universität wien

BACHELORARBEIT / BACHELOR'S THESIS

Titel der Bachelorarbeit / Title of the Bachelor's Thesis

„File Format Security - Hiding Executable Code in Data Files“

verfasst von / submitted by

Samuel Šulovský

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Bachelor of Science (BSc)

Wien, 2022 / Vienna, 2022

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 033521
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Bachelorstudium Informatik
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Ing. Mag. Dr.techn. Edgar Weippl

Acknowledgements

Thank you!

Abstract

This LaTeX template provides example on how to format and display text, mathematical formulas, and insert tables or images. There is a lot more you can do with LaTeX, for more information check out https://en.wikibooks.org/wiki/LaTeX. I'm sure changes work just fine.

Kurzfassung

Das ist eine deutsche Kurzfassung meiner in Englisch verfassten Masterarbeit.

Contents

List of Tables

List of Figures

List of Algorithms

Listings

1. Motivation

Ever since my first foray into the field of information security I have been fascinated by the ways in which different threats to an organisation's security arise. From the ways in which data can be obtained by rummaging through dumpsters where sensitive documents were dumped without being properly destroyed to sophisticated zero-day exploits used to distribute malware, the topic that particularly caught my interest was the way in which malicious payloads can be concealed in relatively mundane looking files.

A perfect example of such file was a malicious Microsoft Word document created by the Lazarus Group Advanced Persistent Threat distributed to victims in South Korea via spear phishing. The malware itself was hidden within a Bitmap Image (BMP) file that was itself concealed as a Portable Network Graphics (PNG) file. This malicious file was extracted to the victim's computer by a macro in the macro-enabled Word document, which is also a very interesting part of the infection process.

This document piqued my interest for a multitude of reasons, chief among which was the interesting mechanism used to infect the victim's device, which used a quirk of the BMP file format and a conversion function built into the Visual Basic for Application (VBA) programming language used to write macros that can be embedded in Word Documents. Using this functionality allowed the malicious payload to be concealed under many layers of file formats while also keeping the extraction process quite simple, almost routine. The attack itself was also rather creative, using an interesting attack vector and custom toolchain typical for this threat actor.

The problem with attacks like this is that since files are all simply a series of bytes in the end, with the interpretation being governed largely by how the operating system or program interacting with the file interprets those underlying bytes. Due to how file formats are defined, some formats are more suitable for attacks than others and with the multitude of formats supported and used over the decades of computing, it is only inevitable that there would be a way to misuse some format in some way – one of which being BMP in this case. I believe it is valuable to inspect this attack to shed light on how potential future attacks could be carried out.

Out of interest as well as scientific rigour I will attempt to recreate this malware, foregoing the malicious payload of course, and analyse its effectiveness. The main questions I will seek to answer are the following:

- In what ways can a file hide malicious content?
- How can a concealed malicious payload be extracted from a file and executed?
- How do the previous questions come together to drop the RAT in the analysed document?

- Can the analysed document be recreated? Does the exploit still work?
- Does the analysed document avoid detection by antivirus software?
- Are common systems still vulnerable?

Thus, the primary goal of this work will be to recreate the malicious Microsoft Word document along with the BMP payload and secondary mocked Windows Executable (EXE) payload without the actual Remote Access Trojan. Recreating this malware should help verify the reproducibility of the original postmortem of the attack and its functionality as well as help gain further insight into how vulnerable current systems are to a similar attack, if at all. Furthermore, this analysis may yield advice other than the simple adage of not opening macro-enabled documents. Though, of course, this is always the best protection mechanism against malware using VBA macros as their attack vector.

To achieve this goal we will create a facsimile of the malicious document as well as the payload it carried. This recreation will be based on the postmortem report of the attack written by Hossein Jazi. The first part of the recreation will be a dummy Windows Executable containing a simple program that indicates the system would have been compromised if the attack was real. This EXE will then be hidden in the BMP the same way as in the original attack and embedded in the macro-enabled Word document, analogous to the attack. Finally, we will be executing this faux-malicious document inside virtual machines running the Windows operating system and tracking how it executes in comparison to the original attack.

The metrics for measuring the success of this experiment will be rather simple – recreating the attack in its full scale will be a full success, while failure after at least one part of the attack succeeds will be deemed a partial success. If the recreated attack is successful, we will further test its functionality on a range of virtual machines running the currently supported versions of the Windows operating system to see if the attack can be reproduced on all, or only some versions. We will also keep an eye out on when or whether antivirus software detects the payload.

In summary, recreating this attack can lend insight into how file formats can be misused to carry malicious payloads and avoid detection while doing so. It also serves as an effort to validate the previous research done of this malware and make sure the results of that research are reproducible. Furthermore, the alternative setup using a facsimile of the malicious file may provide additional insight that had previously gone unnoticed.

2. Related Work

2.1. What Is Malware?

Though defining malware might seem as a simple task, formally defining it has been a difficult open problem in computer virology for a long time; the precise reasoning for this stems from the fact that each algorithm or piece of software can be expressed logically and has certain *intended behaviour*, which often isn't properly defined [KB10]. Kramer and Bradfield posit a logical definition of malware which we won't fully dedicate ourselves to, but their introduction to the concept without the use of logical language is worth mentioning. They posit malware as software that causes the actual behaviour of some other software to differ from its intended behaviour, where this difference stems from an incorrectness in verification or validation of program behaviour, leading to a defining characteristic of malware being the *causation of incorrectness* [KB10].

Moving from this more formal definition to a more informal one, Skoudis defines malware as "[...]a set of instructions that run on your computer and make your system do something that an attacker wants it to do [SZ03]". For our intents and purposes this definition is sufficient, and we will further broaden it to our working definition:

> Malware is software maliciously designed to do whatever its author wants, unconstrained by legality, consent or permission.

