Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ανίχνευση ενδείξεων παραβίασης λειτουργικών συστημάτων Windows 8 (Tracing security breaches in Windows 8 machines)**
Ονοματεπώνυμο Φοιτητή	**Σταματάκης Ξενοφών**
Πατρώνυμο	**Αλέξανδρος**
Αριθμός Μητρώου	**ΜΠΣΠ/ 14083**
Επιβλέπων	**Χρήστος Δουληγέρης, Καθηγητής**
Συνεπιβλέπων	**Σπυρίδων Παπαγεωργίου**

Ημερομηνία Παράδοσης **Νοέμβριος 2017**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Δουληγέρης Χ. Καθηγητής	Βέργαδος Δ. Αναπληρωτής Καθηγητής	Κοτζανικολάου Π. Επίκουρος Καθηγητής

Περίληψη

Στις μέρες μας, το κακόβουλο λογισμικό είναι μια τεράστια απειλή, τόσο για τους απλούς χρήστες όσο και για μεγάλες επιχειρήσεις και οργανισμούς. Με τον όρο κακόβουλο λογισμικό αναφερόμαστε σε όλα εκείνα τα προγράμματα, τα οποία έχουν σαν σκοπό να παραβιάσουν την ακεραιότητα, την διαθεσιμότητα, και την εμπιστευτικότητα των αρχείων ενός υπολογιστή και να προκαλέσουν οποιουδήποτε είδους βλάβη στο σύστημα. Στο πλαίσιο της παρούσας διπλωματικής εργασίας, εξετάστηκαν μέθοδοι για τον εντοπισμό του κακόβουλου λογισμικού και αναπτύχθηκε σχετική εφαρμογή. Η εφαρμογή σχεδιάστηκε με σκοπό να βοηθήσει τους ερευνητές ψηφιακής σήμανσης, να επικεντρώσουν το ενδιαφέρον τους σε συγκεκριμένα αρχεία μειώνοντας σημαντικά τον όγκο των δεδομένων που πρέπει να αναλύσουν με μη αυτοματοποιημένο τρόπο, σε ένα υπολογιστικό σύστημα. Μέσω της εφαρμογής προσφέρονται επίσης, δυνατότητες ανάλυσης πτητικών δεδομένων με τη χρήση υπαρχόντων εργαλείων. Η πειραματική αξιολόγηση έγινε σε λογισμικό windows 8.1 το οποίο πρώτα είχε μολυνθεί με γνωστά κακόβουλα λογισμικά προκειμένου να διερευνηθεί η αποτελεσματικότητα της εφαρμογής.

Abstract

Nowadays, malicious software, or malware as it is better known, poses a significant threat to both everyday users, as well as companies and organisations. We the term malware we describe every software that is built with the intention to invade the integrity, availability, and confidential nature of a computer's files and cause damage to the system. The following thesis is an exploration of the available malware tracking methods, and the subsequent development of a malware tracking and identifying application. The application is designed with the purpose of assisting digital forensics researchers to focus their malware tracking efforts on particular files, and reduce significantly the amount of computer data they have to analyse in a non-automated way. It also supports capabilities for analysing volatile data, using already existent tools. The experimental assessment took place on a Windows 8.1 software, which was infected by commonly used malwares prior to testing, in order to examine the efficiency of the application.

Περιεχόμενα

1 Εισαγωγή

1.1 Περίληψη

Τα τελευταία χρόνια, με την ανάπτυξη της πληροφορικής και των τηλεπικοινωνιών, και της αυξανόμενης χρήσης του διαδικτύου, η εξάρτηση της κοινωνίας από τους υπολογιστές σημειώνει δραματική άνοδο. Είτε πρόκειται για απλούς χρήστες, είτε για μεγάλους οργανισμούς και επιχειρήσεις, τα δεδομένα που διατηρούν σε υπολογιστικά συστήματα είναι ζωτικής σημασίας.

Μια από τις σπουδαιότερες και σοβαρότερες απειλές στο διαδίκτυο σήμερα είναι το κακόβουλο λογισμικό, το λογισμικό εκείνο δηλαδή που σκοπίμως εκπληρώνει τις επιβλαβείς προθέσεις ενός επιτιθέμενου. Τέτοιου είδους λογισμικά έχουν σκοπό να αποκτήσουν πρόσβαση σε υπολογιστικά συστήματα και πόρους δικτύου, να διαταράξουν τις λειτουργίες τους, να συγκεντρώσουν προσωπικές πληροφορίες χωρίς την έγκριση του χρήστη και γενικότερα να παραβιάσουν την ακεραιότητα των δεδομένων του υπολογιστή.

Είτε πρόκειται για αποκόμιση οικονομικού οφέλους, είτε για δολιοφθορά είτε στα πλαίσια προώθησης πολιτικών ιδεών (hactivism), ο αριθμός των κακόβουλων προγραμμάτων που δημιουργούν οι εγκληματίες του κυβερνοχώρου συνεχίζει να αυξάνεται ραγδαία. Σύμφωνα με στατιστικά του AV-TEST, ανεξάρτητου οργανισμού που έχει σαν σκοπό την αξιολόγηση των αντιικών προγραμμάτων, μόνο τον τελευταίο χρόνο, έχουν εντοπιστεί περίπου ενενήντα έξι εκατομμύρια καινούργια malware, με το συνολικό αριθμό τους να πλησιάζει τα εφτακόσια εκατομμύρια [1].

Η ανάπτυξη αυτή των κακόβουλων προγραμμάτων, οδήγησε στη δημιουργία μηχανισμών οι οποίοι θα προστατεύουν τους χρήστες, ανιχνεύοντας την ύπαρξή τους, σε ένα υπολογιστικό σύστημα. Διαφορετικές τεχνικές έχουν προταθεί για τον εντοπισμό κακόβουλου κώδικα με το διαχωρισμό να εντοπίζεται στον τρόπο που λειτουργούν. Από τις πλέον γνωστές και «παραδοσιακές» τεχνικές, είναι η ανίχνευση με βάση τις υπογραφές, ενώ σημαντική έρευνα γίνεται πάνω στον τομέα των ευριστικών μεθόδων ανίχνευσης.

Ένας εξίσου σημαντικός τομέας με την ανάλυση του κακόβουλου λογισμικού, είναι και αυτός της ψηφιακής σήμανσης. Ο κλάδος της ψηφιακής σήμανσης, έχει σαν σκοπό την διερεύνηση υπολογιστικών συστημάτων και δικτύων που φέρεται να έχουν ανάμιξη σε ψηφιακά εγκλήματα με σκοπό τον εντοπισμό πειστηρίων και την παρουσίασή τους σε νομικά πλαίσια.

Λόγω της ευχρηστίας τους και του αποτελεσματικού γραφικού τους περιβάλλοντος, τα windows λειτουργικά συστήματα είναι αυτά που δέχονται τις περισσότερες επιθέσεις [2]. Στην διπλωματική αυτή εργασία, υλοποιείται ένα εργαλείο που σκοπό έχει να βοηθήσει τους ερευνητές της ψηφιακής σήμανσης, να βελτιστοποιήσουν τη διαδικασία συλλογής των ψηφιακών πειστηρίων, σε windows 8 λειτουργικά συστήματα, καθώς και τη διαδικασία εντοπισμού των κακόβουλων προγραμμάτων.

1.2 Κίνητρο

Η ραγδαία αύξηση του αριθμού των κακόβουλων προγραμμάτων, οι επιπτώσεις που έχουν τόσο σε οικονομικούς τομείς [3] όσο και στο κύρος και την αξιοπιστία του κάθε οργανισμού, αλλά και σε θέματα εθνικής ασφάλειας καθώς και η συνεχής εναλλαγή του τοπίου στο χώρο της ασφάλειας μπορούν από μόνα τους να αποτελέσουν ισχυρό κίνητρο ενασχόλησης με το συγκεκριμένο θέμα.

Είναι προφανές, ότι οι επιπτώσεις μη έγκαιρου εντοπισμού μιας ενδεχόμενης παραβίασης ενός υπολογιστή και του τρόπου που επιτεύχθηκε μπορεί να είναι τεράστιες και να οδηγήσουν μέχρι και σε ολική «παράλυση» των κρίσιμων υποδομών κάθε χώρας [4].

Η κύρια πρόκληση, που συνιστά το κίνητρο για την εργασία αυτή είναι ο εντοπισμός του κακόβουλου λογισμικού ανάμεσα σε ένα τεράστιο όγκο δεδομένων που περιλαμβάνει κάθε υπολογιστικό σύστημα.

1.3 Σκοπός και στόχος εργασίας

Στην παρούσα εργασία θα γίνει λόγος για την ανίχνευση κακόβουλων προγραμμάτων χρησιμοποιώντας διαφορετικές προσεγγίσεις. Η εργασία αυτή, αποτελεί μια προσπάθεια μείωσης του όγκου δεδομένων που βρίσκονται σε ένα υπολογιστικό σύστημα, έτσι ώστε να μπορέσουμε να εστιάσουμε σε συγκεκριμένα αρχεία που παρουσιάζουν μεγαλύτερο βαθμό επικινδυνότητας.

Αρχικά, θα μιλήσουμε για τις διάφορες κατηγορίες βάση των οποίων διαχωρίζεται το κακόβουλο λογισμικό, τις τεχνικές που χρησιμοποιούν τα υπάρχοντα συστήματα για τον έλεγχο των αρχείων και τον εντοπισμό του και τις συνήθεις τεχνικές που χρησιμοποιούν οι επιτιθέμενοι για να ξεπεράσουν τους ελέγχους αυτούς.

Τέλος, θα υλοποιήσουμε ένα εργαλείο σε python που σκοπό έχει να συλλέγει πληροφορίες για τα αρχεία και να τις συνδυάζει προκειμένου να μας οδηγεί στην εξαγωγή συμπερασμάτων, για την προέλευση και το σκοπό του κάθε αρχείου.

1.4 Δομή εργασίας

Κεφάλαιο 2 – Κακόβουλο λογισμικό και τεχνικές ανίχνευσης: Στο κεφάλαιο αυτό, γίνεται μια συνοπτική παρουσίαση των βασικότερων κατηγοριοποιήσεων του κακόβουλου λογισμικού. Αναφέρονται συνοπτικά οι βασικές μέθοδοι ανίχνευσής του και τέλος γίνεται μια ειδική αναφορά στις μεθόδους απόκρυψής του, προκειμένου να μην είναι δυνατός ο εντοπισμός του από αντιικά προγράμματα.

Κεφάλαιο 3 – Υλοποίηση: Στο συγκεκριμένο κεφάλαιο παρουσιάζεται η μεθοδολογία υλοποίησης που ακολουθήθηκε για την ανάπτυξη του εργαλείου. Παρουσιάζονται και επεξηγούνται αναλυτικά όλα τα βήματα του αλγορίθμου καθώς και ο σκοπός που εξυπηρετεί κάθε ξεχωριστό βήμα.

Κεφάλαιο 4 – Εγχειρίδιο χρήστη: Το κεφάλαιο αυτό, αποτελεί το εγχειρίδιο χρήσης του εργαλείου. Περιγράφονται όλες οι εξαρτήσεις του, οι διακόπτες και τα ορίσματα που δέχεται και τέλος τα αποτελέσματα που επιστρέφονται με την εκτέλεση του κάθε διακόπτη.

Κεφάλαιο 5 – Πειραματική αξιολόγηση: Στο κεφάλαιο της πειραματικής αξιολόγησης παρατίθενται τα αποτελέσματα εκτέλεσης του εργαλείου σε ένα πραγματικό περιβάλλον windows 8.1, το οποίο έχει μολυνθεί από κακόβουλο λογισμικό.

Κεφάλαιο 6 – Μελλοντικές προσθήκες: Στο κεφάλαιο αυτό, συζητάμε τις μελλοντικές προσθήκες του εργαλείου, που μπορούν να εμπλουτίσουν τις δυνατότητές του, τόσο από άποψη απόδοσης, όσο και από άποψη εναλλακτικής προσέγγισης του θέματος.

Κεφάλαιο 7 – Συμπεράσματα: Στο τελευταίο κεφάλαιο, παρουσιάζεται μια σύνοψη όλης της εργασίας και συζητάμε τα συμπεράσματα που εξήχθησαν από τη διαδικασία της έρευνας και της υλοποίησης.

2 Κακόβουλο λογισμικό και τεχνικές ανίχνευσης

Το κακόβουλο λογισμικό είναι ένας συγκεκριμένος τύπος λογισμικού, που αρχικό σκοπό έχει τον έλεγχο του μολυσμένου συστήματος και στην συνέχεια να προκαλέσει βλάβες σε αυτό. Από την πρώτη στιγμή που έκαναν την εμφάνισή τους τέτοιου είδους προγράμματα, είχαν προκαλέσει ανησυχία τόσο σε απλούς χρήστες όσο και σε δημόσιους οργανισμούς και επιχειρήσεις καθώς είχαν σαν στόχο την κλοπή, την αλλοίωση ή ακόμα και την ολοκληρωτική διαγραφή των δεδομένων.

Η ραγδαία ανάπτυξη των τηλεπικοινωνιών και των υπολογιστικών συστημάτων, οδήγησε σε ένα είδος «πολέμου» μεταξύ της κοινότητας που ασχολείται με την ασφάλεια και των δημιουργών κακόβουλων προγραμμάτων, με τους μεν να χρησιμοποιούν όλα τα διαθέσιμα «όπλα» προκειμένου να ανταπεξέλθουν στην συνεχιζόμενη εξάπλωση των κακόβουλων προγραμμάτων και τους δε, να βρίσκουν νέους τρόπους να τα παρακάμπτουν.

Στις επόμενες ενότητες θα μιλήσουμε για την κατηγοριοποίηση του κακόβουλου λογισμικού, τις τεχνικές ανίχνευσής του και τέλος θα κάνουμε μια μικρή αναφορά στις τεχνικές «απόκρυψης» του κακόβουλου κώδικα, που σκοπό έχουν να περάσει απαρατήρητο από τα αντιικά προγράμματα.

2.1 Κατηγοριοποίηση κακόβουλου λογισμικού

Όπως αναφέρθηκε, το κακόβουλο λογισμικό είναι οποιασδήποτε μορφής κώδικας ο οποίος έχει σαν σκοπό να παραβιάσει την ασφάλεια του λειτουργικού σε σχέση με την διαθεσιμότητα, την ακεραιότητα και την εμπιστευτικότητα των δεδομένων που περιέχει. Ο ραγδαίος ρυθμός αύξησης των παραβιάσεων ασφάλειας εξαιτίας του κακόβουλου λογισμικού, δημιούργησε την ανάγκη της μελέτης και της κατηγοριοποίησης του, με σκοπό να απλουστεύσει την διαδικασία αναγνώρισης και διαχείρισής του. Το κακόβουλο λογισμικό μπορεί να χωριστεί σε δεκαεφτά διαφορετικούς τύπους [5] μεταξύ των οποίων είναι τα trojan, worm και ιοί, ανάλογα με το σκοπό που εξυπηρετεί. Στην ενότητα αυτή θα παρουσιάσουμε τους πιο γνωστούς τύπους κακόβουλου λογισμικού.

1. Ιός: είναι το λογισμικό εκείνο που μετακινείται από ένα υπολογιστή σε άλλο με την εισαγωγή του κώδικά του σε κάποιο άλλο πρόγραμμα. Έχει την ικανότητα να αναπαράγει τον εαυτό του, με σκοπό να μολύνει το σύστημα και να προκαλέσει αλλοιώσεις στα δεδομένα του και εκκινείτε με την εκτέλεση του προγράμματος με το οποίο έχει συνδεθεί.
2. Worm: είναι ένα ακόμα πρόγραμμα που μπορεί να δημιουργεί αντίγραφα του εαυτού του αλλά διαφέρει από τους ιούς κυρίως στο τρόπο μετάδοσης, καθώς μεταδίδεται μέσω δικτύου. Τέλος τα worms είναι αυτόνομα αρχεία δεν χρειάζονται δηλαδή κάποιο host πρόγραμμα και για την μετάδοση τους χρησιμοποιούν είτε κάποια αδυναμία του συστήματος είτε ξεγελούν το χρήστη για να τα εκτελέσει.

3. Spyware: είναι το λογισμικό, το οποίο εγκαθιστάτε στον υπολογιστή εν αγνοία του χρήστη, συνήθως με την εγκατάσταση κάποιου άλλου δωρεάν λογισμικού, με σκοπό να παρακολουθεί και να συλλέγει προσωπικές πληροφορίες και αργότερα τις αποστέλλει στον επιτιθέμενο.

4. Adware: σε αυτή την κατηγορία, εμπίπτουν προγράμματα που σκοπό έχουν να παρατηρούν την δραστηριότητα του χρήστη στο διαδίκτυο και αργότερα να εγκαθιστούν, να προβάλλουν και να αναπαράγουν διαφημίσεις.

5. Trojan: τα Trojan εισχωρούν σε ένα υπολογιστή προσποιούμενα ότι είναι κάποιο νόμιμο πρόγραμμα, με σκοπό να αποκτήσουν προσωπικά δεδομένα όπως κωδικούς χρήστη, ή να καταστρέψουν τα δεδομένα του υπολογιστή. Μια ακόμα λειτουργία τους είναι να δημιουργούν «back doors» για να επιτρέπουν σε κακόβουλους χρήστες να αποκτήσουν πρόσβαση στο σύστημα.

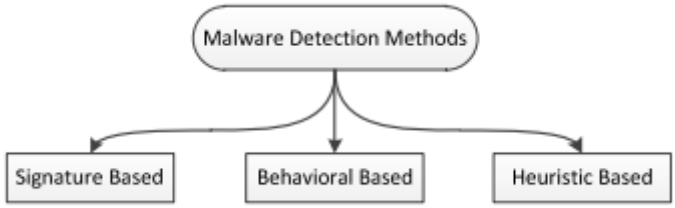
6. Botnet: είναι ένα αυτόνομο κακόβουλο λογισμικό (Trojan, worm) το οποίο δημιουργείται με σκοπό να ανιχνεύει ευάλωτες συσκευές, να αποκτά πρόσβαση και να τις ελέγχει απομακρυσμένα.

7. Ransomware: είναι το λογισμικό εκείνο, το οποίο κρυπτογραφεί τα προσωπικά δεδομένα του χρήστη, καθιστώντας τα απροσπέλαστα και στη συνέχεια ζητάει λύτρα για την ανάκτησή τους.

2.2 Τεχνικές ανίχνευσης κακόβουλου λογισμικού

Οι τεχνικές ανίχνευσης κακόβουλου λογισμικού χρησιμοποιούνται για να αναγνωρίσουν την ύπαρξη ενός τέτοιου προγράμματος και να εμποδίζουν τη μόλυνση ολόκληρου του συστήματος που μπορεί να οδηγήσει σε απώλεια ή κλοπή δεδομένων και την ολοκληρωτική καταστροφή του συστήματος.

Οι τεχνικές αυτές μπορεί να διαχωριστούν σε signature based detection (ανίχνευση με βάση την υπογραφή), behavior based detection (ανίχνευση με βάση την συμπεριφορά), heuristic based detection (ευριστική ανίχνευση) [6].



Εικόνα 1: Μέθοδοι ανίχνευσης malware

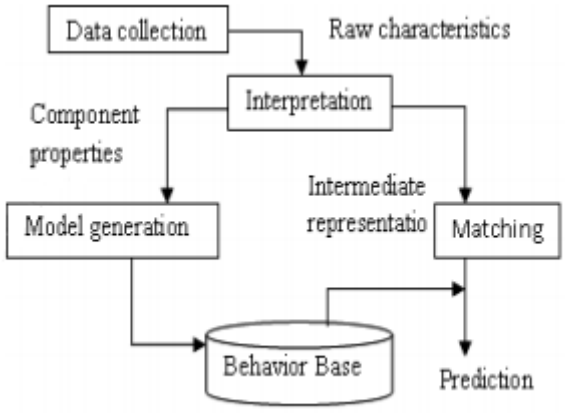
2.2.1 Signature based detection

Η τεχνική αυτή, βασίζεται στην αναζήτηση συγκεκριμένων ακολουθιών από byte, οι οποίες έχουν εντοπιστεί σε κακόβουλα λογισμικά, στα περιεχόμενα ενός αρχείου προκειμένου όταν υπάρξει ταυτοποίηση στις συμβολοσειρές να μπορέσει να το χαρακτηρίσει ως κακόβουλο. Οι ακολουθίες αυτές είναι γνωστές με τον όρο «virus signature» (υπογραφή ιού) και αποτελούν το «δακτυλικό αποτύπωμα» οποιουδήποτε κακόβουλου λογισμικού. Θεωρείται από τις πιο απλοϊκές τεχνικές και είναι άρρηκτα συνδεδεμένη με τη χρήση βάσεων δεδομένων που εμπεριέχουν τις υπογραφές των λογισμικών αυτών. Για την δημιουργία των υπογραφών απαιτείται η ανάλυση του κακόβουλου αρχείου και η εξαγωγή των χαρακτηριστικών του, έτσι ώστε να βρεθεί, μια μοναδική ακολουθία από συμβολοσειρές που το ταυτοποιούν. Η υπογραφή αυτή, μπορεί να είναι εκτός των άλλων και η τιμή της συνάρτησης κατακερματισμού του κακόβουλου λογισμικού.

Τα περισσότερα αντιιικά προγράμματα χρησιμοποιούν αυτή τη μέθοδο και συνεχίζουν να την αναπτύσσουν, παρότι έχει πολλές ανεπάρκειες. Μια από τις σημαντικότερες, αποτελεί η ανάγκη για συνεχή ενημέρωση των βάσεων, έτσι ώστε να περιλαμβάνει, τις πιο πρόσφατες υπογραφές ιομορφικού λογισμικού. Με την καθημερινή αύξηση του αριθμού των malware, οι βάσεις αυτές γίνονται όλο και μεγαλύτερες άρα απαιτείται όλο και μεγαλύτερος αποθηκευτικός χώρος για τη διατήρησή τους, πράγμα το οποίο έχει αντίκτυπο στην ταχύτητα αναζήτησης στη βάση συνεπώς και στην απόδοση ολόκληρου του συστήματος. Επιπλέον, είναι σύνηθες φαινόμενο, το κακόβουλο λογισμικό να μεταλλάσσεται με διάφορους τρόπους όπως κάνουν οι πολυμορφικοί ή οι μεταμορφικοί ιοί, με αποτέλεσμα οι υπογραφές τους να μεταβάλλονται και συνεπώς να μην μπορούν να ταυτοποιηθούν με τις υπάρχουσες υπογραφές. Τέλος το σημαντικότερο μειονέκτημα της τεχνικής αυτής είναι η ανικανότητα ανίχνευσης άγνωστων και καινοτόμων malware, τα οποία μπορούν να εισβάλουν στο σύστημα και να παραμείνουν εκεί έως ότου να αναγνωριστούν και να αναλυθούν από τις εταιρίες κατασκευής αντιιικών προγραμμάτων.

2.2.2 Behavior based detection

Στην συγκεκριμένη τεχνική, ένα πρόγραμμα χαρακτηρίζεται ως κακόβουλο εξετάζοντας και παρατηρώντας τη συμπεριφορά του. Σε αντίθεση με τις παραδοσιακές μεθόδους ανίχνευσης, οι οποίες βασίζονται στις υπογραφές, στην συμπεριφορική ανίχνευση αναλύονται οι ενέργειες που εκτελούνται από ένα πρόγραμμα τόσο σε άλλα αρχεία, όσο και, σε διεργασίες, σε ενέργειες δικτύου αλλά και σε κλειδιά της registry, προκειμένου να αναγνωριστεί και να ταυτοποιηθεί ένα πρόγραμμα ως κακόβουλο. Η διαδικασία αυτή εμφανίζεται στην εικόνα που ακολουθεί.



Εικόνα 2: Workflow συμπεριφορικής ανίχνευσης

Στην κατηγορία αυτή εντάσσονται οι τεχνικές ανίχνευσης ανωμαλιών (anomaly based detection) και η ανίχνευση βάση προδιαγραφών (specification based detection) [7]. Στην πρώτη προσέγγιση, δημιουργείται ένα προφίλ εκτέλεσης ενός προγράμματος το οποίο θεωρείται κανονικό και οποιαδήποτε απόκλιση από αυτό χαρακτηρίζεται ως «ανώμαλο» και επομένως ύποπτο. Το μεγαλύτερο μειονέκτημα της προσέγγισης αυτής είναι ο υψηλός αριθμός από λάθος θετικούς χαρακτηρισμούς εξαιτίας της πολυπλοκότητας των σύγχρονων προγραμμάτων η οποία καθιστά τη δημιουργία του κανονικού προφίλ σχεδόν αδύνατη. Η ανίχνευση βάση προδιαγραφών είναι παρόμοια με την ανίχνευση ανωμαλιών με κύρια διαφορά ότι χρειάζεται η δημιουργία μιας πολιτικής βάση της οποίας θα ελέγχονται όλα τα προγράμματα [6] και θα καθορίζει ποιές ενέργειες θα παρθούν για μια συγκεκριμένη ακολουθία γεγονότων [7]. Με την τεχνική αυτή μπορούν να ανιχνευτούν τόσο γνωστά όσο και άγνωστα κακόβουλα λογισμικά αλλά λόγο της πολιτικής που δημιουργείται ο αριθμός των εσφαλμένος θετικών είναι μεγάλος.

Η τεχνική αυτή, αποτελεί μια εξυπνότερη προσέγγιση στην ανίχνευση του κακόβουλου λογισμικού, καθώς μπορεί να αντιμετωπίσει τις μεταλλάξεις του. Αν και κατά τη μετάλλαξη του κακόβουλου λογισμικού οι υπογραφές του μεταβάλλονται, τείνει να χρησιμοποιεί τους πόρους του συστήματος με τον ίδιο ακριβώς τρόπο [8].

2.2.3 Heuristic based detection

Είναι θέμα διαφωνίας, κατά πόσο οι ευριστικές μέθοδοι ανίχνευσης κακόβουλου λογισμικού είναι όμοιοι με τις συμπεριφοριακές. Στην πραγματικότητα, μια λεπτή γραμμή διαχωρίζει τις μεθόδους αυτές [9] με τις κύριες διαφορές να εντοπίζονται στις λειτουργικότητες και τους τρόπους που ανιχνεύουν το κακόβουλο λογισμικό.

Στις ευριστικές μεθόδους, ακολουθείτε μια διαδικασία απομεταγλώττισης του αρχείου προκειμένου να ελεγχθεί ο κώδικας και η δομή του, έτσι ώστε να μπορέσει να ταυτοποιηθεί με αυτόν ενός ήδη γνωστού κακόβουλου λογισμικού. Επίσης, σε αντίθεση με την συμπεριφοριακή ανίχνευση, στις ευριστικές μεθόδους γίνεται προσομοίωση εκτέλεσης του ύποπτου αρχείου προκειμένου να εντοπιστούν δραστηριότητες όπως «ύποπτες» κλήσεις συστήματος, ενέργειες δικτύου κ.α.

Τα πλεονεκτήματα της προσέγγισης αυτής είναι, ότι δεν χρειάζεται εκπαίδευση και μπορεί να ανιχνεύσει καινούργια κακόβουλα λογισμικά, αφού δεν βασίζεται στις υπογραφές

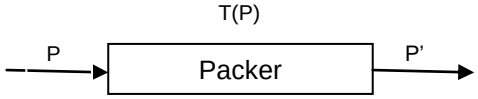
τους. Από τη άλλη, δημιουργεί μεγάλο αριθμό από εσφαλμένος θετικές ενδείξεις και λόγω των διαδικασιών που εκτελεί κάνει το σύστημα πιο αργό. Τέλος, οι προγραμματιστές των κακόβουλων λογισμικών μπορούν εύκολα, με τη χρήση packer να περιπλέξουν τη διαδικασία ανάλυσης τους καθιστώντας τέτοιου είδους μεθόδους αναποτελεσματικές.

2.3 Τεχνικές αποφυγής ανίχνευσης

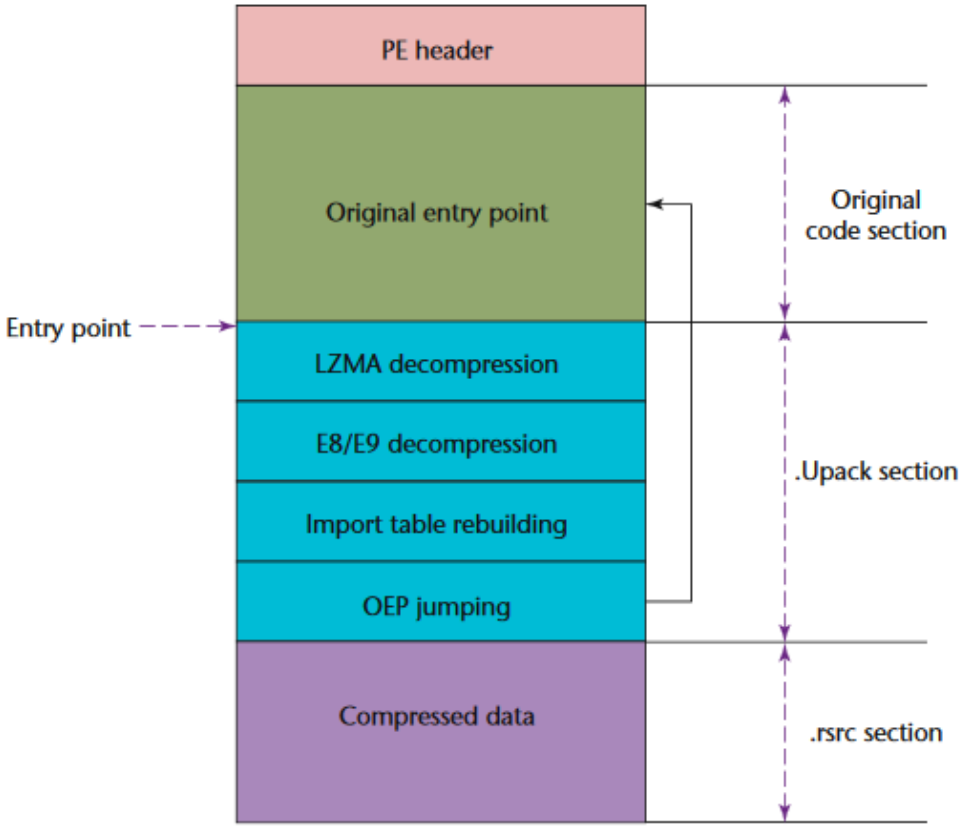
Οι τεχνικές αποφυγής ανίχνευσης, είναι ένα πολύ συχνό φαινόμενο για τα κακόβουλα λογισμικά. Είναι γεγονός ότι πάνω από το 50% των νέων malware είναι απλά «ανακατασκευασμένες» μορφές παλαιότερων τέτοιων λογισμικών και πάνω από το 80% χρησιμοποιούν τέτοιου είδους τεχνικές για «ξεφεύγουν» από την ανίχνευση [10].

Κάποιες από τις πλέον διαδεδομένες τεχνικές είναι ο πολυμορφισμός, ο μεταμορφισμός και η κρυπτογράφηση του εκτελέσιμου κώδικα. Η πιο διαδεδομένη όμως μέθοδος, λόγω της πληθώρας εργαλείων που είναι διαθέσιμα, και αυτή με την οποία θα ασχοληθούμε, είναι το «packing».

Οι packers είναι λογισμικά τα οποία για ένα δεδομένο πρόγραμμα P παράγουν ένα πρόγραμμα P’ το οποίο περιέχει μια κρυπτογραφημένη έκδοση του αρχικού προγράμματος και μια διαδικασία αποκρυπτογράφησης. Με την εκτέλεση του προγράμματος P’ αποκρυπτογραφούνται τα περιεχόμενα του P και εκτελείται το πρόγραμμα.



Στο παράδειγμα που ακολουθεί βλέπουμε το workflow του UPack ενός packer για windows portable executable αρχεία. Τα δεδομένα του αρχικού προγράμματος βρίσκονται στο «.rsrc» section, ενώ οι ρουτίνες αποκρυπτογράφησης στο «.Upack» section.



Εικόνα 3: Workflow του UPack

Για την ανίχνευση των packed εκτελέσιμων, έχουν προταθεί αρκετές λύσεις. Εμείς επιλέξαμε να χρησιμοποιήσουμε την τεχνική ανίχνευσης με τη χρήση υπογραφών. Υπάρχει μια μεγάλη ποικιλία από signature-based detectors στο διαδίκτυο [11] οι οποίοι λειτουργούν βασιζόμενοι σε βάσεις δεδομένων που περιέχουν κανόνες και υπογραφές από διαφορετικούς packers. Η ανίχνευση επιτυγχάνεται αναζητώντας τις συγκεκριμένες υπογραφές είτε στο «entry-point» του PE αρχείου, είτε σε ολόκληρο το αρχείο. Προτείνεται η αναζήτηση να περιορίζεται στο «entry-point» καθώς ο πλήρης έλεγχος του αρχείου είναι επιρρεπής σε μεγάλο αριθμό από εσφαλμένος θετικές ενδείξεις [12].

3 Υλοποίηση

Στο κεφάλαιο αυτό, περιγράφεται ο αλγόριθμος του εργαλείου που αναπτύχθηκε με σκοπό τον έλεγχο των αρχείων και των πτητικών δεδομένων, windows 8 λειτουργικών συστημάτων για εντοπισμό κακόβουλου λογισμικού και ενδείξεις παραβίασης του.

3.1 Περιγραφή Αλγορίθμου

Ο αλγόριθμος αναπτύχθηκε με σκοπό τη συλλογή δεδομένων από υπολογιστές που λειτουργούν με windows 8 λειτουργικό σύστημα. Στα δεδομένα περιλαμβάνονται τα ονόματα και τα πλήρη μονοπάτια των αρχείων του υπολογιστή, οι τιμές της συνάρτησης κατακερματισμού τους και οι πληροφορίες που παρέχονται από το virus total για τον ρυθμό ανίχνευσης του κάθε αρχείου. Περιέχει επίσης πληροφορίες, που αφορούν τον έλεγχο των πόρων και των ψηφιακών υπογραφών των αρχείων του συστήματος, καθώς και τα αποτελέσματα των ελέγχων με κανόνες yara τόσο στα αρχεία του συστήματος όσο και στις ενεργές διεργασίες. Αποθηκεύει επίσης τα αποτελέσματα της ανίχνευσης για χρήση packer σε εκτελέσιμα αρχεία και τέλος, δίνει τη δυνατότητα συλλογής και ανάλυσης, της μνήμης και της registry του υπολογιστή με χρήση εξωτερικών προγραμμάτων ανοιχτού κώδικα.

Για τη διατήρηση των δεδομένων αυτών, κρίθηκε απαραίτητη η δημιουργία μιας sqlite βάσης δεδομένων, η οποία αποτελείται από δυο επιμέρους πίνακες. Στον πρώτο πίνακα, καταχωρούνται όλες οι πληροφορίες του αρχείου της NSRL, ενώ στο δεύτερο τα δεδομένα που συλλέγονται από τον υπολογιστή. Κάθε εγγραφή του πίνακα αφορά ένα ξεχωριστό αρχείο του υπολογιστή και κάθε στήλη του, ένα γνώρισμα του αρχείου αυτού.

3.2 Δημιουργία της βάσης δεδομένων

Όπως αναφέρθηκε, το πρώτο βήμα του αλγορίθμου είναι η δημιουργία της βάσης δεδομένων, στην οποία θα αποθηκεύονται όλες οι πληροφορίες οι οποίες συλλέγονται κατά την εκτέλεση του προγράμματος.

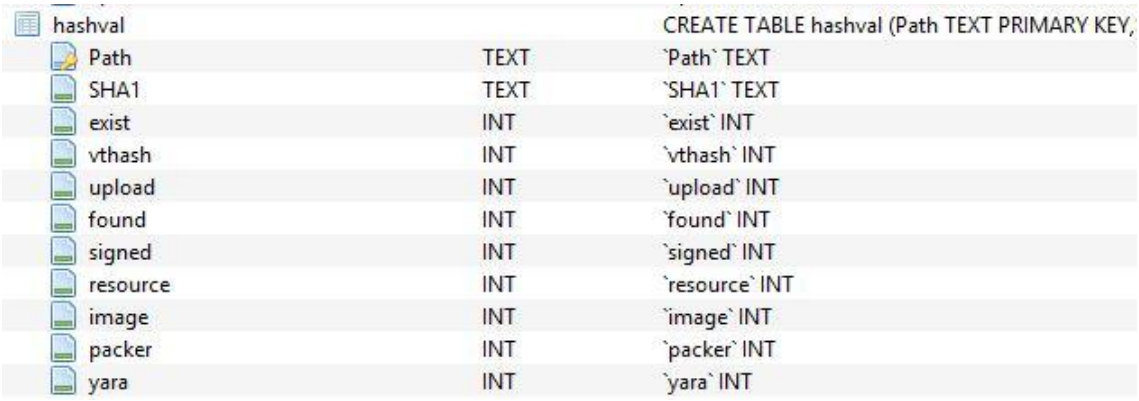
Η βάση αυτή αποτελείται από δύο επιμέρους πίνακες. Στον πρώτο πίνακα με όνομα NIST, περιέχονται όλες οι πληροφορίες του αρχείου NSRL που δημοσιεύεται από τον οργανισμό NIST. Περισσότερες πληροφορίες για το συγκεκριμένο αρχείο παραθέτονται στο κεφάλαιο 4.2.1

Nist		CREATE TABLE Nist(SHA1 TEXT PRIMARY KEY,
SHA1	TEXT	`SHA1` TEXT
MD5	TEXT	`MD5` TEXT
CRC32	TEXT	`CRC32` TEXT
FileName	TEXT	`FileName` TEXT
FileSize	TEXT	`FileSize` TEXT
ProductCode	TEXT	`ProductCode` TEXT
OpSystemCode	TEXT	`OpSystemCode` TEXT
SpecialCode	TEXT	`SpecialCode` TEXT

Εικόνα 4: Πεδία του πίνακα NSRL

Για τη δημιουργία του πίνακα, ο χρήστης πρέπει να δώσει σαν είσοδο το πλήρες μονοπάτι που βρίσκεται αποθηκευμένο το αρχείο NSRL. Το αρχείο αναλύεται αυτόματα και τα περιεχόμενά του καταχωρούνται στον πίνακα, ο οποίος αποτελείται από οκτώ στήλες (SHA1, MD5, CR32, Filename, FileSize, ProductCode, OpSystemCode, SpecialCode) όπως ακριβώς και το αρχείο, με κλειδί της βάσης το πεδίο SHA1. Περισσότερα για τη συγκεκριμένη συλλογή δεδομένων αναφέρονται στο επόμενο κεφάλαιο.

