Πανεπιστήμιο Πειραιώς

«Τμήμα Ψηφιακών Συστημάτων»

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Διδακτική της Τεχνολογίας και Ψηφιακά Συστήματα»

Διπλωματική Εργασία

Συστήματα Ανίχνευσης Επιθέσεων

Ανδρέας Κουρκοβέλης

Φεβρουάριος 2011

Αφιερώνεται στην οικογένειά μου

Περίληψη

Τα τελευταία χρόνια με τις εξελίξεις στο χώρο των τηλεπικοινωνιών και τη ραγδαία ανάπτυξη του διαδικτύου, η πρόσβαση σε αυτό έπαψε να είναι πολυτέλεια και αποτελεί καθημερινή ασχολία για τους περισσότερους χρήστες. Επιπλέον, όλο και περισσότεροι ελεύθεροι επαγγελματίες απαιτούν την παροχή διαδικτυακών υπηρεσιών, ακόμα και εν κινήσει, προκειμένου να είναι πιο αποδοτικοί στην εργασία τους κάνοντας χρήση των πλεονεκτημάτων της τηλεργασίας.

Εξυπηρετώντας την ανάγκη αυτή για κινητικότητα, τα ασύρματα δίκτυα ad-hoc είναι αυτοοργανούμενα δίκτυα, τα οποία δεν απαιτούν την ύπαρξη κάποιας σταθερής δικτυακής υποδομής. Ειδικότερα, τα Mobile Ad-hoc NETworks (M.A.NETs), δηλαδή τα δίκτυα ad-hoc που αποτελούνται από κινητούς χρήστες, γνωρίζουν ιδιαίτερη άνθηση εξαιτίας της πληθώρας των εφαρμογών τους. Επιπλέον, αποτελούν και μια επιστημονική περιοχή με ιδιαίτερο ενδιαφέρον, καθώς οι συνθήκες και οι περιορισμοί που επικρατούν σε αυτά δυσχεραίνουν την λειτουργία τους.

Η σχεδίαση ενός συστήματος ανίχνευσης εισβολής (Intrusion Detection systems, IDS) σε ad-hoc ασύρματα δίκτυα, αποτελεί εξίσου μια ενδιαφέρουσα ερευνητική περιοχή, καθώς τα συστήματα αυτά θα πρέπει να λάβουν υπόψη τις παραμέτρους των ασύρματων δικτύων και ιδιαίτερα την κινητικότητα των χρηστών. Όσο υψηλότερη κινητικότητα έχουν οι χρήστες τόσο επηρεάζονται αρνητικά τα συστήματα ανίχνευσης εισβολών. Μια τεχνική που μπορεί να βελτιώσει τα συστήματα ανίχνευσης εισβολών, σε περιπτώσεις που η κινητικότητα των χρηστών του δικτύου είναι υψηλή, είναι η ομαδοποίηση των χρηστών με βάση την κινητικότητα τους (mobility aware clustering).

Στα πλαίσια της διπλωματικής αυτής πραγματοποιήθηκε θεωρητική μελέτη των συστημάτων ανίχνευσης εισβολών σε ad-hoc δίκτυα, καθώς και μελέτη των τεχνικών ομαδοποίησης χρηστών με βάση την κινητικότητα τους. Έπειτα, έγινε περιγραφή του συστήματος ανίχνευσης εισβολής και προσομοίωσή του με χρήση του λογισμικού NS-2, με αποτέλεσμα τον υπολογισμό της αποδοτικότητας του συστήματος ανίχνευσης εισβολών.

Σκοπός της διπλωματικής εργασίας είναι η μελέτη και η αξιολόγηση της εντροπίας ως μέθοδος ανίχνευσης ανωμαλιών (D.D.o.S. Attack, Worm propagation, Flash Crowd) στα ασύρματα δίκτυα M.A.NETs τόσο με ιεραρχική δρομολόγηση όσο και με δρομολόγηση ad-hoc, καθώς και σύγκριση των δύο περιπτώσεων δρομολόγησης.

Η διπλωματική εργασία αποτελείται από τέσσερα κεφάλαια . Στο πρώτο γίνεται λόγος για τα χαρακτηριστικά και τις εφαρμογές των ασύρματων δικτύων, ενώ στο τέλος αυτού αναφέρονται τα σημαντικότερα πρωτόκολλα δρομολόγησής τους. Το δεύτερο κεφάλαιο παρουσιάζει την ομαδοποίηση των ασύρματων δικτύων και τους αλγόριθμους της. Στο τρίτο κεφάλαιο πραγματοποιείται αναφορά στα συστήματα ανίχνευσης εισβολών καθώς και στις σημαντικότερες ανωμαλίες των δικτύων. Στο τελευταίο κεφάλαιο περιγράφεται η προσομοίωση και η υλοποίηση του αλγόριθμου ανίχνευσης, αναλύονται τα σενάρια επιθέσεων και σχολιάζονται τα αποτελέσματά τους. Τέλος, περιλαμβάνονται η βιβλιογραφία και δύο παραρτήματα με τα αριθμητικά αποτελέσματα του πειράματος και τον κώδικά του συστήματος ανίχνευσης εισβολών.

Ευχαριστίες

Θερμές ευχαριστίες εκφράζω στον Καθηγητή κύριο Σωκράτη Κάτσικα και στο Λέκτορα κύριο Χρήστο Ξενάκη για τη βοήθεια που μου παρείχαν για την ολοκλήρωση της διπλωματικής μου εργασίας.

Τέλος εκφράζω την ευγνωμοσύνη μου στους γονείς μου και την αδελφή μου για την υποστήριξη και τη βοήθειά τους καθ' όλη τη διάρκεια των μεταπτυχιακών σπουδών μου.

Περιεχόμενα

Περιεχόμενα Εικόνων

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή

Τα τελευταία χρόνια η χρήση προσωπικών υπολογιστών από κινούμενους χρήστες παρουσιάζει ιδιαίτερη αύξηση. Η διαρκώς αυξανόμενη ανάγκη για επικοινωνία αποτελεί μια από τις μεγαλύτερες τάσεις της εποχής μας, με αποτέλεσμα τα δίκτυα επικοινωνίας να αποκτούν ιδιαίτερη βαρύτητα σε τεχνολογικό επίπεδο. Στις μέρες μας ο στόχος των ασύρματων δικτύων είναι η εφικτή επικοινωνία παντού και πάντα. Ο στόχος αυτός οδηγεί στη περαιτέρω εξέλιξη των ασύρματων δικτύων έτσι ώστε συσκευές όπως τα notebooks και τα P.D.As να μπορούν να αποτελούν μέρος ενός ενιαίου δικτύου διασύνδεσης.

Με τις πρόσφατες εξελίξεις στην απόδοση των ασύρματων δικτύων, αναμένεται ευρεία διάδοση και χρήση πιο προχωρημένων ασύρματων δικτύων με κινητούς κόμβους, που θα εξελίξουν κατά πολύ τη χρήση του Internet Protocol (I.P.). Το όραμα της ad-hoc δικτύωσης με κινητούς κόμβους είναι να υποστηρίξει αποτελεσματικά τη χρήση των M.A.NETs ενσωματώνοντας λειτουργίες δρομολόγησης στους κινητούς κόμβους. Τέτοια δίκτυα θα έχουν δυναμικές, γρήγορα εναλλασσόμενες, τυχαίες και multi-hop τοπολογίες, που θα αποτελούνται από σχετικά περιορισμένου εύρους ζώνης ασύρματες ζεύξεις.

1.1. Περιγραφή ασύρματων δικτύων

Στην «κοινωνία» του Internet, η υποστήριξη δρομολόγησης για κινητούς κόμβους μορφοποιείται ως τεχνολογία Mobile I.P. Είναι μια τεχνολογία που θα επιτρέπει σε κόμβους (nodes) να επικοινωνούν με το Internet, συνδεόμενοι σε «φιλόξενους» κόμβους (hosts), που θα είναι ήδη συνδεδεμένοι με διάφορα μέσα, πέρα από τη σταθερή τους διεύθυνση. Αυτός ο host μπορεί να είναι φυσικά συνδεδεμένος με το σταθερό δίκτυο σε ένα ξένο υποδίκτυο ή να είναι συνδεδεμένος με μια ασύρματη ζεύξη με μια σύνδεση dial-up ή με οτιδήποτε άλλο. Αυτή η μορφή του «κινητού φιλόξενου κόμβου» απαιτεί διαχείριση διευθύνσεων, βελτιώσεις στη διασύνδεση των πρωτοκόλλων και άλλα σχετικά. Όμως, οι βασικές δικτυακές

λειτουργίες, όπως η δρομολόγηση από κόμβο σε κόμβο, εξαρτώνται ακόμα από τα ήδη υπάρχοντα πρωτόκολλα δρομολόγησης που λειτουργούν στα σταθερά δίκτυα.

Αντίθετα, ο στόχος των M.A.NETs είναι να επεκτείνουν αυτή την κινητικότητα σε αυτόνομα, κινητά, ασύρματα domain, όπου ένα σύνολο κόμβων, από μόνοι τους, αποτελούν την υποδομή για τη δρομολόγηση με έναν ad-hoc τρόπο.

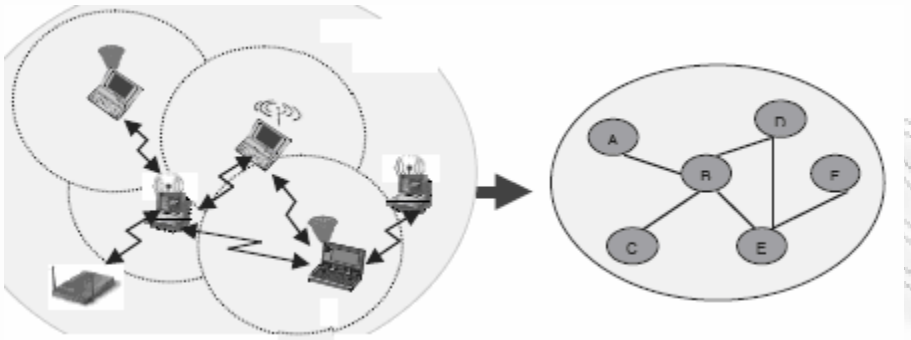
1.2. Χαρακτηριστικά ασύρματων δικτύων

Ένα M.A.NET. αποτελείται από κινητές μονάδες (π.χ. ένα δρομολογητή με πολλούς hosts και ασύρματες συσκευές, που αποκαλούνται κόμβοι), οι οποίες είναι ελεύθερες να μετακινηθούν όπου θέλουν. Αυτοί οι κόμβοι μπορούν να βρίσκονται σε αεροπλάνα, πλοία, φορτηγά, αυτοκίνητα, ακόμα και σε ανθρώπους! Ένα M.A.NET. είναι ένα αυτόνομο σύστημα αποτελούμενο από κινητούς κόμβους. Το σύστημα αυτό μπορεί να λειτουργεί απομονωμένο ή να έχει διεξόδους (gateways) και να επικοινωνεί με ένα σταθερό δίκτυο. Στο δεύτερο τρόπο λειτουργίας, το σύστημα θα λειτουργεί σαν ένα αποκομμένο δίκτυο (stub network). Τα stub networks μεταφέρουν δικτυακή κίνηση που προέρχεται ή κατευθύνεται προς τους εσωτερικούς κόμβους, αλλά δεν επιτρέπει εξωτερική κίνηση να μεταφερθεί μέσω αυτού.

Οι κόμβοι ενός M.A.NET. είναι εξοπλισμένοι με ασύρματους πομπούς και δέκτες χρησιμοποιώντας κεραίες που μπορεί να είναι μη κατευθυντικές (omnidirectional), πολύ κατευθυντικές (point-to-point), πιθανώς μεταβλητές ή συνδυασμός όλων αυτών. Σε κάποιο χρονικό σημείο ανάλογα με τη θέση των κόμβων, την εμβέλεια των πομποδεκτών τους, τη μεταδιδόμενη ισχύ τους και τα επίπεδα παρεμβολών, μια ασύρματη σύνδεση δημιουργείται ανάμεσά τους στη μορφή ενός τυχαίου ad-hoc δικτύου. Αυτή η ad-hoc τοπολογία μπορεί να αλλάξει με την πάροδο του χρόνου, καθώς οι κόμβοι μετακινούνται ή αλλάζουν την ισχύ μετάδοσής τους.

Τα M.A.NETs έχουν πολλά αξιοπρόσεκτα χαρακτηριστικά, όπως :

1. **Δυναμικές τοπολογίες.** Οι κόμβοι είναι ελεύθεροι να μετακινούνται αυθαίρετα, με αποτέλεσμα η τοπολογία του δικτύου, που είναι συνήθως multi-hop, να μπορεί να αλλάζει τυχαία και γρήγορα σε απρόβλεπτες στιγμές. Επίσης, η τοπολογία μπορεί να απαρτίζεται και από αμφίδρομες (bidirectional) αλλά και από μονοκατευθυντικές (unidirectional) ζεύξεις.



Εικόνα 1 Ένα αδόμητο δίκτυο και η τοπολογία του

2. **Ζεύξεις περιορισμένου εύρους ζώνης και μεταβλητής χωρητικότητας.** Οι ασύρματες ζεύξεις θα συνεχίσουν να έχουν σημαντικά μικρότερες χωρητικότητες από τις αντίστοιχες ενσύρματες. Επιπρόσθετα, το πραγματικό ¨throughput¨ στις ασύρματες επικοινωνίες, αν λάβουμε υπόψη τα φαινόμενα της πολλαπλής πρόσβασης (multiple access), του θορύβου, των παρεμβολών, και άλλων, είναι συνήθως πολύ μικρότερο από τη μέγιστη εκπομπή των πομπών. Ένα αποτέλεσμα της σχετικά χαμηλής έως μέτριας χωρητικότητας των ζεύξεων είναι ο συνωστισμός, ο οποίος τυπικά είναι ο κανόνας και όχι η εξαίρεση (π.χ. η αυξανόμενη ζήτηση μιας δικτυακής εφαρμογής πιθανότατα θα πλησιάσει ή και θα ξεπεράσει τη χωρητικότητα του δικτύου). Αφού το κινητό δίκτυο είναι συχνά μια επέκταση ενός σταθερού δικτύου, οι χρήστες αυτού απαιτούν παρόμοιες υπηρεσίες με αυτές του σταθερού δικτύου. Οι απαιτήσεις αυτές θα συνεχίσουν να αυξάνονται με την ανάπτυξη των πολυμέσων και των δικτυακών εφαρμογών.
3. **Λειτουργία με περιορισμένη ενέργεια τροφοδοσίας.** Μερικοί ή όλοι οι κόμβοι σε ένα M.A.NET. μπορεί να βασίζονται σε συσσωρευτές ή σε άλλα εξαντλήσιμα μέσα τροφοδοσίας. Για αυτούς τους κόμβους, το πιο σημαντικό κριτήριο σχεδιασμού βελτίωσης είναι η κατανάλωση ενέργειας.
4. **Περιορισμένη ασφάλεια του φυσικού μέσου μετάδοσης** (physical layer). Τα κινητά ασύρματα δίκτυα είναι γενικά πιο ευπαθή σε θέματα ασφάλειας στο φυσικό επίπεδο σε σχέση με τα ασύρματα δίκτυα. Οι αυξημένες πιθανότητες για επιθέσεις τύπου κρυφακούσματος (eavesdropping), εξαπάτησης (spoofing) και D.o.S. (Denial of Service) θα πρέπει να ληφθούν σοβαρά υπόψη. Εφαρμόζονται ήδη υπάρχοντες τεχνικές ασφάλειας στα ασύρματα δίκτυα για

τη μείωση των τρωτών σημείων στην ασφάλεια τους. Η αποκεντρωμένη φύση της διαχείρισης των M.A.NETs παρέχει επιπλέον κάλυψη για περιπτώσεις που κάποιος κόμβος βγει εκτός λειτουργίας σε σχέση με τις πιο συγκεντρωμένες προσεγγίσεις.

5. **Κλιμάκωση** (Scalability). Σε κάποια πιθανά M.A.NETs, όπως στρατιωτικά δίκτυα ή δίκτυα αυτοκινητόδρομων, ο αριθμός των κόμβων ενδέχεται να είναι σχετικά μεγάλος, μερικών δεκάδων ή ακόμα και εκατοντάδων κόμβων ανά περιοχή δρομολόγησης και επομένως απαιτείται η υποστήριξη κλιμάκωσης σε αυτά. Μπορεί η ανάγκη για κλιμάκωση να μην είναι μοναδική για τα M.A.NETs, αλλά είναι οι μηχανισμοί για την επίτευξή της.

1.3. Ασφάλεια ασύρματων δικτύων

Τα ad-hoc δίκτυα χαρακτηρίζονται από τη μη ύπαρξη κεντρικής λειτουργίας διαχείρισης τους και τη συνεχιζόμενη αλλαγή της τοπολογίας τους. Τα χαρακτηριστικά αυτά κάνουν τα ad-hoc δίκτυα ευπρόσβλητα σε έναν αριθμό επιθέσεων. Για να γίνουν αυτά τα δίκτυα ευρέως αποδεκτά στον εμπορικό κόσμο θα πρέπει να επιλυθεί το θέμα της ασφάλειας, που είναι υπέρτατης σημασίας και εξαιρετικά δύσκολο να επιτευχθεί. Οι λόγοι είναι η ευπάθεια των ασύρματων συνδέσμων, η περιορισμένη φυσική προστασία των κόμβων, η έλλειψη μιας κεντρικής παρακολούθησης καθώς επίσης και η δυνατότητα που υπάρχει να συνδεθεί στο δίκτυο εξωτερικός εχθρικός κόμβος, ο οποίος θα επιχειρήσει να το βλάψει. Ο όρος ασφάλεια υπονοεί την ικανοποίηση πολλών απαιτήσεων και αναγκών όπως :

• **Διαθεσιμότητα**(availability). Ο όρος διαθεσιμότητα δηλώνει ότι οι υπηρεσίες που παρέχονται από όλους τους κόμβους του δικτύου θα πρέπει να συνεχίζουν να παρέχονται ανεξαρτήτως της ύπαρξης επιθέσεων, δηλαδή οι κόμβοι θα πρέπει να μπορούν να επικοινωνούν ανά πάσα στιγμή!

• **Αυθεντικότητα** (authenticity). Με τον όρο αυθεντικότητα εννοείται ότι όσοι κόμβοι συμμετέχουν στην επικοινωνία είναι «γνήσιοι» και μπορούν να αποδείξουν την ταυτότητά τους.

- **Εμπιστευτικότητα** (confidentiality). Για να ικανοποιηθεί η εμπιστευτικότητα θα πρέπει να μην επιτρέπεται σε κάποιον εξωτερικό κόμβο να μπορεί να προσπελάσει την πληροφορία που διασχίζει δύο κόμβους.
- **Ακεραιότητα** (Integrity). Το πακέτο ή το μήνυμα που παραδίδεται δε θα πρέπει να έχει τροποποιηθεί, δηλαδή παραλαμβάνεται ότι έχει αποσταλεί.
- **Επικαιρότητα** (timeliness). Με τον όρο επικαιρότητα εννοούμε ότι οι ενημερώσεις δρομολόγησης θα πρέπει να παραδίδονται εγκαίρως.
- **Απομόνωση** (isolation). Η απομόνωση απαιτεί το πρωτόκολλο που χρησιμοποιείται να είναι ικανό να απομονώνει όσους κόμβους επιδεικνύουν κακή συμπεριφορά.
- **Εξουσιοδότηση** (authorization). Σχετίζεται με τα προνόμια και τις εξουσιοδοτήσεις που σχετίζονται με τον κάθε κόμβο του δικτύου.

1.4. Εφαρμογές ασύρματων δικτύων

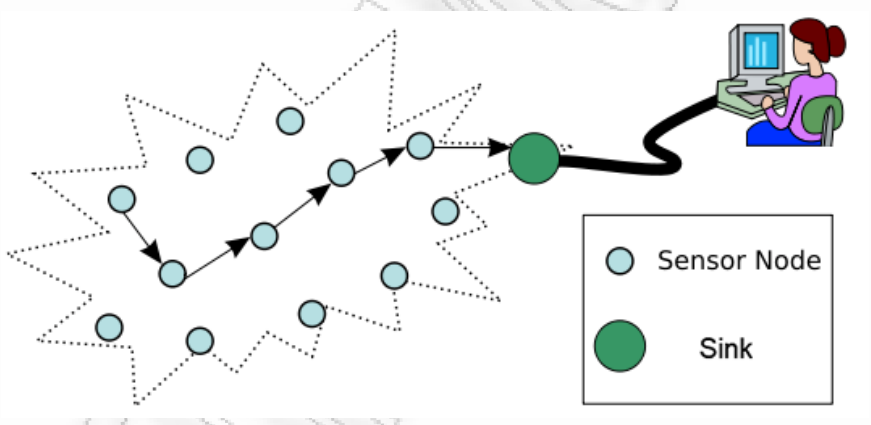
Τα M.A.NETs έχουν πρακτική εφαρμογή σε περιπτώσεις όπου δεν υπάρχει κάποια σταθερή ενσύρματη δικτυακή υποδομή. Τέτοιες περιπτώσεις έχουμε όταν δεν είναι οικονομικά, πρακτικά ή γεωγραφικά εφικτό να δημιουργηθεί η απαραίτητη υποδομή ή επειδή οι καταστάσεις δεν επιτρέπουν την εγκατάστασή της, όπως :

1. Σε καταστάσεις έκτακτης ανάγκης, μετά από φυσικές ή άλλες καταστροφές, όπου όλες οι υποδομές έχουν καταρρεύσει και υπάρχει άμεση απαίτηση για επικοινωνία.
2. Σε εικονικές (virtual) αίθουσες, αίθουσες σεμιναρίων και συνεδριάσεων κυρίως σε ακαδημαϊκό περιβάλλον, όπου η ανάγκη για επικοινωνία πρέπει να έχει ελευθερία χώρου.
3. Σε χώρους υγειονομικής περίθαλψης.
4. Σε συστήματα ασφαλείας ή παρακολούθησης.
5. Σε στρατιωτικές επιχειρήσεις.
6. Σε δημόσιους χώρους, όπως λιμάνια, αεροδρόμια και αυτοκινητόδρομους για παροχή χρησίμων πληροφοριών και τουριστικών οδηγιών.
7. Σε άλλες περιπτώσεις, όπως στο σπίτι για τη διευκόλυνση της διαχείρισης των οικιακών συσκευών και τη παροχή νέων υπηρεσιών από αυτές.

Είναι φανερό ότι τα ασύρματα ad-hoc δίκτυα αποτελούν μια νέα γενιά δικτύων, η οποία θα δώσει λύσεις σε περιπτώσεις όπου μέχρι πρότινος δεν υπήρχαν καθώς επίσης και να διευκολύνει την καθημερινότητα μεγάλης μερίδας ανθρώπων.

1.5. Ασύρματα δίκτυα αισθητήρων

Τα ασύρματα δίκτυα αισθητήρων (Wireless Sensor Networks), αποτελούν μια μορφή των M.A.NETs, σχετικά με τον αριθμό των κόμβων, και τις περιορισμένες δυνατότητές τους. Οι αισθητήρες μπορεί να είναι παραγόμενες συσκευές χαμηλού κόστους που λειτουργούν για χρόνια με μπαταρίες. Ένα σύνολο πολλών αισθητήρων μπορεί να «απλωθεί» σε μια περιοχή που παρακολουθείται, και από τη στιγμή εκείνη και μετά οι αισθητήρες να είναι σταθεροί και να λειτουργούν όσο τους επιτρέπουν οι συσσωρευτές τους.



Εικόνα 2 Ασύρματα δίκτυα αισθητήρων

Γύρω από το δίκτυο των αισθητήρων, τοποθετούνται σταθμοί (sinks) που συλλέγουν τις μετρήσεις των αισθητήρων και μεταφέρουν τις εντολές των ελεγκτών. Έτσι, οι αισθητήρες σχηματίζουν μια δομή που προωθεί πακέτα από αυτούς προς τους σταθμούς και αντίστροφα.

Για να δημιουργηθεί μια υποδομή δρομολόγησης και να υπάρξει κλιμάκωση σε μια μεγάλη γεωγραφική περιοχή, οι αισθητήρες τοποθετούνται έτσι ώστε να σχηματίζουν τη μορφή ενός δέντρου ή τη μορφή διανεμημένου γράφου που έχει σκοπό την βελτιστοποίηση της επικοινωνίας και την ελαχιστοποίηση της κατανάλωσης ισχύος.

Οι αισθητήρες μπορούν να γνωρίζουν μόνο τη δικιά τους θέση και προκειμένου να βελτιωθεί η δημιουργία καινούριων διαδρομών, ανταλλάζουν πληροφορίες σχετικά με τις θέσεις των κόμβων ανάμεσα στους σταθμούς και στους αισθητήρες. Επίσης, για τη βελτίωση της ανίχνευσης και του εντοπισμού μιας συσκευής που εισέρχεται σε μια περιοχή που παρακολουθείται, οι αισθητήρες μπορούν να ανταλλάξουν πληροφορίες σχετικά με τις θέσεις και την κινητικότητα των κόμβων.

1.6. Δρομολόγηση ad-hoc ασύρματων δικτύων

Η ανάπτυξη αποδοτικών πρωτοκόλλων δρομολόγησης είναι μία από τις σημαντικότερες προκλήσεις στα M.A.NETs και θεμελιώδης για την παροχή των υπηρεσιών, για τα οποία προορίζονται. Ορισμένοι μοναδικοί συνδυασμοί χαρακτηριστικών των M.A.NETs κάνουν το πρόβλημα της δρομολόγησης εξαιρετικά ενδιαφέρον. Πρώτον, οι κόμβοι σε ένα M.A.NET. επιτρέπεται να κινούνται ανεξέλεγκτα και άναρχα, με αποτέλεσμα τη δημιουργία ενός πολύ δυναμικά μεταβαλλόμενου δικτύου με ραγδαίες και συνεχείς τοπολογικές αλλαγές, που οδηγούν σε συχνές καταστροφές και υπολογισμούς δρομολογίων. Ένα «καλό» πρωτόκολλο δρομολόγησης για ένα τέτοιο περιβάλλον θα πρέπει να προσαρμόζεται δυναμικά και γρήγορα στην μεταβαλλόμενη τοπολογία του δικτύου. Δεύτερον, τα ασύρματα κανάλια που χρησιμοποιούν τα M.A.NETs παρέχουν χαμηλό εύρος ζώνης, το οποίο αυξομειώνεται ανάλογα με τις συνθήκες (π.χ. λόγω παρεμβολών, κακών καιρικών συνθηκών κ.τ.λ.). Το γεγονός ότι τα ασύρματα δίκτυα είναι εκ φύσεως δίκτυα εκπομπής και οι διαθέσιμες ζώνες συχνοτήτων είναι περιορισμένες, αφήνει ελάχιστο διαθέσιμο εύρος ζώνης για κάθε κόμβο. Συνεπώς, είναι ζωτικής σημασίας για κάθε πρωτόκολλο δρομολόγησης να είναι αποδοτικό όσον αφορά το εύρος ζώνης, προσθέτοντας ελάχιστες επιβαρύνσεις στο δίκτυο και περιορίζοντας τις μεταδόσεις στις απολύτως απαραίτητες. Τρίτον, οι κόμβοι των M.A.NETs αποτελούνται από φορητές συσκευές, οι οποίες λειτουργούν με μπαταρίες και άρα διαθέτουν περιορισμένους ενεργειακούς πόρους. Έτσι, αν επιθυμούμε οι κόμβοι να παραμένουν ενεργοί για μεγαλύτερα χρονικά διαστήματα, θα πρέπει τα πρωτόκολλα δρομολόγησης να είναι αποδοτικά όσον αφορά τη διαχείριση της ενέργειας των κόμβων.

Τα «κλασσικά» πρωτόκολλα δρομολόγησης που έχουν σχεδιαστεί για ενσύρματα δίκτυα είναι ανεπαρκή για τα M.A.NETs, καθώς δεν υποθέτουν μόνο σταθερή τοπολογία, αλλά επίσης εισάγουν και υψηλές επιβαρύνσεις. Το γεγονός αυτό έχει οδηγήσει σε μια έντονη ερευνητική δραστηριότητα και στην ανάπτυξη πολλών εξειδικευμένων πρωτοκόλλων δρομολόγησης για τα M.A.NETs. Ενώ κάποια από τα πρωτόκολλα που έχουν προταθεί είναι ουσιαστικά εκδόσεις υπαρχόντων πρωτοκόλλων ενσύρματων δικτύων, τα υπόλοιπα έχουν εισάγει νέες ιδέες για το πρόβλημα της δρομολόγησης, όπως για παράδειγμα την κατ' απαίτηση (on-demand ή reactive) δρομολόγηση. Σύμφωνα, με αυτή την κατηγορία πρωτοκόλλων τα δρομολόγια δημιουργούνται και συντηρούνται μόνο όταν και όσο χρειάζονται. Η λογική αυτή βρίσκεται σε αντίθεση με την παραδοσιακή βασιζόμενη σε πίνακες (table-driven ή proactive) λογική. Άλλες «επαναστατικές» τεχνικές έχουν να κάνουν με την εκμετάλλευση πληροφοριών για τη γεωγραφική θέση των κόμβων (π.χ. με τη χρήση δεκτών G.P.S.) ή πληροφοριών για την ενέργεια και την ισχύ του σήματος των κόμβων.

Οι υποθέσεις που γίνονται συνήθως από την πλειοψηφία των αλγορίθμων δρομολόγησης για M.A.NETs είναι :

1) Κάθε κόμβος έχει ένα μοναδικό αναγνωριστικό (unique identity), με βάση το οποίο μπορεί να προσδιοριστεί μονοσήμαντα μέσα στο δίκτυο. Επιπλέον, κάθε κόμβος είναι σε θέση να στέλνει και να λαμβάνει πακέτα.

2) Υπάρχει ένα υπό-επίπεδο M.A.C. με δυνατότητες παροχής ενημέρωσης για την κατάσταση του δικτύου.

3) Κάθε κόμβος εκπέμπει περιοδικά ένα αναγνωριστικό μήνυμα (beacon), ώστε να δηλώσει την παρουσία του, επιτρέποντας έτσι σε κάθε κόμβο να γνωρίζει την ύπαρξη και τα χαρακτηριστικά των γειτόνων του.

Παρόλο που πολύ γνωστά πρωτόκολλα δεν απαιτούν τη χρήση κάποιου μηχανισμού αναγνώρισης (beaconing), όπως π.χ. τα A.O.D.V. και D.S.R., η χρήση αυτή είναι ζωτικής σημασίας. Αυτό οφείλεται στη δυναμική φύση των M.A.NETs γιατί αλλιώς οι κόμβοι δεν θα ήταν σε θέση να γνωρίζουν τους γείτονές τους.

Τα πρωτόκολλα δρομολόγησης για τα M.A.NETs μπορούν να κατηγοριοποιηθούν με βάση τα χαρακτηριστικά τους. Στη βιβλιογραφία διακρίνονται δύο βασικές κατηγορίες. Η πρώτη κατηγορία είναι τα πρωτόκολλα βασιζόμενα σε πίνακες (proactive ή table-driven), τα οποία διατηρούν πίνακες με πληροφορίες δρομολόγησης, τους οποίους ενημερώνουν περιοδικά σχετικά με την καινούργια

κατάσταση του δικτύου. Τη δεύτερη κατηγορία αποτελούν τα πρωτόκολλα δρομολόγησης κατ' απαίτηση (reactive ή on-demand driven), τα οποία ανακαλύπτουν και συντηρούν δρομολόγια μόνο όταν και όσο αυτά χρειάζονται.

Επίσης, διακρίνουμε πρωτόκολλα βασισμένα σε περιορισμούς (constraint-based), τα οποία σε αντίθεση με τις παραπάνω πρωτόκολλα (που βασίζουν τις αποφάσεις δρομολόγησής τους στην «αρχή» της συντομότερης διαδρομής), επιλέγουν τα δρομολόγια τους με βάση ποικίλα χαρακτηριστικά, όπως για παράδειγμα την κατανάλωση ενέργειας ή την ισχύ του λαμβανόμενου σήματος. Μια ακόμα κατηγορία αποτελείται από πρωτόκολλα που χρησιμοποιούν σύνολα υποστήριξης (support groups), δηλαδή σύνολα κόμβων των οποίων η κίνηση καθορίζεται από το ίδιο το πρωτόκολλο. Οι κόμβοι του συνόλου υποστήριξης παραλαμβάνουν μηνύματα από τους αποστολείς, όταν τους συναντούν τυχαία στην περιοχή του δικτύου και αναλαμβάνουν να τα παραδώσουν στον προορισμό εκ μέρους τους. Επίσης, γνωστά είναι αρκετά πρωτόκολλα δρομολόγησης που βασίζονται στη γεωγραφική θέση των κόμβων(geographical routing). Επιπλέον, όπως και στα ενσύρματα δίκτυα, υπάρχουν πρωτόκολλα που χρησιμοποιούν κάποια μορφή ιεραρχίας ή γενικότερα διαχωρίζουν τους κόμβους του δικτύου σε ομάδες (hierarchical routing ή cluster-based routing). Μερικές ακόμα τεχνικές βασίζονται σε έννοιες τεχνητής νοημοσύνης (artificial intelligence), όπως τεχνικές νοημοσύνης σμήνους (swarm intelligence). Τέλος, υπάρχουν και υβριδικές λύσεις, οι οποίες προσπαθούν να συνδυάσουν δύο ή περισσότερες από τις παραπάνω τεχνικές.

1.7. Δρομολόγηση ad-hoc ασύρματων δικτύων βασιζόμενη σε πίνακες

Τα πρωτόκολλα δρομολόγησης για αδόμητα δίκτυα που βασίζονται σε πίνακες (proactive ή table-driven) είναι ουσιαστικά μια επέκταση των παραδοσιακών πρωτοκόλλων δρομολόγησης. Το κύριο χαρακτηριστικό αυτής της κατηγορίας πρωτοκόλλων είναι ότι κάθε κόμβος στο δίκτυο διατηρεί ένα δρομολόγιο προς κάθε άλλο κόμβο στο δίκτυο για κάθε δεδομένη χρονική στιγμή. Η δημιουργία και η συντήρηση των δρομολογίων επιτυγχάνεται μέσω ανταλλαγής μηνυμάτων ενημέρωσης (updates), τα οποία περιέχουν πληροφορίες σχετικές με την δρομολόγηση. Μηνύματα ενημέρωσης ανταλλάσσονται μεταξύ των κόμβων του δικτύου είτε περιοδικά (periodic updates) είτε μετά από κάποιο συμβάν (event-

triggered updates). Οι περιοδικές ανταλλαγές μηνυμάτων ενημέρωσης συμβαίνουν σε τακτά χρονικά διαστήματα, ανεξάρτητα από την κινητικότητα ή την κίνηση (traffic) του δικτύου. Αντίθετα, οι βασιζόμενες σε συμβάντα ενημερώσεις συμβαίνουν όταν πραγματοποιείται κάποιο σημαντικό γεγονός, όπως για παράδειγμα η δημιουργία ή η καταστροφή ενός συνδέσμου. Η αύξηση της κινητικότητας του δικτύου επιδρά καθοριστικά στον αριθμό αυτής της κατηγορίας ενημερώσεων και συνήθως οδηγεί σε αύξηση των καταστροφών και των σχηματισμών συνδέσμων.