2.1.1. Motivation

Malicious actors can act in a multitude of ways, with motivations behind their actions grouped into a few overarching categories. While different classifications exist, we will be basing ours on a classification by Brewster et al. While there are many reasons for malware authors to act maliciously, the creation of malware doesn't always have to be malicious. Malware can also be created to showcase a vulnerability and call attention to it, so that the security of the system under attack can be improved without causing any actual damage. These kinds of actors are called *white hat hackers* [Cal11].

Ideological

Attackers motivated by ideology fall into this category. In the taxonomy by Brewster et al. this encompasses the *political, ideological and informational / promotional* categories, wherein the actors act based on a political agenda (such as espionage, sabotage or political protest), a held belief (such as the belief in freedom of information) that views hacking into systems as a necessary act, or the desire to disseminate information and increase public awareness [ASA+15].

A famous example of a political attack is the Stuxnet worm that targeted Iranian nuclear facilities in the year 2010 [ASA+15]. Stuxnet is reported to have been perpetrated by the US and Israeli governments, though unconfirmed. An interesting facet of the Stuxnet worm was the fact that it spread through systems without causing any damage until it arrived at its designated targets, where it activated to sabotage the target systems.

While ideological actors in the taxonomy by Brewster et al. are similar to political attackers, they can be distinguished because the beliefs they hold are personal, such as a protest against something they oppose or their religion [ASA+15]. Informational / promotional actors are, in our opinion, very similar to these kinds of actors, with a famous example being Edward Snowden. Snowden is wanted by US authorities on charges of espionage after stealing and subsequently leaking thousands of documents pertaining to government espionage against its own citizens [sno].

Commercial

Attackers that pursue some sort of financial, commercial or economic gain fall into this category. Brewster et al. distinguish between *financially motivated* actors and *commercially motivated* actors, with the main distinction being that the financially motivated actors act to gain more directly, while the commercial actors might act out of additional reasons such as economic or industrial espionage or theft of company secrets or intellectual property [ASA+15]. We believe these motivations to be sufficiently similar to allow them to be grouped under one umbrella term.

Personal

The final category we observe joins together the remaining categories in the studied taxonomy, encompassing motivations that are directly related to a person's own life. These are distinct from the ideologically motivated actors, as their actions aren't necessarily driven by ideology, but more so by emotion or way of life. This category encompasses actors that act emotionally, such as out of anger, boredom or who seek revenge, actors that hack because they find it fun or challenging, or want validation from peers, or even actors that resort to hacking due to how they choose to live their life, such as trolls hacking as a means of causing emotional distress to their targets [VHIB12, ASA+15].

2.1.2. Types of Malware

Malware comes in many shapes in sizes that have some characteristics in common, while differing on others. The most basic part that all malware has in common is the *payload*, or what the malware is supposed to do [Ayc06, p. 12]. This could be anything, but it is often understood to mean the malicious activity that the malware performs. Another property we consider all malware to have in common is an *attack vector* or how the malware gains access to the victim's system. We will cover attack vectors separately, but some examples include social engineering, phishing, drive-by attacks, droppers or abuse of a vulnerability.

Where malware begins to differ are the other properties – Aycock posits a taxonomy based on the following three characteristics, with each type of malware being classified on a scale roughly akin to "yes, no, maybe" for each property.

> 1. *Self-replicating* malware actively attempts to propagate by creating new copies, or instances, of itself. Malware may also be propagated passively, by a user copying it accidentally, for example, but this isn't self-replication.
> 2. The *population growth* of malware describes the overall change in the number of malware instances due to self-replication. Malware that doesn't self-replicate will always have a zero population growth, but malware with a zero population growth may self-replicate.
> 3. *Parasitic* malware requires some other executable code in order to exist. "Executable" in this context should be taken very broadly to include anything that can be executed, such as boot block code on a disk, binary code in applications, and interpreted code. It also includes source code, like application scripting languages, and code that may require compilation before being executed.
>
> [Ayc06, p. 11-12]

Where these three characteristics are not sufficient to differentiate two types of malware, additional clarification can be provided to distinguish the two; for example while *spyware* and *adware* both are not self-replicating, have no population growth and are not parasitic, their payloads differ – where spyware collects information for exfiltration, adware often uses collected information for advertising purposes, spamming the user with advertisements or exfiltrating information to gain a competitive edge [Ayc06, p. 16-17].

Viruses

Throughout our research, we have repeatedly come across a definition by Cohen, one of the first people to conduct research into computer viruses, which goes as follows: "A virus is a program that can 'infect' other programs by modifying them to include a, possibly evolved, version of itself [Coh94]." The term *infect* is understood to mean the modification of the target program to include a copy of the virus as explained in the remainder of the definition.

Describing viruses using the three metrics outlined by Aycock, we consider viruses to self-replicate, have a positive population growth and be parasitic [Ayc06, p. 14]. Though viruses spread across the infected system (leading to the positive population growth), they notably don't spread through networks, which is the domain of worms, covered in the following section [Ayc06, p. 15].

Viruses are generally noted to be structured in three distinct parts:

- the *infection vector* – how the virus spreads across a system. It doesn't necessarily have to be unique, leading to *multipartite* viruses,

- the *trigger*, which is how the virus decides when the payload should be delivered,
- and finally the *payload* which is what the virus does, other than just replicate throughout the infected system. Usually, the payload intends to cause some sort of damage, or even act as the start of another attack, such as the virus payload intending to open a back door to the system, allowing the attacker to access the system and use it as part of a botnet.

