Master of Science in Engineering: Computer Security
Juni 2022

Deanonymizing Onion Services by Introducing Packet Delay

Johannes Ödén
In cooperation with Axel Gehlin Björnberg

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Master of Science in Engineering: Computer Security. The thesis is equivalent to 20 weeks of full time studies.

The authors declare that they are the sole authors of this thesis and that they have not used any sources other than those listed in the bibliography and identified as references. They further declare that they have not submitted this thesis at any other institution to obtain a degree.

Contact Information:
Author(s):
Johannes Ödén
E-mail: joof16@student.bth.se

In cooperation with Axel Gehlin Björnberg
E-mail: axbj16@student.bth.se

University advisor:
Nurul Momen and Dragos Ilie
Department of Computer Science

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Background. Onion services facilitate two-way communication over the Tor network without letting either party know the other address or location. Many different techniques to break that anonymizing have come forth, but most of them have only been on paper. Some have been tested but then only on a separate network and not on the live Tor network.
Objectives. This thesis presents a technique that, with a minimal intrusion to the Tor network and no manipulation of the Introduction relay or the Rendezvous relay, can break the anonymizing of an Onion service.
Methods. The technique has been tested on the live Tor network with the approval of an ethics board. The Onion service anonymity was broken by having the Guard relay the Onion service used to connect to the Tor network introduce a watermark containing the IP4 address of the Onion service in the TCP packet's Request-Response Time (RRT). The TCP packets were used to transmit the watermark where an HTTP echo request was sent from a Tor client where the RRT was captured, and the watermark was decoded. In order to decode the watermark, the normal RRT of packets on the Tor network was needed, so to get the data, HTTP echo requests were also sent without the watermark.
Results. The watermark was decoded by the Tor client 88.80% of the time out of 607 tries.
Conclusions. While this technique was proven to work, what holds it back is the need for the Onion service to choose the Guard relay that introduces the watermark. The chance of a specific Guard relay is chosen depends on that relays history on the Tor network. However, it's usually about 0.005%, meaning it would need around, 20000 tries to break the anonymity of a random Onion service if only one Guard relay is used.

Keywords: Deanonymizing, Onion Service, Tor network, RRT

Sammanfattning

Bakgrund. Onion services upprättar tvåvägskommunikation över Tor-nätverket utan att låta någon av parterna veta den andres adress eller plats. Det har skapats många olika tekniker för att bryta den anonymiseringen har blivit skapta, men de flesta har bara funnits på papper. Vissa har testats men då bara på ett separat nätverk och inte på det riktiga Tor-nätverket.
Mål. Denna avhandling presenterar en teknik som, med ett minimalt intrång i Tor-nätverket och ingen manipulation av Introduction eller Rendezvous relayen, kan bryta anonymiseringen av en Onion-tjänst.
Metoder. Tekniken har testats på Tor-nätverket med godkännande av en etiknämnd. Onion services anonymitet bröts genom att Guard relay som Onion services använder för att ansluta till Tor-nätverket introducerade en vattenstämpel som innehåller IP4-adressen för Onion services i TCP-paketets Request-Response Time (RRT). TCP-paketen användes för att överföra vattenstämpeln där en HTTP-ekobegäran skickades från en Tor-klient där RRT fångades, och vattenstämpeln avkodades. För att avkoda vattenstämpeln behövdes den normala RRT av paket på Tor-nätverket, så för att få data skickades även HTTP-ekoförfrågningar utan vattenstämpeln.
Resultat. Vattenstämpeln avkodades av Tor-klienten 88,80% av 607 försök.
Slutsatser. Även om den här tekniken visade sig fungera, är det som håller den tillbaka behovet för Onion services att välja Guard relay som introducerar vattenstämpeln. Chansen att en specifikt Guard relay väljs beror på den relay historiken på Tor-nätverket. Men det är vanligtvis cirka 0,005%, vilket betyder att det skulle behövas runt, 20 000 försök att bryta anonymiteten för en slumpmässig Onion services om bara ett Guard relay används.

Acknowledgments

Our supervisors, Nurul Momen and Niklas Palmnert, immensely helped the progress of this thesis and have given advice and resources that have helped solve various problems. Tobias Pulls also provided helpful assistance in finding an appropriate area for introducing time delays and helping us understand how the Tor code functions. Per Hurtig gave useful advice on tools to use. Christoffer Toft gave valuable input regarding the code for the Tor Guard relay. A big thank you to Dragos Ilie for the final draft of the thesis.

Contents

Chapter 1

Introduction

1.1 Background

Here is a summary regarding what the Tor network is and how the Tor network works, as well as how Onion services use the Tor network. In addition, the motivation, goals and scope of the thesis will be described.

1.1.1 Tor Network

All communication protocols send information over the Internet, like Transport Control Protocol (TCP), uses IP addresses to specify the destination and the source. This makes it easy to send replies. This also makes it hard to hide one's IP address and one's identity. The onion router (Tor) network has many users who are activists, whistleblowers, and others who care about being anonymous [36] because it is assumed that the network will increase their anonymity online. It is quite important that the implementation of the network and the protocol it relies upon is as secure and robust as possible. Otherwise, revealed information about people on the network might leak, and the anonymity of Tor-users could be compromised. The Tor network has been and is today used by many simply because it has been very successful in hiding its users' identity compared to other solutions. However, that is not to say that it is impossible to uncover the anonymity of users on the network. Part of the attempt to make the Tor network robust is to have it distributed, which means not having a single control point. Without a centralized point there isn't one person or agency with the ability to control the whole network [36]. The current Tor network is based on the idea of "Onion routing", which was developed in the 1990s, where there should be layers of encryption lying on top of each other like layers of an onion [10].

The Tor network is built out of many volunteer nodes and anyone that wants to, can set up and submit to the network. Pre-allocated Tor directories keep track of available nodes on the network. The nodes, or relays as commonly referred to by the ones responsible for the Tor Project [37], are computers or routers that someone has submitted to the network to contribute to the decentralization of the network. When someone wants to browse the Tor network through a Tor browser, commonly known as a Tor client, the client connects first to one of the Tor directories on the network. This directory gives a list of available and recommended relays to use. After which, the client contacts three relays to send traffic through, from itself and to the wanted

endpoint for any network package. The three contacted relays are connected as a transmission chain, where the first relay (from the view of the Tor client) is known as a **guard relay**. The next relay in the chain is known as a **middle relay**, and the last relay, which in turn is also connected to the endpoint, is known as an **exit relay**.

When packages are sent through an established chain from a Tor client, the payload for an IP address outside the Tor network is encrypted three times, one time for each relay. Its corresponding relay deciphers each layer of encryption. This means that only the exit relay can see the payload and know the receiver's IP address, and the protection relay is thus the only one that knows the Tor client's IP address. When the recipient responds to the client, the process is reversed. Each relay encrypts and sends it as a payload to the next one in the chain. Finally, the client decrypts all layers to get an answer.

1.1.2 Onion services

Onion services or Hidden services are servers that can only be accessed through the Tor network by a so-called Onion address[34]. It is done so that neither the service nor the client will be able to identify each other. An Onion service is a server, which acts as a Tor client. The Onion then chooses and creates chains to several exit relays, which will become Introduction points. The Onion will then register to a Tor Directory as an Onion service and receive an Onion address, as shown in figure 1.1. Any Tor client connecting to the Tor Directory asking for the Onion service's Onion address will be given the address to one of the Introduction points. The client will then connect to one of the Introduction points through its chain. There it will establish a rendezvous point and a one-time secret string. The string will then be sent over to the Onion service. The last steps are for the client to terminate the connection to the Introduction point and for the Onion service to connect to the rendezvous point and send over the secret string as identification. The rendezvous point compares the two secret strings, and if they match, the connection between the client and server is created, and they can communicate without knowing each other's identity. This is shown in figure 1.2.

