Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

`Πτυχιακή εργασία

Μελέτη, ανάλυση και επίδειξη Συστημάτων Ανίχνευσης Εισβολών

Ιωάννης Κοζομπόλης (ΑΜ: 2669)

E-mail: giannis1990@hotmail.com

Επόπτης Καθηγητής: Δρ. Μανιφάβας Χαράλαμπος

Επιτροπή Αξιολόγησης:

Ημερομηνία Παρουσίασης:

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου, που όλα τα χρόνια με βοήθησαν και με στήριξαν οικονομικά, ώστε να φτάσω στο σημείο να τελειώσω το πρόγραμμα σπουδών του τμήματος Εφαρμοσμένης Πληροφορικής και Πολυμέσων και να παραδώσω αυτή την πτυχιακή εργασία. Επίσης, θα ήθελα να ευχαριστήσω τον επόπτη καθηγητή κύριο Χαράλαμπο Μανιφάβα που με εμπιστεύτηκε και μου ανέθεσε αυτή την εργασία.

Abstract

Computer network security has been a promiment issue since the early of computing, and even more so today with the ubiquitous deployment of computer systems. The transmission of data through telecommunication networks introduces various issues, as it is often vulnerable to malicious attacks. Also, the connection of those networks with Internet enables attacks by external users.

One solution to this problem is the use of Intrusion Detection Systems (IDS). In this thesis, we study and analyse IDS. We eplore the different IDS models, that are utilized for detecting attacks. More specifically, we consider two main types of IDS: the Network IDS (NIDS) and Host-Based IDS (HIDS). For demonstration, we use Snort, which is an open source NIDS. Furthermore, we briefly describe the Intrusion Prevention Systems (IPS).

We thoroughly describe Snort. Firstly, we outline its operation and architecture. Then we record a full installation guide with all of the pre-required programs and enumarate the set of rules. Moreover, we describe the installation and operation of BASE, which is a Web Interface for organizing and analyzing Alerts that are triggered by Snort.

The practical part of this thesis includes a detailed description of setup and operation of Snort IDS, as a simple home network. In the examined scenarios, known attacks were perfomed and detected with Snort. Finally, we present the DAQ (Data Acquisition) module and its integration with Snort for running as IPS. Thus, Snort repulses the known attacks.

Σύνοψη

Το πρόβλημα στα δίκτυα υπολογιστών έχει απασχολήσει, απασχολεί και θα απασχολεί όλους εκείνους που εμπλέκονται και διακυβεύονται τα συμφέροντα τους από την χρήση των δικτύων υπολογιστών. Η διακίνηση των δεδομένων μέσω τηλεπικοινωνιακών δικτύων δημιουργεί προβλήματα καθώς αυτά καθίστανται ευπρόσβλητα σε κακόβουλες ενέργειες. Επίσης, η σύνδεση αυτών των δικτύων με το εξωτερικό Internet δίνει την δυνατότητα επιθέσεων προς αυτά τα ιδιωτικά δίκτυα, από εξωτερικούς χρήστες.

Μία λύση για αυτό το πρόβλημα είναι η ασφάλιση των δικτύων και υποδικτύων με Συστήματα Ανίχνευσης Εισβολών (IDS). Στην παρούσα πτυχιακή εργασία γίνεται μελέτη, ανάλυση και επίδειξη Συστημάτων Ανίχνευση Εισβολών. Για την επίδειξη χρησιμοποιείτε το Snort, όπου είναι ένα Σύστημα ανίχνευσης Εισβολών βασισμένο στο δίκτυο (NIDS).

Ποιο συγκεκριμένα, γίνεται περιγραφή των Συστημάτων Ανίχνευσης Εισβολών και των δύο σημαντικών κατηγοριών NIDS και HIDS. Αναλύονται οι τεχνικές ανάλυσης των IDS, όπου ένα IDS χρησιμοποιεί για να ανιχνεύει τις πιθανές επιθέσεις. Μικρή αναφορά γίνεται για τα Συστήματα Αποτροπής Επιθέσεων (IPS).

Στην συνέχεια γίνεται λεπτομερής περιγραφή του Snort. Αρχικά, περιγράφονται οι λειτουργίες και η αρχιτεκτονική του Snort. Ακολουθεί πλήρης περιγραφή εγκατάστασης του με όλα τα προαπαιτούμενα προγράμματα, που είναι απαραίτητα. Επιπλέον, περιγράφεται η εγκατάσταση και λειτουργία του BASE (Basic Analysis and Security Engine), ενός Web Interface για οργάνωση και ανάλυση των Alerts, που πυροδοτούνται από το Snort. Επίσης, γίνεται ανάλυση και περιγραφή των κανόνων του Snort, που κατά μεγάλο ποσοστό οι περισσότερες ανιχνεύσεις επιθέσεων προέρχονται από αυτούς.

Το πρακτικό κομμάτι της εργασίας αυτής, περιλαμβάνει λεπτομερής περιγραφή ρύθμισης και λειτουργίας του Snort σαν IDS, σε απλά οικιακά δίκτυα. Περιγράφονται σενάρια όπου εκτελούνται γνωστές επιθέσεις, οι οποίες ανιχνεύονται με το Snort. Τέλος, γίνεται περιγραφή του DAQ (Data Acquisition) ενός module που παρέχει λειτουργίες, ώστε το Snort να τρέχει σαν IPS. Επίσης, περιγράφονται σενάρια, όπου εκτελούνται γνωστές επιθέσεις, οι οποίες αποκρούονται με το Snort.

Πίνακας περιεχομένων

Πίνακας εικόνων

Κεφάλαιο 1 Συστήματα Ανίχνευσης Εισβολών (IDS)

Βασικοί ορισμοί

Ασφάλεια πληροφοριακών συστημάτων

Ασφάλεια πληροφοριακών συστημάτων είναι ο κλάδος της πληροφορικής που ασχολείται με την προστασία των υπολογιστών, των δικτύων και των δεδομένων που μεταφέρονται μέσω αυτών. Αυτό το πετυχαίνουν αναπτύσσοντας μηχανισμούς ώστε να αποτρέπουν την μη εξουσιοδοτημένη πρόσβαση και χρήση αυτών από κακόβουλους χρήστες.

Η ασφάλεια πληροφοριακών συστημάτων βασίζεται σε 3 ιδέες:

- **Ακεραιότητα:** Αυτή η ιδέα διασφαλίζει ότι τα δεδομένα ενός πληροφοριακού συστήματος δε θα τροποποιηθούν ανεπιθύμητα από μη-εξουσιοδοτημένη πρόσβαση.
- **Διαθεσιμότητα**: Η διαθεσιμότητα μας εξασφαλίζει ότι τα δεδομένα, οι υπολογιστές και το δίκτυο θα είναι διαθέσιμα προς τους χρήστες όποτε τα χρειαστούν.
- **Εμπιστευτικότητα**: Η εμπιστευτικότητα ορίζει ότι οι πληροφορίες και τα δεδομένα που είναι άκρος εμπιστευτικά δε θα έπρεπε ποτέ να δημοσιευτούν σε μη-εξουσιοδοτημένους χρήστες.

Επιθέσεις

Στη γλώσσα των υπολογιστών και των ηλεκτρονικών δικτύων μια επίθεση είναι κάθε προσπάθεια για καταστροφή, αλλαγή, απενεργοποίηση, υποκλοπή ή απόκτηση μη εξουσιοδοτημένης πρόσβασης σε οποιοδήποτε περιουσιακό στοιχείο. Γενικά ότι αγνοεί τις βασικές αρχές Ακεραιότητας, Διαθεσιμότητας και Εμπιστευτικότητας μπορεί να χαρακτηριστεί ως επίθεση.

Μια επίθεση ανάλογα με το τι κάνει μπορεί να χαρακτηριστεί ως Παθητική ή Ενεργητική επίθεση.

- Παθητική επίθεση χαρακτηρίζεται η επίθεση που έχει ως στόχο την ανάκτηση χρήσιμων πληροφοριών χωρίς να επηρεάζει τους πόρους του συστήματος.
- Ενεργητική επίθεση χαρακτηρίζεται η επίθεση που τα αποτελέσματα της επηρεάζουν τους πόρους ενός συστήματος.

Κατηγοριοποίηση γνωστών επιθέσεων:

Παθητικές επιθέσεις:

- Δικτυακές
  - wiretapping
  - Port scanner
  - Idle scan

Ενεργητικές επιθέσεις:

- Denial-of-service attack
- Spoofing
- Δικτυακές
    - Man in the middle
    - ARP poisoning
    - Ping flood
    - Ping of death
    - Smurf attack
- Host
    - Buffer overflow
    - Heap overflow
    - Format string attack

Εισβολή

Ως εισβολή μπορούμε να χαρακτηρίσουμε οποιαδήποτε μη-εξουσιοδοτημένη είσοδο και κατά συνέπεια πρόσβαση σε απλούς προσωπικούς υπολογιστές, διακομιστές και δίκτυα υπολογιστών. Μια τέτοια ενέργεια μπορεί να είναι καθοριστική για την ασφάλεια των προσωπικών δεδομένων. Αυτή η ενέργεια απαγορεύεται και διώκεται ποινικά.

1.1 Τι είναι ένα Σύστημα Ανίχνευσης Εισβολών

Ανίχνευση εισβολών μπορεί να οριστεί ως «... την πράξη για την ανίχνευση ενεργειών που επιχειρούν να θέσουν σε κίνδυνο την εμπιστευτικότητα, την ακεραιότητα ή τη διαθεσιμότητα ενός πόρου.".

Συνεπώς, ένα **Σύστημα Ανίχνευσης Εισβολής (ΣΑΕ)** (**Intrusion Detection System**, **IDS**) αποτελεί σύστημα παρακολούθησης και ανάλυσης των συμβάντων, τα οποία λαμβάνουν χώρα τόσο στους ίδιους τους ηλεκτρονικούς υπολογιστές όσο και στα δίκτυα υπολογιστών. Στόχος είναι ο εντοπισμός ενδείξεων για πιθανές προσπάθειες εισβολής, κατά τις οποίες συχνά εντοπίζονται ίχνη παραβίασης της ακεραιότητας, της εμπιστευτικότητας και της διαθεσιμότητας των πληροφοριακών πόρων. Οι προσπάθειες παράκαμψης των μηχανισμών ασφαλείας μπορεί να προέρχονται από εξωτερικούς χρήστες, προς το εσωτερικό εταιρικό δίκτυο, στους οποίους δεν επιτρέπεται η πρόσβαση στο υπάρχον πληροφοριακό σύστημα. Επίσης, οι προσπάθειες παράκαμψης πιθανόν να προέρχονται από εσωτερικούς χρήστες, με περιορισμένα δικαιώματα πρόσβασης.

Οι λόγοι εγκατάστασης ενός συστήματος ανίχνευσης εισβολής ποικίλουν. Οι πιο σημαντικοί από αυτούς τους λόγους είναι η πρόληψη προβλημάτων, η ανίχνευση παραβιάσεων, η τεκμηρίωση υπαρκτών απειλών, ο έλεγχος ποιότητας για το σχεδιασμό ασφαλείας, καθώς και η θωράκιση παλαιών συστημάτων σε περίπτωση που κρίνεται αναγκαία η διατήρησή τους.

Η εξέλιξη των IDS, είναι ραγδαία τα τελευταία χρόνια. Τα IDS ακόμα και σήμερα βρίσκονται σε ερευνητικά στάδια και συνεχώς γίνονται προσπάθειες για βελτίωσή τους, κυρίως στον τομέα των συμπτωμάτων από **False Positives** και **False Negatives** που παρουσιάζουν.

> ***False Positives:*** *Είναι οι λανθασμένες επισημάνσεις που παράγει ένα IDS, όταν ανιχνεύσει κάποιο γεγονός σαν περίπτωση πιθανής επίθεσης ενώ δεν είναι. Τα False Positives είναι δυνατόν να προκύψουν από κακή ρύθμιση του IDS ή από περιπτώσεις γεγονότων που δεν μπορούν να διαχωριστούν σαφώς από μία επίθεση.*
>
> ***False Negatives:*** *Είναι οι περιπτώσεις επιθέσεων τις οποίες το IDS δεν κατάφερε μετά από την εξέτασή τους να τις επισημάνει. Τα False Negatives συνήθως προκύπτουν από κακή ρύθμιση του IDS ή από την εμφάνιση μίας νέας επίθεσης για την οποία δεν υπάρχει προηγούμενη γνώση.*

Τα IDSs ταξινομούνται από τη λειτουργικότητά τους και ομαδοποιούνται στις ακόλουθες κατηγορίες:

- Συστήματα Ανίχνευσης Εισβολών βασισμένα στο δίκτυο (**NIDS**)
- Συστήματα Ανίχνευσης Εισβολών βασισμένα στον Host (**HIDS**)

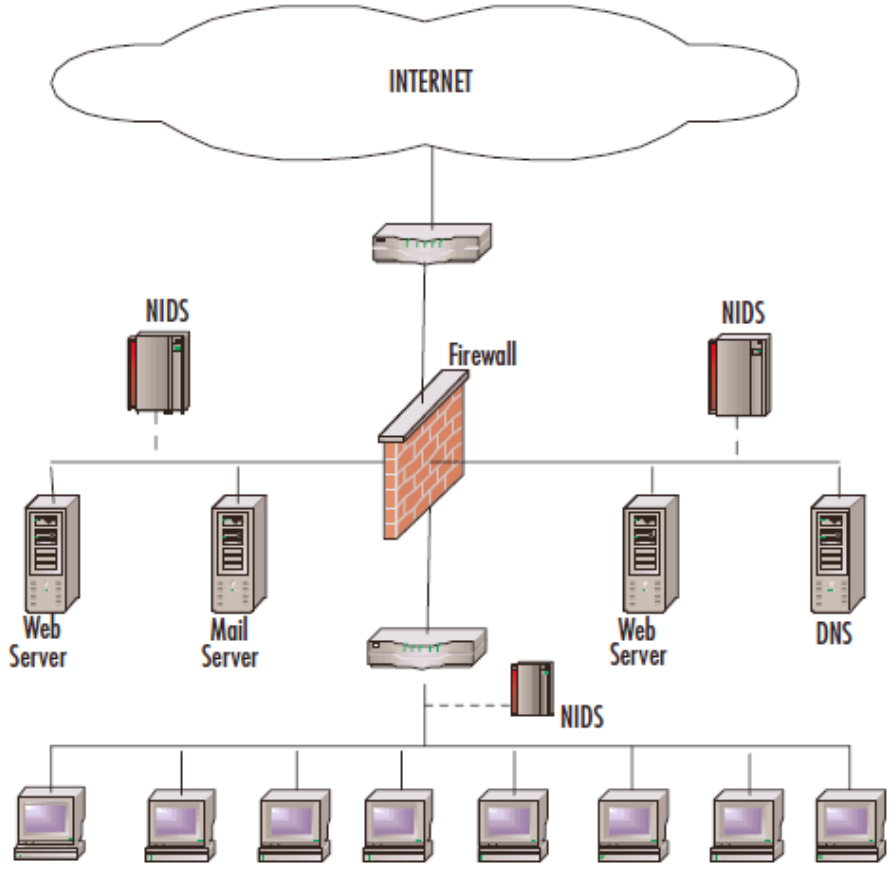
1.2 NIDS

Τα NIDS αντλούν το όνομα τους, από το γεγονός ότι παρακολουθούν το σύνολο ενός δικτύου από την σκοπιά της τοποθεσίας, όπου αυτό έχει αναπτυχθεί. Ποιο ακριβέστερα, ένα NIDS παρακολουθεί την κίνηση όλου του δικτύου ή υποδικτύου. Κανονικά, μία κάρτα δικτύου λειτουργεί σε κατάσταση όπου λαμβάνει μόνο τα πακέτα, που προορίζονται για την δική της διεύθυνση MAC (nonpromiscuous mode). Τα NIDS πρέπει να λειτουργούν έτσι ώστε να παρακολουθούν όλη την κίνηση του δικτύου (promiscuous mode). Σε αυτή τη λειτουργία τα NIDS μπορούν να αφουγκράζονται όλη την επικοινωνία στο τμήμα του δικτύου. Έτσι η κάρτα δικτύου του NIDS, πρέπει να λειτουργεί σε κατάσταση παρακολούθησης (promiscuous mode) και να συνδέσουμε το NIDS σε μία Span πόρτα του Switch ή σε μία δικτυακή τρύπα (network trap) του δικτύου.

Τα NIDS συνήθως αποτελούνται από συστήματα (**Sensors**), τα οποία τοποθετούνται σε διάφορα σημεία ενός δικτύου. Ο Sensor εκτελεί όλες τις λειτουργίες του NIDS και είναι ένα σύστημα αφιερωμένο μόνο για αυτές. Οι Sensors παρακολουθούν την κίνηση του δικτύου, αναλύουν τοπικά τα πακέτα σε πραγματικό χρόνο και καταγράφουν τα αποτελέσματά τους τοπικά ή/και απομακρυσμένα σε ένα κεντρικό σύστημα. Συνήθως οι Sensors είναι υπολογιστές Server, με μεγάλες δυνατότητες επεξεργασίας , μεγάλο αποθηκευτικό χώρο για καταγραφές στο δίσκο και πολύ καλή κάρτα δικτύου ώστε να προλαβαίνει να καταγράφει όλη την κίνηση του δικτύου.

Επίσης οι Sensors έχουν την δυνατότητα να κάνουν κρυφή την παρουσία τους (**Stealth Mode**), έτσι ώστε να μην είναι δυνατό για τον επιτιθέμενο να αντιληφθεί την θέση τους ή και την ύπαρξή τους.

Η παρακάτω εικόνα, παρουσιάζει ένα δίκτυο που χρησιμοποιεί 3 διαφορετικά NIDS. Με λίγα λόγια κάθε υποδικτύο προστατεύεται από ξεχωριστό NIDS. Αυτή είναι μια αρχιτεκτονική των NIDS που προστατεύει σε μεγάλο βαθμό όλο το δίκτυο. Αυτή η τοπολογία των NIDS μπορεί να θεωρηθεί ως περιμετρική ασφάλιση όλου του δικτύου. Μπορούμε να ξεχωρίσουμε ότι υπάρχει ξεχωριστό NIDS για τους Servers και ξεχωριστό για τους εσωτερικούς υπολογιστές του δικτύου.



Εικόνα 1: NIDS μέσα στο δίκτυο

Πλεονεκτήματα των NIDS

- Ελάχιστοι μόνο Sensors μπορούν να προστατέψουν ένα πολύ μεγάλο δίκτυο.
- Η υλοποίηση και η εφαρμογή ενός NIDS σε ένα δίκτυο επηρεάζει ελάχιστα την λειτουργία του δικτύου. Οι Sensors στους οποίους εκτελούνται οι λειτουργίες του NIDS, είναι συνήθως παθητικές συσκευές που απλά παρακολουθούν και επεξεργάζονται την κίνηση του δικτύου, χωρίς να παρεμβάλλονται στην κανονική λειτουργία του. Έτσι, είναι σχετικά εύκολο το να προστεθεί ένας Sensor σε ένα δίκτυο.
- Τα NIDS μπορούν να είναι αρκετά ασφαλή, όσο αναφορά τις επιθέσεις που μπορεί να έχουν αυτά ως στόχο, καθώς έχουν την δυνατότητα να κρύβουν την παρουσία τους από τους επιτιθέμενους.

Μειονεκτήματα των NIDS

- Τα NIDS μπορούν να παρουσιάσουν προβλήματα σε δίκτυα όπου υπάρχει πολύ μεγάλη κίνηση. Τα προβλήματα προκύπτουν όταν σε περιόδους που η κίνηση ενός τέτοιου δικτύου κυμαίνεται σε πολύ υψηλά επίπεδα, το NIDS δεν έχει τους πόρους να επεξεργαστεί όλα πακέτα, με αποτέλεσμα να αγνοήσει κάποια από αυτά, κάτι που μπορεί να οδηγήσει στην αποτυχία αναγνώρισης μίας επίθεσης. Για τον λόγο αυτό γίνονται προσπάθειες έτσι ώστε να παραχθούν NIDS τα οποία θα έχουν την μορφή Hardware, κάτι που θα τα κάνει πιο γρήγορα και πιο ανθεκτικά, αλλά συγχρόνως πιο ακριβά και λιγότερο ευέλικτα.

- Τα NIDS δεν μπορούν να αναλύσουν πληροφορία σε κρυπτογραφημένη μορφή και αυτό είναι ένα πρόβλημα που συναντάται συχνά σήμερα με την χρήση των Virtual Private Networks (VPN).

- Τα περισσότερα NIDS δεν μπορούν να καθορίσουν αν μία επίθεση ήταν επιτυχής. Αυτό που κάνουν είναι απλά να επισημάνουν το γεγονός της εμφάνισης μίας επίθεσης και των συστημάτων που είχε στόχο και στην συνέχεια είναι στην αρμοδιότητα του υπεύθυνου για αυτό το σκοπό ατόμου, να εξετάσει κάθε ένα από αυτά τα συστήματα για να εντοπίσει αν η επίθεση πέτυχε.

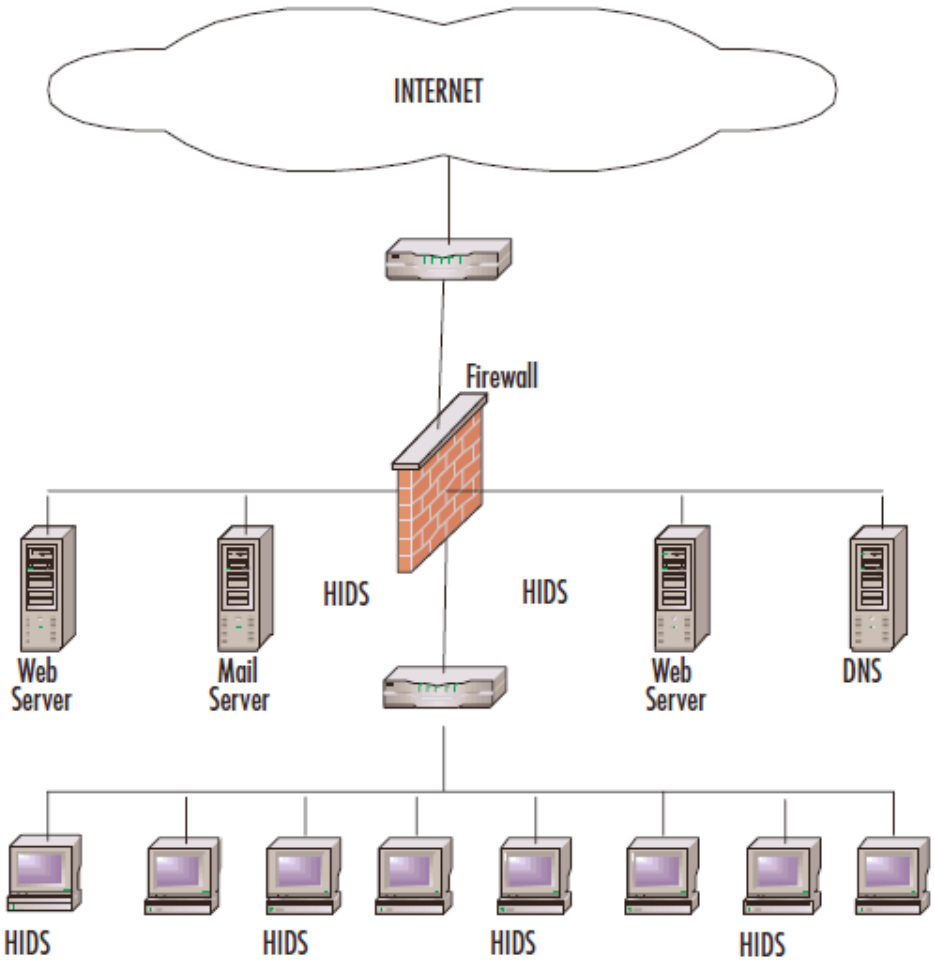
1.3 HIDS

Τα HIDS διαφέρουν σε δύο πράγματα από τα NIDS. Τα HIDS προστατεύουν μόνο τον Host στον οποίο είναι εγκατεστημένα. Επίσης η κάρτα δικτύου δε χρειάζεται να είναι σε κατάσταση που να δέχεται όλη την κίνηση του δικτύου.

Τα HIDS λειτουργούν με την πληροφορία που συλλέγεται από ένα και μόνο σύστημα το οποίο και προστατεύουν. Αυτό δίνει την δυνατότητα στα HIDS να προσφέρουν λεπτομερή πληροφόρηση για μία επίθεση, δίνοντας πληροφορίες για τις διαδικασίες (processes) και τους χρήστες που έλαβαν μέρος στην συγκεκριμένη επίθεση στο σύστημα που προστατεύουν. Η πληροφορία που κυρίως ελέγχουν τα HIDS είναι τα αρχεία καταγραφής του συστήματος. Επίσης, τα HIDS έχουν την δυνατότητα να ελέγξουν το αποτέλεσμα μίας επίθεσης, καθώς έχουν άμεση πρόσβαση, για παρακολούθηση των αρχείων και των διαδικασιών του συστήματος, που συχνά στοχεύουν οι επιτιθέμενοι.

Κάποια HIDS προσφέρουν την δυνατότητα χρήσης μίας κοινής κονσόλας διαχείρισης και ελέγχου πολλών συστημάτων, κάνοντας έτσι πιο εύκολη την χρήση τους.

Στην παρακάτω εικόνα βλέπουμε ένα δίκτυο όπου χρησιμοποιούμε HIDS σε συγκεκριμένους Servers και προσωπικούς υπολογιστές. Το κάθε HIDS είναι ενημερωμένο με τους κατάλληλους κανόνες και μόνο. Το HIDS που τρέχει στον Mail Server θα είναι ενημερωμένο μόνο με κανόνες για προστασία από Mail Server Exploits. Το HIDS στο Web Server θα περιέχει κανόνες για αντιμετώπιση των Web Exploits. Τέλος, οι προσωπικοί υπολογιστές θα είναι ρυθμισμένοι να χρησιμοποιούν τους κοινούς κανόνες για αυτούς.



Εικόνα 2: HIDS μέσα στο δίκτυο

Πλεονεκτήματα των HIDS

- Τα HIDS καθώς λειτουργούν τοπικά στο σύστημα που προστατεύουν, έχουν την ικανότητα να ανιχνεύσουν επιθέσεις που δεν ανιχνεύονται από τα NIDS.
- Μπορούν να λειτουργήσουν σε περιβάλλοντα. που η επικοινωνία μεταξύ των συστημάτων γίνεται σε κρυπτογραφημένη μορφή (VPN), καθώς εξετάζουν την πληροφορία πριν αυτή κρυπτογραφηθεί από το σύστημα αποστολέα και αφού αυτή αποκρυπτογραφηθεί από το σύστημα παραλήπτη.
- Τα HIDS έχουν την δυνατότητα να ελέγξουν και να επιβεβαιώσουν το αποτέλεσμα μίας επίθεσης στο σύστημα που προστατεύουν.
- Επιπλέον πλεονέκτημα των HIDS είναι ότι δε χρειάζεται να χρησιμοποιεί όλους τους κανόνες-υπογραφές. Ανάλογα το σύστημα που προστατεύει χρησιμοποιεί τους ανάλογους κανόνες.

Μειονεκτήματα των HIDSs

- Τα HIDS είναι δύσκολα στην διαχείρισή τους, καθώς πρέπει να ρυθμιστούν ξεχωριστά για κάθε σύστημα που παρακολουθείται.
- Τα HIDS είναι επιρρεπή σε επιθέσεις που έχουν στόχο το σύστημα που προστατεύουν. Καθώς το HIDS υλοποιείται τοπικά στο σύστημα που προστατεύει, αν ο επιτιθέμενος καταφέρει να παραβιάσει το σύστημα αυτό, τότε έχει την δυνατότητα να απενεργοποιήσει και το HIDS και να συνεχίσει ανενόχλητος.
- Τα HIDS δεν μπορούν να εντοπίσουν διάφορες αναγνωριστικές ενέργειες του επιτιθέμενου, όπως τα scans που πραγματοποιεί σε ολόκληρο το δίκτυο.
- Τα HIDS είναι επιρρεπή σε κάποιες Denial Of Service (DoS) επιθέσεις, οι οποίες μπορεί να προκαλέσουν την διακοπή της λειτουργίας τους.
- Τα HIDS επηρεάζουν αρνητικά την απόδοση του συστήματος που προστατεύουν, καθώς χρησιμοποιούν τους πόρους του για να εκτελέσουν τις λειτουργίες τους.

1.5 Τεχνικές Ανάλυσης των IDS

Τεχνική ανάλυσης ορίζουμε τον τρόπο με τον οποίο το IDS οργανώνει και επεξεργάζεται τα γεγονότα από την καταγραφή της κίνησης και αποφασίζει ποια από αυτά αποτελούν μία επίθεση ή απειλή. Οι ποιο γνωστές τεχνικές ανάλυσης είναι η **Ανίχνευση κακής χρήσης** (**Misuse Detection**), **Anomaly Detection** και η **Protocol Anomaly Detection**.

Η πρώτη είναι η τεχνική του *Misuse Detection* η οποία χρησιμοποιείται και από τα περισσότερα IDS και η οποία προσπαθεί να εντοπίσει κάτι που θεωρείται 'ύποπτο', με την έννοια ότι υπάρχει γνώση χρήσης του, σε επιθέσεις που έχουν επαναληφθεί.

Η δεύτερη είναι η τεχνική του *Anomaly Detection,* η οποία προσπαθεί να εντοπίσει πρότυπα δραστηριότητας που δεν θεωρούνται φυσιολογικά και η οποία βρίσκεται σε ερευνητικό στάδιο μέχρι σήμερα.

> **Πρότυπα**, *είναι τα δείγματα που προκύπτουν από την εκτενή παρακολούθηση και μελέτη ενός συνόλου δραστηριοτήτων και την παρουσίαση τους με την μορφή προτύπων που εκφράζουν την δραστηριότητα στο σύνολό της.*

Η τελευταία είναι η τεχνική του *Protocol Anomaly Detection,* η οποία στην ουσία αποτελεί μία παραλλαγή της *Anomaly Detection,* με την διαφορά ότι ελέγχει την δραστηριότητα που έχει σχέση με την λανθασμένη, μη φυσιολογική χρήση των πρωτοκόλλων επικοινωνίας.

Τα IDS που χρησιμοποιούν μόνο την τεχνική του *Anomaly Detection* είναι ελάχιστα, καθώς τα περισσότερα ανιχνεύουν επιθέσεις με την τεχνική του *Misuse Detection* και του *Protocol Anomaly Detection*. Κάθε μία από αυτές τις τεχνικές έχει τα πλεονεκτήματα και τα μειονεκτήματά της, ενώ η καλύτερη προσέγγιση είναι αυτή στην οποία χρησιμοποιείται κατά κύριο λόγο η τεχνική του *Misuse Detection,* η οποία συνδυάζεται με τα αποτελέσματα του *Protocol Anomaly Detection* και κάποιες έξυπνες μεθόδους του *Anomaly Detection.*

1.5.1 Ανίχνευση Κακής Χρήσης (Misuse Detection)

Με την τεχνική του *Misuse Detection* ελέγχεται η δραστηριότητα ενός δικτύου για να εντοπιστούν γεγονότα που μπορεί να ταιριάζουν με κάποια προκαθορισμένα πρότυπα γεγονότων που περιγράφουν μία γνωστή επίθεση. Τα πρότυπα αυτά ονομάζονται *Signatures* (υπογραφές) και για αυτό το λόγο η τεχνική αυτή ονομάζεται και *Signature-based detection*. Ένα *signature* μπορεί για παράδειγμα να περιγράφει κάποια χαρακτηριστικά ενός πακέτου, όπως η εμφάνιση στα data του, ενός συγκεκριμένου λεκτικού που χρησιμοποιείται για μία επίθεση.

Συνήθως για κάθε επίθεση ορίζεται και ξεχωριστό *signature*, αλλά υπάρχουν και προσεγγίσεις όπου ένα signature μπορεί να περιγράφει μία ομάδα από επιθέσεις. Η τεχνική ονομάζεται *Statebased detection*.

Πλεονεκτήματα

- Η τεχνική του *Misuse Detection* έχει την ικανότητα να ανιχνεύει επιθέσεις χωρίς να παράγει πολύ μεγάλο αριθμό από *False Positives*.
- Με την τεχνική αυτή είναι δυνατό να καθοριστεί γρήγορα και αρκετά αξιόπιστα το εργαλείο που χρησιμοποιήθηκε για να υλοποιηθεί η επίθεση.

Μειονεκτήματα

- Με την τεχνική του *Misuse Detection* μπορούν να ανιχνευτούν μόνο γνωστές επιθέσεις και για αυτό το λόγο πρέπει τα *signatures* να ανανεώνονται τακτικά ώστε να καλύπτουν νέες επιθέσεις που εμφανίζονται.
- Η αξιοπιστία της *Misuse Detection* τεχνικής στηρίζεται στην ποιότητα και την σωστή δημιουργία των *signatures* που χρησιμοποιεί. Πολλά IDS χρησιμοποιούν *signatures* που περιγράφουν αυστηρά μία συγκεκριμένη επίθεση και δεν έχουν την δυνατότητα να ανιχνεύουν διάφορες παραλλαγές αυτής. Η *state-based* τεχνική σε πολλές περιπτώσεις καταφέρνει να ξεπεράσει αυτό το πρόβλημα.

1.5.2 Anomaly Detection

Η τεχνική του *Anomaly Detection* προσπαθεί να εντοπίσει μη φυσιολογική, ασυνήθιστη συμπεριφορά ενός δικτύου ή ενός συστήματος. Λειτουργεί με την υπόθεση ότι η δραστηριότητα που παράγεται με την εμφάνιση μίας επίθεσης, παρουσιάζει διαφορές από την φυσιολογική (νόμιμη) δραστηριότητα και για

αυτό υπάρχει η δυνατότητα να ανιχνευτούν τυχόν επιθέσεις, από συστήματα που μπορούν να εντοπίσουν αυτές τις διαφορές.

Αρχικά με την μέθοδο του *Anomaly Detection* δημιουργούνται πρότυπα (patterns) που αντιπροσωπεύουν την φυσιολογική συμπεριφορά των χρηστών ή των συστημάτων ή του traffic ενός δικτύου. Τα πρότυπα αυτά χτίζονται από δεδομένα που συλλέγονται κατά την κανονική λειτουργία και αποτελούν το δείγμα φυσιολογικής δραστηριότητας. Η δημιουργία των patterns είναι το πιο δύσκολο κομμάτι της τεχνικής του *Anomaly Detection* καθώς η δραστηριότητα ενός δικτύου ή ενός συστήματος παρουσιάζει πολλές διακυμάνσεις και δεν είναι εύκολο να μοντελοποιηθεί. Στη συνέχεια συλλέγονται δεδομένα από τα γεγονότα που συμβαίνουν και με διάφορες μεθόδους εξετάζεται κατά πόσο αυτά διαφέρουν από τα patterns της φυσιολογικής δραστηριότητας.

Δυστυχώς τα IDS που χρησιμοποιούν την τεχνική του *Anomaly Detection* δεν είναι αρκετά αξιόπιστα και παρουσιάζουν πολλά συμπτώματα από False Positives και False Negatives, καθώς τα πρότυπα που εκφράζουν την φυσιολογική δραστηριότητα μπορούν να έχουν πολλές παραλλαγές.

Παρόλα αυτά με την τεχνική του *Anomaly Detection* μπορούν να ανιχνευτούν νέες, άγνωστες επιθέσεις, όπως επίσης τα αποτελέσματά τους μπορούν να χρησιμοποιηθούν σαν πηγές πληροφορίας σε IDSs που χρησιμοποιούν την τεχνική του *Misuse Detection*.

Σήμερα είναι ελάχιστα τα IDSs που κάνουν χρήση μόνο του *Anomaly Detection* και η εφαρμογή της τεχνικής αυτής είναι κυρίως για την ανίχνευση των Network και Port scans. Το *Anomaly Detection* αποτελεί ένα καθαρά ερευνητικό αντικείμενο με πολλά ελπιδοφόρα μηνύματα για το μέλλον.

Πλεονεκτήματα

- Η τεχνική του *Anomaly Detection* μπορεί να εντοπίσει κάποια ασυνήθιστη δραστηριότητα και για αυτό το λόγο μπορεί να ανιχνεύσει συμπτώματα μίας επίθεσης, χωρίς να απαιτείται η γνώση λεπτομερειών για αυτή.
- Μπορεί να ανιχνεύσει επιθέσεις που δεν έχουν επαναληφθεί και γενικότερα δεν υπάρχει προηγούμενη γνώση για αυτές.
- Μπορεί να εξάγει πληροφορίες οι οποίες στην συνέχεια να χρησιμοποιηθούν σαν είσοδο σε IDSs που κάνουν χρήση της τεχνικής του *Misuse Detection*.

Μειονεκτήματα

- Η τεχνική αυτή παρουσιάζει πολλά συμπτώματα από *False Positives* και *False Negatives*, καθώς δεν μπορούν να δημιουργηθούν αξιόπιστα και αποδοτικά patterns, λόγω της απρόβλεπτης συμπεριφοράς των χρηστών και των δικτύων.
- Για να δημιουργηθούν τα patterns της φυσιολογικής δραστηριότητας συνήθως απαιτούνται εκτεταμένα εκπαιδευτικά σύνολα που θα χρησιμοποιηθούν ως παράδειγμα.

1.5.3 Protocol Anomaly Detection

Η τεχνική αυτή έχει εμφανιστεί τα τελευταία χρόνια στο χώρο των IDS και στην ουσία είναι μία παραλλαγή της τεχνικής του *Anomaly Detection*. Η διαφορά τους βρίσκεται στο ότι η *Protocol Anomaly Detection* ελέγχει την δραστηριότητα του δικτύου όσο αναφορά την σωστή χρήση των πρωτοκόλλων επικοινωνίας και κυρίως εκείνων που ανήκουν στην οικογένεια του TCP/IP.

Τα πρωτόκολλα επικοινωνίας είναι σύνολα από αρχές και κανόνες που ορίζουν τον τρόπο με τον οποίο επιτυγχάνεται η επικοινωνία μεταξύ δύο διασυνδεδεμένων συστημάτων. Είναι γεγονός ότι ένα πολύ μεγάλο ποσοστό των επιθέσεων που λαμβάνουν χώρα στο Internet υλοποιούνται με την μη φυσιολογική χρήση των πρωτοκόλλων επικοινωνίας.

Η θεωρητική χρήση των πρωτοκόλλων ορίζεται σε επίσημα, ευρέως αποδεκτά έγγραφα τα RFC (Request For Comments) τα οποία περιγράφουν τα standards, που κάθε πρωτόκολλο πρέπει να ακολουθεί κατά την υλοποίησή του. Οι επιθέσεις που στηρίζονται στην μη φυσιολογική χρήση των πρωτοκόλλων, αποβλέπουν στο γεγονός ότι τέτοιου είδους ενέργειες έχουν παραβλεφθεί από τα RFCs ή στην κακή υλοποίηση και εφαρμογή των κανόνων που περιγράφονται στα RFCs, από διάφορους κατασκευαστές λειτουργικών συστημάτων και λογισμικών γενικότερα.

Με την τεχνική του *Protocol Anomaly Detection* παρακολουθείται και αναλύεται η δραστηριότητα που έχει σχέση με την χρήση των πρωτοκόλλων και ελέγχεται για το αν αυτή συμφωνεί με κάποια patterns τα οποία περιγράφουν την φυσιολογική, νόμιμη χρήση των πρωτοκόλλων. Η δημιουργία των patterns είναι πιο εύκολη υπόθεση σε σχέση με αυτή στην τεχνική του *Anomaly Detection,* καθώς σε αυτήν την περίπτωση τα patterns αποτελούνται από τους προκαθορισμένους κανόνες που περιγράφονται από τα RFCs και όχι από κανόνες που περιγράφουν την φυσιολογική δραστηριότητα ενός δικτύου ή ενός συστήματος που μπορεί να παρουσιάζει πολλές διακυμάνσεις.

Πλεονεκτήματα

- Η τεχνική του *Protocol Anomaly Detection* μπορεί να εντοπίσει κάποια ασυνήθιστη δραστηριότητα που αφορά μη φυσιολογική χρήση κάποιου πρωτοκόλλου και για αυτό το λόγο μπορεί να ανιχνεύσει συμπτώματα μίας επίθεσης χωρίς να απαιτείται η γνώση λεπτομερειών για αυτή.
- Μπορεί να ανιχνεύσει επιθέσεις που δεν έχουν επαναληφθεί και γενικότερα δεν υπάρχει προηγούμενη γνώση για αυτές.
- Μπορεί να εξάγει πληροφορίες οι οποίες στην συνέχεια να χρησιμοποιηθούν σαν είσοδο σε IDSs που κάνουν χρήση της τεχνικής του *Misuse Detection*.

Μειονεκτήματα

- Η δημιουργία των patterns δεν μπορεί πάντα να ακολουθεί πιστά στους κανόνες που ορίζονται από τα RFCs, καθώς δεν συμβαίνει το ίδιο και από τα λειτουργικά συστήματα και τα άλλα λογισμικά που κάνουν χρήση των πρωτοκόλλων. Τα patterns που δημιουργούνται πρέπει να λαμβάνουν το γεγονός αυτό υπόψη τους.

- Δεν μπορούν να ανιχνεύσουν επιθέσεις που δεν στηρίζονται στην μη φυσιολογική χρήση των πρωτοκόλλων.
- Όταν ανιχνευτεί μία επίθεση με την τεχνική αυτή, συνήθως δεν προσφέρονται πληροφορίες που να περιγράφουν επαρκώς το είδος της και απαιτείται η συμμετοχή εξειδικευμένων ατόμων που να μπορούν να ερμηνεύσουν τα αποτελέσματα που παράγονται.

1.6 Απόκριση (Responses) των IDSs

Απόκριση, Είναι το σύνολο των ενεργειών που θα εκτελέσει το IDS, όταν ανιχνεύσει μία επίθεση. Υπάρχουν δύο είδη τέτοιων ενεργειών, *οι Παθητικές (Passive)* και *οι Ενεργητικές (Active).* Τα Active Responses έχουν να κάνουν με αυτοματοποιημένη αντιμετώπιση της επίθεσης από το ίδιο το IDS. Τα Passive Responses συνήθως καταγράφουν το γεγονός της επίθεσης και ενημερώνουν με κάποιο τρόπο τους υπεύθυνους, ώστε αυτοί να πάρουν τα κατάλληλα μέτρα.

1.6.1 Ενεργές Αποκρίσεις (Active Responses)

Τα *Active Responses* είναι αυτοματοποιημένες ενέργειες που εκτελούνται από το IDS, όταν ανιχνεύσει συγκεκριμένους τύπους επιθέσεων. Υπάρχουν τρεις κατηγορίες των *Active Responses***:**

Συλλογή επιπρόσθετων πληροφοριών.

Αυτή είναι ίσως η λιγότερο ενεργητική αντίδραση αλλά σε ορισμένες περιπτώσεις η πιο παραγωγική. Η λειτουργία της είναι να συλλεχτούν περισσότερες πληροφορίες για μία πιθανή επίθεση που εντοπίστηκε, οι οποίες θα ξεκαθαρίσουν περισσότερο την κατάσταση, ώστε να ληφθεί η κατάλληλη απόφαση για το αν πρέπει να παρθούν κάποια παραπέρα μέτρα προστασίας. Έτσι κάποιο IDS όταν εντοπίσει μία πιθανή επίθεση, μπορεί για παράδειγμα να αυξήσει το επίπεδο ευαισθησίας των Information Sources που χρησιμοποιεί (πχ. να ρυθμίσει κάποιον Sensor να καταγράφει όλα τα πακέτα ενός δικτύου και όχι αυτά που αφορούν συγκεκριμένα συστήματα ή πόρτες).

Με την συλλογή της επιπρόσθετης πληροφορίας γίνεται δυνατό να συλλεχθούν περισσότερα στοιχεία για μία πιθανή επίθεση, τα οποία εξυπηρετούν τόσο στην αποφυγή λανθασμένων συμπερασμάτων που μπορεί να προκύψουν διαφορετικά, όσο και στον εντοπισμό και την ποινική δίωξη του επιτιθέμενου.

Παρεμπόδιση του επιτιθέμενου.

Ένας άλλος τύπος του *Active Response*, είναι η αναχαίτιση της επίθεσης την ώρα που πραγματοποιείται και στη συνέχεια ή παρεμπόδιση της παραπέρα πρόσβασης του επιτιθέμενου στο προστατευόμενο σύστημα ή δίκτυο. Στην ουσία το IDS εμποδίζει τα πακέτα που έχουν IP διεύθυνση, από την οποία φαίνεται ότι προέρχεται ο επιτιθέμενος και όχι τον ίδιο τον επιτιθέμενο προσωπικά. Πολλές φορές αυτό δεν αποτελεί αξιόπιστη λύση, καθώς οι πιο έμπειροι επιτιθέμενοι χρησιμοποιούν ψεύτικες IP διευθύνσεις. Παρόλα αυτά με τέτοιου είδους ενέργειες είναι δυνατό να εμποδιστούν οι πιο αρχάριοι και να αποθαρρυνθούν οι πιο έμπειροι, που υλοποιούν μία επίθεση. Τέτοιες ενέργειες περιλαμβάνουν :

- Να σταλούν πακέτα (με ενεργοποιημένο το RST flag στον TCP header) τα οποία θα τερματίσουν οποιαδήποτε σύνδεση του επιτιθέμενου με το σύστημα – στόχο.
- Να ρυθμιστούν οι routers και τα firewall του δικτύου, ώστε να μην επιτρέπουν την διέλευση οποιουδήποτε πακέτου, το οποίο έχει διεύθυνση αποστολέα ή παραλήπτη, την IP διεύθυνση την οποία χρησιμοποιεί ο επιτιθέμενος στα πακέτα που στέλνει.
- Να ρυθμιστούν οι routers και τα firewall του δικτύου, ώστε να μην είναι δυνατή πρόσβαση σε πόρτες υπηρεσίες και πρωτόκολλα που κάνει χρήση ο επιτιθέμενος.

Δράση εναντίον του επιτιθέμενου.

Υπάρχουν πολλές σκέψεις για το αν είναι σωστό κατά την ανίχνευση μίας επίθεσης να παρθούν μέτρα που συμπεριλαμβάνουν την δράση εναντίον του επιτιθέμενου. Στην πιο ακραία μορφή της αυτή η δράση θα μπορούσε να είναι η υλοποίηση επίθεσης με στόχο τον επιτιθέμενο ή η συλλογή πληροφοριών για το δίκτυό του. Παρόλο που αυτή η αντιμετώπιση μοιάζει αρκετά αποτελεσματική και δίκαιη, κρύβει πολλούς κινδύνους.

Κατά πρώτο λόγο αυτού του είδους η δράση μπορεί να είναι παράνομη. Επιπρόσθετα καθώς πολλοί επιτιθέμενοι χρησιμοποιούν ψεύτικες IP διευθύνσεις όταν εξαπολύουν μία επίθεση, τέτοιου είδους δράση θα μπορούσε να προκαλέσει ζημιές σε λάθος χρήστες ή και δίκτυα. Τέλος κάτι τέτοιο θα μπορούσε να προκαλέσει περισσότερο τον επιτιθέμενο και αυτός να αντιδράσει εξαπολύοντας μία επίθεση που θα μπορούσε να έχει καταστροφικά αποτελέσματα.

Τέτοιου είδους δράση εναντίον του επιτιθέμενου πρέπει να γίνεται με πολύ προσοχή και πριν κάποιος αποφασίσει να υιοθετήσει αυτήν την τεχνική, καλό είναι να έχει συμβουλευτεί κάποιον ιδικό για τα νομικά θέματα που προκύπτουν.

1.6.2 Παθητικές Αποκρίσεις (Passive Responses)

Τα *Passive Responses* είναι η μέθοδος με την οποία το IDS απλά προμηθεύουν τους αρμόδιους χρήστες, με τις πληροφορίες που αφορούν την ανίχνευση μίας επίθεσης. Στη συνέχεια είναι στην ευθύνη των αρμοδίων να δράσουν κατάλληλα, εκμεταλλευόμενοι τις πληροφορίες αυτές. Αυτού του είδους η αντίδραση είναι και η πιο συνήθης από τα περισσότερα IDS.

Υπάρχουν διάφοροι τρόποι με τους οποίους μπορεί ένα IDS να γνωστοποιήσει τα αποτελέσματά του στους αρμόδιους χρήστες.

Ανακοίνωση των Alerts

Αυτή η τεχνική έχει να κάνει με τον τρόπο που ένα IDS ανακοινώνει και παρουσιάζει στους αρμόδιους χρήστες, τις επισημάνσεις του για μία επίθεση. Μία επισήμανση για την ανίχνευση κάποιας επίθεσης, συνήθως ονομάζεται Alert.

Τα περισσότερα IDSs δίνουν την δυνατότητα στον χρήστη να καθορίσει με σχετική ευχέρεια, την στιγμή και την μορφή που θα παράγονται τα alerts και σε ποιους χρήστες θα παρουσιάζονται.

Ένα IDS είναι δυνατόν να ρυθμιστεί ώστε τα alerts να εμφανίζονται σε πραγματικό χρόνο, την ώρα που εντοπίζεται μία επίθεση, όπως για παράδειγμα με αναδυόμενα παράθυρα στην οθόνη ή μπορεί να ρυθμιστεί ώστε να καταγράφει τα alerts σε κάποιο αρχείο για μετέπειτα εξέταση.

Η μορφή που θα παράγεται ένα alert από το IDS, μπορεί να είναι από μία απλή αναφορά στο είδος της επίθεσης με έναν τίτλο, στον επιτιθέμενο και στο θύμα αυτής, μέχρι και αναλυτική αναφορά που θα περιέχει και πληροφορίες για το πακέτο που οδήγησε στον εντοπισμό της επίθεσης, κάνοντας λεπτομερή περιγραφή του ή αναφορά στο εργαλείο που χρησιμοποιήθηκε για την υλοποίησή της.

Επίσης κάποια IDS έχουν την δυνατότητα να πληροφορούν με Alerts απομακρυσμένα τους εξουσιοδοτημένους χρήστες, είτε με αποστολή e-mail σε αυτούς, είτε ακόμα μέσω κλήσεων ή αποστολή γραπτών μηνυμάτων σε κινητά τηλέφωνα που ανήκουν σε αυτούς.

SNMP Traps

Κάποια IDS έχουν την δυνατότητα να αναφέρουν τα alerts που παράγουν, σε ένα κεντρικό σύστημα διαχείρισης του δικτύου με την χρήση SNMP Traps. Έτσι με την αποστολή σε ένα κεντρικό σύστημα, των Alerts που παράγονται από διάφορα IDS ενός δικτύου, καθώς και άλλων πληροφοριών που εξάγονται από άλλους μηχανισμούς ασφάλειας, όπως Firewalls, είναι δυνατό να γίνει ευκολότερα συσχετισμός μεταξύ των αποτελεσμάτων που έχουν προκύψει από διαφορετικές πηγές και να σχηματιστεί μία πιο σαφής και λεπτομερής εικόνα των γεγονότων.

1.7 Σύστημα Πρόληψης Εισβολών (IPS)

Τα συστήματα πρόληψης επιθέσεων είναι το λογικό επόμενο βήμα των συστημάτων ανίχνευσης επιθέσεων. Από τη στιγμή που ανιχνεύεται μια επίθεση είναι θεμιτό να αντιμετωπιστεί και αυτόματα. Μάλιστα η αντιμετώπιση αυτή είναι καλό να γίνει χωρίς την ανθρώπινη παρέμβαση για να ελαχιστοποιηθεί ο χρόνος αντίδρασης στην επίθεση. Συχνά ένα **IPS** (Intrusion Prevention System) αποτελεί μέρος ενός IDS και δεν αποτελούν ξεχωριστές οντότητες. Αυτό συμβαίνει επειδή το IDS είναι αυτό που ανιχνεύει μια επίθεση, οπότε αυτό πρέπει να πάρει και την πρωτοβουλία για να την σταματήσει.

Στην ουσία ένα IPS, συνδυάζει τα χαρακτηριστικά ενός *Firewall* και ενός IDS, καθώς μπορεί και μπλοκάρει την ανεπιθύμητη κίνηση έχοντας την βοήθεια ανίχνευσης των κακόβουλων πακέτων που προσφέρει ένα IDS. Η διαφορά του από το firewall είναι ότι έχει καλύτερη και πιο ολοκληρωμένη πληροφορία. Αυτό οφείλεται στην ύπαρξη του IDS, το οποίο και παρακολουθεί όλη την δικτυακή κίνηση.

