ZAMBIA INFORMATION COMMUNICATION TECHNOLOGY (ICT) JOURNAL

Volume 1 (Issue 1) (2017) Pages 51-56

Exploitation of DNS Tunneling for Optimization of Data Exfiltration in Malware-free APT Intrusions

Aaron Zimba[1,2]
Department of Computer Science and Technology
University of Science and Technology Beijing[1]
Beijing, China
azimba@xs.ustb.edu.cn

Mumbi Chishimba[2]
Department of Computer Science and Information Technology
Mulungushi University[2]
Kabwe, Zambia
chishimba.mumbi@gmail.com

***Abstract*— One of the main goals of targeted attacks include data exfiltration. Attackers penetrate systems using various forms of attack vectors but the hurdle comes in exfiltrating the data. APT attackers even reside in a host for long periods of time whilst seeking the best option to exfiltrate data. Most data exfiltration techniques are prone to detection by intrusion detection system. Therefore, data exfiltration methodologies that generate little noise if any at all are attractive to attackers and can go undetected for long periods owing the low threshold of generated noise in form network traffic and system calls. In this paper, we present malware-free intrusion, an attack methodology which does not explicitly use malware to exfiltrate data. Our attack structure exploits the use of system services and resources not limited to RDP, PowerShell, Windows accessibility backdoor and DNS tunneling. Results show that it's possible to exfiltrate data from vulnerable hosts using malware-free intrusion as an infection vector and DNS tunneling as a data exfiltration technique. We test the attack on both Windows and Linux system over different networks. Mitigation techniques are suggested based on traffic analysis captured from the established secure DNS tunnels on the network.**

Keywords-DNS; data exfiltration; tunneling; malware-free; Advanced Persistent Threat (APT)

I. INTRODUCTION

The DNS protocol is one of the backbones of the Internet without which traversing or browsing the Internet would probably be an impractical undertaking. The DNS protocol and service (henceforth referred to as DNS) operating on port 53 translates IP addresses to hostnames or human friendly names [1] inadvertently eliminating the need to master numerical IP addresses on the Internet. In every local and global network connected to the Internet, we are sure to find DNS. Therefore, the importance of DNS cannot be overemphasized in the current Internet structure. In light of the aforesaid, DNS traffic forms a huge part of the overall Internet. DNS is thus permitted by default by most firewalls and intrusion detection systems (IDS). Notwithstanding the aforementioned, DNS has not been spared of cyber attacks. The DNS threat index [2], a global indicator of malicious network activities directed towards the Domain Name System shows a more than 12% substantial increment from 2015 to 2016. The diagram in figure 1 below shows the DNS threat index for the year 2016 [3] where 100 is the baseline representative of the average threat activity for the duration.

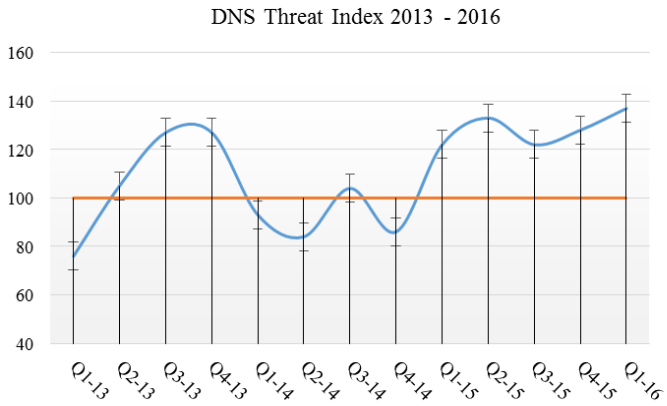

Figure 1. DNS threat index for Q1-2013 to Q1-2016 [3]

The fact that most firewalls and IDSs do not inspect DNS traffic [4] motivates attackers to exploit DNS as a conduit to tunnel network traffic with very little detection if any at all. However, transferring data via DNS is a slow process and therefore a use of DNS for such a task would require a long persistent on the target host. This requirement fits in well with Advanced Persistent Threat (APT) attackers. APT attackers are highly skilled technical cyber threat actors who maintain a long persistent presence on a target. They seek to remain undetected over long periods of time and tend to hibernate to remain undetected. They laterally traverse the target environment and only launch the attack after several months or even years upon completion of the reconnaissance attack phase [5]. APT attackers use various methods to exfiltrate data which are prone to detection by the IDS that be. Leveraging DNS tunneling for data exfiltration is especially attractive since DNS permitted by default in most IDSs and firewalls which in itself presents a low detection rate. Coupling DNS tunneling with malware-free intrusions would give a high turnover of persistent detected presence. This is so because IDSs base their detection mainly on file signature and process behaviour [6] but the file signature in malware-free intrusion is valid and the corresponding process behavior is sanctioned by the operating system since the target file is a system file covertly used for malicious purposes.

