University of Twente

Master Thesis

ANONYMOUS AND HIDDEN COMMUNICATION CHANNELS: A PERSPECTIVE ON FUTURE DEVELOPMENTS

Author:
Erwin Middelesch
e.w.middelesch@alumnus.utwente.nl

Supervisors:
Dr.ir. Aiko Pras (UT)
Dr. Anna Sperotto (UT)
Ing. Sander Degen (TNO)

February 2015

ABSTRACT

It is general knowledge that several organisations have the capabilities to create hidden communication channels which can be used for data exfiltration or to control remote agents. These channels transmit data without the permission and knowledge of the channels owner. This thesis investigates the possible future evolution of these channels, how they might incorporate anonymity, and how they can be implemented. Several existing and novel communication channels are investigated and evaluated by means of set of requirements and several use cases. This results in the creation of a prototype, which is evaluated for its effectiveness to remain hidden. Finally we conclude that a combination of an anonymity protocol and a server based service shows the best results.

This thesis was completed as part of an internship at the Dutch organisation for applied scientific research (TNO).

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

This introduction chapter introduces the problem in section 1.1. Section 1.2 lists relevant background information. This is followed by a description of utilized research methods in section 2.1. Next the research question is presented in section 1.3. Finally, section 1.4 provides an overview of the structure of this thesis.

1.1 PROBLEM STATEMENT

It is public knowledge that several organisations actively engage in espionage via the Internet. These organisations have the capabilities to create and maintain hidden communication channels. That is, communication channels which facilitate the transmission of data without the knowledge of some of the parties involved. These channels can be used to exfiltrate data and to control entities remotely. Several examples of such hidden communication channels have been discovered and those responsible have been identified [1][2]. As international pressure increases it can be expected these organisations will increase their efforts to prevent detection and remain unidentified.

The goal of this thesis is to investigate how such communication channels might function, and how they can be implemented. This knowledge is then used to build a prototype communication system to prove the feasibility of such systems. Finally, results of this thesis can be used to improve detection techniques.

We start by gathering relevant background information. From this information we concluded that not all hidden communication channels can be used anonymously. A network structure was devised which circumvents this problem by introducing a proxy. Several hidden communication methods were determined to be compatible with this new network structure. Two of these communication methods are implemented into a prototype: cloud storage services and messaging protocols. The effectiveness of these methods is evaluated from which we determined that it is feasible to combine both anonymity and hidden communication into a communication channel.

1.2 BACKGROUND

There are several research subjects which are relevant to this thesis. This section will present a high-level overview on these relevant research subjects. Chapter 2 provides a

more detailed investigation of the state of the art, this was compiled as part of a report for the Research Topics[1] course.

One of these relevant subjects is botnet command and control (C&C) channels which are described in section 2.2.1. The control structure utilized by botnets has several characteristics in common with a hidden communication channel, as botnet communication is mainly used to control the bots, but can be used to exfiltrate information as well. Botnets rely on a C&C topology to ensure each bot receives its commands [3]. They can be controlled through covert means to avoid detection. As detection systems improve, the C&C methods become increasingly sophisticated to avoid detection. The methods present in modern day botnets may therefore provide helpful insights.

Advanced persistent threats (APTs) employ C&C structures as well. The most advanced APTs go unnoticed for years, this suggests that the covert channels used for their C&C communications are difficult to detect.

Not only botnets and APTs should be investigated, the countermeasures against botnets are relevant as well. Section 2.2.2 contains an overview of countermeasures. The existing detection methodologies can provide helpful insights on which C&C methods cannot circumvent modern detection methods. The knowledge of current detection methods can be applied to design a communication channel which will not be detected by these detection methods.

As botnets are a dynamic field, new countermeasures are devised continuously. This leads to the development of new hiding techniques, which are described in section 2.2.3. This induces a feedback loop of increasingly sophisticated hiding techniques and countermeasures. One information hiding technique is steganography, which is a method to hide information within other information [4]. This can be achieved through different means, data can be hidden in for example, images or video files, but in the headers of protocols such as TCP/IP as well [5].

Section 2.2.4 contains an overview of anonymity protocols as merely hiding the information is not enough. Protecting the identity of the user is of paramount importance when the hidden messages are discovered. Therefore some methods to remain anonymous have to be investigated. The scope of this investigation is limited to practical implementations, because the final goal is to create a functioning prototype.

1 Research Topics is a 10 credit literature research course

1.3 GOAL, RESEARCH QUESTIONS AND APPROACH

The goal of this thesis is to design and implement a hidden communication channel. Furthermore, the identity of the user has to be protected. This lead to the following research question:

MAIN RESEARCH QUESTION: CAN A COMMUNICATION CHANNEL PRESERVE THE ANONYMITY OF ITS USER AND REMAIN HIDDEN AT THE SAME TIME?

Answering this question is the goal of this thesis. It is divided into sub questions. The answers to the sub questions will provide the answer to the main research question.

SUB QUESTION 1: WHICH METHODS PROVIDE ANONYMITY ON THE INTERNET?

To determine which anonymity protocols are suitable solutions an overview of existing protocols has to be constructed. The protocols will be gathered by searching the relevant literature and state of the art. This overview will be used to evaluate the protocols and answer the third sub question.

SUB QUESTION 2: WHICH HIDDEN COMMUNICATION CHANNELS EXIST?

Just as the previous question, an overview of hidden communication channels must be compiled from the relevant literature and state of the art. An evaluation of these channels will result the channels which are suitable for the prototype. These can then in turn be used to answer the third sub question.

SUB QUESTION 3: WHICH ANONYMITY SOLUTIONS AND DATA HIDING TECHNIQUES CAN BE COMBINED?

It is unlikely that every anonymity protocol and hidden communication channel can function properly together. It is therefore important to investigate if and how these methods can be combined. The resulting knowledge will be used to implement a functional prototype. This prototype will be evaluated to determine if anonymity and covertness can be combined.

1.4 THESIS STRUCTURE

CHAPTER 2: describes the relevant related work and background information.

CHAPTER 3: contains an overview of the requirements and use cases.

CHAPTER 4: some solutions are analysed in this chapter.

CHAPTER 5: describes the design of the communication channel.

CHAPTER 6: the the channel is evaluated in this chapter.

CHAPTER 7: discusses the results.

CHAPTER 8: contains the conclusion.

2 STATE OF THE ART

This chapter contains an overview of the state of the art of the relevant topics. This knowledge was gathered as part of a report for the Research Topics course. The information presented in this chapter is meant to provide a detailed overview of the current state of the art for those who are interested in command and control channels, botnet countermeasures, hidden and covert channels, and anonymous communication.

2.1 RESEARCH METHODS

Hidden communication channels are not, per se, a novel research subject. Malware developers have used hidden C&C methods extensively to control agents. Relevant literature on this subject is therefore abundant. Searching for *Command and Control* in combination with *Malware*, *Botnet*, and *Advanced Persistent Threat* will provide an overview of practical and theoretical C&C methodologies. The keywords *traffic*, *measurements*, and *detection* might result in in-depth discussions on these subjects.

Covert channels are another field of research relating to hidden communication. The field of covert channels is particularly abundant with literature. In order to find relevant literature we have searched for *hidden channel* and *covert channel*. The references in these papers are worth investigating as well. Furthermore, other papers with references to these papers might yield interesting literature.

Anonymous communication on the internet has been a hot topic in recent years. To find the relevant literature, the keywords *anonymous/anonymity* and *protocols*, *internet*, and *communication* will probably yield the most important results. The references in these papers can then be utilized to find other relevant literature.

2.2 OVERVIEW

This section contains an organized overview of the state of the art concerning the following subjects: botnet C&C architectures, botnet countermeasures, covert channels, and anonymous communication. Section 2.3 contains a complete overview of the related work.

2.2.1 *Command and Control*

The C&C protocol is often the only way a botnet can be controlled. It is therefore important that the protocol is robust, efficient, and fast. Over the years a number of different topologies and network protocols have been used for C&C. The following sections will describe these in detail.

2.2.1.1 *Botnet topologies*

There are different topologies present in botnets. Each of these topologies has distinct advantages and disadvantages [3]. The details of each topology is described below.

CENTRAL

A central topology consists of a central point which relays messages to the members of a botnet as shown in figure 1. Bots initiate the exchange of commands by connecting to this central point.

The most common protocols used by this topology are IRC and HTTP. IRC allows the botmaster to send commands to each connected bot simultaneously, thus providing a fast and reliable C&C channel. Websites can be used to communicate with bots over HTTP. Bots can visit the website at regular intervals to receive new commands. Both protocols allow the bots to send information to the botmaster as well.

This topology has several disadvantages as well. Scalability is a limiting factor for the size of the botnet. There is simply a limit on the number of bots which can be controlled at the same time. This topology also has a single point of failure. Capture of a single bot by security researchers will reveal the location of the central point, unless an anonymizing service has been used. The botmaster may loose control of the botnet if the central point is disabled or otherwise compromised [6][7].

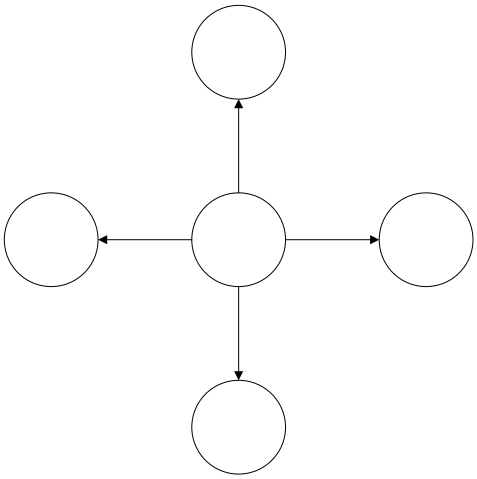

Figure 1: Central topology

PEER-TO-PEER

A peer-to-peer (P2P) botnet lacks a central point. Bots act as both client and server as they relay commands they receive to other bots. The structure is displayed in figure 2

There are two methods botnets can use for the communication between bots, push and pull. If a botnet implements push, bots will push new commands to each bot in their peer list. These bots will then push to the bots in their peer list and so on. This ensures bots only communicate when it is necessary, but bots need to store a peer list. If the botnet uses pull, the bots will regularly connect to one of its peers to check for new commands. Each individual bot has only contact with a small subsection of the botnet, therefore an adversary cannot deduce the extend of the botnet from a captured bot [8].

P2P botnets have been known to abuse existing P2P networks, e.g. Overnet, Waste, Kademlia. But custom implementations have been found as well [3].

P2P botnets often employ strong encryption, the distributed nature makes P2P botnets very robust and scalable, they create less disturbance on the internet and are therefore more difficult to notice.

P2P botnets have disadvantages as well, they are more complex to implement and maintain, commands take longer to propagate through the network, and obtaining data from the bots is not trivial.

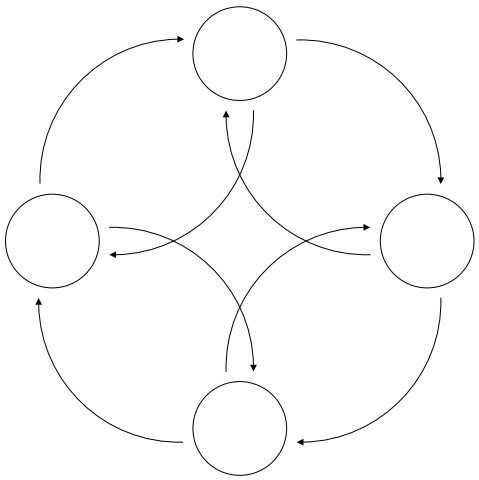

Figure 2: Peer-to-peer topology

HYBRID

In a hybrid C&C topology botnets are divided into two categories; client and servent bots. Figure 3 illustrates this structure. The client bots behave like ordinary peer-to-peer bots. Each client bot has a fixed list of servent bots to contact. The clients contact these servent bots to obtain new commands. Servent bots on the other hand act as both client and server. They actively contact other servent bots for new commands and relay these commands to client bots. Each servent bot generates its own symmetric key to encrypt incoming traffic.

Bots first determine whether they can be contacted reliably from the internet on a static IP address. If this is the case the bot will take on the role of a servent bot, otherwise it will be a client bot [9].

The fixed peer list ensures a captured bot will not leak much information about the botnet. A botmaster can decide it is necessary to update the peer lists, this might be necessary if the structure of the botnet changes drastically.

This approach has a downside as well, it relies heavily on bots with a static IP. These are relatively rare, consumer PCs are ordinarily behind a NAT or firewall. A low number of servent bots effectively means the topology of the botnet is centralized [9].

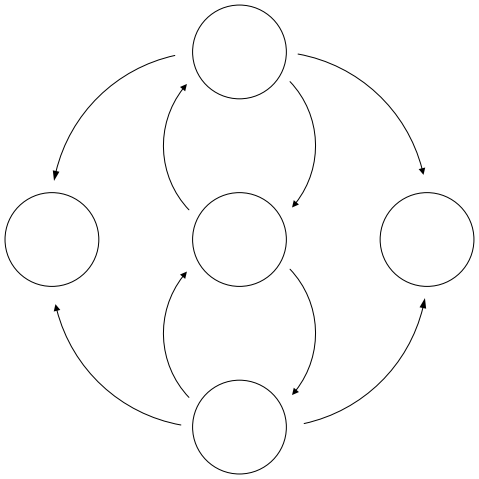

Figure 3: Hybrid topology

UNSTRUCTURED

The unstructured topology functions as follows: each bot knows exactly one other bot, as shown in figure 4. To issue a command, the botmaster searches for a bot and sends the command. The command then propagates along the chain of bots. This makes it difficult to map the botnet, as capturing a bot will only reveal one other bot. But it makes it less robust as well, each time a bot is unavailable the link is severed until the bots reconnect. Furthermore it takes a long time for commands to propagate through the entire network [10].

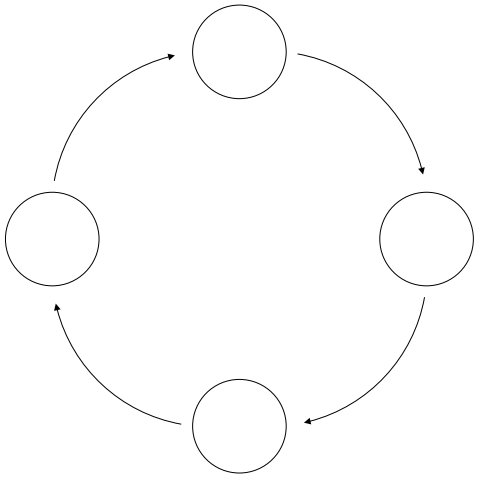

Figure 4: Unstructured topology

2.2.1.2 *C&C Protocols*

Different protocols have been used by botnets in the past. They will be discussed below.

IRC
IRC has been used as a C&C protocol for quite some time. It is a real-time messaging protocol, clients connect to a channel and receive all messages broadcast on this channel. This is useful for a C&C protocol, as the botmaster on has to send a command once and it is propagated to every bot simultaneous. It is easy to detect however, even more so if the commands are encrypted [11].

HTTP
HTTP C&C traffic is hard to detect. The traffic can be hidden in ordinary web traffic and will therefore bypass firewalls easily. The traffic can be encrypted as well (HTTPS) without raising suspicion, thus circumventing packet inspection methods [11].

DNS
DNS has been used as a C&C protocol as well. The bots query a DNS server, which returns a DNS TXT record with an encrypted payload. This is detectable, but there are

ways to hide the messages so detection is infeasible. DNS is rarely blocked or filtered, therefore it is suitable for C&C traffic. This approach as a major disadvantage however, the botmaster looses control of the botnet if the DNS server is compromised [12].

