Analyzing Network Traffic Data Using Hive Queries

Dharaben Patel, Xiaohong Yuan, Kaushik Roy, Aakiel Abernathy
Department of Computer Science
NC A&T State University
Greensboro, NC, USA
dppatel@aggies.ncat.edu;xhyuan@ncat.edu; kroy@ncat.edu; aabernat@aggies.ncat.edu

***Abstract* – Billions of devices are connected together with internet to serve the communication. Network monitoring to detect various security threats has become crucial in any organization. In this paper, we analyze large amount of network traffic data using Hive database in Hadoop Distributed File System (HDFS) environment. Hive queries are developed to identify security threats. The results of queries are demonstrated and the Hive Client application is developed where all the queries can be integrated. An Apache Zeppelin Visualization Tool is also introduced which can provide more insights on the dataset.**

Keywords—Network data security analysis; Hadoop Distributed File System; Visualization Tool; Apache Zeppelin

I. INTRODUCTION

With large amount of network traffic data being captured constantly, big data technology is necessary for network traffic analysis. In our previous work [1], the network traffic data was loaded into Hive database in Hadoop Distributed File System (HDFS) environment. Hive queries were developed for detecting SYN Flood, SYN/FIN, Null and Denial of Service attacks. The dataset used was the network data captured at the 2014 national collegiate cyber defense competition (NCCDC) provided by PREDICT cyber security dataset provider [2]. Since using HDFS could reduce the time needed to identify security threats, it has potential to be used for real time threat detection.

This work extends the previous work in the following ways:

(1) New queries are constructed that detect SYN Scan Attack, Land Attack, Smurf Attack, Fraggle Attack and potential malicious HTTP packets with OPTIONS and User-Agent Method, Cross-Site Scripting Attack, Command Injection Attack and Path Traversal Attack.

(2) Hive Editor was used to create and execute the queries which is a basic editor for SQL command. The queries need to be executed one by one, and the result of those queries are shown in a tabular format. For better output and convenience, a GUI interface needed to be developed where all the queries can be saved and when they are executed, it produces the output in visual format. Therefore, a Hive Client program is developed using JDBC connectivity. The Hive client program will make connection to the Hive Database and fetch the query result from the Hive Database. The Hive Client program needs to be executed using SSH Terminal.

(3) The web-based tool, Apache Zeppelin, is utilized to visualize the network traffic data analysis results.

The rest of the paper is organized as follows. In section II, the comparison between using Wireshark and using Hadoop platform for network data analysis is discussed. Section III presents the new Hive Queries developed for network security threat detection. Section IV describes the client application created to integrate all the Hive Queries. Section V presents the Apache Zeppelin tool and Hive Interpreter to produce visualization for queries. Section VI shows the experiments and results and Section VII concludes the paper.

II. WIRESHARK VS HADOOP ENVIRONMENT

Wireshark is a network analysis tool that can capture packets in real time and display data in a format that can be analyzed by the user [3]. Wireshark includes features such as filters and color-coding that allow users to inspect individual packets [4]. Users can create customized display filters and coloring rules to highlight the packets when inspecting network traffic data [5]. There are two types of coloring rules in Wireshark: temporary rules that are only in effect until the program is terminated, and permanent rules that are saved in a preference file so that they are available the next time Wireshark is run. Wireshark can be used for trouble shooting network problems.

However, there are two issues with Wireshark: large capture files and packet drops while capturing. As we already have the captured dataset, packet drops while capturing is not an issue for us. The main issue in our case is large capture file. When a large capture file (e.g. > 100MB) is uploaded to Wireshark, Wireshark becomes extremely slow for loading, filtering and other actions. NCCDC dataset has 131 MB, therefore, we could not use Wireshark to analyze NCCDC dataset.

Other network forensic analysis tools [6] (NFATs) allow administrators to monitor the networks, gather information about anomalous traffic, assist in network crime investigation and help in generating a suitable incident response. There are many proprietary and open source tools like NetIntercept, NetIntercept, tcptrace, NetDetector, Snort, tcpdump etc. which are used for network forensic analysis. The limitation with all these tools is the same as Wireshark, that is, when the dataset is huge, they cannot function properly. The size of the log data grows to the size of TBs in the real environment, which requires a powerful computation environment to carry out network analysis.

