A Framework for Malicious Host Fingerprinting Using Distributed Network Sensors

submitted in fulfillment of

the requirements for the degree of

MASTER OF SCIENCE

of

RHODES UNIVERSITY

Samuel Oswald Hunter

Grahamstown, South Africa

January 2017

Abstract

Numerous software agents exist and are responsible for increasing volumes of malicious traffic that is observed on the Internet today. From a technical perspective the existing techniques for monitoring malicious agents and traffic were not developed to allow for the interrogation of the source of malicious traffic. This interrogation or reconnaissance would be considered active analysis as opposed to existing, mostly passive analysis. Unlike passive analysis, the active techniques are time-sensitive and their results become increasingly inaccurate as time delta between observation and interrogation increases. In addition to this, some studies had shown that the geographic separation of hosts on the Internet have resulted in pockets of different malicious agents and traffic targeting victims. As such it would be important to perform any kind of data collection over various source and in distributed IP address space.

The data gathering and exposure capabilities of sensors such as honeypots and network telescopes were extended through the development of near-realtime Distributed Sensor Network modules that allowed for the near-realtime analysis of malicious traffic from distributed, heterogeneous monitoring sensors. In order to utilise the data exposed by the near-realtime Distributed Sensor Network modules an Automated Reconnaissance Framework was created, this framework was tasked with active and passive information collection and analysis of data in near-realtime and was designed from an adapted Multi Sensor Data Fusion model. The hypothesis was made that if sufficiently different characteristics of a host could be identified; combined they could act as a unique fingerprint for that host, potentially allowing for the re-identification of that host, even if its IP address had changed. To this end the concept of Latency Based Multilateration was introduced, acting as an additional metric for remote host fingerprinting. The vast amount of information gathered by the AR-Framework required the development of visualisation tools which could illustrate this data in near-realtime and also provided various degrees of interaction to accommodate human interpretation of such data. Ultimately the data collected through the application of the near-realtime Distributed Sensor Network and AR-Framework provided a unique perspective of a malicious host demographic. Allowing for new correlations to be drawn between attributes such as common open ports and operating systems, location, and inferred intent of these malicious hosts. The result of which expands our current understanding of malicious hosts on the Internet and enables further research in the area.

Acknowledgments

There exist extraordinary people in this world that guide us to achieve more than we could ever imagine and without whom I would not be where I am today. I would like to thank my family, friends and colleagues who have provided support and guidance while I completed this thesis.

To my parents, Jeannette and Roland George Hunter who have worked tirelessly and selflessly to support me; pushed, guided and loved me.

To my sister, Jeannine Hunter for being the most incredible person I know and who has motivated me to do better at ever turn.

To my supervisor, Prof Barry Irwin whose continued guidance, support and unwavering patience contributed immensely to the completion of this thesis.

To these individuals I am eternally grateful.

This work was undertaken in the Distributed Multimedia CoE at Rhodes University, with financial support from Telkom SA, Tellabs/CORIANT, Easttel, Bright Ideas 39, THRIP and NRF SA (UID 90243). I would also like to thank the department of DPSS from the Council for Scientific and Industrial Research for financial assistance which made this work possible. I would like to acknowledge and thank MaxMind for the use of their GeoIP database and Elitehost for providing a VPS during the course of this research. Finally, I would like to thank SensePost, for their support during the completion of this thesis. The author acknowledge that opinions, findings and conclusions or recommendations expressed here are those of the author and that none of the above mentioned sponsors accept liability whatsoever in this regard.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

AMQP Advanced Message Queuing Protocol

API Application Program Interface

APT Advanced Persistant Threat

AR-Framework Automated Reconnaissance Framework

ARPA Adress and Routing Area

AS Autonomous Systems

ASN Autonomous System Number

AV Anti-Virus Software

BPF Berkley Packet Filtering

C2 Command and Control

CAIDA Center for Applied Internet Data Analysis

CBG Contraint-Based Geolocation

ccTLD country code Top Level Domain

CnC Command and Control

DDoS Distributed Denial of Service

DoS Denial of Service

FBI Federal Bureau of Investigation

FIFO First In First Out

FN False Negative

FP False Positive

FTP File Transfer Protocol

GA Genetic Algorithms

HIDS Host based Intrusion Detection System

HTTP Hypertext Transfer Protocol

IANA Internet Assignment Numbers Authority

IBR Internet Background Radiation

ICMP Internet Control Message Protocol

IDS Intrusion Detection System

IP Internet Protocol

IRC Internet Relay Chat

ISO International Organisation for Standardisation

ISP Internet Service Provider

JSON JavaScript Object Notation

LBM Latency Based Multilateration

MAC Media Access Control

MSSQL Microsoft SQL Server

NAT Network Address Translation

NIC Network Interface Card

NIDS Network based Intrusion Detection System

NTP Network Time Protocol

OS Operating System

OSINT Open Source Intelligence

P2P Peer-to-Peer

PM Potentially Malicious

PPTP Point to Point Tunneling Protocol

RAT Remote Access Tool

RCE Remote Command Execution

RDP Remote Desktop Protocol

rDSN near real-time Distributed Sensor Network

RFC Request for Comments

RoM Result of Malicious

RTT Round-Trip Time

SID Session ID

SMB Server Message Block

SMTP Simple Mail Transfer Protocol

TBG Topology-based Geolocation

TCP Transmission Control Protocol

TLD Top Level Domain

TM Truly Malicious

TN True Negative

TOR The Onion Router

TP True Positive

UCSD University of California San Deigo

UDP User Datagram Protocol

USD United States dollar

VNC Virtual Network Computing

VPN Virtual Private Network

XMPP Extensible Messaging and Presence Protocol

Let's think the unthinkable, let's do the undoable. Let us prepare to grapple with the ineffable itself, and see if we may not eff it after all.

Douglas Adams - Dirk Gently's Holistic Detective Agency

1 Introduction

Modern networks play host to many malicious agents and the traffic they produce. These agents are constantly evolving their methods of attack and their arsenal of tools. The motivation behind their malicious activity often includes monetary gain, political agenda, or a show of dominance. The subsequent effect of this activity on production networks results in a loss of data, intrusion of privacy, denial of service, monetary and often reputation loss. For effective detection and ultimately prevention of these threats, it is important to monitor the activity of these agents. This work extends existing monitoring techniques from Moore et al. (2004); Harder et al. (2006) along with malicious agent characteristics from Pang et al. (2004); Aben (2009); Irwin (2011). The development of real-time Distributed Sensor Network (rDSN) modules allowed for the near real-time analysis of malicious traffic from distributed, heterogeneous monitoring sensors. For the purpose of this research network telescopes and honeypots were used as input sensors. In order to further utilise the data exposed by the developed rDSN modules an Automated Reconnaissance

Framework (AR-Framework) was created, this framework was tasked with both active and passive information collection and analysis of the collected data in near real-time.

The following Section introduces the concept of unsolicited network traffic, as such, providing context and motivation for the development of the rDSN modules and AR-Framework. The research problem statement which follows, describes the lack of knowledge surrounding the characteristics of malicious hosts. Additionally, it outlines the limitations of previous work and the advantages of monitoring and analysing data from these specific sensors in near real-time.

1.1 Background

Unsolicited network traffic destined towards unsuspecting hosts is known as Internet background noise or Internet Background Radiation (IBR) (Wustrow et al., 2010) and accounts for a large proportion of traffic found on the Internet. Back in November of 2010, Ward (2010) estimated IBR traffic to represent 5.5 gigabits of network traffic per second. According to Wustrow et al. (2010), IBR largely consists of reflected Distributed Denial of Service (DDoS) attacks, Internet worm propagation, network scanning, and misconfigured services or hardware (the latter only responsible for a small fraction of traffic). The vast majority of IBR traffic is considered malicious in nature, or at least as a result of malicious activity. The most common technique for observing IBR is through the use of a network telescope also known as a black hole, Internet sink, darkspace, or darknet (Wustrow et al., 2010). A network telescope is considered as the monitoring of a portion of publicly routed IP address space where no legitimate traffic should exist, thus, unused IP address space.

If a /8 network telescope such as the University of California San Deigo (UCSD) telescope operated by the Center for Applied Internet Data Analysis (CAIDA) (CAIDA, 2013), was used to monitor IBR, the average time, according to Moore et al. (2004) to observe at least one packet from a host transmitting to 10 random IP address every second would take 25.6 seconds. A /8 network telescope network telescope monitors a total of $2^{24}(16,777,216)$ IP addresses. The probability of detecting some event (packet) in most cases is proportional to the size of the monitoring network telescope. That said, researchers such as Harder et al. (2006) have shown that results from smaller network telescopes may be extrapolated to represent a subset of global IBR, this will be discussed further in Section 2.1. While this is not ideal as certain local biases may exists, it does motivate for the usefulness of network telescopes of any size towards IBR

monitoring.

Multiple researchers such as Moore et al. (2004); Bailey et al. (2006); Moore et al. (2006); Wustrow et al. (2010) have shown how network telescopes could be successfully employed to observe anomalous behavior, infer DDoS attacks and track self-propagating worms. As such network telescopes are well-suited as a source for discovering IP addresses of malicious hosts, with the exception of traffic resulting from reflected DDoS attacks as this represents replies from hosts under attack (CAIDA, 2013). The Internet has seen an influx of IBR traffic over the last decade and it is believed to be as a result increasing cyber crime. The severity of malicious activity on the Internet was highlighted by Mikko Hypponen when he stated that all of the companies on the Fortune 500 list[1] had been breached by attackers (MilkenInstitute, 2012). Current detection and mitigation techniques have been failing, according to Symantec InternetSymantec (2012b), the estimated cost of cyber crime had exceeding $388 billion in 2011. A large proportion of this traffic was as a result of botnets, CAIDA have found that traffic destined towards unused address space has evolved to include longer-duration, low-intensity events used for establishment and maintenance of botnets (CAIDA, 2013). According to Weber (2007) in 2006, up to 25% of the computers connected to the Internet belonged to a botnet.

It is clear that there exists a significant amount of malicious traffic on the Internet that researchers are able to detect and capture for analysis (Harder et al., 2006; Wustrow et al., 2010). The analysis of IBR is centered around understanding of traffic flows and the detection, distribution, and frequency of events. Examples of such research includes the early detection of a worm outbreak or analysis of a worm outbreak after the fact (Harder et al., 2006; Wustrow et al., 2010). Most IBR analysis would be performed well after the events had occurred (Bailey et al., 2006; Zseby et al., 2014), as such the analysis would be used to better understand and contextualise identified events. This is mostly as a result of the technologies used during the deployment of network telescopes. Team Cymru (2004) supplied an illustrative document detailing the technical deployment of a network telescope. The Cymru website is also periodically updated to provide additional technical insight towards deployment of a network telescope. The use of packet captures show that analysis was initially determined as being deterministic and post-capture. The lack of live or real-time analysis of IBR traffic as supplied by network telescopes, has left time sensitive analysis and certain interesting information simply unobtainable.

[1] http://fortune.com/fortune500/

1.2 Problem Statement

The existing knowledge base of malicious hosts on the Internet has largely been compiled from the analysis and data mining of traffic produced by those hosts and captured by network telescopes and honeypots. Source traffic from network telescopes and honeypots in most cases consists of some form of malicious intent such as self-propagating worms, brute force login attempts, specially crafted exploits, spam, phishing attacks, and more. Information obtained from these hosts often include data such as passive Operating System (OS) fingerprints, targeted service (port), and depending on the capabilities of the monitoring device, the malicious payload or request. Understanding the underlying characteristics that define these hosts and correlating these characteristics with the types of attacks that hosts were responsible for could provide insight into the malicious host demographic on the Internet.

This thesis details the development of a real-time distributed sensor network, AR-Framework, and various visualisation techniques that have been used to contextualise malicious host traffic as well as host characteristics. Malicious traffic was collected through the use of the network telescopes and honeypots, this data was autonomously disseminated and analysed and additional information collected in order to model a malicious host demographic. Geographically distributed sensors would allow for a more holistic approach towards data collection, in so doing it would be possible to assess how different IP Address space around the world experienced malicious activity. During analysis, useful metrics were identified such as correlations between malicious host characteristics, the attacks associated with them, open port combinations, and intent of hosts based on these characteristics.

The ability to enumerate open ports on hosts was made possible through the near real-time exposure of detected hosts. As not all hosts connected to the Internet make use of static IP addresses, it is impossible to guarantee that an IP address observed today would belong to the same host tomorrow. While this behaviour may change with the adoption of IPv6 over IPv4 addresses, due to an increase in the use of statically assigned addresses, many hosts would still be assigned dynamic IP addresses. Near real-time exposure allowed for the enumeration of hosts to begin almost instantaneously after detection, minimising the chance of a host becoming unavailable at a later stage, as was common with traditional off-line analysis of network telescope data from pcap[2] files (Irwin, 2011). An intended outcome of this research was also to explore techniques for host tracking or re-identification of malicious hosts in dynamic IP address space,

[2]http://www.tcpdump.org/

without the need to deploy "call home" code. The value for unique representations of hosts in dynamic IP address space was motivated by fields such as network forensics, botnet research, malicious host analysis, and information warfare. There exists no complete body of work which provides both a method and illustration of data to represents malicious hosts on the Internet, according to their unique and shared characteristics.

1.3 Research Goals

The objective of this work is to undertake an empirical study of malicious hosts on the Internet. This would firstly require a means of detecting malicious traffic on the Internet in order to isolate the sources of such activity. Characteristics of malicious hosts, which could be collected remotely and used to represent the hosts would also need to be identified. The ability to re-identify malicious hosts in dynamic IP address space could also be advantageous to further research and applicable to several real world scenarios. A means of collecting this data would need to be developed and tested and an analysis of this data performed. In order to approach these objectives, the following research goals were identified:

- Develop a means of identifying and collecting malicious host traffic from distributed, heterogeneous source and expose that information in real-time for further research.
- Identify characteristics that could be remotely gathered and used to represent malicious hosts on the Internet.
- Develop and evaluate new metrics that could be gathered from hosts in order to re-identify them in dynamic IP address space.
- Implement a method to interrogate and actively gather information from malicious hosts on the Internet.

1.4 Research Methodology and Scope

The overall purpose of this research was to expand the current knowledge and understanding of malicious hosts on the Internet. While other research has focused on traffic classification and traffic characteristics, this research focuses on the representation of malicious hosts that are the source of IBR traffic. The rDSN modules and AR-Framework that were developed for this

research and used for data collection was constructed in such a way that multiple heterogeneous data sensors may be used concurrently for data collection. It was also important that analysis of malicious hosts were conducted in near real-time, soon after detection and to prove the usefulness of such an approach. Time constraints between detection and analysis were imposed by the dynamic and interconnected but volatile nature of the Internet, a host observed at one moment, may not be reachable at the next. This constraint meant that for detailed analysis of hosts, fingerprinting was required to commence as soon as a host was detected. Through a method of experimentation, new techniques for host identification, reconnaissance, and tracking could be developed and evaluated.

While the Internet has been in a state of transition from an IPv4 to and IPv6 network, only IPv4 traffic was considered for this research as the network telescopes and honeypots at our disposal where configured for IPv4 address space. With reference to the increasing adoption of IPv6, the biggest implication it would have over its predecessor would be an increase in statically assigned IP addresses. A natural evolution as the allocated address space increases from roughly 4 billion to 3.4×10^{38}. There are currently and will for the foreseeable future, still exist a large proportion of dynamically assigned IPv4 addresses. The vast majority of residential and mobile connectivity is achieved through the leasing of dynamic IP addresses from an Internet Service Provider (ISP). As with the adoption of any new technology, backward compatibility is important and as such IPv4 networks will still be supported for many years.

Data was collected over short periods of time, often lasting only a couple of days. For the purpose of this research this was sufficient, as it will be shown in Section 5 that these datasets maintained a consistent correlation with each other. In order to model information gained by the analysis of this data, visualisation techniques were explored, which led to the development of several near real-time capable visualisations. Data collection was focused on attributes that could be used to represent malicious hosts, while their behaviour and intent was considered, it did not constitute a main focus during the analysis provided in later chapters.

A portion of the data collection in this research included port scanning and fingerprinting of malicious hosts. As this could not be performed passively, with the level of detail required, un-solicited traffic was sent to hosts that were considered as malicious or potentially malicious. To this end, Section 2.3 explores and classifies the different forms of traffic that could be detected and Section 2.6 will cover legal considerations with regard to port scanning and fingerprinting.

While much of the data collection and analysis documented in this work was performed at

the end of 2012, the concepts and techniques that have been developed and outlined will still be applicable for the foreseeable future. That is to say, the attributes that have been collected from malicious hosts remain constant while their values may change over time as new technologies emerge.

1.5 List of Publications

The following papers were published during the course of this research.

S.Hunter and B.Irwin. Tartarus: A Honeypot Based Malware tracking and Mitigation Framework. *In proceedings of the 10th International Information Security South Africa Conference*, 2011

S.Hunter, E.Stalmans, B.Irwin and J.Richter. Remote Fingerprinting and Multisensor Data Fusion. *In proceedings of the 11th International Information Security South Africa Conference*, 2012

S.Hunter, E.Stalmans and B.Irwin. An Exploratory Framework for Non-Aggressive Responses to Hostile Network Traffic. *In proceedings of the 4th Workshop on ICT Uses in Warfare and the Safeguarding of Peace* , 2012

E.Stalmans, S.Hunter and B.Irwin. Geo-spatial Autocorrelation as a Metric For The Detection Of Fast-Flux Botnet Domains. *In proceedings of the 11th International Information Security South Africa Conference*, 2012

S.Hunter, E.Stalmans and B.Irwin. Real-time Distributed Malicious Traffic Monitoring For Honey-pots And Network Telescopes. *In proceedings of the 12th International Information Security South Africa Conference*, 2013

1.6 Document Conventions

For the remainder of this document the terms "malicious" and "potentially malicious" will be used interchangeably while referring to the hosts that form the target of this research. In addition to this the terms "data source" and "data sensors" or "sensors" refer to sources of raw data such as network telescopes and honeypots that were used for the detection of malicious host IP addresses. Footnotes were used in this document to provide URLs pertaining to websites, organisations or

software mentioned. This was done to allow readers quick access to the information rather than having to refer to the Reference section of this document.

The convention for illustrating network ports on hosts included the port number and network protocol at the IP level, thus if a web server was running HTTP on the standard port it would be conveyed with 80/tcp. Similarly a standard DNS server configuration would be 53/udp.

1.7 Thesis Structure

The remainder of this work is structured as follows:

- Chapter 2 provides an overview of the malicious agents responsible for traffic observed on sensors such as honeypots and network telescopes. Means of detecting these agents and the traffic they produce are presented and followed by an introduction to event observation probability. IBR traffic was then further defined into three categories to assist in the categorisation of threats; Potentially Malicious, Truly Malicious, and Result of Malicious traffic. Geolocation techniques for host identification and tracking will be discussed, with detailed references to existing latency based techniques. Methods for remote host enumeration and fingerprinting were also introduced. The Chapter concludes with an overview of legal considerations regarding the use of unauthorised port scanning for host enumeration and fingerprinting.

- Chapter 3 covers the concept of remote host fingerprinting in more detail. Four heterogeneous categories of remote fingerprinting are conceptualised and their application towards uniquely host representation is presented. These categories included logical, physical, associative and behavioural attributes that could be remotely gathered from hosts. The idea of Fast-Flux botnet association as a supporting metric for host fingerprinting is explored. The concept of Latency Based Multilateration (LBM) fingerprinting is then introduced, documenting the assumptions and requirements along with the techniques of measurement and comparison between measurements.

- Chapter 4 provides and overview of the AR-Framework design and implementation. The design of the AR-Framework was divided into four core components each of which are covered in a separate section in Chapter 4. Conceptual modeling of the AR-Framework was accomplished by expanding on the concept of multi-sensor data fusion, originally used for mili-

tary command and control infrastructure. Three visualisation tools are then demonstrated. These tools where developed to visualise and aggregate data from the AR-Framework.

- Chapter 5 provides and discusses results obtained from the analysis of AR-Framework data as detected from the different data sensors. The chapter starts with an overview of the datasets collected which then follows a discussion on host reachability and the feasibility of latency based fingerprinting. The geopolitical distribution of malicious hosts are then presented. This follows a discussion of targeted ports across the different data sensors and the observation of traffic resulting from the Carna Botnet (Shukla, 2015). The operating systems of malicious hosts are presented and a comparison is made between the proportion of systems (servers) seen versus devices (modems). The chapter concludes with a case study of open ports on malicious host and the correlation between targeted ports and exposed ports on malicious hosts.

- Chapter 6 concludes with a summary of research goals and findings. Considerations for future work are made and concluding remarks provided.

It was from the artists and poets that the pertinent answers came, and I know that panic would have broken loose had they been able to compare notes.

H.P. Lovecraft - The Call of Cthulhu

2

Related Work

This chapter provides background and presents existing research to the topic of malicious software agent detection and host fingerprinting. It starts by examining different types of malicious agents found on the Internet and the different techniques available for detecting these agents. Through the investigation of these approaches it was possible to identify the technologies that would be best suited toward detecting malicious hosts on the Internet. Techniques used for fingerprinting and service enumeration were pivotal during data collection and as such required investigation. An overview of this chapter has been provided below.

Chapter Overview

- Section 2.1 identifies the different types of malicious agents found on the Internet. Their characteristics and purpose are discussed, providing the motivation for this work. The agents included in this discussion are viruses, trojan horses, worms, spyware/adware, and

human adversaries. A clear understanding of the malicious agents and the type of traffic they produce enables a determination of the best methods for detection.

- Following from the discussion of malicious agents, the means of detecting these agents are introduced in Section 2.2. Three existing techniques are covered, namely, intrusion detection systems, honeypots, and network telescopes. The section concludes with a discussion based on event observation probability with regards observable IPv4 address space.

- Different types of malicious traffic are discussed in Section 2.1. Internet Background Radiation (IBR) is introduced and then expanded upon by defining three sub categories of IBR; Potentially Malicious, Truly Malicious, and Result of Malicious traffic. Examples of the previously identified malicious software agents are given for each of these categories.

- The idiosyncrasies of geolocating sources of malicious activity on the Internet are introduced in Section 2.4. This includes a review of both repository-based and measurement-based approaches for geolocation. The measurement-based approaches formed a basis for the Latency Based Multilateration technique that was developed during this research.

- Methods of host fingerprinting, including Operating System (OS) identification and service enumeration are discussed in Section 2.5. The discussion includes how existing techniques are applied and form a basis for the techniques used in the AR-Framework which was developed in this research.

- This chapter concludes in Section 2.6 with an overview of potential legal implications regarding the use of fingerprinting techniques, without explicit consent from the host being targeted.

2.1 Malicious Software Agents

In order to detect malicious activity, the malicious software agents actions need to be identified and the traffic they produce understood. This includes identifying their means of communication and propagation, their goals, and any fingerprints available for identification. Once these attributes have been identified, it becomes possible to construct tools and techniques to identify such agents. Both open source (Zhou and Jiang, 2012) and commercial applications (Morales et al., 2010) exist that can be used to identify malicious agents and their activity (or network traffic) and in some cases even their intent. The advantages and disadvantages of existing techniques will be provided.

The classes of malicious agents identified below have similar characteristics and in some cases one malicious agent may be classified into more than one category due to the shared traits or common functionality. Part of this section identifies the attributes or characteristics of malicious agents that could be used to identify them; Section 2.2 will expand on this by identifying some of the techniques available for malicious agent detection. It is possible to define the two main states during which a malicious agent could be identified; identification at rest (that is once the malicious agent is on a system) and identification in transit. The latter identifies agents through inspection of network traffic as opposed to inspection of a static binary, in memory or execution behaviour on a system.

With the exception of the human adversary, all of these malicious agents may be classified as Malware. The most capable malicious agent would be the human adversary who could be presented as a malicious agent and controller for all other malicious agents, yet, the human adversary quite often has other more specific goals in mind and as such was categorised as its own unique malicious entity.

2.1.1 Virus

The term "computer virus" was first introduced in 1984 by the mathematician Dr. Fredric Cohen. According to Szor (2005) the term was introduced based on a recommendation of Cohen's adviser, Professor Leonard Adleman, who had chosen it from science fiction novels. Cohen (1994) provided an informal definition of a computer virus as "a program that is able to infect other programs by modifying them to include a possibly evolved copy of itself". A virus may also be defined as malicious code embedded in a legitimate application. This definition was provided by Blunden (2013) and was expanded by Harris et al. (2008) stating that in order for a virus to infect a host, the legitimate application within which it was embedded requires human interaction and would only infect a system once it has been executed by a user. This definition differs slightly from that of Symantec (2009) who define a virus as "A small program written to alter the way a computer operates, without the permission or knowledge of the user." Furthermore, Symantec state that a virus must meet the following two criteria:

- It must execute itself, for example by placing its own code in the path of execution of another program.
- It must replicate itself, through either infecting another program or replacing an executable

with that of itself.

Viruses range in the threat they pose and ease of detection. In most cases a virus would create an undesirable effect on a system, as such they may be seen as a hindrance or as a form of DoS attack. That said, there are multiple definitions of viruses and the term "computer virus" is often used to group several malicious agents under one definition. For the purpose of this research the definitions provided by Cohen (1994); Harris et al. (2008); Blunden (2013) will be used and other malicious agents such as trojan horses, worms, spyware/Adware, and human adversaries will be placed in their own categories. According to Henchiri and Japkowicz (2006) traditional means of virus detection heavily relied on signature-based heuristics, however, according to Nachenberg (1997) newer, more advanced polymorphic viruses require smarter techniques such as generic decryption and dynamic analysis of the binary at runtime. As viruses do not spread by exploiting a system remotely, they are in most cases detected and identified at rest.

This does not mean that they cannot be detected in transit. Stolfo et al. (2001) explained how active traffic inspection such as performed by an Intrusion Detection System (IDS) could detect malicious various signatures in transit (such as downloading infected media). Inspection of application data in transit could also be circumvented easily if the traffic is encrypted through means such as HTTPS. Some viruses have been advanced with techniques to provide stealth and make them harder to detect once on a system. According to Nachenberg (1997) polymorphic viruses are able to mutate themselves during infection making it difficult for signature-based detection. Viruses spread through deception, masqueraded within existing applications or as a legitimate application itself. Blunden (2013) and Nachenberg (1997) explain that traditionally viruses would spread via contaminated removable storage media such as floppy disks, diskettes, and USB drives. However, with the increase of Internet bandwidth availability, viruses have also been spread through new mediums such as email attachments. According to Cova et al. (2010) drive-by downloads have formed a new delivery vector for viruses through the use of JavaScrypt. Torrents used to share peer-to-peer (P2P) media has also been used as a large scale attack vector, Ramachandran and Sikdar (2006) modeled Malware propagation through Gnutella[1] type P2P networks. As viruses offer a localised threat to the infected hosts and do not spread through remote exploitation of services, but rather through user interaction, they do not form an integral part of this research.

[1] http://rfc-gnutella.sourceforge.net/

2.1.2 Trojan Horses, Remote Access Tools, and Bots

A trojan horse may be considered a malicious application that acts as a backdoor into a computer allowing an attacker to remotely execute commands, download files, and perform other malicious actions on the affected host. Symantec (2009) defined a trojan horse as a file that claims to be something desirable but instead performs malicious actions. The differentiation Symantec (2009) make between a virus and a trojan horse is that trojans do not replicate themselves as viruses do. As such a trojan is generally defined as a non-self replicating malicious application that has been embedded into a legitimate application.

A remote access tool (commonly abbreviated as a RAT) is the portion of code bundled within a trojan horse that provides the remote administration functionality but may also be a stand alone executable functioning solely as a RAT. The RAT would normally connect back to a host under an attacker's control and allow the attacker to execute commands remotely on the infected system. The functionality of connecting back to a command and control server and waiting for instructions has lead to this malicious software also being known as a "bot". Infected hosts are collectively known as a botnet. Much like viruses they are often embedded in existing applications or media and distributed in the same manner. In a recent example of cyber crime, criminals spread malicious links using the *#JeSuisCharlie* Twitter[2] hashtag in order to distribute the DarkComet RAT through social media (Paganini, 2015).

Symantec (2009) define the properties of trojan horses, RATs, and bots (and worms) as "*blended threats*". They state that these threats combine the characteristics of viruses, worms, trojan horses, and other malicious code (RATs and bots) with the ability to exploit vulnerabilities for propagation and produce threats such as Denial of Service (DoS) attacks, spam emails, and remote administrative access. As such, these malicious agents could be seen as assuming many different forms, it is not possible to define them in a highly detailed manner as their attributes and functionality vary greatly between different implementations.

Zeus may be used as an example of a powerful trojan. Binsalleeh et al. (2010) notes that Zeus had infected over 3.6 million computers in the United States alone. According to Ligh et al. (2010) Zeus had also been known as Zbot, PRG, Ntos, and Wsnpoem. These are names associated with the particular malware by different researchers and Anti-Virus (AV) companies. It is commonly found that a single piece of malware or malware family are known under multiple names, as the authors of these malicious tools often wish to remain anonymous, they often do

[2]http://www.twitter.com/

not assign a public name to the malware. Instead, Anti-Virus companies and researchers assign names to threats and as such multiple names may be given and with no consistent approach.

DarkComet is an example of a RAT that provides extensive control over infected victims through an easy to use interface. An example of the several components of the DarkComet RAT is shown in Figure 2.1.

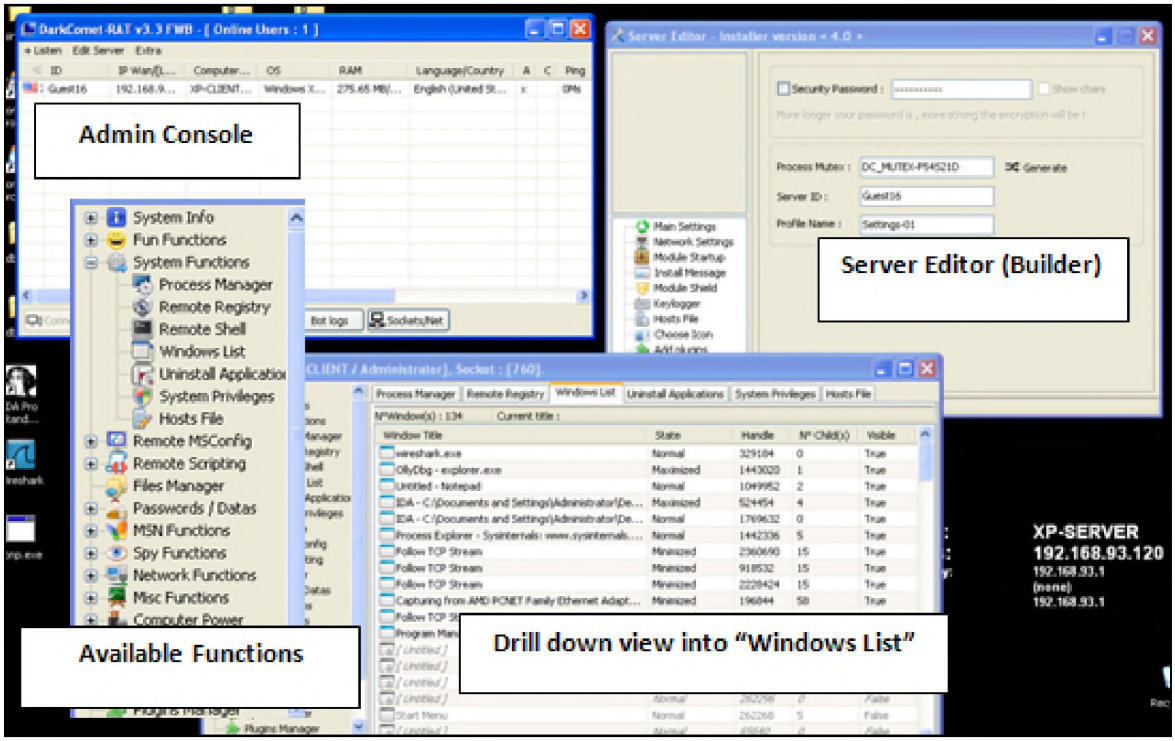

Figure 2.1: Example of a DarkComet interface showing admin console, functionality, and server editor (Paganini, 2015)

Another example of a RAT could be a Metasploit[3] Meterpreter payload. This payload could be delivered with an exploit when compromising a host or created as a standalone executable and then uploaded and executed on a remote machine. Much like viruses; trojan horses, RATs, or bots could be identified by inspection at rest, but also during transit, as there exists network traffic through communication with a command and control (C2 or CnC). Furthermore, researchers have been able to remotely identify CnC servers through open ports awaiting connections from bots and through reverse engineering malware to determine where they would connect.

[3]http://www.metasploit.com/

2.1.3 Worms

Worms are self propagating viruses or trojans over a network. But unlike viruses or trojans, worms do not require interaction from a user in order to execute and spread, which allows them to propagate by themselves in order to infect additional systems, typically through automated scanning and exploitation. It was shown by Bailey et al. (2005a) that worms spread by exploiting vulnerabilities in the Operating System of a targeted host or in application level software. Symantec (2009) defined worms as programs that replicate themselves from system to system without the use of a host file. As such the main difference between worms, viruses, and trojans would be their method of propagation. That said, worms may still employ similar means of propagation as viruses or trojan horses. Symantec (2009) have found that Word or Excel documents have been used to spread worms through embedded macros. As a result of their autonomous capabilities to spread, Luo et al. (2009) noted that worms have the ability to spread at a exponential rate. Which according to Wang et al. (2014), has in the past had devastating effects on network infrastructure such as creating Denial of Service conditions. As worms could be seen as a similar malicious software agent to viruses and trojans with the exception of a self propagating characteristic, they may be identified at rest or in transit. From the perspective of transit-based identification, signatures would be required to identify either unique attributes from worm scanning, or exploitation attempts.

An interesting characteristic of worms was observed by Weaver et al. (2003), whereby worm propagation strategies tended to be biased towards local addresses. An example of this was CodeRed's target scanning, Roman Danyliw (2002) observed that 50% of the time an address with the same first two octets would be chosen, 25% of the time an address with the same first octet, and the last 25% of the time a random address would be chosen . Regarding a worms efficiency to propagate, it stands to reason that due to smaller network distances (latency) it should theoretically spread much faster. Common administrative practices across networks maintained by the same organisations would likely also yield a higher success rate with exploitation attempts by locally biased worms, however, if the administrative practices were good, this would have an adverse effect on its propagation.

2.1.4 Spyware and Adware

Adware and spyware represent a type of software that is often installed onto a host by a user without their knowledge of its function. Once installed and running, adware would display adver-

tisements to a users (Spiegel, 2013), in severe cases, re-rewriting HTML displayed in a browser to insert links to advertisements. Spyware functions with a similar end goal as adware, which is marketing. Spyware would report a user's usage patterns of software or browsing to the 3rd party responsible for running the spyware (Shams et al., 2011). In most cases spyware is not malicious, however, in certain cases key-loggers may be found in spyware, allowing an attacker to capture keystrokes which could lead to the compromise of user credentials, banking fraud, and even identity theft. Spyware and adware may be identified at rest and in transit, as both require external communications to achieve their goal. Furthermore, spyware or adware are often blended with RATs, with many RATs allowing for the installation of spyware, adware, or exposing similar functionality themselves.

2.1.5 Human Adversary

The most interesting and dangerous of malicious agents is the human adversary. While a human adversary would ultimately be responsible for the creation and management of viruses, trojan horses, spyware, and adware, they themselves often go on the offensive. The level of complexity and sophistication of recent malware such as Stuxnet, Duqu, Flame, and Red October have shown a paradigm shift from individuals or small teams to well funded operations (Virvilis and Gritzalis, 2013). Virvilis and Gritzalis (2013) describes how the goal of human adversaries could range from running a botnet to industrial and corporate espionage, and even as force multiplier for traditional warfare and espionage. Combining the use of sophisticated malware and exploits, along with extensive technical expertise all directed towards cyber espionage has seen the advent of of the term "Advanced Persistent Threats" (APT). Mandiant[4], an American cyber security firm released an intelligence report in February of 2013 directly implicating China in cyber espionage targeting at least 141 organisations (FireEye, 2013).

A different type of adversary is presented by Kelly (2012), who notes how the last decade has seen the advent of hacktivism, a form of political and corporate activism sprung forth from skilled human attackers in the digital domain. The attack life-cycle for a human adversary often starts with footprinting, which leads to a triad of possible attack vectors such as host enumeration and exploitation, bruteforce attacks, and social engineering attacks targeting people instead of infrastructure. With the initial goal of attaining a foothold into a target network through a command interface and ultimately access to confidential data, destruction of such data or service

[4]https://www.fireeye.com/

or modification thereof. While human adversaries may be very hard to detect, they to could be detected at rest (malware) and in transit (scanning and exploitation). Its possible that an attacker may leave tools behind or not clear all of their activity in logs or the registry (in the case of Windows-based machines). With regards to identification in transit, the attack life-cycle may leave plenty of evidence behind or ring alarm bells during the attack.

2.2 Means of Detection

The solutions deployed in order to detect malicious threats largely depend on the form of threat being detected. These threats could be broadly defined based on their state within a system, be this system a single physical machine or a network of machines. When considering the state of a threat, it could either be at rest or in transit within the system. This definition has been largely simplified as in truth a threat could be both at rest and performing malicious activities in transit, however, some formal definition is required to adequately discuss the various means of detection.

When defining a threat as being in a state of rest, it is expected that the malicious application is stored either in memory or on media storage such as a hard disk. If the threat is seen as in a state of transit, it may either be transported over a network, or the transit state could represent an action being performed over the network, such as port scanning or exploiting a vulnerability. While defining a threat as being within one of these two states, it should be known that threats could assume multiple states during their lifecycle, similarly multiple heterogeneous means of detection could be used to identify a threat. As such certain means of detection are more capable at detection, depending on this defined "state" of a threat. The following section explores the uses of Intrusion Detection Systems and Anti-Virus software, Honeypots, and Network Telescopes as means of detection.

2.2.1 Intrusion Detection Systems and Anti-Virus Software

Intrusion Detection Systems (IDS) are often placed on the forefront of a network. They provide live analysis of network traffic and trigger events when malicious activity is detected. Bass (2000) describes the primary purpose of IDSs as detection of activities that "compromise the confidentiality, integrity, availability" or attempt to "bypass the security mechanisms of a computer or network".

The fundamental difference between AV software and IDSs would be situational placement of

these different solutions. Where AV software would normally be installed on workstations and servers in order to detect threats at rest, IDSs are often placed at choke points for traffic on a network, as shown in Figure 2.2. This allows IDSs to detect threats in a state of transit. IDS solutions placed on a network have also been referred to as Network Intrusion Detection Systems (NIDS) (Koc et al., 2012); due to the evolution of IDS systems, their application and placement variations have increased. For example, Host-based Intrusion Detection Systems (HIDS) (Wagner and Soto, 2002) have also been used to detect intrusions at a host level as apposed to a network level. Applications of IDS have also exceeded local networks or hosts and have now also been deployed on cloud-based infrastructure (Modi et al., 2013).

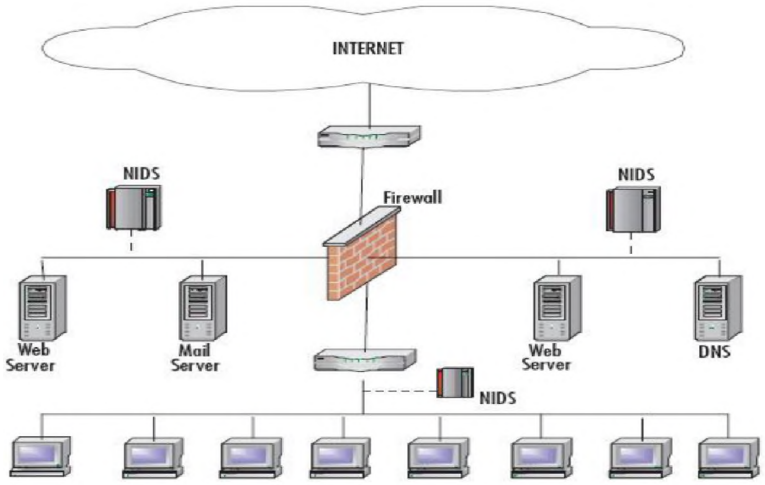

Figure 2.2: Network-based IDS placement (Maltainfosec, 2012)

Similar to AV software, one of the basic methods used by IDSs to detect malicious activity is known as signature-based detection, by inspecting traffic and matching the traffic to known "bad" signatures. Another method used by IDSs is to inspect traffic patterns and detect deviations from historically normal traffic patterns, this is known as anomaly detection and can be used to detect attackers or worms by analysing their fingerprints (Singh et al., 2004). Detection mechanisms for IDSs for various types of intrusions have been implemented successfully using several Genetic Algorithms (GAs) and Genetic Programming approaches for different scenarios. In several cases GAs where used for deriving classification rules (Chittur, 2001; Li, 2004; Lu and Traore, 2004). Hoque et al. (2012) summarised how GAs have been applied to select the required features and to determine the optimal and minimal parameters of some core functions in which different AI methods were used to derive acquisition of rules (Bridges and Vaughn, 2000; Gomez and Dasgupta,

2002; Middlemiss and Dick, 2003).

Most forms of detection have a similiar downside, which is often with the resources required and the effect of congestion or increased latency caused on a network by deploying an IDS. Signature-based detection requires the comparison of traffic which could negatively influence the performance on large networks, especially as signature databases grow over time. As a result, IDSs only detect malicious activity and do not attempt to prevent it. When prevention is required, IDSs are placed alongside or replaced by Intrusion Prevention Systems. These systems make use of the detection capabilities and complement the system by isolating or blocking threats. This, however, comes at an additional cost of resource consumption and often more strain on a network.

2.2.2 Honeypots

Defining the term "honeypot" is challenging due to the vast number of configurations, technologies, techniques, and reasons to deploy them. They also serve many different purposes ranging from detection (Provos and Holz, 2007) to prevention (Hunter et al., 2012a), while also including advanced information gathering capabilities (Dionaea Project, 2012). Honeypots have previously been described as a closely monitored network decoy, used to distract adversary's from valuable machines, provide early warning about new attack and exploitation trends, or provide in-depth examination of an adversaries actions during and after exploitation of a honeypot (Provos, 2004). In a general sense, Spitzner (2003) defined a honeypot as a "security resource whose value lies in being probed, attacker, or compromised".

Honeypots are very similar in nature to network telescopes. They are, however, at the other end of the spectrum with regards to capturing and interacting with nefarious traffic. This is because honeypots normally emulate a vulnerable service or host in order to elicit malicious interaction. Thus acting as a decoy vulnerable system and soliciting bi-directional interaction with malicious agents. Honeypots have been deployed to capture malware (Provos, 2004; Provos and Holz, 2007; Dionaea Project, 2012), monitor vulnerability exploitation (Sochor and Zuzcak, 2014), and provide active network defense (Hunter and Irwin, 2011; Hunter et al., 2012a). Honeypots may be implemented to emulate a known service, some even have the ability to emulate unknown protocols, such as the Dionaea[5] with the NFQ module enabled (Dionaea Project, 2012), which mirrors a protocol in a sense to replay it to the attacker. The Kippo[6] honeypot emulates an SSH[7]

[5]http://dionaea.carnivore.it/
[6]https://github.com/desaster/kippo
[7]http://www.openssh.org/

service with authentication credentials, capturing the keystrokes of attackers and any files they attempt to download. It has also been suggested that honeypots be used in an offensive manner to strike-back at attackers or provide false information that could compromise an attacker or lead them astray during their attack methodology (van der Walt, 2004; Hunter and Irwin, 2011; Hunter et al., 2012a).

Honeypots could be deployed using a single address, such as Dionaea honeypot (Dionaea Project, 2012) or easily function over a range of addresses (Provos, 2003), an example of multiple honeypots over a range of IP addresses is known as a honeynet and could be deployed using a tool such as honeyd[8](Provos, 2003). Some of the differences between honeypots and a network telescope include the increased resource requirements of honeypots and the methods of capturing malicious traffic. The active network telescope implemented by IMS (Bailey et al., 2005a) starts to bridge the gab between network telescopes and honeypots, if only slightly through its ability to complete handshakes. As such honeypots interact in greater detail with threats (such as worms) and are thus more resource intensive and as a result need to be sufficiently hardened to avoid the compromise or disruption (Krawetz, 2004) of the actual honeypot.

In summary, honeypots are able to characterise attacks against services in greater detail while also granting them the ability to capture malware and vulnerability exploitation that requires asynchronous communication as opposed to a malicious agent such as the Microsoft Structured Query Language (MSSQL) Slammer worm that propagates with a single UDP packet which contains both the exploit and payload (Moore et al., 2003).

2.2.3 Network Telescopes

A network telescope has been described by Irwin (2011) as a passive sensor system that collects incoming traffic, sometimes termed "radiation", from the Internet. Different configurations exist for network telescopes, some respond to incoming traffic (Active Telescopes) such as the IMS which makes use of a lightweight responder (Bailey et al., 2005a) while other capture all traffic forwarded to them without any response (Passive Telescopes) (Hunter, 2010; Irwin, 2011). Passive network telescopes have also been referred to as a Darknet, Blackhole, or Sink (Irwin, 2011). Note that the term Darknet has more recently become associated with The Onion Router (TOR) network (Dingledine et al., 2004), representing an "overlay" network that can only be accessed through software such as TOR. A more detailed explanation of active and passive traffic types

[8]http://www.honeyd.org/

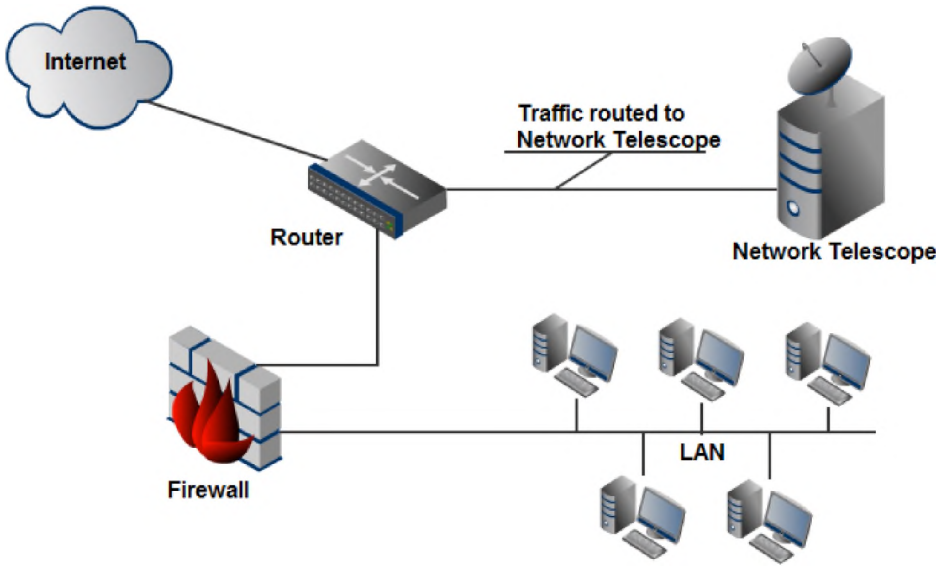

Figure 2.3: A basic network telescope configuration

will be given in Section 2.3. The basic construction of a network telescope would have a server for capturing incoming traffic and a router to forward traffic from public, unused IP address space. An example of a basic network telescope deployment is given in Figure 2.3.

A network telescope's ability to capture traffic destined for unused address space results in it receiving very little, if any, legitimate traffic. Legitimate traffic refers to traffic that is not trying to actively exploit a service, discover a vulnerability, or disrupt a service. The two exceptions here are traffic caused by misconfigured hardware and certain types of DDoS attacks (Moore et al., 2006). An explanation for misconfigured traffic is given by Irwin (2011) as an error possibly occurring when entering the destination, either at the end-point system or through a Network Address Translation (NAT) Gateway. With regards to the DDoS traffic, network telescopes, unless targeted, do not receive traffic from DDoS attacks, but rather the "flooding backscatter" (Pang et al., 2004) received from victims of certain types of DDoS attacks. The traffic captured by the network telescope is thus highly suitable source for providing information regarding events such as DoS attacks (Moore et al., 2006) and the automated propagation of Internet-based worms and viruses (Harder et al., 2006). When considering the traffic observed by a network telescope other researchers refer to the term "backscatter" (Bailey et al., 2006; Harder et al., 2006; Moore et al., 2006) which implies residual traffic observed from other hosts that may have been the target of DDoS attacks and are responding to spoofed source addresses. Backscatter has also

been more broadly defined as traffic observed as a result of activities that caused reflection from the originating host (Irwin, 2011). Other traffic such as network scanning from worms and malicious users also amount to backscatter while a very small portion of backscatter is the result of miss-configured hardware (Bailey et al., 2006).

Even though malicious traffic such as DDoS and worm propagation are globally scoped, data from the IMS indicate widely different trends between separate network telescopes (Harder et al., 2006). These differences were noted across three dimensions namely over all protocols and services, a specific protocol and port, and lastly signatures of known worms. Other work by Shinoda et al. (2005) shared this view that multiple points of monitoring are required coupled with a collective interpretation to provide a more comprehensive view of malicious network traffic. Thus observing traffic from only a single point would provide very little, if any, information about the background activities (Shinoda et al., 2005) as a whole.

There is, however, still important information that could be learned from a single vantage point. Harder et al. (2006) showed how a small /24 telescope was used to identify and distinguish between port scans, host scans, and DDoS attacks. While the results found in Harder et al. (2006) might not correlate strongly with global traffic, their findings were still useful in understanding a subset of current threats.

2.2.4 Event Observation Probability

The amount of traffic a network telescope observes is proportional to the size of the address space monitored by it. The Center for Applied Internet Data Analysis (CAIDA) released "*Network Telescopes: Technical Report*" (Moore et al., 2004) which refers to this as being analogous to that of a astronomical telescope, the larger the telescope size (fraction of address space or telescope aperture), the more basic data becomes available for processing. The size of the address block or "aperture" thus also influences the probability of the network telescope observing a given event. When considering Internet Protocol version 4 (IPv4) address space, which allows a host the assignment of a 32 bit address, there are a total of 2^{32} allocatable IP addresses. Address blocks are commonly assigned according to the number of leading bits that uniquely identify that address, for example a /8 address block would denote a range containing 2^{24} addresses which all share the first 8 bits of their IP address. From this, a /32 would denote a single IP address, thus the probability of a network telescope monitoring a chosen host in IPv4 address space can be determined by $p(x) = 1/2^x$ where p is the probability of monitoring the host and x is the

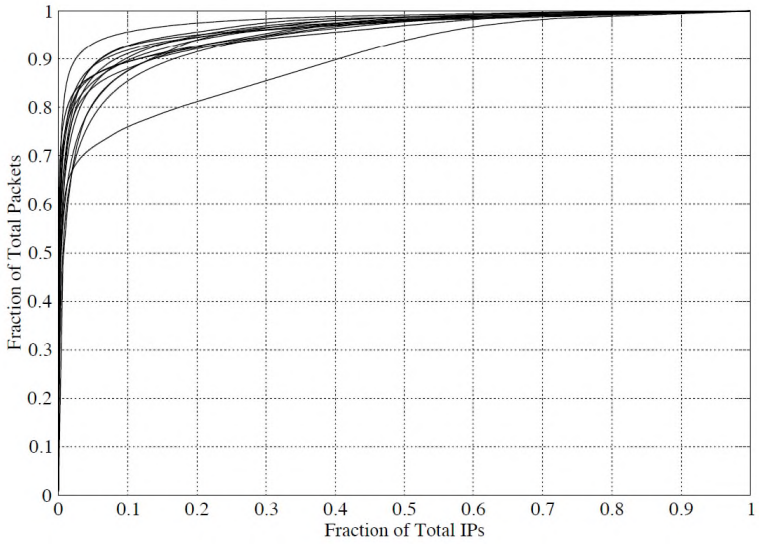

Figure 2.4: Contribution of individual IP to the total number of packets as seen at 14 network telescopes (Bailey et al., 2005b)

size of the address block (Moore et al., 2004). For a network telescope monitoring a /8 network block, $p(8) = 1/2^8 = 1/256$ which means that the telescope has a 0.39% chance of observing a packet from a single unique host. If a SYN flood was being performed, it is then also possible to calculate the expectation of observing the attack with $E(X) = nm/2^{32}$, where n is the number of distinct IP addresses being monitored and m is the total number of unsolicited packets sent during the attack. This is assuming per-packet random source addressing, reliable delivery, and one response generated for every packet in the attack (Moore et al., 2006).

In conjunction with event probabilities, it should be noted that there appears to be a disproportionate distribution between traffic observed and source address distribution. Research by Bailey et al. (2005b) found that nearly 90% of all traffic observed by 14 network telescopes were sent by less than 10% of the total observed source IP Addresses from each of the network telescopes. This disproportionate distribution observed by Bailey et al. (2005b) is illustrated in Figure 2.4.

2.3 Types of Malicious Traffic

Traffic received by network telescopes may be broadly classified into two categories as either active or passive. Passive traffic has been defined by Irwin (2011) as traffic from which no legitimate

Table 2.1: Passive Packet Configurations

TCP			
Flag		*Name*	*BPF Syntax*
RST		Reset	`tcp[tcpflags]=tcp-rst`
ICMP			
Type	*Code*	*Name*	*BPF Syntax*
0	0	Echo Reply	`icmp[icmptype]=icmp-echoreply`
3	any	Destination Unreachable	`icmp[icmptype]=icmp-unreach`
4	0	Source Quench	`icmp[icmptype]=icmp-sourcequench`
11	any	Time to Live Exceeded	`icmp[icmptype]=icmp-timxceed`
12	any	Parameter problem	`icmp[icmptype]=icmp-paramprob`
13	0	Timestamp reply	`icmp[icmptype]=icmp-tstampreply`
16	0	Information reply	`icmp[icmptype]=icmp-ireqreply`
18	0	Address mask reply	`icmp[icmptype]=maskreply.`
31		Datagram conversion error	`icmp[icmptype]=31`
34		IPv6 I-am-here	`icmp[icmptype]=34`
36		Mobile registration reply	`icmp[icmptype]=36`

from (Irwin, 2011)

response could be expected from the receiving system. This implies that a system receiving passive traffic would not reply to those packets and as such the originating source would not be able to gather any information. Causes of passive traffic have been summarised by (Irwin, 2011) as scanning activity from decoy scans, denial of service traffic such as Internet Control Message Protocol (ICMP), ping floods, through misconfiguration or mangled packets. (Irwin, 2011) provided a summary of passive packet configurations that would not result in a response being sent, shown in Table 2.1, along with the BPF syntax that could be used to implement the rules. Shortcodes for the numeric byte index provided by BPF in the libpcap library were used in the table to make semantic sense.

Active traffic was defined by Irwin (2011) as traffic which is expected to elicit a response of some kind from the targeted system. A summary of the most common active TCP and ICMP traffic is provided by Irwin (2011) in Table 2.2. Active traffic often represents port scanning traffic.

Ultimately both active and passive traffic form part of IBR, for the scope and purpose of this work, IBR traffic was further defined into three categories to assist in determining when traffic received by network telescopes should be regarded as malicious for the purposes of fingerprinting malicious hosts.

Table 2.2: Active Packet Configurations

TCP		
Flags	**Name**	**BPF Syntax**
NONE	NULL Scan	`tcp[tcpflags]=0`
FIN	FIN Scan	`tcp[tcpflags]=tcp-fin`
SYN	SYN Scan	`tcp[tcpflags]=tcp-syn`
PSH	PSH Scan	`tcp[tcpflags]=tcp-psh`
URG	URG Scan	`tcp[tcpflags]=tcp-urg`
URG\PSH\FIN	'XMAS' Scan Variations	`tcp[tcpflags]= tcp-urg&tcp-psh&tcp-fin`
PSH\FIN		`tcp[tcpflags]=tcp-psh&tcp-fin`
URG\FIN		`tcp[tcpflags]=tcp-urg&tcp-fin`
URG\PSH		`tcp[tcpflags]=tcp-urg&tcp-psh`
ICMP		
Type	**Name**	**BPF Syntax**
8	Echo request (ping)	`icmp[icmptype]=icmp-echoreq`
13	Timestamp	`icmp[icmptype]=icmp-tstamp`
16	Information request	`icmp[icmptype]=icmp-ireq`
18	Address mask request	`icmp[icmptype]=maskreq`
30	Traceroute	`icmp[icmptype]=30`
33	IPv6 where-are-you	`icmp[icmptype]=34`
35	Mobile Registration Req	`icmp[icmptype]=36`

from (Irwin, 2011)

2.3.1 Internet Background Radiation

IBR provides a unique perspective of nonproductive traffic on the Internet. Pang et al. (2004) defined "background radiation" as traffic sent to unused but publicly reachable IP Address space. In (Pang et al., 2004), the types of traffic often observed were defined as either malicious (flooding backscatter and vulnerability scans from worms) or benign (misconfiguration) broadly classified as nonproductive. This is as a result of the traffic targeting addresses that do not exist, servers that are offline, or servers that were not expecting traffic.

Related work in the field of unsolicited traffic analysis has proven successful in identification and tracking of DDoS attacks (Aben, 2009; Moore et al., 2006), worm propagation (Shannon and Moore, 2004; Harder et al., 2006), and network scanning (Barnett and Irwin, 2008). Furthermore, these studies into worm outbreaks provided details on the method used and the speed and effects of the worm's propagation.

The methodology used by the researchers required the capture of data from network telescopes and off-line analysis thereof to isolate the relevant packets or streams of packets and form their conclusions. This approach differs from the research in this work, by moving the analysis to near real-time instead of offline and focusing rather on the attributes of the hosts responsible for the malicious traffic than only on the traffic itself.

In order to more clearly differentiate between the types of IBR traffic that may be observed, we propose three categories to extend the work of Moore et al. (2004); Shannon and Moore (2004); Pang et al. (2004); Harder et al. (2006); Barnett and Irwin (2008); Aben (2009): Potentially Malicious, Truly Malicious, and Result of Malicious traffic. It is important to note that in this work, unlike Pang et al. (2004), flooding backscatter is not classified as malicious but rather traffic as a result of malicious activity. This was motivated by the fact that malicious hosts would be fingerprinted in this work, as such victims who produce flooding backscatter should not be targeted for reconnaissance.

2.3.2 Potentially Malicious Traffic

Potentially malicious (PM) traffic could be considered as port scanning. Port scanning is the process of enumerating open ports on a host. The output of this process provides an indication of which ports have services listening on them (open), which ports may be accessible through a firewall but not have a service listening (closed) or ports that are not accessible due to a firewall (filtered). Services exposed to the Internet amount in some part to potential attack

vectors available against a server (de Vivo et al., 1999) and as such it becomes important for both administrators of servers and attackers to determine the state of ports. From a malicious perspective, the motivation behind a portscan would be to identify services that may be targeted with an attack, such as exploiting a vulnerability or performing a password guessing attack.

A generalised approach to network-based computer attacks may be broken up into four phases, reconnaissance, exploitation, data ex-filtration, and persistent access. From an attacker's perspective, port scanning or probing (Pemberton, 2007) would be one of the first actions performed during the reconnaissance phase of the attack, after identification of open ports (accessible services), service enumeration would be performed to identify the characteristics of the services, such as software or OS type and version. During the reconnaissance phase, the attacker is not performing any overtly malicious activities. Of course port scanning may be seen as malicious if the intent of the attacker is to gain access to the system, however, without insight into the source of the port scan, it would be difficult to determine the intent. Furthermore, the question arises that if a host underwent a port scan after which the potentially attacker simply moved on and did not take any further action, the port scan would not have lead to any damages and as such, should it be considered malicious? Staniford et al. (2002a) found that the majority of port scans they detected, originated from compromised hosts. This Chapter concludes with a discussion surrounding the legality of port scanning to illustrate that port scanning on its own may or may not be considered malicious, based on the legal outcomes from several court cases.

Port scanners are specialized programs that may be used to determine which TCP or UDP ports of a server have processes listing on them (de Vivo et al., 1999). Nmap[9] is an example of such a port scanner and can be used to enumerate ports and the services listening on a host. It does this by sending packets to various ports and monitors the response received, while also noting when no response is received. The process of port scanning is discussed in more detail in Section 2.5.2, for the purpose of this research, port scanning was considered PM traffic.

With regards to PM traffic and port scanning or "probing" there are generally four methods used (Staniford et al., 2002a; Lee et al., 2003; Pemberton, 2007; Bhuyan et al., 2011), namely horizontal, vertical, strobe, and block scanning. Scanning for a single port across a range of IP addresses is known as Horizontal scanning. The frequency of Horizontal scans typically increase soon after a new vulnerability has been publicised as attackers wish to exploit one specific service. Scanning multiple ports on a single IP address is referred to as Vertical Scanning. Vertical scans

[9]http://www.insecure.org/

provide an attacker with a detailed view of the services exposed by the host that have been scanned and would typically be used to determine what type of exploits could be launched against that host (Bhuyan et al., 2011). A Strobe scan could be described as a combination of both Horizontal and Vertical scans, that is to say multiple ports over multiple IP addresses are probed. Finally a Block scan is a resource intensive scan that consists of all ports on multiple IP addresses. Quantified metrics for both Horizontal and Vertical scans were defined by Yegneswaran et al. (2003), Vertical scans were considered as consisting of six or more ports on a single computer and a Horizontal scan as consisting of five or more IP addresses within a subnet.

A number of studies have investigated scanning techniques (Staniford et al., 2002b; Panjwani et al., 2005; van Riel and Irwin, 2006) and detection thereof. Chen and Heidemann (2004) modeled the random scanning techniques of Internet worms and used that information to automatically detect and quarantine worm propagation through a router-based system. A visualisation framework named InetVis allowing for variable replay rates, adjustable time windows, and navigation of data was created by van Riel and Irwin (2006). This allowed detailed interaction with unsolicited traffic through a visual manner, depicted in Figure 2.5. In previous work we proposed Tartarus, a honeypot-based malware tracking and mitigation framework which utilised ARP poisoning on a local network to quarantine malicious hosts and dynamically generated blacklists to defend against the further spread of these agents (Hunter and Irwin, 2011). Visualisation techniques have also been created to identify port scans and DoS attacks, Kim et al. (2004) shows a 3D mapping of traffic clearly identifying portscans and DoS attack traffic in Figure 2.6.

2.3.3 Truly Malicious Traffic

Truly malicious (TM) traffic may be classified as traffic that has almost certainly some malicious intent. This may include actual exploitation attempts against a service, bruteforce, or password guessing attacks or DoS attacks. From these three types of attacks, only bruteforce or password guessing attacks could potentially be seen as not malicious but rather caused through human error, in very rare cases. For example, an administrator may type in the wrong IP address when attempting to SSH into a server. If the server located at the incorrect IP address was running an SSH server and login attempts were monitored, it would appear as if someone were attempting login by guessing passwords, while in truth it was human error. While possible, this is highly improbable and as such any password brute-forcing or guessing attacks would be considered malicious.

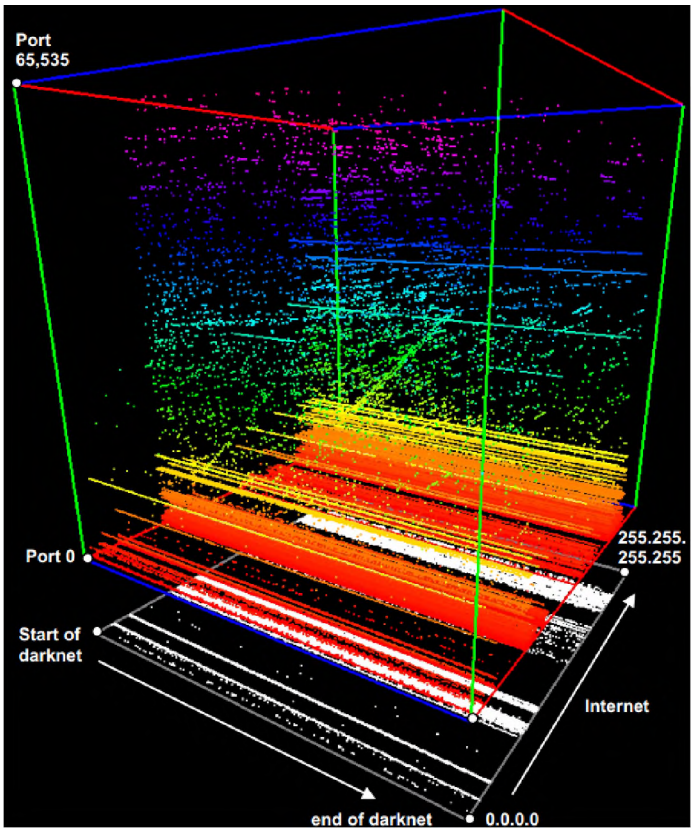

Figure 2.5: 3D Scatter plot of 857,085 packets observed from a network telescope (van Riel and Irwin, 2006)

With regards to TM traffic, only DoS attacks directed at a honeypot, network telescope, or other monitoring device were considered as TM. The purpose of DoS attacks are to exhaust the resources of a target. This may be accomplished through two types of DoS attacks: logic attacks and flooding attacks (Moore et al., 2006). Logic attacks attempt to exploit a vulnerability in a service to cause it to crash or degrade the service. Flooding attacks attempt to exhaust the processing power, memory or network bandwidth of a server by sending large numbers of spurious requests, often from thousands of hosts. Flooding attacks are particularly difficult to detect from the perspective of a victim because of the difficulty to distinguish between legitimate and malicious requests to a resource. Conversely, when using a network telescope, flooding type DoS attacks are relatively easy to detect, as response packets from the victim may end up on the telescope range. As such, only DoS attacks directed at monitoring infrastructure would be considered TM traffic, responses from victims were classified as Result of Malicious (RoM) traffic.

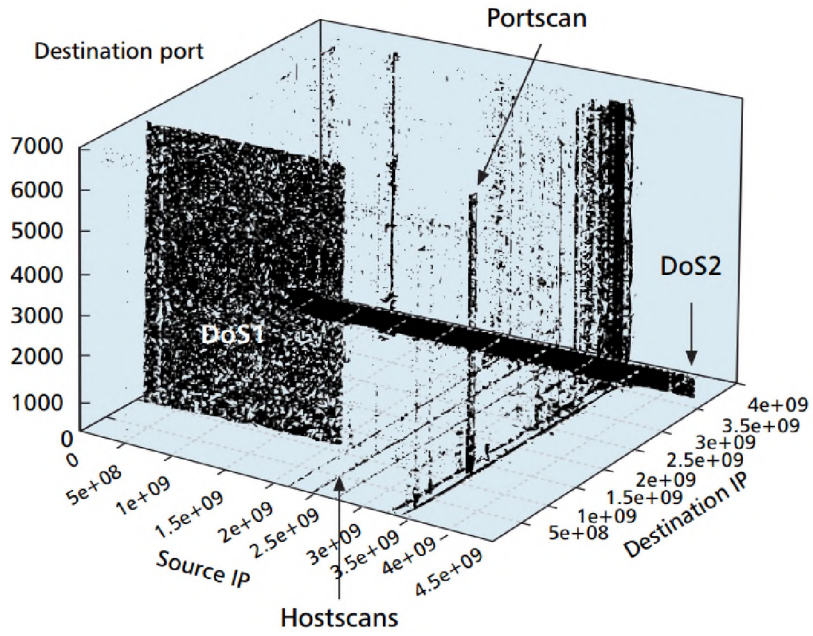

Figure 2.6: 3D plot of trans-pacific traffic (Kim et al., 2004)

Truly malicious traffic may also represent attempts to exploit a vulnerability in a service, the success of which may allow a remote attacker to execute commands, crash the service, or server or leak information from the targeted server. Detection of exploitation attempts are more difficult to observe than other types of malicious traffic as in most cases it requires a vulnerable service or at least highly detailed emulation of the service being exploited. As such, honeypots and more specifically high interaction honeypots are best suited for detecting exploitation attempts.

An example of TM traffic that can easily be detected by a network telescope would be that of the Slammer worm, as its exploit and payload is contained in a single packet (Moore et al., 2003). Another more recent single packet exploit was a remote code execution vulnerability identified in the Network Time Protocol (NTP), where a specially crafted packet could overflow a stack bugger (Fisher, 2014).

2.3.4 Result of Malicious Traffic

Not all IBR traffic may be considered malicious, aside from traffic caused by misconfigurations or other human errors, a large portion of IBR traffic consists of DDoS attacks from SYN floods. The reason for the classification of this traffic as a RoM traffic as opposed to TM, is as a result

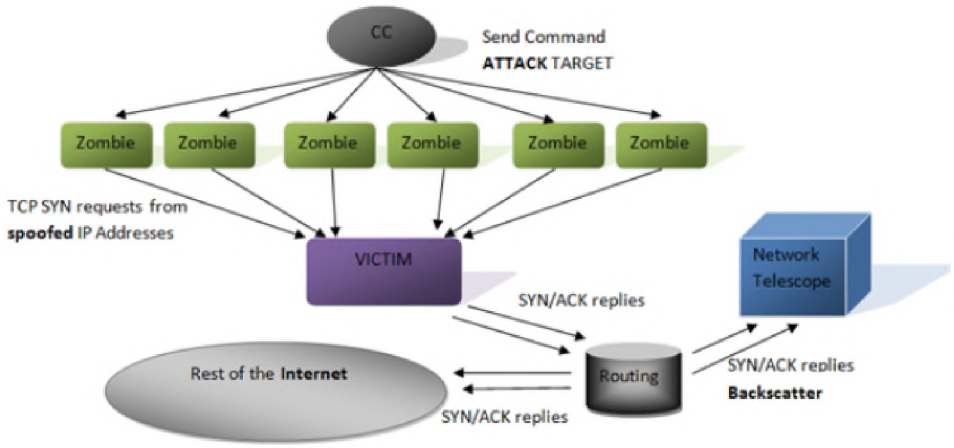

Figure 2.7: Illustration of a DDoS attack in the form of a SYN flood

of the origin of the traffic. The source hosts generating the portion of SYN flood traffic often detected by network telescopes are not the attackers initiating the DoS attack, but rather the victims of the attack.

An example of a SYN flood attack is provided in Figure 2.7. The SYN flood would start when a botnet's command and control element initiates an attack on a target, zombies (or bots) belonging to a particular botnet would periodically check in with the command and control server(s) for instructions, once an attack instruction is received, they would launch the attack. In the case of a SYN flood, this may be achieved by each bot generating large numbers of SYN packets, often with spoofed source IP addresses and sending them to the target (victim). Once received, the target of the attack would reply to the SYN packet with a SYN/ACK response, the second step in the 3-way TCP handshake. Since the bots responsible for generating the attack spoofed the source IP address, the targets of the attack would be left with a half open TCP connection as they would never receive an ACK response to their SYN/ACK. The result of thousands of half open TCP connections would cause the DoS condition for the victim. If some of the spoofed IP addresses from the attack overlapped with the public IP address space monitored by a network telescope, they would be routed to the network telescope which allow it to detect SYN flood DDoS attacks.

Decoy port scans (de Vivo et al., 1999), such as those implemented by the Nmap port scanner to hide a malicious host's IP address would also result in traffic being detected from victims (Irwin, 2011) and as such would also be classified as RoM. This differentiation from PM or TM

traffic needs to be made as this work is concerned with the fingerprinting of malicious hosts. If hosts responsible for RoM traffic were targeted for fingerprinting, the collected data would include both malicious and legitimate host fingerprints.

2.4 Geolocation

Determining the exact geographic source of malicious activity is a non trivial task (Arif, 2010) and often near impossible. While providing services based on a user's physical location has become a new mechanism for tailored services, it often requires specific hardware, software, and user interaction such as authorisation of the service. Many of these requirements are not possible when attempting to geolocate malicious hosts. Yet with the prevalence of malicious activity on the Internet, identifying the location of malicious hosts becomes a very interesting and useful metric. As the Internet is represented by a vast number of ever increasing, interconnected devices and complex networks it becomes difficult to map the physical location of a host or device to a geographic location. Research done by (Evans, 2011) showed that the number of devices connected to the Internet in 2010 had reached 12.5 billion, while the human population at the time was estimated at 6.8 billion, meaning that more devices were connected to the Internet than people on the earth.

The Internet consists and depends on physical infrastructure which exists at exact geographic locations around the world, however, the mapping of a logic Domain Name or Internet Protocol (IP) Address is not as easily translated to a physical location. The IP address (Forouzan et al., 1998), which is used to identify hosts on the Internet, does not provide the mechanisms required for a physical location in the world. While recent Top Level Domain names (TLD's) may provide references to geographic locations, these references are not enforced and often only apply to the content provided (or hosted) when following the domain, as such there is really no implied relationship between Domain Names and the physical location of the the host IP Addresses resolved by the DNS services. Examples of TLD's with geographic references include country code Top Level Domains (ccTLD), which are generally reserved for country, sovereign state, or dependent territory. The Internet Assignment Numbers Authority (IANA) was responsible for the implementation of internalised ccTLDs and have been described in Request For Comments (RFC) 1591 (Klensin, 2001), corresponding to International Organization for Standardization (ISO) 3166-1alpha-2 country codes (ISO, 2013). For example, the ISO 3166-1 alpha 2 country code

for South Africa would be "ZA". Similar to ccTLDs, DNS whois allows for location information to be added to the DNS records, however, this is neither enforced nor has it been widely adopted.

Autonomous Systems (AS) represents collections of IP routing prefixes that have been assigned to one or more network operators. This would represent a common and clearly defined routing policy for the networks to the Internet, Internet Service Providers (ISP) are required to have registered Autonomous System Numbers (ASN) (IANA, 2014), each one being unique and assigned to an AS. These ASNs uniquely identify different networks to the Internet, but it becomes more complex as organisations are allowed to use private AS numbers assigned by an ISP, as such the ISP supports multiple AS numbers but the Internet only sees the routing policy of the ISP. While this task is already difficult, malicious agents can employ a myriad of techniques to further hide their source location. Some of these techniques will be described in the following sections. It is important to note that these techniques are employed mostly by human adversaries, automated malicious entities such as worms and viruses often have no need to hide their source location based on IP Address.

2.4.1 Geolocation of Internet-Based Hosts

There are two broadly defined categories for the geolocation of Internet hosts, repository-based and measurement-based (Arif, 2010). The repository-based approach attempts to find the location based on a lookup, as a result this approach incurs a large overhead (Arif, 2010) as it requires the creation and maintenance of a repository of location information, which increases in size as the list of hosts on the Internet expands and degrades accuracy in time. The measurement-based approach finds location information based on latency measurements from some fixed hosts to targets (Arif, 2010). The advantage of the measurement-based approach is that it provides newer results through lookups, scalability, and more robust means of geolocation than the repository-based approach. Whereas the repository-based approach may produce more accurate results, depending of course on the accuracy and up-to-date status of the IP-to-location mapping dataset maintained (Arif, 2010).

It has previously been noted that the task of identifying an accurate model for latency to distance relationships between hosts on the Internet is a non-trivial task as a result of the variability in latency for a given distance (Arif, 2010). This is as a result of transmission characteristics on the Internet as they are unpredictably influenced by various factors such as circuitous routes between hosts and queuing delays on intermediate routers. Thus requiring a given model to deal

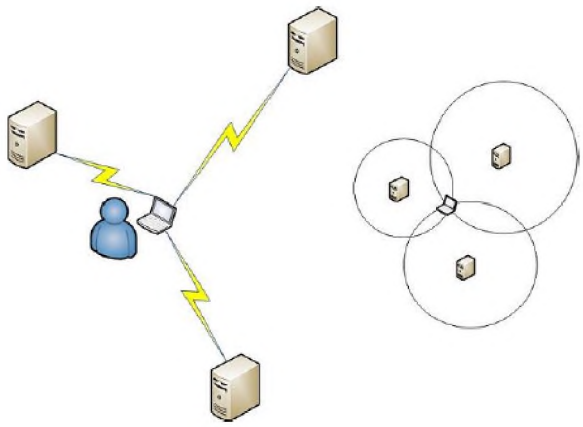

Figure 2.8: Measurement-based geolocation approach, showing measurement latencies to a target and constraints of maximum and minimum approximated distances (Arif, 2010)

with the variability of latency over distance with regards to the geolocation algorithm used.

For the most part, the current approaches for measurement-based geolocation employ end-to-end latency measurements from a set of nodes with known locations to the node for geolocation. The nodes with known locations are referred to as landmarks, whereas the nodes for geolocation are referred to as targets. Based on observed correlations between latency and distance traveled by data packets, these measurement-based approaches determine the estimated location of the target (Gueye et al., 2006; Padmanabhan and Subramanian, 2001; Wong et al., 2007). This is based on a minimum and maximum distance constraint from a given landmark, illustrated in Figure 2.8.

ICMP as specified in RFC 792 (Postel, 1981), is used by hosts and routers to send network-layer information to each other, this typically includes error reporting. Architecturally ICMP messages are carried inside IP datagrams. ICMP messages have a type and code field in their header; there are 13 ICMP types (0 to 12). Tools such as ping use ICMP echo request packets (type 8, code 0) to a destination IP address and wait for a ICMP echo reply (type 0, code 0). The time taken between the ICMP echo request packet being sent and the ICMP reply packet being received indicates the latency between the source and destination IP addresses in the form of Round-Trip Time (RTT) and as such can be used as a form of measurement. It should be noted that the level of accuracy that could be achieved through various geolocations strategies have generally been considered as poor. Muir and Van Oorschot (2009) studied several geolocation techniques and methods of evasion, they concluded that no single known method for Internet geolocation could

be considered "*robust*". The researchers did find that timing based (measurement) geolocation through HTTP refreshes showed promise (Muir and Van Oorschot, 2009), however, this technique would require user interaction and thus could not be applied towards the geolocation of malicious hosts in this study.

2.4.2 Repository-Based Approaches for Geolocation

One of the repository-based approaches to host geolocation involves IP tabulation against physical locations, data from a whois database has been used to create this kind of lookup table (Olson; Moore, Periakaruppan, Donohoe, and claffy, 2000; Muir and Van Oorschot, 2006). Unfortunately, Arif (2010) finds that due to the large number of hosts on the Internet, such an approach could not easily be scaled. A lookup table such as this would also be difficult to maintain with up-to-date and relevant data. The biggest shortfall of a repository-based approach would be the inability to account for dynamic IP allocation. Some variants to IP tabulation include whois lookups based on AS Numbers (Gao, 2000; Muir and Van Oorschot, 2006), whois lookup by domain name (Muir and Van Oorschot, 2006) and DNS LOC records (Muir and Van Oorschot, 2006).

Padmanabhan and Subramanian (2001) proposed two further repository-based approaches, namely GeoTrack and GeoCluster. Traceroute information from a host to the target was used in the GeoTrack method. Traceroute information contains a list of routers between the source and target in the path a packet travels. By using location information in the DNS names of routers along the path, the closest router to the target is selected and its location used as the target location. As such the accuracy of the GeoTrack method proposed by (Padmanabhan and Subramanian, 2001) depends on the distance from the target to the nearest router that contained a location hint in its name. The GeoCluster technique described by Padmanabhan and Subramanian (2001) grouped IP addresses into clusters based on their geographic proximity. Information from user registration records obtained from web-based services, such as email, was combined with the IP address clusters to determine the location mapping. Arif (2010) noted issues regarding the approach such as scalability, reliability, maintenance, and lack of available information from public access.

2.4.3 Measurement-Based Approaches for Geolocation

In addition to the two repository-based approaches proposed by Padmanabhan and Subramanian (2001), a measurement-based approach was also proposed GeoPing. This technique was based

on the assumption that hosts that where geographically close would have similar network delays with respect to other fixed hosts. When latency between a set of landmark nodes to a target were compared, GeoPing would consider the target location the same as that of the node with a known location with the most similar latency measurements. As a result the theoretical accuracy, GeoPing was limited by the distance to the nearest node with a known location (Padmanabhan and Subramanian, 2001). The accuracy of the technique is, however, also largely dependent on latency increasing effects such as congestion, path traveled, medium used, and the consistency of measurements (Kurose and Ross, 2007).

Contraint-based Geolocation (CBG) proposed by Gueye et al. (2006) makes use of ping times from landmarks as method of measuring delay. From this it would be possible to calculate a maximum distance bound for a given latency measurement, based on a distance-to-ping relationship observed between landmarks. During the geolocation process, latency between a landmark and a target would be used to calculate the maximum bound and then represented by a circle drawn around each respective landmark. The target is then assumed to reside in the convex region resulting from the intersection of circles, thus the target location would be estimated as the centroid of the convex region (Padmanabhan and Subramanian, 2001). CBG employed a process named Multilateration, which refers to the process of estimating a position using a number of measured distances to some fixed points. As such, mapping latency to distance in order to produce vectors. This was achieved by first measuring the latency between each landmark to each other landmark, which was used to create the maximum distance bound and constrain the target location. Figure 2.9 provides an example of the CBG approach when three landmarks are used to geolocate a target. The CBG method is the most similar to the Latency Based Multilateration method proposed in this work for host tracking.

Topology-based Geolocation (TBG) as described by Katz-Bassett et al. (2006), provides a similar approach to CBG, where the possible location of a target is also calculated as a convex region. The TBG technique calculated the maximum distance bound based on the maximum transmission speed of packets in fibre. This provided an estimate of the possible region, which was then further refined by taking into account inter-router latencies along the path, as obtained through the trace-route command. A global optimisation was then used to minimise the average position error for a given target and routers, which provided the location of the target.

A third approach named Octant, by Wong et al. (2007), improved the CBG method by including negative constraints (where a node may not be located). This was achieved by defining

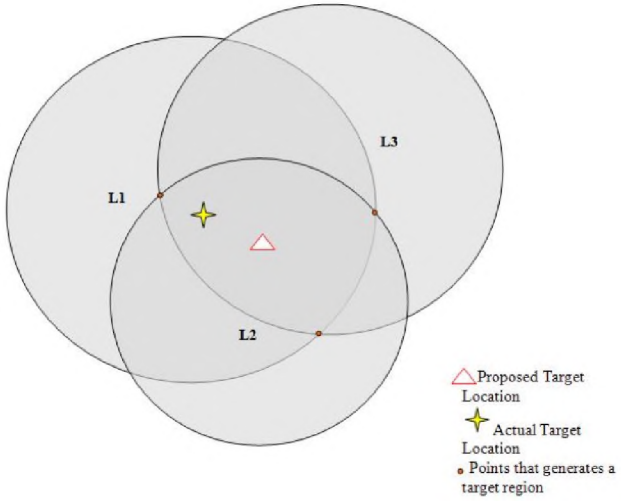

Figure 2.9: Example of CBG maximum bounds and convex region with estimated location and actual location (Arif, 2010)

not only maximum bounds but also minimum bounds from a given landmark where a node may not reside. Maximum and minimum bounds were calculated by first measuring the latency distance between each landmark and each other landmark to create a latency-distance graph. A convex hull was then determined around the points on the graph as illustrated by Figure 2.10. The convex hull included all latency-distance points, while the upper and lower convex hull facets were considered as maximum and minimum bounds of distance against the latency measurements. This further constrains the location of the target node, thus increasing the accuracy over the CBG method through the help of negative constraints. The shaded region in Figure 2.10 denoted valid point locations as bounded by propagation delay of the time of light in fiber. The convex hull around those data-points provided the positive and negative constraints for the node. Given a latency, the top and bottom of the hull represented the outer and inner radius and the vertical lines indicated percentile cutoffs. The convex hull was replaced with conservative positive and negative constraints when insufficient representative nodes remain (Wong et al., 2007).

Latency-to-distance measurements gathered by Arif (2010) indicated that the lower bounds of distance for a given latency were not apparent. As a result Arif (2010) determined that region produced by the work of Wong et al. (2007) with the Octant framework was still relatively large, resulting in lower accuracy. Figure 2.11 provides an example of the Octant approach when three landmarks are used to determine to location of a target.

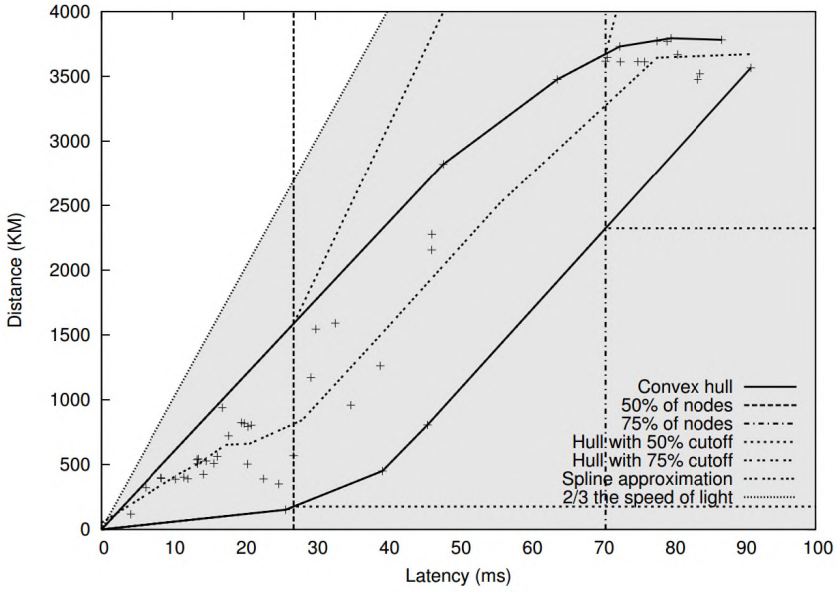

Figure 2.10: Latency to distance plot of peer landmarks for a representative landmark (Wong et al., 2007)

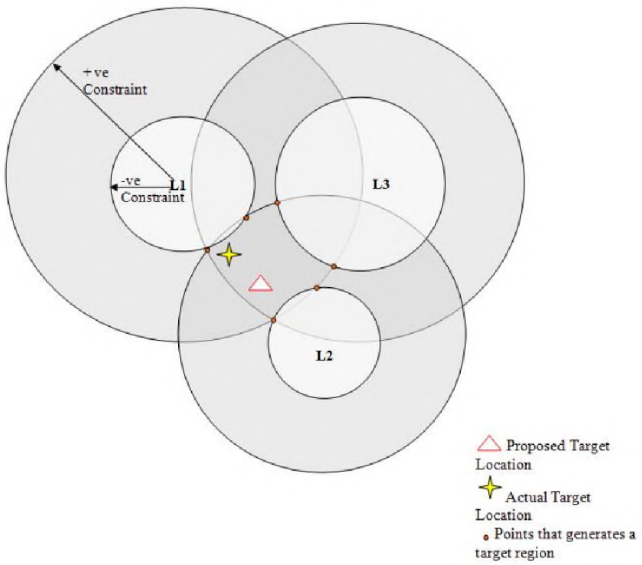

Figure 2.11: Octant example using three landmarks with minimum and maximum bounds (Arif, 2010)

2.5 Host Fingerprinting

Remote host fingerprinting refers to the process of identifying specific features of an OS and network services exposed by a remote host on a network, such as the Internet. This is achieved by connecting to exposed network services on the remote host and analysing the output, both at the application and network level. The output of this process may reveal specific protocol versions, vendor information and configurable parameters and can be stored as a fingerprint (Shu and Lee, 2006). Fingerprints are used in many different applications, for instance network administrators could use fingerprinting to collect information that would assist in network management. Allowing administrators to maintain accurate information pertaining to the numbers, locations, types of hosts, and services exposed by hosts that they manage (Fried et al., 2003). However, fingerprinting has also been classified as one of the major threats to infrastructure security (Bhuyan et al., 2011; Beverly, 2004). Hackers often use port scanners to determine exposed and vulnerable services to attack (de Vivo et al., 1999). Panjwani et al. (2005) has shown through an experimental evaluation how port scans have been linked as precursors to network attacks. By correlating port scans to attacks, Panjwani et al. (2005) determined that very few ICMP scans were followed by attacks, in total only 4% of port scans were followed by actual attacks, as opposed to vulnerability scans which were followed by attacks in over 21% of the cases. The results for portscans that led to actual attacks as found by Panjwani et al. (2005) have been summarised in Table 2.3.

Table 2.3: Distribution of scans leading to attacks

Type of Scan	No. Scans Observed	No. Scans Leading to Attack	Percentage Scans Leading to Attack
Port	694	28	4.03
ICMP	2,797	1	0.04
Vulnerability	1,399	296	21.16
Port & ICMP	11	0	0
Port & Vulnerability	59	42	71.19
ICMP & Vulnerability	184	5	2.72
Port & ICMP & Vulnerability	15	7	46.67

Panjwani et al. (2005)

In this work, fingerprinting is used to identify the characteristics such as OS and network services exposed by malicious hosts on the Internet. This allows for new types of correlations to be made between malicious activity and characteristics and also afford the ability to track

malicious hosts through the generation of potentially unique host fingerprints. It is important to note that fingerprinting can happen at various levels and through various techniques that require different levels of interaction with the host being fingerprinted. For the most part, fingerprinting could to be performed either through passive (Fried et al., 2003; Beverly, 2004; Shu and Lee, 2006) or active (Harris et al., 2008; Shu and Lee, 2006) means. Fried et al. (2003) notes how passive fingerprinting involves observing a trace of traffic without disrupting or interacting with it. Then when analysing the characteristics of the traffic, certain inferences regarding traffic could be made. Fried et al. (2003) described how the characteristics often examined by passive OS fingerprinting techniques include packet-header field values. Active fingerprinting involves sending packets to services and analysing their responses to infer the characteristics of the service or OS. Generally, active fingerprinting is considered more effective as it is possible to select specific inputs (packets) to be sent to services in order to illicit a response (Shu and Lee, 2006).

2.5.1 Operating System Fingerprinting

Due to implementation differences on OSs (Fried et al., 2003), identification of the OS variant becomes possible through fingerprinting. As these differences could be subtle and difficult to identify, most fingerprinting tools require expert manual effort in order to generate discriminative fingerprints and classification models (Richardson et al., 2010). Observable behavioural differences associated with TCP and IP fields have been summarised by (Richardson et al., 2010) in Table 2.4. In this table an "x" indicates that the field's value is likely to be influenced by the specific category, while a dot signifies that it would be hypothetically possible, however, unlikely for the value to be influenced by the category.

Approaches have been taken to eliminate manual intervention, one such method was described by Caballero et al. (2007) and Fried et al. (2003) through the automatic generation of fingerprints using machine learning. While this approach from Caballero et al. (2007) showed promising results in their tests, the application of their approach on a larger, more realistic scale was shown to be less effective by Richardson et al. (2010). This was largely due to the inability of an automated tool to exploit protocol and software semantics (Richardson et al., 2010) required for detailed fingerprint generation.

While many tools exist for remote fingerprinting, one in particular has become increasingly popular over time, this tool is called Nmap (Lyon, 2009). Nmap operates by sending specially crafted probe packets aimed at eliciting responses containing kernel specific differences. The

Table 2.4: Observable Behavioural differences associated with TCP and IP fields

	non-determinism	hidden state	network	hardware	applications	system configuration	OS source code
IP field							
version			•		•	•	x
hdrlen			•		•	•	x
tos			•		•	•	x
len			•		•	•	x
id	x	x	•				x
flags			•				x
frag			•				x
ttl	x		x		x	x	x
proto			•				x
chksum	x	x	x	x	x	x	x
TCP field							
seq	x	x	•				x
ack			•				x
dataofs			•		•	•	x
reserved			•				x
flags			•		•	•	x
win size			•	•	•	•	x
chksum	x	x	x	x	x	x	x
urgptr			•		•	•	x
op order			•		•	•	x
op MSS			•		x	x	x
opt wscale			•	x	x	x	x
opt tsval	x	x	•		•	•	x

Richardson et al. (2010)

designers of Nmap created 16 hand crafted packets in order to achieve this. Nmap parses the responses from these probes and feeds it into a classification model, which consists of a database of thousands of reference summary data structures for known OSs. When the classification model is not able to determine an exact match for an OS, it returns the nearest match, along with a percentage to represent the classification models certainty regarding the result. The nearest match is determined through weighted attributes and expertly classified heuristics. A particular challenged faced by tools such as Nmap as opposed to an automated machine learning approach to OS fingerprinting is that its signature database requires constant updates to keep its classification model capable of identifying new OSs.

2.5.2 Port Scanning and Network Protocol Fingerprinting

Port scanning can be defined as a technique used to identify the state of ports on a remote host, by sending probe packets to a set of ports. Port scanning was briefly mentioned earlier in Section 2.3.2 with regards to PM traffic. A side from potentially identifying the state of each of the 65536 available ports on a host, the ports that are discovered to be open (have a service listening) may be further scrutinised to fingerprint the particular service listing on that port. The results of a scan on a particular port generally result in one of the following three states open, closed, and filtered. These states can be identified through a variety of means, which all depend on the packet being sent to the port and the particular response received (or not received).

The most common types of port scanning techniques have been summarised in Table 2.5.

Table 2.5: Summary of port scanning techniques

Scanning Type	Description
TCP Scanning	Within Nmap, TCP scanning is referred to as a connect scan. With this form of scanning a complete TCP connection is made (3-way TCP handshake), once completed, the connection is closed. If the TCP connection was completed, the port would be considered open. If the connection failed during the 3-way TCP handshake, the port would be considered closed.
SYN Scanning	A SYN scan is also known as a "half open scan", this is because a full TCP connection is never completed during the scan. A port scanner such as Nmap would generate a SYN packet and send it to the target port, if it receives a SYN-ACK response packet it would consider the port as open. If however a RST packet is received, then the port is closed.
UDP Scanning	UDP scanning is also based on the response of the targeted server, if a UDP packet is sent to a port that is not open, the system would respond with an ICMP port unreachable message. The absence of a response would thus be considered as an open port. However, as a result this may lead to false positives as a firewall may drop the incoming UDP packet (indicating an open port when it is in-fact filtered), UDP is also a best effort protocol and does not natively allow for retransmission of packets that have been lost.
ACK Scanning	While ACK scanning does not provide information regarding a port being opened or closed, it is often used to determine if a port is reachable through a firewall.
FIN Scanning	FIN scanning could also be used to bypass a firewall, since most firewalls filter SYN packets, FIN packets can bypass a firewall without modification. When a closed port receives a FIN packet, it replies with a RST packet, indicated the port as closed. Open ports that recieve a FIN packet would ignore them and not send a response, thus indicating an open port.

Once the state of port has been determined as open, the service listening on the port may be fingerprinted. Banner grabbing is a technique often used to enumerate certain services on a port such as Hyper Text Transfer Protocol (HTTP), Secure Shell (SSH), Filte Transfer Protocol (FTP),

Simple Mail Transfer Protocol (SMTP), and Telnet. A banner grab is performed by initiating a connection to a port and observing the response, which often includes useful information such as service and version details. Nmap makes use of two particular methods to enumerate services on a port, the first utilises the nmap-service-probes file. The nmap-service-probes file contains a large number of service signatures, these signatures have been built up over time and are continually updated to identify new services or versions of services. The second, less accurate service identification method that can be performed with Nmap makes use of the nmap-services file which contains service information correlated with port numbers, if this file is not available, Nmap revers to the /etc/services file applicable for the current OS. The nmap-services file is derived from a complication of many sources, and contains more records than the IANA registered port-list (Cotton et al., 2011).

2.6 Legality

The open and inherently decentralised architecture of the Internet has made it difficult for law-makers to define legal boundaries regarding port scanning and fingerprinting. However, as part of this research required the portscanning and fingerprinting of malicious hosts, the legal implications regarding these techniques should be discussed. As with South African law and most laws, the legal process heavily involves the intent to commit a malicious act. So if a port scan was the precursor to an attack, both the attack and the port scan would be seen as malicious. That said it would be difficult to conclude whether the absence of an attack after port scan indicated that the portscan was performed without malicious intent or if there was no attack because the portscan failed to bear fruit. Over the years several cases have been tried with regards to port scanning, the outcome of these cases will be sumarised to give an overview of the current legal climate towards port scanning.

- In June 2003, Avi Mizrahi was accused by Israeli authorities of the offense of attempting to gain unauthorised access of computer material as a result of a port scan he performed against the Mossad website. On the 29th of February, 2004 he was acquitted of all charges with the judge ruling that actions such as port scanning should not be discouraged when they are performed in a positive way (Prawler, 2004).

- On the 9th of April, 2003 in Finland, a 17 year-old was convicted of charge of attempted computer break-in by the Supreme Court of Finland. He was ordered to pay $12 000

United Stated Dollar (USD) to a Finnish bank that he had port scanned in 1998, in an attempt to access their closed network, even though he failed to gain access to the network (Insecure.org, 2003).

- In December 1999, Scott Moulton was arrested by the Federal Bureau of Investigation (FBI) for performing an unauthorised port scan. He was accused of attempted computer trespassing under Georgia's Computer Systems Protection Act and Computer Fraud Act of America. At the time he was performing port scans on several servers for to check their security, with authorisation, during this time he scanned a web server belonging to another company. He was acquitted in 2000, the ruling was that no damage impairing the integrity and availability of the network had arisen (Poulsen, 2000).

Taking these three cases as an example, if a user has permission to port scan a network it is unlikely that legal action would be taken or prevail against them. However, this may also depend on different jurisdictions. If a user was to portscan a network without permission, but with no malicious intent, such as the case of Avi Mizrahi when he scanned the Mossad website, it is likely that a court would not rule against them. That said, they would need to co-operate with the law during the investigation and convince the judge that they had no malicious intention.

With regards to this research, it is difficult to request permission to scan malicious hosts that have been detected through the AR-Framework. Firstly by the time permission was granted (if it was) it is likely that host would no longer be found at the IP address, due to dynamic IP address allocation. Furthermore, the AR-Framework does not port scan any hosts that were not first suspected of some form of malicious traffic, be it TM or PoM traffic. It is for this specific reason that RoM traffic such as the replies of victim from a SYN flooding attack are not port scanned or included in the malicious host data.

2.7 Summary

The effects of malicious agents on systems, networks, or organisations influence the availability, integrity or confidentiality of those entities. This Chapter started by providing a summary of these malicious agents with a focus on their operation and motive. The malicious agents that were covered include, viruses, trojan horses and RATs, self propagating worms, spyware or adware, and human adversaries. By understanding these threats and the types of traffic they produce, it becomes possible to determine the best technologies that may be used to detect them. These

are the same agents responsible for generating the initial traffic that will cause the hosts they operate from to be fingerprinted in this research. While various agents have been shown to exist, these agents would likely operate over the plethora of different host computers and possible configurations, to this end Section 3.1 will introduce the concept of unique host representation through heterogeneous attributes.

Three techniques for detecting these malicious agents were then discussed. IDSs, honeypots and network telescopes. The ability of network telescopes and honeypots to observe unsolicited traffic made them ideal technologies for the identification of malicious hosts on the Internet. It was also shown that when used in isolation, network telescopes and honeypots become biased towards only a subset of threats, highlighting the need for a distributed and heterogeneous means of detection. Section 4.4 builds on these concepts by detailing the development of the real-time Distributed Sensor Network modules for honeypots and network telescopes.

Not all of the traffic received by network telescopes originated from malicious. As a result of this, the different types of malicious traffic were discussed. The discussion started by briefly outlining the difference between active and passive traffic, which then extended to IBR. Three new classifications for IBR traffic were then described. If filters could be implemented based on these classifications, it would ensure that only malicious hosts were targeted for fingerprinting, satisfying the need for targeting only malicious hosts. This type of Traffic Filter will be documented in Section 4.4.3.

The concept of geolocation was then discussed, this included a review of both repository and measurement-based approaches. Determining the location of a malicious host could provide a valuable fingerprinting metric, in addition to providing a form of situational awareness with regards to locality of assets and threats. The concept of Latency Based Multilateration for fingerprinting will be introduced in Section 3.4 and expanded upon with a feasibility study in Section 5.2.1. After geolocation, the concept of host fingerprinting was investigated. This extended to both Operating System enumeration and service enumeration through port scanning. Several common port scanning techniques were also reviewed. The application of these fingerprinting techniques towards malicious hosts would provide insight not that was not previously attainable from the analysis of network telescope traffic and forms the bases of Chapter 3, while Chapter 4 extends on fingerprinting by implementing them into the AR-Framework. It also highlighted the need to perform these fingerprinting techniques in near real-time to detection of malicious hosts as the likelihood of reaching hosts in malicious IP address space would degrade over time.

Further evidence of this need will be provided in Section 3.3 which documents a case study into data collection from hosts associated to a fast-flux botnet.

Finally, this Chapter concluded with a review of three legal cases regarding the implications of application and host fingerprinting without explicit consent from the target. While care would need be taken to fingerprint only malicious hosts, the act of doing so may still be considered illegal depending on the which jurisdiction was applied, further motivating the need for a form of traffic filtering before fingerprinting.

If your enemy offers you two targets, strike at a third.

Robert Jordan - Crossroads of Twilight

3

Remote Host Fingerprinting

This chapter introduces the concept of remote host fingerprinting and its utilisation throughout this work. Several methods and categorisation techniques are discussed and new methods of fingerprinting that were developed are introduced. This chapter provides the theoretical and technical foundation for this work.

Chapter Overview

- In Section 3.1 the hypothesis is made that for the unique and holistic representation of a host on the Internet, the intent and several characteristics of that host would need to be identified. Due to the constraints in which some of these characteristics have to be obtained, it would need to be done in a time-sensitive manner. While some of these characteristics on their own may not be unique, if enough heterogeneous characteristics are collected, they could provide a unique representation of a host and provide insight into their malicious

activity.

- In order to determine these characteristics, four categories of remote host fingerprinting were identified in Section 3.2. These categories were chosen to represent four sufficiently different aspects of a given host on the Internet. They were determined as Logical, Physical, Associative, and Behavioural attributes and were first defined in a research paper entitled "Remote Host Fingerprinting and Multi-sensor Data Fusion" Hunter et al. (2012b).

- Section 3.3 expanded on the concept of Fast-flux botnet association as an Associative fingerprinting technique. Based on the concept of identifying and attributing unique characteristics to hosts belonging to a fast-flux botnet and then making use of those characteristics as a fingerprinting method.

- Section 3.4 details Latency Based Multilateration (LBM) as a means for representing a physical characteristic of a hosts location on the Internet. The concept of LBM was adapted from multilateration, which was a technique that had originally been developed by the United States military as a means to locate downed aircrafts. LBM was first introduced in Hunter et al. (2012b) and then later expanded in Hunter et al. (2013).

3.1 Unique Host Representation and Host Profiles

While it is impossible to conclusively discover the identity of a host without physical access to that device, it may be possible to establish, with some certainty, a unique fingerprint of a host so that it may be re-identified over a period of time. Furthermore, if enough attributes of such a host could be identified, they could be used in conjunction with its observed activity to make inferences beyond what was previously possible with offline analysis of IBR traffic. This unique representation would require the combination of multiple heterogeneous characteristics that are considered to define a host. Each category would need to represent different characteristics that when combined, produce a sufficiently unique representation of that host. Thus generating a representation that could be used as a signature to re-identify a previously observed host and potentially correlate its intent with one of the malicious agents introduced in Section 2.1.

The unique representation of a host does not necessarily require a complete collection of attributes illustrated by the four categories of remote host fingerprinting. A suitably unique representation may be achieved by a subset of these attributes. During the process of establishing

whether a host previously observed is the same as a host currently being observed, a comparison of these attributes would need to be performed. The attributes need only be correlated while there is still uncertainty that a given host has more than one corresponding match. However, additional attributes may also be applied to increase the probability or certainty that a host is represented and re-identified correctly.

By generating a unique and re-usable host fingerprint one greatly increases situational awareness regarding a network and the hosts that interact with it. Hosts previously identified and placed on a traditional IP blacklist may now be placed on a characteristic blacklist, whereby signature matching could be used. This would allow for the mitigation of threats from dynamic IP address space. While this technique may not be viable in all circumstances due to a high overhead and possible false positives, it does hold value for mission critical infrastructure. Fingerprinting is often the first step towards mapping an attack vector. In the event of a sanctioned, offensive, strike back at malicious hosts, the categories of fingerprinting outlined in in Section 3.2 could assist in a verification process. This would ensure that the correct hosts are being targeted with the offensive strike as well as minimise the possibility of collateral damage, such as targeting innocent hosts when considering dynamic IP address space.

Once capable of uniquely representing hosts through remote fingerprinting, profiles may be created documenting certain behaviour from hosts over time. This would allow for a more in-depth analysis and understanding of malicious hosts as a history of past discrepancies may be constructed, allowing researchers to analyse traffic and behavioural patterns over a longer period of time. This could include following infection cycles of a host, attack rate, and a better understanding of what threats a host may pose based on past discrepancies. Successful modeling could lead to means of predicting attacks and assist in the development of new defensive techniques.

It is important to note that fingerprinting relies on traffic from a host, captured either passively or as a result of active probing. The caveat exists that the entity performing remote fingerprinting has no control, or method of verifying, whether the response from a host is genuine or contrived. While it might be unlikely that the host is attempting to interfere or even be aware of the fingerprinting process, researchers have shown this to be possible (Smart et al., 2000; van der Walt, 2004; Hunter et al., 2012a). It is assumed that the vast majority of hosts on the Internet are not attempting to subvert remote fingerprinting attempts. This assumption is based on the level of technical ability required to subvert fingerprinting attempts and the limited reasons for investing the time and effort to do so.

3.2 Fingerprint Categorisation

In order to establish four categories of remote host fingerprinting, it was necessary to consider the heterogeneous characteristics of a host that could be remotely identified. The *Logical* characteristics of a host may be seen as the Operating System and services running on that host. Figure 3.1 presents the four categories of remote host fingerprinting, along with examples of characteristics for each of the categories. The information extracted from the *Logical* representation may be further detailed, by observing the version or specific configuration of a service. *Physical* characteristics are represented by the hardware used by the host as well as the geographic location. The geographic location used to represent a *Physical* characteristic, in this sense, is not the location according to whom the IP address was leased but rather the physical location represented as a LBM fingerprint (Section 3.4), similar to multiple geographically separate trace-routes. *Associative* characteristics are represented by a logical relationship between a host and an entity. This relationship may be of any form, as long as it is possible to identify the relationship remotely. The final category is determined by *Behavioural* characteristics such as availability, uptime and bandwidth capabilities. Metrics from the *Logical* and *Physical* categories are in most cases the easiest to obtain through remote fingerprinting. They also provide a relatively detailed overview of a host. Obtaining data from the *Associative* and *Behavioural* categories are more challenging in addition to this, the data is more qualitative in nature. It is, however, important to note that with all remote fingerprinting techniques, one is not guaranteed a response from the host nor a conclusive fingerprint. Remote fingerprinting is a best effort technique, based on inferences of data returned from a remote host over which the observer has no control. That said, the results from remote fingerprinting at the current point in time are largely accurate.

3.2.1 Logical

Logical characteristics include metrics such as OS name and OS version, open ports, services running on those ports, and the versions of those services as well as service configurations. The three most popular tools used for fingerprinting within this category are nmap[1], p0f[2], and hping[3]. Enumerating the software characteristics of a host may provide insight into the purpose and intent of that host. A host with port 25/tcp (SMTP) open may act as a Spam mail server while a host

[1] http://nmap.org/
[2] http://lcamtuf.coredump.cx/p0f3/
[3] http://www.hping.org/

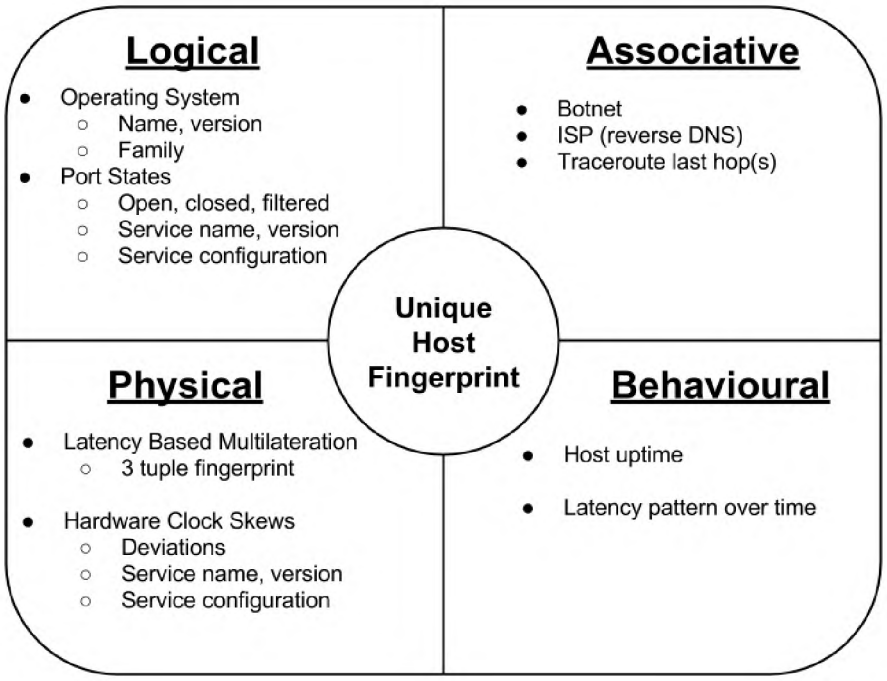

Figure 3.1: The four defined categories for remote host fingerprinting

with port 22/tcp (SSH) open might be used to pivot attacks from. These assumptions should only be made in context of other supporting information. *Logical* characteristics may also be used to map possible attack vectors if targeting that host with offensive capabilities. This would be done by providing information regarding the possible OS and services that may be vulnerable to attack. Thus logical fingerprinting provides information on remote hosts that could be used not only to potentially re-identify them. make assumptions on their purpose but also identify potential weaknesses that may be exploited if an offensive stance was taken. In certain instances it may also provide a completely unique value used to identify the host such as a ssh host key.

3.2.2 Physical

The *physical* attributes of a host are represented by attributes such as hardware and geographical location in the world. As mentioned previously geographic location is in principle synonymous

with the infrastructure that constitutes the Internet, as will be defined as part of the hypothesis for LBM fingerprinting. With regards to pure geographical IP translation it is very difficult and sometimes impossible to determine even the country of origin where traffic or rather intent originated. Proxies and tunneling are methods whereby traffic could be "bounced" between several hosts over the Internet before eventually reaching its intended destination. As such geographic location could be used as a metric for physical attributes, but on its own would not carry much weight. The study discussed in Section 2.5 which referred to the use of clock skews for remote fingerprinting of hardware is a better example of *physical* attribute fingerprinting. In order to fingerprint a device, the study exploited microscopic deviations in device hardware thus allowing Kohno et al. (2005) to uniquely identify devices, even when behind a NAT or firewall. Another physical characteristic of a host would be the hosts Media Access Control (MAC) address, which represents a unique hardware address of a Network Interface Card (NIC). There are, however, two challenges in obtaining the correct MAC address of a host: the MAC address cannot be obtained across subnets, which means that remotely determining the MAC address over the Internet would not be possible. Furthermore it is trivial to spoof a MAC address. The third metric for physical attributes is geographic location. This can be determined by the IP address of a host through an IP to location mapping service, such as provided by MaxMind[4]. However, Poese et al. (2011) found that this method was subject to potential inaccuracy. A more accurate means of creating a logical representation of a host's physical location is through LBM which is introduced in Section 3.4.

3.2.3 Associative

In previous work *Associative* fingerprinting was defined as the "*identification of a relationship, association or affiliation between a host and some physical or logical entity*" (Hunter et al., 2012b). While this approach is normative, to our knowledge it has not been formally defined or used as a fingerprinting technique. An ISP is an example of a host/entity relationship: in it's simplest form an ISP is responsible for providing Internet connectivity and an IP address to a device and could be partially determined through a reverse DNS lookup. This relationship is, however, not absolutely unique and is shared with other hosts associated with that ISP, nevertheless it does provide an Associative fingerprint. Botnets provide another example of a host/entity relationship: the host has a logical association with an entity: the Botnet. This can be logically extended by associating

[4]https://www.maxmind.com/en/geoip2-services-and-databases

the host to a specific Botnet. In Section 3.3 we discuss *Associative* fingerprinting in more detail by showing how the fingerprints of a 756 hosts belonging to the Kelihos/Hlux botnet was achieved. The attributes identified provided a unique fingerprint for hosts belonging to that specific botnet. Another example of an *Associative* fingerprint would be a previously observed indiscretion of a malicious host, assuming the use of a static IP address. Project Honeypot[5] provided a large collection of data from web-based honeypots around the world, associating malicious activity such as spamming, dictionary attackers, and comment spammers with IP addresses. As such an Associative fingerprint and assumption of static IP address usage could be inferred if the host were repeatedly observed performing the same malicious activity over time.

3.2.4 Behavioural

Behaviour attributes of a hosts could be considered to include the time spent on-line or off-line and average congestion or bandwidth available to that host. The behavioural characteristics of a host are the most challenging to monitor and collect remotely. Without authorization to run monitoring tools on the host, the metric becomes a best effort attempt coupled with inferences. The following methods were considered: Connection time monitoring which would require one of two prerequisites - a static IP address or a resolvable domain name. If the host were permanently located at either of those locations it would be possible to send periodic probes and construct a *Behavioural* pattern for the time that host spends connected to the Internet based on the responses. Bandwidth availability and load could be determined through probes such as ICMP ping requests and monitoring the response times which would give an indication of network activity. If, for example, the responses, on average took longer from 17:00 to 17:20 every day, indicating a pattern of increased latency, perhaps caused by performing backups to an off-site location or experiencing other load. That having been said, this method would be very unreliable as any type of congestion between the monitoring host and the monitored host would affect results. This would also require significant overhead due to the constant monitoring required. It is also very likely that ICMP response latency would be negligible if at all noticeable based on host load. This illustrates how difficult true remote monitoring is without a higher form of access to the actual host. These methods could constitute valid *Behavioural* attributes that, together with other fingerprinting techniques, could be used as a metric to uniquely represent the host.

There are existing services that may be used to remotely monitor the availability of hosts on

[5]https://www.projecthoneypot.org/

a network, including their current load and status of services managed by each host, but once again, these monitoring services often require user level access to the hosts being monitored. A more reliable method would be to determine the "uptime" and date since the last time a device was switched. However, it still remains a guess, the method involves checking sequential TCP SYN/ACK packet headers for a time-stamp header (Lyon, 2009). Different operating systems have different methods of updating this time-stamp, thus allowing inferences to be made based on sequentially monitored timestamps. Also note that the "uptime" enumerated in this way would degrade in accuracy as counters overflow and reset over a long period of time. This technique, however, seems to be the most reliable *Behavioural* attribute that could be gathered without much overhead and as such was used whenever possible to represent the *Behavioural* characteristic of a host from data collected by the AR-Framework in Section 4.6.

3.3 Fast-flux Botnet Association Case Study

Associative fingerprinting was previously defined as an affiliation or association between a host and some entity. The method that was used illustrate a technique of *Associative* fingerprinting was to identify a host as belonging to a botnet and if possible a specific botnet. As such this form of fingerprinting would only be possible if it was possible to identify the host as belonging to a botnet. This would be accomplished by inspecting the attributes of certain characteristics of a host and comparing them to known characteristics of hosts belonging to a botnet or variant of such a botnet.

Fast-flux is a technique used to update DNS information at a rapid pace, while it can be used for legitimate purposes such as load sharing, according to Nazario and Holz (2008) it has also become popular with resilient botnets. It has been shown that bot herders make use of Fast-flux DNS techniques to host content within a botnet which allows address mapping to constantly shift between different bots (Stalmans et al., 2012), making it very difficult to shut down. Nazario and Holz (2008) further explains how one of the tasks of the bots within the botnet would be to relay content back to a central server, also known as the mothership, C2 or command and control server. Based on the operation of a botnet, it would be possible to identify particular attributes associated with the characteristics of bots and potentially at the granularity of a particular botnet or at least type of botnet. A tool was created to enumerate Fast-flux botnets, this was done by continually sending queries to a known Fast-flux domain and recording the IP addresses that

were returned, while also noting the frequency by which the IP addresses were returned. At the same time as harvesting IP addresses a port scan was performed on each of the returned hosts. In addition to the port scan a HTTP HEAD request was sent to each host so that the HTTP Headers could be obtained. Stalmans et al. (2012) noted how bots are often used as a content relay in order to increase the resilience of a botnet. In doing so the botnet operator effectively creates thousands of hosts that hide where malicious content such as phishing websites are hosted and other bots acting as command relays. Sample data was obtained by harvesting a 1000 unique IP addresses from the Kelihos/Hlux botnet during March of 2012. This was achieved through multiple requests to resolve IP addresses associated with the botnet domain, for a duration of 10 hours. It was observed that during the process of enumerating IP addresses of bots associated with the particular botnet, some of IP addresses were returned more than three times as frequently as the vast majority of hosts, the frequency count of IP addresses returned are shown in Figure 3.2.

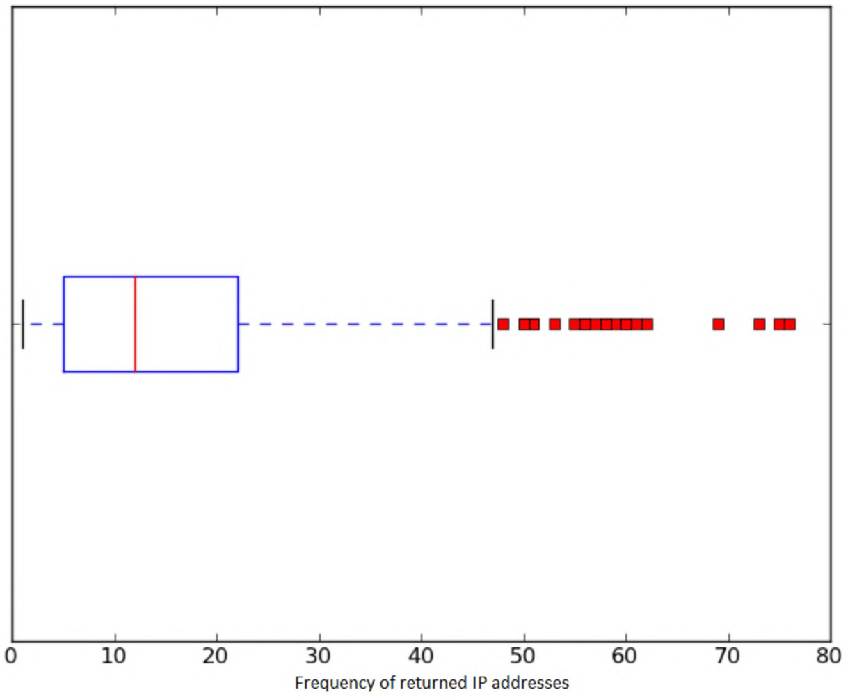

Figure 3.2: Frequency count of IP addresses returned during Kelihos/Hlux botnet enumeration

This could indicate that these outlying hosts may have been used by the botnet operators for specific tasks such as propagating updates or hiding a second layer of Fast-flux name servers. These hosts might also represent the bots with the most available bandwidth and if so, could

Table 3.1: Top 7 TCP ports from portscans of the Kelihos/Hlux botnet

Port	Frequency	Description
80	361	HTTP
135	155	Windows RPC Service
139	149	Windows RPC Service
445	148	Windows RPC/SMB Service
443	56	HTTPS
593	28	Windows RPC Service
25	26	SMTP

become likely targets during a botnet take-down procedure.

The results of the HTTP HEAD request to port 80 found that of the 756 hosts that replied and all of them shared the following two characteristics.

- The web server being used claimed to be: Nginx/0.8.34
- The last modified date in the header was set to: Sun, 11 Mar 2012 18:20:42 GMT.

It is inferred from this that the bots in the Kelihos/Hlux botnet where all configured using the same configuration script. All of the bots where running an Nginx webserver which could act as a reverse proxy. Using these two fields from the HTTP Header it was possible to determine a signature to identify further hosts that may potentially belong to this specific botnet. While this signature may not be valid over years, it would be valid for an extended period of time or at least as long as the botnet in question is functioning. Three weeks after collecting the initial results, the bots maintained the same HTTP responses. Any phishing payloads present on these bots would only be accessible through a specific URL, the bot would then act as a reverse proxy and forward data from the "mothership", this means that the the HTTP GET request to the root directory on the web server is likely to only return details regarding the initial configuration of the bot.

A basic port scan was also run against each of these hosts, however due to the time requirements of a portscan and the dynamic nature of bots connecting and disconnecting from the botnet over time, only 361 of the portscans returned results. The top seven open ports are shown in Table 3.1.

The Microsoft Windows RPC Service has been plagued by many vulnerabilities over the years and it was expected that the vast majority of bots connected to a botnet would have open ports for this service. It is likely that this was the cause of the initial compromise. From the HTTP

HEAD requests it was determined that of the 756 hosts that responded, all were running a Nginx HTTP web server. The port scans, however, were only able to complete against 361 of these hosts. Of the 361 hosts scanned, 26 were running a mail server, these hosts were likely used to send spam emails and propagate phishing attacks. Thus it could now be possible to use the data obtained through this reconnaissance of the Kelihos/Hlux botnet to construct a signature of what other hosts that belong to this specific botnet or other similar botnets would look like. By using this signature it may be possible to differentiate between a subset of legitimate and potentially malicious hosts on the Internet. Unfortunately the amount of data collect through the reconnaissance of the Kelihos/Hlux botnet was not sufficiently large enough to be combined with the data collected and presented in Chapter 5.

3.4 Latency Based Multilateration

Multilateration had originally been developed by the United States of America's military as a navigation based technique to accurately locate downed aircrafts. It was used to determine the location of an object by measuring the difference in distance from multiple stations with known locations. This was achieved by broadcasting signals with known intervals from the stations (ter Kuile, 2009). Unlike triangulation which is concerned with the measurements of absolute distance and angles, multilateration made use of only timing and ranges of distance to plot multiple hyperbolic curves that would intersect. This would reveal a small number of potential locations, thus providing a "fix" on the target location (ter Kuile, 2009).

The adaption of multilateration towards remote host fingerprinting involves the measurements of Round-Trip Time (RTT) to an IP address from three separate hosts, we define these hosts as *Base Stations.* This concept was first presented in Hunter et al. (2012b). It is possible to extend the measurements to make use of more than three *Base Stations* in order to improve the accuracy. However, for the purpose of this research, a prototype using three *Base Stations* were determined adequate.

