ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

DYNAMIC RESOURCE ALLOCATION IN ROOFTOP NETWORKING

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Οικονομάκος Παναγιώτης

Εξεταστική Επιτροπή
Άγγελος Μπλέτσας, Αναπληρωτής Καθηγητής (Επιβλέπων)
Γεώργιος Καρυστινός, Αναπληρωτής Καθηγητής
Αντώνιος Δεληγιαννάκης, Αναπληρωτής Καθηγητής

Σεπτέμβρης 2014

Ευχαριστίες

Θα ήθελα να εκφράσω την ευγνωμοσύνη μου προς τον καθηγητή Άγγελο Μπλέτσα για την ανάθεση αυτής της διπλωματικής εργασίας, την καθοδήγησή του και την υποστήριξη από το αρχικό μέχρι το τελικό στάδιο της υλοποίησης. Η ευρεία γνώση και εμπειρία του στον τομέα των τηλεπικοινωνιών αποτέλεσαν σημαντική βοήθεια στην επίτευξη των στόχων μου. Επίσης θα ήθελα να ευχαριστήσω τον κ. Ντίλη Δημήτρη και τον κ. Παπαδάκη Βασίλη για την σημαντική συνεισφορά τους στην υλοποίηση αυτής της διπλωματικής εργασίας.

Τέλος θα ήθελα να ευχαριστήσω τα αγαπημένα μου πρόσωπα που πάντοτε ήταν δίπλα μου.

Περίληψη

Τα mesh networks είναι δίκτυα τα οποία επιτρέπουν σε ένα σύνολο από κόμβους, χωρίς συγκεκριμένη τοπολογία, να συνεργάζονται με σκοπό την αποτελεσματική δρομολόγηση πακέτων. Η δρομολόγηση των πακέτων γίνεται με άλματα από κόμβο σε κόμβο και μπορεί να αλλάζει, καθώς νέα βέλτιστα μονοπάτια υπολογίζονται. Οι κόμβοι μπορούν να είναι τοποθετημένοι σε κτήρια ή να είναι χρήστες που κινούνται. Στην πρώτη περίπτωση αναφερόμαστε στα rooftop networks. Είναι πολύ φτηνότερα από DSL ή καλωδιακά δίκτυα και μπορούν να συνεισφέρουν στην αποτελεσματικότερη παροχή υψηλής ταχύτητας δικτύου όταν δεν υπάρχει υποδομή για ενσύρματη πρόσβαση.

Για την υλοποίηση της διπλωματικής εργασίας χρησιμοποιήθηκαν Linux Boxes ALIX 2D2 και 3D2 καθώς και WiFi κάρτες συμβατές με οδηγούς λογισμικού (drivers) τύπου atheros. Εγκαταστήσαμε προσωρινά τέσσερις κόμβους στις ταράτσες των κτηρίων Α1, Α2, Κ4 και Μ3 της πολυτεχνειούπολης με δύο WiFi κάρτες στον κάθε κόμβο. Σε κάθε κόμβο η μία WiFi κάρτα χρησιμοποίησε μια omni directional κεραία ενώ η δεύτερη μία grid. Δημιουργήθηκε μηχανισμός που ελέγχει όλα τα συχνοτικά κανάλια σε κάθε ζεύξη, υπολογίζει έναν παράγοντα παρεμβολών (interference factor) ανά κανάλι και τέλος γίνεται μια αλλαγή στο “βέλτιστο” κανάλι (σύμφωνα με τον παραπάνω παράγοντα).

Το αποτέλεσμα της εργασίας ήταν η δημιουργία ενός mesh network με κόμβους τεχνολογίας WiFi, με δυνατότητα δυναμικής κατανομής συχνοτικών πόρων και δυνατότητα διαφορετικές ζεύξεις να έχουν διαφορετική συχνότητα μέσα στο ίδιο δίκτυο.

Περιεχόμενα

Figures

Κεφάλαιο 1

Εισαγωγή

1.1 Rooftop Networking

Τα rooftop networks είναι ασύρματα δίκτυα στα οποία οι κόμβοι είναι συνήθως τοποθετημένοι σε κτήρια. Κάθε κόμβος συμμετέχει στην δρομολόγηση των πακέτων καθώς τα πακέτα φτάνουν στον προορισμό τους με διαδοχικά άλματα από κόμβο σε κόμβο. Η δρομολόγηση των πακέτων μπορεί να αλλάζει ανά τακτά χρονικά διαστήματα, επιλέγοντας κάθε φορά τα βέλτιστα μονοπάτια. Παρέχουν την δυνατότητα δημιουργίας μεγάλων δικτύων με πολλούς κόμβους εύκολα λόγω απουσίας καλωδίων και με χαμηλό κόστος αφού ο απαιτούμενος εξοπλισμός είναι σχετικά φτηνός (WiFi cards, Linux boxes, antennas). Η σύνδεση ενός τέτοιου δικτύου με το Internet είναι πολύ εύκολη καθώς αρκεί μόνο λίγοι από τους κόμβους του δικτύου να έχουν πρόσβαση. Τα πακέτα δρομολογούνται σε αυτούς τους κόμβους (gateways) με διαδοχικά άλματα και έτσι όλοι οι κόμβοι του δικτύου έχουν πρόσβαση στο Internet.

Λόγω του ότι οι κόμβοι δεν έχουν σταθερές συνδέσεις μεταξύ τους καθώς η δρομολόγηση των πακέτων αλλάζει ανάλογα με το πιο μονοπάτι είναι βέλτιστο για μια συγκεκριμένη χρονική στιγμή, τα δεδομένα δεν ακολουθούν συγκεκριμένη διαδρομή για να φτάσουν στα gateways. Αυτό γίνεται ακόμα πιο αισθητό όσο περισσότεροι κόμβοι είναι στο δίκτυο. Έτσι προκύπτει το ερώτημα του πως να δρομολογηθούν τα δεδομένα έτσι ώστε τελικά να βελτιστοποιηθεί η ροή των πακέτων προς τα gateways.

Ένα σημαντικό project που προσπαθεί να βρει λύση σε αυτό το ερώτημα είναι το Roofnet στο οποίο δουλεύει μια ερευνητική ομάδα στο MIT. Το Roofnet είναι ένα project για την δημιουργία ενός mesh network που αποτελείται από ανεξάρτητα φτηνά Linux PCs εξοπλισμένα με WiFi κάρτες που να είναι σε θέση να συνεργάζονται αποτελεσματικά με σκοπό την δρομολόγηση πακέτων.

Παρόμοια συστήματα αναπτύσσονται και σε πανεπιστήμια όπως το Carnegie Mellon, Rice, UCLA και το πανεπιστήμιο του Illinois, καθώς και από εταιρείες όπως η Nokia, Intel και Microsoft. Σε κάθε ένα από τα παραπάνω συστήματα, πακέτα δρομολογούνται ασύρματα μέσω κόμβων που είναι είτε τοποθετημένοι σε κτήρια είτε είναι κινούμενοι χρήστες ή οχήματα. Για παράδειγμα, τέτοια multi-hop mesh δίκτυα περιλαμβάνουν συστήματα για δικτύωση χρηστών με PDAs, άρματα σε ένα πεδίο μάχης ή αισθητήρες σε ένα εργοστάσιο.

Οι περισσότεροι αλγόριθμοι δρομολόγησης που προτείνονται, ακολουθούν την στρατηγική του shortest-path. Αυτά τα πρωτόκολλα προσπαθούν να βρουν διαδρομές με τα λιγότερα άλματα μεταξύ ενδιάμεσων κόμβων από τον αποστολέα στον παραλήπτη. Αυτή η στρατηγική είναι αρκετά αποτελεσματική στις περιπτώσεις των ενσύρματων δικτύων. Στην περίπτωση όμως των ασύρματων δικτύων υπάρχει ένας παράγοντας που έχει πού μεγαλύτερη επίπτωση σε σχέση με τα ενσύρματα, η απόσταση μεταξύ των κόμβων. Όσο μεγαλύτερη απόσταση έχει να διανύσει ένα σήμα, τόσο περισσότερο μειώνεται η ισχύς του. Επίσης, ένας άλλος σημαντικός παράγοντας είναι το ότι η ποιότητα του link μεταξύ δύο κόμβων αλλάζει απρόβλεπτα και γρήγορα από αιτίες όπως υγρασία η κινούμενα εμπόδια.

Το αποτέλεσμα είναι μεγάλος αριθμός από πακέτα να χάνονται, να γίνονται λάθη στην μετάδοση και συνδέσεις που την μία στιγμή να είναι εφικτές και την επόμενη να μην είναι. Έτσι ως συμπέρασμα, ένας αλγόριθμος που ακολουθεί την στρατηγική του shortest-path σε ένα ασύρματο δίκτυο, μπορεί να επιλέξει links με μεγαλύτερη απόσταση μεταξύ διαδοχικών αλμάτων και έτσι τελικά η διαδρομή να μην είναι η βέλτιστη.

Μια διαφορετική προσέγγιση της ερευνητικής ομάδας στο MIT για το Roofnet project είναι οι αλγόριθμοι που προτείνονται από την Internet Engineering Task Force, μια οργάνωση που είναι υπεύθυνη για τα τεχνικά χαρακτηριστικά του Internet. Αυτά τα πρωτόκολλα λειτουργούν αρκετά καλά στην θεωρία και έχουν δοκιμαστεί σε εξομοιώσεις ή σε μικρής κλίμακας δίκτυα σε περιβάλλον εργαστηρίου. Σε πραγματικές συνθήκες όμως αντιμετωπίζουν κάποια προβλήματα τα οποία αρχικά δεν είχαν ληφθεί υπόψη. Ένα τέτοιο πρόβλημα είναι η υπόθεση του ότι αν δύο γειτονικοί κόμβοι μπορούν να εντοπίσουν ο ένας τον άλλον, αυτό σημαίνει ότι είναι σε θέση να ανταλλάξουν και δεδομένα. Αυτό δεν συμβαίνει πάντα καθώς ενώ μερικές φορές μπορούν να εντοπίσουν τα μεταξύ τους πακέτα ανακάλυψης, όταν αρχίσουν να ανταλλάσσουν δεδομένα η επικοινωνία είναι ανεπιτυχής λόγω ανεπαρκούς bandwidth.

Παρακάτω παρουσιάζονται μερικά τεχνικά χαρακτηριστικά του Roofnet project. [1, 2]

Το Roofnet είναι ένα πειραματικό 802.11b/g mesh network που αναπτύσσεται από το Computer Science and Artificial Intelligence Laboratory στο MIT. Το πρωτόκολλο δρομολόγησης που χρησιμοποιείται σε αυτό το δίκτυο είναι το SrcRR.

Το SrcRR πρωτόκολλο είναι μια επέκταση του Dynamic Source Routing protocol. Αφορά την βελτιστοποίηση του throughput στο δίκτυο λαμβάνοντας υπόψη την ποιότητα του σήματος μεταξύ των κόμβων, την ταχύτητα αποστολής δεδομένων και την τυχών εκρηκτική κίνηση δεδομένων. Είναι ανεξάρτητο από IPs καθώς είναι layer 2 routing protocol. Στις περιπτώσεις όπου χρησιμοποιούνται IPs, το SrcRR χρησιμοποιεί 32 bit διευθύνσεις όπου βρίσκονται στους headers. Κάθε κόμβος έχει έναν πίνακα στον οποίο αντιστοιχίζονται όλες οι 32 bit IPs με 48 bit 802.11 MAC addresses, οι οποίες είναι γίνονται γνωστές από τα broadcasts.

Χρησιμοποιείται μια μετρική, το ETX το οποίο είναι ο προβλεπόμενος αριθμός μεταδόσεων από ένα κόμβο σε έναν άλλον, έως να μεταδοθούν τα δεδομένα. Κάνει συνεχείς μετρήσεις για το loss rate μεταξύ κάθε κόμβου και των γειτονικών του και στις δύο κατευθύνσεις κάνοντας broadcasts ανά τακτά χρονικά διαστήματα. Τα broadcast πακέτα στο SrcRR πρωτόκολλο είναι 300 byte και μεταδίδονται σε τυχαία χρονικά διαστήματα με μέσο όρο τα 10 δευτερόλεπτα. Ένας κόμβος υπολογίζει το loss rate κάθε γείτονα βρίσκοντας το ποσοστό των broadcast πακέτων που έχουν ληφθεί τα τελευταία 3 λεπτά (18 broadcast πακέτα). Αναθέτει σε κάθε link μια μετρική που υπολογίζει πόσες φορές ένα πακέτο θα πρέπει να μεταδοθεί ώστε να ληφθεί επιτυχώς, συμπεριλαμβανομένων και τον acknowledgment πακέτων του 802.11. Η ETX μετρική είναι το άθροισμα των παραπάνω link μετρικών. Έτσι για την επιλογή του βέλτιστου link συνυπολογίζεται και ο παράγοντας της απόστασης μεταξύ των κόμβων αλλά και το loss rate ενός link και στις δύο κατευθύνσεις.

Όλοι οι SrcRR κόμβοι έχουν αποθηκευμένο ένα πίνακα για κάθε ETX που αντιστοιχεί σε κάθε link, ο οποίος περιλαμβάνει τις ETX μετρικές που έχει ακούσει ο κάθε κόμβος πρόσφατα. Αν μία ETX μετρική δεν ανανεωθεί για 30 δευτερόλεπτα τότε απορρίπτεται, έτσι ώστε μόνο οι νέες μετρικές να χρησιμοποιούνται για την αναζήτηση βέλτιστης δρομολόγησης. Κάθε φορά που μια ETX μετρική αλλάζει τιμή, ο κόμβος θα τρέξει τον Dijkstra's weighted shortest-path αλγόριθμο με σκοπό να βρει τις νέες διαδρομές προς όλους τους υπόλοιπους κόμβους που έχουν τις ελάχιστες ETX μετρικές. Όλα τα πακέτα που μεταδίδονται στο δίκτυο συμπεριλαμβάνουν και τις ETX τιμές των links από τα οποία πέρασαν, έτσι κάθε κόμβος που λαμβάνει ένα πακέτο μπορεί να ανανεώσει αυτές τις ETX μετρικές που έχει αποθηκευμένες.

Όταν ένας κόμβος θέλει να στείλει δεδομένα σε έναν κόμβο για τον οποίο δεν έχει αποθηκευμένη κάποια διαδρομή, τότε κάνει flood ένα route request. Όταν ένας κόμβος λάβει ένα route request, προσθέτει το ID του και την ETX μετρική που αφορά το link από το οποίο ήρθε το route request και το κάνει broadcast. Όταν ένας κόμβος λάβει για πρώτη φορά ένα route request το κάνει broadcast πάντα. Αν όμως λάβει ένα route request για περισσότερες από μία φορές, θα το κάνει broadcast μόνο στην περίπτωση όπου το άθροισμα των συμπεριλαμβανομένων ETX μετρικών είναι καλύτερο από τις προηγούμενες φορές. Αυτό εγγυάται το ότι θα επιλεγεί η βέλτιστη διαδρομή προς τον προορισμό των route request.

Όταν ένας κόμβος λάβει ένα route request το οποίο προορίζεται για αυτόν τότε αποθηκεύει την διαδρομή από την οποία πέρασε αυτό το route request και την χρησιμοποιεί για να δρομολογήσει ένα route reply. Όταν ο κόμβος από τον οποίο προέρχεται το route request λάβει το route reply, αποθηκεύει την διαδρομή μαζί με το άθροισμα των ETX μετρικών για τα links που περιλαμβάνει. Η διαδρομή με το μικρότερο άθροισμα των ETX μετρικών θα είναι και η διαδρομή που θα επιλεγεί για να μεταδοθούν τα δεδομένα. Κάθε ενδιάμεσος κόμβος που προωθεί ένα πακέτο δεδομένων, ανανεώνει την ETX μετρική του που αφορά το link από το οποίο δέχτηκε αυτό το πακέτο δεδομένων. Αυτό επιτρέπει στον αρχικό κόμβο που στέλνει τα δεδομένα να έχει συγχρονισμένους πίνακες με τις ETX μετρικές με τον τελικό κόμβο και να αντιλαμβάνεται αν μια διαδρομή πλέον δεν είναι η βέλτιστη για δρομολόγηση. Επιπλέον κάθε πακέτο δεδομένων έχει ένα πεδίο για μία επιπλέον ETX μετρική. Κάθε κόμβος που προωθεί ένα πακέτο δεδομένων θα ανανεώσει αυτό το πεδίο με πιθανότητα 1/n, όπου n είναι ο αριθμός των κόμβων στην διαδρομή, με μια ETX τιμή για έναν από τους γειτονικούς του κόμβους, ο οποίος θα επιλεγεί τυχαία. Αυτό επιτρέπει στον αρχικό και στον τελικό κόμβο να μάθουν για links με καλύτερες ETX μετρικές. Αν προκύψει ETX μετρική που έχει αλλάξει, αυτό θα προκαλέσει τον επαναϋπολογισμό των βέλτιστον διαδρομών του κόμβου που αφορά η αλλαγή της ETX μετρικής προς όλους τους άλλους κόμβους, χρησιμοποιώντας όπως έχει ήδη αναφερθεί τον αλγόριθμο Dijkstra.

Το SrcRR πρωτόκολλο αντιμετώπιζε κάποια προβλήματα στην αρχική του μορφή και έτσι έχει δεχθεί μια σειρά βελτιώσεων μερικές από τις οποίες αναφέρονται παρακάτω.

Η πρώτη βελτίωση του SrcRR είναι ένας μηχανισμός ο οποίος αναφέρεται ως INGORE-FAIL, ο οποίος καθιστά το πρωτόκολλο ανθεκτικό σε τυχών προσωρινή απώλεια σύνδεσης ή μεγάλου packet loss. Χωρίς αυτό το μηχανισμό, με την πρώτη αποτυχημένη μετάδοση (δεν έχει ληφθεί πακέτο acknowledgement για 8 δευτερόλεπτα), ειδοποιείται ο αρχικός κόμβος ο οποίος θα αλλάξει διαδρομή. Με αυτό το μηχανισμό μετά την αποτυχημένη μετάδοση δεν θα συμβεί κάτι, ο αρχικός ο κόμβος θα συνεχίσει να δρομολογεί τα πακέτα του από αυτή την διαδρομή έως ότου να προκύψει καλύτερη διαδρομή με βάση την ETX μετρική. Έτσι δίνεται μεγαλύτερο βάρος στην μακροχρόνιες μετρήσεις της ETX μετρικής και όχι σε τυχών προσωρινή μείωση της ποιότητας ενός link.

Η επόμενη βελτίωση που ακολούθησε αφορούσε το bit-rate και αναφέρεται ως RATE-CTL. Παρακάμπτετε το bit-rate του 802.11 και αντί για αυτό κάθε SrcRR κόμβος αποφασίζει το βέλτιστο bit-rate για κάθε γείτονα. Γίνονται broadcast πακέτα σε κάθε δυνατό bit-rate και καταγράφεται το ποσοστό των πακέτων που φτάνουν επιτυχώς. Υπολογίζεται το throughput για κάθε bit-rate πολλαπλασιάζοντας το ποσοστό των επιτυχημένων μεταδόσεων των broadcast πακέτων με το bit-rate και τελικά επιλέγεται το bit-rate με το μεγαλύτερο throughput. Είναι επέκταση του ETX μηχανισμού που όπως έχει αναφερθεί, κάνει broadcast πακέτα σε τυχαία χρονικά διαστήματα με μέσο όρο τα 10 δευτερόλεπτα.

Όταν υπάρξει μια αποτυχία στην μετάδοση ενός πακέτου από έναν κόμβο, ο μηχανισμός PERSISTENCE του SrcRR θα το τοποθετήσει στην αρχή της ουράς των πακέτων προς μετάδοση του συγκεκριμένου κόμβου αντί να το απορρίψει και έτσι ο κόμβος θα επιχειρήσει να στείλει ξανά το πακέτο. Αυτή η επανάληψη μπορεί να γίνει για έναν συγκεκριμένο αριθμό προσπαθειών που αρχικά είχε οριστεί 40 προσπάθειες. Αν ακόμα η αποστολή του πακέτου δεν είναι δυνατή τότε το SrcRR στέλνει ένα Route Error μήνυμα στον αρχικό κόμβο ασχέτως αν ο μηχανισμός IGNORE-FAIL που αναφέρθηκε πριν είναι ενεργοποιημένος. Λόγω του ότι οι ουρές των προς μετάδοση πακέτων κάθε κόμβου είναι μικρές, οι επαναλαμβανόμενες προσπάθειες αποστολής ορισμένων πακέτων συνδυάζονται με τα πακέτα της ουράς. Αυτό γίνεται για να αποφευχθεί το φαινόμενο του να εμποδίζονται να σταλούν πακέτα που βρίσκονται στην ουρά. Αυτό όμως δημιούργησε το πρόβλημα της ανακατάταξη της σειράς αποστολής των πακέτων. Αυτό το πρόβλημα λύθηκε με τον μηχανισμό REORDER ο οποίος θα αναλυθεί αργότερα.

Ο μηχανισμός DAMPING χρησιμεύει στο να αποτρέπεται η συνεχής αλλαγή απόφασης βέλτιστης διαδρομής στην περίπτωση όπου δύο διαδρομές έχουν τις ίδιες μετρικές. Όταν αποφασιστεί μια διαδρομή για την δρομολόγηση πακέτων δεδομένων τότε η επόμενη αλλαγή διαδρομής θα μπορεί να γίνει μετά από 5 δευτερόλεπτα ή αν βρεθεί άλλη διαδρομή με καλύτερη ETX μετρική.

Ο μηχανισμός REORDER, επανατοποθετεί πακέτα εκτός σειράς στην σωστή τους θέση. Ο αρχικός κόμβος έχει έναν αριθμό για κάθε ροή πακέτων και κάθε πακέτο χαρακτηρίζεται από έναν αριθμό ανάλογα με την σειρά αποστολής του στην ροή που ανήκει. Κάθε ενδιάμεσος κόμβος τοποθετεί τα πακέτα με τέτοιον τρόπο ώστε τα πακέτα κάθε ροής να είναι κατά αύξοντα αριθμό. Στον προορισμό τους τα πακέτα τοποθετούνται στην σωστή σειρά πριν παραδοθούν στο επόμενο layer. Για να γίνει αυτό, τα πακέτα που είναι εκτός σειράς κρατούνται σε μια ουρά μέχρι να έρθει η σειρά τους να παραδοθούν. Αν ένα πακέτο που είναι εκτός σειράς παραμείνει σε αυτή την ουρά για περισσότερο από 500 milliseconds απορρίπτεται. Στην περίπτωση που έχει χαθεί ένα πακέτο από overflow σε μία ουρά και δεν είναι απλώς εκτός σειράς θα περιμένουμε άσκοπα αυτό το χρονικό διάστημα. Για να αποφύγουμε την αναμονή των 500 milliseconds σε αυτή την περίπτωση, κάθε κόμβος θυμάται πότε απέρριψε ένα πακέτο μιας συγκεκριμένης ροής λόγω queue overflow και χαρακτηρίζει το επόμενο πακέτο της συγκεκριμένης ροής με ένα congestion bit. Έτσι στον προορισμό θα απορριφθούν όλα τα επόμενα πακέτα αυτής της ροής που έχουν φτάσει ήδη και έχουν χαρακτηριστεί ως εκτός σειράς, επειδή περίμεναν να παραδοθεί το πακέτο που ήδη είχε χαθεί. Επίσης κάθε κόμβος εντοπίζει και απορρίπτει πακέτα που υπάρχουν σε μία ροή περισσότερες από μία φορές, κρατώντας τον αριθμό σειράς των τελευταίων 100 πακέτων κάθε ροής.

Ο μηχανισμός BIG-PROBES προσπαθεί να λύσει το πρόβλημα του λάθος υπολογισμού των loss rates σε διαφορετικές διαδρομές. Τα πακέτα δεδομένων είναι 1500-bytes ενώ τα broadcast πακέτα που χρησιμοποιούνται για τον υπολογισμό των ETX μετρικών είναι 300-bytes. Στις περιπτώσεις όπου οι διαδρομές έχουν τον ίδιο αριθμό ενδιάμεσων κόμβων αυτό δεν έχει μεγάλη επίπτωση καθώς το σφάλμα στον υπολογισμό των loss rates είναι ίδιο και για τις δύο διαδρομές. Αυτό όμως δεν ισχύει στις περιπτώσεις που δεν έχουμε ίδιο αριθμό ενδιάμεσων αλμάτων στις υπό σύγκριση διαδρομές. Η λύση είναι να χρησιμοποιούμε broadcast πακέτα των 1500-bytes αντί για 300-bytes. Σε μερικές περιπτώσεις θα μπορούσαν να χρησιμοποιηθούν τα πακέτα δεδομένων απευθείας για τον υπολογισμό των loss rates. Αυτό όμως δεν καθιστά τα broadcast πακέτα του ETX μηχανισμού άχρηστα, καθώς πρέπει να υπολογιστούν τα loss rates και των links που δεν χρησιμοποιούνται την συγκεκριμένη χρονική στιγμή.

