DNS Tunneling Detection Using Elasticsearch

Ahmad Faisal Sani
Department of Informatics
Universitas Islam Indonesia
Yogyakarta
INDONESIA
14917133@students.uii.ac.id

Mukhammad Andri Setiawan
Department of Informatics
Universitas Islam Indonesia
Yogyakarta
INDONESIA
andri.setiawan@uii.ac.id

Abstract — Domain Name System (DNS) Protocol is a popular medium used by malware to perform 'command and control' in taking over victim's computer, this technique called as DNS tunneling. Moreover, DNS tunneling can also be used to bypass captive portal hotspot in public places and worsen the network quality. However, in more dangerous stage, DNS tunneling can also be used to exfiltrate data from the victim's computer. Instead of using DNS Protocol to translate domain name, the medium misused to bootleg the data. Those are the weaknesses which frequently used by the attacker to deceive network administrator. Our approach to this problem is analyzing the traffic using unique hostname as indicator of compromise and utilizing Elasticsearch tool to detect DNS tunnelling. Elasticsearch will send an email to notify the administrator about DNS tunneling. The email contain information about domain suspected as perpetrator of DNS tunneling. The result from Elasticsearch can be use to add the domain blacklist, so the domain can no longer be use to perform DNS tunneling. Hopefully those combinations are able to support the network administrator to secure the network from DNS tunneling. Moreover, the result of network quality analysis shows that there is a rise in jitter value and packet lost when DNS tunneling happens.

Keywords — DNS tunneling, data exfiltration, traffic analysis, elasticsearch.

I. INTRODUCTION

Recently, Internet Security Researchers found a malware diversified as Remote Access Trojan. This kind of malware uses a new technique which is hard to detect by the Network Administrator. Those malware does 'command and control' and takes over the victim's computer through DNS Protocol, this kind of malware called DNSMessenger [1]. 'Command and Control' or sending data through DNS Protocol called DNS tunneling or DNS ex-filtration. Besides doing 'command and control', the malware uses DNS tunneling to bypass the captive portal or login hotspot in public places [2]. However, data thievery through DNS Protocol considers being more dangerous than bypassing captive portal. The malware could ex-filtrate various data, such as classified trade data, intellectual property, employee data, customer data, and many others. DNS tunneling requires software installed on the victim's computer to work. If the software installed, the attacker can easily bypass the firewall and security system of the victim.

DNS tunneling can also worsen network quality. The research concluded that using DNS tunneling can increase the delay of the entire network up to 140-1500 ms, jitter until 8-57 ms, and DNS Overhead 200-2000% [3]. We can imply that; a client on the network is making a VOIP call or video call, the high jitter causes a voice and video transmission can not run smoothly, the high delay causes a slight or minor delay to the voice and video. DNS Overhead causes an increase in data package size and resulting in higher using of bandwidth. To experience the best quality of the network, Cisco recommends we set the jitter below 30 ms and delay less than 150 ms [4].

Network Administrator usually does not give extra attention to the DNS traffic. DNS Protocol is a protocol uses to translate IP Address into a domain name, with the result that we can access a computer or server using the name without remembering the IP Address of the computer or server. With that common sense, the Administrator is forgetting about the fact that DNS Protocol also can be used to exchange data. That flaw uses by the attacker to 'Command and Control' or steal data using DNS tunneling.

Therefore, DNS traffic in a network should be monitored to prevent DNS tunneling. Actually, the Administrator able to block all DNS traffic to prevent tunneling. However, that is not the ideal solution, because that method will also block the user to acces host address. Another approach is using DNS Sinkhole [5]. DNS Sinkhole is a DNS server which able to give wrong IP address (spoofing) from DNS request, so the destined domain can no longer be accessed. This condition can be use to prevent malware or DNS tunnel from contacting the server.