SIP

The Session Initiation Protocol (SIP) allows two parties to establish a direct connection via the SIP network. The caller first connects to his SIP proxy, the proxy connects to the callee's SIP proxy, which in turn connects to the callee. A direct connection between the caller and callee is then established. This is particularly useful for botnets, direct connections between bots can be established even if one or both of the parties is behind a NAT or firewall [13].

2.2.1.3 *Services*

Several services and services have been used by botnets to relay C&C messages. A service is a communication channel which is operated by a third party.

P2P NETWORK

An existing P2P network has several advantages; the protocol has been tested, the botmaster does not have to worry about reconnecting lost bots, and the C&C traffic is hidden among legitimate traffic.
There are several P2P networks which have been infiltrated by botnets, Overnet, Waste, and Kademlia [3]. These file sharing protocols allow the bots to exchange messages while at the same time pretending to be legitimate users [8].

SOCIAL NETWORKS

Several botnets have been discovered which use social networks to relay C&C messages. The messages can be encrypted strings send as plain text messages. For example a Twitter account could be used to tweet this encrypted message. An other approach is to use steganography. The message is hidden in an image, this image is then shared via the social network. This approach makes the C&C traffic hard to detect, as it would require investigating every suspicious image shared via the network [14] [15].

SKYPE

A theoretical method of building a C&C network over Skype has been proposed. The Skype API can be used to send and receive messages over the Skype network. Because

Skype messages are encrypted it is impossible to determine whether a message is legitimate or not. Because of Skype's widespread use, it can bypass defensive measures such as firewalls [16].

EMAIL

Email is in theory suitable to carry botnet C&C messages as well. The commands can be hidden in spam messages. Bots can then automatically extract the commands from the emails if the machine has internet access. Singh et al. [17] have shown that it is feasible. Furthermore, they have shown that even if the email provider knows about messages embedded into spam, it is still computationally infeasible to check every message.

2.2.1.4 *Overview of related work*

Table 1 contains an overview of C&C subjects discussed by specific papers. This table provides insights in which aspects of botnet C&C methods are focussed on by the scientific community.

The table shows the most commonly discussed topologies are the central and P2P topologies. Hybrid and unstructured topologies are relatively more recent and are less often investigated.

Table 1: Command and control literature overview

	[3]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[17]	[18]	[16]	[19]	[20]	[21]	[22]
Central	X	X	X			X	X					X			X	X	X	
Peer-to-peer	X	X	X	X	X	X	X		X		X		X	X				X
Hybrid	X				X						X							
Unstructured	X	X	X			X												
IRC	X	X	X				X	X										
HTTP			X				X	X			X					X	X	
DNS	X				X			X							X			
SIP									X									
P2P Networks	X		X	X	X		X											X
Social Networks	X									X	X		X				X	
Skype														X				
Email	X											X						

2.2.2 *Botnet Countermeasures*

Because botnet C&C architectures are investigated for suitable communication methods, it is prudent to investigate botnet countermeasures. This provides insight into the functioning of botnet countermeasures, and how these countermeasures can be avoided.

2.2.2.1 *Botnet detection approaches*

Feily et al. [23] describe four botnet detection techniques: signature-based, anomaly-based, DNS-based, and mining-based.

Signature-based detection utilizes signatures, known traffic or instruction patterns, of botnets. Therefore this detection method is not useful for botnets whose signature is unknown. Anomaly-based detection checks for network traffic anomalies. This facilitates the detection of unknown botnets. DNS-based detection applies anomaly detection algorithms to DNS traffic. Mining-based detection uses machine learning, classification, and clustering to detect botnet C&C traffic.
They show that some of these detection methods are able to detect botnets regardless of botnet protocol and structure.

Holz et al. [24] demonstrated a method to analyse and disrupt P2P botnets. They examine the Storm Worm botnet and present methods to disrupt its communication channel.

A novel way of detecting stealthy P2P botnets has been proposed by Zhang et al. [25]. Statistical fingerprints are applied to identify different types of P2P traffic, which facilitates the distinction of botnet traffic from legitimate traffic. This allows the detection of C&C traffic even if legitimate P2P traffic is used in conjunction with botnet traffic.

SIGNATURE-BASED

Signatures of known botnets can be used to detect bots. These signature-based detection methods apply rulesets for each specific botnet to detect bots. This allows for efficient detection of known botnets, but unknown botnets cannot be detected [23].

ANOMALY-BASED

Anomaly-based detection focusses on network traffic anomalies. These traffic anomalies are, for example, high network latency, high traffic volumes, or traffic on unusual ports. This method can detect an unknown botnet, if the bots in question produce network traffic anomalies. If the bots are dormant and waiting for commands, detection using this method is unlikely [23].

DNS-BASED

DNS-based detection is similar to anomaly-based detection, i.e. it applies anomaly detection to DNS traffic to detect bots. Bots within a centralized botnet typically connect to a C&C server. To reach this server the bots may perform DNS queries. This may cause distinctive patterns in DNS traffic, which can be detected. Unknown botnets can be detected, as the details of the botnet do not matter. However, this approach only works if the botnet uses a centralized C&C server which uses a domain name [23].

MINING-BASED

Some kinds of botnet C&C traffic are similar to ordinary traffic. Anomaly-based detection methods will not work in these cases. Mining-based solutions were created to detect these kinds of C&C traffic. These solutions use data mining and machine learning techniques to detect botnet C&C traffic [23].

INFILTRATION

Infiltration is an effective method to disrupt botnets [9]. The attacker joins the C&C channel and sends his own commands to the other bots, thus gaining control of the network.

2.2.2.2 *Botnet detection systems*

Several botnet detection systems have been designed by the scientific community, these will be discussed below.

BOTSNIFFER

BotSniffer is a botnet detection system which uses network anomalies to detect bots. It depends heavily on the protocol and network structure used by the botnet C&C method. It can only detect botnets with a centralized topology which use IRC or HTTP. However it can detect very small botnets, and it has demonstrated a low false positive rate [26].

BOTMINER

Botminer is another botnet detection method that employs anomaly detection. It does not rely on botnet protocols or network topologies, as it assumes that bots within a botnets share the same network traffic characteristics. This allows for low false positive detection of IRC, HTTP, and P2P based C&C traffic [27].

DISPATCHER

Dispatcher is a automatic protocol reverse-engineering tool. It analyses botnet binaries to extract the C&C protocol. The tool can successfully extract the C&C protocol even if code obfuscation or traffic encryption was used. This allows one to take control of botnets by sending reverse-engineered C&C messages [28].

PROVEX

Botnets nowadays prevalently encrypt C&C traffic. This increases the difficulty of botnet detection by applying payload signatures, or makes it impossible. However, if a static key was used, and the key is known, the traffic can be decrypted. ProVeX exploits this by decrypting the payloads of possible botnet traffic with known botnet keys. As it would be cumbersome to specify the C&C protocol semantics, probabilistic functions are used to determine if a payload contains C&C traffic. This is an inefficient approach, as all traffic is decrypted. But it still achieves reasonable performance [29].

2.2.2.3 *Overview of related work*

Table 2 presents an overview of methods to counteract botnets. The table shows most detection methods focus on signature and anomaly detection. A C&C method is therefore more likely to remain undetected if these detection techniques are avoided.

Table 2: Botnet countermeasures literature overview

	[3]	[9]	[10]	[6]	[7]	[8]	[12]	[14]	[30]	[31]	[23]	[26]	[27]	[28]	[29]	[24]	[25]	[22]	[32]
Signatures	X		X	X	X			X	X	X	X				X		X	X	
Anomalies	X		X	X		X	X	X	X	X	X	X	X		X		X	X	
DNS	X		X				X			X	X							X	X
Mining	X		X						X	X	X								
Infiltration	X	X	X			X								X		X			

2.2.3 *Hidden and Covert Channels*

Information hiding is a diverse field. Steganography and covert channels are both ways to hide information, but are radically different approaches. Both share a property, the hiding capacity. This is the maximum rate at which hidden information can reliably communicated over a medium [33].

2.2.3.1 *Protocols*

Network protocols provide numerous options for covert channels. For example, unused header bits, checksum fields and timestamp fields [34]. Some possible covert channels in network protocols are discussed below.

TCP/IP
Placing data in the TCP/IP header is easy, numerous flags and fields are not commonly used today. This does not mean it is undetectable however. It is possible to differentiate modified headers from ordinary ones. [5] therefore proposed to encode the data into initial sequence numbers (ISN). By applying this method the modified headers are virtually indistinguishable from ordinary headers.

Covert channels exist in IPv6 as well. Lucena et al. [35] have identified 22 different covert channels. They proposed methods to mitigate these covert channels, but were not able to block all of them.

DNS
DNS can be used as a covert channel as well. The DNS ID can be chosen by the client and can therefore be used to send information. Encrypted information cannot used directly however, Altalhi et al. [36] have shown this is detectable. They propose therefore to apply steganography to the encrypted information, this ensures the distribution of hidden data is comparable to normal DNS IDs.

BITTORRENT
Bittorrent is a P2P file sharing protocol. The open nature and wide spread use makes it suitable for a covert channel. Users can specify a peer ID and IP address when connecting to a tracker. Other users can request this data from the tracker, thus establishing a

covert channel [37].

RTP

The Real Time Protocol (RTP) uses timestamps to determine the ordering of packets. Because the least significant bits are never required to determine the ordering of packets, they can be used to transmit information. RTP packets are transmitted often, therefore even a couple of bits of information per packet will provide a reasonable transmission rate [38].

2.2.3.2 *Steganography*

Steganography attempts to store information in such a way that its existence is hidden. Multiple carriers for hidden information have been devised over the years [4]. Several of those will be discussed below.

IMAGES

The properties of several image formats can be exploited to hide data. Bits are manipulated in specific locations, these manipulations are so subtle they are not noticeable by the human eye. The trade off between payload size and covertness is obvious. The more data is hidden in an image, the bigger the distortion and therefore the detectability [39].

DOCUMENTS

A covert channel can be established via digital documents. Hidden information can be added to Microsoft Word documents. The information is encoded, made invisible, and added at the ends of paragraphs. The information remains hidden if the document is viewed [40].
Microsoft PowerPoint documents provide room for covert information as well. Several meta data fields and other storage fields provide room for information. These fields are not checked by the application and can therefore be used to hide information [41].

WEBSITES

Websites can be used to send covert messages via hidden information. Because HTML tags are not case sensitive, they can be used to encode data. Each character can be either upper or lower case, thus one bit of information can be stored for each character. This covert channel can provide a reasonable bandwidth, but it is not robust. An

adversary who knows about this type of covert channel can detect it quite easily [42].

2.2.3.3 *Overview of related work*

Table 3 contains an overview on covert channel literature. This overview shows steganography and TCP/IP are most commonly researched.

Table 3: Covert channel literature overview

	[4]	[5]	[33]	[34]	[39]	[43]	[44]	[35]	[36]	[37]	[13]	[45]	[38]	[42]	[40]	[41]	[46]
Steganography	X	X	X	X	X		X		X				X				
TCP/IP		X		X		X	X	X				X					
Images	X				X												
DNS				X					X			X					X
Webpages														X			
Documents															X	X	
Other protocols				X		X				X	X	X	X				

2.2.4 *Anonymous Communication*

Anonymous communication on the internet has been a hot topic in recent years. IP addresses are unique routing addresses, as these addresses belong to specific Internet Service Providers it is possible to ascertain the identity of the person or persons who use a specific IP address. Several methods have been devised which provide anonymity by concealing the IP address. Some argue however that anonymous communication is only feasible for short periods of time. An attacker with enough time and resources will ultimately be able to identify an anonymous user. Anonymity systems always have some kind of edge which can be exploited [47].

2.2.4.1 *Anonymity terminology*

Anonymity in a broad sense encompasses several different terms: unlinkability, unobservability, and pseudonymity [48]. These terms are explained below.

ANONYMITY

Anonymity of a user is defined as: *the state of not being identified within the set of users, which is called the anonymity set*. The anonymity set is the set of all possible users. A protocol is considered anonymous when the probability that an attacker can correctly identify the user is exactly $\frac{1}{n}$, where n is the number of users in the anonymity set [48].

UNLINKABILITY

Unlinkability ensures that multiple actions performed by a single user cannot linked to each other. This means that if a user manipulates a resource multiple times, it is impossible to determine if it was one user or multiple users that manipulated the resource [48].

UNOBSERVABILITY

The unobservability of a user means that nobody will notice if a message has been exchanged between this user and another party. This can be sender unobservability, nobody will notice if the user sends a message. Receiver unobservability means nobody will notice if the user receives a message. With both combined nobody will notice that a message has been exchanged between two users [48].

PSEUDONYMITY

Pseudonyms are dynamic identifiers which are generally difficult to link to a real identity. A user is considered pseudonymous if he uses a pseudonym instead of a real identifier. Sender and receiver pseudonymity are defined as being pseudonymous while sending or receiving a message respectively [48].

2.2.4.2 *Anonymity protocols*

Not all anonymous communication protocols are practical, some are merely theoretical while others require non-existing network topologies. The practical methods are listed below [49].

TOR

Tor is based on onion routing. Onion routing works as follows, if a node wants to make a connection to another party, a chain of nodes or onion routers is created. Each router along the chain removes a layer of encryption and relays the packet. This ensures no router knows the source, destination, and the contents of the packet [50].

Tor is a second generation onion router. It has several improvements over standard onion routing, e.g. perfect forward secrecy, most TCP applications supported by default, TCP streams can share the same route, and congestion control Dingledine et al. [51].

MIX NETWORKS

Mixnet-based systems operate similarly to onion routing. Packets are sent along a chain of mix servers. These servers mix the received packets, remove a layer of encryption and relay them along the chain. The anonymity provided by this system is not unconditional, atleast one of the mix servers needs to be honest. Furthermore, only sender anonymity can be provided [48].

INVISIBLE INTERNET PROJECT

The Invisible Internet Project (I2P) is relatively similar to Tor. The routing of packets uses the same method. I2P only supports UDP messages however, while Tor supports only TCP. This makes I2P more suitable for message and streaming based applications, while Tor is more suitable for web browsing and file transfers [52].

FREENET

Freenet distributes encrypted parts of files amongst its users. Users only connect through intermediate users to retrieve files. If a users wants to store a file, it is encrypted, its file key is generated, it is split into parts, and finally each part is send to a user. When a user receives a part of a file, it randomly decides to store it or pass it on to an other user. To retrieve a file, the file key is spread through the network. If a user is located who has a part of the file, he sends the part via the same route as he received the request [53].

2.2.4.3 *Overview of related work*

An overview on studies concerning anonymous communication methods is presented in table 4. The the table indicates onion routing is evaluated most often, but the scientific community is not merely focussed on a single solution.

Table 4: Anonymous communication literature overview

	[48]	[54]	[47]	[49]	[52]	[51]	[53]	[55]	[56]	[57]	[50]
Onion routing	X		X	X	X	X					X
Mix networks	X		X	X	X						
I2P	X			X	X						
Crowds	X		X						X	X	
Other	X	X	X				X	X			

2.3 RELATED WORK

This section describes the related work in detail and may overlap with the previous section which contains an overview of the related work. The related work is separated in four distinct categories, botnet C&C methods, botnet detection methods, information hiding, and anonymous communication methods.

2.3.1 *Botnet Command and Control*

Silva et al. [3] describe three different topologies employed by botnets. The strengths and weaknesses of each of these communication methods are evaluated.
The first topology consists of a central point which relays messages to the bots. Bots connect to this central point to receive new commands, the main protocols used are

IRC and HTTP. This topology provides fast reaction times and direct feedback from the bots. A single point of failure is a major weakness of this approach however.
The second method is a P2P structure. A variety of different P2P protocols have been used by botnets. The lack of a single point of failure makes this approach robust against disruptions.
Finally the third method is a hybrid method, the bots are divided into two groups. One group consists of bots which act like both a client and a server, the other group acts strictly as a client [9].
A purely theoretical model, the random model, was also suggested. Bots that use the random model do not initiate connections, they wait until the botmaster connects. The botmaster scans the network for bots and sends commands when a connection is established.

