UNIL | Université de Lausanne
Ecole des sciences criminelles
bâtiment Batochime
CH-1015 Lausanne

Maîtrise universitaire ès Sciences en science forensique
Orientation investigation et identification numériques

Mémoire de maîtrise

Windows Server forensics and incident response for the ProxyShell Exploit

Léo Giannardi

Under the supervision of Pr. Frank Breitinger
Expert: Johann Polewczyk

June 2022

UNIL | Université de Lausanne
Ecole des sciences criminelles
bâtiment Batochime
CH-1015 Lausanne

Tél.+41 21 692 46 00 | Fax.+41 21 692 46 05 | info.esc@unil.ch | www.unil.ch/esc

Abstract

In the course of 2021, a new attack surface on Microsoft Exchange Server was largely exploited all over the world. Mail servers are a valuable service, on which companies constantly rely, making it a prime target for cyberattacks. The present thesis focuses on the ProxyShell vulnerability chain, breaking it down step by step. Next, after having reproduced an attack scenario exploiting these vulnerabilities on a functioning Exchange Server, traces generated by the attack are highlighted and discussed from a forensically aware incident-response perspective. Additionally, the forensic images, the lab environment as well as the tools and scripts used to reproduce the attack are provided for educational purposes.

Keywords

Microsoft Exchange Server, ProxyShell Exploit, Forensic Analysis, Incident Response

Résumé

Au cours de l'année 2021, une nouvelle surface d'attaque sur Microsoft Exchange Server a été largement exploitée dans le monde entier. Les serveurs de messagerie sont un service précieux, sur lequel les entreprises s'appuient constamment, ce qui en fait une cible de choix pour les cyberattaques. Le présent mémoire se concentre sur la chaîne de vulnérabilité nommée ProxyShell, en la décomposant étape par étape. Ensuite, après avoir reproduit un scénario d'attaque exploitant ces vulnérabilités sur un serveur Exchange en fonctionnement, les traces générées par l'attaque sont mises en évidence et discutées d'un point de vue forensique et de réponse à incidents. En outre, les images forensiques analysées, l'environnement de laboratoire mis en place ainsi que les outils et les scripts utilisés pour reproduire l'attaque sont fournis à des fins pédagogiques.

Mots-clés

Microsoft Exchange Server, ProxyShell Exploit, Analyse forensique, Réponse à incidents

Table of contents

Introduction

During the 2021 edition of Black Hat conference, security researcher Orange Tsai demonstrated a brand-new exploit called "ProxyShell" to remotely attack on-premises Microsoft Exchange servers. After the vulnerability was publicly exploited in August 2021, thousands of unpatched Exchange servers were attacked and as of today, thousands are still vulnerable to the exploit chain.

In this context, the present thesis focuses on the forensic analysis of a compromised environment due to this exploit. First, state-of-the art forensic analysis and incident-response frameworks will be presented through a literature review. In a second step, and with a more practical approach, the ProxyShell vulnerability chain will be deconstructed step by step and, after having set up a functioning lab environment including a vulnerable Exchange server and having established an experimental attack scenario to make use of the exploit, the forensic analysis of the compromised Exchange server machine will be conducted, and evidence highlighted at each step of the analysis will be associated with each step of the attack scenario before being discussed in terms of relevance and added value.

Finally, the lab environment, the scripts used to conduct the attack as well as the forensic images used during the investigative process will be provided as educational material.

Objectives

Being one of the most exploited vulnerability chains of the year 2021, the ProxyShell exploit, targeting Microsoft Exchange servers all over the world, has been responsible for numerous security incidents on Active Directory domains, leading from data exfiltration to data encryption due to ransomware usage. Since the exploit used to attack a vulnerable environment is decisive regarding the privilege obtained on the infected network, hence the potential damage that can be done, appropriate incident response-related forensic analysis is necessary to assess the situation and potential risks after compromission. Thus, the present thesis focuses on the ProxyShell exploit chain itself and its reproduction in a controlled environment, as well and its forensic analysis.

The first axis of the present thesis revolves around the setup of a functioning environment that would be vulnerable to the exploit chain. In parallel, an attack scenario exploiting the vulnerabilities is constructed and played out in the test network. Then, the forensic consideration of such a situation allows to articulate the following research questions:

- What traces and artefacts are generated by the attack of a Windows Server machine hosting a vulnerable Microsoft Exchange server through the ProxyShell vulnerability chain exploit?
- What information can these traces give to a forensic investigator in an incident response situation?

Additionally, the final purpose of the present thesis is to provide educational material regarding Active Directory domains and Microsoft Exchange, the ProxyShell vulnerability chain, forensic analysis from an incident response perspective.

One initial objective of the present thesis would have been, in a second time, to use the deployed lab environment as a honeypot mail server in order to determine the risk level of the exploit nowadays, as well as try and obtain most realistic traces that would have been generated by a real-world attack on the mock mail server. However, due to security reasons, this part of the research project was impossible to conduct within the framework of the university.

Literature review

Digital forensics

Kent et al. (2006) propose a forensic process applying to digital forensics, which divides in four distinct steps, illustrated in **Figure 1** below:

1. Collection: The goal of the collection is to find any potential sources of data relevant to the incident, label them, and record them. Following that, the data contained in those sources should be obtained while maintaining the sources' integrity.
2. Examination: This step consists of examining the data gathered during the collection phase and extracting the information relevant to the incident while maintaining its integrity.
3. Analysis: Analyse the data resulting of the examination mechanisms to answer the 5WH questions - Who, What, Where, When, Why and How - or assess whether a conclusion can be formed.
4. Reporting: Reporting is the process of producing and presenting the investigation's procedure, methodologies, and tools, as well as the findings from the analysis phase.

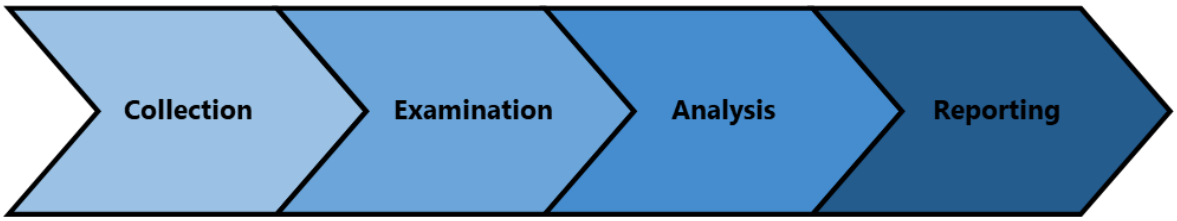

Figure 1: NISTs Forensic Process

Incident response methodology

Lee et al. (2013) give recommendations while undertaking on-site investigation for incident response on Windows environments after a cyberattack, providing the following methodology: first, the status of the victim's system must be assessed, before the investigator can go on and figure out the attack path and identify if malware has been installed on the system by performing disk and memory analysis. The latter can also bring useful information regarding potentially malicious processes run on the victim's system. Network analysis should be performed to monitor malicious traffic as well. Finally, log file analysis is necessary to be able to fully understand the attack.

Malware investigation

Hutchins et al. (2011) present an Intrusion Kill Chain model, which is derived from the U.S. military's kill chain, as an intelligence-driven strategy to be followed in the detection and prevention of cyberattacks and describe the phases adversaries must go through to reach their goals. It summarizes as follows:

1. Reconnaissance (R): The attacker typically uses the Internet to locate, identify, select, and collect information about his target.
2. Weaponization (W): A seemingly valid file is created and given to the target. A payload (malicious code) is utilized to infect the target with this file.
3. Delivery (D): The attacker sends the target the payload.
4. Exploitation (E): The payload is executed by exploiting operating system or installed application vulnerabilities.
5. Installation (I): To ensure persistence, the payload is put in a specific location in the system, on the hard drive or memory of the infected system.
6. Command and Control (C2): To acquire access to the target, the payload establishes a covert communication channel with its originator.
7. Actions on Objective (A): The attacker accomplishes his objectives.

When facing a cyber incident situation, regardless of the step of the kill chain at which the attack is detected, its analysis is crucial for finding out the intrusion phase, allowing to assume that the prior phases have already been executed, and to conduct a thorough analysis on each of them. Thus, by collecting as much data related to the detected intrusion and thoroughly analysing previous phases, actions can be taken to prevent future intrusions.

According to Carvey (2014), an adequate response to malware infection requires a comprehensive understanding of its characteristics, and he summarizes malware into four distinct components: initial infection vector, propagation mechanism, persistence mechanism and artefacts.

Disk Forensic Process

Yaacoub et al. (2021) describe a computer forensics framework considering several steps during the investigative analysis of computer storage area. They propose the following steps:

1. Developing policies & procedures with standard guidelines to support the investigation.
2. Assessing evidence to better understand and grasp the potential evidence.

3. Acquiring evidence in a legal and thoroughly documented way.
4. Examining evidence and analysing it.
5. Documenting & reporting the methods of data acquisition and analysis, potential modification made on the investigated data while focusing primarily on preserving file integrity.

Alazab et al. (2009) describe the strengths of NTFS forensic analysis as follows: "[It allows to] preserve the digital crime scene, obtain the information in slack space, access unallocated space, free space, and used space, recover file fragments, hidden or deleted files and directories, view the partition structure and get date-stamp and ownership of files and folders." Plus, they propose to divide the forensic analysis of an NTFS disk image into the three following steps:

1. Hard disk data acquisition
2. Evidence searching
3. Analysis of NTFS file system.

Memory Forensic Process

Memory forensic can give useful information regarding running processes on the analysed machine, as well as open files, information on network traffic, Internet data, passwords and cryptographic keys, decrypted content that would otherwise be encrypted on the disk and so on (Amari, 2009). As RAM data is volatile by definition, either its capture or its analysis have a significant entropy, thus making the acquisition mechanism critical in the impact it has on the memory itself – if, for instance, it is done using software running on the to-be-acquired system, the capture will alter the memory – as well as the potential traces found in its subsequent analysis (Hausknecht et al., 2015).

Regarding memory analysis, Hausknecht et al. (2015) indicate that "Volatility Framework became prominent as the main tool for "deep" respectively thorough analysis".

Network Forensic Process

Qureshi et al. (2021) review network forensic tools helping to collect and analyse data in the context of digital investigation, such as Wireshark, Tshark, Burp Suite, and tcpdump. In parallel, they have developed a network forensic methodology named OSCAR, standing for the following:

1. Obtaining information regarding the incident, the environment and context.

2. Strategizing by formulating an action plan on how to carry out the investigation, taking account of different sources of evidence and prioritize.
3. Collecting evidence following the previously set up prioritization plan.
4. Analysing evidence.
5. Reporting the results of the previous steps in a synthesized manner.