978-1-5386-1539-3/17/$31.00 ©2017 IEEE

III. HIVE QUERIES DEVELOPED TO ANALYZE THE NETWORK ATTACKS

A. *Queries to select distinct protocol and flags*

We start by developing queries to group the data by protocol and flag in order to find more specific attacks corresponding to specific protocol or flag.

The Hive queries to select distinct protocols are shown below. Fig. 1 shows the classification of network packets among different protocols. Fig. 2 shows the classification of network packets among different flags.

Hive query to select Distinct Protocol

SELECT protocol, count(no) as noofpackets FROM log_data GROUP BY protocol ORDER BY protocol ASC

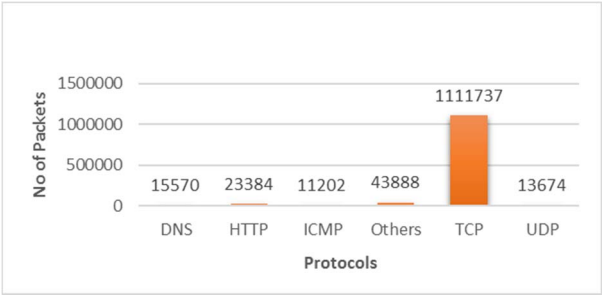

Fig.1. Classification of network packets among different protocols

Hive query to select Distinct Flag

SELECT flag, count(no) as noofpackets FROM log_data GROUP BY flag ORDER BY flag ASC

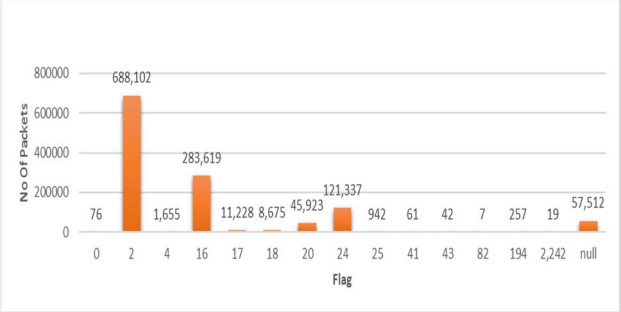

Fig. 2. Classification of network packets among different flags

Table I shows the TCP Flags and the Flag No. corresponding to each Flag [7]. If the URG bit is set, then the Urgent pointer field is significant. If the ACK bit is set, then the Acknowledgment field is significant. PSH indicates Push function, which asks to push the buffered data to the receiver. RST means that the connection is reset. SYN means the packet Synchronizes sequence numbers. FIN indicates it is the last packet from sender.

TABLE I. TCP FLAG AND THEIR EQUIVALENT FLAG NUMBER

Flag	Flag No.
URG	32
ACK	16
PSH	8
RST	4
SYN	2
FIN	1

B. *Queries to detect attacks*

This section describes the queries to detect the following attacks: SYN Scan Attack, Land Attack, Smurf Attack, Fraggle Attack, Identifying Malicious HTTP Attacks which covers packets which uses OPTIONS and User-Agent method, Command Injection Attack and Path Traversal Attack.

B.1 SYN Scan Attack

Port scanning is the process of probing for services provided by a system. SYN Scan is a type of port scan in which SYN packet is generated and sent without making a full TCP connection. The target port responds with a SYN-ACK if it is open or RST if it is closed [8]. SYN Scan attack can be identified by the following query:

Hive query to detect SYN Scan Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE flag = 2 AND source != destination GROUP BY source

B.2 Land Attack

In Land attack the victim is sent a SYN packet which has the same source and destination address. As a result, the victim replies to itself, which leads to an infinite loop and denial of service [8]. Land attack can be identified by the following query:

Hive query to detect Land Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE flag = 2 AND source == destination GROUP BY source

B.3 The Smurf Attack

The Smurf attack is a distributed denial-of-service attack in which a malicious Internet Control Message Protocol (ICMP) packet with a spoofed source IP is broadcasted to every IP in a network. The response from these IPs to the spoofed source IP (the victim computer) will cause denial of service [9]. IP address ending with 255 and 0 are used for broadcasting. So in the query, IP addresses which ends with 255 or 0 are searched for.