1.1.3 Watermarking

A watermark is a non-intrusive marker on a signal or a piece of data, that can either identify the owner or the validity of the signal or data. The watermark marker is usually embedded using a form of steganography, to only be visual during special conditions. A digital watermark found in audio, video or image data, for example, differs from metadata in that a watermark does not add extra data but creates a pattern in existing data so that the size of the file or the length of the signal file does not increase.

Different digital watermarks can be classified by four criteria: robustness, perceptibility, capacity, and embedding method [8]. Robustness is how much noise the marker can be exposed to before it will no longer be detectable as a marker or will lose a critical piece of information. Perceptibility is how visible the marker is in

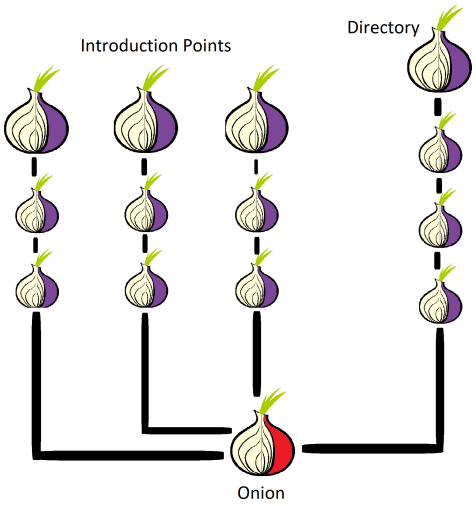

Figure 1.1: *The figure shows the onion service first choose three introduction points and then the contacts the Tor directory to receive an onion address.*

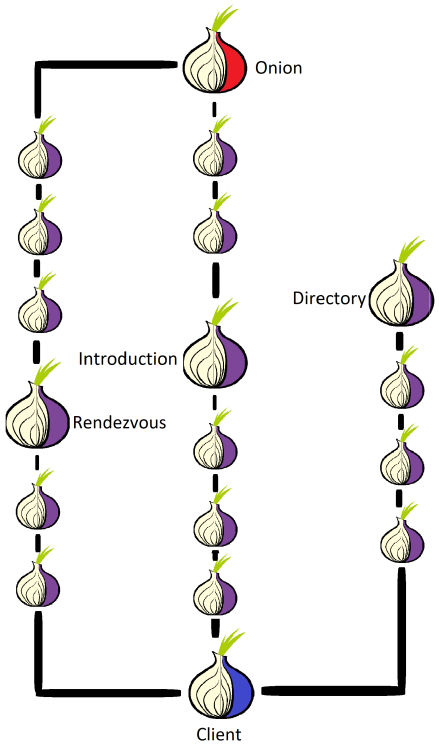

Figure 1.2: *The figure shows the Tor client sending an inquiry to the directory about a onion address. Getting redirected to the introduction point to start introduction with the onion service. Then choose a rendezvous where they will establish the complete conection.*

normal activity. Capacity is how much data can be stored in the marker, while the embedding method is just how the marker is placed in the signal or file and how it can be read to see the data embedded.

1.1.4 Measuring TCP Packets

In measuring the speed and timing of TCP packets, this study writes about three measurmants Request Response Time or Round Trip Time (RRT/RTT), Inter Packet Delays (IPDs), and Burst size [**?**]. RRT is the round trip for a packet to be sent, received, and then responded to. Usually measured by the device sending the packet. The burst size is the number of TCP packets sent containing the same content at the same time. IPDs are the time in between the arrival of two packets in the same Burst. As an example a node begins a TCP handshake by sending four request packets containing the same content. In between the first packet arriving and the second packet arriving is the first IPD. The second IPD is the time in between the arrival of the second packet and the arrival of the third packet, as shown in Figure 1.3. These measurements are usually measured using tools that can capture the packets as they enter and leave the device using e.g. Wireshark, while they can be measured using tools like pings. The tool Ping is commonly to inaccurate, since it measures more how fast connectivity is rather than the individual packets so do not measure Burst size, IPDs and measure RRT of a whole TCP handshake.

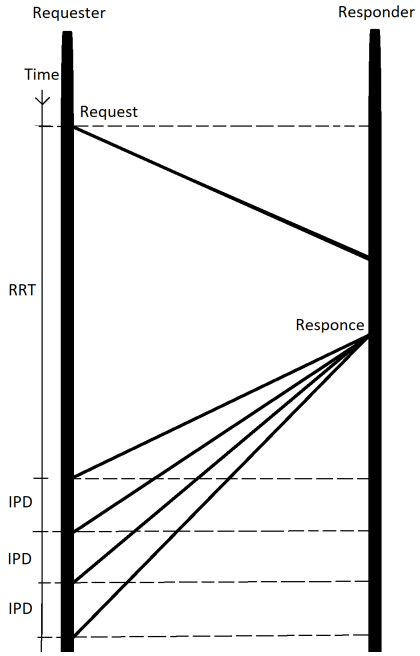

Figure 1.3: *The figure shows a TCP packet's RRT and IPD with a Burst Size of 4, when the packet travels between the requester and the responder.*

1.2 Defining the thesis

This thesis aims to find a way to break the anonymity of Onion services on Tor by lengthening the Request Response Time (RRT) between the Onion service and a Tor client. By lengthening RRT on the Onion service Guard relay a specific amount, a watermark containing the Onion router's IP address can be formed.

In order to make sure the watermark is readable without many false positives, the normal background noise in RRT must be measured so that the watermark is robust enough not to be misread. This is also required when figuring out how much detail in a watermark is detectable and by that know how much data there can be in each watermark.

1.2.1 Uses

As the Tor network is meant to ensure anonymity of its users, it is used by people who put secrecy and anonymity above other deciding factors for which network to use. Tor is used by, for example, journalists and freedom fighters to spread information [36] about the current state of affairs in the world and inform about human rights violations without them risking prosecution.

However, there are also a number of people who use the Tor network for illegal activities such as selling or buying drugs, weapons and more [3] precisely because the website offers anonymity. Therefore, it can also be important to break the anonymity of the users to catch those who break the law. There are different ways to break the anonymity of Onion services on the Tor network which is written in section 2.1 Other techniques. One of these ways of setting up controlled nodes and doing fingerprint analysis on network packets, like Biryukov et al.[10] describes. It is also possible to plant fingerprints in the packet protocol like Lin et al.[19].

For this thesis, we present a new approach to planting fingerprints with network packets that can break anonymity on the Tor network. This new approach is expected to be less detectable than many other solutions. This would make it possible to uncover illegal activity on the Tor network, without necessarily alerting whoever is responsible for the activity. Which would be useful in for example police investigations.

1.2.2 Aims and Objectives

With this thesis, the meaning is to find a better way to break the anonymity of Onion services on Tor. Therefore the aim is to try a new way to do this with fewer resources and in a real environment, not just in simulations like earlier work. In this thesis, fingerprints are planted in the packages by delaying the TCP packages. To examine this the following questions were created:

RQ_1: What is the measurable mean and standard deviation of RRT between a Tor client and an Onion service on the Tor network?

To answer this question, the mean and standard deviation are required to be measured since they give essential information regarding finding a pattern. In the pattern, the normal distribution of the background noise can be found and separated from the fingerprints. This information is crucial to answer $\mathbf{RQ_2}$ and $\mathbf{RQ_3}$. The mean and standard deviation are required because each formed watermark will rely

on the time delays between sent network packets on the Tor network. Though earlier works have done measurements on the Tor network, to find out how much of a fluctuation there is on the Tor network. The implementation of this thesis will still require an up-to-date analysis of the Tor traffic. This is because the usage of the Tor network changes over time, depending on how and when people use it over days, weeks, months, even years[35]. Moreover, the statistical analysis that previous works have done, has been on a smaller scale and not based on connections with Onion services[2]. Whereas, connections with Onion services could increase the possibility for increased fluctuations.