[Har01, Ayc06, p. 7, p. 27]

Viruses, as described, infect files in order to spread and eventually drop their payload. There are numerous ways this infection can take place: The file itself can be modified by the virus to contain a copy of the virus in a process known as overwriting or insertion, the file can be replaced by a copy of the virus file that redirects the user to the correct file after doing whatever it pleases after the user attempts to open the file (think of it as a more malicious shortcut) in what is called a companion virus, or, most importantly for the topic of this work, the virus can be embedded in the macro section of a document format that supports macros, for example Microsoft Word [Ayc06, SZ03].

Macros are a simple tool that was originally intended to increase the productivity of users using word processors such as Microsoft Word. Using macros allows the user to run arbitrary code at will, for example when the document is opened, which is easy to weaponise for malicious use. Though the user is often given a warning about the presence of macros in a document and given the option to run them, these warnings can be easily ignored, or worse, the user can be deceived into allowing macros to be executed by an attacker.

The payload of a virus can be arbitrary, from having no payload at all and simply infecting more and more files, to payloads that randomly delete files, clog up memory, create logic bombs or even back doors to the infected system [Har01, SZ03, Coh94].

Worms

Since viruses spread through files on a system, they are naturally limited in their spread by human interaction [Ayc06]. In the early days of computing and computer viruses, it was common for users to move floppy disks between systems, which allowed an easier spread for the virus, since there was a high rate of mobility of physical disks between computers [Coh94].

Spreading through networks instead of just the local file system is a rather important trait, which is why malware capable of spreading throughout networks was given a new name: *worm*. Worms share many characteristics with viruses – on Aycock's taxonomy they share two properties: they both self-replicate and they both have a positive population growth, but differ in parasiticalness; worms forego being parasitic [Ayc06, p. 15]. Worms are standalone entities and don't infect other files or rely on other executables; they can be thought of as infecting the machine itself [Ayc06, Har01].

The main draw to worms from a malware author's perspective is the fact that worms often spread at a very rapid pace, since they don't have to rely on humans in order

to propagate [Ayc06, SZ03, Har01]. Though their speed is one draw, their reach is a non-negotiable second – since they aren't constrained to being passed around through physical media, they can reach virtually any device in the world, especially in today's interconnected society [SZ03, Ayc06]. This allows attacks to scale well and quickly infect a large amount of systems, which can, depending on the payload, have potentially devastating consequences.

Another upside is that attacks conducted with worms have the potential to be very difficult to attribute to a particular attacker, since the worm rapidly spreads throughout the internet, one victim's computer may be infected by the worm from an IP address located in Ghana, while the next victim might be infected from a Swiss IP, making attribution, persecution as well as countermeasure development more difficult [SZ03].

Worms are structured similarly to viruses, with a few minor differences. Skoudis proposes a model shaped like a missile, wherein a worm is described as having a warhead which serves to penetrate the attacked system, a propagation engine, scanning engine and target selection algorithm to facilitate the propagation of the worm to appropriately chosen and vulnerable victims, and finally a payload to be dropped on infected devices to compromise the system or otherwise cause damage to the victim [SZ03].

There are various ways in which a worm can propagate across a network, often being given a name based on the spread mechanism, e.g. *email worm*. Because worms are standalone programs and don't rely on any particular files they can spread through networks through ways that viruses don't, for example by using buffer overflows or underruns to leech onto open network connections or can even send malicious emails or messages using other messaging clients with infected attachments to a victim's contacts [SZ03, Ayc06].

A particularly destructive use of worms is the creation of *botnets*, large distributed networks of computers that an attacker has established a back door in and compromised to do their bidding. These botnets can then be weaponised to launch Distributed Denial of Service (DDoS) attacks with overwhelming force [SZ03]. Worms are the perfect tool to create these compromised networks as they spread rapidly across the world and can quickly find new victims and vulnerable machines while making the resulting attack harder to trace and more powerful [SZ03].

Trojan Horses

The name of this malware threat comes from the epic poem *Aeneid* by Virgil, which describes the final siege of the war between the Greeks and the Trojans. The Greeks built a large wooden horse and hid a small part of their army inside it, then pretended to leave. The Trojans thought the horse was left as a gift and dragged it into the city. Unbeknownst to them, the horse was full of enemy soldiers that sneaked out of the horse under the cover of the night and opened the city gates from the inside for the attacking Greek army, ending the war then and there.

Similarly to the horse the Greeks constructed in the Trojan war, *trojan horses* are malicious programs that claim to be executing some mundane, harmless task (and they may or may not actually be doing that), while secretly executing some malicious task in

the background, such as keylogging, or establishing a back door to the victim's system [Har01, Ayc06, p. 12-13]. What differentiates them from viruses and worms is that they don't self-replicate and thus don't have a population growth, but can be thought of as parasitic, as they perform tasks other programs might, while being malicious in the background [Ayc06, p. 12].

In the modern world, the lines between the three blur, however. Crucially, *multipartite* malware relies on the user launching an application they think will perform some mundane action (as a trojan would) in order to install a self-replicating worm or virus and trigger the mechanism used to pass it on further [Har01]. This is just one of many complexities in categorising malware, which we will gloss over for the sake of brevity.

Just like for any other types of malware, a payload is important for the trojan to do any real damage. An important type of payload often carried by trojans is a backdoor, leading to a so-called Remote Access Tool (RAT), also called Remote Access Trojan [Har01]. RATs allow an attacker (or user of the RAT) to submit commands to the victim (target) computer, copy files to and from it, making the machine a tool at the attacker's disposal [Har01]. The language of the previous sentence is purposefully vague on whether an attacker is acting maliciously, as remote control (or remote access) software plays a legitimate role in computing – we often want to access a device remotely to work or copy files, such as by using `ssh`.

RATs are a particularly important threat when misused, as they grant an attacker full control over the victim's computer, turning the victim's device into a *zombie*, potentially being able to use it to fuel DDoS attacks, as described in the Worms subsection.