Latency Based Multilateration (LBM) requires that the hosts acting as *Base Stations* are geographically separate. This approach was similar to the Constraint Based Geolocation proposed by Gueye et al. (2006). This separation naturally translates to different locations on the physical infrastructure that represents the Internet. The difference in physical location implies that measurements from each of the *Base Stations* to some unique entity would result in different ranges

of round-trip latency incurred. This is a result of the physical distance between the different *Base Stations* and a targeted host. Some of the *Base Stations* may be closer to the target and others further and under normal network congestion should maintain a constant latency. Packet-switched networks incur various delays while transferring data and LBM takes advantage of these delays in order to remotely fingerprint hosts. While the delays in packet-switched networks make it possible to measure a form of distance between hosts, it may also obscure results during times of abnormal load.

The process of latency multilateration is started when a set of ICMP type 8 (Postel, 1981) ping requests are sent to a host from the three *Base Stations*. When the ICMP type 0 replies are received by the *Base Stations* the RTT are recorded. Outliers are removed by extracting the values that fall between the $25th$ and $75th$ percentile in each respective set of RTTs. By discarding values below the $25th$ percentile and above the $75th$ percentile, the minimum and maximum outliers are removed from the RTTs. These values were found to be inconsistent and reduced grouping of a clear average. However in cases where 40% or more of the sets RTTs were represented by the mode, the mode was used. In statistics the mode is the value that occurs most frequently in a data set. The resulting subset of values from the $25th$ to $75th$ percentile are then used to calculate three mean values; one for each of the three *Base Stations*. Once again if the mode value was selected in a particular set, it is used instead of calculating a mean value. This produces a 3-tuple representation of the average time taken by each set of ICMP ping requests. The process of comparing these results with others to determine the likelihood of two hosts being the same is a non-trivial problem, especially with large datasets. To overcome this the 3-tuple is mapped to euclidean 3-space, representing each of the three mean timings as a value on a x, y and z axis. An example of multiple LBM fingerprints for three separate hosts are given in Figure 3.3. This results in a series of points in space which allows for easier comparisons between hosts as the distance between two points in space is a trivial calculation as shown with Algorithm 3.1. A threshold is determined and used to determine if the distance between two points is small enough to conclude that they may be the same host.

Algorithm 3.1 Adapted Pythagoras for a 3 dimensional Cartesian plane used during LBM calculations

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

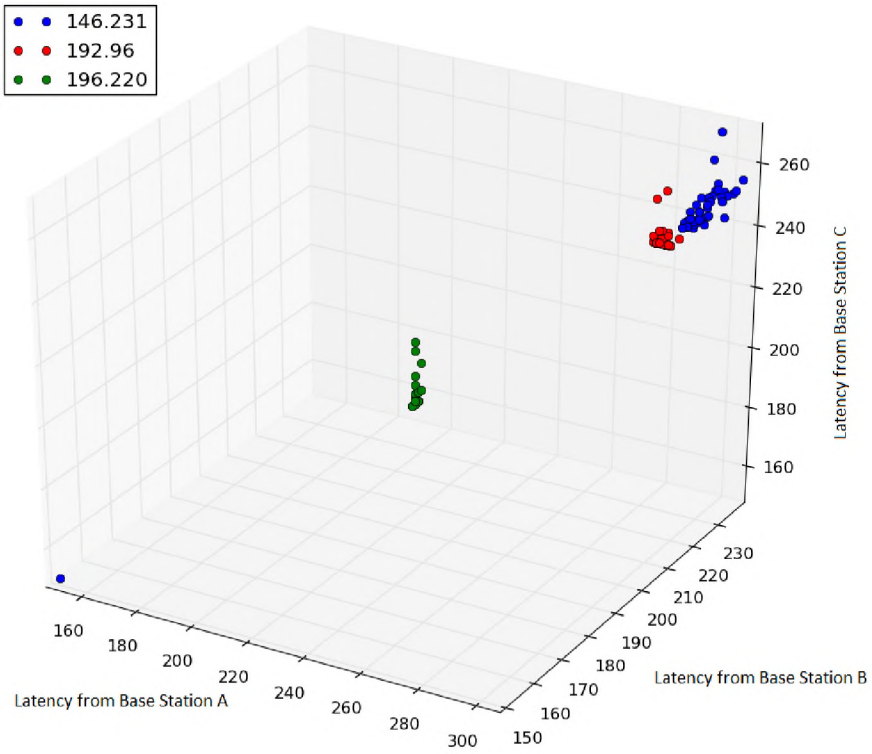

Figure 3.3: Multiple LBM fingerprints from three distinct hosts mapped to euclidean 3-space after outlier removal

3.4.1 Packet Switched Networks

A packet switched network is a communication network where all data is structured into suitably sized blocks known as packets and transmitted over a shared network. The Internet is a packet-switched network that routes packets from a source to destination according to the resources and capacity available. During the routing process various types of delays are incurred, these delays cause latency and are dependent on the physical mediums that the packets traverse and the effect of load on these mediums. The types of delays include processing, queuing, transmission and propagation. Packets traversing a network incur a variable latency which represents the time it takes for those packets to reach their destination. While this latency is variable, it is for the most part relatively consistent, barring events that may cause considerable congestion which could cause sporadic increases in the latency.

An ICMP packet may be used for ping requests such as those performed by the *Base Stations* for LBM measurements. The ICMP protocol has in the past been exploited by denial of service attacks, automated scanning (Lyon, 2009) and worms. In 2003 many Internet Service Providers started filtering ICMP Type 8 packets from their network boundaries as a precaution against the Welchia worm (Symantec, 2007). The self-propagating Welchia worm made use of ping requests in order to locate new computer that it could infect. In addition to Welchia, other host discovering services such as nmap still employ a ping sweep utility to test the availability of remote machines. Ping requests have also been exploited as a means of DoS attack. Two examples of DoS attacks that made use of ICMP type 8 requests include the smurf attack (Symantec, 2012a) and Ping of Death (Kenney, 1996). As a result of these attacks LBM may not return any results as certain hosts on the Internet have been configured not to reply to ICMP ping requests. The requirements and assumptions for LBM fingerprinting follows.

3.4.2 Requirements and Assumptions

The use of LBM fingerprinting requires that the targeted host is connected to the Internet and reachable, in addition to this the host must be able to respond to ICMP type 8 requests. It was explained in the previous Section 3.4.1 that ICMP packets have been used as a mechanism for a simple DDoS attack and as such some firewalls have been configured to drop these requests. In these cases LBM would not be able to retrieve any data. For the timings to be valid, all latency measurements for a particular host need to be done at the same time from each respective *Base Station*. If there exist down stream congestion, then the RTT measurements from each *Base*

Station would be influenced, instead of just one or two as the case may be if the echo requests were sent at variable or sequential points in time. It is because of the possibility of network congestions like these that a RTT mean proportion value is calculated as a secondary measurement, this proportion is detailed more in Section 3.4.3. As network congestion is unpredictable and would influence our latency measurements; outlier removal is always applied to the results. It is also assumed that unlike the research done by (Smart et al., 2000; van der Walt, 2004) the targets of LBM fingerprinting would not be aware of the fingerprinting activity and as a result would not actively try to influence or disrupt the measurements. Any packets lost during the latency measurements are retransmitted. Consistent bandwidth availability from the Base Stations was assumed, while this cannot be guaranteed as the scripts being run were making use of publicly available services, it does mean that the results may be improved upon with regards to accuracy through the use of dedicated *Base Stations.* The Processing phase is imperative to the successful and accurate use of LBM fingerprinting as it attempts to achieve the most consistent set of measurements from noisy datasets.

3.4.3 Latency Measurements

In order to better model the process of LBM, this Section will provide the steps involved during latency measurements for LBM. The measurement process is the first part of the LBM fingerprinting process. After measurements have been obtained and processed the data may be used to compare observed timings between many hosts. The aim of this was to achieve a likeness between two or more hosts that can be quantitatively measured. This measurement may indicate that an LBM fingerprint of one IP address is the same for other unique IP addresses, showing the same host being observed in dynamic IP address space over time. The following three phases model the process of LBM measurements and the processing required to produce a fingerprint.

Collection The data collection phase involves sending 16 ICMP type 8 requests to the host being fingerprinted from each of the three geographically separate *Base Stations.* Due to packet loss it was decided that less than 16 ICMP requests from a single base station would not provide sufficient data. The responses were collected and the RTT's grouped by their respective *Base Station.* Any ICMP requests that had timed out or was dropped would be ignored. This collection phase was performed in parallel between the three Base Stations assuring that any down stream congestion has a higher probability of effecting all results instead of just one or two of the *Base*

Stations, assuming that down stream congestion would be relatively constant. The reason for this will become apparent during the last phase of LBM fingerprinting when a latency proportion is calculated as a second comparative metric.

Processing The processing phase would be responsible for outlier removal and mean calculations, thus reducing each set of measurements to a single, aggregated value that would be used to represent each values within the 3-tuple LBM fingerprint. During outlier removal, the first RTT value from each set is removed, this is done in order to mitigate latency incurred during the inital routing and updating of IP tables as the first ICMP type 8 packets made its way towards a host through various hops on the way. The mode is then calculated, if the mode is found to represent 40% or more of the RTT values in a set, it is used as the mean. If the mode represents less than 40%, further outlier removal would be performed. The additional outlier removal involves extracting all the RTT values from each set that fall between the 25th and 75th percentiles of each respective set. The resulting values are then used to calculate the mean for that set. The mean value from each set is used to represent an axis in euclidean 3 space. Two examples of latency measurements before and after outlier removal are given in Figures 3.4 and 3.5. These histograms show the latency measurements from a single host over a long period of time. Figure 3.4 shows the raw latency measurements before any outlier removal was performed. While Figure 3.5 shows how outlier removal has decreased the size of possible values that would represent the LBM fingerprint. Table 3.4.3 shows the calculations of mean values before and after processing. The value and importance of the processing phase is more clearly shown in Table 3.4 where the standard deviation from the mean is shown before and after processing.

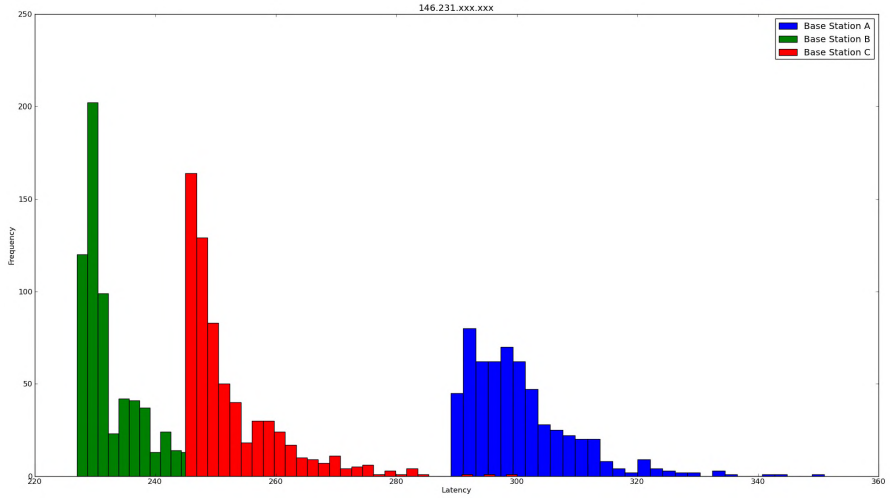

Figure 3.4: Raw response times from several LBM fingerprint attempts for one host

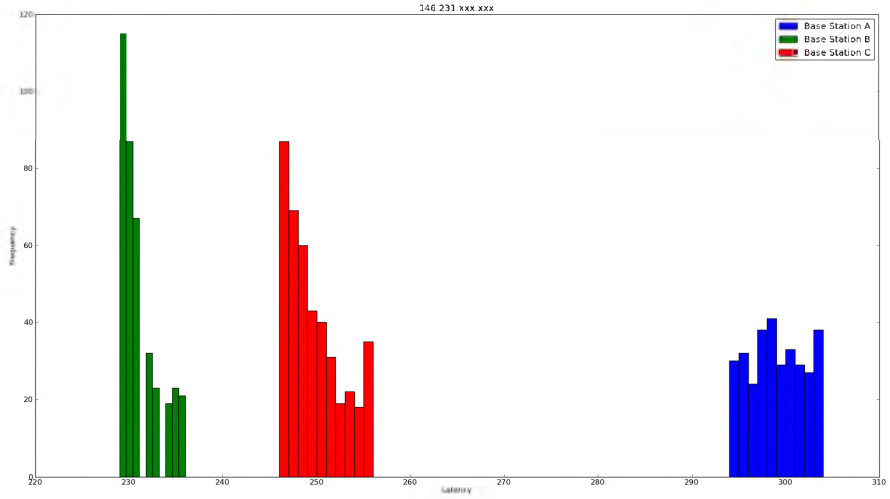

Figure 3.5: Response times from LBM fingerprints with outliers removed

Proportion The proportion value is used to represent the LBM fingerprint in such a way as to mitigate the influence of network congestion, assuming that the congestion affected all of the measurements from the three separate *Base Stations*. The closer the network congestion (if it exists) to the source being fingerprinted, the more likely it would be reflected in all three of the measurements. By calculating the proportions of the three measurements it is possible to identify a relationship between fingerprints that could be used as an additional comparison between probable LBM matches. If

Table 3.2: Mean values calculated before outlier removal.

A	B	C
299	228	247
299	229	245
299	229	248
301	231	249
298	230	249

Table 3.3: Mean values calculated after outlier removal

A	B	C
298	228	245
297	228	245
298	229	246
298	229	246
297	229	247

Table 3.4: Standard deviation of mean values before and after outlier removal

	A	B	C
Before Outlier removal	1.09	1.14	1.67
After Outlier removal	0.54	0.54	0.83

the proportions are similar and the RTT values are different enough such that the matching threshold value would not classify two hosts as being the same, the hosts could be flagged for re-testing. In so doing the proportion calculation could assist in mitigating sporadic congestion.

3.4.4 Latency comparison

After successful LBM measurements and processing, an LBM fingerprint is created as a metric for host re-identification. This 3-tuple value would then be used for comparison between different host measurements to determine the likeness or probability that any two measurements may belong to the same host. The host IP Address could be used to support the comparisons findings, if it is assumed/known that the IP Address space is static. Alternatively, if a reverse DNS is performed and two similar LBM fingerprints share the same ISP, then this would certainly increase confidence in the likelihood that those hosts are the same. ISPs are assigned ranges of IP addresses, which they in turn assign to clients. As they connect and disconnect from the ISP's network, the IP addresses assigned to them may change over time. IP Address ranges are categorised according to ASNs . These numbers are used for route propagation algorithms such as BGP and as a method of identifying the ownership an IP address range (IANA, 2014).

As mentioned earlier the 3-tuple value representing a LBM fingerprint is mapped to euclidean 3 space when performing comparisons. A comparison between two LBM fingerprints is then achieved by calculating the distance between the two points in space using the equation found in Algorithm 3.1. This would produce a value representing the logical difference between the

two given LBM fingerprints in latency, based on physical network infrastructure. This distance is then compared to an appropriate threshold variable. If the distance is smaller than or equal to the threshold variable, it may be said that both LBM fingerprints are from the same host. Furthermore a confidence value would be calculated based on how small the distance between the two hosts are, with reference to the size of the threshold variable. This confidence value would represent the certainty of the LBM comparison that any two fingerprints are from the same host. A small distance would result in a high confidence value. Inversely, the closer the distance comes towards the threshold, the lower a certainty value it would translate to.

The calculation of a proportion to mitigate sporadic but consistent congestion over the time of measurement is done before or after LBM fingerprint comparisons. These proportions would be compared between two LBM fingerprints to confirm the validity of the latency measurement results. The proportions were calculated as a percentage or ratio of the sum of mean latencies across the set of 3-tuple values representing the LBM fingerprint. An illustrative example is given in Algorithm 3.2, using a 3-tuple of (30, 80, 90).

Algorithm 3.2 Calculating the LBM proportion fingerprint

$$(30, 80, 90)$$
$$(30 + 80 + 90) = 200ms$$
$$(30/200, 80/200, 90/200) = (0.15, 0.4, 0.45) = (15$$
$$(15\%, 40\%, 45\%)$$

The sum total of latency for this set would be 200ms. Thus the proportions would be the latency divided by the sum total. These values would represent a relationship between the latency measurements from the 3 *Base Stations*. The comparison between two proportions is done in the same way as the LBM fingerprint comparison; by calculating the distance between two points in space using Algorithm 3.1 and comparing that result to a separate static threshold variable.

It is important to note that this proportion based comparison is not the same as the actual LBM latency comparison, the LBM latency comparison takes precedence. This is because the proportion comparison between two LBM fingerprints may match, no matter how large the difference in latency is. If under normal circumstances with no additional latency induced on the network, the LBM measurements of two known separate hosts are given in Table 3.5.

When calculating and comparing the proportions of theses LBM measurements they appear to match with a difference of 2.83, however, the original LBM latency comparison fails (134.29). The latency measurements were wholly different, however, the proportions between the two sets

Table 3.5: Standard deviation of mean values before and after outlier removal

	LBM Fingerprint	LBM Proportion Fingerprint
Host A	(115,220,80)	(28,53,19)
Host B	(52,107,44)	(26,53,21)
Comparison	134.29	2.83

are the same. This would under normal circumstances indicate the possibility of extra latency being induced into the network at a downstream (close to the final hop) location. Thus affecting all three of the measurements from one of the LBM fingerprints. In order to determine if the larger of the two LBM measurements was affected by latency induced by congestion, new latency measurements would be taken for that IP address. This is assuming the IP address is reachable at this point in time. As such this example shows why the proportion based comparisons serve only as an additional metric to detect the possibility of sporadic network-based congestion.

3.5 Summary

This Chapter presented the concept of unique host representation through the collection of multiple heterogeneous characteristics. Many of these characteristics required data collection to begin shortly after a malicious host had been identified. As stalling this process might lead to the host being associated with a different IP address and as a result, not be enumerated. To this end, Chapter 4 will document the development of the rDSN modules and Automated Reconnaissance Framework for near real-time data collection.

Four categories for remote fingerprinting were identified to represent a host, these categories represented sufficiently different aspects of a given host. This included *Logical*, *Physical*, *Associative*, and *Behavioural* attributes. Two new methods of fingerprinting were introduced, these methods were Fast-flux botnet association, that could potentially be used to represent an *Associative* attribute and Latency Based Multilateration to represent a *Physical* attribute. While Fast-flux botnet association was demonstrated as an example of Associative fingerprinting category, it was ultimately discarded as it would only apply to an incredibly small population of malicious hosts on the Internet. LBM formed part of the *Physical* fingerprinting as it provided a logical representation of a host's physical location on the Internet, its use, however, would require significant resources, which could limit its application.

In the next Chapter we discuss the design of the AR-Framework, which will be used to

fingerprint malicious hosts that have been identified from distributed honeypots and network telescopes. The AR-Framework would employ reconnaissance techniques to gather attributes from malicious hosts on the Internet, these attributes would represent the four categories of fingerprinting presented in this Chapter. Visualisation tools that were developed to aggregate and represent data collection from the AR-Framework, including the four categories of fingerprinting presented in this Chapter will be discussed in Section 4.7.

I don't care if I pass your test, I don't care if I follow your rules. If you can cheat, so can I. I won't let you beat me unfairly - I'll beat you unfairly first.

Orson Scott Card - Ender's Game

4

The Automated Reconnaissance Framework

This Chapter details the Automated Reconnaissance framework (hereafter AR-framework), a prototype developed for this research. The AR-framework was based on an adapted multisensor data fusion model initially developed by Waltz and Llinas (1990) for use in the US military. The model was chosen as a result of its ability to assist in the aggregation of heterogeneous data sources and how it incorporated phases of alignment, analysis, and merging of data to assist in decision-making and support processes. This suited the requirements of the AR-Framework regarding the analysis and retrieval of information from malicious hosts on the Internet. The identification of malicious hosts was achieved through the deployment of honeypots and a network telescope; referred to in the adapted model as data sensors. The sensor's inherent ability to detect, almost exclusively the activity of malicious agents such as worms and human adversaries, as previously discussed in Section 2.1 and 2.2, made them ideal data sources. Visualisation tools that were developed to provide insight into data generated by the data sources and AR-Framework are

discussed towards the end of the Chapter.

Chapter Overview

- Section 4.1 provides an overview of the assumptions and limitations that were required in order to develop the AR-Framework and support its data collection capabilities. This included limitations based on the legality of certain operations and assumptions that were required of AR-framework operation and data collection.

- An introductory, high level overview of the AR-Framework is presented in Section 4.2. The purpose of this overview was to provide a technical introduction to the AR-Framework components and how they function. These functions and the motivation for their design decisions were examined in more detail in Section 4.3.

- The three, core components of the AR-Framework, namely; Collection, Assessment, and Alignment were discussed in Section 4.3 with reference to the adapted multisensor data fusion model.

- Data source management for the AR-Framework was managed by the Collection Phase detailed in Section 4.4. This included the design of the rDSN modules for both the network telescopes and honeypots.

- The Assessment Phase presented in Section 4.5 was responsible for maintaining session management from near real-time data provided by network telescope and honeypot sensors. Once data had been analysed and categorised in the Assessment Phase, the information retrieval (Reconnaissance) process would commence.

- The reconnaissance process was handled by the Alignment Phase detailed in Section 4.6. This phase would execute numerous reconnaissance and fingerprinting modules in order to gather information from a given target, using both active and passive techniques.

- Two data visualisations and aggregated metrics dashboard which were developed to explore information gathered through the AR-Framework are introduced in Section 4.7. The visualisations mapped attributes from Logical and Physical fingerprinting categories previously identified in Section 3.2. Both of these visualisations included a temporal dimension, through either quantitative or qualitative means. The data aggregator was created to display and navigate aggregated and sorted information.

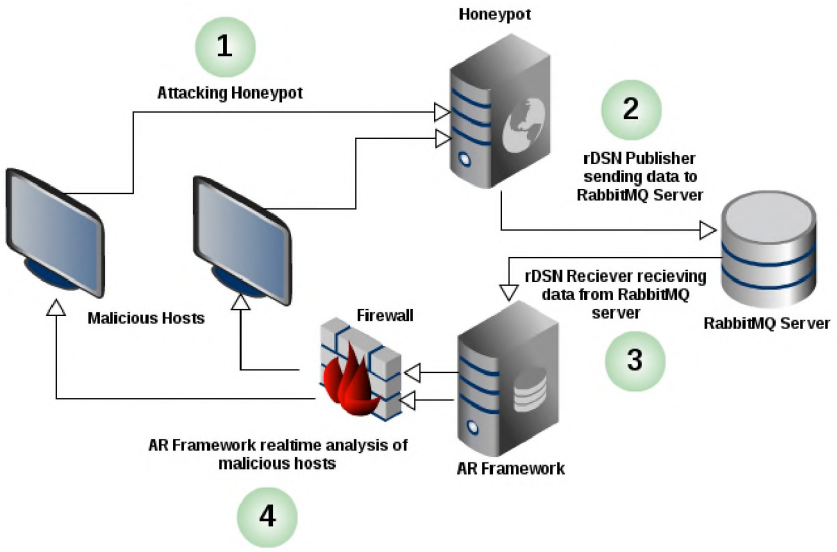

Figure 4.1: AR-Framework deployment with single honeypot

An example of an AR-Framework deployment along with external components are provided in Figure 4.1. This illustrates how activity from malicious hosts is detected by a honeypot (1). The information is then passed to a RabbitMQ[1] Server through a near real-time Distributed Sensor Network (rDSN) module (2). The AR-Framework then receives this information (3) from the RabbitMQ server and as a result triggers data analysis and reconnaissance modules to remotely fingerprint the malicious hosts (4).

The first goal of the AR-Framework was to fingerprint potentially malicious hosts on the Internet. In addition to fingerprinting malicious hosts the framework was responsible for collecting as much information on a given host as possible. This was achieved through the use of fingerprinting techniques such as those outlined in Section 2.5, Open Source Intelligence (OSINT) and inferences specified by the authors and documented information such as RFC6335 (Cotton et al., 2011).

4.1 Limitations and Assumptions

As one of the primary functions of the AR-Framework was to remotely gather fingerprint data, the caveat exists that any elicitation of information from a remote source, cannot categorically be

[1] http://www.rabbitmq.com/

proven as true without control over that source. Thus the authenticity and validity of the data or information that was returned may be either true or partially false. An assumption then needs to be made as to which degree to trust this information. With regards to Open Source Intelligence (OSINT) sources, the AR-Framework only made use of well known sources that have no apparent reason or motive to falsify information in any form. These sources included Maxmind[2] and Project Honeypot[3]. It should be noted that Poese et al. (2011) investigated the accuracy of geolocation services such as MaxMind and found that the services often geolocate IP addresses at a country level successfully, however, they do tend to be biased towards countries that have a more extensive technological footprint. Furthermore, their geolocation capabilities at a city level are far less accurate. While Poese et al. (2011) concludes that these services are not useable for general-purpose geolocation, their comparison data was from a single, albeit large, European ISP and thus their findings did not represent global accuracy. It was mentioned in Section 3.1 that researchers have shown how it was possible to interfere with remote fingerprinting and reconnaissance techniques. Thus it is necessary to stress the point that without physical access to a host, the accuracy of remote fingerprinting can never be accepted as conclusively true. That said, it is highly unlikely that the vast majority of hosts on the Internet would attempt to interfere, let alone be aware of the fingerprinting process. It is thus assumed that all data retrieved and generated by the AR-Framework is true to the best of our knowledge.

It was also important that the majority of the remote fingerprinting processes outlined within the AR-Framework were executed within near real-time as soon as a malicious host was discovered. The dynamic nature of the Internet dictates that there are constantly hosts being connected and disconnected. Failing hardware on the Internet's backbone infrastructure or that of an ISP may result in large network ranges becoming unreachable in an instant and that would only be considering accidental outages. By way of example, on the evening of the 27th of January 2011, the Egyptian government attempted, and largely succeeded in shutting down Internet access to its population (Dunn, 2011). As a result, if a malicious host was not fingerprinted soon after being detected, the risk of not reaching that host increases significantly over time. In Section 3.3, data was enumerated from IP addresses associated with a Fast-flux botnet. These bots were connected to the Internet at the point of time when IP addresses were harvested. In the process of fingerprinting those hosts, which was achieved with multiple threads as soon as an automated script became aware of a new IP address, minutes started elapsing as the fingerprinting modules

[2]https://www.maxmind.com/en/home
[3]https://www.projecthoneypot.org/

tried to keep up with new IP addresses. The fastest module to complete was the netcat[4] based HTTP HEAD request, which only managed complete on 756 of the 1000 identified IP addresses. The more resource intensive and as a result slower, Nmap module only managed to complete scans of 361 of those hosts. With that in mind, the AR-Framework was designed to perform tasks in near real-time, within 2 to 14 seconds of discovering a malicious host from either the network telescope or honeypots. The time delay was incurred due to collection, parsing, and network based transmission delays. Sensor data from network telescopes were also delivered faster than that from honeypots, due to implementation differences in the rDSN modules. The incurred delays from the honeypot rDSN module is discussed further in Section 4.4.2.

Lastly some operations of the AR-Framework may be classified as illegal according to legal cases discussed in Section 2.6 in certain parts of the word, this included the use of Nmap for active fingerprinting. While more advanced information gathering could have been performed, it was decided that these methods would most certainly over step boundaries imposed by laws, resolve the protection afforded from academic association and ultimately produce additional overhead with regards to resource usage. For the purpose of this research the fingerprinting modules used were deemed sufficient to collect the amount of data required to fingerprint hosts according to the four categories defined previously in Section 3.2.

4.2 Design Overview

Some of the reconnaissance modules that would be employed by AR-Framework, such as port scanning, could take significantly longer than other modules, thus needing data to be aligned and associated with the correct originating information from data sensors. Furthermore, session management of new data would need to be performed to insure that the same host was not fingerprinted multiple times in quick succession. As a result it became apparent that information flow was going to play an important role during the analysis of malicious hosts using the AR-Framework. Combined with the large number of responsibilities the AR-Framework would be tasked with, it was decided to group similar functionality, from an architectural viewpoint, according to task. This allowed for clearly defined inputs and outputs from each of the four components within the AR-Framework as well as simplifying the process of extending the Framework as might be required at a later stage. In order to support modularity, decrease deployment time,

[4]http://sectools.org/tool/netcat/

and increase the ability to fine tune the AR-Framework characteristics and functionality, the AR-Framework may be configured using a single text based configuration file. The AR-Framework was written using the Python[5] programming language. Python was chosen as it provided a rapid development environment, was platform independent and provided access to many supporting libraries that were relevant to the AR-Framework.

A high level overview of the AR-Framework was given by the diagram in Figure 4.2. Network telescopes and honeypots represent the data sources in this diagram, they were deployed in separate networks from the AR-Framework. Information from data sources were sent to a RabbitMQ message broker through the Advanced Message Queuing Protocol[6] (AMQP), a message publisher would be deployed on each data source and configured to publish information regarding malicious hosts to a central message broker. These message publishers were called rDSN modules and hard to be written for each of the different types of data sources. The first core component of the AR-Framework, namely the *Collection* Phase, was responsible for subscribing to the appropriate message queues on the RabbitMQ message broker. By using this publish/subscribe model, the *Collection* Phase could easily and in near real-time collect data from the various sensors. In addition to managing the various data sources, the *Collection* Phase was also responsible for filtering the traffic and events it captured, ensuring that only valid and appropriate data was used as targets for active fingerprinting through the AR-Framework. This was done in accordance with the types of traffic identified in Section 2.3, Potentially Malicious (PM), Truly Malicious (TM), and Result of Malicious traffic (RoM).

The Assessment Phase was responsible for aggregating data from the Collection Phase into sessions. This increased the efficiency of the AR-Framework, by ensuring that the same host would not be fingerprinted multiple times in short succession. Minimum and maximum session times may be specified in the configuration file, these values were used to determine the duration of sessions before they expire. During session management a unique Session ID would be created and this information would be passed along with relevant data captured by the Collection Phase to the *Alignment* Phase.

An overview of the *Assessment* and *Alignment* Phases were provided in Figure 4.3. The *Alignment* Phase managed the various reconnaissance modules that were used to analyse malicious hosts. A recon thread was spawned for each IP address supplied to the *Alignment* Phase from the session manager in the *Assessment* Phase. This allowed for the reconnaissance of multiple

[5]https://www.python.org/
[6]http://www.amqp.org/

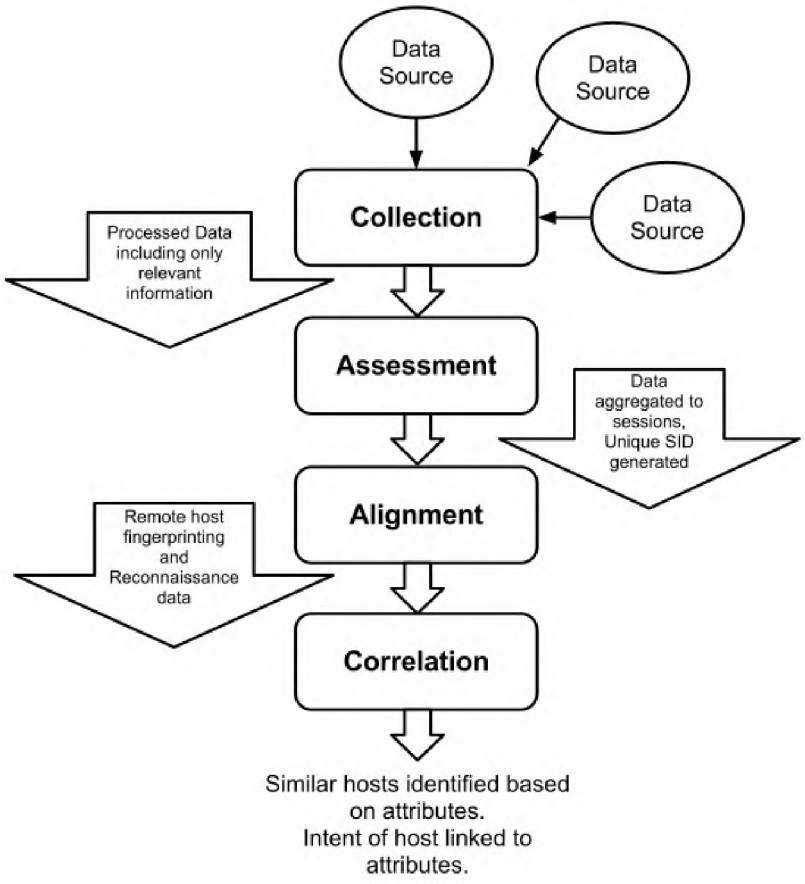

Figure 4.2: Conceptual overview of the AR-Framework showing

hosts to be achieved in parallel. Once a recon thread had completed its operation it would return that data to the Reconnaissance Manager which in turn would insert data retrieved by the recon thread into the AR-Framework database. This database was specified in the AR-Framework configuration file. The data was inserted into the *Alignment* table of the database, using the unique Session ID supplied by the *Assessment* Phase as a foreign key.

4.3 The Adapted Multisensor Data Fusion Model

Multisensor fusion techniques combine data from multiple sensors and related information from data sources to achieve greater confidence and accuracies in results and produce inferences that would not have been as reliable if produced from from a single sensors observations (Waltz and Llinas, 1990; Klein, 1999; Hall, 1992). Fundamentally data fusion involves a hierarchical transformation between observed parameters from multiple and often heterogeneous sources and a decision or inference regarding the characteristics, identity of an entity and interpretation of the observed entity in the context of the observed environment and the relationships to other entities therein (Hall and Llinas, 1997).

Bass (2000) observed how traditional military C2 (command and control) systems would make use of multiple field sensors to observe and measure data primitives such as radiation, acoustic, and thermal energy, nuclear particles, and other observable signals. The sensors would then be made use of in a data fusion process in order to correlate, aggregate and associate the data so that they may assist in a automated decision making process or support systems. While in principle the process of multisensor data fusion would provide significant advantages over the observations of a single data source, Hall and Llinas (1997) states that besides the statistical advantage gained by combining same-source data, the use of multiple types of sensors might in practice produce worse results than could be obtained by making use of a single more appropriate sensor for the application. Hall further explains that these sub optimal results are caused by an attempt to combine accurate with inaccurate or biased data.

In earlier work by Hunter et al. (2012b) an adaptation of the generic data fusion model by Waltz and Llinas (1990) was proposed. The adaption of the model was based on the work done by Bass (1999, 2000) in which he adapted the generic model and outlined a framework for next generation distributed intrusion detection systems. The AR-Framework model was adapted by including active response mechanisms to events, handling unsolicited network traffic as input and

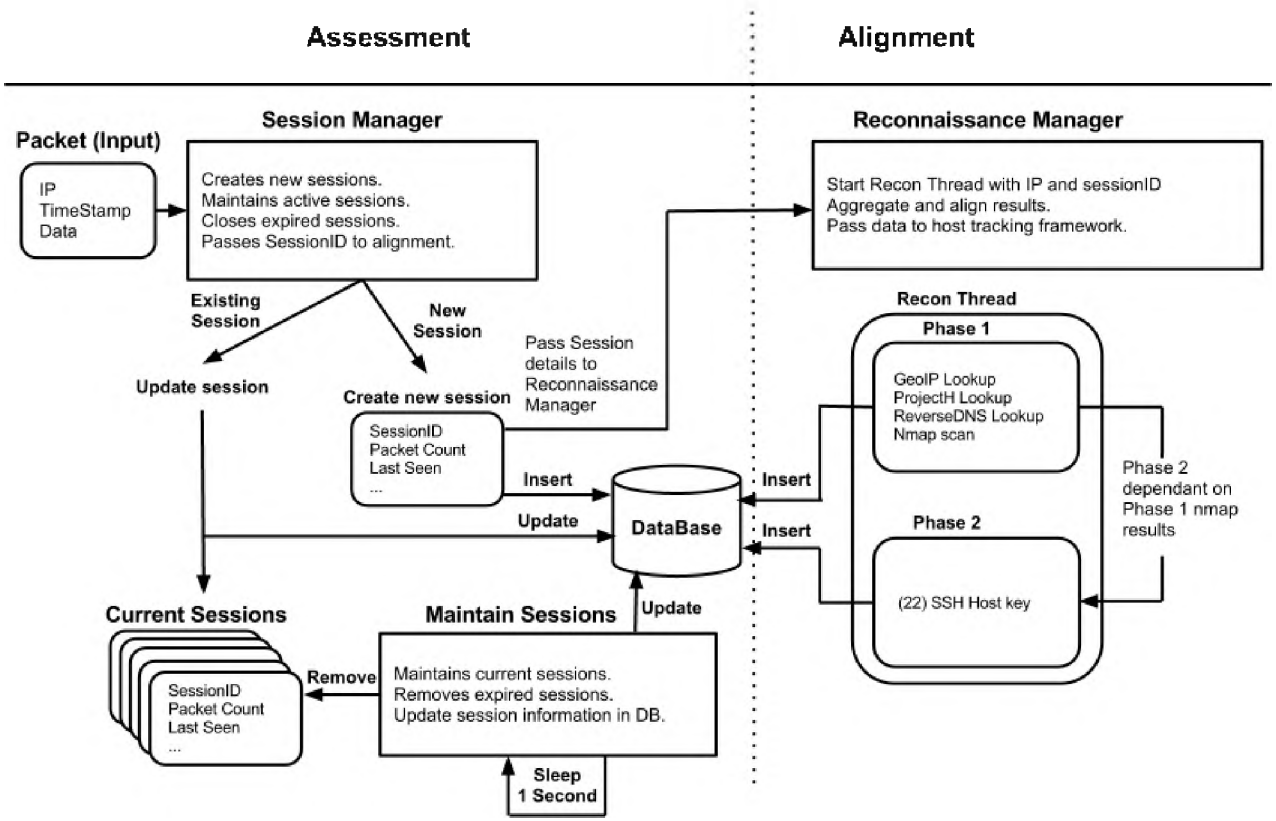

Figure 4.3: Assessment and Alignment Phases of the AR-Framework

generating situational awareness as output. Active reconnaissance of objects modeled in the data fusion process allowed for a more detailed and effective representation of those objects. The active reconnaissance included remote host fingerprinting and use data such as OSINT. The adapted model of multisensor data fusion was used as a reference point during the implementation of the AR-Framework. The model provided a high level representation of the information flow and control within the AR-Framework.

The data sensors used to capture unsolicited traffic, provided input to the multisensor data fusion process. This formed part of the Collection Phase of the AR-Framework detailed in Section 4.4. The alignment of data to a common frame of reference as well as data preparation was achieved during the *Assessment* Phase; discussed in Section 4.5. The active response (reconnaissance) mechanisms found in the model are detailed under the *Alignment* Phase of the AR-Framework in Section 4.6.

4.3.1 Data Fusion

Inputs to the data fusion process include sensor data outlined in Section 4.4, and supporting data that would be collected in near real-time through reconnaissance techniques such as port scanning and OS enumeration, shown in Section 4.6. This last type of support data was where the adaption of the generic multisensor data fusion model become most apparent. The process of multiple sensor fusion, according to Waltz and Llinas (1990), was concerned with the adaption, correlation, and analysis of data from different sources in order to evaluate a situation or event and then assist in decision making processes or initiate direct action. The evaluation of a situation occured naturally on honeypot data sensors as they emulate vulnerable services and monitor the interaction initiated with those services. As such honeypots were aware of the services being targeted by malicious agents. The concept of a network telescope was defined in Section 2.2.3, as such a network telescope was not intrinsically aware of the types of traffic it observed, even though it was mostly malicious. To this end a *Traffic Filter* class was needed. One of the functions of this *Traffic Filter* class was to make inferences on the traffic it parses, essentially mapping the port targeted, to the expected service that would normally run on that port. Figure 4.4 showed the adapted model for monitoring unsolicited network traffic to produce situational awareness, this model provided a starting point for the development of the AR-Framework.

