Leveraging Data from Open-Source Intrusion Detection Systems for Enhancing Security of Systems

Viren Chhabria

A Dissertation

Presented to the University of Dublin, Trinity College

in partial fulfilment of the requirements for the degree of

Master of Science in Computer Science (Data Science)

Supervisor: Dr. Stephen Farrell

August 2019

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Viren Chhabria

August 14, 2019

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Viren Chhabria

August 14, 2019

To my mother, Bhavna R Chhabria and father, Rajkumar T Chhabria, for their endless love and support!

Acknowledgments

I would like to express my sincere gratitude to my supervisor Dr. Stephen Farrell for his constant support and supervision. His knowledge and experience helped me overcome challenging situations during the course of this dissertation.

A big thank you to my parents and family for enabling me to pursue my MSc program at Trinity College Dublin and supporting me throughout this wonderful journey.

I would like to thank Prof. Khurshid Ahmad, for mentoring, motivating and supporting me.

Last, but not the least, I would like to thank my friends. Ankita for motivating me throughout the project. Arun and Debrup for providing me innovative ideas and technical guidance. Rohit, for helping me with machine learning during the project. Dr. Husanbir Singh Pannu for motivating me and providing tips for efficient writing.

Viren Chhabria

University of Dublin, Trinity College
August 2019

Leveraging Data from Open-Source Intrusion Detection Systems for Enhancing Security of Systems

Viren Chhabria, Master of Science in Computer Science

University of Dublin, Trinity College, 2019

Supervisor: Dr. Stephen Farrell

Bad actors are omnipresent across the Internet trying to gain unauthorized access to systems for malicious activities. Intrusion Detection Systems(IDS) are like burglar alarms in computer science designed the detect and prevent intrusion attempts. This makes them a widely researched and important topic in the field of computer science. In this research, 26 months of historical data gathered from live internet facing servers running popular open-source IDS - Fail2Ban and DenyHosts is analyzed. A scoring mechanism ranging from 0-4 was devised to associate a threat level to each intrusion detected. Exploratory Analysis for finding geo-spatial, temporal and other underlying patterns in the data. Predictive Analysis was carried out using supervised machine learning techniques like time-series forecasting and classification to verify whether it is possible to efficiently predict intrusion attempts or the threat associated with them. The result was a software pipeline being created to ingest data from IDS and produce a security dashboard reflecting the security state of the systems and patterns in the attacks. The predictive analytics results were interesting. The time-series forecasts generated using fbProphet were inaccurate, highlighting the uncertainty and variability in attacks. The XGBoost classification model was able to predict the threat level associated with each attack with 75.46% accuracy on the validation set.

Contents

List of Tables

List of Figures

Acronyms

API Application Programming Interfac.

ARIMA Auto Regressive Integrated Moving Average.

ASN Autonomous System Number.

AUC Area Under the Curve.

CSV Comma Separated Values.

CVE Common Vulnerabilities and Exposures.

DNSBL Domain Name System-based Blackhole List.

DSL Domain Specific Language.

GDPR General Data Protection Regulation.

HTTP HyperText Transfer Protocol.

IANA Internet Assigned Numbers Authority.

IDS Intrusion Detection Systems.

IETF Internet Engineering Task Force.

IMAP Internet Mail Access Protocol.

IP Internet Protocol.

JSON JavaScript Object Notation.

MBOX MailBOX.

MERROR Multi-ClassError Rate.

MTA Mail Transfer Agent.

RDAP Registration Data Access Protocol.

ReST Representational State Transfer.

RMSE Root Mean Square Error.

SASL Simple Authentication and Security Layer.

SMTP Simple Mail Transfer Protocol.

SSH Secure Shell.

UUID Universally Unique IDentifier.

VM Virtual Machine.

VPS Virtual Private Server.

WHOIS Who Is.

XGBoost eXtreme Gradient Boosting.

Chapter 1

Introduction

Imagine a huge farm, generating livelihood for a farmer (referred to as farmer A), built with a lot of resources, time, money, hard work and patience. It's a pleasing picture in this day and age, isn't it? This farm has the best soil and the farmer has unique techniques to grow all kinds of fruits, vegetables and other produce. A lot of other farmers, the adversaries, despise that this farmer A is able to produce such a variety and gets a lot sale and customers. The adversaries decide to intrude into farmer A's farm, steal a sample of the soil to figure out how this farm is able to produce such high quality and variety consistently. If farmer A doesn't fence his farm and setup traps on the perimeter of the farm, it would be vulnerable to losing its crop and uniqueness obtained by the intelligence and hard work of the farmer. Therefore it is essential for farmer A to ensure a secure perimeter is setup around the farm to safeguard it from intruders. Similar to the adversaries, there are bad actors frequently scouting the internet for malicious activities. Malicious activities could vary from defacing websites, stealing data, virus propagation to competition or politically motivated attacks. IBM "Cost of a Breach" study 2019 found that malicious attacks are not only a major cause of data breach, but also the most costly. According the study a staggering 48% of the breaches are a consequence of a malicious / criminal attack, and they cost approximately $157 per user as shown in Figure 1.1 and Figure 1.2 [2, 3].

The smartphone revolution aided by improved availability of the internet with falling data prices has lead to a larger number of users accessing the web. The rise in the usage of the internet also also requires security to be the key foundation of

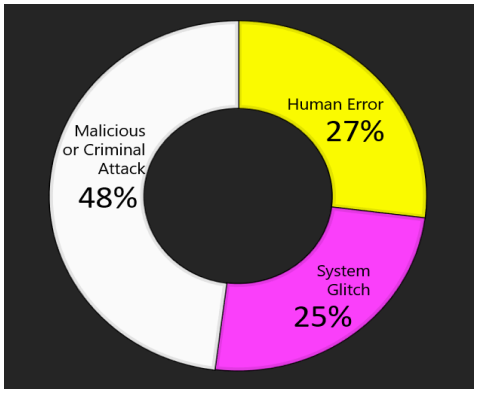


Figure 1.1: Root causes of data breaches

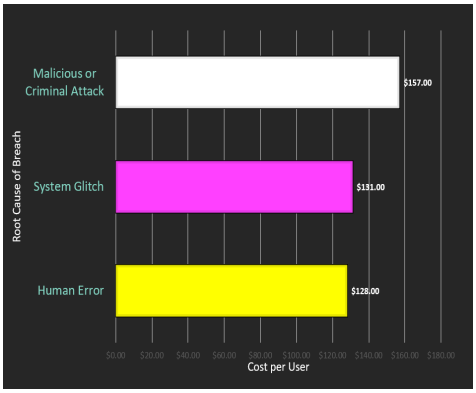


Figure 1.2: Cost per captia of data breach

any service/application. Larger number of user on-boarding results in massive data volumes being gathered across different domains. The data being collected at different levels is leveraged by firms to perform analytics and advertising. Rampant use of user data with or without the consent of respective users causes serious breaches into the privacy of a user. To curb such practices and regulate the use of data, the General Data Protection Regulation (GDPR) was introduced in May 2018 across the European Union (EU) [4, 5].

The wealth of information in the data makes it a promising target for attackers. In January 2019 alone, **1.76 billion** user records were leaked and the cost of cybercrime in 2019 is expected to cross **$2 trillion** [3, 6]. As the numbers indicate, an intrusion in the internet can cause major disruptions to data security and privacy[7]. IDS play a major role in preserving the computer security policies of Confidentiality, Integrity and Availability [8].

Internet facing servers (similar to the farm) are a major target for bad actors. They rely on weak security, easy-to-guess reused passwords or open ports to breach systems.

Remote services running on these machines, such as Secure Shell (SSH) for remote access and Simple Mail Transfer Protocol (SMTP) for mail relay are primary targets of the attackers since they are largely required and have common access ports, unless explicitly changed.

In the present day and age, where there is an emphasis on security, it is critical to secure systems in all domains. As technology has advanced, so has the skill-set of

bad actors. In earlier times, security was primarily a concern of system administrators, whereas, now, barely anyone would think of running an unprotected computer [9]. It is important that intrusions into a system are duly detected and prevented. Intrusion Detection can be defined as the problem of identifying users who are accessing the system without authorization and users who have access to the system but are exceeding their privileges [10].

Attackers are constantly searching the internet for servers that they can exploit for malicious activities. Major targets are the systems that have services such as Secure Shell (SSH) and Simple Mail Transfer Protocol (SMTP) enabled on them [11, 12]. Such attacks can be monitored and prevented using an Intrusion Detection System (IDS), which is like a bugler alarm for servers, configured to monitor hostile activities and intrusion attempts [13, 14, 9].

There are 4 internet facing servers owned by Tolerant Networks, which have widely used open-source IDS, Fail2Ban and DenyHosts, protecting them. These systems send out notification to a configured email address with the basic details of each attack. This data from the emails was collected in 2 phases.

In the first phase, an exploratory analysis of the available data was carried out to try to derive patterns and trends and gain in-depth insights about the attacks. Using the IP information from each of the attacks, further publicly available details about each of attacking IP address was gathered. This information was then used to calculate a malicious score and assigned for each of the attacks. Based on the score, a classification of how dodgy an attacking IP was assigned. Using this custom dataset, collected from live internet facing servers, a security dashboard was created to demonstrate the state of the system, in terms of intrusion attempts detected, the patterns in the attacks and flagged recurrent offenders. A couple of machine learning approaches were explored to predict certain parameters about the attacks. A time series forecasting model was created to ascertain the rate of attacks. A classification model was developed to predict how suspicious an attacking IP is.

In the second phase, another round of data was extracted and passed through the same pipeline for analysis as the first phase. This time around, the machine learning models created in phase 1 were tested with the new real world data.

1.1 Research Question and Aims

The emphasis of security is undeniable in today's world as discussed in the above Section 1. With data on intrusion attempts available from live internet facing servers owned by Tolerant Networks, it was decided to investigate these attacks to produce an exploratory analysis to help learn about the attacker trends and patterns. An attempt is also made to use this data in order to predict the attacks and how malicious they can be. This resulted in the following research question being formed:

How can existing data on intrusion attempts over Secure Shell (SSH), Simple Mail Transfer Protocol (SMTP) and other remote services be leveraged to enhance open source Intrusion Detection Systems (IDS) and overall security of internet facing servers?

The goals of this thesis are listed as follows:

1. Convert the unstructured existing data into structured format that can be analyzed
2. Extract useful information and details about the attackers
3. Persist data in a data store that can be easily accessed and used for analysis
4. Assign a weighted score to each attack based on the reputation of the IP in the public domain and the service being targeted
5. Leverage visualization techniques to analyze geo-spatial, temporal trends and understand attack patterns
6. Identify repeat offenders and the frequency of attacks
7. Produce a security dashboard using open-source tools which can help administrators improve overall security of the system
8. Use Machine Learning techniques to forecast and predict the badness of an attacker

1.2 Contribution

The contribution of this work to the field of computer science is listed as follows:

- A software pipeline for processing intrusion data. The program created accepts the raw data as input, parses the useful information, gathers additional important features about the attackers, calculates a badness score for each attack, classifies the detected intrusion attempt and exports the data in two popular formats
- The data is then persisted to an open-source easy-to-query data store running locally for ease of use in visualization and creating a dashboard
- A security dashboard is created using the data from the persistence using an open-source visualization tool which is compatible with the data store
- Supervised Learning techniques - Time-Series Forecasting and Classification algorithms were used with the collected data to generate predictions about the attacks

1.3 Structure of the Thesis

The remainder of this thesis is structured as follows: The motivation to pursue this research, background to the important concepts referred to in this thesis and with the literature review is in explained in chapter 2. The design of the software pipeline for processing intrusion attempts detected by IDS, algorithms used and evaluation criteria for the methods suggested are described in chapter 3. The architecture designed to implemented the methods along an elaborate explanation of the dataset, feature engineering and the technical details are provided in chapter 4. The detailed analysis and prediction results are discussed in chapter 5. Limitations, scope of future work and concluding remarks are made in chapter 6. Supporting materials such as details about the regex patterns used, phase wise splits of the data and additional prediction graphs are included in the appendix section of the dissertation.

Chapter 2

Background and Related Work

In the previous chapter 1 an overview of the research carried out through the course of this thesis was introduced along with a background to the importance of IDS, the research aims, contribution and the outline of the thesis. In this section, the motivation behind this work, background to the concepts referred to this dissertation and a comprehensive review of the existing work in this area is described in a concise and categorised manner.

2.1 Motivation

Internet facing servers frequently face intrusion attempts on their SSH and SMTP service [11, 12]. Widespread cyber-attacks make the need to gather as much information as possible about them a real demand in the global context [15]. Malicious attacks resulting in breaches are estimated to cause a monetary loss of $2 trillion by the end of 2019 [3, 6]. IDS helps to counteract these attacks, thereby making them an integral cog in the security wheel of a server [13, 9, 14]. It is therefore critical to ensure that the IDS is working efficiently on a given system. The goal is that by finding geo-spatial, temporal and repetitive patterns in the attacks and predicting the threat level for attacks, open-source IDS can be enhanced to be more robust. A stable and robust IDS can improve the security of systems and make them more reliable [16].

2.2 Literature Review

In this section, the literature review has been categorised for better readability and better understanding of the state of the art of work done in this area. It concludes by outlining the contribution of the research done in this thesis.

2.2.1 Early Research

The seminal paper in the field of intrusion detection systems was first introduced in 1980. In [17] the author describes four categories of attackers, namely:

- External Penetrator: a non authorised user accessing a system
- Masquerader: a user who has gained access to a system either as a legitimate user or an External Penetrator, pretends to be another user for illicit activities
- Misfeasor: a legitimate user misusing access for privileged information
- Clandestine: a user that operates with elevated access

The author then goes on to define that it could be possible to detect instances of intrusion by analysis of audit records to determine abnormal usage, frequency, volume of data referenced and patterns to the reference.

In [18] the author states that the above definition is the first literature to present the concept of intrusion detection for computer systems. The author surveys research carried out in the field of computer and network intrusion system until 1998, including features of intrusion detection systems and classification of the systems based on the taxonomy. The findings in this work were that the research interest should lie in understanding of the effectiveness of these systems and defending potential attacks against the IDS itself. This key point made 2 decades ago remains relevant even in the modern world.

Dorothy, in [19] introduced a real-time expert system capable of detecting intrusion attempts. The author describes profiling behavior using metrics and statistical models of identifying anomalous behaviour which is independent of the type of system, category of intrusion or the vulnerability defining a model for a general purpose framework for IDS.

2.2.2 Survey and Review

In 2013, a comprehensive review was carried out in [8]. HJ Liao et. al. believe that existing IDSs pose a threat to the various categories of intrusion and also to the computation power. The authors classify and compare a number of different intrusion detection approaches, methodologies and discuss about open source IDS Snort and ClamAV.A discussion of the pro's and con's of different approaches is highlighted and a well defined summary provided for researchers. The authors also highlight the importance of IDS in virtualized environments since they are extensively used in the cloud environments.

A systematic review of intrusion detection and prevention systems was carried out in [20]. The authors state that IDS existing at the time were inefficient in the cloud computing context. They investigate possible IDS solutions for cloud computing systems and provide four concepts of fuzzy theory, ontology, automated computing of self management and risk management to fulfill the security requirements in these systems.

Knowledge Discovery and Data Mining dataset, famously known as KDD famous benchmark used in intrusion detection research [21, 22]. An empirical analysis of the dataset attributes class wise was carried out in [21]. The dataset consists of 42 attributes divided into 4 classes, namely, basic, content, traffic and host. However, this dataset is 2 decades old, created in 1999. Therefore, it may be advised against using this data for benchmark analysis as the computer trends have changed massively and would not be reflected in the KDD dataset [23].

A study of intrusion attempts on 3 different types of honeypot systems - a small business connected to the internet, a residential connection with DSL connection and a university network was carried out in [24].The specification of each honeypot was different from each other. The system services were modified to ease the analysis of attacks. The systems were run in two batches of 5-6 each and data across the honeypots was aggregated using a centralized database. The key findings were that passwords of varying complexity are attempted by attackers and the attackers used distributed systems to launch coordinated attacks from the same class C network. The authors also find that 93% of the malicious IPs caught during the study were listed in the denyhosts central malicious IP list.

A study of free open source intrusion detection systems was carried out in [14].

The motive of the authors is to create awareness of the threats and available intrusion detection tools to counteract these threats for small businesses.

2.2.3 Attack Analysis

Visualization plays a key role in identifying patterns on attacks, such as time, source and type of attacks [25]. The authors in this research identify that country by country statistics and hourly change visualization make tracking of different sources and identification of source and relation of the intrusion difficult to identify. A visualization of the the threats using 2-d matrix representation of IP addresses is proposed. Authors are able to identify local proximity of IP addresses in worm propagation algorithms.

A cybersecurity dashboard, BubbleNet was designed in [26] to visualize exponentially growing data and help analysts to identify and summarise patterns in the data. MaxMind DB was used to geo-locate records. Bar chart and heatmap were used for temporal views. The evaluation of the dashboard was done using feedback from real users.

Honeypots are widely used webtraps to learn about attackers and their behaviour. [11] uses this approach operating Kippo SSH honeypot for a duration of 4 months. The authors note that attackers use existing tools and ready made dictionary for the attacks. Post collection of the data, the authors produce a number of visualisation such as "Top Usernames", "Top Passwords", "Top Credential combination", "Most probes per day", "Most probes per week", "Most probes per month ", "Top Sources". A limitation is this approach is that the data is collected for a short period and would make it difficult to estimate or use the patterns in making key security considerations.

In [27], it is stated that common tools can be used to protect individual systems in a network. The authors suggest a network-wide implementation strategy by maintaining a common database about attacks and preventing the wastage of resources and provide a preemptive defense mechanism by interpreting the events in the common database. The IDS tools used in this study are Fail2Ban and DenyHosts. 20% of the attacks were preemptively blocked using this mechanism. The attacks were observed on the Carnegie Mellon University network for a period of one year.

2.2.4 Machine Learning approaches to Intrusion Detection

[28] surveys the usage of deep learning approaches to phishing, malware and intrusion detection. Popular existing models are discussed and used to carve out a general framework that can be applied to cybersecurity systems. Mahdavifar et. al. summarize their findings by stating that that deep learning is not required for every domain and should only be used in complex non-linear hypotheses that include a large number of higher order polynomial terms and huge quantity of data. In other cases, simpler approaches would fulfill the requirements.

[7] highlights the importance of detecting intrusion attempts at an early stage. The authors propose a new approach, Neighborhood Outlier Factor (NOF) approach which relies on outlier detection to identify anomaly. The authors conclude that this approach performs better that 3 other approaches compared in this work. However, in this work, the approach was tested only on one dataset, KDD dataset.

In [23] the authors argue that intrusion detection at the network level provides a scalable approach to detecting attacks. Netflow Analysis was used to identify malicious activities. A machine learning approach was suggested to detect brute force attacks at the network level. The authors found that brute force netflows for failed login attempts were similar to failed login attempts of legitimate users, however, the number of attempts were a distinguishing factor. The machine learning model described in this work performs well and achieves 99% AUC, which on general reading appears too good to be true. This could be attributed to the fact that the data was collected in a custom manner for a period of eight days, which may not be sufficient to capture variability in attacks and at the same time the model can be overfitted to the sample data.

[29] proposes another Machine Learning approach to detect brute force attacks at the network level. The authors identify a down-side of host based intrusion detection systems to be that they are unable to detect attacks on distributed systems which are prevalent in the current world. The data used was collected for a period of 24 hours and aggregated using netflow. Four classification learners were chosen and the results were found to be overwhelming as in [23] due to reasons discussed above.

RASSH - reinforced adaptive honeypot was suggested in [15], that adapts based on the attacker's actions. Heliza is honeypot system similar to Kippo, that interacts with attackers using machine learning techniques. The RAASH system was built on top of

Heliza. Reinforcement learning using SARSA algorithm and Markov state model was used to build the proposed system. The system consisted of a set of 75 commands that could be run by attackers and could implement a set of 5 actions to get the most information about the attacker. This system could be used to get better insights about attacker trends and patterns.

Research in the field of Intrusion Detection Systems has been taking place since 1980 as discussed above. Various techniques, such as setting up of honeypot systems to learn about attack patterns, using visualization to get a better idea about ideas, using machine learning for intrusion detection have been used by researchers in the past. However, most of the data in the work discussed in this section was gathered for a short time-frame. In this thesis, the data used was gathered for a realistic time-frame from live servers facing intrusion attempts on a day-to-day basis. Visualization techniques were used to produce a dashboard which can help security administrators make better informed decision based on the scenario. The prediction module doesn't replace the IDS as in most of the work discussed in Section 2.2.4, but aims to use the IDS data to train the machine and provide actionable insights to improve the IDS system. The following sections in this chapter provide a background to the topics that will be referenced in the following chapters of this thesis.

2.3 Types of Intrusion Detection Systems

An IDS typically monitors and analyzes network or host related data in order to detect malicious activities. It inspects all incoming and outgoing activities and sends administrators and operators appropriate warnings for further action. Different categorizations of IDS can be based on the actions taken by them or based on the system they are intended to protect. The prominent broad categorizations relevant to this research is discussed below.

- Host IDS (HIDS): operates at the host level, providing intrusion detection with respect to a single host. This kind of a system monitors various log files for suspicious activity. It can also monitor configuration files for unauthorized changes. HIDS monitors the interaction of the host with other systems as well as the integrity of the host [14, 30, 9].

- Network IDS (NIDS): operates at the network level; setup at strategic points within the network such that inbound and outbound traffic from all devices in the network can be monitored. Instead of monitoring log and configuration files these systems monitor traffic streams and network data. Some intrusion methods are known for exploiting system vulnerabilities by sending malformed packets to a system. NIDS are well equipped to detect these attacks [14, 30, 9].

Further, IDS can also be categorized on whether the monitoring takes place as the event occurs (active) or after the event has occurred (passive).

- Active IDS: uses a pre-defined set of rules to automatically detect and block suspected attacks. This type of a system offers real-time protection [14].
- Passive IDS: monitors the defined log files and activities for suspected attacks and either reports it to the administrator or takes an action as per its defined setup [14].

2.4 Intrusion Detection Systems

In the previous section (Section 2.3) a brief introduction to the types of IDS was given. The data that is analyzed in this work was generated from two of the most popular open source Intrusion Detection Systems - DenyHosts and Fail2Ban [27]. In this section an overview of these two IDS is provided.