Ransomware

Ransomware will be discussed in depth in a further section discussing the current threat landscape, as it is an integral part of it.

2.1.3. Concealing Malware

2.2. Social Engineering and Phishing

One of the most common infiltration vectors used by malicious actors is social engineering, where the victim is led to perform certain actions, divulge information or grant access to a system based on psychological manipulation [KHHW15, HA21]. Social engineering continues to be among the top threats to companies in 2021, preying on the human factor and momentary lapses in judgement by individual workers to attack even the most secure of systems [WPH+21]. The efficiency of these attacks is not hampered by the technical security of a system, since the people using the system are the weakest link in its security [KHHW15, MLV16].

Though social engineering encompasses a multitude of different types of attacks ranging from physical to computer-assisted, we will be focusing on the computer-assisted side of social engineering, namely phishing. Phishing is a sociotechnical social engineering attack usually perpetrated through email or instant messaging against a large amount of

targets in the hopes of the attack being successful against enough targets to be profitable [KHHW15]. Phishing carries an analogy to fishing – messages sent to potential victims are analogous to the line of a fishing rod, with the message contents serving as the bait. The recipient of the phishing message is led to believe they must take an action outlined in the message, usually clicking a link or downloading an attachment, which will lead to the malicious actor stealing personal information from the victim or infecting the device with malware [Hon12]. Forging convincing looking emails, websites or business documents is much easier than perpetrating a similar scheme in real life by, say, opening a fake brick-and-mortar business.

> We've evolved social and psychological tools over millions of years to help us deal with deception in face-to-face contexts, but these are little use to us when we're presented with an email that asks us to do something. [And08]

While we don't concern ourselves with phishing directly, it is the most common attack vector used to deliver infected files like the one we study in this thesis.

2.3. Current Threat Landscape

An overview of the current threat landscape is crucial to help us understand the role our analysed attack plays in the overall landscape. Most notably, since infected document files continue to play a major role in malware attacks, it's important to understand the currently rising risks that such files might pose for us, the leading among them being ransomware at this time [LTT+21a].

Cybersecurity threats have been on a continuing rise in recent years, both in size and scale [LTT+21a]. Remote work during the COVID-19 pandemic greatly increased the attack surface for malicious actors, as well as the time needed to detect intrusions or system compromise, leading to the average cost of data breaches increasing across the board [Seca]. The role of remote work in security related incidents was further highlighted as the pandemic moved into its later stages in 2021 as more companies returned to in-office work. In its 2021 "Cost of a Data Breach" report, IBM reported that security incidents were costlier for companies implementing remote work, with companies where over 50% of the workforce worked remotely taking, on average, 58 days more to identify and contain breaches when compared to companies where less than 50% of workers worked remotely [Secb]. This is in line with in the eponymous report from 2020, where 70% of questioned organisations that implemented remote work as a result of the COVID-19 pandemic expecting the average cost of data breaches to increase [Seca]. The rise in the cost of data breaches continued in the year 2021, rising by a further 10% to an average cost of $4.24 million [Secb].

2.3.1. Supply Chain Attacks

The biggest rising trend of 2021 have been *supply chain attacks*, driven by the rising reliance of companies on third party solutions for their IT needs [LTT+21a, Mor]. The

significance of these attacks has been so high, that the European Union Agency for Cybersecurity (ENISA) created a separate threat landscape report specifically for this threat. Supply chain attacks are fundamentally comprised of two distinct attacks: the first on a supplier which the attackers then use to conduct a second attack on a target which uses the services of the compromised supplier, be it a customer or another supplier [LTT+21b].

While the attack vectors used to conduct the initial attack on the supplier are in line with expectations, consisting of for example social engineering, brute force attacks or abuse of vulnerabilities in software or configurations used by the victim, the attack vector used to conduct the secondary attack is far more dangerous and highlights the true danger and severity of supply chain attacks. Because of the customer-supplier relationship between the two victims, there exists an inherent trust between the two parties that these types of attacks abuse. The danger comes through so-called Trusted Relationship attacks where a relationship of trust between two parties is compromised by an attacker and used to compromise the victim's security by using the trusted relationship as a means of lending themselves legitimacy.

In the realms of software this can mean an attacker compromising a supplier's code repositories, infecting it with malware and then distributing it as an update to the supplier's customers. The customer is lulled into a false sense of security with the update, as it comes from the supplier, and it appears there is no cause for concern. Attacking the supplier and gaining access to their systems can lead to various kinds of abuse such as phishing, distributing malware, or impersonating the supplier's personnel [LTT+21b].

The attack targets vary slightly between the two attacks, with the supplier attack being perpetrated not just in order to conduct the second attack, but also for example to steal code, configurations or even hardware schematics. By far the most common target for supplier assets is code, with two thirds of the attacks studied by ENISA between January 2020 and July 2021 aiming to compromise this asset [LTT+21b].

The attack on the customer carries similar traits with other cyberattacks, the main difference is in the increased ease of infiltration dependent on the success of the first attack. Common targets are for example exfiltrating data, establishing botnets to carry out Distributed Denial of Service (DDoS) attacks or extorting money from the victim via ransomware. Data exfiltration was the most common between all the attacks covered in the ENISA supply chain attack threat landscape with 58% of attacks targeting this asset [LTT+21b].

2.3.2. Ransomware

The term ransomware describes a category of malware threats used to digitally extort victims by denying access to a device or files, unless a ransom sum is paid, usually in Bitcoin or other cryptocurrencies [**?**, p. 150]. Some ransomware attacks, like WannaCry, additionally threaten to delete the victim's files unless the ransom is paid within a certain time limit to further intimidate victims into paying the ransom. The methods used to infect systems with ransomware are in line with other common cyberattacks, for example making use of social engineering or exploiting vulnerabilities [BFSS].

A worrying trend reported by ENISA was the increase in use of zero-day vulnerabilities in the perpetration of ransomware schemes [LTT+21a]. *Zero-day attacks* are a class of attacks that exploit vulnerabilities that have not yet been publicly disclosed [BD12]. They give the attacker a massive advantage as it's virtually impossible to defend against them because of their nature – antivirus software has no hash signature to recognise and the software's developers have no chance to patch the exploit since they aren't aware of it. These kinds of attacks were previously used mainly by Advanced Persistent Threats (APTs) and nation-state threat actors, mainly due to their immense value as a free pass to any target they may wish to attack [LTT+21a]. The fact that zero-day attacks are becoming more common in the ransomware sphere means that the high cost of using a zero-day vulnerability is worth it for the attackers – one is led to believe the payouts are high enough to justify it.

The profitability of ransomware is a large reason for its rise. It's a means of maximising the monetisation of malware as attackers become increasingly motivated by financial gains [TP17, LTT+21a]. Some say we are observing a "golden age" of ransomware, as ransomware becomes more available to threat actors as Ransomware as a Service (RaaS) becomes more and more widespread and attackers target larger targets in search of higher payouts [LTT+21a]. Additionally, amoral threat actors are increasingly targeting vital infrastructure and organisations that rely on access to their data to prevent death of patients or hold very important legal data, asking for exorbitant ransom fees which they hope the victims will pay in order not to be complicit in ,for example, the death of a patient [Mor, LG16, p. 17-18]. High profile hacks and payouts continue to motivate these threat actors to try and replicate these successes and get a large payout.

Another increasing trend in ransomware is the utilisation of multiple axes of attack. The result of these multiple threats has become known as *double extortion ransomware* or even *triple extortion ransomware*. A common double extortion ransomware attack consists of the encryption of the victim's data alongside its exfiltration, with the attacker demanding ransom be paid or their files would not only remain decrypted, but would also be leaked [LTT+21a, 42].

As mentioned above, ransomware is distributed much the same as any other malware threat, which is why the developments in this area are relevant to our topic. Hiding a ransomware payload in a data file is virtually the same as hiding any other malware in the file. Thus, with ransomware on the rise, we can anticipate payloads of infected Microsoft Word or PDF documents to deliver ransomware instead of other malware types. In fact, this is already the case – malicious macro-enabled Word documents or exploited PDF files are already among the attack vectors used in ransomware delivery, often as a form of downloading, de-obfuscating or decrypting the code that takes control of the machine [LG16, p. 8-10].

Ransomware as a Service (RaaS)

As was mentioned in subsection 2.3.1, companies are increasingly turning to external service providers for their IT solutions and the same can be said for threat actors. Though distributing malware with the intent of letting other actors perpetrate attacks is not

a new phenomenon, the Ransomware as a Service (RaaS) business model is surging in popularity in recent years. Conti and REvil, the two threat actors with the largest profits as well as infections in the ransomware space, were both RaaS providers, allowing their customers to easily orchestrate ransomware attacks [LTT+21a, 42].

RaaS also makes it much easier for inexperienced attackers or script kiddies to get access to ransomware and carry out attacks – so long as they have the capital to purchase the ransomware from the dark web [GSBTLR+17]. Additionally, this points to malicious actors growing closer together as a community, allowing each to focus on their own area of expertise (malware creation, social engineering, vulnerability discovery, etc.) and sell expertly crafted solutions to other malicious actors for money or even a share of the profits of an attack carried out using their tool [LTT+21a]. Another shortcut attackers take that has been observed in 2022 by Unit 42 is the purchasing of access to a compromised network directly from so-called *initial access brokers* who provide the ransomware attackers with the ability to directly drop their ransomware into an already compromised network, saving time and money [42]. It is easy to see why this kind of relationship is particularly profitable for attackers perpetrating ransomware attacks.

Furthermore, RaaS comes with an online dashboard for the command and control server, bulk mail spamming services, or even specialised social engineering teams that further amplify the effectiveness of the attack [GSBTLR+17]. These services, naturally, come at a cost. Some compensation methods for the RaaS creators include a subscription based model, direct payment or a cut of profits [LTT+21a, LG16, p. 44-45].

Attributing ransomware attacks to specific threat actors is also becoming more difficult due to this model, as when a certain strain of ransomware is known to, as an example, spread via e-mail in North America, if that same ransomware then starts spreading via malicious banner ads in Africa, linking the two is difficult for security researchers, as it is unclear if they belong to the same family of ransomware, or if they are entirely different [LG16, p. 47]. This hampers not only research into ransomware and how it spreads, but in some cases also recovery efforts. Knowing which strain of ransomware infected a network can in some cases even help avoid paying ransom altogether if the encryption algorithm used is known to be crackable.

All these factors have contributed to the rise in ransomware cases, ransom amounts as well as groups perpetrating these attacks due to RaaS lowering the barrier to entry [42]. Because of this, we can expect the amount of ransomware groups to continue to rise in the future. Since ransomware is an increasingly popular attack type, the ability to hide it in infected data files is crucial, or in other words: being able to understand how ransomware can be hidden within data files is becoming increasingly important.

Multiple Extortion Ransomware

Another trend mentioned in passing in subsection 2.3.1 is the increase in cases where threat actors choose to pressure the victim on multiple fronts, often two, with three becoming an important number as well. Double extortion was first observed in late 2019, but is noted to have exploded in popularity by 2022 with the primary secondary extortion tactic being data exfiltration with the motive of threatening victims into paying quicker,

or demanding additional funds [42, PM21].

This data exfiltration trend arose out of a decreased number of organisations willing to pay ransom due to strong backups – due to easy backup solutions like cloud backups more and more companies were able to recover from serious ransomware attacks by simply resetting their machine to a clean state and restoring a backup, hampering the attackers' ability to secure the ransom payment [PM21]. Attackers often threaten to publish the data on the dark web to name and shame the victims into paying the ransom, because even if the organisation can recover its data using a back up, the threat of a data leak is much more difficult to stop internally.

An example of a threat actor that perpetrates these kinds of attacks is the Conti threat group. The group has been at the forefront of ransomware attacks observed by Unit 42 and has leveraged double extortion attacks to demand high average ransom payments of $1.78 million in 2021 and leaking data of over 600 organisations, including vital infrastructure such as hospitals or law enforcement agencies [42]. It is clear that Conti are ruthless attackers, stopping at nothing to secure profits.

Even though double extortion ransomware is still in its infancy, attackers are already innovating on the concept of using multiple angles of extorting their victims. The term *triple extortion ransomware* was coined mainly in relation to a ransomware attack on Finnish mental health care provider Vastaamo. The attackers targeted the clinic and demanded ransom for decryption of patient data, which they also exfiltrated as part of a double extortion scheme [KV22]. However, what set apart this case was that the attackers then additionally extorted the patients, the victim's customers, by demanding smaller sums of money from them directly, threatening to leak their therapists' session notes [Whi21]. Ultimately, the company buckled under the heavy financial losses and the data breach and declared bankruptcy, ceasing operations completely [Whi21, KV22].

Attacks like the ones described above have been given the name *multiple extortion ransomware* by Payne and Mienie to ransomware that allows the attacker multiple opportunities for extorting payments from their victims. [PM21]. Payne and Mienie further suggest that the evolution of multiple extortion attacks is an important and ongoing development in the current threat landscape:

> There are many ways in which these kinds of multiple-extortion attacks may already be developing organically, with or without forethought. First, third parties who purchase or download a victim's sensitive files may pursue additional attempts to blackmail or otherwise extort money from the victim directly using the same information. Second, criminals may leverage customer, client, patient or employee data from leaked files to harass, intimidate, or extort money from those individuals. Third, and possibly most concerning of all, while an individual or organization might not be a valuable or high-profile target at the present time, the negligible cost of storing stolen data could mean that many years in the future, the sensitive information of a potential world leader or prominent businessperson could be used to blackmail or extort years or possibly decades after an initial attack. [PM21]

We've already been able to see some of their thoughts become reality during the Vastaamo

attack, where the second development cited can be observed with attackers extorting the victim's customers.

2.3.3. State-sponsored Threat Actors

2.4. File Format Security

Files are a fundamental building block of computing, primarily used to store data and index it by a file name. Different types of files serve different purposes, such as storing images, text, code, or even process information (stored in virtual files) on UNIX based computer systems.

A more thorough definition of what a file is, as defined by Silberschatz et al. is as follows:

> A file is a named collection of related information that is recorded on secondary storage. From a user's perspective, a file is the smallest allotment of logical secondary storage; that is, data cannot be written to secondary storage unless they are within a file [SPG91, p. 422].

Silberschatz et al. also appropriately note that the concept of a file is purposefully extremely general, as files can hold arbitrary data in forms such as alphabetic, numeric or binary; the data can be structured freely like in the case of text files, or rigidly structured, such as with Portable Network Graphics (PNG) files [SPG91, p. 422].

Files themselves are outwardly characterised by a *file name* and *file extension* with the name used to identify the file to the user, while the extension is normally used to indicate what the contents of the file may be [SPG91, p .427]. Files can also contain other data about the file itself, such as a unique identifier for the file system, information about the file's size, access control information or a timestamp of the last modification of the file [SPG91, p. 422].

The file extension can also help the user decide what program to use to work with the file in question. For example, opening a binary file in a text editor such as `vim` or `gedit` will display a garbled mess of characters, whereas using a specialised program for work with binaries, such as `Ghidra` will allow the user to properly work with the file.

Most commonly, files are stored on the disk as a simple array of bytes (in UNIX based systems, for example), with the file extension providing hints on how these bytes are to be interpreted, while the operating system often makes no assumptions as to how these bytes are to be interpreted [SPG91, p. 428]. Ultimately, this extension can be arbitrary and if, say, a plain text file (`.txt`) would have its extension changed to `.hjkl` the contents of the file would remain unchanged.

If we used a text editor to open this file it would read the contents correctly, regardless of the nonsensical extension name. Similarly, changing the extension back to the original `.txt` would restore every sense of normalcy about the file we would expect – file extension changes made arbitrarily by the user without the use of some conversion algorithm will have no effect on the contents of the file itself.

Notably, however, it's common for operating systems to only allow certain types of operations on certain file types (often based on the extension of the file) – as an example MS-DOS based systems only allow executing files with the `.com`, `.exe` and `.bat` file extensions, with `.com` and `.exe` being executable files, while `.bat` is a batch file containing commands for the operating system to execute, in ASCII [SPG91, p. 427].

2.4.1. File Formats

Though file extensions serve as an indicator to the user, the computer has to know how to parse the raw byte contents of the file, which is what file format specifications are for. File formats are standards used to encode data into binary to be stored in files, as well as to decode it for future use.

There are many ways in which file formats can be misused to hide malicious content (or intent) from the user, however, there is also one very simple and quite common way in which file *extensions* can be used to lull the victim into a false sense of security. The ILOVEYOU virus that circulated in the year 2000 consisted of an email pretending to have a love letter attached, with the file `LOVE-LETTER-FOR-YOU.TXT.vbs` attached [Har01].

It should be immediately obvious to the reader that this file is not a text file, but instead a VBScript (VBS) file. When this file is executed it launches the virus, since the file is a script that can be run on Windows machines and not a text file as it's masquerading to be. The vulnerability at play, so to speak, is the ability to hide file name extensions from being displayed in the GUI of the operating system. In this case, the attackers further tried to mask the real file extension by displaying the fake extension in upper case letters, relegating the real file extension to lower case in an attempt to make it evade the victim's eye.

While the above case preyed on the victim not paying attention to the file name extension, there are also ways in which files can be made to contain content the user would not expect them to. The process of hiding a file inside another file falls within the definition of steganography, which traditionally concerns itself with hiding messages within other messages [Kes04].

Using steganography, we can hide arbitrary data within image files, or even audio files, to be retrieved at a later point by some specialised decoder. A very common method of hiding data within images, mainly concerning the Bitmap Image (BMP), Graphics Interchange Format (GIF) and Joint Photographic Experts Group (JPEG) image formats, is hiding the message within the least significant bit of each byte of the image file [Kes04]. By definition, the least significant bit contributes very little to the byte in question, leading to changes to the least significant bit being difficult if not impossible for the human eye to see [Kes04].

Additionally, it stands to reason that since each bit can only have one or two values, the chance that a bit of our file and the least significant bit of the byte we are hiding our bit in are identical is 50%, so we can expect to have to change only roughly every other bit when hiding our data. This makes the task of recognising a file as a carrier of a message hidden by steganographic means even more difficult.

It's also possible to use the file format specifications in creative ways to craft valid files that have some admittedly strange properties. File format specifications let the program that implements them know how to parse a file in the given format, often by indicating what the file is somewhere in the beginning (or close to the beginning) of the file in a region known as the file header. This file header should appear at the start of the file, but sometimes, this isn't enforced by specific vendors that chose to implement the standard.

Such is the case of the Adobe Acrobat Reader which relaxes the criteria for the Portable Document Format (PDF) file header to not be required at the start of the file, but instead within the first 1024 bytes of the file [ado06]. While this is a deviation from the standard, it has become widespread among other programs and implementations too, due to the dominance of Adobe within the market and can lead to space in the file being able to be used for other intents and purposes.

Generally speaking, file formats that don't require the header of the format to be located at offset 0, right at the very beginning of the file are troublesome from the point of view of being able to conceal data, but even more troublesome are formats that use a *terminator* character or sequence to signal to the program reading the file that the contents of the file are over, even though there may theoretically be further bytes of data following the terminator.

Portable Network Graphics (PNG) File Format

One notable file format specification that uses a terminator to signal the end of usable data is the Portable Network Graphics (PNG). The PNG file format is a thoroughly defined format for storing raster images in a well-compressed, portable and lossless manner, meant as a replacement for the proprietary, patented Graphics Interchange Format (GIF) file format [ISO03]. It's widely used alongside JPEG for storing image files, boasting an overall higher quality to the JPEG format, which uses a lossy compression algorithm to store data.

The PNG file format specifies that each valid PNG file must end with the four byte sequence `73 69 78 68`, also called `IEND` in the specification, which signifies the end of the PNG data stream [ISO03]. The use of this terminator means that programs responsible for reading these PNGs will *ignore* all data that comes after this `IEND` terminator sequence, as they rightfully think the PNG data stream has ended, allowing us to use the remaining bytes of the file to store further, arbitrary data.

Of course, the intention behind this terminator is clear to us, it was meant to be the end of the file and when it comes to regular PNG files it is. However, due to the arbitrary nature of files, nothing is stopping us from appending more bytes of data to the end of this PNG file.

This is the exact trick that was used by the Lazarus APT group in the malicious document that we analyse to smuggle a malicious payload disguised as a simple PNG image embedded in a Word document [Jaz21].

2.5. Microsoft Word Documents

2.5.1. History of the Word Document File Format

.doc

.docx

2.5.2. Macros and Scripting

2.5.3. Use in Malware

Background

1. What is Malware?
2. Attack Vectors
    - Social Engineering
        - Phishing / Spear Phishing
        - Insider Threat
        - Compromised or Weak Credentials
    - Vulnerabilities (CVE)
3. Malware
    - Payload types
    - Concealing / Obfuscation + common techniques
4. Files
    - What is a file
    - Bitmap images
    - Hiding extra content in files
    - File format hacking
5. Word Documents
    - Format description
    - Macros / Scripting
    - Use in Malware
    - What about not using MS Office suite to open documents
6. Lazarus
    - Threat Actor
    - Advanced Persistent Threat
    - Actor Profile – Lazarus
    - WannaCry?

3. Main Idea

1. Lazarus BMP RAT Analysis
2. My re-implementation in key points
3. Tests on different operating systems
    - Current Windows systems
    - EOL Windows systems
    - Linux systems
    - Running the malware *outside* MS Office

4. Implementation

How I recreated the malware

- RAT creation?
- BMP payload creation
- Word script to execute payload
- Obfuscation
- Phishing document creation

5. Evaluation and Discussion

6. Conclusion and Future Work

Bibliography

[42] Unit 42. Ransomware threat report, 2022. `https://start.paloaltonetworks.com/unit-42-ransomware-threat-report.html`.

[ado06] *PDF Reference - Adobe® Portable Document Format, Version 1.7*. Adobe Systems Incorporated, 6 edition, 2006.

[And08] Ross J Anderson. *Security engineering: A guide to building dependable distributed systems*. John Wiley & Sons, Chichester, England, 2 edition, 2008.

[ASA+15] Babak Akhgar, Gregory B Saathoff, Hamid R Arabnia, Richard Hill, Andrew Staniforth, and Petra Saskia Bayerl. *Application of big data for national security: A practitioner's guide to emerging technologies*. Butterworth-Heinemann, Woburn, MA, 2015.

[Ayc06] John Aycock. *Computer Viruses and Malware*. Springer, 2006.

[BD12] Leyla Bilge and Tudor Dumitraş. Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 833–844, 2012.

[BFSS] William Barker, William Fisher, Karen Scarfone, and Murugiah Souppaya. Ransomware risk management: A cybersecurity framework profile. `https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8374.pdf`.

[Cal11] Tracey Caldwell. Ethical hackers: putting on the white hat. *Network Security*, 2011(7):10–13, 2011.

[Coh94] Frederick B Cohen. *A short course on computer viruses*. Wiley professional computing : Computer viruses / security. Wiley, New York, NY [u.a.], 2. ed., 1. [print.]. edition, 1994.

[GSBTLR+17] Pablo L Gallegos-Segovia, Jack F Bravo-Torres, Víctor M Larios-Rosillo, Paúl E Vintimilla-Tapia, Iván F Yuquilima-Albarado, and Juan D Jara-Saltos. Social engineering as an attack vector for ransomware. In *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, pages 1–6. IEEE, 2017.

[HA21] Mohammad Hijji and Gulzar Alam. A multivocal literature review on growing social engineering based cyber-attacks/threats during the covid-19 pandemic: Challenges and prospective solutions. *Ieee Access*, 9:7152–7169, 2021.

[Har01] David Harley. *Viruses revealed*. Osborne/McGraw-Hill, New York [u.a.], 2001.

[Hon12] Jason Hong. The state of phishing attacks. *Communications of the ACM*, 55(1):74–81, 2012.

[ISO03] Portable network graphics (png) specification (second edition). Standard, World Wide Web Consortium, 2003.

[Jaz21] Hossein Jazi. Lazarus apt conceals malicious code within bmp image to drop its rat. `https://blog.malwarebytes.com/threat-intelligence/2021/04/lazarus-apt-conceals-malicious-code-within-bmp-file-to-drop-its-rat/`, Apr 2021.

[KB10] Simon Kramer and Julian C Bradfield. A general definition of malware. *Journal in computer virology*, 6(2):105–114, 2010.

[Kes04] Gary C Kessler. An overview of steganography for the computer forensics examiner. *Forensic science communications*, 6(3):1–27, 2004.

[KHHW15] Katharina Krombholz, Heidelinde Hobel, Markus Huber, and Edgar Weippl. Advanced social engineering attacks. *Journal of Information Security and Applications*, 22:113–122, 2015. Special Issue on Security of Information and Networks.

[KV22] Nir Kshetri and Jeffrey Voas. Ransomware: Pay to play? *Computer*, 55(03):11–13, 2022.

[LG16] Allan Liska and Timothy Gallo. *Ransomware: Defending against digital extortion*. "O'Reilly Media, Inc.", 2016.

[LTT+21a] Ifigeneia Lella, Marianthi Theocharidou, Eleni Tsekmezoglou, Apostolos Malatras, and European Union Agency for Cybersecurity. *ENISA threat landscape 2021: April 2020 to mid July 2021.* Publications Office, 2021.

[LTT+21b] Ifigeneia Lella, Marianthi Theocharidou, Eleni Tsekmezoglou, Apostolos Malatras, European Union Agency for Cybersecurity, Sebastian Garcia, Valeros Veronica, and Czech Technical University in Prague. *ENISA threat landscape for supply chain attacks.* Publications Office, 2021.

[MLV16] Francois Mouton, Louise Leenen, and H.S Venter. Social engineering attack examples, templates and scenarios. *Computers & Security*, 59:186–209, 2016.

[Mor] Morphisec. Morphisec threat report 2021. https://engage.morphisec.com/2021-morphisec-threat-report.

[PM21] Bryson Payne and Edward Mienie. Multiple-extortion ransomware: The case for active cyber threat intelligence. In *ECCWS 2021 20th European Conference on Cyber Warfare and Security*, page 331. Academic Conferences Inter Ltd, 2021.

[Seca] IBM Security. Cost of a data breach report, 2020. https://www.ibm.com/account/reg/us-en/signup?formid=urx-46542.

[Secb] IBM Security. Cost of a data breach report, 2021. https://www.ibm.com/account/reg/us-en/signup?formid=urx-50915.

[sno] U.s. charges snowden with espionage. https://www.washingtonpost.com/world/national-security/us-charges-snowden-with-espionage/2013/06/21/507497d8-dab1-11e2-a016-92547bf094cc_story.html.

[SPG91] Abraham Silberschatz, James L Peterson, and Peter B Galvin. *Operating system concepts*. Addison-Wesley Longman Publishing Co., Inc., 1991.

[SZ03] Ed Skoudis and Lenny Zeltser. *Malware: Fighting Malicious Code*. Prentice Hall, 2003.

[TP17] Jinal P Tailor and Ashish D Patel. A comprehensive survey: ransomware attacks prevention, monitoring and damage control. *Int. J. Res. Sci. Innov*, 4(15):116–121, 2017.

[VHIB12] Renier P Van Heerden, Barry Irwin, and Ivan Burke. Classifying network attack scenarios using an ontology. In *Proceedings of the 7th International Conference on Information-Warfare & Security (ICIW 2012)*, pages 311–324, 2012.

[Whi21] Lance Whitney. Ransomware attackers are now using triple extortion tactics. https://www.techrepublic.com/article/ransomware-attackers-are-now-using-triple-extortion-tactics/, 5 2021. [Online, accessed 11-05-2022].

[WPH+21] Suzanne Widup, Alex Pinto, David Hylender, Gabriel Bassett, and philippe langlois. Verizon data breach investigations report, 2021. https://www.researchgate.net/publication/351637233_2021_Verizon_Data_Breach_Investigations_Report, 05 2021.

A. Appendix

here you can put further things you want to add like transcripts, questionnaires, raw data...