In this paper, we present a malware-free based attack structure which utilizes malware-free intrusion as an infection vector and DNS tunneling as a data exfiltration technique. We endeavor to demonstrate the feasibility of the proposed attack structure and subsequent data exfiltration in the presence of a system IDS. Based on the behavioral characteristics of the attack, we suggest preventative and mitigation measures for all indicators of compromise (IOCs). This is helpful to the network and systems administrator in identifying DNS attacks not only for data exfiltration but other malicious network activities.

The rest of the paper is organized as follows; Section II presents the attack model while the experiment setup and methodology are presented in Section III. Results and the analyses thereof are discussed in Section IV and we conclusion is drawn in Section V.

II. The Proposed Attack Model

We model the proposed attack structure by constructing an attack network based on attack graphs [7], comprising nodes $n_{i,j} \in N$ and arcs $A_{i,j} \in E$, where $\{i, j = 0,1,2, \dots, \mathbb{N}^+\}$. The resultant is a directed acyclic graph (DAG) illustrated in figure 2 below.

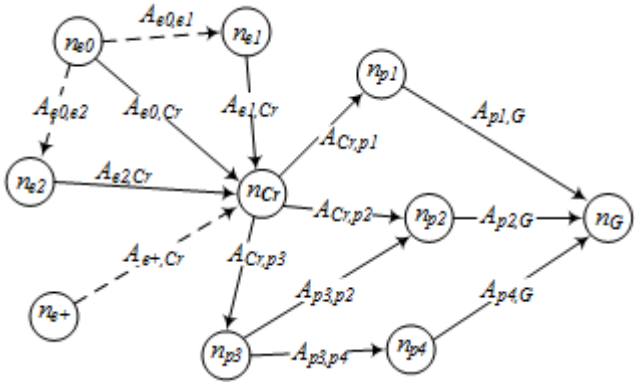

Figure 2. Illustrative attack graph

The nodes $n_{i,j}$ of the attack network are categorized into 4 node sets namely:

- $entry\ nodes: \{n_{e0}, n_{e1}, n_{e2}, \dots, n_{e+}\}$
- $the\ critical\ node: \{n_{Cr}\}$
- $pivot\ nodes: \{n_{p1}, n_{p2}, n_{p3}, \dots, n_{pi}\}$
- $goal\ node: \{n_G\}$

The arcs of the network are directed edges $\{A_{i,j}\}$ denoting traversal from node *i* to node *j*. Since the network structure is a DAG, it follows henceforth that $A_{i,j} \neq A_{j,i}$. We can thus employ Bayesian network statistics to express the likelihood of reaching the goal node n_G given that parent nodes have been accessed prior. *Entry nodes* present types of networks through which the attacker gets network connectivity. Such is not limited to WiFi networks, mobile IP data networks (e.g. Edge, 3G, 4G or 5G), cabled networks etc. Since the threat agent of our model is an APT attacker, we assume that he does not use his own network as an *entry node* to avoid detection.

Upon acquisition of network connectivity, the attacker seeks to use his own DNS whether one is offered by the DHCP server or not. This activity is representative the set of edges $\{n_{e,i} \rightarrow n_{Cr}\}$ denoted by arcs $A_{ei,Cr}$. It's worth noting that even though the arcs $A_{e0,e1}$ and $A_{e0,e2}$ might seem to be of less value since they signify bouncing between networks, such an activity can be employed by an APT attacker to bounce off different IP addresses, a feature synonymous with the Tor network [8]. From the *critical node* n_{Cr}, the attacker uses *pivot nodes* as stepping stones to reach the target host where he intends to exfiltrate data. *Pivot nodes* represent all network artefacts that facilitate access to the target host. After having traversed the relevant *pivot node* to the target, the APT attacker begins data exfiltration via DNS tunneling and this is represented by the end *goal node* n_G.