Τέλος, για να συνυπολογιστεί και ο παράγοντας του bit-rate, μια νέα μετρική άρχισε να χρησιμοποιείται. Η ETT αντικαθιστά την ETX και είναι ο χρόνος που θα χρειαστεί ένα πακέτο να κρατήσει το μέσο (αέρας μιας και αναφερόμαστε σε ασύρματα δίκτυα) απασχολημένο έως ότου να μεταδοθεί. Ελαχιστοποιώντας αυτή την μετρική, μεγιστοποιείται το throughput για κάθε link. Αθροίζοντας τις ETT μετρικές κάθε διαδρομής και επιλέγοντας την ελάχιστη μπορούμε να βρούμε την βέλτιστη διαδρομή. Όπως έχει αναφερθεί και στον RATE-CTL μηχανισμό, κάθε κόμβος στέλνει broadcast πακέτα σε όλα τα bit-rates και βρίσκει το βέλτιστο bit-rate υπολογίζοντας το throughput που κάθε bit-rate επιτυγχάνει. Όταν το ETT είναι ενεργοποιημένο, κάθε κόμβος πολλαπλασιάζει το προβλεπόμενο throughput προς κάθε γείτονά του με την πιθανότητα επιτυχούς παράδοσης ενός acknowledgment πακέτου προς την αντίθετη κατεύθυνση.

$$ETT = \frac{1}{P(ack) * \mathrm{r}}$$

Όπου
P(ack): Πιθανότητα παράδοσης ενός ACK πακέτου από τα broadcast πακέτα προς την αντίθετη κατεύθυνση.
r: Προβλεπόμενο throughput για το συγκεκριμένο link έχοντας επιλέξει το βέλτιστο bit-rate

Η πιθανότητα της επιτυχής παράδοσης ACK πακέτων προκύπτει από το broadcast πακέτων μεγέθους 32 bytes. Τελικά, όταν χρησιμοποιείται το ETT, κάθε κόμβος κάνει broadcast πέντε πακέτα κάθε δέκα δευτερόλεπτα, ένα μικρό στο 1 megabit και ένα των 1500-bytes σε κάθε ένα από τα παρακάτω bit-rates: 1, 2, 5.5 και 11 megabits. Ο μηχανισμός ETT στέλνει κάθε ένα από αυτά τα πέντε broadcast πακέτα σε ανεξάρτητα τυχαία χρονικά διαστήματα με μέσο όρο τα 10 δευτερόλεπτα.

Η ιδέα για την αυτοματοποίηση της διαδικασίας δρομολόγησης σε ένα mesh network είχε πρωτοεμφανισθεί την δεκαετία του 1960 από τον Paul Baran [3]. Ο Baran είχε την ιδέα για ένα δίκτυο με point-to-point microwave links. Αρχικά οι ιδέες του βρήκαν εφαρμογή στο NPL network και στο ARPANet στα τέλη της δεκαετίας του 1960, και ήταν η βάση για τους αλγορίθμους δρομολόγησης που χρησιμοποιούνται σήμερα στο ενσύρματο Internet. Στις δεκαετίες του 1970 και 1980 το project PRNet [4] της DARPA έφερε μερικές καινοτόμες ιδέες στην δικτύωση ασύρματων Mesh networks.

Το Roofnet χρησιμοποιεί πρωτόκολλα δρομολόγησης που έχουν προκύψει από παλαιότερα ερευνητικά project πάνω σε mobile ad-hoc networks (MANET) [5]. Όπως έχει ήδη αναφερθεί, το πρωτόκολλο SrcRR που χρησιμοποιείται τώρα στο Roofnet, έχει προκύψει από το DSR και επίσης γίνονται δοκιμές και με άλλα πρωτόκολλα που έχουν βάση το DSDV [6]. Ο στόχος των παραπάνω πρωτοκόλλων είναι να διαχειρίζονται κινούμενους χρήστες με μεγάλη κινητικότητα μέσα στο δίκτυο. Στο Roofnet λόγω του ότι οι κόμβοι είναι τοποθετημένοι σε ταράτσες κτηρίων και έτσι δεν υπάρχει καμία κινητικότητα στους χρήστες, ο στόχος είναι να βρίσκονται βέλτιστες διαδρομές δρομολόγησης με βάση την ποιότητα των links. Υπάρχουν διάφορα wireless test-beds για την δοκιμή τέτοιων MANET πρωτοκόλλων σε πραγματικές συνθήκες [7, 8, 9, 10, 11].

Τα δίκτυα αισθητήρων χρησιμοποιούν αρκετά συχνά ασύρματα multi-hop networks για την συλλογή δεδομένων και έτσι αντιμετωπίζονται προβλήματα παρόμοια με αυτά του Roofnet. Για παράδειγμα ο Yarvis [12] παρατήρησε ότι η μετρική του hop-count δεν είναι τόσο αποτελεσματική για ένα δίκτυο αισθητήρων και πρότεινε να χρησιμοποιηθεί μια loss-aware μετρική. Επίσης ο Ganesan [13] περιγράφει μερικές παρατηρήσεις του σε ασύρματα δίκτυα αισθητήρων, συμπεριλαμβανομένων και μερικών φαινομένων που είχαν παρατηρηθεί και στο Roofnet.

Κεφάλαιο 2

IEEE 802.11

2.1 Γενική περιγραφή

Το 802.11 είναι ένα standard το οποίο χρησιμοποιείται για την ασύρματη σύνδεση κόμβων σε ένα δίκτυο. Συχνά αναφέρεται ως WiFi και αποτελεί την πιο δημοφιλή λύση για την δικτύωση συσκευών σήμερα. Λόγω της ευελιξίας που προσφέρει και της απόδοσης, τα WiFi hotspots όπως αναφέρονται οι περιοχές που ένας χρήστης μπορεί να έχει πρόσβαση σε ένα τέτοιο δίκτυο, συναντώνται αρκετά συχνά σε ξενοδοχεία, καφετέριες, αεροδρόμια κλπ. Ένα 802.11 δίκτυο βασίζεται στην αρχιτεκτονική των κυψελωτών δικτύων όπου το δίκτυο χωρίζεται σε κελιά. Τα κελιά (Basic Service Set ή BSS) διαχειρίζονται από ένα Base Station (Access Point ή AP). Ένα ασύρματο δίκτυο μπορεί να περιλαμβάνει μόνο ένα κελί με μόνο ένα AP, αλλά στις περισσότερες περιπτώσεις αποτελείται από πολλά κελιά όπου τα AP των κελιών του δικτύου συνδέονται μεταξύ τους μέσω ενός backbone δικτύου (το οποίο αναφέρεται ως Distribution System ή DS), όπως Ethernet. Ένα τέτοιο 802.11 δίκτυο με πολλά κελιά αναφέρεται ως Extended Service Set (ESS).

Το 802.11 λειτουργεί στην ISM (Industrial, Scientific and Medical) ζώνη συχνοτήτων και έτσι δεν χρειάζεται κάποια ειδική άδεια. Η πρώτη έκδοση ήταν το 1997, αργότερα προέκυψαν και άλλες εκδόσεις, κάθε μία έχει τα δικά της χαρακτηριστικά. Οι σημαντικότερες παρουσιάζονται στο επόμενο figure :

	802.11a	**802.11b**	**802.11g**	**802.11n**
Date of standard approval	July 1999	July 1999	June 2003	October 2009
Maximum data rate (Mbps)	54	11	54	~600
Modulation	OFDM	CCK or DSSS	CCK,DSSS or OFDM	CCK,DSSS or OFDM
RF Band	5	2.4	2.4	2.4 or 5
Number of spatial streams	1	1	1	1, 2, 3 or 4
Channel width (MHz)	20	20	20	20 or 40

Figure 1: **Summary of major 802.11 Wi-Fi Standards**

Υπάρχουν δύο είδη ασύρματων δικτύων που προσφέρει το 802.11. Το πρώτο αναφέρεται ως hotspot και στόχος του είναι να καλύψει μία περιοχή προσφέροντας πρόσβαση στο δίκτυο σε έναν αριθμό χρηστών. Είναι πολύ πιο εύκολο να εγκατασταθεί από ένα αντίστοιχο ενσύρματο και πιο φτηνό λόγω της έλλειψης καλωδίων. Ένα ενσύρματο backbone δίκτυο είναι απαραίτητο για την σύνδεση του δικτύου με κάποιον server.

Το δίκτυο μπορεί να χωριστεί σε κελιά και κάθε κελί εξυπηρετείται από ένα σταθμό βάσης (Access Point – AP). Η εμβέλεια κάθε κελιού μπορεί να είναι μεταξύ 30 και 300 μέτρων και εξαρτάται από το περιβάλλον και την θέση του Access Point.

Το δεύτερο είδος είναι τα λεγόμενα Ad-Hoc δίκτυα. Αυτά αποτελούνται από συσκευές που συνδέονται μεταξύ τους απευθείας. Για παράδειγμα ένα Ad-Hoc δίκτυο είναι οι χρήστες που συνδέουν τους υπολογιστές τους με έναν ασύρματο εκτυπωτή. Δεν υπάρχει κάποιο Access Point και έτσι τον ρόλο του διαχειριστή του δικτύου τον αναλαμβάνουν οι περιφερειακές συσκευές στις οποίες συνδέονται οι χρήστες, χρησιμοποιώντας ειδικούς αλγόριθμους και πρωτόκολλα. Έτσι δημιουργείται μια σχέση Master – Slaves μεταξύ της κεντρικής συσκευής και των χρηστών που συνδέονται σε αυτή.

2.2 802.11 Physical Layer

Η πρώτη έκδοση του 802.11 ήταν το 1997 και καθιέρωσε τρεις τρόπους για μετάδοση δεδομένων, Infrared στο 1 Mbps, Frequency hopping spread spectrum (FHSS) στα 1 και 2 Mbps και Direct sequence spread spectrum (DSSS) στα 1 και 2 Mbps. Οι τελευταίες δύο τεχνικές χρησιμοποιούν την ISM ζώνη συχνοτήτων στα 2.4 GHz. Η Infrared τεχνολογία δεν είχε εμπορικές εφαρμογές. Δεν χρησιμοποιείται πλέον αλλά αποτέλεσε βάση για μοντέλα που εκδόθηκαν αργότερα. Αυτά που θα μας απασχολήσουν στα πλαίσια αυτής της διπλωματικής είναι τα 802.11b και 802.11s. [15]

2.2.1 Frequency Hopping Spread Spectrum

Το FHSS είναι μια μέθοδος για την αποστολή σημάτων με την γρήγορη εναλλαγή της κεντρικής συχνότητας του carrier, χρησιμοποιώντας μια ψευδοτυχαία ακολουθία που είναι γνωστή στον πομπό και στον δέκτη. Αποτελεί το πρώτο βήμα για την ανάπτυξη του Direct Sequence Spread Spectrum και για άλλες νεότερες τεχνικές. Η ζώνη των 2.4 GHz χωρίζεται σε κανάλια που απέχουν 1 MHz. Χρησιμοποιεί two-level GFSK διαμόρφωση για bitrate 1 Mbps και four-level GFSK για bitrate 2 Mbps. Χρησιμοποιεί 78 ακολουθίες για την εναλλαγή των συχνοτήτων. Κάθε εναλλαγή πρέπει να απέχει από την προηγούμενη 400ms και 6MHz. Είναι μια τεχνική με ανθεκτικότητα σε narrow band παρεμβολές αλλά δεν χρησιμοποιείται πλέον καθώς υπάρχουν μεγάλες απαιτήσεις σε bandwidth όσο η ταχύτητα μετάδοσης αυξάνεται.

2.2.2 Direct Sequence Spread Spectrum

Το DSSS είναι μια τεχνική διαμόρφωσης όπου προέρχεται από το FHSS. Είναι μια από τις πιο επιτυχημένες τεχνικές ως σήμερα και χρησιμοποιείται σε κυψελωτά δίκτυα, GPS και WLANs. Το σήμα πολλαπλασιάζεται με μια ψευδοτυχαία ακολουθία (PN) που αναφέρεται ως chip. Αυτή η ακολουθία αποτελείται από μια σειρά 1, -1 και είναι πολύ μεγαλύτερης συχνότητας από το σήμα της πληροφορίας που θέλουμε να μεταδώσουμε. Ο δέκτης γνωρίζει αυτή την ακολουθία και έτσι είναι σε θέση να ανακτήσει την πληροφορία. Η ακολουθία που χρησιμοποιείται είναι 11 bit Barker Code. Η ταχύτητα αποστολής είναι τα 11Mbps και απαιτούνται 22 MHz bandwidth.

Το σήμα μετά τον πολλαπλασιασμό με το chip, απλώνεται στις συχνότητες και μοιάζει ως θόρυβος. Δεν επηρεάζεται από τυχών narrow band noise καθώς η επίδραση του chip και του correlator στον δέκτη έχει σαν αποτέλεσμα να εξαπλώνεται το interference στις συχνότητες και η ισχύς του να είναι πολύ χαμηλότερη από την ισχύ του σήματος.

Παρακάτω είναι η μορφή του πακέτου στην DSSS διαμόρφωση.

PLCP Preamble		PLCP Header				
Synchronization 128 bits	SFD 16 bit	Signal 8 bits	Service 8 bits	Length 16 bits	HEC 16 bits	Payload Variable

Figure 2: **IEEE 802.11 PHY Frame Using DSSS**

PLCP Preamble:

Synchronization (128 bits): Χρησιμεύει στον συγχρονισμό του δέκτη.

SFD (16 bit): Start Frame Delimiter, ορίζει την αρχή κάθε frame.

PLCP Header:

Signal (8 bits): Καθορίζει την ταχύτητα που θα μεταδοθεί το payload, 00001010 για 1 Mbps και 00010100 για 2 Mbps.

Service (8 bits): Αυτό το πεδίο έχει δημιουργηθεί για τυχών μελλοντική χρήση.

Length (16 bits): Καθορίζει το μέγεθος του frame

HEC (16 bits): Header Error Control, χρησιμεύει για τον εντοπισμό λαθών στο frame.

Τα πεδία Preamble και Header μεταδίδονται πάντα στο 1 Mbps, το payload θα μεταδοθεί στο bitrate που καθορίζει το πεδίο signal. Χρησιμοποιείται DBPSK για το 1 Mbps και DQPSK για τα 2 Mbps.

2.2.3 802.11b/s

Το 802.11b είναι μια εξέλιξη του 802.11 και εκδόθηκε το 1999. Βελτιώνει το μέγιστο bitrate στα 11 Mbps και εισάγει την χρήση μιας νέας διαμόρφωσης, την Complementary Code Keying (CCK). Στην δομή του πακέτου δίνεται η δυνατότητα του να χρησιμοποιείται ένα PLCP header με ένα μικρό preamble των 56 bits. Σε αυτό το mode μόνο το πεδίο του Synchronization και του Start Frame Delimiter μεταδίδονται στα 1 Mbps. Το υπόλοιπο κομμάτι του header μεταδίδεται στα 2 Mbps χρησιμοποιώντας DQPSK. Το payload μεταδίδεται είτε στα 2 Mbps (DQPSK) είτε στα 5.5 ή 11 Mbps με CCK. Στο παρακάτω figure φαίνεται η δομή του πακέτου στο 802.11b. [16]

1 Mbps DSSS DBPSK						1 Mbps DSSS DBPSK 2Mbps DSSS DQPSK 5.5/11Mbps CCK DQPSK
Sync 128 scrambled 1s	SFD 16 bits	Signal 8 bits	Service 8 bits	Length 16 bits	CRC 16 bits	PSDU Variable

Figure 3: **IEEE 802.11b PHY Long Frame**

1 Mbps DBPSK		2 Mbps DQPSK				2 Mbps DSSS DQPSK 5.5/11Mbps CCK DQPSK
Sync 56 Scrambled 0s	SFD 16 bits	Signal 8 bits	Service 8 bits	Length 16 bits	CRC 16 bits	PSDU Variable

Figure 4: **IEEE 802.11b PHY Short Frame**

Το CCK είναι μία διαμόρφωση που χρησιμοποιείται στο 802.11b και αντικαθιστά τους Barker Codes με σκοπό να αυξηθεί η ταχύτητα αποστολής με το αντάλλαγμα της μικρότερης εμβέλειας. Είναι complex codes, βασίζονται στους polyphase complementary codes και παρουσιάζουν χαρακτηριστικά ορθογωνιότητας μεταξύ τους.

Είναι λιγότερο ανθεκτικοί σε narrow band interference καθώς χρησιμοποιούνται codes μικρότερου μήκους κι έτσι το σήμα απλώνεται λιγότερο στις συχνότητες. Χρησιμοποιούνται 4 ακολουθίες για τα 5.5 Mbps και 64 για τα 11 Mbps. Στην περίπτωση για παράδειγμα που θέλουμε να μεταδώσουμε στα 11 Mbps, ο CCK modulator θα δέχεται την πληροφορία ανά bytes (1.375 Mbytes/s). Τα πρώτα 6 bits χρησιμοποιούνται για να επιλεγεί ένας εκ των 64 διαθέσιμων μοναδικών ακολουθιών. Τα 2 τελευταία bits χρησιμοποιούνται για να επιλεγεί μία εκ των 4 διαθέσιμων περιστροφών του κώδικα (0, 90, 180, 270 μοίρες).

Το 802.11s είναι μία τροποποίηση του 802.11 για τα mesh networks και είναι το πρωτόκολλο που χρησιμοποιείται στο δίκτυο αυτής της διπλωματικής εργασίας. Το physical layer του 802.11s χρησιμοποιεί τα physical layers των 802.11a/b/g/n. Στα πλαίσια αυτής της διπλωματικής χρησιμοποιούμε το ελάχιστο δυνατό bitrate που είναι το 1 Mbps για να επιτύχουμε μέγιστη εμβέλεια, έτσι το physical layer του 802.11s θα χρησιμοποιήσει αυτό του 802.11b με long packet mode και διαμόρφωση DBPSK.

2.3 802.11 Medium Access Control

Το MAC Layer του 802.11 παρέχει ένα σύνολο λειτουργιών με σκοπό την διαχείριση των μεταδόσεων των κόμβων σε ένα ασύρματο δίκτυο. Πιο συγκεκριμένα, ορίζει έναν αριθμό μηχανισμών και πρωτοκόλλων που διαχειρίζονται την πρόσβαση των κόμβων στο κοινό μέσω, τον αέρα. Όπως έχει αναφερθεί, το MAC του 802.11, και κατ' επέκταση του 802.11s που χρησιμοποιούμε, βασίζεται σε physical layers όπως του 802.11b για την αποστολή και λήψη πακέτων καθώς και για λειτουργίες όπως το carrier sensing. [14, 15 ,17]

Παρακάτω παρουσιάζεται η δομή ενός MAC frame και επεξηγούνται συνοπτικά τα πεδία του:

Bytes 2	2	6	6	6	2	6	0-2312	4
Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	Frame Body	CRC

Figure 5: **802.11 MAC Frame**

Duration/ID:
Αποτελείται από 16 bits και έχει δύο χρήσεις ανάλογα τον τύπο του frame (οι τύποι αναφέρονται στο επόμενο figure). Στην περίπτωση όπου το frame είναι Power Save Poll (PS-Poll) τότε αυτό το πεδίο περιέχει το ID του κόμβου. Σε οποιαδήποτε άλλη περίπτωση περιέχει τον χρόνο που θα διαρκέσει η συγκεκριμένη μετάδοση και χρησιμοποιείται για την ενημέρωση του NAV των κόμβων για το Virtual Carrier Sensing (Αναλύεται στο 2.3.1.2).

Address Fields:
Ένα frame μπορεί να περιέχει μέχρι και 4 διευθύνσεις ανάλογα με τα ToDS και FromDS πεδία στο Frame Control τα οποία παρουσιάζονται αργότερα. Συνολικά αποτελείται από 192 bits.
Το Address 1 είναι η διεύθυνση του κόμβου προορισμού. Αν το ToDS bit είναι 1 τότε το Address 1 είναι η διεύθυνση του Access Point (AP). Αν το ToDS είναι 0 τότε το Address 1 είναι η διεύθυνση του τελικού κόμβου προορισμού.
Το Address 2 είναι η διεύθυνση του κόμβου αποστολής. Αν το FromDS bit είναι 1 τότε το Address 2 είναι η διεύθυνση του Access Point (AP). Αν το FromDS bit είναι 0 τότε το Address 2 είναι η διεύθυνση του κόμβου που μεταδίδει το frame.
Το Address 3 περιλαμβάνει την διεύθυνση του αρχικού κόμβου αποστολής του frame στην περίπτωση όπου το FromDS είναι 1. Αν το ToDS είναι 1 τότε το Address 3 είναι η διεύθυνση προορισμού.

Το Address 4 χρησιμοποιείται μόνο για frames μεταξύ Access Points και σε αυτή την περίπτωση και το ToDS και το FromDS έχουν την τιμή 1.

Sequence Control:
Αποτελείται από 16 bits και συμπεριλαμβάνει δύο πεδία, το Fragment Number και το Sequence Number. Το Fragment Number είναι ο αριθμός των fragments που ανήκουν στο ίδιο frame και το Sequence Number είναι ο αριθμός των frames σε μία ροή. Χρησιμοποιούνται για τον εντοπισμό των duplicate πακέτων.

Frame Body:
Αποτελείται μέχρι και από 2312 bytes και περιέχει τα δεδομένα προς αποστολή.

CRC:
Αποτελείται από 32 bit και περιέχει το 32-bit Cyclic Redundancy Check για τον εντοπισμό λαθών.

Το πεδίο Frame Control αποτελείται από τα εξής πεδία τα οποία εξηγούνται συνοπτικά παρακάτω:

bits	2	2	4	1	1	1	1	1	1	1	1
Protocol Version	Type	Subtype	To DS	From DS	More Frag	Retry	Pwr Mgt	More Data	WEP	Order

Figure 6: **802.11 MAC Frame Control Field**

Protocol Version:
Αποτελείται από 2 bits. Έχει σταθερή τιμή 0 και είναι δεσμευμένο για μελλοντική χρήση.

Type και Subtype:
Τα πεδία αυτά αποτελούνται από 6 bits και χρησιμοποιούνται για να καθοριστεί ο τύπος του frame, για παράδειγμα αν πρόκειται για frame δεδομένων, Management frame ή Control frame. Στο παρακάτω figure παρουσιάζονται οι τιμές των πεδίων αυτών.

Type Value b3 b2	Type Description	Subtype Value b7 b6 b5 b4	Subtype Description
00	Management	0000	Association Request
00	Management	0001	Association Response
00	Management	0010	Re-association Request
00	Management	0011	Re-association Response
00	Management	0100	Probe Request
00	Management	0101	Probe Response
00	Management	0110-0111	Reserved
00	Management	1000	Beacon
00	Management	1001	ATIM
00	Management	1010	Disassociation
00	Management	1011	Authentication
00	Management	1100	De-authentication
00	Management	1101-1111	Reserved
01	Control	0000-1001	Reserved
01	Control	1010	PS-Poll
01	Control	1011	RTS
01	Control	1100	CTS
01	Control	1101	ACK
01	Control	1110	CF End
01	Control	1111	CF End + CF-ACK
10	Data	0000	Data
10	Data	0001	Data + CF-ACK
10	Data	0010	Data + CF-Poll
10	Data	0011	Data + CF-ACK + CF-Poll
10	Data	0100	Null Function (No Data)
10	Data	0101	CF-ACK (No Data)
10	Data	0110	CF-Poll (No Data)
10	Data	0111	CF-ACK + CF-Poll (No Data)
10	Data	1000-1111	Reserved
11	Reserved	0000-1111	Reserved

Figure 7: **802.11 MAC Frame Control Field Values**

To DS:
Αποτελείται από 1 bit και παίρνει την τιμή 1 όταν το frame προορίζεται για το Distribution System. Για οποιαδήποτε άλλη περίπτωση το πεδίο αυτό παίρνει την τιμή 0.

From DS:
Αποτελείται από 1 bit και παίρνει την τιμή 1 όταν το frame προέρχεται από το Distribution System.

More Frag: (More Fragments)
Αποτελείται από 1 bit και παίρνει την τιμή 1 όταν το fragment του frame το ακολουθούν και άλλα fragments του ίδιου frame.

Retry:
Αποτελείται από 1 bit και παίρνει την τιμή 1 στην περίπτωση όπου το συγκεκριμένο fragment αποτελεί επαναμετάδοση ενός προηγούμενου fragment και χρησιμοποιείται από τον δέκτη για να αναγνωρίζει duplicate transmissions που μπορεί να συμβούν όταν χαθεί ένα ACK πακέτο.

Pwr Mgt: (Power Management)
Αποτελείται από 1 bit και δηλώνει το Power Management Mode που θα είναι ο κόμβος μετά την αποστολή του frame. Χρησιμοποιείται από κόμβους που αλλάζουν το mode αυτό από Power Save σε Active ή το ανάποδο.

More Data:
Αποτελείται από 1 bit και χρησιμοποιείται για το Power Management. Συγκεκριμένα χρησιμοποιείται από το AP (Access Point) για να δηλώσει ότι έχει πακέτα προς μετάδοση αποθηκευμένα σε buffer. Ο κόμβος μπορεί να αποφασίσει είτε να παραμείνει σε Power Saving Mode συνεχίζοντας το polling είτε να αλλάξει σε Active για να λάβει τα πακέτα.

WEP:
Αποτελείται από 1 bit και παίρνει την τιμή 1 όταν το frame είναι κωδικοποιημένο σύμφωνα με τον WEP αλγόριθμο.

Order:
Αποτελείται από 1 bit και παίρνει την τιμή 1 όταν το frame έχει μεταδοθεί με χρήση του Strictly-Ordered υπηρεσίας. Αυτή η υπηρεσία χρησιμοποιείται για χρήστες που δεν μπορούν να δεχθούν αλλαγή στην σειρά των frames.

2.3.1 Carrier Sensing

Ο μηχανισμός carrier sensing χωρίζεται σε δύο κατηγορίες: το physical carrier sensing και το virtual carrier sensing. Σκοπός του είναι να αντιλαμβάνεται πότε το μέσο είναι αδρανές και έτσι ο κόμβος που θέλει να μεταδώσει πληροφορία να αποφύγει τις συγκρούσεις με άλλους κόμβους. Αν οποιοσδήποτε από τους δύο μηχανισμούς (physical ή virtual) αντιληφθεί κάποια άλλη μετάδοση, το κανάλι χαρακτηρίζεται ως μη αδρανές.