DNS Sinkhole is using a domain list to be blocked. We can manually make the list or download it from website such as; urlblacklist.com, malwaredomain.com, and etc. To obtain the domain suspected doing DNS tunneling is necessary to monitor and log all DNS traffic in the network. Those log can be obtain from many resources, such as DNS server, Intruder Detection System (IDS), proxy, and computer log. To detect DNS tunneling from the log, analysis should be done manually using capture analyzer packet, such as Wireshark. This kind of approcach considered hard to do and take times,

especially if we want to visualize the result, we need another tools. Another approach is using Payload Analysis Method and traffic analysis [2]. Payload analysis able to detect certain DNS tunneling, while traffic analysis able to detect DNS tunneling universally.

Traffic analysis is the approach we use to resolve the issues we stated before. We use traffic analysis with the amount of unique hostname as an indicator of compromise using Elasticsearch. Elasticsearch has components which can be used in this research, such as Packetbeats, Kibana, and Watcher. Packetbeats is a real-time sniffer which will capture the traffic DNS, Watcher will give an email notification when DNS tunneling happened, and Kibana is a panel of visualization which will show a graphic bar of domain names which have the most unique hostname. That combination hopefully helps the administrator to secure and monitor the network.

II. Literature Review

DNS tunneling is a technique to bypass the security control and to infiltrate or ex-filtrate data from a target. This technique is still used because DNS usually do not monitor well. The practitioner blindly trusted DNS is secure [6].

Popular tools to perform DNS tunneling are Iodine [7] and Dnscat2 [8]. Iodine is an app which able to create a tunnel interface between client and server, all the traffic can be passed up through the tunnel. On the other hand, Dnscat2 used for performs 'command and control' between client and server. Both apps can bypass the security control of a network.

DNS has a caching mechanism to accelerate the response from DNS query, therefore, all DNS Tunnel program will create random and long hostname string (unique hostname), so that DNS cannot cache the tunnel and the data thievery become possible.

DNS tunneling detection methods divided into two, which is, payload analysis and traffic analysis. Payload Analysis can only detect certain DNS tunneling, while Traffic Analysis can detect DNS tunneling universally. Farnham tries to detect DNS tunneling using Traffic analysis making unique hostname as an indicator of compromise. The normal amounts of unique hostnames are below 300, after conducting DNS tunneling the amount of unique hostname rapidly increase until 700 [2].

A tool which can be used to detect DNS tunneling with traffic analysis method is Elasticsearch. Elasticsearch is a search engine which builds the base on Apache Lucene and opensource product which also developed with Java. Elasticsearch can conduct a real-time and distributed analysis and also able to do multiple searching mechanism. Elasticsearch can manage various kinds of logs, such as operating system log, web server, log traffic, app log, and Amazon Web Service Log [9].

III. Research Methodology

A. Design and Topology

To support this research, the researcher will build a lab in a virtual neighborhood so that easier and cheaper to maintain. Though only in a virtual neighborhood, the condition already represents the real condition in a real network. The researcher will use a computer with Quad-Core processor, RAM 8GB, and SSD 128GB. The host run Ubuntu 19.10 and using Virtualbox with Hypervisor KVM as the virtualization software.

In the VirtualBox will be built a virtual machine. The virtual machine will run Elasticsearch. The Virtual machine will run with the specification of 2 virtual core, 4GB RAM, network mode bridge, and run CentOS 7.

For the tunneling server, the researcher using a VPS with CentOS7, RAM 1GB, one virtual core CPU. Iodine and DNScat2 used as DNS tunneling software.

The researcher uses Mikrotik RB750 as the router. The router will be used as a gateway and firewall. For the client, the researcher will use Windows 10 and Ubuntu 19.10.

The topology used by the researcher can be seen in an image below. That topology was chosen because it represents the real condition of a network. On of the client computer will be the DNS tunnel client which call the DNS tunnel server. The router will be set to do mirroring port to Elasticsearch server, with that scheme, all the traffic can be read by Elasticsearch to run the inspection.