Τα συστήματα αποτροπής επιθέσεων συναντώνται στους παρακάτω τύπους:

- **Host-Based** (HIPS), βρίσκονται σε ένα συγκεκριμένο μηχάνημα και παρακολουθούν την κίνηση και κάποια άλλα στοιχεία προκειμένου να αποτρέψουν επιθέσεις.
- **Network-based** (NIPS), όπου η εφαρμογή IPS ή το υλικό που παίζει το ρόλο του IPS βρίσκεται στο δίκτυο και έχει IP αυτού του δικτύου. Τα συστήματα αυτά αναλύουν, βρίσκουν, και αναφέρουν συμβάντα που έχουν να κάνουν με ζητήματα ασφάλειας. Είναι σχεδιασμένα ώστε να ελέγχουν τη δικτυακή κίνηση.
- **Content-based** (CBIPS), είναι τα συστήματα εκείνα που παρακολουθούν το περιεχόμενο των πακέτων για μοναδικές ακολουθίες, τις οποίες ονομάζουμε υπογραφές, με σκοπό να αναγνωρίσουν και να αποτρέψουν γνωστούς τύπους επιθέσεων.
- **Rate-based** (RBIPS), είναι τα συστήματα εκείνα που στόχο έχουν να αποτρέψουν επιθέσεις τύπου άρνησης εξυπηρέτησης. Η λειτουργία τους βασίζεται στο γεγονός ότι παρακολουθούν και μαθαίνουν φυσιολογικές δικτυακές συμπεριφορές. Με βάση λοιπόν την πραγματικού χρόνου παρακολούθηση του δικτύου και κάποια στατιστικά στοιχεία μπορούν να αναγνωρίσουν κάποια μη αποδεκτά όρια για συγκεκριμένους τύπους κίνησης, π.χ. TCP, UDP. Οι επιθέσεις αναγνωρίζονται όταν ξεπεραστούν κάποια φράγματα. Τα φράγματα αυτά προσαρμόζονται δυναμικά ανάλογα με την ημέρα, την ώρα, τη βδομάδα κ.α. Όταν η επίθεση αναγνωριστεί, τότε εφαρμόζονται διάφοροι μηχανισμοί αποτροπής όπως είναι για παράδειγμα το port/protocol filtering (black-listing, white-listing).

Οι βασικές περιοχές έρευνας σε αυτό το τομέα είναι παρόμοιες με αυτές των IDS.

Κεφάλαιο 2 Snort

2.1 Τι είναι το Snort;

Το Snort, είναι ένα ελεύθερο και ανοιχτού κώδικα λογισμικό. Λειτουργεί ως **Network Intrusion Prevention System** (NIPS) και **Network Intrusion Detection System** (NIDS). Το Snort δημιουργήθηκε από τον Martin Roeschto 1998. Πλέον αναπτύσσετε από την Sourcefire, όπου ο Roesch ήταν ιδρυτής της.

ToSnort έχει την ικανότητα να εκτελεί σε πραγματικό χρόνο ανάλυση κίνησης και καταγραφή πακέτων, σε επίπεδο δικτύου. Έτσι έχει την δυνατότητα να ανιχνεύει επιθέσεις από κακόβουλους χρήστες. Αυτό το πετυχαίνει ελέγχοντας την κίνηση των πακέτων χρησιμοποιώντας κάποιους κανόνες ή υπογραφές που θα αναλύσουμε αργότερα.

Το Snort μπορεί να ρυθμιστεί να τρέχει σε τρεις λειτουργίες:

- Sniffer mode
- Packet logger
- NIDS mode

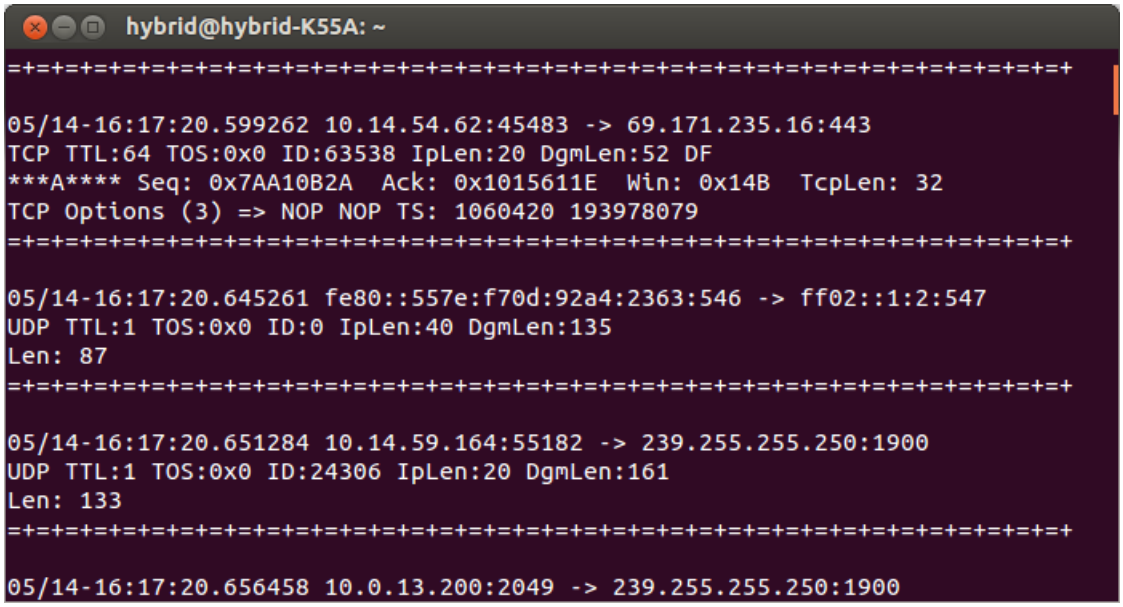
2.1.1 Sniffer mode

Σε αυτή την κατάσταση το Snort απλά διαβάζει τα πακέτα από το δίκτυο και άπλα μας τα δείχνει στην κονσόλα σε συνεχόμενη ροή. Ανάλογα με το τί πληροφορία θέλουμε να εμφανίσουμε από τα πακέτα, ξεκινάμε το Snort με τις κατάλληλες παραμέτρους.

Αν θέλουμε να εκτυπώσουμε στην κονσόλα τις κεφαλές από τα TCP/IP και UDP/ICMP πακέτα, ξεκινάμε το Snort με την παρακάτω εντολή:

./snort –v

Στο παρακάτω σχήμα φαίνεται το Snort να λειτουργεί σε αυτή την κατάσταση.

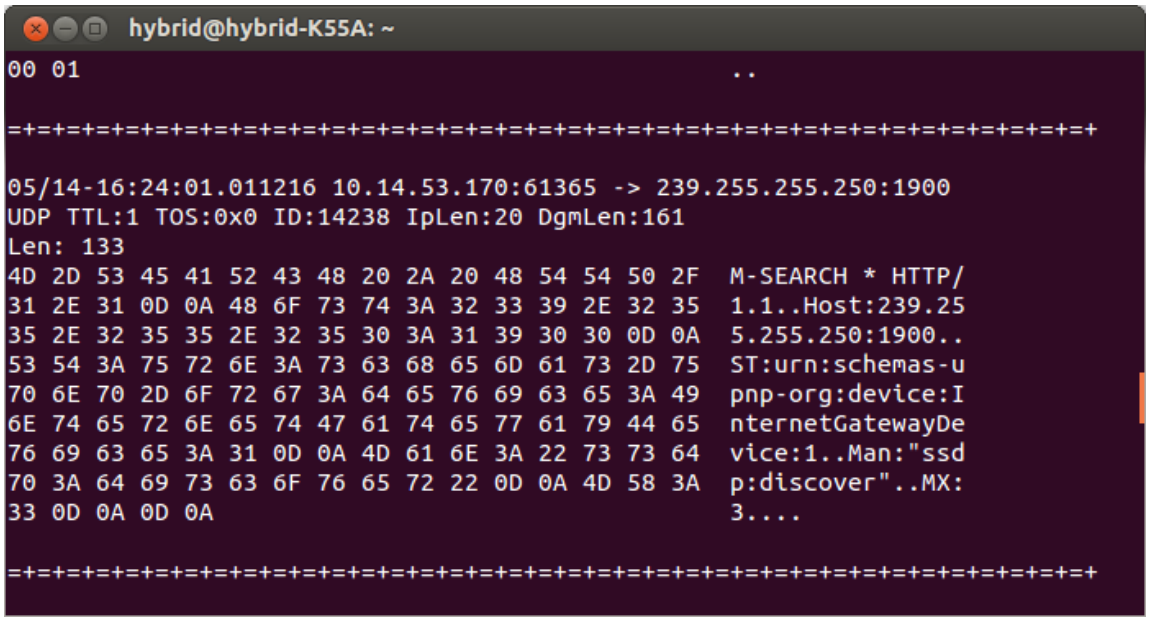


Εικόνα 3: Sniffer mode απλή εμφάνιση των κεφαλών των πακέτων

Αν επιπλέων θέλουμε να δούμε τα δεδομένα εφαρμογής των πακέτων, τότε ξεκινάμε το Snortμε την παρακάτω εντολή:

./snort –vd

Στο παρακάτω σχήμα φαίνεται το Snort να λειτουργεί σε αυτή την κατάσταση.

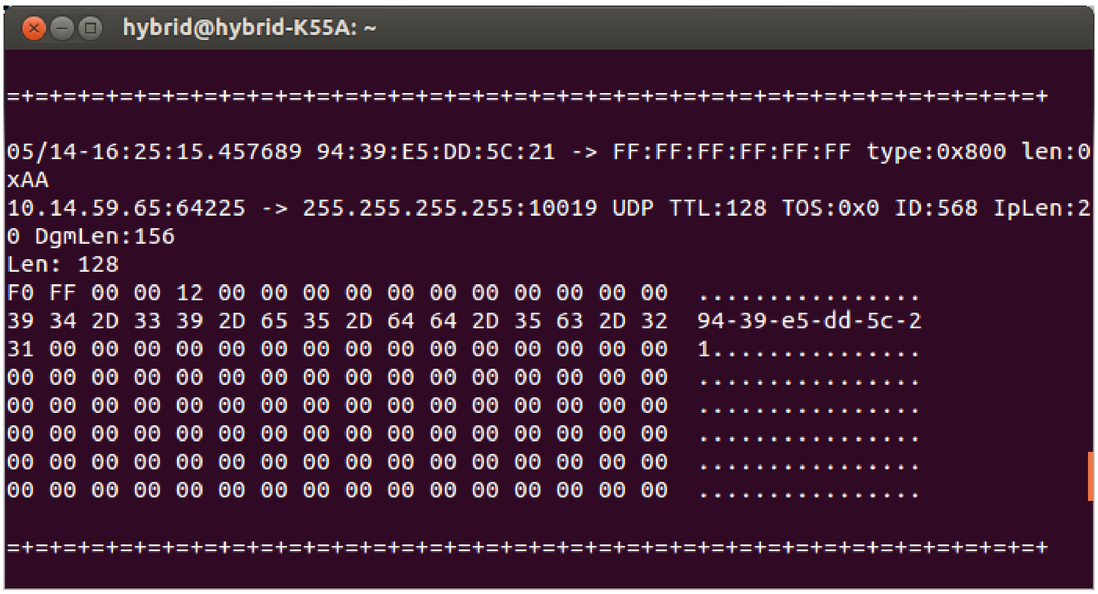


Εικόνα 4: Sniffer mode εμφάνιση δεδομένων εφαρμογής των πακέτων

Αν θέλουμε το Snort να μας δείξει τα πακέτα σε μία ποιό περιγραφική μορφή, δηλαδή να μας δείξει τις κεφαλές από τα πακέτα σε επίπεδο γραμμής δεδομένων (data link layer), τότε ξεκινάμε το Snort με την παρακάτω εντολή:

./snort –vde

Στο παρακάτω σχήμα φαίνεται το Snort να λειτουργεί σε αυτή την κατάσταση



Εικόνα 5: Sniffer mode: Εμφάνιση δεδομένων κεφαλής των πακέτων σε επίπεδο γραμμής δεδομένων

2.1.2 Packet Logger Mode

Με την παραπάνω λειτουργία το Snort απλά παρουσιάζει την κίνηση των πακέτων στην κονσόλα. Αν θέλουμε να καταγράψουμε αυτή την κίνηση στο δίσκο του συστήματος μας πρέπει να καθορίσουμε ένα φάκελο στο σύστημα μας (π.χ. log), όπου θα αποθηκευτεί ένα αρχείο καταγραφής, ώστε να μπορούμε να το επεξεργαστούμε όποτε θέλουμε.

Με την παρακάτω εντολή θα ξεκινήσουμε το Snort σε λειτουργία καταγραφής πακέτων IP και TCP/UDP/ICMP με πληροφορίες της κεφαλίδας των πακέτων σε επίπεδο γραμμής δεδομένων και τα δεδομένα εφαρμογής των πακέτων.

./snort -dev -l ./log

- **-dev**: πληροφορίες της κεφαλίδας των πακέτων σε επίπεδο γραμμής δεδομένων και τα δεδομένα εφαρμογής των πακέτων
- **./log**: Ορισμός καταλόγου στο σύστημα μας, όπου θα καταγράφονται τα πακέτα.

Αν θέλουμε να καθορίσουμε να γίνεται καταγραφή των πακέτων από συγκεκριμένη ομάδα διευθύνσεων IP, τότε πρέπει να προσθέσουμε άλλη μια παράμετρο και η εντολή που θα ξεκινάμε το Snort θα είναι:

./snort -dev -l ./log -h 192.168.1.0/24

Αυτή η εντολή θα καταγράφει τα πακέτα στο φάκελο log από το 192.168.1.0 δίκτυο κλάσης C. Επιπλέον τα πακέτα θα καταγράφονται σε υποκατηγορίες μέσα σε φάκελο που το όνομα του θα βασίζεται στη διεύθυνση του απομακρυσμένου υπολογιστή (non-192.168.1).

Αν το δίκτυο μας υποστηρίζει μεγάλες ταχύτητες και η κίνηση είναι μεγάλη μπορούμε να επιταχύνουμε την καταγραφή κάνοντας καταγραφή σε δυαδική μορφή. Δυαδική λειτουργία καταγράφει τα πακέτα σε tcpdump μορφή σε απλό δυαδικό αρχείο μέσα στο φάκελο καταγραφής που έχουμε ορίσει. Η παράμετρος που τρέχει το Snort σε αυτή τη μορφή είναι η –b:

/snort -l ./log –b

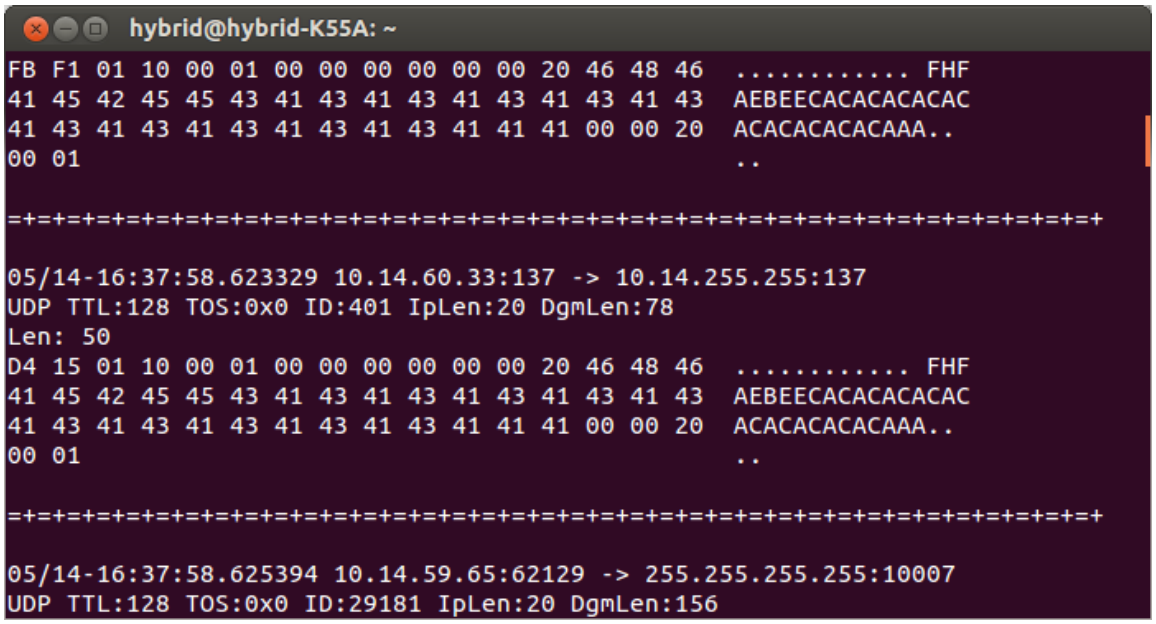
Αν θέλουμε να διαβάσουμε αυτά τα αρχεία καταγραφής σε δυαδική μορφή, το Snort μας δίνει αυτή τη δυνατότητα με την παρακάτω εντολή:

./snort -dv -r packet.log

Αν θέλουμε να εμφανίσουμε από ένα αρχείο καταγραφής μόνο πακέτα ICMP τότε θα πρέπει να ορίσουμε ένα BPF φίλτρο. Η παρακάτω εντολή θα εμφανίσει μόνο τα πακέτα ICMP από το αρχείο καταγραφής:

./snort -dvr packet.log icmp

Στο παρακάτω σχήμα έχουμε εμφανίσει στη κονσόλα μόνο τα πακέτα UDP από ένα αρχείο log.



Εικόνα 6: Logger mode: Εμφάνιση μόνο UDP πακέτων από το log αρχείο στο δίσκο

2.1.3 NIDS mode

Αυτή η κατάσταση του Snort είναι η ποιό ενδιαφέρουσα και στην πραγματικότητα η λειτουργία που κάνει το Snort να συμπεριφέρεται σαν Σύστημα Ανίχνευσης Εισβολών βασισμένο στο Δίκτυο. Η παράμετρος, που κάνει το Snort να λειτούργει σε αυτή την κατάσταση, είναι ο προσδιορισμός του αρχείου ρυθμίσεων (configuration file) του Snort. Είναι ένα αρχείο που περιέχει ρυθμίσεις για την λειτουργία του Snort σαν NIDS. Αναφερόμαστε σε αυτό το αρχείο σε άλλη ενότητα. Αυτό το αρχείο ονομάζεται snort.conf. Η παρακάτω εντολή ξεκινάει το Snort σε κατάσταση NIDS χρησιμοποιώντας αυτό το αρχείο.

./snort -d -h 192.168.1.0/24 -l ./log -c snort.conf

Το Snort θα χρησιμοποιήσει το αρχείο ρυθμίσεων snort.conf και θα καταγράψει στο δίσκο τα πακέτα που έχουν πυροδοτήσει συναγερμούς βάση των κανόνων που έχουν ενεργοποιηθεί από το αρχείο ρυθμίσεων που ορίσαμε. Με αυτή την λειτουργία του Snort θα ασχοληθούμε στο υπόλοιπο αυτής της εργασίας ώστε να ανιχνεύσουμε γνωστές επιθέσεις.

2.2 Αρχιτεκτονική του Snort

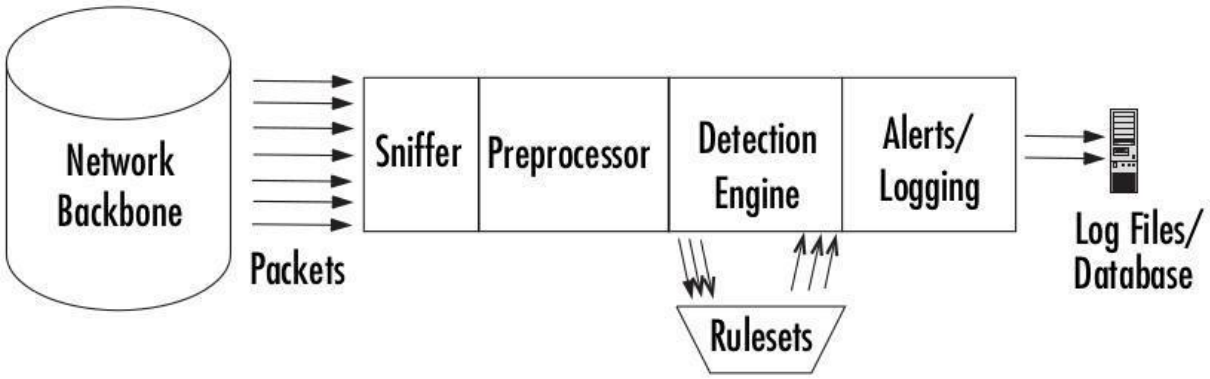
Το Snort έχει πολλές δυνατότητες και λειτουργίες. Πριν ξεκινήσουμε να παρουσιάζουμε τα χαρακτηριστικά του Snort είναι σημαντικό να καταλάβουμε πώς λειτουργεί. Στην παρούσα φάση θα μας βοηθήσει η αρχιτεκτονική του. Η αρχιτεκτονική του Snort, αποτελείτε από τέσσερα βασικά κομμάτια:

- Τον Sniffer.
- Τον Προεπεξεργαστή (Preprocessor)
- Την Μηχανή Ανίχνευσης (Detection Engine)
- Την Έξοδο (Output)

Στην πιο βασική του μορφή, το Snort είναι ένα *sniffer* πακέτων. Ωστόσο, είναι σχεδιασμένο να λαμβάνει πακέτα και να τα επεξεργάζεται μέσω του *Προεπεξεργαστή*, και στη συνέχεια να ελέγχει τα πακέτα βάση κάποιων *κανόνων-υπογραφών* (μέσω της μηχανής ανίχνευσης).

Στο παρακάτω σχήμα περιγράφεται η αρχιτεκτονική του Snort.

1. Αρχικά το Snort συλλέγει όλα τα πακέτα του δικτύου και τα αποκωδικοποιεί για να βρει χρήσιμες πληροφορίες για κάθε πακέτο (Sniffer).
2. Στην συνέχεια όλα τα πακέτα πηγαίνουν στον Προεπεξεργαστή (Preprocessor) όπου χρησιμοποιούνται διάφορα plug-ins για έλεγχο ύποπτης δραστηριότητας.
3. Μετά η Μηχανή Ανίχνευσης (DetectionEngine) χρησιμοποιεί κανόνες και ελέγχει κάθε πακέτο. Αν υπάρχει ύποπτη δραστηριότητα βάση των κανόνων, τότε γεννιέται ένα Alert.
4. Αυτά τα Alerts καταγράφονται στο δίσκο ή στην οθόνη και στην συνέχεια μπορούμε να τα αποθηκεύσουμε σε μια βάση δεδομένων για περισσότερη ανάλυση.

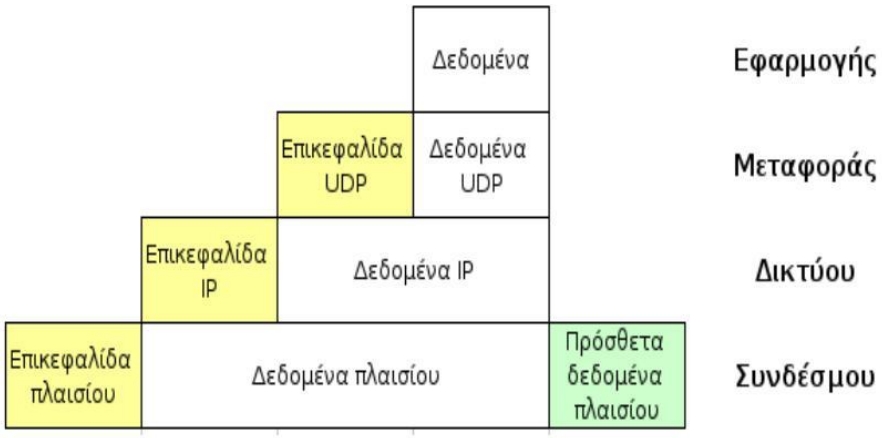


Εικόνα 7: Αρχιτεκτονική του Snort

2.2.1 PacketSniffer

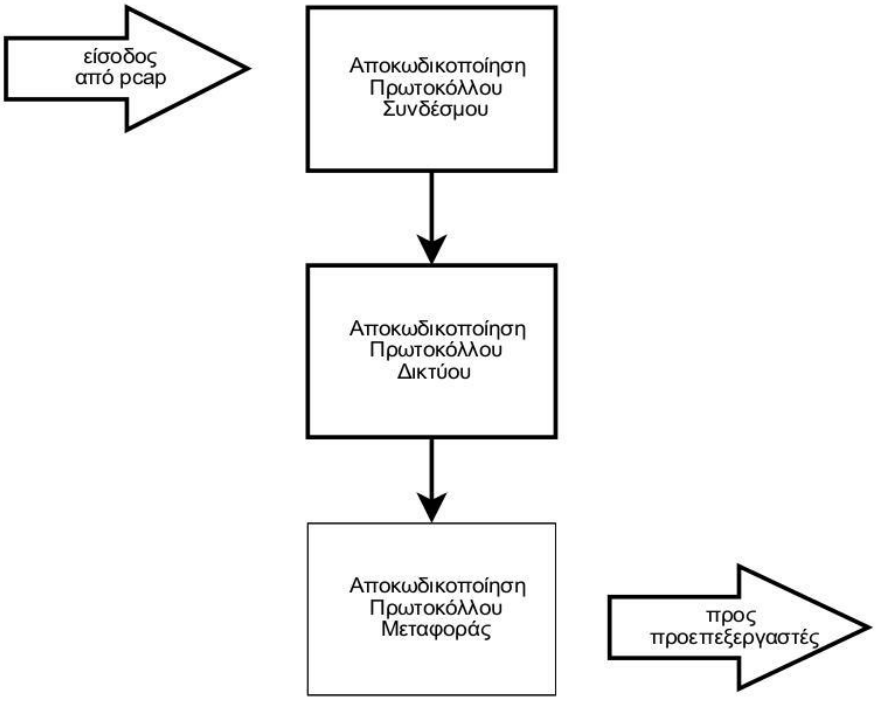
Το *PacketSniffer*, είναι ένα υλικό ή ένα πρόγραμμα, που μπορεί να διαβάζει όλη την κίνηση σε κάποιο σημείο στο δίκτυο. Στην περίπτωση μας, το Snort που είναι εγκατεστημένο στον υπολογιστή μας, διαβάζει όλη την κίνηση, με την βοήθεια της βιβλιοθήκης Libpcam. Είναι μια βιβλιοθήκη που μπορούν να χρησιμοποιήσουν οι εφαρμογές παρακολούθησης για σύλληψη πακέτων ενός δικτύου στο επίπεδο ζεύξης (link layer). Ως υλικό, χρησιμοποιεί την κάρτα δικτύου του υπολογιστή μας για να συλλέξει τα πακέτα. Όπως καταλαβαίνουμε για να συλλέξουμε όλα τα πακέτα του δικτύου, η κάρτα μας πρέπει να είναι συνδεδεμένη σε σημείο που περνάνε όλα τα πακέτα του δικτύου.

Όλα τα πακέτα που συλλέγουμε, είναι σε επίπεδο ζεύξης. Πρέπει με κάποιο τρόπο να αποκωδικοποιηθούν από όλα τα στρώματα, ώστε να συλλέξουμε την χρήσιμη πληροφορία κάθε πακέτου.



Εικόνα 8: Δομή πακέτου στα διάφορα επίπεδα OSI

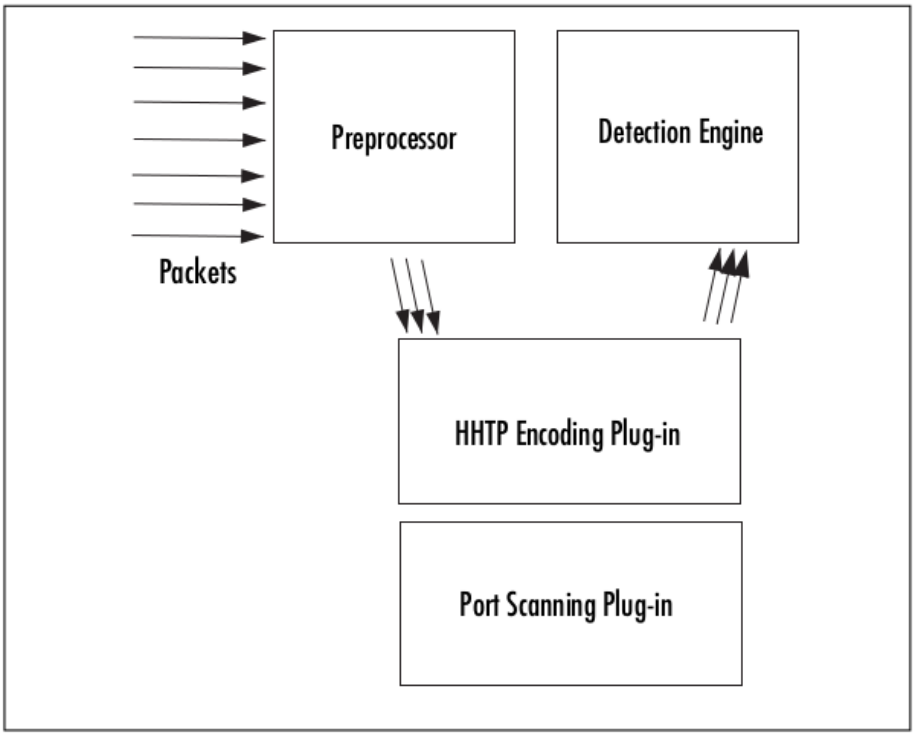
Αυτό το πετυχαίνει το Snort με τους *αποκωδικοποιητές* (Decoders). Όπου παίρνουν σαν είσοδο ότι έχει συλλέξει το Snort με την βοήθεια της Libpcam. Στην συνέχεια αποκωδικοποιούν τα πακέτα από το επίπεδο συνδέσμου, μέχρι το επίπεδο μεταφοράς και στέλνουν την χρήσιμη πληροφορία στους Προεπεξεργαστές (Preprocessors).



Εικόνα 9: Διαδρομή του δικτυακού πακέτου μέσα στον αποκωδικοποιητή πακέτων.

2.2.2 Προεπεξεργαστής (Preprocessor)

Ο Προεπεξεργαστής (Preprocessor) δέχεται τα ακατέργαστα πακέτα από τους αποκωδικοποιητές και τσεκάρει τα πακέτα με συγκεκριμένα plug-ins (όπως ένα RPC plug-in, ένα HTTP plug-in, και ένα port scanner plug-in). Αυτά τα plug-ins τσεκάρουν για κάποιο ιδιαίτερο τύπο συμπεριφοράς των πακέτων. Αν τα πακέτα είναι καθορισμένα να έχουν συγκεκριμένο τύπο συμπεριφοράς, τότε στέλνονται στην *Μηχανή Ανίχνευσης* (DetectionEngine). Στο παρακάτω σχήμα μπορούμε να δούμε πώς ο Προεπεξεργαστής χρησιμοποιεί τα plug-ins για να ελέγχει τα πακέτα.



Εικόνα 10: Ο Προεπεξεργαστής του Snort

Ο Προεπεξεργαστής είναι ένα πολύ σημαντικό χαρακτηριστικό του IDS λόγω του ότι τα plug-ins μπορούν να ενεργοποιηθούν ή να απενεργοποιηθούν κατά βούληση του διαχειριστή του. Αν για παράδειγμα, δεν επιθυμούμε να ελέγχουμε το δίκτυο μας για σαρώσεις θυρών (Port Scanning), μπορούμε να απενεργοποιήσουμε το εν λόγω plug-in, χωρίς να επηρεαστεί καθόλου το υπόλοιπο σύστημα μας. Οι παράμετροι που αφορούν τους Προεπεξεργαστές διαμορφώνονται μέσω του αρχείου snort.conf, όπως θα δούμε και αργότερα.

2.2.3. Μηχανή Ανίχνευσης (DetectionEngine)

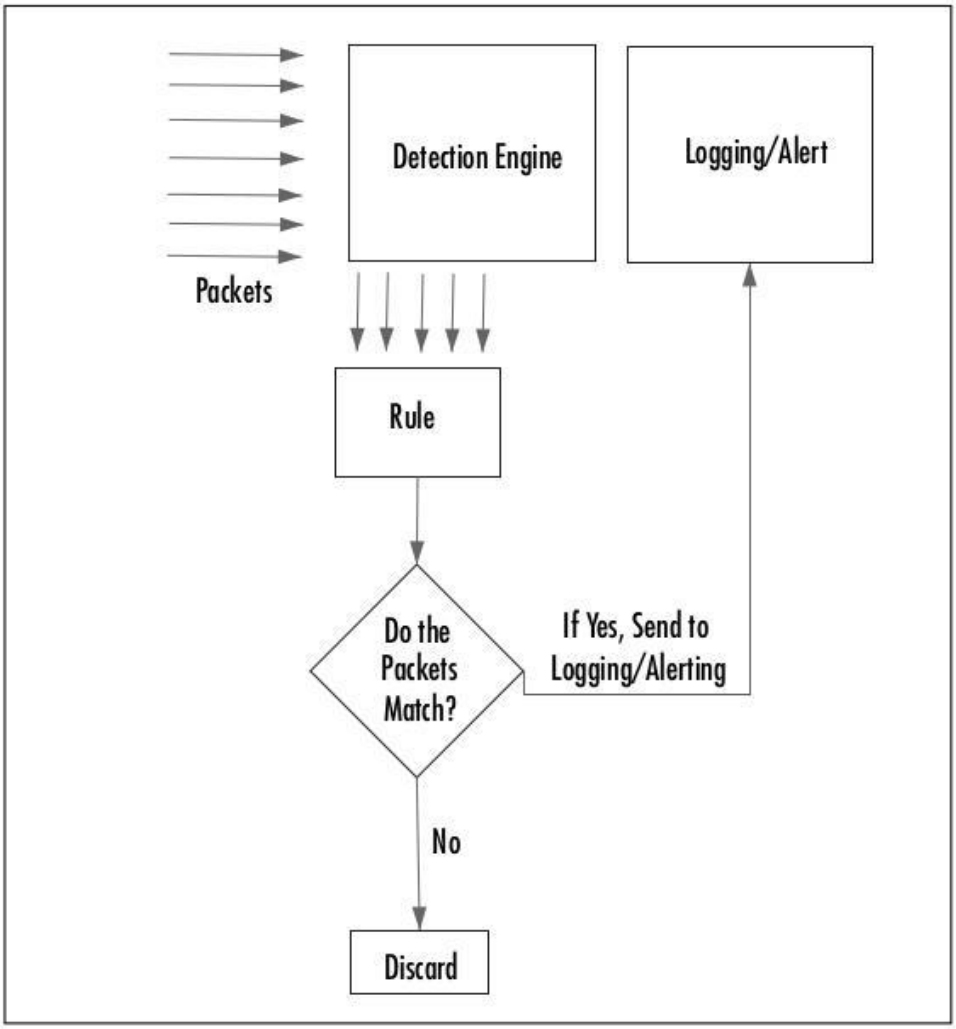
Η Μηχανή Ανίχνευσης (DetectionEngine), παίρνει όλα τα δεδομένα που έρχονται από τον Προεπεξεργαστή και τα Plug-ins του. Αυτά τα δεδομένα τα τσεκάρει βάση ένα σύνολο κανόνων που έχουν γραφτεί για το Snort (Rulesets). Αν κάποιος κανόνας εμπεριέχει δεδομένα ίδια με του πακέτου, τότε το πακέτο στέλνεται στον επεξεργαστή συναγερμών και πυροδοτείτε ένα Alert. Το Snort χρησιμοποιεί αυτούς τους κανόνες-υπογραφές για να κάνει ανίχνευση συγκεκριμένης επίθεσης.

Οι κανόνες αποτελούνται από δύο μέρη:

- Την *κεφαλίδα* του κανόνα, που περιγράφει τον τύπο του πακέτου (TCP, UDP, ICMP κτλ), διευθύνσεις και πόρτες αποστολέα και παραλήπτη.
- Το *περιεχόμενο* του κανόνα, που μπορεί να εμπεριέχεται σε ένα ύποπτο πακέτο.

Όπως καταλαβαίνουμε το Snort είναι ένα σύστημα ανίχνευσης εισβολών βασισμένο σε υπογραφές. Στην περίπτωση μας θα τις αναφέρουμε, ως κανόνες (Rulesets). Οι κανόνες χωρίζονται σε διάφορες κατηγορίες (Trojan horses, buffer overflows, access to various applications) και ενημερώνονται τακτικά.. Κατά την εκκίνηση του Snort οι κανόνες διαβάζονται και τοποθετούνται σε δενδροειδή μορφή. Έτσι τα πακέτα που στέλνονται διαβάζονται από την κεφαλίδα που περιέχει πληροφορίες πρωτοκόλλου κ.τ.λ. Αν για παράδειγμα το πακέτο είναι TCP, τότε η Μηχανή Ανίχνευσης θα επικεντρωθεί στους κανόνες για πακέτα TCP. Στη συνέχεια θα συνεχίσει με πόρτες και διευθύνσεις αποστολέα και παραλήπτη. Αυτό θα κάνει ποιό συγκεκριμένη την αναζήτηση κάποιου κανόνα, που τα περιεχόμενα του θα ταιριάζουν με τα περιεχόμενα του πακέτου. Αν βρει κάποιο κανόνα που να ταιριάζει, τότε το Snort στέλνει τα δεδομένα στην έξοδο σαν Alert. Οι χρήστες του Snort μπορούν να γράψουν τους δικούς τους κανόνες, αλλά περισσότερα θα αναφέρουμε αργότερα.

Στο παρακάτω σχήμα φαίνεται η λειτουργία της Μηχανής Ανίχνευσης με τους κανόνες που περιγράψαμε παραπάνω:



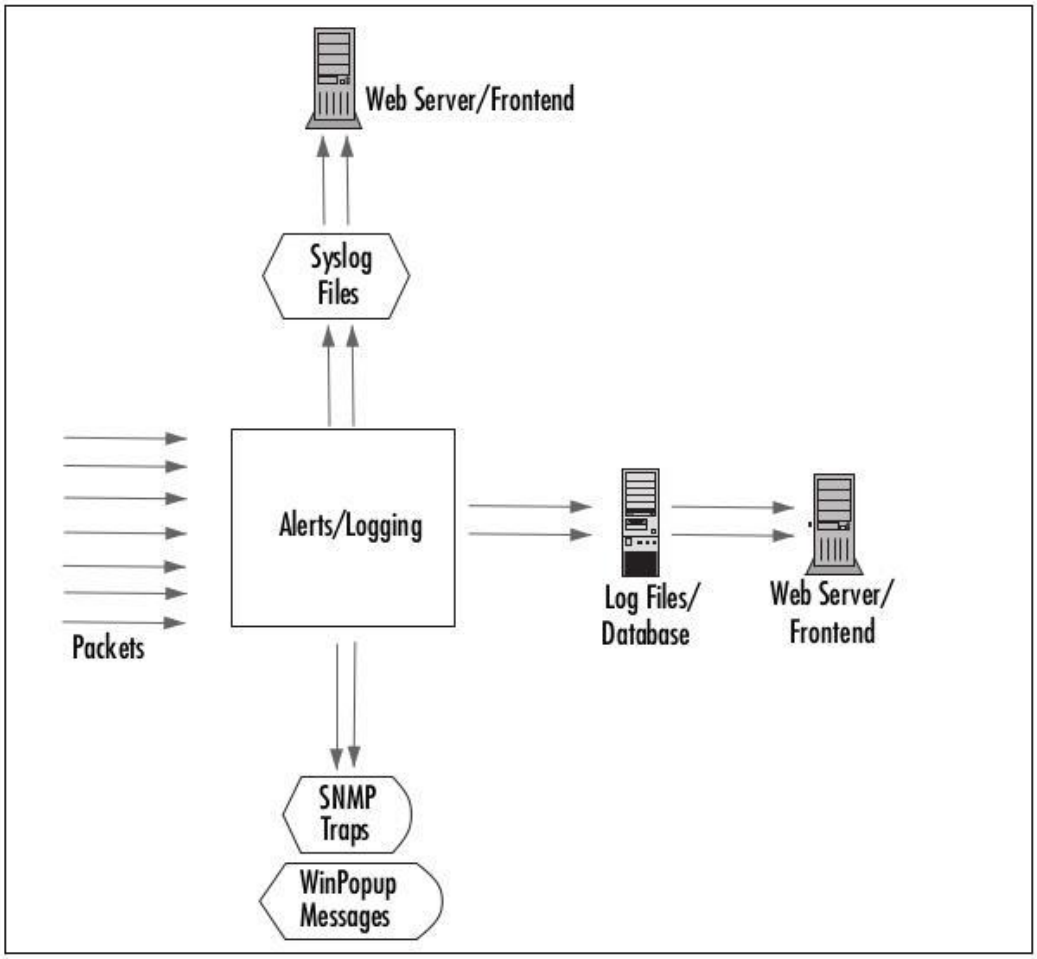
Εικόνα 11: Η Μηχανή Ανίχνευσης του Snort

2.2.4 Έξοδος (Alerting/Logging component)

Αφού τα πακέτα ελεγχτούν από τη Μηχανή Ανίχνευσης, άλλα απορρίπτονται και άλλα στέλνονται στην έξοδο. Αυτά που στέλνονται στην έξοδο ταιριάζουν με κάποιο κανόνα. Έτσι πυροδοτούνται συνεργισμοί και αποθηκεύονται σε Log αρχεία στο δίσκο ή σε βάση δεδομένων SQL, όπως η MySQL και η Postgres.

Αν θέλετε μπορείτε να χρησιμοποιήσετε διάφορα εργαλεία μαζί με το Snort που περιέχουν Plug-ins για Perl, Php και Web εξυπηρετητές, ώστε να δείτε αυτά τα Alerts σε Web διεπαφή. Τα Logs αποθηκεύονται σε αρχεία κειμένου (στο path /var/log/snort) ή σε μία βάση MySQL και Postgres.

Όπως ο Προεπεξεργαστής και η Μηχανή Ανίχνευσης χρησιμοποιούν διάφορα Plug-ins.Έτσι και το alert component χρησιμοποιεί Plug-ins, για να αποθηκεύσει τα alerts σε μία βάση. Επίσης δικτυακά πρωτόκολλα όπως SNMP traps και Win Popup messages. Στο παρακάτω σχήμα φαίνετε ακριβός η λειτουργία του.



Εικόνα 12: Λειτουργία της εξόδου των Alerts

Ο παρακάτω πίνακας περιέχει αρκετά χρήσιμα third-party προγράμματα και εργαλεία που μας βοηθούν να κάνουμε καλύτερη ανάλυση των συναγερμών. Στην παρούσα εργασία θα περιγράψουμε το BASE.

Output Viewer	URL	Περιγραφή
SnortSnarf	www.silicondefense.com/software/snortsnarf	Ένας Snort αναλυτής από Silicon Defense που χρησιμοποιείται για αναγνώριση. Η έξοδος είναι σε HTML.
Snortplot.php	www.snort.org/dl/contrib/data_analysis/snortplot.pl	Ένα Perlscript όπου γραφικά σχεδιάζει τις επιθέσεις.
Swatch	http://swatch.sourceforge.net	Παρέχει σε πραγματικό χρόνο alerts μέσω E-mail.
ACID	http://acidlab.sourceforge.net	Παρέχει logging ανάλυση για το Snort. Απαιτεί PHP, Apache, και το Snort database plug-in.
BASE	http://sourceforge.net/projects/secureideas/	Το Basic Analysis and Security Engine είναι ο καλύτερος τρόπος για να κάνουμε ανάλυση στα Snort alerts.
Loghog	http://sourceforge.net/projects/loghog	Αναλύει τα log αρχεία και ενημερώνει με Email τον διαχειριστή. Επίσης μπλοκάρει τους χρήστες ρυθμίζοντας τους IPTables κανόνες.
Oinkmaster	www.algonet.se/~nitzer/oinkmaster	Χρησιμοποιείται για να κρατάει τους κανόνες ενημερωμένους.
SnortReport	www.circuitsmaximus.com/download.html	Εργαλείο που ενημερώνει σε πραγματικό χρόνο για τους συναγερμους

2.3 Εγκατάσταση του Snort

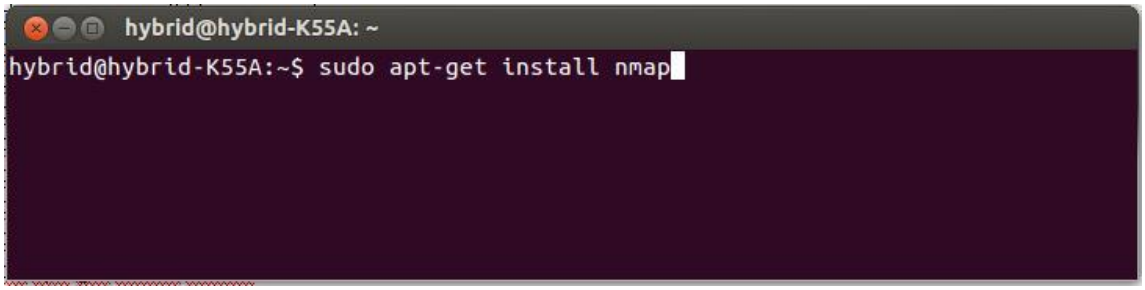
Στην παρούσα εργασία εγκαταστήσαμε το Snort 2.9.4 σε λειτουργικό σύστημα Linux – Ubuntu 12.10 64bit.. Το Snort μπορούμε να το εγκαταστήσουμε εκτός από συστήματα UNIX και σε συστήματα Windows. Αλλά για λόγους ασφάλειας επιλέξαμε σύστημα UNIX. Κάποιος μπορεί να βρει οδηγούς εγκατάστασης κυρίως για συστήματα UNIX στον σύνδεσμο https://www.snort.org/docs ενώ για Windows θα πρέπει να επισκεφτείτε τον σύνδεσμο http://winsnort.com/ .

2.3.1 Απαιτούμενο λογισμικό

Πριν ξεκινήσουμε να εγκαθιστούμε το Snort υπάρχουν κάποια πακέτα που είναι απαραίτητα για την εγκατάσταση του. Αυτά είναι τα παρακάτω:

- **Libpcap**: Είναι βιβλιοθήκη και περιέχει ένα API, που συλλέγει τα πακέτα μέσα στο δίκτυο για ανάλυση κίνησης.
  http://www.tcpdump.org/
- **PCRE**: Η βιβλιοθήκη PCRE ενσωματώνεται σε μια σειρά από εξέχοντα open-source προγράμματα όπως το Apache HTTP Server, PHP και Snort.
  http://www.pcre.org/
- **Libdnet**: Είναι ένα δικτυακό API που παρέχει πρόσβαση σε διάφορα πρωτόκολλα. http://libdnet.sourceforge.net/
- **Barnyard2**: Είναι ένα σύστημα εξόδου του Snort, που διαβάζει τα αρχεία εξόδου του Snort με τους συναγερμούς (Alerts) και τα αποθηκεύει σε μία βάση δεδομένων. http://www.securixlive.com/barnyard2/download.php
- **DAQ**: Είναι το Data-Acquisition API το οποίο είναι απαραίτητο για την έκδοση Snort 2.9.0 και μετά.
  http://www.snort.org/snort-downloads

Πριν ξεκινήσουμε θα χρειαστούμε κάποια επιπλέων πακέτα. Έτσι ανοίγουμε ένα Τερματικό (Terminal) και εκτελούμε τις παρακάτω εντολές:



Εικόνα13: Τερματικό των Ubuntu 12.10

```
sudo apt-get install nmap
sudo apt-get install nbtscan
sudo apt-get install apache2
sudo apt-get install php5
sudo apt-get install php5-mysql
sudo apt-get install php5-gd
sudo apt-get install libpcap0.8-dev
```

```
sudo apt-get install libpcre3-dev
sudo apt-get install g++
sudo apt-get install bison
sudo apt-get install flex
sudo apt-get install libpcap-ruby
sudo apt-get install make
sudo apt-get install autoconf
sudo apt-get install libtool
```
(Στην επόμενη εγκατάσταση θα ζητηθεί κωδικός για τον χρήστη root της MySQL)
```
sudo apt-get install mysql-server
sudo apt-get install libmysqlclient-dev
```

Καλό θα ήταν να ενημερώσουμε και το λογισμικό μας με τις τελευταίες ενημερώσεις ασφάλειας εκτελώντας τις παρακάτω εντολές στο τερματικό.

```
sudo apt-get install mysql-server
sudo apt-get install libmysqlclient-dev
```

2.3.2 Εγκατάσταση Daq (Data acquisition API)

Η τελευταία έκδοση που χρησιμοποιήσαμε είναι η daq-2.0.0.tar.gz και μπορούμε να την βρούμε εδώ:

https://www.snort.org/snort-downloads

Για να την εγκαταστήσουμε την κατεβάζουμε και εκτελούμε τις παρακάτω εντολές στο φάκελο που έχουμε κατεβάσει το αρχείο με την χρήση ενός τερματικού (Όπου "X" βάζουμε αριθμό έκδοσης):

```
sudo tar zxvf daq-X.X.X.tar.gz
cd daq-X.X.X
sudo ./configure
sudo make
sudo make install
```

2.3.3 Εγκατάσταση Libdnet

Μεταβαίνουμε σε αυτό το σύνδεσμο libdnet.googlecode.com/files/libdnet-1.12.tgz και κατεβάζουμε το πακέτο. Μετά ανοίγουμε ένα τερματικό και εκτελούμε τις παρακάτω εντολές στο φάκελο, που έχουμε κατεβάσει το πακέτο.

```
sudo tar zxvf libdnet-1.12.tgz
cd libdnet-1.12/
sudo ./configure
sudo make
sudo make install
sudo ln -s /usr/local/lib/libdnet.1.0.1 /usr/lib/libdnet.1
```

2.3.4 Εγκατάσταση του Snort

Εμείς χρησιμοποιήσαμε την έκδοση 2.9.4 του Snort. Για να εγκαταστήσουμε το Snort μεταβαίνουμε στο σύνδεσμο http://www.snort.org/snort-downloads και κατεβάζουμε τον πηγαίο κώδικα της τελευταίας έκδοσης σε ένα φάκελο. Ανοίγουμε ένα τερματικό και εκτελούμε τις παρακάτω εντολές (ο φάκελος εγκατάστασης σύμφωνα με τις εντολές θα είναι /usr/local/snort)

```
sudo tar zxvf snort-2.9.3.tar.gz
cd snort-2.9.3
sudo ./configure --prefix=/usr/local/snort --enable
-sourcefire
sudo make
sudo make install
sudo mkdir /var/log/snort
sudo mkdir /var/snort
sudo groupadd snort
sudo useradd -g snort snort
sudo chown snort:snort /var/log/snort
```

2.3.5 Εγκατάσταση Κανόνων (Rules)

Μετά την εγκατάσταση του Snort είναι απαραίτητο να εγκαταστήσουμε τους τελευταίους κανόνες. Που θα τους χρησιμοποιήσει για την ανίχνευση επιθέσεων. Είναι απαραίτητο να ενημερώνουμε αυτούς τους κανόνες τακτικά για ανίχνευση καινούριων επιθέσεων. Μεταβαίνουμε στο σύνδεσμο https://www.snort.org/snort-rules και κατεβάζουμε τους τελευταίους κανόνες. Μετά από το τερματικό στο φάκελο που έχουμε κατεβάσει τους κανόνες, εκτελούμε τις παρακάτω εντολές.

```
sudo tar zxvf snortrules-snapshot-XXXX.tar.gz -C /usr/local/snort
sudo mkdir /usr/local/snort/lib/snort_dynamicrules
sudo cp /usr/local/snort/so_rules/precompiled/Ubuntu-10-4/x86-64/2.9.3.0/* /usr/local/snort/lib/
snort_dynamicrules
sudo touch /usr/local/snort/rules/white_list.rules
sudo touch /usr/local/snort/rules/black_list.rules
```

2.3.6 Εγκατάσταση Barnyard2

Το Barnyard2 είναι απαραίτητο για να διαβάζει τα αρχεία εξόδου του Snort και να τα περνάει σε μία βάση δεδομένων. Εδώ πρέπει να αναφέρουμε ότι το Barnyard2 το χρησιμοποιούμε για να μην απασχολούμε το Snort και με αυτή την διαδικασία και να επικεντρώνεται μόνο στην ανίχνευση.

Αρχικά, κατεβάζουμε την τελευταία έκδοση του Barnyard2 από τον σύνδεσμο https://github.com/firnsy/barnyard2/ σε ένα φάκελο. Αφού έχουμε το αρχείο εγκατάστασης εκτελούμε τις παρακάτω εντολές σε ένα τερματικό.

```
sudo tar zxvf barnyard2-X.X.tar.gz
cd<Barnyard2 folder>
sudo autoreconf -fvi -I ./m4
sudo ./configure --with-mysql --with-mysql-libraries=/usr/lib/x86_64-linux-gnu
sudo make
sudo make install
sudo cp etc/barnyard2.conf /usr/local/snort/etc
sudo mkdir /var/log/barnyard2
sudo chmod 666 /var/log/barnyard2
sudo touch /var/log/snort/barnyard2.waldo
sudo chown snort.snort /var/log/snort/barnyard2.waldo
```

Αφού εγκαταστήσουμε το Barnyard είναι απαραίτητο να φτιάξουμε μία βάση δεδομένων, όπου το Barnyard2 θα περνάει τα δεδομένα εξόδου του Snort, που είναι αποθηκευμένα σε Unified logging αρχεία στο φάκελο /var/log/snort/ του συστήματος μας.

Ανοίγουμε ένα τερματικό και γράφουμε τις παρακάτω εντολές για να δημιουργήσουμε μία βάση με όνομα snort και χρήστη root (θα ζητηθεί κωδικός, ο κωδικός είναι αυτός που δώσαμε νωρίτερα για την MySQL):

```
echo "create database snort;" | mysql -u root -p
mysql -u root -p -D snort < ./schemas/create_mysql
```

Καλό θα ήταν να φτιάξουμε ένα επιπλέων χρήστη για την βάση μας με την παρακάτω εντολή (να θυμάστε τον κωδικό “YOURPASSWORD” που θα δώσετε):

```
echo "grant create, insert, select, delete, update on snort.* to snort@localhost identified by 'YOURPASSWORD'" | mysql -u root –p
```

2.3.7 Απαραίτητες ρυθμίσεις Snort και Barnyard

Αφού εγκαταστήσουμε όλα τα παραπάνω τώρα πρέπει να κάνουμε κάποιες απαραίτητες ρυθμίσεις ώστε να τρέξουμε το Snort. Αρχικά πρέπει να ρυθμίσουμε το Snort και μετά το Barnyard. Η ρύθμιση αυτών γίνεται με την τροποποίηση των Configuration αρχείων για το κάθε ένα.