Το βασικό πλεονέκτημα αυτής της κατηγορίας πρωτοκόλλων είναι ότι υπάρχει διαθέσιμη μια διαδρομή προς κάθε κόμβο στο δίκτυο οποτεδήποτε χρειαστεί. Επειδή κάθε κόμβος διατηρεί με συνέπεια ενημερωμένες διαδρομές προς κάθε άλλο κόμβο στο δίκτυο, ένας κόμβος-πηγή που παραλαμβάνει ένα πακέτο δεδομένων, αρκεί να ελέγξει τον πίνακα δρομολόγησής του για να ξεκινήσει την μετάδοσή του, επιτυγχάνοντας έτσι μικρότερες καθυστερήσεις στην παράδοση των πακέτων. Επίσης, τα οδηγούμενα από πίνακες πρωτόκολλα, επιτυγχάνουν την εύρεση των βέλτιστων (συντομότερων) διαδρομών. Από την άλλη, το βασικό μειονέκτημα αυτής της προσέγγισης είναι ότι έχει μεγάλο κόστος σε πακέτα ελέγχου σε περιπτώσεις μεγάλων δικτύων ή υψηλής μεταβλητότητας της τοπολογίας του δικτύου (π.χ. λόγω υψηλής κινητικότητάς ή μεγάλου αριθμού παρεμβολών). Επιπρόσθετα, η ανάγκη για τη διατήρηση πληροφοριών σχετικών με την δρομολόγηση αυξάνεται αρκετά, κάτι το οποίο είναι ανεπιθύμητο λόγω των περιορισμένων πόρων των φορητών υπολογιστικών συσκευών, που συνήθως αποτελούν τους κόμβους ενός M.A.NET. Σε γενικές γραμμές, τα οδηγούμενα από πίνακες πρωτόκολλα έχουν την τάση να συμπεριφέρονται «καλά» σε δίκτυα όπου υπάρχει μεγάλος αριθμός ενεργών συνεδριών (data sessions). Σε αυτές τις περιπτώσεις το κόστος συντήρησης των δρομολογίων που χρησιμοποιούνται από τις συνεδρίες δικαιολογείται λόγω της αυξημένης αξιοποίησής τους (utilization).

1.7.1. Πρωτόκολλο Destination Sequenced Distance Vector (D.S.D.V.)

Το πρωτόκολλο Destination Sequenced Distance Vector (D.S.D.V.) προτάθηκε το 1994 από τους C. Perkins και P. Bhagwat και έχει επικρατήσει ως το βασικό παράδειγμα οδηγούμενου από πίνακες, πρωτοκόλλου δρομολόγησης για αδόμητα δίκτυα. Το D.S.D.V. αποτελεί μία επέκταση του γνωστού αλγορίθμου R.I.P.

Ο βασικός σχεδιαστικός στόχος του D.S.D.V. ήταν η βελτίωση της συμπεριφοράς, και κυρίως της αστάθειας και των κύκλων δρομολόγησης (routing loops) που επιδείκνυε ο R.I.P., σε συνθήκες καταστροφής συνδέσμων. Αυτό οδηγούσε σε χρονικά εξαρτώμενη αντίληψη της τοπολογίας του δικτύου. Ο τρόπος λειτουργίας του είναι παρόμοιος με αυτόν του R.I.P. Τα πακέτα μεταδίδονται μεταξύ των κόμβων χρησιμοποιώντας πίνακες δρομολόγησης αποθηκευμένους σε κάθε κόμβο. Κάθε πίνακας δρομολόγησης, σε κάθε κόμβο, περιέχει όλους τους διαθέσιμους προορισμούς και τον αριθμό των αλμάτων (hops) που απαιτούνται για καθένα από αυτούς, όπως επίσης και το επόμενο άλμα προς κάθε προορισμό. Προφανώς, αντί για τον αριθμό των αλμάτων μπορεί να χρησιμοποιηθεί οποιαδήποτε άλλη μετρική (π.χ. καθυστέρηση). Κάθε καταχώρηση στον πίνακα δρομολόγησης συνδέεται με έναν αριθμό ακολουθίας (sequence number) που είναι και ο μεγαλύτερος που έχει δει ο κόμβος για τον συγκεκριμένο προορισμό. Εξαιτίας της δυναμικής φύσης των αδόμητων δικτύων (οι πληροφορίες που περιέχονται στους πίνακες δρομολόγησης γίνονται μέσα σε μικρό χρονικό διάστημα ασυνεπείς) απαιτείται περιοδική εκπομπή των τοπολογικών αλλαγών (periodic updates). Επίσης, σε περίπτωση σημαντικών τοπολογικών αλλαγών (π.χ. κάποιας δημιουργίας ή καταστροφής συνδέσμου) οι κόμβοι επανεκπέμπουν άμεσα τις πληροφορίες για την καινούργια τοπολογία του δικτύου (incremental updates). Πρέπει να τονιστεί ότι, όπως και σε κάθε άλλο πρωτόκολλο που κάνει χρήση διανυσμάτων απόστασης, η εκπομπή των τοπολογικών πληροφοριών γίνεται ασύγχρονα.

Οι κύριες τοπολογικές πληροφορίες που εκπέμπει κάθε κόμβος για κάθε προορισμό είναι ο αριθμός των αλμάτων από τον κόμβο μέχρι τον προορισμό και ο αριθμός ακολουθίας αυτού. Κάθε κόμβος ανανεώνει τις πληροφορίες του πίνακα δρομολόγησής του βασιζόμενος στον αριθμό ακολουθίας για κάθε προορισμό. Συγκεκριμένα, ο πίνακας δρομολόγησης ενημερώνεται μόνο στην περίπτωση που ο αριθμός ακολουθίας που περιέχεται σε αυτόν είναι μικρότερος από αυτόν που περιέχεται στο μήνυμα των τοπολογικών αλλαγών. Μεταξύ επιλογών με τον ίδιο αριθμό ακολουθίας προτιμάται αυτή με την μικρότερη τιμή για τον αριθμό των αλμάτων. Οι αριθμοί ακολουθίας για έναν προορισμό ανανεώνονται κάθε φορά που συμβαίνει μια τοπολογική αλλαγή που σχετίζεται με τον συγκεκριμένο προορισμό. Ο λόγος ύπαρξής τους είναι η διάκριση μεταξύ παλιών και νέων δρομολογίων και αποδεικνύεται ότι με τη βοήθεια τους αποφεύγεται η δημιουργία κύκλων δρομολόγησης (loop). Επίσης, για την αποφυγή άσκοπων ενημερώσεων, λόγω της

ασύγχρονης φύσης του πρωτοκόλλου, το πρωτόκολλο εισάγει κάποια χρονικά όρια μεταξύ ενός συμβάντος (π.χ. καταστροφή ενός συνδέσμου) και της αποστολής μιας ενημέρωσης για αυτό.

1.8. Δρομολόγηση ad-hoc ασύρματων δικτύων κατ' απαίτηση

Ένα μεγάλο ποσοστό του κόστους δρομολόγησης των βασιζόμενων σε πίνακες πρωτοκόλλων πηγάζει από το γεγονός ότι κάθε κόμβος πρέπει να διατηρεί ένα δρομολόγιο προς κάθε άλλο κόμβο στο δίκτυο για κάθε χρονική στιγμή. Σε ένα ενσύρματο δίκτυο, όπου η συνδεσμολογία μεταβάλλεται σπάνια και το δίκτυο έχει αρκετούς πόρους, η διατήρηση ενημερωμένων εγγραφών της τοπολογίας αξίζει το κόστος. Το προφανές πλεονέκτημα είναι ότι υπάρχει άμεσα διαθέσιμο ένα δρομολόγιο μεταξύ δύο κόμβων του δικτύου, όταν αυτό απαιτείται. Από την άλλη, σε ένα αδόμητο δίκτυο η συνδεσμολογία μπορεί να αλλάζει συχνά και κατά συνέπεια το κόστος συντήρησης των δρομολογίων είναι ιδιαίτερα υψηλό. Εξαιτίας κυρίως αυτού του λόγου, η κατ' απαίτηση προσέγγιση διαφοροποιείται ριζικά από τις παραδοσιακές τεχνικές δρομολόγησης του Internet επειδή δεν διατηρεί συνεχώς δρομολόγια μεταξύ όλων των κόμβων του δικτύου. Αντίθετα, ένα δρομολόγιο ανακαλύπτεται μόνο όταν πραγματικά απαιτείται. Συγκεκριμένα, όταν ένας κόμβος-πηγή επιθυμεί να στείλει πακέτα δεδομένων σε κάποιον προορισμό, ελέγχει αρχικά εάν διαθέτει ήδη κάποιο δρομολόγιο προς τον προορισμό. Στην περίπτωση που δεν υπάρχει δρομολόγιο, εκτελεί μια διαδικασία ανακάλυψης δρομολογίου (route discovery), μέσω της οποίας ανακαλύπτει ένα κατάλληλο δρομολόγιο, εφόσον βέβαια υπάρχει. Έτσι, η ανακάλυψη δρομολογίων γίνεται κατ' απαίτηση. Εφόσον δύο κόμβοι δεν χρειάζεται ποτέ να επικοινωνήσουν, τότε δεν χρειάζεται να σπαταλήσουν τους πόρους τους για να συντηρήσουν ένα δρομολόγιο μεταξύ τους. Η ανακάλυψη δρομολογίων πραγματοποιείται από την εκπομπή μιας αίτησης (request) σε όλο το δίκτυο για τη δημιουργία δρομολογίου.

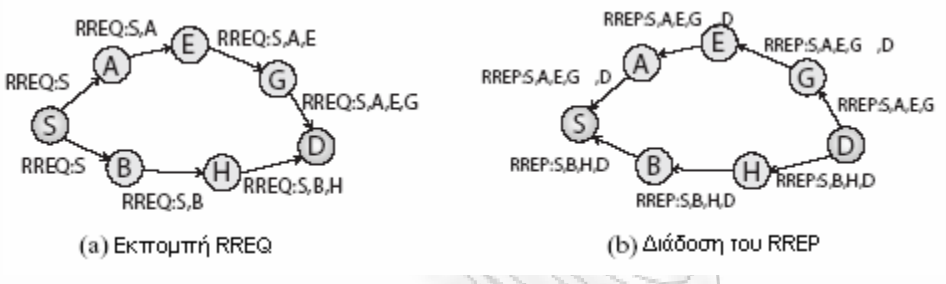
Το κύριο πλεονέκτημα της κατ' απαίτηση προσέγγισης είναι ότι το κόστος σηματοδοσίας (signaling overhead) στη γενική περίπτωση μειώνεται (για την επίτευξη συγκρίσιμων επιδόσεων του δικτύου) σε σχέση με αυτό της βασιζόμενης σε πίνακες προσέγγισης, ιδιαίτερα σε περιπτώσεις δικτύων με χαμηλή ή μέτρια κίνηση. Όταν η κίνηση στο δίκτυο αυξάνεται και ο αριθμός των ενεργών συνόδων δεδομένων

(data sessions) στο δίκτυο γίνεται μεγαλύτερος, τότε το κόστος για την ανακάλυψη δρομολογίων πλησιάζει και σε αρκετές περιπτώσεις ξεπερνά αυτό των βασιζόμενων σε πίνακες πρωτοκόλλων. Το βασικό μειονέκτημα της κατ' απαίτηση δρομολόγησης είναι η εισαγωγή μιας καθυστέρησης στην εύρεση δρομολογίου (route acquisition latency). Αυτό συμβαίνει γιατί όταν απαιτηθεί ένα δρομολόγιο από κάποια πηγή και το δρομολόγιο δεν είναι διαθέσιμο, τα πακέτα δεδομένων θα πρέπει να εισαχθούν σε κάποια ουρά μέχρι να βρεθεί το δρομολόγιο ή να αποφασιστεί ότι δεν υπάρχει τέτοιο δρομολόγιο. Το ίδιο πρόβλημα παρουσιάζεται και σε περιπτώσεις που κάποιος σύνδεσμος του δρομολογίου καταστρέφεται, οπότε τα πακέτα καθυστερούν μέχρι να ανακαλυφθεί ένα νέο δρομολόγιο προς τον προορισμό και αυτά που βρίσκονται καθοδόν (on the fly) απορρίπτονται. Ένα ακόμα πρόβλημα της κατ' απαίτηση δρομολόγησης είναι ότι τα συχνά χρησιμοποιούμενα δρομολόγια, με την πάροδο του χρόνου, παύουν να είναι βέλτιστα. Αυτό συμβαίνει γιατί στα «κλασικά» κατ' απαίτηση πρωτόκολλα ένα δρομολόγιο χρησιμοποιείται από τη στιγμή της ανακάλυψης μέχρι τη στιγμή της καταστροφής του. Όμως, είναι πολύ πιθανό κατά τη διάρκεια της ζωής ενός δρομολογίου, εξαιτίας των χαρακτηριστικών των M.A.NETs, να δημιουργηθεί κάποιο καταλληλότερο δρομολόγιο, το οποίο εντούτοις δεν θα χρησιμοποιηθεί.

1.8.1. Πρωτόκολλο Dynamic Source Routing (D.S.R.)

Η βασική ιδιότητα του D.S.R. πρωτοκόλλου, το οποίο είναι και το πρώτο κατ' απαίτηση πρωτόκολλο που προτάθηκε για τα M.A.NETs, είναι η έννοια της δρομολόγησης πηγής (source routing). Δρομολόγηση πηγής έχουμε όταν ο αποστολέας ενός πακέτου γνωρίζει την πλήρη διαδρομή μέχρι τον προορισμό, την οποία και περιλαμβάνει στην κεφαλίδα (header) κάθε πακέτου. Κάθε διαδρομή που γνωρίζει ο κόμβος αποθηκεύεται σε μια μνήμη διαδρομών (route cache). Κάθε φορά που ο κόμβος επιθυμεί να στείλει ένα πακέτο ερευνά τη μνήμη για μια διαδρομή μέχρι τον προορισμό του πακέτου. Σε περίπτωση που μια τέτοια διαδρομή δεν υπάρχει, χρησιμοποιείται ο πρώτος μηχανισμός που παρέχει το πρωτόκολλο, αυτός της ανακάλυψης δρομολογίου (route discovery). Η ανακάλυψη ενός δρομολογίου πραγματοποιείται με μια διαδικασία πλημμύρας του δικτύου με μηνύματα τύπου αίτησης δρομολογίου (R.REQ.). Κάθε κόμβος που λαμβάνει ένα τέτοιο μήνυμα το

επανεκπέμπει, εκτός και αν είναι ο ενδεικνυόμενος, στο R.REQ. μήνυμα, προορισμός ή αν γνωρίζει μια διαδρομή προς αυτόν, η οποία είναι αποθηκευμένη στη μνήμη διαδρομών του. Στην περίπτωση αυτή απαντάει στο R.REQ. μήνυμα με ένα μήνυμα απάντησης διαδρομής (R.REP.), το οποίο δρομολογείται πίσω στην πηγή του R.REQ. μηνύματος. Κατά την διάδοση του R.REQ. μηνύματος προς τον προορισμό, οι κόμβοι από τους οποίους διέρχεται προστίθενται στην κεφαλίδα του. Η πληροφορία αυτή χρησιμοποιείται στη συνέχεια, με ανεστραμμένη σειρά, για τη δρομολόγηση του μηνύματος R.REP. πίσω στην πηγή.



Εικόνα 3 Διαδικασία ανακάλυψης δρομολογίου από το D.S.R. πρωτόκολλο

Ο δεύτερος μηχανισμός που παρέχει το D.S.R. πρωτόκολλο είναι η συντήρηση δρομολογίου (route maintenance). Ο μηχανισμός αυτός συνίσταται από διαδικασίες για την αντιμετώπιση τυχών καταστροφής δρομολογίων, τα οποία χρησιμοποιούνται από κάποιους κόμβους. Έτσι, αν κάποιος σύνδεσμος που περιέχεται σε κάποιο δρομολόγιο καταστραφεί, τότε οι κόμβοι πηγής όλων των δρομολογίων που τον περιέχουν ενημερώνονται με την χρήση ενός μηνύματος σφάλματος διαδρομής (R.ERR.). Όταν η πηγή λάβει το R.ERR. μήνυμα απομακρύνει από τη μνήμη της όλα τα δρομολόγια που περιέχουν αυτό το σύνδεσμο. Προφανώς, σε περίπτωση που κάποια από τις διαδρομές που απομακρύνθηκε από τη μνήμη συνεχίζει να είναι απαραίτητη, μια νέα διαδικασία ανακάλυψης δρομολογίου θα πραγματοποιηθεί.

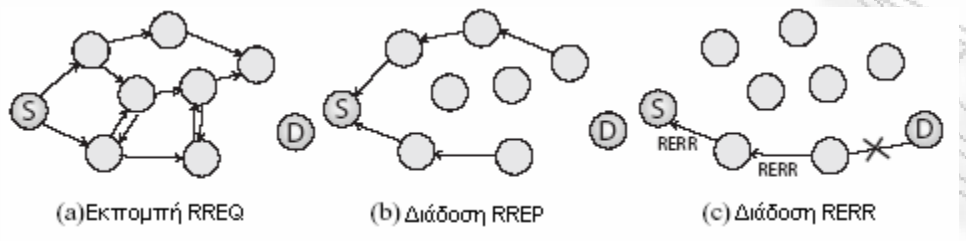
Στην παραπάνω βασική λειτουργία του πρωτοκόλλου μπορεί να προστεθεί ένα μεγάλο σύνολο βελτιώσεων. Έτσι, για παράδειγμα κάθε κόμβος που προωθεί ένα πακέτο, μπορεί να εξάγει από την κεφαλίδα του τη διαδρομή από τον εαυτό του μέχρι τον προορισμό του πακέτου και να την αποθηκεύσει στη μνήμη του για μελλοντική χρήση. Επιπλέον, ένας κόμβος που διαπιστώνει ότι ένας από τους συνδέσμους του έχει καταστραφεί, αντί να ενημερώσει την πηγή ενός πακέτου, μπορεί να ελέγξει τη μνήμη του για μια διαδρομή μέχρι τον προορισμό. Επιπρόσθετα, κάθε κόμβος που

προωθεί ένα R.ERR. πακέτο προς την πηγή, μπορεί να χρησιμοποιήσει την πληροφορία του R.ERR. πακέτου για να απομακρύνει τα ασυνεπή δρομολόγια από τη μνήμη του. Τέλος, σε περίπτωση που το M.A.C. πρωτόκολλο παρέχει τη δυνατότητα ωτακουστικής λειτουργίας (promiscuous mode), δηλαδή την ικανότητα με την οποία ένας κόμβος να λαμβάνει πακέτα που δεν προορίζονται για αυτόν, τότε μπορεί να «ακούει» τις μεταδόσεις των άλλων κόμβων και να ενημερώνει τη μνήμη του με τις πληροφορίες που συγκεντρώνει από αυτές.

1.8.2. Πρωτόκολλο Ad-hoc On-demand Distance Vector (A.O.D.V.)

Το A.O.D.V. πρωτόκολλο είναι ουσιαστικά ένας συνδυασμός των πρωτοκόλλων D.S.R. και D.S.D.V. Δανείζεται τον βασικό κατ' απαίτηση μηχανισμό αναζήτησης δρομολογίων του D.S.R., χρησιμοποιώντας ταυτόχρονα πίνακες δρομολόγησης και αριθμούς ακολουθίας σε κάθε κόμβο, καθώς και περιοδικά σήματα αναγνώρισης (beacons), όπως το D.S.D.V. Όταν μια πηγή (S) χρειάζεται ένα δρομολόγιο σε έναν προορισμό (D), εκπέμπει ένα μήνυμα αίτησης δρομολογίου, το οποίο περιλαμβάνει τον πιο πρόσφατο αριθμό ακολουθίας για τον συγκεκριμένο προορισμό. Το μήνυμα αίτησης δρομολογίου μεταδίδεται σε ολόκληρο το δίκτυο μέχρι να συναντήσει ένα κόμβο (συμπεριλαμβανομένου του προορισμού) που ξέρει ένα δρομολόγιο στον D με αριθμό ακολουθίας μεγαλύτερο ή ίσο με αυτόν που περιέχει το μήνυμα (προφανώς ο D πάντα θα έχει έναν αριθμό ακολουθίας μεγαλύτερο ή ίσο από αυτόν του S). Κάθε κόμβος που προωθεί το μήνυμα δημιουργεί μια αντίστροφη καταχώρηση (δηλαδή μια καταχώρηση για τον κόμβο S) στον πίνακα δρομολόγησής του. Επιπλέον, κάθε κόμβος που γνωρίζει μια διαδρομή στον D, εκδίδει ένα μήνυμα απάντησης δρομολογίου που περιέχει την απόσταση του από τον D, καθώς και τον πιο πρόσφατο αριθμό ακολουθίας για τον D. Κάθε κόμβος που συμμετέχει στην προώθηση του μηνύματος απάντησης δρομολογίου πίσω στον S, δημιουργεί μια κανονική καταχώρηση (δηλαδή μια καταχώρηση για τον D) στον πίνακα δρομολόγησης του. Πρέπει να σημειωθεί ότι για οποιονδήποτε προορισμό, κάθε κόμβος διατηρεί μόνο τις πληροφορίες για τον αριθμό των αλμάτων (ή γενικά για το κόστος του δρομολογίου),για το επόμενο άλμα προς τον προορισμό, καθώς και για τον τελευταίο (μεγαλύτερο) αριθμό ακολουθίας του. Επίσης, είναι αρκετά πιθανό ο κόμβος-πηγή να ανακαλύψει περισσότερα από ένα δρομολόγια, οπότε και επιλέγει

το καταλληλότερο από αυτά για την δρομολόγηση των πακέτων του. Τα υπόλοιπα δρομολόγια καταργούνται μετά την πάροδο κάποιου χρονικού διαστήματος χωρίς να χρησιμοποιηθούν.



Εικόνα 4 Ανακάλυψη και συντήρηση δρομολογίων από το A.O.D.V. πρωτόκολλο

Για την συντήρηση των δρομολογίων ο A.O.D.V. απαιτεί την περιοδική εκπομπή HELLO μηνυμάτων, με εξορισμού διάστημα μεταξύ των εκπομπών ίσο με ένα δευτερόλεπτο. Αποτυχία για λήψη τριών διαδοχικών μηνυμάτων HELLO από κάποιο κόμβο εκλαμβάνεται ως ένδειξη καταστροφής του συνδέσμου με αυτόν τον κόμβο. Σε κάποιες εργασίες, η λειτουργικότητα των HELLO μηνυμάτων του A.O.D.V. πρωτοκόλλου μπορεί να αντικατασταθεί από τις ικανότητες ανίχνευσης κατεστραμμένων συνδέσμων που μπορεί να παρέχει το M.A.C. πρωτόκολλο. Βέβαια, η παραπάνω τροποποίηση αλλοιώνει τη φύση του πρωτοκόλλου, εφόσον οι απαρχαιωμένες καταχωρίσεις του πίνακα δρομολόγησης μπορούν να απομακρυνθούν μόνο όταν υπάρχει κίνηση, οδηγώντας σε ελαφρώς χειρότερες επιδόσεις. Κάθε φορά που με κάποια από τις παραπάνω τεχνικές εντοπιστεί ένας κατεστραμμένος σύνδεσμος, κάθε κόμβος που έχει προωθήσει πακέτα μέσω αυτού, στο πρόσφατο παρελθόν, ενημερώνεται μέσω ενός μηνύματος σφάλματος δρομολογίου, το οποίο περιέχει άπειρη τιμή για το κόστος του δρομολογίου. Κάθε κόμβος που λαμβάνει ένα τέτοιο μήνυμα και εξακολουθεί να χρειάζεται το δρομολόγιο, ξεκινάει μια νέα διαδικασία ανακάλυψης δρομολογίου.

Όπως και στην περίπτωση του D.S.R., ένα πλήθος βελτιώσεων έχει προταθεί για την βελτίωση της απόδοσης του A.O.D.V.. Μία από αυτές της βελτιώσεις είναι η αντικατάσταση της πλημμύρας στη διαδικασία ανακάλυψης δρομολογίων από μια τεχνική αναζήτησης εκτεινόμενου δακτυλίου (expanding ring search). Σύμφωνα με την τεχνική αυτή, η διάδοση των R.REQ. μηνυμάτων επιτρέπεται (μέσω ελέγχου του πεδίου T.T.L. των I.P. πακέτων) σε ολοένα και εκτεινόμενες περιοχές του δικτύου, μέχρι να ανακαλυφθεί ένα δρομολόγιο προς τον προορισμό. Αυτό σημαίνει για παράδειγμα ότι αρχικά ελέγχεται η περιοχή του δικτύου που περιλαμβάνει όλους τους

κόμβους που βρίσκονται σε απόσταση 2 αλμάτων από την πηγή, στη συνέχεια η περιοχή των κόμβων με απόσταση έως 4 άλματα από την πηγή (εφόσον στην πρώτη αναζήτηση δεν ανακαλύφθηκε κάποιο δρομολόγιο προς τον προορισμό), στη συνέχεια οι κόμβοι που απέχουν έως 8 άλματα από την πηγή κ.ο.κ. Μια ακόμα βελτίωση όσον αφορά την συντήρηση των δρομολογίων είναι να προσπαθεί ο ίδιος να ανακαλύψει ένα δρομολόγιο προς τον προορισμό, αντί ο κόμβος να στέλνει ένα R.ERR. μήνυμα στην πηγή. Εάν ο κόμβος καταφέρει να ανακαλύψει ένα δρομολόγιο, λιγότερα πακέτα θα χαθούν, εφόσον η καθυστέρηση επιδιόρθωσης του δρομολογίου είναι μικρότερη. Σε περίπτωση που η επιδιόρθωση του δρομολογίου αποτύχει, ακολουθείται κανονικά η διαδικασία της αποστολής του R.ERR. μηνύματος.

Επιπλέον των παραπάνω τεχνικών, έχουν προταθεί και κάποιες προσθήκες στο A.O.D.V., οι οποίες βελτιώνουν κάτω από συγκεκριμένες συνθήκες τις επιδόσεις του πρωτοκόλλου. Έτσι, για παράδειγμα έχει προταθεί η χρήση χαριστικών R.REP. (gratuitous R.REP.), τα οποία στέλνονται στον προορισμό για να τον ενημερώσουν για την καταστροφή του δρομολογίου. Αυτή η τακτική μπορεί να βοηθήσει στην συνηθισμένη περίπτωση που υπάρχει κίνηση και προς τις δύο κατευθύνσεις του δρομολογίου (π.χ. κίνηση T.C.P). Μια άλλη προσθήκη είναι η χρήση επιβεβαιώσεων των R.REP. (A.C.Ks), οι οποίες χρησιμοποιούνται για την επιβεβαίωση της παράδοσης των μηνυμάτων R.REP. στον επόμενο κόμβο προς την πηγή.

1.8.3. Πρωτόκολλο Associativity Based Routing (A.B.R.)

Το A.B.R. πρωτόκολλο σχεδιάστηκε και αναπτύχθηκε από το πανεπιστήμιο του Cambridge το 1996. Το βασικό χαρακτηριστικό του A.B.R. είναι η έννοια της συσχέτισης (associativity). Το A.B.R. είναι, όπως και το D.S.R., πρωτόκολλο δρομολόγησης πηγής και κατά συνέπεια δεν απαιτεί περιοδικές ενημερώσεις των δρομολογίων του. Το A.B.R. εκτός από πρωτόκολλο κατ' απαίτησης δρομολόγησης, μπορεί να χαρακτηριστεί και ως πρωτόκολλο δρομολόγησης βασιζόμενο σε περιορισμούς (constraint-based), αφού επιβάλλει τη δημιουργία δρομολογίων με συγκεκριμένα χαρακτηριστικά.

Η βασική ιδέα της συσχέτισης είναι ότι στα M.A.NETs δεν έχει νόημα να επιλέγεις ένα δρομολόγιο με βάση τη συντομότερη διαδρομή, εφόσον το δρομολόγιο θα καταστραφεί ή θα καταστεί άχρηστο λόγω της κινητικότητας των κόμβων. Αυτό

που προτείνει το πρωτόκολλο είναι κάθε κόμβος του δικτύου να μαθαίνει την «συσχέτιση» του με τους κόμβους που το περιβάλλουν και με βάση αυτή να επιλέγει την καλύτερη διαδρομή. Η σταθερότητα μιας περιοχής καθορίζεται με τη χρήση ticks συσχέτισης. Η συσχέτιση στο A.B.R. μπορεί να συνυπολογίζει έναν αριθμό μετρικών, όπως την καθυστέρηση των συνδέσμων, την ισχύ του λαμβανόμενου σήματος, την υπολειπόμενη ενέργεια των κόμβων, το φόρτο των συνδέσμων και πολλά άλλα χαρακτηριστικά.

Αυτό που διαφοροποιεί το A.B.R. από τα υπόλοιπα πρωτόκολλα είναι η χρήση της συσχέτισης ως κύριας μετρικής για την επιλογή δρομολογίων. Κατ' αυτό τον τρόπο τα δρομολόγια που επιλέγονται είναι σταθερότερα και κατά συνέπεια καταστρέφονται σπανιότερα. Η λειτουργία του A.B.R. μπορεί να διασπαστεί λογικά σε τρεις φάσεις: τη φάση ανακάλυψης δρομολογίων, τη φάση ανακατασκευής δρομολογίων και τη φάση καταστροφής δρομολογίων.

Η φάση ανακάλυψης δρομολογίων αποτελείται από την εκπομπή μιας αίτησης και έναν κύκλο αναμονής-απάντησης (BQ-REPLY). Αρχικά όλοι οι κόμβοι, εκτός από τους γείτονες του προορισμού D, δεν διαθέτουν δρομολόγια προς αυτόν. Κάθε κόμβος που επιθυμεί ένα δρομολόγιο προς τον D εκπέμπει ένα BQ μήνυμα σε ολόκληρο το δίκτυο αναζητώντας κόμβους που έχουν ένα δρομολόγιο προς τον D. Κάθε ενδιάμεσος κόμβος (intermediate node) που λαμβάνει το BQ μήνυμα προσθέτει τη διεύθυνσή του και την τιμή της συσχέτισης των συνδέσμων του με τους γείτονές του και το επανεκπέμπει, μαζί με κάποιες άλλες απαραίτητες μετρικές. Ο επόμενος ενδιάμεσος κόμβος θα αφαιρέσει τις πληροφορίες της συσχέτισης που εισήγαγε στο BQ μήνυμα, θα τις αξιοποιήσει για την επιλογή του δρομολογίου και στη συνέχεια θα προσθέσει τις δικές του. Κατά συνέπεια το BQ μήνυμα είναι μεταβλητού μεγέθους.

Η δεύτερη φάση του A.B.R. είναι η κατ' απαίτηση επιδιόρθωση δρομολογίων (on demand route recovery phase). Αυτή η φάση είναι χρήσιμη επειδή τόσο τα δρομολόγια όσο και η τοπολογία ενός M.A.NET μεταβάλλονται συνεχώς. Η επιλογή ενός δρομολογίου που αναμένεται να μην καταστραφεί για μεγάλο χρονικό διάστημα δεν είναι από μόνη της αρκετή, καθώς το δρομολόγιο μπορεί να καταστραφεί εξαιτίας της κίνησης ή της απενεργοποίησης κάποιων κόμβων.

Τα κύρια χαρακτηριστικά της επιδιόρθωσης δρομολογίων που παρέχει το A.B.R. είναι:

• Μερική ανακάλυψη δρομολογίων με την χρήση μηνυμάτων τοπικών αιτήσεων (localized queries – L.Q.)

• Διαγραφή άκυρων δρομολογίων (invalid route erasure)

• Ανανέωση έγκυρων δρομολογίων (valid route update)

• Ανακάλυψη νέων δρομολογίων

Οι παραπάνω λειτουργίες μπορεί να ενεργοποιηθούν από κινήσεις των κόμβων του δικτύου, οι οποίες οδηγούν σε καταστροφές συνδέσμων ή γενικότερα από σημαντικά συμβάντα στο δίκτυο (π.χ. επικίνδυνη μείωση της διαθέσιμης ενέργειας ενός κόμβου).

Η τελευταία φάση του A.B.R. είναι η κατ' απαίτηση διαγραφή δρομολογίων (on demand route deletion phase). Η ανάγκη για διαγραφή δρομολογίων προκύπτει ως μέρος της συντήρησης, αφού δρομολόγια που δεν χρησιμοποιούνται οδηγούν σε σπατάλη πόρων και η επιπλέον μελλοντική χρήση τους μπορεί να οδηγήσει σε προβλήματα εφόσον οι πληροφορίες τους είναι απαρχαιωμένες. Υπάρχουν δύο διαθέσιμες μέθοδοι για τη διαγραφή δρομολογίων: της απαλής κατάστασης (soft state) και της σκληρής κατάστασης (hard state). Στην μαλακή προσέγγιση, η λήξη μιας διαδρομής βασίζεται στην έλλειψη κίνησης ή στη λήξη ενός προκαθορισμένου χρονικού διαστήματος. Συγκεκριμένα, κάθε κόμβος παρακολουθεί την κίνηση κάθε δρομολογίου και σε περίπτωση που διαπιστώσει ότι η κίνηση βρίσκεται κάτω από ένα κατώφλι, τη θεωρεί ληγμένη. Στην περίπτωση της σκληρής κατάστασης, η λήξη ενός δρομολογίου πραγματοποιείται με τη μετάδοση ενός συγκεκριμένου μηνύματος ελέγχου.

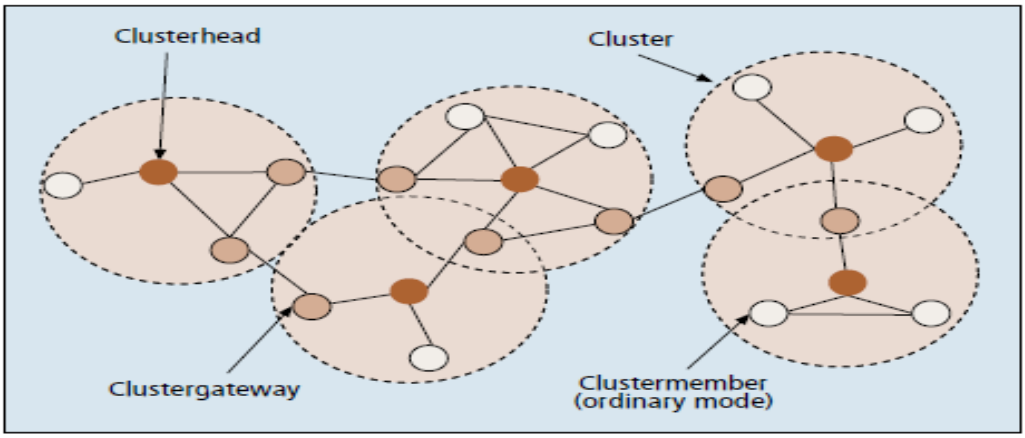
ΚΕΦΑΛΑΙΟ 2

Εισαγωγή

Η ομαδοποίηση είναι ένα σημαντικό ερευνητικό θέμα για τα M.A.NETs, επειδή μπορεί να εγγυηθεί τα θεμελιώδη επίπεδα της απόδοσης των συστημάτων, όπως το throughput και την καθυστέρηση, με την παρουσία της κινητικότητας και ενός μεγάλου αριθμού κινητών τερματικών. Έχει παρουσιαστεί ένας μεγάλος αριθμός τεχνικών ομαδοποίησης ad-hoc δικτύων που εστιάζουν σε διαφορετικούς όρους απόδοσης.

2.1. Ομαδοποίηση ασύρματων δικτύων

Σε μια τεχνική ομαδοποίησης οι κινητοί κόμβοι σε ένα M.A.NET. διαιρούνται σε διαφορετικές εικονικές ομάδες, και διατίθενται γεωγραφικά παρακείμενοι στην ίδια συστάδα, σύμφωνα με κανόνες διαφορετικών συμπεριφορών για τους κόμβους που περιλαμβάνονται σε μια συστάδα, από εκείνους που αποκλείονται από αυτήν. Μια χαρακτηριστική δομή συστάδων παρουσιάζεται στην Εικόνα 5. Φαίνεται ότι οι κόμβοι διαιρούνται σε διάφορες εικονικές ομάδες (με τις διακεκομμένες γραμμές) βασισμένοι σε ορισμένους κανόνες. Σε μια δομή συστάδας, οι κινητοί κόμβοι μπορούν να οριστούν ανάλογα με τη θέση ή τη λειτουργία, όπως ¨αρχηγός της συστάδας¨ (clusterhead), ¨πύλη εξόδου της συστάδας¨ (|clustergateway), ή ¨μέλος της συστάδας¨ (clustermember). Ο clusterhead χρησιμεύει ως ο τοπικός συντονιστής για τη συστάδα του, που ρυθμίζει την εσωτερική κίνηση των συστάδων, τα δεδομένα που διαβιβάζονται κ.α. Ο clustergateway είναι ο κόμβος που συνδέεται με άλλες συστάδες, έτσι ώστε να μπορεί να έχει πρόσβαση στις άλλες γειτονικές συστάδες και στις πληροφορίες που ανταλλάσσονται μεταξύ αυτών. Ο clustermember είναι συνήθως ένας κοινός κόμβος χωρίς οποιεσδήποτε συνδέσεις με άλλες συστάδες.