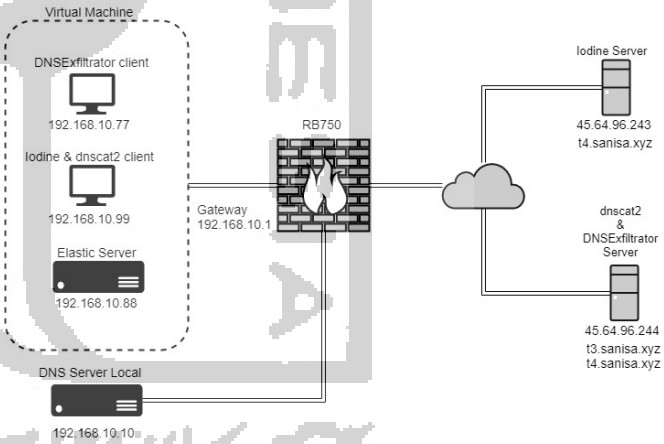

Fig. 1. DNS tunneling simulation topology

B. Log aggregation and simulation.

Log aggregation will be done by Elasticsearch using a plug-in. The plug-in will capture a packet called Packetbeat in real-time. So the data used is adequate, DNS Grind from Pentestmonkey is used to generate traffic simulation [10].

DNS tunnel server Iodine and ~~DNSCat2~~DNScat2 will run the services in DNS tunnel server. The client will also run the DNS tunnel client which will contact the DNS tunnel server.

On the first experiment, the client will browse uses DNS tunnel, this experiment intends to create DNS tunnel log.

Elasticsearch will run the job which will detect DNS tunnel from the collected log. Visualization of the result will be seen on Kibana Dashboard. Elasticsearch will detect and count the amount of unique hostname, domain with the largest amounts of unique hostname will detect as an anomaly. To get the best result, we need more or less 48 hours.

C. Detection and analysis

In this stage, an analysis will be conducted to find out whether Elasticsearch able to detect DNS tunneling or not. The method applied is Traffic Analysis. Each communication on DNS tunneling will create a new hostname, the normal average amount of unique hostname is below 300 [2], because of that, the more unique hostname indicates DNS tunneling is happening. All logs captured by Packetbeat will be processed with a custom script by the Watcher. After that, Watcher counts the amount of unique hostname based on the cardinality of the domain. The amount of unique hostname on a domain will be visualized in Kibana in the form of a graphic bar. If the amount of unique hostname more than 300 and the domain does not exist in the whitelist, the watcher will send an email to the administrator.

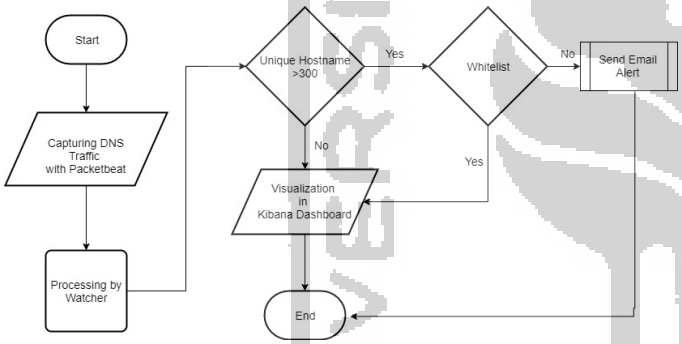

Fig. 2. Flowchart detection of dns tunneling.

IV. RESULTS

Mikrotik router will do a port mirroring to duplicate packets from DNS server to Elasticsearch server. The Elasticsearch server will run Packetbeat app to sniff the DNS packet. A Laptop will be prepared to run the DNS tunneling to the server.

Watcher will find cardinality from the hostname of each domain and come out with the amount of unique hostname. Here is a graphic from Kibana dashboard shows the amount of unique hostname for 15 minutes without DNS tunneling.

Fig. 3 shows the amount of unique hostname in normal situation is below 100.

