Μελέτη Κακόβουλων Λογισμικών

Έρευνα στα πλαίσια μελέτης του Κακόβουλου Λογισμικού KeeFarce

Επίκουρος Καθηγητής Κυριαζής Δημοσθένης

Βασίλειος Μαρκόπουλος (14026)
Π.Μ.Σ. ΨΗΦΙΑΚΑ ΣΥΣΤΗΜΑΤΑ & ΥΠΗΡΕΣΙΕΣ–
ΔΙΚΤΥΟΚΕΝΤΡΙΚΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

Ευχαριστίες

Θα ήθελα να ευχαριστήσω του καθηγητές μου, κ. Ξενάκη και κ. Κυριαζή, για την αμέριστη υποστήριξη και καθοδήγηση που μου παρείχαν. Επίσης, θα ήθελα να ευχαριστήσω ξεχωριστά τον κ. Νταντογιάν Χριστόφορο για την συνεχή καθοδήγηση και βοήθεια που μου παρείχε, καθώς χωρίς την συμβολή του δεν θα μπορούσε να επιτευχθεί το επιθυμητό αποτέλεσμα. Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου και τους καλούς μου φίλους, Λίλα και Χάρη, για την υποστήριξη και την ενθάρρυνση με τις οποίες δεν σταμάτησαν να με τροφοδοτούν.

ΠΕΡΙΛΗΨΗ

Στην παρούσα πτυχιακή εργασία έγινε προσπάθεια παρουσίασης μίας ενδελεχούς και ολοκληρωμένης ανάλυσης Κακόβουλων Λογισμικών, μέσω της μελέτης του λογισμικού *KeeFarce*. Αρχικά παρουσιάζεται ένα γενικότερο θεωρητικό πλαίσιο σύμφωνα με το οποίο θα εντάξουμε στην συνέχεια την δράση του λογισμικού *KeeFarce* και το οποίο θέτει την βάση για την κατανόηση του κάθε βήματος της ανάλυσης. Παράλληλα με την παρουσίαση της μελέτης του *KeeFarce* γίνονται φανερά τα βήματα μίας τεχνολογικά υπέροπλης ανάλυσης χωρίζοντας την μελέτη σε τρία στάδια: εντοπισμός, ανάλυση και εξαγωγή μέτρων προστασίας. Σε κάθε βήμα χρησιμοποιούνται διάφορες τεχνικές, εργαλεία και λειτουργικά συστήματα (*Windows, Ubuntu Linux).* Εργαλεία όπως το *Volatility,* το *Yara*, το *yarGen* και το *Cuckoo SandBox* παρουσιάστηκαν ως πολύτιμη αρωγή στην επιτέλεση ολοκληρωμένης ανάλυσης. Στην συνέχεια παρουσιάστηκε και ένα σενάριο σε γλώσσα *Python* που αυτοματοποιεί την λειτουργία ανίχνευσης του *Yara,* προσφέροντας μία ιδέα για την χρήση τροποποιήσιμων εργαλείων για την ανίχνευση κακόβουλων υπάρξεων. Το συνολικό αποτέλεσμα της μελέτης προσφέρει μία πλήρη εικόνα του κακόβουλου πλέον λογισμικού *KeeFarce* και της δράσης του καθώς και τρόπους αντιμετώπισής του, ενώ παράλληλα γονιμοποιεί το έδαφος για την επέκταση των παρουσιαζόμενων τεχνικών σε επόμενο επίπεδο και για την δημιουργία μίας εξελίξιμης διαδικασίας αυτοπροστασίας από κακόβουλες επιθέσεις λογισμικού. Συμπληρωματική έρευνα παραδειγματικού τύπου παρουσιάζεται στο ΄Β Παράρτημα.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

1. ΕΙΣΑΓΩΓΗ

Στην αυγή του 21ου αιώνα η ροή των εξελίξεων σε όλα τα επίπεδα υπήρξε χειμαρρώδης και καλπάζουσα. Η δύναμη της πληροφορίας ήταν πάντα το μήλο της έριδος για κάθε λάγνο της εξουσίας, ενώ και για κάθε άνθρωπο ξεχωριστά η απόκτηση πληροφοριών έχει τεράστια σημασία στην διευκόλυνση της καθημερινής του ζωής. Κατά την τελευταία δεκαπενταετία, η συνένωση του πρώτου με την ανάγκη για την απόκτηση του δεύτερου, γέννησε την ενσωμάτωση των περισσοτέρων δραστηριοτήτων του ανθρώπου με την επιστήμη της πληροφορικής. Ένα μεγάλο μέρος του πληθυσμού των ανθρώπων της Γης έχει υποστεί θεμελιώδεις αλλαγές στον τρόπο ζωής μέσα από την μετατροπή διαδικασιών που απαιτούν απλή ή σύνθετη προσπάθεια σε αυτοματοποιημένες διαδικασίες υπολογιστικών συστημάτων. Η συνένωση αυτή προκάλεσε ένα κύμα θετικών αλλά και αρνητικών επιπτώσεων.

Οι θετικές αλλαγές με τις οποίες συνοδεύτηκε η ολοένα και αυξανόμενη συνένωση της ζωής του ανθρώπου και των διεργασιών που εκτελούνται μέσω λογισμικών χωρίζονται σε δύο κλάδους. Είναι οι αλλαγές οι οποίες βελτίωσαν την καθημερινότητα κάθε ανθρώπου μεμονωμένα αλλά και ολόκληρων οργανωμένων ομάδων ατόμων, όπως είναι οι κρατικοί οργανισμοί. Οι αλλαγές αυτές αφορούν πολυεπίπεδες μεταμορφώσεις των τρόπων με των οποίων εκτελούνται διάφορες διεργασίες, όπως η διεκπεραίωση τραπεζικών συναλλαγών ή η εκτέλεση καταναλωτικών αγορών. Από την άλλη πλευρά, υπάρχουν και οι αλλαγές που δεν έφεραν βελτίωση αλλά προσέφεραν ικανότητες για την πραγματοποίηση πράξεων αδύνατων με τις υπάρχουσες τεχνικές. Για παράδειγμα, η αεροδυναμική μελέτη οχημάτων γνώρισε τεράστιες εξελίξεις, καθώς η χρήση υπερυπολογιστών για την εκτέλεση υπολογισμών μαθηματικών τύπων που αφορούν σε φυσικά μοντέλα και φαινόμενα προσέφερε την δυνατότητα για την τάχιστη εξαγωγή συμπερασμάτων από μελέτες σε αεροδυναμικές σήραγγες. Εν γένει, η χρήση της τεχνολογίας της πληροφορικής κατέστησε δυνατή την από απόσταση εκτέλεση πλήθους ενεργειών απλοποιώντας στο μέγιστο στην ζωή μας και εξοικονομώντας πολύτιμο χρόνο.

Στον αντίλογο, ήδη από τις αρχές της προηγούμενης δεκαετίας, εκφράστηκαν βαθιές ανησυχίες για την ασφάλεια της εκτέλεσης ευαίσθητων δραστηριοτήτων, όπως π.χ οι τραπεζικές συναλλαγές και η αποθήκευση προσωπικών στοιχείων, σε υπολογιστικές μηχανές. Δυστυχώς, όπως ήταν αναμενόμενο, οι ανησυχίες αυτές δεν άργησαν να επιβεβαιωθούν και στα τέλη της προηγούμενης δεκαετίας έχει ήδη σχηματιστεί ένα τόσο

μεγάλο και ανομοιογενές πλήθος κακόβουλων λογισμικών, η μελέτη του οποίου αποτελεί σήμερα ολόκληρο παρακλάδι της επιστήμης της πληροφορικής με όνομα *Malware Analysis.* Συγκεκριμένα, οι αντιμαχόμενες πλευρές των επιτιθέμενων και των αμυνόμενων έχουν δημιουργήσει μία βιομηχανία η οποία σημειώνει τεράστιους ρυθμούς ανάπτυξης. Συγκεκριμένα στην περίπτωση του κράτους του Ισραήλ, το 2014 ήταν η πρώτη χρόνιά στην οποία τα έσοδα από της επιχειρήσεις κυβερνοάμυνας και άμυνας έναντι των κακόβουλων λογισμικών ξεπέρασαν τα έσοδα από την βαριά βιομηχανία όπλων που διαθέτει, σημειώνοντας μία εξέλιξη που δεν θα αργήσει να εξαπλωθεί και στον υπόλοιπο κόσμο. Το παραπάνω αποτελεί ένα δείγμα την δραστηριότητας που εξελίσσεται μεταξύ των απανταχού *Hackers* και των εταιριών λογισμικών προστασίας.

Ωστόσο, η χρήση αυτοματοποιημένων λογισμικών προστασίας, που παρήχθησαν ως αποτέλεσμα της παραπάνω αντιμαχίας και η οποία δεν έχει σαφή όρια αλλά αποτελεί μία αέναη μάχη κέρδους στην αρένα της εμπορευματοποίησης, δεν ενδείκνυται σε όλες τις περιπτώσεις. Ιδιαίτερα στη περίπτωση στην οποία ο χρήστης έχει συνάφεια με την πληροφορική και την ασφάλεια λογισμικού και θέλει να δημιουργήσει ένα ασφαλές περιβάλλον διενέργειας ευαίσθητων εργασιών. Τότε, θα πρέπει ο ίδιος ο χρήστης να διασφαλίσει την ασφάλεια του υπολογιστικού του συστήματος μέσα από διαρκή παρακολούθηση και διενέργεια ανάλυσης σε δείγματα λογισμικών που έχουν ύποπτη συμπεριφορά.

Στα πλαίσια της αυτοπροστασίας του υπολογιστικού μας συστήματος, εκπονήθηκε η εργασία αυτή με κύριο σκοπό μέσα από την μελέτη του κακόβουλου λογισμικού *KeeFarce* να παρουσιαστεί η βασική δομή διεξαγωγής μία άρτιας, πολυεπίπεδης και ολοκληρωμένης ανάλυσης κακόβουλων λογισμικών. Το λογισμικό που εξετάστηκε, την χρονική περίοδο της εξέτασης, δεν έχει μελετηθεί και δεν υπήρχαν καταγεγραμμένες πληροφορίες για την ακριβή συμπεριφορά και λειτουργία του σε ένα υπολογιστικό σύστημα με λειτουργικό *Windows.* Η εξέταση του έγινε σε αντιπαραβολή με τον λογισμικό το οποίο στοχεύει και ονομάζεται *KeePass.* Η υποκλοπή στοιχείων στην περίπτωση αυτή αποτελεί ένα άριστο δείγμα της παρείσφρησης κακόβουλων οντοτήτων στα προσωπικά δεδομένα του χρήστη, ακόμα και αν αυτά προστατεύονται από κωδικούς ασφαλείας. Για την διεξαγωγή της μελέτης χρησιμοποιήθηκαν σύγχρονες μεθοδολογίες και τεχνολογίες για την παρουσίαση μίας διαδικασίας πρότυπο για την μελέτη κακόβουλων λογισμικών.

2.ΚΑΚΟΒΟΥΛΟ ΛΟΓΙΣΜΙΚΟ

Ένα λογισμικό χαρακτηρίζεται κακόβουλο (Malware) όταν η λειτουργικότητα του αφορά στην επιτέλεση ορισμένων ανεπιθύμητων για τον χρήστη ενεργειών, χωρίς την δική του συναίνεση. [1] Πιο συγκεκριμένα το κακόβουλο λογισμικό ευθύνεται για:

1. Διακοπή ροής ορισμένων λειτουργιών του Υπολογιστή
2. Συλλογή ευαίσθητων πληροφοριών
3. Απόκτηση πρόσβασης σε Ιδιωτικά Συστήματα Υπολογιστών
4. Εμφάνισή ανεπιθύμητων διαφημίσεων

Ένα λογισμικό το οποίο προκαλεί ανεπιθύμητα αποτελέσματα στον υπολογιστή, δεν αποτελεί αυθωρεί κακόβουλο λογισμικό. Αντίθετα λόγω ελλιπούς σχεδίασης και μη χρήσης ορθών τεχνικών προγραμματισμού, είναι δυνατόν ένα πρόγραμμα να προκαλέσει προβλήματα στην ροή λειτουργίας ενός υπολογιστή. Το εν λόγω λογισμικό ονομάζεται *Badware* και η ανάπτυξή και χρήση του δεν αποτελεί κακόβουλη ενέργεια.

Το κακόβουλο λογισμικό δύναται να πάρει πολλές διαφορετικές μορφές, οι οποίες κατηγοριοποιούνται σύμφωνα με μία σωρεία παραγόντων. Ενδεικτικά, μπορούμε να πούμε ότι κατηγοριοποιούνται σύμφωνα με τον τρόπο λειτουργίας και μόλυνσης του συστήματος καθώς και με την ικανότητα να μην γίνονται αντιληπτά και να δρουν στο παρασκήνιο. Επίσης, συμφωνά με τα παραπάνω και σε συνάρτηση με την μορφή με την οποία προγραμματίζονται, ένα κακόβουλο λογισμικό μπορεί να είναι *ιός υπολογιστή, worm, trojan horse, ransomware, spyware, scareware.*

Το κακόβουλο λογισμικό συνήθως αποκρύπτει την παρουσία του και εγκαθίσταται στο σύστημα στόχο είτε ως “μεταμφιεσμένο” σε κάποιο νόμιμο και αβλαβές λογισμικό, είτε με ενσωμάτωση σε ένα νόμιμο και υγιές λογισμικό. Η ενσωμάτωση σε ένα αβλαβές λογισμικό είναι πολλές φορές πρακτική γνωστών εταιριών. Οι εταιρίες αυτές ενσωματώνουν λογισμικό παρακολούθησης (spyware) σε επίσημα προϊόντα τους με σκοπό να επιτελέσουν κάποιες λειτουργίες ανίχνευσης των συνηθειών των χρηστών της εφαρμογής για σκοπούς μάρκετινγκ.

Η προστασία ενός συστήματος από ένα κακόβουλο λογισμικό επιτυγχάνεται αφενός με την συνετή χρήση του διαδικτύου, δηλαδή την αποφυγή διεπαφής με ύποπτες ιστοσελίδες, παραπλανητικούς συνδέσμους και λογισμικού κατεβασμένου από μη έγκυρες πηγές, αφετέρου με την χρήση προγραμμάτων προστασίας από κακόβουλες οντότητες, όπως είναι τα anti-virus, τα anti-malware και τα τείχη προστασίας (firewalls). Στον τομέα της προστασίας έχουν δαπανηθεί τα τελευταία χρόνια εξαιρετικά υψηλά ποσά για την ανάπτυξη αλεξίσφαιρων συστημάτων, μέσα από την εξαντλητική διενέργεια αναλύσεων επί των πλέον σύγχρονων κακόβουλων λογισμικών που αναπτύσσονται με ραγδαίους ρυθμούς.

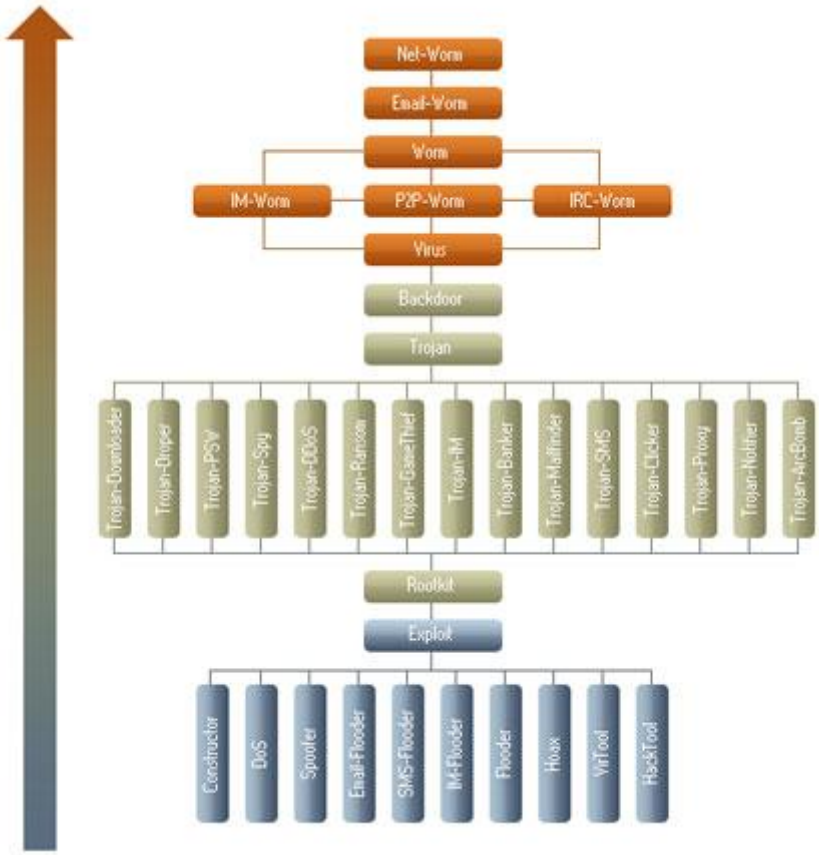
2.1 Κατηγορίες Κακόβουλων Λογισμικών

Η κατηγοριοποίηση των κακόβουλων λογισμικών αποτελεί πολύπλοκη και πολυεπίπεδη διαδικασία, ο χαρακτήρας της οποίας διαφεύγει της απλής ονοματοδοσίας σε ομάδες κοινών χαρακτηριστικών, αλλά αφορά στην ομαδοποίηση που είναι απαραίτητη για την ταυτοποίηση και την προστασία από αυτά. Για την εκάστοτε ομαδοποίηση, ορίζονται κάποια κριτήρια. Τα κριτήρια αυτά δεν είναι δυνατόν να προσδιορίσουν μονοσήμαντα τα κακόβουλα λογισμικά και συχνά συνδυάζονται με άλλων ομάδων κριτήρια για να δημιουργήσουν καινούργιες ομάδες.

Το μεγαλύτερο τροχοπέδη στην προσπάθεια ομαδοποίησης και κατηγοριοποίησης του τεραστίου αριθμού των κακόβουλων λογισμικών που υπάρχουν είναι τόσο η πληθωρικότητα των τεχνολογιών που χρησιμοποιούν, οι οποίες τα ομαδοποιούν όσον αφορά στην λειτουργικότητα, όσο του ίδιου του αριθμού που εμφανίζονται και εξελίσσονται. Είναι λοιπόν προφανές ότι μπορούμε να κάνουμε λόγο μόνο για κατηγοριοποίηση που αφορά συγκεκριμένα χαρακτηριστικά των λογισμικών καθώς και αντιστοίχησης ενός κακόβουλου λογισμικού σε περισσότερες από μία κατηγορίες. Στην συνέχεια θα αναφερθούμε στις πιο βασικές κατηγορίες, οι οποίες είναι και αυτές που χρησιμοποιούμε συχνότερα στην φάση της ανάλυσης των κακόβουλων λογισμικών.

2.1.1 Κατηγοριοποίηση βάσει Δραστηριότητας σε Υπολογιστή Ξενιστή

Σε δημοσίευση των εργαστηρίων ανάλυσης *Kaspersky Lab* [2] αναφέρεται ένα είδος κατηγοριοποίησης σύμφωνα με το είδος της δραστηριότητας που επιτελεί το κακόβουλο λογισμικό αφού μολύνει τον υπολογιστή ξενιστή. Στο εν λόγω άρθρο τονίζεται η σημασία του κριτηρίου κατηγοριοποίησης αυτής της κατηγορίας, καθώς είναι αυτό που χρησιμοποιούν οι περισσότεροι φορείς που ασχολούνται με την ανάλυση κακόβουλων λογισμικών για να επιτύχουν μία βασική κατηγοριοποίηση των υπαρχόντων λογισμικών. Η κατηγοριοποίηση αποτυπώνεται σε ένα *Δένδρο Κατηγοριοποίησης*, στο οποίο κάθε είδος κακόβουλου λογισμικού έχει σαφώς προσδιορισμένη θέση και ξεκάθαρη περιγραφή. Στο *Δένδρο Κατηγοριοποίησης* η φορά από κάτω προς τα επάνω υποδηλώνει την αύξηση της απειλής που αποτελεί το κάθε είδος (Εικόνα 1).



Εικόνα 1: Δένδρο Κατηγοριοποίησης Κακόβουλων Λογισμικών

Η κατηγοριοποίηση αυτή είναι ξεκάθαρη για κακόβουλα λογισμικά μου χρησιμοποιούν συναρτήσεις με αντιστοιχία σε ένα από τα είδη που απεικονίζονται. Ωστόσο, στην πράξη αυτό συμβαίνει σπάνια, αφού τα περισσότερα σύγχρονα κακόβουλα λογισμικά

χρησιμοποιούν συναρτήσεις και τεχνικές διάδοσης που δεν εμπίπτουν σε μία μόνο κατηγορία απειλής. Για παράδειγμα, ένα λογισμικό μπορεί να είναι ικανό να διαδίδεται τόσο μέσω ηλεκτρονικής αλληλογραφίας όσο και μέσω P2P δικτύων. Για το λόγω αυτό το *Kaspersky Lab* διατυπώνει μία ομάδα από κανόνες, οι οποίοι μπορούν να οδηγήσουν στην ξεκάθαρη κατηγοριοποίηση ενός κακόβουλου λογισμικού σύμφωνα με την συμπεριφορά και ανεξαρτήτως των μηχανισμών που χρησιμοποιεί. Οι κανόνες αυτοί είναι οι εξής:

- Κάθε συμπεριφορά κακόβουλου λογισμικού που εμφανίζεται στο *Δένδρο Κατηγοριοποίησης* αντιστοιχεί σε ένα μόνο επίπεδο κινδύνου.
- Οι συμπεριφορές που αποτελούν μεγαλύτερο κίνδυνο έχουν μεγαλύτερη βαρύτητα και υπερνικούν τις συμπεριφορές με μικρότερο κίνδυνο.
- Στην περίπτωση που ένα λογισμικό χρησιμοποιεί τεχνικές που οδηγούν σε συμπεριφορές παρόμοιου επιπέδου κινδύνου, το λογισμικό χαρακτηρίζεται με την γενικότερη μορφή της κατηγορίας στην οποία ανήκει. Δηλαδή, ένα κακόβουλο λογισμικό με παρομοίου κινδύνου συμπεριφορές όπως Trojan-Ransom, Trojan-ArcBomb, Trojan-Clicker, Trojan-DDoS, Trojan-Downloader, Trojan-Dropper, Trojan-IM, Trojan-Notifier, Trojan-Proxy, Trojan-SMS, Trojan-Spy, Trojan-Mailfinder, Trojan-GameThief, Trojan-PSW, or Trojan-Banker, χαρακτηρίζεται ως Trojan.

Άρα στο παράδειγμα μας το κακόβουλο λογισμικό θα προσδιοριστεί ως Email-Worm, καθώς αυτή του η συμπεριφορά έχει μεγαλύτερο βάρος κινδύνου από την συμπεριφορά P2P-Worm. Συνεπώς, προκύπτει μία νέα υπομορφή κατηγοριοποίησης εντός της συμπεριφορικής εξέτασης του λογισμικού, και αυτή δεν είναι άλλη από την διαστρωμάτωση σύμφωνα με το κίνδυνο που απειλεί ένα σύστημα.

2.1.2 Ειδική κατηγοριοποίηση με βάσει συνδυασμό κριτηρίων

Η κατηγοριοποίηση των κακόβουλων λογισμικών δεν σταματά σε προκαθορισμένα όρια και ομάδες διαβαθμισμένης απειλής. Η αναγνώριση ενός τέτοιου λογισμικού ως μέλος ομάδας με συγκεκριμένα χαρακτηριστικά, είναι μία διαδικασία που προσαρμόζεται και εμπλουτίζεται με ίδιο βήμα με την εξέλιξη που παρουσιάζουν τα κακόβουλα λογισμικά. Φυσική προέκταση αυτού, είναι και η εμφάνιση νέων ομάδων κακόβουλων λογισμικών, η ομαδοποίηση των οποίων προέκυψε από την αδυναμία του *Δένδρο Κατηγοριοποίησης* να περιγράψει επακριβώς συνδυασμούς σύνθετων συμπεριφορών που αυτά εμφανίζουν. Συνεπώς,

σύγχρονη τάση αποτελεί η συγγραφή κανόνων που βασίζονται σε ενδείκτες εισβολής (Indicators of Compromise – θα γίνει αναλυτική μνεία σε επόμενη ενότητα) και χρησιμοποιούνται σε εργαλεία ανοιχτού κώδικα (όπως το Yara) με σκοπό να μπορεί κάθε απειλούμενος να προσαρμόζει την προσπάθεια αναγνώρισης κακόβουλων στις παραμέτρους του συστήματος που χρησιμοποιεί, δημιουργώντας ομάδες λογισμικών με συγκεκριμένα καθορισμένες ύποπτες συμπεριφορές.

Σε αυτού του είδους την κατηγοριοποίηση δεν γίνεται να αναφερθούν συγκεκριμένες ομάδες συμπεριφορών και αναγνωριστικών διότι αυτές δεν έχουν κάποια συγκεκριμένη ονοματολογία. Ωστόσο, αξίζει να αναφερθούν οι βασικοί τομείς πάνω στους οποίους βασίζονται οι κανόνες προσδιορισμού της απειλής [3]:

- Λεκτικά σχήματα σε κωδικοποίηση ANSI ή Unicode που βρίσκονται κωδικοποιημένα σε ένα PE (Portable Executable) αρχείο.
- Πληροφορίες που βρίσκονται μέσα σε ένα PE, όπως οι βιβλιοθήκες DLL που χρησιμοποιεί το εκτελέσιμο και οι κλήσεις συναρτήσεων από την DLL.
- Σχέδια που προκύπτουν από την αλληλουχία των Byte μέσα το εξεταζόμενο εκτελέσιμο αρχείο.

Οι κανόνες αυτοί αφορούν στα λειτουργικά συστήματα Windows. Όσον αφορά στους γενικούς κανόνες προσδιορισμού απειλής ανεξάρτητου πλατφόρμας αναφέρουμε τους εξής βασικούς τομείς:

- Μοτίβα συμπεριφοράς και αλληλεπίδρασης με δίκτυο.
- Λειτουργικό σύστημα και ιδιαιτερότητες αυτού.
- Σκοπός προσβολής.
- Κακόβουλο αποτέλεσμα.

Περισσότερα στοιχεία όσον αφορά την δημιουργία τέτοιων κανόνων θα παρουσιαστούν σε επόμενη ενότητα.

2.1.3 Τα πιο διαδεδομένα κακόβουλα λογισμικά

Στις μέρες μας, η ενσωμάτωση λειτουργιών της καθημερινής ζωής του ανθρώπου στην αναπτυσσόμενη ράγα της τεχνολογίας έχει ως αποτέλεσμα την ολοένα και περισσότερη ενσωμάτωση προσωπικών στοιχείων των χρηστών σε διαδικτυακές υλοποιήσεις. Για

παράδειγμα, η χρήση λογαριασμών e-Banking για διεκπεραίωση τραπεζικών συναλλαγών επιφέρει την αποθήκευση εξαιρετικά ευαίσθητων πληροφοριών σε ηλεκτρονικές πλατφόρμες. Συνέπεια, αυτών αποτελεί η δημιουργία ανάλογων κακόβουλων λογισμικών για την απόκτηση αυτών των πληροφοριών, ενώ η επιθέσεις που πραγματοποιήθηκαν τα έτη 2014 και 2015 άγγιξαν πρωτόγνωρα ύψη σε επίπεδο αριθμών. Η δράση των σύγχρονων *Hackers* συνοψίζεται στις παρακάτω πιο διαδεδομένες κατηγορίες κακόβουλων λογισμικών [4]:

- ***Ransomware.***

  Η κατηγορία αυτή εμφανίστηκε πρώτη φορά το 1989 με το κακόβουλο λογισμικό *PC Cyborg.* Τα κακόβουλα λογισμικά που εμπίπτουν σε αυτή την κατηγορία έχουν ως στόχο της κρυπτογράφηση όλων των αρχείων στον σκληρό δίσκο του υπολογιστή του χρήστη με αποτέλεσμα αυτός να μην μπορεί να τα χρησιμοποιήσει. Κατόπιν τούτου ο κακόβουλος *Hacker* ζητάει χρήματα από τον χρήστη για την αποκρυπτογράφηση των αρχείων. Η χρήση αυτού του είδους των κακόβουλων λογισμικών έχει εκτοξευθεί, αφού μόνο το λογισμικό *Cryptowall* έχει κοστίσει στην παγκόσμια οικονομία 18 εκατομμύρια δολάρια σε λιγότερο από ένα χρόνο από την εμφάνισή του. Μερικά άλλα κακόβουλα λογισμικά αυτής της κατηγορίας είναι τα *CryptoLocker,* Troldesh*, Bit Cryptor,* Tox Ransomware*, Alpha Crypt, Los Pollos Hermanos,* Locke*r.*

- ***Exploit Kits***

  Αυτή η κατηγορία δημιουργήθηκε με την πρωτοεμφάνιση του λογισμικού *Blackhole EK* το 2012 και ο αριθμός των μολύνσεων που έχουν προκληθεί την καθιστά σοβαρό κίνδυνο του διαδικτύου. Τα *Exlopit Kits* ακολουθούν διάφορες τεχνικές προσβολής και δράσης, η ανάλυση των οποίων έχει παράγει το ακόλουθο μοτίβο:

  1. Ο χρήστης επισκέπτεται μία εκτεθειμένη στους *Hackers* ιστοσελίδα.
  2. Ο χρήστης αντιμετωπίζει μία σειρά από προωθήσεις σε ιστοσελίδες μέχρι να καταλήξει σε αυτήν που στεγάζεται το κακόβουλο λογισμικό.
  3. Το λογισμικό συλλέγει πληροφορίες για το σύστημα του θύματος και αποφασίζει τον τρόπο εγκατάστασης σε αυτό.
  4. Με την επιτυχή ολοκλήρωση του βήματος 3 το κακόβουλο λογισμικό εγκαθίσταται στον υπολογιστή του θύματος.

  Κακόβουλα λογισμικά τέτοιου τύπου είναι τα *Nuclear Exlpoit Kit, Rig Exploit Kit, HanJuan Exploit Kit, Angler Exploit Kit.*

- ***Τραπεζικά Trojans***

  Βασικός σκοπός αυτόν των κακόβουλων λογισμικών είναι η υποκλοπή στοιχείων τραπεζικών λογαριασμών για την απομύζηση χρηματικών ποσών. Ο τρόπος δράσης τους ποικίλει λόγω εναρμόνισης με σύγχρονες μεθόδους κυβερνοπροστασίας. Για παράδειγμα το λογισμό *Dridex* εκμεταλεύεται τα *macros του Microsoft Office Package,* ενώ το *Vawtrak* εγκαθίσταται είτε μέσω κρυφού κατεβάσματος στο υπολογιστή κατόπιν επιλογής κάποιου κακόβουλου συνδέσμου ηλεκτρονικής αλληλογραφίας, είτε με κάποιον *Malware Downloader,* είτε μέσω ενός *Exploit Kit.*

- ***PoS (Point-Of-Sale) Κακόβουλο Λογισμικό – Ram Scrappers***

  Στόχος αυτής της κατηγορίας κακόβουλων λογισμικών μπορεί να αποτελέσει οποιαδήποτε επιχείρηση χρησιμοποιεί συσκευή συναλλαγών πλαστικού χρήματος για την διεκπεραίωση χρηματικών συναλλαγών. Σκοπός των λογισμικών είναι η υποκλοπή στοιχείων από τις χρησιμοποιούμενες κάρτες. Καθώς οι συναλλαγές αυτές πραγματοποιούνται σε κανάλια επικοινωνίας πλήρως κρυπτασφαλισμένα, ο μόνος τρόπος πρόσβασης σε όλη την ροή μιας ασφαλούς συναλλαγής είναι η τερματική συσκευή. Σε αυτήν εγκαθίσταται ένα κακόβουλο λογισμικό το οποίο μπορεί και αποκτά πρόσβαση στην *RAM* της τερματικής συσκευής. Από εκεί κανείς μπορεί να διαβάσει και να υποκλέψει όλα τα στοιχεία της συναλλαγής, τα οποία δεν έχουν ακόμα κρυπτογραφηθεί. Κατόπιν απόκτησης των επιθυμητών στοιχείων αυτά αποθηκεύονται σε κατάλληλο φάκελο ή φακέλους με ονόματα τα οποία δεν εγείρουν υποψίες ούτε στον χρήστη ούτε στο ίδιο το σύστημα. Τα στοιχεία αυτά εκτός της φυσικής πρόσβασης, μπορούν να αποκτηθούν από τον επιτιθέμενο και μέσω διαδικτύου. Το κακόβουλο λογισμικό ανοίγει ένα δίαυλο επικοινωνίας, πολύ καλά καμουφλαρισμένο, με έναν απομακρυσμένο εξυπηρετητή, στον οποίο βρίσκεται η κακόβουλη οντότητα, μέσω μίας διαδρομή δρομολόγησης που να μην εγείρει υποψίες. Κατόπιν, το κακόβουλο λογισμικό αποστέλλει αυτά τα δεδομένα, κατακερματισμένα σε μικρά πακέτα και σε μεγάλο χρονικό διάστημα ώστε να μην είναι εύκολα ανιχνεύσιμη η όλη διαδικτυακή δραστηριότητα. Παραδείγματα τέτοιων λογισμικών είναι τα *NewPosThings, Punkey, BlackPos.* [5]

- ***Fake Tech Υπηρεσίες Υποστήριξης***

  Η κατηγορία αυτή περιλαμβάνει τεχνικές εξαπάτησης του χρήστη καθώς και λογισμικό το οποίο δρα πάνω σε γνωστούς φυλλομετρητές. Όταν το υπολογιστή του χρήστη μολυνθεί από το κακόβουλο λογισμικό, αυτό προκαλεί την διαρκή εμφάνιση αναδυόμενων παραθύρων στον φυλλομετρητή του χρήστη που του αναφέρουν ότι το

σύστημα του είναι ευάλωτο σε πολυάριθμος απειλές. Του υποδεικνύουν, επίσης, να πραγματοποιήσει κάποια κλήση σε έναν αριθμό, όπως 1-855-791-2391, ή να πατήσει κάποιον σύνδεσμο για μετάβαση στην ιστοσελίδα της τεχνικής υποστήριξης. Αν ο χρήστης εξαπατηθεί και κάνει κάποιες από αυτές τις ενέργειές, είναι πιθανόν να εγκατασταθεί κάποια αφανής σύνδεση με κάποια απομακρυσμένη οντότητα και να υποκλαπούν ευαίσθητα προσωπικά στοιχεία που βρίσκονται στον υπολογιστή.

- ***Κακόβουλα Λογισμικά Αντιικής Προστασίας***
  Αυτή η κατηγορία περιέχει λογισμικό το οποίο ενώ προωθείται από κακόβουλες πήγες ως λογισμικό προστασίας από κακόβουλα λογισμικά, αυτό επιφέρει τα ακριβώς αντίθετα αποτελέσματα. Κατόπιν εγκατάστασης, προκαλεί επιβράδυνση του υπολογιστή, ενώ παράλληλα τον καθιστά περισσότερο ευάλωτο στις επιθέσεις των κακόβουλων λογισμικών. Ωστόσο, προσομοιώνει την λειτουργία ενός υγιούς λογισμικού προστασίας προκαλώντας την εμφάνιση παραπλανητικών ειδοποιήσεων για υποτιθέμενες απειλές. Τέτοιο λογισμικό είναι το *Antivirus Pro 2017.*

- ***Ανεπιθύμητα Προγράμματα (PUP) και Adware***
  Αυτή η κατηγορία συνίσταται από λογισμικά τα οποία εγκαθίσταται ταυτόχρονα με την εγκατάσταση των επιθυμητών από τον χρήστη προγραμμάτων, ενώ δεν εμπίπτουν στην σφαίρα του παρανόμου. Ωστόσο, λειτουργούν διαφημιστικά και επηρεάζουν την καταναλωτική ψυχολογία του χρήστη.

2.2 Μέθοδοι Προσβολής Προγραμμάτων Συστήματος

Τα κακόβουλα λογισμικά αποτελούν αποτέλεσμα των πλέον αναπτυγμένων τεχνικών προγραμματισμού. Είναι προϊόν επίπονης παρατήρησης και βαθιάς γνώσης του τρόπου λειτουργίας των σύγχρονων λειτουργικών συστημάτων και προγραμμάτων. Οι προγραμματιστές που ασχολούνται με τα κακόβουλα λογισμικά, είτε βρίσκονται με το μέρος των επιτιθέμενων για κακόβουλους σκοπούς, είτε στο βρίσκονται με το μέρος των αμυνομένων και των εταιριών προστασίας λογισμικού, έχουν εξαιρετικές ικανότητες και γνώσεις που επιτρέπουν, στους μεν να βρίσκουν τρύπες ασφαλείας σε εμπορικά προγράμματα, το κόστος των οποίων πολλές φορές αγγίζει τα αρκετά εκατομμύρια, και στους δε να διενεργούν αντίστροφη μηχανική (Reverse Engineering), μία διαδικασία συχνά πολλή δυσκολότερη της ανάπτυξης, για την εκμαίευση πληροφοριών χρήσιμων στην

καταπολέμηση των κακόβουλων λογισμικών.

Οι προγραμματιστικές μέθοδοι που χρησιμοποιούνται για την ανάπτυξη κακόβουλων λογισμικών είναι προσανατολισμένες σε εκμετάλλευση δυνατοτήτων των λειτουργικών συστημάτων για την απόκτηση πρόσβασης στην λειτουργική ροή ή τα δεδομένα καλόβουλων λογισμικών. Στην ενότητα αυτή θα παρουσιαστούν οι πιο διαδεδομένες τεχνικές καθώς και κάποια ενδεικτικά κακόβουλα λογισμικά που τις χρησιμοποιούν. Οι τεχνικές αυτές αποτελούν ουσιαστικά και τρόπους επίθεσης, ενώ είναι κατηγοριοποιημένες από το πρότζεκτ *OWASP. OWASP* (Open Web Application Security Project) είναι ένας παγκόσμιος μη κερδοσκοπικός οργανισμός με αποστολή την πριμοδότηση της ασφάλειας των εφαρμογών. Ο οργανισμός εκδίδει κάθε τρία χρόνια μία λίστα με τις δέκα ποιο διαδεδομένες επιθέσεις σε επίπεδο εφαρμογών. Από την λίστα αυτή, παρακάτω παρουσιάζονται εκείνες που συνδέονται άρρηκτα με την χρήση κακόβουλου λογισμικού. [6]

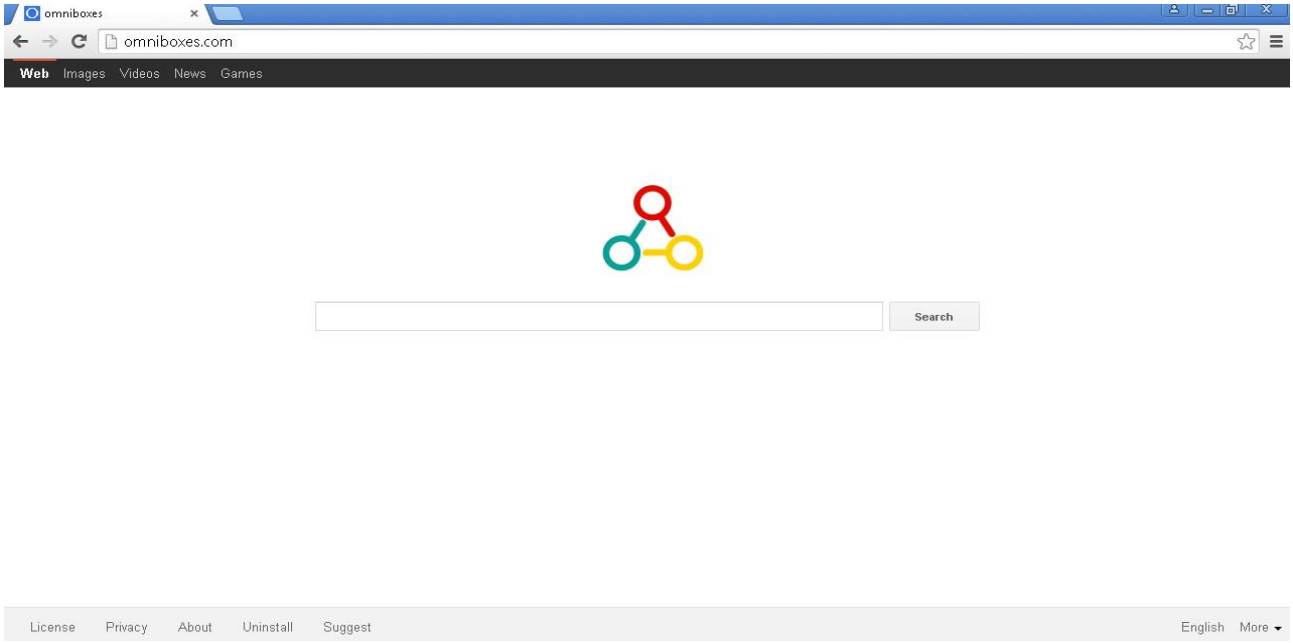
2.2.1 Μη επιβεβαιωμένες Επαναναπροωθήσεις

Αυτή είναι μία τεχνική με την οποία ο χρήστης μεταφέρεται άθελα του σε κακόβουλη ιστοσελίδα. Οι επιτιθέμενοι, ήτοι *Hackers,* χρησιμοποιούν τις διευθύνσεις *URLs* γνωστών και καλόβουλων ιστοσελίδων και διαστρεβλώνοντας τες καταφέρνουν να προωθήσουν τον χρήστη σε μία ανεπιθύμητη τοποθεσία.

Η εν λόγω τεχνική χρησιμοποιείται ακόρεστα στις επιθέσεις *Phising.* Ο τύπος επιθέσεων αυτός αφορά στην προσπάθεια των κακόβουλων δραστών να υποκλέψουν ευαίσθητες πληροφορίες από ένα χρήστη, όπως ονόματα χρήστη, κωδικούς και πληροφορίες των πιστωτικών καρτών, ενώ καμιά φορά και χρήματα απευθείας. Η υποκλοπή αυτή γίνεται με την μεταμφίεση της επιτιθέμενης οντότητας – κακόβουλο λογισμικό ως έμπιστη και διαπιστευμένη οντότητα σε μία ηλεκτρονική συναλλαγή που προσπαθεί να πραγματοποιήσει ένας χρήστης με μία οντότητα (π.χ αγορά ενός αντικειμένου από μία γνωστή ιστοσελίδα, όπως το e-Bay).

Συχνά ο χρήστης – θύμα λαμβάνει ένα μήνυμα το οποίο φαίνεται να έχει σταλεί από μία διαπιστευμένη οντότητα – οργανισμό (π.χ email από τα καταστήματα Zara για προσφορές προϊόντων, το οποίο περιέχει και ένα σύνδεσμο για υποτιθέμενη απευθείας μετάβαση στις προσφορές). Σε αυτό υπάρχει ένα συνημμένο αρχείο ή ένας σύνδεσμος, η χρήση των

οποίων οδηγεί είτε στην εγκατάσταση κακόβουλου λογισμικού, είτε στην προώθηση του χρήστη σε μία ιστοσελίδα, η οποία μοιάζει ίδια με την επίσημη. Στην περίπτωση εγκατάστασης ενός κακόβουλου λογισμικού, αυτό μπορεί να είναι είτε κάποιος Keylogger, ο οποίος θα υποκλέψει κωδικούς που πληκτρολογούμε, είτε κάποιο *Omnibox˙* μία προέκταση δηλαδή εγκατεστημένη στον φυλλομετρητή, η οποία μοιάζει με αθώα προέκταση αλλά ουσιαστικά την ώρα που ο χρήστης πληκτρολογεί μία διεύθυνση στην μπάρα διευθύνσεων, αυτή προτείνει κάποιες διευθύνσεις, οι οποίες αν πατηθούν θα προωθήσουν τον χρήστη σε ιστοσελίδες παρόμοιες των αυθεντικών με σκοπό να υποκλαπούν στοιχεία που θα υποβάλλει.



Εικόνα 2: Omnibox Malware

Στην Εικόνα 2 φαίνεται μία *Omnibox* προέκταση του φυλλομετρητή Google Chrome, η οποία παρουσιάζει την ίδια λειτουργικότητα με μία μηχανή αναζήτησης αλλά με ανεπιθύμητες προωθήσεις σε κακόβουλες ιστοσελίδες.

Η τεχνική του *Phising* [7] αποτελεί μία εύκολη τεχνική για του *Hackers,* στην οποία χρειάζεται να επιστρατεύσουν γνώσεις μάρκετινγκ, ώστε να καταφέρουν το θύμα να αποδεχτεί την πρόσκληση του δολώματος. Για παράδειγμα, μηνύματα που αφορούν κάποιες διάσημες εκδηλώσεις ή κάποια γεγονότα που έχουν αποκτήσει μεγάλη δημοσιότητα, είναι πιθανότερα να καταφέρουν τον χρήστη να πατήσει σε ένα σύνδεσμο με καταστροφικά για αυτόν αποτελέσματα. Στην περίπτωση που ο χρήστης πατήσει τον σύνδεσμο στο μήνυμά, τότε αυτόματα προωθείται σε μία ιστοσελίδα πανομοιότυπη με την αυθεντική στην οποία όταν κάνει κάποια υποβολή στοιχείων για *login*, τα στοιχεία αυτά θα υποκλαπούν.

2.2.2 Cross Site Request Forgery (CSRF) Pharming vulnerability

Η τεχνική *CSRF* είναι συνέχεια εκείνης που περιγράφηκε στην προηγούμενη ενότητα και έχει περιγραφεί ως ένα σημείο. Ωστόσο, μία υποκατηγορία αποτελεί η τεχνική Pharming, στην γίνεται ανακατεύθυνση της κίνησης μίας ιστοσελίδας σε μία άλλη πλαστή. Η επίθεση αυτή πραγματοποιείται είτε με την αλλαγή του *Host* αρχείου σε ένα υπολογιστή είτε με την εκμετάλλευση κάποιας αδυναμίας σε DNS Server λογισμικού.

Κατά την επίθεση *Cross Site Request Forgery (CSRF) Pharming vulnerabillity* χρησιμοποιείται ένα πακέτο διαδικτυακών εργαλείων στην μορφή κακόβουλου λογισμικού, σχεδιασμένα να αλλάζουν τις *domain name system* ρυθμίσεις των *routers* με σκοπό να ανακατευθύνουν την κίνηση σε κακόβουλες ιστοσελίδες, κίνηση που σκοπεύει στην υποκλοπή προσωπικών δεδομένων. [8], [9]

Παράλληλα της προηγούμενης περίπτωσης είναι το *MOOSE,* κακόβουλο λογισμικό που στοχεύει δρομολογητές και συσκευές βασισμένους σε λειτουργικό Linux, με σκοπό την δημιουργία απατών στα μέσα κοινωνικής δικτύωσης, πειρατεύωντας τις διαδικτυακές συνδέσεις των χρηστών. Αποτέλεσμα, αυτού είναι ότι το κακόβουλο λογισμικό κάνει “like” σε δημοσιεύσεις και φωτογραφίες, “follow” σε λογαριασμούς άλλων χρηστών και προβάλλει βίντεο χωρίς την συναίνεση του χρήστη. Ωστόσο, το *MOOSE* δεν εκμεταλλεύεται αδυναμία του λογισμικού των δρομολογητών αλλά κακορυθμισμένους δρομολογητές με αδύναμα και προεγκατεστημένα διαπιστευτήρια εισόδου.

2.2.3 Injection

Η κατηγορία αυτή είναι η πιο διαδεδομένη και η επιτυχία μια τέτοιας μεθόδου επιτρέπει στον επιτιθέμενο να μπορεί να κάνει ότι και το θύμα, έχοντας απεριόριστα δικαιώματα. Η τεχνική αυτή αφορά στην αλλαγή της ροής εκτέλεσης ενός προγράμματος εισάγοντας κώδικα μέσω κάποιας αδυναμίας του προγράμματος. Στην ενότητα αυτή θα αναφερθούμε σε δύο είδη *Injection*: το *Code Injection* και το *DLL Injection.* [10]

2.2.3.1 Code Injection

Η τεχνική της εισαγωγής κώδικα άφορα στην εκμετάλλευση μίας δυσλειτουργίας ενός προγράμματος που προκαλείται από την επεξεργασία μη έγκυρων δεδομένων. Επίσης η τεχνική έχει αποτελέσματα και σε περιπτώσεις στις οποίες κάποια λάθη στην μετάφραση προγραμμάτων οδηγούν σε απόδοση νοήματος εντολών συστήματος σε απλές εντολές χρήστη. Η απρόσεκτη χρήση των δεδομένων που εισάγει ο χρήστης και που αναμένει το πρόγραμμα να επεξεργαστεί και η χρήση μη σωστών προγραμματιστικών τεχνικών θωρακισμένων με θυρίδες ασφαλείας, δίνει την δυνατότητα σε *Hackers* να “εισβάλουν” στο εκάστοτε λογισμικό και να προκαλέσουν αλλαγές κατά το δοκούν. Αναφορικά, ο επιτιθέμενος μπορεί να [11], [12], [13], [14]:

- Μεταβάλλει αυθαίρετα τιμές σε μία Βάση Δεδομένων μέσω μίας κατηγορίες *Code Injection* που ονομάζεται *SQL Injection.* Το αποτέλεσμα αυτής της επίθεσης ποικίλει από αλλαγή στην εμφάνιση μίας ιστοσελίδας έως πλήρη υποκλοπή σημαντικών δεδομένων.
- Εγκατάσταση κακόβουλου λογισμικού ή εκτέλεση κακόβουλου κώδικα σε έναν *server,* με την εισαγωγή κώδικα σεναρίου για *server (PHP η ASP).*
- Αύξηση των δικαιώματων του σε επίπεδο διαχειριστή
- Προσβάλλει διαδικτυακές εφαρμογές με *HTML/Script Injection (Cross-site Scripting).*

2.2.3.2 DLL Injection

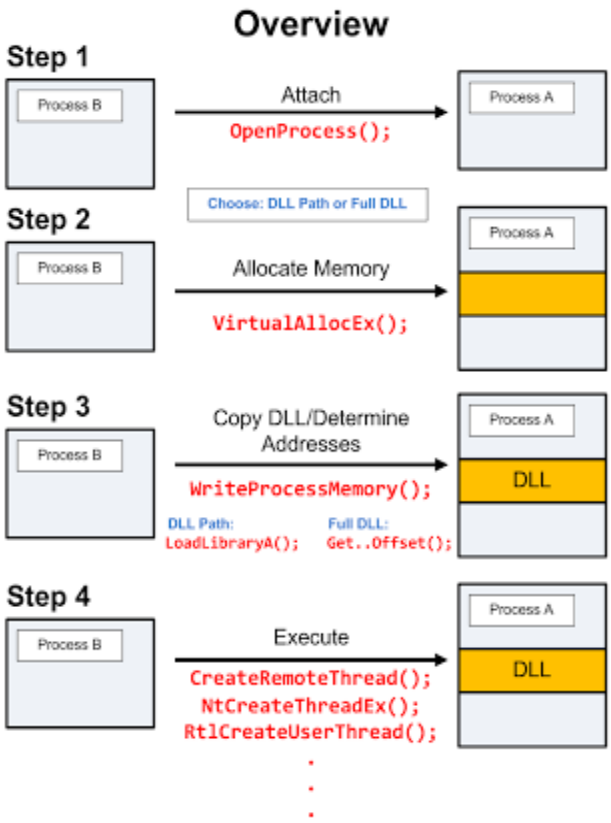
Η Τεχνική *DLL Injection* αφορά στην εισαγωγή κώδικα σε μία διαδικασία που εκτελείται. Η διαφορά με το *Code Injection* είναι ο κώδικας που εισάγεται είναι με την μορφή μίας *Dynamic Linked Library,* καθώς οι *DLL* σε ένα πρόγραμμα φορτώνονται κατά την εκτέλεση, γεγονός που κάνει αυτού του είδους την επίθεση σχεδόν αδύνατο να ανιχνευτεί, αν το πρόγραμμα – στόχος δεν τρέχει. Η επίθεση αυτή έχει ως στόχο την φόρτωση κώδικα στη μνήμη μίας διεργασίας, που δύναται να αλλάξει την συμπεριφορά της, μέσω του “εξαναγκασμού” της διεργασίας – θύματος να φορτώσει την DLL κατά την περίοδο λειτουργίας.

Αυτού του είδους τεχνική επίθεσης απαιτεί, το κακόβουλο λογισμικό που θα πραγματοποιήσει το *DLL Injection*, να κατέχει δικαιώματα διαχειριστή. Ο λόγος για την απαίτηση αυξημένων δικαιωμάτων χρήσης πηγάζει από το γεγονός ότι τεχνική βασίζεται σε συναρτήσεις αποσφαλμάτωσης (Debugging) που προσφέρει το *Windows API*, οι οποίες επιτρέπουν την προσκόλληση και διαχείριση της μνήμης προγραμμάτων. Οι συναρτήσεις

αποσφαλμάτωσης αποτελούν κορυφαίες οντότητες στην διαδικασία εύρεσης σφαλμάτων σε διεργασίες, προσφέροντας πλήρη έλεγχο και εποπτεία στον τρόπο λειτουργίας μίας διεργασίας, δίνοντας παράλληλα την δυνατότητα αλλαγής δεδομένων στην μνήμη της. Ωστόσο, οι απεριόριστες δυνατότητες που προσφέρει η αποσφαλμάτωση αποτελούν παράλληλα και το παραθυράκι για τους κακόβουλους *Hackers,* μέσω του οποίου καταφέρνουν να επηρεάσουν ένα νόμιμο πρόγραμμα που τρέχει σε έναν υπολογιστή.

Εν γένει, η διαδικασία του *DLL Injection* χρησιμοποιείται κατά κόρον από κακόβουλα λογισμικά, με αποτέλεσμα να έχουν εντοπιστεί κάποιες νόρμες στον τρόπο με τον οποίο προγραμματίζεται να δρα αυτή τη τεχνική. Πιο συγκεκριμένα:

1. Προσκόλληση σε μία διεργασία.
2. Διαμοιρασμός καινούργιων θέσεων μνήμης μέσα στον χώρο διευθύνσεων της διεργασίας.
3. Αντιγραφή της βιβλιοθήκης DLL ή της διαδρομής δίσκου στην οποία βρίσκεται, (σχετική διαδρομή), μέσα στην μνήμη της διεργασίας και ρύθμιση κατάλληλων διευθύνσεων μνήμης.
4. “Διαταγή” της διεργασίας να εκτελέσει την βιβλιοθήκη μέσω συναρτήσεων αποσφαλμάτωσης και δικαιωμάτων διαχειριστή.



Εικόνα 3: Διαδικασία Πραγματοποίησης DLL Injection

Η εκτέλεση της εισαγωγής βιβλιοθήκης μπορεί να πραγματοποιηθεί με διάφορες συναρτήσεις αποσφαλμάτωσης, κάθε μία από τις οποίες αποτελεί μία διαφορετική προσέγγιση παραβίασης, προσφέροντας διαφορετικά θετικά και αρνητικά σημεία για τον κακόβουλο προγραμματιστή. Ενδεικτικά κάποιες μέθοδοι *DLL Injection* που χρησιμοποιούνται ευρέως είναι:

- Χρήση της συνάρτησης *QueueUserAPC.*
- Χρήση της συνάρτησης *SetWindowsHookEx* – μέθοδος με περισσότερες δυνατότητες διήθησης αλλά και πρόκληση περισσοτέρων ενδείξεων προσβολής.
- Η μέθοδος "*Code Cave*" - έγχυση κώδικα στην διεργασία – θύμα μέσω διαμοιρασμού καινούργιων θέσεων μνήμης στην διεργασία.

3. ΜΕΛΕΤΗ ΚΑΚΟΒΟΥΛΟΥ ΛΟΓΙΣΜΙΚΟΥ

Η μελέτη ενός κακόβουλου λογισμικού αποτελεί μια διαδικασία αντίστροφης μηχανικής (Reverse Engineering) κατά την οποία το λογισμικό, αφού εντοπισθεί, υποβάλλεται σε τεχνικές παρακολούθησης της συμπεριφοράς του και αποδόμησης του κώδικα του για την κατανόηση της λειτουργίας του, ενώ στο τέλος συνάγονται συμπεράσματα προστασίας από αυτό. Η μελέτη, λοιπόν είναι προφανές ότι περιλαμβάνει τρία στάδια

1. Εντοπισμός Κακόβουλου Λογισμικού
2. Ανάλυση κακόβουλου Λογισμικού
3. Εξαγωγή μέτρων προστασίας από το συγκεκριμένο Κακόβουλο Λογισμικό.

Κάθε στάδιο παρουσιάζει ξεχωριστές προκλήσεις και απαιτεί συγκεκριμένες γνώσεις και προσεκτικό χειρισμό. Το γεγονός της ταχείας ανάπτυξης νέων και τεχνολογικά υπέρτερων κακόβουλων λογισμικών κάθε μέρα, αποτελεί το μοχλό πίεσης στους ενάγοντες την μελέτη, δημιουργώντας στενά χρονικά όρια στα οποία θα πρέπει να έχουν εξαχθεί συμπεράσματα. Ειδάλλως, η ετεροχρονισμένη εξαγωγή συμπερασμάτων καθίσταται άχρηστη καθώς ο χρήστης έχει ήδη υποστεί βλάβη και το κακόβουλο λογισμικό έχει ενδεχομένως περάσει σε νέο στάδιο εξέλιξης με νέες δυνατότητες πρόκλησης ζημιάς στο σύστημα του χρήστη. Αξίζει να σημειωθεί πώς η μελέτη κακόβουλων λογισμικών δεν αποτελεί γραμμική διαδικασία, αλλά κυκλική, η οποία ουσιαστικά δεν τελειώνει ποτέ, ακολουθώντας την εξέλιξη του κάθε κακόβουλου λογισμικού. Κάθε στάδιό της λειτουργεί με είσοδο τα στοιχεία που έχει παράγει σαν έξοδο το προηγούμενο στάδιο αλλά και με την χρήση στοιχείων ανατροφοδότησης από τα τεκταινόμενα.

3.1 Εντοπισμός Κακόβουλων Λογισμικών

Ο εντοπισμός κακόβουλων λογισμικών αποτελεί το πρώτο στάδιο αντιμετώπισης τους, θέτοντας σε συναγερμό όλες τις απαραίτητες ενέργειες για την μετάβαση στο στάδιο της ανάλυσης. Προφανώς, κάνοντας λόγο για εντοπισμό κακόβουλου λογισμικού, εννοούμε λογισμικό πρωτοφανές και μη καταγεγραμμένο ή μελετημένο, αλλιώς δεν υπάρχει λόγος μελέτης. Ο εντοπισμός ενός πρωτοφανούς λογισμικού στο υπολογιστή, το οποίο δεν είναι ευρέως γνωστό και δεν υπάρχει σε κάποια Βάση Δεδομένων με μαύρη λίστα κακόβουλων

προγραμμάτων, μπορεί να είναι είτε ηθελημένος είτε τυχαίος. Ο τυχαίος εντοπισμός εμπεριέχει το στοιχείο της προσεκτικής παρατήρησης του συστήματος για τυχόν ανώμαλη συμπεριφορά. Η παρατήρηση κάποιας διαφορετικότητας στην συνηθισμένη λειτουργία του συστήματος μπορεί να υποδείξει στον παρατηρητή την ύπαρξη κάποιου κακόβουλου λογισμικού και να πυροδοτήσει την περαιτέρω έρευνα. Ο ηθελημένος εντοπισμός αφορά στην γνώση της ύπαρξης κάποιου κακόβουλου λογισμικού και στην υποβολή του συστήματος σε προσεκτική καραντίνα για την εύρεση της κακόβουλης οντότητας. Επίσης, σε περίπτωση που υπάρχει η γνώση για ύπαρξη κακόβουλου λογισμικού μέσω κάποιων ενδείξεων, το σύστημα μπορεί να προσομοιωθεί με την δημιουργία μιας εικονικής μηχανής με τα ίδια χαρακτηριστικά. Έτσι, αποφεύγεται ο κίνδυνος για περαιτέρω πρόκληση βλάβης στο σύστημα, ενώ και η παρακολούθηση των πόρων της εικονικής μηχανής είναι πιο ελεγχόμενη.

Η στενή παρακολούθηση ενός συστήματος σε καθημερινή βάση αποτελεί την πρώτη γραμμή άμυνας απέναντι στους *Hackers.* Η παρείσφρηση κακόβουλων λογισμικών μπορεί να συμβεί ανά πάσα στιγμή και δύναται να προκαλέσει από μικρής σημασίας ενοχλήσεις έως εξαιρετικά σοβαρή απώλεια δεδομένων και αχρήστευση του λειτουργικού. Το είδος των λογισμικών αυτών είναι αυτό που καθορίζει την συμπεριφορά τους και η συμπεριφορά τους είναι αυτή που καθορίζει τον τρόπο ανίχνευσης από τον παρατηρητή. Αν το κακόβουλο λογισμικό έχει σχεδιαστεί με απλή αρχιτεκτονική και επιτελεί βασικές λειτουργίες, τότε και η συμπεριφορά του δεν παρουσιάζει ειδική επιτήδευση. Αντίθετα, ένα πολύπλοκο αρχιτεκτονικά λογισμικό, το οποίο χρησιμοποιεί πολλές τεχνικές για την επίτευξη του σκοπού του, παρουσιάζει επιτηδευμένη συμπεριφορά και κατατάσσεται υβρίδιο ως προς αυτήν. Ο βαθμός πολυπλοκότητας δράσης του κακόβουλου λογισμικού καθορίζει και τον βαθμό δυσκολίας ανίχνευσης του. Δηλαδή, ένα προσεκτικά μελετημένο λογισμικό, που χρησιμοποιεί διακριτικές μεθόδους παρέμβασής στο σύστημα και που το αποτέλεσμα του μεταμφιέζεται καταλλήλως, είναι δύσκολο να γίνει αντιληπτό με την χρήση μέτρων αποκάλυψης.

Παράλληλα, η γνώση των αποτελεσμάτων που παράγονται από την χρήση των τεχνικών που επιστρατεύει το κάθε ένα κακόβουλο λογισμικό αποτελεί πολύτιμη αρωγό και καθοριστικό παράγονται στην ταχεία αποκάλυψη της δράσης αυτού. Για παράδειγμα, η γνώση της αντιστοιχίας μεταξύ των διαφόρων συναρτήσεων αποσφαλμάτωσης που χρησιμοποιούνται και των συγκεκριμένων ενδείξεων – mutexes που εγείρονται στο σύστημα, δίνει την δυνατότητα στον παρατηρητή, παρακολουθώντας το δεύτερο, να εντοπίζει το πρώτο. Συνεπώς, γίνεται εύκολα αντιληπτό ότι η μελέτη και η καλή γνώση της

λειτουργίας υγιών λειτουργικών συστημάτων πριμοδοτεί τον εντοπισμό ανωμαλιών που οφείλονται σε εξωγενείς παράγοντες. Επίσης, μεταβολές στην ομαλή λειτουργία ενός λειτουργικού συστήματος που δεν γίνονται αντιληπτές με μια πρώτη παρατήρηση, μπορούν να εντοπισθούν με την χρήση διαφόρων εργαλείων, όπως ένας ανιχνευτής της κίνησης δικτύου (Netstat), που είτε βρίσκονται ενσωματωμένα στο λειτουργικό σύστημα είτε προσφέρονται ελεύθερα στο διαδίκτυο.

Διαφορετική περίπτωση αποτελεί η αναγνώριση ύποπτης συμπεριφοράς σε ένα σύστημα, όταν ο παρατηρητής είναι υποψιασμένος για την ύπαρξη κακόβουλου λογισμικού. Στην περίπτωση αυτή ο παρατηρητής έχει ενημερωθεί για κυκλοφορία κακόβουλου λογισμικού, πιθανώς από κάποια λήμματα στο διαδίκτυο, ενεργώντας προσεκτικά και πραγματοποιώντας ορισμένες κινήσεις για την αποκάλυψη της παράνομης δραστηριότητας. Για παράδειγμα, ο παρατηρητής δύναται να απενεργοποιήσει την σύνδεση στο διαδίκτυο και να παρακολουθήσει τη δραστηριότητα των διεργασιών που είναι ενεργές στο σύστημα. Μία προσεκτική ματιά ίσως αποκαλύψει κακόβουλο λογισμικό το οποίο ενώ έχει υποκλέψει στοιχεία από λογαριασμό του χρήστη, προσπαθεί να τα στείλει σε κάποιον απομακρυσμένο εξυπηρετητή και η διακοπή της σύνδεσης στο διαδίκτυο προκαλέσει την αδράνεια του έως ότου παρατηρηθεί επανασύνδεση. Επιπροσθέτως, η χρήση μίας εικονικής μηχανής η οποία είναι ρυθμισμένη να τρέχει το ίδιο λειτουργικό σύστημα με τις ίδιες παραμέτρους, μπορεί να υποδείξει στον παρατηρητή κάποια συμπεριφορά η οποία δεν παρατηρείται σε ένα κανονικό μηχάνημα, καθώς πολλά κακόβουλα λογισμικά πλέον έχουν την δυνατότητα να ανιχνεύουν αν το σύστημα που έχουν προσβάλει τρέχει σε υποδομή εικονικής μηχανής ή όχι.

Παράλληλα με όλες τις παραπάνω μεθόδους υπάρχουν και κάποια εργαλεία η χρήση των οποίων δρα ως πολύτιμη αρωγός στην αφύπνιση του παρατηρητή και τον εντοπισμό της κακόβουλης οντότητας. Τέτοια είναι το *Yara* και *Cuckoo*. Το πρώτο βοηθά στην ενημέρωση του χρήστη για εκτελέσιμα αρχεία ή διεργασίες που τρέχουν και εμπεριέχουν ύποπτες αλφαριθμητικές μεταβλητές, οι οποίες μπορεί να υποδεικνύουν μέχρι και αυτούσια τα ονόματα των συναρτήσεων αποσφαλμάτωσης που χρησιμοποιεί κάποιο κακόβουλο λογισμικό. Το δεύτερο χρησιμεύει κατά την περίπτωση που ο παρατηρητής είναι ήδη υποψιασμένος για της ύπαρξη κακόβουλης δραστηριότητας και θέλει να υποβάλει ένα σύστημα εικονικής μηχανής (όμοιο με το πραγματικό) σε ελεγχόμενες συνθήκες εκτέλεσης ορισμένων ύποπτων αρχείων. Για τα δύο αυτά εργαλεία θα γίνει ειδική μνεία σε επόμενο κεφάλαιο.

3.2 Ανάλυση Κακόβουλου Λογισμικού

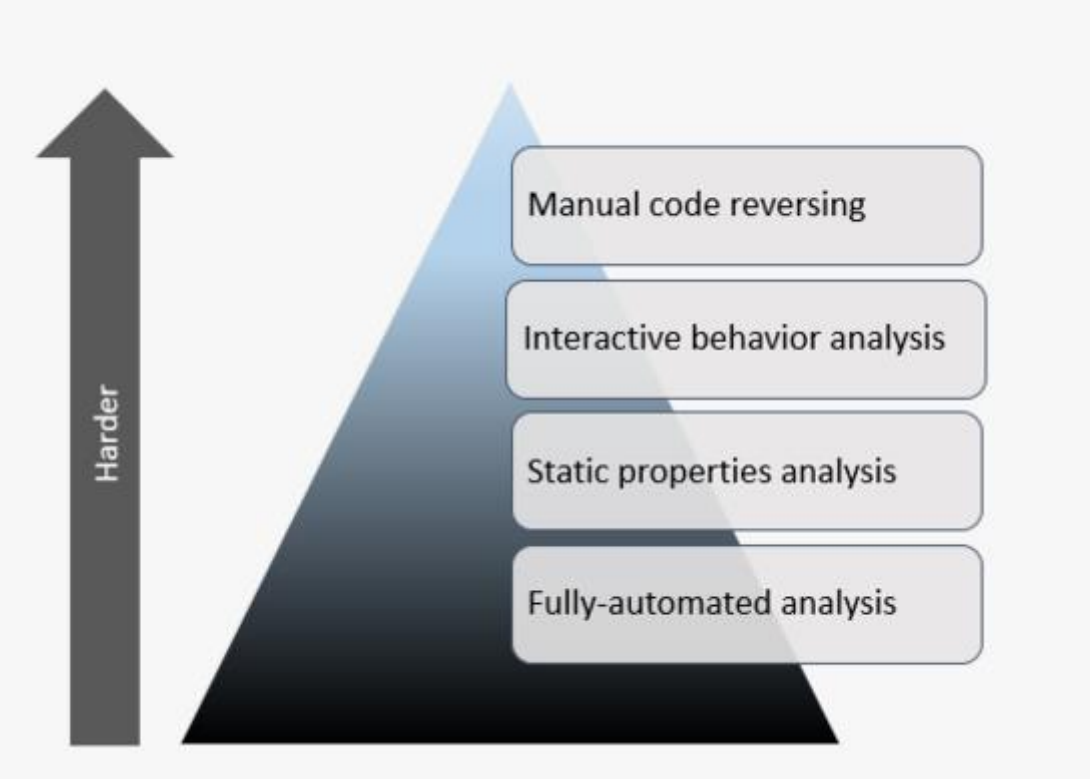
Το στάδιο της ανάλυσης των εντοπισμένων κακόβουλων λογισμικών αποτελεί την πιο επίπονη διαδικασία στην άμυνα απέναντι στις κακόβουλες δραστηριότητες. Στο στάδιο αυτό, εφόσον έχει εγερθεί συναγερμός για την παρουσία κακόβουλου κώδικα, ο παρατηρητής θα προσπαθήσει να μελετήσει την συμπεριφορά του επιτιθέμενου προγράμματος και να εξάγει συμπεράσματα που θα τον βοηθήσουν να εξυγιάνει το σύστημα και να το προστατεύσει από μέλλουσες επιθέσεις του ίδιου τύπου.

Η ανάλυση ενός κακόβουλου λογισμικού πρωτίστως εξαρτάται από το είδος του λειτουργικό στο οποίο έχει σχεδιαστεί να προσβάλει. Κάθε λειτουργικό έχει ποικίλες ιδιαιτερότητες και τα πολλά λογισμικά προγράμματα που εκτελούνται σε αυτό έχουν σχεδιαστεί αποκλειστικά για αυτό. Για αυτού του είδους τα λογισμικά, η ανάλυση ακολουθεί διαφορετικά βήματα και τεχνικές ανάλογα στο λειτουργικό που απευθύνονται. Παράλληλα, ωστόσο σε ένα σύστημα εκτελούνται και πολλά προγράμματα που έχουν σχεδιαστεί με φιλοσοφία προσανατολισμένη στην ανεξαρτησία από το λειτουργικό σύστημα στο οποίο εκτελούνται. Η ανάλυση τέτοιων προγραμμάτων μπορεί να έχει κοινά στοιχεία για όλα τα λειτουργικά συστήματα καθώς και μπορεί να ακολουθηθούν παρόμοια βήματα μελέτης της συμπεριφοράς του εξεταζόμενου λογισμικού. Παράδειγμα αποτελούν τα κακόβουλα λογισμικά που προσβάλουν την λειτουργία ενός φυλλομετρητή.

Ο άλλος πιο σημαντικός παράγοντας που καθορίζει σημαντικά την πορεία, τα βήματα και τον προσανατολισμό της ανάλυσης είναι η επιλογή του τρόπου με την οποία επιθυμεί ο αναλυτής να προσεγγίσει το κακόβουλο λογισμικό. Οι κατηγορίες της ανάλυσης που αφορούν σε αυτή την επιλογή είναι δύο: η *Στατική* και η *Δυναμική* ανάλυση κακόβουλων λογισμικών. Κάθε μία έχει τις δικές της ιδιαιτερότητες, αρετές και μειονεκτήματα. Όλα αυτά θα αναλυθούν στις επόμενες ενότητες. Ωστόσο, ο τύπος της ανάλυσης δεν εξαρτάται μόνο από την επιλογή του αναλυτή, καθώς τις περισσότερες φορές μία εξεταζόμενη περίπτωση απαιτεί την χρήση και των δύο τεχνικών που αναφέρθηκαν. Επιπλέον, η ανάλυση μπορεί να εμπεριέχει και άλλα δύο στάδια, το στάδιο της πλήρους αυτοματοποιημένης ανάλυσης και το στάδιο της αντίστροφης μηχανικής.

Εξετάζοντας, λοιπόν, την ανάλυση των κακόβουλων λογισμικών σαν μία ολοκληρωμένη διαδικασία που περιέχει συγκεκριμένα βήματα ανεξάρτητα της περίπτωσης που εξετάζεται, παρατηρούμε την διαστρωμάτωση των τεχνικών ανάλυσης στην μορφή μίας πυραμίδας.

(Εικόνα 4). Στην πυραμίδα αυτή, οι τεχνοτροπίες ανάλυσης γίνονται όλο και πιο πολύπλοκες όσο πλησιάζουμε στην κορυφή. Ωστόσο, όσο πιο πολύπλοκες γίνονται τόσο μεγαλύτερη διόραση στην συμπεριφορά του κακόβουλου λογισμικού προσφέρουν. Επίσης, το πρώτο στάδιο, στην βάση της πυραμίδας, σε σχέση με το τελευταίο στάδιο, στην κορυφή της πυραμίδας, απέχουν παρασάγγας όσον αφορά στον χρόνο που χρειάζεται για να επιφέρουν αποτελέσματα. Έτσι, στην περίπτωση που δεν υπάρχει χρόνος για χρονοβόρα ανάλυση, δεν είναι υποχρεωτικό να εκτελεστούν όλα τα βήματα, αφού μπορεί και ο συνδυασμός κάποιων από αυτά να επιφέρει ορθό και ακριβές αποτέλεσμα. Κάθε περίπτωση λοιπόν μπορεί να απαιτήσει κάποιο συνδυασμό των τεχνοτροπιών, αφού δεν είναι απαραίτητο να ακολουθεί αυστηρά η εν λόγω διαστρωμάτωση με την σειρά.



Εικόνα 4: Διαστρωμάτωση Διαδικασιών Ανάλυσης Κακόβουλων Λογισμικών [15]

3.2.1 Πλήρως αυτοματοποιημένη ανάλυση κακόβουλων λογισμικών

Η κατηγορία της αυτοματοποιημένης ανάλυσης κακόβουλων λογισμικών αφορά σε λογισμικά σχεδιασμένα για την παροχή υπηρεσιών ανάλυσης κακόβουλων λογισμικών. Οι υπηρεσίες αυτές δεν απαιτούν κάποια παρέμβαση από τον αναλυτή και εξάγουν αποτελέσματα που αφορούν πληροφορίες για κλειδιά μητρώων, τιμές των *mutexes,* δραστηριότητα αρχείων, δικτυακή κίνηση κ.λ.π. Η πληροφορίες που προκύπτουν δεν

προσφέρουν το ίδιο επίπεδο διόρασης στον τρόπο λειτουργίας του κακόβουλο λογισμικού, όπως προσφέρει η ενδελεχής ανάλυση που πραγματοποιείται από έναν αναλυτή. Ωστόσο, η χρήση τέτοιων εργαλείων επιβάλλεται ειδικά από τούς μεγάλους οργανισμούς, στους οποίους το πλήθος των κακόβουλων λογισμικών και η ταχύτητα εμφάνισης τους είναι τέτοια που δεν μπορούν οι αναλυτές να ασχοληθούν με όλα. Τα εργαλεία αυτοματοποίησης της εν λόγω διαδικασίας χρησιμοποιούνται για την στόχευση των κακόβουλων λογισμικών τα οποία είναι σοβαρά και απαιτούν την προσοχή των αναλυτών, παρέχοντας σε αυτούς τις απαραίτητες ενδείξεις προσβολής (IOC) για να πιστοποιήσουν την σοβαρότητα του επιτιθέμενου λογισμικού. Ωστόσο τα εργαλεία αυτά δεν αποτελούν μαύρο κουτί, καθώς τα περισσότερα είναι ανοιχτού κώδικα. Αυτό πρακτικά σημαίνει ότι ο κάθε αναλυτής – προγραμματιστής γνωρίζοντας αρκετά στοιχεία για το εργαλείο που χρησιμοποιεί και χρησιμοποιώντας το *API,* που τα περισσότερα παρέχουν, για την ανάπτυξη καινούργιων ενδείξεων προσβολής, πιο περίτεχνων και ειδικευμένων για συγκεκριμένους τύπους λογισμικών, οι οποίες καθιστούν τα εργαλεία πιο “ικανά” και “έξυπνα”. Με το τρόπο αυτό, ο αναλυτής χτίζει μόνος του την πρώτη γραμμή στην ανάλυση, κοσκινίζοντας τα δεδομένα εισόδου και εξάγοντας πληροφορίες και στατιστικά στοιχεία για την αντιμετώπιση των απειλών.

Τα εργαλεία που αναφέρθηκαν παραπάνω χωρίζονται σε δύο κατηγορίες:

1. *Web – Based* εργαλεία συμπεριφορικής ανάλυσης κακόβουλων λογισμικών.
2. Τοπικά στο υπολογιστή εργαλεία ανάλυσης.

Στη πρώτη κατηγορία ανήκουν τα εργαλεία όπως το *Malwr* και το *Deepviz Malware Analyzer* που είναι λογισμικά με υλοποίηση στο διαδίκτυο και προσφέρουν πληθώρα λειτουργιών ανάλυσης ενός λογισμικού, πιθανώς κακόβουλου, και παράγουν αρκετά χρήσιμες πληροφορίες για την κατανόηση του υποβαλλομένου αρχείου. Στη δεύτερη κατηγορία ανήκουν εργαλεία τα οποία εγκαθίστανται τοπικά στον υπολογιστή του αναλυτή και τρέχουν τοπικά, παρέχοντας μία μεγάλη ποικιλία τεχνοτροπιών για την ανάλυση αρχείων. Το πλεονέκτημα της πρώτης κατηγορίας είναι η εύκολη υποβολή για ανάλυση κάποιου αρχείου, χωρίς να υπάρχει η ανάγκη για την συχνά δύσκολη και χρονοβόρα εγκατάσταση λογισμικού σε κάθε υπολογιστή του αναλυτή. Ωστόσο, η δεύτερη κατηγορία έχει το πολύτιμο πλεονέκτημα της παραμετροποίησης των εργαλείων ανάλυσης. Η παραμετροποίηση των εργαλείων από τον αναλυτή, ώστε να αντιστοιχούν στις πολύ συγκεκριμένες ιδιαιτερότητες και ανάγκες κάθε συστήματος, όπως στην περίπτωση της εγκατάστασης κάποιου *Sandbox (Cuckoo),* συντελεί στην δημιουργία συστημάτων για βαθιά και ενδελεχή ανάλυση λογισμικών.

3.2.2 Στατική ανάλυση κακόβουλων λογισμικών

Η στατική ανάλυση κακόβουλων λογισμικών αποτελεί ένα επόμενο της αυτοματοποιημένης ανάλυσης βήμα που προσφέρει πληροφορίες περισσότερο ενδελεχής και κρυμμένες από την εποπτεία των κοινών εργαλείων. Η στατική ανάλυση φανερώνει τα δομικά στοιχεία του εξεταζόμενου λογισμικού και συνδέεται στενά με την εύρεση της αρχιτεκτονικής και της λογικής με την οποία ο κάθε *Hacker* υλοποιεί το κακόβουλο λογισμικό του. Η ανάλυση στηρίζεται μόνο σε στατικά στοιχεία που περιγράφουν το λογισμικό, ενώ τα στοιχεία που προκύπτουν από τα δυναμικά χαρακτηριστικά εκτέλεσης του δεν συμπεριλαμβάνονται. Αυτό οφείλεται στο γεγονός ότι το λογισμικό εξετάζεται ως προς το εκτελέσιμο αρχείο από το οποίο αυτό εκτελείται και οποιοδήποτε συμπέρασμα αφορά αποκλειστικά χαρακτηριστικά αυτού καθώς και άλλων στοιχείων που γνωρίζουμε ότι καλούνται κατά την φάση της εκτέλεσης, όπως είναι οι δυναμικές βιβλιοθήκες *DLL (*η γνώση αυτή δεν συνεπάγεται από αυτού του είδους την ανάλυση αλλά προκύπτει από την διενέργεια δυναμικής ανάλυσης)*.*

Η στατική ανάλυση είναι χρονοβόρα και απαιτεί από τον αναλυτή πλούσια εμπειρία και βαθιά γνώση του αντικειμένου. Η δομική εξέταση του λογισμικού είναι μία διαδικασία που στην οποία κάθε βήμα στοχεύει στην αποκάλυψη συγκεκριμένων στοιχείων και ο αναλυτής πρέπει να ξέρει που να κατευθυνθεί στην έρευνά του. Σε αντίθεση με την δυναμική ανάλυση, ο αναλυτής δεν υποψιάζεται από περίεργες δραστηριότητες αλλά στοχεύει στην αποκάλυψη στοιχείων ακόμα και όταν αυτά έχουν υποστεί κατάλληλη επεξεργασία από τεχνικές συσκότισης (*Obfuscation techiques*). Επίσης, η διενέργεια στατικής ανάλυσης δεν είναι υποχρεωτικά ένα βήμα πριν το στάδιο της δυναμικής ανάλυσης, όπως παρουσιάζεται στην παραπάνω πυραμίδα, αλλά πολλές φορές και σε συνάρτηση με το είδος της απειλής που εξετάζεται απαιτείται αυτή η σειρά να αλλάξει με σκοπό να υπάρχει μεγαλύτερη κατανόηση του υπό εξέταση λογισμικού. [16]

Η στατική ανάλυση χωρίζεται εμπειρικά και σε συνάρτηση με τα εργαλεία που εν γένει χρησιμοποιούνται σε Βασική και Προχωρημένη στατική ανάλυση. Η Βασική στατική ανάλυση περιλαμβάνει τις εξής δραστηριότητες:

- Διενέργεια της τεχνικής του κατακερματισμού (*Hashing*) στο αρχείου του κακόβουλου λογισμικού για την ταυτοποίηση και αναγνώριση του.
- Ανίχνευση Πακεταρίσματος ή Συσκότισης (*Packed/Obfuscated*) του λογισμικού για να γίνει πιο εύκολη η διαδικασία ανίχνευσης και ανάλυσης του.
- Διενέργεια ανάλυσης της δομής του εκτελέσιμου αρχείου (*PE*)

Χαρακτηριστικά παραδείγματα εργαλείων Βασικής στατικής ανάλυσης αποτελούν τα εξής:

- *VirusTotal.com*
- *Md5deep*
- *PeiD*
- *Exexinfo PE*
- *D4dot*
- *PEView*
- *EXE*
- *PeStudio*

Η Προχωρημένη στατική ανάλυση διενεργείται σε ένα επίπεδο βαθύτερα από ότι η Βασική και περιλαμβάνει:

- Ανάλυση των αλφαριθμητικών που περιέχονται στο εκτελέσιμο αρχείο του κακόβουλου λογισμικού.
- Ανάλυση των *DLL* που φορτώνονται από το εκτελέσιμο του κακόβουλου λογισμικού.
- Ανάλυση των συναρτήσεων που περιέχονται στο κώδικα του εκτελέσιμου αρχείου.
- Διενέργεια αποσυναρμολόγησης (*Disassemble)* του κώδικα και μελέτη των εντολών του προγράμματος σε επίπεδο γλώσσας μηχανής.

Βασικά εργαλεία της προχωρημένης στατικής ανάλυσης αποτελούν τα εξής:

- *BinText*
- *Strings.exe (SysInternalsSuite)*
- *Dependancy Walker*
- *Radare2*

3.2.3 Δυναμική ανάλυση κακόβουλων λογισμικών

Η δυναμική ανάλυση κακόβουλων λογισμικών βρίσκεται ένα βήμα υψηλότερα στην πυραμίδα της ανάλυσης. Σκοπός της είναι η μελέτη της συμπεριφοράς του κακόβουλου λογισμικού την ώρα που αυτό δρα. Για το λόγο αυτό ονομάζεται και συμπεριφορική ανάλυση. Η μελέτη αυτή δύναται να προσφέρει ακόμα πιο διορατικά αποτελέσματα από την στατική ανάλυση, καθώς το λογισμικό εξετάζεται ως προς την εργασία που του έχει ανατεθεί να κάνει, γεγονός που δίνει αμεσότητα και μία ολοκληρωτική κάλυψη των κακόβουλων

ενεργειών για να βλάψει τον χρήστη. Επίσης, με τον τρόπο αυτό, υπερπηδούνται τα εμπόδια απόκρυψης του πηγαίου κώδικα που μπορεί να έχει βάλει ο κακόβουλος προγραμματιστής, ενώ παράλληλα γίνεται και εύρεση κάποιων στοιχείων που συνδέονται άρρηκτα με το κακόβουλο λογισμικό (*DLL,* αρχεία) και που δεν μπορεί να αποκαλύψει πλήρως η στατική ανάλυση. Ακόμη, η δυναμική ανάλυση προσθέτει στην μάχη κατά των κακόβουλου λογισμικού και το στοιχείο της παρακολούθησης της κίνησης προς το διαδίκτυο, ενέργεια την οποία πράττουν τα περισσότερα λογισμικά που έχουν σκοπό να αποκαλύψουν ιδιαίτερα σημαντικές πληροφορίες για το χρήστη σε μία απομακρυσμένη τοποθεσία. Η αποτύπωση μοτίβων διαδικτυακής κίνησης είναι μία υψίστης σημασίας διαδικασία για την καταστολή ενεργειών υποκλοπής προσωπικών στοιχείων. Επιπροσθέτως, η μελέτη της μνήμης *RAM* και του τρόπου χρησιμοποίησης της από το κακόβουλο λογισμικό προσφέρει πολλές πληροφορίες για την συνολική συμπεριφορά του. Όλα τα παραπάνω, προσφέρουν ακόμα περισσότερη πληροφορία όταν ο αναλυτής αλληλεπιδρά με το κακόβουλο λογισμικό μέσα σε ένα ελεγχόμενο περιβάλλον, παρά όταν απλά παρατηρεί την δράση του. Η εν λόγω αλληλεπίδραση, αφορά στην κατά βούληση μίμηση διαφόρων δραστηριοτήτων με σκοπό να προκληθούν συγκεκριμένες συμπεριφορές του κακόβουλου λογισμικού. Αυτή η ικανότητα της κατηγορίας της δυναμικής ανάλυσης είναι αυτή που την κάνει ποιο αποδοτική από τα προτυποποιημένα αποτελέσματα των αυτοματοποιημένων αναλυτικών εργαλείων.

Τα μειονεκτήματα αυτής της ανάλυσης είναι κυρίως τεχνικά και δεν αφορούν την διαδικασία αυτή καθ' αυτή. Η δυναμική ανάλυση απαιτεί σχολαστική προσοχή καθώς η ιδιαιτερότητα της έγκειται αφενός στην εγκατάσταση του απαραίτητου περιβάλλοντος για την διενέργεια της εγκατάστασης, συνήθως σε κάποιο *sandbox* και αφετέρου στον βαθμό επικινδυνότητας που την χαρακτηρίζει όταν αυτή διενεργείται σε ζωντανό περιβάλλον και όχι σε κάποια εικονική μηχανή. Όσον αφορά στο πρώτο, είναι σημαντικό να τονιστεί πώς το η διενέργεια της δυναμικής ανάλυσης σε μία εικονική μηχανή διατρέχει πολλές επιμέρους δυσκολίες, σημαντικότερη από τις οποίες είναι η ικανότητα πολλών σύγχρονων κακόβουλων λογισμικών να ανιχνεύουν αν εκτελούνται σε εικονική μηχανή ή όχι. Η προσεκτική παραμετροποίηση, λοιπόν, τόσο του περιβάλλοντος του εικονικού λειτουργικού συστήματος για να μοιάζει αυθεντικό όσο και των σημείων που εμποδίζουν το κακόβουλο να καταλάβει τον χώρο στον οποίο εκτελείται και πιθανότατα να μεταπηδήσει στο πραγματικό λειτουργικό του αναλυτή, αποτελεί σημαντικότητοο παράγοντα στην επιτυχία της ανάλυσης. Σχετικά με το δεύτερο προβληματισμό, ο αναλυτής πρέπει, στην περίπτωση που επιλέξει να μην χρησιμοποιήσει εικονική μηχανή αλλά το ίδιο του το σύστημα, να διασφαλίζει την ακεραιότητα του συστήματος και την ανακτησιμότητα της υγιούς κατάστασης του συστήματός του, ώστε να διασφαλιστεί και η επιτυχία της διενέργειας της ανάλυσης. Επίσης

ένα μολυσμένο σύστημα, στο οποίο εκτελούνται τα εργαλεία της δυναμικής ανάλυσης, εμπεριέχει τον κίνδυνο να μην μπορεί να παράγει αξιόπιστα αποτελέσματα λόγω ισχυρής παρέμβασης κάποιου πολύ ανεπτυγμένου κακόβουλου λογισμικού.

Η δυναμική ανάλυση χωρίζεται και αυτή σε Βασική και Προχωρημένη. Τα δύο επίπεδα αντικατοπτρίζουν το βαθμό στον οποίο έχει εντρυφήσει ο αναλυτής στην μελέτη του για το κακόβουλο λογισμικό. Η Βασική δυναμική ανάλυση αποτελείται από τις εξής διεργασίες:

- “Στήσιμο” της εικονικής μηχανής και του εικονικού λειτουργικού συστήματος.
- Εγκατάστασης ενός περιβάλλοντος *sandbox.*
- Παρακολούθηση της διεργασίας του κακόβουλου λογισμικού.
- Ανάλυση πακέτων δεδομένων παραγόμενων από το κακόβουλο λογισμικό.

Κάποια από τα πιο δημοφιλή και βασικά εργαλεία της Βασικής δυναμικής ανάλυσης είναι:

- *VirtualBox*
- *Cuckoo sandbox*
- *Process Monitor*
- *Process Explorer*
- *ApateDNS*
- *WireShark*

Η Προχωρημένη δυναμική ανάλυση αφορά στις παρακάτω βασικές διεργασίες:

- Αποσφαλμάτωση του κακόβουλου λογισμικού.
- Ανάλυση των μητρώων.
- Ενδελεχής ανάλυση στο λειτουργικό σύστημα.

Ορισμένα βασικά παραδείγματα εργαλείων της Προχωρημένης δυναμικής ανάλυσης είναι τα εξής:

- *x64dbg*
- *Regshot*

3.2.4 Αντίστροφη Μηχανική Κώδικα

Το τελικό στάδιο της πυραμίδας αποτελεί και το πιο απαιτητικό όσο αφορά στις δεξιότητες και γνώσεις που πρέπει να διαθέτει ο αναλυτής. Σε αυτό το στάδιο, ο αναλυτής φτάνει στο

επίπεδο να επανασυνθέτει τον κώδικα με τον οποίο φτιάχτηκε το κακόβουλο λογισμικό. Η επιτυχία αυτού του σταδίου εγγυάται την πλήρη κατανόηση του τρόπου λειτουργίας του κακόβουλου λογισμικού και την σίγουρη εύρεση κάποιων τρόπων προειδοποίησης και καταστολής της δράσης του. Η μέθοδος αυτή, προσφέρει ορισμένες πληροφορίες οι οποίες είναι αδύνατο να προσδιοριστούν με τα υπόλοιπα στάδια της πυραμίδας. Αυτές είναι:

- Αποκωδικοποίηση κρυπτογραφημένων δεδομένων αποθηκευμένων ή διακινούμενων από το υπό εξέταση δείγμα του κακόβουλου λογισμικού.
- Προσδιορισμός της λογικής λειτουργίας του αλγορίθμου παραγωγής *domain names,* τα οποία χρησιμοποιούνται ως σημεία σύνδεσης του κακόβουλου λογισμικού με απομακρυσμένους εξυπηρετητές που το ελέγχουν.
- Κατανόηση δυνατοτήτων του κακόβουλου λογισμικού που δεν έγιναν εμφανείς με την διενέργεια της συμπεριφορικής ανάλυσης δεδομένων.

Για την επιτέλεση την επανασύνθεσης του κώδικα από τα διαθέσιμα εργαλεία τα παρακάτω εργαλεία είναι απαραίτητα:

- Αποσυναρμολογητής.
- Αποσφαλματωτής.
- Απομεταγλωτιστής.
- Πλήθος *plugins* και εξιδικευμένα εργαλεία.
- Εργαλεία ιατροδικαστικής έρευνας μνήμης.

Η διαδικασία της αντίστροφης μηχανικής απαιτεί πολύ χρόνο αλλά προσφέρει την τελική και πιο εμπεριστατωμένη αποτύπωση της ύπαρξης της κακόβουλης οντότητας.

3.3 Εξαγωγή μέτρων προστασίας από Κακόβουλο Λογισμικό

Το στάδιο της ανάλυσης του κακόβουλου λογισμικού προσφέρει όλο το πλήθος των πληροφοριών, πάνω στις οποίες βασίζεται ο αναλυτής για να προχωρήσει στο τελευταίο και πιο σπουδαίο, από πρακτικής άποψης, στάδιο, αυτό της εξαγωγής μέτρων προστασίας για έναν συγκεκριμένο τύπο κακόβουλου λογισμικού. Κάνοντας λόγο για μέτρα προστασίας εννοούμε είτε την δημιουργία κάποιων αυτοματοποιημένων διαδικασιών εντοπισμού κακόβουλου λογισμικού και αφύπνισης των λειτουργιών άμυνας του συστήματος, είτε την

συγγραφή κανόνων προστασίας σε ένα λογισμικό ανίχνευσης κακόβουλων λογισμικών, όπως είναι το *Yara.*

Η συγγραφή αυτοματοποιημένων διαδικασιών εντοπισμού και ανάλυσης κακόβουλων λογισμικών δύναται να στηριχθεί στην λειτουργικότητα ήδη υπαρχόντων εργαλείων. Τα εργαλεία αυτά μπορεί να εκτελούν μία συγκεκριμένη εργασία, η ενοποίηση των οποίων μπορεί να προσομοιώνει την λειτουργία ενός εξειδικευμένου αντιικού ή συστήματος ανάλυσης. Για παράδειγμα, με την συγγραφή ενός σεναρίου σε γλώσσα προγραμματισμού *Python,* είναι δυνατόν να ενεργοποιηθούν σειριακά διάφορα εργαλεία όπως το *Dumpit και το volatillity* για την διενέργεια ελέγχων σε κάποιο αρχείο που θα δοθεί ως είσοδος. Η έξοδος του σεναρίου θα μπορεί να είναι η απάντηση στο ερώτημα, αν το αρχείο αποτελεί κακόβουλο λογισμικό ή όχι.

Παράλληλα, η συγγραφή κανόνων σε ένα εργαλείο όπως το *Yara,* δύναται να αποτελέσει έναν εξειδικευμένο τρόπο με τον οποίο μπορεί ο εκάστοτε αναλυτής να εντοπίζει κακόβουλο λογισμικό στο υπολογιστή του. Η μέθοδος αυτή απαιτεί από τον αναλυτή να υποβάλλει ένα λογισμικό προς εξέταση κακόβουλης ύπαρξης. Ωστόσο ο συνδυασμός με την συγγραφή αυτοματοποιημένων σεναρίων δύναται να παράγει μία αυτοματοποιημένη μορφή υποβολής και εντοπισμού κακόβουλου λογισμικού. Παράλληλα, ο χρήστης πλέον θα μπορεί μέσω μίας απλής διεπαφής να παρακολουθεί την διενέργεια της εν λόγω υποβολής κάποιων προκαθορισμένων φακέλων και να βλέπει τα αποτελέσματα.

Εν γένει, όλα τα μέτρα προστασίας που δεν αφορούν στην χρήση ενός αυτοματοποιημένου αντιικού λογισμικού από το εμπόριο, δεν απευθύνονται στον μέσο χρήστη. Αν εξαιρέσουμε τις προαναφερθείσες μεθόδους προστασίας, οι οποίες μπορεί να χρησιμοποιηθούν από κάποιο χρήστη στην τελική τους μορφή, η όλη διαδικασία της προφύλαξης και του εντοπισμού κακόβουλων πηγών στο σύστημα προκύπτει από τις γνώσεις που αποκτά ο αναλυτής στο στάδιο της ανάλυσης. Δηλαδή, η μελέτη της δομής ενός κακόβουλου λογισμικού και η προσεκτική παρατήρηση της συμπεριφοράς του, είναι μία διαδικασία που αποτέλεσμα της είναι η παραγωγή πληροφοριών και προτύπων συμπεριφοράς αυτού του λογισμικού. Η αξιοποίηση αυτών οδηγεί στον έγκαιρο εντοπισμό της παρουσίας παρόμοιων λογισμικών που θα προσβάλλουν το σύστημα και στην έγκαιρη αποσόβηση του κινδύνου που μπορεί να προκύψει από την δράση τους.

4. ΣΥΓΧΡΟΝΕΣ ΤΕΧΝΟΛΟΓΙΕΣ ΑΝΑΛΥΣΗΣ ΚΑΚΟΒΟΥΛΩΝ ΛΟΓΙΣΜΙΚΩΝ

Κατόπιν της περιγραφής της διαδικασίας μελέτης των κακόβουλων λογισμικών, θεωρήθηκε σκόπιμο να γίνει ειδική αναφορά σε κάποιες σύγχρονες τεχνολογίες και εργαλεία που αποτελούν ή πρόκειται να αποτελέσουν στο μέλλον σημεία αναφοράς στην διαδικασία της ανάλυσης των λογισμικών αυτών. Στο κεφάλαιο αυτό, λοιπόν, θα παρουσιαστούν ορισμένες τεχνολογίες τις οποίες χρησιμοποιήσαμε και εμείς στην διενέργεια ανάλυσης πάνω σε κακόβουλο λογισμικό, και διαπιστώθηκε αφενός ότι προσφέρουν αυξημένη διόραση στην κατανόηση του τρόπου λειτουργίας των κακόβουλων λογισμικών και αφετέρου ότι μπορούμε να χρησιμοποιήσουμε κάποιες ως μέσο εντοπισμού κακόβουλης οντότητας μέσω της αποκτημένης γνώσης.

Ορισμένες από τις τεχνολογίες που θα περιγραφούν βρίσκονται ακόμα σε αρκετά πρώιμο στάδιο ωρίμανσης και ανάπτυξης, παρόλο της δημοτικότητας που έχουν αποκτήσει. Παράλληλα, πολλά εργαλεία δεν έχουν αναπτυχθεί με τρόπο ώστε να μπορούν να χρησιμοποιηθούν στις καινούργιες εκδόσεις λειτουργικών συστημάτων (*Windows 8.1* και *Windows* 10), ενώ άλλα το προβλέπουν αλλά δεν έχουν δοκιμαστεί στην χρήση στα νέα λειτουργικά συστήματα. Για τον λόγο αυτό, η χρήση των τεχνολογιών ανάλυσης αποτέλεσε μία πρόκληση από μόνη της, καθώς θα έπρεπε πρώτα να εξακριβωθεί πειραματικά σε συνθήκες εργαστηρίου, ότι πληρούν τις απαραίτητες προϋποθέσεις για χρήση σε εκδόσεις νέων λειτουργικών και ότι παράγουν τα ίδια αξιόπιστα και ολοκληρωμένα αποτελέσματα όπως στις δοκιμασμένες εκδόσεις.

Ωστόσο, τα κακόβουλα λογισμικά σχεδιάζονται ώστε να προσβάλουν όλον ένα και πιο καινούργιες πλατφόρμες. Η τάση αυτή έχει την βάση της στο γεγονός ότι κάθε φορά που παρουσιάζονται νέες πλατφόρμες λειτουργικών συστημάτων οι τρύπες ασφαλείας είναι περισσότερες σε σχέση με ένα δοκιμασμένο και θωρακισμένο λειτουργικό, το οποίο έχει υποβληθεί σε πολλαπλές εκδόσεις *Patching* για την εξαφάνιση των κενών ασφαλείας. Επίσης, η μετάβαση ενός χρήστη απαιτεί εξοικείωση˙ η έλλειψη της οποίας καθιστά ανίκανο το χρήστη να μπορεί αποφεύγει κινήσεις που δυνητικά μπορούν να ανοίξουν την είσοδο σε κακόβουλο λογισμικό.

Είναι, λοιπόν, εύκολα αντιληπτό το κενό που δημιουργείται στην εξέλιξη των δύο αντίθετων πλευρών, των κακόβουλων λογισμικών και των εργαλείων μελέτης τους. Όπως αναφέρθηκε, η πιστοποίηση της ασφάλειας και της εγκυρότητας των χρησιμοποιούμενων εργαλείων αποτελεί τροχοπέδη στην γοργή ανάπτυξή τους. Αντίθετα, οι *Hackers* δεν ενδιαφέρονται μόνο να δημιουργήσουν ένα λογισμικό αόρατο και αλεξίσφαιρο, αλλά ένα λογισμικό που μέχρι να μελετηθεί να προκαλέσει συγκεκριμένη ζημιά. Άρα, η ανάπτυξη νέων κακόβουλων λογισμικών είναι εξαιρετικά γοργή με καινοτόμες ιδιότητες. Συμπέρασμα, όλων των παραπάνω είναι αιτιολόγηση της επιλογής μας να εμείνουμε στην δοκιμή εργαλείων σε σύγχρονα περιβάλλοντα ώστε να μπορέσουμε να μελετήσουμε νεόφερτα λογισμικά με καταστροφικές για το χρήστη δυνατότητες.

4.1 Volatility

Τα στάδια του εντοπισμού και της ανάλυσης κακόβουλων λογισμικών απαιτούν την διεξαγωγή έρευνας σε διάφορα τμήματα του συστήματος, όπως ο σκληρός δίσκος και η μνήμη *RAM*. Η μνήμη *RAM (Random Access Memory),* ωστόσο, αποτελεί τον χώρο μνήμης στον οποίο κάθε στιγμή υπάρχει αποτυπωμένη όλη η περίοδος λειτουργίας του υπολογιστή. Αυτό πρακτικά σημαίνει ότι όλη η δραστηριότητα ενός υπολογιστικού συστήματος κάθε δεδομένη στιγμή σχετίζεται με την μνήμη RAM και η προσεκτική μελέτη της αποκαλύπτει όλη αυτήν την δραστηριότητα. Η *RAM* είναι πτητική μνήμη, δηλαδή όταν η περίοδος του υπολογιστή τερματιστεί, όλα τα δεδομένα που είναι αποθηκευμένα στην μνήμη χάνονται. Οπότε κάθε φορά η μελέτη της αφορά σε συγκεκριμένη περίοδο λειτουργίας (μέσα στην οποία δρα και το κακόβουλο λογισμικό).

Στην περίπτωση, λοιπόν, που η μελέτη την μνήμης *RAM* πραγματοποιείται εντός μίας περιόδου λειτουργίας του υπολογιστή, παρουσιάζονται διάφορα προβλήματα και μειονεκτήματα. Αρχικά, είναι πιθανό ένα κακόβουλο λογισμικό να παρουσιάζει μία δράση για συγκεκριμένο χρονικό διάστημα και μετά να σιωπά. Επίσης, μπορεί να παρουσιάζει εναλλαγές στην συμπεριφορά του ανάλογα με τις κινήσεις του χρήστη. Ακόμη, ένα πρόβλημα είναι ότι δεν είναι δυνατόν για κάποιο χρονικό διάστημα να μελετάται η ίδια εικόνα της μνήμης καθώς αυτή αλλάζει καθώς ο υπολογιστής λειτουργεί. Σε κάποια ακραία περίπτωση μπορεί το κακόβουλο λογισμικό να προκαλεί κατάρρευση του συστήματος και αρά να καθίσταται αδύνατη η περαιτέρω μελέτη της αιτίας που την προκάλεσε. Επιπροσθέτως, ακόμα και αν εντοπιστεί ένα κακόβουλο λογισμικό, η παρακολούθηση της

RAM μετά από κάποιο χρονικό διάστημα μπορεί να παράγει διαστρεβλωμένα δεδομένα λόγω της δράσης του εν λόγω λογισμικού. Στην στοίβα των προβλημάτων, προστίθεται και η μη ύπαρξη δυνατότητας εξέτασης της εικόνας της *RAM* από κάποιο εργαλείο σε άλλο λειτουργικό το οποίο παρέχει περισσότερες δυνατότητες ανάλυσης αλλά και την ασφάλεια από την προσβολή από το συγκεκριμένο κακόβουλο λογισμικό καθώς ένα κακόβουλο λογισμικό που βλάπτει έναν υπολογιστή που έχει λειτουργικό *Windows* δεν δύναται να βλάψει έναν υπολογιστή που έχει λειτουργικό *Linux*.

Συνοψίζοντας, η μελέτη της *RAM* κατά την διάρκεια λειτουργίας του λειτουργικού συστήματος μπορεί να προσφέρει αμεσότητα και διαδραστικότητα, αλλά υστερεί σοβαρά σε μεροληψία, αφού η παρακολούθηση και η ανάλυση δεν μπορούν στιγμιαία να εισχωρήσουν εις βάθος. Επίσης, με την ανάπτυξη των σύγχρονων κακόβουλων λογισμικών είναι αρκετά πιθανό να υπάρχει σε κλαποιο από αυτά η τεχνολογία που τα αδρανοποιεί σε περίπτωση που αντιληφθούν ότι η δραστηριότητα τους παρακολουθείται. Η λύση σε όλα τα προαναφερθέντα προβλήματα ακούει στο όνομα *Ανάλυση Στιγμιότυπου Μνήμης.* Το στιγμιότυπο μνήμης είναι μία δυνατότητα που παρέχεται σε όλα τα λειτουργικά, είτε ως ενσωματωμένη λειτουργία του λειτουργικού συστήματος (Λήψη Στιγμιότυπου Μνήμης στα Windows), είτε μέσω κάποιου λογισμικού τρίτου φορέα (dumpit.exe στα *Windows*) και άφορα στην ακριβή αποτύπωση της μνήμης *RAM* σε ένα δυαδικό αρχείο για σκοπό τόσο αποσφαλμάτωσης αστοχιών του συστήματος όσο και μελέτης κακόβουλων λογισμικών. Η τεχνική της Ανάλυση Στιγμιότυπου Μνήμης εμπεριέχει 3 βασικά βήματα:

1. Επιλογή της κατάλληλης στιγμής λήψης του στιγμιοτύπων της μνήμης. Η επιλογή αφορά στην χρονική στιγμή και το τρόπο μελέτης που θέλουμε να αναπτύξουμε. Για παράδειγμα, σε περίπτωση που απλά θέλουμε να αποτυπώσουμε μία τυχαία στιγμή για τον έλεγχο προσβολής από κάποια απειλή, δεν χρειάζεται να κάνουμε κάτι ιδιαίτερο. Αντίθετα, στην περίπτωση όπου έχουμε ένα κακόβουλο λογισμικό και θέλουμε αμερόληπτα να καταγράψουμε τις αλλαγές που προκαλεί στο σύστημα ή σε άλλα λογισμικά προγράμματα πρέπει να επιλέξουμε προσεκτικά μία στιγμή πριν την προσβολή ή όταν αυτό είναι αδρανές και μία αφού η δραστηριότητα του βρίσκεται σε έξαρση, με σκοπό να μελετήσουμε την διαφορά στα στιγμιότυπα.
2. Λήψη του στιγμιότυπου με ένα από τους δύο τρόπος που αναφέρθηκαν παραπάνω.
3. Χρήση του στιγμιότυπου για ανάλυση με το εργαλείο Volatility.

Το *Volatility είναι* ένα ενιαίο πλαίσιο ανάλυσης στιγμιοτύπων μνήμης από λειτουργικά συστήματα *Windows, Linux, Mac OS, Android* με υποστηριζόμενες 32-bit και 64-bit αρχιτεκτονικές. Η ανάλυση του στιγμιότυπου έχει τις ίδιες και ίσως και περισσότερες

δυνατότητες από την ανάλυση σε μία εν λειτουργία μνήμη *RAM* [17]. Το *volatility* παρέχει πληθώρα πλεονεκτημάτων σε σχέση με παρόμοια εργαλεία και τεχνοτροπίες ανάλυσης των συστημάτων για την καταπολέμηση των κακόβουλων λογισμικών, όπως αυτές που φαίνονται στην συνέχεια:

- Ο σχεδιασμός του *Volatility* είναι τέτοιος που επιτρέπει την ενσωμάτωση και την ανάλυση νέων λειτουργικών συστημάτων και αρχιτεκτονικών, μόλις αυτά βγουν στην αγορά. Συνεπώς, το συγκεκριμένο εργαλείο δεν υπόκεινται σε περιορισμούς όσον άφορα στην συμβάδισμα με τα κακόβουλα λογισμικά που σχεδιάζονται για νέα λειτουργικά συστήματα.
- Είναι ανοιχτού κώδικα άδειας GPLv2, το οποίο σημαίνει ότι ο αναλυτής έχει την δυνατότητα και την άδεια να διαβάσει τον πηγαίο κώδικα, να μάθει πληροφορίες από αυτόν σχετικά με τον τρόπο που εξάγονται τα αποτελέσματα και να τον προεκτείνει. Με τον τρόπο αυτό ο αναλυτής έχει την δυνατότητα να γνωρίσει εις βάθος το εργαλείο που χρησιμοποιεί και να το αναπτύξει περαιτέρω ώστε να καλύπτει με μεγαλύτερη επάρκεια τις ανάγκες της ανάλυσης που θέλει να διεξάγει. Παράλληλα μπορεί ο εκάστοτε προγραμματιστής – αναλυτής να διορθώνει προβλήματα του εργαλείου, δίχως να περιμένει απάντηση από τον φορέα που το ανέπτυξε και το υποστηρίζει.
- Είναι γραμμένο με την γλώσσα προγραμματισμού *Python*, μία γλώσσα διεθνώς καθιερωμένη για εφαρμογές μελέτης λογισμικών και αντίστροφης μηχανικής, η οποία παρέχει πληθώρα βιβλιοθηκών που μπορούν να ενσωματωθούν στο εργαλείο και να εκτοξεύσουν την επεκτασιμότητα και την λειτουργικότητα του. Επίσης η σύνταξη των εντολών ανάλυσης που παρέχει το εργαλείο έχει άμεση συνάφεια με την σύνταξη σε γλώσσα *Python* με αποτέλεσμα να δημιουργείται κλίμα οικειότητας σε ένα αναλυτή που γνωρίζει ήδη αυτή την γλώσσα και δεν θα χρειαστεί να εξοικειωθεί και με κάποια άλλη ιδιαίτερη σύνταξη όπως συμβαίνει με άλλα εργαλεία (*windbg*).
- Εγκαθίσταται και τρέχει σε λειτουργικά συστήματα *Windows*, *Linux*, *Mac OS*, όπου δηλαδή μπορεί να τρέξει και η *Python*. Το συγκεκριμένο χαρακτηριστικό είναι πολύ σημαντικό και διαφοροποιεί το εργαλεία από άλλα παρόμοια τα οποία τρέχουν μόνο σε λειτουργικό Windows. Αυτό πρακτικά σημαίνει πώς ο αναλυτής μπορεί να χρησιμοποιήσει το *Volatility* σε οποιαδήποτε σύστημα έχει οικειότητα και θέλει να προσθέσει ένα νέο τρόπο ανάλυσης σε αυτούς που ήδη γνωρίζει.
- Το *Volatility* συνοδεύεται από ένα επεκτάσιμο API, το οποίο διευρύνει τις προγραμματιστικές δυνατότητες του αναλυτή για την ανάπτυξη αυτοματοποιημένων και φιλικών προς τη χρήση μηχανισμών. Για παράδειγμα, μπορεί να δημιουργήσει:
    - Μία διαδικτυακή διεπαφή ή μία γραφική διεπαφή χρήστη.
    - Αυτοματοποιημένη λειτουργικότητα ενός *sandbox κακόβουλου λογισμικού.*

    - Αυτοματοποίηση της μελέτης της μνήμης.
- Ένα σύνολο από λειτουργίες που ούτε ο αποσφαλματωτής της Microsoft δεν παρέχει και που συμβάλουν τα μάλα στην εξειδικευμένη ανάλυση και την διενέργεια αντίστροφης μηχανικής.
- Συμβατότητα ανάλυσης ενός μεγάλου πλήθους τύπων αρχείων. Το *Volatility* είναι συμβατό με:
    - *RAW Dumps (*στιγμιότυπα*)*
    - *Crash Dumps*
    - αρχεία *Hibernation*
    - *Vmware .vmem,* .vmss/.vmsn
    - VirtualBox core dumps
    - *Linux Memory Extractor*
    - *EWF*
    - Φυσική μνήμη μέσω FireWire
- Υλοποιημένο με γρήγορους και αποδοτικούς αλγορίθμους για πραγματοποίηση ταχείας ανάλυσης σε στιγμιότυπα μεγάλης μνήμης συστημάτων.

Το εργαλείο αυτό προσφέρει μέσω μία πληθώρας εντολών την δυνατότητα της βαθιάς εισχώρησης στην πτητική μνήμη του υπολογιστή και την διενέργεια εντοπισμού κακόβουλης δραστηριότητας. Αφού γίνει ο εντοπισμός, το εργαλείο χρησιμοποιείται για την διερεύνηση των τμημάτων του υπολογιστή που έχουν υποστεί κακόβουλη παρέμβαση και τι είδους είναι αυτή. Σε τελικό στάδιο, ο αναλυτής μπορεί να συγκεντρώσει τόσες πληροφορίες που να ταυτοποιεί απόλυτα το κακόβουλο λογισμικό. Η ταυτοποίηση ακολουθείται από την ένταξη του λογισμικού σε μία από τις γνωστές κατηγορίες κακόβουλων λογισμικών και την εξαγωγή μοτίβων συμπεριφοράς του λογισμικού μέσω προσεκτικής παρατήρησης των κινήσεων του που φωτογραφίζονται στο στιγμιότυπο της μνήμης RAM.

4.2 Yara

Όταν ένα κακόβουλο λογισμικό μελετηθεί προκύπτουν διάφορα μοτίβα που το αφορούν και δύναται να το ξεχωρίσουν από άλλα λογισμικά. Αυτά τα μοτίβα είναι διαφόρων ειδών και προσανατολισμών. Κάποιο μοτίβο μπορεί να αφορά στην συμπεριφορά του κακόβουλου λογισμικού, π.χ. την διαδικτυακή δραστηριότητα που εμφανίζει όταν προσπαθεί να συνδεθεί με ένα απομακρυσμένο *domain name.* Άλλα μοτίβα αφορούν στην ίδια την δομική του

σύσταση και τον τρόπο που έχει κατασκευαστεί. Ένα τέτοιο μοτίβο είναι η αλληλουχία των αλφαριθμητικών ή των δυαδικών στοιχείων που εμφανίζονται στο εκτελέσιμο δυαδικό αρχείο. Κατόπιν αποτύπωσης των χαρακτήρων σε κωδικοποίηση *ASCII* ή *Unicode* που εμφανίζονται σε ένα αρχείο μέσω διαφόρων εργαλείων (*strings.exe, radare2* κ.λ.π) ο αναλυτής διενεργεί μελέτη για την ύπαρξη αλληλουχίας αλφαριθμητικών σε κανονική ή δεκαεξαδική μορφή που υποδεικνύουν την διενέργεια κακόβουλης δραστηριότητας. Τέτοια μπορεί να είναι:

- Ονόματα συναρτήσεων που χρησιμοποιούνται συχνά από κακόβουλα λογισμικά, όπως οι συναρτήσεις αποσφαλμάτωσης (π.χ *WriteProcessMemory*) ή άλλων συναρτήσεων με ύποπτα ονόματα, όπως π.χ *InjectHack.*
- Ονόματα διευθύνσεων *URL* τα οποία δεν είναι συνηθισμένα ή δεν έχει ποτέ επισκεφτεί ο χρήστης, όπως π.Χ. *www.hackertick.com.*
- Τα αρχικά γράμματα αναγνώρισης ενός *DOS* εκτελέσιμου αρχείου *PE, 4D 5A* σε δεκαεξαδική μορφή, μέσα σε ένα αρχείο που δεν είναι εκτελέσιμο.
- Κάποιες προειδοποιήσεις ασφάλειας του *.NET* που προφανώς θα προέκυψαν κατά την εκτέλεση του κακόβουλου και μη διαπιστευμένου αρχείου (π.χ *SkipVerification RSDSZ*).
- Εμφάνιση βιβλιοθηκών που δεν είναι κάποιες από τις γνωστές και μπορεί να είναι αποτέλεσμα της τεχνικής *DLL Injection.*

Όλα τα παραπάνω αλφαριθμητικά αποτελούν ενδείξεις προσβολής ή *IOC* και η εξέταση ενός αρχείου ή μίας διεργασίας για τον αν τα περιέχει μπορεί να οδηγήσει στον εντοπισμό πιθανού κακόβουλου λογισμικού το οποίο πρέπει να εξεταστεί περαιτέρω.

```
rule Intruder_Malware: injection
{
    meta:
        description = "Rule set to identify the non-existant Intruder_Malware malware"
        malware_severity = "Critical"

    strings:
        $suspicious_1 = "CreateRuntimeHack"
        $suspicious_2 = "WriteProcessMemory"
        $suspicious_3 = {0A 45 AE F4 3A F7 F9 93 22 C1}
        $suspicious_4 = "DMVIVMOENVLSDNVLSNVSLDKVLSDKV32CSLC"

    condition:
        ($suspicious_1 and $suspicious_2 and $suspicious_3) or $suspicious_4
}
```

Εικόνα 5: Παράδειγμα κανόνα Yara

Ο ρόλος του *Yara είναι* να εξετάζει αρχεία και διεργασίες για το αν περιέχουν αλφαριθμητικά ή ακολουθίες από *Bytes* που υποδηλώνουν ύποπτη ύπαρξη. Η εξέταση αυτή γίνεται με την μορφή κανόνων που γράφονται από τον αναλυτή (Εικόνα 5). [18]

Στην παραπάνω εικόνα φαίνεται ένα παράδειγμα ενός τυπικού *Yara* κανόνα για την ανίχνευση της ύπαρξης τριών αλφαριθμητικών και μίας δεκαεξαδικής αλληλουχίας που φαίνονται στον τομέα *strings*. Στον τομέα *meta* είναι καταχωρημένες κάποιες πληροφορίες για τον κανόνα. Στον τομέα *condition* είναι γραμμένη μία συνθήκη η οποία περιγράφει με ποιο τρόπο και πόσα από τα στοιχεία του τομέα *strings* πρέπει να βρεθούν στο εξεταζόμενο αρχείο για να ισχύει ο κανόνας. Τέλος η περιγραφή *injection* στην αρχή του κανόνα αποτελεί ένα *Tag* το οποίο χρησιμεύει στο φιλτράρισμα των κανόνων που γράφονται για μία έρευνα του *Yara* σύμφωνα με το προσδιοριστικό που έχουμε θέσει.

Ωστόσο η λειτουργικότητα του *Yara* δεν περιορίζεται στην εύρεση στατικών αλφαριθμητικών. Οι επιπλέον δυνατότητες του συνοψίζονται ως εξής:

- Χρήση *Regular Expressions,* π.χ. /md5: [0-9a-zA-Z]{32}/
- Χρήση χαρακτήρων μπαλαντέρ, π.χ. ??
- Συνθήκες ελέγχου με αρκετές επιλογές, όπως έλεγχος μεγέθους αρχείων, έλεγχος αλληλουχιών σε συγκεκριμένες θέσεις στην εικονική μνήμη, επανάληψη στον έλεγχο
- Αναφορά σε άλλο κανόνα.
- Χρήση κάποιων *Modules* για συγγραφή ποιο εξειδικευμένων, αποτελεσματικών και πολύπλοκων κανόνων. Τα υποστηριζόμενα μέχρι αυτή την στιγμή *Modules* είναι: *PE, ELF, Cuckoo, Magic, Hash, Math.*
- Ύπαρξη βιβλιοθήκης *Python* για την συγγραφή προγραμμάτων με την λειτουργικότιτα του *Yara.*
- Ύπαρξη ενός *C API* για την συγγραφή προγραμμάτων με την λειτουργικότητα του *Yara.*

Όσον αφορά την συγγραφή σωστών κανόνων στο *Yara,* αυτοί πρέπει να διέπονται από τα εξής δύο στοιχεία [19]:

- Στατικότητα. Η στατικότητα στην μελέτη δειγμάτων ενός κακόβουλου λογισμικού αφορά στην εύρεση στοιχείων τα οποία παρουσιάζονται όμοια σε όλα τα δείγματα του ίδιου λογισμικού. Ειδάλλως, αν τα στοιχεία που ερευνώνται δεν είναι στατικά, δεν θα είναι δυνατόν να εντοπιστεί το κακόβουλο λογισμικό όλες τις φορές, αλλά ανά περίπτωση.

- Μοναδικότητα. Η μοναδικότητα αφορά στα κριτήρια της έρευνας τα οποία πρέπει να μην αποτελούν κοινό τόπο και για άλλου είδους λογισμικά. Εάν ένα στοιχείο προς εξέταση (π.χ. μία αλληλουχία δεκαεξαδικών αριθμών) αντιπροσωπεύει την σύνδεση του κακόβουλου λογισμικού με μία νόμιμη βιβλιοθήκη, όπως η *System32.dll,* τότε και μία σύνδεση ενός νόμιμου λογισμικού με την ίδια βιβλιοθήκη πιθανότατα θα περιέχει την ίδια ακολουθία. Συνέπεια αυτός θα είναι ένας ψεύτικος συναγερμός, καθώς ένα συνηθισμένο λογισμικό θα αναγνωριστεί ως κακόβουλο. Ο βαθμός αποφυγής έγερσης ψεύτικων συναγερμών από ένα κανόνα *Yara* αποτελεί και το βαθμό αξιοπιστίας του ίδιου κανόνα.

Η χρήση του *Yara* είναι πολύπλευρη. Μερικά παραδείγματα αποτελούν:

- Πολλά ερευνητικά εργαλεία έχουν ενσωματώσει το εργαλείο αυτό για την επίτευξη προχωρημένης ανάλυσης κακόβουλων λογισμικών.
- Το *Virus Total Intelligence* προσφέρει την δυνατότητα στους χρήστες να ανεβάζουν οι ίδιοι δικούς τους κανόνες *Yara* και να υποβάλλουν δείγματα τα οποία ελέγχονται από αυτούς τους κανόνες. Η διαδικασία αυτή ονομάζεται '*Hunting*'.
- Το εργαλείο *Cuckoo SandBox* μπορεί να ενσωματώσει *Yara* κανόνες στην λειτουργικότητα του και να παρουσιάσει τα αποτελέσματα στην αναφορά που εξάγει κατόπιν υποβολής του κακόβουλου δείγματος.

Μειονεκτήματα στην χρήση του Yara

Τα βασικά μειονεκτήματα του *Yara* αφορούν και ολόκληρη την κατηγορία στην οποία αυτό εμπίπτει ως τύπος εργαλείου. Όπως ειπώθηκε τα *Yara* είναι ένα εργαλείο που προσφέρει δυνατότητες ανίχνευσης λογισμικών σύμφωνα με αλφαριθμητικούς κανόνες. Η προστασία βασισμένη στην ανίχνευση «ταυτοτήτων» λογισμικών δεν είναι αρκετή σε πολλές περιπτώσεις. Το βασικό της μειονέκτημα είναι ότι η ταχεία ανάπτυξη των κακόβουλων λογισμικών τα καθιστά ικανά να αποφεύγουν και να «παραπλανούν» το *Yara* με την χρήση διαφόρων τεχνικών όπως η κρυπτογραφία, το «πακετάρισμα» και ο πολυμορφισμός. Σε χρονικά ακόλουθο στάδιο, η μελέτη αυτών των δυνατοτήτων των κακόβουλων λογισμικών συντελεί στην δημιουργία κανόνων ικανών να τα ανιχνεύσουν. Ωστόσο, στο μεσοδιάστημα μεταξύ εμφάνισης ενός κακόβουλου λογισμικού και ανίχνευσης του από τους τροποποιημένους κανόνες ταυτοποίησης του *Yara,* τα κακόβουλα λογισμικά διαφεύγουν της προσοχής και δημιουργούν προβλήματα.

Το παραπάνω μειονέκτημα δεν καθιστά τέτοιου είδους εργαλεία παρωχημένα. Αντίθετα η χρήση του διαδικτύου για την κατασκευή ενός δικτύου πληροφόρησης μεταξύ των αναλυτών για την έγκαιρη γνωστοποίηση των *IOCs* δύναται να ελαχιστοποιήσει τον εν λόγω χρόνο απόκρισης σε τέτοιο βαθμό που να θέτει εργαλεία σαν το *Yara* σε πρώτη τάξης άμυνα απέναντι σε κακόβουλες ενέργειες.

Τέλος, αξίζει να αναφερθεί πώς μέχρι στιγμής δεν έχει αναφερθεί δημόσια κάποιο κακόβουλο λογισμικό που να μπορεί να ανιχνεύει την δράση του *Yara* και να εμπίπτει σε κάποια κατάσταση στην οποία το εργαλείο δεν θα μπορεί να παράγει ορθά αποτελέσματα. Έτσι, το εργαλείο έχει ντετερμινιστική δράση και η χρήση ορθών κανόνων θα αποφέρει σίγουρα αποτελέσματα.

4.3 yarGen

Το *yarGen* είναι ένα εργαλείο με άμεση συνάφεια με το εργαλείο *Yara.* Το *yarGen* είναι μία αυτοματοποιημένη γεννήτρια κανόνων *Yara.* Η λειτουργία του εργαλείου αφορά στην παραγωγή κανόνων *Yara* μέσω της σύγκρισης των αλφαριθμητικών που βρίσκονται μέσα σε αρχεία κακόβουλων λογισμικών, με αλφαριθμητικά, που έχουν συλλεχθεί και αποθηκευτεί σε μία Βάση Δεδομένων, και προέρχονται από μία μεγάλη συλλογή καλόβουλων λογισμικών. Άρα αυτό που ουσιαστικά κάνει το εργαλείο είναι να συγκρίνει τα στοιχεία των κακόβουλων αρχείων που εξετάζονται με άλλα καλόβουλων λογισμικών και να εξάγει κανόνες *Yara* για τον εντοπισμό των κακόβουλων αρχείων. Παράλληλα, δίνεται η δυνατότητα να δημιουργήσει ο εκάστοτε αναλυτής την δική του Βάση Δεδομένων καλόβουλων λογισμικών και να την χρησιμοποιεί ως σημείο αναφοράς για σύγκριση με κακόβουλα αρχεία.

Το αποτέλεσμα του εργαλείου είναι κανόνες με αλφαριθμητικά που δεν υπάρχουν στην βάση δεδομένων που έχει χρησιμοποιηθεί ως σημείο αναφοράς. Ωστόσο αυτό δεν αποδεικνύει αυθωρεί ούτε την ορθότητα των κανόνων, ούτε την αποφυγή λανθασμένων συναγερμών. Πιο συγκεκριμένα, παρουσιάζονται οι παρακάτω λόγοι αποτυχίας των αυτοματοποιημένων κανόνων:

- Μη επαρκής δημιουργία Βάσης Δεδομένων με «καλόβουλα» αλφαριθμητικά, με αποτέλεσμα την στοχοποίηση καλόβουλων αλφαριθμητικών από τον κανόνα, που

απλά δεν έχουν καταγραφεί.

- Ενσωμάτωση αλφαριθμητικών στον κανόνα που αφενός δεν αποτελούν μέρος της καλόβουλης Βάσης Δεδομένων, αφετέρου δεν βγάζουν κάποια νόημα και είναι απλά τυχαία παραγόμενα.
- Ενσωμάτωση τέτοιων αλφαριθμητικών στον κανόνα που τον καθιστούν ικανό να ανιχνεύσει μόνο το συγκεκριμένο κακόβουλο λογισμικό, αλλά κανένα άλλο της ίδιας οικογένειας.

Το εργαλείο, λοιπόν, αυτό χρησιμεύει κατά κύριο λόγο επικουρικά στην δημιουργία σωστών και στοχευμένων κανόνων *Yara.* Η χρήση του μπορεί να βοηθήσει τον αναλυτή να καθορίσει εύκολα και γρήγορα τα αλφαριθμητικά τα οποία θα πρέπει να εξετάσει εις βάθος για να αποφανθεί εάν και πως θα πρέπει να ενταχθούν σε ένα κανόνα.

```
################################################################################
   _____
  / ___/ ____ ____ ____ ____ ____
 / /__ / __ `/ ___/ __ `/ _ \/ __ \
/ /_/ / /_/ / /  / /_/ /  __/ / / /
\____/\__,_/_/   \__, /\___/_/ /_/
                /____/

   Yara Rule Generator
   by Florian Roth
   July 2015
   Version 0.14.0

################################################################################
[+] Reading goodware strings from database 'good-strings.db' ...
    (This could take some time and uses up to 2 GB of RAM)
[+] Initializing Bayes Filter ...
[-] Training filter with good strings from ./lib/good.txt
[+] Processing malware files ...
[-] Processing: /Volumes/Work/MAL/HackingTeam/bin/backdoor.exe
[-] Processing: /Volumes/Work/MAL/HackingTeam/bin/dropper.exe
[-] Processing: /Volumes/Work/MAL/HackingTeam/bin/install.m.apk
[-] Processing: /Volumes/Work/MAL/HackingTeam/bin/ndisk.sys
[-] Processing: /Volumes/Work/MAL/HackingTeam/bin/putty.exe
[-] Processing: /Volumes/Work/MAL/HackingTeam/bin/rcs.exe
[+] Generating statistical data ...
[+] Generating Super Rules ... (a lot of foo magic)
[E] ERROR while generating general condition - check the global rule and remove it if it's faulty
[+] Generating simple rules ...
[-] Applying intelligent filters to string findings ...
[-] Filtering string set for /Volumes/Work/MAL/HackingTeam/bin/rcs.exe ...
[-] Filtering string set for /Volumes/Work/MAL/HackingTeam/bin/putty.exe ...
[-] Filtering string set for /Volumes/Work/MAL/HackingTeam/bin/dropper.exe ...
[-] Filtering string set for /Volumes/Work/MAL/HackingTeam/bin/install.m.apk ...
[-] Filtering string set for /Volumes/Work/MAL/HackingTeam/bin/backdoor.exe ...
[-] Filtering string set for /Volumes/Work/MAL/HackingTeam/bin/ndisk.sys ...
[+] Generating super rules ...
[=] Generated 6 SIMPLE rules.
[=] Generated 0 SUPER rules.
[=] All rules written to yargen_rules.yar
prometheus:yarGen neo$
```

Εικόνα 6: Εργαλείο Yara Rule Generator

Το εργαλείο αυτό συγκρίνει τα αλφαριθμητικά του αρχείου που εξετάζει και επιλέγει αυτά που δεν υπάρχουν στην Βάση Αναφοράς. Κατόπιν, χρησιμοποιεί τον αλγόριθμο κατηγοριοποίησης *naïve-bayes* από τους *Mustafa Atik* και *Nejdet Yucesoy* για την κατηγοριοποίηση των αλφαριθμητικών και τον εντοπισμό των αλφαριθμητικών που συνθέτουν λέξεις υπαρκτές και χρήσιμες αντί λέξεων – «σκουπίδια» αποτέλεσμα κωδικοποίησης και συμπίεσης. Κάθε αλφαριθμητικό αντιστοιχεί με ένα αριθμό σκορ που αφορά στο βαθμό «επικινδυνότητας» ή βαθμό συσχέτισης με κακόβουλα λογισμικά και εν τέλει επιλέγει τα είκοσι πρώτα αλφαριθμητικά με το μεγαλύτερο σκόρ για την σύνθεση του εκάστοτε κανόνα που αντιστοιχεί στο εξεταζόμενο αρχείο. [20]

4.4 Cuckoo SandBox

Όταν ένα σύστημα έχει προσβληθεί από ένα κακόβουλο λογισμικό, ο εντοπισμός και η ανάλυση του, σε πρώτη φάση, είναι αναπόφευκτη καθώς το σύστημα θα πρέπει να εξυγιανθεί. Συνεπώς, παρά τον κίνδυνο που διατρέχει το σύστημα από την δράση του λογισμικού - στόχου, η ανάλυση πρέπει να διενεργηθεί για να δώσει τις απαραίτητες πληροφορίες για την απομάκρυνση της απειλής. Σε δεύτερη φάση, όμως, όταν το λογισμικό θα πρέπει να αναλυθεί εις βάθος και να εξαχθούν έγκυρα πρότυπα συμπεριφοράς, η μελέτη του τρόπου και του “τόπου” διενέργειας της ανάλυσης κατέχει πρωτεύουσα σημασία. Στην περίπτωση αυτή, ο αναλυτής έχει ένα δείγμα κακόβουλου λογισμικού, ωστόσο το σύστημα που θα γίνει η ανάλυση είναι και θα πρέπει να είναι καθαρό και αμόλυντο από κακόβουλες παρουσίες. Συνεπώς, δεν είναι δυνατόν ο αναλυτής να υποβάλλει το πραγματικό σύστημά του στην πλήρη και απρόσκοπτη δράση του κακόβουλου λογισμικού για να παρατηρήσει την συμπεριφορά του, καθώς αυτό αφενός μπορεί να προκαλούσε σοβαρά προβλήματα στο σύστημα και αφετέρου μπορεί να νόθευε τα αποτελέσματα της ανάλυσης. [21]

Το *sandboxing* αποτελεί μία τεχνική που προσπερνά το παραπάνω πρόβλημα και δίνει λύση στο γρίφο της ανάλυσης ενός κακόβουλου λογισμικού, η δράση του οποίου όμως θα πρέπει να είναι ανεμπόδιστη ως σαν να μην παρακολουθούταν. Το *sandbox ή Αμμοδόχος* είναι ένας μηχανισμός ασφαλείας για τον διαχωρισμό τρεχόντων προγραμμάτων. Η χρήση του αφορά στην εκτέλεση λογισμικών που είτε δεν έχουν διαπιστευθεί για την σταθερότητα της χρήσης τους και πρέπει να περάσουν από κάποια στάδιο εξέτασης, είτε αποτελούν πιθανή απειλή και πρέπει να παρακολουθηθούν. Πιο συγκεκριμένα, ο μηχανισμός δημιουργεί ένα πλαίσιο

υπολογιστικών πόρων, στενά ελεγχόμενο και προκαθορισμένο, πάνω στο οποίο τρέχει ένα λειτουργικό σύστημα. Οι πόροι αφορούν κυρίως στην μνήμη, στον χώρο στον δίσκο και στην πρόσβαση στο διαδίκτυο. Ο κατάλληλος περιορισμός και έλεγχος αυτών προσφέρει το πλεονέκτημα του περιορισμού της πρόσβασης ενός τρέχοντος στο *sandbox* κακόβουλου λογισμικού στο φυσικό λειτουργικό σύστημα. Παράλληλα, όμως, αυτό δρα ανενόχλητο μέσα στο πλαίσιο του *sandbox* και μπορεί να εκδηλώσει ανενόχλητο την δράση του. Το *sandboxing,* λοιπόν μπορεί να θεωρηθεί ως μία τεχνική εικονοποίησης (*virtualization*) καθώς δίνει την δυνατότητα στον αναλυτή να εισάγει σε ένα εικονικό λειτουργικό περιβάλλον ένα κακόβουλο λογισμικό και να το παρακολουθεί, χωρίς αυτό να επηρεάζει το φυσικό του σύστημα.

Η επιλογή της διενέργειας της ανάλυσης σε ένα εικονικό περιβάλλον προκαθορισμένων ορίων και δικαιωμάτων, που προσομοιώνει όμως επακριβώς το περιβάλλον λειτουργίας χρήστη, είναι απαραίτητο στάδιο στην ασφαλή και ενδελεχής μελέτη του κακόβουλου λογισμικού. Ωστόσο, υπάρχουν και κάποια μειονεκτήματα. Βασικό μειονέκτημα αποτελεί η ύπαρξη ορισμένων στοιχείων που αποδεικνύουν ότι το λειτουργικό που τρέχει είναι εγκλωβισμένο σε μία εικονική μηχανή, γεγονός που εκμεταλλεύονται μερικά σύγχρονα και αρκετά εξελιγμένα κακόβουλα λογισμικά για να αναγνωρίσουν μία τέτοια κατάσταση και να καταστείλουν την λειτουργία τους για να μην αποκαλυφθεί η συμπεριφορά τους. Επίσης, πολλά από τα εργαλεία *sandbox* που υπάρχουν αργούν να παρέχουν συμβατότητα με νέες εκδόσεις λειτουργικών, καθιστώντας την εικόνοποίηση περιορισμένη σε παλιότερες εκδόσεις. Λύση στο τελευταίο πρόβλημα δίνει το *Cuckoo sandbox* το οποίο παρέχει αρκετά καλή συμβασιμότητα μέχρι και με *Windows 10.* [22]

Το *Cuckoo sandbox* είναι ένα λογισμικό ανοιχτού κώδικα για την επίτευξη αυτοματοποιημένης ανάλυσης ύποπτων αρχείων. Η λειτουργία του αφορά στην χρήση τροποποιημένων τμημάτων εκτέλεσης εικονικού λειτουργικού συστήματος για την παρακολούθηση διεργασιών που εκτελούνται σε ένα απομονωμένο εικονικό περιβάλλον. Η λειτουργικότητα του επεκτείνεται πέραν της απλής εκτέλεσης κακόβουλων λογισμικών σε απομονωμένο περιβάλλον και της παρακολούθησης αυτών, καθώς στο τέλος της εξέτασης της δραστηριότητάς αυτών εξάγεται μία πλήρης αναφορά με λεπτομερή στοιχεία για αυτά, υπό το πρίσμα διαφορετικών οπτικών γωνιών.

Καταρχάς, η επιλογή της διενέργειας αυτοματοποιημένης ανάλυσης κακόβουλων λογισμικών πρέπει να συνοδεύεται από την γνώση για τα μειονεκτήματα που εμπεριέχονται. Η επιτυχία της μεθόδου αυτής δεν αποτελεί ντετερμινιστική διαδικασία και εξαρτάται από

αμέτρητους παράγοντες. Ο αναλυτής στην αρχή θα πρέπει να αποφασίσει κάποια βασικά σημεία:

- Λειτουργικό σύστημα εικονικού περιβάλλοντος για την διενέργεια της εξέτασης.
- Ποια προγράμματα είναι απαραίτητα για την ανάλυση στο εικονικό περιβάλλον.

Κατόπιν θα πρέπει να παραμετροποίησει το εικονικό σύστημα με τέτοιο τρόπο ώστε αφενός να καλύπτονται οι προδιαγραφές της ανάλυσης και αφετέρου να προσομοιώνεται με τον ποιο ακριβή τρόπο η λειτουργία ενός φυσικού συστήματος για την εξαπάτηση του κακόβουλου λογισμικού. Η παραμετροποίηση αυτή αφορά:

- Επιλογή του είδους του αρχείου που θα υποβληθεί σε εξέταση.
- Επιλογή κατάλληλων ρυθμίσεων του εικονικού λειτουργικού περιβάλλοντος.
- Απόκρυψη όσον το δυνατόν περισσότερων στοιχείων ύπαρξης εικονικής μηχανής.
- Δημιουργία στοιχείων συνηθισμένης χρήσης για την εξαπάτηση του κακόβουλου λογισμικού. Τέτοια είναι τα εικόνες, βίντεο, αρχεία, ιστορικό περιήγησης, αγαπημένες σελίδες και *cookies.* Έτσι το κακόβουλο λογισμικό θα νομίζει ότι βρίσκεται σε ένα κανονικό περιβάλλον ενός ανυποψίαστου χρήστη και όχι σε ένα οχυρωμένο κουτί – παγίδα.

Το *Cuckoo sandbox* έχει σχεδιαστεί ώστε είτε να εκτελείται ανεξάρτητο, είτε να ενσωματώνεται σε μεγαλύτερα περιβάλλοντα. Η λειτουργικότητα του συγκεκριμένα συνοψίζεται στην παρακολούθηση και ανάλυση των παρακάτω ειδών ύποπτης ύπαρξης:

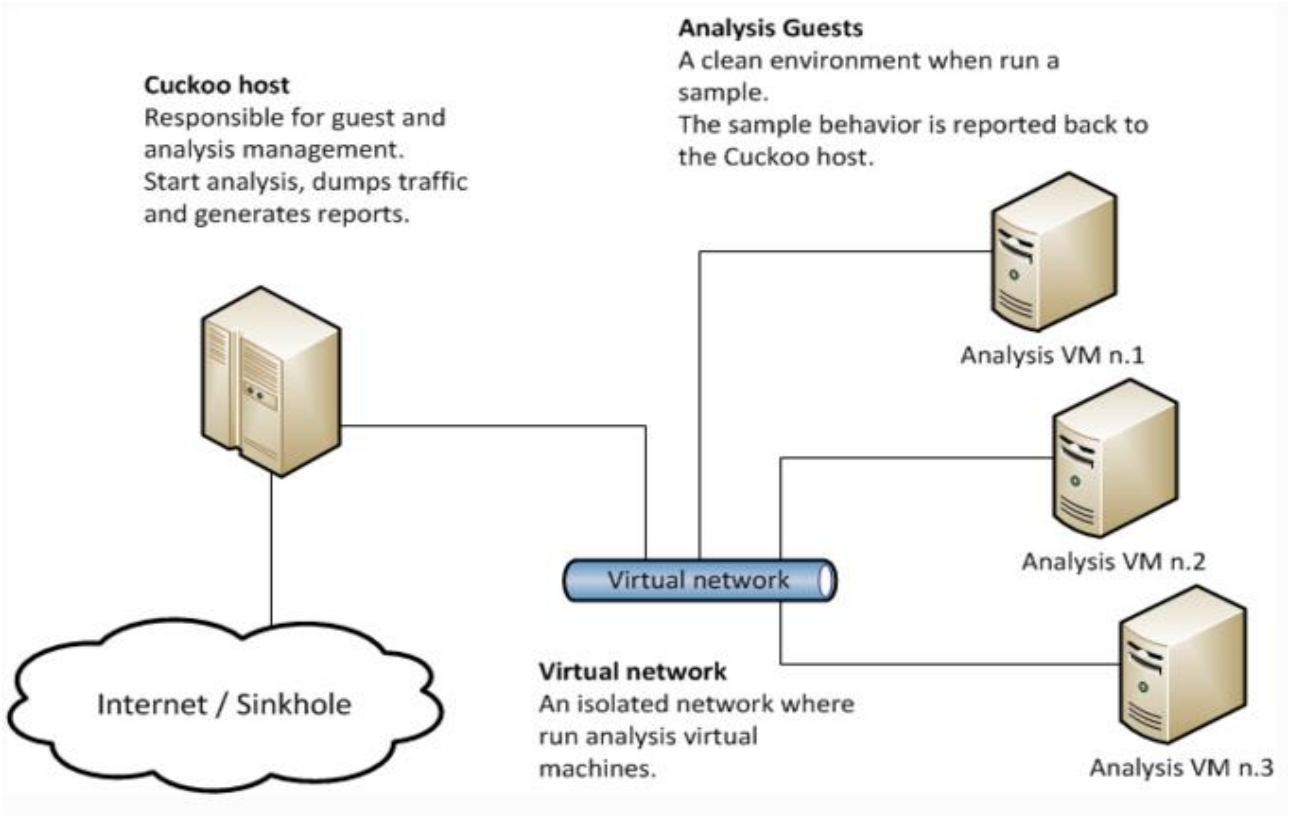
- Εκτελέσιμα αρχεία *Windows*
- Βιβλιοθήκες *DLL*
- Αρχεία *PDF*
- Αρχεία *Microsoft Office*
- Αρχεία *URL* και *HTML*
- Σενάρια *PHP*
- Αρχεία *CSL*
- Σενάρια *CSL*
- Αρχεία *ZIP*
- Αρχεία *JAR* της *JAVA*
- Αρχεία της *Python*
- Και σχεδόν τους περισσότερους άλλους τύπους αρχείων

Το *Cuckoo sandbox* μπορεί να παράγει τα ακόλουθα αποτελέσματα για τα παραπάνω υπό

εξέταση αρχεία:

- Ίχνη καλεσμάτων διεργασιών που παράχθηκαν από την ύποπτη οντότητα.
- Αρχεία που δημιουργήθηκαν, διαγράφτηκαν και κατεβάστηκαν από το διαδίκτυο κατά την εκτέλεση του κακόβουλου λογισμικού-στόχου.
- Στιγμιότυπα μνήμης κακόβουλων διεργασιών.
- Κίνηση διαδικτύου σε μορφή αρχείου *PCAP.*
- Στιγμιότυπα οθόνης που αποτυπώθηκαν κατά την διάρκεια εκτέλεσης.
- Πλήρη στιγμιότυπα μνήμης της εικονικής μηχανής.

Η αρχιτεκτονική με την οποία έχει αναπτυχθεί το εργαλείο είναι αρκετά σημαντική για να κατανοήσει ο αναλυτής τον τρόπο με τον οποίο λειτουργεί ώστε να μπορεί να επέμβει στον πηγαίο κώδικα και να κάνει τις απαραίτητες βελτιώσεις, όπου χρειάζεται, με σκοπό να πετύχει τα ακριβή αποτελέσματα που αυτός επιθυμεί. Το *Cuckoo* αποτελείται από ένα κεντρικό λογισμικό διαχείρισης που αναλαμβάνει τόσο την εκτέλεση του υπό εξέταση δείγματος και της διενέργεια της αυτοματοποιημένης ανάλυσης. Κάθε διαδικασία ανάλυσης διενεργείται σε μία νέα, αμόλυντη και απομονωμένη εικονική μηχανή. Τα βασικά στοιχεία της δομής του εργαλείου είναι ο υπολογιστής – οικοδεσπότης, στον οποίο βρίσκεται και τρέχει το λογισμικό διαχείρισης που υλοποιεί την διαδικασία της ανάλυσης, και οι υπολογιστές – επισκέπτες, στους οποίους διενεργείται η ανάλυση στα δείγματα κακόβουλων λογισμικών.



Εικόνα 7: Λειτουργική Δομή του εργαλείου Cuckoo SandBox

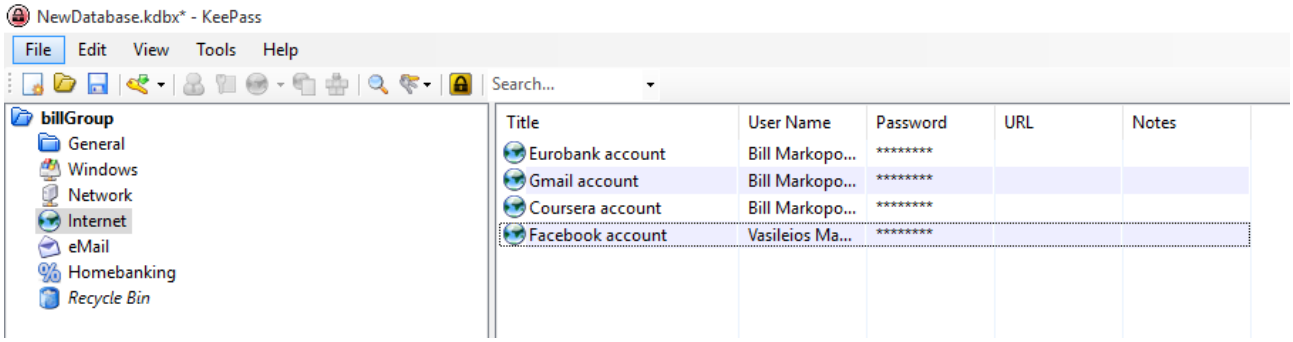
5. ΜΕΛΕΤΗ ΛΟΓΙΣΜΙΚΟΥ *KeeFarce*

Στο κεφάλαιο αυτό θα περάσουμε στην μελέτη ενός λογισμικού που ονομάζεται *KeeFarce.* Η μελέτη στην περίπτωση μας δεν περιέχει το στάδιο του εντοπισμού καθώς το *Keefarce* είναι ένα λογισμικό ανοιχτού κώδικα που προσφέρεται ελεύθερα, στο *GitHub* στην διεύθυνση https://github.com/denandz/KeeFarce , σε όποιον επιθυμεί να το αξιοποιήσει όπως έχει ή να αντλήσει πληροφορίες και τεχνικές από αυτό.

5.1 KeePass

Το *KeeFarce* έχει σχεδιαστεί για αλληλεπίδραση με το λογισμικό KeePass έκδοσης 2.x. Το λογισμικό *KeePass* είναι ένας διαχειριστής κωδικών πρόσβασης ανοιχτού κώδικα. Ο κάθε χρήστης χρειάζεται να συνεχώς να δημιουργεί καινούργιους κωδικούς για να έχει πρόσβαση σε διάφορες ιστοσελίδες και υπηρεσίες. Η χρήση του ίδιου κωδικού κάθε φορά, μπορεί να είναι βολική από την άποψη ότι είναι εύκολο για τον χρήστη να τον μνημονεύει, αλλά αποτελεί μέγα λάθος όσον αφορά την ασφάλεια των συναλλαγών. Στην περίπτωση που ένα κακόβουλο λογισμικό καταφέρει και υποκλέψει τον έναν αυτό κωδικό, μετά αποτελεί κοινή πρακτική των επιτιθέμενων να χρησιμοποιούν τον ίδιο και άλλες τοποθεσίες που είναι εγγεγραμμένο το θύμα για να του υποκλέψουν όλες τις πληροφορίες. Το λογισμικό *KeePass* καλύπτει την ανάγκη του χρήστη για την ασφαλή αποθήκευση όλων των κωδικών του. Το πρόγραμμα δίνει την δυνατότητα στον χρήστη να αποθήκευση όλους του κωδικούς σε μία Βάση Δεδομένων, τα περιεχόμενα της οποίας κρυπτογραφούνται και ασφαλίζονται από έναν κύριο κωδικό ή αρχείο κλειδί. Έτσι ο χρήστης χρειάζεται να θυμάται μόνο έναν κωδικό για έχει πρόσβαση σε όλους τους υπόλοιπους, ενώ ο ίδιος ο κωδικός παραμένει ασφαλής καθώς ο χρήστης δεν τον εισάγει σε καμία τοποθεσία στο διαδίκτυο. Οι Βάσεις Δεδομένων κρυπτογραφούνται με τους καλύτερους και πλέον σύγχρονους αλγορίθμους κρυπτογράφησης, *AES* και *Twofish.* [23]

Το εργαλείο αυτό ευαγγελίζεται την ασφάλεια με την οποία θωρακίζει τα δεδομένα που διαχειρίζεται μέσω ισχυρών κρυπτογραφικών προτύπων. Ωστόσο, όπως αποδείχθηκε στην πράξη η υποκλοπή αυτών των στοιχείων δεν αποτελεί αδύνατη πρακτική καθιστώντας το χρήστη ευάλωτο σε μία καλοστημένη επίθεση.



Εικόνα 8: Βάση Δεδομένων κωδικών KeePass

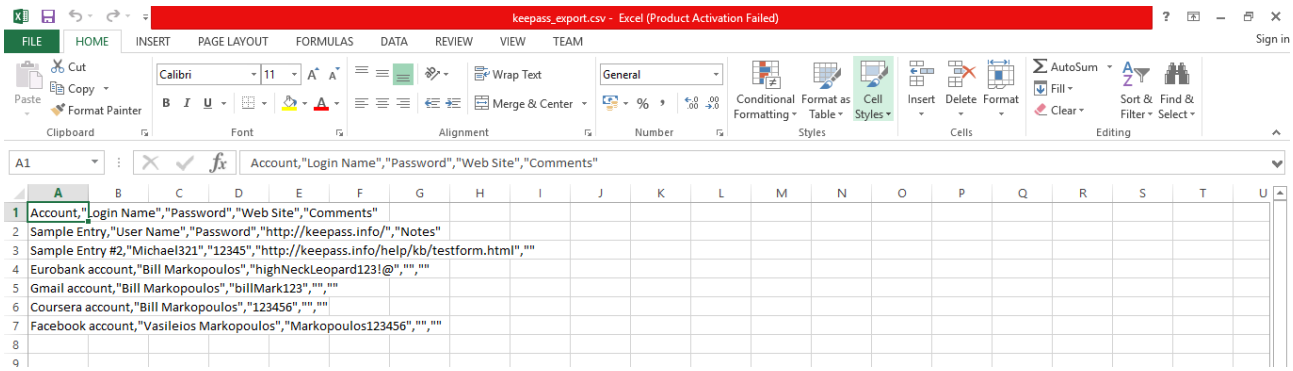
Η έκδοση που χρησιμοποιήσαμε εμείς φαίνεται στην Εικόνα 9.

Εικόνα 9: Πληροφορίες Έκδοσης του KeePass που χρησιμοποιήθηκε για εξέταση

5.2 KeeFarce

Το λογισμικό *Keefarce* επιτρέπει την εξαγωγή από την μνήμη όλων των πληροφοριών που αποθηκεύονται στις Βάσεις Δεδομένων του *KeePass*. Το καθαρό κείμενο που εξάγεται αφορά στα ονόματα χρήστη, κωδικούς, σημειώσεις και διευθύνσεις *URL* και αποθηκεύονται σε ένα αρχείο τύπου *CSV* στη διαδρομή δίσκου *C:Users/%/AppData/Roaming/*. Το λογισμικό αυτό δημοσιεύτηκε στον ιστόχωρο του *GitHub* τον Νοέμβριο του 2015 και καταρρίπτει το μύθο της ασφάλειας των προγραμμάτων διαχείρισης κωδικών πρόσβασης τοπικά στους

υπολογιστές των χρηστών. Στην Εικόνα 10 φαίνονται τα στοιχεία μίας Βάσης Δεδομένων που έχουν υποκλαπεί και αποτυπωθεί σε ένα αρχείο CSV από τον *KeeFarce.* [24]



Εικόνα 10: Αρχείο .csv στο οποίο αποθηκεύονται τα κλεμμένα στοιχεία

Η δημοσίευσή του συνοδεύτηκε από αρκετά άρθρα που κάνουν λόγο για την απειλή που εγέρθηκε με την έλευσή του. Το λογισμικό χρησιμοποιεί κάποιες αρκετά προηγμένες τεχνικές για την συγκομιδή δεδομένων από την μνήμη του υπολογιστή, ενώ παρόλο που το ίδιο χρησιμοποιεί κάποια ονόματα ταμπού (θα αναφερθούν στην συνέχεια) στον χώρο της μελέτης των κακόβουλων λογισμικών, η ελαφρά τροποποίηση του μπορεί να φέρει στο προσκήνιο ένα πολύ ισχυρό κακόβουλο εργαλείο υποκλοπής δεδομένων. Εν γένει η κακόβουλη δυναμική του εν λόγω λογισμικού εκδηλώνεται με την μέθοδο του *DLL Injection.* Η μελέτη του εν λόγω λογισμικού πραγματοποιήθηκε για τους ακολούθους λόγους:

1. Η δράση του μπορεί να προκαλέσει σημαντική απώλεια ευαίσθητων δεδομένων.
2. Δεν έχει δημοσιευτεί ποτέ προηγουμένως παρόμοια μελέτη που να αφορά τον ίδιο λογισμικό.
3. Ενσωματώνει τεχνολογίες σύγχρονες που έχουν σχεδιαστεί για νέα λειτουργικά συστήματα. Οι ίδιες αποτελούν ισχυρό παράδειγμα του πλούτου των πληροφοριών που μπορούν να υποκλαπούν από την μνήμη *RAM* και δύναται να χρησιμοποιηθούν από ποικίλα κακόβουλα λογισμικά στο μέλλον. Η τελευταία παρατήρηση αποκαλύπτει την ουσία της σημασίας μίας μελέτης σε ένα καινοτόμο κακόβουλο λογισμικό. Η δαιμονοποίηση των συστατικών από τα οποία αποτελείται δομικά το *KeeFarce* και των αποτελεσμάτων από την ανάλυση της συμπεριφοράς του, δημιουργεί μία καλή βάση προστασίας απέναντι σε λογισμικά που τις χρησιμοποιούν ή πρόκειται να βασιστούν σε αυτό και να τις χρησιμοποιήσουν.
4. Πλήττει ένα τομέα αρκετά σημαντικό (μνήμη *RAM*), η προσβολή του οποίου έχει πάρει τεράστιες διαστάσεις τα τελευταία τρία χρόνια. Το λογισμικό παρουσιάζει μεγάλη ομοιότητα, ως προς τις τεχνικές που χρησιμοποιεί, με τους *RAM Scappers.* Οι *RAM Scrappers* έχουν αποτελέσει στόχο της κοινότητας των αναλυτών κακόβουλων

λογισμικών καθώς η δράση τους είναι δύσκολο να παρεμποδιστεί ή να αποκρυφθούν δεδομένα από αυτούς, καθώς ότι πληροφορία υπάρχει στον υπολογιστή περνάει κάποια στιγμή από την *RAM.*

5. Η μελέτη του δεν αποτέλεσε ρίσκο για τον υπολογιστή ανάλυσης, καθώς ξέραμε με τι είχαμε να κάνουμε και είχαμε μηχανισμούς να παρεμποδίσουμε την δράση του σε περίπτωση που αυτή ξέφευγε από τα αποδεκτά όρια της εργαστηριακής μελέτης.

5.2.1 Αρχιτεκτονική

Το κακόβουλο λογισμικό *KeeFarce* αποτελείται από τα εξής αρχεία, η παρουσία των οποίων στον ίδιο φάκελο είναι απαραίτητη για την λειτουργία του:

- *KeeFarce.exe*
- *Bootstrap.dll*
- *KeeFarceDLL.dll*
- *Microsoft.Diagnostics.Runtime.dll (*Βασική βιβλιοθήκη των *Windows)*

Στη συνέχεια θα αναλύσουμε κάθε ένα από αυτά τα αρχεία προς χάρη κατανόησης της δομής και της λειτουργικότητας. Ο λόγος που ξεκινάμε από την εξέταση του πηγαίου κώδικα του λογισμικού και όχι από τον εντοπισμό των στοιχείων *IOC,* είναι για την ανάδειξη της εμπειρίας που χρειάζεται πάνω στο θέμα για την ανίχνευση και την ανάλυση μίας ύποπτης κινητικότητας ενός άγνωστου λογισμικού στο σύστημα. Σε μία πραγματική περίπτωση στην οποία θέλουμε να αναλύσουμε ένα κακόβουλο λογισμικό, δεν θα έχουμε στα χέρια μας τον πηγαίο κώδικα αλλά θα πρέπει βήμα βήμα να φτάσουμε όσο πιο κοντά στην αναπαράσταση του μέσω του τελευταίου σταδίου της ανάλυσης, της αντίστροφης μηχανικής. Ωστόσο, όπως αναφέρθηκε και προηγουμένως, στην περίπτωση του *Keefarce,* δεν είναι το ίδιο το λογισμικό που μας απειλεί αλλά οι προεκτάσεις και η χρήση των τεχνικών που ενσωματώνει για την διενέργεια κακόβουλων υποκλοπών. Συνεπώς, γίνεται κατανοητό πως στην περίπτωση μας η εξέταση του πηγαίου κώδικα του είναι ένα βασικό στάδιο στην αναγνώριση των τεχνικών που χρησιμοποιεί και δύναται να χρησιμοποιηθούν και σε άλλες επιθέσεις.

5.2.1.1 KeeFarce.exe

Το αρχείο αυτό είναι εκτελέσιμο και επιτελεί το ρόλο του *injector.* Αυτό, πρακτικά σημαίνει ότι σε αυτό εμπεριέχεται ο κώδικας που ενορχηστρώνει τις κακόβουλες ενέργειες για την επίτευξη της υποκλοπής των στοιχείων των Βάσεων Δεδομένων του *KeePass*. Το εκτελέσιμο περιέχει δύο αρχεία κώδικα στα οποία συμπεριλαμβάνεται η λειτουργικότητα του: το *KeeFarce.cpp* και το *Injection.cpp.*

Το *KeeFarce.cpp* προσδιορίζει το σημείο αρχής εκτέλεσης για το λογισμικό κονσόλας που ευθύνεται για την συνολική εκτέλεση του *DLL Injection.* Στο αρχείο αυτό, αποκτώνται πληροφορίες διαδρομής δίσκου των βιβλιοθηκών *KeeFarceDLL.dll* και *BootstrapDLL.dll* και δίνονται ως είσοδος στην συνάρτηση *InjectAndRunThenUnload* του αρχείου *Injection.cpp* για την διενέργεια του *injection.*
Το *Injection.cpp* περιέχει τις εξής συναρτήσεις:

- *GetProcessIdByName* : με την χρήση αυτής της συνάρτησης αποκτάται το ακέραιο αναγνωριστικό της διεργασίας με όνομα *KeePass.exe*
- *InjectAndRunThenUnload* : η συνάρτηση αυτή αφού δημιουργήσει κατάλληλο χώρο μνήμης στην διεργασία θύμα, φορτώνει μέσω της συνάρτησης *LoadLibraryA* της βασικής βιβλιοθήκης *kernel32.dll* την βιβλιοθήκη *Bootstrap.dll* και αφού αυτή εκτελεστεί ελευθερώνει το χώρο που αυτή κατέλαβε.
- *CallExport* : Η συνάρτηση αυτή χρησιμοποιείται για να αποκτηθεί η διεύθυνση και μπορέσει να εκτελεστεί η συνάρτηση *LoadManagedProject* της βιβλιοθήκης *Bootstrap.dll.*

5.2.1.2 Bootstrap.dll

Η βιβλιοθήκη αυτή ευθύνεται για την φόρτωση μίας άλλης βιβλιοθήκης, της *KeeFarceDLL.dll,* η οποία είναι και αυτή που υποκλέπτει τα δεδομένα. Για επιτευχθεί αυτό η *Bootstrap.dll* περιέχει τις εξής δύο συναρτήσεις:

- *LoadManagedProject* : ευθύνεται για την εκτέλεση της *KeeFarceDLL.dll* μέσα στο τρέχων *app domain.*
- *StartCLR* : ευθύνεται για την εκκίνηση ενός *CLR runtime* για την εκτέλεση λειτουργιών αποσφαλμάτωσης παράλληλα με την εκτέλεση της διεργασίας θύματος *KeePass.exe.* Αυτή η πράξη οδηγεί στη εκτέλεση των περιεχομένων της βιβλιοθήκης *KeeFarceDLL.dll* παράλληλα με το κανονικό πρόγραμμα και την υποκλοπή των

στοιχείων που αυτό περιέχει.

5.2.1.3 KeeFarceDLL.dll

Στο αρχείο *main.cs* της βιβλιοθήκης αυτής περιέχεται η κλάση *KeeFarce,* οι συναρτήσεις τις οποίας υλοποιούν το κακόβουλο έργο της υποκλοπής των δεδομένων μίας **Βάσης Δεδομένων, η οποία είναι, στο παρόντα χρόνο εκτέλεσης του κακόβουλου λογισμικού, στην μνήμη του *KeePass,*** υπό τον μανδύα διαδικασιών αποσφαλμάτωσης. Συγκεκριμένα περιέχονται οι παρακάτω συναρτήσεις:

- *EntryPoint* : χρήση των παρακάτω συναρτήσεων για εξαγωγή σε ένα αρχείο *csv* των περιεχομένων του πρώτου αρχείου κωδικών που θα προσπελάσει στην μνήμη της διεργασίας – θύμα.
- *doExport* : εξαγωγή των δεδομένων της ενεργής, στην μνήμη του *KeePass,* Βάσης Δεδομένων, σε ένα αρχείο *Microsoft Excel.*
- *CreateRuntimeHack* : εκτέλεση πειρατείας δημιουργώντας ένα καινούργιο στιγμιότυπο στο runtime του *KeePass.*

5.2.1.4 Microsoft.Diagnostics.Runtime.dll

Η βιβλιοθήκη αυτή παρέχει ένα σετ από *APIs* για ενδοσκόπηση διεργασιών και στιγμιότυπων που προκύπτουν από προβλήματα συστήματος (*crash dumps*). Η βιβλιοθήκη δίνει, δηλαδή, την δυνατότητα για συγγραφή εργαλείων και επεκτάσεων αποσφαλματωτή. Συνεπώς, η χρήση της είναι απαραίτητη για την δημιουργία κακόβουλων λογισμικών με δυνατότητες αποσφαλμάτωσης, απαραίτητες για την εκπλήρωση εργασιών απόκτησης μνήμης και φόρτωσης δυναμικών βιβλιοθηκών στην μνήμη της διεργασίας – στόχου. Η βιβλιοθήκη ονομάζεται και *CLR MD* και τα στοιχεία που αφήνει η χρήση της στα αρχεία μεταγλώττισης και αποσφαλμάτωσης *.pdb* προδίδουν την ύπαρξη προγράμματος γραμμένου σε γλώσσες *C/C++/Visual Basic/C#/Jscript.NET* το οποίο αποτελεί και το κακόβουλο λογισμικό που αναζητούμε.

5.2.2 Μέθοδος εντοπισμού

Στην ενότητα αυτή θα παρουσιαστεί η προσπάθεια εντοπισμού του λογισμικού *KeeFarce* σε ένα υπολογιστικό σύστημα, με την παραδοχή ότι το ίδιο αποτελεί άγνωστο λογισμικό και ότι η δράση του δεν είναι γνωστή. Το στάδιο του εντοπισμού πραγματοποιήθηκε σε δύο φάσεις. Αρχικά έγινε παρακολούθηση συστήματος *Windows 10* εικονικής μηχανής ζωντανά, ταυτόχρονα με την εξέλιξη των ενεργειών του κακόβουλου λογισμικού. Σε δεύτερη φάση πραγματοποιήθηκε μελέτη στιγμιότυπου μνήμης της μνήμης όλου του συστήματος για τον εντοπισμό υπολειμμάτων της δράσης του *Keefarce.*

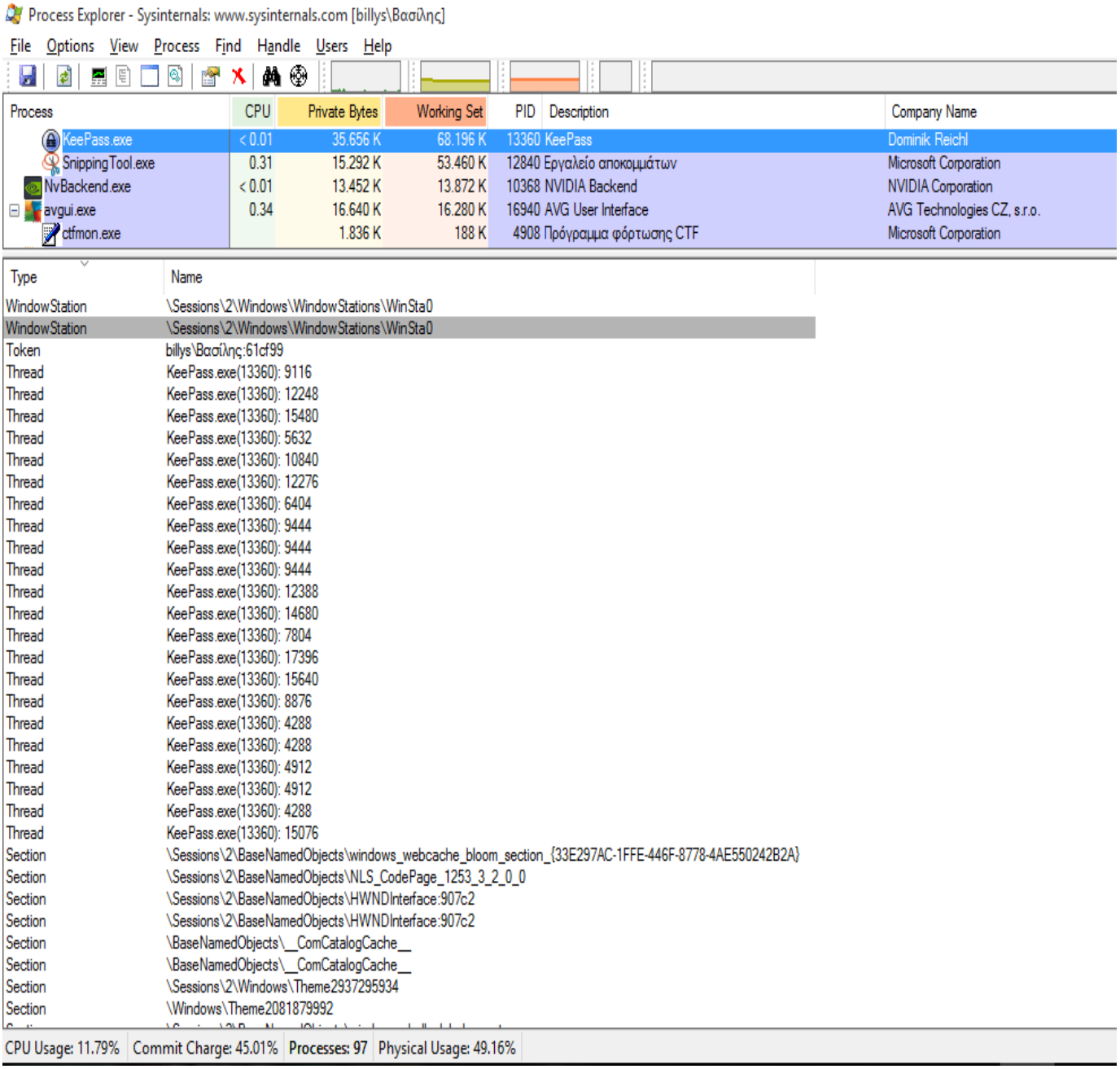
5.2.2.1 Εντοπισμός σε εικονικό λειτουργικό Windows 10

5.2.2.1.1 Παρακολούθηση της Μνήμης RAM

Στην φάση αυτή έγινε, αρχικά, μία προσπάθεια για ακριβή προσομοίωση της λειτουργίας του λειτουργικού *Windows 8.1* σε εικονικό περιβάλλον. Η εικονοποίηση έγινε σε λειτουργικό σύστημα *Windows 10* με χρήση της δυνατότητας *Hyper-V*. Παράλληλα, έγινε και παρακολούθηση σε σύστημα *Windows 10* για σύγκριση αποτελεσμάτων ως προς την δράση του κακόβουλου λογισμικού. Εν τέλει δεν παρατηρήθηκε κάποια ουσιαστική διαφορά, καθώς το *KeeFarce* δεν φαίνεται να λαμβάνει υπόψιν το αν τρέχει σε εικονική μηχανή ή όχι. Συνεπώς, θα παρουσιαστεί η διαδικασία εντοπισμού σε φυσικό λειτουργικό σύστημα *windows 10* καθώς αυτή είναι μία διαδικασία που θα ακολουθήσει και κάποιος αναλυτής όταν δεν γνωρίζει ότι το σύστημα του έχει προσβληθεί από κάτι συγκεκριμένο.

Θα ξεκινήσουμε με αναφορά σε μία εφαρμογή που περιλαμβάνεται στο πακέτο εργαλείων *SysInternal* που δημιουργήθηκε από τον *Mark Russinovich.* Η σουίτα εργαλείων που έχει εκδώσει ο συγκεκριμένος προγραμματιστής είναι ένα σύνολο από κατάλληλες εφαρμογές για διενέργεια μελέτης της περιόδου λειτουργίας και της μνήμης του λειτουργικού συστήματος. Το εργαλείο που μας αφορά λέγεται *Process Explorer* [25]. Ο *Process Explorer* είναι ένα εργαλείο με το οποίο ο αναλυτής μπορεί πέρα των άλλων να παρακολουθεί ζωντανά για κάθε μία διεργασία που εκτελείται στο σύστημα αφενός διαφορές γενικές πληροφορίες που την αφορούν (χρήση *CPU, Pid,* περιγραφή κ.α.), αφετέρου όλα τα προσδιοριστικά των πόρων της (*handles*) και τις δυναμικές βιβλιοθήκες *dll* που σχετίζονται με αυτήν. Αυτό το εργαλείο είναι αρκούντως σημαντικό καθώς η συνεχής παρατήρηση του

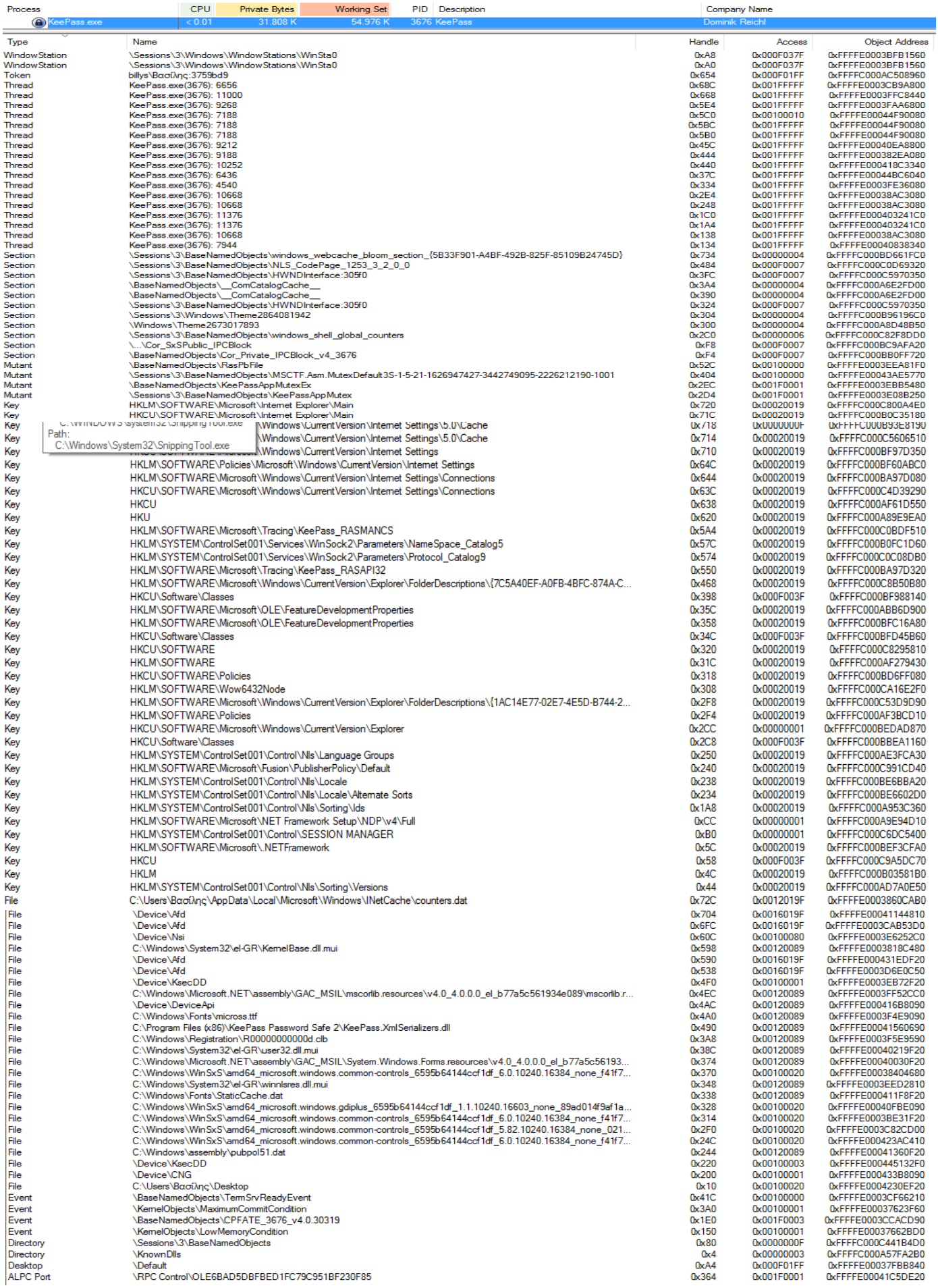
μπορεί να φέρει στο φως κάποια ένδειξη προσβολής από ένα κακόβουλο λογισμικό.



Εικόνα 11: Επιλογή της διεργασίας KeePass για εμφάνιση στοιχείων από τον Process Explorer

Στην δική μας περίπτωση έγινε σύγκριση της εικόνας που παρουσιάζουν τόσο τα *Handles* όσο και οι *dlls* που σχετίζονται με την εφαρμογή *KeePass*, πριν και μετά την εκτέλεση του *KeeFarce*, με σκοπό την ανίχνευση οποιασδήποτε ύποπτης δραστηριότητας.

Αρχικά θα παρακολουθήσουμε τι γίνεται με τα *Handles* πριν την εκτέλεση του κακόβουλου λογισμικού. Αυτά είναι κατηγοριοποιημένα ανά τύπο πόρου όπως εμφανίζονται από τον *Process Explorer* στην Εικόνα 12.



Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
KeePass.exe	< 0.01	31.808 K	54.976 K	3676	KeePass	Dominik Reichl

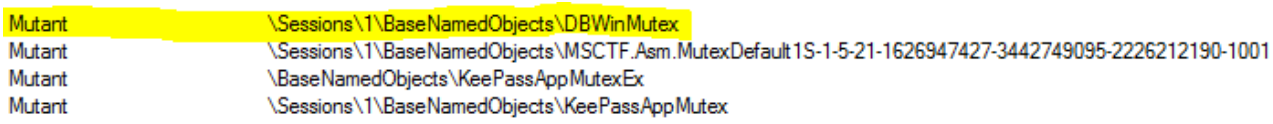
Type	Name	Handle	Access	Object Address
WindowStation	\Sessions\3\Windows\WindowStations\WinSta0	0xA8	0x000F037F	0xFFFFE0003BFB1560
WindowStation	\Sessions\3\Windows\WindowStations\WinSta0	0xA0	0x000F037F	0xFFFFE0003BFB1560
Token	billys\Βασίλης:3759bd9	0x654	0x000F01FF	0xFFFFC000AC508960
Thread	KeePass.exe(3676): 6656	0x68C	0x001FFFFF	0xFFFFE0003CB9A800
Thread	KeePass.exe(3676): 11000	0x668	0x001FFFFF	0xFFFFE0003FFC8440
Thread	KeePass.exe(3676): 9268	0x5E4	0x001FFFFF	0xFFFFE0003FAA6800
Thread	KeePass.exe(3676): 7188	0x5C0	0x00100010	0xFFFFE00044F90080
Thread	KeePass.exe(3676): 7188	0x5BC	0x001FFFFF	0xFFFFE00044F90080
Thread	KeePass.exe(3676): 7188	0x5B0	0x001FFFFF	0xFFFFE00044F90080
Thread	KeePass.exe(3676): 9212	0x45C	0x001FFFFF	0xFFFFE00040EA8800
Thread	KeePass.exe(3676): 9188	0x444	0x001FFFFF	0xFFFFE000382EA080
Thread	KeePass.exe(3676): 10252	0x440	0x001FFFFF	0xFFFFE000418C3340
Thread	KeePass.exe(3676): 6436	0x37C	0x001FFFFF	0xFFFFE00044BC6040
Thread	KeePass.exe(3676): 4540	0x334	0x001FFFFF	0xFFFFE0003FE36080
Thread	KeePass.exe(3676): 10668	0x2E4	0x001FFFFF	0xFFFFE00038AC3080
Thread	KeePass.exe(3676): 10668	0x248	0x001FFFFF	0xFFFFE00038AC3080
Thread	KeePass.exe(3676): 11376	0x1C0	0x001FFFFF	0xFFFFE000403241C0
Thread	KeePass.exe(3676): 11376	0x1A4	0x001FFFFF	0xFFFFE000403241C0
Thread	KeePass.exe(3676): 10668	0x138	0x001FFFFF	0xFFFFE00038AC3080
Thread	KeePass.exe(3676): 7944	0x134	0x001FFFFF	0xFFFFE00040838340
Section	\Sessions\3\BaseNamedObjects\windows_webcache_bloom_section_{5B33F901-A4BF-492B-825F-85109B24745D}	0x734	0x00000004	0xFFFFC000BD661FC0
Section	\Sessions\3\BaseNamedObjects\NLS_CodePage_1253_3_2_0_0	0x484	0x000F0007	0xFFFFC000C0D69320
Section	\Sessions\3\BaseNamedObjects\HWNDInterface:305f0	0x3FC	0x000F0007	0xFFFFC000C5970350
Section	\BaseNamedObjects__ComCatalogCache__	0x3A4	0x00000004	0xFFFFC000A6E2FD00
Section	\BaseNamedObjects__ComCatalogCache__	0x390	0x00000004	0xFFFFC000A6E2FD00
Section	\Sessions\3\BaseNamedObjects\HWNDInterface:305f0	0x324	0x000F0007	0xFFFFC000C5970350
Section	\Sessions\3\Windows\Theme2864081942	0x304	0x00000004	0xFFFFC000B96196C0
Section	\Windows\Theme2673017893	0x300	0x00000004	0xFFFFC000A8D48B50
Section	\Sessions\3\BaseNamedObjects\windows_shell_global_counters	0x2C0	0x00000006	0xFFFFC000C82F8DD0
Section	\...\Cor_SxSPublic_IPCBlock	0xF8	0x000F0007	0xFFFFC000BC9AFA20
Section	\BaseNamedObjects\Cor_Private_IPCBlock_v4_3676	0xF4	0x000F0007	0xFFFFC000BB0FF720
Mutant	\BaseNamedObjects\RasPbFile	0x52C	0x00100000	0xFFFFE0003EEA81F0
Mutant	\Sessions\3\BaseNamedObjects\MSCTF.Asm.MutexDefault3S-1-5-21-1626947427-3442749095-2226212190-1001	0x404	0x00100000	0xFFFFE00043AE5770
Mutant	\BaseNamedObjects\KeePassAppMutexEx	0x2EC	0x001F0001	0xFFFFE0003EBB5480
Mutant	\Sessions\3\BaseNamedObjects\KeePassAppMutex	0x2D4	0x001F0001	0xFFFFE0003E08B250
Key	HKLM\SOFTWARE\Microsoft\Internet Explorer\Main	0x720	0x00020019	0xFFFFC000C800A4E0
Key	HKCU\SOFTWARE\Microsoft\Internet Explorer\Main	0x71C	0x00020019	0xFFFFC000B0C35180
Key	C:\WINDOWS\system32\SnippingTool.exe\Windows\CurrentVersion\Internet Settings\5.0\Cache	0x718	0x00020019	0xFFFFC000B93E8190
Key	\Windows\CurrentVersion\Internet Settings\5.0\Cache	0x714	0x00020019	0xFFFFC000C5606510
Key	\Windows\CurrentVersion\Internet Settings	0x710	0x00020019	0xFFFFC000BF97D350
Key	HKLM\SOFTWARE\Policies\Microsoft\Windows\CurrentVersion\Internet Settings	0x64C	0x00020019	0xFFFFC000BF60ABC0
Key	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Connections	0x644	0x00020019	0xFFFFC000BA97D080
Key	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Connections	0x63C	0x00020019	0xFFFFC000C4D39290
Key	HKCU	0x638	0x00020019	0xFFFFC000AF61D550
Key	HKU	0x620	0x00020019	0xFFFFC000A89E9EA0
Key	HKLM\SOFTWARE\Microsoft\Tracing\KeePass_RASMANCS	0x5A4	0x00020019	0xFFFFC000C0BDF510
Key	HKLM\SYSTEM\ControlSet001\Services\WinSock2\Parameters\NameSpace_Catalog5	0x57C	0x00020019	0xFFFFC000B0FC1D60
Key	HKLM\SYSTEM\ControlSet001\Services\WinSock2\Parameters\Protocol_Catalog9	0x574	0x00020019	0xFFFFC000C0C08DB0
Key	HKLM\SOFTWARE\Microsoft\Tracing\KeePass_RASAPI32	0x550	0x00020019	0xFFFFC000BA97D320
Key	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\FolderDescriptions\{7C5A40EF-A0FB-4BFC-874A-C...	0x468	0x00020019	0xFFFFC000C8B50B80
Key	HKCU\Software\Classes	0x398	0x000F003F	0xFFFFC000BF988140
Key	HKLM\SOFTWARE\Microsoft\OLE\FeatureDevelopmentProperties	0x35C	0x00020019	0xFFFFC000ABB6D900
Key	HKLM\SOFTWARE\Microsoft\OLE\FeatureDevelopmentProperties	0x358	0x00020019	0xFFFFC000BFC16A80
Key	HKCU\Software\Classes	0x34C	0x000F003F	0xFFFFC000BFD45B60
Key	HKCU\SOFTWARE	0x320	0x00020019	0xFFFFC000C8295810
Key	HKLM\SOFTWARE	0x31C	0x00020019	0xFFFFC000AF279430
Key	HKCU\SOFTWARE\Policies	0x318	0x00020019	0xFFFFC000BD6FF080
Key	HKLM\SOFTWARE\Wow6432Node	0x308	0x00020019	0xFFFFC000CA16E2F0
Key	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\FolderDescriptions\{1AC14E77-02E7-4E5D-B744-2...	0x2F8	0x00020019	0xFFFFC000C53D9D90
Key	HKLM\SOFTWARE\Policies	0x2F4	0x00020019	0xFFFFC000AF3BCD10
Key	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer	0x2CC	0x00000001	0xFFFFC000BEDAD870
Key	HKCU\Software\Classes	0x2C8	0x000F003F	0xFFFFC000BBEA1160
Key	HKLM\SYSTEM\ControlSet001\Control\Nls\Language Groups	0x250	0x00020019	0xFFFFC000AE3FCA30
Key	HKLM\SOFTWARE\Microsoft\Fusion\PublisherPolicy\Default	0x240	0x00020019	0xFFFFC000C991CD40
Key	HKLM\SYSTEM\ControlSet001\Control\Nls\Locale	0x238	0x00020019	0xFFFFC000BE6BBA20
Key	HKLM\SYSTEM\ControlSet001\Control\Nls\Locale\Alternate Sorts	0x234	0x00020019	0xFFFFC000BE6602D0
Key	HKLM\SYSTEM\ControlSet001\Control\Nls\Sorting\Ids	0x1A8	0x00020019	0xFFFFC000A953C360
Key	HKLM\SOFTWARE\Microsoft\NET Framework Setup\NDP\v4\Full	0xCC	0x00000001	0xFFFFC000A9E94D10
Key	HKLM\SYSTEM\ControlSet001\Control\SESSION MANAGER	0xB0	0x00000001	0xFFFFC000C6DC5400
Key	HKLM\SOFTWARE\Microsoft\.NETFramework	0x5C	0x00020019	0xFFFFC000BEF3CFA0
Key	HKCU	0x58	0x000F003F	0xFFFFC000C9A5DC70
Key	HKLM	0x4C	0x00020019	0xFFFFC000B03581B0
Key	HKLM\SYSTEM\ControlSet001\Control\Nls\Sorting\Versions	0x44	0x00020019	0xFFFFC000AD7A0E50
File	C:\Users\Βασίλης\AppData\Local\Microsoft\Windows\INetCache\counters.dat	0x72C	0x0012019F	0xFFFFE0003860CAB0
File	\Device\Afd	0x704	0x0016019F	0xFFFFE00041144810
File	\Device\Afd	0x6FC	0x0016019F	0xFFFFE0003CAB53D0
File	\Device\Nsi	0x60C	0x00100080	0xFFFFE0003E6252C0
File	C:\Windows\System32\el-GR\KernelBase.dll.mui	0x598	0x00120089	0xFFFFE0003818C480
File	\Device\Afd	0x590	0x0016019F	0xFFFFE000431EDF20
File	\Device\Afd	0x538	0x0016019F	0xFFFFE0003D6E0C50
File	\Device\KsecDD	0x4F0	0x00100001	0xFFFFE0003EB72F20
File	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\mscorlib.resources\v4.0_4.0.0.0_el_b77a5c561934e089\mscorlib.r...	0x4EC	0x00120089	0xFFFFE0003FF52CC0
File	\Device\DeviceApi	0x4AC	0x00120089	0xFFFFE000416B8090
File	C:\Windows\Fonts\micross.ttf	0x4A0	0x00120089	0xFFFFE0003F4E9090
File	C:\Program Files (x86)\KeePass Password Safe 2\KeePass.XmlSerializers.dll	0x490	0x00120089	0xFFFFE00041560690
File	C:\Windows\Registration\R000000000000d.clb	0x3A8	0x00120089	0xFFFFE0003F5E9590
File	C:\Windows\System32\el-GR\user32.dll.mui	0x38C	0x00120089	0xFFFFE00040219F20
File	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Windows.Forms.resources\v4.0_4.0.0.0_el_b77a5c56193...	0x374	0x00120089	0xFFFFE00040030F20
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.10240.16384_none_f41f7...	0x370	0x00100020	0xFFFFE00038404680
File	C:\Windows\System32\el-GR\winnlsres.dll.mui	0x348	0x00120089	0xFFFFE0003EED2810
File	C:\Windows\Fonts\StaticCache.dat	0x338	0x00120089	0xFFFFE000411F8F20
File	C:\Windows\WinSxS\amd64_microsoft.windows.gdiplus_6595b64144ccf1df_1.1.10240.16603_none_89ad014f9af1a...	0x328	0x00100020	0xFFFFE00040FBE090
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.10240.16384_none_f41f7...	0x314	0x00100020	0xFFFFE0003BE31F20
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_5.82.10240.16384_none_021...	0x2F0	0x00100020	0xFFFFE0003C82CD00
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.10240.16384_none_f41f7...	0x24C	0x00100020	0xFFFFE000423AC410
File	C:\Windows\assembly\pubpol51.dat	0x244	0x00120089	0xFFFFE00041360F20
File	\Device\KsecDD	0x220	0x00100003	0xFFFFE000445132F0
File	\Device\CNG	0x200	0x00100001	0xFFFFE000433B8090
File	C:\Users\Βασίλης\Desktop	0x10	0x00100020	0xFFFFE0004230EF20
Event	\BaseNamedObjects\TermSrvReadyEvent	0x41C	0x00100000	0xFFFFE0003CF66210
Event	\KernelObjects\MaximumCommitCondition	0x3A0	0x00100001	0xFFFFE00037623F60
Event	\BaseNamedObjects\CPFATE_3676_v4.0.30319	0x1E0	0x001F0003	0xFFFFE0003CCACD90
Event	\KernelObjects\LowMemoryCondition	0x150	0x00100001	0xFFFFE00037662BD0
Directory	\Sessions\3\BaseNamedObjects	0x80	0x0000000F	0xFFFFC000C441B4D0
Directory	\KnownDlls	0x4	0x00000003	0xFFFFC000A57FA2B0
Desktop	\Default	0xA4	0x000F01FF	0xFFFFE00037FBB840
ALPC Port	\RPC Control\OLE6BAD5DBFBED1FC79C951BF230F85	0x364	0x001F0001	0xFFFFE00041C5DE20

Path:
C:\Windows\System32\SnippingTool.exe

Εικόνα 12: Τα Handles της διεργασίας που εμφανίζει ο Process Explorer

Κατόπιν εκτέλεσης του *KeeFarce* παρατηρούμε τα εξής:

1. Δημιουργία ενός νέου νήματος – *thread* της εφαρμογής, γεγονός που είναι απίθανο να συμβεί όταν η διεργασία είναι σε κατάσταση ηρεμίας και δεν πραγματοποιεί κάποια ενέργεια. Συνεπώς κάτι άλλο είναι αυτό που προκαλεί και μας προειδοποιεί για την ύπαρξη ενός *Runtime Hack* της εφαρμογής.
2. Η δημιουργία ενός καινούργιου *mutex* όπως φαίνεται στην Εικόνα 13.



Εικόνα 13: Ύποπτο mutex αποσφαλμάτωσης

Το *DBWinMutex* είναι ένα *mutant* αντικείμενο το οποίο εγείρεται στην περίπτωση που εφαρμόζεται κάποια τεχνική αποσφαλμάτωσης πάνω στην διεργασία και υποδηλώνει σε οποιαδήποτε άλλη διεργασία επιθυμεί να προσκολληθεί, με δικαιώματα διαχειριστή, στην διεργασία αυτή ότι κάποια άλλη οντότητα την παρακολουθεί ήδη. Αυτή η παρατήρηση αποτελεί πολύ ισχυρή ένδειξη ότι η λειτουργία της εφαρμογής δεν ρέει φυσιολογικά, καθώς κάποια διεργασία έχει αποκτήσει εποπτεία των πόρων της και δικαίωμα επέμβασης στην μνήμη της. Το *Keefarce* έχει ήδη διαβάσει την μνήμη του *KeePass.*

Ωστόσο οι παρατηρήσεις δεν σταματούν εδώ καθώς και στην καρτέλα των *dlls* παρατηρούνται ενδείξεις προσβολής της εφαρμογής. Αρχικά παραθέτουμε την εικόνα των *dlls* που έχουν φορτωθεί στην εφαρμογή πριν εκτελεστεί το *KeeFarce* στην Εικόνα 14.
Παρατηρούμε ότι όλες οι βιβλιοθήκες που υπάρχουν είναι κοινές και δεν παρουσιάζουν κάτι το ύποπτο. Ωστόσο, παρατηρώντας τις φορτωμένες βιβλιοθήκες κατόπιν εκτέλεσης του KeeFarce εντοπίζουμε κάποιες που είναι αρκετά ύποπτες και καθόλου συνηθισμένες:

- *KeeFarceDLL.dll* : Η βιβλιοθήκη αυτή προφανώς είναι ένδειξη φόρτωσης μία κακόβουλης βιβλιοθήκης και η διαδρομή της στον δίσκο μας δίνει το πρώτο στοιχείο για το μέρος στο οποίο μπορεί να δρα το κακόβουλο λογισμικό(*C:\Program Files (x86)\KeePass Password Safe 2\KeeFarceDLL.dll*)
- *BootstrapDLL.dll* : Άλλη μία βιβλιοθήκη με μη συνηθισμένο όνομα, η διαδρομή της οποία στο δίσκο (*C:\Program Files (x86)\KeePass Password Safe 2\BootstrapDLL.dll*) μας υποδεικνύει πως δεν μπορεί να σχετίζεται με κάποια ενέργεια εκκίνησης του συστήματος αλλά κάποια κακόβουλη ενέργεια.

Process: KeePass.exe Pid: 8468

Name	Description	Path	Company
wtsapi32.dll	Windows Remote Desktop Sessio...	C:\Windows\System32\wtsapi32.dll	Microsoft Corporation
winsta.dll	Winstation Library	C:\Windows\System32\winsta.dll	Microsoft Corporation
winspool.drv	Πρόγραμμα οδήγησης ουράς εκτ...	C:\Windows\System32\winspool.drv	Microsoft Corporation
winnlsres.dll.mui	DLL πόρου NLSBuild	C:\Windows\System32\el-GR\winnlsres.dll.mui	Microsoft Corporation
winnlsres.dll	DLL πόρου NLSBuild	C:\Windows\System32\winnlsres.dll	Microsoft Corporation
winmmbase.dll	Base Multimedia Extension API DLL	C:\Windows\System32\winmmbase.dll	Microsoft Corporation
winmm.dll	MCI API DLL	C:\Windows\System32\winmm.dll	Microsoft Corporation
WindowsCodecs.dll	Microsoft Windows Codecs Library	C:\Windows\System32\WindowsCodecs.dll	Microsoft Corporation
windows.storage.dll	API αποθήκευσης Microsoft WinRT	C:\Windows\System32\windows.storage.dll	Microsoft Corporation
version.dll	Version Checking and File Installati...	C:\Windows\System32\version.dll	Microsoft Corporation
uxtheme.dll	Microsoft UxTheme Library	C:\Windows\System32\uxtheme.dll	Microsoft Corporation
usp10.dll	Uniscribe Unicode script processor	C:\Windows\System32\usp10.dll	Microsoft Corporation
userenv.dll	Userenv	C:\Windows\System32\userenv.dll	Microsoft Corporation
user32.dll.mui	Multi-User Windows USER API Cli...	C:\Windows\System32\el-GR\user32.dll.mui	Microsoft Corporation
user32.dll	Multi-User Windows USER API Cli...	C:\Windows\System32\user32.dll	Microsoft Corporation
twinapi.appcore.dll	twinapi.appcore	C:\Windows\System32\twinapi.appcore.dll	Microsoft Corporation
tiptsf.dll	Πλαίσιο υπηρεσιών κειμένου πλη...	C:\Program Files\Common Files\microsoft shared\ink\tiptsf.dll	Microsoft Corporation
System.XML.dll	.NET Framework	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__b77a5c561934e089\System.XML.dll	Microsoft Corporation
System.Windows.F...	.NET Framework	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Windows.Forms.resources\v4.0_4.0.0.0_el_b77a5c561934e089\System.Windows.Forms.resources.dll	Microsoft Corporation
System.Windows.F...	.NET Framework	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Windows.Forms\v4.0_4.0.0.0__b77a5c561934e089\System.Windows.Forms.dll	Microsoft Corporation
System.Security.dll	System.Security.dll	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Security\v4.0_4.0.0.0__b03f5f7f11d50a3a\System.Security.dll	Microsoft Corporation
System.Drawing.dll	.NET Framework	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Drawing\v4.0_4.0.0.0__b03f5f7f11d50a3a\System.Drawing.dll	Microsoft Corporation
System.dll	.NET Framework	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77a5c561934e089\System.dll	Microsoft Corporation
System.Configuratio...	System.Configuration.dll	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Configuration\v4.0_4.0.0.0__b03f5f7f11d50a3a\System.Configuration.dll	Microsoft Corporation
StaticCache.dat		C:\Windows\Fonts\StaticCache.dat	
sspicli.dll	Security Support Provider Interface	C:\Windows\System32\sspicli.dll	Microsoft Corporation
SortDefault.nls		C:\Windows\Globalization\Sorting\SortDefault.nls	
shlwapi.dll	Βιβλιοθήκη βοηθημάτων Shell Lig...	C:\Windows\System32\shlwapi.dll	Microsoft Corporation
shell32.dll	Κοινόχρηστο Dll για το κέλυφος ...	C:\Windows\System32\shell32.dll	Microsoft Corporation
SHCore.dll	SHCORE	C:\Windows\System32\SHCore.dll	Microsoft Corporation
secur32.dll	Security Support Provider Interface	C:\Windows\System32\secur32.dll	Microsoft Corporation
sechost.dll	Host for SCM/SDDL/LSA Lookup ...	C:\Windows\System32\sechost.dll	Microsoft Corporation
rsaenh.dll	Microsoft Enhanced Cryptographic...	C:\Windows\System32\rsaenh.dll	Microsoft Corporation
rpcrt4.dll	Χρόνος εκτέλεσης κλήσης απομ...	C:\Windows\System32\rpcrt4.dll	Microsoft Corporation
riched20.dll	Rich Text Edit Control, v3.1	C:\Windows\System32\riched20.dll	Microsoft Corporation
R00000000000d.clb		C:\Windows\Registration\R00000000000d.clb	
profapi.dll	User Profile Basic API	C:\Windows\System32\profapi.dll	Microsoft Corporation
powrprof.dll	DLL βοηθού προφίλ ενέργειας	C:\Windows\System32\powrprof.dll	Microsoft Corporation
oleaut32.dll		C:\Windows\System32\oleaut32.dll	Microsoft Corporation
ole32.dll	Microsoft OLE for Windows	C:\Windows\System32\ole32.dll	Microsoft Corporation
nvinitx.dll	NVIDIA shim initialization dll, Versio...	C:\Windows\System32\nvinitx.dll	NVIDIA Corporation
ntdll.dll	DLL επιπέδου NT	C:\Windows\System32\ntdll.dll	Microsoft Corporation
msvcrt.dll	Windows NT CRT DLL	C:\Windows\System32\msvcrt.dll	Microsoft Corporation
msvcr120_clr0400.dll	Microsoft® C Runtime Library	C:\Windows\System32\msvcr120_clr0400.dll	Microsoft Corporation
msls31.dll	Microsoft Line Services library file	C:\Windows\System32\msls31.dll	Microsoft Corporation
msctf.dll	DLL διακομιστή MSCTF	C:\Windows\System32\msctf.dll	Microsoft Corporation
mscorrc.dll	Πόροι χρόνου εκτέλεσης για το ...	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\el\mscorrc.dll	Microsoft Corporation
mscorlib.resources.dll	Βιβλιοθήκη κλάσεων Microsoft Co...	C:\Windows\Microsoft.NET\assembly\GAC_MSIL\mscorlib.resources\v4.0_4.0.0.0_el_b77a5c561934e089\mscorlib.resources.dll	Microsoft Corporation
mscorlib.ni.dll	Microsoft Common Language Runt...	C:\Windows\assembly\NativeImages_v4.0.30319_64\mscorlib\88c60510e9a0b668a5a8e270dba0dfcf\mscorlib.ni.dll	Microsoft Corporation
mscoreei.dll	Microsoft .NET Runtime Execution...	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\mscoreei.dll	Microsoft Corporation
mscoree.dll	Microsoft .NET Runtime Execution...	C:\Windows\System32\mscoree.dll	Microsoft Corporation
micross.ttf		C:\Windows\Fonts\micross.ttf	
locale.nls		C:\Windows\System32\locale.nls	
KernelBase.dll	Windows NT BASE API Client DLL	C:\Windows\System32\KernelBase.dll	Microsoft Corporation
kernel32.dll	Windows NT BASE API Client DLL	C:\Windows\System32\kernel32.dll	Microsoft Corporation
kernel.appcore.dll	AppModel API Host	C:\Windows\System32\kernel.appcore.dll	Microsoft Corporation
KeePassLibC64.dll	KeePass Library 1.29	C:\Program Files (x86)\KeePass Password Safe 2\KeePassLibC64.dll	Dominik Reichl
KeePass.XmlSerializ...		C:\Program Files (x86)\KeePass Password Safe 2\KeePass.XmlSerializers.dll	
KeePass.exe	KeePass	C:\Program Files (x86)\KeePass Password Safe 2\KeePass.exe	Dominik Reichl
imm32.dll	Multi-User Windows IMM32 API Cli...	C:\Windows\System32\imm32.dll	Microsoft Corporation
GdiPlus.dll	Microsoft GDI+	C:\Windows\WinSxS\amd64_microsoft.windows.gdiplus_6595b64144ccf1df_1.1.10240.16603_none_89ad01f9af1a159\GdiPlus.dll	Microsoft Corporation
gdi32.dll	GDI Client DLL	C:\Windows\System32\gdi32.dll	Microsoft Corporation
ExplorerFrame.dll	ExplorerFrame	C:\Windows\System32\ExplorerFrame.dll	Microsoft Corporation
dxgi.dll	DirectX Graphics Infrastructure	C:\Windows\System32\dxgi.dll	Microsoft Corporation
DWrite.dll	Υπηρεσίες τυπογραφίας Microso...	C:\Windows\System32\DWrite.dll	Microsoft Corporation
dwmapi.dll	API διαχείρισης παραθύρων επι...	C:\Windows\System32\dwmapi.dll	Microsoft Corporation
devobj.dll	Device Information Set DLL	C:\Windows\System32\devobj.dll	Microsoft Corporation
dcomp.dll	Microsoft DirectComposition Library	C:\Windows\System32\dcomp.dll	Microsoft Corporation
DataExchange.dll	Data exchange	C:\Windows\System32\DataExchange.dll	Microsoft Corporation
d3d11.dll	Direct3D 11 Runtime	C:\Windows\System32\d3d11.dll	Microsoft Corporation
d2d1.dll	Βιβλιοθήκη D2D της Microsoft	C:\Windows\System32\d2d1.dll	Microsoft Corporation
cryptsp.dll	Cryptographic Service Provider API	C:\Windows\System32\cryptsp.dll	Microsoft Corporation
cryptbase.dll	Base cryptographic API DLL	C:\Windows\System32\cryptbase.dll	Microsoft Corporation
comctl32.dll	Βιβλιοθήκη στοιχείων ελέγχου εμ...	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_5.82.10240.16384_none_0212ec7eba871e86\comctl32.dll	Microsoft Corporation
comctl32.dll	Βιβλιοθήκη στοιχείων ελέγχου εμ...	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.10240.16384_none_f41f7b285750ef43\comctl32.dll	Microsoft Corporation
combase.dll	Microsoft OLE για Windows	C:\Windows\System32\combase.dll	Microsoft Corporation
clrjit.dll	Microsoft .NET Runtime Just-In-Ti...	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clrjit.dll	Microsoft Corporation
clr.dll	Microsoft .NET Runtime Common ...	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clr.dll	Microsoft Corporation
clbcatq.dll	COM+ Configuration Catalog	C:\Windows\System32\clbcatq.dll	Microsoft Corporation
cfgmgr32.dll	Configuration Manager DLL	C:\Windows\System32\cfgmgr32.dll	Microsoft Corporation
C_1252.NLS		C:\Windows\System32\C_1252.NLS	
bcryptprimitives.dll	Windows Cryptographic Primitives ...	C:\Windows\System32\bcryptprimitives.dll	Microsoft Corporation
bcrypt.dll	Βιβλιοθήκη προκαταρκτικών κρυπ...	C:\Windows\System32\bcrypt.dll	Microsoft Corporation
avghooka.dll	AVG Hook Library	C:\Program Files (x86)\AVG\Av\avghooka.dll	AVG Technologies CZ, s.r.o.
advapi32.dll	Εξελιγμένο βασικό API των Wind...	C:\Windows\System32\advapi32.dll	Microsoft Corporation
Accessibility.ni.dll	.NET Framework	C:\Windows\assembly\NativeImages_v4.0.30319_64\Accessibility\56c5aa1d889f4b433e2e26b747489d1f\Accessibility.ni.dll	Microsoft Corporation
~FontCache-Syste...		C:\Windows\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-System.dat	
~FontCache-S-1-5-...		C:\Windows\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-S-1-5-21-1626947427-3442749095-2226212190-1001.dat	
~FontCache-FontFa...		C:\Windows\ServiceProfiles\LocalService\AppData\Local\FontCache\~FontCache-FontFace.dat	

CPU Usage: 16.37% | Commit Charge: 29.57% | Processes: 95 | Physical Usage: 36.68%

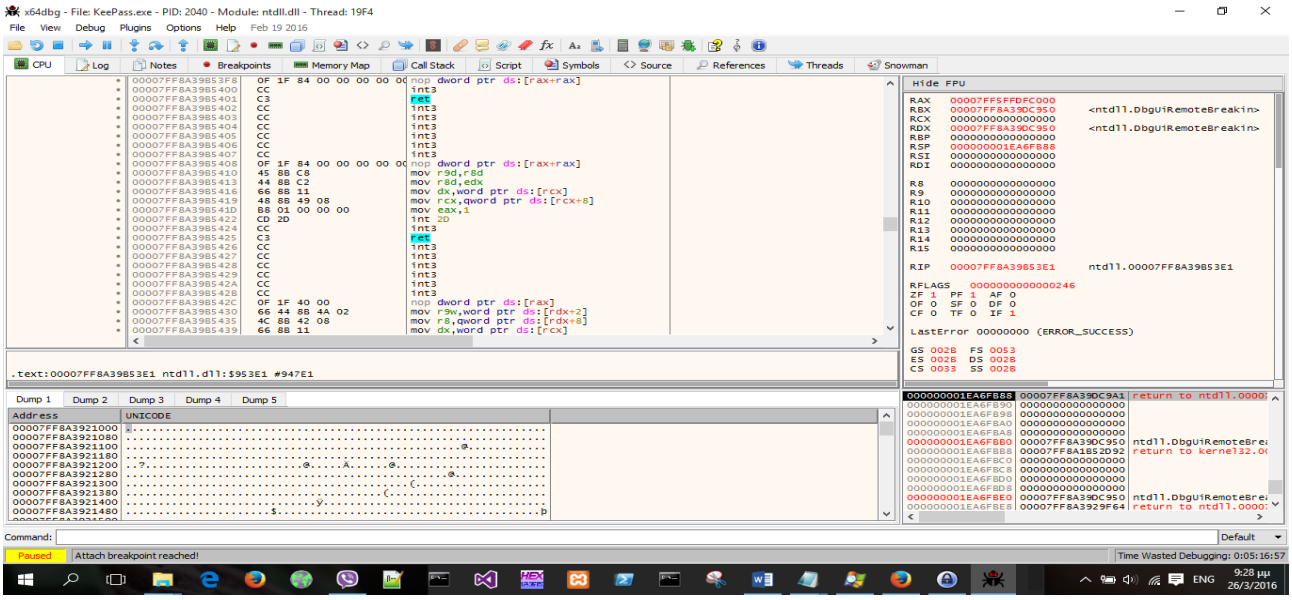
Εικόνα 14: Τα Handles που χρεώνονται στην διεργασία KeePass όπως τα εντοπίζει ο Process Explorer

- *psapi.dll* : Η βιβλιοθήκη αυτή δεν υπήρχε πριν την εκτέλεση του *KeeFarce.* Παρακολουθώντας την εφαρμογή *KeePass* είναι εφικτό να ανιχνεύσουμε κάποια ύποπτη ενέργεια αφού η βιβλιοθήκη αυτή χρησιμοποιείται για να καθίσταται πιο

εύκολη η απόκτηση πληροφοριών σχετικά με διεργασίες και οδηγούς συσκευών [26]. Συνεπώς, αφού η διεργασία μας δεν συσχετίζεται με κάποια άλλη διεργασία, ούτε επιχειρεί με οποιοδήποτε τρόπο να συσχετιστεί, δεν φαίνεται λογική η φόρτωση της σχετικής βιβλιοθήκης. Η φόρτωση υποδεικνύει πιθανή κακόβουλη ενέργεια.

- *mscordacwks.dll* : Η βιβλιοθήκη αυτή προσφέρει το *Data Access Component (DAC)*, το οποίο προσφέρει στην επέκταση χρήσης ενός *native (C, C++)* αποσφαλματωτή την δυνατότητα να διερμηνεύει τις εσωτερικές δομές δεδομένων μίας *.NET* εφαρμογής *managed (C#)* κώδικα. Συνεπώς, η φόρτωσή της στο χώρο μνήμης της εφαρμογής *KeePass* υποδεικνύει πως μία άλλη εφαρμογή προσπαθεί να τρέξει σε παράλληλο *Runtime,* γεγονός ύποπτο και συνηθισμένο στην *dll Injection* τεχνική προσβολής.
- *Microsoft.Diagnostics.Runtime.dll* : Η βιβλιοθήκη αυτή ονομάζεται και *ClrMD* και επιτρέπει την αυτοματοποίηση διαδικασιών επιτήρησης διεργασιών και πρόσβαση σε πληροφορίες αποσφαλμάτωσης. Η φόρτωση αυτής της βιβλιοθήκης είναι εξαιρετικά ύποπτη καθώς επιτρέπει αποσφαλμάτωσης του *KeePass,* στα πλαίσια της οποίας η διεργασία αγκιστρώνεται από μία άλλη (εδώ *KeeFarce*), η οποία μπορεί να μπορεί να διαβάσει και την μνήμη της πρώτης.

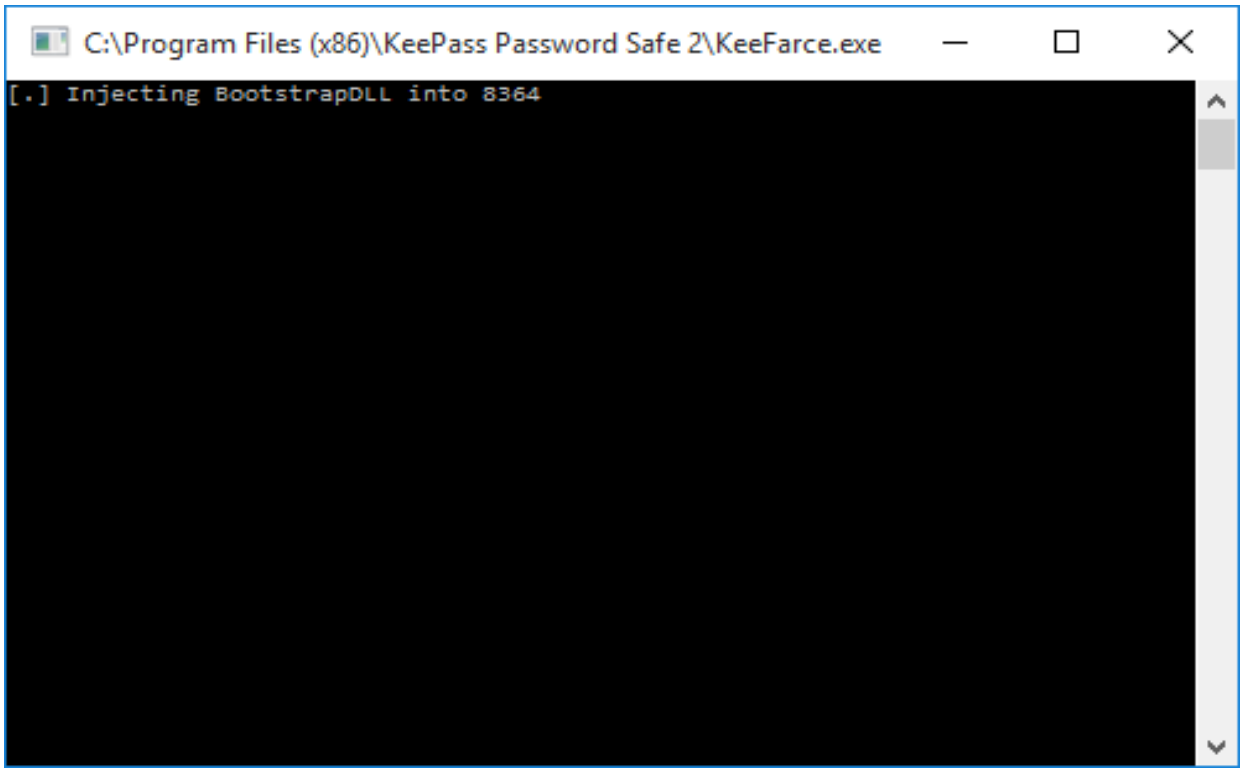
Με τις παραπάνω ενδείξεις μπορεί πολύ γρήγορα ένας αναλυτής να εγείρει την προσοχή του, εξετάζοντας προσεκτικότερα το σύστημά του σχετικά με την εφαρμογή *KeePass,* καθώς φαίνεται πως μία παράνομη οντότητα προσπαθεί να συσχετιστεί με αυτή. Κατόπιν, συσχετισμού των εν λόγω ενδείξεων με τις διαδρομές δίσκου στην οποίες βρίσκονται οι δυναμικές βιβλιοθήκες, ο αναλυτής προχωρά στην εξέταση των αρχείων που του φαίνονται



Εικόνα 15: Αποσφαλματωτής x64dbg

ύποπτα. Αυτά δεν είναι άλλα από τα 4 αρχεία που αναλύσαμε παραπάνω. Μια άλλη λύση για τον εντοπισμό ζωντανά κακόβουλου λογισμικού που σχετίζεται με μία συγκεκριμένη, όμως, διεργασία είναι ο αποσφαλματωτής *x64dbg* [27] (Εικόνα 15).

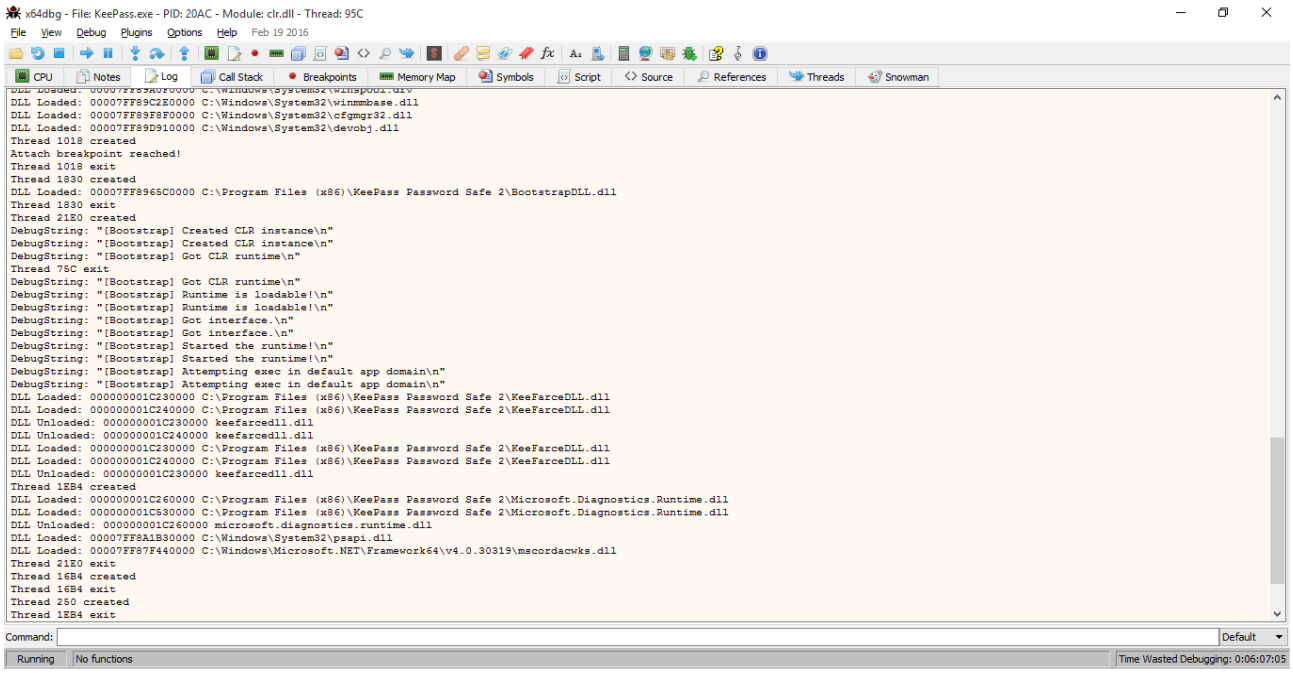
Ο συγκεκριμένος αποσφαλματωτής έχει την δυνατότητα αποσφαλμάτωσης x32 και x64 *Windows* εφαρμογών. Παρέχει πολλές λειτουργίες και πληροφορίες σχετικά με ένα αρχείο ή μία διαδικασία στην οποία προσκολλάται. Κύρια λειτουργία του είναι ο αποδομητής – *disassembler,* από τον οποίο εξάγεται οι εντολές του προγράμματος σε γλώσσα *Assembly.* Κατόπιν προσκόλλησης σε μία διεργασία παρατηρούνται τα εξής δύο φαινόμενα. Αρχικά λόγω της αποσφαλμάτωσης δεν μπορούμε να διαχειριστούμε την εφαρμογή μέχρι να αποκολληθεί ο *x64dbg* από αυτήν. Στη συνέχεια, και αφού γνωρίζουμε ότι οποιαδήποτε εξωτερική ενέργεια που αφορά την εφαρμογή θα εντοπιστεί από τον αποσφαλματωτή και θα παγώσει, παρατηρηρούμε ότι αφού εκτελεσθεί το *KeeFarce*, η ενέργεια του *dll Injection* εντοπίζεται από τον *x64dbg* και παγώνει (Εικόνα 16).



Εικόνα 16: Η εκτέλεση του KeeFarce παγώνει

Έτσι όχι μόνο εντοπίζουμε την προσπάθεια κάποιας άλλης διεργασίας να προσκολληθεί στην εφαρμογή μας αλλά αφού την εντοπίσουμε, μπορούμε να μελετήσουμε βήμα βήμα την

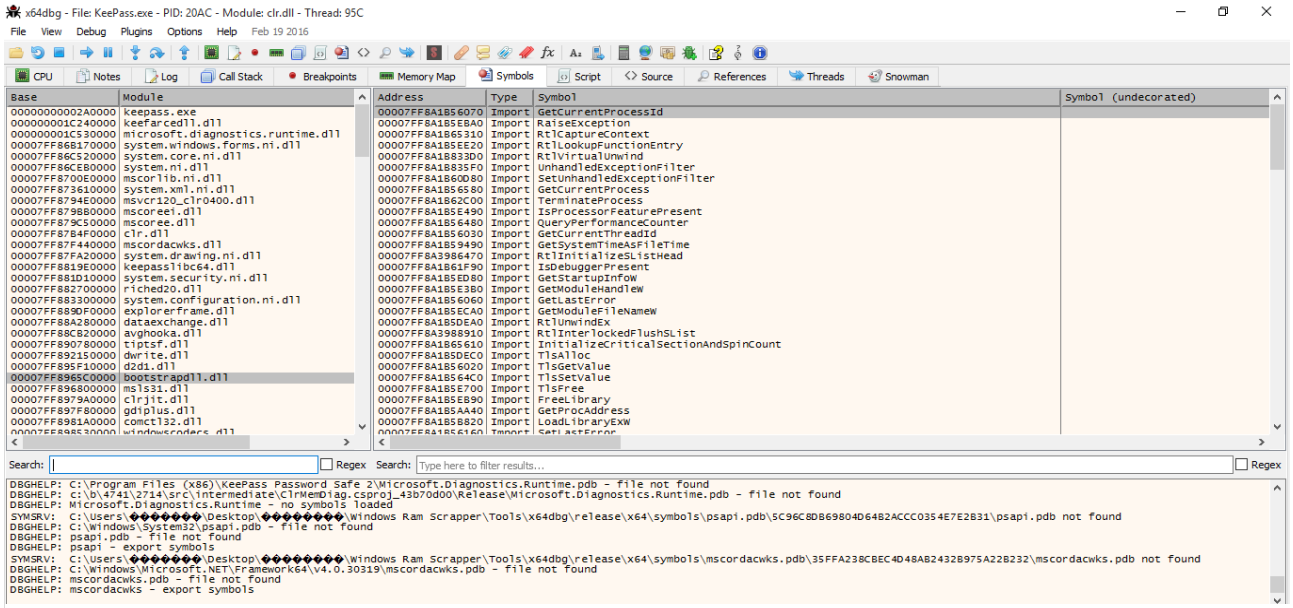
κακόβουλή εκτέλεσή της. Η συγκεκριμένη δυνατότητα δίνεται για κακόβουλα λογισμικά που δεν ανιχνεύουν την προυπάρχουσα προσκόλληση άλλων διεργασιών στην διεργασία – θύμα. Επίσης, εξετάζοντας το πεδίο Log του *x64dbg* φαίνονται όλες οι ενέργειες που σχετίζονται με την διεργασία (Εικόνα 17).



Εικόνα 17: Αρχείο Log της εκτέλεσης του KeePass

Στην συγκεκριμένη περίπτωση, βλέπουμε όλες τις βιβλιοθήκες *dll*, οι οποίες φορτώθηκαν στην εφαρμογή. Γίνεται, λοιπόν, ξεκάθαρο ότι πρόκειται για *dll* Injection καθώς τα ονόματα των βιβλιοθηκών *Bootstrap.dll* και *KeeFarceDLL.dll* δεν παραπέμπουν σε κάτι γνώριμο. Οι παρατηρήσεις που αναφέρθηκαν προηγουμένως για τις βιβλιοθήκες ισχύουν και εδώ.

Μία επιπλέον λειτουργία είναι η εμφάνιση των συναρτήσεων που περιέχονται μέσα σε κάθε



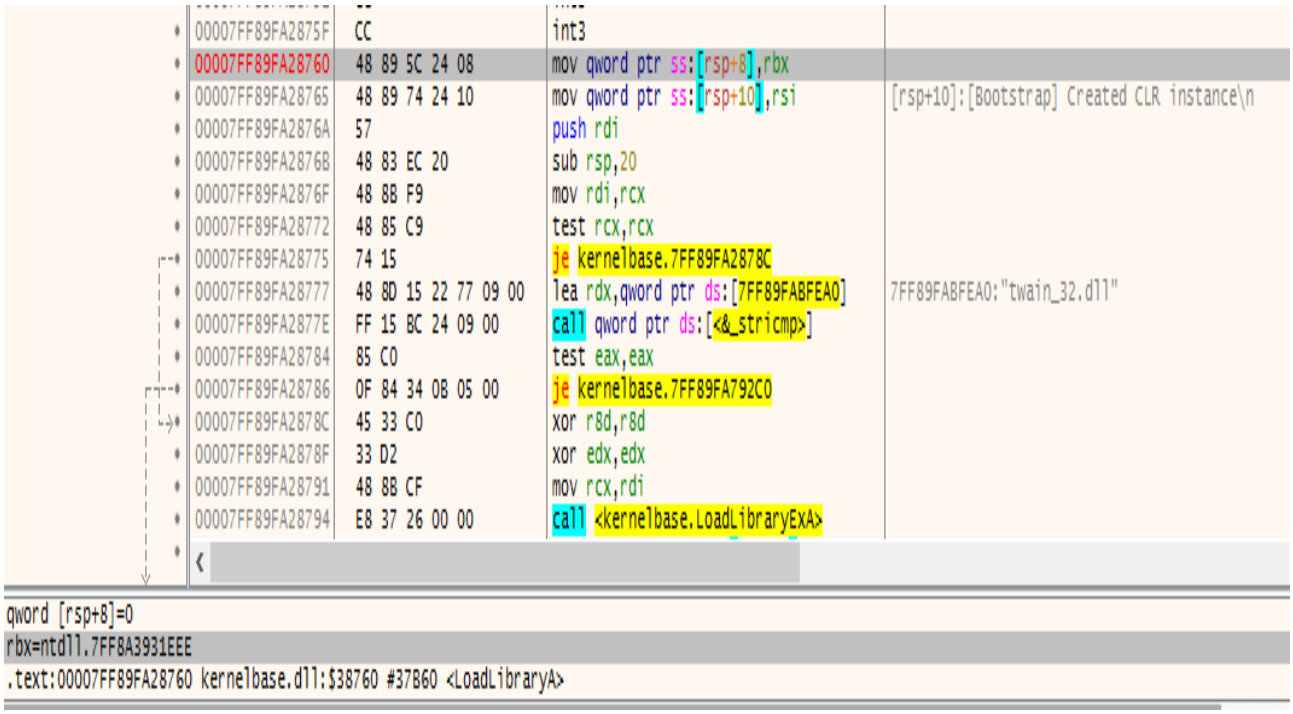
Εικόνα 18: Συναρτήσεις που χρησιμοποιήθηκαν

βιβλιοθήκη, δυνατότητα που μας βοηθάει να αντιληφθούμε την υπόσταση κάθε βιβλιοθήκης και να αναγνωρίσουμε ύποπτα σημάδια (Εικόνα 18).

Επίσης, βλέποντας τον Assembly κώδικα που παράγεται από τον αποσυναρμολογητή, φαίνεται μία αρκετά γνωστή τεχνική που εφαρμόζεται για την επίτευξη του dll Injection. Η τεχνική αφορά σε 2 βασικά βήματα:

- Εύρεση μίας συνάρτησης η οποία υπάρχει σε μία βιβλιοθήκη που φορτώνεται με μεγάλη πιθανότητα στα περισσότερα προγράμματα. Στην περίπτωση μας αυτή είναι η *LoadLibraryA* της βιβλιοθήκης system32.dll.
- Τροφοδότηση της συνάρτησης *LoadLibraryA* με την διαδρομή δίσκου της κακόβουλης δυναμικής βιβλιοθήκης που θέλει κάποιος να φορτωθεί. Ωστόσο, το όνομα της *dll* που θα φορτωθεί θα πρέπει να βρίσκεται ήδη μέσω στο χώρο διευθύνσεων της διεργασίας – θύματος. Άρα καλείται αρχικά η συνάρτηση *VirtualAllocEx* η οποία δημιουργεί χώρο μνήμης μέσα στο χώρο διευθύνσεων της διεργασίας – θύματος, με μέγεθος όσο το όνομα της *dll* που θέλει ο κακόβουλος να φορτώσει. Στη συνέχεια χρησιμοποιείται η συνάρτηση *WriteProcessMemory* για την εγγραφή της διαδρομής δίσκου της dll στην μνήμη στην διεργασία – θύμα.
- Χρήση της συνάρτησης *CreateRemoteThread* για την κλήση της συνάρτησης *LoadLibraryA* μέσα από το χώρο διευθύνσεων της διεργασίας – θύματος.

Στην Εικόνα 19 φαίνεται η κλήση της συνάρτησης *LoadLibraryA* και η φόρτωση της συνάρτησης Bootstrap.dll

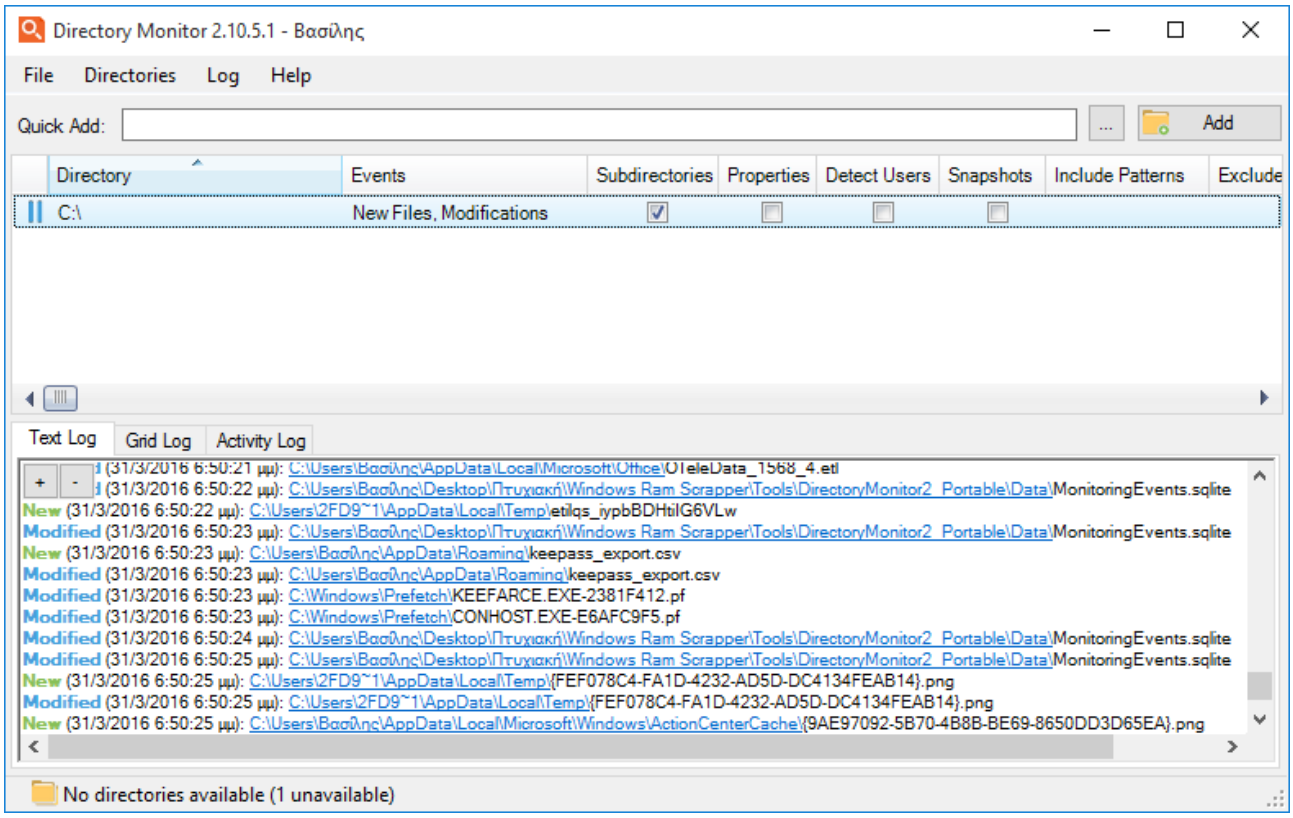


Εικόνα 19: Η συνάρτηση LoadLibraryA στον αποσυναρμολογητή

Επίσης, είναι δυνατόν μέσω της καρτέλας *symbols* να εντοπίσουμε τις συναρτήσεις *VirtualAllocEx, WriteProcessMemory, CreateRemoteThread* και ανατρέχοντας στον αποσφαλματωτή να βρούμε σε ποιο σημείο της εκτέλεσης βρίσκονται.

5.2.2.1.2 Παρακολούθηση του Σκληρού Δίσκου

Τέλος, εντοπισμός δραστηριότητας ύποπτης υποδηλώνεται και στον σκληρό δίσκο εκτός από την μνήμη. Στην ανάλυσή μας έχουμε επικεντρωθεί εξολοκλήρου στην ανάλυση της μνήμης, χωρίς αυτό σημαίνει πως δεν παρακολουθούμε όλες τις βαθμίδες του λειτουργικού συστήματος· απλά δεν χρησιμοποιούμε άλλες τεχνικές εποπτείας άλλων μέσων. Παρόλα αυτά, αξίζει να αναφέρουμε μία περίπτωση παρακολούθησης της δραστηριότητας μέσω μίας εφαρμογής που ονομάζεται *Directory Monitor.* Με την εφαρμογή μπορούμε να εποπτεύουμε όλες τις *I/O* διεργασίες του συστήματος και για όποιον φάκελο θέλουμε συγκεκριμένα. Στην περίπτωσή μας, επιλέξαμε να εποπτεύουμε όλο το δίσκο *C.* Αυτό έχει σαν άμεσο αποτέλεσμα να μπορούμε να διακρίνουμε οποιαδήποτε δραστηριότητα δίσκου και αν πραγματοποιείται.



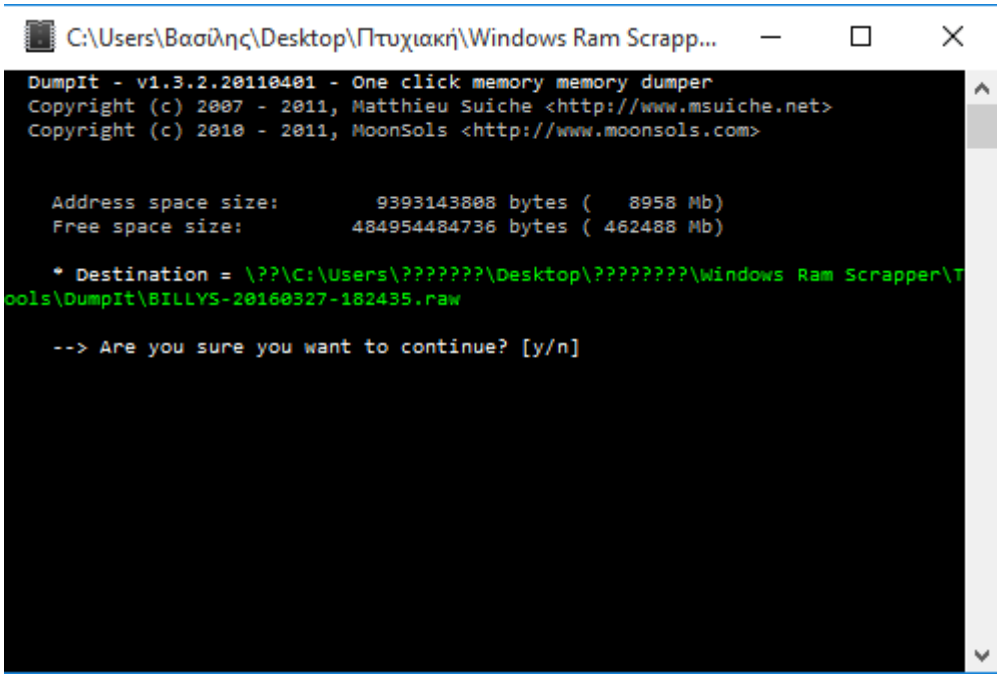
Εικόνα 20: Ενέργειες Σκληρού Δίσκου στο Directory Monitor

Στη διαδικασία παρακολούθησης εντοπίσαμε ότι την στιγμή εκτέλεσης του *KeeFarce* δημιουργήθηκε ένα αρχείο *keepass_export.csv* και αμέσως μετά τροποποιήθηκε. Αυτό υποδεικνύει αρκούντως ύποπτη δραστηριότητα και η πληροφορία αυτή θα μας βοηθήσει στη συνέχεια της μελέτης μας.

5.2.2.2 Εντοπισμός σε Στιγμιότυπο Μνήμης

Ο εντοπισμός του κακόβουλου λογισμικού σε στιγμιότυπο μνήμης έγινε με την χρήση του εργαλείου *Volatility.* Η χρήση, ωστόσο, του εργαλείου δεν περιορίζεται στον εντοπισμό αλλά συνεχίζει και στην εκπόνηση αποτελεσμάτων στα πλαίσια της ανάλυσης κακόβουλου λογισμικού. Στο σημείο αυτό, οφείλεται να αναφερθεί πως οι διαδικασίες εντοπισμού και ανάλυσης πολλές φορές είναι αδιαχώρητες, καθώς πολλές ενέργειες μελέτης του συστήματος για ανίχνευση κακόβουλου λογισμικού εμπίπτουν και στην κατηγορία ανάλυσης του.

Αρχικά θα πρέπει να χρησιμοποιηθεί ένα εργαλείο για την αποτύπωση της μνήμης του συστήματος. Για τον σκοπό αυτό κυκλοφορούν στο διαδίκτυο πολλά εργαλεία με ελαφρά διαφορετικές δυνατότητες, κυρίως όσον αφορά στους τύπους αρχείων που μπορούν να παράγουν. Εμείς χρησιμοποιήσαμε το εργαλείο *Dumpit,* το οποίο αποτυπώνει όλη την μνήμη του συστήματος, την στιγμή που του δίνουμε την εντολή, σε ένα αρχείο τύπου *.RAW.* Στο σημείο αυτό θα πρέπει να αναφερθεί πως κάθε αρχείο αποτύπωσης της μνήμης έχει ένα βασικό περιορισμό. Αυτός έγκειται στο γεγονός της αδυναμίας των εργαλείων να αποτυπώσουν επακριβώς την μνήμη, καθώς κατά την διάρκεια της διαδικασίας αποτύπωσης η μνήμη *RAM* συνεχίζει να μεταβάλλεται και υπάρχει η πιθανότητα κάποια λειτουργία να μείνει ημιτελής, αν δεν προλάβει και ολοκληρωθεί. Για παράδειγμα, μία σελίδα της εικονικής μνήμης μίας διεργασίας μπορεί διαγραφεί πριν αποτυπωθεί, με αποτέλεσμα στην χειρότερη περίπτωση να καταλήξουμε με ένα κατεστραμμένο αρχείο αποτύπωσης μνήμης. [28]



Εικόνα 21: Αποτύπωση Μνήμης με το εργαλείο Dumpit

Χάριν ποικιλομορφίας και απόλυτης συμβατότητας με την λειτουργία του κακόβουλου λογισμικού *KeeFarce,* το οποίο έχει ελεγχθεί ότι δουλεύει σε *Windows 8* και *Windows 8.1,* η αποτύπωση μνήμης έγινε μέσα σε εικονική μηχανή και σε λειτουργικό *Windows 8.1 pro.* Πρακτικά παρατηρήθηκε πως δεν υπάρχει καμία διαφορά στον τρόπο λειτουργίας και συμπεριφοράς του εν λόγω κακόβουλου λογισμικού στα λειτουργικά *Windows 8.1 pro* και *Windows 10.*

Κατόπιν απόκτησης του αρχείου *.Raw* με την μνήμη του λειτουργικού συστήματος, διενεργήθηκαν κάποιες βασικές εντολές του προγράμματος *Volatility* για την μελέτη της ύπαρξης κακόβουλης οντότητας. Αυτές είναι:

- Έρευνα για την εύρεση του κατάλληλου προφίλ για την διενέργεια της ανάλυσης. Η εν λόγω ενέργεια είναι απαραίτητη κάθε φορά που ξεκινάμε την ανάλυση σε ένα οποιοδήποτε αρχείο αποτύπωσης. Η χρησιμότητα της έγκειται στο γεγονός της απόκτησης πολύτιμών πληροφοριών για το αρχείο που μελετάμε (Εικόνα 22).

```
bill@bill-Inspiron-3542:/media/bill/Shared_Backup Partition/windows7_sandbox_sha
red$ volatility -f BILL_PC-20160303-123816_VirtualWindows8.1pro_AfterKeeFArce.ra
w --profile=Win8SP1x64 imageinfo
Volatility Foundation Volatility Framework 2.5
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win8SP1x64
                     AS Layer1 : AMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (/media/bill/Shared_Backup Par
tition/windows7_sandbox_shared/BILL_PC-20160303-123816_VirtualWindows8.1pro_Afte
rKeeFArce.raw)
                      PAE type : No PAE
                           DTB : 0x1a7000L
                          KDBG : 0xf80067b24a30L
          Number of Processors : 1
     Image Type (Service Pack) : 0
                KPCR for CPU 0 : 0xfffff80067b7f000L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2016-03-03 12:38:25 UTC+0000
     Image local date and time : 2016-03-03 14:38:25 +0200
```

Εικόνα 22: Εύρεση στοιχείων στιγμιότυπου μνήμης με τον εργαλείο Volatillity

Βλέπουμε ότι το αρχείο αποτύπωσης μνήμης προέρχεται από λειτουργικό σύστημα Windows 8.1 αρχιτεκτονικής 64 *bits.* Επίσης βλέπουμε ότι δεν χρησιμοποιείται σύστημα *PAE* και ότι έχουμε μόνο ένα πυρήνα (μονοπύρηνη εικονική μηχανή). Τέλος βλέπουμε τα χρονικά χαρακτηριστικά του αρχείου. Για την καλύτερη και γρηγορότερη εκτέλεση των εντολών ανάλυσης σε ένα αρχείο, εκτός από τον προφίλ του λειτουργικού που θα πρέπει να ορίζουμε, θα πρέπει να ξέρουμε και να ορίζουμε την παράμετρο *kdgb.* Αυτή για λειτουργικό σύστημα *Windows 8.1 Windows 10* ταυτίζεται με την παράμετρο KdCopyDataBlock, η οποία είναι μία εικονική διεύθυνση μνήμης [29], [30] (Εικόνα 23).

```
bill@bill-Inspiron-3542:/media/bill/Shared_Backup Partition/windows7_sandbox_shared$ volatility -f BILL_PC-20160303-123816_VirtualWindows8.1pro_
AfterKeeFArce.raw --profile=Win8SP1x64 kdbgscan
Volatility Foundation Volatility Framework 2.5
**************************************************
Instantiating KDBG using: Unnamed AS Win8SP1x64 (6.3.9600 64bit)
Offset (V)                    : 0xf80067b24a30
Offset (P)                    : 0x114d24a30
KdCopyDataBlock (V)           : 0xf80067a639b0
Block encoded                 : Yes
Wait never                    : 0xdd6802002c73ac63
Wait always                   : 0x1639dd9c5b40580
KDBG owner tag check          : True
Profile suggestion (KDBGHeader): Win8SP1x64
Version64                     : 0xf80067b24d90 (Major: 15, Minor: 9600)
Service Pack (CmNtCSDVersion) : 0
Build string (NtBuildLab)     : 9600.16384.amd64fre.winblue_rtm.
PsActiveProcessHead           : 0xfffff80067b3b700 (44 processes)
PsLoadedModuleList            : 0xfffff80067b559b0 (133 modules)
KernelBase                    : 0xfffff8006788e000 (Matches MZ: True)
Major (OptionalHeader)        : 6
Minor (OptionalHeader)        : 3
KPCR                          : 0xfffff80067b7f000 (CPU 0)
```

Εικόνα 23: Εντολή kdbgscan

Όλες οι παρακάτω εντολές έχουν ορισθεί με την εξής μορφή: *volatility –f «αρχείο προς εξέταση» --profile=WinSP1x64 –kdbg=0xf80067a639b0 plugin.*

- Καταγραφή όλων των διεργασιών που εκτελούνταν στο σύστημα πριν την αποτύπωση. Η καταγραφή έγινε με την εντολή *pslist* η οποία φέρνει σαν αποτέλεσμα όλες τις διεργασίες οι οποίος τρέχουν στο σύστημα την ώρα της καταγραφής.

```
bill@bill-Inspiron-3542:/media/bill/Shared_Backup Partition/windows7_sandbox_shared$ volatility -f BILL_PC-20160303-123816_VirtualWindows8.1pro_
AfterKeeFArce.raw --profile=Win8SP1x64 --kdbg=0xf80067a639b0 pslist
Volatility Foundation Volatility Framework 2.5
Offset(V)          Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
------------------ -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------------------------
-
0xffffe000000fd900 System                    4      0     80        0 ------      0 2016-03-03 12:32:03 UTC+0000

0xffffe000014f4500 smss.exe                256      4      2        0 ------      0 2016-03-03 12:32:03 UTC+0000

0xffffe00001434080 csrss.exe               332    324      8        0      0      0 2016-03-03 12:32:12 UTC+0000

0xffffe0000013f080 wininit.exe             396    324      1        0      0      0 2016-03-03 12:32:13 UTC+0000

0xffffe000000e5240 csrss.exe               404    388      9        0      1      0 2016-03-03 12:32:13 UTC+0000

0xffffe00000161200 winlogon.exe            432    388      2        0      1      0 2016-03-03 12:32:13 UTC+0000

0xffffe00002303080 services.exe            496    396      5        0      0      0 2016-03-03 12:32:14 UTC+0000

0xffffe00002325080 lsass.exe               504    396      6        0      0      0 2016-03-03 12:32:14 UTC+0000

0xffffe000023e2080 svchost.exe             556    496     10        0      0      0 2016-03-03 12:32:16 UTC+0000

0xffffe0000243b500 svchost.exe             596    496      8        0      0      0 2016-03-03 12:32:20 UTC+0000

0xffffe00002480080 dwm.exe                 692    432      8        0      1      0 2016-03-03 12:32:20 UTC+0000

0xffffe000024e3080 VBoxService.ex          756    496      9        0      0      0 2016-03-03 12:32:20 UTC+0000

0xffffe0000203d900 svchost.exe             820    496     26        0      0      0 2016-03-03 12:32:22 UTC+0000

0xffffe000024cc400 svchost.exe             840    496     45        0      0      0 2016-03-03 12:32:22 UTC+0000

0xffffe0000254d900 svchost.exe             884    496     21        0      0      0 2016-03-03 12:32:22 UTC+0000

0xffffe00002549900 svchost.exe             928    496     22        0      0      0 2016-03-03 12:32:22 UTC+0000

0xffffe0000259e680 svchost.exe            1016    496     15        0      0      0 2016-03-03 12:32:22 UTC+0000
```

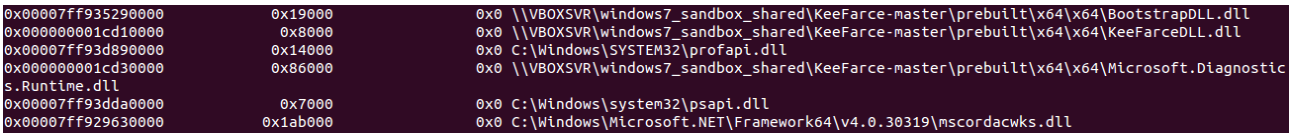
Εικόνα 24: Εντολή pslist

Παρόμοια εντολή είναι η *psscan,* η οποία προσφέρει την δυνατότητα ανίχνευσης και των διεργασιών *zombie* και διεργασιών που προσπαθούν να κρυφτούν, όπως τα *rootkits.* Από την παραπάνω εντολή εντοπίζουμε την διαδικασία που μας ενδιαφέρει και σημειώνουμε τον αναγνωριστικό ακέραιό της (Εικόνα 25).

```
0xffffe00002ed5900 KeePass.exe            1860   2008      4        0      1      0 2016-03-03 12:35:47 UTC+0000
```

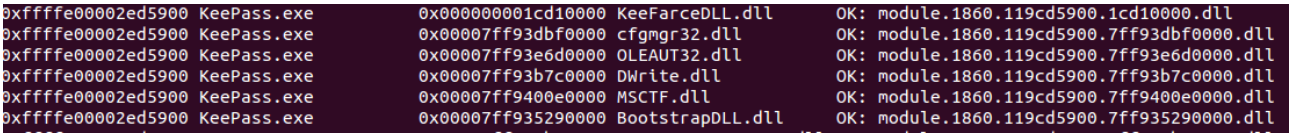
Εικόνα 25: Αναγνωριστικά στοιχεία της διεργασίας KeePass

- Στην συνέχεια έχοντας το *pid* της διεργασίας μπορούμε να ανιχνεύσουμε όλες τις δυναμικές βιβλιοθήκες που έχουν φορτωθεί σε αυτή. Η εντολή που το πραγματοποιηθεί αυτό λέγεται *dlllist.* Εξετάζοντας την λίστα που προκύπτει, ο αναλυτής ψάχνει να βρει ποιες δυναμικές βιβλιοθήκες δεν φαίνεται να ταιριάζουν στον τρόπο λειτουργίας του προγράμματος. Στην Εικόνα 26 φαίνεται πως οι *dlls Bootstrap.dll, KeeFarceDLL.dll, Microsoft.Diagnostics.Runtime.dll, psapi.dll, mscordacwks.dll,* δεν ταιριάζουν με το μοτίβο λειτουργίας της εφαρμογής *KeePass* για τους λόγους που αναφέραμε σε προηγούμενή ενότητα.



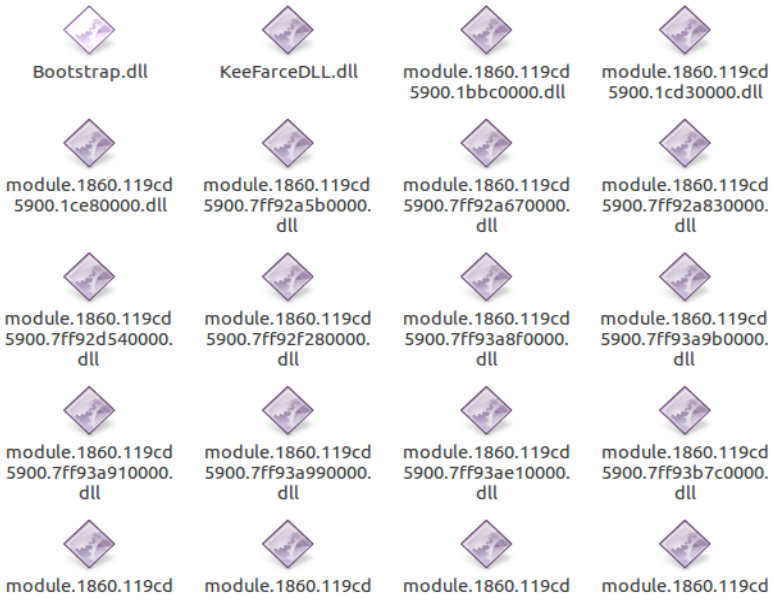
Εικόνα 26: Υπόπτως φορτωμένες δυναμικές βιβλιοθήκες

- Το επόμενο και τελευταίο βήμα είναι να μάθουμε τι κρύβεται μέσα στις βιβλιοθήκες *Bootstrap.dll* και *KeeFarceDLL.dll.* Για να το πετύχουμε αυτό χρησιμοποιούμε μία άλλη εντολή η οποία ονομάζεται *dlldump* (Εικόνα 27).



Εικόνα 27: Αποτέλεσμα εντολής dlldump

Με αυτή την εντολή μπορούμε να εξάγουμε όλα τα φορτωμένα *dll* μίας διεργασίας σε ένα αρχείο και κατόπιν να τα εξετάσουμε. Χρησιμοποιούμε την τελευταία στήλη των αποτελεσμάτων της εντολής για την ταυτοποίηση των εξαγμένων δυναμικών βιβλιοθηκών (Εικόνα 28).



Εικόνα 28: Εξαχθείσες Δυναμικές Βιβλιοθήκες

Στην συνέχεια με την χρήση της εντολής *strings –a –td «dll όνομα» > «όνομα αρχείου»* για κωδικοποίηση *Ascii* και *strings –a –td -el «dll όνομα» > «όνομα αρχείου»* για κωδικοποίηση *Unicode* μπορούμε να εξάγουμε όλα τα αλφαριθμητικά που περιέχονται στα δύο αρχεία και να τα εξετάσουμε. Από την εξέταση αυτή προκύπτουν

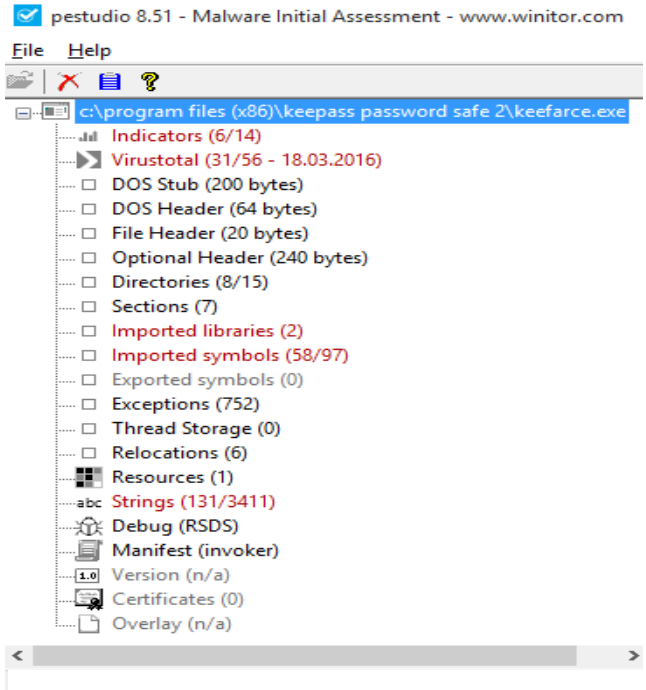
σημαντικές πληροφορίες, όπως οι συναρτήσεις που χρησιμοποιούνται από την βιβλιοθήκη, ενδείξεις που μπορούν να μας οδηγήσουν στο τρόπο με τον οποίο λειτουργούν. Στην συγκεκριμένη περίπτωση κακόβουλου λογισμικού, η ανάλυση με το εργαλείο *Volatility* μέχρι αυτό το σημείο είναι αρκετή για να αντιληφθούν πώς το λογισμικό χρησιμοποιεί την τεχνική *dll Injection.*

5.2.3 Ανάλυση δείγματος

Αφού τελειώσει το στάδιο του εντοπισμού, η μελέτη του κακόβουλου λογισμικού *KeeFarce* περνά στο στάδιο της ανάλυσης του δείγματος που έχουμε αποκτήσει. Το βήμα αυτό αποτελεί το νήμα που συνδέει την εύρεση της απειλής και της δημιουργίας κατάλληλων τειχών προστασίας. Η ανάλυση του δείγματος που διενεργήσαμε περιλαμβάνει δύο διαφορετικές προσεγγίσεις. Η πρώτη είναι η ανάλυση του PE αρχείου του κακόβουλου λογισμικού που εντοπίσαμε και εντάσσεται στο τομέα την αυτοματοποιημένης ανάλυσης. Η δεύτερη προσέγγιση αφορά στην μελέτη των αλφαριθμητικών που παράγονται από την διεργασία του κακόβουλου λογισμικού και του PE αρχείου του.

5.2.3.1 Ανάλυση του PE

Η ανάλυση του PE του κακόβουλου λογισμικού *KeeFarce* έγινε με την βοήθεια του εργαλείου *PeStudio*. Το *PeStudio* είναι μία σουίτα εργαλείων για την μελέτη PE Αρχείων. Οι



Εικόνα 29: Εργαλείο PeStudio

δυνατότητες ανάλυσης που δίνονται από το συγκεκριμένο εργαλείο φαίνονται στην (Εικόνα 29).

Αναφορικά με τα ποιο σημαντικά πεδία έχουμε ως εξής:

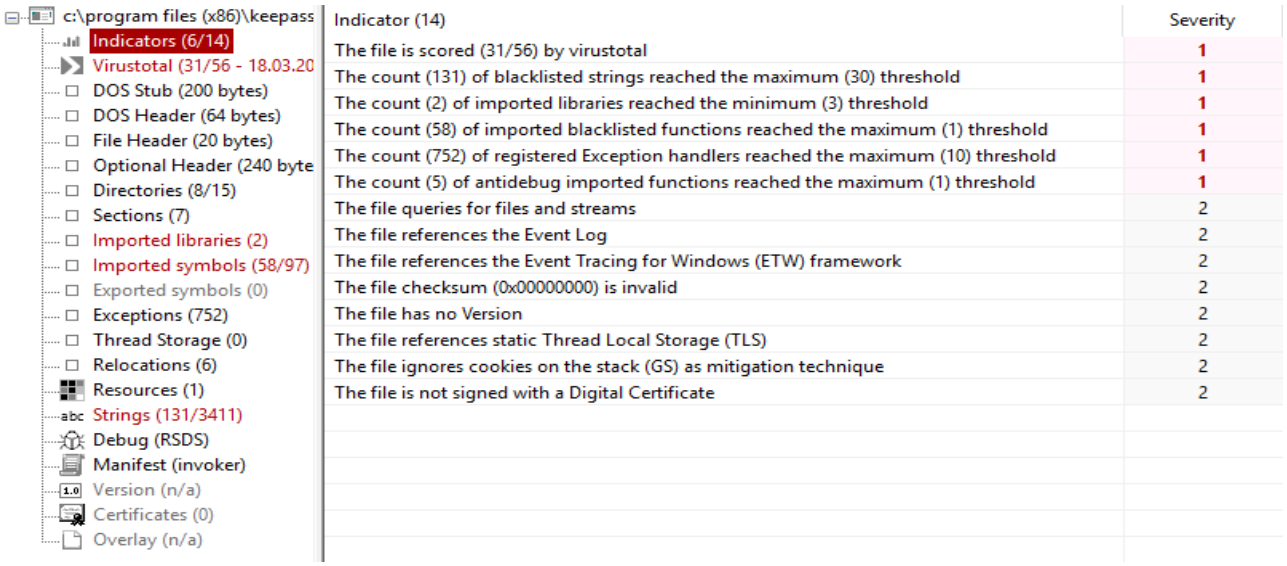
- *Indicators* : Αφορά σε ένα σύνολο διαφορετικών ενδείξεων που καθιστούν ύποπτο και επιβλαβές ένα αρχείο.
- *VirusTotal* : Μία πολύ σημαντική λειτουργία, στην οποία το εργαλείο κάνει αντιπαραβολή το υπό εξέταση αρχείο με τις Βάσεις Δεδομένων των μεγαλύτερων μηχανών ανίχνευσης, όπως το *Kaspersky* και το *McAfee,* εμφανίζοντας τον χαρακτηρισμό που έχει δοθεί από κάθε μηχανή για το συγκεκριμένο αρχείο.
- *Imported Libraries* : Μία λίστα με όλες τις βιβλιοθήκες που έχουν χρησιμοποιηθεί και ένα πεδίο με αυτές που είναι επικηρυγμένες για κακόβουλες.
- *Imported Symbols* : Μία λίστα με όλες τις συναρτήσεις που έχουν χρησιμοποιηθεί από τις εισαγόμενες βιβλιοθήκες, παρέχοντας πληροφόρηση για το ποιες από αυτές έχουν επικηρυχθεί ως πιθανές για χρήση σε κακόβουλο λογισμικό.
- *Strings* : Μία λίστα με όλα τα αλφαριθμητικά στοιχεία που περιέχονται στο υπό εξέταση αρχείο. Παράλληλα παρέχεται ο τομέας που βρέθηκε το αλφαριθμητικό (*text, data*) και μία ένδειξη του αν το αλφαριθμητικό έχει ανακηρυχθεί ως κακόβουλο ή όχι.

Με τον όρο κακόβουλο παραπάνω δεν εννοούμε ότι το συγκεκριμένο στοιχείο δεν μπορεί να αποτελεί μέρος καλόβουλου λογισμικού. Απλά για το συγκεκριμένο στοιχείο έχει αποδειχθεί ότι η ύπαρξή του συχνά συνδέεται με κακόβουλη οντότητα. Επίσης, πρέπει να αναφέρουμε ότι το εργαλείο προσφέρει και άλλες σημαντικές πληροφορίες που πολλές φορές μπορεί να αποτελέσουν και μοναδικά προσδιοριστικά για τον στιγματισμό ενός κακόβουλου λογισμικού. Ωστόσο εμείς επικεντρωνόμαστε στα πιο βασικά, τα οποία προσδιορίζουν συχνότερα μονοσήμαντα ένα κακόβουλο αρχείο. Στην συνέχεια θα αναλύσουμε τα αρχεία *KeeFarce.exe, Bootstrap.dll* και *KeeFarce.dll,* καθώς είμαστε στο στάδιο που έχουμε εντοπίσει και αναγνωρίσει τα παραπάνω ως τμήματα του κακόβουλου λογισμικού. Η αυτοματοποιημένη μελέτη των αρχείων θα αναφερθεί υπό το πρίσμα των παραπάνω δυνατοτήτων που περιγράψαμε ως πιο σημαντικές.

KeeFarce.exe

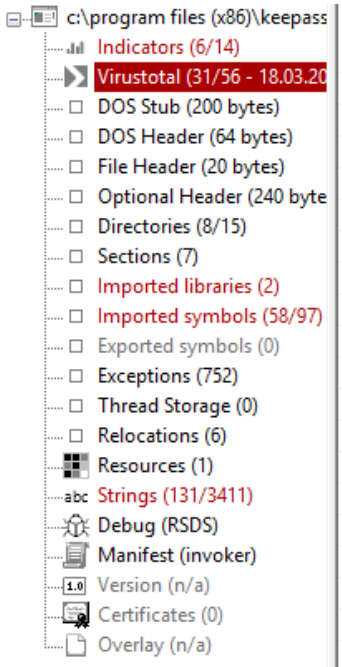
Αρχικά, όσον αφορά τους δείκτες κακόβουλης ύπαρξής, τα αποτελέσματα φαίνονται στην Εικόνα 30 (η σοβαρότητα – *severity* έχει ως μεγαλύτερη τιμή την μονάδα). Στην Εικόνα 31 φαίνεται πως έχει χαρακτηριστεί το συγκεκριμένο αρχείο από τα μεγαλύτερα *Antivirus* προγράμματα. Στην συνέχεια στην Εικόνα 32 και Εικόνα 33 φαίνονται οι βιβλιοθήκες και οι

συναρτήσεις που χρησιμοποιούνται και ποιες από αυτές είναι ύποπτες και επικηρυγμένες. Τέλος στην Εικόνα 34 φαίνονται τα δηλωμένα ως ύποπτα για κακόβουλη χρήση αλφαριθμητικά του αρχείου.



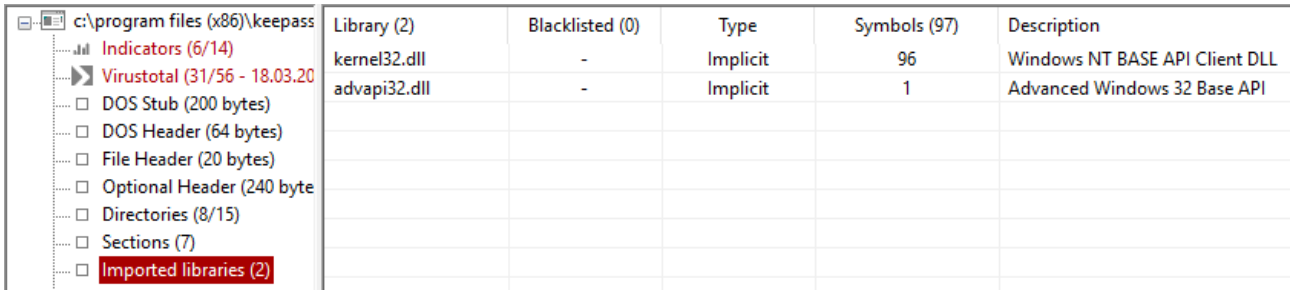
Indicator (14)	Severity
The file is scored (31/56) by virustotal	1
The count (131) of blacklisted strings reached the maximum (30) threshold	1
The count (2) of imported libraries reached the minimum (3) threshold	1
The count (58) of imported blacklisted functions reached the maximum (1) threshold	1
The count (752) of registered Exception handlers reached the maximum (10) threshold	1
The count (5) of antidebug imported functions reached the maximum (1) threshold	1
The file queries for files and streams	2
The file references the Event Log	2
The file references the Event Tracing for Windows (ETW) framework	2
The file checksum (0x00000000) is invalid	2
The file has no Version	2
The file references static Thread Local Storage (TLS)	2
The file ignores cookies on the stack (GS) as mitigation technique	2
The file is not signed with a Digital Certificate	2

Εικόνα 30: Ενδείκτες Κακόβουλης ύπαρξης

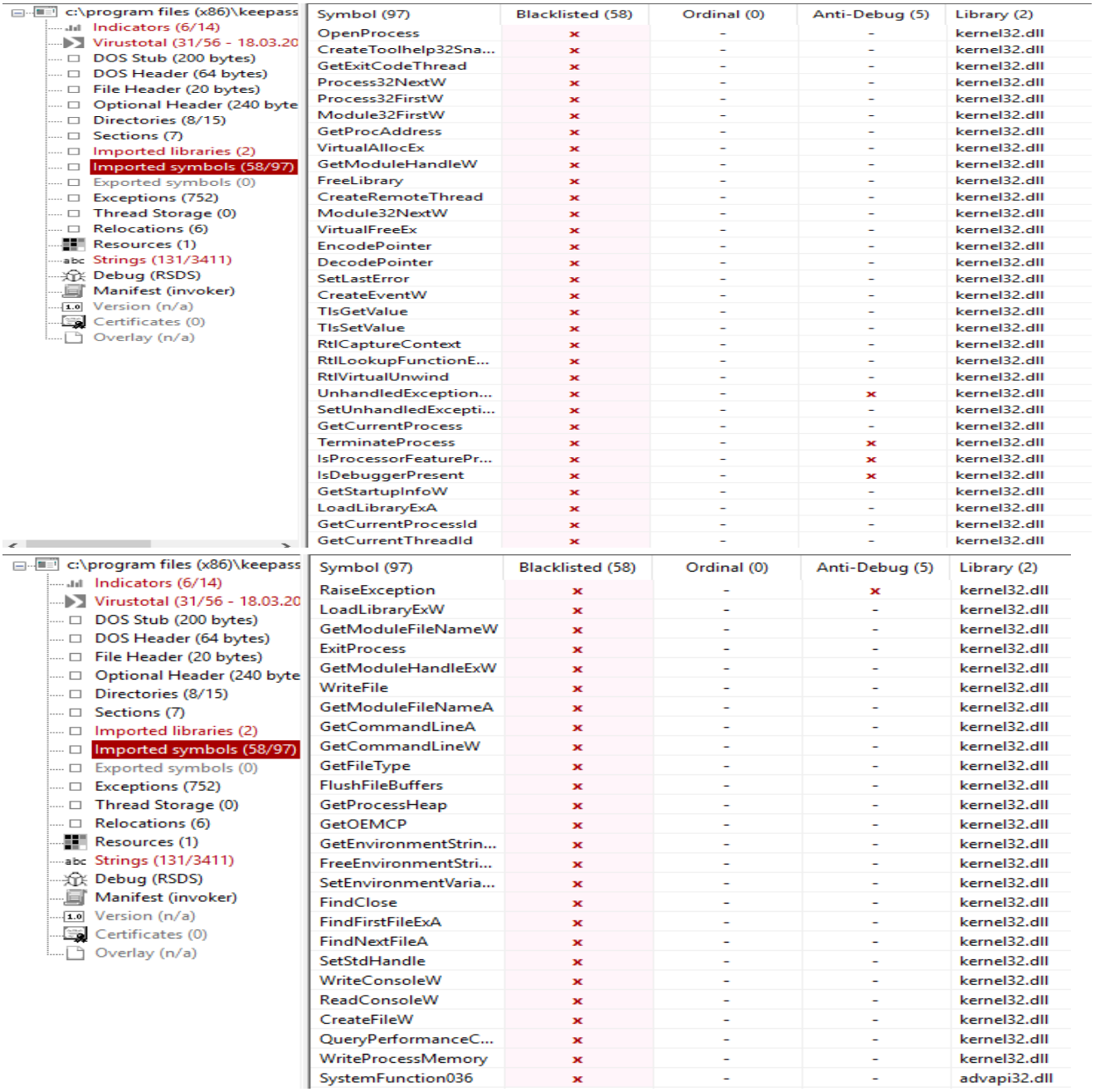


Engine (56)	Positiv (31)	Date (dd.mm.y...	Age (days)
McAfee-GW-Edition	BehavesLike.Win64.SearchSuite.dh	18.03.2016	10
McAfee	Generic HTool.h	18.03.2016	10
CAT-QuickHeal	HackTool.KeeFarce.r7 (Not a Virus)	18.03.2016	10
Microsoft	HackTool:Win32/KeeFarce	18.03.2016	10
Baidu-International	Hacktool.Win64.KeeFarce.A	18.03.2016	10
VIPRE	PassDumper	18.03.2016	10
AVware	PassDumper	18.03.2016	10
Jiangmin	RiskTool.PassDumper.a	18.03.2016	10
AegisLab	Risktool.Win64.Gen!c	18.03.2016	10
Agnitum	Riskware.PassDumper!	16.03.2016	12
NANO-Antivirus	Riskware.Win64.KeeFarce.dymwzl	18.03.2016	10
Fortinet	Riskware/PassDumper	18.03.2016	10
Symantec	SecurityRisk.KeeFarce	18.03.2016	10
TrendMicro-HouseCall	TROJ_FRS.0NA004K615	18.03.2016	10
TrendMicro	TROJ_FRS.0NA004K615	18.03.2016	10
Panda	Trj/CI.A	17.03.2016	11
MicroWorld-eScan	Trojan.Generic.15344787	18.03.2016	10
nProtect	Trojan.Generic.15344787	18.03.2016	10
BitDefender	Trojan.Generic.15344787	18.03.2016	10
Ad-Aware	Trojan.Generic.15344787	18.03.2016	10
F-Secure	Trojan.Generic.15344787	18.03.2016	10
GData	Trojan.Generic.15344787	18.03.2016	10
ALYac	Trojan.Generic.15344787	18.03.2016	10
Emsisoft	Trojan.Generic.15344787 (B)	18.03.2016	10
Arcabit	Trojan.Generic.DEA2493	18.03.2016	10
K7GW	Unwanted-Program (004d5e931)	18.03.2016	10
K7AntiVirus	Unwanted-Program (004d5e931)	18.03.2016	10
Cyren	W64/Trojan.MTSV-2742	18.03.2016	10
Avast	Win32:KFarce-A [Trj]	18.03.2016	10

Εικόνα 31: Σημάνσεις από Antivirus



Εικόνα 32: Εισαγμένες Βιβλιοθήκες



Εικόνα 33: Εισαγμένες συναρτήσεις

Type	Size	Section...	Blacklisted (131)	Item (3411)
ascii	5	?:0x02B5	x	.rsrc
ascii	4	.text:0x...	x	d|.H
ascii	4	.text:0x...	x	I+.H
ascii	11	.text:0x...	x	broken pipe
ascii	18	.text:0x...	x	connection aborted
ascii	30	.text:0x...	x	connection already in progress
ascii	18	.text:0x...	x	connection refused
ascii	16	.text:0x...	x	host unreachable
ascii	12	.text:0x...	x	network down
ascii	13	.text:0x...	x	network reset
ascii	19	.text:0x...	x	network unreachable
ascii	10	.text:0x...	x	owner dead
ascii	17	.text:0x...	x	permission denied
ascii	14	.text:0x...	x	protocol error
ascii	22	.text:0x...	x	protocol not supported
ascii	19	.text:0x...	x	wrong protocol type
ascii	8	.text:0x...	x	FlsAlloc
ascii	7	.text:0x...	x	FlsFree
ascii	11	.text:0x...	x	FlsGetValue
ascii	11	.text:0x...	x	FlsSetValue
ascii	15	.rdata:0...	x	CreateSemaphore
ascii	21	.rdata:0...	x	CreateThreadpoolTimer
ascii	18	.rdata:0...	x	SetThreadpoolTimer
ascii	31	.rdata:0...	x	WaitForThreadpoolTimerCallbacks
ascii	20	.rdata:0...	x	CloseThreadpoolTimer
ascii	20	.rdata:0...	x	CreateThreadpoolWait
ascii	17	.rdata:0...	x	SetThreadpoolWait
ascii	19	.rdata:0...	x	CloseThreadpoolWait
ascii	24	.rdata:0...	x	FlushProcessWriteBuffers
ascii	30	.rdata:0...	x	FreeLibraryWhenCallbackReturns
ascii	25	.rdata:0...	x	GetCurrentProcessorNumber
ascii	18	.rdata:0...	x	CreateSymbolicLink

Type	Size	Section...	Blacklisted (131)	Item (3411)
ascii	19	.rdata:0...	x	GetCurrentPackageId
ascii	28	.rdata:0...	x	GetFileInformationByHandleEx
ascii	26	.rdata:0...	x	SetFileInformationByHandle
ascii	20	.rdata:0...	x	CreateThreadpoolWork
ascii	20	.rdata:0...	x	SubmitThreadpoolWork
ascii	19	.rdata:0...	x	CloseThreadpoolWork
ascii	13	.rdata:0...	x	EventRegister
ascii	19	.rdata:0...	x	EventSetInformation
ascii	15	.rdata:0...	x	EventUnregister
ascii	18	.rdata:0...	x	EventWriteTransfer
ascii	11	.rdata:0...	x	LoadLibrary
ascii	11	.rdata:0...	x	FreeLibrary
ascii	11	.rdata:0...	x	KeePass.exe
ascii	17	.rdata:0...	x	\BootstrapDLL.dll
ascii	46	.rdata:0...	x	[.] Done! Check %%APPDATA%%/keepass_export.csv
ascii	99	.rdata:0...	x	C:\Users\Carlos Danger\Documents\Visual Studio 2015\Projects\KeeFarce\dist\Release\x64\KeeFarce.pdb
ascii	18	.rdata:0...	x	WriteProcessMemory
ascii	13	.rdata:0...	x	LoadLibraryEx
ascii	11	.rdata:0...	x	OpenProcess
ascii	24	.rdata:0...	x	CreateToolhelp32Snapshot
ascii	17	.rdata:0...	x	GetExitCodeThread
ascii	13	.rdata:0...	x	Process32Next
ascii	14	.rdata:0...	x	Process32First
ascii	13	.rdata:0...	x	Module32First
ascii	14	.rdata:0...	x	GetProcAddress
ascii	14	.rdata:0...	x	VirtualAllocEx
ascii	15	.rdata:0...	x	GetModuleHandle
ascii	11	.rdata:0...	x	FreeLibrary
ascii	18	.rdata:0...	x	CreateRemoteThread
ascii	12	.rdata:0...	x	Module32Next
ascii	13	.rdata:0...	x	VirtualFreeEx
ascii	19	.rdata:0...	x	GetCurrentDirectory

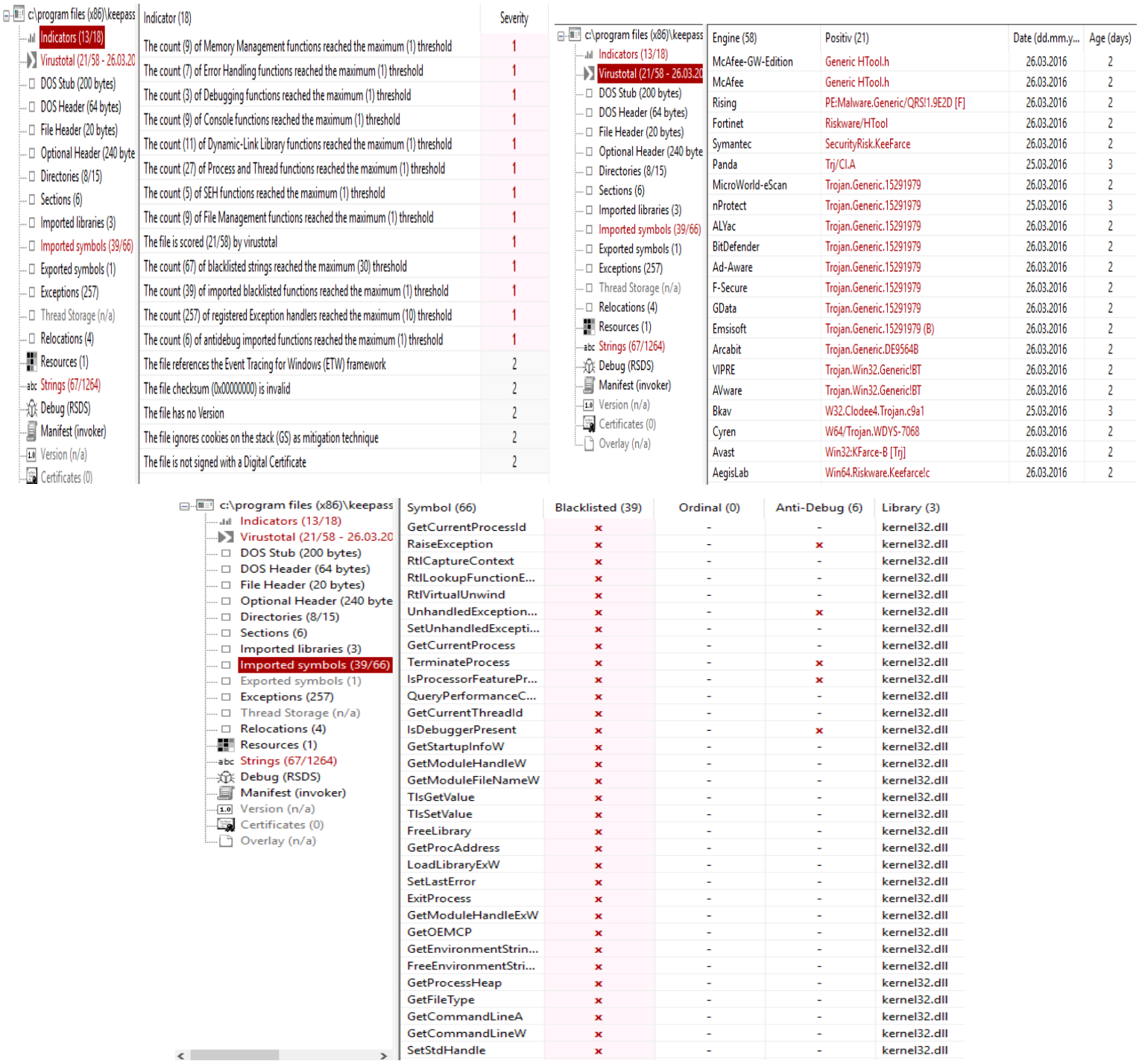
Type	Size	Section...	Blacklisted (131)	Item (3411)
ascii	12	.rdata:0...	x	KERNEL32.dll
ascii	13	.rdata:0...	x	EncodePointer
ascii	13	.rdata:0...	x	DecodePointer
ascii	12	.rdata:0...	x	SetLastError
ascii	11	.rdata:0...	x	CreateEvent
ascii	8	.rdata:0...	x	TlsAlloc
ascii	11	.rdata:0...	x	TlsGetValue
ascii	11	.rdata:0...	x	TlsSetValue
ascii	7	.rdata:0...	x	TlsFree
ascii	13	.rdata:0...	x	CompareString
ascii	8	.rdata:0...	x	SetEvent
ascii	10	.rdata:0...	x	ResetEvent
ascii	17	.rdata:0...	x	RtlCaptureContext
ascii	22	.rdata:0...	x	RtlLookupFunctionEntry
ascii	16	.rdata:0...	x	RtlVirtualUnwind
ascii	24	.rdata:0...	x	UnhandledExceptionFilter
ascii	27	.rdata:0...	x	SetUnhandledExceptionFilter
ascii	17	.rdata:0...	x	GetCurrentProcess
ascii	16	.rdata:0...	x	TerminateProcess
ascii	25	.rdata:0...	x	IsProcessorFeaturePresent
ascii	17	.rdata:0...	x	IsDebuggerPresent
ascii	14	.rdata:0...	x	GetStartupInfo
ascii	23	.rdata:0...	x	QueryPerformanceCounter
ascii	19	.rdata:0...	x	GetCurrentProcessId
ascii	18	.rdata:0...	x	GetCurrentThreadId
ascii	17	.rdata:0...	x	RtlPcToFileHeader
ascii	14	.rdata:0...	x	RaiseException
ascii	11	.rdata:0...	x	RtlUnwindEx
ascii	13	.rdata:0...	x	LoadLibraryEx
ascii	17	.rdata:0...	x	GetModuleFileName
ascii	9	.rdata:0...	x	HeapAlloc
ascii	11	.rdata:0...	x	HeapReAlloc

Type	Size	Section...	Blacklisted (131)	Item (3411)
ascii	17	.rdata:0...	x	GetModuleHandleEx
ascii	12	.rdata:0...	x	GetStdHandle
ascii	9	.rdata:0...	x	WriteFile
ascii	17	.rdata:0...	x	GetModuleFileName
ascii	14	.rdata:0...	x	GetCommandLine
ascii	14	.rdata:0...	x	GetCommandLine
ascii	11	.rdata:0...	x	GetFileType
ascii	16	.rdata:0...	x	FlushFileBuffers
ascii	12	.rdata:0...	x	GetConsoleCP
ascii	14	.rdata:0...	x	GetConsoleMode
ascii	8	.rdata:0...	x	ReadFile
ascii	16	.rdata:0...	x	SetFilePointerEx
ascii	14	.rdata:0...	x	GetProcessHeap
ascii	8	.rdata:0...	x	GetOEMCP
ascii	21	.rdata:0...	x	GetEnvironmentStrings
ascii	22	.rdata:0...	x	FreeEnvironmentStrings
ascii	22	.rdata:0...	x	SetEnvironmentVariable
ascii	9	.rdata:0...	x	FindClose
ascii	15	.rdata:0...	x	FindFirstFileEx
ascii	12	.rdata:0...	x	FindNextFile
ascii	12	.rdata:0...	x	SetStdHandle
ascii	12	.rdata:0...	x	WriteConsole
ascii	11	.rdata:0...	x	ReadConsole
ascii	8	.rdata:0...	x	HeapSize
ascii	10	.rdata:0...	x	CreateFile
ascii	17	.rdata:0...	x	SystemFunction036
ascii	12	.rdata:0...	x	ADVAPI32.dll
unicode	11	.rdata:0...	x	mscoree.dll
unicode	7	.rdata:0...	x	chinese
unicode	7	.rdata:0...	x	CONOUT$
unicode	12	.rdata:0...	x	kernel32.dll
unicode	17	.rdata:0...	x	g\KeeFarceDLL.dll

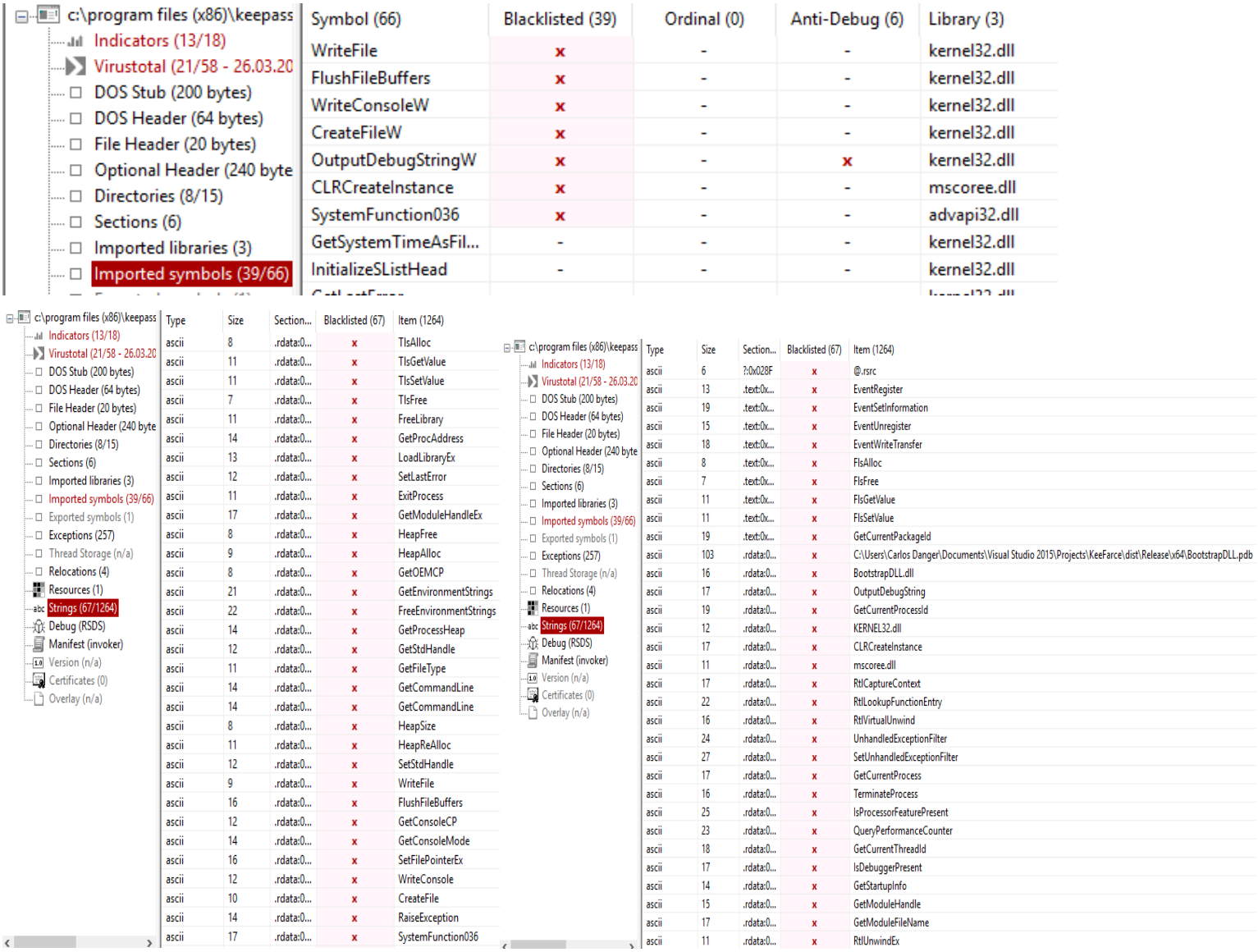
Εικόνα 34: Ύποπτα Αλφαριθμητικά

Bootstrap.dll

Όπως και παραπάνω, η αυτοματοποιημένη ανάλυση του αρχείου θα παρουσιαστεί από μία σειρά Εικόνων των αποτελεσμάτων του εργαλείου *PeStudio.*



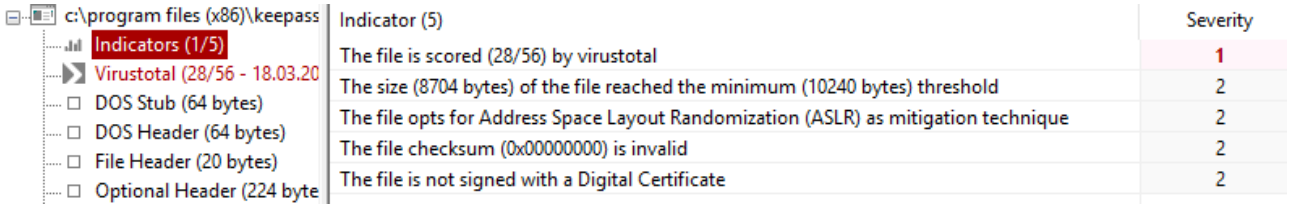
Εικόνα 35: Στοιχεία ύποπτης παρουσίας (1)



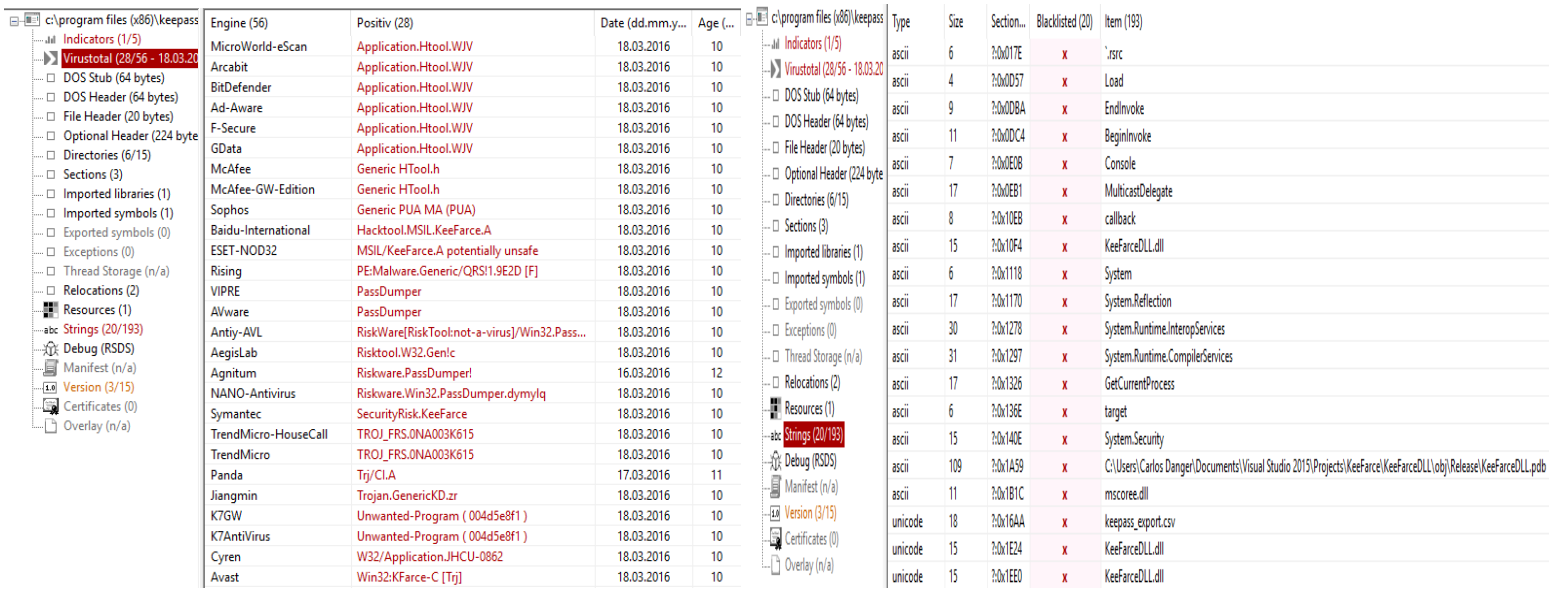
Εικόνα 36: Στοιχεία ύποπτης παρουσίας (2)

KeeFarce.dll

Ομοίως με προηγουμένως, η παρακάτω σειρά εικόνων κάνει εμφανή τα σημεία από τα οποία φαίνεται ότι το υπό εξέταση αρχείο είναι ύποπτο για κακόβουλη δραστηριότητα.



Εικόνα 37: Ενδείκτες κακόβουλης υπόστασης του KeeFarce.dll



Εικόνα 38: Υποδείξεις κακοβουλίας της δυναμικής βιβλιοθήκης KeeFarce.dll

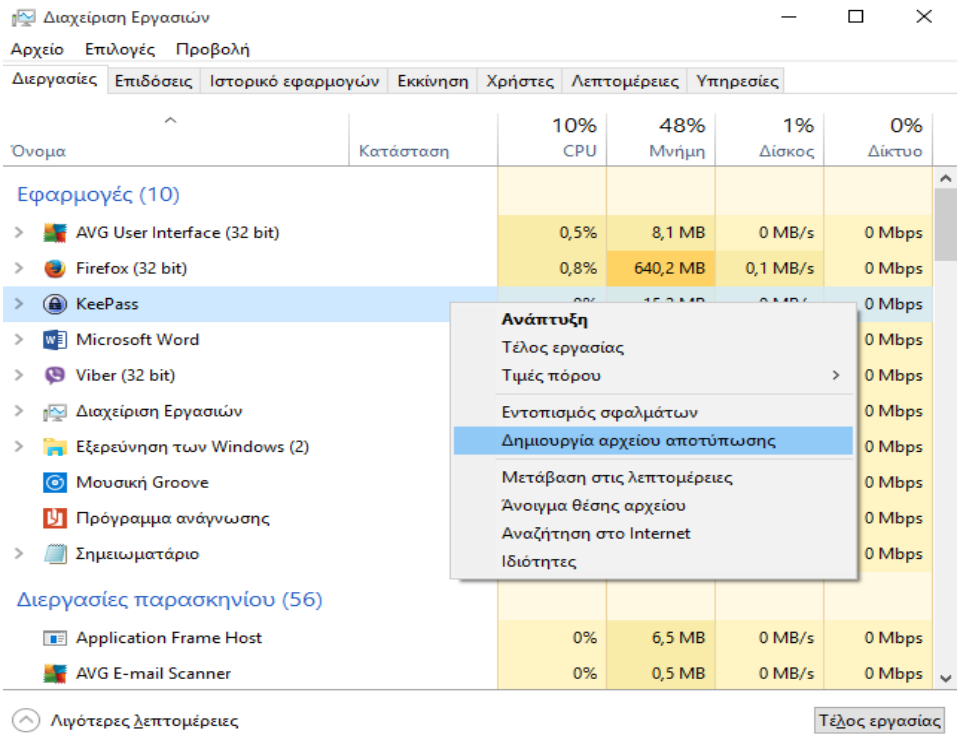
Είναι εμφανές ότι τα παραπάνω αποτελέσματα της αυτοματοποιημένης ανάλυσης είναι ενδείξεις κακόβουλης δραστηριότητας και δεν μπορούν να παρθούν ως απόλυτα δεδομένα. Η ανάλυση τέτοιου είδους απαιτεί την διενέργεια περαιτέρω έρευνας στα στοιχεία που βρέθηκαν για να επαληθευτεί αν ανήκουν σε κακόβουλο λογισμικό, καθώς πολλά από αυτά μπορεί να οδηγήσουν στην κατάδειξη λανθασμένων στόχων (*False Alarm*). Ωστόσο, σαν πρώτο βήμα του σταδίου ανάλυσης, μας δίνει πολλές πληροφορίες και κατευθύνσεις για να ψάξουμε βαθύτερα. Στο επόμενο στάδιο της ανάλυσης θα κινηθούμε εξετάζοντας παρόμοια στοιχεία με αυτά που βρέθηκαν με την αυτοματοποιημένη διαδικασία. Ωστόσο, θα προσπαθήσουμε να διαπιστώσουμε με σιγουριά ποια από αυτά ή ποιες άλλες ενδείξεις υπάρχουν, η χρήση των οποίων δεν θα μας οδηγήσει σε λανθασμένη ανίχνευση κακόβουλου λογισμικού.

5.2.3.2 Ανάλυση Αλφαριθμητικών

Ο συγκεκριμένος τύπος στοχεύει στην εύρεση αλφαριθμητικών σε αρχεία που θεωρούμε ότι αποτελούν τμήματα κακόβουλης δραστηριότητας. Στην περίπτωση μας η αλφαριθμητική ανάλυση αποτελεί τον προχωρημένο τύπο ανάλυσης που επιλέξαμε για να μελετήσουμε το κακόβουλο λογισμικό *KeeFarce*. Ο λόγος αφορά στον τελικό σκοπό της μελέτης που έχουμε ορίσει. Σκοπός μας ήταν να φτιάξουμε κανόνες στο εργαλείο *Yara*, οι οποίοι θα μας βοηθήσουν να ανιχνεύουμε αυτοματοποιημένα και μοναδικά ένα κακόβουλο αρχείο που σχετίζεται με το *KeeFarce*.

Για να πετύχουμε αυτόν τον σκοπό, έπρεπε να βρούμε το σύνολο των αλφαριθμητικών που εμφανίζονται μοναδικά στην επίδραση στην μνήμη του *KeePass* κατόπιν διενέργειας του *KeeFarce.* Υπάρχουν πολλοί τρόποι να γίνει αυτή η ανάλυση. Εμείς επιλέξαμε τον πιο άμεσο, ο οποίος δίνει πάντα γρήγορα και έγκυρα αποτελέσματα σε οποιαδήποτε περίπτωση και βαθμό Διείσδυσης του κακόβουλου λογισμικού, χωρίς να χρειάζεται κανένα επιπλέον εργαλείο. Ή μέθοδος μας αποτελείται από τα εξής βήματα:

- Αποτύπωση της μνήμης του *KeePass* μέσω της Διαχείρισης Εργασιών των *Windows* (Εικόνα 39).
- Χρησιμοποίηση του εργαλείου *strings* του πακέτου εργαλείων *Sysinternals* για την εξαγωγή όλων των αλφαριθμητικών που περιέχονται στο παραγόμενο αρχείο αποτύπωσης από το προηγούμενο βήμα.
- Προσεκτική παρατήρηση για εύρεση στοιχείων κακόβουλης δραστηριότητας, έχοντας υπόψη τα ευρήματα από τα προηγούμενα στάδια.



Εικόνα 39: Διαχειριστής Εργασιών

Το αποτέλεσμα της έρευνα μας στο αρχείο που προέκυψε είναι πολύτιμες πληροφορίες οι οποίες υποδηλώνουν ξεκάθαρη κακόβουλη παρουσία. Εκτός από τα αλφαριθμητικά *KeeFarceDLL* και *Bootstrap* τα οποία μας παραπέμπουν στο *KeeFarce,* έχουμε πληθώρα περιπτώσεων στις οποίες ανιχνεύτηκαν ύποπτα στοιχεία. Για παράδειγμα, εντοπίσθηκε το αλφαριθμητικό *WriteCsvString,* το οποίο παραπέμπει στην εγγραφή αλφαριθμητικών σε ένα αρχείο *.csv*. Εμείς, όμως γνωρίζουμε ότι η εφαρμογή *KeePass* δεν κάνει καμία τέτοια ενέργεια εγγραφής, ούτε παράγει κάποιο *.csv* αρχείο. Συνεπώς, η ύπαρξη στην μνήμη του

τέτοιων συναρτήσεων είναι ισχυρή ένδειξη της τεχνικής *dll Injection.*

Η προσεκτική μελέτη του αρχείου αλφαριθμητικών της μνήμης του *KeePass* καρποφόρησε με την ανάδειξη των παρακάτω μοναδικών προσδιοριστικών στοιχείων του κακόβουλου λογισμικού *KeeFarce:*

- BootstrapDLL.pdb. Η επέκταση του αρχείου αυτού ονομάζεται *Program Database* και αφορά σε στοιχεία αποσφαλμάτωσης και στοιχεία της κατάστασης του *Project.* Αρχεία με τέτοια επέκταση παράγονται κατά την φάση μεταγλώττισης ενός προγράμματος στις γλώσσες *C/C++/Visual Basic/C#/Jscript.NET* με την επιλογή της αποσφαλμάτωσης. Συνεπώς από το συγκεκριμένο αλφαριθμητικό συνάγουμε το συμπέρασμα της ύπαρξης σχέσης μεταξύ της διεργασίας *KeePass*, της οποίας την μνήμη εξετάζουμε και μιας δυναμικής βιβλιοθήκης με όνομα *Bootstrap.dll.* [31]
- KeeFarceDLL.pdb. Ομοίως με το παραπάνω αλφαριθμητικό και στην περίπτωση αυτή έχουμε ένα αρχείο με δεδομένα αποσφαλμάτωσης για μία δυναμική βιβλιοθήκη με όνομα *KeeFarce.dll,* το οποίο ανιχνεύτηκε στην μνήμη της διεργασίας *KeePass,* υποδηλώνοντας πιθανότητα *Dll Injection.*
- mscordacwks.pdb. Το συγκεκριμένο αλφαριθμητικό στην μνήμη του *KeePass* αφορά στην κοινή βιβλιοθήκη *mscordacwks.dll,* η οποία όμως στην περίπτωση αυτή έχει χρησιμοποιηθεί σε μεταγλώττιση και αποσφαλμάτωση, γεγονός που φανερώνει την χρήση της σε πρόγραμμα αποσφαλμάτωσης. Η εμφάνιση της στην μνήμη της διεργασίας θύμα φανερώνει την προσπάθεια κάποια οντότητας να χρησιμοποίησει δυνατότητες αποσφαλμάτωσης μέσω του *CLR.* [32]
- keepass_export.csv. Το συγκεκριμένο αλφαριθμητικό αποτελεί μονοσήμαντος προσδιορισμό ενός κακόβουλου λογισμικού, καθώς υποδεικνύει την δημιουργία ενός αρχείου *.csv* με άμεση συνάφεια με το *KeePass.* Η χρήση του αλφαριθμητικού για έρευνα στον δίσκο του εν λόγω αρχείου μπορεί να οδηγήσει το αναλυτή στην εύρεση του αποτελέσματος της δράσης του *KeeFarce.*
- 5c552ab6-fc09-4cb3-8e36-22fa03c798b7. Ξεχωριστό αλφαριθμητικό, το οποίο αποτελεί και *id* ενός *CLR Instance.* Το αλφαριθμητικό αυτό προδίδει την χρήση CLR για σκοπούς αποσφαλμάτωσης.
- 17589ea6-fcc9-44bb-92ad-d5b3eea6af03. Το αλφαριθμητικό αποτελεί αναγνωριστικό του αρχείου της δυναμικής βιβλιοθήκης *KeeFarceDLL.DLL,* η οποία φαίνεται να έχει φορτωθεί στην μνήμη της διεργασίας *KeePass.* Αυτό το αλφαριθμητικό υποδηλώνει και μονοσήμαντα την δράση του κακόβουλου λογισμικού *KeeFarce* και αποτελεί ένα μοναδικό χαρακτηριστικό αποτύπωμα του.

Ο συνδυασμός των παραπάνω αλφαριθμητικών και η κατάλληλη χρήση τους μπορεί να μας οδηγήσει στον σωστό και ασφαλή εντοπισμό του κακόβουλου λογισμικού KeeFarce μέσω της συγγραφής κανόνων *Yara.*

5.2.4 Συγγραφή κανόνων *Yara* για τον εντοπισμό του *KeeFarce*

Κατόπιν προσεκτικής εκτέλεσης της διαδικασίας της ανάλυσης των δειγμάτων που είχαμε σχετικά με το κακόβουλο λογισμικό *KeeFarce* εξήχθησαν αποτελέσματα ικανά για να περιγράψουν πολύ καλά την παρουσία και την δράση του. Μερικά από αυτά τα στοιχεία, ιδιαίτερα τα «κακόβουλα» αλφαριθμητικά, θα τροφοδοτήσουν το τελευταία στάδιο της μελέτης κακόβουλου λογισμικού που δεν είναι άλλο από την εξαγωγή τεχνοτροπιών ανίχνευσης του κακόβουλου λογισμικού. Η μελέτη κακόβουλών είναι μία αέναη κυκλική διαδικασία, το τέλος του τελευταίου σταδίου της οποία ανατροφοδοτεί την αρχή του πρώτου σταδίου της. Συνεπώς, στο τελευταίο στάδιο θα πρέπει να εξαχθούν στοιχεία που θα βοηθήσουν στον εντοπισμό μίας κακόβουλης οντότητας προς εξέταση στον υπολογιστή, ώστε να ξεκινήσει η ανάλυση της.

Τελευταίο στάδιο στην ανάλυση μας αποτελεί η εξαγωγή κανόνων *Yara* για τον εντοπισμό κακόβουλων αρχείων. Στην περίπτωση μας εξάχθηκαν κανόνες με τους εξής τρόπους:

- Χειροκίνητη παράγωγή κανόνων, με προσωπική αναζήτηση των κατάλληλων αλφαριθμητικών (προηγούμενο στάδιο) και των τρόπων συνδυασμού τους για την εξασφάλιση της εγκυρότητας των κανόνων και της αποφυγής λανθασμένων συναγερμών.
- Αυτοματοποιημένη διαδικασία με την χρήση του εργαλείου *yarGen*
- Συνδυασμός των παραπάνω δύο μεθόδων.

Όσον αφορά στην πρώτη περίπτωση, η δημιουργία του κανόνα είχε ως γνώμονα την χρήση της δυνατότητας του *Yara* για εξέταση τρεχόντων διεργασιών. Η επιλογή μας αυτή έγινε με σκοπό να αποδοθεί μία δυναμικότητα στο αποτέλεσμα, καθώς η εξέταση των αλφαριθμητικών μίας τρέχουσας διεργασίας συνδέεται άμεσα με την εικόνα που παρουσιάζει εκείνη την στιγμή ο χώρος μνήμης που έχει δεσμεύσει στην *RAM.* Η παρατήρηση αυτή οδηγεί σε αναγνώριση απειλών στον υπολογιστή προσομοιώνοντας την λειτουργία ενός απλού αλλά πολύ ισχυρού αντιικού προγράμματος. Συνεπώς, στην δημιουργία του κανόνα παραλήφθηκαν κάποια *modules,* η λειτουργία των οποίων είναι χρήσιμη σε άλλου είδους

χρήση. Η χειροκίνητη παραγωγή κανόνα έδωσε σαν αποτέλεσμα το κανόνα που φαίνεται στην Εικόνα 40.

```
rule KeeFarce_strings
{
    meta:
        description = "A password stealing malware from process address space"
        author = "Vasileios Markopoulos"
        date = "2015/2/21"
    strings:
        $str1 = "BootstrapDLL.pdb" fullword ascii
        $str2 = "KeeFarceDLL.pdb" fullword ascii
        $str3 = "mscordacwks.pdb"
        $str4 = "keepass_export.csv" nocase fullword ascii
        $str5 = {35 63 35 35 32 61 62 36 2d 66 63 30 39 2d 34 63 62 33 2d 38 65 33 36 2d 32 32 66 61 30 33 63 37 39 38 62 37} //Clr instance id value
        $str6 = {31 37 35 38 39 65 61 36 2d 66 63 63 39 2d 34 34 62 62 2d 39 32 61 64 2d 64 35 62 33 65 65 61 36 61 66 30 33} // KeeFarceDLL file id

    condition:
        all of them

}
```

Εικόνα 40: Κανόνας Yara για εντοπισμό του KeeFarce

Στο παραπάνω κανόνα διακρίνονται τα τρία τμήματα που τον αποτελούν:

- Το τμήμα των μεταδεδομένων, στα οποία παρέχονται πληροφορίες για τον κανόνα.
- Το τμήμα των αλφαριθμητικών, στο οποίο αποτυπώνονται τα «κακόβουλα» αναγνωριστικά στοιχεία. Δύο από τα έξι αλφαριθμητικά έχουν δηλωθεί με το δεκαεξαδικό αναλογό τους, διότι διαπιστώθηκε ότι μόνο με αυτόν τον τρόπο μπορούσε να εξασφαλιστεί σε κάθε περίπτωση η ορθή και έγκυρη λειτουργία του κανόνα.
- Το τμήμα της συνθήκης, στο οποίο υποδηλώνεται ο τρόπος συνδυασμού των δηλωμένων αλφαριθμητικών.

Ο κανόνας δεν περιέχει πολύπλοκη αρχιτεκτονική δηλώσεων, αλλά έχει ως φιλοσοφία την διαπιστευμένη ανίχνευση της δράσης του λογισμικού *KeeFarce,* **χωρίς να χρειάζεται καν να το εντοπίσουμε.** Αυτή είναι μία εξαιρετική καινοτομία που πηγάζει από την προσεκτική επιλογή των αλφαριθμητικών που χρησιμοποιούνται και δίνουν στον κανόνα λειτουργία αντιικού προγράμματος. Σε επόμενη ενότητα, μάλιστα, διαπιστεύεται η πλήρης αποφυγή έγερσης λανθασμένων συναγερμών που προσφέρει.

Το αποτέλεσμα του εργαλείου *yarGen* δίνει ένα εντελώς διαφορετικό σετ από κανόνες με διαφορετική φιλοσοφία. Το εργαλείο αυτό αδυνατεί να παράγει κανόνες που στοχεύουν σε τρέχουσες διαδικασίες. Αντίθετα, είναι εξαιρετικό για μία πρώτη δημιουργία κανόνων σχετικά με ύποπτα εκτελέσιμα αρχεία και δυναμικές βιβλιοθήκες. Η παραγωγή αυτού του είδους των κανόνων είναι χρήσιμη σε δεύτερο χρόνο, αφού δηλαδή έχει ανιχνευτεί η φόρτωση της δυναμικής βιβλιοθήκης από το *KeeFarce* και θέλουμε να ψάξουμε στον δίσκο για την ύπαρξη

αρχείων με σχετικό κακόβουλο περιεχόμενο. Το σετ των κανόνων που εξάγεται από το εργαλείο *yarGen* χαρακτηρίζεται από πλουραλισμό, καθώς εξάγει όλα τα αλφαριθμητικά που δεν περιέχονται στην Βάση Δεδομένων. Από τα είκοσι αλφαριθμητικά που θα συνθέσουν τον κανόνα, μερικά έχουν μικρό σκορ και πιθανότητα δεν αποτελούν ένδειξη κακόβουλης παρουσίας. Για το λόγο αυτό, το εργαλείο προσφέρει την δυνατότητα για εξαγωγή κανόνων μόνο με αλφαριθμητικά που έχουν αριθμό σκορ υψηλότερο από ένα κατώφλι που ορίζεται εξωτερικά από τον αναλυτή. Επίσης, πολλά από αυτά προσδιορίζουν μόνο το αρχείο που εξετάζεται και δεν μπορούν να χρησιμοποιηθούν για την ανίχνευση μίας οικογενείας παρόμοιων κακόβουλων αρχείων. Από τους κανόνες που παράγονται, θα πρέπει ο αναλυτής να επιλέξει ποιο αλφαριθμητικά τελικά μπορούν χρησιμοποιηθούν σύμφωνα με την εμπειρία του. Παρακάτω θα παρατεθούν οι πέντε κανόνες που εξήχθησαν όταν το εργαλείο *yarGen* κλήθηκε να εξετάσει το φάκελο του *KeePass.exe,* στον οποίο έχει εισαχθεί το *KeeFarce* και από όπου εκτελείται.

```
rule KeeFarce {
    meta:
        description = "Auto-generated rule - file KeeFarce.exe"
        author = "YarGen Rule Generator"
        reference = "not set"
        date = "2016-04-07"
        hash = "57771f6312b091fb5450112864e56413ae2a2a2874289e8245eb3a0d286577e9"
    strings:
        $s0 = "C:\\Users\\Carlos Danger\\Documents\\Visual Studio 2015\\Projects\\KeeFarce\\dist\\Release\\x64\\KeeFarce.pdb" fullword ascii /* score: '39.00' */
        $s1 = "[.] Done! Check %%APPDATA%%/keepass_export.csv" fullword ascii /* score: '28.00' */
        $s2 = "Process found, but OpenProcess() failed: " fullword ascii /* score: '22.00' */
        $s3 = "CallExport: Could not get module Snapshot for remote process." fullword ascii /* score: '22.00' */
        $s4 = "\\BootstrapDLL.dll" fullword ascii /* score: '22.00' */
        $s5 = "g\\KeeFarceDLL.dll" fullword wide /* score: '21.00' */
        $s6 = "CallExport: Could not get handle to process." fullword ascii /* score: '20.00' */
        $s7 = "[.] Injecting BootstrapDLL into %d" fullword ascii /* score: '17.01' */
        $s8 = "Specified Process not found" fullword ascii /* score: '17.00' */
        $s9 = "CallExport: Cound not find module in remote process." fullword ascii /* score: '16.00' */
        $s10 = "CallExport: Could not create thread in remote process." fullword ascii /* score: '16.00' */
        $s11 = "CallExport: Could not get thread exit code." fullword ascii /* score: '15.00' */
        $s12 = "Copyright (c) by P.J. Plauger, licensed by Dinkumware, Ltd. ALL RIGHTS RESERVED." fullword ascii /* score: '13.00' */
        $s13 = "CallExport: DOS PE header is invalid." fullword ascii /* score: '11.00' */
        $s14 = "CallExport: NT PE header is invalid." fullword ascii /* score: '11.00' */
        $s15 = "operator \"\" " fullword ascii /* score: '9.00' */
        $s16 = "Microsoft.CRTProvider" fullword ascii /* score: '8.00' */
        $s17 = "VirtualAllocEx Failed:" fullword ascii /* score: '8.00' */
        $s18 = "VirtualAllocEx Failed" fullword ascii /* score: '8.00' */
        $s19 = "CallExport failed" fullword ascii /* score: '7.00' */
        $s20 = "CallExport: Symbol names missing entirely." fullword ascii /* score: '7.00' */
    condition:
        uint16(0) == 0x5a4d and filesize < 715KB and all of them
}
```

Εικόνα 41: Κανόνας για το αρχείο KeeFarce.exe από το εργαλείο YarGen

```
rule Microsoft_Diagnostics_Runtime {
    meta:
        description = "Auto-generated rule - file Microsoft.Diagnostics.Runtime.dll"
        author = "YarGen Rule Generator"
        reference = "not set"
        date = "2016-04-07"
        hash = "8ce1e8ec68f8322f9f0f8c9ca09d6927a89ddc512ef2d52b11b0c857a80f7c9c"
    strings:
        $s0 = "TargetZMicrosoft.Diagnostics.Runtime.Interop.WINDBG_EXTENSION_APIS32.#lpWriteProcessMemoryRoutine" fullword ascii /* score: '39.00' */
        $s1 = "TargetYMicrosoft.Diagnostics.Runtime.Interop.WINDBG_EXTENSION_APIS64.#lpReadProcessMemoryRoutine" fullword ascii /* score: '39.00' */
        $s2 = "TargetZMicrosoft.Diagnostics.Runtime.Interop.WINDBG_EXTENSION_APIS64.#lpWriteProcessMemoryRoutine" fullword ascii /* score: '39.00' */
        $s3 = "TargetYMicrosoft.Diagnostics.Runtime.Interop.WINDBG_EXTENSION_APIS32.#lpReadProcessMemoryRoutine" fullword ascii /* score: '39.00' */
        $s4 = "System.Collections.Generic.IEnumerable<Microsoft.Diagnostics.Runtime.Utilities.DumpModule>.GetEnumerator" fullword ascii /* score: '37.00' */
        $s5 = "System.Collections.Generic.IEnumerable<Microsoft.Diagnostics.Runtime.Utilities.DumpThread>.GetEnumerator" fullword ascii /* score: '37.00' */
        $s6 = "TargetMMicrosoft.Diagnostics.Runtime.Interop.DEBUG_READ_USER_MINIDUMP_STREAM.#BufferG" fullword ascii /* score: '37.00' */
        $s7 = "System.Collections.Generic.IEnumerator<Microsoft.Diagnostics.Runtime.Utilities.DumpModule>.get_Current" fullword ascii /* score: '36.00' */
        $s8 = "System.Collections.Generic.IEnumerator<Microsoft.Diagnostics.Runtime.Utilities.DumpThread>.get_Current" fullword ascii /* score: '36.00' */
        $s9 = "c:\\b\\4741\\2714\\src\\intermediate\\ClrMemDiag.csproj_43b70d00\\Release\\Microsoft.Diagnostics.Runtime.pdb" fullword ascii /* score: '35.00' */
        $s10 = "XSystem.Char, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089 " fullword ascii /* score: '34.42' */
        $s11 = "ZSystem.UInt64, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" fullword ascii /* score: '34.00' */
        $s12 = "ZSystem.UInt32, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" fullword ascii /* score: '34.00' */
        $s13 = "ZSystem.Double, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" fullword ascii /* score: '34.00' */
        $s14 = "ZSystem.UInt16, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" fullword ascii /* score: '34.00' */
        $s15 = "XSystem.Byte, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" fullword ascii /* score: '34.00' */
        $s16 = "YSystem.SByte, mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" fullword ascii /* score: '34.00' */
        $s17 = "GetRunningProcessSystemIdByExecutableNameWide" fullword ascii /* score: '33.00' */
        $s18 = "GetRunningProcessSystemIdByExecutableName" fullword ascii /* score: '33.00' */
        $s19 = "GetNumberPossibleExecutingProcessorTypes" fullword ascii /* score: '33.00' */
        $s20 = "mscordacwks.dll" fullword wide /* reversed goodware string 'lld.skwcadrocsm' */ /* score: '33.00' */
    condition:
        uint16(0) == 0x5a4d and filesize < 1578KB and all of them
}
```

Εικόνα 43: Κανόνας για το αρχείο Microsoft.Diagnostics.Runtime.dll από το εργαλείο YarGen

```
rule KeeFarceDLL {
    meta:
        description = "Auto-generated rule - file KeeFarceDLL.dll"
        author = "YarGen Rule Generator"
        reference = "not set"
        date = "2016-04-07"
        hash = "5ea9a04284157081bd5999e8be96dda8fac594ba72955adacb6fa48bdf866434"
    strings:
        $s0 = "C:\\Users\\Carlos Danger\\Documents\\Visual Studio 2015\\Projects\\KeeFarce\\KeeFarceDLL\\obj\\Release\\KeeFarceDLL.pdb" fullword ascii /* score: '45.00' */
        $s1 = "KeeFarceDLL.dll" fullword wide /* score: '23.00' */
        $s2 = "KeePass.DataExchange.PwExportInfo" fullword wide /* score: '20.00' */
        $s3 = "KeePass.DataExchange.Formats.KeePassCsv1x" fullword wide /* score: '18.00' */
        $s4 = "Microsoft.Diagnostics.Runtime.Desktop.V45Runtime" fullword wide /* score: '15.00' */
        $s5 = "AttachToProcess" fullword ascii /* score: '15.00' */
        $s6 = "Microsoft.Diagnostics.Runtime.DacLibrary" fullword wide /* score: '14.00' */
        $s7 = "DataTarget" fullword ascii /* score: '14.00' */
        $s8 = "keepass_export.csv" fullword wide /* score: '14.00' */
        $s9 = "targetObj" fullword ascii /* score: '14.00' */
        $s10 = "get_ClrVersions" fullword ascii /* score: '12.01' */
        $s11 = ".NETFramework,Version=v4.0" fullword ascii /* score: '12.00' */
        $s12 = "[KeeFarceDLL] Dynamic Method init" fullword wide /* score: '12.00' */
        $s13 = "Microsoft.Diagnostics.Runtime" fullword ascii /* score: '12.00' */
        $s14 = "getRootGroupPtr" fullword ascii /* score: '12.00' */
        $s15 = ".NET Framework 4" fullword ascii /* score: '11.00' */
        $s16 = "[KeeFarceDLL] init done" fullword wide /* score: '10.00' */
        $s17 = "KeePass.UI.DocumentManagerEx" fullword wide /* score: '9.00' */
        $s18 = "KeeFarceDLL" fullword wide /* score: '9.00' */
        $s19 = "getActiveDbPtr" fullword ascii /* score: '9.00' */
        $s20 = "mscordacwks" fullword wide /* score: '8.00' */
    condition:
        uint16(0) == 0x5a4d and filesize < 25KB and all of them
}
```

Εικόνα 42: Κανόνας για το αρχείο KeeFarceDLL.dll από το εργαλείο YarGen

```
rule BootstrapDLL {
    meta:
        description = "Auto-generated rule - file BootstrapDLL.dll"
        author = "YarGen Rule Generator"
        reference = "not set"
        date = "2016-04-07"
        hash = "0334c4cf4438eab3982faf71072ae5cb50ca6a9ee49bf0a5bc4c7eeb84e363f1"
    strings:
        $s0 = "C:\\Users\\Carlos Danger\\Documents\\Visual Studio 2015\\Projects\\KeeFarce\\dist\\Release\\x64\\BootstrapDLL.pdb" fullword ascii /* score: '45.00'
        $s1 = "BootstrapDLL.dll" fullword ascii /* score: '24.00' */
        $s2 = "[Bootstrap] Attempting exec in default app domain" fullword wide /* score: '24.00' */
        $s3 = "KeeFarceDLL.KeeFarce" fullword wide /* score: '13.00' */
        $s4 = "[Bootstrap] Started the runtime!" fullword wide /* score: '10.01' */
        $s5 = "[Bootstrap] Runtime is loadable!" fullword wide /* score: '10.00' */
        $s6 = "[Bootstrap] Got CLR runtime" fullword wide /* score: '9.00' */
        $s7 = "operator \"\" " fullword ascii /* score: '9.00' */
        $s8 = "Microsoft.CRTProvider" fullword ascii /* score: '8.00' */
        $s9 = "[Bootstrap] Created CLR instance" fullword wide /* score: '7.00' */
        $s10 = ".CRT$XPXA" fullword ascii /* score: '7.00' */
        $s11 = ".rtc$IZZ" fullword ascii /* score: '6.00' */
        $s12 = ".rtc$TAA" fullword ascii /* score: '6.00' */
        $s13 = ".rtc$TZZ" fullword ascii /* score: '6.00' */
        $s14 = "[Bootstrap] Got interface." fullword wide /* score: '6.00' */
        $s15 = ".rtc$IAA" fullword ascii /* score: '6.00' */
        $s16 = ".CRT$XIC" fullword ascii /* score: '6.00' */
        $s17 = ".CRT$XPA" fullword ascii /* score: '6.00' */
        $s18 = ".CRT$XPZ" fullword ascii /* score: '6.00' */
        $s19 = ".CRT$XPX" fullword ascii /* score: '6.00' */
        $s20 = ".CRT$XTZ" fullword ascii /* score: '6.00' */
    condition:
        uint16(0) == 0x5a4d and filesize < 244KB and all of them
}
```

Εικόνα 44: Κανόνας για το αρχείο BootstrapDLL.dll από το εργαλείο YarGen

Οι κανόνες που παράχθηκαν μας δίνουν μία αρκετά καλή διαισθητική δυνατότητα τόσο για σημαντικά αλφαριθμητικά, πολλά από τα οποία μπορούνε να χρησιμοποιηθούν στην παραγωγή των τελικών κανόνων ανίχνευσης του *KeeFarce,* όσο και για κάποιες λειτουργικές λεπτομέρειες των αρχείων που εξετάζονται. Παρόλα αυτά και οι τέσσερεις παραπάνω κανόνες θα περάσουν από δύο φίλτρα. Αρχικά, θα χρησιμοποιηθεί η δυνατότητα που προσφέρει το *yarGen* για δημιουργία κανόνων που αποτελούνται από αλφαριθμητικά που έχουν σκορ παραπάνω από ένα κατώφλι. Στην συνέχεια, από το αποτέλεσμα θα αφαιρέσουμε διαισθητικά αυτά τα αλφαριθμητικά που είτε δεν προσδιορίζουν το κακόβουλο λογισμικό *KeeFarce,* είτε το «φωτογραφίζουν» με τέτοια λεπτομέρεια που σε περίπτωση που κάποιος αλλάξει ελαφρώς την ονοματολογία κάποιων στοιχείων του λογισμικού, αυτό θα είναι ικανό να περάσει τους ελέγχους του *Yara.*

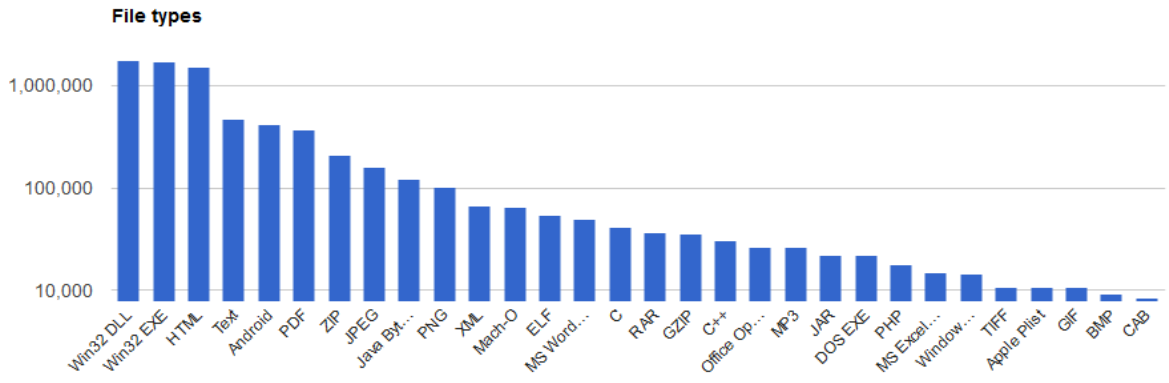
5.2.5 Έλεγχος εγκυρότητας κανόνων

Οι κανόνες του *Yara* σχεδιάζονται με σκοπό την ταυτοποίηση των αλφαριθμητικών που έχουν στο σώμα τους με κάποιο εξεταζόμενο αρχείο για την σήμανση μίας ειδοποίησης. Κατά την φάση του σχεδιασμού και της υλοποίησης επιλέγονται τα κατάλληλα αλφαριθμητικά

τα οποία εντοπίζουν κάποιο συγκεκριμένο κακόβουλο λογισμικό. Στην προσπάθεια αυτή, το μεγαλύτερο τροχοπέδη αποτελούν οι λανθασμένοι συναγερμοί. Λανθασμένος συναγερμός εγείρεται μόλις το *Yara* εντοπίσει ένα αρχείο και το σημάνει ως κακόβουλο χωρίς να είναι. Πηγή του προβλήματος αποτελεί η επιλογή αλφαριθμητικών τα οποία δεν συναντιόνται μόνο στο επίμαχο κακόβουλο λογισμικό αλλά και σε άλλα λογισμικά, είτε κακόβουλα οδηγώντας σε παραπλάνηση, είτε σε καλόβουλα κατευθύνοντας τον αναλυτή στη έρευνα αρχείων χάνοντας πολύτιμο χρόνο. Για την αποφυγή τέτοιων περιπτώσεων, διεξήγαμε μία εκτεταμένη και ενδελεχής έρευνα τόσο σε καλόβουλα αρχεία όσο και σε κακόβουλα δείγματα για να διαπιστεύσουμε την πλήρη ορθότητα των κανόνων μας.

Όσον αφορά στα καλόβουλα αρχεία επιλέξαμε ένα σκληρό δίσκο μεγέθους **465 *GB*** και ο οποίος είναι γεμάτος με αρχεία συνηθισμένα συνολικού μεγέθους **445 *GB***, παντός τύπου και ελεγμένα από το αντιικό πρόγραμμα *AVG Antivitus Free 2016.* Στα αρχεία αυτά έγινε προσεκτική επιλογή και ξεκαθάρισμα για να επιτευχθούν οι δύο παρακάτω απαιτήσεις:

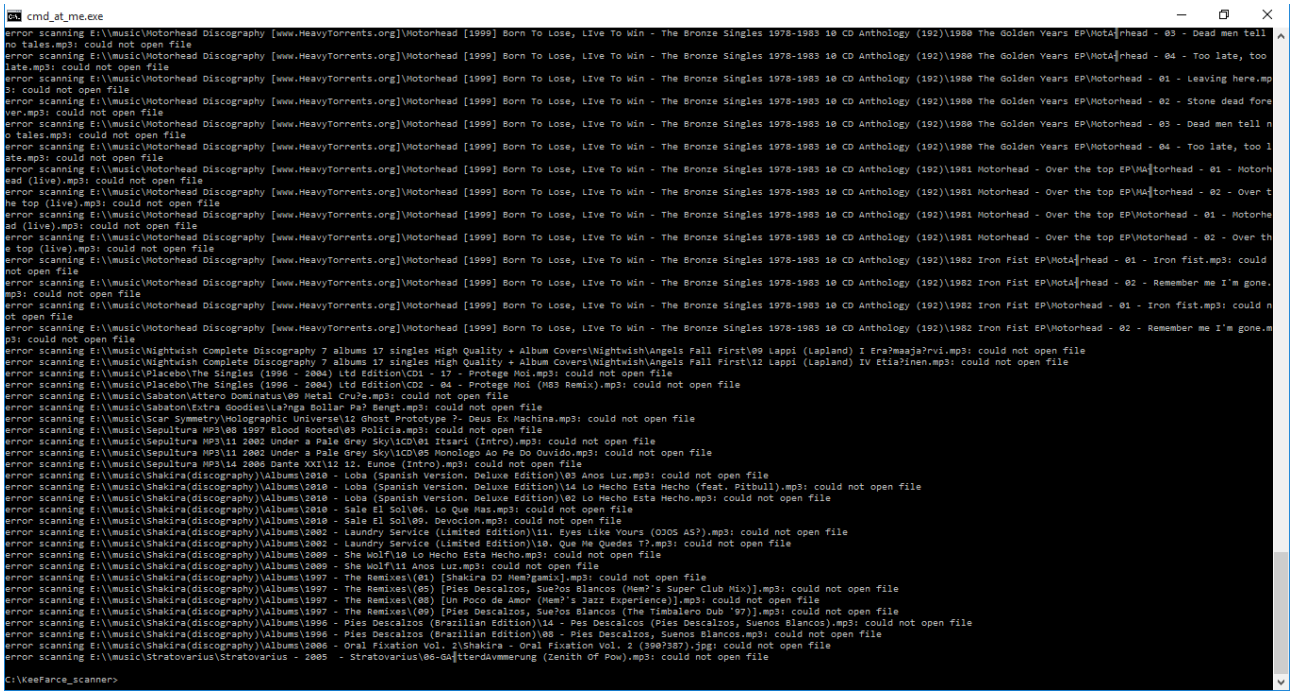
- Τα αρχεία θα έπρεπε να ήταν όσο το δυνατόν περισσότερο διαπιστευμένα ως προς την προέλευση και τον εκδότη καθώς και για το περιεχόμενο. Προς αυτή την κατεύθυνση έγιναν προσπάθειες για να συμπεριληφθούν αρχεία που αποκτήθηκαν από έγκυρες πηγές.
- Η ποικιλομορφία των αρχείων, γεγονός εξαιρετικής σημασίας για την εξαγωγή ασφαλών και έγκυρων αποτελεσμάτων. Πραγματοποιήθηκε προσεκτική επιλογή για να συμπεριληφθούν αρχεία πολλών τύπων όπως παρουσιάζονται στο διάγραμμα της Εικόνας 45, όπως αυτή αποκτήθηκε από την ιστοσελίδα του VirusTotal.



Εικόνα 45: Αριθμός υποβολής τύπων αρχείων στην ιστοσελίδα VirusTotal.com

Η έρευνα στον σκληρό δίσκο έγινε σε αντιπαραβολή με τα *Yara* αρχεία first_yara_rule.yar, second_yara_rule.yar, yarGen_directory_scan.yar και το αποτέλεσμα ήταν πως, εκτός από κάποια σφάλματα που προέκυψαν λόγω αδυναμίας ανοίγματος κάποιων αρχείων ήχου, καμιά έρευνα δεν έδειξε κάποιο λανθασμένο συναγερμό, ενώ ο σκληρός δίσκος σημάνθηκε

ως καθαρός, όπως φαίνεται το παράδειγμα στην Εικόνα 46.



Εικόνα 46: Έρευνα εντοπισμού αρχείου τύπου KeeFarce σε σκληρό δίσκο

Στο δεύτερο μέρος του ελέγχου της εγκυρότητας των κανόνων, αυτοί εξετάστηκαν σε αντιπαραβολή με δύο φακέλους στους οποίους είχε συγκεντρωθεί αρκετά μεγάλος αριθμός από δείγματα κακόβουλων λογισμικών. Ο ένας φάκελος περιείχε κακόβουλα λογισμικά που συλλέχθηκαν από το εργαλείο *maltrieve.*

Το εργαλείο *maltrieve* χρησιμοποιείται για την συλλογή κακόβουλων λογισμικών από διάφορες ιστοσελίδες. Εκτός από την δυνατότητα της αυτοματοποιημένης συλλογής κακόβουλων λογισμικών το *maltrieve* μας δίνει την δυνατότητα να κατεβάζουμε και τους πιο πρόσφατους ιούς, έχοντας μία συνεχώς ανανεωμένη Βάση Δεδομένων κακόβουλών λογισμικών προς ανάλυση. Οι ιστοσελίδες τις οποίες χρησιμοποιεί για να αντλεί τα κακόβουλα λογισμικά είναι οι εξής:

- *Malc0de*
- *Malware Domain List*
- *Malware URLs*
- *VX Vault*
- *URLquery*
- *CleanMX*
- *ZeusTracker*

Το εργαλείο *theZoo* επιτελεί ένα παρόμοιο σκοπό με το *maltrive,* ωστόσο έχει κάποιες σημαντικές διαφορές στην φιλοσοφία. Αρχικά, το *theZoo* δεν δίνει την δυνατότητα για κατέβασμα κακόβουλων λογισμικών από ιστοσελίδες, αλλά έρχεται με μία προεγκατεστημένη Βάση Δεδομένων με κακόβουλα λογισμικά. Η Βάση αυτή ανανεώνεται με κάθε έκδοση του εργαλείου. Για την χειροκίνητη ανανέωση της Βάσης, ο εκάστοτε αναλυτής μπορεί να υποβάλει ένα δείγμα για ένα νέο κακόβουλο λογισμικό, μαζί με κάποιες πληροφορίες, και να το υποβάλει δημόσια στην Βάση Δεδομένων για το βλέπουν και άλλοι. Ακόμη, το *theZoo* είναι σχεδιασμένο για την οργανωμένη αποθήκευση των κακόβουλων λογισμικών. Αυτό πρακτικά σημαίνει ότι τα λογισμικά που έχει, είναι αποθηκευμένα σε μία Βάση Δεδομένων και το εργαλείο προσφέρει κάποιες δυνατότητες προσπέλασης αυτών και άντληση βασικών πληροφοριών με ασφαλή τρόπο. Συνεπώς το εργαλείο έχει την φιλοσοφία της παροχής υπηρεσιών αποθήκευσης και διαχείρισης των δειγμάτων των κακόβουλων λογισμικών του αναλυτή, ενώ το *maltrieve* προσανατολίζεται στην παροχή της δυνατότητας απόκτησης σε ένα απλό φάκελο, διαφόρων σύγχρονων κακόβουλων λογισμικών.

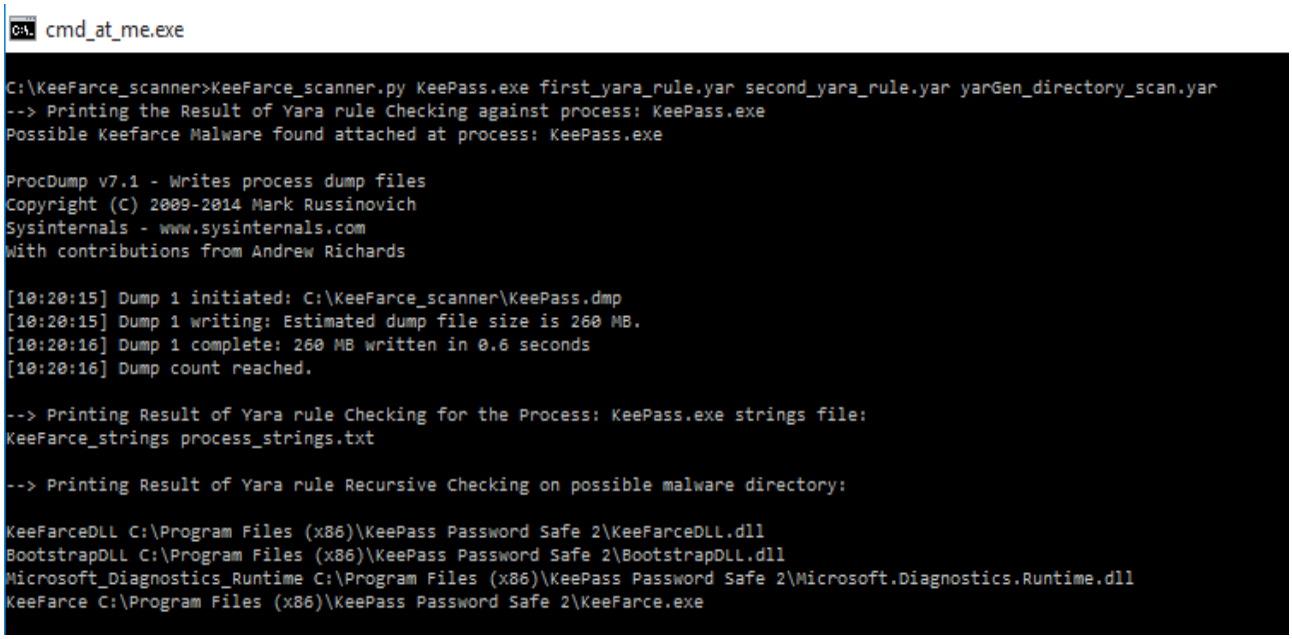
Η έρευνα έγινε και στους δύο φακέλους σε αντιπαραβολή με τους κανόνες αρχεία *first_yara_rule.yar*, *second_yara_rule.yar, yarGen_directory_scan.yar*. Το αποτέλεσμα ήταν να μην εγερθεί κανένας λανθασμένος συναγερμός.

Γενικό αποτέλεσμα των ερευνών διαπίστωσης της εγκυρότητας των κανόνων που δημιουργήσαμε, ήταν ότι κανένας δεν προκάλεσε λανθασμένο συναγερμό και παραπλάνηση, είτε δοκιμάστηκε σε αντιπαραβολή με καλόβουλα, είτε με κακόβουλα αρχεία. Άρα, μπορούμε με αρκετή ασφάλεια να συμπεράνουμε ότι οι εν λόγω κανόνες αποτελούν εξαιρετικούς σηματοδότες της κακόβουλης δραστηριότητας του *KeeFarce,* χωρίς να υπάρχει πρόβλημα αξιοπιστίας.

5.2.6 Αυτοματοποιημένη Διαδικασία εντοπισμού του *KeeFarce*

Στην προσπάθεια για την εκπόνηση ολοκληρωμένης μελέτης του κακόβουλου λογισμικού *KeeFarce* και κατόπιν της εξαγωγής κανόνων ικανών για έγκυρη ανίχνευση, δημιουργήθηκε ένα σενάριο σε γλώσσα *Python* για την αυτοματοποίηση της διαδικασίας εντοπισμού του *KeeFarce.* Πιο συγκεκριμένα, το σενάριο έχει σκοπό την διάγνωση μόλυνσης της διεργασίας *KeePass* από το λογισμικό *KeeFarce,* την απόλυτη ταυτοποίηση του επιτιθέμενου με το εν λόγω κακόβουλο λογισμικό και την εύρεση στον σκληρό δίσκο των αρχείων που αποτελούν

την απειλή και σχετίζονται με το λογισμικό (Εικόνα 47) .



Εικόνα 47: Εκτέλεση εργαλείου KeeFarce_scanner

Το συγκεκριμένο σενάριο επιστρατεύει άλλα εργαλεία (στον ίδιο φάκελο) για να επιτελέσει την λειτουργία της ανίχνευσης. Η λειτουργία του σε βήματα περιγράφει ως εξής:

- Αρχικά, επιστρατεύεται το *Standalone* εκτελέσιμο αρχείο *yara64.exe.* Με την χρήση αυτού επιτυγχάνεται το σκανάρισμα της διεργασίας *KeePass.exe* για την εύρεση συγκεκριμένων αλφαριθμητικών που ταυτοποιούν την μόλυνση της μνήμης της διεργασίας από το *KeeFarce.* Στην περίπτωση που ανιχνευτεί μόλυνση, παράγεται ένα αρχείο *.dmp* της διεργασίας με το εργαλείο *procdump.exe.*
- Στην φάση αυτή εξάγεται από το αρχείο *.dmp, KeePass.dmp,* το αρχείο *.txt* με όλα τα αλφαριθμητικά που περιέχει, με την βοήθεια του εργαλείου *strings.exe.*
- Το αρχείο με τα αλφαριθμητικά ελέγχεται μέσω του *yara64.exe* για περαιτέρω εύρεση στοιχείων προσβολής καθώς μετά από πολλές δοκιμές αποδείχθηκε ότι το αποτέλεσμα αυτού του βήματος δίνει πιο συγκεκριμένα αποτελέσματα από το πρώτο στάδιο.
- Τέλος, με την χρήση κάποιων συναρτήσεων εντοπίζεται ο φάκελος εγκατάστασης του *KeePass* και δίνεται η διαδρομή του δίσκου του φακέλου σαν είσοδος στο *yara64.exe* για τον πιθανό εντοπισμό κακόβουλών αρχείων, αξιοποιώντας την γνώση ότι το *KeeFarce* για να λειτουργήσει χρειάζεται να βρίσκεται αυτό και οι βιβλιοθήκες που αξιοποιεί στον ίδιο φάκελο με το *KeePass.exe.* Επομένως σε περίπτωση που με κάποιον τρόπο ένας *Hacker* προσβάλλει τον υπολογιστή του θύματος και εγκαταστήσει το *KeeFarce,* το σενάριο μπορεί αυτομάτως να εκτελεστεί παράλληλα

με την λειτουργία του *KeePass* για την ανίχνευση της προσβολής.

Για την λειτουργία του παραπάνω σεναρίου απαιτούνται ως είσοδος τρία αρχεία *.yar* τα οποία περιέχουν τους κανόνες με τους οποίους γίνεται η αντιπαραβολή για την ανίχνευση της κακόβουλης ύπαρξης, καθώς και τα τρία *standalone* αρχεία που αναφέραμε.

5.2.7 Ανάλυση στο Cuckoo SandBox

Η ανάλυση στο *Cuckoo Sandbox* αποτελεί μία επιπλέον διαδικασία αυτοματοποιημένης τεχνικής ανάλυσης, η οποία έρχεται να συμπληρώσει την μελέτη ενός κακόβουλου δείγματος. Η ανάλυση με αυτόν τον τρόπο δεν προσφέρει κάποιο επιπλέον επίπεδο διόρασης στην δομή ή στην συμπεριφορά ενός κακόβουλου λογισμικού, δίνοντας ως αποτέλεσμα παρόμοιες πληροφορίες με αυτές που έχουμε εξάγει μέχρι στιγμής. Τα πλεονεκτήματα της τεχνικής, ωστόσο, δεν έγκειται στο βάθος διόρασης αλλά στον τρόπο διενέργειας της ανάλυσης.

Καταρχάς, η ανάλυση πραγματοποιείται σε ένα απολύτως ελεγχόμενο περιβάλλον. Το χαρακτηριστικό αυτό είναι υψίστης σημασίας όταν ο αναλυτής έχει να αντιμετωπίσει κάποιο άκρως επικίνδυνο κακόβουλο λογισμικό το οποίο μπορεί να προκαλέσει ανεπανόρθωτη βλάβη στο σύστημα του, όπως το λογισμικό *CryptoLocker,* το οποίο κρυπτογραφεί δεδομένα του σκληρού δίσκου. Άρα με το *Cuckoo* καταφέραμε να μελετήσουμε τα κακόβουλα λογισμικά τηρώντας την απαραίτητη απόσταση ασφαλείας. Εν συνεχεία, η διαδικασία είναι πλήρως αυτοματοποιημένη, γεγονός που συμβάλει στην εξαγωγή κάποιων πρώτων συμπερασμάτων στην μελέτη ενός δείγματος, τα οποία θα μας οδηγήσουν σε περαιτέρω έρευνα με μειωμένο εύρος. Τέλος, η αναφορά που θα προκύψει, υπό συνθήκες, μπορεί να δώσει όλα τα χαρακτηριστικά πάνω στα οποία θα βασιστούμε για να εντοπίζουμε παρόμοια λογισμικά στο μέλλον.

Μοναδικό, ίσως, μειονέκτημα του εργαλείο, όπως διαπιστώθηκε από εκτενείς δοκιμές, είναι κάποιοι περιορισμοί που θέτει στην ανάλυση. Στον χρόνο εκπόνησης αυτής της έρευνας, το *Cuckoo Sandbox RC1* αποτελεί την τελευταία έκδοση του εργαλείου και η οποία παρουσιάζει σχετικά μειωμένη συμβατότητα με τα *Windows 8.1* και τα *Windows 10,* ενώ παρέχει πλήρη συμβατότητα με τα *Windows 7* και τα *Windows XP.* Παράλληλα, δεν είναι εύκολη η μελέτη κακόβουλων δειγμάτων τα οποία για να λειτουργήσουν χρειάζονται να βρίσκονται σε

συγκεκριμένη διαδρομή στο δίσκο, τόσο αυτά όσο και τα υπόλοιπα στοιχεία που χρειάζονται να εκτελεστούν. Ο τελευταίος περιορισμός είναι καταλυτικός για την μελέτη του *KeeFarce.* Αρχικά να αναφέρουμε πώς το εν λόγω κακόβουλο λογισμικό δεν μπορεί να εκτελεστεί επιτυχώς, στην μορφή που δίνεται ως προ-μεταγλωτισμένο, στα *Windows 7* και για παλαιότερα λογισμικά. Παράλληλα, ενώ για να εκτελεστεί θα πρέπει αυτό και οι βιβλιοθήκες του να βρίσκονται στον ίδιο φάκελο με το *KeePass.exe,* η δυνατότητα αυτή δεν προσφέρεται από το *Cuckoo.* Το εργαλείο δέχεται ως είσοδο κάποιο δείγμα και προβαίνει στην μελέτη του, επιλέγοντας κάθε φορά διαφορετική τοποθεσία στον δίσκο, στην οποία θα εκτελέσει το δείγμα για να το μελετήσει. Αυτό το χαρακτηριστικό λειτουργεί χάριν της επιτυχίας της ανάλυσης για κακόβουλα δείγματα, προσομοιώνοντας την πραγματική συμπεριφορά ενός κακόβουλου λογισμικού, αλλά δεν είναι κατάλληλη για την μελέτη του *KeeFarce.* Παρόλα αυτά, πραγματοποιήθηκε εξέταση του κακόβουλου λογισμικού *KeeFarce* σε λειτουργικό σύστημα *windows 7* και παρόλη την αστάθεια που παρουσιάζει η χρήση του εκεί, η αναφορά της ανάλυσης του *KeeFarce.exe* δίνει κάποια πολύ σημαντικά στοιχεία για την λειτουργία του. Επιπροσθέτως, θεωρήσαμε χρήσιμο να παρουσιάσουμε και τα συμπεράσματα από την ανάλυση δύο άλλων κακόβουλων λογισμικών, τα οποία εμπίπτουν σε παρόμοια κατηγορία συμπεριφοράς και σκοπού προσβολής με το *KeeFarce* και στα οποία μπορούμε να πραγματοποιήσουμε μία πλήρη αυτοματοποιημένη ανάλυση με το *Cuckoo Sandbox* σε εικονικό λειτουργικό *Windows 7.* Αυτά είναι το κακόβουλο λογισμικό *Alina* και το κακόβουλο λογισμικό *Zeus* και η μελέτη τους θα παρουσιαστεί στο παράρτημα Β. Παρακάτω θα αναφερθούμε στην αυτοματοποιημένη ανάλυση του λογισμικού *KeeFarce.*

Παραμετροποίηση Cuckoo Sandbox

Η διεξαγωγή της έρευνας πραγματοποιήθηκε σε σύστημα ξενιστή με λειτουργικό σύστημα *Ubuntu Linux 14.04.* Η διαδικασία της επιτυχούς παραμετροποίησης της εικονικής μηχανής και του *Cuckoo Sandbox* έχουν ως εξής:

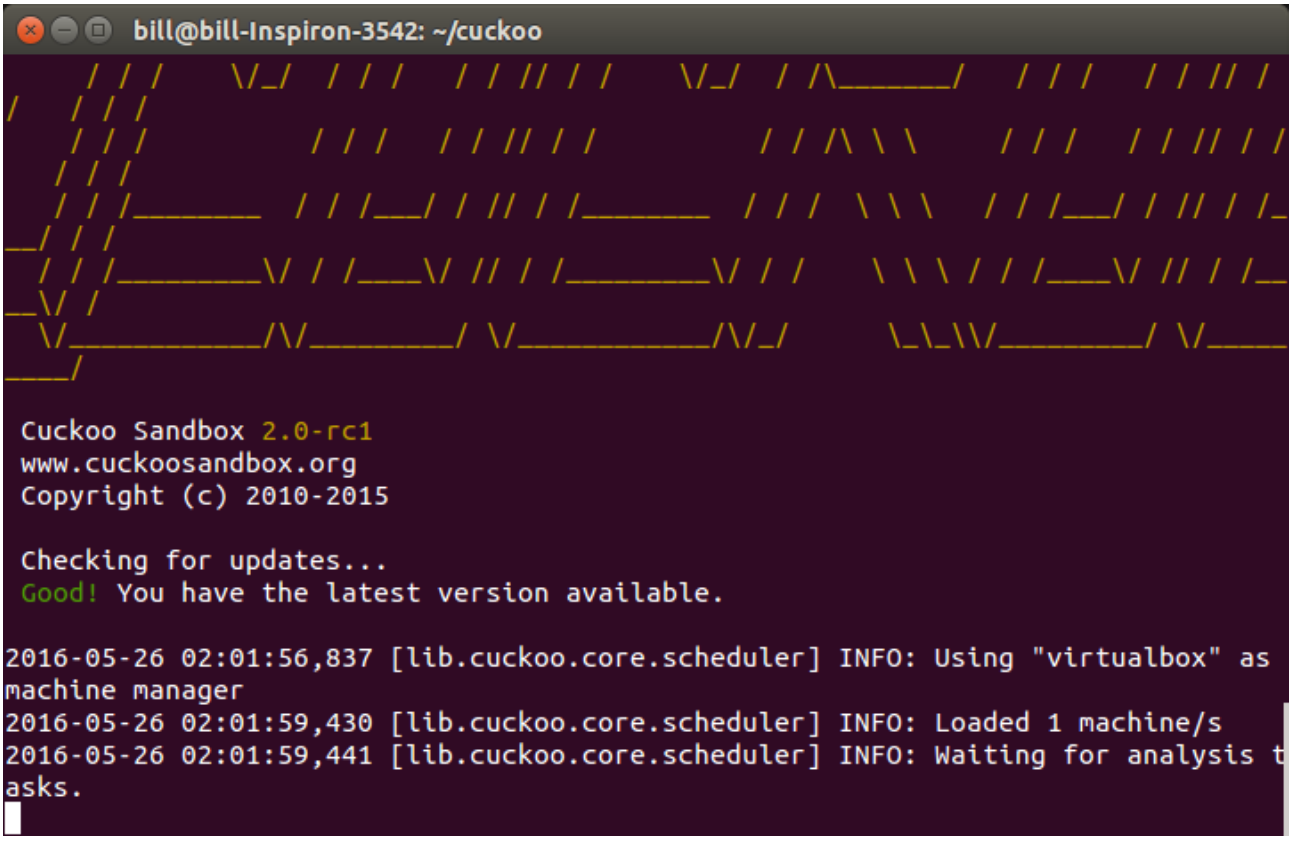
- Επιλογή λειτουργικού συστήματος εικονικής μηχανής. Εμείς για την ανάλυση του προκείμενου κακόβουλου λογισμικού χρησιμοποιήσαμε λειτουργικό σύστημα *Windows 7.*
- Εγκατάσταση του πακέτου της γλώσσας *Python 2.7*
- Δημιουργία ενός φακέλου κοινής χρήσης για τα λειτουργικά συστήματα *Guest* και *Host.* Αυτό είναι απαραίτητο για την κοινή χρήση αρχείων.
- Απενεργοποίηση του τείχους προστασίας των *Windows* και του *User Access Control (UAC).*

- Παραμετροποίηση του δικτύου της εικονικής μηχανής και επιλογή των κατάλληλων διευθύνσεων *IP* για την ομαλή επικοινωνίας *Guest – Host – Cuckoo Sandbox.*
- Εκτέλεση σε τερματικό παράθυρο με δικαιώματα διαχειριστή, του αρχείου *Agent.py* για την επίτευξη της λειτουργίας του *Cuckoo.*
- Παραμετροποίηση αρχείων ρυθμίσεων του *Cuckoo Sandbox.*
- Εκτέλεση της εφαρμογής *KeePass.exe* και λήψη ενός στιγμιοτύπου συστήματος, το οποίο θα χρησιμοποιηθεί για εκκίνηση της εικονικής μηχανής με συγκεκριμένη κατάσταση λειτουργίας από το *Cuckoo Sandbox.*

Κατόπιν εκτέλεσης όλων των παραπάνω ενεργειών, το στιγμιότυπο του συστήματος θα περιέχει ανοιχτά ένα παράθυρο του *KeePass* και ένα τερματικό με ανοιχτό το αρχείο *Agent.py* .

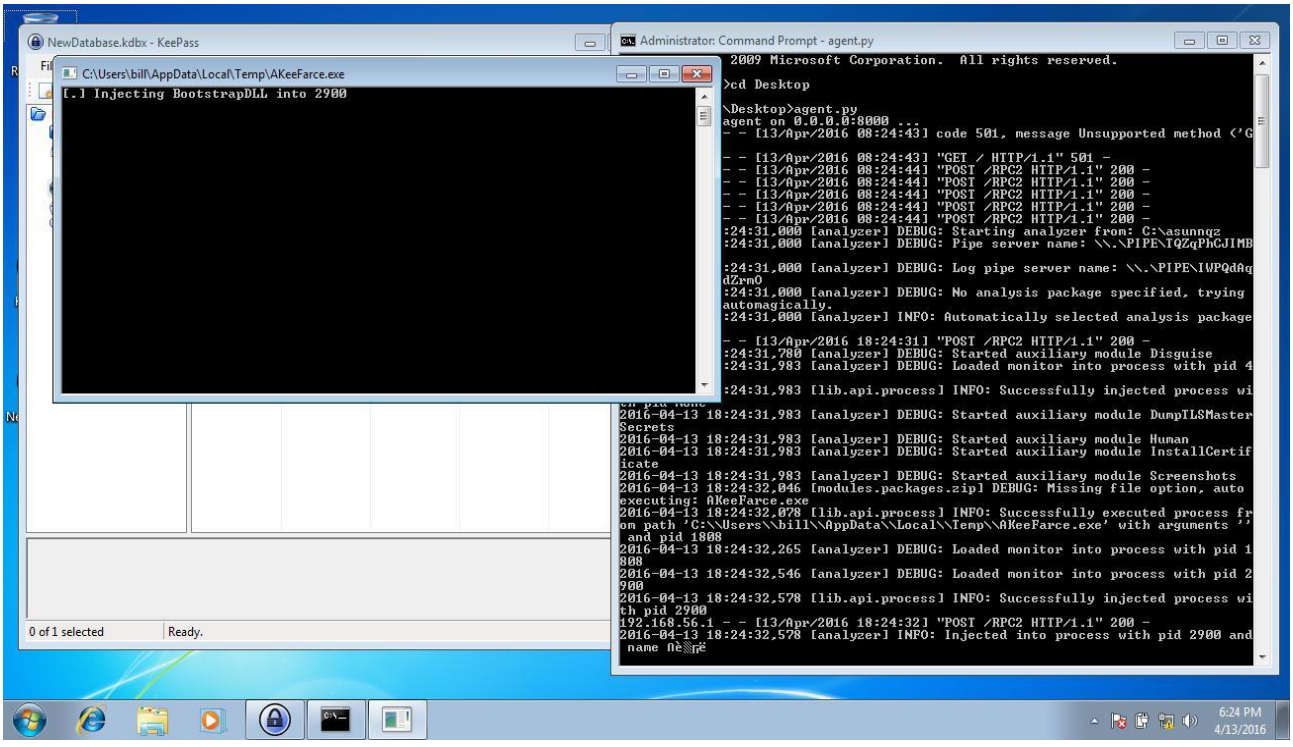
Διενέργεια Ανάλυσης

Κατόπιν επιτυχούς παραμετροποίησης της μερών που θα χρησιμοποιηθούν για την αυτοματοποιημένη ανάλυση, πρέπει να γίνει έναρξη του Cuckoo Sandbox σε ένα καινούργιο τερματικό παράθυρο με την εντολή *python cuckoo.py* (Εικόνα 48).



Εικόνα 48: Το Cuckoo Sandbox σε λειτουργία έτοιμο να υποδεχθεί δείγμα προς εξέταση

Στην συνέχεια, ανοίγοντας ένα νέο τερματικό παράθυρο, με την εντολή *python submit.py - - parameters - αρχείο προς εξέταση,* επιτυγχάνεται η υποβολή του κακόβουλου δείγματος που θέλουμε να υποβληθεί για εξέταση. Εντός των παραμέτρων καθορίζεται μεταξύ πολλών άλλων και ο χρόνος τερματισμού της ανάλυσης και το είδος του εξεταζόμενου αρχείου. Στην περίπτωση μας, παρουσιάστηκε πρόβλημα ως προς την αδυναμία του *Cuckoo* να εξετάζει αρχεία σχετιζόμενα μεταξύ τους, όπως είναι οι 3 βιβλιοθήκες που συνοδεύουν το *KeeFarce.exe* και είναι απαραίτητες για την λειτουργία του. Το εν λόγω εμπόδιο προσπεράστηκε με την χρήση μίας ιδιότητας του *Cuckoo,* η οποία αφορά στο τρόπο με τον οποίο λειτουργεί το πακέτο ανάλυσης *zip* και το οποίο δίνεται και ως παράμετρος της ανάλυσης. Όταν ζητηθεί από το *Cuckoo* να αναλύσει ένα αρχείο με επέκταση *zip,* τότε αυτό αφού κάνει μία γενική ανάλυση του φακέλου, το αποσυμπιέζει και υποβάλλει κάθε ένα από τα περιεχόμενά του σε ανάλυση. Για να το κάνει αυτό, τα αρχεία του φακέλου αποσυμπιέζονται στην ίδια διαδρομή δίσκου μέσα στο εικονικό λειτουργικό. Συνεπώς, συμπιέζοντας τον φάκελο με όλα τα αρχεία του προ-μεταγλωτισμένου *KeeFarce* και υποβάλλοντας τον ολόκληρο για ανάλυση με μορφή *zip,* έχοντας παράλληλα μετονομάσει το αρχείο *KeeFarce.exe* σε *AKeeFarce.exe* για να υποβληθεί πρώτο για εξέταση, επιτυγχάνουμε το όσον το δυνατόν καλύτερο αποτέλεσμα της ανάλυσης του κακόβουλου λογισμικού. Το αποτέλεσμα που προκύπτει φαίνεται στην Εικόνα 49.



Εικόνα 49: Κατά την διάρκεια εκτέλεσης του Cuckoo SandBox

Παρατηρούμε ότι το λογισμικό *KeeFarce.exe,* το οποίο έχει αναγνωριστικό το νούμερο 1808, προσπαθεί να κάνει *inject* στην διαδικασία με αναγνωριστικό 2900, η οποία δεν είναι άλλη από το ανοιχτό παράθυρο *KeePass.exe.* Το δεξί παράθυρο είναι εκείνο που φαίνεται η εξέλιξη της διαδικασίας αυτοματοποιημένης ανάλυσης από το *Cuckoo.* Κατόπιν πέρατος του επιλεγμένου χρόνου ανάλυσης, παράγεται μεταξύ άλλων χρήσιμων αρχείων ανάλυσης και ένα αρχείο με *report.json.* Το αρχείο αυτό εμπεριέχει συμπυκνωμένη όλες τις πληροφορίες που χρειαζόμαστε για να εξάγουμε χρήσιμα συμπεράσματα. Στη συνέχεια παρουσιάζονται και αναλύονται οι βασικές κατηγορίες που αποτελούν το αποτέλεσμα της ανάλυσης:

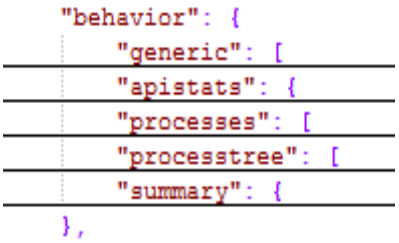
- ***Info.*** Στο πεδίο αυτό παρέχονται γενικές πληροφορίες για το υπό εξέταση αρχείο.
- **Signatures.** Στο πεδίο αυτό αναφέρονται κατηγορίες με μοτίβα συμπεριφοράς, τα οποία είναι προαποθηκευμένα στο εργαλείο (γίνεται να εισαχθούν και νέες υπογραφές) και στις οποίες εμπίπτουν και κάποια σημάδια συμπεριφοράς του κακόβουλου λογισμικού. Συγκεκριμένα για το *KeeFarce,* ανιχνεύτηκαν δύο ύποπτες συμπεριφορές (Εικόνα 50). Η πρώτη αφορά στην ύπαρξη κάποιων *Buffers,* οι οποίοι υποδεικνύουν πιθανότητα ύπαρξης *injected* κώδικα. Τη δεύτερη υπογραφή αφορά στην ύπαρξη μέτρων καταστολής της δράσης του *KeeFarce,* σε περίπτωση που αυτό εκτελείται μέσα σε ένα λογισμική αμμοδόχο.


```
"signatures": [
    {
        "families": [],
        "description": "One or more potentially interesting buffers were extracted, these generally contain injected code, configuration data, etc.",
        "severity": 2,
        "marks": [],
        "references": [],
        "name": "dumped_buffer"
    },
    {
        "families": [],
        "description": "Looks for the Windows Idle Time to determine the uptime",
        "severity": 3,
        "marks": [
        "references": [],
        "name": "antisandbox_idletime"
    }
],
```


Εικόνα 50: Υπογραφές ταυτοποίησης Κακόβουλου Λογισμικού από το Cuckoo SandBox

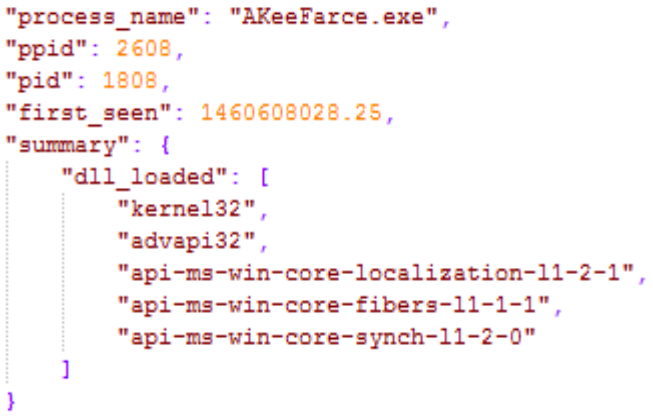
- **Target.** Το πεδίο αυτό αφορά στο τύπο αρχείου το οποίο έχει υποβληθεί για ανάλυση και περιέχει κάποιες βασικές πληροφορίες.
- **VirusTotal.** Το πεδίο αυτό αφορά στην ύπαρξη πληροφοριών κακόβουλης οντότητας στη ιστοσελίδα *VirusTotal* για το υπο εξέταση αρχείο. Στην δική μας περίπτωση δεν διακριβώθηκε κάποια αντιστοίχηση με την ιστοσελίδα καθώς το κακόβουλο λογισμικό δεν είχε ακόμα μελετηθεί στον χρόνο διενέργειας της έρευνας μας.
- **Behaviour.** Στο πεδίο αυτό κατηγοριοποιούνται οι ανιχνευμένες συμπεριφορές και

αναλύονται. Στην δική μας περίπτωση παρατηρήθηκαν τα παρακάτω όπως φαίνονται και στην Εικόνα 51.



Εικόνα 51: Κατηγοριοποίηση Ανιχνεύσιμων Συμπεριφορών

Αναλύοντας διαδοχικά μία μία τις πληροφορίες παρατηρούμε, αρχικά σχετικά με το *generic*, ότι υπάρχει μία διεργασία με όνομα *AKeeFarce.exe.* Αυτή έχει αναγνωριστικό 1808 και έχει φορτωμένες τις βιβλιοθήκες που φαίνονται στην Εικόνα 52.



Εικόνα 52: Ανάλυση διεργασίας AKeeFarce.exe

Στην συνέχεια παρατηρούμε ποια *apis* χρησιμοποιεί η κάθε διαδικασία και στην περίπτωση του *AKeeFarce* παρατηρούμε πολλές διεπαφές ύποπτης χρήσης (Εικόνα 53). Αυτή είναι και η πιο ισχυρή ένδειξη ότι η διεργασία αποτελεί κακόβουλο λογισμικό καθώς σχεδόν όλες οι διεπαφές αντιστοιχούν σε συναρτήσεις αποσφαλμάτωσης κάποιας άλλης διεργασίας. Συνεπώς καταλαβαίνουμε ότι η εν λόγω διεργασία πιθανόν προσπαθεί να αποκτήσει πρόσβαση στον χώρο διευθύνσεων μίας άλλης διεργασίας με χρήση *dll injection* τεχνικής με την ακολουθία βημάτων, όπως αυτή περιγράφηκε στο κεφάλαιο 1.

```
"1808": {
    "CreateToolhelp32Snapshot": 1,
    "Process32NextW": 30,
    "NtAllocateVirtualMemory": 1,
    "WriteProcessMemory": 1,
    "LdrGetProcedureAddress": 54,
    "SetUnhandledExceptionFilter": 1,
    "NtOpenProcess": 1,
    "GetFileType": 3,
    "GetSystemTimeAsFileTime": 1,
    "LdrLoadDll": 8,
    "LdrGetDllHandle": 3,
    "Process32FirstW": 1,
    "NtClose": 1
}
```


Εικόνα 53: Ύποπτες Διεπαφές

Για κάθε μία από τις κλήσεις διεπαφών, μπορούμε να δούμε πιο συγκεκριμένες πληροφορίες από το πεδίο *processes* (Εικόνα 54).


```
"category": "process",
"status": 1,
"stacktrace": [],
"api": "CreateToolhelp32Snapshot",
"return_value": 84,
"arguments": {
    "flags": 2,
    "process_identifier": 0
},
"time": 1460608153.25,
"tid": 248,
"flags": {}
```


Εικόνα 54: Λεπτομέρειες για τις χρησιμοποιούμενες Διεπαφές

- **Debug.** Το πεδίο αυτό αφορά σε πληροφορίες *log* της διαδικασίας της ανάλυσης (Εικόνα 55).


```
"debug": {
    "errors": [],
    "log": [
        "2016-04-13 18:24:31,000 [analyzer] DEBUG: Starting analyzer from: C:\\asunnqs\n",
        "2016-04-13 18:24:31,000 [analyzer] DEBUG: Pipe server name: \\\\.\\PIPE\\TQZqPhCJIMBZajxl\n",
        "2016-04-13 18:24:31,000 [analyzer] DEBUG: Log pipe server name: \\\\.\\PIPE\\IWPQdAqGArHaXIWApaTNdZrm0\n",
        "2016-04-13 18:24:31,000 [analyzer] DEBUG: No analysis package specified, trying to detect it automagically.\n",
        "2016-04-13 18:24:31,000 [analyzer] INFO: Automatically selected analysis package \"zip\"\n",
        "2016-04-13 18:24:31,780 [analyzer] DEBUG: Started auxiliary module Disguise\n",
        "2016-04-13 18:24:31,983 [analyzer] DEBUG: Loaded monitor into process with pid 496\n",
        "2016-04-13 18:24:31,983 [lib.api.process] INFO: Successfully injected process with pid None\n",
        "2016-04-13 18:24:31,983 [analyzer] DEBUG: Started auxiliary module DumpTLSMasterSecrets\n",
        "2016-04-13 18:24:31,983 [analyzer] DEBUG: Started auxiliary module Human\n",
        "2016-04-13 18:24:31,983 [analyzer] DEBUG: Started auxiliary module InstallCertificate\n",
        "2016-04-13 18:24:31,983 [analyzer] DEBUG: Started auxiliary module Screenshots\n",
        "2016-04-13 18:24:32,046 [modules.packages.zip] DEBUG: Missing file option, auto executing: AKeeFarce.exe\n",
        "2016-04-13 18:24:32,078 [lib.api.process] INFO: Successfully executed process from path 'C:\\\\Users\\\\bill\\\\AppData\\\\Local\\\\Temp\\\\AKeeFarce.exe' with arguments '' and pid 1808\n",
        "2016-04-13 18:24:32,265 [analyzer] DEBUG: Loaded monitor into process with pid 1808\n",
        "2016-04-13 18:24:32,546 [analyzer] DEBUG: Loaded monitor into process with pid 2900\n",
        "2016-04-13 18:24:32,578 [lib.api.process] INFO: Successfully injected process with pid 2900\n",
        "2016-04-13 08:26:45,836 [analyzer] INFO: Analysis timeout hit, terminating analysis.\n",
        "2016-04-13 08:26:45,836 [analyzer] INFO: Analysis completed.\n"
    ]
},
```


Εικόνα 55: Πληροφορίες Log

- **Strings.** Το συγκεκριμένο πεδίο αφορά στην παρουσίαση όλων των αλφαριθμητικών που υπάρχουν εντός του φακέλου που υποβάλλαμε για εξέταση.
- **Network.** Είναι το πεδίο στο οποίο παρουσιάζονται όλα τα σχετικά στοιχεία με την κίνηση δικτύου του εικονικού λειτουργικού κατά την υποβολή και δράση του κακόβουλου λογισμικού (Εικόνα 56).


```
"network": {
    "tls": [],
    "udp": [
    "icmp": [],
    "http": [],
    "smtp": [],
    "tcp": [],
    "mitm": [],
    "hosts": [],
    "pcap_sha256": "0892e8ac769fbaf07be88ce523e55cc6fb7dddf39c07ad7184b4fa256b746ecf",
    "dns": [],
    "domains": [],
    "dead_hosts": [],
    "sorted_pcap_sha256": "0b2be6689eb4389d066d8c99778032c441e26bd14f2783e3745f62bd28827223",
    "irc": []
}
```


Εικόνα 56: Πληροφορίες χρήσης Δικτύου

Όλα τα παραπάνω στοιχεία είναι ικανά να μας οδηγήσουν σε κάποια διάγνωση σχετικά με το υποβαλλόμενο αρχείο. Μαζί με την παραπάνω αναφορά, το *cuckoo* εξάγει και άλλα στοιχεία τα οποία βοηθούν στην δημιουργία μίας εικόνας του τι γίνεται στο σύστημα μας κατά την δράση ενός λογισμικού (π.Χ. στιγμιότυπα οθόνης). Η παραπάνω ανάλυση, ωστόσο, δεν περιέχει τον απαιτούμενο αριθμό στοιχείων για την εξαγωγή κατάλληλων συμπερασμάτων καθώς, όπως είπαμε, το *KeeFarce* δεν είναι απολύτως συμβατό με *Windows 7,* στα οποία είναι εφικτή η ανάλυση με *Cuckoo sandbox.*

6. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η εκπόνηση της παρούσας διπλωματικής είχε ως στόχο την παρουσίαση της διεξαγωγής μίας πλήρους ανάλυσης σε ένα κακόβουλο δείγμα, με επαρκές βάθος, σε όλα τα επίπεδα και με χρήση των πλέον σύγχρονων τεχνοτροπιών και μεθόδων. Η ίδια η ανάλυση δεν έφτασε σε επίπεδο αντίστροφής μηχανικής καθώς αυτό δεν αποτελούσε αναγκαίο στην περίπτωση του λογισμικού που εξετάσαμε. Ωστόσο η πληροφορίες που εξάχθηκαν κρίνονται υπεραρκετές τόσο για την κατανόηση του *KeeFarce* όσο και για την ύψωση μέτρων προστασίας κατάλληλων για την αποφυγή του ίδιου αλλά και άλλων λογισμικών παρόμοιων που μπορεί να εμφανιστούν και να συμπεριληφθούν στην ίδια οικογένεια κακόβουλων λογισμικών.

Όσον αφορά στο θέμα των μέτρων ασφαλείας, αυτά υλοποιήθηκαν στην μορφή κανόνων *Yara,* αποτελώντας το απαύγασμα της όλης μελέτης. Η πορεία για την επίτευξη του στόχου της πτυχιακής υπήρξε ενδελεχής ώστε να προκύψουν κανόνες με απόλυτη ακρίβεια και έλλειψη φαινομένων έγερσης λανθασμένων συναγερμών. Η εγκυρότητα των κανόνων διαπιστεύτηκε με αυστηρά κριτήρια δοκιμών και το αποτέλεσμα δύναται να αποτελέσει άριστο δείγμα προστασίας απέναντι σε πιθανές υλοποιήσεις της *KeeFarce* λογικής με την πραγματοποίηση μικρών αλλαγών.

Στην πορεία της διαδικασίας εκπόνησης της μελέτης εξήχθησαν επιμέρους συμπεράσματα, τα οποία αφενός αποτελούν την βάση κατανόησης του λογισμικού που εξετάσθηκε, αφετέρου δύναται να αποτελέσουν ικανή βάση για περαιτέρω μελέτη ίδιου τύπου κακόβουλων λογισμικών. Παράλληλα, με αναγωγή σε ποιο γενικές περιπτώσεις κακόβουλων λογισμικών, η χρήση της μεθοδολογίας που αναπτύχθηκε στην παρούσα πτυχιακή εργασία, μπορεί να χρησιμοποιηθεί για την εξαγωγή συμπερασμάτων και για άλλα πολύ πιο επικίνδυνα κακόβουλα λογισμικά, με βασικό πλεονέκτημα την ασφάλεια του ίδιου του συστήματος του αναλυτή, η οποία εξασφαλίζεται σε μεγάλο βαθμό στην εκπονηθείσα μελέτη.

Η χρήση σχετικά νέων τεχνολογιών διεξαγωγής και υλοποίησης της ανάλυσης κακόβουλών λογισμικών μας προκάλεσε να ασχοληθούμε με δύο από πιο δημοφιλή λειτουργικά συστήματα, *Windows* και *Ubuntu Linux.* Η εμπειρία που αποκτήθηκε και παρουσιάστηκε από την χρήση τεχνικών και εργαλείων και από τους δύο κόσμους, αποτελεί πολύτιμο αρωγό στην διεξαγωγή μελετών ολοκληρωμένων και με ασφάλεια, αξιοποιώντας παράλληλα τις

καλύτερες δυνατότητες και από τα δύο λειτουργικά. Τα κακόβουλο λογισμικό που μελετήθηκε επηρεάζει συστήματα με λειτουργικό *Windows,* ωστόσο η χρήση *Linux* για την διεξαγωγή τμημάτων της μελέτης συνείσφερε με κατακόρυφη αύξηση της ασφάλειας της διαδικασίας και χρήση πολύ εξειδικευμένων τεχνοτροπιών για την εξαγωγή συμπερασμάτων σε δεύτερο χρόνο.

Όσον αφορά στο ίδιο το κακόβουλο λογισμικό που εξετάστηκε, αυτό στον χρόνο διεξαγωγής της έρευνας παρέμενε ανεξερεύνητο σε παγκόσμιο επίπεδο. Στις δημοφιλείς ιστοσελίδες *VirusTotal* και *Malwr,* οι οποίες πρωτοστατούν στην παρουσίαση αναλύσεων κακόβουλων λογισμικών, δεν υπήρχε κάποια αναφορά σχετικά με το *KeeFarce* μέχρι και τα μέσα Φεβρουαρίου του 2016. Ακόμα και όταν παρουσιάστηκαν οι πρώτες αναφορές στο λογισμικό, δεν κάλυπταν εκείνα τα αλφαριθμητικά τα οποία παρουσιάζονται εντός του λογισμικού - θύματος και αποτελούν μονοσήμαντη απόδειξη της προσβολής του συστήματος. Έτσι, η δική μας προσέγγιση ήταν προσανατολισμένη στον έλεγχο του προγράμματος *KeePass* για την ανίχνευση της μόλυνσης του ή όχι, κάτι το οποίο δεν είχε πραγματοποιηθεί. Παράλληλα αναπτύχθηκε και ένα σενάριο τύπου *Python* η χρήση του οποίου αφορά στον εντοπισμό, ανά πάσα στιγμή, της προσβολής του συστήματος από το *KeeFarce,* μέσω ελέγχου της διεργασίας του *KeePass,* με χρήση κανόνων *Yara.* Προέκταση αυτής της πρακτικής μπορεί να εφαρμοστεί για έλεγχο προσβολής από ένα οποιουδήποτε κακόβουλο λογισμικό, με την προσαρμογή της ανίχνευσης των κατάλληλων αλφαριθμητικών. Φυσικά, η παραπάνω διαδικασία πραγματοποιείται σε πραγματικό χρόνο και δεν απαιτείται κάποιο *overhead* στην χρήση του προγράμματος *KeePass.*

Επιπροσθέτως, αξίζει να αναφέρουμε ότι η μελέτη που διεξήχθει άγγιξε όλα τα επίπεδα ανάλυσης του εν λόγω κακόβουλου λογισμικού, μελετώντας το τόσο στατικά όσο και δυναμικά. Η χρήση των κανόνων *Yara* στοχεύει στην ταυτοποίηση του λογισμικού βάσει κάποιων βασικών αλφαριθμητικών τα οποία εμφανίζονται στην μνήμη της διεργασίας – θύματος. Ωστόσο, η ανίχνευση αυτή δεν συμπεριλαμβάνει χαρακτηριστικά της συμπεριφοράς του, τα οποία είναι πιο δύσκολο να μεταποιηθούν, αφού προσδιορίζονται από τον σκοπό του, καθώς επίσης συναντώνται και σε μία μεγαλύτερη οικογένεια κακόβουλων λογισμικών. Συνεπώς, η έρευνα για τον έλεγχο της αμόλυντης λειτουργίας του εκάστοτε συστήματος πρέπει να είναι πολυεπίπεδη και να μην στηρίζεται σε αυτοματοποιημένες λύσεις, καθώς αυτές είναι ευκόλως παρακάμψιμες από κάποιων έμπειρο προγραμματιστή.

Εν κατακλείδι, η έρευνα που πραγματοποιήθηκε στα πλαίσια αυτής της πτυχιακής εργασίας πιστεύουμε ότι κατέδειξε αρκετές πτυχές της ολοκληρωμένης και καθολικής μελέτης που

πρέπει να διεξάγεται σε κάθε περίπτωση έναντι κακόβουλων προσπαθειών για υπονόμευση της ομαλής λειτουργικότητας του συστήματος μας. Πιθανές προεκτάσεις αυτής, όπως ειπώθηκε προηγουμένως, μπορεί να είναι η κατάδειξη βαθύτερων σημείων στην προσπάθεια καταπολέμησης των κακόβουλων λογισμικών. Σημαντική προέκταση είναι η αυτοματοποίηση όλων των λειτουργιών που επιτελούνται στην διάρκεια της μελέτης κακόβουλων λογισμικών, για την προσομοίωση των εμπορικών αντιικών συστημάτων τα οποία υποφέρουν από έλλειψη εξειδίκευσης σε κάθε σύστημα και περιβάλλον καθώς επίσης υπόκεινται σε διεθνή και οικονομικά συμφέροντα. Στο πεδίο της έρευνας, τη εκπόνηση αυτής της πτυχιακής προσφέρει την βάση για την δημιουργία αυτοματοποιημένων και αυτοιαματικών εργαλείων με εξαιρετικά υψηλή εξειδικευμένη χρήση σε νέα και πανίσχυρα κακόβουλα λογισμικά. Συνολικά, επίσης, αναδείχθηκε η μελέτη αλφαριθμητικών ως μία καθόλου αμελητέα δύναμη στην καταπολέμηση κακόβουλων λογισμικών, η μελέτη των οποίων **μπορεί** να μας καταδείξει και δυναμικά στοιχεία συμπεριφοράς. Αυτή η γνώση μπορεί να μας καταδείξει μελλοντικούς τρόπους έρευνας και μελέτης ασθενικών συστημάτων. Τα λειτουργικά συστήματα αυτά δε, δεν χρειάζεται να είναι *Windows* ή *Linux* καθώς παρόμοιες τεχνικές ακολουθούνται και σε μελέτη κακόβουλων *Android* εφαρμογών. Τεχνικές όπως *sandboxing* και αποδόμηση κώδικά στα πλαίσια της αντίστροφή μηχανικής μέσω μηχανισμών αποσφαλμάτωσης (*IDA Pro*) καθώς όλες οι τεχνικές παρακολούθησης της δράσης λογισμικών είναι πλέον κοινότοπες και στην ανάλυση κακόβουλων *Android* εφαρμογών. Συνεπώς, ακολουθώντας τις γραμμές που χαράχθηκαν στην παρούσα μελέτη μπορεί να πραγματοποιηθεί μία πλήρης, εκτεταμένη και ενδελεχής έρευνα, στον πλέον διαδεδομένο και διαρκώς αναπτυσσόμενο πεδίο των συσκευών με λειτουργικό σύστημα *Android,* για την καταπολέμηση των κακόβουλων επιθέσεων.

ΠΑΡΑΡΤΗΜΑ Α

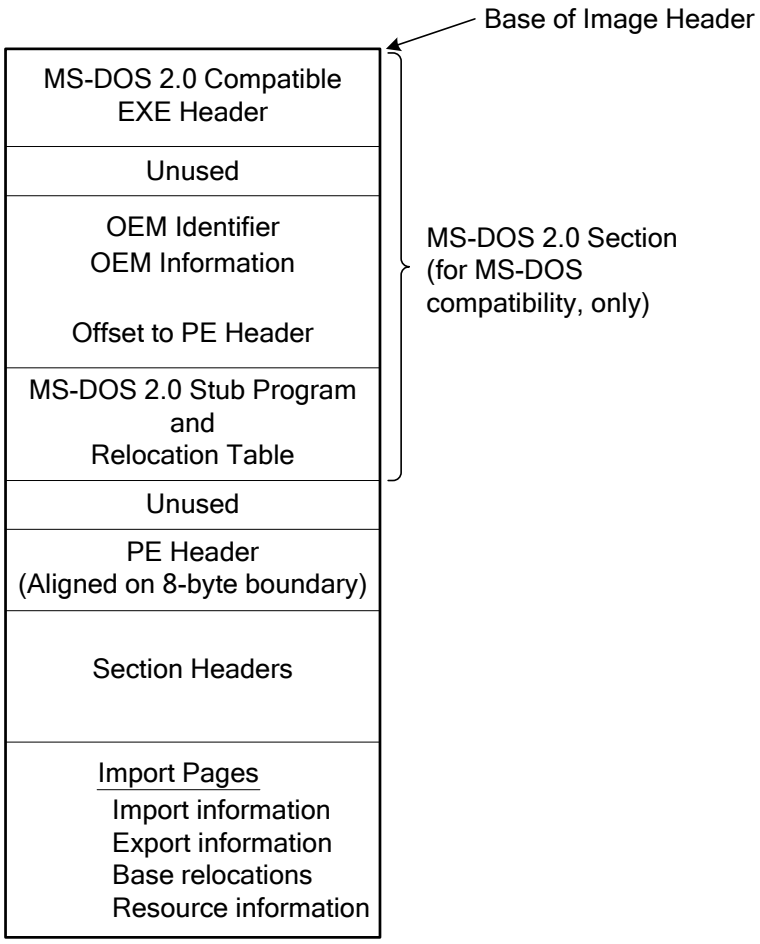
ΛΕΞΙΚΟ

1.PE

Το PE αποτελεί ένα πρότυπο αρχείων που χρησιμοποιείται για εκτελέσιμα αρχεία και *Dinamic Linked Libraries* ή *DLLs* σε 32-bit και 64-bit λειτουργικά συστήματα *Windows.* Το πρότυπο αυτό αποτελεί μία δομή δεδομένων η οποία εμπεριέχει όλες τις πληροφορίες οι οποίες είναι απαραίτητες ώστε να μπορεί *Windows OS Loader* να διαχειριστεί τον εσωκλειόμενο κώδικα. Οι πληροφορίες αυτές είναι

- Αναφορές σύνδεσης των DLL
- Πίνακες εισαγωγής και εξαγωγής του API
- Δεδομένα Διαχείρισης Πόρων
- Τοπικά Δεδομένα Νήματος ή *TLS.*

Ένα αρχείο κατά πρότυπο PE αποτελείται από ένα αριθμό επικεφαλίδων και τμημάτων που υποδεικνύουν στον *Dynamic Linker* πως να χαρτογραφήσει το αρχείο στην μνήμη. Το κάθε τμήμα απαιτεί διαφορετικά δικαιώματα διαχείρισης και είναι δουλειά του *Dynamic Linker* παράλληλα με την χαρτογράφηση του κάθε τμήματος στην μνήμη να αποδώσει τα δικαιώματα αυτά [33]. Η μορφή ενός PE αρχείου φαίνεται στην Εικόνα 57.



Εικόνα 57: Μορφή PE αρχείου [34]

2. Mutex

Ένα *mutex* είναι ένα αντικείμενο αμοιβαίου αποκλεισμού. Στην πράξη, ένα *mutex* είναι ένα αντικείμενο το οποίο προσφέρει την δυνατότητα πολλά νήματα να έχουν πρόσβαση σε ένα πόρο, αλλά όχι ταυτόχρονα [35]. Συνεπώς, το *mutex* είναι ένα μηχανισμός για την ενορχήστρωση της πρόσβασης πολλαπλών νημάτων σε ένα πόρο, όπως η πρόσβαση σε ένα αρχείο. Ένα αντικείμενο *mutex* είναι ένα αντικείμενο συγχρονισμού το οποίο σημαίνεται αν ανήκει ή δεν ανήκει σε ένα νήμα. Μόνο ένα νήμα κάθε φορά μπορεί να έχει πρόσβαση σε ένα *mutex* και το κλειδώνει. Όταν ένα άλλο νήμα θέλει να αποκτήσει πρόσβαση σε έναν κοινό πόρο, τότε ελέγχει αν έχει κυριότητα του *mutex* που σχετίζεται με τον πόρο έχει παρθεί από κάποιο άλλο νήμα. Αν ναι πρέπει να περιμένει για το άλλο νήμα να αποδεσμεύσει τον πόρο ξεκλειδώνοντας το αντίστοιχο *mutex.* Στην περίπτωση που το *mutex* είναι ξεκλείδωτο, το νήμα το κλειδώνει και αποκτά πρόσβαση στον κοινό πόρο. [36]

3.IOC (Indocator of Compromise)

Οι ενδείκτες προσβολής αφορούν σε στοιχεία εγκληματολογικής ερευνάς (*forensics artifacts*) που μπορούν να αναγνωριστούν σε ένα τερματικό υπολογιστή ή δίκτυο και υποδεικνύουν μεγάλη πιθανότητα προσβολής από κακόβουλο λογισμικό. Τα *IOC* μπορεί να είναι:

- Υπογραφές διαφόρων τύπων που αφήνουν τα κακόβουλα λογισμικά
- Διευθύνσεις *IP*
- MD5 αλφαριθμητικό κατακερματισμού
- *URL*
- *Domain Names*

Τέτοιοι ενδείκτες χρησιμοποιούνται στο εργαλείο *Yara* για την δημιουργία κανόνων σύμφωνα με τους οποίους ανιχνεύεται η ύπαρξη κακόβουλης οντότητας.

Ωστόσο η αυξημένη πολυπλοκότητα των κακόβουλων λογισμικών επέβαλλε ανάγκη για αλλαγή στο προσδιορισμό και στην κοινή αναφορά αυτών. Πιο συγκεκριμένα, η ανάγκη για την αλλαγή του προσδιορισμού των κακόβουλων λογισμικών από την παρατήρηση της δομής τους σε παρατήρηση των αποτελεσμάτων της δράσης τους, καλύφθηκε με την δημιουργία των εξελιγμένων πρωτοκόλλων *IOC.* Τέτοιο πρωτόκολλο είναι το *OpenIOC,* το οποίο είναι ένα βασισμένο σε *XML* γλώσσα πρωτόκολλο για την ευέλικτη περιγραφή της συμπεριφορά των κακόβουλων λογισμικών. [37]

4.Obfuscation / Code Packing

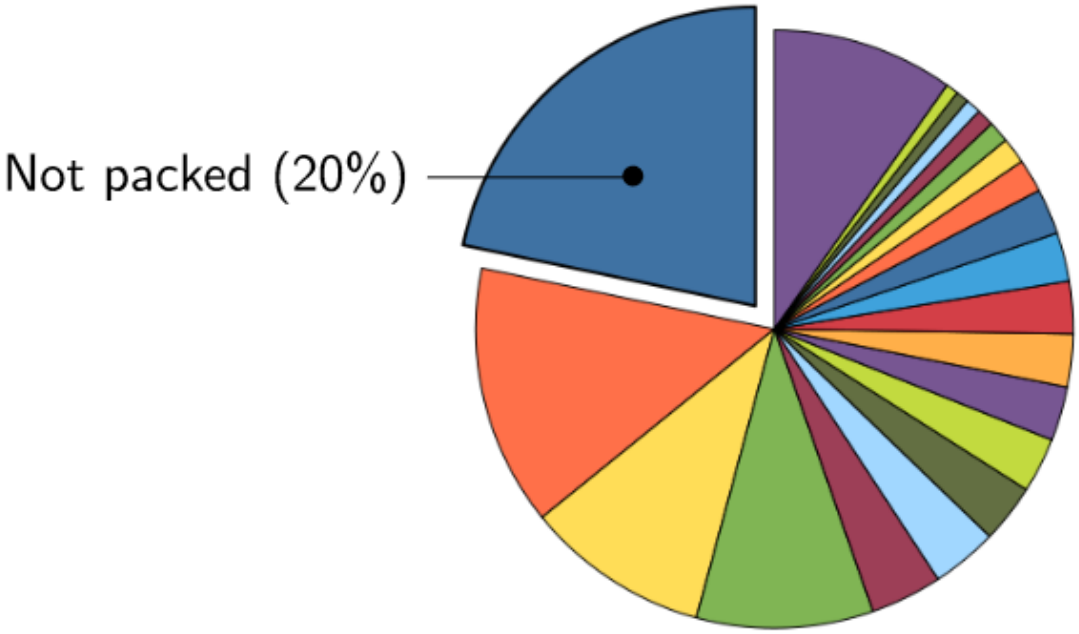
Τα κακόβουλα λογισμικά είναι χρήσιμα για τον δημιουργό τους για όσο καιρό μένουν κρυμμένα και δρουν ανενόχλητα στο σύστημα του ξενιστή. Συνεπώς, είναι ίδιας σημασίας για τον *Hacker,* το λογισμικό του να είναι και αποτελεσματικό και να δρα στα παρασκήνια ανενόχλητο. Η εξέλιξη των εργαλείων προστασίας και ανάλυσης, παράλληλα δίνουν την δυνατότητα στον εκάστοτε αναλυτή να εντοπίσει την δράση ενός κακόβουλου λογισμικού και στην συνέχεια να το υποβάλει στο τραπέζι του ιατροδικαστή και να προβεί σε κάθε είδους τεχνική ανάλυση, μέχρι το σημείο της αντίστροφης μηχανικής για την εύρεση του πηγαίου κώδικα. Αυτό είναι κάτι που οι *Hackers* δεν ανέχονται και επιστρατεύουν συγκεκριμένες τεχνικές για αποκρύψουν την πραγματική φύση του λογισμικού τους. Δύο τεχνικές για το

επιτύχουν αυτό είναι η τεχνική του *Obfuscation* ή Συσκότισης και η τεχνική του *Code Packing* ή Συμπίεσης εκτελέσιμων αρχείων. Οι δύο τεχνικές παρουσιάζουν συνάφεια καθώς κάποιος μπορεί να παρατηρήσει ότι η δεύτερη αποτελεί προέκταση ή υπομορφή της πρώτης.

Η Συσκότιση αφορά σε όλες τις τεχνοτροπίες με τις οποίες επιτυγχάνεται ο δυαδικός κώδικας ενός αρχείου, καθώς και όλα τα λεκτικά δεδομένα που το αποτελούν, να μετατρέπονται σε μορφή μη αναγνωρίσιμη και δύσκολη στην κατανόηση, με σκοπό την αποφυγή της αντίστροφης μηχανικής κώδικα. Η υλοποίηση της μπορεί να αφορά σε κάποια απλή τεχνική, όπως μερικοί χειρισμοί κάποιων bit, αλλά μπορεί να αφορά και σε αρκετά πολύπλοκες και περίτεχνες διαδικασίες, όπως η κρυπτογράφηση (π.χ. πρωτόκολλα *DES, AES* κ.τ.λ.). Η απόκρυψη με οποιονδήποτε τρόπο μέρους ενός λογισμικού, όπως τα αλφαριθμητικά που το αποτελούν, είναι καίριας σημασίας για την αποφυγή της κατανόησης της συμπεριφοράς του λογισμικού [38]. Μερικές διάσημες τεχνικές Συσκότισης είναι οι παρακάτω:

- Η διαδικασία *XOR.* Με την τεχνική αυτή μέρος η όλα τα *bytes* του κακόβουλου αρχείου υπόκεινται σε με διαδικασία *XOR* με κάποια σταθερή ή και μεταβαλλόμενη τιμή.
- Κωδικοποίηση *Base64.* Η κωδικοποίηση αυτή χρησιμοποιεί τα γράμματα της Αλφαβήτου και τον χαρακτήρα ίσον (=) ως χαρακτήρα διακένων για την μετατροπή του αρχικού κειμένου σε κείμενο κρυφό ως προς το νόημα.
- *ROT13*. Η τεχνική αυτή συνίσταται στη μεταλλαγή όλων των γραμμάτων της Αλφαβήτου με διαφορά δεκατρείς θέσεις από το αρχικό. Εξέλιξη αυτής της τεχνικής είναι το *ROT15,* στην οποία η μεταλλαγή γίνεται με μεταβλητή απόσταση.

Σε μεγαλύτερη κλίμακα, η Συσκότιση - Συμπίεση ολόκληρου του αρχείου κώδικα και ο συνδυασμός του με ασυμπίεστο κώδικα σε ένα εκτελέσιμο αρχείο, αποτελεί την τεχνική του συμπίεσης εκτελέσιμων αρχείων ή *Code Packing* τεχνική. Όταν δοθεί η εντολή για το συμπιεσμένο εκτελέσιμο αρχείο να εκτελεστεί, ο κώδικας αποσυμπίεσης επαναδημιουργεί τον αρχικό κώδικα πριν αυτός εκτελεστεί. Τις περισσότερες φορές η διαδικασία αυτή γίνεται αδιαφανώς με αποτέλεσμα το όλο πακέτο να εκτελείται σαν να μην είχε ποτέ υποστεί συμπίεση κώδικα. Η τεχνική αυτή έχει καταστεί εξαιρετικά δημοφιλής και χρησιμοποιείται από τα περισσότερα σύγχρονα κακόβουλα λογισμικά (Εικόνα 58). Σύμφωνα με έρευνα [39] το 80% των σύγχρονων κακόβουλων λογισμικών έχουν συμπιεστεί με την χρήση κάποιου *Packer.*



Εικόνα 58: Ποσοστό συμπίεσης Κακόβουλων Λογισμικών

ΠΑΡΑΡΤΗΜΑ Β

ΜΕΛΕΤΗ ΛΟΓΙΣΜΙΚΩΝ *ZEUS* ΚΑΙ *ALINA* ΣΕ *CUCKOO SANDBOX*

Το *Zeus* είναι ένας ιός τύπου δούρειος ίππος ο οποίος προσβάλει συστήματα τα οποία έχουν λειτουργικό σύστημα *Windows.* Το εν λόγω λογισμικό αποτελεί ένα από τα πλέον διαδεδομένα και επικίνδυνα κακόβουλα λογισμικά που κυκλοφορούν. Η χρήση του αφορά στην διεξαγωγή ενός μεγάλου εύρους κακόβουλων ενεργειών, οι οποίες κυρίως αφορούν στην υποκλοπή τραπεζικών στοιχείων του χρήστη. Αυτό επιτυγχάνεται με πολλές τεχνικές με διαφορετικό προσανατολισμό από αυτές που επιστρατεύει το *KeeFarce*. Ωστόσο, εμείς το μελετάμε καθώς μία πληθώρα παραλλαγών του εμπίπτουν στην κατηγορία των *Ram Scrappers,* κατηγορία που στοχεύει στην υποκλοπή της στοιχείων της μνήμης *RAM,* όπως πράττει και το *KeeFarce.* [40], [41]

Το *Alina* είναι ένα κακόβουλο λογισμικό το οποίο έχει σχεδιαστεί για να υποκλέπτει δεδομένα πιστωτικών και χρεωστικών καρτών τα οποία αποθηκεύονται για μικρό χρονικό διάστημα στην *RAM* τερματικών συσκευών *POS* με λειτουργικό *Windows.* Το λογισμικό επεμβαίνει στην μνήμη *RAM* και υποκλέπτει τα στοιχεία, ενώ παρέχει την δυνατότητα λειτουργιών απομακρυσμένου ελέγχου για αύξηση της λειτουργικότητας και της ευελιξίας του. Το λογισμικό *Alina* παρουσιάστηκε το 2012 και από τότε έχει αποτελέσει την βάση για πληθώρα κακόβουλων λογισμικών με παρόμοιες λειτουργίες. Βασική κοινή ομοιότητα με το *KeeFarce* είναι η επέμβαση στην *RAM* του συστήματος για την διενέργεια λειτουργιών υποκλοπής. [42], [43]

1. *Zeus*

Η ανάλυση του εν λόγω κακόβουλου λογισμικού με την χρήση του *cuckoo sandbox* αποτέλεσε μια ασφαλής διεργασία, η οποία κατέληξε στην εξαγωγή σημαντικών συμπερασμάτων που μπορούν να ανατροφοδοτήσουν περαιτέρω έρευνα. Τα αποτελέσματα θα παρουσιαστούν και θα περιγραφούν χωρισμένα στις παρακάτω κατηγορίες:

- ***Signatures.*** Στην κατηγορία αυτή συγκαταλέγονται οι ταυτοποιήσεις που έκανε το *Cuckoo* αναφορικά με συμπεριφορές και λειτουργίες του *Zeus*, οι οποίες συμπίπτουν με προκαθορισμένες κακόβουλες υπογραφές. Πρακτικά αυτό σημαίνει ότι ανιχνεύτηκαν λειτουργίες οι οποίες χρησιμοποιούνται και από άλλα κακόβουλα λογισμικά και προσδιορίζουν την κακόβουλη λειτουργικότητα του υπό εξέταση αρχείου.


```
"signatures": [
    {
        "families": [],
        "description": "Collects information to fingerprint the system (MachineGuid, DigitalProductId, SystemBiosDate)",
        "severity": 1,
        "marks": [
            {
                "category": "registry",
                "ioc": "HKEY_LOCAL_MACHINE\\SOFTWARE\\Wow6432Node\\Microsoft\\Windows NT\\CurrentVersion\\DigitalProductId",
                "type": "ioc"
            },
            {
                "category": "registry",
                "ioc": "HKEY_LOCAL_MACHINE\\SOFTWARE\\Wow6432Node\\Microsoft\\Windows NT\\CurrentVersion\\InstallDate",
                "type": "ioc"
            }
        ],
        "references": [],
        "name": "recon_fingerprint"
    },
{
    "families": [],
    "description": "Allocates read-write-execute memory (usually to unpack itself)",
    "severity": 2,
    "marks": [
        {
            "call": {
                "category": "process",
                "status": 1,
                "stacktrace": [],
                "api": "NtProtectVirtualMemory",
                "return_value": 0,
                "arguments": {
                    "length": 8192,
                    "protection": 64,
                    "process_handle": "0xffffffff",
                    "base_address": "0x00407000"
                },
                "time": 1460609129.125,
                "tid": 2608,
                "flags": {
                    "protection": "PAGE_EXECUTE_READWRITE"
                }
            },
            "pid": 2240,
            "type": "call",
            "cid": 98
        }
    ],
```


Εικόνα 59: Οικογένειες και υπογραφές Κακόβουλων Λογισμικών που ταιριάζουν στο δείγμα (1)

```
{
    "families": [],
    "description": "Creates executable files on the filesystem",
    "severity": 2,
    "marks": [
        {
            "category": "file",
            "ioc": "C:\\Users\\bill\\AppData\\Roaming\\Daes\\nami.exe",
            "type": "ioc"
        }
    ],
    "references": [],
    "name": "creates_exe"
}
```


Εικόνα 60: Οικογένειες και υπογραφές Κακόβουλων Λογισμικών που ταιριάζουν στο δείγμα (2)

Στις παραπάνω εικόνες φαίνεται ότι η συμπεριφορά του *Zeus* ταυτοποιήθηκε με τρείς υπογραφές, ενώ κάθε μία έχει αντιστοιχιστεί και με ένα βαθμό επικινδυνότητας. Αρχικά φαίνεται πως το κακόβουλο λογισμικό συλλέγει πληροφορίες για την χαρτογράφηση του συστήματος, όπως *MachineGuid, DigitalProductId, SystemBiosDate.* Στην συνέχεια φαίνεται ότι το *Zeus* κατανέμει ελεύθερη μνήμη, πράξη η οποία συνήθως γίνεται με σκοπό να ξεπακεταριστεί ένα κακόβουλο λογισμικό ή να πραγματοποιήσει *dll injection.* Η τρίτη υπογραφή αφορά στην συμπεριφορά του λογισμικού που ανιχνεύτηκε να δημιουργεί εκτελέσιμα αρχείο στο σύστημα. Ένα τέτοιο αρχείο αποτελεί έναν ενδείκτη προσβολής (*IOC*), και δημιουργήθηκε στην διαδρομή δίσκου *C:/Users/bill/AppData/Roaming/Daes/* με όνομα *name.exe.*

- ***Target.*** Στην κατηγορία αυτή υπάρχουν κάποιες πληροφορίες για το υπό εξέταση αρχείο, όπως το *md5* του αρχείου που είναι 4361aca79973bfe468bb244a7ec23d4d.
- ***Behaviour.*** Σε αυτήν την κατηγορία συγκεντρώνονται όλες οι πληροφορίες για την συμπεριφορά του λογισμικού *Zeus.* Η μελέτη των αποτελεσμάτων δίνει στον αναλυτή μία αρκετά πλήρη εικόνα των κινήσεων του κακόβουλου λογισμικού. Τα αποτελέσματα χωρίζονται στις εξής κατηγορίες:
  - ***Generic.*** Η βασική διεργασία που εντοπίστηκε να εκτελείται σχετικά με το *Zeus* είναι το ίδιο το δείγμα του κακόβουλου λογισμικού με όνομα 40d7fc077a716565513b8095b45702aa9b05db276db1af626ca544baaf9fc40d και αριθμό 2240.
  - ***Summary.*** Στην περίληψη φαίνονται συγκεντρωτικά οι ενέργειες που πραγματοποίησε το κακόβουλο λογισμικό στο χρόνο της ανάλυσης. Παρακάτω

φαίνονται τα αποτελέσματα:


```
"summary": {
            "regkey_written": [
                "HKEY_CURRENT_USER\\(Default)"
            ],
            "dll_loaded": [
                "ntmarta.dll",
                "CRYPTSP.dll",
                "NTDLL",
                "shell32.dll",
                "rpcrt4.dll",
                "ole32.dll",
                "shlwapi.dll",
                "ws2_32.dll"
            ],
            "file_opened": [
                "C:\\Users\\bill\\AppData\\Roaming\\Daes",
                "C:\\",
                "C:\\Users\\bill\\AppData\\Roaming\\",
                "C:\\Users\\bill\\AppData\\Roaming\\Nikeyv\\yqiv.cyu",

"C:\\Users\\bill\\AppData\\Local\\Temp\\40d7fc077a716565513b8095b45702aa9b05db276db1af626ca544baaf9fc40d",
                "C:\\Users\\bill\\AppData\\Roaming",
                "C:\\Users\\bill\\AppData\\Roaming\\Daes\\nami.exe",
                "C:\\Users\\bill\\AppData\\Roaming\\Nikeyv"
            ],
            "regkey_opened": [
                "HKEY_LOCAL_MACHINE\\System\\CurrentControlSet\\Control\\LSA\\AccessProviders",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion",
                "HKEY_CURRENT_USER\\SOFTWARE\\Microsoft",
                "HKEY_LOCAL_MACHINE\\Software\\Policies\\Microsoft\\Windows NT\\Rpc",
                "HKEY_LOCAL_MACHINE\\System\\CurrentControlSet\\Services\\LDAP",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\OLE",
                "HKEY_LOCAL_MACHINE\\Software\\Microsoft\\OLE\\Tracing",
                "HKEY_LOCAL_MACHINE\\Software\\Microsoft\\Rpc",
                "HKEY_CURRENT_USER\\Software\\Microsoft\\Aqiz",
                "HKEY_CURRENT_USER\\(Default)"
            ],
            "file_written": [
                "C:\\Users\\bill\\AppData\\Roaming\\Daes\\nami.exe"
            ],
            "file_exists": [
                "C:\\Users\\bill\\AppData\\Roaming\\Daes",
                "C:\\",
                "C:\\Users\\bill\\AppData\\Roaming\\",
                "C:\\Users\\bill\\AppData\\Roaming\\Nikeyv\\yqiv.cyu",

"C:\\Users\\bill\\AppData\\Local\\Temp\\40d7fc077a716565513b8095b45702aa9b05db276db1af626ca544baaf9fc40d",
                "C:\\Users\\bill\\AppData\\Roaming",
                "C:\\Users\\bill\\AppData\\Roaming\\Daes\\nami.exe",
                "C:\\Users\\bill\\AppData\\Roaming\\Nikeyv"
            ],
            "mutex": [
                "Global\\{C78F53E9-0609-F4BD-5DB0-3A0B27792A31}",
                "Global\\{4256EB52-BEB2-7164-5DB0-3A0B27792A31}"
            ],
            "command_line": [
                "\"C:\\Users\\bill\\AppData\\Roaming\\Daes\\nami.exe\""
            ],
            "file_read": [

"C:\\Users\\bill\\AppData\\Local\\Temp\\40d7fc077a716565513b8095b45702aa9b05db276db1af626ca544baaf9fc40d"
            ],
            "regkey_read": [
                "HKEY_LOCAL_MACHINE\\SYSTEM\\ControlSet001\\services\\LDAP\\UseOldHostResolutionOrder",
                "HKEY_CURRENT_USER\\(Default)",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\MICROSOFT\\Rpc\\MaxRpcSize",
                "HKEY_LOCAL_MACHINE\\SYSTEM\\Setup\\SystemSetupInProgress",
                "HKEY_LOCAL_MACHINE\\SYSTEM\\ControlSet001\\Control\\Lsa\\AccessProviders\\MartaExtension",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\MICROSOFT\\OLE\\PageAllocatorUseSystemHeap",
                "HKEY_LOCAL_MACHINE\\SYSTEM\\ControlSet001\\services\\LDAP\\LdapClientIntegrity",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\Wow6432Node\\Microsoft\\Windows
NT\\CurrentVersion\\InstallDate",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\MICROSOFT\\SQMClient\\Windows\\CEIPEnable",
                "HKEY_LOCAL_MACHINE\\SYSTEM\\Setup\\OOBEInProgress",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\Wow6432Node\\Microsoft\\Windows
NT\\CurrentVersion\\DigitalProductId",
                "HKEY_LOCAL_MACHINE\\SYSTEM\\ControlSet001\\services\\LDAP\\UseHostnameAsAlias",
                "HKEY_LOCAL_MACHINE\\SOFTWARE\\MICROSOFT\\OLE\\PageAllocatorSystemHeapIsPrivate",

"HKEY_LOCAL_MACHINE\\SYSTEM\\ControlSet001\\Control\\ComputerName\\ActiveComputerName\\ComputerName"
            ],
            "directory_created": [
                "C:\\Users\\bill\\AppData\\Roaming\\Nikeyv",
                "C:\\Users\\bill\\AppData\\Roaming\\Daes"
            ]
        }
    }
```

Αρχικά παρατηρούμε ότι δημιουργήθηκε ένα κλειδί αδείας συστήματος με τον προεπιλεγμένο χρήστη. Στη συνέχεια βλέπουμε τις βιβλιοθήκες που φορτώθηκαν από την διεργασία. Κατόπιν έρευνας στο διαδίκτυο βρήκαμε ότι οι βιβλιοθήκες αυτές δε αποτελούν σημαντικές βιβλιοθήκες συστήματος. Όντας προαιρετικές και κατόπιν περαιτέρω έρευνας στο διαδίκτυο εξήχθει το συμπέρασμα ότι κάποιες από αυτές, όπως οι *CRYPTSP.dll, NTDLL, ws2_32.dll*, πιθανότατα αποτελούν αλλοιωμένες βιβλιοθήκες με κακόβουλο σκοπό, ιδιαίτερα στην περίπτωση που δεν έχουν φορτωθεί από τις νόμιμες διαδρομές δίσκου. Συγκεκριμένα για την βιβλιοθήκη *NTDLL,* η οποία παρέχει συναρτήσεις πυρήνα συστήματος*,* παρατηρούμε αρχικά το ύποπτο γεγονός της αλλοίωσης του ονόματος της, το οποίο κανονικά είναι *ntdll.dll.* Επίσης, η συγκεκριμένη βιβλιοθήκη έχει επισημανθεί ως κίνδυνος ασφαλείας από την *Microsoft* τον Μάρτιο του 2003. Συνεπώς η χρήση της από πρόγραμμα εγείρει σοβαρά ερωτήματα ως προς την φύση και τον σκοπό του.

Επίσης παρατηρούμε πως το πρόγραμμα διάβασε πληθώρα αρχείων πληροφοριών συστήματος, όπως επίσης δημιούργησε ένα αρχείο *name.exe* το οποίο το εκτέλεσε μέσω εντολής σε τερματικό παράθυρο. Τέλος παρατηρούμε την δημιουργία δύο καταλόγων, των *Nikeyv* και *Daes*. Όσον αφορά την πληροφορία σχετικά με την δημιουργία εκτελέσιμου αρχείου, αυτή πρέπει να αξιοποιηθεί κατάλληλα για περαιτέρω ανάλυση. Συγκεκριμένα θα πρέπει μέσω του φακέλου *files* του αποτελέσματος της ανάλυσης του *Cuckoo,* να πάρουμε το αρχείο name.exe και με την χρήση ενός αποσφαλματωτή να το εξετάσουμε, εφαρμόζοντας παράλληλα τεχνικές αντίστροφης μηχανικής. Η ανάλυση αυτή ξεφεύγει από τον σκοπό του παραρτήματος, ωστόσο υποδεικνύουμε τον τρόπο με τον οποίο μπορούμε να κατανοήσουμε καλύτερα την συνολική λειτουργικότητα του λογισμικού *Zeus.*

- ***Apistats.*** Στο πεδίο αυτό βλέπουμε ποιές διεπαφές έχουν χρησιμοποιηθεί και κατά πόσες φορές. Μία έρευνα σχετικά με τα ονόματα των διεπαφών μπορεί να μας διαφωτίσει σχετικά με τις δυνατότητες αυτών και την πιθανότητα κάποιες από αυτές να χρησιμοποιούνται σε κακόβουλες εφαρμογές.
- ***Procecess.*** Μέσω της υποκατηγορίας *calls* αυτού του πεδίου μπορούμε να δούμε όλες τις κλήσεις του προγράμματος που εμπίπτουν σε κάθε ένα από τις διεπαφές που αναφέραμε και αξιοποιώντας πληροφορίες που διατίθενται, όπως επιστρεφόμενη τιμή για κάθε κλήση και μεταβλητές κλήσης, να εξάγουμε βαθύτερα συμπεράσματα.
- ***Debug.*** Οι πληροφορίες αποσφαλμάτωσης της ανάλυσης έχουν το ρόλο της

παρακολούθησης των τεκταινομένων της διαδικασίας.

- ***Strings.*** Μέσα από την μελέτη των αλφαριθμητικών πιθανότατα μπορούμε να εξάγουμε κάποιους κανόνες *Yara* για την ανίχνευση του λογισμικού *Zeus.*
- ***Network.*** Εξετάζοντας μία μία τις κλήσεις σε επίπεδο δικτύου και *IP* διευθυνσιολογίας μπορούμε να εξετάσουμε αν υπάρχει κάποια κίνηση προς μία *IP* διεύθυνση που δεν είναι λογική για την λειτουργία του συστήματος μας. Στην συγκεκριμένη περίπτωση δεν διακρίναμε κάτι ύποπτο, καθώς όλες οι *IP* αφορούσαν σε *multicast* τεχνικές καταγραφής δρόμων από διάφορα πρωτόκολλα *IP Routing.*

Συνολικά το συμπέρασμα ήταν πως οι περισσότερες πληροφορίες αποκτήθηκαν από το πεδίο *behavior* και οι οποίες μας δίνουν ένα πολύ καλό έναυσμα για περαιτέρω διερεύνηση. Όσον αφορά μία πρώτη εκτίμηση το *Zeus* αποτελεί όντως κακόβουλο λογισμικό.

2. Alina

Το εν λόγω κακόβουλο λογισμικό απαιτεί τον προσεκτικό χειρισμό του αναλυτή, καθώς δεν μπορούμε να γνωρίζουμε κάθε φορά ποιανής παραλλαγής δείγμα έχουμε στην κατοχή μας. Αυτό πρακτικά σημαίνει ότι ενώ μπορεί το αρχικό λογισμικό να μην μπορεί να επιφέρει προβλήματα σε ένα λειτουργικό σύστημα που δεν έχει σχεδιαστεί για συσκευή *POS,* το δείγμα που υποβάλλουμε σε εξέταση είναι πιθανόν να μπορεί να προκαλέσει εκτενή απώλεια πληροφοριών στο σύστημά ανάλυσης. Για το λόγο αυτό διενεργήσαμε την αρχική ανάλυση μέσω του *Cuckoo SandBox.* Τα αποτελέσματα της ανάλυσης καθώς και η μορφή παρουσιάζουν κάποιες ομοιότητες με την μελέτη του λογισμικού *Zeus.* Συνεπώς επιλέξαμε να παρουσιάσουμε τα σημεία στα οποία αναγνωρίζεται η κακόβουλη συμπεριφορά του δείγματος. Παρακάτω, λοιπόν, παρουσιάζονται τα αποτελέσματα της ανάλυσης χωρισμένα σε τομείς:

- ***Signatures.*** Στον τομέα των υπογραφών ανιχνεύτηκαν και ταυτοποιήθηκαν ως ύποπτες τρείς συμπεριφορές του λογισμικού *Alina* (Εικόνα 61).

```
"signatures": [
    {
        "families": [],
        "description": "Creates executable files on the filesystem",
        "severity": 2,
        "marks": [
            {
                "category": "file",
                "ioc": "C:\\Users\\bill\\AppData\\Roaming\\jucheck.exe",
                "type": "ioc"
            }
        ],
        "references": [],
        "name": "creates_exe"
    },
    {
        "families": [],
        "description": "File has been identified by at least one AntiVirus on VirusTotal as malicious",
        "severity": 2,
        "marks": [
            {
                "positives": 1,
                "type": "generic"
            }
        ],
        "references": [],
        "name": "antivirus_virustotal"
    },
    {
        "families": [],
        "description": "Installs itself for autorun at Windows startup",
        "severity": 3,
        "marks": [
            {
                "category": "registry",
                "ioc": "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\jucheck",
                "type": "ioc"
            }
        ],
        "references": [],
        "name": "persistence_autorun"
    }
],
```


Εικόνα 61: Οικογένειες και υπογραφές Κακόβουλων Λογισμικών που ταιριάζουν στο δείγμα

Αρχικά φαίνεται πώς το λογισμικό δημιουργεί εκτελέσιμα αρχεία. Συγκεκριμένα δημιουργεί ένα αρχείο το οποίο ονομάζεται *jucheck.exe*, γεγονός με επικινδυνότητα επιπέδου δύο να προκαλέσει κακόβουλη ενέργεια στο σύστημα. Στην συνέχεια παρατηρούμε μία υπογραφή επιπέδου επικινδυνότητας δύο, η οποία όμως αποτελεί ισχυρότατη ένδειξη ότι το δείγμα μας είναι κακόβουλο λογισμικό. Η υπογραφή αυτή αναφέρει ότι το δείγμα που εξετάζουμε έχει αναγνωριστεί από τουλάχιστον ένα αντϊικό ή την ιστοσελίδα *VirusTotal* ως κακόβουλο. Υπό συνθήκες αυτό μας αρκεί για να ασχοληθούμε περαιτέρω με το δείγμα και αποδεικνύει την χρησιμότητα της αυτοματοποιημένης ανάλυσης. Η τρίτη υπογραφή αφορά στην συμπεριφορά επικινδυνότητας τρία του δείγματος να εγκαθιστά μόνο του τον εαυτό του στην λίστα με τα προγράμματα τα οποία εκτελούνται αυτόματα στην εκκίνηση. Το συγκεκριμένο αποτελεί μία ένδειξη η οποία πρέπει να ερευνηθεί εις βάθος για την παρατήρηση των αλλαγών που επιφέρει το λογισμικό κατά την εκκίνηση του συστήματος, συμπεριφορά

την οποία δεν μπορεί να ελέγξει το *Cuckoo Sandbox.*

- ***VirusTotal***. Στον τομέα αυτό φαίνεται το αποτέλεσμα την ανάλυσης και ταυτοποίησης που έχουν κάνει διάφορα αντιικά προγράμματα, παρουσιασμένα συγκεντρωτικά μέσω της ιστοσελίδας *VirusTotal.* Στην Εικόνα 62 φαίνεται πως το σύστημα προστασίας *NANO-Antivirus* έχει ταυτοποιήσει το δείγμα ως κακόβουλο Δούρειο ίππο που προσβάλει συστήματα *POS* που τρέχουν λειτουργικό *Windows.*


```
"NANO-Antivirus": {
    "detected": true,
    "version": "0.30.26.3725",
    "result": "Trojan.Win32.POSCardStealer.bfeqtu",
    "normalized": [
        "POSCardStealer",
        "bfeqtu"
    ],
    "update": "20151005"
```


Εικόνα 62: Εντοπισμός δείγματος ως Κακόβουλο από το αντïικό NANO-Antivirus

- ***Dropped.*** Στο πεδίο αυτό βλέπουμε αναλυτικά στοιχεία για το εκτελέσιμο αρχείο που παράγεται από το δείγμα. Κατόπιν παρατήρησης των αποτελεσμάτων συμπεραίνουμε πως αυτό είναι και ο βασικός πυρήνας στον οποίο στηρίζεται η κακόβουλη λειτουργικότητα του *Alina.* Αρχικά βλέπουμε το *sha1*, το όνομα και τον τύπο του αρχείου, στοιχεία που χρησιμοποιούνται για έρευνα και ταυτοποίηση (Εικόνα 63).


```
"sha1": "5563e4c2987eda056b3f74716c00d3014b9306bc",
"name": "036e4f452041f9d5_jucheck.exe",
"type": "PE32 executable (GUI) Intel 80386, for MS Windows, UPX compressed"
```


Εικόνα 63: Βασικές πληροφορίες δείγματος

Στην συνέχεια βλέπουμε πώς όλες οι μεγάλες αντιικές Βάσεις Δεδομένων των προγραμμάτων προστασίας περιέχουν το εκτελέσιμο ως απειλή υποκλοπής πληροφοριών σε *POS* συστήματα. Στην Εικόνα 64 φαίνεται και η ονοματολογία του κακόβουλου λογισμικού και πλέον δεν έχουμε αμφιβολία για τον ποιόν του δείγματος.


```
"ESET-NOD32": {
    "detected": true,
    "version": "13236",
    "result": "Win32/Spy.POSCardStealer.D",
    "update": "20160325"
},
```


Εικόνα 64: Ταυτοποίηση δείγματος ως κακόβουλη οντότητα

- ***Behaviour.*** Στον τομέα της συμπεριφοράς παρατηρούμε δύο σημαντικούς υποτομείς από τους οποίους αντλούμε το μεγαλύτερο πλήθος των πληροφοριών ανάλυσης.
  - ***Generic.*** Στην υποκατηγορία αυτή φαίνονται στοιχεία συμπεριφοράς του αρχικού εκτελέσιμου δείγματος (Εικόνα 65).


```
"process_name": "3_4.exe",
"ppid": 992,
"pid": 2472,
"first_seen": 1460609099.625,
"summary": {
    "regkey_written": [
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\jucheck"
    ],
    "dll_loaded": [
        "urlmon.dll",
        "WININET.dll",
        "SHELL32.dll",
        "KERNEL32.DLL",
        "ADVAPI32.dll",
        "PSAPI.DLL",
        "SHLWAPI.dll"
    ],
    "file_opened": [
        "C:\\Users\\bill\\AppData\\Local\\Temp\\3_4.exe",
        "C:\\Users\\bill\\AppData\\Roaming\\jucheck.exe",
        "C:\\"
    ],
    "regkey_opened": [
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run"
    ],
    "file_written": [
        "C:\\Users\\bill\\AppData\\Roaming\\jucheck.exe"
    ],
    "file_exists": [
        "C:\\Users\\bill\\AppData\\Local\\Temp\\3_4.exe",
        "C:\\Users\\bill\\AppData\\Roaming\\jucheck.exe",
        "C:\\"
    ],
    "mutex": [
        "Had3yghhuju98gggd9G6790hfv3.4"
    ],
    "command_line": [
        "alina=C:\\Users\\bill\\AppData\\Local\\Temp\\3_4.exe"
    ],
    "file_read": [
        "C:\\Users\\bill\\AppData\\Local\\Temp\\3_4.exe"
    ],
    "regkey_read": [
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\jucheck",
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\desktop",
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\java",
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\jusched",
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\win-firewall",
        "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\adobeflash",
```


Εικόνα 65: Generic Πληροφορίες

Αρχικά παρατηρούμε ότι το όνομα της διεργασίας είναι *3_4.exe* είναι αρκετά ύποπτο για νόμιμο λογισμικό. Από τις βιβλιοθήκες που φόρτωσε το πρόγραμμα δεν μπορούμε να εξάγουμε κάποιο ασφαλές συμπέρασμα. Ωστόσο όλες μπορούν να χρησιμοποιηθούν και από ένα κακόβουλο πρόγραμμα για επιτέλεση κακόβουλων ενεργειών και αυτό μπορεί να

διαπιστευτεί με περαιτέρω έρευνα. Για παράδειγμα, για την βιβλιοθήκη *urlmon.dll* γνωρίζουμε ότι αφενός πρέπει να ελέγχουμε την διαδρομή στον δίσκο από την οποία φορτώθηκε, καθώς οποιαδήποτε άλλη διαδρομή από την *C:\Program Files\internet explorer platform preview\iepreview.exe.local* δεν είναι νόμιμη, αφετέρου πρέπει να παρακολουθήσουμε την διαδικτυακή κίνηση στην οποία συμμετέχει ο υπολογιστής, καθώς οι αυξομειώσεις της ταχύτητας δικτύου, υπέρμετρη χρήση επεξεργαστικών πόρων, ανακατευθύνσεις σε περίεργες ιστοσελίδες και ενοχλητικά *popup-adds* είναι ενδείξεις μολυσματικής χρήσης της εν λόγω βιβλιοθήκης. Ακόμα, παρατηρούμε ότι έχει εγγραφεί ένα καινούργιο κλειδί αδείας για αυτόματη εκκίνηση του δημιουργημένου εκτελέσιμου αρχείου *jucheck* κατά την εκκίνηση του συστήματος. Επίσης, παρατηρώντας τα κλειδιά αδείας που ανοίχτηκαν από το πρόγραμμα και ψάχνοντας για αλλαγές που μπορεί να έγιναν σε αυτά παρατηρούμε ότι διαβάζεται η τιμή του κλειδιού που αφορά στο τείχος προστασίας των *Windows,* γεγονός όχι φυσιολογικό για μία απλή εφαρμογή.

- ***Processes calls.*** Από εδώ μπορούμε να βρούμε οποιαδήποτε από τις συναρτήσεις που αφορούν στις καταγεγραμμένες κλήσεις διεπαφών και να μελετήσουμε τυχόν ύποπτες συμπεριφορές. Πολλές από τις κλήσεις αφορούν σε διαδικασίες που περιγράφησαν παραπάνω και εμπίπτουν στην δυναμική των τεχνικών που χρησιμοποιούν τα κακόβουλα λογισμικά. Αυτό δεν σημαίνει ότι δεν μπορούν να χρησιμοποιηθούν και από καλόβουλα λογισμικά, αλλά ότι όλες οι πληροφορίες που έχουν συλλεχθεί συνηγορούν σε κακόβουλη ενέργεια.

Στο σημείο αυτό φτάσαμε στο τέλος της παρουσίασης της αυτοματοποιημένης ανάλυσης σχετικά με το δείγμα, για το οποίο και διαπιστώθηκε ότι αποτελεί κακόβουλο λογισμικό της οικογένειας *Alina.*

7. ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

1. Wikipedia, Malware, https://en.wikipedia.org/wiki/Malware#Concealment:_Viruses.2C_trojan_horses.2C_rootkits.2C_backdoors_and_evasion, Προσπελάστηκε 15 Απριλίου 2016

2. Kaspersky Labs, Types of Malwares, Labshttps://usa.kaspersky.com/internet-security-center/threats/malware-classifications#.Vt8jA9-6w8o, Προσπελάστηκε 15 Απριλίου 2016

3. Ekta Gandotra, Divya Bansal, Sanjeev Sofat, 28 March 2014, Malware Analysis and Classification: A Survey

4. CFOC, July 7 2015, Top 10 Malware Attacks of 2015, http://cfoc.org/top-10-malware-attacks-of-2015/

5. Kim Zetter, 30 Sept 2014, How RAM Scrapers Work: The Sneaky Tools Behind the Latest Credit Card Hacks, http://www.wired.com/2014/09/ram-scrapers-how-they-work/ https://blog.kaspersky.com/ram-scrapers-and-other-point-of-sale-malware/3600/

6. Paul Ionescu, April 8, 2015, The 10 Most Common Application Attacks in Action, https://securityintelligence.com/the-10-most-common-application-attacks-in-action/

7. techtarget, phishing, http://searchsecurity.techtarget.com/definition/phishing, Προσπελάστηκε 1 Απριλίου 2016

8. Zyxel, Avoid CSRF Pharming Vulnerability and MOOSE Malware http://www.zyxel.com/support/announcement_csrf_pharming_vulnerability_and_moose_malware.shtml, Προσπελάστηκε 1 Απριλίου 2016

9. Graham Cluley, 26 May 2015, Moose – the router worm with an appetite for social networks http://www.welivesecurity.com/2015/05/26/moose-router-worm/

10. Wikipedia, Code injection, https://en.wikipedia.org/wiki/Code_injection, Προσπελάστηκε 15 Απριλίου 2016

11. OpenSecurity Research, January 8, 2013, Windows DLL Injection Basics, http://blog.opensecurityresearch.com/2013/01/windows-dll-injection-basics.html

12. Amit Malik, DLL Injection and Hooking, http://securityxploded.com/dll-injection-and-hooking.php, Προσπελάστηκε 15 Απριλίου 2016

13. Wikipedia, DLL injection, https://en.wikipedia.org/wiki/DLL_injection, Προσπελάστηκε 15 Απριλίου 2016

14. ViperEye, 2 May 2013, Code Injection Techniques, http://resources.infosecinstitute.com/code-injection-techniques/

15. Lenny Zeltser, 19 February 2015, Mastering 4 Stages of Malware Analysis, https://zeltser.com/mastering-4-stages-of-malware-analysis/

16. Syarif Yusirwan S, Yudi Prayudi and Imam Riadi, May 2015, Implementation of Malware Analysis using Static and Dynamic Analysis Method. International Journal of Computer Applications 117(6):11-15
17. Michael Hale Ligh, 15 April 2016, Volatility, https://github.com/volatilityfoundation/volatility/wiki
18. Victor M. Alvarez, YARA in a nutshell, https://plusvic.github.io/yara/, Προσπελάστηκε 15 Απριλίου 2016
19. Joshua Cannell, 11 October 2013, Using YARA to attribute malware, https://blog.malwarebytes.org/intelligence/2013/10/using-yara-to-attribute-malware/
20. Florian Roth, April 2016, yarGen, https://github.com/Neo23x0/yarGen
21. Wikipedia, Sandbox (computer security), https://en.wikipedia.org/wiki/Sandbox_(computer_security), Προσπελάστηκε 15 Απριλίου 2016
22. Cuckoo Foundation, Revision f3f7cbb1,http://docs.cuckoosandbox.org/en/latest/
23. KeePass, http://keepass.info/, Προσπελάστηκε 15 Απριλίου 2016
24. denandz, 17 Nov 2015, KeeFarce, https://github.com/denandz/KeeFarce
25. Mark Russinovich, 2 February 2016, Process Explorer v16.12, https://technet.microsoft.com/el-gr/sysinternals/ processexplorer
26. MarcuSHE, 19 April 2016, Process Status API, https://msdn.microsoft.com/en-us/library/windows/desktop/ms684884%28v=vs.85%29.aspx
27. x64dbg, http://x64dbg.com/#start, Προσπελάστηκε 15 Απριλίου 2016
28. Michael Hale Ligh Andrew Case Jamie Levy AAron Walters, 2014, The Art of Memory Forensics, 886 pp.
29. Michael Hale Ligh, 14 Oct 2015,Volatility - Windows 8 2012, https://github.com/volatilityfoundation/volatility/wiki/Windows-8-2012,
30. BriMor Labs, 13 August 2014, Analysis of a Windows 8 Memory Dump with Volatility 2.4 ("The New Hotness"), http://www.brimorlabsblog.com/2014/08/analysis-of-windows-8-memory-dump-with.html
31. Microsoft, Developer Network, PDB Files, https://msdn.microsoft.com/en-us/library/yd4f8bd1%28vs.71%29.aspx, Προσπελάστηκε 15 Απριλίου 2016
32. Marc Gravell, 22 March 2009, mscordacwks.dll and mscorwks.dll confusions http://stackoverflow.com/questions/670725/mscordacwks-dll-and-mscorwks-dll-confusions
33. Wikipedia, Portable Executable, https://en.wikipedia.org/wiki/Portable_Executable, Προσπελάστηκε 15 Απριλίου 2016
34. Microsoft, Developer Network, Microsoft PE and COFF Specification, 23 February 2013, https://msdn.microsoft.com/en-us/library/windows/hardware/gg463119.aspx

35. Microsoft, Developer Network**,** Mutex Objects, https://msdn.microsoft.com/en-us/library/windows/desktop/ms684266%28v=vs.85%29.aspx, Προσπελάστηκε 15 Απριλίου 2016
36. webopedia, mutex, http://www.webopedia.com/TERM/M/mutex.html, Προσπελάστηκε 15 Απριλίου 2016
37. Hun-Ya Lock, 21 February 2013, Using IOC (Indicators of Compromise) in Malware Forensics
39. Srdjan Matic, Aristide Fattori, 18 November 2014, Malware Obfuscation Techniques: Packing, http://security.di.unimi.it/sicurezza1314/slides/obfuscation-2014.pdf
40. Brian Donohue, 28 January 2014, RAM Scrapers and Other Point-of-Sale Malware, https://blog.kaspersky.com/ram-scrapers-and-other-point-of-sale-malware/3600/
41. Wikipedia, Zeus (malware), https://en.wikipedia.org/wiki/Zeus_%28malware%29, Προσπελάστηκε 15 Απριλίου 2016
42. Josh Grunzweig, 9 January 2015, Alina Continues to Spreads its Wings http://www.nuix.com/blog/alina-continues-spread-its-wings
43. SR_FI_Team, 21 February 2014, Alina POS Malware, http://community.hpe.com/t5/Security-Research/Alina-POS-Malware/ba-p/6385271#.VxjhVkdUVfA