



ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΗΜΜΥ. ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ

ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ ΙΙ

Αυτοματοποιημένη Ανάπτυξη Τοπολογίας και Ανάλυση Πρωτοκόλλων στο Mininet

Συντάκτης:

Χρήστος Χουτουρίδης [8997]

cchoutou@ece.auth.gr

Διδάσκοντες:

Αναστάσιος Ντελόπουλος

antelopo@ece.auth.gr

Εμανουήλ Τσαρδουλιάς

etsardou@ece.auth.gr

27 Σεπτεμβρίου 2026

1. ΕΙΣΑΓΩΓΗ

Η παρούσα εργασία έχει ως αντικείμενο την **αυτοματοποιημένη ανάπτυξη δικτυακών τοπολογιών στο περιβάλλον Mininet**, με βάση μια δηλωτική περιγραφή σε αρχείο JSON. Στο πλαίσιο αυτό υλοποιείται εφαρμογή σε Python, η οποία διαβάζει περιγραφές της τοπολογίας σε μια πρότυπη **Domain-Specific Language – DSL**, και δημιουργεί δυναμικά τους αντίστοιχους hosts, μεταγωγείς, συνδέσμους και δρομολογητές και εφαρμόζει τις απαιτούμενες ρυθμίσεις δικτύου. Το Mininet χρησιμοποιείται ως περιβάλλον εξομίωσης, ώστε η τοπολογία να μπορεί να κατασκευάζεται και να εξετάζεται με χρήση των ίδιων εργαλείων και πρωτοκόλλων που χρησιμοποιούνται σε ένα πραγματικό σύστημα Linux.

Μετά την ανάπτυξη της τοπολογίας πραγματοποιούνται τρία πειράματα χαμηλού επιπέδου για τη μελέτη των πρωτοκόλλων **DHCP** και **ARP**, καθώς και της **δρομολόγησης μεταξύ διαφορετικών υποδικτύων**. Μέσω καταγραφών με το `tcpdump` εξετάζεται η διαδικασία D.O.R.A. του DHCP, η αντιστοίχιση διευθύνσεων IP σε διευθύνσεις MAC μέσω του ARP και η μεταβολή των Ethernet διευθύνσεων και του TTL κατά τη διέλευση ενός IP πακέτου από τον δρομολογητή. Στόχος των πειραμάτων είναι η σύνδεση της θεωρητικής λειτουργίας των πρωτοκόλλων με την πραγματική μορφή των πακέτων που ανταλλάσσονται στα διαφορετικά επίπεδα του δικτύου.

1.1. Παραδοτέα

Τα παραδοτέα της εργασίας αποτελούνται από:

- Την παρούσα τεχνική αναφορά.
- Το αρχείο [exercise/topology.json](#), το οποίο περιγράφει δηλωτικά την τοπολογία δικτύου που χρησιμοποιείται στην εργασία.
- Τον κατάλογο `source/`, ο οποίος περιέχει το αρχείο [build_network.py](#), καθώς και τα υπόλοιπα αρχεία Python που υλοποιούν την εφαρμογή.
- Τον κατάλογο [exercise/results/](#), ο οποίος περιέχει τις μετρήσεις και τα παραγόμενα αρχεία των πειραμάτων της εργασίας.
- Τους συνδέσμους προς τα αποθετήρια στο [GitHub](#) και στον [προσωπικό μου server](#), τα οποία περιέχουν τον πηγαίο κώδικα, τον κώδικα παραγωγής της αναφοράς και τα αποτελέσματα των πειραμάτων.

2. ΥΛΟΠΟΙΗΣΗ ΤΟΥ Builder

2.1. Δηλωτική γλώσσα περιγραφής τοπολογίας

Για την περιγραφή των δικτυακών τοπολογιών σχεδιάστηκε μία μικρή **δηλωτική γλώσσα ειδικού σκοπού** (Domain-Specific Language – DSL), η οποία χρησιμοποιεί το JSON ως μορφή αναπαράστασης. Στόχος της γλώσσας είναι να περιγράφει την *επιθυμητή κατάσταση του δικτύου* και όχι τη σειρά των εντολών που απαιτούνται για την κατασκευή του στο Mininet. Με αυτόν τον τρόπο η περιγραφή της τοπολογίας παραμένει ανεξάρτητη από λεπτομέρειες της υλοποίησης, όπως τα ονόματα των εικονικών διεπαφών ή η ακριβής ακολουθία κλήσεων προς το Mininet API. Η αναλυτική σχεδίαση και η γραμματική της γλώσσας τεκμηριώνονται επίσης στο αρχείο [Mininet_DSL_Design.md](#) του αποθετηρίου.