2.4.1 DenyHosts

DenyHosts is a cron driven script that is intended to thwart SSH server attacks. It parses through log files and and blocks the offender IPs through a file named $\backslash etc \backslash hosts.deny$. Through this block, follow-up login attempts from offending IPs are prevented.
Denyhosts can be used as a central repository in synchronization mode for proactively blocking known compromised IPs performing brute force SSH attacks. The synchronization mode enables sharing of data with respect to attacks with a centralized server and consumers that have this mode enabled can access this list to proactively block the known offenders [30, 31].

2.4.2 Fail2Ban

Fail2Ban is an automation tool to prevent brute force password guessing attacks. It can cover any service that uses password based authentication. Along with detection, it also offers some prevention features. When the tool finds a suspicious activity, it automatically updates the firewall rules to block the malicious behavior of the source IP. Along with this default action, it also offers arbitrary actions such as triggering notification mails/reports to the administrator. It works in the following ways [27]:

- periodically monitors the log files
- finds log entries that matches with the defined filter patterns which indicate an attack
- update the rules in the firewall table to block an IP for a specified time defined in the Fail2Ban configuration. After this ban time has elapsed, the IP is removed from the block list in the firewall table

The tool by default provides filters for common services such as APACHE-AUTH, SSH, FTP and Postfix. The prevention functionality is performed by updating the host's firewall tables. It can work with Netfilers, IPTables or hosts.deny table of TCP Wrapper. The filters and actions of Fail2Ban is referred to as jail. The typical configuration of Fail2Ban would contain [27]:

- path to the file where authentication information resides
- filter (regex) used to detect authentication failures
- threshold for login failures that indicates an attack
- follow-up action when an attack is detected
- time for which the attacking IP is banned

The threshold value for login failures is set such that genuine login failures are not treated as an attack, i.e to prevent false positives. The ban time is set to temporarily block an IP, while also granting it a second chance. When a log entry matches a certain defined rule, Fail2Ban extracts the required information and forwards it to the action module. The action module defines a set of commands that are executed for different states of the jail [30, 27, 14, 32].

2.5 Application Layer Protocols

In this section, the application layer protocols relevant to this project are discussed. Secure Shell (SSH) and Simple Mail Transfer Protocol (SMTP) are explained as these are among the common remote services running on internet facing servers. A brief explanation of these application layer protocols is necessary as they are subject to attacks by intruders.

2.5.1 Secure Shell (SSH)

SSH is a protocol used for secure access of remote services over an insecure network. It normally uses port 22 assigned by IANA, but it can be configured to use another port as per the needs of the user [33, 34].

2.5.2 Simple Mail Transfer Protocol (SMTP)

SMTP is the basic protocol used for reliable and efficient electronic email transfer. It normally uses the port 25 assigned by IANA [35, 34].

Postfix is an open-source Mail Transfer Agent (MTA) that is used to route and deliver email. It can be used a SMTP server or SMTP client [36, 37, 38]. To prevent unknown users from unauthenticated relaying of messages and spamming, postfix can be used with SASL (Simple Authentication and Security Layer) to authenticate clients. Dovecot, an open source Internet Mail Access Protocol (IMAP) server can be used to handle postfix-sasl authentication [37, 38].

2.5.3 Hypertext Transfer Protocol (HTTP)

HTTP is a stateless application level protocol that provides an uniform interface for interacting with a resource. A HTTP message is either a request or response. It enables client-server communication [39]. Port 80 is generally reserved for HTTP and 443 for HTTP with Transport Layer Security (TLS) [34].

Httpd is a standalone deamon process which is an Apache HTTP server program [40, 41]. apache-auth service is used to enable basic authentication and authorization services on a server [42].

2.6 Summary

Thus far, an introduction to the project has been provided along with a comprehensive literature survey explaining the related work done spanning from 1980 to the current year. An overview of Intrusion Detection Systems along with a brief explanation of the working of two prominent open-source IDS, Fail2Ban and DenyHosts, that are used through the course of this project was provided. Background on some of the common terminology used through the dissertation such as services monitored, VMs was also described. The next chapter will describe the methods used through the course of this project, at some points referencing to the background and literature discussed in this section and building on top of it.

Chapter 3

Method

In the previous chapter 2 an exhaustive survey of the background and related work was carried out. This chapter discusses in depth about the methods used to leverage existing data about intrusion attempts gathered using open-sourcing IDS to carry out exploratory and predictive analysis.

3.1 Design

Figure 3.1 demonstrates the high level design of the pipeline used to perform analysis on the intrusion attempts. The internet facing servers are subject to consistent attacks from a number of different attackers. There are a certain number of ethical researchers who perform ethical connection attempts which also get detected and blocked by the IDS. The IDS is unable to distinguish between ethical researchers and attackers and treats them alike as per the rules configured. The IDS have notifications enabled to send information about the intrusions detected and the corresponding action taken to the administrator's email account.

The data about the intrusion attempts is exported for analysis. Using this data, two forms of analysis is performed, exploratory analysis and predictive analysis. The exploratory analysis is carried out with the help of visualizations and dashboards to provide a holistic view about the security of the system to respective administrators. The motive of the exploratory analysis is to learn about the geo-spatial, temporal and repetitive patterns in the attacks, if they exist. The predictive analysis uses machine

learning techniques in an attempt to create a model capable of understanding underlying patterns in attacks and bring in a level of intelligence to preempt attacks. The motive of the predictive analysis is not to create a perfect model, but to understand the level of variability in the attacks and whether it is possible to gain an upper hand on the bad actors.

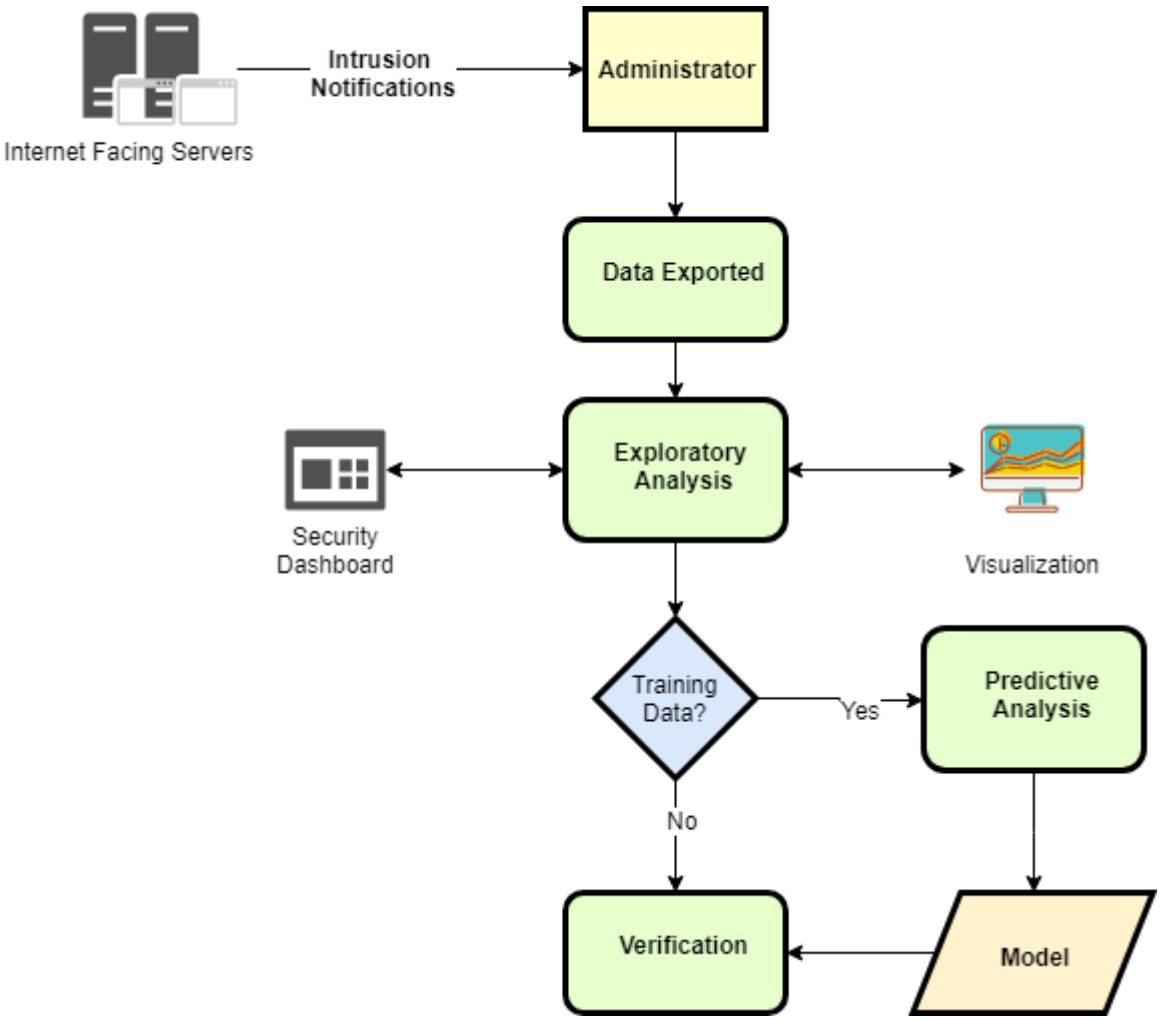


Figure 3.1: A high level view of the design of the system

3.2 Data Management

This section provides a detailed explanation of the data used to carry of the analysis through the course of this project. It explains the data collection and format, features present in the data and the security and privacy concerns around the data that is being used.

3.2.1 Data

There are a set of 4 internet facing Virtual Machines (VM) owned by Tolerant Networks which are frequently attacked on their SSH and SMTP ports [43]. The details of the Virtual Private Server (VPS), their IP, service running and frequently attacked services are shown in Table 3.1.

Table 3.1: Summary of the VPS running open-source IDS tools. The data has been collected from these hosts for analysis in this project

VPS	Domain	IP	Remote Services	Frequent Attacks
hoba	hoba.ie	92.51.243.15	Web, Mail, SSH	apache-auth, dovecot, postfix, sshd
tolerantnetworks	box.tolerantnetworks.ie vps.tolerantnetworks.ie my-own.net	185.24.234.234	Web, Mail, SSH	postfix, sshd
responsible	vps.responsible.ie	185.24.233.211	Web, Mail, SSH	postfix, sshd
jell	jell.ie	185.24.234.243	Web, Mail, SSH	postfix, sshd

These machines have two of the most popular open source IDS - Fail2Ban and Denyhosts installed [27, 32, 31, 30]. Data specific to the attacks detected and blocked by the IDS is available for the time span ranging from June 22, 2017 - August 7, 2019. The data is available in the form of emails which are triggered by the tools for the attacking IP addresses banned/blocked by them. Data used for this research is authoritative and reliable as it has been collected by tools configured by known administrators.

3.2.2 Data Collection

The emails pertaining to the attacks were exported in MBOX files from the administrator account using keywords and filters. Mbox(MailBOX) database format is used

to store and organize messages in the form of text files. It consists of a linear sequence of emails in a single file for each mailbox [44]. Apart from the content of the message, the file consists of meta data about each mail, such as the sender, recipient, date/time, message-id and mail hop details. This unstructured data had to be parsed and converted to a structured form for ease of analysis.

The data, available for the time span ranging from June 22, 2017 - August 7, 2019 was collected in two phases as shown in Table 3.2.

Table 3.2: Summary of the phases in data collection, the duration and purpose of the data

Phase	Duration	Purpose
Phase 1	June 22, 2017 - October 22, 2018	Exploratory Analysis Training the Machine Learning models
Phase 2	October 23, 2018 - August 7, 2019	Exploratory Analysis Verifying the Machine Learning models

3.2.3 Security and Privacy Considerations

Open Source software refers to software whose code is publicly available to view, modify and redistributed along with the original rights as described by the Open Source Initiative (OSI). Though the intention of open source software is good - anyone interested in the software may review and assess the quality apart from using it. Many a time problems maybe discovered and exploited by bad actors [45]. Though the general concerns with open source software is applicable to Fail2Ban and DenyHosts as well, the following discussion brings out some specific concerns with respect to the tools and the project in general.

Security Concerns

- Absence of meticulous evaluation of the software: Unlike proprietary software, the IDS tools discussed here, do not undergo any formal verification or evaluation. A vulnerability database like CVE must be regularly checked for any known vulnerabilities in these tools. The vulnerability score assigned by these database

give a generic threat level, based on the use case and acceptance of the risk, a decision continue to use the software can be made [46, 45]

- Spurious updates: It could be possible for an amature to distribute malware by adding malicious code into the system. IDS tools monitor critical sensitive information. Any version updates should be scrutinized before upgrade to a new version [45]

- Maintenance of open source software: Open Source software are mostly managed and maintained by individuals and/or organizations. Not all software receive financial aid, which can hinder the growth and maintenance of software. Lack of sponsorship would result in minimal to no maintenance of the software hampering timely fixes to known vulnerabilities. For instance Fail2Ban receives frequent updates, whereas the last version of DenyHosts was released in 2008 [47, 48, 45]

- Updates: Use of the latest stable version of the tools to prevent exploitation of known vulnerabilities and in older version while also ensuring continuous protection

- Documentation: Lack of proper documentation may hinder the use and make it difficult to obtain details of the behaviour of the system. Most of the open source tools lack support documentation and the issues with them would need to be resolved through the community, which doesn't guarantee a time-bound resolution

- Accountability: The IDS tools are critical to detect and alert about attacks (and prevent/subside using Fail2Ban). In case of most Open Source tools, there is a lack of accountability in case of an attack getting through or an genuine user getting locked out

- Dependent Libraries: Fail2Ban and DenyHosts are written in Python. The Python version used along with dependent libraries must be secure such that known vulnerabilities of the respective Python version or the libraries are not exploited

- Outdated Configurations: The configuration files of the IDS tools must be regularly updated as per the changing attack trends and patterns. Since both the

tools are Host Based IDS, any change in the logging format without appropriate updates to the IDS configurations can break the IDS on the system

Privacy Concerns

- The data with respect to the hosts and tools are sensitive and confidential to Tolerant Networks in this instance and must be handled with care to preserve privacy. In future, if data from any other source is used, it should be handled responsibly and comply with General Data Protection Regulations (GDPR) [4, 5]
- Synchronization feature of DenyHosts and any feature which sends critical information to a centralized repository must be used with care to prevent any sensitive data from leaking to the public domain

3.3 Algorithms

In this section, the algorithm developed and used to produce the contributions explained in Section 1.2 is discussed in detail. There will be two perspectives discussed here, first the custom algorithm developed and second the machine learning algorithms used for predictive analytics.

3.3.1 Algorithm Developed

Table 3.3 outlines the algorithm created to produce a pipeline that can consume data about intrusion attempts, parse the data, extract useful features and assign a malicious score to quantify the threat level from each intruder. The algorithm outputs data in a format that can be analyzed and it is also persisted in a an easy-to-query persistence store.

Table 3.3: Algorithm developed to create a pipeline for analyzing the intrusion attempts

Step 1:	Consume raw data
Step 2:	Parse data for useful information
Step 3:	Collect relevant data from input - time of attack, attacking IP, remote service attacked, IDS that detected the intrusion attempt
Step 4:	Use IP address to gather publicly available information about the intruder
Step 5:	Retrieve IP registry information
Step 6:	Reference publicly available blacklists to gather reputation details about attacker
Step 7:	Fuse information collected in Step 3 - 6 to calculate malicious score for intruder
Step 8:	Classify intruder based on score generated in Step 7
Step 9:	Create Attack and IP object based on the defined model
Step 10:	Export the data with specific structure to be used for analysis
Step 11:	Persist the data to data store
Step 12:	Use persisted data for visualization, analysis and prediction

3.3.2 Machine Learning Algorithms

The data available was for a period of 2 years as discussed in Section 3.2.2. The Phase 1 data was used for analysis and creation of a prediction model, while Phase 2 data was used for comparative analysis and verifying the prediction. In this section, a brief introduction to Machine Learning, the algorithms used and metrics used for evaluation are described.

Machine Learning was aptly defined by Tom Mitchell et. al. in 1997 as:

A computer program is said to learn from experience E with respect to some task T and some performance measure P if its performance on T, as measured by P, improves with experience E [49].

It provides an advantage over rule based approaches by creating mappings from features to generate a prediction. It is not a one stop shop for all problems, however, it is good to use when there is a problem with too many rules and large quantities of labelled data is readily available. The data for this research was structured and labels were assigned to be used with machine learning algorithms.

Machine Learning can be broadly categorised as Supervised and Unsupervised Learning. Supervised Learning requires labelled data and typical tasks include Regression and Classification. The model learns the behavior from the data and outputs a predicted target or label. Unsupervised Learning on the other hand, works with current data as opposed to predicting future data. Typical tasks include clustering [49, 50]. The reason behind the prediction generated by a machine learning model is not often apparent and the model may appear like a blackbox [51].

A key challenge in cyber attack monitoring systems is the prediction of attacks. If attacks can be forecasted, their damage can be minimized [25]. In this research Supervised Machine Learning approaches are used due to the data that is available. The two supervised learning approaches used are:

1. **Time-Series Forecasting**

   A forecast is an estimate or anticipation of events in the future based on current indications [52]. Time-Series forecasting is a key area of forecasting where past observations of the same variable are collected and analyzed to create a model that describes the base relationship [53]. Forecasts are made using data using one or more time-series. Time series is a sequence of observations made over time [54]. A Time Series forecasting problem can be regarded as a regression problem.

   ARIMA (Auto Regressive Integrated Moving Average) is one of the most popular and widely used time-series model. The major limitation is the pre-assumed linear form of the model, meaning that future values are constrained to be linear functions of past observations [53]. fbProphet, is an open-source time-series forecasting model that takes into account seasonality and trend that is observed in the time-series data [55]. fbProphet internally runs based on ARIMA and Expo-

nential Smoothing techniques and proposes a semi-automated analyst in the loop approach shown in Figure 3.2. This algorithm was found to have substantially lower error when compared to other forecasting methods which made it a suitable choice to use for forecasting temporal attack patterns (rate of attacks) [55, 1].

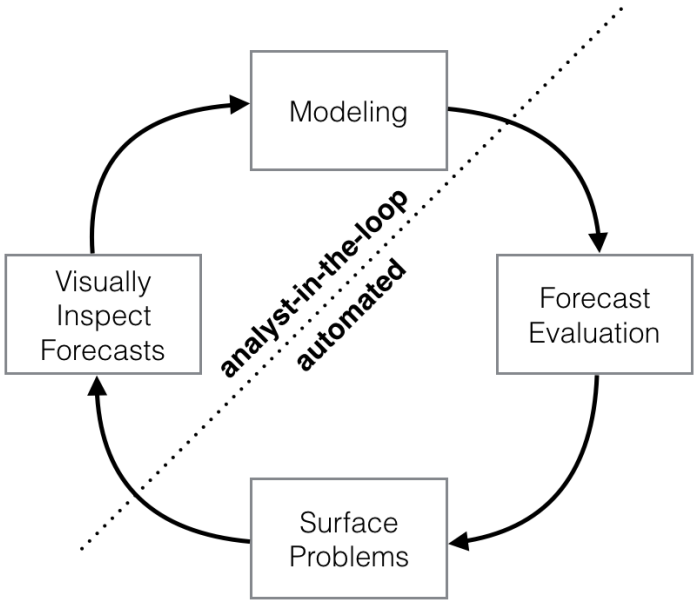


Figure 3.2: Semi automated model keeping analysts in the loop to alter model without background of underlying statistics [1]

2. **Threat Level Classification** The goal of a classification algorithm is to build a model of the class labels in terms of the independent variable or predictor features [56]. A common example of classification problems would be whether or not an email is spam. The classification problem maybe binary or multi-class. In this research, the classification problem is to predict how malicious an IP is. The classes are created based on malicious score that is computed for each attack. The implementation of the score generation is explained in Section 4.3.5.

   Tree boosting is a technique which is known to produce good outcomes for classification machine learning problems [57]. eXtreme Gradient Boosting (XGBoost) is an implementation of gradient boosted decision trees, designed for improving computation speed and model performance. It is widely used and known to produce state-of-the-art results for many problems [58, 59, 60]. Figure 3.3 shows the

high-level flow of an XGBoost Classification algorithm. XGBoost has also been used for intrusion detection and found to produce faster and scalable solutions with high accuracy [59]. XGBoost classifier is the algorithm that is used in this project for identifying the threat from an intruding IP.

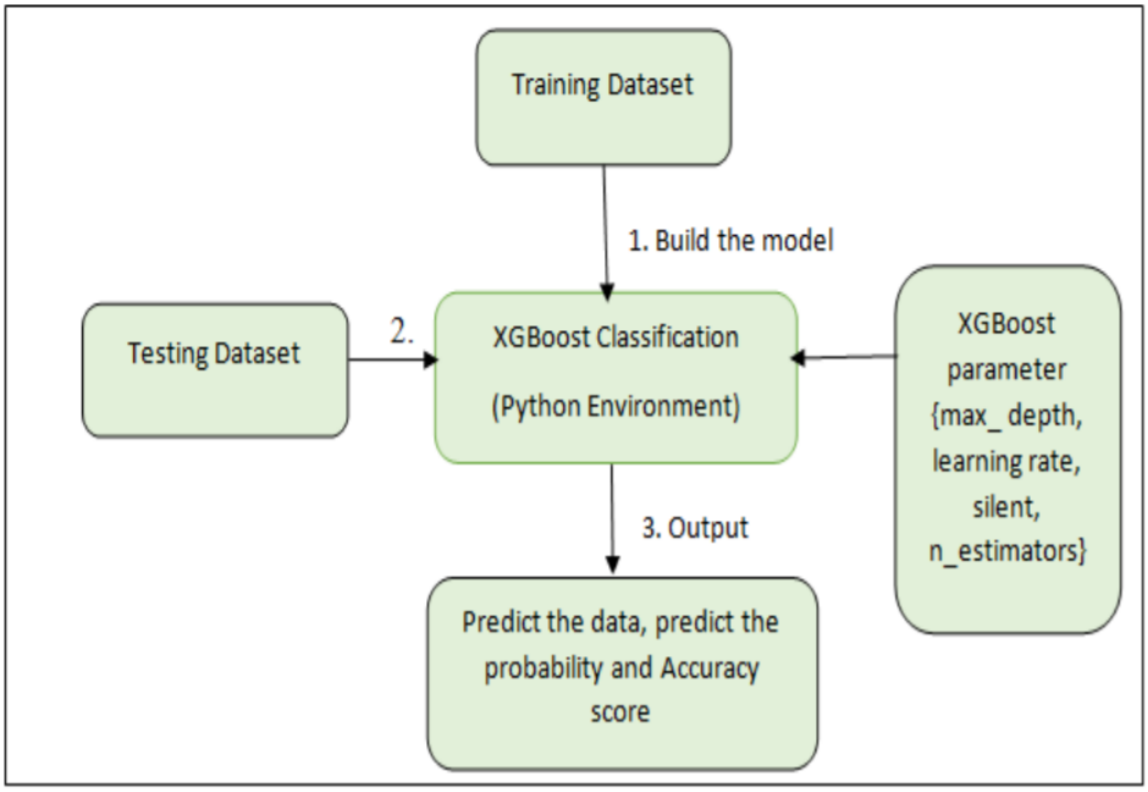


Figure 3.3: A summary view of the high-level working of an XGBoost Classification algorithm

3.4 Evaluation Methods

The metrics chosen to evaluate the Time-Series forecasting model are the Root Mean Square Error (RMSE) and Coefficient of Determination (R2). These are widely used regression evaluation metrics which give a measure of the error (RMSE) and performance (R2) of a model. Ideal RMSE value should be close to 0 while ideal R2 value should be near to 1. These metrics have also been used to measure performance in Time-Series forecasting modules [1, 53, 55].