2.3.1.1 Physical Carrier Sensing

Πριν μεταδώσει κάποιος κόμβος πρέπει να αποκτήσει πρόσβαση στο μέσο, το οποίο όμως πρέπει να το μοιράζεται και με τους υπόλοιπους κόμβους στο δίκτυο. Το MAC του 802.11 ορίζει τον μηχανισμό του Distributed Carrier Sense (DCF) ο οποίος βασίζεται στο πρωτόκολλο Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). Κατά το DCF, οι κόμβοι που θέλουν να μεταδώσουν πρέπει να βεβαιωθούν πρώτα ότι δεν μεταδίδει κάποιος άλλος κόμβος εκείνη την χρονική στιγμή. Ορίζεται ένας χρόνος Distributed Interframe Space (DIFS) και είναι ο χρόνος για τον οποίο ένας κόμβος πρέπει να βλέπει το κανάλι αδρανές, πριν αποφασίσει να αρχίσει να μεταδίδει. Αν το κανάλι δεν είναι αδρανές, θα συνεχίζει να ακούει το κανάλι μέχρι να γίνει αδρανές για χρόνο DIFS πριν μεταδώσει. Στην περίπτωση που περισσότεροι από ένας κόμβοι περιμένουν να γίνει αδρανές το κανάλι και άρα περιμένουν όλοι για χρόνο DIFS, τότε θα υπάρχουν συγκρούσεις μεταξύ των κόμβων. Έτσι ορίζεται ο backoff αλγόριθμος ο οποίος παράγει ένα backoff interval, μετά το οποίο θα επιχειρήσει να αποκτήσει πρόσβαση στο μέσο ξανά.

2.3.1.2 Virtual Carrier Sensing

Ο μηχανισμός αυτός χρησιμοποιεί έναν μετρητή, τον Network allocation Vector (NAV). Αυτός ο μετρητής κρατά μια πρόβλεψη της μελλοντικής κίνησης στο μέσο βασιζόμενο σε πληροφορία που περιέχεται στα RTS/CTS πακέτα που ανταλλάσσονται πριν την πραγματική ανταλλαγή δεδομένων ή στο πεδίο Duration/ID των πακέτων που περιέχουν τα δεδομένα. Τα Request To Send και Clear To Send είναι πακέτα τα οποία χρησιμεύουν στην ενημέρωση του NAV και δεν είναι υποχρεωτική η χρήση τους στο 802.11. Πριν την πραγματική αποστολή δεδομένων, ένας κόμβος μπορεί να στείλει ένα RTS πακέτο για να ενημερώσει τον παραλήπτη ότι έχει δεδομένα προς αποστολή. Ο παραλήπτης θα απαντήσει με ένα CTS πακέτο αν το μέσο είναι αδρανές σύμφωνα με το δικό του carrier sensing. Όταν ο αποστολέας λάβει το CTS πακέτο του παραλήπτη θα αρχίσει την μετάδοση των δεδομένων.

Ο μηχανισμός ενημέρωσης του NAV με την χρήση RTS/CTS πακέτων βοηθάει στην αντιμετώπιση του hidden node φαινομένου το οποίο συναντάται στο Physical Carrier Sensing και περιγράφεται παρακάτω.

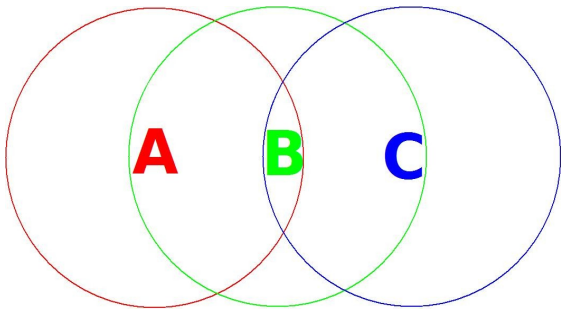


Figure 8: **Hidden Node Problem**

Έστω το εξής σενάριο, όπως φαίνεται και στο παραπάνω figure: έχουμε 3 κόμβους A, B, C και υπάρχουν τα links A-B και B-C μόνο, λόγω του ότι A και C είναι εκτός εμβέλειας. Ο κόμβος A θέλει να στείλει δεδομένα στον κόμβο B και ο κόμβος C στον B. Στην περίπτωση όπου δεν χρησιμοποιούνται RTS/CTS πακέτα, το carrier sensing του A και του C θα αποφασίσει ότι το μέσο είναι αδρανές, αφού δεν μπορούνε να ακούσουνε τις μεταξύ τους μεταδόσεις λόγο εμβέλειας και έτσι θα αρχίσουν να μεταδίδουν. Στον κόμβο B όπου θα φτάσουν τα δεδομένα από τον A και τον C θα έχουμε collision. Στην περίπτωση όμως που χρησιμοποιούσαμε τα RTS/CTS πακέτα, τότε ο κόμβος A και ο κόμβος C θα έστελναν πρώτα στον κόμβο B ένα RTS πακέτο. Ο κόμβος B για να επιτρέψει την μετάδοση σε κάποιον από τους δύο πρέπει να απαντήσει με ένα CTS πακέτο. Αφού γνωρίζει την πρόθεση και του A και του C να στείλουν δεδομένα, θα επιτρέψει αρχικά στον έναν και μετά στον άλλον. Έτσι αποφεύγεται το collision.

2.3.1.3 Interframe Spaces (IFS)

Εκτός από το χρονικό διάστημα DIFS, το MAC του 802.11 ορίζει και άλλα χρονικά διαστήματα του PHY: Short Interframe Space (SIFS), Point Coordination Interframe Space (PIFS) και Extended Interframe Space (EIFS).

Το SIFS αφορά το χρονικό διάστημα μεταξύ μεταδόσεων πακέτων δεδομένων και ACKs ή μετάδοση fragments πακέτων. Είναι το ελάχιστο χρονικό διάστημα που ορίζεται και χρησιμοποιείται όταν ο κόμβος έχει ήδη πρόσβαση στο μέσο. Το ότι χρησιμοποιείται το ελάχιστο χρονικό διάστημα μεταξύ μεταδόσεων μιας ροής δεδομένων αποτρέπει άλλους κόμβους να παρέμβουν σε αυτό το κενό και να αποκτήσουν πρόσβαση στο μέσο. Έτσι δίνεται προτεραιότητα στις ροές δεδομένων που έχουν ήδη αρχίσει να μεταδίδονται.

Το PIFS είναι ένα χρονικό διάστημα που χρησιμοποιείται στην περίπτωση όπου για physical carrier sensing χρησιμοποιείται η Point Coordination Function (PCF) η οποία μπορεί να χρησιμοποιείται αντί για την DCF. Δεν χρησιμοποιείται όμως ευρέως, μιας και μόνο η DCF είναι υποχρεωτική, για αυτό και δεν αναλύεται. Το PIFS έχει να κάνει με τον χρόνο που χρειάζονται τα Access Points να αποκτήσουν πρόσβαση στο μέσο πριν από του άλλους κόμβους και ορίζεται ως το SIFS συν ένα slot time.

Το DIFS που αναλύθηκε ήδη ορίζεται ως το SIFS συν δύο time slots.

Τέλος το EIFS, το οποίο είναι και το μεγαλύτερο χρονικό διάστημα, χρησιμοποιείται όταν πακέτο δεν λήφθηκε σωστά. Ο κόμβος ο οποίος δεν έλαβε σωστά ένα πακέτο, δεν μπορεί να καταλάβει το duration πεδίο του πακέτου για να ενημερώσει το NAV του Virtual Carrier Sensing του. Σκοπός αυτού του χρονικού διαστήματος είναι να αποτρέψει τον κόμβο να συγκρουστεί με μελλοντικά πακέτα της ίδιας ροής, επιτρέποντας στον αποστολέα να αναγνωρίζει ότι ο παραλήπτης δεν μπόρεσε να λάβει σωστά το πακέτο. Το EIFS ξεκινάει όταν αντιληφθεί ότι το μέσο είναι αδρανές μετά την λήψη εσφαλμένου πακέτου, αγνοώντας τον carrier sensing μηχανισμό. Όταν ληφθεί ένα πακέτο σωστά κατά την διάρκεια του EIFS, ο κόμβος τερματίζει το EIFS και επιστρέφει στην αρχική μέθοδο για πρόσβαση στο μέσο (carrier sensing). Το EIFS ορίζεται ως ο χρόνος μετάδοσης ενός ACK πακέτου στο χαμηλότερο δυνατό bitrate + SIFS + DIFS.

Slot Time (μs)	SIFS (μs)	DIFS (μs)	PIFS (μs)
20	10	50	30

Figure 9: **IEEE 802.11b Slot time and Interframe Spaces values**

2.3.1.4 Backoff Algorithm

Στην περίπτωση που ο carrier sense μηχανισμός ενός κόμβου αντιληφθεί ότι το μέσο δεν είναι αδρανές τότε ο κόμβος θα πρέπει να περιμένει χρόνο DIFS κατά την διάρκεια του οποίου το μέσο πρέπει να παραμείνει αδρανές πριν επιχειρήσει να μεταδώσει. Επίσης στην περίπτωση που ένα πακέτο δεν λήφθηκε σωστά, πρέπει ο κόμβος να περιμένει χρόνο EIFS κατά τον οποίο το μέσο πρέπει να είναι αδρανές. Μετά από αυτά τα διαστήματα (DIFS και EIFS) ο κόμβος θα παράξει ένα τυχαίο χρονικό διάστημα κατά το οποίο θα περιμένει επιπλέον πριν μεταδώσει ξανά, εκτός και αν αυτός ο χρόνος περιέχει ήδη μία μη μηδενική τιμή, οπότε δεν παράγει νέα αλλά χρησιμοποιεί αυτή. Αυτός ο μηχανισμός βοηθάει στον περιορισμό τον συγκρούσεων τον κόμβων που καθυστερούν τις μεταδόσεις τους για τον ίδιο λόγο, για παράδειγμα όταν δύο κόμβοι περιμένουν έναν τρίτο να σταματήσει να μεταδίδει για να μεταδώσουν αυτοί. Το backoff time ορίζεται ως εξής:

Backoff Time = Random() x aSlotTime

όπου:

Random() :
Ένας ψευδοτυχαίος αριθμός που προέρχεται από κανονική κατανομή στο διάστημα [0,CW]. Το Contention Window (CW) είναι ένας αριθμός που προκύπτει από τα χαρακτηριστικά του PHY το οποίο ορίζει μια ελάχιστη και μια μέγιστη CW τιμή. Είναι σημαντικό να υπάρχει στατιστική ανεξαρτησία μεταξύ αυτών των τυχαίων αριθμών.

aSlotTime:
Το διάστημα κάθε slot το οποίο ορίζεται και αυτό από τα χαρακτηριστικά του PHY. Ορίζεται με τέτοιον τρόπο ώστε ένας κόμβος να προλαβαίνει να καταλάβει αν κάποιος άλλος κόμβος έχει αποκτήσει πρόσβαση στο μέσο από το αμέσως προηγούμενο slot. Αυτό έχει σαν αποτέλεσμα να μειώνεται η πιθανότητα σύγκρουσης κατά 50%.

Αρχικά το CW έχει την ελάχιστη τιμή που ορίζεται από το PHY Layer. Σε κάθε αποτυχημένη προσπάθεια μετάδοσης αυξάνεται το CW μέχρι να φτάσει την μέγιστη τιμή του. Το βήμα με το οποίο αυξάνεται το CW είναι 2^n-1. Κάθε retry γίνεται με την αποστολή όλων τον σχετιζόμενων frames, που απέχουν μεταξύ τους διάστημα SIFS. Όταν το CW πάρει την μέγιστη τιμή του, την διατηρεί μέχρι να γίνει reset. Το CW θα γίνει reset στην ελάχιστη τιμή του μετά από μια επιτυχημένη μετάδοση ενός frame. Το ελάχιστο CW στο 802.11b ορίζεται στα 15 slots.

2.4 802.11s Routing

Τα πρωτόκολλα για την δρομολόγηση σε ασύρματα mesh δίκτυα έχουν 3 κατηγορίες: proactive, reactive και hybrid [18, 19, 20]. Κατά την proactive δρομολόγηση, οι διαδρομές που θα ακολουθήσουν τα πακέτα στο δίκτυο αποφασίζονται πριν οποιαδήποτε κίνηση δεδομένων. Το πρωτόκολλο κρατά πίνακες δρομολόγησης προς όλους τους κόμβους, ακόμα και αν δεν χρειάζονται. Οι πίνακες αυτοί ενημερώνονται από την περιοδική αποστολή control πακέτων του κόμβου. Αυτό έχει σαν αποτέλεσμα να σπαταλιέται bandwidth αλλά και να έχει κάθε κόμβος γρήγορη πρόσβαση σε μια διαδρομή για να ξεκινήσει άμεσα την αποστολή των πακέτων του. Μερικά παραδείγματα τέτοιων πρωτοκόλλων είναι τα Optimized Link State Routing protocol (OLSR), Destination-sequenced distance vector routing protocol (DSDV) και το Wireless Routing protocol (WRP).

Στα reactive πρωτόκολλα δρομολόγησης, μια διαδρομή μεταξύ κόμβων δημιουργείται μόνο όταν ένας κόμβος θέλει να στείλει δεδομένα σε κάποιον άλλον κόμβο (on-demand). Αυτό μειώνει αρκετά την σπατάλη bandwidth από τα control πακέτα για την διατήρηση όλων των διαδρομών προς όλους τους κόμβους όπως συμβαίνει στα proactive πρωτόκολλα δρομολόγησης. Από την άλλη όμως αυξάνεται ο χρόνος μέχρι ένας κόμβος να αρχίσει να στέλνει τα δεδομένα του, καθώς πρώτα πρέπει να αναζητήσει μια διαδρομή. Τα reactive πρωτόκολλα δρομολόγησης χωρίζονται σε δύο κατηγορίες: source routing και hop-by-hop routing. Στα source routed on-demand πρωτόκολλα, κάθε πακέτο δεδομένων περιέχει όλη την πληροφορία για την διαδρομή προς τον προορισμό. Στα hop-by-hop, κάθε πακέτο περιέχει μόνο την διεύθυνση προορισμού και την διεύθυνση του επόμενου κόμβου. Παραδείγματα reactive πρωτοκόλλων δρομολόγησης είναι τα Dynamic Source Routing Protocol (DSR), ad hoc On-demand Distance Vector (AODV) και το Dynamic MANET On-demand (DYMO).

Τα hybrid πρωτόκολλα δρομολόγησης συνδυάζουν και τις δύο παραπάνω τεχνικές. Ένα τέτοιο πρωτόκολλο είναι και το Hybrid Wireless Mesh Protocol (HWMP) το οποίο είναι το default πρωτόκολλο που χρησιμοποιείται στο 802.11s και το οποίο χρησιμοποιήθηκε και σε αυτή την διπλωματική εργασία. Είναι ένα πρωτόκολλο που λειτουργεί στο MAC Layer. Ορίζονται 4 control packets: Root Announcement (RANN), Path Request (PREQ), Path Reply (PREP) και Path Error (PERR). Τα RANN, PREQ και PREP περιέχουν 3 σημαντικά πεδία: Destination Sequence Number (DSN), Time To Live (TTL) και ένα πεδίο για μια μετρική. [18]

Για την on-demand τεχνική, βασίζεται στο Radio-Metric Ad hoc On Demand Distance Vector (RM-AODV) πρωτόκολλο που είναι μια προσαρμογή του Ad hoc On Demand Distance Vector (AODV) πρωτοκόλλου. Τα DNS και TTL πεδία χρησιμοποιούνται για να αποφευχθεί το count to infinity πρόβλημα και η μετρική χρησιμοποιείται στο να επιλέγονται καλύτερες διαδρομές από αυτές που θα επιλέγονταν υπολογίζοντας απλά των αριθμό των ενδιάμεσων κόμβων προς τον κόμβο προορισμού.

Όταν ένας κόμβος θέλει να στείλει δεδομένα κάνει broadcast ένα PREQ πακέτο που περιλαμβάνει το MAC address του προορισμού. Οι ενδιάμεσοι κόμβοι που λαμβάνουν το PREQ ενημερώνουν την διαδρομή και την μετρική της προς τον αρχικό κόμβο και το προωθούν. Όταν ο κόμβος προορισμού λάβει τα PREQ πακέτα θα ελέγξει την μετρική και θα επιλέξει την βέλτιστη διαδρομή προς τον αρχικό κόμβο. Ο κόμβος προορισμού θα δημιουργήσει ένα PREP πακέτο και θα το στείλει στον αρχικό κόμβο, ακολουθώντας την βέλτιστη διαδρομή που ορίστηκε από κάποιο PREQ. Οι ενδιάμεσοι κόμβοι που λαμβάνουν το PREP ενημερώνουν την διαδρομή προς τον αρχικό κόμβο και την μετρική για αυτή την διαδρομή και προωθούν το πακέτο προς τον αρχικό κόμβο που έστειλε το PREQ. Όταν ο αρχικός κόμβος λάβει το PREP αρχίζει η επικοινωνία.

Παρακάτω φαίνονται οι διαδρομές των PREQ πακέτων που γίνονται broadcast από τον κόμβο Α που αναζητά διαδρομή προς τον Ε, όπως αναλύθηκαν πριν:

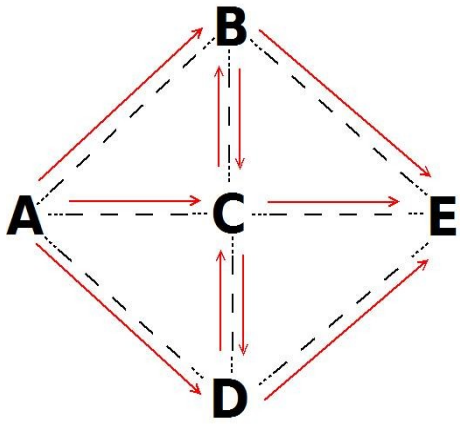


Figure 10: **Broadcast of Reactive PREQ packets in HWMP**

Παρακάτω φαίνεται η διαδρομή του PREP πακέτου, όπως αναλύθηκε πριν (έστω βέλτιστη διαδρομή A-C-E).

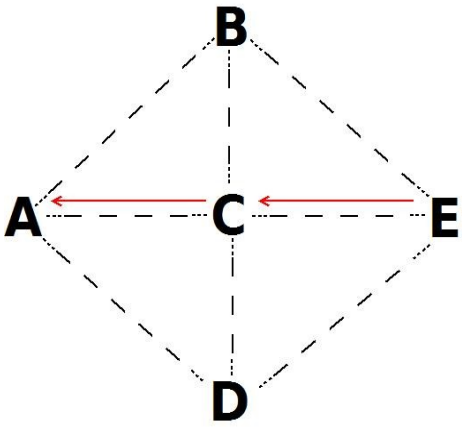


Figure 11: **Unicast of PREP in HWMP**

Στην on-demand τεχνική το HWMP χρησιμοποιεί την expanding ring αναζήτηση με σκοπό να περιορίσει το flooding των routing πακέτων.

Στην proactive τεχνική ορίζεται ένας κόμβος ως root, ο οποίος μπορεί να είναι internet gateway και δημιουργείται ένα δέντρο με ρίζα αυτόν τον κόμβο. Υπάρχουν δύο τρόποι για την δημιουργία αυτού του δέντρου: proactive PREQ (Path Request) ή proactive RANN (Route Announcement) [18].

Κατά το Proactive PREQ, ο root κάνει περιοδικά broadcast PREQ πακέτα που χαρακτηρίζονται από έναν μοναδικό αριθμό. Οι υπόλοιποι κόμβοι που λαμβάνουν τα PREQs, ανανεώνουν τις μετρικές τους και τα πεδία TTL,Hop Count και Metric του PREQ. Καταγράφουν την διαδρομή προς το root και το κάνουν ξανά broadcast. Μία διαδρομή από τον root προς έναν κόμβο καθιερώνεται όταν το Proactive PREP flag στο PREQ πακέτο γίνει 1 από τον root. Ο κόμβος προορισμού τότε κάνει unicast ένα RREP (Route Reply) πακέτο στον root και έτσι καθιερώνεται και η αντίστροφη διαδρομή. Αν το PREP flag δεν είναι 1 τότε ο κόμβος δεν απαντάει με RREP στο PREQ που λήφθηκε. Οι διαδρομές από τον root στους κόμβους καθιερώνονται μόνο όταν υπάρχουν δεδομένα προς μετάδοση για κάποιον κόμβο και αυτό γίνεται για να μειώνεται το routing overhead.

Κατά το Proactive RANN, ο root κάνει περιοδικά broadcasts RANN πακέτα. Αυτά χρησιμοποιούνται μόνο για να διαδίδονται οι μετρικές τον διαδρομών και όχι για να αλλάξουν οι πίνακες δρομολόγησης. Όταν ένας κόμβος θέλει να δημιουργήσει ή να ενημερώσει μια διαδρομή προς τον root τότε θα κάνει unicast ένα PREQ πακέτο στον root με το Destination Only flag ενεργοποιημένο. Η διαδρομή που θα ακολουθήσει το PREQ πακέτο είναι γνωστή καθώς είναι γνωστά τα metrics των links από τα RANN broadcasts του root. Ο root τότε θα απαντήσει με ένα PREP και έτσι καθιερώνεται η διαδρομή.

Στο παρακάτω figure φαίνεται ένα παράδειγμα broadcast στην proactive τεχνική με σκοπό την κατασκευή του δέντρου προς τον root.

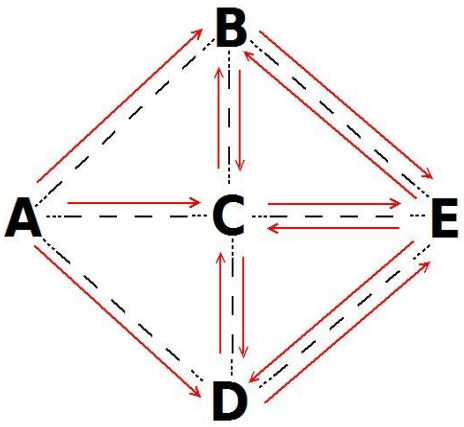


Figure 12: **Broadcast of Proactive PREQ and Proactive RANN packets in HWMP**

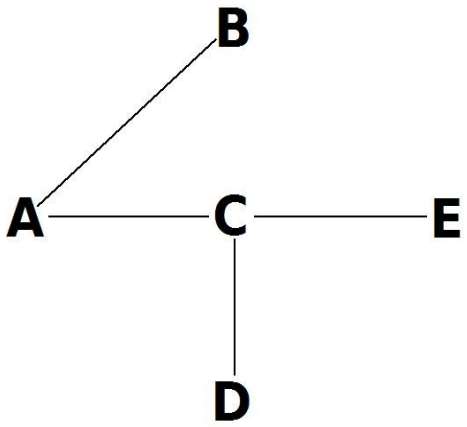


Figure 13: **Example of a Proactive tree formed in HWMP root mode**

Στο HWMP, proactive και reactive routing λειτουργούν ταυτόχρονα. Περιλαμβάνει τις εξής λειτουργίες:

Root Announcement (Broadcast) : ειδοποιεί τους κόμβους ότι έχει οριστεί ένας κόμβος ως root και ενημερώνει κάθε κόμβο για την απόσταση από αυτόν.

Root Request (Broadcast ή Unicast) : ζητά από τον κόμβο προορισμού να δημιουργήσει μία αντίστροφη διαδρομή από τον κόμβο που έστειλε το Root Request.

Root Reply (Unicast) : δημιουργεί μια διαδρομή προς τον αρχικό κόμβο που έστειλε το Root Request και επιβεβαιώνει την αντίστροφη διαδρομή (από τον αρχικό κόμβο στον κόμβο προορισμού).

Root Error (Broadcast) : οι κόμβοι παρακολουθούν τα ενεργά links που έχουν δημιουργηθεί. Όταν ένα link αποτύχει τότε στέλνεται ένα Route Error μήνυμα για να ειδοποιηθούν οι υπόλοιποι κόμβοι για την απώλεια του link.

Η default μετρική που χρησιμοποιεί το HWMP πρωτόκολλο είναι η airtime metric και απεικονίζει το κόστος του καναλιού σε ένα link για την μετάδοση ενός frame. Το airtime cost ($C_{a)}$ αυτό, ορίζεται ως [22]:

$$C_a = [O_{ca} + O_p + B_t/r] / (1-e_{pt})$$

όπου:

O_{ca}: channel access overhead
O_p: protocol overhead
B_t: number of bits in the test frame
r: bitrate in Mbps
e_{pt}: frame error rate

Κεφάλαιο 3

Υλοποίηση

3.1 Εξοπλισμός

Στα πλαίσια αυτής της διπλωματικής εργασίας υλοποιήθηκε ένα demo network στο Πολυτεχνείο Κρήτης με σκοπό την μελέτη του 802.11s. Τοποθετήθηκαν 4 κόμβοι στα κτήρια A1, A2, M3 και K4. Η εγκατάσταση ήταν προσωρινή για τις πειραματικές ανάγκες της διπλωματικής εργασίας. Στο επόμενο figure φαίνεται η τοπολογία αυτού του demo δικτύου :

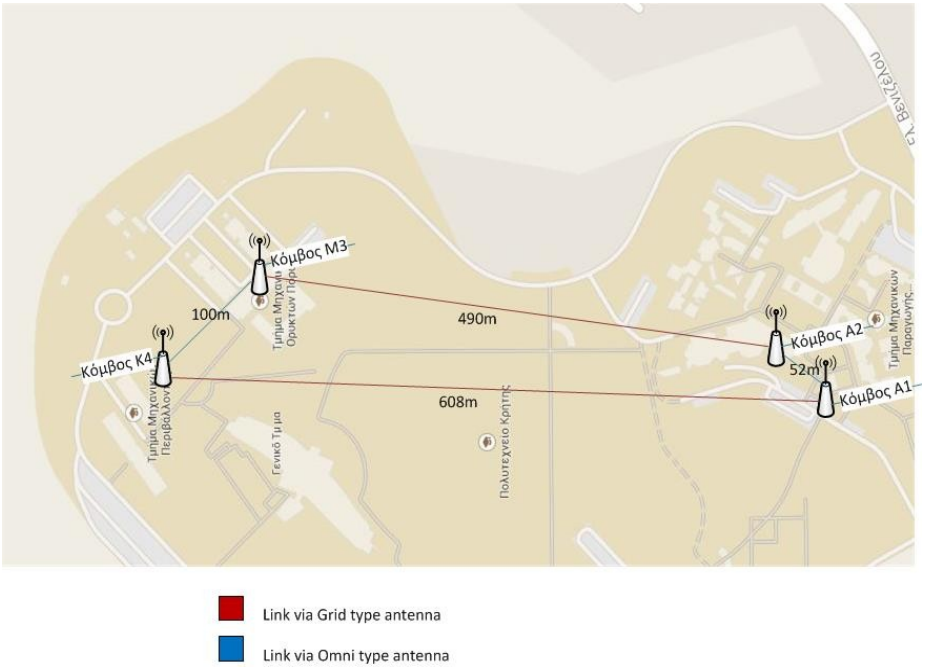


Figure 14: **TUC Demo network**

Τα links των κόμβων M3-A2 και K4-A1 επιτυγχάνονται με τις grid antennas ενώ τα links A1-A2 και K4-M3 επιτυγχάνονται με τις omni directional antennas. Για την αποφυγή interference μεταξύ των 2 grid links έχουμε διαφορετικό polarization στις κεραίες. Κάθε κόμβος έχει και μια Ethernet σύνδεση μέσω της POE τροφοδοσίας που παρέχει πρόσβαση στο backbone network του Πολυτεχνείου. Παρακάτω παρουσιάζονται τα χαρακτηριστικά κάθε κόμβου στο δίκτυο.

