Malware Analysis Project

Disclosure:

We're always looking to improve our homework assignments. If you see any errors, whether they are grammatical or technical, please email the TAs to report them. If anything is unclearly stated, please contact the TAs.

Purpose:

The purpose of this assignment is to have you gain experience with running malware through an analysis engine and perform investigations on a malware's behaviors. You will be running malware through an analysis engine called Cuckoo (http://www.cuckoosandbox.org/). You will learn how to use Cuckoo and how to run malware ***safely***. There is no sense in studying malicious behavior if you're going to contribute to the problem.

Grading:

Your score for this project will be out of 100 points. Phases I (40 points) and II (30 points) are mandatory. You must choose whether to complete Phase III (30 points) or Phase IV (30 points). Partial credit will be given where necessary for all Phases. For your solution, if you complete Phase III, remove Phase IV's solution template (mentioned below). That is, completely remove any mention of it from your solution. If you complete Phase IV, remove Phase III's solution template similarly. No extra credit will be given if you complete both Phases III and IV. Please do not complete and submit both in case you do badly on one of them with the hopes that we will offset the grading in your favor. If you submit both, we will only grade the first Phase our automatic grading script sees.

Setup:

1. Install VirtualBox
    - Delegate at least 4GB (4096 MB) of RAM
        i. This should be enough space so that you can run the malware experiments with reasonable speed.
    - Delegate at least 40GB of Hard Drive space (dynamically sized)
        i. We chose this arbitrary value so that we guarantee that the installation of Ubuntu plus the installation of Cuckoo's dependencies plus the installation of Cuckoo and the malware will not fill up the virtual machine's hard drive. We also chose this amount of space so that you could continue to download and perform more experiments on malware in the future.
2. Install Ubuntu 14.04 LTS 64-bit (http://releases.ubuntu.com/14.04.3/ubuntu-14.04.3-desktop-amd64.iso)
3. Install Guest additions
    - In VirtualBox menu: Devices->Insert Guest Additions CD image
    - Follow the instructions that appear on the Ubuntu screen
    - You will need to be able to resize your window to properly read the malware reports generated by Cuckoo.
4. Restart Ubuntu
5. Download the project folder **to the VM Ubuntu desktop**
    - cd ~/Desktop
    - wget http://evan.gtisc.gatech.edu/copy-to-desktop.zip

6. If this download is too slow, you can download the zip file first to your computer, and then share the folder with VirtualBox and copy the zip file there.
    - https://help.ubuntu.com/community/VirtualBox/SharedFolders
7. Download malware and unzip the folder **to the VM Ubuntu desktop**
    - cd ~/Desktop
    - wget http://evan.gtisc.gatech.edu/malware.zip
8. Unzip both zip files from steps 5 and 6
9. Copy contents of copy-to-desktop **to the VM Ubuntu desktop**
    - cd ~/Desktop
    - cp -r ./copy-to-desktop/* .
    - rmdir ./copy-to-desktop
10. Follow remaining instructions in the README to setup Cuckoo and configure the analysis environment: the Ubuntu Virtual Machine (VM)

Important:

1. If you shut down (or restart) the VM, you must execute `~/Desktop/config.sh` inside the Ubuntu VM **after** the VM has rebooted.

2. There are two key folders you will be using for this entire project: "cuckoo" and "malware." Both folders are located on the Desktop of your user. The folder "cuckoo" contains the Cuckoo software and will be responsible for submitting and analyzing our malware. The folder "malware" contains malware for each Phase of this project.

3. Refrain from updating **any** software inside the virtual machine. We have configured it with precise versions and if one is updated/upgraded the system **will** break.

4. A homework solutions template and example is located with this assignment in case the descriptions for how you should answer each question is unclear. Conform to the template's format. Do not create any kind of formatted document (txt, docx, pdf, etc.) Copy-and-paste and use this template as it is provided. If you do not, our automated grader will not be able to parse your answers correctly and you may receive a 0% on this assignment. Since you are in graduate school at a top-10 school in the country, we expect that you are more than capable of following this strict guideline. Please name your solutions file as your Georgia Tech user ID (e.g., jsmith23, jdoe4, etc.).

