

Technologies for Detecting Malicious Requests in Computer Networks Based on the DNS Protocol

Olga Leshchenko, Oleksandr Trush, Mariya Trush, Andrii Horachuk and
Oleksandr Makhovych

Taras Shevchenko National University of Kyiv, Volodymyrs'ka str. 60, Kyiv, 01033, Ukraine

Abstract

The work provides basic information about technologies and methods of detecting malicious requests in computer networks based on the DNS protocol. Different types of network attacks on the DNS protocol were analyzed, and the main methods of detecting malicious requests in a computer network were investigated, and tools for detecting malicious requests were considered. Technology has been deployed to detect malicious requests and conduct research on computer networks.

The purpose of this publication is to research the technologies for detecting malicious requests in computer networks and to deploy malicious request detection systems based on them, which will help to find malicious requests more effectively.

The practical significance of the obtained results lies in the implemented technologies for detecting malicious requests

Keywords ¹

Information security, IDS, cyber security, detection of malicious requests, elastic, computer networks, DNS protocol, network attacks.

1. Introduction

Analyzing the data presented in the company's annual report DNS Filter during 2021 and 2022, DNS plays a role in approximately 80% of attacks. With DNS-based cyberattack protection, organizations can immediately block up to 33% of DNS-based threats and increase the visibility of additional threats by 47%, protecting against zero-day attacks as well as other known threats [1]. Countering these attacks requires the use of modern cyber security techniques to protect systems, networks and applications from persistent cyber threats. Therefore, to detect malicious network attacks, you need to use tools that analyze DNS queries and block access to malicious, suspicious or other selective domain types. Which can potentially be harmful. DNS is one of the foundations of the Internet, a highly complex and decentralized system that converts human-readable domain names (such as havrs.com) into numeric IP addresses.

DNS has a hierarchical tree structure called the DNS namespace. Each dot in the domain name indicates a division between levels in the tree structure. The top layer of the DNS tree structure represents the root level, which begins with a dot. Below the root level are top-level domains (TLDs), which correspond to child root domains such as .com, .org, .edu.ua and others. Further, a TLD also has child domains that link to second level domains or authoritative domain name servers [2]. Finally, the fully qualified domain name (FQDN) places hostnames or subdomains in the DNS hierarchy. For example, the search process for the next domain name dn.dut.edu.ua always starts with a dot, which is the root server in the DNS tree structure. The root server then forwards the request to the appropriate TLD, which in this case is .edu.ua. An intrusion detection system (IDS) is a system that automates the intrusion detection process. IDS tries to detect incidents in the system before certain policies are violated. There are many causes of system incidents, such as malware infection, an attacker gaining access to the system or probing the system for vulnerabilities, or abuse of the system by authorized users resulting in a security policy violation. The purpose of IDS is to alert system administrators to such

Information Technology and Implementation (IT&I-2023), November 20-21, 2023, Kyiv, Ukraine

EMAIL: olga.leshchenko@knu.ua (O. Leshchenko); oleksandr.trush@knu.ua (O.Trush); mariya-trush@ukr.net (M. Trush); a.gorachuk@gmail.com (A. Horachuk); oleksandr.makhovych@knu.ua (O. Makhovych)

ORCID: 0000-0002-3997-2785 (O. Leshchenko); 0000-0002-4188-2850 (O.Trush); 0000-0002-1673-3351 (M.Trush); 0009-0000-6067-9025 (A. Horachuk); 0000-0002-4684-9881 (O. Makhovych)



© 2023 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

incidents at their earliest stage, before they cause any damage, and to support system administrators in their response to malicious incidents. While many incidents are malicious, others are not, for example, a user may enter an address by mistake, leading to an attempt to gain access to a critical system to which the individual is not authorized. Therefore, an IDS must be able to classify potentially malicious incidents with sufficient accuracy, that is, with a low rate of false negatives and false positives.

2. Types of network attacks on the DNS protocol

DNS Spoofing/Cache Poisoning: a type of attack in which fake DNS data is injected into the cache of a DNS resolver, causing the resolver to return an incorrect IP address for a domain [10]. Instead of going to the correct website, traffic can be redirected to a malicious resource or anywhere else the attacker wants, very often users are redirected to a copy of a known organization's site that is used for malicious purposes, such as spreading malware or collecting sensitive information to log into personal accounts and impersonate another person.

DNS tunneling: this type of attack uses other protocols to tunnel through DNS queries and responses. Attackers can use SSH, TCP, or HTTP to inject malware or stolen information into DNS queries that most firewalls can't always detect [4,5].

Attackers use the DNS resolver to route requests to the attacker's C2 server, where the tunneling program is installed. Once a connection is established between the victim and the attacker via the DNS resolver, the tunnel can be used to exfiltrate data or perform other malicious purposes.

NXDOMAIN attack: is a type of DNS flood attack where an attacker sends a large number of queries to a DNS server requesting records for domain names that do not exist in an attempt to cause a denial of service for legitimate traffic [6]. This can be achieved with sophisticated attack tools that can automatically generate unique subdomains for each request. NXDOMAIN attacks can also target a recursive resolver to flood the resolver's cache with unwanted requests.