Pilli et al. (2010) propose the following subdivision of traffic collection methods regarding network forensics:

1. "Catch-it-as-you-can": All packets passing through a particular traffic point are captured and analysis is subsequently done, requiring large amounts of storage.
2. "Stop-look-and-listen": Each packet is analysed in memory and certain information is saved for future analysis.

ProxyShell

In March 2021, the exploit of a vulnerability chain on Microsoft Exchange Server left a huge number of Exchange servers vulnerable. The vulnerability was reported on January 5, 2021, by Orange Tsai from DEVCORE Research Team, who named it "ProxyLogon", which prompted Microsoft to release patches for Exchange 2013, 2016, 2019 versions, and even patched the vulnerability for the deprecated Exchange 2010 version on March 2nd (MSRC Team, 2021). As of March 5, according to Brian Krebs from KrebsOnSecurity, "At least 30,000 organizations across the United States [...] have over the past few days[1] been hacked" (KrebsOnSecurity, 2021). Alternate sources report over 125,000 unpatched Exchange servers as of March 9 (Unit42, 2021).

In April 2021, Orange Tsai demonstrated and reported another chain of vulnerabilities in the Microsoft Exchange architecture during the Pwn2Own Vancouver 2021 contest, and this time named it "ProxyShell". Once again, since the attack source is exploiting logic bugs in the implementation of Exchange, it left all unpatched servers vulnerable by default. Microsoft released patches for these specific CVEs in May and July 2021[2] (Orange Tsai, 2021a).

However, numerous Exchange servers remained unpatched and, according to Shodan research, as of March 16, 2022, still more than 22,000 servers are vulnerable to the ProxyShell attack worldwide (Shodan, 2022).

[1] The publication is dated March 2021.

[2] (Microsoft, 2021a, 2021b, 2021c).

What made such vulnerabilities particularly critical is that, besides targeting one of the most common mail server solutions, their exploitation could lead an unauthenticated attacker to execute arbitrary commands on the server through an exposed 443 port. Survey conducted by Orange Tsai shows that more than 400,000 servers are exposed to such attacks on the Internet (Orange Tsai, 2021b).

Moreover, even though patches were available, companies and security actors mostly became aware of the attack's criticality in August 2021 only: Switzerland's NCSC[3] reported the vulnerability on August 9; both Sophos[4] and Malwarebytes[5] published guidelines on August 23, France's CERT-FR[6] posted an alert bulletin on August 27.

Regarding guideline and solutions given by such editors, both Swiss and French state publications only advise to apply corrective patches to vulnerable servers, while, in this case, Malwarebytes offers more insight on actual attacks and briefly ransomware as the main threat for vulnerable systems. On the other hand, the news article published by Sophos offers a more detailed outline of the situation, describing the attack's overview, giving mitigation advice and investigate post-infection exposure of the potentially compromised environment.

ProxyShell IoCs

Hammond (2021) propose a list of indicators of compromise (IoCs) to identify an Exchange server as compromised by ProxyShell exploit, such as system configuration files on the disk, created web shells on specific file paths, IP connections and specific User-Agents.

MSRC Team (2021) offers a guideline to investigate compromised servers, as well as indicators of compromise for similar attacks.

[3] (Centre national pour la cybersécurité NCSC, 2021).
[4] (Iddon, 2021).
[5] (Arntz, 2022).
[6] (CERT-FR, 2022).

Methodology

Architecture

The present thesis focuses on the forensic analysis of a Microsoft environment being compromised by the ProxyShell exploit, having as objectives to point out and describe indicators of compromise, traces generated by the attack and their respective source, localization and way to identify them. In that respect, a vulnerable virtualized lab environment was set up and an attack scenario was played out exploiting the vulnerability. The following section describes in detail the environment considered.

The full network consists, first of all, of a Microsoft domain built to simulate a working and simplistic implementation of an Active Directory and mail server architecture. Therefore, two Windows Server 2012 R2 virtual machines, one domain controller and one member server on which Microsoft Exchange 2016 (version 15.1, build 2106.2) is installed were set up. A fully functional Microsoft domain mocking an enterprise network was set up as part of the research. Hence the fictional "ITECT.lan" domain was created, composed of two servers (one domain controller named lglab-dc1 and one server hosting Exchange, named lglab-mail) as well as a host machine, named lglab-host. Two users are members of this fictional domain: Ali Gator, Mike Rosophte and one administrator account (named Administrator). The whole environment is internally connected on the same network subnet (IP range 10.100.10.0/24) and the unpatched Exchange server is accessible only from the inside of this subnet for security reasons. In order to simulate external access to the server, the network traffic is routed through the Linux machine lglab-niffler, configured with iptables to forward packets. To handle network capture, this machine is set as a router configured with two distinct network interfaces, one belonging to the same subnet as the Microsoft domain, the other being only assigned to the lglab-niffler router and another Kali Linux machine, named lglab-cc, put up as a command centre to conduct the ProxyShell attack from. As one of the objectives of the attack scenario was to exfiltrate data, a custom web server configured to handle POST requests from a client was set up on the lglab-cc machine. More detailed configuration settings are described in the following section.

The following table **(Table 1)** lists and details the environment, and the following figure **(Figure 2)** illustrates the network architecture set up for this work:

Operating system	Machine name	Role	IP address(es)	Gateway
Windows Server	lglab-dc1	Domain Controller	10.100.10.125	10.100.10.1
Windows Server	lglab-mail	Mail server	10.100.10.126	10.100.10.128
Windows 10	lglab-host	Host machine	10.100.10.127	10.100.10.128
Kali Linux	lglab-niffler	Router	10.100.10.128; 10.200.10.1	10.100.10.1
Kali Linux	lglab-cc	Command centre	10.200.10.2	10.200.10.1

Table 1: Network topology and machines roles

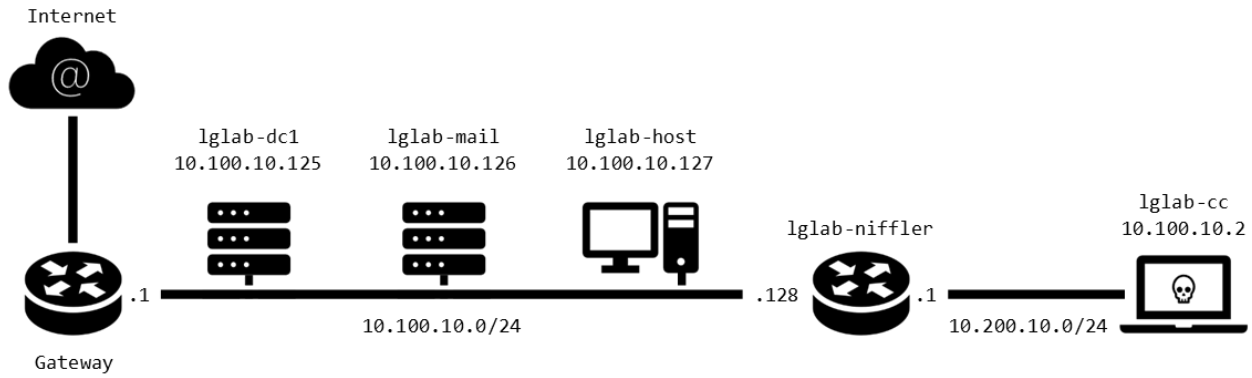

Figure 2: Network topology

Tools and techniques

Lab environment

Regarding the environment virtualization, VMWare ESXi[7] 6.0.0 was installed on a physical HP blade server and all the subsequent VMs are installed on it. General network and subnet configuration was done using the networking utility of the ESXi hypervisor. NAT configuration between the machines, as to route packets through both subnets, was handled with iptables[8] on the lglab-niffler machine.

[7] VMWare ESXi.
[8] IPTables.

Attack Simulation

The attack scenario played out in the context of the present thesis originates from the lglab-cc machine. To do so, a Python-developed attack script was used (see Appendix [1]). Similarly, the web server put up to handle POST requests containing the exfiltrated data was developed using Python (see Appendix [2]). The detail of the attack is described in the following section.

For research purposes, Sophos provided with the PowerShell commands used by the attackers. As part of the attack reproduction, those commands were tweaked as to fit the present thesis's network environment and passed in the scenario (see Appendix [3]).

Forensic Analysis

One of the objectives of the present work being the highlight of traces generated by the ProxyShell attack, a forensic copy of respectively the network traffic in and outgoing the server, its RAM and disk storage was taken in the following way. The network capture and analysis were conducted using Wireshark[9]. In a first instance, test captures were made using the lglab-niffler as a web proxy, configuring web requests forwarding between the attacking machine lglab-cc and the mail server lglab-mail using mitmproxy[10]. This allows to intercept and decrypt HTTPS traffic, as traffic destined to the mail server is redirected to the proxy. However, this method would have only recorded partial traffic. Thus, the solution of setting up the lglab-niffler machine as a router was chosen for practicality, allowing to record the entirety of network traffic. Thus, two packet captures were taken while performing the attack, at each network interface of the mock router lglab-niffler, one facing towards the 10.200.10.0/24 subnet, the other facing towards the 10.100.10.0/24 subnet. lglab-niffler being the default gateway of the attacker machine (lglab-cc) on the first subnet and of the mail server (lglab-mail) and the host machine (lglab-host) on the second, the recorded traffic involves all routed packets coming from and destined to those machines during the time of the attack. The lglab-host machine was integrated to the lab architecture to generate background noise during the network capture. It is indeed connected to the Internet and, during the experiment, had an open session on which a browser was accessing the https://www.20min.ch/fr website. Memory capture was taken first in a clean and normal-behaving environment and after the attack. In parallel, a forensic copy of the disk storage was done before and after the attack. Practically, as

[9] Wireshark.
[10] mitmproxy.

the whole environment is virtualized, snapshots of each VMs were taken and respectively .vmem memory dumps and .vmdk virtual disk were analysed. However, while analysing the memory capture saved while taking a snapshot of the virtual machine, using the ESXi utility, it appeared that the results were partial, and the content of the analysed RAM was degraded. Practically, the analysis using the Volatility framework revealed incomplete results, the RAM profile being only partially recognized. Thus, to ensure optimal results, the whole experiment was replayed and this second time, memory capture was handled within the virtual machine running operating system, using the WinPmem[11] memory capture software standalone from an external drive, as to create the smallest amount of contamination. Consequently, the network, memory and disk images analysed result of a second attack simulation. Plus, while conducting the disk analysis of the machine's extracted incremental snapshot files, they were not recognized as such. Thus, the virtual machine's disk was consolidated before being extracted as a single .vmdk file.