Bailey et al. [10] describe an other communication method, namely an unstructured communication method; no single bot knows about more than one other bot. This communication method has the advantage of being very robust, but this approach has disadvantages as well, scalability for example.

Cooke et al. [6] explain in detail three botnet classifications; centralized, decentralized, and random botnets. Furthermore, standard protocols such as HTML and P2P are evaluated by Zeidanloo and Manaf [11] as carriers for each of these different C&C methods.

An other overview on botnets and their communication methods was presented by Li et al. [7]. They investigated C&C topologies and communication protocols used by botnets. Existing defensive measures are listed as well. Furthermore, a simple case study on an IRC based botnet, SpyBot, provides an insight into the type of commands which are used to control botnets.

Botnet P2P structures are studied in depth by Dittrich and Dietrich [58]. P2P botnets are more resilient to disruptions than traditional C&C methods, but they show that some defensive strategies and countermeasures can be quite effective [8].

Yan et al. [59] present AntBot, a method to protect a P2P botnet against pollution based defensive strategies. Botnet pollution attacks attempt to disrupt a botnet by injecting falsified C&C traffic encrypted with the correct keys. After simulating this method they concluded that AntBot can withstand these types of defensive strategies.

Dietrich and Rossow [12] evaluated DNS as a viable C&C method. Specifically, DNS tunnelling was used to evade detection. A case study on Feederbot was performed which uses DNS for C&C. Detection methods are proposed which differentiates regu-

lar DNS from C&C traffic.

Kartaltepe et al. [14] have studied how existing botnets exploit social networks as C&C mediums. Possible future C&C methods are envisioned and countermeasures are proposed.

Nagaraja and Houmansadr [18] show how a botnet can utilize social networks to communicate covertly. Instructions can be send through commands embedded in images via steganography. This approach is in theory less detectable than traditional C&C methods, because an existing communication method and image steganography are used.

Social networks can also be used in a hybrid P2P structure [15]. The botmaster sends commands through a social network to servent bots, which act as both client and server. These servent bots relay the commands to the client bots. This method is robust as there is no single point of failure, and covert as HTTP traffic via social networks does not raise suspicion.

Nappa et al. [16] propose a botnet model which abuses existing P2P networks, e.g. Skype, as a carrier for C&C traffic. They show that it is a realistic threat which might be abused in the near future.

Singh et al. [17] developed a C&C channel via encoded email messages. They show this type of C&C channel is robust against countermeasures. C&C messages are hidden in emails which will be classified as spam. Scanning every spam message for C&C messages is resource intensive for email providers, disruption is therefore infeasible.

Tankard [30] provides an overview on Advanced Persistent Threats (APT). He describes their characteristics, the offensive measures they employ, and possible ways to protect against them.

APTs are discussed more in depth by Binde et al. [31]. A known APT is discussed, Operation Aurora. The C&C traffic is examined with the ultimate goal of detecting these types of C&C methods in the future.

Command Five Pty Ltd [19] describe a number of discovered APTs in detail. Their communication protocols are dissected, revealing the C&C protocols used.

2.3.2 *Botnet countermeasures*

Feily et al. [23] describe four botnet detection techniques: signature-based, anomaly-based, DNS-based, and mining-based.

Signature-based detection applies signatures, specific patterns of instructions, of known botnets to network traffic. This detection method is therefore not useful for botnets whose signature is unknown. Anomaly-based detection checks for network traffic anomalies, these anomalies are for example, high network latency, high traffic volumes, or traffic on unusual ports. This allows the detection of new and unknown botnets. DNS-based detection applies anomaly detection algorithms to DNS traffic. Mining-based detection uses machine learning, classification, and clustering to detect botnet C&C traffic.
They show that some of these detection methods are able to detect botnets regardless of botnet protocol and structure.

Botsniffer is a system which uses network-based anomaly detection to identify C&C traffic [26]. They argue that bots of the same botnet will show similar network traffic characteristics. Botsniffer applies statistical algorithms to find correlations in network traffic to detect bots.

BotMiner is an other system which uses similarities in network traffic to detect botnets [27]. The detection work as follows; similar communication traffic and similar malicious traffic is clustered. Cross correlation is performed on this clustered data. This identifies the hosts which share both similar communication patters and similar malicious patters, these hosts are most likely to be bots.

Caballero and Poosankam [28] developed Dispatcher, a tool which extracts data from botnet binaries. It obtains the message format and field semantics from analysing its instructions. Dispatcher was used to analyse the MegaD botnet which allowed the researchers to rewrite C&C messages.

ProVeX is a system which is able to detect encrypted C&C communication [29]. It operates by attempting to decrypt C&C traffic with known encryption algorithms and keys. Statistical tests are employed to determine whether the decrypted traffic matches known signatures. This method has a large computational overhead, yet the detection system is able to operate in real time and it scales up to multiple Gbit/s network speeds.

Holz et al. [24] demonstrate a method to analyse and disrupt P2P botnets. They examine the Storm Worm botnet and present methods to disrupt its communication channel.

A novel way of detecting stealthy P2P botnets has been proposed by Zhang et al. [25]. Statistical fingerprints are applied to identify different types of P2P traffic, which facilitates the distinction of botnet traffic from legitimate traffic. This allows the detection of C&C traffic even if legitimate P2P traffic is used in conjunction with botnet traffic.

2.3.3 *Covert & Hidden channnels*

A general study on information hiding was performed by Moulin and O'Sullivan [33]. A notion of hiding capacity was introduced, which indicates the amount of information which can be hidden in a given channel.

Zander et al. [34] provides an overview of covert channels in network traffic. A number of viable covert channel techniques are presented, e.g. unused header bits, checksum fields, timestamp fields, and packet timings. Also a list of covert channel countermeasures is presented which attempt to detect and eliminate the aforementioned covert channels.

Artz [4] provides an overview of steganography tools currently available. Some of the most common techniques are hiding information in images, audio files and the ordering of data. A more recent overview was written by Cheddad et al. [39]

An in-depth overview on network covert channels is presented by Cabuk [43]. The design, analysis, detection, and elimination of these covert channels were discussed. Finally an covert IP channel prototype was implemented which proved hard to differentiate from ordinary traffic.

Embedding covert channels into TCP/IP is discussed by Murdoch and Lewis [5]. They show that hiding data in header fields is not as simple as commonly believed, as they show a method to differentiate modified and unmodified headers. They describe a way to map block cipher output onto TCP ISNs which are not distinguishable from ordinary headers.

Fisk et al. [44] provide an in-depth discussion on the major steganography algorithms used for digital images. They argue there exists a trade-off between robustness and

payload, the more data is embedded in an image, the higher the chance of detection.

Covert channels can be embedded in IPv6 as well [35]. A rough method of finding covert channels in a protocol was described, applying this method to IPv6 resulted in the discovery of 22 covert channels, some of whom were not detected by defensive systems.

DNS can be used as a covert channel as well [36]. The insertion of an encrypted cipher directly into a DNS ID is distinguishable from normal DNS IDs though. It was therefore proposed to apply steganography to insert the cypher into the DNS ID. The results indicated the modified DNS IDs were indistinguishable from ordinary DNS IDs.

Bittorrent can provide a covert channel as well [37]. They argue that such a covert channel can be of great benefit to a botnet as a C&C method.

Berger and Hefeeda [13] investigated the usability of SIP as a carrier for C&C data. They show it offers numerous ways to hide C&C messages within SIP traffic which appears to be legitimate.

A robust covert channel communication system was proposed by Yarochkin and Dai [45]. This system uses multiple covert channels spread across different network protocols. This provides redundancy and increased performance, but this has a cost, namely the detectability is increased.

RTP can be used as a covert channel as well [38]. The data is stored in the least significant bits of the timestamp. This provides a hard to detect covert channel at the cost of low bandwidth.

Dey et al. [42] present a novel way of embedding data in html pages. A redundancy in the html specification is exploited to send data hidden from normal web users. The method is easily detected though, a manual inspection of the webpage's source would immediately reveal the existence of the covert channel.

Sarsoh et al. [40] show a method of hiding information in Microsoft Word Documents. The secret message is made invisible and is added in parts to the ends of paragraphs. PowerPoint files are suitable to hide information as well [41].

2.3.4 *Anonymous communication methods*

Ren and Wu [48] provide a survey on anonymous communications in computer networks. They describe several methods for anonymous communication, some of which are feasible in practice while others are merely theoretical. Furthermore, an overview on anonymity, unlinkability, unobservability and pseudonymity was given. The methods which can be applied in practice are: onion routing, network routing-based techniques, web MIXes, and Hordes. Hordes uses multicast routing to receive data, providing anonymity [54].

Anonymity, unlinkability, unobservability and pseudonymity is discussed in more detail by Danezis and Diaz [47]. They argue that anonymous communication can only be ensured for short periods, attacks will always succeed in the long term by observing the edges of the anonymity system.

Ruiz-Martínez [49] provide an overview of tools and solutions which facilitate anonymous web browsing. Privacy solutions for different network layers are evaluated, the TCP/IP layer, HTTP layer, and application layer. Their goal is to only analyse solutions which can be used in a practical way. The TCP/IP layer solutions they deem practical are Tor, Web MIXes/AN.ON, and the Invisible Internet Project.

An overview of anonymity technology is presented by Li et al. [52]. Measurements show Tor is the most used anonymous communication method. Other methods which are actively used are the Invisible Internet Project (I2P) and proxy servers.

Dingledine et al. [51] introduced Tor, an anonymous communication service. It has a number of improvements over standard onion routing such as congestion control and forward secrecy.

An other method of exchanging data anonymously is Freenet [53]. It is a P2P network application which allows its users to public and retrieve data while providing anonymity for both authors and readers.

Shue and Gupta [55] proposes a method of preserving the anonymity of the sender of a message. The sender changes the source address of packets to the broadcast address of the subnet. Because the broadcast address is used, reply packets will be send to all users in the subnet. Attackers outside the subnet do not know from which of the users inside the subnet the message originated. This scheme provides limited anonymity against attackers within the same subnet as the sender though.

Reiter and Rubin [56] introduced Crowds, a system which groups its users into a geographically diverse collection. Members of these groups issue requests on behalf of other members. This provides anonymity with regards to the origin of the request, as each member is equally likely to be the origin.

Rass et al. [57] improved the Crowds system. Sender and receiver anonymity were added while at the same time providing a bidirectional anonymous channel.

2.3.5 *Non-scientific literature*

Trend Micro discovered a botnet which abuses Evernote, a note storage service. C&C traffic is distributed among bots through Evernote notes. This provides a communication channel which is similar to legitimate traffic and thus hard to detect. Furthermore, stolen information can easily be transmitted back to the botmaster through Evernote [60].

Recently botnets have begun using Tor as a communication channel. Bots connect to the C&C server via HTTP, which greatly reduces their detectability. Because the Tor network hides the location of the C&C server and encrypts the traffic, it is difficult to determine whether a host is compromised or not [20].

Twitter has been used as a C&C channel as well. Bots search Twitter for specific hashtags, which are encrypted and change every day. Tweets are send in irregular intervals from different accounts and are deleted shortly thereafter. To further avoid detection different user agents are used by the bots [21].

ZeroAccess, a P2P botnet, was investigated by Symantec. It proved to be resilient against countermeasures, mainly because of its large peer list. Because of this large peer list, bots are always capable of finding peers, even if a large number of peers is unavailable [22].

DNS can be used to send data covertly by means of a DNS tunnel. Data is encapsulated inside DNS queries and replies. This requires control of a DNS server to receive the queries, extract the data, and insert new data into the reply. Queries can be routed through legitimate DNS servers to make detection even more difficult [46]. Umbrella Security Labs [32] investigated the current and future role of DNS in botnets. They conclude there are currently no effective measures to counteract botnet C&C via DNS.

3 REQUIREMENTS AND USE CASES

This chapter discusses the requirements which the communication channel must adhere to. This chapter starts off with the high-level requirements, these describe the requirements for the system as a whole. This is followed by a list of secondary requirements, these are not strictly necessary to build a functional communication channel, but greatly improve its effectiveness. Finally a number of use-cases are presented.

The high-level requirements followed directly from the goal of this thesis: to design a anonymous and hidden communication channel.

The secondary requirements were devised after inspecting the properties of existing command and control channels and communication channels.

3.1 HIGH-LEVEL REQUIREMENTS

This section describes the high-level requirements. These requirements apply to the communication channel as a whole. These requirements are split into the primary and secondary requirements.

3.1.1 *Primary requirements*

The primary requirements are the most important requirements. Fulfilling these requirements is the main objective of the communication channel.

ANONYMITY

The identity of the user of the communication channel has to be preserved. This requires pseudonymity Anonymity, which is described in section 2.2.4. The identity of the entity which the user is communication with does not have to be protected.

COVERTNESS

The communication channel has to avoid detection. This is only relevant for the communication to and from the client system.

ROBUSTNESS
To ensure the communication channel can be used reliably, it must be robust. If the connection between parties is somehow severed, it should be possible to reconnect.

3.1.2 *Secondary requirements*

These requirements are not necessary to build a functional communication channel. However, they impact the quality of the channel by preventing common attacks and by requiring a minimum quality of service.

BANDWIDTH
The bandwidth requirements are of lesser importance. A high bandwidth is useful to send and receive large amounts of data in a short amount of time, but there are no strict time limits. The bandwidth should therefore be high enough to send large amounts of data in a reasonable time.

LATENCY
The time between sending and receiving of a message must be short. As the main purpose of the communication channel is to facilitate botnet command and control traffic, real-time control is necessary in specific circumstances.

CONFIDENTIALITY
Only the client or server must be able to read messages addressed to him. No adversary should be able to deduce any information about the message, not even the other clients. However, if an adversary gains control over a client, he can read messages addressed to this client, this is unavoidable.

INTEGRITY
The data send to and from the agents must not be able to be modified without notice, whether it be through malicious interference or other factors.

AUTHENTICATION
Only the server must be able to send commands which are deemed authentic by the agents. Furthermore, only the agents must be able to create a legitimate response.

REPLAY RESISTANCE
An attacker should not be able to make an agent repeat executing a command by replaying a previous message to an agent.

MESSAGE UNIQUENESS
An attacker should not be able to deduce that the contents of two messages are identical.

KEY INDEPENDENCE
The system must not fail if one of the agents is compromised and its key is leaked.

DAMAGE CONTROL
If the situation arises where an agent is compromised, the amount of information leaked must be minimized. No information may be leaked which facilitates the identification of the server or other agents.

3.2 USE CASES

The following use cases facilitate the evaluation of communication channels. These use cases pose restrictions which may or may not limit the feasibility of specific C&C methods. The use cases describe environment in which the agent operates. The use cases have been chosen because they resemble the most typical environments: home workstations and office workstations. Because the use cases are described from the perspective of the agent not all requirements are represented in the use cases. Of the primary requirements only covertness and robustness are important. Anonymity is not important as the goal is to hide the identity of the server, not the agents.

This thesis focusses on the network aspects of communication channels, host specific characteristics will therefore be ignored.

3.2.1 *Use Case 1.*

The first use case consists of an agent installed on a basic PC in a home environment. Home users generally have less protective measures in place, but they have full control over their network and may detect behaviour which is not a consequence of their own actions.

The typical home PC is connected to a simple router with NAT, no consumer firewalls or intrusion detection systems are available on today's market and are therefore not present. Furthermore, the bandwidth is low and there are no guarantees the external IP address will not change.