RQ_2: What amount of data can be encoded and sent with artificially introduced time delays in specific patterns?

To implement code to introduce anomalies in time delays, with which to encode and send data, the standard deviation in time delay of regular traffic on the Tor network needs to be calculated, so anomalies in relation to normal time delays can be detected correctly. If they are not detected the wrong address can be given or the background noise cannot be distinguished from the signals. This depends on the results of the previous question, and so, is not practical to gather from previous works for this thesis.

RQ_3: How can the watermark in **RQ_2** or the implementation producing the watermark be detected?

The expected outcome is to test a method to break anonymity by coding a fingerprint in the TCP packets containing the IP address of the Onion service and to know whether the technique is more or less efficient and practical than previous techniques. It is also expected to find the weaknesses in the technique. Knowledge of how-to and how easy it is to detect the technique is as well needed. The way to do this is to propose a new technique of discovering watermarks or find an already existing technique in related work and use what that technique uses as markers to discover watermarks and compare it to this thesis technique. The plan is also to be able to make suggestions for future work.

1.2.3 Scope and Limitations

This thesis is to create a watermark in the RRT of a connection between a Tor client and an Onion service that will tell the Tor client the Onion service's IP address.

The standard deviation in RRT on regular traffic on the Tor network needs to be calculated in order to implement code to be able to introduce anomalies in RRT to correctly detect anomalies with normal RRT. The detection depends on the results of the previous question, and so, is not practical to gather from previous works for this thesis.

This thesis will be limited to finding information on the RRT of packets in the Tor network, a method to introduce patterns with RRT in network packages. A way to encode information about which IP addresses a connected Tor client is connected from. To deanonymize that Tor client. This thesis will not propose any following actions based on any found and deanonymized clients from an implementation of the method in this thesis. Neither will this thesis cover a method to deanonymize middle or exit relays on the Tor network.

1.2.4 In Cooperation With

The experiment and the literature review were in cooperation with another master thesis student, Axel Gehlin Björnberg. He researched how the technique developed in this thesis could be used to deanonymize a regular Tor client while I researched how it could deanonymize an Onion service. Together we set up and used the same guard relays and tested the Tor network together. This is partly the reason this thesis has both data from clients and Onion services from the Tor network. In testing the technique to deanonymize the Onion service the same guard relays and script on the guard relays were used. However, the packets were delayed using a different formula and with a different frequency, than specified in Section 3.3.3, this was done due to the different RRT and the differing amount of packets between Tor clients and Onion services. Axel's thesis also used a different detection approach where only some packets like ACK packets. The testing of the technique was done without the input of each other separately as well as the two theses have conducted a separate analysis of the final results.

1.2.5 Thesis Outline

This thesis contains the following chapters and content:

- Chapter 1:
  - This chapter introduces the subject of this thesis. It includes an explanation for what the Tor network is, how it works and how the method from this thesis is related to it.
- Chapter 2:
  - This chapter describes other and earlier works that have used other methods or investigations of how digital watermarks can be introduced and used in relation to the Tor network to find out or transmit different data.
- Chapter 3:
  - The research questions for this thesis are presented up here, as well as the reasoning behind them. How data collection was planned and done, is also documented in this part. What tools and the reasoning behind the usage of those tools is also found in this section. In addition to this, risks and mitigations of risks are brought up in this section as well as the potential consequences of those risks.

- Chapter 4:
  - Here, results and what the presented data means are presented. The chapter also contains what conclusions can be drawn and what they mean. As well as if it seems that the method from this thesis could be effective in practice.
- Chapter 5:
  - Discussion of results and how conclusions can be interpreted also in the context of additional information. What are the positive and negative consequences of the method and a possible implementation of it.
- Chapter 6:
  - What conclusions that can be drawn from this thesis is presented in this part. What the conclusions mean in a broader context of the Tor network as a whole is brought up in this chapter. In addition to this, ideas for possible future plans are brought up and discussed.

Chapter 2

Related Work

2.1 Other techniques

Murdoch and Danezis's study [24] did a proof of concept on traffic-analysis attack. They showed how time delay could be detected through a Tor circuit with a single relay, where a normal Tor circuit is three relays. Thereby breaking the anonymity of a Tor client.

There are many kinds of attacks on the Tor network, but they can be sorted into three different categories in Tor's threat models [2], depending on how they attack the Tor network. The Global Adversary, when the attacker controls the whole or at least both guard and end nodes of a circuit [4], [23], [25]. Capture Entry Flow, where the traffic between an entry node and a Tor client is captured, used primarily by governments [1], [11], [12]. Compromising Tor Nodes is the last model and where the thesis will fall under. It is when a modified Tor Node is introduced to the Tor network [17], [22], [24], [31]. These can also be categorized as passive attacks if the attacker merely listens to packets on the network, or an active attack if an attacker actively send packets of their own.

The most similar to this thesis is the Inflow method as described by Iacovazzi and others [17]. They implant a Guard relay that will inject time gaps and dummy packets between packets in a download stream. The time gap and number of packets is indicated by a private key and a pseudo-random string which encrypts the IP address to be communicated. In comparison, for this thesis, only time gaps will be used to give information about a connected client's IP address. Extra packets will not be introduced into an established connection. Neither will any information be derived nor based on keys. This thesis will only be relying on measurements from the Tor network and introduced time delays based on previous measurements.

Another way to break anonymity on clients by compromising Tor nodes is to use Yanbing and Xiaolei's way of exploiting the Tor protocol of a controlled entry relay [22]. They used special cells while connecting with a client to confirm they use the controlled relay and signal the controlled relay to record the IP address and the packets.

Tan et al. [31] have a different approach than deanonymizing the Tor network. They used an Eclipse attack [27] to block Onion services with a small number of resources.

They used a security property of distributed hash table (DHT) to monopolize the DHT and block all other connections. They can do this because the descriptor-ids are predictable and HSDirs' watermarks can be calculated. They can then make the attack persistent, only the use with the continued use of a single IP-address.

The work written by Nurmi et al. goes through a few methods of breaking the anonymity of clients on the Tor network [26]. The method which is most similar to the work in this thesis, out of the ones they presented, is their method which leverages traffic and timing correlation attacks. Whereas, the method for this thesis is based on imposed time delay and not inserting more data into an already existing stream of data, or creating a new one.

The work of Chen et al. proposed a method of revealing guard relays that are connected to Onion services [5]. This is done by exploiting a design flaw that allows one to send messages from one node of the network to another. Meaning that one side of a connection can send information about what it knows to a gathering point. And the other side of the same connection can send what that one knows to the same gathering point. Effectively allowing for identification of a connection through tying together information about connections that are supposed to be held separated. This is in contrast to this thesis where information is sent through artificially imposed time delays in already existing traffic, depending on the expected static jitter in the connection over the Tor network.

2.2 Watermarking

Iacovazzi and Elovici divides watermarking into 4 parts, *Filtering, Encoding, Spreading* and *Embedding*[16].

Filtering is the process of filtering the different streams of data and diverting the stream that will carry the watermark, while *Encoding* is the technique used in encoding the watermark in the stream.

Spreading contains the need to make sure the watermark get through by dispersing and spreading the watermark. This can be done by *1) Time diversity* where the watermark is replicated at different times sending multiple examples of the same watermark. This is the most usual of spreading tactics and the one that will be used during this thesis, similarly to how Houmansadr et al. did their RAINBOW technique. Their RAINBOW technique, or scheme, is a watermarking technique where the watermark is put in IPDs of packages. This is then repeated several times. By analyzing the IPD first and from then put them in during moments of low amplitude, their markings are made small and thereby in practice invisible [13].