Conceptually unsolicited traffic sensors, would provide raw data in the form of TCP or UDP packet data from network telescopes. Honeypots would provide similar data, such as targeted

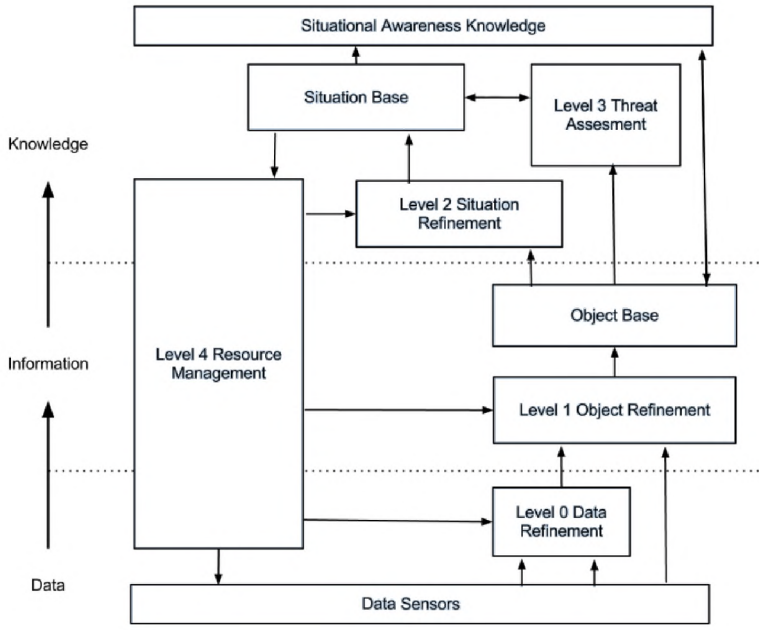

Figure 4.4: Unsolicited network traffic fusion for advanced situational awareness adapted from Waltz and Llinas (1990); Bass (1999)

port and source IP address, but no TCP or UDP header details. This data represented the unsolicited traffic input into the data fusion model. The multisensor data fusion model provided an abstract representation of data aggregation, alignment, and analysis by assigning different levels of processing. These levels form a hierarchy of processing that data undergoes in order to create knowledge. Level 0 data refinement was originally concerned with the calibration of equipment, sensor adjustment and filtering of data (Bass, 2000) such as outlier removal. The following points outline the different levels of processing within the data fusion process.

- **Level 0 data refinement** consists of determining the legality of the source IP address. In the AR-Framework this would be achieved by passing data to the *Traffic Filter* class. An example of data refinement includes the extraction and elimination of loopback source addresses (127.0.0.1) or legitimate hosts generating backscatter as a response to a DDoS attack (RoM Traffic). In the AR-Framework, level 0 data refinement forms part of the Collection Phase detailed in Section 4.4.

- **Level 1 object refinement** amounts to the processing of data received and aligning the data to a common frame of reference such as timestamp, sequence numbers, or location of

IP address through ASN.

  - This included the correlation of data, such as packets with the same source and classification of the object identity such as PM traffic, TM traffic and not RoM traffic as this had been discared at Level 0 data refinement. Level 1 object refinement represented the Assessment Phase of the AR-Framework, found in Section 4.5.

- **Level 1 object refinement** was further concerned with the creation of an object entity and the binding of characteristics to that entity in order to create an Object Base. This was achieved by associating unique ID's to each session being maintained by the AR-Framework. These session IDs (SIDs) also provide a reference for adding characteristics to a malicious entity as they were discovered through the remaining hierarchy of the data fusion process.

- **Level 2 Situational refinement** provided situational knowledge regarding the object base after it has been aligned, correlated, and analysed.

  - Here aggregated sets of objects may be detected by examining their coordinated behaviour, an example of this could be the observation of PM traffic in the form of a horizontal port scan across multiple sensors.
  - The Situational Refinement process takes into consideration the various sensors involved as well as the information provided by Level 4 resource Management, which constitutes any additional real-time data retrieval such as using fingerprinting techniques and searching for past discrepancies associated with characteristics of the Object Base. An example of these past characteristics associated with the malicious entity could be the association with a particular Fast-flux botnet, this example was detailed in Section 3.3.
  - Characteristics of objects that would be inspected at this level include common points of origin (source ip or address range), protocols, common targets, and attack rates.

- **Level 3 Threat Assessment** could make use of the relative time difference between detection of the SSH worm scanning the honeypot and the network telescope to infer the rate at which this host is scanning network ranges.

  - The Threat Assessment process was concerned with the possible threats that unsolicited traffic might pose and their implications, along with information from level 2 Situational Refinement, a Situation Base is constructed.

- **The Situation Base** would represent aggregated information from a single entity or multiple entities from all combined processes before in the hierarchy of multi sensor data fusion.

Thus the Situation Base would represent aggregated information that would constitute a small subset of growing situational awareness that would constitute a collection of intelligence from a malicious host demographic.

It is important to note that unlike the traditional application of multisensor data fusion, data will often only be observed by a single sensor or inversely be observed by multiple sensors but at disjoint points in time. To illustrate with an example, a host infected with an SSH worm scans network ranges in order to find further hosts to infect. It might scan a honeypot first and only a day later (if ever) reach one of the network telescope sensor ranges. The key to managing this disjoint information is stored in the level 4 Resource Management process, this part of the model was used to collect additional information and refer back to old data looking for correlating characteristics. Situational Refinement and Resource Management largely refer to the Alignment Phase and Correlation Phase of the AR-Framework. While the AR-Frameworks design was based on the adapted multisensor fusion model, its function does not always directly follow the model, nor does it include all of the functionality listed here. The purpose of the model was to provide a frame of reference for the development of the AR-Framework prototype; as well as documenting possible future extensions of the AR-Framework.

4.4 Collection

The *Collection* Phase of the AR-Framework was responsible for maintaining data from various sensors. Whenever these sensors captured data they would publish the data as a message to a RabbitMQ message broker. The DataSource Manager (part of the Collection Phase) subscribed to the queues on a RabbitMQ message broker that messages would be published to. As data was published to the RabbitMQ broker from sensors, that data would be pushed to the DataSource Manager in near real-time. The authors made use of the term "near real-time" to reflect within seconds, as opposed to microseconds. The data sensors that were used by the AR-Framework and subsequently managed by the collection component included, two dionaea honeypots and a network telescope operating over two separate /24 network ranges.

Data from the network telescope consisted of packet data, illustrated in Table 4.1. This data was serialised with the help of JavaScript Object Notation (JSON) encoding at the honeypot

Table 4.1: Attributes of interest from network telescope rDSN sensor

Timestamp	TCP Sequence Number
Source IP Address	Destination IP Address
Local Port	Remote Port
Protocol	Flags

Table 4.2: Attributes of interest from honeypot rDSN sensor

Honeypot rDSN data submission
Honeypot name
Source IP Address
Local Port
Remote Port
Connection Protocol
Connection Timestamp

and de-serialised by the DataSource Manager. JSON provided a simple data-interchange format. Data from honeypots were represented as Malicious events and are shown in Table 4.2.

AMQP messaging applications were developed and deployed on the monitored data sensors, these applications were called rDSN modules and published information to a central RabbitMQ broker. In addition to managing the data sensors, the Collection Phase was also responsible for filtering backscatter and invalid IP address locations. It was previously discussed in Section 4.3.1 how this type of filtering formed part of Level 0 Data Refinement during the data fusion process. The *Traffic Filter* class attempts to identify the local service being targeted on unsolicited traffic sensors by noting the local port that was targeted and identifying the most common service to run on the particular port. An overview of the collection component of the AR-Framework is given by Figure 4.5, responsible for managing data from sensors, traffic filtering and categorisation. Once data has been received and filtered, it is passed to the *Assessment* Phase which is responsible for session management. The remainder of this Section will detail the design considerations of the network telescope rDSN module, the honeypot rDSN module and *Traffic Filter* class.

4.4.1 Network Telescope Data Source

The ability of network telescopes to capture traffic destined for unused address space is particularly attractive when considering that no legitimate traffic should exits on an unused range. By capturing only unsolicited traffic, network telescopes could be used to identify the source IP addresses of malicious hosts on the Internet, with the exception of reflected traffic that was previ-

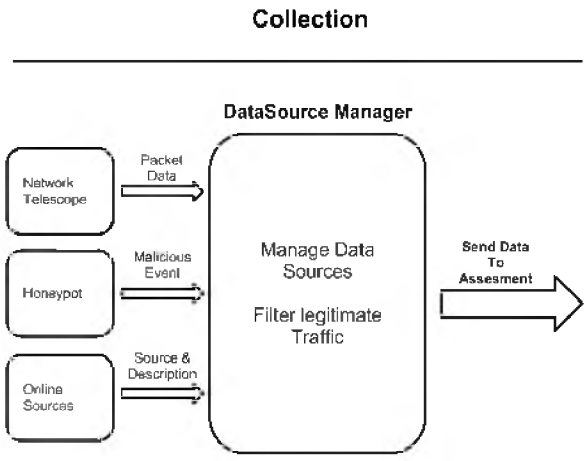

Figure 4.5: Conceptual representation of Collection Phase

ously defined as a Result of Malicious traffic in Section 2.3.4. It was for this reason that network telescopes were chosen as one of the data sensors to be incorporated into the AR-Framework. As a proof of concept, a single network telescope was used for data collection. The network telescope that was used is situated at Rhodes University and will be referred to as RUScope. RUScope listened on two /24 network blocks in South African IP address space. As a result of capturing traffic from two separate /24 networks, the AR-Framework is often referred to using two network telescopes, while conceptually this was true, from a deployment and configuration standpoint, there was only one network telescope. The network telescope was used to provide primitive observations, the main purpose being the capture of data that could be used to identify malicious hosts for fingerprinting through the AR-Framework. Attributes from the traffic that were important have been listed in Table 4.1. These are also the attributes that were published to the RabbitMQ message broker for use by the AR-Framework.

While all of the attributes listed in Table 4.1 are used when analysing data from a typical network telescope, the Source IP address is of most interest in the AR-Framework. As the framework was responsible for the active and passive intelligence gathering of potentially malicious hosts, there was no need for a deep packet analysis of the actual traffic being captured by the network telescope. While backscatter was observed by the network telescope it was important to differentiate it from malicious traffic. As backscatter traffic originates from legitimate hosts who have been the target of malicious behaviour, this traffic was filtered by the *Traffic Filter* class

introduced in Section 4.4.3 as part of the *Collection* Phase.

Traditionally the packet data collected by network telescopes would be inserted into a database at periodic intervals or written to disc as pcap files. There was no real need for real-time analysis of the traffic, as the analysis was concerned with the types of traffic observed, their relationships, and the quantities thereof. As the AR-Framework was concerned with the fingerprinting and data collection of malicious hosts connected to the Internet, it was important that the information retrieval process is started as soon as a malicious host it detected.

An AMQP messaging client (rDSN module) was developed and deployed on the RUScope network telescope. This messaging client's primary purpose was to act as a packet sniffer and publish this data as it was received in real-time. This was achieved by making use of AMQP and a RabbitMQ broker. A publish/subscribe model was chosen as it allowed for scalable data exposure and could assist in distribution work to other analysis modules if need be. The rDSN module made use of a configuration file which allowed for modular configuration and deployment across muliple network telescopes.

The rDSN module could be configured to listen on specific network interfaces, which RabbitMQ server to publish to, the exchange to publish data, verbosity levels, and different Berkley Packet Filtering (BPF) rules for packet capture. Packets captured on the listening interface would be inspected and turned into objects that could be represented in the Python programming language. After transformation, the object data would be serialised using JSON encoding and published to the exchange on the configured RabbitMQ message broker. The exchange used for publishing network telescope data was named 'telescope'. In addition to the object data that was published, the name of the network telescope the data originated from was also included. This was done in order to allow for the deployment and use of multiple network telescopes as data sensors for the AR-Framework.

A reconnect function was built into the rDSN module. Upon failing to connect to the RabbitMQ message broker, the application would enter a "Reconnection State". During this time the application would attempt to re-establish a connection to the RabbitMQ broker at 10 second intervals. Any packet data sniffed during this period of time would be discarded. If the AR-Framework was deployed on a permanent basis and with more resources available for fingerprinting it is suggested that a local database on the network telescope be used to cache data collected during the "Reconnection State". While AMQP has built-in controls that may be configured to act as a buffer, this is not advised when working with the large amount of data that

could be collected by a network telescope.

4.4.2 Honeypot Data Source

For the purpose of this research a medium interaction honeypot was chosen. The honeypot software that was used is called Dionaea, which is the successor to the popular nepenthes honeypot (Dionaea Project, 2012). Dionaea may be configured to emulate vulnerable services such as Microsoft SQL server, Postgress, MySQL, SSH, FTP, and even respond to unknown protocols by replaying incoming traffic to an attacker (Dionaea Project, 2012).[7]Dionaea stores all interactions initiated with it in a local sqlite3 database.

Two Dionaea honeypots were deployed as data sensors for the AR-Framework. The first was setup on an Amazon EC2[8] micro instance. However, due to problems with traffic tunneling to reach the RabbitMQ broker which was sitting behind a firewall, the EC2 Honeypot was only used for a short duration of time. The second was deployed on a VPS donated by a South African hosting company[9]. Both deployments of Dionaea were left with default configurations. An rDSN module was also deployed on each honeypot to publish data to the RabbitMQ broker. The honeypot rDSN module was very similar to network telescope rDSN module, the only difference was the mechanisms for data acquisition. Instead of sniffing packets on an interface, the honeypot rDSN module would query the local Dionaea sqlite3 database, keep track of the last data it published, and send any new data contained in the database. The database was queried at 10 second intervals, allowing for a two second travel time for a message to be published on the RabbitMQ broker and another two seconds for it to be pushed to all subscribers (the AR-Framework), after detection, it should never take more than 14 seconds for an event to reach the AR-Framework. This allowed the reconnaissance of malicious hosts to begin in near real-time.

Data of interest that was submitted to the RabbitMQ server from the honeypot based rDSN module have been listed in Table 4.2. While Dionaea does support the Extensible Messaging and Presence Protocol (XMPP) for sharing connection data and captured binaries, it was decided to use AMQP and the RabbitMQ architecture in favor of XMPP. The motivation for this decision came from the fact that the rDSN module developed for network telescopes was already making use of AMQP and there would be no advantage to setting up an additional server for XMPP. As an alternative to querying the dionaea database for new connections, an ihandler could have been

[7]http://dionaea.carnivore.it/
[8]http://aws.amazon.com/ec2/
[9]http://www.elitehost.co.za/

written to publish data as soon as a new connection is made with the honeypot. Dionaea makes use of a service called ihandler to hook into different event calls, it is possible to write custom ihandlers for logging or interacting with new protocols. While the use of an ihandler would be the more elegant solution, it was not achieved during the course of the research.

4.4.3 Traffic Filtering

A *Traffic Filter* class was used to identify and extract particular types of network traffic from the network telescope data source (Section 4.4.1). As mentioned before in Section 2.1, a network telescope is capable of capturing RoM traffic in the form of backscatter from certain types of DDoS attacks. In the event of this backscatter traffic being detected, the AR-Framework should not run any reconnaissance modules against the source of the traffic as it would represent the victim of an attack.

For the purpose of this research a simplified approach was taken to detect RoM traffic from random source SYN based distributed DDoS attacks. Any TCP based SYN-ACK packets received by the network telescope would be categorised as RoM backscatter traffic during the Collection Phase. The source IP address of this traffic would not be passed to the Alignment Phase for reconnaissance. This filtering of backscatter traffic forms part of the Level 0 Data Refinement process outlined by our adapted multisensor data fusion model in Section 4.3.1.

In addition to backscatter exclusion, the *Traffic Filter* class may be further configured to identify possible intent from network telescope traffic. This is achieved by inferring the intent of traffic based on the ports it is targeting. It is not very likely that a malicious agent would attempt a vertical portscan of an IP address within a network telescope's operating range. In most cases a vertical portscan would be performed by a human adversary in a targeted attack. It is more likely to observe a horizontal scan whereby the malicious agent is looking for only a single or very small number of services to exploit across multiple hosts. Horizontal port scans are far more likely to occur in general as they are employed by self propagating worms. Common services that are targeted include web services, database services, FTP, RPC services, SMB, and SSH.

Based on this information, the *Traffic Filter* class was configured to infer the intended service that was targeted based on the port number being targeted. The *Traffic Filter* class would pass information from the AR-Framework through a call back function used to register new packets from the RabbitMQ server. This would happen before the Session ID was generated or the IP

address of a malicious host is passed to the Alignment Phase. As a result it was possible to exclude backscatter traffic and allow for the inference of targeted service, which would then be inserted into the AR-Framework database after the Session ID was generated during the Assessment Phase.

4.5 Assessment

The *Collection* Phase would receive packet or event information from the various data sources used by the AR-Framework. After traffic filtering, as discussed in Section 4.4.3 the information would be passed to the *Assessment* Phase. The *Assessment* Phase was responsible for the session management of network telescope data and honeypot events based on source IP address and the duration of activity. Figure 4.6 provides an overview of the *Assessment* Phase. A session was considered to have a minimum and maximum lifespan and based on these pre-determined limits, sessions would be maintained or expire. A session threshold would be configured in the AR-Framework configuration file under the assessment section. If an event or packet did not belong to any of the existing sessions being maintained by the Assessment Session Manager, a new session was created. As part of the creation process a unique Session ID (SID) was associated with the session, along with attributes such as Source IP address and a time-stamp of the event or packet. This data would be inserted into the sessions table contained in the AR-Framework database using the SID as a primary key. Once a new session had been created, the source IP and SID would be passed to the *Alignment* Phase for reconnaissance.

4.5.1 Session Management

The Session Manager was responsible for maintaining active sessions, creating new sessions and expiring in-active sessions. A session was removed from the active sessions list whenever it satisfied any of two requirements. The requirement was no packets or event data for the period stipulated by the minimum session threshold in the AR-Framework configuration file. If the duration of time since the last received packet (network telescope) or event (honeypot) exceeded that threshold, the session would be removed from the current active session list. Similarly if the duration between the first observed packet or event and the current time exceeds the maximum session threshold, that session is terminated and removed from the active sessions list. This second requirement was imposed to stop excessively long sessions from permanently being open. As information

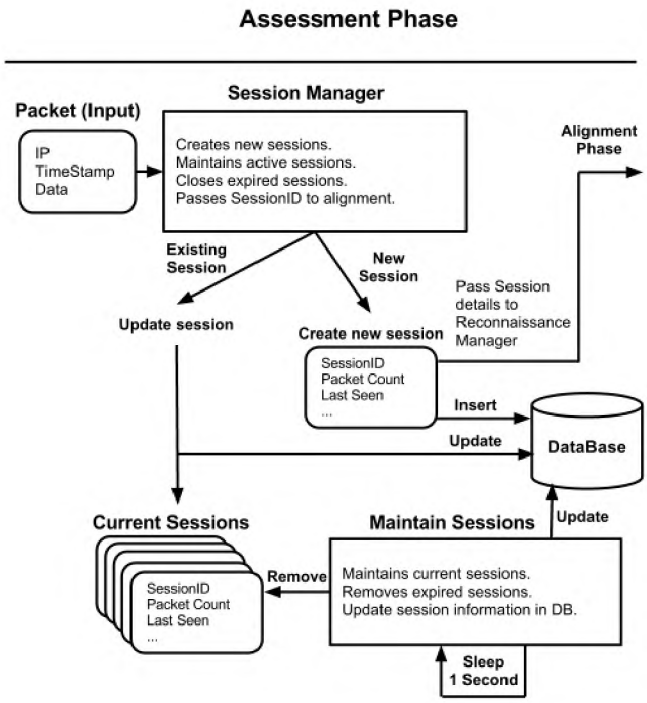

Figure 4.6: Assessment Phase of the AR-Framework, responsible for session management

regarding active sessions are updated whenever new data pertaining to that session is observed and the active sessions are stored in memory until they expire, it could lead to performance and stability problems if sessions where kept open indefinably. Information pertaining to currently active sessions that were updated as new packets or events were observed included packet or event count and the last seen time-stamp.

While the original time-stamp of packets and events were recorded, new timestamps would be generated and used to maintain current sessions. This was done in order to account for data sources producing packets and events from different time zones, which would otherwise impact the session duration thresholds. This method of maintaining session time also allowed for the replay of previously captured pcap files as a possible data source to the AR-Framework. Time-stamp management also formed part of Level 1 Object refinement, introduced in Section 4.3.1. The Session Manager class made use of a thread to periodically check each of the currently active sessions to determine if any of them had expired. As the list of currently active sessions would be accessed by various functions (threshold checks, inserting new sessions, removing expired sessions) it was important to make use of thread locks and avoid race conditions.

In essence the Assessment Phase acted as a buffer between data being passed from the Collection Phase to the Alignment Phase. There would be no need to run all of the fingerprinting modules managed by the Alignment Phase for every packet with the same source IP address. Thus the Assessment Phase reduced the IP data passed to the Alignment Phase to only once for each IP address during its session duration.

4.6 Alignment

The *Alignment* Phase of the AR-Framework was responsible for the active and passive information retrieval with regards to malicious hosts. This information retrieval process was also called reconnaissance, the *Alignment* Phase made use of a Reconnaissance Manager class to maintain currently active recon threads. A recon thread would be spawned whenever the Reconnaissance Manager received a new IP address from the *Assessment* Phase. Currently the Reconnaissance manager and recon thread class only support the fingerprinting of IPv4 addresses. Figure 4.7 provides a conceptual overview of the *Alignment* Phase components and the processing of information therein.

The recon thread was responsible for both passive and active information retrieval. Passive

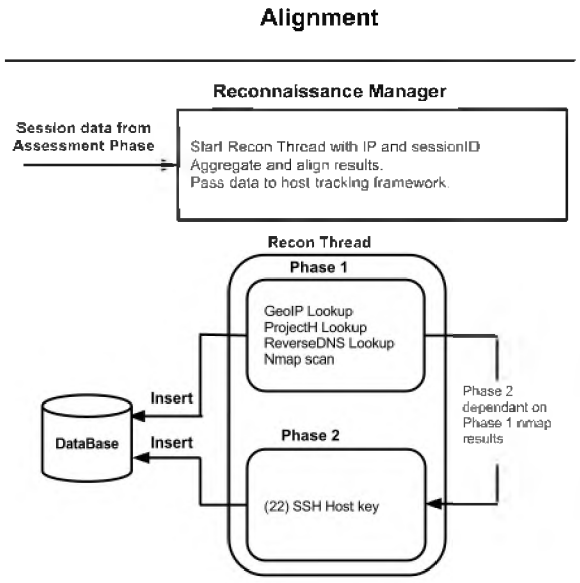

Figure 4.7: Alignment Phase responsible for managing recon threads and correlation of data

information retrieval included activities such as performing a lookup of geographic information related to the IP address or querying OSINT services for previously observed activity and reverse DNS entries. Active information retrieval would involve an Nmap port scan which included the top 1000 most common ports, perform OS enumeration and collection of additional information such as SSH Host key and Uptime, if possible.

Once the initial fingerprinting modules had completed, a LBM fingerprint would be performed. The LBM fingerprint was intentionally left to run last as other fingerprinting techniques could interfere with the latency measurements. When the AR-Framework was first tested, LBM fingerprinting ran on every host, however, it was soon discovered that the ping servers that were used as base stations were becoming overloaded with requests. As such the number of LBM fingerprinting scans had to be scaled down, to this end only hosts identified through honeypots were submitted for LBM fingerprinting; at an average rate of four an hour. When moving to a production environment, scalability could be easily achieved through the deployment of additional VPS instances in the same geographic area. The information gathered by the a recon thread would then be inserted into the AR-Framework database, under the Alignment table. The Session ID generated by the Assessment Phase would be used to associate the collected information with an IP address, which would correspond to a unique session.

The *Alignment* Phase may also be configured through the AR-Framework configuration file. These variables allow for the selection of fingerprinting modules that should be executed. Thus if researchers did not wish to run any active information retrieval modules, such as an Nmap scans they could be omitted.

4.6.1 Open Source Intelligence

OSINT refers to information collected and available through public sources. The AR-Framework and more specifically the Alignment Phase and recon threads made use of OSINT sources as part of the information gathering and fingerprinting process. The OSINT sources used by the AR-Framework include a IPv4 to geolocation lookups (MaxMind), previously observed malicious activity (projecthoneypot.org) and reverse DNS queries.

The process of translating an IP address to geographic data such as country, city, and co-ordinate system is known as geolocation, the primary sources for this data are regional Internet registers. These registers allocate and distribute IP addresses amongst organisations that are located in their respective regions. The registers include AfriNIC[10], ARIN[11], APNIC[12], LACNIC[13], and RIPE NCC[14]. The geolocation for the AR-Framework was achieved through the use of a third party service known as MaxMind. MaxMind provides both free and paid for geolocation based services with varying degrees of accuracy for the different services . The AR-Framework used the free version of the MaxMind GeoLite City database, a binary version of this database was used as it allows for the fastest queries. The binary database was accessed through a python based Application Program Interface (API). The geolocation allowed for the enumeration of country, city, latitude, and longitude based on a given IPv4 address.

The project honeypot website provides access to lookup system from data collected by distributed honeypots. This allows for identification of spammers, spam bots, brute force attackers, and comment spammers based on their IP address. As part of the information gathering and fingerprinting process of the AR-Framework, Project Honeypot was used to gain additional information regarding malicious hosts. If the Project Honeypot database contained information regarding a host observed by the AR-Framework it provided historic data on what a given host has been observed doing in the past. Project Honeypot collect their data by deploying custom

[10]http://www.afrinic.net/
[11]https://www.arin.net/
[12]https://www.apnic.net/
[13]http://www.lacnic.net/web/lacnic/inicio
[14]https://www.ripe.net/

honeypot software on their own networks and volunteer networks.

Reverse DNS lookups where used to determine the domain name associated with a given IP address, if it was resolvable. The most common usage for DNS is known as forward DNS resolution, whereby an IP address associated with a given domain is determined. Requests to determine the domain associated with a given IP address are made to the the Address and Routing Area (ARPA) top level domain of the Internet. For IPv4 addresses the in-addr.arpa domain would be used, queries for IPv6 addresses are made to ip6.arpa. As stated earlier the AR-Framework only supports the fingerprinting and information gathering of IPv4 addresses. The in-addr.arpa domain contains entries for IPv4 addresses which are represented as concatenated sequences of four decimal numbers that are separated by dots and appended with a second level domain suffix. The four decimal numbers represent the IP address of a given host in reverse order, while the second level domain suffix would be in-addr.arpa or a generic reverse dns name assigned by an ISP. The second level domain suffix is useful to group malicious hosts by their ISP (*Associative* fingerprint), thus allowing researchers to identify particularly noisy or malicious Internet address space as the responsibility should fall on an ISP to monitor excessively malicious hosts on their network. As such it also provides information regarding whom to contact when reporting malicious behaviour.

The major advantage of making use of OSINT sources such as these is that they provide passive information. That is to say, the information gathering process takes place without the host being targeted becoming aware of the actual process. The geolocation, reverse DNS, and Project Honeypot look-ups would all be performed without having to send a single packet of data to the host being fingerprinted. These sources provided a wealth of knowledge regarding malicious hosts, were freely available and had no apparent reason or motivation for obscuring results.

4.6.2 Reconnaissance Manager

The Reconnaissance Manager class was tasked with spawning recon threads, maintaining data flow between Assessment and recon threads as well as aggregating data from recon threads, and inserting their completed fingerprinting information into the AR-Framework database. A publishing queue was passed by reference to each of the spawned recon threads so that they could place the information they gathered into the queue. On receiving new data in the queue the Reconnaissance Manager would re-construct a host object and insert the relevant data regarding

that host into the database, using the Session ID as a foreign key. This process was part of Level 1 Object Refinement and Level 2 Situational Refinement, all of which was conceptually controlled by Level 4 Resource Management (Figure 4.4). This formed the main program loop of the Reconnaissance Manager thread. RUscope provided upwards of a thousand unique IP addresses for fingerprinting over a given one hour rolling window. The large number of requests this would require to be directed to Project Honeypot and the free ping services used for the collecting latency measurements for LBM fingerprinting would result in violation of their acceptable use policies. As such it was decided to only perform project honeypot lookups and LBM fingerprinting on malicious hosts identified through the use of honeypots. Each honeypot only observed an average of four malicious hosts per hour.

4.7 Visualisation Tools

Data visualisations are used to convey knowledge and context to an observer, they often act as an aggregation mechanism to provide an overview of data. The process of creating a data visualisation firstly involves the collection of data, such as through network telescopes, honeypots and the AR-Framework. This data would then be processed, analysed, and aligned to some frame of reference, during this process data is turned into information. Finally, this information would be presented to give insight. It is important that data visualisation avoid information overload, which occurs when an observer is bombarded with information and unable to understand what the information means or the context it is in. This is often a shortfall for most visualisations as they either provide too much data, provide data without context, or provide too low a level of information that is not useful to the observer.

The visualisation tools and aggregated metrics dashboard presented in this Section were created to present fingerprint data collected by the AR-Framework from each of the four categories of fingerprinting that were identified in Section 3.2 Logical, Physical, Associative, and Behavioural. The motivation for these tools came from the vast number of malicious hosts detected by the network telescopes and honeypots and subsequent data collected by the AR-Framework. There was a need to investigate this information and present it in an easy to understand manner. The near real-time visualisations (GeoViz and GeoTimeline) were created to provide a dynamic overview of malicious hosts as they are detected and fingerprinted by the AR-Framework. The objective was to provide a high level overview of current malicious host activity but also in a context of

previously observed hosts. A geographic map was chosen as the base of these visualisations. This allowed for a perspective of location with reference to hosts detected by the data sources previously discussed in Section 4.4.

Both the visualisations and aggregated metrics dashboard detailed in this Section were capable of displaying information as a delayed "live" feed. The reason for the delay between detection and presentation was the need for the reconnaissance modules from the AR-Framework to finish collecting data. As such the visualisations show data in near-realtime. There was a further delay in presenting data for the GeoViz visualisation illustrated in Figure 4.8, as the visualisation was only updated with new information at 5 second intervals, this was done to eliminate potential information overload. The GeoViz visualisation was first presented in Hunter and Irwin (2011).

Figure 4.8: GeoViz visualisation with coloured markers indicating age of malicious host observation and info windows showing malicious host characteristics

4.7.1 GeoVis

GeoViz acted as a information dashboard providing an overview of aggregated data as it would be received by the AR-Framework in near real-time, allowing for data collection delay and presentation delay. With reference to the four identified categories, GeoViz provided the location of malicious hosts (*Physical*), their OS and open ports (*Logical*). The visualisation also provided a qualitative representation of time through the use of coloured markers indicating the time delta since observed.

In order to minimise the usage and access time on the AR-Framework database a local sqlite database was used and populated with data from the AR-Framework as it was collected. This local database served as a caching mechanism or buffer between the AR-Framework database and the visualisations. The visualisation checked for new entries in its local copy of the database at 5 second intervals. It was also important to show historic information to provide a context for new data. However, care was needed to avoid information overload, thus GeoViz made use of small colour coded map markers to indicate age. The latest 25 hosts to be added were portrayed with green markers, the next 150 hosts used yellow markers, while the oldest 300 hosts were represented with gray markers. Figure 4.9 presents the GeoViz visualisation with only aged markers and all info windows closed.

Figure 4.9: GeoViz with no *info windows* open, only showing markers representing the locations of malicious hosts and relative age of detection

For the visualisation to remain dynamic, a method was required to depreciate the value of older, less relevant information. This was accomplished by limiting the detail of information that is displayed to a user depending on the age of the data relative to new observations. The near real-time visualisation displayed data gathered by the AR-Framework in a First in First Out (FIFO) manner with regards to the coloured markers and when they change. The most recent hosts should be the focus of the visualisation while the older hosts form part of the background and the bigger picture.

The Google maps API engine was used to build both GeoViz and GeoTimeline. Each of

these markers had an *info window*[15] which would be displayed as the marker was added to the visualisation, with a maximum of five automatically opened *info windows* as they were added to the map. These *info windows* could be opened and closed by clicking on a marker. When new markers were added, the older markers *info windows* would close. An example of an *info window* displaying malicious host data is given in Figure 4.10. These *info windows* contained information regarding the host as was obtained by the AR-Framework which included OS, OS certainty, open ports, date and time of observation.

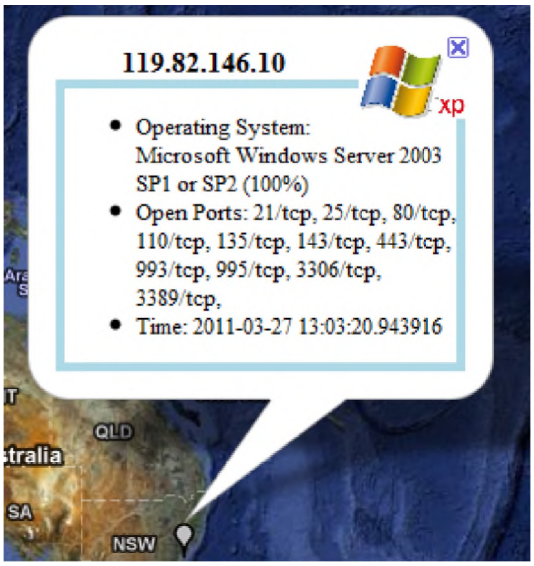

Figure 4.10: GeoViz *info window* for malicious host in Australia, showing OS, open ports, date and time of observation

Hosts that had very recently been added to the visualisation would show more details (IP Address, OS, open ports, and time detected), through the use of an *info window,* while the older hosts would leave focus and only be represented by their coloured map marker. A vast amount of data was contained within the AR-Framework database, more than enough to saturate any size of screen. With this in mind, the visualisation was designed so that this information could be accessed at any time, by allowing the user to click on markers in the background to open their respective *info windows.* The GeoViz visualisation illustrated in Figure 4.11 shows several aged markers and details pertaining to four malicious hosts, the problem of information overload or efficiency at displaying large amounts of information can be seen in this figure as the 5th malicious host's information is concealed.

[15]https://developers.google.com/maps/documentation/javascript/examples/infowindow-simple

Figure 4.11: GeoViz visualisation showing differently aged markers and four *info windows* with malicious host data with OS family icons

4.7.2 GeoTimeline

The second visualisation, GeoTimeline was an extension to GeoViz which allow a user to 'playback' events as they were observed by the network telescopes and honeypots, but aligned with data from the AR-Framework. This was achieved through the addition of a timeline navigation bar. For the GeoTimeline visualisation, a JavaScript library called timemap[16] was used to incorporate a SIMILE timeline[17]. With reference to the four categories, GeoTimeline provided the location of malicious hosts (*Physical*), their OS (*Logical*) and the previously observed intent of the host (*Associative*), which was obtained from Project Honeypot[18], when data was not available from Project Honeypot, the targeted port would be included instead. The functionality to playback collected data was possible by navigating a timeline in the user interface, demonstrated in Figure 4.12.

[16]https://code.google.com/p/timemap/
[17]http://simile-widgets.org/
[18]https://www.projecthoneypot.org/

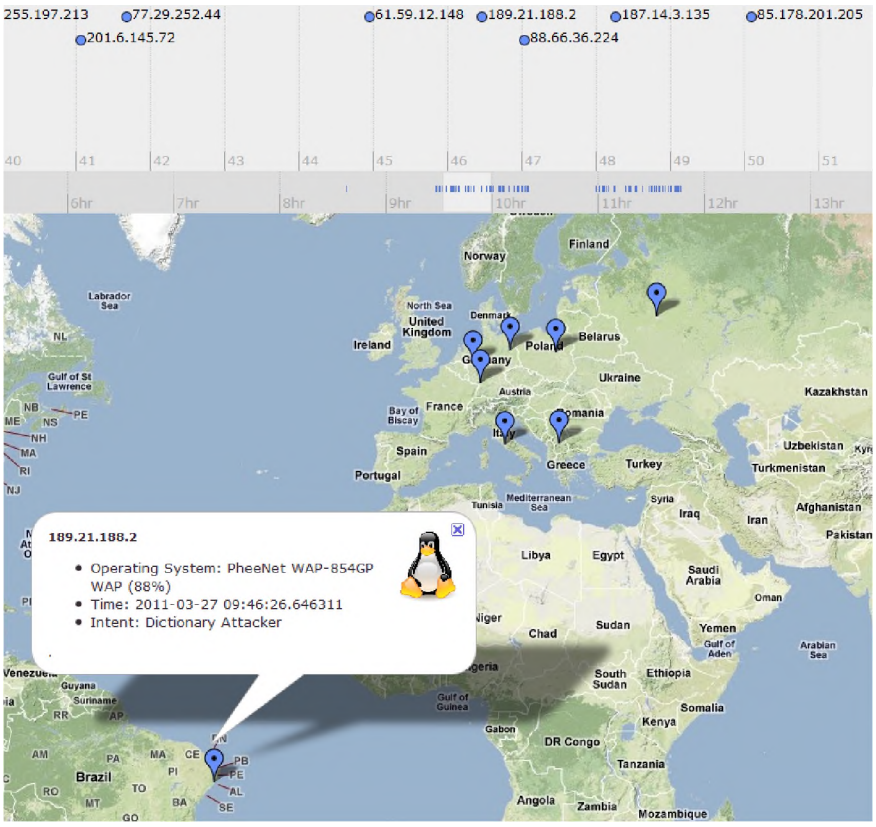

Figure 4.12: GeoTimeline visualisation showing information on a malicious host

4.7.3 Aggregated Metrics Dashboard

The GeoViz and GeoTimeline visualisations provided an overview of data, based around the geolocation of IP addresses. While this worked well to provide "at a glance" information, it did not summarise information gathered by the AR-Framework. To address this an AR-Framework Dashboard was created to provide summarised information in the form of an aggregated metrics dashboard. This included information such as Top n metrics, search functionality, and row by row data for certain malicious host attributes collected by the AR-Framework. The aggregated metrics dashboard was created to summarise AR-Framework data with a Top n metrics approach, where n is the number of aggregated group, in descending order, to summarise data for specific questions. The AR-Framework dashboard pulled data directly out of the AR-Framework database, giving access to new data as it was obtained by the AR-Framework. Figure 4.13 shows the dashboard when viewing the top number of services targeted. The services are determined by the port,

however, if the port is not linked to a well known service, it displayed "Unknown". The AR-Framework data depicted in Figures 4.13, 4.14 and 4.15 only made use of two honeypot data sensor; elitepot and EC2Honeypot. Each of the Figures demonstrates a search bar to filter for specific attributes as well as columns that could be order in an ascending or descending manner. The number of entries returned from the AR-Framework database, for each of the categories could also be adjusted from 10, 100, 1000, or all entries through the user interface.

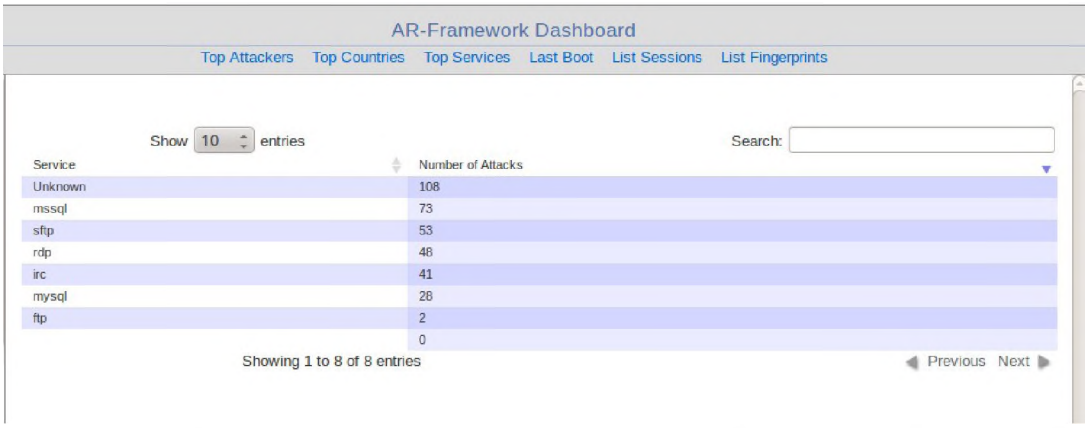

Figure 4.13: Portion of AR-Framework dashboard showing the top services that were targeted

Figure 4.14 shows the entire AR-Framework dashboard, on the left a "latest sessions" view shows the latest malicious host sessions detected by the AR-Framework's rDSN. As data is extracted directly from the AR-Framework database, new entries are added whenever the page is refreshed. On the right of Figure 4.14 is a list of malicious host fingerprints, the attributes detailed include Country and City which represented the *Physical* category of fingerprinting, Project Honeypot entry and reverse DNS representing *Associate* fingerprint, OS representing *Logical,* and last boot representing the *Behavioural* category.

