Design of a Hybrid Command and Control Mobile Botnet

by

Heloise Pieterse

Submitted in fulfilment of the requirements for the degree
Master of Science (Computer Science)
in the Faculty of Engineering, Built Environment and Information Technology
University of Pretoria, Pretoria

July 2014

Publication data:

Heloise Pieterse. Design of a Hybrid Command and Control Mobile Botnet. Master's dissertation, University of Pretoria, Department of Computer Science, Pretoria, South Africa, July 2014.

Electronic, hyperlinked versions of this dissertation are available online, as Adobe PDF files, at:

http://icsa.cs.up.ac.za/

http://upetd.up.ac.za/UPeTD.htm

Design of a Hybrid Command and Control Mobile Botnet

by

Heloise Pieterse
Email: heloisep085@gmail.com

Abstract

Mobile devices have excelled in the 21st century due to the increasing popularity and continuous improvement of mobile technology. Today mobile devices have become all-in-one portable devices, providing interconnectivity, device-to-device communication and the capability to compete with personal computers. The improved capabilities and popularity of mobile devices have, however, caught the attention of botnet developers, allowing the threat of botnets to move into the mobile environment. A mobile botnet is defined as a collection of compromised mobile devices, controlled by a botmaster through a command and control (C&C) network to serve a malicious purpose. Previous studies of mobile botnet designs focused mostly on the C&C structure, investigating other mechanisms as potential C&C channels. None of these studies dealt with the use of a hybrid C&C structure within a mobile botnet design. This research consequently examines the problem of designing a new mobile botnet that uses a hybrid C&C structure. A model of this new hybrid design is proposed, describing the propagation vectors, C&C channels, and the topology. This hybrid design, called the Hybrid Mobile Botnet, explores the efficiency of multiple C&C channels against the following characteristics: no single point of failure must exist in the topology, low cost for command dissemination, limited network activities and low battery consumption per bot. The objectives were measured by using a prototype built according to the Hybrid Mobile Botnet model. The prototype was deployed on a small collection of mobile devices running the Android operating system. In addition, the prototype allowed for the design of a physical Bluetooth C&C channel, showing that such a channel is feasible, able to bypass security and capable of establishing a stealthy C&C channel. The successful execution of the prototype shows that a hybrid C&C structure is possible, allowing for a stealthy and cost-effective design. It also revels that current mobile technology is capable of supporting the development and execution of hybrid mobile botnets. Finally, this dissertation concludes with an exploration of the future of mobile botnets and the identification of security steps users of mobile devices can follow to protect against their attacks.

Keywords: mobile, botnets, mobile botnet, command and control, hybrid, mobile devices, smartphones, tablet computers

Supervisor	**:**	Prof. M. S. Olivier
Department	**:**	Department of Computer Science
Degree	**:**	Master of Science

Acknowledgements

My sincerest thanks to:

- The Almighty, for providing me with the skills, opportunity and strength to complete this work.
- Professor Martin Olivier, for his professional insight, guidance and support.
- To my parents, Marius and Ria Pieterse, for their continued love, encouragement and support.
- Joey Janse van Vuuren, Professor Marthie Grobler and Dr Renier van Heerden, for giving me this opportunity to perform this research, providing valuable guidance and input along the way.
- Glenda Buncombe, for valuable assistance and recommendations by editing this dissertation.
- Friends and colleagues at the Cyber Defence Research Group, DPSS, CSIR, for their continued advice and support.
- Council for Scientific and Industrial Research, for their financial support.

Contents

Chapter 1

Introduction

The last few years have seen a revolution in the development of mobile devices, transforming the devices from basic voice and text functionalities to all-in-one portable devices that represent functionality similar to that of a traditional computer. Today's mobile devices support advanced functionalities, offering interconnectivity capabilities such as Internet access and device-to-device communication while also supporting the execution of various applications. The improvement in mobile device technology and the popularity associated with these devices have caught the attention of malware developers. One of the new threats that current users of mobile devices face is botnets.

Botnets are a well-known threat to users of personal computers and computer networks. They are responsible for the delivery of spam and cause severe damage through distributed denial of service (DDoS) attacks (Grizzard et al. 2007). Botnets commonly make use of a C&C server and can communicate via internet relay chat (IRC) and peer-to-peer (P2P) channels (Giroire et al. 2009). Such a botnet is regarded as a traditional botnet.

Over the last few years there has been a sharp increase in mobile malware (*Mobile Threat Report Q3 2012* 2012) and the expected trend for the future is the continuous growth of mobile malware (*Trends for 2013* 2013). A constant improvement in mobile malware allow developers the opportunity to introduce the concept of traditional botnets to mobile devices. A mobile botnet is thus a network consisting of a collection of compromised mobile devices, controlled by a botmaster, through a C&C network. Due to the popularity of mobile devices and the constant rise in mobile malware, it is only a matter of time before mobile botnets become the frontrunner of the mobile malware evolution.

Section 1.1 sets out the motivation for this research, Section 1.2 outlines the problem statement of this study and Section 1.3 sets out the primary objectives of this study as well as the methodology that was followed to achieve these objectives.

1.1 Motivation

The concept of mobile botnets is not new to the research community as it was first discovered in 2009. During 2009, security analysts detected two new malware variants, Symbian's Yxes (Apvrille 2012) and the ikee.B malware for Apple's iPhones (Porras et al. 2010), which both supported functionalities similar to traditional botnets. Since the discovery of this malware, researchers started to further explore the concept of mobile botnets but there is still a dearth of research in this particular field.

The publications discussed below focus mostly on new mobile botnet designs. Singh et al. (2010) evaluated the suitability of Bluetooth as a C&C channel through various simulations. Zeng, Shin and Hu (2012) proposed a mobile botnet that uses short message service (SMS) for C&C and follows a P2P structure, while Geng et al. (2012) designed an SMS-based heterogeneous mobile botnet. The Andbot, designed by Xiang et al. (2011), uses a centralised topology and the hyper text transfer protocol (HTTP) protocol to disseminate commands while, the SoCellBot, designed by Faghani and Nguyen (2012), uses online social networks to accomplish the same task. Zhao et al. (2012) introduced the concept of cloud computing in mobile botnet designs while Shuai et al. (2013) expanded and improved the Andbot design. Li et al. (2013) designed a Twitter and SMS-based mobile botnet while the JokerBot, designed by Sheu et al. (2013) is an Android-based botnet. Mulliner (2011) provided an iPhone-based mobile botnet using a combination of SMS and HTTP for C&C.

By evaluating the mobile botnet designs currently available (such as those above), it becomes clear that the C&C channels are the focal point of the studies since most explore new mechanisms for C&C. That is because the C&C channels form the core of any mobile botnet and allow for the efficient dissemination of commands. Without such channels, the mobile botnet will become a meaningless entity to the botmaster.

Many traditional C&C channels, such as those based on HTTP, are reusable within the designs of mobile botnets. Previous designs have explored SMS and Bluetooth as potential C&C channels. However, researchers have not yet explored the use of a hybrid C&C structure to disseminate commands in current mobile botnet designs. A hybrid C&C structure refers to the use of three or more C&C channels and can make a mobile botnet harder to detect and allows for improved robustness.

Without knowledge of mobile botnets and their potential capabilities, it will become difficult to protect against their attacks. Such knowledge can be obtained through the experience gained by designing and implementing a new hybrid mobile botnet. Throughout this study specific information about mobile botnets, such as their potential attack strategies and protection measures, can be identified.

With mobile devices becoming more widespread and popular, especially in sensitive environments (e.g. the military), mobile botnet developers will start exploiting mobile devices to perform malicious activities. Without proper knowledge of the capabilities of mobile botnets and their potential threats, designing defence and detection mechanisms to protect against their attacks will be difficult.

1.2 Problem Statement

This research consequently examines the problem of designing a mobile botnet with a hybrid C&C structure to show that current mobile technology exhibits all the necessary capabilities needed for the development and support of hybrid mobile botnet designs.

The problem, as stated above, can be summarised by addressing the following subproblems:

- Allowing for interconnectivity within the hybrid C&C structure.
- Designing a flexible topology structure that can incorporate all the C&C channels as well as the mobility of the devices.

From the newly designed mobile botnet the required information and knowledge necessary for designing detection and defensive mechanisms are obtained.

The following section provides more detail about the objectives of this research as well as the methodology that was followed to achieve them.

1.3 Objectives and Methodology

The objective of this study was to show that current mobile technology exhibit all the necessary capabilities to support the development and execution of hybrid mobile botnets. The primary objectives of the study were:

- To design a new mobile botnet that uses a hybrid C&C structure.
- To develop a prototype of this new design and execute the prototype on real mobile devices.

In order to achieve the above objectives, a model is proposed that sets out the specific structure of the new mobile botnet. The efficiency of a hybrid design was evaluated by building a prototype of the model. To determine if the hybrid design was stealthy, cost-effective and robust, the prototype was tested against the following characteristics: no single point of failure within the topology, low cost for command dissemination, limited network activities and low battery consumption.

After the design of the model and the evaluation of the prototype, it became possible to identify potential mobile botnet attack strategies and security measures to protect against such attacks and to evaluate the future of mobile botnets.

1.4 Dissertation Outline

- Traditional botnets and their characteristics are explored in **Chapter 2**. Firstly, the components of a traditional botnet are explained. Then the reasons behind the development of traditional botnets are examined closely. The consequences of botnet attacks and the potential detection mechanisms to prevent such attacks are reviewed.

- **Chapter 3** contains a description of the evolution of mobile devices and provides an overview of mobile malware. Various mobile operating systems are explored, and their internal design and acceptance of applications evaluated.

- **Chapter 4** provides an overview and brief history of mobile botnets. Firstly, the definition and characteristics of mobile botnets are explained. Secondly, the differences and similarities between traditional botnets and mobile botnets are explored. Thirdly, the potential threats and capabilities of mobile botnets are identified and previously proposed countermeasures that can possibly defend against mobile botnet attacks are presented. Finally, research relating to the design of mobile botnets is discussed.

- A new mobile botnet model which uses multiple C&C channels is proposed in **Chapter 5**. Each component of the mobile botnet, namely propagation vectors, C&C channels and the topology, is explained individually. The process of command dissemination, as followed by the new mobile botnet model, is discussed and analysed step by step.

- The potential attack strategies of mobile botnets and their implementation in the newly designed mobile botnet are discussed in **Chapter 6**. The attack strategies are grouped according to the following categories: monetary gain, information dispersal, information reaping and service interruption.

- In **Chapter 7** the design and implementation of the prototype are described. The prototype acts as proof of concept for the newly designed mobile botnet. In the first section of the chapter, the internal design of the prototype is explained, as is the equipment used in the development of the prototype. The second section presents the execution of the prototype by focusing on two separate demonstrations: command dissemination and execution and the performance of the Bluetooth C&C channel. The the prototype is evaluated in the third section, while the final section concludes with the strengths and weaknesses of the newly designed mobile botnet.

- **Chapter 8** acts as a conclusion chapter by summarising the knowledge gained throughout this study. The growth potential of mobile botnets is examined and security steps that users of mobile devices can take to protect against mobile botnet infections are identified.

- **Chapter 9** concludes the dissertation by briefly summarising the major contributions and findings. Areas are proposed that can be considered for future research.

The following appendices are included:

- **Appendix A** lists the publications derived from this work.
- **Appendix B** lists and defines all the abbreviations used in this work.

Chapter 2

Traditional Botnets

For the past decade botnets (often referred to as traditional botnets throughout this dissertation) have been one of the most serious threats for users of the Internet and personal computers. The development of countermeasures is constantly lagging behind as botmasters improve and develop new botnet techniques. The evolution of botnets dates as far back as 1999, when the first botnets were invented (Ferguson 2010). During that year, Sub7 and Pretty Park introduced a new concept of connecting to an IRC channel and listening for commands (Ferguson 2010). These malware samples sparked the botnet innovation and during the next decade many formidable botnets surfaced, including SpyBot, Zeus, Storm and Conficker (Ferguson 2010). To this day botnets still remain an active security threat.

This chapter introduces the terminology and characteristics of botnets. The focus is on the main components of a botnet: propagation, command and control channels, and topology. Each component is critical to the construction of a botnet and is discussed in detail in this chapter. The rest of the chapter is structured as follows: Section 2.1 provides a discussion of the terminology of botnets while Section 2.2 contains a description of the various botnet characteristics and attacks. The main components of a botnet are discussed in Section 2.3. In Section 2.4 the reasons behind the development of botnets are explored while the consequences of botnet attacks are set out in Section 2.5. A closer look is taken in Section 2.6 at various botnet detection mechanisms, such as honeypots and intrusion detection systems. A short case study about the Zeus botnet is presented in Section 2.7. Section 2.8 concludes the chapter.

2.1 Terminology

A bot, short for robot, is an autonomously running computer program or bot program (Elliott 2010) and is responsible for turning the computer into a controllable entity. These bot programs are distributed to a large number of vulnerable personal computers to eventually form a loosely coordinated network of bots, also called a botnet (Pfleeger and Pfleeger 2006). A botnet is thus a collection of infected computers (Schiller 2007) that is compromised by running the bot program (Elliott 2010). The bot program exploits certain vulnerabilities on the targeted computers and allows a botmaster (also know as a botherder) to control these computers remotely (Mukamurenzi 2008, Wang, Fang, Zhang and Li 2009). The compromised computers running the bot program are also referred to as botclients and a typical botnet consists of one or more botclients (Schiller 2007). The botmaster is the attacker responsible for controlling and issuing commands to the botclients via a C&C infrastructure (Elliott 2010). The C&C infrastructure consists of one or more servers, developed and managed by the botmaster. The C&C server or bot server receives the commands from the botmaster and then disseminates the commands to all the botclients. The botclients are then responsible for executing an attack according to the interpretation of the received command.

A botmaster creates and maintains a botnet through five phases (Feily et al. 2009). These phases are (Feily et al. 2009):

- Initial infection phase: During this phase the botmaster scans a target for certain known vulnerabilities. After selecting a vulnerability, the botmaster uses different exploitation methods to compromise the computer.

- Secondary infection phase: After successfully compromising a vulnerable personal computer, the next step is to infect that computer with the bot program.

- Connection phase: The bot program is responsible for establishing a C&C channel during this phase.

- C&C phase: When the bot program has successfully established the C&C channel, the dissemination of the commands can begin.

- Update and maintenance phase: During the lifetime of the botnet the botmaster must be able to continuously update and maintain the botnet in order to protect against new detection mechanisms.

Each phase plays a vital role in the overall success of the botnet. Should the construction of any of these phases fail, the botnet will also possibly fail or be easily compromised.

The step-by-step creation of botnets has led to the discovery of multiple characteristics. It is these characteristics that allow for a stealthy botnet design, capable of multiple attack opportunities.

2.2 Botnet Characteristics and Attacks

The continuous success of botnets is due to their adaptive infrastructures, the ability of the attacks to be targetable and the melding of multiple threats into one structure (Schiller 2007). Botmasters logically aim to sustain their botnets for as long as possible and therefore secrecy is a critical characteristic (Singh et al. 2010). To achieve secrecy, botnets deploy obfuscation techniques such as polymorphism and metamorphism (Zeng 2012). These techniques cause the bot programs to differ within the same botnet and can overcome signature-based detection techniques. Obfuscation techniques allow the botmaster to mutate the bot code without changing the internal workings of the commands (Zeng 2012). Obfuscation techniques increase the difficulty of detecting botclients participating in a botnet.

The bot program, which can spread via a virus or worm, allows the botnets to become self-propagating entities and once unleashed will continue to grow without any limits or until an adequate size has been reached (Feily et al. 2009). The strength of the botnet lies in the fact that the botclients are not physically owned by the botmasters and can originate in multiple locations all over the world (Feily et al. 2009).

The botclients within a specific botnet must be able to perform actions according to the commands received (Schiller 2007). These commands must be transported to the botclients without the necessity of directly logging into a compromised computer to retrieve the commands (Schiller 2007). The botclients must also act together as a unified structure in order to accomplish a common goal (Schiller 2007). It is the responsibility of the botmaster to determine the common goal which is often characterised by the attack the botmaster wishes to perform. Such attacks can include, but are not limited to, DDoS attacks, installation of additional files, spamming, hosting phishing sites, information stealing and compromising other uninfected computers (Schiller 2007).

Botnets are well equipped for these types of attacks. Firstly, the scattered distribution of the botclients can possibly offer protection to the identity of the botmaster. Secondly, the large collection of botclients that are usually present in a botnet also improve the performance and execution of a specific attack. Finally, by infecting such a large collection of computers the botmaster has the opportunity to misuse the processing power and hard disk space of these computers. This saves the botmaster both time and money, and can lead to more powerful and evasive attacks.

The primary goals of a botnet are divided into the following categories: information dispersion, information harvesting and information processing (Grizzard et al. 2007). Information dispersion refers to the spreading of information by means of spam or DDoS attacks. In contrast, information harvesting focuses on the collection of sensitive information such as personal data, financial data, passwords, or any other data the botmaster may require. Botnets can also process large quantities of data such as cracking passwords which are stored as message digest 5 (MD5) hashes. These categories are not mutually exclusive and it is possible for the botmaster to pursue goals that fall into more than one category.

Many characteristics discussed throughout this section focus on increasing the difficulty of detecting botnets and improving their attacks. To ensure that the above characteristics are achievable and that the botnet attacks are possible, the components of a botnet must be constructed successfully and must work in unison.

2.3 Components of a Botnet

A botnet requires three main components to operate successfully (Geng et al. 2011). These components are responsible for building and sustaining the ever-growing botnet and include the propagation of the bot program to recruit new victims, providing the mechanisms to allow communication with the botclients and obfuscating themselves from detection and attacks (Maheshwari 2012). Throughout this dissertation these three components are referred to as propagation, C&C channels, and topology (Geng et al. 2011). The upcoming sections focus on each of these components individually and provide a brief overview of the component.

2.3.1 Propagation

Propagation refers to the mechanisms that allow for the dissemination of the bot program to computers (Zeng 2012). Common propagation mechanisms to propagate the bot program include vulnerability exploits and social engineering (Zeng 2012, Bailey et al. 2009). The exploitation of vulnerabilities is a tool often utilised by botmasters to propagate the bot program. Potential vulnerability exploits include vulnerabilities found in operating systems, web browsers (such as drive-by downloads), services, applications and remote code execution in network services (Bailey et al. 2009, Provos et al. 2007). By exploiting vulnerabilities, malware such as viruses, worms and Trojan horses can create a back door on an infected computer that will allow a botmaster to install and run the bot program when required (Bailey et al. 2009).

Another popular propagation mechanism is social engineering which requires user involvement to propagate the bot program (Zeng 2012, Bailey et al. 2009). Spam emails are an example of a propagation mechanism which, in some instances, exploits human vulnerabilities to influence the readers of the email to click on an embedded link or attachment. The embedded link or attachment is malicious and contains links pointing to websites hosting the malicious bot program (Zeng 2012, Maheshwari 2012). Upon downloading the attachment or following the link, the malicious content will download and install the bot program on the recipient's personal computer. Another method involving social engineering is to trick the users into installing the bot program by attaching the program to a crack tool for popular games or software applications (Maheshwari 2012).

Propagation is a critical component to the botnet structure and without successful propagation mechanisms, the botnet will remain relatively small and insignificant. As software for personal computers evolves, new vulnerabilities become available for exploitation. With humans remaining the weakest link in the security chain, the botmaster will always have a possible propagation mechanism at hand.

2.3.2 Command and Control Channels

The second component of a botnet is the C&C channels. This particular component is responsible for establishing channels that will allow for the dissemination of commands to the botclients (Zeng 2012). The C&C channel must provide fast and secure communication between a botclient and the botmaster, generally established as a direct connection. Since the transmission control protocol (TCP)/internet protocol (IP) failed to meet these requirements, botmasters looked to other options and soon found that IRC channels provide a safe and secure channel for communication. As IRC-based C&C channels became compromised, botmasters started exploring other possibilities. The popular substitutes explored include domain name system (DNS), HTTP and P2P.

2.3.2.1 IRC C&C

For a substantial part of the botnet evolution, botnet technology has been based on IRC (Schiller 2007). IRC is a text-based open protocol that enables a user to teleconference (Goel et al. 2006) and communicate in real time on the Internet (Ferguson 2010). The first botnets based their technology completely on IRC and during May 1999 the first malicious IRC botnet, called Pretty Park, was discovered (Ferguson 2010). In the early days of botnet technology, IRC provided multiple advantages: interactivity, ease of creation and redundancy (Schiller 2007). When defenders started taking down IRC C&C botnets, botmasters identified a serious flaw with IRC C&C. IRC C&C botnets suffer from a single point of failure (the IRC C&C server) due to the centralised structure (Schiller 2007). Removing the IRC C&C server will bring down the entire botnet. To solve the problem, botmasters deployed multiple IRC C&C servers that were interconnected using IRC server technology, rather than deploying stand-alone servers (Schiller

2007). For a long period IRC proved to be an adequate solution for C&C but the problems surrounding single point of failure caused botmasters to explore other technology.

2.3.2.2 HTTP C&C

To overcome the problem of IRC traffic being easily monitored, botmasters turned to HTTP to use as a C&C channel and so increase the difficulty of detecting the bot traffic (Zeidanloo and Manaf 2010). HTTP is an application-level protocol responsible for distributing data on the Internet (Fielding et al. 1999) between World Wide Web clients and servers (Mah 1997). The advantage provided by the HTTP protocol is the possibility of hiding botnet traffic in the content of normal web traffic (Zeidanloo and Manaf 2010). The botnet traffic is thus formulated as normal HTTP traffic. Botnets using HTTP for C&C also do not maintain the connection with the C&C server, making them difficult to detect (Lee et al. 2008). HTTP C&C provides the ability to bypass firewalls and intrusion detection systems easily (Zeidanloo and Manaf 2010). The internal workings of the HTTP C&C are as follows: a botmaster is responsible for publishing a new command on a web page while the botclients periodically visit this page via HTTP to retrieve any new commands (Wang, Wu, Aslam and Zou 2009). Botnets using HTTP for C&C form a centralised structure, allowing the flaw of single point of failure, similar to IRC C&C, to still be a concern.

2.3.2.3 DNS C&C

DNS provides a mechanism for naming resources in a certain structure so that the names are reusable in different hosts (Mockapetris 1987, Mockapetris and Dunlap 1988). Although not designed to act as a communication channel, botmasters explored with the capabilities of DNS and found that the protocol can indeed be utilised as a C&C channel. Xu et al. (2013) proposed a pure DNS C&C channel that is compatible with existing DNS infrastructure. To construct the DNS channel Xu et al. (2013) applied DNS tunnelling, a technique used for transmitting data via the DNS protocol. In order to communicate, the DNS C&C channel uses two different forms to communicate: codeword mode and tunnelled mode (Xu et al. 2013). The codeword method allows for one-way communication by specifying the command as a hostname while the tunnelled method allows for bi-directional communication between the bot and botmaster, used for collecting stolen information (Xu et al. 2013). Feederbot, identified by Dietrich et al. (2011), is a botnet known to have used DNS as the carrier for C&C messages. In order to transport the messages, Feederbot uses valid DNS syntax with text record (TXT) resource messages, which is human-readable text in the DNS record (Dietrich et al. 2011). DNS as a C&C channel provides the following advantages (Dietrich et al. 2011):

- Due to is limit use as a C&C channel, no specific detection mechanism has yet been designed.
- In environments with strict Internet access, DNS is usually one of the few protocol that is allowed.
- As DNS was designed as a distributed system, the protocol provides resilience.

DNS C&C is therefore more stealthy than application-based (such as HTTP and IRC) C&C channels (Xu et al. 2013).

2.3.2.4 P2P C&C

The final C&C channel explored in this section is based on P2P technology. The first botnet to utilise P2P technology was Sinit (also known as Calypso) and was discovered in 2003 (Schiller 2007). P2P C&C botnets use P2P technologies and protocols to construct the C&C infrastructure with the popular P2P protocols to be exploited being Gnutella and Kademlia (Mukamurenzi 2008). Gnutella is an open, decentralised group membership and search protocol (Ripeanu 2001), while Kademlia constructs a P2P distributed hash table (DHT) (Maymounkov and Mazieres 2002). P2P protocols became more common with the popularity of Napster since it allows for the concept of nodes in a network acting as both clients and servers (Mukamurenzi 2008). Botnets using P2P for C&C focus on resiliency through the use of the P2P network and permit the botclients to communicate with peer bots rather than a central C&C server (Grizzard et al. 2007). Botmasters constructing P2P C&C botnets also have the option to select between closed and open P2P networks. A closed P2P network is dedicated to support bot communication and often uses homemade P2P protocols (Schoof and Koning 2007).

An open P2P network makes use of an existing P2P network without any adaptations (Schoof and Koning 2007). P2P networks are currently a popular choice for C&C as they are decentralised and provide no single point of failure (Schiller 2007). As with all other C&C possibilities, P2P C&C is not free of faults. Common disadvantages of P2P C&C include peer discovery, network responsiveness and command latency (Schoof and Koning 2007, Grizzard et al. 2007).

The C&C channels are responsible for command dissemination and will only be capable of doing so if the C&C channels are constructed according to a well-designed topology. The topology represents the layout of the entire botnet and the successful construction of the topology will ease the process of command dissemination.

2.3.3 Topology

The third component of a botnet, the topology, refers to the internal structure and the organisation of the individual botclients within a botnet (Zeng 2012). Conventional topologies include centralised, decentralised, unstructured and hybrid typologies. Centralised topologies are characterised by a central point that is responsible for the forwarding of the commands to the botclients (Bailey et al. 2009). Such topologies rely on existing protocols on top of the IP to propagate the commands (Leder et al. 2009). A static C&C server is also a characteristic often used by centralised topologies (Leder et al. 2009). The biggest problem faced by centralised topologies is single point of failure of the C&C server (Bailey et al. 2009).

Decentralised topologies use P2P networks to construct the structure of the botnet (Leder et al. 2009). The botmaster uses either established P2P protocols or homemade protocols to route the commands through the topology (Schoof and Koning 2007). Decentralised topologies have no single point of failure but have no guarantee of message delivery or latency (Bailey et al. 2009).

Unstructured topologies are an expansion of decentralised topologies by allowing a botclient to know of only one other botclient (Bailey et al. 2009). Such topologies cause latency to be extremely high with no guarantee of message delivery (Bailey et al. 2009).

Hybrid topologies combine the stealth from centralised topologies and the robustness of distributed topologies (*The Role of DNS in Botnet Command & Control* 2012). One group of the bots acts as servants, using static, non-private IP addresses that can be accessed from the Internet (*The Role of DNS in Botnet Command & Control* 2012). The second group of bots acts as clients, do not accept any incoming connections and have dynamic IP addresses (*The Role of DNS in Botnet Command & Control* 2012). For command routing, hybrid topologies use P2P protocols similar to those used in decentralised topologies (Kim n.d.). Hybrid topologies thus provide the best of both centralised and decentralised topologies.

The topology of a botnet can also have an influence on the structure of the C&C channel. Thus the complete design of the botnet is up to the botmaster and will most likely depend on the goal that the botmaster wants to achieve with the developed botnet.

2.4 Why Do Cyber Criminals Need Botnets?

Cyber criminals design and build botnets to perform automated tasks on behalf of humans since it is easier to control thousands of botclients than to motivate humans to perform the tasks. Computers are far more reliable, less careless and carry fewer risks and therefore are the optimal choice for the construction of a botnet (Elliott 2010). The remainder of this section deals with the motives behind the use of botnets, providing a closer look at the following popular incentives: financial gain, espionage and political protest.

2.4.1 Financial Gain

A popular attraction for botnet development is financial gain (Cole et al. 2007). Botmasters often develop botnets for the purpose of renting or selling the botnet to cyber criminals, allowing the botmaster the opportunity to construct a barrier between themselves and law enforcement (Elliott 2010).

In order to gain financially, botmasters can use a fake website, negotiate a pay-per-click ad deal with legitimate Internet companies and use the botnet to simulate the clicks (Cole et al. 2007).

DDoS attacks are popular attacks performed by botnets since the attacks come from multiple sources that are usually widely distributed. DDoS attack refers to the intentional or unintentional assault on the availability of a system and can cause degradation or complete unavailability of services (Elliott 2010). Due to the large number of computers involved, powerful processing capabilities and wide distribution, botnets have become powerful tools for DDoS attacks. Spam is the abuse of electronic communication media to send out large quantities of unsolicited messages while hiding the sender's identity (Geer 2005). Botnets are appropriate tools for sending out spam messages because the cost of sending out these messages, in terms of software, hardware and bandwidth, is covered by the victims running the botnet (Elliott 2010). Developing a botnet on behalf of a cyber criminal to perform either DDoS attacks or spam will give the botmaster a handy profit.

Botmasters can create botnets to carry out phishing scams. Phishing is a form of social engineering which attempts to acquire sensitive information (Elliott 2010). The botnets can send out emails requesting a user to follow a specific link that will lead them to a fake website, which is hosted by the botmaster (Elliott 2010). If the user completes any input form on the website, the botmaster will acquire the information and can use it either for personal gain (such as identity fraud) or sell the information to other cyber criminals (Elliott 2010).

With so many options available for botmasters to achieve financial gain by building botnets, botmasters will continue to improve and advance botnet development.

2.4.2 Cyber Espionage

Botnets can be used to carry out system intrusions and so enable interested parties to compromise the confidentiality of information. This is possible by allowing the botnet to carry out espionage on government data, national infrastructure, corporate intellectual property, employee data and personal data (Elliott 2010). Cyber espionage is a real and current threat, proven by the recent discovery of the Flame malware. Flame is a modular computer malware, discovered in 2012, and attacks computers running Microsoft's Windows operating system (sKyWIper Analysis Team 2012). This botnet is specifically used for targeted cyber espionage in Middle Eastern countries (sKyWIper Analysis Team 2012). Botnets, such as Flame, show the increased reality of cyber espionage.

2.4.3 Political Protest

Botnets can also be used as a tool for political protest and this often happens in Eastern Europe (Elliott 2010). With a political protest, cyber criminals target government and political websites, causing major disruptions in order to make a political point (Elliott 2010). During 2007, Estonia suffered a three-week wave of politically motivated cyber attacks (Traynor et al. 2009, Lesk 2007). These attacks were prompted by the Estonians' relocation of the Soviet Second World War memorial on April 2007 (Elliott 2010). During the attack websites of government ministries, banks and newspapers were repeatedly forced offline (Elliott 2010). It is believed that botnets were created beforehand to accomplish the DDoS attacks.

Botnets are a real threat and can accomplish multiple attacks. The cyber attacks in Estonia and the Flame malware just show the complexity by which botnets operate. These attacks caused by botnets can have severe consequences for multiple parties and these consequences are further explored in the next section.

2.5 The Consequences of Botnet Attacks

The result of botnet attacks can have severe consequences for government institutions, private corporations and the public. One such consequence is damage to reputation, where massive cyber attacks have the potential to cause a government international embarrassment and weaken the public's impression of private companies (Elliott 2010). Botnets have the potential to cause disruptions of varying degrees. Such disruptions can be website outages, malware infections and severe system intrusions (Elliott 2010). DDoS attacks are mostly responsible for such disruptions, causing a loss of network resources, delaying work and disabling communication between legal network users (Garg and Chawla 2011). Apart from causing disruptions, botnet attacks also have the ability to expose sensitive information and so enable other parties to gather intelligence covertly (Elliott 2010, Ramzan and Wüest 2006). Individuals targeted by spam or phishing attacks can potentially suffer financial loss to cyber criminals with no chance of reclaiming the money (Elliott 2010). Individuals who fall victim to cyber attacks often also suffer an emotional loss. That is because some of the personal data, such as documents and photos, are difficult to replace after a cyber attack that rendered the computer into an unusable state (Elliott 2010).