Στο δεύτερο πίνακα της βάσης δεδομένων με όνομα «hashval» συγκρατούνται όλες οι πληροφορίες που αφορούν τα αρχεία του υπολογιστικού συστήματος.



hashval		CREATE TABLE hashval (Path TEXT PRIMARY KEY,
Path	TEXT	`Path` TEXT
SHA1	TEXT	`SHA1` TEXT
exist	INT	`exist` INT
vthash	INT	`vthash` INT
upload	INT	`upload` INT
found	INT	`found` INT
signed	INT	`signed` INT
resource	INT	`resource` INT
image	INT	`image` INT
packer	INT	`packer` INT
yara	INT	`yara` INT

Εικόνα 5: Πεδία του πίνακα hashval

Τα δεδομένα αυτά είναι, τα πλήρη μονοπάτια των αρχείων, οι τιμές της συνάρτησης κατακερματισμού τους και ο χαρακτηρισμός του virus total για τα αρχεία αυτά, με βάση την τιμή της συνάρτησης κατακερματισμού. Επίσης αποθηκεύει τιμές, οι οποίες προσδιορίζουν κατά πόσο ένα αρχείο έχει υποβληθεί επιτυχώς στο virus total για ανάλυση, αν το συγκεκριμένο hash value του αρχείου βρίσκεται στη βάση της NSRL, αν το αρχείο έχει έγκυρη ψηφιακή υπογραφή από έμπιστη αρχή πιστοποίησης καθώς και αν το αρχείο περιέχει στις λεπτομέρειές του «legal copyright» και τη δική του εικόνα. Διατηρούνται επίσης δεδομένα που υποδηλώνουν την εύρεση υπογραφών που ταιριάζουν σε αυτές γνωστών packers και τέλος στη βάση αποθηκεύονται τα αποτελέσματα των ελέγχων με yara rules που γίνονται σε όλα τα εκτελέσιμα αρχεία. Ο πίνακας αυτός δημιουργείται αυτόματα με την εκτέλεση του δεύτερου βήματος του αλγορίθμου, στο οποίο υπολογίζονται οι τιμές της συνάρτησης κατακερματισμού των αρχείων.

Με την ολοκλήρωση του βήματος αυτού έχει δημιουργηθεί η βάση δεδομένων και ο πρώτος πίνακας έχει ήδη γεμίσει με τα δεδομένα. Ο συγκεκριμένος πίνακας χρησιμοποιείται σε μεταγενέστερο στάδιο του αλγορίθμου και θα περιγραφεί στη συνέχεια

3.3 Υπολογισμός τιμών συνάρτησης κατακερματισμού

Στο επόμενο βήμα, όπως αναφέρθηκε, δημιουργείται ο δεύτερος πίνακας της βάσης δεδομένων. Ο πίνακας δημιουργείται με σκοπό να αποθηκεύσει τα πλήρη μονοπάτια και την τιμή της συνάρτησης κατακερματισμού του κάθε αρχείου, που βρίσκεται στο σύστημα.

Αρχικά, ο χρήστης πρέπει να εισαγάγει το πλήρες μονοπάτι της βάσης, καθώς και τον κατάλογο-ρίζα του υπολογιστή (έστω C:\). Το script αυτόματα ανιχνεύει αν ο χρήστης έχει προνόμια διαχειριστή, σε διαφορετική περίπτωση εμφανίζεται το παράθυρο ελέγχου λογαριασμού χρήστη (UAC), ζητώντας τα απαραίτητα δικαιώματα, προκειμένου να μπορέσει να αποκτήσει πρόσβαση σε αρχεία και φακέλους τα οποία είναι προστατευμένα από το λειτουργικό σύστημα. Έπειτα ανοίγει τη βάση δεδομένων και ελέγχει αν υπάρχει ο δεύτερος πίνακας, στον οποίο αποθηκεύονται τα δεδομένα του υπολογιστή. Στο σημείο αυτό προκύπτουν δύο περιπτώσεις:

1. Ο πίνακας δεν υπάρχει στη βάση: Αυτό οδηγεί στο συμπέρασμα ότι είναι η πρώτη φορά που εκτελείται το εργαλείο, οπότε δημιουργεί τον πίνακα και συνεχίζει τη διαδικασία όπως περιγράφεται στη συνέχεια
2. Ο πίνακας υπάρχει ήδη στη βάση: Ο αλγόριθμος έχει εκτελεστεί ξανά και υπάρχουν ήδη δεδομένα μέσα στον πίνακα. Στην περίπτωση αυτή ο χρήστης καλείται να επιλέξει ανάμεσα στα ακόλουθα:
    1. Διαγραφή του πίνακα: Διαγράφει τον πίνακα και τον δημιουργεί εκ νέου, πράγμα που οδηγεί σε απώλεια όλων των δεδομένων που υπήρχαν στον συγκεκριμένο πίνακα της βάσης δεδομένων και η διαδικασία συνεχίζεται σαν να μην υπήρχε ο πίνακας εξ αρχής.
    2. Διατήρηση του πίνακα: Τα δεδομένα που έχει ο πίνακας παραμένουν και η διαδικασία συνεχίζεται όπως περιγράφεται στη συνέχεια.

Ο πίνακας αποτελείται από έντεκα στήλες (Path, SHA1, exists, vthash, upload, found, signed, resource, image, packer, yara) με πρωτεύον κλειδί το πεδίο Path, αφού αποκλείεται να υπάρχει δύο φορές το ίδιο ακριβώς path (δεν επιτρέπεται δυο αρχεία που είναι στον ίδιο φάκελο, να έχουν το ίδιο όνομα).

Στην περίπτωση που ο πίνακας δεν υπήρχε στη βάση δεδομένων, δημιουργείται και ανοίγεται, με σκοπό να ξεκινήσει η εισαγωγή δεδομένων. Σειριακά όλα τα αρχεία που βρίσκονται κάτω από τον κατάλογο-ρίζα ανοίγονται σε δυαδική μορφή (read binary) και υπολογίζεται η τιμή της συνάρτησης κατακερματισμού τους.

Οι συναρτήσεις κατακερματισμού (hash functions) είναι μια από τις σημαντικότερες και πιο συνηθισμένες ορολογίες στο χώρο της ασφάλειας, καθώς βρίσκουν εφαρμογή σε πολλά διαφορετικά πεδία, όπως προστασία κωδικών χρήστη, δημιουργία ψηφιακών υπογραφών, αυθεντικοποίηση μηνυμάτων και πολλά ακόμη. Είναι συναρτήσεις οι οποίες «συμπιέζουν» μια είσοδο με τυχαίο μέγεθος σε ένα αποτέλεσμα ορισμένου μεγέθους [13], για να είναι όμως χρήσιμες πρέπει να ικανοποιούν και κάποιες ακόμα προδιαγραφές:

- πρέπει να είναι δύσκολο να βρεθούν δύο διαφορετικοί είσοδοι που να έχουν την ίδια έξοδο
- για ένα δεδομένο hash value πρέπει να είναι δύσκολο να εντοπίσουμε μια είσοδο που το αποτέλεσμα της συνάρτησης κατακερματισμού της να έχει την ίδια έξοδο
- για μια δεδομένη είσοδο πρέπει να είναι δύσκολο να βρεθεί μια άλλη είσοδος που να έχει το ίδιο hash value

Από τις πλέον γνωστές και ευρέως χρησιμοποιούμενες συναρτήσεις κατακερματισμού είναι οι SHA (secure hash algorithms). Για την εφαρμογή που αναπτύχθηκε επιλέχθηκε η συνάρτηση SHA1, κυρίως λόγο της βάσης NSRL, η οποία υποστηρίζει μόνο τις συναρτήσεις MD5 και SHA1.

Με σκοπό τη μείωση του όγκου των δεδομένων που θα καταχωρηθούν στη βάση εκτελείται η εντολή «Pathext» έτσι ώστε να βρεθούν οι καταλήξεις των αρχείων που θεωρούνται από το σύστημα εκτελέσιμα και εισάγονται στη βάση μαζί με τα μονοπάτια όλων των βιβλιοθηκών δυναμικής σύνδεσης «.dll».

Δημιουργείται επίσης ένα έγγραφο κειμένου στο οποίο αποθηκεύονται όλα τα αρχεία τα οποία δεν κατάφεραν να ανοιχτούν έτσι ώστε ο χρήστης να μπορέσει να τα ελέγξει χειροκίνητα. Με το πέρας του υπολογισμού το αποτέλεσμα μαζί με το πλήρες μονοπάτι του αρχείου εισάγονται στη βάση στα πεδία SHA1 και Path αντίστοιχα και όλες οι υπόλοιπες στήλες παίρνουν τιμή NULL.

Στην περίπτωση που ο πίνακας υπήρχε ήδη, πράγμα που οδηγεί στο συμπέρασμα ότι υπάρχουν ήδη δεδομένα μέσα στον πίνακα αυτόν, ξεκινάει ξανά η διαδικασία υπολογισμού του hash value για κάθε αρχείο αλλά αυτή τη φορά κατά την εισαγωγή των δεδομένων στη βάση διακρίνονται τρεις διαφορετικές περιπτώσεις:

1. Οι νέες τιμές δεν υπάρχουν στη βάση: Αν μια τιμή δεν βρεθεί στη βάση τότε γίνεται η εισαγωγή της στον πίνακα και το πεδίο exists παίρνει την τιμή δύο.
2. Το πλήρες μονοπάτι υπάρχει με διαφορετικό hash value: Σε αυτή την περίπτωση διαγράφεται η παλιά εγγραφή και εισάγεται το αρχείο με το νέο hash value. Διαφορετικό hash value πρακτικά συνεπάγεται ότι το αρχείο έχει μεταβληθεί από την πρώτη φορά που καταχωρήθηκε στη βάση οπότε το πεδίο exists παίρνει την τιμή δύο.
3. Το μονοπάτι και το hash value παραμένουν τα ίδια: το αρχείο υπήρχε και δεν μεταβλήθηκε από την εισαγωγή του στη βάση οπότε το πεδίο exist γίνεται ένα.

Συνοψίζοντας, όλα τα αρχεία που έχουν τιμή Null στο πεδίο exists υπήρχαν κατά την εκτέλεση του αλγορίθμου την πρώτη φορά. Αν η τιμή είναι δύο, το αρχείο καταχωρήθηκε τη δεύτερη φορά, διαφορετικά αν είναι ένα, το αρχείο υπήρχε στη βάση και έμεινε αναλλοίωτο.

3.4 Σύγκριση δεδομένων με τη βάση NSRL

Στο επόμενο βήμα, δίνεται η δυνατότητα στο χρήστη να εντοπίσει ποια από τα αρχεία βρίσκονται στη βάση της NSRL. Η ταυτοποίηση ενός αρχείου με τη συγκεκριμένη βάση

δεδομένων, συνεπάγεται ότι το αρχείο αυτό αποτελεί κομμάτι κάποια γνωστής εφαρμογής και συνεπώς δεν θεωρείται «ύποπτο».

Ο χρήστης δίνει σαν όρισμα το μονοπάτι που βρίσκεται η βάση δεδομένων, στη συνέχεια η βάση ανοίγεται και όλα τα hash values που βρίσκονται στο πεδίο SHA1 του δεύτερου πίνακα, αναζητούνται στο αντίστοιχο πεδίο του πίνακα ένα, ο οποίος περιέχει το αρχείο της NSRL. Αν υπάρξει ταυτοποίηση στην τιμή τότε το πεδίο found του δεύτερου πίνακα γίνεται ένα, διαφορετικά παίρνει την τιμή μηδέν.

Ο έλεγχος αυτός αποτελεί ένα σημαντικό μέρος του αλγορίθμου. Τα αρχεία που ταυτοποιούνται με τη βάση NSRL εξαιρούνται από τις επόμενες διαδικασίες ελέγχου κάτι το οποίο βοηθάει στη μείωση του όγκου των δεδομένων που πρέπει να αναλυθούν περαιτέρω.

3.5 Αποστολή hash values στο VirusTotal

Στη συνέχεια, δίνεται η δυνατότητα για αυτοματοποιημένη αποστολή των hash values στο virus total. Μέσω του site αυτού, δίνεται η δυνατότητα ελέγχου των τιμών της συνάρτησης κατακερματισμού, ελέγχοντας την τιμή που υποβλήθηκε με μια τεράστια βάση που περιέχει όλα τα hash values από αρχεία που έχουν ήδη αναλυθεί από το site. Αν το hash value βρεθεί τότε το site μας επιστρέφει την αναφορά ανάλυσης που ήδη έχει στη βάση του. Για να μειωθεί ο αριθμός των αιτήσεων που αποστέλλονται από τη διαδικασία εξαιρούνται όπως έχει ήδη αναφερθεί, όλα τα αρχεία που βρέθηκαν στο αρχείο NSRL καθώς θεωρούνται καλόβουλα.

Οι αιτήσεις γίνονται με τη χρήση του api του virus total. Κάθε αίτηση περιέχει ένα hash value και το κλειδί που έχει προμηθευτεί ο κάθε χρήστης από το site. Το κλειδί πληκτρολογείται από το χρήστη όταν ζητηθεί από την εφαρμογή και στη συνέχεια, ο χρήστης καλείται να επιλέξει αν το κλειδί είναι ιδιωτικό ή δημόσιο. Για κάθε αίτημα που ολοκληρώνεται, η απάντηση που λαμβάνεται έχει τη μορφή ενός json αντικειμένου, το οποίο αναλύεται με σκοπό να εξαχθεί όλη η χρήσιμη πληροφορία που θα καθορίσει την τιμή που θα πάρει το πεδίο vthash της βάσης.

Αν από την απάντηση προκύψει ότι το συγκεκριμένο hash value δεν υπάρχει στη βάση του virus total η τιμή του πεδίου vthash γίνεται ίση με μηδέν. Σε διαφορετική περίπτωση αν το αρχείο υπάρχει στη βάση του virus total και έχει ρυθμό ανίχνευσης από τα αντιικά προγράμματα μεγαλύτερο του μηδέν έχει χαρακτηριστεί δηλαδή έστω και από ένα antivirus ως κακόβουλο, τότε η τιμή του πεδίου γίνεται δύο και η πλήρη αναφορά του virus total αποθηκεύεται σε ένα αρχείο με τίτλο το hash value του αρχείου.

Αν η αναφορά του virus total δηλώνει ότι το αρχείο είναι «καθαρό», έχει δηλαδή ρυθμό ανίχνευσης μηδέν, τότε το συγκεκριμένο πεδίο της βάσης παίρνει την τιμή ένα. Όλα τα αρχεία που χαρακτηρίζονται «καθαρά» από το site γράφονται σε ένα αρχείο με όνομα «clean-files.txt».

Είναι σημαντικό να υπενθυμίσουμε, ότι ο ρυθμός των αιτήσεων προς το virus total όσον αφορά το δημόσιο κλειδί, είναι τέσσερις το λεπτό γι αυτό το λόγο υπάρχει μια συνθήκη που εξασφαλίζει ότι για κάθε αίτηση θα υπάρχει ένα «sleep time» 16 δευτερολέπτων. Όλες οι αναφορές, είτε πρόκειται για αρχεία με ρυθμό ανίχνευσης μηδέν είτε για αρχεία που έχουν χαρακτηριστεί ως «κακόβουλα», αποθηκεύονται σε ένα φάκελο που δημιουργείται αυτόματα στον κατάλογο από τον οποίο εκτελούμε το εργαλείο.

3.6 Αποστολή αρχείων στο VirusTotal

Στο επόμενο βήμα, ο χρήστης μπορεί να επιλέξει να ανεβάσει τα αρχεία στο virus total για ανάλυση. Τα αρχεία αναλύονται, χρησιμοποιώντας τις υπηρεσίες πάνω από 50 διαφορετικών αντιικών προγραμμάτων, τα οποία συνεργάζονται με το VirusTotal. Στο σημείο αυτό πρέπει να τονίσουμε ότι αν πρόκειται για δημόσιο κλειδί τα αρχεία δεν πρέπει να ξεπερνούν τα 32MB.

Ο χρήστης, πρέπει να δώσει σαν όρισμα το μονοπάτι που βρίσκεται η βάση και το κλειδί για το api του virus total. Για την απόκτηση ενός δημόσιου κλειδιού απαιτείται η δημιουργία λογαριασμού στην κοινότητα του VirusTotal μέσω της ιστοσελίδας του. Με την ολοκλήρωση της διαδικασίας εγγραφής το κλειδί μπορεί να βρεθεί στα στοιχεία του προφίλ του χρήστη. Στη συνέχεια το script ζητάει δικαιώματα διαχειριστή, προκειμένου να μπορέσει να ανοίξει τα αρχεία σε δυαδική μορφή και να τα αποστείλει στο virus total. Προκειμένου να

ικανοποιηθεί η συνθήκη μεγέθους, πριν την αποστολή ελέγχονται τα αρχεία ως προς το μέγεθος και τελικά αποστέλλονται μόνο όσα ικανοποιούν τον περιορισμό. Για τα αρχεία που δεν ικανοποιούν τους περιορισμούς του virus total η τιμή του πεδίου upload στον πίνακα hashval παίρνει την τιμή τρία.

Για κάθε αίτηση που ολοκληρώνεται, αναλύεται εκ νέου το json αντικείμενο, που αποτελεί μέρος της απάντησης του virus total. Αν από την απάντηση προκύπτει ότι το αρχείο υποβλήθηκε χωρίς σφάλματα, τότε το πεδίο upload παίρνει την τιμή ένα διαφορετικά, αν για οποιοδήποτε λόγο απέτυχε η υποβολή του, η τιμή γίνεται δυο.

3.7 Ανάκτηση αναφορών

Στη συνέχεια, ο χρήστης μπορεί να επιλέξει να ανακτήσει τις αναφορές των αναλύσεων. Για όλα τα path, στα οποία το πεδίο upload είναι ίσο με ένα και το vthash είναι μηδέν, δηλαδή για όλα τα αρχεία που η αποστολή τους στο virus total ολοκληρώθηκε με επιτυχία και το hash value τους δεν υπήρχε στην web υπηρεσία, γίνεται ανάκτηση των τιμών της συνάρτησης κατακερματισμού, από τον πίνακα της βάσης δεδομένων και αποστέλλονται στο virus total όπως ακριβώς και στο προηγούμενο βήμα.

Το δημόσιο κλειδί δεν εγγυάται την άμεση ανάλυση των αρχείων που υποβάλλονται. Γι αυτό το λόγο, έχει προστεθεί μια επιπλέον επιλογή στην συνάρτηση που είναι υπεύθυνη για την ανάκτηση των αναφορών, προκειμένου να ενημερώνει το χρήστη σε περίπτωση που τα αρχεία που έχει υποβάλει είναι ακόμα στην ουρά.

Αν το virus total έχει ολοκληρώσει τη διαδικασία ανάλυσης των αρχείων, τότε οι αναφορές συλλέγονται και το πεδίο vthash του πίνακα μεταβάλλεται με τον ίδιο ακριβώς τρόπο όπως στο υπό κεφάλαιο 3.5

3.8 Έλεγχος ψηφιακών υπογραφών

Στη συνέχεια δίνεται η δυνατότητα στο χρήστη να εντοπίσει ποια από τα αρχεία είναι ψηφιακά υπογεγραμμένα. Η ψηφιακή υπογραφή είναι ένα πολύ σημαντικό στοιχείο για τα αρχεία, καθώς υποδηλώνει την ακεραιότητα του αρχείου και πιστοποιεί τον εκδότη του.

Για τον έλεγχο ύπαρξης ψηφιακών υπογραφών στα αρχεία, ο χρήστης πρέπει να παρέχει το μονοπάτι της βάσης, καθώς και το πλήρες μονοπάτι του εργαλείου Sigcheck. Για κάθε αρχείο ξεχωριστά εκτελείται το συγκεκριμένο πρόγραμμα. Τα αποτελέσματα του εργαλείου αναλύονται αυτόματα αναζητώντας το πεδίο «signed» και αν από την ανάλυση βρεθεί ότι το αρχείο έχει ψηφιακή υπογραφή, τότε το πεδίο signed της βάσης παίρνει την τιμή ένα, διαφορετικά παίρνει την τιμή μηδέν.

Η ύπαρξη και μόνο της ψηφιακής υπογραφής σε ένα αρχείο, δεν είναι αρκετή για να εξασφαλίσει ότι το αρχείο δεν είναι κακόβουλο, αλλά συνδυαστικά με τις υπόλοιπες πληροφορίες που εξάγουμε μπορούμε να οδηγηθούμε σε ασφαλέστερα συμπεράσματα.

3.9 Έλεγχος χρήσης packer

Στο επόμενο βήμα, ελέγχουμε τα PE αρχεία για τη χρήση κάποιου packer στη δημιουργία τους, με τη χρήση του module «pefile» και του αρχείου που περιέχει τις υπογραφές γνωστών packers.

Ο χρήσης, παρέχει τα μονοπάτια της βάσης δεδομένων και του αρχείου που περιέχει τις υπογραφές, στη συνέχεια όλα τα μονοπάτια των εκτελέσιμων αρχείων ανασύρονται από τη βάση δεδομένων, οι υπογραφές φορτώνονται από το αρχείο και ξεκινάει ο έλεγχος των υπογραφών . Για κάθε αρχείο που ταυτοποιείται με κάποια υπογραφή η τιμή του πεδίου packer του πίνακα hashval μεταβάλλεται σε ένα, διαφορετικά γίνεται μηδέν.

Όπως και στην ψηφιακή υπογραφή, η ένδειξη ότι έχει χρησιμοποιηθεί packer σε ένα οποιοδήποτε αρχείο, δεν μπορεί να αποτελέσει καταδικαστικό μέτρο για την ταξινόμησή του, σε κακόβουλο ή καλόβουλο, καθώς, ακόμα και νόμιμα αρχεία εφαρμογών, μπορούν να ταυτοποιηθούν με βάση τις υπογραφές αυτές, καθώς χρησιμοποιούν packers για να μειώσουν

το μέγεθος των εκτελέσιμων, επιτρέποντας τον πιο γρήγορο διαμοιρασμό τους μέσω δικτύου και καθιστούν δύσκολη τη διαδικασία του reverse-engineering.

3.10 Κανόνες Yara

Στο επόμενο στάδιο, δίνεται η δυνατότητα ελέγχου των αρχείων χρησιμοποιώντας κανόνες yara. Όλα τα αρχεία του συστήματος ελέγχονται σειριακά και αναζητούνται συγκεκριμένες συμβολοσειρές στα διάφορα τμήματα του αρχείου, που ταιριάζουν με τις συμβολοσειρές γνωστών κακόβουλων προγραμμάτων. Το στάδιο αυτό χωρίζεται σε δυο επιμέρους διαδικασίες:

1. Μεταγλώττιση των κανόνων: Για την μεταγλώττιση των κανόνων, ο χρήστης πρέπει να ορίσει το φάκελο στον οποίο είναι αποθηκευμένα τα .yar αρχεία. Ο φάκελος αυτός προσπελαύνεται, αναζητώντας όλα τα αρχεία που έχουν την επιθυμητή κατάληξη και στη συνέχεια, τα αρχεία μεταγλωττίζονται και αποθηκεύονται στο φάκελο compiled rules που δημιουργείται στον τρέχον κατάλογο εργασίας.
2. Έλεγχος αρχείων: Προκειμένου να ξεκινήσει ο έλεγχος, ο χρήστης πρέπει να δώσει τα πλήρη μονοπάτια της βάσης δεδομένων και των μεταγλωττισμένων κανόνων. Όλα τα αρχεία της βάσης ανοίγονται και ελέγχονται αναδρομικά από όλους τους κανόνες. Ένας μετρητής αυξάνει για κάθε κανόνα που θα ταυτοποιηθεί και μόλις ολοκληρωθεί η διαδικασία η τιμή του μετρητή καταχωρείται στη βάση στο πεδίο yara.

3.11 Έλεγχος των resources

Κάθε εκτελέσιμο αρχείο εμπεριέχει πληροφορίες που αφορούν τον εκδότη του. Στο βήμα αυτό αναλύονται οι πληροφορίες αυτές του αρχείου με σκοπό τον εντοπισμό του πεδίου «copyright» και τον εντοπισμό ύπαρξης icon στο εκτελέσιμο αρχείο.

Χρησιμοποιώντας το module pefile, εξάγουμε πληροφορίες που αφορούν τα resources του αρχείου και ειδικότερα το icon και το Legal Copyright από τα Portable Executable αρχεία. Ο χρήστης δίνει το μονοπάτι της βάσης, τα αρχεία ανασύρονται σειριακά από τη βάση και αναλύονται με το pefile module. Εάν στο “fileinfo” του αρχείου βρεθεί η συμβολοσειρά «LegalCopyright» τότε στην εγγραφή του πίνακα hashval για το συγκεκριμένο αρχείο το πεδίο resources παίρνει την τιμή ένα.

Στη συνέχεια με τη χρήση του pywin32 module εξάγουμε το icon του κάθε εκτελέσιμου. Αν δεν εντοπιστεί εικόνα στο αρχείο, τότε το πεδίο image γίνεται ίσο με μηδέν, διαφορετικά παίρνει την τιμή ένα.

Οι πληροφορίες που παίρνουμε από αυτό το στάδιο σε συνδυασμό με την ύπαρξη ψηφιακής υπογραφής και το αποτέλεσμα ελέγχου για εντοπισμό υπογραφών packers μας βοηθούν να επικεντρώσουμε το ενδιαφέρον μας σε συγκεκριμένα αρχεία. Είναι φανερό ότι κάθε πληροφορία από μόνη της δεν μπορεί να μας προσφέρει ιδιαίτερες πληροφορίες αλλά αν συνδυαστούν έχουμε περισσότερες πιθανότητες να εντοπίσουμε κακόβουλα αρχεία στο σύστημά μας. Με την ολοκλήρωση και αυτού του βήματος έχουμε συλλέξει όλες τις πληροφορίες για τα αρχεία που χρειαζόμαστε και μπορούμε με πολύ εύκολο τρόπο, να συνδυάσουμε όλες τις απαραίτητες πληροφορίες, οι οποίες θα μας οδηγήσουν στα συμπεράσματα που επιθυμούμε. Με πολύ απλά ερωτήματα στη βάση μπορούμε να εντοπίσουμε τις τομές των συνόλων και να τα αναλύσουμε αποδοτικότερα.

3.12 Έλεγχος ενεργών διεργασιών

Οι διεργασίες του συστήματος μπορούν να κρύβουν ενδείξεις κακόβουλου κώδικα ακριβώς όπως και τα αρχεία. Στο βήμα αυτό, δίνεται η δυνατότητα στο χρήστη να ελέγξει όλες τις ενεργές διεργασίες χρησιμοποιώντας κανόνες yara, για την εύρεση συγκεκριμένων συμβολοσειρών στις διεργασίες που να υποδηλώνουν κακόβουλο κώδικα.

Ο χρήστης, δίνει σαν είσοδο τον κατάλογο που είναι αποθηκευμένοι οι κανόνες yara και η διαδικασία ξεκινάει ελέγχοντας μια προς μια τις διεργασίες με όλους τους κανόνες, οι οποίοι

βρίσκονται στον κατάλογο που έδωσε ο χρήστης. Για κάθε διεργασία δημιουργείται μια αναφορά που περιέχει τα στοιχεία της και αν ταυτοποιήθηκε από κάποιο κανόνα ή όχι.

3.13 Συλλογή και ανάλυση πτητικών δεδομένων

Τέλος, παρέχεται η δυνατότητα αυτοματοποιημένης απόκτησης και ανάλυσης της μνήμης και της registry του υπολογιστή με τη χρήση εξωτερικών εργαλείων.

Για την απόκτησή τους, ο χρήστης πρέπει να δώσει τα μονοπάτια των κατάλληλων προγραμμάτων (DumpIt και Forecopy αντίστοιχα). Το script αναγνωρίζει αν εκτελείται με τα κατάλληλα δικαιώματα, διαφορετικά μέσω του UAC ζητάει δικαιώματα διαχειριστή και εκτελεί αυτόματα τα εργαλεία, συλλέγοντας και αποθηκεύοντας τα πτητικά αυτά δεδομένα στον τρέχων κατάλογο εργασίας.

Για την ανάλυση της μνήμης ο χρήστης πρέπει να δώσει σαν όρισμα την τοποθεσία του volatility και του αντιγράφου της μνήμης. Το εργαλείο εκτελεί αυτόματα την πρώτη εντολή για τον εντοπισμό του κατάλληλου προφίλ και στη συνέχεια ζητάει από το χρήστη, να επιλέξει το κατάλληλο με βάση το αποτέλεσμα του συγκεκριμένου plugin. Έπειτα εκτελείται το εργαλείο με το plugin kdbg για τον εντοπισμό του σωστού offset και ο χρήστης εισάγει την επιθυμητή τιμή. Στη συνέχεια δημιουργείται ένας φάκελος με όνομα reports στον οποίο αποθηκεύονται όλες οι αναφορές από τα υπόλοιπα plugins που θα εκτελεστούν.

Όσον αφορά την ανάλυση της registry, ο χρήστης δηλώνει τα path του εργαλείου regripper και του φακέλου που βρίσκονται τα registry dumps και ξεκινάει η ανάλυσή τους σειριακά. Τα αποτελέσματα αποθηκεύονται στο φάκελο “reg-anal”.

4 Εγχειρίδιο χρήστη

Στην ενότητα αυτή παρέχεται ένα πλήρες εγχειρίδιο χρήσης. Θα ξεκινήσουμε, αναφέροντας όλες τις απαραίτητες βιβλιοθήκες για την εκτέλεση του εργαλείου που αναπτύχθηκε, καθώς και τον τρόπο εγκατάστασης τους, όπως επίσης και όλες τις εξαρτήσεις που προκύπτουν από εξωτερικά εργαλεία και βάσεις δεδομένων. Στη συνέχεια θα παρουσιάσουμε όλους τους διακόπτες, μαζί με τα ορίσματα χρήστη που απαιτούνται και τέλος, θα δώσουμε κάποια παραδείγματα χρήσης του εργαλείου.

4.1 Εξαρτήσεις εξωτερικών βιβλιοθηκών

Στην παράγραφο αυτή, αναφέρονται οι εξαρτήσεις του εργαλείου που δημιουργήθηκε, από εξωτερικές βιβλιοθήκες. Γίνετε μια περιγραφή των δυνατοτήτων της κάθε βιβλιοθήκης καθώς και ο σκοπός που εξυπηρετεί στο εργαλείο αυτό.

4.1.1 Yara-python

Η βιβλιοθήκη «yara-python» αποτελεί μια διασύνδεση μεταξύ του Yara και της python [14] και επιτρέπει τη χρήση όλων των λειτουργιών του μέσω της συγκεκριμένης γλώσσας. Με τη χρήση της συγκεκριμένης βιβλιοθήκης, δίνεται η δυνατότητα μεταγλώττισης, αποθήκευσης και φόρτωσης των κανόνων yara αλλά και ο έλεγχος των αρχείων, των συμβολοσειρών και των διεργασιών του λειτουργικού συστήματος.

Στο εργαλείο που δημιουργήθηκε αξιοποιούνται όλες οι δυνατότητες που αναφέραμε και υποστηρίζει, τόσο τη μεταγλώττιση κανόνων όσο και τη χρήση τους για τον εντοπισμό κακόβουλου λογισμικού σε αρχεία και διεργασίες.

4.1.2 Pypiwin32

Η συγκεκριμένη βιβλιοθήκη, επιτρέπει τη χρήση του win32 api διαμέσου της γλώσσας Python και δίνει τη δυνατότητα δημιουργίας και χρήσης αντικειμένων COM (Component Object Model), καθώς και τη χρήση του περιβάλλοντος pythonwin, το οποίο αποτελεί το περιτύλιγμα της βιβλιοθήκης MFC (Microsoft Foundation Class).

Η βιβλιοθήκη αυτή χρησιμοποιείται στο στάδιο ελέγχου των resources των εκτελέσιμων και ειδικότερα για τον έλεγχο ύπαρξης εικονιδίου.

4.1.3 Requests [Security]

Η βιβλιοθήκη αυτή, αποτελεί επέκταση της βιβλιοθήκης «requests» και περιέχει τρία επιπλέον πακέτα (pyOpenSSL, cryptography, idna), τα οποία επιτρέπουν την ασφαλέστερη δημιουργία SSL συνδέσεων. Κάνοντας χρήση της βιβλιοθήκης αυτής, έχουμε τη δυνατότητα να σχηματίζουμε http ερωτήματα και να ανακτούμε τις απαντήσεις αυτόματα, χωρίς να χρειάζεται να ανησυχούμε για το σωστό σχηματισμό τους.

Η συγκεκριμένη βιβλιοθήκη χρησιμοποιείται για την αποστολή των αιτήσεων στο virus total και την «ανάγνωση» των απαντήσεων που λαμβάνουμε με τη χρήση του ενσωματωμένου json decoder που διαθέτει.

4.1.4 Psutil

Η βιβλιοθήκη psutil (process and system utilities), αποτελεί μια βιβλιοθήκη η οποία είναι ανεξάρτητη πλατφόρμας, αφού υποστηρίζει τόσο τα λειτουργικά windows όσο και Linux, osx, Solaris κ.α., και χρησιμοποιείται για την απόκτηση πληροφοριών από διεργασίες που εκτελούνται και χρήσης των πόρων του συστήματος (CPU, μνήμη, εύρος ζώνης δικτύου, χρήση δίσκων) στην Python.

Η βιβλιοθήκη αυτή, βρίσκει χρήση στην παρακολούθηση του συστήματος, στη δημιουργία προφίλ για τον περιορισμό των πόρων των διεργασιών, αλλά και στην διαχείριση των τρεχόντων διεργασιών. Στο εργαλείο που υλοποιήθηκε η συγκεκριμένη βιβλιοθήκη μας δίνει την δυνατότητα να προσπελάσουμε τις διεργασίες και να τις ανιχνεύσουμε για συμβολοσειρές με τη χρήση του yara.

4.1.5 Pefile

Το pefile, είναι ένα Python module που δημιουργήθηκε για την ανάλυση Portable Executable αρχείων [15]. Με τον όρο Portable executable (PE) ονομάζουμε το format που χρησιμοποιείται από εκτελέσιμα αρχεία (exe,dll,fon) σε windows λειτουργικά συστήματα και περικλείει όλες τις απαραίτητες πληροφορίες που χρειάζεται ο loader για να τρέξει τον εκτελέσιμο κώδικα. Το μεγαλύτερο μέρος των πληροφοριών που εμπεριέχονται, τόσο στις επικεφαλίδες όσο και στα υπόλοιπα τμήματα (sections) των αρχείων, είναι προσβάσιμα μέσω του συγκεκριμένου module και εμφανίζονται σαν γνωρίσματα στο συγκεκριμένο στιγμιότυπο του PE αρχείου.

Μερικές από τις δυνατότητες του pefile είναι ο έλεγχος των επικεφαλίδων, η ανάλυση των δεδομένων των τμημάτων του αρχείου (sections) και των ενσωματωμένων δεδομένων του. Μπορεί επίσης, να διαβάσει συμβολοσειρές από τους πόρους του αρχείου και να προειδοποιήσει για ύποπτες ή κακοσχηματισμένες τιμές (malformed values). Έχει τη δυνατότητα να μεταβάλλει κάποια από τα δεδομένα των PE αρχείων, γράφοντας πάνω από τα ήδη υπάρχοντα και τέλος, υποστηρίζει την ανίχνευση packed εφαρμογών με τη χρήση υπογραφών. Η τεχνική αυτή αποτελεί παραδοσιακό τρόπο [10] ανίχνευσης και κατάταξης των packers, crypters και obfuscators, και βασίζεται στην ταύτιση συγκεκριμένων byte που βρίσκονται σε τμήματα ή στην κεφαλίδα των PE αρχείων, με μια βάση που περιέχει όλες τις γνωστές υπογραφές τέτοιων αλγορίθμων.

Είναι προφανές ότι πρόκειται για ένα πολύ ισχυρό module, καθώς χρησιμοποιείται από πολλά γνωστά πρότζεκτ όπως το Virus Total, το Cuckoo, το multiscanner και άλλα. Στο εργαλείο που αναπτύχθηκε χρησιμοποιείται για τον έλεγχο των PE αρχείων όσον αφορά την

εύρεση υπογραφών από γνωστούς packers αλλά και στην ανάκτηση των λεπτομερειών του αρχείου για τον έλεγχο των resources του κάθε εκτελέσιμου.