Because the user has full control over the machine, access control is limited. The user therefore knows which applications he has installed himself and can therefore easily

spot discrepancies in network traffic if he is inclined to investigate.

This lead to the following typical system overview:

FIREWALL: Windows Firewall + NAT Router
IDS: None
ACCESS CONTROL: Limited
NETWORK KNOWLEDGE: Full
APPLICATION RESTRICTIONS: None
STATIC IP: No
BANDWIDTH: Low

3.2.2 *Use Case 2.*

The second use case consists of a small business workstation in an office environment. Businesses often have an employee responsible for IT or hire an external company. One would therefore expect that appropriate security measures have been taken. Furthermore, small business tend to have atleast some security hardware like a dedicated firewall [61].

Small businesses often have (limited) access to dedicated IT personnel, whether it be internal or external. The workstations are provided with access controls. The access controls are configured in such a way that the users can manage a large part of the system themselves, but prohibit the users from making drastic changes which may jeopardize the functioning of the system. Furthermore, some companies decide to blacklist applications which reduce the performance of employees or pose security risks [62].

The network architecture is basic, the workstations are connected to business broadband through a router. The broadband has a medium speed, faster than most consumers but not the fastest available. Because a router is used the workstations do not have static IPs.

This lead to the following typical system overview:

FIREWALL: Hardware Firewall
IDS: None
ACCESS CONTROL: Medium
NETWORK KNOWLEDGE: Medium
APPLICATION RESTRICTIONS: Blacklist
STATIC IP: No
BANDWIDTH: Medium

Certain webservices may be blocked if they are deemed to lower employee performance [62], these are listed below.

Blacklist:

- Social Media

3.2.3 *Use Case 3.*

The final use case is a corporate environment. Corporate networks generally utilize a whole range of different security measures. For example: strict access policies, dedicated IT department, hardware firewall, IDS, traffic monitoring.

But a dedicated IT department has disadvantages as well. End users often do not have complete control of their workstations. This inhibits their ability to investigate strange behaviour. On the other hand, the IT staff cannot determine whether network traffic originates from a real user or a malicious application, without questioning the user.

System:

FIREWALL: Hardware Firewall
IDS: Hardware IDS
ACCESS CONTROL: Strict
NETWORK KNOWLEDGE: Full
APPLICATION RESTRICTIONS: Whitelist
STATIC IP: Yes
BANDWIDTH: High

Corporate environments filter traffic which is not required for the execution of business processes [62]. The protocols which are generally allowed are listed below.

Protocol Whitelist:

- HTTP(S) (blacklist)
- TCP/IP
- DNS
- Proprietary Microsoft protocols
- Email protocols

Like the previous use case, certain webservices may be blocked.

Blacklist:

- Social Media
- Chat applications

4 ANALYSIS OF SOLUTIONS

This chapter lists and evaluates communication channels which might facilitate hidden communication. Finally the results of this chapter are summarized.

4.1 POSSIBLE SOLUTIONS

In order to select the most suitable covert channel, anonymity protocol, and botnet topology for the system a number of possible solutions will be evaluated. These possible solutions are described below. The next section will evaluate these solutions.

4.1.1 *Botnet topologies*

Section 2.2.1 provides an overview on botnet topologies.

4.1.2 *Covert channels*

Section 2.2.2 contains a number of possible covert methods. This list is not exhaustive however, it was therefore decided to invent novel covert channels, which are listed below.

VOIP

VOIP services are an other possible solution. They provide ample bandwidth and are encrypted. This can be used as a covert channel by, for example, encoding messages as audio or video streams. An other possibility is simply to create a protocol which mimics the properties of VOIP traffic.

EMAIL

Webbased email solutions can be used to exchange messages as well. It might be difficult to avoid spam detectors however, the use of PGP may avoid this issue. Emails have a limited size however, sending large amounts of data is infeasible.

CLOUD STORAGE SERVICES

Cloud storage services like Dropbox, Google Drive, and Microsoft OneDrive can be used to share files between multiple devices instantaneous. Exchanging messages using these services is as simple as encoding the message as a file and uploading it to the server.

MESSAGING PROTOCOLS

Messages are transmitted through the use of a messaging protocol, for example XMPP. Both need to be connected at the same time, otherwise the message will not reach its destination. Chat messages are usually short, this communication protocol is therefore not suitable for long messages.

USENET

Usenet is a distributed network which facilitates the exchange of messages between users. Users can store messages on a usenet server. This message is then distributed throughout the usenet network. The message is then available on every usenet server. This method allows the exchange of large messages, but as messages have to propagate through the network communication is not instantaneous.

SOCIAL NETWORKS

Private messages are a common feature for social networks. The messages are not hidden from the social network itself however. It is therefore possible that the social network will take preventative measures if it detects encrypted communication.

4.1.3 *Anonymity*

Anonymity protocols have been discussed extensively in section 2.2.4. These will be evaluated in the next section.

4.2 EVALUATION

4.2.1 *Botnet Topologies*

4.2.1.1 *Central*

The central topology consists of a central node which relays messages to the other nodes. This has several advantages, messages are delivered fast as there is only a short route the message has to travel. This topology is reliable as well, as there are few points at which communication may break down. Another advantage is the low amount of information each node (except the central node) has regarding the other nodes. This prevents adversaries from an easy way of controlling a large amount of nodes. Finally, the central topology is easy to implement which should result in less errors.

The central topology has disadvantages as well, the central node is an obvious single point of failure. If the central node is compromised, the entire network is lost. An other disadvantage is scalability, the central node has to handle communication to all nodes which may become too much to handle.

4.2.1.2 *Peer-to-peer*

The P2P topology lacks a central node, each node interacts with several other nodes to send and receive messages. This makes the system robust when compared to the other topologies, as there are nodes essential for the operation of the network. Scalability is an advantage as well, an increase in the number of nodes does not increase the load on each of the nodes.

But this topology has disadvantages as well. Because each node communicates with several other nodes, he needs to know these nodes. This information may give an adversary the means to dismantle the entire network if one of the nodes is compromised. Another disadvantage is the complexity required. Nodes have to form the network structure automatically, ensure they stay connected even when other nodes fail, and prevent information leaks as much as possible. The final disadvantage is the message propagation speed. This speed depends on the number of interconnections between nodes, if nodes are connected to a large amount of nodes the propagation speed is relatively high and vice versa. But the propagation speed is lower than other topologies either way.

4.2.1.3 *Hybrid*

The hybrid topology is a combination of both the central and P2P topologies. A select number of nodes act as server nodes. These nodes share messages between them and propagate them to the client nodes. Because the hybrid topology is a combination between the central and P2P topology, it shares some of their advantages and disadvantages as well. One of the advantages is scalability, if the server nodes are under heavy load, simply add more server nodes. Because the number of server nodes is much smaller then the number of client nodes messages can spread through the network relatively fast. The network is robust as well, there is no central node on which the entire network depends.

The hybrid topology has some disadvantages as well, it is inefficient with a small number of nodes, because it requires a reasonable number of server nodes to provide redundancy. Another disadvantage is the complexity. There are two types of nodes, these are sufficiently different that two implementations are required, thus the complexity increases.

4.2.1.4 *Unstructured*

The final topology is the unstructured topology. This topology is structured as a chain, each node knows one other node, messages it receives will be relayed to this node. This has some advantages, it is difficult to map the network because of the limited connections between the nodes. Because messages travel along a chain it is difficult to

detect where a message originated.

The unstructured topology has several major disadvantages. Because of the chain structure messages are propagated at a slow rate through the network. It also makes the network sensitive to disruptions, if one node fails the network is broken.

4.2.1.5 *Push or Pull*

Botnet C&C methods communicate through pull or push messages. Both of these methods have certain advantages. Push messages are fast, use less bandwidth, and are less vulnerable to disruption, as the receiver does not need the identity of the sender.

Pull has some advantages as well, it does not require an open port which other nodes have to connect to. It does not require a constant open connection either. Finally it is less likely to be disrupted by defensive measures like firewalls or intrusion detection systems.

4.2.1.6 *Use cases*

To determine which of the botnet topologies are suitable for the system they are compared to the use cases from section 3.2. Each of the use cases have limitations, it is therefore important to determine if the topologies are unable to function because of these limitations.

USE CASE 1

Because the workstation is connected to the internet via an NAT and dynamic IP address, the P2P and unstructured topologies are not suitable. The hybrid topology is partially suitable, the workstation can act only as a client node, not as a server. The push mechanic requires an open connection to receive new messages, and may therefore be of limited effectiveness. The central topology and pull mechanic are not limited by this use case.

USE CASE 2

The network is separated from the internet by a firewall, topologies which rely on incoming connections will not function. This means the P2P and unstructured topology are not suitable. The hybrid topology is, just like the previous use case, only suitable if the workstation is a client node. The push mechanism requires an open connection to function as the firewall will disallow connections initiated from the internet. The central topology and pull mechanic are not limited by this use case.

USE CASE 3

The final use case has strict network restrictions, this limits the effectiveness of sev-

Table 5: Botnet topology overview

	Use case 1	Use case 2	Use case 3
Central	Yes	Yes	Yes
Peer-to-peer	No	No	No
Hybrid	Partial	Partial	No
Unstructured	No	No	No
Push	No	No	No
Pull	Yes	Yes	Yes

eral topologies: P2P, hybrid, and unstructured. These topologies rely on connections initiated from outside the network, which fall under heavy scrutiny by firewalls and intrusion detection systems. This is the case for the push mechanic as well. The only methods which can be reliably used are the central topology and the pull mechanic.

4.2.1.7 *Overview*

Table 5 provides an overview of the topologies compared to the use cases. The values indicate whether a topology will function with regards to a specific use case. From these evaluations and use cases it can be concluded that the most suitable topology is the central topology. Its major disadvantage, the single point of failure, is an obvious concern. But because the system implements anonymity as well, the risk that the server is compromised is low. Scalability is not one of the requirements so it is not of great concern.

The pull mechanic appears slightly favourable over the push mechanic, but the the push mechanic with an open connection will function as well. Both are suitable, it depends on the specific protocol which of the two is the better choice.

4.2.2 *C&C protocols*

This section describes the advantages and disadvantages of the C&C protocols listed in section 2.2.1. After which these protocols are compared to the use cases to determine their viability. Finally an overview is given on the suitability of each protocol for the system.

4.2.2.1 *IRC*

IRC has been used for quite some time as a C&C protocol. This is because IRC has several advantages, it is easy to implement, it has two way communication, and it is

fast. But because IRC has been used so extensively it is heavily scrutinized and has a high detection rate. Another disadvantage is its central node which is a single point of failure. If this node is compromised the entire network is lost.

4.2.2.2 *HTTP*

HTTP is a straight forward C&C protocol, the server hosts a website, bots connect to the website to send and receive commands. This method has several advantages, it is simple, fast, provides two way communication, and is difficult to detect if encrypted via HTTPS. However, HTTP has some disadvantages as well, the server is an obvious single point of failure and is easy to block. The major disadvantage is that it is difficult to use in combination with an anonymity service, as there is no intermediate node in the protocol, communication between client and server is direct.

4.2.2.3 *DNS*

DNS has been utilized as a C&C protocol as well. Its speed and prevalent use makes is useful for botnets because DNS is rarely filtered. But the bandwidth DNS can provide is low, and the DNS server is an easy target for adversaries.

4.2.2.4 *SIP*

SIP is a relatively novel C&C protocol. It supports two way communication, it has high bandwidth, and is encrypted. But because SIP has a distinct traffic pattern, it is not enough just to use the same packet structure, the traffic characteristics have to be the same as well. Otherwise it is easy to detect the protocol is not used to transmit voice traffic but other data. Furthermore, because SIP is build on top of UDP it has no message integrity, which is necessary for a reliable C&C protocol. This can be solved at a cost of increased complexity.

4.2.2.5 *Use Cases*

To determine which of the C&C protocols are suitable for the system they are compared to the use cases from section 3.2. Each of the use cases have limitations, it is therefore important to determine which of these protocols are crippled because of these limitations.

USE CASE 1

The first use case poses limited restrictions, it is therefore unlikely one of the C&C will be detected or blocked.

USE CASE 2

The second use case features more advanced security measures than the first use case.

Table 6: C&C protocol overview

	Use case 1	Use case 2	Use case 3
IRC	Yes	No	No
HTTP	Yes	Yes	Yes
DNS	Yes	Yes	No
SIP	Yes	Yes	Yes

It is safe to assume IRC will be detected as malicious, as it is hardly used for legitimate purposes in office environments. But the other C&C protocols will most likely avoid detection, as it is not trivial to distinguish malicious traffic from legitimate traffic.

USE CASE 3

The third and final use case features the most advanced security measures of the three uses cases. Modern security measures can detect malicious IRC [63] and DNS [64] traffic so these protocols are not suitable. HTTP and SIP can be encrypted, so it is not possible to detect malicious traffic from its contents. Analysis of the traffic patterns may indicate malicious behaviour, there are currently no solutions which can perform these kinds of analysis automatically and in real-time.

4.2.2.6 *Overview*

Table 6 shows the results of the evaluation. Of the existing C&C protocols, only HTTP and SIP function in each of the use cases, but both of these have their drawbacks. HTTP is difficult to use with regards to anonymity, while SIP is difficult to implement if the goal is to remain as hidden as possible. From this we can conclude a novel C&C method might be more suited for the system.

4.2.3 *Covert channel protocols*

In section 2.2.3 a number of protocols suitable as covert channels were discussed. In this section the advantages and disadvantages of these protocols are evaluated, then they will be tested against the limitations of the use cases described in section 3.2. Finally an overview is presented of the protocols suitable for the system.

4.2.3.1 *TCP/IP*

TCP/IP is one of the most pervasive network protocols in use today. It has numerous header options which can be used to hide data. This data is difficult to detect, only a small amount of hidden data is send in each packet. Statistical analysis is required to discover the existence of a covert channel. But this means the bandwidth is low as well.

Furthermore, the covert channel requires a server to communicate with, the packets send to this server must be directly accessible otherwise the hidden data cannot be extracted. So a server with a legitimate purpose has to be created, otherwise the traffic to this server will raise suspicion.

4.2.3.2 *Bittorrent*

Bittorrent can be used as a covert channel as well. To send messages coverty over bittorrent, each node must join the same swarm[1]. The nodes act as ordinary peers when communication with other peers, when they wish to communicate with other nodes, they hide the message in the bittorrent traffic. This is difficult to detect, as it requires extensive analysis of the network traffic. The high bandwidth and widespread use of bittorrent makes this infeasible.

Bittorrent has some disadvantages as well. The bandwidth overhead is quite substantial, this may attract suspicion which may facilitate the discovery of the malicious bittorrent software on the system.

4.2.3.3 *RTP*

The time stamp field in RTP headers can be used to hide information. This is difficult to detect, but the low bandwidth makes it impractical for any realistic applications.

4.2.4 *Use Cases*

To determine which of the botnet topologies are suitable for the system they are compared to the use cases from section 3.2.

USE CASE 1
The first use case does not limit the functioning of the aforementioned covert channels in any way. The TCP/IP, bittorrent, and RTP are generally used in home networks, so it is unlikely the network traffic will be classified as suspicious.

USE CASE 2
The stricter network surveillance in the second use case will likely cause bittorrent to be detected. Bittorrent is not commonplace in office environment and its large bandwidth usage and its many connections will draw attention. TCP/IP and RTP are more commonly used and will most likely avoid detection.

USE CASE 3
The third use case features the strictest network security. As TCP/IP requires a large

1 Peers sharing a torrent

Table 7: Covert channel protocol overview

	Use case 1	Use case 2	Use case 3
TCP/IP	Yes	Yes	No
Bittorrent	Yes	No	No
RTP	Yes	Yes	Yes

amount of packets to provide a covert channel with a reasonable bandwidth, it is possible to detect with modern intrusion detection systems. TCP/IP is therefore not a suitable covert channel protocol. Bittorrent will most certainly be blocked, as it negatively affects the other users of the network and ordinarily serves no purpose. This leaves RTP as the only suitable protocol.

