# Detecting Infected Hosts and Domains

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Detecting Malware in the form of infected hosts and domains inside of a network is no easy task, especially if all one has to work with are log files that comprise a few months of data and total over 1 Terabyte worth of space. In this paper, we discuss the method we used in accomplishing this goal.

## 1 Introduction

Insert intro for detecting malicious hosts/domains...

### 1.1 Method Explanations

Explanation of methods...

### 1.2 Contributions

Contributions (if any)...

## 2 Our Method

The entirety of the data is stored in the form of DNS logs. In terms of a broad overview, our methodology is simply the following: parse the logs, filter the parse result, build a graph, analyze that graph, get a set of domains, look into the interactions involving those domains, and then report those possibly infected domains and hosts.

For parsing, we check for all responses that contain the letter ”A” separated by a space from all other characters or words. The appearance of the letter A, in this form, is a reference to an address record, meaning there has been some kind of interaction between a domain as well as a host. So, we save that record, and transform it into part of the overall final graph, in which each node represents (labelled as an IP-address) either a domain or a host, and the edge between them represents an interaction between that host and domain (the edge contains the actual string of the interaction). Since a host can have multiple interactions with the same domain, the final graph is actually a multi-way-graph, which can contain multiple edges between a domain and host.

Once our final graph has been constructed from all the responses found in the log, we use Degree Centrality to find out the amount of hosts that are connected to (or have contacted) each of the domains. We then look at which of these domains have a low-degree centrality (small amount of connections), then use those domains to aid in the execution of a Belief Propagation algorithm.

Once the Belief Propagation finishes, it will return a sub-network of possibly infected hosts and domains. We can then look into those possibly infected nodes, and check for suspicious activity.

### 2.1 Low Degree Centrality

Many infected domains tend to be ones that are specifically targeted by malicious hosts. These domains are specifically chosen by malicious hosts because they tend to be ones that have a small amount connections.

A small amount of connections means a small amount of requests are processed by the domain, which potentially means less security, less tracking of hosts and their interactions with the domain, and an overall less chance of the malicious hosts being discovered and dealt with for committing malicious actions. Because of this type of low-detection chance environment, domains with low connections could serve as a front for all kinds of malicious behaviour (they could be C&C, also known as Command and Control domains) and thus, are worth looking into.

Determining which domains have a small amount of connections is quite simple- this is where degree centrality becomes useful. We can simply run a degree centrality algorithm on the entire graph and return all the domains that have a low degree centrality (small number of connections). However, a small degree centrality can also simply mean that a domain is not popular, in which case malicious behaviour cannot be inferred. Instead, since we are looking for possible C&C domains, we create a bound and look for all domains that have a degree centrality within that bound.

Now, since this bound can potentially return a large portion of the graph, we then use some of the domains returned by this centrality as seeds for the BP algorithm (Belief Propagation), while the rest will be later combined with the results of the BP algorithm, then filtered and returned as the final sub-network of possibly infected hosts and domains.

### 2.2 Belief Propagation

We use the Belief Propagation algorithm to return a greater and somewhat more accurate set of possible C&C nodes, which otherwise, would be hard or take a long time to find by investigating nodes with the use of other methods. The premise of the BP algorithm is simple: imagine we have a set of nodes. All those nodes have a label, and each node's neighbour has an idea of the possibility of its neighbouring nodes having a certain label or message. Based on this possibility, a similarity score is assigned to each node. In the case of networks and malicious behaviour, we can think of that message as Malware, and when this algorithm is done running, it returns a score that shows the influence one node might have over other nodes, which potentially represents malicious hosts that have an influence on infected domains as well as other infected hosts (possible C&C pairs).

The main advantage of this algorithm is that we allow the network itself to find some of the possible C&C pairs, which is much more efficient and more accurate than using other means to achieve the same result. And that result will be in the form of nodes which have a high score according to the BP algorithm. These nodes are then combined with the nodes from the previous centrality algorithm (duplicates removed), and are then filtered by checking for malicious activity.

### 2.3 Checking for suspicious activity

Within the edges of the suspicious domains are their interactions with hosts (some of which may also be suspicious). These interactions are in the form of the strings from the log file. Determining whether these strings have any irregular or malicious behaviour is not a simple task as the data can look deceptively normal (as though a non-infected host and/ or domain were interacting). There is, for this reason, no definitive way to be certain of malicious behaviour; however, certain checks can have a decent chance of detecting an actual infected host. Our method employs one of these checks.

The CNAME operator refers to the renaming of an IP-address. Multiple instances of this CNAME within the response of a domain mean that multiple mappings exist for the same IP-address. Normally, a renaming of an IP-address does not happen often, and does not occur more than twice within the same response. So, seeing CNAME appear more often than the norm is indicative of irregular and potentially malicious behaviour (this could, for instance, be multiple redirects caused by Malware).

For this reason, we simply parse and check each response string to see if it contains a certain number of CNAME operators. Doing this check enhances the accuracy of our returned C&C domains, and increases the likelihood that some of those hosts and domains are actually infected.

# 3 Experiments

Due to the sheer size of the data itself, running, let alone reading, all of it would require access to computers or large amounts of time that we simply did not possess. For this reason, it is a more suitable approach to look at a smaller amount of time, for example - a day's worth of data, instead of all the data that comprises several months of network information. Even just one day of data could contain a large amount of infected domains and hosts, and would pose quite a challenge in order to find those that may or may not be infected.

## 3.1 A day's worth of data

Many malicious hubs, malicious hosts, infected domains, and other infected hosts can appear within the time-frame of a day. Domains that were completely fine one day, can become infected on the next. So it is not unreasonable to test and do experiments involving just one day's worth of DNS log data. And in fact, looking at one day might be more beneficial than looking at an entire month, especially if later experiments also focus on just a day of data but later on in that month.
This could potentially form a basis for creating a more in-depth, focused, and localized area of interest within the possible infected sub-network(s) of the entire network, which could be more useful than a broad analysis involving an entire month of data.

# 4 Results

Here are the results of the experiments: ...

## 4.1 Interpretation

The results show...

# 5 Conclusion

In conclusion...

### Acknowledgments

Use unnumbered third level headings for the acknowledgments. All acknowledgments go at the end of the paper. Do not include acknowledgments in the anonymized submission, only in the final paper. ...

### References

Examples:

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D. S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609-616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer-Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.