University of Amsterdam
Graduate School of Informatics

Identifying Spam Malware Infections, using DNS MX Request Analysis

Bas Vlaszaty
Bas.Vlaszaty@os3.nl

Supervisor:
Casper Joost Eyckelhof, Quarantainenet BV

August 11, 2014

Abstract

Malware designed to be used for sending spam is a very real problem. It is important for network administrators to be able to monitor their network and keep it clean from such spambots. Quarantainenet is a company that has a solution to automatically do this task of monitoring the network to see if there are machines present that are sending spam. To reliably do this detection many types of data are being monitored, but at this time DNS MX requests are being ignored. To find out if monitoring DNS MX requests can help in reliably detecting whether a machine has been infected with spamming malware, the DNS MX traffic of a number of clients of Quarantainenet was captured over a two week period. After that the traffic was analysed with four different analysis techniques to see if it was possible to determine whether a machine was sending spam, solely based on these DNS MX requests.

The analysis on its own proved to be very doable, but the difficulties emerged in being able to find a good model for the behaviour of an infected machine. In the available dataset there is only one identified and confirmed occurrence of a spam session. One confirmed case is not enough to base a good model for spamming behaviour on. Using it to train a classifier is also not possible, again because of the fact that there is only one case in the dataset.

The information that could be extracted from the available data does give an indication of the activity of the machine, but due to the nature of DNS MX requests their value in reliably determining whether a machine is infected with spamming malware is limited. Determining whether observed behaviour is good or bad remains an issue.

The analyses do however give insight in what is going on in the network and can still lead to valuable insights for network administrators.

Contents

1 Introduction

As a company it is important to keep your network infrastructure safe and clean of malware. A very common threat is that machines might get infected by botnet software. This botnet software is able to control the machine and use it to execute commands it receives from its command and control server. The owner of the botnet can then issue commands to all the bots in the botnet, for instance to perform DDoS attacks or send email spam. By doing this the victim's infrastructure and computing power is used for these illegal acts.

Botnets vary in size but the large ones can contain millions of machines [1]. These networks are typically rented out to computer criminals to use for their illegal activities. Protecting your infrastructure from those kind of threats is part of every-day reality for network administrators these days. Apart from keeping the systems up to date, constant monitoring of what is going on in the network is essential to detect and clean possible infections. [2] To do this detection, different types of network traffic is constantly being analyzed by network monitoring solutions.

The solution that Quarantainenet [3] has to keep the internal network safe is to continuously monitor traffic on the network. Different sensors in the network monitor the network in different ways for the presence of malware. The sensors include for instance checking traffic against blacklists of known bad ip addresses and domains. It also includes honeypots to detect machines scanning the network or even trying to exploit them. There are also sensors that monitor traffic in different ways. All these sensors contribute to a machine in the network receiving a score on how bad it is behaving in the network. The philosophy is that by combining information from the different sensors, unwanted behaviour in the network will raise the score of a machine. If this combined score exceeds a threshold, the machine can be marked as infected and appropriate measures can be taken. The way the Quarantainenet system works is that when a machine has been marked as infected, it will be placed in a special subnet. From this subnet internet access is restricted, only a few services are available. When a user tries to browse the internet, a warning message will tell him that his machine has been marked as infected and that he should take action to clean the malware. When this is done, full access will be restored.

By effectively separating the infected machine from the rest of the network, the system achieves two things. Firstly the separation will stop the malware from spreading within the local network. This keeps the network safe. Secondly by forcing the user to clean up the infection when it occurs, awareness for the dangers of malware will be raised. With time, users learn to be careful when working on the internet.

In a system like this, the accuracy of the detection is critical. It can only function if the detection is reliable so it will have a high true positive rate and a low false negative rate. Quarantainenet is therefor constantly working on ways to improve the detection. One of the aspects that so far have not been investigated is the analysis of DNS MX requests. This research will attempt to investigate whether analysis of DNS MX traffic can aid in determining a possible malware infection in the network.

1.1 Spam botnets

Spam messages are a big part of the global email traffic. Worldwide around 120 billion spam messages are sent each day, which is around 60% of the total email traffic [4]. This enormous amount of traffic is generated by large networks of computers that have been infected by malware that makes them part of a botnet. This way the computer criminal can use the infrastructure of his victims to send his spam messages. This means he does not have to pay for the network traffic or energy costs of the computers.

Modern botnets are set up as generic bots that can load different modules to perform different tasks. There are modules for sending e-mail spam, for performing DDoS attacks, for searching the machine for credit cards or password details and many more [5]. Knowing that, the term spambot is not completely accurate, because the bot can do more than just send spam. It is however widely used as a term for a bot that is being used for spamming purposes.

1.2 DNS MX

In this research the DNS MX requests are used to try and identify a host infected with spamming malware. DNS MX requests are a part of the Domain Name System (DNS). DNS is a very important system in the structure of the internet. It is used to translate domain names, like for example *www.google.com*, to ip-addresses, which are numerical addresses of machines connected to the internet. This translation service is used by virtually every application that is connected to the internet.

DNS also offers services specifically for sending mail. This is where DNS MX records come in. The MX record stands for Mail eXchange record, and they function in much the same way as described above. The MX record is used to tell a server where to deliver the mail for a certain domain.

Suppose the following scenario:

1. Server A wants to send an email to email-address *someone@example.com.*
2. Server A does an MX request for *example.com* to DNS
3. DNS answers to server A that the mailserver for *example.com* is *mail.example.com*
4. Server A now knows to deliver the email to *mail.example.com*

While this example shows only the basic function of the DNS MX request, it illustrated the principle enough for the purpose of this research.

When sending an email, this process has to be repeated for every destination address. This means that logging all these requests might provide some insight into all the mail that is being sent by a host.

1.3 Research question

With this background knowledge in mind, the central research question is:

Is it possible to reliably identify a machine infected with malicious spamming software by analyzing DNS MX requests from the network.

2 Related work

Detecting spam by analysing DNS MX requests is not a known technique. Some methods exist to use DNS records to counter spam, but those are mostly in the realm of checking the validity of the sender through SPF or DKIM [6].