4.2 Εξαρτήσεις βάσεων δεδομένων

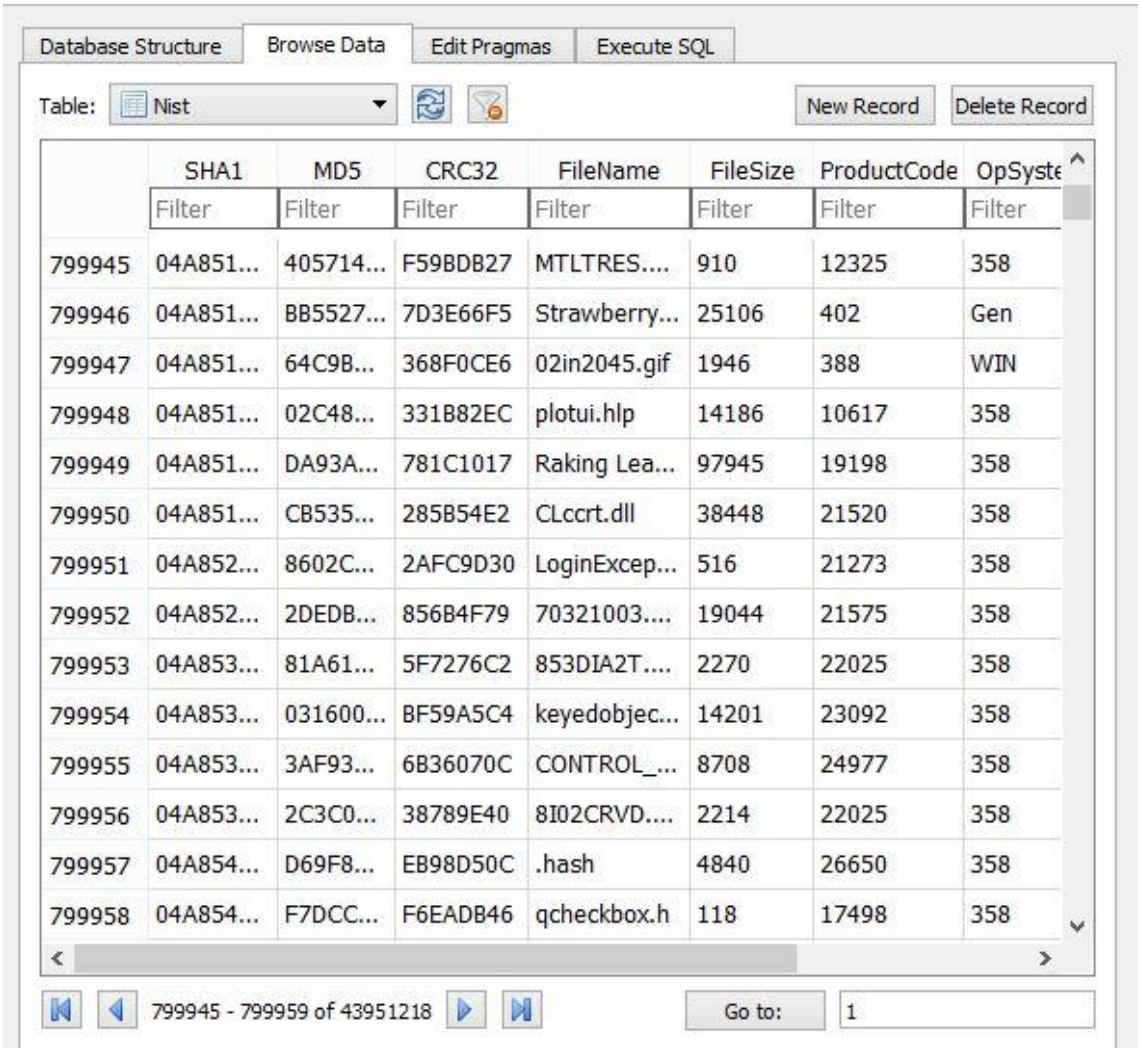
Στην ενότητα αυτή, περιγράφονται οι βάσεις δεδομένων NSRL και υπογραφών packer οι οποίες είναι απαραίτητες για την ορθή λειτουργία του εργαλείου που αναπτύχθηκε. Κάθε συλλογή δεδομένων εξυπηρετεί έναν σημαντικό σκοπό όπως περιγράφεται στις υποενότητες που ακολουθούν.

4.2.1 NSRL

Η NSRL (National Software Reference Library), είναι η εθνική βιβλιοθήκη αναφοράς λογισμικού, η οποία είναι βασισμένη [16] στο Εθνικό Ινστιτούτο προτύπων και τεχνολογίας (NIST), μια υπηρεσία του υπουργείου εμπορίου των ΗΠΑ και διαθέτει τρία στοιχεία:

1. Μια μεγάλη συλλογή πακέτων λογισμικού.
2. Μια βάση δεδομένων, η οποία περιέχει αναλυτικές πληροφορίες ή μεταδεδομένα, για αρχεία τα οποία συνθέτουν τα λογισμικά αυτά.
3. Ένα δημόσιο σύνολο δεδομένων, το NSRL RDS (Reference Data Set), το οποίο εμπεριέχει ένα υποσύνολο από τα μεταδεδομένα τα οποία βρίσκονται στη βάση για κάθε αρχείο της το οποίο ανανεώνεται κάθε τρεις μήνες.

Το σύνολο δεδομένων αυτό, περιέχει όπως φαίνεται και στην εικόνα που ακολουθεί, μεταδεδομένα των αρχείων τα οποία χρησιμοποιούνται για να προσδιορίσουν τα αρχεία και την προέλευσή τους.



	SHA1	MD5	CRC32	FileName	FileSize	ProductCode	OpSyste
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
799945	04A851...	405714...	F59BDB27	MTLTRES....	910	12325	358
799946	04A851...	BB5527...	7D3E66F5	Strawberry...	25106	402	Gen
799947	04A851...	64C9B...	368F0CE6	02in2045.gif	1946	388	WIN
799948	04A851...	02C48...	331B82EC	plotui.hlp	14186	10617	358
799949	04A851...	DA93A...	781C1017	Raking Lea...	97945	19198	358
799950	04A851...	CB535...	285B54E2	CLccrt.dll	38448	21520	358
799951	04A852...	8602C...	2AFC9D30	LoginExcep...	516	21273	358
799952	04A852...	2DEDB...	856B4F79	70321003....	19044	21575	358
799953	04A853...	81A61...	5F7276C2	853DIA2T....	2270	22025	358
799954	04A853...	031600...	BF59A5C4	keyedobjec...	14201	23092	358
799955	04A853...	3AF93...	6B36070C	CONTROL_...	8708	24977	358
799956	04A853...	2C3C0...	38789E40	8I02CRVD....	2214	22025	358
799957	04A854...	D69F8...	EB98D50C	.hash	4840	26650	358
799958	04A854...	F7DCC...	F6EADB46	qcheckbox.h	118	17498	358

Εικόνα 6: Στιγμιότυπο από τον πίνακα NSRL

Για κάθε αρχείο στη βάση της NSRL, δημοσιεύονται τα ακόλουθα δεδομένα:

- Τιμές συναρτήσεων κατακερματισμού «υποστηρίζονται οι συναρτήσεις MD5 και SHA-1» των περιεχομένων των αρχείων. Η τιμή αυτή χρησιμοποιείται για τον μοναδικό προσδιορισμό του κάθε αρχείου ακόμα και σε περίπτωση που έχει μετονομαστεί.
- Δεδομένα για την προέλευση του αρχείου, όπως το λειτουργικό από το οποίο προέρχονται και τον φορέα ανάπτυξής του.
- Διάφορα άλλα δεδομένα του αρχείου, όπως το αρχικό του όνομα και το μέγεθός του.

Η χρησιμότητα της βάσης, έγκειται στην γρήγορη ταυτοποίηση των αρχείων ενός υπολογιστή με βάση μόνο το περιεχόμενό τους. Στις περισσότερες περιπτώσεις, χρησιμοποιείται σε υποθέσεις ψηφιακής σήμανσης με στόχο τη μείωση του όγκου δεδομένων που πρέπει να ελέγξει ο ερευνητής χειροκίνητα και επιτρέπει την εστίαση σε αρχεία που παρουσιάζουν πραγματικό ενδιαφέρον.

Για να τονίσουμε την σημαντικότητα αυτού του εργαλείου αξίζει να επισημάνουμε ότι η βάση αυτή χρησιμοποιείται κυρίως από ειδικούς της ψηφιακής σήμανσης κατά την έρευνά τους σε υπολογιστικά συστήματα αλλά και από κρατικούς οργανισμούς και επιχειρήσεις στη διάρκεια καθημερινών ελέγχων των συστημάτων τους.

Την ίδια λειτουργικότητα εξυπηρετεί και στην παρούσα εργασία καθώς χρησιμοποιείται για το «διαχωρισμό» τον αρχείων σε «καλόβουλα» και «πιθανώς κακόβουλα» μειώνοντας τον όγκο δεδομένων των αρχείων που πρέπει να αναλύσουμε περαιτέρω.

4.2.2 Packer database

Η συγκεκριμένη βάση δεδομένων περιέχει υπογραφές και κανόνες yara βάση των οποίων καθίσταται εφικτός ο εντοπισμός γνωστών packer όπως μπορούμε να δούμε και στην εικόνα που ακολουθεί.

```
[Alloy 4.x -> PGWare LLC]
signature = 9C 60 E8 02 00 00 00 33 C0 8B C4 83 C0 04 93 8B E3 8B 5B FC 81 EB 07 30 40 00 87 DD 6A 04 68 00 10 00 00 68
2E 33 40 00 83 BD E8 32 40 00 01 74 0D 83 BD E4 32 40 00 01 74 2A 8B F8 EB 3E 68 D8 01 00 00 50 FF 95 CC 33 40 00 50 8D
EB 0C 68 D8 01 00 00 50 FF 95 C0 33 40 00 8B BD 2E 33 40 00 03 F8 C6 07 5C 47 8D B5 00 33 40 00 AC 0A C0 74 03 AA EB F8
68 00 00 00 00 80 FF B5 2E 33 40 00 FF 95 B4 33 40 00 83 F8 FF 74 57 89 85 32 33 40 00 8D 85 56 33 40 00 8D 9D 5E 33 40 00
B5 32 33 40 00 FF 95 B8 33 40 00 8B 85
ep_only = true

[Alloy v1.x.2000]
signature = 9C 60 E8 02 ?? ?? ?? 33 C0 8B C4 83 C0 04 93 8B E3 8B 5B FC 81 EB 07 20 40 ?? 87 DD 6A 04 68 ?? 10 ?? ?? 68
ep_only = true

[Aluwain v8.09]
signature = 8B EC 1E E8 ?? ?? 9D 5E
ep_only = true

[Amiga AIFF 8SFX Audio file]
signature = 46 4F 52 4D ?? ?? ?? ?? 38 53 56 58 56 48 44 52
ep_only = false

[Amiga IFF/ILBM Graphics format]
signature = 46 4F 52 4D ?? ?? ?? ?? 49 4C 42 4D 42 4D 48 44
ep_only = false
```

Εικόνα 7: Στιγμιότυπο από το αρχείο υπογραφών των packer

Υπάρχουν διαθέσιμες, σύμφωνα με το [17], αρκετές βάσεις με υπογραφές οι οποίες δημοσιεύονται από διαφορετικούς οργανισμούς και ανανεώνονται συνεχώς προκειμένου να περιλαμβάνουν τις πιο πρόσφατες υπογραφές. Επιλέξαμε να χρησιμοποιήσουμε τη βάση που έχει δημοσιευθεί από τον οργανισμό Sans, για την ανίχνευση χρήσης packer στη εργαλείο που δημιουργήθηκε.

4.3 Εξαρτήσεις εργαλείων

Στο σημείο αυτό, θα μιλήσουμε για όλα τα εργαλεία που χρησιμοποιούνται και το σκοπό που εξυπηρετούν στην υλοποίηση του προγράμματός μας. Θα ξεκινήσουμε με τη γλώσσα στην οποία αναπτύχθηκε το εργαλείο και στη συνέχεια θα αναλύσουμε όλα τα εργαλεία ανοιχτού κώδικα που επιλέξαμε

4.3.1 Python

Η python αναπτύχθηκε το 1990 από τον Guido van Rossum σε μια προσπάθεια να δημιουργήσει τον απόγονο της γλώσσας ABC ο οποίος θα είχε τη δυνατότητα να διαχειρίζεται εξαιρέσεις «exception handling» και να αλληλεπιδρά με το λειτουργικό σύστημα «Amoeba»

Το πνεύμα της συγκεκριμένης γλώσσας συνοψίζεται από τον Tim Peters [18] με πολύ εύστοχο τρόπο:

«Το όμορφο είναι καλύτερο από το άσχημο
Το σαφές είναι καλύτερο από το ασαφές
Το απλό είναι καλύτερο από το σύνθετο
Το σύνθετο είναι καλύτερο από το περίπλοκο»

Η python είναι μιας γενικής χρήσης, υψηλού επιπέδου, γλώσσα προγραμματισμού [19], της οποίας η φιλοσοφία σχεδίασης τονίζει τον ευανάγνωστο κώδικα. Στόχος της Python είναι να συνδυάσει αξιοσημείωτη δύναμη, με ξεκάθαρη σύνταξη και υποστηρίζει διαφορετικές προγραμματιστικές ιδεολογικές δομές, όπως αντικειμενοστρέφια (object-oriented), προστακτική

γλώσσα (imperative) αλλά και συναρτησιακή (functional). Διαθέτει ένα πλήρως δυναμικό «type system», και αυτοματοποιημένη διαχείριση μνήμης.

Ο κώδικας στη συγκεκριμένη γλώσσα είναι αποθηκευμένος σε αρχεία κειμένου με την κατάληξη «.py» και μεταγλωττίζεται σε δυαδικό κώδικα ο οποίος είναι ανεξάρτητος πλατφόρμας (machine independent) και αποθηκεύεται σε αρχεία με κατάληξη «.pyc» τα οποία με τη σειρά τους μπορούν να εισαχθούν και να εκτελεστούν με ταχύτητα. Ο κώδικας μεταγλωττίζεται ξανά μόνο όταν είναι απαραίτητο.

Όλες οι γλώσσες υποστηρίζουν τους βασικούς τύπους, όπως συμβολοσειρές, ακεραίους, floats η python όμως διαθέτει ενσωματωμένους τύπους υψηλότερου επιπέδου όπως λεξικά και λίστες και διεργασίες που εργάζονται πάνω σε αυτούς τους τύπους με τον πλέον αποδοτικό τρόπο. Επιπλέον η διαδραστική λειτουργία της, επιτρέπει τη δοκιμή κάθε συνάρτησης και μεθόδου καθώς γράφεται, κάτι το οποίο προωθεί τη δοκιμή ιδεών στον κώδικα γρήγορα και χωρίς υπολογιστικό κόστος.

Το συντακτικό της γλώσσας είναι ξεκάθαρο, δεν διαθέτει ειδικούς χαρακτήρες και έτσι είναι κατανοητή στην ανάγνωση και γι αυτό το λόγο προτείνεται ως μια από τις ιδανικές γλώσσες για να ξεκινήσει κάποιος. Έχει πάνω από 200 βιβλιοθήκες ενσωματωμένες στη βασική εγκατάστασή οι οποίες περιλαμβάνουν από συναρτήσεις λειτουργικού συστήματος και δομών δεδομένων έως ολοκληρωμένες αναπτύξεις web server. Τέλος, διαθέτει μία τεράστια κοινότητα η οποία δημιουργεί συνεχώς και υποστηρίζει νέα modules και apis.

Η επιλογή της συγκεκριμένης γλώσσας στην εκπόνηση της εργασίας είναι αντικείμενο όλων αυτών των μοναδικών χαρακτηριστικών που αναφέραμε. Ειδικότερα όμως, στην ικανότητα της γλώσσας να «διαβάζει» αρχεία σαν συμβολοσειρές και να τα αναλύει με ταχύτατο τρόπο και του api που μας επιτρέπει να εργαζόμαστε πάνω σε filepath και αρχεία και επιτρέπει την αναδρομική πλοήγηση σε φακέλους του υπολογιστή.

4.3.2 Virus Total

Το VirusTotal αποτελεί μια δωρεάν πλατφόρμα, η οποία συλλέγει πληροφορίες για αρχεία και διευθύνσεις url, με σκοπό να εντοπίσει ιούς, trojans, worms και γενικότερα οποιοδήποτε είδους κακόβουλου λογισμικού, χρησιμοποιώντας πληροφορίες από διαφορετικά αντιικά προγράμματα και σαρωτές διευθύνσεων url.

Είναι σημαντικό να αναφερθεί, ότι τα αποτελέσματα του VirusTotal προκύπτουν από αναλύσεις πάνω από πενήντα διαφορετικών κορυφαίων αντιικών προγραμμάτων, όπως είναι το Kaspersky Lab της εταιρίας Kaspersky, το Intel Security της McAfee, το webroot και άλλα. Όσον αφορά τον έλεγχο διευθύνσεων url και domain, υπάρχει το bitdefender, το Google safebrowsing καθώς και το avira checkurl. Στο παράρτημα 2 δίνεται αναλυτική λίστα όλων των εταιριών που συμβάλουν στο VirusTotal.

Τα κύρια χαρακτηριστικά του VirusTotal περιλαμβάνουν τη δωρεάν διαθεσιμότητά του στους χρήστες -αρκεί να μη χρησιμοποιηθεί για εμπορικούς σκοπούς- τη χρήση μίας μεγάλης ποικιλίας αντιικών προγραμμάτων καθώς και εργαλεία ανάλυσης αρχείων και διευθύνσεων url. Αξίζει να σημειωθεί, πως ανανεώνεται συνεχώς προκειμένου να περιέχει τις πιο ενημερωμένες υπογραφές κακόβουλου λογισμικού και δίνει μια λεπτομερή εικόνα για την ακριβή ταυτότητα ενός κακόβουλου λογισμικού.

Όλα αυτά τα εργαλεία εξυπηρετούν διαφορετικούς σκοπούς οι οποίοι κυμαίνονται από το να παρέχουν δομικές πληροφορίες για μορφές αρχείων Portable Executables της Microsoft έως και να προσδιορίζουν αν τα αρχεία είναι ψηφιακά υπογεγραμμένα ή όχι.

Στο VirusTotal μπορούν να υποβληθούν για ανάλυση εκτός από αρχεία και url διευθύνσεις, διευθύνσεις ip καθώς και τιμές συναρτήσεων κατακερματισμού (hash values) αφού υποστηρίζει τις συναρτήσεις md5, sha1, sha256.

Επίσης προσφέρει διαφορετικούς τρόπους αλληλεπίδρασης με την web υπηρεσία του, οπότε ο κάθε χρήστης μπορεί να επιλέξει μεταξύ των api, email, web ή ακόμα και του VirusTotal uploader, μια εφαρμογή που έχει δημιουργηθεί προκειμένου ο κάθε χρήστης να μπορεί να ανεβάσει αρχεία προς έλεγχο σε αυτό.

Το api του VirusTotal μας επιτρέπει να ανεβάζουμε και να σκανάρουμε αρχεία και url και να έχουμε πρόσβαση στις αναφορές που προκύπτουν. Για τη χρήση του το μόνο που χρειάζεται είναι ένα κλειδί το οποίο δίνεται με την εγγραφή του κάθε χρήστη στην κοινότητα του VirusTotal. Το κλειδί αυτό, μπορεί να είναι είτε δημόσιο (public key), είτε ιδιωτικό (private key), ανάλογα με την προτίμηση του ίδιου του χρήστη.

Όσον αφορά το δημόσιο κλειδί, αυτό δίνεται δωρεάν, αλλά περιλαμβάνει περιορισμούς στη χρήση του, σε αντίθεση με το ιδιωτικό. Ενδεικτικά αναφέρουμε ότι με το δημόσιο κλειδί τα αιτήματα του χρήστη προς το VirusTotal περιορίζονται σε τέσσερα ανά λεπτό, σε αντίθεση με το ιδιωτικό, στο οποίο δίνεται η δυνατότητα επιλογής του ρυθμού των αιτήσεων. Επιπλέον, με το δημόσιο κλειδί υπάρχει περιορισμός στο μέγεθος των αρχείων που υποβάλλονται, με το μέγιστο μέγεθος ενός αρχείου να φτάνει μέχρι τα 32MB. Το ιδιωτικό κλειδί έχει κάποια επιπλέον πλεονεκτήματα, όπως λεπτομερέστερες αναφορές οι οποίες περιλαμβάνουν μεταδεδομένα του αρχείου καθώς και μεταδεδομένα του VirusTotal όπως ημερομηνία υποβολής, λοιπά αρχεία που ανέβηκαν με ένα συγκεκριμένο αρχείο, αποτελέσματα του εργαλείου exifTool κ.α. Το πλέον σημαντικό πλεονέκτημα του ιδιωτικού κλειδιού είναι ότι εγγυάται τη διαθεσιμότητα και την αμεσότητα των δεδομένων.

4.3.3 Yara

Το Yara αποτελεί εργαλείο που έχει σκοπό να βοηθήσει τους ερευνητές να αναγνωρίσουν και να κατατάξουν το κακόβουλο λογισμικό. Το εργαλείο αυτό μπορεί να λειτουργήσει σε διαφορετικά λογισμικά (cross-platform) και μπορεί να εκτελεστεί είτε μέσω της δικής του διεπαφής γραμμής εντολών (CLI), είτε με τη βοήθεια μικροεντολών python (scripts).

Το yara επιτρέπει την ανίχνευση malware, μέσω του εντοπισμού μοτίβων στις υπογραφές κακόβουλων αρχείων με τη χρήση κάποιων κανόνων (yara rules), αναζητά δηλαδή μέσα στα binary αρχεία συγκεκριμένες συμβολοσειρές (strings) που ταιριάζουν με αυτές των κακόβουλων προγραμμάτων. Κάθε κανόνας ξεκινάει με τον ορισμό του ονόματός του και αποτελείται από τουλάχιστον δύο μέρη, τον ορισμό των συμβολοσειρών και τις λογικές συνθήκες.

Οι συμβολοσειρές μπορούν να είναι είτε δεκαεξαδικές τιμές, είτε κείμενο, ή και regular expressions και οι συνθήκες είναι αυτές που καθορίζουν το πότε ένα αρχείο ικανοποιεί τον συγκεκριμένο κανόνα. Στο παράδειγμα που ακολουθεί βλέπουμε ότι ο κανόνας ικανοποιείται είτε όταν βρεθεί η συμβολοσειρά είτε όταν βρεθεί το κείμενο.

```
rule example1
{
    strings:
        $hex_string = {F4 23 34 FB }
        $text_string = "foo"
    condition:
         $hex_string or  $text_string
}
```

Εικόνα 8: Κανόνας Yara

Οι δεκαεξαδικές συμβολοσειρές κάνουν χρήση τριών ειδικών χαρακτήρων, με σκοπό να καταστήσουν τους κανόνες πιο ευέλικτους. Ο πρώτος χαρακτήρας είναι το ερωτηματικό «?» ο οποίος χρησιμοποιείται όταν ένα nibble ή ένα byte μπορεί να πάρει διάφορες τιμές «$hex_string = { F4 23 ?4 FB ??}». Ο δεύτερος μας δίνει την δυνατότητα να κάνουμε άλματα στην συμβολοσειρά και έχει την μορφή [x-y] (όπου x,y δυο ακέραιοι αριθμοί με την προϋπόθεση ότι το x είναι πάντα μεγαλύτερο ή ίσο του y). Χρησιμοποιείται όταν ανάμεσα στη συμβολοσειρά υπάρχουν χαρακτήρες μεταβλητού μεγέθους οι οποίοι δεν έχουν αξία για τον κανόνα μας, όπως

για παράδειγμα η συμβολοσειρά «$hex_string = { F4 23 [2-10] FB }» η οποία μπορεί να έχει μέγεθος από 5 έως 13 byte αλλά για να υπάρξει ταύτιση μόνο τα δύο πρώτα και το τελευταίο byte έχουν αξία. Τέλος έχουμε τον χαρακτήρα «|» ο οποίος χρησιμοποιείται όταν υπάρχουν εναλλακτικές τιμές στα byte μιας συμβολοσειράς, για παράδειγμα «$hex_string = {F4 23 (34|B4 64) FB }». Ο κανόνας αυτός θα επαληθευτεί είτε όταν η συμβολοσειρά πάρει την τιμή “F42334FB”, είτε όταν πάρει την τιμή «F423B464FB».

Το κείμενο είναι μια ASCII συμβολοσειρά η οποία αναζητείται στο αρχείο προκειμένου να ικανοποιηθεί ο κανόνας. Υπάρχουν διάφοροι τροποποιητές που μπορούν να χρησιμοποιηθούν για να κάνουν τον κανόνα πιο αποδοτικό, όπως είναι ο nocase ο οποίος καθιστά τη συμβολοσειρά ανεξάρτητη πεζών-κεφαλαίων, ο «wide» ο οποίος ψάχνει για συμβολοσειρές που έχουν κωδικοποιηθεί με δυο byte ανά χαρακτήρα και ο «fullword» που για την ταύτιση του κανόνα απαιτεί η λέξη να εμφανίζεται και να είναι οριοθετημένη από μη αλφαριθμητικές τιμές.

Τέλος τα regular expressions αποτελούν τη σημαντικότερη λειτουργία του yara, καθώς μας δίνουν τη δυνατότητα να αποκτήσουμε μεγαλύτερο έλεγχο στους κανόνες, δηλώνοντας ένα μοτίβο κειμένου που πρέπει να βρεθεί σε ένα αρχείο προκειμένου να ταυτοποιηθεί ο εκάστοτε κανόνας.

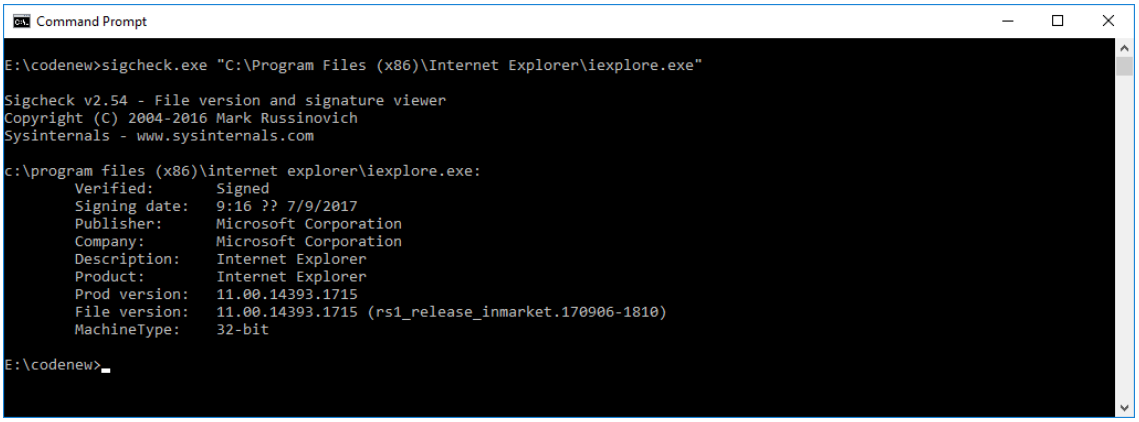
Όπως προαναφέρθηκε, οι συνθήκες είναι αυτές που καθορίζουν πότε ικανοποιείται ο εκάστοτε κανόνας. Υποστηρίζουν στον έλεγχο των συνθηκών, τους βασικούς λογικούς τελεστές (and, or, not), τους συνήθεις σχεσιακούς τελεστές (< =, > =, <, >, ==,!=), τους αριθμητικούς τελεστές (+, -, *, /, %) και τους τελεστές δυαδικών ψηφίων (&, |, <<, >>,~, ^). Στις συνθήκες μπορούμε ακόμα να χρησιμοποιήσουμε διάφορες παραμέτρους όπως το μέγεθος αρχείου, το offset της συμβολοσειράς, το πλήθος εμφάνισης της συμβολοσειράς σε ένα αρχείο και τον αριθμό των συμβολοσειρών που πρέπει να βρεθούν για να επαληθευθεί ο κανόνας. Επιτρέπει επίσης την προσπέλαση δεδομένων σε αρχεία και εικονικές θέσεις μνήμης με την παροχή συγκεκριμένου offset. Τέλος η συνθήκη μπορεί να δεχτεί σαν όρισμα, ένα διαφορετικό κανόνα που έχει ήδη δηλωθεί. Με αυτό τον τρόπο μπορούμε να δημιουργήσουμε κανόνες που εξαρτώνται ο ένας από τον άλλο.

Το yara αποτελεί ένα πολύ ισχυρό εργαλείο και χρησιμοποιείται από πολλούς οργανισμούς [20] για ανίχνευση κακόβουλου λογισμικού. Υποστηρίζεται από μια μεγάλη κοινότητα η οποία έχει αναλάβει να δημιουργεί και να ανανεώνει τους κανόνες [21]. Αυτή η τεχνική ανίχνευσης malware είναι από τις παλαιότερες και πλέον θεωρείται αρκετά ξεπερασμένη, κυρίως λόγω του ότι πλέον οι δημιουργοί των malware, χρησιμοποιούν εφαρμογές όπως «packers»,«crypters» και «obfuscators» για να αλλοιώσουν και να μεταβάλλουν τις συμβολοσειρές στο κακόβουλο λογισμικό έτσι ώστε να μην ταιριάζουν με τις ήδη γνωστές υπογραφές.

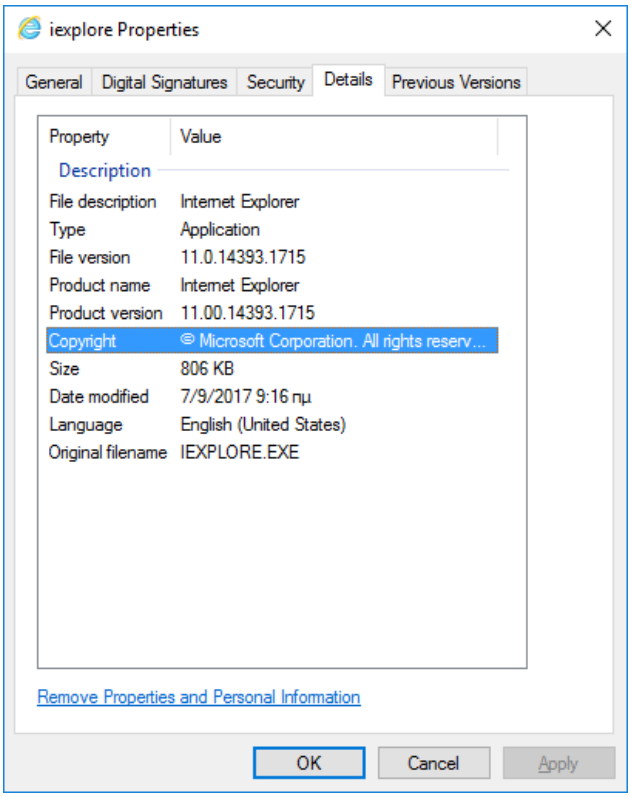
4.3.4 Sigcheck

Το συγκεκριμένο εργαλείο γραμμής εντολών (CLI) αναπτύχθηκε από τη Sysinternals με σκοπό την εκτέλεση ενεργειών που αφορούν την ασφάλεια σε ένα αρχείο. Κύριος σκοπός του, είναι να βεβαιώσει αν κάποιο αρχείο είναι ψηφιακά υπογεγραμμένο με έγκυρο πιστοποιητικό. Είναι επίσης ικανό να εμφανίσει αναλυτικές πληροφορίες για τον φορέα που έχει υπογράψει το αρχείο όπως επίσης και χρονικές σφραγίδες καθώς και λεπτομέρειες της έκδοσης του αρχείου. Μπορεί να υπολογίσει τις τιμές των συναρτήσεων κατακερματισμού και να εξάγει τα αποτελέσματα αυτά σε ένα αρχείο.

Ο έλεγχος της ψηφιακής υπογραφής διαβεβαιώνει της ακεραιότητα και την αυθεντικότητα του αρχείου, καθώς εγγυάται την προέλευση του αρχείου από τον φορέα που το υπέγραψε. Οι πληροφορίες που επιστρέφονται από το Sigcheck είναι οι ακόλουθες (**Εικόνα 9**)



Εικόνα 9: Στιγμιότυπο χρήσης του Sigcheck



Εικόνα 10: Λεπτομέρειες ενός εκτελέσιμου

1. Verified: Σε περίπτωση που το αρχείο είναι υπογεγραμμένο από μια αρχή που έχει δηλωθεί σαν έμπιστη και δεν έχει μεταβληθεί έκτοτε τότε το πεδίο αυτό παίρνει την τιμή “Signed”, διαφορετικά αν το αρχείο δεν είναι υπογεγραμμένο ή έχει περάσει η έγκυρη ημερομηνία ή το αρχείο έχει μεταβληθεί η τιμή γίνεται “Unsigned”
2. Signing data: Εμφανίζει την ημερομηνία υπογραφής του αρχείου
3. Publisher: Επιστρέφει το εταιρικό όνομα που εμφανίζεται στις ιδιότητες του αρχείου
4. Description: Την περιγραφή που δηλώνεται στις ιδιότητες του αρχείου
5. Product: Το όνομα του προϊόντος όπως εμφανίζεται στις ιδιότητες του αρχείου
6. Version: Την έκδοση του προγράμματος όπως εμφανίζεται στις ιδιότητες
7. File version: Ομοίως επιστρέφει την έκδοση του προγράμματος

8. Machine type: Την αρχιτεκτονική του αρχείου

Στο παράρτημα 1 αναφέρονται αναλυτικά όλοι οι διακόπτες και δυνατότητες του προγράμματος.

4.3.5 DumpIt

Κύριους στόχους της ψηφιακής σήμανσης, αποτελούν η συλλογή, η διατήρηση και η ανάλυση των δεδομένων ενός υπολογιστή πάντα με τέτοιο τρόπο ο οποίος θα εγγυάται την ακεραιότητα των δεδομένων που αποκτήθηκαν [22] . Τα δεδομένα αυτά κατατάσσονται σε δύο κατηγορίες, τα πτητικά και τα σταθερά δεδομένα. Σταθερά θεωρούνται τα δεδομένα που είναι αποθηκευμένα στον σκληρό δίσκο ή σε οποιοδήποτε άλλο μέσο και διατηρούνται όταν κλείσει ο υπολογιστής. Σε αντίθεση, ο όρος πτητικά δεδομένα αναφέρεται σε εκείνα τα δεδομένα ενός υπολογιστή τα οποία διαγράφονται όταν τερματιστεί η λειτουργία του ή μεταβάλλονται συνεχώς όσο αυτός λειτουργεί. Στη συνέχεια θα περιγράψουμε τη διαδικασία συλλογής και ανάλυσης δύο εκ των σημαντικότερων πτητικών δεδομένων, της μνήμης και της registry.

Στην ψηφιακή σήμανση, τα πτητικά δεδομένα και ειδικότερα η μνήμη αποτελεί ένα πολύτιμο μέσο στην συλλογή αποδείξεων για μια υπόθεση. Παρόλο που οι παραδοσιακές τεχνικές εστιάζουν στην ανάλυση μη πτητικών δεδομένων από συσκευές όπως είναι οι σκληροί δίσκοι, η μνήμη μπορεί να περιέχει πολύ σημαντικές πληροφορίες για την τρέχουσα κατάσταση ενός υπολογιστή. Τρέχουσες διεργασίες, ανοιχτές πόρτες και συνδέσεις, βιβλιοθήκες που έχουν φορτωθεί και ανοιχτά αρχεία είναι κάποια από τα πολλά στοιχεία που μπορούν να ανασυρθούν από τη μνήμη ενός μηχανήματος. Αν και η σημαντικότητα της απόκτησης και ανάλυσης της μνήμης είναι προφανής, συχνά παρουσιάζονται πολλά προβλήματα [20].

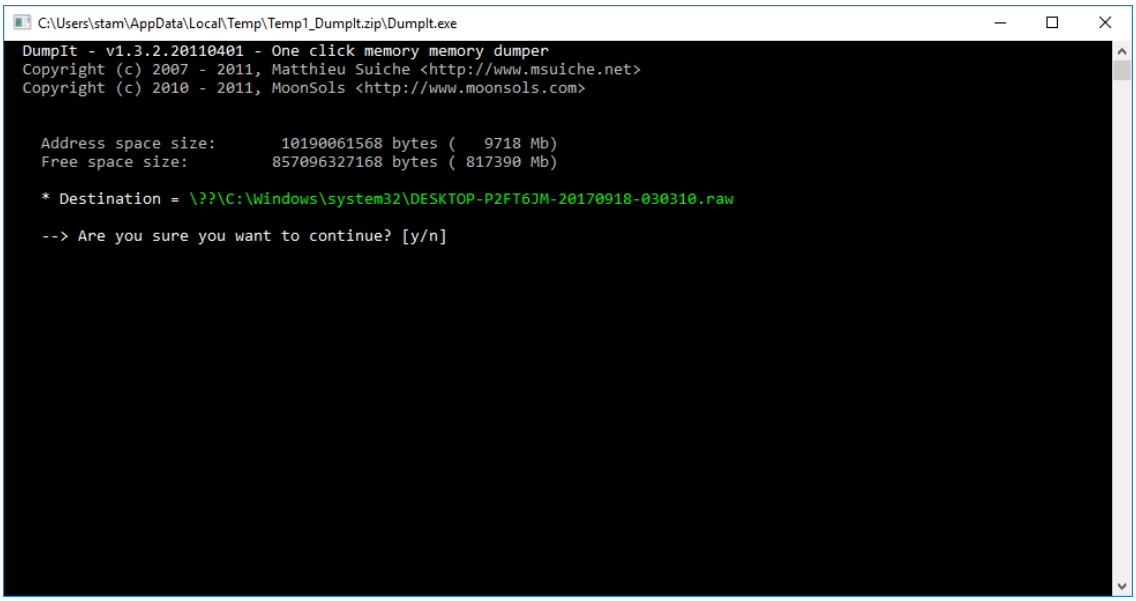
Αρχικά η μεταβατική φύση της μνήμης καθιστά πολύ δύσκολη τη διαδικασία απόκτησής της, χωρίς την αλλοίωση των υπαρχόντων δεδομένων. Όσο περισσότερη ώρα το σύστημα λειτουργεί, τόσο τα δεδομένα που υπάρχουν στη μνήμη μεταβάλλονται, καταστρέφοντας πιθανές αποδείξεις. Παρ' όλα αυτά, κάθε προσπάθεια για πάγωμα του συστήματος μπορεί να οδηγήσει σε απώλεια ή καταστροφή μεγάλου μέρους των δεδομένων. Άλλο ένα σημαντικό πρόβλημα έγκειται στην πολυπλοκότητα των δεδομένων της μνήμης, καθώς κάθε θέση μνήμης μπορεί να περιέχει διαφορετικά δεδομένα από διαφορετικά προγράμματα. Σε αντίθεση με τους σκληρούς δίσκους στους οποίους τα δεδομένα έχουν μια προκαθορισμένη μορφή, η δομή των δεδομένων της μνήμης εξαρτάται από πολλούς διαφορετικούς παράγοντες.

