα στον φάκελο ~/*testing/*
- ***$*** **put('allMh.txt','~/testing/'))**: Αποστολή αρχείου *MH.txt* μέσα στον φάκελο ~/*testing/*
- ***$*** **run('cd testing && sudo python min_builder.py %s edges_mininet.txt MH.txt' %env.host , pty=False)**: Εκτέλεση του script *min_builder.py* μέσα στον φάκελο ~/*testing/*

Μετά από αυτά εμφανίζεται στον Manager η πρόοδος των εντολών αυτών. Παρακάτω θα αναλυθεί το script *min_builder.py.*

5.2.2 min_builder.py

Εδώ θα αναλυθεί η ιδέα πίσω από το συγκεκριμένο script και ο τρόπος λειτουργίας του. Αυτό που πρακτικά πραγματοποιεί είναι, δεδομένων των όσων έχουμε περιγράψει παραπάνω, να κατασκευάζει το μέρος της τοπολογίας που ζητείται κάθε φορά. Ακολουθούν διάφορα τεχνικά προβλήματα που παρουσιάστηκαν καθώς και οι τρόποι με τους οποίους αυτά ξεπεράστηκαν.

Το πρώτο θέμα είναι πως θα υλοποιηθεί η τοπολογία. Δηλαδή τι θα αναπαραστήσουμε με mininet hosts και τι με switches. Το σκεπτικό στο σημείο αυτό ήταν το εξής: Κάθε κακόβουλο-ΑΣ στέλνει κίνηση επομένως αναγκαστικά πρέπει να υλοποιηθεί ως mininet host. Τα υπόλοιπα όμως ΑΣ που απλώς μεταφέρουν την κίνηση πρακτικά λειτουργούν ως switches που προωθούν την κίνηση από το ένα ΑΣ στο άλλο. Φυσικά, κάποια κακόβουλα-ΑΣ χρησιμοποιούνται πέρα από το να στέλνουν κίνηση στο θύμα και στο να μεταφέρουν κίνηση για άλλα ΑΣ (ενδιάμεσα & κακόβουλα ΑΣ). Επιπρόσθετα, ως γνωστόν οι mininet hosts δεν μπορούν να μεταφέρουν κίνηση για άλλους mininet hosts. Επομένως έχουμε φύλλα-κακόβουλα-ΑΣ (φύλλα του δέντρου της επίθεσης) τα οποία δεν μεταφέρουν κίνηση για κανένα άλλο ΑΣ, ενδιάμεσα-κακόβουλα-ΑΣ τα οποία και στέλνουν κίνηση και μεταφέρουν και ενδιάμεσα-ΑΣ τα οποία μόνο μεταφέρουν κίνηση. Για όλους αυτούς τους λόγους αποφασίστηκαν τα εξής:

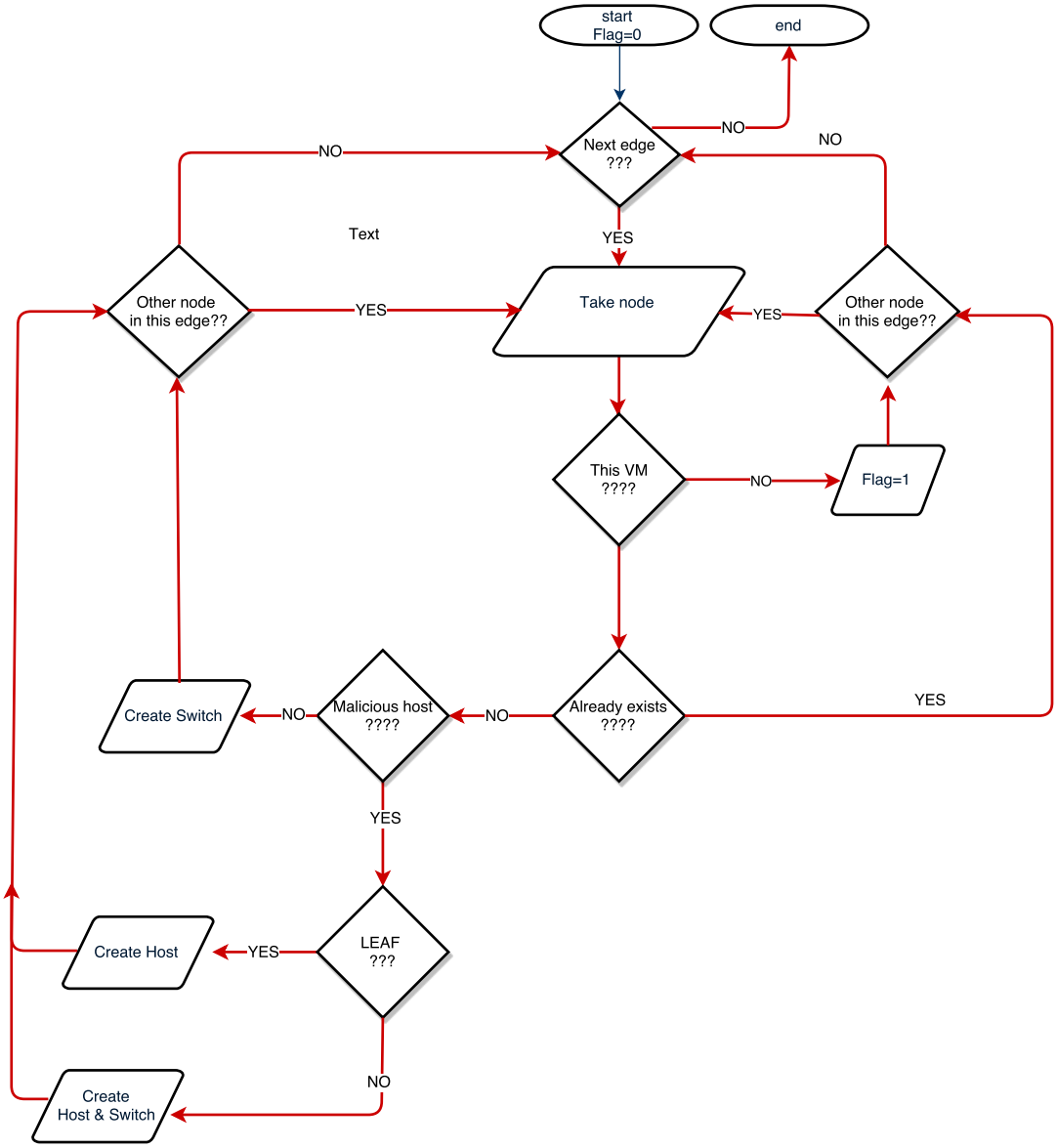
- Φύλλο-κακόβουλο-ΑΣ ➔ mininet **host**
- Ενδιάμεσο-κακόβουλο-ΑΣ ➔ mininet **host** + mininet **switch**
- Ενδιάμεσο-ΑΣ ➔ mininet **switch**

Έτσι, το script ξεκινάει υπολογίζοντας τον βαθμό του κάθε κόμβου της τοπολογίας ώστε να γνωρίζουμε στην συνέχεια ποια κακόβουλα-ΑΣ είναι φύλλα και ποια είναι ενδιάμεσα. Αφού γίνει αυτό κατασκευάζεται μια λίστα που περιέχει τις ακμές όπως υπάρχουν στο *edges_mininet.txt* και στην συνέχεια καλείται η συνάρτηση που θα χτίσει την τοπολογία.

Για να γίνει το χτίσιμο της τοπολογίας, πρέπει να είναι γνωστές κάποιες βασικές πληροφορίες για τους κόμβους που μελετώνται:

- Αν πρέπει να χτιστεί από το VM το οποίο επεξεργάζεται τον κόμβο
- Αν έχει ήδη χτιστεί
- Αν είναι είναι Φύλλο-κακόβουλο-ΑΣ, Ενδιάμεσο-κακόβουλο-ΑΣ ή Ενδιάμεσο ΑΣ

Παρακάτω παρουσιάζεται ένα διάγραμμα ροής που περιγράφει την διαδικασία:



Σχήμα 5.6: Διάγραμμα ροής min_builder.py

Για απλούστευση στο παραπάνω σχήμα έχει παραληφθεί το εξής: Η μεταβλητή flag εκφράζει το αν κάποιο από τους 2 ΑΣ-κόμβους που σχηματίζουν την εκάστοτε ακμή βρίσκεται σε διαφορετικό VM. Έτσι όταν:

-flag=0 ➔ ίδιο VM

-flag=1➔ ο 1^{ος} ΑΣ-κόμβος είναι σε άλλο VM

-flag=2➔ ο 2ος ΑΣ-κόμβος είναι σε άλλο VM

Αποκλείεται και τα 2 ΑΣ να είναι σε άλλο VM από αυτό που επεξεργάζεται την ακμή γιατί τότε το VM αυτό δεν θα ασχολιόταν με την ακμή αυτή καθόλου.

Στο σχήμα φαίνεται μόνο η αλλαγή της μεταβλητής flag από 0 ➔ 1 για να μην προκύψει περισσότερο πολύπλοκο. Η μόνη διαφορά στην πραγματικότητα είναι ότι ενώ όταν μελετάται ο 1ος ΑΣ-κόμβος της ακμής γίνεται ό,τι φαίνεται από το σχήμα, όταν μελετάται ο 2ος ΑΣ-κόμβος στο σημείο που έχουμε flag=1 θα είχαμε flag=2.

Το τι τιμή έχει η μεταβλητή flag χρησιμοποιείται ώστε να αποφασιστεί τι τύπου ακμή πρέπει να κατασκευαστεί για να ενώσει τους 2 κόμβους. Στην περίπτωση που η τιμή της μεταβλητής αυτής είναι 0, θα ενωθούν με ακμή μέσω του mininet (βρίσκονται και η 2 κόμβοι στο ίδιο VM), ενώ στην περίπτωση που η τιμή της είναι 1 ή 2 θα χρειαστεί να κατασκευαστεί ένα tunnel μεταξύ των 2 VM καθώς οι 2 κόμβοι έχουν κατασκευαστεί σε διαφορετικό VM.

Κάτι ακόμα που δεν φαίνεται από το σχήμα είναι η λογική με την οποία αποδόθηκαν οι διευθύνσεις IP στους διάφορους hosts. Έχει σημασία κάθε host να έχει διαφορετικη διεύθυνση IP.

Για τον σκοπό αυτό η απόδοση των διευθύνσεων IP έγινε με την παρακάτω λογική: Οι διευθύνσεις όλων των host είναι της μορφής 10.0.X1.X2. Το κάθε 'X' είναι ένας αριθμός που αποτελείται από 8 bits. Από τους αριθμούς των ΑΣ που έχουμε δεν υπάρχει κάποιος που να χρησιμοποιείται και να είναι μεγαλύτερος από 65.000. Αυτό σημαίνει πως με 16 bit μπορούμε να αναπαραστήσουμε τον αριθμό από κάθε ΑΣ. Αυτό που γίνεται λοιπόν είναι ότι το X1 προκύπτει με το λογικό AND μεταξύ του αριθμού του εκάστοτε ΑΣ με τον αριθμό 65280 (0xb1111111100000000) και ολίσθηση 8 θέσεις δεξιά ώστε να προκύψει ένας 8-bit αριθμός, ενώ το X2 προκύπτει με λογικό AND με το 255 (0xb11111111).

Πρακτικά δηλαδή σπάει ο αριθμός του ΑΣ σε δύο 8-bit αριθμούς και έτσι εξασφαλίζεται και ότι κάθε διαφορετικό ΑΣ αποκτά διαφορετική διεύθυνση IP και ότι κάθε φορά που θα εκτελεστεί το πρόγραμμα στο ίδιο ΑΣ θα αποδοθεί η ίδια διεύθυνση IP.

Έπειτα από τα παραπάνω έχει χτιστεί η τοπολογία στο mininet του κάθε VM και έχει κρατηθεί η πληροφορία του ποια switch πρέπει να χτίσουν tunnels μεταξύ τους ώστε να παρέχουν επικοινωνία στα VM στα οποία ανήκουν. Χρησιμοποιώντας την πληροφορία αυτή χτίζονται τα tunnels αυτά τα οποία είναι είτε τύπου VXLAN είτε τύπου GRE (η αλλαγή ανάμεσα στα 2 είναι εξαιρετικά εύκολη προγραμματιστικά). Η εντολή με την οποία χτίζεται ένα VXLAN tunnel είναι η παρακάτω:

switch.cmd('ovs-vsctl add-port %(switch)s vx%(number1)s -- set interface vx%(number1)s type=vxlan options:remote_ip=%(vm2)s options:key=%(number)s'% {"number":vDict[str(switch)][2][i],"number1":vDict[str(switch)][1][i], "vm2":vDict[str(switch)][0][i],"switch":s})

Από την εντολή αυτή αξίζει να αναφερθεί η επιλογή ***type*=*vxlan*** η οποία δηλώνει τον τύπο του tunnel. Αν ήταν επιθυμητό να χρησιμοποιηθεί gre tunnelling θα άλλαζε ως ***type=gre*.** Επίσης, ενδιαφέρον παρουσιάζει η επιλογή κλειδιού ***options:key=...*** η οποία βοηθάει στο να ξεχωρίζουν τα διάφορα tunnels του ίδιου VM μεταξύ τους.

Αφού τα παραπάνω εκτελεστούν από όλα τα VMs η τοπολογία έχει πλέον εξομοιωθεί. Είναι λοιπόν δυνατό να ξεκινήσει η επίθεση (ή και ο,τιδήποτε άλλο θα ήθελε κανείς να τρέξει φυσικά).

Για τον σκοπό αυτό, ελέγχεται κάθε host και

- Αν είναι το **ΑΣ-θύμα**: Δεν εκτελεί τίποτα ή εκτελεί εντολές monitoring ώστε να μετρήσει και να καταγράψει για παράδειγμα την κίνηση που δέχεται.

- Αν **δεν** είναι το **ΑΣ-θύμα**: Στέλνει κίνηση στο ΑΣ-θύμα.

Σύμφωνα με όσα έχουν αναφερθεί στο κεφάλαιο αυτό λοιπόν, μέσω του εργαλείου εξομοίωσης που περιγράφηκε γίνεται ο διαμοιρασμός της τοπολογίας, έπειτα η παράλληλη κατασκευή των επιμέρους τοπολογιών από το κάθε VM, η κατασκευή των tunnels μεταξύ των switches στα διαφορετικά VM και τέλος η πυροδότηση της επίθεσης.

6 Πειραματική Αξιολόγηση

Στο κεφάλαιο αυτό έγιναν διάφορες μετρήσεις-πειράματα. Οι μεταβλητές κάθε μέτρησης ήταν ο αριθμός των κακόβουλων-ΑΣ επομένως και το αντίστοιχο μέγεθος του κατανεμημένου γράφου επίθεσης αλλά και ο αριθμός των εικονικών μηχανημάτων στον οποίο ο εκάστοτε γράφος διαμοιραζόταν.

Οι μετρήσεις πραγματοποιήθηκαν με την βοήθεια του iperf το οποίο είναι ένα εργαλείο που χρησιμοποιείται για την εύρεση του μέγιστου bandwidth σε δίκτυα IP, αλλά και την εκτίμηση της ποιότητας τους. Το θύμα-ΑΣ εκτελεί το χρέος του iperf server ενώ τα κακόβουλα-ΑΣ στέλνουν κίνηση σε αυτό ως iperf clients.

Για τον κάθε γράφο που εμφανίζεται παρακάτω αναφέρονται τα εξής:

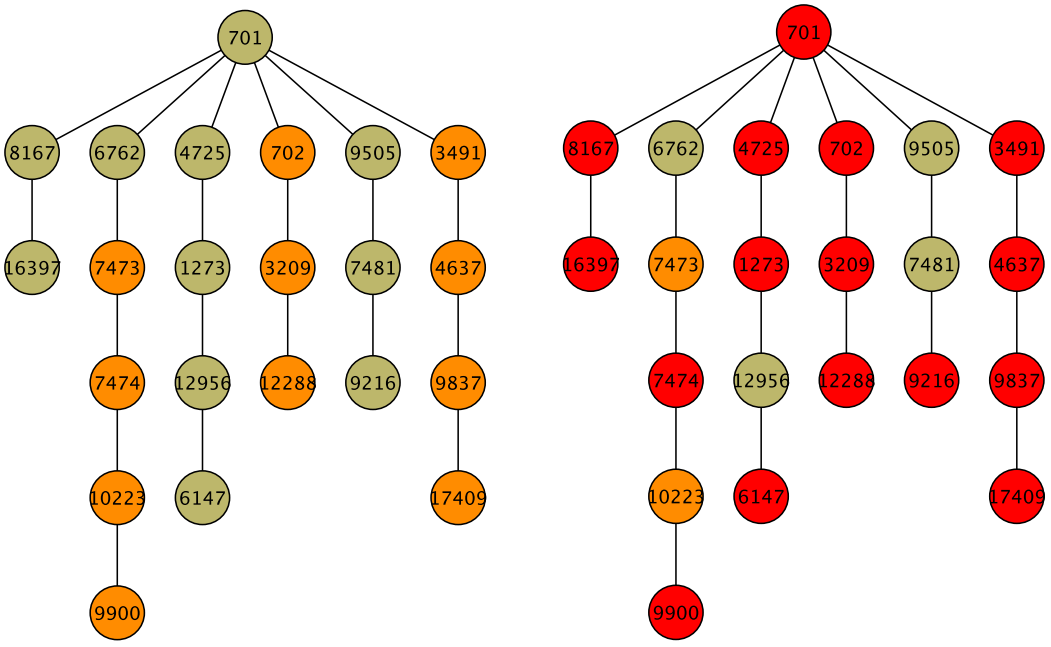
- Ο αριθμός των κόμβων από τους οποίους αποτελείται
- Πόσοι από αυτούς αντιπροσωπεύουν κακόβουλα-ΑΣ
- Σε πόσα εικονικά μηχανήματα διαμοιράστηκε

Για την κάθε μέτρηση παρουσιάζονται 2 σχήματα: στο $1^{ο}$ φαίνεται ο διαμοιρασμός του γράφου ενώ στο $2^{ο}$ φαίνονται ποιοι κόμβοι αφορούν κακόβουλα-ΑΣ (τα υπόλοιπα χρώμματα εκτός των κόκκινων δεν μας ενδιαφέρουν στα $2^{α}$ σχήματα). Αυτές οι απεικονίσεις δημιουργήθηκαν κατά την εκτέλεση του εργαλείου πριν αρχίσει το χτίσιμο του γράφου στα εικονικά μηχανήματα όπως έχει ήδη προαναφερθεί στο κεφάλαιο 5.

6.1 Σχήματα

Γράφος Α

- *22 Κόμβοι*
- *15 κακόβουλοι χρήστες*
- *2 εικονικά μηχανήματα*

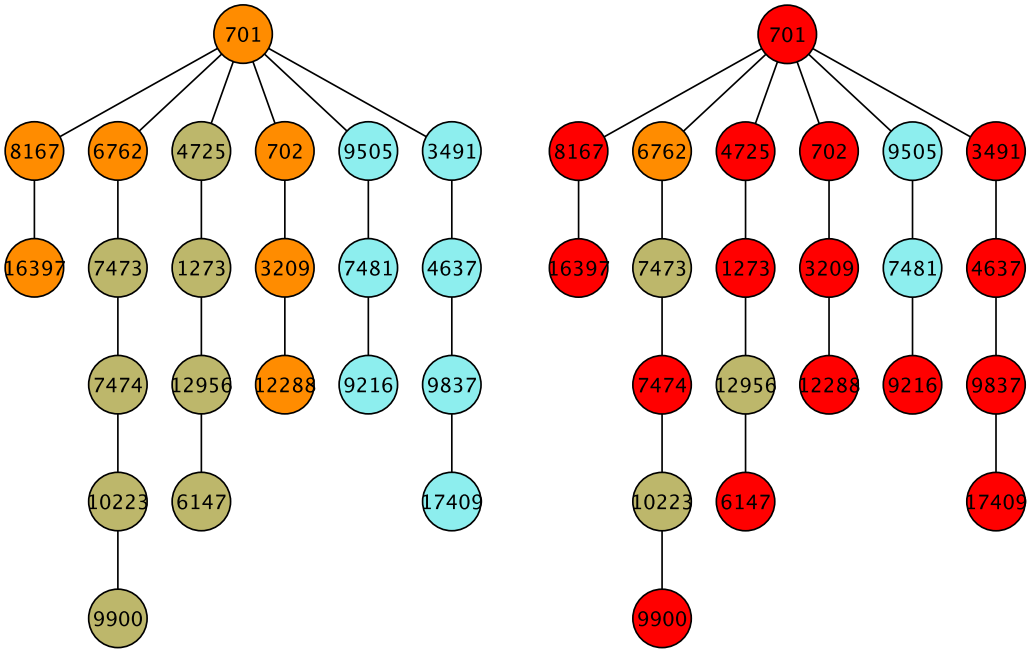


partition graph *malicious hosts graph*

Σχήμα 6.1: Γράφος Α – 22 κόμβοι| 15 κακόβουλοι | 2 VMs

Γράφος Β

- *22 Κόμβοι*
- *15 κακόβουλοι χρήστες*
- *3 εικονικά μηχανήματα*

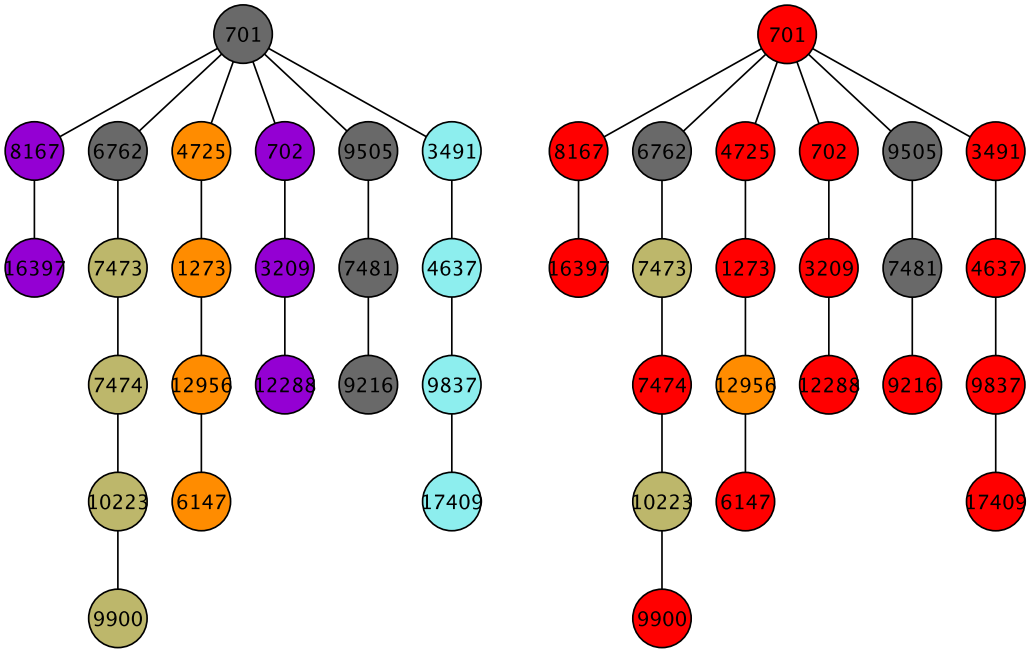


partition graph *malicious hosts graph*

Σχήμα 6.2: Γράφος Β – 22 κόμβοι| 15 κακόβουλοι | 3 VMs

Γράφος Γ

- *22 Κόμβοι*
- *15 κακόβουλοι χρήστες*
- *5 εικονικά μηχανήματα*

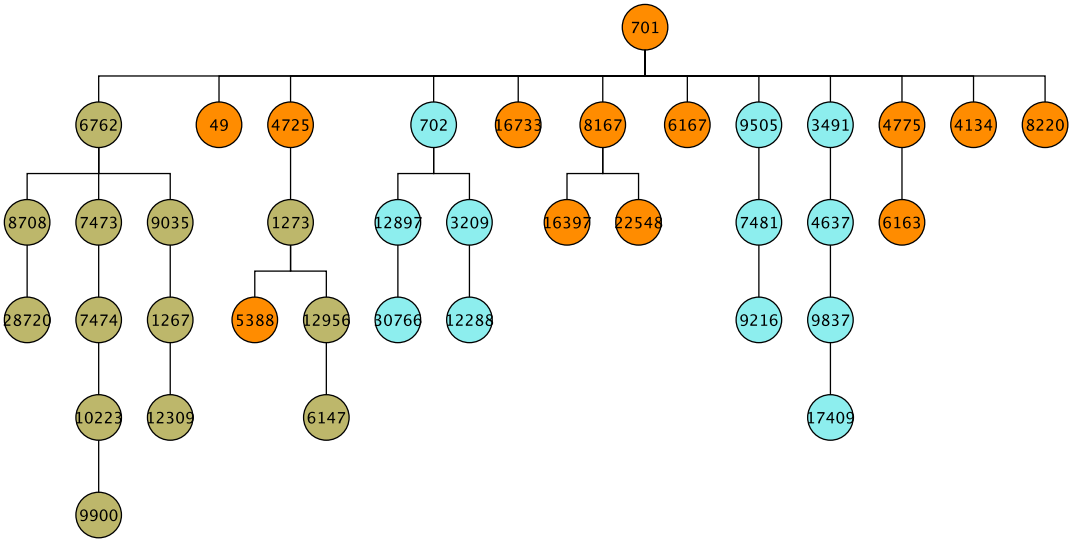


partition graph *malicious hosts graph*

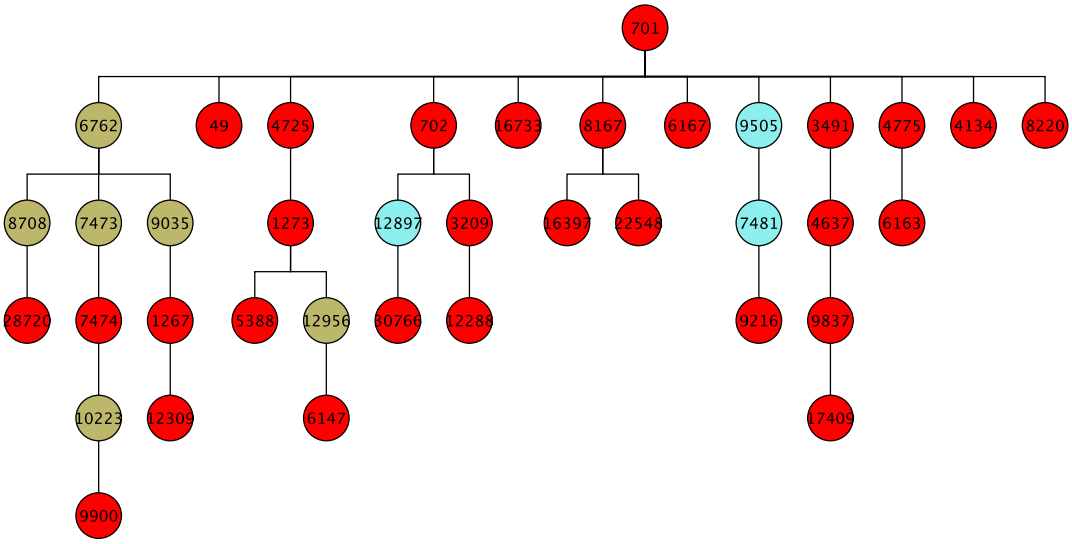
Σχήμα 6.3: Γράφος Γ – 22 κόμβοι| 15 κακόβουλοι | 5 VMs

Γράφος Δ

- *38 Κόμβοι*
- *28 κακόβουλοι χρήστες*
- *3 εικονικά μηχανήματα*



partition graph

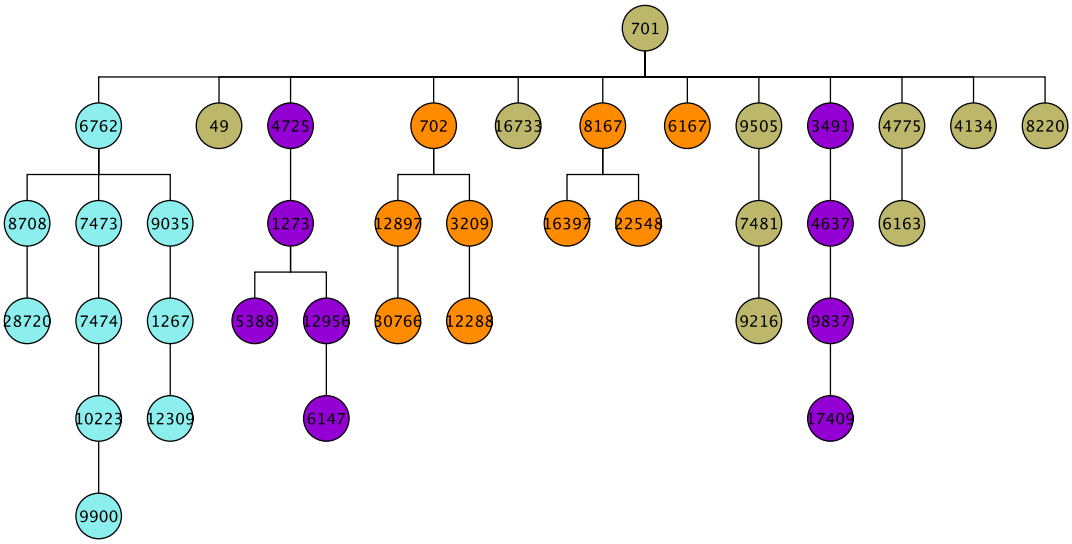


malicious hosts graph

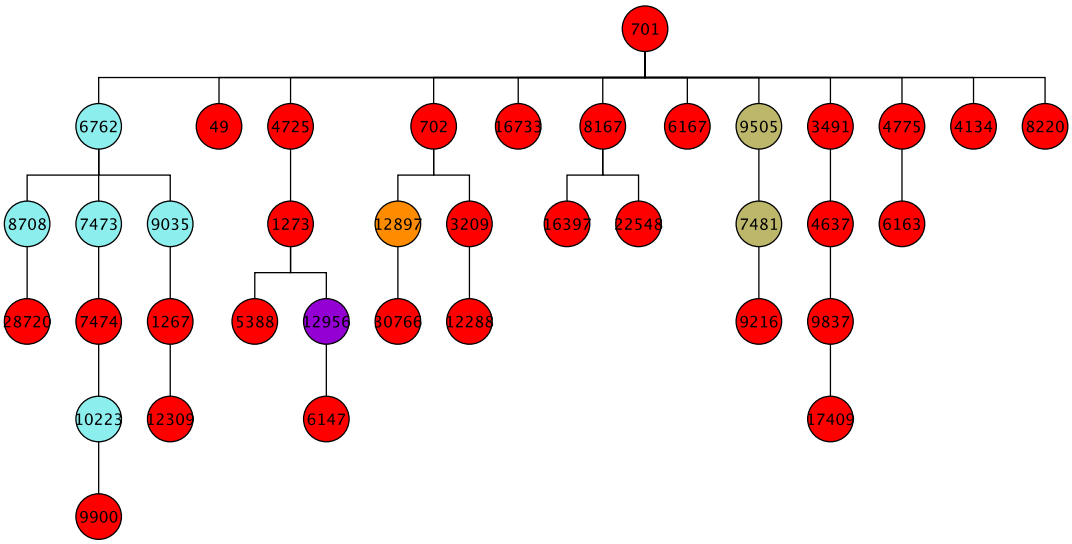
Σχήμα 6.4: Γράφος Δ – 38 κόμβοι| 28 κακόβουλοι | 3 VMs

Γράφος Ε

- *38 Κόμβοι*
- *28 κακόβουλοι χρήστες*
- *4 εικονικά μηχανήματα*



partition graph

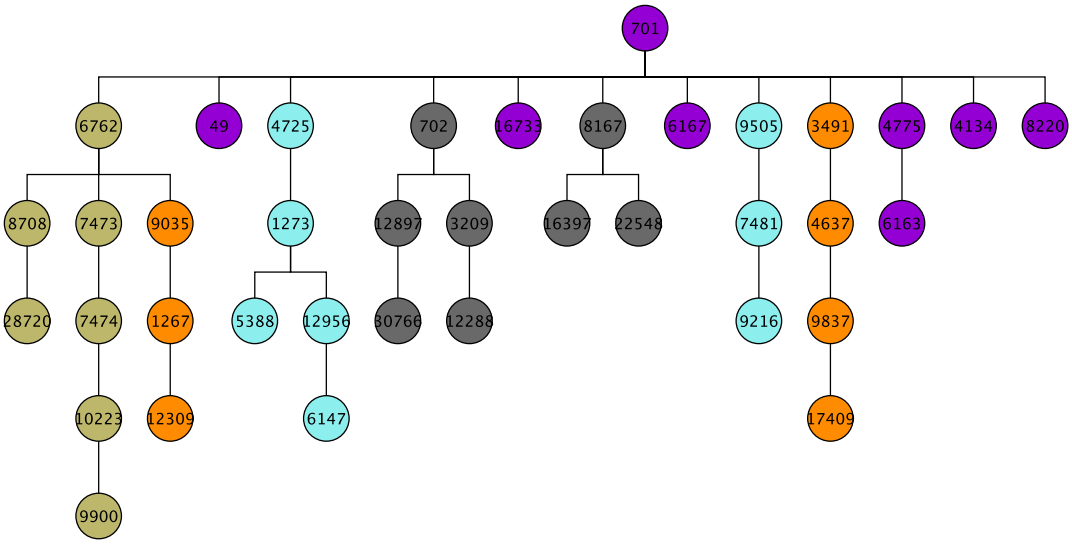


malicious hosts graph

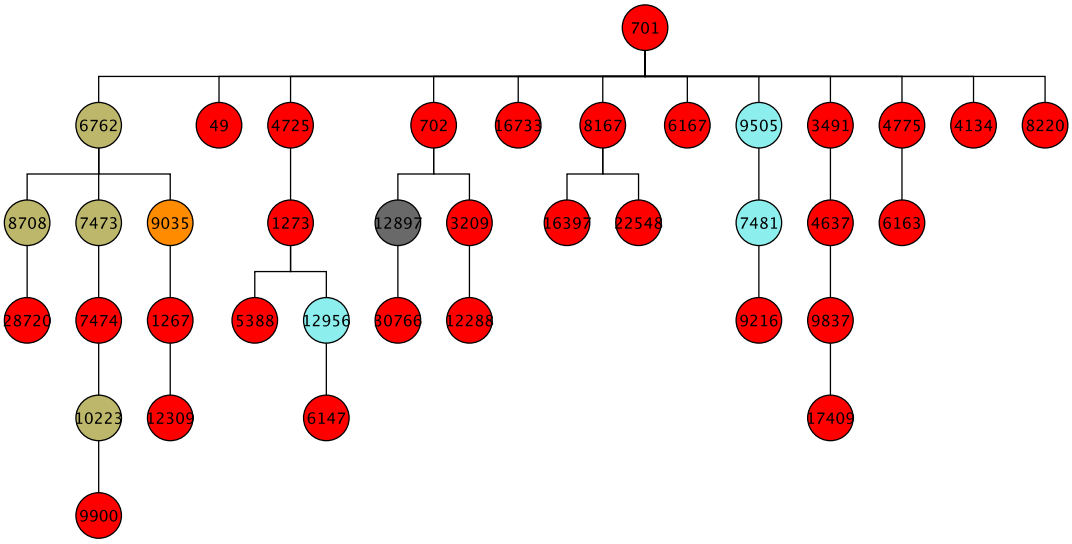
Σχήμα 6.5: Γράφος Ε – 38 κόμβοι| 28 κακόβουλοι | 4 VMs

Γράφος Z

- *38 Κόμβοι*
- *28 κακόβουλοι χρήστες*
- *5 εικονικά μηχανήματα*



partition graph

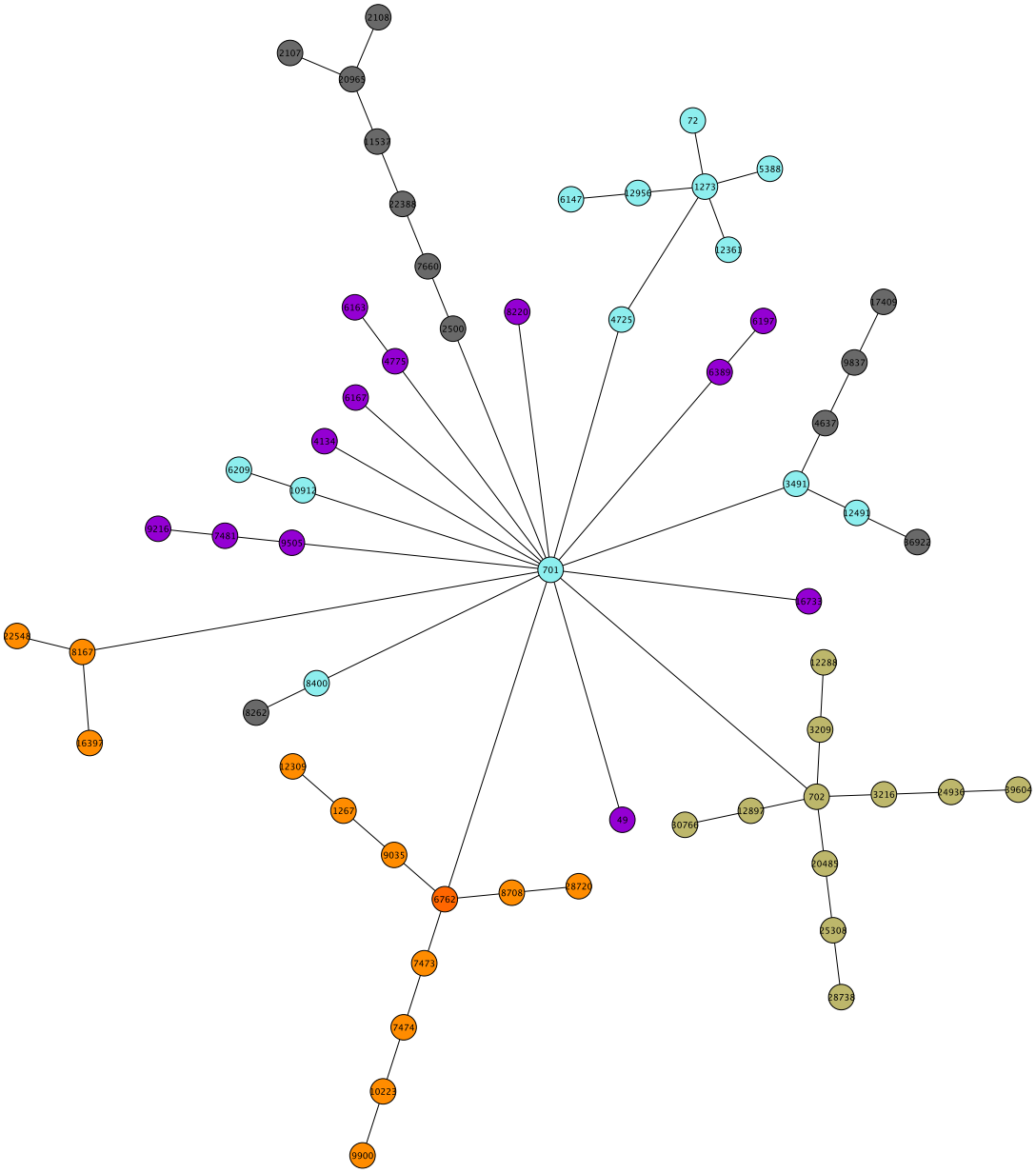


malicious hosts graph

Σχήμα 6.5: Γράφος Z – 38 κόμβοι| 28 κακόβουλοι | 5 VMs

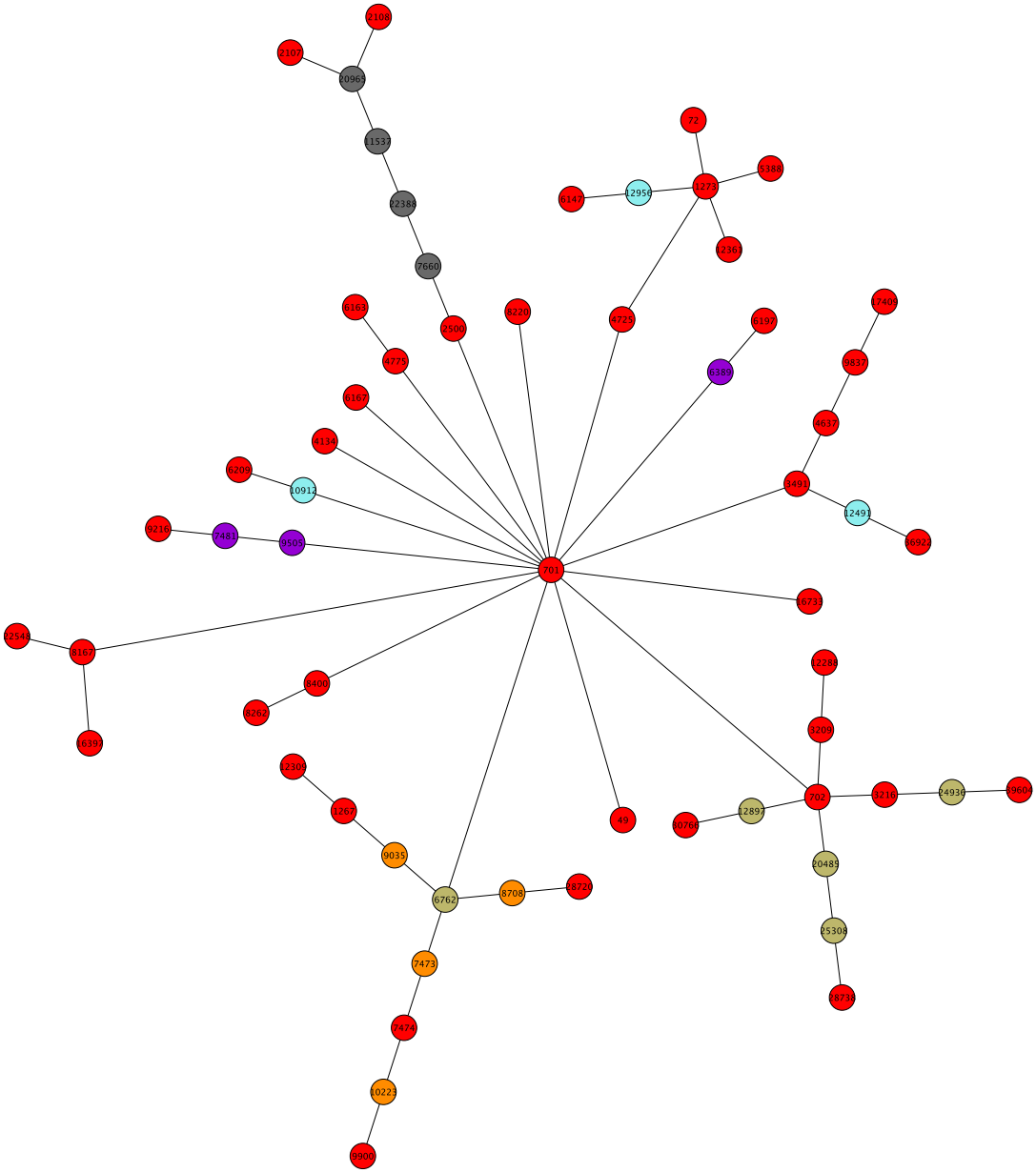
Γράφος Η

- *61 Κόμβοι*
- *40 κακόβουλοι χρήστες*
- *5 εικονικά μηχανήματα*



partition graph

Σχήμα 6.6.1: Γράφος Η – 38 κόμβοι| 28 κακόβουλοι | 3 VMs

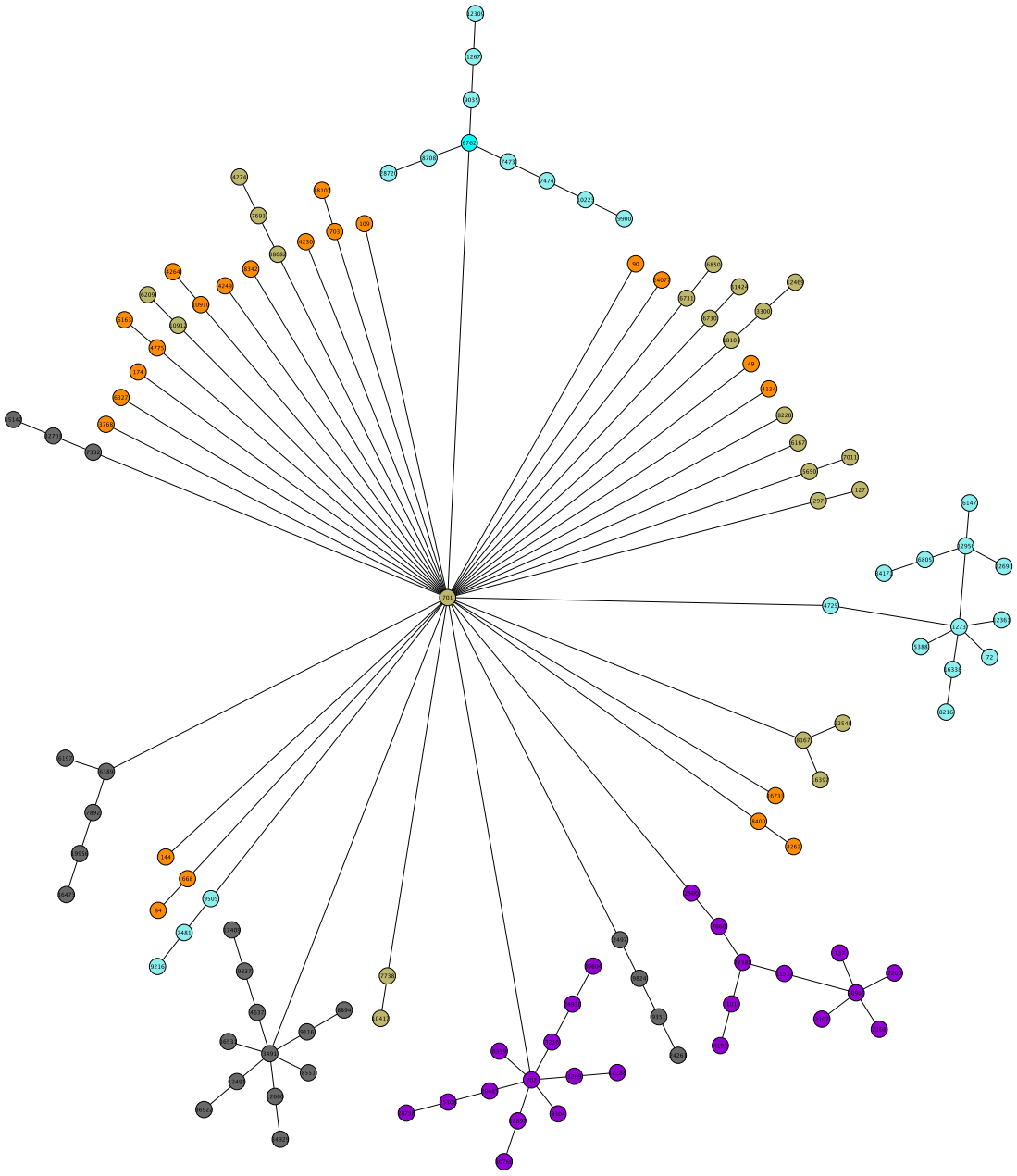


malicious hosts graph

Σχήμα 6.6.2: Γράφος Η – 38 κόμβοι| 28 κακόβουλοι | 3 VMs

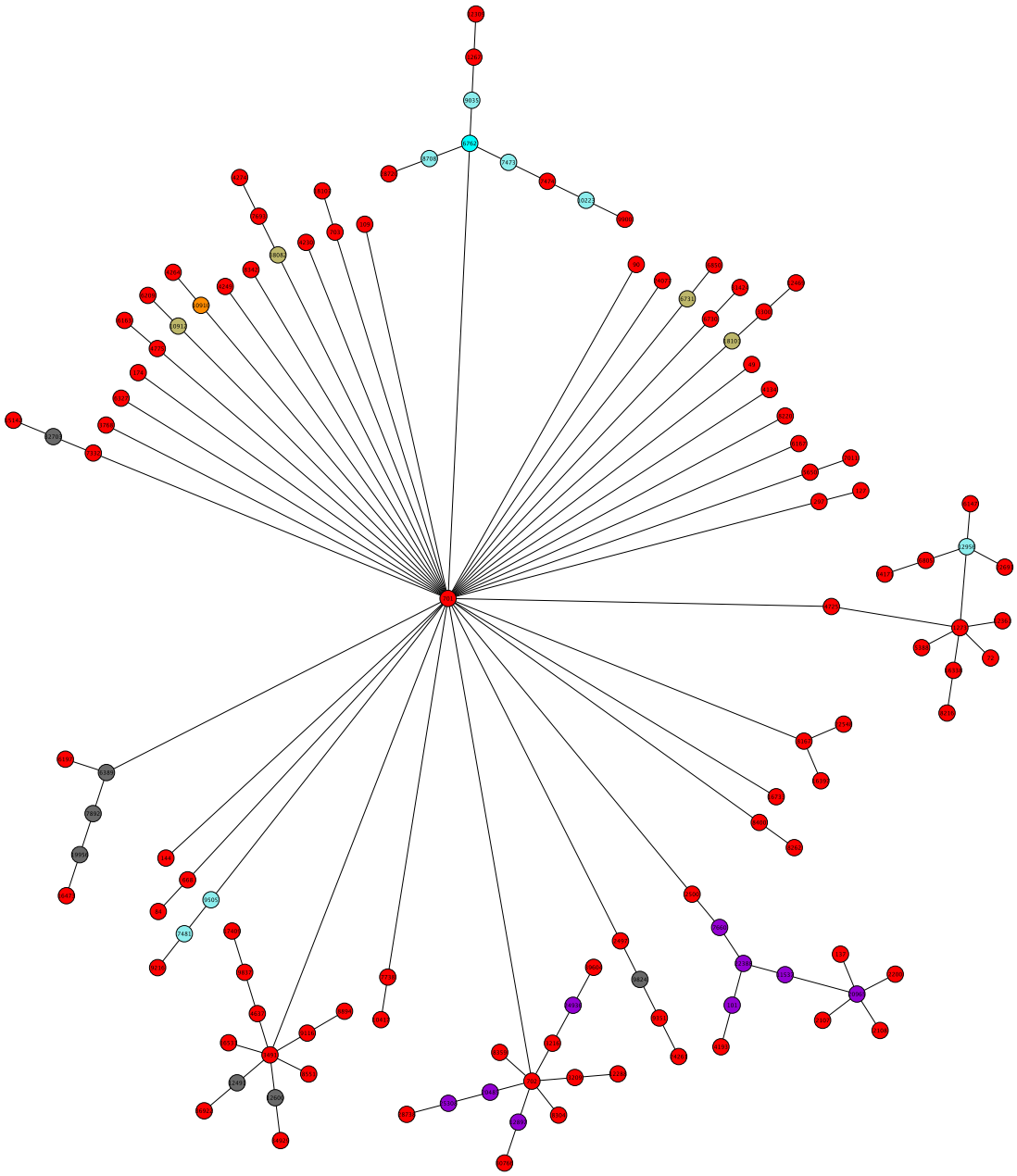
Γράφος Θ

- *120 Κόμβοι*
- *90 κακόβουλοι χρήστες (δεν λειτούργησαν όλοι)*
- *5 εικονικά μηχανήματα*



partition graph

Σχήμα 6.7.1: Γράφος Θ – 120 κόμβοι| 90 κακόβουλοι | 5 VMs



malicious hosts graph

Σχήμα 6.7.2: Γράφος Θ – 120 κόμβοι| 90 κακόβουλοι | 5 VMs

6.2 Μετρήσεις

Για τις παρακάτω μετρήσεις χρησιμοποιήθηκαν εικονικά μηχανήματα με τα εξής χαρακτηριστικά:

CPU	RAM	DISK
1	1024 MB	5 GB

Σχήμα 6.8: Χαρακτηριστικά εικονικών μηχανημάτων

Σηκώνοντας σε 2 εικονικά μηχανήματα από έναν host στο καθένα παρατηρούμε ότι:

VMs	Tunnel	Bandwidth
2	VXLAN	420 Mbps
2	GRE	400 Mbps

Σχήμα 6.9: Δοκιμές VXLAN & GRE

Παρακάτω βρίσκονται οι μετρήσεις που πραγματοποιήθηκαν για τους γράφους που παρουσιαστήκαν παραπάνω. Από την $3^{η}$ στήλη και μετά τα αποτελέσματα εμφανίζονται σε 2 τιμές: η $1^{η}$ αφορά τους hosts που βρίσκονται στο ίδιο vm με το victim (όπου οι ταχύτητες είναι φυσικά μεγαλύτερες καθώς τα πακέτα ανταλλάζονται μεταξύ host που βρίσκονται στο ίδιο μηχάνημα) ενώ η $2^{η}$ αφορά όσους στέλνουν κίνηση από διαφορετικό μηχάνημα από του victim και διασχίζουν μία ή περισσότερες vxlan ακμές ώστε να του επιτεθούν.

Στον παρακάτω πίνακα έχει γίνει μια διαφοροποίηση ανάμεσα στην κίνηση που μεταφέρεται από τους κακόβουλους hosts προς το θύμα -οι οποίοι βρίσκονται στο ίδιο VM με το θύμα- και στην κίνηση που μεταφέρεται από τους κακόβουλους hosts προς το θύμα, οι οποίοι ανήκουν σε διαφορετικό VM από αυτό. Ο διαχωρισμός αυτός είναι απαραίτητος καθώς οι διαφορές στις ταχύτητες ανταλλαγής πακέτων

μεταξύ hosts του ίδιου μηχανήματος και μεταξύ hosts διαφορετικών μηχανημάτων είναι τεράστια.

VMs	*Nodes*	*Malicious hosts intra-victim's VM*	*Malicious hosts inter-victim's VM*	*Avg. Tr. per host intra-victim's VM*	*Avg. Tr. per host inter-victim's VM*	*Band. per host (Mbps) intra-victim's VM*	*Band. Per host (Mbps) inter-victim's VM*
2	**22**	**06**	**09**	**3.90 GB**	**500.0 MB**	**285.6**	**31.67**
3	**22**	**05**	**10**	**3.05 GB**	**366.0 MB**	**220.0**	**25.50**
5	**22**	**01**	**14**	**26.6 GB**	**340.5 MB**	**19100**	**23.8**
3	**38**	**12**	**16**	**1.35 GB**	**213.0 MB**	**96.9**	**14.85**
4	**38**	**07**	**21**	**2.54 GB**	**235.5 MB**	**181.5**	**15.00**
5	**38**	**07**	**07**	**2.70 GB**	**244.5 MB**	**194**	**17.05**
5	**61**	**09**	**31**	**2 GB**	**195 MB**	**140.2**	**13.8**
5	**120**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**	**Xxx**
5	**200**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**

Σχήμα 6.10: Μετρήσεις

Με Gre tunnelling αντί για Vxlan (ενδεικτικά**):**

VMs	*Nodes*	*Malicious hosts intra-victim's VM*	*Malicious hosts inter-victim's VM*	*Avg. Tr. per host intra-victim's VM*	*Avg. Tr. per host Inter-victim's VM*	*Band. per host (Mbps) intra-victim's VM*	*Band. per host (Mbps) inter-victim's VM*
5	**22**	**01**	**14**	**26.6 GB**	**340 MB**	**19100**	**23.0**
5	**38**	**07**	**21**	**2.9 GB**	**210 MB**	**213.5**	**13.9**

Σχήμα 6.10: Ενδεικτικές διαφορές VXLAN-GRE

6.3 Συμπεράσματα

Από τους παραπάνω πίνακες μπορεί να παρατηρηθεί αρχικά ότι με την χρησιμοποίηση του VXLAN tunnelling τα αποτελέσματα είναι ελαφρώς καλύτερα, όπως ακριβώς είχαν υπολογιστεί ότι θα είναι. Η προσαρμοστικότητα του εργαλείου ως προς την ευκολία της αλλαγής του τύπου του tunnelling που θα χρησιμοποιηθεί κάνει το ίδιο το εργαλείο εξαιρετικά χρήσιμο σε κάποιον που θέλει να ερευνήσει τις τεχνολογίες αυτές.

Στην συνέχεια, παρατηρείται ότι η ταχύτητα ανταλλαγής δεδομένων είναι ικανοποιητική και αυτό σημαίνει ότι μπορούν να επιτευχθούν εξομοιώσεις (είτε επιθέσεων είτε άλλων σεναρίων) όπου οι ταχύτητες θα ρυθμίζονται στο ρεαλιστικό σενάριο που εξομοιώνεται.

Άλλο ενδιαφέρον κομμάτι είναι ότι η απόδοση αυξομειώνεται με την προσθήκη εικονικών μηχανημάτων. Δηλαδή, η απόδοση μπορεί να μειωθεί αν ο γράφος μοιραστεί σε πάρα πολλά κομμάτια, γεγονός απόλυτα λογικό αφού το αριθμός των tunnels που θα πρέπει να διασχίσει ένα πακέτο προκειμένου να φτάσει στον προορισμό του αυξάνεται. Αντίστοιχα όμως, η απόδοση μπορεί να μειωθεί αν ο γράφος μοιραστεί σε μικρό αριθμό εικονικών μηχανημάτων. Στην περίπτωση αυτή το κάθε εικονικό μηχάνημα αναλαμβάνει να χτίσει υπογράφο τόσο μεγάλου μεγέθους που οι πόροι του δεν αντέχουν την κατασκευή του. Αυτό συνέβη για παράδειγμα, όταν έγινε η προσπάθεια να χτιστεί γράφος 200 και 120 κόμβων σε 5 εικονικά μηχανήματα (τελευταίες 2 μετρήσεις του πίνακα). Η εκτέλεση της εξομοίωσης στις περιπτώσεις αυτές έγινε αλλά όχι πλήρης και όχι με τρόπο τέτοιο που να την καταστήσει χρήσιμη για τις μετρήσεις που ήταν επιθυμητό να πραγματοποηθούν.

Συνοψίζοντας, μέσα από την διπλωματική αυτή επιτεύχθηκαν οι εξής βασικοί στόχοι:

➔ Φάνηκε ξεκάθαρα η χρησιμότητα και η σημασία της απεικόνισης των γράφων κατανεμημένης επίθεσης άρνησης παροχής υπηρεσίας. Μέσα από τις απεικονίσεις αυτές παρουσιάστηκε το γεγονός ότι κάποιοι κόμβοι είναι πιο «σημαντικοί» από κάποιους άλλους και έτσι για την μελέτης της άμυνας κατά των επιθέσεων αυτών θα μπορούσαν να μελετηθούν και να αναλυθούν τα διάφορα μοτίβα που εμφανίζονται σε τέτοιους γράφους.

➔ Κατασκευάστηκε ένα εργαλείο γρήγορο και σχετικά εύκολο στην χρήση, μέσω του οποίου δίνεται η ευκαιρία στον εκάστοτε ερευνητή που έχει πρόσβαση σε εικονικά μηχανήματα να μελετήσει γράφους ρεαλιστικού μεγέθους και να εξομοιώσει σενάρια του πραγματικού κόσμου. Το εργαλείο είναι χωρισμένο σε τέτοια κομμάτια ώστε να ευνοεί τις αλλαγές και να είναι εύκολα προσαρμόσιμο στις διαφορετικές ανάγκες του κάθε ερευνητή.

➔ Βρέθηκε ένας τρόπος διαμοιρασμού τοπολογιών με ώστε να επιτυγχάνονται όσο το δυνατόν μεγαλύτερες ταχύτητες ως προς την αποστολή και λήψη δεδομένων μεταξύ των κόμβων του δικτύου.

➔ Επιτεύχθηκε ο διαμοιρασμός τοπολογιών σε πολλαπλές cloud υποδομές. Με τον συνδυασμό λοιπόν διαφόρων τέτοιων υποδομών είναι δυνατό να εξομοιωθούν πραγματικά μεγάλου μεγέθους γράφοι, όχι αναγκαστικα με σκοπό την εξομοίωση κατανεμημένων επιθέσεων άρνησης παροχής υπηρεσίας αλλά και για οποιονδήποτε άλλο σκοπό.

6.4 Μελλοντικές επεκτάσεις

Η παρούσα εργασία γίνεται να «σπάσει» σε αρκετά μέρη τα οποία δέχονται βελτιστοποίηση.

Μια σημαντική προσθήκη θα είναι να προστεθεί μια λειτουργία με την χρήση της οποίας θα μπορεί να διαμορφωθεί το μέγεθος της συνολικής κίνησης που φτάνει στον edge δρομολογητή του ΑΣ στον οποίο βρίσκεται το θύμα. Αυτό είναι απαραίτητο, διότι στην περίπτωση που ο εκάστοτε ερευνητής ασχοληθεί με πραγματικά μεγάλους γράφους, η κίνηση που θα φτάνει στο edge router του ΑΣ του θύματος μπορεί να προκαλέσει αληθινή καθυστέρηση στους υπόλοιπους χρήστες που εξυπηρετούνται από αυτό το ΑΣ καθώς το router μπορεί να μην μπορεί να διαχειριστεί τόσο πολύ κίνηση.

Μια άλλη βελτιστοποίηση που μπορεί να γίνει αφορά τον τρόπο που διαμοιράζονται οι κόμβοι της συνολικής τοπολογίας στις διάφορες υποδομές cloud.

Στην παρούσα εργασία, έγινε μελέτη πάνω στον διαμοιρασμό της τοπολογίας σε σχέση με τον αριθμό εικονικών μηχανημάτων χωρίς να υπολογισθεί ότι σημαντικό ρολο «παίζει» και το ποια εικονικά μηχανήματα ανήκουν στις ίδιες υποδομές cloud. Μια επέκταση λοιπόν, θα είναι να δέχεται το εργαλείο ως πληροφορία και τον αριθμό των εικονικών μηχανημάτων ανά υποδομή cloud και να χρησιμοποιεί την πληροφορία αυτή με σκοπό την ελαχιστοποίηση του αριθμού των ακμών μεταξύ των διαφορετικών υποδομών σε συνάρτηση με την ελαχιστοποίηση του αριθμού των ακμών μεταξύ των εικονικών μηχανημάτων που βρίσκονται στην ίδια υποδομή.

Επιπρόσθετα μπορεί να γίνει μια μελέτη γύρω από το tunnelling. Με την χρήση πιο εξελιγμένων εικονικών routers μπορούν να μελετηθούν και άλλα πρωτόκολλα tunnelling όπως το Stateless Transport Tunnelling (STT).

Τέλος, θα ήταν σαφέστατα χρήσιμη η δημιουργία ενός GUI ώστε το εργαλείο αυτό να είναι ακόμα πιο εύχρηστο και να μπορεί να χρησιμοποιηθεί από τον οποιονδήποτε.

Σε γενικότερα πλαίσια η παρούσα εργασία ασχολείται με διάφορους τομείς εξειδίκευσης με αποτέλεσμα να μπορεί να προσεγγιστεί από διάφορες σκοπιές και να βελτιωθεί αντιστοίχως.

Βιβλιογραφία

[1] N. Feamster, J. Rexford, and E. Zegura, “The road to sdn: An intellectual history of programmable networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 87–98, Apr. 2014. [Online]. Available: http://doi.acm.org/10.1145/2602204.2602219

[2] D. L. Tennenhouse and D. J. Wetherall, “Towards an active network architecture,” *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 5, pp. 81–94, Oct. 2007. [Online]. Available: http://doi.acm.org/10.1145/1290168.1290180

[3] A. T. Campbell, H. G. De Meer, M. E. Kounavis, K. Miki, J. B. Vicente, and D. Villela, “A survey of programmable networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 2, pp. 7–23, Apr. 1999. [Online]. Available: http://doi.acm.org/10.1145/505733.505735

[4] Open Networking Foundation, "Software-Defined Networking : The New Norm for Networks," *ONF White Paper,* 2012.

[5] [Online]. OpenFlow protocol, http://archive.openflow.org.

[6] [Online]. OpenFlow protocol http://archive.openflow.org/documents/openflow-wp-latest.pdf

[7] Open Networking Foundation, "OpenFlow Switch Specification Version 1.0". *https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.0.0.pdf.*

[8] [Online]. http://mininet.org/

[9] Bob Lantz, Brandon Heller, Nick McKeown, "*A Network in a Laptop: Rapid Prototyping for Software-Defined Networks*"

[10] [Online]. Available: http://www.arin.net/whois/arinwhois.html.

[11] J. Rickard, "Mapping the Internet with traceroute," *Broadwatch Mag.*, Dec. 1996.

[12] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 733–745, Dec. 2001. [Online]. Available: http://dx.doi.org/10.1109/90.974527.

[13] The CAIDA AS Relationships Dataset, <2015>., http://www.caida.org/data/as-relationships/.

[14] The CAIDA UCSD "DDoS Attack 2007" Dataset, http://www.caida.org/data/ passive/ddos-20070804_dataset.xml.

[15] [Online], http://www.routeviews.org/.

[16] X. Dimitropoulos, D. Krioukov, M. Fomenkov, B. Huffaker, Y. Hyun, k. claffy, and G. Riley, "AS Relationships: Inference and Validation," *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 37, no. 1, pp. 29–40, Jan 2007.

[17] X. Dimitropoulos, D. Krioukov, B. Huffaker, k. claffy, and G. Riley, "Inferring AS Relationships: Dead End or Lively Beginning?" Santorini, Greece, pp. 113–125, May 2005.

[18][Online],http://www.cc.gatech.edu/computing/Telecomm/projects/cryptopan/

[19] Jinliang Fan, Jun Xu, Mostafa H. Ammar, Sue Moon, "*Prefix-Preserving IP Address Anonymization*", Computer Networks, Volume 46, Issue 2 , 7 October 2004, Pages 253-272, Elsevier.

[20] Jinliang Fan, Jun Xu, Mostafa H. Ammar, Sue Moon, "*On the Design and Performance of Prefix-Preserving IP Traffic Trace Anonymization*", ACM SIGCOMM Internet Measurement Workshop 2001, San Francisco, CA, November , 2001.

[21] Jinliang Fan, Jun Xu, Mostafa H. Ammar, Sue Moon, "*Prefix-Preserving IP Address Anonymization: Measurement-based Security Evaluation and a New Cryptography-based Scheme*", to appear in Proceedings of the IEEE International Conference on Network Protocols, Paris, 2002.

[22] [Online]. https://pypi.python.org/pypi/pyasn.

[23] [Online]. https://github.com/hadiasghari/pyasn.

[24] [Online].http://data.4tu.nl/repository/uuid:d4d23b8e-2077-4592-8b47-cb476ad16e12.

[25] [Online]. https://github.com/MiliasV/Project-D/blob/master/step1/ip_to_as.py

[26] [Online]. https://github.com/MiliasV/Project-D/blob/master/step1/mh.py

[27] [Online]. https://github.com/MiliasV/Project-D/blob/master/ step2/tree_of_attack.py.

[28] D. Magoni and J. J. Pansiot, “Analysis of the autonomous system network topology,” *SIGCOMM Comput. Commun. Rev.*, vol. 31, no. 3, pp. 26–37, Jul. 2001. [Online]. Available: http://doi.acm.org/10.1145/505659.505663

[29] [Online]. https://networkx.github.io/

[30] George Karypis and Vipin Kumar, “A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs”. SIAM Journal on Scientific Computing, Vol. 20, No. 1, pp. 359—392, 1999.

[31] [Online], http://glaros.dtc.umn.edu/gkhome/metis/metis/overview

[32] https://github.com/MiliasV/ProjectD/blob/master/ step3/from_paths_to_vms.py

[33] [Online], http://dl9obn.darc.de/programming/python/dottoxml/

[34] [Online], http://www.fabfile.org/

[35] [Online], https://github.com/MiliasV/Project-D/blob/master/
step4/min_builder.py

[36] [Online], https://github.com/MiliasV/Project-D/blob/master/step4/fabfile.py