B.TECH PROJECT REPORT

On

Network Security Systems Log Analysis and Visualization

By

Amit Kumar Meena
160001004

DISCIPLINE OF

COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

DECEMBER 2019

Network Security Systems Log Analysis and Visualization

A PROJECT REPORT

Submitted in partial fulfillment of the
requirements for the award of the degrees

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by :

Amit Kumar Meena

Guided by :

Dr. Neminath Hubballi

Associate Professor, Discipline of Computer Science and Engineering, IIT Indore

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE
DECEMBER 2019

Candidate's Declaration

I hereby declare that the project entitled **Network Security Systems Log Analysis and Visualization** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in **Discipline of Computer Science and Engineering** completed under the supervision of **Dr. Neminath Hubballi**, IIT Indore is an authentic work.
Further, I declare that I have not submitted this work for the award of any other degree elsewhere.

Amit Kumar Meena

Certification by BTP Guide

It is certified that the above statement made by the student is correct to the best of my knowledge .

Dr. Neminath Hubballi,
Associate Professor,
Discipline of Computer Science and Engineering,
IIT INDORE

Preface

This project report on **Network Security Systems Log Analysis and Visualization** is prepared under the guidance of **Dr. Neminath Hubballi**.

In this project, we have done log Analysis using custom python scripts on the logs generated by production level security appliances deployed in our university network. We also describe NViZ an interactive graphical visualization tool developed to visualize log data generated by network security devices and services like firewall, intrusion detection system and domain name system.

I have tried to the best of my abilities and knowledge to explain the observation and trends obtained from log analysis in a lucid manner. The graphical visualization tool NViZ can generate high-level visualization using logs from various network security systems has also been explained in this report. The source code of NViZ has been made public on Github repo.

Amit Kumar Meena,
160001004
B.Tech. IV Year
Discipline of Computer Science and Engineering
IIT INDORE

Acknowledgments

I would like to take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of my course of research.

I would like to express my sincere gratitude to my supervisor **Dr. Neminath Hubballi**, Associate professor, Discipline of Computer Science and Engineering, IIT Indore for his valuable suggestions and keen interest throughout the progress of my course of research work.

I would like to acknowledge IIT Indore for providing all necessary research infrastructures required for undertaking the project.

Amit Kumar Meena,
160001004
B.Tech. IV Year
Discipline of Computer Science and Engineering
IIT INDORE

Abstract

Network perimeter security appliances like firewalls, intrusion detection, and DNS resolver are deployed in a network to protect and serve the devices residing in it. Logs generated by these devices are used to identify security compromises, vulnerable systems, etc and serve as an invaluable asset for a network administrator. We have obtained various observations and trends using logs generated by production level security appliances deployed in our university network. In particular, we process the logs generated by firewall, intrusion detection/prevention system and domain name system to identify trends, generate graphs and gain insights. We process 71 million network connection records which include 95.7 thousand alerts generated by an open-source intrusion detection system collected for 31 days and derive statistics to understand end host-level behavioral trends. In our analysis, we compare hosts that are known to be infected with remaining using a set of relevant parameters and identify differentiated behavioral trends.

We also describe *NViZ* which is an interactive graphical visualization tool that is developed to visualize log data generated by network security devices and appliances like firewall, intrusion detection/prevention system and domain name system. It can generate a wide range of visualization graphs identifying popular websites visited, active users, DNS query patterns, IDS alert types, network connection patterns, user behavior analytic, peer network connections distribution of infected and non-infected machines, etc. *NViZ* can also keep track of the newly generated logs and generate various visualization like firewall actions, DNS traffic by location, IDS alert based on priority, etc in real-time.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computing infrastructure and communication networks are often targeted by cyberattacks. Recently, cyberattacks have become a serious national threat such as shut down industry control systems, and an act of war. Therefore, the issue is suggested about the necessity of security management that is for integrated management of the network system. To protect the systems from attacks, various defense mechanisms like firewalls, intrusion detection systems (IDS), proxy servers, etc are employed. Network perimeter security appliances like firewall, intrusion detection systems mediate communications between hosts in LAN and WAN. They generate a large volume of log information. Computing systems and networks are also regularly audited for finding potential weaknesses. As these systems evolve, their configurations change and auditing these dynamic systems is tedious. Logs are the first place to look for detecting suspicious events, find details of events and also may initiate a forensic audit. This is a challenging, error-prone and laborious exercise. Several works describe automating log analysis to identify real incidents. In general, log analysis is used in a variety of tasks such as identifying anomalies, network threats, SSH brute-force attacks, detecting credential-stealing attacks, detecting malicious sites, malware behavior detection, botnet detection, etc.

Often the alarms[1] generated by the security systems are raw and do not give details of the whole incident. To identify meaningful incidents that require system administrators' attention, enterprises use other log analysis tools known as Secu-

[1]We use the word alarms and alerts interchangeably in this thesis

rity Incident and Event Management (SIEM) systems. These tools are capable of collecting, normalizing, and analyzing security events from different systems and generate a coherent view of the incidents. Visualization tools have also been frequently used by system administrators. A well designed and interactive visualization tool will help the network administrator to a great extent. Several previous works have proposed to correlate alarms generated by information security systems like firewalls, IDS, proxy servers, netflow records and reconstruct attack scenarios and graphs.

We process logs generated by firewall, intrusion detection/prevention system for 30 days to identify trends and gain insights. We not only give statistics of various types of attacks detected and alarms generated by security systems but also report on the behavioral insights of end hosts. In specific, we do the following contributions in this thesis.

1. We divide or group the alarms generated from security monitoring systems using their type, severity, and country of origin.

2. We compare infected and non-infected end hosts/systems behavior with various relevant parameters.

3. Provide insights into the observed behavior of end hosts through analysis.

We describe an interactive visualization tool called $NViZ$. It's is designed and developed using MEAN Stack[2]. $NViZ$ also uses Python script to generate a network graph by spawning script as a child process. It is a menu-driven application with the following features.

1. It supports network security system appliances like firewall, intrusion detection system and DNS server.

2. The tool can show most frequently visited websites, systems that have made most web and DNS requests, systems against which IDS has generated alerts, different IDS alert types, end-host connections logged by the firewall, peer connections, etc.

[2]MEAN is a free and open-source JavaScript software stack for building dynamic web sites and web applications.

3. It has features to adjust duration within which such log data needs to be visualized and also comparison capabilities where two different graphs can be rendered side by side on to the screen.

4. *NViZ* also supports real-time visualization for the live network security systems.

1.1 Motivation

Cyber attacks are increasing at an alarming rate, thus it is necessary to have some reliable network perimeter security devices to protect network infrastructure. There is a wide range of network perimeter security devices available to use but most of them generate raw logs that do not give the entire picture of a cyber incident. These systems monitor network traffic and generate large volumes of log data. Often finding interesting stuff out of these logs is a challenging task. Network administrators often use log parsing, processing, and interactive visualization tools for gaining insights and locate issues.

An efficient log analysis tool can help the network administrator to derive statistics to identify differentiated end host-level behavioral trends using a set of relevant parameters and filters. Also, it can increase the readability of logs along with pinpointing various emerging trends. Visualization tools have been developed to provide an accessible way to see and understand trends, outliers, and patterns in data. A well designed and interactive visualization tool will help the network administrator to comprehend information quickly and identify relationships or patterns easily.

1.2 Organization of the report

The remaining portion of the report is organized as follows. A literature survey is described in chapter 2. Network architecture and the dataset is described in chapter 3. Chapter 4 covers the trends and observations from log analysis.*NViZ* has been explained in chapter 5. Finally, chapter 6 concludes this report.

Chapter 2

Literature Survey

2.1 Log Analysis

In general, log analysis is used in a variety of tasks such as identifying anomalies, network threats, SSH brute-force attacks, detecting credential-stealing attacks, detecting malicious sites, malware behavior detection, botnet detection, etc. Several previous works have proposed to correlate alarms generated by information security systems like firewalls, intrusion detection systems, proxy servers, netflow records and reconstruct attack scenarios and graphs.

Log analysis for identifying anomalies: Anomaly detection is a critical step towards building a secure and trustworthy system. An extensible system called *LogLens* that supports both stateless and stateful bulk log analysis to identify anomalies is presented in $LogLens : AReal - TimeLogAnalysisSystem$ [1]. *DeepLog*, a deep neural network model utilizing Long Short-Term Memory (LSTM), to model a system log as a natural language sequence which allows it to automatically learn log patterns from normal execution, and detect anomalies is proposed by $M.Du, F.Li, G.Zheng$ and $V.Srikumar$[2].

Log analysis for network threats: LogRhytm developed network traffic analysis (NTA) which provides a way to investigate network-based threats as well as neutralize attacks before significant damage is done[3]. A real-time IDS that constructs normal behavior models concerning device access patterns and control activities of individual accounts from their long-term historical alerts in real-time is developed by $J.Chu, Z.Ge, R.Huber, P.Ji, J.Yates$, and $Y.C.Yu$.[4].

Log analysis for detecting malicious sites: An automated URL classification, using statistical methods to discover the tell-tale lexical and host-based properties of malicious Web site URLs is proposed by $J.Ma, L.K.Saul, S.Savage$, and $G.M.Voelker$. These methods can learn highly predictive models by extracting and automatically analyzing tens of thousands of features potentially indicative of suspicious URLs[5].

2.2 Log Visualization

Visualization tools have been developed for detecting scanning activity, other events in the network and also in forensic analysis. A well designed and interactive, visualization tool may have the functionality to visualize a large network, action taken within the network, botnet/malware visualization, etc.

Visualization tools for network: Network scans visualization provides very effective means for detection large scale network scans. A visual interactive network scans detection system called $ScanViewer$ is designed to represent traffic activities that reside in network flows and their patterns by $Z.Jiawan, L.Liang$, $L.Liangfu$, and $Z.Ning$. The ScanViewer combines the characteristics of network scan with novel visual structures and utilizes a set of different visual concepts to map the collected datagram to the graphs that emphasize their patterns.[6].

Visualization tools for malware behaviour: A visualization tool based on treemap called $NetVis$ is introduced to bind the network security technique and general network management together in an integrated visualization by $Z.Kan$, $C.Hu, Z.Wang, G.Wang$, and $X.Huang$[7].

Visualization tools for forensic analysis: A methodology which combines automated analysis of data from security monitors and system logs with human expertise to extract and process relevant data in order to determine the progression of an attack, establish incident categories and characterize their severity, associated alerts with incidents, and identify incidents missed by the monitoring tools is proposed by $A.Sharma, Z.Kalbarczyk, J.Barlow$, and $R.Iyer$ [8].

Chapter 3

Network Architecture & Dataset

The dataset we used in our study consists of logs generated from pfSence[1] firewall/Unified Threat Management System and DNS resolver residing in demilitarized zone deployed in our university network. Our university campus network design is based on a widely used network model having outside (WAN), inside (LAN) and demilitarized zone (DMZ) as shown in Figure 3.1. DMZ contains various servers like webserver, authoritative name server, DNS resolvers, etc. The firewall screens both incoming and outgoing traffic and do filtering based on rules. Firewall pfSense also has an integrated intrusion detection and prevention system Suricata[2]. The Suricata engine operates in a daily update mode though which updates its ruleset daily using Snort rulesets and Emerging threat (ET) open rulesets. In order to log, analyze and filter all DNS requests originated by campus users we have DNS resolvers in the demilitarized zone that pass through the firewall. Thus firewall intercepts, screens and logs all DNS requests. All the events pertaining to screened traffic by the firewall, alarms generated by Suricata are logged in a common log file. Firewall logs indicate the action taken on a particular connection like "pass", "block", while Suricata alarms indicate the different types of attacks detected. For e.g IDS alarms include detected malware, torrent applications, scan activity, etc. There is also several internal servers running different applications which we denoted as an internal server farm in Figure 3.1.