Η απόκτηση ενός αντιγράφου της μνήμης Ram περιλαμβάνει την αντιγραφή και διατήρηση όλων των πτητικών δεδομένων της μνήμης σε ένα χώρο αποθήκευσης, διατηρώντας όμως την εικόνα της μνήμης, όσο το δυνατόν αναλλοίωτη. Ανάλογα με την κατάσταση του υπολογιστή μπορούμε να εφαρμόσουμε διαφορετικές τεχνικές απόκτησης του αντιγράφου της μνήμης. Αν ο υπολογιστής είναι ανοιχτός και έχουμε δικαιώματα διαχειριστή, μπορούμε να εκτελέσουμε το κατάλληλο λογισμικό προκειμένου να αποκτήσουμε το αντίγραφο της μνήμης. Σε περίπτωση όμως, που δεν έχουμε τα απαραίτητα προνόμια ενδείκνυται η εγκατάσταση και χρήση υλικού όπως το Firewire ή το Tribble Card για τη συλλογή της μνήμης. Αν ο υπολογιστής δεν είναι σε λειτουργία, μπορούμε να ανατρέξουμε σε page files, hibernation files και crush dumps, ενώ αν ο πρόκειται για εικονικό μηχάνημα έχουμε τη δυνατότητα να παγώσουμε, να κλωνοποιήσουμε ή και να κρατήσουμε ένα στιγμιότυπο της εικόνας του μηχανήματος για ανάλυση [21]. Κάθε τεχνική παρουσιάζει πλεονεκτήματα και μειονεκτήματα. Στην συλλογή της μνήμης με τη βοήθεια υλικού το κύριο μειονεκτήματα εστιάζεται στο αυξημένο κόστος του εξοπλισμού που είναι απαραίτητος. Παρ' όλα αυτά, το αντίγραφο που θα πάρουμε, θεωρείται πιο αξιόπιστο, καθώς δεν εξαρτάται καθόλου από το λογισμικό, το οποίο μπορεί να έχει αλλοιωθεί από τον επιτιθέμενο. Στην άλλη πλευρά, με τη χρήση λογισμικού η απόκτηση της μνήμης γίνεται πολύ εύκολα καθώς υπάρχει πληθώρα δωρεάν εργαλείων γι αυτό το σκοπό, όμως κατά την εκτέλεση οποιουδήποτε εργαλείου μεταβάλλεται η μνήμη, βάζοντας έτσι σε κίνδυνο σημαντικά δεδομένα, τα οποία μπορεί να χαθούν.

Στο σημείο αυτό, παρατίθεται μια λίστα με τα πιο γνωστά προγράμματα απόκτησης μνήμης:

- KnTTools
- Mandiant Memoryze

- FastDump
- FTK Imager
- EnCase
- Live RAM Capturer
- Winpmem
- MemDump
- DumpIt



Εικόνα 11: Στιγμιότυπο χρήσης του DumpIt

Για τη συλλογή της μνήμης επιλέχθηκε το εργαλείο DumpIt (εικόνα 1), ένα εργαλείο το οποίο προέρχεται από τη μίξη των win32dd και win64dd, σε ένα εκτελέσιμο αρχείο το οποίο προσφέρεται δωρεάν (έκδοση 1.3). Το εργαλείο αυτό, παρουσιάζεται απλοϊκό στη χρήση του, αφού αρκεί ένα διπλό κλικ σε αυτό, ώστε να συλλεχθεί η μνήμη. Δεν απαιτεί εγκατάσταση και μπορεί να εκτελεστεί ακόμα και από αφαιρούμενο αποθηκευτικό μέσο.

Η εικόνα της μνήμης αποθηκεύεται αυτόματα στον ίδιο φάκελο από τον οποίο εκτελείται το εργαλείο, υποστηρίζει 32-bit και 64-bit λειτουργικά Windows και είναι συμβατό με τις εκδόσεις XP, 2003, 2008 R2,7 και 8. Τέλος το DumpIt αποθηκεύει τη μνήμη σε ένα τύπο αρχείου raw ο οποίος αναγνωρίζεται από τα περισσότερα εργαλεία ανάλυσης της μνήμης.

4.3.6 Volatility

Η ανάλυση της μνήμης είναι από τα πλέον σημαντικότερα όπλα που έχει ένας ερευνητής στη διάθεση του και αυτό γιατί στη μνήμη μπορεί να βρίσκονται πληροφορίες, που μπορούν να τον οδηγήσουν στη λύση μιας υπόθεσης όπως τρέχουσες συνδέσεις δικτύου, κωδικούς του χρήστη, διεργασίες που εκτελούνται, κλειδιά κρυπτογράφησης, ενδείξεις rootkit, ανοιχτά αρχεία και πολλά ακόμα δεδομένα [23].

1. Διεργασίες: υπάρχουν διαφόρων ειδών διεργασίες που μπορούν να εντοπιστούν στη μνήμη όπως αυτές οι οποίες ήταν ενεργές καθώς και κρυφές διεργασίες. Μπορούν επίσης, να βρεθούν και διεργασίες που είχαν τερματιστεί πρόσφατα αρκεί ο χώρος που καταλάμβαναν στη μνήμη να μην έχει δεσμευτεί από κάποιο άλλο πρόγραμμα.
2. Αρχεία και Registry handles: Όλα τα αρχεία και τα registry handles που χρησιμοποιούνται από κάποια διεργασία μένουν επίσης στη μνήμη. Έχοντας πρόσβαση σε αυτά τα αρχεία μπορεί να οδηγήσουν τον ερευνητή στον εντοπισμό της θέσης του κακόβουλου λογισμικού στον δίσκο, τις κλήσεις του στο windows api καθώς και το που αποθηκεύει τα αποτελέσματά του.

3. Πληροφορίες δικτύου: Μπορούν να εντοπιστούν πληροφορίες για ενεργές συνδέσεις και ανοιχτές πόρτες σε κατάσταση listening. Οι πληροφορίες δικτύου είναι από τις σημαντικότερες πληροφορίες που μπορούν να αποκτηθούν καθώς είναι πιθανόν να εντοπίσουμε μέχρι και την διεύθυνση δικτύου του επιτιθέμενου.
4. Κωδικοί και κλειδιά κρυπτογράφησης: Οι κωδικοί κατά κανόνα δεν αποθηκεύονται στον δίσκο χωρίς κάποια μέτρα προστασίας παρ' όλα αυτά μένουν στη μνήμη όταν πληκτρολογηθούν και παραμένουν εκεί μέχρι να αντικατασταθούν από άλλα δεδομένα. Είναι προφανές ότι η απόκτηση πρόσβασης σε κρυπτογραφημένα αρχεία και προσωπικούς λογαριασμούς είναι ζωτικής σημασίας.
5. Αποκρυπτογραφημένο περιεχόμενο: Όταν ένα αρχείο αποκρυπτογραφείται φορτώνεται στη μνήμη και παραμένει εκεί ακόμα και μετά το κλείσιμο του αρχείου. Είναι αρκετά πιθανό λοιπόν να ανακτηθεί περιεχόμενο από κρυπτογραφημένα αρχεία ακόμα και αν δεν έχουμε το κλειδί.
6. Κρυφά δεδομένα και κακόβουλος κώδικας: Είναι συνήθης τεχνική για τους επιτιθέμενους να αποθηκεύουν και να τρέχουν τον κακόβουλο κώδικα μέσα από τη μνήμη έτσι ώστε να αποφεύγουν την ανίχνευση από αντιικά προγράμματα αλλά και να μην αφήνουν αποδείξεις. Ο εντοπισμός λοιπόν τέτοιων δεδομένων μπορεί να μας οδηγήσει στην ανακάλυψη σημαντικών πληροφοριών.

Η ανάγκη για την επιλογή του κατάλληλου λογισμικού για την ανάλυση της μνήμης είναι εμφανής μιας και όπως αναφέραμε οι πληροφορίες που μπορούν να αποσπαστούν είναι πολλή σημαντικές. Υπάρχουν αρκετές επιλογές εργαλείων ανάλυσης μνήμης όπως τα Memoryze, KnTList,WMFT(Windows Memory Forensics Toolkit) αλλά το εργαλείο που επιλέχθηκε λόγο των μοναδικών χαρακτηριστικών του είναι το volatility. Το volatility είναι ένα εργαλείο ανοιχτού κώδικα και μπορεί να εκτελεστεί μέσω γραμμής εντολών (CLI). Είναι το μόνο πρόγραμμα που μπορεί να εκτελεστεί και στα τρία λειτουργικά συστήματα (Linux, Windows, Mac) και υποστηρίζει τόσο 32bit όσο και 64bit αντίγραφα μνήμης από τα λειτουργικά αυτά. Επιτρέπει επίσης την ανάλυση όχι μόνο raw τύπου αρχείων αλλά και crash dumps, hibernation files και πολλών ακόμα, καθώς και το μετασχηματισμό από τον ένα τύπο αρχείου σε άλλο. Είναι γραμμένο σε python με τόσο αποδοτικό τρόπο που επιτρέπει την ανάλυση σε λιγότερο χρόνο από τα υπόλοιπα εργαλεία με πολύ μικρή κατανάλωση πόρων του μηχανήματος.

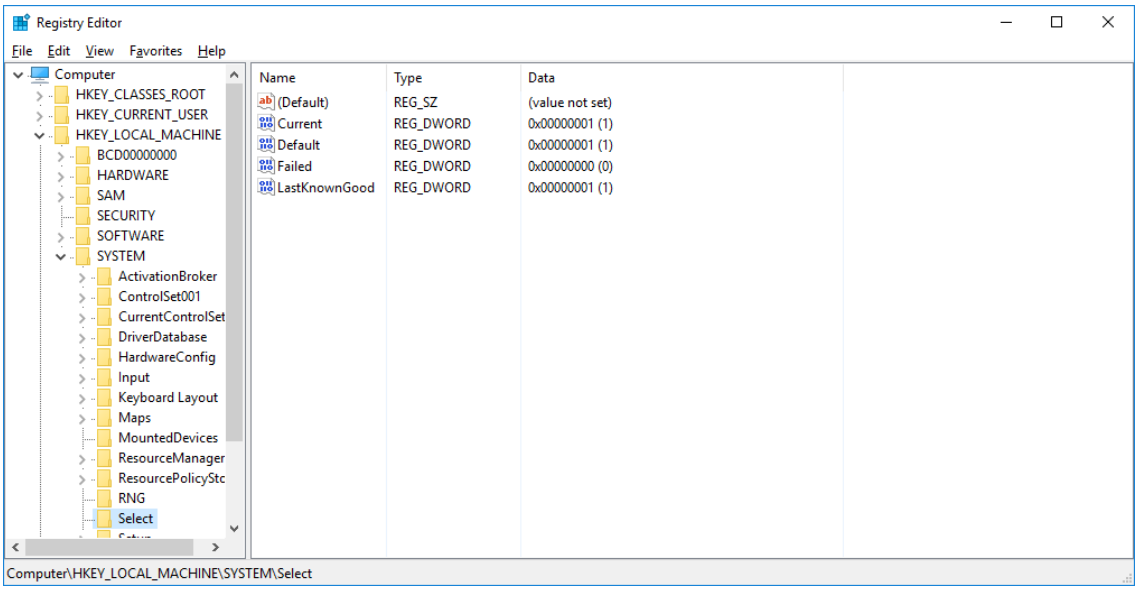
Αυτό που καθιστά το volatility όμως μοναδικό είναι ο τεράστιος αριθμός από πρόσθετα (plugins) που υποστηρίζει. Τα plugins αυτά χωρίζονται σε κατηγορίες ανάλογα με τις λειτουργίες τους το σκοπό που εξυπηρετούν καθώς και το λειτουργικό σύστημα που αφορούν. Για τον σωστό προσδιορισμό του αντιγράφου της μνήμης έχουμε τα πρόσθετα imageinfo, kdbgscan, kpcrscan τα οποία αναγνωρίζουν το λειτουργικό και την έκδοση (service pack) από όπου προήλθε το αντίγραφο, την αρχιτεκτονική του υπολογιστή και τη σωστή διεύθυνση kdbg. Υπάρχει επίσης ένας μεγάλος αριθμός από πρόσθετα όπως τα pslist,dlllist, psscan που χρησιμοποιούνται για να ανασύρουν πληροφορίες που αφορούν διεργασίες, κλήσεις διεργασιών σε dll αρχεία, και κρυφές διεργασίες. Πολλές ακόμα πληροφορίες μπορούν να αποκτηθούν από τη μνήμη με τη χρήση κατάλληλων προσθέτων, όπως ενεργές συνδέσεις και πόρτες, διευθύνσεις της registry στη μνήμη και προγράμματα οδήγησης του πυρήνα του λογισμικού που υπάρχουν στη μνήμη.

Η απόκτηση και ανάλυση της μνήμης είναι δύο πολλή λεπτές αλληλένδετες διαδικασίες αφού σωστή και λεπτομερής ανάλυση της μνήμης βασίζεται στην απόκτηση ενός ακέραιου αντιγράφου και απαιτούν σωστό σχεδιασμό και διερεύνηση των συνθηκών της κάθε υπόθεσης για την επιλογή των κατάλληλων εργαλείων.

4.3.7 Forecopy

Η registry των Windows, είναι μια ιεραρχική βάση δεδομένων η οποία χρησιμοποιείται για την αποθήκευση πληροφοριών του συστήματος όπως ρυθμίσεις, συνδεδεμένες συσκευές, εφαρμογές και πληροφορίες χρηστών. Η registry εμφανίζεται με δενδροειδή μορφή σαν ένα ενοποιημένο σύστημα όπως βλέπουμε στην εικόνα 2 με κάθε φάκελο-ρίζα να αποτελεί μια κυψέλη και κάθε φάκελο-φύλλο να αποτελεί ένα κλειδί. Τα ονόματα των αντικειμένων μέσα σε

κάθε κλειδί αποτελούν τις τιμές του και εξαρτώνται από τα δεδομένα που περιέχουν, τα οποία ερμηνεύονται από το λειτουργικό σύστημα.



Εικόνα 12: Δομή της registry

Παρόλο που εμφανίζεται στο χρήστη σαν ένα ενοποιημένο αρχείο, η registry διατηρείται σε πολλαπλά αρχεία στο σύστημα και πολλά από τα δεδομένα που εμφανίζονται είναι πτητικά και δεν υπάρχουν μέχρι να φορτωθεί το λογισμικό. Μέσα στην registry μπορούν να βρεθούν πληροφορίες ζωτικής σημασίας [24] για την εξέλιξη μίας έρευνας με το ενδιαφέρον να επικεντρώνεται στα ακόλουθα:

1. Autostart Locations: Πρόκειται για κλειδιά στη registry που επιτρέπουν την εκκίνηση εφαρμογών αυτόματα χωρίς την έγκριση του χρήστη είτε με την εκκίνηση του λογισμικού είτε όταν συμβεί μια άλλη ενέργεια (πχ εκκίνηση της γραμμής εντολών).
2. User Activity: Πρόκειται για κλειδιά που περιέχουν πληροφορίες για τις ενέργειες του χρήστη στο λειτουργικό, όπως είναι οι λίστες MRU (Most Recently Used) οι οποίες διατηρούν ένα ιστορικό των πρόσφατων εφαρμογών που εκτελέστηκαν.
3. Αποθηκευτικές συσκευές: Μπορούμε να ανασύρουμε από κλειδιά της registry πληροφορίες για όλα τα αποθηκευτικά μέσα που συνδέθηκαν στον υπολογιστή όπως usb flash drives και φωτογραφικές μηχανές με το μοναδικό σειριακό αριθμό τους.
4. Ασύρματες συνδέσεις: Αναλυτικά δεδομένα για όλα τα δίκτυα στα οποία έχει συνδεθεί ο χρήστης όπως SSID,ip διεύθυνση, ημερομηνία και ώρα σύνδεσης, τα οποία βρίσκονται αποθηκευμένα σε κλειδιά της registry.

Τα αρχεία της registry ανοίγονται από τον πυρήνα του λειτουργικού σε restricted mode, το οποίο εμποδίζει την αντιγραφή τους όταν το λειτουργικό είναι σε λειτουργία, με εξαίρεση τα User Registry Files τα οποία δεν έχουν φορτωθεί [2].

Η αντιγραφή των αρχείων της registry, μπορεί να πραγματοποιηθεί, είτε με τη χρήση ενός live cd σε απενεργοποιημένους υπολογιστές και με την παραδοχή ότι δεν θα αποκτήσουμε όλα τα δεδομένα από τη registry, είτε με τη χρήση εξωτερικών εργαλείων όπως τα ERUNT, Forecopy, FTK Imager. Για την αντιγραφή της registry επιλέξαμε το Forecopy, ένα εργαλείο το οποίο υποστηρίζει την αντιγραφή bit προς bit αρχείων χωρίς να μεταβάλει ή να αλλοιώνει τα μεταδεδομένα και τις χρονικές σφραγίδες των αρχείων.

4.3.8 RegRipper

Η ανάλυση της registry, για καιρό ήταν ένα όπλο για τους ερευνητές, το οποίο δεν αξιοποιούταν πλήρως. Ο τεράστιος όγκος πληροφοριών ο οποίος περιέχεται στη registry, καθιστούσε την ανάλυσή της, μια χρονοβόρα και δύσκολη διαδικασία κάτι το οποίο οδήγησε στην ανάγκη για

την ανάπτυξη εργαλείων τα οποία αυτόματα θα εξήγαγαν και θα παρουσίαζαν όλες τις χρήσιμες πληροφορίες. Το RegRipper είναι ένα εργαλείο ανοιχτού κώδικα, που αναπτύχθηκε γι αυτό ακριβώς το σκοπό. Έχει τη δυνατότητα να εντοπίζει, να συσχετίζει και να παρουσιάζει κλειδιά της registry που περιέχουν τις πληροφορίες που χρειαζόμαστε, όπως MRU lists, autostart locations κ.α. με τη βοήθεια των κατάλληλων plugins. Μπορεί να εκτελεστεί είτε μέσω γραφικού περιβάλλοντος (GUI) είτε μέσω γραμμής εντολών (CLI)

Η registry αποτελεί μια σημαντική πηγή πληροφοριών και η ανάλυση της μπορεί να οδηγήσει στην λύση μιας υπόθεσης αλλά απαιτεί κάτι παραπάνω από την απλή εκτέλεση ενός εργαλείου ακριβώς επειδή ο όγκος των πληροφοριών είναι τεράστιος. Χρειάζεται βαθιά γνώση της δομής της και των πληροφοριών που περιέχει για την σωστή ερμηνεία των αποτελεσμάτων.

4.4 Σχηματική αναπαράσταση εξαρτήσεων

Στο σημείο αυτό, θα παραθέσουμε μια σχηματική αναπαράσταση των εξαρτήσεων του αλγόριθμου, τόσο σε αρχεία και modules όσο και σε εσωτερικές εξαρτήσεις των συναρτήσεων. Στο διάγραμμα που ακολουθεί φαίνονται όλες οι εξαρτήσεις που υπάρχουν στο script που δημιουργήθηκε.

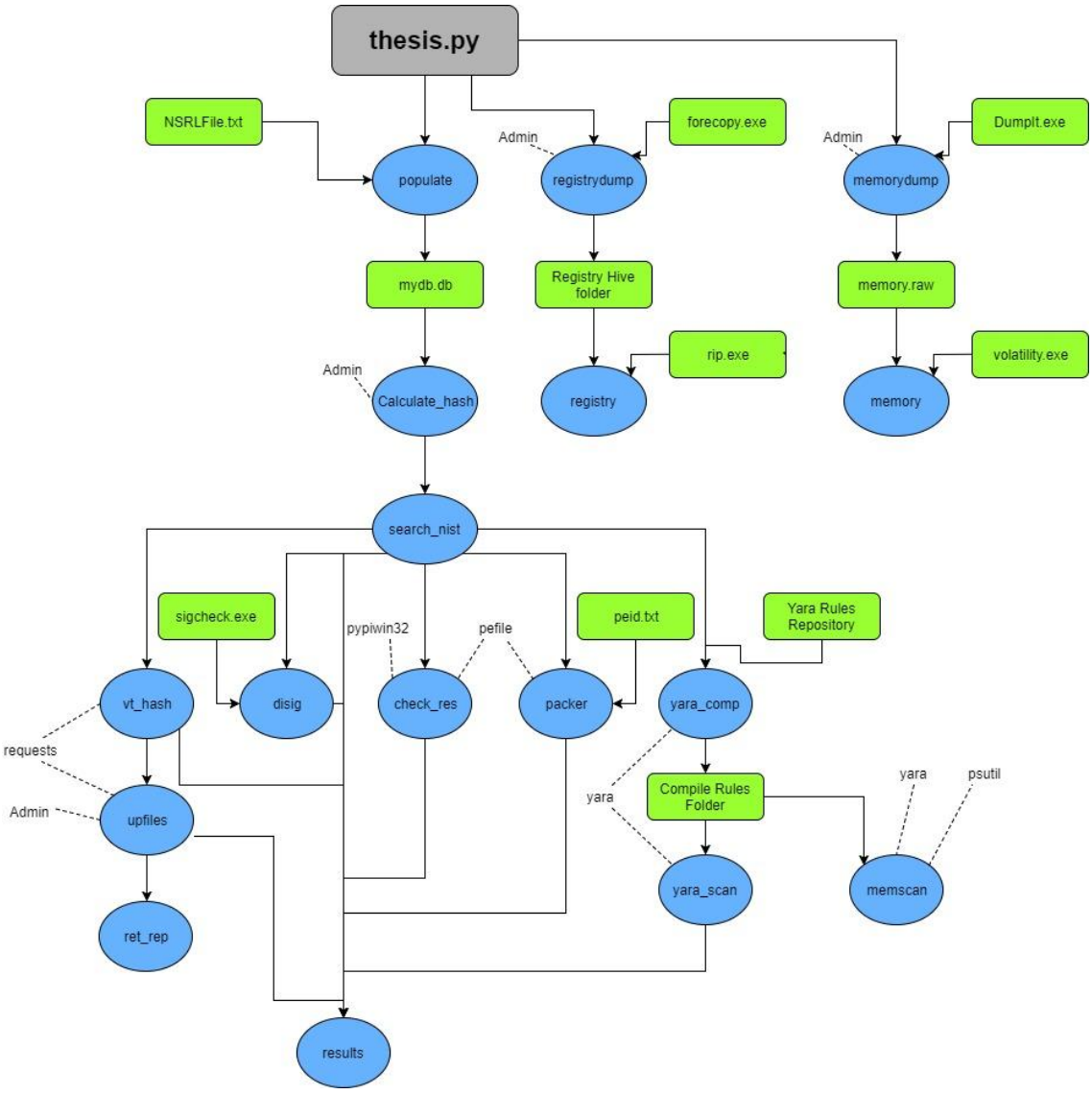
Οι γαλάζιες ελλείψεις, συμβολίζουν τις συναρτήσεις, τα πράσινα τετράγωνα τα αρχεία και οι γραμμές που τα ενώνουν τις εξαρτήσεις τους. Οι διακεκομμένες γραμμές αναπαριστούν τις εξαρτήσεις των συναρτήσεων από τα modules που δεν είναι ενσωματωμένα στην python.

Στον πίνακα που ακολουθεί παραθέτεται επιγραμματικά το όνομα της κάθε συνάρτησης του εργαλείου που αναπτύχθηκε καθώς και η λειτουργία που εξυπηρετεί.

Όνομα συνάρτησης	**Λειτουργία συνάρτησης**
populate	Δημιουργεί τη βάση και τη γεμίζει με τα δεδομένα του αρχείου NSRL
calculate_hash	Υπολογίζει τα hashes και τα εισάγει στη βάση
search_nist	Ελέγχει ποια από τα hashes υπάρχουν στη βάση της NSRL
vt_hash	Στέλνει στο virus total τις τιμές των hash
disig	Ελέγχει τις ψηφιακές υπογραφές
check_res	Ελέγχει τα resources και την εικόνα του αρχείου
upload	Ανεβάζει στο virus total τα αρχεία
ret_rep	Ανασύρει τις αναφορές των αρχείων που απέστειλε
packer	Ανιχνεύει υπογραφές γνωστών packer στα PE αρχεία
yara_comp	Μεταγλωττίζει τους κανόνες
yara_scan	Ελέγχει τα αρχεία με βάση τους κανόνες
registrydump	Αντιγράφει τα αρχεία της registry
registry	Αναλύει τις κυψέλες της registry
memorydump	Αποθηκεύει τα δεδομένα της μνήμης
memory	Αναλύει το dump της μνήμης
mem_scan	Ελέγχει τις διεργασίες με τη χρήση yara rules

results	Γράφει τα αποτελέσματα σε αρχεία

Πίνακας 1: Πίνακας συναρτήσεων



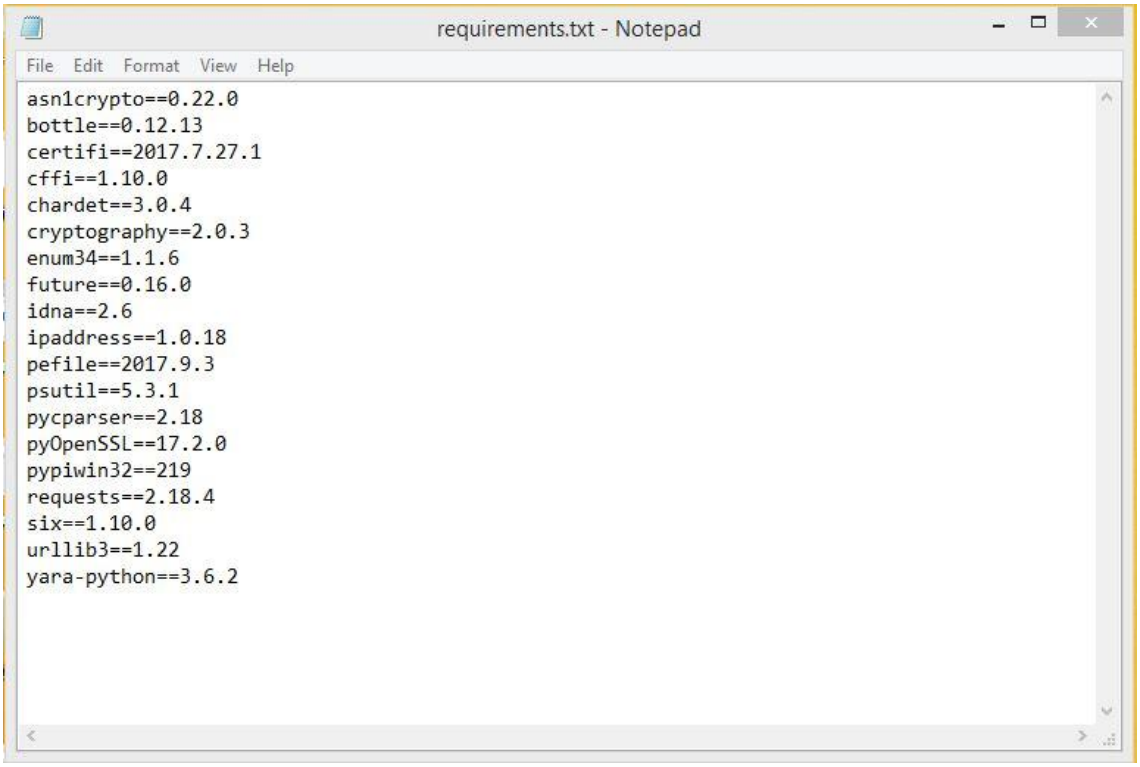
Εικόνα 13: Διάγραμμα εξαρτήσεων του εργαλείου

Στο κεφάλαιο που ακολουθεί θα δούμε τον τρόπο εγκατάστασης των βιβλιοθηκών αυτών οι οποίες είναι αναγκαίες για τη λειτουργία του εργαλείου.

4.5 Εγκατάσταση

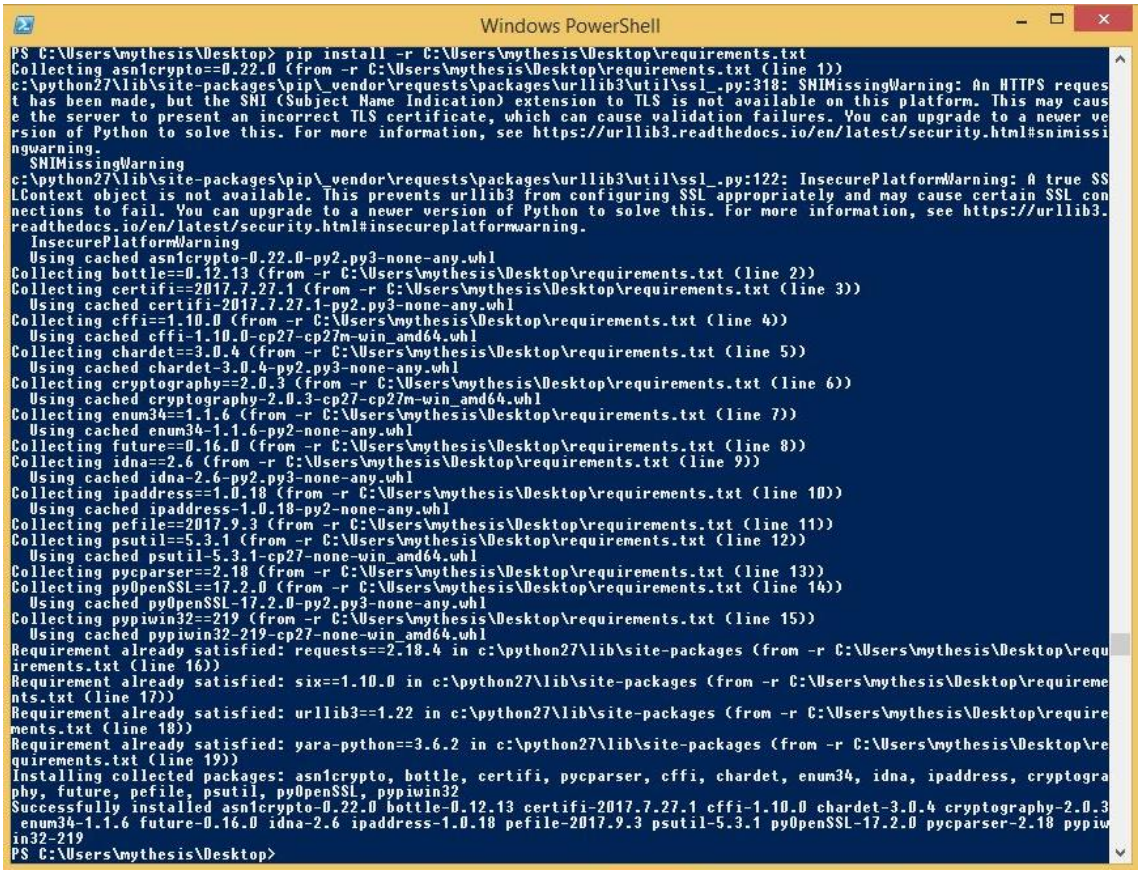
Για την εγκατάσταση των απαραίτητων βιβλιοθηκών, μπορούμε είτε να χρησιμοποιήσουμε το εργαλείο pip είτε να κατεβάσουμε και να «χτίσουμε» τις βιβλιοθήκες χειροκίνητα.

Για την ευκολότερη εγκατάσταση των βιβλιοθηκών παρέχεται το αρχείο «requirements.txt» τα περιεχόμενα του οποίου φαίνονται στην εικόνα που ακολουθεί και περιέχει όλες τις βιβλιοθήκες που χρησιμοποιούνται από το εργαλείο που δημιουργήθηκε.



Εικόνα 14: Περιεχόμενα αρχείου εξαρτήσεων

Για την εγκατάσταση των βιβλιοθηκών με τη χρήση του αρχείου «requirements.txt» εκτελούμε στη γραμμή εντολών την εντολή «pip install –r requirements.txt» και όλες οι βιβλιοθήκες συγκεντρώνονται και εγκαθίστανται στο σύστημα όπως βλέπουμε και στο στιγμιότυπο που ακολουθεί.



Εικόνα 15: Στιγμιότυπο εγκατάστασης εξαρτήσεων

4.6 Ορίσματα χρήσης

Στην παράγραφο αυτή, θα παρουσιάσουμε αναλυτικά, όλα τα ορίσματα χρήσης του εργαλείου που αναπτύχθηκε. Με την παράμετρο «–h» εκτυπώνονται όλες οι παράμετροι και οι λειτουργίες του.

Εικόνα 16: Εκτύπωση βοήθειας εργαλείου

Όπως μπορούμε να δούμε και στο παραπάνω στιγμιότυπο οι παράμετροι του εργαλείου είναι οι εξής:

- **-h, --help:** Εκτυπώνει το κείμενο επεξήγησης παραμέτρων.
- **-p, --populate:** Δημιουργεί τη βάση και τη γεμίζει με τα δεδομένα του αρχείου NSRL.
- **-c, --hash:** Υπολογίζει τα hash values για όλα τα αρχεία και τα εισάγει στη βάση.
- **-hvt, --send_hash:** Αποστέλλει τις τιμές της συνάρτησης κατακερματισμού στο virus total και ανακτά τις αναφορές.
- **-uvt, --upload:** Αποστέλλει τα αρχεία στο virus total για έλεγχο.
- -**rrvt, --reports:** Συλλέγει τις αναφορές των αρχείων που εστάλησαν στο virus total.
- **-s, --search_nist:** Ελέγχει ποιά από τα hash values υπάρχουν στη βάση της NSRL.
- **-ds, --dig_sig:** Ελέγχει τις ψηφιακές υπογραφές των αρχείων.
- **-r, --resources:** Ελέγχει τα resources του αρχείου.
- **-P, --packer:** Ελέγχει τα αρχεία για εντοπισμό συμβολοσειρών από γνωστούς packer.
- **-yc, --yara_comp:** Μεταγλωττίζει τους κανόνες yara.
- **-ys, --yara_scan:** Ελέγχει τα αρχεία χρησιμοποιώντας τους κανόνες yara.
- **-ym, --yara_memory:** Ελέγχει τις ενεργές διεργασίες χρησιμοποιώντας κανόνες yara.
- -**RD, --regdump:** Αντιγράφει τα αρχεία της registry.
- **-R, --registry:** Αναλύει τα αντίγραφα της registry.
- **-MD, --memdump:** Αποθηκεύει τα δεδομένα της μνήμης σε ένα αρχείο.
- -**M, --memory:** Αναλύει τη μνήμη.
- **-rs, --results:** Επιστρέφει τα αποτελέσματα των ελέγχων των αρχείων .

4.7 Παραδείγματα χρήσης

Για τον έλεγχο του εργαλείου, δημιουργήθηκε ένας φάκελος, ο οποίος περιείχε κάποια «.exe» και «.dll» αρχεία, τόσο από έμπιστες πηγές όσο και κακόβουλα, προκειμένου να επιδείξουμε τη λειτουργία του εργαλείου. Στο επόμενο κεφάλαιο των αποτελεσμάτων παρατίθεται μια αναλυτικότερη προσέγγιση.