The metrics chosen to evaluate the XGBoost model are Accuracy and Multi-Class Error Rate (Merror). A confusion matrix is produced for the classification demonstrating the actual values against predicted values. Using the confusion matrix or standard libraries, Accuracy and Merror can be computed. Accuracy represents the fraction of samples predicted correctly and the value should be close to 100%. Merror represents the fraction of wrong cases predicted by the model and its value should be close to 0.

The implementation details and architecture details will be discussed in the following chapter 4 which will build on top of the methods discussed in this section.

Chapter 4

Implementation

The previous chapter 3 described the methods that were designed/chosen for achieving the research aims described in chapter 1. A brief description of the data source and its collection process was also explained. In this chapter, the architecture that implements the methods, in-depth description of the implementation choices and the tools used are discussed.

4.1 Architecture

Figure 3.1 showed the high level view of the design of the pipeline to get meaningful insights out of the intrusion attempts detected. In this section the detailed architecture implemented in this research work is described.

The overall architecture of the system can be seen in Figure 4.1.

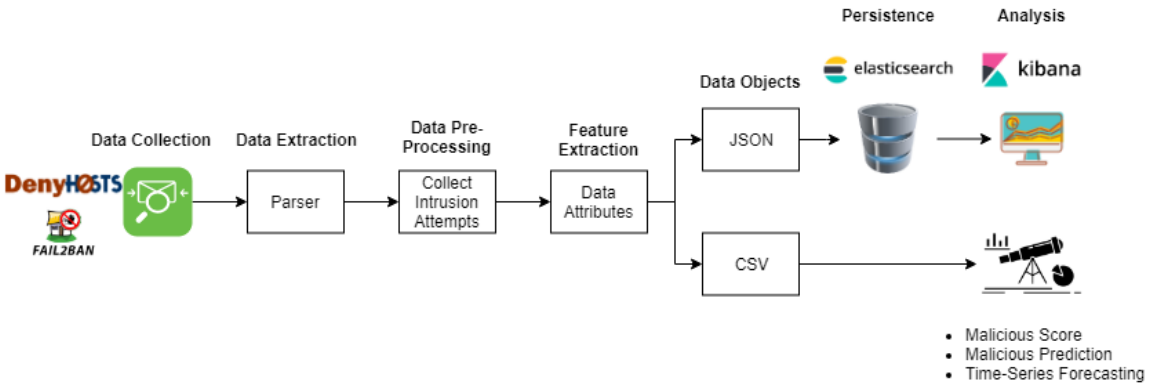


Figure 4.1: A detailed view of the architecture of the system with the phase of the pipeline at each hop

Data Collection

The IDS Fail2Ban and DenyHosts are setup on the VPS as described in Section 3.2. The IDS are configured with default settings except on hoba for postfix-sasl service, where the number of attempts before an IP gets banned for a predefined time is 2. A sample of the configuration used on the server is shown in Figure 4.2.

```
[postfix-sasl]
syslog_mail_warn = /var/log/mail.err
logpath = /var/log/mail.log
enabled = true
failregex = (?i): warning: [-._\w]+\[<HOST>\]: SASL
(?:LOGIN|PLAIN|(?:CRAM|DIGEST)-MD5) authentication failed(:[A-Za-z0-9+/
]*)?$
maxretry = 2
findtime = 36000
bantime = 360000
```

Figure 4.2: An example of the fail2ban configuration for postfix-sasl service

The IDS triggers a notification to the administrator account defined in the configuration. The data from all the email notifications of all the VMs was exported in MBOX format [44].

Data Extraction

The mbox file was given as raw data to the parser module of the program. The parser was responsible for reading through each exported MBOX files and parsing

through each of the IDS notification received.

Data Pre-Processing

Each email in the MBOX file was scanned to filter the relevant email's and use regex to extract the useful data. A few of the important fields that provide key indicators about an attack, such as IP address, service attacked were obtained in this step. These fields are then used in the feature extraction phase to learn about the intruder. An elaborate description of the extraction process is given in Section 4.2.1.

Feature Extraction

The IP address obtained in the previous phase was used to extract information about the attacker. The information included looking up publicly available registration details about an intruder, checking it's reputation in various blacklists and using the available information to compute a malicious score and assigning a threat class to it. Detailed information about the feature extraction process is discussed in Section 4.3.

Data Objects

Once the required information about each attack and intruder is gathered and assigned to a pre-defined data model, the data is exported in 2 different formats. Individual attack and intruder information is saved in JavaScript Object Notation(JSON) which makes it simple to be persisted for analysis and visualization. The attack details were also saved as Comma Separated Values(CSV) which would simplify the process of using it for predictive analysis.

Persistence

ElasticSearch is the open-source used for storing, retrieving and managing document-oriented and semi-structured data [61]. Every feature is exposed as an Application Programming Interface (API) that makes it easy to use [62]. It has its own Domain Specific Language (DSL) to query JSON data. It is a scalable database by allowing users to add nodes. ElasticSearch also provide easy integration with visualization and analytics tools like Kibana [63, 64, 62]. ElasticSearch was deployed as a Docker Container [1] which made its deployment easy to manage.

Analysis

There are 2 parts to the analysis phase. First is the exploratory analysis. Kibana,

[1]Docker is a set of coupled software-as-a-service and platform-as-a-service products that use operating-system-level virtualization to develop and deliver software in packages called containers. - Wikipedia

an open-source visualization plugin for ElasticSearch was used to provide visualization capabilities on top of the indexed content in ElasticSearch. Kibana was also deployed as a Docker Container. Second, the predictive analytics were performed using the algorithms explained in Section 3.3.2.

The programming language used for the implementation of the methods and design was Python (v3.6). The persistence was ElasticSearch Database and Visualization tool used on top of ElasticSearch was Kibana. A summary of the tools and respective libraries used through the course of this project is shown in Section 4.5

4.2 Dataset

The total data points collected during the entire time frame of June 22, 2017 - August 7, 2019 are 21190 intrusion attempts from 6802 intrusion IP address across the 4 VMs described in Section 3.2. The split of the number of data points in each phase is shown in Table 4.1. The percentage split of the data per phase is shown in Figure 4.3.

Table 4.1: Phase wise number of records collected for analysis

Phase	**Duration**	**Data Points**
Phase 1	June 22, 2017 - October 22, 2018	9,663
Phase 2	October 23, 2018 - August 7, 2019	11,526
Total	**21189**	

Figure 4.3: Percentage split of the dataset per phase

4.2.1 Data Pre-Processing

The extracted notification emails apart from containing information about attacks also had other mails containing similar keywords, which were broadly classified as "service notification". The IDS tools were also configured to send out notifications during a modification in the state of the IDS (service restart). These notifications were not used in the scope of this project, however, they needed to be identified and ignored during the pre-processing.
The mbox files were read, parsed and converted to a objects as per the data model. Duplicate email were checked and removed using Message-Id which is the unique ID for every message. The format of mail's from the two IDS, Fail2Ban and DenyHosts varied largely and parsing logic was developed to effectively segregate the notifications. As per the configuration, Fail2Ban notifications are generated for each attack and have details specific to it. Details include the service being attacked, the attacking IP and the target host specified in a concise manner in the subject of the email. A sample is shown below:

$$[Fail2Ban] \quad ServiceName: \quad banned \quad AttackingIP \quad from \quad TargetHost$$

The content of the Fail2Ban notification had details about the "number of attempts" the particular intruder made before getting banned. The message also contained some details about the intruder such as "Orgname" - defines the name of the organization that has registered the IP address, "Inetnum" - the IP network to which the IP belongs and "Netname" - specifies the name of a range of IP address [65]. The above mentioned fields were extracted using pattern matching which are shown in Appendix .1.

DenyHosts on the other hand generates a report of the IPs blocked and added to the "hosts.deny" list and triggers a notification. The subject of these email's is consistent ("DenyHosts report") and the attacking IP addresses is available in the body of the email. The regex used for extracting the IP addresses from the content is shown in Appendix .1.

Each Fail2Ban mail contains at most details about one attacking IP, however, a DenyHosts mail could contain about more than one attacking IP.

The MBOX files were parsed to extract the basic information, namely From, To, Subject, Date/TimeStamp and Message-Id from each mail. The subject fields of each mail were matched with keywords to segregate the mail's based on IDS tool. This

step was essential due to difference in the structure of the notification and information conveyed.

Based on the subject, the following details were extracted:

1. Fail2Ban notifications - regular expressions were applied to the subject for recognizing the service attacked, the attacking IP, number of attempts, org-name, netname, intetnum and targeted host

2. DenyHosts notifications - regular expression was applied to body of the email to extract attacking IP addresses(s). No other useful data was available in these notifications. The service was only sshd and number of attempts were the default value 5.

If a DenyHosts notification contained more than one IP, separate entries were created for each attacking IP, with the same message details for each entry. This was done to ensure the structured data had a unique data point for each intruding IP.

The data/timestamp field was available in the following format from mbox file:

$$Day, \quad dd \quad mmm \quad yyyy \quad hh:mm:ss \quad Timezone$$

This was converted to the form:

$$yyyy - mm - dd \quad hh:mm:ss$$

Table 4.2: The basic dataset attributes by parsing the raw data and applying regex to extract meaningful data

Attributes	**Datatype**
Intrusion IP	Categorical
Target Date/Time	Timestamp
MessageId	Categorical
Service	Categorical
BlockedByIDS	Categorical
TargetIP	Categorical
BlockedAfterAttempts	Numerical

It is important to note that the data extracted for the second phase was extracted for the entire duration of 26 months. The phases had to be segregated for noticing the behavior and evolution of attack patterns.

4.3 Feature Engineering

At the end of the section 4.2.1, the basic information was available for each attack on the hosts as summarized in Table 4.2. To enable exploratory analysis of the attacks, additional information was retrieved about each attacking IP as discussed in the sub-sections below.

4.3.1 Geolocation

The geolocation of the source of the attacks would help in identifying the concentrated zones around the globe from which intrusion attempts can originate and also provide some useful details about the intruders. Contrary to the general perception, geolocation databases are less reliable that they claim to be [66]. In general, the geolocation information of an IP address can be retrieved using WHOIS - a system that provides a directory of contact information for users of the internet. The Registration Data Access Protocol (RDAP) is the successor to WHOIS. It uses ReSTful interfaces over HTTP

to query and provide response in the form of JSON objects [67]. RDAP queries were made about the attacking IPs to obtain the geographic information about them. The performance of retrieval using RDAP was poor since it's services has a throttle rate and the program would have to wait for a cool off period before trying again. To overcome the performance issues, geolocation services provided by MaxMind were used [68]. The open-source geolocation database provided by MaxMind can be dowloaded locally and referenced for lookup [69]. This improved the performance of the geolocation details retrieval by nearly 80%.

The following details were retrieved using the MaxMind Geolocation database:

1. **Autonomous System Number (ASN):** A unique number that identifies each network on the internet
2. **Autonomous System Organization:** The organization with whom the ASN is associated
3. **Continent:** The continent in which the IP is located
4. **Country:** The country in which the IP is located
5. **City:** The city in which the IP is located
6. **Country Time Zone:** The time zone in the source country
7. **Location:** The Latitude and Longitude of the Intrusion IP available in the geolocation database

It should be noted that while MaxMind database is a reliable geolocation database for most of the details extracted, city details may not be accurate [66].

4.3.2 Time at the Country of the Intrusion IP - Source Time

The date/time of the attack was available for the target timezone as discussed in Section 4.2.1. It was hypothesised that a new field obtained by converting the timestamp of the attack to the respective source country's timezone could provide useful insights and aid in the analysis. Therefore, using the time-zone information retrieved for the intrusion IP from the geolocation database, the target time was converted to the source time zone and saved as a new field.

4.3.3 SSH Blacklist

A SSH blacklist is a list of known bad IP addresses that have made attempts to intrude a system and are regarded as untrustworthy.They should be excluded or avoided. A lookup for each unique attacking IP was made against a publicly available blacklist - Nothink blacklist, which is a widely referenced SSH blacklist to check whether the attacker is known to previously attempt SSH intrusion into other systems around the world [70, 71, 72]. A new feature "Blacklist" was created and updated with a boolean value - TRUE indicating presence in the blacklist, and FALSE indicating otherwise.

4.3.4 Email Spam Blacklist

Two widely referenced spam blacklists were referenced for each unique intrusion IP.

1. **Spamhaus Blacklist:** A widely referenced international blacklist to track email spammers and spam-related activities [73]. SpamHaus is estimated to protect 1.9 billion user accounts and is a popularly referenced blacklist [74, 75, 76]. It provides a Domain Name System-based Blackhole List (DNSBL). Sample query to the SpamHaus DNSBL is shown below:

   $$REVERSED - IP.zen.spamhaus.org$$

   Example: To check for the IP 127.0.0.1

   $$1.0.0.127.zen.spamhaus.org$$

   Spamhaus has 3 zones of blacklist [73]:

   - **Spamhaus Block List (SBL):** Consists of IP addresses belonging to verified spam services and known spammers. If an IP is present in this list, the response to the query is in the range 127.0.0.2-3,8-9
   - **Exploits Block List (XBL):** Consists of IP addresses of illegal third party exploits and trojan exploits. If an IP is present in this list, the response to the query is in the range 127.0.0.4-7

- **Policy Block List (PBL):** Consists of IP addresses that should not be delivering unauthenticated SMTP mail. If an IP is present in this list, the response to the query is in the range 127.0.0.10-11

Based on the answer to the Spamhaus DNS query, a new feature, "SpamHaus-Blacklist" was created. The respective zone name was stored if a response was received, else FALSE was assigned to indicate that the IP address is not in any Spamhaus blocklist.

2. **Barracuda Reputation Block List(BRBL):** A common email spam blacklist used to obtain the reputation of an IP [77]. The BRBL uses the standard DNSBL implementation to provide details about the queried IP. The primary residents in the BRBL are IP address observed to send spam or house spammers. A sample query format to the BRBL is shown below:

$$REVERSED - IP.b.barracudacentral.org$$

Example: To check for the IP 127.0.0.1

$$1.0.0.127.b.barracudacentral.org$$

If the queried IP has poor reputation the response to the query would be the IP itself. If the source IP does not have poor reputation, the response would be NX Domain, indicating domain not found.

Based on the answer to the DNS query, a new boolean feature, "Barracudda" was created. TRUE indicated that the IP had poor reputation and FALSE indicated otherwise.

4.3.5 Malicious Score and Threat Level Assignment

A malicious score calculation was devised based on the reputation of an intruder as found in Section 4.3.3 and Section 4.3.4. The service attacked was also taken into account and a different weight was assigned based on the service attacked and blacklist value.

The 3 blacklist values for Barracudda, NothinkBlacklist and SpamhausBlacklist were assigned an initial equal weight of 1. The Table 4.3 shows the weighted score calculation measure used for finding the malicious score. This value was assigned to new feature "MaliciousScore". The overall range of the score varied from 0-4.

Table 4.3: Malicious score calculation weighted based on the service being attacked. Higher weight given to SSH blacklist when sshd is attacked and higher weight given to spam blacklist when postfix is attacked

Service	NothinkBlacklist	Barracudda	SpamhausBlacklist	Minimum Possible Score	Minimum Possible Score
sshd	2	1	1	0	4
postfix	1	1.5	1.5	0	4
other	1	1	1	0	3

Once the malicious score was generated, a threat level class was assigned for each intrusion attempt. 3 classes were defined based on the range of "MalicousScore" as shown in Table 4.4. A new feature "DodgyClass" was created to represent the badness of an attacker.

Table 4.4: Threat Level assigned to each intrusion based on the malicious score

Class	Range of Malicious Score	Threat Level
Ok	0	Low
Dodgy	1-2	Medium
Very Dodgy	3-4	High

4.3.6 Other fields

A custom universally unique identifier (**UUID**) was generated for each attack record to have a unique id associated with each attack.
Phase was assigned to each data point based on when it was extracted as described in Table 3.2.
The calendar **Week** was extracted from the date to see how persistent an intruder can be.

At the end of the pre-processing, the dataset comprised on 29 features and 21189 data points across both phases. Some of the important attributes are shown in Table 4.5:

Table 4.5: Important attributes in the dataset - an overview

Attributes	**Datatype**
Target Date/Time	Timestamp
Source Date/Time	Timestamp
Week	Numerical
AsnNumber	Numerical
AsnOrganization	Categorical
Service	Categorical
Attacking IP	Categorical
Target IP	Categorical
Continent	Categorical
Country	Categorical
City	Categorical
BlockedByIDS	Categorical
BlockedAfterAttempts	Numerical
Barracudda	Boolean
NothinkBlacklist	Boolean
SpamhausBlacklist	Categorical
Latitude	Numerical
Longitude	Numerical
Phase	Categorical
MaliciousScore	Numerical
DodgyClass	Categorical

4.4 Machine Learning: Feature Selection and Model Parameters

The machine learning algorithms that are to be implemented for this work were discussed in Section 2.2.4. The features selected for the time-series forecasting and classification varied largely. The implementation details along with hyper parameters chosen for each of the algorithms is discussed below. The implementation of both the machine learning algorithms was carried out using Python programming language. A summary of the programming language, tools and libraries used is provided in Section 4.5.

4.4.1 Time Series Forecasting

The implementation of the time-series forecasting algorithm was carried out using the fbProphet python library provided by its developers [78, 79]. In contrast to other machine learning algorithms, the variables required for time-series forecasting are one independent variable and one dependent variable.

- The *independent variable* was the "Date" of the attack
- The *dependent variable* was the "Number of Attacks" on the respective date

The intrusion attempts had to be aggregated on a day level and used for this implementation. Post aggregation, it was found that Phase 1 data covered 488 days and Phase 2 data covered 292 days.

The **hyperparameters** required for the forecasting algorithm were:

- **periods:** indicating the duration for which the target/dependent variable is predicted. The forecast in this research was made for 4 months (123 days). This parameter value was set to 123
- **freq:** the frequency of the prediction in terms or daily, monthly or yearly. This parameter was set to "D" indicating day-wise forecast

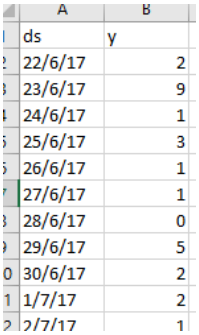
Since the data was from 4 different hosts and different geographic location, they may be subject to different rate and attack patterns. If the entire data was used, the intricate insights from the prediction could get lost. Therefore, separate forecasts were created for:

- each of the target hosts described in Section 3.2
- each continent from which an intrusion attempt is observed

It is important to note that the data could be sparse for some of the splits explained above that could adversely affect the algorithm performance. A workaround added was to include the missing dates with no intrusion attempts detected and set the number of attacks to 0.

The Phase 1 data was used to train the model and Phase 2 data was used to validate the model. Using the training data, a prediction for the following 123 days post October 22, 2018 was made. However, validation of the entire duration post that data may not give a good measure of the model. Therefore, apart from validating for the 4 months post October 22, 2018, the prediction was validated for the next 2 weeks after this date as well.

A snapshot of the data used for training the fbProphet model is shown in Figure 4.4.



	A	B
1	ds	y
2	22/6/17	2
3	23/6/17	9
4	24/6/17	1
5	25/6/17	3
6	26/6/17	1
7	27/6/17	1
8	28/6/17	0
9	29/6/17	5
10	30/6/17	2
11	1/7/17	2
12	2/7/17	1

Figure 4.4: Snapshot of the data aggregated for time-series forecasting. ds represents the data and y represents the target variable, the number of attacks on a given date

Dealing with Outliers

Outliers are data points with values that are abnormal, i.e. vastly different from the from the majority of the values. Outliers can affect a machine learning model adversely. For example, when plotting the rate of attacks for the VPS hoba on the Phase 1(train) data, it was observed there was one outlier with value greater than 800 as shown in Figure 4.5. This could be due to a service restart on the host which could have triggered duplicate notifications.

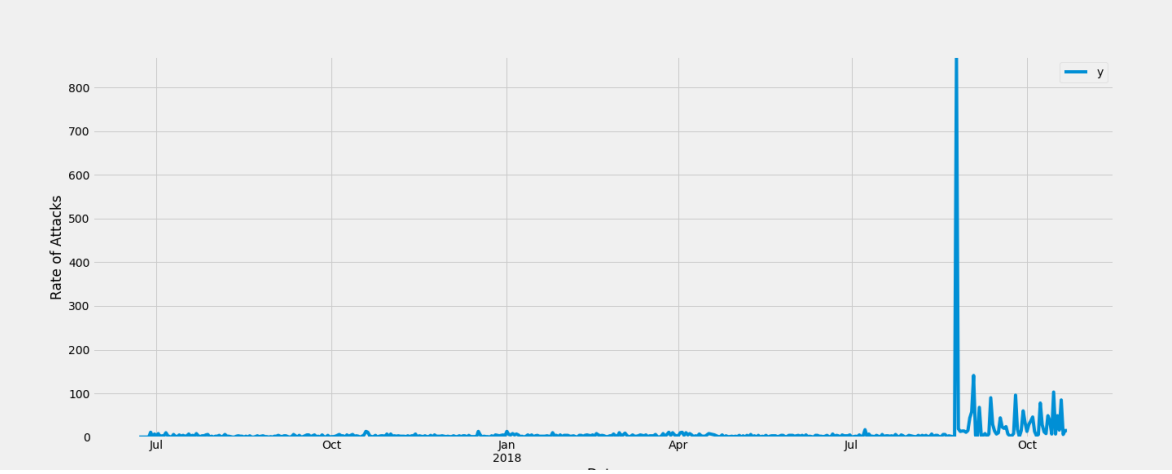


Figure 4.5: Rate of attacks observed on hoba during Phase 1

In this research, all outliers observed with the help of visualizations while preparing the data for time-series forecasting, were removed from the dataset.