[1]pfSense is an open-source firewall/router computer software distribution based on FreeBSD. It is installed on a physical computer or a virtual machine to make a dedicated firewall/router for a network[9].

[2]Suricata is an open source-based intrusion detection system and intrusion prevention system. It was developed by the Open Information Security Foundation[10].

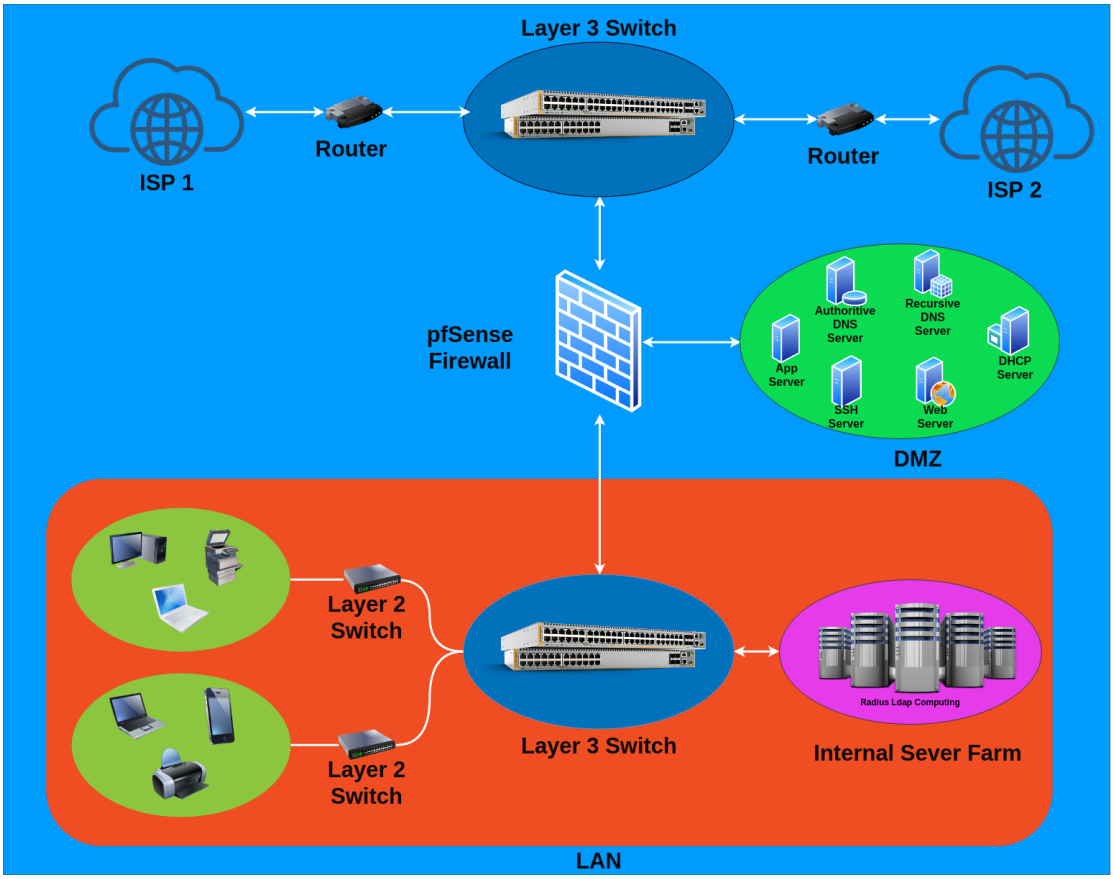

Figure 3.1: Representative Network Architecture

Our university network infrastructure currently serves approximately 1800 users. Going with a conservative estimate of each user has on an average 1.5 devices, it provides connectivity to 2700 devices. We collected logs generated from our university pfSense firewall/UTMS and DNS resolver for the period of one month starting from 11-05-2019 to 10-06-2019. During this period of 31 days, Suricata logged 732 unique types of alerts. In all, 71 million connection records were generated which includes more than 95.7K Suricata alerts. The total size of this data is 132.6 GB with an average of 4.3 GB alert data being generated every day. DNS resolver logged around 6.5 million DNS queries every day with 4000+ unique websites resolved.

00:00:00 10.100.100.251 filterlog:61, 16777216, ,1000000552, igb2, match, pass, out, 4, 0x0, , 64, 21241, 0, DF, 6, tcp, 60, 35.227.227.186, 10.100.57.220, 443, 45084, 0, SA, 2676392704, 347543621, 65535, ,mss;nop; wscale; sackOK; TS

Figure 3.2: Sample Firewall Log

An individual log entry generated by the firewall consists of important fields

like timestamp, hostname, a rule number, a subrule number, tracker, real interface, reason, action, direction, IP version, IPv4/IPv6 info, TCP/UDP info shown in Table 3.1. Figure 3.2 is a sample pfSense firewall alert and we can verify that it includes all the details mentioned in Table 3.1.

Table 3.1: Firewall Alert Fields

Field Name	**Description**
Timestamp	Time at which the action is logged
Hostname	Server running the pfsense server
Rule Number	Rule against which the action is logged
Sub rule number	Sub category of the rule
Tracker	Unique ID per rule, Tracker ID is stored with the rule
Real interface	Internal interfaces on the Routine Engine e.g. em0,em1
Reason	Reason for the log entry (e.g. match)
Action	Action taken that resulted in the log entry (e.g. block, pass)
Direction	Direction of the traffic (in/out)
IP version	IP version (4 for IPv4, 6 for IPv6)
IPv4/IPv6 Info	It include IP specific info like TOS, Protocol ID,Packet Length, Source IP, Destination IP, etc.
TCP/UDP Info	It include Protocol specific info like Data Length, Flags, Source Port, Destination Port, etc.

00:12:34 10.100.100.251 suricata[48238]: [1:2008581:3] ET P2P BitTorrent DHT ping request [Classification: Potential Corporate Privacy Violation] [Priority: 1] UDP 10.100.59.81:4041 -¿ 82.221.103.244:6881

Figure 3.3: Sample Suricata Alert Entry

Figure 3.3 is a sample alert generated from Suricata IDS. Along with common fields like IP addresses and port numbers, this has few additional fields like severity, class of alert, etc as shown in Table 3.2.

00:12:34 10.100.100.251 suricata[48238]: [1:2008581:3] ET P2P BitTorrent DHT ping request [Classification: Potential Corporate Privacy Violation] [Priority: 1] UDP 10.100.59.81:4041 -¿ 82.221.103.244:6881

Figure 3.4: Sample DNS resolver Entry

Figure 3.4 is a sample entry logged by the DNS resolver. Along with common

Table 3.2: IDS Alert Fields

Field Name	**Description**
Timestamp	Time at which the alert is logged
Signature Id	Unique id also given to every kind of alert
Alert Information	Brief about the type of alert
Classification	Classify the alert e.g. Network Scan, Trojan
Alert Priority	Indicate the severity of alert generated
Protocol Info	Connection Protocol (e.g. TCP/UDP)
Machine Info	Source IP and destination IP along with the port number involved

fields like IP addresses and timestamp, this has an additional field named resolved address as shown in Table 3.3.

Table 3.3: DNS Resolver Fields

Field Name	**Description**
Timestamp	Time at which the DNS query is resolved
Source IP	IP address of source machine
Resolved Address	Resolved address from the DNS query

Although DHCP is used in our network to dynamically assign IP addresses to end hosts, the lease time for an IP address is very large (8 days) and the DHCP server stores the IP-MAC address association in its internal database. Thus IP addresses assigned to our internal machines are mostly stable and are renewed. This ensures that the analysis is done like peer connections of infected machines (next section), etc are not influenced due to rotation of IP addresses.

Chapter 4

Log Analysis

We performed log analysis experiments on the dataset to understand the behavioral characteristics of end hosts. To parse the log files and derive statistics, we wrote custom scripts in Python. In the following 8 subsections, we describe these experiments and the results obtained.

4.1 Priority Based Alert Classification

An alert is detected and logged by the firewall integrated intrusion detection/prevention system Suricata. As mentioned previously, every Suricata alert has a priority number attached to it. The priority number denotes how harmful/severe the detected incident is. In Suricata, this priority number can range from 1 to 255. However, the numbers 1 to 4 are most often used. In the Suricata nomenclature, lower the priority number assigned to the alert more severe it is. For example, a ransomware attack is much more serious in comparison to a port scan attack, thus the ransomware attack will have a high priority (1) in comparison to the ssh/port scan incident (2/3).

An analysis has been done on the Suricata logs across all the days and found the distribution of alerts according to their priority. Figure 4.1 gives the fraction of total alerts contributed by each priority type. We can notice that priority 1 and priority 3 types of alerts contribute more to the total in comparison to the priority 2 category alerts. In our dataset, we found the maximum number of priority 1 alerts belong to BitTorrent connections. These are Peer-to-Peer connections established

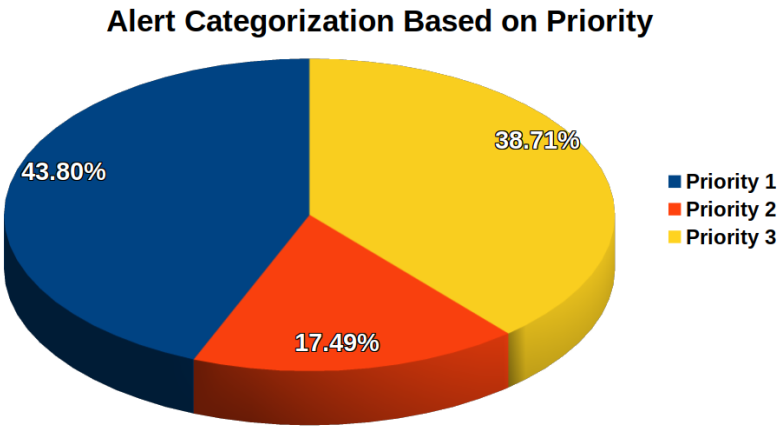

Figure 4.1: IDS Alert Priority Distribution

with peers in WAN without DNS name resolution. On the other hand priority, 3 alerts are generated by a large number of "unusual activity on port 445" detected by the IDS engine.

4.2 Distribution of IDS Alert Types

In the span of 31 days of the continuous monitoring period, Suricata generated various types of alerts. These alerts correspond to different types of infections. We used the signature id field to count the number of alerts generated and its type to group them. This experiment was motivated to understand the distribution of different types of alerts. In the entire dataset, we noticed 732 unique types of alerts generated by the IDS and out of which only a few were repeating in the majority. In particular, we noticed Torrent, Ransomeware, Scan, and Malware command and control (CnC) contribute 96.72% to the total alerts generated by Suricata IDS. Figure 4.2 shows the individual contribution of the top four alerts in the whole Suricata alert dataset and the contribution of the remaining 728 alert types in the alert pool (denoted as others).

