Improving the Stealthiness of DNS-Based Covert Communication

Stephen Sheridan
Technological University Dublin, stephen.sheridan@tudublin.ie

Anthony Keane
Institute of Technology Blanchardstown, anthony.keane@tudublin.ie

Improving the Stealthiness of DNS-Based Covert Communication

Mr Stephen Sheridan, Dr Anthony Keane
Institute of Technology Blanchardstown, Dublin, Ireland
stephen.sheridan@itb.ie
anthony.keane@itb.ie

Abstract: At present, the recommended stance to take regarding Cyber Security is to assume a state of compromise. With the increase in Bring Your Own Device (BYOD), the Internet of Things (IOT) and Advanced Persistent Threats (ATPs), network boundaries have become porous and difficult to defend from external threats. Modern malware is complex and adept at making its presence hard to detect. Recent studies have shown that some malware variants are capable of using multiple covert communication channels for command and control (C2) and data exfiltration activities. Examples of this level of covert communication can be found in malware that targets Point of Sale (POS) systems and it has been hugely successful in exfiltrating large amounts of valuable payment information that can be sold on the black market. In the vast majority of cases, malware needs to communicate with some control mechanism or human controller in order to coordinate attacks, maintain lists of compromised machines and to exfiltrate data. There are many channels that malware can use for its communication. However, in recent times there has been an increase in malware that uses the Domain Name System (DNS) for communications in some shape or form. The work carried out in this paper explores the extent to which DNS can be used as a covert communication channel by examining a number of advanced approaches that can be used to increase the stealthy nature of DNS-based covert channels. Our work describes techniques that can be used to shadow legitimate network traffic by observing network packets leaving a host machine (piggybacking), the use of statistical modelling such as the Poisson distribution and a dynamic Poisson distribution model that can be used to further conceal malicious DNS activity within a network. The results obtained from this work show that current DNS-based C2 and data exfiltration approaches employed by malware have considerable room for improvement which suggests that DNS-based covert communication will remain a realistic threat into the future.

Keywords: Data exfiltration, covert channels, advanced persistent threat (APT), DNS, botnet, command & control (C2).

1. Introduction

Recent industry reports suggest that 91% of malware uses DNS in some shape or form and nearly 60% of organisations do not regularly monitor DNS traffic (Cisco 2016). The distributed and ubiquitous nature of DNS traffic make it an attractive and realistic communication channel for malware. Almost all other network communication protocols rely on DNS and while other protocols can be switched off or blocked at a firewall, it is often impossible to block DNS traffic without having adverse effects on network services and functionality. Since its original inception in the 1980's, the DNS protocol has changed very little and yet still manages to serve as the telephone directory of the Internet as we know it today. It would seem obvious to log and closely monitor DNS traffic given its importance. However, logging and monitoring can be difficult given the high volume of DNS traffic (Gao et al, 2013) and the fact that enabling standard logging in many DNS servers can have a negative impact on performance (Edmonds 2014). In many DNS servers, standard logging is an I/O bound process that can slow down response times to the speed of writing to a hard disk. Even small delays in responding to DNS queries can have an observable impact for end users and so in the majority of cases, DNS logging is switched off in favour of increased performance.

Malware developers are all too aware of the dependence on DNS and the difficulties that organisations can have in monitoring and harvesting useful security intelligence from DNS traffic logs. Therefore, it is no surprise that there has been an increase in malware that exploits the DNS system (Brook 2015.). Recent studies of malware such as MULTIGRAIN (Lynch, Andonov, Teodorescu 2016), that target Windows-based Point-of-Sale (POS) systems provide clear evidence that DNS can be used to exfiltrate large amounts of valuable payment data. Systems that process card data will often restrict, or entirely block HTTP and FTP network protocols that are used for data exfiltration. While network protocols such as HTTP and FTP may be disabled within a restrictive card processing environment, DNS is still necessary to resolve hostnames and is unlikely to be blocked. The characteristics of the MULTIGRAIN POS malware are typical of most malware that exploit the DNS system, in that it communicates initially with a C2 server before it attempts to exfiltrate payment data over DNS. Analysis by Lynch, Andonov, Teodorescu (2016) has shown that MULTIGRAIN checks its buffer approximately every five minutes for payment data and exfiltrates the buffer contents as a series of encoded DNS queries. Like most malware that exploits DNS, MULTIGRAIN's persistence relies on the assumption that system administrators are far too busy monitoring HTTP, FTP and other network protocols to worry about DNS. In this paper, we suggest a number of approaches to increase the stealthiness of DNS-based C2 and data exfiltration traffic. We propose a DNS query schedule based on statistical analysis of the inter arrival times of DNS queries captured over a four-day period from a Passive DNS server. We show that the arrival rate of DNS queries resembles a Poisson arrival process and we propose a method of constructing a dynamic DNS query schedule based on the current DNS activity levels of the host machine. In addition to the Poisson based query schedule we implement a querying strategy that piggybacks legitimate outbound DNS queries leaving the system and we compare and contrast the stealthiness and throughput of each approach against normal baseline traffic. Our study will be of interest to network administrators who are concerned with malware that uses DNS as a covert channel in order to circumvent network security.