Board serial number	WN514599_1037
Hostname	Voyage1
Mac Address (Mesh-0)	00:25:86:e4:c3:14
Mac Address (Mesh-1)	00:25:86:e4:b7:6e
IP Address (Mesh-0)	10.10.10.1
IP Address (Mesh-1)	10.10.10.2
IP Address (Ethernet)	147.27.61.1
Omni Antenna	Mesh0
Grid Antenna	Mesh1
Polarization Grid Antenna	Vertical

Figure 15: **Node 1 of TUC Network (M3 building)**

Board serial number	WN456261_1004
Hostname	Voyage2
Mac Address (Mesh-0)	f8:d1:11:ce:71:8f
Mac Address (Mesh-1)	64:70:02:9f:ae:be
IP Address (Mesh-0)	10.10.10.3
IP Address (Mesh-1)	10.10.10.4
IP Address (Ethernet)	147.27.61.2
Omni Antenna	Mesh0
Grid Antenna	Mesh1
Polarization Grid Antenna	Vertical

Figure 16: **Node 2 of TUC Network (A2 building)**

Board serial number	WN470954_1012
Hostname	Voyage3
Mac Address (Mesh-0)	64:70:02:9f:b0:58
Mac Address (Mesh-1)	64:70:02:9f:b1:03
IP Address (Mesh-0)	10.10.10.5
IP Address (Mesh-1)	10.10.10.6
IP Address (Ethernet)	147.27.61.3
Omni Antenna	Mesh1
Grid Antenna	Mesh0
Polarization Grid Antenna	Horizontal

Figure 17: **Node 3 of TUC Network (A1 building)**

Board serial number	WN514648_1037
Hostname	Voyage4
Mac Address (Mesh-0)	00:25:86:e5:1b:37
Mac Address (Mesh-1)	00:27:19:ba:11:e5
IP Address (Mesh-0)	10.10.10.7
IP Address (Mesh-1)	10.10.10.8
IP Address (Ethernet)	147.27.61.4
Omni Antenna	Mesh1
Grid Antenna	Mesh0
Polarization Grid Antenna	Horizontal

Figure 18: **Node 4 of TUC Network (K4 building)**

Για την υλοποίηση του παραπάνω δικτύου χρησιμοποιήθηκαν τα components που παρουσιάζονται παρακάτω.

Χρησιμοποιήθηκαν ως κόμβοι τα boards ALIX2D2 και ALIX3D2 της PC ENGINES (Euro for scale).

CPU	500 MHz AMD Geode LX800
DRAM	256 MB DDR DRAM
Storage	CompactFlash socket, 44 pin IDE header
Power	DC jack or passive POE, min. 7V to max. 20V
Expansion	2 miniPCI slots, LPC bus
Connectivity	2 Ethernet channels (Via VT6105M 10/100)
I/O	DB9 serial port, dual USB port
Board size	6 x 6" (152.4 x 152.4 mm) - same as WRAP.1E
Firmware	tinyBIOS

Figure 19: **ALIX2D2 Specs**

Figure 20: **ALIX2D2 Front**

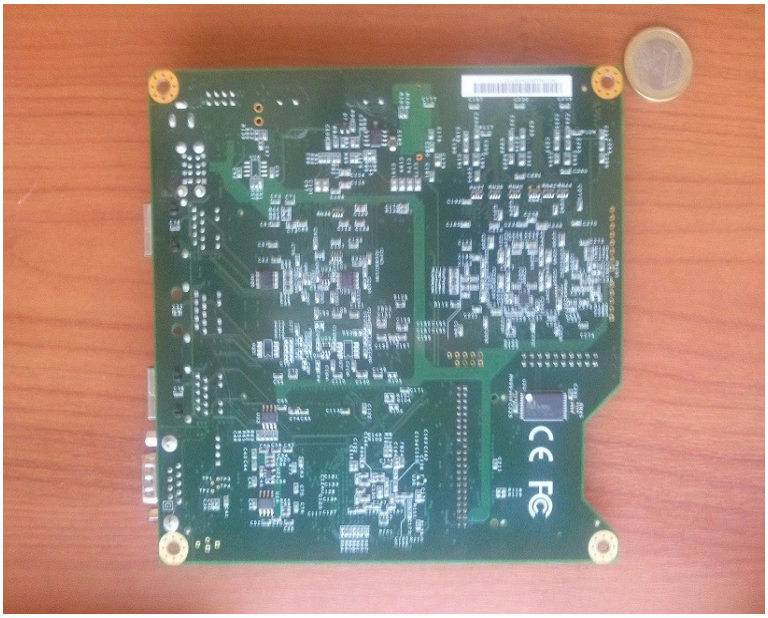


Figure 21: **ALIX 2D2 Back**

CPU	500 MHz AMD Geode LX800
DRAM	256 MB DDR DRAM
Storage	CompactFlash socket
Power	DC jack or passive POE, min. 7V to max. 20V
Expansion	2 miniPCI slots, LPC bus
Connectivity	1 Ethernet channels (Via VT6105M 10/100)
I/O	DB9 serial port, dual USB port
Board size	100 x 160 mm - same as WRAP.2E
Firmware	tinyBIOS

Figure 22: **ALIX3D2 Specs**

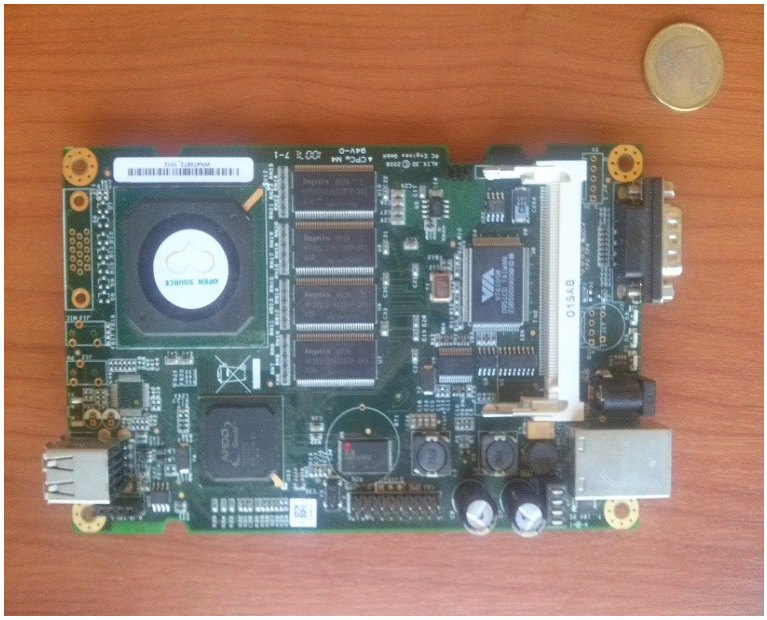


Figure 23: **ALIX3D2 Front**

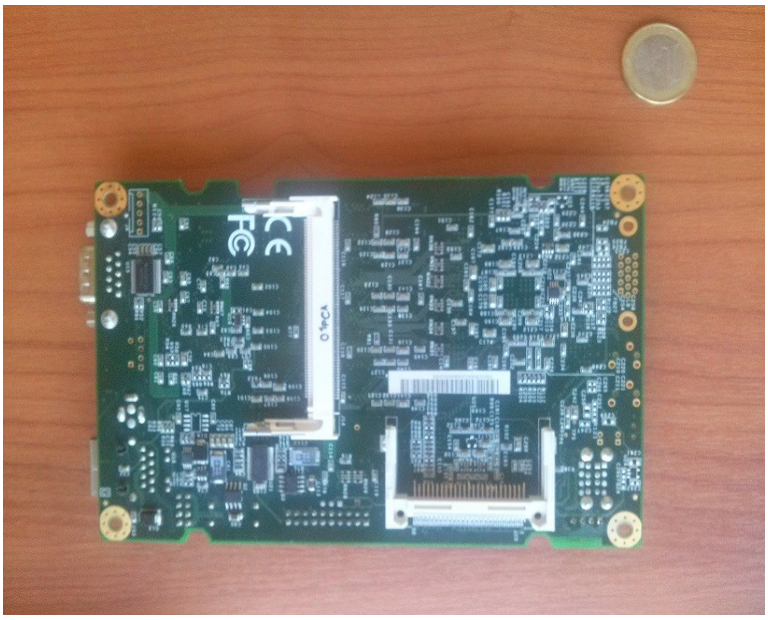


Figure 24: **ALIX3D2 Back**

Για τα network interfaces χρησιμοποιήθηκαν οι mini PCI κάρτες της TP-LINK TL-WN360G και TL-WN861N (Euro for scale).

Standards	IEEE 802.11g, IEEE 802.11b
Wireless Signal Rates with Automatic Fallback	802.11g: Up to 54Mbps (Dynamic) 802.11b: Up to 11Mbps (Dynamic)
Frequency Range	2.4 – 2.4835GHz
Wireless Transmit Power	20dBm (max. EIRP)
Modulation Type	1Mbps DBPSK, 2Mbps DQPSK, 5.5/11Mbps CCK, 6/9/12/18/24/36/48/54Mbps OFDM
Receiver Sensitivity	54Mbps: -68dBm @ 10% PER 11Mbps: -85dBm @ 8% PER 6Mbps: -88dBm @ 10% PER 1Mbps: -90dBm @ 8% PER 256Kbps: -105dBm @ 8% PER
Wireless Security	64/128/152 bit WEP, WPA/WPA2, WPA-PSK/WPA2-PSK (TKIP/AES)
Interface	Mini PCI
Antenna Connector	Ultra-Mini SMT-GSC Antenna Connector (compatible with U.FL)
Certifications	CE, FCC
Dimensions	2.4 x 1.2 x 0.13 in. (60 x 30 x 3.3mm)

Figure 25: **TL-WN360G Specs**

Figure 26: **TL-WN360G Front**

Figure 27: **TL-WN360G Back**

Standards	IEEE 802.11n, IEEE 802.11g, IEEE 802.11b
Wireless Signal Rates with Automatic Fallback	802.11n: Up to 300Mbps (Dynamic) 802.11g: Up to 54Mbps (Dynamic) 802.11b: Up to 11Mbps (Dynamic)
Frequency Range	2.4 – 2.4835GHz
Wireless Transmit Power	20dBm (max. EIRP)
Modulation Type	16-QAM, 64-QAM, CCK, DBPSK, DQPSK, OFDM
Receiver Sensitivity	270Mbps: -68dBm @ 10% PER 130Mbps: -68dBm @ 10% PER 108Mbps: -68dBm @ 10%PER 54Mbps: -68dBm @ 10% PER 11Mbps: -85dBm @ 8% PER 6Mbps: -88dBm @ 10% PER 1Mbps: -90dBm @ 8% PER
Wireless Security	64/128/152 bit WEP, WPA/WPA2, WPA-PSK/WPA2-PSK (TKIP/AES)
Interface	Mini PCI
Antenna Connector	Ultra-Mini SMT-GSC Antenna Connector (compatible with U.FL)
Certifications	CE, FCC
Dimensions	2.4 x 2.0 x 0.1 in. (60 x 51 x 3.3mm)

Figure 28: **TL-WN861N Specs**

Figure 29: **TL-WN861N Front**

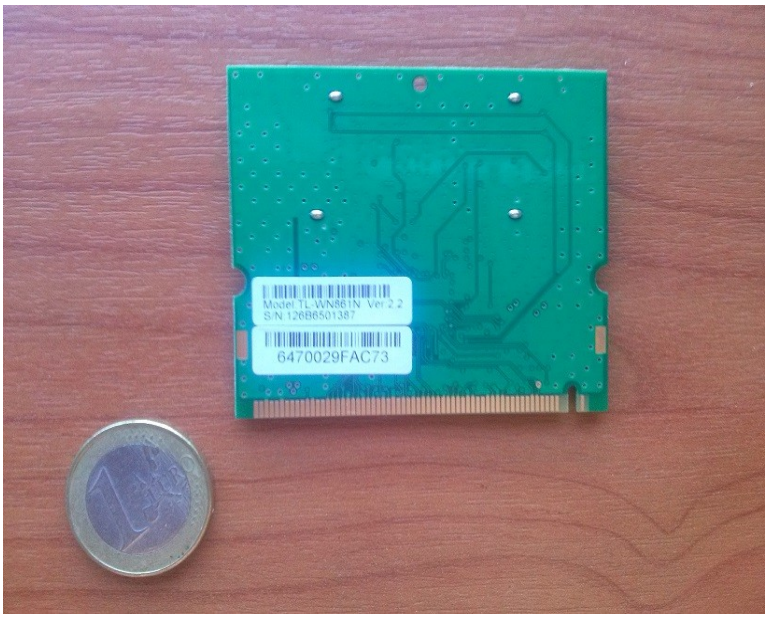


Figure 30: **TL-WN861N Back**

Χρησιμοποιήθηκαν δύο τύποι κεραιών, η Omni Directional κεραία WN722N και η Grid Antenna D2400G24A09 (Euro for scale).

Frequency	2.4GHz
Impedance	50 Ohms
Gain	4 dBi
Radiation	Omni-directional
VSWR (MAX)	1.92:1
HPBW/H	360º
HPBW/V	15º
Polarization	Linear, Vertical
Connector	SMA

Figure 31: **WN722N Omni Directional Antenna Specs**

Figure 32: **WN722N Omni Directional Antenna**

Frequency	2.4 GHz
Gain	24 dBi
Beam width	11°
VSWR	<1.4
Polarization	Vertical/Horizontal
Max Power	100 W
Impedance	50 Ohms
Cross Polarization	<20 dB
Connector	N-50K
Diameter (mm)	600x900
Weight (Kg)	3.05

Figure 33: **D2400G24A09 Grid Antenna Specs**

Figure 34: **D2400G24A09 Grid Antenna**

Για την σύνδεση τους χρησιμοποιήθηκαν οι απαραίτητοι adaptors (Euro for scale):

Omni Directional Antenna to mini PCI Interface : RP-SMA female to Pigtail U.FL male

Grid antenna to mini PCI Interface : N-Type male to Pigtail U.FL male

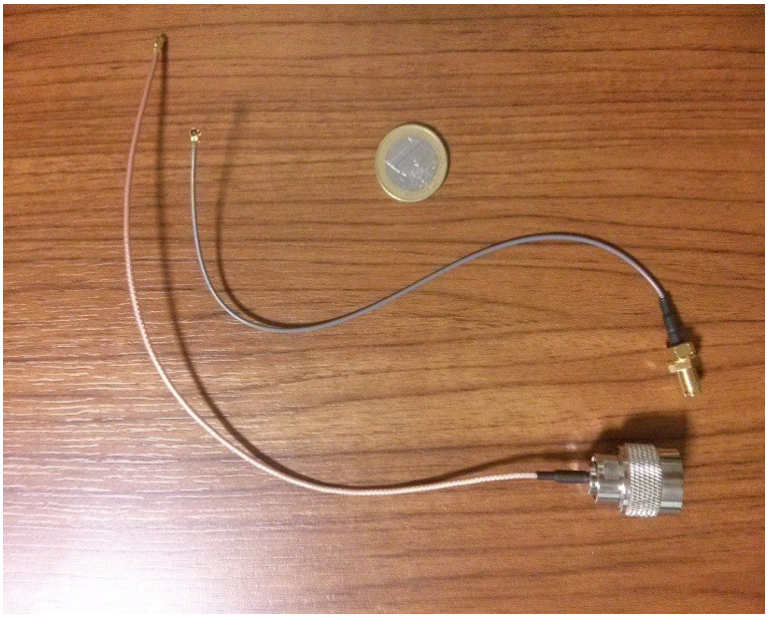


Figure 35: **U.FL to RP-SMA Male and U.FL to Type-N Connectors**

Χρησιμοποιήθηκε το προϊόν “Station Box ALU” της RF Elements. Είναι ένα enclosure από χυτό αλουμίνιο τύπου IP65 που προσφέρει εξαιρετική αντοχή στις καιρικές συνθήκες και βελτιωμένη προστασία στον RF θόρυβο. Η εσωτερική πλάκα στήριξης είναι κατασκευασμένη από ABS και έχει προσχηματισμένες τρύπες για την τοποθέτηση των πιο διαδεδομένων πλατφορμών όπως MicroTik, Alix2 κλπ.

Για την προσαρμογή του U.FL to Type-N connector στο κουτί χρησιμοποιήθηκε ένα N-Type Female Bulkhead to N-Type Female adapter. Επίσης χρειάστηκε και ένα N-Type Male to N-Type Male adapter καθώς και το καλώδιο του feeder της κεραίας είναι N-Type Female.

Για την στερέωση του U.FL to Type-N connector στο κουτί χρησιμοποιήθηκε μία ροδέλα εξωτερικής διαμέτρου 2,5cm με εσωτερικό άνοιγμα 0.90cm και πάχους 2mm. Η ροδέλα κολλήθηκε στην 4xN θέση του enclosure με χρήση εποξικής κόλλας δύο στοιχείων (Epoxy metal) της εταιρείας Bison.

Στα επόμενα figures φαίνονται τα παραπάνω components μαζί με ένα ALIX2D2 boards που έχουν συνδεθεί πάνω του 2 TL-WN861N Wi-Fi κάρτες.

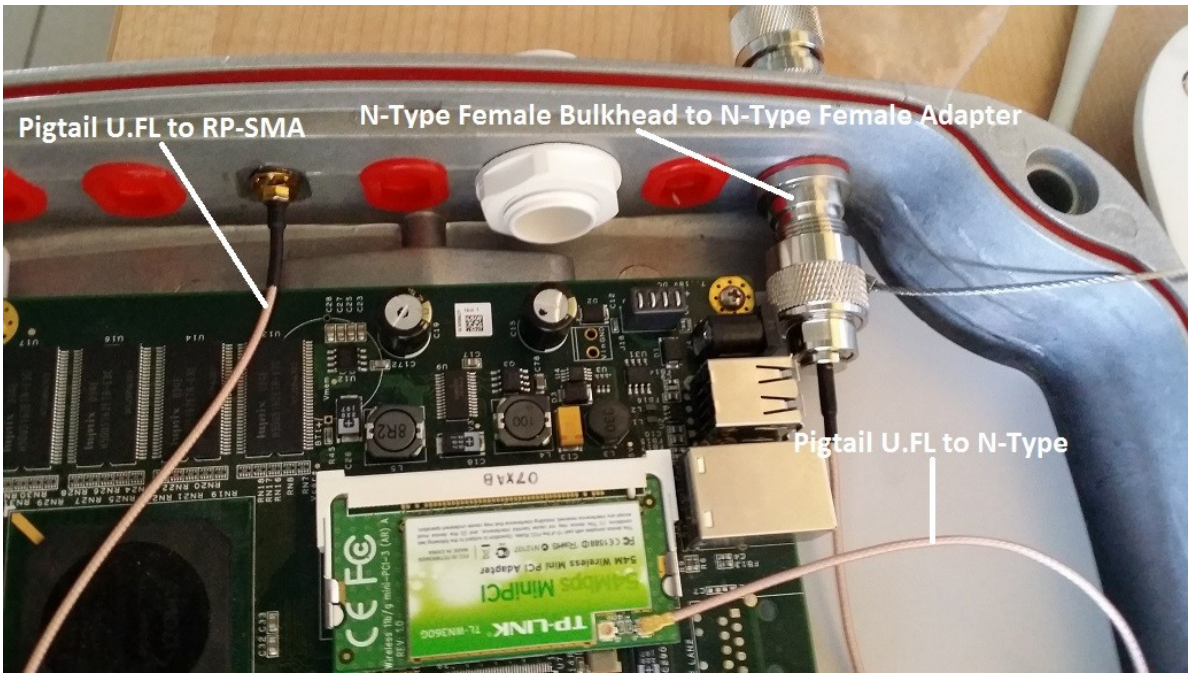


Figure 36: **Connectors used in box (inside view)**

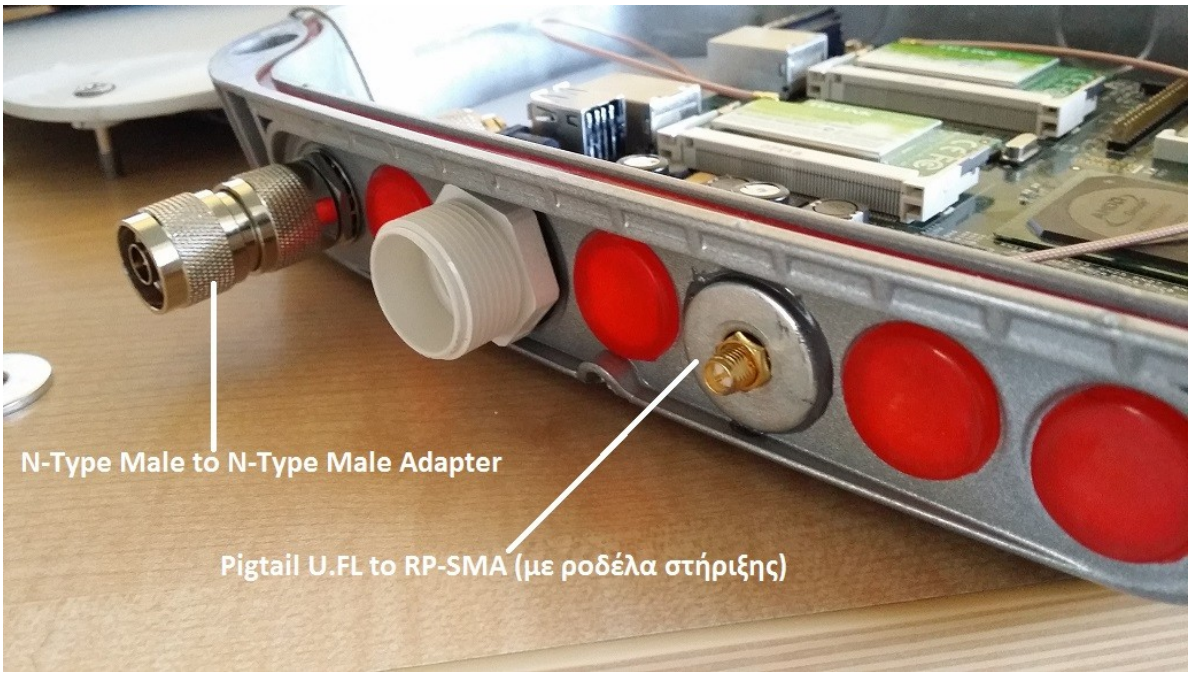


Figure 37: **Connectors used in box (outside view)**

Για την τροφοδοσία χρησιμοποιήθηκαν αρχικά απλά τροφοδοτικά και αργότερα όταν στήθηκαν οι κόμβοι στο demo δίκτυο τροφοδοτήθηκαν με Power Over Ethernet (POE). Το POE line συνδέεται με την Ethernet θύρα του κόμβου και το LAN line με μια θύρα του switch που να παρέχει λειτουργία POE. Το POE που χρησιμοποιήθηκε είναι το UBIQUITI POE-15 (15V, 0.8A, 12W).

Output Voltage	15VDC @0.8A
Input Voltage	90-260VAC @47-63Hz
Input Current	0.3A @120VAC, 0.2A @230VAC
Inrush Current	<15A peak @120VAC, <30A peak @230VAC
Efficiency	70+%
Output Ripple	1% Max
Switching Frequency	200kHz
Line Regulation	+/- 0.5%
Load Regulation	+/- 1%
Operating Temperature	-10^{o}C to +60^{o}C
Storage Temperature	-20^{o}C to +85^{o}C
Operating Humidity	5% to 90% non condensing
Size (LxWxH)	85x43x30 mm
Weight	113grams
AC Ceonnector	IEC-320 C6
Data IN / POE	RJ45 Shielded Socket
80% Current Indicator	Power LED will change color
Surge Protection	Common Mode
Clamping Protection	11V Data, 77.5V Power
Max Surge Discharge	1200A (8/20uS) Power
Peak Pulse Current	36A (10/1000uS) Data
Shunt Capacitance	<5pf data
Response Time	<1nS
Compliance	UL, EN55022 (CISPR22) class B, Meets CE

Figure 38: **UBIQUITI POE-15 (15V, 0.8A, 12W) Specs**

Figure 39: **UBIQUITI POE-15 (15V, 0.8A, 12W)**

Πριν την εγκατάσταση των κόμβων στα επιλεγμένα σημεία έγινε αδιαβροχοποίηση και μόνωση των εξωτερικών διεπαφών με χρήση αυτο-βουλκανιζόμενης ταινίας της εταιρείας HPX.