We also measured the number of connections originating from the IP addresses of these infected machines. Figure 4.3 shows the total number of connections (with repetitions) originating from different types of infected machines. Machines which run BitTorrent generates the maximum number of connections (43.20%). We can notice that although ransomware alerts are very less in number (0.67% of total)

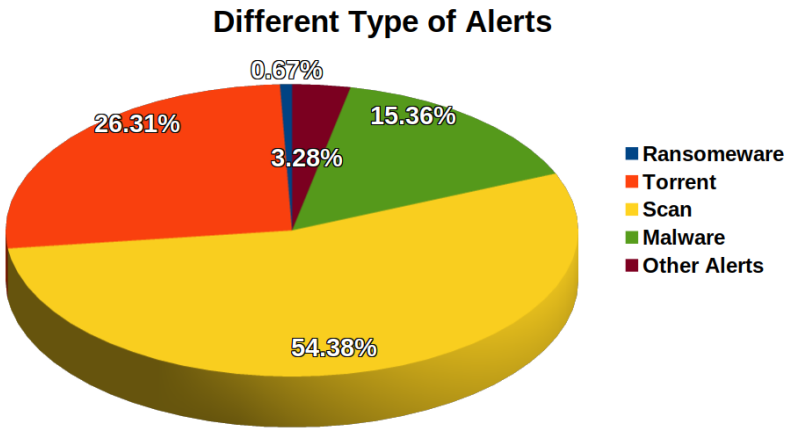

Figure 4.2: Distribution of Alert Types

but these infected machines generate a significant number of connections (22.22%).

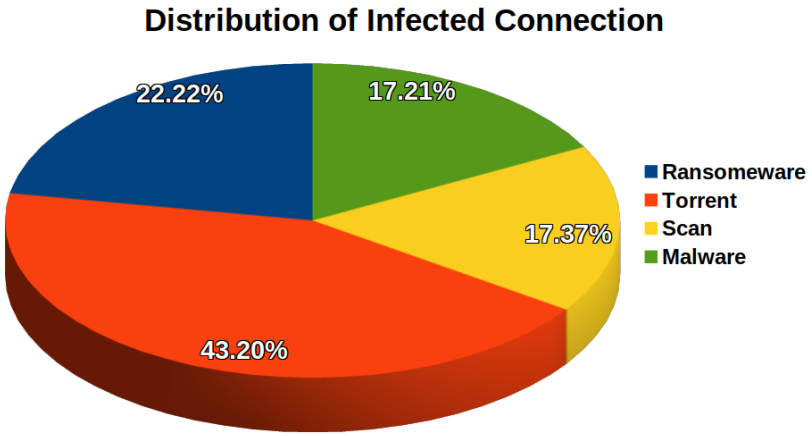

Figure 4.3: Distribution of Infected Connection

4.3 Geographical Distribution

During 31 days of monitoring, we noticed Suricata logged 71894 external IP addresses which were associated with some intrusion/alert. We used reverse IP address lookup to find the location and *cname* of the external IP addresses and observed most of the external IP for which alert is triggered are pure IP addresses against which no services are running. Overall these external machines were found to be located in 194 different countries. Table 4.1 shows distribution of the reported external IP addresses across different countries, as we can see that top 10 countries contribute 69.4% of the reported external IP addresses of various alerts.

Table 4.1: Country-wise Distribution of IDS Alert Sources

Country Name	**IP Count**	**Fraction**
United States	18779	0.275
Russia	6747	0.099
China	3419	0.050
France	3110	0.046
Argentina	3082	0.045
India	2987	0.044
Iran	2387	0.035
Japan	2381	0.035
United Kingdom	2312	0.034
Germany	2145	0.031
Other	20967	0.307

4.4 End Host Level Peer Connections

The firewall pfSense records every connection from an internal machine to an external machine(s) where an external machine can be a website, a server or a pure public IP address. In order to understand the connection patterns from both infected and non-infected machines, we analyzed these firewall connection logs. Let $X=\{x_1, x_2, \cdots, x_n\}$ be the set of internal machines which are identified as infected[1] and $Y=\{y_1, y_2, \cdots, y_m\}$ are non-infected machines. Unique peer connection averages for infected and non-infected machines are calculated as in Equation 4.1 and Equation 4.2 respectively.

$$Peer\ Average(Inf) = \frac{\Sigma_{i=1}^{n} Unique\ Peer\ IPs\ of x_i}{|X|} \tag{4.1}$$

$$Peer\ Average(NInf) = \frac{\Sigma_{i=1}^{m} Unique\ Peer\ IPs\ of\ y_i}{|Y|} \tag{4.2}$$

We measured the peer IP average for the infected machines for four major categories of alerts identified in the previous case (Torrent, Ransomeware, Scan, and Malware CnC) and compared with other machines. Figure 4.4 shows the average of unique connections (IP-IP) made by infected and non-infected machines.

[1]A machine is considered as infected if a Suricata alert is generated against the IP address of that machine

We can notice from the figure that infected machines, in general, have higher unique peer connections with external machines in comparison to a non-infected machine except a couple of instances. These instances refer to Ransomware type alerts which were not recorded on these days. This behavior is due to the following reasons.

1. Hosts running BitTorrent applications tend to have a quite large number of connections with external machines due to their Peer-to-Peer interactions.

2. Trojan virus or malware in the infected machines read user system data and leak it to external machines.

3. Malware daemon running in infected machines keeps sending packets containing various info including the local network structure to external machines.

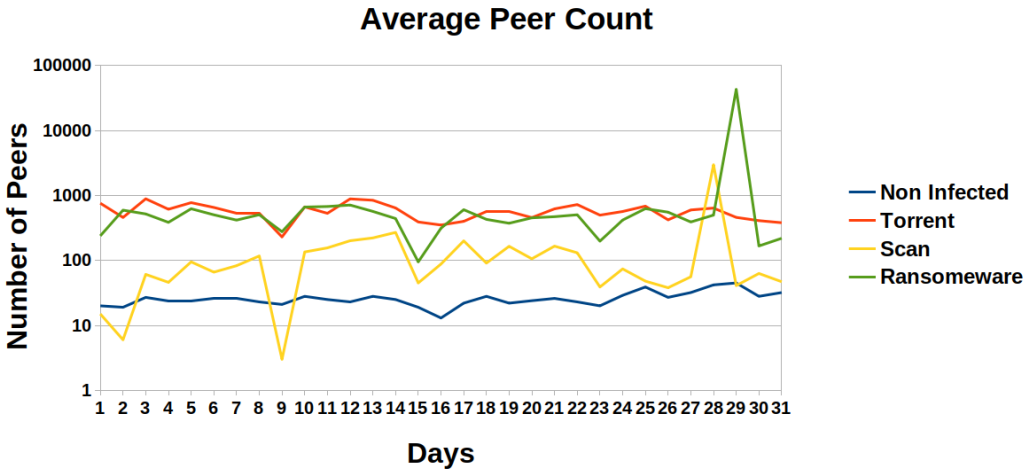

Figure 4.4: Overall Peer Connection Average

A sample communication graph of an end host infected with ransomware with its peers in a span of 24 hours is shown in Figure 4.5. This host has an IP address of 10.XX.XX.110 and it established communication with 311 different external hosts. Out of these, a particular IP 146.112.61.107 was of interest to us as this was being contacted by many other internal hosts. After manual examination, we found that this IP address was resolved to a domain owned by Open DNS. Our local DNS resolver has a recursive DNS entry to Open DNS resolver owned by CISCO

and it was resolved to *hit-malware.opendns.com* which screens the domain names contacted by end hosts to check if they are in the negative database (domains known to spread malware and other illegal software).

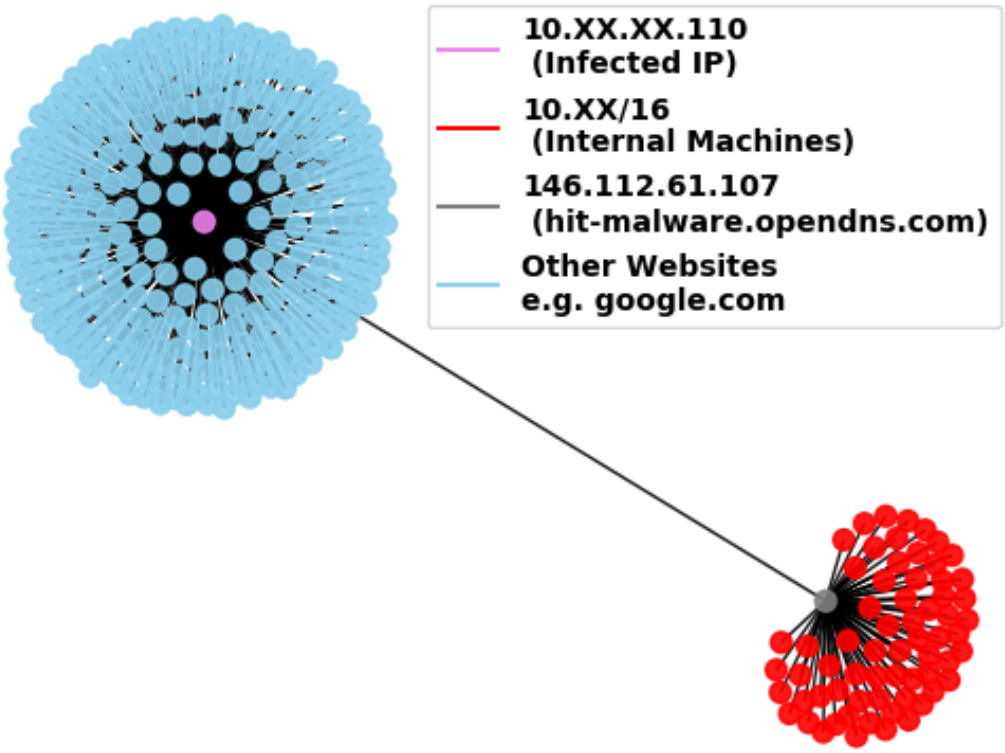

Figure 4.5: Ransomware Infected Machine Communication Network

4.5 Average Connection Per Hour

In this part of the experiment, we study the aggregate number of peer connections made by infected and non-infected machines during their active period. Unlike the previous case, if a machine connects with an external machine for K times then we count it K times. After getting the total connection count for every active IP in the network, we divide this connection count by the total active time within the day. To calculate the active period of a machine we have recorded the time when the firewall records the IP address for the first time during the day and last log time against that IP address. This method of calculating the active time or active hours of a machine is not accurate hence we made a change in the way it is calculated. In the logs, we measured the timestamp of every log against a particular IP address and calculated the time difference between the successive

log entries. If this difference is greater than 5 minutes then we consider this as an inactive time period for that machine and subtract this time period from 24 hours time for that day. This way we calculated the active time period for every machine (infected and non-infected) and calculated the volume of connections generated by each category. Equation 4.3 shows the calculation of connection volume per hour and Equation 4.4 depict the active time calculation for a day.

$$ConnectionsPerHour = \frac{Total\ Connections}{Total\ Active\ Time} \tag{4.3}$$

The total active time of the machine is calculated as in Equation 4.4.

$$ActiveTime = (FT - ST) - InT \tag{4.4}$$

where

1. ST: Time at which the first log entry is recorded by the firewall against a machine.

2. FT: Time at which the last log entry is recorded by the firewall against a machine.

3. InT: The time for which the machine was inactive between the first and last log entries by the firewall. This time is calculated in a multiple of 5 minutes. E.g. if the two successive log entries are 20 minutes apart then the total inactive time of 20 minutes will be deducted from 24 hours of the maximum active period.

We calculated the connection count for the active time period for both infected and non-infected machines from the logs. Figure 4.6 shows the active run time for infected and non-infected machines. We can notice that infected machines were more active during the day in comparison to other machines. This difference is also due to a large number of machines/hosts which connect to a network for a very short period of time.