The remainder of this paper is organised as follows. In section 2, we describe the basic mechanisms that need to be in place in order to communicate using DNS queries and we outline some of the constraints and limitations in using the DNS system as a C2 and data exfiltration channel. In section 3, we analyse a four day long Passive DNS log and show how the arrival times of DNS queries resemble a Poisson arrival process and in particular how the inter arrival times (deltas) between DNS queries are exponentially distributed. Using this analysis, we propose a dynamic DNS query schedule that uses deltas that are exponentially distributed based on the current DNS activity levels of the host machine. In section 3, we also outline a DNS query strategy that observes network activity on the host machine and only sends outbound DNS queries when it detects legitimate DNS activity, thereby blending its malicious traffic with benign traffic. In section 4, we compare and contrast the data exfiltration capabilities, in particular stealthiness and throughput, of each of the DNS query schedules proposed in section 3, by analysing results from a number of quantitative experiments. Finally, our conclusions are given in section 5.

2. Constraints and Limitations of DNS-based Covert Communication

In this section, we discuss the mechanisms required to carry out DNS-based covert communications and we outline the protocol constraints and limitations that must be adhered to when transmitting arbitrary data over DNS. The mechanisms and infrastructure required to communicate over DNS are quite simple and do not require special equipment or skills. Figure 1, shows a basic DNS communication scenario were an infected machine sends covert messages to a malicious DNS nameserver in the form of encoded DNS queries. The malicious nameserver strips out the encoded message and replies back to the infected machine in the form of DNS query responses in order to complete the communication. It is important to note that outgoing encoded DNS queries do not have to be sent directly to the malicious DNS nameserver. The hierarchical and distributed nature of the DNS system will ensure that encoded DNS queries are routed to the appropriate nameserver based on the domain name that forms part of the query.

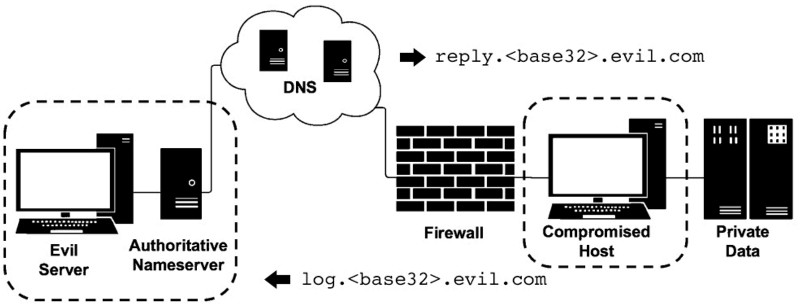


Figure 1: Basic DNS-based covert communication scenario.

2.2 DNS packet size and fragmentation

The DNS protocol operates at layer 7 of the OSI network model and communicates over UDP using port 53. Therefore, it is a connectionless protocol and packets are not guaranteed to reach their intended destination. From a malware perspective, this means that all DNS-based communication must originate from the client side and both the server and client must agree on some kind of packet numbering scheme in order to ensure that arbitrary data can be reconstructed once it has been exfiltrated. Packet ordering only becomes a problem if the size of the arbitrary data being exfiltrated plus the size of the UDP and DNS headers is larger than the maximum size of a DNS packet which is 512 bytes and if dropping packets is not an option.

In the case of stealing payment information from POS systems, a single encoded DNS query might represent an atomic piece of payment information such as a 16-digit credit card number plus a 4-digit expiry date and a 3-digit CCV number. In this case, there is no fragmentation of data and the loss of a few packets out of many thousands of exfiltrated packets is not a major concern.