On the subject of network traffic analysis there are a number of examples of previously done research that might prove to be useful in this project. [7] and [8] show methods of creating classifiers to detect malicious types of network traffic. Different kinds of classifiers are used to be able to accurately classify sample traffic. [9] uses a similar approach by investigating different types of classifiers to detect traffic from P2P botnets during their initialization phase. In this research additional requirements other then accuracy are set for the proposed methods. These are requirements on the subject of for instance adaptability of the classifiers. It is concluded that none of the proposed methods meets all the set requirements.

In [10] research is performed on the characteristics of different types of malware. This gives a good insight into the inner workings of this type of software. Understanding the malware is a big step in finding ways of detecting and disabling it.

3 Approach

This section covers the structure of the research conducted. To answer the research question, a dataset is needed that is to be analysed. This dataset was gathered with the help of customers of Quarantainenet.

Next to the dataset, there has to be a way to validate the findings. There has to be a way to check if the result of the analysis is accurate or a false positive. For this validation data is available from sources independant from the main dataset. Then the analysis techniques have be developed. Four different techniques are designed to analyse the data: frequency analysis, periodicity analysis, entropy analysis and flow analysis. Each of these techniques is explained in depth in the next section. With the analysis techniques in place, the dataset is analysed to see if it is possible to identify some suspicous events. Then finally the suspicious events are compared to the verification data to see if they are actually examples of malicious activity going on in the network or if some other explanation can be found.

The results from this process will be used to answer the research question.

3.1 Data

The data that was used in the research was gathered by Quarantainenet from 3 of their customers. For privacy reasons these customers will remain anonymous. When collecting these datasets, two things were recorded for each customer. The first dataset, of which some statistics are shown in table 1, contains records of all the MX requests that were seen in the network in a certain period of time. This is the network traffic and will be used to do the analysis on. Important to note is that this is real, unannotated data. This means that there is no information available on whether a host in the dataset is actually infected or not. It is also uncertain if there is an example of a spam session in the dataset. This uncertainty can make it difficult to find out what a spam session might look like.

The second dataset contains all the incidents that are reported by the other sensors of the Quarantainenet network monitoring system. This information might be useful in finding an actually infected machine in the network. The collection ran while the project was in progress.

The final datasets contain data from a period of about 2 weeks.

Name	Number of records	Duration
Dataset A	3028	11 days
Dataset B	67.386	11 days
Dataset C	1.975.765	11 days

Table 1: Captured datasets

More detailed information about the datasets can be found in appendix

A.

Each entry in the raw dataset is a string containing 3 pieces of information:

1. **Timestamp**
   Time the request was made, formatted as [**2014/02/16 10:20.30**]
2. **Source ip**
   The ip address of the machine that made the request.
3. **Domain**
   The domain of which the mailserver was requested.

In a preprocessing stage these pieces of information are parsed and then stored in a CSV format for later use.

3.2 Verification data

As the dataset is not annotated, possible findings will have to be evaluated in some way. There has to be a manner in which the detection of a suspicious event can be classified as a true positive or a false positive. This way the analysis techniques can be tested and improved. To be able to do this, verification data is required. This data will provide an objective truth for the evaluations

There are three sources available to be used as verification data:

1. **Online blacklists** Blacklists like the Spamhaus Zen blacklist can check if an ip address is known to be a spammer. If a lot of spam activity comes from an ip, it will end up on such a blacklist.
2. **Generated incidents** The Qmanage network monitoring system automatically generates reports based on different sensors in the network. These reports can tell us something about for instance malware infections.
3. **Customer reports** The customers that provided us with the log data were specifically asked to report any unusual activity that might have to do with spam.

These sources together will be used as a ground truth. If information from these sources can be attributed to a suspicious network event, it is a strong indication that malicious activity is going on.

4 Analysis methods

For the analysis of the data 4 methods are devised. They will attempt to find irregularities in the dataset, and possibly identify a spam session.

4.1 Frequency analysis

The first method of analysis is based on the frequency of the traffic. Sending spam will normally happen in big volumes, so one can assume that in order to send these large quantities of spam, a lot of DNS MX requests will be nescessary. This is why looking at the rate at which the requests are performed should be a good indicator of what is happening on a certain machine.

To be able to analyse the frequencies, first a histogram is created from the logs based on the timestamp of the events. Typical binsize is 10 minutes. This histogram will give an indication of the host activity in the given timeframe. Creating a histogram is a frequently used technique to convert a series of separate events into a graph showing the activity. This visual representation will provide a good overview of the progress of activity.

4.2 Periodicity recognition

When looking at traffic in a network, a part of the traffic is often generated by automatic processes, that are on a periodic schedule. Extracting the periodicity from the traffic might prove to be a good way to see the nature of the traffic generated by a machine.

4.2.1 Calculating autocovariance

The periodicity detection is implemented by calculating the autocovariance. Autocovariance is the covariance of a series of values against a time-shifted version of itself. In simple terms, it is a measure of the similarity of $f(x)$ with $f(x+t)$, where t is the so-called *lag*. It is the shift of the function compared to the original.

When the autocovariance is calculated, a covariance graph is returned. It will show the similarity for each lag value. To determine the periodicity of the original signal, the highest peak from the covariance graph where $t > 0$ is picked.

Because of the nature of this calculation, values for negative t are the same as for t so they can be discarded. Also note that the highest peak in the covariance graph will always be for $t = 0$ by definition. Covariance with lag of 0 represents the similarity of the series with itself without being shifted. This will cause the correlation to be perfect and will generate the highest covariance value.

4.2.2 Periodicity strength

Now the lag with the highest covariance value has been found, the periodicity in the data has been found. However, not every set of data actually contains a periodic process. The algorithm described in the previous section will always return the lag for which the covariance is the highest, but there is no measure on how good of a match it actually is. To be a valuable analysis, the periodicity calculation needs some measure of certainty that this periodicity is actually there.