We assume all necessary preconditions for establishing a DNS tunnel have been implemented on the target host via a malware-free intrusion. In this model, we use a malware-free intrusion from our previous work [9] as an infection vector to implant an agent for facilitating a DNS tunnel.

Scenario I: If all the *pivot nodes* $\{n_{p1}, n_{p2}, n_{p3}, \dots, n_{pi}\}$ need to be traversed in order to actualize n_G, then the probability of exfiltrating data is given as a probability intersection of events $\Pr(n_G)$:

$$\begin{aligned}\Pr(n_G \mid n_{p1}, n_{p2}, n_{p4}) &= \Pr(n_{p1}) \cap \Pr(n_{p2}) \cap \Pr(n_{p4}) \\ &= \Pr(n_{p1}) \cdot \Pr(n_{p2}) \cdot \Pr(n_{p4}) \end{aligned} \quad (1)$$

If any of the probabilities in Equation (1) equals 0, then $\Pr(n_G) = 0$. It's visible from figure 2 that the *critical node* n_{Cr} exerts a decisive influence these values. To capture these relationships between the *critical node* and *pivot nodes*, we construct an adjacency matrix A_G:

$$A_G = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (2)$$

It's apparent from Equation (2) that n_{Cr} is the isthmus of the graph linking entry nodes to pivot nodes and it represents a single point of failure. Therefore, if $\Pr(n_{Cr}) = 0$, then $\Pr(n_G) = 0$ regardless of the values of $\Pr(n_{pi})$. **Scenario II**: However, if $\Pr(n_{Cr}) \neq 0$ and if at least one of the pivot nodes should not equal 0 in order to attain n_G, then the probability is given as a probability union of events $\Pr(n_G)$:

$$\begin{aligned}\Pr(n_G \mid n_{p1}, n_{p2}, n_{p4}) &= \Pr(n_{p1}) \cup \Pr(n_{p2}) \cup \Pr(n_{p4}) \\ &= \Pr(n_{p1}) + \Pr(n_{p2}) + \Pr(n_{p4}) - \Pr(n_{p1}) \cdot \Pr(n_{p2}) \\ &\quad - \Pr(n_{p1}) \cdot \Pr(n_{p4}) - \Pr(n_{p2}) \cdot \Pr(n_{p4}) \\ &\quad + \Pr(n_{p1}) \cdot \Pr(n_{p2}) \cdot \Pr(n_{p4}) \end{aligned} \quad (3)$$

It's vivid from Equation (3) that the attack has a higher probability of success in the disjunction than conjunction scenario. We now endeavor to demonstrate this attack model.

III. Experiment Setup and Methodology

We set up the experiment in a server-client model where the server-side is the attacker's machine and the client-side is the victim's machine. The server runs Kali Linux and we test two clients; Windows 7 and Linux. The test-bed is shown in figure 3 below.

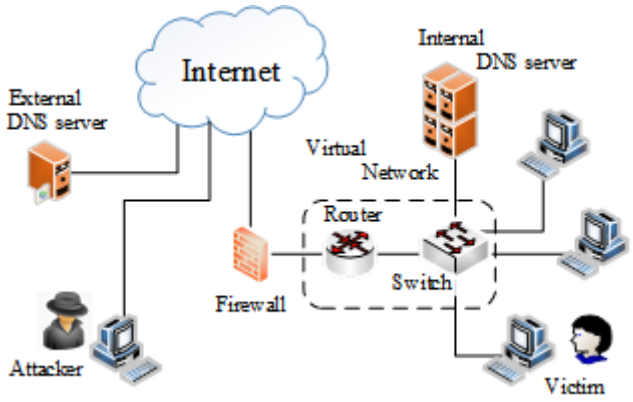

Figure 3. DNS tunneling experiment setup

The attacker has a public IP address whereas the clients have private IP addresses from the local area network (LAN). We use *Dnscat2* [10] for tunnel establishment on the server-side and we implant the client agent on the victim (client-side) using malware-free intrusion [9]. The attacker gains entry to the network, equivalent to accessing any of the entry nodes $\{n_{e0}, n_{e1}, n_{e2}, \ldots, n_{e+}\}$ in the attack model, using a cabled Ethernet connection with a public IP address. On the client, the attacker probes network configurations to determine whether to pursue attack Scenario I or II. The results of the probe are shown in figure 4 below.

```
student@node-00:~$ cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 10.1.1.1
student@node-00:~$ sudo bash
root@node-00:/home/student# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags
0.0.0.0         10.1.1.1        0.0.0.0         UG
10.1.1.0        0.0.0.0         255.255.255.0   U
root@node-00:/home/student#
```

Figure 4. DNS configurations and search hierarchy

The results above show that the LAN uses the default gateway as the DNS server and further, a Wireshark capture reveals that the router actually forwards the DNS requests to the internal DNS server. Furthermore, tracing of the captured network traffic reveals that the internal DNS server forwards un-cached and unknown queries to its forwarders, the ISP DNS servers. The attacker further probes the security policy on the use of unknown DNS servers in the network. This is done by issuing a *dig* query to a Google public DNS server for name resolution of the popular ecommerce site Amazon. The results for this probe are shown in figure 5 below. We leave out the output IP addresses for ethical and security reasons. The results show that it is possible to for a client to use an external DNS server, and this allows us to pursue attack Scenario II [*cf.* Equation (3)]. This implies that the client can query any DNS server on the Internet for name resolution. In our setup, the client communicates with the Dnscat2 server to register its presence. It is after this beaconing to the attacker DNS server app that the initial communication is set up. The client can be set to send *keep-alive* requests since the Dnscat2 cannot reach the client residing on a private LAN without any port-forwarding.

```
$ dig @8.8.8.8 amazon.com

; <<>> DiG [redacted] <<>> @8.8.8.8 amazon.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 48495
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 6, ADDITIONAL: 6

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;amazon.com.                    IN      A

;; ANSWER SECTION:
amazon.com.             35      IN      A       [redacted]
amazon.com.             35      IN      A       [redacted]
amazon.com.             35      IN      A       [redacted]
amazon.com.             35      IN      A       [redacted]
amazon.com.             35      IN      A       [redacted]
amazon.com.             35      IN      A       [redacted]

;; Query time: 340 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; MSG SIZE  rcvd: 388
```

Figure 5. DNS query probe using external DNS server

Attack Scenario I [*cf.* Equation (1)] is pursued under environments with stringent DNS network security controls. Such include the requirement to only use internal DNS servers for name resolutions, proxy configurations for Internet access on LAN hosts, use of captive portals on WiFi networks, use of only trusted forwarders apart from those listed by the ISP etc. These requirements would require further attack actions from the attacker which is equivalent to traversing *pivot nodes* prior to establishment of a session with the Dnscat2 server.

Having established the pre-conditions necessary for launching attack Scenario I, the attacker proceeds to establish a DNS tunnel and subsequently exfiltrate target data. DNS tunnel establishment and the associated data exfiltration are elaborated in the proceeding section.

IV. Attack Results and Analysis

Our attack structure is based on malware-free intrusions, i.e. the attacks utilize system resources and the flawed configurations thereof to exploit a vulnerable host. In light of this, we employ the use of PowerShell [11] in Windows to query the attacker's server instead of using a stand-alone client. This has the advantage of the increasing the attacker's stealthiness in that virtually no malware is implanted which in turns evades the IDS that be. We load the script [12] into PowerShell, as shown in figure 6 below, which enables the victim to communicate with the Dnscat2 server.

After having initialized the script as shown below, we pass arguments specifying the IP address of the Dnscat2 server which resided on the public Internet and the connection was successful.

The PowerShell establishes a DNS session on port 53 as though it were querying a name resolution. Since DNS requests are rarely counter-checked as was in our test network, the established connection is transparent to most applications.