Figure 4.7 shows how the Connection Per Hour vary for both categories of machines. We can notice from Figure 4.7 that, infected machines had several peer external connections (not unique) compared to non-infected machines as the

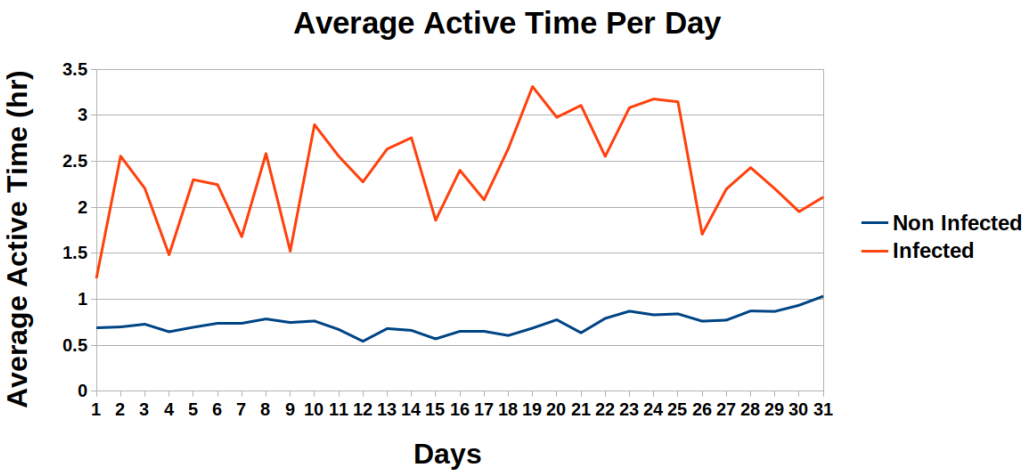

Figure 4.6: Machine Active Run Time

malware/daemon residing in the infected machine keep sending packets or data to the outside network.

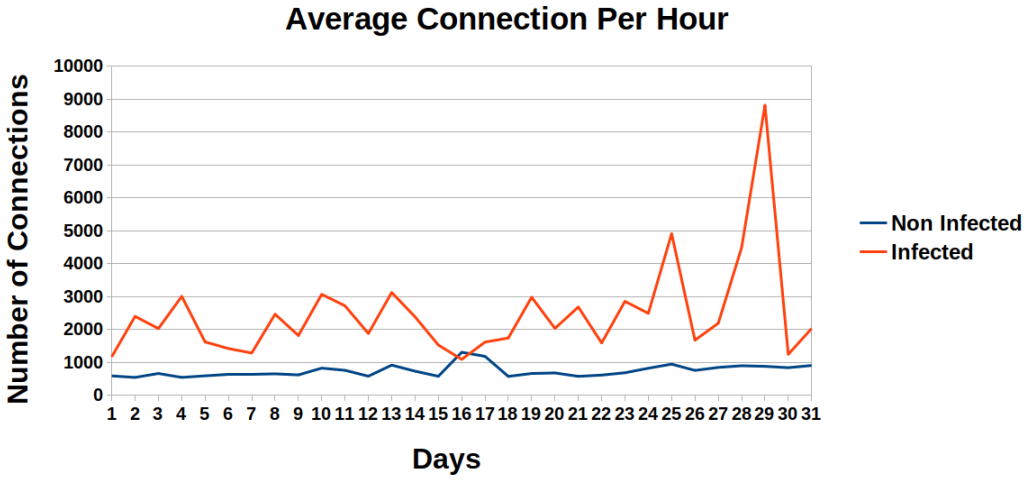

Figure 4.7: Average Connection Per Hour

4.6 DNS Name Resolution

In our next behavioral analysis, we studied the Domain Name System queries raised by both infected and non-infected machines. As mentioned earlier and shown in Figure 3.1, our DNS resolver is located in DMZ. Thus every DNS query passes through the firewall. We counted the DNS name resolution requests using port number 53 from the firewall logs. Figure 4.8 shows the difference in average

DNS queries raised by infected and non-infected machines as seen from the log analysis over the 31 days. We can see from the figure that the number of DNS queries generated by the infected machines is significantly higher in comparison to the non-infected machines. This behavior is justified as infected machines tend to contact many other machines, try spreading and contact malware domains and other infected machines. However, we noticed the following during our analysis.

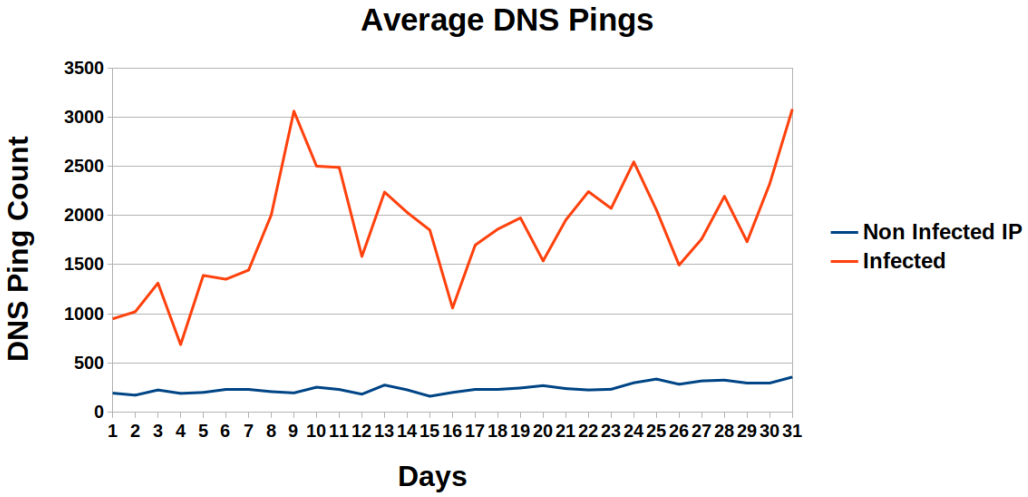

Figure 4.8: DNS Query Resolution

1. Machines that are infected with malware or running BitTorrent will establish connections without a name resolution.

2. Our DNS resolver is configured to block any repeated name resolution requests from a single client within a short span of time. As malware tends to generate repeated queries to the same domain names once this threshold is hit. Subsequent requests are not honored by the resolver. Thus the firewall had entries to only DNS queries but not for the responses.

4.7 Port Diversity

In this experiment, we study the diversity of ports used by the infected and non-infected machines. The rationale is to understand whether machines use a fixed set of ports for communication or there will be large diversity. In Figure 4.9 we have plotted the number of ports used by both types of machines (infected and

non-infected) for the 31 days. Again as in the previous cases, infected machines showed huge port diversity in comparison to machines which are not infected. This large diversity is mainly due to Peer-to-Peer connections opened by applications like BitTorrent clients that use random port numbers and this coupled with a large number of peer connections opened by machines running ransomware are attributed to this.

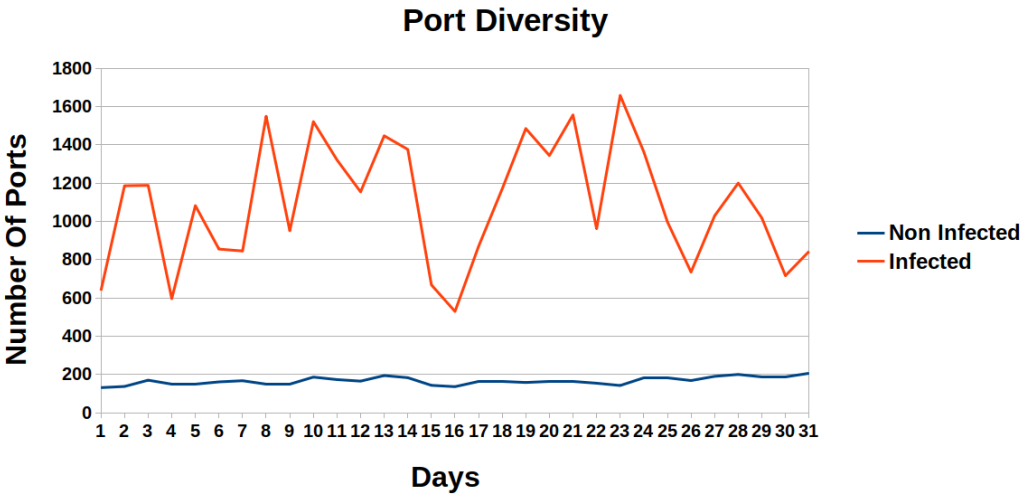

Figure 4.9: Port Diversity

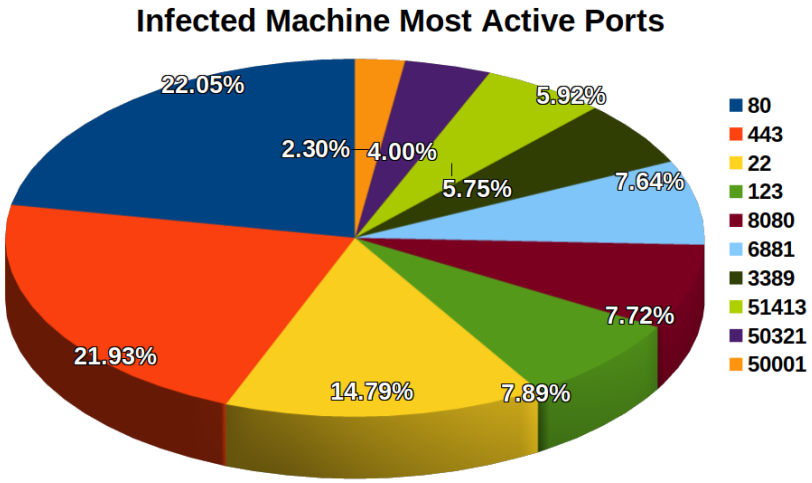

Figure 4.10: Infected Machine Most Active Ports Distribution

We also studied the usage of port numbers by both types of machines. Figure 4.10 and Figure 4.11 show the distribution of the top 10 ports used by infected and non-infected hosts. We can see from the Figure 4.10 that in infected machines use unconventional port numbers (51413, 50321, 50001, 6881) and possess a significant

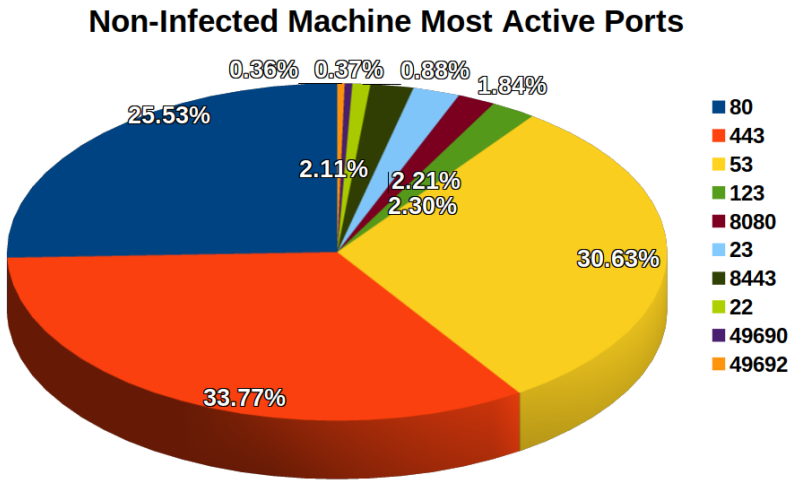

Figure 4.11: Non-Infected Machine Most Active Ports Distribution

share (19.86%) in comparison to other commonly used port numbers like web service (80), TLS (443) and SSH (22), etc. In the case of non-infected machines web service (80), TLS (443) and DNS (53) have a significant share (89.9%).