2.3 DNS beacon messages and data exfiltration

The connectionless communication model also means that infected machines must transmit regular beacon messages to C2 servers to alert them of their existence and to check for updates or instructions from cyber criminals. Work carried out by Shalaginov, Franke and

Huang (2016) and the authors of this paper, Sheridan and Keane (2016), has shown that malware and freely available DNS tunnelling tools such as IODINE (Ekman 2006) and DNSCat (Bowes 2012) use periodic beaconing by default (IODINE ~2secs, DNSCat ~ 5secs) which is easy to detect over time. In the case of the MULTIGRAIN POS malware, it checks its buffers for payment information approximately every 5 minutes which should result in a periodic increase in DNS activity. The periodicity of DNS-based covert channel beacons represents a weak point in the communication chain that can be easily detected once DNS log files or sample traffic is available for analysis.

2.4 DNS query name restrictions

According to RFC 1035 (Mockapetris 1987), domain names can contain the set of alphanumeric characters a-z, A-Z, 0-9 and can include the hyphen symbol but domain names cannot start or end with a hyphen. Although domain names can contain both upper and lower case characters they are not case sensitive. In addition to a limited character set, domain names are also limited in terms of their max length. A fully qualified domain name (FQDN) can contain a maximum of 255 octets (bytes/characters) of which 253 are usable because all FQDN's have an initial length octet and a trailing '.' character that represents the root of the domain. This represents a severe limitation in terms of the amount of data that can be packed into upstream communications (DNS queries) as data must be encoded in a way that adheres to the strict character set rules. Therefore, in order to transmit arbitrary data over the DNS protocol, it must first be encoded using base32 or some other encoding scheme that will translate data (binary or text) into the limited character set allowed by the DNS protocol. The observation of strangely encoded long DNS queries can in themselves be a tell tail sign that DNS-based C2 and data exfiltration is occurring (Born and Gustafson 2010). However, this type of observation requires deeper packet inspection that is often not an option when dealing with large traffic volumes.

2.5 Summary

In terms of data exfiltration, DNS would seem to have a number of severe limitations. It is unlikely that DNS would be a malware developers first choice in selecting a covert communication channel for exfiltrating large files. However, given the right circumstances such as a restricted network environment, small or easily fragmented payload data and time, it is possible to use DNS to exfiltrate a large amount of data over a long period of time. In this paper, we focus on traffic blending and we outline two techniques that focus on breaking the periodicity of DNS-based C2 and data exfiltration in order to avoid detection.

3. DNS Traffic Blending Approaches

Malware that is designed to exfiltrate valuable information over DNS must maximise its throughput while ensuring that DNS traffic is not increased to levels that might raise suspicion. In terms of C2 and data exfiltration, the relationship between throughput and stealth is inversely proportional in that an increase in throughput will increase the risk of detection and therefore reduce stealthiness. Therefore, techniques that camouflage malicious DNS traffic with legitimate traffic are worth exploring in terms of the level of stealth and throughput they provide. There are many causes of legitimate DNS traffic, of which the most common are Web Browsing, Email, System updates, Antivirus Software and third party applications. It is essential to have an understanding of how legitimate DNS traffic behaves on a particular system in order to develop techniques that will camouflage malicious DNS traffic. In section 3.1 of this paper we analyse 4 days of legitimate DNS traffic that was recorded using a DNS server running Berkeley Internet Name Domain (Bind9) and dnstap (Edmonds and Vixie 2013). We use the results from our analysis of this data in the following sections of this paper to develop DNS query schedules that blend DNS-based data exfiltration traffic with a host's legitimate DNS activity.

3.1 Passive DNS log file analysis

In computer networks, a common assumption is that packet arrivals are independent and unpredictable (Atefi, et al. 2016). Traffic measurements are useful for testing actual network performance and for problem solving. Traffic measurements are also employed to develop abstract traffic models that can be used to allocate network resources in order to meet quality of service (QOS) targets. In this section, we analyse 4 days of legitimate DNS traffic that was recorded using a Passive DNS server running Bind9 and dnstap. The recorded traffic represents a 24hr view of DNS traffic from hosts on a college research network. As we are interested in the use of DNS for data exfiltration, our study focuses on the rate of outbound DNS traffic (queries) between the hours of 9am and 6pm when there is increased network activity. One of the oldest and most commonly used arrival models is the "Poisson arrival model" (Williamson 2001). Using the Poisson model traffic is characterized by assuming that the packet arrivals x_n have the following properties, where the probability of the next arrival x_n being less than or equal t is $P(x_n \leq t) = 1-e^{-\lambda t}$.

1. they are independent
2. they are exponentially distributed with the rate parameter λ

Figure 2 shows a kernel density estimate of DNS query deltas for each of the four days in question. As can be seen from figure 1, the plot of DNS query deltas for each day resemble a long tailed exponential distribution which is characteristic of a Poisson model.