4.4.2 Threat Level Classification

The XGBoost algorithm to classify the threat levels categorized in Section 4.3.5 was implemented using the xgboost python library [80]. The features were to be selected from the important attributes listed in Table 4.5. A correlation heatmap of the numerical variable was generated and is shown in Figure 4.6 to aid in the feature selection.

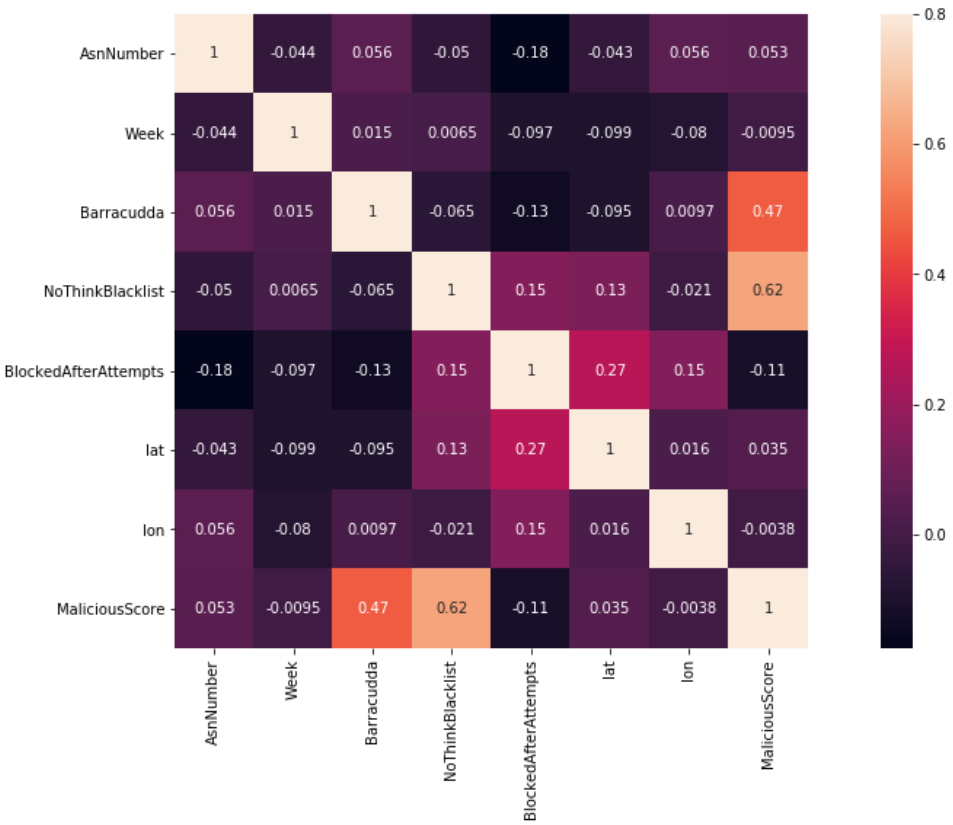


Figure 4.6: Correlation Heatmap of the numerical features in the dataset. Values closer to 1 or -1 indicate high correlation.

The important points to note with respect to the features are as follows:

- the *target variable* for this algorithm was the "DodgyClass". Distribution of the classes is shown in Figure 4.7

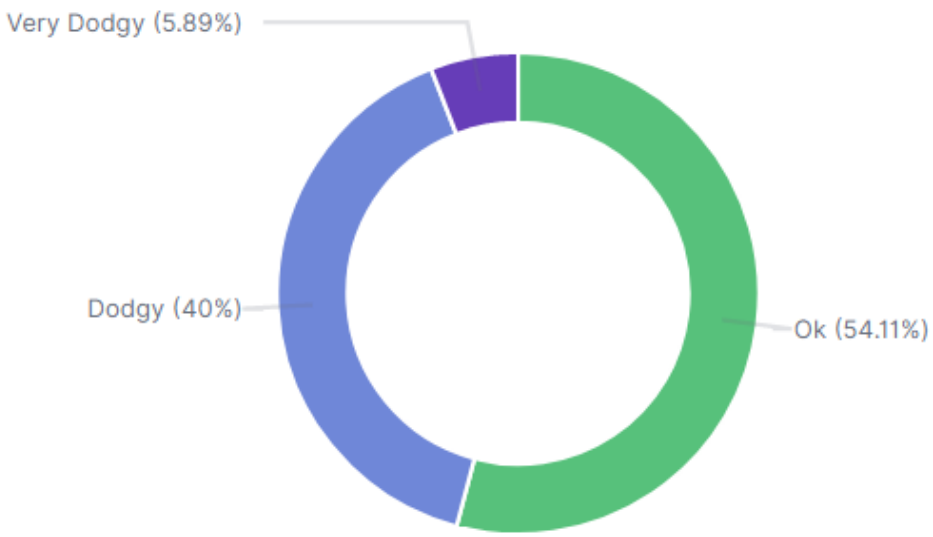


Figure 4.7: The distribution of the classes across the 9663 data points in Phase1 dataset

- the "City" details from previous research are found to be inaccurate, so this attribute was removed from the features to be used for prediction [66]
- the "SourceTime" was generated by converting the "TargetTime". Both these variables are highly correlated, hence SourceTime can be omitted
- the "Phase" variable is irrelevant to the prediction since data from Phase1 1 was used for training
- the "AsnNumber" and "AsnOrganization" fields are correlated to each other, therefore "AsnOrganization" field was dropped
- the blacklists are highly correlated to MaliciousScore as shown in Figure 4.6. This is expected since the values of the blacklist are used to compute the MaliciousScore. Also, the MaliciousScore is used to define threat levels of "DodgyClass". Therefore, the features "Barracudda","NothinkBlacklist","SpamHausBlacklist" and "MaliciousScore" were dropped from the list of features

The independent variables selected for predicting the threat level are shown in Table 4.6

Table 4.6: Features selected for the classification of threat level

Attributes	**Datatype**
Target Date/Time	Timestamp
Week	Numerical
AsnNumber	Numerical
Service	Categorical
Attacking IP	Categorical
Target IP	Categorical
Country	Categorical
BlockedByIDS	Categorical
BlockedAfterAttempts	Numerical
Latitude	Numerical
Longitude	Numerical

The features selected for any machine learning task may comprise of numerical or categorical values. The computer systems are able to understand and deal with numbers only. Therefore it is essential to convert the categorical features to numbers. The categorical features selected for building the XGBoost model in this work were converted to numerical values using a technique known as one-hot encoding.

The Phase 1 data was used to train and test the model. The train and test data split of the data was in the ratio of 90:10. Phase2 data was used to validate the model. 10-fold cross validation technique was used to tune the n_estimator (number of decision tree classifiers used) hyperparameter of the XGBoost algorithm. The cross validation results are shown in Figure 4.8. The merror value for the test data reduces negligibly after 17, therefore this was chosen as the value for n_estimators.

```
[0]     train-merror:0.167042+0.00975411        test-merror:0.192117+0.012419
[1]     train-merror:0.14854+0.00637154 test-merror:0.177187+0.0155106
[2]     train-merror:0.14384+0.00879567 test-merror:0.17387+0.0180521
[3]     train-merror:0.14043+0.0115979  test-merror:0.172833+0.0188894
[4]     train-merror:0.137308+0.0115383 test-merror:0.16765+0.0190775
[5]     train-merror:0.135557+0.0104346 test-merror:0.168687+0.0164768
[6]     train-merror:0.13369+0.00932357 test-merror:0.166302+0.0161459
[7]     train-merror:0.131145+0.00901794        test-merror:0.165991+0.016327
[8]     train-merror:0.127861+0.00913891        test-merror:0.164539+0.0160586
[9]     train-merror:0.12474+0.00830873 test-merror:0.161221+0.0171759
[10]    train-merror:0.121963+0.00719572        test-merror:0.15977+0.0152724
[11]    train-merror:0.118542+0.00593481        test-merror:0.156971+0.0147488
[12]    train-merror:0.117505+0.00574718        test-merror:0.155623+0.0157078
[13]    train-merror:0.116549+0.00608046        test-merror:0.154587+0.0154709
[14]    train-merror:0.116099+0.00701418        test-merror:0.152927+0.0154995
[15]    train-merror:0.114901+0.00643612        test-merror:0.15303+0.0162889
[16]    train-merror:0.112932+0.00617352        test-merror:0.149505+0.0156384
[17]    train-merror:0.111906+0.0064175 test-merror:0.14992+0.015979
[18]    train-merror:0.112148+0.00600784        test-merror:0.14992+0.0155248
```

Figure 4.8: 10-fold cross validation results for selecting n_estimator parameter of XGBoost classification algorithm

One-hot encoding and splitting of the Phase 1 data into train-test parts was done with the help of sklearn python library [81].

4.5 Programming Language, Libraries and Tools

The summary of the prominent programming language, libraries and tools used in the implementation of the proposed methods (chapter 3) and architecture(Section 4.1) is shown below in Table 4.7.

Table 4.7: A summary of the tools, programming language used along with their purpose in the implementation phase

Architecture Phase	Programming Language/Tool	Library	Purpose of Library/Tool
Data Collection	Shell (BASH)	NA	NA
Data Extraction	Python (v3.6)	glob	retrieve list of files in a directory
		mailbox	parse messages in a MBOX file
Data Pre-Processing		datetime	managing and converting datetime formats
		re	pattern matching - regex
Feature Extraction		uuid	generate UUID for each attack
		urllib.requeset	retrieve Nothink SSH blacklist
		dns.resolver	query Barracudda and Spamhaus DNSBL
		geoip2.database	lookup IP address in MaxMind Geolocation database
Data Objects		csv	export data as CSV
		json	export data as JSON
Persistence		elasticsearch	persist data to ElasticSearch database
Analytics		pandas	handling data in a dataframe
		numpy	handling data in an array
		fbprohpet	create fbProphet model for time-series forecasting
		xgboost	create XGBoost classification model
		sklearn	evaluation metrics for regression and classificaion
		matplotlib	generating visualizations programmatically
	Kibana	NA	searching and visualizing data indexed in ElasticSearch
	Microsoft Excel	NA	viewing data, creating pivots and basic graphs

4.6 Summary

In this chapter, the detailed implementation of the methods introduced in chapter 3 was described with appropriate examples. The data collected was passed through a pipeline, developed through the course of this project, and the processed information was output from two perspectives. First, the data was exported as "Attacks" which resulted in 21189 data points. Second, from the perspective of unique intruding "IP" address and comprised of 6802 data points. A sample "Attack" json object is shown in Figure 4.9.

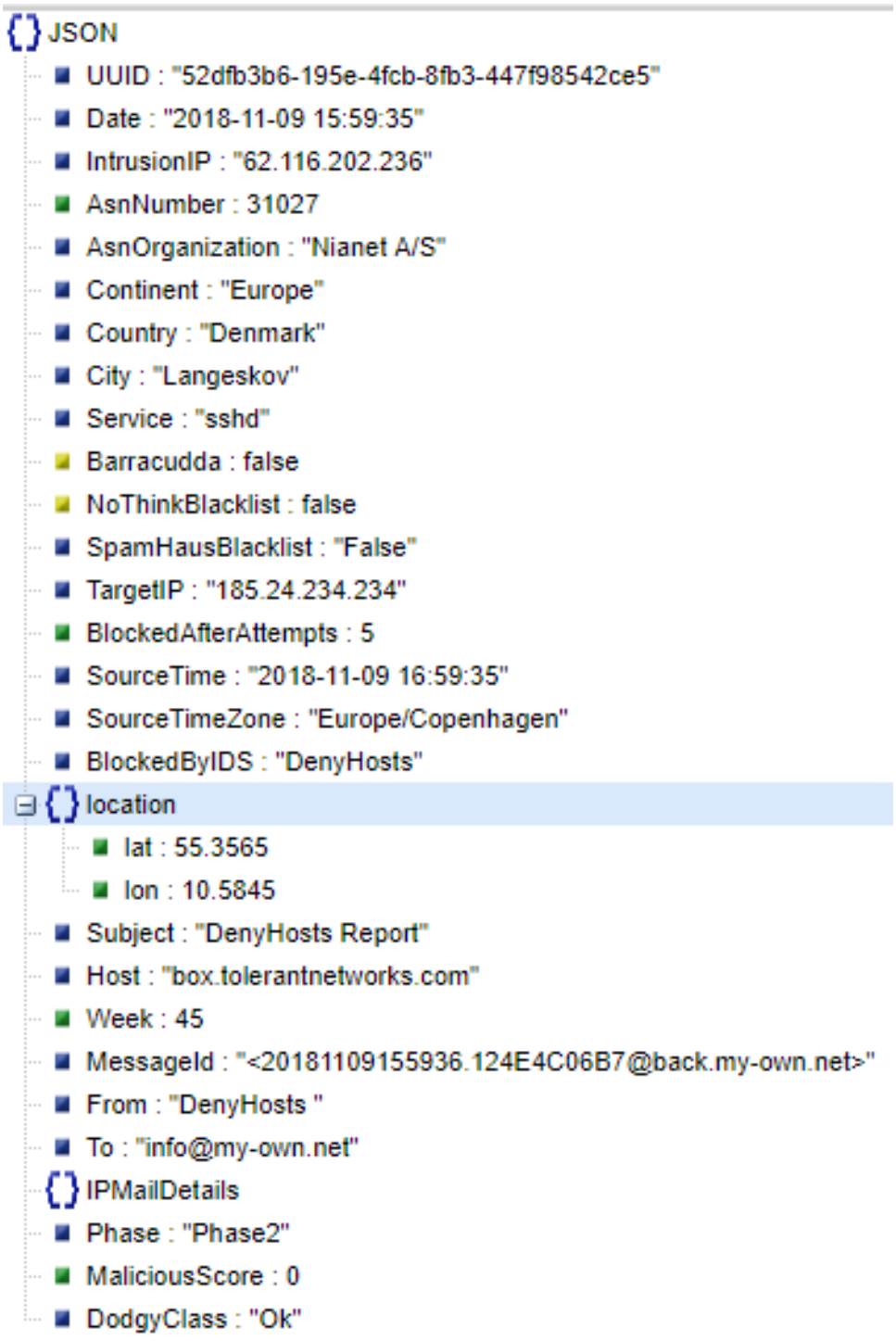


Figure 4.9: Sample json exported for an intrusion attempt

A sample intruding "IP" json is shown in Figure 4.10.

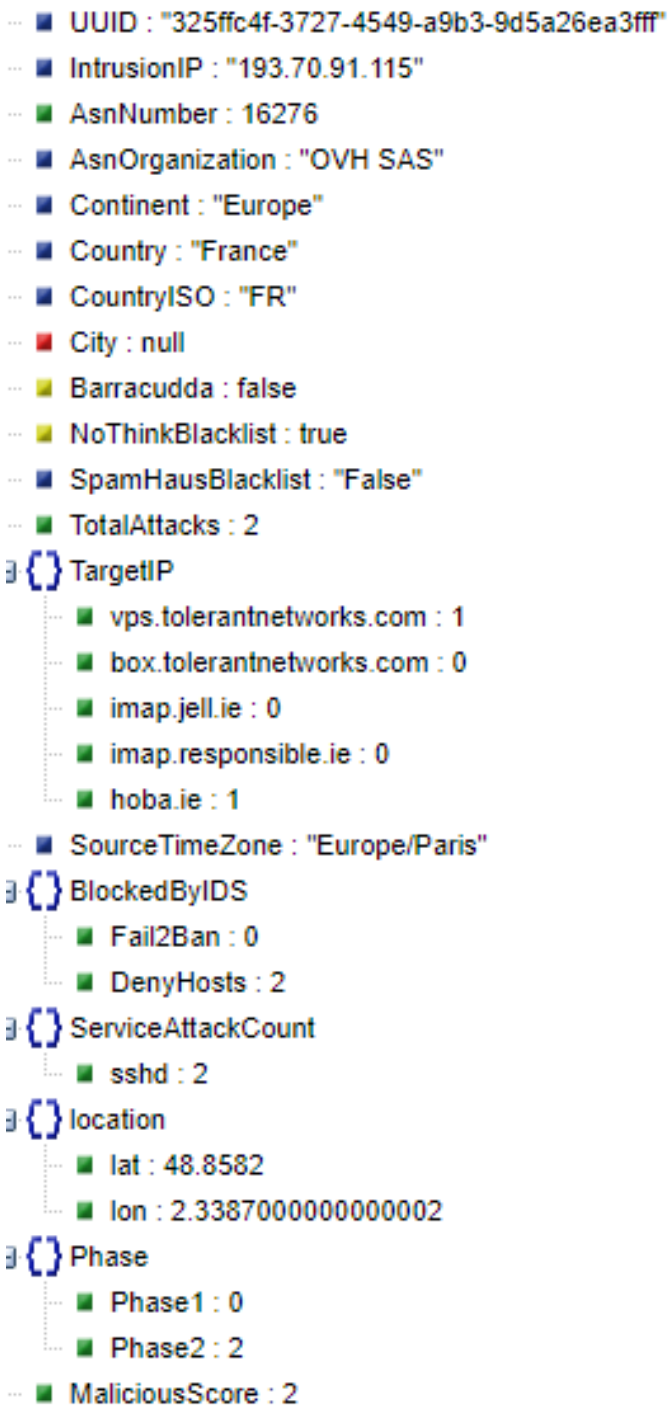


Figure 4.10: Sample json exported for an intruding IP

The "Attack" and "IP" jsons were indexed in ElasticSearch and analysis was carried out using visualizations which were created using Kibana. The visualizations along with their analysis is shown in chapter 5. A snippet of the csv exported for intrusion attempts is shown in Figure 4.11. The csv data was used as input for creating the machine learning models as explained in Section 4.4. The result and discussion of the performance of the models is discussed in chapter 5.

B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
IntrusionIP	Date	AsnNumber	AsnOrgan	Continent	Country	City	Service	Week	Barracudd	NoThinkB	SpamHaus	TargetIP	BlockedAt	BlockedBy	SourceTim	Source	lat	lon	Phase	Maliciou	DodgyClass
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok
101.89.216.223	#########	4812	China Tele	Asia	China		postfix-	32	FALSE	FALSE	PBL	92.51.243.	2	Fail2Ban	#########	Asia/Sh	31	121	Phase	1.5	Dodgy
218.92.0.208	#########	4134	No.31,Jin-	Asia	China		sshd	32	FALSE	FALSE	FALSE	185.24.234	5	Fail2Ban	#########	Asia/Sh	32	119	Phase	0	Ok

Figure 4.11: Snippet of exported csv for intrusion attempts

Chapter 5

Results and Discussion

In the previous chapter 4 the implementation of the proposed software pipeline as per the methods explained in chapter 3, were described. The data was successfully collected, transformed, features added, persisted and exported. The results, in terms of the analysis carried out with the help with different visualization techniques and the performance of the machine learning models (Section 2.2.4) are discussed in this chapter.

5.1 Analysis

Based on the data available, the analysis can be carried out from 3 perspectives: Phase 1, Phase 2 and Overall with both phases combined. This section will provide a detailed discussion of the analysis based on the overall time frame. The phase wise graphs will be provided in the appendix for detailed viewing. The visualizations created were placed in a dashboard that was created on Kibana [82]. The snippets of the visualizations and the dashboards will be shown in various parts of this section.

5.1.1 Intrusion Numbers and Attack Trends

Table 5.1 shows the phase wise and overall number of intrusion attempts and the unique intrusion IPs. Figure 5.1 shows the graphical representation from the dashboard view created. It can be seen that the number of attacks have increased by approximately 20% and number of unique intruders increased by 83% between Phase 1 and Phase 2.

Table 5.1: Phase wise number of records collected for analysis

Phase	Duration	Intrusion Attempts	Unique Intruding IPs
Phase 1	June 22, 2017 - October 22, 2018	9,663	2462
Phase 2	October 23, 2018 - August 7, 2019	11,526	4526
Overall	June 22, 2017 - August 7, 2019	21189	6743

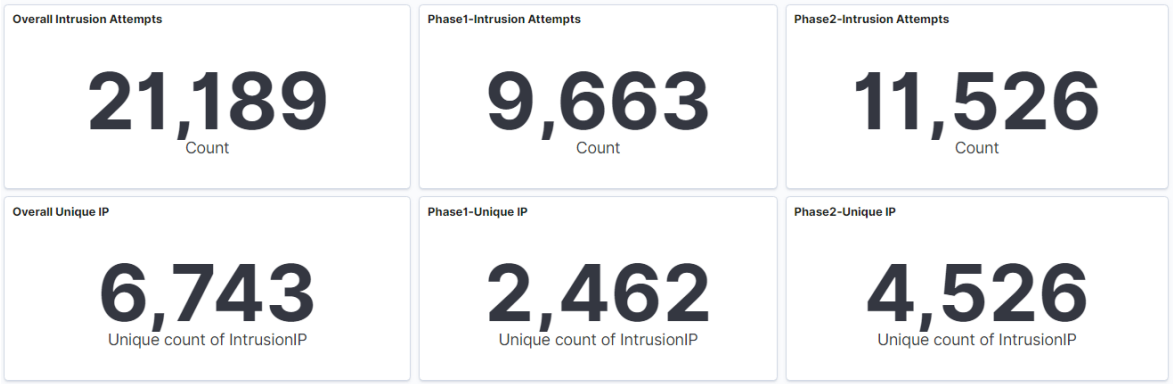


Figure 5.1: Dashboard providing summary of the intrusion attempts and unique attackers seen during each phase and overall

Figure 5.2 shows the number of VPS targeted by a unique intruding IP for the overall 26 months timeframe. To translate it to words, 4000+ unique intrusion IPs attacked any one of the 4 VPS, while just less than 500 unique intruders attacked all 4 VPS. Nearly 60% of the intruding IPs attempted a breach on one of the 4 severs being analysed. The phase wise pattern is similar and can been seen in the Appendix .2, Figure 1 and Figure 2.