Εικόνα 5 Δομή των Clusters

2.2. Ομαδοποίηση ασύρματων δικτύων

Η αρχιτεκτονική συστάδων εγγυάται το θεμελιώδες επίτευγμα απόδοσης σε ένα M.A.NET. με ένα μεγάλο αριθμό κινητών τερματικών. Μια δομή συστάδων παρέχει τουλάχιστον τρία πλεονεκτήματα. Κατ' αρχάς, μια δομή συστάδων διευκολύνει τη χωρική επαναχρησιμοποίηση των πόρων για να αυξήσει τη χωρητικότητα των συστημάτων. Με τη μη υπέρθετη δομή πολλαπλής συστάδας, δύο συστάδες μπορούν να αναπτύξουν την ίδια συχνότητα ή τον ίδιο κώδικα εάν δεν είναι γειτονικές συστάδες. Επίσης, μια συστάδα μπορεί καλύτερα να συντονίσει τα γεγονότα μετάδοσής της με τη βοήθεια ενός ειδικού κινητού κόμβου, όπως ενός clusterhead, που κατοικεί σε αυτό. Αυτό μπορεί να εξοικονομήσει πολλούς πόρους που χρησιμοποιούνται για την αναμετάδοση, ως αποτέλεσμα της μειωμένης σύγκρουσης στην μετάδοση τους. Το δεύτερο όφελος είναι στη δρομολόγηση. Το σύνολο των clusterheads και των clustergateways μπορεί να αναπτύξει ένα εικονικό ¨backbone¨ για τη δρομολόγηση των συστάδων, και έτσι η παραγωγή και η διάδοση της δρομολόγησης των πληροφοριών μπορούν να περιοριστούν σε αυτό το σύνολο κόμβων. Τέλος, μια δομή συστάδων προκαλεί ένα ad-hoc δίκτυο να εμφανίζεται μικρότερο και πιο σταθερό. Όταν ένας κινητός κόμβος αλλάζει τη συνδεδεμένη συστάδα του, μόνο οι κινητοί κόμβοι που κατοικούν στις αντίστοιχες συστάδες πρέπει να το πληροφορηθούν. Κατά συνέπεια, οι τοπικές αλλαγές δεν χρειάζονται να εξεταστούν και να ενημερωθούν από ολόκληρο το δίκτυο και οι πληροφορίες που υποβάλλονται σε επεξεργασία και αποθηκεύονται από κάθε κινητό κόμβο μειώνονται κατά πολύ.

2.3 Κόστος ομαδοποίησης ασύρματων δικτύων

Η ομαδοποίηση είναι σημαντική για ένα δίκτυο για την επιτυχία της εξελιξιμότητας, στην παρουσία ενός μεγάλου αριθμού κινητών κόμβων και υψηλής κινητικότητας. Εντούτοις, ένα M.A.NET. με συστάδες έχει τις παρενέργειες και τα μειονεκτήματά του επειδή η κατασκευή και η διατήρηση μιας δομής συστάδων απαιτούν συνήθως επιπλέον κόστος έναντι ενός επίπεδου M.A.NET. Το κόστος είναι βασικό ζήτημα για να επικυρώσει την αύξηση της αποτελεσματικότητας και της εξελιξιμότητας μιας δομής συστάδων. Με το να αναλύσουν το κόστος μιας τεχνικής ομαδοποίησης σε διαφορετικές όψεις ποιοτικά ή ποσοτικά, η χρησιμότητα και τα μειονεκτήματά της μπορούν σαφώς να διευκρινιστούν. Οι όροι του κόστους της ομαδοποίησης περιγράφονται ως εξής:

1. Για να διατηρηθεί μια δομή συστάδων σε ένα δυναμικά μεταβαλλόμενο σενάριο απαιτείται συχνά ρητή ανταλλαγή μηνυμάτων μεταξύ των ζευγαριών των κινητών κόμβων. Όταν η τοπολογία δικτύου αλλάζει γρήγορα και περιλαμβάνει πολλούς κινητούς κόμβους, η ανταλλαγή της πληροφορίας της ομαδοποίησης αυξάνεται δραστικά. Η συχνή ανταλλαγή πληροφοριών μπορεί να καταναλώσει το εύρος ζώνης και να μειώσει την ενέργεια των κινητών κόμβων γρήγορα, έτσι ώστε οι εφαρμογές των «ανώτερων στρωμάτων» (upper-layers) να μην μπορούν να εφαρμοστούν, λόγω της ανεπάρκειας των διαθέσιμων πόρων ή της έλλειψης υποστήριξης από τους σχετικούς κινητούς κόμβους.

2. Μερικές τεχνικές ομαδοποίησης μπορούν να αναγκάσουν τη δομή συστάδων να ανοικοδομηθεί εντελώς σε ολόκληρο το δίκτυο, όταν πραγματοποιηθούν μερικά τοπικά γεγονότα, όπως π.χ. η μετακίνηση ή η απενεργοποίηση ενός κινητού κόμβου, με συνέπεια κάποια επανεκλογή clusterhead (re-clustering). Αυτή η διαδικασία ονομάζεται «ripple effect of re-clustering». Αυτή η επανεκλογή ενός clusterhead μπορεί να έχει επιπτώσεις στη δομή πολλών συστάδων με αποτέλεσμα να ξεκινήσει την επανεκλογή clusterhead στο δίκτυο. Κατά συνέπεια, αυτό μπορεί πολύ να έχει επιπτώσεις στην απόδοση των upper-layers πρωτοκόλλων.

3. Επιπλέον, οι περισσότερες τεχνικές χωρίζουν την ομαδοποίηση σε δύο φάσεις, το σχηματισμό και τη συντήρηση και υποθέτουν ότι οι κινητοί κόμβοι είναι στατικοί, όταν ο σχηματισμός συστάδων είναι σε εξέλιξη. Αυτό συμβαίνει επειδή για τον αρχικό σχηματισμό συστάδων, αυτών των τεχνικών, ένας κινητός κόμβος μπορεί να αποφασίσει να γίνει clusterhead μόνο αφού ανταλλάξει κάποιες συγκεκριμένες πληροφορίες με τους γείτονές του και βεβαιωθεί ότι κρατά μερικές συγκεκριμένες ιδιότητες στη γειτονιά του. Σε μια «παγωμένη» περίοδο κίνησης, κάθε κινητός κόμβος μπορεί να λάβει εξακριβωμένες πληροφορίες από τους γειτονικούς κόμβους, και η αρχική δομή συστάδων μπορεί να διαμορφωθεί με μερικά συγκεκριμένα χαρακτηριστικά. Εντούτοις, αυτή η υπόθεση μπορεί να μην ισχύσει σε ένα πραγματικό σενάριο, όπου οι κινητοί κόμβοι μπορούν να κινούνται τυχαία όλη την ώρα.
4. Ένας άλλος όρος είναι ο κύκλος υπολογισμού, ο οποίος δείχνει τον αριθμό κύκλων στον οποίο μια διαδικασία σχηματισμού συστάδων μπορεί να ολοκληρωθεί. Για τις τεχνικές ομαδοποίησης που στηρίζονται σε μια «παγωμένη» περίοδο κινήσεων, ο κύκλος υπολογισμού είναι σημαντικός όρος μιας και όσους περισσότερους κύκλους απαιτεί μία τεχνική ομαδοποίησης για το σχηματισμό συστάδων του, τόσο μεγαλύτερη «παγωμένη» περίοδος απαιτείται για τους κινητούς κόμβους. Στην πραγματικότητα, η τοπολογία ενός M.A.NET. αλλάζει συχνά με τη μετακίνηση των κινητών κόμβων. Για τις περισσότερες τεχνικές ομαδοποίησης, η διαδικασία σχηματισμού συστάδων τους μπορεί να εκτελεσθεί παράλληλα σε ολόκληρο το δίκτυο και θα οδηγήσει στη γρήγορη χρονικά σύγκλιση σχηματισμού συστάδων. Αλλά σε αυτές τις τεχνικές, δεν μπορούν όλοι οι κινητοί κόμβοι να αποφασίσουν τη θέση τους συγχρόνως μέσα σε έναν κύκλο, και μπορεί να απαιτήσουν έναν μεταβαλλόμενο αριθμό κύκλων για να τελειώσουν την αρχική κατασκευή συστάδων. Κατά συνέπεια, ο χρόνος που απαιτείται για αυτούς τους αλγορίθμους δεν μπορεί να είναι οριακός και μπορεί να ποικίλει για τις διάφορες τοπολογίες δικτύων.

Ως εκ τούτου, η απαραίτητη ρητή ανταλλαγή μηνυμάτων ελέγχου, η ¨ripple effect of re-clustering¨, και η στάσιμη υπόθεση για το σχηματισμό συστάδων είναι τα κύρια κόστη ενός ομαδοποιημένου M.A.NET. έναντι ενός επίπεδου M.A.NET.

2.4. Αλγόριθμοι ομαδοποίησης ασύρματων δικτύων

Στο πλαίσιο της διαχείρισης της ιεραρχικής τοπολογίας, ένα υποσύνολο των κόμβων του δικτύου επιλέγεται για να χρησιμεύσει ως το ¨backbone¨ των δικτύων πέρα από το οποίο, υποστηρίζονται οι ουσιαστικές λειτουργίες ελέγχου των δικτύων. Η διαχείριση της ιεραρχικής τοπολογίας καλείται συχνά ομαδοποίηση, και αποτελείται επιλέγοντας ένα σύνολο clusterheads με τέτοιο τρόπο ώστε ότι ο κάθε κόμβος να συνδέεται με ένα clusterhead, και τα clusterheads να συνδέονται το ένα με το άλλο με τη βοήθεια των clustergateways. Μόλις ολοκληρωθεί η εκλογή, ο συνηθισμένος κόμβος μπορεί να κλείσει την ασύρματη εκπομπή ή λήψη προκειμένου να διατηρήσουν ενέργεια. Το «κλειδί» για την διαχείριση της ιεραρχικής τοπολογίας είναι εκλογή του clusterhead σε μια ομάδα. Διάφοροι αλγόριθμοι έχουν προταθεί για να επιλέξουν τους clusterheads στα M.A.NETs.

2.4.1. Αλγόριθμος Max-Degree

Ο αλγόριθμος Max-Degree είναι ένας αλγόριθμος που χρησιμοποιείται συχνά, και στον οποίο οι κόμβοι με τον υψηλότερο βαθμό είναι πιθανότερο να γίνουν clusterheads. Οι γείτονες ενός clusterhead γίνονται μέλη εκείνης της συστάδας και δεν μπορούν πλέον να συμμετέχουν στη εκλογική διαδικασία. Πειράματα καταδεικνύουν ότι το σύστημα έχει ένα χαμηλό ποσοστό αλλαγών clusterhead, αλλά επειδή η τοπολογία δικτύων αλλάζει αυτήν την προσέγγιση μπορεί να οδηγήσει σε έναν υψηλό κύκλο εργασιών των clusterheads.

2.4.2. Αλγόριθμος Lowest-I.D.

Διάφορες προσεγγίσεις χρησιμοποιούν τα προσδιοριστικά των κόμβων για να εκλέξουν τον clusterhead με ένα ή πολλαπλάσια άλματα. Αυτό το είδος αλγορίθμου ορίζει μια μοναδική ταυτότητα (I.D.) σε κάθε κόμβο και επιλέγει τον κόμβο με την ελάχιστη I.D. για clusterhead . Το μειονέκτημα αυτού του είδους αλγορίθμου είναι η προκατάληψή του προς εκείνους τους κόμβους με το μικρότερο προσδιοριστικό.

2.4.3. Αλγόριθμος Node-Weight

Ο Basagni έχει εισαγάγει δύο αλγορίθμους, που ονομάζονται Distributed Clustering Algorithms (D.C.A.) και Distributed Mobility Adaptive Clustering (D.M.A.C.). Ένας κόμβος επιλέγεται να είναι clusterhead εάν το βάρος του είναι περισσότερο από οποιαδήποτε βάρος των γειτόνων του. Και οι δύο αλγόριθμοι εκτελούνται σε κάθε κόμβο με μοναδική γνώση την ταυτότητα των γειτονικών κόμβων του.

Κανένα από το παραπάνω τρία είδη αλγορίθμων δεν οδηγεί σε μια βέλτιστη εκλογή των clusterheads δεδομένου ότι κάθε ένας εξετάζει μόνο ένα υποσύνολο των παραμέτρων που ενδεχομένως μπορεί να επιβάλει περιορισμούς στο σύστημα. Κάθε ένας από αυτούς τους αλγορίθμους είναι κατάλληλος για μια συγκεκριμένη εφαρμογή των M.A.NETs. Το πιο σημαντικό είναι ότι αυτοί οι αλγόριθμοι δεν ζητούν από τους κανονικούς κόμβους να κλείσουν την ασύρματη εκπομπή ή λήψη προκειμένου να διατηρήσουν ενέργεια.

ΚΕΦΑΛΑΙΟ 3

Εισαγωγή

Τα τελευταία χρόνια με την εκθετική αύξηση του Internet, όσον αφορά τα συστήματα που συνδέονται σε αυτό και τις συνεχώς αναπτυσσόμενες εφαρμογές και δικτυακές υπηρεσίες, έχει αυξηθεί και το πλήθος των κακόβουλων χρηστών, οι οποίοι υλοποιούν ολοένα και πιο έξυπνες, πολύπλοκες και επιζήμιες δικτυακές επιθέσεις.

Με δεδομένη την εξέλιξη αυτή, τα κλασσικά μέτρα ασφάλειας δεν φαίνεται να επαρκούν για την προστασία των συστημάτων και των πληροφοριών που περιέχουν αυτά και συνεχώς γίνεται προσπάθεια για ανάπτυξη νέων μηχανισμών ασφάλειας, που θα παρέχουν την επιθυμητή προστασία από δικτυακές επιθέσεις. Μία σχετικά νέα και συνεχώς αναπτυσσόμενη μέθοδος προστασίας, είναι η αυτοματοποιημένη Ανίχνευση Επιθέσεων (Intrusion Detection)

3.1 Συστήματα ανίχνευσης επιθέσεων

Ο όρος Intrusion Detection σημαίνει Ανίχνευση Επιθέσεων και παρακολουθεί τα γεγονότα που συμβαίνουν σε ένα σύστημα ή ένα δίκτυο και τα αναλύει για σημάδια επιθέσεων. Επίθεση χαρακτηρίζεται ως οποιαδήποτε προσπάθεια για παραβίαση της εμπιστευτικότητας, της ακεραιότητας, της διαθεσιμότητας των μηχανισμών ασφάλειας ενός συστήματος ή ενός δικτύου. Οι επιθέσεις πραγματοποιούνται από άτομα που έχουν πρόσβαση στους στόχους τους μέσω του Internet, από χρήστες που προσπαθούν να αποκτήσουν περισσότερα δικαιώματα από αυτά που τους έχουν δοθεί και από εξουσιοδοτημένους χρήστες, οι οποίοι εκμεταλλεύονται τα δικαιώματα που τους έχουν δοθεί με κακό σκοπό. Ο όρος Intrusion Detection Systems (I.D.S.) σημαίνει Συστήματα Ανίχνευσης Επιθέσεων, δηλαδή Λογισμικού (Software) ή Υλικού (Hardware) προϊόντα, που αυτοματοποιούν τη διαδικασία παρακολούθησης και ανάλυσης. Η εξέλιξη των I.D.S., είναι ραγδαία τα τελευταία χρόνια και συνεχώς γίνονται προσπάθειες για βελτίωσή τους, κυρίως στον

τομέα των συμπτωμάτων από καταστάσεις «False Positives» και «False Negatives» που παρουσιάζουν.

Με την τρέχουσα μορφή τους τα I.D.S. παρέχουν σημαντική υποστήριξη στα ήδη υπάρχοντα μέτρα προστασίας ενός δικτύου και σε συνδυασμό με άλλους μηχανισμούς ασφάλειας, αποτελούν ένα σημαντικό εργαλείο για την παρακολούθηση και την αποτροπή δικτυακών επιθέσεων.

3.2. Χρησιμότητα συστημάτων ανίχνευσης επιθέσεων

Καθώς οι δικτυακές επιθέσεις έχουν αυξηθεί κατά πολύ τα τελευταία χρόνια τόσο σε πλήθος όσο και σε βαθμό επικινδυνότητας, τα I.D.S. αποτελούν μία απαραίτητη προσθήκη στην πολιτική ασφάλειας κάθε οργανισμού. Η Ανίχνευση Επιθέσεων επιτρέπει στους οργανισμούς να προστατέψουν τα συστήματά τους και τις πληροφορίες που βρίσκονται σε αυτά, από κινδύνους που προκύπτουν από την αυξημένη δικτυακή διασύνδεση μεταξύ των συστημάτων τους.

Υπάρχουν διάφοροι λόγοι για τους οποίους είναι απαραίτητη η χρήση των I.D.S. όπως :

1. Για ανίχνευση επιθέσεων και άλλων παραβιάσεων ασφάλειας που δε ανιχνεύονται από άλλα μέτρα προστασίας. Ο επιτιθέμενος μπορεί να αποκτήσει πρόσβαση σε ένα ή περισσότερα συστήματα, όταν διάφορες, γνωστές αδυναμίες ασφάλειας των συστημάτων αυτών δεν έχουν διορθωθεί. Παρόλο που κάθε διαχειριστής (administrator) πρέπει και μπορεί σχετικά εύκολα να διορθώνει τις αδυναμίες αυτές, υπάρχουν διάφοροι λόγοι για τους οποίους αυτό δεν συμβαίνει.
    - Σε περιβάλλοντα με πολλά συστήματα, ο administrator συνήθως δεν έχει την δυνατότητα αλλά ούτε και το χρόνο να ενημερώσει τα συστήματα που πρέπει με νέες διορθώσεις των αδυναμιών ασφάλειάς τους.
    - Οι χρήστες των συστημάτων κάνουν χρήση διάφορων λογισμικών που θεωρούνται επικίνδυνα, με την έννοια ότι μπορούν να προκαλέσουν τρύπες ασφάλειας σε ένα σύστημα.
    - Τόσο οι administrators όσο και οι χρήστες κάνουν λάθη στη ρύθμιση και τη χρήση των συστημάτων και των υπηρεσιών που προσφέρουν.

- Οι χρήστες χρησιμοποιούν μειωμένης ασφάλειας μηχανισμούς πρόσβασης στα συστήματα, όπως π.χ. ατυχώς επιλεγμένα passwords.

Σε έναν ιδανικό κόσμο οι δημιουργοί λογισμικού θα μείωναν στο ελάχιστο τις αδυναμίες ασφάλειάς στα προϊόντα που διανέμουν και οι διαχειριστές θα ενημέρωναν και θα διόρθωναν τα συστήματά τους γρήγορα και αξιόπιστα. Στον πραγματικό όμως κόσμο αυτό σπάνια συμβαίνει, ενώ νέες αδυναμίες και ελαττώματα στην ασφάλεια συστημάτων εμφανίζονται σε καθημερινή βάση. Με την χρήση ενός I.D.S. η προσπάθεια ή η επιτυχία ενός επιτιθέμενου να παραβιάσει κάποιο σύστημα μέσω της εκμετάλλευσης μιας γνωστής αδυναμίας σε αυτό, θα γινόταν αντιληπτή. Επίσης με την βοήθεια του I.D.S, γνωστοποιείται η αδυναμία που οδήγησε στην παραβίαση του συστήματος και παράγονται χρήσιμα συμπεράσματα που βοηθούν στην αποκατάσταση του συστήματος και τη διόρθωση της αδυναμίας, που οδήγησε στην παραβίασή του.

2. Για την ανίχνευση αναγνωριστικών ενεργειών που προηγούνται μίας επίθεσης. Για την πραγματοποίηση μίας επίθεσης συνήθως υπάρχουν κάποια στάδια που προηγούνται αυτής. Ο επιτιθέμενος πρώτα εξετάζει τον υποψήφιο στόχο του, ώστε να συγκεντρώσει πληροφορίες για αυτόν και να εντοπίσει ένα σημείο εσόδου, το οποίο θα του επιτρέψει να πραγματοποιήσει την επίθεση με επιτυχία. Αυτό επιτυγχάνεται μέσω του «Scanning». Δίχως την ύπαρξη ενός I.D.S, ο επιτιθέμενος είναι πολύ πιθανό να πραγματοποιήσει τις αναγνωριστικές του κινήσεις ανενόχλητος και χωρίς να γίνει αντιληπτός. Ένα I.D.S. θα είχε τη δυνατότητα να εντοπίσει τις κινήσεις αυτές του επιτιθέμενου και να πάρει κάποια μέτρα, όπως να καταγράψει το γεγονός, να ειδοποιήσει τους υπεύθυνους ασφάλειας και να εμποδίσει τον επιτιθέμενο να τις ολοκληρώσει.
3. Για τη συγκέντρωση πληροφοριών που αφορούν επιθέσεις που πραγματοποιήθηκαν, οι οποίες θα βοηθήσουν στην αποκατάσταση των συστημάτων που παραβιάστηκαν και στη διόρθωση αδυναμιών και παραλήψεων στα ήδη υπάρχοντα μέτρα ασφάλειας. Ακόμα και στην περίπτωση που ένα I.D.S. δεν μπορεί να εμποδίσει μία επίθεση, μπορεί να συλλέξει διάφορες πληροφορίες και στοιχεία για αυτήν που θα χρησιμοποιηθούν τόσο για την αποκατάσταση του συστήματος και την

διόρθωση των αδυναμιών ασφάλειάς του, όσο και για τον εντοπισμό του επιτιθέμενου και την ποινική δίωξή του.

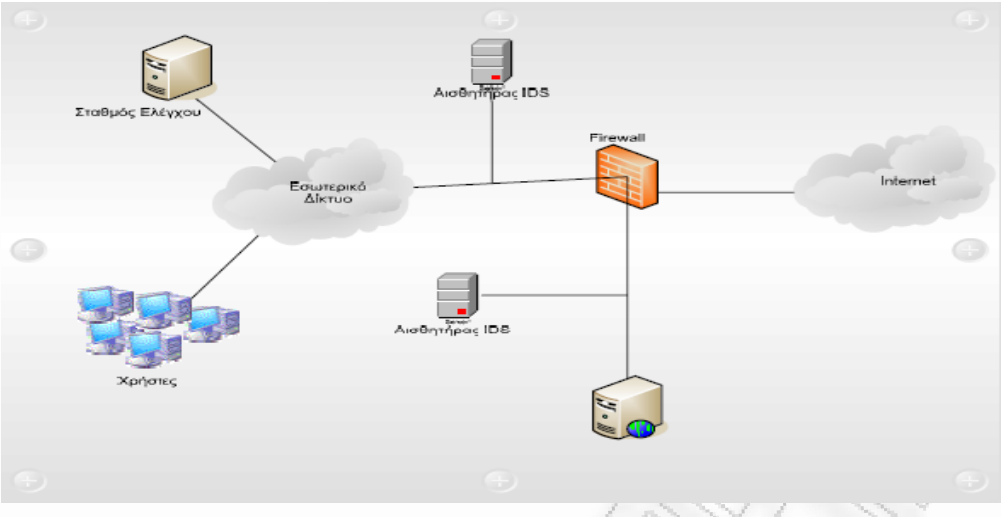
4. Για να αποτραπούν επίδοξοι επιτιθέμενοι, καθώς υπάρχει μεγαλύτερο ρίσκο να εντοπιστούν και να τιμωρηθούν. Όταν ο υποψήφιος επιτιθέμενος συνειδητοποιήσει ότι ένα δίκτυο ή ένα σύστημα προστατεύεται από ένα I.D.S., διστάζει να συνεχίσει την προσπάθειά του καθώς υπάρχουν περισσότερες πιθανότητες να γίνει αντιληπτός και να συλληφθεί.
5. Για αποτελεσματικότερη σχεδίαση και εφαρμογή πολιτικής ασφάλειας. Με την χρήση των I.D.S. συλλέγονται πληροφορίες και παρατηρούνται «patterns» από ενέργειες που πραγματοποιούνται καθημερινά εναντίον ενός δικτύου και των συστημάτων του, τα οποία μπορούν να βοηθήσουν στη σχεδίαση πιο αξιόπιστων μέτρων ασφάλειας, προσαρμοσμένων ώστε να αντιμετωπίζουν τα γεγονότα και τους κινδύνους που απειλούν το συγκεκριμένο δίκτυο, και να οδηγήσουν στην αποτελεσματικότερη προστασία του.

3.3. Κατηγορίες συστημάτων ανίχνευσης επιθέσεων

Τα συστήματα ανίχνευσης επιθέσεων συναντώνται σε δύο τύπους:

1. N.I.D.S. (Network-based I.D.S.)
2. H.I.D.S. (Host-based I.D.S.)

Τα N.I.D.S. είναι τα πιο διαδεδομένα και εξετάζουν τη διερχόμενη δικτυακή κίνηση (traffic) για ίχνη εισβολής. Στην Εικόνα 1 παρουσιάζεται η διάταξη ενός N.I.D.S. στην οποία περιέχονται δύο αισθητήρες σε διαφορετικά σημεία ο καθένας, που επικοινωνούν με ένα σταθμό παρακολούθησης δεδομένων στο εσωτερικό δίκτυο.



Εικόνα 6 Διάταξη N.I.D.S.

Τα H.I.D.S. παρακολουθούν τη δραστηριότητα χρηστών και εφαρμογών στο τοπικό μηχάνημα για ίχνη εισβολής. Αυτού του είδους τα I.D.S. παρέχουν πιο ακριβή πληροφορία για την ύπαρξη ή μη κάποιας επίθεσης και αυτό γιατί μπορούν να καταλάβουν τι συμβαίνει κάθε φορά στο σύστημα. Έτσι, αν συμβεί μια άγνωστης μορφής επίθεση κατά την οποία γίνεται προσπάθεια να επιτευχθεί η δυσλειτουργία του υπολογιστή, το H.I.D.S. θα το αναγνωρίσει ως επίθεση, ενώ αντίθετα το N.I.D.S. δεν θα αντιληφθεί την επίθεση. Γενικά θεωρείται πως η χρήση H.I.D.S. έχει καλύτερα αποτελέσματα από τη χρήση N.I.D.S.

Υπάρχουν τρία είδη μηχανισμών με τους οποίους μπορεί κάποιο I.D.S. να αποφασίσει αν υπάρχει επίθεση ή όχι:

1. Ανάλυση με βάση τα γεγονότα ή υπογραφές (events ή signatures)
2. Στατιστική ανάλυση
3. Προσαρμόσιμα συστήματα

Τα συστήματα που βασίζονται σε events ή signatures λειτουργούν με τρόπο αντίστοιχο με τον τρόπο λειτουργίας των antivirus προγραμμάτων. Η εταιρεία ανάπτυξης του συστήματος I.D.S. παράγει κάθε φορά μια λίστα από «Signatures», δηλαδή μια λίστα με ακολουθίες ενεργειών που θεωρεί ύποπτες ή ενδεικτικές επίθεσης. Το I.D.S. ερευνά και αναλύει το περιβάλλον ελέγχοντας για γνωστές ακολουθίες ενεργειών. Μόλις ανιχνευτεί μια τέτοια ακολουθία, τότε το I.D.S. ενημερώνει τον σταθμό ελέγχου για το συμβάν.

Τα συστήματα που βασίζονται στη στατιστική ανάλυση κατασκευάζουν στατιστικά πρότυπα για το περιβάλλον τους. Τέτοια πρότυπα μπορεί να είναι η μέση

διάρκεια μιας συνόδου, ο μέσος αριθμός βημάτων περιήγησης σε ένα web site, η μέση συχνότητα εμφάνισης μιας I.P. διεύθυνσης κ.αλ. Τα πρότυπα αυτά καθορίζουν αυτό που αποκαλείται «Συνήθης Συμπεριφορά» και κάθε απόκλιση από αυτήν θεωρείται ως ένδειξη περίεργης συμπεριφοράς. Τα συστήματα ξεκινούν από γενικούς κανόνες για το περιβάλλον και στη συνέχεια μαθαίνουν ή προσαρμόζονται σε τοπικές καταστάσεις που διαφορετικά θα τις θεωρούσαν ασυνήθιστες. Μετά από την αρχική περίοδο μάθησης, το σύστημα καταλαβαίνει την αλληλεπίδραση ανθρώπων – περιβάλλοντος και προειδοποιεί τους υπεύθυνους για ασυνήθιστες δραστηριότητες. Η έρευνα σε αυτόν τον τομέα συνεχίζεται διαρκώς.

Οποιοδήποτε I.D.S. θα απολέσει κάποιες πληροφορίες όταν υπάρχει ύποπτη δραστηριότητα και θα έχει ενδείξεις κινδύνου ακόμη και όταν όλα είναι φυσιολογικά. Οι δύο αυτές καταστάσεις ονομάζονται «false negatives» και «false positives» αντίστοιχα. Για αυτό δε θα πρέπει να υποτιμούμε και να αγνοούμε τον ανθρώπινο παράγοντα, η ύπαρξη του οποίου θα βελτιώσει περισσότερο την αλληλεπίδραση του I.D.S. με το περιβάλλον. Η τεχνητή νοημοσύνη των μηχανών στα συστήματα ανίχνευσης επιθέσεων εξελίσσεται με την διαρκή έρευνα.

3.3.1. Network I.D.Ss

Το N.I.D.S. συνήθως αποτελείται από δύο μέρη: τους αισθητήρες και το σταθμό διαχείρισης - ανάλυσης. Ο αισθητήρας βρίσκεται σε ένα τομέα του δικτύου και παρακολουθεί για ύποπτη κίνηση. Ο σταθμός διαχείρισης λαμβάνει τις ενδείξεις κινδύνου από τους αισθητήρες και τις μεταβιβάζει στον διαχειριστή του συστήματος, δηλαδή στον διαχειριστή ασφαλείας του δικτύου. Οι αισθητήρες είναι συνήθως συστήματα που υπάρχουν μόνο για να παρακολουθούν το δίκτυο. Έχουν ένα δικτυακό «interface» που αναλύει τα πάντα, δηλαδή λαμβάνουν όλη τη δικτυακή κίνηση, όχι μόνο ότι προορίζεται για τη δικιά τους I.P. διεύθυνση, αλλά και το διερχόμενο από αυτούς «traffic» με σκοπό την περαιτέρω ανάλυση. Αν ανιχνεύσουν κάτι ύποπτο το μεταβιβάζουν στον σταθμό διαχείρισης - ανάλυσης. Ο σταθμός αυτός μπορεί να δείξει τα σήματα κινδύνου, που έλαβε από τους αισθητήρες ή να πραγματοποιήσει επιπλέον ανάλυση.

Τα N.I.D.S. έχουν αρκετά πλεονεκτήματα και μειονεκτήματα, τα οποία αναφέρονται παρακάτω :

Τα πλεονεκτήματα των N.I.D.S. :

1. Τα N.I.D.S. μπορούν να ανιχνεύσουν κάποιες από τις επιθέσεις που χρησιμοποιούν το δίκτυο.
2. Τα N.I.D.S. έχουν την τάση να είναι καλύτερα αυτοδιατηρούμενα από ότι τα H.I.D.S. Τρέχουν σε ένα συγκεκριμένο σύστημα και η εγκατάστασή τους είναι απλή και πραγματοποιείται σε μια τοποθεσία στο δίκτυο που δίνει τη δυνατότητα παρακολούθησης ευαίσθητης κίνησης δεδομένων, χωρίς εξουσιοδότηση ή κάποιων ειδών πρόσβασης με κατάχρηση προνομίων εξουσιοδότησης
3. Ένα N.I.D.S. δεν απαιτεί μετατροπές στους servers μιας επιχείρησης ή στους hosts για να εγκατασταθεί. Αυτό είναι μεγάλο όφελος, γιατί συνήθως οι servers έχουν κλειστές ανοχές όσο αφορά τη C.P.U., το I/O και τη χωρητικότητα του δίσκου. Η εγκατάσταση επιπλέον λογισμικού ίσως να δημιουργήσει προβλήματα λειτουργικότητας.
4. Το N.I.D.S. δεν αποτελεί κρίσιμο παράγοντα για την λειτουργικότητα του δικτύου, και αυτό γιατί δεν λειτουργεί ως δρομολογητής ή ως κάποια άλλη κρίσιμη συσκευή. Άρα, τυχόν αποτυχία στο σύστημα του I.D.S. δε θα έχει σημαντική επίδραση στην επιχείρηση. Ένα επιπλέον όφελος είναι ότι πιθανότατα θα συναντήσουμε λιγότερη αντίδραση από ανθρώπους εντός του εργασιακού περιβάλλοντος καθώς δεν θα απαιτηθεί να εγκαταστήσουν αυτοί κάτι στα συστήματά τους.

Τα μειονεκτήματα των N.I.D.S. :

1. Ένα N.I.D.S. απλά εξετάζει τη δικτυακή σύνδεση στον τομέα που είναι συνδεδεμένο και μόνο. Δεν μπορεί να ανιχνεύσει μία επίθεση που γίνεται σε διαφορετικό τμήμα του δικτύου. Το πρόβλημα αυτό γίνεται μεγαλύτερο σε ένα περιβάλλον με πολλαπλές δικτυώσεις. Για να καλύψει τις ανάγκες του σε δικτυακή κάλυψη, ένας μεγάλος οργανισμός θα πρέπει να αγοράσει πολλούς αισθητήρες κάτι που σημαίνει επιπλέον κόστος.
2. Τα N.I.D.S. συνήθως χρησιμοποιούν ανάλυση signatures για να καλύψουν τις προδιαγραφές απόδοσης. Έτσι ανιχνεύονται κοινές προγραμματισμένες επιθέσεις από εξωτερικές πηγές, αλλά αυτή η μέθοδος δεν είναι επαρκής για

πιο πολύπλοκα είδη επιθέσεων. Αυτές απαιτούν καλύτερη ικανότητα για ανάλυση του περιβάλλοντος.

3. Ένα N.I.D.S. μπορεί να χρειαστεί να μεταδώσει μεγάλες ποσότητες δεδομένων στο κεντρικό σύστημα ανάλυσης. Πολλά τέτοια συστήματα χρησιμοποιούν επιθετικές μεθόδους ελάττωσης δεδομένων για να μειώσουν το παραγόμενο «traffic» επικοινωνίας. Επίσης, προωθούν αρκετές από τις διαδικασίες επιλογής ενέργειας μόνο στον αισθητήρα και χρησιμοποιούν το σύστημα ανάλυσης ως οθόνη της κατάστασης του δικτύου ή ως κέντρο επικοινωνίας, παρά για πραγματική ανάλυση. Το μειονέκτημα εδώ είναι ότι παρέχεται ελάχιστος συντονισμός μεταξύ των αισθητήρων, δηλαδή οποιοσδήποτε αισθητήρας δεν γνωρίζει αν κάποιος άλλος έχει ανιχνεύσει μια επίθεση. Ένα τέτοιο σύστημα δεν μπορεί συνήθως να ανιχνεύσει συνεργατικές ή πολύπλοκες επιθέσεις.
4. Ένα N.I.D.S. πιθανόν να αντιμετωπίσει δυσκολίες στο χειρισμό επιθέσεων στη διάρκεια κρυπτογραφημένων συνόδων. Ευτυχώς, είναι πολύ λίγες οι επιθέσεις που πραγματοποιούνται εντός μιας κρυπτογραφημένης συνόδου, εκτός από τις επιθέσεις εναντίον ευπαθών Web Servers. Αυτό το γεγονός θα γίνει περισσότερο εμφανές με την μετάβαση στο I.P.v.6.