As a measure for this, $cov(t)/cov(0) = c$ was used where $cov(t)$ is the covariance value of the highest peak, $cov(0)$ is the covariance value for a lag of zero, so this is the highest achievable covariance. c will be between 0 and 1 where a value of *1* would mean perfect correlation and a value of *0* would mean no correlation.

This method normalizes the autocovariance graph. Normalized autocovariance is also know as autocorrelation. Using this normalized graph makes it possible to compare the highest peaks at $t{>}0$ from different graphs. Using these heights, we can measure the strength of the periodicity and compare them.

4.3 Entropy based detection

In "Entropy Based Analysis of DNS Query Traffic in the Campus Network" [11] a method is described to detect presence of spambot and/or DDoS traffic in the network by analysing the DNS query entropy. While this research did not cover specific usage on DNS MX requests, it is still an interesting concept to try. In the paper, the entropy of the contents of the requests would go down when there was a spamrun or DDoS attack. When applied to DNS MX requests, the expected results would be the same. With a spamrun in progress, the entropy of the contents of the requests would be expected to go down.

Again, the applicability to DNS MX requests will need to be tested.

This analysis is based on creating a histogram of the records from the dataset, and calculating the Shannon entropy over each of the bins of the histogram.

The Shannon entropy of data is given by:

$$H(X) = -\sum_{x} p(x) \log p(x).$$

Here $p(x)$ is the probability mass of x in X. In other words, the Shannon entropy is a summation of the chance times the log of the chance of each element in X.

This will result in a high entropy for a dataset with an even distribution of elements, and a lower entropy for a dataset in which some of the elements occur much more often then others. Entropy is therefore a measure for randomness, because with true randomness the chances of every option occurring are equal.

When calculating the entropy of different sized datasets, it appeares that the entropy will change when the dataset is smaller or larger. A set containing more records will generally have a higher entropy score then a set containing less records. In order to try and remove this effect, ideally the amount of records in each bin should be about the same. For this experiment a binsize of 24 hours was chosen. This way the effect of the night hours getting very low scores because there is barely any traffic is mitigated. The paper describing this technique also uses 24 hour binsizes. The reason for that is not discussed, but it might also be to mitigate this effect.

4.4 Flow extraction

The flow analysis takes a different approach than the previous analysis methods. In the methods described before, often a histogram was used as a base for quantifying the network traffic. In "Detection of Spam Hosts and Spam Bots Using Network Flow Traffic Modeling" [12] an analysis method is described that is based on so-called network flows. In the research a flow is seen as a communication session between two hosts. In a flow, different metrics of the communication session are logged, like for example the duration of the flow and the number of bytes transferred.

In the case of DNS MX requests, there are no communication sessions as described above, as the communication session only consists of a single request, followed by a single answer. In the available dataset, the answer is not even logged, so if flows would be extracted in the same way, all flows would consist of exactly one request, and it would be no different from the records in the dataset.

In an attempt to use this analysis method, a different way of extracting flows was devised. For this analysis a flow is seen as a period of continuous activity of a host. This means that a request from a host will start a flow, and subsequent requests will be added to this flow. The flow ends when there has been no activity for a number seconds, set by the algorithms *inactivity threshold* When after this period of inactivity a new request is made, this will start a new flow.

By extracting flows in this manner, the activity sessions of a host can be characterised by a number of metrics. Metrics extracted from each flow are:

1. **Source ip:** ip address of the host doing the requests.
2. **Flow start time:** Timestamp of the first request in the flow.
3. **Flow end time:** Timestamp of the last request in the flow.
4. **Duration:** Duration of the flow in seconds.

5. **Volume:** Number of requests in the flow.
6. **Rate:** Average number of requests per second.
7. **Domains:** List of the requested domains.
8. **Unique domains:** Number of unique domains.

For the extraction of the flows from the datasets, typically an inactivity threshold of 60 seconds was used. This would generally result in a set containing a large amount of very small flows and a small amount of large flows. This characterizes the nature of the DNS MX traffic quite well, where most hosts only send a couple of requests at a time, and some hosts send big batches at once.

One of the advantages of describing the dataset as a number of flows, is that it becomes trivially easy to for instance disregard all short or low-volume flows, as they will most likely be of no interest when looking for a machine that sends large amounts of spam. It will be the flows containing a large amount of requests that will be most interesting.

5 Results

The analysis techniques described above were used on the available datasets. As the datasets are not annotated, it is in many cases difficult to confirm whether the analysis method does not work correctly or if there simply is no spam session in the data to detect. In the 3 datasets, one event was detected that was later confirmed to have been a spam session. Because there is only one confirmed case it is not possible to draw conclusions towards the effectiveness of the proposed analysis methods.

That being said, this section will focus on the detection results concerning this case, and will include thoughts on the effectiveness of the analysis methods in this specific case.

5.1 Frequency analysis

From the frequency analysis a number of interesting results can be gained. The graph showing the frequency provides a good overview on the activity of a host, as can be seen in figure 1. This image is an example of a typical activity pattern of a mailserver in the network of a large organization. The plot is based on a histogram with a bin size of 30 minutes. On the y-axis one can see that during the day typically around 3000 to 4000 requests are made each hour.

As could be expected, the activity during the night is very low, where during office hours the activity is much higher. While the patterns for each day are not exactly the same, there are clear peaks visible around 12:00/13:00 and 17:00.