2) Frequency diversity: also referred to as DSSS (Direct Sequence Spread Spectrum), is the use of the spectrum outside the required bandwidth of the watermark signal. This is what Huang et al. did. They created a long pseudo-noise code (PN code) in order to track suspicious communications over anonymous sites on the Internet. The

benefits of the method are foremost its invisibility that comes by discarding regular patterns and self similarly from the watermarks [15]. It is because it's discarding of the regular patterns and thereby disrupting the normal flow that this method would not benefit this thesis and was therefore forsaken.

3) Space diversity: are when the watermark is spread over different channels. Houmansadr and Borisov for BOTMOSAIC created the first version with their watermark for detecting IRC botnets [14]. This method requires multiple detectors and was unsuited for our environment and thus disregarded.

Embedding is the part where the carrier to the watermark is chosen. The study found four categories of watermarking *Content-based, Timing-based, Size-based* and *Rate-based*. The current thesis falls under *Timing-based* where it will do a Simple delay. The other method mentioned in Iacovazzi and Elovici paper is a mean balancing method introduced by Peng et al.[28] where each part of the signal is divided between multiple carries and the mean is taken which reveal the watermark. It is more secure, however it requires larger spread for each watermark than the Simple delay method.

In Iacovazzi and Elovici paper the Simple delay method used the IPDs (InterPacket Delays) which is the time differentials of a burst och packets in response to a single request. The current thesis will rather test using the RRT instead.

What Iacovazzi and Elovici mention is important for watermarking is the invisibility and the robustness of a watermark. In order to test a watermark, at least one of these need to be tested. To test the visibility of watermarks, a method BACKLITE was proposed. BACKLITE used the RRT, IPD and burst size to calculate five streams of variables. These variables build on both the mean and the histogram of RRT, IPD and burst size[21].

Chapter 3

Method

3.1 Selected Methods

To answer the three research questions, two methods were selected: a literature review and an experiment. The experiment was used to answer RQ1 and RQ2, while RQ3 required both the literature review and the experiment to be fully answered. The experiment where necessary because previous works had investigated the background noise of the Tor network, however, due to the constant expansion and the changing network new data were needed to answer RQ1. For RQ2 the experiment was needed to test the created pattern on the live network, previous works had used simulations and not been tested on the live network. These data were then used to answer RQ3. RQ3 also required a literature review to find how weaknesses in other techniques and how to exploit similar weaknesses in deanonymization techniques.

3.2 the Literature review

The literature review was conducted in order to answer RQ3. The study was meant to find out what is known about potential vulnerabilities in the Tor network, and how others have tried to break the anonymity on the Tor network.

A literature mapping review was conducted into related works. The databases used were 'IEEE Xplore' [18] and 'SpringerLINK' [30] with the search terms "tor", "Onion services" and "Hidden services" within the last six years. The reason behind only using articles from the last six years where chosen is because it is six years ago Tor network began to truly grow in relays numbers [7]. The articles found were judged for relevance by their abstracts in regards to if they were usable or not. The snowballing technique was then used to find new interesting and related papers. The literature found was used as the core studies in a systematic mapping study.

3.3 The experiment

The goal was to gather information on the traffic of the Tor network to measure the background noise and to test the deanonymization technique.

The experiment had two phases: the information gathering phase and the testing phase. In the information-gathering phase, we set up Tor nodes and Tor clients to

measure the background noise of the RRT of the packets traveling through the tor network, and then we used the data to create a pattern we then could test. In the testing phase, the nodes were used to imprint the IP address in the TCP packets, how to optimize the pattern, and test how reliable the pattern was.

3.3.1 Tools

To be able to gather data for this thesis, a few different tools were used. **HTTPing** [39] was used in order to send echo requests through the Tor network's TCP traffic. **Torsocks** [6] sent data through the Tor network that would not have been feasible to send through the Tor-browser. **Tor** [38] was used to send the network traffic for this thesis onto the Tor network. We created Python scripts that detected introduced watermarks in network streams. Various shell scripts connected various programs needed for the thesis. Among them being the ones mentioned above.

HTTPing

HTTPing is a tool created to enable the use of Echo-requests over TCP connections [39]. Echo-requests are usually done with the ICMP (Internet Control Message Protocol). However, as traffic on the Tor network is only in the form of TCP packets, this tool worked better for this work. The version for HTTPing was httping-1.4.4 during this thesis.

Torsocks

Torsocks is the tool that was used to send TCP-based traffic onto the TOR network. It takes whatever TCP data is sent into it and sends it out on the Tor network with whatever Tor client is available locally on the running machine. For this thesis, this tool was used to send HTTPing requests over the Tor network.

Tor client

A Tor client is an instance of the Tor program that has been used as a proxy for other programs. This is the common use case for the Tor program if not given a specific configuration to act as a node for the Tor network. The version for Tor client was Tor-0.4.2.7 during this thesis.

IPtables

IPtables is the tool used to set up rules on what should be done with various network packages that enter or leave the computer that IPtables is used on. These rules are based on, for example, if a network package comes from a specific source or is meant for some specific destination. For this thesis, IPtables was used to redirect network packages that come from a specific source into the program that has been used to introduce watermarks.

Wireshark

Wireshark is a program that is efficient and often used for analyzing network traffic and specific network packages. This allows a user to inspect network packets and the relationship between them in identified network streams of packets. Wireshark was used in this thesis to analyze network traffic at the beginning experiment of this thesis. The version for this tool was Wireshark-3.2.3 during this thesis.

Tshark

Tshark is the same as Wireshark but without a graphical interface. It is useful for automizing the analysis of network packages and combining it with other programs. In this thesis, Tshark was combined with Python to analyze network streams and find introduced watermarks in those streams. The version for this the same version as for Wireshark, version 3.2.3.

Python

Python is a programming language that can be used for whatever one wants to implement. In this thesis, Python was used to use Tshark to analyze network traffic from Tor and introduce watermarks were supposed to. The version for Python was 3.8.10 during this thesis.

Dash

Dash is a shell that is often used to run shell scripts as it is simple in its implementation. Dash was used during this thesis in order to run shell scripts that were needed. The version for this tool was 5.8-2.1 during this thesis.

3.3.2 Data gathering

In order to calculate the mean and standard deviation of RRT and test the clarity of the watermark, there needs to be an infrastructure that both supports and protects the data of individuals as stated in section 3.3.4. This was done by setting up both the client that will send requests and the Onion server that will send the responses.

In order to also check our infrastructure to a regular Onion service, a known Onion service was used for a control sample. The known Onion service used was DuckDuckGo's Onion service "3g2upl4pq6kufc4m.Onion". This was used to show whether the data collected from our infrastructure was representative of normal Tor traffic.

In order for the infrastructure to be used to test the watermark, it had to contain the guard relays which later would contain the code for introducing the watermark to make sure that if the guard relays added any interference, it would be picked up in the data. As shown in figure 3.1.

Four guard relays were set up. With the service Linode [20] which offers cloud computing services, we set up two isolated and separated guard relays. Both relays

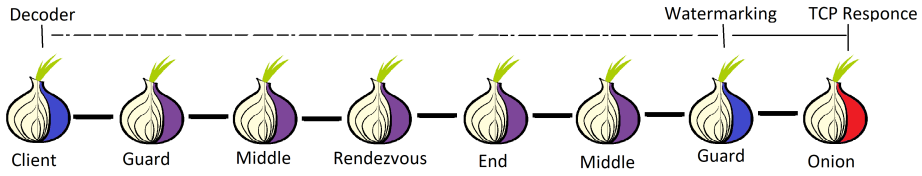

Figure 3.1: *The figure shows the complete connection between the client and the onion service for when testing the watermark, with the Guard relay on the onion service side watermarking the TCP packets.*

were set up in Germany to get a geographic difference from the other two relays in Sweden on a Raspberry Pi four and a desktop, both running a Linux operating system.

