Κεφάλαιο 2

Πηγές δεδομένων του Honeynet

Πηγές Δεδομένων του Honeynet

Οι πηγές δεδομένων, είναι εξαρτήματα του μηχανισμού **Data Capture** που μας παρέχουν στοιχεία για κάθε γεγονός που διαδραματίζεται μέσα στο **Honeynet**. Με την ανάλυση των δεδομένων αυτών εξάγουμε πληροφορίες για τις μεθόδους και τα εργαλεία των **blackhats** αλλά και για κάθε άλλη απειλή (virus, worms, DDoS attacks) που αλληλεπιδρά με το **Honeynet**

Υπάρχουν τρεις πηγές δεδομένων, η αλλιώς τρία επίπεδα του **Data Capture**. Η πρώτη πηγή είναι το **firewall** και δίνει σαν δεδομένα τα **firewall logs**.
Η δεύτερη πηγή δεδομένων είναι το **IDS**. Το IDS έχει δύο κρίσιμες λειτουργίες, το **sniffing** , δηλαδή το να καταγράφει όλη την εισερχόμενη και εξερχόμενη δικτυακή κίνηση του honeynet, και την α**νίχνευση επιθέσεων** (intrusion detection) . Μέσα από μια βάση αποτυπωμάτων (Signatures) που διατηρεί, καταφέρνει να επισημαίνει τα πακέτα που θεωρούνται ύποπτα για γνωστές επιθέσεις.
Τα δεδομένα που μας παρέχει είναι, τα **snort IDS logs**, το **full network traffic binary capture** και τα **ASCII decoded network sessions files.**
Η τρίτη πηγή δεδομένων, βασίζεται σε καταγραφή στοιχείων που προκύπτουν από κάθε **honeypot**. Τα δεδομένα που αντλούμε από αυτό το επίπεδο (ή πηγή), είναι τα system logs , για παράδειγμα τις διεργασίες που εκτελέστηκαν ή τις συνδέσεις που γίνανε. Τα application logs είναι η δεύτερη κατηγορία δεδομένων και μας δείχνει στοιχεία για την κατάσταση των εφαρμογών.
Τέλος τα **keystrokes** (πληκτρολογήσεις) είναι τα επόμενα δεδομένα που συλλέγουμε από κάθε **honeypot**. Τα δεδομένα κάθε πηγής περιέχουν πληροφορίες που περιγράφουν τα γεγονότα που εξελίσσονται στο **Honeynet** από διαφορετική οπτική γωνία. **Η συσχέτιση** των πληροφοριών αυτών μας δίνει την ολοκληρωμένη εικόνα. Επίσης η πολλαπλότητα των πηγών και η κατά μέρος αλληλοεπικαλυπτόμενη πληροφορία που μπορεί να εξαχθεί από αυτές, παρέχουν ένα πλεονασμό που μας επιτρέπει να έχουμε καλή εικόνα των γεγονότων, ακόμα και όταν κάποια από αυτές αστοχήσει. Οι διαφορετικές πηγές δεδομένων, μας παρέχουν υλικό για να κάνουμε τις αναλύσεις και να βγάλουμε αξιόλογα συμπεράσματα. Κάθε πηγή δεδομένων καλείται να καλύψει την περίπτωση αποτυχίας κάποιας άλλης πηγής και κάθε πηγή συνδέει ένα διαφορετικό κομμάτι

του παζλ του **honeynet**. Συνδυάζοντας την πληροφορία από κάθε πηγή συνθέτουμε μια ολοκληρωμένη εικόνα για κάθε δραστηριότητα.

Πρώτο επίπεδο – Firewall

Firewall logs.

Τα **Firewall logs** είναι οι πληροφορίες που μας δίνει το **Firewall** για την κίνηση στο **honeynet**. Σε ένα honeynet δεύτερης γενιάς (GenII) που είναι ο μόνος τύπος **honeynet** που χρησιμοποιείται από το Ελληνικό έργο **honeynet**, το **Firewall** υλοποιείται με τα **IPTbales** , τα οποία ρυθμίζονται για αυτή την λειτουργία στο **script *rc.firewall*** του **honeywall**. Μια από τις λειτουργίες του **firewall** είναι και η καταγραφή της έναρξης των νέων εισερχομένων και εξερχόμενων συνδέσεων στο αρχείο */var/log/messages*. Το αρχείο αυτό περιέχει τα γεγονότα που συμβαίνουν στο σύστημα, ένα μικρό κομμάτι μπορούμε να δούμε στον παρακάτω πίνακα 2-1

```
.......
Mar  8 18:25:00 bilem3 kernel: OUTBOUND CONN UDP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=194.177.211.5 DST=66.187.233.4 LEN=76 TOS=0x10 PREC=0x00 TTL=64 ID=0 DF PROTO=UDP SP
T=123 DPT=123 LEN=56
Mar  8 18:28:17 bilem3 su(pam_unix)[15066]: session opened for user news by (uid=0)
Mar  8 18:28:18 bilem3 su(pam_unix)[15066]: session closed for user news
Mar  8 18:33:31 bilem3 kernel: OUTBOUND CONN UDP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=194.177.211.5 DST=66.187.233.4 LEN=76 TOS=0x10 PREC=0x00 TTL=64 ID=0 DF PROTO=UDP SP
T=123 DPT=123 LEN=56
Mar  8 18:42:03 bilem3 kernel: OUTBOUND CONN UDP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=194.177.211.5 DST=66.187.233.4 LEN=76 TOS=0x10 PREC=0x00 TTL=64 ID=0 DF PROTO=UDP SP
T=123 DPT=123 LEN=56
.....
```

Πίνακας 2-1 - /var/log/messages

Ας πάρουμε για παράδειγμα το ακόλουθο μήνυμα του **firewall**:

Feb 11 12:07:03 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth2 OUT=br0 PHYSOUT=eth1 SRC=194.135.57.8 DST=194.177.211.3 LEN=48 TOS=0x00 PREC=0x80 TTL=109 ID=6988 DF PROTO=TCP SPT=3506 DPT=3127 WINDOW=16384 RES=0x00 SYN URGP=0

Πίνακας 2-2 – firewall log

Αυτό το μήνυμα μας ενημερώνει για την εκκίνηση μιας νέας εισερχόμενης TCP σύνδεσης. Περιέχει πληροφορίες για αυτήν την σύνδεση όπως η ημερομηνία και η ώρα , το όνομα του υπολογιστή, ποιο στοιχείο παρήγαγε το log (kernel για firewall logs), ποιες είναι οι IP και πόρτες πηγής και προορισμού. Δηλαδή έχουμε ένα εισερχόμενο TCP πακέτο από την ip 194.135.57.8 και πόρτα 3506 στο honeypot ip 194.177.211.3 στην πόρτα 3127.
Ποιο αναλυτικά, ας πάρουμε ένα ένα τα πεδία από το παραπάνω **firewall log**