Figure 40: **Waterproofing external Connections**

Τα επιμέρους τμήματα του κάθε κόμβου τοποθετήθηκαν πάνω σε γαλβανιζέ σωλήνα πάχους 1 ½ ίντσας και μήκους 2 μέτρων. Το σασί του κόμβου τοποθετήθηκε στο πίσω πάνω μέρος της σωλήνας, με τα ανοίγματα προς τα κάτω για λόγους προστασίας από τη βροχή αφενός και κατά δεύτερο λόγο για να μην υπάρξει ανάγκη χρησιμοποίησης μεγαλύτερου μήκους καλωδίου από αυτό που έφερε ο feeder της Grid antenna.

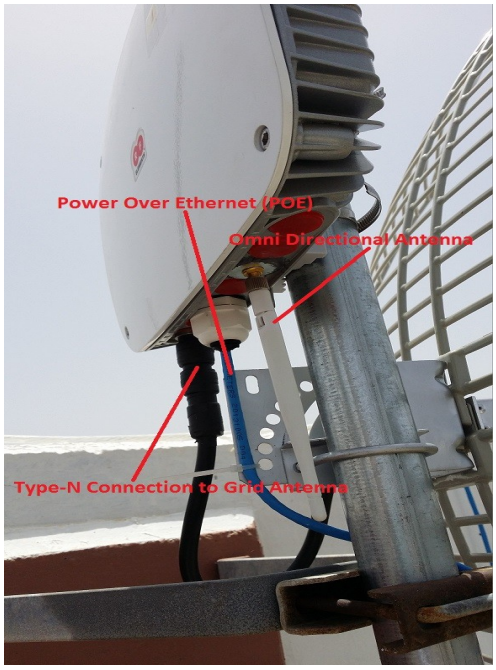


Figure 41: **Complete Node (figure1)**

Figure 42: **Complete Node (figure2)**

Figure 43: **Complete Node (figure3)**

Η αρχική ρύθμιση των κόμβων έγινε μέσω ενός Null Modem cable. Ως storage devices των boards χρησιμοποιήθηκαν Compact Flash Cards των 16 GB. Η σύνδεση των κόμβων με το backbone δίκτυο του Πολυτεχνείου έγινε μέσω Ethernet (μέσω της POE τροφοδοσίας) .

3.2 Λογισμικό

Λειτουργικό

Το λειτουργικό που χρησιμοποιήθηκε είναι το Voyage Linux 0.9.2. Είναι ένα λειτουργικό με βάση το Debian και είναι κατάλληλο για χρήση σε x86 embedded συστήματα όπως είναι και τα boards ALIX2D2 και ALIX3D2. [21]

Drivers

Τα network interfaces που χρησιμοποιήθηκαν χρησιμοποιούν Atheros drivers. Συγκεκριμένα το TL-WN360G που χρησιμοποιεί το chipset AR5007G χρησιμοποιεί τον driver Atheros ath5k. Το TL-WN861N που χρησιμοποιεί το chipset AR9223 χρησιμοποιεί τον driver Atheros ath9k.

Οι ath5k και ath9k drivers είναι open source drivers που προέρχονται από τον MadWifi driver. Ο MadWifi ήταν ο πιο διαδεδομένος driver για WLANs στα Linux αλλά ξεπεράστηκε κάποια στιγμή από τους ath5k και ath9k. Βασιζόταν στο Hardware Abstraction Layer (HAL) που δεν ήταν open source και που σκοπό είχε οι εφαρμογές που τρέχουν σε έναν κόμβο να χρησιμοποιούν το hardware χωρίς να τους απασχολεί ο τύπος του. Στους ath5k και ath9k έχει αντικατασταθεί αυτός ο closed source κώδικας με open source. Είναι συμβατοί με τις περισσότερες συσκευές και ιδικά με τις νέες που χρησιμοποιούν Atheros chipsets. Ένα σημαντικό πλεονέκτημα των ath5k και ath9k είναι ότι υποστηρίζουν το 802.11s standard για τα mesh networks που χρησιμοποιήθηκε και σε αυτή τη διπλωματική εργασία. Ο ath9k driver έχει την δυνατότητα εξυπηρέτησης interfaces που χρησιμοποιούν το 802.11n standard το οποίο επιτρέπει την χρήση πολλαπλών κεραιών με σκοπό την αύξηση του throughput που όπως φαίνεται και στο figure 1 μπορεί να φτάσει και τα 600Mbps.

Serial Access to Nodes

Για την αρχική ρύθμιση των κόμβων χρειαζόμαστε πρόσβαση στον κόμβο από την Serial θύρα μέσω του Null Modem cable. Στον υπολογιστή που συνδέθηκε με την serial θύρα κάθε κόμβου χρησιμοποιήθηκε η εντολή cu με την οποία μπορούμε να επικοινωνούμε μέσω serial terminal με τα boards.

SSH Access to Nodes

Αφού γίνουν οι αρχικές ρυθμίσεις των κόμβων και συγκεκριμένα των network interfaces ώστε να μπορούμε να έχουμε πρόσβαση στο backbone network του Πολυτεχνείου, μπορούμε να χρησιμοποιούμε Secure Socket Layer (SSH) σύνδεση για πρόσβαση στα boards. Στον υπολογιστή που συνδέεται στα boards εγκαταστάθηκε το openSSH client και στους κόμβους εγκαταστάθηκε το openSSH client και openSSH server. Έτσι μπορούμε να έχουμε πρόσβαση από τον υπολογιστή στους κόμβους αλλά και από κόμβο σε κόμβο.

Open80211s

Το open80211s είναι μια open source υλοποίηση του καθιερωμένου standard IEEE 802.11s για mesh networks, το οποίο είναι ακόμα υπό ανάπτυξη. Στόχος του είναι να δημιουργήσει την πρώτη open source υλοποίηση του 802.11s, να παρέχει πρόσβαση στην ανάπτυξη και την χρήση του 802.11s και να συμπτύξει τα διάφορα πρωτόκολλα για mesh δίκτυα σε ένα standard. Χρησιμοποιείται σε πλήθος συσκευών όπως PC, embedded Linux συστήματα, Android κλπ.

Βασίζεται στο mac80211 module του Linux το οποίο αναλαμβάνει όλη την διαχείριση των 802.11 frames σε software αντί για hardware. Αυτή η τεχνική ονομάζεται SoftMAC και επιτρέπει στις συσκευές να έχουν καλύτερο έλεγχο στο hardware τους και στους developers να μπορούν να κάνουν αναβαθμίσεις χωρίς να χρειάζεται να πειράζουν το firmware της κάρτας. Οι περισσότερες 802.11 συσκευές χρησιμοποιούν αυτή την τεχνική.

Χρησιμοποιήθηκε η open mesh έκδοση του open80211s κατά την οποία όλοι οι κόμβοι που έχουν σωστές ρυθμίσεις (matching Mesh IDs) μπορούν να συνδεθούν στο ίδιο mesh network. Αυτή είναι μια απλοϊκή προσέγγιση καθώς δεν περιλαμβάνει authentication ή frame encryption και η κίνηση του δικτύου είναι ορατή σε όλους τους κόμβους που συνδέονται.

Kernel

Το open80211s είναι μέρος του Linux kernel. Όλοι οι kernels που δημοσιεύθηκαν μετά τον Σεπτέμβρη του 2011 περιέχουν και την open mesh έκδοση του IEEE 802.11s. Ο kernel που χρησιμοποιήθηκε σε αυτή την διπλωματική εργασία είναι ο multichannel kernel του open80211s.

Το 802.11s standard παρέχει ένα σύνολο κανόνων για την συνεργασία των mesh κόμβων που λειτουργούν μέσα σε ένα mesh basic service set (MBSS). Το standard ορίζει ότι όλοι οι κόμβοι πρέπει να λειτουργούν στο ίδιο κανάλι. Η δυνατότητα του να χρησιμοποιούνται πολλαπλά κανάλια μέσα σε ένα δίκτυο κι έτσι κάθε link να μπορεί να έχει διαφορετικό κανάλι από τα υπόλοιπα, παρέχεται από αυτό τον kernel που αναπτύχθηκε στα πλαίσια του open80211s και χρησιμοποιήθηκε και σε αυτή την διπλωματική εργασία. Η χρήση διαφορετικών καναλιών στο ίδιο mesh δίκτυο έχει σαν αποτέλεσμα να μειώνεται ο ανταγωνισμός για την πρόσβαση στο μέσο μεταξύ των κόμβων. Ο kernel αναλαμβάνει να προωθεί πακέτα μεταξύ των κόμβων σε ένα τέτοιο δίκτυο με πολλαπλά κανάλια χρησιμοποιώντας το πρωτόκολλο δρομολόγησης HWMP για loop avoidance αντί ένα non-wireless πρωτόκολλο όπως το Spanning Tree Protocol (STP).

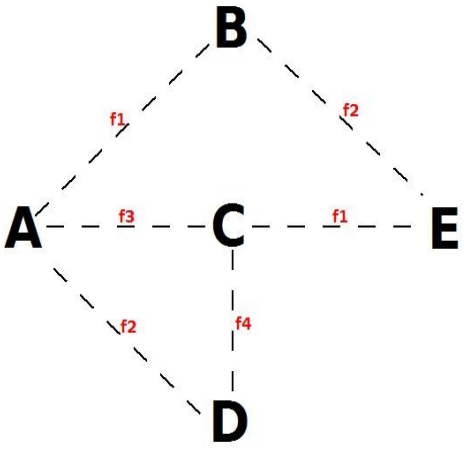


Figure 44: **Example of a Multichannel Mesh Network**

Channel Switching Mechanism

Το βασικό στοιχείο αυτής της διπλωματικής εργασίας ήταν η δημιουργία ενός μηχανισμού που θα επιτρέπει σε κάθε link να εντοπίζει πιθανές παρεμβολές και να αποφασίζει να αλλάζει στο κανάλι που έχει την καλύτερη ποιότητα για εκείνη την χρονική στιγμή.

Ένας κόμβος θα κάνει αίτηση σε έναν άλλον γειτονικό για να κάνει ένα Scan τα κανάλια ώστε να βρεθεί αυτό με το μικρότερο interference. Ο κόμβος που θα λάβει την αίτηση θα κάνει το scan, θα υπολογίσει ένα Interference Factor για κάθε κανάλι και θα διαλέξει αυτό με το μικρότερο. Στην συνέχεια ενημερώνει τον κόμβο που έκανε την αίτηση για το πιο κανάλι επιλέχθηκε και αλλάζουν και οι δύο κόμβοι το κανάλι του link τους ταυτόχρονα.

Η διαδικασία αλλαγής του καναλιού είναι άμεση χωρίς downtime του link. Η ανταλλαγή αυτών τον control μηνυμάτων γίνεται με TCP για να βεβαιωθούμε ότι τα πακέτα φτάνουν στον προορισμό τους. Αν χρησιμοποιούσαμε UDP τότε υπήρχε περίπτωση να αλλάζει ένας μόνο εκ των δύο κόμβων λόγω αποτυχίας αποστολής ενός control μηνύματος, με αποτέλεσμα να έχουμε fail του link μιας και το interface του ενός άκρου του link θα λειτουργεί σε διαφορετικό κανάλι από το άλλο.

Αυτή η διαδικασία μπορεί να επεκταθεί σε όλο το δίκτυο κι έτσι θα επιλέγεται το βέλτιστο κανάλι για κάθε link έπειτα από τα scans που εκτελούνται. Για κάθε link αναφερόμαστε στον κόμβο που κάνει την αίτηση για το scan ως Talker και στον κόμβο που εκτελεί το scan, παράγει τα Interference Factors των καναλιών και αποφασίζει το βέλτιστο κανάλι, ως Listener. Από την στιγμή που οι κόμβοι του δικτύου είναι στάσιμοι, ξέρουμε εκ των προτέρων ποια είναι τα πιθανά active links, έτσι μπορούμε να ορίσουμε εμείς σε κάθε link ποιος θα είναι ο Talker και ποιος ο Listener. Στην περίπτωση που έχουμε fail ενός link, απλά δεν θα γίνεται scan. Η δρομολόγηση θα ακολουθήσει κάποιο άλλο πιθανό active link στο οποίο ο μηχανισμός αυτός θα συνεχίσει να λειτουργεί και τα scans θα γίνονται επιτυχώς. Ο αλγόριθμος που χρησιμοποιείται για να παραχθούν τα Interference Factors των καναλιών σε κάθε scan είναι ο Survey Based Algorithm. Ο Survey Based Algorithm συλλέγει data για κάθε κανάλι και στη συνέχεια υπολογίζει ένα Interference Factor. Τα data που χρειάζονται για τον υπολογισμό του interference factor είναι:

Active Time: συνολικός χρόνος που μείναμε σε ένα κανάλι για μετρήσεις
Busy Time: χρόνος που το κανάλι έχει τόσο interference που καθιστά την μετάδοση αδύνατη

Το interference factor για κάθε κανάλι είναι ένας λόγος του busy time / active time.

Το active time, ο χρόνος δηλαδή που θα μείνει σε κάθε κανάλι για να συλλέξει data είναι 2 seconds. Έχουμε 13 channels οπότε το scanning θα πάρει 26 seconds.

Οι δύο τύποι mini PCI καρτών που χρησιμοποιήθηκαν (TL-WN360G και TL-WN861N) έχουν διαφορετική συμπεριφορά στην διαδικασία του Scanning λόγω του ότι χρησιμοποιούν διαφορετικούς drivers (ath5k και ath9k) αντίστοιχα. Έτσι χρειάστηκε να γραφτεί ξεχωριστός κώδικας για την διαδικασία του Scanning για κάθε μία κάρτα.

Έχει χρησιμοποιηθεί κώδικας σε C καθώς και script αρχεία. Ο κώδικας του Listener αρχικά περιμένει για μια TCP connection. Όταν δεχθεί μια TCP connection καλεί ένα αριθμό από scripts τα οποία με την σειρά τους καλούν κώδικες σε C που είναι υπεύθυνοι για την προσπέλαση όλων των καναλιών και την εξαγωγή των Interference Factor. Όταν επιλεγεί το βέλτιστο κανάλι τότε ο κώδικας του Listener αλλάζει ρόλο και γίνεται Talker. Ανοίγει μια TCP connection με τον αρχικό Talker και του στέλνει το νέο κανάλι. Τέλος ο κώδικας αναλαμβάνει να εκτελέσει τις απαραίτητες εντολές για την αλλαγή του interface στο νέο κανάλι. Το πιο interface θα αλλάξει κανάλι καθορίζεται από τα ορίσματα που δέχεται ο αρχικός Talker (interfaces και IPs των Talker και Listener).

Ο Talker δέχεται ως όρισμα τα interfaces και τις IPs των 2 κόμβων που θα εκτελέσουν το channel switch. Ανοίγει μια TCP connection με τον Listener και του στέλνει τις απαραίτητες πληροφορίες για το channel switch (τα ορίσματα για το interface και την IP του Listener που θέλουμε να εκτελέσουν το scan). Έπειτα αλλάζει ρόλο σε Listener και περιμένει τον αρχικό Listener να ανοίξει μια νέα TCP connection μαζί του με σκοπό να του στείλει το κανάλι στο οποίο θα αλλάξουν οι δύο κόμβοι. Όταν του στείλει το νέο κανάλι ο κώδικας αναλαμβάνει να εκτελέσει τις απαραίτητες εντολές ώστε το Interface του κόμβου να αλλάξει κανάλι.

Κεφάλαιο 4

Demo

Παρακάτω παρουσιάζεται η λειτουργία του λογισμικού που αναπτύχθηκε σε αυτή την διπλωματική εργασία. Το Demo έγινε σε περιβάλλον εργαστηρίου, και όχι από το δίκτυο που έχει στηθεί στο Πολυτεχνείο. Το λογισμικό που χρησιμοποιείται στο demo αυτό είναι το ίδιο με αυτό των κόμβων στο δίκτυο που στήθηκε στο Πολυτεχνείο, οπότε θα έχει ακριβώς την ίδια συμπεριφορά.

Έχουμε 3 κόμβους : A, B και C. Ο κόμβος A και C είναι ALIX3D2 και ο κόμβος B είναι ALIX2D2. Ο κόμβος A και C έχουν ένα network interface ο κάθε ένας που αποτελείται από μία TL-WN360G κάρτα δικτύου. Ο κόμβος B έχει δύο network interfaces που αποτελούνται από δύο TL-WN861N. Σε κάθε network interface έχει τοποθετηθεί μία omni directional κεραία μέσω ενός RP-SMA female to Pigtail U.FL male adaptor.

Αρχικά δημιουργούνται τα απαραίτητα mesh network interfaces σε κάθε κόμβο, ένα παράδειγμα παρουσιάζεται παρακάτω :

iw wlan0 interface add mesh0 type mp	: δημιουργείται το mesh interface mesh0
iw mesh0 set channel 1	: ορίζεται το κανάλι
ifconfig mesh0 10.10.10.1 up	: ορίζεται η IP του interface
iw mesh0 mesh join meshnet	: ορίζεται το mesh ID που θα μπει το interface

Όλα τα interfaces ορίζονται στο κανάλι ένα. Στα παρακάτω figures φαίνονται τα interfaces, οι IPs και οι MAC addresses των κόμβων με την εντολή ifconfig :

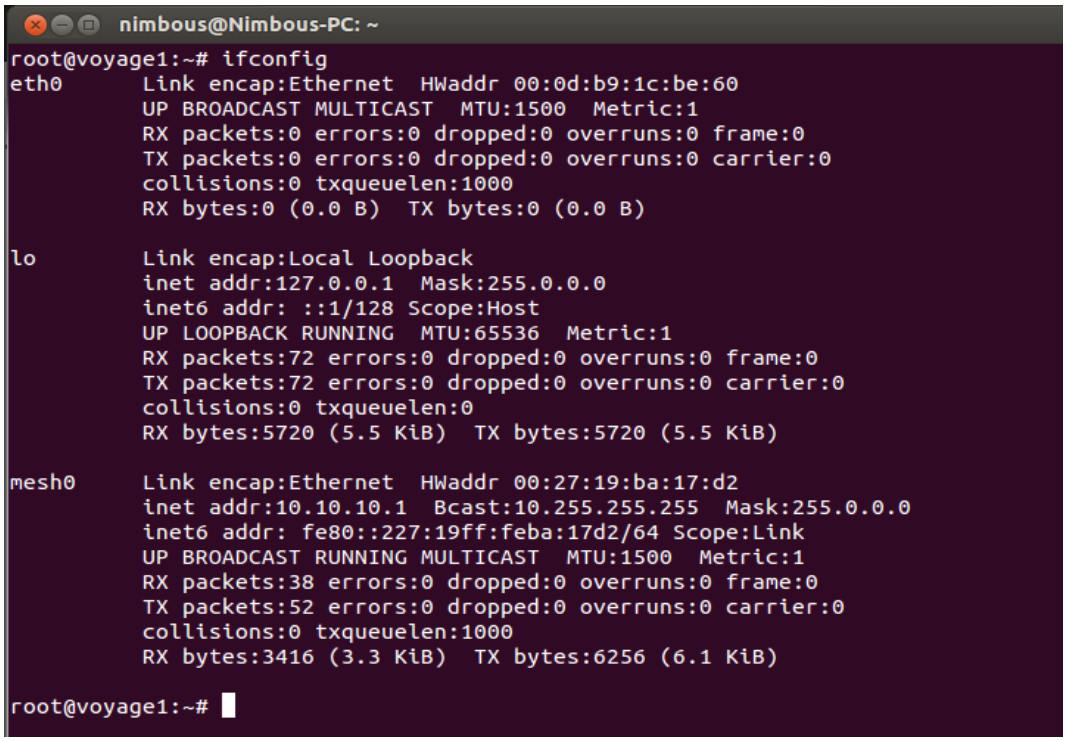


Figure 45: **Node A Interfaces**

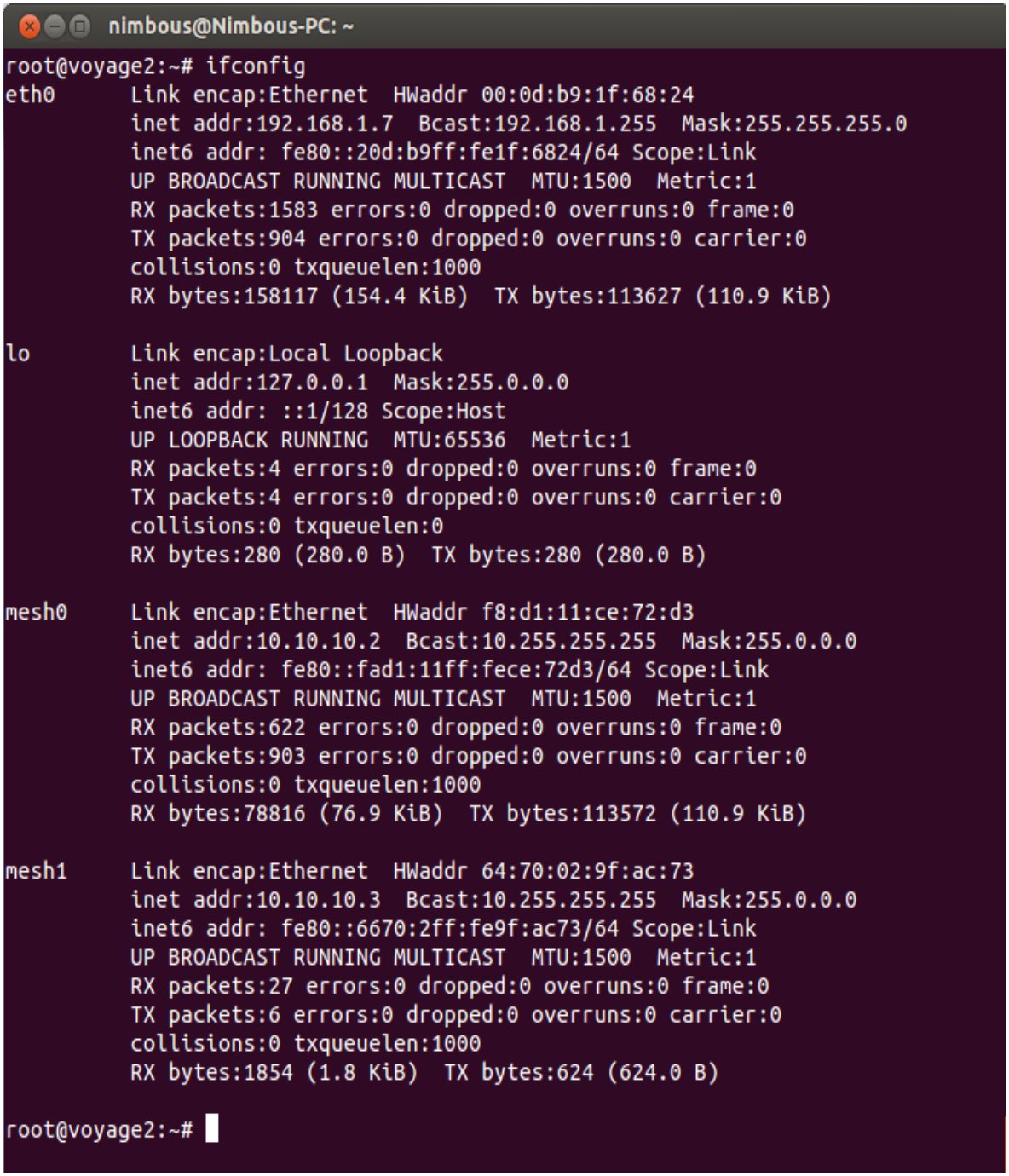


Figure 46: **Node B Interfaces**

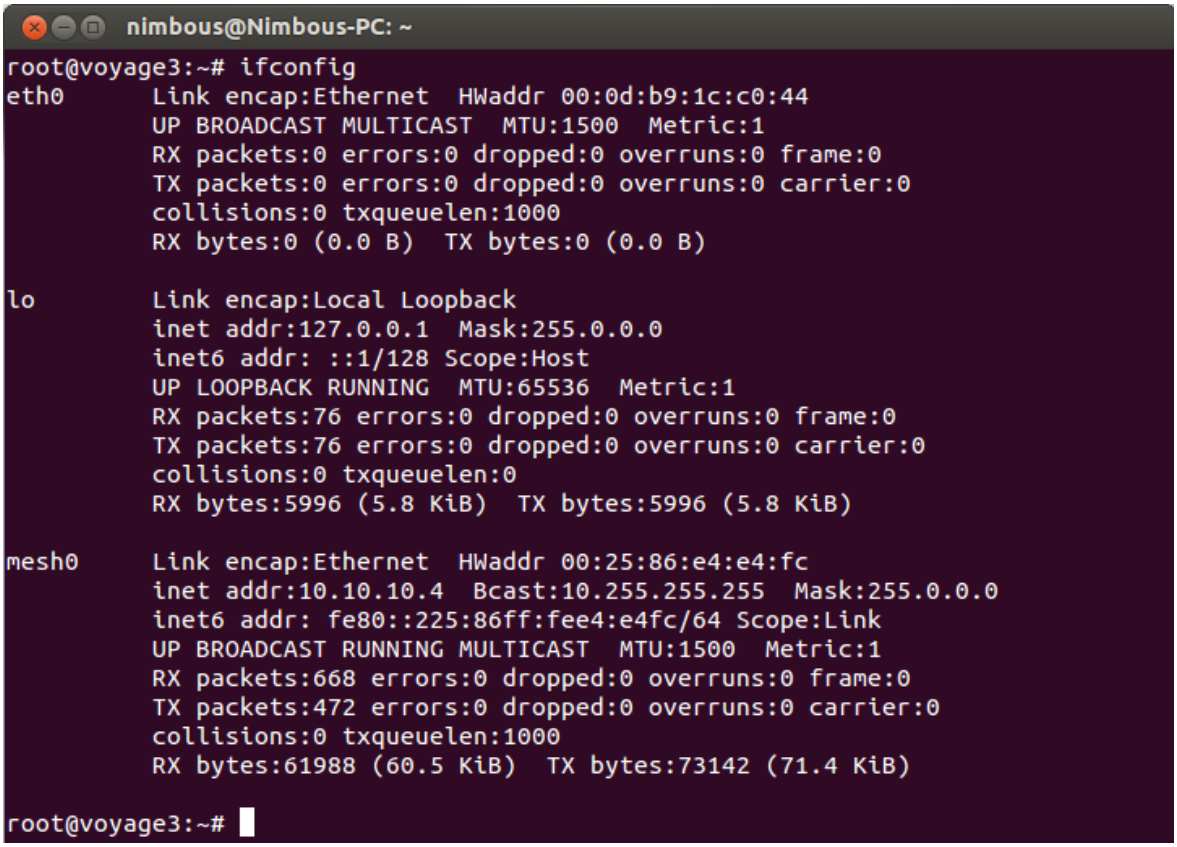


Figure 47: **Node C Interfaces**

Για τις ανάγκες του demo θα κάνουμε block τον κόμβο A από τον C έτσι ώστε να μπορούν να επικοινωνούν μόνο μέσω του B. Αυτό γίνεται κάνοντας block στα path tables κάθε κόμβου την αντίστοιχη MAC του άλλου κόμβου, έτσι :

Στον κόμβο A :
iw dev mesh0 station set 00:25:86:e4:e4:fc plink_action block

Στον κόμβο C:
iw dev mesh0 station set 00:27:19:ba:17:d2 plink_action block

Έτσι μπορούμε να δούμε στο path table του κόμβου A ότι δεν υπάρχει direct path στον κόμβο C.