Either protecting against or recovering from botnet attacks can be expensive for government institutions, private corporations and individuals. Government institutions and private corporations must continuously invest in anti-virus, intrusion detection and firewall software in order to prevent malware infections (Elliott 2010). Individuals will mostly rely on anti-virus and security applications to protect against botnet attacks.

Due to the severe consequences of botnet attacks, it is necessary for security experts to continuously improve and develop new botnet detection mechanisms.

2.6 Botnet Detection Mechanisms

As botnets continue to evolve, security analysts constantly propose and design new botnet detection mechanisms. There are mainly two general approaches for botnet detection, which include the use of honeypots or honeynets and traffic monitoring by means of an intrusion detection system (IDS) (Zeidanloo and Manaf 2010, Zhu et al. 2008).

2.6.1 Honeypots

A honeypot is essentially a computer system that is designed as a trap, capable of drawing the attention of the attackers (Zeidanloo and Manaf 2010). Honeynets are basically a collection of honeypots, acting together as a functioning network (Gupta 2002). A honeypot or honeynet is able to collect the following information relating to botnet activities (Zeidanloo and Manaf 2010):

- Signature of the bots for content-based detection.
- Information about the C&C channels and servers.
- Unknown security holes of the penetrated computer system.
- Tools and techniques used by the botmaster.

- Motivation of the botmaster.

The techniques incorporated by honeypots or honeynets are very efficient for monitoring and tracking botnet activities. They also allows for the collection of bot binaries and support the analysis and examination of botnet behaviour (Zeidanloo and Manaf 2010).

Honeypots are limited in terms of functionality and can only detect botnet activity once a bot infects the computer system. Only after the initial infection can the analysis and examination of the botnet begin (Zeidanloo and Manaf 2010). To monitor the activities over a longer period of time, IDSs are deployed to monitor the traffic and allow for the possibility of detecting the botnet before infection can occur.

2.6.2 Intrusion Detection Systems

Most IDSs used for botnet detection fall into one of the following categories: signature-based, anomaly-based, DNS-based or mining-based botnet detection techniques (Zeidanloo and Manaf 2010, Feily et al. 2009). Signature-based botnet detection techniques make use of common signatures of current botnets. By using well-known signatures, this technique can immediately detect botnets and there is no opportunity for false positives. However, this technique can only detect current or active botnets. Therefore unknown botnets, such as zero-day botnet attacks that exploits one or more security vulnerabilities on the very same day that these vulnerabilities becomes known, will pass by unnoticed (Zeidanloo and Manaf 2010).

Anomaly-based detection techniques overcome the shortcomings of the signature-based detection techniques by trying to detect botnet activities based on network traffic anomalies, such as high network latency, high volumes of traffic, traffic on unusual ports and unusual system behaviour. Closely examining the network traffic anomalies allows this technique the opportunity to detect unknown botnets. The anomaly-based detection technique can further be categorised into host-based and network-based detection techniques. Host-based detection techniques closely monitor and analyse the internal activities of a computer system and do not pay attention to network traffic. Network-based detection techniques are completely the opposite, ignoring the internal activities of the host computer system and instead focusing on the network traffic. The monitoring of the network traffic can either be active or passive. With active monitoring, test packets are continuously injected into the network and the reactions of the network are then examined. With passive monitoring, the traffic is just constantly inspected as it passes by while causing no interference with the traffic (Zeidanloo and Manaf 2010).

DNS-based detection techniques closely relate to anomaly-based techniques since similar algorithms are applied. This particular detection technique focuses on the DNS information generated by a botnet and will search for anomalies in DNS traffic (Feily et al. 2009).

Anomaly-based detection techniques are not always capable of detecting C&C traffic. This is mostly due to low traffic volume and network latency caused by C&C traffic. To overcome this limitation, data mining techniques such as machine learning, classification and clustering can also be used efficiently to detect C&C traffic (Feily et al. 2009).

The detection techniques described in this section have become valuable weapons for the prevention and detection of botnet attacks. However, they are struggling to keep up with the continuous improvement of botnet technology.

2.7 Case Study: Zeus Botnet

Zeus botnet (also known as Zbot) is one of the most sophisticated and wide-spread Trojans that currently exists (Wyke 2011). With a focus primarily on information stealing, especially online credentials, Zeus has caused the loss of millions worldwide (Wyke 2011). The Zeus Trojan is bundled as symplistic toolkit, which includes the necessary components to install and launch the bot. The toolkit is a complete package that is readily available and therefore popular infection vectors are drive-by-downloads and spam (Falliere and Chien 2009).

Upon installation, the Zeus bot executes the following actions (Wyke 2011):

- Copy the executable to another location and execute the copy.
- Lower browser security settings by changing IE registry entries.
- Injects code into other processes.

As mentioned above, the main purpose of Zeus is to steal online credentials for financial gains and this is achieved by performing four main actions (Falliere and Chien 2009):

- Gathering system information.
- Stealing online credentials
- Perform web page injections
- Communicating with the C&C server to receive additional tasks to perform.

Zeus automatically gathers a collection of system information and send the information to the C&C server (Falliere and Chien 2009). This information includes (Falliere and Chien 2009):

- A unique bot identification string
- Name of the botnet
- Version of the bot
- Operating system version
- Operating system language
- Local time of the compromised computer
- Uptime of the bot
- Last report time
- Country of the compromised computer
- IP address of the compromised computer
- Process names

To obtain online credentials, Zeus follows two distinct steps: performing automatic actions hardcoded in the binary and executing actions specified in a configuration file. As new actions are required to be executed by the Zeus bot, a newer version of the configuration file is downloaded from the C&C server. The actions found encoded in the binary or configuration file are executed and automatically steal information stored in protected storage. The information found in protected storage include Internet Explorer passwords and e-mail passwords, which were sent in clear text across the network (Falliere and Chien 2009). The stolen online credentials are then send to the C&C server via encrypted HTTP POST requests (Alazab et al. 2011).

The Zeus toolkit contains a collection of web injection samples, which are defined in the configuration file. The web injection samples are responsible for injecting additional hyper text markup language (HTML) code into online banking pages. The code bypass security implementations and cause users to input credential information that is not normally required by banks (Falliere and Chien 2009).

Besides stealing online credentials, Zeus has a variety of built-in commands that are used to perform additional tasks. The C&C server is responsible to determine if the bot should execute any of the additional tasks. Some of the available tasks include (Falliere and Chien 2009):

- Reboot: reboot the computer
- Rexec: download and execute a file
- Lexec: execute local file
- Resetgrab: steal information from the protected storage
- Upcfg: update configuration file

For protection, Zeus applies obfuscation methods such as polymorphic encryption that allows for immediate re-encryption upon infection, allowing the bot to avoid signature-based detection (Alazab et al. 2011). Newer version of Zeus saw an increase in obfuscation with additional encryption being applied to the original RC4 encryption key (Wyke 2011).

The ease of use, simplistic design and effectiveness have allowed Zeus to become one of the most popular malware toolkits to date. The current success of Zeus provides an opportunity for the malware to evolve further and become even more widespread, especially if the demand for information stealing Trojans continuous to rise.

2.8 Conclusion

This chapter provided a short introduction to botnets by focusing on their characteristics and components. Each component, namely propagation, command and control and topology, was discussed in detail, providing a general overview of the internal workings of botnets. The motivations behind the development of botnets were also determined, with financial gain, cyber espionage and political protest forming the popular motives. Botnet attacks can have severe consequences for government institutions, private corporations and individuals, and therefore various botnet detection mechanisms have been developed. The most commonly used detection mechanisms to prevent attacks by botnets are honeypots and IDSs. Although botnets are still common threats for technology users of today, there has recently been a shift in focus for malware developers. This is due to the increasing popularity of mobile devices. With the significant improvement in smartphone designs and the release of tablet computers, users are no longer only storing valuable information on desktop computers. Mobile devices and popular mobile operating systems as used by these devices are considered in the next chapter, and a closer look is taken at the evolution mobile malware.

Chapter 3

Mobile Devices and Mobile Malware

Over the past decade mobile devices have become a daily companion for people. The popularity of these devices can be attributed to the wide range of functionalities supported by these devices. Some of the advanced functionalities include cost-free communication mechanisms (such as Wireless Fidelity (Wi-Fi), Bluetooth and near field communication (NFC)), improved multimedia features (such as video recording, music playback and high quality display) and the possibility of downloading additional applications with added functionalities. These features have only recently been supported by mobile devices and that is due to the improved capabilities of mobile operating systems. Such improvements have also given malware developers the opportunity to target these devices. This research focuses only on smartphones and tablet computers.

This chapter introduces mobile devices, with their evolution being explained in Section 3.1. Section 3.2 focuses on mobile operating systems (OSs), exploring their differences and similarities. The steady growth of mobile malware is described in Section 3.3, while Section 3.4 concludes the chapter.

3.1 Evolution of Mobile Devices

Mobile devices, such as smartphones and tablet computers, have become an integral part of everyday living during the last few years. Such devices are constant companions and provide a wide range of functionalities for the end-users. The following sections deal with the path these devices took to achieve the functionality available today.

3.1.1 Smartphones and their Place in History

Commercial mobile technology started in the early 1940s and was based on radio technology and instruments (Vochin 2009, Farley 2005). Early forms of mobile phones closely resembled two-way radios, but were enormous and did not truly conform to the aspect of portability (Vochin 2009). During 1973 Motorola took a big step forward in mobile phone development when Dr Martin Cooper placed the first phone call on the DynaTAC prototype (Honan 2009). The DynaTAC was not commercially available until 1983 and at that time it was known as the first handheld cellular phone, despite it weighing almost a kilogram (Farley 2005, Catanzariti 2009). For the remainder of the 1980s, mobile phones were mostly known for their in-car use (Catanzariti 2009) but during the 1990s this situation changed quickly. This change was supported by the significant shrinking of the weight and size of mobile phones (Farley 2005).

Motorola's StarTAC became the first mobile phone that was actually small enough to be placed into a pocket (Honan 2009) and this proved to be a significant step forward for mobile phone development. During the early 1990s the Bell South/IBM Simon Personal Communicator attempted to become the first commercially available smartphone (Subramony 2011). The device supported a touch screen to simplify the user interface and included multiple personal digital assistant (PDA) functionalities such as an appointment calendar, address book, email and file management (Lewis 1996). Around the same time Nokia's 1011 became the first commercially available mobile phone to support the global system for mobile communications (GSM) and therefore was also the first phone to support text messaging (Koblentz 2011). By 1996, GSM mobile phones spread beyond the borders of Europe and could be found in Australia, Russia, South Africa and even the United States (US) (Agar 2004). Devices such as the above provided a firm foundation for the future development and growth of mobile phone technology.

In 1996 Nokia released the 9000 series of mobile phones which became one of the biggest stepping stones for smartphone development. By many considered as the first true smartphone, the Nokia 9000 Communicator looked like an ordinary phone but could flip open, revealing a second screen and a QWERTY keyboard (Lewis 1998). The Nokia

9000 Communicator combined fax, voice and email into a single portable device while also allowing access to the Internet (Lewis 1998).

The turn of the millennium saw great leaps in mobile phone functionality. During 1999 the Kyocera VP-210 mobile phone became the first mobile videophone, featuring a built-in camera and colour display (Okada 2005). The phone had relatively fast transmission speed and allowed for real-time communication (Okada 2005). In 2000 Ericsson Mobile Communications unveiled the first Bluetooth phone, the T36, which was able to stay active for long periods and transfer large quantities of data (Comerford 2000). The turn of the millennium proved to be the turning point for the evolution of mobile phone technology.

The popularity of combined PDA and phone functionality led to the release of the Blackberry 5810 and Palm Treo 600 devices (Chick 2004). The 5810 featured electronic messaging, personal organiser, calendar and physical keyboard (Simão et al. 2011). The Palm Treo 600 featured a full QWERTY keyboard, a colour display and ran the Palm OS 5 operating system (Belov et al. 2005). By 2004 Motorola provided the next leap in mobile phone evolution with the release of the Motorola RAZR V3 (Catanzariti 2009). The phone featured a stylish anodised aluminium cover and included the following advance functionalities: a video graphics array (VGA) camera, quad-band compatibility and Bluetooth support (Catanzariti 2009).

The last few years witnessed the revolution of smartphones, starting with the HTC Universal released in 2005. This was the first smartphone to be released by HTC and featured the Windows Mobile 5 operating system (Froehlich et al. 2007). 2007 saw the dawn of Apple's iPhone, unveiled at the MacWorld expo in San Francisco (Hall and Anderson 2009). This was the first smartphone that had mass appeal to the general community, providing a slick touch screen interface and a convenient mobile gaming platform (Hall and Anderson 2009). At the end of 2008, the HTC Dream smartphone was released. It featured a full slide-out QWERTY keyboard, a touch screen interface and a touch-based OS that focused on user personalisation (Catanzariti 2009). This was also the first smartphone to use the Android OS, designed to compete with Apple's iPhone.

The past twenty years thus saw a revolution in the development of mobile phones, transforming the devices from basic voice and text to the all-in-one portable devices with advanced capabilities and features.

3.1.2 The Rise of Tablet Computers

The idea of tablet computers is not novel and designs for such computers appeared as early as the late 1960s. One of the first attempts at a tablet computer came in the form of the Dynabook (Press 1992). The architectural design of this tablet was done by Alan Kay and the name suggested that this device would be both portable and dynamic (Press 1992). The first commercially successful portable computers, Osborne 1 and the Radio Shack Model 100, aimed to achieve different goals. The Osborne 1 closely resembled a desktop computer, using the same central processing unit (CPU), bus and operating system (Press 1992). The Radio Shack Model 100 acted as an input device used in conjunction with a desktop computer (Press 1992).

In 1985 Pencept Inc. introduced the Penpad tablet computer. This tablet featured a touch-based input system that could recognise both handwriting and limited hand gestures (Schedeen 2010). A few years later GRiD Systems released the GRiDPad; the first portable tablet computer. The GRiDPad also had the ability to recognise hand-printed characters and was used primarily for data collection (Moran 2006). The Apple Newton followed shortly thereafter but ultimately Apple veered away from building a tablet computer and instead developed a PDA (Schedeen 2010).

2001 saw the eventual rise of tablet computers. In that year Microsoft released the Windows XP tablet edition, which supported a variant of the Windows XP OS with additional functionality to allow pen-based input (Godwin-Jones 2003). In 2003 the Compaq TC1000, a silver tablet/notebook with a detachable 10.4-inch touch screen, was released (Sharples and Beale 2003). However, the computer lacked proper computing performance and therefore struggled to succeed. Performance issues were not the only feature that delayed the rise of tablet computers; the cost associated with these types of devices was also a factor.

April 2010 saw a revolution in tablet computers with the launch of Apple's iPad. The iPad featured a 9.7-inch, 1024 x 768 display, providing a choice between Wi-Fi only or Wi-Fi+third generation (3G) models while using a multitouch screen rather than a physical keyboard (Meurant 2010). Following on the heels of the iPad was the Samsung's Galaxy Tab and the Motorola Zoom tablet.

By 2012 tablet computers had broken into the market and now feature a wide variety of functionalities. Popular tablets released during 2012 include Apple's third generation iPad (iPad 3), Google's Nexus 7, Sony's Tablet S and the Samsung Galaxy Tab 8.9. Most of these tablets offer liquid-crystal display (LCD) or light-emitting diode (LED) displays, Wi-Fi, Bluetooth connectivity and impressive performance capabilities.

Throughout the remainder of this dissertation, the reference to mobile devices will include only smartphones and tablet computers, and will exclude any other portable or mobile devices such as PDAs, notebooks and netbooks. In this dissertation a smartphone is defined as a mobile phone capable of supporting computer functionalities and running general-purpose applications. A tablet computer is defined as a general-purpose computer integrated into a single panel and using a touch screen as the input device. The popular operating systems found on these mobile devices are introduced and their design structures are discussed in the next section.

3.2 Popular Mobile Operating Systems

OSs have become an integral component of any computing device. They can be described as an organisational unit within a computer system and are in fact the interface between applications and the underlying hardware (Speckmann 2008). The most important function of any OS is the administration of the available operating and hardware resources (Speckmann 2008). Five popular mobile OSs are described: Android OS, iPhone OS, BlackBerry OS, Windows Phone 7 OS and the Symbian OS. With each OS the focus will be on the security structures, isolation of applications and resources, the development of applications and their publication on markets.

3.2.1 Android Operating System

The Android OS is a Linux-based operating system, targeting mobile devices such as smartphones and tablet computers. Developed by Google in conjunction with the Open Handset Alliance, Android runs on top of a Linux kernel and all hardware support is provided by Linux (Barrera and Van Oorschot 2011).

The software stack of Android is divided into four layers (Speckmann 2008):

- The application layer, providing access to a set of basic applications.
- The application framework, used to implement a standard structure between the applications and the OS.
- The libraries, written in C/C++ and called through a Java interface.
- Lastly, the Linux kernel, which provides access to low-level resources.

To improve security, process and file system isolation is achieved by making each application run as its own user (Barrera and Van Oorschot 2011). Furthermore, Android only allows applications to interact and use system resources based on specific permissions. This permission-based structure requires developers to define any functionality their specific application might require (Barrera and Van Oorschot 2011). Such applications are written in Java and run in a custom virtual machine called Dalvik. Google offers a single point of sale with the Google Play Store but has minimal involvement with the publication of applications on the Store (Holzer and Ondrus 2011). Banned applications can still be distributed beyond the Play Store by using third-party application markets.

3.2.2 iPhone Operating System

The iPhone operating system (iOS) is a mobile OS, developed and distributed by Apple, for their line of mobile devices. Apple's mobile devices that support the iOS include the iPhone, iPod Touch and the iPad, allowing developers to easily migrate applications between all these devices (Barrera and Van Oorschot 2011).

The iOS is responsible for managing the device hardware and providing basic technologies needed for native applications. To achieve this, the iOS uses a simplified software structure that consists of two layers. The bottom layer is the Mach kernel and hardware drivers, which control the execution of the programs on the device (*iPhone OS Technology Overview* 2009). The top layer contains core technologies and interfaces used for application development (*iPhone OS Technology Overview* 2009).

Separation and isolation of the applications on the iOS occur by means of a sandboxing mechanism, in which a policy file provides restriction to certain device features and data (Barrera and Van Oorschot 2011). Applications for the iOS, written in Objective-C, can only communicate with hardware through a set of published application program interfaces (APIs) (Barrera and Van Oorschot 2011). A major distinction from the Android OS is that Apple developers must forward their developed applications to Apple for approval, complying with Apple's strict review process (Holzer and Ondrus 2011). Only accepted and signed applications will be published on Apple's application market (Barrera and Van Oorschot 2011).

3.2.3 BlackBerry Operating System

BlackBerry OS is a mobile OS developed by Research In Motion (RIM) for their BlackBerry range of smartphones (Lin and Ye 2009). The OS was specifically targeted towards enterprise customers and includes features such as push email and groupware support (Lin and Ye 2009). For security, the BlackBerry OS uses sandboxing for isolating applications at runtime and achieves this through the Java virtual machine (JVM) (Barrera and Van Oorschot 2011). Third-party applications, written in Java, are also supported by the BlackBerry OS (Barrera and Van Oorschot 2011). The process of application approval for the BlackBerry OS follows a similar method to that of Android. Applications not approved by RIM can still be distributed beyond the market (Barrera and Van Oorschot 2011).

3.2.4 Windows Phone 7 Operating System

Windows Phone 7 OS, developed by Microsoft, was designed to regain market share from the competitors (particularly the Android OS and iOS). This OS is based on the Windows CE 6 kernel and comes with a new interface, which includes a multi-touch screen and an on-screen virtual keyboard. Instead of the well-known icons for applications, this OS uses a different feature referred to as tiles. In terms of security, the Windows Phone 7 security model employs the principle of least privilege. Furthermore, the execution environment for applications is restricted (run within a sandbox) and cannot directly access operating system internals. Windows Phone 7 OS allows the installation of third-party applications and includes a set of default applications: a web browser, email clients, multimedia players and the Office suite. The market is managed by Microsoft and a strict approval process similar to that of Apple is followed. With the approval process, every submitted application is checked and signed before it can be downloaded by users (Schaefer et al. 2011). Windows Phone 7 OS is now slowly being replaced by Windows Phone 8, OS which was released at the end of 2012.

3.2.5 Symbian Operating System

Symbian OS is a multitasking operating system, first developed in the late 1990s (Babin 2008). Symbian delivers an advanced, open standard OS and supports the complex requirements for network protocols worldwide and also allows for a broad, international developer's community (Ancarani and Shankar 2003).

The Symbian OS architecture consists of five layers (Ancarani and Shankar 2003):

- The bottom layer consists of a core that contains the kernel, file server, memory management and device drivers.
- The system layer provides for communication, computing services and database management.
- Application engines provide software developers with the necessary tools to create user interfaces.
- The user interface software layer is responsible for visualising the look and feel of the OS.
- At the top is the applications which provide the end-user with functionalities.

The Symbian OS is also open to third party application development, with the applications required to be developed in C++ (Babin 2008). Such native applications have direct access to storage memory, the phone interface and messaging (Babin 2008). In order to add some level of security to their applications, developers can use digital certificates or signatures (Babin 2008). Applications for the Symbian OS can be found on the Nokia Ovi Store.

3.2.6 Conclusion of Mobile Operating Systems

Each OS mentioned in this section shares one common property: they are specifically designed for mobile devices. Although similar in purpose and provided functionalities, there is one characteristic that separates the mobile operating systems: the application approval process. OSs such as Android, BlackBerry and Symbian follow a less strict process and provide minimal involvement with the publication of applications. Windows Phone 7 and iOS follow a more rigorous process and have strict policies in place for accepting an application.

This application approval process of the mobile OSs is one of the aspects malware developers will look at before choosing their targeted mobile OS. This is because the application approval process can impact the ability of malware to propagate successfully.

The next section deals with the evolution of mobile malware and will reveal the mobile OSs that are continuously targeted by malware.

3.3 Mobile Malware

Malware is often defined as any type of hostile, intrusive or annoying software that is especially designed to manipulate a device without the user's consent (La Polla et al. 2012). Mobile malware is thus malicious software targeted specifically at mobile devices. Several past papers have been published on the evolution of mobile malware (Hypponen 2006, Lawton 2008), describing the evolution of state-of-the-art mobile malware from 2004 to 2006. The focus in this section, however, is on malware that has contributed significantly to the evolution of mobile malware and provided the required functionality that eventually led to the rise of mobile botnets.

The first malicious worm, Cabir, for mobile phones was discovered in 2004 (Chen and Peikari 2008). This worm runs on mobile phones that support the Nokia-licensed Symbian Series 60 platform (Leavitt 2005). Cabir replicates via Bluetooth and arrives on a victim's phone as a software installation script (SIS) file (Leavitt 2005). In addition, the worm also has the capability to interfere with the host device's Bluetooth system, forcing it to continuously scan for other enabled devices (Leavitt 2005). Cabir is considered to be the first mobile malware capable of spreading itself by exploiting networking technologies

(La Polla et al. 2012).

During the same year a new Trojan horse started targeting Nokia phones. The Trojan, named Skulls, appeared to be an application that allowed users to process, select and remove design themes for their phone screens (Leavitt 2005). In the background, however, Skulls disabled Symbian applications and only phones with third-party file managers could remove this Trojan (Leavitt 2005). Skulls was one of the first mobile malware samples capable of causing a denial of service (DoS) (Leavitt 2005).

A new worm, called CommWarrior, surfaced in 2005 (Sarwar et al. 2007). This worm operates on the Symbian Series 60 platform and in addition to the capability of replicating via Bluetooth, this malware was the first to use multimedia messaging service (MMS) messages as a replicating mechanism (Xia et al. 2008). CommWarrior replicates over Bluetooth by means of a randomly named SIS file, while MMS messages are used to forward this infected SIS file to other users (Xia et al. 2008).

Shortly after the arrival of CommWarrior followed the first SMS worm for the Symbian OS. This malware, called Feak, was capable of sending SMS messages to all of the contacts listed on a particular phone (La Polla et al. 2012).

RedBrowser, discovered in 2006, was the first SMS Trojan to target mobile phones capable of supporting Java 2 platform, micro edition (J2ME) applications (La Polla et al. 2012). By running on top of the J2ME platform, this malware could be independent of any particular operating system (La Polla et al. 2012). In the background the malware sends SMS messages to premium-rate services located in Russia. Today, RedBrowser is known as the first for-profit malware attack (Soitinaho 2007). 2006 also revealed the first spyware for the mobile platform. The spyware, called Mobispy, targeted Symbian Series 60 devices and collected the users' private information such as SMS messages and call logs (Soitinaho 2007).

As the sophistication of mobile malware continued to grow, the PMCryptic malware proved to be the next step in the evolution of mobile malware. PMCryptic is considered to be the first polymorphic worm and infects phones capable of running the Windows CE platform (Hinson 2010). Each time the worm spreads, a new polymorphic copy of itself is generated by appending 255 bytes of random data, protecting the worm against discovery. PMCryptic is also responsible for several unwanted actions such as dialling premium numbers (Hinson 2010).

The first worm to target iPhones, ikee, was discovered during 2009. The worm had the ability to alter the wallpapers of the iPhone but did not perform any malicious activities (La Polla et al. 2012).

With the arrival of the Android OS in 2008, malware developers started shifting their attention to this new OS. Two years later the first SMS Trojan for the Android OS was discovered. The malware, named FakePlayer, appeared as a legitimate movie player application but in the background sent out SMS messages to premium-rate numbers without the user's consent (Zhou and Jiang 2012). Another Android malware, Geinimi, appeared at the end of 2010. This was the first Android malware to display traditional botnet functionalities such as receiving commands from a remote server (Zhou and Jiang 2012). Geinimi also collected personal information and forwarded the information to the remote server, encrypting the communication to the remote C&C server via the data encryption standard (DES) encryption scheme (Zhou and Jiang 2012).

2010 proved to be an important period in the evolution of mobile malware. Zeus-in-the-Mobile (ZitMo) was the first malware to infect both Blackberry and Android devices (La Polla et al. 2012). In order to improve online banking security, banks use mobile devices for out-of-band authentication. Thus for certain transactions the banks send an SMS message to a mobile number associated with the user's account, containing a transaction authentication number (TAN) (Davi et al. 2012). To complete the transaction, the user enters this mobile TAN (mTAN) on to the bank's website (Davi et al. 2012). The purpose of ZitMo is to steal these mTAN codes without alerting the users. It works closely with the regular Zeus Trojan, which is responsible for stealing personal user data relating to online banking activities (La Polla et al. 2012). Both malware enable a cyber criminal to connect to a victim's bank account using the stolen credentials and to transfer funds (Mansfield-Devine 2012*b*). During the last few years new versions of ZitMo have appeared, namely SpyEye-in-the-Mobile (SpitMo) and Carberp-in-the-Mobile (CitMo) (Maslennikov 2013).

SMSZombie, discovered in July 2012, targets Android devices and about 500 000 mobile devices in China have been infected thus far (Mansfield-Devine 2012*a*). This complex and sophisticated malware takes advantage of a vulnerability in the China Mobile SMS

Payment process (Fisher 2012). The malware attempts to gain administrator-level privileges and is designed to steal money by sending SMS messages to the attackers (Fisher 2012). Once installed, SMSZombie disables the user's ability to remove the application and also hides the Cancel button during installation (Clapsadl 2012).

During the last decade the goal of mobile malware has not changed, with the malware either focusing on stealing money, taking control of the device or causing a DoS attack. The targeted OS platform has, however, changed frequently. For a large part of the mobile malware evolution the Symbian OS was the targeted platform, with only a few malware samples targeting the Windows Mobile OS. Recently, the Android OS has become the focal point for malware developers and this is verified by the sharp increase in Android malware during 2012. During 2012 65% mobile threats targeted the Android OS but by the end of 2012 this was nearly 94% (Maslennikov 2013). Most of the newly released mobile malware in 2013 is expected to continue targeting the Android OS.

3.4 Conclusion

This chapter focused on mobile devices, specifically smartphones and tablet computers. The evolution of these devices was examined closely, revealing the individual devices that contributed significantly to the improvement of mobile device technology. Furthermore, popular mobile OSs often associated with these devices (such as Android OS, iPhone iOS, BlackBerry OS, Windows Phone 7 OS and the Symbian OS) were also compared and evaluated. At the time of writing, the dominant mobile operating systems were Android and iOS. From the evaluation of the evolution of mobile malware in this chapter, it becomes possible to see that during the last few years attacks on mobile devices have intensified. Recently invented mobile malware has significantly improved its ability to remain hidden and it can now perform more severe attacks. Although this rise of malware does not have an obvious impact on the popularity of mobile devices, it is creating possibilities for the development of new threats. One such threat is mobile botnets.

Chapter 4

Overview of Mobile Botnets

Until recently, mobile devices remained unaffected by the security threats posed by botnets due to limited interconnectivity capabilities such as Internet access and device-to-device communication. This situation has changed significantly with the continuous growth in capabilities offered by new mobile devices.

Today, mobile devices have multiple possibilities to allow for communication, including 3G, Wi-Fi, Bluetooth, NFC and Internet connectivity. Due to this increasing popularity and improvement in technology offered by mobile devices, the threat of botnets is expected to move towards mobile networks.

The ultimate goal of mobile botnets will be similar to the attacks of traditional botnets, however, the targets will change to mobile devices such as smartphones and tablet computers (Enck et al. 2009).

This chapter covers mobile botnets with Section 4.1 focusing on their history of mobile botnets. Section 4.2 provides the definition and characteristics of mobile botnets, while Section 4.3 deals with the differences and similarities to traditional botnets. The potential capabilities and threats of mobile botnets as well as their proposed countermeasures are evaluated in Sections 4.4 and Section 4.5 respectively. Research relating to recent studies of mobile botnet designs is mentioned in Section 4.6. Section 4.7 concludes the chapter.

4.1 Short History of Mobile Botnets

Mobile botnets are starting to make an impact on mobile devices and a group of malware generations has contributed to the development of mobile botnet technology. These malware generations include the Symbian worm Yxes, the iPhone ikee.B botnet and the Android botnet Geinimi.

Early in 2009 security analysts detected a new Symbian malware targeting Symbian phones running the OS 9 operating system. The malware was responsible for sending SMS messages without the user's consent, retrieving the international mobile station equipment identity (IMEI) and international mobile subscriber identity (IMSI) numbers of the phone and communicating with remote servers. To connect and communicate with a remote server the malware relied on the following routines: retrieving the Internet settings of the phone, establishing a stealthy communication channel, creating HTTP requests and handling the responses received. The ability of the malware to connect to the Internet had many believing that it was part of a mobile botnet. The malware, however, lacked C&C channels and although it had the ability to contact remote servers, processing of commands was still limited (Apvrille 2012).

At the end of 2009 a new malware started targeting Apple iPhones. The malware, named ikee.B, includes C&C logic that allows the botmaster to have complete control over the infected iPhone. The malware infection begins with a remote infected iPhone detecting the presence of other uninfected iPhones running the secure shell (SSH) network service with the default password, 'alphine'. When the infected iPhone detects a vulnerable iPhone, the attacker performs a remote login to the vulnerable iPhone and uploads the ikee.B file in the following directory: /private/var/mobile/home/. A shell script then creates a dedicated directory on the infected iPhone to host the ikee.B malware. To propagate, ikee.B scans specific IP addresses for SSH services and then attempts to connect to the responding service as root by using the default password, 'alphine'. The malware is responsible for archiving all the SMS messages on the infected iPhone and then forwarding the information, along with other data collected from the phone, to a server located in Lithuania. All the ikee.B mobile clients maintain an ongoing communication channel with the server. However shortly after the outbreak of the ikee.B botnet, the server was discovered and taken down. Despite the limited growth potential of the ikee.B, it shows that iPhones can be attacked by mobile malware (Porras et al. 2010).