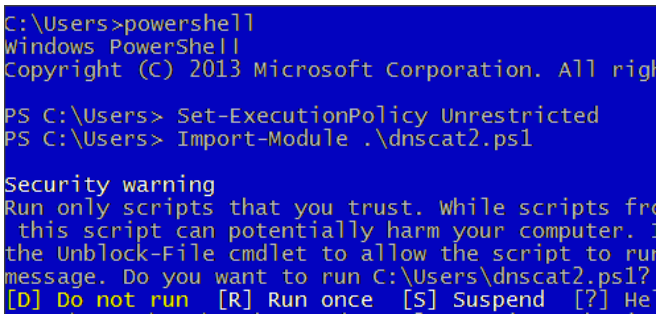

Figure 6. Exploitation of PowerShell for tunneling client

We also ran a terminal script on a Linux victim machine and the connection was successful on port 53 as shown in figure 7 below. Both victim clients connected to the same public IP address hosting the Dnscat2 service. Further Dnscat2 establishes a secure communication channel between the parties as shown in the figure below with a passphrase for verification and any sniffing of the exfiltrated traffic yields nothing meaningful. DNS uses the provided record types to evaluate the requested service and there are 83 records types as per IANA specification [13]. It's visible from the output of the established session that the records in use at TXT, CNAME and MX records. The TXT record, also known as the Sender Policy Framework (SPF) stores text strings with the most common being the domains of the valid email address senders of a given domain and IP address [14]. The CNAME (Canonical Name) record is used as an alias to an A record which is essentially a mapping of a given hostname and domain to an IP address for a valid forward lookup whereas the PTR record is responsible for the opposite, reverse lookup [15]. The MX records provides mapping for mail exchanger records, those responsible for handling the emailing service of the specified domain [16].

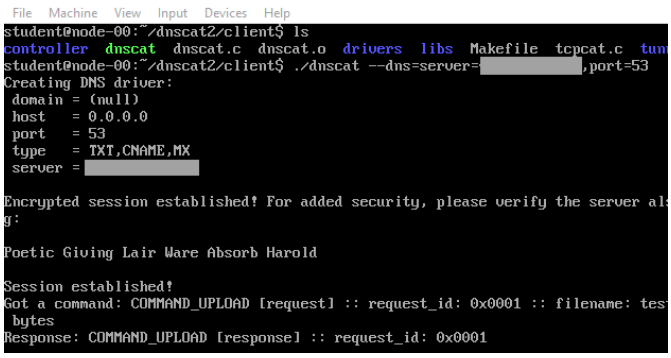

Figure 7. Dnscat2 client session on Linux victim

These are records which an administrator would expect to see on the network and it is the more reason why an adversary such as Dnscat2 would have them as default configuration settings. The other common record types include NS (Name server) and SOA records. The other types are not common and would but not always imply IOCs if present in DNS traffic logs. The figure 8 below shows the initialization of the Dnscat2 server which runs a daemon listening for incoming connections from client. It's worth noting that the server explicitly specifies that all connections are encrypted. The cryptographic suite used in this case includes Salsa 20 [17]. SHA3 and ECDH for symmetric key generation while the SHA3 together with Salsa 20 are used for signing and encryption. The short authentication strings used as passphrases need to match on both ends of the connection or else the connection will be terminated. This tries to proof against Man-In-The-Middle (MITM) attacks [18]. A more resilient form authentication involves the use of use of a pre-shared secret. Similarly, a mismatch in the pre-shared key leads to rejection of the connection. For our demonstration, we go with the default generated short authentication string "*Poetic Giving Lair Ware Absorb Harold*" which matches on both the client and server-side.

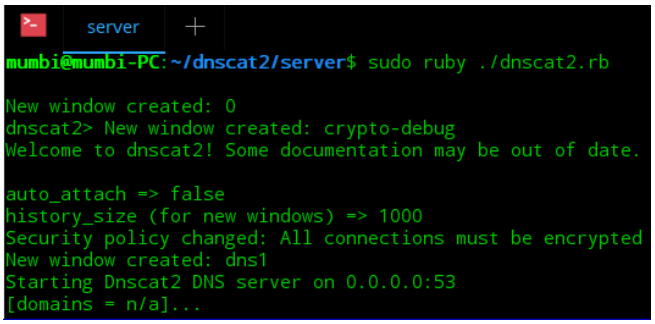

Figure 8. Initialization of the tunneling server

After successfully establishing a DNS tunnel between the client and the server, we go ahead to pass a small file through the tunnel. This is equivalent to exfiltrating the data itself. Tunneling via DNS cannot support large files, therefore if the target file is big, it has to be truncated and transferred in parts. Figure 9 below shows successful file transfer through the tunnel, where the target file is "*testing.txt*".