4.2.4.1 *Overview*

The overview in table 7 indicate only RTP will function reliably under these circumstances. But RTP provides a low amount of bandwidth for a covert channel, far too low to send many messages in a short amount of time. Neither of these covert channel protocols are therefore suitable for the system.

4.2.5 *Steganography*

Instead of embedding the hidden information in the protocols itself, it is also possible to hide information in files. Section 2.2.3.2 investigates steganography in detail. This section will start by evaluating several file types which may be suitable to hide information. The file types will then be reviewed against limitations posed by use cases. Finally an overview on the different file types is presented.

4.2.5.1 *Images*

Images can be used to hide information. This is difficult to detect, even if the image is available in plaintext. Statistical analysis is required to uncover the hidden data, which is highly impractical. The downside of hiding information in messages is the bandwidth, each image can only store a limited amount of hidden data, about 1% of the image data can be used [65]. This method is thus impractical for large messages.

4.2.5.2 *Documents*

Data can be hidden in documents as well. Documents are used by everyone, and will therefore not raise suspicion. Relatively large amounts of data can be hidden in documents, so using documents as a covert channel can provide a relatively high

amount of bandwidth when compared with the other file types. Though if this method attracts the attention of security vendors it is rather trivial to block.

4.2.5.3 *Websites*

Websites can be used to hide information as well. This can be done in many ways, e.g. hidden fields, redundancies in tags, and comments. Websites have the advantage they can be encrypted without raising suspicion. The constant checking for new messages may cause unwanted consequences however, it may for example appear as if employees are visiting websites instead of working.

4.2.5.4 *Use Cases*

To determine which of the steganography methods are suitable for the system they are compared to the use cases from section 3.2.

USE CASE 1
The first use case has a limited amount of bandwidth. Because image steganograpy has an overhead of about 99%, it is unlikely that the bandwidth will be sufficient to send anything other than short messages. The overhead of the other methods, documents and websites, is far lower. These are therefore suitable with regards to this use case.

USE CASE 2
As the second use case has access to higher quality internet access. Even though the different steganogaphic methods all introduce a data overhead, the increased bandwidth when compared to the previous use case ensures all methods can send messages of adequate length.

USE CASE 3
The bandwidth in this use case is highest among the use cases. Overhead is therefore not an issue. This use case does feature some high grade security systems. Large amounts of traffic to specific domains may make it appear employees are wasting time on the internet. This may lead to the website getting blocked [62].

4.2.5.5 *Overview*

The results shown in table 8 indicate documents are the only viable method in all use cases. But just hiding data in a document is not enough. The document has to be transported between the sender and receiver somehow. Steganography on its own is not a C&C channel, it requires an infrastructure to deliver the documents, which makes this solution complex.

Table 8: Steganography overview

	Use case 1	Use case 2	Use case 3
Images	No	Yes	Yes
Documents	Yes	Yes	Yes
Websites	Yes	Yes	Yes

4.2.6 *Anonymity protocols*

This section begins by a description of the advantages and disadvantages of a number of anonymity protocols. Anonymity protocols differ in the type of connection, popularity, and complexity. These protocols will not be compared with the use cases, as anonymity is a requirement for the server, not for the agents. Finally an overview on anonymity protocols is presented which determines the suitable protocols.

4.2.6.1 *Tor*

Tor is the most popular anonymity protocol at time of writing. It is easy to use, and supports the most used type of connections: TCP. This makes it suitable for internet browsing, sending emails, and file sharing. But it is unsuitable for real time applications such as video and voice communication services. Because of its popularity effort has been made to be able to detect tor traffic [66], this means it is not suitable as a covert channel.

4.2.6.2 *Mix networks*

Mix networks supports all IP traffic, the network consist of servers which mix incoming traffic in such a way that it is impossible to detect which output packet corresponds to an input packet. This depends on the number of other users, and requires atleast one trustworthy mix server. Mix networks are not as popular as other anonymity protocols, and rely on the trustworthiness of a select number of mix servers.

4.2.6.3 *Invisible Internet Project*

I2P has gained in popularity lately. It has an advantage over Tor, it supports most protocols, not just TCP. But whereas Tor is mostly used to interact with services outside the Tor network, I2P's main focus lies on communication within the network. I2P has exitnodes where traffic can leave the network, but these are rare compared to Tor. Because I2P is relatively recent it has not been under as much scrutiny as the other protocols.

4.2.6.4 *Freenet*

Freenet takes an other approach to anonymity. It is an anonymous distributed file storage system. This can be used to store a message on one node, while an other node retrieves the message. However, the exchange of messages can only be done by members of the Freenet network, there is no way to communicate with nodes outside the network as is the case with the other protocols. Furthermore, the anonymity is stronger the more people join the network. As freenet is not as popular as Tor or I2P, its level of anonymity is therefore lower as well.

4.2.6.5 *Overview*

Of the protocols described above, Tor and I2P are the clear winners. They are the most popular, and can work together with other services to act covertly. Tor is the optimal choice if TCP is sufficient, otherwise I2P will be sufficient.

4.2.7 *Novel covert channels*

This section evaluates the novel covert channels proposed in section 4.1.2. The advantages and disadvantages of each covert channel are described first. This is followed by a comparison of these covert channels to the use cases described in section 3.2. Finally an overview is presented.

4.2.7.1 *VOIP*

Nowadays VOIP is uses nearly everywhere, which makes it suitable as a covert channel. It has other advantages as well, it has high bandwidth, is encrypted, and has two way communication. But as it is based on UDP it has no protection against packet loss and data corruption. Furthermore it has a distinct traffic pattern which has to be mimicked, for example, connections usually last minutes, not hours.

4.2.7.2 *Email*

Email can be used to send covert messages as well. Email is accessible almost everywhere, is easy to use, nearly instantaneous, and provides two way communication. But because spam is a large problem, email providers have installed advanced anti-spam measures. These measures may interfere with C&C messages.

4.2.7.3 *Cloud storage services*

Cloud storage services are meant to store and retrieve files and share them with others. These services feature a large bandwidth capacity which makes it feasible to send large messages. Furthermore, these services implement push messages, if a shared

file is updated, it is pushed to the other members of the share nearly instantaneous. But relying on these types of services has disadvantages as well. The service has full control over the messages, it can delete and modify files at will. Furthermore it has incentive to block abuse. It is not known whether these services have capabilities to detect malicious behaviour.

4.2.7.4 *Messaging services*

Messaging services are obviously suitable to send messages. They provide instantaneous two way communication. An other advantage of messaging services is that they support encryption. The bandwidth provided by messaging services is low however, as they are meant to share short plaintext messages. This might make it trivial to detect if the service is used for anything other than text messages, the C&C traffic must therefore mimic genuine traffic.

4.2.7.5 *Usenet*

Usenet is a distributed discussion system. It can therefore be used to share messages. It has several advantages, usenet servers generally have high bandwidth, the connections are encrypted, it is resilient due to the number of different servers, and it is difficult to trace the origin of a message. Usenet has several disadvantages as well, the messages are public so the messages may draw unwanted attention, it has no error correction, this has to be done manually, and it is easy to block as it is trivial to detect if a server is a usenet server or not.

4.2.7.6 *Social Networks*

Social networks can be used to send private messages. This can be abused by sending C&C messages between nodes as private messages on the social network. This is difficult to detect, as communication with these social networks is encrypted. However, there are organisations which block social networks by default as it reduces productivity. Furthermore, social networks have incentive to block accounts used by bots.

4.2.7.7 *Use Cases*

To determine which of these covert channels will function under different circumstances they are compared to the use cases from section 3.2.

USE CASE 1

The first use case places no restrictions on the applications and services which can be used. This leads to the conclusion that all of these novel covert channels are feasible, the network cannot distinguish legitimate use from illegitimate use.

Table 9: Novel covert channel overview

	Use case 1	Use case 2	Use case 3
VOIP	Yes	Yes	Yes
Email	Yes	Yes	Yes
Cloud storage services	Yes	Yes	Yes
Messaging services	Yes	Yes	Yes
Usenet	Yes	Yes	Yes
Social networks	Yes	No	No

USE CASE 2

Some companies have decided to block social networks [62]. Social networks can therefore not be reliably used, and are thus unsuitable. The other covert channels are not limited by these restrictions and will therefore function as expected.

USE CASE 3

The final use case poses the same restrictions as the previous use case. This prevents social networks from being used reliably, while the other covert channels avoid these restrictions and will function as expected.

4.2.7.8 *Overview*

Of the covert channels shown in table 9, only social networks do not pass the use cases. The other covert channels are feasible in theory, but some of the disadvantages make them impractical to use. The complexity of utilizing VOIP as a cover channel is not worth it if there exist alternatives which lack this complexity. The remaining covert channels each have their up and downsides, there is no channel which is intrinsically more suitable as a C&C protocol than the others.

5 COMMUNICATION CHANNEL DESIGN

This chapter describes how the evaluations presented in the previous chapter lead to the communication channel design. It starts off by explaining why the communication between the server and the agents has to be split into multiple paths. Next, the requirements for each of these paths are examined. Then a communication protocol is devised. After which the choices for the anonymity protocol and covert channels are explained. Finally, the implementation of the system is presented.

5.1 COMMUNICATION OVERVIEW

From the evaluation in the previous chapter can be concluded that there is no trivial method to combine anonymity and covertness on the same connection. Covert solutions rely on hiding information in trusted connections, anonymous communication prohibits this as an anonymous connection is not trustworthy. In this a section a method is proposed to separate the two requirements. The communication between server and agent is split into two parts: the server and agent both communicate exclusively with a proxy. This allows for the communication between the server and the proxy to be anonymous, while the communication between the agent and the proxy is covert. This proxy can be any protocol or service which can relay traffic from one user to another.

The addition of the proxy leads to the following network layout: two communication paths, with two sub-paths each. The communication path from the C&C server to the agent has been divided into a sub-path from the C&C server to the proxy and a sub-path from the proxy to the agent. The reverse path is divided identical, the sub-paths are from the agent to the proxy, and from the proxy to the C&C server. Figure 5 provides an overview of the network structure.

5.2 PATH SPECIFIC REQUIREMENTS

Traffic flows between the C&C server and the agent in both directions. The requirements for traffic from and to the C&C server are not identical to the requirements for traffic to and from the agent. For example, on the server's end anonymity is required, while on the agent's end covertness is required.

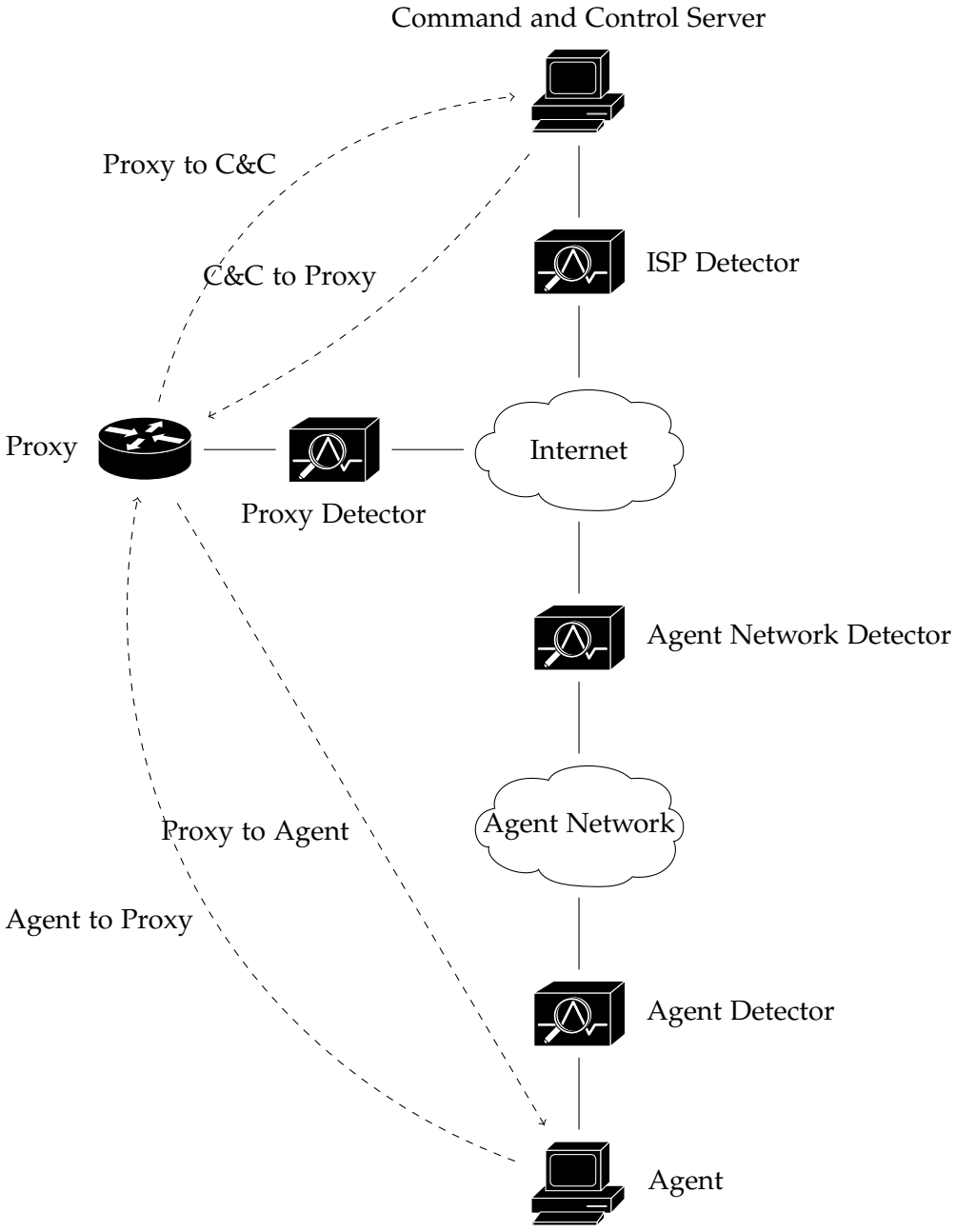

Figure 5: System overview

The requirements for each communication path have been described below. At the end of the section a summary is presented which lists the requirements per path.

5.2.1 *C&C server to agent*

The requirements for the communication from the C&C server to the agent are split in three sets of requirements: requirements concerning all communication from the C&C server to the agent, requirements which only apply to communication from the C&C server to the proxy, and finally the requirements which only apply to the communication from the proxy to the agent. The following paragraph describes the requirements for the entire communication path between C&C server and agent.

The latency from the C&C server to the agent must be low. This allows the agent to perform time-sensitive tasks. As the C&C server sends mostly short messages to the agent, bandwidth requirements are therefore low. The connection between the C&C server has to be robust, once the connection has been broken, for example by an IDS, it is not trivial to reconnect.

There are two requirements which are not equally important for each path: anonymity and covertness. The following paragraphs will describe these requirements for each specific sub-path.

5.2.1.1 *C&C server to proxy*

Anonymity is highly important with regards to the communication between the C&C server and the proxy. This is the last line of defence for the C&C system. It must therefore not be possible to ascertain the whereabouts of the system even if the proxy and the agents are compromised. The second requirement is covertness, which is of medium importance for the sub-path in question. As long as the proxy does not have incentive to actively disrupt the messages from the server it is sufficient.

5.2.1.2 *Proxy to agent*

In contrary to the previous sub-path, anonymity is not important, in fact it is detrimental to covertness. Traffic originating from a legitimate proxy adds credibility to messages. Anonymous messages will therefore arouse suspicion quicker. Covertness on the other hand is of high importance, it stands to reason that the most rigorous checks by automated defence systems are performed on incoming traffic. To avoid detection it is therefore important that the messages appear legitimate.