Για να ρυθμίσουμε το Snort ανοίγουμε το Configuration αρχείο (snort.conf) που βρίσκεται στο φάκελο /usr/local/snort/etc/ σύμφωνα με την εγκατάσταση που περιγράψαμε ποιο πάνω. Έτσι το ανοίγουμε με δικαιώματα διαχειριστή γράφοντας την παρακάτω εντολή σε ένα Τερματικό:

sudo nano /usr/local/snort/etc/snort.conf

Βρίσκουμε τις παρακάτω γραμμές και τις αλλάζουμε από:

var WHITE_LIST_PATH ../rules
var BLACK_LIST_PATH ../rules

Σέ:

var WHITE_LIST_PATH /usr/local/snort/rules
var BLACK_LIST_PATH /usr/local/snort/rules

Αλλάζουμε αυτές τις γραμμές από:

dynamicpreprocessor directory /usr/local/snort/lib/snort_dynamicpreprocessor/
dynamicengine /usr/local/snort/lib/snort_dynamicengine/libsf_engine.so
dynamicdetection directory /usr/local/snort/lib/snort_dynamicrules

Σέ:

dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/
dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
dynamicdetection directory /usr/local/lib/snort_dynamicrules

Κάτω απο αυτή τη γραμμή κώδικα:

#output unified2: filename merged.log, limit 128, nostamp, mpls_event_types, vlan_event_types

Εισάγουμε:

output unified2: filename snort.u2, limit 128

Αποθηκεύουμε το αρχείο και είμαστε έτοιμοι.

Για να ρυθμίσουμε το Barnyard ανοίγουμε το Configuration αρχείο (barnyard2.conf) που βρίσκεται στο φάκελο /usr/local/snort/etc/ σύμφωνα με την εγκατάσταση που περιγράψαμε ποιο πάνω. Έτσι το ανοίγουμε με δικαιώματα διαχειριστή γράφοντας την παρακάτω εντολή σε ένα Τερματικό:

sudo nano /usr/local/snort/etc/barnyard2.conf

Και αλλάζουμε τις παρακάτω γραμμές απο:

config reference_file: /etc/snort/reference.config
config classification_file: /etc/snort/classification.config
config gen_file: /etc/snort/gen-msg.map
config sid_file: /etc/snort/sid-msg.map

#config hostname: thor
#config interface: eth0

#output database: log, mysql, user=root password=test dbname=db host=localhost

Σε:

(Προσοχή στο YOURPASSWORD βάζουμε τον κωδικό που δώσαμε όταν φτιάξαμε καινούριο χρήστη για την βάση μας):

config reference_file: /usr/local/snort/etc/reference.config
config classification_file: /usr/local/snort/etc/classification.config
config gen_file: /usr/local/snort/etc/gen-msg.map
config sid_file: /usr/local/snort/etc/sid-msg.map

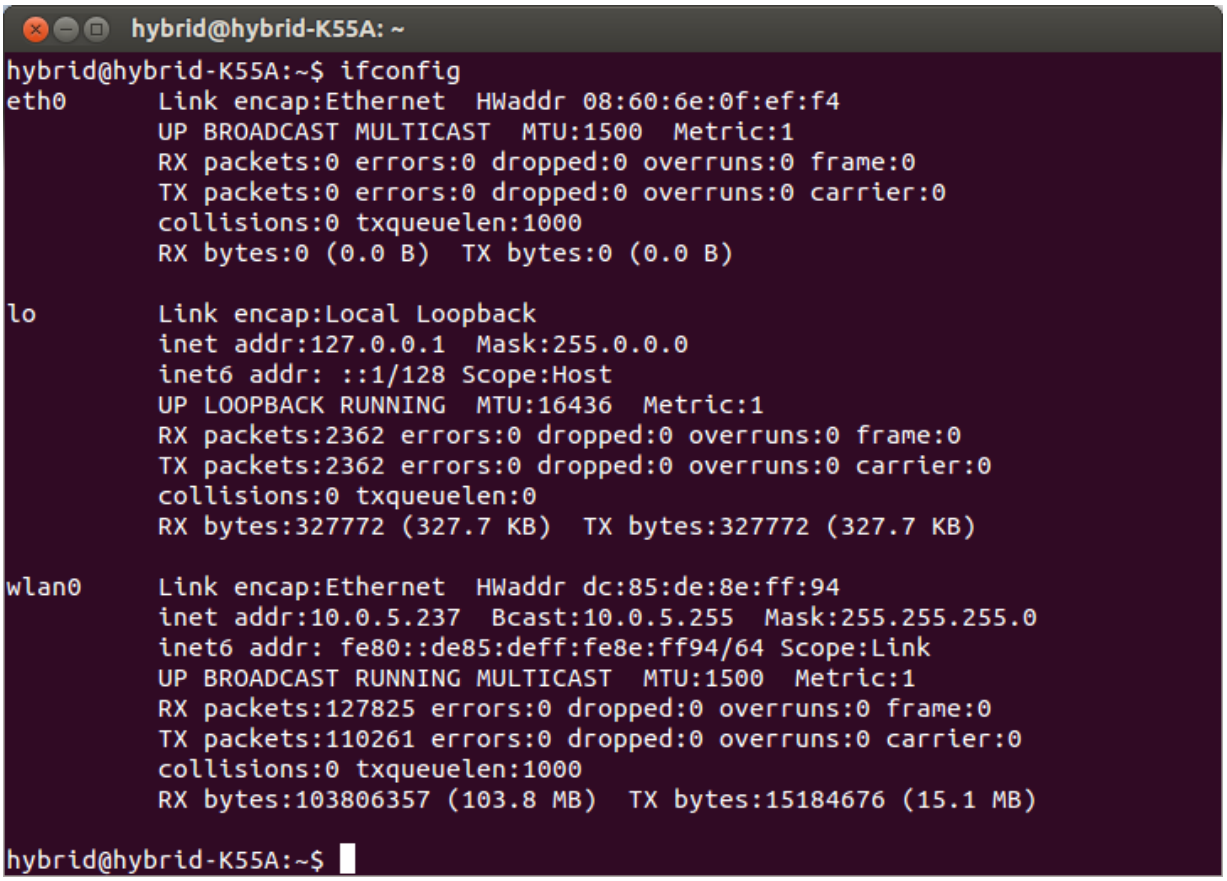
config hostname: localhost
config interface: eth1

output database: log, mysql, user=snort password=YOURPASSWORD dbname=snort host=localhost

Αυτές είναι βασικές ρυθμίσεις που κάναμε για να τρέξουμε το Snort. Απλά προσδιορίσαμε κάποια path για ορισμένα αρχεία που χρησιμοποιεί το Snort και το Barnyard. Επίσης το είδος του αρχείου εξόδου που θα παράγει το Snort και τα στοιχεία της βάσης, που θα αποθηκεύει τους συναγερμούς το Barnyard.

2.3.8 TestSnort

Εφόσον έχουμε κάνει τις απαραίτητες ρυθμίσεις μπορούμε να τεστάρουμε αν τρέχει το Snort. Αρχικά, πρέπει να επιλέξουμε ποιά κάρτα δικτύου θα χρησιμοποιήσουμε για να πιάσουμε πακέτα. Αυτό γίνεται ανοίγοντας ένα τερματικό και γράφοντας την εντολή ifconfig:

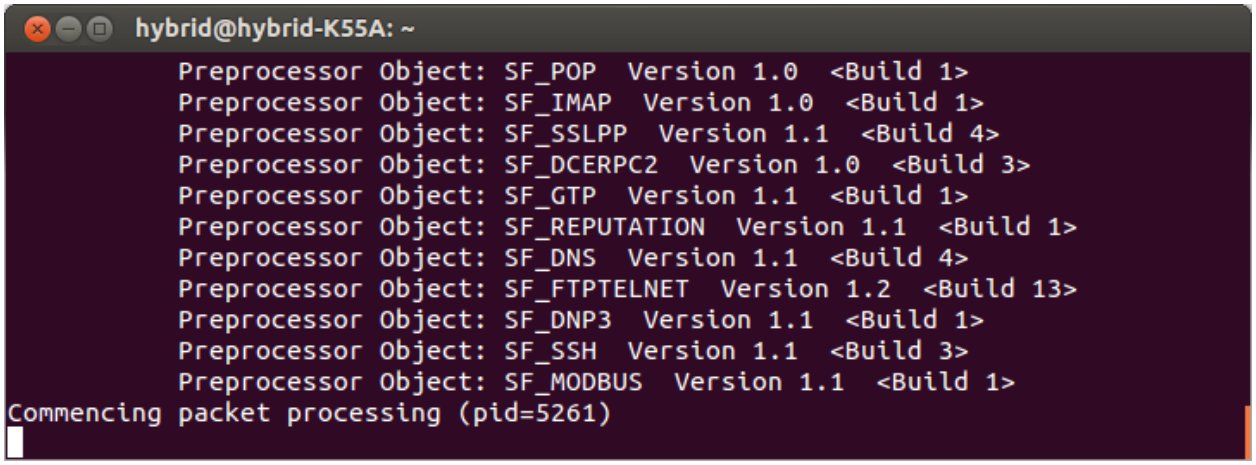


Εικόνα 14: Ενεργές κάρτες δικτύου στο σύστημα μας

Σε αυτό το παράδειγμα θα χρησιμοποιήσω την wlan0, που είναι η ασύρματη κάρτα δικτύου. Έτσι στο τερματικό αν γράψουμε την παρακάτω εντολή θα ξεκινήσει το Snort,χρησιμοποιώντας το αρχείο ρυθμίσεωνsnort.conf και την ασύρματη κάρτα για να ελέγχει την κίνηση:

sudo /usr/local/snort/bin/snort -c /usr/local/snort/etc/snort.conf –i wlan0

Αν στην οθόνη λάβουμε το μήνυμα “Commencingpacketprocessing” τότε η εγκατάσταση και οι ρυθμίσεις μας είναι σωστές και μπορούμε να συνεχίσουμε.



Εικόνα 15: Το Snort λειτουργεί και ελέγχει τα πακέτα

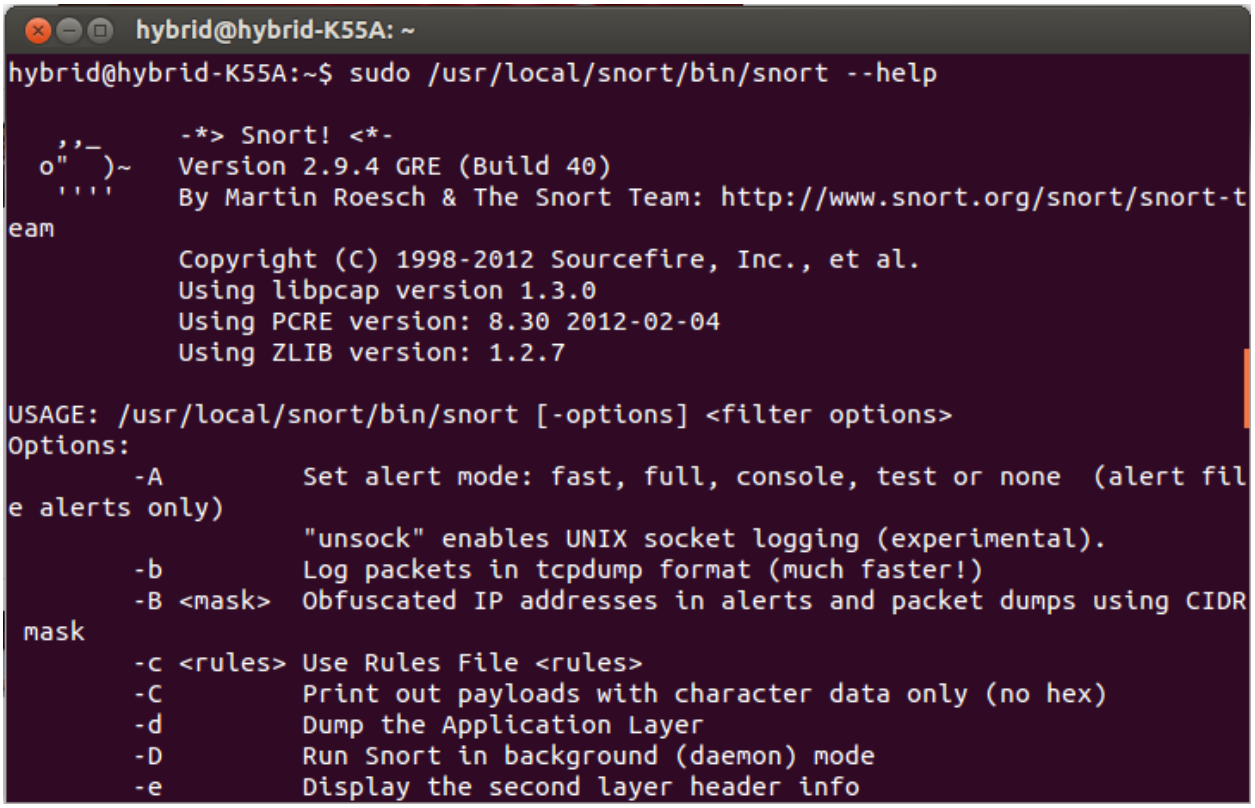
2.3.9 Παράμετροι του Snort

Τρέχοντας το Snort από command –line έχει αρκετές λειτουργίες όπου μπορούμε να τις εκτελέσουμε με διάφορες παραμέτρους που υπάρχουν και θα περιγράψουμε. Με τις παραμέτρους μπορούμε να ελέγξουμε τα πάντα από logging, alerts και scan modes μέχρι δικτυακές επιλογές και ρυθμίσεις συστήματος. Επίσης οι παράμετροι υπερτερούν από τις ρυθμίσεις του αρχείου ρυθμίσεων snort.conf.

Μπορούμε να δούμε την λίστα ,με τις διάφορες παραμέτρους γράφοντας σε ένα τερματικό την εντολή:

sudo /usr/local/snort/bin/snort –help

Θα μας εμφανίσει στην Consola αρκετές πληροφορίες



Εικόνα 16: Εμφάνιση παραμέτρων του Snort στο Τερματικό

–A<alert>. Με αυτή την παράμετρο επιλέγουμε τον τρόπο καταγραφής των Alerts σε full, fast, console ή τίποτα. Η επιλογή full δίνει τους συναγερμούς στο αρχείο συναγερμών. Η επιλογή fast γράφει στο αρχείο συναγερμών μόνο το timestamp, το μήνυμα, τις IP και τις πόρτες των πακέτων. Η επιλογή console εμφανίζει στην οθόνη τα Alerts. Η κενή επιλογή δεν δείχνει και γράφει τα Alerts πουθενά.

–b. Καταγράφει τα πακέτα σε tcpdump format. Όλα τα πακέτα καταγράφονται στην γονική τους δυαδική κατάσταση σε ένα αρχείο καταγραφής.

-B<mask>. Κωδικοποιεί τις διευθύνσεις IP των Alerts και των πακέτων καταγραφής. Χρησιμοποιεί την μάσκα των διευθύνσεων. Είναι χρήσιμο όταν θέλουμε να αποκρύψουμε τις διευθύνσεις.

–c<rules>. Γράφουμε το path του Configuration file για το Snort, ώστε όταν τρέξουμε το Snort να τρέξει με τις κατάλληλες ρυθμίσεις.

–C. Απορίπτει τους ASCII χαρακτήρες στα περιεχόμενα (data) των πακέτων.

–d. Απορρίπτει τα δεδομένα του επιπέδου εφαρμογής (Application layer).

–D.Τρέχει το Snort σαν δαίμονα. Οι συναγερμοί στέλνονται στο /var/log/snort/alert. Αν είμαστε σίγουροι ότι δεν υπάρχει κάποιο σφάλμα κατά την εκκίνηση του Snort καλό είναι να το ξεκινάμε έτσι.

–e. Δείχνει/καταγράφει τα δεδομένα της κεφαλής του πακέτου σε Ethernet layer

–F<bpf>. Διαβάζει BPF φίλτρα από το **<bpf>** αρχείο.

–g<gname>. Τρέχει το snort σαν groupID **<gname>** μετά την προετοιμασία. Είναι σαν μέτρο ασφάλειας γιατί μετά την προετοιμασία του πετάει τα δικαιώματα του Root..

–h<hn>. Μπορούμε να προσδιορίσουμε το ποιούς Hosts θέλουμε να προστατέψουμε.

–i<if>. Προσδιορίζουμε το όνομα της κάρτας δικτύου όπου θα χρησιμοποιήσουμε για να πιάσουμε κίνηση**<if>.**

–I. Εισάγει το όνομα της κάρτας δικτύου που χρησιμοποιείτε στα output του snort.

–k <checksum mode>. Ορισμός **<checksum mode>** σε all, noip, notcp,noudp, noicmp, or none. Ρυθμίζουμε το Snort μην κάνει έλεγχο του checksum των πακέτων γιατί αυτή την λειτουργία μπορεί να την κάνει και ένα firewall. Έτσι γλιτώνουμε χρήσιμο χρόνο στην επεξεργασία των πακέτων.

–K <**mode**>. Ρύθμιση καταγραφής πακέτων. Η προεπιλεγμένη είναι σε κατάσταση pcap, άλλες ρυθμίσεις είναι η ASCII και η NONE.

–l <ld>. Ορίζουμε τον κατάλογο όπου θα αποθηκεύονται τα αρχεία καταγραφής. Αν δε χρησιμοποιήσουμε αυτή την παράμετρο το Snort θα χρησιμοποιήσει τον κατάλογο ως προεπιλογή /var/log/snort.

–L <file>. Καταγράφει τα αποτελέσματα σε αρχείο τύπου tcpdump που θα ορίσουμε εμείς.

–m <umask>. Set the umask for all of Snort's output files to the indicated mask.

–M. Καταγράφει μυνήματα, όχι Alerts σε Syslog.

–n <cnt>. Έξοδος αφού ληφθούν <cnt> πακέτα.

–N. Απενεργοποίηση καταγραφής. Alerts συνεχίζουν να λειτουργούν κανονικά.

–o. Αλλαγή της σειράς με την οποία εφαρμόζονται οι κανόνες στα πακέτα. Για παράδειγμα αντί να ορίσουμε να εφαρμόζονται τα είδη κανόνων με την σειρά Alert | Pass | Log, μπορούμε να αλλάξουμε την σειρά εφαρμογής τους σε Pass | Alert | Log.

–O. Αυτή η παράμετρος αποκρύπτει τις διευθύνσεις IP από ta alerts που εμφανίζονται στην οθόνη ή στα αρχεία καταγραφής να εμφανίζονται με μορφή xxx.xxx.xxx.xxx

–p. Απενεργοποιεί την λειτουργία της κάρτας δικτύου από κατάσταση καταγραφής όλων τον πακέτων μέσα στο δίκτυο (promiscuous). Πλέον καταγράφει μονο τα πακέτα προς και από το συστημά μας.

–q. Αυτή η παράμετρος τρέχει το snort και χωρίς να δείχνει τα λογοτυπα και τα reports στο τέλος.

–r <tf>.Με αυτή την παράμετρο μπορουμέ να προσδιορίσουμε ένα tcpdump αρχείο που έχουμε καταγράψει και να το ανοίξουμε να το δούμε με το Snort.

–R <id>. Περιλαμβάνει το **<id>** μέσα στο snort_intf**<id>.**pid αρχείο.Αυτό είναι χρήσιμο όταν το χρησιμοποιούμε ταυτόχρονα διαφορετικές κάρτες δικτύου.

–s. Καταγράφει μυνήματα από Alerts στο syslog. On Linux boxes, they will appear in/var/log/secure; /var/log/messages on many other platforms.You can change the logging facility by using the syslog output plug-in, at which point you should not use the –s switch (command-line alert/log switches override any config file output variables).

–S <n=v>. Αν στους κανόνες που χρησιμοποιούμε για το Snort χρησιμοποιούμε μία μεταβλητή n μπορούμε να την ορίσουμε με αυτή την παράμετρο να είναι ιση με v. Για παράδειγμα θα μπορούσαμε να ορίσουμε την τιμή της μεταβλητής HOME_NET που περιλαμβάνεται σε αρκετούς κανόνες.

–t <chroot>. Με αυτή την παράμετρο μπορούμε να αλλάξουμε τον ριζικό κατάλογο του Snort στο σύστημά μας.

–T. Αυτή η παράμετρος τρέχει το snort και ελέγχει όλες τις παραμέτρους που έχουμε ορίσει και τις ρυθμίσεις που έχουμε κάνει. Είναι χρήσιμο για έλεγχο πρίν ξεκινήσουμε το Snort σαν δέμονα, όπου σε αυτή την κατάσταση δε γίνεται κανένας έλεγχος για την ομαλή λειτουργεία του Snort.

–u <uname>. Αλλάζει το UID το Snort τρέχει κάτω απο **<uname>** μετά την αρχικοποίηση.

–U. Ενεργοποιεί τις UTC χρονοσφραγίδες.

–v. Εκτυπώνει τις κεφαλές των πακέτων που καταγράφει μέσα στο δίκτυο

–V. Δείχνει τον αριθμό έκδοσης του Snort και μετά κάνει έξοδο.

–w. Καθαρίζει τα 802.11πλαίσια διαχείρισης και ελέγχου.

–X. Καθαρίζει τα αρχικά δεδομένα των πακέτων που ξεκινούν στο επίπεδο ζευξης δεδομένων.

–y. Περιλαμβάνει το έτος στις χροσφραγίδες των πακέτων.

–Z <file>. Ορίζουμε τον performonitor προεπεξεργαστή το path και το όνομα.

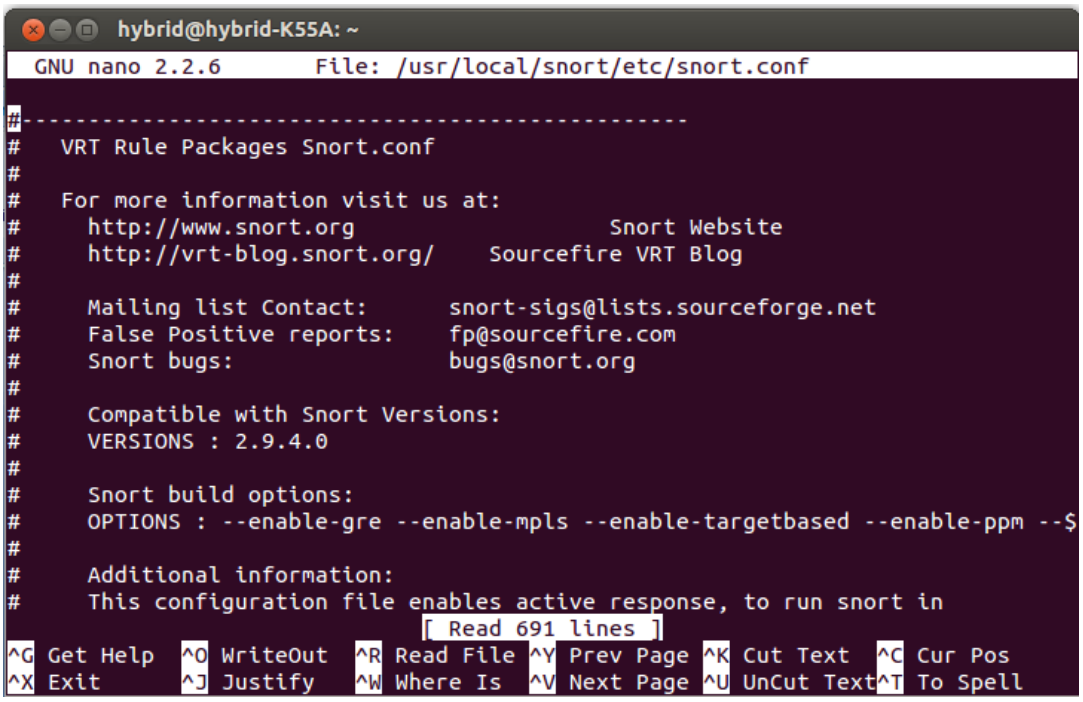
–z. Τρέχει το snort σε λειτουργία διασφάλισης.

–?. Δείχνει τις παραμέτρους που μπορούμε να χρησιμοποιήσουμε και κάνει έξοδο.

2.4 Αρχείο ρυθμήσεων Snort (Configuration file)

Το αρχείο ρυθμίσεων του Snort περιέχει αρκετές χρήσιμες ρυθμίσεις. Οι ρυθμίσεις αυτές βοηθούν στο να κάνουμε ποιό συγκεκριμένη την προστασία σε Hosts και Servers μέσα στο δίκτυο. Το αρχείο αυτό ονομάζεται snort.conf και βρίσκετε στο φάκελο snort/etc/. Στην εγκατάσταση που πραγματοποιήσαμε ποιό πάνω το ακριβές του path είναι το /usr/local/snort/etc/ . Για να ανοίξουμε το αρχείο και να το επεξεργαστούμε πρέπει να έχουμε δικαιώματα διαχειριστή. Έτσι πρέπει να ανοίξουμε ένα τερματικό και μπροστά από την εντολή να γράψουμε sudo. Η εντολή θα είναι:

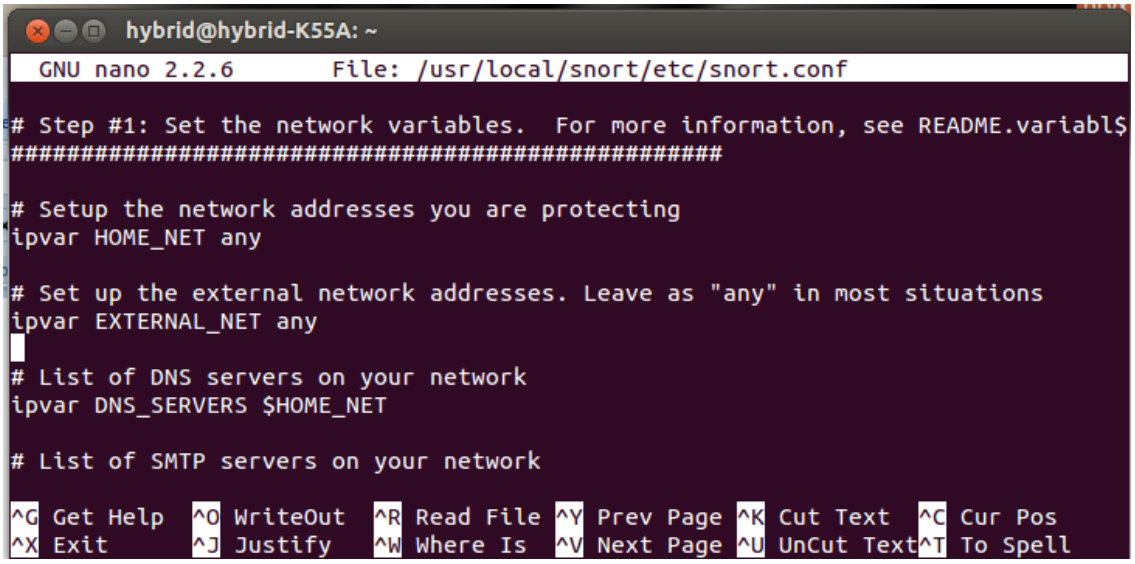
sudo nano /usr/local/snort/etc/snort.conf



Εικόνα 17: Επεξεργασία του configuration file με τον nano

Αφού ανοίξουμε το αρχείο θα δούμε ότι υπάρχουν διάφορα Steps. Αυτά τα steps είναι τα παρακάτω:

1. Ορισμός μεταβλητών του δικτύου.
2. Ρύθμηση του αποκοδικοποιητή
3. Ρύθμηση της base detection engine
4. Ρύθμιση των paths των δυναμικών βιβλιοθηκών
5. Ρύθμηση των προεπεξεργαστών
6. Ρύθμηση των plugin εξόδου (Outputplugins)
7. Ορισμός των αρχείων κανόνων που θα χρησιμοποιήσει το Snort
8. Ορισμός των αρχείων κανόνων προεπεξεργαστή και αποκωδικοποιητή
9. Ορισμός των αρχείων κανόνων Shared Object



Εικόνα 18: Step1 Ορισμός μεταβλητών του δικτύου

Σε αυτό το σημείο θα δούμε πώς μπορούμε να κάνουμε διάφορες ρυθμίσεις που προσδιορίζουν πώς το Snort θα συμπεριφέρεται. Το αρχείο ρυθμίσεων είναι άριστα γραμμένο και πολύ εύκολο στην χρήση του. Για να κάνετε το Snort να λειτουργεί με τον τρόπο που θέλετε, ακλουθήστε τα παρακάτω βήματα:

1. Ρυθμίζουμε την μεταβλητή HOME_NET για το ποιά εσωτερικά δίκτυα θέλουμε να προστατεύσουμε. Αυτή η μεταβλητή χρησιμοποιείτε και απο τους κανόνες. Απο default η μεταβλητή είναι ορισμένη σαν ipvar HOME_NET any και καλύπτει όλα τα εσωτερικά δίκτυα. Αν θέλουμε να προσδιορίσουμε συγκεκριμένα πρέπει να την ορίσουμε κάπως έτσι var HOME_NET 192.168.1.0/24 ή var HOME_NET [192.168.1.0/24,192.168.2.0/24] για δύο δίκτυα.
2. Ρυθμίζουμε την μεταβλητή EXTERNAL_NET για το ποιά δίκτυα ή Hosts είναι ύποπτα προς τα εσωτερικά δίκτυα που προστατεύουμε. Απο Default είναι ορισμένη ipvar EXTERNAL_NET any. Συνιστάτε να είναι ρυθμισμένη για όλα τα δίκτυα προς τα δίκτυα που προστατεύουμε και αυτό γίνετε ορίζοντας την έτσι: ipvar EXTERNAL_NET !$HOME_NET.
3. Μετά ορίζουμε τους Servers που τρέχουν συγκεκριμένες υπηρεσίες. Για παράδειγμα ένα HTTPserver και ένα DNSserver. Αυτοί ασχολούνται με συγκεκριμένες υπηρεσίες και δε μας ενδιαφέρει το Snort να τους εξετάζει για άλλου είδους επιθέσεις. Έτσι για παράδειγμα αν έχουμε ένα WEBserver που τρέχει στο 192.168.1.11 και 192.168.1.12 . Σε αυτή την περίπτωση θα ορίσουμε ipvar HTTP_SERVERS [192.168.1.11/32,192.168.1.12/32]. Απο Default είναι ipvar HTTP_SERVERS $HOME_NET.
4. Ρυθμίζουμε τις πόρτες που χρησιμοποιούν οι υπηρεσίες των Servers. Για παράδειγμα για HTTP πόρτες portvar HTTP_PORTS 80.
5. Ρύθμηση της μεταβλητής RULE_PATH όπου προσδιορίζει πού είναι οι κανόνες που θα χρησιμοποιήσει το snortγια να πυροδοτήσει συναγερμούς. Στην δική μας εγκατάσταση το path είναι /usr/local/snort/rules/ .

6. Στο step 6 κάνουμε κάποιες ρυθμίσεις για τα αρχεία εξόδου που παράγει το Snort τα οποία τα επεξεργαζόμαστε και βλέπουμε τις πυροδοτήσεις. Εμείς έχουμε ορίσει outputunified2: filenamesnort.u2, limit 128. Παράγει ένα αρχείο μορφής unified2 με όνομα snort.u2 χωρίς το timestamp και το μέγιστο μέγεθος κάθε αρχείου να μην ξεπερνάει τα 128 Kbytes. Αν τα ξεπεράσει παράγει άλλο με νέο timestamp. Αυτά τα αρχεία συμφώνα με την εγκατάσταση μας βρίσκονται στο path /var/log/snort/ .
7. Στο step 7 επιλέγουμε ή ορίζουμε ποιά αρχεία κανόνων θα χρησιμοποιήσει το Snort. Σε περίπτωση που φτιάξουμε δικό μας αρχείο με κανόνες πρέπει να δηλώσουμε το path του.

Όταν ξεκινάμε το Snort μπορούμε να προσδιορίσουμε ποιά κάρτα δικτύου θα χρησιμοποιήσει για να συλλέξει πακέτα με την παράμετρο -i. Αν δεν επιλέξουμε αυτόματα θα χρησιμοποιηθεί η πρώτη κάρτα δικτύου απο την λίστα. Χρησιμοποιώντας την παράμετρο -c μπορούμε να προσδιορίσουμε ποιό αρχείο ρυθμίσεων (Configuration file) θα χρησιμοποιήσει το Snort. Μπορούμε να έχουμε διαφορετικά αρχεία ρυθμίσεων και κάθε φορά να χρησιμοποιούμε όποιο επιθυμούμε. Μία άλλη σημαντική παράμετρος είναι η –A, που προσδιορίζει τι τύπου συναγερμούς (Alerts) θα παράγει το Snort.

Με την παρακάτω εντολή ξεκινάμε το Snort και μας δείχνει τους συναγερμούς στην κονσόλα, χρησιμοποιούμε το αρχείο ρυθμίσεων snort.conf που έχουμε στο /usr/local/snort/etc/ και χρησιμοποιούμε την ασύρματη κάρτα δικτύου με όνομα wlan0 για να πιάσουμε πακέτα.

```
sudo /usr/local/snort/bin/snort -A console -c /usr/local/snort/etc/snort.conf -i wlan0
```

Ας δούμε τώρα ένα παράδειγμα που θα μας βοηθήσει να τεστάρουμε την λειτουργία του Snort. Αρχικά θα εισάγουμε κάποιους κανόνες απλούς στο αρχείο κανόνων local.rules. Στην συνέχεια θα ενεργοποιήσουμε αυτούς από το αρχείο ρυθμίσεων και θα τρέξουμε το Snortώστε να δούμε τους συναγερμούς κάνοντας έναping και ανοίγοντας μια ιστοσελίδα.

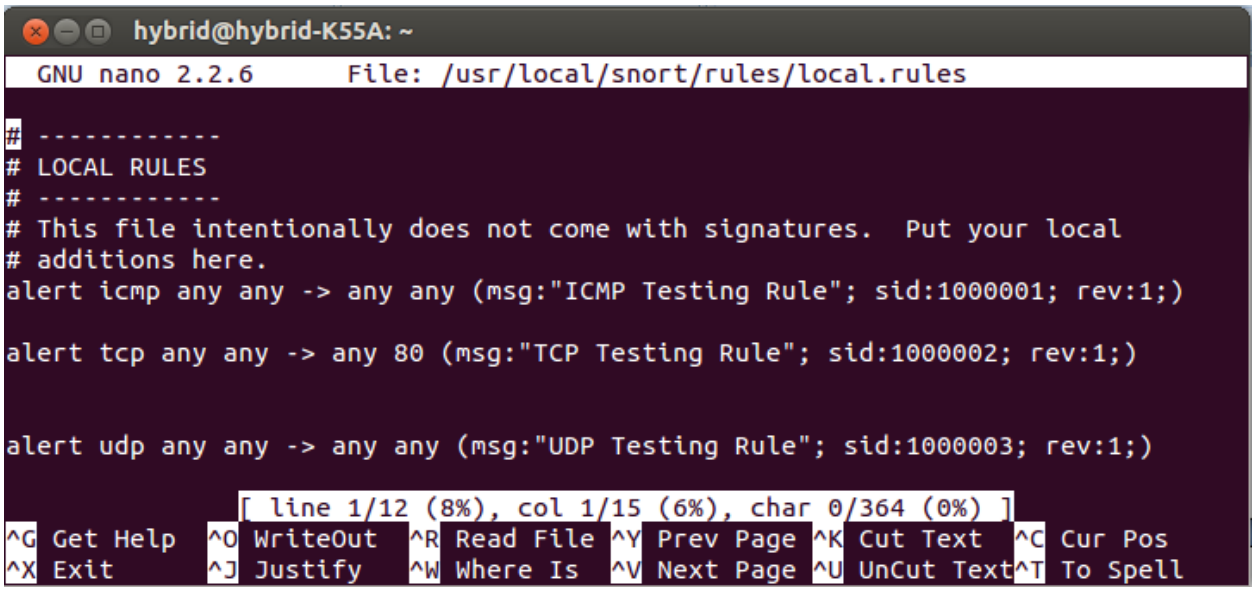
Γράφουμε την εντολή sudo nano /usr/local/snort/rules/local.rules για να ανοίξουμε το αρχείο με τους τοπικούς κανόνες. Στην συνέχεια γράφουμε τους παρακάτω κανόνες:

```
alert icmp any any -> any any (msg:"ICMP Testing Rule"; sid:1000001; rev:1;)

alert tcp any any -> any 80 (msg:"TCP Testing Rule"; sid:1000002; rev:1;)

alert udp any any -> any any (msg:"UDP Testing Rule"; sid:1000003; rev:1;)
```

Ο πρώτος πυροδοτεί συναγερμό για πακέτα ICMP από οπουδήποτε προς οποιονδήποτε Host. Ο δεύτερος εξετάζει τα πακέτα TCP και ο τρίτος τα UDP. Να αναφέρουμε πως θα μιλίσουμε αργότερα για τους κανόνες.



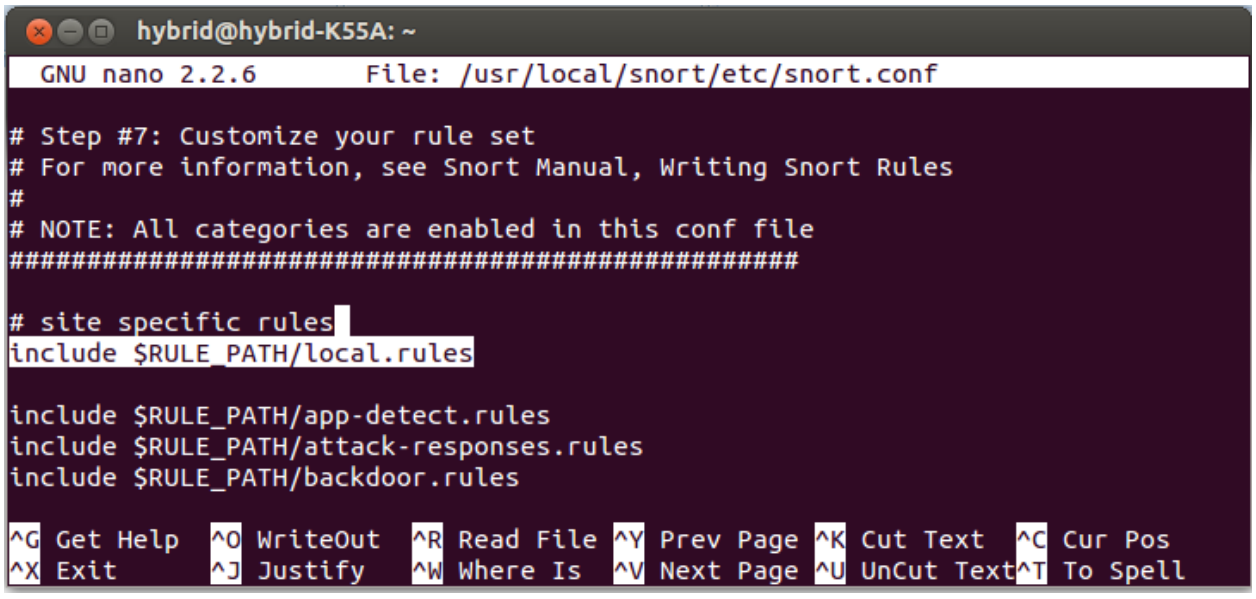
Εικόνα 19: Εισαγωγή κανόνων στο αρχείο κανόνων local.rules

Γράφουμε την εντολή για να ανοίξουμε το αρχείο ρυθμίσεων και να ρυθμίσουμε το Snort να χρησιμοποιεί αυτούς τους κανόνες.

sudo nano /usr/local/snort/etc/snort.conf

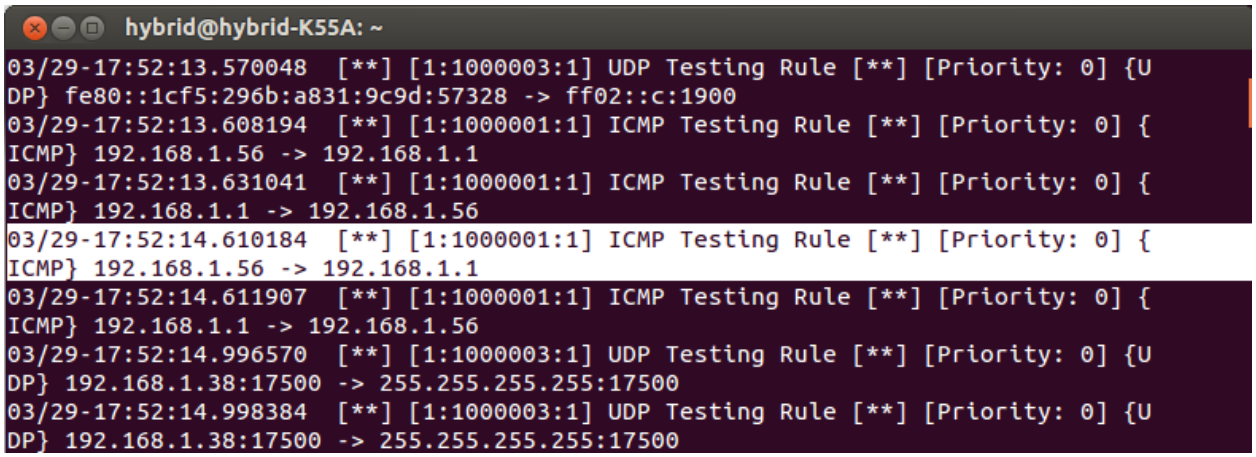
Πηγαίνουμε στο step #7 και προσθέτουμε την γραμμή κώδικα

include $RULE_PATH/local.rules

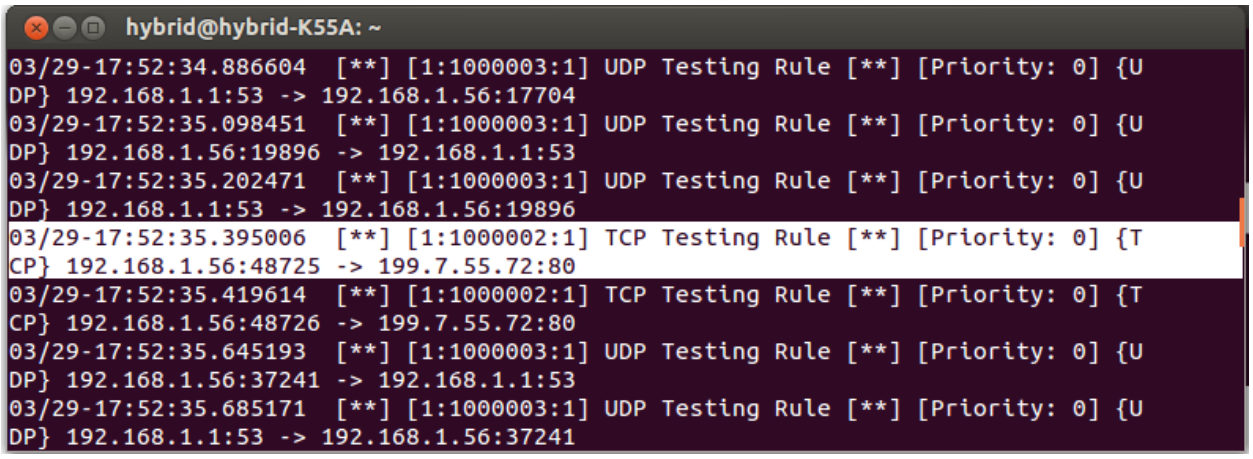


Εικόνα 20: Ρύθμιση του αρχείου ρυθμίσεων να περιλαμβάνει τους local κανόνες

Τώρα αν ξεκινήσουμε το Snort με την εντολή sudo /usr/local/snort/bin/snort -A console -c /usr/local/snort/etc/snort.conf -i wlan0 , κάνουμε ένα ping και ανοίξουμε μια ιστοσελίδα θα δούμε στην κονσόλαAlerts απο ICMP, TCP και UDP πακέτα.



Εικόνα 21: Συναγερμός απο ICMP πακέτο που ανιχνευτικέ απο τον κανόνα που γράψαμε



Εικόνα 22: Συναγερμός απο το TCP πακέτο που πιάστηκε απο τους κανόνες που γράψαμε

2.5 BASE

Το Snort παράγει συναγερμούς και έχει την δυνατότητα να τους καταγράψει σε Log αρχεία στο δίσκο ή να τα δείξει στην κονσόλα. Αλλά αυτό δεν είναι αρκετό για να κάνουμε ανάλυση των διάφορων συναγερμών που θα παράγει. Για αυτό το λόγο χρειαζόμαστε ένα φιλικό με το χρήστη περιβάλλων που θα μπορούμε να επεξεργαστούμε και να οργανώσουμε αυτούς τους συναγερμούς εύκολα. Το BASE (Basic Analysis and Security Engine) είναι μια Web εφαρμογή front-end που υποβάλει ερωτήματα και αναλύει τους συναγερμούς που παράγονται απο το Snort.

2.5.1 Εγκατάσταση BASE

Για να εγκαταστήσουμε το BASE πρέπει να κατεβάσουμε την τελευταία έκδοση του BASE σε ένα φάκελο στο σύστημα μας απόhttp://base.secureideas.net/. Επίσης θα χρειαστούμε την ADOdb μία βιβλιοθήκη απόhttp://sourceforge.net/projects/adodblite/?source=dlp.

Αφού κατεβάσουμε τις εκδόσεις ανοίγουμε ένα τερματικό και μεταβαίνουμε στο φάκελο που έχουμε κατεβάσει τα αρχεία και γράφουμε τις παρακάτω εντολές:

```
sudo tar -xvzf base-1.4.5.tar.gz -C /var/www/
sudo tar -xvzf adodb481.tgz -C /var/www/
```

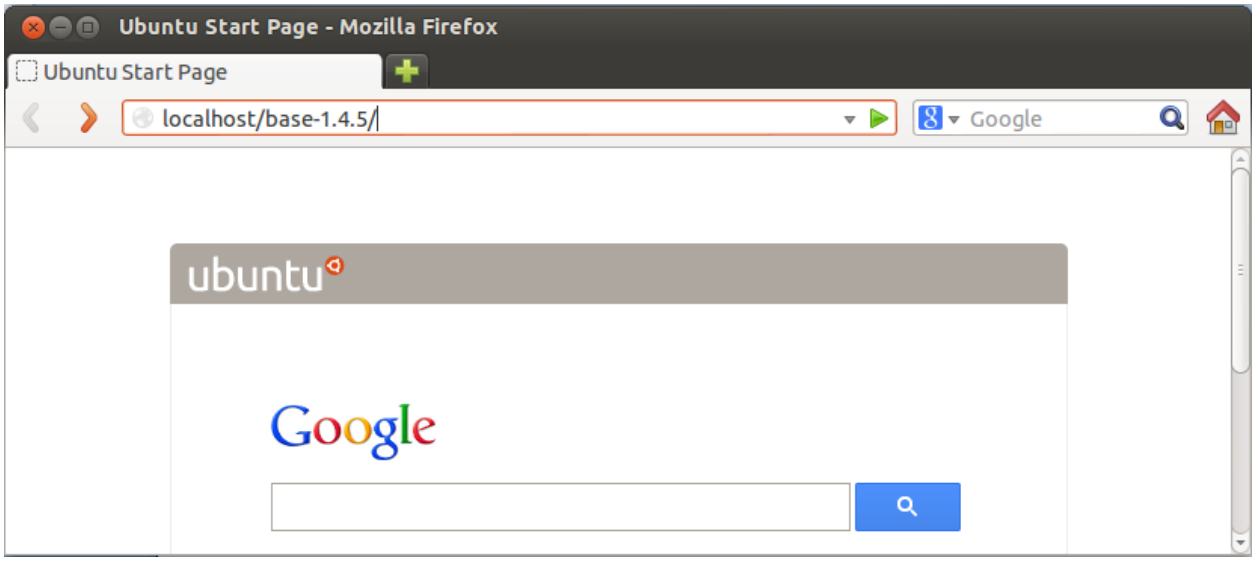
Έτσι βάζουμε τα αρχεία στον Apache server που έχουμε εγκαταστήσει στο σύστημα μας.

Μετά μεταβαίνουμε στο φάκελο και αλλάζουμε τα δικαιώματα του φακέλου του BASE για να μπορέσουμε να κάνουμε την εγκατάσταση.

```
cd /var/www/
sudo chmod 757 base-1.4.5/
```

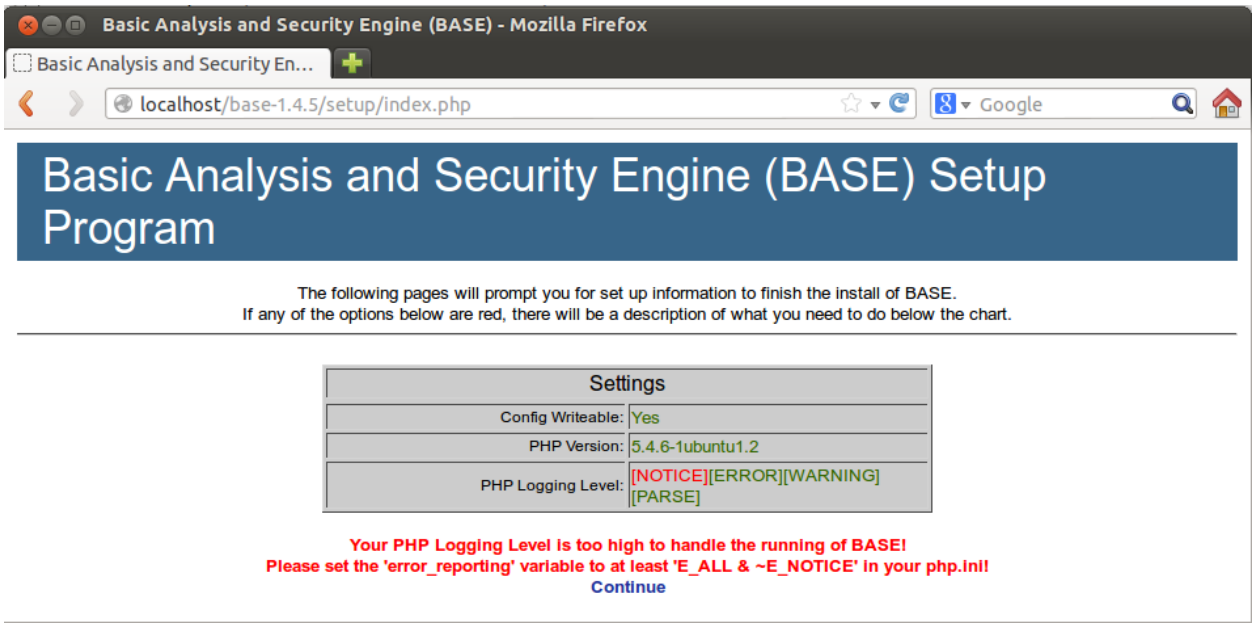
Τώρα είμαστε έτοιμοι να ξεκινήσουμε την εγκατάσταση ανοίγοντας ένα φυλλομετρητή και γράφοντας:

```
localhost/base-1.4.5
```



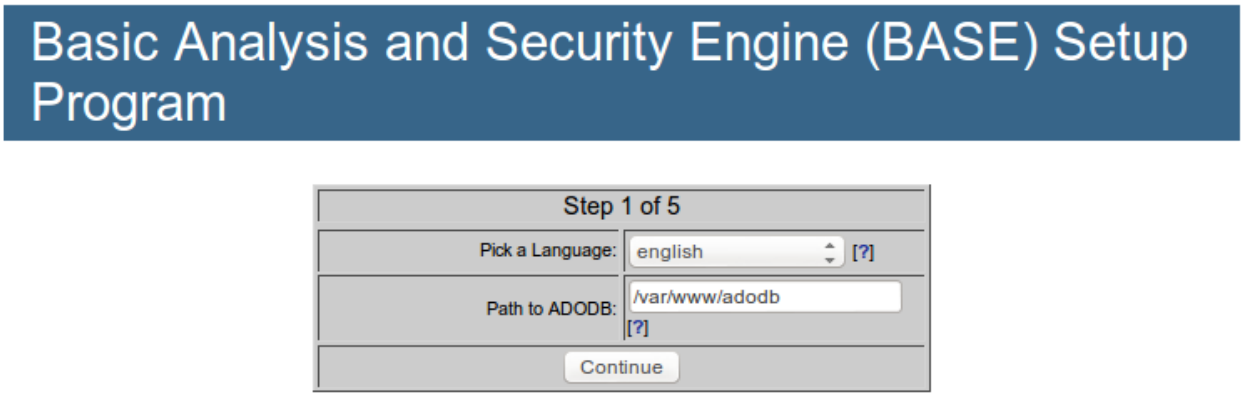
Εικόνα 23: Μετάβαση στη BASE για εγκατάσταση

Αμέσως θα ανοίξει η σελίδα εγκατάστασης και πατάμε στο “Continue”

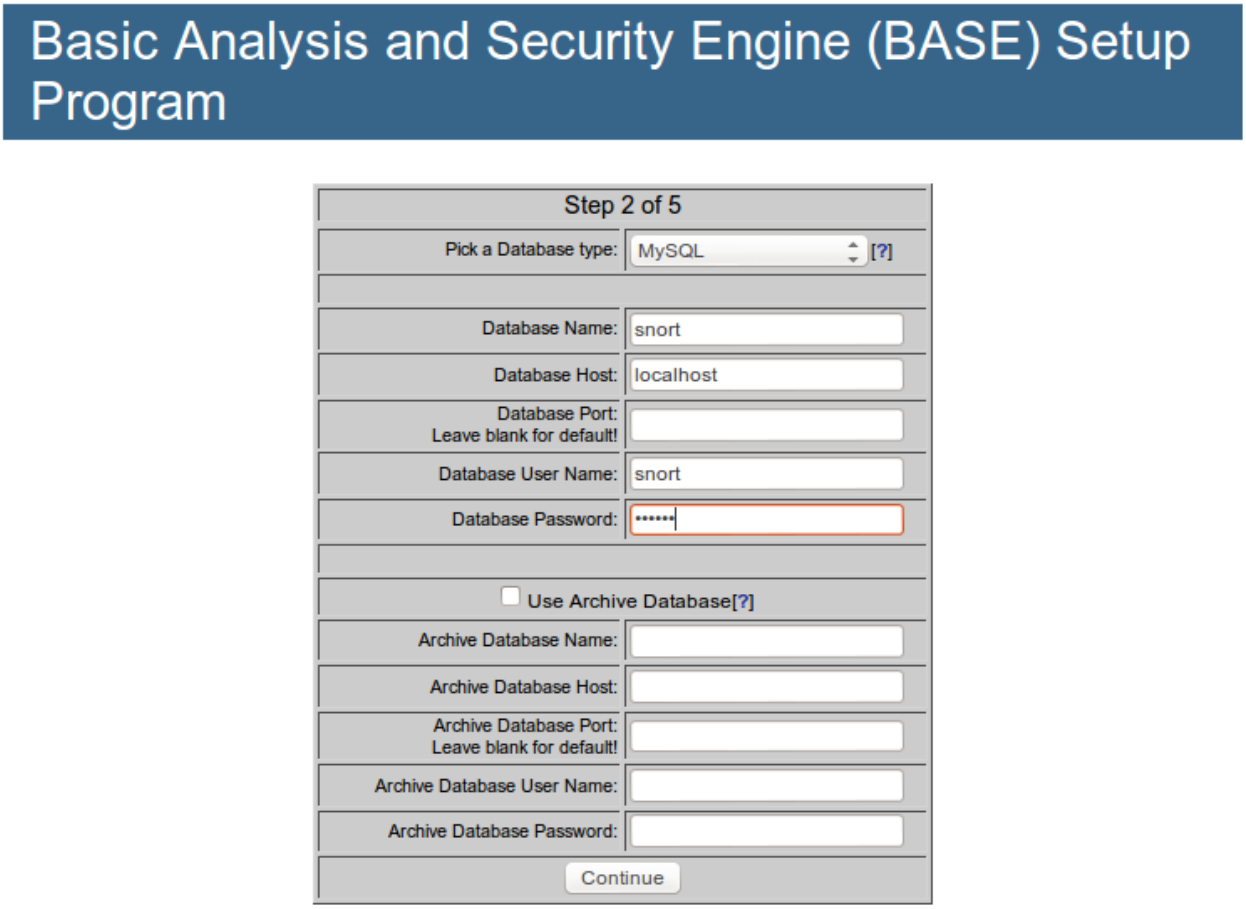


Βήμα 1: Στην επόμενη οθόνη επιλέγουμε γλώσσα και δίνουμε το path για την ADODB και πατάμε "Continue"

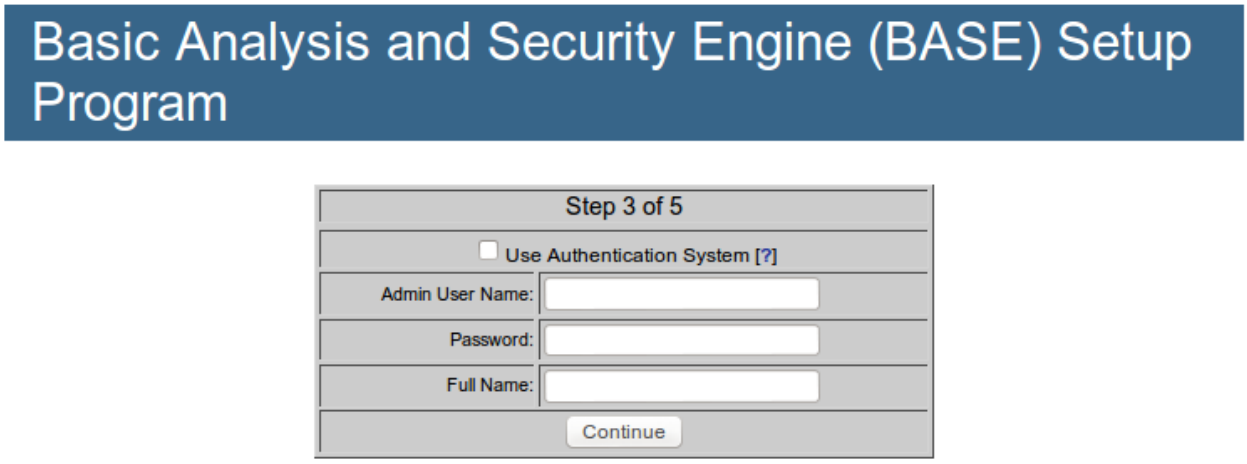
.