At the end of 2012 security analysts discovered a new Trojan horse, Geinimi, infecting Android mobile devices. Geinimi is the first Android malware to display functionalities closely relating to those of traditional botnets. Besides the basic botnet functionalities, Geinimi raised the sophistication of mobile botnet technology significantly. The malware deployed an off-the-shelf byte code obfuscator to hide the botnet activities and encrypted chunks of the C&C traffic (Strazzere and Wyatt 2011).

The malware generations mentioned above provided the foundation for the future development of mobile botnets. New mobile botnets are regularly discovered on various mobile platforms and with every new discovery there is an increase in the level of sophistication provided by the mobile botnet technology. There is thus a constant battle between the botmasters and the security analysts, and if no attention is given to the rising developments of mobile botnets, the botmasters may win the battle. The next section focuses on the definition and characteristics associated with mobile botnets that security analysts can possibly use to defend against mobile botnet attacks.

4.2 Definition and Characteristics

The upcoming generation of network technology will be mobile broadband, allowing mobile devices to stay online and connected to the Internet at all times (Flø and Jøsang 2009). With a definite connection between the Internet and the mobile network, it becomes possible for malware to move freely between these networks (Flø and Jøsang 2009). By allowing a botclient to run on a mobile device, a botmaster is able to exploit services in the mobile network (Flø and Jøsang 2009), which opens the way for a new bundle of threats for the users of mobile devices. One such threat is mobile botnets.

A mobile botnet is a collection of compromised mobile devices, controlled by a botmaster through a C&C network for a malicious purpose (Geng et al. 2012). Botmasters deploying mobile botnets are faced with a set of new challenges (Mulliner and Seifert 2010). Firstly, mobile devices are limited due to the use of batteries (Geng et al. 2012) and will thus provide the mobile botnet with limited power supply. Extensive execution requirements of a mobile botnet can cause the battery to drain quickly, alerting the user and leading to the discovery of the mobile botnet. Secondly, mobile telecommunications, such as the sending of SMS messages or phone calls, result in costs for the people participating in the communication and a significant rise in the bill can lead to the exposure of the mobile bot (Geng et al. 2012, Mulliner and Seifert 2010). Thirdly, constant changes in connectivity are a common characteristic for mobile devices and can cause a mobile botnet to become unstable (Mulliner and Seifert 2010). This is due to the fact that the connectivity of mobile devices can be influenced by the physical environment surrounding the device or personal factors relating to the usage of the mobile device (Geng et al. 2012). Finally, due to the lack of public IP addresses, mobile devices use network address translation (NAT) gateways for connection and are thus not directly reachable (Geng et al. 2012, Mulliner and Seifert 2010). This situation is, however, changing with the introduction of IPv6 addresses on mobile devices (Andrici 2012).

These challenges need to be taken into consideration by the botmaster when developing a mobile botnet as they will influence the internal design of the mobile botnet. To overcome these challenges, the botmaster can look towards the similarities found between mobile and traditional botnets.

4.3 Differences and Similarities to Traditional Botnets

The internal workings and design of a mobile botnet do not differ from traditional botnets, which were discussed in Chapter 2. The design of a mobile botnet also requires the three main components: propagation vectors, C&C channels and topology (Zeng 2012). The purpose of a mobile botnet is exactly the same as that of a traditional botnet: to perform a certain type of attack according to the wishes of the botmaster. The most obvious difference is the components used to drive the bot program. With mobile botnets, mobile devices such as smartphones and tablet computers are used to host the bot program, whereas with traditional botnets, personal computers are usually the preferred choice. The impact of the attack is another difference between traditional and mobile botnets. With the technology-driven era we live in today, vast quantities of information, including sensitive information, are stored on personal computers. Attacks by traditional botnets can thus have severe consequences for businesses, governments and individuals. With mobile botnets, the attack space is limited in terms of storage and processing power, causing the impact of their attack to be less severe and mostly affecting individuals. Due to the increasing acceptance of bring your own device (BYOD) policies, mobile botnets are sure to start infiltrating businesses and government sectors in the near future.

Although most of the features offered by traditional botnets are applicable to mobile botnets, there are additional features offered by mobile devices that are not always available for traditional botnets to use. In terms of C&C, mobile botnets have the possibility of utilising SMS, MMS or NFC as C&C channels to disseminate commands.

The topology of botnets is influenced by the selection of the components used to host the bot program. With traditional botnets, personal computers usually host the bot program, meaning that the botnet will have a static structure with relatively few changes appearing in the structure. With mobile botnets, smartphones usually host the bot programs, causing the topology to be more dynamic as the smartphones travel around with the users. Thus the chosen component to host the bot program will determine whether the topology will be static or dynamic.

Although there are small differences between traditional and mobile botnets, the additional features offered by mobile devices provide a new dimension to the development of botnet technology. This new dimension offered by mobile devices lead to new capabilities and threats, which are identified in the next section.

4.4 Potential Capabilities and Threats

Although mobile botnets have only recently become an important topic of interest among researchers, they have already made an impact on society. Mobile botnets, such as these examined in Section 4.1, reveal improved designs and will continue to advance as botmasters explore new mobile technologies. This section focuses on the potential capabilities of mobile botnets and the threats associated with them.

Mobile botnets have the potential to operate as surveillance software or spyware

(Wilson and Kifayat 2012). The motivation behind such an attack is the ability to obtain large quantities of data from the mobile device (*Lookout Mobile Threat Report* 2011). Such data is collected without the user's knowledge or approval and can include the phone call history, text messages, browser history and photos (*Lookout Mobile Threat Report* 2011). Additional functions that spyware can perform include listening into phone calls, monitoring text messages, pinpointing the location of the mobile device and monitoring Internet usage (Wilson and Kifayat 2012). Information collected by a mobile bot acting as spyware can be used to formulate various attacks such as phishing and social engineering (Wilson and Kifayat 2012).

Mobile botnets can also cause privacy threats and gather sensitive information from the mobile device. Sensitive information found on the mobile device includes the phone number, contact list, location data, personally identifiable information and even financial information (*Lookout Mobile Threat Report* 2011). Collecting a subset of such sensitive information can give the botmaster the opportunity to perform targeted attacks such as identity theft, spear phishing (instead of targeting random individuals, spear phishing targets specific individuals or companies (Hong 2012)), whaling (similar to spear phishing but instead focuses on high profile targets such as senior executives (Hong 2012)) and financial fraud.

Mobile devices can expose the cellular network to threats, such as a cellular DDoS attack. A cellular DDoS attack has the same aim and purpose as a computer-based DDoS attack: causing a disruption in or prevention of available services. Such an attack primarily targets call centres, either automated or manned, and can severely cripple their services (Wilson and Kifayat 2012). This threat can become a significant cyber weapon for terrorism (Wilson and Kifayat 2012).

It is important to remember that mobile botnets are malware, designed to engage in malicious behaviour on the mobile device. Their capabilities are advanced and they have the potential to perform various actions without a user's knowledge. Such actions include making charges to the user's phone bill, sending unsolicited SMS messages to the numbers in the contact list, installing or removing applications or giving a botmaster complete control over the mobile device (*Lookout Mobile Threat Report* 2011).

The potential capabilities and threats posed by mobile botnets are similar to those of traditional botnets. There is, however, one difference and that lies in the awareness and knowledge of the malware. There is more awareness of the threats of traditional botnets compared with mobile botnets and most people take the necessary precautionary steps to prevent their attacks. With so little knowledge of mobile malware, and especially mobile botnets, currently available, people do not feel the need to protect their mobile devices adequately and this leaves many opportunities for botmasters to explore and exploit. The possible countermeasures for mobile botnets as proposed by researchers in current literature are examind in Section 4.5.

4.5 Proposed Countermeasures

Mobile botnets are a real threat and are capable of advance attack capabilities and threats. As with traditional botnets, there is a need to design and develop countermeasures for mobile botnets. Since mobile botnets provide new capabilities for botnet design, it will be necessary to look beyond traditional countermeasures.

Current mobile botnets mainly use social engineering techniques, such as spam SMS messages, and malicious applications published on websites to propagate the bot code. The propagation mechanisms are very important contributors to the overall success of a mobile botnet. Without properly constructed propagation mechanisms, the mobile botnet will not be able to grow to a sufficient size and can become a meaningless entity. Therefore, the propagation mechanisms have become the starting point for the development of countermeasures. In order to counter spam SMS messages, defenders can deploy a worm detection system at the short message service center (SMSC) level (Xiang et al. 2011). The SMSC is a component of the mobile network and is responsible for the delivery of SMS messages (Brown et al. 2007). Such detection schemes of the SMSC will be able to detect multiple mobile botnets that use static SMS messages either for propagation or for C&C.

Defending against malicious applications is more complex but not impossible. With malicious applications acting as propagation mechanisms, defenders must pay more attention to software distribution management (Xiang et al. 2011). Thus defenders should use cloud-based sandboxes (a controlled environment within a cloud used to analyse malicious applications), virtual machines or updated anti-virus software to analyse and verify the new applications before making them available to the end-users (Xiang et al. 2011). If the defenders are able to discover malicious applications by using these techniques, the malicious applications can be removed before reaching the end-users and the propagation of the mobile botnet will fail.

The use of SMS messages as both propagation mechanisms and C&C channels have become popular with general mobile botnet designs. The reason for the popularity of SMS messages is the secrecy that can accompanying such messages. Botmasters can easily hide malicious content within SMS messages and is also possible to send silent SMS messages (Croft and Olivier 2007). If defenders could destroy such secrecy and inform mobile device users of every activity taking place on the mobile device, such malicious use of SMS messages would never succeed. Thus one step that mobile OS vendors can take is to allow a dialogue pop-up to occur for every suspicious activity, such as the silent sending of SMS messages by an application (Hua and Sakurai 2012). With the Android OS, pop-up dialogues can be successful as a detection mechanism of silent SMS messages since each Android application can only access their own activities and services. Therefore, a malicious application cannot prevent a pop-up caused by another application. Such a detection mechanism can further be improved by assigning system privileges to the application. If the malicious application is disguised as a user application on an unrooted device, the application will not be able to access the detection mechanism and therefore not prevent the pop-up dialogues from appearing. Although this is not a complete solution, it can alert the mobile device user of potential malicious applications.

Another countermeasure that can defend against mobile botnets is the use of honeypots (Hua and Sakurai 2012, Hua 2012). Honeypots are a common defence mechanism against traditional botnets (see Section 2.6.1) and defenders can apply the same principles to safeguard against mobile botnets. Defenders can develop special honeypots to aid the process of detecting mobile botnets by focusing on the propagation mechanisms. If the mobile botnets use wireless protocols such as Bluetooth or Wi-Fi to spread, the defenders can distribute a vulnerable honey-phone in a crowded location and enable the Bluetooth and/or Wi-Fi (Hua and Sakurai 2012, Hua 2012). The honey-phone will then wait for an attack to occur and the installation of the malicious bot code (Hua and Sakurai 2012, Hua 2012). Once installed, the defenders can reverse engineer and analyse the code with the possibility of detecting the C&C server and finally destroying the mobile botnet (Hua and Sakurai 2012, Hua 2012).

An additional possibility to counter against mobile botnets is to use patching mechanisms similar to those found in the PC world (Zeng et al. 2012). To prevent mobile botnets from infecting mobile devices by exploiting vulnerabilities, mobile OS vendors and application providers need to release patches to end-users in a timely manner (Zeng et al. 2012). Ensuring that the mobile OS and all of the installed applications are up to date can seriously cripple the performance of a mobile botnet. This countermeasure, however, requires active involvement from the mobile device user in order to be successful (this is further explored in Chapter 8).

One last measure to counter against mobile botnets is to use host-based approaches similar to those used to detect traditional botnets. Host-based techniques, such as signature-based or behaviour-based techniques, will be able to detect mobile botnets at runtime, minimising the impact of the mobile botnet (Zeng et al. 2012).

To be able to defend against mobile botnets, additional protection mechanisms are necessary. The possible countermeasures and how they can be applied were examined in this section. The following section will deal with advanced mobile botnet designs as found in current literature.

4.6 Related Research

This section introduces past literature relevant to the research carried out in this study. All of the related literature included in this section focuses on the development of mobile botnets and the evaluation of mobile botnet C&C.

Singh et al. (2010) evaluated the suitability of Bluetooth as a possible C&C channel in the design of a mobile botnet. With the Bluetooth C&C structure, each mobile bot acts as a peer in the mobile botnet, listening for new commands and forwarding the commands to the other discovered bots. During the initial infection, the mobile bots register a universal unique identifier (UUID) in the service register present in the mobile device. This allows the mobile bot to be discovered by the other bots as they come within range. The mobile bot then waits for new incoming connections and when such a connection arrives, the mobile bot establishes a two-way Bluetooth connection for communication. An advantage of Bluetooth C&C is that it prevents a defender from easily taking down the mobile botnet since the defender needs to be in range of the mobile bots when communication takes place. This may not always be possible due to the changing topology of the network. Even though Bluetooth provides the botmaster with a stealthy C&C channel, it requires the mobile bots to be within range of one another to propagate the commands successfully (Singh et al. 2010). This proposed mobile botnet design only explores a single C&C channel, namely Bluetooth, and only used simulations and not a physical constructed channel to explore the Bluetooth C&C channel. The close proximity of Bluetooth connections will limit the efficiency of the mobile botnet and therefore the Bluetooth C&C channel on its own is not a complete solution as C&C for a mobile botnet.

Zeng et al. (2012) proposed an SMS commanded and controlled and P2P structured mobile botnet. For propagation the mobile botnet employs user involvement and vulnerability exploits. The mobile botnet uses an SMS C&C channel to provide communication between the mobile bots and the botmaster. For the mobile bots to successfully receive the commands via SMS messages, each mobile bot includes an 8-byte passcode. In addition, the SMS messages are formulated as spam messages to further prevent the detection of the commands sent via SMS messages. To disseminate the SMS messages containing the commands, the botmaster exploits SMS services provided by the Internet to forward the messages. The topology of the mobile botnet is P2P and the study experiments with both structured (Kademlia) and unstructured (Gia) P2P topologies. Based on the results of the simulations, a modified Kademlia topology is a better choice as the topology for a mobile botnet (Zeng et al. 2012). This mobile botnet only uses SMS messages as a C&C channel and did not explore other possibilities for C&C. The use of SMS messages alone is not a complete solution as this can become a very costly channel.

Xiang et al. (2011) proposed the design of a mobile botnet, named Andbot, running on mobile devices hosting the Android operating system. The Andbot uses a centralised C&C topology and permits the mobile bots to connect to a fixed number of C&C servers to obtain the commands. Andbot also provides the following features: stealth by using HTTP-based uniform resource locator (URL) Flux protocol, access to the Internet via a background service and resilience to the most known defence strategies such as DNS sinkhole and command injections. The support of various commands and the low consumption of cost, network traffic and battery power are additional advantages provided by the Andbot design (Xiang et al. 2011). The design of the Andbot lacked the experimentation of new features associated with mobile devices and only introduced concepts of traditional botnets on mobile platforms.

Geng et al. (2012) proposed an SMS-based heterogeneous mobile botnet. The mobile botnet is built on a heterogeneous multitree network and uses SMS messages to disseminate commands. The design of the mobile botnet consists of the following different node types: bot master node, collection node, bot server node, region bot server node and bot node. The bot master node is responsible for controlling all of the nodes in the mobile botnet and has a direct connection to the bot servers. The collection node stores all of the valuable information it receives from the bots in the mobile botnet. The bot server nodes are only in control of a subnetwork that consists of a region bot server node and bot nodes. The region bot server node receives commands from the bot server and forwards them to the bot nodes in the subnetwork. The bot node, or leaf node, receives the commands and is then responsible for the execution of the command (Geng et al. 2012). This design is an expansion of the research done by Geng et al. (2011). The SMS-based heterogeneous mobile botnet does provide a well-established proof-of-concept mobile botnet for future developments but lacks the uses of multiple C&C channels. The heterogeneous multitree network makes this mobile botnet flexible and scalable, but the replacement of failed bots within the network can become expensive when using SMS messages.

Hua and Sakurai (2012) focused on designing two separate mobile botnets using SMS messages and Bluetooth technology for command propagation. The SMS-based mobile botnet uses SMS messages to propagate the C&C messages by using a simple flooding algorithm. The proximity-based mobile botnet uses Bluetooth to forward the C&C messages and form the communication channel around seed nodes, which are selected based on their contact frequency, with other infected nodes. After multiple simulations, the authors proved that a uniform random graph is the most efficient topology for the SMS-based mobile botnet and that human mobility features can improve the command propagation of proximity-based mobile botnets (Hua and Sakurai 2012). Although Hua and Sakurai (2012) do explore with both SMS and Bluetooth as potential C&C channels, the authors did not explore the combination of the two channels and the effects it might have on the overall mobile botnet design. In terms of the Bluetooth C&C, only simulations were used to construct the channel, but no physical Bluetooth C&C channel was designed.

Faghani and Nguyen (2012) proposed a mobile botnet design, called SoCellBot, which uses online social networks (OSNs) to propagate commands. The purpose of the OSNs is to minimise the use of SMS messages and therefore each bot will receive the command through an online social network messaging service (OSNMS). This mechanism overcomes the existing challenges associated with SMS-based botnets such as cost and detection by users or cellular network providers. This design shows the feasibility of mobile botnets exploiting OSNs such as Facebook and Twitter (Faghani and Nguyen 2012). Only using social networks as a C&C channel causes the mobile botnet to become dependent on mobile devices with the required social networks, limiting the growth potential of the botnet.

Zhao et al. (2012) provided the first mobile botnet design that explores the concept of cloud computing as a C&C channel. The design, called cloud to device messaging (C2DM) botnet, uses Google's C2DM service for the Android platform. It involves no direct communication between the botmaster and the mobile bots but instead exploits the C2DM service as a relay, using the push notification service as the C&C channel. This provides the botmasters with the ability to send commands to the mobile bots via the C2DM service while simultaneously hiding the botnet traffic within legitimate C2DM traffic caused by other mobile applications (Zhao et al. 2012). The C2DM botnet is the first mobile botnet design to move away from traditional C&C designs such as SMS and explore other C&C possibilities. The use of specific platforms such as Google's C2DM and the Android OS limits the adaptability of this design.

Shuai et al. (2013) proposed a design method of a mobile botnet, which is based on the HTTP and the shorten URL (S-URL) Flux protocols. This mobile botnet, called the SUBot, is an expansion and improvement of the Andbot (Xiang et al. 2011). The difference between the two mobile botnet designs is the addition of the S-URL Flux protocol to the design of the C&C channel of the SUBot. This protocol provides the SUBot with the necessary functionality to only access one network server to retrieve the command, instead of using two network servers as done by Andbot (Shuai et al. 2013). Although the SUBot design is an improvement on the original Andbot design, it still lacks the experimentation of using multiple C&C channels.

Li et al. (2013) proposed a mobile botnet design that uses a combination of Twitter and SMS as a C&C channel. The mobile botnet primarily uses the Twitter server to disseminate commands but if any bot's Twitter account fails, the SMS C&C channel is used instead. The SMS C&C channel thus acts as a disaster recovery channel that can communicate with the lost mobile bots. The structure of the mobile botnet follows a P2P-structured topology with no centralised C&C infrastructure, thus improving the robustness of the design (Li et al. 2013). This design is highly dependent on a single C&C channel (Twitter) and if the Twitter server ever fails, this design will become a simplistic SMS-based mobile botnet.

Sheu et al. (2013) designed an Android-based mobile botnet called the JokerBot. This design uses its own communication mechanism and exploits built-in functions of smartphones. The communication mechanism is built around SMS and HTTP and is used by all of the mobile bots to communicate. JokerBot also deploys special techniques to bypass Android's Internet user permission (Sheu et al. 2013). Similar to the C2DM botnet (Zhao et al. 2012), this design is also platform specific and therefore has limited adaptability possibilities.

Mulliner (2011) designed an iPhone-based mobile botnet. The design combines an SMS-HTTP hybrid approach with a P2P topology. The purpose of the hybrid schema is to split the command dissemination into two separate parts: SMS and HTTP. Pre-crafted SMS messages are stored as encrypted files on a website and the URL of this website is sent to a random selection of bots by the botmaster (Mulliner 2011). The design showed that hybrid C&C schemas are possible within mobile botnet designs. However, the design has one flaw: a single point of failure. If the website goes down, the bots using the HTTP C&C channel will not be able to receive the new command. Using multiple URLs can, however, potentially solve the concern of single point of failure.

The multiple mobile botnet designs explored in this section show that current mobile technology exhibits all the capabilities required to develop and improve mobile botnets.

4.7 Conclusion

This chapter revealed that mobile botnets are a real threat and can have severe consequences if ignored. Their potential was confirmed by exploring their definition, characteristics and history. A comparison was also made between mobile botnets and traditional botnets in order to establish their differences and similarities. The potential capabilities and threats of mobile botnets, as explored in existing literature, were briefly discussed, followed by possible countermeasures. Multiple mobile botnet designs were explored and it was found that the primary focus of these designs is the structure of the C&C channels. None of these designs, however, investigated the possibility of using a hybrid design, which will use multiple C&C channels to disseminate the commands. In view of the above, this study designs a new mobile botnet that uses a hybrid C&C structure consisting of the following C&C channels: SMS, Bluetooth and HTTP. This new design will show that current mobile technology exhibits the necessary capabilities to support hybrid mobile botnet development and execution. The construction of the new mobile botnet design can lead to the identification of mobile botnet attack strategies and security steps, which can be used to defend against their attacks.

Chapter 5

Hybrid Mobile Botnet Design

A variety of mobile botnet designs are found in literature, as mentioned in the previous chapter, but these designs also introduce a collection of shortcomings in the overall study of mobile botnets. Firstly, none of the above mobile botnet designs explores the possibility of using hybrid C&C structures. Each design thus far uses no more than two C&C channels, with the popular channels being Bluetooth and SMS messages. Secondly, where Bluetooth is explored as a potential C&C channel, only simulations are used to design such a channel and in no literature thus far has a physical Bluetooth C&C channel been constructed. Finally, in order to defend against mobile botnet attacks, their potential capabilities and threats must be known.

This chapter considers the detailed design of a proof of concept mobile botnet, called the Hybrid Mobile Botnet. The purpose of this new design is to determine the efficiency of using a hybrid structure consisting of multiple C&C channels to achieve a stealthy, cost-effective and robust design, while also illustrating that current mobile technology exhibits all the required capabilities needed to develop and support hybrid mobile botnets. Knowledge obtained from designing this new mobile botnet can aid the analysis of potential mobile botnet attack strategies and the design of security steps to help users defend against such attacks.

An in-depth study is given in this chapter of the internal design of the Hybrid Mobile Botnet. As with any other mobile botnet design, the Hybrid Mobile Botnet consists of the following main components: propagation vectors, C&C channels and topology.

The propagation vectors are responsible for disseminating the malicious bot code to the mobile devices; the C&C channels transport the command messages and the topology describes the structure of the mobile botnet.

In order to achieve a stealthy, cost-effective and robust design, the Hybrid Mobile Botnet should have the following characteristics:

- No single point of failure in the mobile botnet topology.
- Low (monetary) cost for command dissemination.
- Limited network activities (such as Internet and Bluetooth connectivity).
- Low battery consumption per mobile bot.

The rest of the chapter is structured as follows: the propagation vectors, the C&C channels and the topology layout of Hybrid Mobile Botnet are introduced in Sections 5.1, 5.2 and 5.3, respectively. Section 5.4 provides a detailed description of the spreading of the commands within the topology of the Hybrid Mobile Botnet. Section 5.5 concludes the chapter.

5.1 Propagation Vectors

In general, propagation vectors are responsible for spreading the malicious code to uncontaminated components. With computer viruses, a host program is modified to include an evolved copy of the virus code (Cohen 1987). The virus is thus dependent on the host program for propagation. A computer worm is a self-contained and self-propagating program which does not require the use of a host program for propagation (Serazzi and Zanero 2004). The worm will exploit vulnerabilities within a computer network that will allow for propagation. A Trojan horse can be described as a destructive program that masquerades as a legitimate program (Hughes and DeLone 2007).

A popular propagation technique currently used by mobile botnets and mobile malware comes in the form of a Trojan horse (Eldridge 2013). Trojan horses are responsible for creating a backdoor on mobile devices, giving botmasters access to the devices

(Fuentes 2010). This propagation technique, however, relies on the user to download and manually install the application infected with the Trojan horse.

Since the propagation vectors of the Hybrid Mobile Botnet are not the main focus of this design, the Hybrid Mobile Botnet will follow the current convention and propagate by means of an infected application. Due to the reliance on user involvement, this propagation vector can suffer from a slow infection rate. A second propagation vector, propagation by contact numbers, can be activated by the botmaster should the first propagation vector proceed too slowly. The second propagation vector allows the Hybrid Mobile Botnet to grow faster and gives the botmaster control of the Hybrid Mobile Botnet as soon as infection starts.

5.1.1 Propagation via an Application

Propagation via an application allows the malicious bot code to distribute by means of a repackaged application. Such an application is well known and legitimate but the original code has been altered and repackaged with the additional bot code. A user installs the application but is unaware of the additional configurations taking place in the background of the mobile device.

For this propagation vector to succeed, the botmaster selects an application that is currently popular among the users of mobile devices. The popularity of the selected application will determine the rate at which the bot code spreads. In the next step the botmaster reverse engineers the application and includes the malicious bot code without affecting the original code modules. The botmaster then returns the repackaged application to the application market where it awaits downloading.

The motivation behind deploying this propagation vector is twofold. Firstly, returning the malicious application to the application market provides the Hybrid Mobile Botnet with the ability to reach a wide audience. Secondly, choosing a popular application also allows for the possibility that the malicious application can spread by word of mouth within social circles.

It is because of the above motivations that the Hybrid Mobile Botnet deploys this propagation vector. Should this propagation vector result in a slow infection rate, the botmaster has the option of using an additional propagation vector: propagation by contact numbers.

5.1.2 Propagation by Contact Numbers

This propagation vector utilises the contacts of a mobile device and spam messages to enhance the distribution of the bot code to other mobile devices. The botmaster selects SMS messages to deliver the spam messages since SMS messages are ubiquitous and available on most mobile platforms. The spam SMS message includes a link pointing to a website that is hosting the malicious application.

Before sending the spam SMS messages, the process of selecting a subset of the contact numbers must occur. The mobile bot collects all the existing contacts that reside on the infected mobile device. Only a certain subset of the collected contact numbers receives the spam SMS messages. This subset is chosen based on a specific selection process which consists of the following two steps: establishing the type of the contact number selected (whether it is a mobile phone or landline number) and the usage of the specific contact number. The outcome of the process mentioned above will determine whether a contact number is selected to fall in the subset or not.

The first step of the process is to determine whether the selected contact is a mobile phone number or not. In South Africa this is possible by evaluating the area code. In other countries, such as the US, applications are available that can distinguish between mobile phone and landline numbers. If the contact is a mobile phone number, the process proceeds to the next step. If the contact number is a landline number, the process discards the contact number and repeats the first step with a new contact number.

The process proceeds to the next step where the selected contact number enters a procedure to evaluate the frequency of use. The procedure determines the frequency of use of a specific contact by evaluating the missed and received calls, dialled numbers as well as the SMS and MMS messages stored in the inbox. A high occurrence of the contact in these locations indicates a regular use of the contact and so will increase the possibility of the recipient following a link included in the spam SMS message due to the familiarity with the sender. The process will discard any contact number with a very low frequency of use as the spam SMS messages have a smaller chance of succeeding. All of the remaining contact numbers proceed through the same process. The botmaster can alter this process during the lifetime of the mobile botnet, especially if a higher frequency of spam SMS messages is required.

This propagation vector allows the botmaster to disseminate the bot code at a faster rate and to reach a large number of mobile devices over a large geographical area. The familiarity associated with the contact numbers will possibly increase the opportunity for this propagation vector to succeed.

The purpose of the propagation vectors is to distribute the bot code and allow for the growth of the mobile botnet. In the case of the Hybrid Mobile Botnet, this is the responsibility of the propagation vectors described above.

5.2 Command and Control Channels

The C&C channels are the most important component of the mobile botnet as they are responsible for disseminating the commands from the botmaster to all the mobile bots currently active in the mobile botnet. Due to the critical nature of the C&C channels, they form an attractive target for a defender trying to bring the mobile botnet down or for other botmasters to intercept this communication channel (Mulliner 2010). It is therefore necessary to carefully plan and design the C&C channels to be able to successfully deliver the commands while remaining resilient against possible attacks. Most of the available mobile botnet designs use a single C&C channel, with the popular options being SMS and Bluetooth. With the C&C structure being a popular target for defenders (Liu et al. 2011), a single C&C channel can easily become a single point of failure within the mobile botnet design. This particular problem has already occurred in traditional botnet designs which used IRC for C&C (Fedynyshyn et al. 2011). Therefore to achieve robustness and stealth the Hybrid Mobile Botnet follows a hybrid C&C structure and will use a combination of the popular C&C channels, SMS and Bluetooth, while also including HTTP. The purpose of the selection of these C&C channels is further explained in the upcoming sections.

5.2.1 SMS C&C Channel

SMS is one of the many services offered by mobile phone networks. The networks use SMS for text messaging and the execution of background services that are not always visible to the end-users (Mulliner and Seifert 2010). There are multiple advantages of SMS messages that make them a suitable channel for C&C. These advantages include (Zeng et al. 2012):

- Ubiquity: Most mobile phones can handle SMS messages.
- Offline accommodation: A service centre stores SMS messages if the recipient's mobile device is turned off.
- Hiding malicious content: An SMS message can hide malicious content.
- Multiple send and receive possibilities: Currently there are multiple ways to send and receive free SMS messages; one such an example is using a Gmail account to send SMS messages.

For the botmaster to utilise SMS messages as a C&C channel, certain characteristics must be taken into consideration. These include the cost of sending SMS messages, preventing the detection of the SMS messages by the mobile device users, covertly receiving SMS messages and hiding the SMS messages from being filtered by the telecom operators (Hua and Sakurai 2011). Taking the above characteristics into consideration and minimising their impact will lead to the design of a stealthy SMS C&C channel. These characteristics will also structure the internal design of the SMS C&C channel.

Sending SMS messages is a costly business and it is a characteristic of the SMS C&C channel that the botmaster wants to minimise or avoid entirely. Currently there are services available that offer free SMS texting via web interfaces. Such an example is Text4FreeOnline and it offers free text messaging to mobile devices around the world (Text4FreeOnline 2010). Such websites offer the botmaster the opportunity to send multiple SMS messages without incurring any costs and possibly also keeping the identity of the botmaster hidden.

If such free texting is unavailable or unsuitable, the botmaster will have to directly forward the SMS messages containing the commands. To hide his/her identity, the botmaster can send SMS messages from unprotected devices (mobile devices that are lost or left unoccupied) or regularly swop out the subscriber identity module (SIM) cards. Although this second measure can incur costs for the botmaster, the cost will be minimal due to the layout of the topology of the Hybrid Mobile Botnet (see Section 5.3).

SMS bundles are available for mobile device packages, offering a specific number of free SMS messages. Due to the inability of determining whether the infected mobile device has such a bundle, the Hybrid Mobile Botnet will still minimise the use of SMS messages.

Mobile device users are bound to grow suspicious if they regularly receive unexpected spam SMS messages. Their suspicion is likely to rise even more if the spam SMS messages contain strange content. Such suspicion will eventually lead to the discovery of the mobile bot and can even lead to the discovery of the Hybrid Mobile Botnet. To prevent the user from detecting the commands being sent as SMS messages, the mobile bot will intercept all incoming SMS messages that contain a specific passcode in the message body before reaching the inbox. The mobile bot will interpret the SMS message, extract the command and afterwards delete the message. This passcode is unique to the Hybrid Mobile Botnet and enables the mobile bot to identify messages sent from the botmaster. All other SMS messages will pass through to the inbox and so avoid any suspicion from growing.