3.3.2. Host I.D.S.

Τα H.I.D.S. ψάχνουν για ίχνη εισβολής στο τοπικό σύστημα του host. Χρησιμοποιούν συχνά το μηχανισμό ελέγχου και καταγραφής του host σαν πηγή πληροφοριών για ανάλυση. Πιο συγκεκριμένα ψάχνουν για ασυνήθη δραστηριότητα που περιορίζεται στον τοπικό host, όπως «logins», παράξενη πρόσβαση σε αρχεία, μη εγκεκριμένη αύξηση δικαιωμάτων ή μετατροπές σε δικαιώματα του συστήματος. Η συγκεκριμένη αρχιτεκτονική χρησιμοποιεί μηχανισμούς βασισμένους σε κανόνες για την ανάλυση της δραστηριότητας.

Τα H.I.D.S. έχουν αρκετά πλεονεκτήματα και μειονεκτήματα, τα οποία αναφέρονται παρακάτω :

Τα πλεονεκτήματα των H.I.D.S. :

1. Ένα H.I.D.S. μπορεί να αποτελέσει πολύ δυνατό εργαλείο ανάλυσης πιθανών επιθέσεων. Για παράδειγμα, είναι σε θέση μερικές φορές να πει τι ακριβώς

έκανε ο εισβολέας, ποιες εντολές εκτέλεσε, ποια αρχεία έτρεξε και ποιες ρουτίνες του συστήματος κάλεσε αντί για μια αόριστη υπόθεση ότι προσπάθησε να εκτελέσει μια επικίνδυνη εντολή. Άρα τα H.I.D.S. συνήθως παρέχουν πολύ πιο λεπτομερείς και σχετικές πληροφορίες από ότι τα N.I.D.S.

2. Τα H.I.D.S. έχουν μικρότερους false positive ρυθμούς από ότι τα N.I.D.S. Αυτό συμβαίνει γιατί το εύρος των εντολών που εκτελούνται σε ένα συγκεκριμένο host είναι πολύ πιο εστιασμένο, παρά τα είδη της κίνησης πακέτων που ρέουν σε ένα δίκτυο. Αυτή η ιδιότητα μπορεί να μειώσει την πολυπλοκότητα των H.I.D.S.
3. Μπορούν να χρησιμοποιηθούν σε περιβάλλοντα όπου δεν χρειάζεται πλήρης ανίχνευση εισβολών ή όταν δεν υπάρχει διαθέσιμο bandwidth για επικοινωνία του αισθητήρα με τον σταθμό ανάλυσης. Τα H.I.D.S. είναι πλήρως αυτοσυντηρούμενα, κάτι που τους επιτρέπει, σε κάποιες περιπτώσεις, να εκτελούνται από read-only μέσα. Έτσι, οι εισβολείς δύσκολα μπορούν να εξουδετερώσουν το I.D.S.
4. Σε ένα H.I.D.S. είναι ευκολότερο να σχηματιστεί μία ενεργή αντίδραση σε περίπτωση επίθεσης, όπως ο τερματισμός μιας υπηρεσίας ή το «logging off» ενός επιτιθέμενου χρήστη.

Τα μειονεκτήματα των HIDS :

1. Τα H.I.D.S. απαιτούν εγκατάσταση στο σύστημα που θέλουμε να προστατεύσουμε.
2. Τα H.I.D.S. έχουν την τάση να εξαρτώνται από το υπάρχον σύστημα καταγραφής (logging system) και ελέγχου του server. Εάν ο server δεν λειτουργεί έτσι ώστε η καταγραφή και ο έλεγχος να είναι σε ικανοποιητικό επίπεδο, θα πρέπει να γίνει αλλαγή στις ρυθμίσεις του. Αυτό αποτελεί τεράστιο πρόβλημα αλλαγής στη διαχείριση του server.
3. Τα H.I.D.S. είναι σχετικά ακριβά. Πολλοί οργανισμοί δεν έχουν την οικονομική δυνατότητα να προστατέψουν ολόκληρα δικτυακά τμήματα με τη χρήση H.I.D.S. Αντίθετα, θα πρέπει να επιλέξουν ποια συστήματα θα προστατέψουν και ποια όχι. Αυτό το γεγονός αφήνει μεγάλα κενά στην κάλυψη της ανίχνευσης εισβολών στο δίκτυο, αφού ένας εισβολέας σε ένα γειτονικό, αλλά απροστάτευτο σύστημα μπορεί να υποκλέψει πληροφορίες από το δίκτυο.

4. Τα H.I.D.S. είναι πιο ευάλωτα, σε μεγαλύτερο ακόμα βαθμό από τοπικούς περιορισμούς. Αγνοούν εντελώς το περιβάλλον του δικτύου, άρα ο χρόνος ανάλυσης που απαιτείται για την εκτίμηση ζημιών από πιθανή εισβολή αυξάνει γραμμικά με τον αριθμό των host που προστατεύονται.

3.4. Τεχνικές Ανίχνευσης Επιθέσεων

Μια άλλη κατηγοριοποίηση των I.D.S. γίνεται με βάση την τεχνική που χρησιμοποιούν για να ανιχνεύσουν τις εισβολές. Οι τεχνικές που χρησιμοποιούνται στην ανίχνευση εισβολών χωρίζονται σε δύο είδη:

1. Ανίχνευση Διαταραχών (Anomaly Detection)
2. Ανίχνευση Κακής Συμπεριφοράς (Misuse Detection)

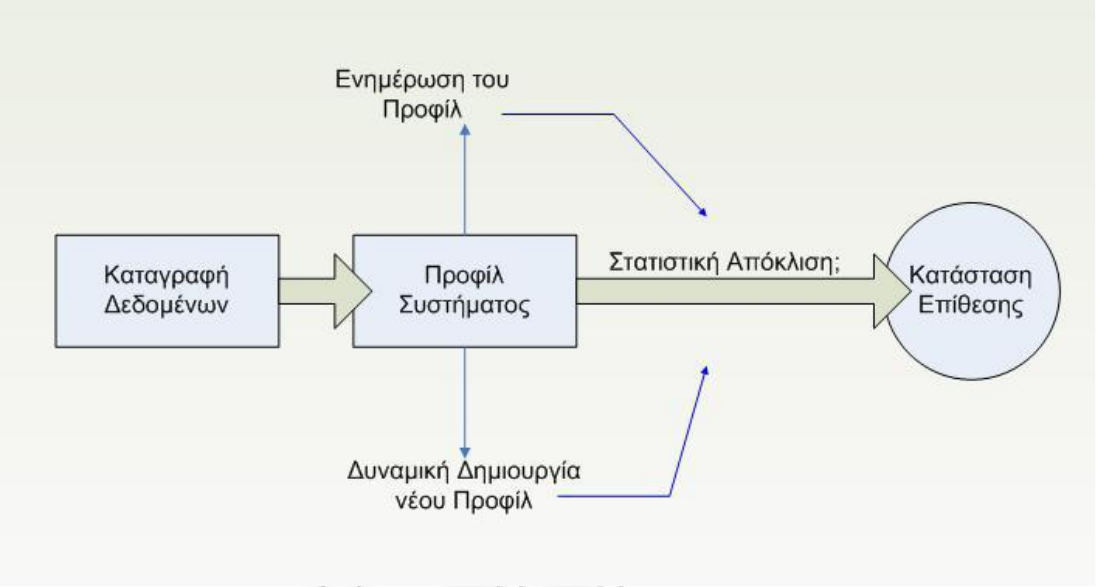
3.4.1. Τεχνική Ανίχνευσης Διαταραχών

Οι τεχνικές ανίχνευσης διαταραχών καταλήγουν στο συμπέρασμα ότι όλες οι επιθετικές δραστηριότητες είναι αναγκαστικά ανωμαλίες. Αυτό σημαίνει ότι αν μπορούσαμε να καθιερώσουμε ένα «σύνηθες προφίλ δραστηριότητας» για ένα σύστημα, θα ήμασταν σε θέση, θεωρητικά, να σημαδέψουμε όλες τις καταστάσεις του συστήματος που αποκλίνουν από το καθιερωμένο προφίλ. Αυτό θα γίνει με βάση ένα, στατιστικά, σημαντικό νούμερο προσπαθειών εισβολής. Παρόλα αυτά αν συλλογιστούμε ότι το σύνολο των επιθετικών δραστηριοτήτων αλλάζει την κατάσταση του σύνολο των δραστηριοτήτων διαταραχής παρά να το αφήνει στην αρχική μορφή, βγάζουμε κάποιες ενδιαφέρουσες εκδοχές:

• Ασυνήθιστες δραστηριότητες που δεν έχουν χαρακτήρα εισβολής χαρακτηρίζονται ως επιθετικές.

• Επιθετικές δραστηριότητες που δεν είναι ασυνήθιστες , καταλήγουν σε «false negatives» (γεγονότα που δεν χαρακτηρίζονται επιθέσεις, ενώ στην πραγματικότητα είναι).Αυτό είναι ένα ιδιαίτερα επικίνδυνο πρόβλημα και μάλιστα σοβαρότερο από το πρόβλημα των false positive.

Τα κυριότερα ζητήματα στην ανίχνευση διαταραχών σε συστήματα ανίχνευσης επιθέσεων, είναι να γίνονται οι επιλογές στα επίπεδα των ορίων, ώστε κανένα από τα δύο παραπάνω προβλήματα να μην μεγιστοποιείται. Σημαντική είναι, επίσης και η επιλογή των χαρακτηριστικών στην παρακολούθηση δεδομένων. Τα συστήματα ανίχνευσης διαταραχών είναι υπολογιστικά ακριβά, λόγω του κόστους του ελέγχου και της συνεχούς ανανέωσης (updating) των μετρικών του προφίλ ενός συστήματος. Ένα σχηματικό παράδειγμα ενός τυπικού anomaly detection συστήματος είναι το παρακάτω:

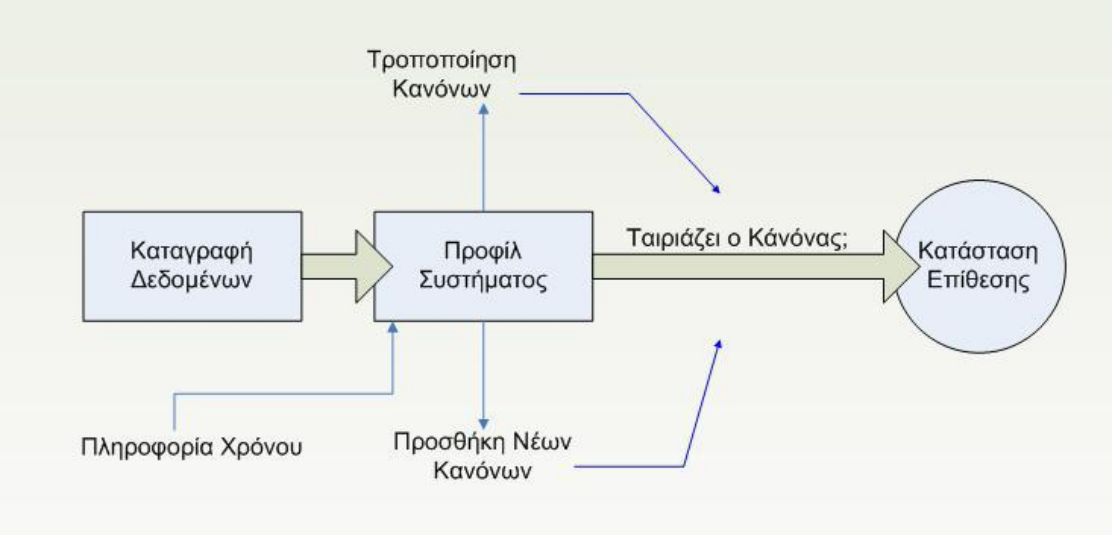


Εικόνα 7 Παράδειγμα anomaly detection συστήματος

3.5.2. **Ανίχνευση Κακής Συμπεριφοράς**

Η ιδέα πίσω από την «misuse detection» είναι ότι υπάρχουν τρόποι αναπαράστασης επιθέσεων με τη μορφή ενός προτύπου ή signature, ώστε ακόμα και οι παραλλαγές της επίθεσης να μπορούν να ανιχνευτούν. Άρα τα συστήματα αυτά που μοιάζουν πολύ με τα antivirus, μπορούν να ανιχνεύσουν πολλά ή όλα τα γνωστά πρότυπα εισβολής, αλλά δεν είναι αποτελεσματικά σε άγνωστες τεχνικές επίθεσης. Σημαντικό είναι να τονίσουμε πως τα anomaly detection συστήματα προσπαθούν να μαντέψουν το συμπλήρωμα της «κακής» συμπεριφοράς, ενώ τα misuse detection συστήματα προσπαθούν να αναγνωρίσουν γνωστές «κακές» συμπεριφορές. Το σημαντικότερο ζήτημα στα misuse detection συστήματα είναι το πώς θα δημιουργήσουμε signatures που να περιγράφουν όλες τις πιθανές παραλλαγές μιας σχετικής επίθεσης και πώς θα δημιουργήσουμε signatures που αγνοούν την μη

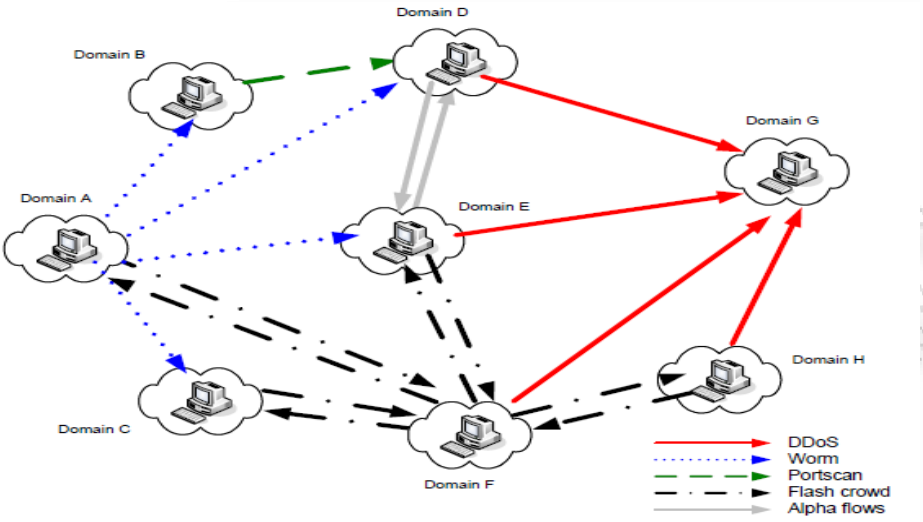
επιθετική δραστηριότητα. Ένα σχηματικό παράδειγμα ενός τυπικού misuse detection συστήματος είναι το παρακάτω:



Εικόνα 8 Παράδειγμα συστήματος ανίχνευσης «κακής συμπεριφοράς»

3.6. **Παρουσίαση των χαρακτηριστικότερων ανωμαλιών δικτύου**

Σε αυτή την ενότητα περιγράφουμε ένα ευρύ φάσμα ανωμαλιών δικτύου που προκαλούνται είτε από κακόβουλους χρήστες είτε από νόμιμα γεγονότα κίνησης δικτύου. Η ανίχνευση και η ταξινόμηση των ανωμαλιών δικτύου είναι πολύ σημαντική για τους διαχειριστές δικτύων, επειδή η κατανόηση της φύσης της ανωμαλίας θα τους βοηθήσει να πάρουν τα κατάλληλα μέτρα προκειμένου να επαναφέρουν το δίκτυο στην κανονική λειτουργία του. Εστιάζουμε σε τρεις γνωστές ανωμαλίες (προερχόμενες από κακόβουλη χρήση του δικτύου), οι οποίες θα μπορούσαν να χαρακτηριστούν ως επιθέσεις δικτύων (D.D.o.S., Worm propagation, Portscan) και δύο άλλες ανωμαλίες που προκαλούνται από νόμιμη χρήση του δικτύου (Flash crowd, Alpha flows). Η Εικόνα 2 απεικονίζει πώς αυτές οι πέντε ανωμαλίες μπορούν να εμφανιστούν σε μια τοπολογία δικτύου.



Εικόνα 9 Χαρακτηριστικότερες Ανωμαλίες Δικτύου

1. **Επίθεση D.D.o.S.** Μια επίθεση D.D.o.S. (Distributed Denial of Service) χαρακτηρίζεται από μια προσπάθεια να αποτραπεί η νόμιμη χρήση μιας υπηρεσίας. Γενικά, οι επιθέσεις D.D.o.S. εκμεταλλεύονται γνωστές ευπάθειες ενός πρωτοκόλλου επικοινωνίας προκειμένου το θύμα να μην μπορεί να εξυπηρετήσει τα νόμιμα αιτήματα υπηρεσιών που προσφέρει. Μια κοινή πρακτική για τους επιτιθεμένους, στα μεγάλης κλίμακας δίκτυα, είναι να στέλνουν ένα μεγάλο αριθμό πακέτων στο θύμα. Ένας συχνός τύπος επίθεσης D.D.o.S. είναι το «SYN flooding», όπου οι κακόβουλοι υπολογιστές στέλνουν έναν μεγάλο αριθμό πακέτων «T.C.P. SYN» στο θύμα, καθιστώντας κατά συνέπεια τον υπολογιστή-στόχο ανίκανο να επεξεργαστεί όλα αυτά τα αιτήματα. Άλλοι τύποι επιθέσεων D.D.o.S. είναι οι επιθέσεις «U.D.P. flooding» και «I.C.M.P. flooding» όπου ένας μεγάλος αριθμός πακέτων U.D.P. ή I.C.M.P. αντίστοιχα, στέλνεται προς το δίκτυο του θύματος από πολλαπλές πηγές προκειμένου να καταναλωθεί το διαθέσιμο εύρος ζώνης (bandwidth) της σύνδεσης δικτύου του θύματος.

2. **Worm propagation.** Με τον όρο «worm» ορίζουμε ένα κακόβουλο πρόγραμμα το οποίο αυτοδιαδίδεται μέσω δικτύου και προσπαθεί να μολύνει άλλους υπολογιστές εκμεταλλευόμενο μια συγκεκριμένη ευπάθεια στον υπολογιστή-θύμα. Κατά τη διάρκεια της φάσης διάδοσης, ο μολυσμένος υπολογιστής στέλνει ένα μικρό αριθμό πακέτων ανά στόχο, σε έναν μεγάλο πλήθος υπολογιστών στο διαδίκτυο.

3. **Δραστηριότητα Portscan.** Η δραστηριότητα Portscan περιλαμβάνει την κυκλοφορία που προκαλείται από έναν υπολογιστή που στέλνει πακέτα ελέγχου σε ένα ευρύ φάσμα θυρών ενός συγκεκριμένου υπολογιστή προκειμένου να ελέγξει ποιες υπηρεσίες είναι διαθέσιμες.
4. **Flash crowd.** Το Flash Crowd αποτελεί μια μεγάλη νόμιμη ζήτηση για μια συγκεκριμένη υπηρεσία (π.χ. πολλοί πελάτες που «κατεβάζουν» ταυτόχρονα μία νέα διανομή Linux ή κάποιο αρχείο αναβάθμισης ασφάλειας από έναν κεντρικό υπολογιστή μέσω H.T.T.P. / F.T.P.). Αυτό το γεγονός οδηγεί στην αύξηση και της εισερχόμενης (αιτήματα) και εξερχόμενης κίνησης (απαντήσεις) από τον κεντρικό υπολογιστή.
5. **Alpha flows.** Οι ροές «Alpha» συνθέτουν μια ανωμαλία δικτύων στην οποία η κίνηση του δικτύου αυξάνεται σε μεγάλο βαθμό από μερικές μόνο συνδέσεις μεταξύ δύο υπολογιστών. Η κίνηση αυτή προκαλείται συνήθως από μεγάλες μεταφορές αρχείων πάνω από υψηλού εύρους ζώνης συνδέσεις ή από πειράματα δικτύων μεταξύ διαφορετικών περιοχών (domains).

3.7. **Οι αλγόριθμοι ανίχνευσης απότομης μεταβολής**

Προκειμένου να ανιχνευθεί μια επίθεση D.D.o.S., πρέπει να είμαστε σε θέση να ανιχνεύουμε όσο το δυνατόν πιο γρήγορα τις αλλαγές που συμβαίνουν στον αριθμό των I.P. διευθύνσεων που είναι συνδεμένες στο θύμα καθ' όλη τη διάρκεια της προσομοίωσης. Πιο συγκεκριμένα πρέπει να μπορούμε να ανιχνεύουμε τις αλλαγές που συμβαίνουν στο μέτρο της μεταβλητής x_n, η οποία αποτελεί και τη χαρακτηριστική ποσότητα στο μηχανισμό της ανίχνευσης. Εντούτοις, η μεταβλητή αυτή είναι μια τυχαία μεταβλητή λόγω της στοχαστικής φύσης της κίνησης του Διαδικτύου. Συνεπώς, θα παρουσιάσουμε το θεωρητικό υπόβαθρο των σημαντικότερων αλγορίθμων ανίχνευσης.

3.7.1. Αλγόριθμος CU.SUM.

Ο αλγόριθμος του συσωρευτικού αθροίσματος CU.SUM. (Cumulative SUM) είναι ένας αρκετά διαδεδομένος αλγόριθμος στη στατιστική επεξεργασία τυχαίων ακολουθιών, ο οποίος μπορεί να ανιχνεύσει οποιαδήποτε μεταβολή συμβαίνει στη μέση τιμή μιας στατιστικής διαδικασίας. Ο αλγόριθμος CU.SUM. βασίζεται στο γεγονός ότι όταν εμφανιστεί μια μεταβολή στη μέση τιμή μιας τυχαίας ακολουθίας, η κατανομή της πιθανότητας της τυχαίας αυτής ακολουθίας θα αλλάξει επίσης. Γενικά, ο αλγόριθμος CU.SUM. απαιτεί ένα παραμετρικό μοντέλο για την τυχαία ακολουθία έτσι ώστε μια συνάρτηση πυκνότητας πιθανότητας να μπορεί να εφαρμοστεί για να παρακολουθηθεί η ακολουθία. Δυστυχώς, το Διαδίκτυο είναι μια πολύ δυναμική και πολύπλοκη οντότητα, και η θεωρητική κατάρτιση μοντέλων που μπορούν να περιγράψουν την κίνησή του είναι ένα ιδιαίτερα σύνθετο και δυστυχώς ακόμα άλυτο πρόβλημα. Κατά συνέπεια, ένα βασικό πρόβλημα που καλούμαστε να αντιμετωπίσουμε είναι με ποιο τρόπο θα τυποποιηθεί η τυχαία ακολουθία $\{x_n\}$. Δεδομένου ότι οι μη παραμετρικές μέθοδοι δεν απαιτούν κάποιο συγκεκριμένο πρότυπο, στο οποίο να υπακούει η τυχαία μεταβλητή, είναι οι καταλληλότερες για το διαδίκτυο. Η κύρια ιδέα που κρύβεται πίσω από τον μη παραμετρικό αλγόριθμο CU.SUM. είναι ότι συσσωρεύονται οι τιμές εκείνες της ακολουθίας $\{x_n\}$ οι οποίες είναι σημαντικά υψηλότερες από τη μέση τιμή της υπό συνθήκες κανονικής λειτουργίας. Ένα από τα πλεονεκτήματα αυτού του αλγορίθμου είναι ότι παρακολουθεί και καταγράφει τις τιμές της τυχαίας ακολουθίας κατά τρόπο διαδοχικό έτσι ώστε η ανίχνευση να επιτυγχάνεται σε πραγματικό χρόνο.

Έχει αποδειχθεί ότι εάν οι τιμές σε μια χρονική ακολουθία είναι ανεξάρτητες και κατανεμημένες ομοιόμορφα σύμφωνα με ένα παραμετρικό μοντέλο, ο αλγόριθμος CU.SUM. είναι ασυμπτωτικά βέλτιστος για ποικίλα προβλήματα ανίχνευσης του σημείου εκείνου στο οποίο συμβαίνει απότομη μεταβολή στη μέση τιμή μιας τυχαίας ακολουθίας.

Υπάρχουν δύο βασικά κριτήρια που χρησιμοποιούνται για την αξιολόγηση των συστημάτων ανίχνευσης επιθέσεων D.D.o.S. Το πρώτο κριτήριο είναι η πιθανότητα ψευδούς συναγερμού, πράγμα το οποίο αποτελεί μια από τις μεγαλύτερες έννοιες όλων όσων ασχολούνται με την ανίχνευση ανωμαλιών. Εάν ένα σύστημα ανίχνευσης παράγει πάρα πολλούς ψευδείς συναγερμούς, θα απαιτηθεί πάρα πολύς

χρόνος μέχρι να διαπιστωθεί εάν ο συναγερμός δείχνει μια πραγματική επίθεση ή όχι. Εάν η αντίδραση ενός συστήματος σε μια επίθεση (όπως π.χ. το φιλτράρισμα πακέτων που μπορεί να εφαρμόζουν μερικά συστήματα) μπαίνει σε λειτουργία κάθε φορά που ενεργοποιείται ένας ψευδής συναγερμός, η αθώα νόμιμη κίνηση θα τιμωρηθεί άδικα και οι κανονικές υπηρεσίες του εκάστοτε δικτύου θα διαταραχθούν. Το δεύτερο κριτήριο είναι ο χρόνος ανίχνευσης, δηλαδή ο χρόνος που χρειάζεται το σύστημα ανίχνευσης για να ανιχνεύσει μια επίθεση και να αρχίσει να ενεργοποιεί τις υπάρχουσες άμυνες προκειμένου να κατασταλεί η επίθεση. Ασφαλώς, θα αποτελούσε μεγάλο πλεονέκτημα για ένα σύστημα ανίχνευσης επιθέσεων να ανιχνευθεί η επίθεση το συντομότερο δυνατόν, αφού σε μια τέτοια περίπτωση οι απαραίτητοι μηχανισμοί αντίδρασης θα ενεργοποιούνταν νωρίτερα και θα περιόριζαν ή και θα εξάλειφαν με τον τρόπο αυτό τις ανεπιθύμητες συνέπειες της επίθεσης.

Δυστυχώς, τα δύο αυτά κριτήρια είναι αντικρουόμενα μεταξύ τους. Είναι δύσκολο να περιορίσει κανείς το χρόνο ανίχνευση και να μειώσει ταυτόχρονα την πιθανότητα ψευδούς συναγερμού. Επομένως, μεταξύ των δύο αυτών κριτηρίων θα πρέπει να γίνει κάποιος συμβιβασμός. Ο αλγόριθμος CU.SUM. είναι βέλτιστος στην ελαχιστοποίηση του χρόνου ανίχνευσης καθώς επίσης και στη μείωση της πιθανότητα ψευδούς συναγερμού.

3.7.2. **Μέθοδος Ανάλυσης Κύριων Συνιστωσών**

Ο στόχος της μεθόδου της Ανάλυσης Κύριων Συνιστωσών είναι να εφαρμοστεί μια μεθοδολογία σύνθεσης και συνδυασμού δεδομένων ετερογενών οργάνων μέτρησης του δικτύου, προκειμένου να παρασχεθεί ένα γενικευμένο πλαίσιο, ικανό να ανιχνεύσει ένα ευρύ φάσμα ανωμαλιών, όπως αυτές που μπορούν να οδηγήσουν σε αλλαγές στη σύνθεση της κίνησης του δικτύου ή τις κατευθύνσεις της κίνησης μέσα στο δίκτυο. Αυτό επιτυγχάνεται με την εφαρμογή μιας βασισμένης στη μέθοδο P.C.A. (Principal Components Analysis) προσέγγισης σε διάφορα μετρικά ταυτόχρονα μιας ή περισσότερων συνδέσεων. Γενικά, για κάθε σύνδεση δικτύων υπάρχουν διάφορα μετρικά που περιγράφουν την κίνηση που περνά μέσω αυτών των συνδέσεων. Προκειμένου να διαμορφωθεί καλύτερα και να αντιπροσωπευθεί αυτό, δημιουργείται ένα σύνολο εικονικών συνδέσεων (virtual

links) για κάθε πραγματική σύνδεση, με κάθε εικονική σύνδεση να αντιστοιχεί σε ένα διαφορετικό μετρικό.

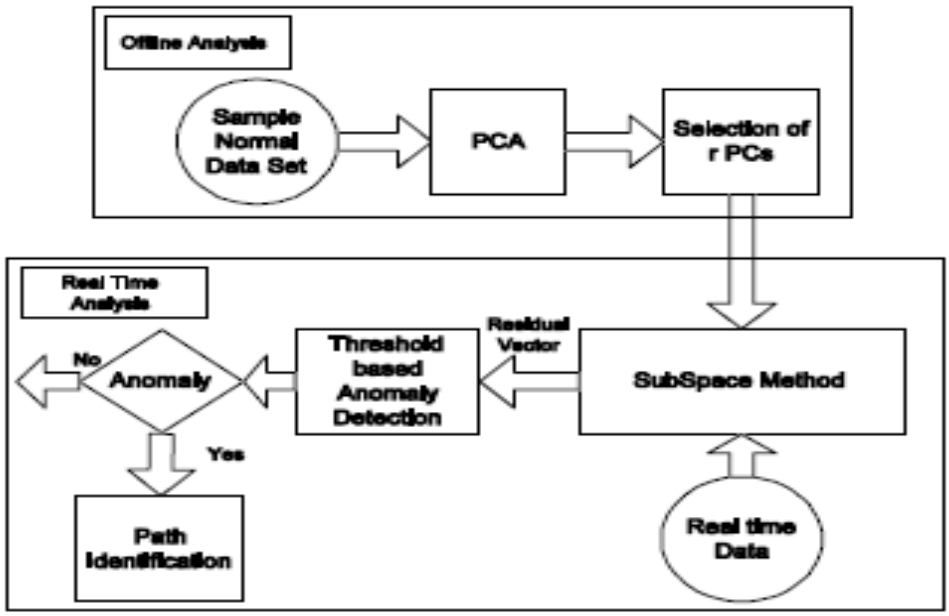
Η βασική ιδέα πίσω από την Ανάλυση Κύριων Συνιστωσών είναι ο εντοπισμός των γραμμικών συνδυασμών των αρχικών μεταβλητών οι οποίοι είναι γραμμικά ανεξάρτητοι μεταξύ τους. Ο στόχος είναι να μειωθεί ο αριθμός των αρχικών μεταβλητών, διατηρώντας τόσες ώστε το σύστημα να εμπεριέχει όσο το δυνατόν μεγαλύτερο μέρος της διακύμανσης των αρχικών μεταβλητών. Οι μεταβλητές αυτές ονομάζονται Κύριες Συνιστώσες (Κ.Σ.) και υπολογίζονται από τα ιδιοδιανύσματα του πίνακα συνδιακύμανσης ή του πίνακα συσχετισμού των αρχικών μεταβλητών.

Η συνολική διαδικασία μπορεί να χωριστεί σε δυο βασικά μέρη, όπως φαίνεται στην Εικόνα 3 :

α) τη διαδικασία εκμάθησης (offline analysis), όπου δημιουργείται το μοντέλο της κίνησης από δεδομένα που θεωρούνται κανονικά (δεν περιέχουν ανωμαλίες) και

β) τη διαδικασία ανίχνευσης που γίνεται σε πραγματικό χρόνο (real time analysis) και η οποία συγκρίνει την τρέχουσα κίνηση με τη μοντελοποιημένη με στόχο την ανίχνευση ανωμαλιών.

Κατά τη διαδικασία εκμάθησης, η μέθοδος εφαρμόζεται σε ένα δείγμα της κίνησης που θεωρείται ότι δεν περιέχει ανωμαλίες με σκοπό να εξαχθούν οι Κύριες Συνιστώσες που επαρκούν για τη περιγραφή των σημαντικών συσχετίσεων στα δεδομένα. Ο αριθμός τους εξαρτάται από το δίκτυο και τον αριθμό εικονικών συνδέσεων, και αντιπροσωπεύει τον αριθμό των Κ.Σ. που απαιτείται για τη σύλληψη του ποσοστού της διακύμανσης που χρειάζεται το σύστημα για να μοντελοποιήσει την κανονική κίνηση του δικτύου. Οι επιλεγείσες Κ.Σ. (αποτέλεσμα της διαδικασίας εκμάθησης) θα χρησιμοποιηθούν στη Μέθοδο Υποχώρων (Subspace Method).



Εικόνα 10 Παρουσίαση της μεθοδολογίας P.C.A.

Ο στόχος της Μεθόδου Υποχώρων είναι να διαχωριστούν τα τρέχοντα δεδομένα κίνησης σε δύο διαφορετικά τμήματα: ένα που να περιέχει τα δεδομένα της κίνησης που θεωρείται κανονική (ynorm) και μοιάζει με την μοντελοποιημένη κίνηση και ένα τμήμα το οποίο περιγράφει το υπόλοιπο τμήμα της κίνησης (yres). Γενικά, οι ανωμαλίες τείνουν να οδηγήσουν σε μεγάλες διακυμάνσεις στο υπόλοιπο τμήμα yres, δεδομένου ότι παρουσιάζουν διαφορετικά χαρακτηριστικά από αυτά της μοντελοποιημένης κίνησης.

Κατά τη διάρκεια της διαδικασίας ανίχνευσης που γίνεται σε πραγματικό χρόνο, το τρέχον διάνυσμα της κίνησης προβάλλεται σε δύο διαφορετικούς υποχώρους, με τη χρήση των Κ.Σ. που υπολογίστηκαν στη διαδικασία εκμάθησης. Όταν εμφανιστεί μια ανωμαλία, το διάνυσμα yres παρουσιάζει μία μεγάλη διακύμανση σε μερικές από τις μεταβλητές του. Ένα συνηθισμένο στατιστικό μέτρο που χρησιμοποιείται για την ανίχνευση ανωμαλίας είναι το τετραγωνικό σφάλμα πρόβλεψης (Squared Prediction Error, S.P.E.). Αν λοιπόν το S.P.E. ξεπεράσει κάποιο προκαθορισμένο κατώφλι τότε μπορούμε να θεωρήσουμε ότι έχουμε ανωμαλία στο δίκτυο.

3.7.3. **Ευφυής Μέθοδοι Δειγματοληψίας**

Ερευνητικά αποτελέσματα κατέδειξαν ότι η δειγματοληψία ροών σε σχέση με τη δειγματοληψία πακέτων βελτιώνει την ακρίβεια των στατιστικών του δικτύου. Το γεγονός αυτό καθιστά τη δειγματοληψία ροών καταλληλότερη για την εφαρμογή της στο πεδίο της ανίχνευσης ανωμαλιών δικτύου. Στη συνέχεια, παρουσιάζουμε συνοπτικά δύο γνωστές τεχνικές δειγματοληψίας ροών. Η πρώτη είναι, η «Επιλεκτική Δειγματοληψία» (Selective Sampling) η οποία όπως έχουμε αναφέρει έχει ως στόχο τις μικρές ροές, ενώ ο δεύτερη αναφέρεται ως «Έξυπνη δειγματοληψία» (Smart Sampling), και επιλέγει τις μεγάλες ροές. Και οι δύο αυτές τεχνικές παρουσιάζουν έναν «ευκαιριακό» (opportunistic) χαρακτήρα στη λειτουργία τους, δεδομένου ότι στοχεύουν στην εκμετάλλευση του γεγονότος ότι ένα μεγάλο μέρος των πληροφοριών (που αφορούν κυρίως την ανωμαλία) περιλαμβάνεται μέσα σε ένα μικρό μέρος των ροών. Επομένως, ανωμαλίες που γίνονται αντιληπτές συνήθως από «ακραία» στοιχεία σε σχέση με τα υπόλοιπα μπορούν να αποκαλυφθούν ευκολότερα μέσα σε ένα κατάλληλα επιλεγμένο σύνολο στοιχείων, όπως αυτό που μπορεί να προκύψει από τεχνικές «ευφυούς δειγματοληψίας».