Πεδίο	***Πρωτόκολλο***	***Σχόλια***
Feb 11 12:07:03	IP	Η ημερομηνία και η ώρα
bridge kernel	OS	Το hostname και στοιχείο που παρήγαγε το log (kernel)
INBOUND TCP	IP	Εισερχόμενη σύνδεση TCP
IN=br0 PHYSIN=eth2	IP	Το πακέτο εισήλθε από το interface eth2 του bridging
OUT=br0 PHYSOUT=eth1	IP	Και έφευγε από το interface eth1
SRC=194.135.57.8	IP	Η IP που έστειλε το πακέτο
DST=194.177.211.3	IP	Η IP προορισμού
LEN=48	IP	Το μέγεθος όλου του IP πακέτου
TOS=0x00	IP	Type-Of-Service . Το TOS έχει normal τιμή 0x00, σπάνια αλλάζει από την default τιμή.
PREC=0x80	IP	Προσδιορίζει την προτεραιότητα του IP πακέτου κατά την δρομολόγηση του.
TTL=109	IP	Time-To-Live. Ο χρόνος σε sec που απομένει στο πακέτο να κινείται στο Internet, πριν χαρακτηριστεί τελειωμένο με TTL = 0.
ID=6988	IP	Ο αριθμός αναγνώρισης του πακέτου (χρησιμεύει στο de-fragmentation).
DF	IP	Do-Not-Fragment bit. Ο αποστολέας έχει δηλώσει απαγόρευση του fragmentation
PROTO=TCP	IP	Το transport layer πρωτόκολλο είναι TCP
SPT=3506	TCP	Η Source-Port του αποστολέα
DPT=3127	TCP	Η Destination-Port του παραλήπτη
WINDOW=16384	TCP	Το μέγεθος του buffer που έχει ορίσει το software του TCP για το μέγεθος των data που θα δέχεται.
RES=0x00	TCP	Αυτό είναι ένα πεδίο των 6-bit και κρατείται για μελλοντική χρίση

Πεδίο	*Πρωτόκολλο*	*Σχόλια*
SYN	TCP	Ορίζει τον σκοπό του πακέτου. Στην συγκεκριμένη περίπτωση ορίζει την εκκίνηση μιας TCP σύνδεσης. Προσδιορίζεται από flag ενός bit.
URGP=0	TCP	TCP URGENT Flag για να προωθηθεί κατά προτεραιότητα.

Πίνακας 2-3 – ανάλυση δομής του Firewall log (τεχνική αναφορά noc Ariadne-t 2000)

Που μπορούν να βρούνε όμως παραγωγική εφαρμογή αυτά τα logs; Η γνώση για τον ακριβή χρόνο όλων των συνδέσεων που πραγματοποιήθηκαν στο **Honeynet** είναι απαραίτητη για να μπορέσουμε να ανακαλύψουμε την αλληλουχία των γεγονότων. Πολλές φορές κοιτώντας μόνο τα **firewall logs** μπορούμε να καταλάβουμε ότι ένα **Honeypot** βρίσκεται κάτω από τον έλεγχο ενός **blackhat**. Για παράδειγμα, στον παρακάτω πίνακα 2-4, βλέπουμε στο πρώτο **firewall log** ότι γίνεται μια εισερχόμενη σύνδεση προς το **honeypot** στην πόρτα 21 (**ftp**). Στην συνέχεια φαίνεται στα 2ο ,3ο και 4ο Log, το **honeypot** να κάνει τέσσερις συνδέσεις προς την **IP** που έκανε στο πρώτο log την εισερχόμενη σύνδεση προς την πόρτα 20.

```
May 23 05:58:50 bilem3 kernel: INBOUND TCP: IN=br0 PHYSIN=eth2 OUT=br0 PHYSOUT=eth1
SRC=217.227.22.164 DST=192.168.0.1 LEN=52 TO
S=0x00 PREC=0x80 TTL=114 ID=23646 DF PROTO=TCP SPT=3649 DPT=21 WINDOW=32767 RES=0x00
SYN URGP=0
May 23 05:58:53 bilem3 kernel: OUTBOUND CONN TCP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=192.168.0.1 DST=217.227.22.164 LEN
=48 TOS=0x00 PREC=0x00 TTL=128 ID=16550 DF PROTO=TCP SPT=20 DPT=3650 WINDOW=16384
RES=0x00 SYN URGP=0
May 23 05:58:57 bilem3 kernel: OUTBOUND CONN TCP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=192.168.0.1 DST=217.227.22.164 LEN
=48 TOS=0x00 PREC=0x00 TTL=128 ID=16560 DF PROTO=TCP SPT=20 DPT=3651 WINDOW=16384
RES=0x00 SYN URGP=0
May 23 05:59:01 bilem3 kernel: OUTBOUND CONN TCP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=192.168.0.1 DST=217.227.22.164 LEN
=48 TOS=0x00 PREC=0x00 TTL=128 ID=16572 DF PROTO=TCP SPT=20 DPT=3652 WINDOW=16384
RES=0x00 SYN URGP=0
May 23 05:59:07 bilem3 kernel: OUTBOUND CONN TCP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2
SRC=192.168.0.1 DST=217.227.22.164 LEN
=48 TOS=0x00 PREC=0x00 TTL=128 ID=16584 DF PROTO=TCP SPT=20 DPT=3653 WINDOW=16384
```