In order to prevent the SMS messages containing the command being filtered by the telecom operators, these messages will be formulated as normal conversation messages. In addition, the passcode will be added to the SMS message in order for the mobile bot to identify the SMS message as a command sent by the botmaster. Formulating the command messages as normal conversation messages will prevent the telecom operators from identifying these messages as malicious.

The communication occurring across the SMS C&C channel is unidirectional, meaning that the commands are sent in one direction only. When a mobile bot receives the command via an SMS message, it does not send any response back across this channel as this will incur costs. The mobile bot instead directly connects to the control server to confirm that the command has been received.

The above discussion reveals that it is possible to construct a stealthy SMS C&C channel, which is capable of command dissemination. SMS provides a cost-effective, ubiquitous and easy to use C&C channel.

5.2.2 Bluetooth C&C Channel

The Bluetooth technology was invented in 1994 by L.M. Ericsson (Sairam et al. 2002). During the winter of 1998 Ericsson, Nokia, Intel, IBM and Toshiba further evolved the Bluetooth standard by establishing the Bluetooth Special Industry Group (SIG) (Bisdikian 2001). The past decade have seen 3COM, Microsoft, Lucent and Motorola also participating in SIG (Sairam et al. 2002). The purpose of Bluetooth wireless technology is to serve as a replacement for the interconnected cables found between various personal devices such as notebooks and small-scale devices (Bisdikian 2001). Bluetooth provides short-range, low-cost and user-friendly connectivity between portable devices and also allows for ad hoc connectivity between such devices (Bisdikian 2001). Bluetooth has been available on mobile devices since 2000 (*Bluetooth* 2013), providing the opportunity to develop a stealthy mechanism for command dissemination by exploiting various Bluetooth vulnerabilities. Due to these advantages of Bluetooth wireless technology, it becomes a suitable C&C channel within the structure of a mobile botnet.

To design a suitable Bluetooth C&C channel, there is an important aspect concerning Bluetooth that must be taken into consideration. Bluetooth, like any other electronic component, consumes battery power. If Bluetooth is left on indefinitely, it will quickly consume the battery power of the mobile device, which can lead to the discovery of the mobile bot. To minimise the consumption of the battery due to Bluetooth activities, Bluetooth will only be active during specific periods of a day as well as only for a limited period. These periods of activity are based on patterns of human mobility. Previous research by Aviv et al. (2010) reveals that human mobility patterns can successfully be incorporated into the design of communication structures as demonstrated by the design of the Human-to-human Mobile Ad hoc Networking paradigm (Aviv et al. 2010).

Human mobility patterns usually have two useful properties: regularity and chaotic (Aviv et al. 2010). These patterns tend to be regular since people repeatedly visit the same places over and over again (Aviv et al. 2010). The patterns can become chaotic in the sense that an act of randomness can alter the regular pattern, causing a given person to contact different people over a specific time or visit different places (Aviv et al. 2010). The patterns that tend to be regular are divided into three periods of mobility: no mobility, low mobility and high mobility. These periods of mobility are defined in terms of stability and availability.

- No mobility:
  - Stability: Stability is high with no changes in geographical positioning.
  - Availability: Active for long periods, during nightfall and early morning hours, when people are sleeping.
- Low mobility:
  - Stability: Stability is moderate with infrequent changes in geographical positioning.
  - Availability: Active for moderate periods, during daylight hours, when people are actively working.
- High mobility:
  - Stability: Stability is low with frequent changes in geographical positioning.
  - Availability: Active for short periods, during morning hours and late afternoons as people travel to their required destinations.

During the period of no mobility, no changes occur in geographical positioning since people are at home and usually sleeping during this period. To minimise battery consumption, the mobile bot will not activate the Bluetooth C&C channel during this period due to the low contact frequency with other devices.

As stated previously, people usually follow regular patterns and visit the same locations at regular intervals. These places are possibly offices, homes, schools, colleges, etc. Movement of people at these locations tends to be moderately stable as they remain in their offices or classrooms for long periods, thus allowing for low mobility. During this period, a specific person comes into close contact with many other people regularly and for substantial periods.

Then there are periods when the mobility of people is quite high and they move around frequently. During these periods of high mobility, a person comes into close contact with many different people but only for very short time intervals. Mobile devices will not be always able to construct the Bluetooth C&C channel during these periods, making communication during these periods impractical.

From the three periods of mobility described above, the period of low mobility provides the most stable period for the longest available time and therefore the Bluetooth C&C channel will only be active during this particular period. Activating Bluetooth only during periods of low mobility will in turn minimise the impact of the Bluetooth C&C channel on battery consumption.

Besides the consumption of battery power, Bluetooth technology requires user involvement to pair two mobile devices. Users are required to enable the Bluetooth, authorise the connection and authenticate the connecting device. For the Bluetooth C&C channel to be successful, the channel must require no user involvement. Therefore, to exclude any user involvement, the Bluetooth C&C channel performs the pairing of the devices on behalf of the user. Once the mobile devices are paired, communication can proceed via Bluetooth without requiring any user involvement. Thus this period of low mobility provides the best opportunity for command dissemination. The mobile bots activate the Bluetooth C&C channel for specific time intervals during this period to enable the communication to occur. The available time intervals are limited to minimise the consumption of battery power.

The communication across the Bluetooth C&C channel is bidirectional, forming a two-way communication channel. In order to improve the performance of the Hybrid Mobile Botnet, a mobile bot that successfully receives the command will return a response to the mobile bot responsible for sending the command. This allows for easy identification of mobile bots that still need to receive the command and in turn will minimise the use of the Bluetooth C&C channel.

The requirements for a physical Bluetooth C&C channel are thus:

- Only be active during the period of low mobility.

- Only be enabled for limited periods.
- Must require no user involvement.
- Communication across the channel must be bidirectional.

Bluetooth is a suitable C&C channel for mobile devices at close range. Due to human mobility periods, mobile devices come into close contact with other mobile devices at regular intervals during a day. These regular intervals allow for the mobile bots to disseminate commands cost-effectively via Bluetooth. Meeting the above listed requirements will lead to the development of an effective and efficient communication channel for the Hybrid Mobile Botnet.

5.2.3 HTTP C&C Channel

The botmaster requires knowledge about the Hybrid Mobile Botnet and all the bots that are actively participating. To identify a mobile bot within the Hybrid Mobile Botnet, the botmaster requires information such as the mobile phone number, the Bluetooth media access control (MAC) address and location data of the mobile device. For a botmaster to acquire this information, both the SMS and Bluetooth C&C channels are inadequate. Although it is possible to send the collected information via SMS messages to the botmaster, this will lead to an increase in the bill which may in turn lead to the detection of the mobile bot and the identity of the botmaster. Using Bluetooth to send the collected information is also impractical since the botmaster needs to be within range of the mobile bots. Thus the Hybrid Mobile Botnet requires an additional channel to transport the information to the botmaster. The additional channel uses HTTP and it allows a mobile bot to transfer information to the control server. The botmaster hosts the control server which is responsible for managing and storing information related to the Hybrid Mobile Botnet.

HTTP was selected as the third C&C channel because most mobile devices are continuously connected to the Internet. Since mobile traffic remains relatively inexpensive, the chances are slim that mobile device users will discover the additional networking activities.

The purpose of the HTTP C&C channel is to forward the information collected by a mobile bot to the control server. Such information includes mobile phone number, Bluetooth MAC address, location data and stolen information which can include the IMEI and IMSI numbers. This information will only be transported across the HTTP C&C channel during the period of low mobility when the mobile bot first infects the mobile device. At all other times the HTTP C&C channel will remain inactive.

It is the responsibility of the Hybrid Mobile Botnet to encrypt any information sent to the control server. The control server then stores the received information in the encrypted format. The botmaster also deploys multiple backups of the information stored by the control server in case of a failure or external compromise. At any time during the lifetime of the Hybrid Mobile Botnet the botmaster can contact the control server and retrieve the necessary information.

The communication across the HTTP C&C channel can occur as both unidirectional or bidirectional, depending on the content transported across this channel by a mobile bot. With the initial structuring of the Hybrid Mobile Botnet the communication is unidirectional since the mobile bot only transfers information to the control server without requiring a response in return. During the process of command dissemination all communication across the HTTP C&C channel is bidirectional. With every upload of data, the control server sends a response back to the mobile bot with a message that contains specific information relating to the Hybrid Mobile Botnet.

The HTTP C&C channel is thus there purely to support the construction of the ever-changing Hybrid Mobile Botnet.

The combination of SMS, Bluetooth and HTTP provides the Hybrid Mobile Botnet with a hybrid C&C structure. This allows the botmaster to spread the monetary cost of communication across these multiple channels, minimise battery consumption and limit network activities. The hybrid C&C structure improves the robustness of the Hybrid Mobile Botnet and increases the difficulty for defenders to detect this mobile botnet.

5.3 Topology of the Mobile Botnet

A mobile botnet consist of a collection of compromised mobile devices that need to be organised into a structure. This structure is often referred to as the topology of the mobile botnet. The topology of the mobile botnet allows for the efficient dissemination of the commands to all the bots currently participating in the mobile botnet. The topology of the Hybrid Mobile Botnet is closely modelled on the cluster-based scheme for mobile ad hoc networks (MANETs) (Tseng et al. 2002). In this scheme, the MANET is divided into clusters, which is a collection of mobile hosts. Each cluster contains a cluster head that is the representative of that specific cluster (Tseng et al. 2002). The cluster-based scheme was selected as the topology design for the Hybrid Mobile Botnet because it allows for reduced redundancy, contention and minimised collisions (Tseng et al. 2002).

In order to describe the topology of the Hybrid Mobile Botnet in further detail, certain terminology must be clarified:

Mobile bot is a compromised mobile device, infected with malicious bot code. A mobile bot can assume two distinct roles during its participation in the Hybrid Mobile Botnet: cluster head bot or receiver bot.

Cluster head bot is a mobile bot that directly receives commands via SMS messages from the botmaster or another cluster head bot. It is responsible for forwarding the commands to the receiver bots according to its locally stored bot list and to other cluster head bots according to its locally stored command list. The control server is responsible for electing the cluster head bots.

Receiver bot is a mobile bot that receives commands from the cluster head bot. A receiver bot can be in one of the following distinct stages: active or inactive. An active receiver bot participates in the process of command dissemination while an inactive receiver bot executes the commands, performs activities on the local mobile device and excludes any command dissemination.

Botmaster is the entity responsible for forwarding the commands via SMS messages to a selection of cluster head bots.

Bot list is a file stored on each mobile bot that lists the bot IDs of the other mobile bots within a specific cluster botnet.

Command list contains the mobile phone numbers of the cluster head bots to which the command must be forwarded.

Cluster botnet is a collection of mobile bots capable of communicating via the Bluetooth C&C channel.

Global botnet is a collection of all the cluster botnets.

Control server is a server managed by the botmaster that stores information about the mobile bots actively participating in the mobile botnet.

Active period refers to the period that allows the mobile bots to exchange commands via the Bluetooth C&C channel. During this period, two mobile bots within range will pair using Bluetooth and exchange a command. As soon as the active period expires, the mobile bots disable the Bluetooth and communication is halted till the arrival of the next active period. An active period occurs at a specific time and location.

Bot ID is the Bluetooth MAC address of a mobile device.

The topology of the Hybrid Mobile Botnet is presented in Figure 5.1, which illustrates the cluster head bots and receiver bots, as well as the flow of communication between them. For simplicity and presentation purposes, the HTTP C&C channels between the mobile bots and the control server are not shown.

The global botnet consists of a collection of cluster botnets and forms a dynamic structure due to the constantly changing cluster botnets. Each cluster botnet, which also forms a dynamic structure, consists of a collection of mobile bots. The dynamic properties of both a cluster botnet and the global botnet are made possible by the infected mobile devices that move around according to certain human mobility patterns (see Section 5.2.2). Thus the topology of the Hybrid Mobile Botnet will continuously change as various mobile bots leave or join a specific cluster botnet over time.

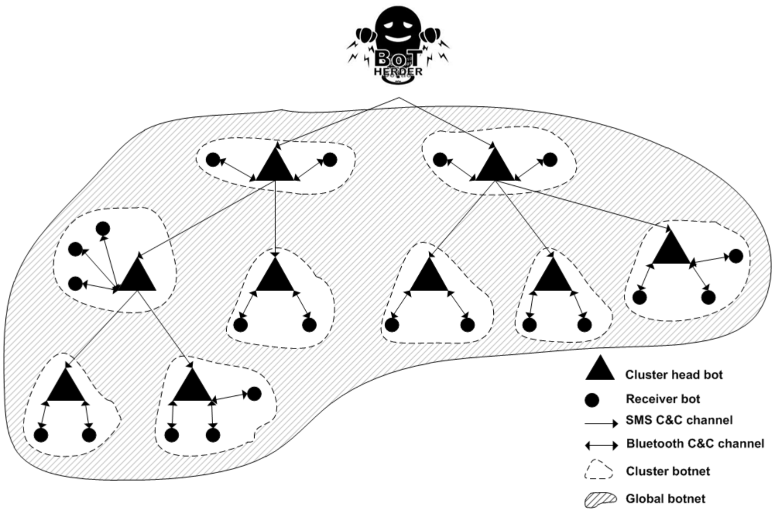


Figure 5.1: A visual representation of the topology of the Hybrid Mobile Botnet

Besides the dynamic property of both the cluster botnets and the global botnet, both still require structures that support the flow of communication. To allow communication to occur within a cluster botnet, the mobile bots utilise the Bluetooth C&C channel. The Bluetooth C&C channel is instrumental in the establishment of the topology of a cluster botnet since this channel requires the mobile bots to be within close range (10 metres) to communicate via Bluetooth. The requirement of close proximity therefore causes the cluster botnets to also be location dependent. Due to both the location dependence and the dynamic structure of cluster botnets, they will only exist for a specific period at a specific location. Thus only during these periods (referred to as the active periods) will the cluster head bot exchange a command via Bluetooth. Command dissemination will continue until all of the receiver bots within a cluster botnet have received the command.

To be of any significance to the botmaster, the mobile botnet must consist of a large collection of mobile bots. Such a large mobile botnet will lead to the development of many cluster botnets and will in turn also contain a substantial number of cluster head bots. With the Bluetooth C&C channel being location dependent, it is inadequate to use it as the global communication medium between all the cluster head bots. The remaining option for the botmaster is to send the commands via SMS messages to all of the cluster head bots. Due to the possible larger number of cluster head bots, it can become impractical for the botmaster to directly send the commands. To keep costs low, the SMS C&C channel utilises an arbitrary tree structured topology. The arbitrary property enables any cluster head bot to send out a specific number of SMS messages as specified by the botmaster. The arbitrary tree structured topology improves the stealth of the mobile botnet, keeps costs low and increases the difficulty of predicting the flow of command dissemination. The SMS C&C channel also offers the botmaster the ability to contact any available mobile bot within a cluster botnet and can also then regularly switch the communication among the various mobile bots.

The dynamic topology increases the complexity of detecting the Hybrid Mobile Botnet, but it also complicates the process of command dissemination. By using the C&C channels as described in this chapter thus far, the mobile bots can communicate in an effective and timely manner. The next section will focus on the dissemination of the commands through this topology.

5.4 Command Dissemination

The C&C infrastructure is a critical component of any mobile botnet since it allows the botmaster the possibility to communicate with the mobile bots in a timely manner. Without the option of sending and/or receiving commands, the mobile botnet becomes a purposeless entity. To enable command dissemination, the Hybrid Mobile Botnet deploys all of the C&C channels as described in this chapter. The remainder of this section deals with the basic tasks performed by a mobile bot to allow for command dissemination within the Hybrid Mobile Botnet.

5.4.1 Predefined Command Activities

All mobile bots include instructions for the execution of specific predefined activities that enable the process of command dissemination. These activities are essential to the successful dissemination of commands within the Hybrid Mobile Botnet. The activities fall into one of two categories: static or dynamic. Static activities only occur once during the initial infection of the mobile device.

- Static activities include:
  - Collection of the mobile phone number of the infected mobile device.
  - Collection of the Bluetooth MAC address of the mobile device.
- Dynamic activities occur at regular intervals during the lifetime of a mobile bot and include the following:
  - Collection of location data during the active period.
  - Verification that the control server is still active.

The mobile bots forward all the information collected by both the static and dynamic activities to the control server via the HTTP C&C channel.

5.4.2 Process of Command Dissemination

When the mobile bots have successfully completed the predefined command activities, they assume the role of an inactive receiver bot. A mobile bot acting as an inactive receiver bot does not participate in any command dissemination and only performs the following steps in chronological order:

1. It collects location data during the active period and stores the collected data in a file on the mobile device.
2. At the end of the low mobility period the mobile bot forwards the collected location data to the control server.
3. Directly after the upload of the location data, the mobile bot queries the control server and retrieves the Bluetooth MAC address of the elected cluster head bot.
4. It verifies the Boolean value of the command dissemination flag.
5. If the command dissemination flag is set to false, the mobile bot returns back to step 1.

Should the command dissemination flag be set to true, the mobile bot enters the active receiver bot role and performs all of the following steps:

1. The mobile bot retrieves the bot ID of the cluster head bot and updates the locally stored bot list.
2. It enables the Bluetooth and attempts a connection with the cluster head bot during the active period.
3. When the command is received from the cluster head bot, the mobile bot executes the command, returns to the role of an inactive receiver bot and proceeds with the collection of location data.

To initialise the process of command dissemination the botmaster sends the command to a minimum of two randomly selected cluster head bots. When a mobile bot receives a command via an SMS message from the botmaster, it immediately halts the collection of location data and assumes the role of a cluster head bot. The cluster head bot is responsible for executing all of the following steps:

1. It retrieves and intercepts the SMS message as sent from the botmaster or other cluster head bot.
2. It extracts the command from the SMS message.
3. The bot connects to the control server via the HTTP C&C channel.
4. It retrieves a list of randomly selected cluster head bots to which the command must be forwarded and updates the locally stored command list accordingly.
5. It retrieves all the bot IDs of the mobile bots falling within this specific cluster botnet and updates the locally stored bot list accordingly.
6. It sends out SMS messages containing the command to all of the mobile bots listed in the command list.
7. The bot enables and initialize Bluetooth connections with the active receiver bots during the active period.

8. When all of the mobile bots in the cluster botnet have received the command, the cluster head bot returns to the role of an inactive receiver bot.

To prevent a cluster head bot from indefinitely trying to send a command to a receiver bot that may no longer participate in the Hybrid Mobile Botnet, a send limitation is included. Each cluster head bot has only two opportunities to forward the command to the specific receiver bot during the active period. Should both attempts fail, the cluster head bot continues to forward the command to the remaining receiver bots. The send limitation enables the mobile bot to conserve battery power while preventing redundancy.

The steps described above allow the mobile bots to perform command dissemination in a timely manner.

5.5 Conclusion

This chapter established a foundation by introducing the design of the Hybrid Mobile Botnet. It provided an in-depth study of the three main components of the Hybrid Mobile Botnet (propagation vectors, command and control and topology) with specific focus on the execution of these components. The Hybrid Mobile Botnet performs propagation by means of an infected application, with the option to speed up the propagation by activating an additional propagation vector, propagation by contact numbers. To improve stealth, cost-effectiveness and robustness, the Hybrid Mobile Botnet follow a hybrid C&C structure that makes use of multiple C&C channels, namely SMS, Bluetooth and HTTP. The mobility of the mobile devices allows for dynamic topology, making it more difficult to detect. The command dissemination process, as followed by the Hybrid Mobile Botnet, was also clearly explained. As stated at the start of Chapter 4, the ultimate goal of a mobile botnet remains similar to that of traditional botnets and will resemble attacks such as DoS attacks and the sending of spam messages. The next chapter focuses on the different attack strategies that mobile botnets, as well as the Hybrid Mobile Botnet, can implement.

Chapter 6

Attack Strategies and their Implementation in the Hybrid Mobile Botnet

The previous chapter introduced the basic design and structure of the Hybrid Mobile Botnet. Propagation, C&C, topology and command dissemination were the main focal points of the chapter. However, there was no discussion in the potential output of the execution of the commands that can be disseminated efficiently by means of this design. This chapter introduces the various attack strategies that the Hybrid Mobile Botnet can implement and the effect of the chosen attack strategy on the process of command dissemination. The implementation of an attack follows three different stages. Section 6.1 focuses on the selection of an attack strategy, with the orientation of the selected attack strategy identified in Section 6.2 and its impact on the process of command dissemination in terms of the implementation in the Hybrid Mobile Botnet discussed in Section 6.3. Section 6.4 concludes the chapter.

6.1 Attack Strategies

A botmaster develops any mobile botnet with a specific purpose in mind. This purpose is to achieve a certain goal. This goal varies from botmaster to botmaster but will most

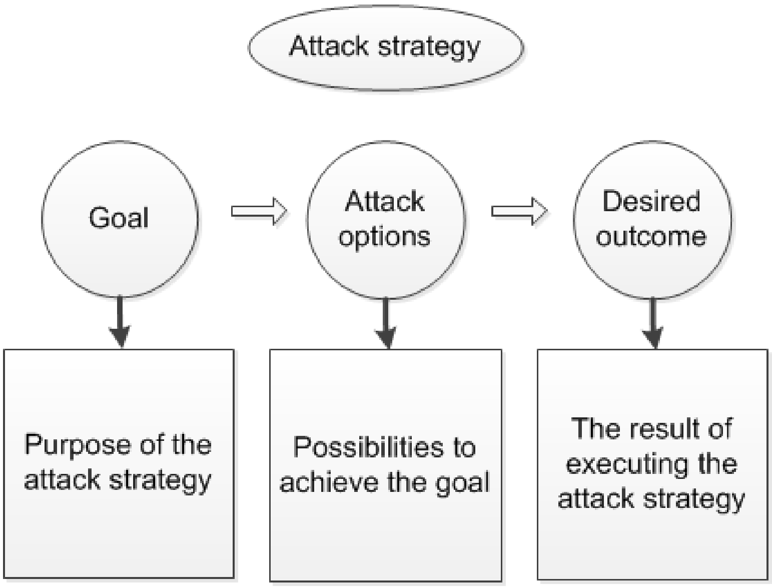

Attack strategy
Goal
Attack options
Desired outcome
Purpose of the attack strategy
Possibilities to achieve the goal
The result of executing the attack strategy

Figure 6.1: Visual layout of an attack strategy

likely be built around the following examples: collecting information, stealing money, blocking services, or sending spam messages. These various goals are summarised into one of the following attack strategies: monetary gain, information dispersal, information reaping (controlled and uncontrolled) and service interruption. Each attack strategy is defined according to a generic structure, which consists of the following elements:

- Goal: The purpose of the attack strategy.
- Option(s): The possibilities to achieve this goal.
- Outcome: The result of the execution of the attack strategy.

6.1.1 Monetary Gain

The attack strategy of monetary gain refers to options that allow the mobile botnet to impact its victims financially. The goal is thus to perform activities on the victims' infected mobile devices (mobile bots) for monetary purposes. A possible attack option to achieve this goal is to command the mobile bots to either send large quantities of

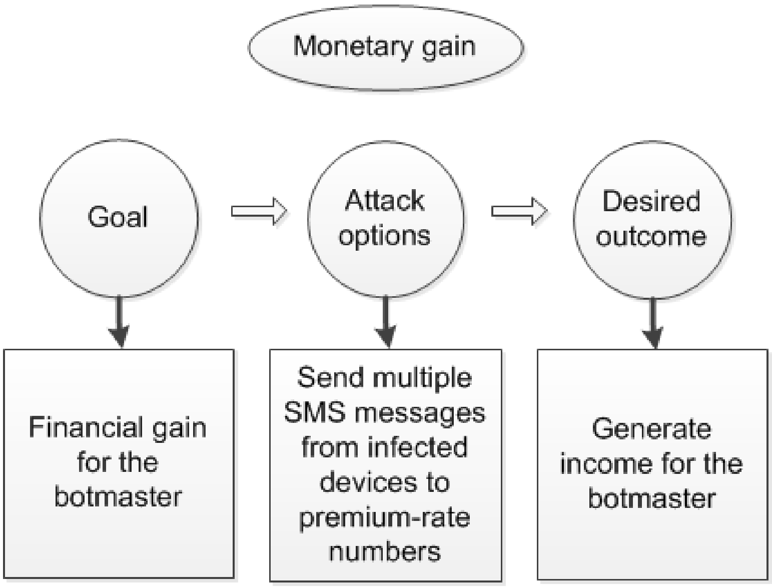

Monetary gain
Goal
Attack options
Desired outcome
Financial gain for the botmaster
Send multiple SMS messages from infected devices to premium-rate numbers
Generate income for the botmaster

Figure 6.2: Visual layout of the attack strategy of monetary gain

SMS messages or continuously call premium-rate numbers. Premium-rate numbers are telephone numbers reserved for particular services and are charged at a higher rate than normal phone calls. By sending SMS messages or making phone calls at regular intervals to such numbers, the mobile botnet can generate substantial amounts of money for the botmaster.

An additional option to achieve the goal of this particular attack strategy is to exploit SMS voting. SMS voting is a process that allows people the possibility to vote for a particular contest or poll by means of SMS messages. In order to alter a vote, the botmaster can command the mobile bots to send SMS messages at regular intervals and so increase the chances of a specific contender. The contender to be voted for can either be the botmaster or a person paying the botmaster to perform this task.

The outcome of this attack strategy is the collection of money for the botmaster and the generation of high bills for the victims with the infected mobile devices.

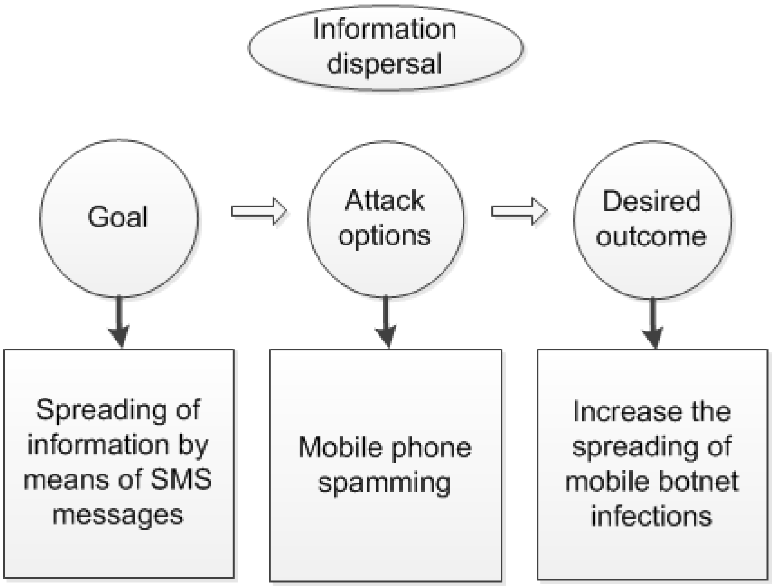

Information dispersal
Goal
Attack options
Desired outcome
Spreading of information by means of SMS messages
Mobile phone spamming
Increase the spreading of mobile botnet infections

Figure 6.3: Visual layout of the attack strategy of information dispersal

6.1.2 Information Dispersal

Information dispersal is an attack strategy with attack options focused on spreading information or malicious content. The ultimate goal behind this attack strategy is to send large quantities of information or malicious content via SMS messages. The possible option to achieve this goal is mobile phone spamming.

Mobile phone spamming refers to the act of sending large quantities of unwanted SMS messages, which can possibly contain links to phishing sites or silently download unwanted malware and so also increase mobile botnet infections. Besides creating great annoyance for end-users and disrupting availability, mobile phone spamming can also lead to fraud, financial and identity theft or act as a malware delivery service (Delany et al. 2012).

The outcome of deploying a mobile botnet to perform mobile phone spamming is serious damage for the victims, including the possibility of fraud, theft and malware infections. Mobile phone spamming can lead to extremely high bills for the victims. The definite outcome for the botmaster by exploiting this attack option is the spreading of mobile botnet malware.

6.1.3 Information Reaping

The attack strategy of information reaping refers strictly to the collection of information. Such information can include anything such as text, audio or even video. The goal of this attack strategy is to collect information, either directly from the mobile device or by exploiting built-in functionalities, for personal gain. The possible attack options to achieve this goal are exploiting vulnerabilities on the mobile device that allow the botmaster to collect personal or device information, or exploiting functionalities of the mobile device and recording information of the surrounding area. Personal information that a mobile bot can steal from a mobile device can include, but is not limited to, phone number, messages, contacts, photos, emails, missed and received calls. Information collected from the device can include, but is not limited to, IMEI number, IMSI number and the device model. All of this information is specific to each mobile device and permits the botmaster to uniquely identify each mobile bot within the mobile botnet.

With the functionalities of mobile devices constantly evolving, mobile devices have become a tool that offers much more than just telephone services. Mobile devices are now equipped with a vast range of possibilities, including audio recording, digital cameras, video recording and global positioning system (GPS) navigation. These functionalities not only improve the mobile phone experience for users, but also provide botmasters with new possibilities to collect information. These functionalities enable the botmaster to collect information of the area surrounding the mobile device. For example:

- Audio recording: enables the recording of conversations and meetings.
- Digital cameras: enable the taking of images of the area surrounding the mobile device.
- Video recording: enables the recording of sound and images.
- GPS: pinpointing the geographical location of a mobile device.

Stealing personal and device information provides the botmaster with the opportunity to clone the mobile device and then operate it in a normal way without incurring any of the cost. In addition, the cloned mobile device allows the botmaster to continuously

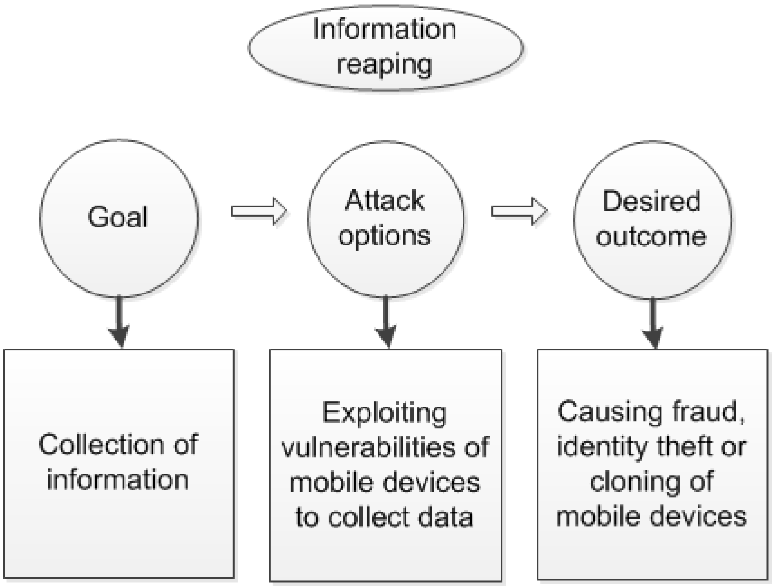

Information reaping
Goal
Attack options
Desired outcome
Collection of information
Exploiting vulnerabilities of mobile devices to collect data
Causing fraud, identity theft or cloning of mobile devices

Figure 6.4: Visual layout of the attack strategy of information reaping

steal new information (such as messages and calls) as it arrives on the device. Exploiting functionalities on the mobile device permits the botmaster to collect information of the surroundings without the requirement of physically being there. This provides the opportunity to collect information at locations where access is tightly restricted.

The attack strategy of information reaping can occur in two different formations: controlled and uncontrolled. Controlled information reaping refers to the possibility of only collecting information from mobile devices that are located at a specific location. This location is chosen by the botmaster and will be of interest to the person. With the attack strategy of controlled information reaping the quality, and not the quantity, of the information is important.

Uncontrolled information reaping allows the botmaster to collect information from any available mobile device, regardless of the location of the device. Thus the botmaster wants to collect as much information as possible and the quantity, not the quality, is important.