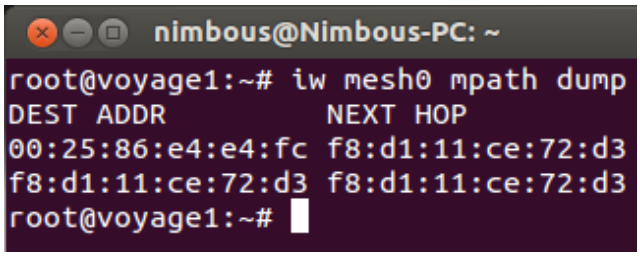


Figure 48: **Node A Path Table**

Στο επόμενο figure βλέπουμε ότι ο κόμβος A βρίσκει τον C (μέσω του B)

```
nimbous@Nimbous-PC: ~
root@voyage1:~# ping -c 5 10.10.10.4
PING 10.10.10.4 (10.10.10.4) 56(84) bytes of data.
64 bytes from 10.10.10.4: icmp_req=1 ttl=64 time=10.2 ms
64 bytes from 10.10.10.4: icmp_req=2 ttl=64 time=1.67 ms
64 bytes from 10.10.10.4: icmp_req=3 ttl=64 time=2.22 ms
64 bytes from 10.10.10.4: icmp_req=4 ttl=64 time=2.27 ms
64 bytes from 10.10.10.4: icmp_req=5 ttl=64 time=7.80 ms

--- 10.10.10.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4011ms
rtt min/avg/max/mdev = 1.671/4.847/10.257/3.511 ms
root@voyage1:~#
```

Figure 49: **Test Ping from Node A to C before Channel Switching**

Έτσι το δίκτυο μας έχει αυτή την μορφή αρχικά:

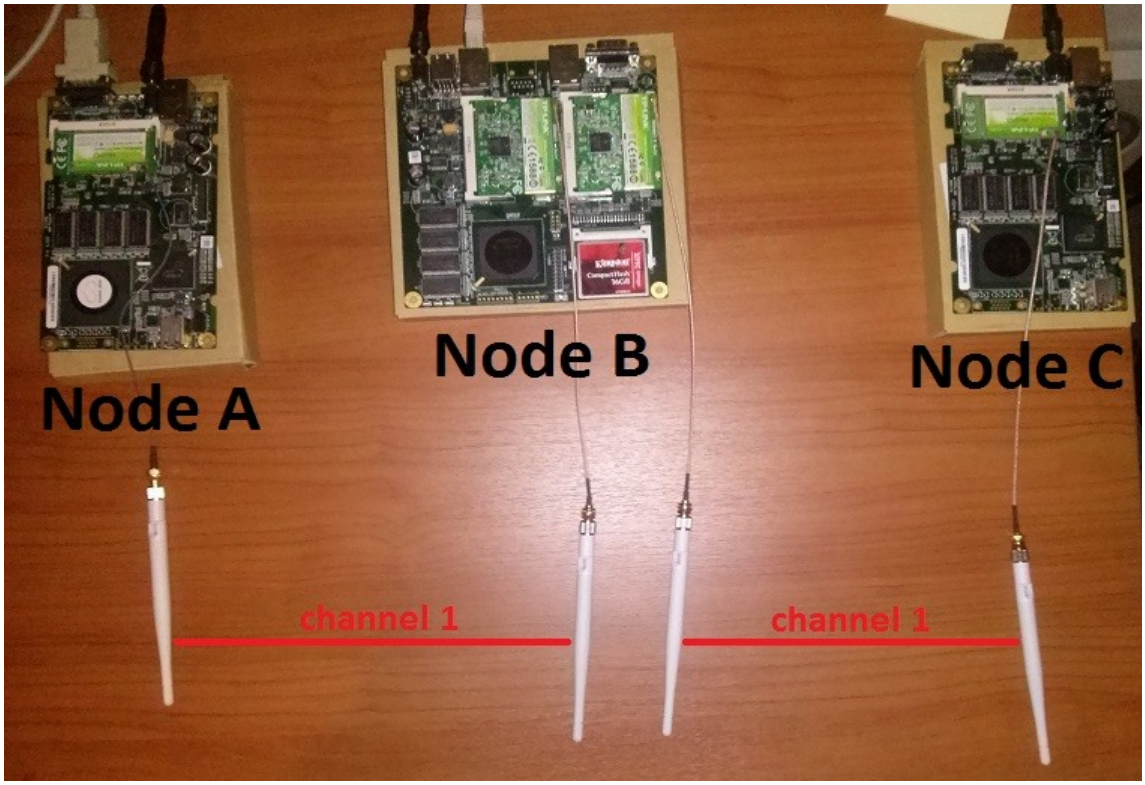


Figure 50: **Demo Network Initial State**

Ο multichannel kernel που χρησιμοποιούμε μας επιτρέπει να έχουμε στο ίδιο δίκτυο διαφορετικά κανάλια σε κάθε link. Για να το δείξουμε αυτό θα τρέξουμε τον μηχανισμό για channel selection στο link 1 ώστε να επιλεγεί το βέλτιστο κανάλι της δεδομένης χρονικής στιγμής. Αρχικά ο κόμβος Α θα αναλάβει τον ρόλο του Talker για το link A-B. Θα κάνει μια αίτηση για channel scan στον κόμβο Β ο οποίος έχει τον ρόλο του Listener. Το interface του κόμβου Β για το link A-B θα κάνει ένα scan τα κανάλια, θα υπολογίσει ένα Interference Factor για κάθε κανάλι, θα επιλέξει αυτό με το μικρότερο, θα ενημερώσει τον κόμβο Α και μετά θα κάνουν το channel switch και οι δύο ταυτόχρονα.

Στα επόμενα δύο figures παρουσιάζεται η λειτουργία του talker και του listener αντίστοιχα. Αρχικά ο talker δέχεται κάποια ορίσματα. Αυτά είναι με την σειρά :

1. Η IP του listener που θα κάνει το scan
2. Ένα debug μήνυμα για διευκόλυνση στον κώδικα (chswitch+)
3. Το interface του listener που θα κάνει το scan
4. Το interface του talker που θα ακολουθήσει το channel switch του listener
5. Και τέλος η IP του talker που θα ακολουθήσει το channel switch του listener

Αυτά τα ορίζουμε εμείς κατά την κλήση του talker καθώς όπως έχει ήδη αναφερθεί, γνωρίζουμε εκ των προτέρων τα πιθανά active links ενός rooftop network.

Στον talker εμφανίζεται ένα μήνυμα << change another link? >>, το αγνοούμε καθώς είναι εκτός του ενδιαφέροντος αυτού του demo. Παρέχει την δυνατότητα εναλλαγής και της συχνότητας δεύτερου link με την εμπλοκή και ενός τρίτου κόμβου. Αρχικά ο talker κάνει μια TCP connection με τον listener που περιμένει. Ο listener όταν επιλέξει κανάλι το στέλνει στον talker (client: received 8) και έπειτα ακολουθούν οι εντολές που εκτελεί ο talker για να αλλάξει το interface που έχουμε ορίσει εμείς από τα ορίσματα του talker.

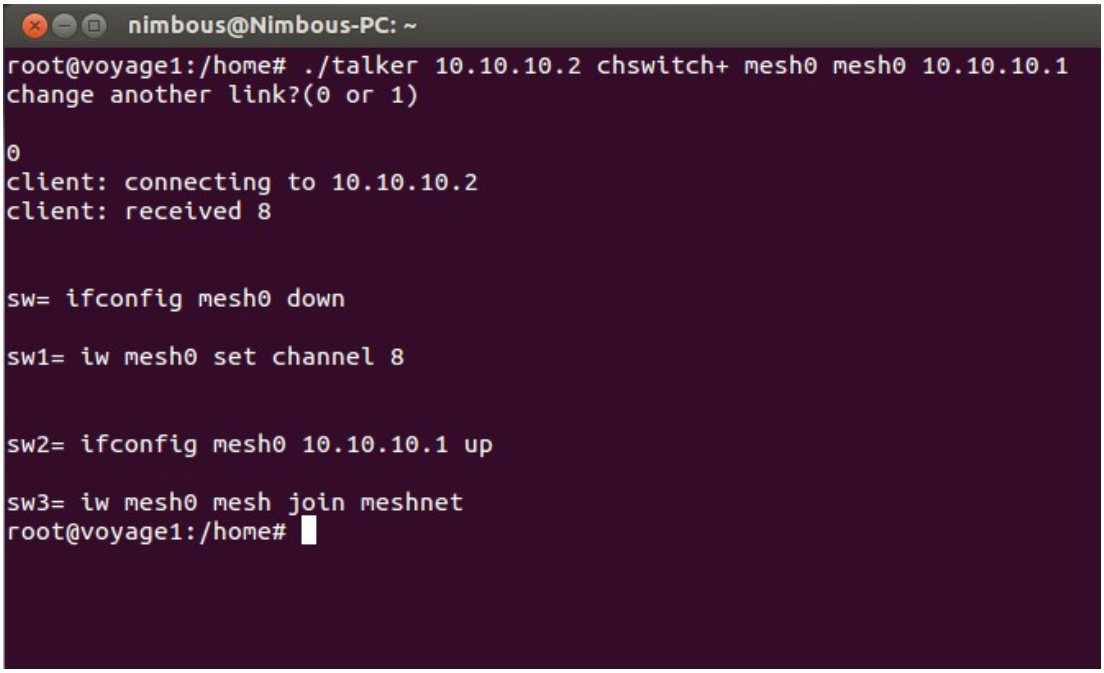


Figure 51: **Talker**

Από την άλλη ο listener περιμένει ένα TCP connection από τον talker (waiting for connections...). Όταν γίνει ένα connection τότε ανάλογα με τα ορίσματα που είχαν δοθεί στην κλίση του talker, ο listener θα αναθέσει σε ένα interface να κάνει το scan, στην συγκεκριμένη περίπτωση είναι στο interface mesh0 με IP 10.10.10.1. Αρχικά γίνεται ένα reset των προηγούμενων μετρήσεων (Reseting Survey Values) και έπειτα ακολουθεί το scan. Επιλέγεται το κανάλι με το μικρότερο Interference Factor το οποίο στην συγκεκριμένη περίπτωση είναι το κανάλι 8, το οποίο στέλνετε και στον talker (server: sending 8). Τέλος ακολουθούν οι εντολές που εκτελεί ο listener για να κάνει το channel switch.

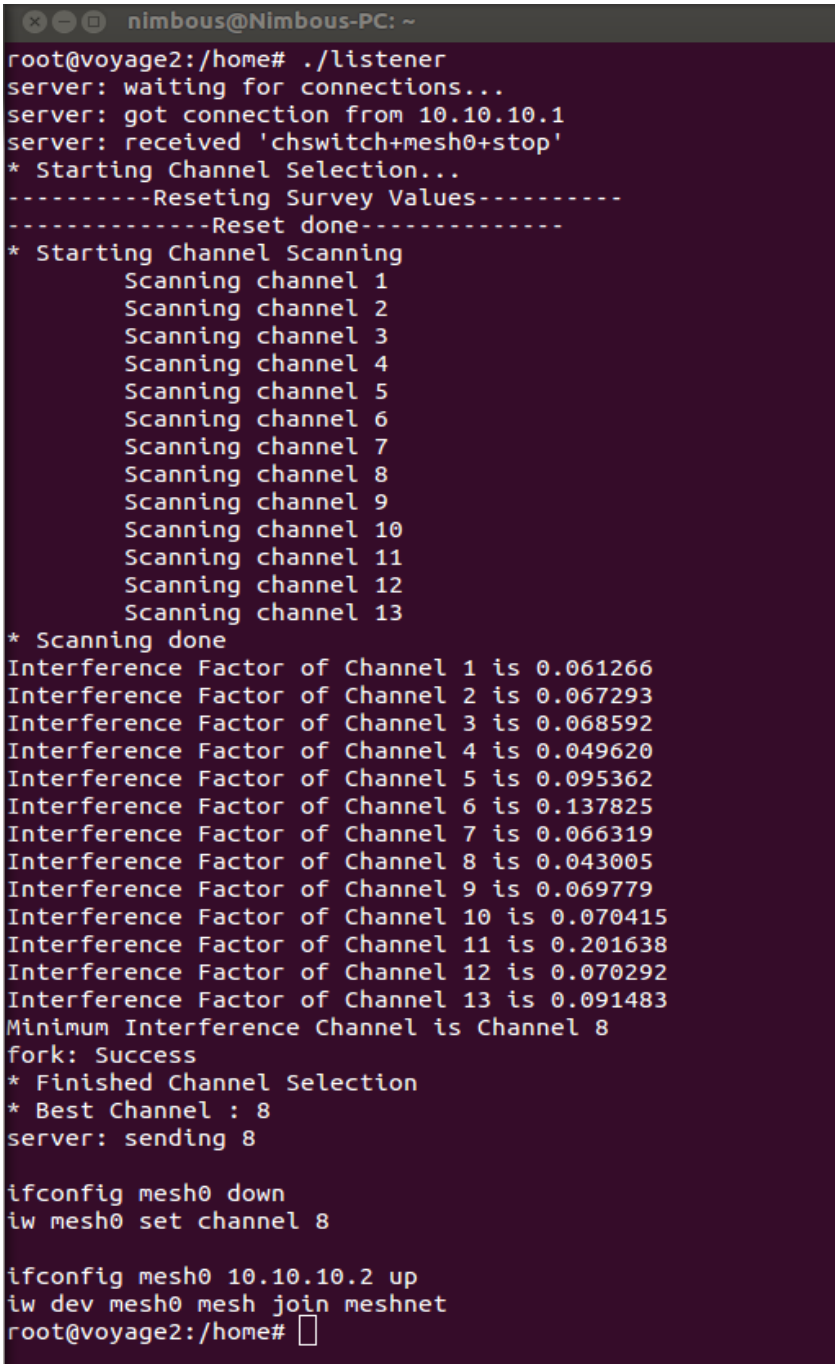


Figure 52: **Listener**

Μετά το channel switch το δίκτυο μας θα έχει αυτή την μορφή :

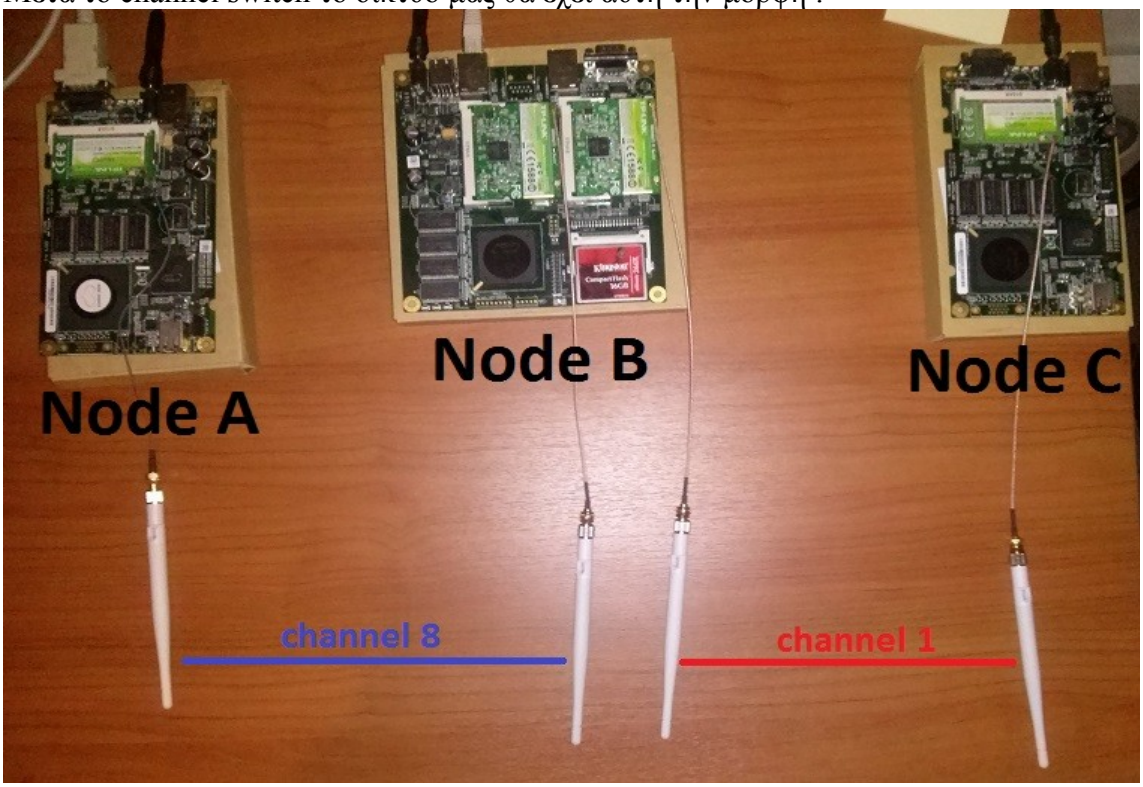


Figure 53: **Demo Network Final State**

Στο επόμενο figure βλέπουμε ότι ο κόμβος A μπορεί ακόμα να βρει τον C μέσω του B με την χρήση δύο καναλιών, καθώς και το path table του κόμβου A.

```
nimbous@Nimbous-PC: ~
root@voyage1:/home# ping -c 3 10.10.10.4
PING 10.10.10.4 (10.10.10.4) 56(84) bytes of data.
64 bytes from 10.10.10.4: icmp_req=1 ttl=64 time=17.2 ms
64 bytes from 10.10.10.4: icmp_req=2 ttl=64 time=1.74 ms
64 bytes from 10.10.10.4: icmp_req=3 ttl=64 time=1.84 ms

--- 10.10.10.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 1.741/6.953/17.269/7.294 ms
root@voyage1:/home# iw mesh0 mpath dump
DEST ADDR         NEXT HOP          IFACE      SN      METRIC
f8:d1:11:ce:72:d3 f8:d1:11:ce:72:d3 mesh0      0       683
64:70:02:9f:ac:73 f8:d1:11:ce:72:d3 mesh0      2       8193
00:25:86:e4:e4:fc f8:d1:11:ce:72:d3 mesh0      30      835
root@voyage1:/home#
```

Figure 54: **Demo Network testing and Node A Path Table**

Κεφάλαιο 5

Future Work

Το μεγαλύτερο πρόβλημα του μηχανισμού channel selection και channel switching που αναπτύχθηκε σε αυτή τη διπλωματική εργασία είναι ο χρόνος που χρειάζεται για να γίνει το scan των καναλιών. Έτσι σε ένα περιβάλλον όπου οι συνθήκες στα κανάλια αλλάζουν σχετικά γρήγορα, υπάρχει η περίπτωση να επιλέγουμε μη βέλτιστο κανάλι, καθώς οι μετρήσεις που θα παίρνουμε στο τέλος του scan θα είναι out-dated. Για παράδειγμα αν ενώ γίνονται scan τα τελευταία κανάλια, αρχίσει και επηρεάζει τα πρώτα έντονο interference, δεν θα μπορέσουμε να το εντοπίσουμε μέχρι το επόμενο scan. Έτσι μπορεί να επιλέξουμε κάποιο κανάλι που θεωρείται βέλτιστο αλλά δεν είναι λόγω out-dated πληροφορίας για την ποιότητα του.

Δεν μπορεί να ελαττωθεί ο χρόνος που παραμένουμε σε κάθε κανάλι για μετρήσεις καθώς μετά αρχίζουμε και έχουμε προβλήματα από την γρήγορη εναλλαγή καναλιών. Μερικά τέτοια προβλήματα που αντιμετωπίστηκαν είναι η αποτυχία στην εναλλαγή καναλιού, η αποτυχία λήψης σωστών μετρήσεων των καναλιών αλλά και διακοπή λειτουργίας του interface με μόνο τρόπο να επανέλθει το reset.

Ένα βασικό χαρακτηριστικό που πρέπει να συμπεριλαμβάνεται στον channel selection μηχανισμό είναι το να μπορούν οι κόμβοι να αντιλαμβάνονται πότε πρέπει να εκτελέσουν ένα scan. Αυτό θα μπορούσε να γίνεται με συλλογή στατιστικών ώστε να ξέρουμε πια χρονικά διαστήματα υπάρχουν οι περισσότερες μεταβολές στο Interference της συγκεκριμένης περιοχής και έτσι ο κόμβος να χρειάζεται να εκτελεί scans πιο τακτικά.

Ένα άλλο χαρακτηριστικό που πρέπει να προστεθεί στον channel selection μηχανισμό είναι μια διαδικασία επαναφοράς των κόμβων σε περίπτωση προβλήματος κατά την διάρκεια ενός channel switching. Ένα τέτοιο πρόβλημα θα μπορούσε να είναι για παράδειγμα η αποτυχία της TCP σύνδεσης για την ανταλλαγή των control μηνυμάτων. Σε αυτή την περίπτωση μπορεί κάποιος κόμβος να τεθεί εκτός δικτύου λόγω του ότι το interface του κατέληξε να χρησιμοποιεί διαφορετικό κανάλι από αυτό που είχε συμφωνήσει με τον κόμβο στην άλλη μεριά της ζεύξης. Οι κόμβοι θα πρέπει να αντιληφθούν το πρόβλημα και να επαναφέρουν τα interfaces τους σε ένα προσυμφωνημένο κανάλι. Η επιλογή αυτού του καναλιού θα μπορούσε να γίνει με βάση τα στατιστικά κάθε περιοχής που αναφέρθηκαν πριν. Έτσι ανάλογα με το πότε εμφανιστεί το πρόβλημα, θα επιλεγεί το κανάλι που είναι πιθανότερο να έχει το μικρότερο interference στο κάθε link.

Τέλος το open80211s είναι ακόμα υπό ανάπτυξη και η κοινότητα που το αποτελεί παράγει συνεχώς νέα χαρακτηριστικά δημοσιεύοντας νέους kernels ή κάνοντας update παλιούς. Θα μπορούσαν να γίνουν merge τέτοιοι kernels στο μέλλον ώστε να παρέχεται στο δίκτυο μεγαλύτερο εύρος δυνατοτήτων. Μια τέτοια δυνατότητα που μελετήθηκε αλλά τελικά δεν υλοποιήθηκε είναι η λειτουργία του power saving καθώς αντιμετωπιζόντουσαν κάποια προβλήματα συμβατότητας με το hardware που χρησιμοποιούσαμε ήδη με τον multichannel kernel.

Α. Παράρτημα: Διαδικασία ρύθμισης των κόμβων και παραδείγματα βασικών δικτυώσεων

Στα πλαίσια αυτής της διπλωματικής εργασίας αναπτύχθηκε και ένα tutorial για την εύκολη εισαγωγή στο embedded networking με χρήση των ALIX Boards. Σε αυτό το tutorial παρουσιάζεται η διαδικασία εγκατάστασης λειτουργικού σε ένα ALIX Board και οι ρυθμίσεις για την δημιουργία βασικών δικτυώσεων. Σε αυτό το tutorial έχουν χρησιμοποιηθεί USB κάρτες δικτύου αντί για τις mini PCI Που αναφέρθηκαν πριν.

Α.1 Σκοπός

Σκοπός αυτού του tutorial είναι να δείξει την διαδικασία που πρέπει να ακολουθηθεί ώστε να ρυθμιστούν τα boards για την υλοποίηση κάποιων τοπολογιών δικτύων. Το δίκτυα που θα υλοποιηθούν περιλαμβάνουν Access Points, ad-hoc nodes, mesh nodes κλπ, και σκοπός της υλοποίησης τους είναι να γίνει μια πρώτη επαφή και εξοικείωση με τα network interfaces των boards και το configuration τους.

Α.2 Προετοιμασία

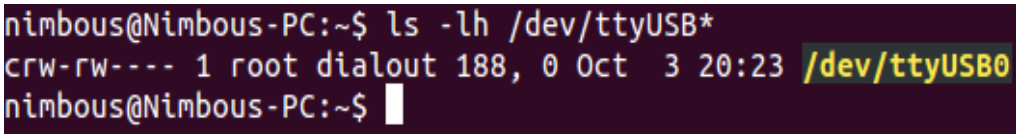
Η υλοποίηση των δικτύων έγινε χρησιμοποιώντας το λειτουργικό Voyage-Linux 0.9.2 για τα boards και Ubuntu 12.0.4 στο PC.
Για τα δίκτυα θα χρειαστούμε συνολικά:
3 ALIX boards (2d2 ή 3d2)
6 usb wifi cards (tp-link και level one)
3 compact flash memories
1 RS-232 null modem cable
1 RS-232 serial to usb adaptor
3 usb extenders
1 ethernet cable

Α.3 Σύνδεση PC-Board

Συνδέουμε το RS-232 null modem cable με το usb adaptor και αυτό με το PC. Ελέγχουμε αν αναγνωρίζεται χρησιμοποιώντας την εντολή:

sudo ls -lh /dev/ttyUSB*

πρέπει να εμφανιστεί κάτι τέτοιο:



Στην συγκεκριμένη περίπτωση το RS-232 null modem cable με τον usb to serial adaptor είναι στο /dev/ttyUSB0.