Traffic that is machine-initiated or timer driven, such as MULTIGRAIN's ~5min payment exfiltration schedule, will not strictly adhere to exponentially distributed inter arrival times. Therefore, DNS-based C2 and data exfiltration schedules should ensure that inter arrival times are exponentially distributed in order to increase stealthiness and avoid detection. The following sections of this paper outline three approaches that can replace timer based communication schedules.

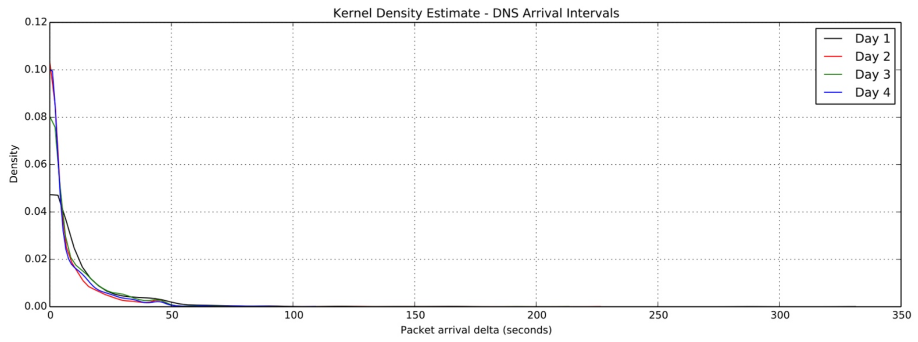


Figure 2: Kernel Density Estimate of DNS query inter arrival times recorded between 9am and 6pm over a four-day period using a Passive DNS server running Bind9 and DNSTap.

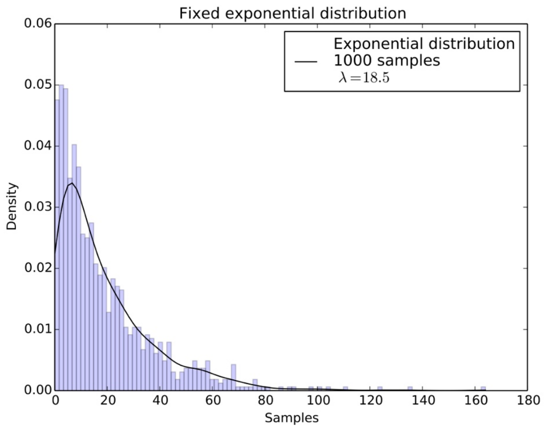
3.2 Traffic blending with a fixed exponential distribution

Detecting periodic patterns in DNS traffic is trivial once appropriate log files are available to analyse. A naive approach to overcoming this periodic weakness is to generate a pseudo random schedule whereby the time between DNS queries is selected at random from a range of possible delays. The problem with this approach is that the pseudo random delays will form a normal distribution over time and it is difficult to select a lower and upper range for the random delays. According to work carried out by Butler, Xu and Yao (2011), it makes more sense to select the delay between DNS queries from an exponential distribution that is scaled to resemble delays observed in some baseline traffic measurement. To achieve this, it is necessary to observe legitimate DNS traffic in order to calculate the mean delay between DNS queries. The mean delay can then be used to generate an exponential distribution that resembles normal network behaviour with a rate parameter of $\lambda = \text{mean delay}$. Table 1 gives a breakdown of delays between DNS queries recorded from our four-day Passive DNS traffic capture.

Table 1: Analysis of DNS query inter packet delays over a four-day period.

Count	**Mean**	**Std Dev**	**Min**	**Max**	**25%**	**50%**	**75%**
8048	18.5	248	0	7231	0.01	0.9	9

Using the data gathered in table 1, it is possible to schedule covert DNS queries with inter packet delays drawn from an exponential distribution with a rate of $\lambda = 18.5$. Figure 3, shows an exponential distribution with a rate of $\lambda = 18.5$ and the pseudo code used to schedule data exfiltration over DNS.