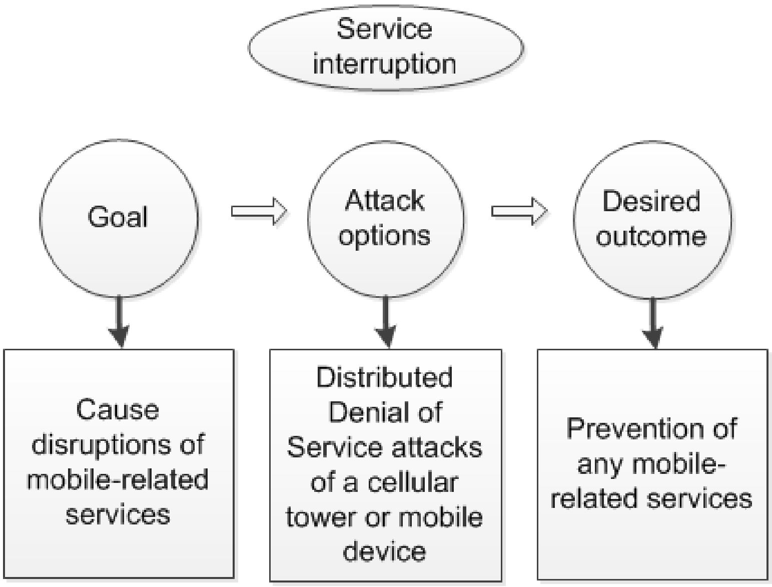

Service interruption
Goal
Attack options
Desired outcome
Cause disruptions of mobile-related services
Distributed Denial of Service attacks of a cellular tower or mobile device
Prevention of any mobile-related services

Figure 6.5: Visual layout of the attack strategy of service interruption

6.1.4 Service Interruption

The service interruption attack strategy refers to the disruption of services. Thus the goal behind this attack strategy is to cause a degree of disruption either within a cellular network or on a single mobile device. The first attack option to achieve disruptions is to cause a service interruption within the cellular network by means of a DDoS attack. An important component of any cellular network is the home location register (HLR) (Traynor et al. 2009). To be able to perform any cellular activity, the network assigns users to a certain HLR according to their phone numbers (Traynor et al. 2009). Since the continuous availability of HLRs is responsible for the functionality of a cellular network, HLRs easily become targets for attacks (Geng et al. 2012). The botmaster can deploy the mobile botnet to overload a specific HLR with great amounts of traffic and so cause a DDoS attack. The botmaster can also use a DDoS to target a single mobile device. This is achievable by commanding the mobile bots to target a specific mobile device and overload it with incoming traffic.

A DDoS attack on a specific HLR will block all the users who make use of that HLR and prevent any cellular activities. It will cause disruptions to occur over a very large geographical area. The outcome of a DDoS attack on a specific mobile device will block any traffic reaching that device, to great inconvenience of the user of the mobile device. The service interruption attack strategy can cause great levels of damage, both to the victims of the mobile bots and other uninfected parties.

6.2 Attack Orientation

Certain attack strategies require a certain attack orientation in order to execute successfully. The attack orientation refers to the focus of the attack strategy. There are two possible focuses: density or location. Attack orientation based on density constructs the focal point on a large mobile botnet regardless of geographical positioning. The larger the mobile botnet, the better it will serve the goal as set out by the selected attack strategy. The size of the mobile botnet will thus impact the overall efficiency of the eventual attack.

Attack orientation based on location constructs the focal point on a specific geographical location. The chosen location is of specific interest to the botmaster and the goal he/she wants to achieve. Such locations can possibly include a certain location within a building, such as an office of a senior employee, a meeting room or a cellular tower located closely to a certain place of interest. The size of the mobile botnet is of secondary importance as the position of the bots is more important but the size will impact the efficiency of the eventual attack.

The attack strategies based on their respective attack orientation can be categorised as follows:

Attack strategies with a focus on density:

- Information reaping - uncontrolled
- Information dispersal
- Monetary gain

Attack strategies with a focus on location:

- Information reaping - controlled
- Service interruption

6.3 Impact on the Process of Command Dissemination

The basic design and structure of the Hybrid Mobile Botnet enable the botmaster to apply any of the attack strategies as described in Section 6.1. The implementation of an attack strategy and its associated attack orientation do not have any impact on the execution of the propagation vectors, the design of the C&C channels or the layout of the topology. The choice of an attack strategy does, however, impact the process of command dissemination, since the associated attack orientation will determine whether all or only a selection of the mobile bots must receive the command.

In the case of an attack orientation based on location, it is unnecessary to disseminate the command to mobile bots that are currently not near the identified location since they will be unable to participate in the attack. Thus attack strategies consisting of an attack orientation based on location will execute the following updated process of command dissemination for a cluster head bot:

1. The botmaster sends the SMS messages containing the command and the GPS coordinates of the location that is identified to receive the attack.
2. The cluster head bot retrieves and intercepts the SMS message as sent from the botmaster or other cluster head bot.
3. The cluster head bot extracts the command from the SMS message.
4. It extracts the GPS coordinates of the identified location.
5. It connects to the control server via the HTTP C&C channel.
6. The cluster head bot verifies whether it is closely located to the required location.
7. If the cluster head bot is indeed closely located to the required location, it retrieves the list of randomly selected cluster head bots to which the command must be forwarded and updates the locally stored command list accordingly. The selected cluster head bots are closely located to the identified location.

8. The cluster head bot retrieves all the bot IDs of the mobile bots falling within this specific cluster botnet and updates the locally stored bot list accordingly.

9. It enables and initialises Bluetooth connections with the receiver bots during the active period.

10. When all of the mobile bots in the cluster botnet have received the command, the cluster head bot returns to the role of an inactive receiver bot.

The steps described above will ensure that only mobile bots that come within range of the identified location will receive the command. All of the other mobile bots will continue acting as inactive receiver bots and will not participate in the eventual attack.

With attack strategies using an attack orientation based on density, the geographical positioning of the mobile bots is not important. The attack only requires availability of as many mobile bots as possible. Therefore it is necessary that the command reach all of the mobile bots. This will require the process of command dissemination as described in Section 5.4.2 without any alteration.

Adapting the process of command dissemination to fit the required attack orientation of a selected attack strategy will ensure the efficient use of the mobile botnet to achieve the required goal.

6.4 Conclusion

This chapter introduced a collection of attack strategies that the Hybrid Mobile Botnet and all other mobile botnet designs in general can follow. The available attack strategies are monetary gain, information dispersal, information reaping and service interruption, and each provides the necessary tools that the botmaster requires to achieve his/her goal. All possible attacks that mobile botnets can launch are collected in these four distinct categories and it is up to the botmaster to select the appropriate attack strategy. Each individual attack strategy can be influenced by the attack orientation, which refers to the focal point of the attack strategy. This focal point can either be based on density or location. The implementation of any of these four attack strategies does not have any impact on the design of the Hybrid Mobile Botnet and only an additional element is required when the selected attack orientation is based on location. With the design of the Hybrid Mobile Botnet complete and the possible attacks identified, the next chapter focuses on the implementation of a prototype of the Hybrid Mobile Botnet to show the possibility of such a mobile botnet in a real environment.

Chapter 7

Prototype of the Hybrid Mobile Botnet

A new mobile botnet design, called the Hybrid Mobile Botnet, was introduced in Chapter 5. The purpose of this new design is to explore the efficiency of a hybrid C&C structure and to illustrate that current mobile technology exhibits all the required capabilities needed to develop and support hybrid mobile botnets. To determine the effectiveness of the Hybrid Mobile Botnet, a prototype built according to the Hybrid Mobile Botnet model as explained in Chapter 5 is discussed. This prototype is then measured against the following characteristics to determine if a stealthy, cost-effective and robust design was achieved: no single point of failure must exist in the topology, low monetary cost for command dissemination, limited network activities and low battery consumption per mobile bot.

The goal of this prototype was to evaluate the effectiveness of the Hybrid Mobile Botnet on real devices, measure the above characteristics and obtain knowledge from the experience that could aid the development of potential security measures against mobile botnets.

The rest of the chapter is structured as follows: the design of the prototype is described in Section 7.1 and the reasons for many of the design decisions made are explained. The execution of the prototype is discussed in Section 7.2, while Section 7.3 provides a discussion of the evaluation of the prototype. The strengths and weaknesses of the Hybrid Mobile Botnet are identified in Section 7.4 and Section 7.5 concludes the chapter.

7.1 Prototype Design

For the prototype to function successfully, multiple components were required. The components required included a small collection of mobile devices and a web server to host the control server. The next sections focus on the decisions made regarding the equipment used during the prototype design and the internal execution structure of the prototype.

7.1.1 Equipment

The prototype was deployed using a small collection of mobile devices, each infected with the malicious bot program. The bot program was specifically developed for the Android OS and can run on Android version 2.3.3 and above. There were multiple reasons behind the selection of the Android OS as the development platform. Firstly, as of the final quarter of 2012 the Android OS led the market share with 69.7% of smartphone sales, well ahead of Apple's iOS (20.9%) and Research in Motion (3.5%) (Gartner 2013), making it currently the most popular mobile OS for mobile devices. Secondly, the openness of the design and the ease of customising are the aspects of the Android OS that are allowing this OS to lead the field of mobile OSs. Thirdly, besides the popularity, the Android OS also allows any user to create, develop and upload applications to Google's Play Store.

The following devices were used to host the malicious bot program: Samsung Galaxy Pocket, Samsung Galaxy S2 and Google's Nexus 7 tablet. A wide variety of devices were chosen to show that the Hybrid Mobile Botnet is not limited to a specific type of mobile device but is capable of executing on both smartphones and tablet computers. For the successful execution of the prototype, each device was infected with exactly the same malicious bot program. During the execution of the prototype the battery consumption, data consumption and the analysis of the installed anti-virus applications were evaluated.

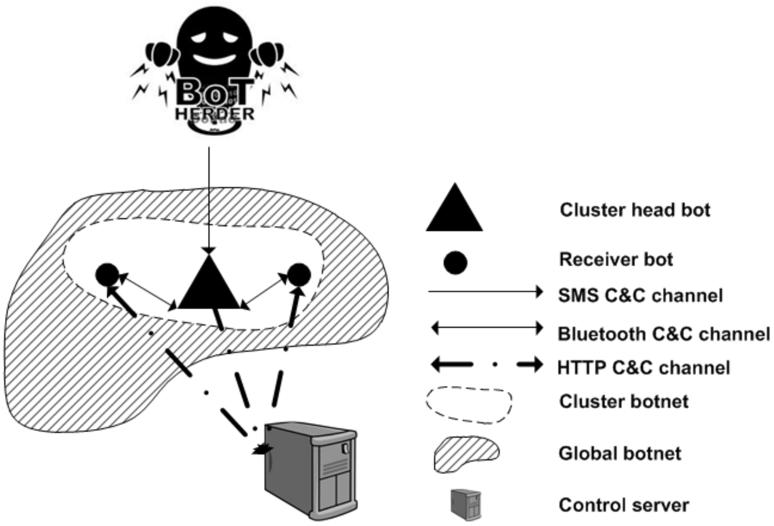


Figure 7.1: The topology of the Hybrid Mobile Botnet prototype

The internal interaction and execution between these devices as they operate in the prototype are further explained below.

7.1.2 Internal Design and Execution

Three Android devices and the control server were used to support the execution of the prototype. The three Android devices selected were sufficient to simulate the behaviour of the Hybrid Mobile Botnet since one device could act as the cluster head bot while the other assumed the role of receiver bots. The prototype supports the construction of all three C&C channels, namely SMS, Bluetooth and HTTP. Due to the size of the prototype, only a single propagation vector was required, namely propagation by an infected application. The control server was hosted on the http://twiggie.cuchulain.co.za domain. The topology and visualisation of the Hybrid Mobile Botnet prototype is shown in Figure 7.1, displaying only three mobile bots and the control server.

Each mobile device was a critical component of the prototype and assumed multiple roles while executing. At the start of the execution of the prototype, all three mobile devices assumed the role of an inactive receiver bot. As the execution of the prototype progressed, the roles of the mobile bots changed.

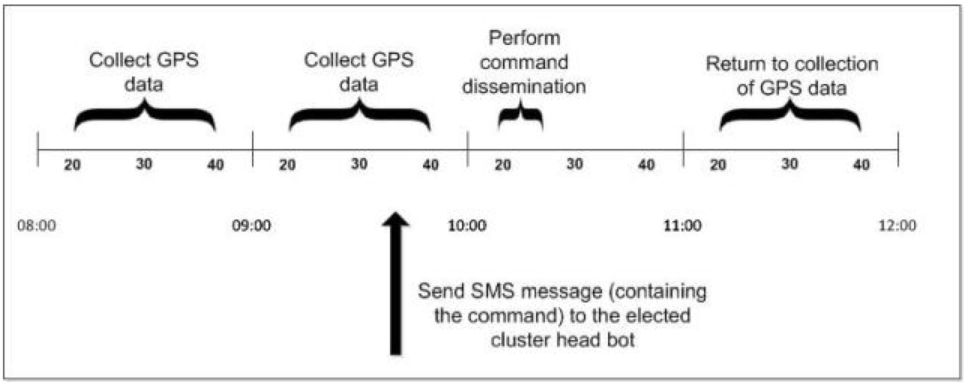

Collect GPS data
Collect GPS data
Perform command dissemination
Return to collection of GPS data
20
30
40
20
30
40
20
30
40
20
30
40
08:00
09:00
10:00
11:00
12:00
Send SMS message (containing the command) to the elected cluster head bot

Figure 7.2: Timeline of execution of the prototype

Apart from the responsibility each mobile device had towards the prototype, additional evaluation of the Hybrid Mobile Botnet was also required. During the execution of the prototype, each device was responsible for evaluating certain aspects of the mobile botnet, namely battery consumption, data consumption and anti-virus analysis. None of these additional activities had an influence on the execution of the prototype.

The prototype was designed to execute at hourly intervals to simplify evaluation of performance and execution. The first two hours focused on establishing the structure of the Hybrid Mobile Botnet by means of GPS data while command dissemination was performed during the third hour after receiving the command via an SMS message during the previous hour. The final hour again focused on collecting GPS data to determine the structure of the Hybrid Mobile Botnet. The proposed timeline of execution for the prototype is shown in Figure 7.2.

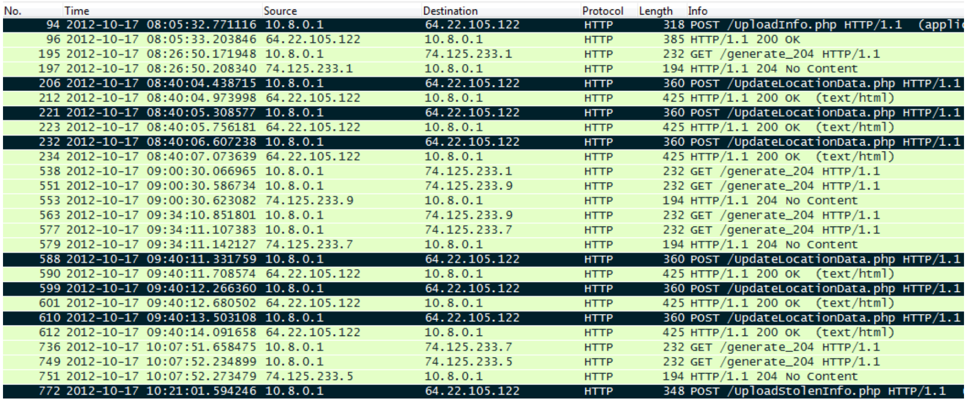
With every hour interval, the time period between twenty past and twenty to, for a total period of 40 minutes, represents the period of low mobility (see Section 5.2.2) and is the time when the mobile bots were active. For the first two hours of execution, the mobile devices acted as inactive receiver bots and collected GPS data at ten-minute intervals. These intervals are referred to as the active periods and were also used during command dissemination. During the last active period (40-minute interval), the collected GPS data of the current period of low mobility was uploaded to the control server. The response from the control server included the Bluetooth MAC address of the elected cluster head bot and the command dissemination flag. The command dissemination flag indicated whether the following period of low mobility would perform command dissemination or continue with the collection of GPS data. Only the botmaster can start the process of command dissemination by sending the command via an SMS message to the cluster head bot. If the response returned from the control server was true for the start of command dissemination, then during the following period of low mobility the mobile bots would only focus on sending or receiving commands. Thus the mobile bot receiving the command assumed the role of a cluster head bot while the other mobile bots assumed the role of an active receiver bot. When the mobile bots completed the process of command dissemination and the execution of the received command, all of the mobile bots returned to the role of an inactive receiver bot and continued with the collection of GPS data.

7.2 Prototype Execution

To successfully track the execution of the prototype, the tPacketCapture[1] application was installed on the Nexus 7 tablet (the only mobile device to support the tPacketCapture application). This application performs packet capturing without the requirement of running the device as root. The captured data were saved as a packet capture (PCAP) file and can be viewed using Wireshark. Using this application made it possible to capture the communication occurring between a mobile bot and the control server. The Nexus 7 tablet monitored the HTTP traffic that occurred across a Wi-Fi connection using this application.

In addition, the messages were logged in Eclipse, the platform used to design and develop the Hybrid Mobile Botnet. In order to capture all the messages, one of the mobile devices was connected to Eclipse via a universal serial bus (USB) connection. To allow the USB connection from the computer to the mobile device to capture the log messages, USB debugging was enabled on the mobile device. This setting is considered insecure and only used for development purposes when messages need to be log. The messages captured by Eclipse were used instead of the messages logged by the control server as a complete inventory of mobile device activities was logged.

[1] https://play.google.com/store/apps/details?id=jp.co.taosoftware.android.packetcapture&hl=en



No.	Time	Source	Destination	Protocol	Length	Info
94	2012-10-17 08:05:32.771116	10.8.0.1	64.22.105.122	HTTP	318	POST /UploadInfo.php HTTP/1.1 (appli
96	2012-10-17 08:05:33.203846	64.22.105.122	10.8.0.1	HTTP	385	HTTP/1.1 200 OK
195	2012-10-17 08:26:50.171948	10.8.0.1	74.125.233.1	HTTP	232	GET /generate_204 HTTP/1.1
197	2012-10-17 08:26:50.208340	74.125.233.1	10.8.0.1	HTTP	194	HTTP/1.1 204 No Content
206	2012-10-17 08:40:04.438715	10.8.0.1	64.22.105.122	HTTP	360	POST /UpdateLocationData.php HTTP/1.1
212	2012-10-17 08:40:04.973998	64.22.105.122	10.8.0.1	HTTP	425	HTTP/1.1 200 OK (text/html)
221	2012-10-17 08:40:05.308577	10.8.0.1	64.22.105.122	HTTP	360	POST /UpdateLocationData.php HTTP/1.1
223	2012-10-17 08:40:05.756181	64.22.105.122	10.8.0.1	HTTP	425	HTTP/1.1 200 OK (text/html)
232	2012-10-17 08:40:06.607238	10.8.0.1	64.22.105.122	HTTP	360	POST /UpdateLocationData.php HTTP/1.1
234	2012-10-17 08:40:07.073639	64.22.105.122	10.8.0.1	HTTP	425	HTTP/1.1 200 OK (text/html)
538	2012-10-17 09:00:30.066965	10.8.0.1	74.125.233.1	HTTP	232	GET /generate_204 HTTP/1.1
551	2012-10-17 09:00:30.586734	10.8.0.1	74.125.233.9	HTTP	232	GET /generate_204 HTTP/1.1
553	2012-10-17 09:00:30.623082	74.125.233.9	10.8.0.1	HTTP	194	HTTP/1.1 204 No Content
563	2012-10-17 09:34:10.851801	10.8.0.1	74.125.233.9	HTTP	232	GET /generate_204 HTTP/1.1
577	2012-10-17 09:34:11.107383	10.8.0.1	74.125.233.7	HTTP	232	GET /generate_204 HTTP/1.1
579	2012-10-17 09:34:11.142127	74.125.233.7	10.8.0.1	HTTP	194	HTTP/1.1 204 No Content
588	2012-10-17 09:40:11.331759	10.8.0.1	64.22.105.122	HTTP	360	POST /UpdateLocationData.php HTTP/1.1
590	2012-10-17 09:40:11.708574	64.22.105.122	10.8.0.1	HTTP	425	HTTP/1.1 200 OK (text/html)
599	2012-10-17 09:40:12.266360	10.8.0.1	64.22.105.122	HTTP	360	POST /UpdateLocationData.php HTTP/1.1
601	2012-10-17 09:40:12.680502	64.22.105.122	10.8.0.1	HTTP	425	HTTP/1.1 200 OK (text/html)
610	2012-10-17 09:40:13.503108	10.8.0.1	64.22.105.122	HTTP	360	POST /UpdateLocationData.php HTTP/1.1
612	2012-10-17 09:40:14.091658	64.22.105.122	10.8.0.1	HTTP	425	HTTP/1.1 200 OK (text/html)
736	2012-10-17 10:07:51.658475	10.8.0.1	74.125.233.7	HTTP	232	GET /generate_204 HTTP/1.1
749	2012-10-17 10:07:52.234899	10.8.0.1	74.125.233.5	HTTP	232	GET /generate_204 HTTP/1.1
751	2012-10-17 10:07:52.273479	74.125.233.5	10.8.0.1	HTTP	194	HTTP/1.1 204 No Content
772	2012-10-17 10:21:01.594246	10.8.0.1	64.22.105.122	HTTP	348	POST /UploadStolenInfo.php HTTP/1.1

Figure 7.3: Captured data of a mobile bot's network activities

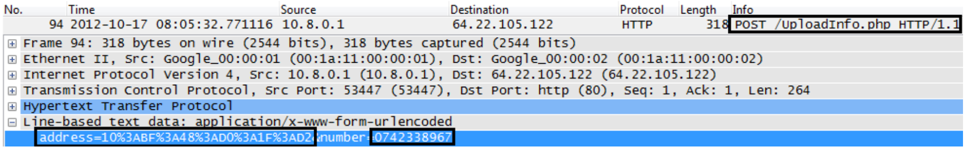
To determine whether the execution of the prototype was successful, the mobile bots had to collect information form the mobile devices. Thus the chosen attack strategy for the prototype was uncontrolled information reaping (see Section 6.1.3).

Upon receiving the command, these mobile bots had to collect the IMEI and IMSI numbers of their respective devices and forward the information to the control server. For the purpose of the executions, the IMEI, IMSI and mobile phone numbers were simulated on the Nexus 7 tablet since this particular tablet was not equipped with any mobile network connectivity.

The execution of the prototype was broken into two separate demonstrations. The first demonstration shows the command dissemination and execution of the prototype. The second demonstration focuses on the execution of the Bluetooth C&C channel.

7.2.1 Demonstration 1: Command Dissemination and Execution

The data captured during the execution of the prototype is displayed in Figure 7.3. The time when the packets were captured supports the expected times as displayed in the timeline (see Figure 7.2). The lines highlighted in black are the captured packets (formatted units of data) that relate directly to the execution of the prototype, and will be examined more closely in the remainder of this section. In order to visualise the communication taking place between the mobile bot and the control server, no encryption



```
No.     Time                         Source        Destination      Protocol  Length  Info
     94 2012-10-17 08:05:32.771116   10.8.0.1      64.22.105.122    HTTP         318  POST /UploadInfo.php HTTP/1.1
⊞ Frame 94: 318 bytes on wire (2544 bits), 318 bytes captured (2544 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 10.8.0.1 (10.8.0.1), Dst: 64.22.105.122 (64.22.105.122)
⊞ Transmission Control Protocol, Src Port: 53447 (53447), Dst Port: http (80), Seq: 1, Ack: 1, Len: 264
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: application/x-www-form-urlencoded
    address=10%3ABF%3A48%3AD0%3A1F%3AD2&number=0742338967
```

Figure 7.4: Captured data sent via the UploadInfo.php script

was used during the execution of the prototype.

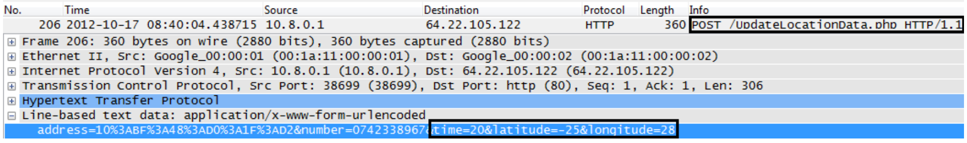
The first connection made to the control server is via the UploadInfo.php script (see Figure 7.4) to upload the information collected during the execution of the static activities. The static activities refers to the collection of the mobile phone number (074 233 8967) and the Bluetooth MAC address (10:BF:48:AD:1F:D2) of the mobile device. The mobile bot was then responsible for forwarding the collect information to the control server. The Hybrid Mobile Botnet requires this information to be able to uniquely identify each mobile bot.

Multiple connections occurred between the mobile bot and the control server during 08:40:04 and 08:40:06 via the UploadLocationData.php script. During each connection the mobile bot uploaded the collected GPS data associated with a specific active period (see Figures 7.5, 7.6 and 7.7).

After the last connection (at the 40-minute interval) the control server responded back with the Bluetooth MAC address of the elected cluster head bot (28:98:7B:3A:79:8A) and the command dissemination flag (currently set to false). The false property of the flag meant that the following period of low mobility would continue with the collection of GPS data as the mobile bot continued as an inactive receiver bot (see Figure 7.8).

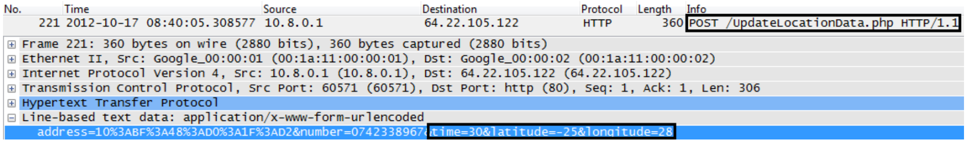
The next connections occurred between 09:40:11 and 09:40:13, during which the collected GPS data was once again uploaded to the control server. The response received from the control server changed, however (see Figure 7.9). The command dissemination flag was now set to true, meaning that the botmaster sent the command via an SMS message to the elected cluster head bot somewhere between 08:40 and 09:40. Thus the remaining inactive receiver bots would now assume the role of active receiver bots and during the next period of low mobility would perform command dissemination and execution.

Figure 7.10 reveals that the mobile bot successfully received and executed the com-



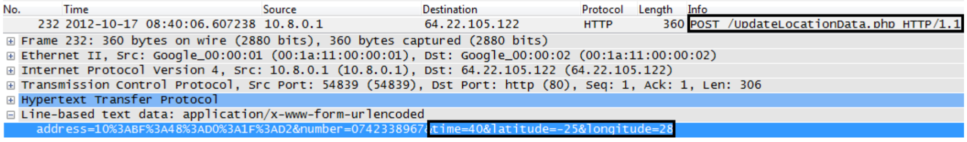
```
No.        Time                        Source              Destination          Protocol  Length  Info
       206 2012-10-17 08:40:04.438715 10.8.0.1            64.22.105.122        HTTP         360 POST /UpdateLocationData.php HTTP/1.1
⊞ Frame 206: 360 bytes on wire (2880 bits), 360 bytes captured (2880 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 10.8.0.1 (10.8.0.1), Dst: 64.22.105.122 (64.22.105.122)
⊞ Transmission Control Protocol, Src Port: 38699 (38699), Dst Port: http (80), Seq: 1, Ack: 1, Len: 306
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: application/x-www-form-urlencoded
    address=10%3ABF%3A48%3AD0%3A1F%3AD2&number=0742338967&time=20&latitude=-25&longitude=28
```

Figure 7.5: Collected GPS data for the 20-minute interval



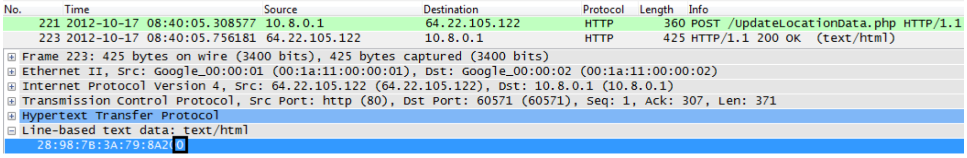
```
No.        Time                        Source              Destination          Protocol  Length  Info
       221 2012-10-17 08:40:05.308577 10.8.0.1            64.22.105.122        HTTP         360 POST /UpdateLocationData.php HTTP/1.1
⊞ Frame 221: 360 bytes on wire (2880 bits), 360 bytes captured (2880 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 10.8.0.1 (10.8.0.1), Dst: 64.22.105.122 (64.22.105.122)
⊞ Transmission Control Protocol, Src Port: 60571 (60571), Dst Port: http (80), Seq: 1, Ack: 1, Len: 306
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: application/x-www-form-urlencoded
    address=10%3ABF%3A48%3AD0%3A1F%3AD2&number=0742338967&time=30&latitude=-25&longitude=28
```

Figure 7.6: Collected GPS data for the 30-minute interval



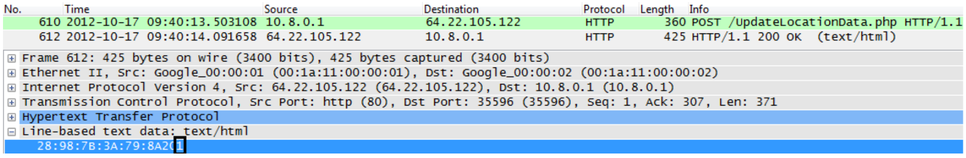
```
No.        Time                        Source              Destination          Protocol  Length  Info
       232 2012-10-17 08:40:06.607238 10.8.0.1            64.22.105.122        HTTP         360 POST /UpdateLocationData.php HTTP/1.1
⊞ Frame 232: 360 bytes on wire (2880 bits), 360 bytes captured (2880 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 10.8.0.1 (10.8.0.1), Dst: 64.22.105.122 (64.22.105.122)
⊞ Transmission Control Protocol, Src Port: 54839 (54839), Dst Port: http (80), Seq: 1, Ack: 1, Len: 306
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: application/x-www-form-urlencoded
    address=10%3ABF%3A48%3AD0%3A1F%3AD2&number=0742338967&time=40&latitude=-25&longitude=28
```

Figure 7.7: Collected GPS data for the 40-minute interval



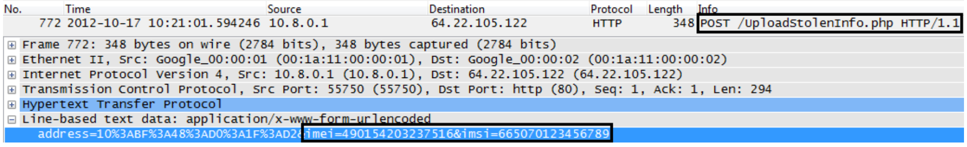
```
No.        Time                        Source              Destination          Protocol  Length  Info
       221 2012-10-17 08:40:05.308577 10.8.0.1            64.22.105.122        HTTP         360 POST /UpdateLocationData.php HTTP/1.1
       223 2012-10-17 08:40:05.756181 64.22.105.122       10.8.0.1             HTTP         425 HTTP/1.1 200 OK  (text/html)
⊞ Frame 223: 425 bytes on wire (3400 bits), 425 bytes captured (3400 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 64.22.105.122 (64.22.105.122), Dst: 10.8.0.1 (10.8.0.1)
⊞ Transmission Control Protocol, Src Port: http (80), Dst Port: 60571 (60571), Seq: 1, Ack: 307, Len: 371
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: text/html
    28:98:7B:3A:79:8A200
```

Figure 7.8: First response from the control server



```
No.        Time                        Source              Destination          Protocol  Length  Info
       610 2012-10-17 09:40:13.503108 10.8.0.1            64.22.105.122        HTTP         360 POST /UpdateLocationData.php HTTP/1.1
       612 2012-10-17 09:40:14.091658 64.22.105.122       10.8.0.1             HTTP         425 HTTP/1.1 200 OK  (text/html)
⊞ Frame 612: 425 bytes on wire (3400 bits), 425 bytes captured (3400 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 64.22.105.122 (64.22.105.122), Dst: 10.8.0.1 (10.8.0.1)
⊞ Transmission Control Protocol, Src Port: http (80), Dst Port: 35596 (35596), Seq: 1, Ack: 307, Len: 371
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: text/html
    28:98:7B:3A:79:8A201
```

Figure 7.9: Second response from the control server



```
No.        Time                        Source              Destination          Protocol  Length  Info
       772 2012-10-17 10:21:01.594246 10.8.0.1            64.22.105.122        HTTP         348 POST /UploadStolenInfo.php HTTP/1.1
⊞ Frame 772: 348 bytes on wire (2784 bits), 348 bytes captured (2784 bits)
⊞ Ethernet II, Src: Google_00:00:01 (00:1a:11:00:00:01), Dst: Google_00:00:02 (00:1a:11:00:00:02)
⊞ Internet Protocol Version 4, Src: 10.8.0.1 (10.8.0.1), Dst: 64.22.105.122 (64.22.105.122)
⊞ Transmission Control Protocol, Src Port: 55750 (55750), Dst Port: http (80), Seq: 1, Ack: 1, Len: 294
⊞ Hypertext Transfer Protocol
⊟ Line-based text data: application/x-www-form-urlencoded
    address=10%3ABF%3A48%3AD0%3A1F%3AD2&imei=490154203237516&imsi=665070123456789
```

Figure 7.10: Stolen IMEI and IMSI numbers

mand since it located the IMEI (490154203237516) and the IMSI (665070123456789) numbers. The mobile bot then forwarded these numbers to the control server via the UploadStolenInfo.php script.

