Visualizing Malware Propagation Across Local Area Networks

Jacob J Williams

Abstract

1. Introduction

Malware is perhaps one of the greatest threats in modern day IT, it is constantly evolving both terms of who it targets and what it exploits. Due to the evolutionary nature of malware modern security has turned into a cat and mouse game, security analysts and malware writers always vying to overtake the other by finding new ways to protect or exploit systems. Unfortunately it is common in the industry for the malware writers always to be numerous steps ahead, with those who work in security being forced to operate reactively to malware, this is why the security industry needs more who are willing to study malware in order to prevent it.

How malware propagates is key to understand how malware operates, external propagation; that being how malware bridges the air gap between networks is commonly studied and is fairly established in various forms of study, mathematically and visually. How malware propagates within a local area network however is studied rather one sidedly, leaning towards the mathematical side. These studies are extremely comprehensive and explore numerous methodologies of formulating how malware propagates, but it requires some complex knowledge and understanding of mathematical models and theories and as a result could be seen as being rather unfriendly to those wishing to involve themselves with Cyber Security from other backgrounds. Therefore the goal of this study to to attempt to visualise malware propagation using variables and characteristics from machines on local area networks and produce a more easily interpretable way of showing the nature of propagation.

1.1 Aims and Objectives

i. To create a visualization tool for examining the propagation of threats across a local area network with an initial focus on the subset of Ransomware and then proceeding to the superset of malware afterwards.
ii. To create clear and interpretable visualisations of propagations.
    a) To inform other researchers of the details of the propagation in order to assist in the further development of anti-malware capabilities.
    b) To inform those in positions whose decisions can affect the security of a network, and where they should be deploying network security features.
    c) To aid students in the study of malware behaviour and its impact across networks.
iii. To identify key characteristics of malware propagation that can be used in the response to an attack.

The second aim is the crux of this research, as aforementioned there already exist numerous studies into malware propagation but in the form of mathematical formulas and graphs that are only interpretable by those with prior experience or study in the field of Cyber Security or mathematics. Therefore this research hope to contribute to the study of malware prevention by creating resources for those who either just beginning their studies or those less knowledgeable in the field of Security and wish to utilise the results of such studies in configuring their network security.

After the creation of the visualizations, this research will seek to answer several questions both in relation to individual malware and in a generalisation

i. Is malware propagation in certain malware deterministic?
ii. Does IP address locality affect the nature of propagation?
iii. How does network structure affect propagation?

2. Related Literature

As aforementioned, the vast majority of research into malware propagation is mathematical in nature, but this does not make it irrelevant to this study, as researching what we already know of malware propagation will allow us to create hypotheses and possibly support not only our own findings but others findings as well. *Yu et al (2015)* explored the prospect of malware propagation by applying epidemic theory and creating a two-layer epidemic model. Yu et al (2015) focus on epidemic modelling because it is *"more focused on the number of compromised hosts and their distributions"* compared to another method called Control System Theory, which Yu et al (2015) claims are more inclined to *"try to detect and contain the spread of malware"*. This distinction is interesting as it categorises the two types of studies seen in the field of malware propagation, those who observe the spread of malware to define its nature are studying it Epidemiologically, and those study it to configure systems to prevent propagation are studying it under Control System Theory. With this distinction in mind, one could claim that this study will be observing Epidemiologically. Yu et als (2015) results are interesting, providing evidence that propagation in large scale networks possess three different discernible stages, *"Exponential distribution in its early stage"*, *"power law distribution with short tail in late stage"*, and *"power law distribution in its final stage"*. This aligns with

how one might imagine malware propagation, with it at first infecting everything it can and once there is nothing more to infect the rate of infection quite obviously decreases exceedingly quickly. Yu et al (2015) explores the rate at which malware propagates to great depths, but like many mathematical models doesn't explore the nature of propagation, such as whether the way it happens is deterministic or if IP address locality matters.