Hive query to detect Smurf Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'ICMP' AND destination like '%.255' AND info like '%Echo (ping) request%' GROUP BY source

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'ICMP' AND destination like '%.0' AND info like '%Echo (ping) request%' GROUP BY source

B.4 Fraggle Attack

Instead of using ICMP packets for denial of service attack, a Fraggle attack uses UDP packets [10]. In the following query, IP addresses ending with 255 and 0 are also searched for.

Hive query to detect Fraggle Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'UDP' AND destination like '%.255' GROUP BY source

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'UDP' AND destination like '%.0' GROUP BY source

B.5 Identifying dangerous use of HTTP Methods

The OPTIONS method can be used by the attacker to determine which methods are supported in the web server. These methods can then be used for attacks [11]. The following query identifies HTTP packets with OPTIONS methods.

Hive query to detect HTTP packets with OPTIONS method

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'HTTP' AND info LIKE '%OPTIONS%' GROUP BY source

The "User-Agent" header is used for sending information about the client to the server, such as the browser, host operating system and language. Attacker could spoof "User-Agent" header to retrieve web content designed for other browser types or devices. The "User-Agent" header could also be used to send information with malicious intention [11]. The following query identifies HTTP packets with "User-Agent" header.

Hive query to detect HTTP packets with "User-Agent" header

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'HTTP' AND info LIKE "%User-Agent%" GROUP BY source

B.6 Cross Site Scripting Attack

Cross Site Scripting (XSS) is a type of attack in which an attacker injects malicious scripts into a web application which are executed by a browser. XSS vulnerabilities can be exploited to read cookies, redirect a user to malicious site, or modify the content on a page [11]. XSS attack can be identified by searching for HTTP packets that include strings "<script>".

Hive query to detect Cross Site Scripting Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'HTTP' AND info LIKE "%<script>%" GROUP BY source

B.7 Command Injection Attack

In command injection attack, operating system commands are injected and executed. An example command that could be injected to the web application is "cat /etc/passwd", which intends to see the passwords of registered users [11]. Below is the query that searches for "etc/passwd" in the HTTP packets:

Hive query to detect Command Injection Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'HTTP' AND info LIKE "%etc/passwd%" GROUP BY source

B.8 Path Traversal Attack

The path traversal attack tries to access files and directories outside the current directory by using "(../)" sequences and its variations. Through path traversal attackers may gain unauthorized access to critical system files [11]. The following query finds malicious packets which indicate Path Traversal Attacks.

Hive query to detect Path Traversal Attack

SELECT source, count(no) AS noofpackets FROM log_data WHERE protocol = 'HTTP' AND info LIKE "%../../etc/passwd%" GROUP BY source

IV. Hive JDBC Client program

We built many Hive queries to find network attacks on the network traffic dataset which is uploaded on HDFS. It was inconvenient to execute each and every query on the dataset and check the results. Therefore, we created a client application from where we can execute all the queries and see the results at one place. As a first step, we created the Hive JDBC Client program which communicates to Hive with the JDBC Connectivity and fetch the results from the Hive Database. Fig. 3 shows the JDBC Client Program. In order to run the Client program, the Hadoop HDFS environment needs to be configured as shown in Fig. 4. Configuration file can be run by typing the ./filename and it will prompt the output in the terminal.

```
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.DriverManager;
public class HiveCreateDb {
   private static String driverName = "org.apache.hive.jdbc.HiveDriver";
   public static void main(String[] args) throws SQLException {
     try{
     Class.forName(driverName);
     } catch (ClassNotFoundException e) {
       e.printStackTrace();
       System.exit(1);
     }
    Connection con = DriverManager.getConnection("jdbc:hive2://localhost:10000/logdata", "admin", "***********");
    Statement stmt = con.createStatement();
    Resultset res = stmt.executeQuery("SELECT source, count(no) as Packets FROM log_data WHERE flag = 0 GROUP BY source");
    System.out.println("Result:");
    System.out.println(" Source \t Packets ");
    while (res.next()) {
        System.out.println(res.getInt(1) + " " + res.getString(2));
    }
    con.close();
  }
}
```

Fig.3. Sample hive client program

V. HIVE JDBC GUI – APACHE ZEPPELIN HIVE INTERPRETER

After the Hive JDBC Client Program was created, the next step was to visualize the query results. The Apache's Zeppelin [12] can be used for Data Analytics and Visualization. Apache Zeppelin is a web-based tool that can create data-driven and interactive documents with Hive.

A. *Apache Zeppelin Installation on Hadoop Environment*

As the Network traffic dataset is stored on the HDFS, the visualization tool also needs to be on the same environment. Apache Zeppelin works with the existing Hive Database so it is ideal for the visualization and analytics. Apache Zeppelin provides a platform to create both the front end which is the GUI and the back end. In our scenario, the back end connects to the Hive Database. Fig. 5 shows the user interface of Zeppelin installed on the HDFS environment.

```
dfs@HadoopCluster:/home/hdfs$ cat HiveCreateDb.sh
!/bin/bash
ADOOP_HOME=/opt/cloudera/parcels/CDH/lib/hadoop
IVE_HOME=/opt/cloudera/parcels/CDH/lib/hive

cho -e '1\x01foo' > /tmp/a.txt
cho -e '2\x01bar' >> /tmp/a.txt

ADOOP_CORE="/opt/cloudera/parcels/CDH/lib/hadoop/hadoop-common-2.6.0-cdh5.4.7.jar:/opt/cloudera/parcels/CDH/lib/hadoop/had
op-common-2.6.0-cdh5.4.7-tests.jar:/opt/cloudera/parcels/CDH/lib/hadoop/hadoop-common-tests.jar"
LASSPATH=.:$HADOOP_CORE:$HIVE_HOME/conf

or i in ${HIVE_HOME}/lib/*.jar ; do
    CLASSPATH=$CLASSPATH:$i
one
echo $CLASSPATH

ava -cp $CLASSPATH TestNetworkAttacks
dfs@HadoopCluster:/home/hdfs$
```