This step-by-step analysis of the captured packets shows that the prototype executed correctly, without any errors. This prototype thus illustrates that the current mobile technology exhibits all the capabilities required for developing and supporting a hybrid mobile botnet. The following section will focus on the Bluetooth C&C channel and the steps taken to circumvent the Bluetooth security features.

7.2.2 Demonstration 2: Execution of the Bluetooth C&C Channel

This section provides an in-depth look into the initialisation of the Bluetooth C&C channel, construction of communication across the channel and measurement of the execution. The researcher decided to focus more closely on this particular C&C channel since it required an extensive development phase to overcome all of the security features associated with Bluetooth and move past previous research by Singh et al. (2010) and Hua and Sakurai (2012), which only focuses on simulations to explore the potential of Bluetooth C&C channels. Since the development of Bluetooth technology, vulnerabilities associated with the technology have continuously been discovered. Many of these vulnerabilities were not thought to be significant problems until the adoption of Bluetooth technology by mobile devices (Barnes 2002).

Current security threats targeting Bluetooth technology are due to vulnerabilities allowing eavesdropping and impersonation. Eavesdropping allows an attacker to ‘listen’ to messages being exchanged during the pairing of devices (Jakobsson and Wetzel 2001). This is possible if there is no encryption on the application layer or if the attacker is able to impersonate a device (Jakobsson and Wetzel 2001). Impersonation occurs when an attacker poses as a legitimate Bluetooth device, allowing access to unauthorised data.

These vulnerabilities called for an improvement of Bluetooth security. One of the steps taken by Bluetooth developers to improve the security associated with Bluetooth connections is including a process called pairing. The process of pairing refers to a trusted relationship between two mobile devices which are formed by secret codes, also known as pins (Minar and Tarique 2012). The purpose of this process is to create a common link key that will allow for secure communication between devices (Gehrmann and Nyberg 2001). This process often involves user interaction where the users are responsible for confirming the identity of the mobile devices. This level of user interaction also increases the security surrounding Bluetooth connections and can prevent unauthorised users from misusing Bluetooth technology.

The simple secure pairing (SSP) protocol has been included in the Bluetooth Core specification since version 2.1 (Haataja and Toivanen 2008) and was used by all of the mobile devices included in this prototype. This protocol specifies the necessary steps for two Bluetooth devices to establish a shared common link key for subsequent secure communication (Phan and Mingard 2012). Most mobile devices employ SSP by using numeric comparison. With numeric comparison, a 6-digit number is displayed on both mobile devices and it is the responsibility of the users to compare these numbers on both devices. If the numbers are identical, the user selects 'Yes' and the pairing can proceed. Once the mobile devices has successfully completed the pairing process, a bond is established between the devices and communication can proceed.

7.2.2.1 Initialisation of the Bluetooth C&C channel

For the Bluetooth C&C channel of the Hybrid Mobile Botnet to succeed, the botmaster had to simulate the pairing process and exclude any user interaction.

The following features were thus required in order to establish a stealthy but secure Bluetooth C&C channel:

- Each receiver bot must know the Bluetooth MAC address of the cluster head bot while the cluster head bot must know the Bluetooth MAC addresses of all the receiver bots.

- No user involvement must be required (the user must not be requested to complete a pairing request).

- No traces left must be left behind on either of the mobile devices due to the pairing request (the mobile bots must remove the bond when the process of command dissemination is complete).

- The mobile devices must not be required to be in discoverable mode when establishing the Bluetooth C&C channel, which is possible by having access to the Bluetooth MAC addresses.

The development platform for the Bluetooth C&C channel was the Android OS (see Section 3.2.1). To create this channel on Android devices, additional private Bluetooth APIs (Application Programming Interfaces) were required: IBluetooth.aidl and IBluetoothCallback.aidl. These private APIs allow for direct manipulation of specific Bluetooth functionalities, including the pairing process.

To establish the Bluetooth C&C channel, the pairing process must be automated, which requires the participation of two devices. The cluster head bot will initialise the Bluetooth connection and perform the data transfer, while the receiver bot will accept the Bluetooth connection and receive the data. The first step of initialising the Bluetooth C&C channel is to determine if both of the participating devices have a Bluetooth adapter. If both devices have access to a Bluetooth adapter, the devices enable the Bluetooth. To ensure that a successful Bluetooth C&C is established, the cluster head bot initiates the pairing process. As soon as the cluster head bot comes within range of the receiver bot, the pairing process begins. For successful pairing, the Bluetooth of the receiver bot must already be enabled before the cluster head bot comes within range or enables its own Bluetooth. Both devices set the pin that will be used during this process. For the purpose of this demonstration the pin was set to 123456. The cluster head bot is then responsible for the following two steps: setting the pairing confirmation to true and cancelling the requirement for user input. The first step allows the cluster head bot to connect to the receiver bot without requiring a user to press the confirmation button on either of the devices. The second step cancels the requirement for user input, removes the pop-up dialogue from the screen and allows the pairing process to proceed without alerting the users of the devices. After successfully completing the pairing process, a bond will be created between the two devices.

By creating the bond between the two mobile devices, the Bluetooth C&C channel is constructed between the cluster head bot and the receiver bot. The cluster head will then be able to start the process of command dissemination. Below follows a detailed description of how communication occurs between two bonded mobile bots.

7.2.2.2 Construction of Communication across the Channel

Once the bond between the devices is established, the initialisation of the Bluetooth C&C channel is completed and communication across the channel can occur. To allow for data transfer, the radio frequency communication (RFCOMM) protocol is used. RFCOMM is a transport protocol that emulates serial connections and provides transport capabilities between Bluetooth-enabled devices (Panse and Kapoor 2012). The RFCOMM protocol acts as a cable replacement protocol by emulating the RS-232 control and data signals over the Bluetooth baseband (Bruno et al. 2002). The cluster head bot creates an insecure RFCOMM Bluetooth socket while the receiver bot is listening on a previously created insecure RFCOMM Bluetooth socket. Insecure RFCOMM sockets are used to avoid any additional authentication. When the devices established the sockets, a communication channel is created via the RFCOMM protocol and data transfer can proceed. The cluster head bot is responsible for sending the data to the receiver bot which will then respond to the received data. The data transfer can continue for as long as necessary but if the Bluetooth is left on indefinitely, the battery power of the two devices will quickly be consumed. Therefore the bond created between the devices must be removed. Removing the bond dismantles the Bluetooth C&C channel between the two devices, leaving no traces of the communication on either of the two devices. If required, the Bluetooth C&C can be constructed between the two devices again.

7.2.2.3 Measurement of Execution

To verify that the Bluetooth C&C channel was constructed successfully and that command dissemination did indeed take place during the execution of the prototype, the Ubertooth One development tool is used to capture the encoded packets. Ubertooth One is an open source 2.4 gigahertz wireless development platform, designed for Bluetooth experiments (Ubertooth 2010). At the time of performing this research, Ubertooth One was the best tool available for capturing Bluetooth packets.

The architecture of Ubertooth One is shown in Figure 7.11. Once plugged into a computer (see Figure 7.12), the small USB dongle is ready to start capturing Bluetooth packets (Ubertooth 2010).

Ubertooth One works closely with various wireless monitoring tools in order to vi-

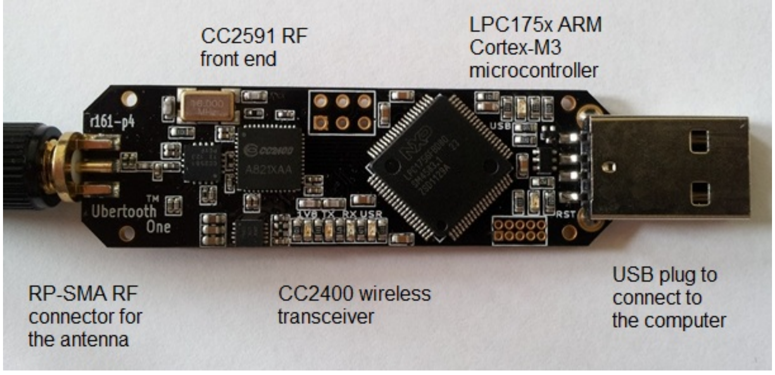


Figure 7.11: Architecture of the Ubertooth One development tool

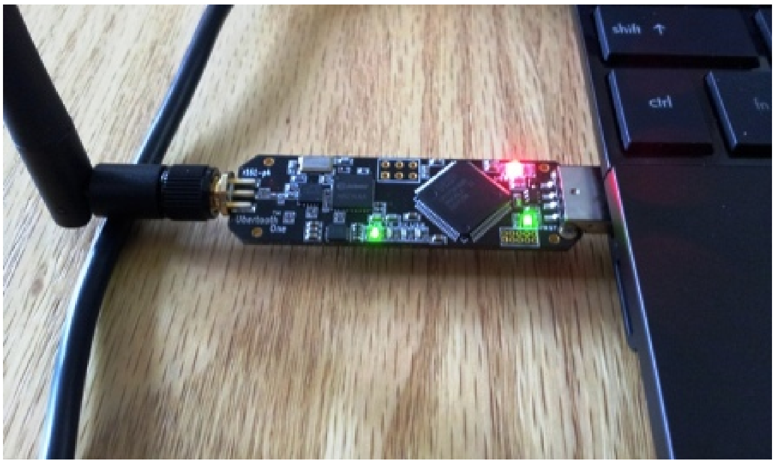
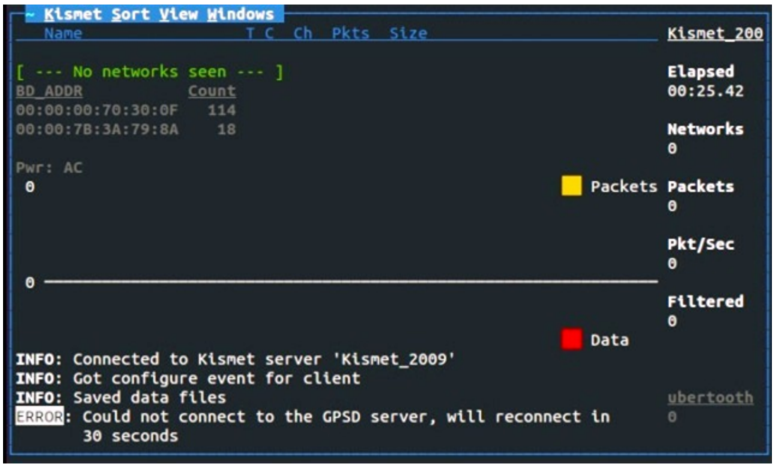


Figure 7.12: Ubertooth One plugged into a computer



```
  Name                      T C  Ch  Pkts  Size          Kismet_200

[ --- No networks seen --- ]                              Elapsed
BD_ADDR              Count                                00:25.42
00:00:00:70:30:0F     114
00:00:7B:3A:79:8A      18                                 Networks
                                                          0
Pwr: AC
 0                                          Packets       Packets
                                                          0

                                                          Pkt/Sec
                                                          0
 0 ───────────────────────────────────────────────────
                                                          Filtered
                                                          0
                                            Data
INFO: Connected to Kismet server 'Kismet_2009'
INFO: Got configure event for client
INFO: Saved data files                                    ubertooth
ERROR: Could not connect to the GPSD server, will reconnect in   0
       30 seconds
```

Figure 7.13: Screenshot of Kismet software showing the captured Bluetooth packets

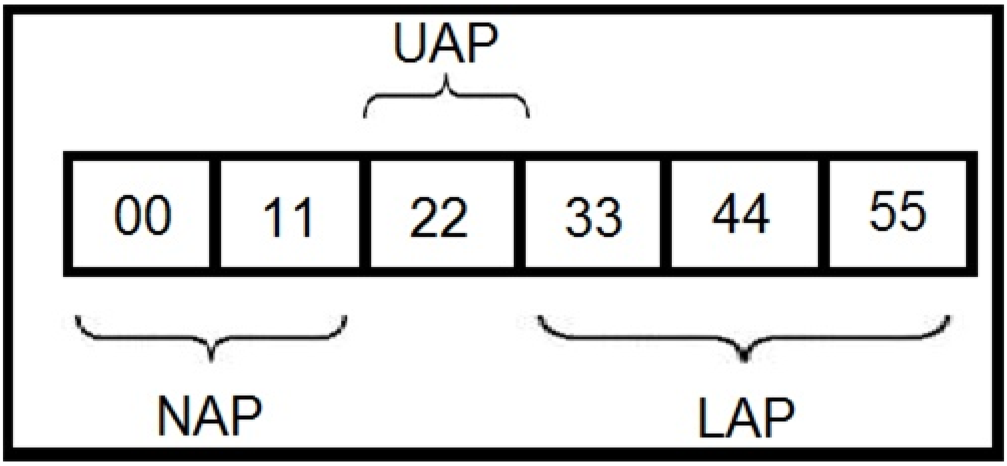


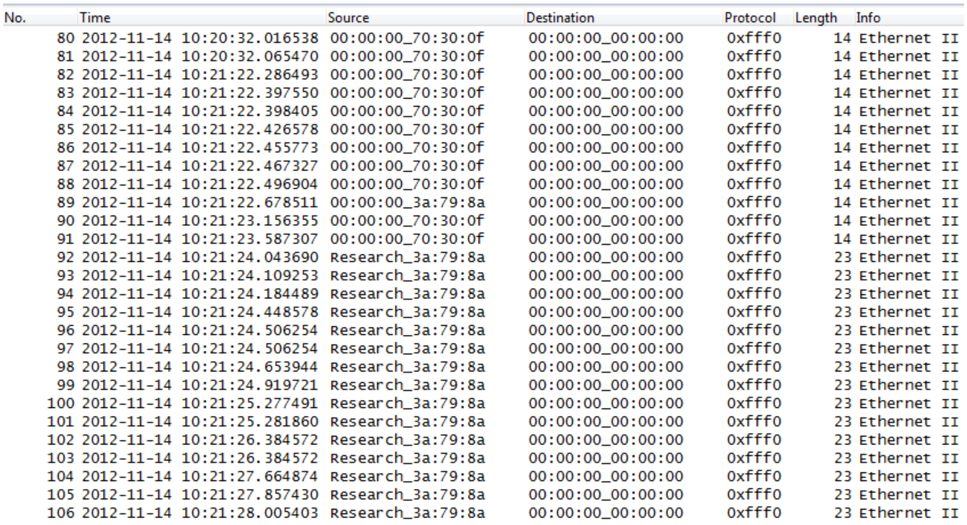
Figure 7.14: Bluetooth MAC address layout

sualise the captured packets. For the purpose of verifying the constructed Bluetooth C&C channel, Kismet software, as recommend by the Ubertooth One project, was used. Kismet is an 802.11 layer 2 wireless network detector, sniffer and intrusion detection system (Kismet n.d.). Both Ubertooth One and Kismet were active while the prototype was executing. The Bluetooth packets, as captured using Ubertooth One and Kismet, are shown in Figure 7.13.

Figure 7.13 also shows that the following Bluetooth MAC addresses were participating in the process of command dissemination: 00:00:00:70:30:0F and 00:00:7B:3A:79:8A. The Bluetooth MAC address (see Figure 7.14), which is similar to a MAC address of a computer, consists of 48 bits (Hager and Midkiff 2003).

The 48 bits can further be divided into a 16-bit non-significant address portion (NAP), an 8-bit upper address portion (UAP) and a 24-bit lower address portion (LAP) (Tariq et al. 2000). Both Ubertooth One and Kismet are only capable of identifying the LAP address portion, but in some instances can also identify the UAP by analysing the timing and additional characteristics of multiple packets (Ubertooth 2010).

According to the timeline of execution of the prototype (see Figure 7.2), the process of command dissemination was expected to take place during the following period: 10:20 and 10:25. To verify that command dissemination had indeed taken place during the above period and that the Bluetooth C&C channel was functioning correctly, the cap-



No.	Time	Source	Destination	Protocol	Length	Info
80	2012-11-14 10:20:32.016538	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
81	2012-11-14 10:20:32.065470	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
82	2012-11-14 10:21:22.286493	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
83	2012-11-14 10:21:22.397550	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
84	2012-11-14 10:21:22.398405	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
85	2012-11-14 10:21:22.426578	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
86	2012-11-14 10:21:22.455773	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
87	2012-11-14 10:21:22.467327	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
88	2012-11-14 10:21:22.496904	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
89	2012-11-14 10:21:22.678511	00:00:00_3a:79:8a	00:00:00_00:00:00	0xfff0	14	Ethernet II
90	2012-11-14 10:21:23.156355	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
91	2012-11-14 10:21:23.587307	00:00:00_70:30:0f	00:00:00_00:00:00	0xfff0	14	Ethernet II
92	2012-11-14 10:21:24.043690	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
93	2012-11-14 10:21:24.109253	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
94	2012-11-14 10:21:24.184489	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
95	2012-11-14 10:21:24.448578	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
96	2012-11-14 10:21:24.506254	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
97	2012-11-14 10:21:24.506254	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
98	2012-11-14 10:21:24.653944	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
99	2012-11-14 10:21:24.919721	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
100	2012-11-14 10:21:25.277491	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
101	2012-11-14 10:21:25.281860	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
102	2012-11-14 10:21:26.384572	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
103	2012-11-14 10:21:26.384572	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
104	2012-11-14 10:21:27.664874	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
105	2012-11-14 10:21:27.857430	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II
106	2012-11-14 10:21:28.005403	Research_3a:79:8a	00:00:00_00:00:00	0xfff0	23	Ethernet II

Figure 7.15: A snapshot of the captured packets in Wireshark

tured packets were analysed in Wireshark. A snapshot of the captured packets is shown in Figure 7.15.

Analysis of the captured packets shows that the devices participating in the Bluetooth C&C channel were active between 10:20 and 10:22. Most of the communication occurred between 10:21:22 and 10:21:27, showing that the construction, initialisation and communication occurred within one minute. This short set-up of the Bluetooth C&C channel makes this channel very efficient and feasible where quick data transfer is required.

This snapshot of the captured Bluetooth packets verifies that the Bluetooth C&C channel was constructed successfully and was active during the expected period. Furthermore, of the two mobile devices participating in the process of command dissemination it can be deduced that the mobile device with the LAP of 70:30:0F was indeed an active receiver bot since this bot activated its Bluetooth first. Thus the mobile device with the LAP of 3A:79:8A was the cluster head bot that forwarded the command to the active receiver bot.

The complete execution of the Bluetooth C&C channel is shown in Table 7.1, showing the time stamp, the participant and the activity performed by the participant. Time synchronisation was achieved between the mobile devices using network provided time.

Table 7.1: Log file captured during the execution of the Bluetooth C&C Channel

Timestamp	Participant	Activity
07-17 12:42:48.328	Receiver Bot	Received Bluetooth Event that can be ignored
07-17 12:43:08.063	Receiver Bot	Enable Bluetooth
07-17 12:43:12.570	Cluster Head Bot	Enable Bluetooth
07-17 12:43:14.228	Receiver Bot	Received Bluetooth Event that can be ignored
07-17 12:43:17.103	Receiver Bot	Start Bluetooth Connection
07-17 12:43:17.128	Receiver Bot	Wait for Pairing Request to Start
07-17 12:43:17.173	Receiver Bot	Received Bluetooth ON Event
07-17 12:43:20.367	Cluster Head Bot	Start Bluetooth Connection
07-17 12:43:20.367	Cluster Head Bot	Wait for Pairing Request to Start
07-17 12:43:25.164	Cluster Head Bot	Create Bond with Receiver
07-17 12:43:25.164	Cluster Head Bot	Set Pins
07-17 12:43:25.171	Cluster Head Bot	Set Pairing Confirmation to True
07-17 12:43:25.171	Cluster Head Bot	Cancel User Input for Pairing
07-17 12:43:25.783	Receiver Bot	Set Pins
07-17 12:43:26.898	Cluster Head Bot	Send Data
07-17 12:43:26.937	Cluster Head Bot	Response From Receiver: Data Received
07-17 12:43:27.906	Cluster Head Bot	Removed Paired Device

The use of the Ubertooth One tool and Kismet software allowed for the analysis of the Bluetooth C&C channel, showing that this channel was indeed established and that command dissemination can occur across it.

In this section the internal design of the Bluetooth C&C channel was explored, focusing on the pairing process and the communication between the mobile bots, which have already bonded. The pairing process, as followed by the Bluetooth C&C channel, requires no user involvement and allows mobile devices to pair without being in a discoverable mode. The mobile devices are not required to be discoverable since the mobile bots have access to all of the necessary Bluetooth MAC addresses to allow the pairing to proceed. Removing the bond between the mobile bots after completing the command dissemination ensures that the establishment and the execution of the Bluetooth C&C channel remain hidden to the users of the mobile devices.

The next section deals with the evaluation of the prototype, with a closer look at the characteristics that the Hybrid Mobile Botnet aimed to achieve and the analysis of the infected mobile devices.

7.3 Prototype Evaluation

The execution of the prototype was replicated on three separate occasions, allowing for the evaluation of the prototype. In this section special attention is given to the objectives of the Hybrid Mobile Botnet and whether the prototype was able to meet them, focusing especially on battery and data consumption. The analysis of an infected mobile device is also important and will aid the evaluation of the performance of the prototype. This includes assessing mobile security and anti-virus software and whether they are able to detect the Hybrid Mobile Botnet.

7.3.1 Achievement of the Hybrid Mobile Botnet Objectives

One of the goals of the prototype was to explore the efficiency of a hybrid C&C structure against the following characteristics: no single point of failure within the topology, low monetary cost for command dissemination, limited network activities and low battery consumption per bot.

The achievement of the first two characteristics follows from the design of the Hybrid Mobile Botnet. During the execution of the prototype, the last characteristics were closely examined.

The first characteristic of the Hybrid Mobile Botnet was accomplished by deploying multiple strategies. Firstly, the botmaster stores multiple copies of the information on the control server; thus should either the server or the information on the server become compromised, the botmaster can use one of the saved copies to restore the control server. Secondly, the Hybrid Mobile Botnet ensures that each mobile bot contains a file (bot list) with all of the bot IDs (Bluetooth MAC addresses) of all the other mobile bots within a cluster botnet. The bot list allows the cluster head bots to continue disseminating the commands should the control server ever become inaccessible. Thus, if the control server becomes unavailable the Hybrid Mobile Botnet will still be able to function to a certain degree by using the information in the bot lists. The provided strategies to overcome the problem of the control server becoming a single point of failure are not a complete solution. Therefore, to avoid the concern of a single point of failure, each mobile bot has only the domain name of the control server. Thus when the IP address of the control server is detected as malicious and is removed, the botmaster can move the control server to a new location with a different IP address while the mobile bots still connect to the same domain name. This technique has been previously proposed by Schiller (2007).

The formation of cluster botnets, which can only communicate via Bluetooth, ensures that there is no monetary cost involved during the process of command dissemination for cluster botnets. The use of the arbitrary tree structured topology limits the number of SMS messages that can be sent from individual cluster head bots, which also causes the cost of communication to be low, thus achieving the second characteristic.

A sudden increase in data consumption can potentially alert the user of a mobile device of the presence of a mobile bot. To avoid detection, the data consumption of the mobile bot must be as low as possible. After each successful execution of the prototype, two mobile devices were analysed to determine the data consumption of a mobile bot and in addition, the difference between the data consumption of a cluster head bot and a receiver bot was also determined. The data consumption of a mobile bot is displayed in Table 7.2 according to the role the mobile bot assumed during the execution of the prototype. With each experiment the data consumption of the cluster head bot was higher than that of the receiver bot since the cluster head bot was responsible for retrieving the Bluetooth MAC addresses and locations of all of the receiver bots within a cluster botnet. However, the difference is small enough not to be easily distinguishable between a cluster head bot and a receiver bot by simply looking at the data consumption.

Table 7.2: Data consumption of a mobile bot according to the role assumed during the execution of the prototype

Experiment	Receiver bot	Cluster head bot
1	5.20 KB	5.69 KB
2	5.00 KB	7.00 KB
3	4.61 KB	5.06 KB

On average, a mobile bot consumes 5.427 KB during its execution with a standard deviation of 0.846 KB. This low standard deviation indicates that the collected data are closely located to the mean, revealing that a mobile bot will mostly likely consume between 4.58 KB and 6.27 KB on a daily basis. During a weekly period the average data consumption of a mobile bot is less than 40 KB and over a monthly period it is less than 160 KB.

With many mobile devices being accompanied by data bundles, this low consumption of data will not alert the mobile device user and will also increase the difficulty of detecting the mobile bot. In addition, only connecting a limited number of times to the control server (once when the application is first installed and then after every period of low mobility) ensures that the characteristic of limited network activities is achieved.

A significant decrease in battery power will also potentially alert the user of a mobile device of the presence of the mobile bot which, in turn, can lead to the mobile bot's discovery. Therefore, throughout the execution of the prototype the consumption of the battery power was closely monitored.

To monitor the consumption of battery power effectively, the GSam Battery Monitor[2] application was installed on all the mobile devices involved in the execution of the prototype. This application can monitor each installed application or service on a mobile device individually and determine the consumption of battery power. During the three separate executions of the prototype, each mobile bot only consumed 0.1% of the battery's power over a period of three hours. During 24 hours the mobile device lost only 0.8% of its power while the mobile bot executed. Thus the execution of the mobile had

[2] https://play.google.com/store/apps/details?id=com.gsamlabs.bbm&hl=en

little influence on the battery, increasing the difficulty of detecting the mobile bot and confirming the achievement of the very last characteristic of the Hybrid Mobile Botnet.

By meeting all of these characteristics the Hybrid Mobile Botnet is shown to be stealthy, cost-effective and robust. It thus has the ability to remain undetected for lengthy periods. There is, however, still the possibility of detecting the Hybrid Mobile Botnet by other strategies. Such strategies include using mobile security and anti-virus software.

7.3.2 Analysis of Infected Mobile Device

With today's modern computers, anti-virus and security software have become important components of everyday computing. Such software provides end-users with a certain level of protection against malware, if the software is updated regularly. With the recent increase in popularity of mobile devices, anti-virus creators have started developing security applications for these devices. The ability of these applications to detect the Hybrid Mobile Botnet is reported on next.

Four mobile anti-virus applications were installed on each mobile device prior to the execution of the prototype. The selected applications are AVG Anti-virus[3], Avast Mobile Security[4], Lookout Security & Anti-virus[5] and Norton Mobile Security[6]. AVG is well known for their security software for personal computers and with their mobile application they combat malware and provide loss and theft protection. Avast Mobile Security provide similar functionalities to AVG but in addition offer privacy reports, SMS/call filtering and management of applications. Besides the basic security features, Lookout also offers the possibility of locating a missing device and provides backup functionalities. Norton released a mobile version which features the following functionalities: remote wiping, remote locking and call/text blocking.

All of the above anti-virus applications were active during the execution of the proto-

[3] http://www.avg.com/za-en/for-mobile

[4] http://www.avast.com/free-mobile-security

[5] https://www.lookout.com/

[6] https://mobilesecurity.norton.com/

type but failed to identify any malicious activities. After the execution of the prototype, all anti-virus applications preformed a scan of all the available applications on the mobile device. None of these scans found any malicious applications.

After the scan, AVG Anti-virus reported on an insecure setting. The setting, USB debugging, is intended for development purposes only. It is used to copy data between the mobile device and a computer, and also allows for the installation of applications without any notification. This setting should only be enabled for development purposes and since this setting was required to log all messages while the prototype executed (see Section 7.2), it is not uncommon to discover this warning.

Also discovered during the evaluation of the anti-virus applications is the fact that they share most of the same permissions as the mobile bot application (for Android permissions see Section 3.2.1). The permissions are access to location data, reading identity information and accessing messages. So it becomes impractical to determine whether an application is malicious or not by simply looking at the permissions.

The analysis of the anti-virus applications shows that new malicious mobile software can go undetected. This inability of anti-virus applications to identify the mobile bot application and the sharing of multiple permissions improves the secrecy with which the Hybrid Mobile Botnet can operate.

7.4 Strengths and Weaknesses of the Hybrid Mobile Botnet

The potential strengths and weaknesses of the Hybrid Mobile Botnet are identified in this section. The strengths are evaluated against the following properties: stealth, robustness, reliability, adaptability and cost-effectiveness. The section concludes with a short discussion of the weaknesses of the Hybrid Mobile Botnet.

7.4.1 Strengths

7.4.1.1 Stealth

Firstly, the Hybrid Mobile Botnet can only access the Internet in the background and does not alert the user of the mobile device about the activities taking place on the device. Secondly, the use of HTTP, which is a popular protocol for Internet traffic, as a C&C channel allows the malicious traffic to bypass firewalls. Thirdly, the ability of the Hybrid Mobile Botnet to intercept incoming SMS messages before they reach the device's SMS application improves the secrecy of this mobile botnet. In addition, the Hybrid Mobile Botnet only intercepts SMS messages containing the commands and no traces of these messages can be found on the mobile device, further protecting the identity of the botmaster and other cluster head bots. Finally, the Bluetooth C&C channel is established without any user involvement and automatically performs the required Bluetooth tasks in order to pair two mobile bots. The automated Bluetooth connectivity increases the secrecy with which the Hybrid Mobile Botnet can operate. The use of a hybrid C&C structure allows the Hybrid Mobile Botnet to be split up in multiple clusters, making it harder to detect and destroy.

7.4.1.2 Robustness

The ability of the topology structure of the Hybrid Mobile Botnet to change according to the mobility of the mobile devices makes this mobile botnet more robust against detection measures. The continuous change in geographical positioning, even though the changes might be small, make the use of signature-based detection measures difficult since this information is included in the content sent over the HTTP C&C channel. To detect the use of Bluetooth as a C&C channel, the defender has to be physically close to this channel. This is not always practical which increases the difficulty of detecting the Hybrid Mobile Botnet. The strict control of the Bluetooth C&C channel and the restriction of the number of attempts to forward a command to a receiver bot also decreases the consumption of battery power. The low battery consumption improves the strength of the Hybrid Mobile Botnet, allowing the botnet to operate for longer periods without alerting the user of the mobile device. The formation of the cluster botnets removes the necessity to keep information about the entire botnet on each mobile bot, making the design more robust against potential detection measures.