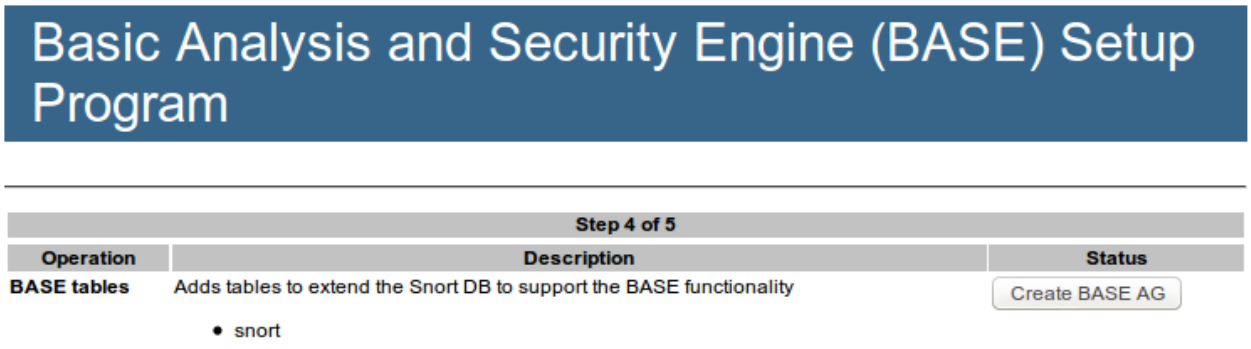
Βήμα 2: Δίνουμε τα στοιχεία της βάσης που έχουμε φτιάξει για το Snort



Βήμα 3: Δίνουμε στοιχεία για Authentication αν θέλουμε να ορίσουμε διαχειριστή.



Βήμα 4: Πατάμε στο "Create BASE AG"



Και μετά στο “Step 5”

Basic Analysis and Security Engine (BASE) Setup Program

Successfully created 'acid_ag'
Successfully created 'acid_ag_alert'
Successfully created 'acid_ip_cache'
Successfully created 'acid_event'
Successfully created 'base_roles'
Successfully INSERTED Admin role
Successfully INSERTED Authenticated User role
Successfully INSERTED Anonymous User role
Successfully INSERTED Alert Group Editor role
Successfully created 'base_users'

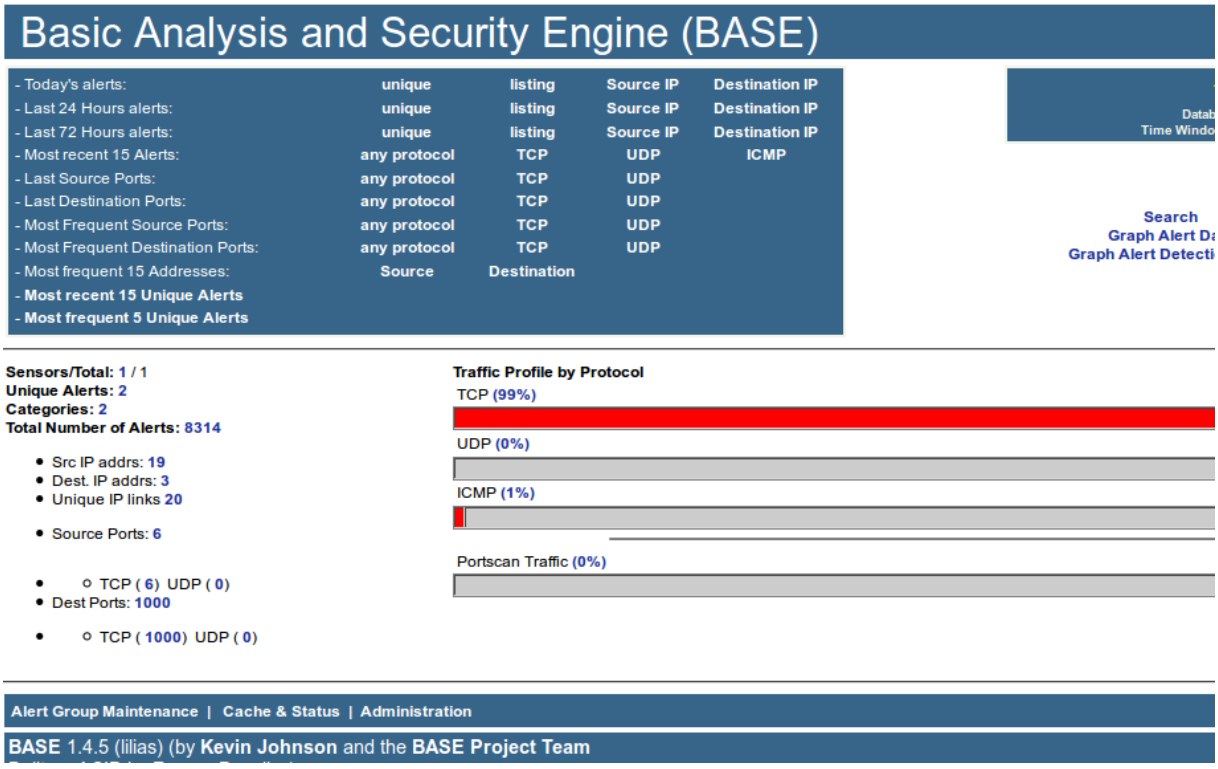
Step 4 of 5		
Operation	**Description**	**Status**
BASE tables	Adds tables to extend the Snort DB to support the BASE functionality • snort	DONE

The underlying Alert DB is configured for usage with BASE.

Additional DB permissions
In order to support Alert purging (the selective ability to permanently delete alerts from the database) and DNS/whois lookup caching, the DB user "snort" must have the DELETE and UPDATE privilege on the database "snort@localhost"

Now continue to **step 5**...

Αυτό ήταν τώρα μπορούμε να χρησιμοποιήσουμε την BASE για να επεξεργαστούμε και να αναλύσουμε τα Alerts μας.



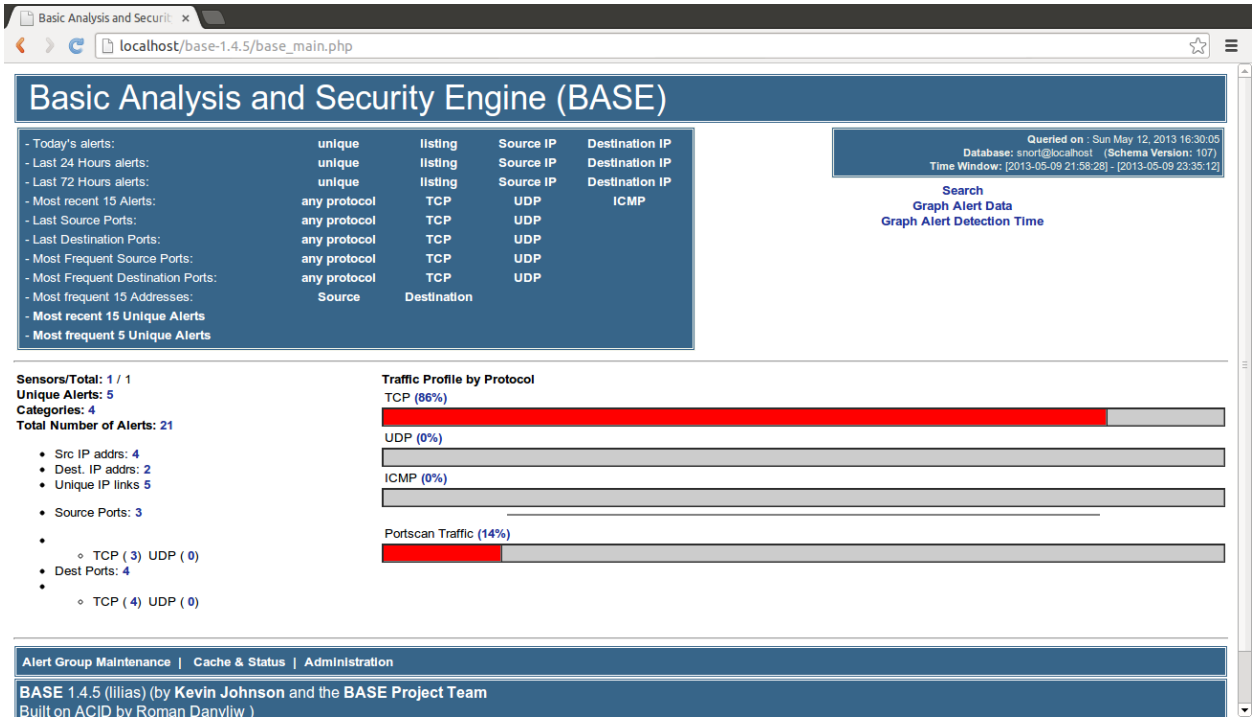
Προσοχή, όπως αναφερθήκαμε στην εγκατάσταση του Snort για να μεταφερθούν οι πληροφορίες από τα Log αρχεία στην βάση δεδομένων που φτιάξαμε πρέπει να τρέχουμε το Barnyard. Αν δεν βλέπεται τίποτα στην BASE πρέπει να τρέξετε το Barnyard. Χρησιμοποιούμε το Barnyard αντί του Snort για να το απαλλάξουμε από αυτή την διαδικασία και το Snort να ασχολείται μόνο με την κίνηση στο δίκτυο. Το Barnyardξεκινάει με την παρακάτω εντολή (τα paths είναι καθορισμένα με την εγκατάσταση που κάναμε στην παρούσα εργασία):

sudo /usr/local/bin/barnyard2 -c /usr/local/snort/etc/barnyard2.conf –G /usr/local/snort/etc /genmsg.map -S /usr/local/snort/etc/sid-msg.map -d /var/log/snort -f snort.u2 –w /var/log/snort/barnyard2.waldo

2.5.2 Λειτουργία του BASE

Η αρχική σελίδα του BASE, όπως φαίνεται στην παρακάτω εικόνα, παρέχει γρήγορη πρόσβαση σε πολλές διαφορετικές και ενδιαφέρουσες πληροφορίες γύρω από τα Alertsπου παράγει το Snort.

- Κάτω δεξιά της σελίδας, παρουσιάζεται ποιο ποσοστό από το σύνολο των ληφθέντων πακέτων ανήκει σε κάθε ένα από τα 3 πρωτόκολλα TCP, UDP, ICMP. Ειδική κατηγορία αποτελεί το PortscanTrafic, όπου εκεί δίνεται το ποσοστό των πακέτων που ανεξαρτήτως πρωτοκόλλου αποτελούσαν απόπειρα σαρώσεις πολλών θυρών ενός μηχανήματος.
- Πάνω αριστερά, υπάρχουν σύνδεσμοι για τις λίστες καταγραφής των πακέτων που ελήφθησαν τις τελευταίες 24 ή 72 ώρες, τις ποιο συχνές διευθύνσεις IPπροέλευσης ή προορισμού που έχουν εμφανιστεί, για τα 5 ποιο συχνά είδη ειδοποιήσεων, τις ποιο συχνά θύρες προορισμού ή προέλευσης και για πολλές ακόμα κατηγοριοποιήσεις των παραπάνω στοιχείων.
- Κάτω αριστερά, δίνονται ο αριθμός των ειδοποιήσεων, των IPπροέλευσης και προορισμού, των θυρών προέλευσης και προορισμού καθώς και σύνδεσμοι για την απευθείας προσπέλαση αυτών των δεδομένων.
- Ο σύνδεσμος “GraphAlertData” πάνω και δεξιά στην παρακάτω εικόνα οδηγεί σε μια άλλη διεπαφή του BASEαπό την οποία ο χρήστης μπορεί να εξάγει εύκολα συγκριτικά διαγράμματα από συγκεκριμένες χρονικές περιόδους. Ενδεικτικά, μερικά από τα δυνατά διαγράμματα είναι η κατανομή του αριθμού των ειδοποιήσεων ανα ώρα, ημέρα ή μήνα, ο αριθμός ειδοποιήσεων για κάθε μία διεύθυνση IPπροέλευσης ή προορισμού και ο αριθμός ειδοποιήσεων για κάθε χώρα προέλευσης ή προορισμού σε παγκόσμιο χάρτη και σε απλό διάγραμμα με μπάρες. Αξιοσημείωτη είναι και η δυνατότητα επιλογής της χρονικής περιόδου από την οποία θα εξαχθούν τα γραφήματα.

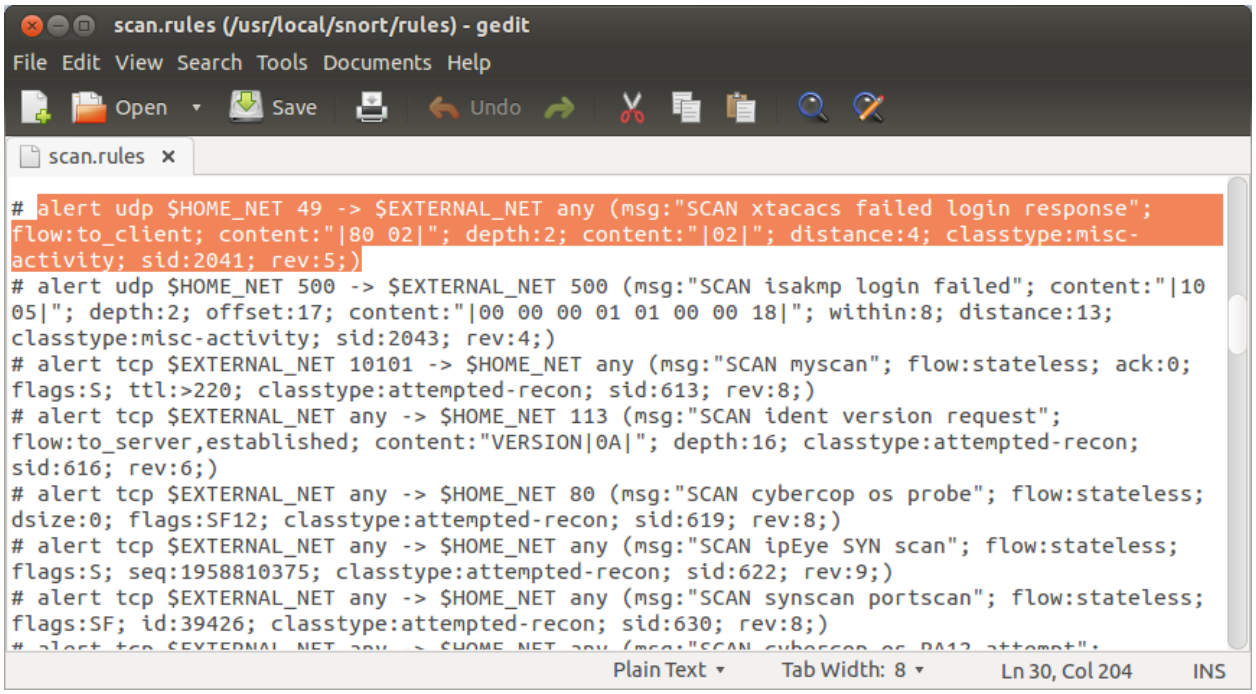


Εικόνα 24: Αρχική σελίδα του BASE και δυνατότητες

Κεφάλαιο 3 Κανόνες του Snort

Οι κανόνες στο Snort είναι ίσως το κομμάτι που πρέπει να δώσουμε μεγαλύτερη προσοχή. Είναι μια καλή αρχή για να καταλάβει κάποιος το Snort και να αρχίσει να ανιχνεύει τις πρώτες του επιθέσεις. Όπως αναφερθήκαμε και στην αρχιτεκτονική του Snort οι κανόνες λειτουργούν μαζί με την Μηχανή Ανίχνευσης (DetectionEngine) και πυροδοτούν συναγερμούς για πιθανές επιθέσεις.

Η Μηχανή Ανίχνευσης (DetectionEngine) εξετάζει κάθε πακέτο και το ελέγχει βάση των κανόνων που έχουμε ενεργοποιήσει. Οι κανόνες είναι γραμμένοι με απλή γλώσσα περιγραφής, αλλά με μεγάλη πληθώρα επιλογών, που κάνουν την αναζήτηση ύποπτων πακέτων πολύ ευέλικτη. Οι κανόνες γράφονται σε αρχεία με κατάληξη .rules και κάθε τέτοιο αρχείο (ruleset) περιέχει κανόνες για κάποιο συγκεκριμένο τύπο επίθεσης. Για παράδειγμα, το αρχείο “scan.rules” περιέχει κανόνες που είναι γραμμένοι για πακέτα που σχετίζονται σε επιθέσεις σάρωσης (scannetwork).



Εικόνα 25: Κανόνες για επιθέσεις σάρωσης (ScanRules)

Όλα αυτά τα αρχεία με τους κανόνες βρίσκονται στο φάκελο rules εκεί που έχουμε εγκαταστήσει το Snort. Για παράδειγμα για την εγκατάσταση που πραγματοποιήσαμε εμείς οι κανόνες βρίσκονται στον κατάλογο /usr/local/snort/rules/ . Επίσης κάθε ruleset πρέπει να το ενεργοποιούμε απο το αρχείο ρυθμίσεων του Snort (snort.conf).

Όλοι οι κανόνες κάθε μήνα ανανεώνονται και προσαρμόζονται ανάλογα με τις επιθέσεις. Κάθε χρήστης μπορεί να βρίσκει τους καινούριους κανόνες στην ιστοσελίδα του Snort www.snort.org.

3.1 Δομή κανόνα

Όλοι οι κανόνες αποτελούνται από δύο ενότητες, την επικεφαλίδα του κανόνα (ruleheader) και τις ρυθμίσεις του κανόνα (rule option). Στην επικεφαλίδα του κανόνα περιέχονται:

- Πληροφορίες για την ενέργεια που θα εκτελέσει το Snort σε περίπτωση που επαληθευτεί ο κανόνας,
- Το πρωτόκολλο
- Διευθύνσεις IP αποστολέα και παραλήπτη
- Θύρες (ports) αποστολέα και παραλήπτη.

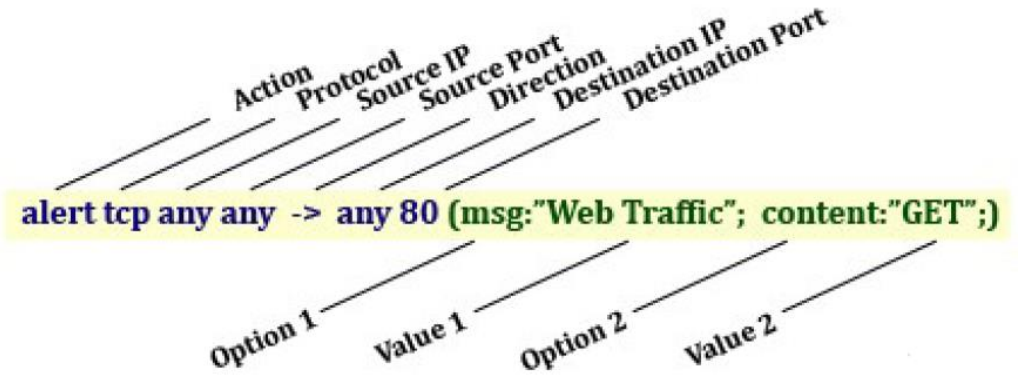
Το τμήμα των ρυθμίσεων του κανόνα περιλαμβάνει:

- Μηνύματα προειδοποίησης (alerts) που θα παράγει το Snort σε περίπτωση επαλήθευσης
- Πληροφορίες για το ποια τμήματα του εισερχόμενου πακέτου πρέπει να εξεταστούν ώστε να αποφασίσουμε αν το πακέτο είναι ύποπτο, οπότε το Snort θα δράσει κατάλληλα, ή όχι.

Ένας τυπικός κανόνας του Snort φαίνεται παρακάτω (το τμήμα της επικεφαλίδας είναι έξω απο τις παρενθέσεις, ενώ μέσα στις παρενθέσεις βρίσκεται το τμήμα των ρυθμίσεων):

alert tcp any any ->any 80 (msg:"Web Traffic"; content:"GET";)

Ο παραπάνω κανόνας, η ενέργεια που θα εκτελέσει είναι συναγερμός (Alert), το πρωτόκολλο του πακέτου θα είναι TCP, διεύθυνση αποστολέα οποιαδήποτε, η πόρτα αποστολέα οποιαδήποτε, η διεύθυνση παραλήπτη οποιαδήποτε, η πόρτα παραλήπτη 80. Επίσης βάσει του τμήματος ρυθμίσεων το μήνυμα που θα μας ενημερώνει το snort είναι "WebTraffic" και το περιεχόμενο του πακέτου πρέπει να περιέχει την εντολή GET.



Εικόνα 26: Δομή κανόνα

3.2 Επικεφαλίδα Κανόνα

Η επικεφαλίδα κάθε κανόνα περιέχει τις πληροφορίες που καθορίζουν από που έρχεται το ύποπτο πακέτο, που πηγαίνει και τι είδους είναι, καθώς επίσης και τι πρέπει να κάνουμε στην περίπτωση που μας έρθει ένα πακέτο με τα συγκεκριμένα χαρακτηριστικά του κανόνα.

Το πρώτο στοιχείο σε κάθε κανόνα είναι η ενέργεια που θα εκτελεστεί (action), δηλαδή ο τρόπος που θα αντιδράσει το Snort σε περίπτωση που βρεθεί ένα πακέτο το οποίο ικανοποιεί όλα τα στοιχεία του κανόνα. Υπάρχουν οι εξής πέντε προκαθορισμένες ενέργειες στο Snort:

- **alert** – δημιουργία μηνύματος προειδοποίησης και καταγραφή του πακέτου.
- **log** – καταγραφή του πακέτου.
- **pass** – επιτρέπεται η διέλευση του πακέτου.
- **activate** – δημιουργία προειδοποίησης και ενεργοποίηση ενός dynamic κανόνα.
- **dynamic** – παραμένει ανενεργός μέχρι να επαληθευτεί ο αντίστοιχος activate κανόνας και μετά ενεργεί ως κανόνας τύπου log.
- **drop** – Μπλοκάρει και καταγράφει το πακέτο
- **reject** – Μπλοκάρει το πακέτο, το καταγράφει και στη συνέχεια στέλνει ένα TCP reset αν το πρωτόκολλο είναι TCPή αν είναιICMP στέλνει ICMP port unreachable
- **sdrop** – Μπλοκάρει το πακέτο αλλά δε το καταγράφει.

alert tcp any any -> any 80 (msg:"Web Traffic”; content:”GET”;)

Το επόμενο πεδίο στην επικεφαλίδα ενός κανόνα είναι το πρωτόκολλο. Μέχρι στιγμής το Snort υποστηρίζει τέσσερα πρωτόκολλα στην ανάλυση εισερχομένων πακέτων, τα πρωτόκολλα tcp, udp, ip και icmp. Στο μέλλον πιθανόν να υπάρξει υποστήριξη και για άλλα πρωτόκολλα, όπως ARP, IGRP, OSPF και άλλα.

alerttcp any any ->any 80 (msg:"Web Traffic”; content:”GET”;)

Αμέσως μετά το πρωτόκολλο βρίσκεται το τμήμα που περιέχει τις πληροφορίες για τις διευθύνσεις IP και τις θύρες του κανόνα. Θα περιγράψουμε πρώτα τη μορφή των διευθύνσεων IP.

Η λέξη-κλειδί ‘any’ μπορεί να χρησιμοποιηθεί όταν θέλουμε τον κανόνα να ισχύει για οποιαδήποτε διεύθυνση IP. Κάθε διεύθυνση αποτελείται από μία συνεχόμενη διεύθυνση IP (δηλαδή τέσσερις ακέραιους αριθμούς από 0 μέχρι 255 με ‘.’ Ανάμεσα τους) και ένα μπλοκ CIDR. Το μπλοκ CIDR το οποίο υποδηλώνει τη μάσκα δικτύου που εφαρμόζεται στην διεύθυνση κάθε πακέτου πρωτού γίνει η σύγκριση με τη διεύθυνση του κανόνα. Για παράδειγμα, ο συνδυασμός 192.168.1.0/24 υποδηλώνει την ομάδα διευθύνσεων από 192.168.1.0 μέχρι 192.168.1.255. Η χρήση του CIDR μας επιτρέπει να δηλώνουμε με ένα σύντομο τρόπο μεγάλες ομάδες διευθύνσεων. Εάν κάποιο εισερχόμενο πακέτο έχει τη διεύθυνσή του μέσα στο συγκεκριμένο πεδίο τιμών τότε επαληθεύεται αυτό το κομμάτι του κανόνα.

Στις διευθύνσεις μπορεί να εφαρμοστεί ένας τελεστής αντιστροφής, ο οποίος συμβολίζεται με '!' και λέει στο Snort ότι ταιριάζουν όλες οι διευθύνσεις εκτός από την αναγραφόμενη. Για παράδειγμα, στην ακόλουθη επικεφαλίδα κανόνα δε μας ενδιαφέρει η διεύθυνση αποστολέα, καθώς είναι ρυθμισμένη σε any, και θέλουμε η διεύθυνση του παραλήπτη να μην ανήκει στην ομάδα 192.168.1.0/24.

alert tcp any any -> !192.168.1.0/24 111 (...)

Το Snort υποστηρίζει επίσης τη δήλωση ομάδας από συγκεκριμένες διευθύνσεις. Μια ομάδα διευθύνσεων εσωκλείεται σε αγκύλες και κάθε διεύθυνση διαχωρίζεται από τις γειτονικές τις με ','. Δεν επιτρέπεται η ύπαρξη κενού (space) στο εσωτερικό μιας ομάδας διευθύνσεων. Για παράδειγμα, η έκφραση ' ![192.168.1.0/24,10.1.1.0/24] ' μας λέει ότι ενδιαφερόμαστε για τις διευθύνσεις που δεν ανήκουν σε καμία από τις δύο ομάδες διευθύνσεων.

Εκτός από τις διευθύνσεις πρέπει να δηλωθούν και οι θύρες (ports) για τις οποίες ισχύει ο κανόνας. Και σε αυτή την περίπτωση μπορεί να χρησιμοποιηθεί η λέξη any αν θέλουμε να δηλώσουμε ότι ο κανόνας ισχύει για όλες τις θύρες. Μπορούμε επίσης να δώσουμε συγκεκριμένη τιμή σε μία θύρα, για παράδειγμα την τιμή 80 για http.

Επιπλέον, υπάρχει η δυνατότητα να δηλώσουμε μία ομάδα συνεχόμενων θυρών χρησιμοποιώντας ':' ανάμεσα στα δύο όρια της ομάδας. Για παράδειγμα, η έκφραση '1:1024' ισχύει για όλες τις θύρες από 1 μέχρι 1024. Όταν δηλώνουμε μία ομάδα μπορούμε να παραλείψουμε το ένα άκρο του ορισμού δηλώνοντας, με τον τρόπο αυτό, ότι ενδιαφερόμαστε για όσες διευθύνσεις είναι μικρότερες ή μεγαλύτερες από μια άλλη. Δηλαδή, μια έκφραση της μορφής '500: ' ισχύει για θύρες μεγαλύτερες οι ίσες από την 500.

Και στην περίπτωση των θυρών υπάρχει ο τελεστής αντιστροφής '!' που μπορεί να εφαρμοστεί πριν από κάθε έκφραση και αντιστρέφει την ισχύ της.

Το τελευταίο συστατικό της επικεφαλίδας ενός κανόνα είναι ο τελεστής κατεύθυνσης. Ο τελεστής αυτός βρίσκεται ανάμεσα στα δύο ζεύγη διεύθυνσης IP-θύρας και συμβολίζει την κατεύθυνση της κίνησης που μας ενδιαφέρει. Συνήθως ο τελεστής κατεύθυνσης γράφεται σαν '->', οπότε το ζεύγος που βρίσκεται αριστερά του αντιστοιχεί στη διεύθυνση IP και τη θύρα του αποστολέα, ενώ το ζεύγος δεξιά του τελεστή αντιστοιχεί στον παραλήπτη. Υπάρχει επίσης και ο αμφίδρομος τελεστής '<>' με τον οποίο δηλώνουμε ότι δε μας ενδιαφέρει ο προσανατολισμός της κίνησης. Ένα παράδειγμα χρήσης του αμφίδρομου τελεστή για την καταγραφή και των δύο πλευρών μίας επικοινωνίας telnet είναι το ακόλουθο.

log tcp !192.168.1.0/24 any <> 192.168.1.0/24 23

Πρέπει να σημειωθεί ότι δεν υπάρχει τελεστής <- ώστε να υπάρχει συνέπεια στον τρόπο που γράφονται οι κανόνες, δηλαδή το πρώτο ζεύγος να αντιστοιχεί πάντα στον αποστολέα και το δεύτερο στον παραλήπτη.

3.3 Ρυθμίσεις(Option) Κανόνα

Οι ρυθμίσεις των κανόνων αποτελούν την καρδιά του μηχανισμού ανίχνευσης του Snort καθώς συνδυάζουν ευκολία χρήσης, αποτελεσματικότητα και ευελιξία. Κάθε ρύθμιση σε έναν κανόνα διαχωρίζεται από την επόμενη με το χαρακτήρα ‘;’. Οι λέξεις-κλειδιά των ρυθμίσεων διαχωρίζονται από τα ορίσματά τους μέσω ‘:’. Οι ρυθμίσεις των κανόνων υπάγονται στις ακόλουθες τέσσερις κύριες κατηγορίες:

- **Βοηθητικές:** Παρέχουν επιπλέον πληροφορίες για τον κανόνα χωρίς, όμως, να σχετίζονται με την αναζήτηση.
- **Περιεχομένου:** Όλες οι επιλογές του τύπου αυτού προσδιορίζουν δεδομένα που θα αναζητηθούν στο περιεχόμενο (payload) του πακέτου και μπορούν να συσχετίζονται μεταξύ τους.
- **Μη-περιεχομένου:** Εξετάζουν πληροφορίες που δεν βρίσκονται στο περιεχόμενο (payload) του πακέτου.
- **Ολοκληρωμένης αναζήτησης:** Προσδιορίζουν την εκτέλεση ορισμένων γεγονότων που θα συμβούν όταν επαληθευτεί ο κανόνας.

Στη συνέχεια ακολουθεί μια σύντομη περιγραφή ρυθμίσεων που υπάγονται σε κάθε κατηγορία (αλλά όχι όλων, για περισσότερη πληροφορία μπορεί κάποιος να μελετήσει το user manual του Snort στο τμήμα των κανόνων).

Βοηθητικές ρυθμίσεις

- **msg:** η ρύθμιση ‘msg’ καθορίζει το μήνυμα που θα εμφανιστεί μαζί με το περιεχόμενο του πακέτου ή σε μία προειδοποίηση. Περιλαμβάνει απλές συμβολοσειρές κειμένου, στις οποίες χρησιμοποιείται το σύμβολο ‘\’ σαν χαρακτήρας διαφυγής όταν θέλουμε να δηλώσουμε ένα σύμβολο που μπορεί να μπερδέψει τον parser του Snort (όπως τον χαρακτήρα ‘;’).

alert tcp any any -> any 7070 (msg:"IDS411/dos-realaudio"; flags:AP; content:"|fff4 fffd 06|"; reference:arachnids,IDS411;)

- **reference:** επιτρέπει στον κανόνα να αναφερθεί σε ένα εξωτερικό σύστημα αναγνώρισης επιθέσεων. Σαν ορίσματα υποστηρίζονται ορισμένα συστήματα καθώς και μοναδικά URL. Χρησιμοποιείται από ένα κατάλληλο plugin εξόδου το οποίο παρέχει σύνδεση στην τοποθεσία όπου βρίσκονται επιπλέον πληροφορίες για την προειδοποίηση που παράγει ο κανόνας.
- **sid:** χρησιμοποιείται ως αναγνωριστικό ανάμεσα στους διάφορους κανόνες. Κάθε κανόνας πρέπει να έχει και διαφορετικό “sid”. Χρησιμοποιείται σε συνδυασμό με τη ρύθμιση “rev”.
- rev: αντιστοιχεί στον αριθμό αναθεώρησης (revision) του κανόνα. Σε συνδυασμό με τη ρύθμιση “sid” επιτρέπει την ανανέωση και αντικατάσταση των κανόνων με νεότερες πληροφορίες.
- **classtype:** χρησιμοποιείται για την κατηγοριοποίηση των κανόνων σε διαφορετικές κλάσεις. Κάθε κλάση αντιστοιχεί σε ένα διαφορετικό είδος επίθεσης εναντίον του συστήματος, ενώ επίσης έχει και μια καθορισμένη προτεραιότητα. Ο χρήστης έχει τη δυνατότητα να καθορίσει δικές του κλάσεις.

- **priority**: αναθέτει την επιθυμητή προτεραιότητα στον συγκεκριμένο κανόνα. Μπορεί να χρησιμοποιηθεί και σε περίπτωση που θέλουμε να αλλάξουμε την προκαθορισμένη προτεραιότητα της ρύθμισης "classtype". Όσο μικρότερη είναι η τιμή που δίνουμε στη ρύθμιση αυτή, τόσο πιο σημαντικός είναι ο κανόνας (δηλαδή η τιμή 1 αντιστοιχεί στη μεγαλύτερη προτεραιότητα).

Ρυθμίσεις περιεχομένου

- **content:** αποτελεί ένα από τα σημαντικότερα χαρακτηριστικά του Snort, καθώς επιτρέπει στον χρήστη να ορίζει κανόνες οι οποίοι αναζητούν συγκεκριμένες συμβολοσειρές στο εσωτερικό των πακέτων. Για τους κανόνες που περιέχουν την επιλογή "content" εκτελείται πρώτα ένας έλεγχος του πακέτου με τον κατάλληλο αλγόριθμο και, σε περίπτωση που βρεθεί το ζητούμενο περιεχόμενο, εξετάζονται οι υπόλοιπες παράμετροι. Ο έλεγχος κάνει διαχωρισμό μεταξύ κεφαλαίων και μικρών γραμμάτων (δηλαδή είναι case sensitive). Το όρισμα αυτής της ρύθμισης μπορεί να δοθεί με τη μορφή απλής συμβολοσειράς, με δεκαεξαδική μορφή ή με συνδυασμό των δύο. Το κομμάτι που αντιστοιχεί σε δεκαεξαδική μορφή ορίζεται με '|' στην αρχή και το τέλος, ενώ στο εσωτερικό κάθε χαρακτήρας γράφεται με δύο δεκαεξαδικά ψηφία και διαχωρίζεται από τον επόμενο με κενό.

  Ένα παράδειγμα content με συνδυασμό συμβολοσειρών και δεκαεξαδικών είναι το {content:"|5c 00|P|00|I|00|P|00|E|00 5c|";}. Στον ίδιο κανόνα μπορούν να υπάρχουν περισσότερες από μία εντολές "content". Επίσης, μπορούμε να έχουμε τον τελεστή αντιστροφής πριν από το όρισμα του content, γεγονός που σημαίνει ότι ο κανόνας ισχύει όταν δεν υπάρχει το συγκεκριμένο περιεχόμενο μέσα στο πακέτο. Η εντολή "content" μπορεί να συνδυαστεί με μια ή περισσότερες εντολές τροποποίησης, οι οποίες επηρεάζουν τον τρόπο με τον οποίο θα γίνει η αναζήτηση του περιεχομένου.

  Αυτές είναι οι:

  1. **depth:** καθορίζει μέχρι πόσο βάθος μέσα στο πακέτο θα εξετάσουμε για την ύπαρξη του ζητούμενου περιεχομένου. Για παράδειγμα, αν η τιμή του depth ήταν 5 τότε θα αναζητούσαμε το περιεχόμενο μέσα στα πρώτα 5 bytes των δεδομένων του πακέτου (δηλαδή τα 5 bytes αμέσως μετά τις επικεφαλίδες).

  2. **offset:** με την εντολή αυτή ο χρήστης δηλώνει πόσα bytes μετά από την αρχή των δεδομένων θέλει να ξεκινήσει η αναζήτηση για το περιεχόμενο. Δηλαδή ένα offset ίσο με 5 σημαίνει πως θα ξεκινήσει η αναζήτηση 5 bytes μετά την αρχή των δεδομένων.

  3. **distance:** για να χρησιμοποιηθεί η εντολή αυτή χρειάζεται προηγουμένως να έχουμε ορίσει δύο content προς αναζήτηση. Η εντολή αυτή υποδηλώνει μέχρι πόσα bytes από το τέλος του πρώτου content θα ψάξουμε για το δεύτερο content. Μπορεί να θεωρηθεί κάτι αντίστοιχο με το depth ως προς τη θέση που τελειώνει το πρώτο content.

4. **within:** και αυτή η εντολή απαιτεί την ύπαρξη δύο content. Με την εντολή αυτή δηλώνουμε μέχρι πόσα bytes επιτρέπουμε να παρεμβάλλονται ανάμεσα σε δύο content.

5. **nocase:** η εντολή αυτή υποδηλώνει ότι η αναζήτηση του περιεχομένου θα γίνει χωρίς διαφοροποίηση ανάμεσα σε κεφαλαία και μικρά γράμματα (δηλαδή χωρίς case sensitivity).

6. **rawbytes:** η εντολή αυτή επιτρέπει στον κανόνα να εξετάζει τα αρχικά περιεχόμενα του πακέτου, αγνοώντας οποιαδήποτε αποκωδικοποίηση μπορεί να έγινε από τους προεπεξεργαστές.

- **uricontent:** η παράμετρος "uricontent" στη γλώσσα του Snort υποδηλώνει αναζήτηση στο κοινωνικοποιημένο πεδίο αίτησης URI. Αυτό σημαίνει πως, σε περίπτωση που ένας κανόνας περιλαμβάνει στοιχεία που κοινωνικοποιούνται, όπως %2f ή αλλαγές καταλόγου, ο κανόνας αυτός δε θα ενεργοποιηθεί ποτέ. Αυτό γίνεται επειδή ο κανόνας αναζητά για πράγματα που αφαιρούνται κατά την κοινωνικοποίηση.
  Για παράδειγμα, το URI: /scripts/..%c0%af../winnt/system32/cmd.exe?/c+ver
  κοινωνικοποιείται σε:
  /winnt/system32/cmd.exe?/c+ver

- **isdataat:** με την εντολή αυτή προσθέτουμε τον περιορισμό πως θέλουμε να υπάρχουν δεδομένα στη συγκεκριμένη θέση, δηλαδή η θέση αυτή να μη βρίσκεται εκτός ορίων του πακέτου. Μπορεί να χρησιμοποιηθεί ταυτόχρονα με ένα "content" και με την παράμετρο "relative", στην οποία περίπτωση θέλουμε να υπάρχει περιθώριο N bytes από το τέλος του "content".
- **pcre:** επιτρέπει τη συγγραφή κανόνων σε μορφή κανονικών εκφράσεων συμβατών με perl.
- **byte_test:** συγκρίνει ένα πεδίο από bytes με μία συγκεκριμένη ακολουθία. Η σύγκριση μπορεί να γίνει με αρκετούς τελεστές (για παράδειγμα <,>,=,!,&).
- **byte_jump:** η εντολή αυτή μας επιτρέπει να εξάγουμε ορισμένα bytes από τα δεδομένα, να τα μετατρέπουμε στον αντίστοιχο αριθμό και να εκτελούμε ένα άλμα κατά τόσα bytes για περαιτέρω εξέταση περιεχομένου ή σύγκριση bytes. Με τον τρόπο αυτό μπορούν να δημιουργηθούν μέθοδοι ανίχνευσης που λαμβάνουν υπόψη τις αριθμητικές τιμές που περιέχονται στα δεδομένα.

Ρυθμίσεις μη-περιεχομένο

- **fragoffset:** επιτρέπει τη σύγκριση του πεδίου fragment offset της επικεφαλίδας IP με μια καθορισμένη τιμή. Για παράδειγμα, αν η τιμή fragoffset είναι ίση με 0 τότε ο κανόνας ενεργοποιείται στα πακέτα που είναι αδιάσπαστα ή είναι το πρώτο κομμάτι ενός μεγαλύτερου πακέτου.
- **ttl:** σύγκριση της τιμής TTL (TimeToLive) της επικεφαλίδας IP με μία συγκεκριμένη τιμή ή πεδίο τιμών.
- **tos:** έλεγχος του πεδίου TOS (TypeOfService) του IP με μία τιμή.

- **id:** έλεγχος του πεδίου id της IP επικεφαλίδας για μια συγκεκριμένη τιμή. Ορισμένα εργαλεία θέτουν συγκεκριμένες τιμές σε αυτό το πεδίο για διάφορους σκοπούς.
- **ipopts:** ελέγχει αν υπάρχουν στο πακέτο ορισμένες επιλογές (options) του πρωτοκόλλου IP. Ο αριθμός και το είδος των options για τις οποίες γίνεται έλεγχος καθορίζεται από το χρήστη.
- **Fragbits:** εξετάζει αν τα bits του κατακερματισμού και το δεσμευμένο bit είναι ενεργοποιημένα στην επικεφαλίδα IP.
- **dsize:** εξετάζει αν τα δεδομένα του πακέτο έχουν μέγεθος που κυμαίνεται μέσα σε καθορισμένα όρια. Μπορεί να χρησιμοποιηθεί για τον έλεγχο ασυνήθιστα μεγάλων πακέτων και για την ανίχνευση υπερχείλισης buffer.
- **flags:** ελέγχει τις τιμές που έχουν συγκεκριμένα flags της επικεφαλίδας TCP. Ταflags που εξετάζονται και οι επιθυμητές τιμές καθορίζονται από τον χρήστη.
- **flow:** χρησιμοποιείται παράλληλα με τον προεπεξεργαστή επανασύνδεσης ροών TCP (TCPstreamreassembly). Επιτρέπει την εφαρμογή του κανόνα μόνο σε συγκεκριμένη κατεύθυνση της ροής, για παράδειγμα στη ροή προς τον server.
- **seq:** εξετάζει αν ο αριθμός ακολουθίας (sequence) στην επικεφαλίδα TCP έχει μια συγκεκριμένη τιμή.
- **ack:** ελέγχει μήπως ο αριθμός επιβεβαίωσης (acknowledgement) στην επικεφαλίδα TCP έχει μια συγκεκριμένη τιμή.
- **window:** εξετάζει το μέγεθος του παραθύρου TCP.
- **type:** ελέγχει για ένα συγκεκριμένο τύπο (type) ICMP.
- **icode:** συγκρίνει τον κωδικό (code) ICMP με μία τιμή ή εξετάζει αν βρίσκεται μέσα σε συγκεκριμένα όρια.
- **icmp_id:** ελέγχει για συγκεκριμένη τιμή του ICMPid. Αποτελεί χρήσιμο εργαλείο καθώς ορισμένα προγράμματα κρυφών καναλιών χρησιμοποιούν συγκεκριμένες τιμές των πεδίων ICMP.
- **icmp_seq:** με την εντολή αυτή ελέγχουμε την τιμή ακολουθίας του ICMP.
- **rpc:** η εντολή αυτή χρησιμοποιείται για την ανίχνευση των αριθμών της εφαρμογής (application), της έκδοσης (version) και της διαδικασίας (procedure) σε αιτήσεις SUNRPCCALL. Πρέπει να σημειωθεί πως η εντολή RPC είναι πιο γρήγορη από ότι αν ψάχναμε για τις συγκεκριμένες τιμές χρησιμοποιώντας εντολές content.
- **ip_proto:** αυτή η εντολή επιτρέπει τη σύγκριση του αριθμού του IP με ορισμένες τιμές.
- same_ip: εξετάζει αν η διεύθυνση αποστολέα και παραλήπτη είναι ίδιες.

Ρυθμίσεις ολοκληρωμένης αναζήτησης

- **logto:** η χρήση της εντολής σε έναν κανόνα καθορίζει το αρχείο μέσα στο οποίο θα αποθηκεύονται όσα πακέτα επαληθευτούν.
- **session:** χρησιμοποιείται για να εξάγει δεδομένα από επικοινωνίες (sessions) TCP. Υπάρχουν περιπτώσεις όπου είναι επιθυμητό να δούμε τι στάλθηκε από τους χρήστες κατά τη διάρκεια μίας επικοινωνίας telnet ή ftp για παράδειγμα. Η χρήση της εντολής "session" μπορεί να επιβραδύνει σημαντικά το Snort, επομένως είναι καλύτερα να αποφεύγεται η χρήση της σε περιπτώσεις έντονης κίνησης.

- **resp:** με την εντολή αυτή επιχειρείται το κλείσιμο μίας επικοινωνίας (session) σε περίπτωση που επαληθευτεί ο κανόνας. Στο Snort, η τακτική αυτή καλείται Ευέλικτη Απάντηση (FlexibleResponse). Η λειτουργία του μηχανισμού ευέλικτης απάντησης δεν είναι προκαθορισμένα ενεργή, χρειάζεται να ενεργοποιηθεί από τον χρήστη. Πάντως χρειάζεται προσοχή στη χρήση της εντολής αυτής καθώς είναι πιθανό να δημιουργηθεί άπειρος βρόχος.
- **react:** και αυτή η εντολή στηρίζεται στο μηχανισμό ευέλικτης απάντησης και παρέχει επιπλέον τρόπους διαχείρισης των ροών που ενεργοποιούν τον αντίστοιχο κανόνα. Παρέχει τη δυνατότητα κλεισίματος της ροής που ενεργοποίησε τον κανόνα καθώς και την δυνατότητα εμφάνισης κατάλληλου προειδοποιητικού μηνύματος στο χρήστη. Όπως με τη "resp", έτσι και με τη "react" υπάρχει ο κίνδυνος δημιουργίας άπειρου βρόχου.
- **tag:** επιτρέπει την καταγραφή περισσότερων πακέτων εκτός από το ένα που επαληθεύει τον κανόνα. Μόλις ο κανόνας που περιέχει την εντολή "tag" ενεργοποιηθεί, η επιπλέον κίνηση του δικτύου που περιλαμβάνει τον ίδιο αποστολέα ή/και τον παραλήπτη καταγράφεται, ώστε να είναι δυνατή η μετέπειτα ανάλυση της κίνησης που παρουσιάστηκε αμέσως μετά την επίθεση.

Κεφάλαιο 4 Ανίχνευση Επιθέσεων με το Snort

Το Snort όπως έχουμε πει είναι ένα εργαλείο που διαβάζει όλα τα εισερχόμενα και εξερχόμενα πακέτα με σκοπό να ανιχνεύσει επιθέσεις μέσα στο δίκτυο. Σε αυτό το κομμάτι θα περιγράψουμε κάποιες γνωστές επιθέσεις και θα τις ανιχνεύσουμε με το Snort. Να αναφέρουμε ότι το Snort έκτος από IDS μπορεί να λειτουργήσει και σαν IPS, όπου θα περιγράψουμε αυτή την λειτουργία αργότερα.

Το λειτουργικό που χρησιμοποιήσαμε για να εκτελέσουμε τις επιθέσεις είναι το Backtrack 5 που είναι βασισμένο σε Linux και περιέχει εγκατεστημένα αρκετά προγράμματα για επιθέσεις. Το Backtrack είναι εγκατεστημένο σε εικονικό υπολογιστή με VirtualBox. Το Snort είναι εγκατεστημένο σε λειτουργικό σύστημα Linux-Ubuntu 12.10 64 bit.

4.1 Ανίχνευση Port Scan με το Snort

Περιγραφή επίθεσης

Κάθε εισβολέας επιλέγει έναν υπολογιστή στόχο. Αρχικά θα προσπαθήσει να καθορίσει ποιο λειτουργικό σύστημα εκτελεί και ποιες υπηρεσίες παρέχει στους πελάτες του δικτύου. Σε ένα δίκτυο TCP/IP (όπως το Internet), οι υπηρεσίες παρέχονται σε αριθμημένες συνδέσεις, που καλούνται θύρες. Οι θύρες στις οποίες αποκρίνεται ένας υπολογιστής καθορίζουν συνήθως το λειτουργικό σύστημα και τις παρεχόμενες υπηρεσίες του υπολογιστή στόχου.

Υπάρχουν αρκετά εργαλεία στο Internet, τα οποία μπορεί να χρησιμοποιήσει ένας επιτιθέμενος για να καθορίσει ποιες θύρες αποκρίνονται σε αιτήσεις συνδέσεων δικτύου. Αυτά τα εργαλεία δοκιμάζουν κάθε θύρα με τη σειρά και αναφέρουν στον εισβολέα ποιες θύρες αρνούνται τις συνδέσεις και ποιες όχι. Ο εισβολέας μπορεί κατόπιν να επικεντρώσει την προσοχή του στις θύρες που αποκρίνονται σε υπηρεσίες, οι οποίες συχνά μένουν ανασφάλιστες ή έχουν προβλήματα ασφάλειας.

Η σάρωση θυρών μπορεί να αποκαλύψει ποιο λειτουργικό σύστημα χρησιμοποιεί ο υπολογιστής, επειδή κάθε λειτουργικό σύστημα έχει ένα διαφορετικό σύνολο προεπιλεγμένων υπηρεσιών. Αυτές οι πληροφορίες είναι που "λένε" στον επιτιθέμενο, ποια εργαλεία να χρησιμοποιήσει για να εισβάλει σε ένα δίκτυο.