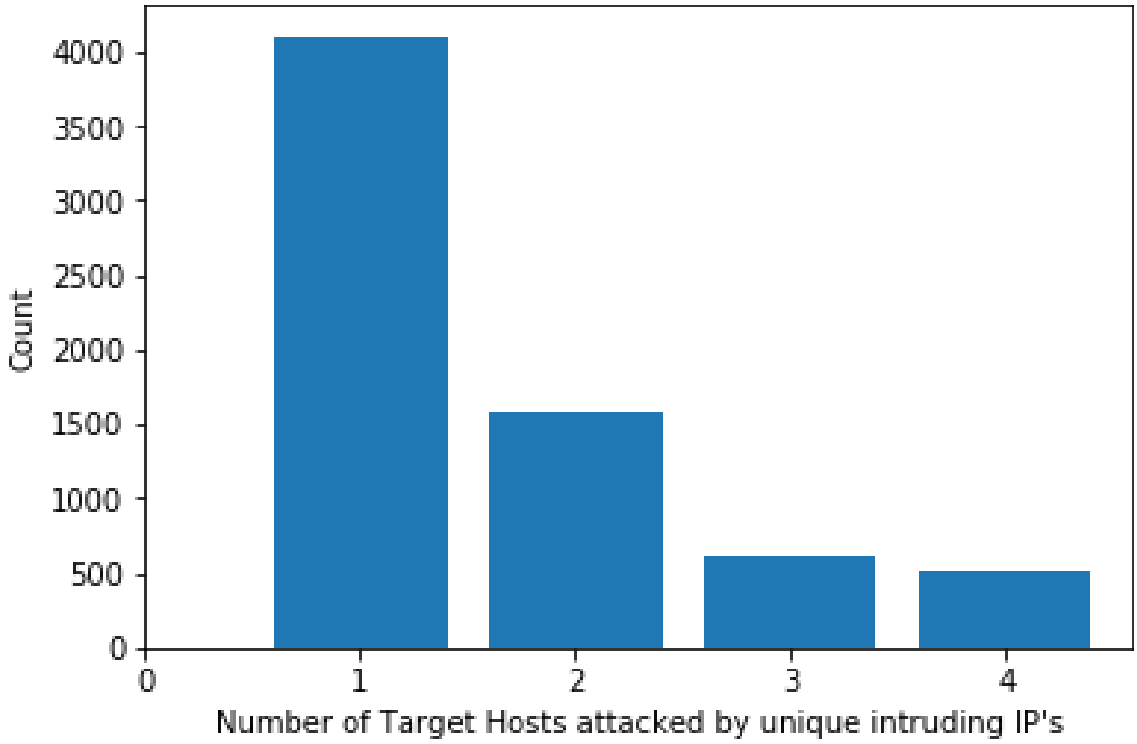


Figure 5.2: Number of hosts attacked by a unique source IP during the overall time frame

5.1.2 Geo-Spatial Trends

The geographical data obtained from each IP can be inaccurate based on the information updated and available with the respective registry [66]. It is also a possibility that a proxy is used by the attackers to mislead security investigations. Intrusion attempts from 145 countries were observed in the overall time frame, while in Phase 1 and Phase 2, 121 and 137 unique countries were observed respectively. The geographic distribution of intrusion attempts across the overall time frame can be seen in Figure 5.3. Phase wise geographic attack distribution can be seen in Appendix .4, Figure 5 and Figure 6.

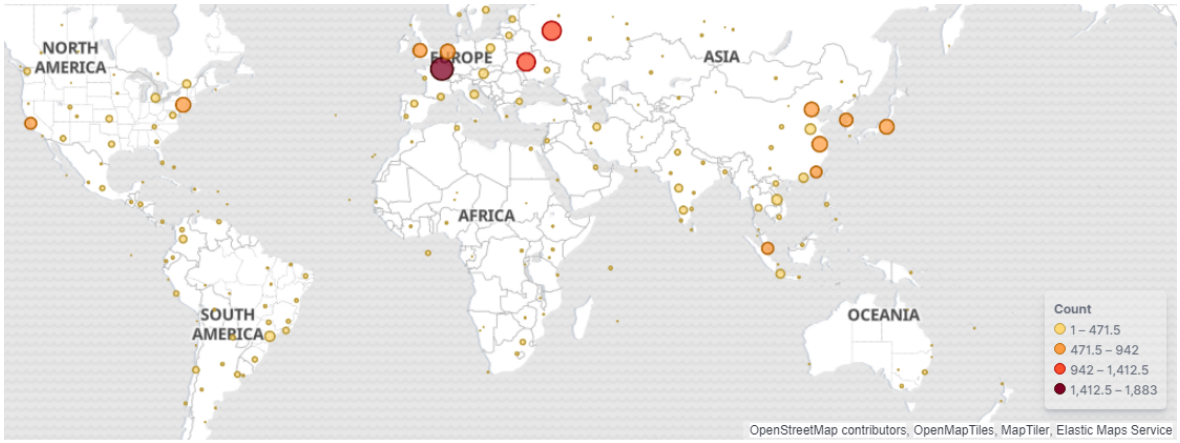


Figure 5.3: Geographic distribution of the 21189 intrusion attempts observed during the overall time frame

The geographic distribution of the unique intruding IPs based on their country of origin across the overall time frame can be seen in Figure 5.4. Phase wise geographic distribution of unique attacking IPs can be seen in Appendix .5, Figure 7 and Figure 8.

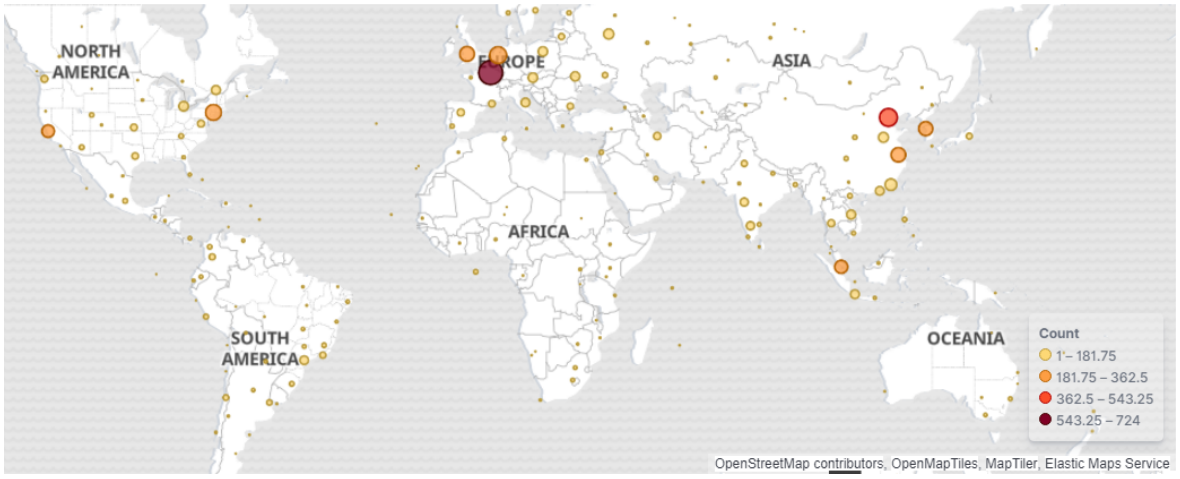


Figure 5.4: Geographic distribution of the 6743 unique intruders observed during the overall time frame

Top 10 Country Analysis

Based on the available data as shown in Figure 5.5, China and United States are the countries from where nearly 24% of all the attacks were observed during the overall time frame. An interesting point to note is that in Phase 1, Russia was the 2nd highest

source of attacks after China, but in Phase 2 Russia was observed at 9th position. China was still a top source from where attacks originated in Phase 2 and was positioned 2nd, while the highest number of intrusion attempts were seen from United States. Phase wise top 10 country of attack origination can be seen in Appendix .3, Figure 3 and Figure 4.

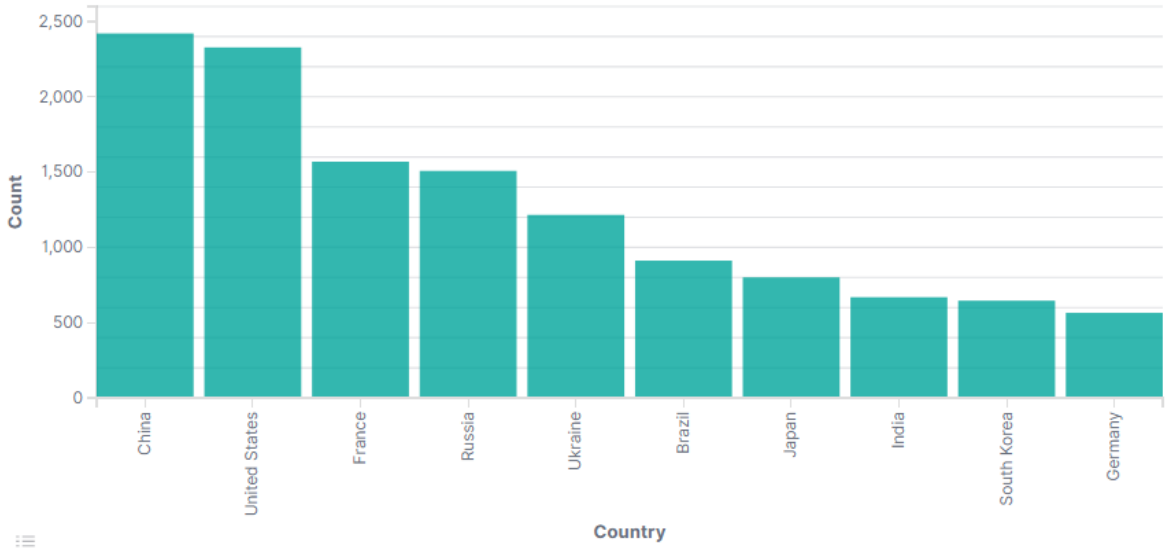


Figure 5.5: Top 10 countries as source of attacks

5.1.3 Temporal Trends

The temporal trends in the processed data was analyzed from the monthly, weekly and hourly perspective.

1. **Monthly**

   The overall rate of attacks per month during the overall time-frame for which the data is collected is shown in Figure 5.6. The colour coding represents the phase during which the attacks were observed. August 2018 to December 2018 were the months where a large number attacks occurred across the 4 target servers. This can largely be attributed to one host - hoba which received a higher number of intrusion attempts compared to the other hosts, however, tolerantnetworks also had a higher number of attempts in November 2018 and December 2018 as seen in Figure 5.7.

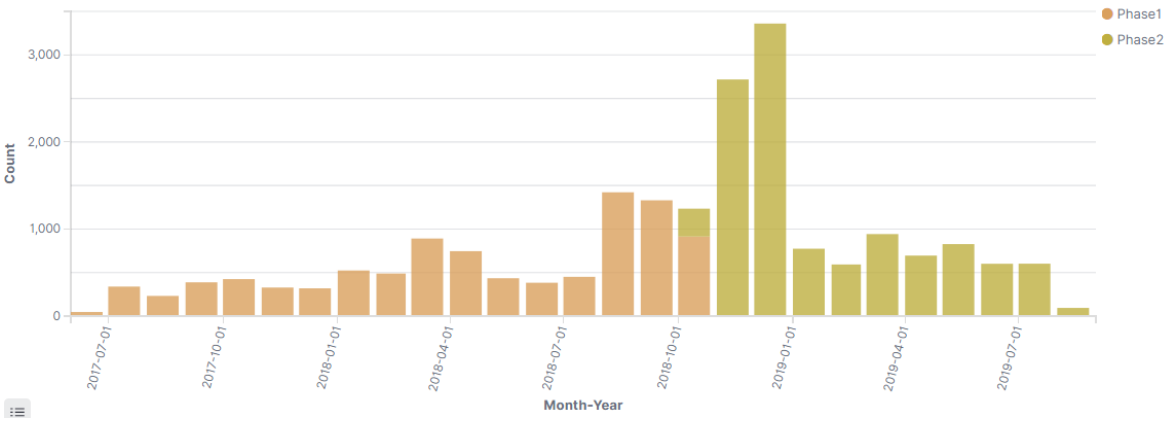


Figure 5.6: Number of Attacks per month across all hosts

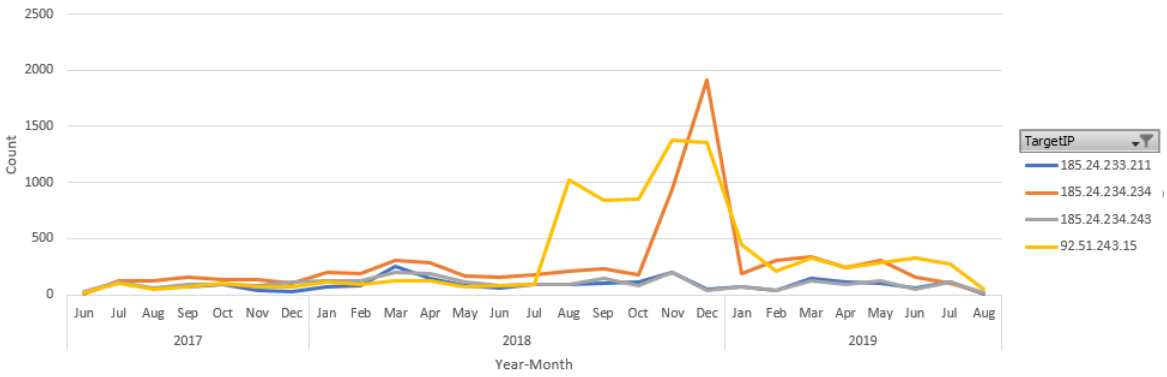


Figure 5.7: Number of Attacks per host per month during the entire time-frame. Jun 2017 - Oct 2018 can be considered as Phase 1 and Nov 2018 - Aug 2019 as Phase 2

2. **Weekly**

   The week number in the calendar year was extracted from the date during the feature engineering phase as described in Section 4.3.6. The frequency of attacks per calendar week of the year was plotted as can be seen in Figure 5.8. The stacked bars in the graph shows a split of the target hosts which were attacked. Week 48 and 49 are the calendar weeks that have faced the most intrusion attempts, with hoba and tolerantnetworks(box and vps) as the major targets.

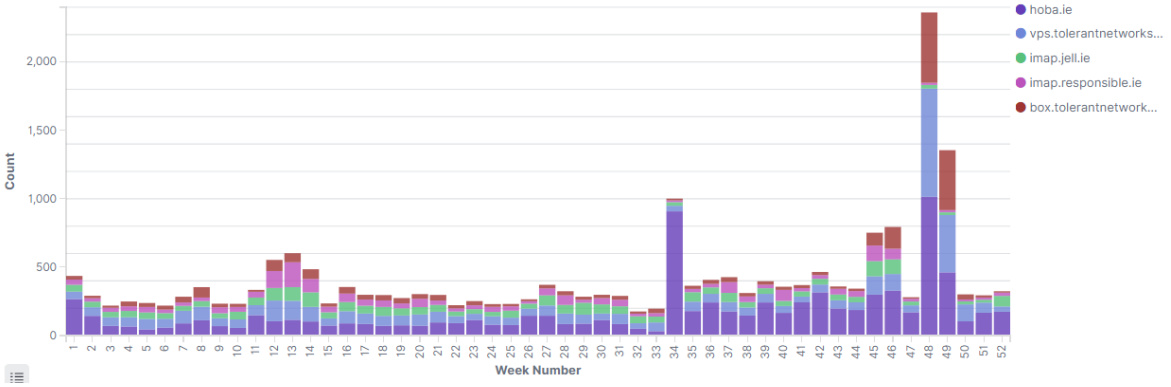


Figure 5.8: Distribution of attacks per calendar week of the year

3. **Hourly**

   During the data collection, the date/time of the attack was converted to the timezone of the respective origin country as explained in Section 4.3.2. An interesting facet being observed here is the peak time across different locations from where intrusion attempts originate. It can be observed from Figure 5.9 that the afternoon timing between 14:00-15:00 during the overall time frame is the peak hour with 1802 attempts observed across different countries in their respective time-zone for launching attacks. In Figure 5.10 the hourly distribution of intrusion attempts faced in the target time-zone is shown. 00:00-01:00 is the most vulnerable hour for the target VPS with 1672 intrusion attempts detected by the IDS running on the target servers.

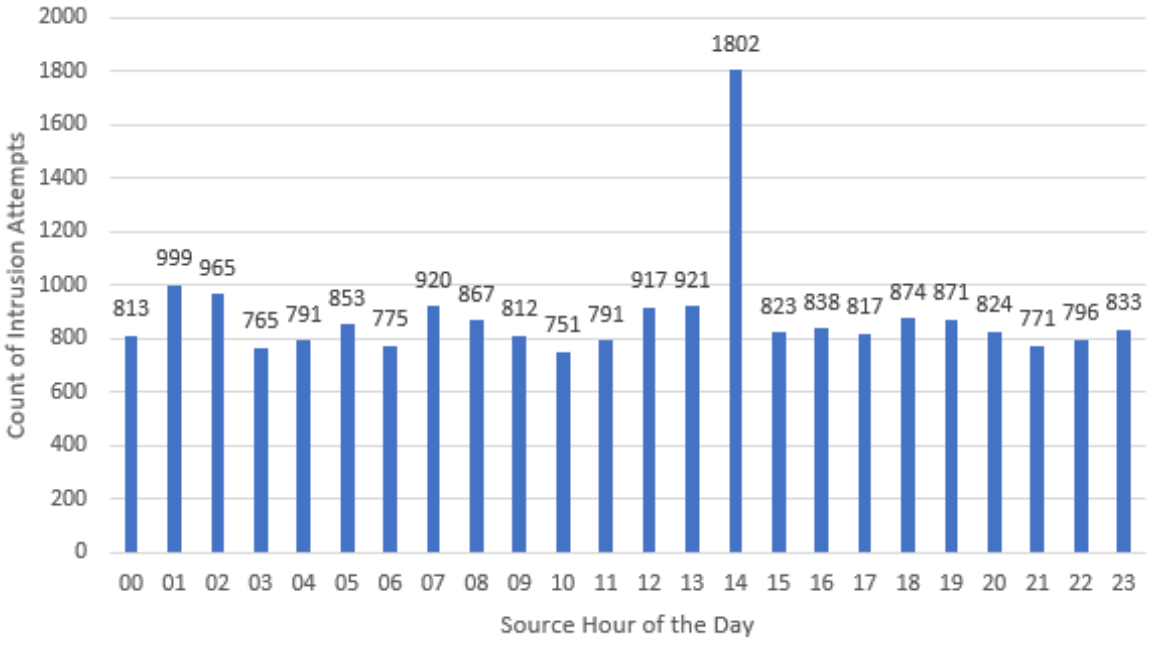


Figure 5.9: Hourly distribution of attack origination across different countries in their respective source time-zone

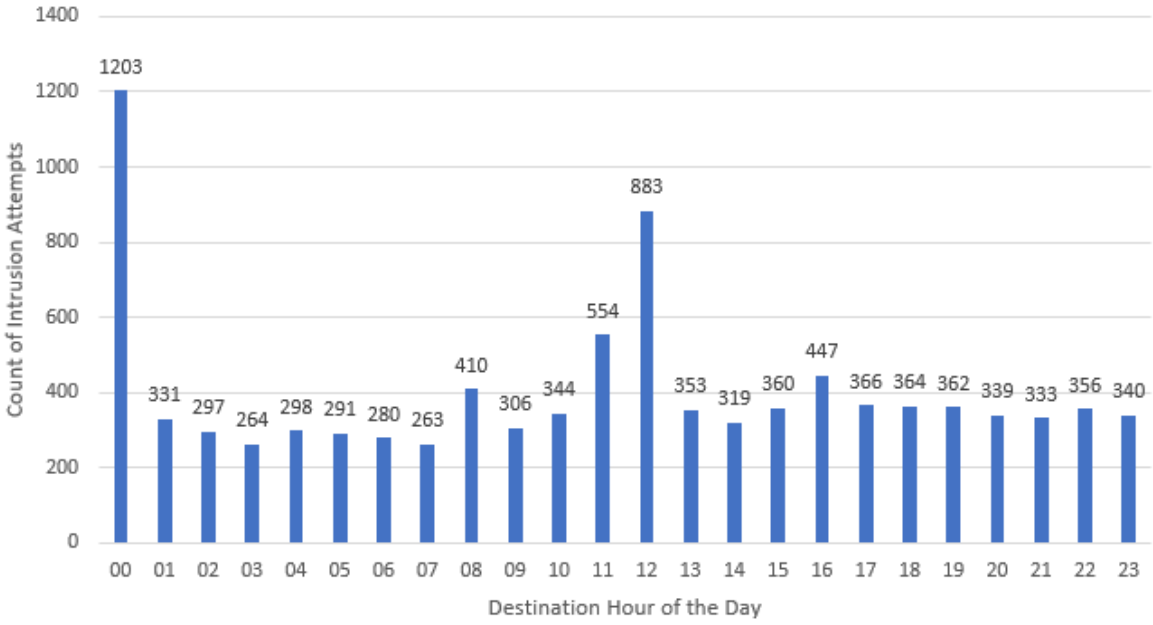


Figure 5.10: Hourly distribution of attacks in the target (Ireland) time-zone

Phase wise temporal patterns for both source and target time-zones can be seen in Appendix .6, Figure 9, Figure 10, Figure 11 and Figure 12.

5.1.4 Intrusion IPs Blocked by the IDS

It is important to analyze the IP addresses from which intrusion attempts were detected. The scope of the analysis is limited to the top 10 intrusion's detected across the 4 VPS in each phase. The top 10 IP addresses from which intrusions were detected by the IDS in the overall time frame are shown in Figure 5.11. The figure also shows a view of the target hosts each of the top 10 intruding IPs attempted to breach. The IP with the highest number of intrusion attempts, "185.110.112.49" primarily targeted "jell.ie" and "vps.tolerantnetworks.com".

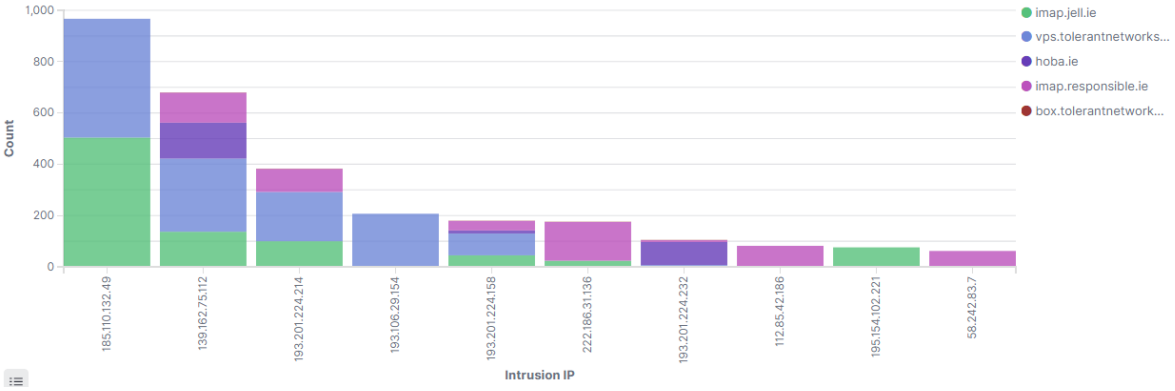


Figure 5.11: Top 10 intrusion IPs in terms of number of attacks detected in the overall time frame

Detailed Analysis of Top 10 Intrusion IP Addresses

Additional level of analysis was carried out for the top 10 IP by checking the number of attacks for each they attempted against each of the target hosts while also factoring in the dodgy class calculated for each of the intruding IP address in the entire timeframe as seen in Figure 5.12. The saturation of the color red represents the number of attacks.

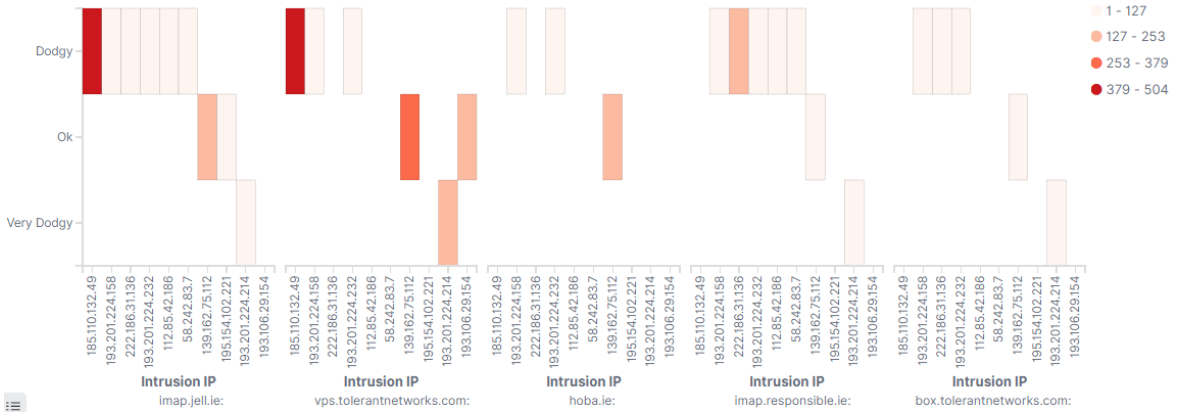


Figure 5.12: Top 10 intrusion IPs in terms of number of attacks detected, host targeted and the dodgy level calculated in the overall time frame

Phase wise detailed analysis of the top 10 intruding IP addresses is shown in Appendix .7, Figure 13 and Figure 14. An interesting point to note is that some of the top 10 IP addresses from both the phases overlap, indicating that the intruders are persistent. There were 16 unique IP addresses intersecting among the top 10 attackers in both the phases.

A detailed summary of the 16 unique intrusion IP addresses found to be in the top 10 attackers list for Phase 1 and Phase 2 can be seen in Table 5.2.

Table 5.2: A summary of the unique top 10 IP addresses based on number of attacks that were detected by the IDS running across the 4 VPS - Overall time frame

Intrusion IP	**ASN**	**DodgyClass**	**Phase**	**Intrusion Attempts**	**Country**
112.85.42.186	CHINA UNICOM China169 Backbone	Dodgy	Phase2	82	China
116.31.116.26	CHINANET Guangdong province network	Dodgy	Phase2	35	China
139.162.75.112	Linode, LLC	Ok	Phase1	658	Japan
185.110.132.49	Rise-v, Ltd.	Dodgy	Phase1	901	Russia
185.110.132.49			Phase2	66	
188.92.77.235	Sia Nano IT	Dodgy	Phase2	38	Latvia
193.106.29.154	Infium, UAB	Ok	Phase2	207	Ukraine
193.201.224.158	PE Tetyana Mysyk	Dodgy	Phase1	110	Ukraine
193.201.224.158			Phase2	70	
193.201.224.214		Very Dodgy	Phase1	250	
193.201.224.214			Phase2	133	
193.201.224.232		Dodgy	Phase1	72	
193.201.224.232			Phase2	33	
193.201.224.241		Dodgy	Phase1	36	
195.154.102.221	Online S.a.s.	Ok	Phase1	76	France
222.186.31.136	AS Number for CHINANET jiangsu province backbone	Dodgy	Phase1	176	China
23.249.165.200	ColoCrossing	Dodgy	Phase2	48	United States
49.88.112.71	No.31,Jin-rong Street	Dodgy	Phase2	44	China
58.218.198.161	AS Number for CHINANET jiangsu province backbone	Ok	Phase1	56	China
58.242.83.7	CHINA UNICOM China169 Backbone	Dodgy	Phase1	62	China

ASN Word Cloud

A word cloud representation of the AsnOrganization feature was created on the Kibana dashboard.

Figure 5.13: ASN Organization word cloud for overall time-frame

Phase wise AsnOrganization word cloud is shown in Appendix .7, Figure 15 and Figure 16

5.1.5 Remote Services Attack Trends

The intrusion attempts detected were primarily mounted against remote services such as SSH and SMTP (postfix). Figure 5.14 shows the major remote services attacked across the entire time frame among the VPS being monitored. It can be seen that sshd remote service on all the hosts, was largely targeted by the intruders, while "hoba" was also attacked on the postfix-sasl service.

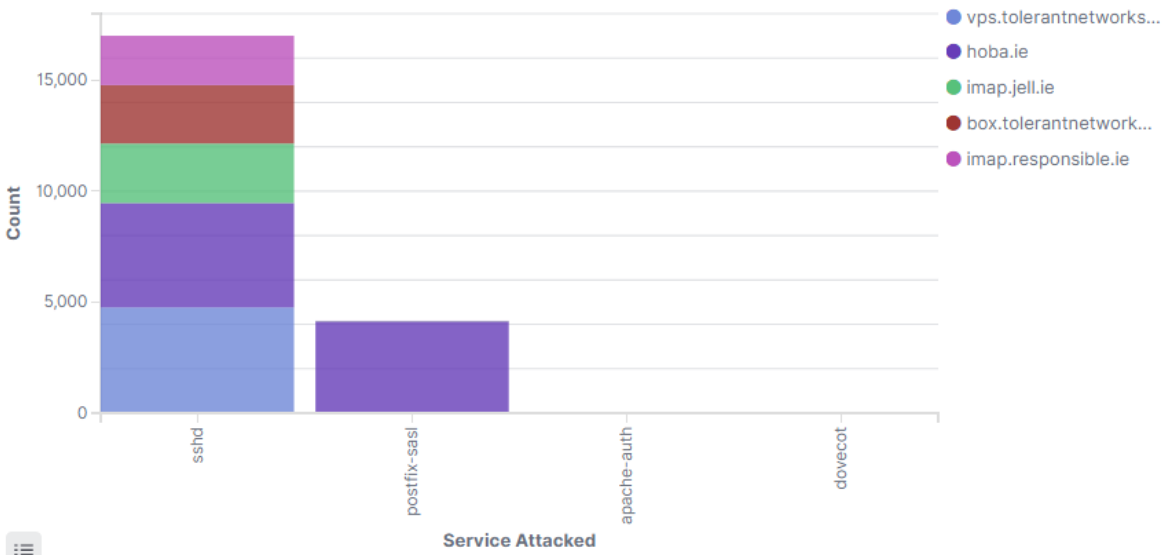


Figure 5.14: Count of intrusion attempts against remote services running on the VPS for the overall time-frame

80% of the intrusion attempts were against the sshd remote service. postix-sasl was the next major service attacked with 20% of the attacks. The intrusion attempts against the other remote services were negligible as shown in Figure 5.15.

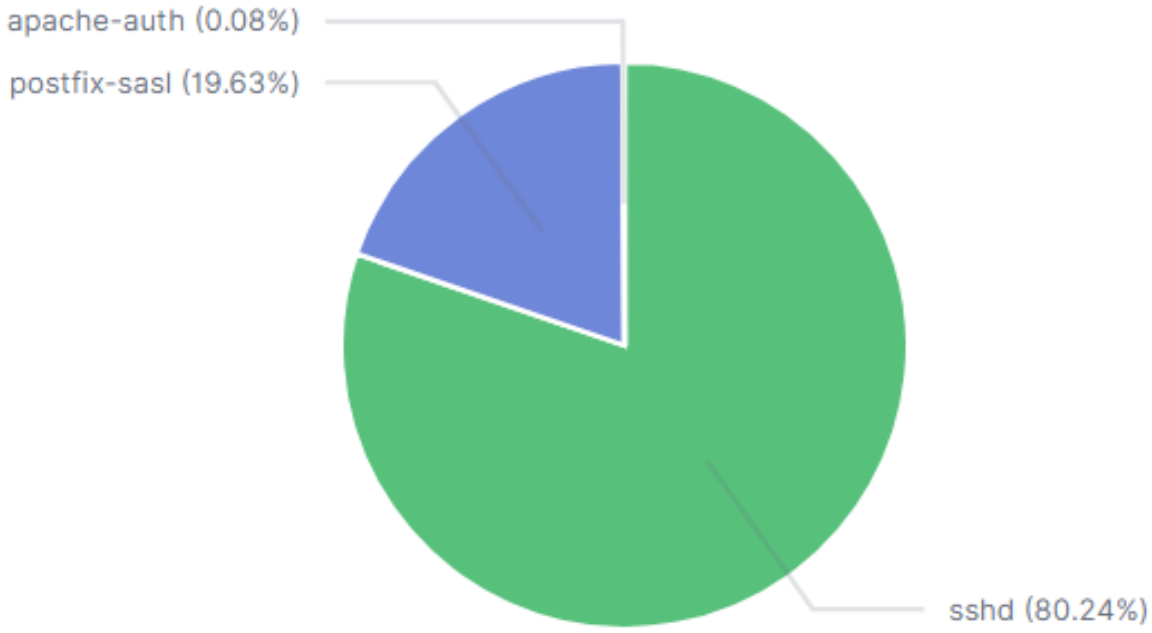


Figure 5.15: Percentage distribution of remote services attacked during the overall time-frame

Phase wise count of services attacked and its percentage distribution is shown in Appendix .8, Figure 17, Figure 18, Figure 19 and Figure 20.

5.1.6 Discussion

The exploratory analysis of the intrusion attempts gathered over a period of 26 months uncovered interesting geo-spatial, temporal and other trends with respect to persistence of an attacker and the popular services targeted. The visualizations shown in this section were captured from the security dashboard created on Kibana as a part of this research work. 3 different perspectives were provided in terms of Overall trends, Phase 1 trends and Phase 2 trends.

When looking into the detailed analysis of the top 10 attacking IP addresses from Figure 5.12, Figure 13 and Figure 14, it is observed that in the *Overall* and *Phase 1* time frame, just 30% of the top 10 IP addresses had low threat level, whereas in *Phase 2* just 10% of the top 10 IP addresses had a low threat level. Also, it can be seen from Table 5.2 the intruders are getting smarter and using distributed servers(example:

193.201.224.*) in the same subnet to avoid getting flagged by the IDS rules. This was also a finding in the research work by Owens et. al. in [24]. Though a majority of the IP addresses investigated in the top 16 unique intruding IP addresses were found to be dodgy, there could be some ethical security researchers attempting to intrude into systems to verify the security of servers on the web and identify vulnerabilities before they are exploited by bad actors. This analysis could also be used by security researchers and administrators to identify persistent attackers faster and place requisite blocks in place to safeguard the systems.

5.2 Prediction Results

The metrics to evaluate the machine learning algorithms in this work were defined in Section 3.4. The prediction results are stated below.

5.2.1 Time-Series Forecast

The time-series forecast was run on the Phase 1 data. The predictions generated were for the 4 months following October 22, 2018. In general, it is a rule of thumb to regularly re-train the model to capture the changing trends. Therefore, apart from the 4 months following the Phase 1 data, the model was also validated for the 15 days post October 22, 2018. A forecast of the rate of intrusion attempts was made for each of the target hosts. Additionally, one forecast was made for the rate of intrusion attempts expected from the observed continents, namely - Africa, Asia, Europe, North America, South America and Oceania.

For each of the prediction plot discussed in this section, black dots represent the observed points or train data. The blue range represents the uncertainty in the data. The blue line from October 2018 until February 2019 represents the prediction. The prediction plot for the target host hoba as seen in Figure 5.16. As per the forecast model, the prediction is of a drastic spike in the number of attacks over the 4 months following Phase 1.

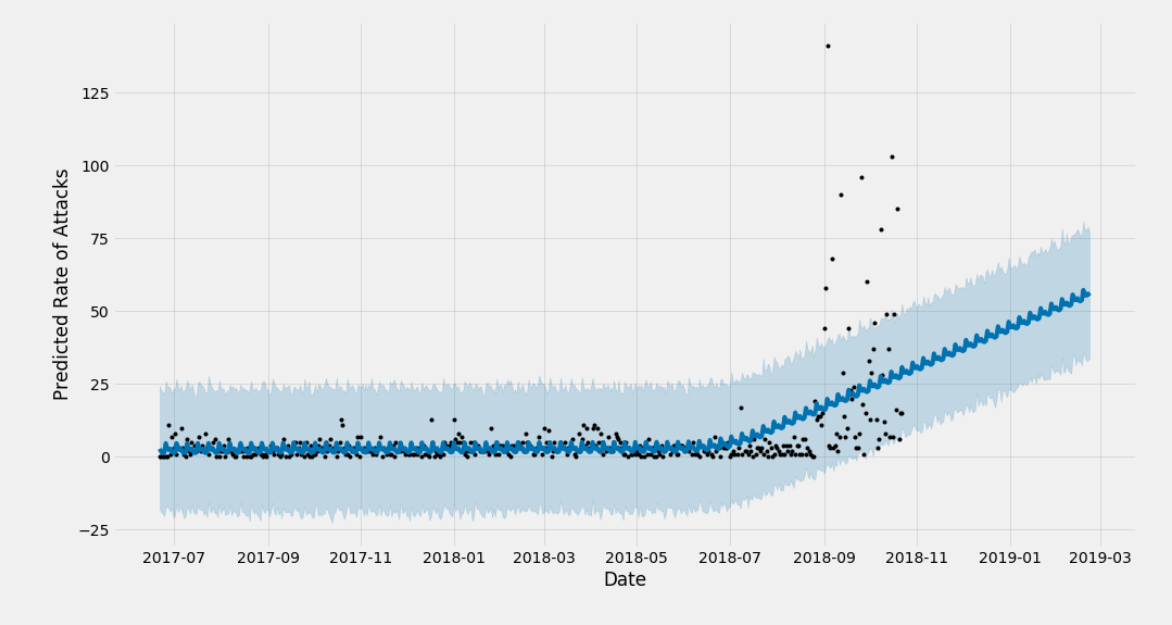


Figure 5.16: The time-series prediction plot for hoba

Similarly, from a continent perspective, the prediction plot for Europe is shown in Figure 5.17. This plot has higher uncertainty due to the scattered data points as represented by the black dots.

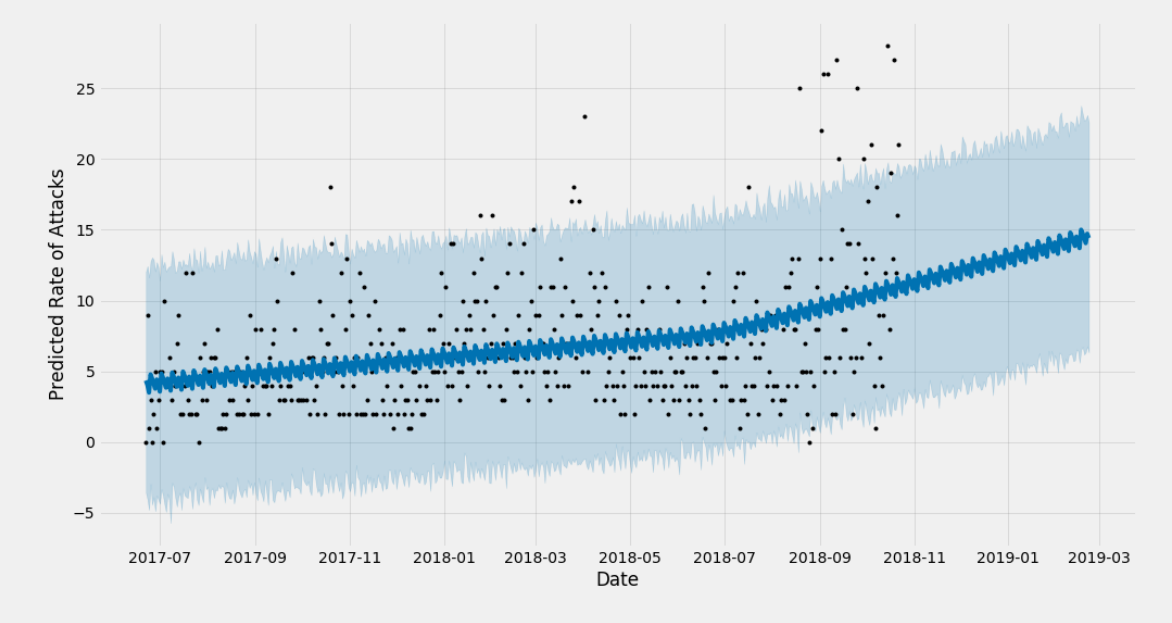


Figure 5.17: The time-series prediction plot for Europe

The plots for the remaining target hosts and continents can be seen in Appendix .9.

Table 5.3 shows the summary of RMSE and R2 scores obtained for each of the prediction's validation set.

Table 5.3: A summary of each of the prediction generated, their validation days and respective R2 and RMSE

Prediction Type	**Host**	**Validation Days (post October 22, 2018)**	**R2**	**RMSE**
Target Host	hoba	123	-0.23	50.23
		15	-0.06	25.02
	responsible	123	0.00	4.75
		15	-0.16	7.35
	jell	123	0.03	4.05
		15	-0.12	2.83
	tolerantnetworks	123	-0.10	70.42
		15	-0.08	7.00
Continent	Africa	123	-0.31	3.06
		15	-0.02	2.25
	Europe	123	-0.08	44.77
		15	-0.26	8.00
	Asia	123	-0.05	34.01
		15	-0.01	11.94
	North America	123	-0.05	22.63
		15	-0.03	4.89
	South America	123	-0.52	13.54
		15	-0.03	10.30
	Oceania	123	-0.07	1.46
		15	-0.08	1.05

5.2.2 Threat Level Classification

The Phase 1 dataset used to train the XGBoost classifier was split into train and test set in the ratio 90:10. The test accuracy obtained was 92.74%. The trained model was

then validated using the data from Phase 2 for the 20 days following October 22, 2018. Figure 5.18 shows the confusion matrix to evaluate the accuracy of the multi-class classification problem. The diagonal values represent the correctly predicted cases for each class. The validation set accuracy was found to be 75.46%.
Note: in Figure 5.18 "normal" represents the "Ok" class and "harmful" represents the "Very Dodgy" class defined in Section 4.3.5. This mapping was done for ease of interpretation of the figure.

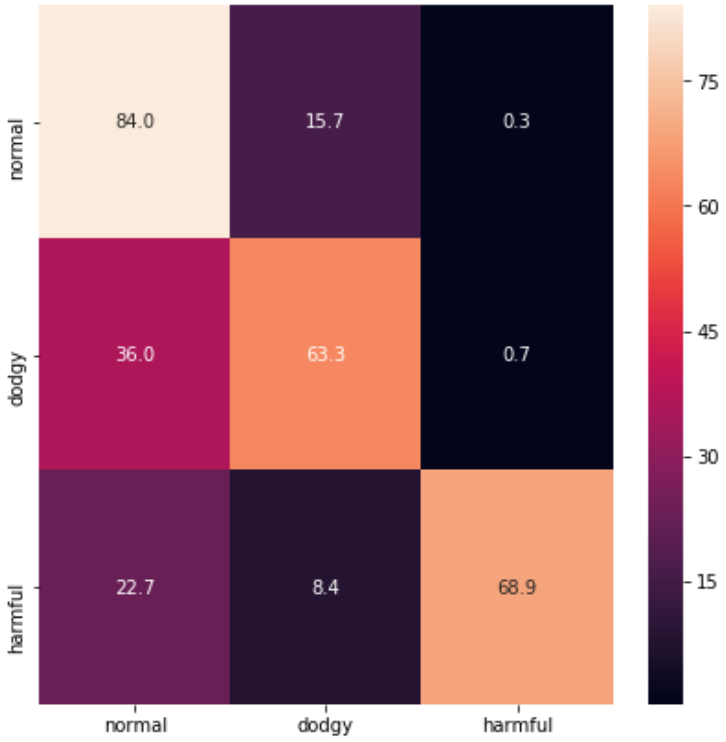


Figure 5.18: Confusion matrix comparing the actual values (y-axis) against the predicted values (x-axis) for each of the 3 classes

5.2.3 Discussion

The predictive analysis part of this research work produced interesting results. The **time-series forecast** model for estimating the rate of attacks, produced inaccurate predictions for both target hosts and continent. The sparseness of the respective subset of the Phase 1 data used to train the model and high variation of rate of attacks can

be the cause of the underfitted model. The forecast was validated using 2 subsets of the Phase 2 data. One, for the 4 months following October 22, 2018 and another for 15 days following this cut-off date. This was done to ensure that the variability in data and model trained with older data doesn't affect the performance. However, in both cases it was found that the R2 score and RMSE which should be in the range of 0-1, and ideally closer to 1 and 0 respectively, were way off the mark as shown in Table 5.3. This could be attributed to the following factors:

1. constantly changing attack patterns and improved ability of attackers
2. the time-series forecasting method used in this research work is not suitable for this type of data though it was found to be successful in other domains [55]
3. potentially incorrect data on some days, due to system maintenance or remote service restarts

The **XGBoost classification** model, was validated against 20 days(after October 22, 2019) of real world data collected in Phase 2. The model created was able to accurately predict the threat level of 75.46% of the intrusion attempts, i.e. 3 out of 4 intrusions attempts. This model coupled with the analytics dashboard described in Section 5.1 could help administrators take informed decisions and improve the security of systems.

Chapter 6

Conclusion and Future Work

The previous chapter 5 provided the results obtained in this research work and a commentary on the same. In this chapter, the concluding remarks, limitations of this work and the scope for future work will be discussed.