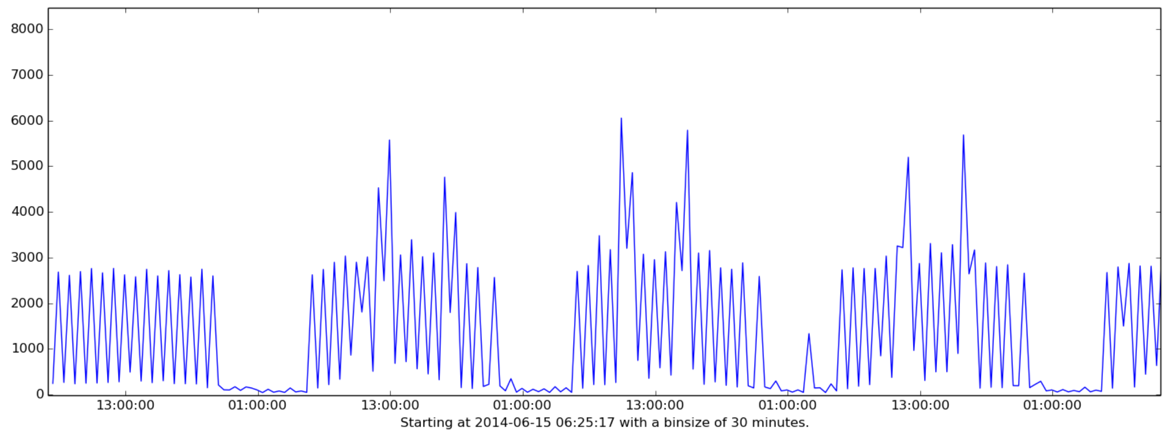

Figure 1: Example of a histogram showing typical daily activity pattern from dataset C. It shows number of requests per hour against time.

Figure 2 shows the frequency graph from dataset A. There is not that much traffic visible. There are some peaks that go as high as 500 requests in 10 minutes. The fact that the rates are not that high and the total volume of the peaks is also limited, does not suggest this is spam activity.

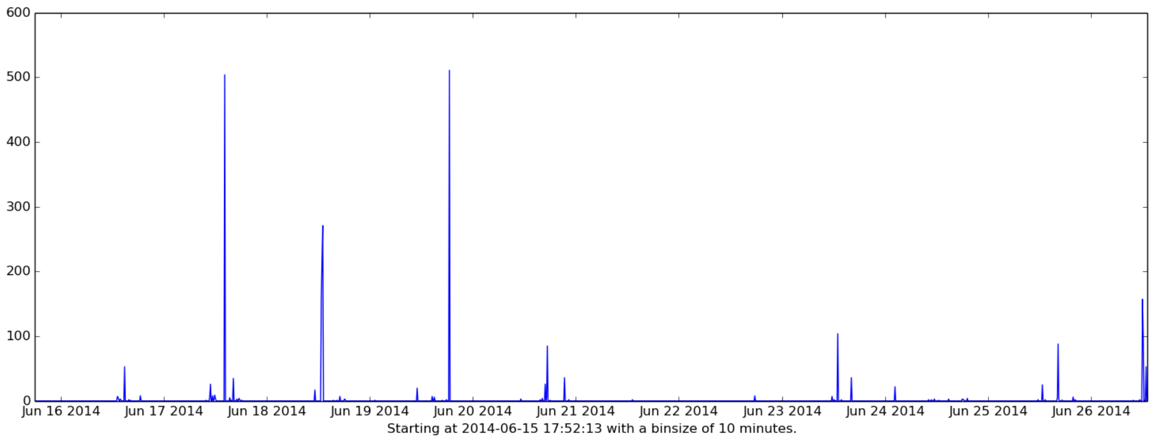

Figure 2: Frequency graph from dataset A

In figure 3 we can see the frequency graph from dataset B. This set contains a bit more traffic than set A. One of the biggest differences is that this set contains much more traffic around some peaks. When looking at the peaks, the biggest peak comes in at around 1200 requests in 10 minutes. Although this is more than what was seen in the previous set, it is still not clear evidence of some malicious activity.

In figure 4 something suspicious is going on.

On the bottom of the graph the daily rhythm is visible, with the nights being recognisable by the low activity. On June 19th there is big spike visible where the activity peaks at around 44.000 requests per 10 minutes. This kind of volume in traffic, along with the fact that there are no other peaks in the dataset that are even remotely close does suggest that this might be malicious traffic. After notifying the customer about this incident, he confirmed that this was indeed a spamrun that had been performed from one of their servers.

After closer inspection the peak in traffic is generated by one specific host. Figure 5 shows the frequency graph from that host. This shows that