4.8 Recurrent Connections Across Days

$$Overlap(Inf) = \frac{\Sigma_{i=1}^{n} Unique\ Peer\ IPs\ of\ x_i\ on\ day\ D_1 \bigcap D_2}{|X|} \tag{4.5}$$

$$Overlap(NInf) = \frac{\Sigma_{i=1}^{m} Unique\ Peer\ IPs\ of\ y_i\ on\ day\ D_1 \bigcap D_2}{|Y|} \tag{4.6}$$

In this experiment, we study the infected and non-infected machines' recurrent connections across days. Infected machines being part of some command and control or Peer-to-Peer group tend to be exchanging regular or frequent communication (which is often periodic) with their peers. If the domain names or the peers to which they are connecting to are fixed they will show these repetitive or recurrent network connections. While this is true about infected machines others (non-infected machines) may not visit the same website or establish a connection with the same external IP every day (with few exceptions like update managers). Even if a user has a particular behavioral tendency then also these overlaps will not be significant. E.g. let's assume that a particular user visits news websites

every day then also these website visits may be redirected to different IP addresses as such sites will feed the content through Content Delivery Networks (CDNs) and the servers often use load balancing. In order to understand the extent of such

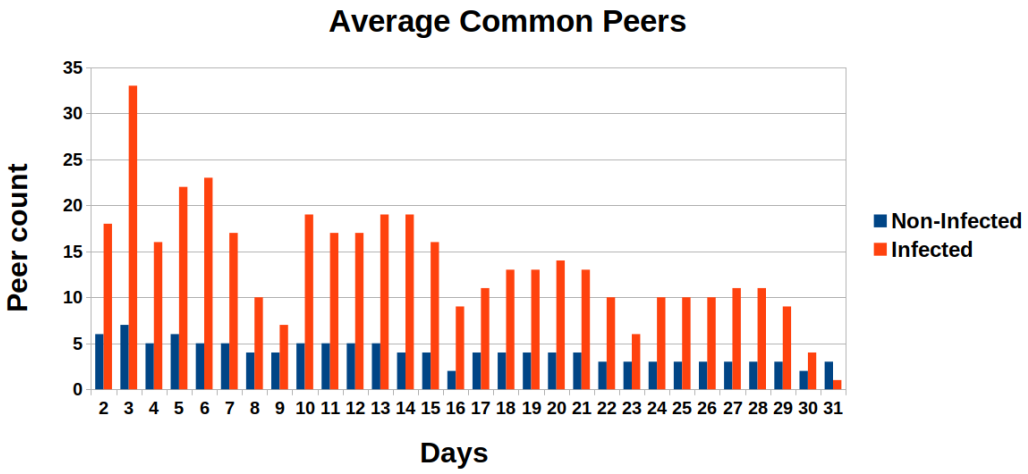

Figure 4.12: Average Common Peers for Infected and Non-Infected Hosts

repetitive connection behavior, we studied the number of overlaps seen in peer IP addresses of both infected and non-infected machines.

Let $X=\{x_1, x_2, \cdots, x_n\}$ be the set of internal machines which are identified as infected and $Y=\{y_1, y_2, \cdots, y_m\}$ are non-infected machines. The recurrent peer connection averages for infected and non-infected machines are calculated as in Equation 4.5 and Equation 4.6 respectively. We use the counts of unique peer IP addresses of a machine from the infected and non-infected machines. One example histogram plot for infected and non-infected machines for day 1 and its recurrence pairs identified across all the remaining 30 days is shown in Figure 4.12. We can notice that infected machines indeed have a fairly large number of overlapping or recurrent IP addresses in comparison to non-infected machines. Similar behavior is observed for other days too. We omit to show all other cases for space constraint reasons.

Chapter 5

Log Visualization

In this chapter we describe a log visualization tool *NViZ*. *NViZ* can show interactive visualization of logs for different types of analysis from three different sources namely DNS, IDS, and Firewall. These visualization plots help gain different insights. Log parsing and analysis are done with custom Python scripts and the user interface is developed with the MEAN stack development framework. We present the features available with *NViZ* in 6 subsections below.

5.1 Common Functionalities

NViZ visualization tool has a main control panel which allows users to navigate and select various functionalities. Figure 5.1 is a snapshot showing different features of the tool. The main features includes the following.

A. Navigation Bar: Navigation bar allows the users to navigate through differ-

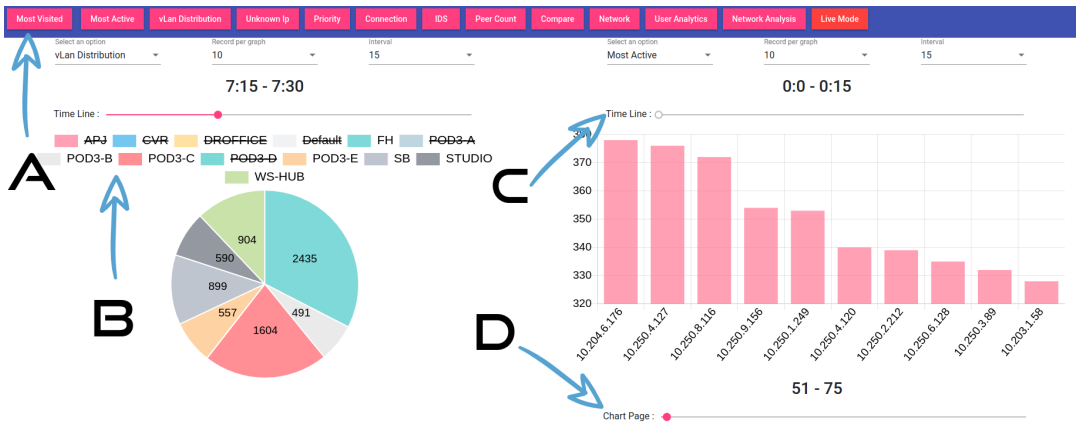

Figure 5.1: Visualization Tool

ent visualization charts.

B. Pie-Chart Legends: *NViZ* can generate Pie charts for visualizing various activities and events. User can select and eliminate certain portion of the data from pie-chart by clicking on respective legend.

C. Timeline Controller: Different visualization charts are rendered by reading the log data within a window period. This slider bar allows to select a time interval for which user would like to see the visualization plot. For e.g., in the Figure 5.1 right hand side has a time window of 00:00 A.M. to 00:15 A.M is selected.

D. Chart Page Controller: Large volumes of data makes it hard to visualize within the limited space. For e.g., if the entries on X-axis are 1000 within the selected window time then such large ticks on X-axis are split into different pages using descending order of values on Y-axis. One can navigate through different pages using this controller and see different pages.

5.2 DNS Log Analysis

Our university network is divided into different VLANs encompassing various buildings. Different buildings on the campus have different IP address ranges that are assigned through DHCP leases. Any DNS query coming from a host can identify the location of the host. Using these logs *NViZ* can generate four different visualization plots as below.

1. **Most Popular Websites**: *NViZ* can identify the most popular websites

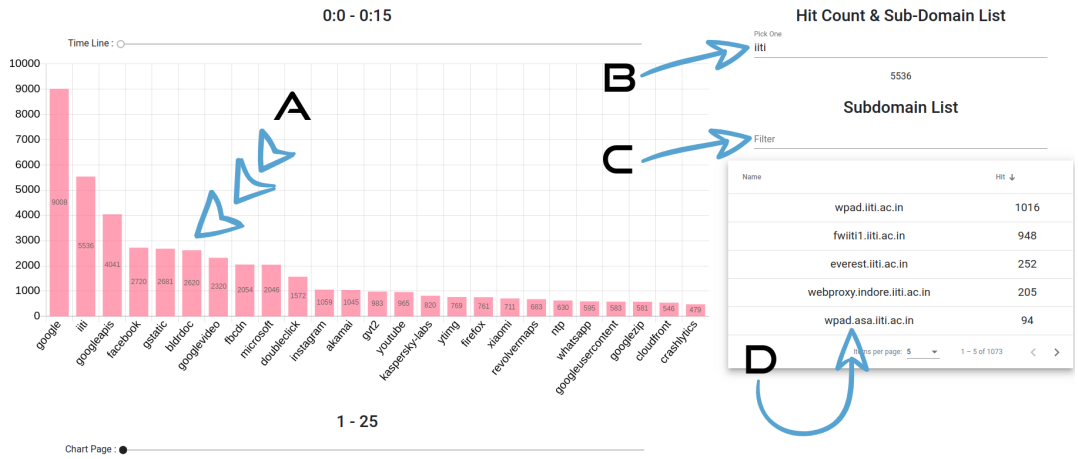

Figure 5.2: Most Popular Websites

which are visited within the time window selected by the user. This is done by parsing the DNS logs. Each DNS query has a domain name that it wishes to resolve and by collecting these details it can generate an interactive plot. In Figure 5.2 we can see this data being visualized in two parts as the chart(A) where X-axis has labels of domain names and on the Y-axis we have the number of hits received within the specified time interval for that domain. This allows the network administrator to know which high-level domains are popular in what interval of time and this can help the administrator manage the network better. Users can also select a domain name by selecting it from the select box(B). This functionality has an auto-complete feature. Once the user selects a high-level domain, she can see the number of hits for that particular high-level domain along with the number of hits of its sub-domains in the form of table (D). Filter input(C) allows the user to filter the table data based on the input keyword.

2. **Most Active Users**: The second type of visualization done using the DNS

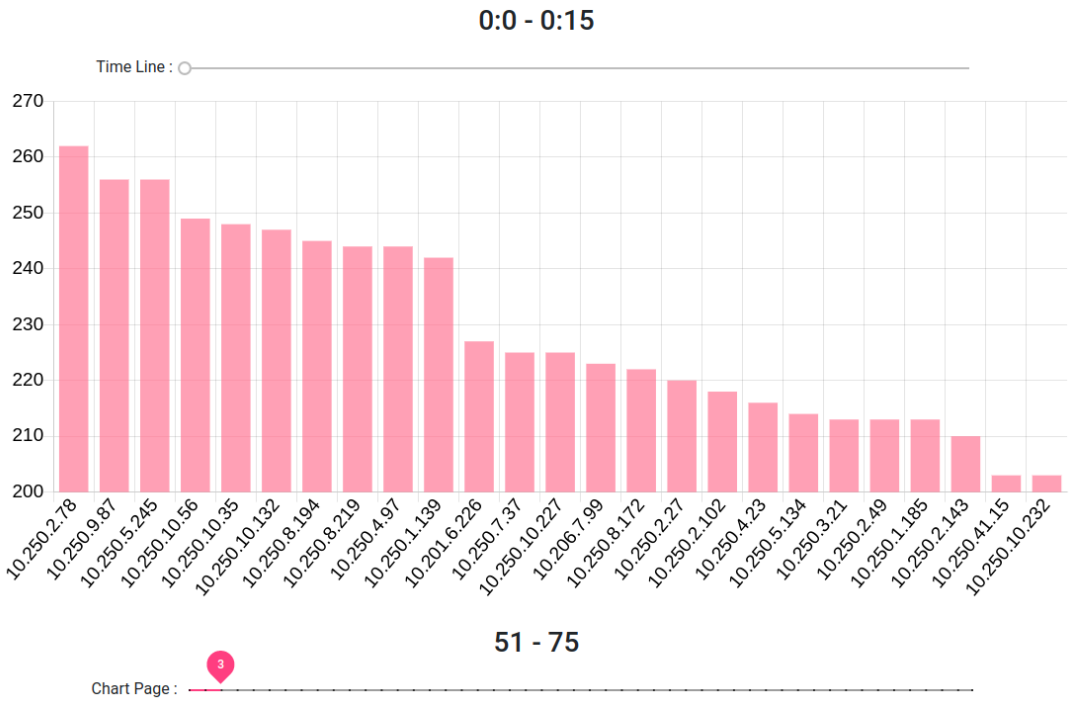

Figure 5.3: Most Active Client

logs is to find the most active users. Figure 5.3 is a bar-graph where the X-axis has the client IP addresses and on the Y-axis the number of DNS queries resolved by DNS resolver for that IP address is shown. As each IP address is assigned to a device (with a large lease period of 8 days) it can be thought of as unique for

a user. This information can be used for various network management activities like finding the devices which are sending unusual traffic to the DNS server.