6.1 Conclusion

Intrusion Detection Systems are undeniably an integral part of any network or internet facing server. It was seen through the course of this work, that the open-source IDS - Fail2Ban and DenyHosts are effective in defending the servers from intruders. Unlike much of the literature surveyed in this research, the data used for the analysis was from live production systems(though not having high visibility), reflecting the trend of attacks in the real world. The time-frame during which the data was collected, is elaborate spanning over a span of 26 months, thus making the data source largely reliable and authoritative. The software pipeline created is able to ingest IDS data and create meaningful structured objects that can be analyzed. The dodgy parameter created using a weighed model taking into account the service attacked and public reputation of an intruding IP address provided a good measure of the threat perception from each attack/unique attacker. The security dashboard created for exploratory analysis is able to provide actionable insights that can be used to enhance the security of the systems. The predictive analysis provided interesting results. It was observed in this research work that time-series forecasting using machine learning, is not a good

approach to estimate the rate of attacks which a system may be subject to, though it was found to effective in other domains during the literature survey [1, 55, 53]. This can be attributed to the uncertainty around attacks patterns as well as the intruders getting smarter. The classification model created to predict the threat-level for an attacker/attacking IP address performed well with accuracy of 92.74% on the test data (10% of Phase 1 data) and accuracy of 75.46% on validation data (20 days post Phase 1)- representing real world data.

In the past, there has been extensive research in the analysis of IDS [11, 26, 14] and in using machine learning directly for intrusion detection, independent of the existing IDS [23, 7, 29, 28]. This work showcases that data from IDS and machine learning approaches can be used to enhance the security of systems by leveraging the meaningful insights available from historic data and assigning a threat level to the intrusion attempts.

To conclude the story about farmer A introduced in chapter 1, if the farmer is able to install the right open-source fencing around the farm and gather details about the intrusion attempts by the adversaries, the system developed in this research will be able to help the farmer enhance the security of the farm.

6.2 Limitations

- The data set could comprise of a small number of duplicate intrusion attempts. In cases of system or IDS restart, the IDS system triggers a new notification for previously reported intrusion attempts. The intrusion is consequently reported with a new timestamp and message-id

- Denyhosts ony detects intrusion attempts against the sshd service. This is one of the reasons for the dominance of the sshd service in the analysis discussed in Section 5.1.5

- Fail2Ban sends a separate notification for each intrusion attempt detected whereas Denyhosts can flag multiple intrusion attempts in a single notification. This leads to all the intruding IP addresses in a Denyhosts notification getting tagged to the same date/time as per the current program design. This could explain the peaks observed in hourly distribution in Figure 5.9 and Figure 5.10

6.3 Future Work

- The software pipeline created currently works on historic data and required manual data export. This pipeline could be updated to work on streams of data from IDS systems thereby providing near-time status of the system on the security dashboard
- The machine learning models are efficient in prediction for a limited time-span post the initial training. They need to be regularly re-trained to improve their performance. In the future work, the historic data can be used to incrementally train and evaluate the model in multiple phases
- New dimension about the intrusion attempts can be looked at from the firewall perspective to gain more insights about the attackers

Bibliography

[1] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, pp. 37–45, 2018.

[2] I. Security, "Cost of a data breach report." https://www.ibm.com/security/data-breach/, 2019. Accessed: 2019-08-04.

[3] R. Mardisalu, "14 most alarming cyber security statistics in 2019." https://thebestvpn.com/cyber-security-statistics-2019/, May 2019. Accessed: 2019-08-04.

[4] M. Goddard, "The eu general data protection regulation (gdpr): European regulation that has a global impact," *International Journal of Market Research*, vol. 59, no. 6, pp. 703–705, 2017.

[5] "Eu general data protection regulation." https://eugdpr.org/, 2018. Accessed 2019-08-04.

[6] J. Moar, "Juniper research: The future of cybercrime & security: Financial and corporate threats & mitigation." https://www.juniperresearch.com/researchstore/innovation-disruption/cybercrime-security, August 2018. Accessed: 2019-08-04.

[7] J. Jabez and B. Muthukumar, "Intrusion detection system (ids): anomaly detection using outlier detection approach," *Procedia Computer Science*, vol. 48, pp. 338–346, 2015.

[8] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[9] M. Rouse, "What is intrusion detection system (ids)? - definition from whatis.com." https://searchsecurity.techtarget.com/definition/intrusion-detection-system, 2018. Accessed: 2019-08-04.

[10] S. R. Snapp, J. Brentano, G. Dias, T. L. Goan, L. T. Heberlein, C.-L. Ho, and K. N. Levitt, "Dids (distributed intrusion detection system)-motivation, architecture, and an early prototype," 2017.

[11] I. Koniaris, G. Papadimitriou, and P. Nicopolitidis, "Analysis and visualization of ssh attacks using honeypots," in *Eurocon 2013*, pp. 65–72.

[12] D. Schneier, "Ssh brute force attacks still going strong." https://itknowledgeexchange.techtarget.com/security-bytes/ssh-brute-force-attacks-still-going-strong/, 2018. Accessed: 2019-08-04.

[13] A. Carasik-Henmi, T. W. Shinder, C. Amon, R. J. Shimonski, and D. L. Shinder, eds., *Chapter 4 - Introduction to Intrusion Detection Systems*, pp. 111–124. Burlington: Syngress, 2003.

[14] S. S. Tirumala, H. Sathu, and A. Sarrafzadeh, "Free and open source intrusion detection systems: A study," in *2015 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 1, pp. 205–210.

[15] A. Pauna and I. Bica, "Rassh - reinforced adaptive ssh honeypot," in *2014 10th International Conference on Communications (COMM)*, pp. 1–6.

[16] R. Mitchell and I. Chen, "Effect of intrusion detection and response on reliability of cyber physical systems," *IEEE Transactions on Reliability*, vol. 62, no. 1, pp. 199–210, 2013.

[17] J. P. Anderson, "Computer security threat monitoring and surveillance," *Technical Report, James P. Anderson Company*, 1980.

[18] S. Axelsson, "Research in intrusion-detection systems: A survey," tech. rep., Technical report 98–17. Department of Computer Engineering, Chalmers , 1998.

[19] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on software engineering*, no. 2, pp. 222–232, 1987.

[20] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. JúNior, "An intrusion detection and prevention system in cloud computing: A systematic review," *Journal of network and computer applications*, vol. 36, no. 1, pp. 25–41, 2013.

[21] P. Aggarwal and S. K. Sharma, "Analysis of kdd dataset attributes-class wise for intrusion detection," *Procedia Computer Science*, vol. 57, pp. 842–851, 2015.

[22] "Kdd cup dataset." http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999. Accessed: 2019-08-04.

[23] M. M. Najafabadi, T. M. Khoshgoftaar, C. Calvert, and C. Kemp, "Detection of ssh brute force attacks using aggregated netflow data," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 283–288.

[24] J. Owens and J. Matthews, "A study of passwords and methods used in brute-force ssh attacks," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*.

[25] H. Koike, K. Ohno, and K. Koizumi, "Visualizing cyber attacks using ip matrix," in *IEEE Workshop on Visualization for Computer Security, 2005.(VizSEC 05).*, pp. 91–98, IEEE, 2005.

[26] S. McKenna, D. Staheli, C. Fulcher, and M. Meyer, "Bubblenet: A cyber security dashboard for visualizing patterns," in *Computer Graphics Forum*, vol. 35, pp. 281–290, Wiley Online Library, 2016.

[27] F. B. Manolache, H. Qingping, and O. Rusu, "Analysis and prevention of network password guessing attacks in an enterprise environment," in *2014 RoEduNet Conference 13th Edition: Networking in Education and Research Joint Event RENAM 8th Conference*, pp. 1–7.

[28] S. Mahdavifar and A. A. Ghorbani, "Application of deep learning to cybersecurity: A survey," *Neurocomputing*, vol. 347, pp. 149–176, 2019.

[29] M. M. Najafabadi, T. M. Khoshgoftaar, C. Kemp, N. Seliya, and R. Zuech, "Machine learning for detecting brute force attacks at the network level," in *2014 IEEE International Conference on Bioinformatics and Bioengineering*, pp. 379–385.

[30] R. Larue-Langlois, "Top 10 intrusion detection tools: Your best free options for 2019." https://www.addictivetips.com/net-admin/intrusion-detection-tools/, August 2018. Accessed: 2019-08-04.

[31] "Denyhosts." http://denyhosts.sourceforge.net/. Accessed: 2019-08-04.

[32] "Fail2ban." https://www.fail2ban.org/wiki/index.php/Main_Page/. Accessed: 2019-08-04.

[33] T. Ylonen and C. Lonvick, "Rfc 4253: The secure shell (ssh) transport layer protocol," *The Internet Society.https://tools.ietf.org/html/rfc4253*, 2006.

[34] M. Cotton, L. Eggert, J. Touch, M. Westerlund, and S. Cheshire, "Internet assigned numbers authority (iana) procedures for the management of the service name and transport protocol port number registry.," *RFC*, vol. 6335, pp. 1–33, 2011.

[35] J. Klensin, "Simple mail transfer protocol (rfc5321)," *Network Working Group, Internet Engineering Task Force. http://tools. ietf. org/html/rfc5321*, 2008.

[36] W. Venema, "Postfix." http://www.postfix.org/start.html/, 1998. Accessed: 2019-08-04.

[37] K. D. Dent, *Postfix: The Definitive Guide: A Secure and Easy-to-Use MTA for UNIX*. " O'Reilly Media, Inc.", 2003.

[38] F. Milicchio and W. A. Gehrke, "Electronic mail," *Distributed Services with OpenAFS: for Enterprise and Education*, pp. 237–262, 2007.

[39] R. Fielding and J. Reschke, "Hypertext transfer protocol (http/1.1): Semantics and content," *Internet Engineering Task Force. https://tools.ietf.org/html/rfc7231*, 2014.

[40] "Apache httpd." https://httpd.apache.org/docs/2.4/programs/httpd.html. Accessed: 2019-08-04.

[41] R. T. Fielding and G. Kaiser, "The apache http server project," *IEEE Internet Computing*, vol. 1, no. 4, pp. 88–90, 1997.

[42] "Apache authentication and authorization." http://httpd.apache.org/docs/current/howto/auth.html. Accessed: 2019-08-04.

[43] "Tolerant networks." https://tolerantnetworks.com/about-us.html. Accessed: 2019-08-04.

[44] E. A. Hall, "The application/mbox media type," 2005.

[45] S. Vadalasetty, "Security concerns in using open source software for enterprise requirements." https://www.sans.org/reading-room/whitepapers/awareness/security-concerns-open-source-software-enterprise-requirements-1305, 2019.

[46] "Cve details." https://www.cvedetails.com/. Accessed 2019-08-04.

[47] denyhosts, "Denyhosts releases." https://sourceforge.net/projects/denyhosts/files/denyhosts/. Accessed 2019-08-04.

[48] fail2ban, "Fail2ban releases." https://github.com/fail2ban/fail2ban/releases. Accessed 2019-08-04.

[49] T. M. Mitchell *et al.*, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, no. 37, pp. 870–877, 1997.

[50] T. O. Ayodele, "Types of machine learning algorithms," in *New advances in machine learning*, IntechOpen, 2010.

[51] M. Du, N. Liu, and X. Hu, "Techniques for interpretable machine learning," *arXiv preprint arXiv:1808.00033*, 2018.

[52] O. E. Dictionary, *"forecast, n."*. Oxford University Press.

[53] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[54] C. Chatfield, *Time-series forecasting*. Chapman and Hall/CRC, 2000.

[55] S. Kulkarni, S. N. Mandal, G. S. Sharma, M. R. Mundada, *et al.*, "Predictive analysis to improve crop yield using a neural network model," in *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 74–79, IEEE, 2018.

[56] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Machine learning: a review of classification and combining techniques," *Artificial Intelligence Review*, vol. 26, no. 3, pp. 159–190, 2006.

[57] P. Li, "Robust logitboost and adaptive base class (abc) logitboost," *arXiv preprint arXiv:1203.3491*, 2012.

[58] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), pp. 785–794, ACM, 2016.

[59] Z. Chen, F. Jiang, Y. Cheng, X. Gu, W. Liu, and J. Peng, "Xgboost classifier for ddos attack detection and analysis in sdn-based cloud," in *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 251–256, IEEE, 2018.

[60] D. Nielsen, "Tree boosting with xgboost-why does xgboost win" every" machine learning competition?," Master's thesis, NTNU, 2016.

[61] "Elasticsearch." https://www.elastic.co/. Accessed: 2019-08-04.

[62] M. Makadia, "What is elasticsearch and how can it be useful?." https://dzone.com/articles/what-is-elasticsearch-and-how-it-can-be-useful, 2017. Accessed: 2019-08-04.

[63] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide*. O'Reilly Media, Inc., 1st ed., 2015.

[64] J. Bai, "Feasibility analysis of big log data real time search based on hbase and elasticsearch," in *2013 ninth international conference on natural computation (ICNC)*, pp. 1166–1170, IEEE, 2013.

[65] J. L. S. Damas, A. Robachevsky, and D. Walker, "Ripe whois database query reference manual." https://www.ripe.net/publications/docs/ripe-358, 2005. Accessed: 2019-08-04.

[66] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye, "Ip geolocation databases: Unreliable?," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 2, pp. 53–56, 2011.

[67] B. Nikkel, "Registration data access protocol (rdap) for digital forensic investigators," *Digital Investigation*, vol. 22, pp. 133 – 141, 2017.

[68] MAXMIND, "Maxmind geoip2: Ip geolocation and online fraud prevention." https://www.maxmind.com/en/home. Accessed: 2019-08-04.

[69] MAXMIND, "Geolite2 free downloadable databases." https://dev.maxmind.com/geoip/geoip2/geolite2/. Accessed: 2019-08-04.

[70] "Nothink ssh blacklist." "http://www.nothink.org/blacklist/blacklist_ssh_all.txt". Accessed: 2019-08-04.

[71] A. Dhammi and M. Singh, "Behavior analysis of malware using machine learning," in *2015 Eighth International Conference on Contemporary Computing (IC3)*, pp. 481–486, IEEE, 2015.

[72] Y. Haga, A. Saso, T. Mori, and S. Goto, "Increasing the darkness of darknet traffic," in *2015 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, IEEE, 2015.

[73] "Spamhaus project." https://www.spamhaus.org/faq/section/DNSBL%2520Usage#202/, 1998. Accessed: 2019-08-04.

[74] "Get out of the haus! how to get delisted from spamhaus." https://blog.returnpath.com/

get-out-of-the-haus-how-to-get-delisted-from-spamhaus/, 2016. Accessed: 2019-08-04.

[75] J. Jung and E. Sit, "An empirical study of spam traffic and the use of dns black lists," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 370–375, ACM, 2004.

[76] A. Ramachandran, N. Feamster, and S. Vempala, "Filtering spam with behavioral blacklisting," in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 342–351, ACM, 2007.

[77] T.-S. Wang, H.-T. Lin, W.-T. Cheng, and C.-Y. Chen, "Dbod: Clustering and detecting dga-based botnets using dns traffic analysis," *Computers & Security*, vol. 64, pp. 1 – 15, 2017.

[78] "Prophet - forecasting at scale." https://facebook.github.io/prophet/, 2018. Accessed: 2019-08-04.

[79] "fbprohpet python library pypi." https://pypi.org/project/fbprophet/. Accessed: 2019-08-04.

[80] "Xgboost." https://xgboost.readthedocs.io/en/latest/. Accessed: 2019-08-04.

[81] "Scikit-learn machine learning in python." https://scikit-learn.org/stable/. Accessed: 2019-08-04.

[82] "Kibana." https://www.elastic.co/products/kibana. Accessed: 2019-08-04.

Appendix

The appendix section is used to provide supporting material for the dissertation such as regex patterns used, phase wise split graphs and additional prediction graphs.

.1 Pattern Matching

```
IP: '\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}'
orgname: 'org\-name\:\s+(.+?)\n'
inetnum: 'inetnum\:\s+(.+?)\n'
netname: 'netname\:\s+(.+?)\n'
intrusion attempts: 'Fail2Ban\safter\\n(.+?)\sattempts\s'
service: '\[Fail2Ban\]\s(.+?)\:'
```

.2 Intrusion Attack Trends

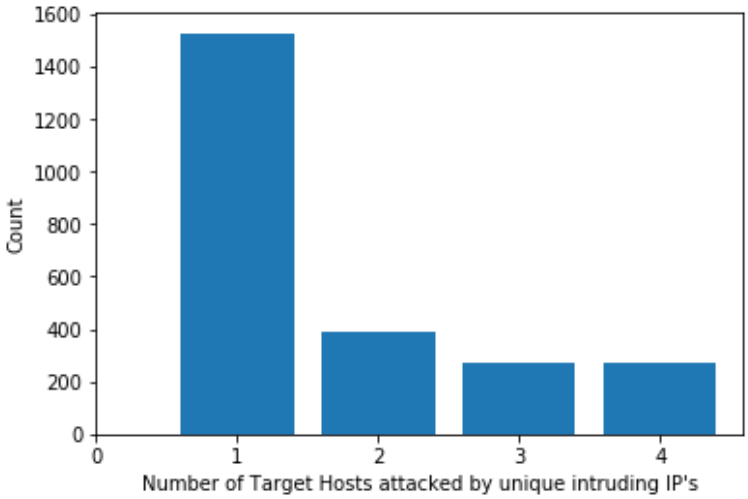


Figure 1: Number of hosts attacked by a unique source IP - Phase1

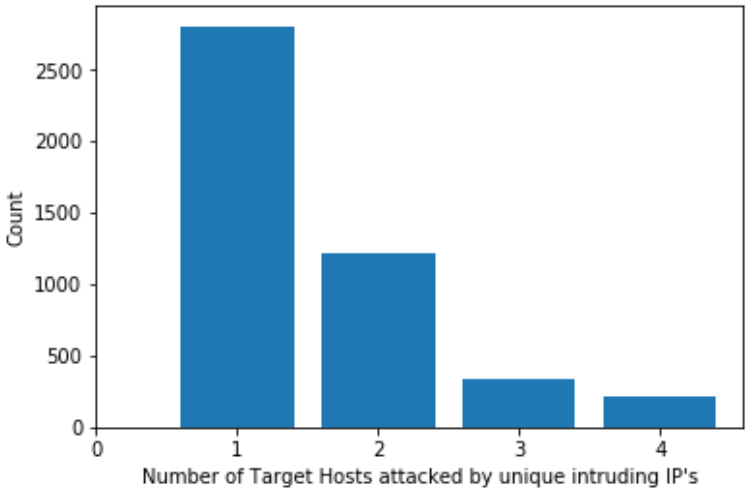


Figure 2: Number of hosts attacked by a unique source IP - Phase2

.3 Top 10 Country - Attack Origination

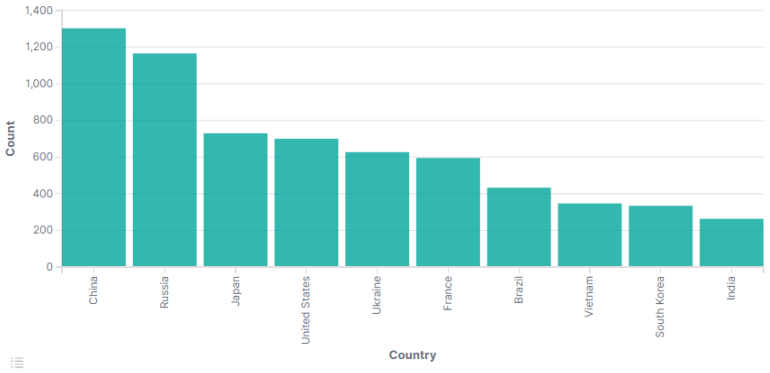


Figure 3: Top 10 countries as source of attacks - Phase 1

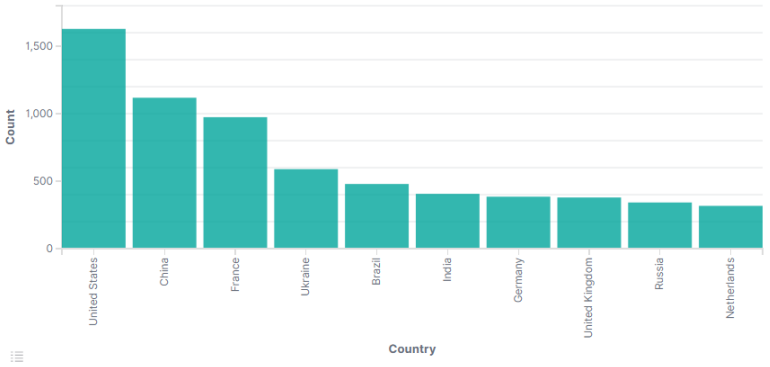


Figure 4: Top 10 countries as source of attacks - Phase 2

.4 Geographic - Attack Origination Distribution

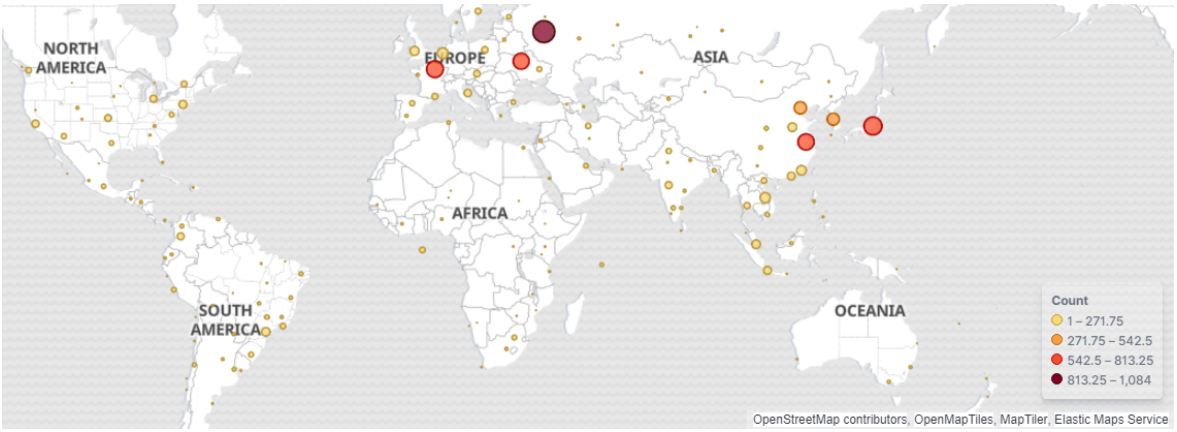


Figure 5: Geographic distribution of the 9663 intrusion attempts observed during Phase 1