Phantom domain attack. It has similar results to the NXDOMAIN attack on the DNS resolver. The attacker sets up a bunch of "phantom" domain servers that either respond very slowly to requests or don't respond at all [10]. The resolver then receives a flood of requests to these domains, causing it to wait for responses, resulting in slow performance and denial of service.

Random subdomain attack: in this case, the attacker sends DNS queries for multiple random, nonexistent subdomains of the same legitimate site. The goal is to create a denial of service for the authoritative name server of the domain, making it impossible to find the website from the name server. As a side effect, authoritative domain name server the attacker could also suffer because their recursive resolver cache would be loaded with bad requests.

Domain blocking attack: attackers orchestrate this form of attack by setting up special domains and resolvers to establish TCP connections to other legitimate resolvers. When target resolvers send requests, these domains send back slow streams of random packets, tying up resolver resources.

Attack on DNS amplification and DoS, DDoS: a DNS amplification attack is a type of DDoS attack in which attackers use publicly available open DNS servers to flood a target with DNS response traffic. An attacker sends a DNS lookup request to an open DNS server with the source address spoofed as the target address. When a DNS server sends a response to a DNS record, it is sent to the target group instead.

DNS hijacking. There are three types of DNS hijacking:

- Attackers can hack a domain registrar account and change your DNS nameserver to one they control.
- Bad subjects can change the A record for your domain's IP address to point to their address instead.
- Attackers can compromise an organization's router and change the DNS server that is automatically forwarded to each device when users log on to your network.

Over the years, attackers have successfully deployed various DNS-based attacks against company networks and their users. Attackers often use DNS to establish command and control (C2) of a botnet network. Doing so may result in unauthorized network access, lateral movement, or data exfiltration.

3. Deployment of malicious request detection technologies

To detect malicious traffic in computer networks, most specialists use tools to automate the

collection and subsequent analysis of traffic. There are two methods of collecting DNS data for analysis. The discovery system transmits DNS data modules either by passively captured DNS traffic of the network interface, or by reading existing PCAP files through a parser.

Using the Zeek network monitoring tool to collect information about packets from network interfaces in a log file format, and specifying configuration parameters to process packet data, all data will be stored in JSON format. This will make it possible to transfer the data received from network interfaces in a convenient form, reduce the time for their processing and convenient search for the necessary information. Processing a large amount of data in JSON format allows a software component called logstash, which has the ability to collect, process and send processed information to agents for synchronization with the Logz.io platform using Filebeat (Beats).

The Logz.io platform is used to manage logs, monitor infrastructure and interact with Cloud SIEM to unify monitoring, troubleshooting and security tasks. The next stage in the deployment of technologies for detecting malicious requests in information systems is the Elastic platform, which receives filtered data from Logz.io that is used for further analysis. As shown in (Fig. 1), the considered technologies form a complex of technical means for collecting, analyzing and detecting malicious requests in computer networks.

For researching the received data, Elastic offers machine learning capabilities that allow you to create patterns that will help choose the best classifier for quickly detecting anomalies and malicious attacks.

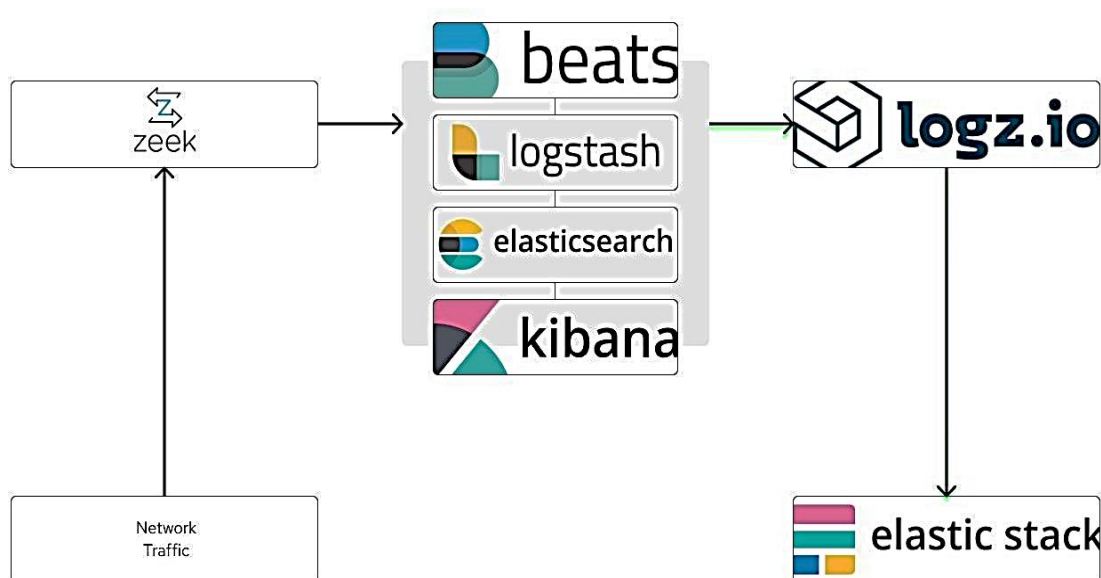


Figure 1. Network data analysis process

The task of machine learning is to detect a rare and unusual DNS query that indicates network activity with unusual DNS domains. This may be related to initial access, retention, command-and-control activity, or exfiltration (Figure 2). For example, when a user clicks on a link in a phishing email or opens a malicious document, a request may be sent to download and run a payload from an unusual domain. Once the malware is running, it can query the malicious DNS domain that the malware uses to communicate with the command and control server.