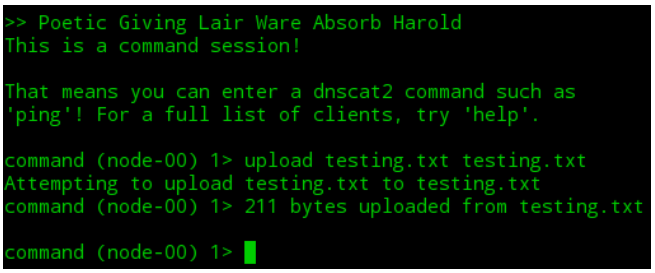

Figure 9. Successful file transfer through the tunnel

We also run Wireshark to capture the traffic traversing the network via this tunnel. We notice the use of the 3 record types (TXT, CNAME and MX) earlier seen during the tunnel establishment phase. These queries are prepended by a tag and the word "*dnscat*" as shown in figure 10 below. All these queries as seen as standard DNS requests by the sniffer despite the unusual long sub-domain name. The queries are tracked by Query IDs (QID) in Hex for both directions of traffic flow.

```
101 Standard query 0x66af CNAME dnscat.2d5101574b1f72b87c3b9600f49063d601
156 Standard query response 0x66af CNAME dnscat.2d5101574b1f72b87c3b9600f49063d601 CNA
101 Standard query 0x22df MX dnscat.22ab01574bb4b57297a19200f57d79a108
158 Standard query response 0x22df MX dnscat.22ab01574bb4b57297a19200f57d79a108 MX 10
101 Standard query 0x3a44 CNAME dnscat.575a01574ba30b243e213b00f667405888
156 Standard query response 0x3a44 CNAME dnscat.575a01574ba30b243e213b00f667405888 CNA
101 Standard query 0x3bf8 MX dnscat.5b9401574babeb74ee69e800f7002d2f5c
158 Standard query response 0x3bf8 MX dnscat.5b9401574babeb74ee69e800f7002d2f5c MX 10
101 Standard query 0x06c0 MX dnscat.6cc701574b31ef57286cc900f89c6b7e38
158 Standard query response 0x06c0 MX dnscat.6cc701574b31ef57286cc900f89c6b7e38 MX 10
101 Standard query 0x22f3 TXT dnscat.46a801574bea804a8e227c00f921e5b411
148 Standard query response 0x22f3 TXT dnscat.46a801574bea804a8e227c00f921e5b411 TXT
101 Standard query 0x4ea4 TXT dnscat.653501574b68323551a9eb00fac9e03938
148 Standard query response 0x4ea4 TXT dnscat.653501574b68323551a9eb00fac9e03938 TXT
101 Standard query 0x0d64 TXT dnscat.2b1301574b2816b37cd95100fb2161aebf
148 Standard query response 0x0d64 TXT dnscat.2b1301574b2816b37cd95100fb2161aebf TXT
 60 Who has 192.168.87.1? Tell 192.168.87.128
 42 192.168.87.1 is at 00:50:56:c0:00:08
101 Standard query 0x7984 TXT dnscat.525901574be1d33bb1f66a00fc3409355b
148 Standard query response 0x7984 TXT dnscat.525901574be1d33bb1f66a00fc3409355b TXT
101 Standard query 0x4b9f TXT dnscat.394601574b71d5a9737b6700fd1b68d13
```

Figure 10. DNS tunnel traffic capture

The network interface used for the capture listened promiscuously to all traffic hence the presence of packets from other subnets. It's worth noting from the above that DNS supersedes other network requests after initialization of the tunnel and subsequent file transfer. This is against the normal backdrop where DNS requests constitute a small portion of the overall network traffic. We generate network traffic graphs both before and after the creation and use of the DNS tunnel. The diagram in figure 11 below shows observed normal network traffic with DNS (red graph) present in small quantities before establishment of the tunnel.

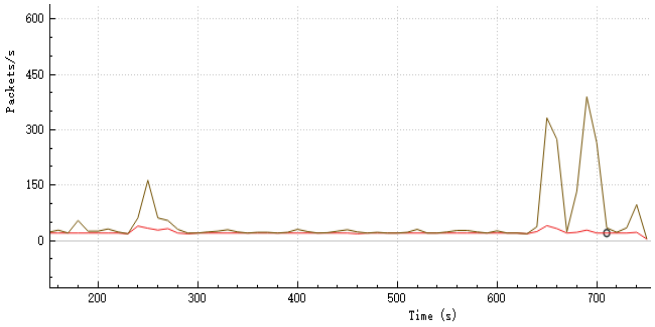

Figure 11. Normal network traffic without DNS tunneling

The DNS traffic is shown in red and is generally at par with the overall network activity. However, after creation of the DNS tunnel and subsequent file transfer, there's a huge spike in DNS requests. This is unusual of DNS traffic and the fact that the connections thereof are encrypted is a strong implication of IOCs. Figure 12 below shows the logarithmic graph of DNS requests after establishment of the tunnel. From figure 10, it is clear that not only is the entropy high of the generated domains by the Domain Generating Algorithm (DGA) [19] but the size of the string as well. Furthermore, the number of generated sub-domains is equally high which is another implication of IOCs. There are three other noticeable characteristics from the DNS tunneling traffic logs;