5.2.2 *Agent to C&C server*

The communication path from agent to C&C server is divided into two sub-paths as well. This leads again to three sets of requirements: requirements for the entire path, and for both sub-paths. The following paragraph lists the requirements for all communication from agent to C&C server.

The latency requirements for communication from the agent to the C&C server are lenient. Responses to commands from the server are not time sensitive and therefore do not require low latency. Bandwidth on the contrary is of high importance. Actions performed by the agent may generate large volumes of data. This data has to be transmitted reliably to the server in a reasonable amount of time, without attracting unwanted attention. The bandwidth requirements are therefore high. Robustness is of less importance, the server will notice if this path is no longer available and can recover the connection. This will take time however, connection loss should therefore be avoided if possible.

The following paragraphs describe anonymity and covertness requirements with regards to the two sub-paths, as the requirements are different for each sub-path.

AGENT TO PROXY

Anonymity is not important for the communication from the agent to the proxy. On the contrary, anonymous communication may give the messages an air of suspicion. Covertness on the other hand is important. Even though outgoing traffic is generally inspected less intensive than incoming traffic, it is still important to reduce the risk of detection as much as possible.

PROXY TO C&C SERVER

Anonymity is an important aspect in the communication from the proxy to the C&C server. But anonymity is not necessarily part of the communication protocol itself. Anonymity can be achieved by, for example, placing a message on a public website. Determining where the message originated is trivial compared to determining who has accessed the message. The covertness of the message is less important. If the message has made its way from the agent to the proxy it stands to reason it can reach the C&C server as well.

5.2.3 *Path requirement summary*

Below are the requirements listed for each path. Each requirement is rated from low to

high based on importance, e.g. "Latency: High" indicates latency is highly important for the path in question and should therefore be as short as possible.

5.2.3.1 *C&C server to agent:*

LATENCY: High
BANDWIDTH: Low
ROBUSTNESS: High

C&C SERVER TO PROXY:

ANONYMITY: High
COVERTNESS: Medium

5.2.3.2 *Proxy to agent:*

ANONYMITY: Low
COVERTNESS: High

5.2.3.3 *Agent to C&C server:*

LATENCY: Low
BANDWIDTH: High
ROBUSTNESS: Medium

AGENT TO PROXY:

ANONYMITY: None
COVERTNESS: High

PROXY TO C&C SERVER:

ANONYMITY: High
COVERTNESS: Low

5.3 TRANSLATED REQUIREMENTS

This section explains how each of the requirements is fulfilled.

CONFIDENTIALITY Encryption: By encrypting all messages with the public key of the recipient only the recipient can read the message.

INTEGRITY Signatures: To ensure the messages are not modified intentional or unintentional they are signed by the sender. The receiver can verify the message to ensure its integrity.

AUTHENTICATION Signatures: The authenticity of messages can be verified by the usage of signatures. Messages are signed by the sender and are verified by the receiver.

REPLAY RESISTANCE Unique message identifier: Each message has a unique identifier, it is therefore impossible to replay a message without detection.

MESSAGE UNIQUENESS Unique message identifier: Each message has a unique identifier and is therefore unique.

KEY INDEPENDENCE Unique keys: Server and agents each have their own public and private key pair.

DAMAGE CONTROL Public keys: Once data is transmitted only the intended receiver may decrypt it. This prevents an adversary from intercepting messages to and from agents outside their control.

5.4 SELECTED SOLUTIONS

5.4.1 *Network topology*

The use of the proxy diminishes the disadvantages of the central topology, scalability and the single point of failure. Scalability is delegated to the proxy, agents connect to the proxy. So an increase in the number of agents increases the load for the proxy, not for the C&C server. The C&C server as a single point of failure is less of a risk as well. Adversaries cannot directly disrupt or take control of the server. Therefore the central topology is the optimal choice, as it fast, easy to implement, and information leaks are minimized.

5.4.2 *Anonymity*

Section 4.2.6 evaluated several anonymity protocols. The conclusion from this evaluation was that Tor has several advantages over the other protocols. Tor is also suitable to function within the proposed network structure consisting of several paths and a proxy. The large number of exit nodes allow Tor clients to connect to services outside the network fast and easy, while remaining anonymous. Tor has therefore been selected as the anonymity protocol which will be used in the system.

5.4.3 *Covert channels*

Several covert channels have been evaluated in section 4.2.3. A number of these use servers which relay communication between multiple clients. This fits the proposed network structure, these servers can act as the proxy. But none of the covert channels are suitable to transmit every type of message. For example, some messages need to be transmitted instantaneous, while other messages need to be available for a long time. Furthermore, a single covert channel cannot provide the desired redundancy. It was therefore decided to use multiple covert channels at the same time. This improves the redundancy of the system, while at the same time the probability of detection will decrease. This approach increases the number of suitable covert channels dramatically. Each of the covert channels listed below can be useful for a specific function, because each has its own specific advantages.

EMAIL: Fast, messages are stored.

CLOUD STORAGE SERVICES: High bandwidth, supports large messages, messages are stored.

MESSAGING SERVICES: Fast, suitable for short messages.

USENET: Redundant, messages are stored, high bandwidth.

Because of time constraints it was infeasible to implement each of these covert channels. It was therefore decided to implement cloud storage services and messaging services. These two channels complement each other as one is suitable for short messages which need to be delivered at once, while the other is suitable for larger messages which are not immediately required.

5.5 IMPLEMENTATION

This section describes several choices which were made during the implementation of the system.

5.5.1 *Description*

The implemented system consists of a server build in Django, and an agent implemented in Python. Because the server and agent are both written in Python the codebases shared. The server has a web interface which allows a user to view the status of the agents connected to the network, to send commands, and view responses. The agents are daemons which run in the background and are only active when executing a command.

5.5.2 *Solved problems*

During the design of the system several problems were encountered and solved. The problems are listed below together with their solutions.

A COMPROMISED CLIENT COULD ALLOW AN ADVERSARY TO CREATE FAKE CLIENTS.
Clients must be registered with a unique private key and identifier which are generated during the installation process.

A CLIENT MAY NOT RECEIVE A MESSAGE.
Clients send a response after receiving message. The server logs the messages for which no response was received and allows the user to resend the message.

A MESSAGE MAY DEPEND ON A PREVIOUS MESSAGE.
Multiple messages can be combined into a single message.

THE SERVER MAY NOT RECEIVE A MESSAGE.
The server does not know at which path the message failed, so the original message can be retransmitted to the client.

A COMMUNICATION METHOD MAY BE BLOCKED.
Plugins allow the addition of new communication methods.

HOW TO SUPPORT PLUGINS
The use of python allows the loading of new modules at runtime. New plugins can be added to the clients by storing them in a specific directory.

CLIENTS MUST STORE INFORMATION, MESSAGE SEQUENCE NUMBERS FOR EXAMPLE
Config files will be used to store client specific information.

5.5.3 *Message types and properties*

The system supports several types of messages. These are listed below together with a description.

COMMAND: execute a command on a client.

FILE: send a file to a client.

FILE REQUEST: retrieve a file from a client.

ENABLE PLUGIN: enable a specific plugin on a client.

DISABLE PLUGIN: disable a specific plugin on a client.

INFO: retrieve information from a client such as operating system information and plugin status.

Messages have several properties, some of which are determined by the contents, others by the user.

LENGTH: the length of the message.

RECIPIENTS: the number of recipients.

VOLATILITY: determines if a message must be send to clients which are offline at the moment.

URGENCY: determines the urgency of the message.

5.5.4 *Plugin properties*

Each plugin has several properties which affect its usage for each type of message. The plugin is assigned an arbitrary value for each property to determine its relative performance for that property. The selection algorithm described below determines the optimal plugin for a specific message based on these properties.

BANDWIDTH: scale from one to ten, ten is the highest bandwidth, one the lowest.

BROADCAST: one if the plugin supports broadcast messages, zero if not.

VOLATILITY: one if the plugin supports the buffering of messages. Zero if a message is lost if the plugin is not connected.

LATENCY: scale from one to ten, ten is the lowest latency, one is the highest latency.

5.5.5 *Plugin selection algorithm*

As agents are not directly controlled, they have to automatically select the optimal plugin to send a specific message. The selection algorithm compares the properties of the message to the properties of the plugins.

STEP 1: if the message is a broadcast message, the plugin must support broadcasts.

STEP 2: if the message is not volatile, the plugin must support non volatile transmissions.

STEP 3: determine bandwidth requirements based on the length of the message. Longer messages require a higher bandwidth score and vice versa.

STEP 4: select the plugin with the highest latency score from the plugins which passed the previous steps.

5.5.6 *Server workflow*

The server consists of two separate threads, the UI thread and the background thread. The UI thread handles all interaction with the user. If the user wishes to send a message, the UI thread places that message in the outgoing message queue. The background thread constantly checks this queue for new messages. Incoming messages are inserted directly into the database by the background thread.

The background thread starts by searching the plugins folder for plugins. Each plugin is started in a separate thread. This ensures the plugins can listen for messages at all times and are not blocked by other plugins. Incoming messages are placed in the incoming message queue of the background process. The outgoing message queue is constantly checked for new messages. If a new message is present the optimal plugin is selected and the message is transmitted.

5.5.7 *Agent workflow*

An agent consists of two threads as well, the main thread and the background thread. The background thread functions the same as the background thread of the server. The main thread reads messages from the incoming message queue, executes the appropriate actions, and place a response in the outgoing message queue.

6 EVALUATION

In this chapter the selected communication methods are evaluated with regards to their ability to remain undetected. This evaluation is performed to show the communication method fulfils the covertness requirement described in section 3. The cloud storage services are evaluated based on its properties and usage. On the other hand, XMPP is evaluated empirically.

6.1 CLOUD STORAGE SERVICES

Cloud storage services are used by legitimate users to store and retrieve files. The network characteristics legitimate usage are: long-lived connections with bursts of encrypted data of random length with random intervals. The system uses these services to send and receive messages of random length and with random intervals. An adversary will therefore not be able to determine if a connection belongs to a legitimate user or the system, as the traffic characteristics are identical and the data is encrypted.

6.2 XMPP

XMPP is generally used to send short messages, but it is capable of transferring files between users as well. The system uses XMPP to send encrypted messages, because of the encryption the length of the messages is increases. This may result in obvious traffic patterns. This section investigates whether this is the case by comparing network traffic from a legitimate chat session to traffic generated by the system.

6.2.1 *Test Process*

The traffic was gathered with tcpdump[1] over the course of a day. Both the chat application (Pidgin) and the implemented system connected from the same machine to the same server (`jabberd.eu`). During this time about 200 chat messages and about 20 system messages were captured.

Initially several factors caused discrepancies in the data. These discrepancies were too severe to draw conclusions from the data. The test was therefore repeated while avoiding these issues. The causes for these discrepancies are listed below.

1 http://www.tcpdump.org/

DIFFERENT ENCRYPTION PROTOCOLS

When an application connects to a server via SSL an encryption protocol is negotiated. As the system and the chat application use different libraries to connect to the server they may negotiate different protocols. This may cause a difference in the amount of data which can be transmitted per packet, and the amount of handshake messages are required. The problem was avoided by finding a server which negotiates the same encryption protocol with both clients. `jabberd.eu` has this property.

RECONNECTS

The initial handshake requires a large amount of messages. A reconnect will therefore cause a spike in traffic unrelated to the transmitted data. This will therefore make the data more difficult to interpret. This problem was managed by running the test multiple times until no reconnects occurred.

KEEP ALIVE INTERVAL

XMPP requires an open connection between client and server. Keep alive messages are transmitted at regular intervals to ensure this connection is not closed by intermediate nodes such as firewalls and routers. This interval can vary depending on the combination of client and server. A lower keep alive interval causes a client to transmit more data and packets, thus skewing the results. This problem was avoided by searching for a server which used the same keep alive interval for both the chat application and the system. Again, `jabberd.eu` has this property.

6.2.2 *Results*

Three properties of the generated traffic have been analysed, bandwidth, packets per minute, and average packet length. These three properties allow for a simple comparison between traffic generated by the system and real chat traffic.

6.2.2.1 *Bandwidth*

Figure 6 shows the amount of bandwidth generated for the duration of the test. A logarithmic scale was chosen to improve the readability of the graph. The graph starts at 500 bytes, as the data below this value is not relevant. Both the system and the chat application transmit roughly 500 bytes per minute to maintain their connection.

The graph shows no major differences between both sources of traffic. Neither produces significantly taller or larger number of spikes. Though the spikes generated by the system are sometimes closely grouped together. We conclude from this graph that it is difficult to differentiate the implemented system from real chat traffic based on bandwidth.

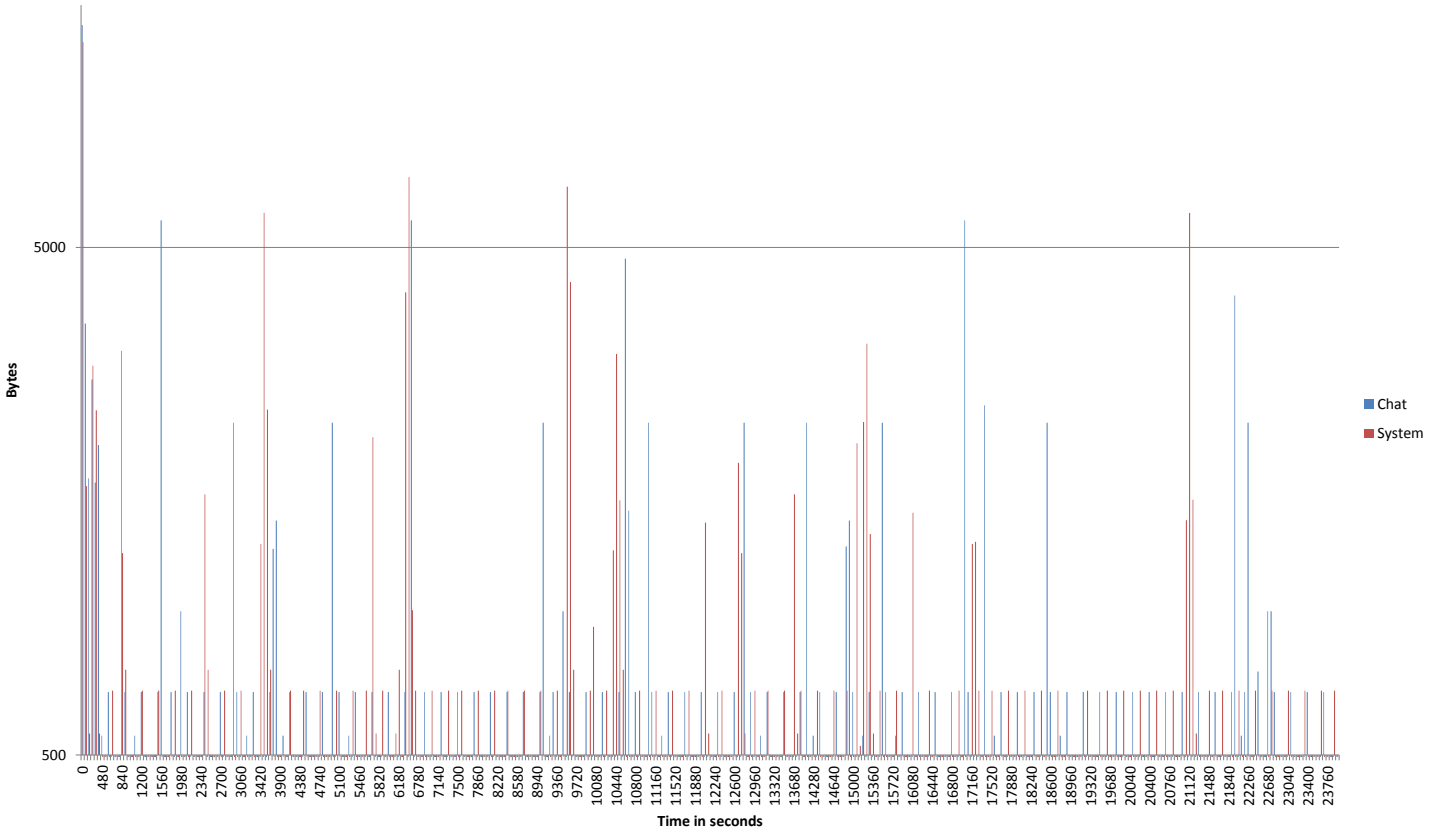

Figure 6: Bandwidth

6.2.2.2 *Packets per minute*

Figure 7 displays the number of packets recorded for each minute. Just as the previous graph, this graph is logarithmic to improve its readability. The graph starts at three packets per minute, as both the system and the chat application both transmit atleast three packets per minute to maintain their connection.