7.4.1.3 Reliability

As with all other mobile botnet designs, the control server plays a critical role in the overall success of the Hybrid Mobile Botnet. It improves management and repair of the mobile botnet, taking the task away from both the individual mobile bots and the botmaster. The control server can also easily become a single point of failure within the overall design of the mobile botnet, rendering the botnet non-functional should the server ever be compromised. To improve reliability, the very first command dissemination sequence will ensure that each mobile bot contains the necessary information regarding the specific cluster botnet it belongs to. The identity of the cluster head bot is known to all the receiver bots, while each individual receiver bot knows the Bluetooth MAC address of the cluster head bot. This information, which is stored on the mobile devices, enables the Hybrid Mobile Botnet to continue functioning to a certain degree even if the control server is no longer available.

7.4.1.4 Adaptability

As explained throughout this dissertation, the C&C channels play an important role in the design of a mobile botnet. Without the C&C channels the botmaster will not be able to communicate with the mobile botnet or forward the necessary commands. The Hybrid Mobile Botnet uses multiple C&C channels, namely SMS, Bluetooth and HTTP, to improve the stealth of the mobile botnet. If the SMS C&C channel fail, the botmaster can adapt the internal structure of the Hybrid Mobile Botnet to use the HTTP C&C channel to disseminate the commands. Whenever a mobile bot contacts the control server, the response can also include the command. If the Bluetooth C&C channel ever fails, the cluster head bots can adapt their execution strategy and instead use the SMS C&C channel to disseminate the commands to the receiver bots. The ability to switch between the C&C channels does compromise the performance and cost-effectiveness of the Hybrid Mobile Botnet, but the adaptability allows the Hybrid Mobile Botnet to continue functioning even if a C&C channel becomes compromised.

7.4.1.5 Cost-effectiveness

There are multiple factors that contribute to the minimal cost impact of the Hybrid Mobile Botnet. Firstly, the Hybrid Mobile Botnet is inactive for long periods (e.g. at night), thus preserving the battery power of the mobile device. Secondly, the use of the arbitrary tree structure topology limits the number of SMS messages that each individual cluster head bot can send, making the Hybrid Mobile Botnet more cost-effective in monetary terms. With the Bluetooth C&C channel, the Hybrid Mobile Botnet also achieves cost-effectiveness in monetary terms because establishing Bluetooth connections will not inflict any costs on the accounts of the users with the infected mobile devices. In addition, the use of the Bluetooth C&C channel minimises the necessity for sending out SMS messages, further improving the cost-effectiveness of the Hybrid Mobile Botnet. Finally, by collecting a subset of information from the device and sending the collected information on a single occasion to the control server the data consumption of the Hybrid Mobile Botnet is limited. Limiting the connections to the control server not only decreases the data consumption, but also improves the difficulty of detecting the Hybrid Mobile Botnet. Keeping the Hybrid Mobile Botnet cost-effective will cause it to avoid detection efficiently.

7.4.1.6 Scalability

The Hybrid Mobile Botnet addresses and accommodates scalability through multiple factors. Firstly, the internal structure of the Hybrid Mobile Botnet supports the construction of cluster botnets, which reduce the number of mobile bots that must receive the command from their respective cluster head bot. Should a specific cluster quickly grow in size, additional cluster head bots can be identified to support the sending of the command. Secondly, the use of multiple C&C channels allows the Hybrid Mobile Botnet to spread the load of command dissemination across these channels. As the Hybrid Mobile Botnet grows and the load on the C&C channels increase, more cluster head bots can be identified to support the sending of the command. Finally, with the Hybrid Mobile Botnet limiting the number of connections made to the control server (only contacting the control server after initial installation and at the end of every period of low mobility), the server will be able to support the addition of new mobile bots.

7.4.2 Weaknesses

The Hybrid Mobile Botnet is not without any weaknesses. Firstly, even though it is capable of functioning without the control server, the performance of the botnet will still be affected. Without a functioning control server the Hybrid Mobile Botnet will be restricted to previously formed cluster botnets but due to the mobility of the mobile devices, these cluster botnets can still continuously change. Thus there might only be a few mobile bots still available in the cluster botnet, severely impairing the effectiveness of certain attack strategies. Secondly, a loss of any of the C&C channels will impact the performance of the Hybrid Mobile Botnet even though it will still be able to function to a certain degree. The use of the HTTP C&C channel instead of the SMS C&C channel will still allow the botmaster to disseminate commands to the mobile bots, but the bots can only retrieve the command when connecting to the control server, delaying the eventual attack. Replacing the Bluetooth C&C channel with the SMS C&C channel will impact the cost-effectiveness of the Hybrid Mobile Botnet since more SMS messages will need to be sent. However, these weaknesses mentioned above do not impact the functionality of the Hybrid Mobile Botnet as it is still able to receive and execute commands.

The section provided an overview of the strengths and weaknesses of the Hybrid Mobile Botnet, concluding the evaluation of the Hybrid Mobile Botnet.

7.5 Conclusion

To measure the characteristics (as set out in Chapter 5) and determine whether current mobile technology can support hybrid mobile botnets, a prototype of the new Hybrid Mobile Botnet design was built which included limited, but the essential, functionality. This prototype consisted of a collection of mobile devices, all infected with the malicious bot code. By executing the prototype on several separate occasions in a controlled environment and analysing the results, it was found that the Hybrid Mobile Botnet met all of the pre-described characteristics: no single point of failure within the topology structure, low monetary cost for command dissemination, limited network activities and low battery consumption per mobile bot. By the end of the evaluation process, it was found that the Hybrid Mobile Botnet can avoid detection efficiently and even if a part of the structure collapses (such as the control server), the mobile botnet will still be able to function to a certain degree. The successful execution of the Hybrid Mobile Botnet prototype shows that a hybrid C&C structure is feasible and that current mobile technology can support the development and execution of hybrid mobile botnets efficiently. The knowledge obtained by designing a new mobile botnet is expounded on in the next chapter and this knowledge is summarised in terms of the growth potential of mobile botnets and the security steps users can take to protect against their attacks.

Chapter 8

The Future of Mobile Botnets and Security Measures

The Hybrid Mobile Botnet prototype reported on in the previous chapter shows that current mobile devices contain all the necessary capabilities to develop hybrid mobile botnets. In addition, the prototype also revealed that mobile botnets are a real threat for the users of mobile devices. Botmasters will constantly search for new ways to develop mobile botnets and as the technology of mobile devices continues to improve, the ability to develop mobile botnets improves as well.

There are lessons to be learned from designing and experimenting with a new mobile botnet design. The growth potential of mobile botnets is determined in Section 8.1 by examining additional mechanisms for mobile botnet development, improvements of mobile device technology and attractions for botmasters. The focus in Section 8.2 is on the security steps that users can follow to protect their mobile devices against malware infections, with special attention given to mobile botnets. Section 8.3 concludes the chapter.

8.1 Growth Potential of Mobile Botnets

The successful development and deployment of the Hybrid Mobile Botnet prototype shows that hybrid mobile botnets are indeed a real threat to users of mobile devices and that their capabilities are actually far advanced. With mobile botnets being a real threat and the continuous improvement in mobile technology, the future of mobile botnets looks promising and this section will focus on the growth potential of mobile botnets. In more detail, additional mechanisms that can be used in mobile botnet development are discussed, along with improvements of mobile device technology and attractions that draw botmasters to continuously improve mobile botnet designs.

8.1.1 Additional Propagation and C&C Mechanisms for Mobile Botnet Development

Current mobile botnet designs commonly exploit well-known mechanisms of mobile devices such as SMS messages, Bluetooth technology, social media, P2P structures and the HTTP protocol. The technology of mobile devices is, however, continuously evolving with new improvements being made on a regular basis. Some of the latest trends included in mobile devices are cloud computing, NFC and Wi-Fi Direct.

Cloud computing is currently one of the latest trends in modern computing and offers a new dimension to everyday computing. The National Institute of Standards and Technology (NIST) defines cloud computing as “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction” (Mell and Grance 2011). The most important aspect of cloud computing is the shift in geography of computing (Hayes 2008). With cloud computing data, information and even the applications are hosted by various computers that are scattered across the globe. The decentralized nature of cloud computing, as opposed to a web server, allows for the sharing of resources. Therefore the C&C server can easily be moved between all of the computers within the cloud, hiding the C&C server and so increasing the difficulty of detecting the location of the server. Cloud computing, however, raises questions about privacy, reliability and, more importantly, security (Hayes 2008).

One of the leaders of the cloud computing evolution is Google. Google’s approach towards cloud computing is mostly web-based, where any mobile device with a web browser and an Internet connection can access most of Google’s services such as Gmail, Google Calendar, Google Docs, Google App Engine and Google Cloud Storage (*Google’s*

Approach to IT Security 2012). The advantage of this approach is that access to the data is no longer device dependent. This interconnectivity between mobile devices provides mobile botnets with an additional propagation mechanism. For example, if a mobile botnet can infect a smartphone with the malicious bot code, the mobile botnet can then exploit the interconnectivity provided by cloud computing to spread the bot code to other mobile devices. To achieve this, the mobile bot can perhaps infect a document stored in Google Docs by attaching the malicious bot code. Should this particular document be opened on a tablet computer or another smartphone, the mobile bot will immediately install itself onto the new devices as well.

Zhao et al. (2012) explored the possibility of using cloud computing as a propagation mechanism and found that cloud computing can be a feasible mechanism to use for C&C.

Although such a propagation mechanism is yet to be researched further in literature, it provides mobile botnets with a new possibility to grow. As cloud computing evolves, botmasters will surely start searching for new vulnerabilities to exploit within cloud computing technology.

NFC allows for two-way communication between electronic devices by forming a short-range, high-frequency wireless connection that enables data transfer to occur between a reader and a target (Eamrurksiri and Xiang 2012). The reader, also called the initiator, is responsible for generating a radio frequency (RF) signal and the controlling of the data transfer. The request is answered by the target if the RF signal is established successfully (Curran et al. 2012). Since the range of NFC is only a few centimetres, it is mostly used for mobile payments (Eamrurksiri and Xiang 2012, Curran et al. 2012). As NFC becomes more popular, malicious exploits of this technology are also rising. One such example is where NFC is used as a form of advertising and interested users can tap their mobile device onto the advert to view the message. Fraudsters can remove the legitimate tags and replace them with tags that will infect the device with malicious code (Curran et al. 2012).

Even though NFC is still a relatively new technology in the mobile environment, it is quickly starting to find its way onto mobile devices. If botmasters can find vulnerabilities within NFC technology to exploit (NFC is not encrypted according to the International Organization for Standardization (ISO) standard (Curran et al. 2012)), NFC can become a valuable propagation mechanism for mobile botnets. To use NFC as a propagation mechanism, a botmaster can place malicious NFC tags in public locations. If a user places the mobile device against the tag, an advertisement will show but in the background malicious code is installed onto the device. The purpose of the advertisement is to not alert the user of the malicious activity taking place on the device. Botmasters can also possibly use NFC as a C&C channel to disseminate a command to the mobile bots but due to the short distance of communication required by NFC, this can be impractical.

Another technology botmasters can turn to is Wi-Fi Direct. With Wi-Fi Direct, devices can create a one-to-one connection or group several devices simultaneously (*Wi-Fi Certified Wi-Fi Direct* 2010). This technology allows devices to directly connect to one another without accessing a traditional network. Some of the advantages of Wi-Fi Direct are mobility, portability, immediate utility, ease of use and secure connections (*Wi-Fi Certified Wi-Fi Direct* 2010). With Wi-Fi Direct being capable of peer-to-peer connections (*Wi-Fi Certified Wi-Fi Direct* 2010), this technology provides botmasters with yet another mechanism for propagation or command dissemination. A mobile botnet can possibly exploit vulnerabilities of Wi-Fi Direct and allow the infected mobile devices to establish a peer-to-peer connection when in range that can either be used for the propagation of the bot code or for the creation of a C&C channel for command dissemination.

Technologies such as cloud computing, NFC and Wi-Fi Direct are still relatively new but are quickly evolving and making their way towards mobile devices. As with all other technologies, these technologies are bound to have hidden vulnerabilities that botmasters can exploit during the development of new and improved mobile botnets. However, it is not only new technology that can have an impact on mobile botnet development, but also the continuously improving technology of mobile devices.

8.1.2 Improvements of Mobile Device Technology

As with all forms of technology, mobile devices such as smartphones and tablet computers are continuously evolving and improving. The contributors to this continuous improvement are mobile processors, the growth of internal memory and the popularity of mobile applications.

Mobile processors have come a long way since the introduction of these processors by Advanced RISC Machines (ARM), a British multinational semiconductor company (Tariq 2012). ARM designs mobile processors that consume less energy and perform similar to a processor of a regular computer. At the start of 2013 there were four major players in the mobile processor industry: Samsung, Nvidia, Texas Instruments and Qualcomm (Tariq 2012). 2013 also saw mobile processors move beyond dual-core processors and they are now shipping with 1-gigahertz quad-core processors. The most advanced mobile processors released at the start of 2013 were Samsung's Exynos 5 Octa processor, which features eight CPU cores (Franklin 2013) and Nvidia's Tegra 4 processor, which features a quad-core processor together with a fifth, low power core to save battery life (Souppouris 2013).

Mobile processors, such as those mentioned above, allow for improved performance, quicker processing and support the execution of complex tasks. Besides supporting more advanced activities, they are also now capable of supporting improved mobile botnet designs. Thus improved mobile processors will allow for improved mobile botnet designs that will be able to execute faster, perform complex tasks and be more reliable.

As with personal computers, the internal memory of mobile devices has increased significantly during the last decade. Today's mobile devices have a capacity of internal memory ranging between 2 GB and 16 GB. Most of the mobile devices available today also feature slots that allow for the expansion of memory by inserting additional memory cards. The need for large quantities of memory is to accommodate the user's photos, music files and videos. This vast amount of internal memory is also necessary to support the execution of applications and therefore can also easily support mobile botnet activities. As the internal memory of mobile devices continues to grow, it provides botmasters with the ability to design greater mobile botnets that can execute more complex tasks and store greater amounts of information.

The development of mobile applications has greatly simplified since the arrival of smartphones and tablet computers. In the past, users of mobile devices were able to develop their own applications but due to the difficulty of adding an application to the device, this was rarely done. Nowadays it is far easier to develop applications and by simply pressing a button, the application can be installed onto the mobile device. The ease with which users can develop and install applications today makes the development of mobile botnets much more feasible. There is no longer a need for extensive knowledge about mobile devices or malicious coding in order to develop a mobile botnet since any person with a bit of knowledge about programming and botnet structures can do so. Due to the popularity of mobile applications, any malicious application can potentially find its way onto mobile devices.

Mobile device technology will continue to grow, providing users with faster mobile devices that can store greater amounts of data and support more advanced applications. The improvement of mobile device technology will also enable botmasters to develop complex and advanced mobile botnets that are capable of exploiting these improvements. These continuous improvements of mobile device technology are not the only attraction for mobile botnet developers and other potential attractions will further be explored in the upcoming section.

8.1.3 Attractions for Botmasters

Although the improvements in mobile technology are clearly the main attraction for botmasters, there are also various additional factors that draw botmasters towards mobile devices. These attractions are mostly motivated by financial gain, information stealing and the opportunity to exploit vulnerabilities.

Botmasters are often motivated by greed when they develop mobile botnets and the technique they currently use to achieve financial gain is by sending out SMS messages to premium-rate numbers (Spreitzenbarth and Freiling 2012). However, deploying mobile botnets is not the only possibility for botmasters to gain financially. The last few years have seen the introduction of the mobile wallet, a type of electronic wallet that is able to carry out electronic transactions by using a mobile device that supports NFC (Amoroso and Magnier-Watanabe 2011). This concept is slowly being introduced to users of mobile devices by Google's Google Wallet and the Wallet Hub designed for Windows Phone 8.

At the start of 2012, security firm Zvelo discovered a vulnerability in the personal identification number (PIN) process used by the Google Wallet application. Zvelo found that the PIN, which the users need in order to confirm purchases made with their mobile device, can be cracked using an exhaustive numerical search (Purewal 2012). This is possible since the Google Wallet PIN information is stored on the device and not on the NFC chip (Purewal 2012). Botmasters can use mobile botnets to perform the exhaustive search and obtain the PIN, enabling a botmaster to use the infected mobile device to make purchases.

Unlike mobile wallet applications, mobile banking applications are not new to mobile devices and allow end-users to complete completing banking transactions in the comfort of their homes by using their mobile devices. Many users of these applications are, however, wondering about the security of the applications. According to a study performed in 2011 by the digital forensics and security firm, viaForensics, mobile banking applications are not very secure (Crosman 2011). The study found that 25% of the evaluated mobile banking applications did not provide adequate security measures (Crosman 2011). Information such as passwords, partial credit card details, payment history and transaction details were easily retrieved from the mobile device (Crosman 2011). Botmasters can use mobile botnets to collect this information, which the botmasters can then further exploit for possible financial gain.

Mobile devices are great sources of data and personal information. Besides financial information, mobile devices also store contact information, location data, credentials and private details. Botmasters can use mobile botnets to obtain such information and then further exploit the information by executing targeted attacks such as identity theft or financial fraud. If such information is not directly available on the mobile devices, mobile botnets can exploit vulnerabilities in order to gain access to the required information. Mobile OSs are filled with many hidden vulnerabilities and all the botmasters are required to do is find the appropriate vulnerability to accomplish the task.

The growth potential of mobile botnets looks promising with the addition of new mechanisms, such as cloud computing, NFC and Wi-Fi direct, continuous improvements in mobile device technology and a vast collection of promising attractions. With so many possibilities for mobile botnets to grow, users of mobile devices require protective measures to defend against mobile botnet infections. The next section introduces security steps that users of mobile devices can follow to defend against potential mobile botnet attacks.

8.2 Security Steps for Users of Mobile Devices

Mobile botnets are a real threat and can severely impact the end-users, either financially or simply emotionally. Protection measures to defend against mobile botnets have not yet evolved to the point where they are as efficient as those measures used to defend against traditional botnets. The most basic step that mobile users can take to protect themselves against mobile botnets is to install anti-virus and security applications. Such applications can only detect previously identified mobile botnets and will not always detect newly designed mobile botnets as shown in Section 7.3.2. Users of mobile devices must therefore take additional steps in order to protect their mobile devices against malware infections, including mobile botnets.

Although these steps are defined according to the experience gained by using the Android OS as the developing platform, they can also be followed on mobile devices with different mobile OSs.

8.2.1 Step 1: Caution

The first step involves caution, and therefore users of mobile devices must take the necessary precautionary steps to avoid danger or common mistakes. Protection of mobile devices starts with the installation of anti-virus and security applications. Even if these applications are not capable of detecting new threats, they still provide the user with valuable services to secure the mobile device if it ever gets lost or stolen. Such services include locating the device by using the GPS functionality, remotely locking the device via the Internet and remotely wiping the device, which will delete most of the personal information stored on the mobile device such as contacts, text messages and photos. So besides their shortcomings, anti-virus and security applications still offer many beneficial services for users of mobile devices.

Many users today are rooting their Android devices. This enables a user to obtain superuser privileges, which then allow the user to alter certain system settings that would otherwise not have been possible. Rooting a mobile device, such as a smartphone, is a complex process and if not performed correctly users can end up bricking their devices, which renders them unusable. Furthermore, some companies today state that if a user roots their purchased device, the warranty associated with that specific device becomes void. The user will thus not have any assistance from the company if anything goes wrong with the mobile device. Lastly and most importantly, giving the mobile device superuser privileges also enables the mobile botnet to use these privileges and this can therefore cause much more harm to the device. Thus rooting mobile devices is not recommended for users.

Mobile devices have become popular due to their ability to host various applications. Such applications can either be downloaded from trustworthy market stores, such as Google's Play Store, or from third-party market stores. It is the responsibility of the user to decide where the applications are downloaded from and it is therefore important to know that malicious applications are mostly found on third-party market stores. In order to run a lower risk of getting infected by mobile botnets, users should only download applications from trusted market stores.

Mobile botnets are known to use networking technologies, such as Bluetooth, to propagate and perform command dissemination (Yan 2006). It will not be long before mobile botnets also start exploiting NFC and Wi-Fi technologies in order to improve their growing potential. Users who continuously enable Bluetooth and Wi-Fi are at a higher risk of getting infected by mobile botnets. To protect their mobile devices, users should keep Bluetooth and Wi-Fi turned off whenever possible. If the user requires any of these technologies, they must carefully select the network to which the device will connect. Users must only connect to secure Wi-Fi networks and regularly monitor the connection. Whenever possible, such as at night, the user should turn off the data network. This will prevent any Internet activities from taking place and, should a mobile botnet infect the mobile device, this will limit the botnet's ability to communicate beyond the device. Another step users can take to protect their mobile devices is to manage the location settings. If the user wants to keep their location a secret and prevent mobile botnets from exploiting the location data, users must turn off all forms of location assessment whenever possible.

Users must always use a password or pattern to lock their mobile device and so prevent an unauthorised person from accessing the device. They must avoid weak passwords or patterns that can easily be guessed or enumerated. Furthermore, users must avoid storing sensitive information on their devices but, if this is not possible, sensitive information must be encrypted by using a secure encryption algorithm. Should an unauthorised person gain access to the mobile device, such a person will not be able to view the sensitive information.

The last cautionary step that users of mobile devices can take is to restrict usage of their devices. Users must never leave their devices unattended or lend them to strangers. By ignoring the cautionary steps set out in this section, users will give mobile botnet developers the opportunity to infect mobile devices.

8.2.2 Step 2: Investigate

The second step involves the investigation of a mobile device. It is the responsibility of the user to regularly check the account associated with the particular device. By investigating the accounts regularly, the user will be able to determine if there is any malicious activity taking place on the device. The first indication of potential malicious activities can be a sharp increase in the bill. If this is indeed the case, the user must search for specific activities such as the sending of SMS messages or phone calls to premium-rate numbers. Detecting malicious activities frequently can possibly save the user money and frustration.

Applications are an integral aspect of mobile devices as they provided the users with endless functionality. The popularity of applications has caused them to be used for malicious purposes such as aiding the propagation of a mobile botnet. It is therefore necessary for users to properly check the permissions of applications. The user must do this before and after the installation of an application. When installing an application, the user must thoroughly read through all the permissions before continuing with the installation. For each permission, the user must ask whether this particular permission is really necessary and if it applies to the functionality provided by the application. If the user finds any suspicious permissions, the installation of the application must immediately be halted. Carefully evaluating the permissions of applications can equip the user with the necessary knowledge to prevent mobile botnet infections on their mobile devices.

Users must systematically check the applications and files stored on their mobile devices. Mobile botnets are capable of installing additional applications and files as required by the botnet in order to operate successfully. Users must therefore be aware of all the applications and files stored on the device in order to be able to detect any malicious content. In addition, users must also check the settings of their devices and any settings that a mobile botnet can potentially exploit must be turned off.

By investigating their mobile devices frequently, users will develop the necessary awareness to detect potential mobile botnet infections.

8.2.3 Step 3: Monitor

For users of mobile devices, simply being cautious and checking the applications is not enough. Users must also continuously monitor their devices for any potential malicious activities. To monitor a mobile device, users must start with the evaluation of data consumption and battery life. Mobile botnets are known to consume data since the botnet must connect to a C&C server from time to time. Even though such consumption might be insignificant on a daily basis, the increase of data usage after a monthly period can be more severe. It is therefore the responsibility of the user to regularly monitor the consumption of data and to be aware of the average consumption of data on a monthly or weekly basis. Should there be a sudden increase in data consumption over a specific period, the user must repeat the first two steps in order to check for potential malicious content.

The consumption of battery power is also a significant tool for the detection of mobile botnets. Mobile botnets often continuously execute activities in the background without alerting the user of the mobile device. Such activities, depending on their frequency, will consume battery power. Most mobile devices on average have a battery life of approximately 9 hours, but this will differ from one device to the next as each person uses their device differently. It is therefore once again the responsibility of the user to frequently monitor the consumption of battery power. Should the user notice a sudden decrease in battery power, it can be that one of the latest installed applications or files contains malicious content. Such detection of mobile botnets will only be possible if the user regularly monitors the battery consumption of the mobile device.

It is important to frequently monitor applications and the actions they perform.

The user must do this after the installation of an application as well as after each update. Although most malicious activities occur in the background, they can have a negative influence on the infected application. It is the responsibility of the user to look out for abnormalities in applications. Such abnormalities can include: applications that continuously crash, slow performing applications, functionality of the applications hindered and the crashing of the mobile device. All of these abnormalities are possible characteristics of a malicious application.

The user must also monitor Wi-Fi and Bluetooth connections and regularly check the connections made to Wi-Fi networks and Bluetooth devices. Should the user locate any suspicious connections to unknown networks or devices, it can be an indication of potential malicious content installed on the device that is exploiting these technologies. To discover the malicious content, the user must revisit each application and carefully check the permissions associated with an application.

Users must frequently monitor the data consumption, battery life and applications installed on the mobile devices. Such knowledge can aid the user with mobile botnet detections and infections.

8.2.4 Step 4: Update

Most mobile OSs have been plagued by exploits. The Android OS is often targeted by premium service abusers or adware (*Evolved Threats in a Post-PC World* 2013). Malware developers are the front-runners of discovering vulnerabilities and successfully exploiting them in the execution of mobile botnets. It is only when researchers discover the malware that the vulnerability becomes publicly known. Developers of the mobile OS, with which the vulnerability is associated, must quickly fix the vulnerability and release a patch to the end-users.

In order to protect against malware infections such as those of mobile botnets, the user must keep the mobile OS of their devices updated. This will require the downloading and installation of patches and new versions on a regular basis. Besides the mobile OS, users must also keep the installed anti-virus and security applications updated. Anti-virus and security applications can only protect mobile devices from known malware infections. If such applications are not updated frequently, previously detected malware can possibly slip onto the device.

To ease the process of regularly updating the software and applications of the device, users can enable auto update. This will automatically update the mobile OS and other applications should new patches or versions become available.

If users are able to frequently update the software of their mobile device, malicious content will not easily find their way onto the device.

8.2.5 Step 5: Remove

The last step involves the removal of applications and files that are no longer being used or that are potentially malicious. Mobile botnets can only cause damage as long as they remain undiscovered. Once located, the user must immediately remove the infected application or file from the device. If a user discovers any suspicious applications or files on the device that were either not installed by the user or not there previously, such an application or file must also be removed immediately.

Any applications or files that were previously installed or added by the user but are no longer being used must also be removed from the device to minimise the impact of mobile botnets. It is possible for malware developers to hide malicious content in other applications or files.

This removal of unwanted software from a mobile device is the final step that users can take to protect their device against potential mobile botnet infections.

8.3 Conclusion

This chapter focused on the lessons learned from designing and developing a new mobile botnet. Firstly, mobile botnets are here to stay and with the ever-improving mobile technology, mobile botnets will surely continue to grow and improve. In order to confirm this growth potential of mobile botnets, three dimensions that can aid the growth of mobile botnets were discussed in this chapter. Technologies such as cloud computing, NFC and Wi-Fi Direct provide mobile botnets with additional mechanisms for propagation and command dissemination. Continuous improvement of mobile device technology will facilitate the development of mobile botnets and will allow for more advanced botnet designs in the future. Attractions such as financial gain, information stealing and vulnerability exploits will continue to attract malware developers towards mobile devices. Secondly, anti-virus and security applications cannot always protect mobile devices against new malware infections. It is the responsibility of the user to be constantly aware of the activities taking place on the mobile device. The five security steps described in this chapter offer users of mobile devices additional protection if performed frequently. These steps do not guarantee complete protection against malware infections, but can create awareness about how to protect against potential threats. In the end, all the power lies with the users and they will need to decide on how the mobile devices are used.

Chapter 9

Conclusions

As mobile devices become more powerful, they become ideal targets for mobile malware. One such threat that users of mobile devices are facing is mobile botnets. In this chapter the objectives concerning this study, which focused on designing a new mobile botnet, are revisited by reviewing the original problem statement. Section 9.1 provides a summary of the conclusions of this research and suggestions for future research are offered in Section 9.2.

9.1 Summary of Conclusions

The aim of this research was to explore the possibility of using a hybrid C&C structure within a mobile botnet design and to show that mobile technology is capable of supporting hybrid mobile botnet designs. This led to the design of the Hybrid Mobile Botnet model, which utilises multiple C&C channels, namely SMS, Bluetooth and HTTP. To measure and analyse the effectiveness and performance of this new model, a prototype was designed and executed on a small collection of mobile devices running the Android OS.

The value of the prototype was twofold - it showed that a hybrid C&C structure leads to a stealthy, cost-effective and robust design and also that current mobile technology is capable of supporting hybrid mobile botnets. Firstly, the use of multiple C&C channels within a dynamic environment allows the Hybrid Mobile Botnet to easily evade detection and remain hidden for long periods. The analysis of the prototype thus shows that this new mobile botnet exhibits the following qualities: stealth, reliability, robustness, adaptability and cost-effectiveness. Secondly, the successful execution of the prototype on real mobile devices shows that current mobile technology can support the development and execution of hybrid mobile botnets efficiently.

While conducting the study, contributions were made regarding the design of a Bluetooth C&C channel, potential attack strategies of mobile botnets and security measures for users of mobile devices. Using Bluetooth as a C&C channel in mobile botnet designs is not a novel idea and the feasibility of this technology in mobile botnet designs has been explored in previous literature. Previous research, however, only used simulations when exploring with Bluetooth as a C&C channel. During the development of the Hybrid Mobile Botnet, a physical Bluetooth C&C channel was designed and deployed. The successful execution of this channel shows that the security surrounding Bluetooth technology can be circumvented, allowing Bluetooth to act as a suitable mobile botnet C&C channel.

Throughout this study information concerning mobile botnets was also collected. Potential attack strategies of mobile botnets were identified, allowing for the classification of mobile botnets according to a specific goal the botnet aims to achieve. These various attack strategies also aid the development of detection measures since the strategies summarise the goal of a particular attack of a mobile botnet.

With mobile botnet protection and detection measures still in the early phases of development, users of mobile devices often fall victim to malicious applications. A short series of steps were proposed in this study that users of mobile devices can follow to potentially protect against mobile botnet infections. These steps, if performed regularly, enable users to be in control of their mobile devices and provide them with the necessary knowledge to potentially identify malicious activities. These steps do not guarantee complete protection against malware infections, but can create the necessary awareness surrounding future malware threats.

As mobile devices become more powerful, they become ideal targets for mobile botnets. The growing potential of mobile botnets goes hand in hand with the ever-improving mobile technology and the threats they pose will continue to rise alongside the increasing popularity of mobile devices. These contributions of this research offer necessary knowledge and insight concerning mobile botnets and can support future research.

To summarise, the future and potential of mobile botnets look bright. With the technology of mobile devices continuously improving, new opportunities to exploit these devices become available. Although protective mechanisms, such as security applications, are available, these mechanisms are used infrequently and often provide inadequate protection against new mobile botnet designs. Mobile botnets are here to stay, will continue to improve their techniques alongside mobile technology and will pose security threats to users of mobile devices.

9.2 Future Work

The relative newness of the field of mobile botnets allows the scope for future research to be wide open. A few suggestions for future research on mobile botnets are given below.

9.2.1 Improved Mobile Botnet Designs

This research provided a new Hybrid Mobile Botnet design, which is capable of evading detection and remaining hidden for extensive periods. To evade signature-based detection measures effectively, this design can be further improved by regularly changing the content of the communication occurring between the mobile bots and the control server. In addition, the mobile botnet activities can be randomised to improve the versatility of the design, increasing the difficulty of detecting the mobile botnet. There is also the possibility of exploring with other technology as potential C&C channels. Current possibilities include Wi-Fi Direct, NFC and cloud computing, each of which can further improve and advance mobile botnet designs.