5. Do not run the malware for more than suggested (10 minutes). After that point in time, things can get dangerous, as I have not run these malware for long durations in time. So I don't know what the malware will do after running it for 10 minutes.

Phase I [40 points]: Learn how to run malware using Cuckoo and get familiar with reading its reports

1. Open Terminal
2. Open two tabs in Terminal
3. In one tab, start Cuckoo:
    a. cd Desktop/cuckoo
    b. ./cuckoo.py
4. In the second tab, submit your pieces of malware

    a. cd Desktop/cuckoo
    b. python ./utils/submit.py --platform windows --package exe --machine WindowsXPSP3 --timeout 600 ../malware/PhaseI
5. This particular command tells Cuckoo to run all malware contained in the folder "../malware/PhaseI" on the virtual machine called "WindowsXPSP3" for maximum time of 600 seconds (10 minutes). Note that it may take Cuckoo a long time to analyze the data generated by the malware sample, so running all of the malware may take somewhere between 1 and 2 hours total.
6. To check up on its progress, look at the output generated in the first tab of Terminal (where Cuckoo is running). You will see some warnings every now and again. Don't worry, these warnings are by design. However, you should **not** see any errors.
7. Wait for Cuckoo to finish analyzing all 7 pieces of malware. The terminal will say "Task #7: analysis report completed".
8. In the second tab in Terminal, run: `python ./utils/web.py`
9. Open Firefox and navigate to the URL "localhost:8080"
10. On this webpage, click "Browse" at the **top** of the page and you will see the results of Cuckoo's analysis organized nicely for you. Expand the Ubuntu VM's screen size and Firefox's window size within the Ubuntu VM so that you can see the "Browse" link.
11. Use these reports to answer the questions for Phase I.

You will note that not all malware may actually run for the full 10 minutes. This can be for various reasons: the Command & Control (C&C) server decides that the malware is being analyzed and does not want it to be run anymore, the malware decides to exit because it has completed its nefarious task and does not need to run any more, etc.

Make sure to copy the data from the reports AS IS. Regardless if the data appears redundant, strangely capitalized or punctuated, copy-and-paste it exactly. Non-uniformity in the Cuckoo reports adds an additional layer of integrity check for the answers to this assignment.

<u>Questions:</u>

Malware1:

1. What is the SHA-256 hash of the executable?
2. What files are listed under "Behavior Summary"? These files have been manipulated by the malware (created/modified/etc.).
3. What registry keys does it manipulate under "Behavior Summary"?
4. Does it create any mutexes ("yes" or "no" without quotations)?
5. What **Library(s)** does it import?

Malware2:

1. What is the SHA-256 hash of the executable?
2. What files were **dropped**?
3. What registry keys does it manipulate under "Behavior Summary"?
4. What mutexes does it create?
5. What **Library(s)** does it import?
6. What domain does it perform a DNS lookup on?

7. The malware performs an HTTP ____ Request to http://total-updates.com/windebug/updcheck.php.

Malware3:

1. What is the SHA-256 hash of the executable?
2. What mutexes does it create?
3. What **Library(s)** does it import?
4. Under the "Behavior Summary" section, what two processes are mentioned in the report?

Malware4:

1. What is the SHA-256 hash of the executable?
2. What **Library(s)** does it import?
3. How many HTTP requests does it make (as a digit: e.g., 4, 8, 12, etc.)?
4. Besides DNS and HTTP requests, the malware also performs an ____ Request.
5. Under the "Behavior Summary" section, how many processes are mentioned in the report (as a digit: e.g., 1, 2, 3, etc.)?
6. What does VirusTotal's "ESET-NOD32" classify this executable as (i.e., its "Result")? Copy this classification in its entirety.

Malware5:

1. What is the SHA-256 hash of the executable?
2. How many mutexes does it create (as a digit: e.g., 1, 2, 3, etc.)?
3. What **Library(s)** does it import?
4. How many bytes is the file size of autoexec.bat?
5. Under the "Behavior Summary" section, what two processes are mentioned in the report?

Malware6:

1. What is the SHA-256 hash of the executable?
2. What **Library(s)** does it import?
3. How many files does it **drop** (as a digit)?
4. How many registry keys does it interact with (as a digit)?
5. How many mutexes does it create (as a digit)?

Malware7:

1. What is the SHA-256 hash of the executable?
2. What **Library(s)** does it import?
3. What domains does it perform a DNS lookup on?
4. The malware performs an HTTP ____ Request.
5. Based on the screenshots taken during execution, what type of malware is this most likely? Include only the letter of your answer. No other notation. E.g., a or b or c alone.
    a. Adware
    b. Botnet
    c. Ransomware

Phase II [30 points]: Learn how identify different behaviors in malware

Now that you have a bit more experience with running malware, now it is your job to investigate and label some of the more sophisticated malware's behaviors from Phase I. Use the Cuckoo reports from Phase I label the malware's behavior. Note that malware can share the same behaviors. So initially you should assume that each malware we question you about below has every behavior listed. It's your job to determine if that assumption is actually true.

Hint: Look at the system call sequence and determine what malware is doing. Note that each Cuckoo report may contain multiple processes with many different system call sequences. If **any** of the behaviors are seen in **any process** in the report, then that malware has that behavior.

More helpful hints are written inline with each choice below.

Choose your answers from one or more of the following choices:

a. Tries to unhook Windows functions monitored by Cuckoo
   - Hint: Cuckoo will generate an anomaly message ("_anomaly_" API name) containing the phrase "Function hook was modified!"

b. Steals private information from Internet Explorer
   - Hint: malware will inspect "…\\Temporary\\ Internet\ Files\\Content.IE5\\index.dat" or "…\\History\\History.IE5\\index.dat"

c. Installs itself for autorun at Windows startup
   - Hint: malware could modify the register "HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Run" or "…\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon" among other registers.

d. Detects VirtualBox using ACPI tricks
   - Hint: malware checks to see if the register "HARDWARE\\ACPI\\DSDT\\VBOX__" exists

e. Checks for the presence of known devices from debuggers and forensics tools
   - Hint: malware checks for files "SICE", "SIWVID", "SIWDEBUG", "REGVXG", "FILEVXG", "REGSYS", "FILEM", "TRW", "ICEXT", and/or "NTICE"

f. Detects the presence of Wine emulator
   - Hint: malware checks to see if register "HKEY_CURRENT_USER\\Software\\Wine" exists

g. Collects information to fingerprint the system (MachineGuid, DigitalProductId, SystemBiosDate)
   - Hint: malware will query for a registry value of either MachineGuid, DigitalProductId, or SystemBiosDate. If either of these values are queried 3 or more times (together or separately), then this rule holds.

h. Creates Zeus (Banking Trojan) P2P mutexes
   - Hint: malware will use any one or more of the following mutexes:
     i. Global\{DF5816D6-312D-D5EE-C14D-3B00F2DD1812}
     ii. Global\{41E756AD-7156-4B51-C14D-3B00F2DD1812}
     iii. Local\{41E756AD-7156-4B51-C14D-3B00F2DD1812}
     iv. Global\{41E756AC-7157-4B51-C14D-3B00F2DD1812}
     v. Local\{41E756AC-7157-4B51-C14D-3B00F2DD1812}

vi. Global\{2001E0BE-C745-2AB7-C14D-3B00F2DD1812}
vii. Local\{2001E0BE-C745-2AB7-C14D-3B00F2DD1812}
viii. Global\{2001E0B9-C742-2AB7-C14D-3B00F2DD1812}
ix. Local\{2001E0B9-C742-2AB7-C14D-3B00F2DD1812}
x. Global\{FAEB01BF-2644-F05D-C14D-3B00F2DD1812}
xi. Local\{FAEB01BF-2644-F05D-C14D-3B00F2DD1812}
xii. Global\{4A390394-246F-408F-C14D-3B00F2DD1812}
xiii. Local\{6041339D-1466-6AF7-C14D-3B00F2DD1812}

i. Operates on local firewall's policies and settings
   - Hint: malware will operate on registry "HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\SharedAccess\\Parameters\\FirewallPolicy\\StandardProfile"

j. Creates a Windows hook that monitors keyboard input (keylogger)
   - Hint: malware will call "SetWindowsHookExA" or "SetWindowsHookExW" with HookIdentifier as either "2" or "13" and ThreadId as "0".

k. A process attempted to delay the analysis task
   - Hint: malware uses Windows function "NtDelayExecution"

Questions:

List behaviors for:
Malware1, Malware4, Malware5, and Malware6

Phase III [30 points]: Learn how to trigger dormant malware behavior

You have been given 3 more pieces of malware in the folder "~/Desktop/malware/PhaseIII". These malware will only activate if a certain condition is true. Your job is to find that particular condition via brute-force techniques. Use the workflow for running Cuckoo in PhaseI for this phase.

Questions:

Malware8:
This malware's activity is triggered on some day in the month of March in the year 2004. Your job is to find what day it executes on. Submit your answer as a digit (e.g., 1, 2, 3, etc.).

Hint: Write a script that submits the malware at every day in the month at **2PM** (because of some offset time bugs with the virtual machine). Brute-force the solution. To find out how to run the malware at a specific time, read Cuckoo's documentation (http://docs.cuckoosandbox.org/en/latest/).

Malware9:
This malware's activity is triggered after a certain amount of time has passed since it was executed. In essence, it delays its activities in order to evade analysis by malware analysis environments that employ fixed-time executions on its malware (which is a majority of malware analysis engines today).

Your job is to find out how many milliseconds the malware delays its **initial** activities. Submit your answer as a number (e.g., 1234, 234532, 352373, etc.)

Hint: Look at the system call sequence. What's the **first** attempt to delay execution at the start of the program?

Malware10 and 11:

Some malware won't show any behavior if they don't have certain filename(s). This is also an effort to evade detection, as malware researchers usually rename the malware for various reasons. Run malware10 and malware11 for 180 seconds each. Compare the Cuckoo reports:

1. Which malware has the proper trigger name (malware10.exe or malware11.exe)? (i.e., which executable shows more behavior?)
2. What extra file is **dropped** by the properly named malware (besides the malware itself)?
3. What two domains are contacted by the properly named malware?
4. Looking at the screenshots generated by the **improperly** named malware, what is the message that is displayed on the screen?

Phase IV [30 points]: Learn how malware is run safely

There are various ways to run malware in a safe environment. Running the malware inside of a virtual machine is a good start. That pretty much covers the system-side of things, but what about the network-side? We can use firewall rules in order to prohibit the malware from spreading throughout our network, sending spam, etc. We can even rate-limit the network connection (http://askubuntu.com/questions/20872/how-do-i-limit-internet-bandwidth) so that if a DDoS attack is used by our malware, we won't cause too much harm to the rest of the Internet.

Your job is to read and interpret the firewall rules we've employed on our malware analysis system (Ubuntu).

Execute the following command in a Ubuntu Terminal: `sudo iptables-save`

Read these rules outputted to the screen, read iptables documentation on the Internet, and answer the questions below.

Hint: Here is some good iptables material to read, as iptables can be difficult to read/understand and even more difficult to write properly.

- https://en.wikipedia.org/wiki/Iptables
- https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-iptables.html
- https://wiki.debian.org/iptables
- http://www.howtogeek.com/177621/the-beginners-guide-to-iptables-the-linux-firewall/

Questions:

1. What IP address **CIDRs** are not allowed to be communicated with by our malware?

Hint: Cuckoo uses the IP addresses 192.168.56.1 and 192.168.56.101 to connect the malware to the Internet.