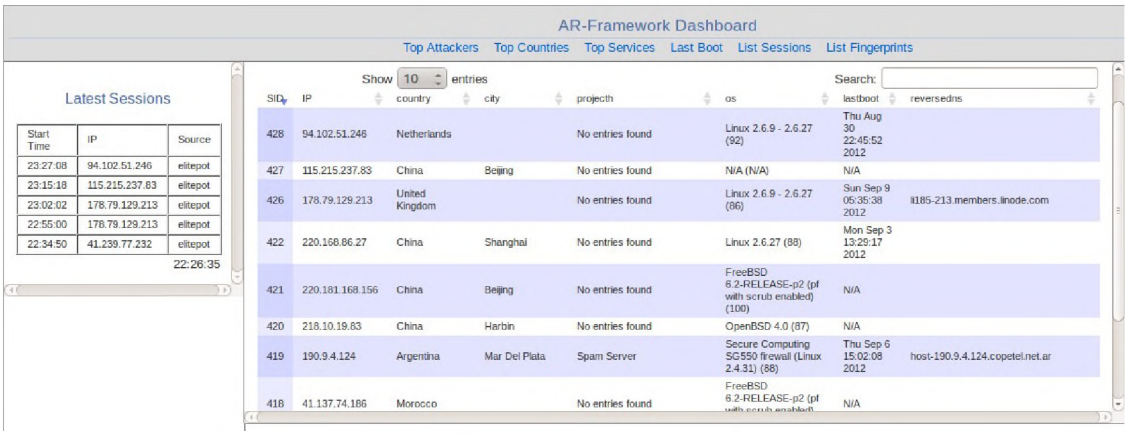

Figure 4.14: Dashboard showing the latest malicious host sessions on the left and the list of malicious host fingerprints on the right

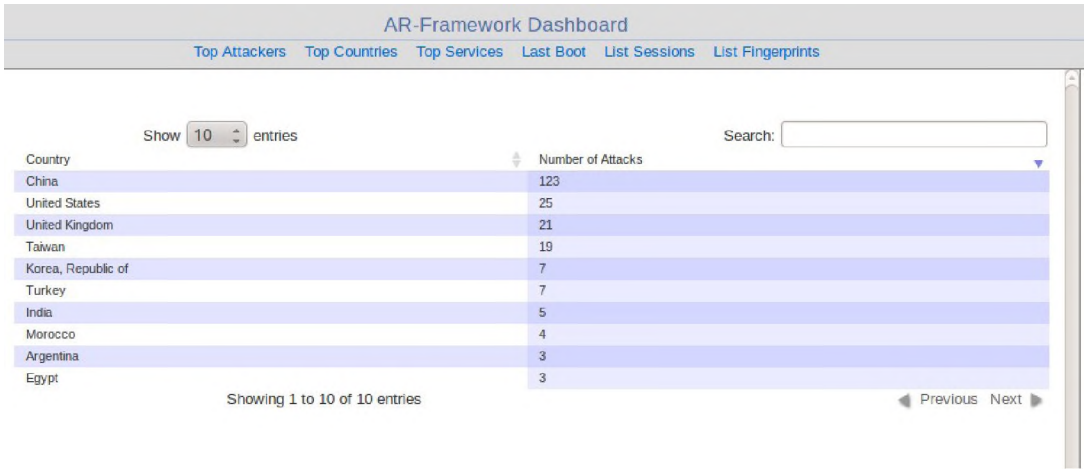

Figure 4.15: Number of attacks based on country

The two visualisation tools and aggregated metrics dashboard provided not only near real-time information exposure, but also allowed for interaction to explore the data over time and inspect aggregated information. They allowed a user to inspect data that had been collected and aligned from multiple heterogeneous source such as network telescopes, honeypots, and the AR-Framework. The tools demonstrated the usefulness of being able to aggregate information and provide a means to easily interpret large amounts of information.

4.8 Summary

This Chapter presented the adapted multisensor data fusion model alongside the design concepts of the AR-Framework prototype. First the limitations and assumptions that would be required of data collection and operation of the AR-Framework were discussed. This included considerations for the accuracy of OSINT sources, the need for near-realtime fingerprinting and limitations to the type of fingerprinting that could be legally performed. The adapted multisensor data fusion model was presented. This model was used for the design and implementation of the AR-Framework and assisted in modeling the aggregation of heterogeneous data sources. It made use of different levels of processing throughout the lifetime of information flow, this incorporated the alignment, analysis, and merging of data.

The four main components of the AR-Framework were then discussed, these components included three phases: *collection*, *assessment*, and *alignment*. Collection represented the rDSN modules that were deployed on the network telescope and honeypots, along with the *Traffic Filter* class that was responsible for discarding legitimate hosts from reconnaissance in the AR-Framework. The *assessment* phase was tasked with session management of incoming malicious host traffic, maintaining and expiring sessions according to configurable variables in the AR-Framework configuration file. As such, the assessment phase would control which IP addresses were passed to the *alignment* phase for reconnaissance. The *alignment* phase incorporated the actual reconnaissance functionality of the AR-Framework prototype. A reconnaissance manager class was used to spawn recon threads and correlate data returned by them. These recon threads performed both active and passive information collection using sources such as MaxMind, Project Honeypot, Nmap, and LBM fingerprints. The fingerprint data collected through this process represented the four categories of fingerprinting identified in Section 3.2.

The information collected by the rDSN modules and AR-Framework were then presented through the development of two visualisation tools (GeoViz and GeoTimeline) and an aggregated metrics dashboard. All three of these tools afforded the user various levels of interaction and insight into the data that had been collected from multiple sources. Their particular usefulness was illustrated by their ability to aggregate information and provide a means to easily interpret a large amount of information without information overload.

The next Chapter will present, in more detail the data collected through the deployment of rDSN modules and AR-Framework. The Chapter starts with a LBM feasibility study in Section 5.2, detailing the the response rate of malicious and legitimate hosts to LBM fingerprinting. The

feasibility study then concludes by presenting the results of host re-identification based on LBM fingerprints.

The remaining Sections of Chapter 5 will then present data collected from malicious hosts on the Internet, detailing malicious host characteristics. This includes a discussion of the geopolitical distribution of malicious hosts based on the data collected from three different rDSN modules in Section 5.3, using OSINT sources discussed earlier in the Chapter. The investigation continues in Section 5.4 with a discussion of the most targeted ports across the different sensors, this also lead to the observation of the Carna botnet which operated during 2012. The results then move from data collected by the rDSN modules to data collected by the AR-Framework, with a discussion of the most common operating systems found on malicious hosts in Section 5.5, as detected through the application reconnaissance manager and recon threads detailed earlier in this Chapter. Chapter 5 will conclude with a case study of the open ports found on malicious hosts, including the most common open port combinations in Section 5.6.

There is nothing like looking, if you want to find something. You certainly usually find something, if you look, but it is not always quite the something you were after.

J.R.R. Tolkien - The Hobbit

5 Results

This chapter presents the data collected through the deployment of rDSN modules and the AR-Framework described previously. The chapter begins with an overview of the datasets collected throughout this work along with the time-frames of collection. The remainder of the chapter consists of five sections discussing various aspects of the data that was collected. This chapter serves to illustrate the successful application of rDSN modules and the AR-Framework in the real-world.

Chapter Overview

- Section 5.1 will provide an overview of the datasets that were collected using the AR-Framework. These datasets were analysed to generate the results presented in the remaining sections of this Chapter.
- A Latency Based Multilateration feasibility study will be presented in Section 5.2. The

study investigates the likelihood of receiving responses from ICMP echo requests, from both malicious and legitimate hosts on the Internet. Results from host re-identification based on LBM fingerprints with a scaling threshold variable will also be presented.

- A discussion of the geopolitical distribution of malicious hosts will be given in Section 5.3. This discussion focused on the comparison of the origin of malicious traffic based on the three different sensors that were used. The size of each sensor and the country within which each sensor was located was taken into account during the observations made. The results include the most prolific countries with regards to malicious hosts and an interesting observation based on observed malicious events and the country within which the sensor resides.

- Section 5.4 will provide the analysis of the most targeted ports across the distributed network sensors. Comparisons were also made between each of the different sensors and the ports that were targeted on them. This provided insight into the most targeted services on the Internet and lead to the observation of the Carna botnet during its 2012 "census of the Internet".

- A brief overview of the most common malicious host operating system that were observed will be discussed in Section 5.5. The operating systems of malicious hosts was further discussed in the final section of this chapter.

- Section 5.6 will present a case study of malicious host ports. The study will group the most common open port combinations found on malicious hosts along with operating system families. And will conclude with an investigation into the correlation of open ports on malicious hosts and the ports they targeted.

- This chapter concludes in Section 5.7 with a summary of the results and an evaluation of the the rDSN modules and AR-Framework deployment in the real world.

5.1 Data Collection

Datasets were collected over a period of four months. Two data sets were collected with the AR-Framework; ARF1 and ARF2, a summary of this data is contained in Table 5.1. While three data sets were collected for Latency Based Multilateration analysis; LBM1, LBM2 and LBM3.

Table 5.1: Datasets collected using AR-Framework and independent LBM scripts

Name	Description	Start	End
ARF2	Ruscope196, Ruscope146, EliteHoneypot	2012-09-24	2012-10-04
ARF1	Ruscope, Elitepot, EC2Honeypot	2012-08-26	2012-09-22
LBM1	South African hosts LBM data	2012-09-26	2012-09-27
LBM2	LBM data for 983 International Alexa hosts	2012-12-17	2012-12-17
LBM3	LBM data for 550 South African Alexa hosts	2012-10-03	2012-10-04

Changes were made to the rDSN module between ARF1 and ARF2, by increasing the granularity of data collection on the network telescope to reflect two separate network ranges (146.x.y.z and 196.x.y.z). Furthermore, data collection from the EC2Honeypot was not performed for the ARF2 dataset due to technical issues with Amazon infrastructure and the AMPQ protocol. However, a honeypot located in South African IP address space was used for data collection in the ARF2 data set, this honeypot will be referred to as EliteHoneypot. During the collection period for ARF1 and ARF2, occasional interruptions were experienced due to connectivity problems with the ISP being used. The results presented in this chapter were based on the aggregated data of the ARF1 and ARF2 datasets.

5.2 Latency Based Multilateration Feasibility Study

LBM fingerprinting was proposed in Section 3.4, outlining the concept of how it would function and the methodology of obtaining the 3-typle values that constitute a LBM fingerprint for a given host. This process involved sending ICMP type 8 requests to a targeted host from three geographically separate Base Stations. The ICMP type 0 responses would be received, resulting in three sets of several Round Trip Time (RTT) measurements. Outliers were removed from each set of RTTs and the average RTT or latency from each *Base Station* to targeted host would be calculated. The output of this process produced a single 3-tuple value in the form of (latecyA, latencyB, latencyC) which represented a LBM fingerprint. If the majority of hosts connected to the Internet did not respond to ICMP requests, there would be little reason to pursue the use of LBM fingerprinting. As such the success rate of ICMP type 8 echo requests, receiving an ICMP type 0 reply was determined in Section 5.2.1.

The process of comparing two LBM fingerprints to each other was detailed in Section 3.4.4. Two LBM fingerprints would be compared to determine if they matched, in which case the two LBM fingerprints may potentially be considered as having originated from the same host, thus

providing a means to re-identify hosts in dynamic IP address space. The comparison process involved calculating the distance between two LBM fingerprints where the 3-tuple values had been mapped to euclidean 3-space. Thus providing two points in space, calculating the distance between these points in space would provide an indication of how similar the two fingerprints were. In order to determine if they matched, a threshold value would first need to be determined, this value would then represent the maximum distance between the two points for the LBM fingerprints to have originated from the same host. The calculation of the threshold and comparison of LBM fingerprints will be presented in Section 5.2.2.

5.2.1 Host Latency and Reach-ability Comparisons

A prerequisite of establishing the validity of LBM fingerprints and their application towards host re-identification was the determination of the success rate of hosts responding to ICPM type 8 echo requests. Furthermore, performing a comparison between the ICMP reach-ability of malicious and non-malicious hosts may yield interesting results as it would most likely be assumed that the ICMP reach-ability between malicious and non-malicious hosts should not be any different. If they did yield different results, an investigation into the configuration of these two sets of hosts on the Internet might warrant further research.

Gathering data from non-malicious hosts for the purpose of LBM fingerprinting appeared to be problematic, largely due to the uncertainty or lack of means to determine if a given IP was not malicious without previously observing nefarious activity from it. Section 2.2 detailed three technologies that could be used to detect malicious hosts; Intrusion Detection Systems, honeypots and network telescopes. Yet there is no real manner in which to detect if a host has been non-malicious, until it has been observed as malicious. To this end the Alexa[1] database was used to select non-malicious hosts. Alexa maintains a database of the most popular web servers on the Internet, of course it was assumed that the most popular websites, would in all likelihood not be malicious. It should be noted that the Alexa database may not contain a truly representative set of legitimate hosts on the Internet, as IP addresses belonging to DSL subscribers or the vast number of other legitimate servers on the Internet would not be present. This was, however, deemed as the most reasonable approach to identify non-malicious hosts on the Internet for the purpose of this experiment.

Data for this experiment was collected by measuring the LBM Fingerprint of 4777 hosts of

[1] http://www.alexa.com/topsites

Table 5.2: ICMP Reach-ability comparison between non-malicious and malicious hosts

Non-malicious Hosts	**1533**	**Malicious Hosts**	**3244**
Reachable	993 (64.77%)	Reachable	2273 (70.07%)
Unreachable	540 (35.23%)	Unreachable	971 (29.93%)

which 1533 (32%) represented non-malicious hosts taken from the Alexa database (LBM2 and LBM3 in Table 5.1) and the remaining 3244 (68%) representing malicious hosts identified through network telescope and honeypot sensors (ARF1 and ARF2 in Table 5.1). The 1533 hosts identified from the Alexa database was derived from the DNS resolution of the top 500 Websites, while the remaining 3244 malicious hosts represented the total number of LBM fingerprints collected from ARF data.

LBM fingerprints were collected for all of the 4777 hosts in this dataset. If any of the three Base Stations used for collection received at least one ICMP type 0 reply, the host was considered reachable. Thus hosts were only marked as unreachable if all three Base Stations failed to receive an ICMP type 0 reply.

It was found that 67.42% of the 4777 hosts had responded to the ICMP type 8 request packets, seen in Table 5.2. This indicated that LBM fingerprinting would be feasible from the perspective of collection. When comparing the two host populations, no significant difference was noted, as shown by Figure 5.1. 64.77% of non-malicious hosts were reachable, compared to a reach-ability of 70.07% for malicious hosts. The 5.3% would indicate that malicious hosts were more likely to respond to ICMP type 8 requests. However, due to the limited representation of non-malicious hosts in this dataset the increased likelihood of malicious hosts responding to ICMP type 8 requests could not be conclusively proven, a more representative dataset would be required.

5.2.2 Host Re-identification Based on LBM Fingerprints

In order to evaluate the feasibility of LBM fingerprints towards host re-identification, a suitable threshold value needed to be determined. This threshold value would represent the minimum distance required between two LBM fingerprints mapped in euclidean 3-space for them to be considered to represent the same host. Thus allowing for host re-identification in dynamic IP address space. An iterative approach was taken to determine the optimal threshold variable by preforming multiple LBM comparisons for threshold values ranging from 1ms to 350ms. The accuracy of these threshold values for host re-identification based on only LBM Fingerprints

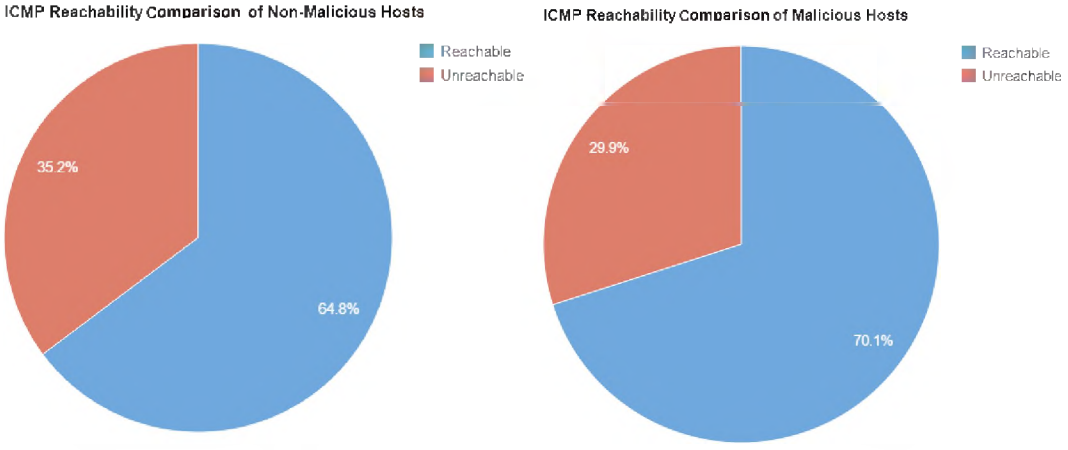

Figure 5.1: Reachability comparison between non-malicious and malicious hosts

were measured using a pool of LBM fingerprints (PopulationPool) from both non-malicious and malicious hosts.

An additional pool of LBM fingerprints (ABPool) that had been gathered over a period of three months (LBM1 in 5.1), fingerprinting two non-malicious hosts (host A and host B) every hour was used as the known LBM fingerprints. Two LBM fingerprints were chosen at random from the ABPool for host A, one of these LBM fingerprints would then be inserted into a set including all hosts monitored (PopulationPool). The distance between the second LBM fingerprint and all hosts in the set would then calculated. Hosts who had a LBM Difference that was equal to or smaller to a threshold were then considered to be the same, according to the LBM comparison and marked as a true positive if their IPs were also the same (since the original PopulationPool did not include an entry for host A). This method was repeated for threshold values from 1ms to 350ms, during each iteration of the threshold value, the test was run a total of 200 times. For each set of tests, the rate of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) were measured. These results were then used to calculate the accuracy and precision of each threshold variable for LBM fingerprinting and were then repeated for LBM measurements from host B in the ABPool. Figure 5.2 shows the re-identification of host A across multiple thresholds, while Figure 5.3 shows the same results for host B.

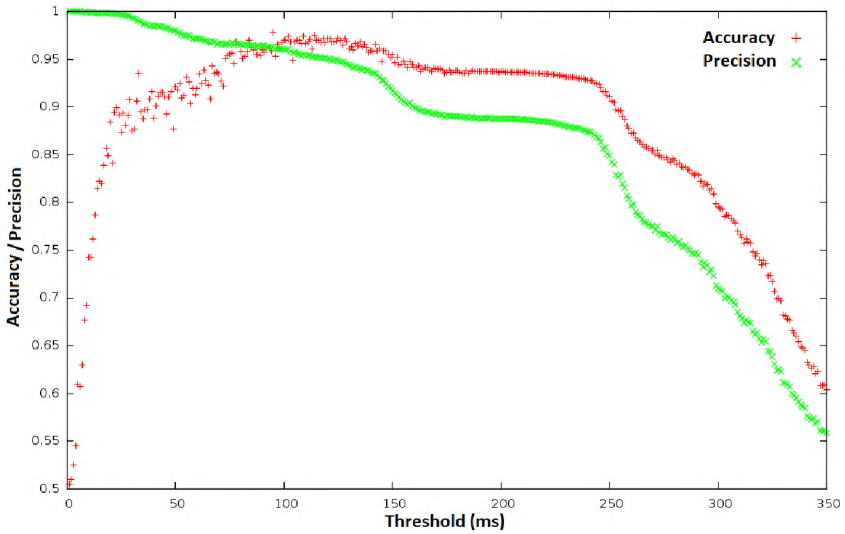

Figure 5.2: Accuracy and precision measurements for incremental threshold values for host A

Figure 5.2 shows an accuracy and precision of 96% converging at the point were the threshold value was set to 94ms. While Figure 5.3 shows the same convergence at 95% with a threshold value of 109 ms. Ultimately for host re-identification a slightly higher threshold value should be used to insure a higher true positive rate as the increased false positive rate could be circumvented by progressive validation of other attributes of the observed host; such as open ports, reverse DNS lookup and service versions. This, however, does indicate the LBM fingerprinting for host re-identification should only be used as a supporting metric and not on its own.

The accuracy of LBM fingerprinting would likely also be effected by the geographic location of the PingBase stations used for measurements, if the stations were not sufficiently distanced from each other, similar latency measurements would exist. This would decrease the accuracy of re-identification and was observed by looking at all of the latency measurements across the three PingBases.

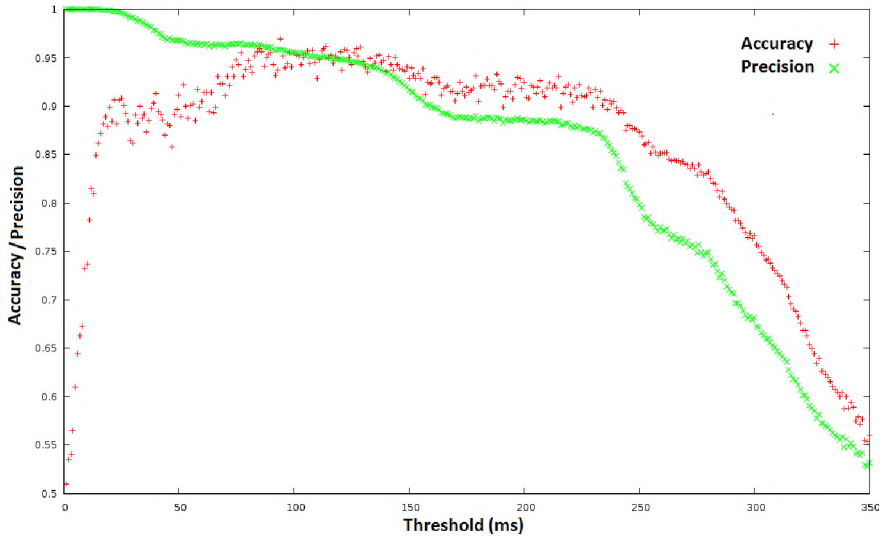

Figure 5.3: Accuracy and precision measurements for incremental threshold values for host B

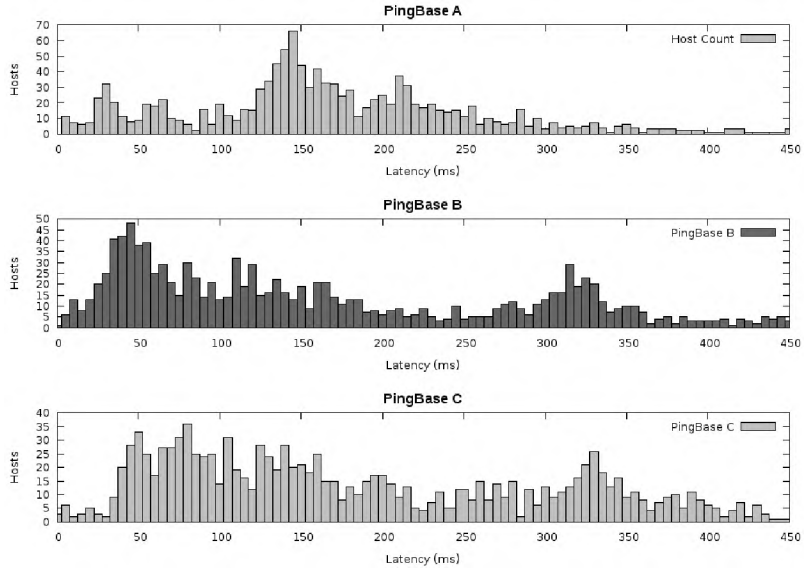

Figure 5.4: Latency comparisons between the three geographically separated ping *Base Stations*

This Figure shows 5.4 histograms representing the measurements of LBM fingerprint latency from the three PingBase stations. It is clear that PingBase B and C have a similar distribution in measurements, contrasted by PingBase A, this is likely as a result of the close geographic locality of PingBase B and C. Figure 5.5 show the estimated geographic location of PingBases A, B and C based the MaxMind database. This confirms the assumption as PingBase B and C are much

closer to each other, C situated in the Ukraine and B in the Czech Republic while A was located far from the other two in the United States of America. Ideally the PingBases should be located far from each other, such as one in North America, one in Europe and one in east or central Asia. However, as a proof of concept we believe these results prove the possible use of LBM fingerprinting for host re-identification as a supporting metric.

Figure 5.5: Geographic locations of the three ping Base Stations

This feasibility study for LBM fingerprinting concluded that hosts could be re-identified in dynamic IP address space using LBM fingerprint comparisons with an accuracy between of 94% and 96% using threshold values between 85ms and 125ms.

5.3 Geopolitical Distribution of Malicious Hosts

The Internet spans the world over connecting millions of devices, this provides a geographically and topologically diverse landscape for malicious hosts to exist in. Previous research from Harder et al. (2006) stated that even a small network telescope's result are relevant and may be extrapolated as a valid sample of malicious traffic. By gathering data from three heterogeneous data sensors a large number of malicious hosts were detected from around the world. The AR-Framework along with the rDSN modules allowed multiple small sensors to be used together to provide a more holistic and representative approach towards unsolicited network traffic collection. Figure 5.6 shows the geographic location of all malicious hosts detected through the RUScope sensor.

The network telescope operated across two /24 IPv4 network blocks, as a result it detected significantly more malicious hosts than both of the honeypots combined as each honeypot only monitored one IP Address. The number of hosts observed by the network telescope was also

Figure 5.6: Geolocation of malicious hosts detected by the network telescope sensor

larger due to the manner in which events were recorded. The network telescope registered an event as a SYN packet to any port, whereas the honeypot sensors only registered events targeting a service on which they were listening. As a result, the number of detected events and correlating malicious hosts would be far less on the honeypots as compared to the network telescope.

A similar trend in geographic distribution of hosts were observed when comparing the locations of malicious hosts in Figure 5.6 to the 460 million Internet connected devices observed by the Carna botnet in 2012, shown in Figure 5.7. As expected, locations with a high density of Internet connected devices across the world, have more malicious hosts than others. This verifies Harder et al. (2006) statement that the observations from small network telescope are still relevant as observed hosts are relatively evenly distributed based on geopolitical device density.

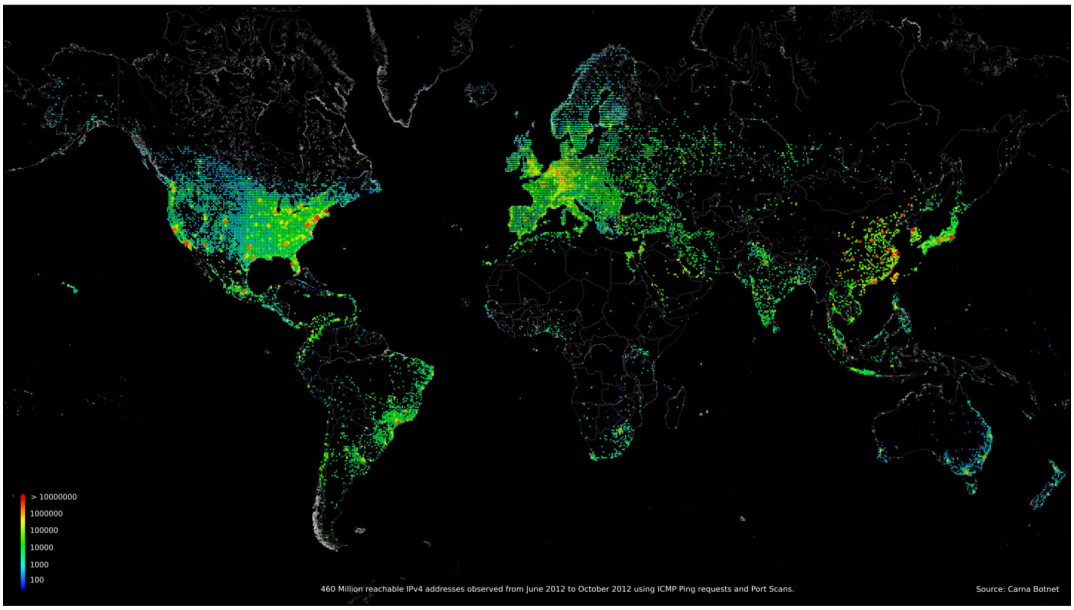

Figure 5.7: Reachable IPv4 addresses from the Carna botnet (Unknown, 2012)

Figure 5.8 shows the top 10 countries of originating malicious host traffic as observed by the RUScope sensor. China was the most observed country of origin with 15.77%, followed by the United States with 9.8% and the Russian Federation with 6.85%. Both the United States and China have previously been observed by Kim et al. (2012), as two of the top countries from which cyber attacks originate from.

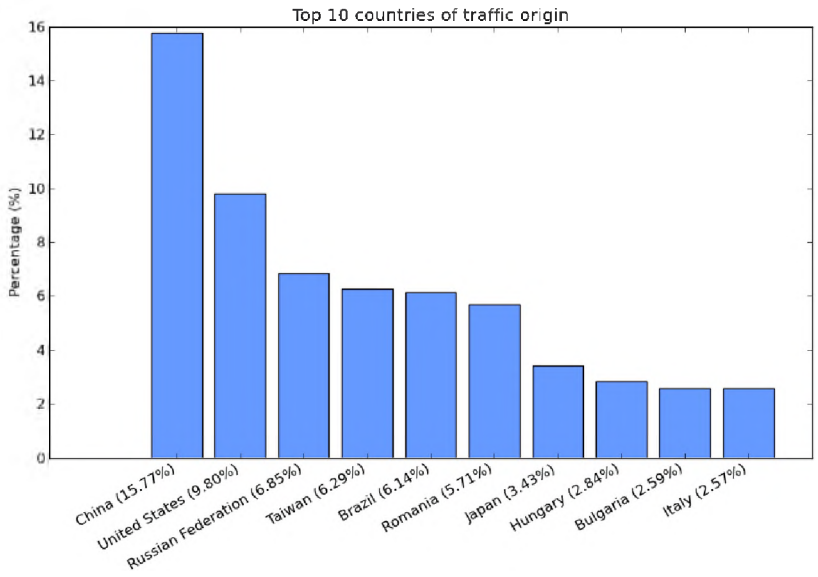

Figure 5.8: Top 10 countries from which malicious hosts were detected using the RUScope sensor

The difference in the number of observed hosts was clear when comparing Figures 5.6 and 5.9. Figure 5.9 showed the location of malicious hosts detected through the two honeypot sensors, green markers represented data collected by the EC2Honeypot sensor, while blue markers indicated data from the EliteHoneypot sensor. While the overall density was much lower, the geographic distribution of malicious host locations remained similar to those observed by the RUScope sensor.

Figure 5.9: Geolocation of malicious hosts detected by honeypot sensors

When inspected on a more granular level, the density of malicious host observations appeared very different from the observations made by the RUScope sensor. Table 5.3b shows the top 10 countries of malicious traffic origin as observed by the EC2Honeypot. China was again observed as the most prolific country of origin for malicious hosts, from the EC2Honepot sensor represented 58.6% of malicious hosts. This percentage was significantly higher than the 15.77% observed from the RUScope sensor. The observations between the EC2Honeypot and RUScope sensors shared five countries in the top 10. Three out of these five countries were shared in the top four between both sensors, the top four have been highlighted in Tables 5.3a, 5.3b and 5.3c. This indicated that while similar observations were made between the sources, geographically separated sensors would be required for a more holistic representation of observed events and malicious hosts. The EC2honeypot was situated in United States IP address space, while the RUScope and EliteHoneypot sensor were located in South African IP address space. This indicated that their may exist different distributions of origin based on the IP address space being targeted.

Table 5.3c shows the top 10 countries of malicious traffic origin as observed by the Elite-Honeypot. Once again China was observed the country with the most malicious hosts, totaling 40.58%. Similar to the EC2Honeypot, the percentage was significantly higher than that observed by the RUScope sensor and also significantly higher than those of the other countries that were observed by the EliteHoneypot. All three sensor's top 10 lists shown in Tables 5.3a, 5.3b and 5.3c included four countries, namely China, the United States, Russia and Taiwan, which have been highlighted. One could argue that the chance of malicious activity originating from any of

Table 5.3: Top 10 countries from which malicious hosts were observed based on unique IP address

(a) RUScope

Country	Hosts
China	**15.77%**
United States	**9.80%**
Russia	**6.85%**
Taiwan	**6.29%**
Brazil	6.14%
Romania	5.71%
Japan	3.43%
Hungary	2.84%
Bulgaria	2.59%
Italy	2.57%

(b) EC2Honeypot

Country	Hosts
China	**58.5%**
United Kingdom	9.52%
Taiwan	**8.84%**
United States	**8.16%**
Republic of Korea	4.08%
India	2.72%
Russia	**2.04%**
Argentina	0.68%
Japan	0.68%
Vietnam	0.68%

(c) EliteHoneypot

Country	Hosts
China	**40.58%**
United States	**12.7%**
Egypt	3.80%
Russia	**3.01%**
India	2.75%
Taiwan	**2.49%**
United Kingdom	1.96%
Turkey	1.96%
Morocco	1.83%
Brazil	1.83%

these four countries would be significantly higher than any other country.

While China was consistently observed as the country with the most malicious hosts, an interesting observation could be made about the United States. The United States was observed as the second highest country for malicious traffic to originate from on two of the three sensors. The two sensors where the United States ranked second were both in South African IP address space. The only sensor that was situated in United States IP address space, ranked malicious traffic from the United States in fourth place. In this context there would not be enough data to conclude a bias towards malicious hosts from a country attacking another more often than it's own, however, it may warrant further research. This observation would provide an apposing view to existing research claiming local bias for malicious activity, or more specifically automated worm propagation.

5.4 Analysis of Targeted Ports from Distributed Network Sensors

The investigation into the most commonly targeted network ports across all data sensors provided a cursory view of the most targeted ports or services on the Internet. It was shown in the Section 5.3 that the sensors captured malicious traffic from geographically distributed locations. In this section the topic moves away from the origin of malicious traffic in order to look at the ports that were targeted by the observed malicious hosts. The insight into which ports malicious hosts would most likely target could be used as a focus point with regards to securing hosts against a potential compromise. The services exposed on these ports would be the most important when it comes to patch management or limiting exposure through firewalls. The targeted ports could also be indicative of current or new trends, even indicating potential new exploits in the wild before they have been made public. Portions of this data was previously shown in examples through the visualisation tools outlined in Section 4.7. The graphs found in this section illustrate the most targeted ports and as such the most targeted services across the different sensors. The exploration of this data as a whole and according to individual sensors provide a unique view into the targeting trends of malicious hosts over heterogeneous sources.

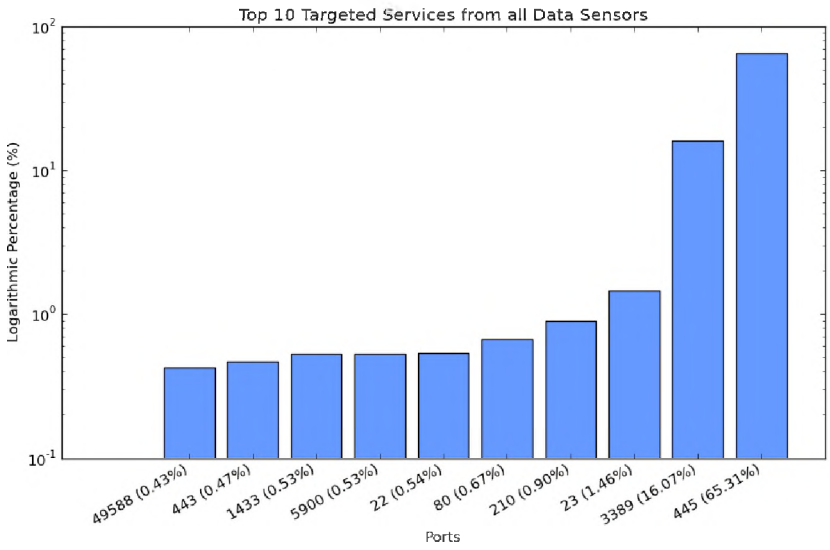

Figure 5.10: Most targeted TCP ports across all data sensors

Figure 5.10 shows the aggregated data from both honeypot sensors and the network telescope with regards to the top 10 most commonly targeted ports. The three most commonly targeted ports accounted for 82.84% of all observed traffic, with the remaining ports each accounting for less than 1% of the targeted ports. Port 445/tcp was targeted the most by malicious hosts, this port is most commonly used for the Server Message Block (SMB) protocol. The SMB protocol is synonymous with file-sharing in Microsoft Windows systems and was heavily exploited by the Conficker worm (Aben, 2009; Irwin, 2011). Conficker exploited the MS08-067 vulnerability (TechCenter, 2008), which resulted in Remote Command Execution (RCE) on the vulnerable host with system privileges. This observation indicates that even though the data was captured four years after the vulnerability was exploited, it was still being targeted on the Internet.

The second most commonly targeted port was found to be 3389/tcp which is used by the Remote Desktop Protocol (RDP) to provide remote administration capabilities. The 3rd most commonly targeted port, 23/tcp is used for Telnet, also providing an administrative interface. Both RDP and Telnet are often secured through a username and password combination. Authentication based services like these are often targeted by brute-force or dictionary based password guessing attacks. Overall, the likelihood of success towards compromising a system by guessing credentials would be substantially higher as it does not depend on exploiting a specifically vulner-

able version of software, but rather human error. Further ports used for administrative services in the top 10 included 22/tcp (SSH) and 5900/tcp (VNC).

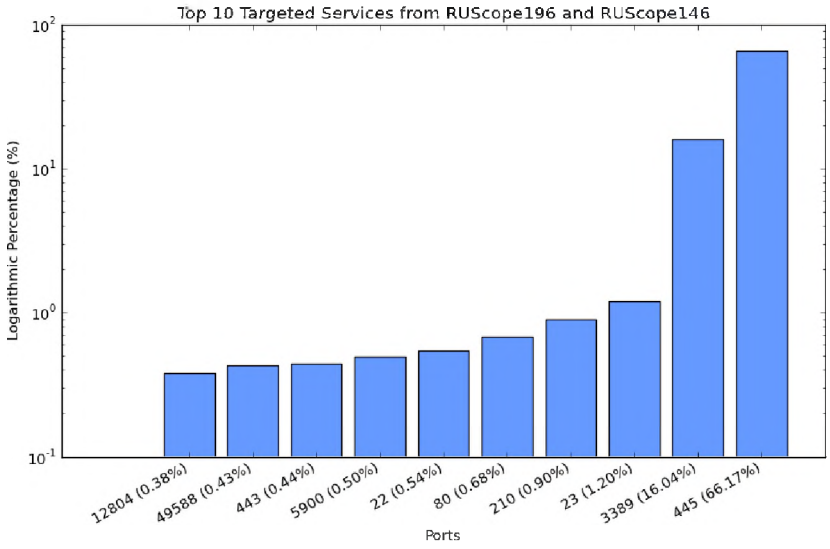

Figure 5.11: Most targeted ports across the network telescope sensors

Figure 5.11 summarised the top 10 targeted ports as observed by only the network telescope sensors, this included both the 196 and 146 /24 networks. When compared to the overall observed malicious interaction with all of the sensors from Figure 5.10, the graphs are almost identical. This was as a result of the larger sensor size of the network telescope when compared to that of the two honeypots. The interesting exception here was that port 1433/tcp (MSSQL) did not feature in the top 10 targeted ports from the network telescope sensor, indicating that it would be one of the top observed ports on one or both of the honeypot sensors. While MSSQL is not used for system administration such as RDP, Telnet or SSH, if successfully exploited it could lead to RCE with system privileged. This is often as a result of weak credentials for the default sa account.

When comparing the targeted ports from the two network telescope ranges in Tables 5.4a and 5.4b, three out of the top five ports correlate (445/tcp, 3389/tcp, 210/tcp). One port in particular that was shared by both the network telescope sources that has not yet been discussed was 210/tcp. This port is most commonly used for the Z39.50 protocol Kunze et al. (2013), an international client-server, application layer communications protocol, used to retrieve informa-

Table 5.4: Top 10 most targeted ports across network telescope sensors

(a) RUScope196

Port	Traffic (%)
445	86.77
3389	6.43
535	0.59
13595	0.33
210	0.30
23	0.30
22	0.24
80	0.23
51389	0.22
443	0.17

(b) RUScope146

Port	Traffic (%)
3389	46.79
210	4.56
23	3.78
445	3.35
5900	2.78
443	2.78
80	1.39
41511	1.38
22	1.36
37432	1.17

tion from a database over TCP/IP networking. According to Le Malécot and Inoue (2014) the 210/tcp port may also have been used by the Carna Botnet Shukla (2015); Unknown (2012) to control bots after they had been deployed on a device.

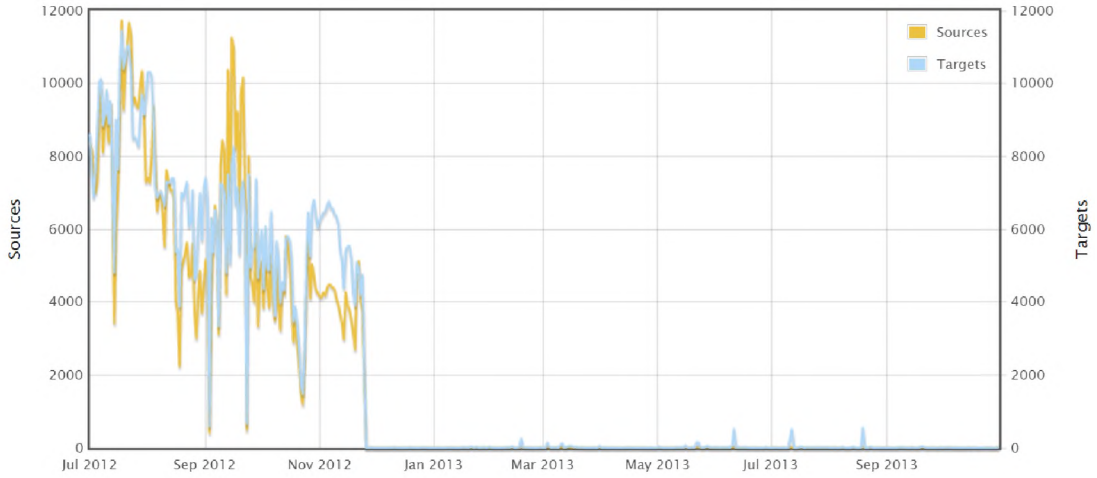

Figure 5.12: Traffic targeting port 210/tcp, observed by the Sans Internet Storm Center

Figure 5.12 shows traffic for port 210/tcp from the 24th of July 2014 until the 24th of September 2013 from the ICS Sans sensors (SANS, 2012). The AR-Framework collected data over the months of August, September and October of 2012 which correlates with the observations in Figure 5.12. Unfortunately, this data was only analysed after this peak period, had it been observed earlier this could have been used in a similar fashion as the Associate fingerprinting technique

which was outlined in Section 3.3. The Associative fingerprinting technique demonstrated the application of data collected from hosts belonging to the Kelihos/Hlux botnet towards fingerprinting. The observation of increased traffic to port 210/tcp on both network telescope sensors also affirm Harder et al. (2006) observations of a small network telescope being able to observe global events, keeping in mind that the RUScope sensor operated over only two /24 networks.

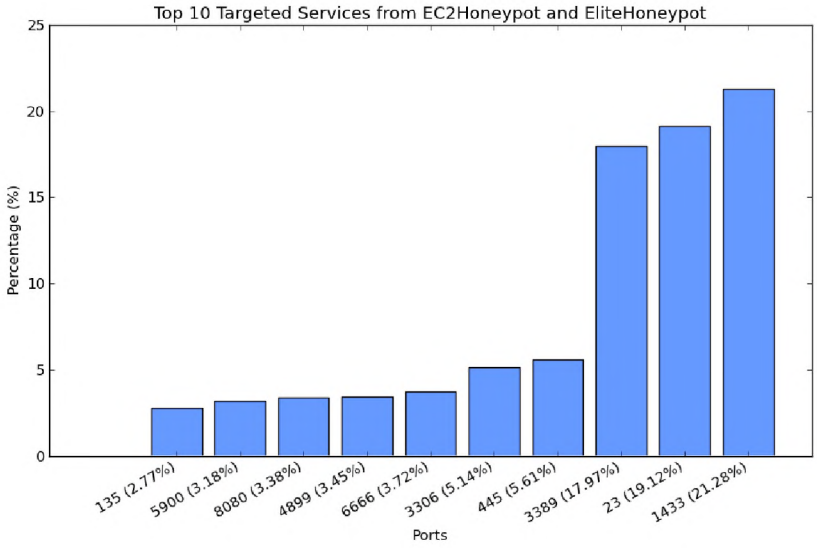

Figure 5.13: Most targeted services across both honeypot sensors

Figure 5.13 shows the most targeted services across both honeypot sensors. The first port of interest here was that of 1433/tcp used for MSSQL, which did not make the top 10 targeted ports for the network telescopes sensor. One reason for this might be the a mechanism employed by Microsoft to assist other services in discovering the TCP port used for hosting the MSSQL service. While MSSQL is most commonly exposed through port 1433/tcp, it does not have to be, and like any service could be exposed on any available port. In order to assist other services to discover on which port a MSSQL server is running, they could query port 1433/udp, which would reply with the correct port to initiation MSSQL connections with. This becomes important when considering the particular difference in targeted port observations between the network telescope and honeypots. It may even provide insight towards the method employed by malicious agents to discover the MSSQL service, due to the network communication differences between these sensor types. Passive network telescopes such as RUScope were described in Section 2.2.3 as sinkholes

Table 5.5: Top 10 targeted ports on honeypot sensors

EC2Honeypot		**EliteHoneypot**	
Port	**Traffic (%)**	**Ports**	**Traffic (%)**
1433	23.81	23	21.09
6666	18.52	1433	20.93
3389	13.52	3389	18.68
3306	11.64	445	6.36
23	5.82	3306	4.19
1080	4.76	4899	3.88
9097	2.12	8080	3.8
1095	1.59	5900	3.57
6014	1.59	135	3.18
60019	1.06	443	2.64

that capture all traffic destined towards them but do not reply to any of it. With reference to the disproportionate observation of port 1433/tcp traffic, the honeypots would reply to an initial probe to 1433/udp, as such the malicious host would be more likely to attempt a connection to 1433/tcp on the honeypot. This may indicate that passively configured network telescopes have certain biases towards being targeted due to their lack of interaction.

Table 5.5 shows the top 10 targeted ports on the EC2Honeypot and EliteHoneypot sensors. Both have 1433/tcp and 3389/tcp in the top 3 most targeted ports, however, the EC2Honeypot sensor observed 6666/tcp as its second highest targeted port, representing 18.52% of all traffic targeting the EC2Honeypot. Port 6666/tcp is often used legitimately for Internet Relay Chat (IRC), however, over the years many different types of trojan horses and RATs have also used 6666/tcp for communication. Two other notable ports were 1080/tcp shown as 4.76% on the EC2Honeypot sensor and 8080/tcp on the EliteHoneypot sensor. Both of these ports are often used to expose proxy services or web based administrative interfaces. If these services are not password protected or port wrapped, they could allow malicious users to re-direct traffic through them or even lead to complete compromise of the host.

Some of the discrepancies observed between targeted ports on the network telescope sensors in Table 5.4 versus the honeypot sensor in Table 5.5 would appear to agree with Shinoda et al. (2005), who stated that a single source for capturing malicious activity could not accurately represent the extrapolation of such activity. This is contrary to the observations made by Harder et al. (2006) and our own observations of 210/tcp activity on the network telescopes where a global event was observed on a small sensor. While all of the data sensors shared some of the

same observations, the distributions were in some cases significantly different. Indicating that different types of malicious agents were attacking different network ranges and that certain biases may exist based on sensor capabilities. Fortunately this problem may be solved by the deployment of additional rDSN modules to increase the collection scope and heterogeneity of sensor devices.

5.5 Malicious Host Operating Systems

Through the AR-Framework it was possible to interrogate malicious hosts in order to determine their operating systems. This active OS fingerprinting was performed using Nmap version 6.00 with the purpose of identifying the most often compromised operating systems.

Figure 5.14 shows the top 10 operating systems running on malicious hosts as detected by the data sensors. The most observed operating system was FreeBSD 6.2-RELEASE-p2 with 45.7%, this operating system is often found on servers acting as a firewall and would be expected to be one of the most observed operating systems. FreeBSD 7.0 was the 5th most observed with 5.07%, along with FreeBSD 8.0 and 4.6 also in the top 10. In total, FreeBSD operating systems accounted for 53.65% of the total observed operating systems. The majority of hosts with access to the Internet would be on an internal network with access to the Internet provided through NAT. Resulting in single firewall IP addresses representing the public Internet facing IP address of many hosts behind that firewall (internal network).

The next three most popular operating systems were all from Microsoft; Windows XP SP2 accounting for 11.36% as the second most common operating system. Windows Server 2003 and XP SP3 were the 3rd and 4th most common operating systems. All three of these operating systems were vulnerable to MS08-067 (TechCenter, 2008), the vulnerability exploited by the Conficker worm (Aben, 2009; Irwin, 2011, 2012). As newer operating systems at the time of observation, such as Windows Server 2008 were not observed at all in the top 10 list, this may be an indication that the majority of the Microsoft Windows hosts observed had been previously compromised and were now being used for malicious purposes.

5.6 A Case Study of Malicious Host Ports

Through the application of the AR-Framework and rDSN modules, it became possible to perform reconnaissance on malicious hosts, in near real-time. This allowed for the enumeration of characteristics from these malicious hosts that had previously not been possible within the field of

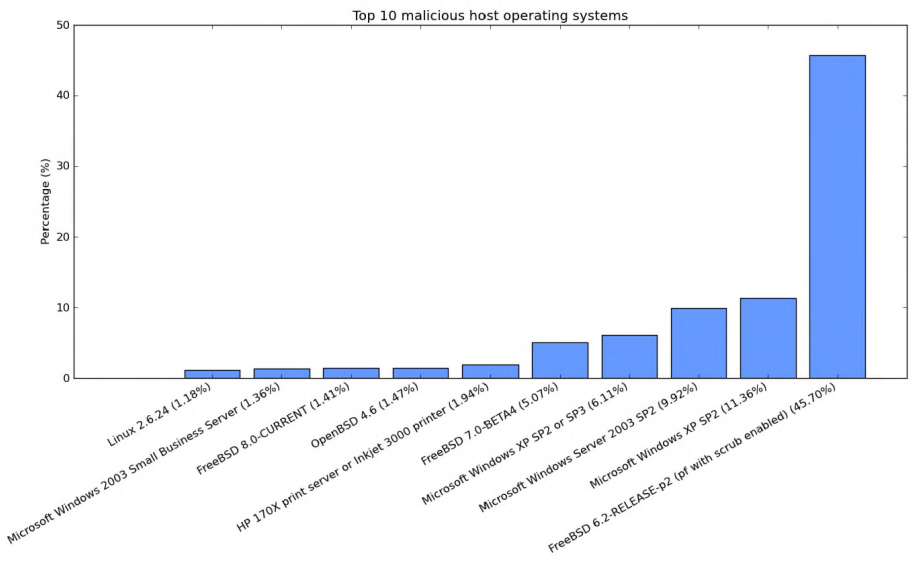

Figure 5.14: Top 10 malicious host operating systems across all data sensors

unsolicited traffic analysis. While port scanning has been a well defined technique for many years, to our knowledge it has never been used in conjunction with network telescopes and honepots for the purpose of gathering information from the malicious hosts. To this end we have analysed the most commonly exposed port combinations and aligned these with operating system fingerprints to provide a new perspective on the malicious hosts that were responsible for the unsolicited traffic. This case study explores those correlations in an effort to extend existing knowledge of malicious hosts on the Internet.

5.6.1 Open Port Statistics for Malicious Hosts

By analysing open port combinations from the AR-Framework it was possible to identify interesting correlations between open ports and the operating systems of malicious hosts. This data also varied significantly according to the type of sensor used to detect malicious interaction. This may be an indication that the sensors detected malicious traffic from different kinds of malicious hosts, as was observed by the differences of the most targeted services according to the different sensors in Section 5.4.

The top 10 most common open ports found on malicious hosts have been listed in Table 5.6, this included malicious hosts detected by all data sensors. Similar to the most targeted

Table 5.6: Top 10 open ports found on malicious hosts

Port	Service Name	Hosts (%)
3389	RDP	43.71
443	HTTPS	29.96
21	FTP	16.84
1723	PPTP	11.98
23	TELNET	11.12
22	SSH	9.56
110	POP3	7.15
5900	VNC	6.56
143	IMAP	4.53
88	HTTP (commonly)	4.31

ports, ports that provided management services were most common in the top 10 list. The ports included 3389/tcp (RDP), 21/tcp (FTP), 23/tcp (Telnet), 22/tcp (SSH) and 5900/tcp (VNC). These management service ports constituted half of the top 10 list and accounted for a combined total of 87.79% of open ports across all malicious hosts. Another common port listed in Table 5.6 was 1723/tcp, often used for the Point to Point Tunneling Protocol (PPTP) service to provide Virtual Private Network (VPN) access. This port is often found to be open on Internet facing servers or firewalls to allow access to an internal or protected networks. Worryingly, if a large portion of the hosts with 1723/tcp exposed had been previously compromised, attackers would likely have access to the internal network associated with those VPN entry points.

Figure 5.15 shows the top six open ports found on malicious hosts as a proportional to each sensor and grouped by the data sensor used to detect the malicious hosts. This new style of visualisation was chosen as it allows for effective comparision of metrics, such as source sensor and observed open port on malicious host. From this graph in Figure 5.15, it was clear that port 3389/tcp was the most common open port across detected malicious hosts, this port was also one of the most targeted ports by malicious hosts. A close second but with far less open ports detected from network telescope data was port 21/tcp. The next port was 443/tcp, with almost the exact same distribution across all sensors. When hosting malicious content such as malware it becomes advantageous to do so over an encrypted channel such as SSL in an attempt to circumvent network based IDS software (Marpaung et al., 2012). This simple IDS evasion technique could explain the why port 80/tcp does not feature in the top six ports and rather 443/tcp. The last two notable ports were 22/tcp and 23/tcp; used for SSH and Telnet respectively. Both used for remote administration of a system and also previously found to be in the top 10 list for most

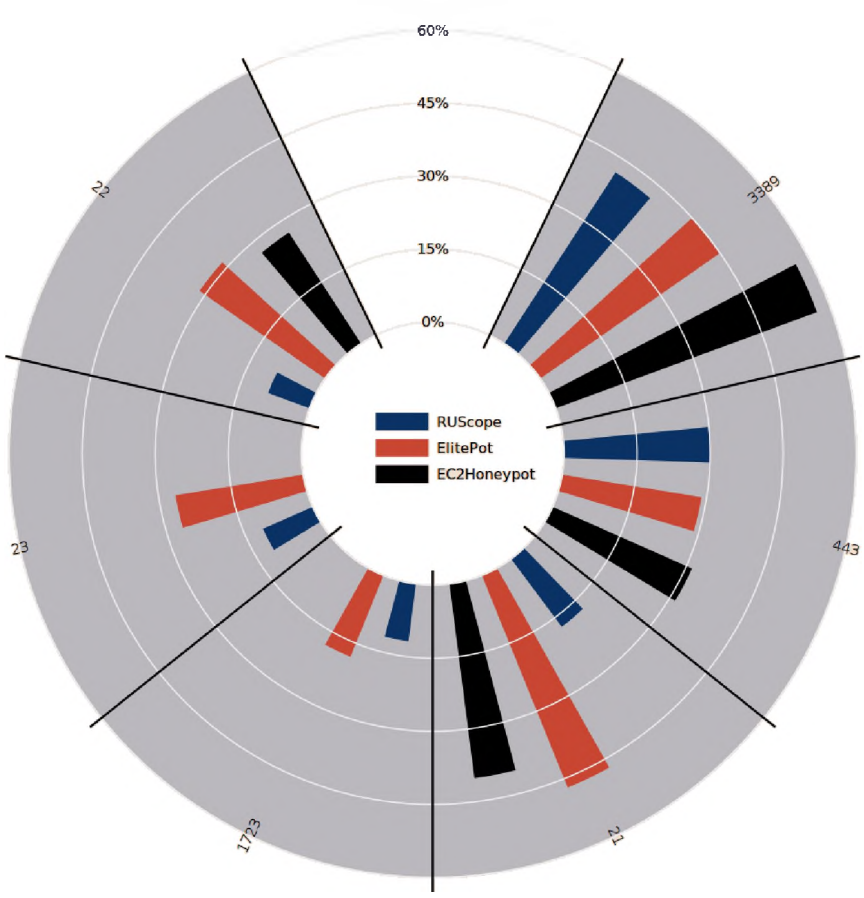

Figure 5.15: Most common open ports detected on malicious hosts, grouped by data sensor.

targeted ports.

For Figures 5.16, 5.17 and 5.18 the top combinations of two or more open ports were taken from single sensors and grouped by operating system family. With regards to Figure 5.18, both RUScope146 and RUScope196 were used. Data from malicious hosts registered by the Elite-Honeypot sensor was shown in Figure 5.16, while Figure 5.17 represented malicious hosts from the EC2Honeypot sensor. The operating systems were determined through Nmap scans completed by the AR-Framework, this information was grouped according to major operating system family types, namely Windows, Linux, BSD and all other OS types.

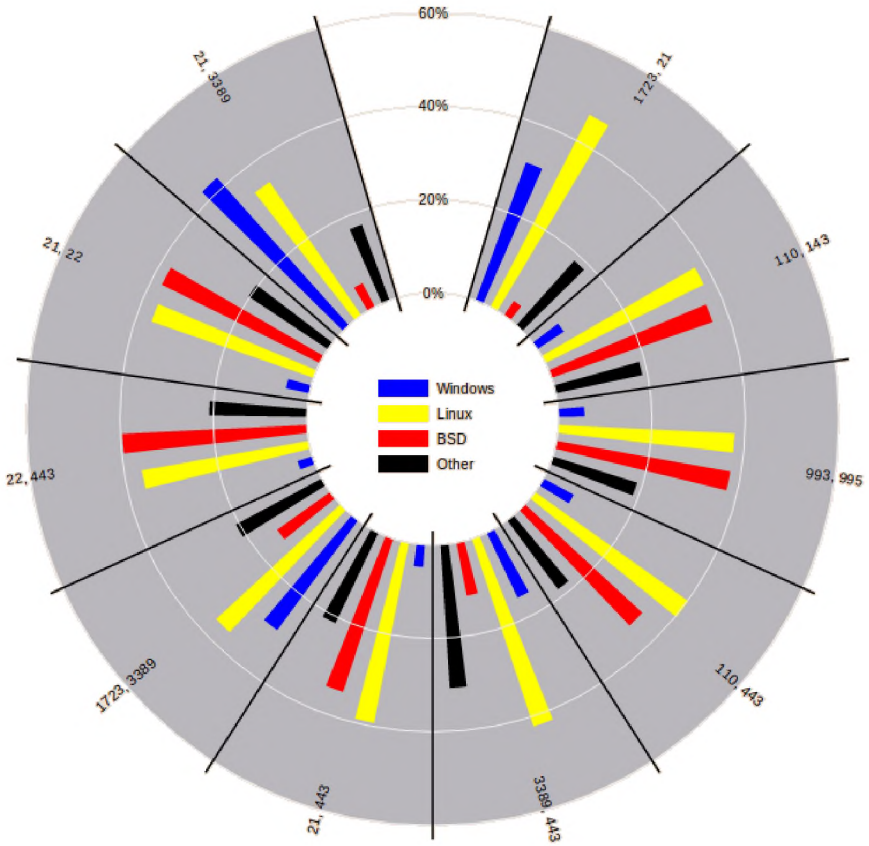

Figure 5.16: Top combinations of two or more open ports with correlating OSs from the Elite-Honeypot

Figure 5.16 clearly shows Linux and BSD systems dominating all port combinations except for the port 21/tcp and 3389/tcp combination, as would be expected with RDP being a protocol found on systems running Microsoft Windows. Explanations for the large proportion of 3389/tcp

port instances found on Linux hosts could be as a result of port forwarding or other services running on 3389/tcp. One of the most common combinations of ports included 110/tcp, 143/tcp, 993/tcp and 995/tcp; often also grouped in different permutations. These four ports are most often used for providing email services; 110/tcp for POP3 while 143/tcp is used for IMAP. Their encrypted counterparts; 993/tcp for IMAP over SSL and 995/tcp being used for POP3 over SSL. These malicious hosts were most likely used for sending phishing emails; this theory was motivated more by the explicit combinations of port 110/tcp and 443/tcp as well as 143/tcp and 443/tcp were mail servers may be used to send phishing mails or spam, while simultaneously hosting content for the particular campaign on port 443/tcp (HTTPS).

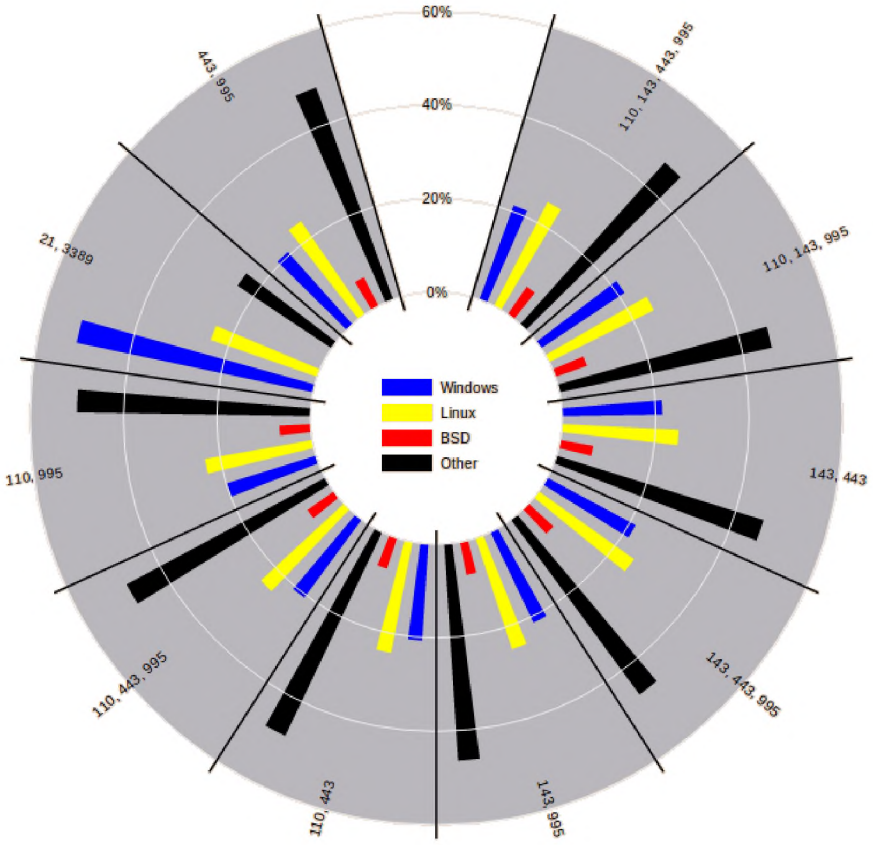

Figure 5.17: Top combinations of two or more open ports with correlating OSs from the EC2Honeypot

Malicious host port combinations from the network telescope are shown in Figure 5.18. There

was a clear difference between the combinations of ports exposed by malicious hosts observed by the honeypots and those that were observed by the network telescope. Firstly the top port combinations from the network telescope sensor clearly favors Windows hosts, which is evident by the fact that the RDP port 3389/tcp, was present in six of the top 10 port combinations. A new port also featured in the network telescope dataset, namely 1723/tcp (PPTP). Port 5900/tcp, another remote administrative port was also observed. Port 5900/tcp is often used for Virtual Network Computing (VNC) offering a similar remote management service as RDP.

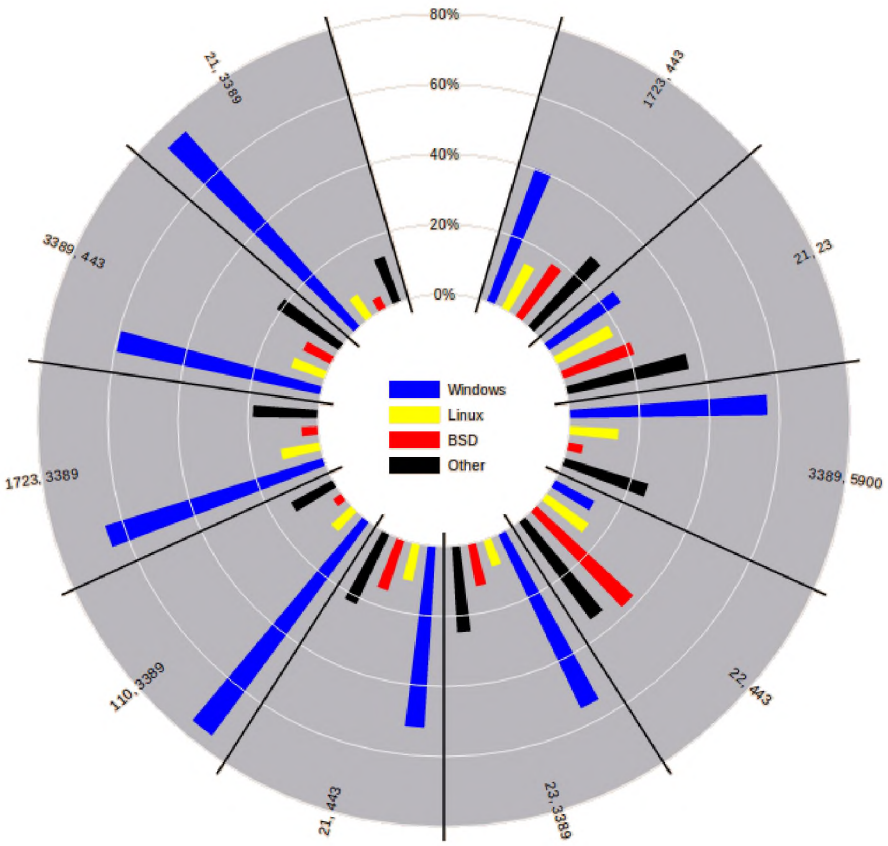

Figure 5.18: Top combinations of two or more open ports with correlating OSs from the network telescope sensors

One of only two port combination from the network telescope dataset that was not dominated by Windows OSs was the port 22/tcp and 443/tcp. SSH is considered the de facto remote system administration tool for Linux and BSD based systems, which is re-affirmed from the lack of Windows operating system representation in this combination. The second combination not

dominated by Windows hosts was for ports 21/tcp and 23/tcp; FTP and Telnet respectively. This category was dominated by "Other" operating systems that were not categorised directly under the three previous operating system families. It is likely that these hosts represent devices such as routers and switches located on public Internet facing endpoints. These devices are often administered over SSH or Telnet.

5.6.2 Targeted Sensor Port and Exposed Malicious Host Port Correlation

The AR-Framework allowed for the near real-time reconnaissance of malicious hosts detected by the distributed, heterogeneous sensors. This reconnaissance included a port scan of every malicious host, thus identifying the services these malicious hosts expose to the Internet. For the first time it was possible to investigate whether the services that were targeted on the sensors, were also exposed on the malicious hosts. This correlation could provide an indication of whether the malicious host, itself was compromised through the same service it had targeted on the data sensors.

Towards this end, the data collected by the AR-Framework was analysed by looking at all the port scan data that had been collected on malicious hosts and comparing that with the port that was targeted on the data sensors. Figure 5.19 shows the result of this investigation, only 11.5% of all malicious hosts exposed the same port that they had targeted. This low correlation could be the result of many factors. Firstly, the service might have been the same, but was no longer exposed to the Internet. It is belived that hackers would at times fix the vulnerability they used to compromise a host, decreasing the likelihood that it could be compromised by someone else, however, there is no way of proving this to be the case. Administrators of the host may also have blocked access to the port using a firewall.

Another reason for this result may have been manner in which the Assessment phase of the AR-Framework function or more specifically the Session Manager component functioned. The Session Manager kept track of incoming packets or events and maintained active sessions to insure that a single malicious host was not fingerprinted multiple times for each packet received. A limitation of the way in which it was implemented, resulted in only the first packet being received was used to record the targeted port. This shortcoming would limit the identification of multiple targeted ports, however, if this was assumed, then data from the honeypot sensors that identified attacks per event targeting a service, would not be effected. The data was analysed

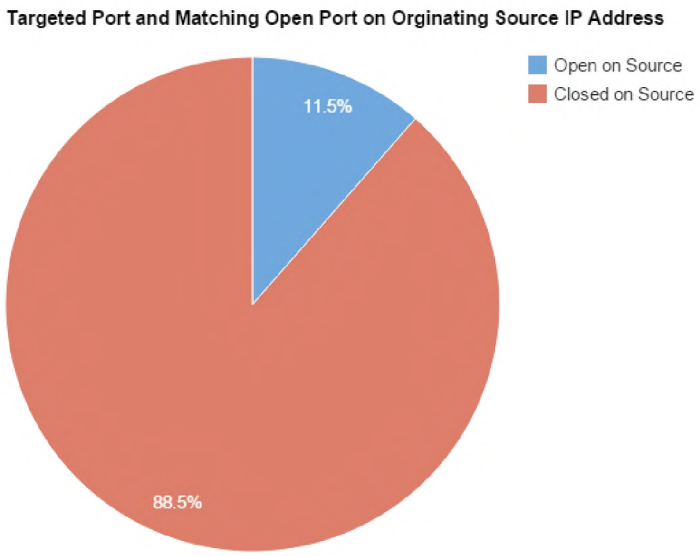

Figure 5.19: Targeted port and open port correlation on malicious hosts.

using only data collected from honeypots, in this case the open port correlation was even lower at 7.67%, disproving the theory. As such multiple vectors of attack may have been employed, one to compromise the host and others to attack new hosts or the owner of the host was acting maliciously.

The next step of analysis involved the identification of the correlated ports that were most common; Figure 5.20 shows the 10 most common correlations by port. These results were dominated by a single port and as a result the data has been displaced on a logarithmic scale. The most prolific port was 3389/tcp, commonly used for RDP. Port 3389/tcp represented 88.17% of the total proportion of correlating ports, with the second most common port being 445/tcp representing only 5.71% of the ports.

Figure 5.10 showed the top 10 most targeted services that were observed across all data sensors, when this was compared with Figure 5.20, eight out of the 10 most commonly targeted ports were in the list of correspondingly open and targeted ports. While the hit ratio of targeted port and open port on the originating host was only 11.5%, this correlation may well indicate that the eight ports that correlated between the most targeted services and the matching targeted port and originating host port, were the ports used to initially compromise those hosts. These eight ports have been listed in Table 5.7.

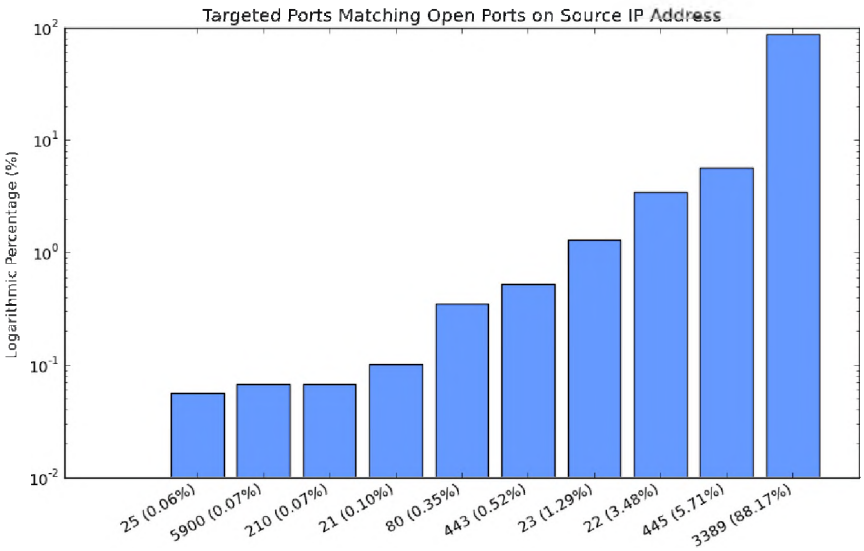

Figure 5.20: Top 10 targeted ports with corresponding open ports on the malicious host

Table 5.7: Potential services used for initial compromise and further attacks

Port Number	Service Name
3389	RDP
5900	VNC
445	SMB
443	HTTPS
210	Z39.50
80	HTTP
23	Telnet
22	SSH

5.7 Results

The deployment of rDSN modules on the three data sensors allowed for the identification of malicious hosts in near real-time. This data was exposed to the AR-Framework which in turn performed active reconnaissance on each of the identified hosts and aggregated this data according to the adapted multi-sensor data fusion model outlined in 4.3. The rDSN modules allowed for malicious hosts to be identified from multiple heterogeneous sources, thus providing the foundation for a holistic representation of malicious activity from distributed sources.

5.7.1 Latency Based Multilateration

A LBM feasibility study was conducted in Section 5.2.1 in order to determine what proportion of hosts on the Internet would respond to ICMP type 8 requests. On average, malicious hosts were found to be more likely to respond to ICMP type 8 requests by 5.3%. The re-identification experiment for LBM fingerprinting followed. It concluded that hosts could be re-identified in dynamic IP address space using LBM fingerprint comparisons with an accuracy between of 94% and 96% using threshold values between 85ms and 125ms. It was found that a higher threshold value would need to be used insure a higher true positive rate as an increased false positive rate could be circumvented by progressive validation of other attributes of the observed host; such as open ports, reverse DNS and service versions.

5.7.2 Geographic Observations

Findings regarding the geopolitical distribution of malicious hosts were presented in Section 5.3. The observations from the largest sensor, RUScope, presented a similar distribution of malicious hosts as Internet connected devices found by the Carna botnet. Indicating that from a geographical standpoint, where more hosts exist, more malicious hosts would exist. Comparisons between the origins of malicious traffic and the three sensors that were used showed that China was responsible for the most malicious activity. The two sensors in South African IP address space observed the United States as the second most prolific country, while the sensor in United States IP address space only observed the United States as the fourth highest origin of malicious traffic. All three sensor's top 10 lists shared the same four countries, namely China, the United States, Russia and Taiwan. Indicating that malicious activity was more likely to originate from these four countries than any other country. Information such as this could be used to generate

dynamic geo-based blacklists for highly sensitive infrastructure.

5.7.3 Targeted Ports

Results pertaining to the most commonly targeted network ports were provided in Section 5.4. The three most commonly targeted ports on the network telescope sensor accounted for 82.84% of all observed traffic. Ports relating to administrative services were found to be the most targeted across all of the sensors and included 22/tcp, 23/tcp , 3389/tcp , and 5900/tcp. One of the differences observed between the network telescope sensors and the honeypot sensors was the detection of traffic targeting 1433/tcp. This was explained as a possible limitation of passive network telescopes not being able to reply to MSSQL port discovery queries targeted towards 1433/udp. Traffic originating from the Carna botnet was also observed, which correlated with a significant increase in traffic to port 210/tcp as observed by the ICS Sans sensors (SANS, 2012) over the same period of time. The insight into the most targeted services could be used for defenses purposes. The services exposed on these ports would be the most important when it comes to patch management or limiting exposure through firewalls.

5.7.4 Malicious Host Operating Systems

An overview of malicious host operating system were provided in Section 5.5. The findings presented the most commonly found operating systems on malicious hosts, as determined through active fingerprinting as apposed to passive fingerprinting. The most common operating systems were found to be different releases of FreeBSD, accounting for 53.65% of observed hosts. FreeBSD is often used for maintaining firewalls and as a result could be an indication of a malicious host sitting behind a firewall. Since FreeBSD systems are often administrated over SSH this could also indicate why 22/tcp was one of the most targeted services across data sensors. Microsoft Windows was also found to be one of the most targeted operating systems, with Windows XP and Windows Server 2003 representing 28.75% of all operating systems. Both of these distributions of Microsoft Windows was vulnerable to MS08-067 which was previously exploited by Conficker. Providing motivation for the most targeted port 445/tcp (SMB) observed on the network telescope sensor.

5.7.5 Port Combinations

Section 5.6 presented open port combinations of malicious hosts along with correlations between these open ports and the different malicious host operating systems that were observed. This provided insight into the configuration of previously compromised hosts. Many of the ports exposed on these malicious hosts were also found in the top 10 lists for the most targeted services. The exposed ports also included administrative ports, while the port combinations in most cases correlated with what would be expected from the operating systems exposing those combinations. This data could be used in areas such as threat intelligence and risk management on networks to expose the likelihood of potential compromise based on the configurations exposed by malicious hosts on the Internet. Targeted ports and exposed ports were then correlated in an investigation to determine the likelihood of a malicious host targeting the same service as it exposed. Only 11.5% of malicious hosts were found to be exposing the same port as which they targeted on one of the sensors. Of the 11.5% of malicious hosts with correlating open and targeted ports, 88.17% represented 3389/tcp with the second most common port being 445/tcp with only 5.71%. Eight out of the 10 most commonly targeted ports were in the list of correspondingly open and targeted ports. This correlation may may indicate that the eight ports that correlated between the most targeted services and the matching targeted port and originating host port, were the ports used to initially compromise those hosts. As such the services exposed on these eight ports should be the most important services to secure in order to increase the defensive posture of a network against automated threats.

5.8 Summary

This chapter presented a discussion on the data collected through the application of rDSN modules and the AR-Framework in the real world. Based on the results and interpretation thereof, the rDSN modules successfully allowed for the collection and exposure of data in near real-time from heterogeneous sensors. The AR-Framework successfully performed active reconnaissance on these malicious hosts, including multiple fingerprinting techniques and alignment of this data according the adapted multi-sensor fusion model presented in Section 4.3. LBM fingerprinting proved successful in host re-identification in dynamic IP address space as a supporting metric. During the interpretation of results, an improvement to the AR-Framework Session Manager became apparent due to its inability to record multiple targeted ports in the same session. This

shortcoming has been listed in Section 6.4 for future work.

There is no real ending. It's just the place where you stop the story.

Frank Herbert

6 Conclusion

The Chapter begins by providing a summary of the research detailed in each Chapter. The research goals are then evaluated based on the work that has been presented. The Chapter concludes by providing a direction for future work, which could be used to address identified shortcomings while also extending this research towards further study.

6.1 Summary of Thesis

The goal of this work was to develop the means to undertake an empirical study of malicious hosts on the Internet. The subsequent purpose of this was to present the characteristics of the hosts responsible for malicious traffic on the Internet. This included the analysis of malicious host characteristics towards a better understanding of a malicious hosts demographic and the application of host re-identification techniques in dynamic IP address space. Thus providing a new perspective on the sources of malicious traffic on the Internet and expand the available

methods for gathering information on these hosts. Before evaluating the research goals outlined in Chapter 1, a summary of the contributions from each Chapter has been provided.

Chapter 2

Before information could be gathered on malicious hosts, they first need to be identified; to this end network telescopes and honeypots were used as data sensors. As Internet Background Radiation (IBR) may contain traffic from victims (non malicious hosts) of DDoS attacks (Moore et al., 2006), care had to be taken to exclude legitimate hosts from being interrogated. To help distinguish between the different types of IBR traffic, three new sub-categories were defined in Section 2.3, Potentially Malicious, Truly Malicious, and Result of Malicious traffic. An overview of fingerprinting and port scanning techniques were provided including geolocation based techniques. The need for careful selection of hosts to analyse was highlighted in Section 2.6 due to the potential legal implications associated with unsolicited port scanning. Characteristics from flooding attacks where a victim's responses would be captured by network telescopes were identified and filtered out by the rDSN modules deployed on the network telescopes. Due to the operation of honeypots, filtering of traffic was not required from rDSN modules deployed on the honeypot sensors.

Chapter 3

In Chapter 3 the hypothesis was made that if sufficiently different characteristics of a host could be identified that together they could act as unique fingerprint for that host. Potentially allowing for the re-identification of that host, even if the host's IPv4 address had changed. Four categories of fingerprinting were identified in Chapter 3, logical, physical, associative, and behavioural. The concepts of associative fingerprinting and Latency Based Multilateration were also introduced in Chapter 3 as a new metric for remote host fingerprinting.

Chapter 4

Previous research by Pemberton (2007) had shown that IBR traffic was not uniformly distributed across IP address space. This required that the tools developed to gather information from sensors such as network telescopes and honeypots would have to operate in a distributed nature with multiple sensors. The rDSN modules introduced in Chapter 4 were developed with this in mind, allowing for deployment on multiple honeypots and network telescopes and exposing aggregated

data from them in near real-time to a central server for analysis through a message queuing protocol. Interrogation of malicious hosts required the consumption of the data exposed to the central server, this was achieved through the development of the AR-Framework. The Multi-sensor Data Fusion model presented by Hall and Llinas (1997) and Waltz and Llinas (1990) was adapted and the AR-Framework developed from the adaptation. The framework would subscribe to the exposed message queues and make use of the data exposed to perform reconnaissance on the malicious hosts identified by sensors. Port scans, OS fingerprinting, geolocation, and open source data retrieval was performed on the identified malicious hosts. Three visualisation tools were demonstrated at the end of Chapter 4. These tools allowed for the exposure of data from the AR-Framework in near real-time and also provided various degrees of interaction to accommodate human understanding of the data. This included a near real-time visualisation, an interactive time-line visualisation and an aggregated metrics visualisation.

Chapter 5

Results gathered through the deployment of rDSN modules and the AR-Framework were presented in Chapter 5. The purpose of the chapter was to illustrate the successful application of rDSN modules and the AR-Framework in the real-world. An LBM feasibility study was conducted and determined that 67.42% of hosts responded to ICMP type 8 requests. An experiment in host re-identification through LBM fingerprints resulted in successful identification of hosts with an accuracy between 94% and 96% based on optimal threshold variables. Higher threshold values had to be used to insure a higher true positive rate while the increased false positive rate could be circumvented by progressive validation of other attributes of the observed host from the four categories of fingerprinting.

The ability of multiple sensors to detect geographically distributed malicious hosts proved successful, while also indicating differences based on the IP address space within in which the sensors were deployed and the size of the sensors. China was determined as the most prolific originating country for malicious traffic, while an interesting observation was made based on the United States based honeypot and targeted traffic.

Results pertaining to the most commonly targeted network ports were provided in Section 5.4. The three most commonly targeted ports on the network telescope sensor accounted for 82.84% of all observed traffic. Ports relating to administrative services were found to be the most targeted services across all of the sensors and included 22/tcp (SSH), 23/tcp (Telnet), 3389/tcp

(RDP), and 5900/tcp (VNC). A possible limitation of passive network telescopes was identified in the sense that they were not able to reply to MSSQL port discovery queries targeted towards 1433/udp. Traffic as a result of the Carna botnet was also observed and correlated with an increase in traffic to port 210/tcp as observed by the ICS Sans sensors (SANS, 2012).

An overview of malicious host operating system observations were provided in Section 5.5. The findings presented the most commonly found operating systems on malicious hosts as determined through active fingerprinting. The most common operating systems were found to different releases of FreeBSD, accounting for 53.65% of observed operating systems.

A case study presenting the open port combinations of malicious hosts along with correlations between these open ports and the different malicious host operating systems was conducted. This provided insight into the configuration of previously compromised hosts. Many of the ports exposed on these malicious hosts were also found in the top 10 lists for the most targeted services. The exposed ports also included administrative ports, while the port combinations in most cases correlated with what would be expected from the operating systems exposing those combinations. Targeted ports and exposed ports were then correlated in an investigation to determine the likelihood of a malicious host targeting the same service as it exposed. Of the 11.5% of malicious hosts with correlating open and targeted ports, 88.17% represented 3389/tcp with the second most common port being 445/tcp with only 5.71%. Eight out of the 10 most commonly targeted ports were in the list of correspondingly open and targeted ports. This correlation may indicate that the eight ports that correlated between the most targeted services and the matching targeted port and originating host port, were the ports used to initially compromise those hosts. As such the services exposed on these eight ports would likely be the most important services to secure in order to increase the defensive posture of a network.

6.2 Research Goals

The following research objectives were outlined in Chapter 1, an evaluation of each has been provided according to research presented in this work.

1. Develop a means of identifying and collecting malicious host traffic from distributed, heterogeneous source and expose that information in real-time for further research.

Chapter 2 discussed several sources of malicious traffic and how they could be detected through

the application of different sensors such as honeypots and network telescopes. Distinctions between Truly Malicious, Potentially Malicious and Result of Malicious traffic were also outlined in Chapter 2. This provided a means to distinguish between different forms of nefarious traffic and highlighted the potential legal implications should active reconnaissance against these sources be performed. To this end the rDSN modules were presented in Chapter 4, allowing for the collection of malicious host traffic from heterogeneous sources and exposing that data in near real-time through a message queuing protocol. It also lead to the inclusion of a traffic filtering mechanism to restrict Result of Malicious traffic from triggering active reconnaissance of hosts that have been victims of DDoS attacks.

2. Identify characteristics that could be remotely gathered and used to represent malicious hosts on the Internet.

Four categories for remote fingerprinting were identified in Chapter 3, each category detailed characteristics that could be used to represent a host. The categories were determined by evaluating sufficiently different aspects that when combined could provide a representation of a host. They included *Logical*, *Physical*, *Associative,* and *Behavioural* attributes. The characterstics within these categories included existing fingerprinting techniques such as OS fingerprinting and port scanning, while also included new methods.

3. Develop and evaluate new metrics that could be gathered from hosts in order to re-identify them in dynamic IP address space.

Two new methods of fingerprinting were introduced in Chapter 3, these methods were Fast-flux botnet association, that could potentially be used to represent an *Associative* attribute and Latency Based Multilateration to represent a *Physical* attribute. Fast-flux botnet association was ultimately discarded as it would only apply to a small population of malicious hosts on the Internet. LBM formed part of the *Physical* fingerprinting as it provided a logical representation of a host's physical location on the Internet, its use, however, required significant resources. The application of LBM was evaluated with a feasibility study in Chapter 5 which found that 67.42% of hosts tested had reliably responded to ICMP type 8 requests. The application of LBM towards host fingerprinting and re-identification achieved a 96% precision score with a threshold of 94ms, however, due to increasing false positive rates it was determined that it should be used as a supporting metric and not its own. Thus allowing for validation with other attributes, such as open ports, reverse DNS lookups and operating system of the targeted host.

4. Implement a method to interrogate and actively gather information from

malicious hosts on the Internet.

The AR-Framework was detailed in Chapter 4, with its design based off of the adaption of the MultiSensor Data Fusion model. The AR-Framework consumed data in near real-time from a message queue that was populated through the reployment of multiple rDSN modules. Visualisation tools where developed and shown in Chapter 4, which allowed for interaction with data that would be collected by the rDSN modules and the AR-Framework. Chapter 5 presented the actual data collected through the deployment of the AR-Framework, which lead to interesting correlations between characteristics of malicious hosts and their activity on the Internet. Thus affording the capability of actively interrogating malicious hosts on the Internet, in order to gather information from them. This included an analysis of the most commonly targeted ports, malicious host operating systems and correlations between open ports and the operating systems of malicious hosts. Furthermore, targeted ports and open ports on malicious hosts provided insight into potential defensive measures that could be taken to increase the security of hosts on the Internet.

6.3 Significance of Research

This work focused largely on the development of tools and frameworks to aid in the study of malicious hosts on the Internet. The rDSN modules were developed and successfully tested on honeypots and a network telescope. Through those modules it became possible to expose data from distributed sources, addressing the concerns raised by Pemberton (2007) of disproportionately targeted IPv4 address ranges and differences in observations by Shinoda et al. (2005) on separate sensors. It also enables researchers to overcome the different trends observed by separate network telescopes shown by Harder et al. (2006), thus allowing for a more holistic representation of data. The near real-time exposure of this data, directly addressed the problem of degrading accuracy of information over time and could enable new types of time-sensitive analysis and defensive measures to be implemented.

The AR-Framework was developed from an adapted MultiSensor Data Fusion model. This allowed for modularised development and processing of data, it could also easily be extending to accommodate additional information gathering techniques. It also allowed for the interrogation of malicious hosts detected through rDSN modules, providing a new perspective on the characteristics of malicious hosts, such as open ports and more detailed OS identification than allowed

with passive techniques.

The LBM fingerprinting technique presented in this research showed promise as an additional metric for host fingerprinting. Allowing hosts to be re-identified, even if their IP address had changed, as was shown with the blind testing presented in Section 5.2.2. Unfortunately LBM was found to require significant resources to be used on a large scale.

The visualisation techniques demonstrated in Chapter 4 provided not only near real-time information, but also allowed for interaction to explore the collected data over time. Showing that visualisation techniques are important to aggregate information and provide a means to easily interpret a large amount of information.

Finally the data collected trough the use of the rDSN and AR-Framework provided an interesting view on the malicious host demographic. Allowing for correlations to be drawn between common open ports and operating systems, location, and intent. In so doing, expanding our understanding of a subset of malicious hosts on the Internet today and enabling further research in the area.

6.4 Future Work

In conclusion of this research, several areas for expansion have become evident. These further areas of research include, extension of the available rDSN modules, the addition of a correlation and tracking engine to the AR-Framework, and an automated blacklist generation module for the AR-Framework. The purpose of these extensions would be to increase the number of attributes collected from malicious hosts on the Internet, while also increasing the number of sensors supported for collection. The addition of a correlation and tracking engine along with automated blacklist generation could extend the AR-Framework's ability beyond data collection and into the realm of active network defense.

After data was collected and analysed, a shortcoming of the AR-Framework was identified. The Session Manager of the AR-Framework only recording the first port targeted by a malicious host, limiting the detection of port scans or exploitation attempts against multiple ports in a small time frame. While this did not affect the work presented, this additional data could in future prove useful for further analysis alongside the AR-Framework's existing capabilities.

6.4.1 rDSN Modules

Extend the deployment capabilities of the rDSN modules to support additional sensors. As a proof of concept, only two rDSN modules were developed, the first could be applied to network telescopes, the later was use on Dionaea honeypots. While the implementation of the Dionaea rDSN module was sufficient for the purpose of exposing data from the honeypot, it was far from elegant. Dionaea provides data access functionality through an iHandler module, which would provide a much more efficient solution. This is one of the planned future extensions. Further modules could also be created for deployment on other honeypot solutions and IDSs.

Publishing data to the message queues from rDSN modules was considered a more generic approach. In order to create new modules, the event generation or logging mechanisms for the particular honeypot solution needs to be identified and publish capabilities included. If modification of events or logging mechanisms are not possible, the storage mechanisms could be used. As was done with the Dionaea rDSN module, periodically the rDSN module would inspect the Dionaea database for new entries, the relative data aggregated and submitted.

By extending the rDSN modules to support additional honeypot deployments and IDS solutions, it would allow for a much larger and heterogeneous deployment of monitoring sensors. This in turn would result in richer data collection.

6.4.2 AR-Framework Session Manager

The Session Manager was responsible for maintaining active sessions, creating new sessions and expiring in-active sessions. Information pertaining to currently active sessions that were updated as new packets or events were observed included packet or event count and the last seen time-stamp. The IPv4 address and targeted port were recorded when the first packet was received by the Session Manager and never thereafter (until the session expired). As a result, the Session Manager did not allow for the recording of additionally targeted ports for that active session. This allowed reconnaissance modules from the AR-Framework to only run once per malicious host session, however, this also resulted in the inability of the AR-Framework to store port-scans or multiple exploitation attempts detected within the same session by data sensors.

As such the Session Manager should be modified along with the current database structure, so in addition to updating the packet count and time-stamp when a new packet was received, a list of all targeted ports would also be recorded.

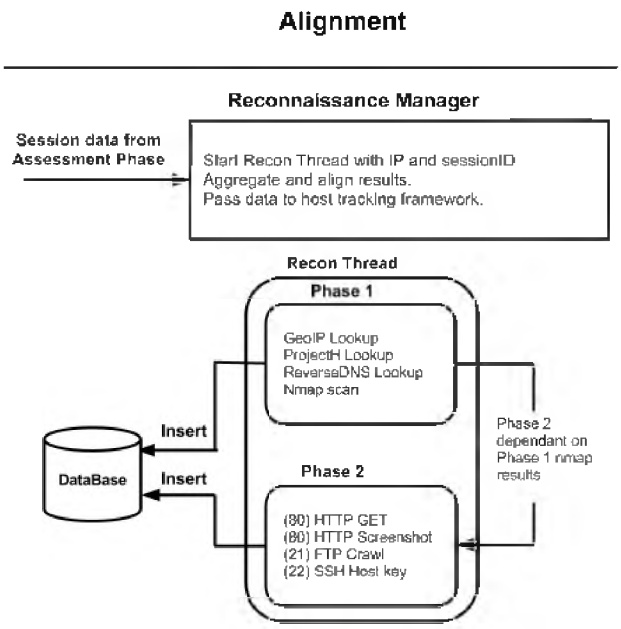

Figure 6.1: Addition of Phase 2 reconnaissance modules to AR-Framework

6.4.3 AR-Framework Reconnaissance Modules

Additional reconnaissance modules should be created in order to gather information from malicious hosts. Some of the modules that have been planned for inclusion into the AR-Framework include several conditional modules which would run in Phase 2 of the Recon Thread depicted in Figure 6.1. These modules would only run once a port or service correlating to the module had been determined as open or existing. For web based services an HTTP GET request should be performed and a screenshot of the presented web page gathered. For FTP, the module should attempt anonymous authentication and crawl files and directories recording any information found.

These additional reconnaissance modules would significantly increase the detail of information gathered by the AR-Framework on malicious hosts. However, the legal implications of these additional modules should be taken into consideration. For instance, due to legal constraints, it may not be possible to attempt anonymous authentication to an FTP service. Depending on the number of sensors in use, a study on the performance and resource requirements of these additional modules would also need to be conducted.

6.4.4 AR-Framework Correlation Engine

It could be advantageous to implement a correlation engine to re-identify hosts analysed by the AR-Framework in near real-time. A suggested approach for this correlation would be to consider each of the attributes collected over the four categories of fingerprinting identified in Section 3.2, *Logical*, *Physical*, *Associative*, and *Behavioural*. The first step would be the assignment of weighted values to each of the characteristics identified below, the purpose of which is to indicate their importance with regard to uniqueness or probability of change.

Logical - Open ports, Operating System family and version, Services and versions
Physical - Geolocation
Associative - ISP identifier (obtained from reverse DNS)
Behavioural - Up-time and/or latency delta over time delta

An example of differently weighted characteristics could be open ports, which might be weighted higher than version of services or operating system version, due to the likelihood of change over time. These attributes would be saved as a profile for a host based on a detected event from the AR-Framework, when a new event is detected the new profile is compared with all the stored profiles and the most likely matching profile entries are determined.

The efficiency of this may be improved through the use of *absolute truth changes* and progressive comparison. An example of an absolute truth change would be the comparison of operating system family, it would be inefficient to compare a profile that contains an OS family entry of Microsoft Windows with those of other OS families such as Linux or BSD.

By implementing a correlation engine alongside the AR-Framework, hosts could be re-identified in near real-time. Increasing the certainty that a given host has been seen before and indicating an increased threat from that host. This increased perceived threat would be particularly useful when considering the implementation of automated blacklist entries from a defensive standpoint.

6.4.5 Dynamic Blacklist Generation From AR-Framework Data

It has been shown that a significant proportion of malicious hosts are bound to more than one IP address over time. Wahid (2013) found that this was true for over 80% of the observed hosts during their study. This would indicate that dynamically generating a blacklist (or block-list) of hosts that have been identified with malicious intent could be of significant value over a static

blacklist. This blacklist would function in a similar manner as those often deployed to minimise email spam (Cook et al., 2006). It would also be possible to minimise the chance of blocking legitimate hosts by assigning more weight to malicious hosts detected on multiple occasions over time, while also removing IP addresses from the blacklist over time if they are no longer observed.

If a correlation engine is implemented alongside the AR-Framework, these "repeat offenders" could be easily identified and marked as an increased threat, in near real-time. If the threat is seen as numeric value, it could increase on each observation and gradually degrade over time, thus only adding the malicious host IP address to a blacklist once it exceeds some threat level threshold. Similar to the correlation engine's *absolute truth changes* or weighted attributes, certain types of detected events or attributes of hosts could be used to increase their perceived level of threat. For example, an event such as a port scan might be assigned a lower weight, while an exploitation attempt against a honeypot would be assigned a greater weight. At the same time, characteristics could be used to increase or decrease the threat, if a new host is detected and its OS Family is Microsoft Windows and it has the open port combination of (3389/tcp, 5900/tcp) open (see Figure 5.16) it should be ranked as potentially more malicious.

References

E. Aben. Conficker/Conficker/Downadup as seen from the UCSD Network Telescope. Online, 2009. URL `http://www.caida.org/research/security/ms08-067/conficker.xml`. Accessed 2016-12-17.

M. J. Arif. *Internet Host Geolocation Based On Probabistic Latency Models.* PhD thesis, University of Melbourne, 2010.

M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. Watson. The Internet Motion Sensor: A Distributed Blackhole Monitoring System. In *In Proceedings of Network and Distributed System Security Symposium (NDSS 05)*, pages 167–179, 2005a.

M. Bailey, E. Cooke, F. Jahanian, N. Provos, K. Rosaen, and D. Watson. Data Reduction for the Scalable Automated Analysis of Distributed Darknet Traffic. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, (IMC '05), pages 21–21, Berkeley, CA, USA, 2005b. USENIX Association. URL `http://dl.acm.org/citation.cfm?id=1251086.1251107`.

M. Bailey, E. Cooke, F. Jahanian, A. Myrick, and S. Sinha. Practical Darknet Measurement. In *Information Sciences and Systems, 2006 40th Annual Conference on*, pages 1496–1501. IEEE, 2006. doi: 10.1.1.85.6578.

R. J. Barnett and B. Irwin. Towards a Taxonomy of Network Scanning Techniques. In *Proceedings of the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing countries: riding the wave of technology*, SAICSIT '08, pages 1–7, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-286-3. doi: 10.1145/1456659.1456660.

T. Bass. Intrusion Detection Systems and Multisensor Data Fusion. *Communications. ACM*, 43 (4):99–105, Apr. 2000. ISSN 0001-0782. doi: 10.1145/332051.332079.

T. Bass. Multisensor Data Fusion for Next Generation Distributed Intrusion Detection Systems. In *In Proceedings of the IRIS National Symposium on Sensor and Data Fusion*, pages 24–27, 1999.

R. Beverly. A Robust Classifier for Passive TCP/IP Fingerprinting. In *Proceedings of the 5th Passive and Active Measurement (PAM) Workshop*, pages 158–167, Apr. 2004.

M. H. Bhuyan, D. Bhattacharyya, and J. Kalita. Surveying Port Scans and Their Detection Methodologies. *The Computer Journal*, 54(10):1565–1581, Oct. 2011. ISSN 0010-4620. doi: 10.1093/comjnl/bxr035.

H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Youssef, M. Debbabi, and L. Wang. On the Analysis of the Zeus Botnet Crimeware Toolkit. In *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference on*, pages 31–38. IEEE, 2010.

B. Blunden. *The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System*. Jones and Bartlett Publishers, Inc., USA, 2nd edition, 2013. ISBN 9781449626365.

S. M. Bridges and R. B. Vaughn. Fuzzy Data Mining and Genetic Algorithms Applied to Intrusion Detection. In *Proceedings of 12th Annual Canadian Information Technology Security Symposium*, pages 109–122, 2000.

J. Caballero, M. G. Kang, S. Venkataraman, D. Song, P. Poosankam, and A. Blum. FiG: Automatic Fingerprint Generation. In *In 14th Annual Network and Distributed System Security Conference (NDSS)*, 2007.

CAIDA. The UCSD Network Telescope. Online, December 2013. URL `http://www.caida.org/projects/network_telescope/`. Accessed 2015-12-17.

X. Chen and J. Heidemann. Detecting early worm propagation through packet matching. *ISI Tech. Report*, 585, 2004. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.8119&rep=rep1&type=pdf`.

A. Chittur. Model generation for an intrusion detection system using genetic algorithms. *High School Honors Thesis, Ossining High School. In cooperation with Columbia Univ*, 2001. URL `http://ids.cs.columbia.edu/sites/default/files/gaids-thesis01.pdf`.

F. B. Cohen. *A Short Course on Computer Viruses.* John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 1994. ISBN 0-471-00769-2.

D. Cook, J. Hartnett, K. Manderson, and J. Scanlan. Catching spam before it arrives: domain specific dynamic blacklists. In *Proceedings of the 2006 Australasian workshops on Grid computing and e-research-Volume 54*, pages 193–202. Australian Computer Society, Inc., 2006.

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 registery. Technical report, RFC 6335, 2011.

M. Cova, C. Kruegel, and G. Vigna. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In *Proceedings of the 19th international conference on World wide web*, pages 281–290. ACM, 2010.

T. Cymru. The Darknet Project. Online. URL `http://www.team-cymru.org/darknet.html`. Accessed 2017-11-19.

M. de Vivo, E. Carrasco, G. Isern, and G. O. de Vivo. A Review of Port Scanning Techniques. *SIGCOMM Computer Communications Review*, 29(2):41–48, Apr. 1999. ISSN 0146-4833. doi: 10.1145/505733.505737.

R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th Usenix Security Symposium*, 2004.

Dionaea Project. Dionaea - catches bugs. Online, 2012. URL `http://dionaea.carnivore.it/`. Accessed 2015-12-17.

A. Dunn. Unplugging a nation: State media strategy during Egypt's January 25 uprising. *The Fletcher Forum of World Affairs*, 35:15, 2011.

D. Evans. The Internet of Things. Online, April 2011. URL `http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf`. Accessed 2016-12-17.

FireEye. APT1: Exposing One of China's Cyber Espionage Units. Technical report, FireEye, 2013. URL `https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf`.

D. Fisher. Exploits Circulating for Remote Code Execution Flaws in NTP Protocol. Online, December 2014. URL `http://threatpost.com/exploits-circulating-for-remote-code-execution-flaws-in-ntp-protocol/110001`. Accessed 2016-12-16.

B. Forouzan, C. Coombs, and S. C. Fegan. *Introduction to Data Communications and Networking.* McGraw-Hill, Inc., New York, NY, USA, 1998. ISBN 0-256-23044-7.

D. Fried, K. Piwowarski, and W. Streilein. Passive operating system identification from TCP/IP packet headers. In *In Proceedings of the International Conference on Data Mining Workshop on Data Mining for Computer Security*, 2003.

L. Gao. On Inferring Autonomous System Relationships in the Internet. *IEEE/ACM Transactions on Networking*, 9:733–745, 2000. doi: 10.1109/90.974527.

J. Gomez and D. Dasgupta. Evolving fuzzy classifiers for intrusion detection. In *Proceedings of the 2002 IEEE Workshop on Information Assurance*, volume 6, pages 321–323. New York: IEEE Computer Press, 2002.

B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-based Geolocation of Internet Hosts. *IEEE/ACM Trans. Netw.*, 14(6):1219–1232, Dec. 2006. ISSN 1063-6692. doi: 10.1109/TNET.2006.886332.

D. Hall and J. Llinas. An Introduction to Multisensor Data Fusion. *Proceedings of the IEEE*, 85 (1):6–23, 1997. ISSN 0018-9219.

D. L. Hall. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Inc., Norwood, MA, USA, 1992. ISBN 0890065586.

U. Harder, M. W. Johnson, J. T. Bradley, and W. J. Knottenbelt. Observing Internet Worm and Virus Attacks with a Small Network Telescope. *Electronic Notes In Theoretical Computer Science*, 151(3):47–59, June 2006. ISSN 1571-0661. doi: 10.1016/j.entcs.2006.03.011.

S. Harris, A. Harper, C. Eagle, and J. Ness. *Gray Hat Hacking, Second Edition.* McGraw-Hill, Inc., New York, NY, USA, 2 edition, 2008. ISBN 0071495681, 9780071495684.

O. Henchiri and N. Japkowicz. A Feature Selection and Evaluation Scheme for Computer Virus Detection. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06,

pages 891–895, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2701-9. doi: 10.1109/ICDM.2006.4.

M. S. Hoque, M. A. Mukit, and M. A. N. Bikas. An Implementation of Intrusion Detection System Using Genetic Algorithm. *Computing Research Repository*, abs/1204.1336, 2012. doi: 10.5121/ijnsa.2012.4208.

S. Hunter, E. Stalmans, B. Irwin, and J. Richter. An Exploratory Framework for Non-Aggressive Response to Hostile Network Traffic. 01 2012a.

S. O. Hunter. A Network Telescope Information Visualisation Framework. *Honour's thesis, Rhodes University*, 2010. URL `http://www.cs.ru.ac.za/research/g07h3314/oldsite/resources/thesis.pdf`.

S. O. Hunter and B. Irwin. Tartarus: A honeypot based malware tracking and mitigation framework. In *Information Security for South Africa (ISSA), 2011*. ISSA, Pretoria, South Africa, 2011. ISBN 978-1-4577-1482-5.

S. O. Hunter, E. Stalmans, B. Irwin, and J. Richter. Remote fingerprinting and multisensor data fusion. In H. S. Venter, M. Loock, and M. Coetzee, editors, *Information Security for South Africa (ISSA), 2012*, pages 1–8. IEEE, 2012b. ISBN 978-1-4673-2160-0. doi: 10.1109/ISSA.2012.6320449.

S. O. Hunter, B. Irwin, and E. Stalmans. Real-time distributed malicious traffic monitoring for honeypots and network telescopes. In *Information Security for South Africa (ISSA), 2013*, pages 1–9. IEEE, 2013. doi: 10.1109/ISSA.2013.6641050.

IANA. Autonomous System Numbers, February 2014. URL `https://www.iana.org/assignments/as-numbers/as-numbers.xhtml`. Accessed 2016-06-27.

Insecure.org. First ruling by the Supreme Court of Finland on attempted break in. Online, 2003. URL `http://insecure.org/stf/fin.html`. Accessed 2015-12-17.

B. Irwin. A network telescope perspective of the Conficker outbreak. In *Information Security for South Africa (ISSA), 2012*, pages 1–8, Aug 2012. doi: 10.1109/ISSA.2012.6320455.

B. V. W. Irwin. *A Framework for the Application of Network Telescope Sensors in a Global IP Network*. PhD thesis, Rhodes University, 2011.

ISO. Country Code - ISO 3166. Online, 2013. URL `http://www.iso.org/iso/countrycodes.htm`. Accessed 2016-08-15.

E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe. Towards IP Geolocation Using Delay and Topology Measurements. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 71–84, New York, NY, USA, 2006. ACM. ISBN 1-59593-561-4. doi: 10.1145/1177080.1177090.

B. B. Kelly. Investing in a Centralized Cybersecurity Infrastructure: Why Hacktivism Can and Should Influence Cybersecurity Reform. *Boston University Law Review*, 92:1663, 2012.

M. Kenney. Ping of Death. Online, October 1996. URL `http://insecure.org/sploits/ping-o-death.html`. Accessed 2016-08-15.

H. Kim, I. Kang, and S. Bahk. Real-time visualization of network attacks on high-speed links. *Network, IEEE*, 18(5):30–39, 2004.

S. H. Kim, Q.-H. Wang, and J. B. Ullrich. A Comparative Study of Cyberattacks. *Communications of the ACM*, 55(3):66–73, 2012.

L. A. Klein. *Sensor and Data Fusion Concepts and Applications.* Society of Photo-Optical Instrumentation Engineers (SPIE), Bellingham, WA, USA, 2nd edition, 1999. ISBN 0819432318.

J. Klensin. Reflections on the DNS, RFC 1591, and Categories of Domains. Technical report, RFC 1591, 2001.

L. Koc, T. A. Mazzuchi, and S. Sarkani. A Network Intrusion Detection System Based on a Hidden Naive Bayes Multiclass Classifier. *Expert Systems with Applications*, 39(18):13492–13500, Dec. 2012. ISSN 0957-4174. doi: 10.1016/j.eswa.2012.07.009.

T. Kohno, A. Broido, and K. C. Claffy. Remote Physical Device Fingerprinting. In *IEEE Symposium on Security and Privacy*, pages 211–225. IEEE Computer Society, 2005.

N. Krawetz. Anti-honeypot technology. *Security & Privacy, IEEE*, 2(1):76–79, 2004.

J. A. Kunze, R. Denenberg, and D. Lynch. Uniform Resource Locators for Z39.50. Technical report, RFC 2056, 2013.

J. F. Kurose and K. W. Ross. *Computer Networking: A Top-Down Approach (4th Edition).* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2007. ISBN 0321497708.

E. Le Malécot and D. Inoue. The Carna Botnet Through the Lens of a Network Telescope. In *Foundations and Practice of Security*, pages 426–441. Springer, 2014.

C. B. Lee, C. Roedel, and E. Silenok. Detection and characterization of port scan attacks. Technical report, Department of Computer Science and Engineering, University of California, 2003.

W. Li. Using Genetic Algorithm for Network Intrusion Detection. *Proceedings of the United States Department of Energy Cyber Security Group*, pages 1–8, 2004. doi: 10.1.1.89.3125.

M. Ligh, S. Adair, B. Hartstein, and M. Richard. *Malware Analyst's Cookbook and DVD: Tools and Techniques for Fighting Malicious Code.* Wiley Publishing, 2010. ISBN 978-0470613030.

W. Lu and I. Traore. Detecting new forms of network intrusion using genetic programming. *Computational Intelligence*, 20(3):475–494, 2004.

W. Luo, J. Liu, J. Liu, and C. Fan. An analysis of security in social networks. In *Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on*, pages 648–651. IEEE, 2009.

G. Lyon. Nmap Network Scanning. Chapter 8, Remote OS Detection. Online, 2009. URL `http://nmap.org/book/osdetect-usage.html`. Accessed 2016-08-15.

Maltainfosec. The Concept of Intrusion Detection Systems. Online, 2012. URL `http://maltainfosec.org/archives/26-The-Concept-of-Intrusion-Detection-Systems.html`. Accessed 2016-01-02.

J. A. Marpaung, M. Sain, and H.-J. Lee. Survey on malware evasion techniques: state of the art and challenges. In *Advanced Communication Technology (ICACT), 2012 14th International Conference on*, pages 744–749. IEEE, 2012.

M. Middlemiss and G. Dick. Feature selection of intrusion detection data using a hybrid genetic algorithm/KNN approach. In *Design and Application of Hybrid Intelligent Systems*, pages 519–527. IOS Press, 2003.

MilkenInstitute. Currency of ideas insights from the institute. Online, May 2012. URL `http://www.milkeninstitute.org/newsroom/newsroom.taf?function=currencyofideas&blogID=462`. Accessed 2016-02-17.

C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan. Review: A Survey of Intrusion Detection Techniques in Cloud. *Network and Computer Applications*, 36(1):42–57, Jan. 2013. ISSN 1084-8045. doi: 10.1016/j.jnca.2012.05.003.

D. Moore, R. Periakaruppan, J. Donohoe, and k. claffy. Where in the world is netgeo.caida.org? In *International Networking Conference (INET) '00*, Yokohama, Japan, Jul 2000. The Internet Society.

D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the slammer worm. *IEEE Security & Privacy*, 1(4):33–39, 2003.

D. Moore, C. Shannon, G. Voelker, and S. Savage. Network Telescopes. Technical report, Cooperative Association for Internet Data Analysis (CAIDA), Jul 2004. URL `https://www.caida.org/publications/papers/2004/tr-2004-04/tr-2004-04.pdf`.

D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage. Inferring Internet denial-of-service activity. *ACM Transactions on Computer Systems*, 24(2):115–139, May 2006. ISSN 0734-2071. doi: 10.1145/1132026.1132027.

J. A. Morales, R. Sandhu, and S. Xu. Evaluating detection and treatment effectiveness of commercial anti-malware programs. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pages 31–38. IEEE, 2010.

J. A. Muir and P. C. Van Oorschot. Internet Geolocation and Evasion. Technical report, School of Computer Science, Carleton University, 2006.

J. A. Muir and P. C. Van Oorschot. Internet Geolocation: Evasion and Counterevasion. *ACM Comput. Surv.*, 42(1):4:1–4:23, Dec. 2009. ISSN 0360-0300. doi: 10.1145/1592451.1592455. URL `http://doi.acm.org/10.1145/1592451.1592455`.

C. Nachenberg. Computer Virus-antivirus Coevolution. *Commun. ACM*, 40(1):46–51, Jan. 1997. ISSN 0001-0782. doi: 10.1145/242857.242869. URL `http://doi.acm.org/10.1145/242857.242869`.

J. Nazario and T. Holz. As the net churns: Fast-flux botnet observations. In *2008 3rd International Conference on Malicious and Unwanted Software (MALWARE)*, pages 24–31. IEEE, Oct. 2008.

R. Olson. IP to Latitude/Longitude. Online. URL http://www.mcs.anl.gov/~olson/IPtoLL.html. Accessed 2016-03-22.

V. N. Padmanabhan and L. Subramanian. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '01, pages 173–185, New York, NY, USA, 2001. ACM. ISBN 1-58113-411-8. doi: 10.1145/383059.383073.

P. Paganini. Criminals exploited "Je suis Charlie" to spread Darkcomet Malware. Online, January 2015. URL http://securityaffairs.co/wordpress/32332/cyber-crime/criminals-je-suis-charlie-darkcomet.html. Accessed 2016-09-12.

R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson. Characteristics of Internet Background Radiation. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, IMC '04, pages 27–40, New York, NY, USA, 2004. ACM. ISBN 1-58113-821-0. doi: 10.1145/1028788.1028794.

S. Panjwani, S. Tan, and K. M. Jarrin. An Experimental Evaluation to Determine if Port Scans Are Precursors to an Attack. In *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, DSN '05, pages 602–611, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2282-3. doi: 10.1109/DSN.2005.18.

D. S. Pemberton. *An Empirical Study of Internet Background Radiation Arrival Density and Network Telescope Sampling Strategies.* PhD thesis, Victoria University of Wellington, 2007.

I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye. IP geolocation databases: Unreliable? *ACM SIGCOMM Computer Communication Review*, 41(2):53–56, 2011.

J. Postel. Internet Control Message Protocol, RFC 792. Technical report, RFC 792, 1981.

K. Poulsen. Port scans legal judge says. Online, December 2000. URL http://www.securityfocus.com/news/126. Accessed 2016-09-12.

M. Prawler. Jerusalem Magistrates Court. CC. 003047/03. Online, February 2004. URL http://www.law.co.il/media/computer-law/mizrachi_en.pdf. Accessed 2016-01-02.

N. Provos. Honeyd-a virtual honeypot daemon. In *10th DFN-CERT Workshop, Hamburg, Germany*, volume 2, 2003.

N. Provos. A Virtual Honeypot Framework. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 1–1, Berkeley, CA, USA, 2004. USENIX Association. doi: 10.1.1.126.2682.

N. Provos and T. Holz. *Virtual honeypots: from botnet tracking to intrusion detection.* Addison-Wesley Professional, 2007. ISBN 9780321336323.

K. Ramachandran and B. Sikdar. Modeling malware propagation in gnutella type peer-to-peer networks. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 8–pp. IEEE, 2006.

D. W. Richardson, S. D. Gribble, and T. Kohno. The Limits of Automatic OS Fingerprint Generation. In *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*, AISec '10, pages 24–34, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0088-9. doi: 10.1145/1866423.1866430.

A. H. Roman Danyliw. Code Red Worm Exploiting Bugger Overflow in IIS Indexing Service DLL. Online, January 2002. URL `http://www.cert.org/historical/advisories/CA-2001-19.cfm`. Accessed 2016-09-12.

SANS. TCP/UDP port activity. Online, 2012. URL `https://isc.sans.edu/port.html?port=210`. Accessed 2016-09-12.

R. Shams, M. Farhan, S. A. Khan, and F. Hashmi. Comparing Anti-Spyware products - A different approach. In *Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International*, volume 1, pages 75–80. IEEE, 2011.

C. Shannon and D. Moore. The Spread of the Witty Worm. *IEEE Security and Privacy*, 2(4): 46–50, July 2004. ISSN 1540-7993. doi: 10.1109/MSP.2004.59.

Y. Shinoda, K. Ikai, and M. Itoh. Vulnerabilities of Passive Internet Threat Monitors. In *Proceedings of the 14th Conference on USENIX Security Symposium - Volume 14*, SSYM'05, pages 14–14, Berkeley, CA, USA, 2005. USENIX Association. doi: 1251398.1251412.

G. Shu and D. Lee. Network Protocol System Fingerprinting - A Formal Approach. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 23-29 April 2006, Barcelona, Catalunya,*

Spain, 2006. doi: 10.1109/INFOCOM.2006.157. URL https://doi.org/10.1109/INFOCOM.2006.157.

P. Shukla. The Compromised Devices of the Carna Botnet. 10(2):547–627, 2015. ISSN 2192-4260. URL http://www.sicherheitsforschung-magdeburg.de/publikationen/journal.html.

S. Singh, C. Estan, G. Varghese, and S. Savage. Automated Worm Fingerprinting. In *Proceedings of the 6th Conference on Symposium on Operating Systems Design & Implementation - Volume 6*, (OSDI'04), pages 4–4, Berkeley, CA, USA, 2004. USENIX Association.

M. Smart, G. R. Malan, and F. Jahanian. Defeating TCP/IP stack fingerprinting. In *Proceedings of the 9th conference on USENIX Security Symposium - Volume 9*, (SSYM'00), pages 17–17, Berkeley, CA, USA, 2000. USENIX Association.

T. Sochor and M. Zuzcak. *Study of Internet Threats and Attack Methods Using Honeypots and Honeynets*, chapter 1, pages 118–127. Springer, 2014.

Y. Spiegel. Commercial software, adware, and consumer privacy. *International Journal of Industrial Organization*, 31(6):702–713, 2013.

L. Spitzner. Open source honeypots: Learning with Honeyd, 2003. URL https://www.symantec.com/connect/articles/open-source-honeypots-learning-honeyd. Accessed 2016-12-14.

E. Stalmans, S. O. Hunter, and B. Irwin. Geo-spatial autocorrelation as a metric for the detection of Fast-Flux botnet domains. In *Information Security for South Africa (ISSA), 2012*, pages 1–7. IEEE, 2012. doi: 10.1109/ISSA.2012.6320433.

S. Staniford, J. A. Hoagland, and J. M. McAlerney. Practical automated detection of stealthy portscans. *Journal of Computer Security*, 10(1):105–136, 2002a.

S. Staniford, V. Paxson, and N. Weaver. How to Own the Internet in Your Spare Time. In *Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, USA, August 5-9, 2002*, pages 149–167, 2002b. URL http://www.usenix.org/publications/library/proceedings/sec02/staniford.html.

S. J. Stolfo, W. Lee, P. K. Chan, W. Fan, and E. Eskin. Data mining-based intrusion detectors: an overview of the columbia IDS project. *ACM SIGMOD Record*, 30(4):5–14, 2001.

Symantec. W32.Welchia.Worm. Online, February 2007. URL http://www.symantec.com/security_response/writeup.jsp?docid=2003-081815-2308-99. Accessed 2016-10-11.

Symantec. Smurf DoS attacks. Online, 2012 2012a. URL http://www.symantec.com/security_response/glossary/define.jsp?letter=s&word=smurf-dos-attack. Accessed 2016-10-09.

Symantec. Symantic Threat Report. Online, April 2012b. URL http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_2011_21239364.en-us.pdf. Accessed 2016-10-04.

Symantec. What is the diffrence between a virus, worm and Trojans? Electronic, January 2009. URL http://www.symantec.com/business/support/index?page=content&id=TECH98539. Accessed 2016-10-11.

P. Szor. *The Art of Computer Virus Research and Defense*. Addison-Wesley Professional, 2005. ISBN 0321304543.

Team Cymru. The darknet project. Online, June 2004. URL http://www.cymru.com/Darknet/index.html. Accessed 2016-10-11.

TechCenter. Mictrosoft Security Bulletin MS08-067 - Critical. Online, October 2008. URL http://technet.microsoft.com/en-us/security/bulletin/ms08-067. Accessed 2016-12-04.

A. ter Kuile. Multilateration - MLAT in Action. Online, December 2009. URL http://www.multilateration.com/surveillance/multilateration.html. Accessed 2016-08-15.

Unknown. Internet Census 2012, 2012. URL http://internetcensus2012.bitbucket.org/paper.html. Accessed 2016-12-14.

C. van der Walt. When the tables turn, 2004. URL http://www.blackhat.com/presentations/bh-asia-04/bh-jp-04-pdfs/bh-jp-04-sensepost/bh-jp-04-sensepost.pdf. Accessed 2016-10-11.

J.-P. van Riel and B. Irwin. InetVis, a visual tool for network telescope traffic analysis. In *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 85–89. ACM, 2006.

N. Virvilis and D. Gritzalis. The big four-What we did wrong in advanced Persistent Threat detection? In *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*, pages 248–254. IEEE, 2013.

D. Wagner and P. Soto. Mimicry Attacks on Host-based Intrusion Detection Systems. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 255–264, New York, NY, USA, 2002. ACM. ISBN 1-58113-612-9. doi: 10.1145/586110.586145.

A. Wahid. *Estimating the Internet malicious host population while preserving privacy.* PhD thesis, University of Melbourne, 2013. URL `http://hdl.handle.net/11343/38139`.

E. L. Waltz and J. Llinas. *Multisensor Data Fusion.* Artech House, Inc., Norwood, MA, USA, 1990. ISBN 0890062773.

Y. Wang, S. Wen, Y. Xiang, and W. Zhou. Modeling the propagation of worms in networks: A survey. *Communications Surveys & Tutorials, IEEE*, 16(2):942–960, 2014.

M. Ward. Tuning in to the background hum of the net. Online, November 2010. URL `http://www.bbc.co.uk/news/technology-11863294`. Accessed 2016-12-04.

N. Weaver, V. Paxson, S. Staniford, and R. Cunningham. A taxonomy of computer worms. In *Proceedings of the 2003 ACM workshop on Rapid malcode*, pages 11–18. ACM, 2003.

T. Weber. Criminals 'may overwhelm the web'. Online, January 2007. URL `http://news.bbc.co.uk/2/hi/business/6298641.stm`. Accessed 2016-12-04.

B. Wong, I. Stoyanov, and E. G. Sirer. Octant: A Comprehensive Framework for the Geolocalization of Internet Hosts. In *Proceedings of the 4th USENIX Conference on Networked Systems Design and Implementation*, NSDI'07, pages 23–23, Berkeley, CA, USA, 2007. USENIX Association.

E. Wustrow, M. Karir, M. Bailey, F. Jahanian, and G. Huston. Internet Background Radiation Revisited. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 62–74, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0483-2. doi: 10.1145/1879141.1879149.

V. Yegneswaran, P. Barford, and J. Ullrich. Internet intrusions: global characteristics and prevalence. *ACM SIGMETRICS Performance Evaluation Review*, 31(1):138–147, 2003.

Y. Zhou and X. Jiang. Dissecting android malware: Characterization and evolution. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 95–109. IEEE, 2012.

T. Zseby, A. King, M. Fomenkov, and K. Claffy. Analysis of Unidirectional IP Traffic to Darkspace with an Educational Data Kit. Online, Feb 2014. URL `https://www.caida.org/publications/papers/2014/analysis_unidirectional_ip_traffic/analysis_unidirectional_ip_traffic.pdf`. Accessed 2017-05-04.