- *volume of DNS traffic from a single IP address*: All the encrypted DNS queries from the traffic logs came from a single IP address which did not issue any other types of requests other than DNS.
- number of hostnames per domain: the number of host for the domain was unexceptionally high. It is interesting to note that the number of generated hostnames was way higher than the number of IP addresses support by the subnet of the Dnscat2 server.
- amount of DNS traffic for that domain: the traffic to the target domain was more than all types of requests for that specified time period.

Though it is possible for the attacker to easily set up multiple domains with the advent of cloud computing, we still contend that even if traffic is spread across multiple domains, the volume will still be noticeable.

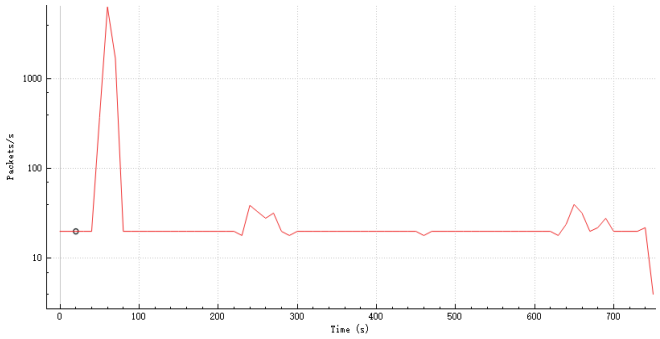

Figure 12. DNS traffic with tunneling

It is important to pay attention to DNS traffic on the network due to the fact that it is one of the few protocols permitted through firewalls and IDs on the majority of networks. Traditionally DNS tunneling has been used for command and control [20] [21] but recent times has seen it being employed in various data exfiltration techniques by more pronounced APT attackers [22] [23]. Though our attack methodology involved the importation of a script from the victim's machine, an attacker can upload the script directly from the network as demonstrated in malware-free intrusions [24]. Therefore, mitigation techniques cannot solely depend on IDS and firewalls. There is need to use objective behaviour analysis of network traffic from time to time with logging where possible. It is not feasible to thwart all DNS tunneling attempts due to the open nature of the DNS structure headachy. However, some possible mitigation measures include limiting the number and types of DNS servers that host from a network are able to query. The internal DNS should be set as the DNS server to forward request to other servers. Monitoring anomalies in DNS requests particularly the aforementioned IOCs. This could include isolating a host that is issuing excessive encrypted DNS requests to a specific domain, and as a consequence to also blacklist such suspected domains. Hostnames with a very high entropy and unusual length ought to be investigated too. It is important to note that during mitigation, an administrator might be dealing with attack Scenario I or attack Scenario II in which case the mitigation approach would be different. An intuitive mitigation method in attack Scenario II is to consider the geographic location of the domain and the nature of business

the domain engages in. This would call for more than just normal LAN traffic analysis.

V. CONCLUSION

In this paper, we have demonstrated a novel way of exfiltrating data from a targeted host using system resources and not any extra malware. This is especially attractive to APT threat actors who seek to maintain a long undetected presence. The results show that the attack proceed without detection as per constraints of the attack model. Most attacks use DNS tunneling for command and control activities but the advent of malware-free intrusions extends the applications. One of the sure way of thwarting this type of attack is getting rid of the client agent on the victim host but in the case of Windows, it might be a little daunting considering that PowerShell can be utilized to inject client code in form of a script to facilitate tunneling. Attacks on DNS are prone to increase as attackers are looking for new ways to compromise systems. This is due to the fact that no one DNS server can resolve all the hostnames on the Internet. Therefore mitigation measures should not only be left to signature based detection but behavioral based considering that new tunneling tools are emerging with new signatures altogether.

REFERENCES

[1] Vixie, Paul. "DNS complexity." Queue 5, no. 3 (2007): 24-29.

[2] DNS Threat Index. [Online] https://www.infoblox.com/solutions/network-security/dns-threat-index/

[3] Infoblox DNS Threat Index Q1 Report .Know the Threat Level of DNS-Based Malware. [Online] https://www.infoblox.com/wp-content/uploads/infoblox-white-paper-dns-threat-index-q1-2016-report.pdf

[4] Wool, Avishai. "A quantitative study of firewall configuration errors." Computer 37, no. 6 (2004): 62-67.

[5] Kao, Da-Yu. "Performing an APT Investigation: Using People-Process-Technology-Strategy Model in Digital Triage Forensics." In Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual, vol. 3, pp. 47-52. IEEE, 2015.

[6] Depren, Ozgur, Murat Topallar, Emin Anarim, and M. Kemal Ciliz. "An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks." Expert systems with Applications 29, no. 4 (2005): 713-722.

[7] Sheyner, Oleg, Joshua Haines, Somesh Jha, Richard Lippmann, and Jeannette M. Wing. "Automated generation and analysis of attack graphs." In Security and privacy, 2002. Proceedings. 2002 IEEE Symposium on, pp. 273-284. IEEE, 2002.

[8] McCoy, Damon, Kevin Bauer, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. "Shining light in dark places: Understanding the Tor network." In International Symposium on Privacy Enhancing Technologies Symposium, pp. 63-76. Springer, Berlin, Heidelberg, 2008.

[9] Aaron Zimba, Zhaoshun Wang,"Malware-Free Intrusions: Exploitation of Built-in Pre-Authentication Services for APT Attack Vectors", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.7, pp.1-10, 2017.DOI: 10.5815/ijcnis.2017.07.01

[10] Buczak, Anna L., Paul A. Hanke, George J. Cancro, Michael K. Toma, Lanier A. Watkins, and Jeffrey S. Chavis. "Detection of Tunnels in PCAP Data by Random Forests." In Proceedings of the 11th Annual Cyber and Information Security Research Conference, p. 16. ACM, 2016.

[11] Holmes, Lee. Windows PowerShell Cookbook: The Complete Guide to Scripting Microsoft's New Command Shell. " O'Reilly Media, Inc.", 2010.

[12] Luke Baggett. Dnscat2 PowerShell Script. [Online] https://github.com/lukebaggett/dnscat2-powershell

[13] (DNS) Parameters - Internet Assigned Numbers Authority. [Online] https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml

[14] Schlitt, Wayne, and Meng Weng Wong. "Sender policy framework (SPF) for authorizing use of domains in e-mail, version 1." (2006).

[15] Jung, J., Sit, E., Balakrishnan, H. and Morris, R., 2002. DNS performance and the effectiveness of caching. IEEE/ACM Transactions on networking, 10(5), pp.589-603.

[16] Larson, M., Massey, D., Rose, S., Arends, R. and Austein, R., 2005. Resource records for the DNS security extensions. Resource.

[17] Bernstein, D.J., 2008. The Salsa20 family of stream ciphers. Lecture Notes in Computer Science, 4986, pp.84-97.

[18] Nayak, G.N. and Samaddar, S.G., 2010, July. Different flavours of man-in-the-middle attack, consequences and feasible solutions. In Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on (Vol. 5, pp. 491-495). IEEE.

[19] Zhou, Y., Li, Q.S., Miao, Q. and Yim, K., 2013. DGA-Based Botnet Detection Using DNS Traffic. J. Internet Serv. Inf. Secur., 3(3/4), pp.116-123.

[20] Dietrich, C.J., Rossow, C., Freiling, F.C., Bos, H., Van Steen, M. and Pohlmann, N., 2011, September. On Botnets that use DNS for Command and Control. In Computer Network Defense (EC2ND), 2011 Seventh European Conference on (pp. 9-16). IEEE.

[21] Xu, K., Butler, P., Saha, S. and Yao, D., 2013. DNS for massive-scale command and control. IEEE Transactions on Dependable and Secure Computing, 10(3), pp.143-153.

[22] ProjectSauron: top level cyber-espionage platform covertly extracts encrypted government comms. [Online] https://securelist.com/faq-the-projectsauron-apt/75533/

[23] Tankard, C., 2011. Advanced persistent threats and how to monitor and deter them. Network security, 2011(8), pp.16-19.