9.2.2 Detection Mechanisms for Mobile Botnets

The prototype developed during this research showed that mobile devices are capable of supporting hybrid mobile botnet designs and that they are indeed a real threat to users of mobile devices. Available security applications mostly rely on signature-based detection techniques and therefore often fail to detect new mobile botnet designs. With the ongoing improvement in mobile technology, behaviour-based detection mechanisms are becoming a possibility. An example of a behaviour-based detection technique is to refine and automate the security steps identified in this research into an application, instead of relying on the user to execute these steps. This can be the first step taken regarding the development of behaviour-based detection mechanisms for mobile devices.

Bibliography

Agar, J. (2004), *Constant Touch: A Global History of the Mobile Phone*, Totem Books.

Alazab, M., Venkatraman, S. and Watters, P. (2011), Cybercrime: The case of obfuscated malware, *in* '*7th International and 4th e-Democracy, Joint Conferences*', *Thessaloniki, Greece*, pp. 204–211.

Amoroso, D. and Magnier-Watanabe, R. (2011), 'Building a research model for mobile wallet consumer adoption: The case of mobile Suica in Japan', *Journal of Theoretical and Applied Electronic Commerce Research* **7**(1), 94–110.

Ancarani, F. and Shankar, V. (2003), 'Symbian: Customer interaction through collaboration and competition in a convergent industry', *Journal of Interactive Marketing* **17**(1), 56–76.

Andrici, M. (2012), 'ICS brings IPv6 compatibility to the TMobile Samsung Galaxy S2'. Symantec. Available online: [http://www.androidauthority.com/ics-brings-ipv6-compatibility-to-the-t-mobile-samsung-galaxy-s2-94743/] (Accessed: 23 September 2012).

Apvrille, A. (2012), 'Symbian worm Yxes: Towards mobile botnets?', *Journal in Computer Virology* **8**(4), 117–131.

Aviv, A., Sherr, M., Blaze, M. and Smith, J. (2010), Moving targets: Geographically routed human movement networks, Technical report, Department of Computer & Information Science, University of Pennsylvania.

Babin, S. (2008), *Developing Software for Symbian OS 2nd Edition: A Beginner's Guide to Creating Symbian OS v9 Smartphone Applications in C*, 2nd edn, Wiley.

Bailey, M., Cooke, E., Jahanian, F., Xu, Y. and Karir, M. (2009), A survey of botnet technology and defenses, *in* '*Proceedings of the 2009 Cybersecurity Applications & Technology Conference for Homeland Security, CATCH'09*', *Washington, DC, USA*, pp. 299–304.

Barnes, S. (2002), 'Under the skin: Short-range embedded wireless technology', *International Journal of Information Management* **22**(3), 165–179.

Barrera, D. and Van Oorschot, P. (2011), 'Secure software installation on smartphones', *Security & Privacy* **9**(3), 42–48.

Belov, N., Braude, I., Krandick, W. and Shaffer, J. (2005), Wireless internet collaboration system on smartphones, *in* '*Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS), 17th International Conference on Advance Information Systems Engineering*', *Porto, Portugal*, pp. 675–689.

Bisdikian, C. (2001), 'An overview of the Bluetooth wireless technology', *Communications Magazine, IEEE* **39**(12), 86–94.

Bluetooth (2013). Available online: [http://www.bluetooth.com/Pages/History-of-Bluetooth.aspx] (Accessed: 28 June 2013).

Brown, J., Shipman, B. and Vetter, R. (2007), 'SMS: The short message service', *Computer* **40**(12), 106–110.

Bruno, R., Conti, M. and Gregori, E. (2002), 'Bluetooth: Architecture, protocols and scheduling algorithms', *Cluster Computing* **5**(2), 117–181.

Catanzariti, R. (2009), 'The mobile phone: A history in pictures'. PCWorld. Available online: [http://www.pcworld.com/article/172837/the_mobile_phone_a_history_in_pictures.html] (Accessed: 16 November 2012).

Chen, T. and Peikari, C. (2008), 'Malicious software in mobile devices', *Handbook of Research on Wireless Security* **5**, 1–10.

Chick, C. (2004), 'Managing your e-mail remotely', *Searcher - The Magazine for Database Professionals* **12**(6), 50–53.

Clapsadl, M. (2012), Standardizing the security of mobile app store platforms, PhD thesis, Utica College.

Cohen, F. (1987), 'Computer viruses: Theory and experiments', *Computers & Security* **6**(1), 22–35.

Cole, A., Mellor, M. and Noyes, D. (2007), Botnets: The rise of the machines, *in* '*Proceedings on the 6th Annual Security Conference*', *Las Vegas, NV, USA*.

Comerford, R. (2000), 'Handhelds duke it out for the Internet', *Spectrum, IEEE* **37**(8), 35–41.

Croft, N. and Olivier, M. (2007), 'A silent SMS denial of service (DoS) attack'. Tech Republic White Paper.

Crosman, P. (2011), 'Security warning: 25% of mobile banking apps flunk test'. American Banker. Available online: [http://www.americanbanker.com/issues/176_153/mobile-app-security-1040990-1.html] (Accessed: 12 November 2012).

Curran, K., Millar, A. and Mc Garvey, C. (2012), 'Near field communication', *International Journal of Electrical Computer Engineering (IJECE)* **2**(3), 371–382.

Davi, L., Dmitrienko, A., Liebchen, C. and Sadeghi, A. (2012), Over-the-air cross-platform infection for breaking mTAN-based online banking authentication, *in* '*Black-Hat Abu Dhabi 2012*', *Abu Dhabi*.

Delany, S., Buckley, M. and Greene, D. (2012), 'SMS spam filtering: Methods and data', *Expert Systems with Applications* **39**(10), 9899–9908.

Dietrich, C. J., Rossow, C., Freiling, F., Bos, H., van Steen, M. and Pohlmann, N. (2011), On botnets that use dns for command and control, *in* '*Proceedings of European Conference on Computer Network Defence(EC2ND)*', *Gothenburg, Sweden*, pp. 9–16.

Eamrurksiri, T. and Xiang, A. (2012), Near field communication, TSKS03 Wireless System Report.

Eldridge, A. (2013), 'Mobile malware gains ground in 2013'. Available online: [http://www.callnerds.com/mobile-malware-2013/] (Accessed: 2 July 2013).

Elliott, C. (2010), 'Botnets: To what extent are they a threat to information security?', *Information Security Technical Report* **15**(3), 79–103.

Enck, W., Ongtang, M. and McDaniel, P. (2009), On lightweight mobile phone application certification, *in* '*Proceedings of the 16th ACM conference on Computer and Communications Security*', *Chicago, IL, USA*, pp. 235–245.

Evolved Threats in a Post-PC World (2013), Technical report, Trend Micro Incorporated.

Faghani, M. R. and Nguyen, U. T. (2012), Socellbot: A new botnet design to infect smartphones via online social networking, *in* '*25th IEEE Canadian Conference on Electrical and Computer Engineering*', *Montreal, Canada*, pp. 1–5.

Falliere, N. and Chien, E. (2009), Zeus: King of the bots, Technical report, Symantec Corporation.

Farley, T. (2005), 'Mobile telephone history', *Telektronikk* **101**(3/4), 22.

Fedynyshyn, G., Chuah, M. and Tan, G. (2011), Detection and classification of different botnet C&C channels, *in* '*Proceedings of the 8th International Conference on Autonomic and Trusted Computing*', *Banff, Canada*, pp. 228–242.

Feily, M., Shahrestani, A. and Ramadass, S. (2009), A survey of botnet and botnet detection, *in* '*Third International Conference on Emerging Security Information, Systems and Technologies, SECURWARE'09*', *Athens/Glyfada, Greece*, pp. 268–273.

Ferguson, R. (2010), The botnet chronicles: A journey to infamy, Technical report, Trend Micro Incorporated.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and Berners-Lee, T. (1999), 'Hypertext transfer protocol'. Available online: [http://www.ietf.org/rfc/rfc2616.txt] (Accessed: 4 July 2012).

Fisher, D. (2012), 'SMSZombie malware infecting Android devices, stealing money'. Symantec. Available online: [http://threatpost.com/en_us/blogs/smszombie-malware-infecting-android-devices-stealing-money-082012] (Accessed: 5 October 2012).

Flø, A. and Jøsang, A. (2009), Consequences of botnets spreading to mobile devices, *in* '*Short-Paper Proceedings of the 14th Nordic Conference on Secure IT Systems (NordSec 2009)*', *Oslo, Norway*, pp. 37–43.

Franklin, E. (2013), 'Arm demos Samsung's next major CPU, the Exynos 5 Octa'. Cnet. Available online: [http://reviews.cnet.com/8301-13970_7-57571315-78/arm-demos-samsungs-next-major-cpu-the-exynos-5-octa/] (Accessed: 6 March 2013).

Froehlich, J., Chen, M., Consolvo, S., Harrison, B. and Landay, J. (2007), MyExperience: A system for in situ tracing and capturing of user feedback on mobile phones, *in* '*Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*', *San Juan, Puerto Rico*, pp. 57–70.

Fuentes, D. (2010), 'Trojan horses in mobile devices', *Computer Science and Information Systems* **7**(4), 813–822.

Garg, K. and Chawla, R. (2011), 'Detection of DDOS attacks using data mining', *International Journal of Computing and Business Research* **2**(1).

Gartner (2013), 'Gartner says worldwide mobile phone sales declined 1.7 percent in 2012'. Available online: [http://www.gartner.com/newsroom/id/2335616] (Accessed: 4 March 2013).

Geer, D. (2005), 'Malicious bots threaten network security', *Computer* **38**(1), 18–20.

Gehrmann, C. and Nyberg, K. (2001), Enhancements of Bluetooth baseband security, *in* '*Proceedings of Nordsec 2001*', *Copenhagen, Denmark*, pp. 191–230.

Geng, G., Xu, G., Zhang, M., Guo, Y., Yang, G. and Wei, C. (2012), 'The design of SMS based heterogeneous mobile botnet', *Journal of Computers* **7**(1), 235–243.

Geng, G., Xu, G., Zhang, M., Yang, Y. and Yang, G. (2011), An improved SMS based heterogeneous mobile botnet model, *in* '*International Conference on Information and Automation (ICIA)*', *Shenzhen, China*, pp. 198–202.

Giroire, F., Chandrashekar, J., Taft, N., Schooler, E. and Papagiannaki, D. (2009), 'Exploiting temporal persistence to detect covert botnet channels', *Recent Advances in Intrusion Detection* **5758**, 326–345.

Godwin-Jones, R. (2003), 'Emerging technologies', *Language Learning & Technology* **7**(2), 12–16.

Goel, S., Baykal, A. and Pon, D. (2006), 'Botnets: The anatomy of a case', *Journal of Information Systems Security (JISSEC)* **1**(3), 45–60.

Google's Approach to IT Security (2012), Technical report, Google.

Grizzard, J., Sharma, V., Nunnery, C., Kang, B. and Dagon, D. (2007), Peer-to-peer botnets: Overview and case study, *in* '*Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets*', *Cambridge, MA, USA*, p. 1.

Gupta, N. (2002), Improving the effectiveness of deceptive honeynets through an empirical learning approach, *in* '*3rd Australian Information Warfare & Security Conference*', *Perth, Western Australia*.

Haataja, K. and Toivanen, P. (2008), Practical man-in-the-middle attacks against Bluetooth secure simple pairing, *in* '*4th International Conference on Wireless Communications, Networking and Mobile Computing*', *Dalian, China*, pp. 1–5.

Hager, C. and Midkiff, S. (2003), Demonstrating vulnerabilities in Bluetooth security, *in* '*Global Telecommunications Conference, GLOBECOM'03*', *San Francisco, USA*, pp. 1420–1424.

Hall, S. and Anderson, E. (2009), 'Operating systems for mobile computing', *Journal of Computing Sciences in Colleges* **25**(2), 64–71.

Hayes, B. (2008), 'Cloud computing', *Communications of the ACM* **51**(7), 9–11.

Hinson, J. (2010), Code white: A signed code protection mechanism for smartphones, Master's thesis, Graduate School of Engineering and Management, Air Force Institute of Technology, Air University.

Holzer, A. and Ondrus, J. (2011), 'Mobile application market: A developer's perspective', *Telematics and Informatics* **28**(1), 22–31.

Honan, M. (2009), 'From brick to stick: A history of mobile devices'. Wired. Available online: [http://www.wired.com/gadgets/gadgetreviews/multimedia/2009/02/gallery_cell_phone_history?slide=1&slideView=3] (Accessed: 16 November 2012).

Hong, J. (2012), 'The state of phishing attacks', *Communications of the ACM* **55**(1), 74–81.

Hua, J. (2012), A study on anomaly-based countermeasures against malware constructing botnets, PhD thesis, Graduate School of Information Science and Electrical Engineering, Kyushu University.

Hua, J. and Sakurai, K. (2011), A SMS-based mobile botnet using flooding algorithm, *in* '*Proceedings of the 5th IFIP WG 11.2 International Conference on Information Security Theory and Practice: Security and Privacy of Mobile Devices in Wireless Communication*', *Crete, Greece*, pp. 264–279.

Hua, J. and Sakurai, K. (2012), 'Botnet command & control based on short message service and human mobility', *Computer Networks: The International Journal of Computer and Telecommunications Networking* **57**(2), 579–597.

Hughes, L. and DeLone, G. (2007), 'Viruses, worms, and Trojan horses: Serious crimes, nuisance, or both?', *Social Science Computer Review* **25**(1), 78–98.

Hypponen, M. (2006), 'Malware goes mobile', *Scientific American* **295**(5), 70–77.

iPhone OS Technology Overview (2009), Technical report, Apple Inc.

Jakobsson, M. and Wetzel, S. (2001), Security weaknesses in Bluetooth, *in* '*Proceedings of the 2001 Conference on Topics in Cryptology: The Cryptographer's Track at RSA*', *San Francisco, USA*, pp. 176–191.

Kim, M. (n.d.), 'Botnet detection and response technology'. 2nd EU-Korea Cooperation Forum on ICT Research. Available online: [http://www.eurosouthkorea-ict.org/documents/forum2_ppt/mijoo_kim.pdf] (Accessed: 6 July 2012).

Kismet (n.d.), 'Kismet'. Available online: [http://www.kismetwireless.net/] (Accessed: 7 October 2012).

Koblentz, E. (2011), How it started: Mobile Internet devices of the previous millennium, *in* '*Human-Computer Interaction and Innovation in Handheld, Mobile and Wearable Technologies*', *IGI Global*, pp. 172–174.

La Polla, M., Martinelli, F. and Sgandurra, D. (2012), 'A survey on security for mobile devices', *Communications Surveys & Tutorials* **15**(1), 1–26.

Lawton, G. (2008), 'Is it finally time to worry about mobile malware?', *Computer* **41**(5), 12–14.

Leavitt, N. (2005), 'Mobile phones: The next frontier for hackers?', *Computer* **38**(4), 20–23.

Leder, F., Werner, T. and Martini, P. (2009), 'Proactive botnet countermeasures - an offensive approach', *The Virtual Battlefield: Perspectives on Cyber Warfare 3* pp. 211–225.

Lee, J., Jeong, H., Park, J., Kim, M. and Noh, B. (2008), The activity analysis of malicious HTTP-based botnets using degree of periodic repeatability, *in* '*International Conference on Security Technology, SECTECH'08*', *Hainan Island, China*, pp. 83–86.

Lesk, M. (2007), 'The new front line: Estonia under cyberassault', *IEEE Security and Privacy* **5**(4), 76–79.

Lewis, J. (1996), Reaping the benefits of modern usability evaluation: The Simon story, *in* '*Advances in Applied Ergonomics: Proceedings of the 1st International Conference on Applied Ergonomics (ICAE'96)*', *Istanbul*, pp. 752–757.

Lewis, T. (1998), 'Information appliances: Gadget netopia', *Computer* **31**(1), 59–68.

Li, Y., Zhai, L., Wang, Z. and Ren, Y. (2013), Control method of Twitter-and SMS-based mobile botnet, *in* '*International Standard Conference on Trustworthy Computing and Service, ISCTCS 2013*', *Beijing, China*, pp. 644–650.

Lin, F. and Ye, W. (2009), Operating system battle in the ecosystem of smartphone industry, *in* '*International Symposium on Information Engineering and Electronic Commerce, IEEC'09*', *Ternopil, Ukraine*, pp. 617–621.

Liu, C., Lu, W., Zhang, Z., Liao, P. and Cui, X. (2011), A recoverable hybrid C&C botnet, *in* '*6th International Conference on Malicious and Unwanted Software (MALWARE)*', *Fajardo, USA*, pp. 110–118.

Lookout Mobile Threat Report (2011), Technical report, Lookout Mobile Security.

Mah, B. (1997), An empirical model of HTTP network traffic, *in* '*Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM'97*', *Kobe, Japan*, pp. 592–600.

Maheshwari, N. (2012), Botnets - secret puppetry with computers, Technical report, Department of Computer Science, University of Arizona.

Mansfield-Devine, S. (2012*a*), 'Android malware and mitigations', *Network Security* **2012**(11), 12–20.

Mansfield-Devine, S. (2012*b*), 'Paranoid Android: Just how insecure is the most popular mobile platform?', *Network Security* **2012**(9), 5–10.

Maslennikov, D. (2013), 'Mobile malware evolution: Part 6'. Securelist. Available online: [http://www.securelist.com/en/analysis/204792283/Mobile_Malware_Evolution_Part_6] (Accessed: 6 March 2013).

Maymounkov, P. and Mazieres, D. (2002), Kademlia: A peer-to-peer information system based on the xor metric, *in* '*First International Workshop on Peer-to-Peer Systems*', *Cambridge, MA, USA*, pp. 53–65.

Mell, P. and Grance, T. (2011), The NIST definition of cloud computing, Technical report, National Institute of Standards and Technology.

Meurant, R. (2010), The iPad and EFL digital literacy, *in* '*Signal Processing and Multimedia*', *Jeju Island, Korea*, pp. 224–234.

Minar, N. and Tarique, M. (2012), 'Bluetooth security threats and solutions: A survey', *International Journal of Distributed and Parallel Systems (IJDPS)* **3**(1), 86–94.

Mobile Threat Report Q3 2012 (2012), Technical report, F-Secure.

Mockapetris, P. (1987), 'Domain names - implementation and specification'. Available online: [http://www.ietf.org/rfc/rfc1035.txt] (Accessed: 4 July 2012).

Mockapetris, P. and Dunlap, K. (1988), 'Development of the domain name system', *ACM* **18**(4), 123–133.

Moran, J. (2006), College student's acceptance of tablet personal computers: A modification of the unified theory of acceptance and use of technology model, PhD thesis, Capella University.

Mukamurenzi, N. (2008), Storm Worm: A P2P botnet, Master's thesis, Norwegian University of Science and Technology.

Mulliner, C. (2010), Smartphone botnets, *in* '*Proceedings of the Sixth GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING), Technical Report SR-2011-01*', *Bochum, Germany*, p. 10.

Mulliner, C. (2011), On the impact of the cellular modem on the security of mobile phones, PhD thesis, Technical University Berlin.

Mulliner, C. and Seifert, J. (2010), Rise of the ibots: Owning a telco network, *in* '*Proceedings of the 5th IEEE International Conference on Malicious and Unwanted Software (Malware)*', *Nancy, France*, pp. 71–80.

Okada, T. (2005), Youth culture and the shaping of Japanese mobile media: personalization and the Keitai Internet as multimedia, *in* '*Personal, Portable, Pedestrian: Mobile Phones in Japanese Life*', *The MIT Press*, pp. 41–60.

Panse, T. and Kapoor, V. (2012), 'A review on security mechanism of Bluetooth communication', *International Journal of Computer Science and Information Technologies* **3**(2), 3419–3422.

Pfleeger, C. and Pfleeger, S. (2006), *Security in Computing*, 4th edn, Prentice Hall.

Phan, R. and Mingard, P. (2012), 'Analyzing the secure simple pairing in Bluetooth v4.0', *Wireless Personal Communications* **64**(4), 719–737.

Porras, P., Saidi, H. and Yegneswaran, V. (2010), An analysis of the ikee.B iPhone botnet, *in* '*Security and Privacy in Mobile Information and Communication Systems*', *Catania, Sicily*, pp. 141–152.

Press, L. (1992), 'Dynabook revisited - portable computers past, present and future', *Communications of the ACM* **35**(3), 25–32.

Provos, N., McNamee, D., Mavrommatis, P., Wang, K. and Modadugu, N. (2007), The ghost in the browser analysis of web-based malware, *in* '*Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets*', *Cambridge, MA, USA*, p. 4.

Purewal, S. (2012), 'Digital pickpockets can dip into Google Wallet'. Available online: [http://www.itbusiness.ca/news/digital-pickpockets-can-dip-into-google-wallet/17250] (Accessed: 12 October 2012).

Ramzan, Z. and Wüest, C. (2006), Phishing attacks: Analyzing trends in 2006, *in* '*Fourth Conference on Email and Anti-spam*', *California, USA*.

Ripeanu, M. (2001), Peer-to-peer architecture case study: Gnutella network, *in* '*First International Conference on Peer-to-Peer Computing*', *Linkoping, Sweden*, pp. 99–100.

Sairam, K., Gunasekaran, N. and Redd, S. (2002), 'Bluetooth in wireless communication', *Communications Magazine* **40**(6), 90–96.

Sarwar, U., Ramadass, S. and Budiarto, R. (2007), A framework for detecting Bluetooth mobile worms, *in* '*Telecommunications and Malaysia International Conference on Communications. ICT-MICC 2007*', *Penang, Malaysia*, pp. 343–347.

Schaefer, T., Höfken, H. and Schuba, M. (2011), Windows phone 7 from a digital forensics perspective, *in* '*Digital Forensics and Cyber Crime*', pp. 62–76.

Schedeen, J. (2010), 'The history of the tablet PC'. IGN. Available online: [http://www.ign.com/articles/2010/04/01/the-history-of-the-tablet-pc] (Accessed: 13 November 2012).

Schiller, C. (2007), *Botnets: The Killer Web App*, Syngress.

Schoof, R. and Koning, R. (2007), 'Detecting peer-to-peer botnets'. University of Amsterdam.

Serazzi, G. and Zanero, S. (2004), Computer virus propagation models, *in* '*Tutorials of the 11th IEEE and ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems, MASCOTS'03*', *Orlando, Florida, USA*, pp. 26–50.

Sharples, M. and Beale, R. (2003), 'A technical review of mobile computational devices', *Journal of Computer Assisted Learning* **19**(3), 392–395.

Sheu, Y., Hsu, F., Hwang, Y., Jiang, R., Jhang, J. and Huang, P. (2013), 'JokerBot - an Android-based botnet', *Applied Mechanics and Materials* **284**, 3454–3458.

Shuai, W., Xiang, C., Peng, L. and Dan, L. (2013), S-URL flux: A novel C&C protocol for mobile botnets, *in* '*International Standard Conference on Trustworthy Computing and Service, ISCTCS 2013*', *Beijing, China*, pp. 412–419.

Simão, A., Sícoli, F., Melo, L., Deus, F. and Sousa Júnior, R. (2011), 'Acquisition and analysis of digital evidence in Android smartphones', *The International Journal of Forensic Computer Science* **6**(1), 28–43.

Singh, K., Sangal, S., Jain, N., Traynor, P. and W., L. (2010), Evaluating Bluetooth as a medium for botnet command and control, *in* '*7th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*', *Bonn, Germany*, pp. 61–80.

sKyWIper Analysis Team (2012), skywiper (a.k.a. flame a.k.a. flamer): A complex malware for targeted attacks, Technical Report v1.05, Laboratory of Cryptography and System Security (CrySyS Lab), Budapest University of Technology and Economics.

Soitinaho, J. (2007), 'Security threats of mobile service user', *TKK T-110.5290 Seminar on Network Security, Helsinki University of Technology* .

Souppouris, A. (2013), 'Nvidia unveils Tegra 4, 'world's fastest mobile processor''. The Verge. Available online: [http://www.theverge.com/2013/1/6/3844860/nvidia-tegra-4-announcement-specs-availability] (Accessed: 6 March 2013).

Speckmann, B. (2008), The Android mobile platform, PhD thesis, Eastern Michigan University.

Spreitzenbarth, M. and Freiling, F. (2012), Android malware on the rise, Technical report, University of Erlangen, Germany.

Strazzere, T. and Wyatt, T. (2011), Geinimi Trojan technical teardown, Technical report, Lookout Mobile Security.

Subramony, A. (2011), 'A visual history of the cell phone'. Mac Life. Available online: [http://www.maclife.com/article/gallery/visual_history_cell_phone] (Accessed: 12 November 2012).

Tariq, M., Czerepinski, P., Nix, A., Bull, D. and Canagarajah, N. (2000), 'Robust and scalable matching pursuits video transmission using the Bluetooth air interface standard', *IEEE Transactions on Consumer Electronics* **46**(3), 673–681.

Tariq, Z. (2012), 'Smartphone processors: Some fundamentals'. Available online: [http://www.etechmag.com/2012/03/22/smartphone-processors-some-fundamentals.html] (Accessed: 12 October 2012).

Text4FreeOnline (2010), 'Text4freeonline'. Available online: [http://text4freeonline.com/] (Accessed: 4 March 2013).

The Role of DNS in Botnet Command & Control (2012), Technical report, Umbrella Security Labs Whitepaper.

Traynor, P., Lin, M., Ongtang, M., Rao, V., Jaeger, T., McDaniel, P. and La Porta, T. (2009), On cellular botnets: Measuring the impact of malicious devices on a cellular network core, *in* '*Proceedings of the 16th ACM Conference on Computer and Communications Security*', *Chicago, IL, USA*, pp. 223–234.

Trends for 2013 (2013), Technical report, ESET Latin America's Lab.

Tseng, Y., Ni, S., Chen, Y. and Sheu, J. (2002), 'The broadcast storm problem in a mobile ad hoc network', *Wireless Networks* **8**(2/3), 153–167.

Ubertooth (2010), 'Project ubertooth'. Available online: [http://ubertooth.sourceforge.net/] (Accessed: 7 October 2012).

Vochin, A. (2009), 'History of mobile phones'. Available online: [http://gadgets.softpedia.com/newsPDF/History-of-Mobile-Phones-3578.pdf] (Accessed: 12 November 2012).

Wang, P., Wu, L., Aslam, B. and Zou, C. (2009), A systematic study on peer-to-peer botnets, *in* '*Proceedings of 18th International Conference on Computer Communications and Networks, ICCCN*', *San Francisco, California, USA*, pp. 1–8.

Wang, W., Fang, B., Zhang, Z. and Li, C. (2009), A novel approach to detect irc-based botnets, *in* '*International Conference on Networks Security, Wireless Communications and Trusted Computing, NSWCTC'09*', *Wuhan, Hubei China*, pp. 408–411.

Wi-Fi Certified Wi-Fi Direct (2010), Technical report, Wi-Fi Alliance.

Wilson, S. and Kifayat, K. (2012), When the droid became the bot: Trends, threats and investigation of a mobile botnet, *in* '*The 13th Annual Post Graduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PG Net 2012)*', *Novotel, Liverpool*.

Wyke, J. (2011), What is zeus, Technical report, SophosLabs.

Xia, W., Li, Z., Chen, Z. and Yuan, Z. (2008), 'Commwarrior worm propagation model for smart phone networks', *The Journal of China Universities of Posts and Telecommunications* **15**(2), 60–66.

Xiang, C., Binxing, F., Lihua, Y., Xiaoyi, L. and Tianning, Z. (2011), Andbot: towards advanced mobile botnets, *in* '*Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats*', *Boston, USA*, p. 11.

Xu, K., Butler, P., Saha, S. and Yao, D. (2013), 'DNS for massive-scale command and control', *IEEE Transactions on Dependable and Secure Computing* **10**(3), 143–153.

Yan, G. (2006), Bluetooth worms: Models, dynamics, and defense implications, *in* '*Computer Security Applications Conference, 2006, ACSAC'06*', *Miami, Florida, USA*, pp. 245–256.

Zeidanloo, H. and Manaf, A. (2010), 'Botnet detection by monitoring similar communication patterns', *(IJCSIS) International Journal of Computer Science and Information Security* **7**(3), 36–45.

Zeng, Y. (2012), On detection of current and next-generation botnets, PhD thesis, The University of Michigan.

Zeng, Y., Shin, K. and Hu, X. (2012), Design of SMS commanded-and-controlled and P2P-structured mobile botnets, *in* '*WISEC '12 Proceedings of the Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks*', *Tucson, Arizona, USA*, pp. 137–148.

Zhao, S., Lee, P., Lui, J., Guan, X., Ma, X. and Tao, J. (2012), Cloud-based push-styled mobile botnets: A case study of exploiting the cloud to device messaging service, *in* '*Proceedings of the 28th Annual Computer Security Applications Conference, ACSAC'12*', *New Orleans, Louisiana, USA*, pp. 119–128.

Zhou, Y. and Jiang, X. (2012), Dissecting Android malware: Characterization and evolution, *in* '*2012 IEEE Symposium on Security and Privacy (SP)*', *San Francisco, California, USA*, pp. 95–109.

Zhu, Z., Lu, G., Chen, Y., Fu, Z., Roberts, P. and Han, K. (2008), Botnet research survey, *in* '*Proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference*', *Turku, Finland*, pp. 967–972.

Appendix A

Publications and Conference Presentations

- Pieterse, H. and Olivier, M.S. (2012), Android botnets on the rise: Trends and characteristics, in *'Information Security for South Africa', Johannesburg, South Africa*, pp. 1-5.
- Pieterse, H. (2012), A study of mobile botnets: analysis of attack strategies, in *'4th CSIR Biennial Conference: Real Problems Relevant Solutions', Pretoria, South Africa.*
- Pieterse, H. and Olivier, M. (2013), 'Design of a hybrid command and control mobile botnet', *Journal of Information Warfare* 12(1), 70-82.
- Pieterse, H. and Olivier, M. (2013), Design of a hybrid command and control mobile botnet, in *'The 8th International Conference on Information Warfare and Security', Denver, USA*, pp. 183-192.
- Pieterse, H. and Olivier M.S. (2014), 'Bluetooth Command and Control channel', *Computers and Security* Vol. 45, 75-83.

Appendix B

List of Abbreviations

Below is a list of all the abbreviations used in this work [1].

3G third generation

API application program interface

ARM Advanced RISC Machines

BYOD bring your own device

C&C command and control

C2DM cloud to device messaging

CitMo Carberp-in-the-Mobile

CPU central processing unit

DDoS distributed denial of service

DES data encryption standard

DHT distributed hash table

DNS domain name system

[1] Abbreviations based on the Chicago style.

DoS denial of service

GPS global positioning system

GSM global system for mobile communications

HLR home location register

HTML hyper text markup language

HTTP hyper text transfer protocol

IDS intrusion detection system

IMEI international mobile station equipment identity

IMSI international mobile subscriber identity

iOS iPhone operating system

IP internet protocol

IRC internet relay chat

ISO International Organization for Standardization

J2ME Java 2 platform, micro edition

JVM Java virtual machine

LAP lower address portion

LCD liquid-crystal display

LED light-emitting diode

MAC media access control

MANET mobile ad hoc network

MD5 message digest 5

MMS multimedia messaging service

mTAN mobile TAN

NAT network address translation

NAP non-significant address portion

NFC near field communication

NIST National Institute of Standards and Technology

P2P peer-to-peer

PCAP packet capture

PDA personal digital assistant

PIN personal identification number

OS operating system

OSN online social network

OSNMS online social network messaging service

RF radio frequency

RFCOMM radio frequency communication

RIM Research In Motion

SIG Special Industry Group

SIM subscriber identity module

SIS software installation script

SMS short message service

SMSC short message service center

SpitMo SpyEye-in-the-Mobile

SSH secure shell

SSP simple secure pairing

S-URL shorten URL

TAN transaction authentication number

TCP transmission control protocol

TXT text record

UAP upper address portion

US United States

URL uniform resource locator

USB universal serial bus

UUID universal unique identifier

VGA video graphics array

Wi-Fi Wireless Fidelity

ZitMo Zeus-in-the-Mobile