The task of machine learning is to detect an unusually high number of DNS queries when using DNS tunneling, this can be used for command and control activities, saving or extracting data. For example, dnscat2 tends to generate a lot of DNS queries for the top-level domain because it uses the DNS protocol to tunnel data. It should be noted that often when DGA malware is actively trying to contact a C&C server, it tends to generate a flurry of DNS requests at the same time (Figure 3).

Time series analysis of putative malicious domains shows activity peaks over time and little noise between the peaks. The peaks indicate that the model classified many domains as harmful in a short period of time, and thus obtained data of true DGA activity, which is demonstrated in (Fig. 4).

At this point, an analysis will be performed to see if Elasticsearch can detect DNS tunneling or not. The applied method of traffic analysis. Each request in the DNS tunnel will create a new hostname, the normal average number of unique hostname is below 250, so a more unique hostname indicates DNS tunneling. All logs are recorded and will be processed by Watcher using a custom script. The Watcher

then counts the number of unique names based on the strength of the domain. The number of unique host names in the domain is visualized in Kibana in the form of a graphic panel. If the number of unique hostnames is more than 250 and the domain does not exist in the list of trusted domains, an email will be sent with information about the detected anomalies.

DNS Tunneling

About

A machine learning job detected unusually large numbers of DNS queries for a single top-level DNS domain, which is often used for DNS tunneling. DNS tunneling can be used for command-and-control, persistence, or data exfiltration activity. For example, dnscat tends to generate many DNS questions for a top-level domain as it uses the DNS protocol to tunnel data.

Author Elastic

Severity Low

Risk score 21

Reference URLs

- <https://www.elastic.co/guide/en/security/current/prebuilt-ml-jobs.html>

False positive examples

- DNS domains that use large numbers of child domains, such as software or content distribution networks, can trigger this alert and such parent domains can be excluded.

Definition

Rule type	Machine Learning
Anomaly score threshold	50
Machine Learning job	packetbeat_dns_tunneling
Timeline template	None