Αρχικά για τη δημιουργία της βάσης εκτελούμε την εντολή με το διακόπτη –p όπως φαίνεται στο στιγμιότυπο που ακολουθεί. Ο χρήστης δίνει σαν είσοδο το πλήρες μονοπάτι του αρχείου της NIST και στη συνέχεια πληκτρολογεί το όνομα της νέας βάσης δεδομένων η οποία «γεμίζει» με τα δεδομένα του αρχείου. Με την ολοκλήρωση του βήματος ο πίνακας «NIST» της βάσης, έχει την μορφή που παρουσιάστηκε στην Εικόνα 6.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -p E:\stam\NSRLFile.txt
Give a database name: mydb
```

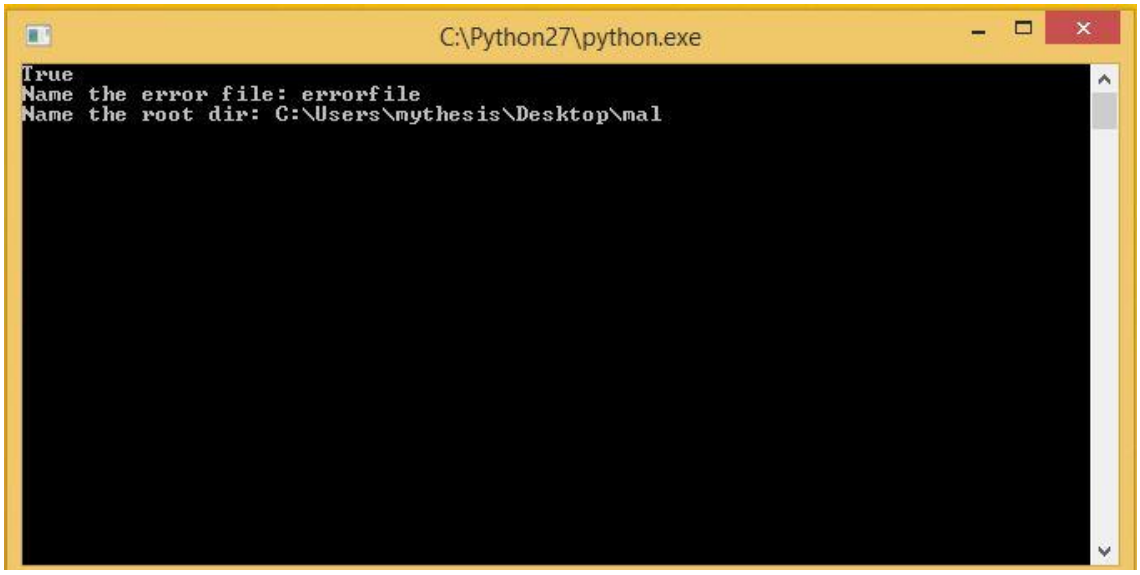
Εικόνα 17: Εντολή δημιουργίας της βάσης

Στη συνέχεια, στο διακόπτη «–c» δίνουμε σαν όρισμα το πλήρες μονοπάτι της βάσης δεδομένων, στην οποία θα αποθηκευτούν τα path και τα hash value όλων των αρχείων.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -c C:\Users\mythesis\Desktop\mydb.db
True
You're not an admin
```

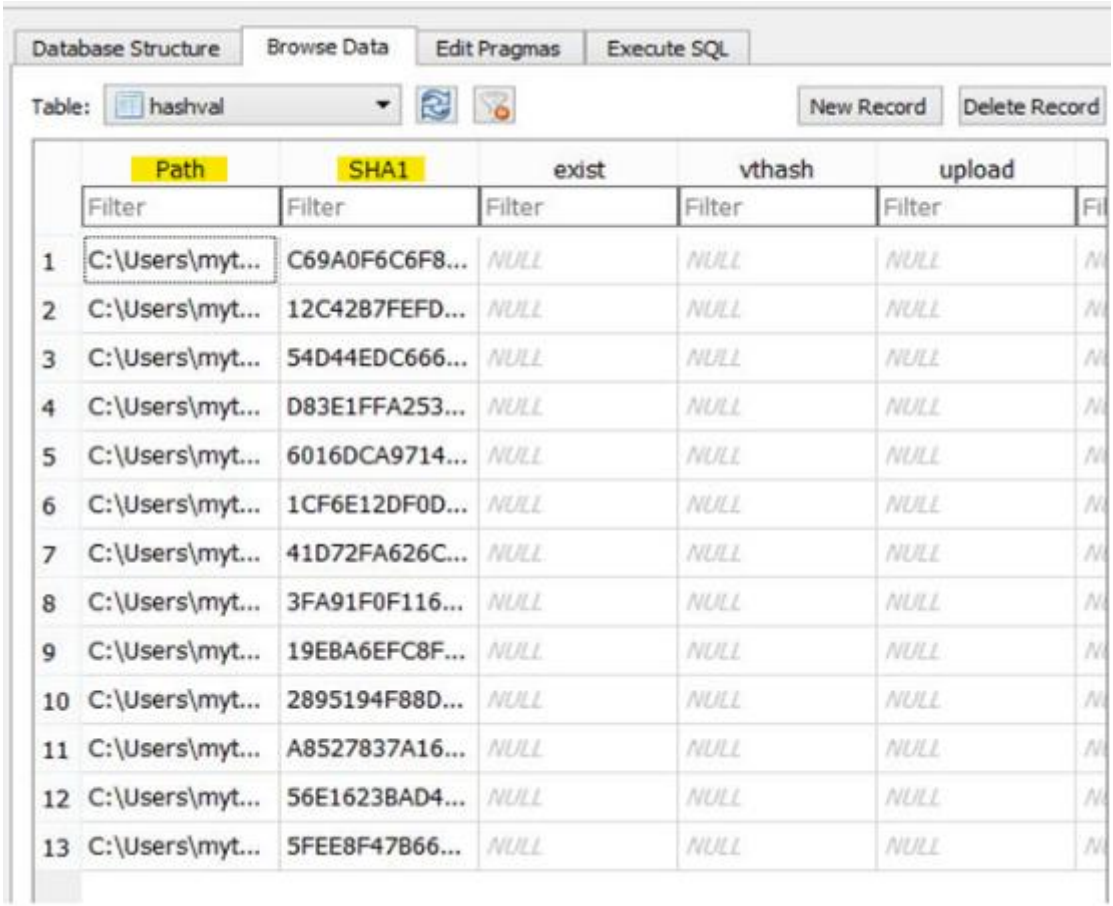
Εικόνα 18: Εντολή υπολογισμού hash values των αρχείων του υπολογιστή

Το script ανιχνεύει τα δικαιώματα εκτέλεσής του και αν εκτελείται με προνόμια χρήστη, ζητάει αύξηση των δικαιωμάτων του. Στη συνέχεια, ζητάει από το χρήστη να δώσει ένα όνομα για το έγγραφο κειμένου, το οποίο θα περιέχει όλα τα μονοπάτια των αρχείων που δεν μπόρεσαν να ανοιχτούν για να υπολογιστεί η τιμή της συνάρτησης κατακερματισμού τους και το «root directory» δηλαδή τον κατάλογο κάτω από τον οποίο θα προσπελάσει όλα τα αρχεία.



Εικόνα 19: Στιγμιότυπο εκτέλεσης του διακόπτη -c

Με την εκτέλεση του συγκεκριμένου διακόπτη, τα δεδομένα της βάσης παίρνουν την μορφή που εμφανίζεται στο επόμενο στιγμιότυπο.



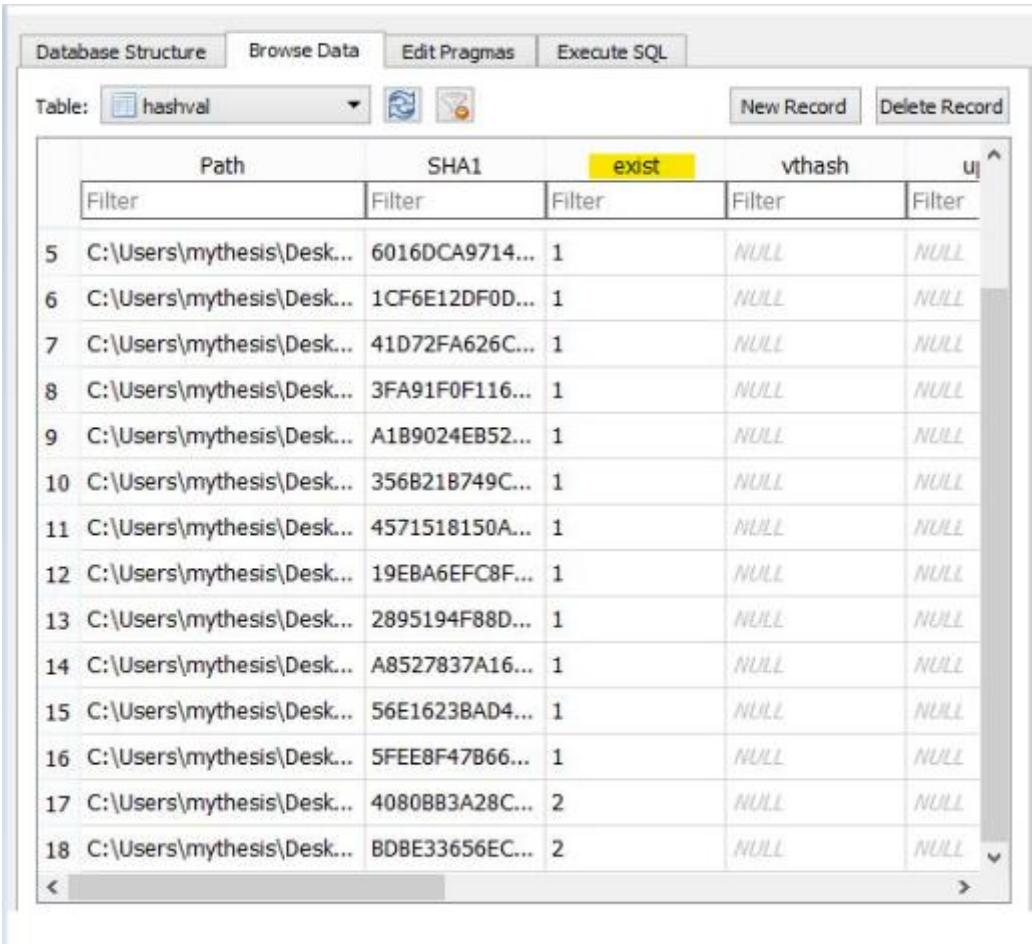
	Path	SHA1	exist	vthash	upload
	Filter	Filter	Filter	Filter	Filter
1	C:\Users\myt...	C69A0F6C6F8...	NULL	NULL	NULL
2	C:\Users\myt...	12C4287FEFD...	NULL	NULL	NULL
3	C:\Users\myt...	54D44EDC666...	NULL	NULL	NULL
4	C:\Users\myt...	D83E1FFA253...	NULL	NULL	NULL
5	C:\Users\myt...	6016DCA9714...	NULL	NULL	NULL
6	C:\Users\myt...	1CF6E12DF0D...	NULL	NULL	NULL
7	C:\Users\myt...	41D72FA626C...	NULL	NULL	NULL
8	C:\Users\myt...	3FA91F0F116...	NULL	NULL	NULL
9	C:\Users\myt...	19EBA6EFC8F...	NULL	NULL	NULL
10	C:\Users\myt...	2895194F88D...	NULL	NULL	NULL
11	C:\Users\myt...	A8527837A16...	NULL	NULL	NULL
12	C:\Users\myt...	56E1623BAD4...	NULL	NULL	NULL
13	C:\Users\myt...	5FEE8F47B66...	NULL	NULL	NULL

Εικόνα 20: Στιγμιότυπο της βάσης με την εκτέλεση του διακόπτη -c

Αν κατά την εκτέλεση του διακόπτη, ανιχνευτούν δεδομένα στη βάση, τότε στο χρήστη θα εμφανιστεί το ακόλουθο μήνυμα, δίνοντας του την δυνατότητα να επιλέξει αν τα παλιά δεδομένα θα αντικατασταθούν από τα νέα, ή αν θα διατηρηθούν στη βάση.

Εικόνα 21: Στιγμιότυπο εκτέλεσης του διακόπτη -c

Με την ολοκλήρωση του υπολογισμού, η βάση δεδομένων παίρνει την ακόλουθη μορφή.



	Path	SHA1	exist	vthash	u
	Filter	Filter	Filter	Filter	Filter
5	C:\Users\mythesis\Desk...	6016DCA9714...	1	NULL	NULL
6	C:\Users\mythesis\Desk...	1CF6E12DF0D...	1	NULL	NULL
7	C:\Users\mythesis\Desk...	41D72FA626C...	1	NULL	NULL
8	C:\Users\mythesis\Desk...	3FA91F0F116...	1	NULL	NULL
9	C:\Users\mythesis\Desk...	A1B9024EB52...	1	NULL	NULL
10	C:\Users\mythesis\Desk...	356B21B749C...	1	NULL	NULL
11	C:\Users\mythesis\Desk...	4571518150A...	1	NULL	NULL
12	C:\Users\mythesis\Desk...	19EBA6EFC8F...	1	NULL	NULL
13	C:\Users\mythesis\Desk...	2895194F88D...	1	NULL	NULL
14	C:\Users\mythesis\Desk...	A8527837A16...	1	NULL	NULL
15	C:\Users\mythesis\Desk...	56E1623BAD4...	1	NULL	NULL
16	C:\Users\mythesis\Desk...	5FEE8F47B66...	1	NULL	NULL
17	C:\Users\mythesis\Desk...	4080BB3A28C...	2	NULL	NULL
18	C:\Users\mythesis\Desk...	BD8E33656EC...	2	NULL	NULL

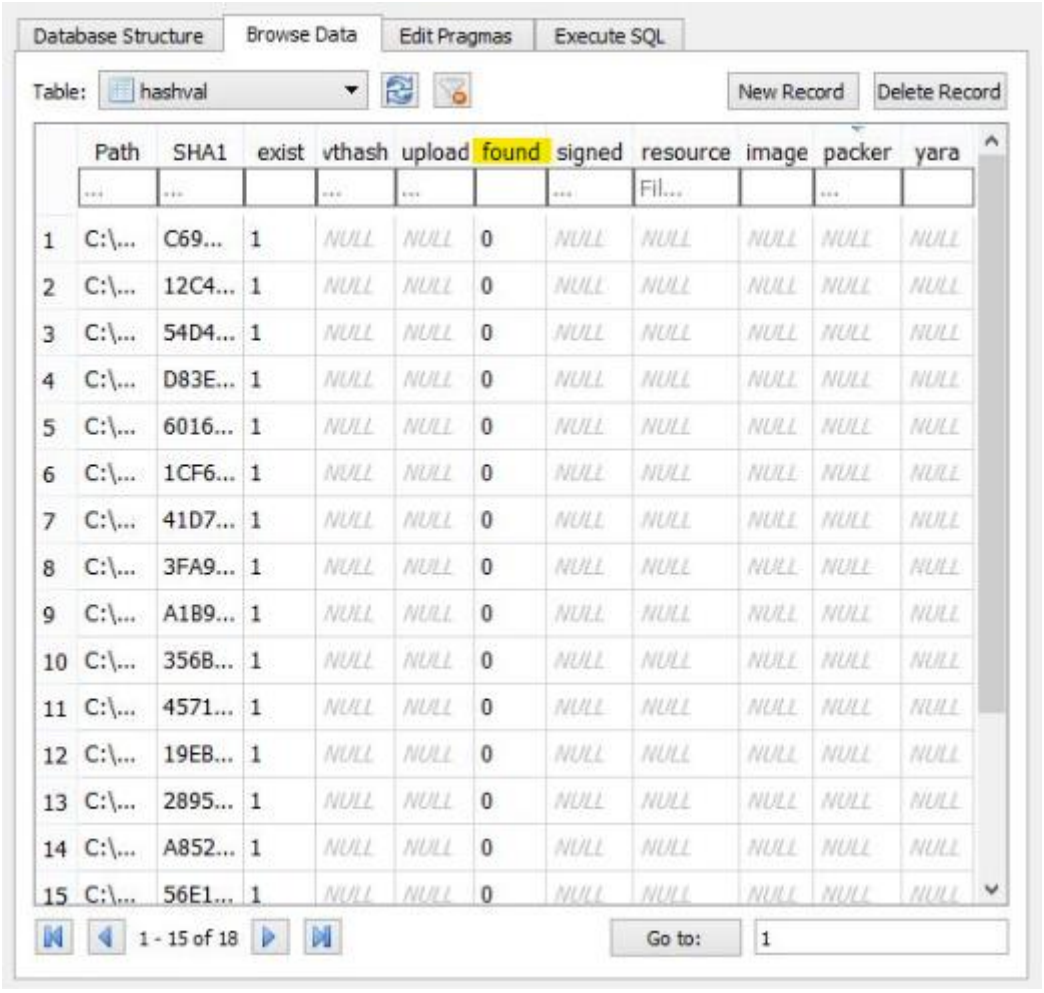
Εικόνα 22: Στιγμιότυπο της βάσης μετά τον επαναϋπολογισμο των hash values

Στη συνέχεια, με τη χρήση του διακόπτη «–s», όλα τα hash values του πίνακα αναζητούνται στη βάση της NIST. Οι παράμετροι του διακόπτη είναι μόνο το path στο οποίο είναι αποθηκευμένη η βάση δεδομένων που έχουμε δημιουργήσει.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -s C:\Users\mythesis\Desktop\mydb.db
```

Εικόνα 23: Εντολή αναζήτησης των hash values στη βάση NSRL

Με την ολοκλήρωση του βήματος, ο πίνακας έχει πάρει την μορφή όπως εμφανίζεται στο στιγμιότυπο που ακολουθεί.



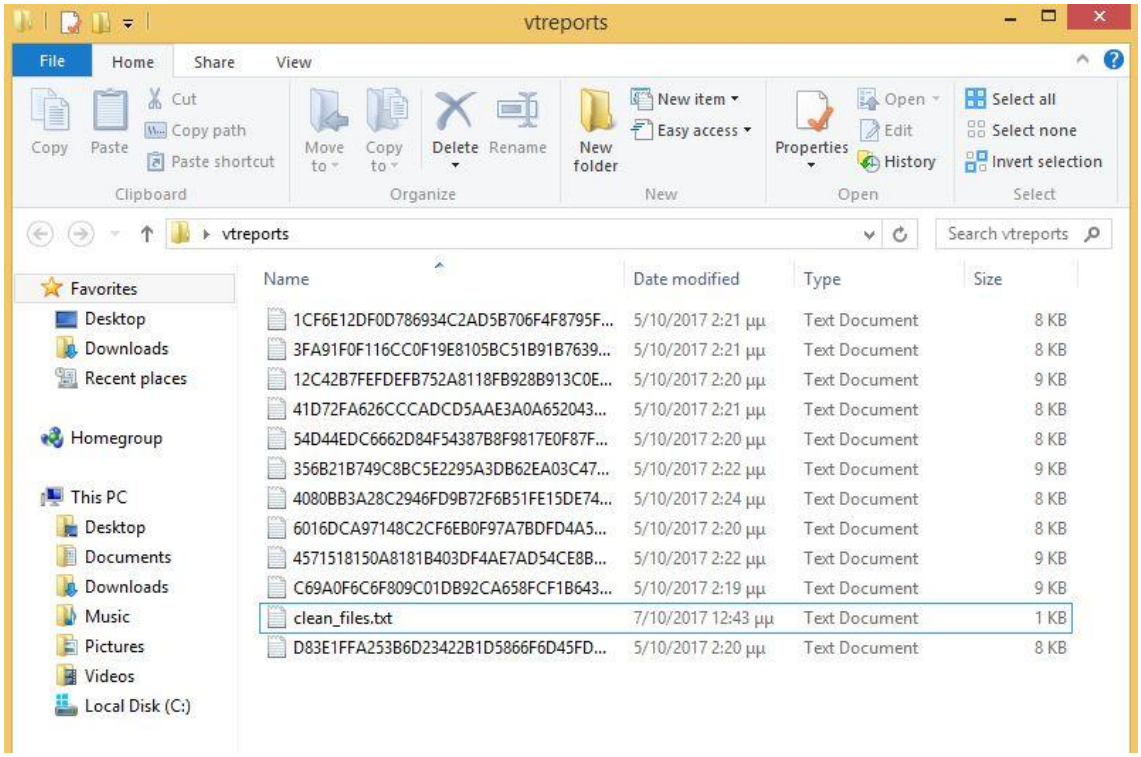
Εικόνα 24: Στιγμιότυπο της βάσης μετά τη χρήση του διακόπτη -s

Με τη χρήση του διακόπτη «–hvt» όπως έχει ήδη αναφερθεί, αποστέλλονται τα hash values στο virus total για ανάλυση. Η μόνη παράμετρος του διακόπτη, είναι το μονοπάτι της βάσης. Κατά την εκτέλεση, ζητείτε από το χρήστη να δώσει το κλειδί του virus total και να επιλέξει κατά πόσο το κλειδί είναι ιδιωτικό ή όχι

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -hvt C:\Users\mythesis\Desktop\mydb.db
Enter your API key: 62239eba8f6f8f49a792a445d652d6dccfd616fbb15458507c6ffed079d392c6
Is your VT key private (y/n)?: n
```

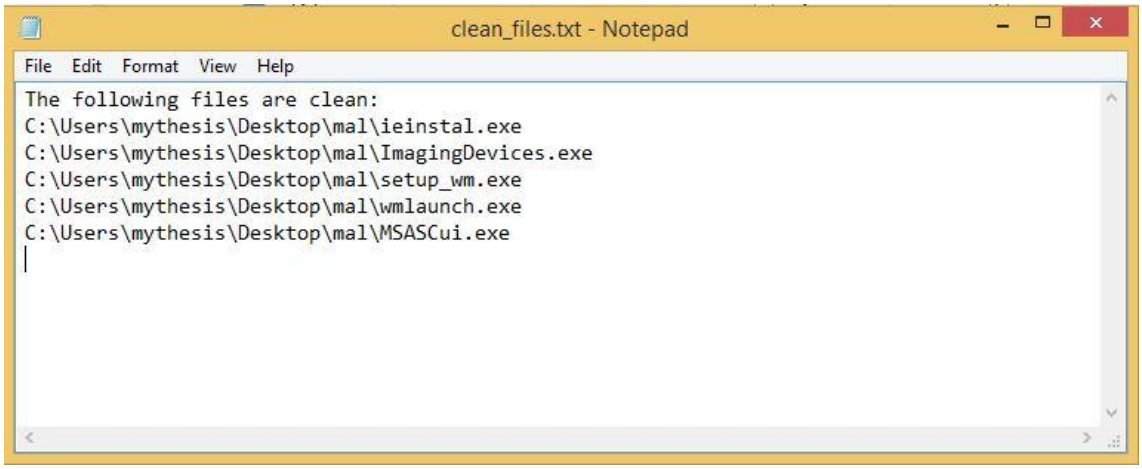
Εικόνα 25: Εντολή αποστολής hash values στο virus total

Με την ολοκλήρωση της διαδικασίας αποστολής των τιμών της συνάρτησης κατακερματισμού, έχει δημιουργηθεί ένας νέος φάκελος στον κατάλογο από τον οποίο εκτελούμε το script με όνομα vtreports, ο οποίος περιέχει όλες τις αναφορές.



Εικόνα 26: Φάκελος αναφορών του VirusTotal

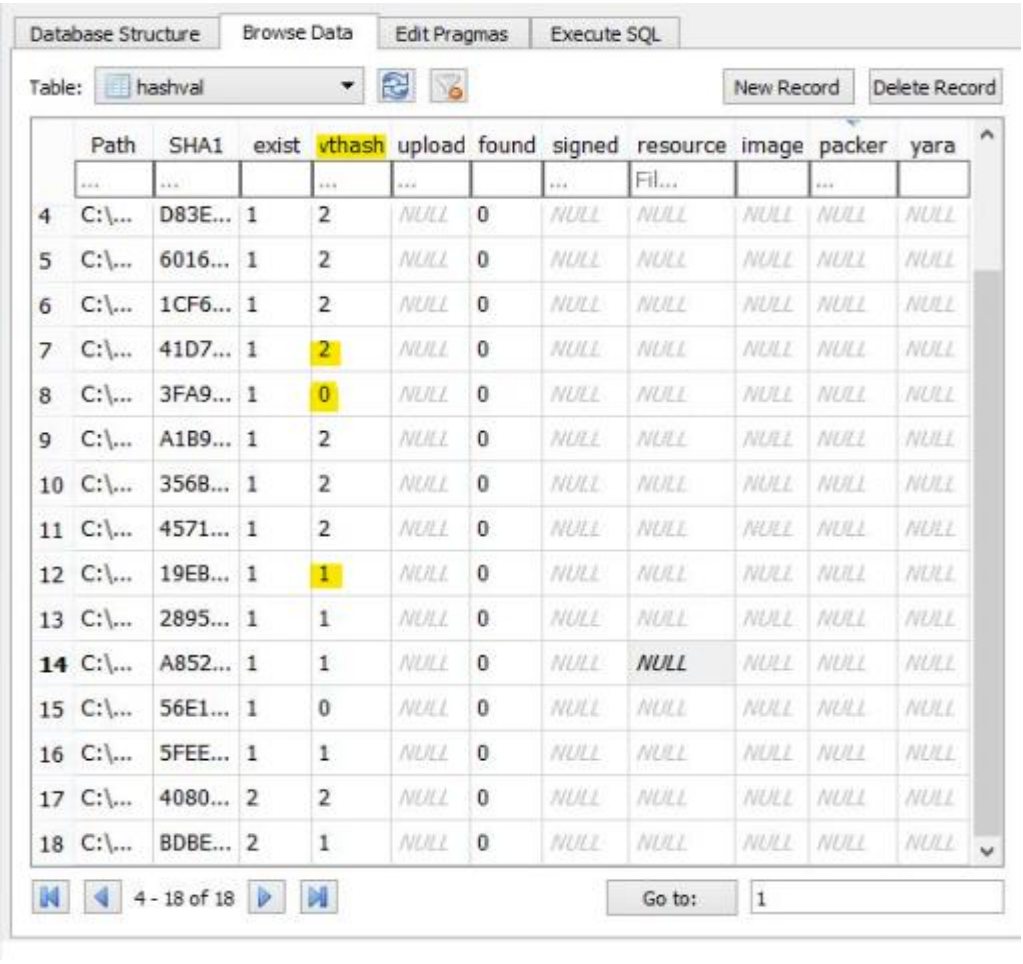
Το αρχείο «clean_files.txt», περιέχει τα μονοπάτια όλων των αρχείων που είχαν detection rate μηδέν, τα υπόλοιπα αρχεία είναι, αρχεία αναφορών του virus total. Τα περιεχόμενα των αρχείων εμφανίζονται στα δύο επόμενα στιγμιότυπα.



Εικόνα 27: Περιεχόμενα αρχείου clean_files.txt

Εικόνα 28: Περιεχόμενα αναφοράς κακόβουλου λογισμικού

Η βάση δεδομένων έχει μεταβληθεί σύμφωνα με το ακόλουθο στιγμιότυπο



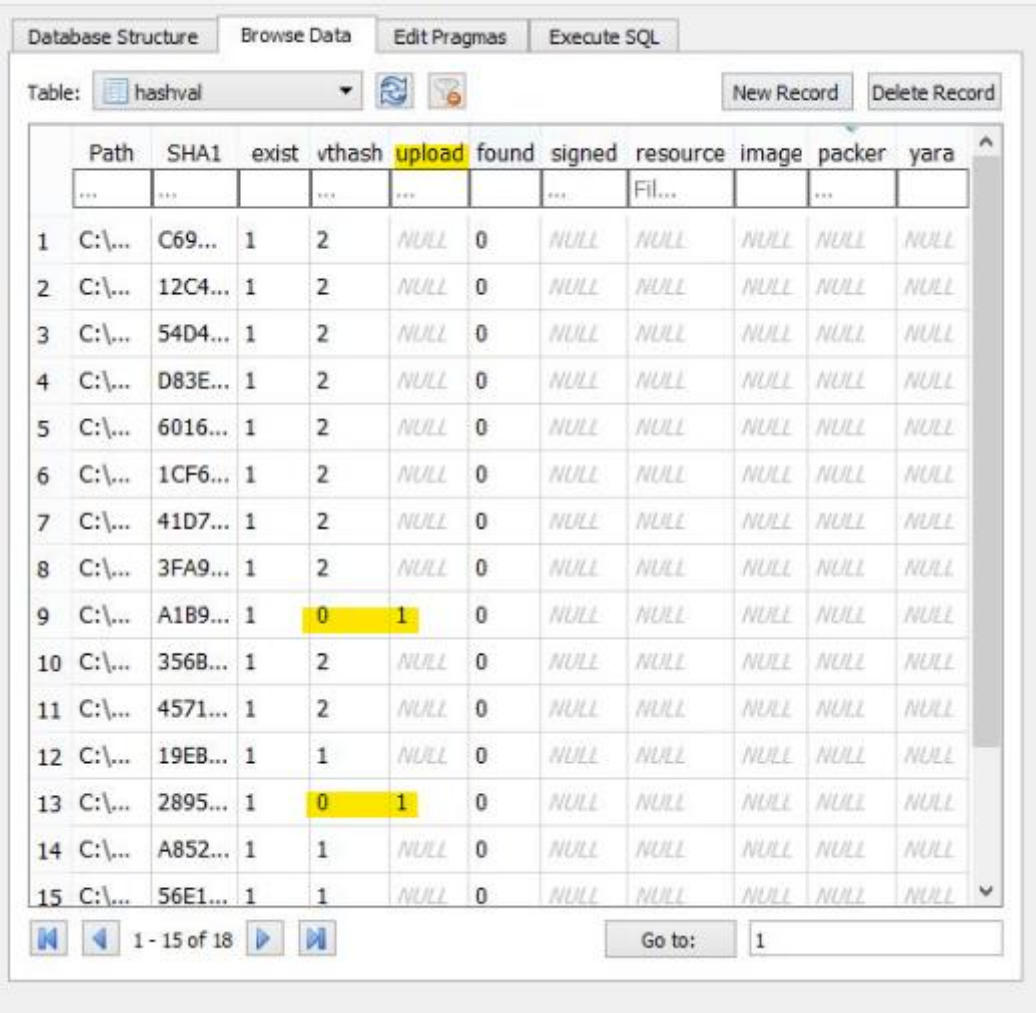
Εικόνα 29: Διαμόρφωση της βάσης μετά την εκτέλεση του διακόπτη -hvt

Με τη χρήση του διακόπτη «–uvt», τα αρχεία που δεν εντοπίστηκαν στο virus total με το hash value, ανεβαίνουν στο site για ανάλυση. Ο διακόπτης αυτός έχει σαν παραμέτρους το μονοπάτι της βάσης, καθώς και το κλειδί που έχει προμηθευτεί ο χρήστης από το virus total

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -uvt C:\Users\mythesis\Desktop\mydb.db 62239eba8f6f8f49a792a445d652d6dc
cfd616fbb15458507c6ffed079d392c6
You're not an admin
```

Εικόνα 30: Εντολή αποστολής αρχείων στο VirusTotal

Το script, ζητάει δικαιώματα διαχειριστή προκειμένου να μπορέσει να προσπελάσει όλα τα αρχεία και να τα αποστείλει στο virus total για ανάλυση. Με το πέρας της διαδικασίας, η βάση δεδομένων έχει πάρει την ακόλουθη μορφή



Εικόνα 31: Διαμόρφωση βάσης μετά την εκτέλεση του διακόπτη -uvt

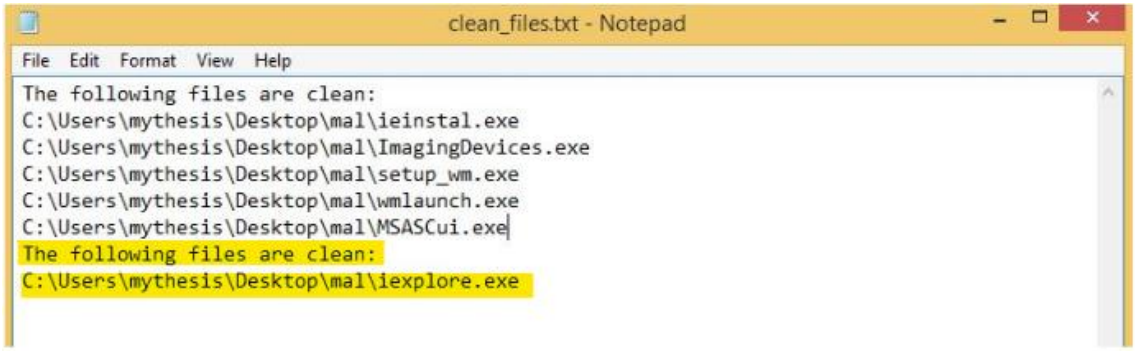
Ο διακόπτης «–rrvt» είναι υπεύθυνος για την ανάκτηση των αναφορών των αρχείων, που έγιναν upload στο virus total και απαιτεί σαν παράμετρο, το μονοπάτι της βάσης δεδομένων.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -rrvt C:\Users\mythesis\Desktop\mydb.db
Enter your API key: 62239eba8f6f8f49a792a445d652d6dccfd616fbb15458507c6ffed079d392c6
Is your VT key private (y/n)?: n
```

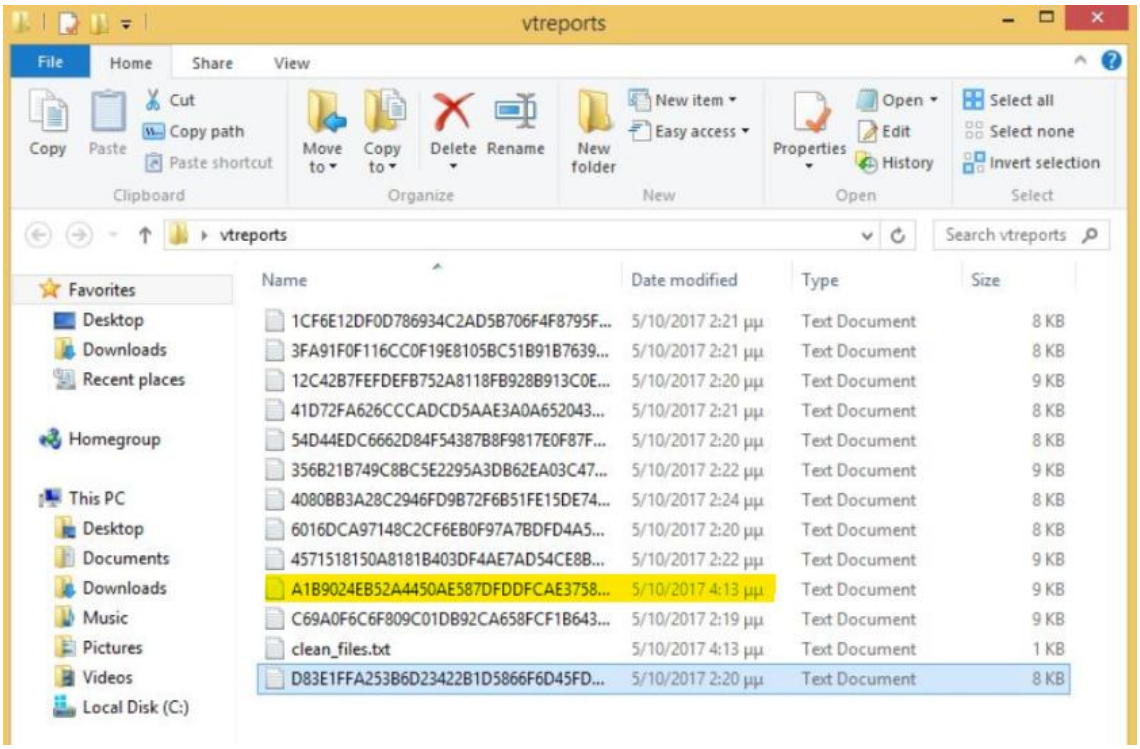
Εικόνα 32: Εντολή ανάκτησης αναφορών

Με την ολοκλήρωση της διαδικασίας, στον ήδη υπάρχων φάκελο των αναφορών (vtreports) έχουν αποθηκευτεί οι αναφορές των αρχείων και το αρχείο «clean_files.txt» έχει

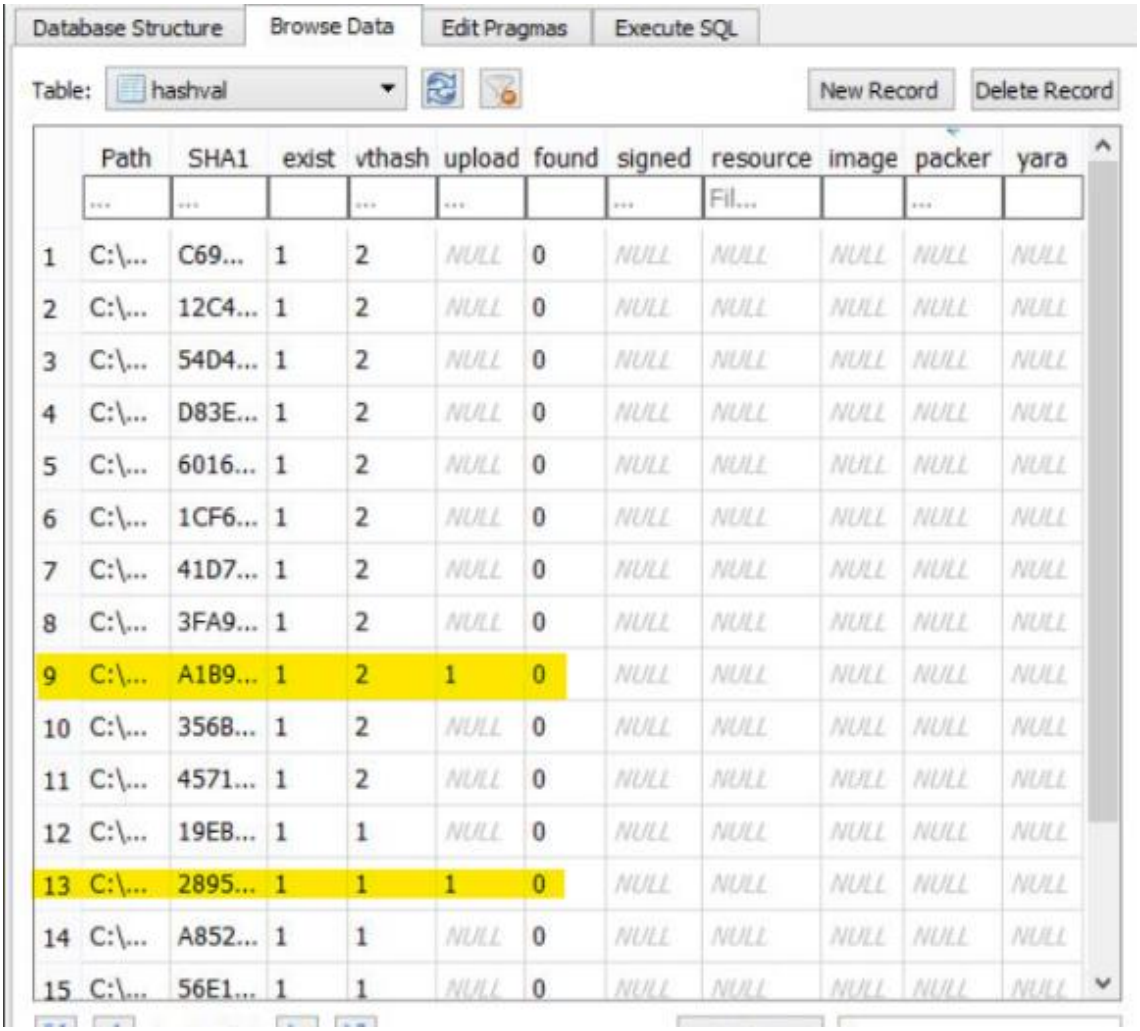
μεταβληθεί για να περιλαμβάνει τα νέα δεδομένα, καθώς και οι αντίστοιχες τιμές της βάσης έχουν ανανεωθεί.