Υπάρχουν αρκετές τεχνικές για μία τέτοια επίθεση όπως TCP scanning, SYN scanning, UDP scanning, ACK scanning, FIN scanning, X-masscan, NullScan κτλ. Σε αυτό το παράδειγμα θα εκτελέσουμε FIN και X-masScan. Για να εκτελέσουμε αυτές τις επιθέσεις θα χρησιμοποιήσουμε το πρόγραμμα nmap που υπάρχει εγκατεστημένο στο Backtrack 5. Ο στόχος θα είναι το τοπικό μηχάνημα όπου τρέχει το Snort και ελέγχει όλα τα εισερχόμενα και εξερχόμενα πακέτα

Ρύθμιση Snort

Προετοιμάζουμε το Snort κάνοντας κάποιες απαραίτητες ενέργειες. Αρχικά ελέγχουμε το αρχείο με τους κανόνες σάρωσης (scan.rules), που βρίσκεται στο path /usr/local/snort/rules/, αν έχει γραμμένους κανόνες που θα ανιχνεύουν αυτούς τους τύπους Portscanning. Αν όχι πρέπει να φτιάξουμε ή να τους βρούμε και να τους γράψουμε στο αρχείο. Για παράδειγμα για τον εντοπισμό FINscan πρέπει να υπάρχει ο κανόνας:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN FIN"; flow:stateless; flags:F,12; reference:arachnids,27; classtype:attempted-recon; sid:621; rev:7;)

Εξήγησητηςκεφαλής(header) τουκανόνα“**alert tcp $EXTERNAL_NET any -> $HOME_NET any**”:

- **alert:** Πυροδοτείσυναγερμό σε περίπτωση που ο κανόνας επαληθευτεί
- **tcp:** Ο κανόναςελέγχει τα TCP πακέτα
- **$EXTERNAL_NET:** Έλεγχει αν η διεύθυνση αποστολέα του πακέτου ανοίκει στο εξωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any:** Το πακέτο να προέρχεται από οποιαδήποτε πόρτα του αποστολέα (Δε μας ενδιαφέρει αυτό το πεδίο)
- **->:** Είναι τελεστής κατεύθυνσης πού ότι βρίσκεται αριστερά του αντιστοιχεί στη διεύθυνση IP και τη θύρα του αποστολέα, ενώ το ζεύγος δεξιά του τελεστή αντιστοιχεί στον παραλήπτη.
- **$HOME_NET:** Έλεγχει αν η διεύθυνση παραλήπτητου πακέτου ανοίκει στο εσωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any:** Το πακέτο προορίζεται σε οποιαδήποτε πόρτα του παραλήπτη (Δε μας ενδιαφέρει αυτό το πεδίο).

Εξήγησητωνρυθμήσεων (Options) του κανόνα “**(msg:"SCAN FIN"; flow:stateless; flags:F,12; reference:arachnids,27; classtype:attempted-recon; sid:621; rev:7;)**“:

- **msg:"SCANFIN";** : Όταν πυροδοτιθεί ο παραπάνο συναγερμός θα περιέχεται το μήνυμα “SCANFIN” για να μας διευκολίνει στην αναγνώριση της επίθεσης.
- **flow:stateless; :** Ενεργοποιήτε συναγερμός ανεξάρτητα της κατεύθυνσης της ροής των TCP πακέτων.
- **flags:F,12; :** Επίσης ελέγχεται αν στην επικεφαλίδα του πακέτου είναι το πεδίο flageίναι ενεργοποιημένο “F” με τιμή 1 ή 2.
- **reference:arachnids,27;**: Συνδεέται με το εξωτερικό σύστημα **“arachnids”** για περισσότερες πληροφορίες για την προηδοποίηση που παράγει ο κανόνας.
- **classtype:attempted-recon;:** Αυτός ο κανόνας θα κατηγοροποιηθεί σε κλάση τύπου “attempted -recon”. Κάθε κλάση αντιστοιχεί σε ένα διαφορετικό είδος επίθεσης εναντίον του συστήματος, ενώ επίσης έχει και μια καθορισμένη προτεραιότητα. Ο χρήστης έχει τη δυνατότητα να καθορίσει δικές του κλάσεις.
- **sid:621; :** Ο κανόνας αυτός έχει σαν αναγνωριστικό αριθμό ανάμεσα στους υπόλοιπους κανόνες τον αριθμό “621”.

- **rev:7; :** Ο κανόνας αυτός θα έχει αριθμό αναθεώρησης 7. Σε συνδυασμό με τη ρύθμιση "sid" επιτρέπει την ανανέωση και αντικατάσταση των κανόνων με νεότερες πληροφορίες.

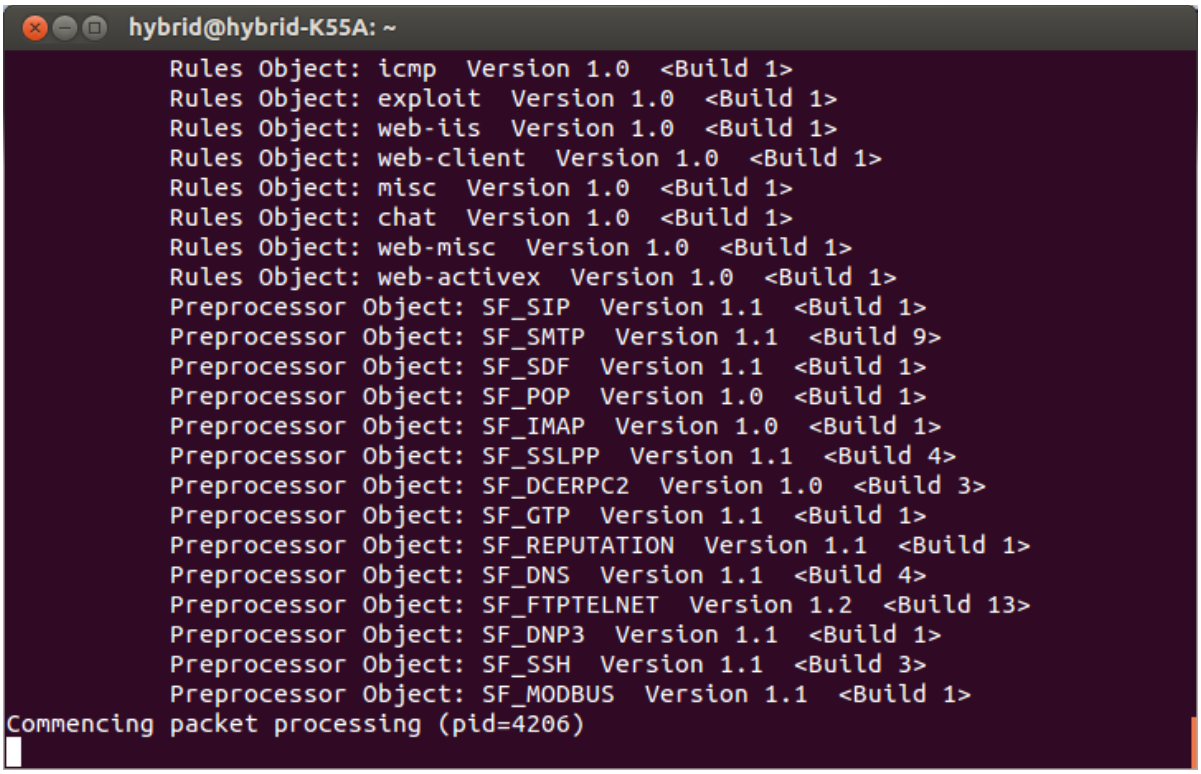
Επίσης στο αρχείο ρυθμίσεων του Snort πρέπει να ελέγξουμε αν είναι ενεργοποιημένο το συγκεκριμένο ruleset (scan.rules).

Αφού κάνουμε τις ρυθμίσεις τώρα μπορούμε να τρέξουμε το Snort με την εντολή:

sudo /usr/local/snort/bin/snort -c /usr/local/snort/etc/snort.conf -i wlan0 -A console

- **-c** /usr/local/snort/etc/snort.conf: Πποσδιορίζουμετο configuration fileπουθαχρησιμοποιήσει.
- **-i** wlan0: Επιλέγουμε την ασύρματη κάρτα για να πιάσουμε τα πακέτα.
- **-A** console: Επιλέγουμε τα alerts να τα δούμε στην κονσόλα.

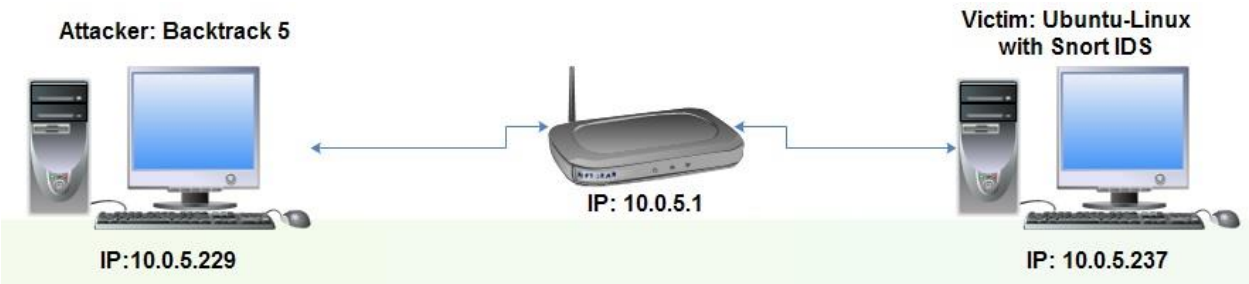
Έτσι το Snort πρέπει τώρα να είναι σε κατάσταση IDS και να ελέγχει τα πακέτα.



Εικόνα 27: Το snort τρέχει και περιμένει ενδεχόμενη επίθεση

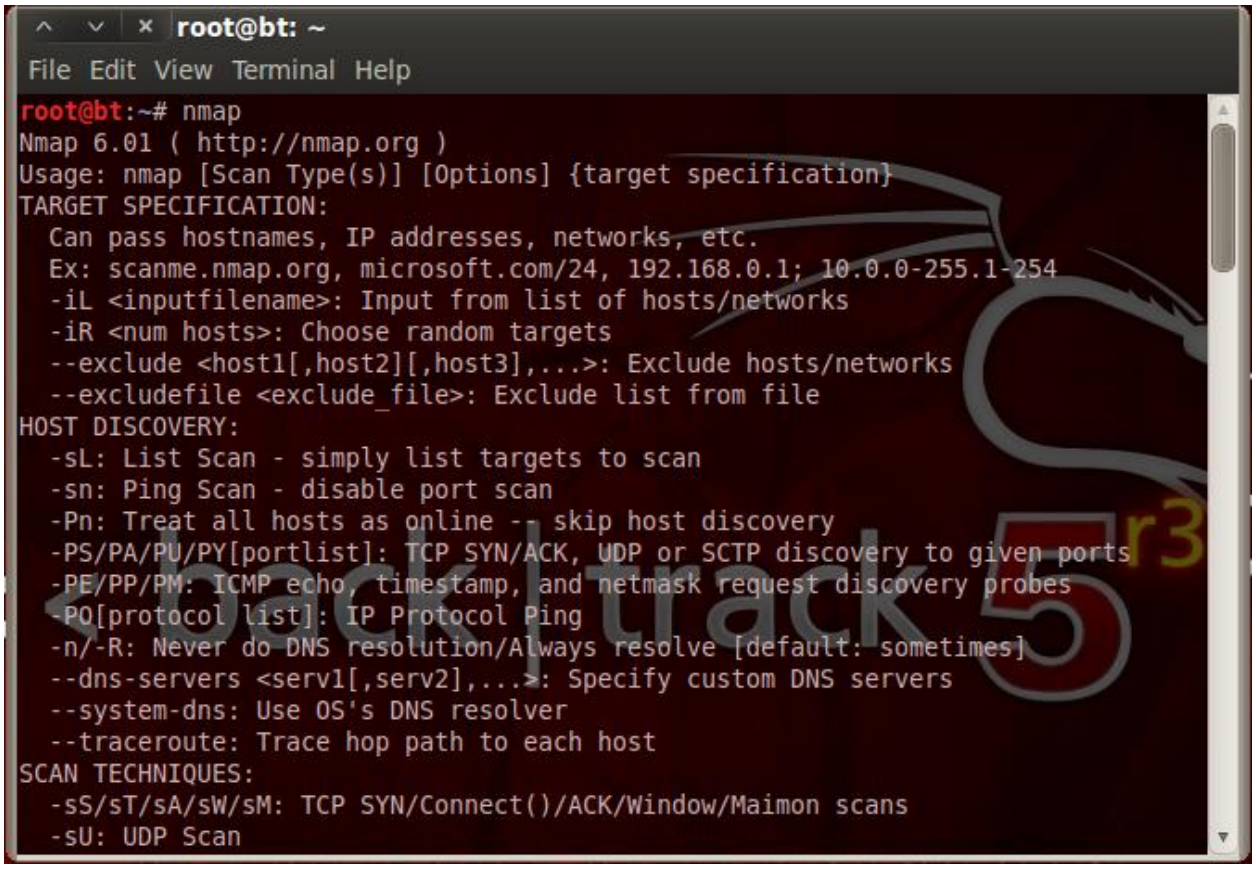
Εκτέλεση επίθεσης

. Η επίθεση πραγματοποιήθηκε εσωτερικά σε ασύρματο τοπικό δίκτυο όπως φαίνετε στο παρακάτω σχήμα:



Εικόνα 28: Σχημα επίθεσης PortScan

Αφού προετοιμαστήκαμε για επιθέσεις portscan τώρα θα τρέξουμε από το Backtrack 5 επιθέσεις portscanning. Όπως είπαμε θα χρησιμοποιήσουμε το nmap. Έτσι στο Backtrack ανοίγουμε ένα τερματικό και γράφουμε nmap, ώστε να δούμε όλες τις εντολές που θα χρειαστούμε για να κάνουμε τις επιθέσεις μας.

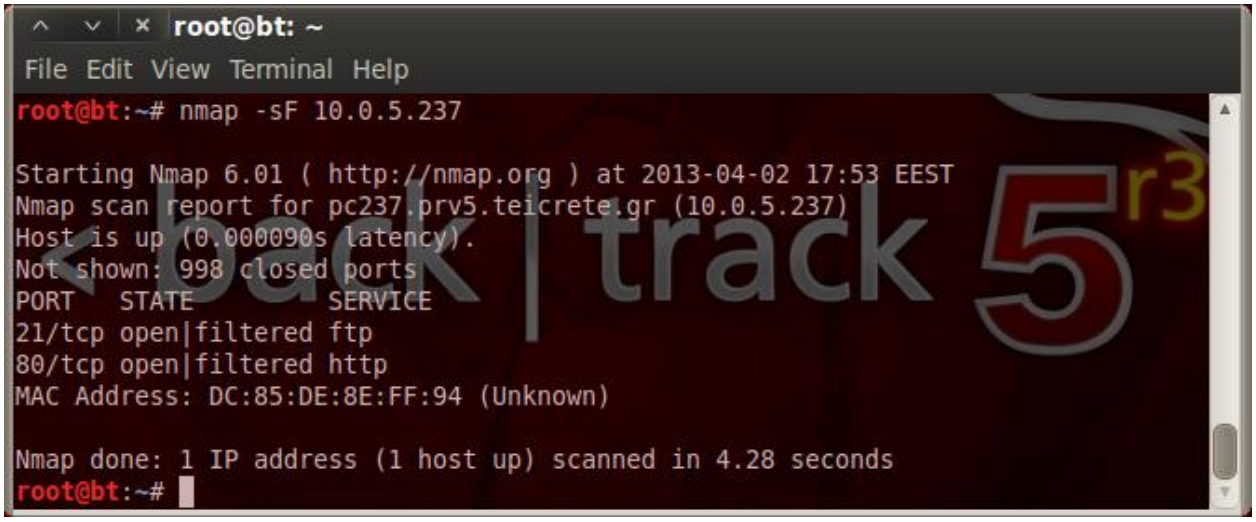


Εικόνα29: nmap usage στο backtrack terminal

Έτσι για να εκτελέσουμε FIN scan στις πόρτες του host στόχου με IP 10.0.5.237, πρέπει να γράψουμε την εντολή.

nmap –sF 10.0.5.237

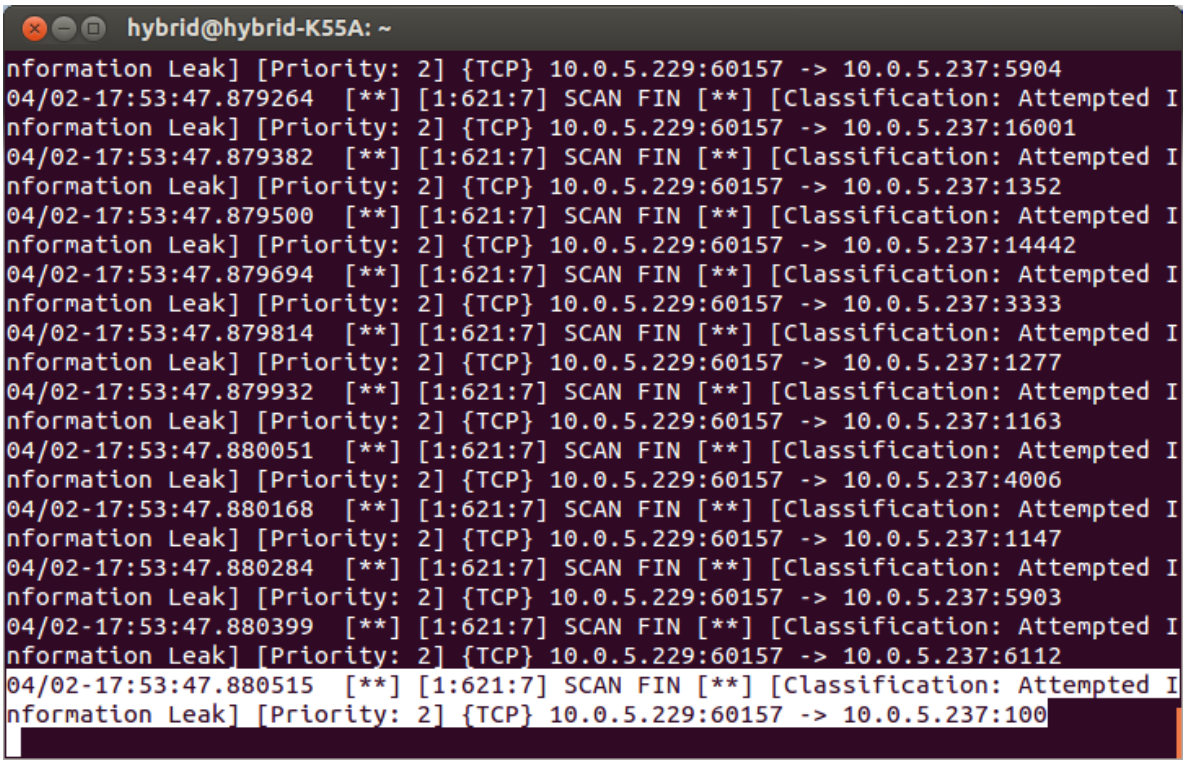
Βλέπουμε ότι υπάρχουν 2 πόρτες ανοιχτές και άλλες 998 κλειστές. Αυτό σημάνει ότι το nmap έστειλε 1000 αιτήσεις με πακέτα TCPFIN.



Εικόνα 30: Εκτέλεση επίθεσης FINSCAN

Ανίχνευση επίθεσης

Τώρα το Snort πρέπει να έχει 1000 συναγερμούς για επίθεση SCAN FIN. Έτσι αν δούμε στην κονσόλα που τρέχει το Snort θα δούμε αυτούς τους συναγερμούς. Στην παρακάτω εικόνα φαίνονται τα alerts που έχει παράγει το Snort με μήνυμα SCAN FIN.

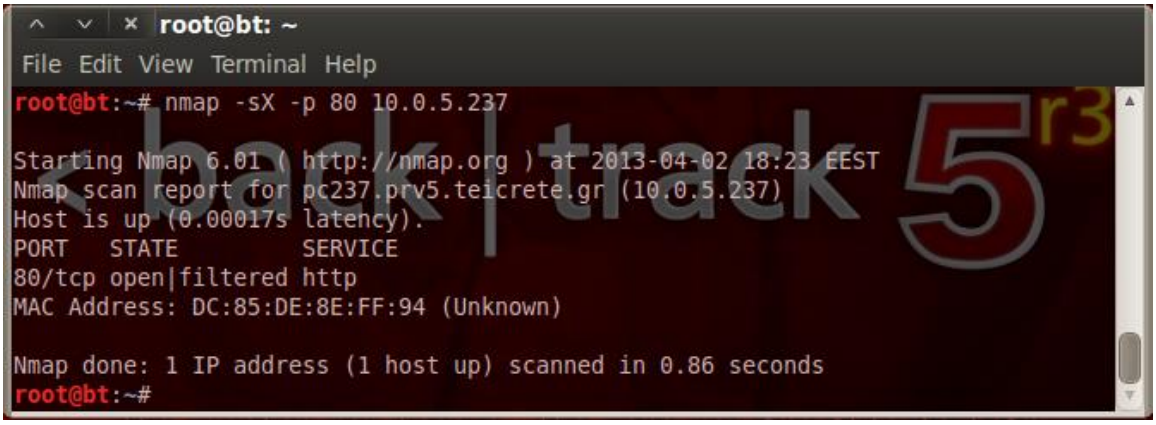


Εικόνα 31: Alertsαπό επίθεση FINSCAN

Στο κάθε alert φαίνεται η ημερομηνία, η ώρα, το προειδοποιητικό μήνυμα, η προτεραιότητα του συναγερμού για επεξεργασία από το snort και οι διευθύνσεις αποστολέα και παραλήπτη. Να υπενθυμίσουμε ότι τα Alerts μπορούμε να τα επεξεργαστούμε με διάφορα εργαλεία, όπως είναι το BASE που περιγράψαμε πριν.

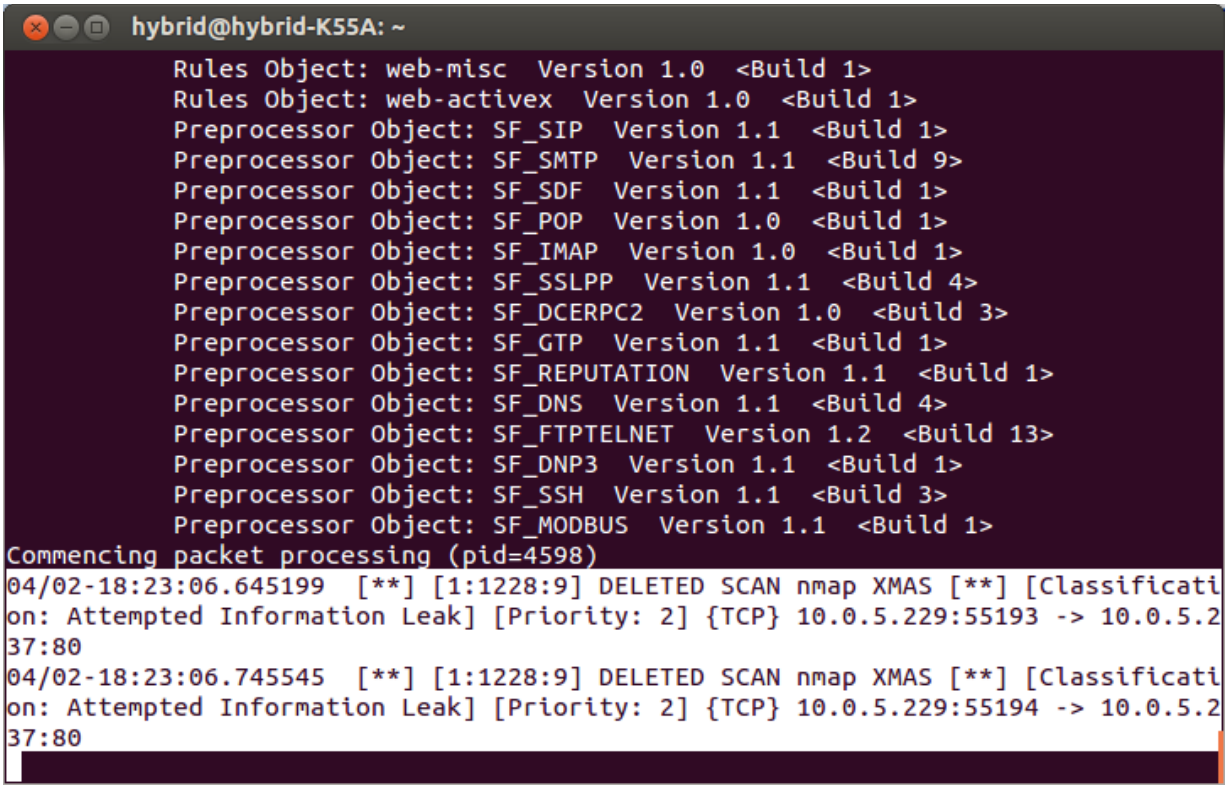
Με τον ίδιο τρόπο εκτελούμε επίθεση Xtmas scan αλλά συγκεκριμένα για την θύρα 80 τώρα. Γράφουμε στο τερματικό του Backtrack 5 την εντολή:

nmap –sX –p 80 10.0.5.237



Εικόνα 32: Εκτέλεση επίθεσης Xtmas στην θύρα 80 του 10.0.5.237

Τώρα το nmap έστειλε πακέτα για να ελέγξει μόνο μία θύρα, οπότε οι συναγερμοί δεν θα είναι όπως πριν. Στην παρακάτω εικόνα φαίνονται τα Alerts από το Snort.



Εικόνα 33: Alertsαπό επίθεση Xmas στην θύρα 80 του 10.0.5.237

Το DELETED στο μήνυμα δηλώνει ότι ο κανόνας ανοίκει στην κατηγορία deleted.rules. Είναι μια κατηγορία που περιέχει όλους τους κανόνες που έχουν καταργιθεί ή διαγραφεί ή αντικατασταθεί. Το Snort δε διαγράφει τελείως ποτέ αυτους τους κανόνες, γιατί υπάρχει περίπτωση να χρησιμοποιηθούν άμεσα.

Ανίχνευση PortScan με τον Προεπεξεργαστή sfPortscan

Το Snort για την ανίχνευση Port Scan διαθέτει τον Προεπεξεργαστή sfPortscan που μπορεί να ανιχνεύσει κάποιες μεθόδους Port Scan αλλα όχι όλες (περισσότερα στο Snort Manual για τους προεπεξεργαστές). Για αυτό και χρησιμοποιήσαμε κανόνες για την ανίχνευση FIN scan και XMAS scan. Επίσης χρειαζόμαστε τους κανόνες για να ανιχνεύσουμε μία μονο σάρωση θύρας. Με τους παραπάνω κανόνες το πετυχαίνουμε και αυτό αλλά αν γίνεται σάρωση σε πολλές θύρες π.χ. με FINscan, τότε θα παραχθούν αρκετά Alerts όπως είδαμε.

Ο Προεπεξεργαστής sfPortscan έχει το πλεονέκτημα όμως να παράγει ένα Alert για κάθε σάρωση και όχι χιλιάδες για κάθε πόρτα που σαρώνεται. Σε αυτό το παράδειγμα θα εκτελέσουμε TCP PortScan και UDP PortScan και θα ανιχνεύσουμε αυτές τις σαρώσεις με τον Προεπεξεργαστή. Επίσης θα ρυθμήσουμε το Snort να αποθηκεύει τα alerts σε log αρχεία ώστε να τα επεξεργαστούμε και να τα δούμε στο Base.

Αρχικά πρέπει να τρέξουμε το Snort ώστε να αποθηκεύει τους συναγερμούς σε log αρχεία. Έτσι θα ξεκινήσουμε το Snortμε την εντολή:

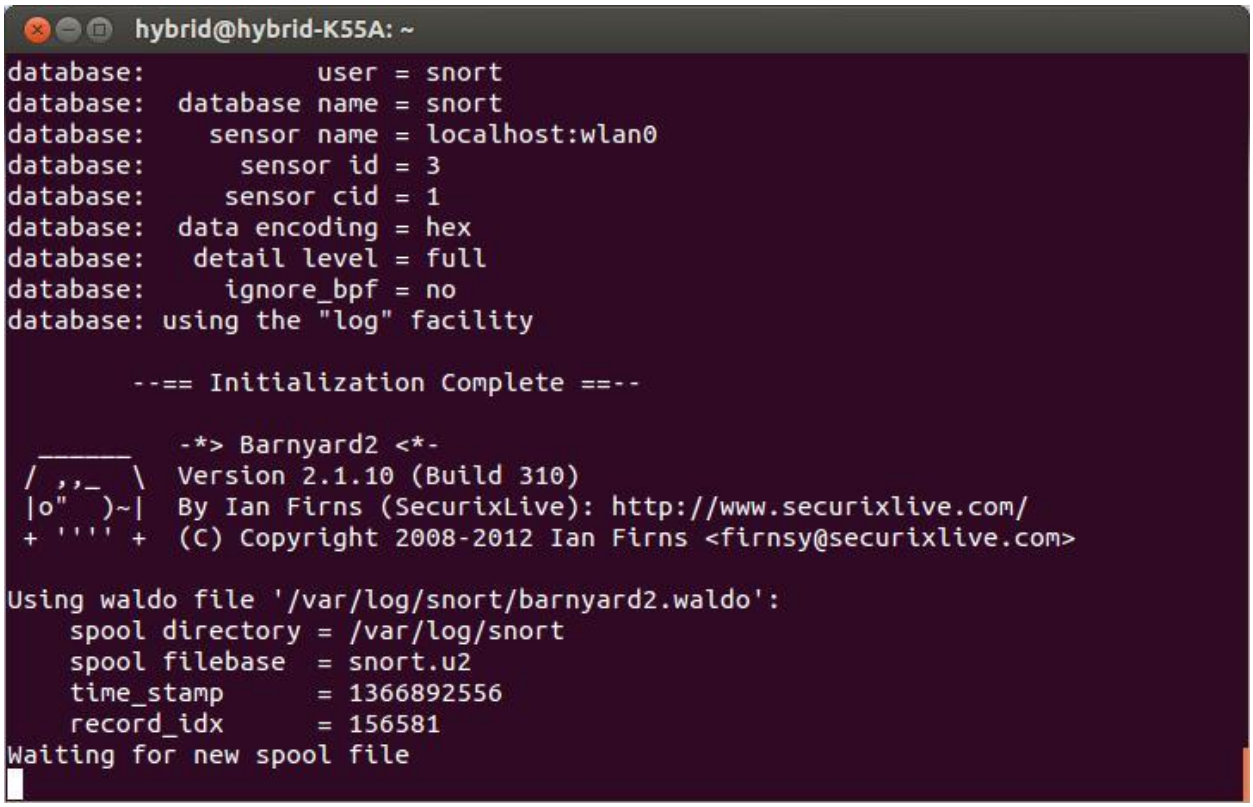
sudo /usr/local/snort/bin/snort -c /usr/local/snort/etc/snort.conf –i wlan0 -l /var/log/snort

- **-c /usr/local/snort/etc/snort.conf**: Ππροσδιορίζουμε το configuration file που θα χρησιμοποιήσει.
- **-i wlan0:** Επιλέγουμε την ασύρματη κάρτα για να πιάσουμε τα πακέτα.
- **-l /var/log/snort:** Επιλέγουμε την διαδρομή που θα αποθηκευτούν τα alerts σε αρχεία log

Όπως έχουμε περιγράψει σε προηγούμενο κεφάλαιο θα τρέξουμε και το Barnyard2, ώστε να διαβαστούν τα logαρχεία, η χρήσιμη πληροφορία να αποθηκευτεί στη βάση δεδομένων που φτιάξαμε κατά την εγκατάσταση του Snortκαι το BASE να πάρει τα δεδομένα από αυτή ώστε να μπορέσουμε να τα δούμε τα Alerts σε Web Interface. Η εντολή που θα χρησιμοποιήσουμε για να τρέξουμε το Barnyard2 συμφωνα με την εγκατάσταση που περιγράψαμε στο κεφάλαιο εγκατάστασης του Snort θα είναι:

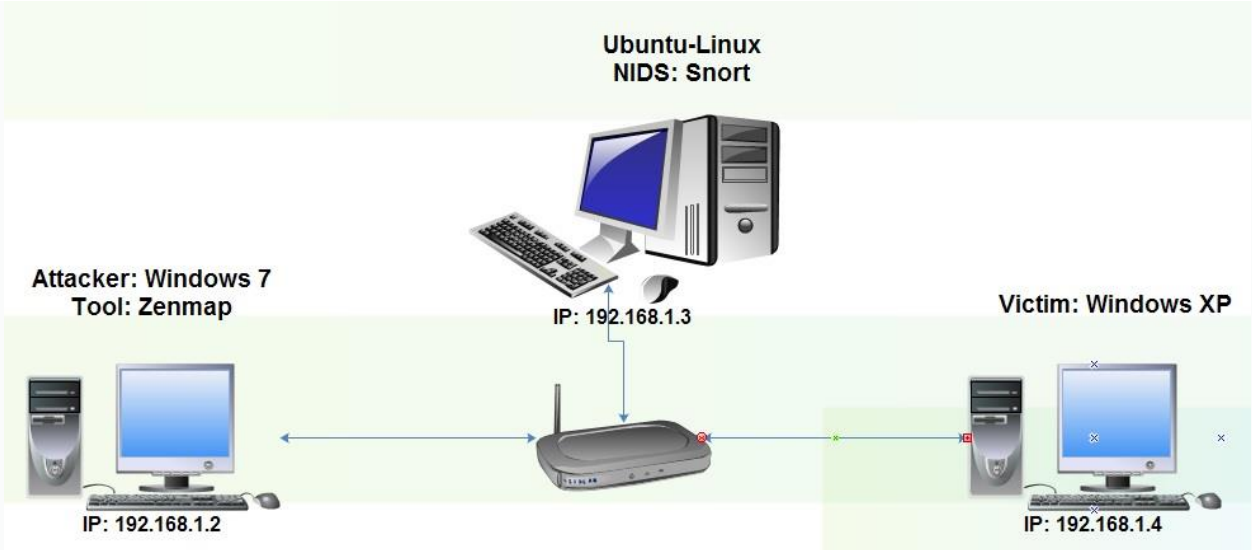
sudo /usr/local/bin/barnyard2 -c /usr/local/snort/etc/barnyard2.conf -G /usr/local/snort/etc/gen-msg.map -S /usr/local/snort/etc/sid-msg.map -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo

- **-c /usr/local/snort/etc/barnyard2.conf**: ΠροσδιορίζουμετοαρχείορυθμήσεωντουBarnyard2
- **-G /usr/local/snort/etc/gen-msg.map**: Προσδιορίζουμεαπόποιοαρχείο θα διαβάσει το gen-msgmap
- **-S /usr/local/snort/etc/sid-msg.map**: Προσδιορίζουμεαπόποιοαρχείο θα διαβάσει το sid-msgmap
- **-d /var/log/snort**: Προσδιορίζουμε το path με τα log αρχεία
- **-fsnort.u2**: Προσδιορίζουμε το όνομα των log αρχείων
- **-w /var/log/snort/barnyard2.waldo**: Προσδιορίζουμε ένα αρχείο waldo που επιτρέπει το Barnyard να λειτουργεί σε συνεχόμενη λειτουργία και να διαβάζει ότι καινούριο alert γίνεται log.



Εικόνα 34: Τρέχοντας το Barnyard2

Οι επίθεσεις πραγματοποιήθηκαν από τοπικο υπολογιστή προς άλλο τοπικό υπολογιστήκαι ανιχνεύτικαν από το τοπικό υπολογιστή με το Snort στο ίδιο δίκτυο όπως φαίνεται στο παρακάτω σχήμα:

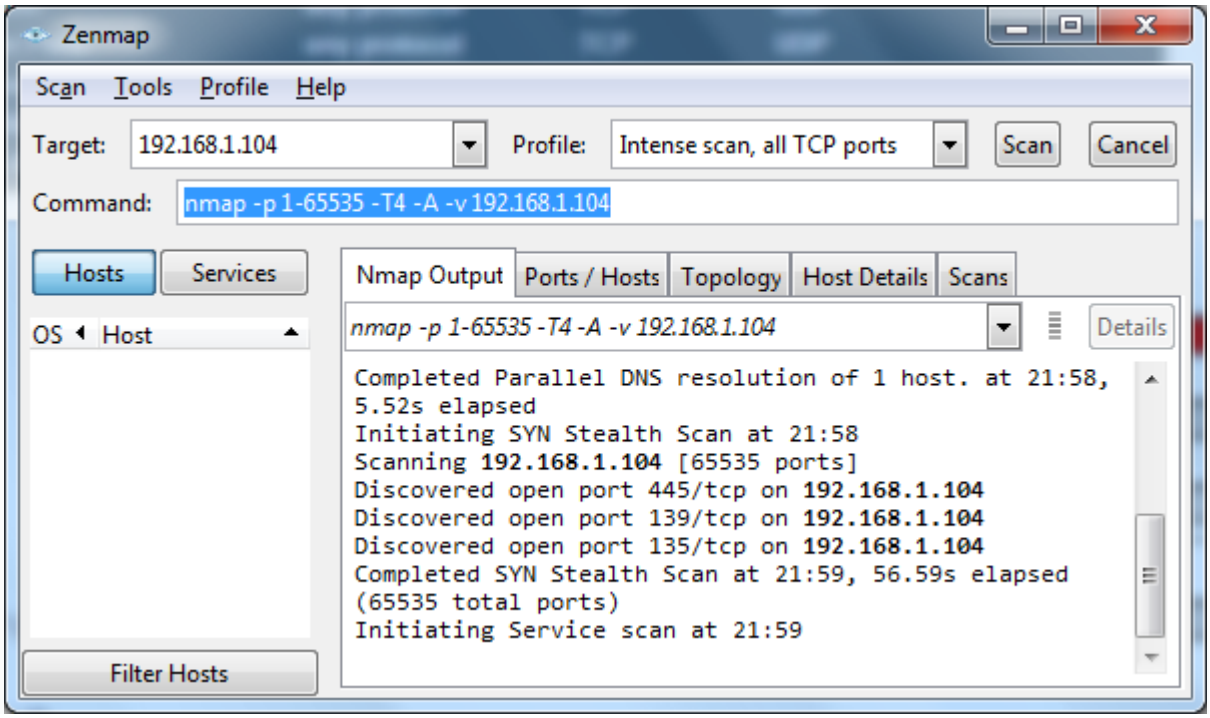


Εικόνα 35: Σχήμα επίθεσης TCPκαι UDPPortScan

Χρησιμοποιήσαμε το Zenmap δηλαδή το nmap σε γραφικό περιβάλλον και εκτελέσαμε την παρακάτω εντολή για να κάνουμε TCP PortScan και να βρούμε διάφορες πληροφορίες για το θύμα:

nmap -p 1-65535 -T4 -A -v 192.168.1.104

- **-p 1-65535:** Να σαρώσει τις πόρτες από 1-65535 του θύματος
- **-T4 :** Ταχύτητα με την οποία θα γίνεται η σάρωση
- **-A :** Να βρεί πληροφορίες όπως το λογισμικό που τρέχει,το όνομα του , traceroutek.τ.λ.
- **-v:** Increase verbosity level (use -vv or more for greater effect)
- **192.168.1.104:** Η IP του Host που θέλουμε να σκανάρουμε

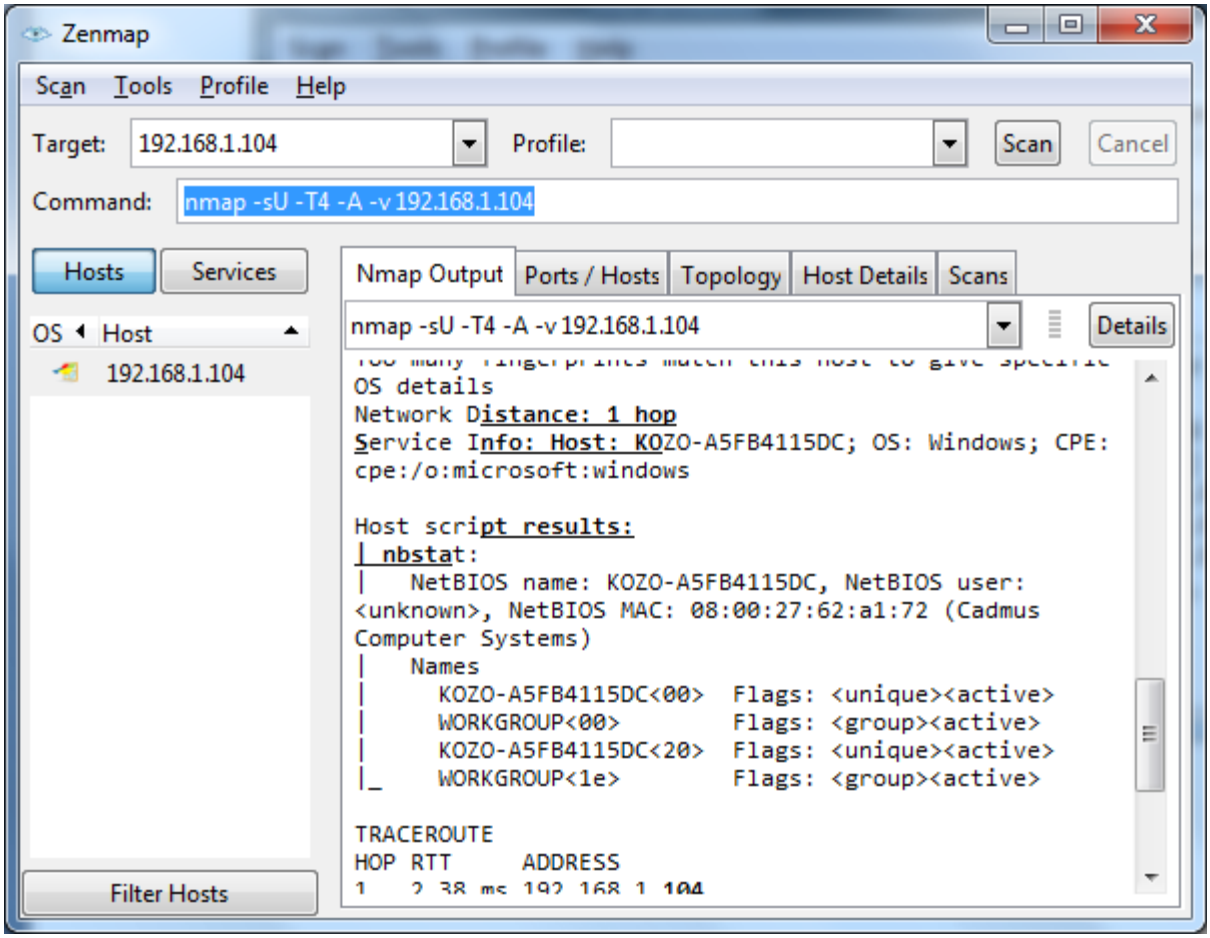


Εικόνα 36: TCPPortScanμε το Zenmap

Επίσης θα εκτελέσουμε και UDP PortScan με την εντολή:

nmap -sU -T4 -A -v 192.168.1.104

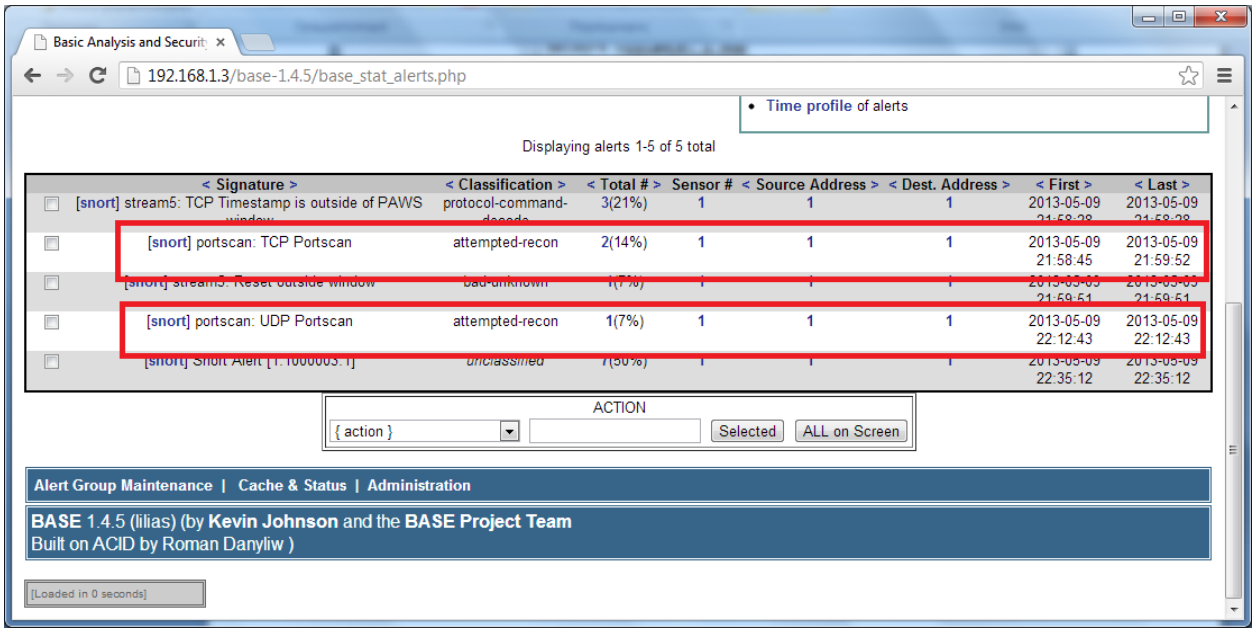
1. **-sU :** Θα στέλνει UDPπακέτα για να κάνει σάρωση
2. **-T4 :** Η σάρωση θα γίνεται με αρκετά μεγάλη ταχύτητα
3. **-A :** Να βρεί πληροφορίες όπως το λογισμικό που τρέχει,το όνομα του , traceroute κ.τ.λ.
4. **-v :** Increase verbosity level (use -vv or more for greater effect)
5. **192.168.1.104:** Η IP του Host που θέλουμε να σκανάρουμε



Εικόνα 37: UDPPortScanμε το Zenmap

Αφού έχουμε εκτελέσει αυτές τις σαρώσεις τώρα είμαστε σε θέση να τις ανιχνεύσουμε. Τα Alerts που έχουν πυροδοτιθεί από τον προεπεξεργαστή sfPortscan έχουν καταγραφεί στο σκληρό δίσκο πλέον και όχι στην κονσόλα. Επίσης το Barnyard2 έχει διαβάσει αυτά τα αρχεία και έχει περάσει τα δεδομένα στην βάση δεδομένων. Οπότε αν ανοίξουμε το BASE που συνδέεται με την βάση θα βρούμε Alerts για TCP και UDP PortScan.

Στην παρακάτο εικόνα φαίνονται στο BASE τα Alert που πυροδοτίθηκαν από τις δύο σαρώσεις θυρών που εκτελέσαμε:



Ποιο συγκεκριμένα έχουμε 2 Alertγια TCP PortScan που αναφαίρεται το ID , η υπογραφή που περιγράφει την επίθεση, η ημερομηνία και ώρα, η διεύθηνση ipαποστολέα και παραλήπτη.

ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >
#0-(4-6)	[snort] portscan: TCP Portscan	2013-05-09 21:59:52	192.168.1.2	192.168.1.104
#1-(4-4)	[snort] portscan: TCP Portscan	2013-05-09 21:58:45	192.168.1.2	192.168.1.104

Παρομοίος θα δουμε και πληροφορίες για το UDPPortScanalert.

ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >
#0-(4-7)	[snort] portscan: UDP Portscan	2013-05-09 22:12:43	192.168.1.2	192.168.1.104

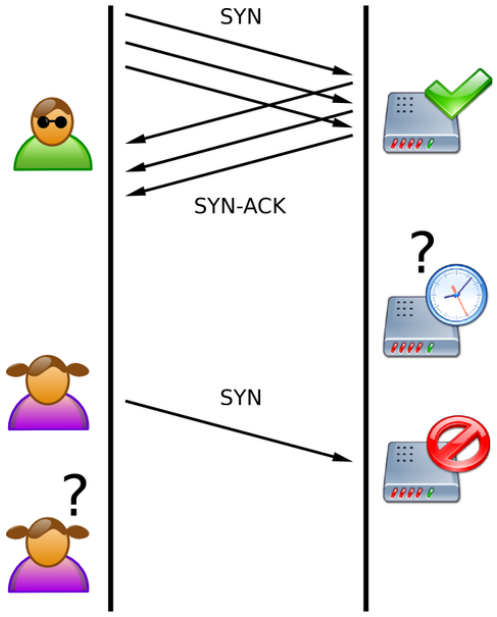
4.2 Ανίχνευση SYN flood με το Snort

Περιγραφή επίθεσης

Η επίθεση SYN flood είναι ένα είδος επίθεσης άρνησης πρόσβασης (DOS - Denial of Service) κατά την οποία ο επιτιθέμενος αποστέλλει πολλαπλές αιτήσεις SYN προς το θύμα. Αυτή η επίθεση βασίζετε στη τριμερή χειραψία TCP μεταξύ client και server.

Ο επιτιθέμενος αποστέλλει στον διακομιστή-θύμα πολλαπλά πακέτα TCPSYN. Ο διακομιστής θεωρεί ότι τα πακέτα αυτά προέρχονται από κανονικό χρήστη, οπότε απαντά με πακέτα SYN-ACK σύμφωνα με την διαδικασία χειραψίας του πρωτοκόλλου TCP. Ο επιτιθέμενος όμως δεν αποστέλλει πακέτα ACK για να ολοκληρωθεί η χειραψία, αλλά αφήνει τον διακομιστή να περιμένει. Επειδή για κάθε ημιτελή σύνδεση TCP ο διακομιστής ξοδεύει υπολογιστικούς πόρους, μετά από κάποιο συγκεκριμένο αριθμό τέτοιων συνδέσεων ο διακομιστής φτάνει στα όριά του και δεν μπορεί να εξυπηρετήσει τους νόμιμους χρήστες.

Στην παρακάτω εικόνα φαίνετε σχηματικά ο επιτιθέμενος και ο νόμιμος χρήστης. Αρχικά ο επιτιθέμενος στέλνει μια πλημύρα από πακέτα SYN με αποτέλεσμα ο πάροχος να μην μπορεί να απαντήσει στο αίτημα του νόμιμου χρήστη για σύνδεση.



Εικόνα 38: SYN flood

Σε αυτό το παράδειγμα η επίθεση πραγματοποιήθηκε εσωτερικά σε τοπικό δίκτυο

Ρύθμιση snort

Το Snort πρέπει να το ρυθμίσουμε όπως την προηγούμενη επίθεση που περιγράψαμε παραπάνω. Ανοίγουμε το ruleset με όνομα dos.rules και ελέγχουμε αν υπάρχουν κανόνες για ανίχνευση SYN flood. Να σημειώσουμε ότι αυτό το ruleset περιέχει κανόνες για ανίχνευση επιθέσεων Άρνησης Υπηρεσιών (DoS Attacks). Επίσης καλό θα ήταν να ελέγξουμε και το ruleset με όνομα ddos.rules που είναι για Κατανεμημένες επιθέσεις Άρνησης Υπηρεσιών (Distributed-DoSattacks).

Αν δεν υπάρχει ο κανόνας θα πρέπει να τον γράψουμε ή να τον βρούμε και να τον προσθέσουμε στο ruleset. Ο κανόνας για ανίχνευση SYN flood από πακέτα TCP είναι:

alert tcp any any -> $HOME_NET any (flags:S; threshold: type threshold, track by_dst, count 20, seconds 3; msg:"DDoS SYN flood attack detected!";sid:12121;)

Εξήγηση κεφαλής (header) του κανόνα "**alert tcp any any -> $HOME_NET any**" :

- **alert:** Πυροδοτεί συναγερμό σε περίπτωση που ο κανόνας επαληθευτεί.
- **tcp:**Ο κανόνας ελέγχει τα TCP πακέτα.
- **any:**Η διεύθυνση αποστολέα να είναι οποιαδήποτε ip.
- **any:** Το πακέτο να προέρχεται από οποιαδήποτε πόρτα του αποστολέα (Δε μας ενδιαφέρει αυτό το πεδίο).
- **->:** Είναι τελεστής κατεύθυνσης πού ότι βρίσκεται αριστερά του αντιστοιχεί στη διεύθυνση IP και τη θύρα του αποστολέα, ενώ το ζεύγος δεξιά του τελεστή αντιστοιχεί στον παραλήπτη.
- **$HOME_NET:** Έλεγχει αν η διεύθυνση παραλήπτη του πακέτου ανήκει στο εσωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any:**Το πακέτο προορίζεται σε οποιαδήποτε πόρτα του παραλήπτη (Δε μας ενδιαφέρει αυτό το πεδίο).

Εξήγηση ρυθμίσεων (Option) του κανόνα **"(flags:S; threshold: type threshold, track by_dst, count 20, seconds 3; msg:"DDoS SYN flood attack detected!";sid:12121;)**"

- **flags:S; :** Έλεγχος αν στην επικεφαλίδα των πακέτων το πεδίο flageίναι "S".
- **threshold: type threshold, track by_dust, count 20, seconds 3; :** Ο συναγερμός πυροδοτείτε όποτε αυτό το όριο ξεπερνιέται, προς μία μοναδική ipπαραλήπτη και αυτά τα πακέτα είναι 20, σε 3 δευτερόλεπτα.
- **msg:"DDoS SYN flood attackdetected!"; :** Το μήνυμα που θα εμφανίζεται κατά την πυροδότηση του συναγερμού για να μας ενημερώσει θα είναι "DDoS SYN flood attack detected!".
- **sid:12121; :** Το αναγνωριστικό του κανόνα για να ξεχωρίζει από τους υπόλοιπους είναι "12121".