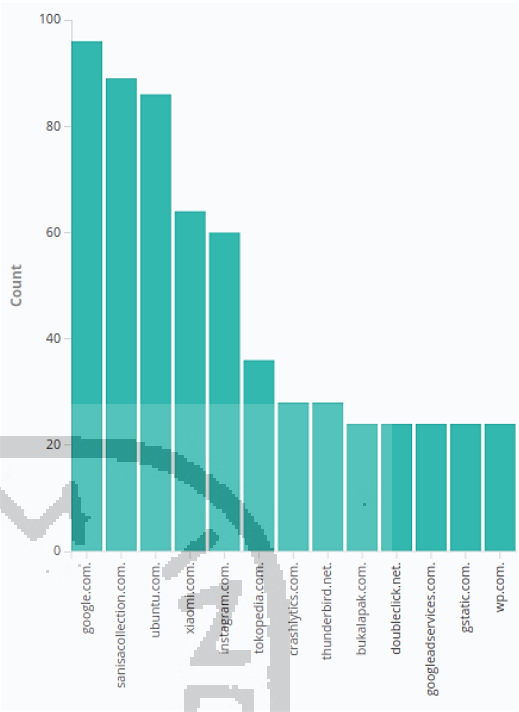

Fig. 3. Unique hostname on DNS with normal traffic.

In this research we use Iodine, Dnscat2, and malware DNSExfiltrator. We choose those tools because already represent several methods in DNS tunneling.

Tunneling using Iodine

In this first experiment, we use Iodine to perform DNS tunnel. We try to browse through DNS tunnel.

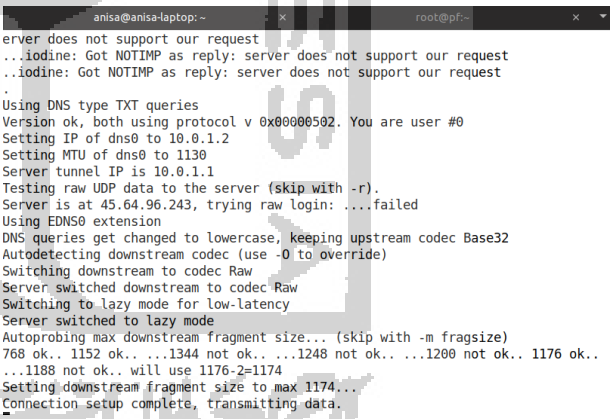

Fig. 4. Tunneling process using Iodine

The domain name which the researcher uses as DNS tunnel server is sanisa.xyz. When the tunnelling runs for 3 minutes, the amounts of unique hostname on that domain spike up until 700.

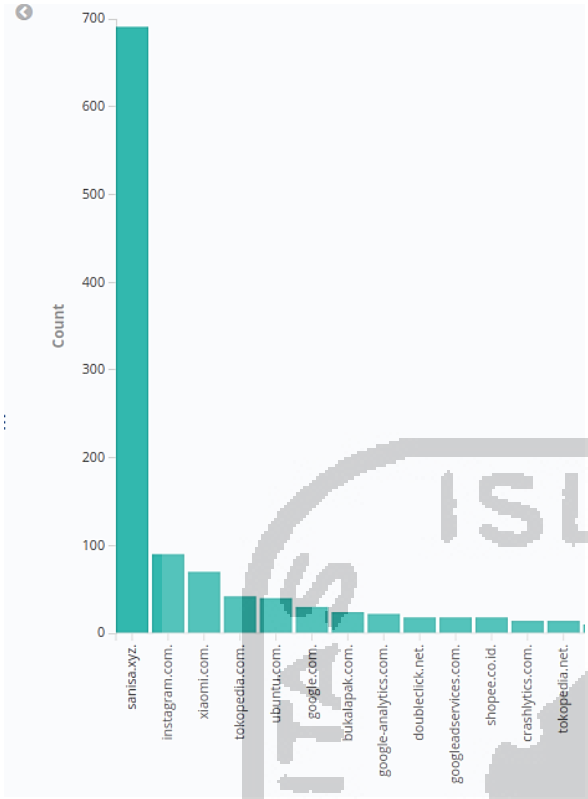

Fig. 5. The spike on unique hostname when tunneling using Iodine on domain sanisa.xyz