2.1.1. Σχεδιαστική προσέγγιση

Ο σχεδιασμός της γλώσσας πραγματοποιήθηκε **από πάνω προς τα κάτω**, ξεκινώντας από τις έννοιες που είναι φυσικό να χρησιμοποιούνται για την περιγραφή ενός δικτύου, όπως οι *κόμβοι*, οι *διεπαφές*, οι *σύνδεσμοι* και οι *υπηρεσίες*. Παράλληλα, κάθε αφηρημένη έννοια επιλέχθηκε έχοντας υπόψη τις διαθέσιμες πρωτογενείς λειτουργίες του Mininet και του δικτυακού υποσυστήματος του Linux. Η προσέγγιση αυτή επιτρέπει στη γλώσσα να παραμένει υψηλού επιπέδου, χωρίς να εισάγει κατασκευές που δεν μπορούν να μεταφραστούν με σαφή και συστηματικό τρόπο στο υποκείμενο περιβάλλον εκτέλεσης.

Οι βασικές λειτουργίες που απαιτούνται από την εργασία μπορούν να αναχθούν σε ένα σχετικά μικρό σύνολο πρωτογενών κατασκευών του Mininet και του Linux. Η δημιουργία hosts και switches, η σύνδεσή τους μέσω links και η εκτέλεση εντολών μέσα στους αντίστοιχους network namespaces επαρκούν για την κατασκευή της απαιτούμενης τοπολογίας. Ένας δρομολογητής μπορεί αντίστοιχα να μοντελοποιηθεί ως host με περισσότερες από μία διεπαφές, κατάλληλη Layer-3 διευθυνσιοδότηση και ενεργοποιημένη προώθηση IP. Ο DHCP server μπορεί να θεωρηθεί ως υπηρεσία που εκτελείται σε έναν κόμβο και συνδέεται με συγκεκριμένη διεπαφή.

Πριν οριστικοποιηθούν οι οντότητες της DSL, εξετάστηκαν οι βασικές λειτουργίες που παρέχει το Mininet και το δικτυακό υποσύστημα του Linux για την κατασκευή και παραμετροποίηση μιας τοπολογίας. Οι κύριες λειτουργίες που χρησιμοποιήθηκαν ως βάση είναι οι `addHost()`, `addSwitch()` και `addLink()`, καθώς και η δυνατότητα εκτέλεσης εντολών μέσα σε κάθε network namespace μέσω του `node.cmd()`. Στις λειτουργίες αυτές προστίθενται η παραμετροποίηση διεπαφών και δρομολόγησης μέσω εργαλείων του Linux και η εκκίνηση διεργασιών, όπως ο `dnsmasq`, μέσα στον αντίστοιχο κόμβο. Η ανάλυση αυτή χρησιμοποιήθηκε για να **σταθεροποιηθεί το σύνολο των αφαιρέσεων της DSL**, έτσι ώστε κάθε υψηλού επιπέδου κατασκευή να έχει σαφή δρόμο μετάφρασης προς τις διαθέσιμες λειτουργίες του backend.

Με βάση τις παρατηρήσεις αυτές επιλέχθηκε ένα σύνολο οντοτήτων ώστε η DSL να **καλύπτει τον χώρο των κατασκευών που απαιτούνται από την εργασία**. Αντίστοιχα, το σύνολο των primitives του Mininet και του Linux που χρησιμοποιείται είναι επαρκές ώστε κάθε κατασκευή της DSL να μπορεί να μεταφραστεί σε συγκεκριμένες λειτουργίες του backend. Με αυτή την έννοια, τόσο η γλώσσα ως προς τον χώρο του προβλήματος όσο και τα χρησιμοποιούμενα primitives ως προς τον χώρο της υλοποίησης λειτουργούν «επί». Δηλαδή, δεν παραμένει απαίτηση της εργασίας χωρίς αναπαράσταση στη DSL ούτε κατασκευή της DSL χωρίς αντίστοιχη υλοποίηση στο Mininet και στο Linux. Οι βασικές οντότητες της γλώσσας είναι οι *κόμβοι*, οι *διεπαφές*, οι *σύνδεσμοι*, οι *ρυθμίσεις κόμβων* και οι *υπηρεσίες*.