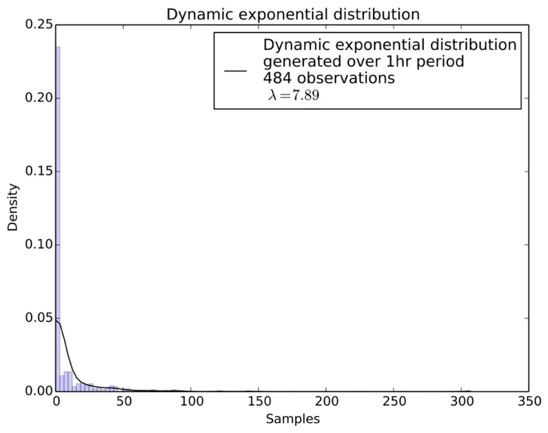


```
While(bytes left to exfiltrate){
    Encode and send DNS exfil packet
    Randomly select time delay T from
    distribution
    Sleep for T
}
```

Figure 3: Exponential distribution based on four-day Passive DNS analysis and pseudo code for exfiltration schedule.

3.3 Traffic blending with a dynamic exponential distribution

One weakness of the traffic blending technique discussed in section 3.2 is that DNS traffic does not have a constant arrival rate. Exponential distributions based on historical analysis of DNS inter packet delays will produce traffic patterns that resemble legitimate network traffic in the long term but will not reflect short term trends that might occur from hour to hour or day to day. A data exfiltration schedule that models short term trends can take advantage of periods of increased DNS traffic by exfiltrating more data and can stay under the radar of a system administrator during periods of decreased activity. The traffic model discussed in section 3 can be modified so that it builds a dynamic exponential distribution by observing actual DNS traffic. Figure 4 shows an exponential distribution generated by observing actual DNS queries on a host machine, calculating the inter packet delays (deltas) and adding them to a circular buffer. A circular buffer is used so that older deltas will be replace by newer values when the buffer is full. Figure 4 also shows the pseudo code used to schedule data exfiltration. As can be seen, the *sleep* step has been replaced with code that observes DNS activity for the time period T which is randomly selected from the continuous exponential distribution.



```
Create a small circular buffer
containing initial exponential
distribution
While(bytes left to-exfiltrate){
       Encode and send DNS exfil packet
       Randomly select time delay T from
       distribution
       Observe actual DNS traffic for
       time T, calculate inter packet
       deltas and append to circular
       buffer
}
```

Figure 4: Exponential distribution based on a 1hr exfiltration/observation period and pseudo code for exfiltration schedule.

The circular buffer effectively acts as a short-term memory that reflects the most recent levels of DNS traffic. The size of the circular buffer determines the memory size and therefore, determines how reactive the exfiltration schedule is to short-term trends.

3.3 Traffic blending based on “piggybacking” observed traffic

Assuming that a piece of malware has the ability to observe DNS traffic on the host machine, it is then possible to “piggyback” legitimate DNS queries so that the timing of DNS-based C2 and data exfiltration correlates with normal traffic patterns. The advantage to using this approach is that malware developers do not have to create traffic models that resemble an infected machines normal behaviour. The throughput of the “piggybacking” approach is heavily dependent of the level of network activity on the host machine. If the host machine is busy, then many DNS-based C2 and data exfiltration packets will leave the system. Conversely, if the host machine in quiet, then very little DNS-based C2 and data exfiltration packets will leave the system.

4. Analysis of Traffic Blending Approaches

In this section, we analyse each of the traffic blending approaches discussed in section 3. Our analysis takes the form of a series of one-hour long trial runs of each traffic blending approach were the throughput is measured in one minute intervals and compared against legitimate DNS queries during the same one-hour period.

4.1 Fixed exponential distribution and “Piggybacking”

In our first experiment, we tested a fixed exponential distribution with $\lambda = 18.5$ over the course of one hour. At the end of testing, 895 DNS-based data exfiltration queries left the host machine giving a communication ratio of 1:5 with legitimate DNS queries during the same time period.

As can be seen in figure 5(a), data exfiltration queries are randomly distributed and significantly lower than legitimate DNS queries. The data exfiltration DNS queries are well camouflaged by legitimate traffic but the throughput is quite low, particularly at ~40mins where legitimate DNS activity is high. Figure 5(b) represents another hour-long experiment using the "Piggyback" blending technique. In this case, 3190 data exfiltration queries are sent giving a ratio of 1:1 with legitimate DNS queries. It is clear from figure 5(b) that there is a very close correlation between the timing of data exfiltration DNS queries and legitimate traffic. This is a useful feature in terms of stealthy behaviour. However, piggybacking high levels of legitimate traffic so closely exposes more data exfiltration traffic for analysis. In effect, every time a legitimate DNS query is sent there is a chance of being detected.