Έχει καταδειχθεί ότι οι μικρές ροές είναι συνήθως η πηγή πολλών επιθέσεων δικτύων (π.χ. επιθέσεις D.D.o.S., portscans, αυτοδιαδιδόμενοι ιοί), και επομένως πρέπει να επιλεχτούν κατά προτίμηση προκειμένου να επιτευχθεί υψηλή αποτελεσματικότητα στην ανίχνευση ανωμαλιών δικτύου. Η «Επιλεκτική Δειγματοληψία» ακολουθεί αυτό το παράδειγμα και η επιλογή μιας μεμονωμένης ροής είναι βασισμένη στην ακόλουθη έκφραση:

$$p(x) = \begin{cases} c & x \le z \\ \dfrac{z}{n \cdot x} & x > z \end{cases}$$

όπου με x συμβολίζουμε το μέγεθος της ροής σε πακέτα, $0 < c \le 1$, $n \ge 1$ και z είναι ένα κατώφλι (μετρούμενο σε πακέτα).

Αντίθετα, η «Έξυπνη δειγματοληψία» είναι ένας τύπος δειγματοληψίας βασισμένος σε ροές που εστιάζει στην επιλογή των μεγάλων ροών. Πιο συγκεκριμένα, στην έξυπνη δειγματοληψία μια ροή του μεγέθους x επιλέγεται με πιθανότητα p(x) σύμφωνα με την ακόλουθη έκφραση:

$$p(x) = \begin{cases} x/z & x < z \\ 1 & x \geq z \end{cases}$$

όπου το x είναι το μέγεθος ροής σε bytes και z είναι ένα κατώφλι. Θεωρούμε το x ως μέγεθος ροής σε πακέτα. Όπως παρατηρούμε από τη δεύτερη σχέση, οι ροές που είναι μεγαλύτερες στο μέγεθος από το κατώφλι z επιλέγονται με πιθανότητα ίση με 1, ενώ οι ροές που είναι μικρότερες από z επιλέγονται με πιθανότητα ανάλογη προς το μέγεθός τους. Αυτή η τεχνική δειγματοληψίας είναι κατάλληλη για τον εντοπισμό ανωμαλιών που προκαλούνται από μεγάλες ροές όπως τα «Flash crowds» και οι «ροές Alpha».

3.7.4. Εντροπία

Στην ενότητα αυτή παρουσιάζουμε μια μέθοδο για την ανίχνευση ανωμαλιών με βάση την εντροπία, η οποία ανιχνεύει ανωμαλίες εξετάζοντας διάφορες κατανομές από χαρακτηριστικά της δικτυακής κίνησης.

Η εντροπία H(X) ενός συνόλου X= $\{x_1, x_2, \ldots x_n\}$ορίζεται ως:

$$H(X) = -\sum_{i=1}^{N} p_i \log_2(p_i)$$

όπου N ο αριθμός των στοιχείων που ανήκουν στο X και p_i είναι η πιθανότητα $P[X = x_i]$. Η Εντροπία είναι στην ουσία μια εκτίμηση του πόσο τυχαία κατανεμημένο είναι ένα σύνολο στοιχείων. Υψηλές τιμές της εντροπίας αντιστοιχούν σε διασκορπισμένη κατανομή πιθανοτήτων, ενώ μεγάλες τιμές στην Εντροπία σημαίνουν ότι υπάρχουν λίγα στοιχεία με πολύ μεγάλη πιθανότητα σε σχέση με τα υπόλοιπα στο σύνολο. Οι πιθανές τιμές της Εντροπίας, όπως αυτές ορίζονται στο

προηγούμενο τύπο έχουν εύρος μεταξύ 0 - $\log_2 N$. Από τη στιγμή που μπορεί κανείς να εξετάσει διαφορετικά σύνολα στοιχείων, με διαφορετικό αριθμό στοιχείων, ο επόμενος τύπος χρησιμοποιείται για να ομαλοποιήσει την τιμή της εντροπία, διαιρώντας την H(X) με την μέγιστη τιμή της ($\log_2 N$). Η κανονικοποιημένη εντροπία ορίζεται από τον επόμενο τύπο:

$$H(X) = -\frac{\sum_{i=1}^{N} p_i \log_2(p_i)}{\log_2 N}$$

Τα στοιχεία στο σύνολο αντιστοιχούν σε διαφορετικές τιμές χαρακτηριστικών γνωρισμάτων της δικτυακής κίνησης. Μερικές από τις κατανομές στοιχείων οι οποίες έχει αποδειχθεί ότι παρουσιάζουν ενδιαφέρον στην ανίχνευση ανωμαλιών είναι :

- Η διεύθυνση πηγής και προορισμού του πακέτου I.P.
- Η πόρτα πηγής και προορισμού στην επικεφαλίδα του T.C.P./ I.P.
- Το μέγεθος της ροής (flow-size)
- Το μέγεθος του πακέτου I.P.

Οι ανιχνευτές ανωμαλιών ελέγχουν την κατάσταση του συστήματος παρακολούθησης, κάνοντας περιοδικές συγκρίσεις μεταξύ ενός προφίλ που χαρακτηρίζεται ως κανονικό και τα τρέχοντα δεδομένα που λαμβάνονται από διαδοχικές μετρήσεις. Κάθε απόκλιση από την κατάσταση η οποία θεωρείται κανονική, από το μοντέλο δηλαδή του δικτύου, υποδεικνύει την παρουσία μιας ανωμαλίας.

Κατά την παρακολούθηση της κυκλοφορίας του δικτύου, τα δεδομένα που συλλέγονται είναι συνήθως χωρισμένα σε χρονοσχισμές (slots ή time bins) και ο αλγόριθμος ανίχνευσης ανωμαλιών εφαρμόζεται για κάθε time bin διαδοχικά.

ΚΕΦΑΛΑΙΟ 4

Εισαγωγή

Στο κεφάλαιο αυτό θα παρουσιάσουμε την αρχιτεκτονική ενός I.D. S. και πιο συγκεκριμένα ενός συστήματος, το οποίο προσαρμόζεται σε ένα ασύρματο δίκτυο. Το σύστημα επικεντρώνεται στην ανίχνευση γνωστών ανωμαλιών, όπως η εξάπλωση Worms, οι επιθέσεις D.D.o.S. και οι ροές Flash crowd.

4.1. Συλλογή δεδομένων

Το πρώτο βήμα για τον εντοπισμό ανωμαλιών είναι η παρακολούθηση του δικτύου. Σε ένα ενσύρματο δίκτυο, ο διαχειριστής θα πρέπει να εγκαταστήσει ένα σύστημα για την παρακολούθηση της κίνησης σε όλους τους δρομολογητές και στη συνέχεια να συγκεντρώσει όλα τα δεδομένα της δικτυακής κίνησης σε ένα κεντρικό σημείο για την ανάλυσή τους. Αντίστοιχα, σε ένα ασύρματο δίκτυο, ο διαχειριστής θα πρέπει να εγκαταστήσει ένα σύστημα παρακολούθησης του δικτύου σε όλα τα σημεία πρόσβασης, συλλέγοντας στοιχεία για την κίνηση από και προς όλους τους ασύρματους κόμβους.

Υπάρχουν πολλά συστήματα παρακολούθησης και φιλτραρίσματος των δεδομένων που χρησιμοποιούνται για την καταγραφή της δικτυακής κίνησης. Υπάρχουν επίσης πολλοί διαφορετικοί τύποι και μορφές αρχείων σύλληψης (capture formats), όπως τα Tcpdump, Catapult DCT2000, Cisco Secure IDS iplog, Microsoft Network Monitor, Novell LANalyzer, RADCOM WAN / LAN Analyzer κ.ά.

Ένα από τα πιο συχνά χρησιμοποιούμενα εργαλεία είναι το tcpdump, ένα εργαλείο ανοικτού κώδικα που διατίθεται για συστήματα Unix και Linux. Το Tcpdump εξετάζει την κίνηση που διέρχεται μέσω μιας συγκεκριμένης διεπαφής (interface) και καταγράφει πληροφορίες σχετικά με όλα τα πακέτα που ταιριάζουν στην έκφραση που διατυπώθηκε κατά το χρόνο εκτέλεσης. Με άλλα λόγια, συλλέγει και αποθηκεύει πληροφορίες για πακέτα, αφού πρώτα φιλτράρει τη κίνηση με βάση την έκφραση που ορίζει ο χρήστης κατά την εκκίνηση του προγράμματος. Με τον

τρόπο αυτό επιτυγχάνει οικονομία στο χώρο αποθήκευσης και κάνει πιο εύκολη την ανάλυση των δεδομένων με γυμνό μάτι. Ένα άλλο πολύ συχνά χρησιμοποιούμενο εργαλείο για την ανάλυση του δικτύου είναι το Wireshark, το οποίο προσφέρει επιθεώρηση πακέτων σε βάθος (deep packet inspection), η οποία είναι προσαρμοσμένη στα χαρακτηριστικά του πρωτόκολλου που χρησιμοποιείται στο πακέτο. Με τον τρόπο αυτό είναι ευκολότερη η ανάλυση της κίνησης από τον χρήστη.

4.2. Ανίχνευση Ανωμαλιών με βάση την Εντροπία

Στην παρακάτω ενότητα, θα παρουσιαστεί ο τρόπος με τον οποίο ανωμαλίες, όπως η εξάπλωση Worms, οι επιθέσεις τύπου D.D.o.S. και οι ροές Flash crowd επηρεάζουν τις προαναφερόμενες μετρήσεις της ενότητας 3.7.4. και με ποιο τρόπο είναι δυνατόν να τις εντοπίσει κανείς, χρησιμοποιώντας την Εντροπία.

4.2.1. Ανίχνευση εξάπλωσης Worms

Ένα worm μπορεί είτε να διαδοθεί από υπολογιστή σε υπολογιστή με ξεχωριστές συνδέσεις που εκκινεί το ίδιο ή μπορεί να ενσωματωθεί σε προϋπάρχουσα επικοινωνία. Στην εργασία μας θα επικεντρωθούμε στην πρώτη περίπτωση, η οποία είναι και η πιο συνηθισμένη.

Όταν ένα worm ταχείας σάρωσης προσπαθεί να εξαπλωθεί από τον αρχικό ξενιστή χρησιμοποιώντας πολλαπλές ξεχωριστές συνδέσεις, ένας μικρός σχετικά αριθμός μολυσμένων υπολογιστών σκανάρει το δίκτυο ψάχνοντας για πιθανά θύματα. Συνήθως η κίνηση που δημιουργείται από αυτή τη διαδικασία μπορεί να ανιχνευθεί γιατί έχει διαφορετικά χαρακτηριστικά από την κανονική κίνηση του δικτύου.

Πιο συγκεκριμένα, οι ροές που παράγονται από το worm αποκτούν σημαντικό πλήθος σε σύγκριση με το συνολικό αριθμό των ροών, η I.P. διεύθυνση πηγής του κάθε επιτιθέμενου αναφέρεται σε ένα μεγάλο αριθμό πακέτων και καθώς το σύνολο των επιτιθέμενων (κατά την διαδικασία εξάπλωσης) είναι πολύ μικρό σε σχέση με το σύνολο των διευθύνσεων I.P. στο δίκτυο, η κατανομή των πηγαίων διευθύνσεων I.P.

στην επικεφαλίδα του I.P., περιέχει λιγότερη εντροπία. Έχουμε δηλαδή ένα μικρό σύνολο διευθύνσεων με μεγάλη πιθανότητα εμφάνισης. Από την άλλη πλευρά, η κατανομή των διευθύνσεων I.P. προορισμού είναι τυχαία, συνεπώς η εντροπία της συγκεκριμένης κατανομής παραμένει ίδια είτε αυξάνεται.

Ως εκ τούτου, ένας ασφαλής τρόπος για την ανίχνευση της εξάπλωση worms, είναι ο έλεγχος της σχέσης μεταξύ της εντροπίας στη διεύθυνση I.P. προέλευσης και της εντροπίας στη διεύθυνση I.P. προορισμού. Πιο συγκεκριμένα, το σύστημα πρέπει να εκτιμήσει την αναλογία υπό κανονικές συνθήκες και να την αποθηκεύσει στο προφίλ του δικτύου. Υπό κανονικές συνθήκες, όπου η κυκλοφορία είναι κατανεμημένη μεταξύ ασύρματων κόμβων του δικτύου, ο λόγος αυτός αναμένεται να είναι κοντά στο 1. Στην περίπτωση της εξάπλωσης ενός worm, αναμένουμε ο δείκτης να μειωθεί σημαντικά.

4.2.2. Ανίχνευση επιθέσεων D.D.o.S.

Μια επίθεση D.D.o.S. έχει τα ακόλουθα χαρακτηριστικά αποτελέσματα στην κίνηση του δικτύου: Μεγάλες ποσότητες εισερχόμενης κίνησης έχουν ως στόχο ένα συγκεκριμένο υπολογιστή στο δίκτυο. Η κακόβουλη κίνηση μπορεί να αποτελείται είτε από πακέτα τύπου T.C.P. / I.P. (Transmission Control Protocol/ Internet Protocol) είτε από πακέτα U.D.P. (User Datagram Protocol). Στην πρώτη περίπτωση, η πηγαία πόρτα είναι συνήθως η ίδια σε όλα τα εισερχόμενα πακέτα, ενώ στην περίπτωση πλημμύρας με πακέτα U.D.P. θα μπορούσε να είναι είτε ίδια είτε διαφορετική στα εισερχόμενα πακέτα. Και στις δύο περιπτώσεις ωστόσο, η διεύθυνση I.P. προορισμού είναι η ίδια.

Από την άλλη πλευρά, η εξερχόμενη κίνηση από τον κόμβο-θύμα είναι πολύ λιγότερη σε όγκο, δεδομένου ότι συνήθως είτε φιλτράρεται είτε απορρίπτεται η εισερχόμενη κίνηση λόγω της έλλειψης πόρων από το λειτουργικό σύστημα του θύματος. Εάν οι ροές που παράγονται από την επίθεση αποκτήσουν ένα σημαντικό μερίδιο του συνόλου των ροών, η διεύθυνση I.P. προορισμού του στόχου θα πρέπει να υπάρχει σε πολλά πακέτα και κατά συνέπεια η εντροπία στην κατανομή της διεύθυνσης I.P. προορισμού θα μειωθεί σημαντικά. Από την άλλη πλευρά, η κατανομή των πηγαίων διευθύνσεων I.P. πρέπει να είναι πιο τυχαία σε σχέση με την

κατανομή σε κανονικές συνθήκες κυκλοφορίας. Έτσι η εντροπία είναι ίδια ή αυξημένη.

Ως εκ τούτου, ένα ασφαλές κριτήριο για την ανίχνευση επιθέσεων D.D.o.S. είναι ο έλεγχος στη σχέση μεταξύ της εντροπίας στη διεύθυνση I.P. προέλευσης και της εντροπίας στη κατανομή διευθύνσεων I.P. προορισμού. Σε μια επίθεση D.D.o.S., αναμένουμε ο δείκτης να αυξηθεί σημαντικά. Το αποτέλεσμα λοιπόν μιας επίθεσης D.D.o.S. σε ένα δίκτυο έχει αντίστροφα αποτελέσματα από ότι η εξάπλωση ενός worm.

4.3. Περιγραφή της προσομοίωσης και της υλοποίησης του αλγορίθμου ανίχνευσης επιθέσεων

Τα πειράματα που έγιναν στα πλαίσια της εργασίας αυτής βασίζονται στην προσομοίωση με τη χρήση του προσομοιωτή δικτύου N.S.-2.

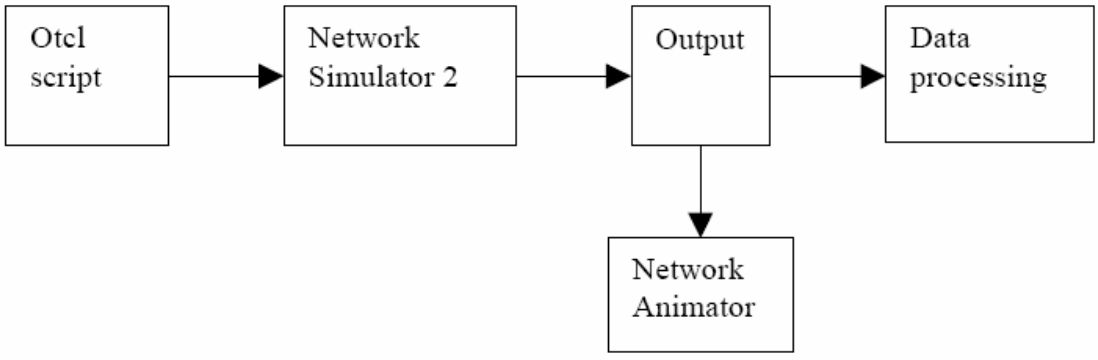
Στα ενσύρματα δίκτυα το N.S. μπορεί να χρησιμοποιηθεί για την προσομοίωση της δρομολόγησης (I.P., multicast routing) με διαφορετικές ουρές: drop-tail, RED (Random Early Drop), FQ (Fair Queuing), SFQ (Stochastic Fair Queuing). Προσομοιώνει επίσης το στρώμα μεταφοράς: T.C.P., U.D.P. και τη μετάδοση πακέτων multicast και επίσης υποστηρίζει διαφορετικές πηγές δικτυακής κίνησης όπως web, ftp, telnet, Constant Bit Rate (C.B.R.), και στοχαστική.

Στα ασύρματα δίκτυα, το N.S. υποστηρίζει ad-hoc δρομολόγηση και mobile I.P. και έχει επίσης αλγόριθμους για πιο περίπλοκα πρωτόκολλα, όπως η διάδοση Directed diffusion και sensor-M.A.C. Παράλληλα, υπάρχει η δυνατότητα ιεραρχικής δρομολόγησης σε ασύρματα και ενσύρματα δίκτυα, με τη χρήση ομάδων (clusters). Τη δυνατότητα αυτή θα χρησιμοποιήσουμε στη συνέχεια, προκειμένου να προσομοιώσουμε τα ιεραρχικά δίκτυα.

4.3.1. Ο Προσομοιωτής Network Simulator-2

Ο προσομοιωτής Network Simulator-2 είναι το αποτέλεσμα μιας συνεχιζόμενης προσπάθειας έρευνας και ανάπτυξης ενός προσομοιωτή διακριτών γεγονότων με στόχο την έρευνα δικτύων. Προσφέρει θεμελιώδη υποστήριξη για την προσομοίωση ενός πλήθους πρωτοκόλλων όλων των δικτυακών επιπέδων τόσο για ενσύρματα όσο και για ασύρματα τοπικά και δορυφορικά δίκτυα. Εξαιτίας των δυνατοτήτων που διαθέτει και του πλήθους των προσφερόμενων υλοποιήσεων πρωτοκόλλων, έχει άτυπα καθιερωθεί ως ο βασικότερος προσομοιωτής για την αξιολόγηση πρωτοκόλλων για M.A.NETs.

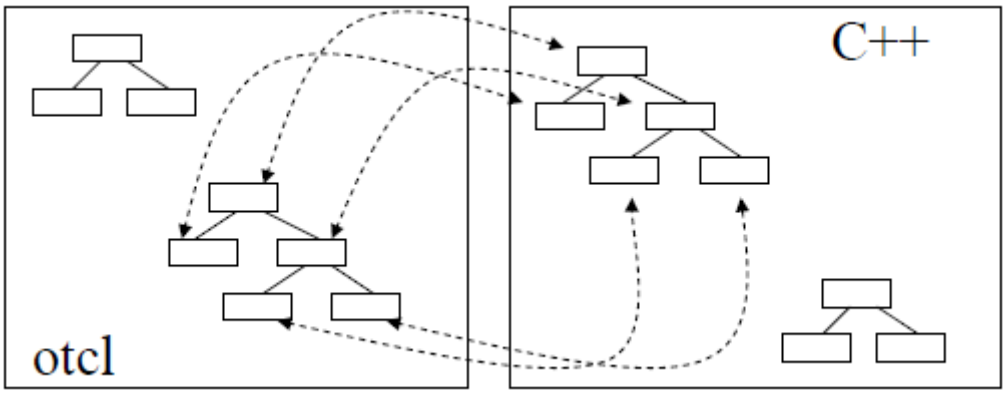
Ο N.S.-2 βασίζεται σε έναν αντικειμενοστραφή προσομοιωτή που είναι γραμμένος σε C++ και σε έναν διερμηνευτή της O.T.c.l. (αντικειμενοστραφής επέκταση της scripting γλώσσας Tool Command Language) που χρησιμοποιείται για να εκτελούνται τα σενάρια (scripts) του χρήστη. Αυτό σημαίνει ότι ο χρήστης γράφει ένα σενάριο O.T.c.l., όπου καθορίζει τις παραμέτρους του δικτύου (τις πηγές, τους προορισμούς, τον τύπο της κίνησης) και πια πρωτόκολλα θα χρησιμοποιήσει. Το σενάριο έπειτα χρησιμοποιείται από τον προσομοιωτή κατά τη διάρκεια της προσομοίωσης. Το αποτέλεσμα αυτής της προσομοίωσης είναι ένα ή περισσότερα αρχεία καταγραφής (trace file), τα οποία μπορούν να χρησιμοποιηθούν για την επεξεργασία των δεδομένων της προσομοίωσης (π.χ. υπολογισμό καθυστέρησης και ποσοστού παράδοσης), καθώς και για την γραφική απεικόνισή της μέσω του Network Animator (N.A.M.). Μια γενική άποψη της δομής του N.S.-2 φαίνεται στην Εικόνα 4 που ακολουθεί.



Εικόνα 11 Γενική άποψη της δομής του NS-2

4.3.2. Δομή του Network Simulator-2

Ο προσομοιωτής περιλαμβάνει ένα πολύ μεγάλο αριθμό από εφαρμογές, πρωτόκολλα, τύπους δικτύων, στοιχεία δικτύων και μοντέλα κίνησης. Αυτά ονομάζονται "Αντικείμενα Προσομοίωσης". Χρησιμοποιεί δύο διαφορετικές γλώσσες προγραμματισμού γιατί στην πραγματικότητα έχει δύο διαφορετικά πράγματα να κάνει. Από τη μία η λεπτομερής προσομοίωση πρωτοκόλλων χρειάζεται μία γλώσσα προγραμματισμού συστήματος η οποία θα μπορεί να χειριστεί αποτελεσματικά byte και κεφαλίδες πακέτων και να υλοποιεί αλγόριθμους που εκτελούνται πάνω από μεγάλα σύνολα δεδομένων. Για αυτές τις ενέργειες ο γρήγορος χρόνος εκτέλεσης είναι σημαντικός. Από την άλλη, ένα μεγάλο μέρος της έρευνας αφορά παραμέτρους και σχηματισμούς ελαφρά διαφοροποιημένους καθώς και γρήγορη εξέταση ενός αριθμού σεναρίων. Σε αυτές τις περιπτώσεις η ταχύτητα αλλαγής και επανεκτέλεσης είναι περισσότερο σημαντική. Αφού η παραμετροποίηση τρέχει μια φορά, ο γρήγορος χρόνος σχεδιασμού σε αυτό το κομμάτι είναι περισσότερο σημαντικός. Ο N.S.-2 επιτυγχάνει και τους δύο στόχους. Η C++, ως γλώσσα προγραμματισμού, είναι γρήγορη στην εκτέλεση αλλά αργή στην αλλαγή του κώδικα, κάτι που την κάνει κατάλληλη για λεπτομερή υλοποίηση πρωτοκόλλων. Αντίθετα, η O.T.c.l., ως γλώσσα κωδικοποίησης σεναρίων (scripting language), εκτελείται πολύ πιο αργά αλλά είναι ιδανική για γρήγορες αλλαγές στον κώδικα, κάτι που την κάνει ιδανική για παραμετροποίηση και έλεγχο των προσομοιώσεων. Δηλαδή έχουμε έναν συμβιβασμό ανάμεσα στην μεγάλη ταχύτητα εκτέλεσης που παρέχεται από μια γλώσσα όπως η C++ και στους μικρούς χρόνους γραφής ενός σεναρίου σε μια γλώσσα όπως η O.T.c.l. Αυτή η διαλειτουργικότητα υλοποιείται με την παρουσία δύο ιεραρχιών κλάσεων, οι οποίες είναι η μεταφρασμένη ιεραρχία της C++ και η διερμηνευμένη ιεραρχία της O.T.c.l., μεταξύ των οποίων υπάρχει μία προς μία αντιστοιχία κλάσεων (Εικόνα 5).



Εικόνα 12 Απεικόνιση της «1-1» αντιστοιχίας μεταξύ των κλάσεων της μεταφρασμένης ιεραρχίας της C++ και της διερμηνευμένης ιεραρχίας της O.T.c.l.

Η μεταφρασμένη ιεραρχία της C++ επιτρέπει να επιτυγχάνεται αποδοτικότητα στην προσομοίωση και ταχύτερη εκτέλεση. Αυτό επιτρέπει τη μείωση του χρόνου επεξεργασίας των δεδομένων και των γεγονότων. Η O.T.c.l. χρησιμοποιείται για τον καλύτερο έλεγχο της προσομοίωσης. Στα σενάρια O.T.c.l. που δίνονται από το χρήστη καθορίζονται η τοπολογία του δικτύου, τα συγκεκριμένα πρωτόκολλα κάθε επιπέδου, καθώς και οι εφαρμογές που θα προσομοιωθούν. Επίσης, καθορίζονται τα δεδομένα που θα καταγραφούν και, μερικώς, η μορφή τους. Η O.T.c.l. μπορεί να κάνει χρήση των μεταφρασμένων αντικειμένων σε C++ μέσω μιας O.T.c.l. διεπαφής (interface), αφού δημιουργείται ένα αντικείμενο O.T.c.l. για κάθε αντικείμενο C++.

Ο N.S.-2 είναι προσομοιωτής διακριτών γεγονότων (discrete events). Αυτό σημαίνει ότι η χρονική πρόοδος μιας προσομοίωσης εξαρτάται από τη χρονική ακολουθία των γεγονότων, τα οποία διατηρούνται από έναν χρονοπρογραμματιστή (scheduler). Έτσι, κάθε γεγονός είναι ένα αντικείμενο στην ιεραρχία κλάσεων της C++ με ένα μοναδικό αναγνωριστικό (I.D.), έναν προγραμματισμένο χρόνο πραγματοποίησης και έναν δείκτη σε ένα αντικείμενο που χειρίζεται το γεγονός. Ο χρονοπρογραμματιστής χρησιμοποιεί μία διατεταγμένη δομή δεδομένων με τα γεγονότα, τα οποία πρόκειται να εκτελεστούν, και τα εκτελεί διαδοχικά. Οι χρονικές στιγμές κατά τις οποίες θα συμβούν τα γεγονότα καθορίζονται από το σενάριο του χρήστη αλλά και τυχαία.

4.3.3. Περιγραφή του πειράματος

Για την επικύρωση της λειτουργίας του συστήματος I.D.S. έχουμε σχεδιάσει και εκτελέσει τρία διαφορετικά σενάρια. Κάθε σενάριο αποτελείται από ένα ασύρματο δίκτυο 30 κόμβων. Κάθε κόμβος, ορίζεται ως μέρος ενός array node_(), είναι κινητός και αλλάζει θέσεις τυχαία σε μια επιφάνεια 1000m x 1000m. Επιπλέον, υπάρχει U.D.P. και T.C.P. κίνηση μεταξύ των κόμβων, που δημιουργείται τυχαία. Κάθε σενάριο αποτελείται από ένα κεντρικό t.c.l. script που περιέχει βασικές πληροφορίες για την προσομοίωση. Το script περιλαμβάνει: ένα script που καθορίζει τη κανονική κίνηση (background traffic) και ένα script για την προσομοίωση της συγκεκριμένης ανωμαλίας. Ο χρόνος προσομοίωσης είναι 200sec σε κάθε σενάριο.

Η δρομολόγηση μεταξύ των κόμβων γίνεται με δύο διαφορετικούς τρόπους. Στην πρώτη σειρά πειραμάτων, χρησιμοποιείται ad-hoc δρομολόγηση ανάμεσα στους 30 κόμβους με τη χρήση του πρωτοκόλλου D.S.D.V. Στη δεύτερη περίπτωση, θεωρούνται 3 σταθεροί κόμβοι οι οποίοι έχουν στατική δρομολόγηση μεταξύ τους. Αυτοί οι κόμβοι είναι οι κεντρικοί κόμβοι τριών ξεχωριστών ομάδων, με κάθε ομάδα να περιέχει, κατά την εκκίνηση του πειράματος, 10 δευτερεύοντες κόμβους. Στη συνέχεια, καθώς οι συνολικά 30 δευτερεύοντες κόμβοι κινούνται τυχαία στην επιφάνεια, με τη χρήση πρωτοκόλλου mobile I.P. μπορεί να αλλάξουν ομάδα. Και στις δύο περιπτώσεις, οι αρχικές θέσεις των 30 κόμβων είναι οι ίδιες και χρησιμοποιείται το ίδιο σενάριο παραγωγής τυχαίας κίνησης μεταξύ τους. Να σημειωθεί ότι στην περίπτωση της ιεραρχικής δρομολόγησης, οι κύριοι κόμβοι δεν παράγουν δικτυακή κίνηση αλλά χρησιμοποιούνται μονάχα για την προώθηση των πακέτων και την επικοινωνία κόμβων που ανήκουν σε διαφορετικές ομάδες.

Ένας κινητός κόμβος του δικτύου αποτελείται από στοιχεία όπως Link Layer, Interface Queue, M.A.C. (Media Access Control) στρώμα, το ασύρματο κανάλι εκπομπής και λήψης σημάτων κ.λπ. Κατά την έναρξη μιας ασύρματης προσομοίωσης, πρέπει να καθορίσουμε τον τύπο για κάθε ένα από αυτά τα στοιχεία. Επιπλέον, πρέπει να καθοριστούν οι περισσότερες παράμετροι όπως το είδος της κεραίας, το μοντέλο ραδιοδιάδοσης και ο τύπος του πρωτόκολλου ad-hoc δρομολόγησης που χρησιμοποιείται από τους κόμβους.

4.5.2 Αρχικές ρυθμίσεις του πειράματος

Σε κάθε πείραμα, υπάρχουν κάποιες βασικές ρυθμίσεις για τη λειτουργία της προσομοίωσης. Πιο συγκεκριμένα:

```
# ======================================================================
# Define options
# ======================================================================

set opt(chan)    Channel/WirelessChannel        ;# channel type
set opt(prop)    Propagation/TwoRayGround       ;# radio-propagation model
set opt(netif)   Phy/WirelessPhy                ;# network interface type
set opt(mac)     Mac/802_11                     ;# MAC type
set opt(ifq)     Queue/DropTail/PriQueue        ;# interface queue type
set opt(ll)      LL                             ;# link layer type
set opt(ant)     Antenna/OmniAntenna            ;# antenna model
set opt(ifqlen)          50                     ;# max packet in ifq
set opt(nn)              30                     ;# number of mobilenodes
set opt(adhocRouting)    DSDV                   ;# routing protocol

set opt(x)       1000                           ;# x coordinate of topology
set opt(y)       1000                           ;# y coordinate of topology
set opt(seed)    123.0                           ;# random seed
set opt(stop)    200                            ;# time to stop simulation


# ======================================================================
```

Στη συνέχεια δημιουργούμε το αντικείμενο του προσομοιωτή που είναι απαραίτητο σε κάθε τρέξιμο του N.S., ορίζουμε τον τρόπο και το αρχείο στο οποίο θα αποθηκεύονται τα αποτελέσματα από την προσομοίωση και δημιουργούμε το αντικείμενο G.O.D. (General Operations Director), που χρησιμοποιείται εσωτερικά από το στρώμα M.A.C. Το αντικείμενο αυτό χρησιμοποιείται για την αποθήκευση πληροφοριών σχετικά με την τοπολογία του δικτύου και έχει ένα πίνακα με τον ελάχιστο αριθμό hops που χρειάζονται για να επικοινωνήσει ο ένας κόμβος με κάποιον άλλο.

```
# Create simulator
set ns_     [new Simulator]

# Set up trace file
$ns_ use-newtrace
set tracefd [open results/trace.tr w]
$ns_ trace-all $tracefd


# Create the "general operations director"
# Used internally by MAC layer: must create!
create-god $val(nn)
```

4.5.3. Σενάριο για ad hoc δρομολόγηση

Στην περίπτωση της δρομολόγησης ad-hoc, με τους 30 κόμβους, κάθε κόμβος ορίζεται από τα παρακάτω στοιχεία.

```
# Configure for mobile nodes
$ns_ node-config -adhocRouting $opt(adhocRouting) \
                 -llType $opt(ll) \
                 -macType $opt(mac) \
                 -ifqType $opt(ifq) \
                 -ifqLen $opt(ifqlen) \
                 -antType $opt(ant) \
                 -propType $opt(prop) \
                 -phyType $opt(netif) \
                 -channelType $opt(chan) \
                 -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON \
                -macTrace OFF \
                -movementTrace OFF
```

Στη συνέχεια, δημιουργούνται 3 γειτονιές από 10 κόμβους η κάθε μια, και για κάθε κόμβο ενεργοποιείται η τυχαία κίνηση εντός του προκαθορισμένου χώρου.

```
#create nodes in first neighborhood
for {set i 0} {$i < 10 } {incr i} {
    set node_($i) [$ns_ node ]

    $node_($i) set X_ [expr $i + 200.0000000]
    $node_($i) set Y_ [expr $i + 200.0000000]
    $node_($i) set Z_ 0.0

    $node_($i) random-motion 1
    $node_($i) start
}

#create nodes in second neighborhood
for {set i 10} {$i < 20 } {incr i} {
    set node_($i) [$ns_ node ]

    $node_($i) set X_ [expr $i + 600.0000000]
    $node_($i) set Y_ [expr $i + 200.0000000]
    $node_($i) set Z_ 0.0

    $node_($i) random-motion 1
    $node_($i) start
}

#create nodes in 3rd neighborhood
for {set i 20} {$i < 30 } {incr i} {
    set node_($i) [$ns_ node]

    $node_($i) set X_ [expr $i + 400.0000000]
    $node_($i) set Y_ [expr $i + 400.0000000]
    $node_($i) set Z_ 0.0

    $node_($i) random-motion 1
    $node_($i) start
}
```

4.5.4. Σενάριο για ιεραρχική δρομολόγηση

Στην περίπτωση της ιεραρχικής δρομολόγησης, έχουμε 2 ειδών κόμβους. Αρχικά ορίζονται οι επικεφαλείς κάθε ομάδας, που είναι 3 κόμβοι για τους οποίους απενεργοποιούμε την τυχαία κίνηση, και ορίζουμε μεταξύ τους στατικές ζεύξεις.

```
# Configure for ForeignAgent and HomeAgent nodes
$ns_ node-config -mobileIP ON \
                 -adhocRouting $opt(adhocRouting) \
                 -llType $opt(ll) \
                 -macType $opt(mac) \
                 -ifqType $opt(ifq) \
                 -ifqLen $opt(ifqlen) \
                 -antType $opt(ant) \
                 -propType $opt(prop) \
                 -phyType $opt(netif) \
                 -channelType $opt(chan) \
        -topoInstance $topo \
                 -wiredRouting ON \
        -agentTrace ON \
                 -routerTrace OFF \
                 -macTrace OFF

# Create C1 and C2
set C1 [$ns_ node 1.0.0]
set C2 [$ns_ node 2.0.0]
set C3 [$ns_ node 3.0.0]
$C1 random-motion 0
$C2 random-motion 0
$C3 random-motion 0

# Position (fixed) for base-station nodes.
$C1 set X_ 200.000000000000
$C1 set Y_ 200.000000000000
$C1 set Z_ 0.000000000000

$C2 set X_ 800.000000000000
$C2 set Y_ 200.000000000000
$C2 set Z_ 0.000000000000

$C3 set X_ 500.000000000000
$C3 set Y_ 500.000000000000
$C3 set Z_ 0.000000000000


        #create links between BaseStation nodes
        $ns_ duplex-link $C1 $C2 50Mb 2ms DropTail
        $ns_ duplex-link $C1 $C3 50Mb 2ms DropTail
        $ns_ duplex-link $C2 $C3 50Mb 2ms DropTail
```

Στη συνέχεια ορίζουμε τους κινούμενους κόμβους. Οι κόμβοι αυτοί έχουν διεύθυνση I.P. σχετική με αυτή του κεντρικού κόμβου σε κάθε ομάδα. Ορίζεται αρχικά ένας πίνακας με όλες τις διευθύνσεις, και στη συνέχεια, δίνεται διεύθυνση σε κάθε κόμβο. Για παράδειγμα, στην πρώτη ομάδα, όπου ο κεντρικός κόμβος C1 έχει διεύθυνση 1.0.0 οι δευτερεύοντες κόμβοι έχουν διευθύνσεις από 1.0.1 έως 1.0.10.

```
# mobile nodes address list
set temp1 {1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10 2.0.1 2.0.2 2

#create nodes in same cluster as C1
for {set i 0} {$i < 10 } {incr i} {
    set node_($i) [$ns_ node [lindex $temp1 $i]]
    set Nodeaddress [AddrParams addr2id [$C1 node-addr]]
    [$node_($i) set regagent_] set home_agent_ $Nodeaddress

    $node_($i) set X_ [expr $i + 200.0000000]
    $node_($i) set Y_ [expr $i + 200.0000000]
    $node_($i) set Z_ 0.0

    $node_($i) random-motion 1
    $node_($i) start
}

#create nodes in same cluster as C2
for {set i 10} {$i < 20 } {incr i} {
    set node_($i) [$ns_ node [lindex $temp1 $i]]
    $node_($i) random-motion 1
    set Nodeaddress [AddrParams addr2id [$C2 node-addr]]
    [$node_($i) set regagent_] set home_agent_ $Nodeaddress

    $node_($i) set X_ [expr $i + 800.0000000]
    $node_($i) set Y_ [expr $i + 200.0000000]
    $node_($i) set Z_ 0.0

    $node_($i) random-motion 1
    $node_($i) start
}

#create nodes in same cluster as C3
for {set i 20} {$i < 30 } {incr i} {
    set node_($i) [$ns_ node [lindex $temp1 $i]]
    set Nodeaddress [AddrParams addr2id [$C3 node-addr]]
    [$node_($i) set regagent_] set home_agent_ $Nodeaddress

    $node_($i) set X_ [expr $i + 500.0000000]
    $node_($i) set Y_ [expr $i + 500.0000000]
    $node_($i) set Z_ 0.0

    $node_($i) random-motion 1
    $node_($i) start
}
```

4.5.5. Σενάρια κανονικής κίνησης και ανωμαλιών

Στη συνέχεια φορτώνονται τα αρχεία που περιέχουν τις τυχαίες συνδέσεις ανάμεσα στους κόμβους. Χρησιμοποιούνται ένα ή δύο αρχεία για την δημιουργία της κανονικής κίνησης (background traffic) και ένα αρχείο για την προσομοίωση της ανωμαλίας.

```
# load background traffic
source tcp-scenario.tcl
source udp-scenario.tcl

#load anomaly
source anomaly.tcl
```

Ακολουθούν παραδείγματα ορισμού δικτυακής κίνησης. Στο επόμενο παράδειγμα δημιουργείται σύνδεση F.T.P. (File Transfer Protocol) ανάμεσα σε δύο κόμβους.

```
# 1 connecting to 2 at time 35.786254813795097
#
set tcp_(0) [$ns_ create-connection  TCP $node_(1) TCPSink $node_(2) 0]
$tcp_(0) set window_ 32
$tcp_(0) set packetSize_ 512
set ftp_(0) [$tcp_(0) attach-source FTP]
$ns_ at 35.786254813795097 "$ftp_(0) start"
```

Για τη δημιουργία ροής U.D.P. πακέτων ανάμεσα σε δύο κόμβους παρατίθεται το επόμενο παράδειγμα.

```
# 1 connecting to 2 at time 35.786254813795097
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.25
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 35.786254813795097 "$cbr_(0) start"
```

Για τη δημιουργία των ανωμαλιών, σε ξεχωριστό αρχείο δημιουργείται δικτυακή κίνηση ανάλογα με το είδος της ανωμαλίας που θέλουμε να προσομοιώσουμε.