The Tor client which sent the requests was also set up with the service Linode in Germany, while the Onion service was set up in Sweden on a Raspberry Pi, which was using the Raspberry Pi OS [32].

The Tor client was set up first by downloading the Tor project from their GitHub project [38] and installing it by following the instruction on the README text file. The Tor guard relays were installed the same way using the same code. However, a Tor relay is created by going to the "torrc" file usually located at "/etc/tor/", there all instructions and settings options are located for a change from Tor client to a Tor relay. The settings we chose were to change the ORPort to between port 9002-9007 for the different relays.

The setup for the Onion service is the same as the setup for the relays except for the settings in the "torrc" file. In the "torrc" file, we chose a file address where a web server needed to be and files specifically for the Onion service could be placed. The web server we used was a default Apache2 web server. Apache2 was chosen for it had a website that could respond to the HTTP protocol. In the Onion services "torrc" file, we also added the command "EntryNode <node-id>, <node-id>" which forces the Onion service to use specific guard relays as the initial point of entry to the Tor network. The "fingerprint" are from the four guard relays set up before, their ids were collected from the Tor metrics page [35].

In order to get consistent data over many weeks, a script was written in Shell-scripts, which would first send 100 HTTP echo requests using HTTping through the Tor network with the command "torsocks HTTping -r $ADDRESS -c 100 -f", 100 echo request per connection were chosen due to time restraint. The address first used would be of DuckDuckGo Onion service, and then the command used again with our own Onion service. The RRT collected from these 200 requests would then be saved in two files. The commands repeat in the script 50 times, 50 time was chosen in conjuntion with before due to time restraint. The script was placed in the Tor client placed in Germany, where a schedule was placed to activate the script every 6 hours. The script was allowed to run for around two months.

3.3.3 Testing

The script for introducing the watermark was written in python. The script listened to port 9002, which is the port the IPtable sends all packets that were sent from the IP address used by our Onion service. There, the script picked up all the packets and created a socket for each service and sent packets. Here, the IP address from the sender of the packets was collected. The packets received were then copied and sent to port 9050, which is the port the Tor guard relay listens for packets. However, every tenth packet sent through the script was held while the sleep command was activated. In order to determine the length of the sleep, a formula was needed, which was compact enough to not disturb the normal flow and big enough to be discovered by the detector. The formula developed was '$nr * 1.5 + stdev * 2 + avg$', where the 'stdev' is the standard deviation from the average 'avg' of an HTTP request-response over the Tor network and the 'nr' is the current number of the IP address in a cycle. A cycle is the going through a IP-address from the sender of a packet in order to encode the IP adress. The numbers 0-9 are the numbers in an IP4 and therefore the number 10 is to mark the end and the beginning between two cycles. The standard deviation in the formula was multiplied by 2 in ordet to make the result bigger than 90% the packages. 1.5 were added to give the 'nr' added difference from each other.

The client-side ran the same script as the client when collecting the data earlier in the thesis. After running, it starts a python script that takes the collected data and removes the average of regular HTTP echo requests calculated from the collected data. Then starts to take every step of the formula used by the Guard relay. First, taking the data minus the average time two, then dividing the result by 1.5. Most of the background static in the RRT is removed no, and that leaves the watermark. The python script then takes what it thinks is the IP.

3.3.4 Ethics

In Sweden, Etikprövningsmyndigheten is responsible for evaluating how proposed research treats personal and sensitive data. Personal data is information that can be tied to a living person. Examples of this would be a personal identification number, phone number, first and surnames, email address, home address. IP addresses are many times considered part of the home address for users on the internet. Examples of sensitive data would be sexual orientation or current sex-life, race or ethnic background, political opinions, religious or philosophical beliefs, membership status in unions, data regarding physical and mental health, genetical and bio-metrical information, as defined by Etikprövningsmyndigheten [9].

During this thesis, the traffic needed from the Tor network will be created by the thesis owner's own and controlled Tor client and guard relays. The information which would be looked at in the created Tor traffic is the time delay between arriving network packets. The Tor traffic will be filtered such that only traffic originating from the thesis owner's IP address is used for this research. Traffic originating from other's

IP addresses will not be collected. Etikprövningsmyndigheten states that student's research thesis on the master level are not covered by etikprövningslagen. However, it is still important for this research to be done ethically. Steps have been taken to limit data collection, and data that is not necessary is not collected. What is done is explained in section 5.4 Mitigation of risks.

The main ethical dilemma for this thesis could be if an organization or state actor implements this solution on a large scale and deploys a lot of modified guard nodes. This could break anonymity for targeted Onion services and anonymity for all clients and Onion services that connect to these guard nodes. As such, all who could detect these watermarks would be able to break the anonymity of Onion services and clients on the Tor network. This means that if a Tor client is used to hide the identity of a person, which could be a target of persecution on religious or sexual grounds, and that client would be caught in an investigation. That client and the user could have their anonymity broken without it being the original intent of the organization that set up the guard nodes. In order to make it harder for outside actors to detect the watermark, the guard nodes could be modified such that they only send the "fingerprint" when a signal is given from a controlled client by the ones that first put up the controlled guard node. That way, the ones that make the deanonymization possible can at the very least ensure that there are no watermarks to detect when they are not doing an investigation. A signal to activate a guard node could be as simple as five packets in a row with a predefined delay.

Tor developers may use the results from this thesis to patch away the technique used in this thesis. It could make it harder for law enforcement to track down illegal activities on the platform which is at odds with the purpose of this thesis. In order to combat this, new technique will be required when Tor developers start patching. Until then this technique will function.

The technique in this thesis could be used by a state actor specifically to persecute people trying to uphold human rights or report on conflicts regarding human rights. This is not what the technique is meant to do. Any such use of the technique is not supported by the thesis owners. The purpose of the thesis is to provide means to deanonymize criminals, as defined by the Swedish law [29].

Chapter 4

Results and Analysis

4.1 The Tor Network

Table 4.1 shows which type of connection was made (Circuit), how many times it was made (Nr of Datapoints), the average mean of the connections (Mean), and the standard deviation for them (Stdev).

Table 4.1: *The table shows RRT of each circuit tested. The 'Mean' is the average RRT of the circuit, and the 'Stdev' is the standard deviation of RRT from the average RRT. Each circuit is the communication between two points over the Tor network. 'Owned webserver' is the circuit between our webserver and our Tor client. 'Google' is the cirkuit between a Google webserver and our Tor client. 'DuckDuckGo Onion' is the circuit between our Tor client and DuckDuckGos Onion service. 'Owned Onion' is the circuit between our Tor client and Our Onion service.*

Circuit	Mean	Stdev	Nr of Datapoints
Owned webserver	363.01	448.92	661300
Google	300.09	391.16	783459
DuckDuckGo Onion	1328.50	2890.84	777391
Owned Onion	866.29	769.65	612311

As seen in the table 4.1, a request sent with HTTP request over the Tor network varies from connection to connection. However, it can generally be said that a normal connection would result in a mean RRT of 300 milliseconds for connections made to google.se over the Tor network, with a standard deviation of 391 milliseconds. With a web server that had been set up for this thesis, the same tests were done to get the results seen for "Owned webserver" in table 4.1, with a mean RRT of about 363 milliseconds and a standard deviation of about 449 milliseconds.