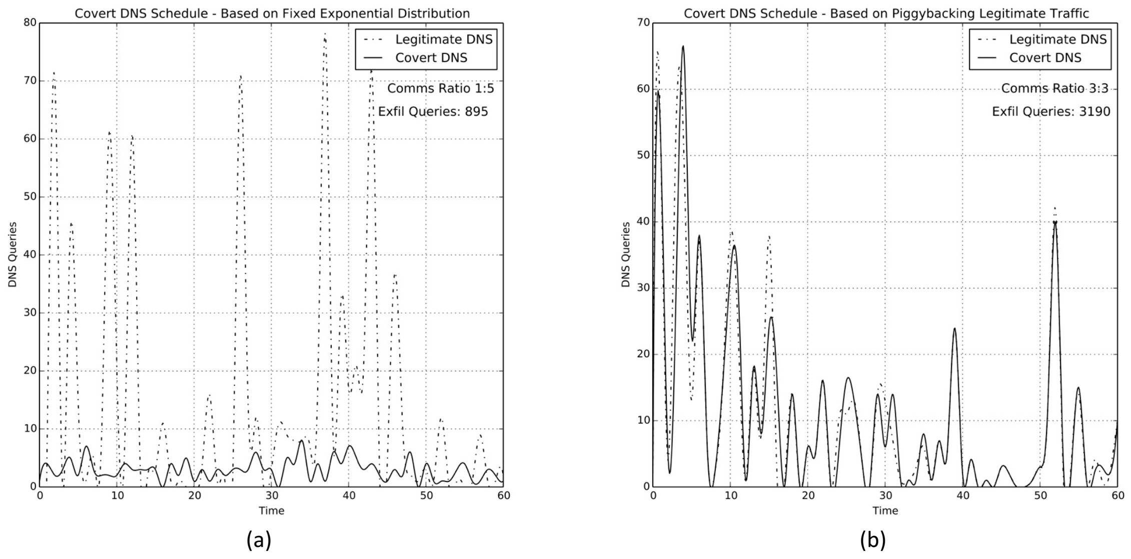


Figure 5: (a) Data exfiltration using fixed exponential distribution with $\lambda = 18.5$. (b) Data exfiltration based on "piggybacking" legitimate traffic.

4.2 Dynamic exponential distribution

In our second set of experiments we obtained results for the dynamic exponential distribution exfiltration schedule by running three hour-long tests with varying circular buffer sizes. Figure 6 shows the results for each test and gives the communications ratio, number of exfiltration DNS queries and the buffer size used. Figure 5(a) shows that a buffer size of 50 results in data exfiltration pattern that resembles legitimate traffic up to a ~40mins where there is a significant spike in exfiltration traffic. The spike in exfiltration traffic correlates with a spike in legitimate traffic but is far higher which explains the 4:1 ratio of exfiltration packets to legitimate packets. This can be explained by the fact that the buffer started to fill with smaller inter packet deltas as the legitimate DNS traffic spiked and as the legitimate DNS traffic decreased the buffer was not updated with longer inter packet deltas fast enough so exfiltration traffic continued to spike. As can be seen in figure 6(b), a buffer size of 500 results in exfiltration traffic volume that is higher than legitimate traffic, with a ratio of 2:1, but does not contain any large spikes even though the legitimate DNS traffic has three spikes at ~15mins, ~30mins and ~45mins.

In figure 6(c) we can see that increasing the buffer size to 1000 results in a traffic ratio of 1:1 and an exfiltration traffic pattern that closely resembles legitimate traffic. As stated in section

3.3, the circular buffer acts like a memory, so increases in the buffer size expand the memory capacity flattening out any short-term traffic fluctuations.