Tunneling using dnscat2

In the next experiment, we use Dnscat2 to perform 'command and control' on domain sanisa.xyz

```
root@t3:~/dnscat2/server# ruby dnscat2.rb --dns "domain=t3.sanisa.xyz,host=45.
64.96.244" --no-cache

New window created: 0
dnscat2> New window created: crypto-debug
Welcome to dnscat2! Some documentation may be out of date.

auto_attach => false
history_size (for new windows) => 1000
Security policy changed: All connections must be encrypted
New window created: dns1
Starting Dnscat2 DNS server on 45.64.96.244:53
[domains = t3.sanisa.xyz]...

Assuming you have an authoritative DNS server, you can run
the client anywhere with the following (--secret is optional):

  ./dnscat --secret=9f85eb3604a88478b9686f4be4dd3c81 t3.sanisa.xyz

To talk directly to the server without a domain name, run:

  ./dnscat --dns server=x.x.x.x,port=53 --secret=9f85eb3604a88478b9686f4be4dd3
c81

Of course, you have to figure out <server> yourself! Clients
will connect directly on UDP port 53.


dnscat2> 
```

Fig. 6. Tunneling process using dnscat2

When the dnscat2 runs for 3 minutes, the amount of unique hostname on domain sanisa.xyz escalates until 900.

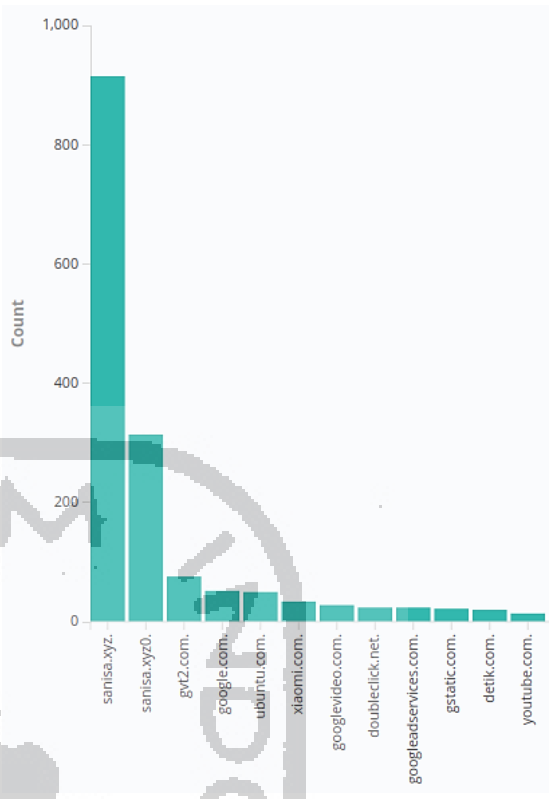

Fig. 7. The escalation of unique hostname when tunneling using dnscat2 on domain sanisa.xyz.

Tunneling using malware

In this experiment, the researcher using a malware called *DNSExfiltrator*. That malware listed in virustotal.com with hash value:

"*ed937bcd5dc05f1021aa83afdb47af266083ef47228e23a32292bad577c53191*".