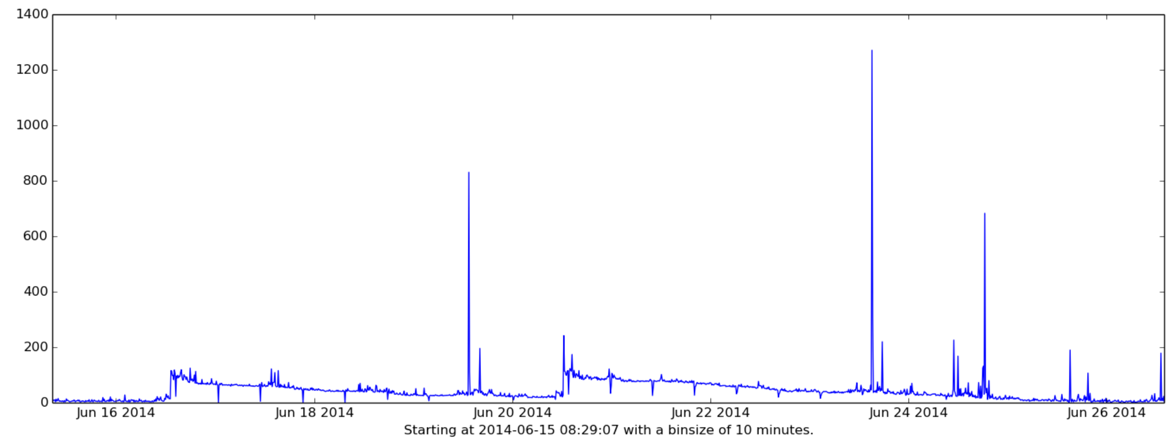

Figure 3: Frequency graph from dataset B

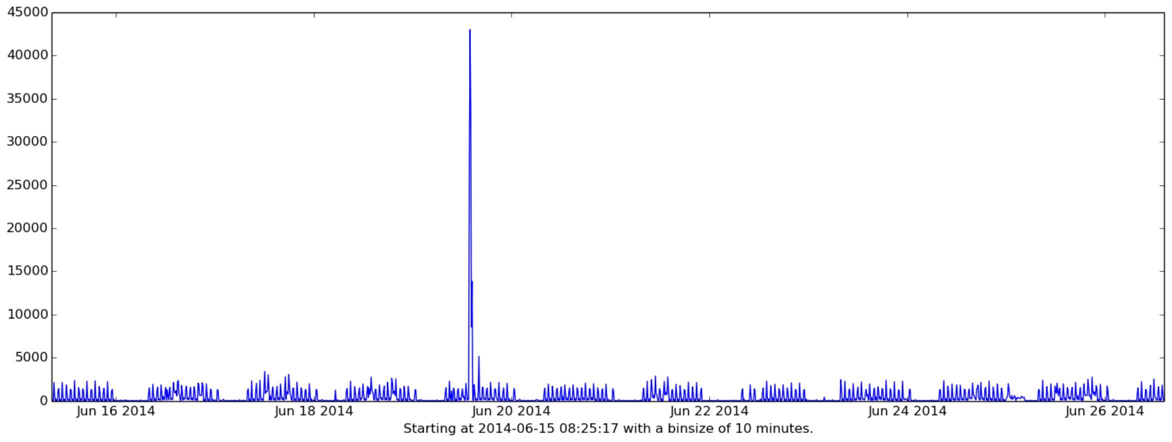

Figure 4: Frequency graph from dataset C

while the host is responsible for this large spike in network traffic, the other activity of this host barely shows up on graph.

This is an example of a spam run. As can be seen, it is clearly identifiable from the frequency analysis.

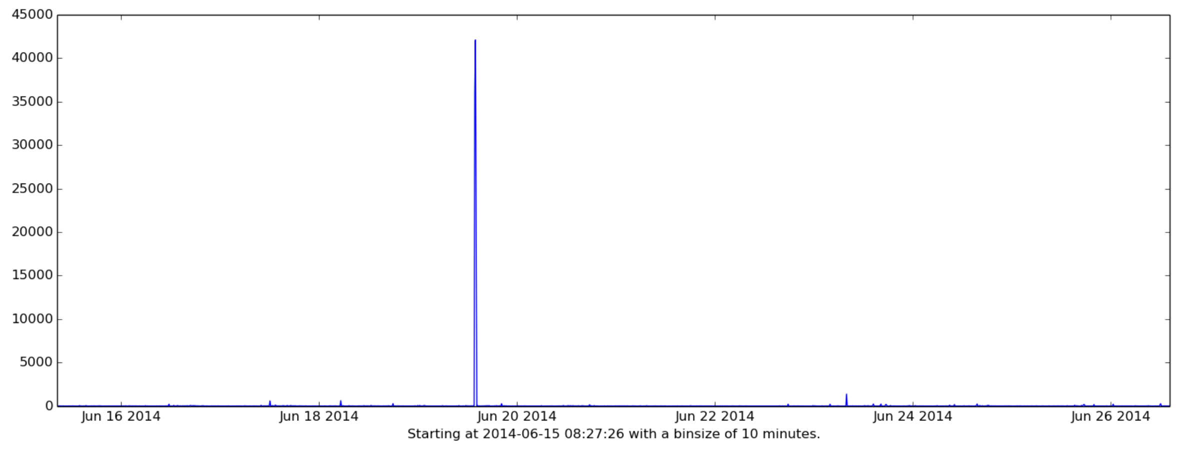

Figure 5: Frequency graph from the spam ip

5.2 Periodicity analysis

In this method of analysis, clear periodic patterns in the activity of a host will generate a periodicity with a high reliability score. When the analysis is run on a host that shows a highly periodic pattern, see figure 7, a very good estimation of the periodicity is obtained. The red line shows the actual data, the blue line shows the normalized results from the autocovariance. The green line is the lag value, which represents the lag value for which the correlation is highest.

This example shows a very strong periodicity. The strength value for this periodicity is *0.98.* From looking at different examples from the datasets, a correlation strength value of more than *0.7* means that a recognisable periodic pattern is present. Lower values tend to contain more noise and less recognisable periodicity. The threshold above which the periodicity is strong enough has to be determined for each application.

Using this method, it is possible to extract patterns from the activity of hosts. The value of this type of analysis in locating spambots however seems to be very low. From what can be seen from the data, an active spambot does not look like a periodic phenomenon, so this detection method is not capable of detecting it.

Periodicity analysis might be useful in doing the exact opposite of spambot detection. It might be capable of filtering out all the periodic processes from the data to get a better view at the non-periodic events. To determine the viability of this application, it will need further research.

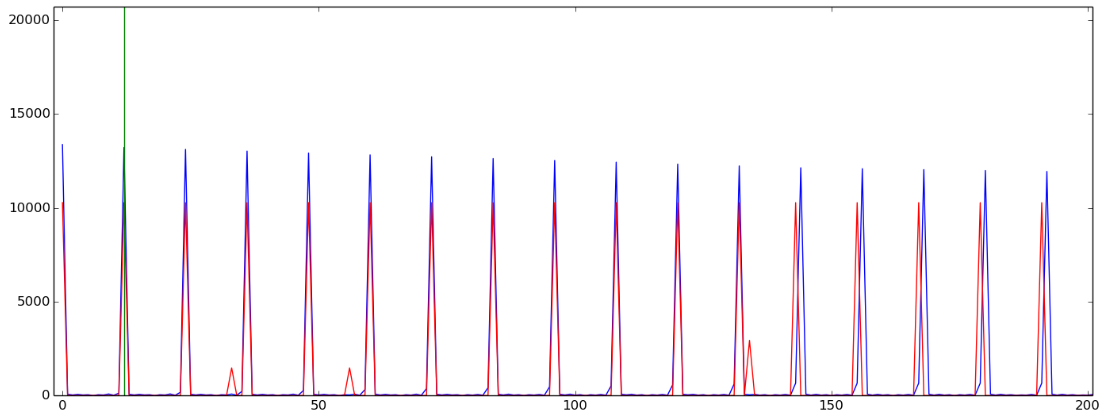

Figure 6: Periodicity from a very periodic activity pattern. Calculated periodicity strength: 0.98