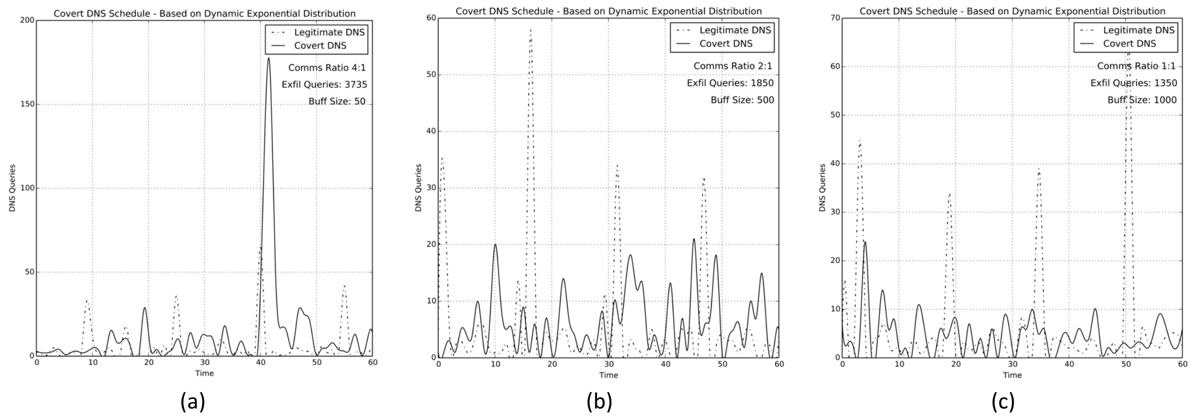


Figure 6: Data exfiltration based on dynamic/continuous exponential distribution with circular buffer size (a) 50, (b) 500 and (c) 1000.

4.2 Comparative results

In this section, we compare results obtained from all three traffic blending approaches. The results were obtained by taking the average of five one-hour runs for each traffic blending approach. As can be seen in figure 7, the piggybacking schedule exfiltrates 638 packets on average over 5 runs and while the Poisson and dynamic Poisson exfiltrate 179 and 270 packets. In terms of throughput, the piggybacking schedule wins out but this is largely dependent on the level of network activity on the host machine.

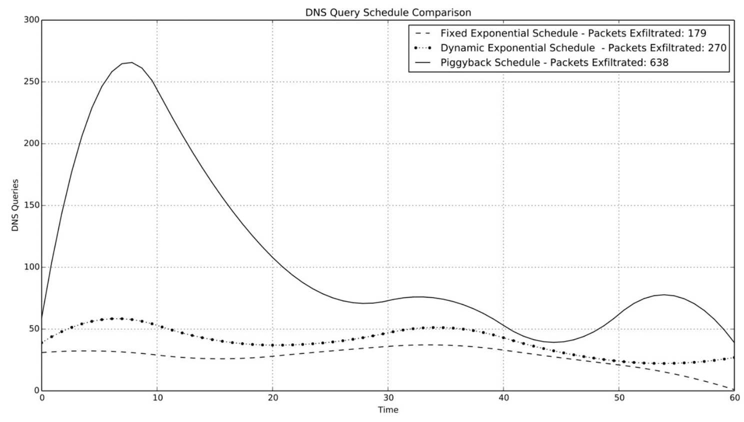


Figure 7: Comparative results for each traffic blending approach.

The piggybacking approach is far superior when it comes to following the DNS traffic trends of the host machine. This characteristic is useful in terms of throughput but will have a negative impact on stealthiness. It would be relatively easy for a network administrator to detect that DNS queries for the same domain are occurring every time there is a legitimate DNS request.

The Poisson schedule with $\lambda = 18.5$, exfiltrates the least packets but its throughput should remain constant regardless of the level of network activity on the host machine. If the host machines actual network activity levels have a mean DNS query rate of 18.5, then this approach should offer a moderate throughput while keeping exfiltration queries at a stealthy level. However, if the host machines network activity levels are significantly different to the mean used for the fixed exponential distribution then the traffic trend line for exfiltration packets will not blend with actual network activity. This will lead to far too many exfiltration packets leaving the host machine, or a less than efficient throughput.

The dynamic exponential distribution schedule, using a circular buffer of size 1000 has a throughput that lies between the other traffic blending approaches. As this approach observes legitimate DNS queries it has the ability to adapt throughput and as a result stealthiness, based on the level of network activity on the host machine. As stated in section 4.2, the size of the buffer will determine how sensitive the exfiltration schedule is to legitimate spikes in DNS traffic. Using a dynamic exponential distribution to control delays between exfiltration queries offers a level of flexibility that a fixed exponential distribution and piggybacking cannot.

6. Conclusion

In this paper, we have discussed some of the limitations of DNS-based data exfiltration. In particular, we have highlighted the periodic or timer based nature of DNS-based C2 and data exfiltration schedules as being a significant weak point when it comes to stealthy communications. In section 3.3, we have demonstrated that is possible to create a dynamic exponential distribution to model inter packet delays that can be used to schedule DNS-based data exfiltration a stealthy manner. In addition, we have shown that a circular buffer with a variable size can be used as a kind of memory of recent legitimate DNS activity which to our knowledge is a new idea. In section 4 we present experimental results that demonstrate the data throughput and stealthiness of each of the DNS-based data exfiltration schedules covered in this paper. We present comparative results for each exfiltration schedule in section 4.2 which show that a dynamic exponential distribution offers reasonable data throughput and can at the same time blend DNS-based data exfiltration packets with legitimate traffic.

