Malware development threats with modern technologies

Received (in revised form): 26th July, 2022

Lawrence Amer
Cybersecurity Manager, PwC, Hong Kong

Lawrence Amer is red teaming and research lead at PwC Hong Kong (DarkLab). He has practical knowledge of advanced red teaming and advisory simulation with extensive experience in penetration testing and cloud security. Lawrence's expertise In security research is acknowledged by different vendors across the region for reporting medium to high severity vulnerabilities (SteelSeries, Microsoft, SAP, Yahoo).

PwC, Hong Kong
Mob: +852 68952893; E-mail: lawrence.amer@hk.pwc.com; contact@lawrenceamer.me

Abstract Despite a significant increase in the level of defence strategies across the enterprise, cyberattacks continue to have a damaging impact on organisations. Due to insufficient threat intelligence capabilities established in many organisations, attackers use this weakness to port their attack procedure and plan future attacks. From highlighting the problem to solving it, this paper explores possible attack procedures and builds awareness to shortcut the risk and reduce the possibility of compromise. It describes a case study of cyberattacks to illustrate the pros and cons of advanced detection and prevention systems.

KEYWORDS: malware, red teaming, simulated attacks, threat actors, detection, prevention, emulation

INTRODUCTION

Cyber security is a complex problem. There are many potential attack vectors and a wide variety of possible targets. A comprehensive security strategy must take all of these factors into account. To stay ahead of the curve, security professionals must be proactive in their approach to managing risk. One way to do this by modelling potential cyber security solutions. This paper explores the risk of recently developed techniques for offensive campaigns and engagements.

In order to defend and build solid defensive strategies, it is required to understand the attacking procedures or tactics of hackers. These include the highlights of scoped cyberattack levels and targeted infrastructures, operating systems (OS) and software. It is claimed that more than 70 per cent of attacks target Windows OS by exploiting system weakness or human-based errors. Studies show that 90 per cent of successful malware attacks between 2010 and 2018 were delivered through e-mails, approximately 51 per cent were trojans, and seven out ten of these malicious payloads were ransomware.

In 2021, more than 60 per cent of organisations suffered malware attacks, increasing by 15 per cent in 2022. According to reports, this increase is the result of malware spread from employee to employee or through phishing targeted campaigns.[1] Statistics shared by Veeam,[2] the largest independent research project in the data protection industry, summarised a few of the key findings:

- Only 24 per cent of organisations were not attacked by ransomware;
- Sixteen per cent were attacked once in 2021;
- Sixty per cent were attacked more than once in 2021.

A recent technical analysis conducted by the IBM-XForce[3] team suggests that most ransomware investigations in 2019 were associated with an initial TrickBot infection resulting in Ryuk ransomware attack. In these attacks, Empire was the most frequent tool for 37 per cent of all interactive session tools. In 2020, tools such as Empire were replaced with Cobalt Strike. X-Force observed that ransomware attacks continued to rely upon many of the same protocols and default permissions utilised in 2019 and 2020 to achieve their goal (see Figure 1).[4]

By reading shared reports' key highlights[5] or information,[6] we conclude that most of these attacks have been successfully carried out. In contrast, security solutions are turned on and well configured, which leads to open questions about the tactics used and how hackers build an awareness of how an antivirus (AV) or advanced monitoring service works.

Root methods

Although new attacks come with different base codes, security solutions guarded the OS before reaching any OS level execution, despite kernel or user mode landing.

There are essential factors that stop and prevent an offensive engagement, categorised as defence layers or multilayers of enterprise protection mechanisms such as:

- Binaries' popular obfuscations techniques;
- Memory execution monitoring in place;
- Tampering analysis technologies implemented;
- Indicator of compromise shipped frequently.

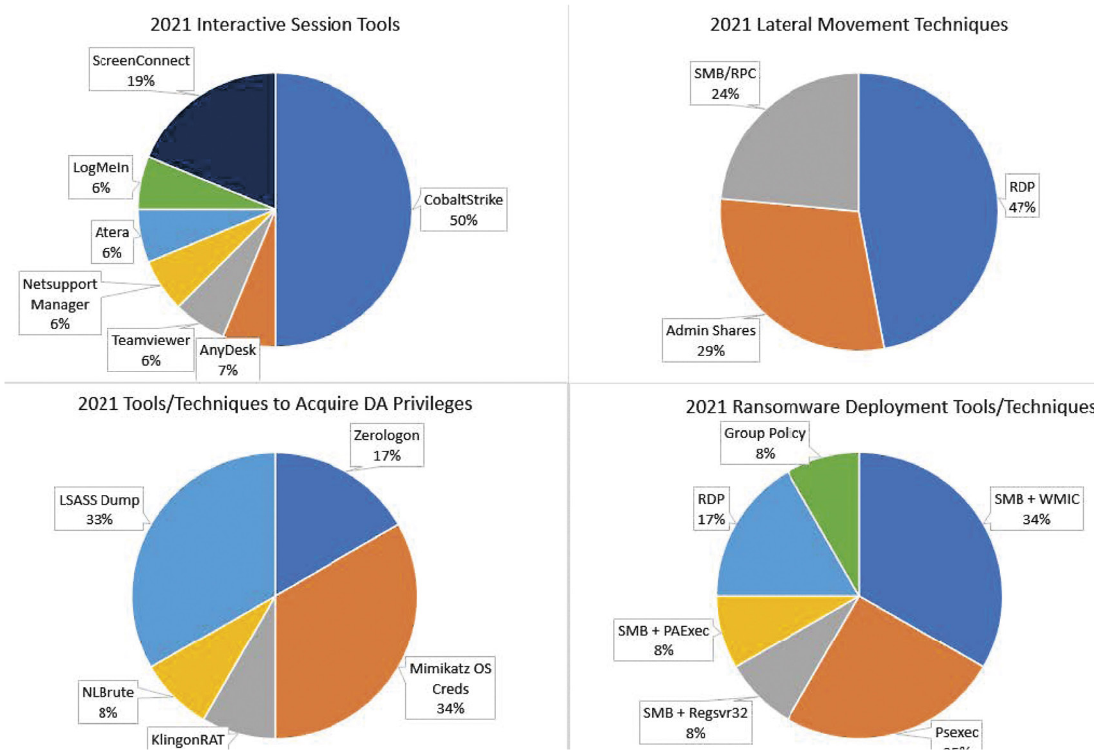


Figure 1: Tool techniques and procedures

On the other hand, there are root methods of exploitation that pose a risk to an organisation (see Table 1).[7]

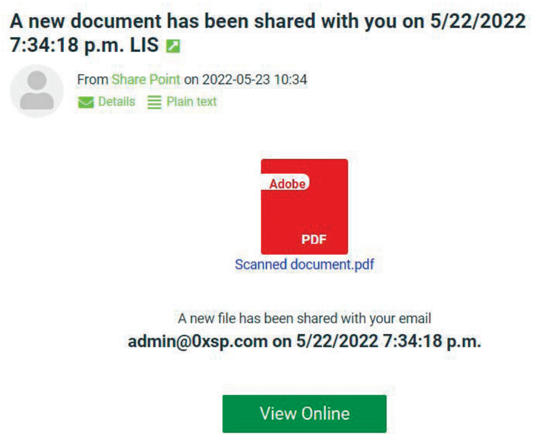
Social engineering

Since employees are on the frontline of defence in any organisation, they are considered the weakest part of the whole defence cycle. Attackers could leverage employees' lack of security awareness to gain internal foothold access in several ways, such as phishing attacks (e-mails, short messaging service [SMS], social networks, etc.).

Figure 2 illustrates a phishing campaign attempt. Since it appears legitimate, users who do not have enough knowledge could fall into a trap and click on or even open a malicious document.

Table 1: Root methods of exploitation

Method	Rank/popularity
Social engineering	54% of ransomware attacks[8]
Unpatched software	8% of cyberattacks[9]
Password reuse/guessing	19% of ransomware attacks[10]
Exposed remote desktop protocol (RDP) access	20% of ransomware attacks[11]
Insufficient access control	21% of ransomware attacks[12]



A new document has been shared with you on 5/22/2022 7:34:18 p.m. LIS

From Share Point on 2022-05-23 10:34

A new file has been shared with your email admin@0xsp.com on 5/22/2022 7:34:18 p.m.

Figure 2: Malicious document via phishing campaign

Unpatched/outdated software

Having unpatched software/products is an absolute risk. Threat actors target users or even online services which could be susceptible to a known vulnerability. For example, Follina (CVE-2022-30190) Microsoft Office Zeroday has become the most used exploit payload by threat actors due to no existing patch deployed.[13]

Password guessing and reuse

Unsurprisingly, password guessing accounts for 19 per cent of ransomware attacks.[14] Whether a red teaming simulation or regular pen testing, it still counts that using a weak or cached password is a typical security finding.

Despite the fact that two-factor authentication (2FA) can be used to protect against these types of attack, threat actors can still obtain initial access to an organisation's employees' accounts, whether on cloud or on-premise, using a weak password, password reuse or combined with some social engineering development skills such as recent technique shared by a security researcher,[15] which could trick users to bypass 2FA protection. Sometimes, they perform single password spraying to check if other users use the same password.

Exposed RDP access

Remote desktop protocol (RDP) exposure is one of the common risks for companies; according to research shared on ResearchGate, over 4,493,357 RDP services were on Shodan between 2017 and 2020, with more than 1,400 of these located in the US (see Figure 3).[16] The research also indicates that healthcare is one of the industries most vulnerable to cyberattacks as it evolves rapidly and shifts to digitally enabled healthcare services. Open-source intelligence tools such as Shodan or alternatives make it easy for threat actors to spot the exposed RDP ports, allowing them to gain unauthorised access to these services.

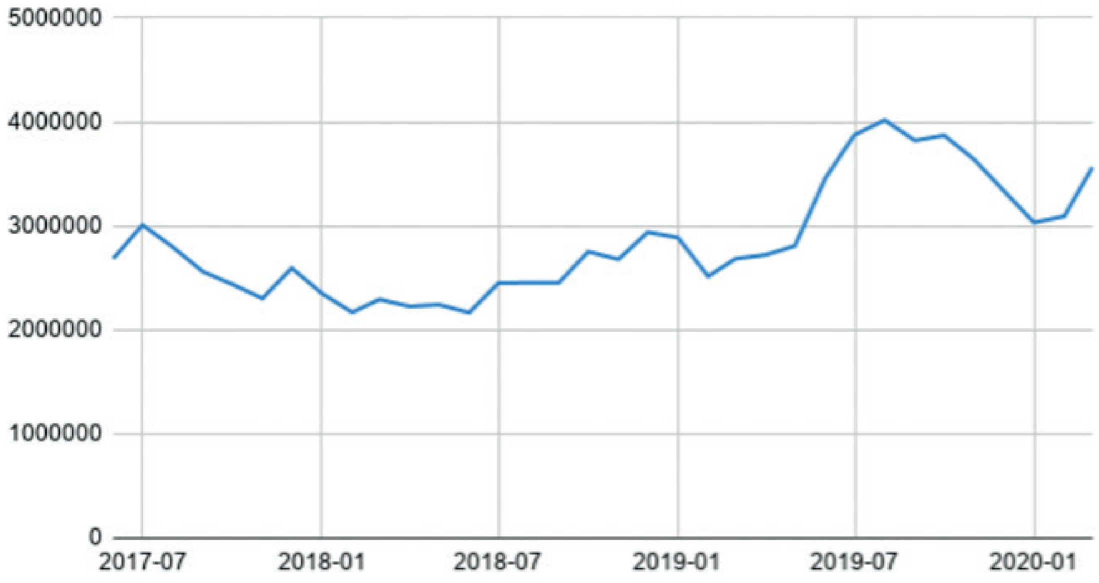


Figure 3: The increase in incidence of exposed RDP services

Insufficient access controls

This stage comes after the initial infiltration. Attackers will then try to move laterally across the network to identify prized assets and locate any critical services to exploit or breach out. Leveraging poor access management could allow threat actors to compromise and obtain unauthorised access to other resources inside the organisation's network.

Build a case

To build effective defensive rules, one must first identify offensive tactics and techniques adopted by threat actors, by collecting indicators of compromise and developing detection rules.[17] This is reminiscent of the famous quote of Sun Tzu:[18] 'To know your enemy, you must become your enemy.'

A report shared by the Global Research and Analysis team (GReAT) at Kaspersky lists the main advanced persistent threat (APT) trends in Q1 2022.[19] Geopolitics has always been the key driver of APT development.

Further development of low-level implants such as MoonBounce unified extensible firmware interface (UEFI) malware has hit the headlines due to the novel way it hides from AVs.[20] The malware works by bypassing the protection offered by UEFI and modifying the Windows boot loader stored in the EFI system partition (ESP). However, MoonBounce does not target the ESP, rather it resides within a small chip on the motherboard called the serial peripheral interface (SPI). Furthermore, injecting itself into the data stream during the initial system boot at this stage makes the malware invisible to any security scanner that examines the content of the ESP.

On 20th January 2022, Kaspersky published a report titled 'MoonBounce: The dark side of UEFI Firmware', which concluded:

- The inspected UEFI firmware was tampered with to embed a malicious code;
- The purpose of the implant is to facilitate the of deployment user-mode malware that stages the execution of further payloads downloaded from the Internet;
- The infection chain itself does not leave any traces on the hard drive, as its components only operate in memory.

The new type of firmware rootkit, CosmicStrand,[21] was discovered recently by security researchers. Such types of malware ensure a computer remains in an infected state even if the OS is reinstalled.[22]

According to Kaspersky and Qihoo360,[23] the recent rootkit is associated with unknown Chinese-speaking threat actors and it is still not known how the initial infections happened. The goal of the attack is to tamper with the OS loading process to deploy a kernel-level implant into a Windows machine every time it is booted, using the entrenched access to launch shellcode that connects to a remote server to fetch the actual malicious payload to be executed on the system.[24,25]

INTRODUCTION OF MALWARE DEVELOPMENT

Development of malware that eludes detection of an anti-malware solution is challenging and exceptional. Recent estimates by Symantec claim that malware continues to grow in quality and quantity. Malware authors have continued to discover new platforms to feast on and new ways to avoid detection.[26]

In the early 1990s, a German programmer named knzyvo[27] created a disk operating system (DOS) virus called Whale,[28] one of the outstanding features of which was polymorphous mechanisms. Although the program code repeated the same procedure with each infection, it was designed to look different each time.[29]

Many different malware classes were spotted between 1971, starting with Creeper System malware, and 1999, with a worm variant dubbed Kak Worm. This was a torment for the newly established AV solution to stop, as it responded with simple pattern-based approaches.

One of the enduring fascinations of malware development is the developers' desire to be creative and devise something unique. However, it is still unusual to encounter advanced tactics such as obfuscation, encryption or diverting AV sandbox emulators. Between 2011 and 2020, anti-malware companies started to unleash enterprise solutions to predict and perform malware analysis using sandboxing, which makes an intelligent decision whether an object is malicious or safe to use based on its behaviour.

Sandboxing or code emulators technologies are powerful tools against newly deployed malware. However, even if this malware is not known before or has been programmed with a unique code structure, it will be detected due to execution tracing collected artefacts.

For example, suppose there is a custom executable containing functions that delete volume shadow service (VSS) copies. In that case, an anti-malware solution will flag it as malicious, identified as a possible ransomware variant depending on the collected artefacts from the initial execution of the custom executable inside the AV sandbox emulator.

According to Kaspersky Labs, the emulator executes the object's instructions one by one in a safe virtual environment, collects artefacts and passes them to the heuristic analyser to detect malicious behaviour features of a binary file or a script (see Figure 4).[30]

An emulator emulates only the execution of the sample itself. It temporarily creates objects that the sample interacts within. On the other hand, unlike an emulator, a full-featured sandbox is a 'heavyweight' method. It emulates the whole environment and runs a scanned sample in a virtual machine with a regular OS and applications installed.

Detection workflow

Modern consumer AV software is highly complex and uses several techniques to identify malware, such as hashing, static signaturing or static heuristic analysis.[31] Moreover, anti-malware companies adhere to the emulator and sandboxing technologies that share the standard detection workflow, detailed in the following execution flow:

- The emulator receives a request to scan an object (an executable file or a script) from another component of a security solution;
- The emulator safely executes the object's instructions in a virtual environment, starting from the object's entry point. If an instruction interacts with the environment (OS, registry, other files, web, memory), the emulator imitates a response from these objects;
- The emulator collects artefacts and passes them to the heuristic analyser. The analyser passes a verdict based on these artefacts to the component that requested the analysis;
- The emulation stops.

Collected artefacts by emulator:

- Binary scripts;
- Application programming interface (API) call log;
- All changes in the file system, system registry;
- Arguments and returns of string operations;
- Calls of embedded functions and functions provided by the environment;
- Event drops and child (simulation) scripts;
- Memory dumps.

However, it is slightly different when dealing with a sandboxing environment.[32] The sandbox receives a request to scan an object (a file or URL) with instructions: the OS and the configuration for running the object, the object's execution parameters and other third-party applications installed in the virtual machine (VM). After running the object, the sandbox collects artefacts throughout the specified timespan. For example, if the object interacts with other processes or URLs with known reputations, the sandbox captures this. The sandbox analyses artefacts and delivers its verdict to the requesting system: malware or benign. If a particular suspicious activity is found during the sample's execution, the sandbox also returns a detailed description of the activity.

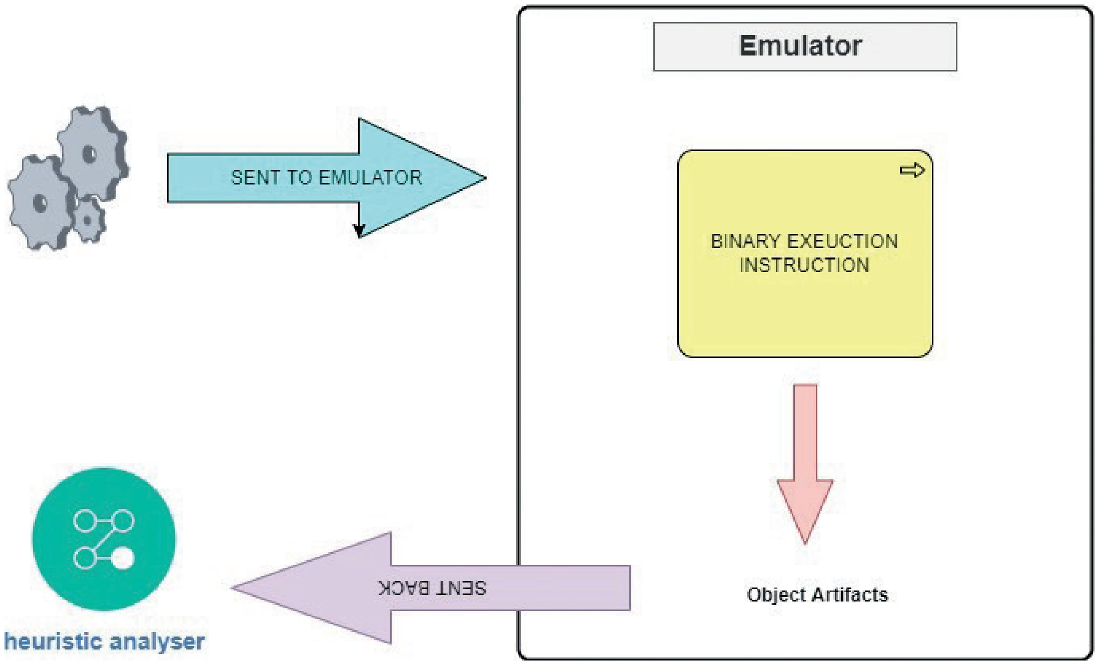


Figure 4: The emulator

Artefacts collected by the sandbox:

- Application execution logs (API function calls with their parameters and execution events);
- Memory dumps;
- Loaded modules;
- Changes in a file system/registry;
- Network traffic (packet capture [PCAP] files);
- Screenshots;
- Artefacts of exploit activity.

Evasion techniques

From the perspective of malware development, it is typical in some cases to detect anti-malware emulators. However, it is challenging to do that inside a sandbox environment which is regularly enhanced to detect any evasion technique.

Taking the Kaspersky antivirus emulator (KAV) as an example for our case study, the following points allow KAV to detect any evasion technique:

- The emulator recognises packed files and adjusts emulation depth accordingly;
- Hardware acceleration gives the emulator enough power to pass through unpacking;
- The emulator imitates information about the environment and system resources.

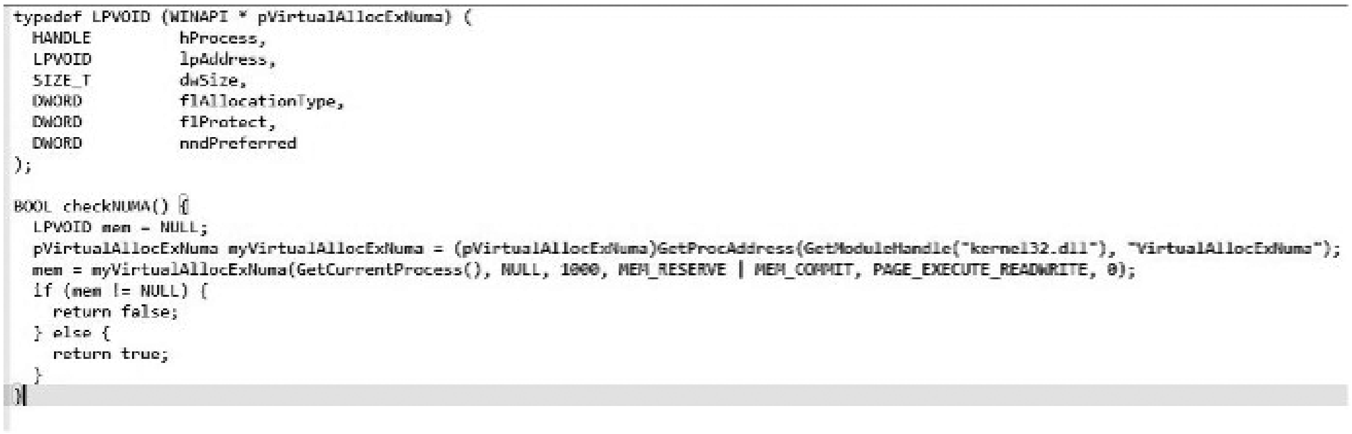
It is possible to bypass these enabled hooks by implanting a specific Win32 API function at the main execution entry point of a binary.

Since memory allocation could not work properly inside an AV emulator environment, malware could use such a technique to detect an emulator (see Figure 5) and force the program to exist before the AV solution could trace out execution events of the malware and flag the malicious functions.[33]

Due to the main execution time limitation of the AV emulation scanner, AV will not spend much time and central processing unit (CPU) resources on one object as it could also lead to denial of service (DoS).[34] As a result, malware developers use a method to require the AV to go through a code that consumes more memory and CPU resources, forcing AV to abandon it from scanning the remaining code blocks. For example, the code in Figure 6 demonstrates the technique by filling a vast amount of memory, which would stop the AV from handling this massive portion of allocated memory resources.

However, performing the previous techniques in the sandboxing environment would fail and be detected, as sandbox technologies are realistic execution environments.

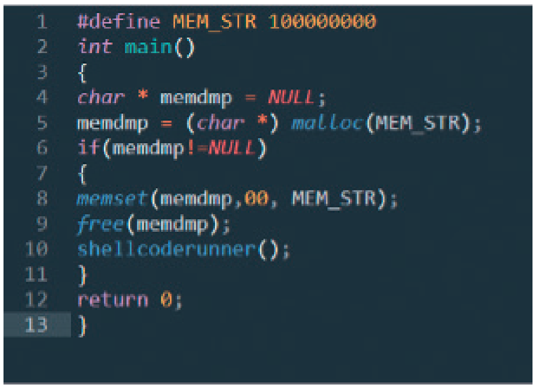
Sandbox runs a suspicious object in a VM with a fully featured OS and detects the object's malicious activity by analysing its behaviour.[35] It is challenging to deploy an



```
typedef LPVOID (WINAPI * pVirtualAllocExNuma) (
  HANDLE         hProcess,
  LPVOID         lpAddress,
  SIZE_T         dwSize,
  DWORD          flAllocationType,
  DWORD          flProtect,
  DWORD          nndPreferred
);

BOOL checkNUMA() {
  LPVOID mem = NULL;
  pVirtualAllocExNuma myVirtualAllocExNuma = (pVirtualAllocExNuma)GetProcAddress(GetModuleHandle("kernel32.dll"), "VirtualAllocExNuma");
  mem = myVirtualAllocExNuma(GetCurrentProcess(), NULL, 1000, MEM_RESERVE | MEM_COMMIT, PAGE_EXECUTE_READWRITE, 0);
  if (mem != NULL) {
    return false;
  } else {
    return true;
  }
}
```

Figure 5: Check NUMA code snippet technique



```
#define MEM_STR 100000000
int main()
{
char * memdmp = NULL;
memdmp = (char *) malloc(MEM_STR);
if(memdmp!=NULL)
{
memset(memdmp,00, MEM_STR);
free(memdmp);
shellcoderunner();
}
return 0;
}
```

Figure 6: Memory-consuming code

evasion technique to defeat the sandboxing and avoid behaviour analysis with such a technology. Sandbox allows researchers to visualise the behaviour of malware in a short period of time and eliminate the tactic.[36] Nevertheless, everything is possible with some combined tricks.

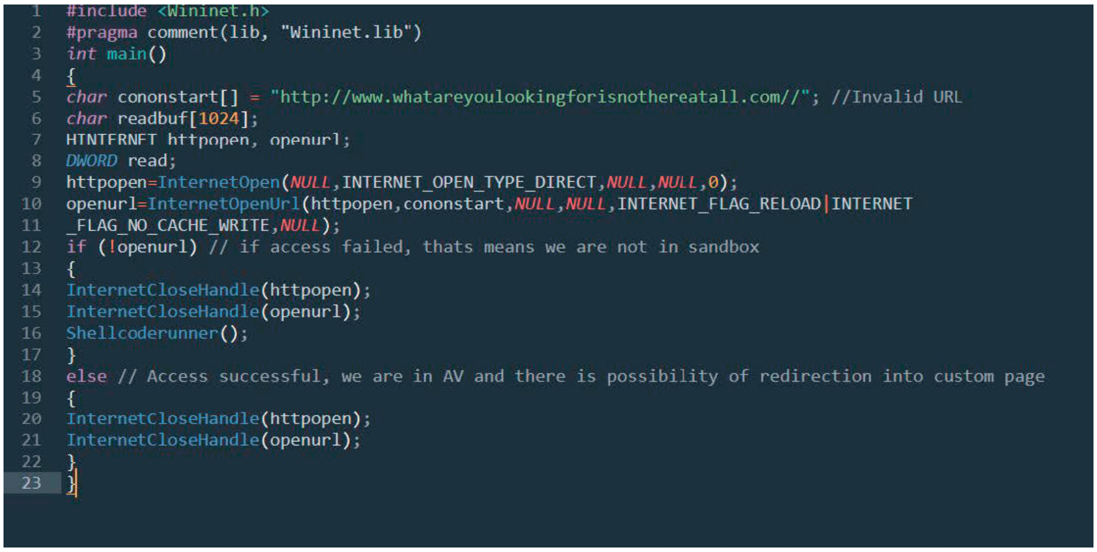
Implementing an inactive domain checker (see Figure 7) is a classic method that works well with some sandboxing products. Such a technique aims to add a URL check function that will try to load a page from a parked domain name. Furthermore, the code is executed in the sandbox environment if it works successfully.

Many emulators will fail to interact with interactive visual inputs. For example, the malware authors could add a pop-up form at the initial execution of the payload and ask users to enter specific characters to begin the execution or abort, which would fail in an emulated environment. Attackers could steal the information submitted using the wincred.h library.[37,38]

The virtual environment randomises the names of the allocated resources, such as computer names or usernames, or other system information. Malware developers could leverage the collected insightful information from their targets and build a code function to determine if their target usernames already exist in the Windows environment. At the same time, this will not work in the virtual environment as malware developers could force a smooth execution termination if their lookup results did not match.

SUMMARY

Since approximately 60 per cent of malware attacks in 2021 were ransomware, this percentage could significantly increase during 2022 combined with the exploitation



```
#include <Wininet.h>
#pragma comment(lib, "Wininet.lib")
int main()
{
char cononstart[] = "http://www.whatareyoulookingforisnothereatall.com//"; //Invalid URL
char readbuf[1024];
HINTERNET httpopen, openurl;
DWORD read;
httpopen=InternetOpen(NULL,INTERNET_OPEN_TYPE_DIRECT,NULL,NULL,0);
openurl=InternetOpenUrl(httpopen,cononstart,NULL,NULL,INTERNET_FLAG_RELOAD|INTERNET
_FLAG_NO_CACHE_WRITE,NULL);
if (!openurl) // if access failed, thats means we are not in sandbox
{
InternetCloseHandle(httpopen);
InternetCloseHandle(openurl);
Shellcoderunner();
}
else // Access successful, we are in AV and there is possibility of redirection into custom page
{
InternetCloseHandle(httpopen);
InternetCloseHandle(openurl);
}
}
```

Figure 7: In-active domain checker code function

of uncovered zero-days including more advanced infiltration tactics.

Malware development is designed to test the security defence of an environment without having a negative impact on it. Putting efforts into learning the attack cycle would aim to ease the process of building the perfect prevention technologies.

Despite AV companies introducing several enterprise products to detect and predict any possible malicious indicators, it is still challenging for them to stop some unique prevention tactics.

References

1. Sobers, R. (2021 [updated July 2022]), '134 Cybersecurity Statistics and Trends for 2021', Varonis, available at https://www.varonis.com/blog/cybersecurity-statistics (accessed 26th July, 2022).
2. Russell, D. and Buffington, J. (May 2022), 'Announcing the 2022 Ransomware Trends Report', Veeam, available at https://www.veeam.com/blog/2022-ransomware-trends-report.html (accessed 26th July, 2022).
3. Perera, M., Alexander, J. and Garcia, A. Q. (June 2022), 'Countdown to Ransomware: Analysis of Ransomware Attack Timelines', Security Intelligence, available at https://securityintelligence.com/ posts/analysis-of-ransomware/ (accessed 26th July, 2022).
4. *Ibid*.
5. *Ibid*.
6. Sobers, ref. 1 above.
7. Carpenter, P. (December 2021), '5 Defenses for 5 Ransomware Root Causes', CPO Magazine, available at https://www.cpomagazine.com/cyber-security/5-defenses-for-5-ransomware-root-causes/ (accessed 26th July, 2022).
8. Datto (2020), 'Datto's Global State of the Channel Ransomware Report', available at https://www.datto.com/resource-downloads/Datto-State-of-the-Channel-Ransomware-Report-v2-1.pdf (accessed 26th July, 2022).
9. EdgeScan (2020), '2020 Vulnerability Statistics Report', available at https://cdn2.hubspot.net/hubfs/4118561/BCC030%20Vulnerability%20Stats%20Report%20(2020)_WEB.pdf (accessed 26th July, 2022).
10. Wharton, G. (2021), 'Hiscox Cyber Readiness Report: Don't let cyber be a game of chance', Hiscox, available at https://www.hiscoxgroup.com/sites/group/files/documents/2021-04/Hiscox%20Cyber%20Readiness%20Report%202021.pdf (accessed 26th July, 2022).
11. Datto, ref. 7 above.
12. *Ibid*.
13. Furner, C. (May 2022), 'Microsoft RCE "Follina" Zero-Day (CVE-2022-30190) Found In MSDT, Office', available at https://www.blumira.com/cve-2022-30190-follina/ (accessed 26th July, 2022).
14. Wharton, ref. 9 above.
15. Media (June 2022), 'Microsoft WebView2 apps used in novel phishing technique', available at https://www.scmagazine.com/brief/social-engineering/microsoft-webview2-apps-used-in-novel-phishing-technique (accessed 26th July, 2022).
16. Al Qartah, A. (August 2020), 'Evolving Ransomware Attacks on Healthcare Providers', Thesis for Master of Science in Cybersecurity, available at https://www.researchgate.net/figure/RDP-Exposure-Measured-by-Shodan-Matherly-J-2020-In-addition-by-performing-a-Shodan_fig2_344450646 (accessed 26th July, 2022).
17. Crowdstrike (May 2021), 'Indicator of compromise (IOC) Security', available at https://www.crowdstrike.com/cybersecurity-101/indicators-of-compromise/ (accessed 26th July, 2022).
18. Tzu, S., 'The Art of War', Wikipedia, available at https://en.wikipedia.org/wiki/The_Art_of_War (accessed 26th July, 2022).
19. Global Research & Analysis Team, Kapersky Lab (April 2022), 'APT trends report Q1 2022', SecureList, available at https://securelist.com/apt-trends-report-q1-2022/106351/ (accessed 26th July, 2022).
20. SecureTeam (January 2022), 'What is Moonbounce Malware?', available at https://secureteam.co.uk/articles/information-assurance/what-is-moonbounce-malware/ (accessed 26th July, 2022).
21. Global Research & Analysis Team, Kapersky Lab (July 2022), 'CosmicStrand: The discovery of a sophisticated UEFI firmware rootkit', SecureList, available at https://securelist.com/cosmicstrand-uefi-firmware-rootkit/106973/ (accessed 26th July, 2022).
22. Laksmanan, R. (July 2022), 'Experts Uncover New "CosmicStrand" UEFI Firmware Rootkit Used by Chinese Hackers', The Hacker News, available at https://thehackernews.com/2022/07/experts-uncover-new-cosmicstrand-uefi.html (accessed 26th July, 2022).
23. Global Research & Analysis Team, Kapersky Lab, ref. 21 above.
24. Laksmanan, ref. 22 above.
25. Global Research & Analysis Team, Kapersky Lab, ref. 21 above.
26. See Broadcom, available at https://community.broadcom.com/symantecenterprise/communities/community-home/librarydocuments/viewdocument (accessed 26th July, 2022).
27. The Malware Wiki, 'Whale', available at https://malwiki.org/index.php?title=Whale (accessed 26th July, 2022).
28. *Ibid*.
29. Ruef, M. (2017), 'Malware Development – Professionalization of an Ancient Art', *Malware Analysis and Development*, available at https://www.researchgate.net/publication/336838505_Malware_Development_-_Professionalization_of_an_Ancient_Art/link/

5db5d3b192851c577ecebd16/download (accessed 26th July, 2022).
30. Kapersky, 'Emulator', available at https://www.kaspersky.com/enterprise-security/wiki-section/products/emulator (accessed 26th July, 2022).
31. Blackthorne, J., Bulazel, A., Fasano, A., Biernat, P. and Yener, B. (August 2016), 'AV Leak: Fingerprinting Antivirus Emulators through Black-Box Testing', WOOT, available at https://www.usenix.org/system/files/conference/woot16/woot16-paper-blackthorne.pdf (accessed 26th July, 2022).
32. Kapersky, 'Sandbox', available at https://www.kaspersky.com/enterprise-security/wiki-section/products/sandbox (accessed 26th July, 2022).
33. Oxsp SRD (May 2022), 'Mortar Loader v2', available at https://0xsp.com/offensive/mortar-loader-v2/\#Divert_sandbox_emulation_detection (accessed 26th July, 2022).
34. Nasi, E. (2014), 'Bypass Antivirus Dynamic Analysis', WikiLeaks, available at https://wikileaks.org/ciav7p1/cms/files/BypassAVDynamics.pdf (accessed 26th July, 2022).
35. Kapersky, ref. 32 above.
36. Roccio, T. (September 2019), 'Evolution of Malware Sandbox Evasion Tactic – A Retrospective Study', McAfee, available at https://www.mcafee.com/blogs/other-blogs/mcafee-labs/evolution-of-malware-sandbox-evasion-tactics-a-retrospective-study/ (accessed 26th July, 2022).
37. Oxsp SRD (April 2022), 'Hunting Windows Credentials (CredUIPromptForWindowsCredentials)', available at https://0xsp.com/offensive/hunting-windows-credentials-creduipromptforwindowscredentials/ (accessed 26th July, 2022).
38. Microsoft (July 2022), 'CredUIPromptForWindowsCredentialsAFunction (wincredh)', available at https://docs.microsoft.com/en-us/windows/win32/api/wincred/nf-wincred-creduipromptforwindowscredentialsa (accessed 26th July, 2022).