3. **DNS Query Distribution Over Virtual-LAN**: Figure 5.4 is a pie-chart

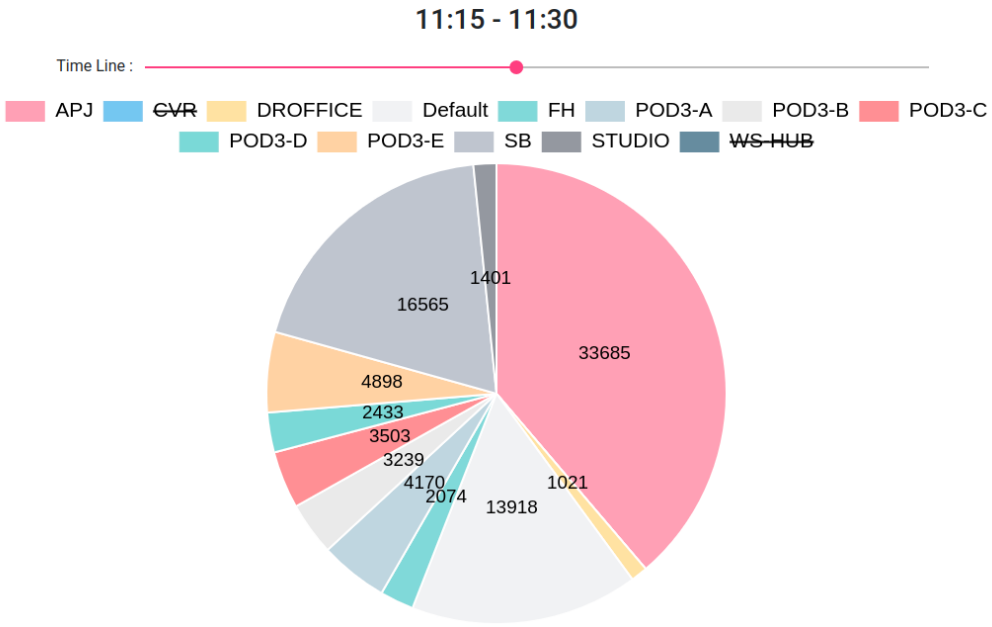

Figure 5.4: Virtual Lan Division

showing the distribution of the number of DNS requests received from different virtual LANs for the selected interval of time. A certain portion of data can be removed from the chart by clicking on the respective legend. This visualization helps to monitor the traffic received from different virtual LANs and can be used to identify anomalies which may require the network administrator's attention.

4. **DNS Query Distribution Over Unknown IPs**: Figure 5.5 is a pie-chart showing the distribution of the number of DNS requests received from different machines that do not belong to any of the defined Virtual-Lan for the selected interval of time. This visualization helps to monitor the traffic received from machines which are using static IPs.

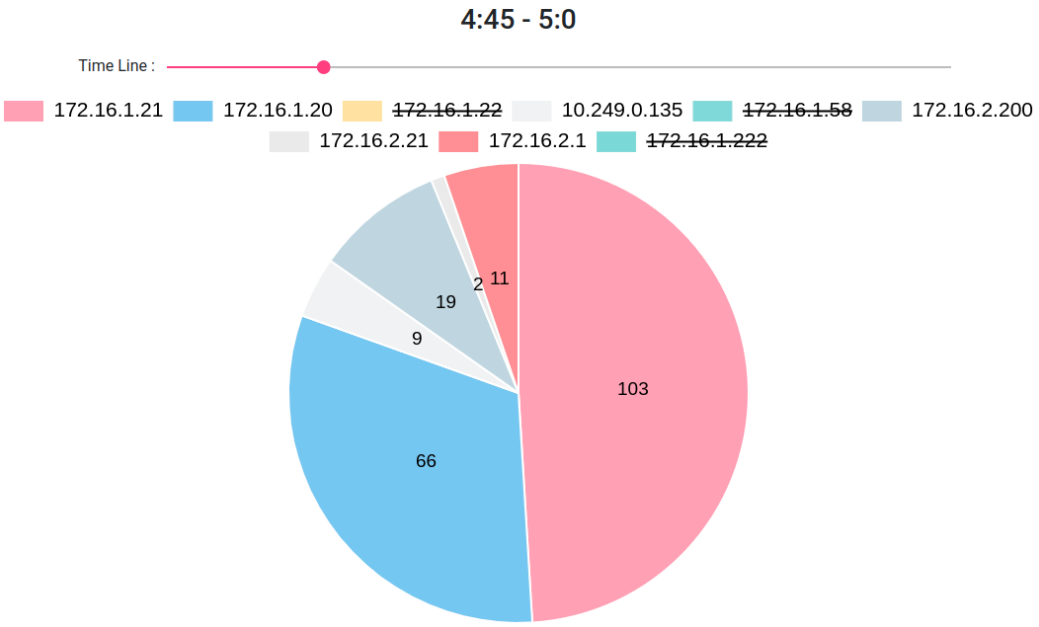

Figure 5.5: Unknown IPs Division

5.3 Intrusion Detection System Alert Visualization

Second type of logs *NViZ* can visualize is of IDS. Currently it can show graphs for the following visualizations.

1. **IDS Alerts Classification by Priority**: Suricata IDS alerts/logs have a

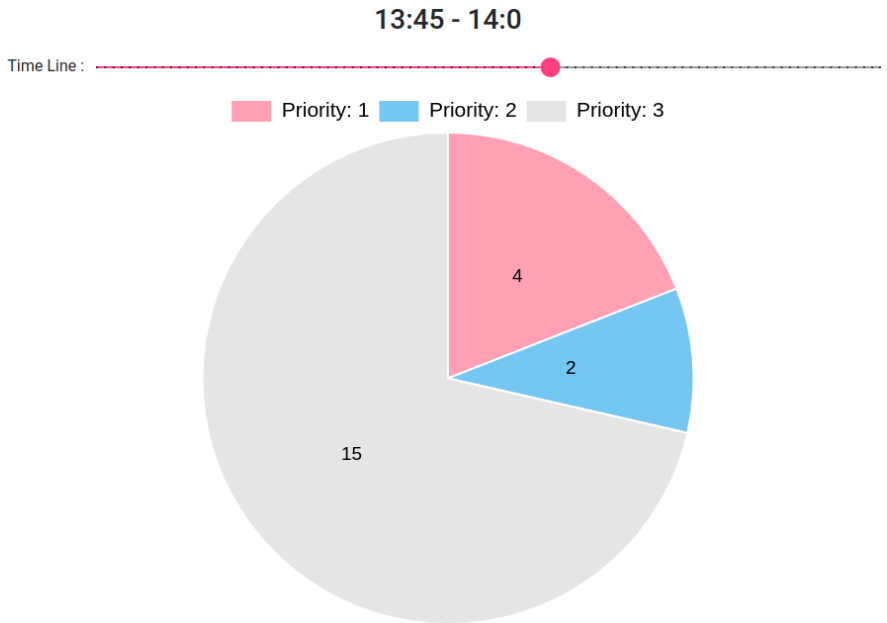

Figure 5.6: Priority Categorization

priority number with them. One visualization that can be done with *NViZ* is grouping the alerts of different priority type and showing the distribution using Pie-chart legend. Figure 5.6 shows this visualization. This will help a system administrator to gain insights about types of alerts being generated.

2. **Distribution of IDS Alert Types**: This visualization shows the distribu-

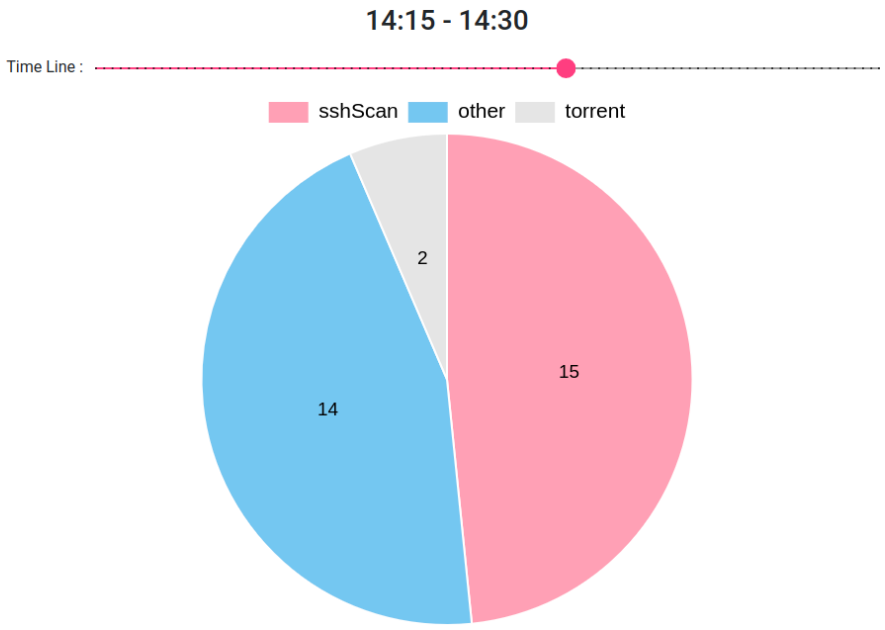

Figure 5.7: IDS Alert Types

tion of different types of alerts for a time interval using Pie-chart legend. From this visualization different infection types can be seen. Figure 5.7 is a snapshot of this visualization.

5.4 Firewall Log Visualizatin

Third type of logs *NViZ* can visualize is of firewall logs. The following three types of visualizations are possible with firewall logs.