Για να χρησιμοποιήσουμε το καλώδιο πρέπει πρώτα να εισάγουμε έναν χρήστη στο dialout group. Χρησιμοποιούμε την εντολή:

sudo adduser my_username dialout

όπου my_username το όνομα του user με το οποίο θα κάνουμε connect στο board.

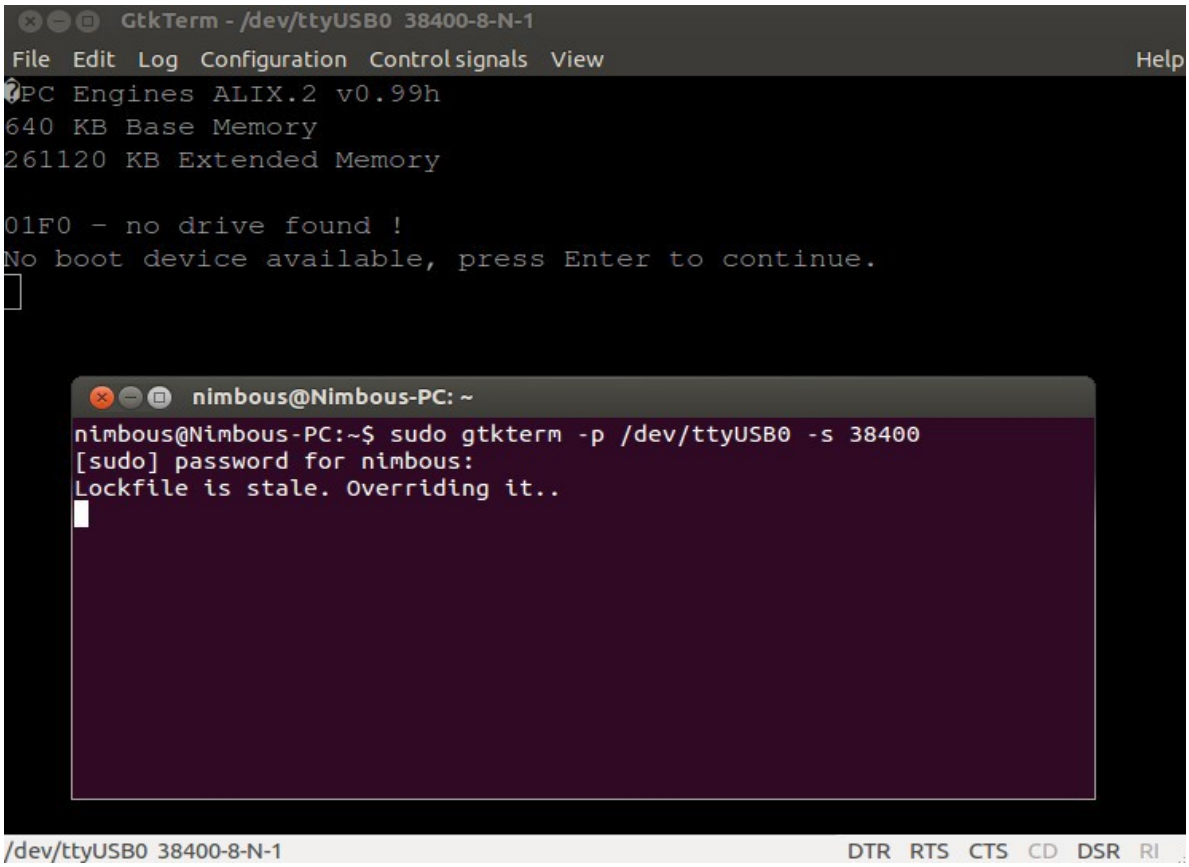
Κατεβάζουμε το gtkterm,που είναι ένα serial terminal, χρησιμοποιώντας την εντολή:

sudo apt-get install gtkterm

Συνδέουμε το RS-232 null modem με το ALIX board. Συνδεόμαστε στο RS-232 null modem cable χρησιμοποιώντας την εντολή:

sudo gtkterm -p /dev/ttyUSB0 -s 38400

Ανοίγουμε την τροφοδοσία και πρέπει να δούμε κάτι τέτοιο:



Μερικές φορές θα εμφανίζεται το μήνυμα Lockfile is stale, το αγνοούμε.

Κλείνουμε το board χρησιμοποιώντας την εντολή:

shutdown now

και μόλις ολοκληρωθεί η διαδικασία, βγάζουμε την τροφοδοσία.

Α.4 Εγκατάσταση και ρύθμιση του Voyage-Linux 0.9.1

Συνδέουμε την CF κάρτα στο CF Reader και στο PC και ανοίγουμε ένα terminal. Χρησιμοποιούμε την εντολή:

sudo mkdir /tmp/cf

μετά θα πρέπει να βρούμε πιο device είναι η CF κάρτα μας.

sudo fdisk -l

```
nimbous@Nimbous-PC:~$ sudo fdisk -l

Disk /dev/sda: 64.0 GB, 64023257088 bytes
255 heads, 63 sectors/track, 7783 cylinders, total 125045424 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x18474e39

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1            2048      206847      102400    7  HPFS/NTFS/exFAT
/dev/sda2   *      206848   125042687    62417920    7  HPFS/NTFS/exFAT

Disk /dev/sdb: 2000.4 GB, 2000398934016 bytes
255 heads, 63 sectors/track, 243201 cylinders, total 3907029168 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk identifier: 0x18474e12

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048   204802047   102400000    5  Extended
/dev/sdb2       204802048  3907028991  1851113472    7  HPFS/NTFS/exFAT
/dev/sdb5            4096    61444095    30720000   83  Linux
/dev/sdb6        61446144    65542143     2048000   82  Linux swap / Solaris
/dev/sdb7        65544192   204802047    69628928   83  Linux

Disk /dev/sdc: 4009 MB, 4009549824 bytes
124 heads, 62 sectors/track, 1018 cylinders, total 7831152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0001926a

   Device Boot      Start         End      Blocks   Id  System
nimbous@Nimbous-PC:~$
```

Όπως βλέπουμε παραπάνω, η CF κάρτα μας είναι το /dev/sdc.

Αρχικά θα σβήσουμε τυχών partitions που υπάρχουν στην κάρτα. Κατεβάζουμε το GParted από το Ubuntu Command Center και επιλέγουμε το device sdc.

Πρέπει όλος ο χώρος να είναι unallocated. Αν υπάρχουν partitions τα σβήνουμε.

Μετά χρησιμοποιούμε την εντολή:

sudo fdisk /dev/sdc

και πρέπει να κάνουμε τις παρακάτω επιλογές:

```
nimbous@Nimbous-PC:~$ sudo fdisk /dev/sdc

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-7831151, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-7831151, default 7831151):
Using default value 7831151

Command (m for help): a
Partition number (1-4): 1

Command (m for help): p

Disk /dev/sdc: 4009 MB, 4009549824 bytes
124 heads, 62 sectors/track, 1018 cylinders, total 7831152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0001926a

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1   *        2048     7831151     3914552   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
nimbous@Nimbous-PC:~$
```

Μετά χρησιμοποιούμε την εντολή:

sudo mkfs.ext2 /dev/sdc1

και πρέπει να κάνουμε τις παρακάτω επιλογές:

```
nimbous@Nimbous-PC:~$ sudo mkfs.ext2 /dev/sdc1
mke2fs 1.42 (29-Nov-2011)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
244800 inodes, 978638 blocks
48931 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1002438656
30 block groups
32768 blocks per group, 32768 fragments per group
8160 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Writing superblocks and filesystem accounting information: done

nimbous@Nimbous-PC:~$
```

Μετά χρησιμοποιούμε την εντολή:

sudo tune2fs -c 0 /dev/sdc1

```
nimbous@Nimbous-PC:~$ sudo tune2fs -c 0 /dev/sdc1
tune2fs 1.42 (29-Nov-2011)
Setting maximal mount count to -1
nimbous@Nimbous-PC:~$
```

Μετά πρέπει να κατεβάσουμε το λειτουργικό μας (voyage-linux 0.9.2). Χρησιμοποιούμε τις εντολές:

cd /home/username/Downloads , όπου username το username του PC μας (πχ. Nimbous)

sudo wget http://www.voyage.hk/download/voyage/voyage-0.9.2.tar.bz2

sudo tar --numeric-owner -jxf voyage-0.9.2.tar.bz2

cd voyage-0.9.2

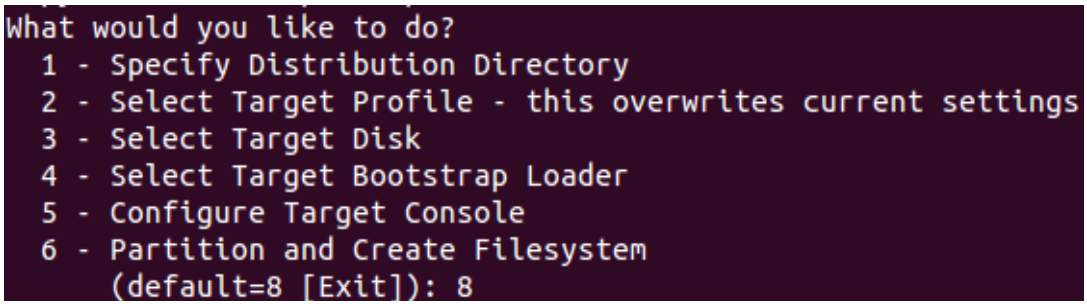
sudo ./usr/local/sbin/voyage.update

Και κάνουμε τις εξής επιλογές:

```
nimbous@Nimbous-PC:~/Downloads/voyage-0.9.1$ sudo ./usr/local/sbin/voyage.update
What would you like to do?
  1 - Create new Voyage Linux disk
  2 - Update existing Voyage configuration
  3 - Exit
      (default=1 [Create new Voyage Linux disk]): 1

some mandatory options are unset, please enter them interactively
Where is the Voyage Linux distribution directory?
  (default=/home/nimbous/Downloads/voyage-0.9.1):

What would you like to do?
  1 - Specify Distribution Directory
  2 - Select Target Profile - this overwrites current settings
  3 - Select Target Disk
  4 - Select Target Bootstrap Loader
  5 - Configure Target Console
  6 - Partition and Create Filesystem
      (default=2 [Select Target Profile - this overwrites current settings]): 2

Please select Voyage profile:
  1 - Keep existing settings
  2 - 4501
  3 - 4511/4521
  4 - 4801
  5 - 5501
  6 - 6501
  7 - ALIX
  8 - Generic PC
  9 - Notebook (pcmcia)
  10 - WRAP
      (default=7 [ALIX]): 7

What would you like to do?
  1 - Specify Distribution Directory
  2 - Select Target Profile - this overwrites current settings
  3 - Select Target Disk
  4 - Select Target Bootstrap Loader
  5 - Configure Target Console
  6 - Partition and Create Filesystem
      (default=3 [Select Target Disk]): 3
```

```
Partitions information
major minor  #blocks  name

   8        0   62522712 sda
   8        1     102400 sda1
   8        2   62417920 sda2
   8       16 1953514584 sdb
   8       17          1 sdb1
   8       18 1851113472 sdb2
   8       21   30720000 sdb5
   8       22    2048000 sdb6
   8       23   69628928 sdb7
  11        0    1048575 sr0
   8       32    3915576 sdc
   8       33    3914552 sdc1

Which device accesses the target disk [/dev/hde]? /dev/sdc

Which partition should I use on /dev/sdc for the Voyage system [1]? 1

Device information for /dev/sdc1
device               fs_type   label      mount point              UUID
-------------------------------------------------------------------------------
/dev/sdc1            ext2                 (not mounted)            b5615cd4-bd6

Where can I mount the target disk [/tmp/cf]? /tmp/cf

What would you like to do?
  1 - Specify Distribution Directory
  2 - Select Target Profile - this overwrites current settings
  3 - Select Target Disk
  4 - Select Target Bootstrap Loader
  5 - Configure Target Console
  6 - Partition and Create Filesystem
      (default=4 [Select Target Bootstrap Loader]): 4

Which loader do you want (grub or lilo) [grub]? grub

Which partition is used for bootstrap [1]? 1

What would you like to do?
  1 - Specify Distribution Directory
  2 - Select Target Profile - this overwrites current settings
  3 - Select Target Disk
  4 - Select Target Bootstrap Loader
  5 - Configure Target Console
  6 - Partition and Create Filesystem
      (default=5 [Configure Target Console]): 5

Select terminal type:
  1 - Serial Terminal
  2 - Console Interface
      (default=1 [Serial Terminal]): 1
```

```
Please choose speed:
  1 - 2400
  2 - 4800
  3 - 9600
  4 - 19200
  5 - 38400
  6 - 57600
  7 - 115200
      (default=5 [38400]): 5

What would you like to do?
  1 - Specify Distribution Directory
  2 - Select Target Profile - this overwrites current settings
  3 - Select Target Disk
  4 - Select Target Bootstrap Loader
  5 - Configure Target Console
  6 - Partition and Create Filesystem
      (default=6 [Partition and Create Filesystem]): 6

What shall I do with your Flash Media?
  1 - Partition Flash Media and Create Filesystem
  2 - Use Flash Media as-is
      (default=1 [Partition Flash Media and Create Filesystem]): 1

What would you like to do?
  1 - Specify Distribution Directory
  2 - Select Target Profile - this overwrites current settings
  3 - Select Target Disk
  4 - Select Target Bootstrap Loader
  5 - Configure Target Console
  6 - Partition and Create Filesystem
      (default=7 [Copy Distribution to Target]): 7


Configuration details:
----------------------

Distribution directory:   /home/nimbous/Downloads/voyage-0.9.1

Disk/Flash Device:        /dev/sdc
Installation Partition:   /dev/sdc1
Create Partition and FS:  yes
Bootstrap Partition:      /dev/sdc1

Will be mounted on:       /tmp/cf

Target system profile:    ALIX
Target console:           serial
Target baud rate:         38400

Bootstrap installer:      grub
Bootstrap partition:      /dev/sdc1

OK to continue (y/n)? y
```



Βάζουμε την CF κάρτα στο board, συνδέουμε το RS-232 null modem cable και το Ethernet.

Σημείωση: Στις default ρυθμίσεις των interfaces του Voyage-Linux, το interface eth0 (Ethernet) είναι ήδη ρυθμισμένο για να εντοπίζει δίκτυο. Οπότε αν συνδέσουμε εμείς το board με το router με Ethernet θα βρει κατευθείαν το δίκτυο και θα μπει. Χρησιμοποιούμε την εντολή:

sudo gtkterm -p /dev/ttyUSB0 -s 38400

Ανοίγουμε την τροφοδοσία και πρέπει να δούμε τα Voyage-Linux να κάνουν boot. Κάνουμε login με username:root και password:voyage

Χρησιμοποιούμε την εντολή: remountrw, την οποία θα πρέπει να την χρησιμοποιούμε μετά από καθε boot. Χρησιμοποιούμε την εντολή:

passwd

και επιλέγουμε νέο password για το root.

Τέλος κάνουμε update

sudo apt-get update

Για να κλείσουμε το board χρησιμοποιούμε την εντολή:

shutdown now

και αφού τελειώσει η διαδικασία, βγάζουμε την τροφοδοσία.

Α.5 Υλοποίηση των δικτύων

Αρχικά θα αλλάξουμε hostname. Αυτό θα είναι ιδιαίτερα χρήσιμο αργότερα που θα έχουμε πολλά boards με ssh σύνδεση. Για αυτό το παράδειγμα θα χρησιμοποιήσουμε hostnames voyage1, voyage2, ... στα boards μας. Για να αλλάξουμε hostname, γράφουμε το hostname που θέλουμε (voyage1) στο αρχείο /etc/hostname, χρησιμοποιώντας την εντολή:

vi /etc/hostname

Μετά θα στήσουμε έναν ssh server στο board (και έναν ssh client που θα χρειαστεί αργότερα) για να μπορούμε να το ελέγχουμε χωρίς το serial cable. Χρησιμοποιούμε τις εντολές :

sudo apt-get install openssh-server openssh-client

sudo /etc/init.d/ssh start

και αντίστοιχα stop ή restart αν θέλουμε να σταματήσουμε η να επανεκινήσουμε τον ssh server.

Για να συνδεθούμε με ssh στο board θέλουμε κάποιον ssh client στο PC. Χρησιμοποίησα το Putty SSH Client το οποίο μπορούμε να το βρούμε και να το κατεβάσουμε από το Ubuntu Software Center.

Πριν συνδεθούμε με ssh στο board πρέπει να γνωρίζουμε την IP που έχει πάρει το board από την wired σύνδεση με Ethernet με τον router. Για να την βρούμε χρησιμοποιούμε την εντολή:

ifconfig

Βλέπουμε την IP του interface eth0 και ανοίγουμε το Putty SSH Client. Ως IP βάζουμε την IP του Ethernet interface του board και επιλέγουμε SSH στο port 22, μετά Open. Κάνουμε Accept το πιστοποιητικό και κάνουμε login με το user name και το password του χρήστη του board που θέλουμε να συνδεθούμε. Μπορούμε τώρα να αποσυνδέσουμε το RS-232 serial cable και να δουλεύουμε με ssh μέσω Ethernet.

Επόμενο βήμα είναι να εγκαταστήσουμε το απαραίτητο λογισμικό για τις usb wifi cards. Για τις level one χρησιμοποιούμε την εντολή:

sudo apt-get install firmware-ralink

Για τις tp-link χρησιμοποιούμε τις εντολές:

cd /lib/firmware

sudo wget wireless.kernel.org/download/htc_fw/1.3/htc_9271.fw

Κάνουμε reboot χρησιμοποιώντας την εντολή:

reboot

Όταν ανοίξει το board ξανά, θα πρέπει να αναγνωρίζει τις tp-link και τις level one usb wifi κάρτες και να δουλεύουν κανονικά. Για να το ελέγξουμε αυτό θα προσπαθήσουμε να συνδεθούμε στο δίκτυο του σπιτιού μας με αυτές.

Θα ρυθμίσουμε ένα interface στο board να συνδέεται με wifi στο δίκτυο του σπιτιού μας και θα δοκιμάσουμε και τις usb wifi κάρτες.

Χρησιμοποιούμε την εντολή:

vi /etc/network/interfaces

Το αρχείο που άνοιξε έχει πολλά scripts σε σχόλια Αυτά καλό θα ήταν να μην τα πειράξουμε, αλλά να τα αφήσουμε και να τα χρησιμοποιούμε ως παράδειγμα για τυχών interfaces που θα θέλουμε να ρυθμίσουμε στο μέλλον. Ότι θέλουμε να κάνουμε εμείς, απλά το προσθέτουμε. Έτσι για την συγκεκριμένη περίπτωση θα χρειαστεί να προσθέσουμε αυτό το κομμάτι στο αρχείο interfaces :

```
#connect to home network requesting ip from router dhcp
auto wlan0
iface wlan0 inet dhcp
wpa-ssid [redacted]
wpa-psk [redacted]
```

όπου wpa-ssid το όνομα του δικτύου μας και wpa-psk το pre-shared key του δικτύου μας.

Σημείωση: Στο δίκτυο μου χρησιμοποιώ WPA2-PSK authentication type. Οι ρυθμίσεις για WPA2- PSK και WPA-PSK είναι ίδιες. Σε όλο το tutorial υποθέτουμε ότι χρησιμοποιούμε ένα από τα δύο προαναφερθέντα authentication types.

Συνδέουμε την tp-link usb wifi κάρτα σε έναν usb extender και στο board. Χρησιμοποιούμε usb extender καθώς στα ALIX 2d2 boards, δεν μπορούμε να βάλουμε την tp-link κατευθίαν στο usb γιατί μας εμποδίζει το καλώδιο τροφοδοσίας. Στα ALIX 3d2 boards, οι θύρες usb είναι πολύ κοντά η μία στην άλλη κι έτσι δεν μπορούμε να βάλουμε 2 usb wifi κάρτες. Οπότε αναγκαστικά αν θέλουμε σε ένα board να έχουμε 2 usb wifi κάρτες, θα πρέπει την μία να την συνδέουμε με usb extender πρώτα.

Κάνουμε restart το network του board χρησιμοποιώντας την εντολή:

/etc/init.d/networking restart

Όταν τελειώσει το restart του network χρησιμοποιούμε την εντολή:

ifconfig

και πρέπει να δούμε το interface wlan0 να έχει πάρει IP από το router μας.

Βγάζουμε την tp-link και βάζουμε την level one. Κάνουμε πάλι restart το network:

/etc/init.d/networking restart

και χρησιμοποιούμε πάλι την εντολή:

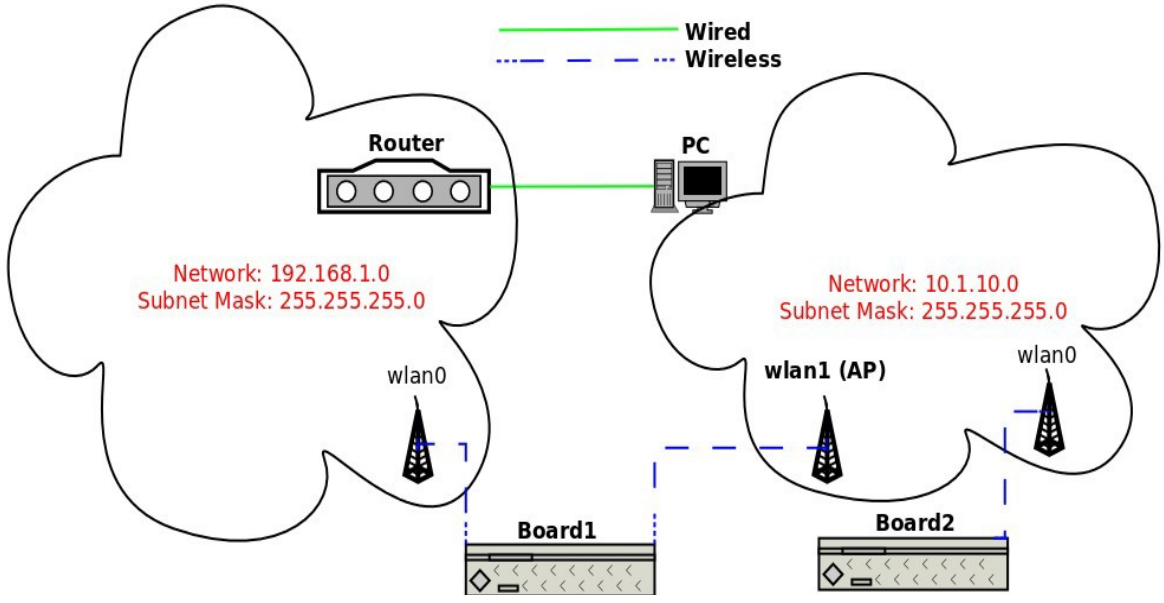
ifconfig

πάλι πρέπει να δούμε το interface wlan0 να έχει πάρει IP από το router μας.

Αφού βεβαιωθήκαμε πως οι usb wifi κάρτες δουλεύουν, θα προχωρήσουμε στην υλοποίηση των δικτύων.

Την διαδικασία που περιγράφτηκε ως τώρα, ακολουθούμε για κάθε board που θέλουμε να δουλέψουμε.

A.5.1 Access Point



Το interface που μόλις ρυθμίσαμε στο πρώτο board θα μας χρειαστεί και τώρα. Πρέπει να συνδέσουμε στο board άλλη μια usb wifi κάρτα η οποία θα χρησιμεύσει ως Access Point (AP). Όπως και πριν, θα προσθέσουμε στο αρχείο interfaces με την εντολή:

vi /etc/network/interfaces

Αυτό το κομμάτι:

```
#create a AP
auto wlan1
iface wlan1 inet static
address 10.1.10.1
netmask 255.255.255.0
broadcast 10.1.10.255
hostapd /etc/hostapd/hostapd.wlan1.conf
up nat.sh wlan1 wlan0 "10.1.10.2/24"
```

Μετά πρέπει να δημιουργήσουμε το αρχείο hostapd.wlan1.conf το οποίο θα περιλαμβάνει τις ρυθμίσεις του AP μας.

Χρησιμοποιούμε την εντολή:

vi /etc/hostapd/hostapd.wlan1.conf

και προσθέτουμε αυτό το κομμάτι:

```
interface=wlan1
driver=nl80211
logger_syslog=-1
logger_syslog_level=2
logger_stdout=-1
logger_stdout_level=2
debug=4
#dump_file=/tmp/hostapd.dump
#ctrl_interface=/var/run/hostapd
#ctrl_interface_group=0
channel=6
hw_mode=g
macaddr_acl=0
auth_algs=3
eapol_key_index_workaround=0
eap_server=0
wpa=3
ssid=voyage1-AP-wpa
wpa_passphrase=[redacted]
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
eapol_version=1
```

όπου wpa_passphrase είναι το pre-shared key του δικτύου. Μπορούμε τώρα να κάνουμε restart το network χρησιμοποιώντας την εντολή:

/etc/init.d/networking restart

και όταν γίνει θα δούμε, το wifi δίκτυο voyage1-AP-wpa.

Μπορούμε να κάνουμε connect χρησιμοποιώντας το wpa_passphrase που ορίσαμε και να έχουμε πρόσβαση στο Internet.

Τελειώσαμε με τις ρυθμίσεις του πρώτου board.