In contrast to the two previous tests, the two rounds of tests towards an Onion server that was set up for this thesis were not as similar to each other as the previous two were. As seen in table 4.1, tests towards DuckDuckGo's onion service resulted in a mean RRT of about 1329 milliseconds as well as a standard deviation of 2891 milliseconds. However, the setup onion service got a much lower result, with a mean of about 866 milliseconds and a standard deviation of about 770 milliseconds. Requests are made over a connection with an Onion service, which means that the total length of the established connection is double as long. This seems to correlate with

the result from the 'Owned Onion' circuit.

Figure 4.1 shows RRT measurements on HTTP requests done over the Tor network. The figure is a histogram where the average RRT of a circuit on the Tor network with a normal connection from a Tor client to a normal server. Figure 4.1 visualizes the relationship of the 'Mean' and the 'Stdev' from table 4.1. What figure 4.1 also shows is that there are not many RRTs over 1000 milliseconds or 1 second mark. Figure 4.2 shows the same for the circuit between an Onion service and a Tor client. However, the mark has moved from one to two seconds.

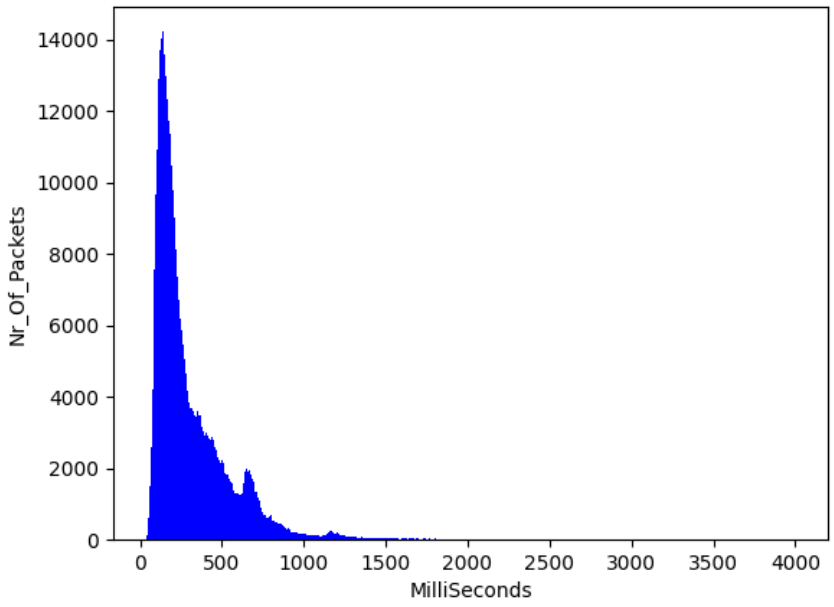

Figure 4.1: *A histogram of how many RRTs are the same collected from a Tor client sending echo requests to a normal server. The data is from both the 'Owned webserver' and the 'Google' circuit.*

When analyzing the data collected no variation can be found in the RRT sent over a Tor connection or a Onion service connection in regards to the time of day or the day of the week. This most likely has to do with that with the relays chosen at random and Tor relays that are spread over the entire globe so no matter where the client are the packets are most like sent all over the globe.

4.2 Watermarking accuracy

From testing the formula '$nr * 1.5 + stdev * 2 + avg$' with the standard deviation (stdev) and the average (avg) given in the section 4.1 from the 'Owned Onion' circuit. The number sent (nr) was correctly decoded 88.80% of the time from a data set of 607 numbers. However, the problem with this is that an IP contains 12 numbers.

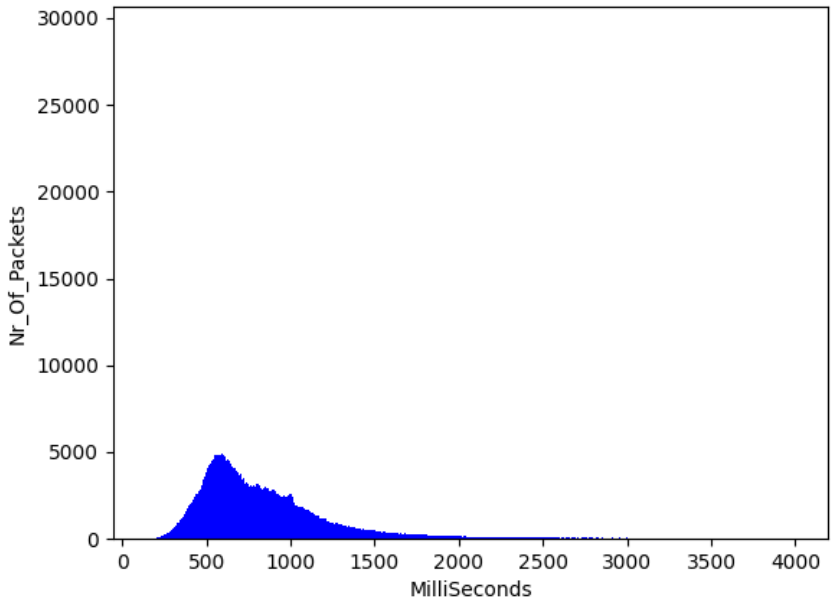

Figure 4.2: *A histogram of how many RRTs are the same collected from a Tor client sending echo requests to an Onion service. The data is from both the 'DuckDuckGo Onion' and the 'Owned Onion' circuit.*

For the IP address to be detected correctly, 12 numbers in a row need to be detected correctly. Using the same dataset as before gives 50 sets of 12 numbers, which are the numbers needed to create an IP address, and out of those, 60% was correctly decoded. Still the majority, but not accurate.

Nevertheless, because each number is sent separately and the IP address 12 numbers could repeat, the whole IP address does not need to be analyzed together. Only the IP addresses numbers positions needs to be together. In the table 4.2 each column is a position on the IP address, and the numbers are in order of entry. By taking the median of each column, the IP address is shown clearly if the accuracy of each number is above 50%. By having a script that collects and decodes a certain amount of numbers and calculates the median of each column, that script can then be run a controlled number of times, basically controlling the accuracy of the watermark depending on how many numbers decoded each time and how many times the script was run.

Table 4.2: *The table should be read horizontally and each time number 10 appears the detector jumps down one row. Each column shows a position of an IP-address and each number are decoded from the dataset of 607 numbers. To make the columns more visible, lines were drawn between them. The medium of each column, with an accuracy over 50% the medium will show the correct number for an IP address. Here the median for each column is 037003050042 which is the correct IP adress.*

0	0	0	3	7	0	0	3	0	5	0	0
1	3	7	0	0	3	0	5	0	0	4	2
0	3	7	0	0	4	0	5	0	0	4	2
0	3	7	0	0	3	0	5	0	0	4	2
0	3	7	0	0	3	0	5	0	0	4	2
0	3	7	0	0	3	0	5	0	0	4	3
0	3	7	0	0	3	0	5	0	0	4	2
0	3	7	0	0	3	0	5	0	0	4	2
0	0	0	0	0	3	0	5	0	0	5	3
0	3	7	0	0	3	0	5	0	0	4	2
0	3	7	0	0	3	0	5	0	0	4	2

Chapter 5

Discussion

5.1 Possibility and consequences

The results of this thesis shows that it is indeed possible to uncover the IPv4 addresses of clients on the Tor network. In this case, specifically, through the use of introduced watermarks in Tor network traffic. It has also been shown that this can be done relatively accurately with the technique in this thesis.

It is possible to attempt to target onions services with the technique. This is due to the fact that there are a limited number of guard relays, which means that every time a connection is done with the targeted onion service, the service picks a random Tor guard relay. If this guard relay is the one being controlled by the thesis, then it can break the anonymity, otherwise the thesis can break off the connection with the onion service and re-establish it. The onion service will then choose another random guard relay. This would take many attempts as there are many guard relays. Every Tor guard relay has around a 0.005%, chance of getting chosen, the information taken from Tor metrics[35]. The chance to get chosen will grow and shrink depending on how reliable that relay is. However, it is very much possible, as every new connection needs a new guard relay. If one standard guard relay is set up with the watermarking code on, then a script targeting a specific Onion service has a 0.005% chance to see the Onion service IPv4 address every time it is connected. Run the script on a 20000 loop, and it should be able to find the IP address of any Onion service[35].

