**Malware Forensics (MalFor)**

**Coursework**

**REPORT**

Student No.: UP2009045

Word Count:  2042

**Disclaimer**  
By submitting this report for marking, **I fully understand** that and **I agree**:

- This is an independent piece of coursework, and it is expected that I have taken responsibility for all the design, implementation, analysis of results and writing of the report. And, it is 2000 words (+/-10%).

- For the Turnitin plagiarism software, the match score of the report must be below 15% overall and less than 5% to an individual source (excluding appendices).  Marks will be investigated accordingly. if appropriate.

- This marking scheme sets out what would be expected for each marking band for this module’s coursework.

- The University “general criteria applicable to essays, reports and aspects of projects and dissertations” applies (on Moodle).

- It is blind marked.  
- Any technical help, high-level advice and suggestions which may be provided in-person (e.g., labs) and electronically, has no contribution to or an indication of my final mark.  
- Knowledge of a peer’s work and their final mark is not justification for what my own mark should be.

- Any examples provided were used for ideas only, and “having followed an example” is not justification for what my mark should be.

- The coursework malware was used for analysis only and was isolated to a safe working  
environment (a virtual machine), to prevent the malware from infecting the host.

# Summary

Longtext who have been asked to investigate data exfiltration regarding Pavel Checkov and core IP. Longtext have identified a single server in relation to the malware which was executed by the user along with various commands. The server identified is a command and control (C2) server for an internet relay chat (IRC) botnet. Once the malware is executed on the system, it seems as if it is no longer running however, continues to run in the background; for a standard system user this would be hard to spot and report.

Further investigation revealed the C2 server was communicating over a default port (6667) for IRC traffic and naming conventions were also suspicious with names being ‘bot’ followed by random numbers. Following on from this, a ‘#malfor-woods’ channel was found which is password protected and is where the bot-herder conducts commands.

Two passwords were discovered during both static and dynamic analysis, ‘fancybear’ a static un-encrypted password assigned to each bot which provided us with access to the IRC server and ‘richmond’ which provided access to the channel for controlling the botnet.

During further analysis and debugging using IDA, we found there were various commands expecting to be received by the malware listed in Table 1. During creation, the author, who we believe to be Julian Murphy also implemented anti-debugging techniques to prevent disassembly of this executable using programs such as IDA.

Finally, we have included recommendations for remediation along with future prevention within the body of this report.

# Introduction

Following a recent security incident involving a Gazprom employee exfiltrating company plans for a new gas pipeline, it has been revealed that this employee infected company systems with malware which requires further investigation. Gazprom has provided Longtext with the malware in question for us to conduct the investigation of which the details, steps and the outcome will be documented here.

This malware analysis will review how the malware functionality and effectiveness in exfiltrating said information via static and dynamic analysis, analysis of any associated infrastructure and dis-assembly of the malware functionality.

Following the analysis of the malware, we Longtext will provide some remedial steps which can be used to remove this risk and recommendations to prevent this risk in future.

# Malware Analysis

## Static analysis

Beginning with static analysis, we firstly analysed the malware file provided by Gazprom using a combination of:

* Strings - This gathers whole text and numerical information from the malware files.
* PEiD – This allows us to identify if the file is packed or not along with displaying any dynamic link libraries (DLL) it imports/exports.
* PEView – Allows for analysis of the file structure viewing file properties, headers etc.

VirusTotal – Usage of 72 well known and accurate malware engines to scan and provide information on the file along with gathering information such as previous community comments on the file.

This is done without executing the malware, however, is still conducted within a safe environment such as a virtual machine where the malware would not affect the host and external domain.

### Strings

A computer screen shot of a computer

Description automatically generated

Figure 1 - Strings output identifying IP address, operating system and ports 80 & 6667

Beginning with Strings, Figure 1 displays the output after running strings on the malware file Gazprom provided. Looking over this output, we noticed a few pieces of information:

Firstly, the block of text at the bottom of Figure 1 identifies an internet relay chat botnet which would go hand in hand with “6667”. This is the default port for IRC traffic (Wireshark, 2020), we can also see a public IP address 167.99.88.222 and a potential operating system OSX, which could mean this is being hosted on a Mac and using a bash terminal.

A building with a large area of land

Description automatically generated with medium confidence

Figure 2 - Equation Group website linked to discovered IP

As displayed in Figure 2, after visiting the IP address found in the malware this displayed a standard webpage with “EQUATION GROUP”. Further research into this group shows the threat actor group has an expansive past in cyber attacks (Checkpoint Research Team, 2015); further analysis did not reveal anything, for example looking up robots.txt led to a link which would not load. However, here we can see the site does not have SSL which explains the localhost:80 found previously in strings.

### PEiD

A screenshot of a computer

Description automatically generated

Figure 3 - PEiD output displaying unpacked .exe file

Moving over to PEiD, after performing a hardcore scan on the malware file provided was built in Boreland Delphi shown in Figure 3. Delphi is no longer a commonly used program and although this file uses Delphi, the file was never packed during creation.

A screenshot of a computer

Description automatically generated

Figure 4 – List of imports

The hardcore scan also reveals a list of DLLs that are used within the malicious executable and reviewing these does not reveal any DLL in relation with the previously found botnet.

### PEView

A screenshot of a computer

Description automatically generated

Figure 5 - File header information

Reviewing the malware file header in PEView shows when the malware was created, in this case October 31st 2023 and was intended for 32-bit systems, however, this could be to maximise the attack surface as 32-bit programs will work on 64-bit systems but not the other way around.

### VirusTotal

A screenshot of a computer

Description automatically generated

Figure 6 - VirusTotal scan result

A screenshot of a computer

Description automatically generated

Figure 7 - Vendor results

Scanning the file in through virus total, Figure 6, shows 60 out of 72 vendors detected this file was malicious and in Figure 7 we can see this was detected as an IRC botnet/banload by most vendors

## Dynamic analysis

Following on from and using the information gathered during static analysis, dynamic analysis will take a deeper dive into the workings of the executable. During static analysis we gathered this was a potential IRC botnet, therefore we would expect to see some sort of network traffic in relation to this and will be using Wireshark to trace packets sent back and forth. This section is conducted within a secure environment.

### Wireshark

#### Frame analysis

A screenshot of a computer

Description automatically generated

Figure 8 - Traffic sent after running .exe

After running the executable Wireshark detected both inbound and outbound TCP/IRC traffic between the Equation Group address and our machine requesting a password (PASS), username (USER) and nickname (NICK) showing a logon event, this is displayed in the green box of Figure 6. After this in the red box, a channel join.

A screenshot of a computer

Description automatically generated

Figure 9 - USER command and username bot10601 detected in Wireshark

A screenshot of a computer

Description automatically generated

Figure 10 - PASS command and password fancybear detected in Wireshark

Taking a deeper look at these frames, we can see the username and password which we can use to log into the IRC server as a bot. We wouldn’t need the nickname as that is simply a display name and does not provide any authentication. Figure 7 shows us the user is “bot” followed by a random selection of numbers and Figure 8 shows the password is “fancybear”.

#### HTTP Export List

A screenshot of a computer

Description automatically generated

Figure 11 - HTTP exports after running malware

A screenshot of a computer program

Description automatically generated

Figure 12 - Frame information of identified HTTP export

A screenshot of a computer

Description automatically generated

Figure 13 - #malfor-woods text can also be seen on the C2 link confirming this is a C2 channel

Reviewing HTTP exports Figure 9, shows a singular export to the IP address identified during static analysis and reviewing the frame for this, Figure 10, further confirms that the host is operating on OSX. A cnc.php link can be seen under ‘Request URI’ of Figure 10, cnc commonly meaning command and control which would coincide with our findings of there being an IRC command and control server.

#### TCP Stream

A screenshot of a computer screen

Description automatically generated

Figure 14 - TCP stream displaying IRC connection to server and #malfor-woods channel

After re-executing the malware and again tracking this in Wireshark to see if there were any further changes, following the TCP stream, Figure 12, to display the full conversation between the malware and the command and control server does not show any significant changes.

However, each time the malware is executed a new Bot name is assigned, meaning this is designed to be distributed to multiple systems; the same user cannot log into an IRC chat at the same time. In Figure 12 we can also see that the bot is getting “You are not channel operator” suggesting it does not have permission to run a command which is expected.

Here we can also see a potential author “Julian Murphy” and the origins of the malware which is part of a University of Portsmouth malware server.

# Analysis Conclusion

After analysing the executable provided, it is clearly designed to be distributed and function as a bot which can be controlled via IRC. After executing the malware, the program hides in the background logging into the server and channel awaiting a response from the command and control server. Once the IRC operator sends a command, for example “EXEC IEEXPLORER.exe” this will then conduct the command on each device infected with the malware.

Looking over the program, it seems the only resemblance to an author would be that displayed in the contact information “julian.murphy@port.ac.uk”.

# Accessing the Server

This section will provide an overview of using the information gathered in previous steps to gain access to the server and channel. Using Hexchat, and client which can be used to interact with IRC servers and their channels.

A screenshot of a computer

Description automatically generated

Figure 15 - Connecting to server via Hexchat using credentials found in Wireshark

As shown in Figure 13, I installed Hexchat on the test machine and entered the username and password collected from the previous steps. I purposefully left the channel key empty here as I wanted to test if there were any difference between auto-joining versus joining manually.

A screen shot of a computer

Description automatically generated

Figure 16 - Successful log in to server, channel requires separate password

After connecting, the server outputs the exact same MOTD and information, Figure 14, that can be seen in Figure 12.

A white arrow pointing to a black background

Description automatically generated

Figure 17 - Reviewing strings, a section for joining a channel can be found containing the password

A screenshot of a computer program

Description automatically generated

Figure 18 - Wireshark frame displaying channel connection and password

Performing some further basic static and dynamic analysis, we can see that the password previously found in the TCP stream within Wireshark also seen in Figure 16, can be retrieved using strings, Figure 15.

A screenshot of a computer

Description automatically generated

Figure 19 - Using the password 'richmond' provides access to #malfor-woods

With the password which can be found in Wireshark, Figures 14,16, during dynamic analysis and in strings, Figure 15, we were able to gain access to the botnet server channel “#malfor-woods”. From here we can review some commands to control the botnet which can be found looking at both Strings and IDA.

# Debugging & Further Data Gathering

Using IDA Freeware we carried out debugging on the malware file, running through the executable step by step and analysing the file further to find methods of and how this could exactly be executed.

## Anti-Debugging Technique

A computer error message

Description automatically generated

Figure 20 - First window generated by malware

A screenshot of a computer error

Description automatically generated

Figure 21 - Ant-Debugging window generated by malware

A screenshot of a computer code

Description automatically generated

Figure 22 - Code block for first two forms

A close-up of a computer screen

Description automatically generated

Figure 23 - Code block for exiting debugging

## C2 Commands

A screenshot of a computer program

Description automatically generated

Figure 24 - Commands available

After beginning the debugging, the first window is generated by the malware, Figure 18, this contains a warning and asks for confirmation to continue. This is then followed by an Anti-debug window, Figure 19, to prevent the debugging process, after “OK” is selected debugging automatically exits.

These are shown further in depth via the code view within Figure 20. It is here that we can see the ‘jmp’ reference to “locret\_402B82”. Figure 21 displays this return instruction and shows the purpose to end debugging.

# Recommendations for malware remediation and future prevention

## Remediation

Beginning with the remediation of the executable, any affected systems within Gazprom’s internal infrastructure should be isolated and have their network connection terminated. This is to prevent any further infection via the botnet/re-infection.

#### Isolation

Once these systems have been isolated, the executable process should also be terminated, this will close the malware and its connection the command and control server, removing the ability for the owner to send commands.

### Removal

Next the removal of the malicious files, i.e., malfor-cw-sample.exe should be removed from all systems, affected or unaffected.

#### Review

Finally, review currently affected systems ensuring the malware has been sufficiently removed and these can be released from isolation.

### Future Prevention

As this data leak was caused by an insider of the estate for future prevention, we need to look at two areas, staff and hardware/software:

#### Update

Firstly, any operating systems and software should be kept up to date, scanning such as Tenable’s Nessus can be implemented to automate these vulnerability assessments and find anything which is out of date or vulnerable and needs patching against new CVE’s. Ensuring systems are up to date and patches against any newly found vulnerabilities will greatly reduce the risk of threat actors gaining access to and exfiltrating data.

#### Endpoint Security

##### XDR/SIEM Implementation

Secondly, Gazprom should also consider introducing if not already implemented an XDR solution or at the very least a SIEM which will produce alerts based on values found during collection of network logs then any system activity such as the executed commands will be detected and flagged as malicious as they will not have come from that user.

##### Firewall & Device Security

Gazprom should also review the effectiveness of their current endpoint security on their firewalls and user devices. Firewalls should be setup with sufficient rules to detect and prevent any malicious traffic by controlling what inbound/outbound traffic is denied or allowed throughout the network.

Creating a strong endpoint security infrastructure both through implementation of threat detection and good security practices on endpoints will greatly increase the likelihood of an attacker being alerted on if they get into the network and reduce the risk of attackers gaining access to core IP.

##### Staff training

As this was a rogue employee within the estate, it’s of utmost importance to Gazprom that they provide staff training to employees within the organization to highlight the key factors and devastating outcomes of breaking data security, not only for the organization, but the user themselves. Reinforcement of security policies, identifying suspicious activities and reporting concerns via security awareness training is important to ensure best security practices throughout the business (ADJACENT DIGITAL POLITICS LTD, 2023)

Contents

[Summary 2](#_Toc152928598)

[Introduction 3](#_Toc152928599)

[Malware Analysis 4](#_Toc152928600)

[Static analysis 4](#_Toc152928601)

[Strings 4](#_Toc152928602)

[PEiD 5](#_Toc152928603)

[PEView 6](#_Toc152928604)

[VirusTotal 6](#_Toc152928605)

[Dynamic analysis 8](#_Toc152928606)

[Wireshark 8](#_Toc152928607)

[Frame analysis 8](#_Toc152928608)

[HTTP Export List 9](#_Toc152928609)

[TCP Stream 10](#_Toc152928610)

[Analysis Conclusion 11](#_Toc152928611)

[Accessing the Server 12](#_Toc152928612)

[Debugging & Further Data Gathering 14](#_Toc152928613)

[Anti-Debugging Technique 14](#_Toc152928614)

[C2 Commands 15](#_Toc152928615)

[Recommendations for malware remediation and future prevention 16](#_Toc152928616)

[Remediation 16](#_Toc152928617)

[Isolation 16](#_Toc152928618)

[*Removal* 16](#_Toc152928619)

[Review 16](#_Toc152928620)

[Future Prevention 16](#_Toc152928621)

[Update 16](#_Toc152928622)

[Endpoint Security 16](#_Toc152928623)

[XDR/SIEM Implementation 16](#_Toc152928624)

[Firewall & Device Security 16](#_Toc152928625)

[Staff training 17](#_Toc152928626)

Checkpoint Research Team. (2015). *Intelligence Report: Equation Group - Check Point Blog*. https://blog.checkpoint.com/security/intelligence-report-equation-group/

Wireshark. (2020, August 11). *IRC*. https://wiki.wireshark.org/IRC.md

[Figure 1 - Strings output identifying IP address, operating system and ports 80 & 6667 4](#_Toc152752498)

[Figure 2 - Equation Group website linked to discovered IP 5](#_Toc152752499)

[Figure 3 - PEiD output displaying unpacked .exe file 5](#_Toc152752500)

[Figure 4 – List of imports 6](#_Toc152752501)

[Figure 5 - File header information 6](#_Toc152752502)

[Figure 6 - Traffic sent after running .exe 8](#_Toc152752503)

[Figure 7 - USER command and username bot10601 detected in wireshark 8](#_Toc152752504)

[Figure 8 - PASS command and password fancybear detected in wireshark 8](#_Toc152752505)

[Figure 9 - HTTP exports after running malware 9](#_Toc152752506)

[Figure 10 - Frame information of identified HTTP export 9](#_Toc152752507)

[Figure 11 - #malfor-woods text can also be seen on the C2 link confirming this is a C2 channel 9](#_Toc152752508)

[Figure 12 - TCP stream displaying IRC connection to server and #malfor-woods channel 10](#_Toc152752509)

[Figure 13 - Connecting to server via hexchat using credentials found in wireshark 12](#_Toc152752510)

[Figure 14 - Successful log in to server, channel requires seperate password 12](#_Toc152752511)

[Figure 15 - Reviewing strings, a section for joining a channel can be found containing the password 13](#_Toc152752512)

[Figure 16 - Wireshark frame displaying channel connection and password 13](#_Toc152752513)

[Figure 17 - Using the password 'richmond' provides access to #malfor-woods 13](#_Toc152752514)

[Figure 18 - First window generated by malware 14](#_Toc152752515)

[Figure 19 - Ant-Debugging window generated by malware 14](#_Toc152752516)

[Figure 20 - Code block for first two forms 14](#_Toc152752517)

[Figure 21 - Code block for exiting debugging 14](#_Toc152752518)

[Figure 22 - Commands available 15](#_Toc152752519)

Table 1 – Discovered Items

|  |  |
| --- | --- |
| Item | Significance |
| Username | botxxxxx |
| Channel Name | Malfor-woods |
| Password | Fancybear |
| Password | Richmond |
| IP Address | 167.99.88.222 |
| Port | 6667 |
| Malware Attack | Internet Relay Chat Botnet |
| Author | Julian Murphy |
| Origin | University of Portsmouth |
| Commands | ‘EXEC’  ‘OPEN’  ‘PING’  ‘!’  ‘MSPAINT’  ‘SHUTDOWN’  ‘HELLO’  ‘IDENT’ |