Για την ρύθμιση του wlan0 στο δεύτερο board χρησιμοποιούμε την εντολή:

vi /etc/network/interfaces

και προσθέτουμε αυτό το κομμάτι:

```
#connect to voyage1 AP
auto wlan0
iface wlan0 inet dhcp
wpa-driver wext
wpa-ssid voyage1-AP-wpa
wpa-psk [redacted]
wpa-key-mgmt WPA-PSK
wpa-pairwise TKIP
wpa-group CCMP TKIP
wpa-proto RSN
wireless-mode Managed
```

όπου το wpa-psk είναι το pre-shared key που έχουμε ορίσει στο AP του wlan1 στο board1.

Αν κάνουμε restart το network χρησιμοποιώντας την εντολή:

/etc/init.d/networking restart

και μετά κάνουμε

ifconfig

τότε θα δούμε ότι το wlan0 έχει πάρει IP από το AP του board 1 και άρα έχει μπει επιτυχώς στο δίκτυο του.

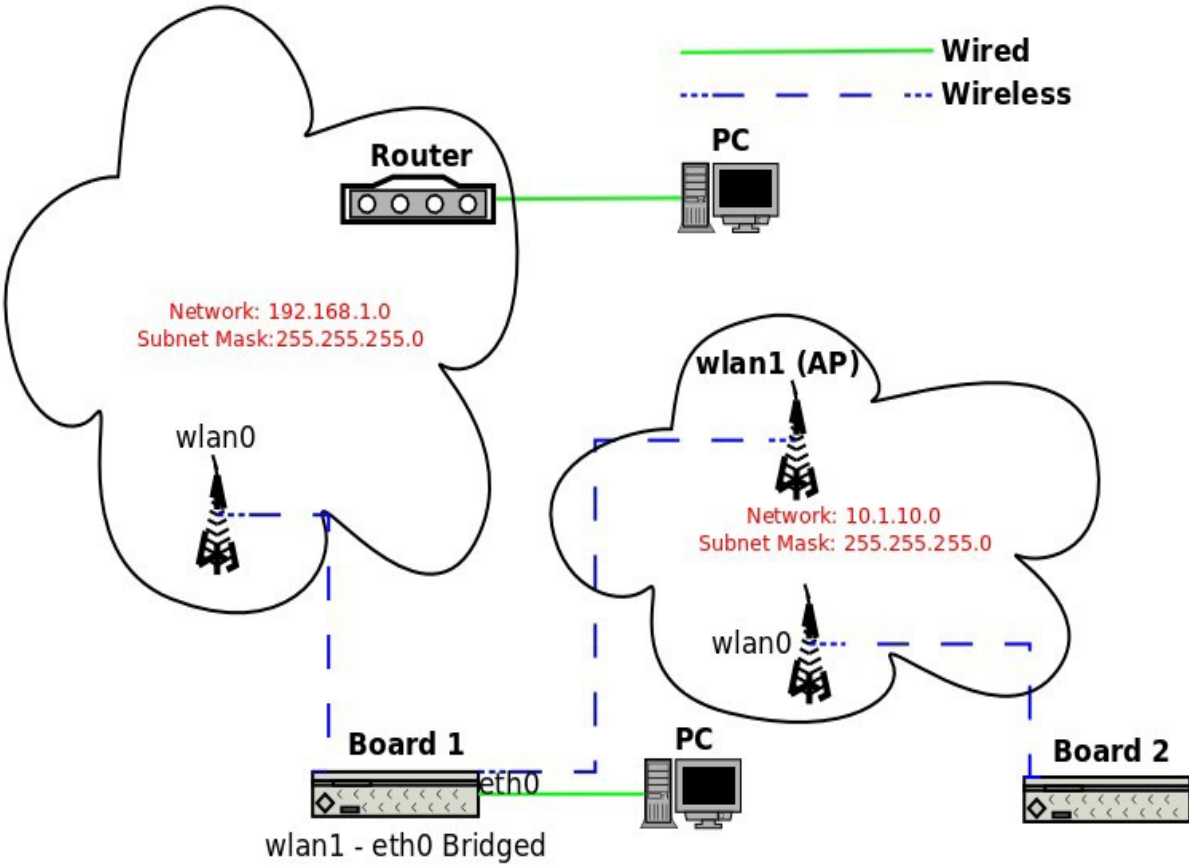
Για τον έλεγχο του δεύτερου board με ssh έχουμε δύο επιλογές. Η πρώτη επιλογή είναι να κάνουμε ssh στο πρώτο board και από εκεί να κάνουμε ssh στο δεύτερο χρησιμοποιώντας την εντολή:
ssh username_of_board2@ip_of_board2, πχ ssh root@10.1.10.83

Η δεύτερη επιλογή είναι να κάνουμε ssh μέσω του PC μας. Για αυτό όμως θα πρέπει να κάνουμε έναν νέο routing rule στο router μας, ώστε να γνωρίζει πως να δρομολογήσει τα πακέτα του PC με κατεύθυνση το δίκτυο 10.1.10.0. Ο νέος routing rule στο router μας πρέπει να είναι αυτός:

#	Dest IP	Mask	Gateway IP	Metric	Device	Edit	Drop
0	10.1.10.0	255.255.255.0	192.168.1.4	15	N/A		

Το Gateway IP είναι η IP που έχει δοθεί στο interface wlan0 του board1.

A.5.2 Bridging



Η επόμενη τοπολογία που θα εφαρμόσουμε είναι η εφαρμογή Bridging μεταξύ του wlan1 και του eth0 του board1. Με αυτό τον τρόπο οποιαδήποτε συσκευή συνδέσουμε στο eth0 στο board1, μπαίνει αμέσως στο δίκτυο του wlan1. Λόγω του bridge θα πρέπει να αλλάξουμε τα scripts για τα interfaces eth0 και wlan1 καθώς και να προσθέσουμε ένα για το bridge, το br0. Για να γίνει αυτό χρησιμοποιούμε στο board1 την εντολή:

vi /etc/network/interfaces

Βάζουμε σε σχόλια:

```
#create a AP
#auto wlan1
#iface wlan1 inet static
#address 10.1.10.1
#netmask 255.255.255.0
#broadcast 10.1.10.255
#hostapd /etc/hostapd/hostapd.wlan1.conf
#up nat.sh wlan1 wlan0 "10.1.10.0/24"
```

```
#auto eth0
#iface eth0 inet dhcp
```

και προσθέτουμε αυτό το κομμάτι:

```
auto eth0
iface eth0 inet manual

auto wlan1
iface wlan1 inet manual
hostapd /etc/hostapd/hostapd.wlan1.conf

auto br0
iface br0 inet static
address 10.1.10.1
netmask 255.255.255.0
broadcast 10.1.10.255
bridge_ports wlan1 eth0
```

Έτσι κάναμε bridge το interface wlan1 και eth0. Κάνουμε restart τα networks και των 2 boards.

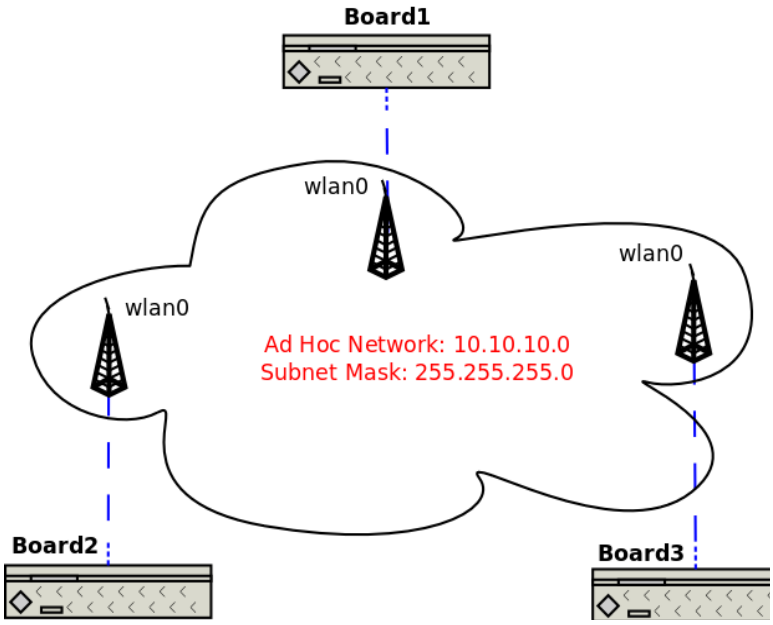
/etc/init.d/netwotking restart

Και όταν ανοίξουν ξανά, έχουμε υλοποιήσει το δίκτυο.

Τελικά κατά την εφαρμογή του παραδείγματος κατέληξα να έχω τις εξής Ips:

PC (connected to router): 192.168.1.2
PC (connected to board1 eth0): 10.1.10.129
board1 wlan0: 192.168.1.4
board1 br0 (bridged eth0 – wlan1, AP): 10.1.10.1
board2 wlan0: 10.1.10.83

A.5.3 Ad-Hoc Network



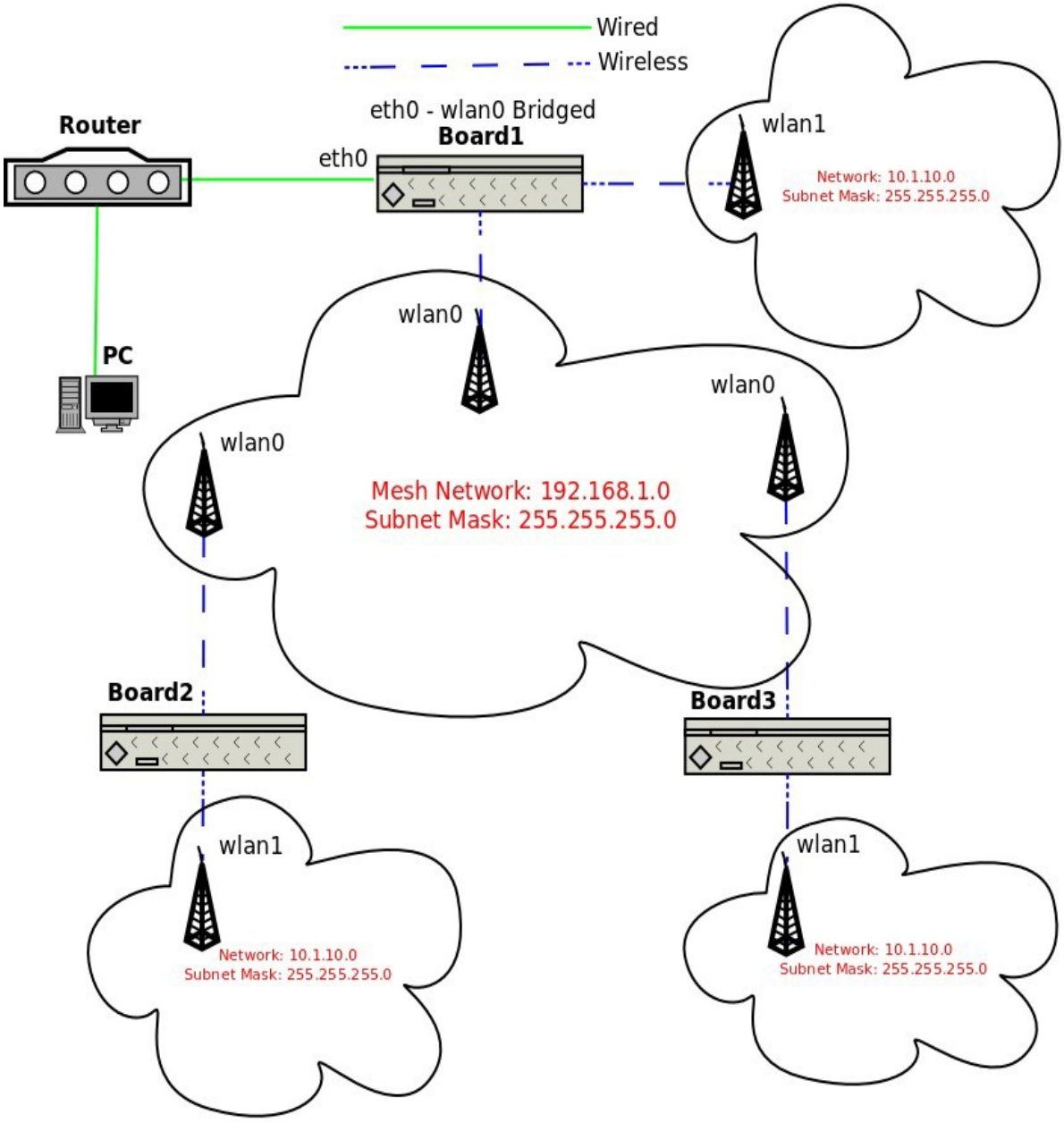
Η επόμενη τοπολογία που θα εξετάσουμε είναι μια απλή εφαρμογή κόμβων σε ad-hoc mode. Τα ad-hoc δίκτυα είναι στο layer 2 του OSI model κι έτσι δεν προσφέρουν δυνατότητες δρομολόγησης. Αυτό σημαίνει ότι κάθε κόμβος στο δίκτυο μας θα μπορεί να μιλήσει στους κόμβους που απέχουν από αυτόν ένα μόνο hop.

Αφού βάλουμε σε σχόλια όλες τις προηγούμενες προσθήκες που κάναμε στο αρχείο /etc/network/interfaces, προσθέτουμε το παρακάτω κομμάτι:

```
#Ad-Hoc
auto wlan0
iface wlan0 inet static
address 10.10.10.1
network 255.255.255.0
wireless-channel 1
wireless-essid AdHoc Network
wireless-mode ad-hoc
```

Αντίστοιχα και τους άλλους 2 κόμβους βάζουμε σε σχόλια όλα τα υπόλοιπα και προσθέτουμε το ίδιο κομμάτι στο /etc/network/interfaces, αλλάζοντας την IP. Για παράδειγμα στο δεύτερο board θα έχουμε IP 10.10.10.2 και στο τρίτο θα έχουμε 10.10.10.3. Κάνουμε restart το δίκτυο στα boards και έχουμε υλοποιήσει το ad-hoc δίκτυο.

5.4 Mesh Network



Σε αυτό το δίκτυο το board1 θα χρησιμοποιηθεί ως gateway για το mesh δίκτυο. Θα δρομολογεί όλα τα πακέτα Internet του δικτύου. Για να γίνει αυτό, κάνουμε bridge το eth0 με το mesh interface του board1 (wlan0) και συνδέουμε το eth0 με το router μας. Κάθε κόμβος έχει ένα interface για το mesh δίκτυο και άλλο ένα που υλοποιεί ένα AP. Τα devices που θα συνδεθούν σε κάθε AP έχουν πρόσβαση στο Internet.

Το board1 δημιουργεί το mesh network, παίρνοντας IP από τον dhcp server του router. Αντίστοιχα και τα board2, board3 κάνουν connect στο mesh network και παίρνουν IPs του ίδιου δικτύου.

Στο αρχείο /etc/network/interfaces σε όλα τα boards, βάζουμε όλα τα interfaces σε comments εκτός από αυτά για τα AP στα wlan1 που δημιουργήσαμε πριν.

Για το board1 που είναι gateway θα δώσουμε αυτές τις εντολές:

```
brctl addbr br0
brctl addif br0 eth0
iw dev wlan0 interface add mesh1 type mp
iw dev mesh1 set channel 1
brctl addif br0 mesh1
ifconfig mesh1 0.0.0.0 up
ifconfig eth0 0.0.0.0 up
iw dev mesh1 mesh join meshnet
dhclient br0
```

Απο τις οποίες θα δημιουργηθεί το mesh interface, θα γίνει bridge το eth0 με το mesh interface, θα δημιουργηθεί το mesh δίκτυο και θα πάρει ο κόμβος IP από τον router.

Για τους άλλους 2 κόμβους που δεν είναι gateways δίνουμε αυτές τις εντολές:

```
iw dev wlan0 interface add mesh2 type mp
iw dev mesh2 set channel 1
ifconfig mesh2 up
iw dev mesh2 mesh join meshnet
dhcpcd mesh2
```

```
iw dev wlan0 interface add mesh3 type mp
iw dev mesh3 set channel 1
ifconfig mesh3 up
iw dev mesh3 mesh join meshnet
dhcpcd mesh3
```

Κι έτσι έχουμε υλοποιήσει το mesh network.

Στα παρακάτω screenshots φαίνονται οι mac addresses των mesh interface των κόμβων καθώς και ο κάθε κόμβος πως δρομολογεί τα πακέτα του.

```
mesh1     Link encap:Ethernet  HWaddr 00:11:6b:41:d8:1b
          inet6 addr: fe80::211:6bff:fe41:d81b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:248 errors:0 dropped:0 overruns:0 frame:0
          TX packets:397 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:40444 (39.4 KiB)  TX bytes:63087 (61.6 KiB)
```

```
mesh2     Link encap:Ethernet  HWaddr 00:11:6b:4f:38:17
          inet addr:192.168.1.10  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::211:6bff:fe4f:3817/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:253 errors:0 dropped:0 overruns:0 frame:0
          TX packets:158 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:27931 (27.2 KiB)  TX bytes:32745 (31.9 KiB)
```

```
mesh3     Link encap:Ethernet  HWaddr 00:11:6b:4f:38:1d
          inet addr:192.168.1.6  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::211:6bff:fe4f:381d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:170 errors:0 dropped:0 overruns:0 frame:0
          TX packets:121 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18359 (17.9 KiB)  TX bytes:22263 (21.7 KiB)
```

Χρησιμοποιούμε σε κάθε κόμβο την εντολή:

iw dev mesh_interface mpath dump

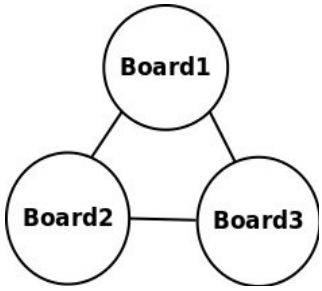
η οποία δείχνει τον τρόπο που δρομολογεί ο κάθε κόμβος τα πακέτα του.

```
root@voyage1:~# iw dev mesh1 mpath dump
DEST ADDR         NEXT HOP          IFACE
00:11:6b:4f:38:1d 00:11:6b:4f:38:1d mesh1
00:11:6b:4f:38:17 00:11:6b:4f:38:17 mesh1
```

```
root@voyage2:iw dev mesh2 mpath dump
DEST ADDR         NEXT HOP          IFACE
00:11:6b:4f:38:1d 00:11:6b:4f:38:1d mesh2
00:11:6b:41:d8:1b 00:11:6b:41:d8:1b mesh2
```

```
root@voyage3:~# iw dev mesh3 mpath dump
DEST ADDR         NEXT HOP          IFACE
00:11:6b:4f:38:17 00:11:6b:4f:38:17 mesh3
00:11:6b:41:d8:1b 00:11:6b:41:d8:1b mesh3
```

Όπως βλέπουμε από τα παραπάνω οι κόμβοι χρησιμοποιούν τα direct paths που τους ενώνουν για να μιλήσουν μεταξύ τους. Άρα το δίκτυο μας έχει αυτή την μορφή:



Μια βασική ιδιότητα των κόμβων σε ένα mesh network είναι η δυνατότητα δρομολόγησης. Σε αντίθεση με το ad-hoc δίκτυο που εξετάσαμε πριν, αν κόψουμε ένα direct path μεταξύ δύο κόμβων τότε θα πρέπει ο τρίτος κόμβος να αναλάβει την δρομολόγηση των πακέτων για την μεταξύ τους επικοινωνία. Έτσι για παράδειγμα αν κόψουμε το direct path των board 1 και 3 θα πρέπει το board 2 να αναλάβει να δρομολογήσει τα πακέτα από το board 1 στο board 3 και ανάποδα. Και αφού το board1 είναι και το gateway, το board 3 θα έχει πρόσβαση στο Internet από το board 2.

Για να κόψουμε το direct path των board 1 και board 3 θα πρέπει και στα 2 boards να χρησιμοποιήσουμε την εντολή:
Στο board1:

iw dev mesh1 station set 00:11:6b:4f:38:1d plink_action block

Στο board3:

iw dev mesh3 station set 00:11:6b:41:d8:1b plink_action block

Χρησιμοποιούμε την εντολή:

iw dev mesh_interface mpath dump

Στο board1:

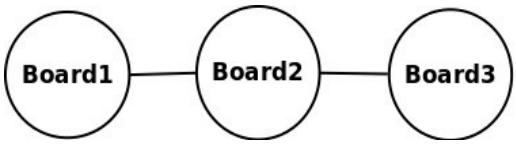
```
root@voyage1:~# iw dev mesh1 mpath dump
DEST ADDR         NEXT HOP          IFACE
00:11:6b:4f:38:1d 00:11:6b:4f:38:17 mesh1
00:11:6b:4f:38:17 00:11:6b:4f:38:17 mesh1
```

Στο board2:

```
root@voyage3:~# iw dev mesh3 mpath dump
DEST ADDR         NEXT HOP          IFACE
00:11:6b:4f:38:17 00:11:6b:4f:38:17 mesh3
00:11:6b:41:d8:1b 00:11:6b:4f:38:17 mesh3
```

Παρατηρούμε ότι όντως το board 2 έχει αναλάβει την δρομολόγηση των πακέτων όπως περιμέναμε. Τα ping λειτουργούν και το board 3 έχει ακόμα πρόσβαση στο Internet.

Άρα τώρα το δίκτυο μας έχει αυτή την μορφή:



Βιβλιογραφία

[1] Daniel Aguayo, John Bicket and Robert Morris, “SrcRR: A High Throughput Routing Protocol for 802.11 Mesh Networks (DRAFT)”.
http://pdos.csail.mit.edu/rtm/srcrr-draft.pdf

[2] Daniel Aguayo, John Bicket, Sanjit Biswas, Douglas S. J. De Couto, Robert Morris, “MIT Roofnet Implementation”.
http://archiv.iwi.uni-hannover.de/lv/seminar_ss04/www/Jan_Gacnik/bibliography/MIT-ROOFNET.pdf

[3] Paul Baran, “On distributed communications”, The RAND Corp., California, Santa Monica, Tech. Memo. RM-3764-PR, August 1964.

[4] John Jubin and Janet D. Tornow, “The DARPA packet radio network protocols”, Proceedings of the IEEE, Vol. 75, No. 1, January 1987.

[5] David B.Johnson, "Routing in AdHoc Networks of Mobile Hosts”, in Proceedings of the Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, pp.158-163, December 1994.

[6] Charles E. Perkins and Pravin Bhagwat, “Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers”, in Proc. ACM SIGCOMM Conference on Communications Architectures, Protocols and Applications, London, UK, September 1994.

[7] Kwan-Wu Chin, John Judge, Aidan Williams and Roger Kermode, “Implementation experience with MANET routing protocols”, ACM SIGCOMM Computer Communications Review, Vol. 32, No. 5, November 2002.

[8] Henrik Lundgren, Erik Nordstrom and Christian Tschudin, “Coping with communication gray zones in IEEE 802.11b based ad hoc networks”, The Fifth International Workshop on Wireless Mobile Multimedia, WOWMOM 2002, Atlanda, Georgia, September 2002.

[9] David A. Maltz, Josh Broch and David B. Johnson, “Experiences designing and building a multi-hop wireless ad hoc network testbed”, Tech. Rep. CMU-CS-99-116, School of Computer Science, Carnegie Mellon University, March 1999.

[10] David A. Maltz, Josh Broch and David B. Johnson, “Quantitative lessons from a full-scale multi-hop wireless ad hoc network testbed”, In Proceedings of the IEEE Wireless Communications and Networking Conference, Chicago, September 2000.

[11] Erik Nordstrom, Per Gunningberg and Henrik Lundgren, “A Testbed and Methodology for Experimental Evaluation of Wireless Mobile Ad hoc Networks”, In Proceedings of the First International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM), Trento, Italy, February 2005.

[12] Mark Yarvis, W. Steven Conner, Lakshman Krishnamurthy, Jasmeet Chhabra, Brent Elliott and Alan Mainwaring, "Real-world experiences with an interactive ad hoc sensor network", In Proceedings of the International Workshop on Ad Hoc Networking, Vancouver, British Columbia, Canada, pp. 143-151, August 2002.

[13] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin and S. Wicker, "Complex behavior at scale: An experimental study of low-power wireless sensor networks", Tech. Rep. UCLA/CSD-TR 02-0013, UCLA Computer Science, 2002.

[14] A technical tutorial on the IEEE 802.11 protocol by Pablo Brenner Director of Engineering.
http://www.seas.gwu.edu/~cheng/232/LectureNotes/802.11-tutorial.pdf

[15] ANSI/IEEE Std 802.11, 1999 Edition Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. LAN MAN Standards Committee of the IEEE Computer Society.

[16] http://www.its.ohiou.edu/kruse/its437/slides-80211phy.pdf

[17] Andrzej Duda, "Understanding the Performance of 802.11 Networks", In Personal, Indoor and Mobile Radio Communications, PIMRC 2008, IEEE 19th International Symposium, LIG Lab., Grenoble Inst. Of Technol., 2008.

[18] S. M. S. Bari, F. Anwar and M. H. Masud, "Performance Study of Hybrid Wireless Mesh Protocol (HWMP) for IEEE 802.11s WLAN Mesh Networks", International Conference on Computer and Communication Engineering (ICCCE 2012), Kuala Lumpur, Malaysia, July 2012.

[19] Venkat Mohan.S and Dr. Kasiviswanath.N, "Routing Protocols for Wireless Mesh Networks", International Journal of Scientific & Engineering Research Volume 2, Issue 8, August 2011.

[20] George Athanasiou, Thanasis Korakis, Ozgur Ercetin and Leandros Tassiulas, "A Cross-Layer Framework for Association Control in Wireless Mesh Networks", IEEE Transactions on Mobile Computing, Vol. 8, No. 1, pp. 65-80, January 2009.

[21] http://linux.voyage.hk/

[22] Tin-Yu Wu and Hung-Lin Chan, "Integrate Airtime Metric and Geocast over P2P–Based VoD Streaming Cache", Tamkang Journal of Science and Engineering, Vol. 13, No. 1, pp. 99-106, 2010