4.5.6. Τελικές ρυθμίσεις συστήματος

Τέλος, ορίζεται η αρχή και το τέλος της προσομοίωσης και το κλείσιμο μεταβλητών και αρχείων.

```
# Tell all nodes when the siulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).0 "$node_($i) reset";
}

$ns_ at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"
$ns_ at $opt(stop).0001 "stop"

proc stop {} {
    global ns_ tracefd
    close $tracefd
}

# some useful headers for tracefile
puts $tracefd "adhoc routing scenario"
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp \
    $opt(adhocRouting)   seed $opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```

4.5.7. Παρσάρισμα του αρχείου καταγραφής του N.S. και των αρχείων επεξεργασίας των δεδομένων

Κατά τη διάρκεια της προσομοίωσης, το N.S. αποθηκεύει την κίνηση που δημιουργεί υπό μορφή αποστολής ή απολαβής πακέτων από κάποιον κόμβο. Η μορφοποίηση του αρχείου έχει την παρακάτω μορφή:

```
s -t 0.022530246 -Hs 28 -Hd -1 -Ni 28 -Nx 0.00 -Ny 0.00 -
Nz 0.00 -Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -
Mt 0 -Is 28.255 -Id -1.255 -It message -Il 32 -If 0 -Ii 0
-Iv 32


r -t 0.023810246 -Hs 27 -Hd -1 -Ni 27 -Nx 0.00 -Ny 0.00 -
Nz 0.00 -Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md ffffffff -
Ms 1c -Mt 800 -Is 28.255 -Id -1.255 -It message -Il 32 -If
0 -Ii 0 -Iv 32
.
```

Κάθε γραμμή είναι ένα συμβάν, που αφορά ένα συγκεκριμένο πακέτο. Πιο συγκεκριμένα, τα πεδία έχουν το παρακάτω νόημα:

```
Field 0: event type
   s: send     r: receive  d: drop     f: forward
Field 1: General tag
     -t: time
Field 3: Next hop info
      -Hs: id for this node
     -Hd: id for next hop towards the destination
Field 4: Node property type tag
     -Ni: node id
     -Nx -Ny -Nz: node's x/y/z coordinate
     -Ne: node energy level
     -Nl: trace level, such as AGT, RTR, MAC
     -Nw: reason for the event
Field 5: Packet information at IP level
     -Is: source address. Source port number
     -Id: dest address.dest port number
     -It: packet type
      -Il: packet size
     -If: flow id
     -Ii: unique id
     -Iv: ttl value
Field 6: packet info at MAC level
   -Ma: duration
    -Md: dest's ethernet address
   -Ms: src's ethernet address
   -Mt: ethernet type
Field 7: Packet info at "Application level" which consists
of the type of application like arp, tcp, ack...
```

Για την υλοποίηση του αλγόριθμου ανίχνευσης ανωμαλιών χρειάστηκαν τρία διαφορετικά βήματα. Το πρώτο βήμα είναι η επεξεργασία του αρχείου καταγραφής του N.S. και η συλλογή της πληροφορίας που μας ενδιαφέρει. Το σύνολο των γνωρισμάτων της δικτυακής κίνησης που χρησιμοποιήθηκαν είναι τα ακόλουθα: source address, destination address, destination port, packet length, protocol.

Η υλοποίηση του προγράμματος επεξεργασίας των δεδομένων είναι σε Perl scripts και ονομάζεται «getResults.pl». Το script καλείται ως **«getResults.pl trace.tr trace.txt»**, όπου το «trace.tr» είναι το αρχείο καταγραφής του N.S. και το «trace.txt» είναι το αρχείο εξόδου του προγράμματος.

Αυτό το script επεξεργάζεται κάθε γραμμή και κρατά μονάχα όσες αρχίζουν από r (receive), (δηλαδή μόνο για πακέτα που έχουν παραληφθεί με επιτυχία) και αφορούν πακέτα επικοινωνίας ανάμεσα σε κόμβους (είτε T.C.P. είτε U.D.P.), αδιαφορώντας για τα πακέτα σηματοδοσίας που έχουν να κάνουν με τη δρομολόγηση. Στη συνέχεια, κρατά τα πεδία που αφορούν τα χαρακτηριστικά της κίνησης που αναφέραμε προηγουμένως καθώς και το χρόνο του συμβάντος και αποθηκεύει τα αποτελέσματα στο αρχείο εξόδου. Η έξοδος από το πρόγραμμα αυτό έχει την παρακάτω μορφή:

time	**protocol**	**source address**	**destination address**	**packet length**	**destination port**
6.506205912	1	10	11	40	0
6.511456912	1	10	11	60	0
6.511456912	2	11	10	40	0
6.513520912	2	11	10	60	0
6.513520912	1	10	11	552	1
6.513520912	1	11	11	552	2
6.519940912	1	11	11	572	1
6.519940912	2	10	10	40	1
6.522164912	2	10	10	60	1

Το επόμενο βήμα είναι η υποδιαίρεση του χρόνου σε σχισμές (time bins) και η εξαγωγή της εντροπίας για κάθε κατανομή. Η διαδικασία αυτή γίνεται από ένα άλλο perl script το οποίο ονομάζεται **«getEntropies.pl»**. Δέχεται ως όρισμα το αρχείο εξόδου από το «getResults.pl» και εμφανίζει στο standard output την ακόλουθη μορφή αποτελεσμάτων:

```
adhoc routing scenario
M 0.0 nn 30 x 1000 y 1000 rp  DSDV   seed 123.0
M 0.0 prop Propagation/TwoRayGround ant Antenna/OmniAntenna