This graph shows that the system sends considerably more packets than the chat application. If this graph is compared with figure 6, we can conclude that the packets send by the system are significantly smaller than the packets send by the chat application, as the bandwidth is comparable. We have been unable to determine what is responsible for this discrepancy. The system has no influence over the size of the packets, the most likely cause is therefore the different software which was used to connect to the server. Different XMPP implementations But this issue does prevent us from drawing conclusions based on this test, further testing is therefore required.

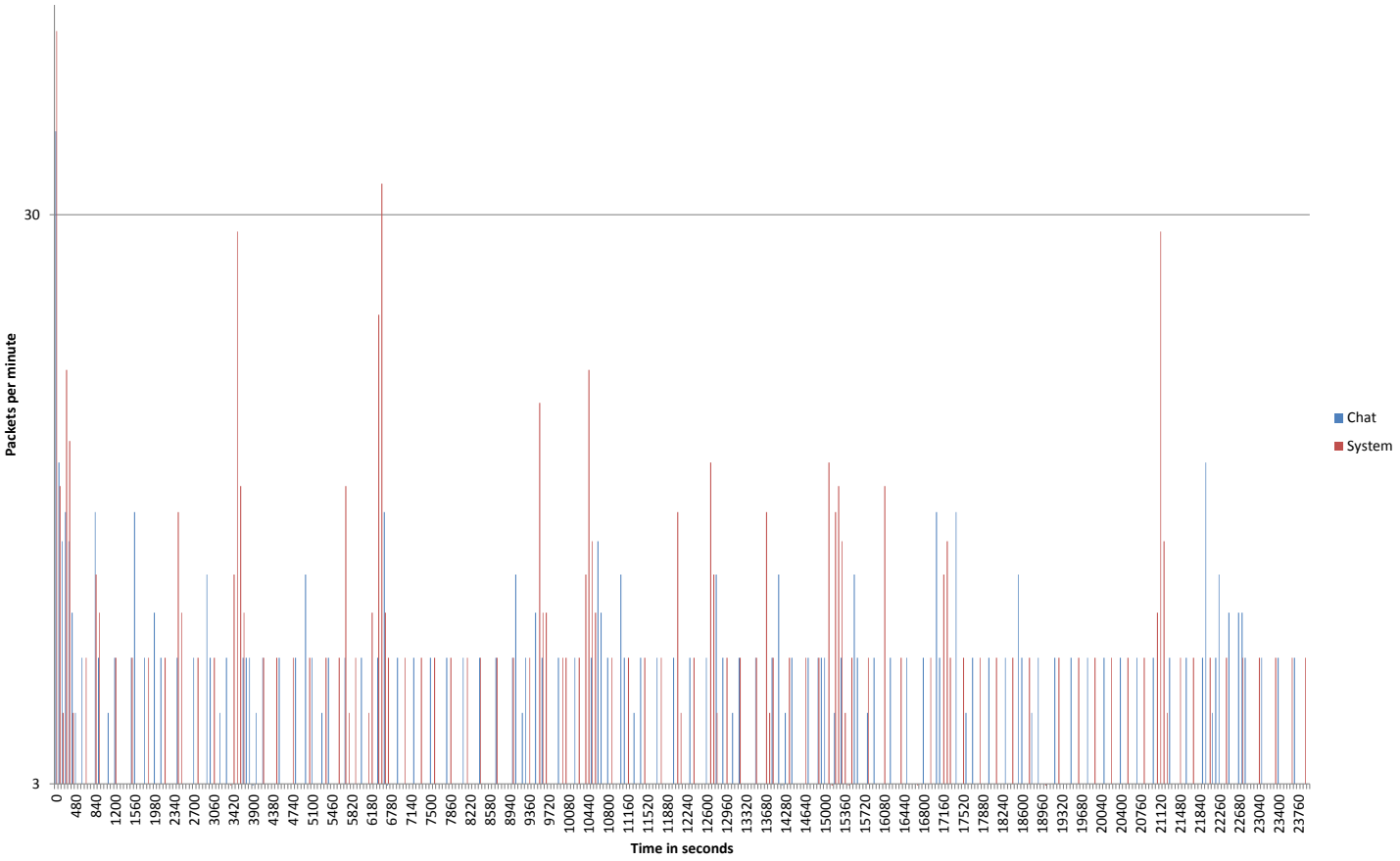

Figure 7: Packets per minute

6.2.2.3 *Average packet length*

Figure 8 shows the average packet length of all packets transmitted within a specific time frame. The minimum packet length is 160 bytes, the graph therefore starts at this point.

This graph shows that there are no significant differences in packet length between both the system and ordinary chat traffic. Though the data indicates the chat traffic produces larger packets more often. This is in line with the previous graph which showed that the system sends more packets. As was mentioned before, this is probably an implementation difference between both XMPP libraries.

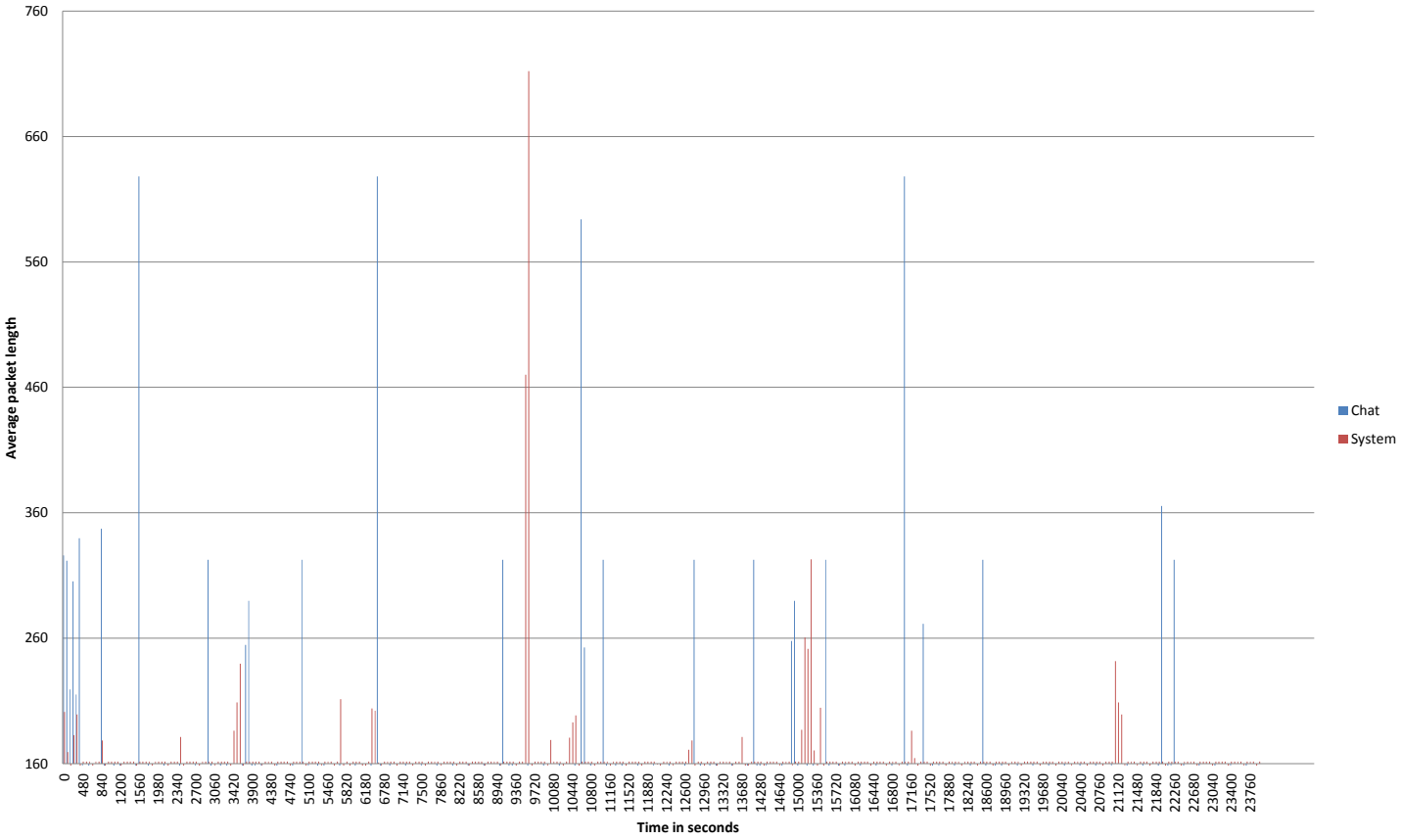

Figure 8: Average packet length

7 DISCUSSION

This chapter discusses the results from the previous chapter and elaborates on the findings presented in this thesis. First the research questions and findings are discussed. This is followed by some limitations. Finally the ethics of this subject are discussed.

7.1 RESEARCH QUESTIONS AND FINDINGS

This section starts by presenting the findings of the sub questions, followed by the main research question.

SUB QUESTION 1: WHICH METHODS PROVIDE ANONYMITY ON THE INTERNET? This question was answered by the evaluation of a number of anonymity protocols in section 4.2.6. From this evaluation was concluded that Tor provides a practical method of obtaining anonymity on the Internet which is suitable for our purposes.

SUB QUESTION 2: WHICH HIDDEN COMMUNICATION CHANNELS EXIST? Section 4.1.2 shows that there are a number of existing and novel methods which facilitate the hiding of data in communication channels. These methods are: RTP, VOIP, email, cloud storage services, messaging services, and usenet.

SUB QUESTION 3: WHICH ANONYMITY SOLUTIONS AND DATA HIDING TECHNIQUES CAN BE COMBINED? In section 5.1 we conclude there is no trivial method to combine anonymity and data hiding techniques. A different network structure was therefore proposed which introduces a proxy. Because of this proxy the need for a single connection to both provide anonymity and covertness disappears. This approach showed to be compatible with Tor, and a number of data hiding techniques: email, cloud storage services, messaging services, and usenet.

MAIN RESEARCH QUESTION: CAN A COMMUNICATION CHANNEL PRESERVE THE ANONYMITY OF ITS USER AND REMAIN HIDDEN AT THE SAME TIME? The answers of the sub questions showed it is theoretically possible to combine anonymity and covertness in communication channels. In chapter 5 a system was presented which implemented this communication channel. The data hiding techniques implemented in this system were evaluated in chapter 6. From this evaluation we can conclude that these techniques are not trivial to detect. We therefore concluded it is possible for a

communication channel to preserver the anonymity of its user and remain hidden at the same time.

7.2 LIMITATIONS

This section lists some limitations of the research presented in this thesis. First a limitation with regards to the messaging protocol test are discussed, followed by the environment of the system. Finally the contradiction of anonymity and authenticity is discussed.

7.2.1 *Messaging protocol test*

The test which was performed was limited in scope. The traffic generated by system was compared with a relatively small amount of real chat traffic, which may or may not be representative of real traffic in general. We could therefore not conclude that it is impossible to detect hidden data in messaging protocols. The only conclusion we can draw is that it is not trivial to detect, because the traffic differences fall within a reasonable margin of error. Further research is therefore necessary.

7.2.2 *Environment*

The system assumes that the communication channels used by the system are present in the environment of the clients. If this is not the case the effectiveness of the communication channel is greatly reduced and may be trivial to detect. Future research should therefore be performed to determine if it is possible for a communication channel to react to its environment. That is, the system mimics the traffic generated by a client. For example, if the client uses XMPP, the system should use XMPP to communicate as well.

7.2.3 *Anonymity versus authenticity*

On of the main goals of the system is to remain anonymous. However signatures are used in order to guarantee that only the system can send legitimate commands to the clients. It is therefore impossible to repudiate that a message originated from a specific system if the private key used to sign messages is discovered on that system. This is of limited concern, an adversary has to gain access to the system to be able to prove that the messages originated from that system. But this issue does show the proposed method does not guarantee the users anonymity under all circumstances.

7.3 ETHICS

The research performed in this thesis raises ethical problems that have to be addressed. In theory the methods described could be used for nefarious purposes. Is it therefore unethical to publish these methods? We believe this is not the case.

We arrived at this conclusion by weighing the benefits of publishing this thesis against the possible harm this may cause. This is discussed below.

7.3.1 *Benefits*

Research into methods to circumvent detection systems allows for the improvement of these detection systems. The security community must research these new methods to be able to prevent them. If those with ulterior motives are the only researchers developing these methods it is only possible to react to these methods. It is therefore important to publish about these methods otherwise the scientific community will fall behind in this field of study.

7.3.2 *Possible harm*

The methods described in this thesis can potentially be used by people or organisations for nefarious purposes. But because the method is public it is possible to devise countermeasures even if the methods are not used as of yet. It is far more dangerous if these methods are not published, but instead developed in secret. The detection of a threat is less difficult if the method is known.

7.3.3 *Distribution of the prototype*

We do intent to limit the distribution of the prototype. Only on request will one be able to gain access to the source code. Requests will be verified to ensure it is intended for research purposes.

8 CONCLUSION

In this thesis anonymous command and control channels were investigated. It is to be expected that these types of hidden communication channels will be used by organisations in the future for nefarious purposes. It is therefore important we gain an understanding of these types of channels. The next section summarizes the research performed in this thesis. This is followed by our findings. Next the future work is described. Finally some predictions are presented.

8.1 SUMMARY

We started this thesis with the following research question:

CAN A COMMUNICATION CHANNEL PRESERVE THE ANONYMITY OF ITS USER AND REMAIN HIDDEN AT THE SAME TIME?

This question was answered by first investigating the relevant literature. The related work spans several topics, botnet C&C channels, botnet detection methods, covert channels, and anonymity protocols.

After gathering the necessary information the requirements for the communication channel were decided. Next a set of use cases were presented, these use cases describe several environments in which the system should function.

An overview of C&C channels, covert channels, and anonymity protocols was presented next. Each of these was then evaluated by means of the aforementioned use cases. This lead to the conclusion that no single solution was suitable to both preserve the identity of its user and remain hidden.

This lead to the introduction of the proxy, this proxy intermediates the communication between two ends of the communication channel. This effectively splits the channel into two paths. Because of this, the anonymity can be preserved on one path, while the communication on the other path remains hidden. This required a re-evaluation of the communication channels examined before. From this evaluation it was concluded that there were several communication channels which are feasible in the envisioned

network structure.

For the implementation cloud storage services and messaging protocols were selected as hidden communication channels, while Tor was selected to provide anonymity. These were then implemented into a prototype.
Finally the selected channels were tested to validate they were actually hidden.

8.2 FINDINGS

The evaluation of existing command and control methods, anonymity protocols, and covert channels resulted in the following: it is not trivial to combine existing methods to create a functional solution. Introducing a proxy solved this issue as it facilitated the separation of anonymous and hidden traffic.