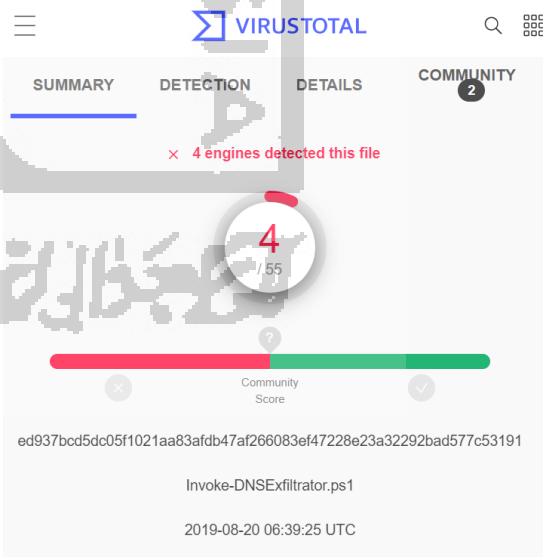

Fig. 8. Malware DNSExfiltrator status based on virustotal.com

This malware can send a file through DNS protocol. On the side of the server, this malware uses python language, but on the side of the client (the victim), we use *powershell* windows. On this trial, we send a file with a name "data.pdf" file size: 685KB to the server with domain t4.sanisa.xyz.

```
C:\Users\unknown-win\Documents\dnsex>powershell -c "ipmo .\Invoke-DNSExfil
-d t4.sanisa.xyz -p faisalganteng

Security warning
Run only scripts that you trust. While scripts from the internet can be us
computer. If you trust this script, use the Unblock-File cmdlet to allow th
message. Do you want to run C:\Users\unknown-win\Documents\dnsex\Invoke-DN
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
[*] Compressing (ZIP) the [data.pdf] file in memory
[*] Encrypting the ZIP file with password [faisalganteng]
[*] Encoding the data with Base64URL
[*] Total size of data to be transmitted: [835516] bytes
[+] Maximum data exfiltrated per DNS request (chunk max size): [226] bytes
[+] Number of chunks: [3697]
[*] Sending 'init' request
[*] Sending data...
```

Fig. 9. DNS tunneling process using DNSExfiltrator to send a file data.pdf

When the DNS tunnelling process runs, there are escalations of unique hostname until more than 5000 as shown in the graphic below:

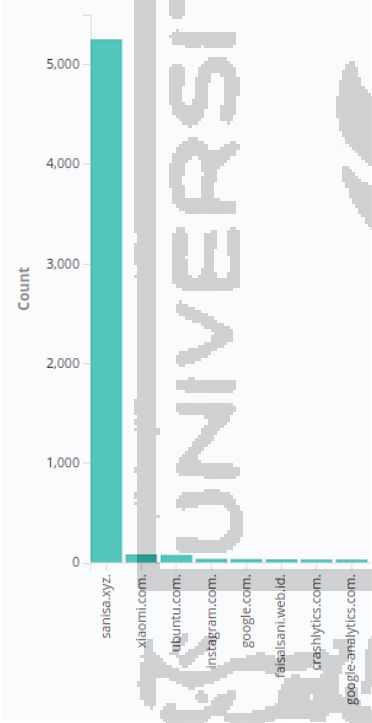

Fig. 10. The escalation of unique hostname when tunnelling using DNSExfiltrator on domain sanisa.xyz.

To assure that Elasticsearch successfully detect DNS tunneling, during the process of tunneling we do capture packet using Wireshark.

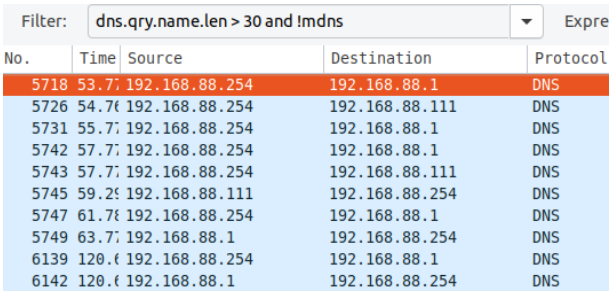

Fig. 11. Wireshark process to sniff the DNS traffic.

To analyze the DNS traffic using wireshark, the researcher use filter *"dns.qry.name.len > 30 and !mdns"* which means, DNS query filter with the amount of subdomain/hostname more than 30 and is not multicast DNS. The filtering result shows there are long query with encoding base64 on domain sanisa.xyz.

Fig. 12. Pict 11. The result of filtering Query DNS on Wireshark.

In a journal written by Leijenhorst, DNS tunneling can also worsen the network quality [3]. Therefore in this research we also analyzing jitter value and packet loss in the UDP protocol. We would like to find out the effect of DNS tunneling to the network. We provide to clients, the first one perform DNS tunneling, the second one is normal client. Both clients run in one network. In this experiment we use iperf3. And the result as shown below:

TABLE 1. NETWORK QUALITY ANALYSIS

Client	Parameter	Before Tunneling	On Tunneling
Client Running DNS Tunnel	Jitter	0.792 ms	3.726 ms
	Packet Loss	0 %	36 %
Client Not Running DNS Tunnel	Jitter	0.15 ms	1.096 ms
	Packet Loss	0 %	1.7 %

From the experiments we can conclude that there is a rise of jitter value and packet loss. The rise happens significantly on the client who did DNS tunneling. On the other client who did not perform DNS tunneling, the rise still happen but not significant, still acceptable according to Cisco recommendation [4].

Notification to the Administrator

When the unique hostname more than 300 and the domain are not in the whitelist, the watcher will be triggered and send an email notification about suspicious activities which indicated DNS tunneling. An email notification is sent to Administrator, with the contents:

"This Domain has high unique hostnames:{sanisa.xyz.={unique_hostnames=661,total_bytes_in=371114.0,total_bytes=1490168.0,total_requests=2024, total_bytes_out=1119054.0}}"

The email notifies the administrator that the domain sanisa.xyz suspects of doing DNS tunneling. From the experiments, Elasticsearch, Packetbeat, and Watcher can be used to detects DNS tunneling.

V. CONCLUSIONS AND RECOMMENDATIONS

In our experiments, we conducted our simulation with Iodine, dnscat2, and executing DNSExfiltrator malware. We can conclude that traffic analysis in a way of counting unique hostname as an indicator of DNS tunneling with Elasticsearch is successfully detecting the DNS tunneling and able to notify the administrator about the attacker. The output from the detection can add up the list of blacklisted domain. On top of that, on the network quality analysis, DNS tunneling can increase jitter value and packet loss on the network but not significant and still acceptable.

REFFERENCES

[1] E. Brumaghin and C. Grady, "Talos Blog || Cisco Talos Intelligence Group - Comprehensive Threat Intelligence: Covert Channels and Poor Decisions: The Tale of DNSMessenger," 2017. [Online]. Available: https://blog.talosintelligence.com/2017/03/dnsmessenger.html. [Accessed: 03-Apr-2019].

[2] G. Farnham, "Detecting DNS tunneling," 2013.

[3] T. Van Leijenhorst, K. Chin, and D. Lowe, "On the Viability and Performance of DNS tunneling," 2008.

[4] "Acceptable Jitter and Latency." [Online]. Available: https://getvoip.com/blog/2018/12/20/acceptable-jitter-latency/. [Accessed: 29-Jun-2019].

[5] G. Bruneau, "DNS Sinkhole," 2010.

[6] M. Branscombe, "Why you need to care more about DNS." [Online]. Available: https://www.cio.com/article/2948378/why-you-need-to-care-more-about-dns.html.

[7] "Iodine DNS tunneling Tools." [Online]. Available: https://code.kryo.se/iodine/.

[8] "DNScat2 C&C tunneling tools." [Online]. Available: https://github.com/iagox86/dnscat2.

[9] Elasticsearch, "Elasticsearch: RESTful, Distributed Search & Analytics | Elastic." [Online]. Available: https://www.elastic.co/products/elasticsearch. [Accessed: 03-Apr-2019].

[10] "DNS Grind." [Online]. Available: https://github.com/pentestmonkey/dns-grind.