2. What IP address is all email traffic forwarded to?
3. Do the rules accept SSH connections? ("yes" or "no" without quotes)
4. Do the rules allow the analysis machine to be ping'ed? ("yes" or "no" without quotes)
5. Why do the rules drop outbound connections to ports 135, 139, and 445? (Pick your answer from the choices below)

   Hint: Google these port numbers. They are used by Windows malware.
   Hint2: http://www.berghel.net/col-edit/digital_village/dec-05/dv_12-05.php

   a. They are primarily used by malware to send spam.
   b. They are primarily used by malware to propagate.
   c. They are primarily used by malware to launch DoS attacks.
   d. They are primarily used by malware to detect themselves being analyzed.

Reflection:

Well cool. Now you've got some experience under your belt with analyzing malware. For this project you used an analysis tool that does the analysis for you. In practice, entire teams of people are devoted to work on a single malware executable at a time to debug it, disassemble it and study its binary, perform static analysis techniques, dynamic analysis techniques, and other techniques not included in Cuckoo to thoroughly understand what the malware is doing. Luckily for you, it takes an enormous amount of time to perfect/improve the skills of malware analysis, so we didn't require it for this project. However, to give you a scale of how much work this all takes, consider that antivirus companies receive somewhere on the order of 250,000 samples of (possible) malware. We had you analyze 10 binaries. Imagine the types of systems needed to handle this amount of malware and study it thoroughly enough for that day, because the next day they're going to receive 250,000 new samples. If a malware analysis engine is unable to analyze a piece of malware within a day, they've already lost to malware authors. Also consider that not all of the 250,000 samples will be malicious. According to [1], as many as 3-30% may be benign!

Another way to look at the size issue of malware analysis, consider this paper [2] where the authors discovered that major notorious malware samples had actually been submitted months, even years before the malware was detected and classified as malicious in the wild.

Remember, analyzing malware is a delicate and potentially dangerous act. Please be cautious and use good practices when analyzing malware in the future. If you let malware run for too long, you may be contributing to the problem and may be contacted by the FBI (and other authorities) as a result of this unintentional malicious contribution. At Georgia Tech, researchers, professors, and graduate students are able to analyze malware in controlled environments and have been given permission by the research community to perform these analyses long-term. We make efforts to contact the general research community and Georgia Tech's OIT Department to inform them that we are running malware so they won't raise red flags if they detect malicious activity coming out of our analysis servers.

References:

[1] Rossow, Christian, Christian J. Dietrich, Chris Grier, Christian Kreibich, Vern Paxson, Norbert Pohlmann, Herbert Bos, and Maarten Van Steen. "Prudent Practices for Designing Malware Experiments: Status Quo and Outlook." In *Security and Privacy (SP), 2012 IEEE Symposium on*, 65–79. IEEE, 2012. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6234405.

[2] Graziano, Mariano, Davide Canali, Leyla Bilge, Andrea Lanzi, and Davide Balzarotti. "Needles in a Haystack: Mining Information from Public Dynamic Analysis Sandboxes for Malware Intelligence." In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association. Accessed September 23, 2015. https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-graziano.pdf.

For your curious mind:

In PhaseI, all of these malware are unmodified real-world samples. Some of them, you'll notice, aren't even detected/classified as malicious by all or a majority of antivirus companies. This is because either (1) the samples are still too new and are still being studied and classified by antivirus companies and malware research facilities or (2) the samples are classified differently between antivirus companies. Truly there is disagreement in the malware research community as to what exactly classifies malicious activity. For example, some say that adware is a form of malware, while others do not. Can you think of arguments for either side? Let's take this kind of thinking one step further. As a thought experiment, ask yourself this: If a piece of software has malicious code contained within it, but the malicious code is never executed when it is run, is/should that software be considered malicious?

In PhaseII, we modified real-world malware source code to create triggers seen in other real-world malware. We designed it this way because it's nicer if we can control and determine the malware's behavior by modifying its source.

Be careful if you ever get your hands on malware source code. We always make sure we read and fully understand malware source code before we compile and run it. Remember, safety is the number one priority in malware analysis.

If you're interested in reading more information about researching malware, we recommend you read "The Art of Computer Virus Research and Defense" by Peter Szor. It's known in the research community as a must-read for those interested in studying malware.