time    Source(IP)          Dest(IP)            Packet Length       Dest(ports)
20      0.157997553323609   0.157997553323609   0.158069929675923   0.766532617557327
40      0.17445141232091    0.17445141232091    0.151489029848617   0.782068777371331
60      0.215233495780343   0.215379051674588   0.151604716512765   0.786428155404009
80      0.244683629032725   0.245149921401407   0.151429869393321   0.788442024607478
100     0.250828410235162   0.251174459695486   0.151634309942203   0.773461109488929
120     0.26082446093467    0.261139572752419   0.151748280879842   0.768749084075584
140     0.304577058377477   0.305300579013413   0.151676100448683   0.753993476839353
160     0.30875753830378    0.308800327171655   0.151853392046021   0.773720229425483
180     0.312940553013044   0.312931997355591   0.1518146164201 0.782005884486352
```

Κάθε γραμμή αντιστοιχεί σε μια διαφορετική χρονοσχισμή και κάθε στήλη αντιστοιχεί στην τιμή της εντροπίας της κατανομής για κάποιο συγκεκριμένο γνώρισμα. Το script αυτό επεξεργάζεται την έξοδο από το «getResults.pl» και για κάθε γνώρισμα, διατηρεί ένα διαφορετικό hash table. Τα κλειδιά του hash table είναι οι πιθανές τιμές του γνωρίσματος, ενώ η τιμή που αντιστοιχεί στο κλειδί είναι ένας μετρητής. Σε κάθε γραμμή, ένα κλειδί προτίθεται ή ενημερώνεται δηλαδή αυξάνεται η τιμή του κατά 1. Στο τέλος της χρονοσχισμής, υπολογίζεται η εντροπία από τις σχετικές συχνότητες εμφάνισης κάθε γνωρίσματος, με βάση το αντίστοιχο hash table. Πριν περάσουμε στην επόμενη χρονοσχισμή, όλα τα hash table μηδενίζονται.

Το τελικό βήμα είναι η χρήση των τιμών της εντροπίας για την ανίχνευση ανωμαλιών. Με βάση τον αλγόριθμο που παρουσιάστηκε σε προηγούμενη ενότητα, ένα νέο script, το **«anomalyDetection.pl»** ελέγχει το λόγο ανάμεσα στην εντροπία της κατανομής των I.P. διευθύνσεων προορισμού και στην εντροπία της κατανομής των I.P. διευθύνσεων προέλευσης. Σε κάθε βήμα γίνεται προσπάθεια ανίχνευσης επίθεσης D.D.o.S., εξάπλωσης worm και flash crowd.

Ακολουθεί έξοδος του προγράμματος που έχει ανιχνεύσει ανωμαλία κατά τη χρονοσχισμή 80-120.

anomalyDetection.pl attack.txt

time	**Attack type**	**Info**
80:	**WORM ALERT:**	**Source IP entropy: 0.162**
		Dest IP entropy: 0.2302
		Ratio:0.705146

Καθώς ο λόγος είναι πολύ μικρότερος από 1, ο αλγόριθμος ανιχνεύει την ανωμαλία.

4.6. Αριθμητικά αποτελέσματα

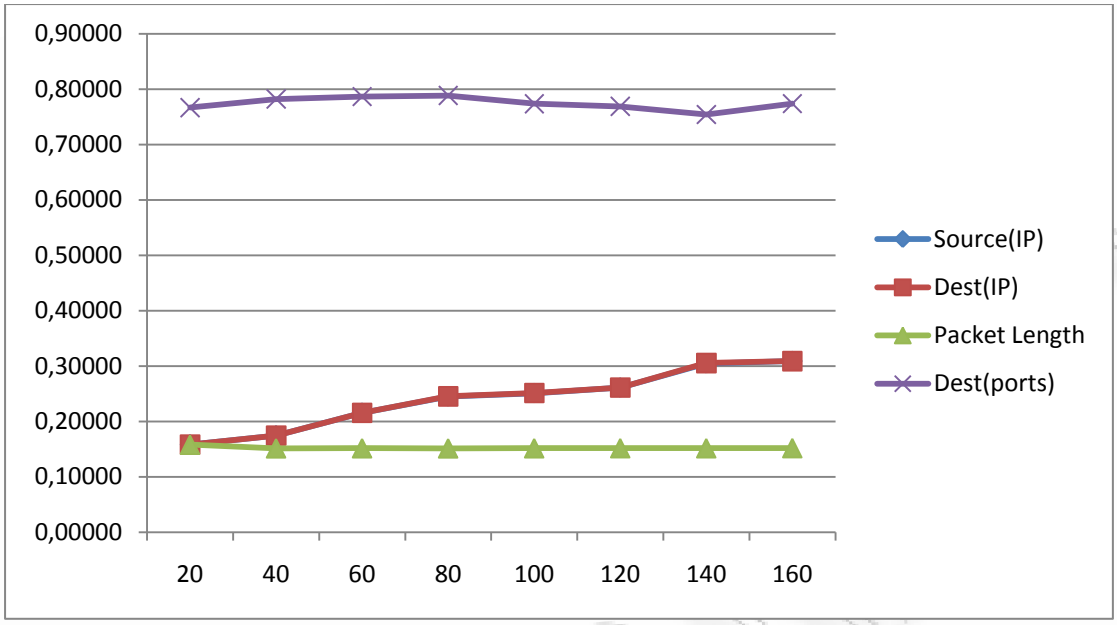
4.6.1. Δρομολόγηση ad–hoc

Στην ενότητα αυτή παρουσιάζουμε τα αριθμητικά αποτελέσματα για τρία διαφορετικά σενάρια ανωμαλίας. Το πρώτο σενάριο αντιστοιχεί σε μια επίθεση D.D.o.S. , το δεύτερο σενάριο προσομοιώνει το αρχικό στάδιο της εξάπλωσης ενός worm ενώ το τελευταίο σενάριο προσομοιώνει την περίπτωση ενός flash crowd. Σε κάθε σενάριο, η κανονική κίνηση είναι η ίδια (U.D.P. και T.C.P. κίνηση ανάμεσα σε διάφορους κόμβους του δικτύου) και αλλάζει μονάχα η κίνηση που προκαλείται από την ανωμαλία. Φυσικά, η τελική κίνηση σαν άθροισμα των 2 προηγούμενων είναι διαφορετική σε κάθε περίπτωση.

4.6.2. 1ο σενάριο - Επίθεση D.D.o.S.

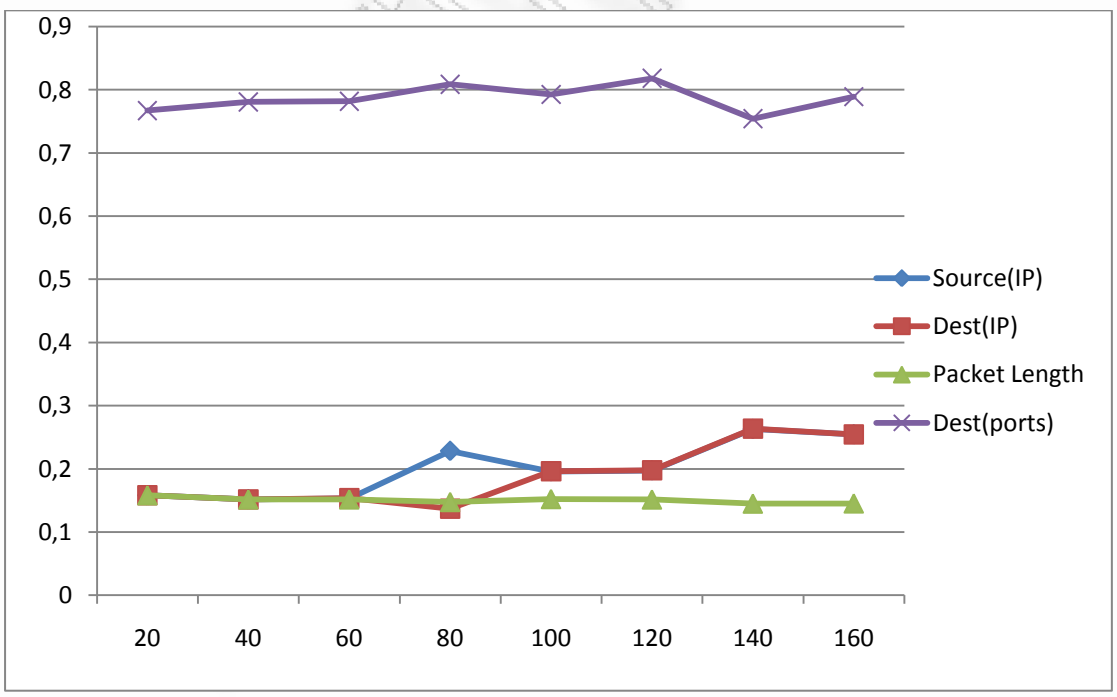
Το πρώτο σενάριο αντιστοιχεί σε μια επίθεση τύπου D.D.o.S., που δημιουργείται από διάφορους κόμβους στο δίκτυο και στοχεύει σε ένα συγκεκριμένο κόμβο του δικτύου. Η επίθεση ξεκινά μετά τη χρονική στιγμή 60 και διαρκεί για λιγότερο από 20 δευτερόλεπτα. Τα πακέτα της επίθεσης είναι U.D.P. με διαφορετική I.P. πηγής και κοινή I.P προορισμού.

Τα ακόλουθα διαγράμματα παρουσιάζουν τις εντροπίες των κατανομών σαν συναρτήσεις τους χρόνου. Το πρώτο διάγραμμα αντιστοιχεί μονάχα σε κανονική κίνηση ενώ το δεύτερο διάγραμμα αντιστοιχεί στην επίθεση.



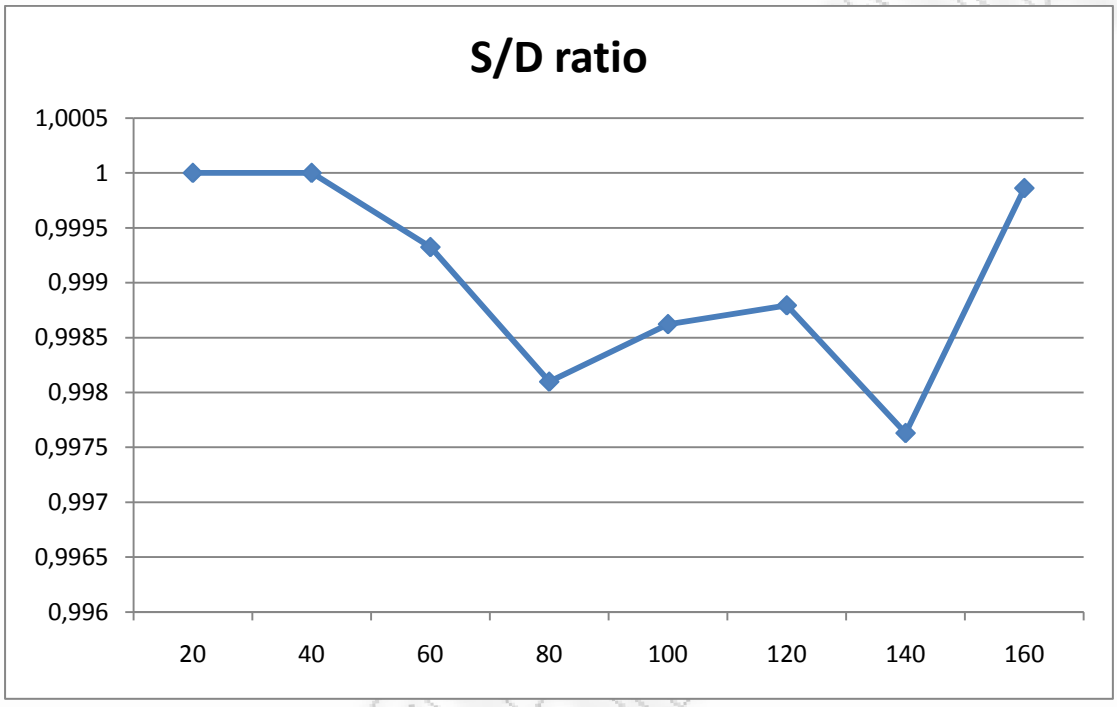
Εικόνα 13 Εντροπίες στην κανονική κίνηση

Η επίθεση ανιχνεύεται εύκολα από τη διαφορά ανάμεσα στις εντροπίες των source I.P. και destination I.P.

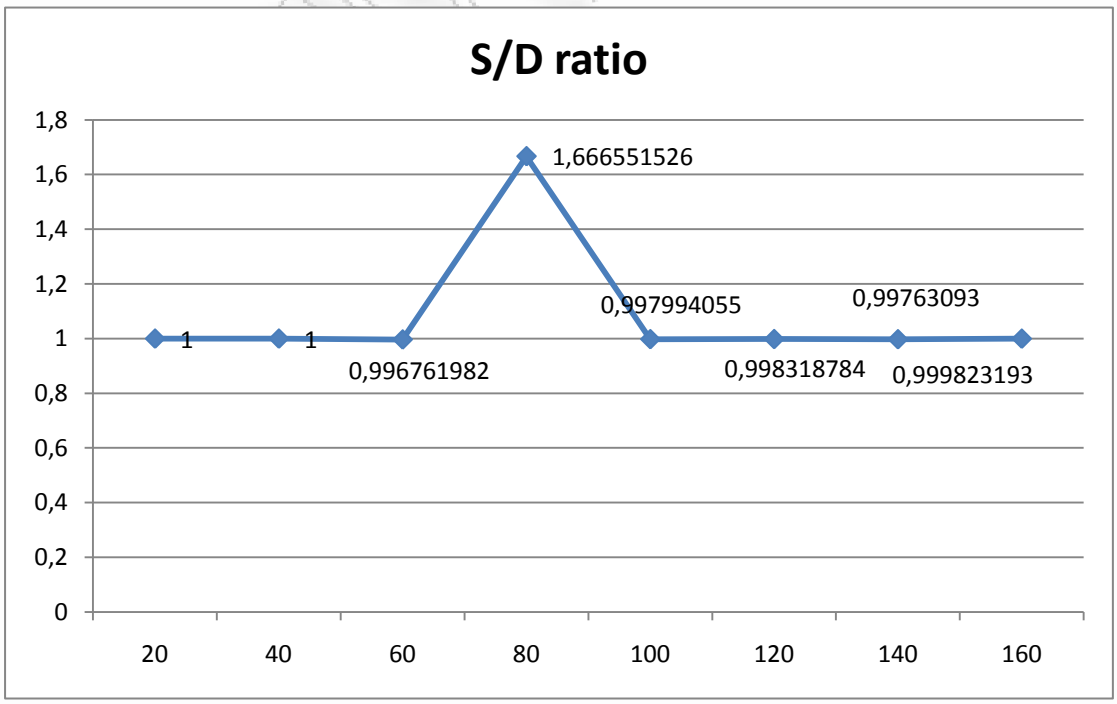


Εικόνα 14 Εντροπίες στο σενάριο D.D.o.S.

Στο επόμενο διάγραμμα παρουσιάζουμε την αναλογία S/D (Source address entropy / Destination address) entropy για το σενάριο D.D.o.S. . Η αύξηση στο λόγο S/D αποκαλύπτει την επίθεση D.D.o.S.



Εικόνα 15 Λόγος S/D στην κανονική κίνηση

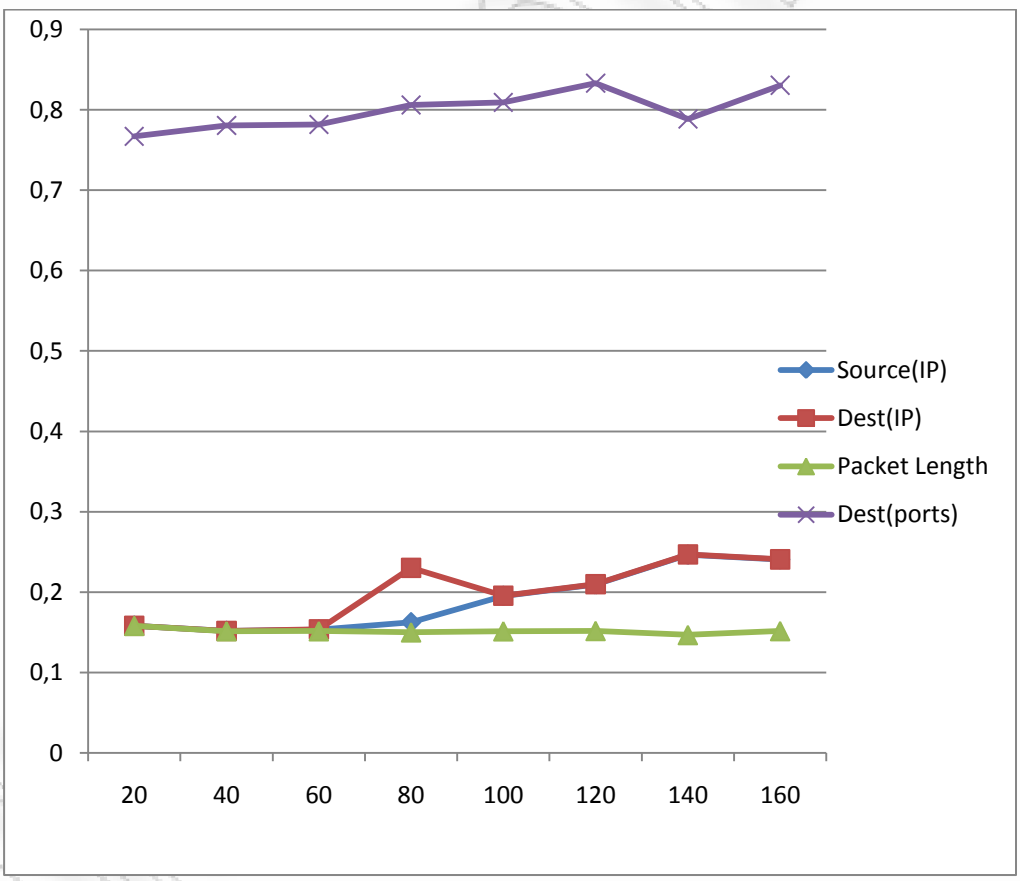


Εικόνα 16 Λόγος S/D για το σενάριο επίθεσης D.D.o.S.

4.6.3. 2ο σενάριο – Εξάπλωση Worm

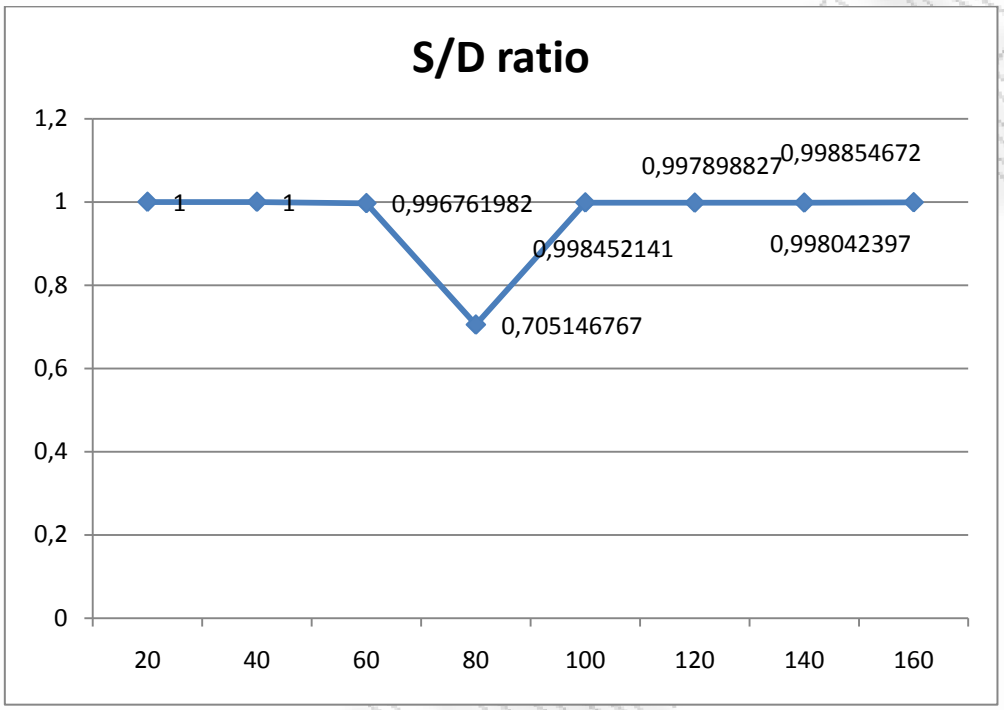
Το δεύτερο σενάριο αντιστοιχεί σε εξάπλωση ενός worm που εκκινείτε από έναν μολυσμένο κόμβο στο δίκτυο και στοχεύει σε ένα μεγάλο πλήθος άλλων κόμβων του δικτύου. Η επίθεση ξεκινά μετά τη χρονική στιγμή 60 και διαρκεί για λιγότερο από 20 δευτερόλεπτα. Τα πακέτα της επίθεσης είναι U.D.P. με κοινή I.P. πηγής και διαφορετική I.P. προορισμού.

Τα ακόλουθα διαγράμματα παρουσιάζουν τις εντροπίες των κατανομών σαν συναρτήσεις τους χρόνου.



Εικόνα 17 Εντροπίες για το σενάριο εξάπλωσης worm

Στο επόμενο διάγραμμα παρουσιάζουμε την αναλογία S/D (Source address entropy / Destination address) entropy για το σενάριο D.D.o.S. . Η μείωση στο λόγο S/D αποκαλύπτει το αρχικό στάδιο εξάπλωσης ενός worm.

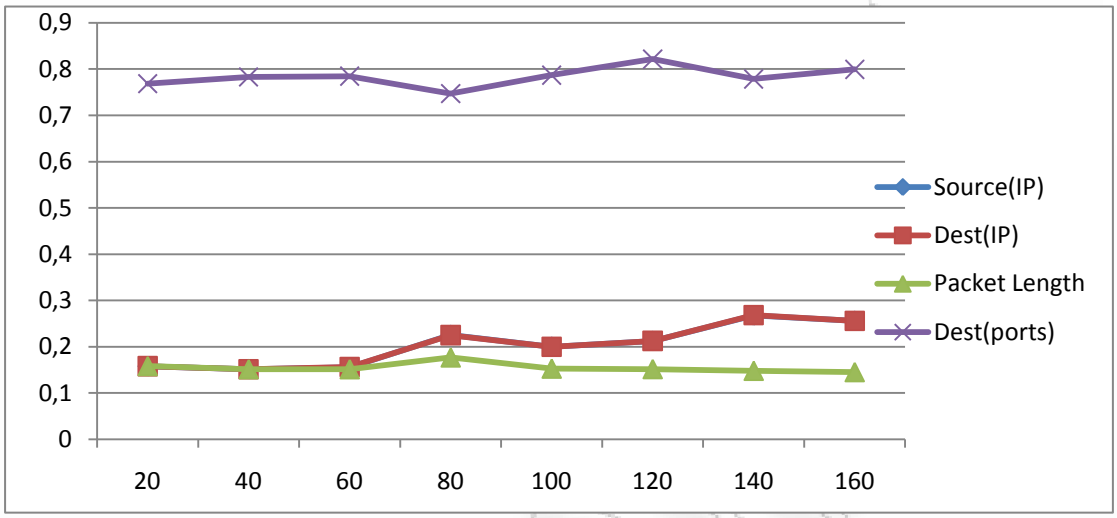


Εικόνα 18 Λόγος S/D για το σενάριο εξάπλωσης worm

4.6.4. 3ο σενάριο – Flash crowd

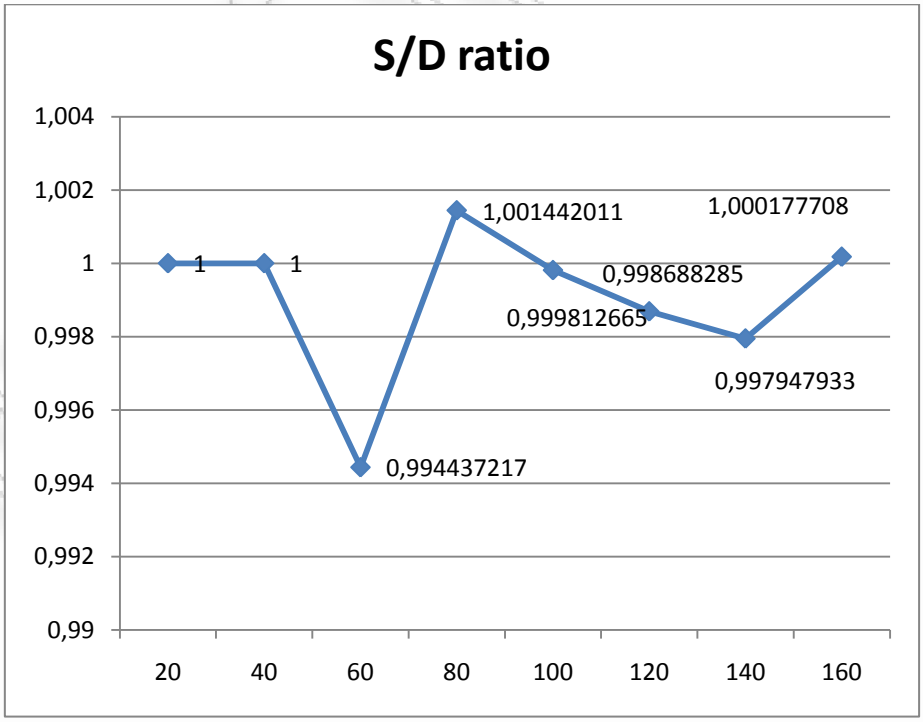
Το τρίτο σενάριο αντιστοιχεί στη περίπτωση ενός flash crowd. Πρόκειται για την περίπτωση όπου διάφοροι χρήστες στο δίκτυο προσπαθούν να επικοινωνήσουν με ένα συγκεκριμένο κόμβο και δημιουργούν μεγάλες ροές πληροφορίες από τον κόμβο αυτό προς τους ίδιους. Η ανωμαλία ξεκινά μετά τη χρονική στιγμή 60 και διαρκεί για λιγότερο από 20 δευτερόλεπτα. Αν και το flash crowd μπορεί να θεωρηθεί ανωμαλία, δεν είναι επίθεση καθώς δε δημιουργείται από κακόβουλους χρήστες. Συνεπώς, ο αλγόριθμος ανίχνευσης δε θα έπρεπε να χαρακτηρίζει την αύξηση στην κίνηση που προκαλείται στην περίπτωση αυτή ως ανωμαλία. Η ανωμαλία αποτελείται από T.C.P. συνδέσεις από διάφορους κόμβους προς κάποιον F.T.P. server.

Το ακόλουθο διάγραμμα παρουσιάζει τις εντροπίες της κατανομής των γνωρισμάτων της κίνησης ως συναρτήσεις του χρόνου.



Εικόνα 19 Εντροπίες στο σενάριο flash crowd

Όπως φαίνεται από το παρακάτω διάγραμμα, η αλλαγή στο λόγο S/D κατά τη διάρκεια της ανωμαλίας δεν επηρεάζεται αρκετά ώστε να θεωρηθεί ως επίθεση. Συνεπώς, ο αλγόριθμος σωστά δεν ανιχνεύει κάποια επίθεση στη συγκεκριμένη περίπτωση.



Εικόνα 20 Λόγος S/D για το σενάριο flash crowd

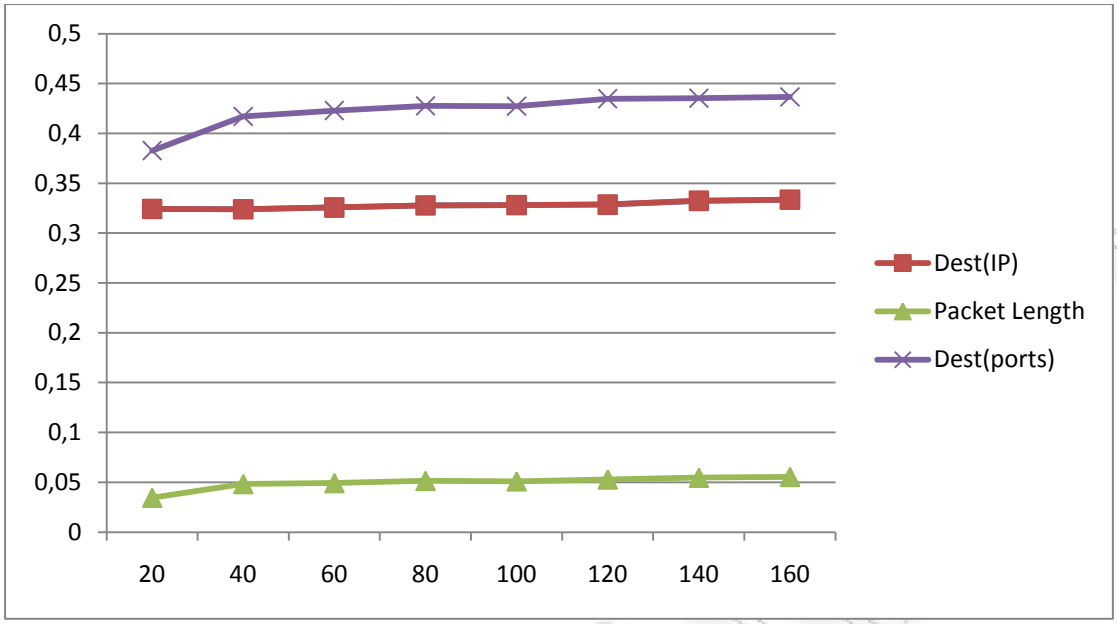
4.7. Ιεραρχική Δρομολόγηση

Στην ενότητα αυτή παρουσιάζουμε τα αριθμητικά αποτελέσματα για τρία διαφορετικά σενάρια ανωμαλίας. Το πρώτο σενάριο αντιστοιχεί σε μια επίθεση D.D.o.S., το δεύτερο σενάριο προσομοιώνει το αρχικό στάδιο της εξάπλωσης ενός worm ενώ το τελευταίο σενάριο προσομοιώνει την περίπτωση ενός flash crowd. Σε κάθε σενάριο, η κανονική κίνηση είναι η ίδια (U.D.P. και T.C.P. κίνηση ανάμεσα σε διάφορους κόμβους του δικτύου) και αλλάζει μονάχα η κίνηση που προκαλείται από την ανωμαλία. Φυσικά, η τελική κίνηση σαν άθροισμα των 2 προηγούμενων είναι διαφορετική σε κάθε περίπτωση.

4.7.1. 1ο σενάριο – Επίθεση D.D.o.S.

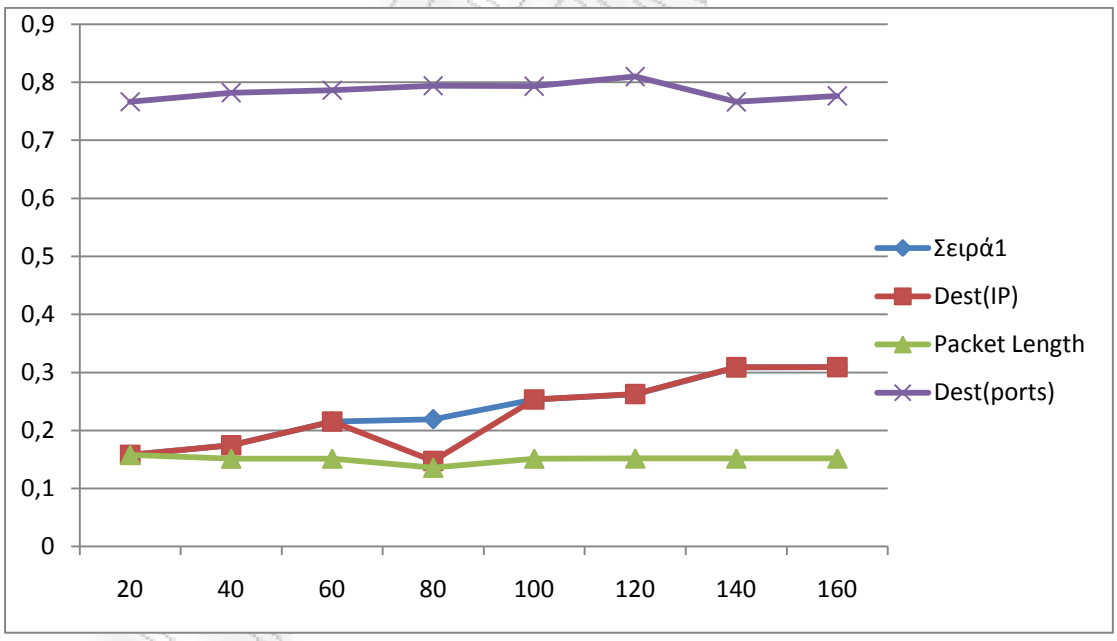
Το πρώτο σενάριο αντιστοιχεί σε μια επίθεση τύπου D.D.o.S. που δημιουργείται από διάφορους κόμβους στο δίκτυο και στοχεύει σε ένα συγκεκριμένο κόμβο του δικτύου. Η επίθεση ξεκινά μετά τη χρονική στιγμή 60 και διαρκεί για λιγότερο από 20 δευτερόλεπτα. Τα πακέτα της επίθεσης είναι U.D.P. με διαφορετική I.P. πηγής και κοινή I.P. προορισμού.

Τα ακόλουθα διαγράμματα παρουσιάζουν τις εντροπίες των κατανομών σαν συναρτήσεις τους χρόνου. Το πρώτο διάγραμμα αντιστοιχεί μονάχα σε κανονική κίνηση ενώ το δεύτερο διάγραμμα αντιστοιχεί στην επίθεση.



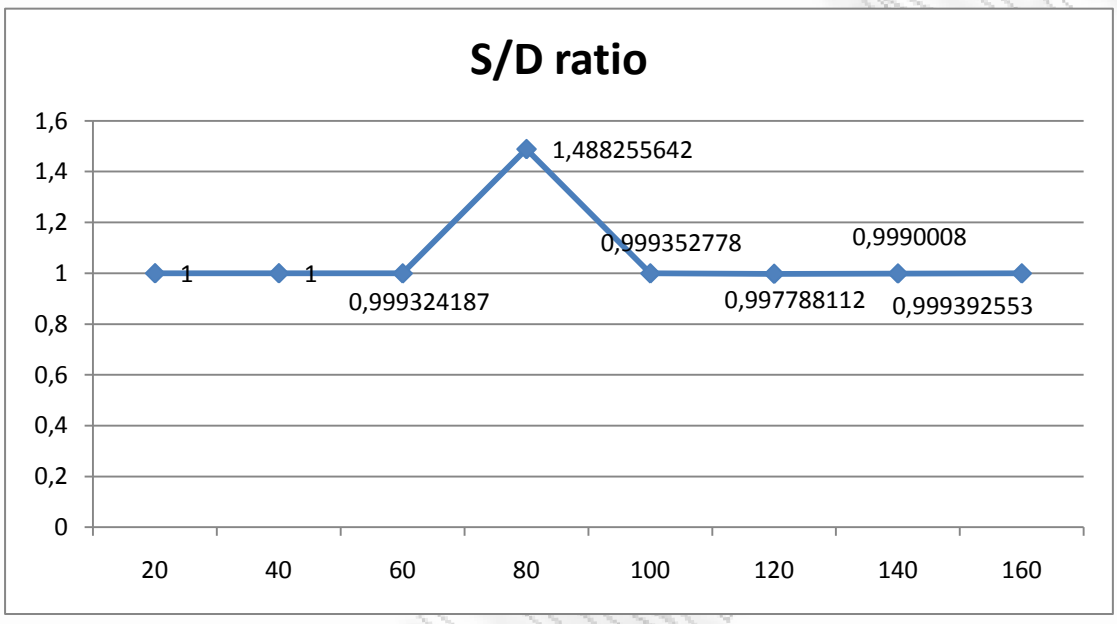
Εικόνα 21 Εντροπίες στην κανονική κίνηση

Η επίθεση ανιχνεύεται εύκολα από τη διαφορά ανάμεσα στις εντροπίες των source I.P. και destination I.P.



Εικόνα 22 Εντροπίες για το σενάριο D.D.o.S.

Στο επόμενο διάγραμμα παρουσιάζουμε την αναλογία S/D (Source address entropy / Destination address) entropy για το σενάριο D.D.o.S. . Η αύξηση στο λόγο S/D αποκαλύπτει την επίθεση D.D.o.S. .

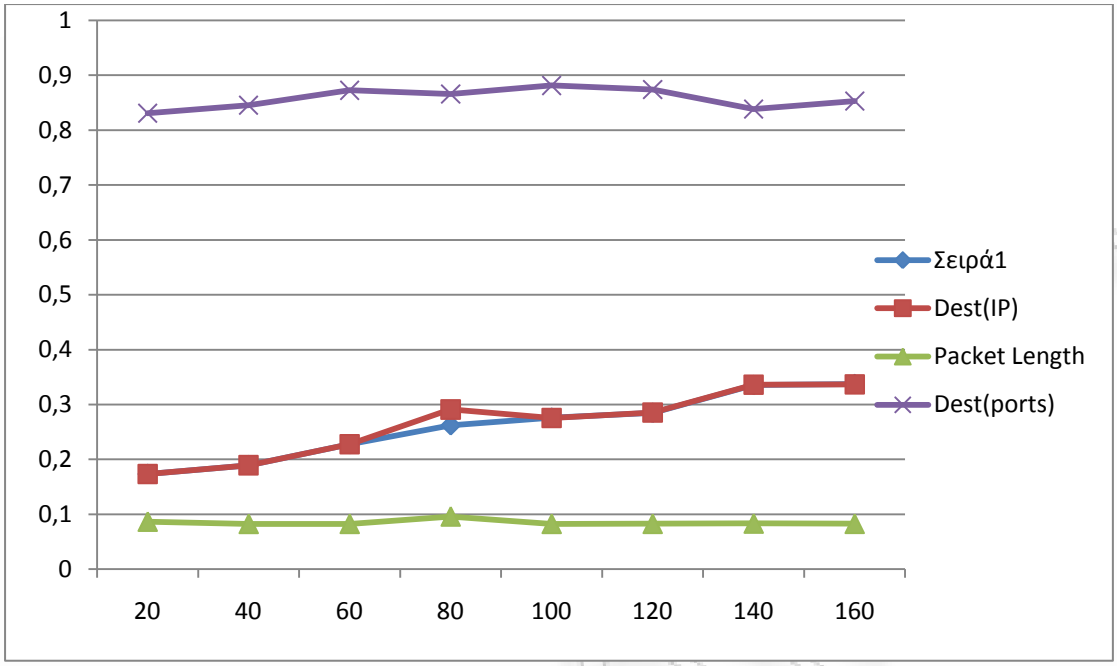


Εικόνα 23 Λόγος S/D για το σενάριο επίθεσης D.D.o.S.

4.7.2. 2ο σενάριο – Εξάπλωση worm

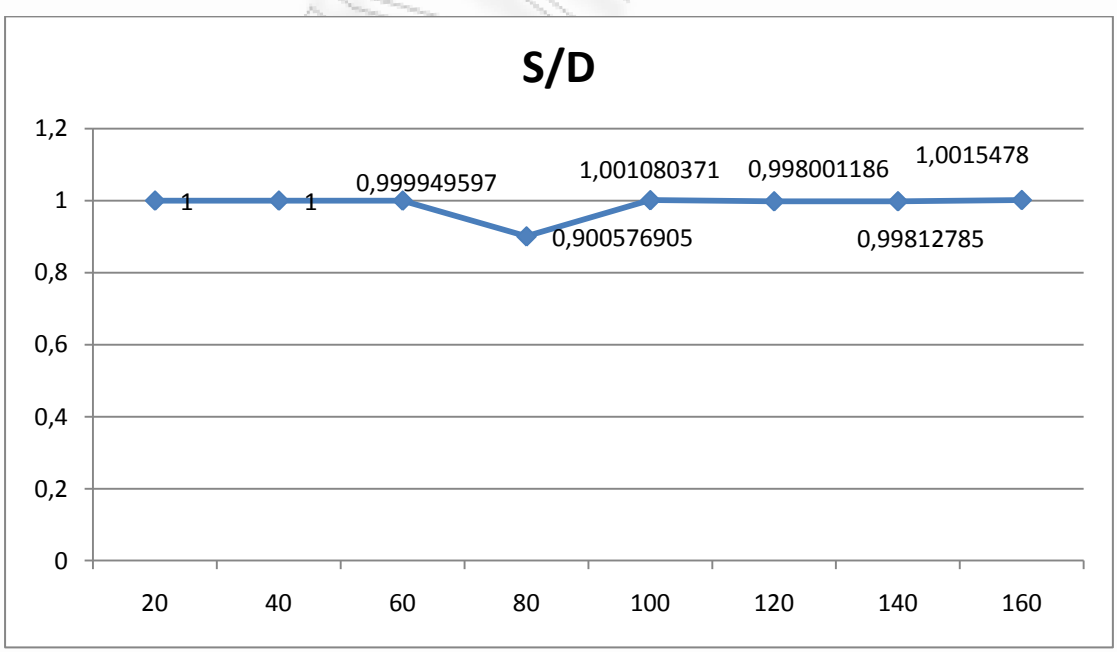
Το δεύτερο σενάριο αντιστοιχεί σε εξάπλωση ενός worm που εκκινείτε από έναν μολυσμένο κόμβο στο δίκτυο και στοχεύει σε ένα μεγάλο πλήθος άλλων κόμβων του δικτύου. Η επίθεση ξεκινά μετά τη χρονική στιγμή 60 και διαρκεί για λιγότερο από 20 δευτερόλεπτα. Τα πακέτα της επίθεσης είναι U.D.P. με κοινή I.P. πηγής και διαφορετική I.P. προορισμού.

Τα ακόλουθα διαγράμματα παρουσιάζουν τις εντροπίες των κατανομών σαν συναρτήσεις τους χρόνου.



Εικόνα 24 Εντροπίες για το σενάριο worm propagation

Στο επόμενο διάγραμμα παρουσιάζουμε την αναλογία S/D (Source address entropy / Destination address) entropy για το σενάριο D.D.o.S. . Η μείωση στο λόγο S/D αποκαλύπτει το αρχικό στάδιο εξάπλωσης ενός worm. Στη περίπτωση αυτή παρατηρούμε ότι ο αλγόριθμος ανίχνευσης δεν αποδίδει το ίδιο καλά όσο στην περίπτωση της δρομολόγησης ad hoc.

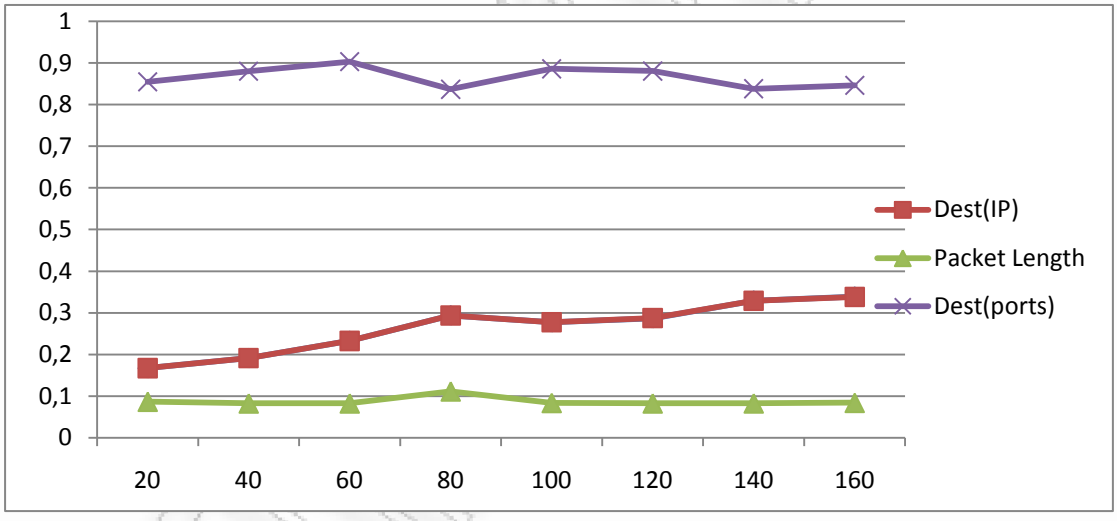


Εικόνα 25 Λόγος S/D για το σενάριο εξάπλωσης worm

4.7.3. 3ο σενάριο – Flash crowd

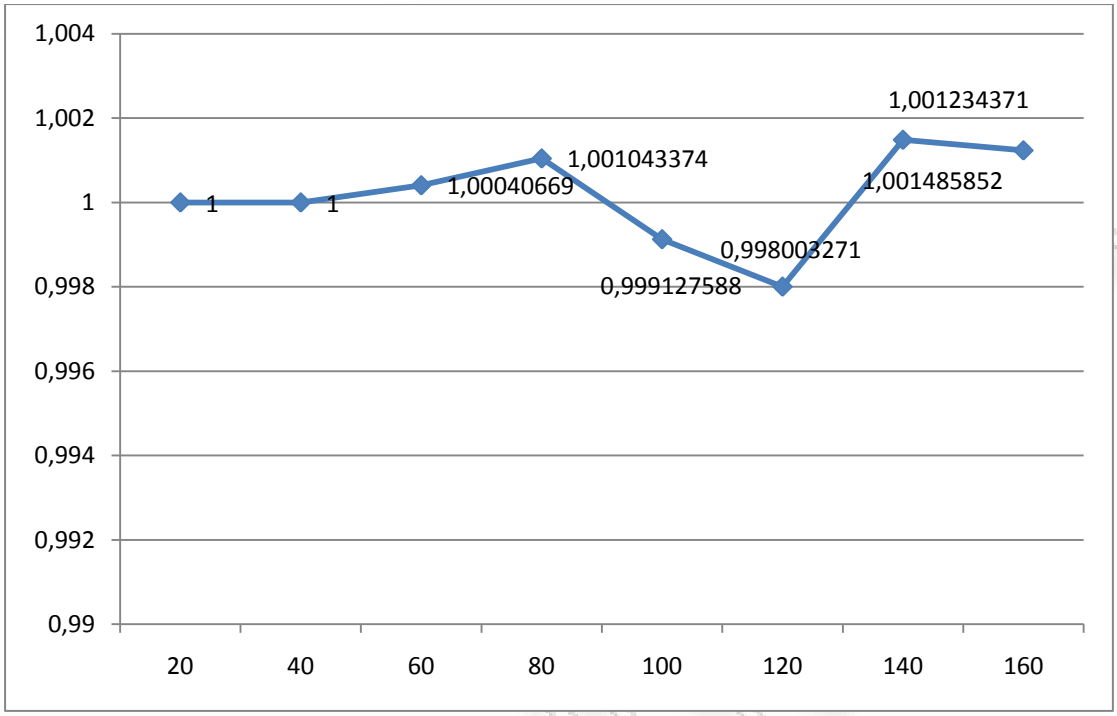
Το τρίτο σενάριο αντιστοιχεί στη περίπτωση ενός flash crowd. Η ανωμαλία ξεκινά μετά τη χρονική στιγμή 60 και διαρκεί για λιγότερο από 20 δευτερόλεπτα. Αν και το flash crowd μπορεί να θεωρηθεί ανωμαλία, δεν είναι επίθεση καθώς δε δημιουργείται από κακόβουλους χρήστες. Συνεπώς, ο αλγόριθμος ανίχνευσης δε θα έπρεπε να χαρακτηρίζει την αύξηση στην κίνηση που προκαλείται στην περίπτωση αυτή ως ανωμαλία. Η ανωμαλία αποτελείται από T.C.P. συνδέσεις από διάφορους κόμβους προς κάποιον F.T.P. server.

Το ακόλουθο διάγραμμα παρουσιάζει τις εντροπίες της κατανομής των γνωρισμάτων της κίνησης ως συναρτήσεις του χρόνου.



Εικόνα 26 Εντροπίες για το σενάριο flash crowd

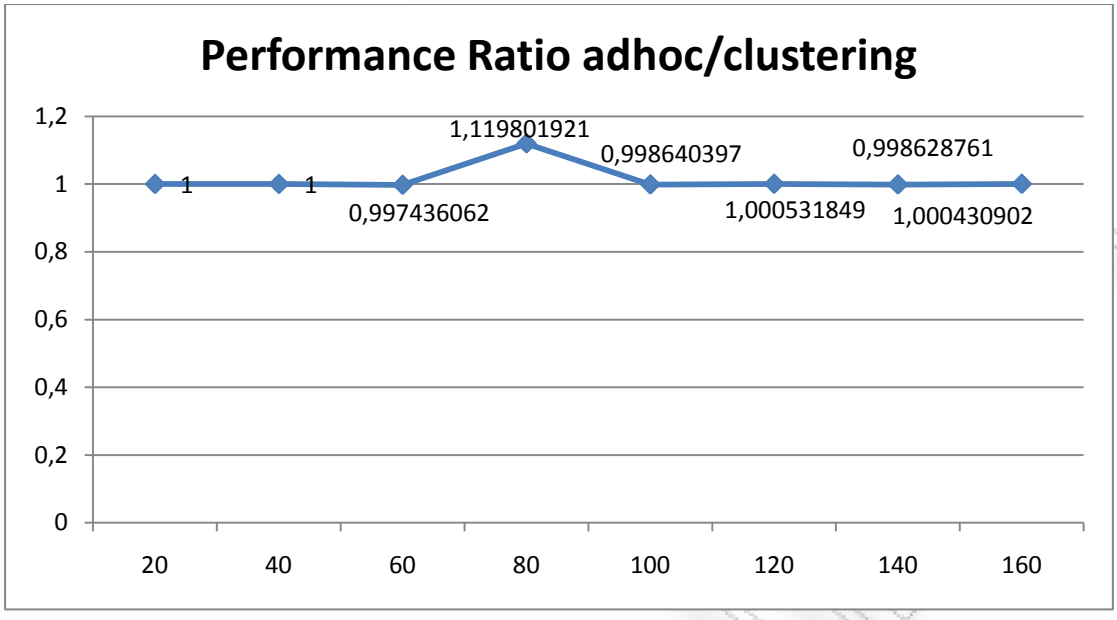
Όπως φαίνεται από το παρακάτω διάγραμμα, η αλλαγή στο λόγο S/R κατά τη διάρκεια της ανωμαλίας δεν επηρεάζεται αρκετά ώστε να θεωρηθεί ως επίθεση. Συνεπώς, ο αλγόριθμος σωστά δεν ανιχνεύει κάποια επίθεση στη συγκεκριμένη περίπτωση.



Εικόνα 27 Λόγος S/D για το σενάριο flash crowd

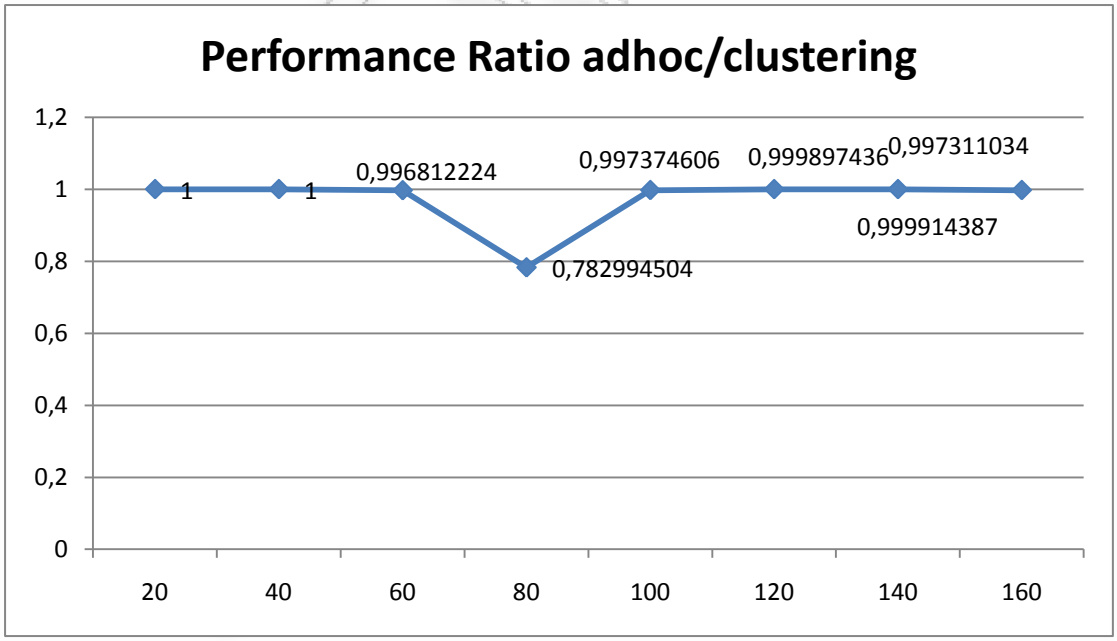
4.8. Συγκριτικά αποτελέσματα και σχολιασμός

Κάνοντας μια σειρά από πειράματα, καταλήξαμε στα παρακάτω συγκεντρωτικά διαγράμματα. Σε κάθε ένα από τα παρακάτω διαγράμματα φαίνεται ο λόγος απόδοσης του αλγορίθμου, όταν η δρομολόγηση γίνεται ad hoc και όταν η δρομολόγηση γίνεται ιεραρχικά. Και στις 2 περιπτώσεις το μέγεθος της επίθεσης είναι το ίδιο, και συμβαίνει πάντα ανάμεσα στις χρονικές στιγμές 80-100.



Εικόνα 28 Λόγος απόδοσης του αλγόριθμου σε επίθεση D.D.o.S. και διαφορετική δρομολόγηση

Παρατηρούμε ότι στην περίπτωση επιθέσεων D.D.o.S., ο αλγόριθμος αποδίδει χειρότερα στην περίπτωση της ιεραρχικής δρομολόγησης. Μάλιστα, στην περίπτωση της δρομολόγησης ad hoc η απόδοση του αλγορίθμου είναι περίπου 12% καλύτερη.

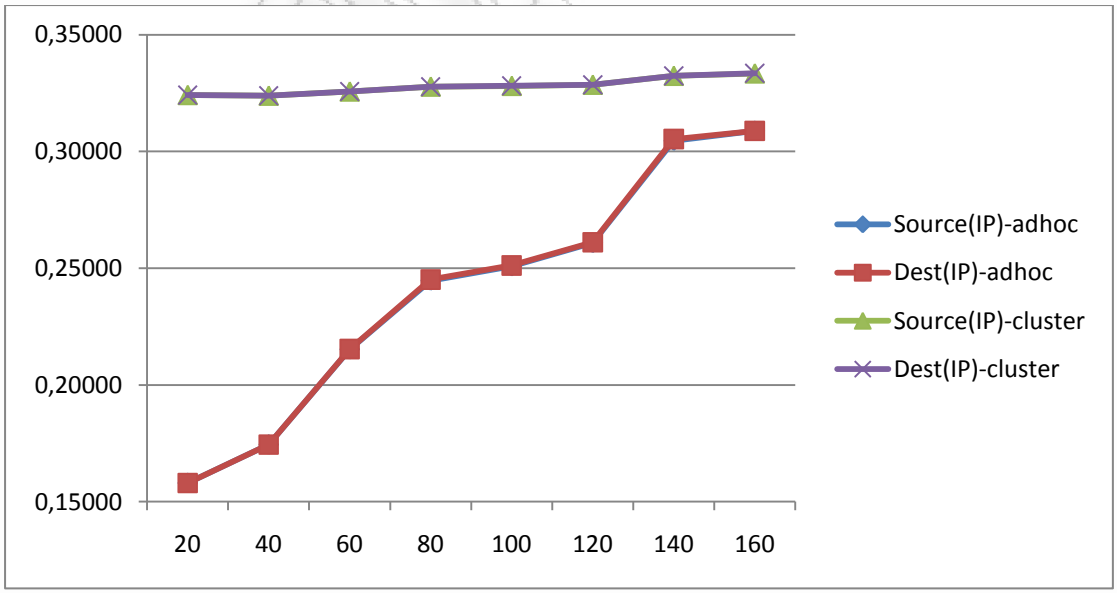


Εικόνα 29 Λόγος απόδοσης του αλγόριθμου σε εξάπλωση worm και διαφορετική δρομολόγηση

Παρατηρούμε ότι στην περίπτωση εξάπλωσης worm, ο αλγόριθμος αποδίδει χειρότερα στην περίπτωση της ιεραρχικής δρομολόγησης. Μάλιστα, στη περίπτωση της δρομολόγησης adhoc η απόδοση του αλγορίθμου είναι 22% καλύτερη περίπου. Η μείωση στην απόδοση λόγω της ιεραρχικής δρομολόγησης είναι σημαντικότερη στη περίπτωση της εξάπλωσης worm, αλλά δεν είναι αρκετή για να θέσει τον αλγόριθμο ανίχνευσης αναξιόπιστο.

Η ακρίβεια του αλγορίθμου ανίχνευσης επηρεάζεται γιατί οι δυο διαφορετικοί τρόποι δρομολόγησης επηρεάζουν την εντροπία στην κανονική κατάσταση. Πιο συγκεκριμένα, στην περίπτωση της δρομολόγησης ad hoc, η εντροπία συνεχώς αυξάνεται, καθώς νέοι κόμβοι αρχίζουν να επικοινωνούν μεταξύ τους κατά τη διάρκεια της προσομοίωσης και η τυχαιότητα των διευθύνσεων πηγής και προορισμού αυξάνεται. Στη περίπτωση όμως της δρομολόγησης μέσω ομάδων, αλλάζει πολλές φορές η διεύθυνση προορισμού και πηγής, με αποτέλεσμα να δυσχεραίνεται η ανίχνευση. Αυτό φαίνεται και από το γεγονός ότι, αν και στην περίπτωση της ad hoc δρομολόγησης, συνεχώς νέοι κόμβοι αρχίζουν να επικοινωνούν και η εντροπία των source και destination I.P. θα έπρεπε να αυξάνεται, κάτι τέτοιο δε συμβαίνει.

Τα παραπάνω συνοψίζονται στο επόμενο διάγραμμα που δείχνει τις εντροπίες των κατανομών για source και destination I.P., με τις δυο διαφορετικές δρομολογήσεις.



Εικόνα 30 Σύγκριση στην εντροπία των κατανομών source και destination I.P. με διαφορετική δρομολόγηση

Βιβλιογραφικές αναφορές

- amitabh mishra, ketan nadkarni, and animesh patcha, virginia tech, “intrusion detection in wireless ad hoc networks”, topics in wireless security, 48-60, ieee wireless communications, february 2004
- tiranuch anantvalee, jie wu, “a survey on intrusion detection in mobile ad hoc networks” wireless/mobile network security, department of computer science and engineering florida atlantic university, boca raton, fl 33428, pp. 170 – 196, 2006 springer
- paul brutch, calvin ko, network associates laboratories, “challenges in intrusion detection for wireless ad-hoc networks”
- jane y. yu and peter h. j. chong, nanyang technological university, “a survey of clustering schemes for mobile ad hoc networks”, first quarter 2005, volume 7, no.1, ieee communications surveys & tutorials, first quarter 2005
- ameer ahmed abbasi, mohamed younis, “a survey on clustering algorithms for wireless sensor networks”,computer communications 30, 2826–2841 , 21 june 2007
- bo sun and lawrence osborne, yang xiao, sghaier guizani, “intrusion detection techniques in mobile ad hoc and wireless sensor networks”, security in wireless mobile ad hoc and sensor networks, 1536-1284, ieee wireless communications, october 2007
- georgios kambourakis, costas lambrinoudakis, socratis katsikas, “intrusion detection systems for wireless networks: a survey”
- s. axelsson, “intrusion detection systems: a taxomomy and survey,” tech. report no. 99-15, dept. of comp. eng., chalmers univ. of technology, sweden, mar. 20, 2003.
- Biswanath Mukherjee, Todd L. Heberlein, and Karl N. Levitt, “Network Intrusion Detection”, IEEE Network, May/June 1994
- Libpcap library – http://www.tcpdump.org
- http://nslab.ee.ntu.edu.tw/courses/ns-tutorial/ntu-tutorial.html
- http://www.isi.edu/nsnam/ns/tutorial/
- NS-2: http://www.isi.edu/nsnam/ns/

- Weaver, N., Paxson, V., Staniford, S., and Cunningham, R. 2003. A taxonomy of computer worms. In Proceedings of the 2003 ACM Workshop on Rapid Malcode (Washington, DC, USA, October 27 - 27, 2003).
- CERT. CERT Advisory CA-2001-22 w32/Sircam Malicious Code, http://www.cert.org/advisories/ca-2001-22.html.
- eEye Digital Security. .ida "Code Red" Worm, http://www.eeye.com/html/research/advisories/al20010717.html.
- David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford, and Nicholas Weaver. Inside the slammer worm. IEEE Magazine of Security and Privacy, pages 33–39, July/August 2003 2003
- Markus Kern. Re: Codegreen beta release, http://online.securityfocus.com/archive/82/211462
- Jason V Miller, Jesse Gough, Bartek Kostanecki, Josh Talbot, and Jensenne Roculan. Microsoft dcom rpc worm alert, https://tms.symantec.com/members/analystreports/030811-alert-dcomworm.pdf.
- Wagner, A. and Plattner, B. 2005. Entropy Based Worm and Anomaly Detection in Fast IP Networks. In Proceedings of the 14th IEEE international Workshops on Enabling Technologies: infrastructure For Collaborative Enterprise (June 13 - 15, 2005). WETICE. IEEE Computer Society, Washington
- I. Ari et al., "Managing Flash Crowds on the Internet," Proc. MASCOTS '03, Orlando, FL, Oct. 2003, pp. 246–49.
- Jelena Mirkovic, Janice Martin and Peter Reiher, "A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms", Computer Science Department - University of California, Los Angeles, Technical report #020018
- Jelena Mirkovic, Gregory Prier, Peter Reiher, "Attacking DDoS at the Source,University of California Los Angeles, Computer Science Department, Los Angeles, USA, Proceedings of the 10 th IEEE International Conference on Network Protocols (ICNP'02) © 2002 IEEE

- Charalampos Patrikakis, Michalis Masikos, and Olga Zouraraki, “Distributed Denial of Service Attacks”, The Internet Protocol Journal, Volume 7, Number 4, 2004
- S. Ranjan et al., “DoWitcher: Effective Worm Detection and Containment in the Internet Core,” IEEE INFOCOM ’07, Anchorage, AK, May 2007, pp. 2541–45.
- P. Barford and D. Plonka, “Characteristics of Network Traffic Flow Anomalies,”Proc. 1st ACM SIGCOMM Internet Measurement Wksp., San Francisco, CA, Nov. 2001, pp. 69–74.
- Androulidakis, G.; Chatzigiannakis, V.; Papavassiliou, S.; , "Network anomaly detection and classification via opportunistic sampling," Network, IEEE , vol.23, no.1, pp.6-12, January-February 2009
- A. Lakhina, M. Crovella, C. Diot, "Mining Anomalies Using Traffic Feature Distributions",in Proc. of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), pp. 217 – 228 , 2005
- daniel j. burroughs, linda f. wilson, george v. cybenko, “analysis of distributed intrusion detection systems using bayesian methods”, thayer school of engineering, dartmouth college hanover, nh 03755
- fang-yie leu, wei-jie yang, fang-yie leu, wei-jie yang, department of computer science and information engineering, tunghai university, taiwan t. enokido et al. (eds.): euc workshops 2005, lncs 3823, pp. 1255 – 1264, ifip international federation for information processing 2005
- anthony d. wood ,john a. stankovic,” denial of service in sensor networks”, university of virginia, ieee october 2002
- sundeep pattem, bhaskar krishnamachari, ramesh govindan, “the impact of spatial correlation on routing with compression in wireless sensor networks”, dept. of electrical engineering-systems university of southern california los angeles, ipsn’04, april 26–27, 2004
- p. basu, n. khan, and t.d.c. little,” a mobility based metric for clustering in mobile ad hoc networks”, department of electrical and computer engineering,

boston university,workshop on wireless networks and mobile computing, phoenix, az, april 2001.

- bruce mcdonald, taieb f. znati,," a mobility-based framework for adaptive clustering in wireless ad hoc networks", ieee journal on selected areas in communications, vol. 17, no. 8, august 1999
- xu li, zheng bao-yu, guo gong-de," mobility-aware and load balancing based clustering algorithm for energy conservation in m.a.nets, mar. 2005 journal of electronic science and technology of china vol.3 no.1
- jelena mirkovic, peter reiher, "a taxonomy of ddos attack and ddos defense mechanisms|» , 3564 boelter hall computer science department
  los angeles, ca 90095, 2002

Παράρτημα Α - Πίνακες αριθμητικών αποτελεσμάτων

1.1. Δρομολόγηση ad-hoc

1.1.1. Κανονική Κίνηση

time	source address	destination address	packet length	destination port
20	0,15800	0,15800	0,15807	0,76653
40	0,17445	0,17445	0,15149	0,78207
60	0,21523	0,21538	0,15160	0,78643
80	0,24468	0,24515	0,15143	0,78844
100	0,25083	0,25117	0,15163	0,77346
120	0,26082	0,26114	0,15175	0,76875
140	0,30458	0,30530	0,15168	0,75399
160	0,30876	0,30880	0,15185	0,77372
180	0,31294	0,31293	0,15181	0,78000

1.1.2. Επίθεση D.D.o.S.

time	source address	destination address	packet length	destination port
20	0,158071547	0,158071547	0,158187508	0,766969889
40	0,151658063	0,151658063	0,151516568	0,780687246
60	0,153210601	0,153708311	0,151545272	0,781699319
80	0,228320487	0,137001757	0,147418560	0,808481469
100	0,19590720	0,196300969	0,152149041	0,792578559
120	0,197456757	0,197789283	0,151447592	0,818049619
140	0,262888483	0,263512763	0,144920482	0,754184316
160	0,254406039	0,254451027	0,145224633	0,788843024

1.1.3. Επίθεση Worm propagation

time	source address	destination address	packet length	destination port
20	0,158071547	0,158071547	0,158187508	0,766969889
40	0,151658063	0,151658063	0,151516568	0,780687246
60	0,153210601	0,153708311	0,151545272	0,781699319
80	0,162344071	0,230227349	0,150147391	0,806140368
100	0,195219568	0,195522209	0,151477287	0,809380407
120	0,20939810	0,209839008	0,151865118	0,833263573
140	0,246305583	0,246788697	0,146786389	0,788589554
160	0,240513773	0,240789556	0,151699831	0,830506151

1.1.4. Επίθεση Flash crowd

time	source address	destination address	packet length	destination port
20	0,157992885	0,157992885	0,158209262	0,768263222
40	0,151641698	0,151641698	0,151532016	0,783033799
60	0,155578713	0,156449005	0,151468294	0,784689255
80	0,225451058	0,225126423	0,177008798	0,747031426
100	0,199871541	0,199908991	0,152784312	0,786844930
120	0,212400945	0,212679920	0,151746979	0,821420556
140	0,267655872	0,268206249	0,148084746	0,778559360
160	0,255840686	0,255795229	0,145228226	0,799360041

1.2. Αναλογία Source address entropy / Destination address entropy

1.2.1. Κανονική Κίνηση

ratio source address / destination address	source address	destination address
1	0,15800	0,15800
1	0,17445	0,17445
0,999324187	0,21523	0,21538
0,998097930	0,24468	0,24515
0,998622274	0,25083	0,25117
0,998793320	0,26082	0,26114
0,997630137	0,30458	0,30530
0,999861435	0,30876	0,30880
1,000028760	0,31294	0,312931

1.2.2. Επίθεση D.D.o.S.

ratio source address / destination address	source address	destination address
1	0,158071547	0,158071547
1	0,151658063	0,151658063
0,996761982	0,153210601	0,153708311
1,666551526	0,228320487	0,137001757
0,997994055	0,19590720	0,196300969
0,998318784	0,197456757	0,197789283
0,997630930	0,262888483	0,263512763
0,999823193	0,254406039	0,254451027

1.2.3. Επίθεση Worm propagation

ratio source address / destination address	**source address**	**destination address**
1	0,158071547	0,158071547
1	0,151658063	0,151658063
0,996761982	0,153210601	0,153708311
0,705146767	0,162344071	0,230227349
0,998452141	0,195219568	0,195522209
0,997898827	0,20939810	0,209839008
0,998042397	0,246305583	0,246788697
0,998854672	0,240513773	0,240789556

1.2.4. Επίθεση Flash crowd

ratio source address / destination address	**source address**	**destination address**
1	0,157992885	0,157992885
1	0,151641698	0,151641698
0,994437217	0,155578713	0,156449005
1,001442011	0,225451058	0,225126423
0,999812665	0,199871541	0,199908991
0,998688285	0,212400945	0,212679920
0,997947933	0,267655872	0,268206249
1,000177708	0,255840686	0,255795229

2.1. Ιεραρχική Δρομολόγηση

2.1.1. Κανονική Κίνηση

time	source address	destination address	packet length	destination port
20	0,3241667	0,324167819	0,034639153	0,382748568
40	0,3238868	0,323889589	0,048345569	0,416960325
60	0,325665	0,325663851	0,049299444	0,422930131
80	0,3277591	0,327760316	0,051538507	0,427701331
100	0,3280652	0,328092969	0,051054858	0,427242593
120	0,3285719	0,328580426	0,053010584	0,434787322
140	0,3323837	0,332404717	0,054717439	0,435368163
160	0,3333971	0,333475215	0,05552881	0,436515514
180	0,3328989	0,332928137	0,056297525	0,440584865

2.1.2. Επίθεση D.D.o.S.

time	source address	destination address	packet length	destination port
20	0,157997553	0,157997553	0,158069930	0,766532618
40	0,174451412	0,174451412	0,151489030	0,782068777
60	0,215233496	0,215379052	0,151604717	0,786428155
80	0,219321510	0,147368170	0,135819009	0,793966816
100	0,253185930	0,253349904	0,151596320	0,793234637
120	0,262129571	0,262710657	0,151830045	0,809772592
140	0,308564711	0,308873337	0,151645106	0,766233047
160	0,308874841	0,309062581	0,151732442	0,776603064
180	0,310545198	0,31073476	0,151781885	0,765858643

2.1.3. Επίθεση Worm propagation

time	source address	destination address	packet length	destination port
20	0,173314361	0,173314361	0,086744949	0,830746634
40	0,189473984	0,189473984	0,0827113	0,845366833
60	0,227522741	0,227534209	0,082749807	0,872476275
80	0,262079336	0,291012722	0,096122098	0,86568556
100	0,275898806	0,275601054	0,082754559	0,881244228
120	0,284835775	0,285406249	0,082797871	0,874080346
140	0,335491926	0,336121195	0,083353943	0,838441352
160	0,337302502	0,336781232	0,082847822	0,85301806
180	0,338103452	0,338479114	0,082804689	0,860072893

2.1.4. Επίθεση Flash crowd

time	source address	destination address	packet length	destination port
20	0,167189889	0,167189889	0,086814302	0,854603381
40	0,191270764	0,191270764	0,082690598	0,880083393
60	0,232710487	0,232615885	0,082726796	0,903101662
80	0,293787241	0,293481031	0,111535862	0,836631913
100	0,277108523	0,277350487	0,083834987	0,885829755
120	0,286884514	0,287458491	0,082744553	0,880423763
140	0,329177	0,328688618	0,082791166	0,838004662
160	0,338782677	0,33836501	0,084172829	0,846072768
180	0,337649156	0,338016181	0,082817022	0,857522535

2.2. Αναλογία Source address entropy / Destination address entropy

2.2.1. Κανονική Κίνηση

ratio source address / destination address	source address	destination address
1	0,3241667	0,324167819
1	0,3238868	0,323889589
0.993607169	0,325665	0,325663851
0.999996289	0,3277591	0,327760316
0.999915362	0,3280652	0,328092969
1.000205902	0,3285719	0,328580426
0.999936772	0,3323837	0,332404717
0.999765754	0,3333971	0,333475215
0.999912182	0,3328989	0,332928137

2.2.2. Επίθεση D.D.o.S

ratio source address / destination address	source address	destination address
1	0,157997553	0,157997553
1	0,174451412	0,174451412
0,999324187	0,215233496	0,215379052
1,488255642	0,219321510	0,147368170
0,999352778	0,253185930	0,253349904
0,997788112	0,262129571	0,262710657
0,9990008	0,308564711	0,308873337
0,999392553	0,308874841	0,309062581
0,999389956	0,310545198	0,31073476

2.2.3. Επίθεση Worm propagation

ratio source address / destination address	**source address**	**destination address**
1	0,173314361	0,173314361
1	0,189473984	0,189473984
0,999949597	0,227522741	0,227534209
0,900576905	0,262079336	0,291012722
1,001080371	0,275898806	0,275601054
0,998001186	0,284835775	0,285406249
0,99812785	0,335491926	0,336121195
1,0015478	0,337302502	0,336781232
0,998890149	0,338103452	0,338479114

2.2.4. Επίθεση Flash crowd

ratio source address / destination address	**source address**	**destination address**
1	0,167189889	0,167189889
1	0,191270764	0,191270764
1,00040669	0,232710487	0,232615885
1,001043374	0,293787241	0,293481031
0,999127588	0,277108523	0,277350487
0,998003271	0,286884514	0,287458491
1,001485852	0,329177	0,328688618
1,001234371	0,338782677	0,33836501
0,998914177	0,337649156	0,338016181

3.1. Λόγος Απόδοσης Source address entropy / Destination address entropy της ad-hoc δρομολόγησης με την ιεραρχική δρομολόγηση

3.1.1. Επίθεση D.D.o.S.

time	(ratio S/D) / (ratio S/D)
20	1
40	1
60	0,997436062
80	1,119801921
100	0,998640397
120	1,000531849
140	0,998628761
160	1,000430902
180	0

3.1.2. Επίθεση Worm propagation

time	(ratio S/D) / (ratio S/D)
20	1
40	1
60	0,996812224
80	0,782994504
100	0,997374606
120	0,999897436
140	0,999914387
160	0,997311034
180	0

3.1.3. Επίθεση Flash crowd

time	**(ratio S/D) / (ratio S/D)**
20	1
40	1
60	0.994032953
80	1.003982215
100	1.000685675
120	1.000683845
140	0.996473300
160	0.998944639
180	0

Παράρτημα Β - Κώδικας

Πρόγραμμα: anomalyDetection.pl