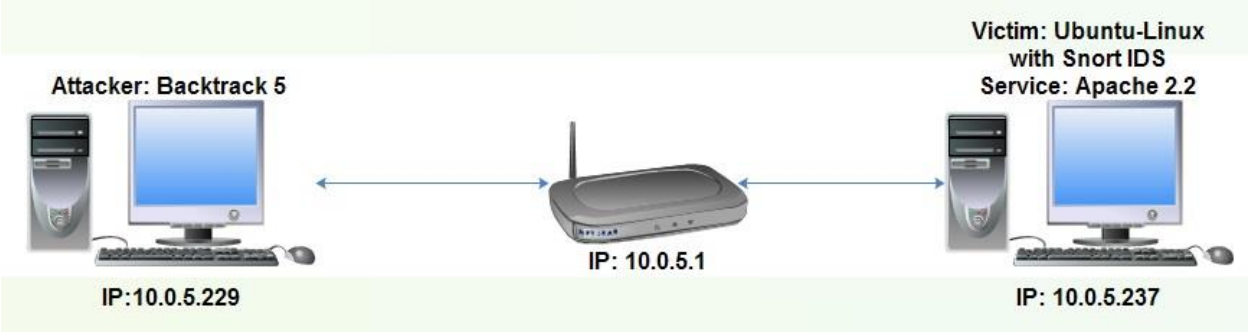
Αφού κάνουμε τις ρυθμίσεις τώρα μπορούμε να τρέξουμε το Snort με την εντολή:

sudo /usr/local/snort/bin/snort -c /usr/local/snort/etc/snort.conf -i wlan0 -A console

- **-c** /usr/local/snort/etc/snort.conf: Προσδιορίζουμε το configuration file που θα χρησιμοποιήσει.
- **-i** wlan0: Επιλέγουμε την ασύρματη κάρτα για να πιάσουμε τα πακέτα.
- **-A** console: Επιλέγουμε τα alerts να τα δούμε στην κονσόλα.
-

Εκτέλεση επίθεσης

Η επίθεση εκτελέστικε εσωτερικά σε ασύρματο τοπικό δίκτυο όπως φαίνετε στο παρακάτω σχήμα:



Εικόνα 39: Σχήμα επίθεσης Syn Flood

Για να εκτελέσουμε την επίθεση θα χρησιμοποιήσουμε hping3 από το τερματικό του Backtrack 5. Σκοπός μας είναι να στείλουμε μια πλημύρα από πακέτα SYN στον ApacheServer του θύματος, με ip 192.168.1.3. Η εντολή που θα δώσουμε στο τερματικό για να στείλουμε πλημύρα από τέτοια πακέτα είναι:

hping3 --flood -S -p 80 192.168.1.3

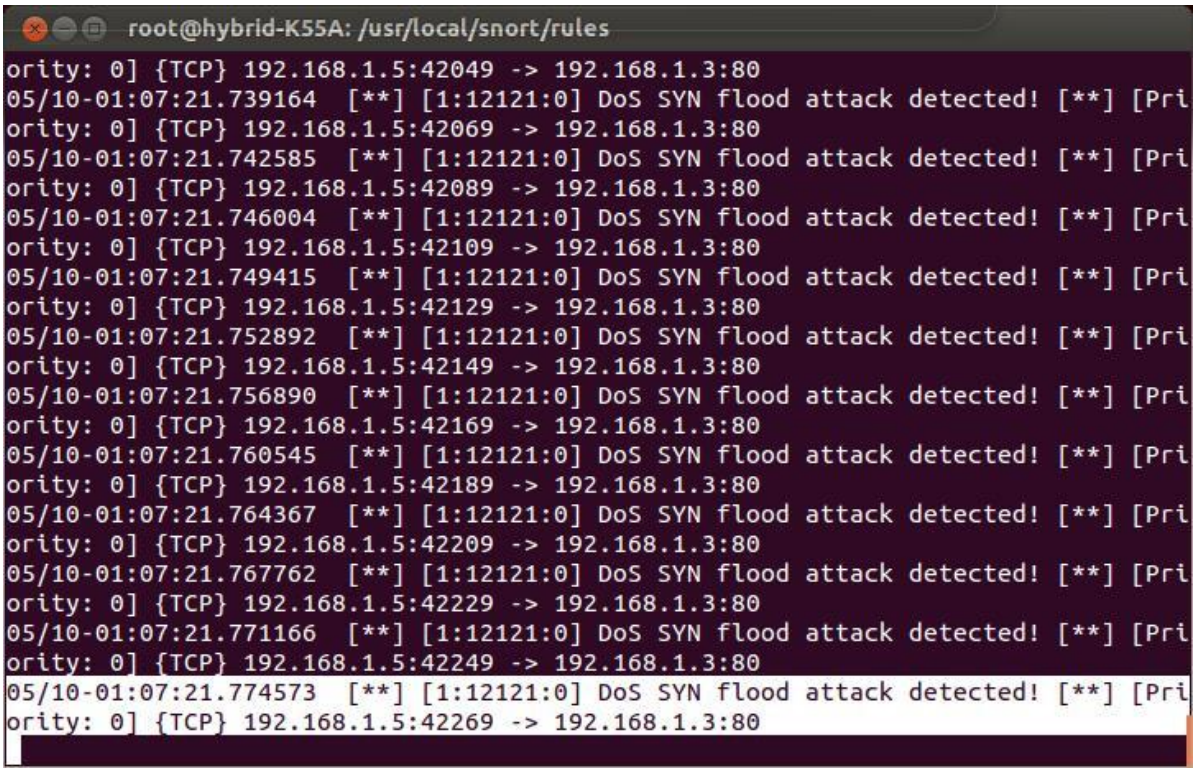
- --flood : θα στέλνει όσα ποιό πολλά πακέτα μπορεί η κάρτα δικτύου
- -S : Το TCP πακέτο να έχει flag SYN
- -p 80 : Τα πακέτα να στέλνονται στη θύρα 80

Εικόνα 40: Εκτέλεση SYN Flood στην πόρτα 80 του 10.0.5.237

Για 5 δευτερόλεπτα περίπου που τρέξαμε αυτή την επίθεση στάλθηκαν247582 αιτήσεις για σύνδεση. Τι θα γινόταν αν υπήρχαν πολλοί υπολογιστές, που θα έκαναν την ίδια επίθεση;

Ανίχνευση επίθεσης

Όπως καταλαβαίνουμε το Snort θα παράγει ένα μεγάλο αριθμό από Alerts όπως την παρακάτω εικόνα.



Εικόνα 41: Ανίχνευση SYN flood

Τα alerts θα είναι όπως το παρακάτω και θα αναφέρει ημερομηνία, ώρα, αναγνωριστικό κανόνα, μήνυμα είδους επίθεσης, προτεραιότητα και διευθύνσεις αποστολέα και παραλήπτη με τις αντίστοιχες πόρτες.

04/03-18:49:50.283221 [**] [1:12121:0] DoS SYN flood attack detected! [**] [Priority: 0] {TCP} 192.168.1.5:42269 -> 192.168.1.3:80

4.3Ανίχνευση UDP Flood

Περιγραφή επίθεσης

Η επίθεση UDP flood (UDP flood attack) είναι μία υποπερίπτωση των επιθέσεων άρνησης υπηρεσιών (Denial of Service - DOS) στην οποία χρησιμοποιούνται πακέτα UDP. Η αντίστοιχη μορφή της είναι η TCP Flood την οποία περιγράψαμε πριν.

Μία επίθεση UDP flood περιλαμβάνει την αποστολή ενός πολύ μεγάλου αριθμού UDP πακέτων σε τυχαίες πόρτες ενός υπολογιστή. Ο υπολογιστής που δέχεται την επίθεση θα πρέπει αρχικά να διαπιστώσει εάν κάποια από τις υπηρεσίες του ακούει στην συγκεκριμένη πόρτα και εάν δεν ακούει να απαντήσει με ένα πακέτο ICMP Destination Unreachable. Άρα λοιπόν, η εισροή μεγάλου αριθμού UDP πακέτων στον υπολογιστή που υφίσταται την επίθεση τον αναγκάζει να απαντήσει με εξίσου μεγάλο αριθμό πακέτων ICMP, γεγονός που τελικά εμποδίζει άλλους απλούς χρήστες από το να χρησιμοποιήσουν τις υπηρεσίες του υπό επίθεση υπολογιστή.

Σε αυτό το σενάριο θα χρησιμοποιήσουμε ένα host, ο οποίος θα στέλνει πλημύρα απο udp πακέτα στον τοπικό host που προστατεύεται απο το Snort.

Ρύθμιση snort

Η ρύθμιση του Snort είναι απλή σε αυτή την περίπτωση, διότι απλά πρέπει να το ρυθμίσουμε να πυροδοτεί συναγερμούς, σε περίπτωση που βλέπει μεγάλο αριθμό απο UDP πακέτα εισέρχονται στον Host χωρίς λόγο.

Έτσι απλά πρέπει να ενεργοποιήσουμε ή να φτιάξουμε ένα κανόνα στο ruleset dos.rules και να ενεργοποιήσουμε αυτό το ruleset απο το αρχείο ρυθμίσεων του snort. Ο κανόνας θα πρέπει να ελέγχει αν κάποιος host δέχεται υπερβολικά μεγάλο αριθμό απο UDP πακέτα απο ένα άλλο host. Στο σενάριο αυτό φτιάξαμε ένα κανόνα που αν ο host μας δεχτεί πάνω απο 10.000 πακέτα UDP σε 5 δευτερόλεπτα τότε να να πυροδοτείτε alert. Ο κανόνας που χρησιμοποιήσαμε είναι ο παρακάτω:

alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"UDP flooding"; threshold: type threshold, track by_src, count 10000, seconds 5; sid: 10000002; rev: 1;)

Εξήγηση της κεφαλής (header) του κανόνα "**alert udp $EXTERNAL_NET any -> $HOME_NET any**":

- **alert:** Πυροδοτεί συναγερμό σε περίπτωση που ο κανόνας επαληθευτεί
- **udp:**Ο κανόνας ελέγχει τα udp πακέτα
- **$EXTERNAL_NET:** Έλεγχει αν η διεύθυνση αποστολέα του πακέτου ανήκει στο εξωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any:** Το πακέτο να προέρχεται από οποιαδήποτε πόρτα του αποστολέα (Δε μας ενδιαφέρει αυτό το πεδίο)
- **->:** Είναι τελεστής κατεύθυνσης πού ότι βρίσκεται αριστερά του αντιστοιχεί στη διεύθυνση IP και τη θύρα του αποστολέα, ενώ το ζεύγος δεξιά του τελεστή αντιστοιχεί στον παραλήπτη.
- **$HOME_NET:**Ελέγχει αν η διεύθυνση παραλήπτη του πακέτου ανήκει στο εσωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort.
- **any:**Το πακέτο προορίζεται σε οποιαδήποτε πόρτα του παραλήπτη (Δε μας ενδιαφέρει αυτό το πεδίο).

Εξήγηση των ρυθμίσεων (Options) του κανόνα **"(msg:"UDP flooding"; threshold: type threshold, track by_src, count 10000, seconds 5; sid: 10000002; rev: 1;)**".

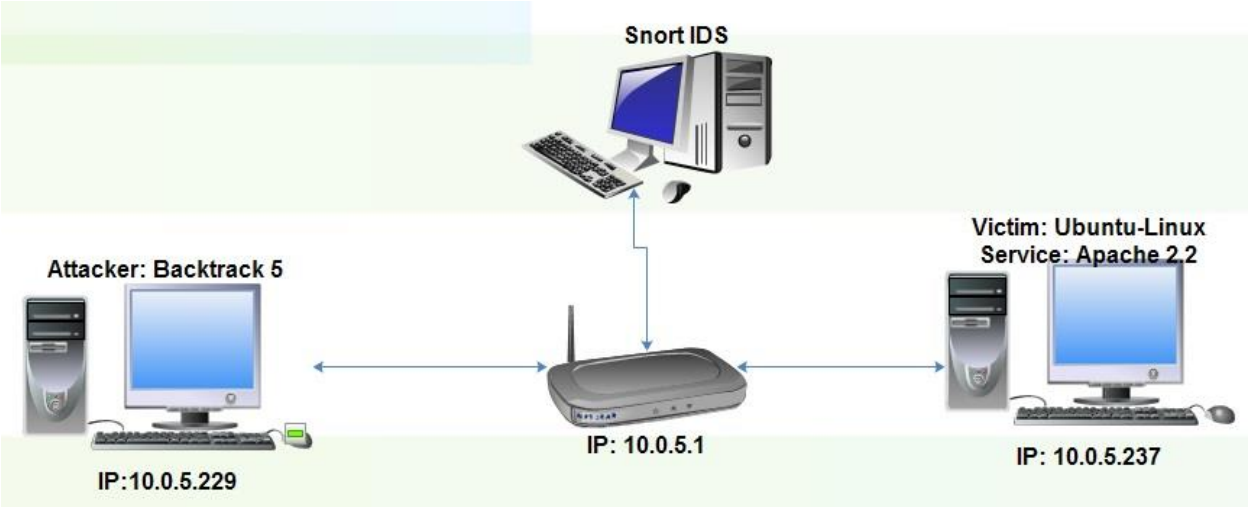
- **msg:"UDP flooding"; :** Το μήνυμα που θα εμφανίζεται κατά την πυροδότηση του συναγερμού για να μας ενημερώσει θα είναι "UDP flooding".
- **threshold: typethreshold, trackby_src, count 10000, seconds 5; :** Ο συναγερμός πυροδοτείτε όποτε αυτό το όριο ξεπερνιέται, από μία μοναδική ipαποστολέα και αυτά τα πακέτα είναι 10000, σε 5 δευτερόλεπτα.
- **sid:100000002; :** Το αναγνωριστικό του κανόνα για να ξεχωρίζει από τους υπόλοιπους είναι "10000002".
- **rev:1; :** Αριθμός έκδοσης του κανόνα είναι 1.

Αφού το ρυθμίσουμε το τρέχουμε το Snort με την παρακάτω εντολή, ως IDS και περιμένουμε να ανιχνεύσει πιθανές επιθέσεις.

sudo /usr/local/snort/bin/snort -c /usr/local/snort/etc/snort.conf -i wlan0 -A console

Εκτέλεση επίθεσης

Σε αυτό το σενάριο η επίθεση εκτελέστηκε εσωτερικά στο ασύρματο τοπικό δίκτυο όπως φαίνετε στο παρακάτω σχήμα:

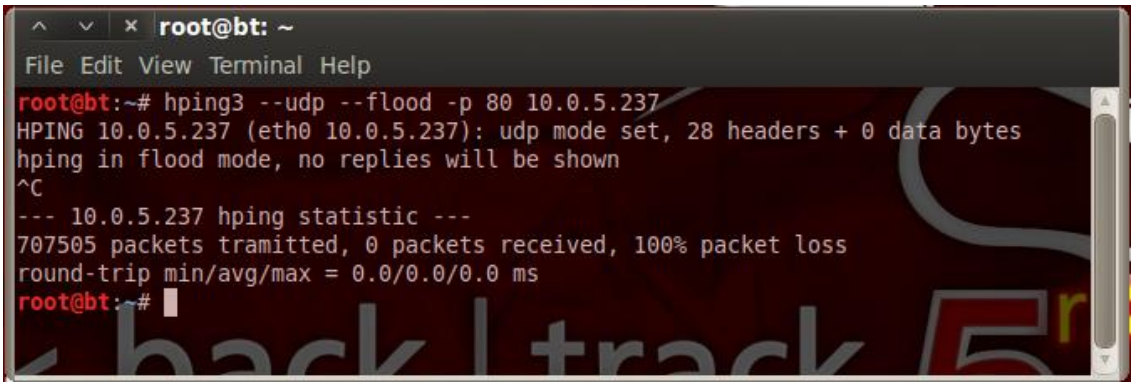


Εικόνα 42: Σχήμα επίθεσης UDP Flood

Για να εκτελέσουμε την επίθεση χρησιμοποιήσαμε το εργαλείοhping3, που υπάρχει εγκατεστημένο στο Backtrack 5 R3. Η εντολή που δώσαμε στο τερματικό για να εκτελεστεί η επίθεση προς τον host μας είναι η παρακάτω.

hping3 --udp --flood -p 80 10.0.5.237

- --udp : Θα στέλνει udp πακέτα
- --flood : Θα στέλνει όσο ποιό πολλά γίνεται
- -p 80 10.0.5.237: Τα πακέτα θα στέλνονται στην θύρα 80 του θύματος με ip 237



Εικόνα 43: Εκτέλεση επίθεσης UDP Flood

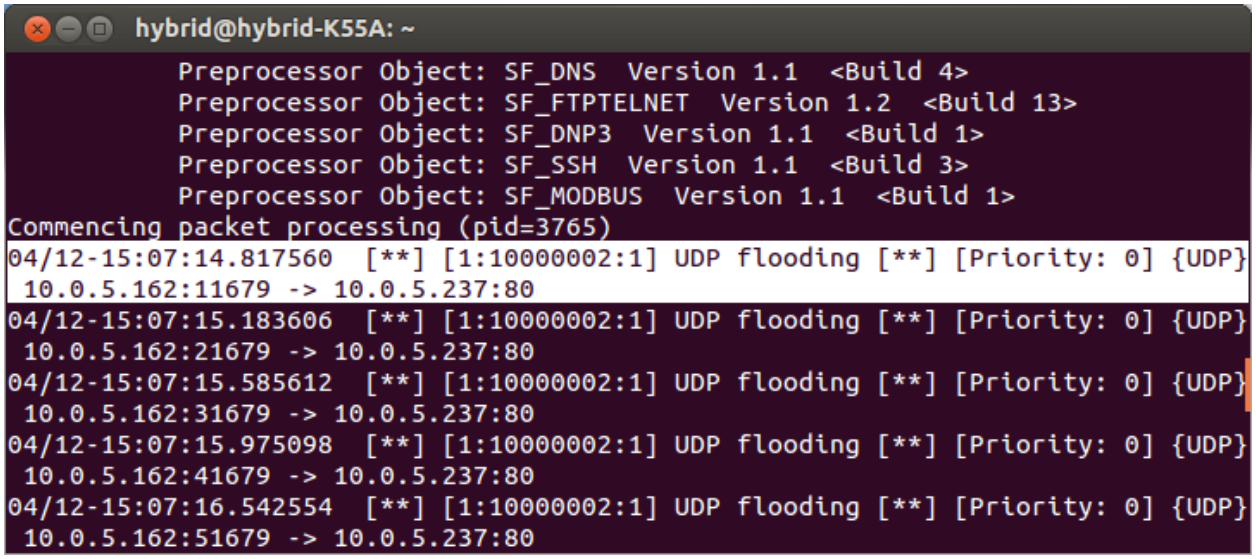
Ανίχνευση επίθεσης

Μετά την εκτέλεση της παραπάνω επίθεσης το Snort Θα αρχίσει να παράγει μία σειρά απο συναγερμούς για UDP flooding. Σύμφωνα με τον κανόνα που φτιάξαμε το alert θα είναι έτσι:

04/12-15:07:14.817560 [**] [1:10000002:1] UDP flooding [**] [Priority: 0] {UDP} 10.0.5.162:11679 -> 10.0.5.237:80

Βλέπουμε ότι ο Attacker Host με IP 10.0.5.162 απο την πόρτα 11679 κάνει επίθεση UDP flooding, προς τον Host με IP 10.0.5.162 στην πόρτα 80.

Στην εικόνα βλέπουμε τα alerts που εμφανίστηκαν στην κονσόλα.



Εικόνα 44: Ανίχνευση επίθεσης UDP Flood με το Snort

4.4 Ανίχνευση ARP Spoofing

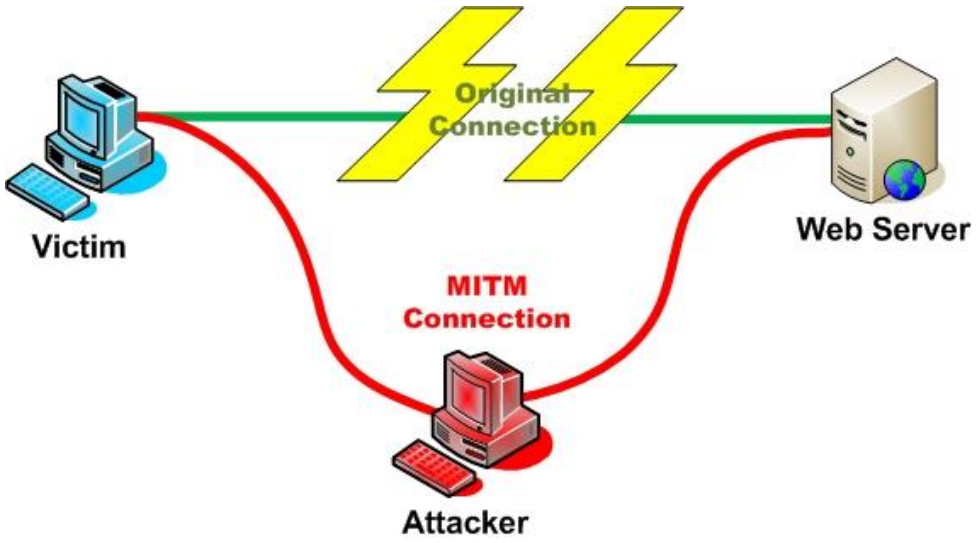
Περιγραφή επίθεσης

Το ARP spoofing (πλαστοπροσωπία ARP) ή ARP poisoning (δηλητηρίαση ARP) είναι ένας τύπος παραβίασης σε δίκτυο υπολογιστών το οποίο βασίζεται στο πρωτόκολλο ARP.

Ο κακόβουλος χρήστης μπορεί, μεταδίδοντας λανθασμένα πακέτα ARP, να μπερδέψει άλλους host ώστε να στείλουν τα πλαίσια δεδομένων (data frames) τους σε άλλον υπολογιστή χωρίς να το αντιληφθούν. Μπορεί τότε να παρακολουθήσει την επικοινωνία μεταξύ (π.χ. ως βάση για μια επίθεση τύπου man-in-the-middle attack):

- Δύο Hosts
- Ενός Host και ενός υπόδικου
- Ενός Host και του Διαδικτύου
- Οποιοδήποτε συνδυασμό των παραπάνω παραβιάσεων

Στην παρούσα εργασία πραγματοποιήσαμε μια τέτοια επίθεση μέσα στο δίκτυο μεταξύ ενόςHost και της gateway. Όπως καταλαβαίνουμε, αν παρακολουθήσουμε μια τέτοια επικοινωνία είμαστε σε θέση να αντλήσουμε χρήσιμες πληροφορίες από τα πακέτα που ανταλλάσουν μεταξύ τους. Για να το πετύχουμε αυτό στέλνουμε ARP πακέτα στο Host με ip της gateway και mac address την δική μας. Επίσης στέλνουμε στην gateway ARP πακέτα με ip του Host και mac address την δική μας. Έτσι πετυχαίνουμε overwrite στους πίνακες ARP και τα πακέτα έρχονται στην δική μας mac address. Μετά με διάφορα εργαλεία καταγράφουμε αυτή την κίνηση και μπορούμε να αντλήσουμε χρήσιμες πληροφορίες όπως passwords, urls, εικόνες κ.τ.λ.



Εικόνα 45: Σχηματική απεικόνιση επίθεσης Man in the middle

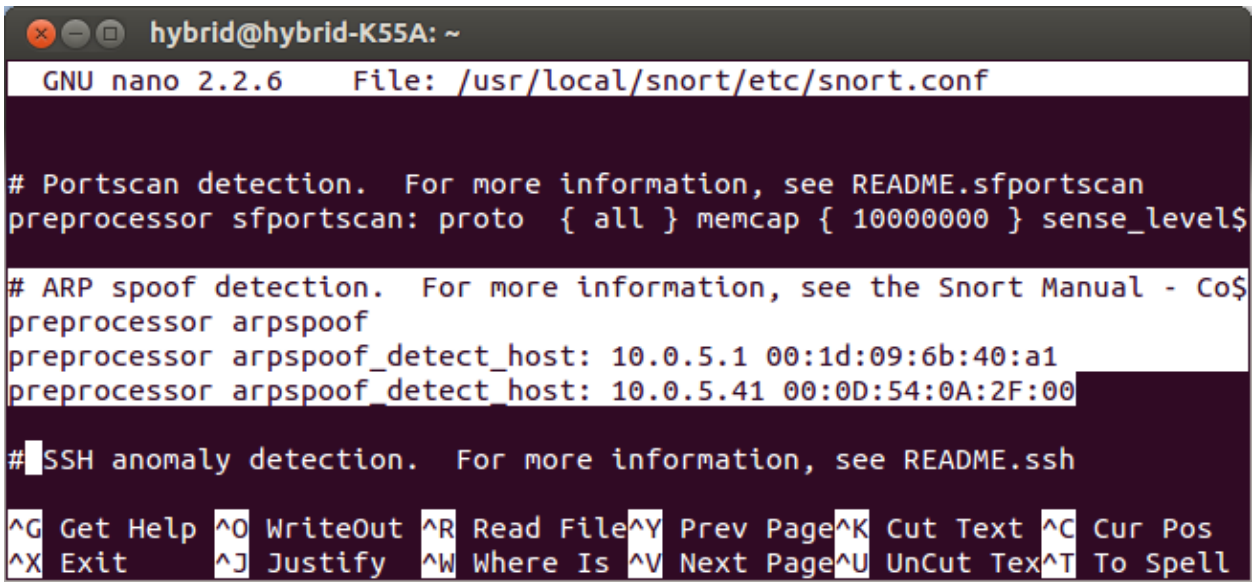
Ρύθμιση Snort

Το Snort είναι σε θέση να αναγνωρίσει μια τέτοια δραστηριότητα χάρη τον ARP Spoof Preprocessor που διαθέτει. Αυτός ο Προεπεξεργαστής αποκωδικοποιεί τα ARP πακέτα και ελέγχει τις διευθύνσεις ip και mac address με τις πραγματικές. Αν βρει διαφορές τότε πυροδοτεί ένα Alert με GID 112 και SID 1 ή 2 ή3 ή 4 ανάλογα με τι έχει ανιχνεύσει. Στην περίπτωση μας πρέπει να είναι 4 γιατί πραγματοποιούμε overwrite στους πίνακες ARP. Περισσότερες πληροφορίες στο user manual του Snort 2.9.4 .

Αυτό που πρέπει να κάνουμε είναι να ενεργοποιήσουμε και να ρυθμίσουμε αυτόν το Προεπεξεργαστή. Έτσι πρέπει να ανοίξουμε το αρχείο ρυθμίσεωνsnort.conf και να μεταβούμε στο step#5, όπου μπορούμε να ρυθμίσουμε και να ενεργοποιήσουμε τους Προεπεξεργαστές. Εκεί ενεργοποιούμε τον Προεπεξεργαστή preprocessor arp spoof σβήνοντας την # από το μπροστά μέρος και προσθέτουμε τον Host και την Gateway με την ip και mac address.

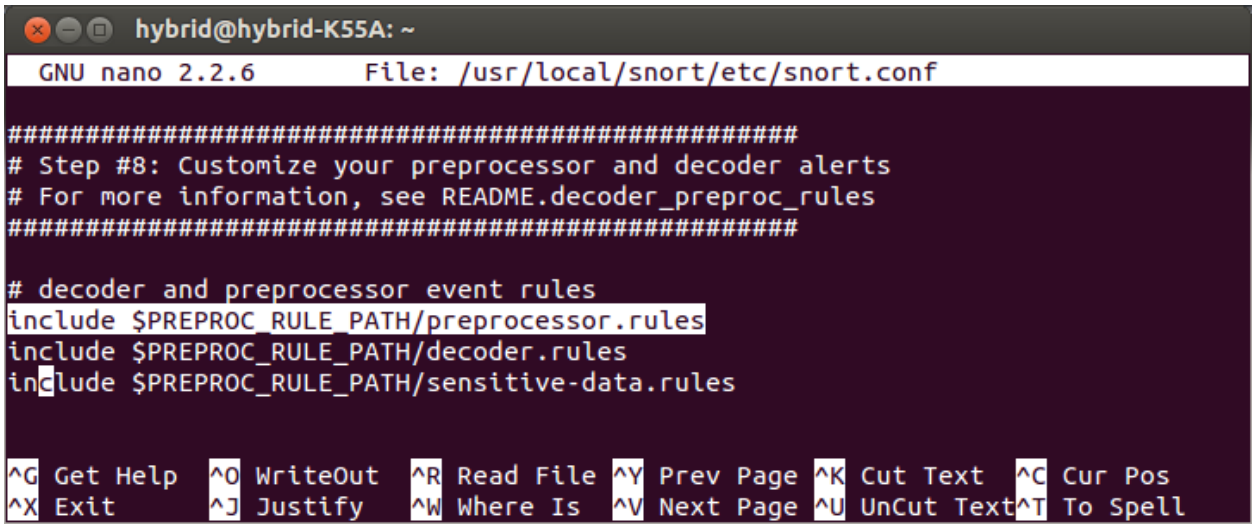
preprocessor arpspoof_detect_host: 10.0.5.1 00:1d:09:6b:40:a1

preprocessor arpspoof_detect_host: 10.0.5.41 00:0D:54:0A:2F:00



Εικόνα 46: Ενεργοποίηση του arp spoof Προεπεξεργαστή

Επίσης πρέπει να ενεργοποιήσουμε και τους κανόνες των Προεπεξεργαστών στο αρχείο ρυθμίσεων snort.conf στο step #8.



Εικόνα 47: Ενεργοποίηση κανόνων Προεπεξεργαστών

Ποιό συγκεκριμένα πρέπει να ελέγξουμε αν σε αυτό το ruleset υπάρχει κανόνας που να πυροδοτεί ένα Alert για ARP Spoofing. Έτσι ανοίγουμε αυτό το ruleset και ελέγχουμε για το παρακάτω Alert.

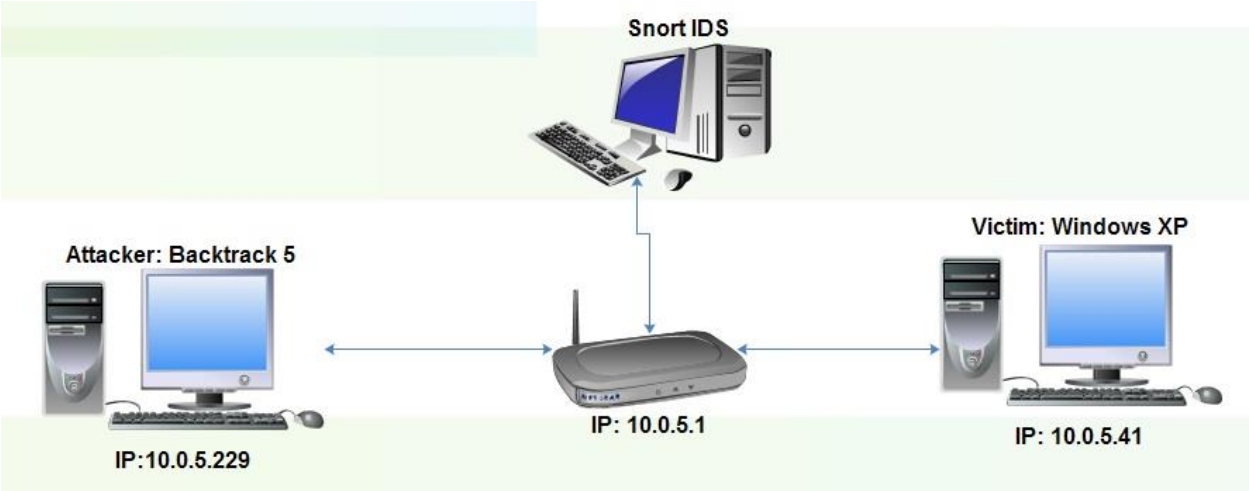
alert (msg: "ARPSPOOF_ARP_CACHE_OVERWRITE_ATTACK"; sid: 4; gid: 112; rev: 1; metadata: rule-type preproc ; classtype:bad-unknown;)

Ο κανόνας αυτός ανήκει στους κανόνες προεπεξεργαστών. Χρησιμοποιείτε μόνο για να πυροδοτήσει συναγερμό και να δώσει πληροφορίες για την επίθεση που ανιχνεύει ο προεπεξεργαστής ARP Spoof.

- **alert :**Ο κανόνας πυροδοτεί συναγερμό
- **msg:** "ARPSPOOF_ARP_CACHE_OVERWRITE_ATTACK"; : Ο συναγερμός θα περιέχει το μήνυμα "ARPSPOOF_ARP_CACHE_OVERWRITE_ATTACK"
- **sid: 4;** : Τοαναγνωριστικό του κανόνα θα είναι 4.
- **gid: 112;** : Ο κανόνας ενεργοποιείτε από τον προεπεξεργαστή ARP Spoof σύμφωνα με την τιμή 112 του gid.
- **rev: 1;**: Η έκδοση του κανόνα είναι πρώτη αφού η τιμή τουrevείναι 1.
- **metadata:**rule-type preproc ; : Ο κανόνας αυτός περιέχει επιπλέων πληροφορίες ότι είναι τύπου προεπεξεργαστή (preproc).
- **classtype: bad-unknown;** : Ο κανόνας ανήκει στην κατηγορία κανόνων bad-unknown

Εκτέλεση επίθεσης

Σε αυτό το σενάριο, η επίθεση πραγματοποιήθηκε στο τοπικό δίκτυο της βιβλιοθήκης του Τ.Ε.Ι. Κρήτης όπου πραγματοποιήσαμε ARP spoofing μεταξύ ενός Host και της Gateway του συγκεκριμένου υποδικτύου.



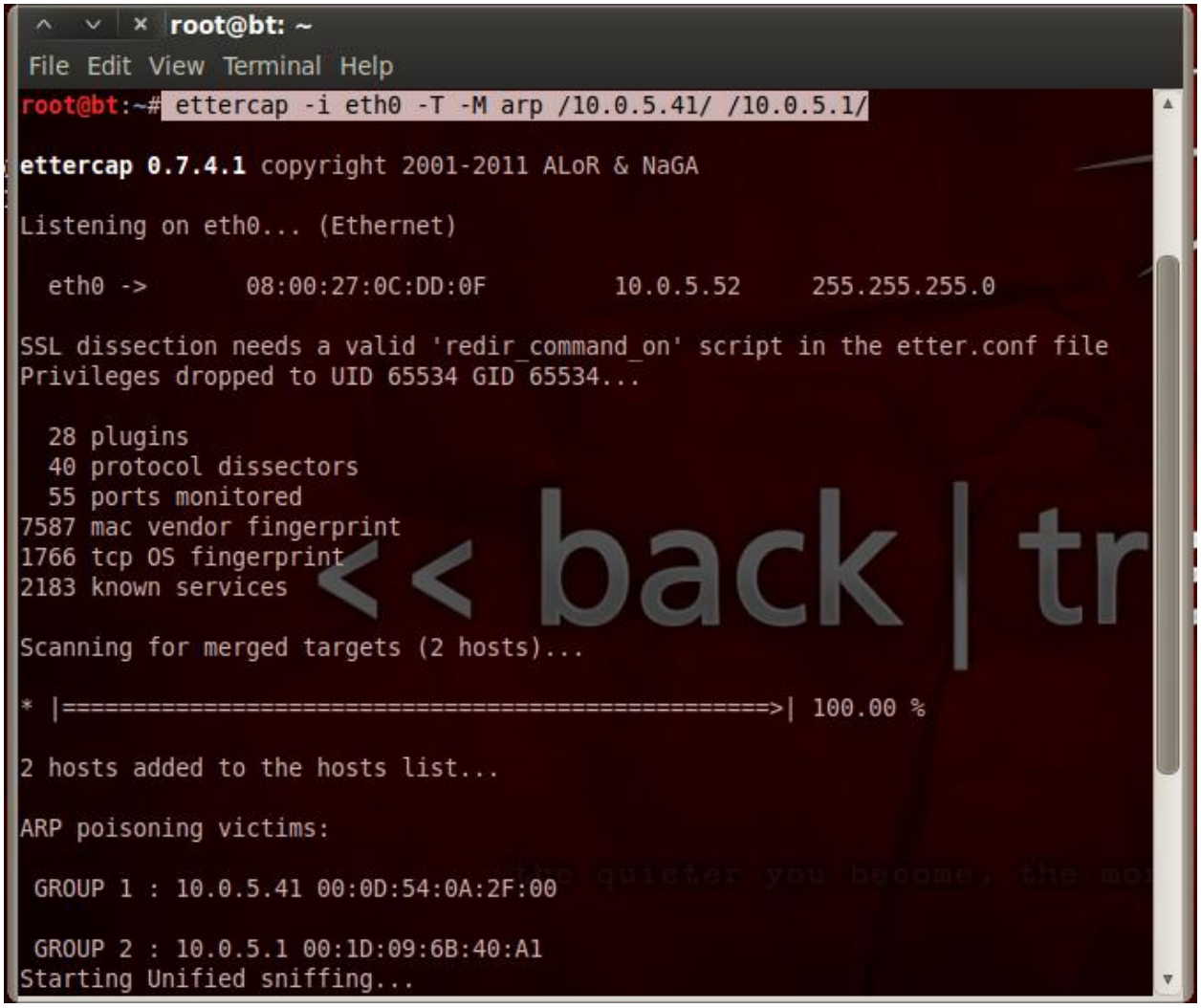
Εικόνα 48: Σχήμα επίθεσης ARP Spoofing

Όπως και στις προηγούμενες επιθέσεις έτσι και σε αυτή θα χρησιμοποιήσουμε το Backtrack 5 R3. Εκεί υπάρχει το χρήσιμο εργαλείο ettercap. Με αυτό το εργαλείο έχουμε την δυνατότητα να κάνουμε δηλητηρίαση της γραμμής με ARP Spoofing και να ελέγχουμε όλη την κίνηση σε Logαρχεία στο δίσκο.

Εκτελούμε την επίθεση γράφοντας στο τερματικό του Backtrack 5 την εντολή:

ettercap -i eth0 -T -M arp /10.0.5.41/ /10.0.5.1/

- **-ieth0** : Προσδιορίζουμε την κάρτα δικτύου που επικοινωνεί με το δίκτυο
- **-T**: Να τρέξει σε textmode το πρόγραμμα
- **-M arp**: Να κάνει Man-in-the-Middle Attack χρησιμοποιόντας τεχνική ARP Spoofing
- **/10.0.5.41/** : Διεύθυνση υπολογιστή στόχου
- **/10.0.5.1/** : Διεύθυνση της Gateway

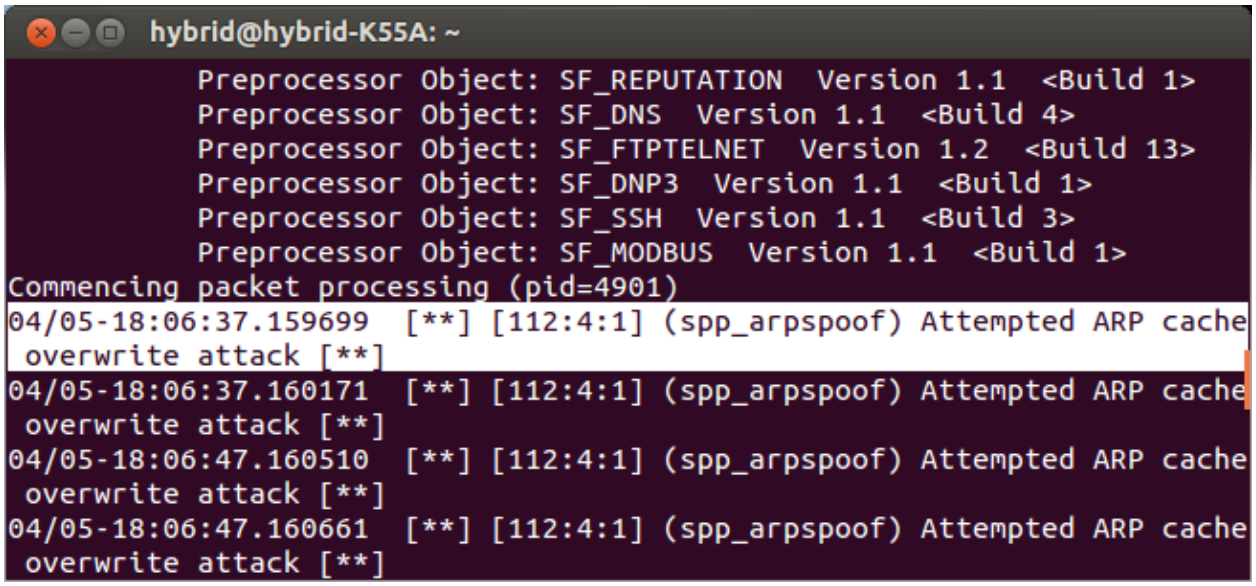


Εικόνα 49: Εκτέλεση επίθεσης ARPSpoofing (Δηλητηρίαση γραμμής)

Ανίχνευση επίθεσης

Αφού τρέξουμε την επίθεση και το Snort είναι συνδεδεμένο και ακούει όλη την κίνηση του δικτύου, σε κάθε overwrite που πραγματοποιείτε στους ARP πίνακες του Host και της Gateway θα πυροδοτείτε το παρακάτω Alert.

04/05-18:06:37.159699 [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]



Εικόνα 50: Ανίχνευση επίθεσης ARPSpoofing με το Snort

4.5 Ανίχνευση SubSeven Backdoor

Περιγραφή Επίθεσης

Backdoor σε ένα σύστημα υπολογιστή, είναι μια μέθοδος που παρακάμπτει την ταυτοποίηση και εξασφαλίζει παράνομη απομακρυσμένη σύνδεση σε ένα υπολογιστή. Μια τέτοια σύνδεση μπορεί να επιτρέπει σε ένα κακόβουλο χρήστη να έχει σε πλήρη έλεγχο τον υπολογιστή κάποιου άλλου και να κάνει στην κυριολεξία ότι θέλει.

Οι hackers χρησιμοποιούν αρκετά τέτοια προγράμματα ώστε να πετύχουν μια τέτοια σύνδεση, όπως το Back Orifice, το NetBus και το SubSeven. Αυτά τα προγράμματα αποτελούνται απο δύο κομμάτια, το Server και τον Client. Ο Server είναι το πρόγραμμα που στέλνετε στον υπολογιστή στόχο και μέσο αυτού θα επιτεχθεί η σύνδεση. Ο client είναι το πρόγραμμα που θα υπάρχει στον υπολογιστή του επιτιθέμενου και θα δίνει εντολές προς τον υπολογιστή στόχο.

Ο επιτιθέμενος αν καταφέρει να στείλει στο θύμα το μικρό προγραμματάκι και να μην το εντοπίσει ο χρήστης με κάποιο Antivirus ή κάποιο άλλο μηχανισμό ασφάλειας τότε είναι σε θέση να πάρει τον έλεγχο. Αμέσως αυτό το πρόγραμμα ανοίγει μια θύρα και ο επιτιθέμενος χρησιμοποιώντας την IP και την πόρτα κάνει TCP σύνδεση και στέλνει διάφορες εντολές που το πρόγραμμα στον υπολογιστή στόχο εκτελεί.

Τέτοιου είδους προγράμματα έχουν την δυνατότητα να εκτελέσουν αρκετές κλήσεις συστήματος όπως restart του υπολογιστή, διαγραφή αρχείων, άνοιγμα κάμερας, αλλαγή ρυθμίσεων λειτουργικού και ότι άλλο μπορεί να φανταστεί ο δημιουργός ενός τέτοιου προγράμματος.

Για αυτή την εισβολή χρησιμοποιήσαμε το SubSevenένα πρόγραμμα ειδικό για Backdoor. Επίσης χρησιμοποιήσαμε δύο υπολογιστές με Windows XP sp3 όπου ο ένας είναι επιτιθέμενος και ο άλλος το θύμα. Τέλος τον υπολογιστή με το Snort, που ελέγχει όλη την κίνηση μέσα στο δίκτυο.

Ρύθμιση Snort

Το Snort χρησιμοποιεί κανόνες για να ανιχνεύει προγράμματα που ανοίγουν Backdoors. Ελέγχει τα πακέτα και τις συνδέσεις μέσα στο δίκτυο και μπορεί να εντοπίσει την λειτουργία τέτοιων προγραμμάτων και να ενημερώσει τον διαχειριστή του δικτύου.

Έτσι κάνοντας μια μικρή μελέτη στους κανόνες, θα βρούμε κανόνες που έχουν την δυνατότητα να εντοπίσουν την λειτουργία του SubSeven μέσα στο δίκτυο. Το ruleset malware-cnc.rules περιέχει κανόνες για τέτοιου είδους προγράμματα όπως και τα ruleset backdoor.rules και malware-backdoor.rules.

Έτσι ανοίγοντας το αρχείο malware-cnc.rules τσεκάρουμε αν υπάρχει ο κανόνας:

alert tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"MALWARE-CNC SubSeven client connection to server"; flow:to_client,established; content:"connected."; nocase; content:"Legends"; distance:0; fast_pattern; nocase; pcre:"/^connected\x2e[^\x0D\x0A]*20\d\d[^\x0D\x0A]*ver\x3A\s+Legends \s2\x2e1/smi"; metadata:impact_flag red, policy balanced-ips drop, policy security-ips drop, service http; reference:url,www.threatexpert.com/report.aspx?md5=da8d7529a8a37335064ade9d04df08ad; classtype:trojan-activity; sid:15938; rev:6;)

Εξήγηση της κεφαλής (header) του κανόνα "**alert tcp $HOME_NET any -> $EXTERNAL_NET any**":

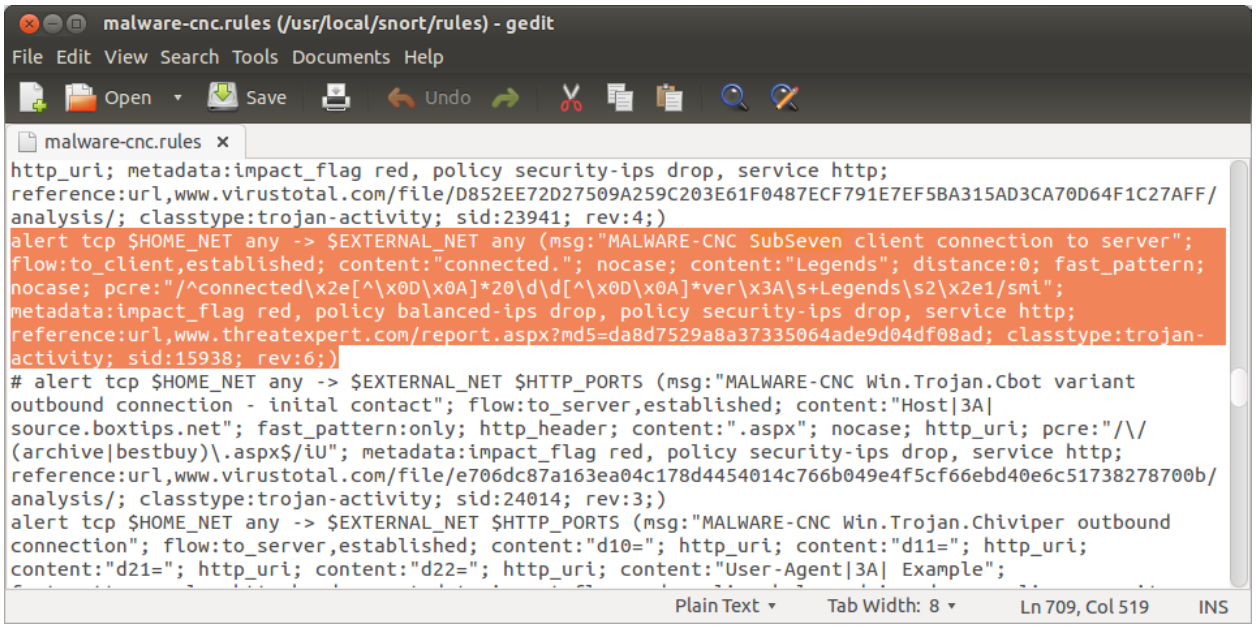
- **alert:** Πυροδοτεί συναγερμό σε περίπτωση που ο κανόνας επαληθευτεί
- **tcp:**Ο κανόνας ελέγχει τα TCP πακέτα
- **$HOME_NET:**Ελέγχει αν η διεύθυνση αποστολέα του πακέτου ανήκει στο εσωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any:** Το πακέτο να προέρχεται από οποιαδήποτε πόρτα του αποστολέα (Δε μας ενδιαφέρει αυτό το πεδίο)
- **->:** Είναι τελεστής κατεύθυνσης πού ότι βρίσκεται αριστερά του αντιστοιχεί στη διεύθυνση IP και τη θύρα του αποστολέα, ενώ το ζεύγος δεξιά του τελεστή αντιστοιχεί στον παραλήπτη.
- **$EXTERNAL_NET:**Ελέγχει αν η διεύθυνση παραλήπτη του πακέτου ανήκει στο εξωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any:**Το πακέτο προορίζεται σε οποιαδήποτε πόρτα του παραλήπτη (Δε μας ενδιαφέρει αυτό το πεδίο).

Εξήγηση ρυθμίσεων (Options) του κανόνα"msg:"MALWARE-CNC SubSeven client connection to server"; flow:to_client,established; content:"connected."; nocase; content:"Legends"; distance:0; fast_pattern; nocase; pcre:"/^connected\x2e[^\x0D\x0A]*20\d\d[^\x0D\x0A]*ver\x3A\s+Legends\s2\x2e1/smi"; metadata:impact_flag red, policy balanced-ips drop, policy security-ips drop, service http; reference:url,www.threatexpert.com/report.aspx?md5=da8d7529a8a37335064ade9d04df08ad; classtype:trojan-activity; sid:15938; rev:6;":

- **msg:"MALWARE-CNC SubSeven client connection to server";** : Όταν πυροδοτείτε Alert θα μας ενημερώνει για το είδος της επίθεσης με το μήνυμα "MALWARE-CNC SubSeven client connection to server".
- **flow:to_client,established;** : Για να ενεργοποιηθεί ο συναγερμός πρέπει να υπάρχει ροή TCPπακέτων προς τον Clientκαι η σύνδεση μεταξύ τους να έχει επιτεχθεί.
- **content:"connected.";** : Το περιεχόμενο του πακέτου να περιέχει την λέξη "connected".
- **nocase;** : Στην αναζήτηση των περιεχομένων του πακέτου να μην γίνεται διαχωρισμός κεφαλαίων και μικρών γραμμάτων.
- **content:"Legends";** : Το πακέτο επίσης πρέπει να περιέχει την λέξη "Legends".
- **distance:0;**: Με την τιμή '0' δηλώνουμε ότι δε μας ενδιαφέρει πόσα bytesθα απέχει το ένα περιεχόμενο από το άλλο μέσα στο πακέτο "connected" και "Legends" αντίστοιχα.
- **fast_pattern;** : Λέμε στο Snortνα ψάχνει πρώτα για το δεύτερο περιεχόμενο "Legends" γιατί είναι ποιο σπάνιο να βρεθεί από το πρώτο και μετά να ελέγξει και για το πρώτο.
- **nocase;**: Στην αναζήτηση των περιεχομένων του πακέτου να μην γίνεται διαχωρισμός κεφαλαίων και μικρών γραμμάτων.
- **pcre:"/^connected\x2e[^\x0D\x0A]*20\d\d[^\x0D\x0A]*ver\x3A\s+Legends\s2\x2e1/smi";** : Περιέχει κανόνα σε μορφή συμβατό με Perl
- **metadata: impact_flag red, policy balanced-ips drop, policy security-ips drop, service http;** : Εδώ έχουν ορίστεί κάποιες επιπλέων πληροφορίες για τον κανόνα όπως ότι
- **reference:url,www.threatexpert.com/report.aspx?md5=da8d7529a8a37335064ade9d04df08ad;**: Με την βοήθεια ενός κατάλληλου plugin το Snort συνδέεται με εξωτερικά συστήματα για

άντληση περισσότερης πληροφορίας για την επιπλέον προειδοποίηση και πληροφόρηση. Σύμφωνα με την ρύθμιση αυτή το Snortθα συνδεθεί με την ιστοσελίδα "http://www.threatexpert.com/report.aspx?md5=da8d7529a8a37335064ade9d04df08ad" για περισσότερες πληροφορίες προειδοποίησης.

- **classtype:trojan-activity;** : Κατατάσσουμε τον κανόνα στην κατηγορία "trojan-activity".
- **sid:15938**; : Ο κανόνας αυτός έχει αναγνωριστικό 15938 ανάμεσα στους υπόλοιπους κανόνες που χρησιμοποιούμε.
- **rev:6;**: Η έκδοση του κανόνα αυτού είναι η 6η .