1. **Connection Request Pass/Block from Users**: Every log entry generated by the firewall has a field that indicates whether the connection originating from an IP address was allowed or blocked. Connections from all the internal IP addresses are visualized as a bar graph with an X-axis showing the IP address and Y-axis indicating the count of such cases. There are two such bars for every IP

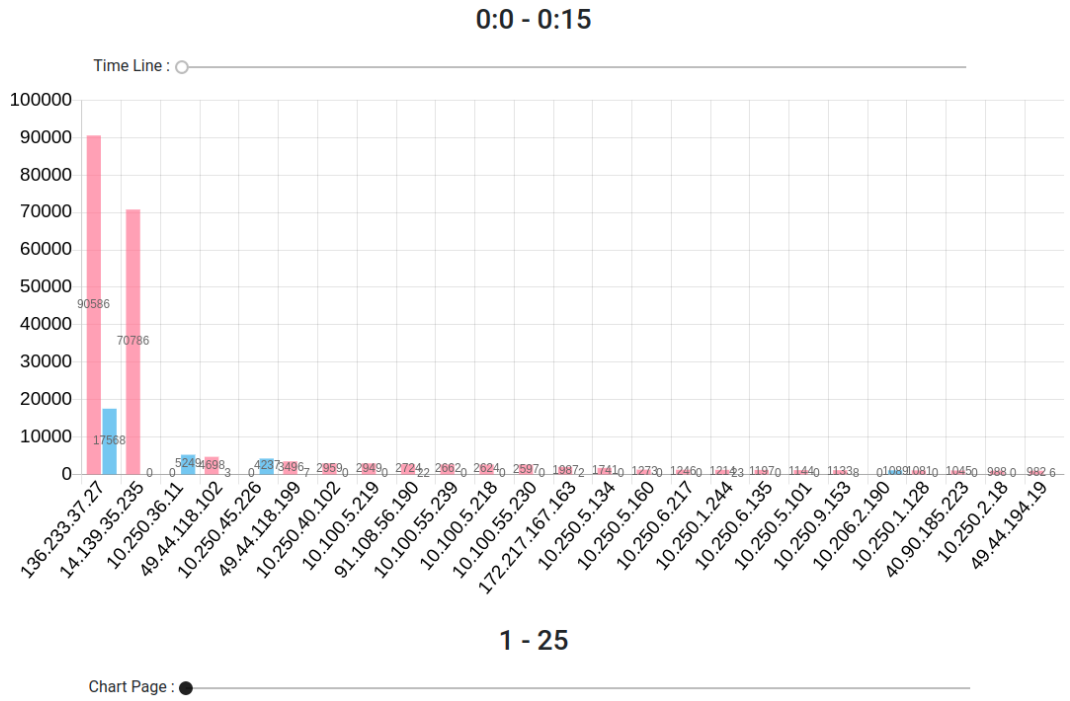

Figure 5.8: Connection Pass/Fail Per IP

address one for the "pass" case and second for the "block" case. Figure 5.8 is a snapshot of this visualization.

2. **End Host Level Peer Connections**: The firewall pfSense records every

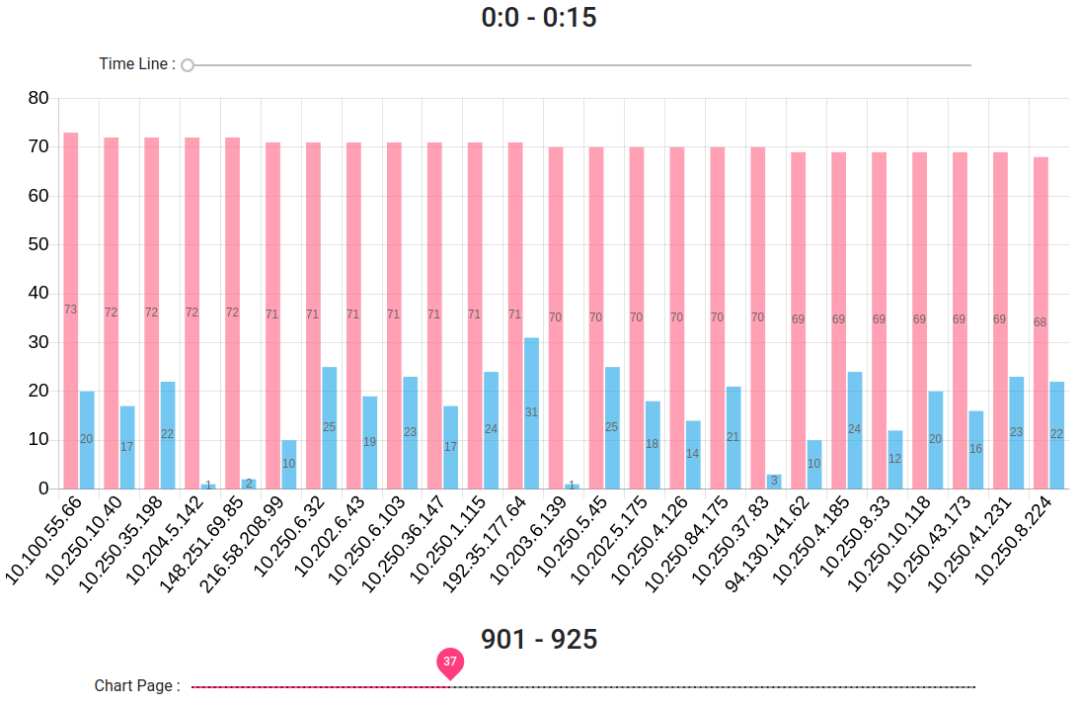

Figure 5.9: Peer Connection

connection from an internal machine to external machine(s) where an external

machine can be a website, a server or any other public IP address. This visualization helps in understanding the peer connection patterns of an internal host or a user. This peer connectivity is rendered as a bar graph.

Figure 5.9 has a snapshot of a bar-graph where on the X-axis we have the IP address of the devices and on Y-axis we have the hit count. The red-colored bar shows the total number of connections from that IP address and the blue-colored bar represents the unique peer connections.

3. **Network Connection Visualization**: The Third type of visualization that

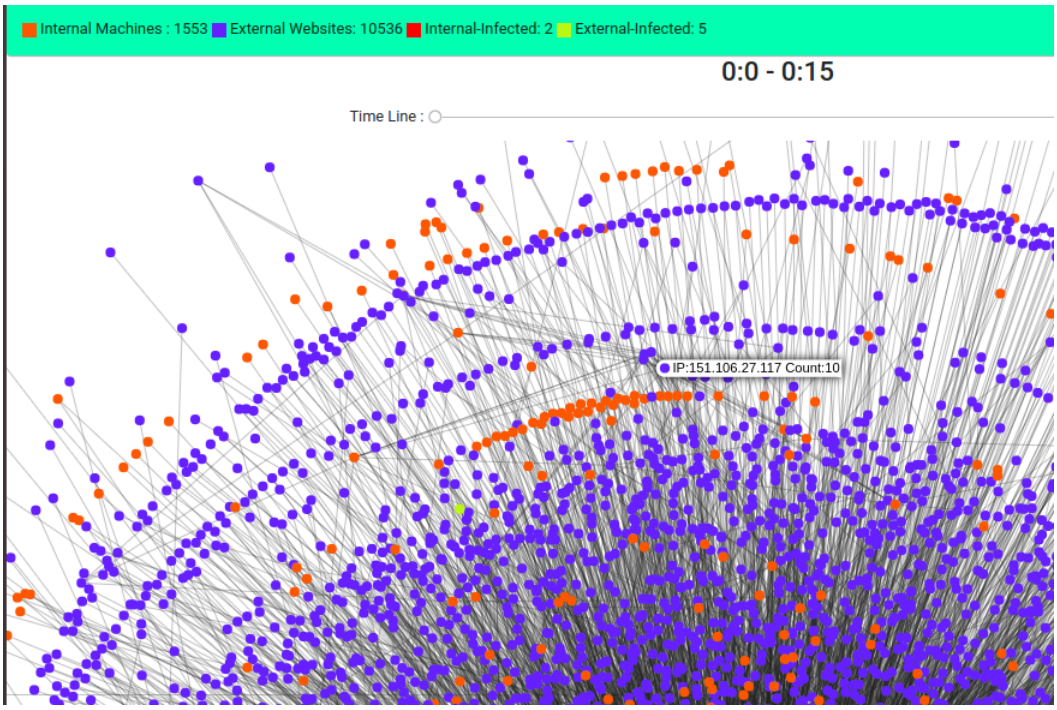

Figure 5.10: Communication Visualization

can be done with *NViZ* using firewall logs is showing the entire set of communications from all internal hosts within a window period. A snapshot of 15 minutes of communication is shown in Figure 5.10. Each node in this graph represents one host and nodes are color-coded with blue and red colors. Red-colored nodes are the ones against that IP address an IDS alert is generated and is infected. Blue colored nodes are normal nodes.

5.5 User Activity Visualization

NViZ can also visualize the firewall, intrusion detection/prevention system and DNS resolver for a specific user. Currently it can show graphs for the following visualizations.

1. **DNS and Firewall Logs Visualization**: *NViZ* can generate three graphs

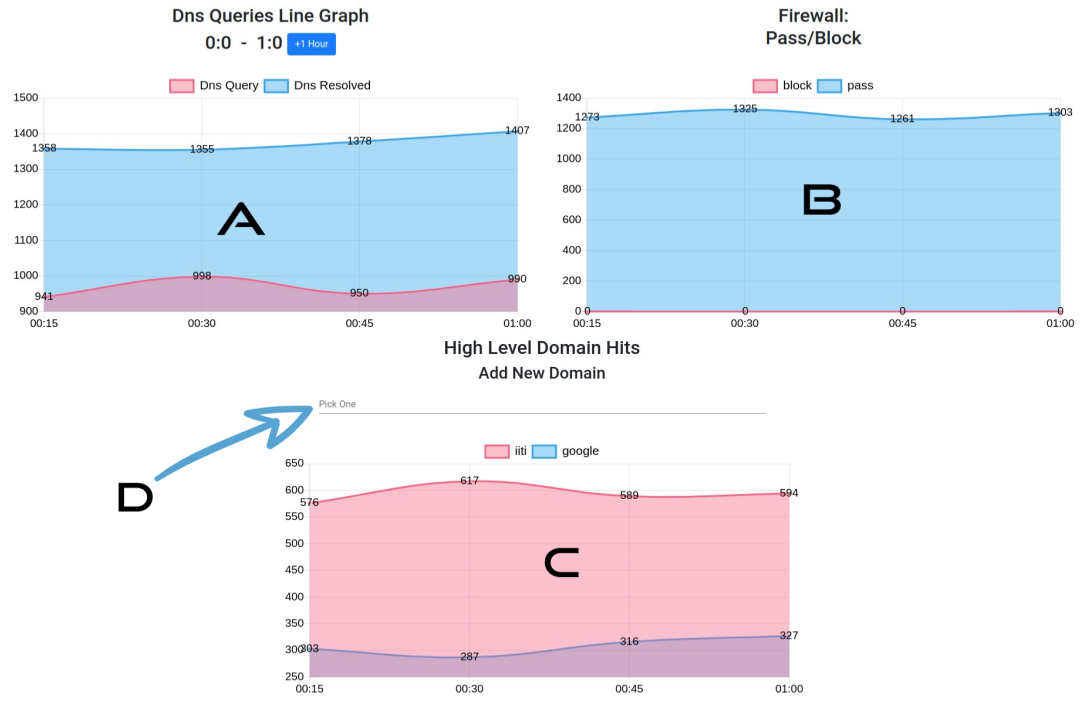

Figure 5.11: User Activity Analysis

for a specific user for the time window selected. This is achieved by correlating the results obtained from parsing the DNS and Firewall logs. Figure 5.11 has the following graphs:

A. DNS Queries Line Graph: This visualization has timestamp labels on the X-axis and on Y-axis we have the hit count for DNS queries sent and DNS queries resolved within the specified time interval.

B. Firewall: Pass/Block: Connections from the user are visualized as a line graph with an X-axis showing the timestamp and Y-axis indicating the count of passed and blocked connections within the specified time interval.

C. High Level Domain Hits: In this visualization, where the X-axis has timestamp labels and on the Y-axis we have the number of hits received by selected domains within the specified time interval. A network administrator can add a

new domain using the select box(D). This functionality has an auto-complete feature.