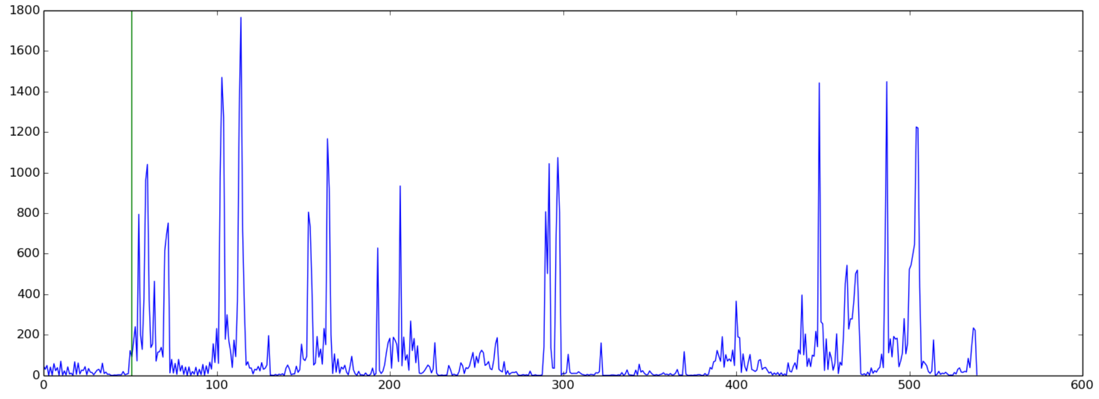

Figure 7: Example of a low periodicity score. Calculated periodicity strength: 0.33

5.3 Entropy analysis

Based on past research [11], the entropy of the requests should go down when a spam run is in progress. The graphs shown are based on a binsize of 24 hours. For this graph, the entropy was calculated over 24 hour periods. The blue datapoints represent the entropy and the green datapoints represent the traffic volume.

In figure 8 the result of the analysis on dataset A is visible. When the volume is very low, the entropy also goes down. In general the entropy follows the trend of the volume.

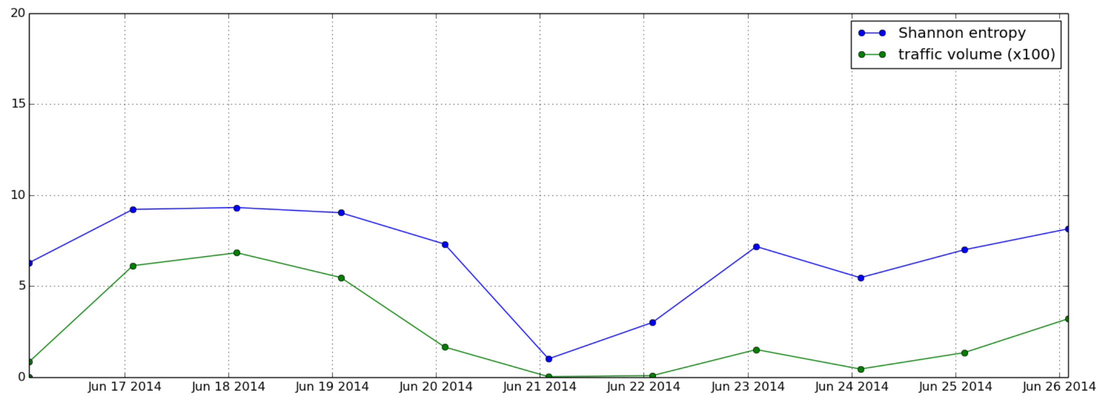

Figure 8: Entropy vs number of requests per day, dataset A

Figure 9 shows the entropy vs volume of dataset B. As can be seen, the entropy is very stable, while the volume fluctuates somewhat. In this dataset the entropy does not follow the trend of the volume as accurately as in dataset A. This stability indicates that entropy is a stable measure under normal circumstances.

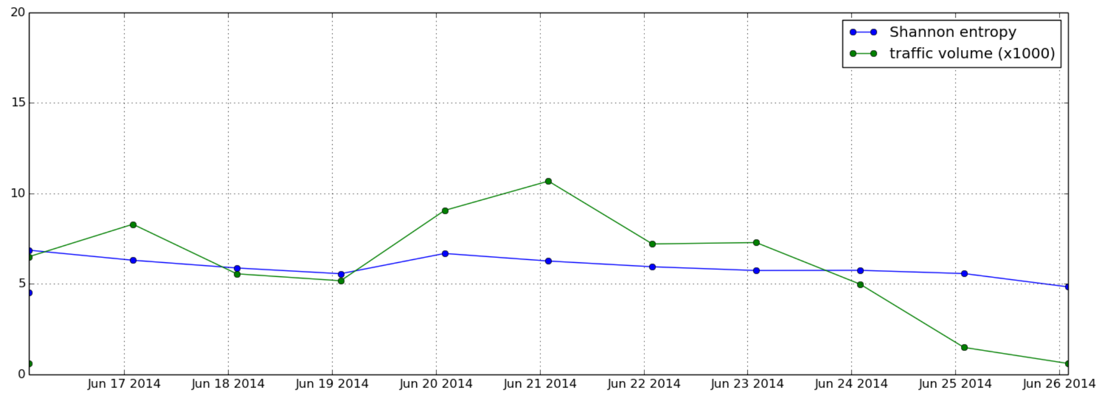

Figure 9: Entropy vs number of requests per day, dataset B

Figure 10 shows a graph of the entropy analysis performed on dataset C. In the traffic volume the day the spamrun took place is clearly visible. Interestingly the average entropy on the days excluding june 19th is 11.1 On the day of the spamrun, it drops to 8.1. This result shows that at least in this case the occurance of a spam run indeed does lower the entropy of the content of the DNS MX requests.