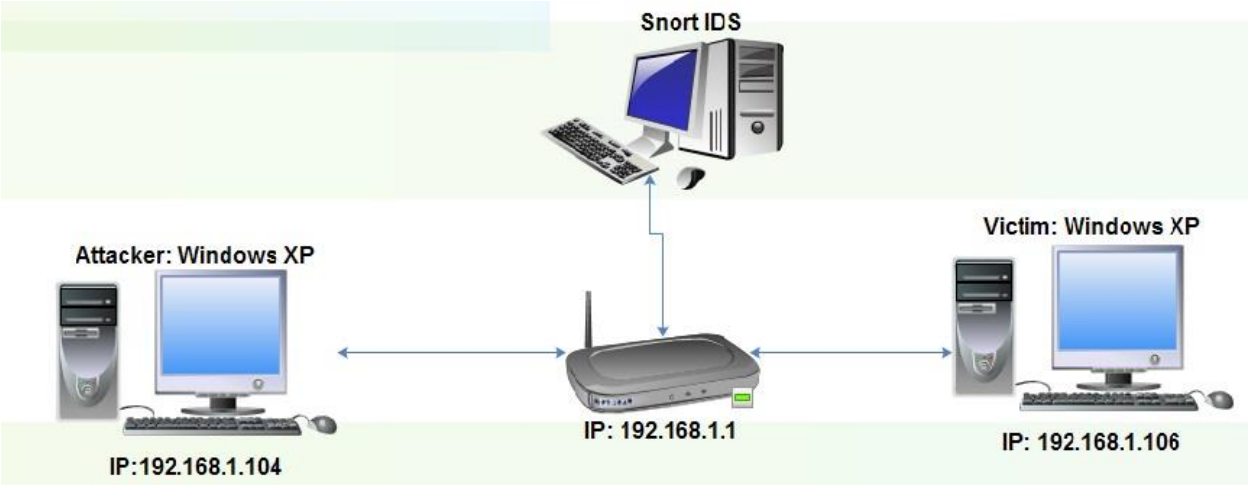


Εικόνα 51: Κανόνας ανίχνευσης Subseven Backdoor

Αφού ενεργοποιήσουμε τον κανόνα και το συγκεκριμένο ruleset απο το αρχείο ρυθμίσεων, είμαστε έτοιμοι να τρέξουμε το Snort και να ανιχνεύσουμε εισβολή απο το SubSeven.

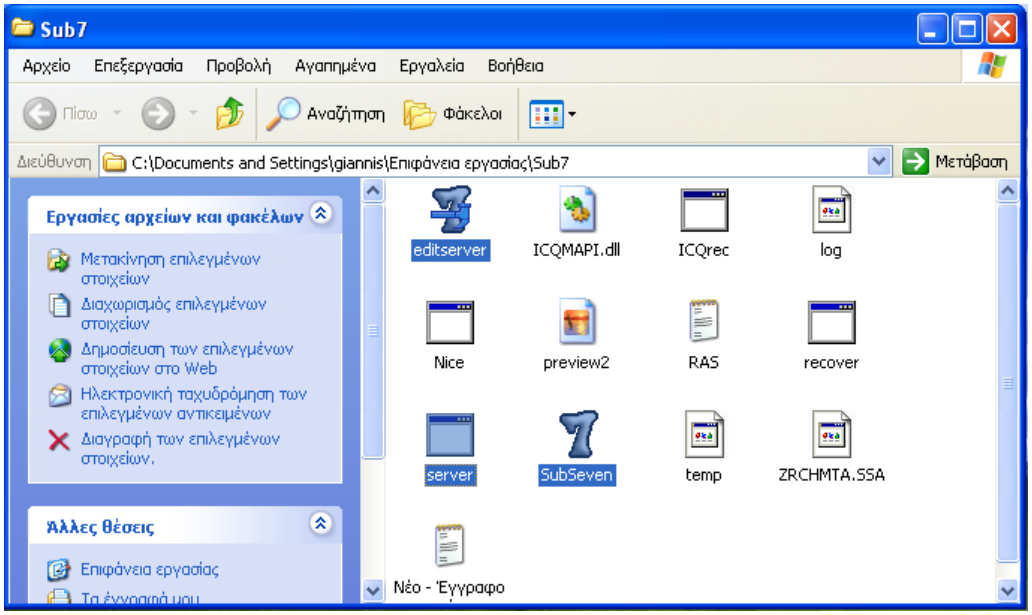
Εκτέλεση Επίθεσης

Σε αυτό το σενάριο η επίθεση ανιχνεύεται και γίνεται στο τοπικό δίκτυο όπως φαίνεται στο παρακάτω σχήμα:



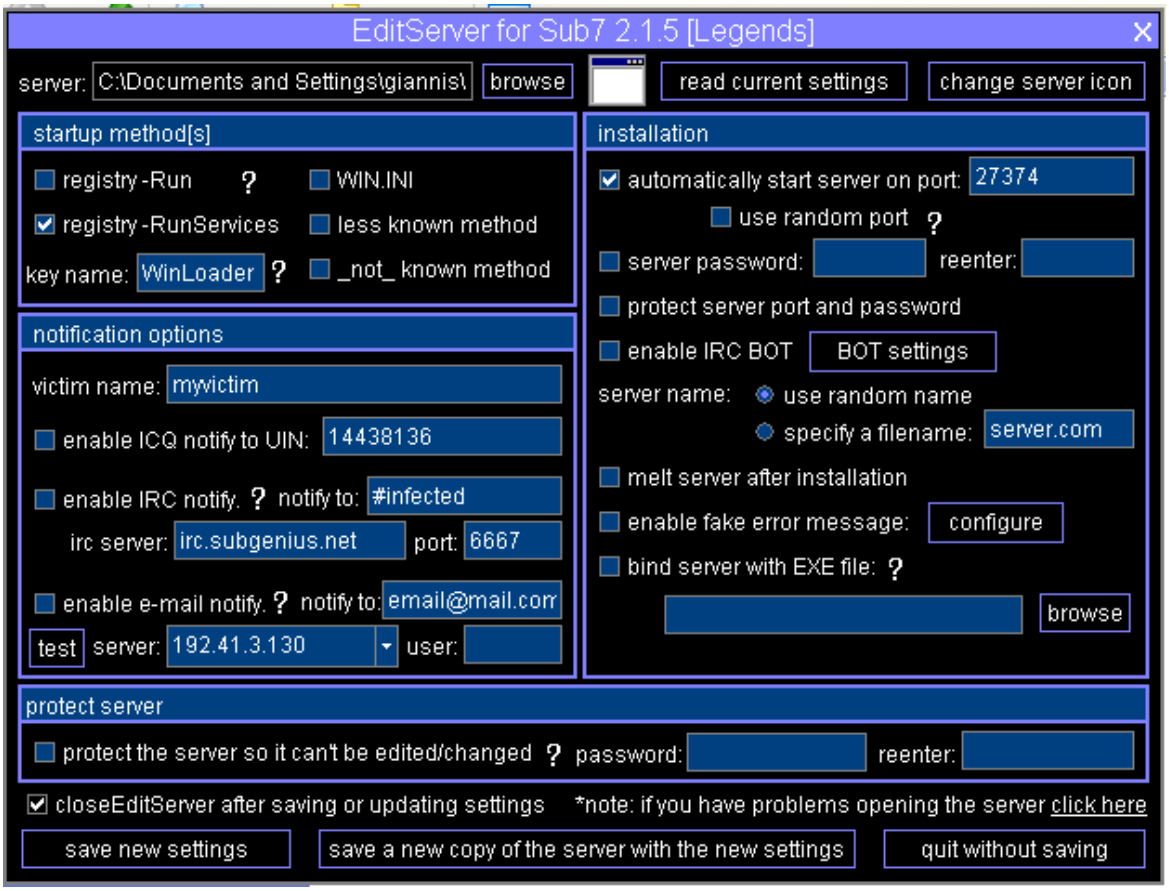
Εικόνα 52: Σχήμα επίθεσης Malware Suubseven Backdoor

Για την εκτέλεση αυτής της επίθεσης χρησιμοποιήσαμε το SubSeven 2.1.5. Το κατεβάσαμε στον υπολογιστή με Windows xp sp3 σε συμπιεσμένη μορφή και μετά την αποσυμπίεση σε ένα φάκελο βρήκαμε τα παρακάτω αρχεία. Εμείς θα χρησιμοποιήσουμε το editserver, το server και το Subseven.



Εικόνα 53: Subseven files

Ο server είναι το πρόγραμμα που στέλνουμε στο θύμα και αρχικά το ανοίγουμε με το editserver και κάνουμε κάποιες ρυθμίσεις. Αφού ανοίξουμε το πρόγραμμα editserver, πατάμε στο browse και επιλέγουμε το αρχείο server. Μετά πατάμε read current settings για να δούμε τις ρυθμίσεις που έχει αυτό το αρχείο. Εμείς αφήσαμε τις βασικές ρυθμίσεις και αποθηκεύσαμε το αρχείο αυτό.



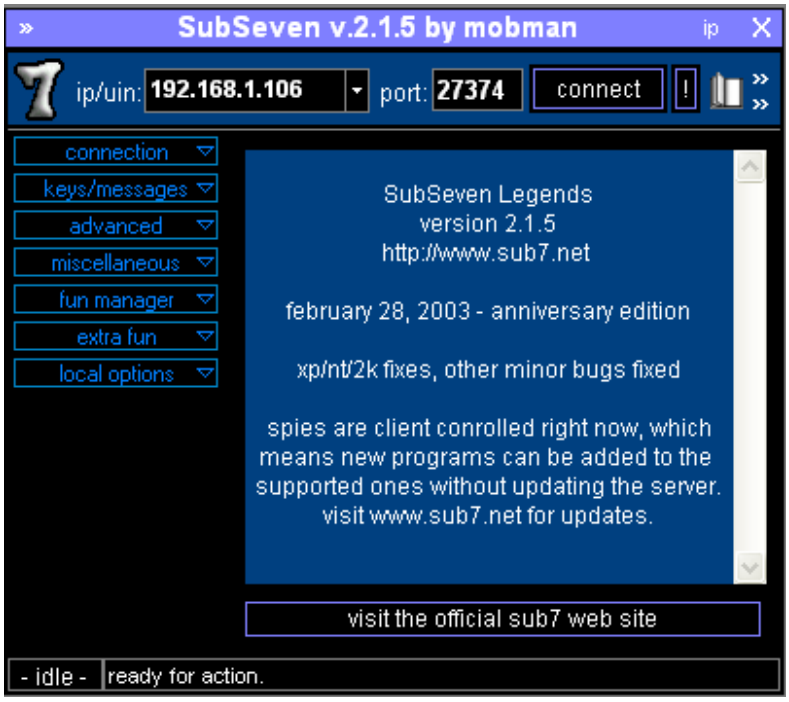
Εικόνα 54: Subseven editserver program

Αφού κάνουμε τις ρυθμίσεις που θέλουμε στέλνουμε το αρχείο server που δημιουργήσαμε στο θύμα με κάποιο τρόπο. Με το που θα εκτελεστεί αυτό το αρχείο θα τρέχει στον υπολογιστή του χωρίς να φαίνετε. Στην περίπτωσή μας το στείλαμε σε ένα υπολογιστή με Windows XP sp3. Για περισσότερες πληροφορίες ρυθμίσεις του server:

http://hackerthedude.blogspot.gr/2009/10/hacking-with-sub-7-hack-any-computer.html

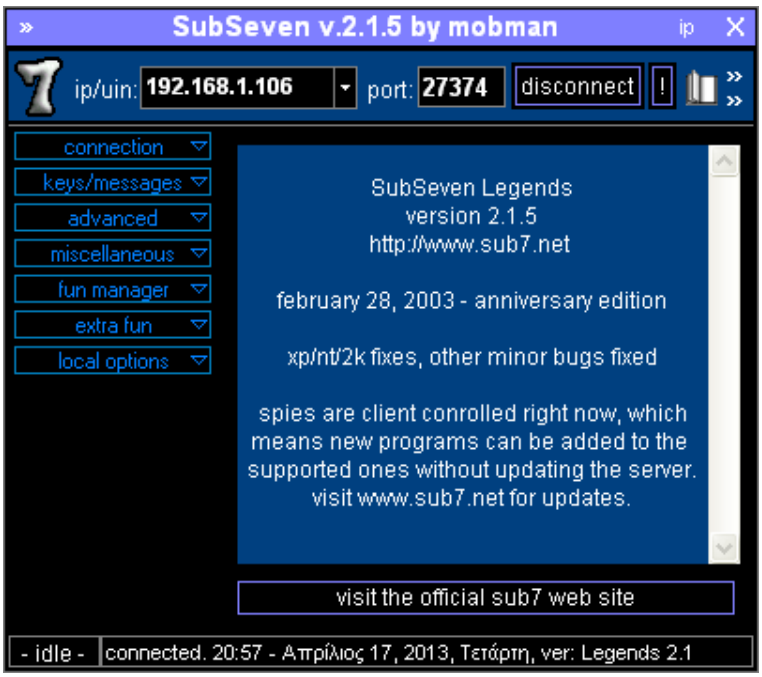
Αφού τώρα το αρχείο server τρέχει στον υπολογιστή του θύματος έχει δημιουργηθεί ένα κενό ασφάλειας. Με το πρόγραμμα Subseven στην πλευρά του επιτιθέμενου μπορούμε να εισβάλουμε στο σύστημα κάνοντας σύνδεση με την ip και την θύρα του θύματος που έχουμε ρυθμίσει να ανοίγει.

Τρέχοντας το Subseven δίνουμε την ip του θύματος και την πόρτα που έχουμε ρυθμίσει να ανοίγει το πρόγραμμα server.Μετά πατάμε το κουμπί Connect. Στην περίπτωση μας το θύμα έχει ip 192.168.1.106 και η θύρα που ρυθμίσαμε να ανοίγει είναι η 27374.



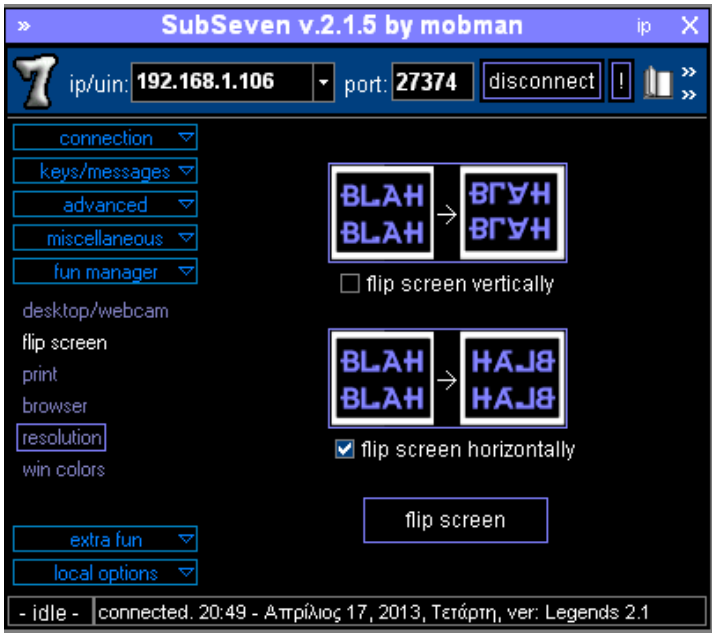
Εικόνα 55: Subseven client program

Αμέσως μετά πρέπει το πρόγραμμα να μας επιβεβαιώσει ότι έγινε σύνδεση με το μήνυμα κάτω αριστερά "connected". Τώρα μπορούμε να κάνουμε διάφορα πράγματα στον υπολογιστή του άλλου.



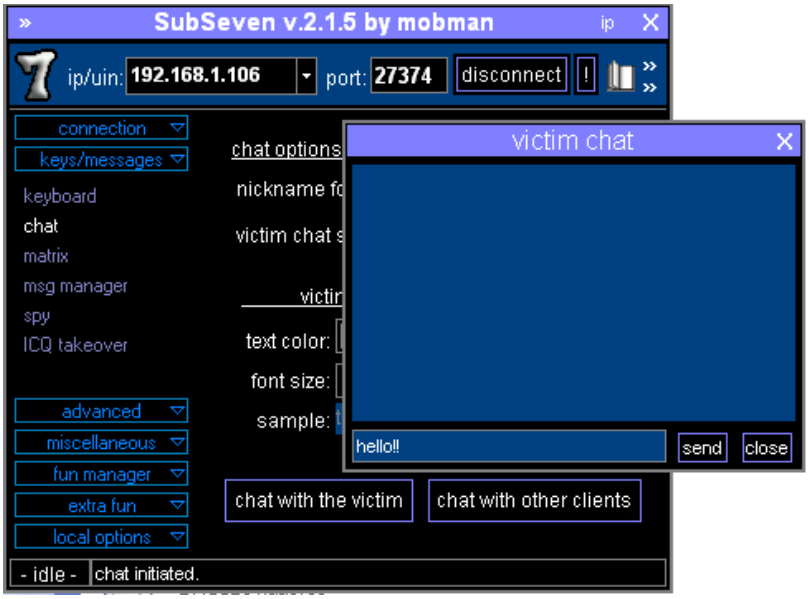
Εικόνα 56: Subseven client connected

Στην προκειμένη περίπτωση γυρίσαμε την οθόνη του ανάποδα πατώντας funmanager ->flipscreen ->flipscreenhorizontally ->flipscreen



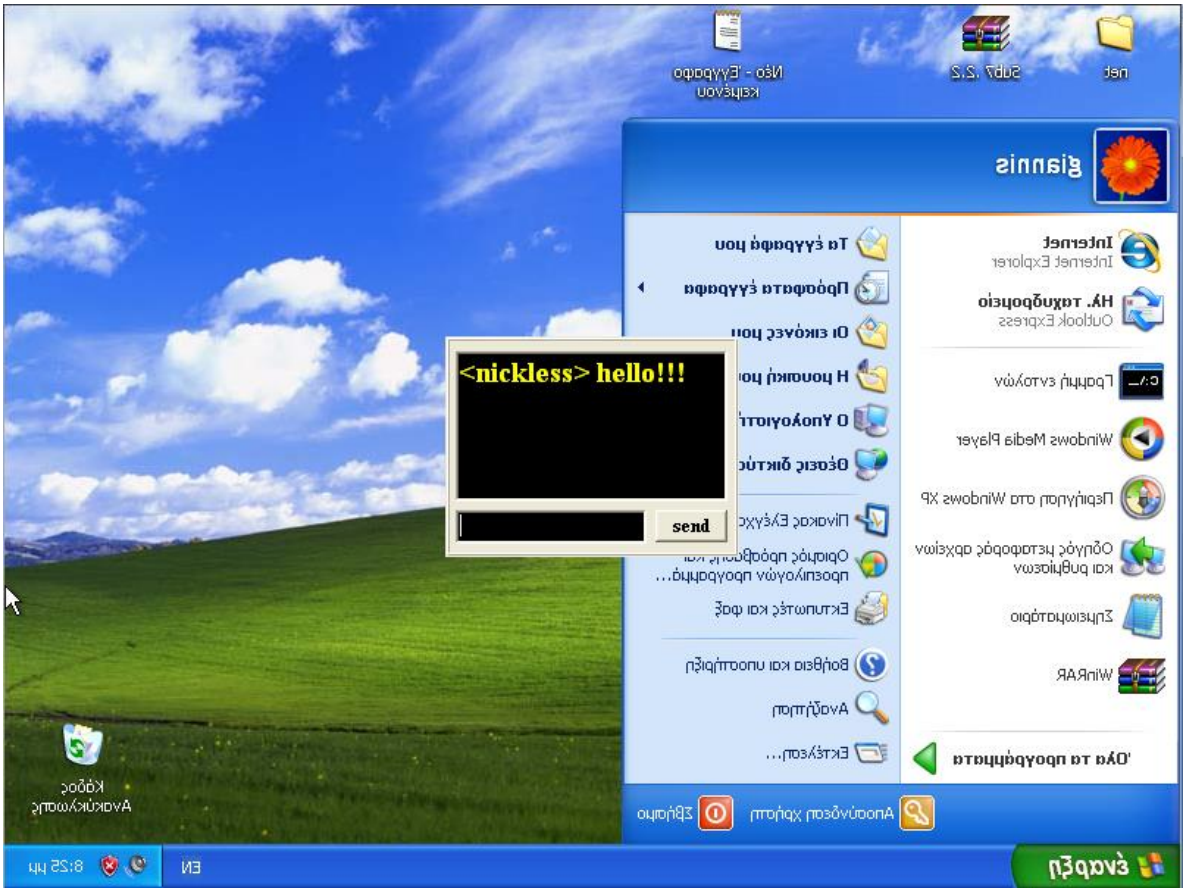
Εικόνα57: Subseven flip screen

Επίσης κάναμε chat με το θύμα πατόντας στο keys/messages -> chat -> Chat with the victim



Εικόνα 58: Subseven chat

Απο ένα screenshot απο την επιφάνεια εργασίας του θύματος το αποτέλεσμα είναι το παρακάτω:



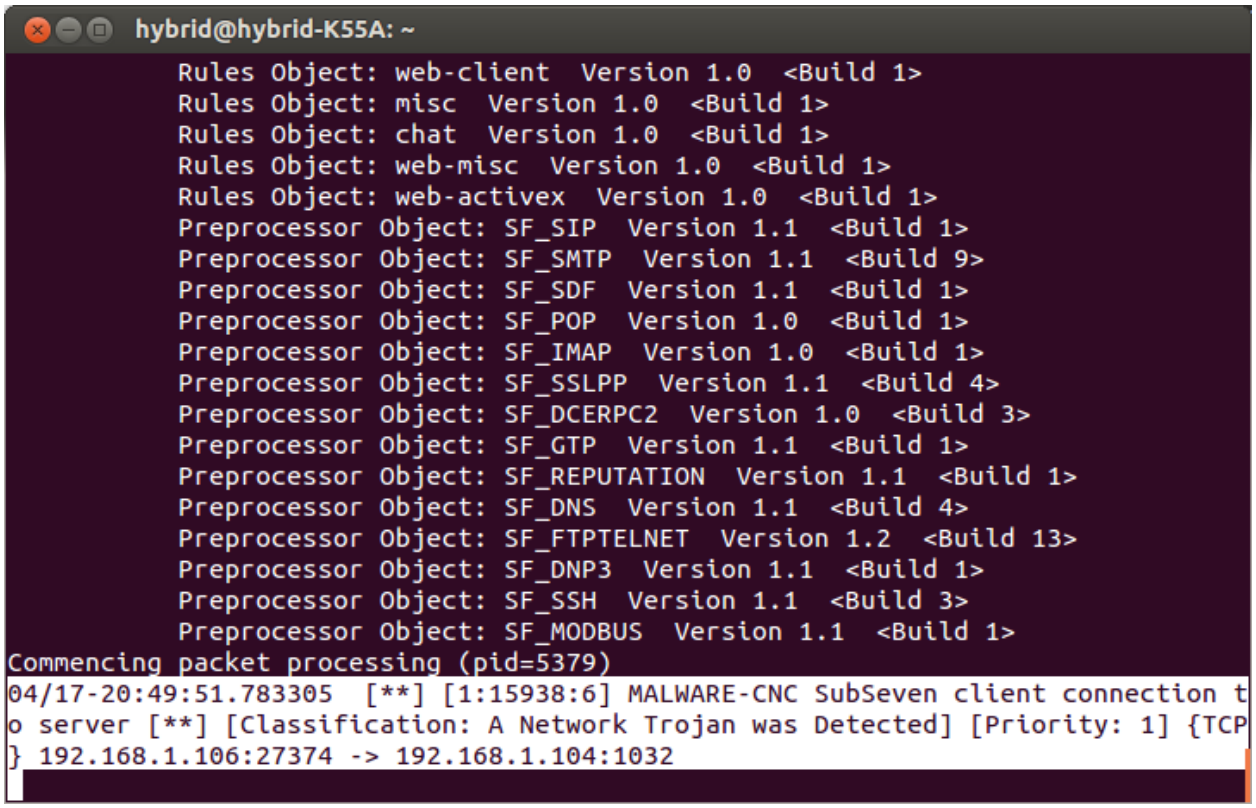
Εικόνα 59: Αποτέλεσμα Subseven

Ανίχνευση Επίθεσης

Μια τέτοια επίθεση είναι πολύ σοβαρή και ο επιτιθέμενος μπορεί στην κυριολεξία να κάνει ότι θέλει. Ανιχνεύεται με ένα καλό antivirus ή ένα καλό firewall. Εμείς καταφέραμε να την ανιχνεύσουμε με το Snortπαρακολουθώντας την κίνηση και τις συνδέσεις στο δίκτυο με τον κανόνα που ορίσαμε στην ρύθμιση του Snort.

Πριν ο επιτιθέμενος ανοίξει μία backdoor και αρχίζει να στέλνει εντολές στον υπολογιστή μας πρέπει πρώτα να συνδεθεί με τον υπολογιστή θύμα. Έτσι ο κανόνας που ορίσαμε ανιχνεύει μια σύνδεση του Subseven μέσα στο δίκτυο μας και πυροδοτεί ένα συναγερμό.

04/17-20:49:51.783305 [**] [1:15938:6] MALWARE-CNC SubSeven client connection to server [**] [Classification: A Network Trojan was Detected] [Priority: 1] {TCP} 192.168.1.106:27374 -> 192.168.1.104:1032



4.6 Ανίχνευση Exploit MS08_067_netapi

Περιγραφή επίθεσης

Το exploit είναι ένα πρόγραμμα το οποίο παρακάμπτει την ασφάλεια των υπολογιστών. Υπάρχουν πολλοί τρόποι να χρησιμοποιηθούν οι τρύπες ασφάλειας. Εάν, ένας προγραμματιστής κάνει ένα λάθος σε ένα πρόγραμμα, είναι μερικές φορές δυνατό να παρακάμψει την ασφάλεια. Η κωδικοποίηση τέτοιων προγραμμάτων, τα οποία επιτίθενται (χακαρονται) στα λάθη του προγραμματισμού ή των τρυπών ασφάλειας είναι η τέχνη του exploitation ή του exploit.

Τα exploits είναι χιλιάδες και κυκλοφορούν στο διαδίκτυο ελεύθερα. Φτιάχνονται και χρησιμοποιούνται συνήθως για να αυτοματοποιούν τις διαδικασίες μιας επίθεσης, ώστε κάποιος να μπορεί εύκολα να τεστάρει την ασφάλεια του συστήματος του.

Ένας καλός τρόπος για να βρει κάποιος πολλά exploits είναι να χρησιμοποιήσει το Metasploit. Το Metasploit είναι ένα project, που περιέχει πάρα πολλά exploits και βοηθάει στην εύκολη και γρήγορη εκτέλεση τους. Κάποιος μπορεί να το κατεβάσει και να το εγκαταστήσει σε συστήματα με λειτουργικό Linux και Windows.

Στο παρόν παράδειγμα χρησιμοποιούμε το exploit ms08_067_netapi που εκμεταλλεύεται την τρύπα ασφαλείας που υπάρχει σε κάποια συστήματα Microsoft Windows. Συγκεκριμένα εκμεταλλεύεται τα ελαττώματα του κώδικα της βιβλιοθήκης NetAPI32.dll. Αυτή η ευπάθεια αφορά την απομακρυσμένη εκτέλεση κώδικα. Ένας εισβολέας που εκμεταλλεύεται με επιτυχία αυτό το θέμα ευπάθειας θα μπορούσε να αποκτήσει απομακρυσμένα τον πλήρη έλεγχο ενός συστήματος που επηρεάζεται. Στα συστήματα που βασίζονται σε Microsoft Windows 2000, Windows XP και Windows Server 2003, ένας εισβολέας θα μπορούσε να εκμεταλλευτεί αυτήν την ευπάθεια μέσω RPC (Remote procedure call) χωρίς έλεγχο ταυτότητας και να εκτελέσει αυθαίρετο κώδικα.

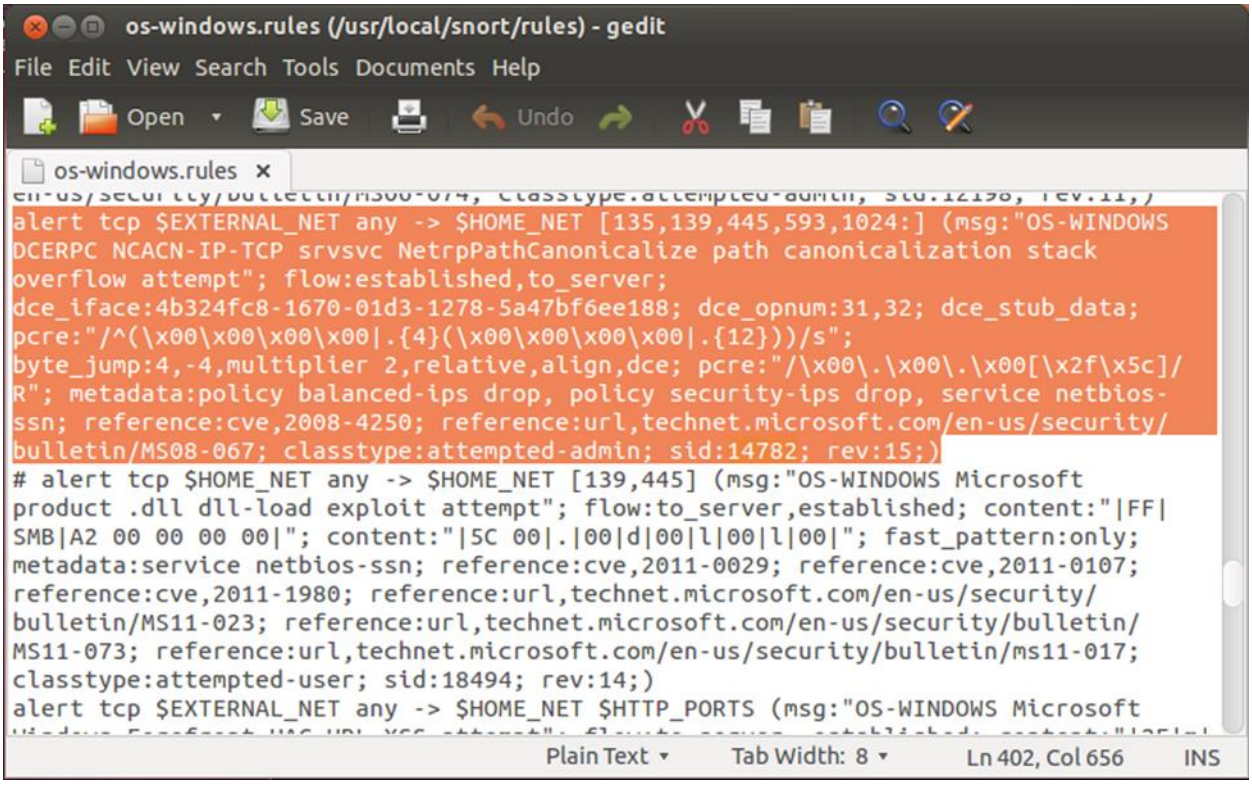
Ρύθμιση Snort

Το Snort χρησιμοποιεί κανόνες για να ανιχνεύει exploits και επιθέσεις προς συστήματα Windows. Ελέγχει τα πακέτα και τις συνδέσεις μέσα στο δίκτυο και μπορεί να εντοπίσει κάποια επίθεση από exploit.

Έτσι κάνοντας μια μικρή μελέτη στους κανόνες, θα βρούμε κανόνες που έχουν την δυνατότητα να εντοπίσουν διάφορα exploits. Το ruleset os-windows.rules περιέχει κανόνες για επιθέσεις προς συστήματα με λειτουργικό windows.

Έτσι ανοίγοντας το αρχείο os-windows.rules τσεκάρουμε αν υπάρχει ο κανόνας που κάνει ανίχνευση ενός τέτοιου exploit. Ο κανόνας είναι ο παρακάτω:

alert tcp $EXTERNAL_NET any -> $HOME_NET [135,139,445,593,1024:] (msg:"OS-WINDOWS DCERPC NCACN-IP-TCP srvsvc NetrpPathCanonicalize path canonicalization stack overflow attempt"; flow:established,to_server; dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188; dce_opnum:31,32; dce_stub_data; pcre:"/^(\x00\x00\x00\x00|.{4}(\x00\x00\x00\x00|.{12}))/s"; byte_jump:4,-4, multiplier 2, relative,align,dce; pcre:"/\x00\.\x00\.\x00[\x2f\x5c]/R"; metadata:policy balanced-ips drop, policy security-ips drop, service netbios-ssn; reference:cve,2008-4250; reference: url,technet.microsoft.com/en-us/security/bulletin/MS08-067; classtype:attempted-admin; sid:14782; rev:15;)



Εικόνα 60: Κανόνας για ανίχνευση exploit ms08_067_netapi

Εξήγηση της κεφαλής (header) του κανόνα "**alert tcp $EXTERNAL_NET any -> $HOME_NET [135,139,445,593,1024:]":**

- **alert**: Πυροδοτεί συναγερμό σε περίπτωση που ο κανόνας επαληθευτεί
- **tcp**: Ο κανόνας ελέγχει τα TCP πακέτα
- **$EXTERNAL_NET**:Ελέγχει αν η διεύθυνση αποστολέα του πακέτου ανήκει στο εξωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **any**: Το πακέτο να προέρχεται από οποιαδήποτε πόρτα του αποστολέα (Δε μας ενδιαφέρει αυτό το πεδίο)
- ->: Είναι τελεστής κατεύθυνσης πού ότι βρίσκεται αριστερά του αντιστοιχεί στη διεύθυνση IP και τη θύρα του αποστολέα, ενώ το ζεύγος δεξιά του τελεστή αντιστοιχεί στον παραλήπτη.
- **$HOME_NET**: Ελέγχει αν η διεύθυνση παραλήπτη του πακέτου ανήκει στο εσωτερικό δίκτυο (είναι μια μεταβλητή που την ορίζουμε στο αρχείο ρυθμίσεων του Snort)
- **[135,139,445,593,1024:]**: Ελέγχει αν το πακέτο προορίζεται στις θύρες 135,139,445,593 και 1024.

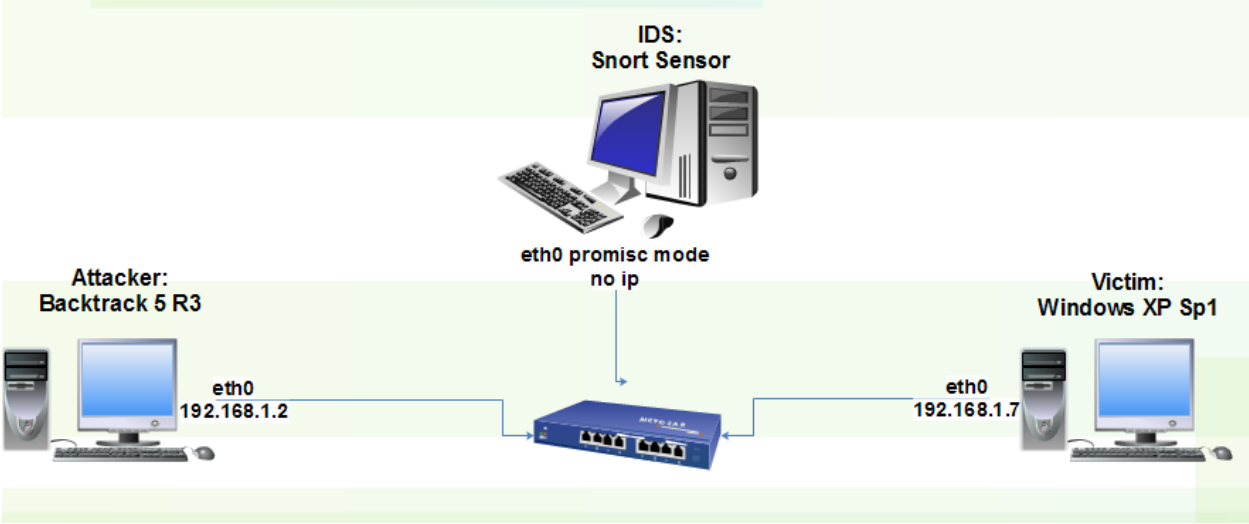
Εξήγηση ρυθμίσεων (Options) του κανόνα“ **msg:"OS-WINDOWS DCERPC NCACN-IP-TCP srvsvc NetrpPathCanonicalize path canonicalization stack overflow attempt"; flow:established,to_server; dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188; dce_opnum:31,32; dce_stub_data; pcre:" /^(\x00\x00\x00\x00|.{4}(\x00\x00\x00\x00|.{12}))/s"; byte_jump:4,-4,multiplier 2,relative,align,dce; pcre:"/\x00\.\x00\.\x00[\x2f\x5c]/R"; metadata:policy balanced-ips drop, policy security-ips drop, service netbios-ssn; reference:cve,2008-4250; reference:url,technet.microsoft.com/en-us/security/ bulletin/MS08-067; classtype:attempted-admin; sid:14782; rev:15;)**”**:**

- **msg:" OS-WINDOWS DCERPC NCACN-IP-TCP srvsvc NetrpPathCanonicalize path canonicalization stack overflow attempt ";** : Όταν πυροδοτείται Alert θα μας ενημερώνει για το είδος της επίθεσης με το μήνυμα “OS-WINDOWS DCERPC NCACN-IP-TCP srvsvc NetrpPathCanonicalize path canonicalization stack overflow attempt”.
- **flow:established,to_server;**: Για να ενεργοποιηθεί ο συναγερμός πρέπει να υπάρχει ροή TCP πακέτων προς τον Server και η σύνδεση μεταξύ τους να έχει επιτύχει.
- **dce_iface:4b324fc8-1670-01d3-1278-5a47bf6ee188;** : Αυτή η ρύθμιση αναφαίρετε στον προεπεξεργαστή DCE/RPC 2 περισσότερες πληροφορίες στο User Manual του Snort.
- **dce_opnum:31,32; :** Αυτή η ρύθμιση αναφαίρετε στον προεπεξεργαστή DCE/RPC 2 περισσότερες πληροφορίες στο User Manual του Snort.
- **dce_stub_data;** Αυτή η ρύθμιση αναφαίρετε στον προεπεξεργαστή DCE/RPC 2 περισσότερες πληροφορίες στο User Manual του Snort**.**
- **pcre:"/^(\x00\x00\x00\x00|.{4}(\x00\x00\x00\x00|.{12}))/s"; :** Κανόνας συμβατός με γλώσσα prce.
- **byte_jump:4,-4,multiplier 2,relative,align,dce; :** Αυτή η ρύθμιση αναφαίρετε στον προεπεξεργαστή DCE/RPC 2 περισσότερες πληροφορίες στο User Manual του Snort.
- **pcre:"/\x00\.\x00\.\x00[\x2f\x5c]/R"; :** Κανόνας συμβατός με γλώσσα prce.
- **metadata:policy balanced-ips drop, policy security-ips drop, service netbios-ssn;**
- **reference:cve,2008-4250; :** Με την βοήθεια ενός κατάλληλου plugin το Snort συνδέεται με εξωτερικά συστήματα για άντληση περισσότερης πληροφορίας για την επιπλέον προειδοποίηση και πληροφόρηση. Σύμφωνα με την ρύθμιση αυτή το Snortθα συνδεθεί με την ιστοσελίδα **“ http://cve.mitre.org/cgi-bin/cvename.cgi?name= 2008-4250” για** περισσότερες πληροφορίες προειδοποίησης
- **reference:url,technet.microsoft.com/en-us/security/bulletin/MS08-067; :** Με την βοήθεια ενός κατάλληλου plugin το Snort συνδέεται με εξωτερικά συστήματα για άντληση περισσότερης πληροφορίας για την επιπλέον προειδοποίηση και πληροφόρηση. Σύμφωνα με την ρύθμιση αυτή το Snortθα συνδεθεί με την ιστοσελίδα “**http://technet.microsoft.com/en-us/security/bulletin/MS08-067**” για περισσότερες πληροφορίες προειδοποίησης
- **classtype:attempted-admin; :** Κατατάσσουμε τον κανόνα στην κατηγορία “ attempted-admin ”
- **sid:14782;** : Ο κανόνας αυτός έχει αναγνωριστικό 14782 ανάμεσα στους υπόλοιπους κανόνες που χρησιμοποιούμε.
- **rev:15;**: Η έκδοση του κανόνα αυτού είναι η 15.

Εκτέλεση Επίθεσης

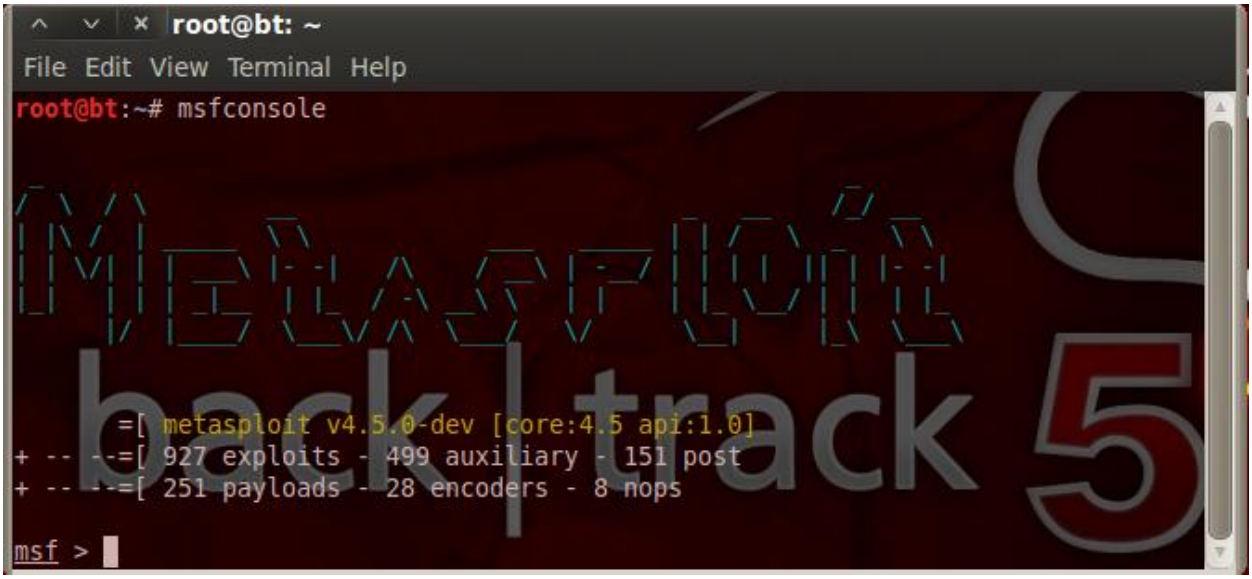
Για να πραγματοποιήσουμε αυτή την επίθεση χρησιμοποιήσαμε το Backtrack 5 R3, όπου έχει το Metasploit ενσωματωμένο με τα διάφορα exploits που κυκλοφορούν. Επίσης η επίθεση εκτελέστηκε προς υπολογιστή με σύστημα Windows XP sp1. Ο στόχος της επίθεσης είναι να συνδεθούμε απομακρυσμένα με τον υπολογιστή στόχο και να έχουμε πλήρης πρόσβαση στα δεδομένα και το λειτουργικό του σύστημα μέσω VNC (Virtual Network Computing). Τέλος, στο ίδιο δίκτυο υπάρχει συνδεδεμένο και το Snort ελέγχοντας όλη την κίνηση του δικτύου κάνοντας ανίχνευση των επιθέσεων.

Στο παρακάτω σχήμα φαίνεται το δίκτυο που εκτελέστηκε η επίθεση:



Εικόνα 61: Σχήμα δικτύου όπου πραγματοποιήθηκε το exploit ms08_067_netapi

Για να εκτελέσουμε το exploit χρησιμοποιήσαμε το Metasploit που ενσωματώνεται στο Backtrack. Ανοίγουμε ένα τερματικό στο Backtrack και γράφουμε “msfconsole” ώστε να ανοίξει η κονσόλα του Metasploit που περιέχει σχεδόν όλα τα exploits.



Εικόνα 62: Metasploit

Αρχικά θα επιλέξουμε το exploit που θέλουμε να χρησιμοποιήσουμε. Αυτό γίνεται με την εντολή:

use windows/smb/ms08_067_netapi

Μετά θα κάνουμε κάποιες ρυθμίσεις του exploit που με αυτές θα γίνει η επίθεση προς το θύμα.

Ρυθμίζουμε την IP του θύματος με την εντολή:

set rhost 192.168.1.7

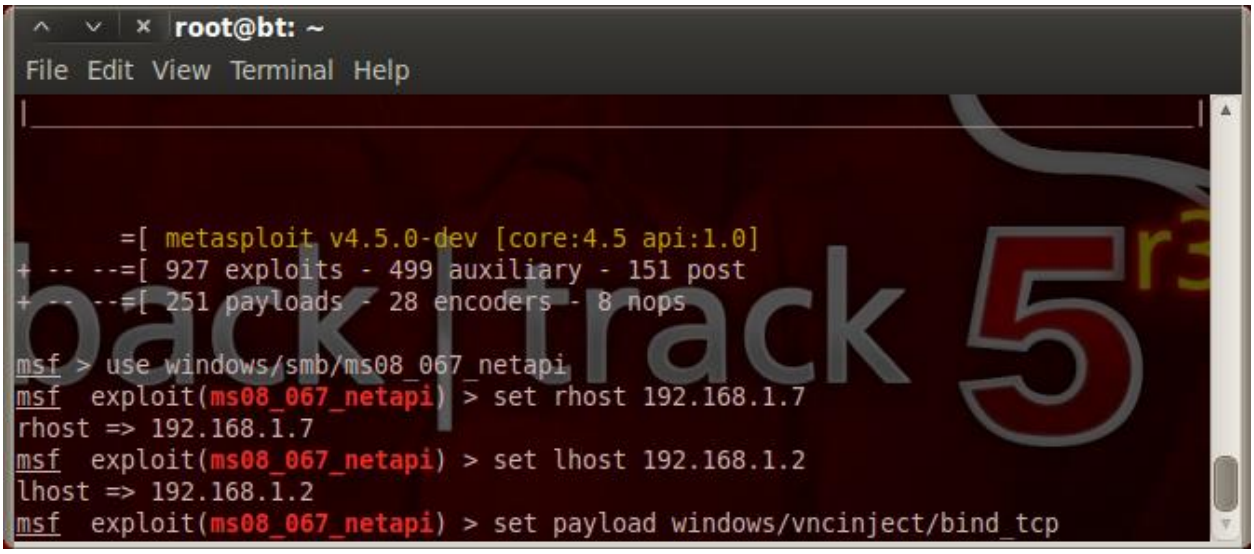
Επίσης ρυθμίζουμε την IP του δικού μας host με την εντολή:

set lhost 192.168.1.2

Η πόρτα προορισμού είναι ρυθμισμένη από προεπιλογή να είναι 445.

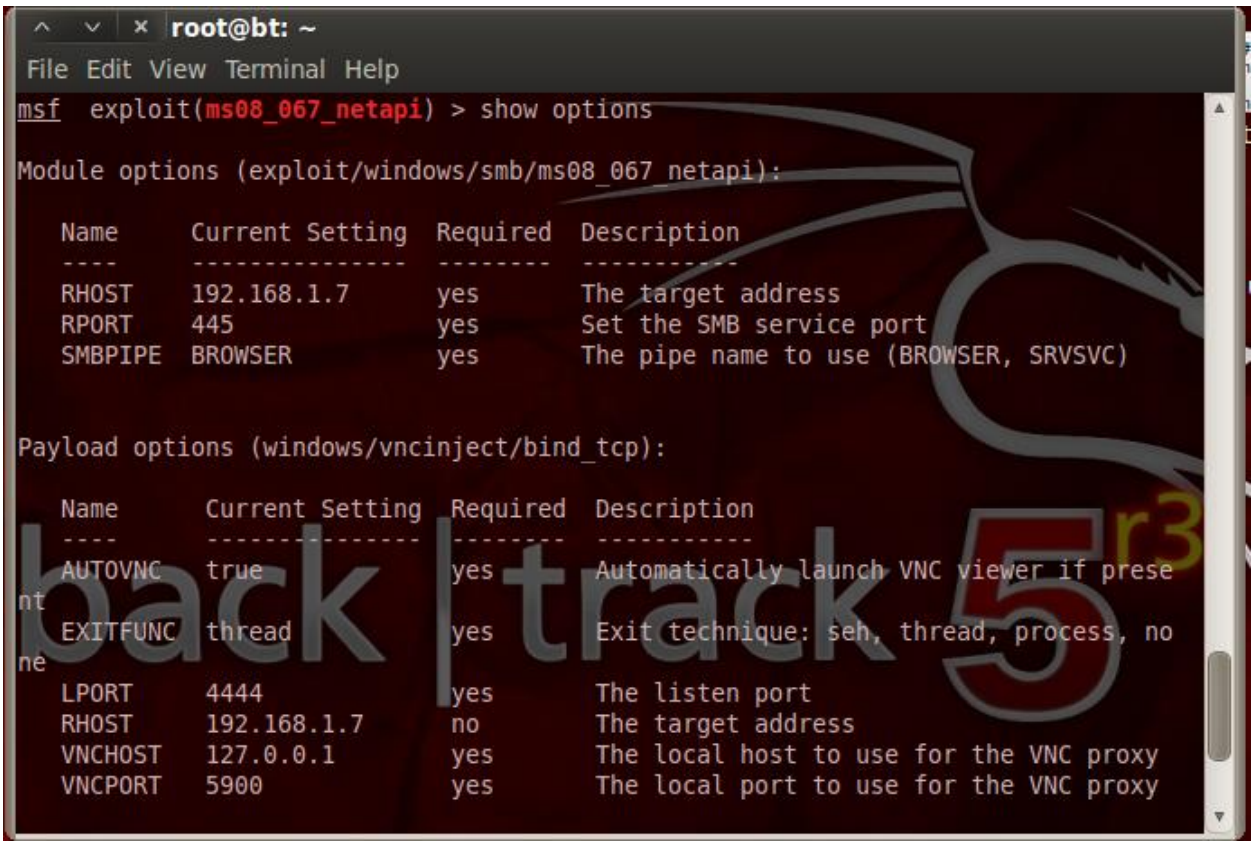
Τέλος με την παρακάτω εντολή ρυθμίζουμε ποιο payload θα χρησιμοποιήσουμε (στο συγκεκριμένο παράδειγμα χρησιμοποιούμε ένα payload όπου ανοίγει απομακρυσμένη σύνδεση με το θύμα βλέποντας σε γραφικό περιβάλλων την επιφάνεια εργασίας του).

set payload windows/vncinject/bind_tcp



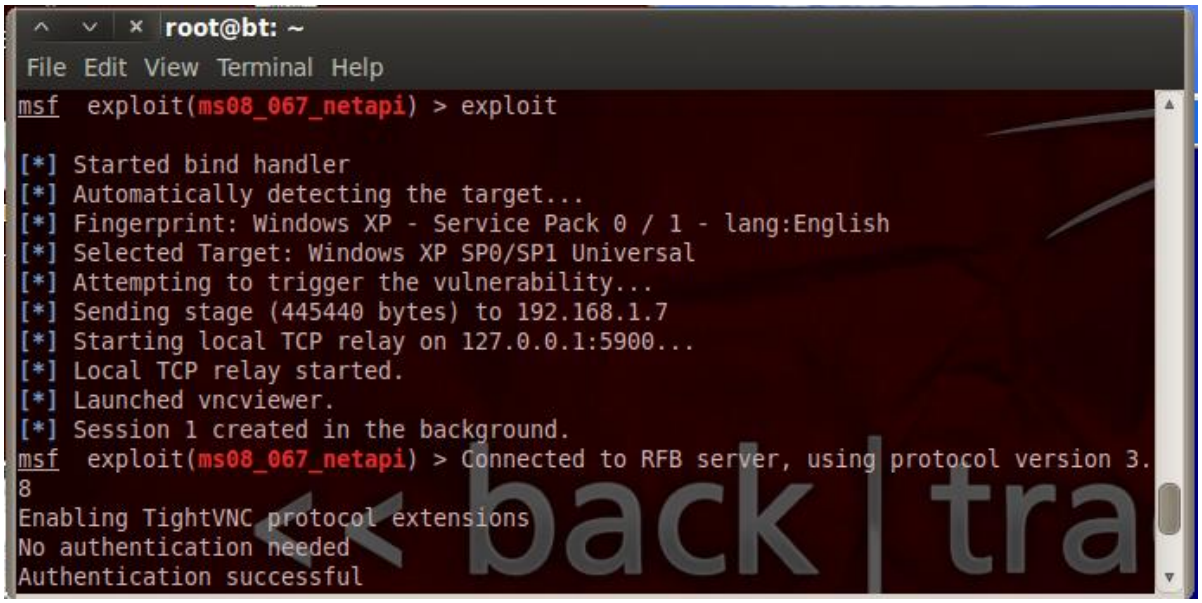
Εικόνα 63: Επιλογή και ρύθμιση exploit

Αν γράψουμε στην κονσόλα show options θα δούμε όλες τις ρυθμίσεις και αυτές που κάναμε εμείς:



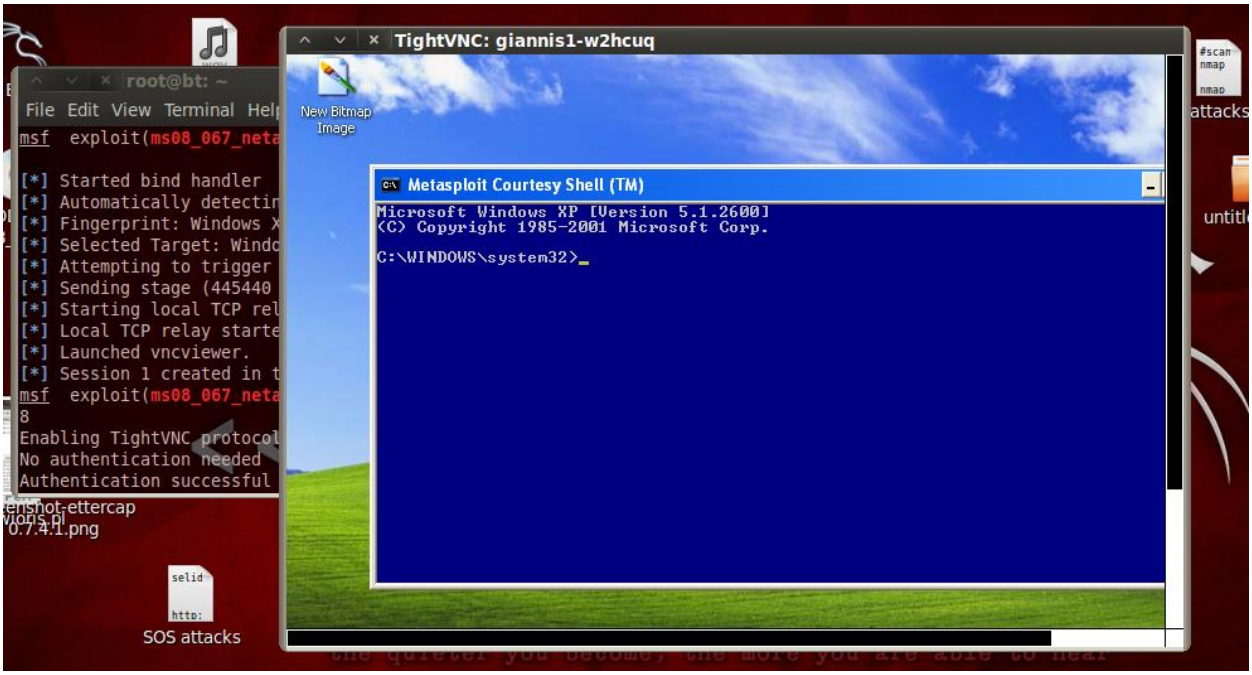
Εικόνα 64: Ρυθμίσεις του exploit

Τέλος γράφοντας exploit μπορούμε να τρέξουμε το exploit και να ανοίξουμε απομακρυσμένη σύνδεση με το θύμα:



Εικόνα 65: Επιτυχής σύνδεση με το θύμα

Αμέσως θα ανοίξει στο Backtrack μια επιφάνεια εργασίας του θύματος δίνοντας μας πλήρης δικαιώματα.