Schedule

Figure 2. Creating an ML problem

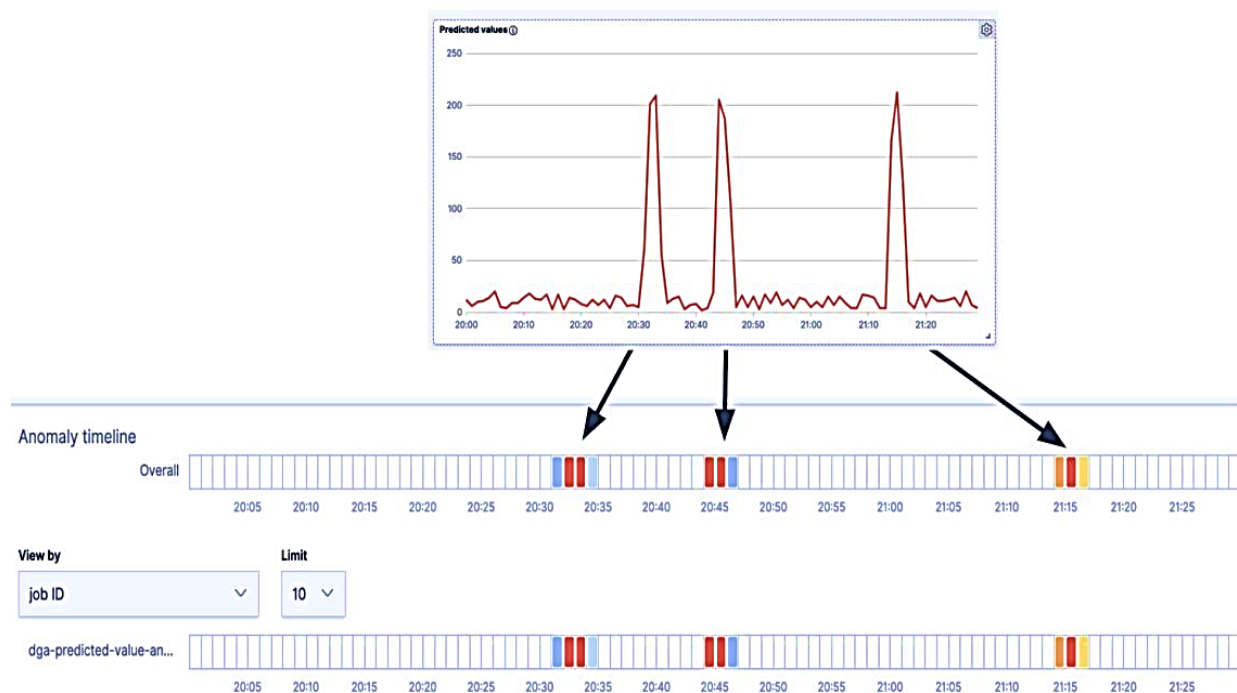


Figure 3. Report of anomalies when working with dnscat2

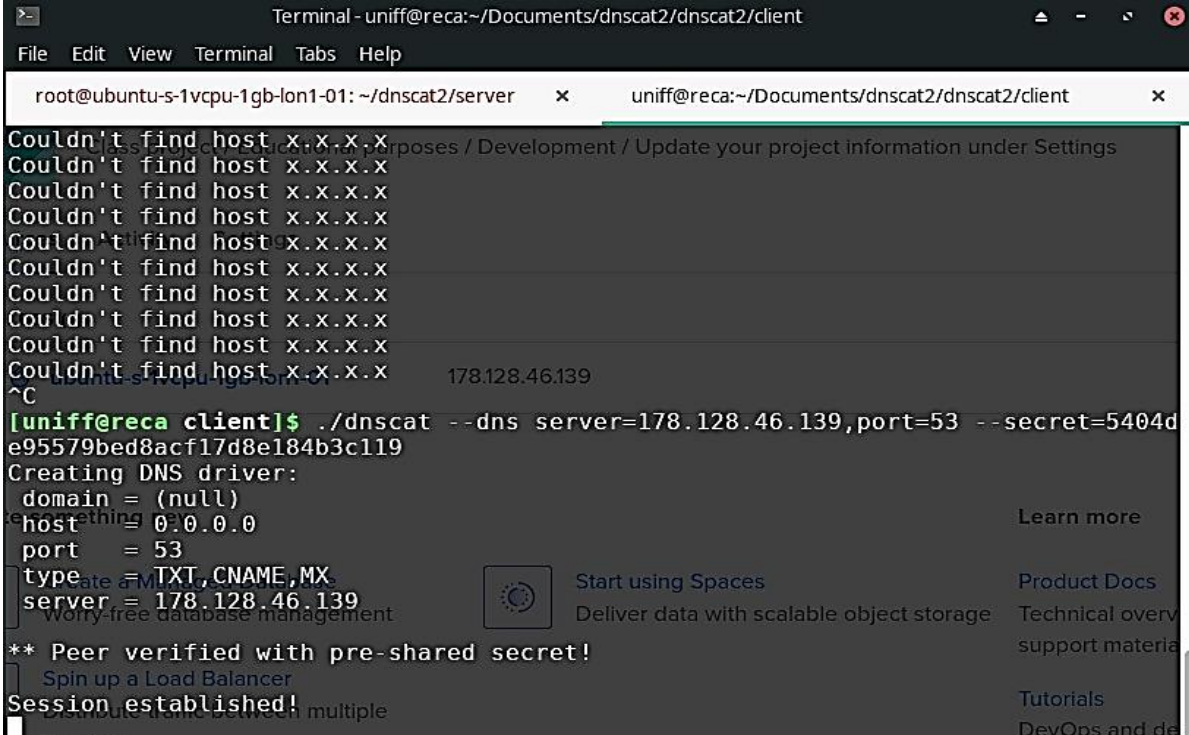


Figure 4. Activity graph of malicious domains

3.1. Analysis of malicious requests in computer networks

By simulating the scenario and the main target of the attack that was directed against the malicious query detection system during the testing and evaluation phase. It should be remembered that in the TCP scenario, a significant number of DNS queries are sent through DNS, which makes it visible to vulnerability detection systems. In custom DNS exfiltration scenarios and command and control scenarios, the attacker sends a limited number of DNS queries, making it less visible and difficult for the system to detect suspicious activities that occur on the computer network.

Using the dnscat2 tool, which is shown in (Fig. 5.), and is chosen as a modern service for creating an attack based on DNS. To generate artificial data to investigate the attacks, a virtual server running Ubuntu was created and the dnscat2 software was installed to interact with the host machine running on Arch Linux along with Wireshark.



```
Terminal - uniff@reca:~/Documents/dnscat2/dnscat2/client
File Edit View Terminal Tabs Help
root@ubuntu-s-1vcpu-1gb-lon1-01: ~/dnscat2/server x uniff@reca:~/Documents/dnscat2/dnscat2/client x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
Couldn't find host x.x.x.x
178.128.46.139
^C
[uniff@reca client]$ ./dnscat --dns server=178.128.46.139,port=53 --secret=5404de95579bed8acf17d8e184b3c119
Creating DNS driver:
domain = (null)
host = 0.0.0.0
port = 53
type = TXT,CNAME,MX
server = 178.128.46.139
** Peer verified with pre-shared secret!
Session established!
```

Figure 5. Generating dnscat2 queries

To evaluate the effectiveness of methods for detecting vulnerable queries, let's create a regular DNS data set. The network environment for the deployment of malicious request detection technologies has been established. Having collected DNS traffic for one week, it was prepared for analysis (Fig. 6). There are two important reasons for capturing plain DNS traffic in the experiment. The first reason is to compare and study the normal behavior of DNS traffic with DNS traffic that has a DNS tunneling attack, while the second reason is to identify and set a threshold value based on the normal level of DNS traffic. Due to the importance of the DNS protocol, DNS tunneling tools have been used to simulate tunneling attacks using various methods.

For example, the DNS tunneling tool (dns2tcp) uses TXT record types to perform tunneling, while the Iodine DNS tunneling tool uses NULL records. By collecting its own dataset, it started implementing DNS datasets that contain malicious DNS traffic. The DNS tunneling dataset includes a variety of DNS traffic generated by DNS tunneling tools, including Iodine, DNScat2, dns2tcp, and fraud-bridge, which are described in Table 1. Finally, the normal and malicious DNS datasets are combined and then imported into a detection system and are evaluated.

The detection system consists of two detection modules: payload and traffic analysis modules. Each module uses a base value to detect abnormal activity. For example, the payload analysis module sets the base value of the fully qualified domain name to 30 characters. This value was chosen based on recommendations from cyber security experts at DG Security. However, it is now necessary to choose which threshold value of the number of DNS requests to use during a certain period of time to identify malicious domain names for the traffic analysis module.

The threshold value is used in the traffic analysis module for machine learning training purposes. After capturing the DNS data, we generate text representation signatures from the rendering. If it's a malicious signature for a domain, we compare the number of DNS requests for the domain name to the baseline value and then label the training dataset as malicious or benign. Thus, there is a need to conduct practical experiments to select the most beneficial values for the number of DNS requests during a set period. The traffic analysis module is specifically aimed at detecting high-throughput DNS tunneling attacks using imaging signatures and a machine learning classifier.

No.	Time	Source	Destination	Protocol	Length	Info
479	69.247...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x08b3 CNAME dnscat.5235013f0b530bf5af6ca5
481	69.250...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x042b CNAME dnscat.4bf9013f0b331dd68f24dd
483	69.252...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x3ba1 TXT dnscat.3077013f0b26c6d47779a900
485	69.254...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x774d MX dnscat.7ba4013f0b997fb4ace30c003
487	69.256...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x6bae MX dnscat.3826013f0b7e68bf2df2ec003
489	69.258...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x457d TXT dnscat.316e013f0b93b739b89b4100
491	69.260...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x6fc2 MX dnscat.37fa013f0b8b6fd01e367c003
493	69.262...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x1a3c CNAME dnscat.1b12013f0b52040591e0f4
495	69.263...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x5619 TXT dnscat.73c4013f0b65905fb8260000
497	69.264...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x27c0 TXT dnscat.2e52013f0b014e4bc59c5b00
500	69.266...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x782c CNAME dnscat.5641013f0ba005da3e32af
502	69.267...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x6bac TXT dnscat.473c013f0b80e94be97d4a00
504	69.268...	192.168.1.83	192.168.1.80	DNS	300	Standard query 0x4389 TXT dnscat.12ae013f0b6d79bb43b3e600

Figure 6. Results of tunneling data collection

Table 1

Attack simulation tools

Tools	Scenario simulation	Aim
iodine	TCP over DNS	Establishing a secure VPN connection
dns2tcp	TCP over DNS	Establishing a secure VPN connection
DNSExfiltrator	DNS exfiltration technique	Withdrawal of confidential data
Cobalt Strike	Command and control	Execute commands and download files
dnscat2	Command and control	Execute commands and download files
PacketWhisper	Command and control	Withdrawal of confidential data

A DNS tunneling attack, such as TCP over DNS, typically sends a large number of DNS queries to establish communication links. Therefore, for these types of attacks, a threshold can be set where the volume of DNS traffic is one or two standard deviations above the average daily traffic. Instead, we focused on the more difficult problem of determining a threshold value for DNS exfiltration techniques scenarios. The main reason for this type is to send a certain number of requests depending on the file size. The DNSExfiltrator tool is used to simulate an exfiltration attack that transfers files outside of a compromised network to empirically determine the threshold. Several files between 1 KB and 1 MB were transferred over the network using the DNSExfiltrator tool and the number of fragments representing DNS queries was measured. This experiment used a maximum full domain of 255 bytes and a maximum label length of 63 bytes. Reducing the query sizes and DNS labels to lower values will result in even more DNS queries being sent for a given file size. Threshold value is set based on file size and number of DNS blocks. We will consider time and size as important factors in this evaluation, since the training phase of the traffic analysis module relies on capturing DNS data every n minutes, where the settings are based on daily network traffic.

For example, suppose an attacker wants to export a 100 KB text file containing information about a stolen credit card outside of an organization's network using a DNS exfiltration technique. A 100 KB file requires 600 DNS queries within 25 seconds to successfully remove it from the organization's network. In addition, the detection system needs to capture DNS data every minute, and to detect an attack, the threshold value must be less than 600. In the experiment with the detection system, the traffic analysis system was chosen to capture DNS packets every 30 seconds and 100 was set as the threshold value, assuming that the attacker needs to extract a file of at least 50 KB, which would result in about 300 DNS queries in 20 seconds. It should be noted that sometimes an attacker can use a delay technique between each DNS request to reduce traffic noise, but the payload analysis module in this work is configured to track these situations (Figure 7.).

So, after conducting an analysis of malicious DNS traffic requests, we received a graph of detected

anomalies, among which we chose domain names with the highest index, which we will add to the blacklist. And also configured the automatic ability to read DNS traffic from existing PCAP files to identify and detect the attack offline from previously captured network traffic.

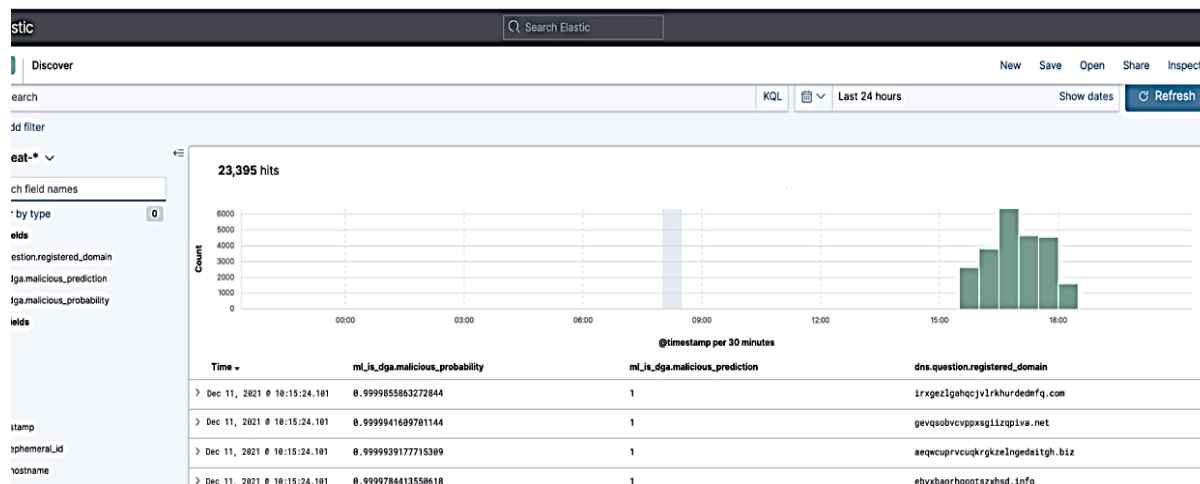


Figure 7. Report on received anomalies

4. The results of the implementation of malicious request detection technologies

After investigating the tunneling and exfiltration attacks, the resulting data was collected and sent using Wireshark and Elastic, which were used to collect and detect abnormal traffic indicators. Using Wireshark, data from the network interface was additionally obtained for more accurate use of them during analysis (Fig. 8.).

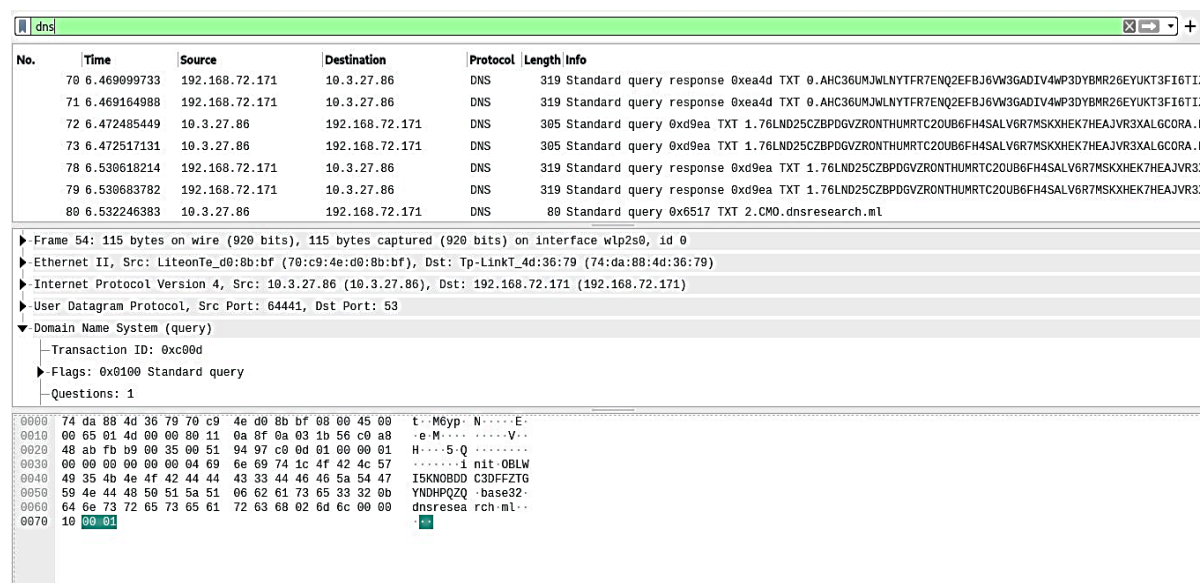


Figure 8. DNS tunneling network traffic result

After analyzing the data from the received reports, domain addresses were added to the black list and new search criteria for malicious domain names were formed. The result of the performed work was the deployment of network traffic monitoring tools that were obtained using attack creation tools.

Having received data from network interfaces, which were later sent to Elastic for processing and visualization, the results of network activity during research (Fig. 9.). The study was conducted in several stages to reject data that did not carry research value. Implementing DNS traffic monitoring technologies can generate many anomaly reports for the Security Center (SOC) team as they monitor and analyze the state of security within companies. In addition to monitoring firewalls and IPS systems to detect indicators of DNS compromise, infected hosts, or DNS tunneling attempts, SOC teams can develop

scripts to counter and detect such domains by using DNS filtering systems to create blacklists. That will help protect against attacks in the future. (Fig. 10).

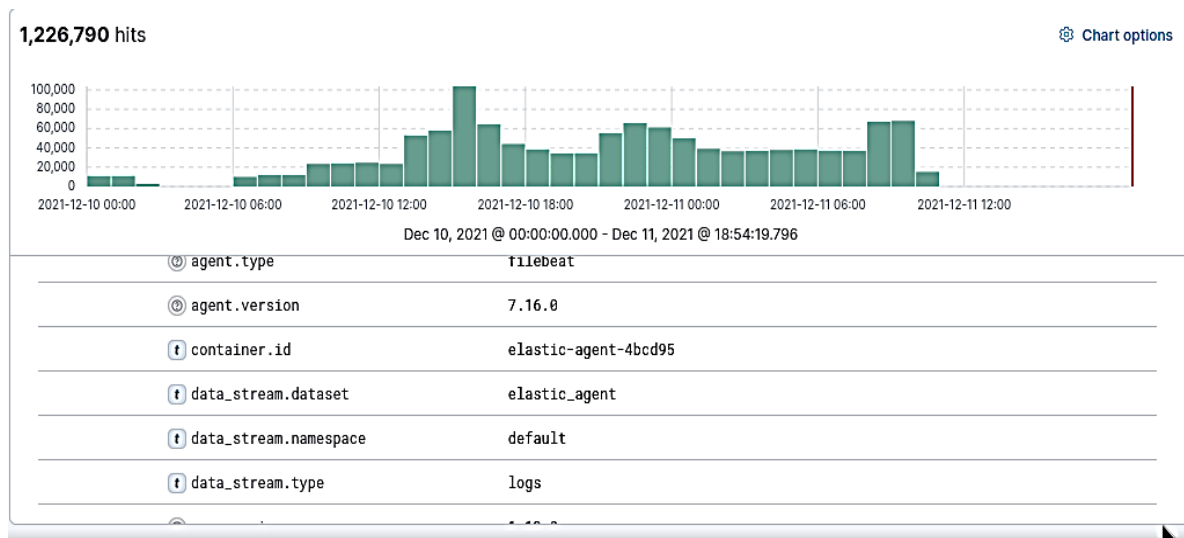


Figure 9. Graph of network traffic during testing

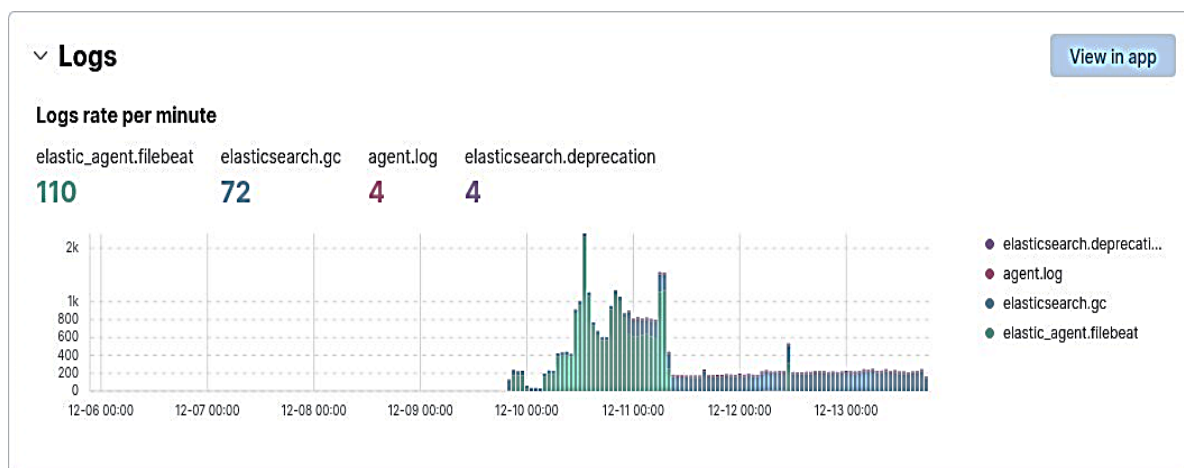


Figure 10. Graph of traffic anomalies

5. Conclusions

Vulnerability detection technologies and the results of network traffic research were studied. Implemented anomaly detection system aimed at analyzing DNS-based network attacks. Two detection modes are analyzed, which target DNS tunneling attacks and DNS exfiltration.