As an extension of this work we plan to carry out further experiments to gain a better understanding of how the circular buffer size effects data throughput and stealthiness. It may also be possible to alter the circular buffer size algorithmically so that it adapts to legitimate DNS traffic patterns over time.

References

A. Shalaginov, K. Franke, and X. Huang (2016) “Malware beaconing detection by mining large-scale dns logs for targeted attack identification,” in 18th International Conference on Computational Intelligence in Security Information Systems. WASET.

Atefi, K. et al. (2016) Traffic behavior of Local Area Network based on M/M/1 queuing model using poisson and exponential distribution. In 2016 IEEE Region 10 Symposium (TENSYMP). pp. 19–23.

Born, K. & Gustafson, D. (2010). NgViz: Detecting DNS Tunnels Through N-gram Visualization and Quantitative Analysis. In Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research. CSIIRW ’10. New York, NY, USA: ACM, pp. 47:1–47:4.

Bowes, R. (2012) dnscat2. GitHub. Available at: https://github.com/iagox86/dnscat2 [Accessed July 5, 2015].

Brook, C. (2015) Tracking Malware That Uses DNS for Exfiltration. Threatpost | The first stop for security news. Available at: https://threatpost.com/tracking-malware-that-uses-dns-for-exfiltration/111147/ [Accessed September 16, 2015].

Butler, P., Xu, K. & Yao, D. (2011) Quantitatively Analyzing Stealthy Communication Channels. In J. Lopez & G. Tsudik, eds. Applied Cryptography and Network Security. Lecture Notes in Computer Science. International Conference on Applied Cryptography and Network Security. Springer Berlin Heidelberg, pp. 238–254.

Cheng, G. and Gong, J. (2001) Traffic behavior analysis with Poisson sampling on high-speed network. In 2001 International Conferences on Info-Tech and Info-Net. Proceedings (Cat. No.01EX479). pp. 158–163 vol.5.

Cisco (2016) Cisco Annual Security Report, Cisco. Available at: http://www.cisco.com/c/m/en_us/offers/sc04/2016-annual-security-report/index.html.

Edmonds, R. (2014) dnstap: high speed DNS logging without packet capture. Available at: http://dnstap.info/slides/dnstap_nanog60.pdf [Accessed December 10, 2016].

Edmonds, R. and Vixie, P. (2013). dnstap. Available at: http://dnstap.info/ [Accessed January 17, 2017].

Ekman, E. (2006) Iodine. Available at: http://dev.kryo.se/iodine/wiki [Accessed January 22, 2016].

Gao, H. et al. (2013) An Empirical Reexamination of Global DNS Behavior. In Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM. SIGCOMM '13. New York, NY, USA: ACM, pp. 267–278.

Jain, R. and Routhier, S. (1986) Packet Trains--Measurements and a New Model for Computer Network Traffic. IEEE Journal on Selected Areas in Communications, 4(6), pp.986–995.

Leijenhorst, T., Chin, K.-W. and Lowe, D. (2008) On the viability and performance of DNS tunneling. In The 5th International Conference on Information Technology and Applications (ICITA 2008). Available at: http://www.uow.edu.au/~kwanwu/DNSTunnel.pdf.

Lynch, C., Andonov, D. and Teodorescu, C., 2016. MULTIGRAIN – POINT OF SALE ATTACKERS MAKE AN UNHEALTHY ADDITION TO THE PANTRY. www.fireeye.com. Available at: https://www.fireeye.com/blog/threat-research/2016/04/multigrain_pointo.html [Accessed April 21, 2016].

Mockapetris, P. (1987) RFC 1035 Domain Names - Implementation and Specification. Available at: https://www.ietf.org/rfc/rfc1035.txt [Accessed November 21, 2014].

Paxson, V. and Floyd, S. (1995) Wide Area Traffic: The Failure of Poisson Modeling. IEEE/ACM Transactions on Networking, 3(3), pp.226–244.

Sheridan, S. & Keane, A. (2015) Detection of DNS-Based Covert Channel Beacon Signals. Journal of Information Warfare, 14(4).

Williamson, C. (2001) Internet traffic measurement. IEEE Internet Computing, 5(6), pp.70–74.

Yaneza, J. (2015) NewPosThings Has New PoS Things - TrendLabs Security Intelligence Blog. TrendLabs Security Intelligence Blog. Available at: http://blog.trendmicro.com/trendlabs-security-intelligence/newposthings-has-new-pos-things/ [Accessed February 7, 2017].