Guillen and Rey (2018) researched what is in concept an extension of Yu et als (2015) work. Guillen and Rey (2018) acknowledge the existence of *"compartment devices"* which are devices that *"cannot be directly targeted by malware, but can be used to propagate it".* It presents an interesting figurative, the devices that propagate without being infected do not contribute towards the infection rate themselves but they greatly increase the general infection rate. This presented an interesting line of though, that if you were to visualise malware propagation solely through signatures or system resources then these compartment devices would seemingly be unaffected but devices near to them would seemingly spontaneously become infected. If one was to attempt to study malware behaviour in networks that frequently possess these compartment devices then network trace analysis would be key. Hosseni and Azgomi (2016) formulated a model for representing malware propagation based on a rumour spreading model. Identifying five different types of machines (nodes) that perform different roles in the propagation of malware, these being *"susceptible, exposed, infectious, recovered, vaccinated".* This study lends well to classifying the different states a machine in an active attack on a LAN may take and may be useful to consider during visualization.

Zhuo and Nadjin (2012) conducted a study with the goal of visualizing malware network traces, in doing so they created the tool MalwareVis. MalwareVis is able to pull heterogeneous attributes like protocol, IP address, and more from the network trace log. Network traces are a key part of categorizing malware since often the malware has to communicate with an outside command and control server in order to deploy, receive orders, or exfiltrate data. Analysing network traces is often a long and arduous job in malware analysis because the logs contain large volumes of data, as a result, the study focuses more on specific protocols such as TCP and DNS. Whilst the study is interesting and attempts to visualize a typically complex part of malware analysis, it still produces a visualisation that cannot be understood without in-depth study of the formulas it is built upon. To the average researcher this will prove no trouble, but perhaps to those getting into the field it will prove much more difficult and perhaps yield very little to their understanding. Gove and Deason (2018) also explore the concept of identifying malware through network activity. The target of their study is the period network activity of malware, which they acknowledge as being a difficult task due to it being *"easily drowned out by non-malicious network activity".* Gove and Deason (2018) utilise a novel algorithm based on Discrete Fourier Transforms, and they pair the output of this with aggregation summary tables that will inform users which detection are worth investigating and which are not. This study is highly complex but also very interesting, as it presents a method of analysing data that is usually very difficult to obtain. The downside is the method in which the data is collected is similarly as complex, whilst network traces might be an excellent way to study propagation in particular it is appearing that such a method may not be conceivable within the time frame of our study.

The study of malware detection is useful to this study due to the key overlap of malware characteristics. If certain characteristics are better at identifying malware then they will most certainly be better for visualising malware, therefore much study was conducted into the detection of malware and the identification of their characteristics. Rhode et al (2018) conducted study where they used Recurrent Neural Networks to predict malware, in the way of investigating *"whether or not an executable is malicious based on a short snapshot of behavioural data"*. The recurrent neural network works using hyper-parameters, but what these hyper-parameters work with is what is interesting. Rhode et al (2018) compile minimum and maximum values of inputs to identify the benign or malicious nature of samples. These input values are things such as *"total processes, max process ID, cpu user (%), cpu system (%)"* and more, in short they identify malicious activity when the executable exhibits resource usage past a certain threshold which is typical of other malicious samples. This study is useful as it presents the concept of using system resources and set thresholds to visualise malware, one could visualise all resource usage but have the visualisation be different for those who pass the said threshold, perhaps in a green-amber-red fashion.

Mills et al also used a form of machine learning in order to detect malware. Mills et al used Random Forest to create NODENS, a lightweight malware detection platform that can be deployed on cheaper hardware. After *"removing duplicate or unnecessary attributes"* a total of twenty-two features were identified for the classification model. All these characteristics were once again related to system resource usage, such as *"total processor time, user processor time, Non-paged / paged memory sizes"* and more. This reinforces the proposed method of visualising propagation, since resource usage is a recurring method of signalling what is and is not malware. Xiao et al (2019) conducted a study with the goal of detecting malware through behaviour graphs. The behaviour graphs were constructed by monitoring and gathering the order in which malware uses system calls, with this you can deem certain orders of system calls benign and others malicious. This study is interesting as well as comprehensive, and provides a good amount of detail in regards to what systems calls malware frequently uses that one could use to get started conducting a similar study. Our study will probably not utilise such a method, but it is worth noting that the system calls are usually in relation to a system resource such as CPU, memory, etc.

Bai et al (2014) used a method of mining format information from malware executables in order to identify characteristics that would make an executable appear malicious. They identified 197 different features and used them to train an algorithm that was able to achieve 99.1% accuracy in the testing environment. Whilst this method of detection is interesting and certainly seems effective, the characteristics it identifies assist in detecting malware before it is able to execute, which is not particularly useful to us when we need the malware to execute and propagate. What is useful however is that some of the characteristics captured seem to relate in some way to how the executable will go about using system resources when executed.

Patel and Tailor (2020) researched a method of not only monitoring a system for ransomware, but also counteracting it. To counteract the ransomware, the system they have produced first identifies the encryption of a protected folder it is explicitly monitoring, once it has it creates a large dummy file for the ransomware to encrypt. Whilst it is spending the time encrypting the file, the system changes its properties to read only so that the ransomware can no longer encrypt the files on the machine. Whilst what they've created is interesting and somewhat unique, one must question the usefulness of it since it doesn't particularly prevent ransomware, but merely gives you an opportunity to prevent it. One must also question the validity of the testing, since the study full well admits that *"we designed the attacks"* and one could argue due to this it doesn't necessarily reflect a real world scenario. Unlike the others, the method this study uses to identify ransomware isn't particularly useful, as the reliability of identifying ransomware based on a monitored folder may prove unreliable when extracting data since the functions used to extract said data may already be blocked by the ransomware before it is counteracted or the preventative measure of locking down the system may create a scenario where we cannot extract the data.

It was also deemed worthwhile to comprehensively research malware analysis methods, particularly in regards to sandboxes, since they will be used to observe the propagation in the first place. The first paper by Afianian et al (2019) is a comprehensive summary of knowledge regarding common methods that malware use to evade detection and analysis. The paper begins begins by acknowledging that modern day defensive and analysis techniques can be *"readily foiled by zero-day fingerprinting techniques or other evasion tactics such as stalling"*. The paper then walks through the different studied methods used to avoid analysis, dividing them into two distinct categories: detection dependent evasion, where the *"goal is to detect its environment to verify if it is a sandbox or not"*; and detection independent evasion, which are methods that do *"not rely on detecting the environment".* This paper as mentioned previously is very comprehensive and therefore very useful for learning how malware avoids sandboxing in a very brief and concise manner, even giving explicit examples of what malware tends to look for in regards to environmental details. Chakkaravarthy et al (2019) take a step back from individual machines and take a look at methods malware uses to avoid detection within the network, such as payload fragmentation, session splicing, and more. All of these details whilst interesting were not our key focus on this report, what was is the fact that the paper includes full details on the configuration of their virtual machines including OS, network interface, processor/core/memory, etc. Interestingly enough, this study seems to use NAT based configuration which would mean the infected virtual network still has connection to the host machine, presenting a possible danger.

3. Methodology

With the objectives in mind the approach to fulfilling them will be linear in nature. There is a defined order to which the objectives must be completed, not out of preference but out of dependency. Because of this how the project will progress is fairly predictable, the following sub section will provide detail and justifications for this.

3.1 Timeline

The first stage of this project will be the creation of the tool that allows us to extract details from an infected machine to later analyse the propagation. This tool must not only extract the various system details but also potentially extract the time in which the system appears to be infected. There are numerous subsections to this stage, both pre and post tool creation. Before we can begin creating the tool we must first research malware analysis methods concerning the creation of sandbox environments in order to assure that the system we are testing on will not allow the malware to propagate outside of the virtual environment. Furthermore, the methods that malware analysts use to trick malware with a weaker context awareness algorithms into thinking it is in fact in a legitimate users system must be explored. On the other side of the aisle, it will be prudent to research which malware exploit vulnerabilities that allow them to escape virtual environments, at first to avoid them, but if time allows possibly to study them specifically.

The creation of the tool will follow, and then be followed itself by the testing phases. At first the tool will be tested on systems that are not under active attack, then proceed to active attack testing once the extraction functionality is deemed acceptable. This section aligns with the first objective.

The next stage will involve the deployment of the tool to extract data from machines that are experiencing an active attack. Development will technically be continuing during this stage as different malware tend to exhibit different characteristics and therefore the tool may have to be adapted to suit best what the current active attack is. This is in alignment with objective three, the for the sake of transparency this project will include all of the tested characteristics, whereas the ones visualised may only be the ones with significant results.

The final stage will be to analyse, visualise, and draw conclusions from the results, keeping to the requirements outline within objective two. Stage two and three will occur multiple times, each run analysing a different malware. How many times this runs is determined by the amount of time surplus.

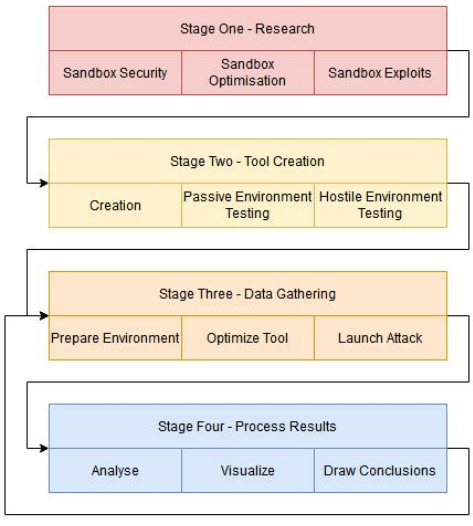

Figure 1 - Methodology, Stage by Stage

3.2 Ethics

There are two acknowledged ethical concerns around this project, one which acts as a requirement and another that must be declared. The requirement being that due to working with legitimate malware strains, the sandbox environment must be as secure as possible because anything less may allow the malware to propagate through the sandbox and into the home/work/school network, there is could pose a threat to any device on said network. Therefore one can consider it an ethical requirement to not only optimise the sandbox environment but also research the malware strains thoroughly for any potential context based exploits that may endanger the external network.

The declaration is that at no point in this project will malware be created, only malware that has already circulated, been studied, and prevented will be studied.

3.3 Tool Creation

Data gathering and analyses formatting ideas:

- Format by malware
  - Include the data gathered, the process of visualization, and analyses based on the malware.
  - Have a general analysis, observations, and conclusions afterwards.
- Format by process.
  - All of the data gathering then all of the visualizing then all of the analyses.

First way would format better, keep all the data relevant to each other together, wont confuse the reader.
Second way is more linear?

References

Afianian, A., Niksefat, S., Sadeghiyan, B. and Baptiste, D. (2019) Malware Dynamic Analysis Evasion Techniques: A Survey. *Acm Computing Surveys* [online]. 52 (6), pp. 126:1-126:28. [Accessed 26 May 2020].

Bai, J., Wang, J. and Zou, G. (2014) A Malware Detection Scheme Based on Mining Format Information. *Scientific World Journal* [online]. 2014 [Accessed 01 June 2020].

Chakkaravarthy, S.S., Sangeetha, D. and Vaidehi, V. (2019) A Survey on Malware Analysis and Mitigation Techniques. *Comouter Science Review* [online]. 32, pp. 1-23. [Accessed 26 May 2020].

Gove, R. and Deason, L. (2018) Visualizing Automatically Detected Periodic Network Activity. *2018 Ieee Symposium on Visualization For Cyber Security* [online]. [Accessed 02 March 2020].

Hosseini, S. and Azgomi, M.A. (2016) A Model For Malware Propagation in Scale-free Networks Based on Rumor Spreading Process. *Computer Networks* [online]. 108, pp. 97-107. [Accessed 01 June 2020].

Mills, A., Spyridopoulos, T. and Legg, P. (No date) Efficient and Interpretable Real-time Malware Detection Using Random Forest. *(No Place)* [online]. [Accessed 16 February 2020].

Patel, A. and Tailor, J. (2020) A Malicious Activity Monitoring Mechanism to Detect and Prevent Ransomware. *Computer Fraud and Security* [online]., pp. 14-19. [Accessed 16 February 2020].

Rhode, M., Burnap, P. and Jones, K. (2018) Early Stage Malware Prediction Using Recurrent Neural Networks. *Computers and Security* [online]. 77, pp. 578-594. [Accessed 15 February 2020].

Xiao, F., Lin, Z., Sun, Y. and Ma, Y. (2019) Malware Detection Based on Deep Learning of Behaviour Graphs. *Mathematical Problems in Engineering* [online]. 2019 [Accessed 01 June 2020].

Yu, S., Gu, G., Barnawi, A., Guo, S. and Stojmenovic, I. (2015) Malware Propagation in Large-scale Networks. *Ieee Transactions on Knowledge and Data Engineering* [online]. 27 (1), pp. 170-179. [Accessed 14 February 2020].

Zhuo, W and Nadjin, Y (2012) MalwareVis: Entity-based Visualization of Malware Network Traces. *Acm International Conference Proceeding Series* [online]., pp. 41-47. [Accessed 15 February 2020].