The introduction of the proxy changes the network structure. This requires solutions which are compatible with this new network structure. After evaluating existing and new communication channels we concluded there were several channels which can facilitate hidden communications. From the evaluations of the anonymity protocols we concluded that Tor is the optimal choice.

Two types of communication channels were implemented: two cloud storage services and a messaging protocol. The implemented cloud storage services behave as legitimate users, we consider them therefore hidden by design. This is not the case for the messaging service however, it was therefore tested. The results show that the network traffic generated by the system is not easily distinguishable from ordinary chat traffic.

8.3 FUTURE WORK

The current implementation is limited in scope. The communication channels are selected because they are unlikely to raise suspicion. But this approach is not perfect, it is trivial to detect if the communication channels are not actively used by legitimate users in their environment. If no legitimate users use XMPP regularly for example, the sudden rise in its usage may raise suspicion. It is therefore important that further research should be performed on hiding traffic in the activities of legitimate users.

One improvement is to hide among the genuine traffic generated by the machine. For example, the machine has an active connection to a cloud backup service. A second connection to the same service is unlikely to be noticed by anyone.

An other improvement is abusing the software present on the system. For example, if a messaging application is present on the system it can be used to send hidden messages. The user will not notice, and outside observers cannot tell the messages did not originate from the user.

The communication channel could react to the actions performed by the user as well. So in stead of communicating in specific intervals the communication is related to the randomness of the user. This eliminates obvious traffic pattens.

9

BIBLIOGRAPHY

[1] FireEye, "SUPPLY CHAIN ANALYSIS: From Quartermaster to Sunshop-FireEye," 2014. [Online]. Available: https://www.fireeye.com/resources/pdfs/fireeye-malware-supply-chain.pdf

[2] Mandiant, "APT1: Exposing One of China's Cyber Espionage Units," 2013. [Online]. Available: http://intelreport.mandiant.com/Mandiant_APT1_Report.pdf

[3] S. S. Silva, R. M. Silva, R. C. Pinto, and R. M. Salles, "Botnets: A survey," Computer Networks, vol. 57, no. 2, pp. 378–403, Feb. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1389128612003568

[4] D. Artz, "Digital steganography: hiding data within data," internet computing, IEEE, no. June, pp. 75–80, 2001. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=935180

[5] S. Murdoch and S. Lewis, "Embedding covert channels into TCP/IP," Information Hiding, 2005. [Online]. Available: http://link.springer.com/chapter/10.1007/11558859_19

[6] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understanding, detecting, and disrupting botnets," Proceedings of the Steps to Reducing Unwanted Traffic on the Internet Workshop, 2005. [Online]. Available: http://www.usenix.org/event/srutio5/tech/full_papers/cooke/cooke_html/ https://www.usenix.org/event/srutio5/tech/full_papers/cooke/cooke_html/

[7] C. Li, W. Jiang, and X. Zou, "Botnet: Survey and Case Study," 2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC), pp. 1184–1187, Dec. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epico3/wrapper.htm?arnumber=5412718

[8] P. Wang, L. Wu, B. Aslam, and C. C. Zou, "A Systematic Study on Peer-to-Peer Botnets," 2009 Proceedings of 18th International Conference on Computer Communications and Networks, pp. 1–8, Aug. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epico3/wrapper.htm?arnumber=5235360

[9] P. Wang, S. Sparks, and C. C. Zou, "An Advanced Hybrid Peer-to-Peer Botnet," IEEE Transactions on Dependable and Secure Computing, vol. 7, no. 2,

pp. 113–127, Apr. 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4569852

[10] M. Bailey, E. Cooke, and F. Jahanian, "A survey of botnet technology and defenses," Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications & Technology, 2009. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4804459

[11] H. R. Zeidanloo and A. A. Manaf, "Botnet Command and Control Mechanisms," 2009 Second International Conference on Computer and Electrical Engineering, pp. 564–568, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5380180

[12] C. Dietrich and C. Rossow, "On Botnets that use DNS for Command and Control," Seventh European Conference on Computer Network Defense (EC2ND), 2011, 2011. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6377756

[13] A. Berger and M. Hefeeda, "Exploiting SIP for botnet communication," 2009 5th IEEE Workshop on Secure Network Protocols, pp. 31–36, Oct. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5342244

[14] E. Kartaltepe, J. Morales, S. Xu, and R. Sandhu, "Social network-based botnet command-and-control: emerging threats and countermeasures," ACNS'10 Proceedings of the 8th international conference on Applied cryptography and network security, pp. 511–528, 2010. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-13708-2_30

[15] L. Cao and X. Qiu, "ASP2P: An advanced botnet based on social networks over hybrid P2P," Wireless and Optical Communication Conference (WOCC), 2013 22nd, pp. 677–682, May 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6676460http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6676460

[16] A. Nappa, A. Fattori, M. Balduzzi, M. Dell'Amico, and L. Cavallaro, "Take a deep breath: a stealthy, resilient and cost-effective botnet using skype," Proceedings of the 7th international conference on Detection of intrusions and malware, and vulnerability assessment, 2010. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-14215-4_5

[17] K. Singh, A. Srivastava, J. Giffin, and W. Lee, "Evaluating email's feasibility for botnet command and control," 2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN), pp. 376–385,

2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4630106

[18] S. Nagaraja and A. Houmansadr, "Stegobot: a covert social network botnet," Information Hiding, 2011. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-24178-9_21

[19] Command Five Pty Ltd, "Command and Control in the Fifth Domain," Tech. Rep. February, 2012.

[20] Y. Klijnsma, "Large botnet cause of recent Tor network overload," 2013. [Online]. Available: http://blog.fox-it.com/2013/09/05/large-botnet-cause-of-recent-tor-network-overload/

[21] P. James, "Flashback Mac Malware Uses Twitter as Command and Control Center," 2012. [Online]. Available: http://www.intego.com/mac-security-blog/flashback-mac-malware-uses-twitter-as-command-and-control-center/

[22] A. Neville and R. Gibb, "ZeroAccess Indepth," Symantec Security Response, 2013.

[23] M. Feily, A. Shahrestani, and S. Ramadass, "A Survey of Botnet and Botnet Detection," 2009 Third International Conference on Emerging Security Information, Systems and Technologies, pp. 268–273, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5210988

[24] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling, "Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm." LEET'08 Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats, 2008. [Online]. Available: https://www.usenix.org/event/leet08/tech/full_papers/holz/holz_html/

[25] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, and X. Luo, "Detecting stealthy P2P botnets using statistical traffic fingerprints," 2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN), pp. 121–132, Jun. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5958212

[26] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting botnet command and control channels in network traffic," Proceedings of the 15th Annual Network and Distributed System Security Symposium., 2008. [Online]. Available: http://corescholar.libraries.wright.edu/cse/7/

[27] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "BotMiner: Clustering Analysis of Network Traffic for Protocol-and Structure-Independent Botnet Detection." USENIX Security Symposium, pp.

139–154, 2008. [Online]. Available: https://www.usenix.org/legacyurl/botminer-clustering-analysis-network-traffic-protocol-and-structure-independent-botnet-det

[28] J. Caballero and P. Poosankam, "Dispatcher: Enabling active botnet infiltration using automatic protocol reverse-engineering," Proceedings of the 16th ACM conference on Computer and communications security, 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1653737

[29] C. Rossow and C. Dietrich, "Provex: Detecting botnets with encrypted command and control channels," Detection of Intrusions and Malware, and Vulnerability Assessment Lecture Notes in Computer Science Volume 7967, 2013. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-39235-1_2

[30] C. Tankard, "Advanced Persistent threats and how to monitor and deter them," Network Security, vol. 2011, no. 8, pp. 16–19, Aug. 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1353485811700861

[31] B. Binde, R. McRee, and T. O'Connor, "Assessing outbound traffic to uncover advanced persistent threat," SANS Institute. Whitepaper, 2011. [Online]. Available: https://www.sans.edu/student-files/projects/JWP-Binde-McRee-OConnor.pdf

[32] Umbrella Security Labs, "The Role of DNS in Botnet Command & Control," 2012.

[33] P. Moulin and J. O'Sullivan, "Information-theoretic analysis of information hiding," IEEE Transactions on Information Theory, vol. 49, no. 3, pp. 563–593, Mar. 2003. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1184136

[34] S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols." Source of the DocumentIEEE Communications Surveys and Tutorials, pp. 44–57, 2007. [Online]. Available: http://researchbank.swinburne.edu.au/vital/access/services/Download/swin:9173/SOURCE2

[35] N. Lucena, G. Lewandowski, and S. Chapin, "Covert channels in IPv6," Privacy Enhancing Technologies, pp. 147–166, 2006. [Online]. Available: http://link.springer.com/chapter/10.1007/11767831_10

[36] A. Altalhi, M. Ngadi, S. Omar, and Z. Sidek, "DNS ID Covert Channel based on Lower Bound Steganography for Normal DNS ID Distribution," IJCSI International Journal of Computer Science Issues, vol. 8, no. 6, pp. 149–156, 2011. [Online]. Available: http://www.ijcsi.org/papers/IJCSI-8-6-3-149-156.pdf

[37] J. Desimone, D. Johnson, B. Yuan, and P. Lutz, "Covert Channel in the BitTorrent Tracker Protocol," Dept. of Networking, Security, and Systems

Administration (GCCIS)–Conference Proceedings, 2012. [Online]. Available: https://ritdml.rit.edu/handle/1850/15928

[38] C. Forbes, "A New covert channel over RTP," Dept. of Networking, Security, and Systems Administration (GCCIS), 2009. [Online]. Available: http://scholarworks.rit.edu/theses/806/

[39] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," Signal Processing, vol. 90, no. 3, pp. 727–752, Mar. 2010. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0165168409003648

[40] J. T. Sarsoh, K. M. Hashem, and H. I. Hendi, "An Effective Method for Hidding Data in Microsoft Word Documents," Global Journal of Computer Science and Technology, Vol 12, No 12-E (2012), vol. 12, no. 12, pp. 6–10, 2012.

[41] J. Choi, "Methods to Hide Malicious Codes in PowerPoint," International Journal of e-Education, e-Business, e-Management and e-Learning, vol. 3, no. 6, pp. 488–491, 2013. [Online]. Available: http://www.ijeeee.org/index.php?m=content&c=index&a=show&catid=41&id=602

[42] S. Dey, H. Al-Qaheri, and S. Sanyal, "Embedding Secret Data in Html Web Page," arXiv preprint arXiv:1004.0459, pp. 1–10, 2010. [Online]. Available: http://arxiv.org/abs/1004.0459

[43] S. Cabuk, "Network covert channels: Design, analysis, detection, and elimination," CERIAS Tech Report 2006-53, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1329837

[44] G. Fisk, M. Fisk, C. Papadopoulos, and J. Neil, "Eliminating steganography in Internet traffic with active wardens," Information Hiding, pp. 1–17, 2003. [Online]. Available: http://link.springer.com/chapter/10.1007/3-540-36415-3_2

[45] F. Yarochkin and S. Dai, "Introducing P2P architecture in adaptive covert communication system," First Asian Himalayas International Conference on Internet, 2009. AH-ICI 2009., pp. 1–7, Nov. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5340293http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5340293

[46] R. Cave, "Identifying Covert Channels in DNS: Separating the Forest from the Trees," 2013. [Online]. Available: http://www.solutionary.com/resource-center/blog/2013/09/identifying-covert-channels-in-dns/

[47] G. Danezis and C. Diaz, "A survey of anonymous communication channels," Computer Communications, no. January, pp. 1–46, 2008. [Online]. Available: http://www.cosic.esat.kuleuven.be/publications/article-927.pdf

[48] J. Ren and J. Wu, "Survey on anonymous communications in computer networks," Computer Communications, vol. 33, no. 4, pp. 420–431, Mar. 2010. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0140366409002989

[49] A. Ruiz-Martínez, "A survey on solutions and main free tools for privacy enhancing Web communications," Journal of Network and Computer Applications, vol. 35, no. 5, pp. 1473–1492, Sep. 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804512000665

[50] M. Reed, P. Syverson, and D. Goldschlag, "Anonymous connections and onion routing," IEEE Journal on Selected Areas in Communications, vol. 16, no. 4, pp. 482–494, May 1998. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=668972

[51] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Proceedings of the 13th conference on USENIX Security Symposium, 2004. [Online]. Available: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA465464

[52] B. Li, E. Erdin, M. Gunes, G. Bebis, and T. Shipley, "An overview of anonymity technology usage," Computer Communications, vol. 1, no. 775, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366413001096

[53] I. Clarke, O. Sandberg, B. Wiley, and T. Hong, "Freenet: A distributed anonymous information storage and retrieval system," Designing Privacy Enhancing Technologies, 2001. [Online]. Available: http://link.springer.com/chapter/10.1007/3-540-44702-4_4

[54] C. Shields and B. Levine, "A protocol for anonymous communication over the Internet," Proceedings of the 7th ACM conference on Computer and communications security, 2000. [Online]. Available: http://dl.acm.org/citation.cfm?id=352607

[55] C. a. Shue and M. Gupta, "Hiding in Plain Sight: Exploiting Broadcast for Practical Host Anonymity," 2010 43rd Hawaii International Conference on System Sciences, pp. 1–10, 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5428665

[56] M. Reiter and A. Rubin, "Crowds: Anonymity for web transactions," ACM Transactions on Information and System Security (TISSEC), 1998. [Online]. Available: http://dl.acm.org/citation.cfm?id=290168

[57] S. Rass, R. Wigoutschnigg, and P. Schartner, "Doubly-anonymous crowds: Using secret-sharing to achieve sender-and receiver-anonymity," Journal

of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, vol. 43, no. August, pp. 27–41, 2011. [Online]. Available: http://isyou.info/jowua/papers/jowua-v2n4-2.pdf

[58] D. Dittrich and S. Dietrich, "P2P as botnet command and control: a deeper insight," MALWARE 2008. 3rd International Conference on Malicious and Unwanted Software, 2008., no. June, 2008. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4690856

[59] G. Yan, D. T. Ha, and S. Eidenbenz, "AntBot: Anti-pollution peer-to-peer botnets," Computer Networks, vol. 55, no. 8, pp. 1941–1956, Jun. 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1389128611000533

[60] N. Tamaña, "Backdoor Uses Evernote as Command-and-Control Server," 2013. [Online]. Available: http://blog.trendmicro.com/trendlabs-security-intelligence/backdoor-uses-evernote-as-command-and-control-server/

[61] Fortinet, "One in Five SMB Retailers Not PCI Compliant, Lack Security Fundamentals," 2014. [Online]. Available: http://www.fortinet.nl/press_releases/2014/smb-retailers-survey-pci-compliant-lack-security-fundamentals.html

[62] Ipanema Technologies, "Killer Apps," 2012. [Online]. Available: http://www.ipanematech.com/library/WP_2013_Killler_Apps_Survey_Ipanema_Easynet_EN.pdf

[63] Z. Chi and Z. Zhao, "Detecting and blocking malicious traffic caused by irc protocol based botnets," pp. 485–489, Sept 2007. [Online]. Available: http://ieeexplore.ieee.org/iel5/4351441/4351442/04351531.pdf?arnumber=4351531

[64] Farnham, Greg, "Detecting DNS Tunneling," 2013. [Online]. Available: http://www.sans.org/reading-room/whitepapers/dns/detecting-dns-tunneling-34152

[65] Fridrich, Jessica and Lisonek, Petr and Soukal, David, "On steganographic embedding efficiency," 2006. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.83.5344

[66] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," pp. 183–195, 2005. [Online]. Available: https://www.cl.cam.ac.uk/~sjm217/papers/oakland05torta.pdf