This brings up the possible problems with the technique for this thesis. The technique of uncovering the IP address of Onion services from this thesis is not meant to be used to uncover journalists, freedom fighters, or people that would be subject to harassment or worse due to political, ethnic, religious, or sexual affiliation. The technique is only supposed to be used to uncover the IP addresses of criminals, as described in section 3.3.4. With that in mind, it is, however, very hard if not impossible to ensure that someone would not be able to use the technique for ill intentions.

As such, the implications that the results of this thesis bring are not to be taken lightly. It is possible to uncover sensitive information about connected clients on the Tor network. Any implementation based on the technique should be limited in scope and usage. Any steps possible should be taken to ensure that only sought-after clients should be uncovered. This would, however, not solve the issue of whether someone would use the implementation for uncovering journalists or others who ought to be

protected. This issue is not easy to solve, if not even impossible, to solve. As mentioned in section 3.3.4, it would simply be possible not to publish this thesis and, as such, not make another way to uncover the client's identity.

However, it is also the case that the process of making any system secure and robust is to point out the flaws that it has. Without knowing the flaws of a system, it cannot be improved. As such, the knowledge of this technique is deemed necessary enough to make known, even if it could bring some consequences as well. The integrity of Tor users, and specifically those who are not supposed to be uncovered. The way to insure security is to continuously update and investigate into new techniques.

5.2 Measurements of response-time

As presented in section 4, the RRT of various connections on the Tor network seems to be different from each other in terms of the time it takes to get a response to an HTTP request sent on an established connection. With regular client connections on the Tor network, the RRT seems to be much shorter than the RRTs for connections made to onion services.

This is probably due to several factors, one being that connections to onion services have to go through two Tor connections rather than just one. If nothing else, this would potentially result in an increase to double the RRT as compared to a normal connection just from the increased number of hops each network package has to make. The additional increase in RRT could be due to it being more tricky to look up where the end destination is in the Tor network.

5.3 Detection

As discussed in section 2.2. Measurements on packets' behavior that can detect watermarks are RRT, IPD, and burst size.

Iacovazzi and Elovicis [16] proposed a way of detecting watermarked packets was BACKLITE. BACKLITE was not tested against the watermark proposed. However, some predictions can be made on if BACKLITE would be able to detect the watermark proposed. The watermark in this paper does not use or change the burst size of each request. The packets used in this paper are either for the three-way handshake of the TCP protocol or simple request responses over the HTTP protocol, which do not require multiple responses for each request. This watermark should make IPD and burst size useless as markers. Even if some form of packet failer produces more requests for responses, it would not be able to use it as a marker because it would be too irregular for when it happens.

RRT, however, should be detected easily by BACKLITE. There are ways BACKLITE would not be able to detect these watermarks. They rely on the fact that BACKLITE requires example data to be able to judge a network. This example

data is usually collected by having BACKLITE listen to the normal flow of the network and use that as the norm. If BACKLITE instead would listen to the network while watermarking was in progress, then it would contaminate the example data and therefore BACKLITE could consider the watermark as normal activity. The other way to remain undetected would be by having the script on the guard relay listen to the packets sent to the guard relay and only have it activate when the packets have some sort of activation signal. Then the script would only be active for a short while. This would make it harder for BACKLITE to detect the watermark in the first place.

5.4 Validity

There are a few risks in this thesis. Some of them could make the process of conducting the thesis more time-consuming. Other risks could jeopardize the thesis as a whole.

The risk that could make the thesis take longer is that a Tor relay could take longer than expected to be recommended as a guard relay by existing Tor directories than anticipated by the recommended specifications [33]. Another risk that could make the thesis take longer is that if a Tor relay loses its guard status, it will take longer to either regain it or set up a new guard relay.

To reduce the risk that relays not being recommended and that relay guard status drops, several guard relays were deployed and with higher specifications than recommended. They would get recommended sooner than otherwise, and if one were lost, a backup could be used while a new one was set up.

The risk of having a relay drop, there is always a chance that the infrastructure around the deployed Tor relays fails. For example, the computer that hosts a Tor relay crashes for whichever reason. The electricity in the house or city/town the computer is in gets cut off. To mitigate this or other risks regarding access and possible up-time of deployed Tor relays, it was planned to have relays in different geological locations. So that if the power or internet connection goes down for one relay, other ones can still be used for the thesis.

Another risk that could jeopardize the thesis are the risks that invalidate the data. While taking measurements in network traffic, there is always a risk of exceptions affecting the results of measurements, depending on when the measurements were taken. In order to make sure this did not affect the data, several measurements were collected over a longer period as described in section 3.3.2 Gather data for this thesis. This will decrease the risk of one or a few exceptions giving a wrong picture of the entire set on the Tor network. It will at the same time also give a better picture of the Tor network over a longer period than otherwise. Another risk would be that a relay that introduces time delay would be flagged by a Tor directory as an unhealthy relay and therefore not recommend its use. This can be mitigated by either decreasing the time delay or the frequency of artificially introduced time delay. Or allowing the

relay to look for an activation signal to begin the transmission through time delay only when the connection is established and the watermark needs to be introduced.

A separate but still critical risk is the leaking of private data, defined by Etikprövningsmyndigheten[9]. This is mitigated by designing the collecting of data and the testing of the technique using only our own setup endpoints and guard relays. This means that only our own IP addresses are revealed and sent over the Tor network. This leaves, however, a risk that a Tor client still connects to the guard relay setup for this thesis. This is why all guard relays setup had a whitelist that only allows our own IP addresses. The whitelist was made by setting a rule in Iptables that took all packets with IP addresses we used and redirected them to the thesis code on that guard relay. All other packets were sent through the guard relay without interference. This method were approved by Etikkommittèn Sydost.

Chapter 6

Conclusions and Future Work

6.1 Thesis questions

For this thesis, three thesis questions were posed. During the thesis, attempts have been made to answer those questions.

RQ_1: *What is the measurable mean and standard deviation of RRT between a Tor client and an Onion service on the Tor network?*
The first posed the question of the measurable mean and standard deviation of RTT between a Tor client and an Onion service on the Tor network. The results from many rerun tests indicate that the mean RRT between a client and an Onion service is around 1000 milliseconds, give or take a few hundred milliseconds.

RQ_2: *What amount of data can be encoded and sent with artificially introduced time delays in specific patterns?*
The second question was about how much data could be encoded and sent with the help of introduced watermarks. The answer to this would be that the IP address of a connected Onion service can be encoded and sent with this thesis's implementation. However, theoretically, any letter could be encoded and sent as well. Giving the option of sending much more complex and more information about connected Onion services. The only limit would be the required time to send all the information. If more data were collected, a more compact pattern could possibly be form containing more data, but it would not help in detecting the watermark.

RQ_3: *How can the watermark in* **RQ_2** *or the implementation producing the watermark be detected?*
The third question poses how the used watermark, or the controlled node that introduces the watermark, could be detected. This could be done by analyzing the time-delays of packets in Tor connections. If a connection has a very predictable time delay for the most part but now begins to have sudden spikes in delay at repeated intervals. Then that connection might very well have a controlled guard relay, based on the method used.

6.2 Conclusion

It is possible to find the IP address of connected clients. The thesis method of doing so, and the Tor network's characteristics that make this possible, needs to be taken into account with future implementations of Tor.

Until the thesis method can be mitigated, it is possible to uncover IP addresses. Therefore it is more uncertainty regarding users anonymity on the Tor network now than before.