Fig.4. Configurations for hive client program

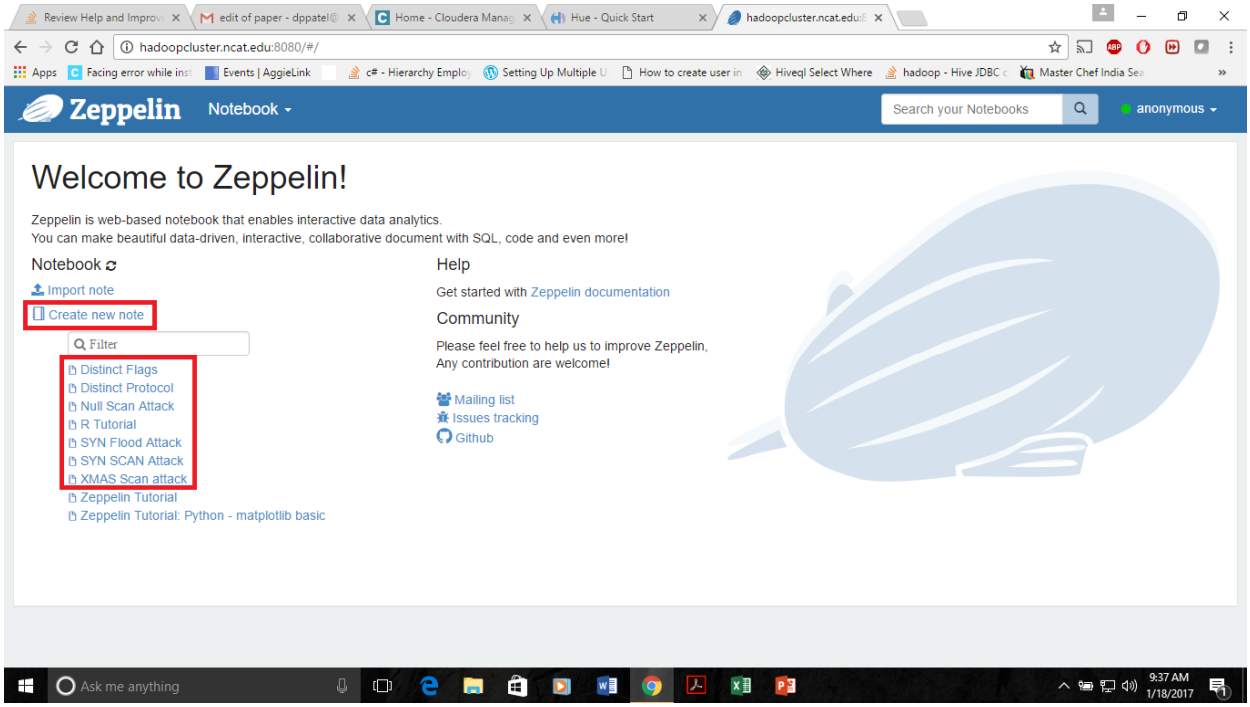

Fig.5. The user interface of Zeppelin on Hadoop environment

B. *Creating and Using an Interpreter*

Apache Zeppelin allows different interpreters such as Apache Spark, Python, JDBC, etc. to be plugged into Zeppelin. To use Hive code in Zeppelin, we used %jdbc interpreter. In the older version of Apache Zeppelin, there was separate Interpreter for Hive but in the newer version of Apache Zeppelin, Hive Interpreter is deprecated and merged into JDBC Interpreter. We can use Hive Interpreter by using JDBC Interpreter with the same functionality. Table II shows the setting of property values of the Hive Interpreter [13].

TABLE II. HIVE INTERPRETER CONFIGURATION PROPERTIES

Property	Value
hive.driver	org.apache.hive.jdbc.HiveDriver
hive.url	jdbc:hive2://localhost:10000
hive.user	hiveUser
hive.password	hivePassword

VI. EXPERIMENT & RESULTS

The queries developed above were executed on the NCCDC dataset with 1219454 packets. The queries found 7346 SYN Scan packets. 27 Smurf packets, 17 Fraggle packets, and 632 dangerous HTTP packets (Fig. 6). The visualization results generated in Zeppelin is shown in Fig. 7.

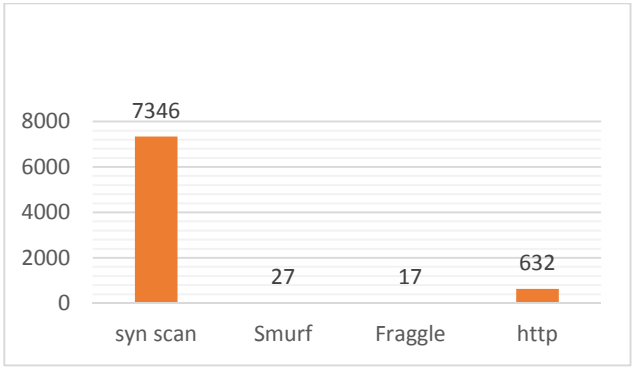

Fig. 6. Number of malicious packets detected in NCCDC sample dataset of 1219454 packets

VII. CONCLUSION

This paper describes the Hive queries we developed to identify network security threats in HDFS environment. Queries for detecting SYN Scan Attack, Land Attack, Smurf Attack, Fraggle Attack and potential malicious HTTP packets with OPTIONS and User-Agent Method, Cross-Site Scripting Attack, Command Injection Attack and Path Traversal Attack were developed and executed on NCCDC dataset. An application program we created which integrates all the queries developed was also described. Apache Zeppelin, the web-based notebook that enables interactive data analytics, was installed on the current HDFS environment to visualize the results. Future work includes developing more queries to identify network attacks, and using Apache Zeppelin to visualize the query results.