2.1.2. Βασική γραμματική

Η τοπολογία περιγράφεται από τρεις κύριες κατηγορίες αντικειμένων: **nodes**, **links** και **services**. Η βασική δομή της γλώσσας μπορεί να συνοψιστεί ως εξής:

```
Network
|-- Nodes
|   |-- name
|   |-- type
|   |-- interfaces
|   `-- settings
|
|-- Links
|   |-- name
|   |-- endpoints
|   |   |-- node
|   |   `-- interface
|   `-- settings
|
`-- Services
    |-- type
    |-- node
    |-- interface
    `-- settings
```

Κάθε κατηγορία έχει **διαφορετική σημασιολογική ευθύνη**. Ένας κόμβος(Node) αναπαριστά έναν συμμετέχοντα του δικτύου, μία διεπαφή(interface) ένα σημείο σύνδεσης που ανήκει σε έναν κόμβο και ένας σύνδεσμος(link) συνδέει δύο τέτοια σημεία. Οι ρυθμίσεις(settings) κόμβου περιγράφουν ιδιότητες του ίδιου του

δικτυακού namespace, ενώ οι υπηρεσίες (services) αναπαριστούν ανεξάρτητες διεργασίες που εκτελούνται μέσα σε αυτό.

2.1.3. Κόμβοι και διεπαφές

Οι κόμβοι δηλώνονται με ένα μοναδικό λογικό όνομα και έναν τύπο. Οι τύποι που υποστηρίζονται στην παρούσα υλοποίηση είναι `host`, `switch` και `router`. Ο τύπος εκφράζει τη *σημασιολογία του κόμβου* και όχι απαραίτητα το συγκεκριμένο αντικείμενο του Mininet που θα χρησιμοποιηθεί για την υλοποίησή του.

Οι διεπαφές **ανήκουν στους κόμβους** και προσδιορίζονται από λογικά ονόματα, τα οποία δεν απαιτείται να ταυτίζονται με τα ονόματα που θα δημιουργηθούν τελικά στο Linux. Για παράδειγμα, μία διεπαφή με λογικό όνομα `lan` μπορεί κατά την υλοποίηση να αντιστοιχιστεί στη φυσική διεπαφή `rx0-eth0`. Η επιλογή αυτή επιτρέπει στο αρχείο JSON να χρησιμοποιεί ονόματα που εκφράζουν τον ρόλο μιας διεπαφής και όχι λεπτομέρειες της διαδικασίας δημιουργίας της.

Μία διεπαφή μπορεί να χρησιμοποιεί στατική διευθυνσιοδότηση, DHCP ή να μην διαθέτει καθόλου Layer-3 παραμετροποίηση. Στην περίπτωση στατικής διευθυνσιοδότησης δηλώνεται η διεύθυνση μαζί με το prefix και, προαιρετικά, μία προεπιλεγμένη πύλη. Επιπλέον μπορούν να δηλωθούν ιδιότητες όπως η διεύθυνση MAC και το MTU.

Ιδιαίτερη σημασία έχει η διάκριση μεταξύ της **απουσίας** του πεδίου `interfaces` και της δήλωσης ενός κενού αντικειμένου `interfaces`. Όταν το πεδίο παραλείπεται, ο κόμβος θεωρείται ότι διαθέτει *δυναμικό σύνολο διεπαφών* και νέες διεπαφές μπορούν να δημιουργηθούν κατά την επίλυση των συνδέσμων. Αντίθετα, όταν το πεδίο υπάρχει, ακόμη και αν είναι κενό, το σύνολο των διαθέσιμων διεπαφών θεωρείται *σταθερό*.

2.1.4. Σύνδεσμοι και αυτόματη επιλογή διεπαφών

Κάθε σύνδεσμος αποτελεί αυτοτελή οντότητα και συνδέει ακριβώς δύο endpoints. Κάθε endpoint προσδιορίζει τον κόμβο στον οποίο ανήκει και τη λογική διεπαφή μέσω της οποίας πραγματοποιείται η σύνδεση. Η διεπαφή μπορεί να δηλωθεί ρητά ή να χρησιμοποιηθεί ο ειδικός επιλογέας `auto`.