2. **Network Connection Visualization**: Another type of visualization that

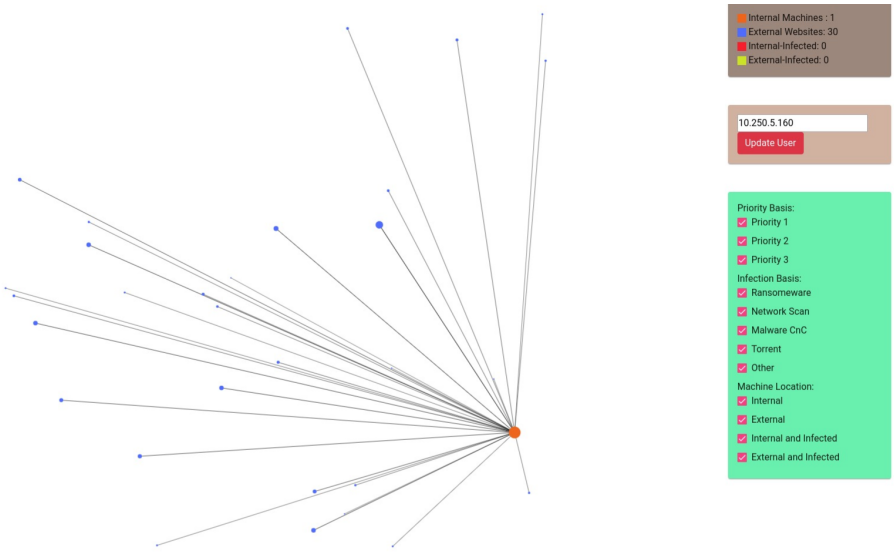

Figure 5.12: User Communication Visualization

can be done with *NViZ* using firewall logs is showing the entire set of communications from a particular host within a window period. A snapshot of 15 minutes of communication is shown in Figure 5.12 along with a few other functionalities. Each node in this graph represents a machine and nodes are color-coded with blue, orange, red and yellow colors. Red-colored nodes are the internal machine against which an IDS alert is generated. Orange represents the normal internal machine. Blue colored nodes are normal external websites whereas yellow nodes represent the external machine against which an IDS alert has been raised. A node size differs based on the number of connections made to the machine, higher the number is bigger the node is. Filter Box(B) can be used by a network administrator to hide/show a particular type of machine.

5.6 Real-Time Visualization

NViZ can also render different kinds of visualization in real-time. It keeps track of the log file in which the network security system is currently writing using the *tail* program. Figure 5.13 shows the following visualization:

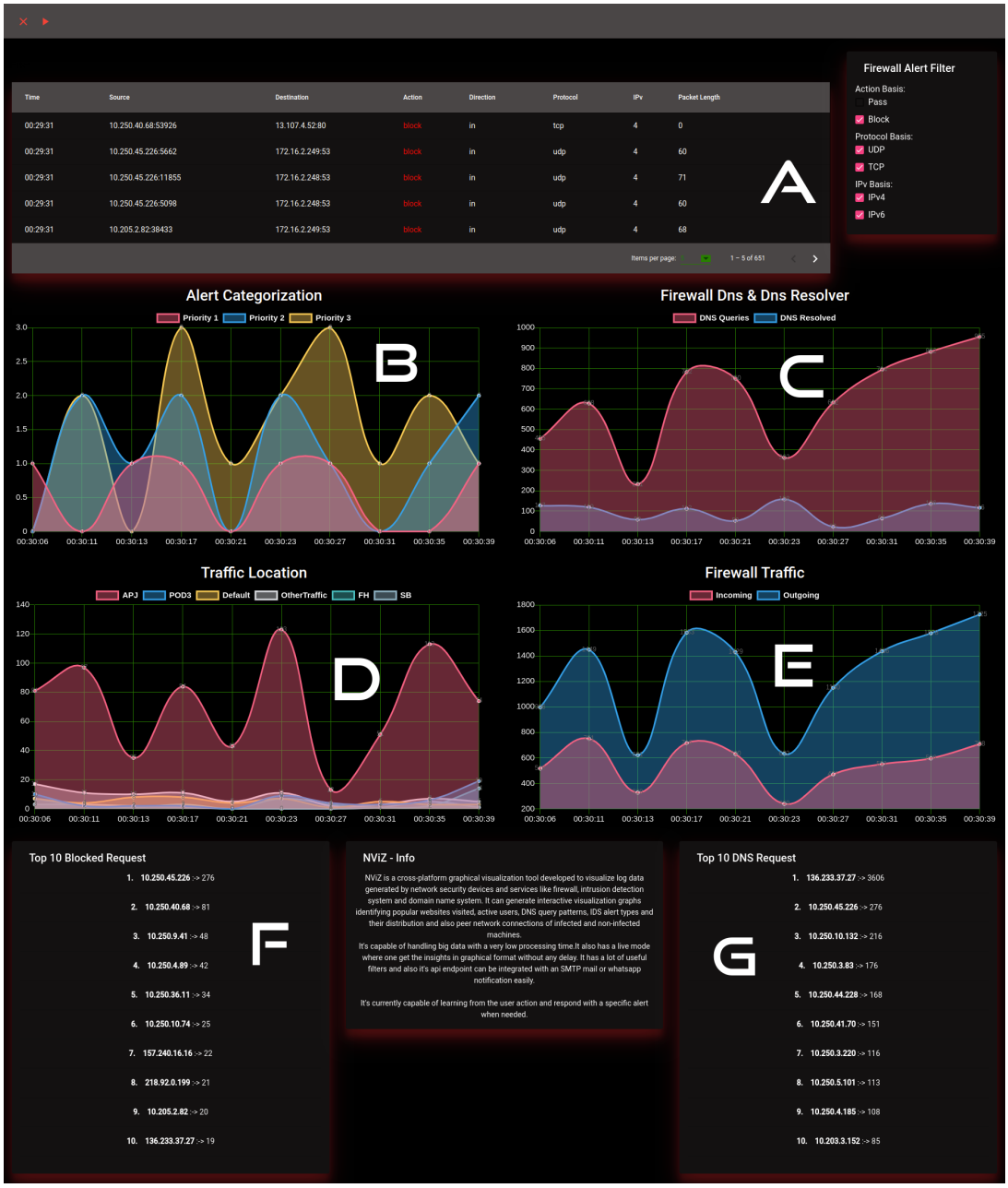

Figure 5.13: Real-time Visualization

A. Firewall Logs Filter: A huge amount of firewall logs are generated in real-time but most of the time network administrator is only interested in a particular type of events. This visualization can be used to filter firewall logs based on the IP version, firewall action, and protocol type.

B. Alert Categorization Based on Priority: One visualization that can be done in real-time with *NViZ* is grouping the alerts of different priority types and showing the distribution using a line graph where the X-axis has timestamp labels and on the Y-axis we have the number of alerts of different priority number.

C. DNS Query & DNS Resolved: This visualization has timestamp labels on the X-axis and on Y-axis we have the hit count for DNS queries sent and DNS queries resolved in real-time.

D. Traffic Location: This visualization shows the distribution of the number of DNS requests received from different virtual LANs in real-time. This visualization helps to monitor the traffic received from different virtual LANs and can be used to identify anomalies that may require the network administrator's attention.

E. Firewall Traffic: Connections from all the internal IP addresses are visualized in real-time as a line graph with an X-axis showing the timestamp and Y-axis indicating the number of incoming and outgoing connections.

F. Top 10 Users Based on Blocked Request: This visualization list the top 10 users in decreasing order of the number of requests blocked by the firewall. This information can be used by a network administrator to find infected machines.

G. Top 10 Users Based on DNS Query: This visualization list the top 10 users in decreasing order of the DNS requests sent. This visualization allows a network administrator to keep track of the machines sending high traffic to the DNS Server in compare to a normal user.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Perimeter security appliances and network services generate logs about various events. These logs are consulted by network administrators to identify infected machines, vulnerable systems, misconfigurations, etc within their networks. In this project, we report on a log analysis study generated by production level security appliances. We analyzed these logs using different parameters to find the end host-level behavioral trends. Our analysis successfully identified clearly differentiated behavior of infected and non-infected hosts. In particular, our analysis establishes that infected machines have a higher number of external peers, a large number of connections, more DNS queries, higher port diversity, and common peers over a period compared to their non-infected counterparts. Further our study reveals that a small number of IDS alert types constitute a majority share. We intend to use these behavioral characteristics to propose anomaly detection systems, early warning systems and also in developing easy to use data visualization techniques that will be deployed and tested in real networks.

We also presented $NViZ$, a cross-platform graphical tool to analyze and show logs generated by perimeter security devices like firewall, IDS and DNS resolver. The tool has the capability to draw more than fifteen visualizations from the logs. The tool is interactive and allows comparison between two types of graphs for gaining insights.$NViZ$ can also be used for visualization in real-time. It allows the network administrators to get a complete view of the activity and actions

which are taking place in the entire network using various visualization format.

6.2 Future Work

1. Run the automated log analysis on a different dataset and compare the trends and observation obtained
2. Add a BotNet attack visualization feature in $NViZ$
3. Convert $NViZ$ into a smart visualization tool by adding features like alert pop-up, auto IP blocking, etc.

Bibliography

[1] Biplob Debnath, Mohiuddin Solaimani, Muhammad Ali Gulzar, Nipun Arora, Cristian Lumezanu, Jianwu Xu, Bo Zong, Hui Zhang, Guofei Jiang, and Latifur Khan. Loglens: A real-time log analysis system. In *IEEE 38th International Conference on Distributed Computing Systems*, ICDCS '18, pages 1052–1062. IEEE, 2018.

[2] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1285–1298. ACM, 2017.

[3] Logrythm. `https://logrhythm.com/`. Online; accessed 4 September 2019.

[4] Jie Chu, Zihui Ge, Richard Huber, Ping Ji, Jennifer Yates, and Yung-Chao Yu. Alert-id: Analyze logs of the network element in real time for intrusion detection. In *Proceedings of the 15th International Conference on Research in Attacks, Intrusions, and Defenses*, RAID'12, pages 294–313. Springer-Verlag, 2012.

[5] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Beyond blacklists: Learning to detect malicious web sites from suspicious urls. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 1245–1254. ACM, 2009.

[6] Zhang Jiawan, Li Liang, Lu Liangfu, and Zhou Ning. A novel visualization approach for efficient network scans detection. In *SECTECH '08: Proceedings of the 2008 International Conference on Security Technology*, pages 23–26. IEEE Computer Society, 2008.

[7] Zhongyang Kan, Changzhen Hu, Zhigang Wang, Guoqiang Wang, and Xiaolong Huang. Netvis: A network security management visualization tool based on treemap. In *ICACC'10: Proceedings of the 2nd International Conference on Advanced Computer Control*, pages 18–21. IEEE, 2010.

[8] A. Sharma, Z. Kalbarczyk, R. Iyer, and J. Barlow. Analysis of credential stealing attacks in an open networked environment. In *Proceedings of the 2010 Fourth International Conference on Network and System Security*, NSS '10, pages 144–151. IEEE Computer Society, 2010.

[9] Wikipedia contributors. Pfsense — Wikipedia, the free encyclopedia, 2019. [Online; accessed 5-December-2019].

[10] Wikipedia contributors. Suricata (software) — Wikipedia, the free encyclopedia, 2019. [Online; accessed 5-December-2019].