Πίνακας 2-4

Το **honeypot**, υπό κανονικές συνθήκες, δεν κάνει συνδέσεις προς τα έξω. Άρα κάτι ύποπτο συμβαίνει, από αυτό το παράδειγμα μπορούμε να καταλάβουμε, ότι η IP 217.227.22.164, προκάλεσε μία ενέργεια όπου σαν αποτέλεσμα είχε να στέλνει, το **honeypot**, πακέτα για εξωτερική σύνδεση. Μετά από αυτό ξέρουμε ότι αυτή η IP που επικοινώνησε με το **honeypot**, κατάφερε να προκαλέσει μια εξερχόμενη σύνδεση από το **honeypot** και τον χρόνο που έγινε αυτό. Αυτή η πληροφορία είναι ικανή για να υποψιαστούμε ότι πρόκειται για επιτυχημένη επίθεση και να συνεχίσουμε την διερεύνηση σε βάθος. Η αξία λοιπόν των firewall logs, είναι περισσότερο για το λεγόμενο **transcation recording**, δηλαδή η γνώση του πότε και ποιος επικοινώνησε με ποιόν. Τα **firewall logs** τα λαμβάνουμε από ένα εξάρτημα (λογισμικό) του **Honeynet**, το ***swatch***. Με το πρόγραμμα ***swatch*** καταφέρνουμε να έχουμε σχεδόν online ενημέρωση για ύποπτες κινήσεις στο **honeynet**. Το **swatch** δουλεύει ως εξής, διαβάζει ένα **ascii** αρχείο και αν σε μια γραμμή εντοπίσει κάποια λέξη που θα ορίσουμε εμείς, στέλνει με email την εν λόγο γραμμή. Έτσι μπορούμε να ρυθμίσουμε το **swatch** όταν εντοπίζει στο */var/log/message* σύνδεση προς τα έξω (απλά την λέξη OUTBOUND), να μας ενημερώνει με ένα email. Με αυτό τον τρόπο μπορούμε να ενημερωθούμε με ένα email ότι ξεκινάει μια σύνδεση προς τα έξω, η οποία φυσικά δεν είναι φυσιολογική, διότι το σύστημα που ζητάει την σύνδεση OUTBOUND είναι όπως ξέρουμε ένα **honeypot**, και ότι αλληλεπιδρά με αυτό είναι ύποπτο, πόσο μάλλον όταν το ίδιο το **honeypot** ζητά σύνδεση προς τα έξω ενώ δεν υπάρχει κάποιος φυσιολογικός χρήστης να αλληλεπιδρά με αυτό. Αρά κατά ενενήντα τις εκατό κάποιος έχει παραβιάσει το **honeynet** και προσπαθεί μέσα από αυτό να κάνει κάποιες συνδέσεις προς τα έξω από το **honeynet**, είτε για να κατεβάσει κάποια εργαλεία, είτε να αναζητήσει ή να επιτεθεί σε άλλα μηχανήματα ή και για άλλους λόγους. Αυτή η μέθοδος της συλλογής των **firewall logs** σε συνδυασμό με το **swatch** κάνει πολύ πιο γρήγορο τον εντοπισμό, τουλάχιστον των σημαντικότερων γεγονότων που εξελίσσονται μέσα στο **honeynet**.

Δεύτερο επίπεδο - Snort IDS Logs

Μία άλλη πηγή δεδομένων είναι τα **logs**, που παράγει το **intrution dedection system (IDS- για περισσότερες πληροφορίες βλέπε πτυχιακή Ιωάννη Παπαπάνου κεφάλαιο 2) snort**, όταν αυτό ρυθμιστεί να προειδοποιεί για ότι θεωρεί ύποπτο, αλλά και για να καταγράφει ότι πακέτο κινείται,

λειτουργώντας δηλαδή σαν **sniffer** . Τα **logs** του **snort** είναι μια ακόμη πηγή δεδομένων, που συμπληρώνει την πληροφορία που συγκεντρώνουμε από το διάβασμα των **firewall logs**.

Το **snort** μπορεί να παράγει τρεις διαφορετικούς τύπους Log που θα τους δούμε αναλυτικά στην συνέχεια. Οι τύποι αυτοί είναι:

- Τα **snort alerts** , είναι μηνύματα που παράγει το **IDS** όταν καταλαβαίνει ότι μπορεί να γίνεται επίθεση, **scanning** ή και κάποια άλλη μη φυσιολογική κίνηση. Τα **alerts** μπορούν να παραχθούν με δύο βασικές μορφές (alert fast, alert full).
- **Full Network traffic Binary capture,** δηλαδή, καταγραφή όλης της δικτυακής κίνησης σε δυαδική μορφή.
  - Και **ascii decoded sessions**, καταγράφει τις συνόδους (**SESSIONS**) δηλαδή την ανταλλαγή πληροφοριών σε μορφή **ASCII** χαρακτήρων.

Alert snort logs

Προς το παρόν θα ασχοληθούμε με τα **alert snort logs**. Τα **alerts** καταγράφονται με τρεις μορφές.

Καταγραφή των snort alerts στο /var/log/messages του συστήματος όπως και στα **firewall logs.** Αν παρατηρήσουμε ένα log που καταγράφηκε από το **snort** θα δούμε ότι μοιάζει με ένα **firewall log** βέβαια στην πληροφορία για το στοιχείο που δημιούργησε το log υπάρχει η λέξη **snort** και ακόμα ανάλογα με την δικτυακή δραστηριότητα προσδιορίζει την σημαντικότητα των κινήσεων και εκτός από την περιγραφή μας εμφανίζει την προτεραιότητα με το πεδίο Priority δείχνοντας την σημασία του alert.

```
Feb 12 02:04:07 redhat7_3 snort: [1:491:6] INFO FTP Bad login [Classification: Potentially Bad Traffic] [Priority: 2]: {TCP} 172.16.98.10:21 -> 192.168.0.135:32934
```

Πίνακας 2-5 - alert snort logs

Στο παράδειγμα που φαίνεται παραπάνω, έχουμε μία αποτυχημένη προσπάθεια για login σε έναν **FTP server** (port 21).Η προτεραιότητα αυτού του γεγονότος είναι δύο, δηλαδή αυξημένη στην κλίμακα 1-4 που διαθέτει το snort. Βεβαίως, αποτυχημένη προσπάθεια πρόσβασης μπορεί να έκανε και ένας εξουσιοδοτημένος χρήστης γράφοντας για παράδειγμα λάθος το **password** του , αλλά θα μπορούσε να ήταν και ένας επιτιθέμενος που πραγματοποιούσε επίθεση με την τεχνική

της ανεύρεσης εύκολων κωδικών, δοκιμάζοντας δηλαδή πολλούς συνδυασμούς προβλέψιμων κωδικών όπως user1, user123, admin123 και ου το καθεξής. Τα alerts με priority 3 είναι λιγότερο σημαντικά όπως κάποιο **ping**, ενώ τα alerts με priority 1, είναι τα πλέον σημαντικά όπως η γνωστή στο **snort** συμπεριφορά κάποιου **worm**.

Εκτός από τα **logs** στο /var/log/messages το **snort** δημιουργεί δύο αρχεία με τα alerts που παράγει, το snort_fast και το snort_full. Αυτά τα αρχεία δημιουργούνται στο directory που έχει ρυθμιστεί για να καταγράφονται τα alerts του snort.

Κάθε alert που καταγράφεται στο snort_fast μοιάζει πολύ με αυτά που καταγράφονται στο / var/log/messages. Παρακάτω βλέπουμε το ίδιο γεγονός που καταγράφηκε και στο / var/log/messages (Πίνακας 2-5) πως καταγράφηκε στο snort_fast. Παρατηρούμε μικρές διαφορές, όπως την δομή ημερομηνίας και ώρας και ότι στο snort_fast λείπει η πληροφορία για το ποιο στοιχείο παρήγαγε το alert, αφού σε αυτό το αρχείο καταγράφονται μόνο **snort alerts**.

```
02/12-02:04:07.954148 [**] [1:491:6] INFO FTP Bad login [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.16.98.10:21 -> 192.168.0.135:32934
```

Πίνακας 2-6

Ταυτόχρονα στο snort_full καταγράφονται περισσότερες πληροφορίες όπως το μέγεθος, τον τύπο του πακέτου και άλλα. Παρακάτω (πίνακας 2-7) βλέπουμε το alert που κατέγραψε το snort στο snort_full για το ίδιο γεγονός που σχολιάσαμε παραπάνω (πίνακας 2-6).

```
[**] [1:491:6] INFO FTP Bad login [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
02/12-02:04:07.954148 172.16.98.10:21 -> 192.168.0.135:32934
TCP TTL:64 TOS:0x10 ID:29427 IpLen:20 DgmLen:74 DF
***AP*** Seq: 0xA41CE3BF  Ack: 0x3610CE33  Win: 0x16A0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 110563 755529
```

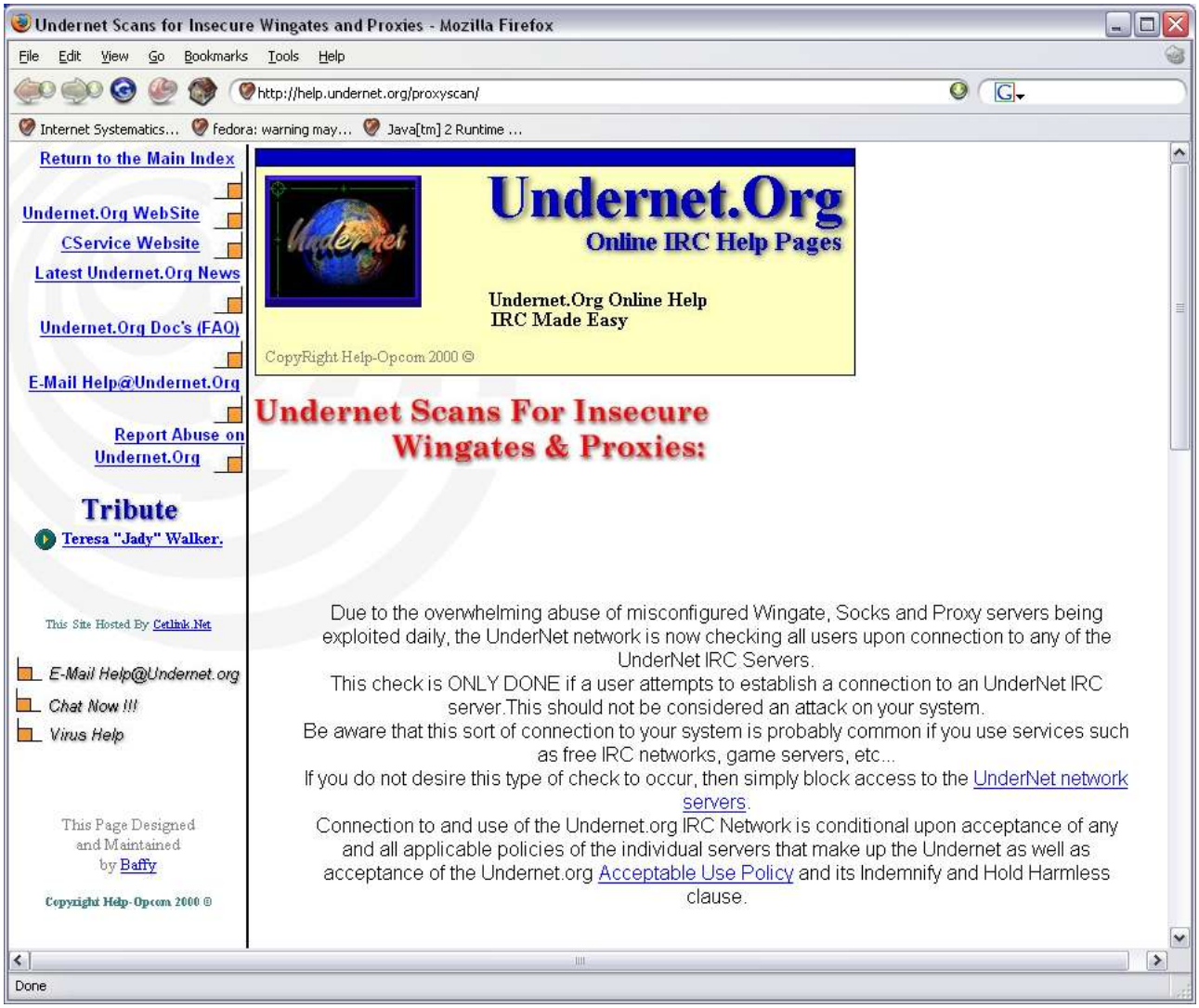
Πίνακας 2-7

Μια χρήσιμη πληροφορία που επισυνάπτεται στα **alerts** του snort_fulleίναι τα **URL's,** βλέπε πίνακα 2-8 .

```
[**] [1:615:5] SCAN SOCKS Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/11-07:19:35.621331 148.240.197.225:3554 -> 194.177.211.6:1080
TCP TTL:110 TOS:0x80 ID:34860 IpLen:20 DgmLen:48 DF
******S* Seq: 0x62E2CFAC  Ack: 0x0  Win: 0xFC00  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
[Xref => http://help.undernet.org/proxyscan/]
```

Πίνακας 2-8

Σε αυτά τα URLs βρίσκουμε πληροφορίες σχετικές με το συγκεκριμένο alert, όπως φαίνεται παρακάτω στην εικόνα 2-1.



Εικόνα 2-1

Full Network traffic Binary capture

Ίσως η σημαντικότερη πηγή δεδομένων είναι το **binary** αρχείο όλης της δικτυακής κίνησης που καταγράφεται σε μορφή **tcpdump** δυαδικού αρχείου (βλέπε κεφάλαιο 4 Πίνακα 4-2). Η παραγωγή αρχείων τύπου **tcpdump** από το **snort**, είναι πολύ βολική. Εκτός από το tcpdump πολλά άλλα προγράμματα μπορούν να διαβάσουν τέτοιου τύπου αρχεία, όπως για παράδειγμα το **ethereal**. Το μέγεθος του δυαδικού αρχείου, εξαρτάται από την κίνηση που έχουμε στο **honeynet** μας και η διαχείριση του, γίνεται δύσκολη όταν το μέγεθος του ξεπεράσει μερικές δεκάδες Mb's. Για να ξεπεράσουμε το πρόβλημα χρησιμοποιούμε προγράμματα όπως το tcpslice, tcpdump και άλλα, για να τεμαχίσουμε το αρχείο ή να εξάγουμε τα κομμάτια που μας ενδιαφέρουν. Το πλεονέκτημα, αυτής της πηγής δεδομένων είναι ότι μπορούμε να έχουμε μία πλήρη εικόνα της κίνησης κάθε πακέτου από και προς το **honeynet** . Όπως είπαμε, αν έχουμε αυξημένη κίνηση το δυαδικό αρχείο είναι αρκετά δύσχρηστο, αλλά σε συνδυασμό με άλλες πηγές δεδομένων μπορεί να γίνει πολύ αποδοτικό.

Ας δούμε ένα παράδειγμα συνδυασμού πηγών δεδομένων για ανάλυση, έστω ότι βλέπουμε ένα **snort IDS alert**, που έχουμε λάβει με email με την διαδικασία που αναφέραμε παραπάνω. Το alert έχει βαθμό προτεραιότητας 1 και είναι αυτό που βλέπουμε παρακάτω

```
Jan 16 01:25:14 bilem3 snort: [1:1250:7] WEB-MISC Cisco IOS HTTP configuration
attempt [Classification: Web Application Attack] [Priority: 1]: {TCP}
213.112.19.220:46966 -> 192.168.0.1:80
```

Πίνακας 2-9

Έχουμε δηλαδή μία επίθεση σε web εφαρμογή (port 80), με τίτλο επίθεσης WEB-MISC Cisco IOS HTTP configuration attempt, με IP διεύθυνση επιτιθέμενου 213.112.19.220.
Ας δούμε λοιπόν πώς αυτή η πληροφορία μας βοηθάει να βρούμε περισσότερες λεπτομέρειες χρησιμοποιώντας το δυαδικό αρχείο (**binary**).

1. Με τη χρήση του **unix** εργαλείου **tcpdump** με παραμέτρους
   -r <όνομα binary> για να διαβάσουμε το **binary**
   -n να εμφανίζεται οι IP νούμερα και όχι ονόματα domain
   net <IP>

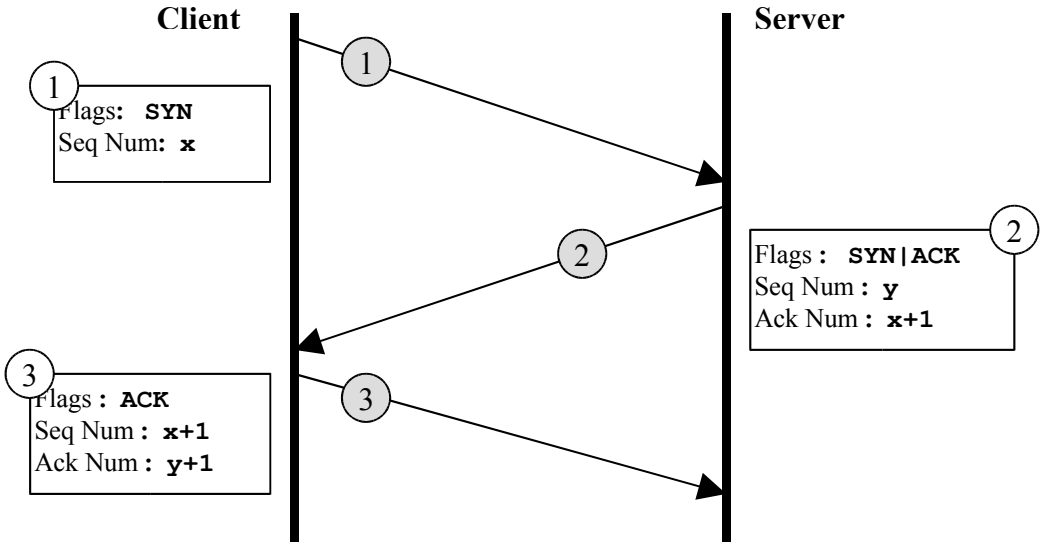
tcpdump -r snort.log.1074211512 -n net 213.112.19.220

Παίρνουμε σαν αποτέλεσμα όλη την κίνηση που έχει καταγραφεί στο **binary** για την IP 213.112.19.220.

```
1.01:25:14.491029 213.112.19.220.46966 > 192.168.0.1.http: S 718711607:718711607(0) win 16384 <mss
1460,nop,nop,sackOK> (DF) [tos 0x80]
2.01:25:14.492183 192.168.0.1.http > 213.112.19.220.46966: S 2823995958:2823995958(0) ack 718711608 win
5840 <mss 1460,nop,nop,sackOK> (DF)
3.01:25:14.610796 213.112.19.220.46966 > 192.168.0.1.http: . ack 1 win 17520 (DF) [tos 0x80]
4.01:25:14.612031 213.112.19.220.46966 > 192.168.0.1.http: P 1:42(41) ack 1 win 17520 (DF) [tos 0x80]
5.01:25:14.612299 192.168.0.1.http > 213.112.19.220.46966: . ack 42 win 5840 (DF)
6.01:25:14.731756 213.112.19.220.46966 > 192.168.0.1.http: P 42:149(107) ack 1 win 17520 (DF) [tos 0x80]
7.01:25:14.732075 192.168.0.1.http > 213.112.19.220.46966: . ack 149 win 5840 (DF)
8.01:25:18.032935 192.168.0.1.http > 213.112.19.220.46966: P 1:573(572) ack 149 win 5840 (DF)
9.01:25:18.067630 192.168.0.1.http > 213.112.19.220.46966: F 573:573(0) ack 149 win 5840 (DF)
10.01:25:18.185084 213.112.19.220.46966 > 192.168.0.1. http: . ack 574 win 16948 (DF) [tos 0x80]
11.01:25:18.185278 213.112.19.220.46966 > 192.168.0.1. http: F 149:149(0) ack 574 win 16948 (DF) [tos 0x80]
12.01:25:18.185489 213.112.19.220.46966 > 192.168.0.1. http: P 2105284925:2105285032(107) ack 2189683095
win 5840 [tos 0x10]
13.01:25:18.185489 192.168.0.1.http > 213.112.19.220.46966: . ack 150 win 5840 (DF)
```

Πίνακας 2-10

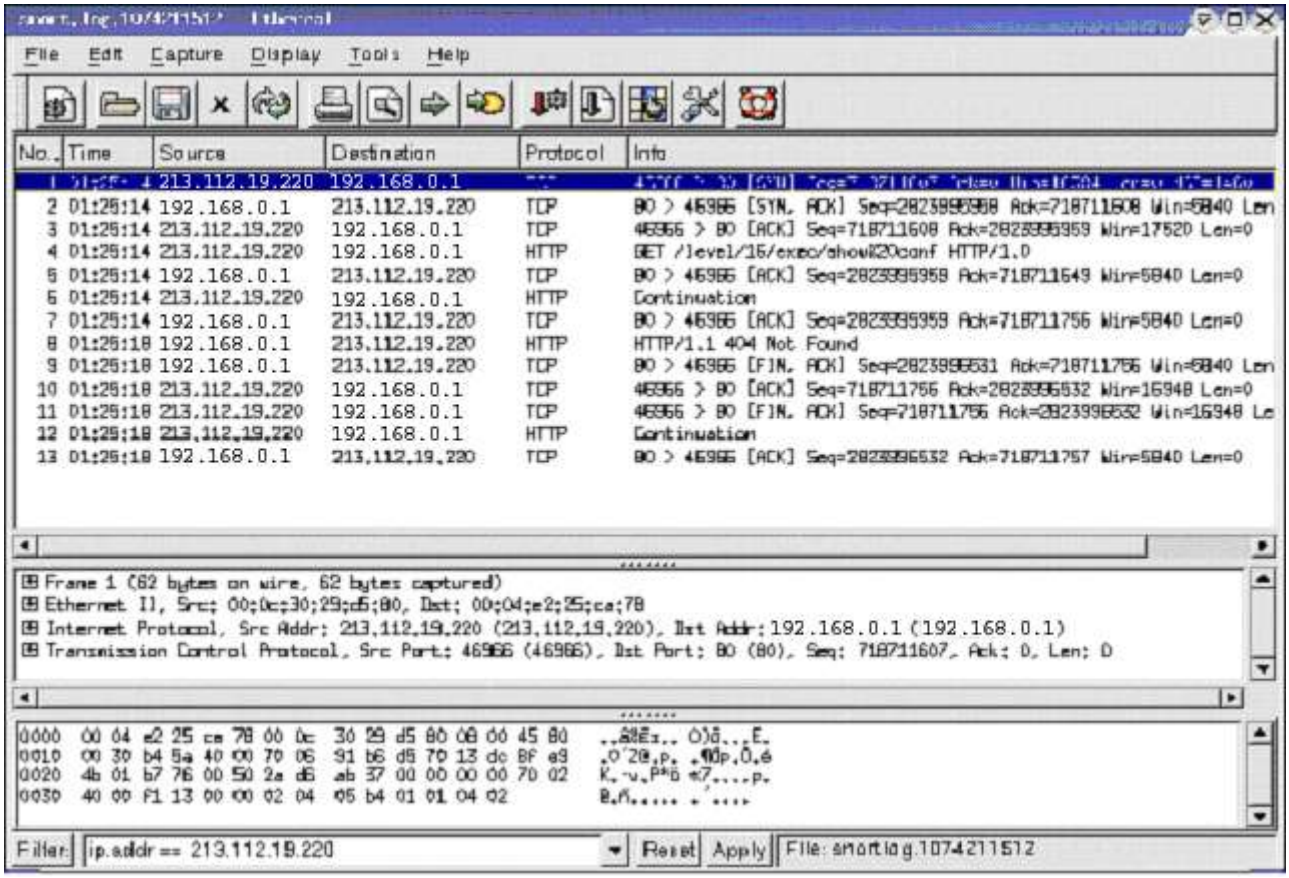
Το αποτέλεσμα είναι 13 πακέτα που συσχετίζονται με την συγκεκριμένη IP. Όπως παρατηρούμε στα τρία πρώτα πακέτα 1,2,3 επιτυγχάνεται η 3-way-handshake διαδικασία, δηλαδή: ο επιτιθέμενος στέλνει ένα SYN πακέτο σε ένα **honeypot** στην πόρτα 80. Αμέσως απαντάει (γραμμή 2) το **honeypot** με SYN-ACK, και ολοκληρώνεται η σύνδεση με ACK που στέλνει ο επιτιθέμενος.



Εικόνα 2-2: 3-Way Handshake – Εγκαθίδρυση σύνδεσης

Στην συνέχεια έχουμε προώθηση data (PUSH) γραμμές 4,5,6,12 και αιτήσεις διακοπής σύνδεσης (FIN) γραμμές 9,11.

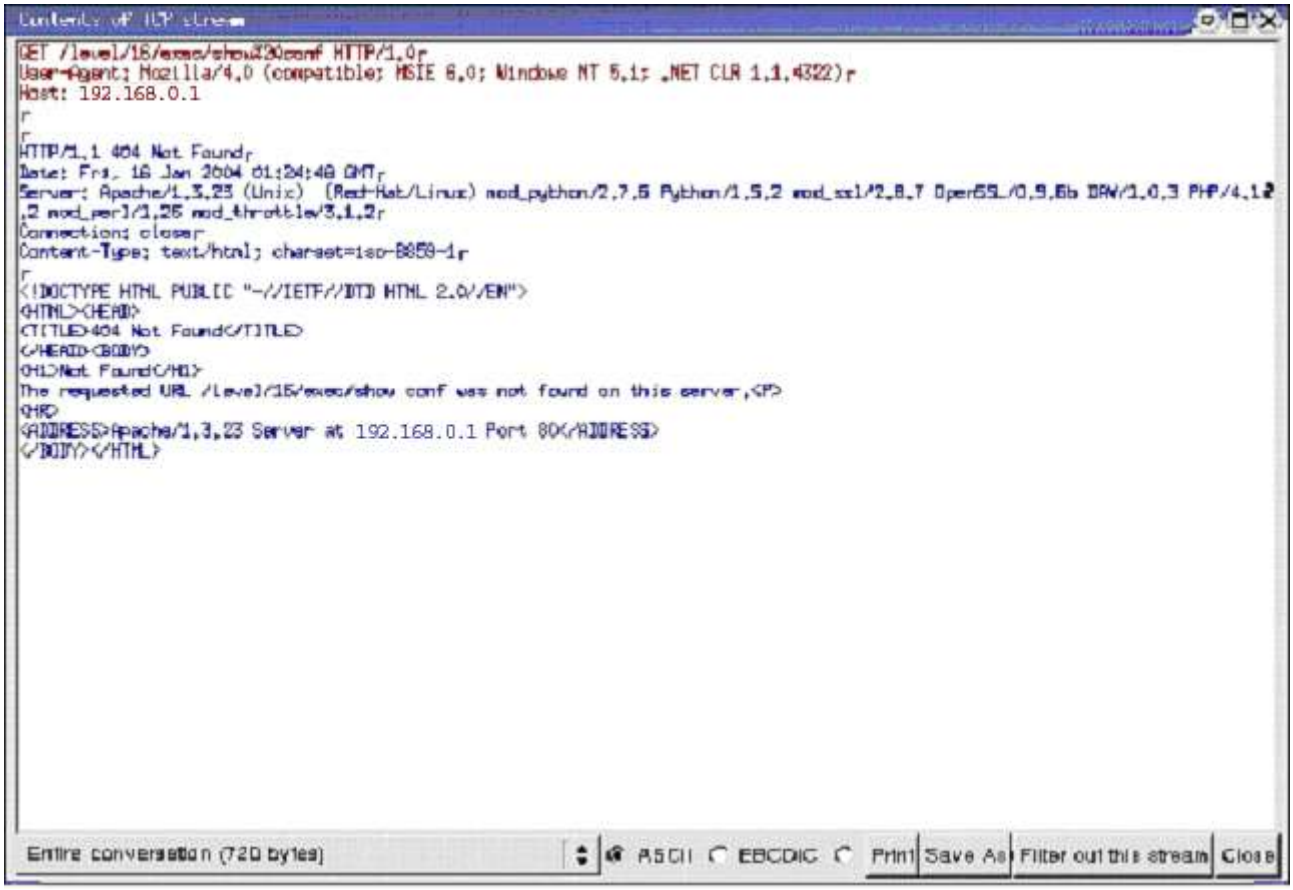
2. Εναλλακτικά μπορούμε να χρησιμοποιήσουμε **GUI (Graphic User Interface)** εφαρμογή για να διαβάζει τα **binary** τύπου **tcpdump** που συνήθως είναι πιο ευκολόχρηστες και εξίσου αποτελεσματικές. Μία τέτοια **GUI** εφαρμογή είναι το **ethereal** . Το **ethereal** είναι ένα εργαλείο αρκετά φιλικό με τον χρήστη, και με πολλές δυνατότητες. Για το προηγούμενο παράδειγμα, μπορούμε να ανοίξουμε το **binary** αρχείο (file -> open) που μας ενδιαφέρει και με την χρήση φίλτρων μπορούμε να έχουμε το αποτελέσματα που θέλουμε. Χρησιμοποιώντας το φίλτρο ip.addr == 213.112.19.220 μας επιστρέφει τα πακέτα που είδαμε και στο παράδειγμα παραπάνω.



Εικόνα 2-3

Η εμφάνιση των πακέτων εδώ είναι περισσότερο ευανάγνωστη. Στην γραμμή 4 βλέπουμε την HTTP εντολή που στάλθηκε (GET) από τον επιτιθέμενο μετά το **3-way-handshake.**

Ακόμα μία λειτουργία που έχει το **ethereal** (http://www.ethereal.com/) είναι , tools -> Follow TCP Stream ή δεξί click και Follow TCP Stream. Με αυτό το εργαλείο του **ethereal** βλέπουμε τι μεταφέρθηκε μέσα στα TCP πακέτα. Στο παράδειγμα μας βλέπουμε στην εικόνα 2-4, τα παρακάτω.



Εικόνα 2-4

Με κόκκινο είναι τα δεδομένα που στέλνει ο επιτιθέμενος ενώ τα μπλε αυτά που απαντάει το **honeypot**. Κρύβουμε τα πραγματικά στοιχεία για να μην αποκαλύπτουμε την τοποθεσία του **honeynet.**

Το μειονέκτημα ενός τέτοιου εργαλείου σε σχέση με το **tcpdump** είναι η ταχύτητα. Είναι φυσικό ένα **GUI** εργαλείο να υστερεί ενός **command line** εργαλείου στον παράγοντα που λέγεται ταχύτητα, ειδικά όταν ο όγκος των data που πρέπει να επεξεργαστούν είναι πολύ μεγάλος. Αυτός είναι ένας από τους λόγους για τον οποίο δεν μπορούμε να ορίσουμε ότι το ένα εργαλείο είναι καλύτερο από το άλλο. Η χρήση και των δύο είναι σημαντική στην επίτευξη μιας ανάλυσης.

ASCII Decoded network Session Files

Η πληροφορία μου μεταφέρεται στα πακέτα που διακινούνται στο δίκτυο honeynet, μπορεί να απομονωθεί από τα υπόλοιπα στοιχεία (protocol headers) που προσδιορίζουν τις κατευθύνσεις, τα μεγέθη και άλλες χρήσιμες πληροφορίες των πακέτων, και να αποθηκευτεί μόνο το εκτυπώσιμο με χαρακτήρες **ASCII** μέρος του φορτίου(payload) μιας σύνδεσης που επιτυγχάνει μια IP με μια άλλη IP.
Με παραμετροποίηση του **snort**, μπορούμε να επιτύχουμε μια δομή καταλόγων αρκετά βολική στην αναζήτηση των συνόδων (sessions) που επιτυγχάνει μια IP προς και από το **honeynet**.

Μια δομή καταλόγων που χρησιμοποιούμε με βάση την ημερομηνία είναι η παρακάτω:

<προκαθορισμένο path>/Year/Month_Year/Day_month/snort/ip_address/<Sessions>:Sprt-Dprt

Με αυτή την μορφή τα sessions καταγράφονται σε μια διαδρομή καταλόγων που αποτελείται, από το έτος , τον μήνα, την ημερομηνία (πχ ../2004/Jan_2004/Jan_19/), snort και την IP διεύθυνση που ξεκινάει μία σύνδεση. Τα sessions είναι αρχεία **ASCII** που περιέχουν το payload των συνδέσεων και έχουν την μορφή: SESSION:πόρτα_πηγής - πόρτα_προορισμού. Για παράδειγμα το SESSION:12345-80 περιέχει το **payload** μιας σύνδεσης που ξεκίνησε η μηχανή με IP ίδια με το όνομα του καταλόγου στον όποιο βρίσκεται το **ASCII** αρχείο και ξεκινάει μια σύνδεση από την πόρτα 12345 προς την πόρτα 80 κάποιου άλλου μηχανήματος.

Πίνακας 2-11 - Παράδειγμα σε μορφή Δένδρου

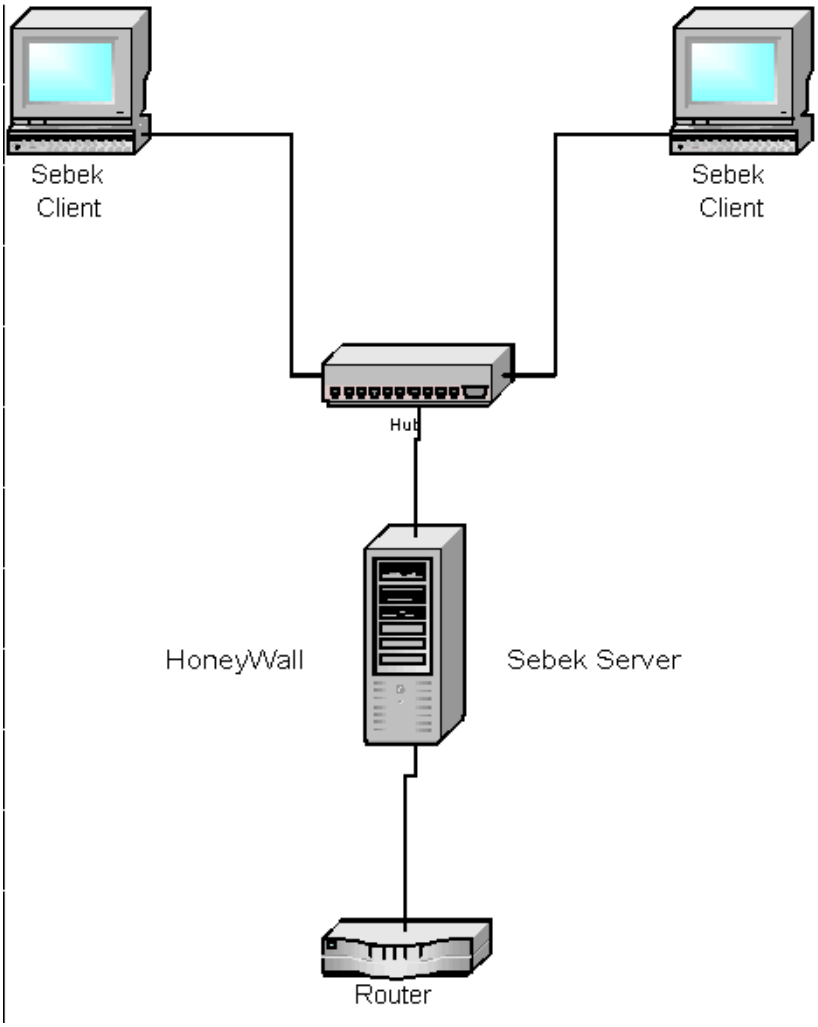
```
2004/
`-- Feb_2004
   |-- Feb_06
   |   `-- snort
   |       |-- 192.168.0.135
   |       |   |-- SESSION:33232-21
   |       |   |-- SESSION:33234-22
   |       |   |-- SESSION:33239-21
```

Τρίτο επίπεδο - Honeypots

Keystroke logs

Μια σημαντική πηγή δεδομένων του **honeynet** είναι τα **keystrokes (Πληκτρολογήσεις)**. Αυτά δηλαδή που πληκτρολογεί όποιος αλληλεπιδρά με το **honeynet**. Η συλλογή των **keystrokes** δεν είναι εύκολη και γίνεται πραγματικά περίπλοκη γιατί απαιτείται να μην γίνει αντιληπτή η λειτουργία αυτή από τον επιτιθέμενο.

Ένα από τα καταλληλότερα εργαλεία για τέτοια χρήση είναι το **sebek**. Η φιλοσοφία λειτουργίας του, όπως φαίνεται και στο σχήμα, είναι η εξής:



Εικόνα 2-3 sebek

Στα **honeypots** υπάρχει εγκαταστημένο το **sebek** client και στο **honeywall** τρέχει το **sebek** server. Η δουλειά του **sebek** client είναι να συλλέγει τις **πληκτρολογίσεις** που γίνονται στο μηχάνημα που είναι εγκατεστημένο και να τις στέλνει σαν ένα φυσιολογικό πακέτο, connectionless πρωτοκόλλου UDP, και ταυτόχρονα να κρύβει την ύπαρξη του ως διεργασία που τρέχει. Για να σταλούν τα **keystrokes** το **sebek client** δημιουργεί ένα ψεύτικο UDP πακέτο προς κάποια **mac address** (όχι IP address, η IP που φαίνεται μπορεί να είναι μια οποιαδήποτε IP που θα επιλέξουμε εμείς, η οποία επιλέγεται ώστε να φαίνεται φυσιολογικό το πακέτο).
Το **sebek** server ελέγχει την κίνηση των πακέτων μέσα στο **honeynet** και συλλέγει τα πακέτα που έχει δημιουργήσει το **sebek** client και κινούνται προς τον **router**. Αυτά τα πακέτα δίνουν τα ακόλουθα logs (πίνακας 2-12)

```
1. [2004-02-18 10:48:30  192.168.0.5 2699 mingetty 0]root
2. [2004-02-18 10:49:30  192.168.0.5 2700 bash 0]vi /etc/syslog.com[BS]nf
3. [2004-02-18 10:49:45  192.168.0.5 2754 vim 0]i
4.[2004-02-18 10:49:45   192.168.0.5 2754 vim 0]
5. [2004-02-18 10:50:08  192.168.0.5 2754 vim 0]*.*@192.168.0.130
6. [2004-02-18 10:50:10  192.168.0.5 2754 vim 0][ESC]:wq
7. [2004-02-18 10:50:18  192.168.0.5 2700 bash 0]service syslog restart
8. [2004-02-18 10:50:25  192.168.0.5 2700 bash 0]logout
```

Πίνακας 2-12

Σε κάθε **keystroke** βλέπουμε

[<ημερομηνία ώρα> <IP του μηχανήματος > <ο αριθμός του procces που παρήγαγε το keystroke> <το όνομα αυτού του prcess> <ο αριθμός του χρήστη uid >] <τα keystrokes>

Στο παραπάνω παράδειγμα (πίνακας 2-12) γίνονται τα ακόλουθα:

Γραμμή 1, mingetty του linux , προφανώς στο login, πληκτρολογείται η λέξη root

Γραμμή 2, είμαστε πλέον στο bash, που σημαίνει ότι το login πέτυχε χωρίς να μπορούμε να δούμε το password, και πληκτρολογείται vi /etc/syslog.com[BS]nf . Εδώ παρατηρούμε το [BS] (backspace) , δηλαδή γράφοντας την εντολή πατήθηκε μια φορά το backspace, έσβησε τον προηγούμενο χαρακτήρα και συνέχισε, το αποτέλεσμα είναι τελικά vi /etc/syslog.conf.

Στη γραμμή 3, έχει μπει στον **vi editor** και πληκτρολογείται i, για insert , ένα enter στην γραμμή 4, ενώ στην γραμμή 5 προστίθεται την γραμμή *.*@192.168.0.130 .

Γραμμή 6 , σώζει και βγαίνει από τον vi.

Γραμμή 7, ξανά ξεκινάει το service syslog restart για να ενεργοποιηθούν οι νέες ρυθμίσεις και βγαίνει με logout, γραμμή 8.