Όταν χρησιμοποιείται το `auto` σε κόμβο με σταθερό σύνολο διεπαφών, επιλέγεται μία από τις δηλωμένες διεπαφές που δεν έχει ήδη δεσμευτεί από άλλη σύνδεση. Αν δεν υπάρχει ελεύθερη διεπαφή, η τοπολογία θεωρείται μη έγκυρη και **δεν δημιουργείται επιπλέον θύρα σιωπηρά**. Όταν αντίθετα το σύνολο διεπαφών έχει παραλειφθεί, το `auto` επιτρέπει τη δυναμική δημιουργία νέας λογικής διεπαφής για τις ανάγκες του συνδέσμου.

Οι σύνδεσμοι μπορούν επίσης να περιλαμβάνουν παραμέτρους προσομοίωσης χαρακτηριστικών του φυσικού μέσου. Η τρέχουσα γλώσσα υποστηρίζει περιορισμό εύρους ζώνης, καθυστέρηση, απώλεια πακέτων και jitter μέσω των πεδίων `bandwidth_mbps`, `delay_ms`, `loss_percent` και `jitter_ms`.

2.1.5. Ρυθμίσεις κόμβων και υπηρεσίες

Οι ρυθμίσεις που αφορούν τη λειτουργία του ίδιου του κόμβου τοποθετούνται στο αντικείμενο `settings` του αντίστοιχου node. Για τον τύπο `router`, για παράδειγμα, υποστηρίζονται η ενεργοποίηση του IP forwarding και η απενεργοποίηση του reverse path filtering. Οι ιδιότητες αυτές αντιμετωπίζονται ως *χαρακτηριστικά του network namespace* και όχι ως ανεξάρτητες υπηρεσίες.

Οι υπηρεσίες μοντελοποιούνται ξεχωριστά επειδή αντιστοιχούν σε **διεργασίες που εκτελούνται πάνω σε έναν κόμβο**. Στην παρούσα εργασία υποστηρίζεται η υπηρεσία `dhcp-server`, η οποία συνδέεται με έναν κόμβο και μία από τις διεπαφές του. Οι παράμετροι της υπηρεσίας περιλαμβάνουν το εύρος διευθύνσεων που μπορούν να εκχωρηθούν και την προεπιλεγμένη πύλη που διαφημίζεται στους DHCP clients.

Η θέση εκτέλεσης μιας υπηρεσίας και η διεπαφή στην οποία συνδέεται αποτελούν μέρος της ίδιας της ταυτότητάς της και για τον λόγο αυτό δηλώνονται έξω από το αντικείμενο `settings`. Με αυτόν τον διαχωρισμό, το `settings` περιγράφει αποκλειστικά τη συμπεριφορά της υπηρεσίας και όχι το σημείο στο οποίο αυτή τοποθετείται μέσα στην τοπολογία.

2.1.6. Αυστηρότητα της γραμματικής

Η γραμματική σχεδιάστηκε ώστε να είναι **αυστηρή** και να μην αγνοεί άγνωστα πεδία ή μη υποστηριζόμενες τιμές. Ένα τυπογραφικό λάθος σε ένα όνομα ιδιότητας δεν αντιμετωπίζεται επομένως ως απουσία της ιδιότητας, αλλά ως σφάλμα της περιγραφής. Η επιλογή αυτή περιορίζει την πιθανότητα μία συντακτικά αποδεκτή αλλά διαφορετική από την επιθυμητή τοπολογία να δημιουργηθεί χωρίς εμφανή προειδοποίηση.

Η γλώσσα διαχωρίζει επίσης τη *δομική εγκυρότητα* μιας περιγραφής από τη *σημασιολογική συνέπειά* της. Ένα αντικείμενο μπορεί να ακολουθεί σωστά τη γραμματική αλλά να περιγράφει, για παράδειγμα, έναν σύνδεσμο προς ανύπαρκτο κόμβο ή μία στατική πύλη εκτός του αντίστοιχου υποδικτύου. Η διάκριση αυτή αξιοποιείται στη συνέχεια από την αρχιτεκτονική του builder, όπου η ανάλυση της γραμματικής και ο σημασιολογικός έλεγχος πραγματοποιούνται σε **διαφορετικά στάδια**.

2.2. Συμπεράσματα

...

3. ΣΥΝΟΨΗ

Τέλος, ...