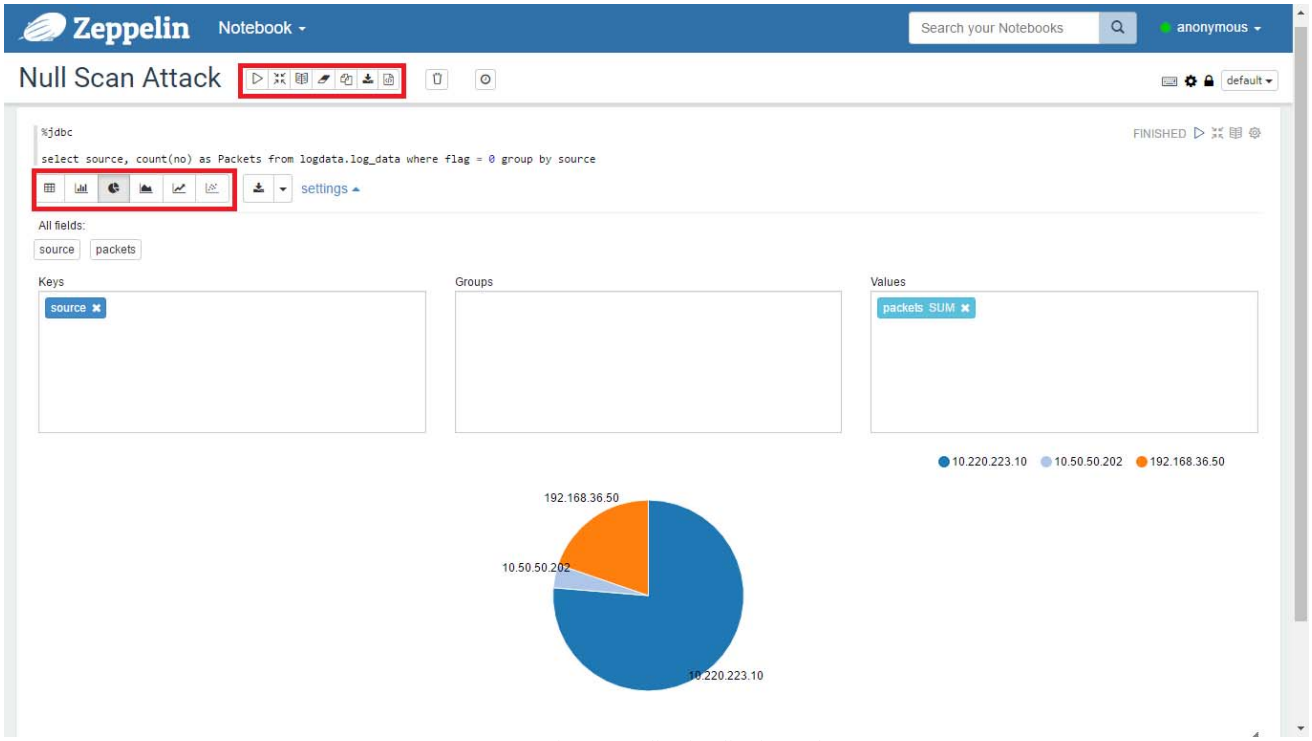

Fig. 7. Zepplin visualizatin results

ACKNOWLEDGEMENT

This work is partially supported by NSF under the grant CNS-1460864. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF.

REFERENCES

[1] Bachupally, Y. R., Yuan, X., & Roy, K. (2016, March). Network security analysis using Big Data technology. In SoutheastCon, 2016 (pp. 1-4). IEEE.

[2] PREDICT: Protected Repository for the Defense of Infrastructure against Cyber Threats.

[3] Wireshark. Available at: *https://www.wireshark.org/*.

[4] C. Sanders. *Practical packet analysis: Using wireshark to solve real-world network problems*. No Starch Press, 2011.

[5] SANS Institute InfoSec Reading Room. Wireshark: A guide to Color My Packets. Available at: https://www.sans.org/reading-room/whitepapers/detection/wireshark-guide-color-packets-35272

[6] Pilli, E. S., Joshi, R. C., & Niyogi, R. (2010). A generic framework for network forensics. International Journal of Computer Applications, 1(11).

[7] TCP(Transmission Control Protocol). Available at: http://www.linktionary.com/t/tcp.html

[8] Pilli, E. S., Joshi, R. C., & Niyogi, R. (2011, February). Data reduction by identification and correlation of TCP/IP attack attributes for network forensics. In Proceedings of the International Conference & Workshop on Emerging Trends in Technology (pp. 276-283). ACM.

[9] Kaspersky Lab. What is a Smurf Attack? – Definition. Available at: https://usa.kaspersky.com/internet-security-center/definitions/smurf-attack#.WMJZKU3dXIU

[10] Singh, A. Demystifying Denial-Of-Service attacks, part one. Available at: https://www.symantec.com/connect/articles/demystifying-denial-service-attacks-part-one e.

[11] Http malicious requests. Available at: https://www.sans.org/reading-room/whitepapers/detection/identify-malicious-http-requests-34067

[12] Apache Zeppelin Available at: https://zeppelin.apache.org/docs/0.5.6-incubating/interpreter/hive.html

[13] Apache Zeppelin – Hive Settings. Available at: https://zeppelin.apache.org/docs/0.6.2/interpreter/hive.html