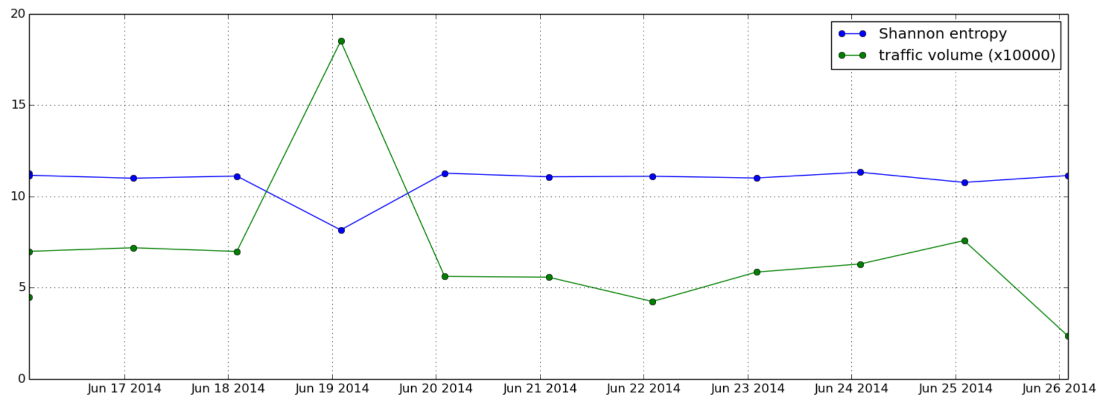

Figure 10: Entropy vs number of requests per day, dataset C

5.4 Flow analysis

Running this analysis on the datasets generates a large number of flows. Flow statistics can be seen in table 2 As is shown in the table, around 75% of the flows, in some datasets even more, contain less than 10 records. These flows are not interesting when looking at detection of spam runs as such low volumes are not likely to be the result of spamming activity.

Dataset	**# records**	**# Flows**	**Flows >10**	**Ratio of flows ¿ 10**
Set A	308	108	27	0.25
Set B	67.386	3356	1305	0.39
Set C	1.975.765	12240	2474	0.20

Table 2: Number of flows

For further analysis the focus will be on dataset C, as that is the dataset where the other analysis techniques identified the spam run. Table 3 shows the results from flow analysis on dataset C. In this table, only flows with a volume of more than 1000 were included. From this analysis it shows that there are two flows present in the dataset that stand out. They both show a very high volume of traffic, as well as a very high request rate. This combination of the volume and rate identify both flows as potential spam runs. Host 1 is indeed the host that generated the spam run identified in section 5.1. This analysis revealed that when this spamrun from host 1 ended, host 2 started a spamrun. Because this run is smaller than the first, it was not noticed in the frequency analysis. In this analysis however, it stands out straight away.

Host	**# Duration**	**Volume**	**Rate (req/s)**
1	1456	**100983**	**69.36**
1	311	1376	4.42
2	509	**21920**	**43.06**
3	5083	3054	0.60
3	4242	2466	0.58
3	4857	2815	0.58
3	2387	1198	0.50
3	4689	3414	0.73
3	3844	2193	0.57
3	1172	2946	2.51
3	3853	2184	0.57
3	2258	1021	0.45
3	1884	2234	1.19
3	5467	2925	0.54
3	4375	1614	0.37

Table 3: Flows with volume >1000 from dataset C

This flow analysis succeeds in pinpointing the hosts involved in the spamrun, as well as showing its duration and volume. While it is only one case, the case does show up very clearly as a non-regular event. By grouping together the traffic into flows, characteristics of the flows becomes very visible. This makes it easy to distinguish between the type of flow. Further research has to prove the usability of the technique in different cases. It could also show whether classification between good and bad traffic flows using machine learning techniques is feasible. To do this an annotated dataset with many examples of spam sessions will be necessary.

5.5 Other observations

After analysing the traffic in 4 different ways, it is very interesting to see that there are a considerable number of hosts doing DNS MX requests. There are many hosts that are not known mailservers that still do these requests. In most cases only very small amounts are done, but still, they are done by hosts that normally would not need to do them. One could say that misconfiguration is the source but then there are a lot of badly configured hosts in the investigated networks. Hosts in a properly configured network should not be doing DNS MX requests, so every MX request done might be a bit suspicious in itself.

What also became apperant when testing the analysis methods was that the verification data was not very useful. From the online spam blacklists there were no results in the sense that none of the monitored IP addresses that were doing DNS MX requests were on a spam blacklist. Some addresses did result in a warning, but the content of the warning was that the IP was not recognised as a mailserver. This kind of warnings do not have much value.

The incident reports from Qmanage were also difficult to correlate with the discovered spam session. The hosts that generated the most incidents in the Qmanage log generally did not do any MX requests. In dataset C, from the 100 hosts generating the most incidents, only 3 actually did any MX requests. From that we can conclude that there is no real correlation between the incidents and the DNS MX requests.

The most useful source of truth was the confirmation of the client that the incident from dataset C actually was a spam run. This was also the only case of a client providing us with information about a confirmed case of malicious activity.

6 Conclusion

In this research the viability of spambot detection based on DNS MX requests was investigated. Four distinct analysis methods were created, each looking at the datasets in a different way.

First the frequency analysis was performed. This method of analysis can provide a good insight in the activity level of the host. We have seen that for the datasets the frequency graph is very much as expected. The day/night cycle is clearly visible, as well as specific peaks in the traffic during certain hours of the day. The spam run included in the dataset clearly stands out.

The periodicity analysis succeeds in finding a repetitive pattern in the dataset. The value of this analysis in the detection of spambots is low as spambots do not seem to display a periodic pattern, but rather do their requests in a single spamrun. This technique might prove useful in identifying other processes on the network though. When applying this analysis it is important to interpret the results correctly. A periodic pattern could also be generated by a proces that constantly needs DNS MX information but caches the result of the requests for a period of time. Not realising that the analysis could be influenced by effects like this could lead to erroneous conclusions.

Entropy analysis of the content of the MX requests shows the same behaviour as described in previous research. When the spam run is happening, the entropy does go down. Finding out the mechanics around this phenomenon will require additional research. It also remains to be seen how accurate this analysis really is. A proper evaluation on this technique might prove interesting.