Εικόνα 33: Διαμόρφωση των αρχείων αναφορών



Εικόνα 34: Διαμόρφωση του φακέλου αναφορών



	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	...	...		...	...		...	Fil...		...	
1	C:\...	C69...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
2	C:\...	12C4...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
3	C:\...	54D4...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
4	C:\...	D83E...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
5	C:\...	6016...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
6	C:\...	1CF6...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
7	C:\...	41D7...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
8	C:\...	3FA9...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
9	C:\...	A189...	1	2	1	0	NULL	NULL	NULL	NULL	NULL
10	C:\...	3568...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
11	C:\...	4571...	1	2	NULL	0	NULL	NULL	NULL	NULL	NULL
12	C:\...	19EB...	1	1	NULL	0	NULL	NULL	NULL	NULL	NULL
13	C:\...	2895...	1	1	1	0	NULL	NULL	NULL	NULL	NULL
14	C:\...	A852...	1	1	NULL	0	NULL	NULL	NULL	NULL	NULL
15	C:\...	56E1...	1	1	NULL	0	NULL	NULL	NULL	NULL	NULL

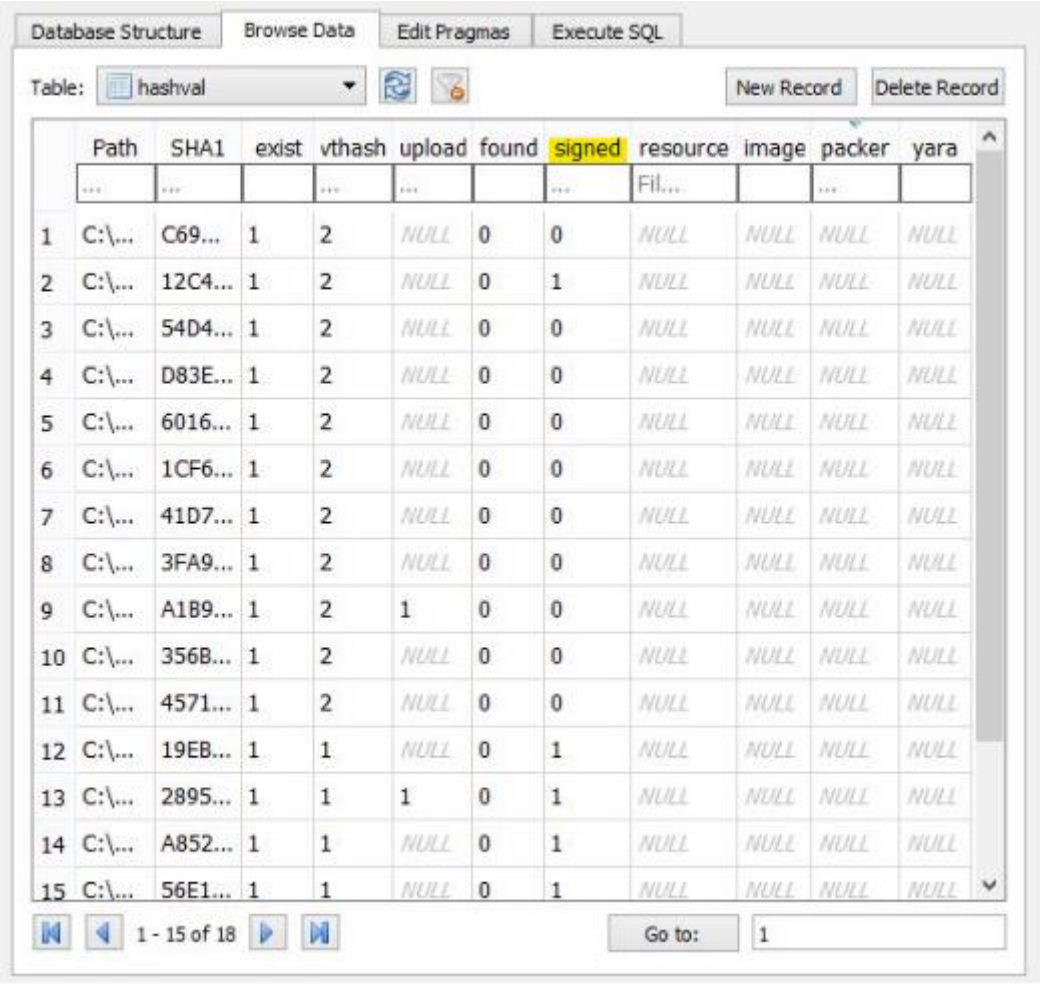
Εικόνα 35: Μετάπλαση περιεχομένων της βάσης με την εκτέλεση του διακόπτη --rrvt

Ο διακόπτης «–ds» δέχεται σαν όρισμα το μονοπάτι της βάσης και τον κατάλογο στον οποίο βρίσκετε το εργαλείο sigcheck.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -ds C:\Users\mythesis\Desktop\mydb.db E:\codenew\sigcheck.exe
```

Εικόνα 36: Εντολή ανίχνευσης ψηφιακών υπογραφών

Με την ολοκλήρωση της εκτέλεσης, το αντίστοιχο πεδίο του πίνακα έχει αποκτήσει τιμές όπως φαίνεται στο στιγμιότυπο που ακολουθεί.

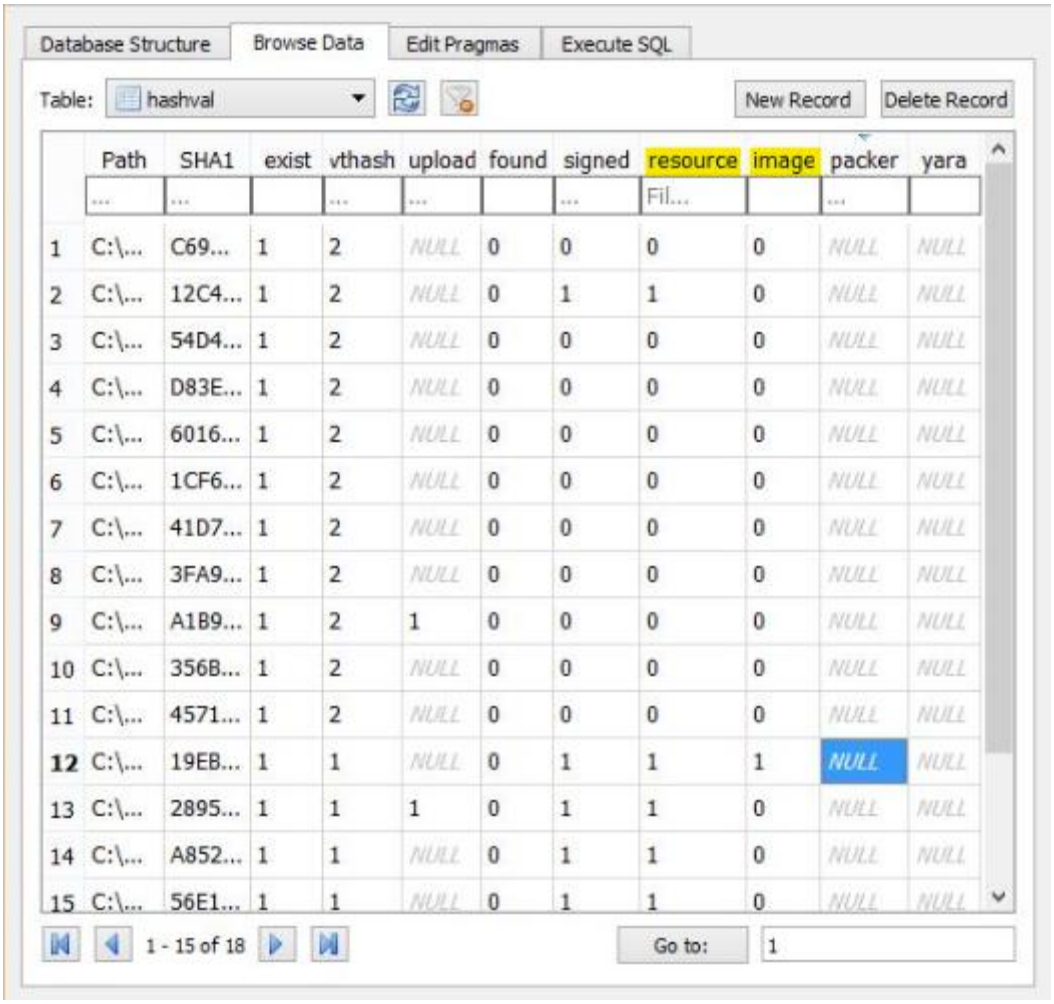


Εικόνα 37: Διαμόρφωση της βάσης μετά την εκτέλεση του διακόπτη -ds

Ο επόμενος διακόπτης είναι ο «–r», ο οποίος θέλει σαν όρισμα μόνο το path της βάσης και δίνει τιμές στα πεδία «resource» και «image» της βάσης.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -r C:\Users\mythesis\Desktop\mydb.db
```

Εικόνα 38: Εντολή ελέγχου των resources

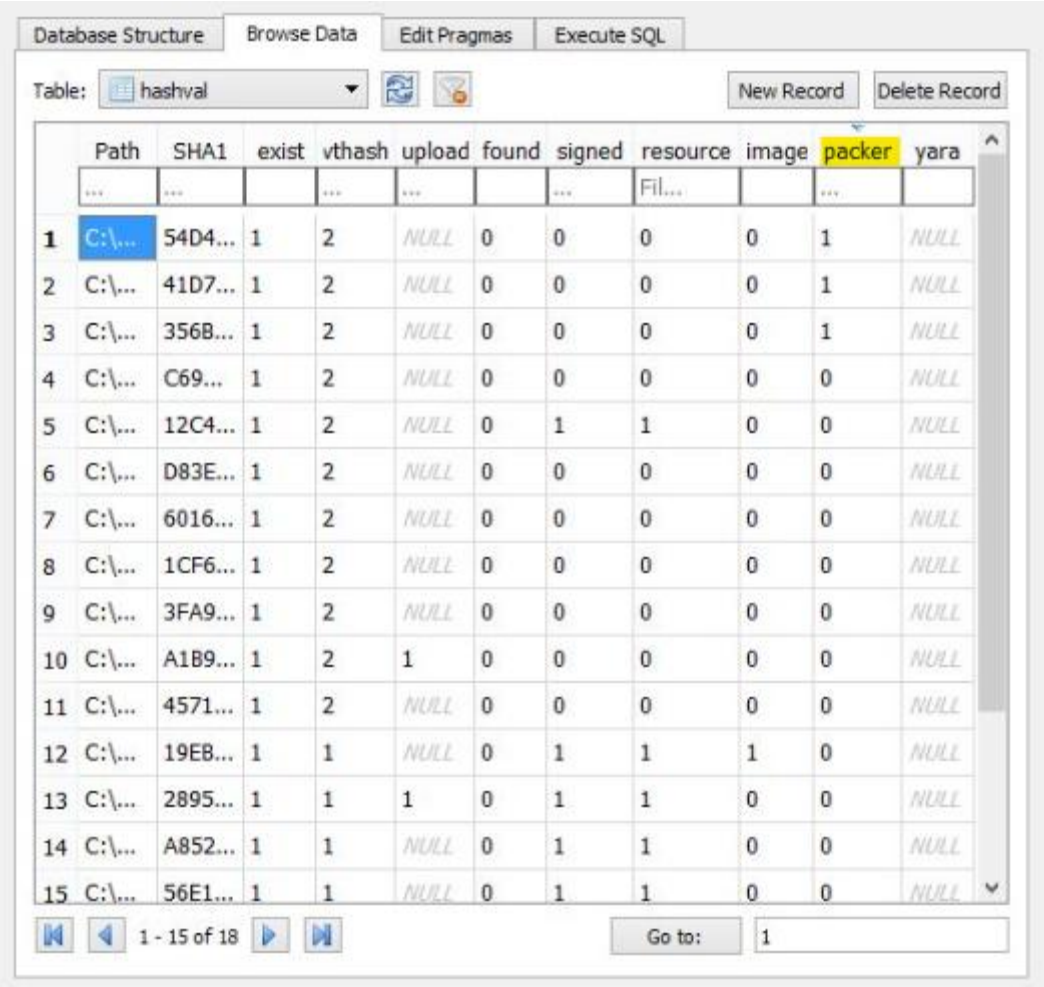


Εικόνα 39: Διαμόρφωση της βάσης μετά την εκτέλεση του διακόπτη -r

Στη συνέχεια έχουμε το διακόπτη «–P», ο οποίος δέχεται σαν παραμέτρους τα μονοπάτια της βάσης και του αρχείου, που περιέχει τους κανόνες για τις υπογραφές των packer.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -P C:\Users\mythesis\Desktop\mydb.db E:\codenew\userdb.txt
```

Εικόνα 40: Εντολή ελέγχου χρήσης packer



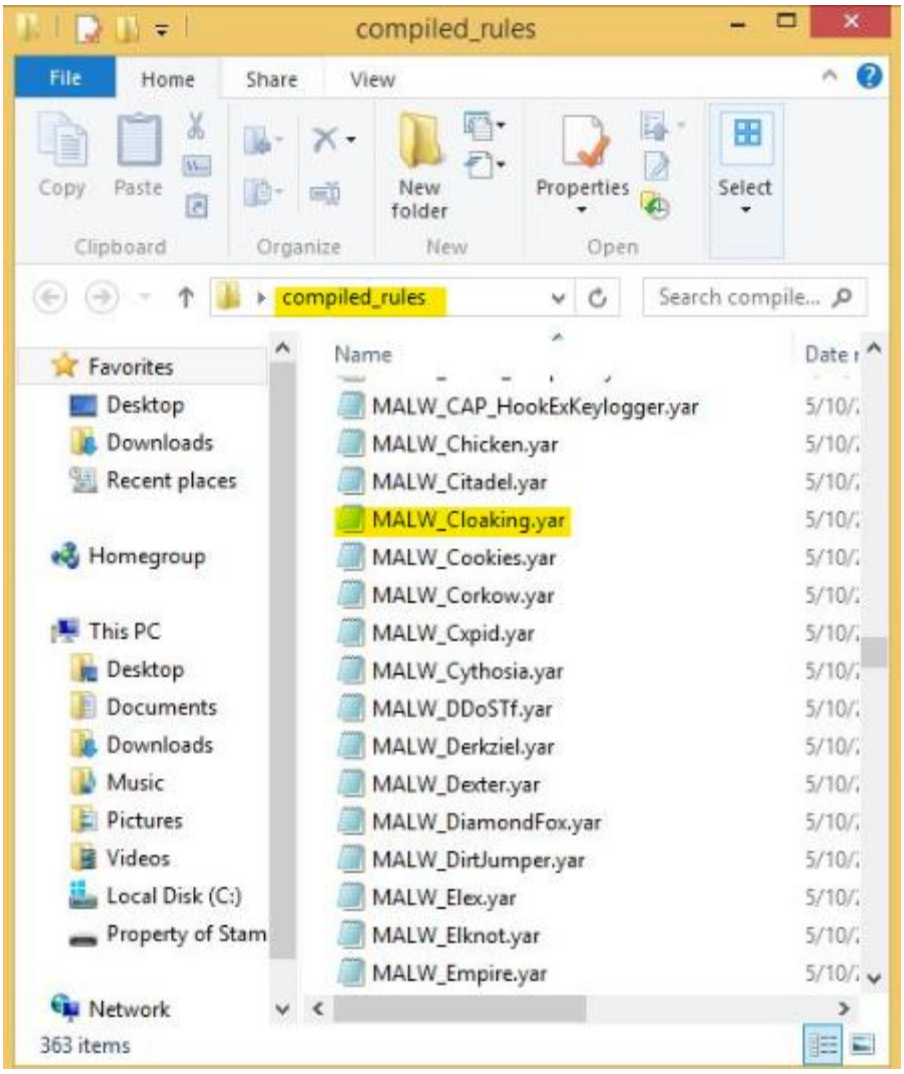
Εικόνα 41: Μετάπλαση της βάσης μετα την εκτέλεση του διακόπτη -P

Στη συνέχεια παρουσιάζεται ο διακόπτης «–yc» ο οποίος δέχεται σαν όρισμα το μονοπάτι που βρίσκονται αποθηκευμένοι οι κανόνες yara.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -yc E:\rules-master
```

Εικόνα 42: Εντολή μεταγλώττισης κανόνων yara

Με την ολοκλήρωση του βήματος, οι μεταγλωττισμένοι κανόνες αποθηκεύονται στο φάκελο «compiled_rules» που δημιουργείται στο κατάλογο εργασίας



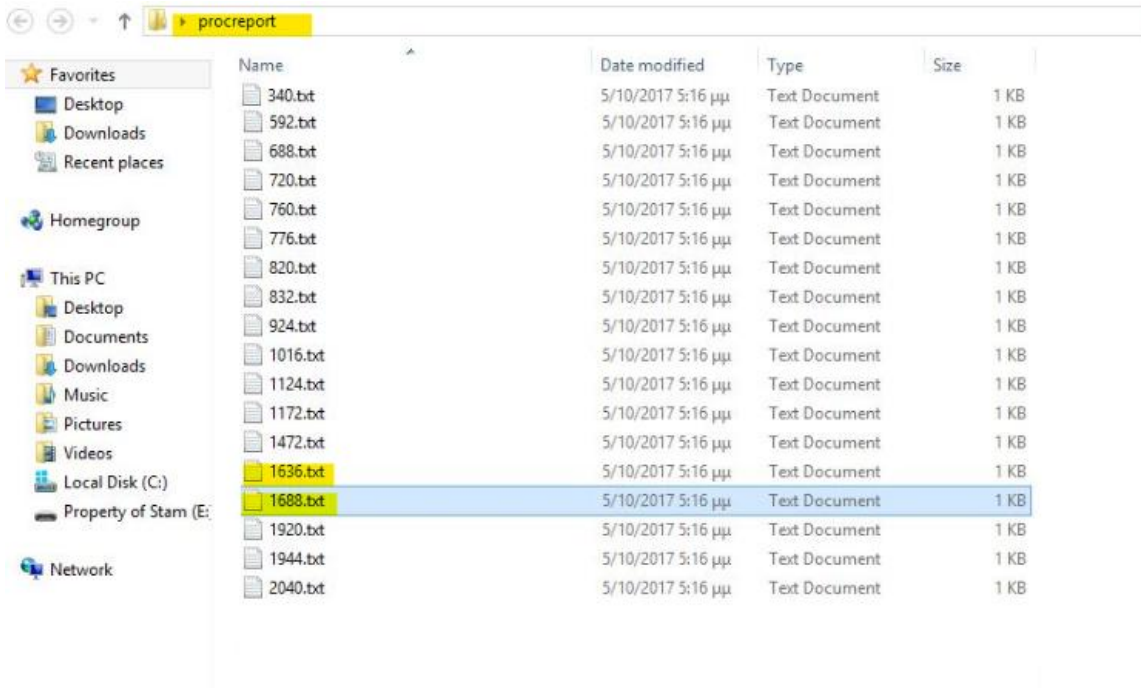
Εικόνα 43: Μεταγλωττισμένοι κανόνες yara

Στη συνέχεια, ο διακόπτης «–ym» δέχεται σαν όρισμα το μονοπάτι των μεταγλωττισμένων κανόνων και τους χρησιμοποιεί για να ελέγξει τις τρέχουσες διεργασίες.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -ym C:\Users\mythesis\Desktop\compiled_rules\memory
```

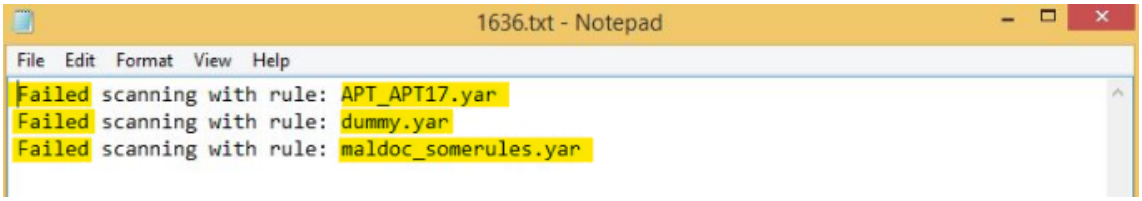
Εικόνα 44: Εντολή ελέγχου διεργασιών με κανόνες yara

Τα αποτελέσματα αποθηκεύονται στο φάκελο «procreport» και για κάθε διεργασία δημιουργείται μια αναφορά

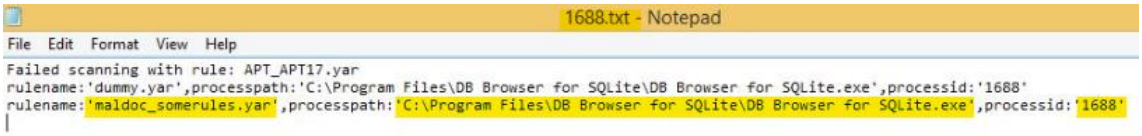


Εικόνα 45: Φάκελος αναφορών ελέγχου διεργασιών

Τα περιεχόμενα των αναφορών έχουν την ακόλουθη μορφή. Στο πρώτο στιγμιότυπο, βλέπουμε μια διεργασία στην οποία δεν εντοπίστηκαν οι συγκεκριμένοι κανόνες, σε αντίθεση με το δεύτερο στιγμιότυπο στο οποίο η συγκεκριμένη διεργασία έχει ταυτοποιηθεί με τους κανόνες «maldoc_somerules.yar» και «dummy.yar». Στα περιεχόμενα της αναφοράς εκτός από τον κανόνα που ταυτοποιήθηκε, μπορούμε επίσης να δούμε το μονοπάτι εκκίνησης της συγκεκριμένης διεργασίας αλλά και το process id της.



Εικόνα 46: Αποτελέσματα μη ταυτοποίησης κανόνων

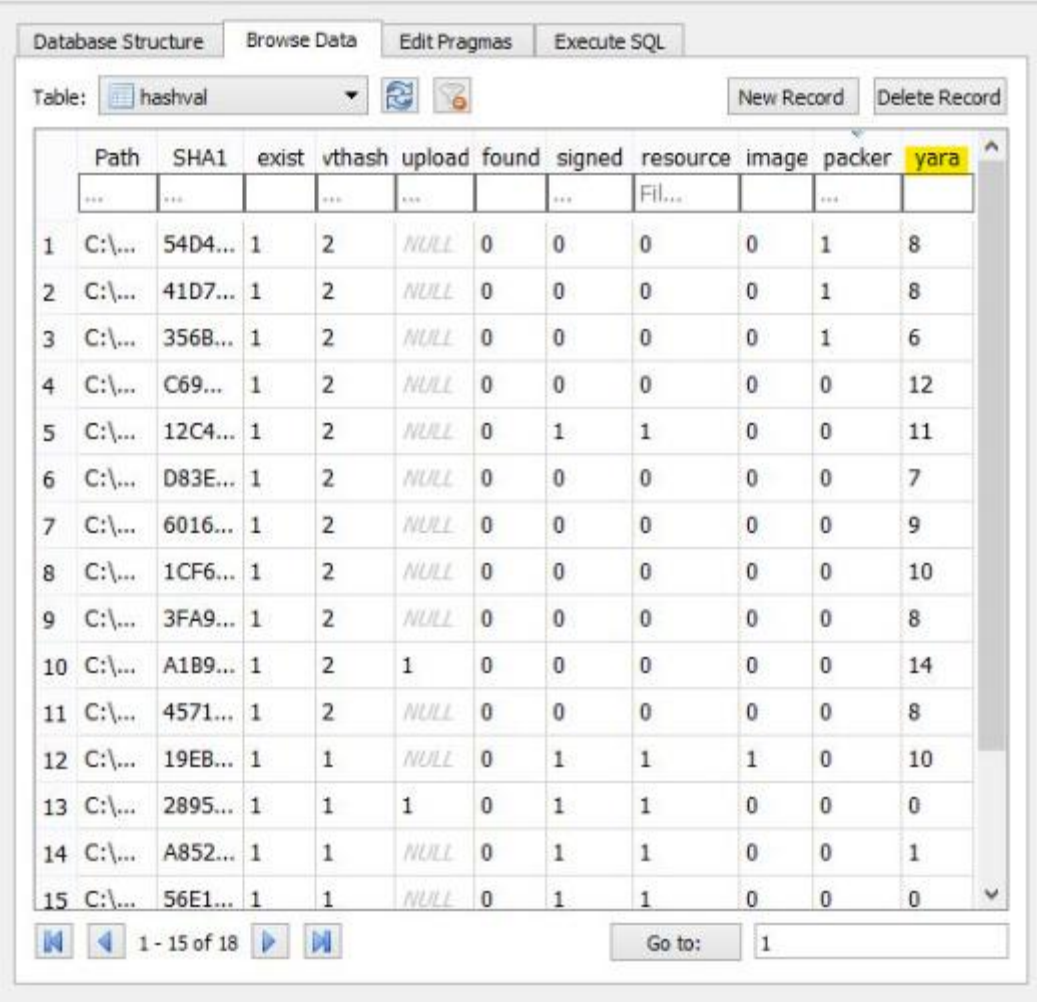


Εικόνα 47: Αποτελέσματα ταυτοποιημένων κανόνων

Ο διακόπτης που γεμίζει την τελευταία στήλη της βάσης δεδομένων είναι ο «–ys», ο οποίος θέλει σαν όρισμα τα μονοπάτια της βάσης δεδομένων και του καταλόγου με τους αποθηκευμένους κανόνες yara

Εικόνα 48: Εντολή ελέγχου αρχείων με κανόνες yara

Κάθε αρχείο ξεχωριστά, ελέγχεται με όλους τους κανόνες που έχουν μεταγλωττιστεί. Ένας μετρητής συγκρατεί των αριθμό των κανόνων που ταυτοποιήθηκαν με το συγκεκριμένο αρχείο και στη συνέχεια η τιμή του μετρητή καταχωρείται στη βάση δεδομένων.



Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: hashval — New Record — Delete Record

	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	...	...		...	...		...	Fil...		...	
1	C:\...	54D4...	1	2	NULL	0	0	0	0	1	8
2	C:\...	41D7...	1	2	NULL	0	0	0	0	1	8
3	C:\...	356B...	1	2	NULL	0	0	0	0	1	6
4	C:\...	C69...	1	2	NULL	0	0	0	0	0	12
5	C:\...	12C4...	1	2	NULL	0	1	1	0	0	11
6	C:\...	D83E...	1	2	NULL	0	0	0	0	0	7
7	C:\...	6016...	1	2	NULL	0	0	0	0	0	9
8	C:\...	1CF6...	1	2	NULL	0	0	0	0	0	10
9	C:\...	3FA9...	1	2	NULL	0	0	0	0	0	8
10	C:\...	A1B9...	1	2	1	0	0	0	0	0	14
11	C:\...	4571...	1	2	NULL	0	0	0	0	0	8
12	C:\...	19EB...	1	1	NULL	0	1	1	1	0	10
13	C:\...	2895...	1	1	1	0	1	1	0	0	0
14	C:\...	A852...	1	1	NULL	0	1	1	0	0	1
15	C:\...	56E1...	1	1	NULL	0	1	1	0	0	0

1 - 15 of 18 — Go to: 1

Εικόνα 49: Διαμόρφωση βάσης μετα την εκτέλεση του διακόπτη -ys

Ο διακόπτης «–rs» δέχεται σαν είσοδο το πλήρες path της βάσης. Ο χρήστης επιλέγει τα δεδομένα που τον ενδιαφέρουν και σχηματίζεται αυτόματα το ερώτημα στη βάση.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -rs C:\Users\mythesis\Desktop\mydb.db
Answer the following questions with y or n, leave blank if you dont care about that value
Do u want to get the files that already existed in the db (y/n): y
Characterized as malicious from virus total : y
Uploaded to virus total  : n
Found in NSRL : t
Found in NSRL :
Are signed:
have resources :
Have image :
Are packed:
Have hits from yara rules : Y
Enter a name for report file: report
PS C:\Users\mythesis\Desktop>
```

Εικόνα 50: Παράδειγμα χρήσης του διακόπτη -rs

Στο παραπάνω στιγμιότυπο, ζητήσαμε να μας επιστραφούν τα μονοπάτια όλων των αρχείων τα οποία χαρακτηρίστηκαν ως κακόβουλα από το VirusTotal, χωρίς να χρειαστεί να τα ανεβάσουμε εμείς και έχουν κάνει match τουλάχιστον σε ένα κανόνα yara.Τα αποτελέσματα γράφονται σε ένα αρχείο «.csv» με την ακόλουθη μορφή.

	A	B	C	D	E	F	G	H	I	J	K
1	Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
2	C:\Users\mythesis\Desktop\mal\Ransomware.Cerber\c	C69A0F6C6F809C01DB92CA658FCF1B643391A2B7	1	2	None	0	0	0	0	0	12
3	C:\Users\mythesis\Desktop\mal\Trojan.Win32.Bechiro	12C42B7FEFDEFB752A8118FB928B913C0EF7562D	1	2	None	0	1	1	0	0	11
4	C:\Users\mythesis\Desktop\mal\Win32 Turla\decrypted	54D44EDC6662D84F54387B8F9817E0F87F9FE70E	1	2	None	0	0	0	0	1	8
5	C:\Users\mythesis\Desktop\mal\Win32 Turla\decrypted	D83E1FFA253B6D23422B1D5866F6D45FD09A08C3	1	2	None	0	0	0	0	0	7
6	C:\Users\mythesis\Desktop\mal\Win32 Turla\decrypted	6016DCA97148C2CF6EB0F97A7BDFD4A523BD0241	1	2	None	0	0	0	0	0	9
7	C:\Users\mythesis\Desktop\mal\Win32 Turla\decrypted	1CF6E12DF0D786934C2AD5B706F4F8795F776C8B	1	2	None	0	0	0	0	0	10
8	C:\Users\mythesis\Desktop\mal\Win32 Turla\decrypted	41D72FA626CCCADCD5AAE3A0A652043E4A4F7B4A	1	2	None	0	0	0	0	1	8
9	C:\Users\mythesis\Desktop\mal\Win32 Turla\decrypted	3FA91F0F116CC0F19E8105BC51B91B7639605663	1	2	None	0	0	0	0	0	8
10	C:\Users\mythesis\Desktop\mal\dumped.exe	A1B9024EB52A4450AE587DFDDFCAE37581DAA5E3	1	2	1	0	0	0	0	0	14
11	C:\Users\mythesis\Desktop\mal\fax_390392029_0725	356B21B749C8BC5E2295A3DB62EA03C47CB4C1CF	1	2	None	0	0	0	0	1	6
12	C:\Users\mythesis\Desktop\mal\file_4571518150a8181	4571518150A8181B403DF4AE7AD54CE8B16DED0C	1	2	None	0	0	0	0	0	8
13											
14											

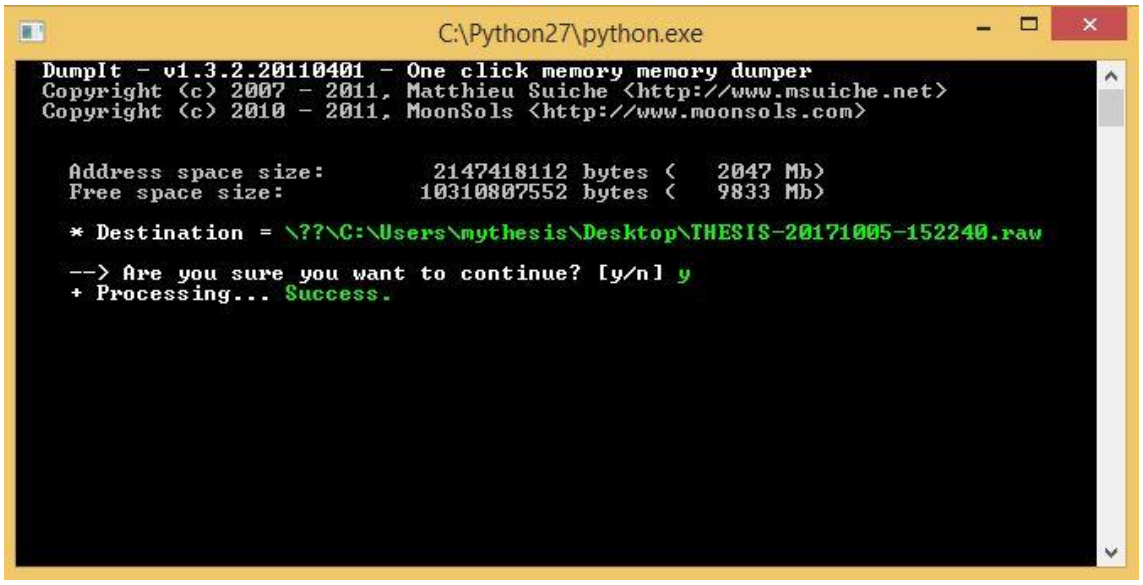
Εικόνα 51: Αποτελέσματα χρήσης του διακόπτη -rs

Με χρήση του διακόπτη «–MD» δίνεται η δυνατότητα να γίνει dump η μνήμη σε ένα αρχείο. Το όρισμα του διακόπτη είναι το μονοπάτι του εργαλείου DumpIt όπως φαίνεται και στο παράδειγμα που ακολουθεί.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -MD C:\Users\mythesis\Downloads\DumpIt.exe
You're not an admin.
```

Εικόνα 52: Εντολή εκτέλεσης DumpIt

Το script αυτόματα κάνει elevate τα δικαιώματα προκειμένου να μπορέσει να εκτελέσει το εργαλείο και το «.raw» αρχείο που δημιουργείται αποθηκεύεται στον τρέχον κατάλογο εργασίας.



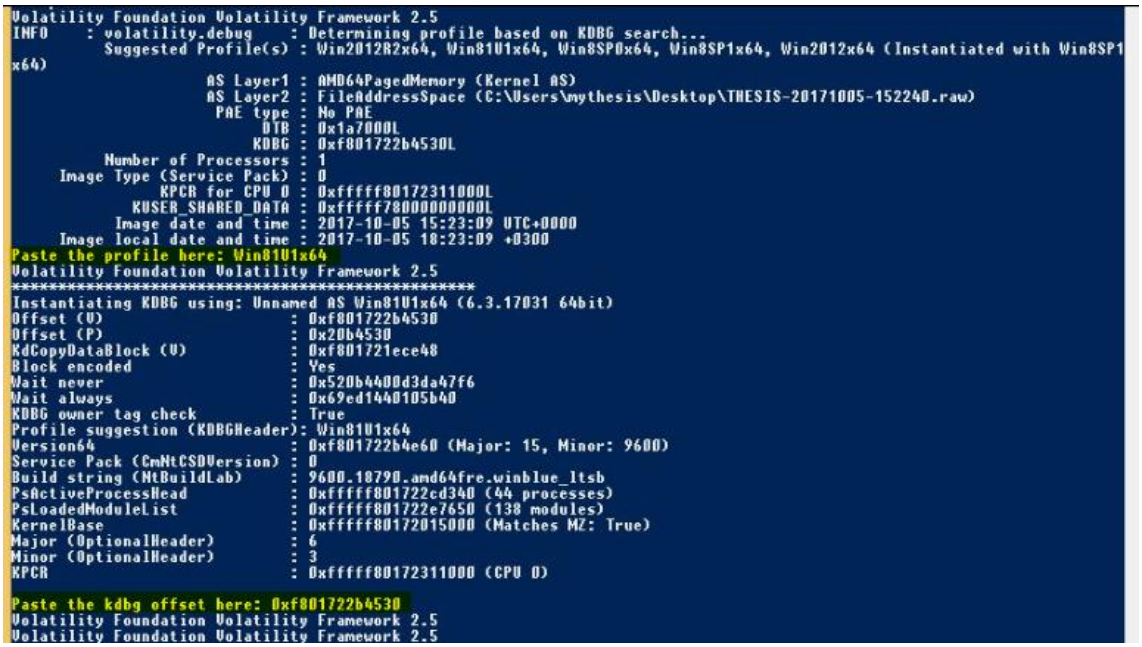
Εικόνα 53: Στιγμιότυπο εκτέλεσης DumpIt

Ο διακόπτης «–M» δέχεται σαν όρισμα το μονοπάτι του «.raw» αρχείου και το πλήρες path του εργαλείου Volatility, που χρησιμοποιείται για την ανάλυση του αρχείου της μνήμης.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -M E:\codenew\volatility-2.5.standalone.exe C:\Users\mythesis\De
sktop\THESIS-20171005-152240.raw
```

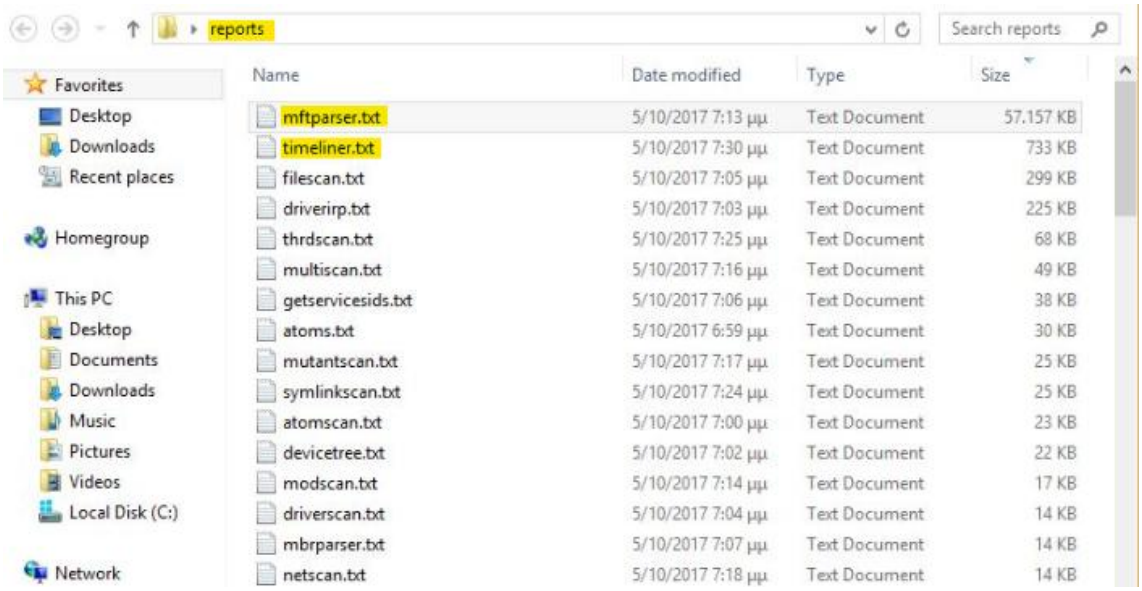
Εικόνα 54: Εντολή ανάλυσης μνήμης

Κατά την εκτέλεσή του, εκτελεί τα πρόσθετα «imageinfo» και «kdbgscan» και ζητάει από το χρήστη να επιλέξει αυτά που αντιστοιχούν στην εικόνα της μνήμης που θέλει να αναλύσει όπως στο ακόλουθο παράδειγμα και στην συνέχεια ξεκινάει η αυτοματοποιημένη εκτέλεση των υπολοίπων προσθέτων.



Εικόνα 55: Στιγμιότυπο ανάλυσης μνήμης

Τα αποτελέσματα αποθηκεύονται σε ένα φάκελο που δημιουργείται αυτόματα με το όνομα «reports» και περιέχει όλες τις αναφορές των προσθέτων που εκτελέστηκαν

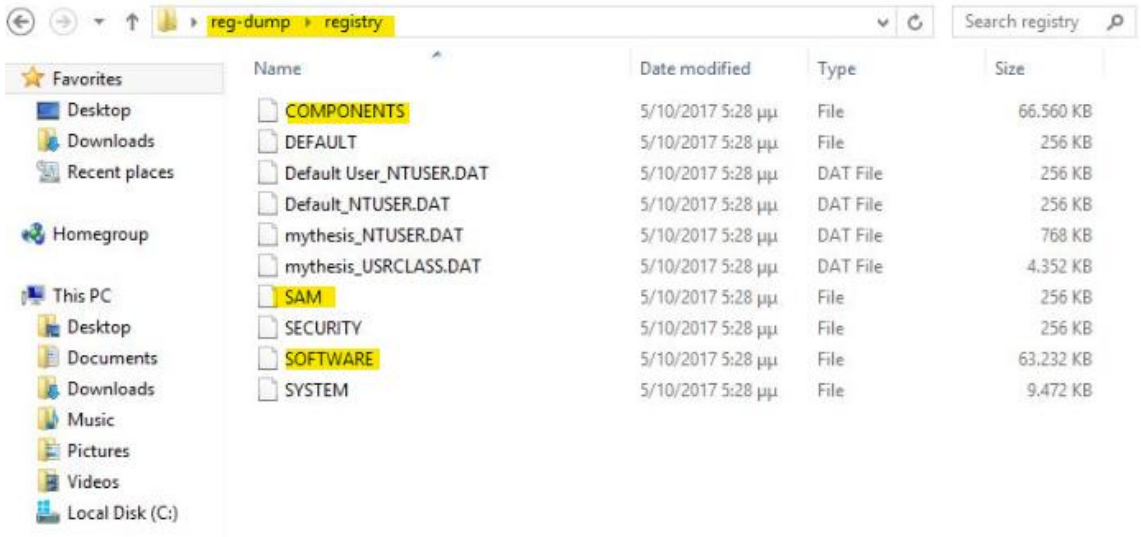


Εικόνα 56: Φάκελος αναφορών volatility

Ο διακόπτης «–RD» δέχεται σαν όρισμα το πλήρες μονοπάτι του εργαλείου που θα χρησιμοποιήσουμε για να αντιγράψουμε τις κυψέλες της registry(Forecopy). Όπως και στη μνήμη έτσι και στη registry το script ζητάει προνόμια διαχειριστή προκειμένου να εκτελεστεί.

Εικόνα 57: Εντολή εκτέλεσης Forecopy

Τα αντιγραμμένα registry hives αποθηκεύονται στο φάκελο «reg-dump» που δημιουργείται αυτόματα στον κατάλογο εργασίας.



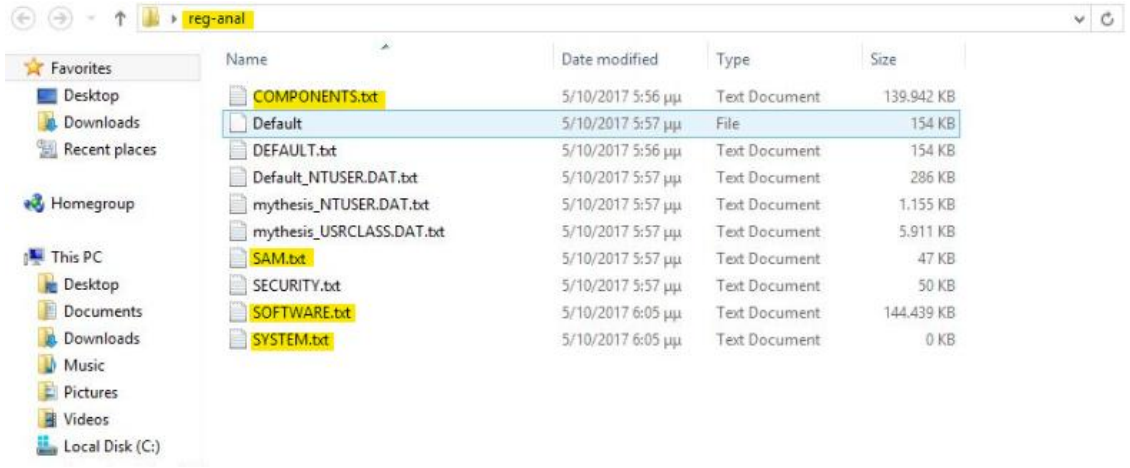
Εικόνα 58: Αποθήκευση registry hives

Για την ανάλυση τους χρησιμοποιείται ο διακόπτης «–R», ο οποίος θέλει σαν παραμέτρους τον κατάλογο που έχουν αποθηκευτεί τα registry hives και το μονοπάτι του εργαλείου που θα χρησιμοποιηθεί για την ανάλυσή τους.

```
PS C:\Users\mythesis\Desktop> python .\thesis.py -R E:\codenew\RegRipper2.8-master\rip.exe C:\Users\mythesis\Desk
top\reg-dump\registry
E:\codenew\RegRipper2.8-master
Parsed Plugins file.
Launching baseline v.20130211
baseline complete.
Launching del v.20140807
del complete.
del_tln complete.
Launching fileless v.20150110
Use of uninitialized value $list in pattern match (m//) at PERL2EXE_STORAGE/utf8_heavy.pl line 399.
fileless complete.
Launching findexes v.20090728
findexes complete.
Launching installedcomp v.20130911
installedcomp complete.
Launching installer v.20120917
installer complete.
Launching malware v.20161210
malware complete.
Launching null v.20160119
null complete.
Launching regtime v.20080324
regtime complete.
Launching regtime_tln v.20080324
regtime_tln complete.
Launching rlo v.20130904
rlo complete.
Launching sizes v.20150527
sizes complete.
Launching uninstall v.20140512
uninstall complete.
Launching uninstall v.20120523
uninstall_tln complete.
Launching wallpaper v.20080810
Software\Microsoft\Windows\CurrentVersion\Explorer\Wallpaper\MRU not found.
wallpaper complete.
Parsed Plugins file.
Launching baseline v.20130211
baseline complete.
Launching del v.20140807
del complete.
del_tln complete.
Launching fileless v.20150110
```

Εικόνα 59: Εντολή ανάλυσης των αρχείων της registry

Οι αναφορές της ανάλυσης αποθηκεύονται στο φάκελο «reg_anal» που δημιουργείται στον κατάλογο από τον οποίο εκτελείται το script, με ονόματα αντίστοιχα με το αρχείο της registry που αφορούν.



Εικόνα 60: Φάκελος αναφορών από την ανάλυση της registry

Στο σημείο αυτό έχουν επεξηγηθεί όλοι οι διαθέσιμοι διακόπτες με τις λειτουργίες τους. Στο κεφάλαιο που ακολουθεί θα διεξαχθεί ένα πείραμα το οποίο περιλαμβάνει τη μόλυνση ενός windows 8 υπολογιστή με γνωστά κακόβουλα λογισμικά, προκειμένου να ελέγξουμε την αποτελεσματικότητα εντοπισμού τους από το εργαλείο που αναπτύχθηκε.

5 Πειραματική Αξιολόγηση

Για την πειραματική αξιολόγηση, δημιουργήθηκε ένα εικονικό μηχάνημα με τέσσερα gigabyte μνήμη ram και εξήντα gigabyte σκληρό δίσκο, το οποίο λειτουργεί με λογισμικό windows 8.1x64. Τα επιπλέον λογισμικά που εγκαταστάθηκαν είναι η γλώσσα python 2.7x64 η οποία είναι απαραίτητη για την εκτέλεση του εργαλείου και ο DB Browser for SQLite ο οποίος χρησιμοποιείται για την οπτικοποίηση των δεδομένων που καταχωρούνται στη βάση.

Το κακόβουλο λογισμικό ανακτήθηκε από το πρότζεκτ «theZoo», το οποίο αποτελεί μια open source συλλογή από κακόβουλα λογισμικά, που σκοπό έχει να επιτρέψει την μελέτη και την ανάλυση τους καθαρά για ακαδημαϊκούς σκοπούς [25]. Τα κακόβουλα λογισμικά τα οποία αποθηκεύτηκαν στον υπολογιστή είναι τα ακόλουθα:

- Artemis
- Ransomware.Cerber
- Trojan.Win32.Bechiro.BCD
- Win32.Turla
- 131.exe
- Dumped.exe
- fax_390392029_072514.exe
- file_4571518150a8181b403df4ae7ad54ce8b16ded0c.exe

Συνολικά τα οκτώ αυτά κακόβουλα λογισμικά αποτελούνται από δεκατρία «.exe» και «.dll» αρχεία.

Η διαδικασία του πειράματος ξεκίνησε δίνοντας σαν φακέλους ρίζα τους «C:\Windows\System32» και «C:\Users» και χωρίς να έχουμε εκτελέσει κάποιο από τα malware. Ο συνολικός όγκος των δεδομένων που έπρεπε να αναλυθούν ήταν 4067 αρχεία. Οι διακόπτες εκτελέστηκαν με τη σειρά που αναλύθηκαν στο προηγούμενο κεφάλαιο. Αρχικά, υπολογίσαμε τα hash values όλων των αρχείων αυτών και τα αποθηκεύσαμε στη βάση δεδομένων. Στο δεύτερο βήμα, ελέγξαμε ποια από τα παραπάνω hash values υπάρχουν στη βάση της NSRL, με συνολικά 133 αρχεία να ταυτοποιούνται και να εξαιρούνται των επόμενων διαδικασιών.

Στη συνέχεια, ελέγξαμε τα αρχεία στη βάση του VirusTotal χρησιμοποιώντας το hash value τους, οι αναφορές του οποίου υπέδειξαν συνολικά 23 κακόβουλα αρχεία στον

υπολογιστή. Δεν χρειάστηκε να ανεβάσουμε κάποιο αρχείο καθώς όλες οι τιμές της συνάρτησης κατακερματισμού υπήρχαν ήδη στη βάση του VirusTotal.

Στη συνέχεια, εκτελέσαμε τους διακόπτες ελέγχου για ψηφιακές υπογραφές, για χρήση packer, για έλεγχο των resources και υλοποιήσαμε ένα πρώτο έλεγχο χρησιμοποιώντας κανόνες yara τόσο σε αρχεία όσο και σε διεργασίες. Οι κανόνες yara ανακτήθηκαν από μια μεγάλη αποθήκη κανόνων που είναι διαθέσιμη στο GitHub [26].

Στο σημείο αυτό παραθέτουμε τα αποτελέσματα της εκτέλεσης του εργαλείου με στιγμιότυπα.

Name	Date modified	Type	Size
0EE26ADFE8F75D4A05D940629D9210C1C...	15/10/2017 7:02 μμ	Text Document	8 KB
1CF6E12DF0D786934C2AD5B706F4F8795F...	15/10/2017 7:02 μμ	Text Document	8 KB
1FB37E98D3ECE458C00466F6E0E3F0FBE0...	15/10/2017 7:02 μμ	Text Document	8 KB
2E184807B383DDA251D249BF421D659298...	15/10/2017 7:02 μμ	Text Document	8 KB
3FA91F0F116CC0F19E8105BC51B91B7639...	15/10/2017 6:59 μμ	Text Document	8 KB
08B9F5874AD1DC3EE1093C9CD08737645...	15/10/2017 6:59 μμ	Text Document	8 KB
8DEA5EACA7A3C4F139A010F6336C893C...	15/10/2017 7:00 μμ	Text Document	7 KB
9D78DE808B60DF9CC85AB993F477797D9...	15/10/2017 7:00 μμ	Text Document	7 KB
12C42B7FEFDEFB752A8118FB928B913C0E...	15/10/2017 7:00 μμ	Text Document	9 KB
41D72FA626CCCADCD5AAE3A0A652043...	15/10/2017 7:00 μμ	Text Document	9 KB
54D44EDC6662D84F54387B8F9817E0F87F...	15/10/2017 7:00 μμ	Text Document	8 KB
086E50434459760924F7BF8B854CAC1427...	15/10/2017 7:00 μμ	Text Document	7 KB
294AA1ABF2DE840F938AF347E06FA15364...	15/10/2017 7:00 μμ	Text Document	8 KB
356B21B749C8BC5E2295A3DB62EA03C47...	15/10/2017 7:00 μμ	Text Document	9 KB
2132A26AD90D5726229398105B9D2A5FB...	15/10/2017 7:00 μμ	Text Document	7 KB
2246FA0E873D0C4110A015789ED10F4F94...	15/10/2017 7:00 μμ	Text Document	7 KB
4080BB3A28C2946FD9B72F6B51FE15DE74...	15/10/2017 7:00 μμ	Text Document	5 KB
5741AF8CC8A4DED2780CB3F37CA29A57...	15/10/2017 7:00 μμ	Text Document	7 KB
6016DCA97148C2CF6EB0F97A7BDFD4A5...	15/10/2017 7:00 μμ	Text Document	9 KB
4571518150A8181B403DF4AE7AD54CE8B...	15/10/2017 7:01 μμ	Text Document	9 KB
A1B9024EB52A4450AE587DFDDFCAE3758...	15/10/2017 7:02 μμ	Text Document	9 KB
C69A0F6C6F809C01DB92CA658FCF1B643...	15/10/2017 7:02 μμ	Text Document	9 KB
CE527E4B98099F7CE14518A029EB334BD9...	15/10/2017 7:02 μμ	Text Document	7 KB
clean_files.txt	15/10/2017 7:02 μμ	Text Document	444 KB
D83E1FFA253B6D23422B1D5866F6D45FD...	15/10/2017 7:02 μμ	Text Document	8 KB

Εικόνα 61: Περιεχόμενα φακέλου αναφορών

Όπως αναφέρθηκε ήδη το VirusTotal χαρακτήρισε 23 αρχεία ως κακόβουλα. Μέσα σε αυτά βρίσκονται όλα τα γνωστά malware που είχαμε «κατεβάσει» αλλά και κάποια αρχεία «.dll» του συστήματος τα οποία σημείωσαν πολύ χαμηλό detection rate.

	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	Filter	Filter	Filter	2	Filter	0	Filter	Filter	Filter	Filter	Filter
1	C:\Windows\System32\NlsLe...	2132A26AD90...	2	2	NULL	0	1	1	0	1	4
2	C:\Windows\System32\shells...	086E5043445...	2	2	NULL	0	1	1	0	1	4
3	C:\Users\mythesis\Desktop\...	356B21B749C...	NULL	2	NULL	0	0	0	0	1	5
4	C:\Users\mythesis\Desktop\...	D83E1FFA253...	NULL	2	NULL	0	0	0	0	0	6
5	C:\Users\mythesis\AppData\...	2246FA0E873...	NULL	2	NULL	0	0	0	0	1	7
6	C:\Users\mythesis\Desktop\...	54D44EDC666...	NULL	2	NULL	0	0	0	0	1	7
7	C:\Users\mythesis\Desktop\...	41D72FA626C...	NULL	2	NULL	0	0	0	0	1	7
8	C:\Users\mythesis\Desktop\...	3FA91F0F116...	NULL	2	NULL	0	0	0	0	1	7
9	C:\Users\mythesis\Desktop\...	4571518150A...	NULL	2	NULL	0	0	0	0	0	7
10	C:\Users\mythesis\Desktop\r...	CE527E4B980...	NULL	2	NULL	0	0	0	0	0	7
11	C:\Windows\System32\PrintI...	0EE26ADFE8F...	2	2	NULL	0	1	1	1	1	7
12	C:\Windows\System32\msoe...	2E184807B38...	2	2	NULL	0	1	1	0	0	8
13	C:\Users\mythesis\Desktop\...	6016DCA9714...	NULL	2	NULL	0	0	0	0	0	9
14	C:\Windows\System32\dnsap...	1FB37E98D3E...	2	2	NULL	0	1	1	0	0	9
15	C:\Windows\System32\Multi...	294AA1ABF2D...	2	2	NULL	0	1	1	1	1	9
16	C:\Users\mythesis\Desktop\...	1CF6E12DF0D...	NULL	2	NULL	0	0	0	0	1	10
17	C:\Users\mythesis\Desktop\...	0889F5874AD...	NULL	2	NULL	0	0	1	0	0	11
18	C:\Users\mythesis\Desktop\	12C42B7EEED	NULL	2	NULL	0	1	1	0	0	11

1 - 18 of 23 Go to: 1

Εικόνα 62: Πλήθος κακόβουλων σύμφωνα με το VT

Εικόνα 63: Στιγμιότυπο εσφαλμένης θετικής ένδειξης

Από το συνολικό όγκο των 3934 αρχείων που αναλύθηκαν 3877 είχαν ψηφιακή υπογραφή (Εικόνα 64) μόνο 77 αρχεία δεν εμφάνιζαν πληροφορίες σχετικά με το copyright στις λεπτομέρειές τους (Εικόνα 65) ενώ από 3402 αρχεία δεν μπορέσαμε να εξάγουμε το icon (Εικόνα 66). Συνολικά 1595 αρχεία ταυτοποιήθηκαν με υπογραφές κάποιου γνωστού packer (Εικόνα 67) ενώ όλα τα αρχεία σημείωσαν matches με τουλάχιστον ένα κανόνα yara, κυρίως λόγο του πλήθους των κανόνων που χρησιμοποιήσαμε (Εικόνα 68)

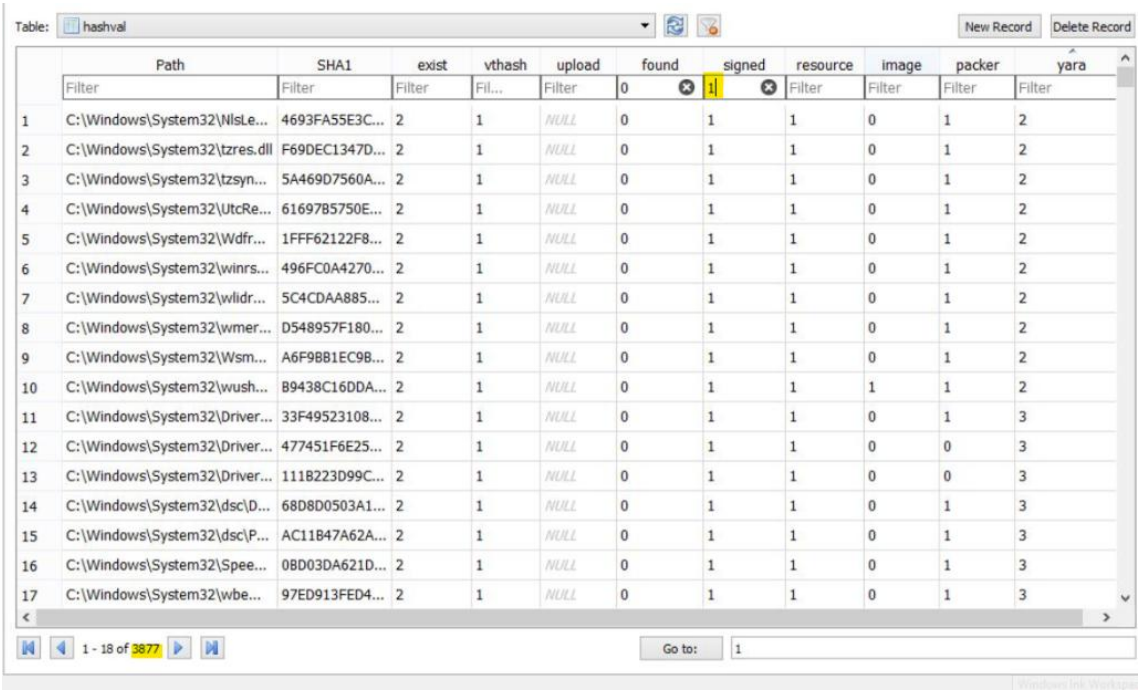
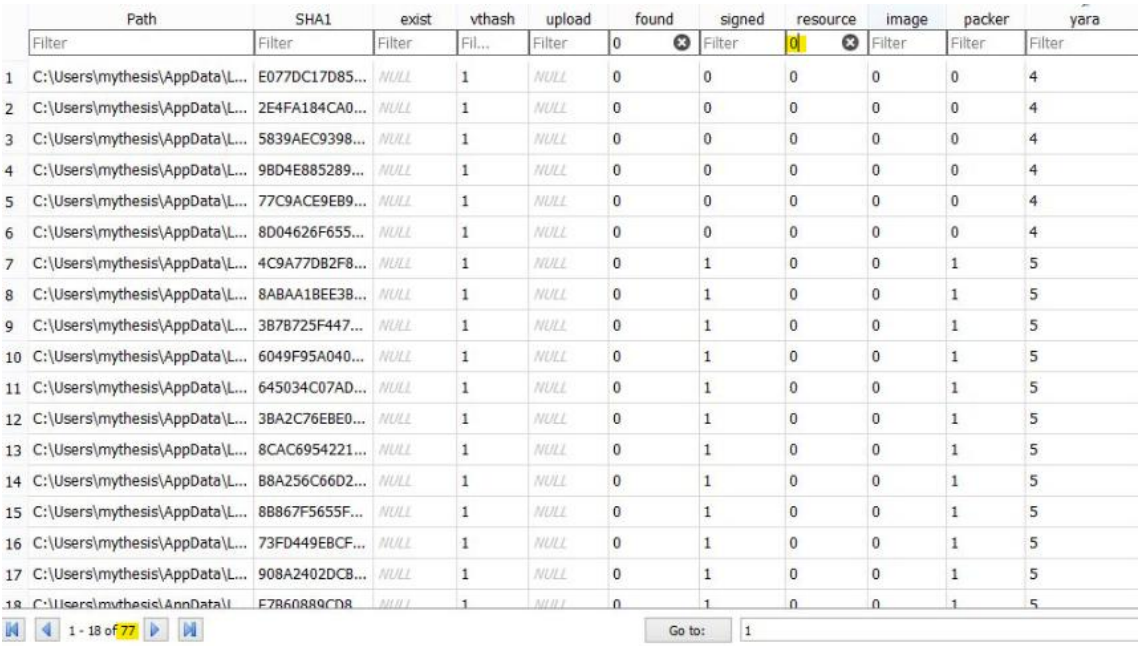


Table: hashval — New Record — Delete Record

	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	Filter	Filter	Filter	Fil...	Filter	0	1	Filter	Filter	Filter	Filter
1	C:\Windows\System32\NlsLe...	4693FA55E3C...	2	1	NULL	0	1	1	0	1	2
2	C:\Windows\System32\tzres.dll	F69DEC1347D...	2	1	NULL	0	1	1	0	1	2
3	C:\Windows\System32\tzsyn...	5A469D7560A...	2	1	NULL	0	1	1	0	1	2
4	C:\Windows\System32\UtcRe...	61697B5750E...	2	1	NULL	0	1	1	0	1	2
5	C:\Windows\System32\Wdfr...	1FFF62122F8...	2	1	NULL	0	1	1	0	1	2
6	C:\Windows\System32\winrs...	496FC0A4270...	2	1	NULL	0	1	1	0	1	2
7	C:\Windows\System32\wlidr...	5C4CDAA885...	2	1	NULL	0	1	1	0	1	2
8	C:\Windows\System32\wmer...	D548957F180...	2	1	NULL	0	1	1	0	1	2
9	C:\Windows\System32\Wsm...	A6F9BB1EC9B...	2	1	NULL	0	1	1	0	1	2
10	C:\Windows\System32\wush...	B9438C16DDA...	2	1	NULL	0	1	1	1	1	2
11	C:\Windows\System32\Driver...	33F49523108...	2	1	NULL	0	1	1	0	1	3
12	C:\Windows\System32\Driver...	477451F6E25...	2	1	NULL	0	1	1	0	0	3
13	C:\Windows\System32\Driver...	111B223D99C...	2	1	NULL	0	1	1	0	0	3
14	C:\Windows\System32\dsc\D...	68D8D0503A1...	2	1	NULL	0	1	1	0	1	3
15	C:\Windows\System32\dsc\P...	AC11B47A62A...	2	1	NULL	0	1	1	0	1	3
16	C:\Windows\System32\Spee...	0BD03DA621D...	2	1	NULL	0	1	1	0	1	3
17	C:\Windows\System32\wbe...	97ED913FED4...	2	1	NULL	0	1	1	0	1	3

1 - 18 of 3877 — Go to: 1

Εικόνα 64: Πλήθος ψηφιακά υπογεγραμμένων αρχείων



	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	Filter	Filter	Filter	Fil...	Filter	0	Filter	0	Filter	Filter	Filter
1	C:\Users\mythesis\AppData\L...	E077DC17D85...	NULL	1	NULL	0	0	0	0	0	4
2	C:\Users\mythesis\AppData\L...	2E4FA184CA0...	NULL	1	NULL	0	0	0	0	0	4
3	C:\Users\mythesis\AppData\L...	5839AEC9398...	NULL	1	NULL	0	0	0	0	0	4
4	C:\Users\mythesis\AppData\L...	9BD4E885289...	NULL	1	NULL	0	0	0	0	0	4
5	C:\Users\mythesis\AppData\L...	77C9ACE9EB9...	NULL	1	NULL	0	0	0	0	0	4
6	C:\Users\mythesis\AppData\L...	8D04626F655...	NULL	1	NULL	0	0	0	0	0	4
7	C:\Users\mythesis\AppData\L...	4C9A77DB2F8...	NULL	1	NULL	0	1	0	0	1	5
8	C:\Users\mythesis\AppData\L...	8ABAA1BEE3B...	NULL	1	NULL	0	1	0	0	1	5
9	C:\Users\mythesis\AppData\L...	3B7B725F447...	NULL	1	NULL	0	1	0	0	1	5
10	C:\Users\mythesis\AppData\L...	6049F95A040...	NULL	1	NULL	0	1	0	0	1	5
11	C:\Users\mythesis\AppData\L...	645034C07AD...	NULL	1	NULL	0	1	0	0	1	5
12	C:\Users\mythesis\AppData\L...	3BA2C76EBE0...	NULL	1	NULL	0	1	0	0	1	5
13	C:\Users\mythesis\AppData\L...	8CAC6954221...	NULL	1	NULL	0	1	0	0	1	5
14	C:\Users\mythesis\AppData\L...	B8A256C66D2...	NULL	1	NULL	0	1	0	0	1	5
15	C:\Users\mythesis\AppData\L...	8B867F5655F...	NULL	1	NULL	0	1	0	0	1	5
16	C:\Users\mythesis\AppData\L...	73FD449EBCF...	NULL	1	NULL	0	1	0	0	1	5
17	C:\Users\mythesis\AppData\L...	908A2402DCB...	NULL	1	NULL	0	1	0	0	1	5
18	C:\Users\mythesis\AppData\L...	F7B60889CD8...	NULL	1	NULL	0	1	0	0	1	5

1 - 18 of 77 — Go to: 1

Εικόνα 65: Πλήθος αρχείων χωρίς πληροφορίες copyright

	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	Filter	Filter	Filter	Fil...	Filter	0	Filter	Filter	0	Filter	Filter
1	C:\Windows\System32\NlsLe...	4693FA55E3C...	2	1	NULL	0	1	1	0	1	2
2	C:\Windows\System32\tzres.dll	F69DEC1347D...	2	1	NULL	0	1	1	0	1	2
3	C:\Windows\System32\tzsyn...	5A469D7560A...	2	1	NULL	0	1	1	0	1	2
4	C:\Windows\System32\UtcRe...	61697B5750E...	2	1	NULL	0	1	1	0	1	2
5	C:\Windows\System32\Wdfr...	1FFF62122F8...	2	1	NULL	0	1	1	0	1	2
6	C:\Windows\System32\winrs...	496FC0A4270...	2	1	NULL	0	1	1	0	1	2
7	C:\Windows\System32\wlidr...	5C4CDAA885...	2	1	NULL	0	1	1	0	1	2
8	C:\Windows\System32\wmer...	D548957F180...	2	1	NULL	0	1	1	0	1	2
9	C:\Windows\System32\Wsm...	A6F9B81EC9B...	2	1	NULL	0	1	1	0	1	2
10	C:\Windows\System32\Driver...	33F49523108...	2	1	NULL	0	1	1	0	1	3
11	C:\Windows\System32\Driver...	477451F6E25...	2	1	NULL	0	1	1	0	0	3
12	C:\Windows\System32\Driver...	111B223D99C...	2	1	NULL	0	1	1	0	0	3
13	C:\Windows\System32\dsc\D...	68D8D0503A1...	2	1	NULL	0	1	1	0	1	3
14	C:\Windows\System32\dsc\P...	AC11B47A62A...	2	1	NULL	0	1	1	0	1	3
15	C:\Windows\System32\Spee...	0BD03DA621D...	2	1	NULL	0	1	1	0	1	3
16	C:\Windows\System32\wbe...	97ED913FED4...	2	1	NULL	0	1	1	0	1	3
17	C:\Windows\System32\Wind...	EBA28B09749...	2	1	NULL	0	1	1	0	1	3

1 - 18 of 3402 Go to: 1

Εικόνα 66: Πλήθος αρχείων χωρίς εικόνα

	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	Filter	Filter	Filter	Fil...	Filter	0	Filter	Filter	Filter	1	Filter
1	C:\Windows\System32\NlsLe...	4693FA55E3C...	2	1	NULL	0	1	1	0	1	2
2	C:\Windows\System32\tzres.dll	F69DEC1347D...	2	1	NULL	0	1	1	0	1	2
3	C:\Windows\System32\tzsyn...	5A469D7560A...	2	1	NULL	0	1	1	0	1	2
4	C:\Windows\System32\UtcRe...	61697B5750E...	2	1	NULL	0	1	1	0	1	2
5	C:\Windows\System32\Wdfr...	1FFF62122F8...	2	1	NULL	0	1	1	0	1	2
6	C:\Windows\System32\winrs...	496FC0A4270...	2	1	NULL	0	1	1	0	1	2
7	C:\Windows\System32\wlidr...	5C4CDAA885...	2	1	NULL	0	1	1	0	1	2
8	C:\Windows\System32\wmer...	D548957F180...	2	1	NULL	0	1	1	0	1	2
9	C:\Windows\System32\Wsm...	A6F9B81EC9B...	2	1	NULL	0	1	1	0	1	2
10	C:\Windows\System32\wush...	B9438C16DDA...	2	1	NULL	0	1	1	1	1	2
11	C:\Windows\System32\Driver...	33F49523108...	2	1	NULL	0	1	1	0	1	3
12	C:\Windows\System32\dsc\D...	68D8D0503A1...	2	1	NULL	0	1	1	0	1	3
13	C:\Windows\System32\dsc\P...	AC11B47A62A...	2	1	NULL	0	1	1	0	1	3
14	C:\Windows\System32\Spee...	0BD03DA621D...	2	1	NULL	0	1	1	0	1	3
15	C:\Windows\System32\wbe...	97ED913FED4...	2	1	NULL	0	1	1	0	1	3
16	C:\Windows\System32\Wind...	EBA28B09749...	2	1	NULL	0	1	1	0	1	3
17	C:\Windows\System32\Wind...	27DE1C42F37...	2	1	NULL	0	1	1	0	1	3

1 - 18 of 1595 Go to: 1

Εικόνα 67: Πλήθος αρχείων που βρέθηκαν θετικά στη χρήση packer

	Path	SHA1	exist	vthash	upload	found	signed	resource	image	packer	yara
	Filter	Filter	Filter	Fil...	Filter	0	Filter	Filter	Filter	Filter	Filter
1	C:\Users\All Users\Microsoft\...	60A53652DF9...	NULL	1	NULL	0	1	1	0	0	19
2	C:\Windows\System32\MRT-...	836DAA7E1F4...	2	1	NULL	0	1	1	1	1	18
3	C:\Windows\System32\MRT....	836DAA7E1F4...	2	1	NULL	0	1	1	1	1	18
4	C:\Users\mythesis\Desktop\...	4080BB3A28C...	NULL	2	NULL	0	0	0	0	1	17
5	C:\Windows\System32\Driver...	0B152C7E9E5...	2	1	NULL	0	1	1	0	0	16
6	C:\Windows\System32\spool...	0B152C7E9E5...	2	1	NULL	0	1	1	0	0	16
7	C:\Windows\System32\ie4uin...	7BFB1BC6672...	2	1	NULL	0	1	1	0	1	16
8	C:\Windows\System32\SkyDr...	86F5ED538E6...	2	1	NULL	0	1	1	1	1	16
9	C:\Users\All Users\Package C...	3FEDA4E77DC...	NULL	1	NULL	0	1	1	0	1	15
10	C:\Users\mythesis\AppData\L...	292CB9216E3...	NULL	1	NULL	0	0	1	0	0	15
11	C:\Users\mythesis\AppData\L...	292CB9216E3...	NULL	1	NULL	0	0	1	0	0	15
12	C:\Users\mythesis\AppData\L...	292CB9216E3...	NULL	1	NULL	0	0	1	0	0	15
13	C:\Users\mythesis\AppData\L...	292CB9216E3...	NULL	1	NULL	0	0	1	0	0	15
14	C:\Users\mythesis\AppData\L...	292CB9216E3...	NULL	1	NULL	0	0	1	0	0	15
15	C:\Users\mythesis\Desktop\...	A1B9024EB52...	NULL	2	NULL	0	0	0	0	1	15
16	C:\Windows\System32\Adva...	28E758FADC3...	2	1	NULL	0	1	1	0	0	15
17	C:\Windows\System32\Driver...	F63EAAB0F5E...	2	1	NULL	0	1	1	1	0	15

1 - 18 of 3934 Go to: 1

Εικόνα 68: Στιγμιότυπο ταυτοποιήσεων από κανόνες yara

Στη συνέχεια εκτελέστηκε ο διακόπτης «-rs» για την εξαγωγή των αποτελεσμάτων σε αρχεία. Τα αρχεία αναφορών είναι διαθέσιμα στο παράρτημα 3

Αρχικά, επιλέξαμε να μας επιστραφούν όλα τα αρχεία, που χαρακτηρίστηκαν κακόβουλα από το VirusTotal, χωρίς να δώσουμε καθόλου βάρος στα υπόλοιπα χαρακτηριστικά της βάσης. Όλα τα κακόβουλα αρχεία που βρίσκονταν στους φακέλους που υποδείξαμε εντοπίστηκαν με επιτυχία, με δέκα ακόμα αρχεία να έχουν ανιχνευτεί λανθασμένα.

A20 C:\Windows\System32\msoeacct.dll

	A	B	C	D	E	F	G	H	I	J	K
1	Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
2	C:\Users\mythesis\AppData\Roaming\Microsoft\Install	2246FA0E873D0C411	None	2	None	0	0	0	0	1	7
3	C:\Users\mythesis\Desktop\mal\Artemis!InstallBC201	08B9F5874AD1DC3E	None	2	None	0	0	1	0	0	11
4	C:\Users\mythesis\Desktop\mal\Ransomware.Cerber\	C69A0F6C6F809C01D	None	2	None	0	0	0	0	1	12
5	C:\Users\mythesis\Desktop\mal\Trojan.Win32.Bechiro	12C42B7FEFDEFB75	None	2	None	0	1	1	0	0	11
6	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypte	54D44EDC6662D84F9	None	2	None	0	0	0	0	1	7
7	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypte	D83E1FFA253B6D23	None	2	None	0	0	0	0	0	6
8	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypte	6016DCA97148C2CF0	None	2	None	0	0	0	0	0	9
9	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypte	1CF6E12DF0D786934	None	2	None	0	0	0	0	1	10
10	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypte	41D72FA626CCCADC	None	2	None	0	0	0	0	1	7
11	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypte	3FA91F0F116CC0F19	None	2	None	0	0	0	0	1	7
12	C:\Users\mythesis\Desktop\mal\131.exe	4080BB3A28C2946FD	None	2	None	0	0	0	0	1	17
13	C:\Users\mythesis\Desktop\mal\dumped.exe	A1B9024EB52A4450A	None	2	None	0	0	0	0	1	15
14	C:\Users\mythesis\Desktop\mal\fax_390392029_0725	356B21B749C8BC5E2	None	2	None	0	0	0	0	1	5
15	C:\Users\mythesis\Desktop\mal\file_4571518150a818	4571518150A8181B40	None	2	None	0	0	0	0	0	7
16	C:\Users\mythesis\Desktop\rastrea2r-master\rastrea2	CE527E4B98099F7CB	None	2	None	0	0	0	0	0	7
17	C:\Users\mythesis\Desktop\rastrea2r-master\rastrea2	9D78DE808B60DF9C0	None	2	None	0	0	0	0	0	13
18	C:\Users\mythesis\Downloads\Dumplt.exe	5741AF8CC8A4DED2	None	2	None	0	1	0	0	1	14
19	C:\Windows\System32\dnsapi.dll	1FB37E98D3ECE458	2	2	None	0	1	1	0	0	9
20	C:\Windows\System32\msoeacct.dll	2E184807B383DDA29	2	2	None	0	1	1	0	0	8
21	C:\Windows\System32\MultiDigiMon.exe	294AA1ABF2DE840F	2	2	None	0	1	1	1	1	9
22	C:\Windows\System32\NlsLexicons0039.dll	2132A26AD90D57262	2	2	None	0	1	1	0	1	4
23	C:\Windows\System32\PrintIsolationHost.exe	0EE26ADFE8F75D4A	2	2	None	0	1	1	1	1	7
24	C:\Windows\System32\shellstyle.dll	086E50434459760924	2	2	None	0	1	1	0	1	4
25											
26											

Εικόνα 69: Αναφορά κακόβουλου λογισμικού 1

Στη συνέχεια ζητήσαμε να μας επιστραφεί μια αναφορά για όλα τα αρχεία για τα οποία δεν βρέθηκε ψηφιακή υπογραφή, έχουν hits από κανόνες yara και έχει ανιχνευτεί χρήση packer. Συνολικά οκτώ από τα δεκατρία κακόβουλα αρχεία βρίσκονται στην αναφορά καθώς πέντε από αυτά φέρονται να μην έχουν ενδείξεις χρήσης κάποιου γνωστού packer.

	A	B	C	D	E	F	G	H	I	J	K
1	Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
2	C:\Users\mythesis\AppData\Local\Temp\pip-epr5fe-uninstall\p	1EFF048C01FD4BB618E858F89FB3FF	None	1	None	0	0	0	0	1	9
3	C:\Users\mythesis\AppData\Roaming\Microsoft\Installer\{C0C	2246FA0E873D0C4110A015789ED10F4	None	2	None	0	0	0	0	1	7
4	C:\Users\mythesis\Desktop\mal\Ransomware.Cerber\cerber.e	C69A0F6C6F809C01DB92CA658FCF1E	None	2	None	0	0	0	0	1	12
5	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_inj_s	54D44EDC6662D84F54387B8F9817E0F	None	2	None	0	0	0	0	1	7
6	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_inj_s	1CF6E12DF0D786934C2AD5B706F4F8	None	2	None	0	0	0	0	1	10
7	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_rkctl_	41D72FA626CCCADCD5AAE3A0A6520	None	2	None	0	0	0	0	1	7
8	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_rkctl_	3FA91F0F116CC0F19E8105BC51B91B	None	2	None	0	0	0	0	1	7
9	C:\Users\mythesis\Desktop\mal\131.exe	4080BB3A28C2946FD9B72F6B51FE15	None	2	None	0	0	0	0	1	17
10	C:\Users\mythesis\Desktop\mal\dumped.exe	A1B9024EB52A4450AE587DFDDFCAE	None	2	None	0	0	0	0	1	15
11	C:\Users\mythesis\Desktop\mal\fax_390392029_072514.exe	356B21B749C8BC5E2295A3DB62EA03	None	2	None	0	0	0	0	1	5
12	C:\Users\mythesis\Desktop\rastrea2r-master\rastrea2r-master	1A1998A6D9A0CC0D4EE1E1D0641E8	None	1	None	0	0	0	1	1	11
13											
14											
15											
16											

Εικόνα 70: Αναφορά κακόβουλου λογισμικού 2

Δοκιμάσαμε επίσης, να εξάγουμε μια αναφορά για όλα τα αρχεία τα έχουν ταυτοποιηθεί για χρήση packer, έχουν hits από κανόνες yara, δεν έχουν ψηφιακή υπογραφή αλλά ούτε icon και λοιπά resources με αποτέλεσμα να εντοπίσουμε πάλι οκτώ από τα δεκατρία κακόβουλα αρχεία.

	A	B	C	D	E	F	G	H	I	J	K
1	Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
2	C:\Users\mythesis\AppData\Local\Temp\pip-epr5fe-uninstall\p	1EFF048C01FD4BB618E858F89FB3FF	None	1	None	0	0	0	0	1	9
3	C:\Users\mythesis\AppData\Roaming\Microsoft\Installer\{C0C	2246FA0E873D0C4110A015789ED10F4	None	2	None	0	0	0	0	1	7
4	C:\Users\mythesis\Desktop\mal\Ransomware.Cerber\cerber.e	C69A0F6C6F809C01DB92CA658FCF1E	None	2	None	0	0	0	0	1	12
5	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_inj_s	54D44EDC6662D84F54387B8F9817E0F	None	2	None	0	0	0	0	1	7
6	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_inj_s	1CF6E12DF0D786934C2AD5B706F4F8	None	2	None	0	0	0	0	1	10
7	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_rkctl_	41D72FA626CCCADCD5AAE3A0A6520	None	2	None	0	0	0	0	1	7
8	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypted_rkctl_	3FA91F0F116CC0F19E8105BC51B91B	None	2	None	0	0	0	0	1	7
9	C:\Users\mythesis\Desktop\mal\131.exe	4080BB3A28C2946FD9B72F6B51FE15	None	2	None	0	0	0	0	1	17
10	C:\Users\mythesis\Desktop\mal\dumped.exe	A1B9024EB52A4450AE587DFDDFCAE	None	2	None	0	0	0	0	1	15
11	C:\Users\mythesis\Desktop\mal\fax_390392029_072514.exe	356B21B749C8BC5E2295A3DB62EA03	None	2	None	0	0	0	0	1	5
12											
13											
14											

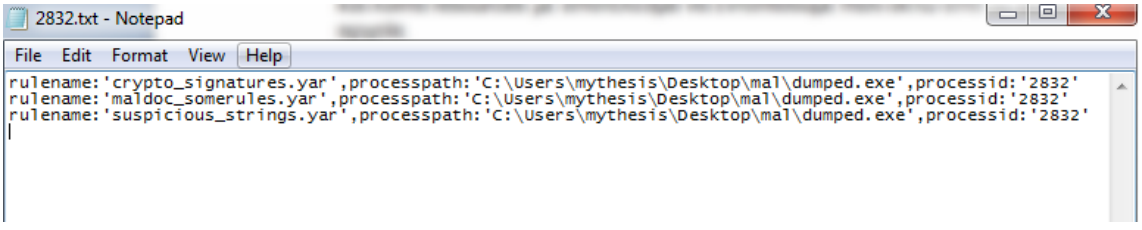
Εικόνα 71: Αναφορά κακόβουλου λογισμικού 3

Στη συνέχεια, ζητήσαμε την αναφορά όλων των αρχείων που χαρακτηρίστηκαν κακόβουλα από το VirusTotal, δεν έχουν ψηφιακή υπογραφή, δεν έχουν resources και icon ενώ έχουν hits από κανόνες yara και έχουν βρεθεί θετικά στη χρήση packer. Καταφέραμε να μειώσουμε τον αριθμό των false negative της πρώτης αναφοράς αλλά «χάσαμε» ξανά πέντε κακόβουλα αρχεία.

	A	B	C	D	E	F	G	H	I	J	K
1	Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
2	C:\Users\mythesis\AppData\Roaming\Microsoft\Instal	2246FA0E873D0C4110A015789ED10F4F9485D5	None	2	None	0	0	0	0	1	7
3	C:\Users\mythesis\Desktop\mal\Ransomware.Cerber	C69A0F6C6F809C01DB92CA658FCF1B643391A	None	2	None	0	0	0	0	1	12
4	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypt	54D44EDC6662D84F54387B8F9817E0F87F9FE	None	2	None	0	0	0	0	1	7
5	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypt	1CF6E12DF0D786934C2AD5B706F4F8795F776	None	2	None	0	0	0	0	1	10
6	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypt	41D72FA626CCCADCD5AAE3A0A652043E4A4F	None	2	None	0	0	0	0	1	7
7	C:\Users\mythesis\Desktop\mal\Win32.Turla\decrypt	3FA91F0F116CC0F19E8105BC51B91B7639605	None	2	None	0	0	0	0	1	7
8	C:\Users\mythesis\Desktop\mal\131.exe	4080BB3A28C2946FD9B72F6B51FE15DE74CBB	None	2	None	0	0	0	0	1	17
9	C:\Users\mythesis\Desktop\mal\dumped.exe	A1B9024EB52A4450AE587DFDDFCAE37581DA	None	2	None	0	0	0	0	1	15
10	C:\Users\mythesis\Desktop\mal\fax_390392029_072	356B21B749C8BC5E2295A3DB62EA03C47CB4	None	2	None	0	0	0	0	1	5
11											
12											

Εικόνα 72: Αναφορά κακόβουλου λογισμικού 4

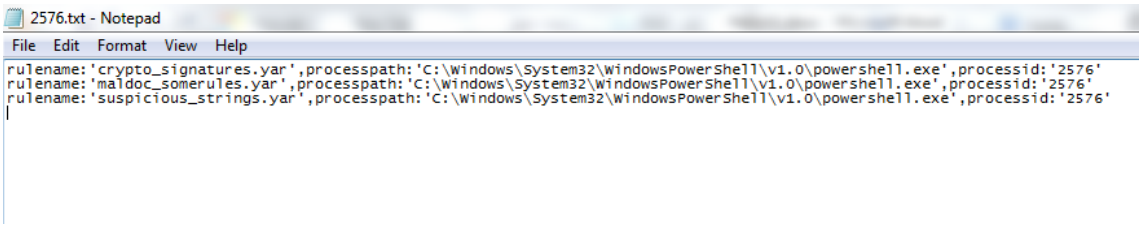
Τέλος εκτελέστηκε το κακόβουλο λογισμικό «dumped.exe» με σκοπό να εκκινήσουμε τη διεργασία του και να την ελέγξουμε μαζί με όλες τις υπόλοιπες ενεργές διεργασίες χρησιμοποιώντας κανόνες yara. Χρησιμοποιήσαμε τρείς ξεχωριστούς κανόνες, τον «crypto_signatures.yar», τον «maldoc_somerules.yar» και τον «suspicious_strings.yar». Τα στιγμιότυπα της ανάλυσης παρατίθενται στις επόμενες εικόνες.



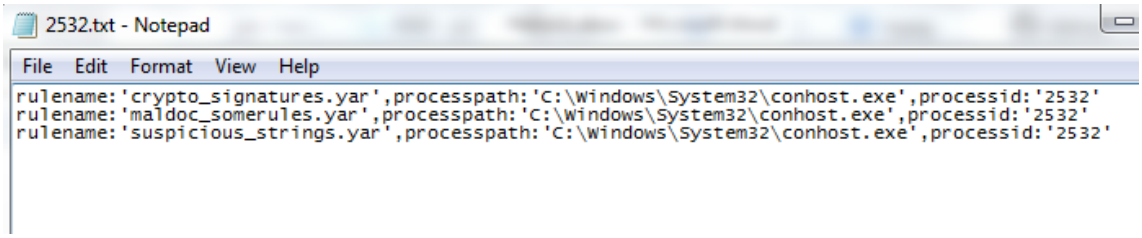
Εικόνα 73: Αναφορά διεργασίας dumped.exe

Εκτός από την διεργασία «dumped.exe» κακόβουλες χαρακτηρίστηκαν και οι διεργασίες:

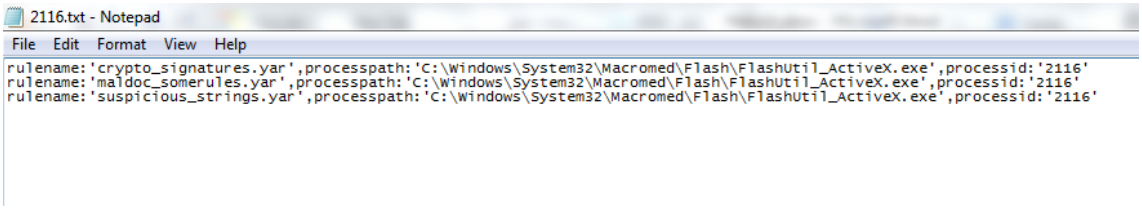
- powershell.exe
- conhost.exe
- FlashUtil_ActiveX.exe
- DB Browser for SQLite.exe



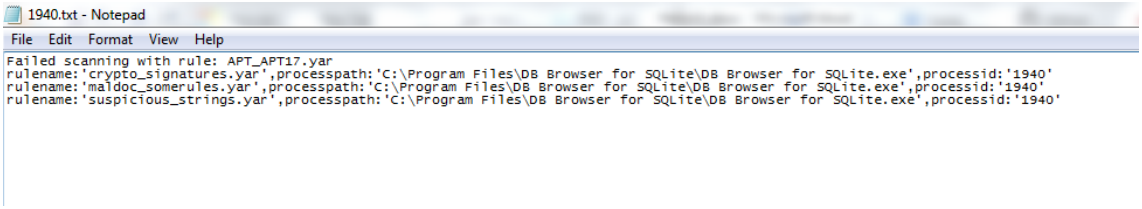
Εικόνα 74: Αναφορά διεργασίας powershell



Εικόνα 75: Αναφορά διεργασίας conhost



Εικόνα 76: Αναφορά διεργασίας FlashUtil_ActiveX



Εικόνα 77: Αναφορά διεργασίας DB Browser for SQLite

Η ερμηνεία των αποτελεσμάτων, είναι στα χέρια του εκάστοτε ερευνητή, καθώς δίνεται η δυνατότητα να επιλέξει σε ποία από τα πεδία της βάσης θέλει να επικεντρωθεί.

6 Μελλοντικές Προσθήκες

Το εργαλείο αναπτύχθηκε σε πρώτο στάδιο, για να υλοποιεί «on-demand» ελέγχους στα αρχεία υπολογιστών που λειτουργούν με windows 8 λειτουργικό σύστημα. Η χρήση του, περιορίζεται στη συγκέντρωση πληροφοριών από διάφορες πηγές, όπως για παράδειγμα το virus total προκειμένου να βοηθήσει σε τομείς ψηφιακής σήμανσης, υλοποιώντας αυτοματοποιημένους ελέγχους στα αρχεία του συστήματος.

Οι κύριες λειτουργίες του εργαλείου, αφορούν την ανίχνευση malware με τη χρήση υπογραφών από γνωστά κακόβουλα λογισμικά. Μια σημαντική προσθήκη στο εργαλείο, θα μπορούσε να είναι η δυνατότητα να αναλύει τα αρχεία, για τα οποία υπάρχουν ενδείξεις που

παραπέμπουν σε κακόβουλο λογισμικό μέσω συμπεριφοριακής ανάλυσης, χρησιμοποιώντας αυτοματοποιημένα συστήματα ανάλυσης malware όπως για παράδειγμα το «cuckoo» [27].

Μια ακόμα προσθήκη που θα παρουσίαζε ενδιαφέρον, είναι μια αποδοτικότερη επιστροφή των αποτελεσμάτων. Θα μπορούσαμε μετά από μελέτη να δώσουμε βάρη σε κάθε πεδίο της βάσης δεδομένων και να ρυθμίσουμε ένα κατώφλι πάνω από το οποίο θα χαρακτηριζόταν ένα αρχείο ως κακόβουλο. Είναι εμφανές, ότι δεν μπορεί να έχει κάθε πεδίο της βάσης τον ίδιο αντίκτυπο για το χαρακτηρισμό ενός αρχείου ως κακόβουλο. Ο χαρακτηρισμός από το virus total, είναι πολύ πιο σημαντικός από τις ταυτοποιήσεις του yara και αντίστοιχα, οι κανόνες yara αποτελούν πιο σημαντικό κριτήριο από την ύπαρξη icon σε ένα εκτελέσιμο. Είναι σημαντικό να βρεθεί λοιπόν, μια «φόρμουλα» βάση της οποίας, όλα τα παραπάνω δεδομένα θα συνδυάζονται με σωστά βάρη προκειμένου να φτάσουμε στο επιθυμητό αποτέλεσμα.

7 Συμπεράσματα

Λόγο της ραγδαίας αύξησης των κακόβουλων προγραμμάτων και των παραβιάσεων των υπολογιστικών συστημάτων κρίθηκε χρήσιμη η ανάπτυξη ενός εργαλείου το οποίο στόχο θα είχε να βοηθήσει και να αυτοματοποιήσει τη διαδικασία εντοπισμού κακόβουλου κώδικα τόσο σε αρχεία όσο και σε πτητικά δεδομένα.

Για τον εντοπισμό κακόβουλου λογισμικού μελετήθηκαν διαφορετικοί τρόποι προσέγγισης όπως εντοπισμός με τη χρήση υπογραφών, εντοπισμός βάση της συμπεριφοράς του αρχείου και ευριστικές μέθοδοι. Κάθε προσέγγιση παρουσιάζει ισχυρά πλεονεκτήματα αλλά και αδυναμίες.

Το εργαλείο που δημιουργήθηκε, εστιάζει στη συλλογή δεδομένων από τα αρχεία windows λειτουργικών συστημάτων προκειμένου να τα κατατάξει με βαθμό επικινδυνότητας, χρησιμοποιώντας διαφορετικές μεθόδους. Κύριο χαρακτηριστικό του εργαλείου αποτελεί η ανίχνευση βάση υπογραφών αλλά προσφέρονται μέσω εξωτερικών εργαλείων όπως για παράδειγμα με τη χρήση του virus total και ευριστικές μέθοδοι ανίχνευσης. Η ερμηνεία των δεδομένων που συλλέγονται, παρόλα αυτά, παραμένει στα χέρια του ερευνητή.

Οι λειτουργίες που προσφέρονται από το εργαλείο είναι η ανίχνευση χρήσης packer και ο έλεγχος των resources των portable executable αρχείων, ο έλεγχος των αρχείων με τη χρήση του virus total και κανόνων yara, ο εντοπισμός ύπαρξης ψηφιακής υπογραφής και η αναζήτηση των hash values των προγραμμάτων στη βάση της NSRL. Όσον αφορά τα πτητικά δεδομένα, επιτρέπει την ανίχνευση κακόβουλου κώδικα σε διεργασίες με τη χρήση κανόνων yara αλλά και αποθήκευση και ανάλυση της μνήμης και της registry με τη χρήση δοκιμασμένων και κορυφαίων open source λογισμικών.

Μελλοντική ανάπτυξη του εργαλείου θα μπορούσε να προσθέσει επιπλέον λειτουργίες τόσο στην αυτοματοποιημένη παρουσίαση των αποτελεσμάτων όσο και στην προσθήκη επιπλέον μεθόδων ανίχνευσης κακόβουλου κώδικα.

Το εργαλείο δοκιμάστηκε επιτυχώς σε ένα ελεγχόμενο windows 8.1 περιβάλλον το οποίο ηθελημένα είχε μολυνθεί από γνωστά κακόβουλα λογισμικά και τα αποτελέσματα που ανακτήθηκαν ήταν ικανοποιητικά καθώς καταφέραμε να αναγνωρίσουμε όλα τα «μολυσμένα» εκτελέσιμα με τον αριθμό των εσφαλμένως θετικών και εσφαλμένος αρνητικών ενδείξεων να μεταβάλετε ανάλογα με τα πεδία που επιθυμούσε να εστιάσει ο αναλυτής.

8 Βιβλιογραφία

1] Av-test.org, "AV-TEST - The Independent IT-Security Institute," [Online]. Available: https://www.av-test.org/en/statistics/malware/. [Accessed 12 10 2017].

T. Roy and A. Jain, "Windows Registry Forensics: An Imperative Step in Tracking Data

2] Theft via USB Devices," *International Journal of Computer Science and Information Technologies,* vol. 3, no. 3, pp. 4427-4433, 2012.

3] H. Saini, Y. S. Rao and T. C. Panda, "Cyber-Crimes and their Impacts: A Review," *International Journal of Engineering Research and Applications,* vol. 2, no. 2, pp. 202-209, 2012.

4] L. J. Janczewski and A. M. Colarik, Cyber Warfare and Cyber Terrorism, Information Science Reference, 2007.

5] M. F. Zolkipli and A. Jantan, "An approach for malware behavior identification and classification," *Computer Research and Development,* vol. 1, no. 3, 2011.

6] J. Landage and M. Wankhade, "Malware and Malware Detection Techniques: A Survey," *Internation Journal of Engineering Research & Technology,* vol. 2, no. 12, 2013.

7] A. Mujumdar, G. Masiwal and D. B. B. Meshram, "Analysis of Signature-Based and Behavior-Based Anti-Malware Approaches," *International Journal of Advanced Research in Computer Engineering and Technology,* vol. 2, no. 6, 2013.

8] M. Damshenas, A. Dehghantanha and R. Mahmoud, "A Survey on Malware Propagation, Analysis and Detection," *International Journal of Cyber-Security and Digital Forensics,* vol. 2, no. 4, pp. 10-29, 2013.

9] S. A. Amro and A. Alkhalifah, "A Comparative Study of Virus Detection Techniques," *International Journal of Computer, Electrical, Automation, Control and Information Engineering,* vol. 9, no. 6, 2015.

10] S. V. S. B. T. Y. Li Sun, "Pattern Recognition Techniques for the Classification of Malware Packers," in *Information Security and Privacy*, Sydney, Australia, 2010.

11] "PEiD - aldeid," [Online]. Available: https://www.aldeid.com/wiki/PEiD. [Accessed 25 05 2017].

12] E. Carrera, "Ero Carrera's blog," 09 06 2007. [Online]. Available: http://blog.dkbza.org/2007/06/pefile-and-packer-detection.html. [Accessed 10 08 2017].

13] B. Preneel, "Cryptographic Hash Functions," *European Transactions on Telecommunications,* vol. 5, no. 4, pp. 431-448, 2010.

14] V. M. Alvarez, "yara-python 3.6.3 : Python Package Index," Python Software Foundation, [Online]. Available: https://pypi.python.org/pypi/yara-python. [Accessed 06 10 2017].

15] E. Carrera, "GitHub," [Online]. Available: https://github.com/erocarrera/pefile. [Accessed 07 10 2017].

16] NIST, "NSRL Introduction," NIST, [Online]. Available: https://www.nist.gov/software-quality-group/national-software-reference-library-nsrl. [Accessed 15 8 2017].

17] H. Carvey, Windows Forensic Analysis Toolkit, Fourth Edition: Advanced Analysis Techniques for Windows 8, Syngress, 2014.

18] T. Peters, "PEP 20 -- The Zen of Python," [Online]. Available: https://www.python.org/dev/peps/pep-0020/. [Accessed 13 08 2017].

19] M. Hammond and A. Robinson, Python Programming on Win32: Help for Windows Programmers, O'Reilly Media.

20] D. Farmer and W. Venema, Forensics Discovery, Addison-Wesley , 2005.

21] M. H. Ligh, A. Case, J. Levy and A. Walters, The art of memory forensics, Wiley, 2014.

22] US-CERT, "Computer Forensics," [Online]. Available: https://www.us-cert.gov/sites/default/files/publications/forensics.pdf. [Accessed 18 09 2017].

S. INSTITUTE, "Techiniques and Tools for Recovering and Analyzing Data from Volatile
23] Memory," 2009. [Online]. Available: https://www.sans.org/reading-room/whitepapers/forensics/techniques-tools-recovering-analyzing-data-volatile-memory-33049. [Accessed 22 08 2017].

C. Harlan, "The Windows Registry as a forensics resource," *Digital Investigation: The*
24] *International Journal of Digital Forensics & Incident Response,* vol. 2, no. 3, 2005.

Y. Nativ, S. Shalev and S. Divskinsy, "GitHub," [Online]. Available:
25] https://github.com/ytisf/theZoo. [Accessed 20 06 2017].

"Yara-Rules/rules," [Online]. Available: https://github.com/Yara-Rules/rules. [Accessed 3 09
26] 2017].

C. Foundation, «Cuckoo Sandbox Book,» [Ηλεκτρονικό]. Available: https://cuckoo.sh/docs.
27] [Πρόσβαση 09 05 2017].

R. Perdisci, A. Lanzi and W. Lee, "Classification of Packed Executables for Accurate
28] Computer Virus Detection," *Pattern Recognition Letters,* vol. 29, no. 14, pp. 1941-1946, 2008.

C. Foundation, "Cuckoo Sandbox Book," [Online]. Available: https://cuckoo.sh/docs.
29] [Accessed 09 05 2017].

Παράρτημα

1. Sigcheck

usage: sigcheck [-a][-h][-i][-e][-l][-n][[-s]|[-c|-ct]|[-m]][-q][-r][-u][-vt][-v[r][s]][-f catalog file] <file or directory>
usage: sigcheck -d [-c|-ct] <file or directory>
usage: sigcheck -o [-vt][-v[r]] <sigcheck csv file>
usage: sigcheck -t[u][v] [-i] [-c|-ct] <certificate store name|*>

Parameter	Description
-a	Show extended version information. The entropy measure reported is the bits per byte of information of the file's contents.
-c	CSV output with comma delimiter
-ct	CSV output with tab delimiter
-d	Dump contents of a catalog file
-e	Scan executable images only (regardless of their extension)
-f	Look for signature in the specified catalog file
-h	Show file hashes
-i	Show catalog name and signing chain
-l	Traverse symbolic links and directory junctions
-m	Dump manifest
-n	Only show file version number
-o	Performs Virus Total lookups of hashes captured in a CSV file previously captured by Sigcheck when using the -h option. This usage is intended for scans of offline systems.
-q	Quit (no banner)
-r	Disable check for certificate revocation
-s	Recurse subdirectories
- t[u][v]	Dump contents of specified certificate store ('*' for all stores). Specify -tu to query the user store (machine store is the default). Append '-v' to have Sigcheck download the trusted Microsoft root certificate list and only output valid certificates not rooted to a certificate on that list. If the site is not accessible, authrootstl.cab or authroot.stl in the current directory are used instead, if present.
-u	If VirusTotal check is enabled, show files that are unknown by VirusTotal or have non-zero detection, otherwise show only unsigned files.
-v[rs]	Query VirusTotal (www.virustotal.com) for malware based on file hash. Add 'r' to open reports for files with non-zero detection. Files reported as not previously scanned will be uploaded to VirusTotal if the 's' option is specified. Note scan results may not be available for five or more minutes
-vt	Before using VirusTotal features, you must accept VirusTotal terms of service. See: https://www.virustotal.com/en/about/terms-of-service/ If you haven't accepted the terms and you omit this option, you will be interactively prompted.

2. Virus total

File characterization tools & datasets	Antivirus products	Website/domain scanning engines & datasets
Androguard (Anthony Desnos)	AegisLab (AegisLab)	ADMINUSLabs (ADMINUSLABS)
Cuckoo Sandbox (Claudio Guarnieri)	Agnitum (Agnitum)	AegisLab WebGuard (AegisLab)
ExifTool (Phil Harvey)	AhnLab (V3)	Alexa (Amazon)
Magic descriptor (Linux)	Alibaba Group (Alibaba)	AlienVault (AlienVault)

NSRL information (NIST's National Software Reference Library)	Antiy Labs (Antiy-AVL)	Antiy-AVL (Antiy Labs)
PDFiD (Didier Stevens)	ALWIL (Avast! Antivirus)	AutoShun (RiskAnalytics)
pefile (Ero Carrera)	Arcabit (Arcabit)	Avira Checkurl (Avira)
PEiD (Jibz)	AVG Technologies (AVG)	Baidu-International (Baidu)
Sigcheck (Mark Russinovich)	Avira (AntiVir)	BitDefender (BitDefender)
Snort (Sourcefire)	BluePex (AVware)	Blueliv (Blueliv)
ssdeep (Jesse Kornblum)	Baidu (Baidu-International)	CRDF (CRDF FRANCE)
Suricata (Open Information Security Foundation)	BitDefender GmbH (BitDefender)	C-SIRT (Cyscon SIRT)
Taggant packer information tool (ReversingLabs)	Bkav Corporation (Bkav)	CLEAN MX (CLEAN MX)
TrID (Marco Pontello)	ByteHero Information Security Technology Team (ByteHero)	Comodo Site Inspector (Comodo Group)
UEFI Firmware parser (Teddy Reed)	Cat Computer Services (Quick Heal)	CyberCrime (Xylitol)
Wireshark (Wireshark Foundation)	Check Point Software Technologies (ZoneAlarm by Check Point)	desenmascara.me (An emiliocasbas.net project)
Zemana behaviour (Zemana)	ClamAV (ClamAV)	Dr.Web Link Scanner (Dr.Web)
CarbonBlack (CarbonBlack)	CMC InfoSec (CMC Antivirus)	Emsisoft (Emsi Software GmbH)
	Comodo (Comodo)	ESET (ESET)
	Cyren (Cyren)	FortiGuard Web Filtering (Fortinet)
	CrowdStrike (CrowdStrike Falcon (ML))	FraudSense (FraudSense)
	Doctor Web, Ltd. (DrWeb)	G-Data (G Data)
	Endgame (Endgame)	Google Safebrowsing (Google)
	ESTsecurity (ALYac)	K7AntiVirus (K7 Computing)
	Emsi Software GmbH (Emsisoft)	Kaspersky URL advisor (Kaspersky)
	Eset Software (ESET NOD32)	Malc0de Database (Malc0de)
	Fortinet (Fortinet)	Malekal (Malekal's MalwareDB)
	FRISK Software (F-Prot)	Malwarebytes hpHosts (Malwarebytes)
	F-Secure (F-Secure)	Malwared (Malware Must Die)
	G DATA Software (GData)	Malware Domain Blocklist (DNS-BH - Malware Domain Blocklist)
	Hacksoft (The Hacker)	Malware Domain List (Malware Domain List)
	Hauri (ViRobot)	MalwarePatrol (MalwarePatrol)
	Ikarus Software (Ikarus)	Malwares.com (Saint Security)
	Invincea (X by Invincea)	Netcraft (Netcraft)
	INCA Internet (nProtect)	OpenPhish (FraudSense)
	Jiangmin	Opera (Opera)
	K7 Computing (K7AntiVirus, K7GW)	Palevo Tracker (Abuse.ch)
	Kaspersky Lab (Kaspersky)	Phishtank (OpenDNS)
	Kingsoft (Kingsoft)	Quttera (Quttera)
	Lavasoft (Ad-Aware)	Rising (Rising)

	Malwarebytes Corporation (Malwarebytes Anti-malware)	SCUMWARE (Scumware.org)
	Intel Security (McAfee)	SecureBrain (SecureBrain)
	Microsoft (Malware Protection)	Sophos (Sophos)
	Microworld (eScan)	Spam404b(Spam404)
	Nano Security (Nano Antivirus)	SpyEye Tracker (Abuse.ch)
	Palo Alto Networks (Palo Alto Networks (Known Signatures))	StopBadware (StopBadware)
	Panda Security (Panda Platinum)	Sucuri SiteCheck (Sucuri)
	Qihoo 360 (Qihoo 360)	ThreatHive (The Malwarelab)
	Rising Antivirus (Rising)	Trend Micro Site Safety Center (Trend Micro)
	SentinelOne (SentinelOne (Static ML))	Trustwave (Trustwave)
	Sophos (SAV)	urlQuery (urlQuery.net)
	SUPERAntiSpyware (SUPERAnti Spyware)	Virusdie External Site Scan (Virusdie LLC)
	Symantec Corporation (Symantec, Symantec Mobile Insight)	VX Vault (VX Vault)
	Tencent (Tencent)	Web Security Guard (Crawler, LLC)
	ThreatTrack Security (VIPRE Antivirus)	Forcepoint ThreatSeeker (Forcepoint)
	TotalDefense (TotalDefense)	Webutation (Webutation)
	Trend Micro (TrendMicro, TrendMicro-HouseCall)	Wepawet (iseclab.org)
	Trustlook (Trustlook Antivirus)	Yandex Safebrowsing (Yandex)
	VirusBlokAda (VBA32)	ZCloudsec (Zcloudsec)
	Zillya! (Zillya)	ZDB Zeus (ZDB Zeus)
	Webroot (Webroot)	Zeus Tracker (Abuse.ch)
	WhiteArmor (WhiteArmor)	Zvelo (Zvelo)
	Zoner Software (Zoner Antivirus)	

3. Αρχεία αναφορών

Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
C:\Users\m	1EFF048C	None		1 None	0	0	0	0	1	9
C:\Users\m	2246FA0E	None		2 None	0	0	0	0	1	7
C:\Users\m	C69A0F6C	None		2 None	0	0	0	0	1	12
C:\Users\m	54D44EDC	None		2 None	0	0	0	0	1	7
C:\Users\m	1CF6E12D	None		2 None	0	0	0	0	1	10
C:\Users\m	41D72FA6	None		2 None	0	0	0	0	1	7
C:\Users\m	3FA91F0F	None		2 None	0	0	0	0	1	7
C:\Users\m	4080BB3A	None		2 None	0	0	0	0	1	17
C:\Users\m	A1B9024E	None		2 None	0	0	0	0	1	15
C:\Users\m	356B21B7	None		2 None	0	0	0	0	1	5

Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
C:\Users\π	1EFF048C0	None		1 None	0	0	0	0	1	9
C:\Users\π	2246FA0E8	None		2 None	0	0	0	0	1	7
C:\Users\π	C69A0F6C6	None		2 None	0	0	0	0	1	12
C:\Users\π	54D44EDC(	None		2 None	0	0	0	0	1	7
C:\Users\π	1CF6E12DF	None		2 None	0	0	0	0	1	10
C:\Users\π	41D72FA62	None		2 None	0	0	0	0	1	7
C:\Users\π	3FA91F0F1	None		2 None	0	0	0	0	1	7
C:\Users\π	4080BB3A2	None		2 None	0	0	0	0	1	17
C:\Users\π	A1B9024EE	None		2 None	0	0	0	0	1	15
C:\Users\π	356B21B74	None		2 None	0	0	0	0	1	5
C:\Users\π	1A1998A6I	None		1 None	0	0	0	1	1	11

Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
C:\Users\π	2246FA0E8	None		2 None	0	0	0	0	1	7
C:\Users\π	C69A0F6C6	None		2 None	0	0	0	0	1	12
C:\Users\π	54D44EDC(	None		2 None	0	0	0	0	1	7
C:\Users\π	1CF6E12DF	None		2 None	0	0	0	0	1	10
C:\Users\π	41D72FA62	None		2 None	0	0	0	0	1	7
C:\Users\π	3FA91F0F1	None		2 None	0	0	0	0	1	7
C:\Users\π	4080BB3A2	None		2 None	0	0	0	0	1	17
C:\Users\π	A1B9024EE	None		2 None	0	0	0	0	1	15
C:\Users\π	356B21B74	None		2 None	0	0	0	0	1	5

Path	hash	exists	vthash	upload	found	signed	resource	image	packer	yara
C:\Users\π	2246FA0E8	None		2 None	0	0	0	0	1	7
C:\Users\π	08B9F5874	None		2 None	0	0	1	0	0	11
C:\Users\π	C69A0F6C6	None		2 None	0	0	0	0	1	12
C:\Users\π	12C42B7FE	None		2 None	0	1	1	0	0	11
C:\Users\π	54D44EDC(	None		2 None	0	0	0	0	1	7
C:\Users\π	D83E1FFA2	None		2 None	0	0	0	0	0	6
C:\Users\π	6016DCA9:	None		2 None	0	0	0	0	0	9
C:\Users\π	1CF6E12DF	None		2 None	0	0	0	0	1	10
C:\Users\π	41D72FA62	None		2 None	0	0	0	0	1	7
C:\Users\π	3FA91F0F1	None		2 None	0	0	0	0	1	7
C:\Users\π	4080BB3A2	None		2 None	0	0	0	0	1	17
C:\Users\π	A1B9024EE	None		2 None	0	0	0	0	1	15
C:\Users\π	356B21B74	None		2 None	0	0	0	0	1	5
C:\Users\π	457151815	None		2 None	0	0	0	0	0	7
C:\Users\π	CE527E4B9	None		2 None	0	0	0	0	0	7
C:\Users\π	9D78DE80{	None		2 None	0	0	0	0	0	13
C:\Users\π	5741AF8CC	None		2 None	0	1	0	0	1	14
C:\Window	1FB37E98D		2	2 None	0	1	1	0	0	9
C:\Window	2E184807B		2	2 None	0	1	1	0	0	8
C:\Window	294AA1ABI		2	2 None	0	1	1	1	1	9
C:\Window	2132A26AI		2	2 None	0	1	1	0	1	4
C:\Window	0EE26ADFE		2	2 None	0	1	1	1	1	7
C:\Window	086E50434		2	2 None	0	1	1	0	1	4