```
#!/usr/bin/perl
#DEFINE RULES
# rule 1: ratio of source/dest ips should be ~=1
$normal_IP_Ratio_MAX = 1.05;
$normal_IP_Ratio_MIN = 0.95;

#rule 2:
#END DEFINE RULES
$numArgs = $#ARGV + 1;
if($numArgs!=1){
        print "Usage: ./anomalyDetection inputFile\n";
        exit(1);
}
$infile = $ARGV[0];
open IN, $infile or die $!;

$time_bin = $step;
$count=0;

print "time\tAttack type\t\tInfo\n";
open IN, $infile or die $!;
while (<IN>) {
        if(($_ =~/^(\d+)/)){
                @metrics = split(/\t/, $_);
```

```
                # rule 1: ratio of source/dest ips should be ~=1
                $IP_ratio = $metrics[1]/$metrics[2];
                if($IP_ratio>$normal_IP_Ratio_MAX) {
                        print "$metrics[0]\tDDOS ALERT:\tSource IP entropy:
$metrics[1]\t\tDest IP entropy: $metrics[2]\t\tRatio:$IP_ratio\n";
                }

                # rule 2: ratio of source/dest ips should be ~=1
                if($IP_ratio<$normal_IP_Ratio_MIN) {
                        print "$metrics[0]:\tWORM ALERT:\tSource IP entropy:
$metrics[1]\t\tDest IP entropy: $metrics[2]\t\tRatio:$IP_ratio\n";
                }


        }
}
```

Πρόγραμμα: getEntropies.pl

```
#!/usr/bin/perl
#log2() function
 sub log2 {
        my $n = shift;
        return log($n)/log(2);
}

#trim function, removes whitespaces and \n
sub trim($){
        my $string = shift;
        $string =~ s/^\s+//;
        $string =~ s/\s+$//;
        $string =~ s/\n$//;
        return $string;
}

#time step
$step =20;

$numArgs = $#ARGV + 1;
if($numArgs!=1){
        print "Usage: ./getEntropies tracefile\n";
        exit(1);
}
```

```
$infile = $ARGV[0];
open IN, $infile or die $!;

$time_bin = $step;
$count=0;

print "time\tSource(IP)\t\tDest(IP)\t\tPacket Length\t\tDest(ports)\n";
open IN, $infile or die $!;
while (<IN>) {
        @data = split(/\t/,$_);
        $count++;
        if($count==1){
        next;
        }

        if($data[0]<$time_bin){
                #get hashes of metrics
                $sources{$data[2]}++;
                $dests{$data[3]}++;
                $lengths{$data[4]}++;
                $port = trim($data[5]);
                $ports{$port}++;
        }
        else{ #end of time bin, calculate and print entropies for each metric
                print $time_bin . "\t";
                $count = $count-1;

                $sum=0;
                while (($key, $value) = each(%sources)){
                #entropy sum
                #$value/$count gives the frequency (possibility) of a specific key
                        $sum = $sum -  $value/$count * log2($value/$count);
                }
```

```
                #normalization of entropy
                print $sum/log2($count) . "\t";

                $sum=0;
                while (($key, $value) = each(%dests)){
                        $sum = $sum -  $value/$count * log2($value/$count);
                }
                print $sum/log2($count) . "\t";

                $sum=0;
                while (($key, $value) = each(%lengths)){
                        $sum = $sum -  $value/$count * log2($value/$count);
                }
                print $sum/log2($count) . "\t";

                $sum=0;
                foreach $key (sort keys %ports) {
                        $sum = $sum -  $ports{$key}/$count *
log2($ports{$key}/$count);
                }
                print $sum/log2($count) . "\n";

                $count=1;
                #next time_bin
                $time_bin = $time_bin +$step;

                #reset hashes for next time bin
                undef %sources;
                undef %dests;
                undef %lengths;
                undef %ports;
        }
}
close IN;
```

Πρόγραμμα: getResults.pl

```
#!/usr/bin/perl

$numArgs = $#ARGV + 1;
if($numArgs!=2){
        print "Usage: ./getResults tracefile output\n";
        exit(1);
}
$infile = $ARGV[0];
$outfile = $ARGV[1];
open IN, $infile or die $!;
open OUT, ">$outfile" or die $!;

print OUT "time\t\t proto\t source\t dest\t length\t port\n";
while (<IN>) {
        if(($_ =~/^r /) && ($_ !~/message/ )){

                $_ =~/^r -t (\d+\.\d*)/;
                $time = $1;
                #print "time:$time\n";

                $_ =~/-Il (\d+)/;
                $length = $1;
                #print "length:$length\n";

                $_ =~/-It (\w+)/;
                $proto = $1;
                #print "proto:$proto\n";

                $_ =~/-Is (\d+)\./;
                $source = $1;
                #print "source:$source\n";
```

```
                $_ =~/-Id (\d+)\./;
                $dest = $1;
                #print "dest:$dest\n";

                if($proto eq 'cbr'){
                        $_ =~/-Pi (\d+)/;
                        $port = $1;
                        #print "port:$port\n";
                }
                else{
                        $_ =~/-Ps (\d+)/;
                        $port = $1;
                        #print "port:$port\n";
                }

                if($proto eq 'tcp'){
                        $type =1;
                }
                elsif($proto eq 'ack'){
                        $type =2;
                }
                else{
                        $type=3;
                }

                print OUT "$time\t $type\t $source\t $dest\t $length\t $port\n";
        }
}
close IN;
close OUT;
```

Cluster routing script: cluster.tcl

```
# ======================================================================
# Define options
# ======================================================================

set opt(chan)   Channel/WirelessChannel          ;# channel type
set opt(prop)   Propagation/TwoRayGround     ;# radio-propagation model
set opt(netif)  Phy/WirelessPhy                  ;# network interface type
set opt(mac)    Mac/802_11                       ;# MAC type
set opt(ifq)    Queue/DropTail/PriQueue          ;# interface queue type
set opt(ll)     LL                               ;# link layer type
set opt(ant)    Antenna/OmniAntenna              ;# antenna model
set opt(ifqlen)         50                       ;# max packet in ifq
set opt(nn)             30                       ;# number of mobilenodes
set opt(adhocRouting)   D.S.D.V.                 ;# routing protocol

set opt(x)      1000                             ;# x coordinate of topology
set opt(y)      1000                             ;# y coordinate of topology
set opt(seed)   123.0                            ;# random seed
set opt(stop)   200                              ;# time to stop simulation



# ======================================================================


# create simulator instance
set ns_   [new Simulator]

#ns seed
ns-random $opt(seed)

# set up for hierarchical routing
$ns_ node-config -addressType hierarchical
```

```
AddrParams set domain_num_ 4                ;# number of domains
lappend cluster_num 2 1 1 1                 ;# number of clusters in each domain
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 1 11 11 11            ;# number of nodes in each cluster
AddrParams set nodes_num_ $eilastlevel      ;# of each domain

# Set up trace file
$ns_ use-newtrace
set tracefd [open results/trace.tr w]
$ns_ trace-all $tracefd



# Create topography object
set topo   [new Topography]

# define topology
$topo load_flatgrid opt(x)  opt(y)

# create God
#   3 for C1,C2 and C3
create-god [expr $opt(nn) + 3]



# Configure for ForeignAgent and HomeAgent nodes
$ns_ node-config -mobileIP ON \
                 -adhocRouting $opt(adhocRouting) \
                 -llType $opt(ll) \
                 -macType $opt(mac) \
                 -ifqType $opt(ifq) \
                 -ifqLen $opt(ifqlen) \
                 -antType $opt(ant) \
                 -propType $opt(prop) \
                 -phyType $opt(netif) \
```

```
        -channelType $opt(chan) \
            -topoInstance $topo \
        -wiredRouting ON \
            -agentTrace ON \
        -routerTrace OFF \
        -macTrace OFF

# Create C1 and C2
set C1 [$ns_ node 1.0.0]
set C2 [$ns_ node 2.0.0]
set C3 [$ns_ node 3.0.0]
$C1 random-motion 0
$C2 random-motion 0
$C3 random-motion 0

# Position (fixed) for base-station nodes.
$C1 set X_ 200.000000000000
$C1 set Y_ 200.000000000000
$C1 set Z_ 0.000000000000

$C2 set X_ 800.000000000000
$C2 set Y_ 200.000000000000
$C2 set Z_ 0.000000000000

$C3 set X_ 500.000000000000
$C3 set Y_ 500.000000000000
$C3 set Z_ 0.000000000000


$ns_ node-config -wiredRouting OFF
```

```
# mobile nodes address list
set temp1 {1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10 2.0.1 2.0.2 2.0.3
2.0.4 2.0.5 2.0.6 2.0.7 2.0.8 2.0.9 2.0.10 3.0.1 3.0.2 3.0.3 3.0.4 3.0.5 3.0.6 3.0.7 3.0.8
3.0.9 3.0.10 }  ;

#create nodes in same cluster as C1
for {set i 0} {$i < 10 } {incr i} {
        set node_($i) [$ns_ node [lindex $temp1 $i]]
        set Nodeaddress [AddrParams addr2id [$C1 node-addr]]
        [$node_($i) set regagent_] set home_agent_ $Nodeaddress

        $node_($i) set X_ [expr $i + 200.0000000]
        $node_($i) set Y_ [expr $i + 200.0000000]
        $node_($i) set Z_ 0.0

        $node_($i) random-motion 1
        $node_($i) start
}

#create nodes in same cluster as C2
for {set i 10} {$i < 20 } {incr i} {
        set node_($i) [$ns_ node [lindex $temp1 $i]]
        $node_($i) random-motion 1
        set Nodeaddress [AddrParams addr2id [$C2 node-addr]]
        [$node_($i) set regagent_] set home_agent_ $Nodeaddress

        $node_($i) set X_ [expr $i + 800.0000000]
        $node_($i) set Y_ [expr $i + 200.0000000]
        $node_($i) set Z_ 0.0

        $node_($i) random-motion 1
        $node_($i) start
}
```

```
#create nodes in same cluster as C3
for {set i 20} {$i < 30 } {incr i} {
        set node_($i) [$ns_ node [lindex $temp1 $i]]
        set Nodeaddress [AddrParams addr2id [$C3 node-addr]]
        [$node_($i) set regagent_] set home_agent_ $Nodeaddress

        $node_($i) set X_ [expr $i + 500.0000000]
        $node_($i) set Y_ [expr $i + 500.0000000]
        $node_($i) set Z_ 0.0

        $node_($i) random-motion 1
        $node_($i) start
}

#load motion file
#source motion.tcl

#create links between BaseStation nodes
$ns_ duplex-link $C1 $C2 50Mb 2ms DropTail
$ns_ duplex-link $C1 $C3 50Mb 2ms DropTail
$ns_ duplex-link $C2 $C3 50Mb 2ms DropTail


# load background traffic
source tcp-scenario.tcl

#load anomaly


# Tell all nodes when the siulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
   $ns_ at $opt(stop).0 "$node_($i) reset";
}
```

```
$ns_ at $opt(stop).0 "$C1 reset";
$ns_ at $opt(stop).0 "$C2 reset";

$ns_ at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"
$ns_ at $opt(stop).0001 "stop"

proc stop {} {
   global ns_ tracefd
   close $tracefd
}

# some useful headers for tracefile
puts $tracefd "hierarhical routing scenario"
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp \
          $opt(adhocRouting)   seed $opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```

Adhoc routing script: ad-hoc.tcl

```
# ======================================================
# Define options
# ======================================================

set opt(chan)   Channel/WirelessChannel        ;# channel type
set opt(prop)   Propagation/TwoRayGround    ;# radio-propagation model
set opt(netif)  Phy/WirelessPhy                ;# network interface type
set opt(mac)    Mac/802_11                     ;# MAC type
set opt(ifq)    Queue/DropTail/PriQueue        ;# interface queue type
set opt(ll)     LL                             ;# link layer type
set opt(ant)    Antenna/OmniAntenna            ;# antenna model
set opt(ifqlen)         50                     ;# max packet in ifq
set opt(nn)             30                     ;# number of mobilenodes
set opt(adhocRouting)   D.S.D.V.              ;# routing protocol

set opt(x)      1000                        ;# x coordinate of topology
set opt(y)      1000                        ;# y coordinate of topology
set opt(seed)   123.0                       ;# random seed
set opt(stop)   200                         ;# time to stop simulation

# ======================================================

# create simulator instance
set ns_   [new Simulator]

#ns seed
ns-random $opt(seed)

# Set up trace file
$ns_ use-newtrace
set tracefd [open results/trace.tr w]
$ns_ trace-all $tracefd
```

```
# Create topography object
set topo   [new Topography]

# define topology
$topo load_flatgrid opt(x)  opt(y)

# create God

#   3 for C1,C2 and C3
create-god [expr $opt(nn) ]

# Configure for mobile nodes
$ns_ node-config -adhocRouting $opt(adhocRouting) \
                 -llType $opt(ll) \
                 -macType $opt(mac) \
                 -ifqType $opt(ifq) \
                 -ifqLen $opt(ifqlen) \
                 -antType $opt(ant) \
                 -propType $opt(prop) \
                 -phyType $opt(netif) \
                 -channelType $opt(chan) \
                                  -topoInstance $topo \
                                 -agentTrace ON \
                                 -routerTrace ON \
                                 -macTrace OFF \
                                 -movementTrace OFF
```

```
#create nodes in first neighborhood
for {set i 0} {$i < 10 } {incr i} {
        set node_($i) [$ns_ node ]

        $node_($i) set X_ [expr $i + 200.0000000]
        $node_($i) set Y_ [expr $i + 200.0000000]
        $node_($i) set Z_ 0.0

        $node_($i) random-motion 1
        $node_($i) start
}
#create nodes in second neighborhood
for {set i 10} {$i < 20 } {incr i} {
        set node_($i) [$ns_ node ]

        $node_($i) set X_ [expr $i + 600.0000000]
        $node_($i) set Y_ [expr $i + 200.0000000]
        $node_($i) set Z_ 0.0

        $node_($i) random-motion 1
        $node_($i) start
}
#create nodes in third neighborhood
for {set i 20} {$i < 30 } {incr i} {
        set node_($i) [$ns_ node]

        $node_($i) set X_ [expr $i + 400.0000000]
        $node_($i) set Y_ [expr $i + 400.0000000]
        $node_($i) set Z_ 0.0

        $node_($i) random-motion 1
        $node_($i) start
}
```

```
# load background traffic
source tcp-scenario.tcl
source udp-scenario.tcl

#load anomaly
source anomaly.tcl

# Tell all nodes when the siulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).0 "$node_($i) reset";
}

$ns_ at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"
$ns_ at $opt(stop).0001 "stop"

proc stop {} {
    global ns_ tracefd
    close $tracefd
}

# some useful headers for tracefile
puts $tracefd "adhoc routing scenario"
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp \
          $opt(adhocRouting)   seed $opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```

Παράδειγμα από script για DDoS επίθεση

```
#
#
# 1 connecting to 10 at time 62.5
#
set ddos_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $ddos_(0)
set ddos_null_(0) [new Agent/Null]
$ns_ attach-agent $node_(10) $ddos_null_(0)
set ddos_cbr_(0) [new Application/Traffic/CBR]
$ddos_cbr_(0) set packetSize_ 100
$ddos_cbr_(0) set interval_ 0.001
$ddos_cbr_(0) set random_ 1
$ddos_cbr_(0) set maxpkts_ 3000
$ddos_cbr_(0) attach-agent $ddos_(0)
$ns_ connect $ddos_(0) $ddos_null_(0)
$ns_ at 62.5 "$ddos_cbr_(0) start"
#
#
# 11 connecting to 10 at time 62.5
#
set ddos_(1) [new Agent/UDP]
$ns_ attach-agent $node_(11) $ddos_(1)
set ddos_null_(1) [new Agent/Null]
$ns_ attach-agent $node_(10) $ddos_null_(1)
set ddos_cbr_(1) [new Application/Traffic/CBR]
$ddos_cbr_(1) set packetSize_ 100
$ddos_cbr_(1) set interval_ 0.001
$ddos_cbr_(1) set random_ 1
$ddos_cbr_(1) set maxpkts_ 3000
$ddos_cbr_(1) attach-agent $ddos_(1)
$ns_ connect $ddos_(1) $ddos_null_(1)
$ns_ at 62.5 "$ddos_cbr_(1) start"
```

```
#
#
# 21 connecting to 10 at time 62.5
#
set ddos_(2) [new Agent/UDP]
$ns_ attach-agent $node_(21) $ddos_(2)
set ddos_null_(2) [new Agent/Null]
$ns_ attach-agent $node_(10) $ddos_null_(2)
set ddos_cbr_(2) [new Application/Traffic/CBR]
$ddos_cbr_(2) set packetSize_ 100
$ddos_cbr_(2) set interval_ 0.001
$ddos_cbr_(2) set random_ 1
$ddos_cbr_(2) set maxpkts_ 3000
$ddos_cbr_(2) attach-agent $ddos_(2)
$ns_ connect $ddos_(2) $ddos_null_(2)
$ns_ at 62.5 "$ddos_cbr_(2) start"
#
#
# 4 connecting to 10 at time 62.5
#
set ddos_(3) [new Agent/UDP]
$ns_ attach-agent $node_(4) $ddos_(3)
set ddos_null_(3) [new Agent/Null]
$ns_ attach-agent $node_(10) $ddos_null_(3)
set ddos_cbr_(3) [new Application/Traffic/CBR]
$ddos_cbr_(3) set packetSize_ 100
$ddos_cbr_(3) set interval_ 0.001
$ddos_cbr_(3) set random_ 1
$ddos_cbr_(3) set maxpkts_ 3000
$ddos_cbr_(3) attach-agent $ddos_(3)
$ns_ connect $ddos_(3) $ddos_null_(3)
$ns_ at 62.5 "$ddos_cbr_(3) start"
```

Παράδειγμα από script για εξάπλωση flash

```
#
# 1 connecting to 5
#
set tcp_(16) [$ns_ create-connection  TCP $node_(1) TCPSink $node_(5) 0]
$tcp_(16) set window_ 32
$tcp_(16) set packetSize_ 1500
set ftp_(16) [$tcp_(16) attach-source FTP]
$ns_ at 61.8 "$ftp_(16) start"
$ns_ at 77.8 "$ftp_(16) stop"
#
# 2 connecting to 5
#
set tcp_(17) [$ns_ create-connection  TCP $node_(2) TCPSink $node_(5) 0]
$tcp_(17) set window_ 32
$tcp_(17) set packetSize_ 1500
set ftp_(17) [$tcp_(17) attach-source FTP]
$ns_ at 61.8 "$ftp_(17) start"
$ns_ at 77.8 "$ftp_(17) stop"
#
# 3 connecting to 5
#
set tcp_(18) [$ns_ create-connection  TCP $node_(3) TCPSink $node_(5) 0]
$tcp_(18) set window_ 32
$tcp_(18) set packetSize_ 1500
set ftp_(18) [$tcp_(18) attach-source FTP]
$ns_ at 61.8 "$ftp_(18) start"
$ns_ at 77.8 "$ftp_(18) stop"
```

```
#
# 4 connecting to 5
#
set tcp_(19) [$ns_ create-connection  TCP $node_(14) TCPSink $node_(5) 0]
$tcp_(19) set window_ 32
$tcp_(19) set packetSize_ 1500
set ftp_(19) [$tcp_(19) attach-source FTP]
$ns_ at 61.8 "$ftp_(19) start"
$ns_ at 70.8 "$ftp_(19) stop"
#
# 6 connecting to 5
#
set tcp_(20) [$ns_ create-connection  TCP $node_(16) TCPSink $node_(5) 0]
$tcp_(20) set window_ 32
$tcp_(20) set packetSize_ 1500
set ftp_(20) [$tcp_(20) attach-source FTP]
$ns_ at 61.8 "$ftp_(20) start"
$ns_ at 74.8 "$ftp_(20) stop"
#
# 7 connecting to 5
#
set tcp_(21) [$ns_ create-connection  TCP $node_(17) TCPSink $node_(5) 0]
$tcp_(21) set window_ 32
$tcp_(21) set packetSize_ 1500
set ftp_(21) [$tcp_(21) attach-source FTP]
$ns_ at 61.8 "$ftp_(21) start"
$ns_ at 77.8 "$ftp_(21) stop"
```

```
#
# 8 connecting to 5
#
set tcp_(22) [$ns_ create-connection  TCP $node_(28) TCPSink $node_(5) 0]
$tcp_(22) set window_ 32
$tcp_(22) set packetSize_ 1500
set ftp_(22) [$tcp_(22) attach-source FTP]
$ns_ at 68.8 "$ftp_(22) start"
$ns_ at 72.8 "$ftp_(22) stop"
#
# 9 connecting to 5
#
set tcp_(23) [$ns_ create-connection  TCP $node_(29) TCPSink $node_(5) 0]
$tcp_(23) set window_ 32
$tcp_(23) set packetSize_ 1500
set ftp_(23) [$tcp_(23) attach-source FTP]
$ns_ at 67.8 "$ftp_(23) start"
$ns_ at 72.8 "$ftp_(23) stop"
#
# 10 connecting to 5
#
#set tcp_(24) [$ns_ create-connection  TCP $node_(21) TCPSink $node_(5) 0]
#$tcp_(24) set window_ 32
#$tcp_(24) set packetSize_ 1500
#set ftp_(24) [$tcp_(24) attach-source FTP]
#$ns_ at 61.8 "$ftp_(24) start"
#$ns_ at 77.8 "$ftp_(24) stop"
```

Παράδειγμα από script για εξάπλωση worm

```
#
#
# 1 connecting to 2 at time 62.5
#
set worm_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $worm_(0)
set worm_null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $worm_null_(0)
set worm_cbr_(0) [new Application/Traffic/CBR]
$worm_cbr_(0) set packetSize_ 100
$worm_cbr_(0) set interval_ 0.001
$worm_cbr_(0) set random_ 1
$worm_cbr_(0) set maxpkts_ 2000
$worm_cbr_(0) attach-agent $worm_(0)
$ns_ connect $worm_(0) $worm_null_(0)
$ns_ at 62.5 "$worm_cbr_(0) start"
#
set worm_(1) [new Agent/UDP]
$ns_ attach-agent $node_(1) $worm_(1)
set worm_null_(1) [new Agent/Null]
$ns_ attach-agent $node_(3) $worm_null_(1)
set worm_cbr_(1) [new Application/Traffic/CBR]
$worm_cbr_(1) set packetSize_ 100
$worm_cbr_(1) set interval_ 0.001
$worm_cbr_(1) set random_ 1
$worm_cbr_(1) set maxpkts_ 2000
$worm_cbr_(1) attach-agent $worm_(1)
$ns_ connect $worm_(1) $worm_null_(1)
$ns_ at 62.5 "$worm_cbr_(1) start"
```

```
#
set worm_(2) [new Agent/UDP]
$ns_ attach-agent $node_(1) $worm_(2)
set worm_null_(2) [new Agent/Null]
$ns_ attach-agent $node_(4) $worm_null_(2)
set worm_cbr_(2) [new Application/Traffic/CBR]
$worm_cbr_(2) set packetSize_ 100
$worm_cbr_(2) set interval_ 0.001
$worm_cbr_(2) set random_ 1
$worm_cbr_(2) set maxpkts_ 2000
$worm_cbr_(2) attach-agent $worm_(2)
$ns_ connect $worm_(2) $worm_null_(2)
$ns_ at 62.5 "$worm_cbr_(2) start"
#
set worm_(3) [new Agent/UDP]
$ns_ attach-agent $node_(1) $worm_(3)
set worm_null_(3) [new Agent/Null]
$ns_ attach-agent $node_(5) $worm_null_(3)
set worm_cbr_(3) [new Application/Traffic/CBR]
$worm_cbr_(3) set packetSize_ 100
$worm_cbr_(3) set interval_ 0.001
$worm_cbr_(3) set random_ 1
$worm_cbr_(3) set maxpkts_ 2000
$worm_cbr_(3) attach-agent $worm_(3)
$ns_ connect $worm_(3) $worm_null_(3)
$ns_ at 62.5 "$worm_cbr_(3) start"
```