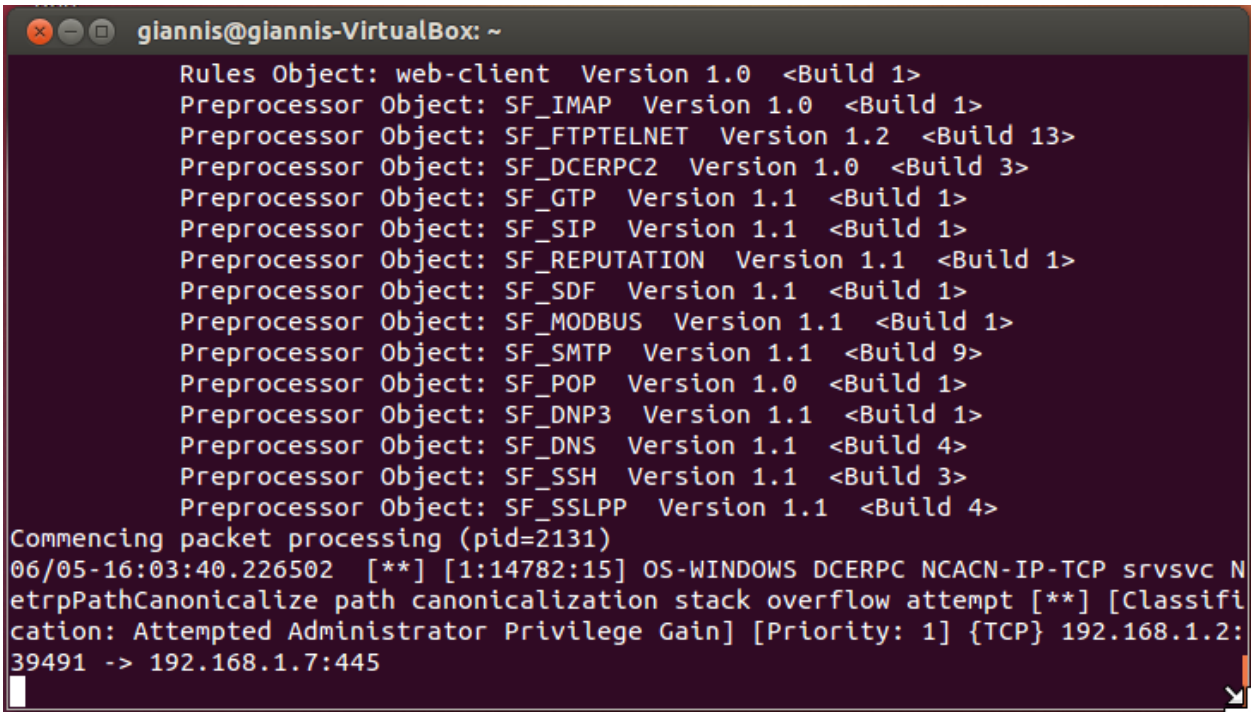


Εικόνα 66: Άνοιγμα απομακρυσμένης σύνδεσης με το θύμα μέσω VNC

Ανίχνευση επίθεσης

Το Snort θα μας ενημερώσει με ένα Alert για αυτή την επίθεση. Μια τέτοια επίθεση είναι πολύ επικίνδυνη και πραγματοποιείτε εύκολα σε συστήματα όπως αυτό. Έτσι ο διαχειριστής του δικτύου αν ανιχνεύσει μια τέτοια επίθεση πρέπει να λάβει μέτρα αποκλεισμού του επιτιθέμενου από το δίκτυο.

Στη παρακάτω εικόνα φαίνεται η ανίχνευση αυτής της επίθεσης από το Snort:



Η επίθεση πραγματοποιήθηκε στης 06/05 και ώρα 16:03:40,2. Το αναγνωριστικό SID για αυτή την επίθεση είναι 14782. Με αυτό το αναγνωριστικό μπορούμε να βρούμε πληροφορίες στο site του snort για το είδος της επίθεσης. Επίσης έχει προτεραιότητα 1(Priority), που σημαίνει ότι το Snort δίνει πρώτα προτεραιότητα στην ανίχνευση τέτοιων επιθέσεων. Παρατηρούμε ότι η επίθεση πραγματοποιείται από τον host με IP 192.168.1.2 με πόρτα 39491, προς τον host με IP 192.168.1.7 στην πόρτα 445.

Κεφάλαιο 5 Αποτροπή επιθέσεων με το Snort

Όπως έχουμε περιγράψει σε προηγούμενη ενότητα, το Snort συνήθως χρησιμοποιείτε σαν ανοιχτού κώδικα Σύστημα Ανίχνευσης Εισβολών (Intrusion Detection System) με μεγάλη τεκμηρίωση και άριστη υποστήριξη επικοινωνίας. Παρόλα αυτά το Snort έχει την δυνατότητα να λειτουργήσει και ως Σύστημα Αποτροπής Εισβολών (Intrusion Prevention System). Αυτό γινόταν εφικτό και γίνεται ακόμα με διάφορες προσθήκες όπως το SnortSam. Πλέον το Snort από την έκδοση 2.9 και μετά, με την προσθήκη του DAQ επιτρέπει καινούρια ευελιξία διαχωρίζοντας τις δικτυακές λειτουργίες σύλληψης σε ρυθμιζόμενες προσθήκες δίνοντας στο χρήστη τη δυνατότητα να καθορίζει, τι θα χρησιμοποιεί κάθε φορά για να κάνει σύλληψη της κίνησης. Κάποιες από αυτές τις ρυθμίσεις παρέχουν και inline Αποτροπή Εισβολών (Intrusion Prevention).

5.1 DAQ

Ξεκινώντας με την έκδοση 2.9 του Snort, το Snort χρησιμοποιεί καινούριους μηχανισμούς για σύλληψη των πακέτων. Τις Data Acquisition (DAQ) βιβλιοθήκες, που ανακοινώθηκαν για πρώτη φορά τον Αύγουστο του 2010 στο VRT block. Πριν από αυτές, το Snort ενσωμάτωνε άλλες δικτυακές λειτουργίες σύλληψης πακέτων. Με το DAQ όλες οι λειτουργίες αυτές έχουν διαχωριστεί σε διάφορα modules όπου μπορούν να επιλεχθούν και να ενεργοποιηθούν με διάφορες παραμέτρους κατά την εκκίνηση του Snort. Το DAQ σχεδιάστηκε για να παρέχει inline λειτουργικότητα και να παρέχει ευελιξία στην προσθήκη καινούριων modules.

Το DAQ για να λειτουργήσει προαπαιτεί δύο βιβλιοθήκες:

- Την **Libpcap** για να κάνει σύλληψη των πακέτων
- Την **Libdnet** για άλλες δικτυακές λειτουργίες

Τέσσερα από τα έξι modules του Snort επιτρέπουν inline λειτουργία και απόρριψη πακέτων. Πριν, αν κάποιος ήθελε να χρησιμοποιήσει αυτή τη λειτουργία, έπρεπε να να κάνει compile το Snort με κάποιο patch. Ο όρος “inline” προσδιορίζει την θέση του Snort ανάμεσα από δύο συσκευές ή δίκτυα. Αυτή η θέση του επιτρέπει να ελέγχει όλη την κίνηση μεταξύ των δύο, συνεπώς θα έχει την δυνατότητα να απορρίπτει πακέτα που προέρχονται από πιθανή επίθεση.

Τα modules του DAQ που περιλαμβάνονται με το Snort είναι τα παρακάτω:

- **Pcap**: Είναι η προεπιλεγμένη ρύθμιση όπου το Snort χρησιμοποιείτε για Sniffing και σαν IDS.
- **Afpacket**: Το Snort τρέχει “inline” πάνω σε σύστημα Linux χρησιμοποιώντας 2 γεφυρωμένες κάρτες δικτύου.
- **Ipq**: Το Snort τρέχει “inline” πάνω σε σύστημα Linux χρησιμοποιώντας το netfilter, αντικαθιστά το snort_inline patch που είχε δημιουργηθεί για το Snort
- **Nfq**: Το Snort τρέχει “inline” πάνω σε Linux χρησιμοποιώντας το netfilter

- **Ipfw**: Το Snort τρέχει inline πάνω σε OpenBSD και FreeBSD χρησιμοποιώντας εκτροπή υποδοχών μαζί με pf και ipfw firewalls
- **Dump**: Επιτρέπει την δοκιμή των μηχανισμών "inline" και κανονικής λειτουργικότητας του Snort

5.2 Το Snort σαν IPS

Μια συνήθης συμβουλή για την ανάπτυξη του Snort σε inline ρύθμιση, είναι η προσοχή των False Positives. Όπως έχουμε αναφέρει στην θεωρία αυτής της εργασίας τα false positives είναι η ιδιαιτερότητα του Snort κάποιες φορές να πυροδοτεί συναγερμούς χωρίς να υπάρχει κάποια παράβαση της ασφάλειας. Αυτά τα Alerts μπορεί να πυροδοτηθούν για πολλούς λόγους, όπως επειδή κάποιος κανόνας δεν είναι ακριβής στην περιγραφή της πιθανής επίθεσης. Τα false positives που υπάρχουν στο Snort όταν λειτουργεί σε παθητική λειτουργία σαν IDS, είναι λιγότερο επικίνδυνα από όταν το Snort τρέχει σε inline mode, διότι όταν το Snort τρέχει σε αυτή την λειτουργία υπάρχει περίπτωση να διακόψει νόμιμες συνδέσεις.

Τα IPS false positives μπορεί να γίνουν επικίνδυνα σε μεγάλα και πολύπλοκα δίκτυα. Σκεφτείτε μία μεγάλη επιχείρηση όπου η ομάδα ασφάλειας διαχειρίζεται ένα IPS και διάφορες λειτουργίες του δικτύου και συνεπώς όλη την δικτυακή υποδομή. Ανάλογα με το πού θα βρίσκεται το IPS μηχάνημα μέσα στο δίκτυο οι χρήστες θα αντιμετωπίζουν αρκετά προβλήματα συνδεσιμότητας, εξαιτίας των false positives. Η αντιμετώπιση προβλημάτων σε μια τέτοια περίπτωση είναι περίπλοκη διότι δύο διαφορετικές ομάδες με διαφορετικά συστήματα πρέπει να δουλέψουν ώστε να βρουν την αιτία.

Μία επιλογή ώστε να μειωθεί ο κίνδυνος των false positives είναι να επιλέγουμε να εφαρμόζονται μόνο οι ποιο αξιόπιστοι κανόνες, ώστε να απορρίπτουμε ύποπτη κίνηση. Αυτοί οι κανόνες θα πρέπει να είναι ελεγμένοι και ρεσταρισμένοι σε ένα IDS πρώτα όπου και αν υπάρχει false positive δεν υπάρχει πρόβλημα αν εξεταστούν σωστά τα Alerts. Ασφαλώς χρησιμοποιώντας μόνο λίγους κανόνες το IPS μας παρέχει λιγότερη ασφάλεια από τα διάφορα exploits.

Μια άλλη επιλογή είναι το σημείο όπου θα συνδέσουμε το IPS. Για παράδειγμα αν στήσουμε το σύστημά μας ανάμεσα από ένα Web Server και το υπόλοιπο δίκτυο θα πρέπει να χρησιμοποιήσουμε μόνο WEB κανόνες. Όπως καταλαβαίνουμε έτσι μειώνουμε τα false positives.

5.3 DAQ Pcap module

Η λειτουργία DAQ Pcap είναι η προεπιλεγμένη λειτουργία που τρέχει το Snort αν δεν επιλέξουμε κάποια άλλη κατά την εκκίνηση του. Με αυτή την λειτουργία το Snort όπως έχουμε περιγράψει μπορεί να τρέξει στις 3 βασικές του λειτουργίες. Μπορεί να τρέξει σαν Sniffer και να συλλέγει όλη την κίνηση, να καταγράφει αυτή την κίνηση σε log αρχεία και να τρέχει σαν IDS σαν δαίμονας.

Η λειτουργία Sniffer δεν είναι και τόσο χρήσιμη διότι αν η κίνηση είναι μεγάλη δε θα προλαβαίνουμε να βλέπουμε τα πακέτα στην κονσόλα του υπολογιστή μας. Έτσι σε αυτή τη λειτουργία θα πρέπει να χρησιμοποιούμε κάποια φίλτρα για να βλέπουμε μονό συγκεκριμένη κίνηση που μας ενδιαφέρει. Προϋποθέτει την σύνδεση του Snort σε μία SPAN πόρτα ενός switch ή σε μια δικτυακή τρύπα (tap). Έτσι ξεκινάμε το Snort όπως ήδη έχουμε αναφέρει με την εντολή "snort –v –I eth0" ή μπορούμε να το ξεκινήσουμε με την εντολή "snort –v –i eth0 host 192.168.1.23" ώστε να δούμε την κίνηση μόνο από ένα host με IP 192.168.1.23.

Όπως είπαμε αυτή η λειτουργία δεν είναι τόσο χρήσιμη και αξιόπιστη. Γι αυτό και έχουμε την δυνατότητα να καταγράφουμε τα πακέτα σε log αρχεία. Αυτό το πετυχαίνουμε ξεκινώντας το Snort σε με μία επιπλέον παράμετρο " snort –v –i eth0 –l /var/log/snort" , όπου θα καταγράφει όλη την κίνηση στο φάκελο /var/log/snort, ώστε να επεξεργαστούμε την κίνηση με κάποιο ειδικό πρόγραμμα όπως το wireshark.

Τέλος μπορούμε να τρέξουμε το Snort σαν IDS προσθέτοντας την –D παράμετρο.

Για περισσότερες πληροφορίες για αυτές τις λειτουργίες μπορείτε να δείτε στο Κεφάλαιο 2 στις υποενότητες 2.1.1, 2.1.2, 2.1.3 αυτής της εργασίας. Στη συνέχια αυτής υης εργασίας θα περιγράψουμε την λειτουργία Afpacket και θα εκτελέσουμε κάποια σενάρια όπου το Snort θα ανταποκρίνεται σαν IPS.

5.4 DAQ Afpacket module

Αυτή η λειτουργία του DAQ είναι διαθέσιμη μόνο σε λειτουργικό Linux. Η λειτουργία Afpacket εκμεταλλεύεται τους κανόνες του Snort και δύο κάρτες δικτύου ώστε να απορρίπτει τα πακέτα από κάποια πιθανή επίθεση. Αυτό το πετυχαίνει χωρίς να βασίζεται σε κάποιο firewall όπως το netfilter. Έτσι το Snort χρησιμοποιεί ένα ζεύγος από κάρτες δικτύου σαν διαφανή γέφυρα από όπου περνάει όλη η κίνηση της γραμμής και δρομολογείτε κατάλληλα ή απορρίπτεται από το Snort.

Αν θέλει κάποιος η λειτουργία αυτή να είναι επιτυχείς θα πρέπει να λάβει υπόψη του κάποια πράγματα:

- Αρχικά, θα πρέπει να υπάρχουν δύο κάρτες δικτύου όπου η μία θα συνδέεται με το ένα μέρος του δικτύου και η άλλη με το άλλο.
- Για να δημιουργηθεί η γέφυρα μεταξύ των δύο καρτών θα πρέπει το Snort να τρέχει σαν δαίμονας και να έχουν επιλεγεί αυτές οι κάρτες δικτύου, ώστε να δρομολογείτε ή να απορρίπτεται η κίνηση από το ένα στο άλλο δίκτυο.

- Θα πρέπει το Snort να κάνει restart ή να ειδοποιεί τον διαχειριστή σε περίπτωση που κρασάρει, ώστε να αποκατασταθεί άμεσα η σύνδεση των δύο σημείων μέσα στο δίκτυο.
- Κανονικά, αν θέλουμε να δημιουργήσουμε μια γέφυρα με δυο κάρτες δικτύου θα πρέπει να χρησιμοποιήσουμε και διάφορες εφαρμογές. Παρόλα αυτά, η λειτουργία Afpacket του Snort διαχειρίζεται το γεφύρωμα από μόνη της. Το μόνο που χρειάζεται είναι να ρυθμίσουμε τις κάρτες να είναι ανοιχτές και να δέχονται όλη την κίνηση.
- Επίσης για λόγους ασφαλείας μπορούμε να ρυθμίσουμε τις κάρτες να μην έχουν κάποια διεύθυνση IP.

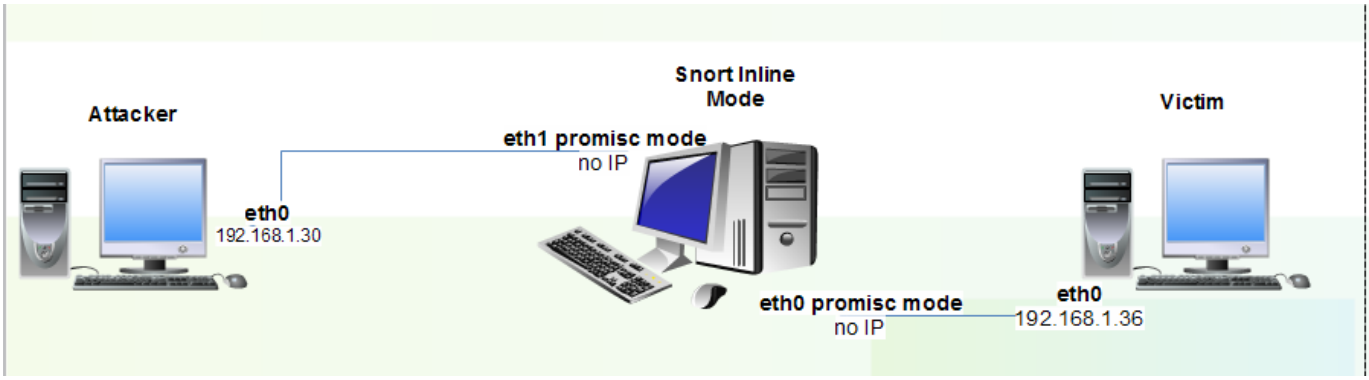
5.4.1 Ρυθμίσεις για σενάριο Afpacket mode

Στο παρόν παράδειγμα τρέξαμε ένα πολύ απλό παράδειγμα το οποίο μας δείχνει πώς με την Afpacket λειτουργία μπορούμε να κάνουμε το Snort με τους κανόνες που διαθέτει να λειτουργεί σαν IPS. Για την υλοποίηση του σεναρίου χρησιμοποιήσαμε 3 εικονικούς υπολογιστές :

Attacker: Backtrack 5 R3 Linux

Snort Sensor: Snort inline mode on Ubuntu Linux

Victim: Windows xp sp3



Εικόνα 67: Απεικόνηση Snort σε inline σύνδεση

Όπως βλέπουμε σχηματικά το Snort διαθέτει δύο κάρτες δικτύου την eth0 και eth1 ρυθμισμένες να ακούνε όλη την κίνηση από τις δύο πλευρές του δικτύου. Στις κάρτες δεν έχουμε δώσει κάποια IP για περισσότερη ασφάλεια του IPS μας. Οι εντολές που χρησιμοποιήσαμε για να κάνουμε τις δυο κάρτες να λειτουργούν σε αυτή την κατάσταση είναι:

ifconfig eth0 up promisc

ifconfig eth1 up promisc

Όπως έχουμε πει το Snort με την λειτουργία Afpacket λαμβάνει την κίνηση και από τις δύο πλευρές με την αντίστοιχη κάρτα δικτύου και μετά αφού ελέγξει την κίνηση δρομολογεί ή αποπίπτει τα πακέτα προς τον προορισμό τους. Για να γίνει αυτό εφικτό πρέπει να δύο δίκτυα που συνδέει το Snort να έχουν την ίδια μάσκα δικτύου.

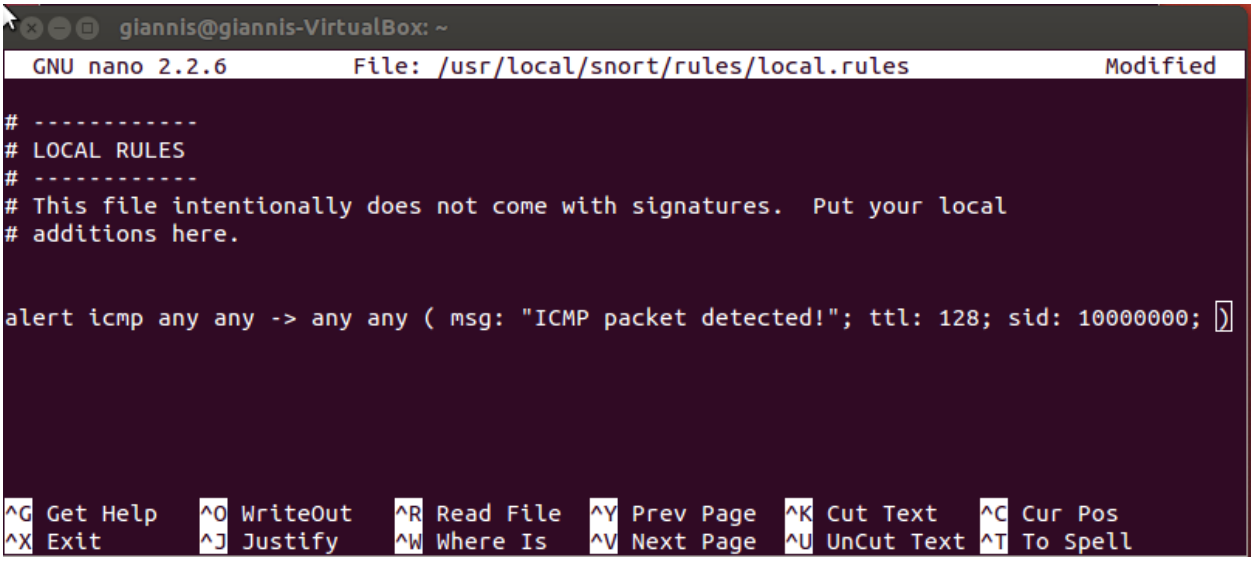
Αφού από υλικό και συνδεσιμότητα είμαστε έτοιμοι αυτό που μένει να κάνουμε είναι να ρυθμίσουμε το Snort να ξεκινάει κατάλληλα. Αρχικά ανοίγουμε το αρχείο ρυθμίσεων του Snort (snort.conf), όπως έχουμε περιγράψει σε προηγούμενο κεφάλαιο και προσθέτουμε τις παρακάτω γραμμές στις ρυθμίσεις του DAQ, ώστε να ενεργοποιήσουμε την λειτουργία Afpacket inline.

config daq: afpacket

config daq_mode: inline

Στο παρόν παράδειγμα απλά θα κάνουμε ping από τον attacker στο victim και απλά θα γράψουμε ένα κανόνα όπου ανιχνεύει αυτά τα πακέτα ping. Αρχικά θα δούμε τα Alert και στην συνέχεια θα αλλάξουμε τον κανόνα για να απορρίπτει το Snort αυτά τα ping. Συνεπώς, το Snort θα λειτουργήσει σαν IPS και θα είναι ένα καλό παράδειγμα για το πώς να ρυθμίζουμε και να αλλάζουμε τους κανόνες και το Snort ώστε να λειτούργει σαν IPS μέσω της λειτουργίας Afpacket του DAQ.

Τον κανόνα τον γράψαμε στο ruleset local.rules και στην συνέχεια ενεργοποιήσαμε από το αρχείο ρυθμίσεων το συγκεκριμένο ruleset. Παρακάτω φαίνετε ο κανόνας που γράψαμε:



Εικόνα 68: Κανόνας για Alert από κάποιο Ping

Αφού κάναμε τις ρυθμίσεις τώρα είμαστε έτοιμοι και μπορούμε να ξεκινήσουμε το Snort με τις κατάλληλες παραμέτρους:

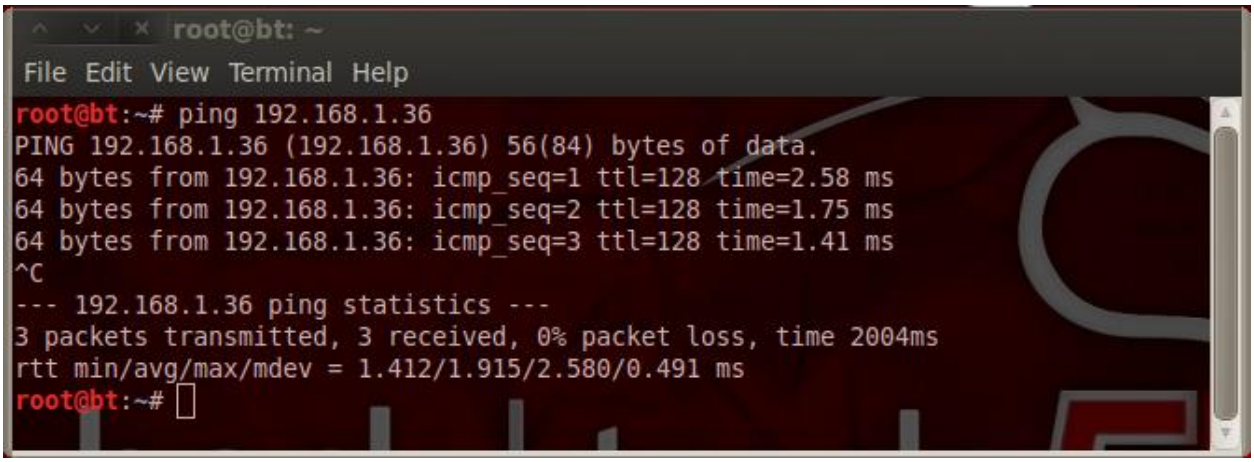
/usr/local/snort/bin/snort –Q –c /usr/local/snort/etc/snort.conf –A console –i eth0:eth1

- **-Q** : Τρέχει το Snort σε inline λειτουργία. (Συγκεκριμένα αυτό το κάναμε πρίν που ρυθμίσαμε το αρχείο ρυθμίσεων ο DAQ να είναι σε inline λειτουργία "config daq_mode: inline").
- **-c** : Προσδιορίζουμε πού είναι το αρχείο ρυθμίσεων του Snort.
- **-i** : Προσδιορίζουμε ποιες κάρτες δικτύου θα χρησιμοποιήσει το Snort.

Σενάριο Ping και απόκριση με Alert

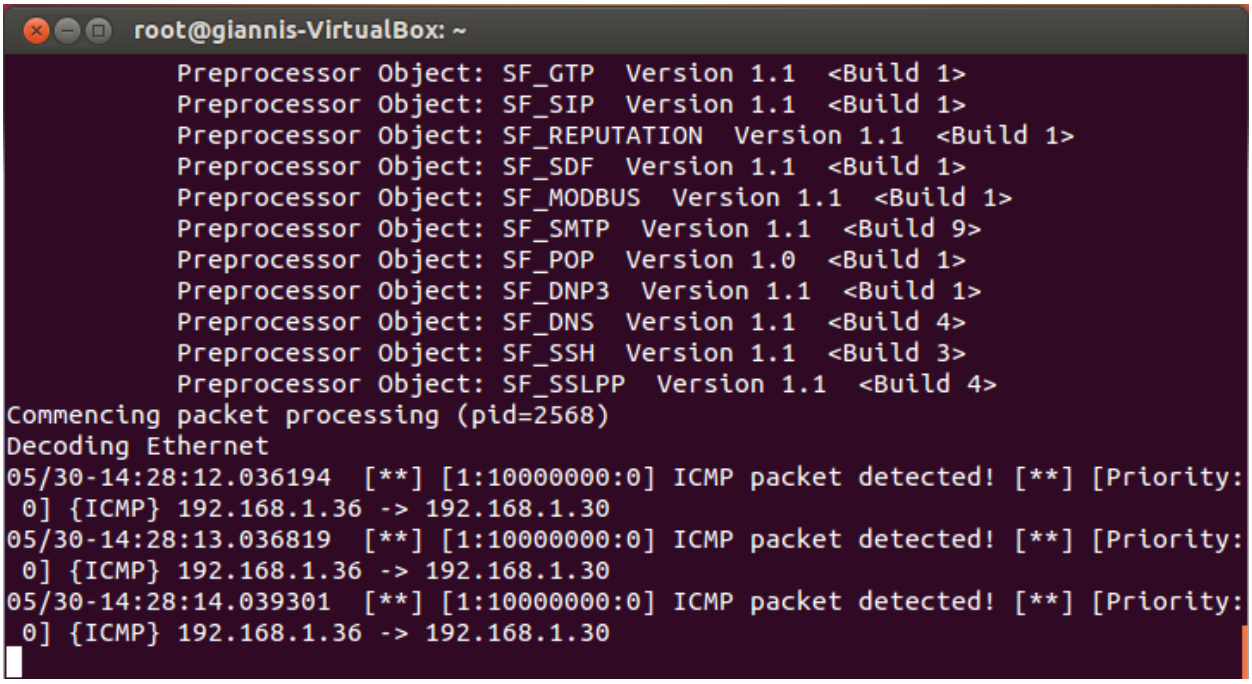
Σε αυτό το σενάριο κάναμε ping προς το θύμα με αποτέλεσμα όλα τα request να παραδοθούν κανονικά και το Snort να παράγει μόνο Alerts.

Παρακάτω φαίνεται το Ping που εκτελέσαμε από τον attacker προς το victim:



Εικόνα 69: Επιτυχείς εκτέλεση Ping προς το θύμα

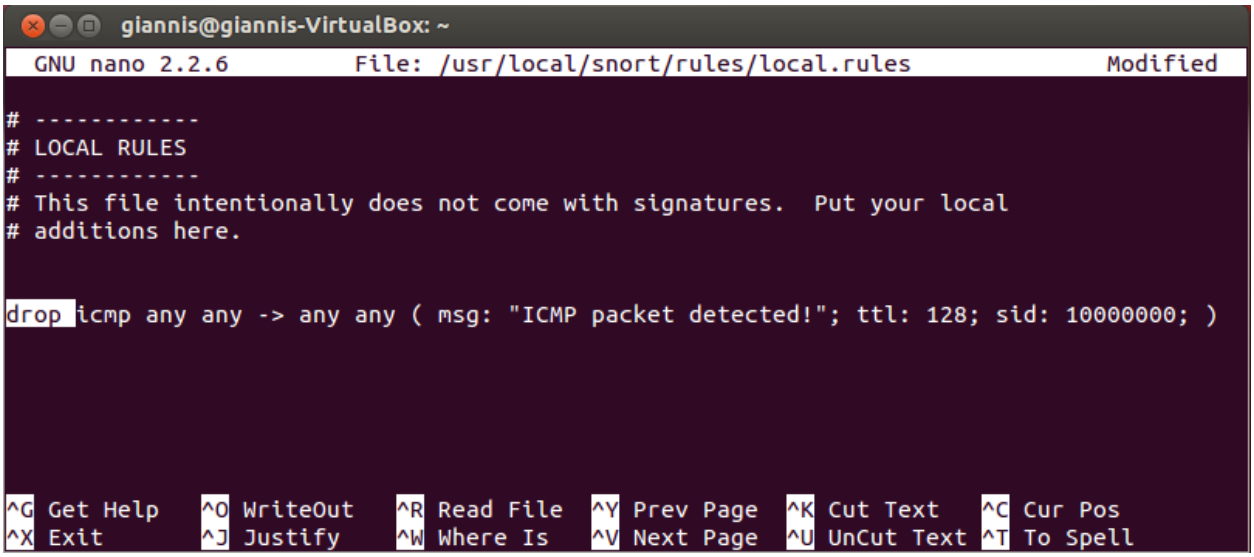
Επίσης παρακρατώ βλέπουμε τα Alerts από τον κανόνα που γράψαμε για την ανίχνευση των Ping:



Εικόνα 70: Ανίχνευση πακέτων απο το Ping του attacker

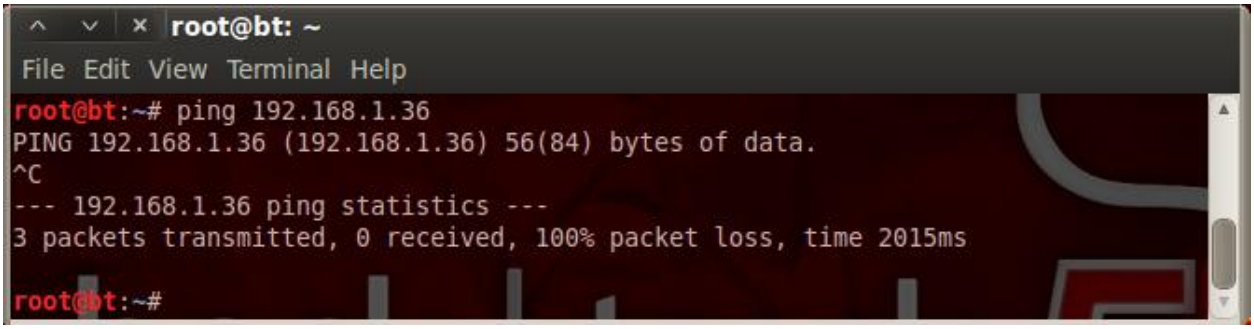
Αποτροπή Ping προς το θύμα

Για να πετύχουμε την αποτροπή του Ping από το θύμα και το Snort να λειτουργήσει σαν IPS θα πρέπει να αλλάξουμε τον κανόνα, ώστε το Snort να μην πυροδοτεί κάποιο Alert, αλλά να απορρίπτει αυτές τις αιτήσεις. Αυτό το πετυχαίνουμε έχοντας το Snort σε inline λειτουργία όπως το έχουμε ρυθμίσει στην παρούσα φάση και αλλάζοντας τον κανόνα αντί για Alert να είναι Drop:



Εικόνα 71: Αλλαγή κανόνα ώστε να αποτρέπονται τα πακέτα ICMP για Ping

Έτσι θα παρατηρήσουμε ότι αν κάνουμε Ping προς το θύμα κανένα πακέτο δε θα φτάσει :



Εικόνα 72: Αποτυχία εκτέλεσης Ping προς το victim

Επίσης στην κονσόλα του Snort θα δούμε αυτά τα πακέτα να γίνονται απορρίπτονται και δε θα φτάσουν ποτέ στον τελικό χρήστη.

Με τον ίδιο τρόπο μπορούμε να αλλάξουμε κάθε κανόνα του Snort από Alert σε Drop ώστε τα πακέτα να απορρίπτονται από το Snort και να μην μπορεί ο επιτηθέμενος να εκτελέση κάποια επίθεση. Πάντα όμως πρέπει να προσέχουμε οι κανόνες που χρησιμοποιούμε να είναι αξιόπιστοι και ελεγμένοι.

5.4.2 Αποτροπή επίθεσης SYN Flood σε Apache Server

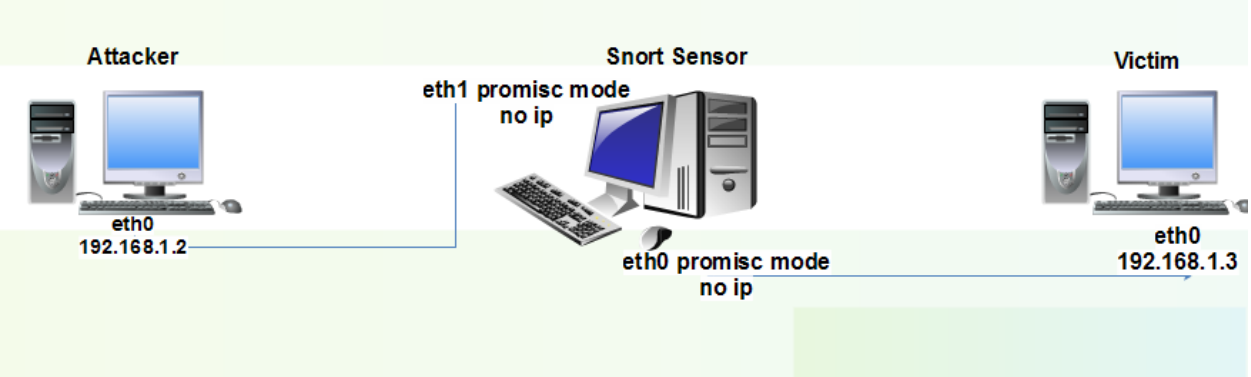
Σε αυτό το παράδειγμα εκτελέσαμε την επίθεση SYN Flood, ώστε να ρίξουμε τον Apache Server που φιλοξενεί την ιστοσελίδα μας. Η επίθεση είναι παρόμοια με αυτή που περιγράψαμε στο προηγούμενο κεφάλαιο. Για να πραγματοποιήσουμε αυτό το σενάριο χρησιμοποιήσαμε τους παρακάτω Hosts:

Attacker: Backtrack 5 R3 Linux

Snort Sensor: Snort inline mode on Ubuntu Linux

Victim: Windows xp sp3 with Apache2

Να σημειώσουμε ότι το Snort το τρέξαμε σε inline λειτουργία και συγκεκριμένα σε Afpacket, όπως την περίπτωση αποτροπής του Ping που περιγράψαμε στην προηγούμενη ενότητα.



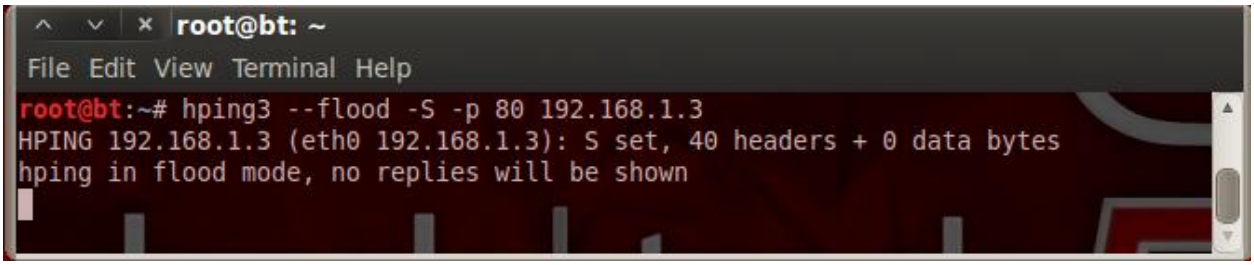
Εικόνα 73: Σχήμα επίθεσης SYN Flood σε Apache Server

Αρχικά εκτελέσαμε την επίθεση χωρίς να θέσουμε σε λειτουργία τον κανόνα, που αποτρέπει μία τέτοια επίθεση. Έτσι καταφέραμε να ρίξουμε τον Apache Server και η ιστοσελίδα μας, που τρέχει στον Apache Server να μην μπορεί να εμφανιστεί στο browser μας λόγο του μεγάλου φόρτου αιτήσεων από την επίθεση SYN Flood. Τελικά καταφέραμε με την βοήθεια του Snort να αποτρέψουμε αυτή την επίθεση και ο Apache Server να λειτουργεί κανονικά.

Στη παρακάτω εικόνα εκτελέσαμε SYN Flood προς τον Apache Server. Στείλαμε όσο ποιο πολλές αιτήσεις γίνεται προς αυτόν. Αυτό το πετύχαμε με τον attacker και την βοήθεια του **hping3** γράφοντας την εντολή:

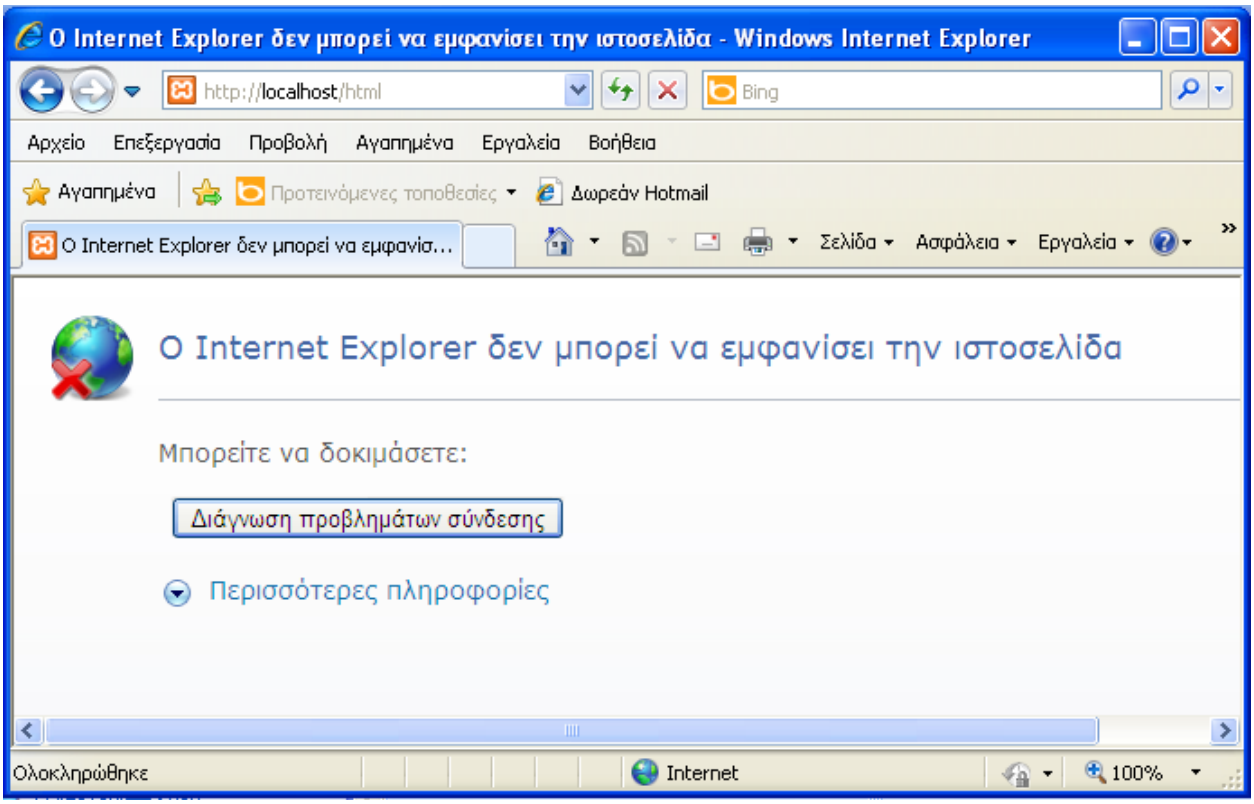
hping3 --flood -S -p 80 192.168.1.3

- **--flood** : θα στέλνει όσα ποιό πολλά πακέτα μπορεί η κάρτα δικτύου
- **-S** : Το TCP πακέτο να έχει flag SYN
- **-p 80** : Τα πακέτα να στέλνονται στη θύρα 80



Εικόνα 74: Εκτέλεση SYN Flood προς τον Apache Server

Μετά την εκτέλεση αυτής της επίθεσης μεγάλος αριθμός από αιτήματα σύνδεσης θα κατακλύσουν τον Apache Server που δέχεται τέτοια αιτήματα στην πόρτα 80. Στο παρακάτω σχήμα φαίνεται η αδυναμία του Server να ανταποκριθεί στο αίτημα μας με αποτέλεσμα να μην μπορεί να μας εμφανίσει την ιστοσελίδα μας:

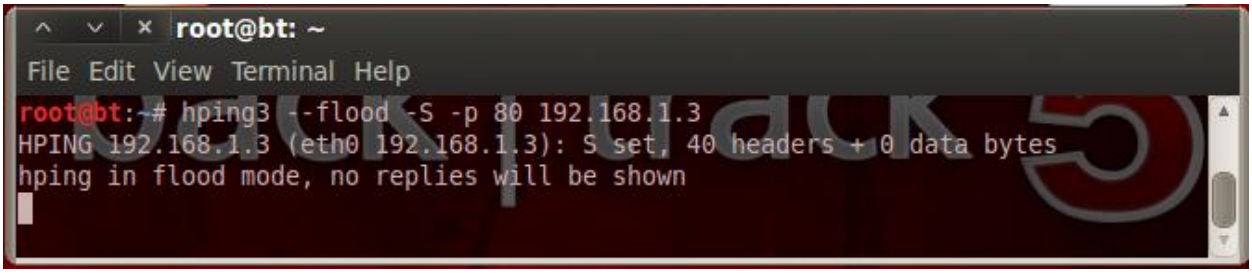


Εικόνα 75: Αδυναμία ανταπόκρισης από Apache Server

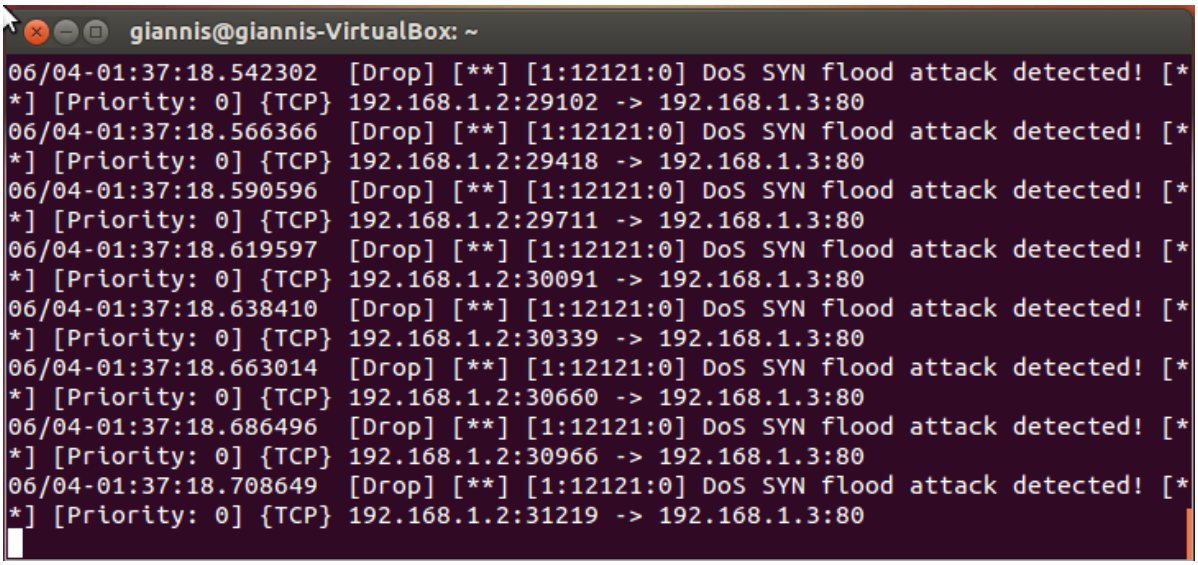
Στη δεύτερη φάση του σεναρίου μας ενημερώσαμε τον κανόνα του Snort που ανιχνεύει την επίθεση SYN Flood να έχει απόκριση και όταν ανιχνεύει μία τέτοια κίνηση μέσα στο δίκτυο να την απορρίπτει. Όπως είπαμε και παραπάνω για να το πετύχουμε αυτό πρέπει να ενημερώσουμε τον αντίστοιχο κανόνα να κάνει drop τα πακέτα που ανιχνεύει αλλάζοντας την λειτουργία μπροστά του κανόνα σε drop.

Εικόνα 76: Ρύθμιση κανόνα να απορρίπτει τα πακέτα από SYN Flood

Αφού θέσουμε σε λειτουργία ένα τέτοιο κανόνα μπορούμε να τρέξουμε πάλι το Snort. Τώρα αν τρέξουμε πάλι την συγκεκριμένη επίθεση από τον Attacker:

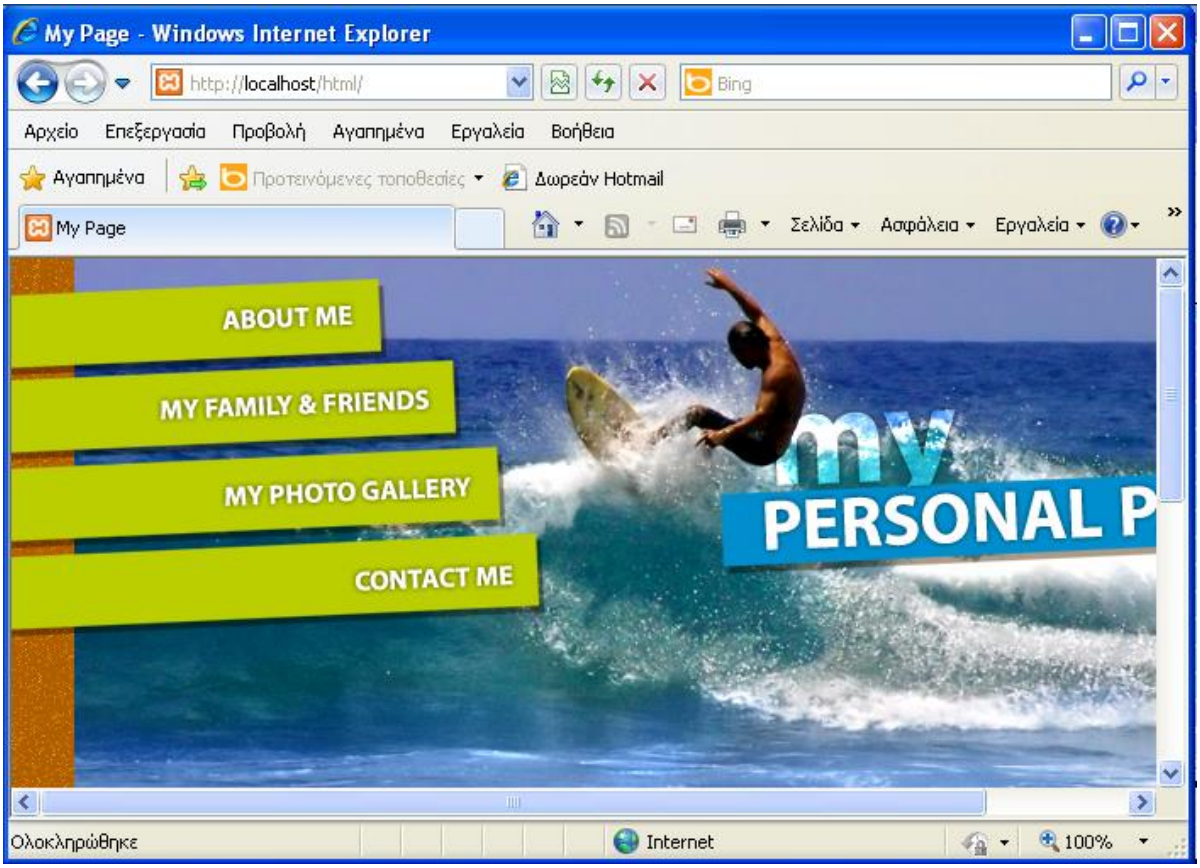


Αρχικά το Snort θα παράγει ένα σύνολο από καταγραφές με πακέτα που έχουν απορριφθεί από μία τέτοια επίθεση:



Εικόνα 77: Αποτροπή πακέτων από επίθεση SYN Flood

Τελικά αν ελέγξουμε την σελίδα μας θα παρατηρήσουμε ότι δεν υπάρχει κανένα πρόβλημα και ο επιτιθέμενος έχει αποτύχει:



Εικόνα 78: Κανονική λειτουργία Apache Server μετά από επίθεση SYN Flood

Βιβλιογραφία

[1] Andrew R. Baker and Joel Esler, 2007, Snort IDS and IPS Toolkit

[2] Christopher Gerg and Kerry J. Cox, 2004, Managing Security with Snort and IDS Tools

[3] Rafeeq Ur Rehman, 2003, Intrusion Detection with SNORT

[4] Charlie Scott, Paul Wolfe and Bert Hayes, 2004, Snort For Dummies

[5] Stephen Northcutt and Judy Novak, 2002, Network Intrusion Detection

[6] Ali A. Ghorbani , Wei Lu and Mahbod Tavallaee, 2009, Network Intrusion Detection and Prevention: Concepts and Techniques

[7] Chris Murphy, 2012, An Analysis of the Snort Data Acquisition Modules , http://www.sans.org/reading_room/whitepapers/detection/analysis-snort-data-acquisition-modules_34027

[8] Snort Web Page, http://snort.org/

[9] Snort Team, 2013, SNORTR Users Manual 2.9.4, http://s3.amazonaws.com/snort-org/www/assets/166/snort_manual.pdf

[10] David Gullet, 2012, Instalation Guide: Snort 2.9.3.0 on Ubuntu 12.04 LTS, http://www.snort.org/assets/158/snortinstallguide293.pdf