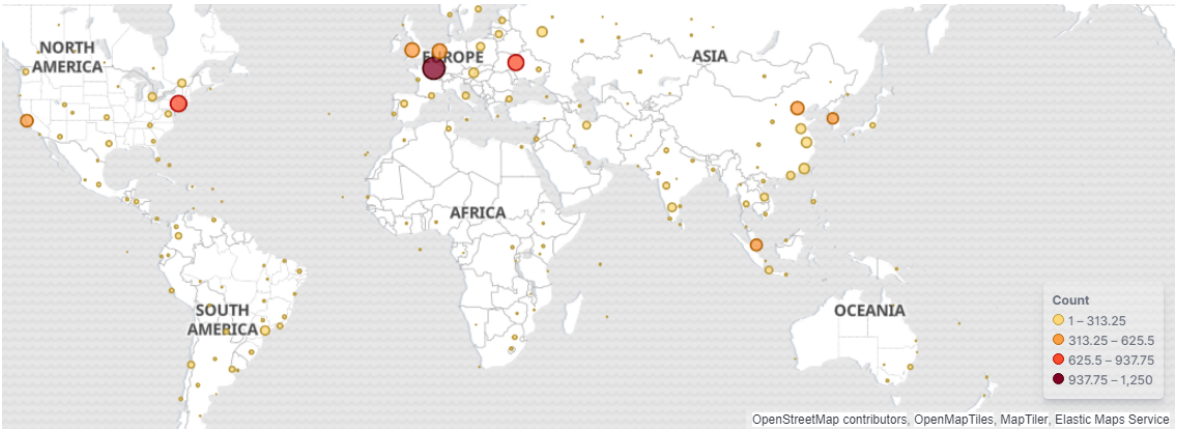


Figure 6: Geographic distribution of the 11526 intrusion attempts observed during Phase 2

.5 Geographic - Unique IP Distribution

Figure 7: Geographic distribution of the 2426 intrusion attempts observed during Phase 1

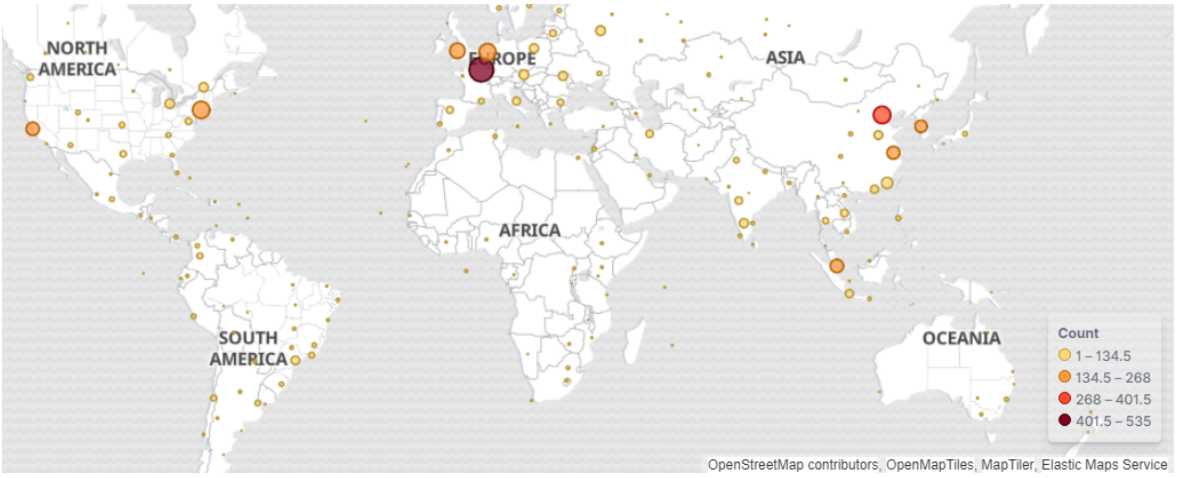


Figure 8: Geographic distribution of the 4526 intrusion attempts observed during Phase 2

.6 Temporal Distribution of Attacks in Source and Target TimeZones

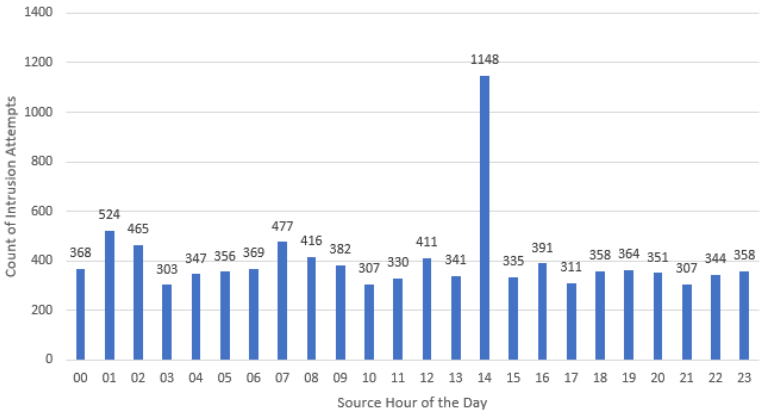


Figure 9: Hourly distribution of attack origination across different countries in their respective source time-zone - Phase 1

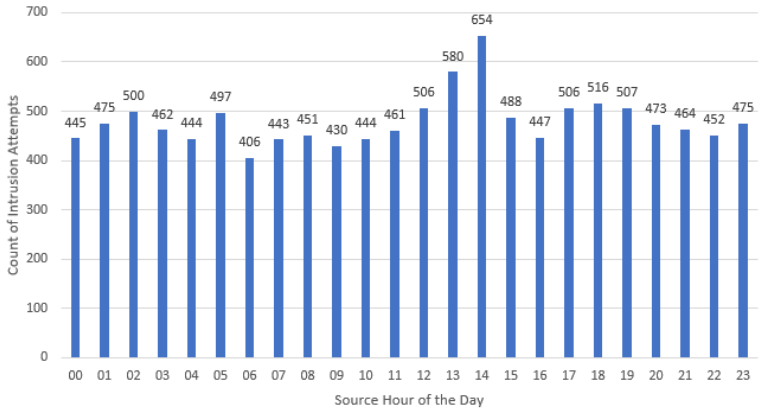


Figure 10: Hourly distribution of attack origination across different countries in their respective source time-zone - Phase 2

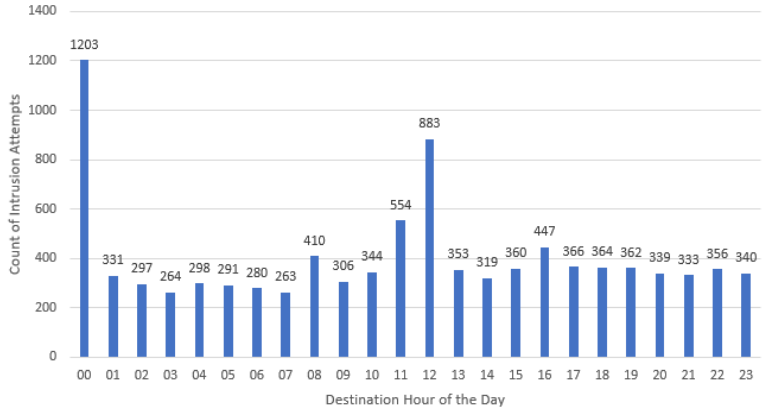


Figure 11: Hourly distribution of attacks in the target (Ireland) time-zone - Phase 1

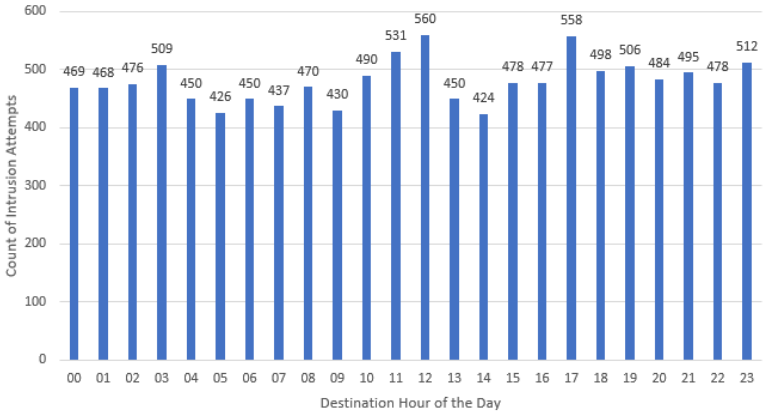


Figure 12: Hourly distribution of attacks in the target (Ireland) time-zone - Phase 2

.7 Top 10 Intrusion IP

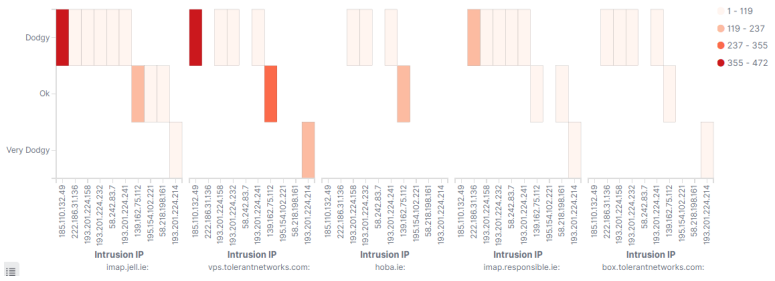


Figure 13: Top 10 intrusion IPs in terms of number of attacks detected, host targeted and the dodgy level calculated - Phase 1

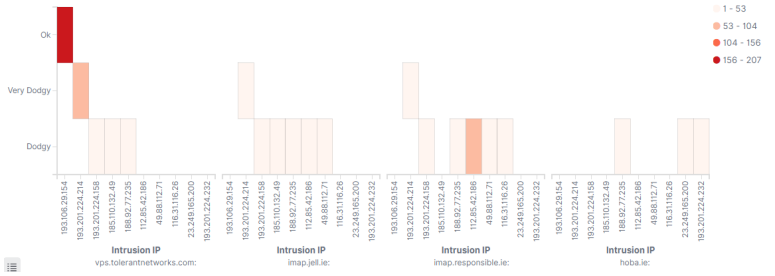


Figure 14: Top 10 intrusion IPs in terms of number of attacks detected, host targeted and the dodgy level calculated - Phase 2

Figure 15: ASN Organization word cloud - Phase 1

Figure 16: ASN Organization word cloud - Phase 2

.8 Services Attacked

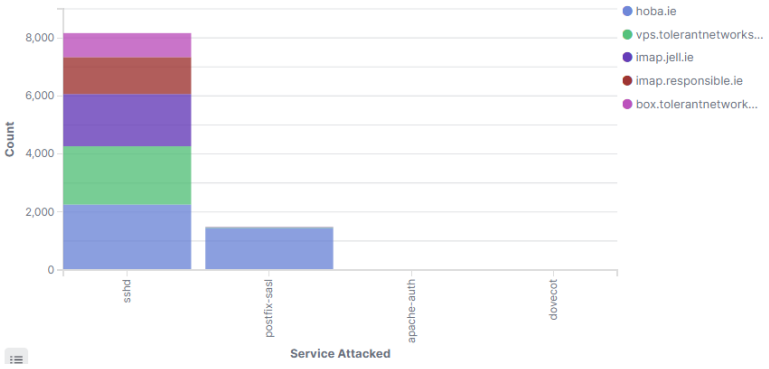


Figure 17: Count of intrusion attempts against remote services running on the VPS - Phase 1

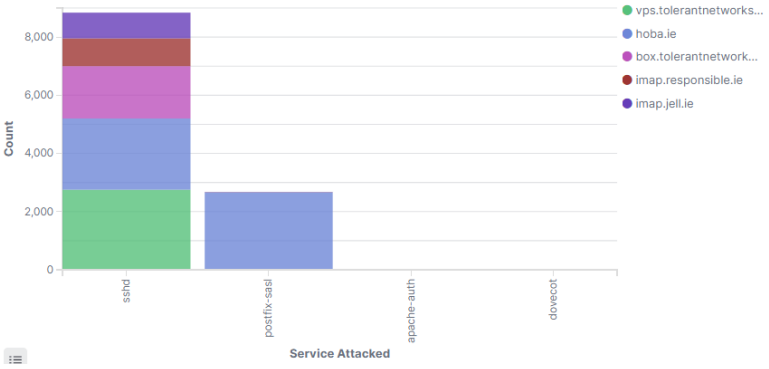


Figure 18: Count of intrusion attempts against remote services running on the VPS - Phase 2

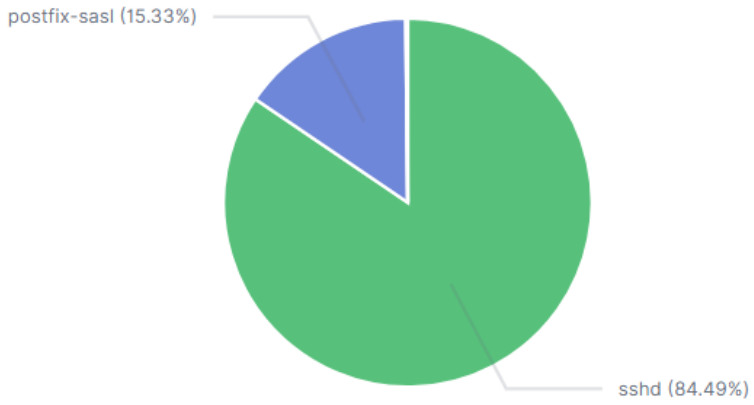


Figure 19: Percentage distribution of remote services attacked - Phase 1

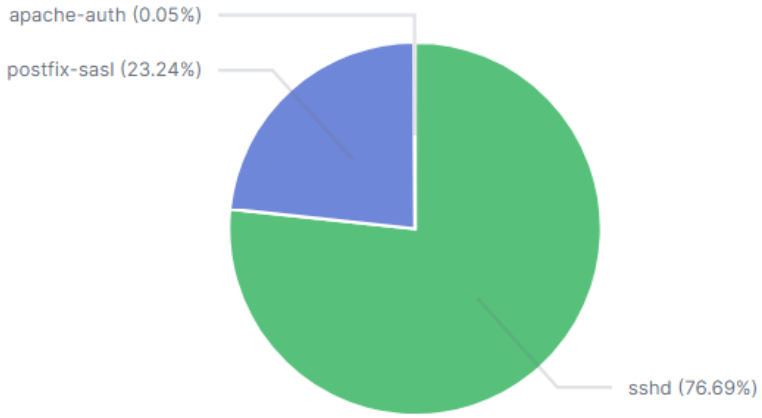


Figure 20: Percentage distribution of remote services attacked - Phase 2

.9 Time Series Prediction Plots

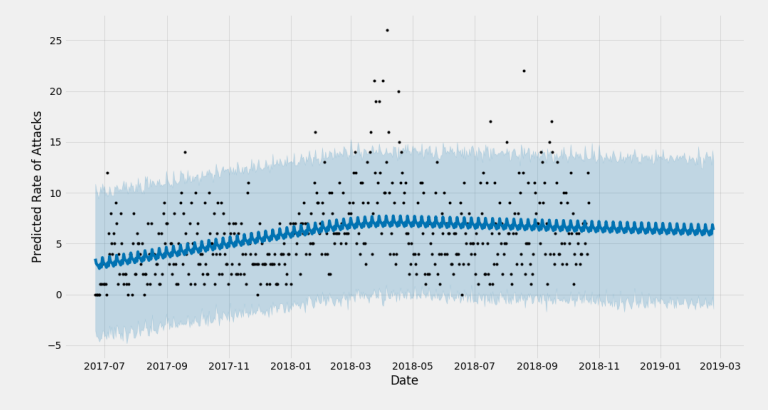


Figure 21: The time-series prediction plot for tolerantnetworks

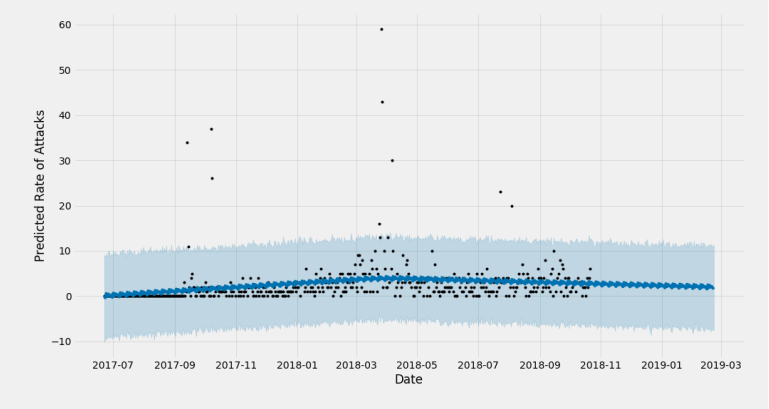


Figure 22: The time-series prediction plot for responsible

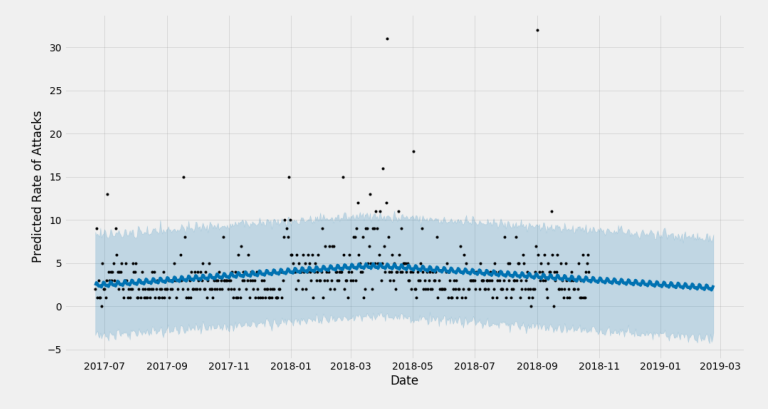


Figure 23: The time-series prediction plot for jell

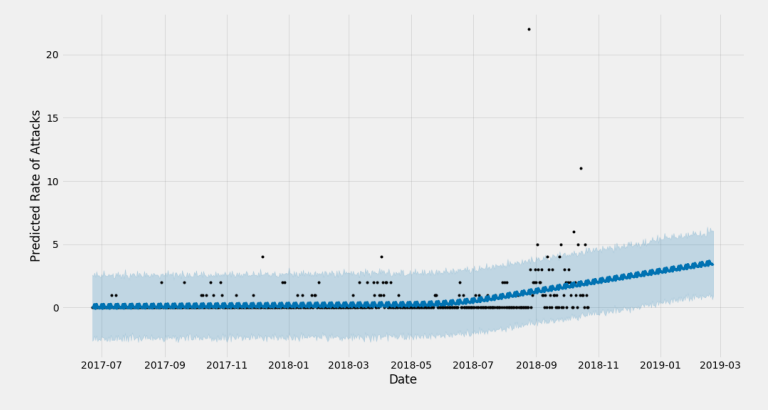


Figure 24: The time-series prediction plot for Africa

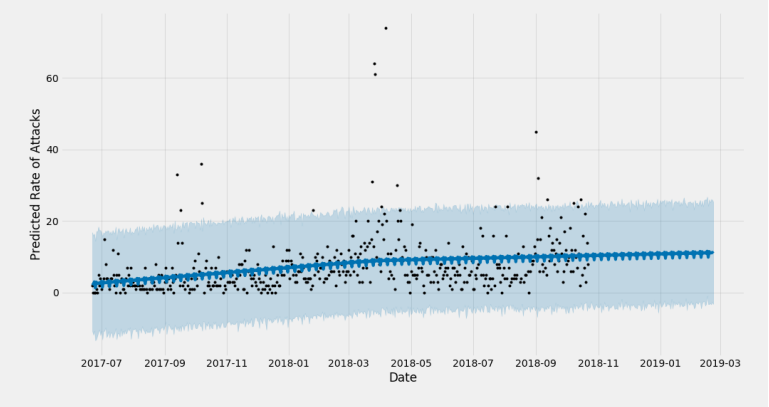


Figure 25: The time-series prediction plot for Asia

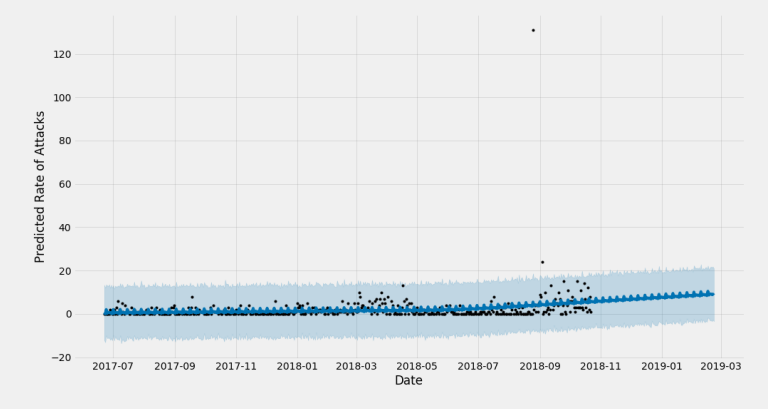


Figure 26: The time-series prediction plot for North America

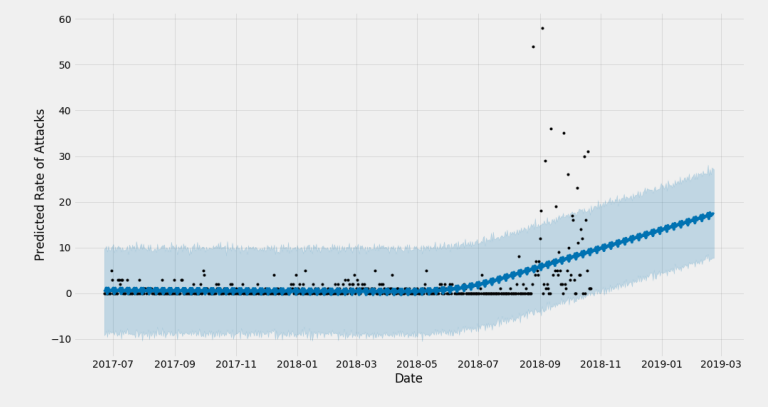


Figure 27: The time-series prediction plot for South America

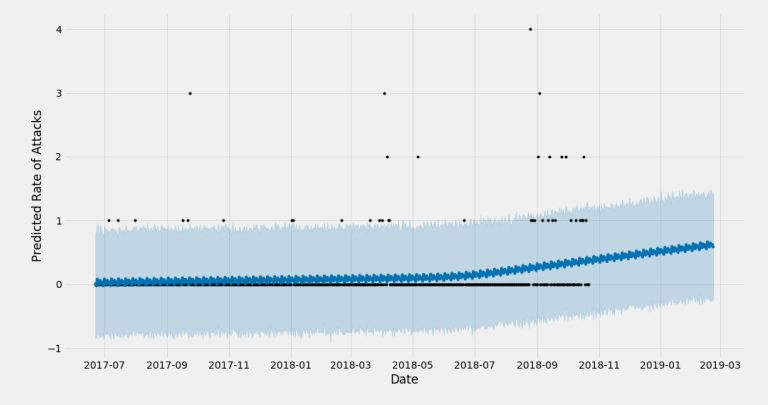


Figure 28: The time-series prediction plot for Oceania