The last tested analysis technique, the flow analysis, shows to be a promising direction to evaluate further. This new way of identifying sessions of requests does group the requests together nicely. From the results on the available datasets, detecting the spam session becomes a matter of setting an appropriate threshold. Previous research on detecting spambots based on flow data also shows that flow data is a good input for different types of classifiers.

Detecting spambots by analyzing DNS MX requests seems to be feasible. There is information in the set of MX requests from which it is possible to identify a session of suspicious traffic. While the peaks in traffic indicate a suspicious session, it remains difficult to judge the nature of the traffic. In other words, it is relatively easy to see that something is going on, but classification of the session towards good or bad traffic has not been tested enough to be seen as a reliable source of detection.

Because of the nature of these findings, the indication given by the analyzers is mainly useful as additional evidence in a classification system based on multiple parameters. It can provide the support to make a decision on whether a spam run is in progress, but it is not strong enough to completely base a classification on.

With all this being said, a very important remark must be made. In the dataset used for this research there was only one confirmed and clearly visible case of a spamrun present in the DNS MX traffic. This means that this particular case might very well not conform to the typical behavior of a spambot.

Evaluating the proposed techniques on a larger, annotated dataset will be necessary to show the actual accuracy of the techniques.

All in all, identification of a machine infected with spamming malware by analyzing only DNS MX requests is possible in the sense that it is possible to see that something is going on. By judging the number of request and the rate, it is possible to recognise suspicious events. It does however remain difficult to reliably differentiate between good and bad traffic. In this dataset there are not enough examples of bad traffic to base a classifier on. Also it remains to be seen whether the timestamp and the query content contain enough information to base such a classification on.

7 Future work

The most important aspect to the recommendations for future work is finding a dataset that will make it possible to thoroughly evaluate any proposed analysis method. This dataset should ideally consist of a large number of spamruns generated by a number of different types of spambots. It should also include a number of examples of high-volume sessions that are not generated by a spambot, for instance a large mailing like a newsletter with a lot of recipients. The set should also include a very detailed annotation. A description with information about every spamrun; its volume, its duration and its source. With a description like that available, proposed techniques can be evaluated on points like accuracy, false positive rate, performance and other metrics. This will provide a real comparison and real insight in the viability of the methods, and will also aid in developing even more accurate techniques.

Besides better evaluation techniques, different types of analysis techniques might also be an area for future work. For instance deeper analysis of the requested domains might aid in the classification process. To create these new methods the most important thing is to have a good understanding of the characteristics of a spamrun.

References

[1] "Botnet sizes," http://www.abuse.ch/?p=3294, accessed: 2014-07-02.

[2] "Mcafee quarterly threats report first quarter 2014," http://www.mcafee.com/au/resources/reports/rp-quarterly-threat-q1-2014.pdf, accessed: 2014-07-02.

[3] "Quarantainenet," http://www.quarantainenet.nl/.

[4] Symantec, "Internet security threat report 2014 :: Volume 19," http://www.symantec.com/content/en/us/enterprise/other_resources/b-intelligence_report_05-2014.en-us.pdf, accessed: 2014-07-02.

[5] "Malware analysis articles," http://www.abuse.ch/?cat=3, accessed: 2014-07-02.

[6] B. Leiba and J. Fenton, "Domainkeys identified mail (dkim): Using digital signatures for domain verification." in *CEAS*, 2007.

[7] R. Villamarín-Salomón and J. C. Brustoloni, "Bayesian bot detection based on dns traffic similarity," in *Proceedings of the 2009 ACM Symposium on Applied Computing*, ser. SAC '09. New York, NY, USA: ACM, 2009, pp. 2035–2041. [Online]. Available: http://doi.acm.org/10.1145/1529282.1529734

[8] E. Stalmans, "A framework for dns based detection and mitigation of malware infections on a network." in *ISSA*, H. S. Venter, M. Coetzee, and M. Loock, Eds. ISSA, Pretoria, South Africa, 2011. [Online]. Available: http://dblp.uni-trier.de/db/conf/issa/issa2011.html#Stalmans11

[9] S. Saad, I. Traor, A. A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, and P. Hakimian, "Detecting p2p botnets through network behavior analysis and machine learning." in *PST*. IEEE, 2011, pp. 174–180. [Online]. Available: http://dblp.uni-trier.de/db/conf/pst/pst2011.html#SaadTGSZLFH11

[10] C. Rossow, C. J. Dietrich, H. Bos, L. Cavallaro, M. van Steen, F. C. Freiling, and N. Pohlmann, "Sandnet: Network traffic analysis of malicious software," in *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, ser. BADGERS '11. New York, NY, USA: ACM, 2011, pp. 78–88. [Online]. Available: http://doi.acm.org/10.1145/1978672.1978682

[11] D. A. L. Romaa, S. Kubota, K. Sugitani, and Y. Musashi, "Dns based spam bots detection in a university," *Intelligent Networks and*

Intelligent Systems, International Workshop on, vol. 0, pp. 205–208, 2008. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/ICINIS.2008.54

[12] W. K. Ehrlich, A. Karasaridis, D. Liu, and D. Hoeflin, “Detection of spam hosts and spam bots using network flow traffic modeling,” in *Proceedings of the 3rd USENIX Conference on Large-scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More*, ser. LEET’10. Berkeley, CA, USA: USENIX Association, 2010, pp. 7–7. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855686.1855693

Appendices

A Dataset

Name	# records	From date	To date	Sources	Domains
Dataset A	3028	2014/06/15 15:52.13	2014/06/26 11:18.43	15	1966
Dataset B	67.386	2014/06/15 06:29.07	2014/06/26 12:01.01	49	2390
Dataset C	1.975.765	2014/06/15 06:25.17	2014/06/26 12:27.15	72	19.728

Table 4: Captured datasets

These are the cleaned datasets, external ip's were removed,

B Code

The sourcecode used to do the analyses discussed in this paper can be found here: `https://www.os3.nl/2013-2014/students/bas_vlaszaty/rp2`

Please note that this code is in a very experimental state. It is there to be used but no support will be given.