The network analysis was fully conducted using Wireshark. The memory capture was further analysed using the Volatility 2.6[12] framework as well as the hexadecimal editor HxD[13] for exploring the raw hexadecimal data. The disk analysis was conducted primarily using X-Ways Forensics[14], but external applications were used for the in-depth analysis of extracted files. Log Parser Lizard[15] was used to explore log files. USN-Journal-Parser[16] was used to analyse the operating system's USN Journal. MFT[17] was used to analyse the $MFT system file. LogFileParser[18] was used to analyse the $LogFile system file. HxD was used for exploring the raw hexadecimal data of some analysed files.

[11] WinPmem.
[12] Volatility.
[13] HxD.
[14] X-Ways Forensics.
[15] Log Parser Lizard.
[16] USN-Journal-Parser.
[17] MFT.
[18] LogFileParser.

Detailed attack description

Background

Microsoft Exchange 2016 and Active Directory

Microsoft Exchange Server is a platform for e-mail, calendaring, contact management, scheduling and collaboration developed by Microsoft. It is installed on the Windows Server operating system. Exchange Server was created by Microsoft to allow users to access the messaging platform from mobile devices, desktops, and web-based systems. Users of Exchange can collaborate by exchanging calendars and documents. Organizations may use the platform's storage and security features to archive content, conduct searches, and complete compliance activities, allowing the integration of SharePoint, Azure and other products (Nedjimi and Thobois, 2016).

Exchange 2016 relies primarily on Active Directory to obtain user account data essential to the proper functioning of the messaging system as well as to store Exchange-related data. Active Directory is a centralized directory-like database storing network objects and allowing to structure, organize and control network resources in Windows environments. Exchange uses Active Directory simultaneously to access and write user-related data and configuration parameters, using the LDAP protocol to query through its databases (Nedjimi and Thobois, 2016).

Exchange Server 2016 implements two distinct server roles, which are Mailbox servers and Edge Transport Servers. The present work will focus exclusively on Mailbox servers, which handle mailbox database and e-mail storage, mail routing and client access services (JoanneHendrickson, 2022; Nedjimi and Thobois, 2016).

Client Access Services

"The Client Access Services on Exchange Mailbox servers are responsible for accepting all forms of client connections. The Client Access (frontend) services proxy these connections to the back-end services on the destination Mailbox server (the local server or a remote Mailbox server that holds the active copy of the user's mailbox). Clients don't directly connect to the back-end services." (JoanneHendrickson, 2022). The communication is shown on **Figure 3** below.

The web implementation of the CAS is built on Microsoft IIS, the Microsoft .NET platform-specific web server. In the specific case of Exchange Server, "[...] there are two websites inside

the IIS. The 'Default Website' is the Frontend [...], and the 'Exchange Backend' is where the business logic is. [...] The Frontend is binding with ports 80 and 443, and the Backend is listening on ports 81 and 444. All the ports are binding with 0.0.0.0, which means anyone could access the Frontend and Backend of Exchange directly." (Orange Tsai, 2021b). Practically, both front and back-end are accessible to all network segments and interfaces.

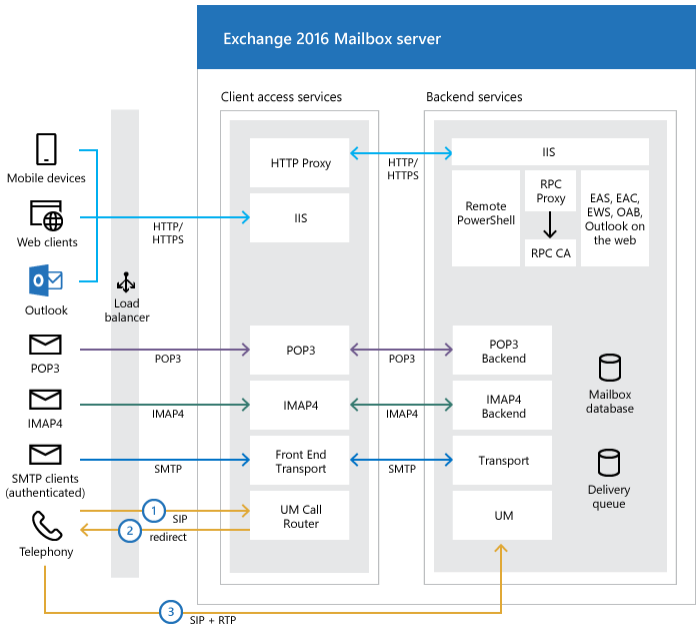

Figure 3: Exchange 2016 Mailbox Server architecture[19]

Autodiscover

"Autodiscover is the protocol used by the automatic discovery service that allows Outlook clients [...] to be configured automatically. With a reduced number of parameters, which are limited to the e-mail address and password, the user automatically recovers all the parameters necessary for his connection according to his context." (Nedjimi and Thobois, 2016)[20]. For internal clients within the Active Directory domain, the global process used is the following: The Outlook client attempting to connect through Autodiscover contacts an available domain controller via a LDAP request, which sends back the connections URL without any authentication. Then, the client sends an HTTPS POST request to the client access proxy of the Exchange Server via the URL previously obtained, and the server authenticates the Outlook client this way, allowing it to download configuration parameters and allow access.

[19] (JoanneHendrickson, 2022).
[20] Loose translation.

Server-Side Request Forgery

Placing itself at the 10th position of the OWASP top 10–2021, which documents the 10 most critical security risks to web applications, server-side request forgery defines the following behaviour: a web application fetches a remote resource without validating the URL provided by the user, even when protected by subsequent layers of security and access control lists. It allows a hacker to force the program to send a specially constructed request to an unexpected location (OWASP Top 10:2021, 2021).

"In a Server-Side Request Forgery (SSRF) attack, the attacker can abuse functionality on the server to read or update internal resources. The attacker can supply or modify a URL which the code running on the server will read or submit data to, and by carefully selecting the URLs, the attacker may be able to read server configuration such as AWS metadata, connect to internal services like HTTP enabled databases or perform post requests towards internal services which are not intended to be exposed." (OWASP, 2022).

The following diagram **(Figure 4)** schematically illustrates how the SSRF attack is conducted:

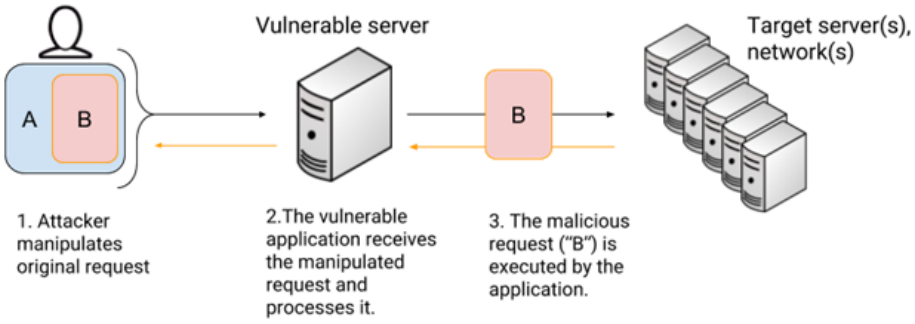

Figure 4: Server-side request forgery[21]

[21] (Shorebreak Security, 2022).

Detailed attack scenario

The section below describes and details step by step the actions taken on the server, first to exploit the vulnerability chain, then to conduct a realistic attack after having set foot on the infected server. Each command or attack step is documented in Appendix [3].

ProxyShell exploit

The vulnerability chain exploitation resulting in the so-called ProxyShell attack consists of several distinct steps:

- SSRF into the web server's backend to bypass authentication.
- Escalate privilege once authenticated.
- Create a Remote PowerShell session on the Exchange Management Shell.
- Arbitrarily write files on the server.

Once the high-privileged PowerShell session is up, arbitrary cmdlets can be passed, leading for instance to Exchange data exfiltration or, in this specific case, to the installation of a web shell in the server to launch additional attacks (e.g. ransomware upload, lateral movement in the network, more severe data exfiltration, data tampering).

First step of the attack is the exploit of CVE-2021-34473, also known as "Pre-auth Path Confusion". When a client HTTP request is classified as an Explicit Logon Request in the context of the Autodiscover service, its handler does an improper URI validation before granting access to the resources, resulting in Exchange normalizing the URL and removing part of it before forwarding it to the backend. This allows to bypass server authentication, which means accessing arbitrary backend URLs as the highest privileged entity in Windows systems, namely *NT AUTHORITY\SYSTEM*. However, contrary to the SSRF in ProxyLogon, it cannot be used solely, but can be used in conjunction with other vulnerabilities. In this specific case, it is exploited to obtain specific credentials on the domain, namely the LegacyDN credential, used to identify a user mailbox. The recovery of LegacyDN being the first SSRF in this context, it will be referred to as **ProxyShell part 1** below. The successfully recovered LegacyDN can then be used as a parameter for a second SSRF attack to obtain a valid user security ID (SID) and modify it to an administrator where appropriate, that is, if the SID is not already an administrator. Practically, this means setting the RID value to "500". This second SSRF will be referred to as **ProxyShell part 2** below.

The next step consists of exploiting the second vulnerability in the chain, CVE-2021-34523: "Elevation of Privilege on Exchange PowerShell Backend". Once the administrator SID is obtained, it is possible to create an Exchange PowerShell Remoting session via the Autodiscover website, considering that, in the vulnerable Exchange PowerShell backend implementation, user identity can be specified in the URL by tweaking the X-Rps-CAT token. Typically, such interaction would happen only for internal backend communication, but assuming that there is direct backend access, this can be used to authenticate as an Exchange administrator within the Active Directory domain, meaning the possibility to execute arbitrary Exchange PowerShell cmdlets as an administrator. Practically, two HTTP requests were sent to the server to test and forge a valid X-Rps-CAT token, then the same token was used to set up a Remote Exchange PowerShell session. This part of the exploit will be referred to as **ProxyShell part 3** below.

Last part of the exploit chain is a post-authentication remote code execution allowed by arbitrary file write using Exchange PowerShell commands (CVE-2021-31207). One way to achieve this goal is to export the content of a user's mailbox to an arbitrary file in an arbitrary file path on the server. Practically, this is done by invoking the New-MailboxExportRequest cmdlet.

Hence it is possible to send malicious code, in this case a web shell, through SMTP to the desired mailbox in the domain and export it. However, standard mailbox exports are usually stored in PST format, which is encoded with Permutative Encoding[22]. Thus, in order to make the payload interpretable it has to be pre-encoded before being sent, turning it into the original code once exported. The mail sending will be referred to as **ProxyShell part 5** below.

Practically, exporting the web shell to the Exchange webroot as an ASP.net website allows the execution of a Windows Command Prompt (cmd.exe) as *NT AUTHORITY\SYSTEM*, which basically grants access to the whole machine. This part of the exploit will be referred to as **ProxyShell part 5** below. **Figure 5** below illustrates and summarizes the exploit, and practical requests sent for each part previously described can be found in the attack script (See Appendix [1]).

[22] Permutative Encoding.

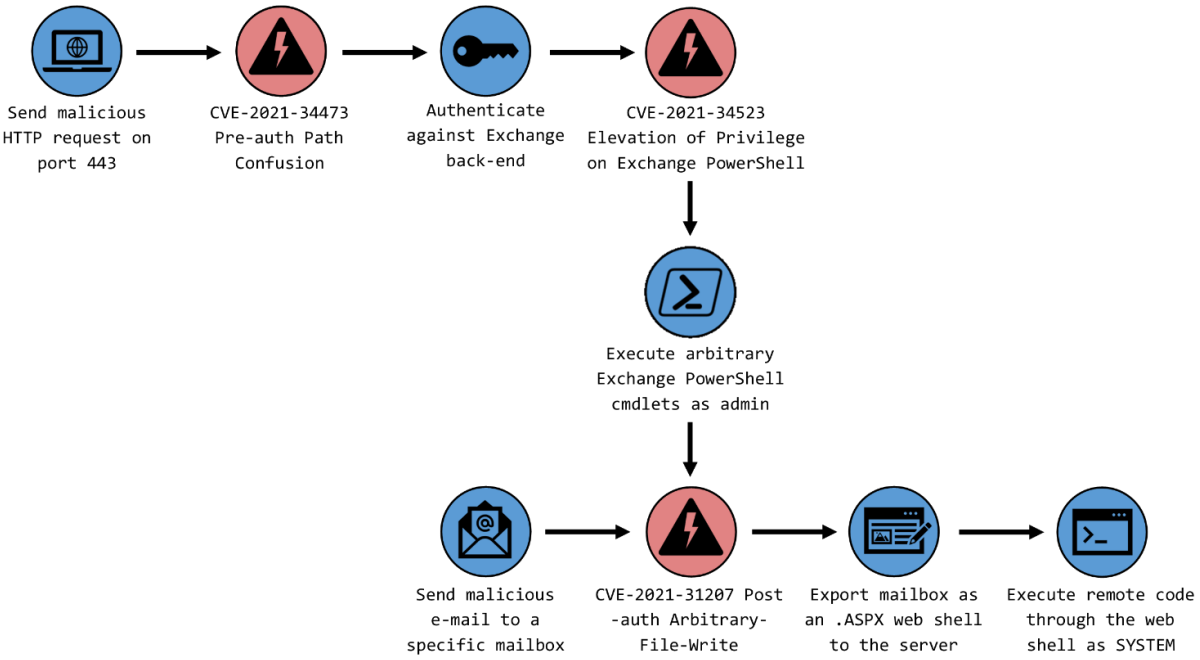

Figure 5: Illustration of the generic ProxyShell exploit

Post-ProxyShell operations

With the aim of being as credible as possible while conducting the attack, further operations were undertaken after the initial infection, still as part of the experiment. Thus, findings by Gallagher and Mackenzie (2021) described the behaviour of attackers affiliated to Conti ransomware as well as their attack method. Their investigation of a compromised environment revealed the following results: "[…] the Conti affiliates managed to gain access to the target's network and set up a remote web shell in under a minute. Three minutes later, they installed a second, backup web shell. Within 30 minutes they had generated a complete list of the network's computers, domain controllers, and domain administrators. Just four hours later, the Conti affiliates had obtained the credentials of domain administrator accounts and began executing commands." To simulate real-world behaviour, tweaked PowerShell commands derived from actual commands leaked by Conti were passed to the web shell to conduct network and filesystem reconnaissance on the infected server before dumping the memory of the server's LSASS.exe service, responsible for user identification on the machine, thus containing the password hashes of every user having run a session on the machine since the service's execution start. Finally, the memory dump was sent to a remote web server with HTTP. The list and brief description of each command is detailed in Appendix [3]. Each of them will be referred to as **PowerShell command n°[X]** below, [X] being the command number as stated in their respective appendix. Furthermore, **PowerShell commands n° 2 to 7** each output data

to a text file accessible through the network. Thus, this specifically created web page was accessed with the attacker machine (lglab-cc) to read the outputted data for each PowerShell command. They will be referred to as **Browser web accesses** below.

The main goal of the attack replication is to show that the ProxyShell vulnerability chain can be used to exfiltrate data from and upload data to the server, thus the scenario stops after having set up a webshell allowing binary files upload on the server and sending out the memory dump of the local authentication process (LSASS) of the server, which contains the NT hashes of all accounts logged onto the server. To summarize, the simulated attack results in the following:

- Local intrusion into the vulnerable mail server machine.
- Privilege escalation inside the Exchange framework, Exchange administrator rights are obtained while interacting with Exchange PowerShell, allowing arbitrary file write and remote code execution resulting in the delivery of a web shell on the machine.
- Local privilege escalation inside the mail server machine, SYSTEM rights are obtained through the previously dropped web shell.
- Arbitrary file write and remote code execution on the machine resulting in the writing of another web shell allowing to drop binary files onto the server. This second web shell was not actively used during the attack scenario.
- Network reconnaissance and data exfiltration and domain credential dump giving the ability to the attacker to crack dumped passwords and escalate privilege, this time at the Active Directory Domain level.

Traces, artefacts and digital evidence

The general investigation of the traces left by attacking the mail server was performed following a general-to-specific mindset, focusing on the examination of the three captures obtained by respectively recording network traffic during the attack, imaging the memory prior and after the attack was carried out, and imaging the virtual disk of the lglab-mail mail server virtual machine. As the attack scenario was known beforehand and deconstructed step by step, the analytical process proper to the attack reconstruction on the investigative side was short-circuited. Thus, the following section focuses on examining traces that have been generated by the attack found on collected data and associate them with specific steps of the exploit.

Furthermore, keeping in mind that real-world investigative processes regarding similar compromised systems have the objective of reconstructing the occurred events from the traces found, the following results, traces and locations highlighted are associated with each attack step described earlier, in order to provide an answer to the first research question of the present thesis, that is "What traces and artefacts are generated by the attack of a Windows Server machine hosting a vulnerable Microsoft Exchange server through the ProxyShell vulnerability chain exploit?". Additionally, for each evidence support analysed, relevant traces and locations found are discussed and evaluated in order to answer the second research question, that is "What information can these traces give to an investigator in an incident-response situation?".

Network

Network traffic forensic analysis in an incident response situation is conducted to monitor and identify potentially malicious traffic transiting through the observed network. Thus, abnormal amounts of packets, unexpected traffic between machines, in and outgoing, unforeseen protocol usages, repeated connections with unknown IP addresses are actively searched. In this case, a network capture was done on the simulated router machine, lglab-niffler, containing all routed packets coming from and destined to all connected machines, namely the attacker machine (lglab-cc) on the first subnet and of the mail server (lglab-mail) and the host machine (lglab-host) on the second. The latter machine was integrated to the network to generate background noise while performing the network capture.

Network Statistics

Table 2, created with data coming from the capture of the router's interface facing towards the 10.100.10.0/24 subnet, shows the 10 most heavy IP to IP conversations in terms of throughput during the recording. The top two exchanges both represent communication between lglab-mail and lglab-cc – the attacker machine's IP address being substituted to the router's IP due to configuration – during the time of the attack, clearly standing out from the background noise.

Address A	Address B	Bytes	Bytes A → B	Bytes B → A
10.100.10.126	10.200.10.2	45,320,831	44993335	327,496
10.100.10.128	10.100.10.126	4,919,802	241,567	4678235
216.58.215.238	10.100.10.128	1,141,774	230461	911,313
10.100.10.127	193.246.48.171	87274	5408	81,866
10.100.10.128	193.246.48.171	87046	5270	81,776
18.185.159.166	10.100.10.127	83546	76365	7181
18.185.159.166	10.100.10.128	83447	76503	6944
10.100.10.127	151.101.65.108	69596	2821	66,775
10.100.10.128	151.101.65.108	69,284	2707	66,577
10.100.10.128	142.250.203.99	63649	13,607	50,042

Table 2: 10 most heavy conversations on the network

Figure 6, created with data coming from the capture of the router's interface facing towards the 10.100.10.0/24 subnet, shows the time in function of the bytes transiting through the network. First, it shows a huge spike at t=325 seconds, which corresponds to 42.5 Mb of data transitioning nearly instantly at this specific time, almost all of it between lglab-mail and lglab-cc. This can be linked to the LSASS process dump exfiltration through HTTP, namely **PowerShell command n° 13**. Additionally, all 12 spikes reaching approximately 350 kilobytes correspond as well to network communication between the same machines, linking them to the **PowerShell command n° 1 to 12** passed through the web shell.

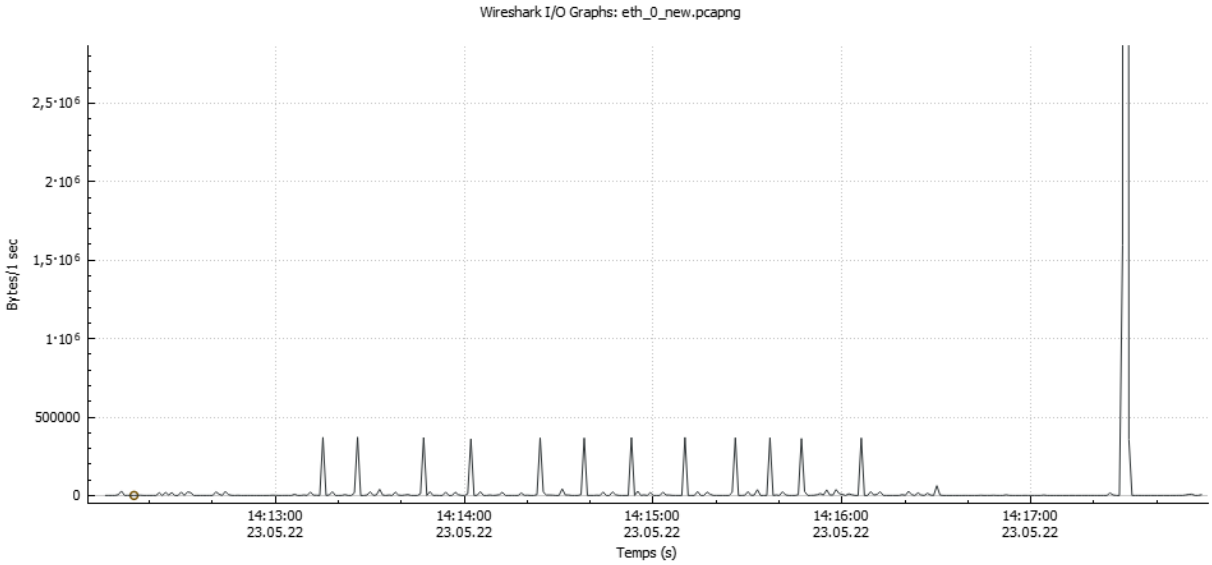

Figure 6: Bytes per second during the attack

To oppose to the total amount of traffic recorded on the network, **Figure 7**, created with data coming from the capture of the router's interface facing towards the 10.100.10.0/24 subnet, but filtering on packets having either as source or destination lglab-cc's IP address (respectively lglab-niffler's IP masquerading lglab-cc) shows the time in function of the bytes transiting through the network.

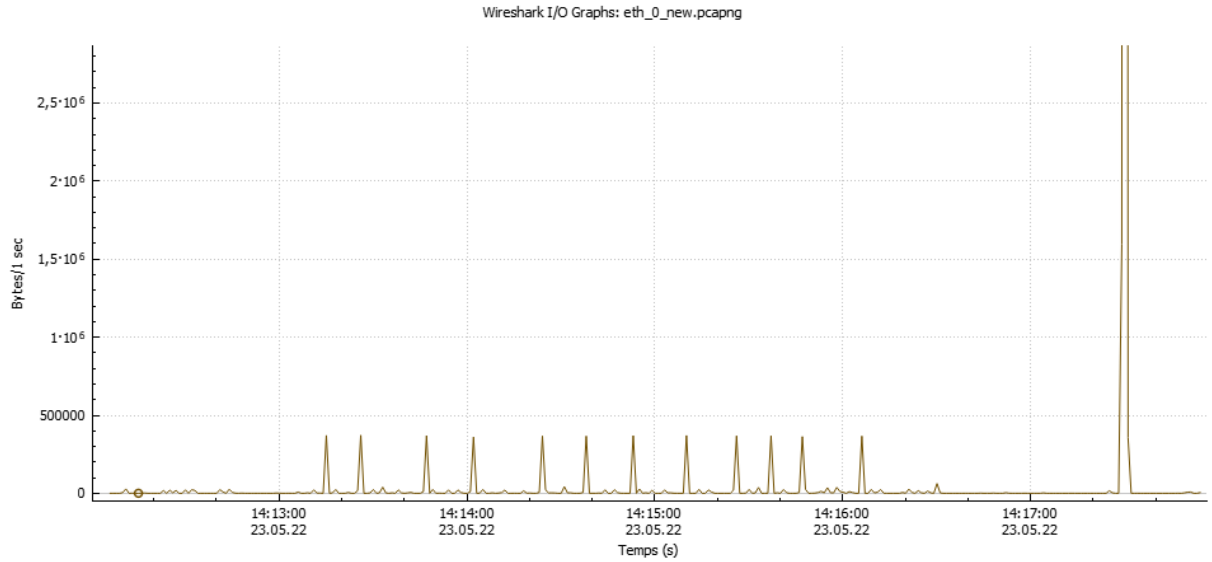

Figure 7: Bytes per second, filtering on lglab-cc

SMTP packets

On both captures, an SMTP stream can be highlighted between lglab-mail and lglab-cc, showing the mail sent by the attacker to the targeted user (See Appendix [4]). Several artefacts can be pointed out, notably the date and time of the mail sending, the source IP address, the source mail address (and destination), as well as the mail itself, containing the SMTP-encoded payload decoding as the web shell, corresponding to **ProxyShell part 5**.

HTTP POST Request

Appendix [5] shows the HTTP POST request passed to the attacker's web server to exfiltrate LSASS process memory dump, corresponding to **PowerShell command n° 13**. The request parameters can highlight several artefacts that could be useful to better understand the attack scenario from an investigative point of view, such as the User-Agent, showing the client application used to pass the request – in this case: PowerShell. Moreover, the server's HTTP response shows the web server type, in this case, a Python web server.

Discussion

Network forensic analysis, conducted on data collected under the "Catch-it-as-you-can" collection model, allowed to obtain relevant results regarding network traffic traces generated by the attack.

First, anomalous network traffic was detected. Indeed, the amount of data transmitted in TCP packets through the network can be highlighted as malicious. In this case, the network behaviour at the time of the capture is significantly different from normal, thus causes of the perturbations can be assessed while monitoring the network and identifying the connections causing this amount of traffic overhead.

TCP connections between the mail server lglab-mail and lglab-niffler's IP address (masquerading lglab-cc) can be pointed out as responsible for such anomalous network behaviour. As this address is not within the internal subnet on which the mail server is configured, this must be investigated and the connected IP can be determined as at least unexpected, at most malicious.

Spikes detected in **Figure 6** are defining clear spikes detaching themselves from the background noise. Comparing it with **Figure 7** indicates that TCP connections between lglab-mail and lglab-cc are responsible for these data spikes. However, further interpretation solely

at the network level is difficult, still pointing these network anomalies can prove added value while conducting further investigations on the disk and/or the memory.

In parallel, outbound traffic using HTTP on port 80, directed to lglab-cc's IP address is detected on the network capture, which can be qualified as abnormal behaviour on the Exchange server and can lead to further investigation, giving insight on data exfiltration.

The body of the malicious SMTP e-mail sent to the target mailbox as part of **ProxyShell part 5** contains relevant data regarding the attack setup, that is the actual functioning web shell code. But as the code itself is encoded, and inbound SMTP traffic is common to witness on a mail server, it would be hard to intercept and interpret correctly in a real-world scenario.

Disk

The forensic analysis of the server's hard drive can be an entry point for incident response investigation on the compromised server. After identifying indicators of compromise, a thorough search and highlight of both active and latent files – and their respective metadata – associated with the attack on the disk image of the machine is conducted. Then, meta files proper to the NTFS file system are analysed.

File Creation

This section describes and details the files that are not normally present on a clean installation of a Windows server hosting Microsoft Exchange, thus being a result of the ProxyShell vulnerability chain exploitation. As the creation of all subsequent files is explicit in the previously described attack scenario, their presence on the disk is quite obvious, nevertheless they are evident indicators of compromise worth mentioning.

eabphi.aspx

Located at the web root of the IIS server *\inetpub\wwwroot\aspnet_client\eabphi.aspx*, this file is created as a consequence of **ProxyShell part 4.**

To be noted, its header starts with the **!BDN** magic word, identifying it as a PST file. Inspecting its content with a hexadecimal editor reveals functional code allowing arbitrary command execution on the server. This file's structure is similar to several public indicators of compromise and, as it allows remote code execution through an HTTPS connection to the web server, is characteristic of the CVE-2021-31207 vulnerability exploit.

ResourceHandler.aspx

Located at path *\Program Files\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\owa\auth\Current\themes\ResourceHandler.aspx*, this file is a direct consequence of **PowerShell commands n° 3 to 5.** It is a .aspx web page allowing, when accessed from a web browser, to drop binary executables on the target machine. As it is basically an embedded C# script in HTML code which is not obfuscated, it can be reverse engineered quite simply.

a.txt

Located at *\ProgramData\a.txt*, this file is a direct consequence of **PowerShell commands n° 3 to 6**, containing in the first place the content of the ResourceHandler.aspx web shell but emptied with later passed PowerShell commands. It shows as an empty text file while the analysis is performed.

test.txt

Located at *\inetpub\wwwroot\aspnet_client\test.txt* path on the web root of the IIS server, this file is a direct consequence of **PowerShell commands n° 2 to 7**, having contained in the first place the output of the commands, being accessible through the network. It shows as a text file which content is the "teset" string value while the analysis is performed.

a.zip

Located at the *\ProgramData\a.zip* path, this file is a direct consequence of the **PowerShell command n° 12.** Its content is the live memory dump of the local authentication process (LSASS) of the lglab-mail machine, exported with an extension mismatch as a *.zip* archive and contains the NT hashes of all accounts logged onto the server.

Windows Event Logs

Windows Event Logs serve as a primary source of evidence in forensic investigations because Windows itself logs all system activities. Such generated logs can be analysed to create a timeline based on the logging data and discovered artefacts.

Windows PowerShell.evtx

Located at *\Windows\System32\winevt\Logs\Windows PowerShell.evtx*, the Windows PowerShell event log file records details about PowerShell operations, such as engine start and stop or executed commands. In this case, the file logged information regarding **PowerShell**

commands n° 1 to 13, such as the log creation timestamp, corresponding to the time of the executed command on the machine, the event ID related to the command passed and the command itself as well as its parameters.

MSExchange Management.evtx

Located at *\Windows\System32\winevt\Logs\MSExchange Management.evtx*, the Microsoft Exchange Management Shell log file records details related to interactions with the Exchange Shell and executed cmdlets via the Shell. In this case, information regarding **ProxyShell part 4** are logged into the file, such as both EMS cmdlets passed through SSRF, namely the cmdlet assigning mailbox export rights to the impersonated user and the actual mailbox export cmdlet. The logged data also gives information about the user calling the cmdlet, its source (Remote PowerShell), the PID and name of the process handling the cmdlet, and the timestamp of the passed cmdlet.

IIS Logs

Log files generated by the IIS web server are stored in W3C Extended Log File Format. Such logs record HTTP requests sent against the websites handled by IIS. Each log file name contains by default its date of creation.

U_ex[YYMMDD].log

Log files generated by the IIS web server are located at *\inetpub\logs\LogFiles\W3SVC1*. During the period of interest, useful information can be extracted from IIS logs, focusing on the attacker's IP address.

Each step of the exploit can be pointed out in the log entries recorded, namely both SSRF against */autodiscover/autodiscover.json* to obtain respectively the targeted user's LegacyDN (**ProxyShell part 1**), and a valid administrator SID (**ProxyShell part 2**), HTTP GET requests to forge and test for a valid X-Rps-CAT token, establishment of the Remote Exchange PowerShell session (**ProxyShell part 3**) as well as interaction with the subsequently created */aspnet_client/eabphi.aspx* web shell and */aspnet_client/test.txt* text file. Moreover, the number of POST requests logged against the web shell and the text file correspond respectively to **PowerShell commands n° 1 to 13** passed through the web shell, and the **Browser web accesses** to the test.txt file through the attacker machine's web browser. The length of time that the action took, in milliseconds, is also logged as well as the User-Agent used to initiate

the requests and the URL query itself. However, the HTTP message body of each request is not logged (See Appendix [6]).

Exchange Logs

Microsoft Exchange Server stores numerous journal files recording established connections and actions taken on the server. Each log file name contains by default its date of creation, normalized in UTC+0, detailed in each of the following subsection with a generic naming consistency.

Rps_Http_[YYYYMMDDHH-X].LOG

Files located at *\Program Files\Microsoft\Exchange Server\V15\Logging\CmdletInfra\Powershell-Proxy\Http*, log data related to the */PowerShell* record web accesses on the Exchange Server. In this case, focusing on traffic with the attacker's IP address, the HTTP requests corresponding to **ProxyShell part 3**, that is the establishment of the elevated Remote PowerShell session from the attacker's machine are recorded. The impersonated user's e-mail address is associated with those logs. Furthermore, as for the IIS logs, the length of time that the action took, in milliseconds, is also logged as well as the User-Agent used to initiate the requests and the URL query itself. However, the HTTP message body of each request is not logged (See Appendix [7]).

Rps_Cmdlet_[YYYYMMDDHH-X].LOG

Files located at *\Program Files\Microsoft\Exchange Server\V15\Logging\CmdletInfra\Powershell-Proxy\Cmdlet* contain all PowerShell cmdlets passed to the Exchange Management Shell. In this case, HTTP requests corresponding to **ProxyShell part 4** can be found within the latest created file on the directory. The impersonated user account having passed the commands is associated with each record, and the cmdlet itself is logged (See Appendix [8]).

audit_[YYYYMMDD-X].LOG

File path *\Program Files\Microsoft\Exchange Server\V15\Logging\LocalQueue\Exchange* contains one audit log file per PowerShell cmdlet passed onto the Exchange Management Shell, corresponding to **ProxyShell part 4**. Thus, the impersonated user account having passed the commands is associated with each record, and the cmdlet itself is logged (See Appendices [9-10]).

HttpProxy_[YYYYMMDDHH-X].LOG

Files located at *\Program Files\Microsoft\Exchange Server\V15\Logging \HttpProxy\Autodiscover* path record requests against the Autodiscover service of the Exchange Server. Thus, HTTP requests corresponding to **ProxyShell parts 1 to 3** can be found within the latest created file on the directory. The HTTP request's URL query, the User-Agent used to initiate the requests as well as the client IP address are logged (See Appendix [11]).

Autod_[YYYYMMDDHH-X].LOG

Files located at *\Program Files\Microsoft\Exchange Server\V15\Logging \HttpProxy\Autodiscover* path record authentications against the Autodiscover service of the Exchange Server. Thus, the HTTP request corresponding to the successful authentication of the **ProxyShell part 3** process can be found within the latest created file on the directory. The authenticated user, in this case *NT AUTHORITY\SYSTEM*, the User-Agent used to initiate the request as well as the client IP address are logged (See Appendix [12]).

SEND[YYYYMMDDHH-X].LOG

The Microsoft Exchange transport pipeline, responsible for forwarding inbound SMTP traffic for the mail server, acting as a stateless proxy, records forwarded SMTP events and stores the logged information as comma-separated values at the following location: *\Program Files\Microsoft\Exchange Server\V15\TransportRoles\Logs\FrontEnd\ProtocolLog\SmtpSend*. Sender e-mail and IP addresses, as well as e-mail headers are recorded in this file (See Appendix [13]).

USN Journal

Windows keeps a metafile called $Extend$UsnJrnl in a special data stream called $J that keeps track of filesystem activities. This stream keeps track of the files that have been changed in the volume by storing records of file system operations. Its forensic analysis can be valuable

to obtain information regarding modified files, such as the file name, timestamp, and reason for modification on the volume. The analysis of the file system's USN journal allowed to point out the creation and successive modification timestamps of the previously analysed files that have been generated by explicit parts of the attack scenario, namely **ProxyShell part 4** and **PowerShell commands n° 2 to 7 and 12.** Thus, information regarding the truncation of the content of the *ProgramData**a.txt* file is logged in the USN journal, as well as regarding the modification of the *inetpub**wwwroot**aspnet_client**test.txt* file several times during the period of interest (See Appendix [14]).

$MFT

The Master File Table of $MFT is an NTFS metafile responsible for file and disk space allocation on the operating system as well as file metadata. Thus, its forensic analysis can provide useful file information such as file and folder creation dates, file modification dates, access dates, last written dates or physical and logical file size. $MFT analysis of the compromised server gave the following relevant result: file located at *inetpub**wwwroot**aspnet_client**test.txt* gets a standard entry in the $MFT. However, as an entry's size is typically 1024 bytes, files smaller than 600 bytes, also called resident files, are stored directly inside the $MFT. Regarding this particular file, its $MFT headers indicate a used entry size of 304 bytes. As the previous USN journal analysis highlighted, the content of the file was overwritten several times, which is a direct consequence of **PowerShell commands n° 2 to 7**. Manual carving in the file's $MFT entry allows then to recover part of the overwritten content, which is directly stored as resident at the $DATA attribute of the MFT (See Appendix [15]).

$LogFile

$LogFile is an NTFS volume log file which catching all changes to the NTFS metadata. Its analysis can be useful to keep track of file creation, deletion, renaming, copy, and so on. As each change on the system's metadata is first logged into the $LogFile before being committed to the actual system, it is possible to recover overwritten system data on the $LogFile. However, its main restriction is its maximum size, which by definition means that the oldest records are automatically overwritten once maximum size is reached. Thus, only recent changes made to the system are recoverable.

In this case, the overwritten content of the *\inetpub\wwwroot\aspnet_client\test.txt* for each overwrite during the attack scenario, being a direct consequence of **PowerShell commands n° 2 to 7**, can be pointed out on the $LogFile, as resident data attributes are present in this file (See Appendix [16]).

Discussion

The presence of unexpected files, especially the eabphi.aspx web shell, is clear indicators of compromise of the server. Presence of files at the web application root of the IIS server means those files are accessible through the network to which the web server is connected – in this case the internal subnets created within the lab framework, but in the real world, the Internet. The eabphi.aspx web shell's content is mostly human unreadable, the functional code being obfuscated amid encoded .PST data, as the web shell originates from an exported mailbox through the Exchange Management Shell. Also, the simple awareness of the files does not lead to anything regarding the actions taken on the compromised server. Furthermore, a.txt and test.txt files are nearly empty, which limits their potential forensic exploitability either in an investigative or evaluative process.

The analysis of the Windows Event log files provides useful information with respect to the investigative process of a potentially compromised machine due to the ProxyShell exploit. Automatically configured log files record data regarding PowerShell commands invoked through the web shell as well as Exchange Management Shell cmdlets passed; thus their presence allows to grasp all further actions taken on the already attacked server. The full presence of the very commands passed linked to their creation timestamp can help recreate a timeline of the events, this way contributing to the iterative process of forensic investigation. However, those logs are not directly linked to a specific user, IP address or file as identifier of the source of the command. Besides, Windows Event logs are often, in real world attack situations, simply erased by attackers after in a second step, making their forensic recovery and exploitation on an investigative process uncertain.

The consideration of the log files generated by the IIS web server can provide useful information regarding the initial infection vector that is the actual exploit of the ProxyShell vulnerability chain, since HTTP queries are logged, which means the resource accessed and the query parameters. Plus, further interactions with the eabphi.aspx web shell and the test.txt text file are logged as well, as they were accessed through the network, them being published by the IIS server itself. Consequently, the – attacker – client IP can be identified as well as the User-

Agent used to send the requests, which can be revealing of malicious activity. Nevertheless, since the HTTP body of each request is not logged, information provided by the analysis of IIS log files is only superficial.

Similarly, the investigation of Exchange server logs can provide useful information regarding the initial infection vector that is the actual exploit of the ProxyShell vulnerability chain, since HTTP queries are logged, which means the resource accessed and the query parameters. However, contrary to the previously described IIS log analysis, Exchange logs do not contain information related to other steps than the actual ProxyShell attack. Thus, their forensic consideration would only help to ensure the attack type underwent by the compromised server as well as identifying the source IP address responsible for the attack.

In the case of Windows Event Logs as well as Exchange logs, the last write and access timestamp of the log file itself precede some individual records write timestamp within the file itself. This can be misleading for an investigator during the log file analysis process, as some records written into the latest log file are posterior as the very file's last modification date. This inconsistency in dating timestamps is due to the difference in time zone consideration between the applications.

The forensic analysis of NTFS metafiles such as the USN journal, the $MFT file and the $LogFile file can prove useful to detect write and modification timestamps of specific files on the disk, in addition to file content recovery under certain circumstances. Therefore, if attackers are familiar with anti-forensic techniques and try to cover their traces after the exploit, for instance by deleting log files, tools used, or temporary files created while conducting their attack. Regarding the investigation of the ProxyShell exploit itself, timestamps of modification of previously described created files, with limited forensic usability, are recorded on the USN journal, contributing to the iterative process of forensic investigation.

Overwritten content of resident $MFT entries can be recovered partially in the $MFT itself, and fully inside the $LogFile metafile, but the intervention speed is crucial while considering the forensic investigation of the latter. Indeed, this file only keeps short-term records, which are themselves overwritten once the maximum size of the file is reached. Therefore, provided that the forensic investigation of the compromised machine – or at least the capture of the system's hard disk – is done in a reasonable amount of time, they can prove great usefulness if anti-forensic techniques are used to camouflage the actions taken by the attackers.

No further useful evidence was found regarding user account usage. As the CVE-2021-34523, granting Exchange administrator rights on the Exchange Management Shell is exploited remotely, it generates no further traces on the impersonated account's profile on the server system. Registry hives analysis, either system or user-related, revealed no results.

By default, Prefetch is disabled on Windows Servers, which induces a significative step back while performing the forensic analysis of the attack. Prefetches can be used while performing a forensic investigation to know which applications were executed on a system. Each .pf file will include the last time of execution, the number of times run, and device and file handles used by the program. Thus, considering the lab architecture as simulating an environment configured with little to no prior forensic awareness, their consideration in the present scenario was not an option.

To summarize, disk analysis of the compromised machine makes it possible to highlight each step of the attack performed against the Exchange server, thereby reconstructing the played scenario and identify the compromised assets of the Exchange server itself and the Active Directory domain, provided sufficient knowledge on the attack vector and post-exploit actions taken, hence giving them a substantial forensic weight. However, considered individually, the traces highlighted above do not provide an overall view of the level of compromise of the network nor of the attack path followed; therefore, it is necessary to pool traces from various sources to obtain a better view of the situation.

Memory

Memory analysis is a key step when investigating compromised machines due to malware activity. In this situation, malware is not directly executed on the target server, each and every step of the attack scenario being conducted using natively installed software. Thus, malicious process identification while analysing the memory capture is expected to deliver mitigated results.

Also, considering the fact that the system's memory capture was taken using a running executable on the machine, after the analysis of the virtual memory file generated by the VMWare ESXi hypervisor revealed incomplete results, it is expected to find memory contamination resulting of the execution of the memory capture software.

Volatility framework

Running process analysis and comparison between the memory capture taken respectively before and after the attack showed little to no apparent results. As a matter of fact, running processes are basically the same between both captures as well as their hierarchy, that is no unexpected child process call was identified on the compromised server's memory. However, an orphan powershell.exe process with start and exit timestamps corresponding to **PowerShell command n° 13** was present on the memory capture.

Despite no apparent different between running processes on the clean versus the attacked memory capture, several differences regarding dynamic link libraries (.dll) loaded by processes associated with the IIS web server of the Exchange server can be pointed out.

The analysis of the MSExchangePowerShellAppPool application pool of the w3wp.exe process shows that numerous .dll libraries are called and not yet freed from the memory in the post-attack capture. Notably, the rasapi32.dll and rtutils.dll, used in the context of Remote Access Services can be highlighted.

Similarly, the analysis of the MSExchangeAutodiscoverAppPool application pool of the w3wp.exe process also shows that the rasapi32.dll and rtutils.dll were loaded. Plus, the winhttp.dll, used to communicate with distant HTTP servers, loaded library can be found on the post-attack capture as well.

Following the memory analysis methodology with the usage of appropriate Volatility commands on the memory dump, the investigation of cmd.exe command line history did not give any results, neither did the analysis of network connections associated with the memory dumps investigated. Thus, connections established with the lglab-cc IP address are not explicitly stored in RAM.

Raw analysis and data recovery

After having run a thoroughly automated search using the Volatility framework, raw string search in the hexadecimal data of the memory capture made it possible to highlight the following data of interest found on the RAM:

- All **PowerShell commands n° 1 to 13** as well as their parameters.
- The overwritten content of the file *\inetpub\wwwroot\aspnet_client\test.txt* as found on the system file $LogFile.
- The content of the *\inetpub\wwwroot\aspnet_client\eabphi.aspx* web shell file.

- The content of each audit_[YYYYMMDD-X].LOG file, described in the previous section, containing data regarding each PowerShell cmdlet passed onto the Exchange Management Shell, corresponding to **ProxyShell part 4.**
- The content of the SMTP message sent at the **ProxyShell part 5.**
- Part of the content of the U_ex[YYMMDD].log file, described in the previous section, containing data corresponding to **ProxyShell parts 1 to 3, PowerShell commands n° 1 to 13** as well as **Browser web accesses.**

Discussion

The memory analysis of the compromised server allows to propose several conclusions regarding evidence found. First, the forensic consideration of memory evidence while investigating a vulnerability exploit is genuinely different from the investigation conducted while searching for malware execution on the RAM. Indeed, as, in this case, no external malicious process was executed on the compromised machine, the generic methodology based on the analysis of the process tree and execution location of suspicious processes is void while investigating exploited but truthful processes. Thus, while comparing process execution in a clean running environment and the post-attack memory capture using Volatility, few differences only could be brought to light. Regarding abused processes, loaded dynamic libraries link did not match between both captures, however, the forensic added value of such observation is thin.

Nevertheless, the post-attack memory capture was done shortly after the exploit, which could reveal, during the raw analysis of its raw hexadecimal content, the presence of the content of many files analysed in the previous section. As a result, memory analysis, in this case, could find its utility by providing a second source for evidence already highlighted during disk analysis. From a different angle, it can bring forensic added value if the attackers are forensically aware and apply anti-forensic techniques as file deletion, thus making the RAM analysis results associated to malicious files manipulation first-hand evidence. Yet the major drawback of raw memory analysis is that it lacks structure and consistency in its results, therefore making it substantially harder to locate relevant evidence from an incident response point of view having no prior knowledge of the attack scenario being played out.

Discussion

Digital forensic analysis methodology works from general to specific, one of the key elements being the identification of a period of interest, practically being the time interval during which the investigated attack took place. In the context of digital evidence and the highlight of potentially relevant traces, timestamps, and more specifically last modified or accessed date recorded on the metadata are often considered as the key element to link traces to the period of interest. To conduct a thorough analysis of an infected server, compromised by the ProxyShell exploit chain, reconstruction of the attack chain is possible by observing and investigating the server's disk and memory, as well as the network traffic during the time of the attack. Consequently, the network analysis can, in a first instance, alert of abnormal behaviour when compared to standard network traffic. Malicious IP addresses can be identified as well as TCP connections, which can narrow down the subsequent evidence search process. The compromised hard drive forensic analysis can hold traces that can be associated to each part of the attack scenario. However, the pooling of relevant artefacts found is necessary to obtain a clear perspective of the attack path followed. Memory analysis gives poor information regarding malicious process activity in the case of assessing the ProxyShell exploit chain itself but can be useful to provide a second level of artefact detection, as long as its capture is temporally close enough to the occurrence of the attack.

Table 3 summarizes, for each previously examined evidence source, which steps of the attack can be pointed out.

Source\Attack part	ProxyShell part 1	ProxyShell part 2	ProxyShell part 3	ProxyShell part 4	ProxyShell part 5	PowerShell commands	Browser web accesses
File presence on disk						2 to 7, 12	
Windows EVTX						1 to 13	
IIS logs						1 to 13	
Exchange Logs							
$USN Journal						2 to 7, 12	
$MFT						2 to 7, 12	
$LogFile						2 to 7	
Volatility analysis						13	
Raw search in RAM						1 to 13	
Network analysis						1 to 13	

Table 3: Evidence source for each step of the attack scenario

Furthermore, as to maximize potential evidence sources while performing incident response-related forensic analysis of such an infected network, some external devices, or implemented solutions on the investigated architecture, such as external log forwarding, in-detail network monitoring using deep packet inspection, possible antivirus action, detection and mitigation, and so on, could either provide more specific traces, guiding the iterative forensic process more specifically or provide additional security against anti-forensic measures.

Several aspects can be discussed regarding the evaluation of the realistic nature of the lab architecture set up, the attack scenario played out and the investigative part, which are addressed hereafter.

First, there are two prior requirements for the attack script to operate properly: one must know the mail address of a valid member of the targeted domain, as well as the mail server's IP address (or DNS name). Indeed, SMTP mail must be sent to a valid mailbox on the server, and in order to perform **ProxyShell parts 1 and 2**, the used script requires to impersonate an existing user within the Active Directory domain. Those are common parameters that would be realistic in a real-world occurrence of the attack, provided open-source investigation from the attacking side.

Then, e-mail sent as **ProxyShell part 5** is unencrypted SMTP mail sent to the standard port 25 of the targeted Exchange server, using a mock e-mail address and python configured mail server. Any mail server, to be able to receive arbitrary e-mails through the Internet, should have a listening port 25 open, thus making the mail sending realistic on this side. However, as the whole environment was set up locally, no anti-spam was configured on the lglab-mail server. In that respect, in a real-world implementation, the sent mail would be rejected because it would trigger a reverse DNS mismatch. But, to avoid configuration overhead while setting up the lab architecture, this issue was ignored as it would not prevent the attack in a real-world setting. Indeed, a prepared attacker would use a valid mail server that would not generate any error while sending the web shell via SMTP.

PowerShell command n° 13 exfiltrates the LSASS memory dump through an HTTP request sent to the web server hosted on the attacking machine lglab-cc. From a realistic point of view, one could argue that an HTTPS request would be more secure or even that HTTP port 80 would be configured as closed outbound on the server. Again, to prevent configuration overhead while setting up the lab architecture, this issue was ignored. Plus, ports 443 and 80 must be open outbound for Windows Update. It is, however, important to point out that the cleartext

HTTP request would not be available in the network analysis if the request was encrypted using SSL.

Additionally, anti-forensics techniques such as log or file deletion and more sophisticated obfuscation of the actions taking part of the attack scenario are nearly absent. Indeed, as one of the objectives of the present thesis being the reveal and evaluation of traces generated by the attacks, their presence and detection were necessary. However, further research could be conducted to try and reproduce another attack scenario more directed towards dissimulation and anti-forensics.

Furthermore, either SSRF requests to the vulnerable web server access services, Exchange Management Shell cmdlets remotely passed to the server and PowerShell commands used after the initial compromission are drawn from real-world exploit instances.

Regarding the probability of the attack occurring nowadays, according to (Shodan, 2022), still more than 22,000 servers are vulnerable to the ProxyShell attack worldwide to this day. In that respect, as Microsoft released a patch negating the vulnerability, best practice would advise to install it as soon as possible. However, applying the security patch is not always trivial, as its prior requirement is to have one of the latest Microsoft Exchange Cumulative Updates installed. Yet installing such an update is a synonym of a full Exchange reinstallation, which impacts the whole Active Directory domain architecture, as Exchange relies heavily on Active Directory. Therefore, even nearly one year after the first attacks, the ProxyShell exploit chain could still be a threat for unpatched Exchange servers.

Limits and further work

The experiences and results covered in the present thesis suggest several aspects that could deserve further study.

As the attack scenario constructed as part of the present thesis was mainly oriented over the analysis of traces generated by each step of the attack, it could be interesting, in the future, to play out diverse and more complex attack scenarii, for instance more directed towards lateral movement in the Active Directory domain, or involving actual malware execution, the initial infection vector still being the ProxyShell exploit chain. To help conduct such research, the usage of the second web shell allowing to drop binary file on the compromised server could be used.

Furthermore, on another level, existing tools used to identify indicators of compromise could be tested out on the present environment, which could also be groundwork for developing automated detection tools, as well as forensic analysis tools to parse traces related to this vulnerability chain.

As one of the initial goals of the current thesis would have been to use the deployed lab environment as a honeypot mail server in order to assess the risk level of the exploit today and to attempt to collect the most accurate traces of an actual attack on the mock mail server, the provided lab environment could be used as groundwork to conduct such research.

The world digital investigation and cybersecurity are moving quite fast. Thus, web publications on similar subjects, such as threat analysis reports or blog pages were published during the writing process of the thesis, such as Cybereason Nocturnus IR (2022), Davinsi Labs (2022) and Goddard (2021).

Conclusion

The objective of the present thesis was, in the first instance, to cover through a literature study the state-of-the-art forensic analysis and incident-response frameworks. In a second, more practical step, the ProxyShell vulnerability chain was deconstructed step by step, and after setting up a working lab environment with a vulnerable Exchange server and creating an experimental attack scenario to use the exploit, forensic analysis of the compromised Exchange server machine was carried out, with evidence highlighted at each step of the analysis and being associated with each step of the attack.

Network traffic analysis allowed to highlight and describe anomalous network activity due to the attack, disk forensic analysis of the compromised machine revealed relevant artefacts, files and log records related to the intrusion, and live memory analysis offered a second instance of some artefacts already present on the disk.

Thus, the investigation of an environment that has been compromised through the ProxyShell exploit can bring useful and relevant information to an incident-response point of view. It is possible to detect the initial infection vector and reconstruct the attack path in case of infection. However, in order to maximize the presence of evidence on the analysed supports, the forensic process, starting from the collection of relevant data sources must be as close as possible as the attack occurrence, or at least its discovery.

Several perspectives of further research could be relevant. For instance, deploying a similar lab environment as a honeypot mail server accessible from the Internet. Alternately, more complex attack scenarii could be set up, played out and forensically analysed.

In addition, the current thesis provided educational content on Active Directory domains and Microsoft Exchange, the ProxyShell vulnerability chain, and forensic investigation for incident response.

Acknowledgements

Many thanks to Professor Frank Breitinger for its supervision of the thesis and his advice, to Schenk S.A. for providing the necessary equipment and licenses for this work, and especially to Fabrizio Giannardi for his explanations, advice and expertise regarding the Active Directory and Microsoft Exchange environments. Lastly, I would like to acknowledge all those who reviewed this work.

Bibliography

Alazab M, Venkatraman S and Watters P (2009) EFFECTIVE DIGITAL FORENSIC ANALYSIS OF THE NTFS DISK IMAGE. *Applied Computing* 4(3): 551–558.

Amari K (2009) *Techniques and Tools for Recovering and Analyzing Data from Volatile Memory*.

Arntz P (2022) Patch now! Microsoft Exchange is being attacked via ProxyShell. Available at: https://blog.malwarebytes.com/exploits-and-vulnerabilities/2021/08/patch-now-microsoft-exchange-attacks-target-proxyshell-vulnerabilities/ (accessed 21 March 2022).

Carvey HA (2014) *Windows Forensic Analysis Toolkit: Advanced Analysis Techniques for Windows 8*. Fourth edition. Amsterdam: Syngress.

Centre national pour la cybersécurité NCSC (2021) Nouvelles failles de sécurité sur les serveurs Microsoft Exchange. Available at: https://www.ncsc.admin.ch/ncsc/fr/home/aktuell/im-fokus/exchange-server-2021-8.html (accessed 21 March 2022).

CERT-FR (2022) Multiples vulnérabilités dans Microsoft Exchange. Available at: https://www.cert.ssi.gouv.fr/alerte/CERTFR-2021-ALE-017/ (accessed 21 March 2022).

Cybereason Nocturnus IR (2022) Threat Analysis Report: DatopLoader Exploits ProxyShell to Deliver QBOT and Cobalt Strike. Available at: https://www.cybereason.com/blog/research/threat-analysis-report-datoploader-exploits-proxyshell-to-deliver-qbot-and-cobalt-strike (accessed 14 June 2022).

Davinsi Labs (2022) Threat Advisory: How to respond to ProxyShell, the latest exploit against Exchange. Available at: https://davinsi.com/news/threat-advisory-how-respond-proxyshell-latest-exploit-against-exchange (accessed 14 June 2022).

Gallagher S and Mackenzie P (2021) Conti affiliates use ProxyShell Exchange exploit in ransomware attacks. In: *Sophos News*. Available at: https://news.sophos.com/en-us/2021/09/03/conti-affiliates-use-proxyshell-exchange-exploit-in-ransomware-attacks/ (accessed 13 May 2022).

Goddard J (2021) ProxyNoShell: A Change in Tactics Exploiting ProxyShell Vulnerabilities. Available at: https://www.mandiant.com/resources/change-tactics-proxyshell-vulnerabilities (accessed 14 June 2022).

Hammond J (2021) Microsoft Exchange Servers Still Vulnerable to ProxyShell Exploit. Available at: https://www.huntress.com/blog/rapid-response-microsoft-exchange-servers-still-vulnerable-to-proxyshell-exploit (accessed 16 March 2022).

Hausknecht K, Foit D and Burić J (2015) RAM data significance in digital forensics. In: *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2015, pp. 1372–1375. DOI: 10.1109/MIPRO.2015.7160488.

Hutchins EM, Cloppert MJ and Amin RM (2011) *Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains*.

Iddon G (2021) ProxyShell vulnerabilities in Microsoft Exchange: What to do. In: *Sophos News*. Available at: https://news.sophos.com/en-us/2021/08/23/proxyshell-vulnerabilities-in-microsoft-exchange-what-to-do/ (accessed 16 March 2022).

JoanneHendrickson (2022) Exchange Server architecture. Available at: https://docs.microsoft.com/en-us/exchange/architecture/architecture (accessed 24 February 2022).

Kent K, Chevalier S, Grance T, et al. (2006) *Guide to integrating forensic techniques into incident response*. NIST SP 800-86. Gaithersburg, MD: National Institute of Standards and Technology. DOI: 10.6028/NIST.SP.800-86.

KrebsOnSecurity (2021) At Least 30,000 U.S. Organizations Newly Hacked Via Holes in Microsoft's Email Software. Available at: https://krebsonsecurity.com/2021/03/at-least-30000-u-s-organizations-newly-hacked-via-holes-in-microsofts-email-software/ (accessed 21 March 2022).

Lee K, Lee C and Lee S (2013) On-site investigation methodology for incident response in Windows environments. *Computers & Mathematics with Applications* 65(9): 1413–1420. DOI: 10.1016/j.camwa.2012.01.029.

Microsoft (2021a) CVE-2021-31207 - Guide des mises à jour de sécurité - Microsoft - Vulnérabilité d'exécution de code à distance dans Microsoft Exchange Server. Available at: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-31207 (accessed 21 March 2022).

Microsoft (2021b) CVE-2021-34473 - Guide des mises à jour de sécurité - Microsoft - Vulnérabilité d'exécution de code à distance dans Microsoft Exchange Server. Available at: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-34473 (accessed 21 March 2022).

Microsoft (2021c) CVE-2021-34523 - Guide des mises à jour de sécurité - Microsoft - Vulnérabilité d'élévation de privilèges dans Microsoft Exchange Server. Available at: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-34523 (accessed 21 March 2022).

MSRC Team (2021) On-Premises Exchange Server Vulnerabilities Resource Center. Available at: https://msrc-blog.microsoft.com/2021/03/02/multiple-security-updates-released-for-exchange-server/ (accessed 21 March 2022).

Nedjimi B and Thobois L (2016) *Exchange Server 2016 : configurez et gérez votre environnement de messagerie*. Ressources informatiques. Saint-Herblain: Éditions ENI.

Orange Tsai (2021a) From Pwn2Own 2021: A New Attack Surface on Microsoft Exchange - ProxyShell! In: *Zero Day Initiative*. Available at: https://www.thezdi.com/blog/2021/8/17/from-pwn2own-2021-a-new-attack-surface-on-microsoft-exchange-proxyshell (accessed 16 March 2022).

Orange Tsai (2021b) Orange: A New Attack Surface on MS Exchange Part 1 - ProxyLogon! Available at: https://blog.orange.tw/2021/08/proxylogon-a-new-attack-surface-on-ms-exchange-part-1.html (accessed 24 February 2022).

OWASP (2022) Server Side Request Forgery Software Attack. Available at: https://owasp.org/www-community/attacks/Server_Side_Request_Forgery (accessed 16 March 2022).

OWASP Top 10:2021 (2021). Available at: https://owasp.org/Top10/ (accessed 21 March 2022).

Pilli ES, Joshi RC and Niyogi R (2010) Network forensic frameworks: Survey and research challenges. *Digital Investigation* 7(1–2): 14–27. DOI: 10.1016/j.diin.2010.02.003.

Qureshi S, Tunio S, Akhtar F, et al. (2021) Network Forensics: A Comprehensive Review of Tools and Techniques. *International Journal of Advanced Computer Science and Applications* 12(5). DOI: 10.14569/IJACSA.2021.01205103.

Shodan (2022) Shodan Report vuln: cve-2021-34473. Available at: https://www.shodan.io/search/report?query=vuln%3Acve-2021-34473 (accessed 15 June 2022).

Shorebreak Security (2022) SSRF's up! Real World Server-Side Request Forgery (SSRF). Available at: https://www.shorebreaksecurity.com/blog/ssrfs-up-real-world-server-side-request-forgery-ssrf/ (accessed 16 March 2022).

Unit42 (2021) Remediation Steps for the Microsoft Exchange Server Vulnerabilities. Available at: https://unit42.paloaltonetworks.com/remediation-steps-for-the-microsoft-exchange-server-vulnerabilities/ (accessed 21 March 2022).

Yaacoub J-PA, Noura HN, Salman O, et al. (2021) Digital Forensics vs. Anti-Digital Forensics: Techniques, Limitations and Recommendations. arXiv:2103.17028. American University of Beirut. Available at: http://arxiv.org/abs/2103.17028 (accessed 10 June 2022).

Appendices

As most of the appendices linked to this work are quite verbose, or their format is not adapted to be shown on a text document, they were uploaded on a public GitHub, accessible at the following URL: GitHub link. The permalink to each of the previously cited resource is displayed hereafter.

1. The exploit PoC script used to conduct the attack: exploit.py.
2. The python web server used to exfiltrate data: python_web_server_auth.py.
3. PowerShell commands passed to the server, briefly described: PowerShell commands.
4. SMTP stream from the Wireshark capture: SMTP stream.
5. HTTP stream from the Wireshark capture: HTTP stream.
6. Relevant content of IIS log file IIS_Log_Content.xlsx.
7. Relevant content of the Rps_Http_2022052312-1.LOG Exchange log file.
8. Relevant content of the Rps_Cmdlet_2022052312-1.LOG Exchange log file.
9. Relevant content of the audit20220523-2.LOG Exchange log file.
10. Relevant content of the audit20220523-3.LOG Exchange log file.
11. Relevant content of the HttpProxy_2022052312-1.LOG Exchange log file.
12. Relevant content of the Autod_2022052312-1.LOG Exchange log file.
13. Relevant content of the SEND2022052312-1.LOG Exchange log file.
14. Parsed content of the $USN Journal system file.
15. Highlighted $MFT entry for test.txt.
16. Each $LogFile entry for test.txt.