The proposed detection system is implemented using Elastic visualization methods and machine learning classifiers to distinguish DNS-based attacks. The development of a detection system using visualization and machine learning is singled out. It is noted that there are several alternative solutions for network traffic visualization of rare and unusual DNS queries that indicate network activity with unusual domains.

The results of the implementation of the system for detecting malicious requests of computer networks have been obtained.

The detection system has some limitations and shortcomings that need to be eliminated in future works. DNS proxy vulnerabilities to various DNS protocols and server attacks, including DoS and DDoS, server hijacking, DNS spoofing, and cache poisoning attacks. Addressing these issues is planned to be part of future work to create a comprehensive malicious query detection system using proprietary machine learning algorithms, such as source port and transaction randomization, to prevent DNS cache poisoning attacks.

In addition, DNSSEC, an additional layer of security to the existing DNS protocol, should be considered and implemented in the future to validate the chain of trust using public key cryptography

for authentication and data integrity, and to improve the overall system level of malicious query detection.

6. References

- [1] Dudnik, Y. Kravchenko, O. Trush, O. Leshchenko, N. Dakhno and V. Rakytskyi, "Study of the Features of Ensuring Quality Indicators in Multiservice Networks of the Wi-Fi Standard," 2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT), 2021, pp. 93-98, doi: 10.1109/ATIT54053.2021.9678691.
- [2] Trush, O., Dudnik, A., Trush, M., Leshchenko, O., Shmat, K., & Mykolaichuk, R. (2022, December). Mask Mode Monitoring Systems Using IT Technologies. In 2022 IEEE 4th International Conference on Advanced Trends in Information Theory (ATIT) (pp. 219-224). IEEE. DOI: 10.1109/ATIT58178.2022.10024216
- [3] Butkiewicz, M., Madhyastha, H. V., & Sekar, V. (2013). Characterizing web page complexity and its impact. *IEEE/Acm Transactions On Networking*, 22(3), 943-956.
- [4] DAMBALLA. Botnet Communication Topologies. Understanding the intricacies of botnet command-and-control. Retrieved from https://www.damballa.com/downloads/r_pubs/WP_Botnet_Communications_Primer.pdf.
- [5] The Federal Bureau of Investigation. Demarest, J. (2017). Statement Before the Senate Judiciary Committee, Subcommittee on Crime and Terrorism, Washington, D.C. Retrieved from <http://www.fbi.gov/news/testimony/taking-down-botnets>.
- [6] Detecting DNS Tunneling. Farnham, G., Atlasis, A.: SANS Institute InfoSec Reading Room, 2013. pp. 1–32.
- [7] A comparison of distance-based semi-supervised fuzzy c-means clustering algorithms. Lai, D.T.C., Garibaldi, J.M.: *Fuzzy Systems (FUZZ)*, In IEEE International Conference, 2015. P. 1580–1586.
- [8] As the Net Churns: Fast-Flux Botnet Observations. Nazario, J., Holz, T.: In: Conference on Malicious and Unwanted Software (Malware'08), 2008. P. 24–31. Kaur, S., Kaur, K., & Kaur, P. (2016). An empirical performance evaluation of universities website. *International Journal of Computer Applications*, 146(15), 10-16.
- [9] Matam, S., & Jain, J. (2017). *Pro Apache JMeter: web application performance testing*. Apress.
- [10] Heričko, T., Šumak, B., & Brdnik, S. (2021, September). Towards Representative Web Performance Measurements with Google Lighthouse. In *Proceedings of the 2021 7th Student Computer Science Research Conference* (p. 39)
- [11] James, I. (2019). *Webwaves: Web page auditing using Lighthouse*. Preview, 2019(203), 50-51.
- [12] Patil, S. A. (2020). Comparative SEO Techniques Analysis on core WebPages and its Effectiveness in Context of Google Search Engine, *International Journal of Scientific Development and Research*, Vol 5, Is.3 (pp. 420-428)
- [13] Rosenfeld, L. & Morville, P. & Arango, J. (2015). *Information Architecture: For the Web and Beyond*, O'Reilly Media
- [14] Z. Qian, H. Miao and H. Zeng, "A Practical Web Testing Model for Web Application Testing," 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System, Shanghai, China, 2007, pp. 434-441, doi: 10.1109/SITIS.2007.16.