6.3 Future work

For future work, it would be possible to create a program that tries to create new connections on the Tor network until a specific and sought-after Onions service is found. Though, such a future work would bring along many ethical issues as it would reveal a lot of Tor users while searching for the right one. It would be better to see if there is a way to target specific Onion services without having to create new connections over and over again a couple of thousand times, if not even more.

What would also work as future work is to find another way to introduce watermarks or encode more information in each watermark. Another future work that could be done would be to test which detection methods of discovering watermarks would detect the ones introduced by this method. Among the detection methods, BACKLITE would be a suitable contender, as it has proved to be effective at finding other kinds of watermarks in traffic on the Tor network.

Bibliography

[1] D. Arp, F. Yamaguchi, and K. Rieck, "Torben: A practical side-channel attack for deanonymizing tor communication," ser. ASIA CCS '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 597–602. [Online]. Available: https://doi.org/10.1145/2714576.2714627

[2] L. Basyoni, N. Fetais, A. Erbad, A. Mohamed, and M. Guizani, "Traffic analysis attacks on tor: A survey," in *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, 2020, pp. 183–188.

[3] A. Biryukov, I. Pustogarov, F. Thill, and R. Weinmann, "Content and popularity analysis of tor hidden services," in *2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2014, pp. 188–193.

[4] S. Chakravarty, M. V. Barbera, G. Portokalidis, M. Polychronakis, and A. D. Keromytis, "On the effectiveness of traffic analysis against anonymity networks using flow records," in *Passive and Active Measurement*, M. Faloutsos and A. Kuzmanovic, Eds. Cham: Springer International Publishing, 2014, pp. 247–257.

[5] M. Chen, X. Wang, T. Liu, J. Shi, Z. Yin, and B. Fang, "Signalcookie: Discovering guard relays of hidden services in parallel," in *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019, pp. 1–7.

[6] S. Clowes. (accessed: 03.08.2021) torsocks. [Online]. Available: https://linux.die.net/man/8/torsocks

[7] I. D. Cooper, "What is a "mapping study?"," in *Journal of the Medical Library Association*, 2016, p. 76–78. [Online]. Available: https://doi.org/10.3163/1536-5050.104.1.013

[8] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital watermarking and steganography*, 2nd ed. Burlington, MA, USA, 2008.

[9] Etikprövningsmyndigheten. (accessed: 03.06.2021) Etikprövningsmyndigheten. [Online]. Available: https://etikprovningsmyndigheten.se/

[10] B. Falex, P. Ivan, and W. Ralf-Philipp, "Trawling for tor hidden services: Detection, measurement, deanonymization," vol. IEEE Symposium on Security and Privacy, pp. 80–94, 2013.

[11] Y. Gilad and A. Herzberg, "Spying in the dark: Tcp and tor traffic analysis," in *Privacy Enhancing Technologies*, S. Fischer-Hübner and M. Wright, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 100–119.

[12] G. He, M. Yang, J. Luo, and X. Gu, "Inferring application type information from tor encrypted traffic," in *2014 Second International Conference on Advanced*

Cloud and Big Data, 2014, pp. 220–227.

[13] A. Houmansadr, N. Kiyav, and N. Borisov, "Rainbow:a robust and invisible non-blind watermark for network flows," in *Proc. NDSS*, 2019.

[14] A. Houmansadr and N. Borisov, "Botmosaic: Collaborative network watermark for the detection of irc-based botnets," *Journal of Systems and Software*, vol. 86, no. 3, pp. 707–715, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121212003068

[15] J. Huang, X. Pan, X. Fu, and J. Wang, "Long pn code based dsss watermarking," in *2011 Proceedings IEEE INFOCOM*, 2011, pp. 2426–2434.

[16] A. Iacovazzi and Y. Elovici, "Network flow watermarking: A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 1, pp. 512–530, 2017.

[17] A. Iacovazzi, S. Sarda, and Y. Elovici, "Inflow: Inverse network flow watermarking for detecting hidden servers," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 747–755.

[18] IEEE. (accessed: 02.04.2021) Ieee explore. [Online]. Available: https://ieeexplore-ieee-org.miman.bib.bth.se/search/advanced

[19] J. Lin, J. Gao, Z. Wu, C. Si, and B. Sun, "Deanonymizing tor in a stealthy way," in *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*, 2019, pp. 1–8.

[20] Linode, LLC. (accessed: 03.08.2021) Linode. [Online]. Available: https://www.linode.com/

[21] X. Luo, P. Zhou, J. Zhang, R. Perdisci, W. Lee, and R. K. C. Chang, "Exposing invisible timing-based traffic watermarks with backlit," in *Proceedings of the 27th Annual Computer Security Applications Conference*, ser. ACSAC '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 197–206. [Online]. Available: https://doi-org.miman.bib.bth.se/10.1145/2076732.2076760

[22] Y. Ma and X. Xu, "Locating tor's hidden service clients based on protocol feature," in *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 2017, pp. 282–285.

[23] Ming Song, Gang Xiong, Zhenzhen Li, Junrui Peng, and Li Guo, "A deanonymize attack method based on traffic analysis," in *2013 8th International Conference on Communications and Networking in China (CHINACOM)*, 2013, pp. 455–460.

[24] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *2005 IEEE Symposium on Security and Privacy (S P'05)*, 2005, pp. 183–195.

[25] M. Nasr, A. Bahramali, and A. Houmansadr, "Deepcorr: Strong flow correlation attacks on tor using deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1962–1976. [Online]. Available: https://doi.org/10.1145/3243734.3243824

[26] J. Nurmi and M. S. Niemelä, "Tor de-anonymisation techniques," in *Network and System Security*, Z. Yan, R. Molva, W. Mazurczyk, and R. Kantola, Eds.

Cham: Springer International Publishing, 2017, pp. 657–671.

[27] OWASP. (accessed: 03.06.2021) Eclipse-tutorial. [Online]. Available: https://github.com/find-sec-bugs/find-sec-bugs/wiki/Eclipse-Tutorial

[28] Pai Peng, Peng Ning, D. S. Reeves, and Xinyuan Wang, "Active timing-based correlation of perturbed traffic flows with chaff packets," in *25th IEEE International Conference on Distributed Computing Systems Workshops*, 2005, pp. 107–113.

[29] S. rikstad. (accessed: 03.06.2021) Dokument lagar. [Online]. Available: https://www.riksdagen.se/sv/dokument-lagar/

[30] Springer. (accessed: 02.04.2021) Springerlink. [Online]. Available: https://link-springer-com.miman.bib.bth.se/

[31] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian, "Toward a comprehensive insight into the eclipse attacks of tor hidden services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1584–1593, 2019.

[32] The Raspberry Pi Foundation. (accessed: 03.08.2021) Raspberry pi. [Online]. Available: https://www.raspberrypi.org/software/operating-systems/

[33] The Tor Project. relays-requirements. [Online]. Available: https://community.torproject.org/relay/relays-requirements/

[34] ——. (accessed: 03.08.2021) Onion services. [Online]. Available: https://community.torproject.org/onion-services/

[35] ——. (accessed: 03.08.2021) Tor metrics. [Online]. Available: https://metrics.torproject.org

[36] ——. (accessed: 03.08.2021) Tor project. [Online]. Available: https://community.torproject.org/

[37] ——. (accessed: 03.08.2021) Tor relays. [Online]. Available: \https://2019.www.torproject.org/docs/faq.html.en#MostNeededRelayType

[38] torproject. (accessed: 03.08.2021) tor. [Online]. Available: https://github.com/torproject/tor

[39] Vanheusden.com. (accessed: 03.08.2021) Httping. [Online]. Available: https://www.vanheusden.com/httping/

Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden