ΔΙΑΧΕΙΡΗΣΗ ΔΙΚΤΥΩΝ – ΤΕΛΙΚΗ ΕΡΓΑΣΙΑ

ΠΡΟΣΟΜΟΙΩΣΗ ΔΙΚΤΥΟΥ LAN, ΠΡΟΣΘΗΚΗ FIREWALL ΚΑΙ ΕΥΡΕΣΗ ΒΕΛΤΙΣΤΩΝ ΔΙΑΔΡΟΜΩΝ

ΟΜΑΔΑ:

ΒΛΑΧΟΥ ΜΕΛΙΝΑ-ΜΑΡΙΑ,

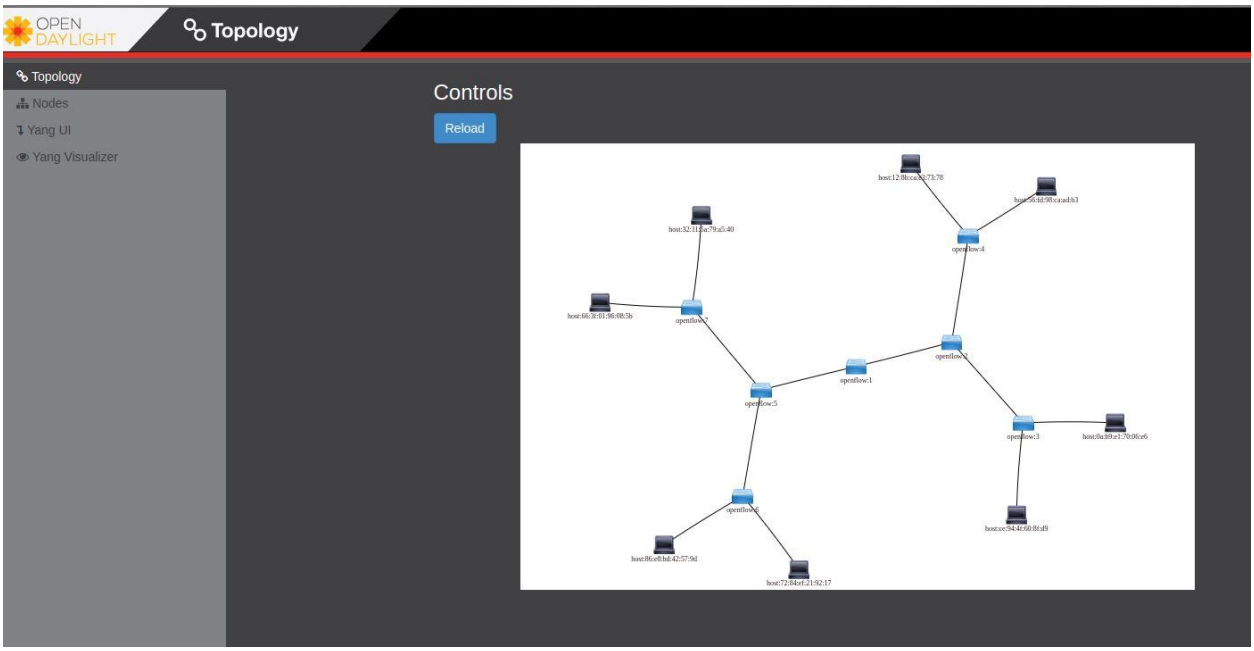
ΚΑΡΑΝΙΚΟΛΑΣ ΘΑΝΑΣΗΣ,

ΤΑΒΟΥΛΑΡΗΣ ΦΩΤΗΣ-ΕΚΤΟΡΑΣ

Εισαγωγη:

Στην εργασια αυτη υλοποιειται η προσωμοιωση ενος δικτυου LAN το οποιο αποτελειται απο εικονικους hosts και switches. Οι τοπολογιες του δικτυου μας δημιουργουνται στα αρχεια topology.py και topology2.py μεσω του εργαλειου mininet και οπτικοποιουνται μεσω του εργαλειου OpenDayLight. Υπαρχουν δυο τοπολογιες, μια για το καθε application που υλοποιηθηκε. Τα δυο applications ειναι τα Firewall και ευρεση βελτιστης διαδρομης με τον αλγοριθμο Bellman Ford οπου αναλυονται παρακατω.

- Η 1η τοπολογια (topology.py) στην οποια εφαρμοζεται το Firewall αποτελειται απο 8 κομβους (hosts) που ο καθενας εχει την δικη του IP της μοφης '10.0.0.X' και απο 7 μεταγωγεις (switches) οι οποιοι οργανωνονται σε 2 layers. Αν ανοιξουμε το OpenDayLight (http://localhost:8181/index.html#/topology) αφου εχουμε τρεξει το topology.py θα δουμε το παρακατω:



- Η 2η (topology2.py) αποτελειται απο 2 hosts και 7 switches.

Οδηγιες για εκτελεση:

Απαραιτητη προυποθεση ειναι να υπαρχουν εγκατεστημενα τα Mininet, OpenDayLight, Python 2.7 τα οποια εγκαταστησαμε συμφωνα με τις οδηγιες στις διαφανειες.

Για την 1η τοπολογια και το firewall:

- To start the firewall: `./run.sh`
- To create the topology: `sudo python topology.py`
- To start OpenDayLight: `./bin/karaf –of13` (inside distribution-karaf-0.5.4-Boron-SR4/)

Για την 2η τοπολογια και το Bellman Ford:

- To start the Bellman Ford algorithm: `./bfrun.sh`
- To create the topology: `sudo python topology2.py`
- To start OpenDayLight: `./bin/karaf –of13` (inside distribution-karaf-0.5.4-Boron-SR4/)

Εαν επιθυμειται να ξανατρεξει καποια τοπολογια προτεινουμε ενδιαμεσα :

- `./clean.sh`

Εχοντας ξεκινησει τις δυο τοπολογιες για να ελεγξουμε οτι λειτουργουν μπορουμε να κανουμε τα εξης:

mininet> pingall (διαπιστωνουμε ποιο host επικοινωνει με ποιο)

mininet> h1 ping h2 (διαπιστωνουμε αν το h1 μπορει να στειλει πακετα στο h2 για παραδειγμα, αντιστοιχα και για αλλα hosts)

- Στην περιπτωση του firewall θα δουμε οτι (συμφωνα με κανονες που εχουμε ορισει) καποια hosts δεν επικοινωνουν με καποια αλλα γιατι θεωρουνται ως κακοβουλα και δημιουργειται αμοιβαιος αποκλεισμος τους. Οποτε με την πρωτη εντολη θα βλεπαμε ενα X σε αυτα τα hosts και με την δευτερη εφοσον το h1 με h2 ειναι αποκλεισμενα, θα βλεπαμε οτι δεν παραδωθηκε κανενα πακετο. Ολα αυτα παραθετονται με φωτογραφιες παρακατω.
- Στην περιπτωση του Bellman Ford θα αρχισουμε να βλεπουμε πακετα απ το ενα host στο αλλο και θα εκτυπωθει η βελτιστη διαδρομη τοσο απο το 1 στο 2 οσο και απο το 2 στο 1

Αυτο που επιτυγχανουμε ειναι σε καθε περιπτωση οσοι hosts επιθυμουμε να μπορουν να στελνουν πακετα μηνυματων μεταξυ τους με την βοηθεια των μεταγωγεων που βρισκονται αναμεσα τους και παιρνουν το μηνυμα απο τη θυρα εξοδου του αποστολεα και το μεταφερουν στην θυρα εισοδου του παραληπτη

APPLICATION 1- FIREWALL

Το firewall ειναι ενα τοιχος προστασιας που υπαρχει σε καθε host ωστε να προστατευει τον «υπολογιστη» απο κακοβουλους χρηστες. Η δουλεια του ειναι με προκαθορισμενα κριτηρια να επιτρεπει η να απορριπτει(μπλοκαρει) χρηστες που προσπαθουν να επικοινωνησουν με τον καθε host. Ολες οι πληροφοριες (μηνυματα) που εισερχονται η εξερχονται απο ενα δικτυο περνουν μεσα απο αυτο το τοιχος προστασιας.

Στη συνεχεια βλεπουμε πως το δικο μας firewall εχει αποτρεψει την επικοινωνια καποιων hosts συμφωνα με τους κανονες που εχουμε θεσει(οι οποιοι λαμβανονται απ το firewall μεσω του firewallpolicies.csv):

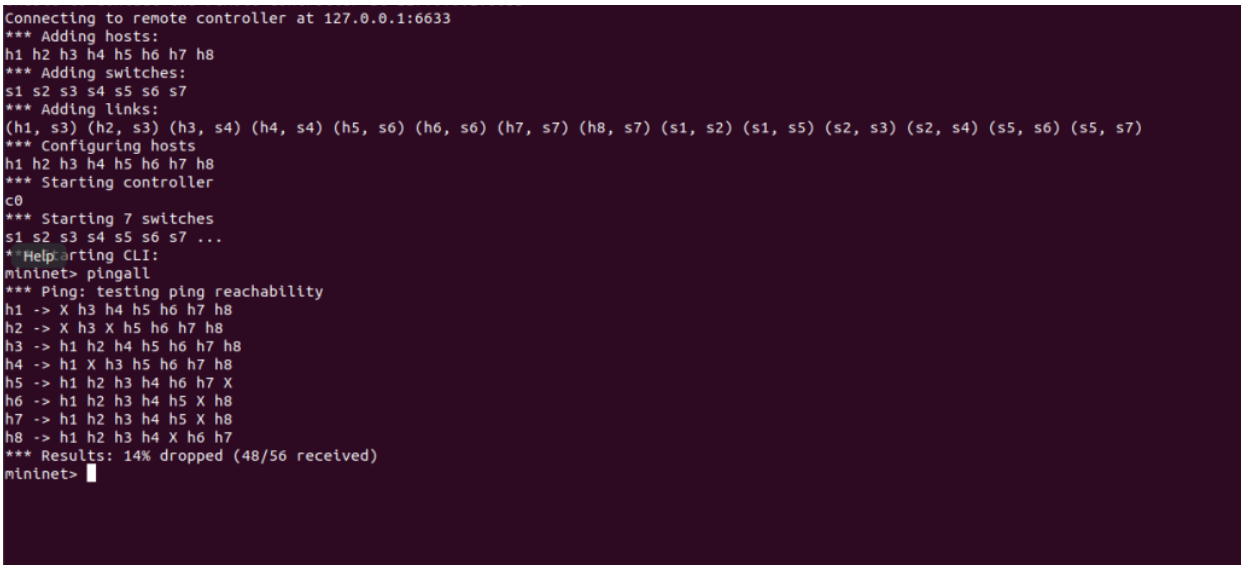
1	10.0.0.1	10.0.0.2
2	10.0.0.2	10.0.0.4
3	10.0.0.5	10.0.0.8
4	10.0.0.6	10.0.0.7

Μετα την εκτελεση του run.sh οι κανονες αυτοι φαινονται ως εξης:

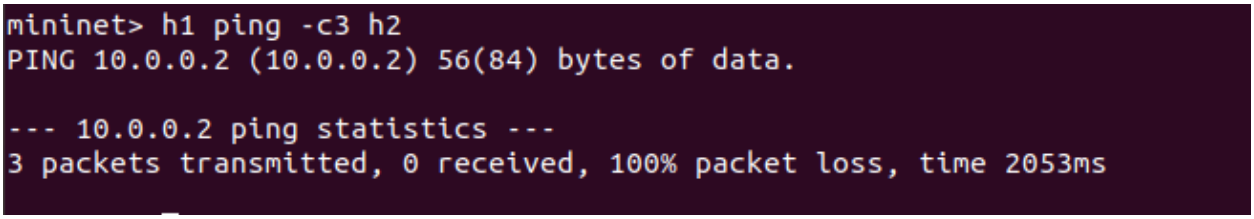
```
INFO:openflow.of_01:[00-00-00-00-00-01 5] connected
DEBUG:forwarding.l2_learning:Connection [00-00-00-00-00-01 5]
INFO:misc.firewall:Adding rule: source 10.0.0.1 - destination 10.0.0.2
INFO:misc.firewall:Adding rule: source 10.0.0.2 - destination 10.0.0.4
INFO:misc.firewall:Adding rule: source 10.0.0.5 - destination 10.0.0.8
INFO:misc.firewall:Adding rule: source 10.0.0.6 - destination 10.0.0.7
```

Εχουμε αποτρεψει την επικοινωνια δηλαδη μεταξυ: **h1-h2, h2-h4, h5-h8, h6-h7**

Κανοντας pingall εχουμε:



οπου παρατηρουμε στα αποκλεισμενα hosts ενα X στη θεση τους αντιστοιχα. Αν στη συνεχεια δοκιμασουμε να κανουμε ping στα 1 και 2 στελνοντας 3 πακετα θα δουμε:



Οτι δηλαδη κανενα πακετο δε μπορεσε να φτασει απο το 1 στο 2 οπως και θα επρεπε. Συμπεραινουμε λοιπον τη σωστη λειτουργια του firewall.

APPLICATION 2- SHORTEST PATH WITH BELLMAN FORD

Η εφαρμογη αυτη βρισκει την συντομοτερη διαδρομη απο εναν host1 σε εναν host2 μεταξυ των μεταγωγεων που παρεμβαλονται αναμεσα τους συμφωνα με τον αλγοριθμο Bellman Ford και σε καθε βημα αποθηκευει τα switches και ports τους ωστε να βρει το επιθυμητο path.

Αρχικα, κανοντας pingall επιβεβαιωνουμε οτι η επικοινωνια εχει πετυχει:

```
*** Creating network
*** Adding controller
Connecting to remote controller at 127.0.0.1:6653
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(h1, s1) (s1, s2) (s1, s3) (s2, s4) (s2, s5) (s3, s6) (s3, s7) (s7, h2)
(s7, s5)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
mininet>
```

Μετα την εκτελεση του bellmanFord.py βλεπουμε στην οθονη τη συντομοτερη διαδρομη οπως φαινεται παρακατω:

```
~~~~~~~~~~~~~~~~~~~~~~
[01 - 07]
Path:
[00-00-00-00-00-01, 00-00-00-00-00-03, 00-00-00-00-00-07]
~~~~~~~~~~~~~~~~~~~~~~
```

Εδω λοιπον βλεπουμε οτι η διαδρομη που επελεξε ο αλγοριθμος ειναι η **s1-s3-s7**

Οπου συμφωνα με το διαγραμμα της τοπολογιας που εχουμε φτιαξει ειναι σωστο: